Report on

"Detection of phishing websites using machine learning techniques"

Submitted for the requirement of

Project course

BACHELOR OF ENGINEERING

COMPUTER SCIENCE & ENGINEERING

Submitted to:

Surabhi Kaul(E8527)
(Supervisor)

Submitted By:

ALOK RAJ - 20BCS4883
Aditya Rana- 20BCS5643
Ashish Kumar – 20BCS5317
Gautam Aggrawal-20BCS2292
Dhruv Malhotra- 20BCS7135

Himanshu Sharma(E13027)
(Co-Supervisor)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
CHANDIGARH UNIVERSITY, GHARUAN
NOVEMBER 2022

ABSTRACT

With raising in-depth amalgamation of the Internet and social life, the Internet is looking differently at how people are learning and working, meanwhile opening us to growing serious security attacks. The ways to recognize various network threats, specifically attacks not seen before, is a primary issue that needs to be looked into immediately. The aim of phishing site URLs is to collect the private information like user's identity, passwords and online money related exchanges. Phishers use the sites which are visibly and semantically like those of authentic websites. Since the majority of the clients go online to get to the administrations given by the government and money related organizations, there has been a vital increment in phishing threats and attacks since some years.

As technology is growing, phishing methods have started to progress briskly and this should be avoided by making use of anti-phishing techniques to detect phishing. Machine learning is a authoritative tool that can be used to aim against phishing assaults. There are several methods or approaches to identify phishing websites.

The machine learning approaches to detect phishing websites have been proposed earlier and have been implemented. The central aim of this project is to implement the system with high efficiency, accuracy and cost effectively. That is been achieved. The project is implemented using 4 machine learning supervised classification models. The four classification models are K-Nearest Neighbor, Kernel Support vector machine, decision tree and random forest classifier. It was established that the Random forest classifier provides best accuracy for the selected dataset and gives an accuracy score of 96.82%.

Contents

ACKNOWLEDGEMENT i

ABSTRACT ii

CONTENTS ii

1 INTRODUCTION **1**

1.1 Background 1

1.2 Literature Survey 3

Motivation.. 13

Problem Statement... 14

Aim and Objective.. 14

Scope... 14

Challenges.. 14

Organization of the thesis 15

2 Fundamentals **16**

K-Nearest Neighbor algorithm.. 16

Kernel Support Vector Machine 17

Decision Tree 18

Random Forest Classifier 19

3 SYSTEM REQUIREMENT SPECIFICATION **21**

Hardware Requirements: 21

Software requirements.. 21

Supporting Python modules 21

Other Non-Functional Requirements 23

4 SYSTEM DESIGN 25

System Architecture 25

Data Flow Diagrams 26

Data Flow Diagram – Level 0 26

Data Flow Diagram – Level 1 27

Data Flow Diagram – Level 2 27

UML Activity Diagram 28

Summary 29

5 IMPLEMENTATION 30

Process involved in implementation 30

Classifiers 32

6 TESTING AND VALIDATION 34

Unit Testing 34

Unit Testing of KNN algorithm -1 35

Unit Testing of KNN algorithm -2 35

Unit Testing of kernel SVM algorithm -1 36

Unit Testing of kernel SVM algorithm -2 36

Unit Testing of Decision tree algorithm -1 37

Unit Testing of Decision tree algorithm -2 37

Unit Testing of RFC algorithm -1 38

Unit Testing of RFC algorithm -2 38

Integration Testing 38

Importing modules 39

Importing dataset 39

Importing user defined function 40

System testing 40

System testing 40

7 EXPERIMENTAL ANALYSIS AND RESULTS 41

7.1 Experimental analysis 41

7.1.1 KNN 42

7.1.2 Kernel SVM 42

7.1.3 Decision Tree 43

7.1.4 Random Forest Classifier 43

7.2 Comparative plots evaluating performance of the four algorithms 44

7.2.1 Accuracy score 44

7.2.2 Recall score 45

7.2.3 Precision 46

7.2.4 F1 score 47

7.3 Results 48

7.3.1 KNN 48

7.3.2 Kernel SVM 48

7.3.3 Decision Tree 49

7.3.4 Random Forest Classifier 50

8 CONCLUSION AND FUTURE WORKS 51

Conclusion 51

Future Enhancement.. 52

BIBLIOGRAPHY...53

APPENDICES61

List of Figures

1.1 Applications of Machine Learning in Cyber Security (Source: Paper- “Application of deep learning to cybersecurity: A survey”) 3

K-Nearest Neighbor classification (Source: an article titled “k Nearest Neighbor Classifier (kNN)-Machine Learning Algorithms”) 17

Initial graph (Source: article - “SVM and Kernel SVM”)................................17

After using the kernel and after the transformations (Source: Article - “SVM and Kernel SVM”)...18

Example of a decision tree (Source: xoriant.com)..19

Random forest classification (Source: Article titled “Random Forest classification and its implementation in Python”)..20

System Architecture...25

DFD - level 0..26

DFD - level 1..27

DFD - level 2..28

UML activity diagram...29

Implementation 30

The features in the dataset 31

Confusion matrix..41

KNN - Confusion matrix...42

Kernel SVM - confusion matrix..42

Decision Tree - confusion matrix..43

Random forest classifier - confusion matrix 43

Accuracy formula 44

Comparative plot of accuracy scores 44

Recall score 45

Comparative plot of recall scores 45

Precision score 46

Comparative plot of precision scores 46

F1 score 47

Comparative plot of F1 scores 47

Prediction by KNN 48

Prediction by Kernel SVM 49

Prediction by Decision tree 49

Prediction by Random forest classifier 50

1 Snapshot - Importing dataset 59

2 Snapshot - Splitting the dataset 59

3 Snapshot - Feature scaling 59

4 Snapshot - Accuracy score 59

5 Snapshot - Anaconda IDE 60

6 Snapshot - Jupyter web application 61

List of Tables

1.1 Literature Survey 12

3.1 Supporting python modules. 23

Testing of KNN algorithm -1 35

Testing of KNN algorithm -2 35

Testing of kernel SVM algorithm -1 36

Testing of kernel SVM algorithm -2 36

Testing of Decision tree algorithm -1 37

Testing of Decision tree algorithm -2 37

Testing of RFC -1 38

Testing of RFC -2 38

Import modules........ 39

Import dataset 39

Import function........ 40

System testing........ 40

GLOSSORY

ML	Machine Learning
AI	Artificial Intelligence
PC	Personal Computer
IDS	Intrusion Detection Systems
HTTPS	Hypertext Transfer Protocol Secure
URLs	Uniform Resource Locators
CSS	Cascading Style Sheets
CNN	Convolutional Neural Network
SVM	Support Vector Machine
KNN	K-Nearest Neighbor
KSVM	Kernel Support Vector Machine
RFC	Random Forest Classifier
DT	Decision Tree
DFD	Data flow diagram
UML	Unified Modeling Language
CM	Confusion Matrix

Chapter 1

INTRODUCTION

Background

Artificial intelligence is a new innovative science that reviews and creates hypotheses, strategies, procedures, and applications that recreate, grow and broaden human knowledge. ML is an arm of artificial intelligence and it is analogous to (and frequently overlap with) computational measurements, [1] that also concentrates on making predictions with the use of PCs. Machine leaning has solid relationship with scientific improvement, which tells methods, hypothesis and utilization regions to the field. ML is sometimes, in a while combined with data mining [2], but the data mining subfield focuses more on preparatory information investigation and is called as unsupervised learning. ML can likewise be unsupervised and be utilized to learn and set up pattern profiles for various entities and then used to find important anomalies. [3].

Cyber security is a set of innovations and procedures intended to secure PCs, networks, projects and information from assaults and unapproved access, modification, or annihilation [4] A system security framework comprises of a system assurance framework and furthermore a PC protection framework. Every one of these frameworks incorporates firewalls, antivirus programming, and intrusion detection system (IDS). IDSs help find, decide and distinguish unapproved system conduct [5], for instance, use, replicating, change and annihilation.

There are three important kind of network analysis for Intrusion detection system: misuse-based, also known as anomaly-based, signature-based, and hybrid.

- Misuse based detection strategies [6] mean to distinguish realized attacks by utilizing the marks of these attacks.
- Anomaly-based methods study the typical system and its conduct and distinguish anomalies as deviations from ordinary behavior.
- Hybrid detection conflates anomaly and misuse detection [7]. It is utilized to expand the rate of detection of accepted intrusions and to decrease the rate of false positives of unknown attacks.

The applications of machine learning (ML) methods in cybersecurity is rising than ever before as shown in fig 1.1. Beginning from IP traffic categorization, separating malicious traffic for intrusion detection, Machine learning is the one of the best answers that can impact against zero-day attacks. New exploration is being done by utilization of measurable traffic characteristics and ML techniques [8]. The word phishing was introduced in the year 1987 [9]. Phishing is an online thievery that robs an individual's private data and identity data. It is a sort of extortion where the assailant gets complete access to other individual's private data [10][11]
A hoax website similar to the authentic one is easily generated by an skillful designer and hence recognising the website as hoax can be tedious. Hence, we fall into such pits. These phishing websites call on users to give their account details by affirming itself as a genuine site, for instance., with the use of HTTPS. That convinces a user to rely on this fake site. They reassure of security and privacy although, gain the user's identity data. People make most money exchanges online. Taking care of the bills or transferring money [12], almost everything is made through sites or applications. Hence, identifying such fake website is of real significance. Based on the records that was discharged by Anti-Phishing Working Group, the total number of distinctive phishing sites recorded until 2018 September were 647,592 [13]. Once the attacker gets access to the passwords any harmful purpose is made easier.

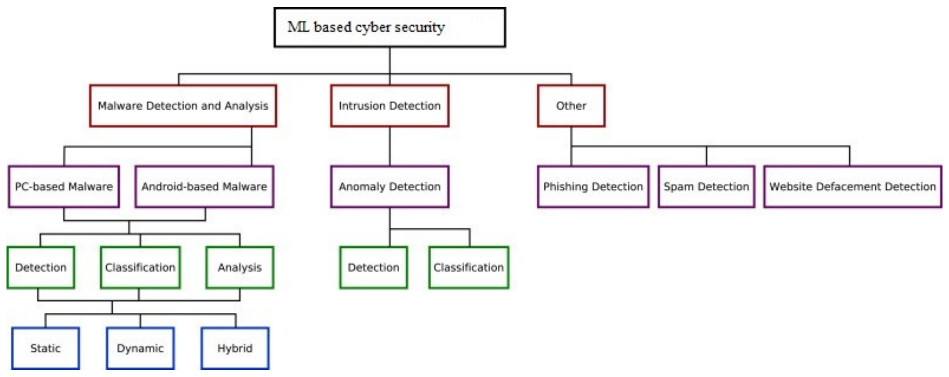


Figure 1.1: Applications of Machine Learning in Cyber Security (Source: Paper- “Application of deep learning to cybersecurity: A survey”)

Because of increase in the phishing attacks, numerous results are proposed which generates a solution to the issue. To build a framework which guarantees a solution against the phishing attack, there are several ways. Various other methods for detecting phishing attack are there like black list, Fuzzy rule-based, white list-based, cantina-based, machine learning based, Heuristic and image-based approaches [14][15]. There are several other studies that talks about a variety of methods and techniques [16][17][18] to detect the different types of phishing attacks [19][20][21]. Phishing sites looks to be like a genuine website and several individuals have problem in recognising such websites. Few anti-phishing techniques are in built in some of the browsers [22].

Literature Survey

A literature survey is an insightful article that presents the existing information including considerable discoveries just as theoretical and methodological commitments to a specific topic.

A very effective detection of phishing website model which is focused on optimal feature selection technique and also based on neural network (OFS-NN) is proposed [23]. In this proposed model, an index called feature validity value(FVV) has been generated to check the effects of all those features on the detection of such websites. Now, based on this newly generated index, an algorithm is developed to find from the phishing websites,the optimal features. This selected algorithm will be able to overcome the problem of over-fitting of the neural network to a great extend. These optimal features are then used to build an optimal classifier that detects phishing URLs by training the neural network.

A theory called Fuzzy Rough Set(FRS) [24] was devised to a tool that finds the most appropriate features from a few standardised dataset. These features are then sent to a few classifiers for detection of phishing. To investigate the feature selection for FRS in building a generalized detection of phishing, the models by a different dataset of 14,000 website samples are trained.

