

A MINI PROJECT REPORT ON

Prediction of Phishing web site using Machine Learning Algorithm

*Submitted to the
University of Madras
in partial fulfillment of the requirements
for the award of the degree of*

**MASTER OF SCIENCE
IN
INFORMATION TECHNOLOGY**

By

**SARVESHVER S
REG NO:35120010**

Under The Guidance And Supervision Of

**Mr. M. NITHYANANDAN M.Sc., M.Phil.,
Guest Lecturer**



**DEPARTMENT OF NETWORK SYSTEMS AND INFORMATION
TECHNOLOGY**

UNIVERSITY OF MADRAS

GUINDY CAMPUS

CHENNAI – 600 025

NOVEMBER-2021

**DEPARTMENT OF NETWORK SYSTEMS AND
INFORMATION TECHNOLOGY**



CERTIFICATE

This is to Certify that this report titled **PREDICTION OF PHISHING WEB SITE USING MACHINE LEARNING ALGORITHM** a bonafide record of the internship project work done by Mr. **S. SARVESHVAR (Reg no:35120010)** towards partial fulfillment of the requirement for award of the Degree of **M.Sc. Information Technology**, University of Madras, Guindy Campus, Chennai - 600025, during the academic year 2021-2022.

Head In-Charge of the Department

(Prof. P. Thangavel M.Sc.,M.Tech.,Ph.D.,)

Internship Project Guide

(Mr. M. Nithyanandan M.Sc., MPhil.,)

Submitted to the Viva voce Examination held on _____

EXAMINER

ABSTRACT

The Internet has become an indispensable part of our life, However, It also has provided opportunities to anonymously perform malicious activities like Phishing. Phishers try to deceive their victims by social engineering or creating mock-up websites to steal information such as account ID, username, password from individuals and organizations. Although many methods have been proposed to detect phishing websites, Phishers have evolved their methods to escape from these detection methods. One of the most successful methods for detecting these malicious activities is Machine Learning. This is because most Phishing attacks have some common characteristics which can be identified by machine learning methods. Using skykit collect the data set for our project In this project we find the accuracy of website .

Additionally, discuss the performance from the given datasets with evaluation of classification report and identify the confusion matrix. To propose a machine learning-based method to accurately predict the phishing web site or not by given attributes in the form of best accuracy from comparing supervise classification machine learning algorithms.

Keywords: Dataset, Phishing website, Random forest, Python, machine learning.

TABLE OF CONTENT

S.NO	CONTENT	PAGE.NO
	ABSTRACT	1
01.	INTRODUCTION	4
	1.1 PROJECT OVERVIEW	5
02.	SYSTEM ANALYSIS	8
	2.1 FEASIBILITY STUDY	9
	2.2 EXISTING SYSTEM	10
	2.3 PROPOSED SYSTEM	11
03.	SYSTEM CONFIGURATION	12
	3.1 HARDWARE SPECIFICATION	13
	3.2 SOFTWARE SPECIFICATION	13
	3.3 ABOUT THE SOFTWARE	13
04.	DESIGN METHODOLOGY	23
	4.1 SYSTEM DESIGN	24
	4.2 WORK FLOW DIAGRAM	25
	4.3 USER CASE DIAGRAM	26
05.	MODULES DESCRIPTION	28
06.	ALGORITHM AND TESTING	37
	6.1 ALGORITHM	38
	6.2 TESTING	40
	6.3 TYPES OF TESTING	41
07.	SCREENSHOTS	44

<i>08.</i>	<i>CONCLUSION</i>	<i>55</i>
<i>09.</i>	<i>BIBLIOGRAPHY</i>	<i>57</i>

LIST OF FIGURES

<i>FIGURES.NO</i>	<i>FIGURES NAME</i>	<i>PAGE.NO</i>
<i>1.1</i>	<i>PROCESS OF MACHINE LEARNING</i>	<i>6</i>
<i>1.2.2</i>	<i>PREPARING OF DATA</i>	<i>7</i>
<i>4.1.1</i>	<i>SYSTEM ARCHITECTURE</i>	<i>24</i>
<i>4.2</i>	<i>FLOW DIAGRAM</i>	<i>25</i>
<i>4.3</i>	<i>USER CASE DIAGRAM</i>	<i>26</i>
<i>4.3.1</i>	<i>SEQUENCE DIAGRAM</i>	<i>27</i>
<i>5.1.3</i>	<i>DATA SPLITTING</i>	<i>32</i>
<i>6.1.2</i>	<i>K NEIREST NAIBOUR</i>	<i>40</i>
<i>7.1</i>	<i>DATA VALIDATION PROCESS</i>	<i>45</i>
<i>7.1.2</i>	<i>VISUALIZAGTION</i>	<i>47</i>
<i>7.1.3</i>	<i>APPLYING ALGORITHM</i>	<i>26</i>

