1. INTRODUCTION

1.1 Introduction

Internet use has become an essential part of our daily activities as a result of rapidly growing technology. Due to this rapid growth of technology and intensive use of digital systems, data security of these systems has gained great importance. The primary objective of maintaining security in information technologies is to ensure that necessary precautions are taken against threats and dangers likely to be faced by users during the use of these technologies. Phishing is defined as imitating reliable websites in order to obtain the proprietary information entered into websites every day for various purposes, such as usernames, passwords and citizenship numbers. Phishing websites contain various hints among their contents and web browser-based information . Individual(s) committing the fraud sends the fake website or e-mail information to the target address as if it comes from an organization, bank or any other reliable source that performs reliable transactions. Contents of the website or the e- mail include requests aiming to lure the individuals to enter or update their personal information or to change their passwords as well as links to websites that look like exact copies of the websites of the organizations concerned. An attack can have devastating results. For individuals, this includes unauthorized purchases, the stealing of funds, or identify theft.

Moreover, phishing is often used to gain a foothold in corporate or governmental networks as a part of a larger attack, such as an advanced persistent threat (APT) event. In this latter scenario, employees are compromised in order to bypass security perimeters, distribute malware inside a closed environment, or gain privileged access to secured data.

An organization succumbing to such an attack typically sustains severe financial losses in addition to declining market share, reputation, and consumer trust. Depending on scope, a phishing attempt might escalate into a security incident from which a business will have a difficult time recovering. Phishing attack protection requires steps be taken by both users and enterprises.

For users, vigilance is key. A spoofed message often contains subtle mistakes that expose its true identity. These can include spelling mistakes or changes to domain names, as seen in the earlier URL

example. Users should also stop and think about why they're even receiving such an email. Phishing attacks are the practice of sending fraudulent communications that appear to come from a reputable source. It is usually done through email. The goal is to steal sensitive data like credit card and login information, or to install malware on the victim's machine. Phishing is a common type of cyber attack that everyone should learn about in order to protect themselves. Phishing is the fraudulent use of electronic communications to deceive and take advantage of users. Phishing attacks attempt to gain sensitive, confidential information such as usernames, passwords, credit card information, network credentials, and more. By posing as a legitimate individual or institution via phone or email, cyber attackers use **social engineering** to manipulate victims into performing specific actions—like clicking on a malicious link or attachment—or willfully divulging confidential information.

Both individuals and organizations are at risk; almost any kind of personal or organizational data can be valuable, whether it be to commit fraud or access an organization's network. In addition, some phishing scams can target organizational data in order to support espionage efforts or state-backed spying on opposition groups. Phishing attempts most often begin with an email attempting to obtain sensitive information through some user interaction, such as clicking on a malicious link or downloading an infected attachment.

Phishing is a form of fraud in which an attacker masquerades as a reputable entity or person in email or other communication channels. The attacker uses phishing emails to distribute malicious links or attachments that can perform a variety of functions, including the extraction of login credentials or account information from victims. Phishing is popular with cybercriminals, as it is far easier to trick someone into clicking a malicious link in a seemingly legitimate phishing email than trying to break through a computer's defenses.

Phishing attacks typically rely on social networking techniques applied to email or other electronic communication methods, including direct messages sent over social networks, SMS text messages and other instant messaging modes

1.2 Purpose of the project

Phishing is a cyber attack that uses disguised email as a weapon. The goal is to trick the email recipient into believing that the message is something they want or need — a request from their bank, for instance, or a note from someone in their company — and to click a link or download an attachment.

What really distinguishes phishing is the form the message takes: the attackers masquerade as a trusted entity of some kind, often a real or plausibly real person, or a company the victim might do business with. It's one of the oldest types of cyberattacks, dating back to the 1990s, and it's still one of the most widespread and pernicious, with phishing messages and techniques becoming increasingly sophisticated.

The phishing scheme could use email, text, or web page.Phishing emails are the most notorious forms of phishing campaigns. The bad actor will send a fake email that will contain links to false websites that appear to be associated with a legitimate business, but is being used to gather anything from passwords, to social security or account numbers. There are other forms of phishing campaigns and techniques that are used to track potential victims, including vishing, SMiShing, spy-phishing, watering hole attacks, even spam. SMS phishing (or smishing/SMiShing) is a phishing campaign that uses a bait text message to lure potential victims. Spear-phishing uses an email that has more specific information than a standard phishing email. The attacker will spend time researching the potential victims online and social media presence to gather information that will allow them to create a false sense of familiarity. Vishing uses telephone communications to social-engineer personal information. A watering hole attack is focused on a particular group, eventually affecting members of that group. Spam is a well known email type, and spy-phishing is using a phishing method to install spyware onto a potential victim’s computer.

The aim of these attacks is to steal the information used by individuals and organizations to conduct transactions. Phishing websites contain various hints among their contents and web browser-based information. The purpose of this study is to perform Extreme Learning Machine (ELM) based classification for 30 features including Phishing Websites Data in UC Irvine Machine Learning Repository database.

1.3 Problem statement

Phishing is a major threat to all Internet users and is difficult to trace or defend against since it does not present itself as obviously malicious in nature. In today's society, everything is put online and the safety of personal credentials is at risk. Phishing can be seen as one of the oldest and easiest ways of stealing information from people and it is used for obtaining a wide range of personal details. It also has a fairly simple approach – send an email, email sends victim to a site, site steals information. The primary objective of maintaining security in information technologies is to ensure that necessary precautions are taken against threats and dangers likely to be faced by users during the use of these technologies . Phishing is defined as imitating reliable websites in order to obtain the proprietary information entered into websites every day for various purposes, such as usernames, passwords and citizenship numbers. Phishing websites contain various hints among their contents and web browser-based information. Individual(s) committing the fraud sends the fake website or e-mail information to the target address as if it comes from an organization, bank or any other reliable source that performs reliable transactions. Contents of the website or the e-mail include requests aiming to lure the individuals to enter or update their personal information or to change their passwords as well as links to websites that look like exact copies of the websites of the organizations concerned. Phishing websites contain various hints among their contents and web browser-based information Individual(s) committing the fraud sends the fake website or e-mail information to the target address as if it comes from an organization, bank or any other reliable source that performs reliable transactions. Contents of the website or the e-mail include requests aiming to lure the individuals to enter or update their personal information or to change their passwords as well as links to websites that look like exact copies of the websites of the organizations concerned. Many articles have been published about how to predict the phishing websites by using artificial intelligence techniques. We examined phishing websites and extracted features of these web sites. we defined features of phishing attack and we proposed a classification model in order to classification of the phishing attacks. This method consists of feature extraction from websites and classification section. In the feature extraction, we have clearly defined rules of phishing feature extraction and these rules have been used for obtaining features. In order to classification of these feature, SVM, NB and ELM were used. In the ELM, 6 different activation functions were used and ELM achieved highest accuracy score.

1.4 Solution for the problem statement:

Extreme Learning Machine (ELM) is a feed-forward artificial neural network (ANN) model with a single hidden layer. For the ANN to ensure a high-performing learning, parameters such as threshold value, weight and activation function must have the appropriate values for the data system to be modeled. In gradient-based learning approaches, all of these parameters are changed iteratively for appropriate values. Thus, they may be slow and produce low-performing results due to the likelihood of getting stuck in local minima. In ELM Learning Processes, differently from ANN that renews its parameters as gradient-based, input weights are randomly selected while output weights are analytically calculated. As an analytical learning process substantially reduces both the solution time and the likelihood of error value getting stuck in local minima, it increases the performance ratio. In order to activate the cells in the hidden layer of ELM, a linear function as well as non-linear (sigmoid, sinus, Gaussian), non-derivable or discrete activation functions can be used Extreme Learning Machine (ELM) is a feed-forward artificial neural network (ANN) model with a single hidden layer. For the ANN to ensure a high-performing learning, parameters such as threshold value, weight and activation function must have the appropriate values for the data system to be modeled. In gradient-based learning approaches, all of these parameters are changed iteratively for appropriate values. Thus, they may be slow and produce low-performing results due to the likelihood of getting stuck in local minima.In ELM Learning Processes, differently from ANN that renews its parameters as gradient-based, input weights are randomly selected while output weights are analytically calculated. As an analytical learning process substantially reduces both the solution time and the likelihood of error value getting stuck in local minima, it increases the performance ratio. In order to activate the cells in the hidden layer of ELM, a linear function as well as non-linear (sigmoid, sinus, Gaussian), non-derivable or discrete activation functions can be used.

Procedural steps for solving the classification problem presented is as follows:

- **Identification of the problem**

This study attempts to solve the problem as to how phishing analysis data will be classified.

- **Data set**

Approximately 11,000 data containing the 30 features extracted based on the features of websites in UC Irvine Machine Learning Repository database.

- **Modeling**

After the data is ready to be processed, modeling process for the learning algorithm is initiated. The model is basically the construction of the need for output identified in accordance with the task qualifications.

2. SYSTEM ANALYSIS

2.1 Study Of The System

- Numpy
- Pandas
- Matplotlib
- Scikit –learn

1 . Numpy:

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

2.Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

3. Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

4. Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

- **NumPy**: Base n-dimensional array package
- **SciPy**: Fundamental library for scientific computing
- **Matplotlib**: Comprehensive 2D/3D plotting
- **IPython**: Enhanced interactive console
- **Sympy**: Symbolic mathematics
- **Pandas**: Data structures and analysis
- Extensions or modules for SciPy care conventionally named SciKits.

2.2. Proposed System

Thus study the problem of predicting online purchase conversions in an e-commerce site. To understand user behavior and intent on the web, exist ing predictors leverage the traditional search pattern of entering queries then clicking on interesting results. However, conversion takes more than a click. That is, after repeatedly clicking around and being exposed to advertising (i.e., retargeted), users‘ ultimate success metric of the marketplace search is buying products. Beyond the traditional mechanism, our contribution is to allow the predictors to consider dynamic marketplace mechanisms for a deeper prediction of both clicks and purchases. Specifically, inspired by traditional search problems we focus on two research questions:
–Prediction from market‖ and –Predictability from individual‖ for conversion.

2.3. Input And Output

The following some are the projects inputs and outputs.

Inputs:

- Importing the all required packages like numpy, pandas, matplotlib, scikit – learn and required machine learning algorithms packages .
- Setting the dimensions of visualization graph.
- Downloading and importing the dataset and convert to data frame.

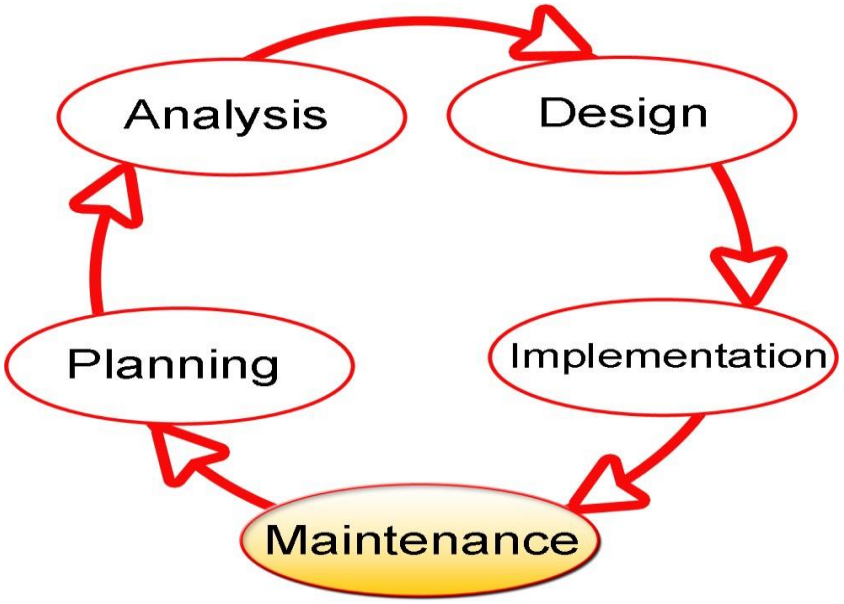
Outputs:

- preprocessing the importing data frame for imputing nulls with the related information.
- All are displaying cleaned outputs.
- After applying machine learning algorithms it will give good results and visualization plots.

2.4. Process Models Used With Justification

SDLC Model:

Software Development Life Cycle (SDLC)



The Software Development Lifecycle(SDLC) for small to medium database application development efforts.

This project uses iterative development lifecycle, where components of the application are developed through a series of tight iteration. The first iteration focus on very basic functionality, with subsequent iterations adding new functionality to the previous work and or correcting errors identified for the components in production.

The six stages of the SDLC are designed to build on one another, taking outputs from the previous stage, adding additional effort, and producing results that leverage the previous effort and are directly traceable to the previous stages. During each stage, additional information is gathered or developed, combined with the inputs, and used to produce the stage deliverables. It is important to not that the additional information is restricted in scope, new ideas that would take

the project in directions not anticipated by the initial set of high-level requirements or features that are out-of-scope are preserved for later consideration.

Too many software development efforts go awry when development team and customer personnel get caught up in the possibilities of automation. Instead of focusing on high priority features, the team can become mired in a sea of nice to have features that are not essential to solve the problem, but in themselves are highly attractive. This is the root cause of large percentage of failed and or abandoned development efforts and is the primary reason the development team utilizes the iterative model.

Roles and Responsibilities of PDR AND PER

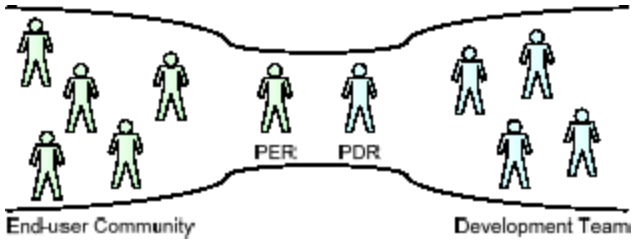
The iterative lifecycle specifies two critical roles that act together to clearly communicate project 0issues and concepts between the end-user community and the development team.

Primary End-user Representative (PER)

The PER is a person who acts as the primary point of contact and principal approver for the end-user community. The PER is also responsible for ensuring that appropriate subject matter experts conduct end-user reviews in a timely manner.

PER-PDR Relationship

The PER and PDR are the brain trust for the development effort. The PER has the skills and domain knowledge necessary to understand the issues associated with the business processes to the supported by the application and has a close working relationship with the other members of the end-user community. The PDR has the same advantages regarding the application development process and the other members of the development team together, they act as the concentration points for knowledge about the application to be developed.



The objective of this approach is to create the close relationship that is characteristic of a software project with one developer and one end-user in essence, this approach the ―pair programming‖ concept from Agile methodologies and extends it to the end- user community. While it is difficult to create close relationships between the diverse members of an end-user community and a software development team, it is much simpler to create a close relationship between the lead representatives for each group.

When multiple end-users are placed into relationship with multiple members of a development team, communication between the two groups degrades as the number of participants grows. In this model, members of end-user community may communicate with members of the development team as needed, but it is the responsibility of all participants to keep the PER and PDR apprised of the communications for example, this allows the PER and PDR to resolve conflicts that arise when two different end-users communicate different requirements for the same application feature to different members of the development team.

Input Design

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.
- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

Input Stages:

The main input stages before the information gets stored in the database media: Ex: In this project voter either existing or new user data will be stored in database as the inputs given by users….

- Data recording ,Data transcription, Data conversion, Data verification
- Data control, Data transmission, Data validation, Data correction

Output Design

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provide a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization,
- Internal Outputs whose destination is within organization and they are the
- User‘s main interface with the computer.
- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

The outputs were needed to be generated as a hard copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

Design Principles & Methodology:

Object Oriented Analysis And Design

When Object orientation is used in analysis as well as design, the boundary between OOA and OOD is blurred. This is particularly true in methods that combine analysis and design. One reason for this blurring is the similarity of basic constructs (i.e.,objects and classes) that are used in OOA and OOD. Through there is no agreement about what parts of the object-oriented development process belongs to analysis and what parts to design, there is some general agreement about the domains of the two activities.

The fundamental difference between OOA and OOD is that the former models the problem domain, leading to an understanding and specification of the problem, while the latter models the solution to the problem. That is, analysis deals with the problem domain, while design deals with the solution domain. However, in OOAD subsumed in the solution domain representation. That is, the solution domain representation, created by OOD, generally contains much of the representation created by OOA. The separating line is matter of perception, and different people have different views on it. The lack of clear separation between analysis and design can also be considered one of the strong points of the object-oriented approach the transition from analysis to design is –seamless‖. This is also the main reason OOAD methods-where analysis and designs are both performed.

The main difference between OOA and OOD, due to the different domains of modeling, is in the type of objects that come out of the analysis and design process.

Features of OOAD:

- It users Objects as building blocks of the application rather functions
- All objects can be represented graphically including the relation between them.
- All Key Participants in the system will be represented as actors and the actions done by them will be represented as use cases.
- A typical use case is nothing bug a systematic flow of series of events which can be well described using sequence diagrams and each event can be described diagrammatically by Activity as well as state chart diagrams.

The Genesis Of UML:

Software engineering has slowly become part of our everyday life. From washing machines to compact disc player, through cash machines and phones, most of our daily activities use software, and as time goes by, the more complex and costly this software becomes. The demand for sophisticated software greatly increases the constraints imposed on development teams. Software engineers are facing a world of growing complexity due to the nature of applications, the distributed and heterogeneous environments, the size of programs, the organization of software development teams, and the end-users ergonomic expectations.

To surmount these difficulties, software engineers will have to learn not only how to do their job, but also how to explain their work to others, and how to understand when others work is explained to them. For these reasons, they have (and will always have) an increasing need for methods.

From Functional to Object-Oriented Methods

Although object-oriented methods have roots that are strongly anchored back in the 60s, structured and functional methods were the first to be used. This is not very surprising, since functional methods are inspired directly my computer architecture (a proven domain well known to computer scientists). The separation of data and code, just as exists physically in the hardware, was translated into the methods; this is how computer scientists got into the habit of thinking in terms of system functions.