Feature engineering plays a vital role in finding solutions for detection of phishing websites, although the accuracy of the model greatly will be based on knowledge of the features. though the features taken from all these various dimensions are understandable, the limitation lies in the time taken to collect these features. To fix this drawback, the authors have proposed a multidimensional phishing detection feature [25] approach that concentrates on a rapid detection technique by making use of deep learning (MFPD)

To detect phishing occurrence accurately, a three phase detection called Web Crawler based Phishing Attack Detector (WC-PAD) [26] has been proposed. This takes the web's content, traffic and URL as input features. Now considering these features, classification is done.

PhishingNet [27], is an approach based on deep learning for detecting phishing URLs in a timely manner.

A detection system was developed which can match the dynamic environment and phishing websites. Because the approach considers various types of distinctive features from source code of webpages and URLs [28], this is a fully client side solution and needs no support of a third party.

A method called parse tree validation [29] has been proposed to find if a webpage is phishing or legitimate. This is an innovative approach to find such web sites by intercepting every hyperlinks of a present page through API of Google, and developing a parse tree from all those hyperlinks that were intercepted. In this, parsing begins from the root node. It goes by the Depth-FirstSearch (DFS) algorithm to determine if any child node has the same value as the root node.

A model as a solution was the focus in a study [30] that uses Random Forest classifier for detection of phishing websites by URL method.
An approach that combines to form an online tool, the collection, validation and detection of phishing websites. [31]. This online tool monitors in real-time the blacklist of PhishTank, validates and detects phishing website.
A framework was developed, known as ”Fresh-Phish” [32], that generates for phishing websites, present machine learning data. By using 30 various features of website which can be queried using Python, a very large dataset is built and the various ML classifiers are analyzed against this generated dataset to find out which has highest accuracy. This model analyzes both the accuracy as well as the time taken by the model to train.
A determined bond was built between the content-based heuristics and the authenticity of the website by evaluating both the phishing and legitimate websites’ training set. A framework called Phishing-Detective is presented [33] which detects the websites as phishing based on existing heuristics as well as new heuristics
An productive way using C4.5 decision tree classifier [34] as well as certain features of the URL was proposed to detect websites that are phishing.
There are many schemes for detection of phishing websites, among which the visual similarity scheme is collecting glances. The screenshot of the website is taken and stored in a database. It checks if the input screenshot of the website is same as the one stored in the database. If yes, then that website is predicted as phishing. But, if there are several similar websites, which ever is the first website that is given as input is taken as legitimate. Hence, it cannot predict correctly the authentic website and therefore recognising the goal website becomes tedious [35]. This detection method is proposed with target website finder by making use of images and CSS.

No	**Paper Title**	**Method/Techniques**	**Publish year**	**Limitations**

1	OFS-NN: ”An Effective Phishing Websites Detection Model Based on Optimal Feature Selection and Neural Network”	Proposed method has 3 stages:1. Defines a new index -FVV. 2. Designs an optimal feature selection algorithm.3. Produce the OFS-NN model	2019	The continuous growing of features that are sensitive of phishing attacks need collection of more features for the OFS
2	”Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection”	The proposed method uses Fuzzy Rough Set (FRS) theory to identify the features. The decision boundary is decided lower and upper approximation region. Using the lower and upper approximation memberships, a set member is decided to which category it belongs	2019	The specific features used in the method is not specified.

3	”Phishing Website Detection based on Multidimensional Features driven by Deep Learning”	The proposed method has the following stages: 1.character succession features of the URL are extricated as well as utilized for fast characterization 2. the LSTM (long short-term memory) network is utilized to catch setting semantic and dependency features of URL character groupings. 3. softmax classifies the features extracted	2019	It requires more computation and therefore an expensive method
4	”WC-PAD: Web Crawling based Phishing Attack Detection”	It is a 3-phase detection of phishing attack approach. The 3 phases of WC-PAD are 1) blacklist of DNS 2) Approach based on Heuristics and 3) Approach based on Web crawler. Feature extraction as well as phishing attack detection both makes use of web crawler.	2019	Time consuming as it involves three phases and each website has to go through the three phases.

5	”Phishing URL Detection via CNN and Attention-Based Hierarchical RNN”	CNN module is used to derive representation of spatial feature that is character level of the URLs. Then the representational features are combined by using a CNN of 3 layers to create precise feature representations of URLs. That is then used for training the classifier of phishing URLs.	2019	false positive rate is high
6	”An Adaptive Machine Learning Based Approach for Phishing Detection Using Hybrid Features”	A phishing detection system was developed by making use of classifier of Machine learning called XCS. It is an adaptive ML technique that is online. This advances a lot of rules called classifiers. This model derives 38 features from source code of webpage and URLs.	2019	

7	”Phishing Detection in Websites using Parse Tree Validation”	If the number of recurrence of root node is: 1. more than half the number of nodes, then probability of authenticity is more. 2. quarter the number of nodes, the probability of authenticity is moderate. 3. less than the quarter number of nodes, then probability of authenticity is low which means phishing probability is high.	2018	The false negative and false positive rates are high.
8	”A new method for Detection of Phishing Websites: URL Detection”	The three major phases in this work are Parsing, Heuristic Classification of data, Performance Analysis in this model. All of these phases use various and distinctive methods for data processing to get results that are better.	2018	Does not give full information about the techniques used.

9	”PhishBox: An approach for phishing validation and detection”	The approach that is proposed makes use of 2 phase detection model to increase its performance. 1. An ensemble model is designed for validating the phishing data and for decreasing the cost of labeling manually,active learning is applied. 2.The model for detection is being trained using these validated data.	2018	The blacklist contained invalid data when monitored with an interval set as 12 hours.
10	”Fresh-Phish:A framework for Auto-Detection of Phishing Websites”	This framework was developed considering there are no other open source frameworks which, for a given website, measures the features. The work also created an updated set of data that could be used by researchers for their work. Analysis of TensorFLow based neural network and linear classifier and SVM with kernels both Gaussian and linear were done against dataset of FreshPhish	2017	Less accuracy and assumption of the dataset considered for legitimate website is accurate.

11	”Phishing Website Detection Frame-work Through Web Scraping and Data Mining”	A web crawler that scrapes the constituents of both le-gitimate and phishing web-sites was developed. The constituents were then an-alyzed to get the heuris-tics rate and their commit-ment scale factor towards the wrongness of a site. A data mining tool was used to analyze the data that was derived from the web scraper and patterns were found.	2017	Exact accuracy of the model is not mentioned.
12	”Phishing Sites Detec-tion based on C4.5 Decision Tree Algo-rithm”	The approach proposed makes use of features that were extracted from the URL to make decision about the legitimacy of the URL given as input. To generate the rules, the c4.5 algorithm was used. The rules produced are utilized to order the submitted URL as genuine or phishing with better productivity.	2017	Overall accu-racy is less as the paper con-siders limited URL features.

13	”Visual Similarity-based Phishing Detection Scheme using Image and CSS with Target Website Finder”	The main focus is on the fact that authentic web-sites are usually linked by many websites and those websites are regarded as le-gitimate, the screenshot and CSS of which are stored in a database. Because CSS is a file which characterizes the sites visual substance, as-saulter regularly take real CSS to imitate the real site. Hence, by finding the site which counterfeits appear-ance or CSS of real site, we identify phishing site and its objective at the same time.	2017	The websites that are linked at least by one website are also recorded in the white list assuming it to be legitimate.

Table 1.1: Literature Survey.

From the above, ML methods plays a vital role in many applications of cybersecurity and shall remain an encouraging path that captivates more such investigations. When coming to the reality, there are several barriers that are limitations during implementa-tions. As discussed, there are many approaches earlier proposed for detecting phishing website attack and they also have their own limitations. Therefore, the aim of the project is detection of phishing website attack using a novel Machine learning technique.

Motivation

There are many Anti phishing techniques that helps us protect from phishing sites. Mozilla Firefox, Safari and Google chrome makes use of Google Safe Browsing (GSB) [13] service that will block the phishing websites. There are also many such tools like McFee Site Advisor, Quick Heal, Avast and Netcraft which are widely used. GSB analyzes a URL by making use of the blacklist approach. The main disadvantage of GSB was that it was unable to detect the phishing website since updation of blacklist was not done. In case of Netcraft, a website that phishing was recorded as phishing although it wasn't blocked. The blocking is done by Netcraft only when it is sure 100% that the website is phishing. The warning is given only when the user clicks the right button on the icon to find the risk rating. The risk is when the individual doesn't check the rating or makes a decision to use it after checking the rating. Security against security attacks online is provided by some soft wares like QuickHeal and Avast. The functioning of Avast anti-virus was checked after installing it. The Avast browser was not able to successfully find the phishy URL that was successfully determined by Netcraft and GSB.

This above mentioned points accepts the necessity of anti phishing tools that are advanced in nature. It is noteworthy that these tools must be installed independently. A lay person might never install tools if he is not aware of practices like phishing. If that is the case, then people rely only on GSB service. Hence, the awareness considering such anti phishing tools and phishing is very important. Also, no individual should fully rely on tools because it is seen that they might lead to misclassification.

Problem Statement

The problem is derived after making a thorough observation and study about the method of classification of phishing websites that makes use of machine learning techniques. We must design a system that should allow us to:

- Accurately and efficiently classify the websites into legitimate or phishing.
- Time consumed for detection should be less and should be cost effective.

Aim and Objective

The project's objectives are as follows:

- To study various automatic phishing detection methods
- To identify the appropriate machine learning techniques and define a solution using the selected method
- To select an appropriate dataset for the problem statement
- To apply appropriate algorithms to achieve the solution to phishing attacks

Scope

The focus of the project is on machine learning (ML) methods for network analysis of intrusion detection especially phishing websites attack.

Challenges

The challenges faced during the project are as follows:

- Finding the appropriate dataset.
- Feature extraction required the study of various modules and understanding each module and getting the expected outcome from it.

Organization of the thesis

Chapter 1 incorporates a presentation about the application of ML in cyber security. It details the problem statement, objectives and scope of the project. It also tells about the challenges faced during the development of the project. **Chapter 2** incorporates the study and research about the phishing attacks and its detection using Machine learning techniques. It gives a detailed description of the earlier works done in this front and the limitations of those related works. **Chapter 3** discusses about the software and hardware requirements which is necessary for the system. The chapter details about the minimum requirements needed for the project and also about the modules of Python that are used. **Chapter 4** tells about the system design and its representation using architecture, data flow diagrams and activity diagram. It gives a graphical and diagrammatic representation of the system for better understanding and the system's, user's and run time perspective of the project. In **chapter 5**, the implementation of this project is being examined. The chapter details about the dataset used, the steps involved in the implementation, the classifiers used, etc. In **chapter 6**, the test cases are being examined and a comparison of the expected output and the actual output is being made to validate our result. In **chapter 7**, the outcome obtained and the environmental setup up of the project is being discussed. I conclude the project in **chapter 8** and also discuss about the future enhancements to the project.

Chapter 2

Fundamentals

In ML and statistics, classification method is an approach involving supervised learning where computer program gains information from input and afterward utilizes this figuring out how to characterize new observations. Here are few classification techniques used in the detection of phishing URLs.

K-Nearest Neighbor algorithm

The k-nearest neighbors classifier is a basic, simple to-actualize administered ML algorithm that can be utilized to take care of both classification and regression issues. The KNN algorithm [36] presumes that comparative things are real in closeness. As such, comparable things are close to one another. The KNN algorithm relies on the assumption that being authentic enough for the algorithm to be beneficial. KNN catches the possibility of similarity with computing the separation between focuses on a graph as shown in fig 2.1.

There are several ways of calculating the distance between the points [37]. However, the Euclidean distance computation is one of the most popular ones [38].