CHAPTER 1

INTRODUCTION

CHAPTER -1

1. INTRODUCTION

1.1 PROJECT OVERVIEW

Machine learning is to predict the future from past data. Machine learning (ML) is a type of artificial intelligence (AI) that provides computers with the ability to learn without being explicitly programmed. Machine learning focuses on the development of Computer Programs that can change when exposed to new data and the basics of Machine Learning, implementation of a simple machine learning algorithm using python.

Process of training and prediction involves use of specialized algorithms. It feed the training data to an algorithm, and the algorithm uses this training data to give predictions on a new test data. Machine learning can be roughly separated in to three categories. There are supervised learning, unsupervised learning and reinforcement learning. Supervised learning program is both given the input data and the corresponding labelling to learn data has to be labelled by a human being beforehand. Unsupervised learning is no labels. It provided to the learning algorithm. This algorithm has to figure out the clustering of the input data. Finally, Reinforcement learning dynamically interacts with its environment and it receives positive or negative feedback to improve its performance.

Data scientists use many different kinds of machine learning algorithms to discover patterns in python that lead to actionable insights. At a high level, these different algorithms can be classified into two groups based on the way they “learn” about data to make predictions: supervised and unsupervised learning. Classification is the process of predicting the class of given data points. Classes are sometimes called as targets/ labels or categories. Classification predictive modelling is the task of approximating a mapping function from input variables(X) to discrete output variables(y). In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation.

This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, bio metric identification, document classification etc.

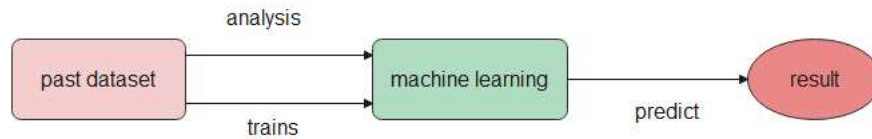


Fig 1.1.1 process of machine learning

Supervised Machine Learning is the majority of practical machine learning uses supervised learning. Supervised learning is where have input variables (X) and an output variable (y) and use an algorithm to learn the mapping function from the input to the output is $y = f(X)$.

The goal is to approximate the mapping function so well that when you have new input data (X) that you can predict the output variables (y) for that data. Techniques of Supervised Machine Learning algorithms include logistic regression, multi-class classification, Decision Trees and support vector machines etc. Supervised learning requires that the data used to train the algorithm is already labelled with correct answers. Supervised learning problems can be further grouped into Classification problems. This problem has as goal the construction of a succinct model that can predict the value of the dependent attribute from the attribute variables.

The difference between the two tasks is the fact that the dependent attribute is numerical for categorical for classification. A classification model attempts to draw some conclusion from observed values. Given one or more inputs a classification model will try to predict the value of one or more outcomes. A classification problem is when the output variable is a category, such as “red” or “blue”.

