VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI, KARNATAKA - 590018

Project Report on

"DETECTING PHISHING WEBSITE USING MACHINE LEARNING"

In Partial Fulfilment of the Requirement for the Award of the Degree of Bachelor of Engineering in Computer Science Engineering

Submitted by,

AJAYKUMAR MAJUKAR	**2KL15CS005**
NADEEM MD SALIM NAKKASHI	**2KL15CS040**
DARSHAN BELGAONKAR	**2KL15CS017**

Under the Guidance of,
PROF. SAVITA BAKARE

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING "Accredited by NBA"

KLE Dr. M. S. SHESHGIRI COLLEGE OF ENGINEERING AND TECHNOLOGY BELAGAVI – 590008

2018-2019

KLE Dr. M S SHESHGIRI

COLLEGE OF ENGINEERING & TECHNOLOGY,

BELAGAVI -590008

Department of Computer Science Engineering

Certificate

This is to certify that the project work entitled **"DETECTING PHISHING WEBSITE USING MACHINE LEARNING"** has been carried out by**, AJAYKUMAR MAJUKAR, NADEEM MD SALIM NAKKASHI** & **DARSHAN BELGAONKAR** are bonafide student of KLE Dr. M S Sheshgiri College of Engineering & Technology Belgaum, in partial fulfilment for the award of Bachelor of Engineering in Computer Science branch of the Visvesvaraya Technological University Belagavi, during the academic year 2018-19. It is certified that all corrections/suggestions indicated have been incorporated in the report.

GUIDE	HOD	PRINCIPAL
Prof. Savita Bakare	**Prof. B. A. Patil**	**Dr. Basavaraj Katageri**

EXTERNAL VIVA

Examiner:	**Signature with date**
1________________	
2________________	

DECLARATION

We hereby declare that the Project work entitled " **DETECTING PHISHING WEBSITE USING MACHINE LEARNING**" has been independently carried out by students mention below, under the guidance of **Prof. Savita Bakare**, Department of Computer Science Engineering, KLE Dr. MS. Sheshgiri College of Engineering & Technology, Belagavi in partial fulfilment of the requirement for the award of degree of Bachelor of Engineering in Computer Science Engineering at Visvesvaraya Technological University Belagavi.

We further declare that no part of it has been submitted for the award, or diploma any university or institute previously.

Place: Belagavi	**AJAYKUMAR MAJUKAR**	**2KL15CS005**
Date: 14/05/2019	**NADEEM MD SALIM NAKKASHI**	**2KL15CS040**
	DARSHAN BELGAONKAR	**2KL15CS017**

ACKNOWLEDGEMENT

Before we turn towards the project, we would like to add a few heartfelt words for the people who have been part of this project by supporting and encouraging us.

In particular, we would like to take this opportunity to express our honour, respect, deep gratitude and genuine regards to our Guide, **SAVITE BAKARE,** Assistant Professor, Department of Computer science for giving us all guidance required for our project apart from being a constant source of inspiration and motivation.

We are grateful to **PROF. B. A. PATIL**, Head of Department, Computer science Department, for providing us the necessary help and encouragement whenever needed which has resulted in the success of this Project.

We wish to express our sincere thanks to **Dr. BASAVARAJ KATAGERI**, Principal of K.L.E'S College of Engineering and Technology, Belgaum for providing a healthy environment in our college, which helped us in concentrating on our task.

We would like to sincerely thank **U. V. SOMANATTI** Associate Professor, Computer science Department for his kind help and valuable suggestions.

We owe special thanks to our Parents for their moral support and warm wishes, and finally we would express our appreciation to all our friends for their support which helped us to complete this project successfully.

ABSTRACT

Phishing can be described as a way by which someone may try to steal some personal and important information like login id’s, passwords, and details of credit/debit cards, for wrong reasons, by appearing as a trusted body. Many websites, which look perfectly legitimate to us, can be phishing and could well be the reason for various online frauds. These phishing websites may try to obtain our important information through many ways, for example: phone calls, messages, and pop-up windows. So, the need of the hour is to secure information that is sent online and one concrete way of doing so is by countering these phishing attacks. This project is focused on various Machine Learning algorithms (KNN, Decision Tree, Random Forest, SVM) aimed at predicting whether a website is phishing or legitimate. Machine learning solutions are able to detect zero-hour phishing attacks and they are better at handling new types of phishing attacks, so they are preferred over conventional methods.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

Internet has tremendously changed the way we work and communicate with each other. There are applications like e- mail, file transfer, voice communication, YouTube etc. that are available for users to use. But with its humongous success has come its weaknesses and vulnerabilities. The protocols and applications responsible for its success are being exploited by malicious users and hackers for gaining limelight. Phishing websites is one such area where administrators need new techniques and algorithms to protect naive users from getting exploited. Phishing is an attempt of fraud aimed at stealing our information, which is mostly done by emails. The ideal way to save ourselves from these phishing attacks is by observing such an attack. These phishing emails mostly come from trusted sources and try to retrieve our valuable information, for instance our passwords, bank details or even SSN. Many a times, these attacks come from sites where we have not even made any type of account. The procedure followed by phishers includes us reaching their website through the means of an email. In those emails, they make us click on a certain link that directs us to their websites. Asking for personal information is something that legitimate websites would hardly do. The looks of these phishing websites are quite similar to their respective legitimate ones and the only distinguishing factor is their URLs. Various initiations appearing from social websites, banks and online payment portals are used to deceive users. These phishing emails mostly contain links to websites that are affected with malware. Some of the ways to tackle these phishing attacks include generating awareness among people and training the users.

"Phish" is pronounced just like it's spelled, which is to say like the word "fish" — the analogy is of an angler throwing a baited hook out there (the phishing email) and hoping you bite. The term arose in the mid-1990s among hackers aiming to trick AOL users into giving up their login information. The "ph" is part of a tradition of whimsical hacker spelling, and was probably influenced by the term "phreaking," short for "phone phreaking," an early form of hacking that involved playing sound tones into telephone handsets to get free phone calls.

Some phishing scams have succeeded well enough to make waves:

- Perhaps one of the most consequential phishing attacks in history happened in 2016, when hackers managed to get Hillary Clinton campaign chair John Podesta to offer up his Gmail password.
- The "fappening" attack, in which intimate photos of a number of celebrities were made public, was originally thought to be a result of insecurity on Apple's iCloud servers, but was in fact the product of a number of successful phishing attempts.
- In 2016, employees at the University of Kansas responded to a phishing email and handed over access to their paycheck deposit information, resulting in them losing pay.

Figure 1 depicts different steps involved in a phishing attack

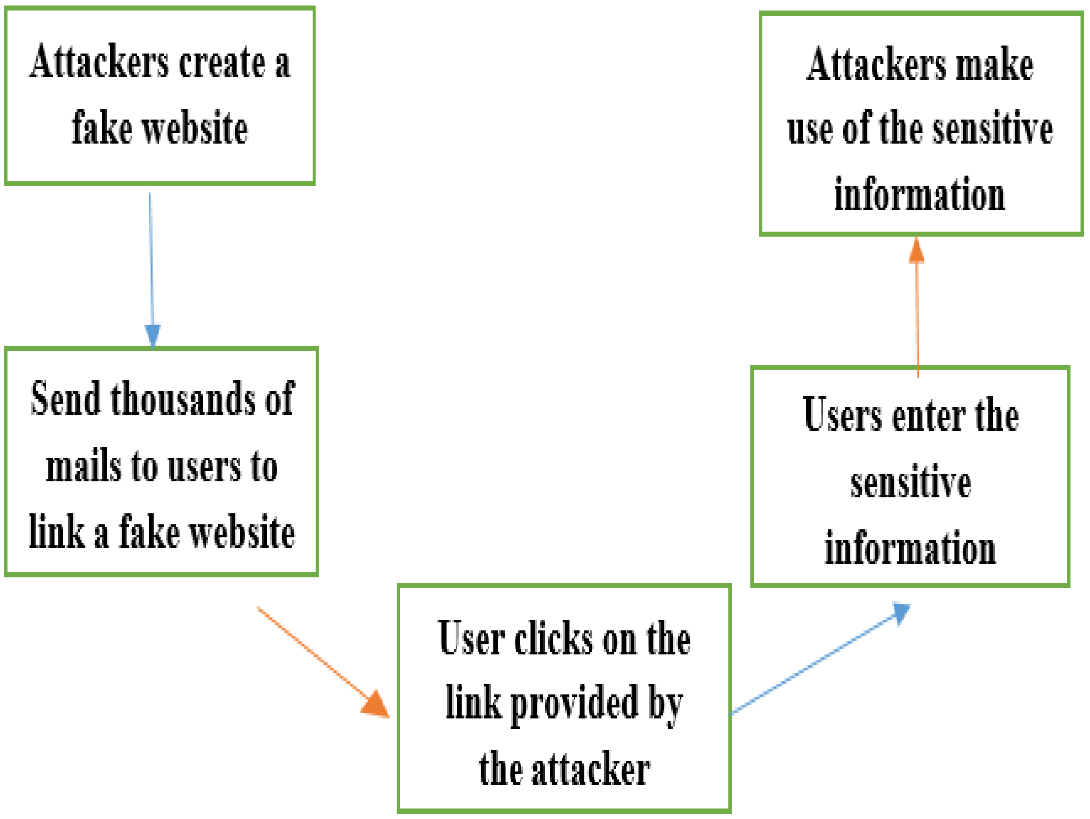

Attackers create a fake website
Send thousands of mails to users to link a fake website
User clicks on the link provided by the attacker
Users enter the sensitive information
Attackers make use of the sensitive information

Figure.1-: Block diagram of phishing attack

1) **Creating a fake website**: As part of phishing attack, attackers create a fake website which appears similar to original website. They use the main features of the original website such as logo, design of a website to create a fake website so that users cannot suspect such fake websites.

2) **Linking a fake website through email**: Once creation of the fake website is done, attackers send thousands of e-mails to multiple users and make email recipients(users) to click a URL which re-directs to the fake website.

3) **Clicking a malicious URL**: The users who were not aware of the malicious URL provided in the email, clicks it which directs to the fake website provided by the attackers. This is where the phishing attack begins.

4) **Entering sensitive information**: Once the user is redirected to the fake website, the sensitive information such as login credentials and other details are entered by the user in order to access the website created by the attacker.

5) **Compiling the stolen data and using it**: Once the user enters the sensitive information, all the sensitive data is collected so that the attacker can sell the data or use it for his/her own purpose.

1.1 Phishing Kit

The availability of phishing kits makes it easy for cyber criminals, even those with minimal technical skills, to launch phishing campaigns. A phishing kit bundles phishing website resources and tools that need only be installed on a server. Once installed, all the attacker needs to do is send out emails to potential victims. Phishing kits as well as mailing lists are available on the dark web. A couple of sites, Phishtank and OpenPhish, keep crowd-sourced lists of known phishing kits.

The Duo Labs report, phish in a Barrel, includes an analysis of phishing kit reuse. Of the 3,200 phishing kits that Duo discovered, 900 (27 percent) were found on more than one host. That number might actually be higher, however.

"Why don't we see a higher percentage of kit reuse? Perhaps because we were measuring based on the SHA1 hash of the kit contents. A single change to just one file in the kit would appear as two separate kits even when they are otherwise identical," said Jordan Wright, a senior R&D engineer at Duo and the report's author.

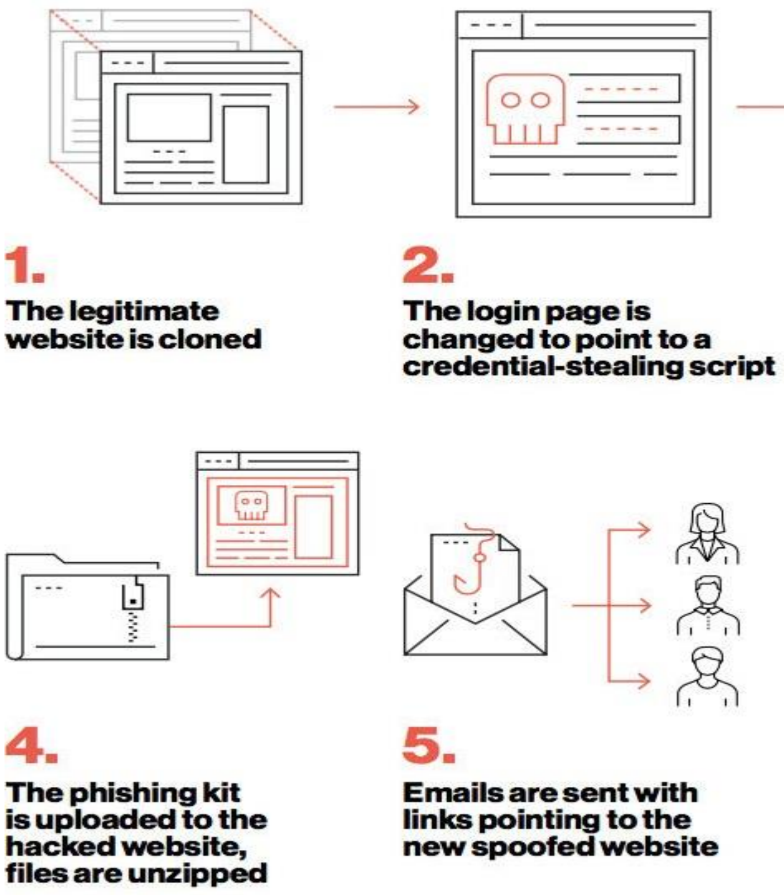

1.
The legitimate website is cloned
2.
The login page is changed to point to a credential-stealing script
3.
The modified files are bundled into a zip file to make a phishing kit
4.
The phishing kit is uploaded to the hacked website, files are unzipped
5.
Emails are sent with links pointing to the new spoofed website

Figure.2- Anatomy of a phishing kit

Analysing phishing kits allows security teams to track who is using them. "One of the most useful things we can learn from analysing phishing kits is where credentials are being sent. By tracking email addresses found in phishing kits, we can correlate actors to specific campaigns and even specific kits," said Wright in the report. "It gets even better. Not only can we see where credentials are sent, but we also see where credentials claim to be sent from. Creators of phishing kits commonly use the 'From' header like a signing card, letting us find multiple kits created by the same author."