This approach is natural when looked at in its historical context, but today, because of its lack of abstraction, it has become almost completely anachronistic. There is no reason to impose the underlying hardware on a software solution. Hardware should act as the servant of the software that is executed on it, rather than imposing architectural constraints.

Towards A Unified Modelling Language

The unification of object-oriented modeling methods became possible as experience allowed evaluation of the various concepts proposed by existing methods. Based on the fact that differences between the various methods were becoming smaller, and that the method wars did not move object-oriented technology forward any longer, Jim Rumbaugh and Grady Booch decided at the end of 1994 to unify their work within a single method: the

Unified Method. A year later they were joined by Ivar Jacobson, the father of use cases, a very efficient technique for the determination of requirements.

Booch, Rumbaugh and Jacobson adopted four goals:

- To represent complete systems (instead of only the software portion) using object oriented concepts.
- To establish an explicit coupling between concepts and executable code.
- To take into account the scaling factors that are inherent to complex and critical systems.

3.FEASIBILITY STUDY

Preliminary investigation examine project feasibility the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- Economical Feasibility
- Operational Feasibility
- Technical Feasibility

3.1 Economical Feasibility

A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs.

The system is economically feasible. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, There is nominal expenditure and economical feasibility for certain.

3.2 Operational Feasibility

Proposed projects are beneficial only if they can be turned out into information system. That will meet the organization‘s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation. Some of the important issues raised are to test the operational feasibility of a project includes the following: -

- Is there sufficient support for the management from the users?
- Will the system be used and work properly if it is being developed and implemented?

- Will there be any resistance from the user that will undermine the possible application benefits?

This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits.

The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

3.3 Technical Feasibility

The technical issue usually raised during the feasibility stage of the investigation includes the following:

- Does the necessary technology exist to do what is suggested?
- Do the proposed equipments have the technical capacity to hold the data required to use the new system?
- Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?
- Can the system be upgraded if developed?
- Are there technical guarantees of accuracy, reliability, ease of access and data security?

Earlier no system existed to cater to the needs of _Secure Infrastructure Implementation System‘. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides an easy access to the users. The database‘s purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability and security. The software and hard requirements for the development of this project are not many and are already available in-house at NIC or are available as free as open source. The work for the project is done with the current equipment and existing software technology.

4. SOFTWARE REQUIREMENT SPECIFICATION

A **Software Requirements Specification** (**SRS**) – a requirements specification for a software system – is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development life cycle domain, typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- **Business requirements** describe in business terms what must be delivered or accomplished to provide value.
- **Product requirements** describe properties of a system or product (which could be one of Several ways to accomplish a set of business requirements.)
- **Process requirements** describe activities performed by the developing organization. For instance, process requirements could specify specific methodologies that must be followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement); a requirement that the product be maintainable (a Product requirement) often is addressed by imposing requirements to follow particular development styles

Purpose

An systems engineering, a requirement can be a description of what a system must do, referred to as a Functional Requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called Non-functional requirements, or 'performance requirements' or 'quality of service requirements.' Examples of such requirements include usability, availability, reliability, supportability, testability and maintainability.

A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality (e.g. of user interfaces) with other owned products.

In software engineering, the same meanings of requirements apply, except that the focus of interest is the software itself.

4.1 Functional Requirements

- Accuracy of model should be high then only we can get perfect results.
- Need to be analyze the data and remove the unwanted data, if there is any missing values there need to remove those missing values or else has to put suitable for it.
- Feature selection is the major part of the data analysis get the perfect feature to build model

4.2 Non Functional Requirements

The major non-functional Requirements of the system are as follows

Usability

The system is designed with completely automated process hence there is no or less user intervention.

Reliability

The system is more reliable because of the qualities that are inherited from the chosen platform java. The code built by using java is more reliable.

Performance

This system is developing in the high level languages and using the advanced front-end and back-end technologies it will give response to the end user on client system with in very less time.

Supportability

The system is designed to be the cross platform supportable. The system is supported on a wide range of hardware and any software platform, which is having JVM, built into the system.

Implementation

The system is implemented in web environment using struts framework. The apache tomcat is used as the web server and windows xp professional is used as the platform.

Interface the user interface is based on Struts provides HTML Tag

4.3 Hardware Requirements:

- RAM : 4GB and Higher
- Processor : Intel i3 and above
- Hard Disk : 500GB: Minimum

4.4 Software Requirements:

- OS: Windows or Linux
- Python IDE : python 2.7.x and above
- Jupyter IDE
- Setup tools and pip to be installed for 3.6 and above
- Language : Python Scripting

5. PYTHON FRAMEWORK

Introduction

Introduction to Django This book is about Django, a Web development framework that saves you time and makes Web development a joy. Using Django, you can build and maintain high-quality Web applications with minimal fuss. At its best, Web development is an exciting, creative act; at its worst, it can be a repetitive, frustrating nuisance. Django lets you focus on the fun stuff — the crux of your Web application — while easing the pain of the repetitive bits. In doing so, it provides high-level abstractions of common Web development patterns, shortcuts for frequent programming tasks, and clear conventions for how to solve problems. At the same time, Django tries to stay out of your way, letting you work outside the scope of the framework as needed. The goal of this book is to make you a Django expert. The focus is twofold. First, we explain, in depth, what Django does and how to build Web applications with it. Second, we discuss higher-level concepts where appropriate, answering the question ‒How can I apply these tools effectively in my own projects?‖ By reading this book, you‘ll learn the skills needed to develop powerful Web sites quickly, with code that is clean and easy to maintain.

What Is a Web Framework?

Django is a prominent member of a new generation of Web frameworks. So what exactly does that term mean? To answer that question, let‘s consider the design of a Web application written using the Common Gateway Interface (CGI) standard, a popular way to write Web applications circa 1998. In those days, when you wrote a CGI application, you did everything yourself — the equivalent of baking a cake from scratch. For example, here‘s a simple CGI script, written in Python, that displays the ten most recently published books from a database:

```
import MySQLdb

print "Content-Type: text/html"
print
print "<html><head><title>Books</title></head>"
print "<body>"
print "<h1>Books</h1>"
print "<ul>"

connection = MySQLdb.connect(user='me', passwd='letmein', db='my_db')
cursor = connection.cursor()
cursor.execute("SELECT name FROM books ORDER BY pub_date DESC LIMIT 10")
for row in cursor.fetchall():
    print "<li>%s</li>" % row[0]

print "</ul>"
print "</body></html>"

connection.close()
```

This code is straightforward. First, it prints a ―Content-Type‖ line, followed by a blank line, as required by CGI. It prints some introductory HTML, connects to a database and executes a query that retrieves the latest ten books. Looping over those books, it generates an HTML unordered list. Finally, it prints the closing HTML and closes the database connection.

With a one-off dynamic page such as this one, the write-it-from-scratch approach isn‘t necessarily bad. For one thing, this code is simple to comprehend — even a novice developer can read these 16 lines of Python and understand all it does, from start to finish. There‘s nothing else to learn; no other code to read. It‘s also simple to deploy: just save this code in a file called latestbooks.cgi, upload that file to a Web server, and visit that page with a browser. But as a Web application grows beyond the trivial, this approach breaks down, and you face a number of problems:

Should a developer really have to worry about printing the ―Content-Type‖ line and remembering to close the database connection? This sort of boilerplate reduces programmer productivity and introduces opportunities for mistakes. These setup- and teardown-related tasks would best be handled by some common infrastructure.

- What happens when this code is reused in multiple environments, each with a separate database and password? At this point, some environment-specific configuration becomes essential.

- What happens when a Web designer who has no experience coding Python wishes to redesign the page? Ideally, the logic of the page — the retrieval of books from the database.

What Is A Script?

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode.

Scripts are reusable

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

You will need a text editor

Just about any text editor will suffice for creating Python script files.
You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

Difference between a script and a program

Script:

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, where as the applications they control are traditionally compiled to native machine code.

Program:

The program has an executable form that the computer can use directly to execute the instructions.

The same program in its human-readable source code form, from which executable programs are derived(e.g., compiled)

5.1 Python

what is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don‘t know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to ―big name‖ languages. Hopefully I can explain it for you.

Python concepts

If your not interested in the the hows and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it‘s a great one to start programming with.

- Open source general-purpose language.
- Object Oriented, Procedural, Functional
- Easy to interface with C/ObjC/Java/Fortran
- Easy-ish to interface with C++ (via SWIG)
- Great interactive environment

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.

Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.

Python is Interactive − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

5.2 History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

5.3 Python Features

Python's features include −

Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.

Easy-to-read − Python code is more clearly defined and visible to the eyes.

Easy-to-maintain − Python's source code is fairly easy-to-maintain.

A broad standard library − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

Databases − Python provides interfaces to all major commercial databases.

GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features, few are listed below −

- It can be used as a scripting language or can be compiled to byte-code for building large It supports functional and structured programming methods as well as OOP.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Dynamic vs Static

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of ―thing‖ each data value is.

For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a ―float‖ type.

This tells the compiler that the only data that can be used for that variable must be a floating point number, i.e. a number with a decimal point.

If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn‘t require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating point number) you need in your program. With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn‘t matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating point number and one is an integer. Python realizes that it‘s more accurate to keep track of decimals so it automatically calculates the result as a floating point number

Variables

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

Standard Data Types

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types −

- Numbers
- String
- List
- Tuple
- Dictionary

Python Numbers

Number data types store numeric values. Number objects are created when you assign a value to them.

Python Strings

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

Python Lists

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([ ] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator.

Python Tuples

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

The main differences between lists and tuples are: Lists are enclosed in brackets ([ ]) and their elements and size can be changed, while tuples are enclosed in parentheses (( )) and cannot be updated. Tuples can be thought of as **read-only** lists.

Python Dictionary

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object. Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

Different modes in python

- Python has two basic modes: normal and interactive.
- The normal mode is the mode where the scripted and finished .py files are run in the Python interpreter.
- Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

Python libraries

- **Requests:** The most famous http library written by kenneth reitz. It‘s a must have for every python developer.
- **Scrapy:** If you are involved in webscraping then this is a must have library for you. After using this library you won‘t use any other.
- **wxPython:** A gui toolkit for python. I have primarily used it in place of tkinter. You will really love it.
- **Pillow:** A friendly fork of PIL (Python Imaging Library). It is more user friendly than PIL and is a must have for anyone who works with images.
- **SQLAlchemy:** A database library. Many love it and many hate it. The choice is yours.
- **BeautifulSoup:** I know it‘s slow but this xml and html parsing library is very useful for beginners.
- **Twisted**: The most important tool for any network application developer. It has a very beautiful api and is used by a lot of famous python developers.
- **NumPy**: How can we leave this very important library ? It provides some advance math functionalities to python.
- **SciPy**: When we talk about NumPy then we have to talk about scipy. It is a library of algorithms and mathematical tools for python and has caused many scientists to switch from ruby to python.
- **matplotlib:** A numerical plotting library. It is very useful for any data scientist or any data analyzer.

- **Pygame:** Which developer does not like to play games and develop them ? This library will help you achieve your goal of 2d game development.
- **Pyglet:** A 3d animation and game creation engine. This is the engine in which the famous python port of minecraft was made
- **pyQT:** A GUI toolkit for python. It is my second choice after wxpython for developing GUI‘s for my python scripts.
- **pyGtk:** Another python GUI library. It is the same library in which the famous Bittorrent client is created.
- **Scapy:** A packet sniffer and analyzer for python made in python.
- **pywin32:** A python library which provides some useful methods and classes for interacting with windows.
- **nltk:** Natural Language Toolkit – I realize most people won‘t be using this one, but it‘s generic enough. It is a very useful library if you want to manipulate strings. But it‘s capacity is beyond that. Do check it out.
- **nose:** A testing framework for python. It is used by millions of python developers. It is a must have if you do test driven development.
- **SymPy:** SymPy can do algebraic evaluation, differentiation, expansion, complex numbers, etc. It is contained in a pure Python distribution.
- **IPython:** I just can‘t stress enough how useful this tool is. It is a python prompt on steroids. It has completion, history, shell capabilities, and a lot more. Make sure that you take a look at it.

Numpy

NumPy‘s main object is the homogeneous multidimensional array. It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In NumPy dimensions are called axes. The number of axes is rank.

- Offers Matlab-ish capabilities within Python
- Fast array operations
- 2D arrays, multi-D arrays, linear algebra etc.

matplotlib

- High quality plotting library.

Python lass and Objects

These are the building blocks of OOP. class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g. a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently ‒the big thing‖ in most programming languages. Hence, there are several chapters dedicated to OOP later in the book. The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something. Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects. When I first learned C and C++, I did great; functions just made sense for me. Having messed around with BASIC in the early ‘90s, I realized functions were just like subroutines so there wasn‘t much new to learn. However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered. Once you learn OOP, you‘ll realize that it‘s actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing. One thing to note about Python and OOP: it‘s not mandatory to use objects in your code in a way that works best; maybe you don‘t need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want. As you‘ve already seen, Python can do just fine with functions. Unlike languages such as Java, you aren‘t tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here‘s a brief list of Python OOP ideas:

- The class statement creates a class object and gives it a name. This creates a new namespace.
- Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: ClassName.Attribute.

• Class attributes export the state of an object and its associated behavior. These attributes are shared by all instances of a class.

• Calling a class (just like a function) creates a new instance of the class. This is where the multiple copies part comes in.

• Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.

• Using the term self identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

Inheritance

First off, classes allow you to modify a program without really making changes to it. To elaborate, by subclassing a class, you can change the behavior of the program by simply adding new components to it rather than rewriting the existing components. As we‘ve seen, an instance of a class inherits the attributes of that class. However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses. The subclasses can override the logic in a superclass, allowing you to change the behavior of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behavior closer to Python‘s object model and the object interfaces are more consistent to Python‘s built-in objects, hence overloading is easier to learn and use. User-made classes can override nearly all of Python‘s built-in operation methods

Exceptions

I‘ve talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program‘s flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn‘t exist, or when the program reaches a marker, such as the completion of a loop. Exceptions, by definition, don‘t occur very often; hence, they are the "exception to the rule" and a special class has been created for them. Exceptions are everywhere in Python. Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples:

- Accessing a non−existent dictionary key will raise a KeyError exception.
- Searching a list for a non−existent value will raise a ValueError exception
- Calling a non−existent method will raise an AttributeError exception.
- Referencing a non−existent variable will raise a NameError exception.
- Mixing datatypes without coercion will raise a TypeError exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files. This is the most common way to use exceptions. When programming with the Python command line interpreter, you don‘t need to worry about catching exceptions. Your program is usually short enough to not be hurt too much if an exception occurs. Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem. However, if the same error occurred in your real program, it will fail and stop working. Exceptions can be created manually in the code by raising an exception. It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don‘t put any overhead on the code processing. Because exceptions aren‘t supposed to happen very often, they aren‘t processed until they occur. Exceptions can be thought of as a special form of the if/elif statements. You can realistically do the same thing with if blocks as you can with exceptions. However, as already mentioned, exceptions aren‘t processed until they occur; if blocks are processed all the time. Proper use of exceptions can help the performance of your program. The more infrequent the error might occur, the better off you are to use exceptions; using if blocks requires Python to always test extra conditions before continuing. Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

- I won‘t spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.
- You probably won‘t have to do this very often but it‘s nice to have the option when necessary.
- However, before making your own exceptions, make sure there isn‘t one of the built-in exceptions that will work for you.
- They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.
- Making your own exceptions involves object-oriented programming, which will be covered in the next chapter to make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g. making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.
- To make a custom exception, simply inherit the base exception and define what it will do.

5.4 Python modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

Testing code

As indicated above, code is usually developed in a file using an editor. To test the code, import it into a Python session and try to run it. Usually there is an error, so you go back to the

file, make a correction, and test again. This process is repeated until you are satisfied that the code works. The entire process is known as the development cycle.There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.
This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

5.5 Functions in Python

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function**.**
You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub-program and called when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs.
When the task is carred out, the function can or can not return one or more values.
There are three types of functions in python:

help(),min(),print().

Python Namespace

Generally speaking, a **namespace** (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a namespacing system from daily life, i.e. the naming of people in firstname and familiy name (surname).
An example is a network: each network device (workstation, server, printer, ...) needs a unique name and address. Yet another example is the directory structure of file systems.
The same file name can be used in different directories, the files can be uniquely accessed via the pathnames. Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace. This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories) Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python:

- **global names :** of a module
- **local names :** in a function or method invocation
- **built-in names:** this namespace contains built-in functions (e.g. abs(), cmp(), ...) and built-in exception names

Garbage Collection

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector. The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

Python XML Parser

XML is a portable, open source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs The Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX : Here, you register callbacks for events of interest and then let the parser proceed through the document.

This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API : This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree − based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

5.6 Python Web Framework

A web framework is a code library that makes a developers life easier when building reliable,scalable and maintainable web applications.

These common operations are include:

1. URL routing
2. HTML,XML,JSON and other output format templating
3. Database manipulation
4. Session storage and retrival

MySQL Connector/Python enables Python programs to access MySQL databases, using an API that is compliant with the Python Database API Specification v2.0 (PEP 249). It is written in pure Python and does not have any dependencies except for the Python Standard Library. For notes detailing the changes in each release of Connector/Python, see MySQL Connector/Python Release Notes.

MySQL Connector/Python includes support for:

- Almost all features provided by MySQL Server up to and including MySQL Server version 5.7.
- Converting parameter values back and forth between Python and MySQL data types, for example Python datetime and MySQL DATETIME. You can turn automatic conversion on for convenience, or off for optimal performance.
- All MySQL extensions to standard SQL syntax.