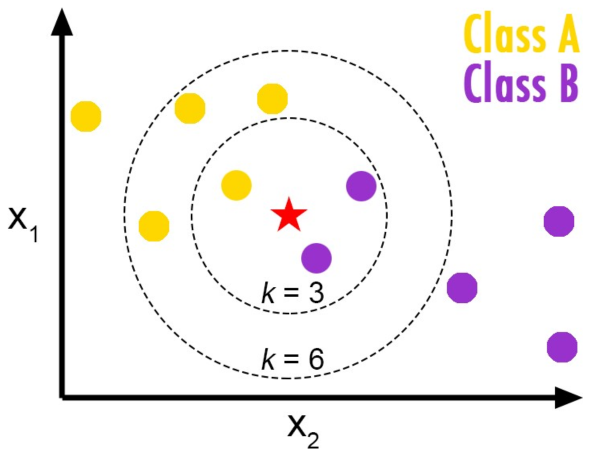


Figure 2.1: K-Nearest Neighbor classification (Source: an article titled “k Nearest Neighbor Classifier (kNN)-Machine Learning Algorithms”)

Kernel Support Vector Machine

The fundamental thought is that when a data set is indistinguishable in the present dimensions, include another dimension, perhaps that way the information will be distinct [39]. This is called the kernel trick. Mapping to higher dimension is not blindly including an additional dimension.An example of mapping from 1D to 2D is as shown in fig 2.2 and fig 2.3.

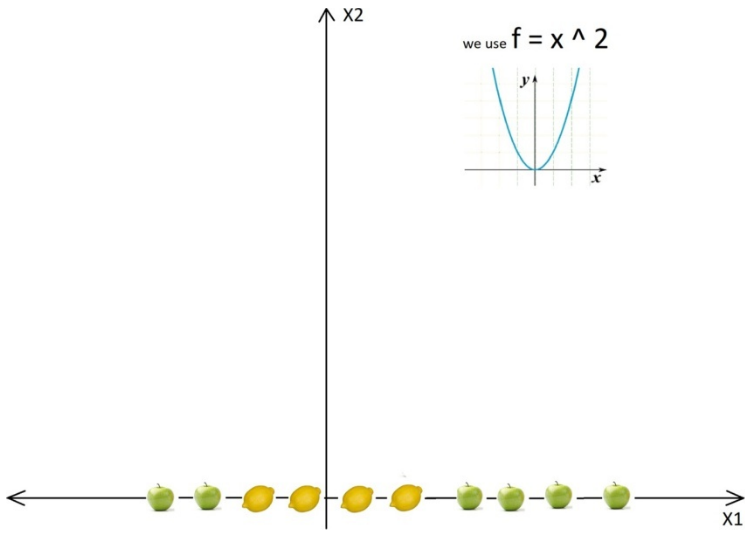


Figure 2.2: Initial graph (Source: article - “SVM and Kernel SVM”)

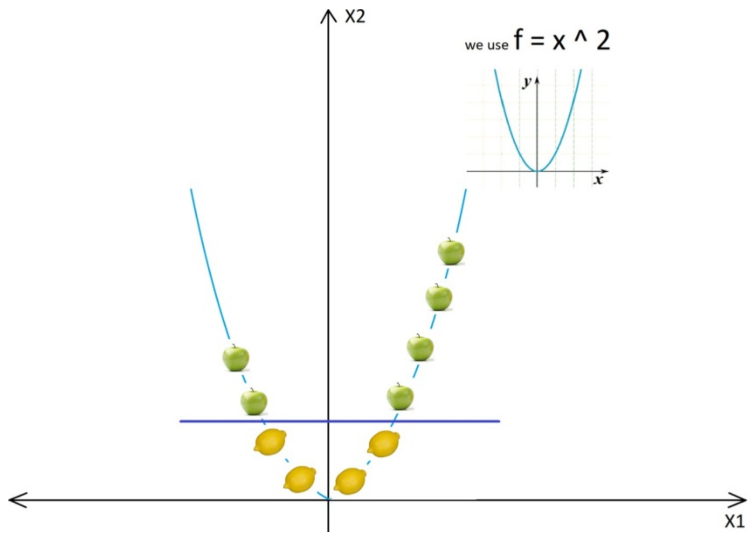


Figure 2.3: After using the kernel and after the transformations (Source: Article - “SVM and Kernel SVM”)

We must transform it in such a way that we create this level separation intentionally. The transformation is called kernel. Some of the most popular ones are Gaussian kernel, Sigmoid kernel, Radial Basis Function, etc.

Decision Tree

A decision tree [40] - a basic representation that classifies instances. A decision tree constitutes of the following:

- Nodes: specific attributes’ estimation is tested by nodes.
- Branches: they are the interface with following nodes or the leaf nodes and relates to the result.
- Leaf nodes: Nodes that are terminal and anticipate the result.

Let us understand it with an example: Consider fig 2.4. To predict whether a person is unfit or fit, when data like diet patterns, physical action, age, etc are given. The decision nodes are the issues like ”What is the age?”, ’Does he/she works out?’, ’Does he/she eat pizzas'? Also, the leaf nodes tells about the results i.e., "unfit" or "fit".

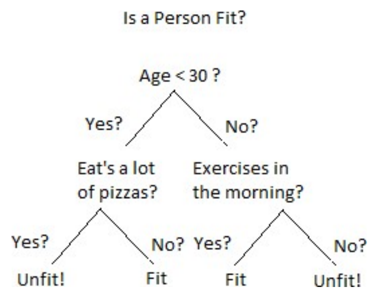


Figure 2.4: Example of a decision tree (Source: xoriant.com)

Binary recursive portioning is the process through which the tree will be created. This is a recursive procedure of parting the information into partitions, afterward separating it again on every one of the branches. In Decision Tree Classification, new instance is characterized by giving it to a progression of attempts that decide the class name of the model.These attempts are composed to a structure of hierarchy and is known as a decision tree. Decision Trees abides by Divide-and-Conquer method.

Random Forest Classifier

Random forest,as the name implies, constitutes of many separate decision tress which all works as an ensemble Each separate tree of the Random forest[41] gives out a class forecast and the class with the most votes transforms into our model's desire as shown in fig 2.5.

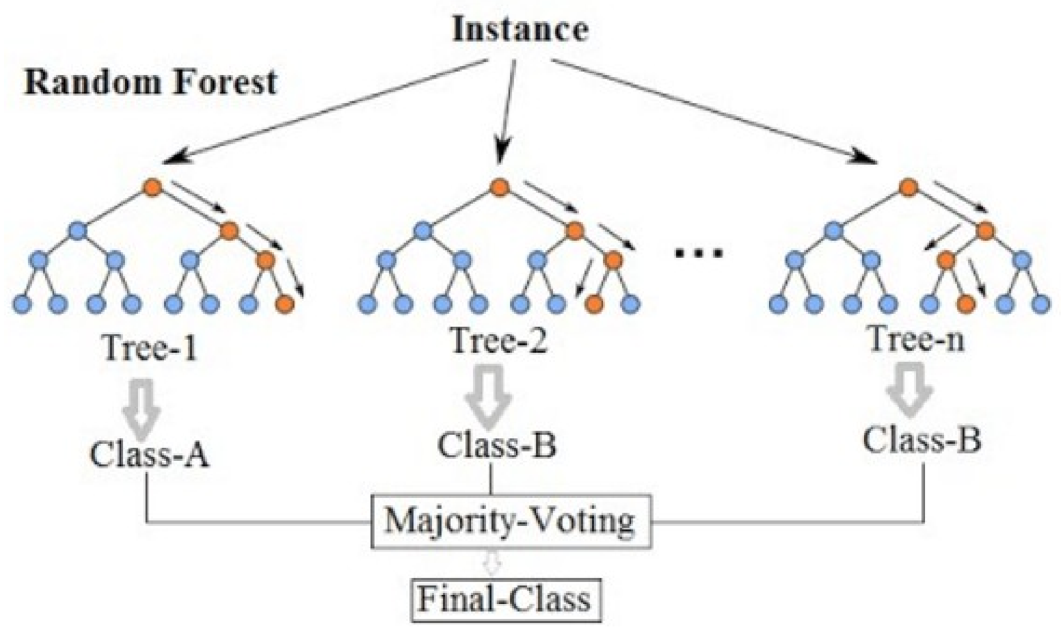


Figure 2.5: Random forest classification (Source: Article titled “Random Forest classification and its implementation in Python”)

The principal idea propelling random forest is a straightforward however an amazing way — the knowledge of groups. In information science talk, the clarification that the random forest model works so well is: A colossal number of commonly uncorrelated models (trees) functioning as a council will outrun any of the its fundamental models exclusively.

Chapter 3

SYSTEM REQUIREMENT SPECIFICATION

Hardware Requirements:

- Processor CPU - Intel Pentium Dual Core and Higher
- Hard Disk capacity - 512MB Space required minimum
- RAM - 4GB minimum

Software requirements

- Programming language - Python
- Operating system - Windows 8.1 or above
- IDE - Anaconda , iPython version 3.x

Supporting Python modules

Python has an approach to place definitions in a document and use them in a content or in an intuitive case of the interpreter. Such a file is known as a module; definitions from a module can be brought into different modules or into the fundamental module. Some of the modules used in the project are as shown in Table 3.1 [42]:

No	**Python Modules**	**Description**
1	Ipaddress	ipaddress gives the capacities to generate, control and work on IPv4 and IPv6 addresses and networks.
2	Re	This module gives regular expression matching activities like those found in Perl.
3	urllib.request	The urllib.request module characterizes functions and classes which help in opening URLs (for the most part HTTP) in a complex world.
4	BeautifulSoup	BeautifulSoup is a package in python for parsing HTML and XML records. It makes a parse tree for parsed pages that can be utilized to extricate information from HTML, which is valuable for web scraping.
5	Socket	The BSD interface of socket is given access by this module

6	Requests	The HTTP requests are allowed to send by this module making use of Puthon.
7	Whois	WHOIS is an inquiry and response convention that is comprehensively used for addressing databases that store the selected customers or trustees of an Internet resource. for example, a domain name, an autonomous framework or an IP address block , also simultaneously used for broad extend of information.

Table 3.1: Supporting python modules.

Other Non-Functional Requirements

A non-functional requirement is a determination that depicts the framework's activity abilities and requirements that improve its usefulness.
Some of them are as follows:

- Reusability: the same code with limited changes can be used for detecting phishing attacks variants like smishing, vishing, etc.
- Maintainability: The implementation is very basic and includes print statements that makes it easy to debug.
- Usability: The software used is very user friendly and open source. It also runs on

any operating system.

- Scalability: The implementation can include detection of vishing, smishing, etc.

Chapter 4

SYSTEM DESIGN

System Architecture

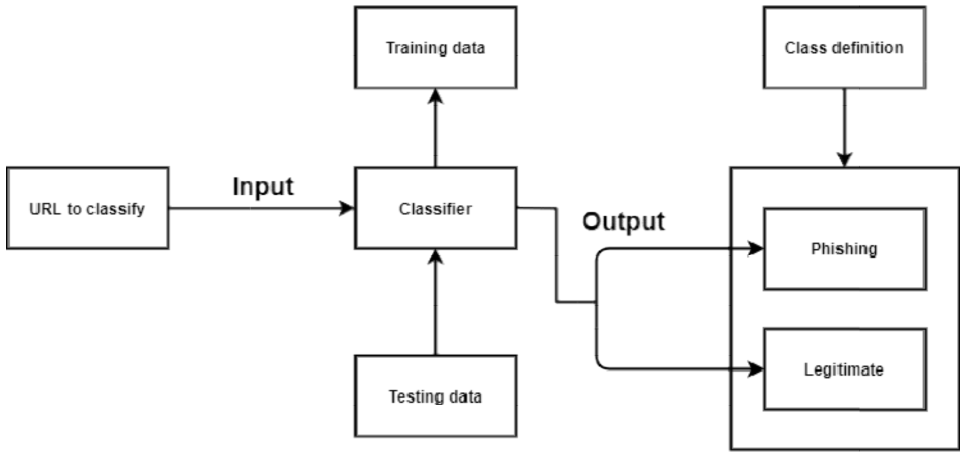


Figure 4.1: System Architecture

The architecture of the system is as shown in fig 4.1; the URLs to be classified as legitimate or phishing is fed as input to the appropriate classifier. Then classifier that is being trained to classify URLs as phishing or legitimate from the training dataset uses the pattern it recognized to classify the newly fed input.