1.2 Types of Phishing

Phishing attacks are broadly classified into four categories such as Phishing, Spear Phishing, Whale phishing and Clone Phishing.

Phishing: A phishing technique where an attacker impersonates a trustworthy entity in order to obtain sensitive information such as usernames, passwords, bank account number etc.

Spear phishing

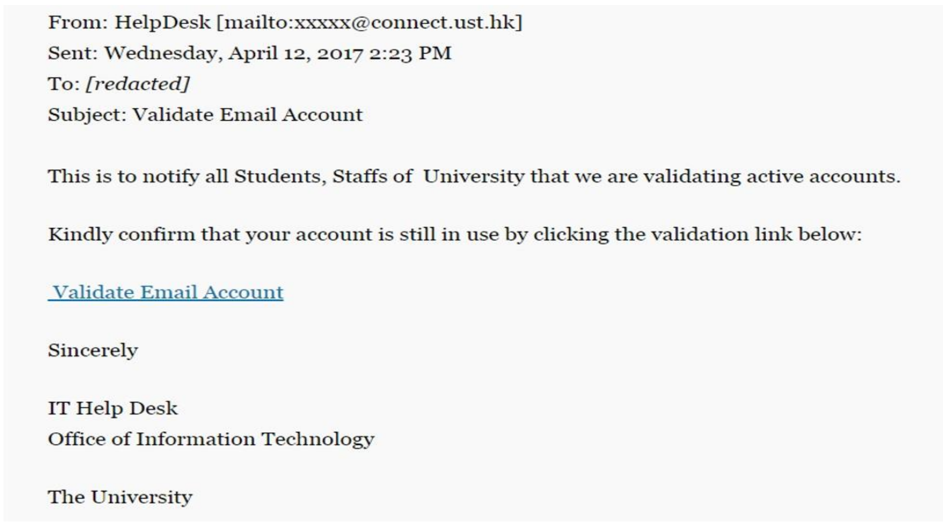
When attackers try to craft a message to appeal to a specific individual, that's called spear phishing. (The image is of a fisherman aiming for one specific fish, rather than just casting a baited hook in the water to see who bites.) Phishers identify their targets (sometimes using information on sites like LinkedIn) and use spoofed addresses to send emails that could plausibly look like they're coming from co-workers. For instance, the spear phisher might target someone in the finance department and pretend to be the victim's manager requesting a large bank transfer on short notice.

Whale phishing

Whale phishing, or whaling, is a form of spear phishing aimed at the very big fish — CEOs or other high-value targets. Many of these scams target company board members, who are considered particularly vulnerable: they have a great deal of authority within a company, but since they aren't full-time employees, they often use personal email addresses for business-related correspondence, which doesn't have the protections offered by corporate email.

Clone Phishing

A phishing technique which mimics a legitimate email account by using a genuine email and changing links.



From: HelpDesk [mailto:xxxxx@connect.ust.hk]
Sent: Wednesday, April 12, 2017 2:23 PM
To: *[redacted]*
Subject: Validate Email Account

This is to notify all Students, Staffs of University that we are validating active accounts.

Kindly confirm that your account is still in use by clicking the validation link below:

Validate Email Account

Sincerely

IT Help Desk
Office of Information Technology

The University

Figure.3- Malicious Web Links embedded in emails

CHAPTER 2

Literature Survey

The authors in this paper [1] presents a study on using a concept feature to detect web phishing problem. The features introduced in Carnegie Mellon Anti-phishing and Network Analysis Tool (CANTINA), they applied additional domain top-page similarity feature to a machine learning based phishing detection system. Features like age of domain, known images, suspicious URL etc.

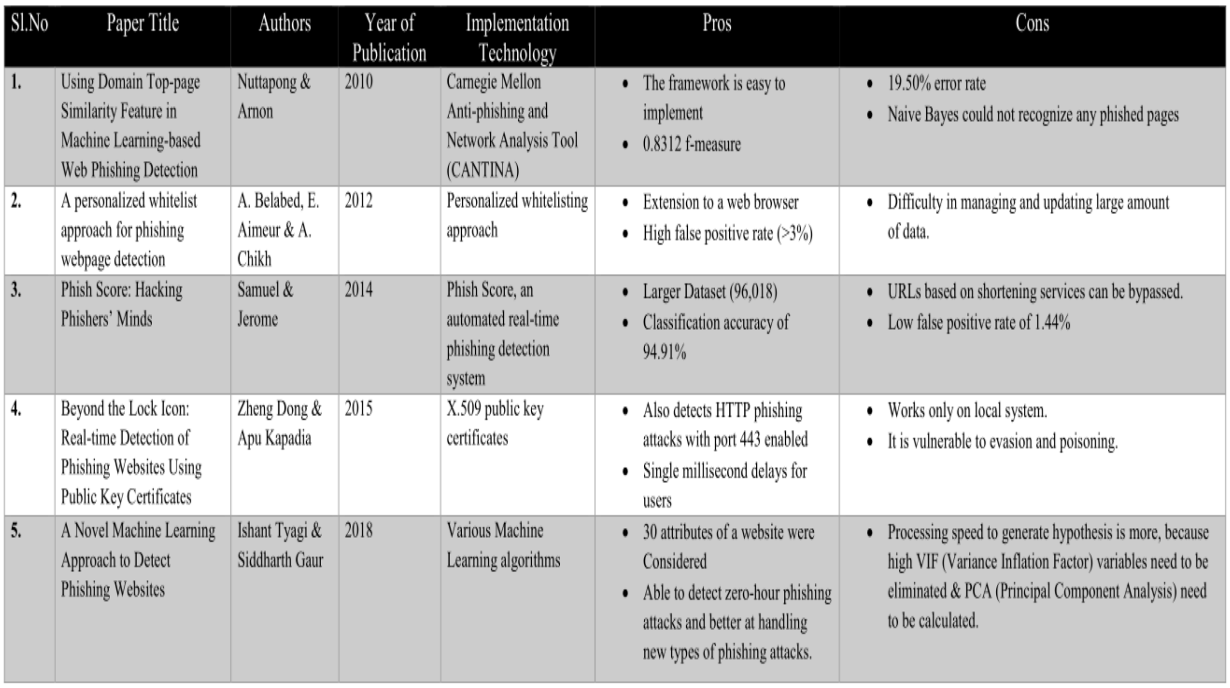
The authors in this paper [2] presents an automatic approach detecting phishing attacks. Their approach combines a personalized whitelisting approach with machine learning techniques. The whitelist is used as filter that blocks phish web pages used to imitate innocuous user behaviour. The phishing pages that are not blocked by the whitelist pass are further filtered using a Support Vector Machine classifier designed and optimized to classify these threats.

In this paper [3] they introduce Phish Score, an automated real-time phishing detection system. They observed that phishing URLs usually have few relationships between the part of the URL that must be registered (upper level domain) and the remaining part of the URL (low level domain, path, query). Hence, they define this concept as intra-URL relatedness and evaluate it using features extracted from words that compose a URL based on query data from Google and Yahoo search engines. These features are then used in machine learning based classification to detect phishing URLs from a real dataset.

In this research paper [4], the authors have used machine-learning algorithms to detect phishing websites using features from X.509 public key certificates. They have collected and tested certificates from all confirmed phishing websites associated with Phish Tank entries, regardless of whether HTTPS was included in the listed URL. They illustrate that their certificate-based approach greatly increases the difficulty of masquerading undetected for phishers, with single millisecond delays for users. They further show that this approach works not only against HTTPS-enabled phishing attacks, but also detects HTTP phishing attacks with port 443 enabled.

This paper [5] is focused on various Machine Learning algorithms aimed at predicting whether a website is phishing or legitimate. The algorithms used are Decision Tree, Random Forest, Generalized Linear Model. Machine learning solutions are able to detect zero-hour phishing attacks which are better at handling new types of phishing attacks, so they are preferred in prediction a website to be phishing or legitimate. The processing speed to generate hypothesis is more, because high VIF (Variance Inflation Factor) variables need to be eliminated & PCA (Principal Component Analysis) need to be calculated. Hence, we are to trying to optimize the time required to generate the hypothesis by implementing different algorithms to get predicted accuracy.

Table.1- Literature Survey



Sl.No	Paper Title	Authors	Year of Publication	Implementation Technology	Pros	Cons
1.	Using Domain Top-page Similarity Feature in Machine Learning-based Web Phishing Detection	Nuttapong & Arnon	2010	Carnegie Mellon Anti-phishing and Network Analysis Tool (CANTINA)	• The framework is easy to implement • 0.8312 f-measure	• 19.50% error rate • Naive Bayes could not recognize any phished pages
2.	A personalized whitelist approach for phishing webpage detection	A. Belabed, E. Aimeur & A. Chikh	2012	Personalized whitelisting approach	• Extension to a web browser • High false positive rate (>3%)	• Difficulty in managing and updating large amount of data.
3.	Phish Score: Hacking Phishers' Minds	Samuel & Jerome	2014	Phish Score, an automated real-time phishing detection system	• Larger Dataset (96,018) • Classification accuracy of 94.91%	• URLs based on shortening services can be bypassed. • Low false positive rate of 1.44%
4.	Beyond the Lock Icon: Real-time Detection of Phishing Websites Using Public Key Certificates	Zheng Dong & Apu Kapadia	2015	X.509 public key certificates	• Also detects HTTP phishing attacks with port 443 enabled • Single millisecond delays for users	• Works only on local system. • It is vulnerable to evasion and poisoning.
5.	A Novel Machine Learning Approach to Detect Phishing Websites	Ishant Tyagi & Siddharth Gaur	2018	Various Machine Learning algorithms	• 30 attributes of a website were Considered • Able to detect zero-hour phishing attacks and better at handling new types of phishing attacks.	• Processing speed to generate hypothesis is more, because high VIF (Variance Inflation Factor) variables need to be eliminated & PCA (Principal Component Analysis) need to be calculated.

CHAPTER 3

Problem Identification

Phishing URLs considers as the fastest rising online crime method used for stealing personal financial data and perpetrating identity theft. Individuals who respond to phishing URL, and input the requested financial or personal information into e-mails, websites, or pop-up windows put themselves and their institutions at risk.

The Microsoft Consumer Safety Index survey showed that the annual worldwide impact of phishing email was US $5 billion. On the other hand, the cost of repairing their impact is US $6 billion (MCSI reveals the impact of poor online safety behaviours).

With the massive work exists for phishing detection task, there is no set of features that has been determined as the best to detected phishing. Moreover, the same nondeterministic scenario is applied for the underling classification algorithm. Finally, there is a need to keep on enhancing the accuracy of the detection techniques. Overall the problems carried out in this project are as following:

- How to determine the best set of features to be used with phishing detection.
- How to select the best classification algorithm to be used for phishing detection.
- How to enhance the performance of the best selected features and classifiers.
- How to integrate multiple classification algorithms for phishing detection and to evaluate such integration.

CHAPTER 4

Objectives

- To carry out an exploratory analysis of the Phishing Websites Data Set and an interpretation of it.
- To determine and evaluate the best set of features to be used for phishing detection.
- To create a new dataset which has recent websites entries to get a better accuracy.
- To determine the best classification algorithm for phishing detection.
- To distinguish the phishing websites from the legitimate websites and ensure secure transactions to users.

CHAPTER 5

System Design

5.1 System Architecture

This section focuses on the system architecture of the proposed system in detecting the phishing URLs. The main goal of the proposed system is to detect an URL which is provided as input by the user as a phished, suspicious or legitimate URL. The system design involves designing a User Interface through which user inputs an URL and thereafter, the system displays the output results to the user. Once the input URL is submitted, the system extracts the website features using python standard built-in functions and collect all features which would be used in classification phase for classifying the input URL.

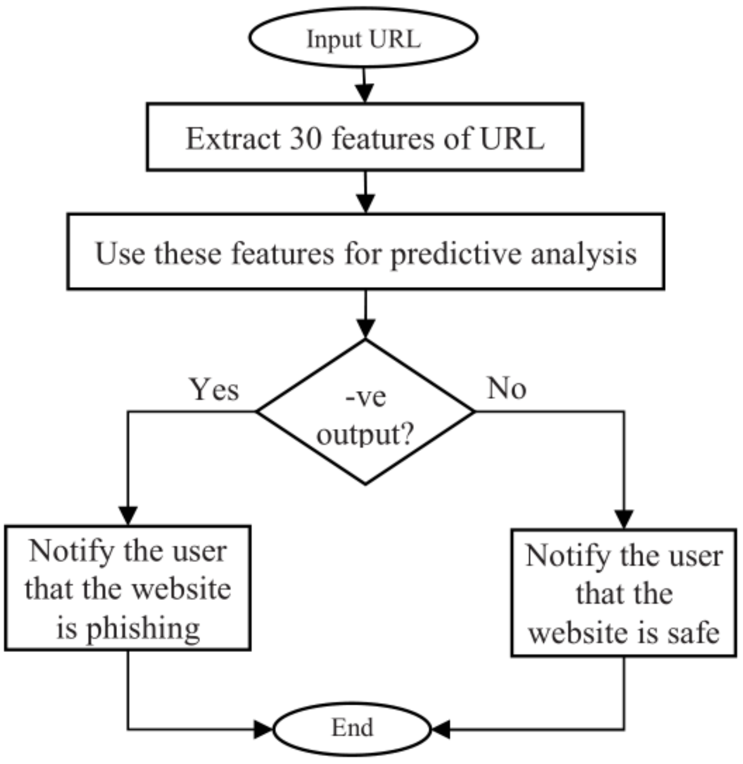

Input URL
Extract 30 features of URL
Use these features for predictive analysis
-ve output?
Yes
No
Notify the user that the website is phishing
Notify the user that the website is safe
End

Figure.4 - System Architecture of Phishing Detection System

The following are the steps included in the system architecture shown in Figure 5.1.