- Protocol compression, which enables compressing the data stream between the client and server.
- Connections using TCP/IP sockets and on Unix using Unix sockets.
- Secure TCP/IP connections using SSL.
- Self-contained driver. Connector/Python does not require the MySQL client library or any Python modules outside the standard library

DataSets

The dataset object is similar to the ADO recordset object, but more powerful, and with one other important distinction: the dataSet is always disconnected. The dataset object represents a cache of data, with database-like structures such as tables, columns, relationships, and constraints. However, though a dataset can and does behave much like a database, it is important to remember that dataset objects do not interact directly with databases, or other source data. This allows the developer to work with a programming model that is always consistent, regardless of where the source data resides. Data coming from a database, an XML file, from code, or user input can all be placed into dataset objects. Then, as changes are made to the dataset they can be tracked and verified before updating the source data. The getchanges method of the dataset object actually creates a second dataset that contains only the changes to the data. This dataset is then used by a data adapter (or other objects) to update the original data source.
The dataset has many XML characteristics, including the ability to produce and consume XML data and XML schemas. XML schemas can be used to describe schemas interchanged via web services. In fact, a dataset with a schema can actually be compiled for type safety and statement completion.

6. SYSTEM DESIGN

6.1. INTRODUCTION

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer‘s goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word ―Quality‖. Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a customer‘s view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

6.2. Normalization

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updation, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

Insertion anomaly: Inability to add data to the database due to absence of other data.

Deletion anomaly: Unintended loss of data due to deletion of other data.

Update anomaly: Data inconsistency resulting from data redundancy and partial update

Normal Forms: These are the rules for structuring relations that eliminate anomalies.

First Normal Form:

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

Second Normal Form:

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

1) Primary key is a not a composite primary key
2) No non key attributes are present
3) Every non key attribute is fully functionally dependent on full set of primary key.

Third Normal Form:

A relation is said to be in third normal form if their exits no transitive dependencies.

Transitive Dependency:

If two non key attributes depend on each other as well as on the primary key then they are said to be transitively dependent. The above normalization principles were applied to decompose the data in multiple tables thereby making the data to be maintained in a consistent state.

6.3. E – R Diagrams

- The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
- The entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.

- The set of primary components that are identified by the ERD are
    - Data object
    - Relationships
    - Attributes
    - Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships.

6.4. Data Flow Diagrams

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD‘S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process. Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a ‒bubble Chart‖ has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

DFD Symbols:

In the DFD, there are four symbols

1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing .
4. An open rectangle is a data store, data at rest or a temporary repository of data.

Constructing a DFD:

Several rules of thumb are used in drawing DFD‘S:

- Process should be named and numbered for an easy reference. Each name should be representative of the process.
- The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
- When a process is exploded into lower level details, they are numbered.
- The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out. Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

Sailent Features Of DFD’S

- The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
- The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
- The sequence of events is not brought out on the DFD.

Types Of Data Flow Diagrams

1. Current Physical
2. Current Logical
3. New Logical
4. New Physical

Current Physical:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

Current Logical:

The physical aspects at the system are removed as mush as possible so that the current system is reduced to its essence to the data and the processors that transforms them regardless of actual physical form.

New Logical:

This is exactly like a current logical model if the user were completely happy with he user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

New Physical:

The new physical represents only the physical implementation of the new system.

Rules Governing The DFD'S

Process

1) No process can have only outputs.
2) No process can have only inputs. If an object has only inputs than it must be a sink.
3) A process has a verb phrase label.

Data Store

1) Data cannot move directly from one data store to another data store, a process must move data.
2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store
3) A data store has a noun phrase label.

Source OR Sink

The origin and / or destination of data.

1) Data cannot move direly from a source to sink it must be moved by a process
2) A source and /or sink has a noun phrase land.

Data Flow

1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.
2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.
3) A data flow cannot go directly back to the same process it leads. There must be atleast one other process that handles the data flow produce some other data flow returns the original data into the beginning process.
4) A Data flow to a data store means update (delete or change).
5) A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

7. SYSTEM TESTING AND IMPLEMENTATION

7.1 Introduction to Testing

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The increasing visibility of software as a system element and attendant costs associated with a software failure are motivating factors for we planned, through testing. Testing is the process of executing a program with the intent of finding an error. The design of tests for software and other engineered products can be as challenging as the initial design of the product itself.

There of basically two types of testing approaches. One is Black-Box testing – the specified function that a product has been designed to perform, tests can be conducted that demonstrate each function is fully operated. The other is White-Box testing – knowing the internal workings of the product ,tests can be conducted to ensure that the internal operation of the product performs according to specifications and all internal components have been adequately exercised.

White box and Black box testing methods have been used to test this package. The entire loop constructs have been tested for their boundary and intermediate conditions. The test data was designed with a view to check for all the conditions and logical decisions. Error handling has been taken care of by the use of exception handlers.

7.2 Testing Strategies:

Testing is a set of activities that can be planned in advanced and conducted systematically. A strategy for software testing must accommodation low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high-level tests that validate major system functions against customer requirements. Software testing is one element of verification and validation. Verification refers to the set of activities that ensure that software correctly implements as specific function. Validation refers to a different set of activities that ensure that the software that has been built is traceable to customer requirements.

The main objective of software is testing to uncover errors. To fulfill this objective, a series of test steps unit, integration, validation and system tests are planned and executed. Each test step is accomplished through a series of systematic test technique that assist in the design of test cases. With each testing step, the level of abstraction with which software is considered is broadened.

Testing is the only way to assure the quality of software and it is an umbrella activity rather than a separate phase. This is an activity to be preformed in parallel with the software effort and one that consists of its own phases of analysis, design, implementation, execution and maintenance.

Unit Testing:

This testing method considers a module as single unit and checks the unit at interfaces and communicates with other modules rather than getting into details at statement level. Here the module will be treated as a black box, which will take some input and generate output. Outputs for a given set of input combination are pre-calculated and are generated by the module.

System Testing:

Here all the pre tested individual modules will be assembled to create the larger system and tests are carried out at system level to make sure that all modules are working in synchronous with each other. This testing methodology helps in making sure that all modules which are running perfectly when checked individually are also running in cohesion with other modules. For this testing we create test cases to check all modules once and then generated test combinations of test paths through out the system to make sure that no path is making its way into chaos.

Integration Testing

Testing is a major quality control measure employed during software development. Its basic function is to detect errors. Sub functions when combined may not produce than it is desired. Global data structures can represent the problems. Integrated testing is a systematic

technique for constructing the program structure while conducting the tests. To uncover errors that are associated with interfacing the objective is to make unit test modules and built a program structure that has been detected by design. In a non - incremental integration all the modules are combined in advance and the program is tested as a whole. Here errors will appear in an end less loop function. In incremental testing the program is constructed and tested in small segments where the errors are isolated and corrected.

Different incremental integration strategies are top – down integration, bottom – up integration, regression testing.

Top-Down Integration Test

Modules are integrated by moving downwards through the control hierarchy beginning with main program. The subordinate modules are incorporated into structure in either a breadth first manner or depth first manner. This process is done in five steps:

- Main control module is used as a test driver and steps are substituted or all modules directly to main program.
- Depending on the integration approach selected subordinate is replaced at a time with actual modules.
- Tests are conducted.
- On completion of each set of tests another stub is replaced with the real module
- Regression testing may be conducted to ensure that new errors have not been introduced.

This process continuous from step 2 until entire program structure is reached. In top down integration strategy decision making occurs at upper levels in the hierarchy and is encountered first. If major control problems do exists early recognitions is essential.

If depth first integration is selected a complete function of the software may be implemented and demonstrated.

Some problems occur when processing at low levels in hierarchy is required to adequately test upper level steps to replace low-level modules at the beginning of the top down testing. So no data flows upward in the program structure.

Bottom-Up Integration Test

Begins construction and testing with atomic modules. As modules are integrated from the bottom up, processing requirement for modules subordinate to a given level is always available and need for stubs is eliminated. The following steps implements this strategy.

- Low-level modules are combined in to clusters that perform a specific software sub function.
- A driver is written to coordinate test case input and output.
- Cluster is tested.
- Drivers are removed and moving upward in program structure combines clusters.
- Integration moves upward, the need for separate test driver‘s lesions.

If the top levels of program structures are integrated top down, the number of drivers can be reduced substantially and integration of clusters is greatly simplified.

Regression Testing

Each time a new module is added as a part of integration as the software changes. Regression testing is an actually that helps to ensure changes that do not introduce unintended behavior as additional errors. Regression testing maybe conducted manually by executing a subset of all test cases or using automated capture play back tools enables the software engineer to capture the test case and results for subsequent playback and compression. The regression suit contains different classes of test cases. A representative sample to tests that will exercise all software functions. Additional tests that focus on software functions that are likely to be affected by the change.

7.3 Implementation

Implementation is the process of converting a new or revised system design into operational one. There are three types of Implementation:

- Implementation of a computer system to replace a manual system. The problems encountered are converting files, training users, and verifying printouts for integrity.
- Implementation of a new computer system to replace an existing one. This is usually a difficult conversion. If not properly planned there can be many problems.

- Implementation of a modified application to replace an existing one using the same computer. This type of conversion is relatively easy to handle, provided there are no major changes in the files.
- Implementation in Generic tool project is done in all modules. In the first module User level identification is done. In this module every user is identified whether they are genuine one or not to access the database and also generates the session for the user. Illegal use of any form is Strictly avoided.
- In the Table creation module, the tables are created with user specified fields and user can create many table at a time. They may specify conditions, constraints and calculations in creation of tables. The Generic code maintain the user requirements through out the project.
- In Updating module user can update or delete or Insert the new record into the database. This is very important module in Generic code project. User has to specify the file value in the form then the Generic tool automatically gives whole filed values for that particular record.
- In Reporting module user can get the reports from the database in 2Dimentional or 3Dimensional view. User has to select the table and specify the condition then the report will be generated for the user.

Sample Code

1.Phising .py

```
#!/usr/bin/env python
# coding: utf-8

# In[48]:

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report,confusion_matrix
from sklearn.metrics import accuracy_score

# In[47]:

from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.svm import SVC
get_ipython().run_line_magic('matplotlib', 'inline')
sns.set_style('whitegrid')
from sklearn_extensions.extreme_learning_machines.elm import GenELMClassifier
from sklearn_extensions.extreme_learning_machines.random_layer import RBFRandomLayer,
MLPRandomLayer

# In[14]:

phishing=pd.read_csv('phishcoop.csv')

# In[15]:

phishing.head(2)

# In[17]:

X=phishing.iloc[:,:-1]
y=phishing.iloc[:,-1]

# In[18]:

phishing.columns

# In[27]:

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3)


from sklearn.naive_bayes import GaussianNB
```

```
gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)
print("Gaussian Naive Bayes model accuracy(in %):", metrics.accuracy_score(y_test, y_pred)*100)

# In[34]:

print('Train size: {train}, Test size: {test}'.format(train=X_train.shape[0], test=X_test.shape[0]))
```

2.Try_url.py

```
# -*- coding: utf-8 -*-

#importing libraries
from sklearn.externals import joblib
import check_url

#load the pickle file
classifier = joblib.load('completed_models/svm_final.pkl')

#input url
print("enter url")
url = input()

#checking and predicting
checkprediction = check_url.main(url)
prediction = classifier.predict(checkprediction)
print(prediction)
```

3.Checkurl.py

```
# -*- coding: utf-8 -*-

import regex
from tldextract import extract
import ssl
import socket
from bs4 import BeautifulSoup
import urllib3.request
import whois
import datetime


def url_having_ip(url):
#using regular function
#    symbol =
regex.findall(r'(http((s)?)://)((((\d)+).)*)((\w)+)(/((\w)+))?',url)
 #   if(len(symbol)!=0):
  #      having_ip = 1 #phishing
   # else:
    #    having_ip = -1 #legitimate
```

```
    #return(having_ip)
    return 0


def url_length(url):
    length=len(url)
    if(length<54):
        return -1
    elif(54<=length<=75):
        return 0
    else:
        return 1


def url_short(url):
    #ongoing
    return 0

def having_at_symbol(url):
    symbol=regex.findall(r'@',url)
    if(len(symbol)==0):
        return -1
    else:
        return 1

def doubleSlash(url):
    #ongoing
    return 0

def prefix_suffix(url):
    subDomain, domain, suffix = extract(url)
    if(domain.count('-')):
        return 1
    else:
        return -1

def sub_domain(url):
    subDomain, domain, suffix = extract(url)
    if(subDomain.count('.')==0):
        return -1
    elif(subDomain.count('.')==1):
        return 0
    else:
        return 1

def SSLfinal_State(url):
    try:
#check wheather contains https
        if(regex.search('^https',url)):
            usehttps = 1
        else:
            usehttps = 0
#getting the certificate issuer to later compare with trusted issuer
        #getting host name
        subDomain, domain, suffix = extract(url)
```

```
        host_name = domain + "." + suffix
        context = ssl.create_default_context()
        sct = context.wrap_socket(socket.socket(), server_hostname =
host_name)
        sct.connect((host_name, 443))
        certificate = sct.getpeercert()
        issuer = dict(x[0] for x in certificate['issuer'])
        certificate_Auth = str(issuer['commonName'])
        certificate_Auth = certificate_Auth.split()
        if(certificate_Auth[0] == "Network" or certificate_Auth ==
"Deutsche"):
            certificate_Auth = certificate_Auth[0] + " " + certificate_Auth[1]
        else:
            certificate_Auth = certificate_Auth[0]
        trusted_Auth =
['Comodo','Symantec','GoDaddy','GlobalSign','DigiCert','StartCom','Entrust','V
erizon','Trustwave','Unizeto','Buypass','QuoVadis','Deutsche Telekom','Network
Solutions','SwissSign','IdenTrust','Secom','TWCA','GeoTrust','Thawte','Doster'
,'VeriSign']
#getting age of certificate
        startingDate = str(certificate['notBefore'])
        endingDate = str(certificate['notAfter'])
        startingYear = int(startingDate.split()[3])
        endingYear = int(endingDate.split()[3])
        Age_of_certificate = endingYear-startingYear

#checking final conditions
        if((usehttps==1) and (certificate_Auth in trusted_Auth) and
(Age_of_certificate>=1) ):
            return -1 #legitimate
        elif((usehttps==1) and (certificate_Auth not in trusted_Auth)):
            return 0 #suspicious
        else:
            return 1 #phishing

    except Exception as e:

        return 1

def domain_registration(url):
    try:
        w = whois.whois(url)
        updated = w.updated_date
        exp = w.expiration_date
        length = (exp[0]-updated[0]).days
        if(length<=365):
            return 1
        else:
            return -1
    except:
        return 0

def favicon(url):
    #ongoing
    return 0
```

```
def port(url):
    #ongoing
    return 0

def https_token(url):
    subDomain, domain, suffix = extract(url)
    host =subDomain +'.' + domain + '.' + suffix
    if(host.count('https')): #attacker can trick by putting https in domain
part
        return 1
    else:
        return -1

def request_url(url):
    try:
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        imgs = soup.findAll('img', src=True)
        total = len(imgs)

        linked_to_same = 0
        avg =0
        for image in imgs:
            subDomain, domain, suffix = extract(image['src'])
            imageDomain = domain
            if(websiteDomain==imageDomain or imageDomain==''):
                linked_to_same = linked_to_same + 1
        vids = soup.findAll('video', src=True)
        total = total + len(vids)

        for video in vids:
            subDomain, domain, suffix = extract(video['src'])
            vidDomain = domain
            if(websiteDomain==vidDomain or vidDomain==''):
                linked_to_same = linked_to_same + 1
        linked_outside = total-linked_to_same
        if(total!=0):
            avg = linked_outside/total

        if(avg<0.22):
            return -1
        elif(0.22<=avg<=0.61):
            return 0
        else:
            return 1
    except:
        return 0


def url_of_anchor(url):
    try:
```

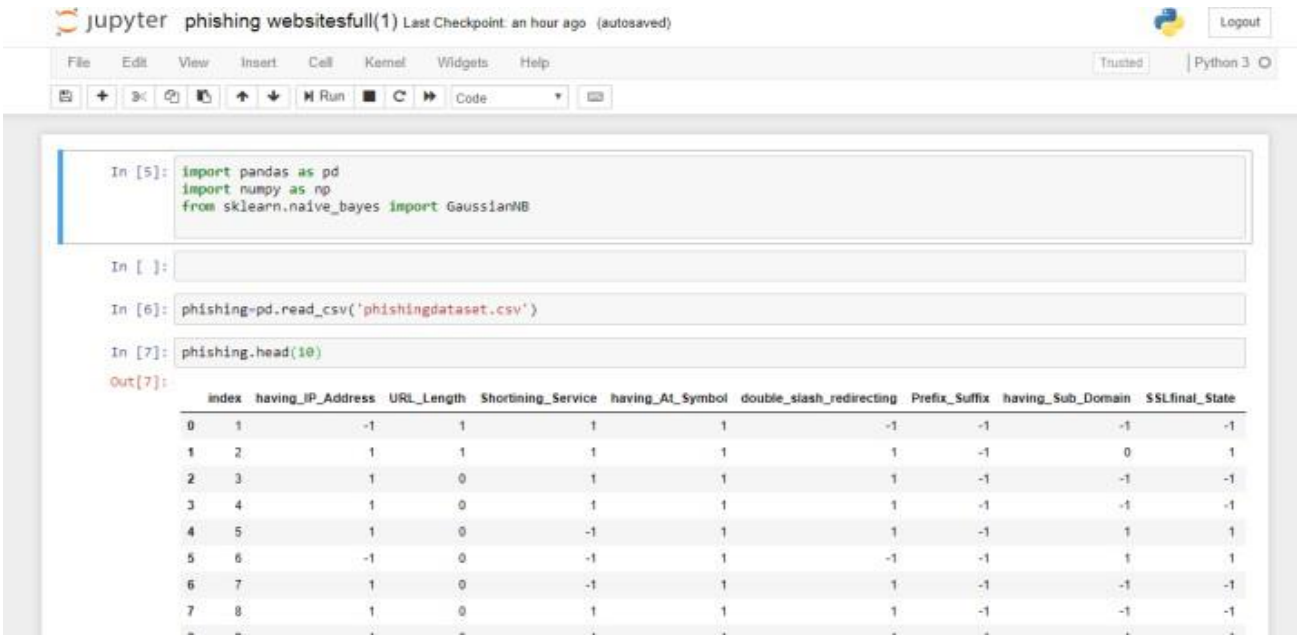
```
        subDomain, domain, suffix = extract(url)
        websiteDomain = domain

        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        anchors = soup.findAll('a', href=True)
        total = len(anchors)
        linked_to_same = 0
        avg = 0
        for anchor in anchors:
            subDomain, domain, suffix = extract(anchor['href'])
            anchorDomain = domain
            if(websiteDomain==anchorDomain or anchorDomain==''):
                linked_to_same = linked_to_same + 1
        linked_outside = total-linked_to_same
        if(total!=0):
            avg = linked_outside/total

        if(avg<0.31):
            return -1
        elif(0.31<=avg<=0.67):
            return 0
        else:
            return 1
    except:
        return 0

def Links_in_tags(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')

        no_of_meta =0
        no_of_link =0
        no_of_script =0
        anchors=0
        avg =0
        for meta in soup.find_all('meta'):
            no_of_meta = no_of_meta+1
        for link in soup.find_all('link'):
            no_of_link = no_of_link +1
        for script in soup.find_all('script'):
            no_of_script = no_of_script+1
        for anchor in soup.find_all('a'):
            anchors = anchors+1
        total = no_of_meta + no_of_link + no_of_script+anchors
        tags = no_of_meta + no_of_link + no_of_script
        if(total!=0):
            avg = tags/total

        if(avg<0.25):
            return -1
        elif(0.25<=avg<=0.81):
            return 0
        else:
```

```
            return 1
    except:
        return 0

def sfh(url):
    #ongoing
    return 0

def email_submit(url):
    try:
        opener = urllib.request.urlopen(url).read()
        soup = BeautifulSoup(opener, 'lxml')
        if(soup.find('mailto:')):
            return 1
        else:
            return -1
    except:
        return 0

def abnormal_url(url):
    #ongoing
    return 0

def redirect(url):
    #ongoing
    return 0

def on_mouseover(url):
    #ongoing
    return 0

def rightClick(url):
    #ongoing
    return 0

def popup(url):
    #ongoing
    return 0

def iframe(url):
    #ongoing
    return 0

def age_of_domain(url):
    try:
        w = whois.whois(url)
        start_date = w.creation_date
        current_date = datetime.datetime.now()
        age =(current_date-start_date[0]).days
        if(age>=180):
            return -1
        else:
            return 1
    except Exception as e:
        print(e)
```