The features such as IP address, URL length, domain, having favicon, etc. are extracted from the URL and a list of its values is generated. The list is fed to the classifiers such as KNN, kernel SVM, Decision tree and Random Forest classifier. These models' perfor-

mance is then evaluated and an accuracy score is generated. The trained classifier using the generated list predicts if the URL is legitimate or phishing.
The list contains values 1, 0 and -1 if the features exist, not applicable and if the features doesn't exist respectively. There are 30 features being considered in this project.

Data Flow Diagrams

DFDs are used to depict graphically the data flow in a system [43]. It explains the processes involved in a system from the input to the report generation. It shows all possible paths from one entity to another of aa system. The detail of a data flow diagram can be represented in three different levels that are numbered 0, 1 and 2.
There are many types of notations to draw a data flow diagram among which Yourdon-Coad and Gane-Sarson method are popular. The DFDs depicted in this chapter uses the Gane-Sarson DFD notations.

Data Flow Diagram – Level 0

DFD level 0 is called a Context Diagram. It is a simple overview of the whole system being modeled.Fig 4.2 shows the DFD level 0 of the system.

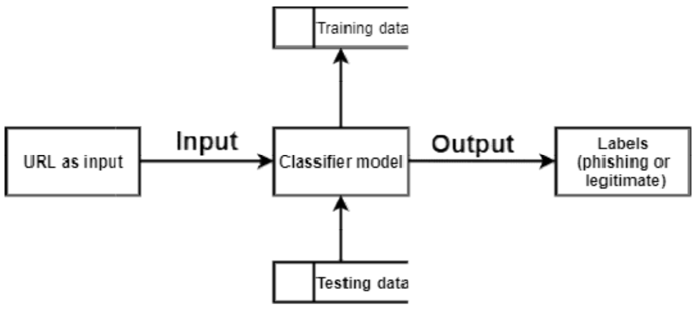


Figure 4.2: DFD - level 0

It shows the system as a high-level process with its relationship to the external entities. It should be easily acknowledged by a wide range of audience from stakeholders to developers to data analysts.

Data Flow Diagram – Level 1

DFD level 1 gives a more detailed explanation of the Context diagram. The high-level process of the Context diagram is broken down into its subprocesses. The DFD level 1 of the system is depicted in fig 4.3

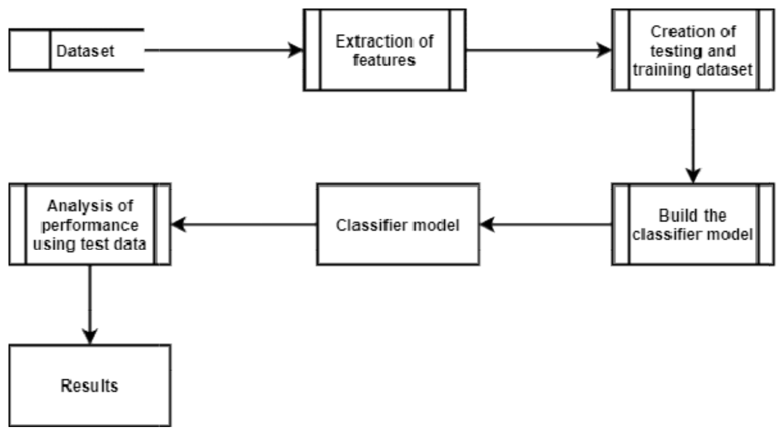

Dataset
Extraction of features
Creation of testing and training dataset
Build the classifier model
Classifier model
Analysis of performance using test data
Results

Figure 4.3: DFD - level 1

The Level 1 DFD takes a step deep by including the processes involved in the system such as feature extraction, splitting of dataset, building the classifier, etc. and hence gives a more detailed vision of the system.

Data Flow Diagram – Level 2

DFD level 2 goes one more step deeper into the subprocesses of Level 1. Fig 4.4 shows the DFD level 2 of the system. It might require more text to get into the necessary level of detail about the functioning of the system.
The Level 2 gives a more detailed sight of the system by categorizing the processes involved in the system to three categories namely preprocessing, feature scaling and classification. It also graphically depicts each of these categories in detail and gives a complete idea of how the system works.

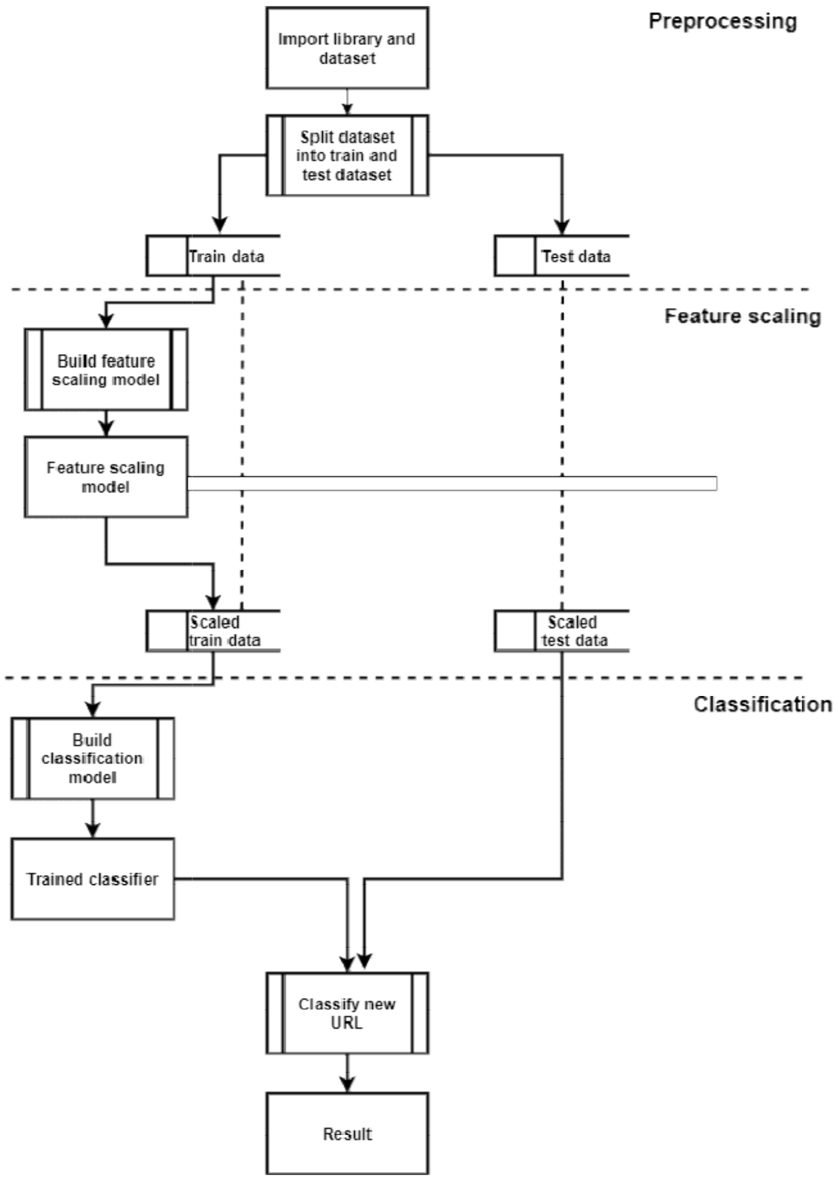


Figure 4.4: DFD - level 2

UML Activity Diagram

Activity diagram is a behavioral diagram [44].The fig 4.5 shows the activity diagram of the system.

It depicts the control flow from a start point to an end point showing various paths which exists during the execution of the activity.

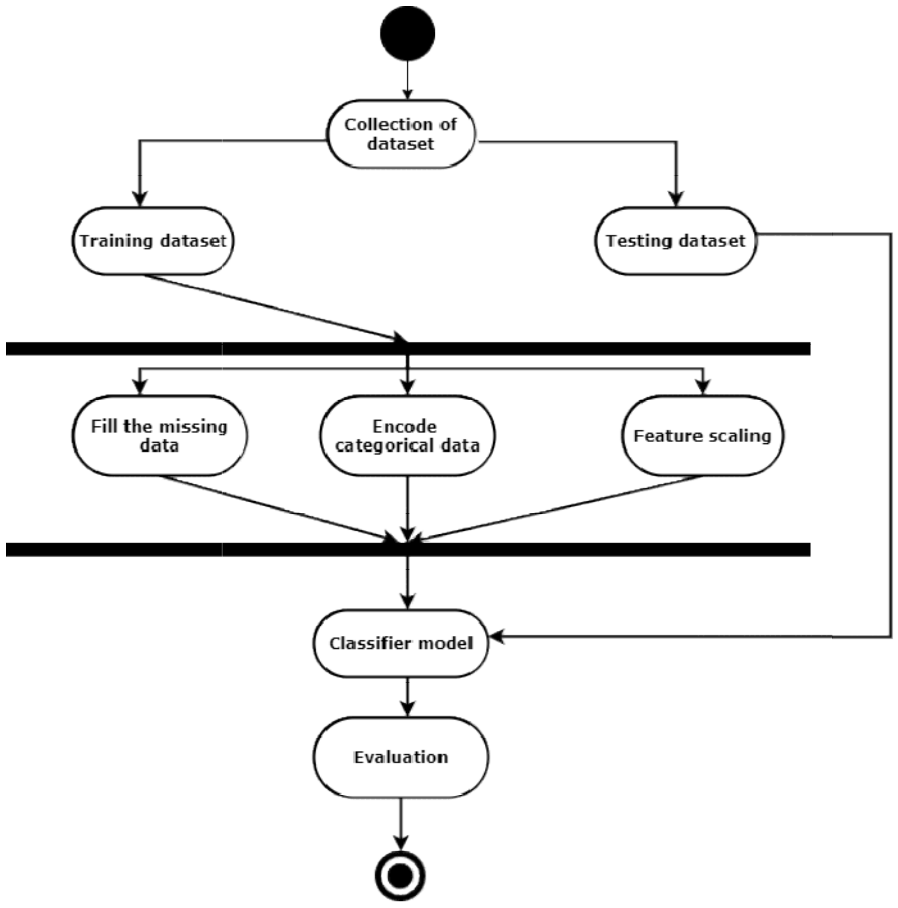


Figure 4.5: UML activity diagram

Summary

The system’s architecture, the processes involved from input to output with varying levels of complexity and the system’s behaviour is graphically represented for better understanding of the system in the above chapter.

Chapter 5

IMPLEMENTATION

This chapter of the report illustrates the approach employed to classify the URLs as either phishing or legitimate. The methodology involves building a training set. The training set is used for training a machine learning model, i.e., the classifier. Fig 5.1 shows the diagrammatic representation of the implementation.

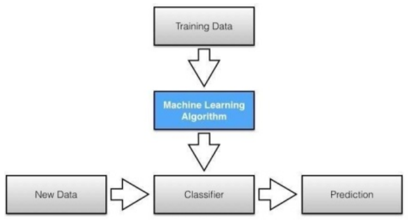


Figure 5.1: Implementation

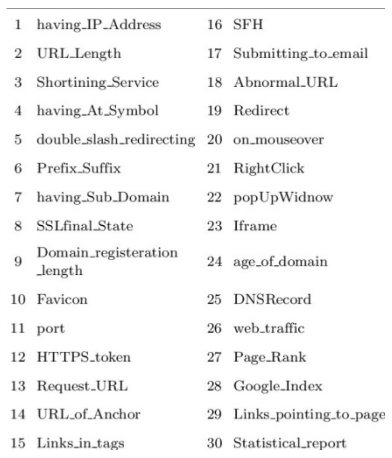
Process involved in implementation

The first step of the research work was determining the right data set. The dataset selected was collected from Kaggle for this task. The reasons behind selecting this dataset are several. It includes:

- The data set is large, so working with it is intriguing
- The number of features in the data set is 30 giving a wide range of features mak-

ing the predictions a little more accurate. The fig 5.2 shows the features being considered.

- The number of URLs is quite evenly distributed among the 2 categories.



1	having_IP_Address	16	SFH
2	URL_Length	17	Submitting_to_email
3	Shortining_Service	18	Abnormal_URL
4	having_At_Symbol	19	Redirect
5	double_slash_redirecting	20	on_mouseover
6	Prefix_Suffix	21	RightClick
7	having_Sub_Domain	22	popUpWidnow
8	SSLfinal_State	23	Iframe
9	Domain_registeration _length	24	age_of_domain
10	Favicon	25	DNSRecord
11	port	26	web_traffic
12	HTTPS_token	27	Page_Rank
13	Request_URL	28	Google_Index
14	URL_of_Anchor	29	Links_pointing_to_page
15	Links_in_tags	30	Statistical_report

Figure 5.2: The features in the dataset

- **Splitting**: the dataset into training part of dataset and testing part of dataset. The dataset was split into training and testing dataset with 75% for training and 25% for testing using the "train_test_split" method. The splitting was done after assigning the dependent variables and independent variables.