1. **User Input**: As part of first step, the user inputs a URL (either Phishing or a Legitimate URL). Once the URL is fed to the system, the system extracts features such as length of URL, HTTP with SSL Check. These features give a brief overview on category of URL thereby increasing the response time of the system.

2. **Feature Extraction**: In this step, all the relevant features of the URLs are extracted which are used to differentiate between phishing URLs and legitimate URLs. A URL feature is classified into three groups such as Address-bar based features, Abnormal features, HTML and JavaScript based features and Domain based features. These features will be later discussed in section 5.2.

3. **Predictive Analysis**: The features which are extracted from the previous step are subjected to different heuristics. A total of 30 features will be used to determine whether a URL is a phished, suspicious or legitimate one. Based on the features extracted, the proposed rules are applied in order to categorize a URL. These features will be discussed in detail in section 5.3.

4. **Evaluation**: The results of the classification are evaluated and the user is notified whether the given URL is Phished or legitimate.

5.2 Categories of URL Features

The URL features [16] are grouped into 4 categories as shown below.

- Address bar related features
- Abnormal based features
- HTML and JavaScript based features
- Domain based features

1. Address bar-related features: The features which are related to the address of an URL are referred as address bar-related features. These includes the length of the host URL, number of dots and slashes, special characters, HTTP and SSL check, @symbol, and IP Address.

a. Length of the host URL: URL is an alphanumeric string which is used to access the network resources on World Wide Web (WWW). The URL is a combination of network protocol, hostname and the path. The length of a hostname of a URL is one of the key features to be extracted while detecting the phishing URLs.

b. Number of dots and slashes: Sometimes, URL consists of multiple domains. The subdomains are part of the domain names which further narrow down the hierarchy of Domain Name Systems. The number of dots and slashes which exists in a URL determines the number of subdomains in the URL to verify whether the URL is phishing, legitimate or suspicious.

c. Special Characters: A URL contains special characters such as #, -, in order to differentiate terms in hostname and distinguish between the hostname and the domain name of a particular website. Certain special characters are used by the attackers to trick the web users to access the links with these characters in order to perform phishing attacks.

d. HTTP with SSL Check: Certain URLs use transport layer security to protect the URL from the attacks. The HTTPS protocol adds a security layer in order to transfer the sensitive information across the network without any issues. So, in order to determine whether a URL is legitimate or not, parameters such as HTTPS, authenticity of certificate and age of the certificate plays a vital role.

e. @ Symbol: The @ symbol is used by attackers to make the web browser ignore everything prior to it and redirects the user to the link typed after it.

f. IP Address: An IP Address is a unique identifier given to an identify a computer on the network. Attackers use the IP address instead of the domain name to trick the web users. Any legitimate URL is formed by using the hostname and pathname but not using IP address and path name.

2.Abnormal based features: The URL features which relates to anomalies or discrepancies between the W3C objects and Web Identity are known as abnormal based features. These features are mostly related to the source code of the web page. These features will play an important role in identifying the phishing websites. These features include Request URL (RURL), URL of an anchor (AURL), Server Form Handler (SFH)

a. Request URL: For most of the legitimate websites, the external objects such as external scripts, CSS, images and other attachments are tied to their own domain. So, the Request URL feature can be easily used to categorize the websites by checking whether the external files are linked to the original domain or not.

b. Anchor of a URL: This feature is similar to the Request URL. This feature verifies that all the anchors in a specific web page should be pointed to same domain on the web page itself. In this way, all the anchors in a tested URL can be verified to check whether that particular website has a phishing attack or not.

c. Server Form Handler: For any webpages or websites which need authentication or authorization, a server form with username and password are to be filled in order to access that particular site. A server form handler is served as an important feature to differentiate phishing sites from those of the legitimate websites which takes the following form.

<form action= “/login/login.jsp” method=”post” target=”_login”>

The above form tag is used in server-side script to perform an action based on the user’s navigation. The “action” in the above tag describes the path and “method” describes the type of HTTP method used in handling a page request.

3. HTML and JavaScript based Features: The features which are related to HTML tags and JavaScript functions are treated as HTML and JavaScript based features. These features include Redirect Page, Disabling Right Click and Using Pop-up window and onMouseOver.

a. Redirect Page: Redirecting a web page is a technique of navigating the web users to a different webpage other than the requested page. Many attackers use open redirect pages found on the webpages to redirect the link to their illegitimate sites. So, the number of redirects used in a web page determines whether the website is a legitimate or not.

b. onMouseOver feature: The onMouseOver event is triggered in JavaScript whenever the mouse pointer overs an element. The attackers use the JavaScript code to display an unauthentic URL in the status bar of a web page in order to trick the web users. So, this feature is used to determine whether the status bar is changed or not when onMouseOver event is triggered.

c. Right Click: This feature is similar to the onMouseOver feature. The attackers use the JavaScript code to hide the source code from the web users by disabling the right click function. Using this feature, one can easily distinguish the phishing websites to that of a legitimate website.

4. Domain based features: The URL features related to domain name-based information is known as Domain based features. These features include Alexa Page Rank, Age of the Domain, DNS Record and Website Traffic.

a. Alexa Page Rank: The ranking of a website indicates the popularity of the website and therefore, many users access it. It is to be understood that attackers maintain the phishing URLs for a certain amount of time. Once the link or URLs are expired, they no more appear on Internet. The Alexa page ranking is one among the many features used to detect the malicious URLs.

b. Age of the Domain: This feature gives the approximate age of that particular domain of a website. The more age the website is, the more legitimate it is. So, this feature is used in detecting whether a website is legitimate, suspicious or phished based on the website age extracted from WHOIS Database.

c. DNS Record: DNS Records are mapping filenames which informs the DNS server about the IP address associated with each website on the Internet. Many legitimate websites contain the Owner of the Domain, Date and Time Created details which differentiates the legitimate sites to that of phished websites.

d. Website Traffic: In general, there are more visits to the legitimate websites. Due to this, there is high traffic as they are frequently visited. In contrast to this, phishing sites can be identified easily based on website traffic as they have no web traffic.

5.3 Heuristics used in the System

A heuristic is a method or rule which is used in problem solving for practical purposes. Here in this system, the heuristics are stated based on which the proposed system is built which classifies a website or URL. Each heuristic is grouped based on the feature categories which was described in section 5.2.

Heuristic 1: Length of host URL

If length of the URL is < 54 → Legitimate

Else if length of the URL is >= 54 and <= 75 → Suspicious

Otherwise → Phishing

Description: This association rule indicates that if the length of the URL is more than 75, the URL is categorized as a phishing URL.

Heuristic 2: Number of dots and slashes

If the number of the dots in the domain part of the URL < 3 → Legitimate

Else if number of the dots in the domain part = 3 → Suspicious

Otherwise → Phishing

Description: The number of the dots in the domain part indicates the number of subdomains used in the URL. Therefore, if the number of the sub domains are more than 3, the URL is classified as phishing.

Heuristic 3: Using @ symbol

If the domain name of the URL includes @ symbol → Phishing

Otherwise → Legitimate

Description: This rule [19] indicates that a URL is denoted as phishing URL if the URL contains any special characters such as @.

Heuristic 4: Using Special Character

If URL contains any of special characters such as dash, underscore, comma, semicolon $\rightarrow$ Phishing

Otherwise $\rightarrow$ Legitimate

Description: This rule indicates that when special characters such as dash, underscore, comma, semicolon are part of the input URL, then it is a phishing URL.

Heuristic 5: Using HTTPS with Secure Socket Layer

If the URL contains HTTPS with Trusted SSL $\rightarrow$ Legitimate

Else if the URL contains HTTPS with Untrusted SSL $\rightarrow$ Suspicious

Otherwise $\rightarrow$ Phishing

Description: Most of the legitimate URL contains HTTPS protocol with trusted SSL. Also, it is deducted that a phishing URL doesn't contain HTTPS as the attackers don't add an extra layer of security SSL to the malicious phishing URLs.

Heuristic 6: Using IPAddress

If the URL contains IP Address as part of it $\rightarrow$ Phishing

Otherwise $\rightarrow$ Legitimate

Description: If an IP Address is part of the URL, it is an indication of someone is trying to access the sensitive information through a phishing attack. So, if a URL contains IP Address, the system will mark it as phishing else legitimate.

Heuristic 7: Using Request URL

If the web page contains 22% of the request URLs loaded from other domains
$\rightarrow$ Legitimate

Else if the web page contains request URLs% between 22 and 61 Suspicious

Otherwise $\rightarrow$ Phishing

Description: According to the security analysts, the legitimate URLs contains only 22% of the content such as images, CSS etc., loaded from the other domains. Whereas Phishing Websites contains more % of the content loaded from multiple domains.

Heuristic 8: Using Anchor URL

If less than 31% anchor part of the URL is connected to other domains
$\rightarrow$ Legitimate
Else if % anchor part of the URL is in between 31 and 67 Suspicious

Otherwise $\rightarrow$ Phishing

Description: This rule indicates that a smaller number of the objects loaded from different domains in anchor of an URL, the less malicious the URL.

Heuristic 9: Using Server Form Handler (SFH)

If the SFH of a URL contains blank or empty $\rightarrow$ Phishing

Else if SFH of a URL belongs to a distinct domain $\rightarrow$ Suspicious

Otherwise $\rightarrow$ Legitimate

Description: This rule indicates that phishing URLs doesn't contain SFH because the information submitted by the web users are not handled by the external domains. Based on the SFH handler, the URL is categorized accordingly.

Heuristic 10: Using Redirect Page

If the number of the redirects in the website <= 1 → Legitimate

Else if the number of the redirects in the webpage is in between 1 and 4 → Suspicious

Otherwise → Phishing

Description: This rule indicates that a greater number of the redirects in a web page, more malicious the URL as an authentic code using HTML or any server-side script contains redirection to a single page.

Heuristic 11: Using onMouseOver

If an onMouseOver event is triggered which changes the status bar → Phishing

Else if it doesn't change the status bar → Suspicious Otherwise → Legitimate

Description: This rule indicates that attackers use the JavaScript function to make sure that fake URL is displayed in the status bar.

Heuristic 12: Using Right Click

If in a web page, the JavaScript Right Click code is disabled → Phishing

Else if the web page produces an alert on right click → Suspicious

Otherwise → Legitimate

Description: A legitimate webpage never creates an alert or disable any functionality.

Heuristic 13: Age of Domain

If the age of the domain is greater than or equal to 6 months → **Legitimate**

Otherwise → **Phishing**

Description: The age of the domain can be extracted from a WHOIS database. Therefore, the domain of all legitimate URLs has an age greater than 6 months which indicates that the URL is legitimate.

Heuristic 14: DNS Record

If there exists a DNS record for an input URL → **Legitimate**

Otherwise → **Phishing**

Description: If a URL has to be classified as a legitimate one, it should have a DNS record else it is treated as phishing URL.

Heuristic 15: Alexa Page Rank

If a web page rank is less than 100000 → **Legitimate**

Otherwise → **Phishing**

Description: The web page rank is a key indicator which is governed by the factors such as the number of pages visited by the web users as well as the number of the visitors to a website or URL. Therefore, the less the page rank of a web site, the more legitimate the URL.

Heuristic 16: Website Traffic

If Website traffic of a URL is high → **Legitimate**

Otherwise → **Phishing**

Description: There exists a high traffic for legitimate URLs because of frequent visits by the users. If there are no frequent visits to the URL, the URL is marked as phishing.

Heuristic 18: Shortened URL

Description:

Phishers often shorten a URL on the “World Wide Web” i.e. making a website URL much shorter but it still leads to the required website. This can be done by using a “HTTP Redirect” on a domain name that is short, which directs to the webpage with long URL. An example is: “http://portal.hud.ac.uk/” which may be shortened to “bit.ly/19DXSk4”. So, if a website shortens itself, then it can be considered as phishing.

Heuristic 19: Double slash redirecting

Description:

The presence of “//” implies that the user will be directed to another website. What phishers do is that the put the address of their malicious website beyond the original “//” and users they redirect the users to their required webpage. So, finding out the location of “//” can be useful to find out whether we are being redirected or not. If the position of “//” is at 6th or 7th position (HTTP of HTTPS), then we can be assured that we are not being redirected.

Heuristic 20: Prefix suffix

Description:

Phishers add prefix of suffix which are separated by a dash (“-”) symbol in the domain name to mislead the users. Although, in reality, one can hardly find any “-” symbol in legitimate URLs. For instance, in the given weblinkhttp://www.Confirme-paypal.com/, phishers have added dash symbol between Confirme and paypal and its difficult for regular user to detect the trick.

Heuristic 21: Iframe tag

Description:

Another trick that phishers do is that they hide a webpage within a legitimate webpage by using “iframe” tag in the HTML script. So, phishers can make use of the “frameBorder” attribute which causes the browser to render a visual delineation.

Heuristic 22: Links in <Meta>, <Script>, and <Link> tags

Description:

Legitimate websites use <Meta> tags to show metadata about the HTML document, <Script> tags are used to create a client-side script, and <Link> tags to obtain other web resources. One can assume that these tags are linked to the same domain of the webpage.