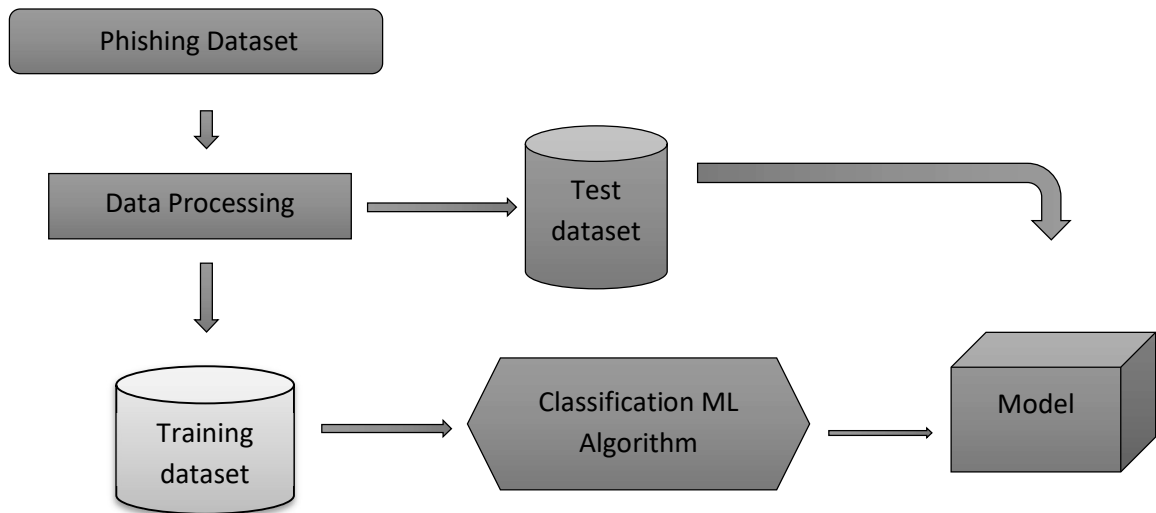


Fig1.1.2 Preparing data set

PREPARING THE DATASET

The dataset is now supplied to machine learning model on the basis of this data set the model is trained.

- It will getting from online source like www.kaggle.com.

CHAPTER 2

SYSTEM ANALYSIS

CHAPTER-2

2. SYSTEM ANALYSIS

2.1 FEASIBILITY STUDY

- A feasibility report is a testimony that attempts to create some sort of action. A feasibility report also determines whether or not the investigated task can be done with the amount of resources available OR how many resources will be necessary in order to complete the task.
- The results determine whether the solution should be implemented.
- This activity takes place during the project initiation phase and is made before significant expenses are engaged

DEFINITION OF FEASIBILITY STUDY

- A feasibility study is an evaluation of a proposal designed to determine the difficulty in carrying out a designated task. Generally, a feasibility study precedes technical development and project implementation.
- A feasibility study looks at the viability of an idea with an emphasis on identifying potential problems and attempts to answer one main question: Will the idea work and should you proceed with it?

OBJECTIVE

- The feasibility study answers the basic questions: is it realistic to address the problem or the opportunity under consideration?
- And it produces a final proposal for the management, this final report might include.

FEASIBILITY INCLUDES

- Project name
- Problem or opportunity definition
- Project description

FIVE COMMON FACTORS (TELOS)

1. Technology and system feasibility
2. Economic feasibility
3. Legal feasibility
4. Operational feasibility
5. Schedule feasibility

OPERATIONAL FEASIBILITY

Is a measure of how well a proposed system solves the problems, and takes advantages of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development?

SCHEDULE FEASIBILITY

A project will fail if it takes too long to be completed before it is useful. Typically, this means estimating how long the system will take to develop, and if it can be completed in a given time period using some methods like payback period. Schedule feasibility is a measure of how reasonable the project timetable is. Given our technical expertise, are the project deadlines reasonable? Some projects are initiated with specific deadlines. You need to determine whether the deadlines are mandatory or desirable.

2.2 EXISTING SYSTEM

In previous model they use different types of ML classification model to find the accuracy of the phishing websites it's not more effective to predictive the model.

TECHNIQUE

- Decision tree

DRAWBACK

- Less accuracy

2.3 PROPOSED SYSTEM

In this model we implement random forest algorithm and k nearest neighbor algorithm to find the best prediction

TECHNIQUE

- Random Forest

ADVANTAGES

We find the best accuracy than previous models

- Exploration data analysis of variable identification :
 - Loading the given dataset
 - Import required libraries packages
 - Analyze the general properties
 - Find duplicate and missing values
 - Checking unique and count values
- Uni-variate data analysis :
 - Rename, add data and drop the data
 - To specify data type
- Exploration data analysis of bi-variate and multi-variate :
 - Plot diagram of pairplot, heatmap, bar chart and Histogram
- Method of Outlier detection with feature engineering :
 - Pre-processing the given dataset
 - Splitting the test and training dataset
 - Comparing the Decision tree and Logistic regression model and random forest etc.
- Comparing algorithm to predict the result:
 - Based on the best accuracy