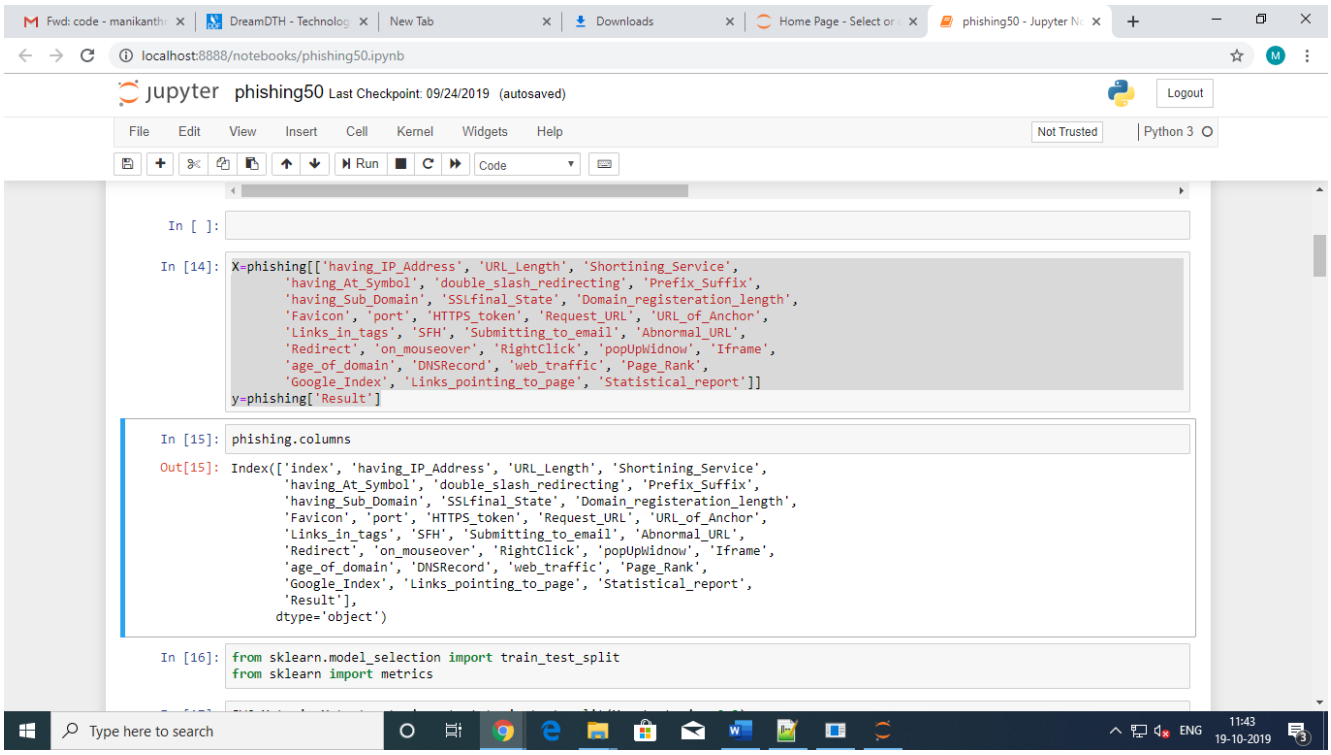
```
        return 0

def dns(url):
    #ongoing
    return 0

def web_traffic(url):
    #ongoing
    return 0

def page_rank(url):
    #ongoing
    return 0

def google_index(url):
    #ongoing
    return 0


def links_pointing(url):
    #ongoing
    return 0

def statistical(url):
    #ongoing
    return 0

def main(url):




    check =
[[url_having_ip(url),url_length(url),url_short(url),having_at_symbol(url),

doubleSlash(url),prefix_suffix(url),sub_domain(url),SSLfinal_State(url),

domain_registration(url),favicon(url),port(url),https_token(url),request_url(u
rl),

url_of_anchor(url),Links_in_tags(url),sfh(url),email_submit(url),abnormal_url(
url),

redirect(url),on_mouseover(url),rightClick(url),popup(url),iframe(url),

age_of_domain(url),dns(url),web_traffic(url),page_rank(url),google_index(url),
              links_pointing(url),statistical(url)]]


    print(check)
    return check
```

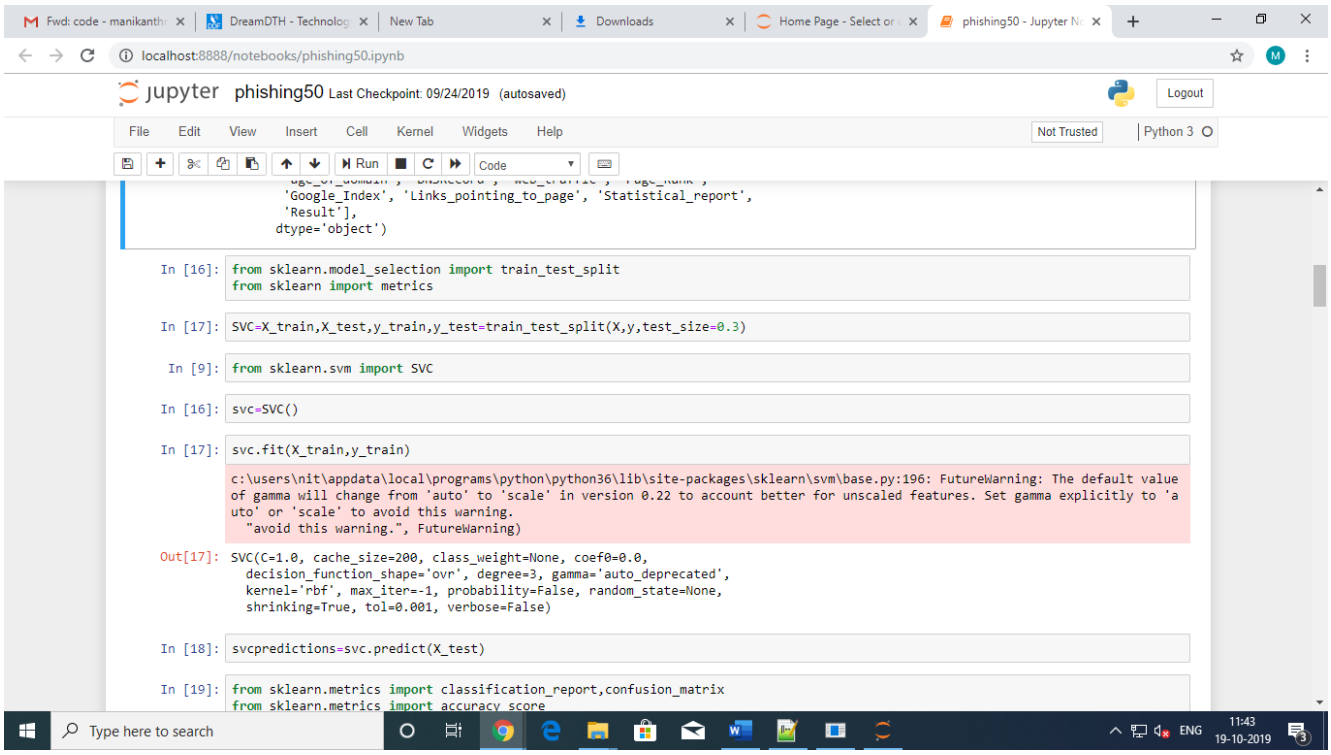
8. OUTPUT SCREENS

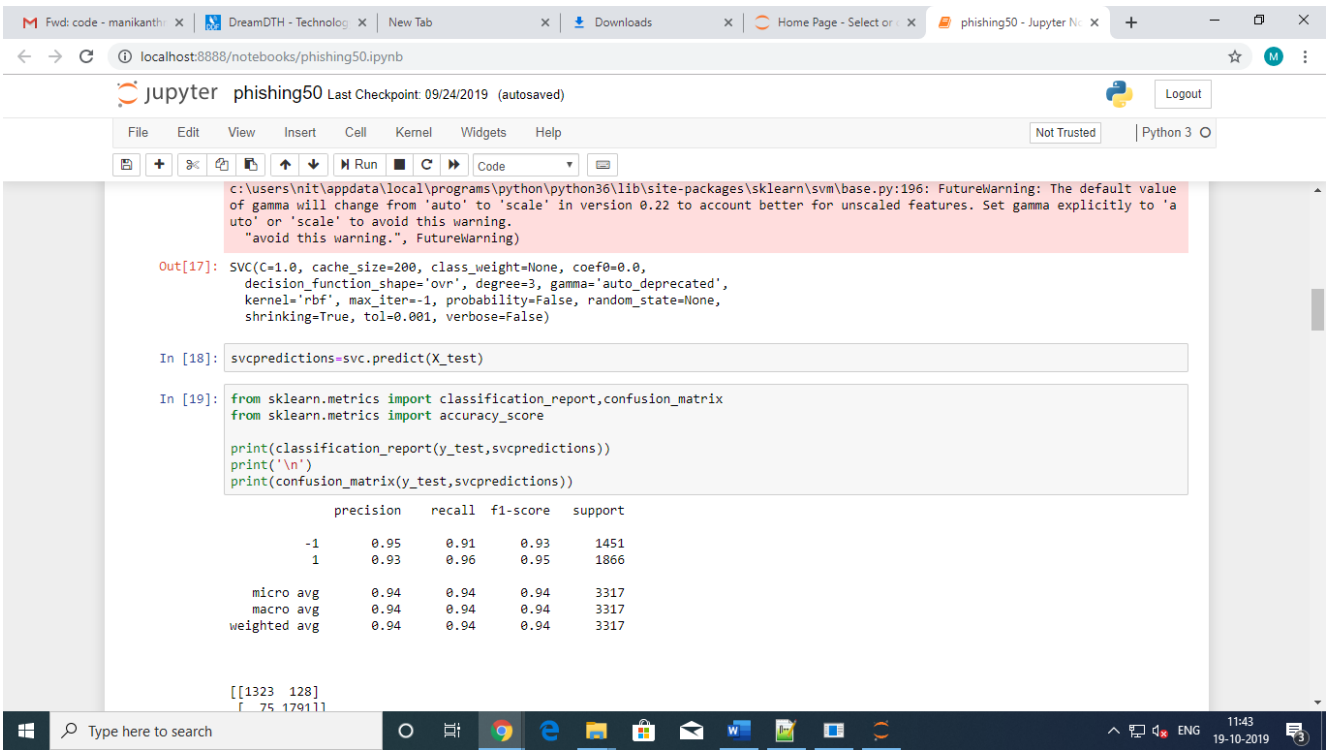
loading data**:**

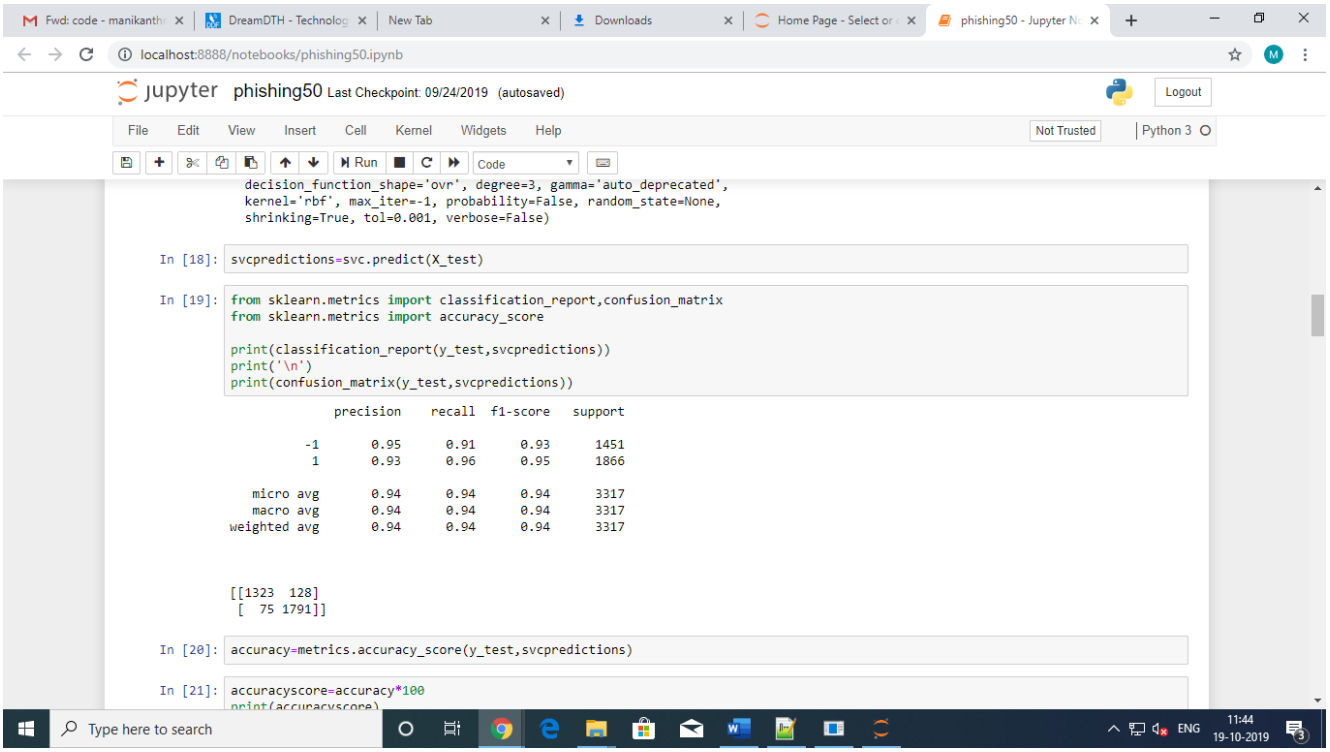


	index	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State
0	1	-1	1	1	1	-1	-1	-1	-1
1	2	1	1	1	1	1	-1	0	1
2	3	1	0	1	1	1	-1	-1	-1
3	4	1	0	1	1	1	-1	-1	-1
4	5	1	0	-1	1	1	-1	1	1
5	6	-1	0	-1	1	-1	-1	1	1
6	7	1	0	-1	1	1	-1	-1	-1
7	8	1	0	1	1	1	-1	-1	-1