So, if the percentage of “<Meta>”, “<Script>”, and “<Link>” tags is less than 17%, then we call our website legitimate. For it to be in category of suspicious, it must have its percentage less than 81 but at the same time greater than 17. If its percentage exceeds 81, then it falls in the category of phishing.

Heuristic 23: Domain registration length

Description:

Domain registration length should be at least one year as we are aware of the fact that trustworthy domains are regularly paid for several years in advance.

Heuristic 24: Using non-standard port

Description:

If a certain service (like HTTP) is up or down, then this feature comes into the picture. It is advised to merely open those ports only which are required. Some firewalls, Proxy and Network Address Translation (NAT) servers will, by default, bar almost every port and open the selected ones only. Opening all the ports puts user’s information in danger as the attacker may run service of its choice.

Heuristic 25: Abnormal URL

Description:

WHOIS database is used for pulling this feature. The identity is part of URL, for an innocent website. So, in a website, if URL doesn't contain host name, then it is considered phishing, and else it is acceptable.

Heuristic 26: Statistical based reports feature

Description:

This feature basically searches the domain name and IP address of the website and searches it among the "Top 10 Domains" as well as "Top 10 IPs". If any matches occur, the website is considered as phishing. The statistics were provided by PhishTank's statistical reports published in the years 2010 to 2012.

Heuristic 27: Sub Domain and Multi Sub Domains

Description:

This feature is based on the number of dots in the URL. For example: http://www.iitd.ac.in. Here the "in" is country-code Top Level Domain (CCTLD). The "ac" part is an abbreviation for "academics", the joined "ac.in" is called a Second-Level Domain (SLD) and "iitd" is the real name of domain. At first, we neglect (www.) part and then (CCTLD) (if present) from URL. At last, we count the remaining dots. If the count is greater than one, then the URL is marked as "suspicious" since it has one sub domain. Else if the count is greater than two, it is marked as "Phishing" as it will have multiple sub domains. Else it is legitimate if it has no sub domains.

Heuristic 28: Favicon

Description:

Many phishers use fake favicon but it can potentially expose them. A favicon is an icon affiliated with a particular webpage. Favicon is displayed as a pictorial reminder of website's identity in address bar by current users as graphical browsers and newsreaders. If there is a mismatch in the favicon of the domain and the URL in the address bar, then one can be assured that it is a phishing attempt.

Heuristic 29: Submitting Information to Email

Description:

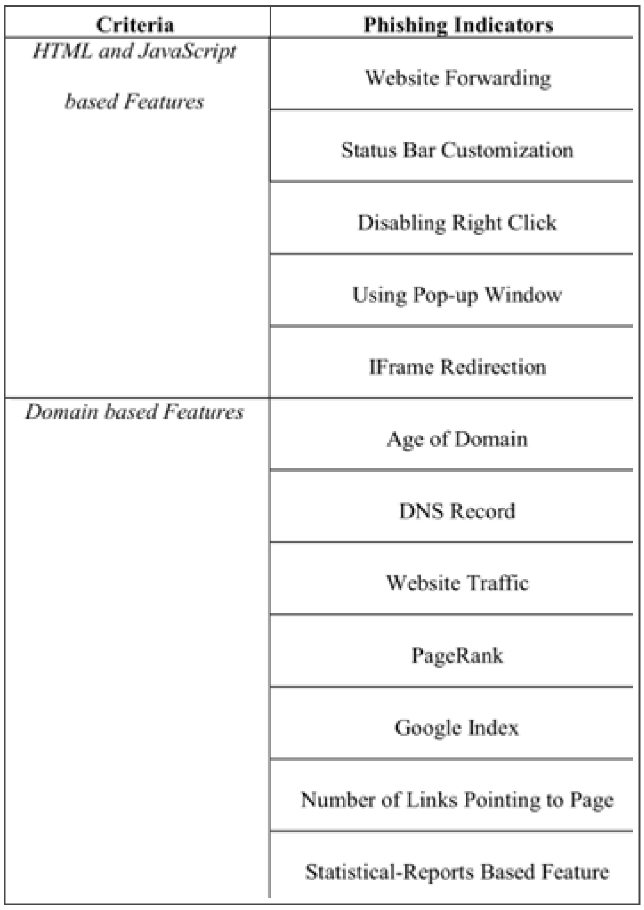
We often submit our personal information to various web forms. User's private information can be averted to attacker's email. This may be done in two ways: first by using mail () in PHP and second being "mailto". So, if either of these two functions is used, then the website can be phishing.

Heuristic 30: Using Pop-up Window

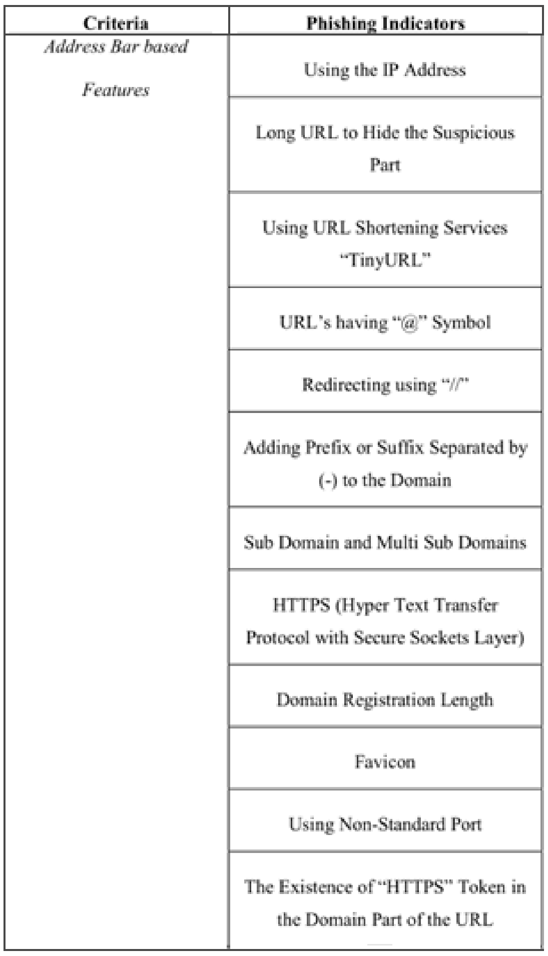
Description:

If a pop-up window in website asks for private information, then this website might as well be a phishing one as usually, legitimate does not ask their users to submit any important piece of information through a pop-up window.

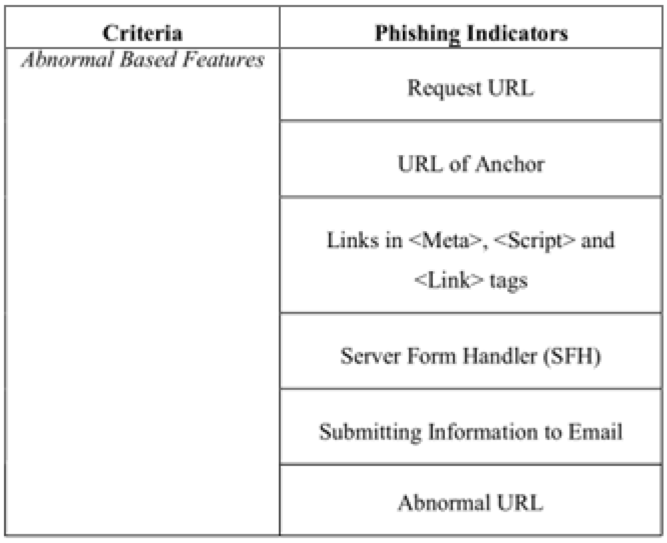
Table.2- 30 Features of the Dataset



Criteria	Phishing Indicators
HTML and JavaScript based Features	Website Forwarding
	Status Bar Customization
	Disabling Right Click
	Using Pop-up Window
	IFrame Redirection
Domain based Features	Age of Domain
	DNS Record
	Website Traffic
	PageRank
	Google Index
	Number of Links Pointing to Page
	Statistical-Reports Based Feature



Criteria	Phishing Indicators
Address Bar based Features	Using the IP Address
	Long URL to Hide the Suspicious Part
	Using URL Shortening Services "TinyURL"
	URL's having "@" Symbol
	Redirecting using "//"
	Adding Prefix or Suffix Separated by (-) to the Domain
	Sub Domain and Multi Sub Domains
	HTTPS (Hyper Text Transfer Protocol with Secure Sockets Layer)
	Domain Registration Length
	Favicon
	Using Non-Standard Port
	The Existence of "HTTPS" Token in the Domain Part of the URL



Criteria	Phishing Indicators
Abnormal Based Features	Request URL
	URL of Anchor
	Links in <Meta>, <Script> and <Link> tags
	Server Form Handler (SFH)
	Submitting Information to Email
	Abnormal URL

CHAPTER 6

Methodology

In this Section, the machine-learning algorithms used for classification has been explained.

K-nearest neighbors (KNN):

KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry. It is commonly used for its easy of interpretation and low calculation time.

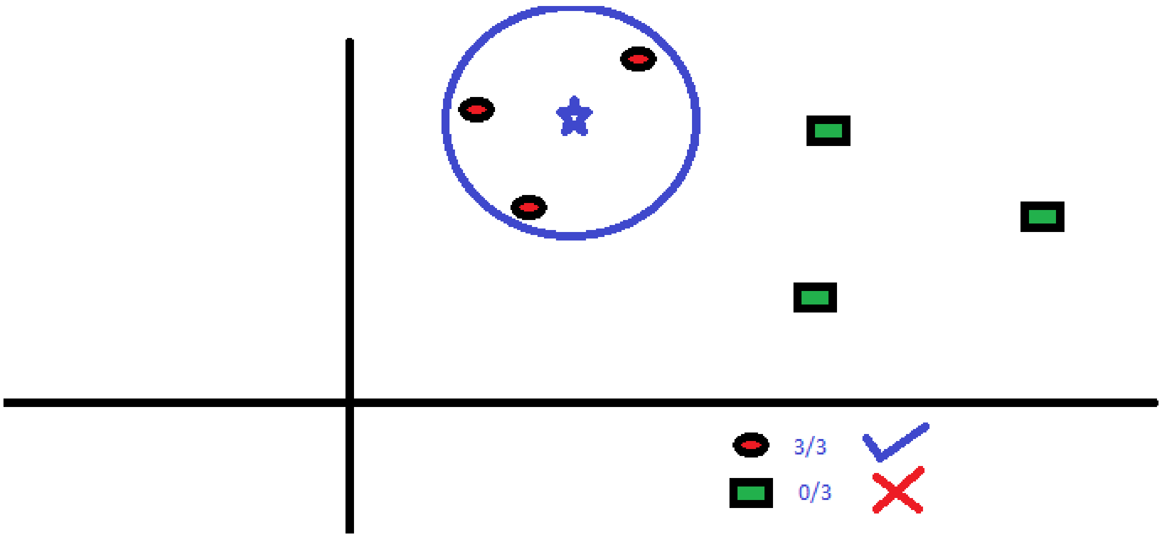


Figure.5- KNN Diagram

The three closest points to Blue Star are all Red Circle. Hence, with good confidence level we can say that the Blue Star should belong to the class Red Circle. Here, the choice became very obvious as all three votes from the closest neighbor went to Red Circle. The choice of the parameter K is very crucial in this algorithm.

First let us try to understand what exactly does K influence in the algorithm. If we see the last example, given that all the 6-training observation remain constant, with a given K value we can make boundaries of each class. These boundaries will segregate RC from GS. The same way, let’s try to see the effect of value “K” on the class boundaries. Following are the different boundaries separating the two classes with different values of K.

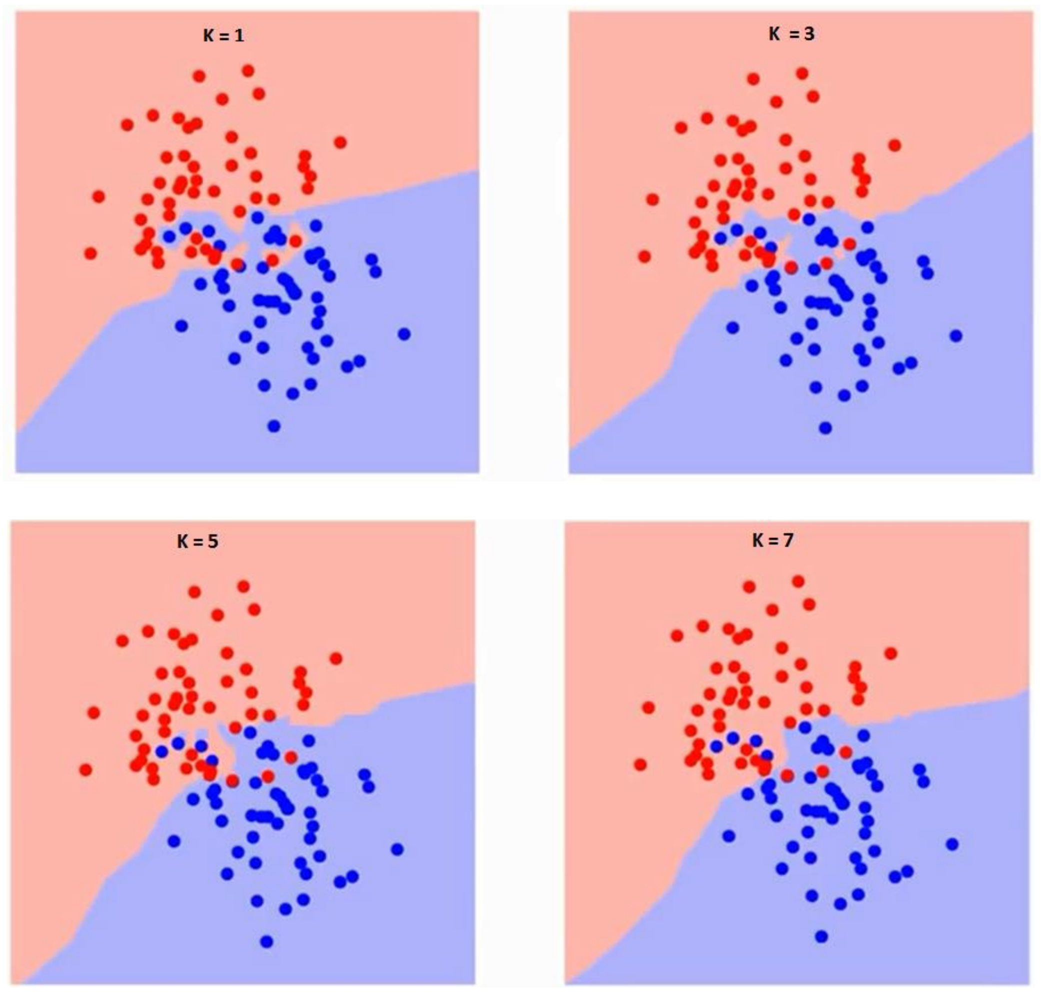


Figure.6- KNN Diagram 2

If you watch carefully, you can see that the boundary becomes smoother with increasing value of K. With K increasing to infinity it finally becomes all blue or all red depending on the total majority. The training error rate and the validation error rate are two parameters we need to access on different K-value. Following is the curve for the training error rate with varying value of K:

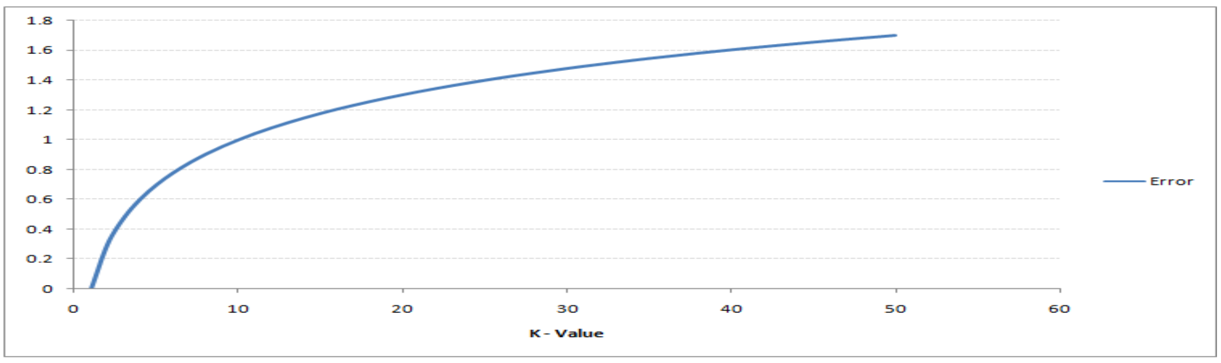


Figure.7- KNN Curve Graph

As you can see, the error rate at K=1 is always zero for the training sample. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with K=1. If validation error curve would have been similar, our choice of K would have been 1. Following is the validation error curve with varying value of K:

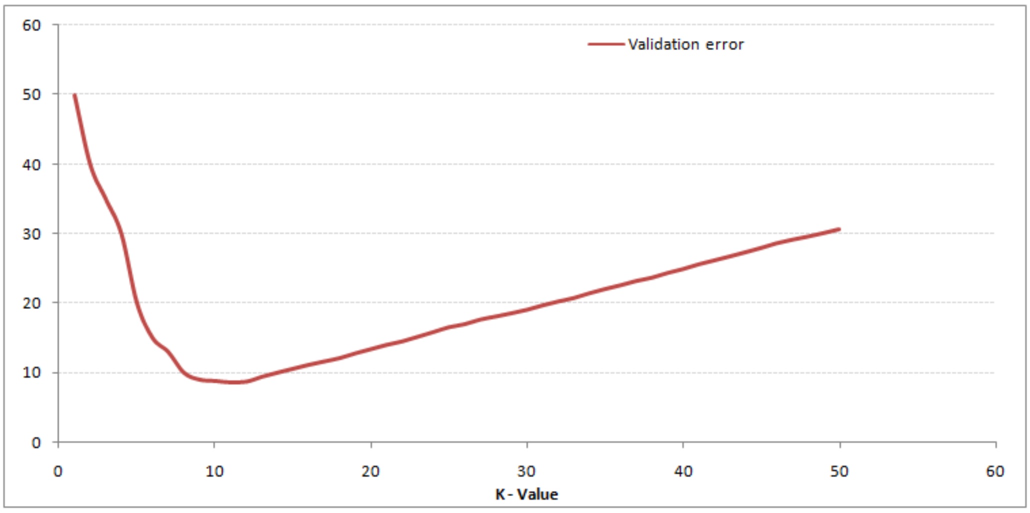


Figure.8- Validation Error curve of KNN

This makes the story clearer. At K=1, we were overfitting the boundaries. Hence, error rate initially decreases and reaches a minimal. After the minima point, it then increases with increasing K. To get the optimal value of K, you can segregate the training and validation from the initial dataset. Now plot the validation error curve to get the optimal value of K. This value of K should be used for all predictions.

Random Forest:

Random Forest adds randomness to the generation of decision trees. Instead of relying on one single decision tree to cover the entire dataset and features, this approach selects features and training data randomly from the given sets and constructs a series of decision trees based on these randomly selected inputs. The output of Random Forest is then calculated by the outputs of the contained decision trees. For a new data, each tree gives a classification. The forest chooses the classification having the most votes.

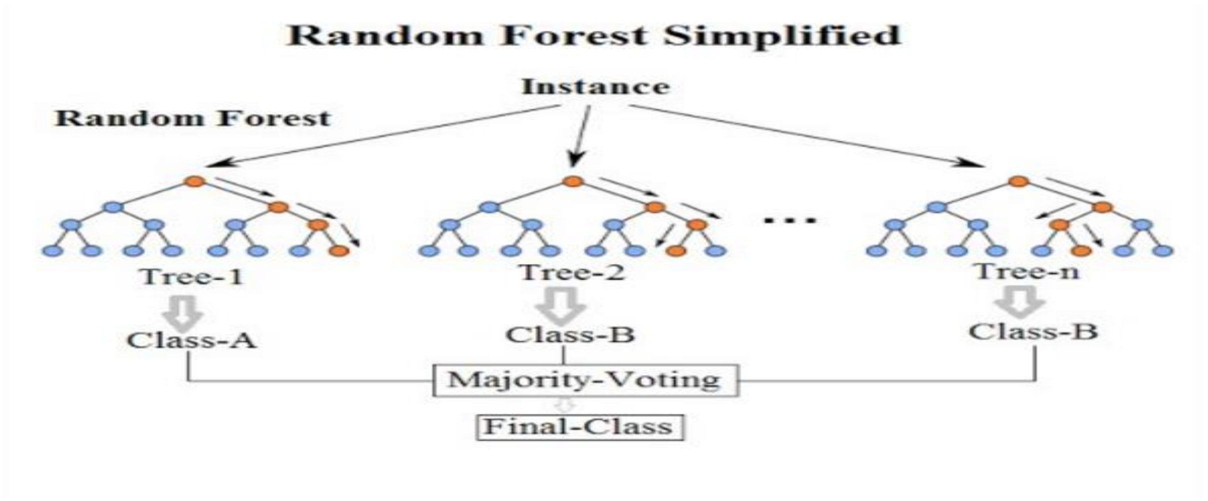


Figure.9- Random Forest Flowchart

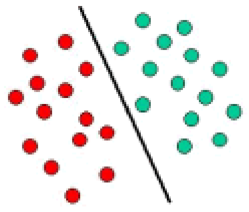
A decision tree is the building block of a random forest and is an intuitive model. We can think of a decision tree as a series of yes/no questions asked about our data eventually leading to a predicted class (or continuous value in the case of regression). This is an interpretable model because it makes classifications much like we do: we ask a sequence of queries about the available data we have until we arrive at a decision (in an ideal world).

The technical details of a decision tree are in how the questions about the data are formed. A decision tree is built by determining the questions (called splits of nodes) that, when answered, lead to the greatest reduction in Gini Impurity. What this means is the decision tree tries to form nodes containing a high proportion of samples (data points) from a single class by finding values in the features that cleanly divide the data into classes.

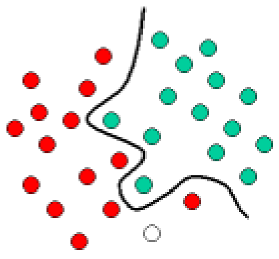
Support Vector Machine:

Support Vector Machine algorithm or SVM is used for classification of two groups. The machine follows the concept in such a way that nonlinear input vectors are mapped into a feature space with very large dimensions. In This features space there exists a linear decision-making boundary. The attempts are focused to choose a line with higher safety margin and in fact, the optimal dividing line. The equation is for finding the optimal line for data using a QP method which is useful in solving restricted problems. Unlike neural networks, support vector machine method is not stuck in a local maximum, and training them is also easier. They perform very well for high-dimensional data.

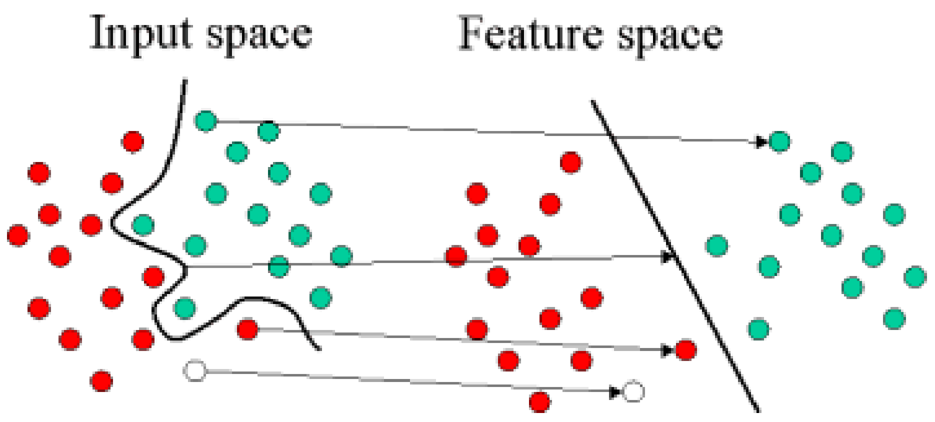
Support Vector Machines are based on the concept of decision planes that define decision boundaries. A decision plane is one that separates between a set of objects having different class memberships. A schematic example is shown in the illustration below. In this example, the objects belong either to class GREEN or RED. The separating line defines a boundary on the right side of which all objects are GREEN and to the left of which all objects are RED. Any new object (white circle) falling to the right is labelled, i.e., classified, as GREEN (or classified as RED should it fall to the left of the separating line).



The above is a classic example of a linear classifier, i.e., a classifier that separates a set of objects into their respective groups (GREEN and RED in this case) with a line. Most classification tasks, however, are not that simple, and often more complex structures are needed in order to make an optimal separation, i.e., correctly classify new objects (test cases) on the basis of the examples that are available (train cases). This situation is depicted in the illustration below. Compared to the previous schematic, it is clear that a full separation of the GREEN and RED objects would require a curve (which is more complex than a line). Classification tasks based on drawing separating lines to distinguish between objects of different class memberships are known as hyperplane classifiers. Support Vector Machines are particularly suited to handle such tasks.



The illustration below shows the basic idea behind Support Vector Machines. Here we see the original objects (left side of the schematic) mapped, i.e., rearranged, using a set of mathematical functions, known as kernels. The process of rearranging the objects is known as mapping (transformation). Note that in this new setting, the mapped objects (right side of the schematic) is linearly separable and, thus, instead of constructing the complex curve (left schematic), all we have to do is to find an optimal line that can separate the GREEN and the RED objects.



Decision Tree:

Decision tree builds classification or regression models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes. A decision node (e.g., Outlook) has two or more branches (e.g., Sunny, Overcast and Rainy). Leaf node (e.g., Play) represents a classification or decision. The topmost decision node in a tree which corresponds to the best predictor called root node. Decision trees can handle both categorical and numerical data.