CHAPTER 3

SYSTEM

CONFIGURATION

CHAPTER-3

3. SYSTEM CONFIGURATION

3.1 HARDWARE REQUIREMENTS:

PROCESSOR	:	Intel I3
RAM	:	12 GB
HARD DISK	:	200 GB

3.2 SOFTWARE REQUIREMENTS:

OPERATING SYSTEM	:	Windows
TOOL	:	Anaconda with Jupyter Notebook

3.3 ABOUT THE SOFTWARE

SOFTWARE DESCRIPTION

Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system “Conda”. The Anaconda distribution is used by over 12 million users and includes more than 1400 popular data-science packages suitable for Windows, Linux, and MacOS. So, Anaconda distribution comes with more than 1,400 packages as well as the Conda package and virtual environment manager called Anaconda Navigator and it eliminates the need to learn to install each library independently. The open source packages can be individually installed from the

Anaconda repository with the conda install command or using the pip install command that is installed with Anaconda. Pip packages provide many of the features of conda packages and in most cases they can work together. Custom packages can be made using the `conda build` command, and can be shared with others by uploading them to Anaconda Cloud, [PyPI](#) or other repositories. The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, you can create new environments that include any version of Python packaged with conda.

ANACONDA NAVIGATOR

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows you to launch applications and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda.org or in a local Anaconda Repository. Anaconda. Now, if you are primarily doing data science work, Anaconda is also a great option. Anaconda is created by Continuum Analytics, and it is a Python distribution that comes preinstalled with lots of useful python libraries for data science.

Anaconda is a distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. In order to run, many scientific packages depend on specific versions of other packages. Data scientists often use multiple versions of many packages and use multiple environments to separate these different versions.

The command-line program conda is both a package manager and an environment manager. This helps data scientists ensure that each version of each package has all the dependencies it requires and works correctly. Navigator is an easy, point-and-click way to work with packages and environments without needing to type conda commands in a terminal window. You can use it to find the packages you want, install them in an environment, run the packages, and update them – all inside Navigator.

The following applications are available by default in Navigator:

- Jupyter Lab
- Jupyter Notebook
- Spyder
- PyCharm
- VSCode
- Glueviz
- Orange 3 App
- RStudio
- Anaconda Prompt (Windows only) and Anaconda PowerShell (Windows only)

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution. Navigator allows you to launch common Python programs and easily manage conda packages, environments, and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository. Anaconda comes with many built-in packages that you can easily find with `conda list` on your anaconda prompt. As it has lots of packages (many of which are rarely used), it requires lots of space and time as well. If you have enough space, time and do not want to burden yourself to install small utilities like JSON, YAML, you better go for Anaconda.

3.3.2 CONDA :

Conda is an open source, cross-platform, language-agnostic package manager and environment management system that installs, runs, and updates packages and their dependencies. It was created for Python programs, but it can package and distribute software for any language (e.g., R), including multi-language projects. The conda package and environment manager is included in all versions of Anaconda, Miniconda, and Anaconda Repository.

Anaconda is freely available, open source distribution of python and R programming languages which is used for scientific computations. If you are doing any machine learning or deep learning project then this is the best place for you. It consists of many softwares which will help you to build your machine learning project and deep learning project. these softwares have great graphical user interface and these will make your work easy to do. you can also use it to run your python script. These are the software carried by anaconda navigator.

3.3.3 JUPYTER NOTEBOOK

This website acts as “meta” documentation for the Jupyter ecosystem. It has a collection of resources to navigate the tools and communities in this ecosystem, and to help you get started. Project Jupyter is a project and community whose goal is to "develop open-source software, open-standards, and services for interactive computing across dozens of programming languages". It was spun off from IPython in 2014 by Fernando Perez.

Notebook documents are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc...).

Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc.) as well as executable documents which can be run to perform data analysis.

3.3.4 THE JUPYTER NOTEBOOK

The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and much more.

- Download and install anaconda and get the most useful package for machine learning in Python.
- Load a dataset and understand its structure using statistical summaries and data visualization.
- Machine learning models, pick the best and build confidence that the accuracy is reliable.

Python is a popular and powerful interpreted language. Unlike R, Python is a complete language and platform that you can use for both research and development and developing production systems. There are also a lot of modules and libraries to choose from, providing multiple ways to do each task. It can feel overwhelming.

The best way to get started using Python for machine learning is to complete a project.

- It will force you to install and start the Python interpreter (at the very least).
- It will give you a bird's eye view of how to step through a small project.
- It will give you confidence, maybe to go on to your own small projects.

When you are applying machine learning to your own datasets, you are working on a project. A machine learning project may not be linear, but it has a number of well-known steps:

- Define Problem.
- Prepare Data.
- Evaluate Algorithms.
- Improve Results.
- Present Results.