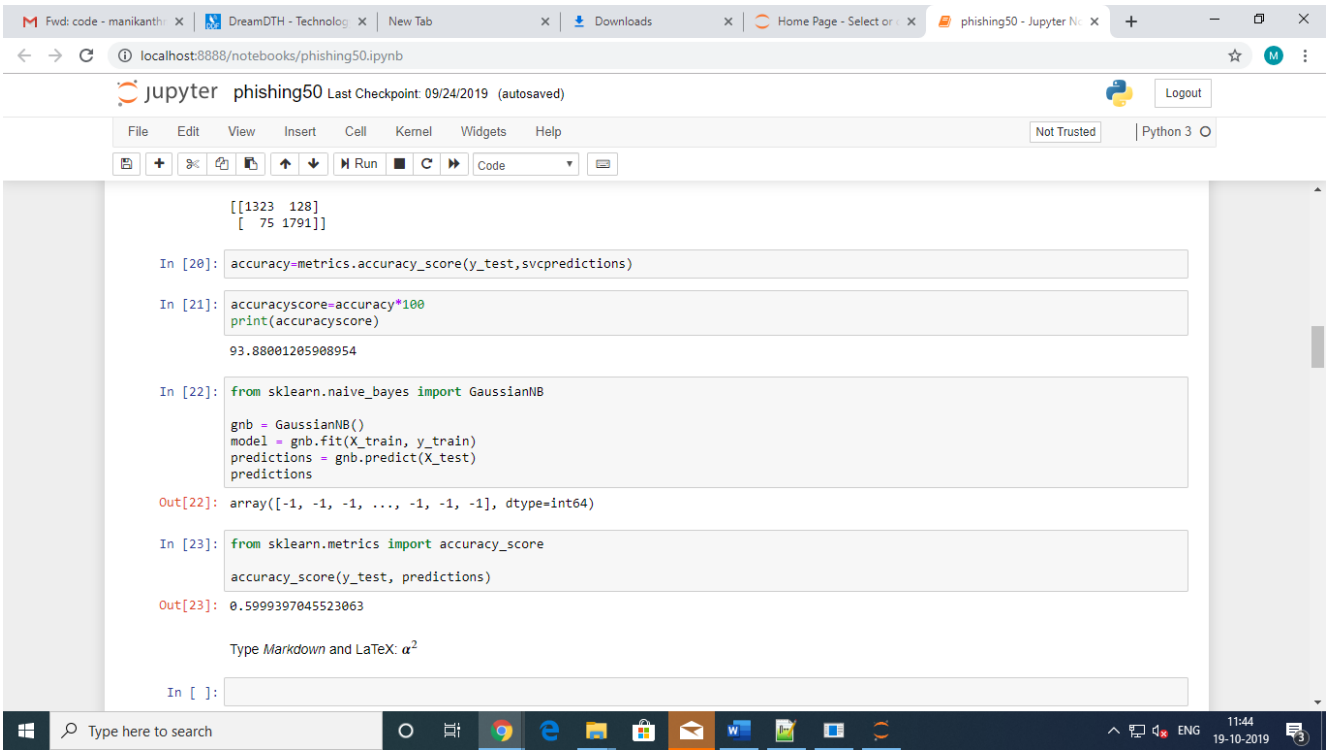



jupyter phishing50 Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
'Google_Index', 'Links_pointing_to_page', 'Statistical_report',
'Result'],
dtype='object')
In [16]: from sklearn.model_selection import train_test_split
from sklearn import metrics
In [17]: SVC=X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3)
In [9]: from sklearn.svm import SVC
In [16]: svc=SVC()
In [17]: svc.fit(X_train,y_train)
c:\users\nit\appdata\local\programs\python\python36\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Out[17]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
kernel='rbf', max_iter=-1, probability=False, random_state=None,
shrinking=True, tol=0.001, verbose=False)
In [18]: svcpredictions=svc.predict(X_test)
In [19]: from sklearn.metrics import classification_report,confusion_matrix
from sklearn.metrics import accuracy_score
Type here to search
ENG
11:43
19-10-2019

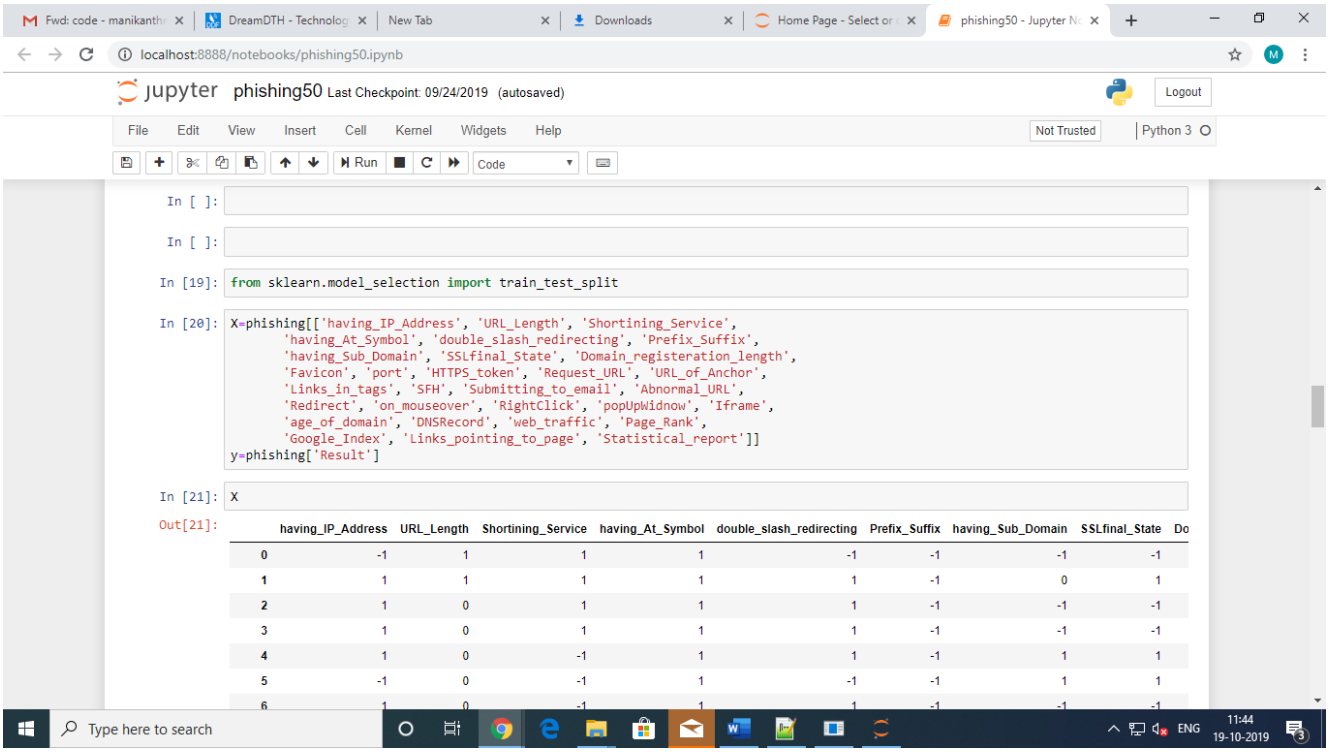

jupyter phishing50 Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
c:\users\nit\appdata\local\programs\python\python36\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid this warning.
"avoid this warning.", FutureWarning)
Out[17]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
kernel='rbf', max_iter=-1, probability=False, random_state=None,
shrinking=True, tol=0.001, verbose=False)
In [18]: svcpredictions=svc.predict(X_test)
In [19]: from sklearn.metrics import classification_report,confusion_matrix
from sklearn.metrics import accuracy_score
print(classification_report(y_test,svcpredictions))
print('\n')
print(confusion_matrix(y_test,svcpredictions))
precision recall f1-score support
-1 0.95 0.91 0.93 1451
1 0.93 0.96 0.95 1866
micro avg 0.94 0.94 0.94 3317
macro avg 0.94 0.94 0.94 3317
weighted avg 0.94 0.94 0.94 3317
[[1323 128]
Type here to search
ENG
11:43
19-10-2019



```
                decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
                kernel='rbf', max_iter=-1, probability=False, random_state=None,
                shrinking=True, tol=0.001, verbose=False)

In [18]: svcpredictions=svc.predict(X_test)

In [19]: from sklearn.metrics import classification_report,confusion_matrix
         from sklearn.metrics import accuracy_score

         print(classification_report(y_test,svcpredictions))
         print('\n')
         print(confusion_matrix(y_test,svcpredictions))

                      precision    recall  f1-score   support

                  -1       0.95      0.91      0.93      1451
                   1       0.93      0.96      0.95      1866

            micro avg       0.94      0.94      0.94      3317
            macro avg       0.94      0.94      0.94      3317
         weighted avg       0.94      0.94      0.94      3317



         [[1323  128]
          [  75 1791]]

In [20]: accuracy=metrics.accuracy_score(y_test,svcpredictions)

In [21]: accuracyscore=accuracy*100
         print(accuracyscore)
```



```
         [[1323  128]
          [  75 1791]]

In [20]: accuracy=metrics.accuracy_score(y_test,svcpredictions)

In [21]: accuracyscore=accuracy*100
         print(accuracyscore)

         93.88001205908954

In [22]: from sklearn.naive_bayes import GaussianNB

         gnb = GaussianNB()
         model = gnb.fit(X_train, y_train)
         predictions = gnb.predict(X_test)
         predictions

Out[22]: array([-1, -1, -1, ..., -1, -1, -1], dtype=int64)

In [23]: from sklearn.metrics import accuracy_score

         accuracy_score(y_test, predictions)

Out[23]: 0.5999397045523063

         Type Markdown and LaTeX: α²

In [ ]:
```



jupyter phishing50 Last Checkpoint: 09/24/2019 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [ ]:

In [ ]:

In [19]:
```
from sklearn.model_selection import train_test_split
```

In [20]:
```
X=phishing[['having_IP_Address', 'URL_Length', 'Shortining_Service',
       'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
       'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
       'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
       'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
       'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
       'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
       'Google_Index', 'Links_pointing_to_page', 'Statistical_report']]
y=phishing['Result']
```

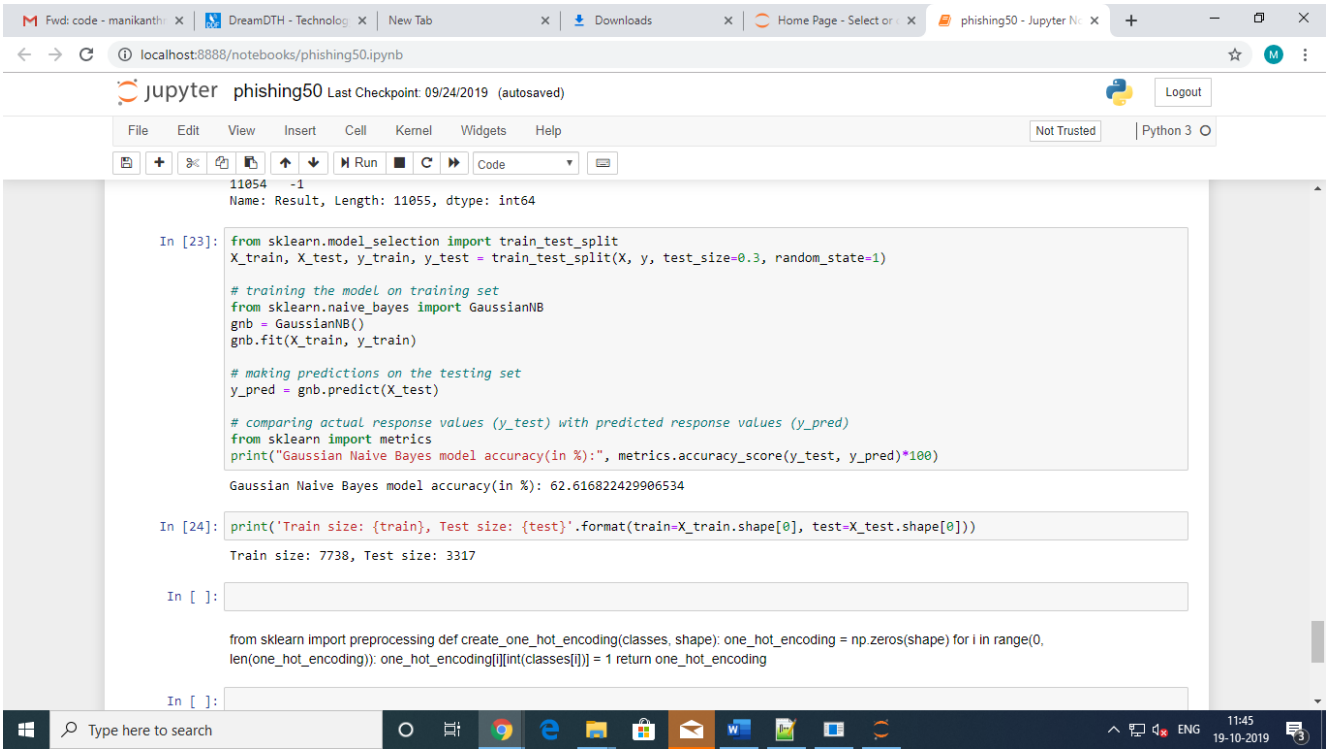
In [21]:
```
X
```

Out[21]:

	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Do
0	-1	1	1	1	-1	-1	-1	-1	
1	1	1	1	1	1	-1	0	1	
2	1	0	1	1	1	-1	-1	-1	
3	1	0	1	1	1	-1	-1	-1	
4	1	0	-1	1	1	-1	1	1	
5	-1	0	-1	1	-1	-1	1	1	
6	1	0	-1	1	1	-1	-1	-1	

Type here to search

ENG 11:44 19-10-2019



jupyter phishing50 Last Checkpoint: 09/24/2019 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

```
11054   -1
Name: Result, Length: 11055, dtype: int64
```

In [23]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=1)

# training the model on training set
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)

# making predictions on the testing set
y_pred = gnb.predict(X_test)

# comparing actual response values (y_test) with predicted response values (y_pred)
from sklearn import metrics
print("Gaussian Naive Bayes model accuracy(in %):", metrics.accuracy_score(y_test, y_pred)*100)
```

```
Gaussian Naive Bayes model accuracy(in %): 62.616822429906534
```

In [24]:
```
print('Train size: {train}, Test size: {test}'.format(train=X_train.shape[0], test=X_test.shape[0]))
```

```
Train size: 7738, Test size: 3317
```

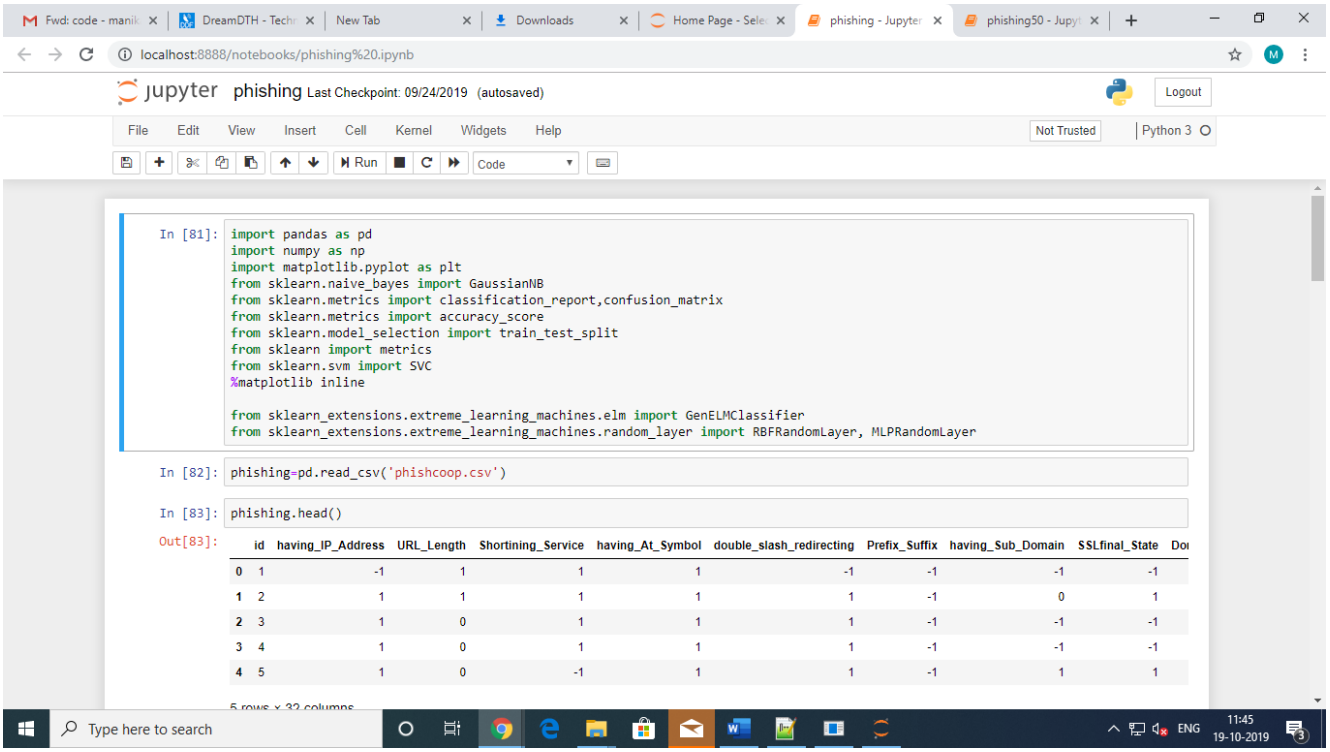
In [ ]:

from sklearn import preprocessing def create_one_hot_encoding(classes, shape): one_hot_encoding = np.zeros(shape) for i in range(0, len(one_hot_encoding)): one_hot_encoding[i][int(classes[i])] = 1 return one_hot_encoding

In [ ]:

Type here to search

ENG 11:45 19-10-2019



jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [81]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import classification_report,confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn import metrics
from sklearn.svm import SVC
%matplotlib inline

from sklearn_extensions.extreme_learning_machines.elm import GenELMClassifier
from sklearn_extensions.extreme_learning_machines.random_layer import RBFRandomLayer, MLPRandomLayer
```