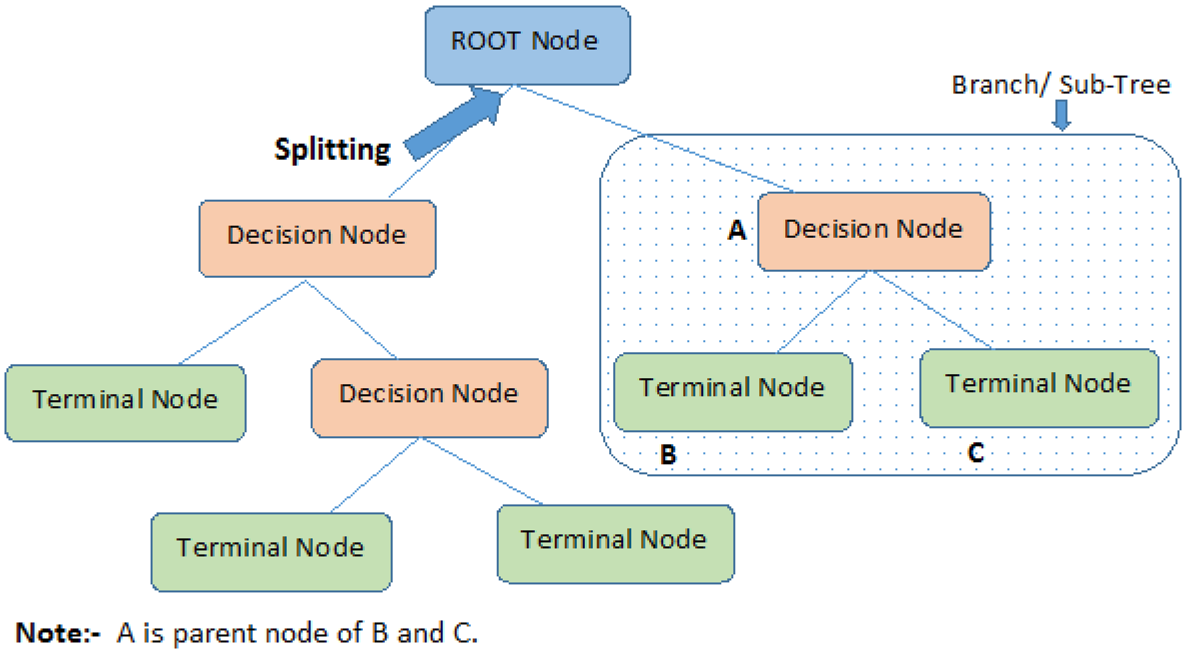


Figure.10- Decision Tree Flowchart

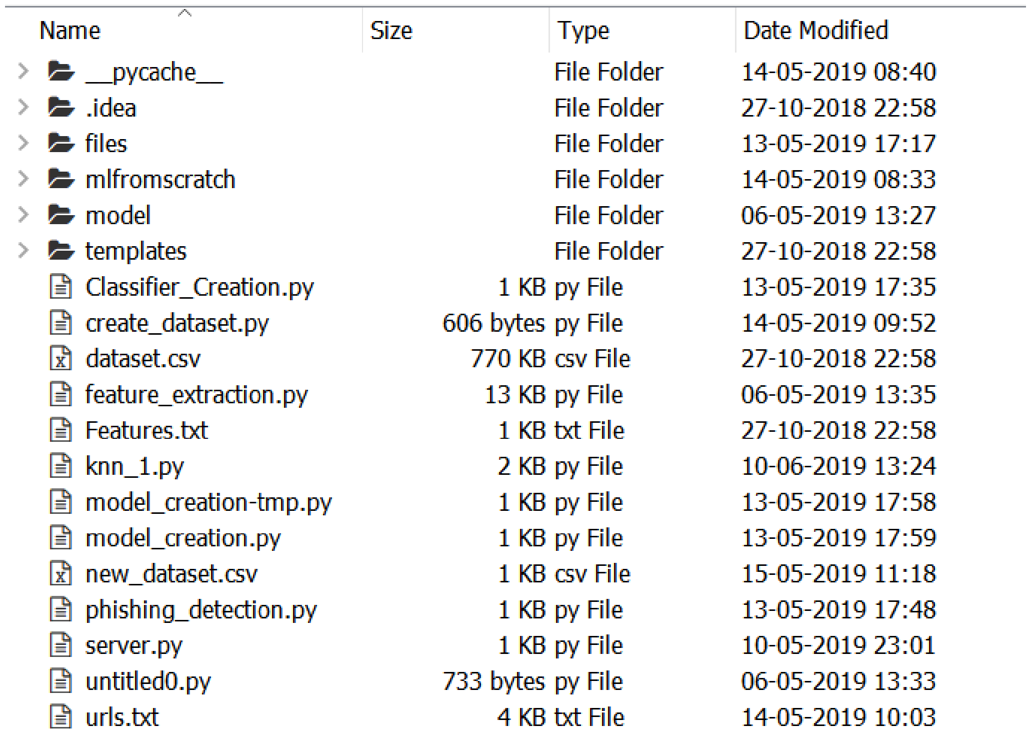
The core algorithm for building decision trees called ID3 by J. R. Quinlan which employs a top-down, greedy search through the space of possible branches with no backtracking. ID3 uses Entropy and Information Gain to construct a decision tree. In ZeroR model there is no predictor, in One R model we try to find the single best predictor, naive Bayesian includes all predictors using Bayes' rule and the independence assumptions between predictors but decision tree includes all predictors with the dependence assumptions between predictors

A decision tree is built top-down from a root node and involves partitioning the data into subsets that contain instances with similar values (homogenous). ID3 algorithm uses entropy to calculate the homogeneity of a sample. If the sample is completely homogeneous the entropy is zero and if the sample is an equally divided it has entropy of one.

CHAPTER 7

Implementation

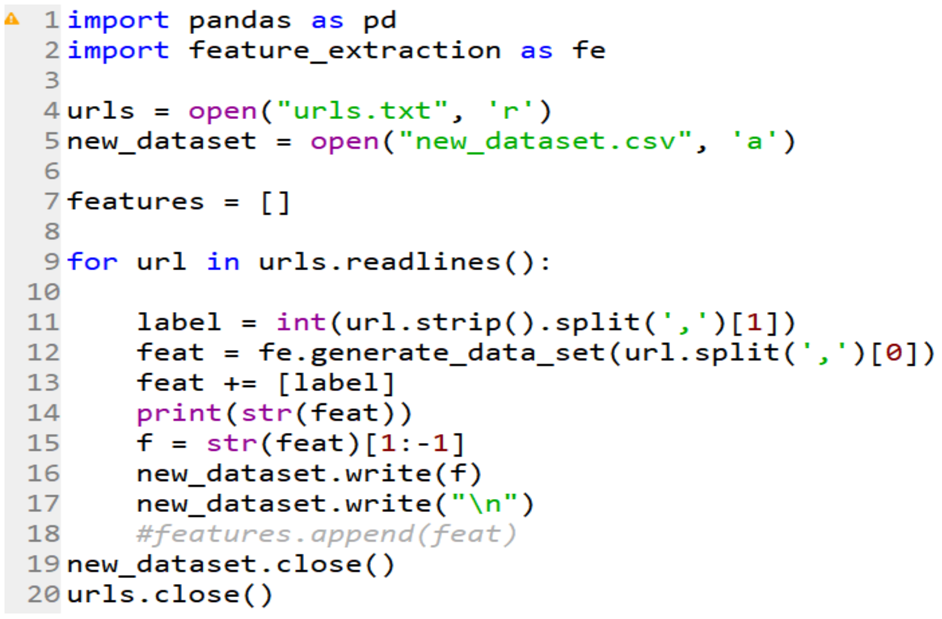
File Structure



Name	Size	Type	Date Modified
__pycache__		File Folder	14-05-2019 08:40
.idea		File Folder	27-10-2018 22:58
files		File Folder	13-05-2019 17:17
mlfromscratch		File Folder	14-05-2019 08:33
model		File Folder	06-05-2019 13:27
templates		File Folder	27-10-2018 22:58
Classifier_Creation.py	1 KB	py File	13-05-2019 17:35
create_dataset.py	606 bytes	py File	14-05-2019 09:52
dataset.csv	770 KB	csv File	27-10-2018 22:58
feature_extraction.py	13 KB	py File	06-05-2019 13:35
Features.txt	1 KB	txt File	27-10-2018 22:58
knn_1.py	2 KB	py File	10-06-2019 13:24
model_creation-tmp.py	1 KB	py File	13-05-2019 17:58
model_creation.py	1 KB	py File	13-05-2019 17:59
new_dataset.csv	1 KB	csv File	15-05-2019 11:18
phishing_detection.py	1 KB	py File	13-05-2019 17:48
server.py	1 KB	py File	10-05-2019 23:01
untitled0.py	733 bytes	py File	06-05-2019 13:33
urls.txt	4 KB	txt File	14-05-2019 10:03

Figure.11- File Structure of the Project

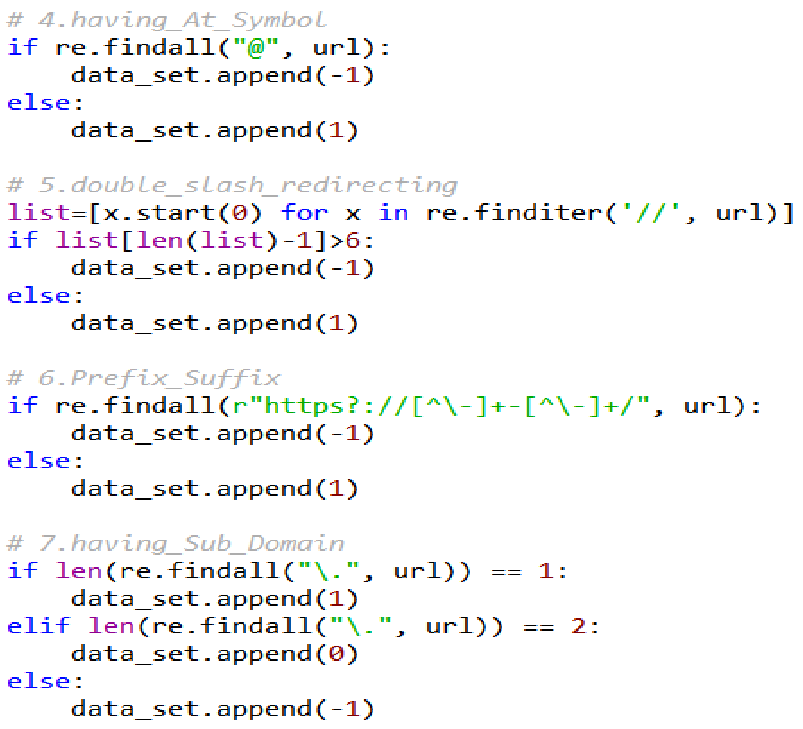
Dataset Creation



```python
import pandas as pd
import feature_extraction as fe

urls = open("urls.txt", 'r')
new_dataset = open("new_dataset.csv", 'a')

features = []

for url in urls.readlines():

    label = int(url.strip().split(',')[1])
    feat = fe.generate_data_set(url.split(',')[0])
    feat += [label]
    print(str(feat))
    f = str(feat)[1:-1]
    new_dataset.write(f)
    new_dataset.write("\n")
    #features.append(feat)
new_dataset.close()
urls.close()
```

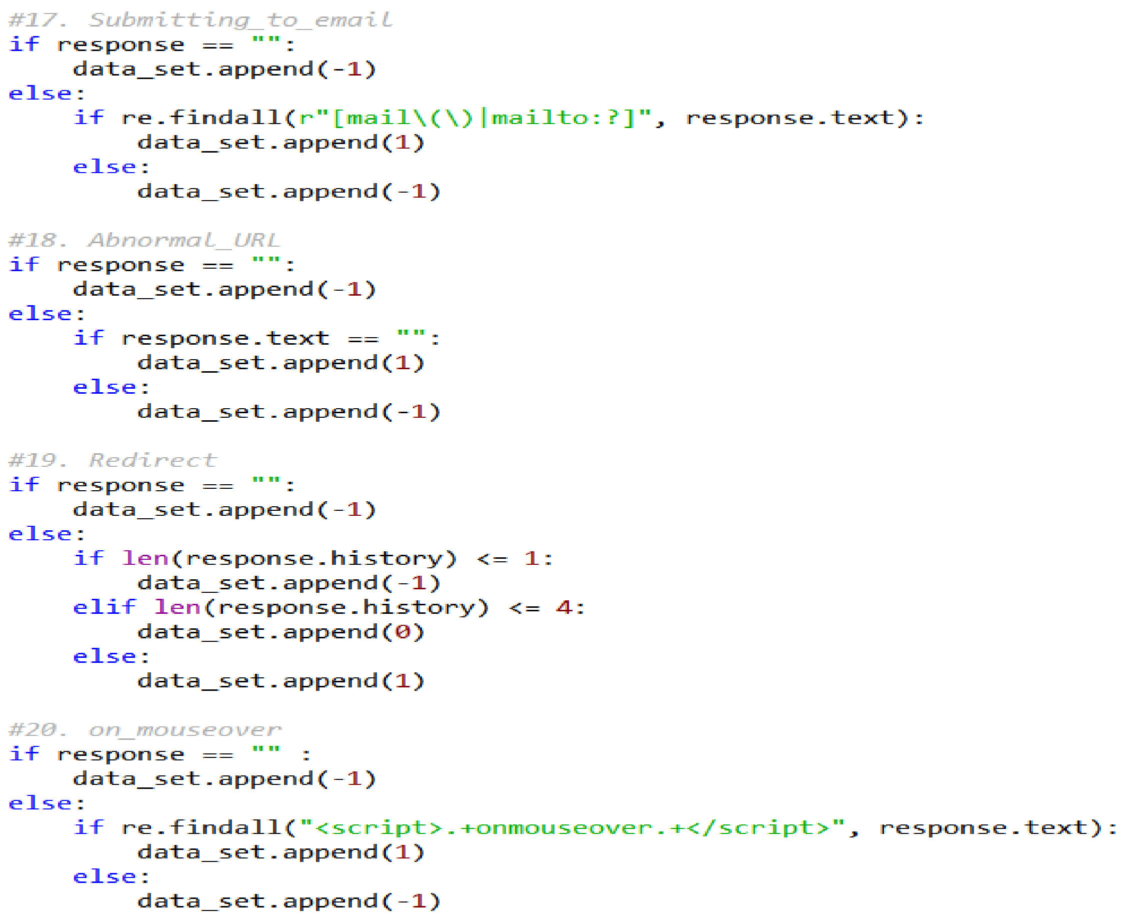
Figure.12- Code for Dataset creation

Feature Extraction



```
# 4.having_At_Symbol
if re.findall("@", url):
    data_set.append(-1)
else:
    data_set.append(1)

# 5.double_slash_redirecting
list=[x.start(0) for x in re.finditer('//', url)]
if list[len(list)-1]>6:
    data_set.append(-1)
else:
    data_set.append(1)

# 6.Prefix_Suffix
if re.findall(r"https?://[^\-]+-[^\-]+/", url):
    data_set.append(-1)
else:
    data_set.append(1)

# 7.having_Sub_Domain
if len(re.findall("\.", url)) == 1:
    data_set.append(1)
elif len(re.findall("\.", url)) == 2:
    data_set.append(0)
else:
    data_set.append(-1)
```