- **Preprocessing**: Preprocessing involves filling the missing data or removing the missing data and getting a clean dataset [45]. But the dataset chosen was already preprocessed and did not require any further preprocessing from my end. The only step to be performed in preprocessing was feature scaling.

- **Feature scaling**: Feature Scaling is a procedure to normalize the independent variable present in the information in a fixed range [46]. It is performed during the data pre-processing to deal with varying magnitudes. There are two ways of feature scaling – Normalization and Standardization. The project uses standardization feature scaling methods.

  The variables should be put in the same scale, else one variable might dominate others hence might affect the result.

Standardization: Standardization is another scaling procedure where the values are based on the mean with a unit standard deviation. This implies the mean of that attribute gets zero and the resultant distribution has a unit standard deviation.

Xstd = (x – mean(x))/ standard deviation(x) —————- Eq: 5.1

Normalization: Normalization is a scaling method where values are moved and rescaled so they wind up going somewhere in the range of 0 and 1. It is otherwise called Min-Max scaling.

Xnorm = (x – min(x))/(max(x) – min(x)) ———————-Eq: 5.2

The project uses StandardScaler. It fits and transforms only the independent variables. The dependent variables need not be scaled in classification method. The dummy variables which we get from categorical data may or may not be scaled depending on context.

- **Feature extraction**: Feature values are extracted using python modules like whois, requests, socket, re, ipaddress, BeautifulSoup, etc. to get information regarding ip address, length of url, domain name, subdomains, presence of favicon, etc. The value obtained is stored in a list. This is being done because the dataset is in this format and hence the classifier will be trained with input of this format. Therefore, when a URL is passed as input to the system, it converts it into a python list of 30 elements each representing its respective feature and there after that list is fed to the trained classifier. The classifier that is being used includes KNN, kernel SVM, Decision Tree and random forest classifier.

Classifiers

- **sklearn.neighbors.KNeighborsClassifier**

  Classifier implementing k-nearest neighbors.

  Parameters used:

  – N_neighbors: It is the number of neighbors to be considered while categorizing and was considered 5 in the algorithm

- Metric: It depicts the distance metric to be used. The one used in the algorithm is ‘minkowski’
- p: It is the power parameter for the metric. The algorithm uses p = 2 which is equivalent to Euclidean distance

- **sklearn.svm.SVC**

  Classifier used to implement kernel SVM.

  Parameters used:

  - Kernel: the value is set for this parameter in the algorithm is “rbf” and hence considers nonlinear method.

- **sklearn.tree.DecisionTreeClassifier**

  Classifier that is used to implement decision tree.

  Parameters used:

  - criterion: the function that is used to measure the quality of a split. The one that is used in the algorithm is “entropy”

- **sklearn.ensemble.RandomForestClassifier**

  Classifier that is used to implement random forest classifier.

  Random forest,as the name implies, constitutes of many separate decision tress which all works as an ensemble Each separate tree of the Random forest gives out a class forecast and the class with the most votes transforms into our model’s desire as

  Parameters used:

  - N_estimators: The number of trees in the forest. The number used in the algorithm is 10.
  - criterion: the function that is used to measure the quality of a split. The one that is used in the algorithm is “entropy”

Chapter 6

TESTING AND VALIDATION

In this chapter, we check for the working of the proposed system by testing and comparing the result of the algorithm and the actual result. It is basically validating the system. The testing is done for each algorithm with a legitimate and phishing URL and the results are as follows.

Below are the section to be concentrated in testing chapter

Unit Testing

Unit Testing is a testing approach where the units of the modules are investigated to check regardless of whether they are fit as a fiddle to be utilized.

Unit Testing of KNN algorithm -1

Test case	01
Test Name	“Testing of KNN -1”
Input	http://crikster.co.za/altcustomer CARD/altCustomerB/images/js.php &quot;c &gt;&lt;/script&gt;&lt;script type=&quot;text/javascript&quot;&gt; var siteURL = ’http://crikster.co.za/altcustom
Expected output	Phishing
Actual Output	Phishing
Remark	Success

Table 6.1: Testing of KNN algorithm -1

Unit Testing of KNN algorithm -2

Test case	02
Test Name	“Testing of KNN -2”
Input	https://twitter.com/login
Expected output	Legitimate
Actual Output	Legitimate
Remark	Success

Table 6.2: Testing of KNN algorithm -2

Unit Testing of kernel SVM algorithm -1

Test case	03
Test Name	“Testing of kernel SVM -1”
Input	http://h.paypal.de-checking.net /de/ID.php?u=LhsdoOKJfsjdsdvg
Expected output	Phishing
Actual Output	Phishing
Remark	Success

Table 6.3: Testing of kernel SVM algorithm -1

Unit Testing of kernel SVM algorithm -2

Test case	04
Test Name	“Testing of kernel SVM -2”
Input	https://www.udemy.com/
Expected output	Legitimate
Actual Output	Legitimate
Remark	Success

Table 6.4: Testing of kernel SVM algorithm -2

Unit Testing of Decision tree algorithm -1

Test case	05
Test Name	“Testing of Decision tree -1”
Input	paypal.de@secure-server.de/secure-environment
Expected output	Phishing
Actual Output	Phishing
Remark	Success

Table 6.5: Testing of Decision tree algorithm -1

Unit Testing of Decision tree algorithm -2

Test case	06
Test Name	“Testing of Decision tree -2”
Input	https://www.wikipedia.org/
Expected output	Legitimate
Actual Output	Legitimate
Remark	Success

Table 6.6: Testing of Decision tree algorithm -2

Unit Testing of RFC algorithm -1

Test case	07
Test Name	“Testing of Random forest classifier -1”
Input	http://63.17.167.23/pc/ verification.htm?=https://www.paypal .com/
Expected output	Phishing
Actual Output	Phishing
Remark	Success

Table 6.7: Testing of RFC -1

Unit Testing of RFC algorithm -2

Test case	08
Test Name	“Testing of Random forest classifier -2”
Input	https://calendar.google.com/calendar/r
Expected output	Legitimate
Actual Output	Legitimate
Remark	Success

Table 6.8: Testing of RFC -2

Integration Testing

Integration Testing is a testing approach where the units of the modules are integrated and then investigated to check regardless of whether they are fit to be utilized.

Importing modules

Test case	09
Test Name	“Importing modules”
Input	Import ”module” statements
Expected output	The module to be imported
Actual Output	The module was imported and could be used
Remark	Success

Table 6.9: Import modules

Importing dataset

Test case	10
Test Name	“Importing dataset”
Input	Import ”dataset” statement
Expected output	The dataset to be imported
Actual Output	The dataset was imported and could be used
Remark	Success

Table 6.10: Import dataset

Importing user defined function

Test case	11
Test Name	“Importing user defined function”
Input	Import ”extraction” function
Expected output	The function to be imported that returns a list
Actual Output	The function was imported and returned the list as expected
Remark	Success

Table 6.11: Import function

System testing

System testing is a testing approach that checks for completely integrated system’s validation.

System testing

Test case	12
Test Name	“System testing”
Input	Sample URL provided to check whether it is a phishing or legitimate URL
Expected output	All the modules like importing of modules, dataset and functions defined and provide the result
Actual Output	The application reacts as expected
Remark	Success

Table 6.12: System testing

Chapter 7

EXPERIMENTAL ANALYSIS AND RESULTS

In this chapter, the execution and results of the project are being discussed.

Experimental analysis

Confusion matrix(CM) is a graphical summary of the correct predictions and incorrect predictions that is made by a classifier that can be used to determine the performance. In abstract terms, the CM is as shown in fig 7.1:

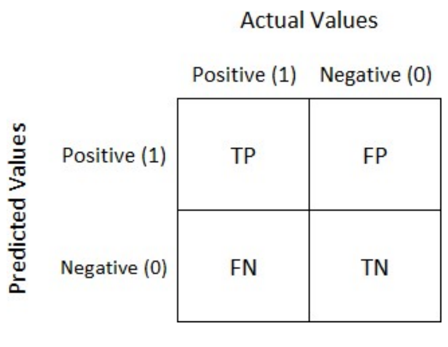


Figure 7.1: Confusion matrix

In the above figure TP is True positive, TN is True negative, FP is False Positive and

FN is False Negative. The confusion matrix of the algorithms used are as shown:

KNN

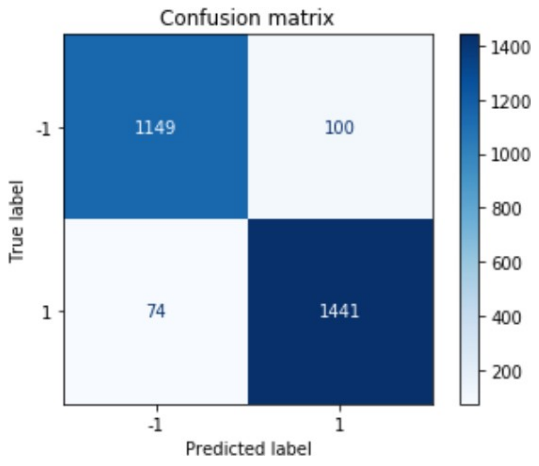


Figure 7.2: KNN - Confusion matrix

Kernel SVM

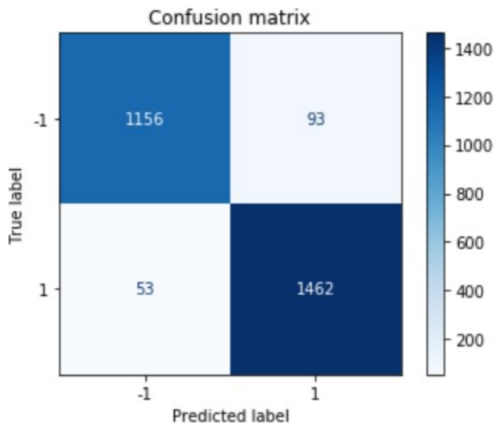


Figure 7.3: Kernel SVM - confusion matrix

Decision Tree

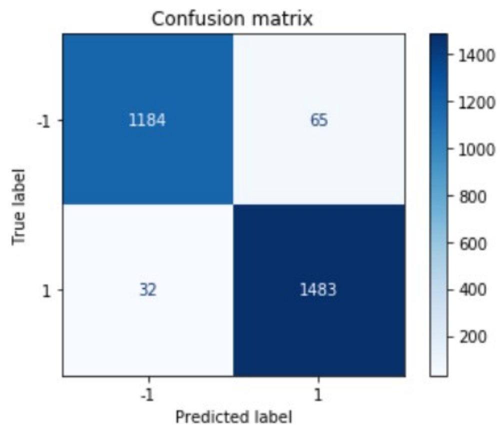


Figure 7.4: Decision Tree - confusion matrix

Random Forest Classifier

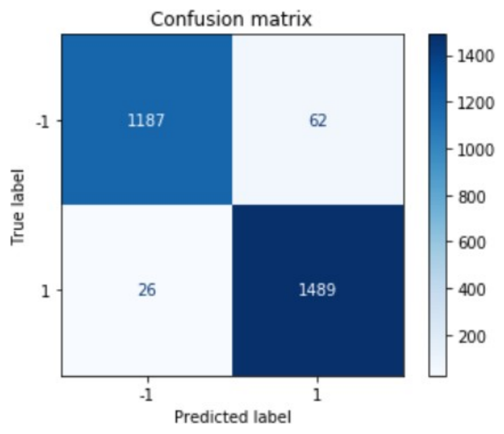


Figure 7.5: Random forest classifier - confusion matrix

Comparative plots evaluating performance of the four algorithms

Accuracy score

The accuracy is the fraction of sample corrected correctly. The below fig 7.6 shows the formula used for accuracy. The fig 7.7 is a comparative plot that compares the accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

Fraction predicted correctly

Figure 7.6: Accuracy formula

of the four algorithms namely; KNN, Kernel SVM, Decision tree and random forest classifier.