The best way to really come to terms with a new platform or tool is to work through a machine learning project end-to-end and cover the key steps. Namely, from loading data, summarizing data, evaluating algorithms and making some predictions.

Here is an overview of what we are going to cover:

1. Installing the Python anaconda platform.
2. Loading the dataset.
3. Summarizing the dataset.
4. Visualizing the dataset.
5. Evaluating some algorithms.
6. Making some predictions.

3.3.5 PYTHON

INTRODUCTION:

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.

Guido van Rossum began working on Python in the late 1980s, as a successor to the ABC programming language, and first released it in 1991 as Python 0.9.0. Python 2.0 was released in 2000 and introduced new features, such as list comprehensions and a garbage collection system using reference counting.

Python 3.0 was released in 2008 and was a major revision of the language that is not completely backward-compatible. Python 2 was discontinued with version 2.7.18 in 2020.

3.3.6 HISTORY:

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde& Informatica (CWI) in the Netherlands as a successor to ABC programming language, which was inspired by SETL, capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12 July 2018, when he announced his "permanent vacation" from his responsibilities as Python's Benevolent Dictator For Life, a title the Python community bestowed upon him to reflect his long-term commitment as the project's chief decision-maker. In January 2019, active Python core developers elected a 5-member "Steering Council" to lead the project. As of 2021, the current members of this council are Barry Warsaw, Brett Cannon, Carol Willing, Thomas Wouters, and Pablo Galindo Salgado.

Python 2.0 was released on 16 October 2000, with many major new features, including a cycle-detecting garbage collector and support for Unicode. Python 3.0 was released on 3 December 2008. It was a major revision of the language that is not completely backward-compatible.

Many of its major features were backported to Python 2.6.x and 2.7.x version series. Releases of Python 3 include the 2 to 3 utility, which automates (at least partially) the translation of Python 2 code to Python 3. Python 2.7's end-of-life date was initially set at 2015 then postponed to 2020 out of concern that a large body of existing code could not easily be forward-ported to Python 3. No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.6.x and later are supported. Python 3.9.2 and 3.8.8 were expedited as all versions of Python (including 2.7) had security issues, leading to possible remote code execution and web cache poisoning.

3.3.7 DESIGN PHILOSOPHY & FEATURE

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by meta-programming and meta-objects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution.

Python's design offers some support for functional programming in the Lisp tradition. It has filter, map and reduce functions; list comprehensions, dictionaries, sets, and generator expressions. The standard library has two modules (itertools and functools) that implement functional tools borrowed from Haskell and Standard ML.

The language's core philosophy is summarized in the document The Zen of Python (PEP 20), which includes aphorisms such as:

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Readability counts.

Rather than having all of its functionality built into its core, Python was designed to be highly extensible (with modules). This compact modularity has made it particularly popular as a means of adding programmable interfaces to existing applications.

Van Rossum's vision of a small core language with a large standard library and easily extensible interpreter stemmed from his frustrations with ABC, which espoused the opposite approach. Python strives for a simpler, less-cluttered syntax and grammar while giving developers a choice in their coding methodology. In contrast to Perl's "there is more than one way to do it" motto, Python embraces a "there should be one— and preferably only one — obvious way to do it" design philosophy. Alex Martelli, a Fellow at the Python Software Foundation and Python book author, writes that "To describe something as 'clever' is not considered a compliment in the Python culture."

Python's developers strive to avoid premature optimization, and reject patches to non-critical parts of the C-Python reference implementation that would offer marginal increases in speed at the cost of clarity.

When speed is important, a Python programmer can move time-critical functions to extension modules written in languages such as C, or use PyPy, a just-in-time compiler. Python is also available, which translates a Python script into C and makes direct C-level API calls into the Python interpreter.

Python's developers aim to keep the language fun to use. This is reflected in its name a tribute to the British comedy group Monty Python and in occasionally playful approaches to tutorials and reference materials, such as examples that refer to spam and eggs (a reference to a Monty Python sketch) instead of the standard foo and bar.

A common neologism in the Python community is *pythonic*, which can have a wide range of meanings related to program style. To say that code is *pythonic* is to say that it uses Python idioms well, that it is natural or shows fluency in the language, that it conforms with Python's minimalist philosophy and emphasis on readability. In contrast, code that is difficult to understand or reads like a rough transcription from another programming language is called *unpythonic*. Users and admirers of Python, especially those considered knowledgeable or experienced, are often referred to as *Pythonistas*.