In [82]:
```
phishing=pd.read_csv('phishcoop.csv')
```

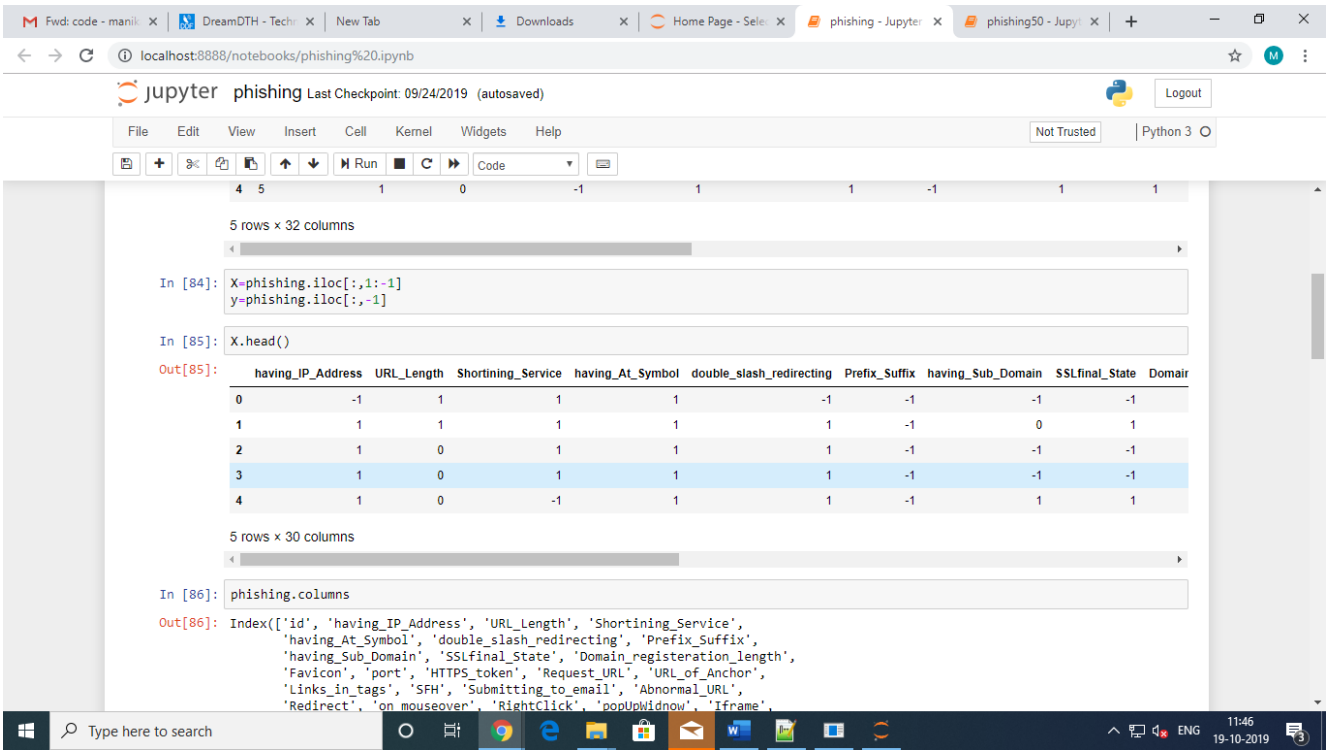
In [83]:
```
phishing.head()
```

Out[83]:

	id	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Do
0	1	-1	1	1	1	-1	-1	-1	-1	
1	2	1	1	1	1	1	-1	0	1	
2	3	1	0	1	1	1	-1	-1	-1	
3	4	1	0	1	1	1	-1	-1	-1	
4	5	1	0	-1	1	1	-1	1	1	

5 rows × 32 columns

Type here to search ENG 11:45 19-10-2019



jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

4	5	1	0	-1	1	1	-1	1	1

5 rows × 32 columns

In [84]:
```
X=phishing.iloc[:,1:-1]
y=phishing.iloc[:,-1]
```

In [85]:
```
X.head()
```

Out[85]:

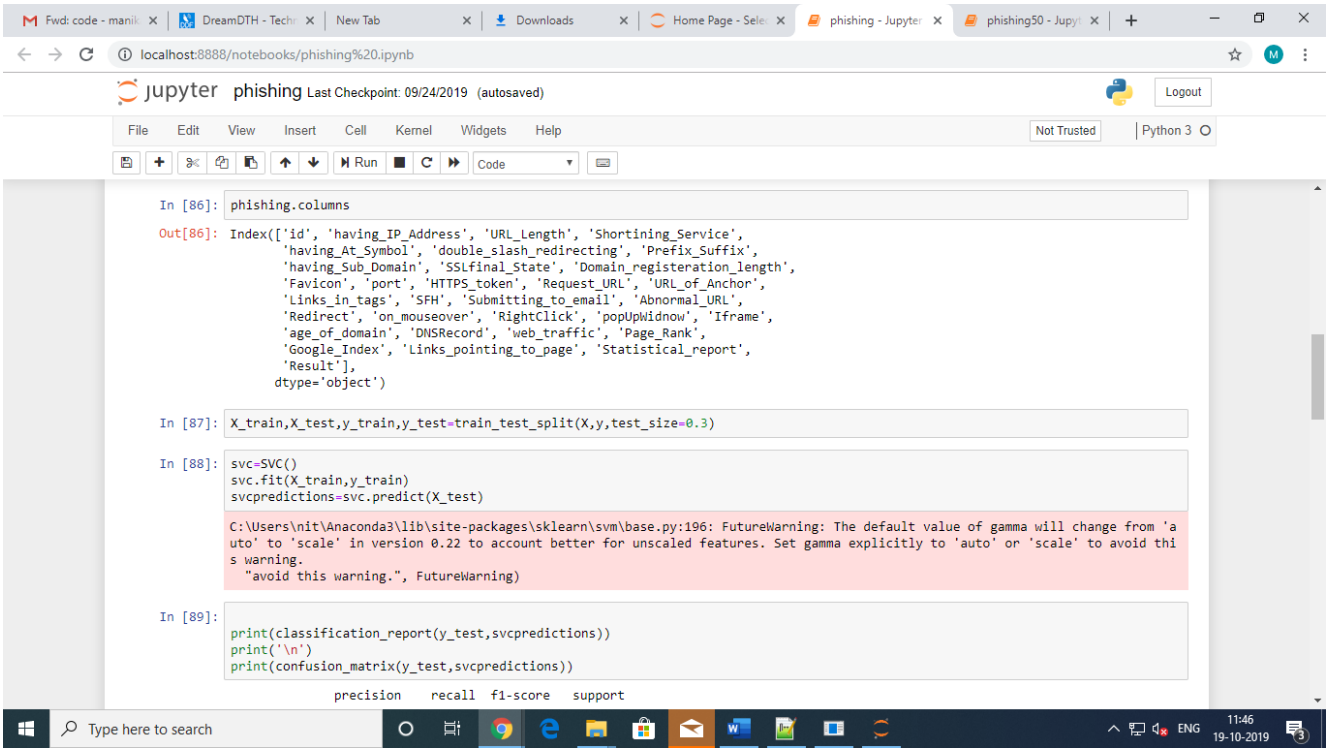
	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Domair
0	-1	1	1	1	-1	-1	-1	-1	
1	1	1	1	1	1	-1	0	1	
2	1	0	1	1	1	-1	-1	-1	
3	1	0	1	1	1	-1	-1	-1	
4	1	0	-1	1	1	-1	1	1	

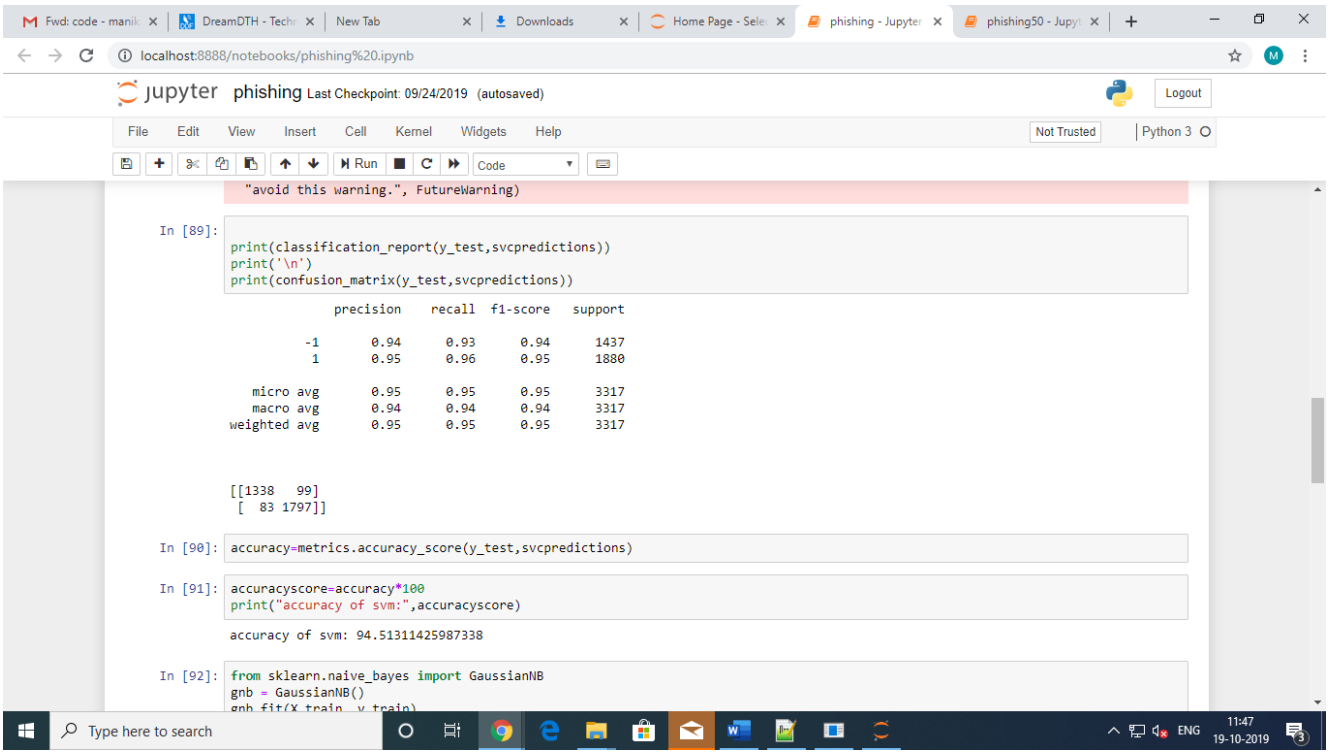
5 rows × 30 columns

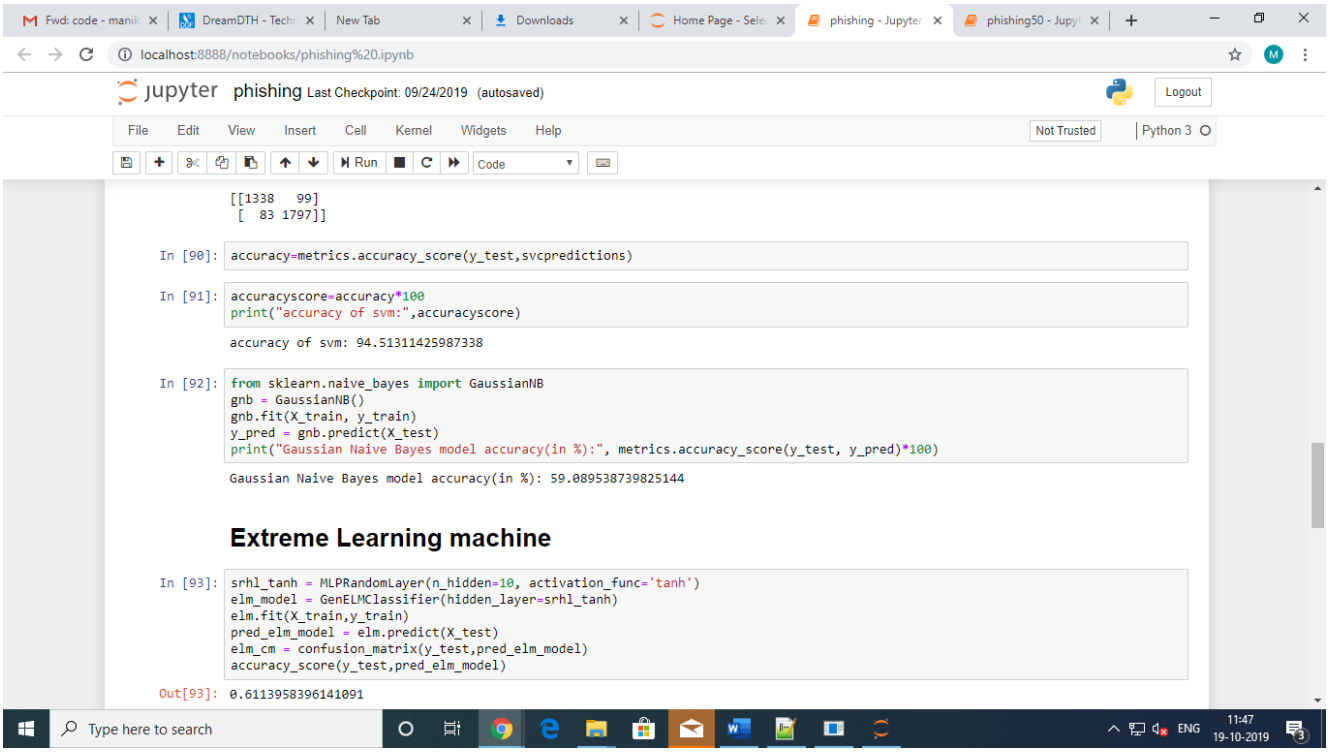
In [86]:
```
phishing.columns
```

Out[86]:
```
Index(['id', 'having_IP_Address', 'URL_Length', 'Shortining_Service',
       'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
       'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
       'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
       'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
       'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
```

Type here to search ENG 11:46 19-10-2019


jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
In [86]:
phishing.columns
Out[86]:
Index(['id', 'having_IP_Address', 'URL_Length', 'Shortining_Service',
'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
'Google_Index', 'Links_pointing_to_page', 'Statistical_report',
'Result'],
dtype='object')
In [87]:
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3)
In [88]:
svc=SVC()
svc.fit(X_train,y_train)
svcpredictions=svc.predict(X_test)
C:\Users\nit\Anaconda3\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value of gamma will change from 'a
uto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'auto' or 'scale' to avoid thi
s warning.
"avoid this warning.", FutureWarning)
In [89]:
print(classification_report(y_test,svcpredictions))
print('\n')
print(confusion_matrix(y_test,svcpredictions))
precision recall f1-score support
Type here to search
ENG
11:46
19-10-2019


jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
"avoid this warning.", FutureWarning)
In [89]:
print(classification_report(y_test,svcpredictions))
print('\n')
print(confusion_matrix(y_test,svcpredictions))
precision recall f1-score support
-1 0.94 0.93 0.94 1437
1 0.95 0.96 0.95 1880
micro avg 0.95 0.95 0.95 3317
macro avg 0.94 0.94 0.94 3317
weighted avg 0.95 0.95 0.95 3317
[[1338 99]
[83 1797]]
In [90]:
accuracy=metrics.accuracy_score(y_test,svcpredictions)
In [91]:
accuracyscore=accuracy*100
print("accuracy of svm:",accuracyscore)
accuracy of svm: 94.51311425987338
In [92]:
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
Type here to search
ENG
11:47
19-10-2019



```
[[1338   99]
 [  83 1797]]
```

In [90]:
```
accuracy=metrics.accuracy_score(y_test,svcpredictions)
```

In [91]:
```
accuracyscore=accuracy*100
print("accuracy of svm:",accuracyscore)
```

```
accuracy of svm: 94.51311425987338
```

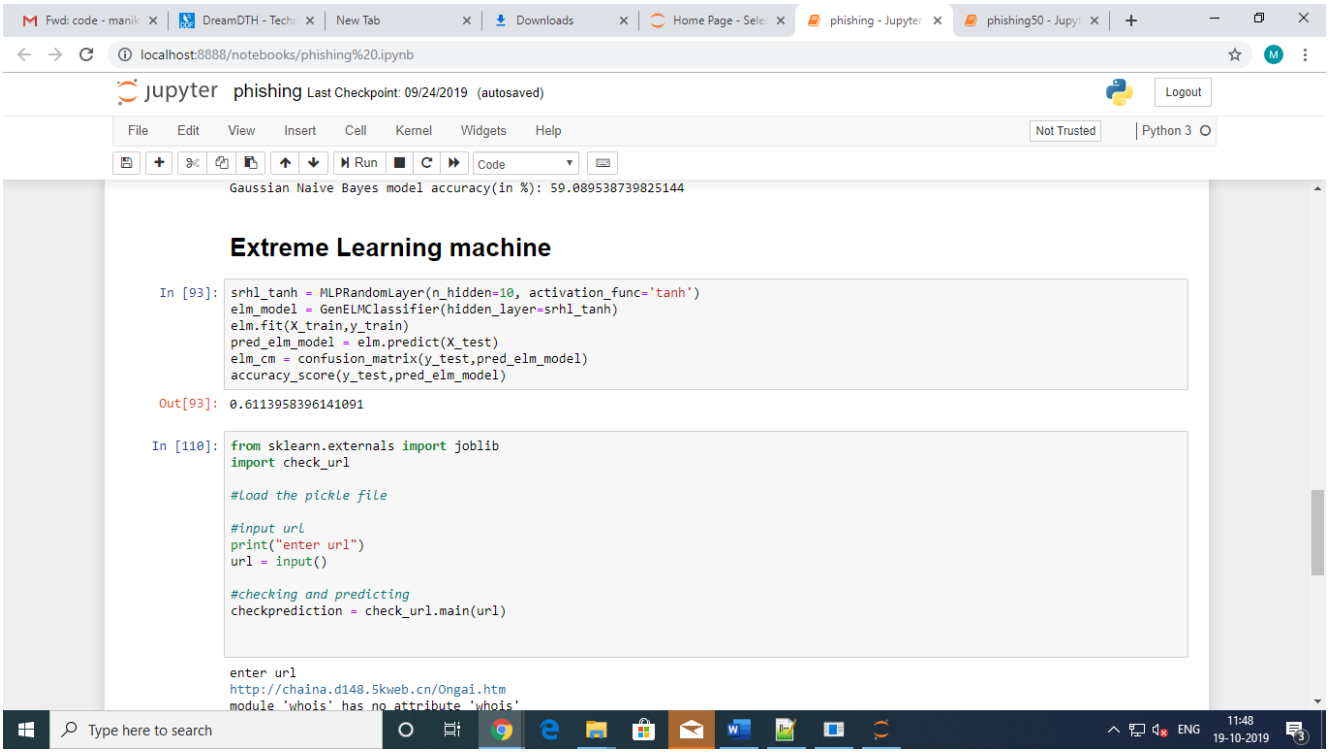
In [92]:
```
from sklearn.naive_bayes import GaussianNB
gnb = GaussianNB()
gnb.fit(X_train, y_train)
y_pred = gnb.predict(X_test)
print("Gaussian Naive Bayes model accuracy(in %):", metrics.accuracy_score(y_test, y_pred)*100)
```

```
Gaussian Naive Bayes model accuracy(in %): 59.089538739825144
```