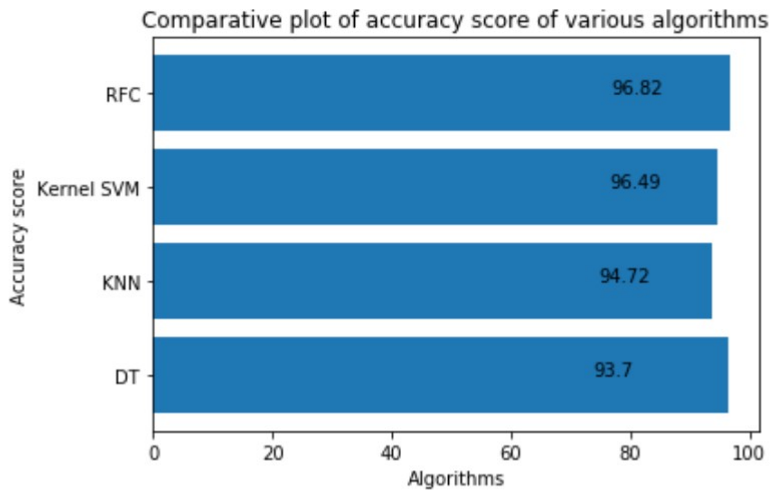


Figure 7.7: Comparative plot of accuracy scores

Recall score

The recall score is the fraction of positive events that was correctly predicted. Fig 7.8 shows the formula used for recall score.

$$\text{Recall (Sensitivity)} = \frac{TP}{TP + FN}$$

Fraction of positives predicted correctly

Figure 7.8: Recall score

The fig 7.9 is a comparative plot that compares the recall score of the four algorithms.

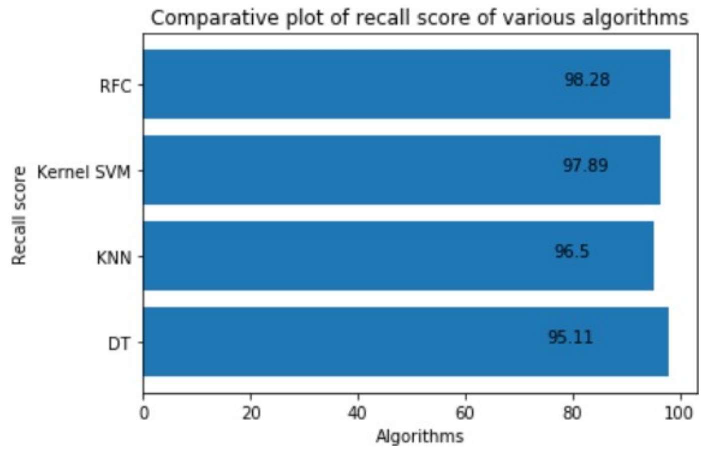


Figure 7.9: Comparative plot of recall scores

Precision

Precision is the fraction of positive events that are really positive. Fig 7.10 shows the formula to calculate the precision from the CM.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Fraction of predicted positives that are actually positive

Figure 7.10: Precision score

The fig 7.11 is a comparative plot that compares the precision score of the four algorithms.

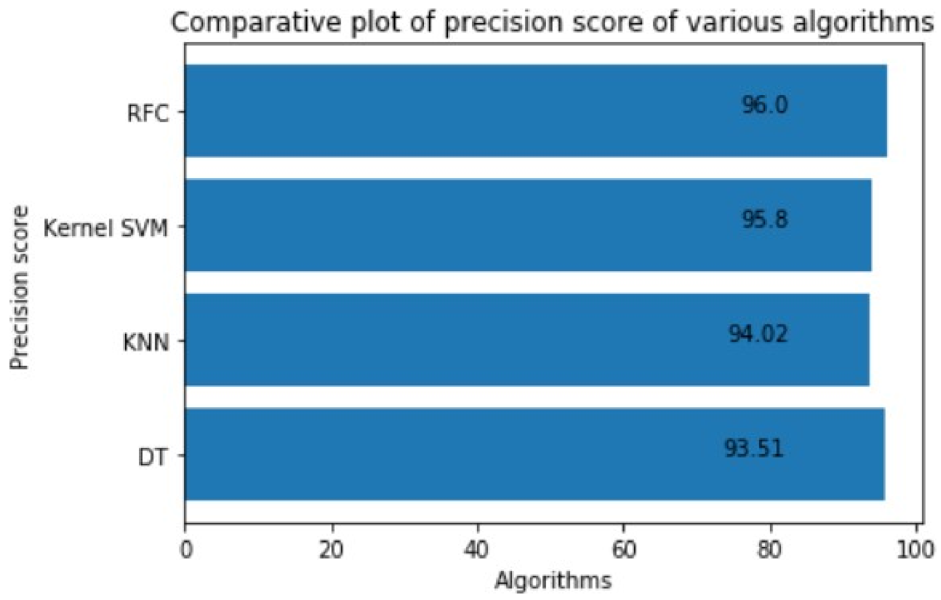


Figure 7.11: Comparative plot of precision scores

F1 score

F1 score is calculated as the harmonic mean of precision and recall. The higher the F1 score, the better the model. Fig 7.12 shows the formula for evaluating the F1 score.

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2 * (precision * recall)}{precision + recall}$$

Figure 7.12: F1 score

The fig 7.13 is a comparative plot that compares the F1 score of the four algorithms.

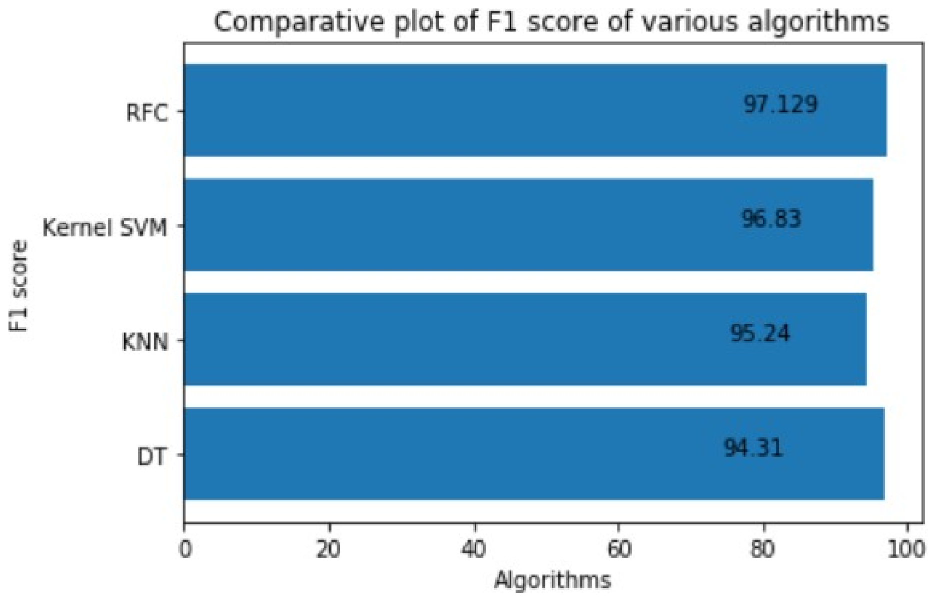
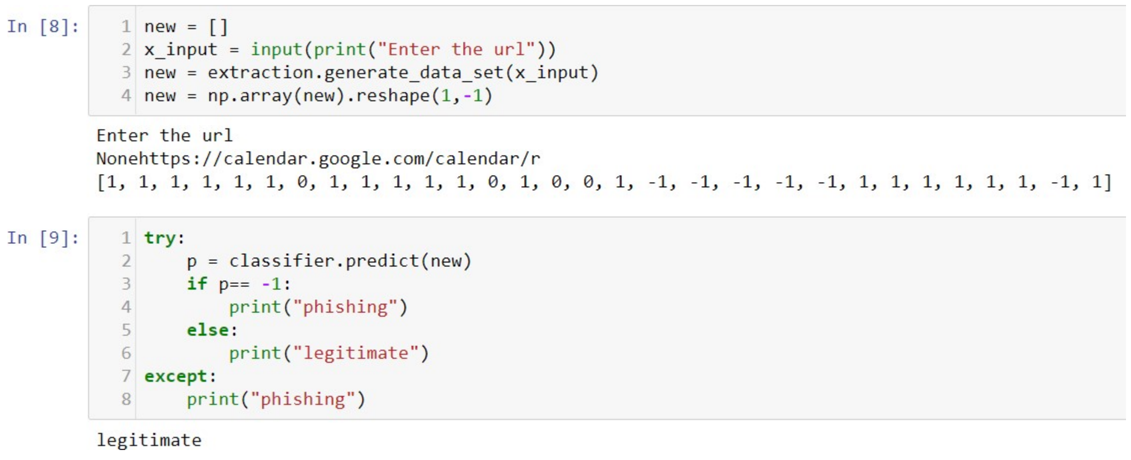


Figure 7.13: Comparative plot of F1 scores

Results

KNN

- Input URL - https://calendar.google.com/calendar/r
- Algorithm – KNN
- Expected outcome – Legitimate
- Obtained – Legitimate



```
In [8]:
new = []
x_input = input(print("Enter the url"))
new = extraction.generate_data_set(x_input)
new = np.array(new).reshape(1,-1)
```

```
Enter the url
Nonehttps://calendar.google.com/calendar/r
[1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, -1, -1, -1, -1, -1, 1, 1, 1, 1, 1, 1, -1, 1]
```

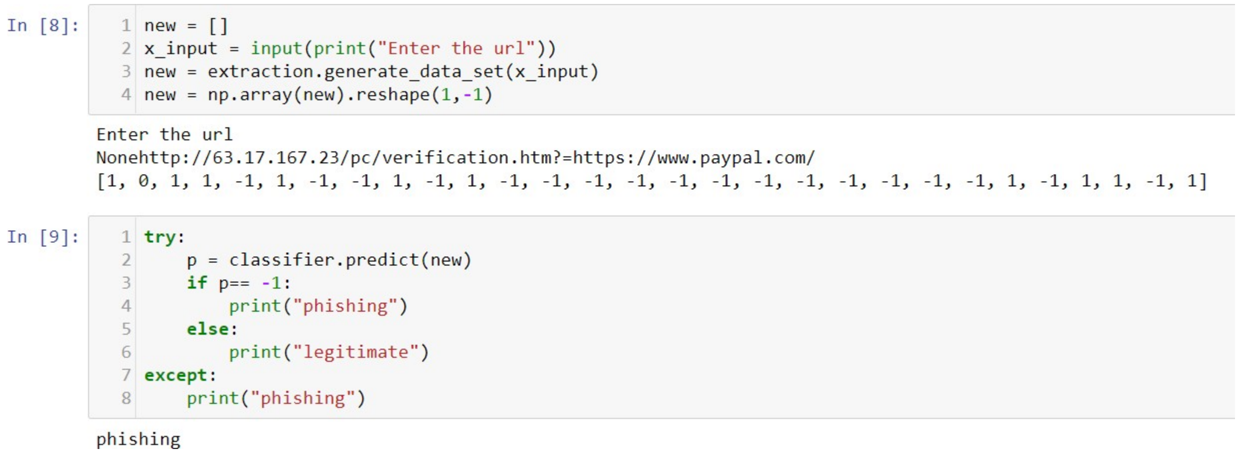
```
In [9]:
try:
    p = classifier.predict(new)
    if p== -1:
        print("phishing")
    else:
        print("legitimate")
except:
    print("phishing")
```

```
legitimate
```

Figure 7.14: Prediction by KNN

Kernel SVM

- Input URL - http://63.17.167.23/pc/verification.htm?=https://www.paypal.com/
- Algorithm – Kernel SVM
- Expected outcome – Phishing
- Obtained – Phishing