3.3.8 SYNTAX AND SEMANTICS :

Python is meant to be an easily readable language. Its formatting is visually uncluttered, and it often uses English keywords where other languages use punctuation.

Unlike many other languages, it does not use curly brackets to delimit blocks, and semicolons after statements are allowed but are rarely, if ever, used. It has fewer syntactic exceptions and special cases than C or Pascal.

INDENTATION :

Main article: Python syntax and semantics & Indentation

Python uses whitespace indentation, rather than curly brackets or keywords, to delimit blocks. An increase in indentation comes after certain statements; a decrease in indentation signifies the end of the current block. Thus, the program's visual structure accurately represents the program's semantic structure. This feature is sometimes termed the off-side rule, which some other languages share, but in most languages indentation does not have any semantic meaning. The recommended indent size is four spaces.

3.3.9 STATEMENTS AND CONTROL FLOW :

PYTHON'S STATEMENTS INCLUDE:

- The assignment statement, using a single equals sign =.
- The if statement, which conditionally executes a block of code, along with else and elif (a contraction of else-if).
- The for statement, which iterates over an iterable object, capturing each element to a local variable for use by the attached block.
- The while statement, which executes a block of code as long as its condition is true.
- The Try statement, which allows exceptions raised in its attached code block to be caught and handled by except clauses; it also ensures that clean-up code in a finally block will always be run regardless of how the block exits.
- The raise statement, used to raise a specified exception or re-raise a caught exception.
- The class statement, which executes a block of code and attaches its local namespace to a class, for use in object-oriented programming.
- The def statement, which defines a function or method.
- The with statement, which encloses a code block within a context manager (for example, acquiring a lock before the block of code is run and releasing the lock afterwards, or opening a file and then closing it), allowing resource-acquisition-is-initialization (RAII) – like behaviour and replaces a common try/finally idiom.
- The break statement, exits from a loop.
- The continue statement, skips this iteration and continues with the next item.
- The del statement, removes a variable, which means the reference from the name to the value is deleted and trying to use that variable will cause an error. A deleted variable can be reassigned.
- The pass statement, which serves as a NOP. It is syntactically needed to create an empty code block.
- The assert statement, used during debugging to check for conditions that should apply.
- The yield statement, which returns a value from a generator function and yield is also an operator. This form is used to implement co-routines.
- The return statement, used to return a value from a function.