Extreme Learning machine

In [93]:
```
srhl_tanh = MLPRandomLayer(n_hidden=10, activation_func='tanh')
elm_model = GenELMClassifier(hidden_layer=srhl_tanh)
elm.fit(X_train,y_train)
pred_elm_model = elm.predict(X_test)
elm_cm = confusion_matrix(y_test,pred_elm_model)
accuracy_score(y_test,pred_elm_model)
```

Out[93]:
```
0.6113958396141091
```



```
Gaussian Naive Bayes model accuracy(in %): 59.089538739825144
```

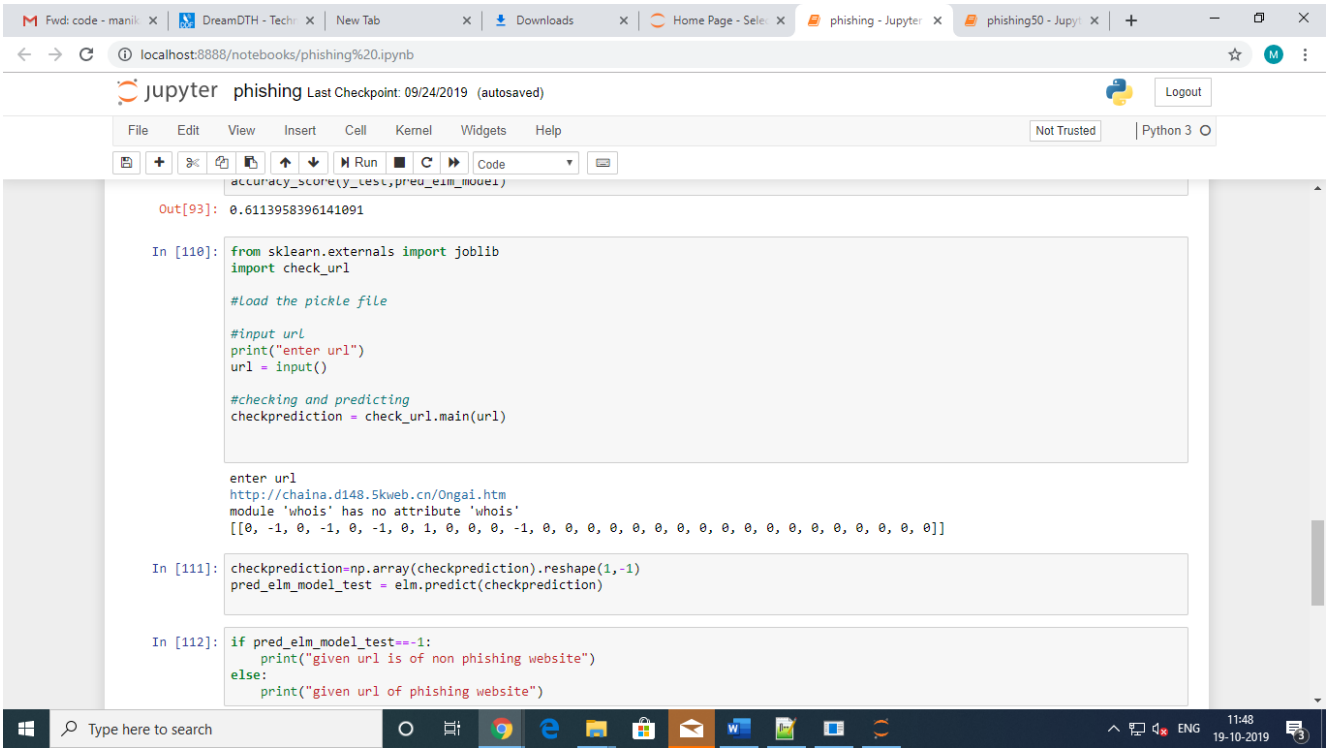
Extreme Learning machine

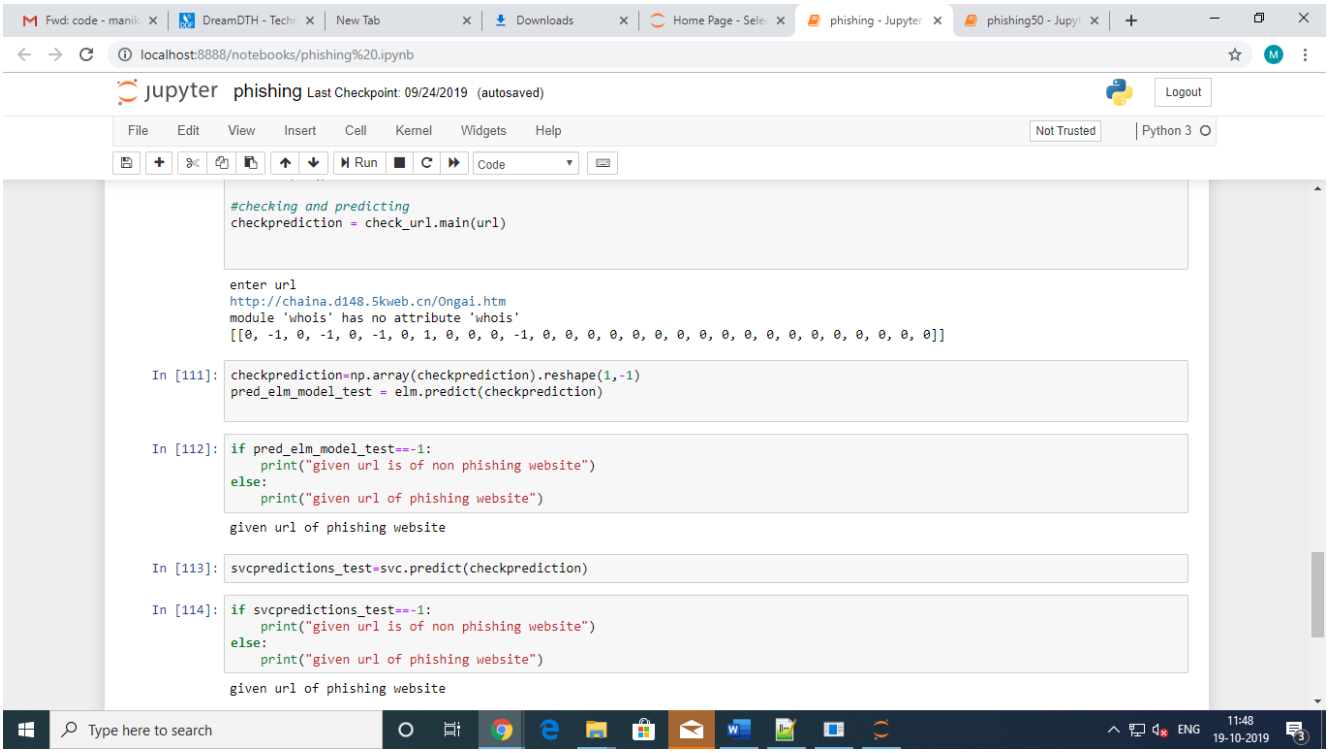
In [93]:
```
srhl_tanh = MLPRandomLayer(n_hidden=10, activation_func='tanh')
elm_model = GenELMClassifier(hidden_layer=srhl_tanh)
elm.fit(X_train,y_train)
pred_elm_model = elm.predict(X_test)
elm_cm = confusion_matrix(y_test,pred_elm_model)
accuracy_score(y_test,pred_elm_model)
```

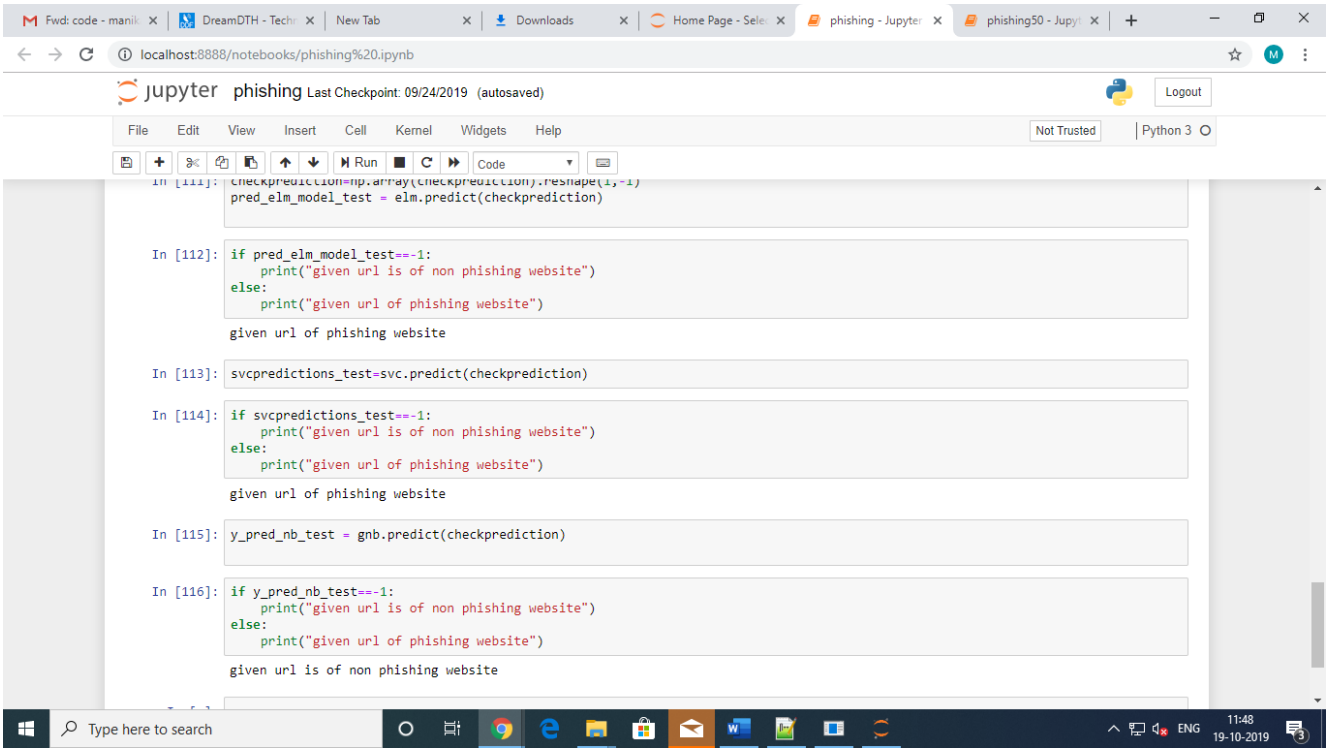
Out[93]:
```
0.6113958396141091
```

In [110]:
```
from sklearn.externals import joblib
import check_url

#Load the pickle file

#input url
print("enter url")
url = input()

#checking and predicting
checkprediction = check_url.main(url)
```

```
enter url
http://chaina.d148.5kweb.cn/Ongai.htm
module 'whois' has no attribute 'whois'
```


jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
Out[93]: 0.6113958396141091
In [110]:
from sklearn.externals import joblib
import check_url
#Load the pickle file
#input url
print("enter url")
url = input()
#checking and predicting
checkprediction = check_url.main(url)
enter url
http://chaina.d148.5kweb.cn/Ongai.htm
module 'whois' has no attribute 'whois'
[[0, -1, 0, -1, 0, -1, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
In [111]:
checkprediction=np.array(checkprediction).reshape(1,-1)
pred_elm_model_test = elm.predict(checkprediction)
In [112]:
if pred_elm_model_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
Type here to search
ENG
11:48
19-10-2019


jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
#checking and predicting
checkprediction = check_url.main(url)
enter url
http://chaina.d148.5kweb.cn/Ongai.htm
module 'whois' has no attribute 'whois'
[[0, -1, 0, -1, 0, -1, 0, 1, 0, 0, 0, -1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]]
In [111]:
checkprediction=np.array(checkprediction).reshape(1,-1)
pred_elm_model_test = elm.predict(checkprediction)
In [112]:
if pred_elm_model_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
given url of phishing website
In [113]:
svcpredictions_test=svc.predict(checkprediction)
In [114]:
if svcpredictions_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
given url of phishing website
Type here to search
ENG
11:48
19-10-2019


localhost:8888/notebooks/phishing%20.ipynb
jupyter phishing Last Checkpoint: 09/24/2019 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code
In [111]:
pred_elm_model_test = elm.predict(checkprediction)
In [112]:
if pred_elm_model_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
given url of phishing website
In [113]:
svcpredictions_test=svc.predict(checkprediction)
In [114]:
if svcpredictions_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
given url of phishing website
In [115]:
y_pred_nb_test = gnb.predict(checkprediction)
In [116]:
if y_pred_nb_test==-1:
print("given url is of non phishing website")
else:
print("given url of phishing website")
given url is of non phishing website
Type here to search
ENG
11:48
19-10-2019

Trail data


In [ ]:
In [8]:
X=phishing[['having_IP_Address', 'URL_Length', 'Shortining_Service',
'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
'Google_Index', 'Links_pointing_to_page', 'Statistical_report']]
y=phishing['Result']
In [9]:
phishing.columns
Out[9]:
Index(['index', 'having_IP_Address', 'URL_Length', 'Shortining_Service',
'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
'Google_Index', 'Links_pointing_to_page', 'Statistical_report',
'Result'],
dtype='object')

```
In [10]: from sklearn.model_selection import train_test_split
         from sklearn import metrics

In [11]: SVC=X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3)

In [12]: from sklearn.svm import SVC

In [13]: svc=SVC()

In [44]: svc.fit(X_train,y_train)

         c:\users\hi\appdata\local\programs\python\python36\lib\site-packages\sklearn\svm\base.py:196: FutureWarning: The default value
         of gamma will change from 'auto' to 'scale' in version 0.22 to account better for unscaled features. Set gamma explicitly to 'a
         uto' or 'scale' to avoid this warning.
           "avoid this warning.", FutureWarning)

Out[44]: SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
           decision_function_shape='ovr', degree=3, gamma='auto_deprecated',
           kernel='rbf', max_iter=-1, probability=False, random_state=None,
           shrinking=True, tol=0.001, verbose=False)

In [45]: svcpredictions=svc.predict(X_test)
```

Accuracy result:

```
In [45]: svcpredictions=svc.predict(X_test)

In [46]: from sklearn.metrics import classification_report,confusion_matrix
         from sklearn.metrics import accuracy_score

         print(classification_report(y_test,svcpredictions))
         print('\n')
         print(confusion_matrix(y_test,svcpredictions))

                       precision    recall  f1-score   support

                   -1       0.96      0.92      0.94      1463
                    1       0.94      0.97      0.95      1854

            micro avg       0.95      0.95      0.95      3317
            macro avg       0.95      0.95      0.95      3317
         weighted avg       0.95      0.95      0.95      3317


         [[1353  110]
          [  62 1792]]

In [47]: accuracy=metrics.accuracy_score(y_test,svcpredictions)
```

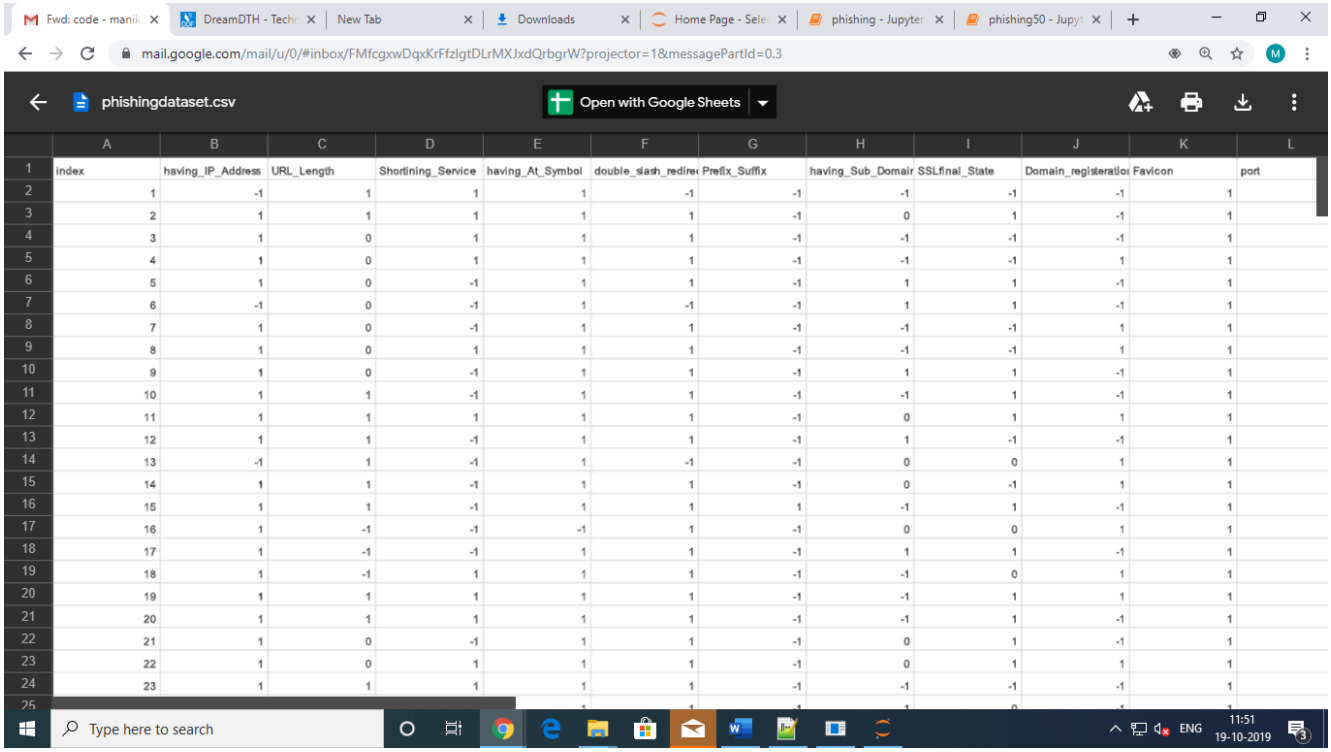
```
In [49]: from sklearn.model_selection import train_test_split

In [50]: X=phishing[['having_IP_Address', 'URL_Length', 'Shortining_Service',
                     'having_At_Symbol', 'double_slash_redirecting', 'Prefix_Suffix',
                     'having_Sub_Domain', 'SSLfinal_State', 'Domain_registeration_length',
                     'Favicon', 'port', 'HTTPS_token', 'Request_URL', 'URL_of_Anchor',
                     'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_URL',
                     'Redirect', 'on_mouseover', 'RightClick', 'popUpWidnow', 'Iframe',
                     'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
                     'Google_Index', 'Links_pointing_to_page', 'Statistical_report']]
         y=phishing['Result']

In [51]: X

Out[51]:
```