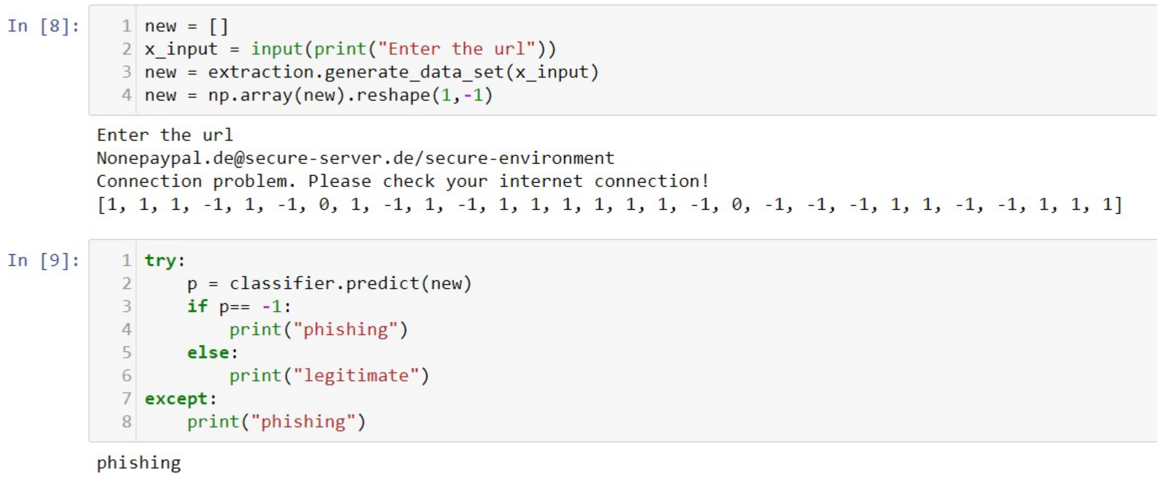


```
In [8]: new = []
        x_input = input(print("Enter the url"))
        new = extraction.generate_data_set(x_input)
        new = np.array(new).reshape(1,-1)

Enter the url
Nonehttp://63.17.167.23/pc/verification.htm?=https://www.paypal.com/
[1, 0, 1, 1, -1, 1, -1, -1, 1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, -1, 1, 1, -1, 1]

In [9]: try:
            p = classifier.predict(new)
            if p== -1:
                print("phishing")
            else:
                print("legitimate")
        except:
            print("phishing")

phishing
```

Figure 7.15: Prediction by Kernel SVM

Decision Tree

- Input URL - paypal.de@secure-server.de/secure-environment
- Algorithm – Decision tree
- Expected outcome – Phishing
- Obtained – Phishing

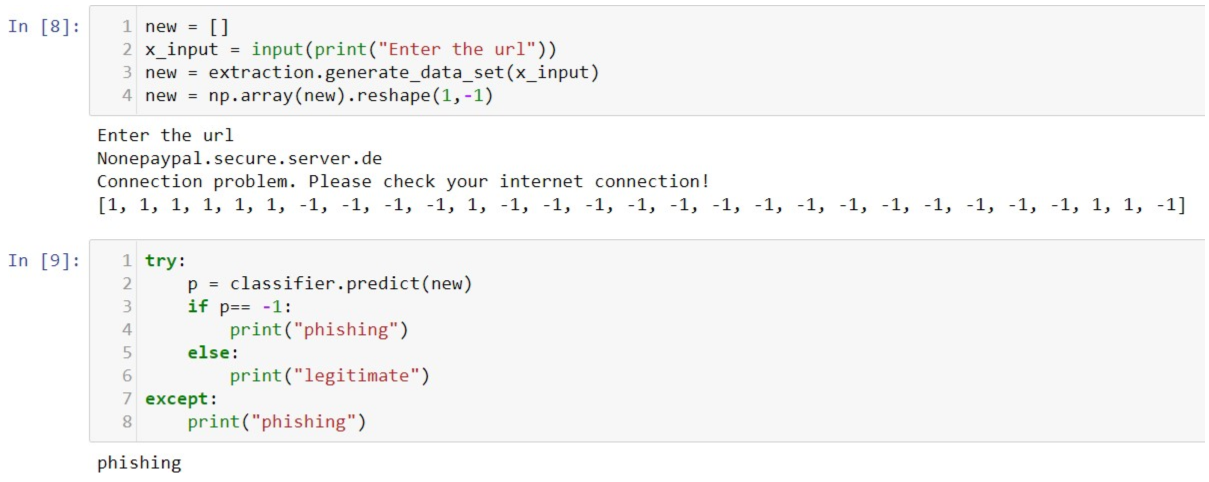


```
In [8]: new = []
        x_input = input(print("Enter the url"))
        new = extraction.generate_data_set(x_input)
        new = np.array(new).reshape(1,-1)

Enter the url
Nonepaypal.de@secure-server.de/secure-environment
Connection problem. Please check your internet connection!
[1, 1, 1, -1, 1, -1, 0, 1, -1, 1, -1, 1, 1, 1, 1, 1, 1, -1, 0, -1, -1, -1, 1, 1, -1, -1, 1, 1, 1]

In [9]: try:
            p = classifier.predict(new)
            if p== -1:
                print("phishing")
            else:
                print("legitimate")
        except:
            print("phishing")

phishing
```

Figure 7.16: Prediction by Decision tree

Random Forest Classifier

- Input URL - paypal.secure.server.de
- Algorithm – Random Forest classifier
- Expected outcome – Phishing
- Obtained – Phishing



```
In [8]:
new = []
x_input = input(print("Enter the url"))
new = extraction.generate_data_set(x_input)
new = np.array(new).reshape(1,-1)
```

```
Enter the url
Nonepaypal.secure.server.de
Connection problem. Please check your internet connection!
[1, 1, 1, 1, 1, 1, -1, -1, -1, -1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, 1, 1, -1]
```

```
In [9]:
try:
    p = classifier.predict(new)
    if p== -1:
        print("phishing")
    else:
        print("legitimate")
except:
    print("phishing")
```

```
phishing
```

Figure 7.17: Prediction by Random forest classifier

Chapter 8

CONCLUSION AND FUTURE WORKS

Conclusion

The demonstration of phishing is turning into an advanced danger to this quickly developing universe of innovation. Today, every nation is focusing on cashless exchanges, business online, tickets that are paperless and so on to update with the growing world. Yet phishing is turning into an impediment to this advancement. Individuals are not feeling web is dependable now. It is conceivable to utilize AI to get information and assemble extraordinary information items. A lay person, completely unconscious of how to recognize a security danger shall never invite the danger of making money related exchanges on the web. Phishers are focusing on installment industry and cloud benefits the most.

The project means to investigate this region by indicating an utilization instance of recognizing phishing sites utilizing ML. It aimed to build a phishing detection mechanism using machine learning tools and techniques which is efficient, accurate and cost effective. The project was carried out in Anaconda IDE and was written in Python.

The proposed method used four machine learning classifiers to achieve this and a comparative study of the four algorithms was made. A good accuracy score was also achieved. The four algorithms used are K-Nearest neighbor, Kernel Support Vector Machine, Decision Tree and Random Forest Classifier. All the four classifiers gave promising results with the best being Random Forest Classifier with an accuracy score of 96.82%. The accuracy score might vary while using other datasets and other algorithms might provide better accuracy than random forest classifier. Random forest classifier is an ensemble classifier and hence the high accuracy. This model can be deployed in real time to detect the URLs as phishing or legitimate.

Future Enhancement

Further work can be done to enhance the model by using ensembling models to get greater accuracy score. Ensemble methods is a ML technique that combines many base models to generate an optimal predictive model. Further reaching future work would be combining multiple classifiers, trained on different aspects of the same training set, into a single classifier that may provide a more robust prediction than any of the single classifiers on their own.

The project can also include other variants of phishing like smishing, vishing, etc. to complete the system. Looking even further out, the methodology needs to be evaluated on how it might handle collection growth. The collections will ideally grow incrementally over time so there will need to be a way to apply a classifier incrementally to the new data, but also potentially have this classifier receive feedback that might modify it over time.

Bibliography

[1] Reid G. Smith and Joshua Eckroth. Building ai applications: Yesterday, today, and tomorrow. *AI Magazine*, 38(1):6–22, Mar. 2017.

[2] Panos Louridas and Christof Ebert. Machine learning. *IEEE Software*, 33:110–115, 09 2016.

[3] Michael Jordan and T.M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science (New York, N.Y.)*, 349:255–60, 07 2015.

[4] Steven Aftergood. Cybersecurity: The cold war online. *Nature*, 547:30+, Jul 2017. 7661.

[5] Aleksandar Milenkoski, Marco Vieira, Samuel Kounev, Alberto Avritzer, and Bryan Payne. Evaluating computer intrusion detection systems: A survey of common practices. *ACM Computing Surveys*, 48:12:1–, 09 2015.

[6] Chirag N. Modi and Kamatchi Acha. Virtualization layer security challenges and intrusion detection/prevention systems in cloud computing: a comprehensive review. *The Journal of Supercomputing*, 73(3):1192–1234, Mar 2017.

[7] Eduardo Viegas, Altair Santin, Andre Fanca, Ricardo Jasinski, Volnei Pedroni, and Luiz Soares de Oliveira. Towards an energy-efficient anomaly-based intrusion detection engine for embedded systems. *IEEE Transactions on Computers*, 66:1–1, Jan 2016.

[8] Y. Xin, L. Kong, Z. Liu, Y. Chen, Y. Li, H. Zhu, M. Gao, H. Hou, and C. Wang. Machine learning and deep learning methods for cybersecurity. *IEEE Access*, 6:35365–35381, 2018.

[9] Neha R. Israni and Anil N. Jaiswal. A survey on various phishing and anti-phishing measures. *International journal of engineering research and technology*, 4, 2015.

[10] Pingchuan Liu and Teng-Sheng Moh. Content based spam e-mail filtering. pages 218–224, 10 2016.

[11] N. Agrawal and S. Singh. Origin (dynamic blacklisting) based spammer detection and spam mail filtering approach. In *2016 Third International Conference on Digital Information Processing, Data Mining, and Wireless Communications (DIPDMWC)*, pages 99–104, 2016.

[12] Vikas Sahare, Sheetalkumar Jain, and Manish Giri. Survey:anti-phishing framework using visual cryptography on cloud. *JAFRC*, 2, 01 2015.

[13] S. Patil and S. Dhage. A methodical overview on phishing detection along with an organized way to construct an anti-phishing framework. In *2019 5th International Conference on Advanced Computing Communication Systems (ICACCS)*, pages 588–593, 2019.

[14] Dipesh Vaya, Sarika Khandelwal, and Teena Hadpawat. Visual cryptography: A review. *International Journal of Computer Applications*, 174:40–43, 09 2017.

[15] Saurabh Saoji. Phishing detection system using visual cryptography, 03 2015.

[16] C. Pham, L. A. T. Nguyen, N. H. Tran, E. Huh, and C. S. Hong. Phishing-aware: A neuro-fuzzy approach for anti-phishing on fog networks. *IEEE Transactions on Network and Service Management*, 15(3):1076–1089, 2018.

[17] K. S. C. Yong, K. L. Chiew, and C. L. Tan. A survey of the qr code phishing: the current attacks and countermeasures. In *2019 7th International Conference on Smart Computing Communications (ICSCC)*, pages 1–5, 2019.

[18] G. Egozi and R. Verma. Phishing email detection using robust nlp techniques. In *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 7–12, 2018.

[19] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang. Phishing-alarm: Robust and efficient phishing detection via page component similarity. *IEEE Access*, 5:17020–17030, 2017.

[20] G. J. W. Kathrine, P. M. Praise, A. A. Rose, and E. C. Kalaivani. Variants of phishing attacks and their detection techniques. In *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 255–259, 2019.

[21] Muhammet Baykara and Zahit Gurel. Detection of phishing attacks. pages 1–5, 03 2018.

[22] Prof. Gayathri Naidu . A survey on various phishing detection and prevention techniques. *International Journal of Engineering and Computer Science*, 5(9), May 2016.

[23] E. Zhu, Y. Chen, C. Ye, X. Li, and F. Liu. Ofs-nn: An effective phishing websites detection model based on optimal feature selection and neural network. *IEEE Access*, 7:73271–73284, 2019.

[24] Mahdieh Zabihimayvan and Derek Doran. Fuzzy rough set feature selection to enhance phishing attack detection, 03 2019.

[25] P. Yang, G. Zhao, and P. Zeng. Phishing website detection based on multidimensional features driven by deep learning. *IEEE Access*, 7:15196–15209, 2019.