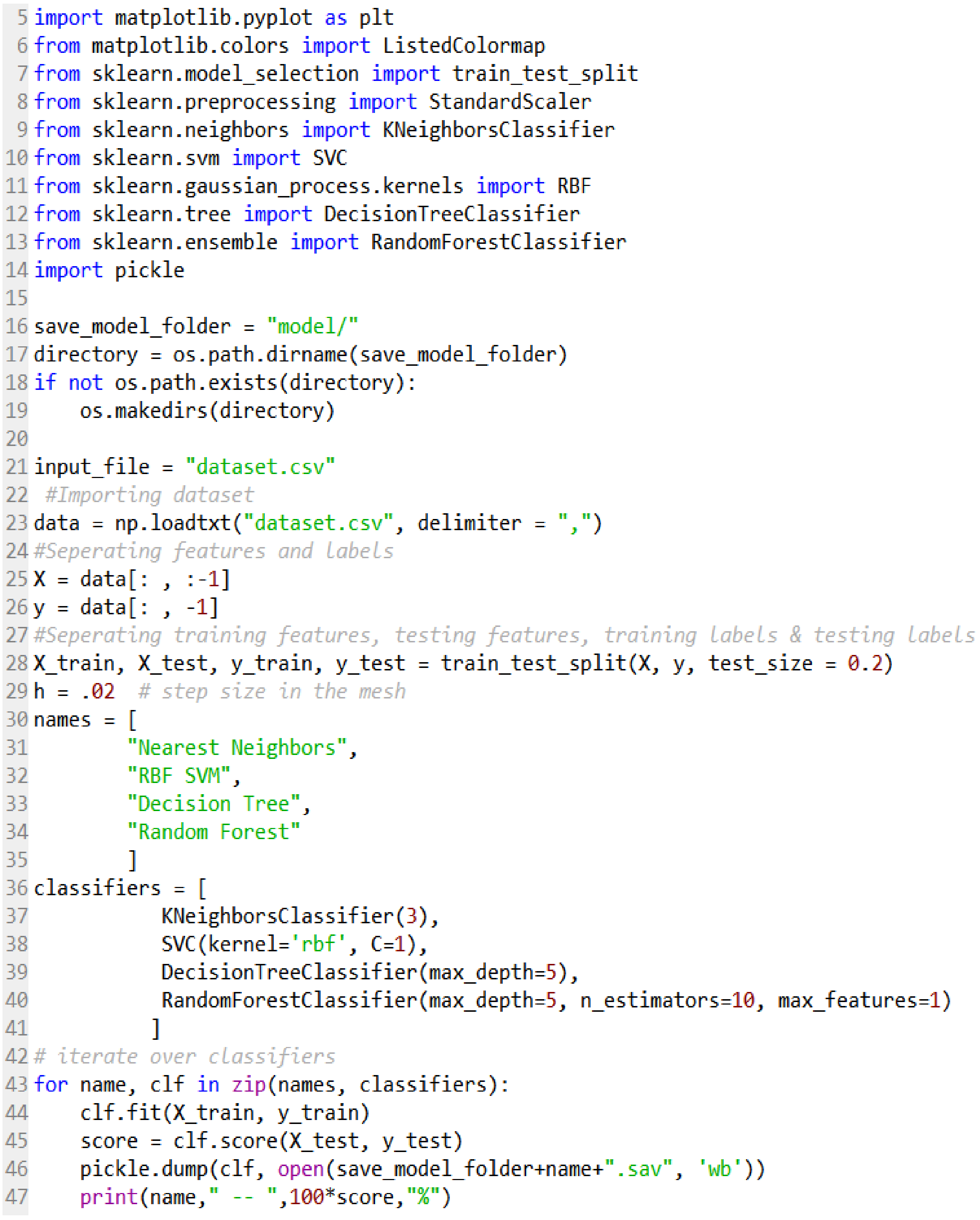
Figure.13- Code for Feature Extraction



```
#17. Submitting_to_email
if response == "":
    data_set.append(-1)
else:
    if re.findall(r"[mail\(\)|mailto:?]", response.text):
        data_set.append(1)
    else:
        data_set.append(-1)

#18. Abnormal_URL
if response == "":
    data_set.append(-1)
else:
    if response.text == "":
        data_set.append(1)
    else:
        data_set.append(-1)

#19. Redirect
if response == "":
    data_set.append(-1)
else:
    if len(response.history) <= 1:
        data_set.append(-1)
    elif len(response.history) <= 4:
        data_set.append(0)
    else:
        data_set.append(1)

#20. on_mouseover
if response == "" :
    data_set.append(-1)
else:
    if re.findall("<script>.+onmouseover.+</script>", response.text):
        data_set.append(1)
    else:
        data_set.append(-1)
```

Figure.14- Code for Feature Extraction

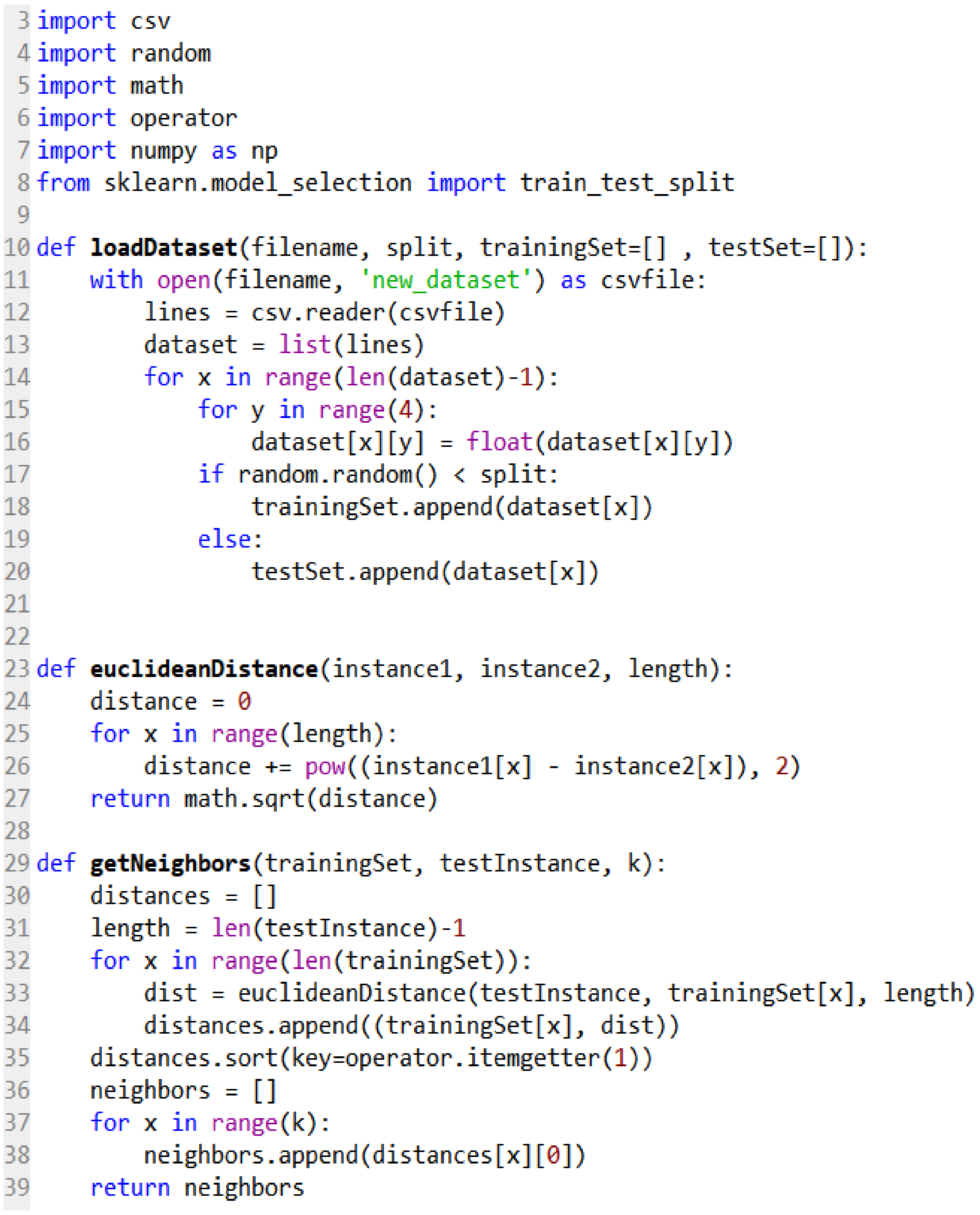
Model Creation



```python
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import SVC
from sklearn.gaussian_process.kernels import RBF
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
import pickle

save_model_folder = "model/"
directory = os.path.dirname(save_model_folder)
if not os.path.exists(directory):
    os.makedirs(directory)

input_file = "dataset.csv"
 #Importing dataset
data = np.loadtxt("dataset.csv", delimiter = ",")
#Seperating features and labels
X = data[: , :-1]
y = data[: , -1]
#Seperating training features, testing features, training labels & testing labels
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
h = .02  # step size in the mesh
names = [
        "Nearest Neighbors",
        "RBF SVM",
        "Decision Tree",
        "Random Forest"
        ]
classifiers = [
           KNeighborsClassifier(3),
           SVC(kernel='rbf', C=1),
           DecisionTreeClassifier(max_depth=5),
           RandomForestClassifier(max_depth=5, n_estimators=10, max_features=1)
          ]
# iterate over classifiers
for name, clf in zip(names, classifiers):
    clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    pickle.dump(clf, open(save_model_folder+name+".sav", 'wb'))
    print(name," -- ",100*score,"%")
```

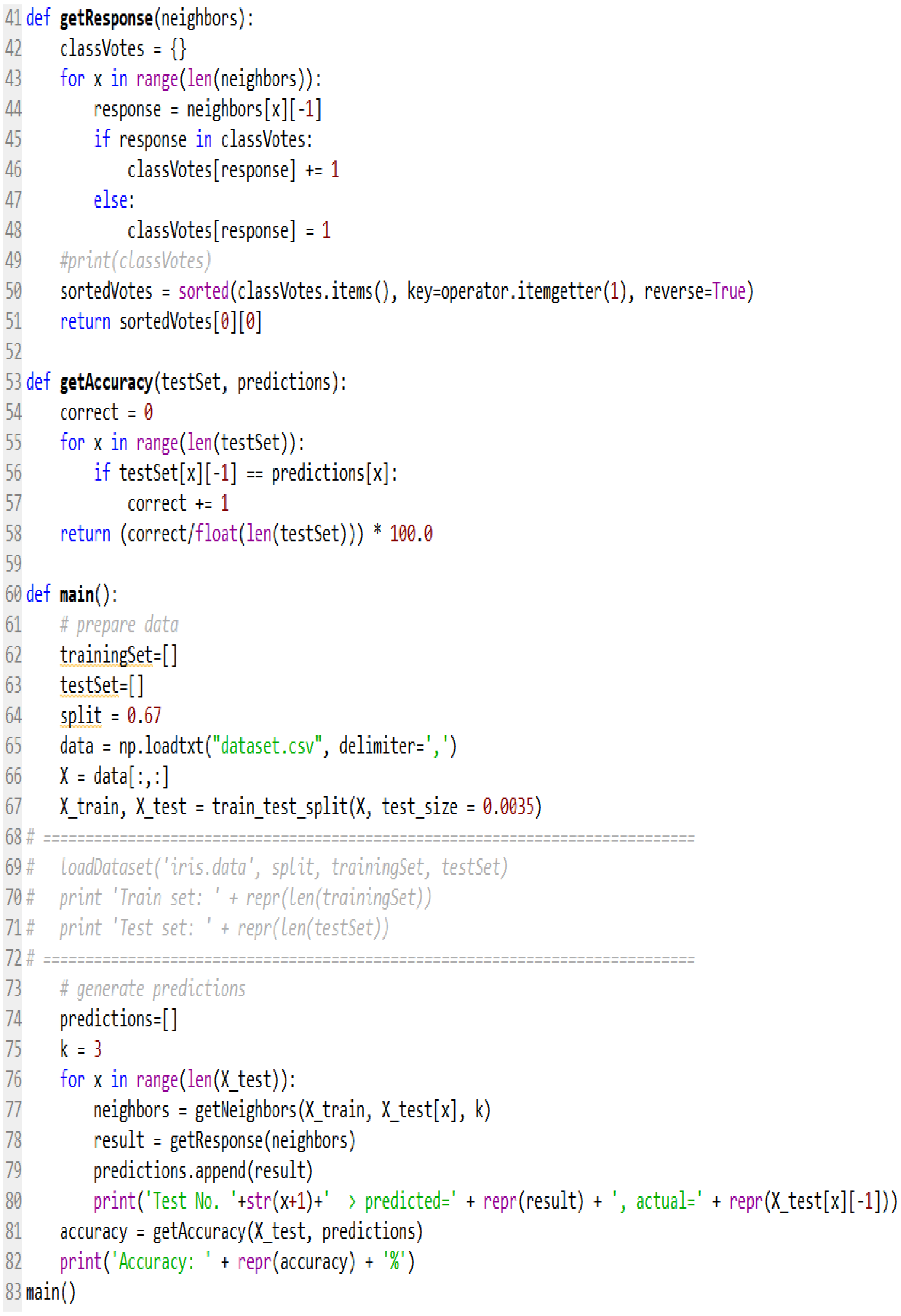
Figure.15- Code for Model Creation

KNN Classifier



```python
import csv
import random
import math
import operator
import numpy as np
from sklearn.model_selection import train_test_split

def loadDataset(filename, split, trainingSet=[] , testSet=[]):
    with open(filename, 'new_dataset') as csvfile:
        lines = csv.reader(csvfile)
        dataset = list(lines)
        for x in range(len(dataset)-1):
            for y in range(4):
                dataset[x][y] = float(dataset[x][y])
            if random.random() < split:
                trainingSet.append(dataset[x])
            else:
                testSet.append(dataset[x])


def euclideanDistance(instance1, instance2, length):
    distance = 0
    for x in range(length):
        distance += pow((instance1[x] - instance2[x]), 2)
    return math.sqrt(distance)

def getNeighbors(trainingSet, testInstance, k):
    distances = []
    length = len(testInstance)-1
    for x in range(len(trainingSet)):
        dist = euclideanDistance(testInstance, trainingSet[x], length)
        distances.append((trainingSet[x], dist))
    distances.sort(key=operator.itemgetter(1))
    neighbors = []
    for x in range(k):
        neighbors.append(distances[x][0])
    return neighbors
```