	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redirecting	Prefix_Suffix	having_Sub_Domain	SSLfinal_State	Do
0	-1	1	1	1	-1	-1	-1	-1	
1	1	1	1	1	1	-1	0	1	
2	1	0	1	1	1	-1	-1	-1	
3	1	0	1	1	1	-1	-1	-1	
4	1	0	-1	1	1	-1	1	1	
5	-1	0	-1	1	-1	-1	1	1	
6	1	0	-1	1	1	-1	-1	-1	
7	1	0	1	1	1	-1	-1	-1	

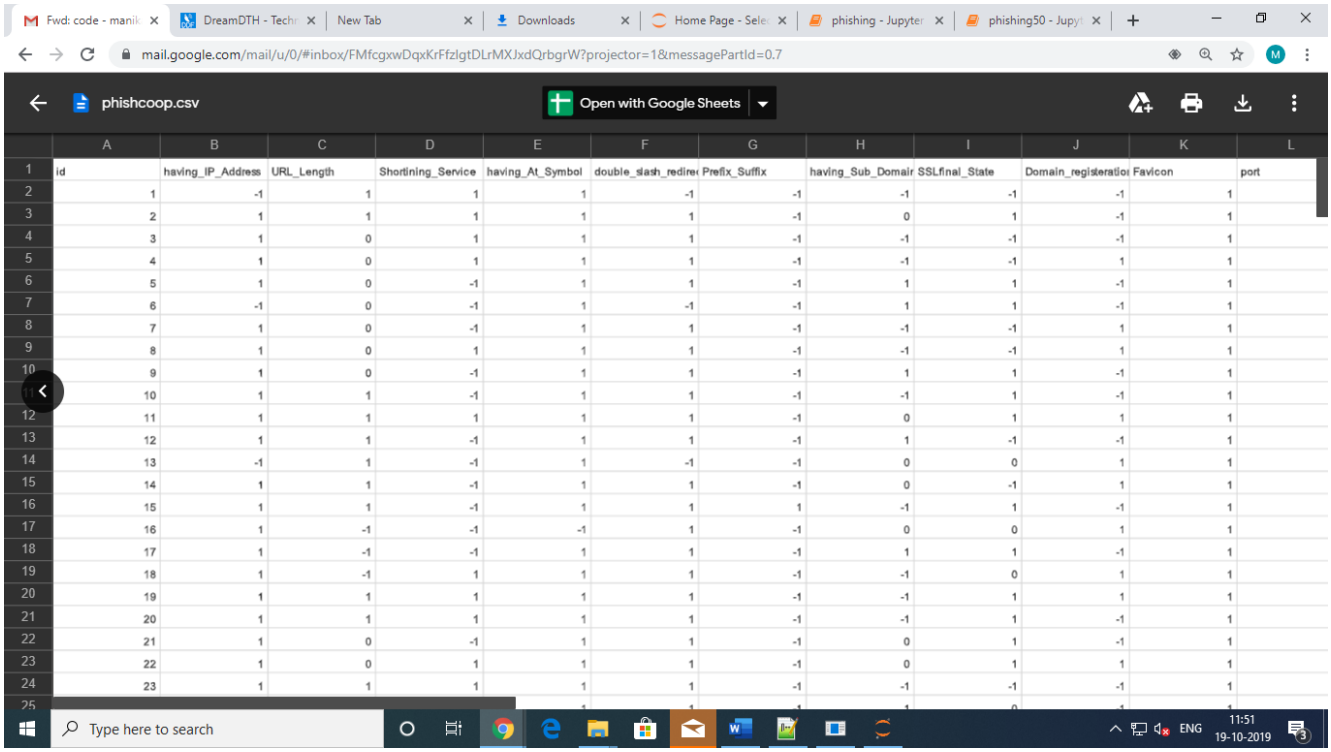
Extreme Learning machine

```
In [55]: input_length = X_train.shape[1]
         hidden_units = 1000

         Win = np.random.normal(size=[input_length, hidden_units])
         print('Input Weight shape: {shape}'.format(shape=Win.shape))

         Input Weight shape: (30, 1000)

In [56]: def input_to_hidden(X):
             a = np.dot(X, Win)
             a = np.maximum(a, 0, a)
             return a

In [57]: X = input_to_hidden(X_train)
         Xt = np.transpose(X)
         Wout = np.dot(np.linalg.inv(np.dot(Xt, X)), np.dot(Xt, y_train))
         print('Output weights shape: {shape}'.format(shape=Wout.shape))

         Output weights shape: (1000,)

In [58]: def predict(x):
             x = input_to_hidden(x)
             y = np.dot(x, Wout)
             return y
```

Data Set Used :



	A	B	C	D	E	F	G	H	I	J	K	L
1	index	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redire	Prefix_Suffix	having_Sub_Domair	SSLfinal_State	Domain_registratio	Favicon	port
2	1	-1	1	1	1	-1	-1	-1	-1	-1	1	
3	2	1	1	1	1	1	-1	0	1	-1	1	
4	3	1	0	1	1	1	-1	-1	-1	-1	1	
5	4	1	0	1	1	1	-1	-1	-1	1	1	
6	5	1	0	-1	1	1	-1	1	1	-1	1	
7	6	-1	0	-1	1	-1	-1	1	1	-1	1	
8	7	1	0	-1	1	1	-1	-1	-1	1	1	
9	8	1	0	1	1	1	-1	-1	-1	1	1	
10	9	1	0	-1	1	1	-1	1	1	-1	1	
11	10	1	1	-1	1	1	-1	-1	1	-1	1	
12	11	1	1	1	1	1	-1	0	1	1	1	
13	12	1	1	-1	1	1	-1	1	-1	-1	1	
14	13	-1	1	-1	1	-1	-1	0	0	1	1	
15	14	1	1	-1	1	1	-1	0	-1	1	1	
16	15	1	1	-1	1	1	1	-1	1	-1	1	
17	16	1	-1	-1	-1	1	-1	0	0	1	1	
18	17	1	-1	-1	1	1	-1	1	1	-1	1	
19	18	1	-1	1	1	1	-1	-1	0	1	1	
20	19	1	1	1	1	1	-1	-1	1	1	1	
21	20	1	1	1	1	1	-1	-1	1	-1	1	
22	21	1	0	-1	1	1	-1	0	1	-1	1	
23	22	1	0	1	1	1	-1	0	1	1	1	
24	23	1	1	1	1	1	-1	-1	-1	-1	1	



	A	B	C	D	E	F	G	H	I	J	K	L
1	id	having_IP_Address	URL_Length	Shortining_Service	having_At_Symbol	double_slash_redire	Prefix_Suffix	having_Sub_Domair	SSLfinal_State	Domain_registratio	Favicon	port
2	1	-1	1	1	1	-1	-1	-1	-1	-1	1	
3	2	1	1	1	1	1	-1	0	1	-1	1	
4	3	1	0	1	1	1	-1	-1	-1	-1	1	
5	4	1	0	1	1	1	-1	-1	-1	1	1	
6	5	1	0	-1	1	1	-1	1	1	-1	1	
7	6	-1	0	-1	1	-1	-1	1	1	-1	1	
8	7	1	0	-1	1	1	-1	-1	-1	1	1	
9	8	1	0	1	1	1	-1	-1	-1	1	1	
10	9	1	0	-1	1	1	-1	1	1	-1	1	
11	10	1	1	-1	1	1	-1	-1	1	-1	1	
12	11	1	1	1	1	1	-1	0	1	1	1	
13	12	1	1	-1	1	1	-1	1	-1	-1	1	
14	13	-1	1	-1	1	-1	-1	0	0	1	1	
15	14	1	1	-1	1	1	-1	0	-1	1	1	
16	15	1	1	-1	1	1	1	-1	1	-1	1	
17	16	1	-1	-1	-1	1	-1	0	0	1	1	
18	17	1	-1	-1	1	1	-1	1	1	-1	1	
19	18	1	-1	1	1	1	-1	-1	0	1	1	
20	19	1	1	1	1	1	-1	-1	1	1	1	
21	20	1	1	1	1	1	-1	-1	1	-1	1	
22	21	1	0	-1	1	1	-1	0	1	-1	1	
23	22	1	0	1	1	1	-1	0	1	1	1	
24	23	1	1	1	1	1	-1	-1	-1	-1	1	

8.SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.1 TYPES OF TESTS

- Unit testing Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

- Integration testing Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components. 100

- Functional test Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items: Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised. Output : identified classes of application outputs must be

exercised. Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined

. • System Test System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

• White Box Testing White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level. 101

• Black Box Testing Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot ―see‖ into it. The test provides inputs and responds to outputs without considering how the software works. Unit Testing: Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases. Test strategy and approach Field testing will be performed manually and functional tests will be written in detail. Test objectives • All field entries must work properly. • Pages must be activated from the identified link. • The entry screen, messages and responses must not be delayed. Features to be tested

• Verify that the entries are of the correct format

• No duplicate entries should be allowed

• All links should take the user to the correct page. Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at

the company level – interact without error. 102 Test Results: All the test cases mentioned above passed successfully. No defects encountered.

- Acceptance Testing User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements. Test Results: All the test cases mentioned above passed successfully. No defects encountered. TESTING METHODOLOGIES The following are the Testing Methodologies: o Unit Testing. o Integration Testing. o User Acceptance Testing. o Output Testing. o Validation Testing.

- Unit Testing Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module‘s control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing. During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

- Integration Testing Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a 103 set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design. The following are the types of Integration Testing: 1.Top Down Integration This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner. In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards. 2. Bottom-up Integration This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:
  - ♣ The low-level modules are combined into clusters into clusters that perform a specific Software sub-

function. ♣ A driver (i.e.) the control program for testing is written to coordinate test case input and output. ♣ The cluster is tested. ♣ Drivers are removed and clusters are combined moving upward in the program structure The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

• User Acceptance Testing User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in 104 touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system.

• Output Testing After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

• Validation Checking Validation checks are performed on the following fields. Text Field The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message. Numeric Field The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested. A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system. 105 Preparation of Test Data Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data: Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves. It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure. Using Artificial Test Data: Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program. The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.The package ―Virtual Private Network‖ has satisfied all the requirements specified as per software requirement specification and was accepted. 106 USER TRAINING Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user‘s requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier. TESTING STRATEGY A strategy for system

testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements. Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING: Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is 107 achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation. UNIT TESTING In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module. In Due Course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

9. SYSTEM SECURITY

9.1. Introduction

The protection of computer based resources that includes hardware, software, data, procedures and people against unauthorized use or natural

Disaster is known as System Security.

System Security can be divided into four related issues:

- Security
- Integrity
- Privacy
- Confidentiality

System Security: refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

Data Security: is the protection of data from loss, disclosure, modification and destruction.

System Integrity: refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

Privacy: defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

Confidentiality: is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

9.2. Security In Software

System security refers to various validations on data in form of checks and controls to avoid the system from failing. It is always important to ensure that only valid data is entered and only valid operations are performed on the system. The system employees two types of checks and controls:

Client Side Validation

Various client side validations are used to ensure on the client side that only valid data is entered. Client side validation saves server time and load to handle invalid data. Some checks imposed are:

- JavaScript in used to ensure those required fields are filled with suitable data only. Maximum lengths of the fields of the forms are appropriately defined.
- Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out at the client side to save the server time and load.
- Tab-indexes are set according to the need and taking into account the ease of user while working with the system.

Server Side Validation

Some checks cannot be applied at client side. Server side checks are necessary to save the system from failing and intimating the user that some invalid operation has been performed or the performed operation is restricted. Some of the server side checks imposed is:

- Server side constraint has been imposed to check for the validity of primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results into a message intimating the user about those values through the forms using foreign key can be updated only of the existing foreign key values.
- User is intimating through appropriate messages about the successful operations or exceptions occurring at server side.
- Various Access Control Mechanisms have been built so that one user may not agitate upon another. Access permissions to various types of users are controlled according to the organizational structure. Only permitted users can log on to the system and can have access according to their category. User- name, passwords and permissions are controlled o the server side.
- Using server side validation, constraints on several restricted operations are imposed.

10. CONCLUSION

In this paper, we defined features of phishing attack and we proposed a classification model in order to classification of the phishing attacks. This method consists of feature extraction from websites and classification section. In the feature extraction, we have clearly defined rules of phishing feature extraction and these rules have been used for obtaining features. In order to classification of these feature, SVM, NB and ELM were used. In the ELM, 6 different activation functions were used and ELM achieved highest accuracy score.

11. FUTURE SCOPE

The present project is aimed at classification of phishing websites based on the features. For that we have taken the phishing dataset which collected from uci machine learning repository and we built our model with three different classifiers like SVC, Naïve Bayes, ELM and we got good accuracy scores. There is a scope to enhance it further .if we can have more data our project will be much more effective and we can get very good results. For this we need API integrations go get the data of different websites.

12. BIBLIOGRAPHY

For software installation:

https://www.anaconda.com/download/

https://www.python.org/downloads/release/python-360/

Modules:

- Install numpy
- Install pandas
- Install matplotlib
- Install scikit – learn

References:

[1] G. Canbek and ù. Sa'Õro'lu, ‒A Review on Information, Information Security and Security Processes,‖ Politek. Derg., vol. 9, no. 3, pp. 165– 174, 2006.

[2] L. McCluskey, F. Thabtah, and R. M. Mohammad, ‒Intelligent rule- based phishing websites classification,‖ IET Inf. Secur., vol. 8, no. 3, pp. 153–160, 2014. [3] R. M. Mohammad, F. Thabtah, and L. McCluskey, ‒Predicting phishing websites based on self-structuring neural network,‖ Neural Comput. Appl., vol. 25, no. 2, pp. 443–458, 2014.

[4] R. M. Mohammad, F. Thabtah, and L. McCluskey, ‒An assessment of features related to phishing websites using an automated technique,‖ Internet Technol. …, pp. 492–497, 2012.

[5] W. Hadi, F. Aburub, and S. Alhawari, ‒A new fast associative classification algorithm for detecting phishing websites,‖ Appl. Soft Comput. J., vol. 48, pp. 729–734, 2016.

[6] N. Abdelhamid, ‒Multi-label rules for phishing classification,‖ Appl. Comput. Informatics, vol. 11, no. 1, pp. 29–46, 2015.

[7] N. Sanglerdsinlapachai and A. Rungsawang, ‒Using domain top-page similarity feature in machine learning-based web phishing detection,‖ in 3rd International Conference on Knowledge Discovery and Data Mining, WKDD 2010, 2010, pp. 187–190.

[8] W. D. Yu, S. Nargundkar, and N. Tiruthani, ‒A phishing vulnerability analysis of web based systems,‖ IEEE Symp. Comput. Commun. (ISCC 2008), pp. 326–331, 2008.

[9] P. Ying and D. Xuhua, ―Anomaly based web phishing page detection,‖ in Proceedings - Annual Computer Security Applications Conference, ACSAC, 2006, pp. 381–390.

[10] M. Moghimi and A. Y. Varjani, ―New rule-based phishing detection method,‖ Expert Syst. Appl., vol. 53, pp. 231–242, 2016.

[11] DATASET: Lichman, M. (2013). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science

[12] G.-B. Huang et al., ―Extreme learning machine: Theory and applications,‖ Neurocomputing, vol. 70, no. 1–3, pp. 489–501, 2006.

[13] C. S. Guang-bin Huang, Qin-yu Zhu, ―Extreme learning machine: A new learning scheme of feedforward neural networks,‖ Neurocomputing, vol. 70, pp. 489–501, 2006.

[14] T. S. Guzella and W. M. Caminhas, ―A review of machine learning approaches to Spam filtering,‖ Expert Systems with Applications, vol. 36, no. 7. pp. 10206–10222, 2009.

[15] Ö. F.. Ertu÷rul, AúÕrÕ Ö÷renme Makineleri ile biyolojik sinyallerin gizli kaynaklarÕna ayrÕútÕrÕlmasÕ. D.Ü. Mühendislik Dergisi Cilt: 7, 1, 3-9- 2016 [16] M. E. Tagluk, M. S. Mamiú, M. Arkan, and Ö. F. Ertugrul, ―Aúiri Ögrenme Makineleri ile Enerji Iletim Hatlari Ariza Tipi ve Yerinin Tespiti,‖ in 2015 23rd Signal Processing and Communications Applications Conference, SIU 2015 - Proceedings, 2015, pp. 1090– 1093.

[17] Ö. Faruk Ertu'rul and Y. Kaya, ―A detailed analysis on extreme learning machine and novel approaches based on ELM,‖ Am. J. Comput. Sci. Eng., vol. 1, no. 5, pp. 43–50, 2014.

[18] Ö. F. Ertugrul, ―Forecasting electricity load by a novel recurrent extreme learning machines approach,‖ Int. J. Electr. Power Energy Syst., vol. 78, pp. 429–435, 2016.

[19] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, ―Extreme learning machine: Theory and applications,‖ Neurocomputing, vol. 70, no. 1, pp. 489–501,2006.