[26] T. Nathezhtha, D. Sangeetha, and V. Vaidehi. Wc-pad: Web crawling based phishing attack detection. In *2019 International Carnahan Conference on Security Technology (ICCST)*, pages 1–6, 2019.

[27] Y. Huang, Q. Yang, J. Qin, and W. Wen. Phishing url detection via cnn and attention-based hierarchical rnn. In *2019 18th IEEE International Conference On*

Trust, Security And Privacy In Computing And Communications/13th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE), pages 112–119, 2019.

[28] M. M. Yadollahi, F. Shoeleh, E. Serkani, A. Madani, and H. Gharaee. An adaptive machine learning based approach for phishing detection using hybrid features. In *2019 5th International Conference on Web Research (ICWR)*, pages 281–286, 2019.

[29] C. E. Shyni, A. D. Sundar, and G. S. E. Ebby. Phishing detection in websites using parse tree validation. In *2018 Recent Advances on Engineering, Technology and Computational Sciences (RAETCS)*, pages 1–4, 2018.

[30] S. Parekh, D. Parikh, S. Kotak, and S. Sankhe. A new method for detection of phishing websites: Url detection. In *2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pages 949–952, 2018.

[31] J. Li and S. Wang. Phishbox: An approach for phishing validation and detection. In *2017 IEEE 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, pages 557–564, 2017.

[32] H. Shirazi, K. Haefner, and I. Ray. Fresh-phish: A framework for auto-detection of phishing websites. In *2017 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 137–143, 2017.

[33] A. J. Park, R. N. Quadari, and H. H. Tsang. Phishing website detection framework through web scraping and data mining. In *2017 8th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 680–684, 2017.

[34] Lisa Machado and Jayant Gadge. Phishing sites detection based on c4.5 decision tree algorithm. pages 1–5, 08 2017.

[35] S. Haruta, H. Asahina, and I. Sasase. Visual similarity-based phishing detection scheme using image and css with target website finder. In *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pages 1–6, 2017.

[36] Afroz Chakure. ”k-nearest neighbors (knn) algorithm”. https://towardsdatascience.com/k-nearest-neighbors-knn-algorithm-bd375d14eec7, 07 2019.

[37] Kittipong Chomboon, Pasapitch Chujai, Pongsakorn Teerarassammee, Kittisak Kerdprasop, and Nittaya Kerdprasop. An empirical study of distance metrics for k-nearest neighbor algorithm. pages 280–285, 01 2015.

[38] Natasha Sharma. ”importance of distance metrics in machine learning modelling”. https://towardsdatascience.com/importance-of-distance-metrics-in-machine-learning-modelling-e51395ffe60d, 01 2019.

[39] Czako Zoltan. ”svm and kernel svm”. https://towardsdatascience.com/ svm-and-kernel-svm-fed02bef1200, 12 2018.

[40] Afroz Chakure. ”decision tree classification, an introduction to decision tree classifier”. https://towardsdatascience.com/decision-tree-classification-de64fc4d5aac, 07 2019.

[41] Tony Yiu. ”understanding random forest, how the algorithm works and why it is so effective”. https://towardsdatascience.com/understanding-random-forest-58381e0602d2, 06 2019.

[42] ”modules - python documentation”. https://docs.python.org/3/tutorial/modules.html, 2020.

[43] ”dfd symbols”. https://www.visual-paradigm.com/guide/data-flow-diagram/what-is-data-flow-diagram/, 2020.

[44] ”what is activity diagram?”. https://www.visual-paradigm.com/guide/ uml-unified-modeling-language/what-is-activity-diagram/, 2020.

[45] Theodoros Iliou, Christos-Nikolaos Anagnostopoulos, Marina Nerantzaki, and George Anastassopoulos. A novel machine learning data preprocessing method for enhancing classification algorithms performance. 09 2015.

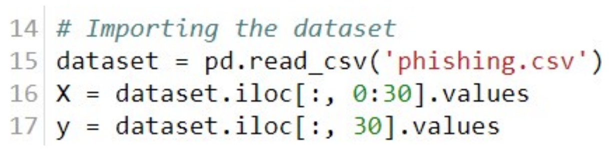
[46] Sebastian Raschka. About feature scaling and normalization and the effect of standardization for machine learning algorithms. 07 2014.

Appendices

Appendix A

Sample Coding

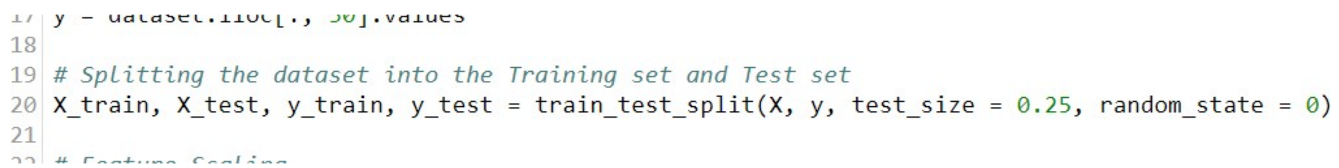
Importing dataset



```
# Importing the dataset
dataset = pd.read_csv('phishing.csv')
X = dataset.iloc[:, 0:30].values
y = dataset.iloc[:, 30].values
```

Figure 1: Snapshot - Importing dataset

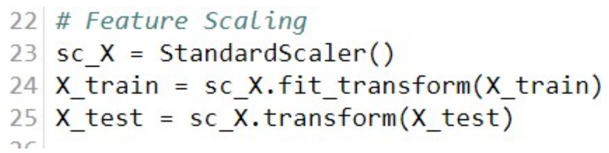
Splitting the dataset



```
y = dataset.iloc[:, 30].values

# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state = 0)

# Feature Scaling
```

Figure 2: Snapshot - Splitting the dataset

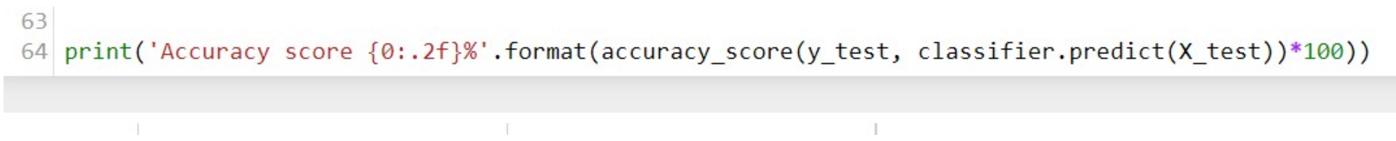
Feature scaling



```
# Feature Scaling
sc_X = StandardScaler()
X_train = sc_X.fit_transform(X_train)
X_test = sc_X.transform(X_test)
```

Figure 3: Snapshot - Feature scaling

Accuracy score



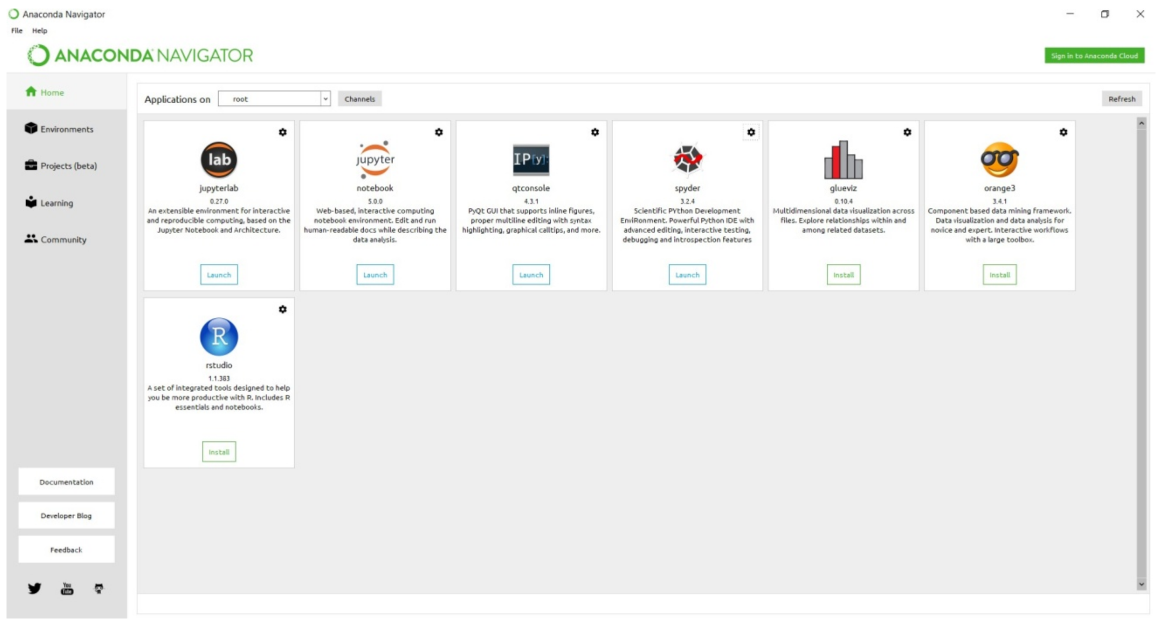
```

print('Accuracy score {0:.2f}%'.format(accuracy_score(y_test, classifier.predict(X_test))*100))
```

Figure 4: Snapshot - Accuracy score

Appendix B

Sample Snapshots

Integrated development environment



jupyterlab
0.27.0
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

notebook
5.0.0
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

qtconsole
4.3.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

spyder
3.2.4
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

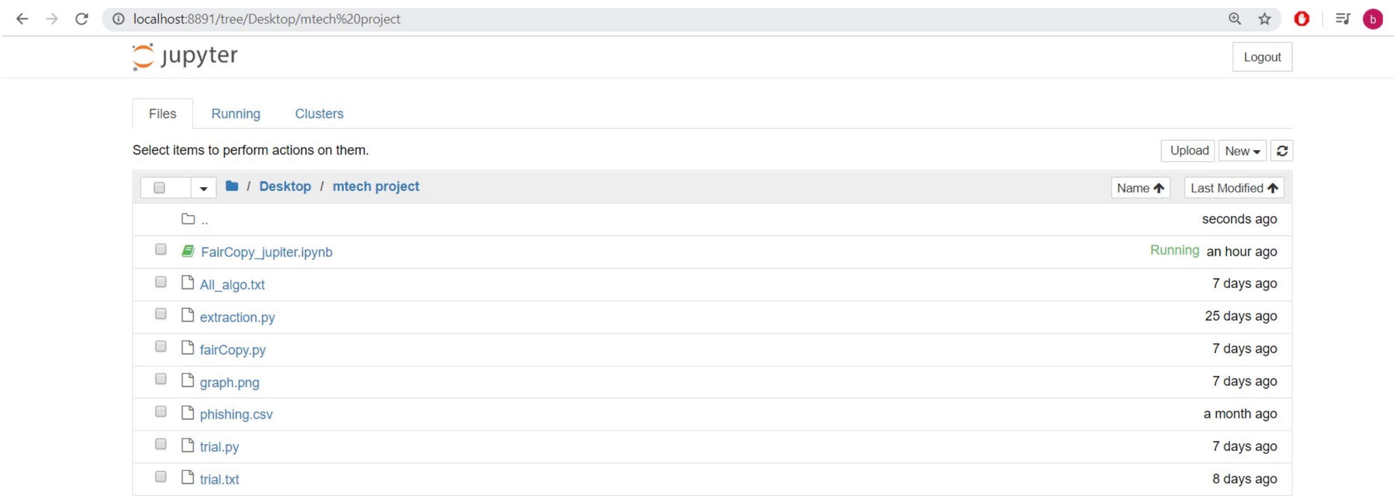
glueviz
0.10.4
Multidimensional data visualization across files. Explore relationships within and among related datasets.

orange3
3.4.1
Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

rstudio
1.1.383
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Figure 5: Snapshot - Anaconda IDE

Web application



localhost:8891/tree/Desktop/mtech%20project

Select items to perform actions on them.

/ Desktop / mtech project

Name	Last Modified
..	seconds ago
FairCopy_jupiter.ipynb	Running an hour ago
All_algo.txt	7 days ago
extraction.py	25 days ago
fairCopy.py	7 days ago
graph.png	7 days ago
phishing.csv	a month ago
trial.py	7 days ago
trial.txt	8 days ago

Figure 6: Snapshot - Jupyter web application