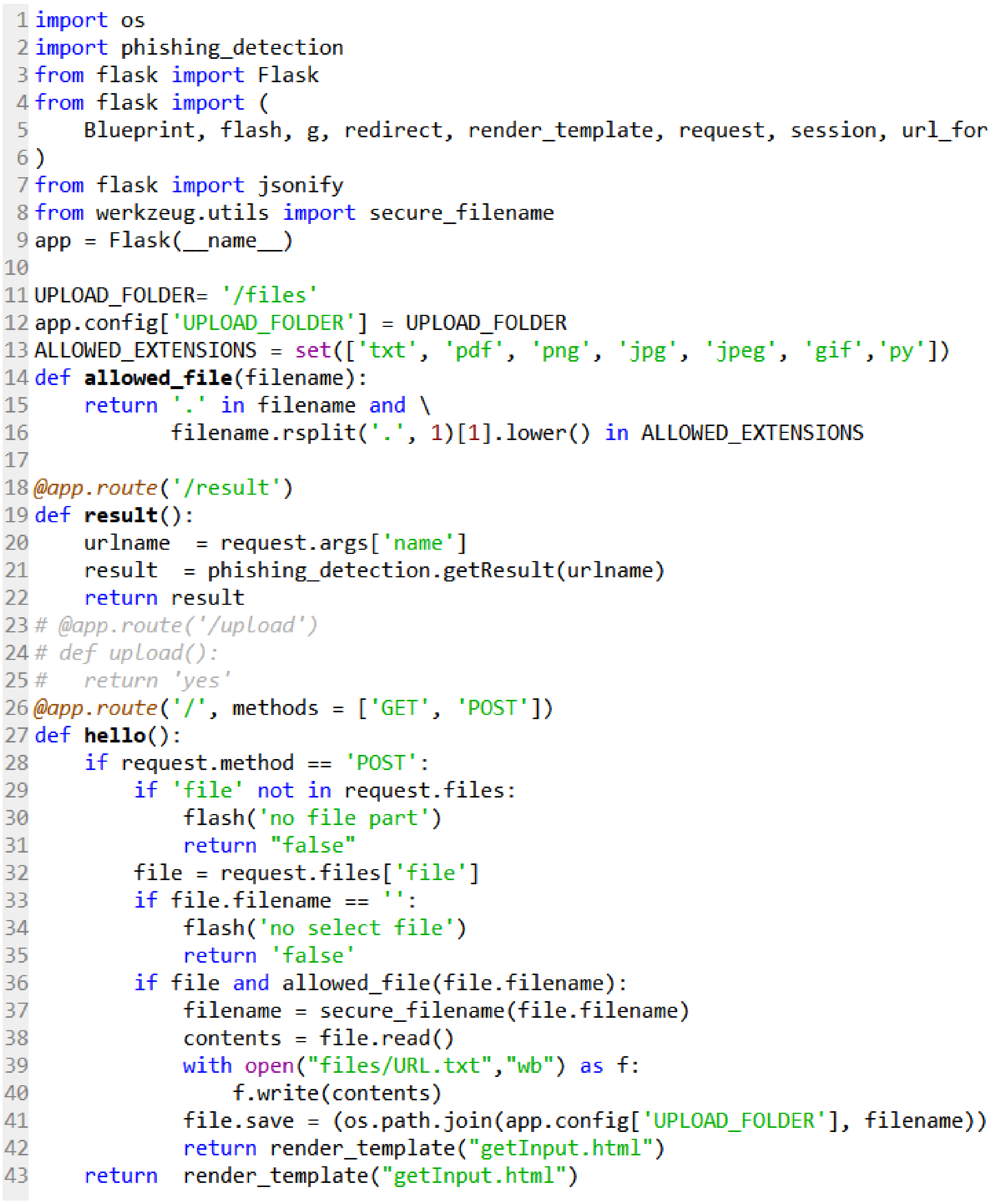
Figure.16- Code for KNN Classifier



```python
def getResponse(neighbors):
    classVotes = {}
    for x in range(len(neighbors)):
        response = neighbors[x][-1]
        if response in classVotes:
            classVotes[response] += 1
        else:
            classVotes[response] = 1
    #print(classVotes)
    sortedVotes = sorted(classVotes.items(), key=operator.itemgetter(1), reverse=True)
    return sortedVotes[0][0]

def getAccuracy(testSet, predictions):
    correct = 0
    for x in range(len(testSet)):
        if testSet[x][-1] == predictions[x]:
            correct += 1
    return (correct/float(len(testSet))) * 100.0

def main():
    # prepare data
    trainingSet=[]
    testSet=[]
    split = 0.67
    data = np.loadtxt("dataset.csv", delimiter=',')
    X = data[:,:]
    X_train, X_test = train_test_split(X, test_size = 0.0035)
# ============================================================================
#   loadDataset('iris.data', split, trainingSet, testSet)
#   print 'Train set: ' + repr(len(trainingSet))
#   print 'Test set: ' + repr(len(testSet))
# ============================================================================
    # generate predictions
    predictions=[]
    k = 3
    for x in range(len(X_test)):
        neighbors = getNeighbors(X_train, X_test[x], k)
        result = getResponse(neighbors)
        predictions.append(result)
        print('Test No. '+str(x+1)+'  > predicted=' + repr(result) + ', actual=' + repr(X_test[x][-1]))
    accuracy = getAccuracy(X_test, predictions)
    print('Accuracy: ' + repr(accuracy) + '%')
main()
```

Figure.17- Code for KNN Classifier

Creation of Server



```python
import os
import phishing_detection
from flask import Flask
from flask import (
    Blueprint, flash, g, redirect, render_template, request, session, url_for
)
from flask import jsonify
from werkzeug.utils import secure_filename
app = Flask(__name__)

UPLOAD_FOLDER= '/files'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
ALLOWED_EXTENSIONS = set(['txt', 'pdf', 'png', 'jpg', 'jpeg', 'gif','py'])
def allowed_file(filename):
    return '.' in filename and \
           filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route('/result')
def result():
    urlname  = request.args['name']
    result  = phishing_detection.getResult(urlname)
    return result
# @app.route('/upload')
# def upload():
#    return 'yes'
@app.route('/', methods = ['GET', 'POST'])
def hello():
    if request.method == 'POST':
        if 'file' not in request.files:
            flash('no file part')
            return "false"
        file = request.files['file']
        if file.filename == '':
            flash('no select file')
            return 'false'
        if file and allowed_file(file.filename):
            filename = secure_filename(file.filename)
            contents = file.read()
            with open("files/URL.txt","wb") as f:
                f.write(contents)
            file.save = (os.path.join(app.config['UPLOAD_FOLDER'], filename))
            return render_template("getInput.html")
    return  render_template("getInput.html")
```

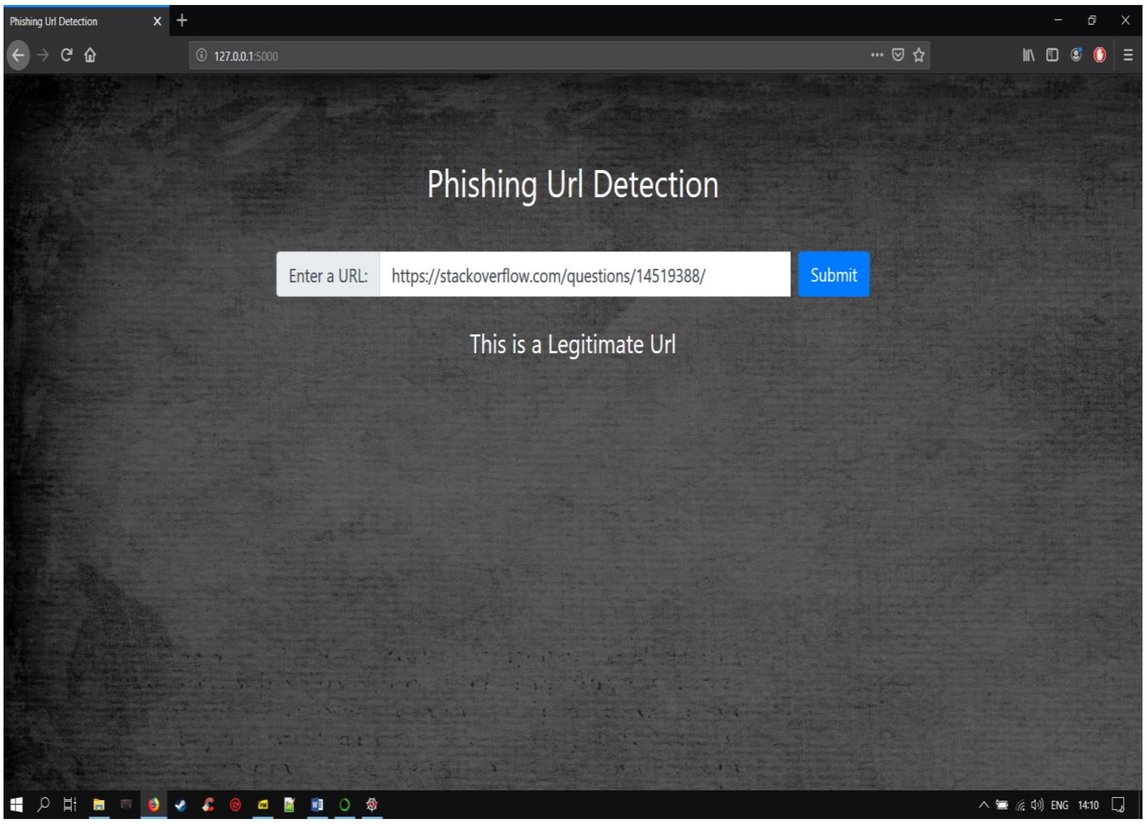
Figure.18- Code for Server Creation

getResult function ()

```python
import numpy as np
import feature_extraction
from sklearn.ensemble import RandomForestClassifier as rfc
#from sklearn.svm import SVC
#from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression as lr
from flask import jsonify
import pickle

def getResult(url):
    clf = pickle.load(open("model/Nearest Neighbors.sav", 'rb'))
    X_new = []
    X_input = url
    X_new=feature_extraction.generate_data_set(X_input)
    X_new = np.array(X_new).reshape(1,-1)

    try:
        prediction = clf.predict(X_new)
        if prediction == -1:
            return "Phishing Url"
        else:
            return "Legitimate Url"
    except:
        return "Phishing Url"
```

Figure.19- Code for getResult Function

Homepage of Phishing Detection System



https://stackoverflow.com/questions/14519388/

This is a Legitimate Url

Figure.20- Homepage of Phishing Detection Website

CONCLUSION

Phishing websites mainly retrieve user’s information through login pages. They are interested in the bank details of the users. Out of the many features considered to detect the phishing website, the most important one is HTTPS with SSL i.e. whether a website uses HTTPS, issuer of certificate is trusted or not, and the age of certificate should be at least one year. In this regard, the Phishing Website Dataset is tested to predict the accuracy of phishing detection evaluation based on four classifier algorithms (KNN, RBF-SVM, Decision Tree and Random Forest). The accuracy using K-Nearest Neighbor is 95.20%, RBF (Radial Basis Function) Support Vector Machine is 94.70%, Decision Tree is 91.94% and Random Forest is 87.74%. Hence, we conclude that K-Nearest Neighbors classifier results best in terms of accuracy among the four classifier algorithms.

FUTURE SCOPE

Feature selection techniques need more improvement to cope with the continuous development of new techniques by the phishers over the time. As part of future work, it can be enhanced to include the following functionalities.

- ✓ Certain additional heuristics such as Number of Links Pointing to Page, Google Index, TTL value of the domain can be implemented in addition to the heuristics.
- ✓ The discriminative classifier algorithms such as Generalized Linear Model, Gradient Boosting (GBM), Boosting can be used to predict the URL category by training huge amount of the data extracted from the datasets.

REFERENCES

[1] Nuttapong Sanglerdsinlapachai, Arnon Rungsawang "**Using Domain Top-page Similarity Feature in Machine Learning-based Web Phishing Detection**", in 2010 Third International Conference on Knowledge Discovery and Data Mining

[2] A. Belabed, E. Aïmeur, A. Chikh "**A personalized whitelist approach for phishing webpage detection**", in 2012 Seventh International Conference on Availability, Reliability and Security

[3] Samuel Marchal, Jérôme Francois, Radu State, Thomas Engel "**Phish Score: Hacking Phishers' Minds** in 10th CNSM and Workshop at 2014 IFIP

[4] Zheng Dong, Apu Kapadia, Jim Blythe and L. Jean Camp "**Beyond the Lock Icon: Real-time Detection of Phishing Websites Using Public Key Certificates**"

[5] Ishant Tyagi, Jatin Shad, Shubham Sharma "**A Novel Machine Learning Approach to Detect Phishing Websites**", in 5th International Conference on Signal Processing and Integrated Networks (SPIN), 2018