CHAPTER-4

DESIGN

METHODOLOGY

CHAPTER 4

4. DESIGN METHODOLOGY

4.1 SYSTEM ARCHITECTURE

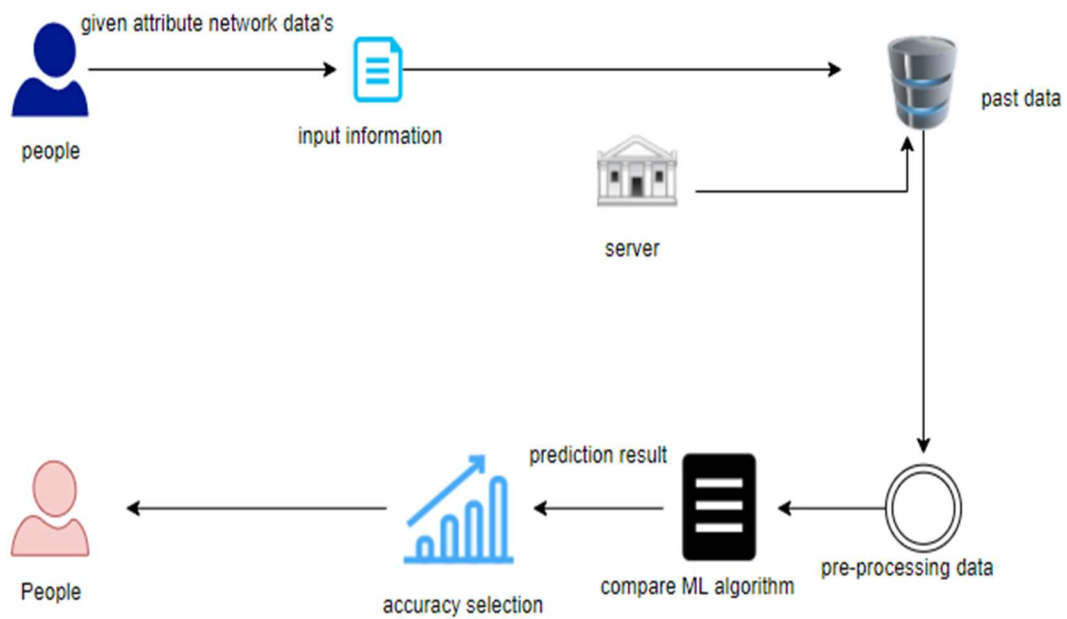


Fig:4.1.1 SYSTEM DESIGN

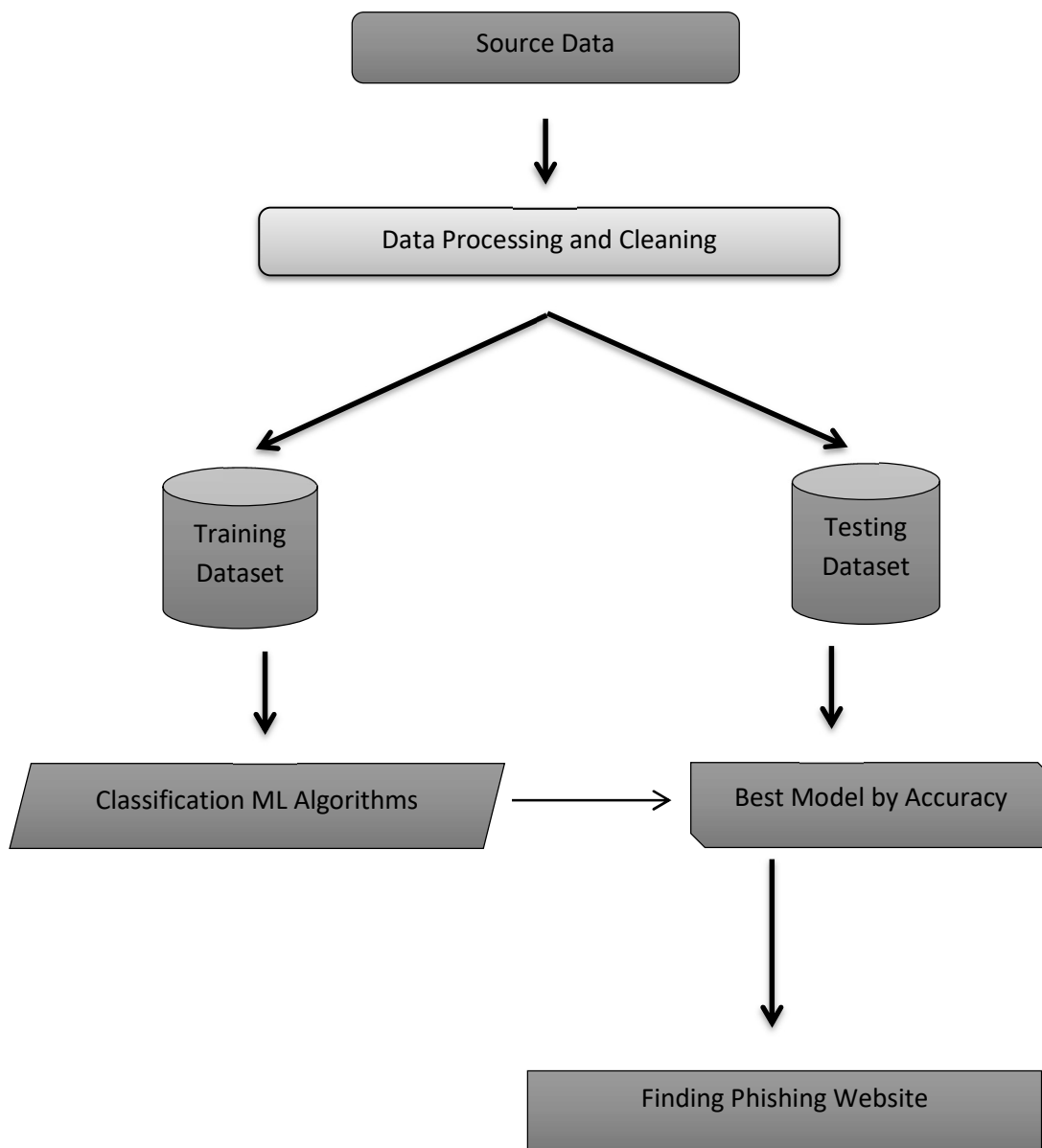


Fig 4.2 WORK FLOW DIAGRAM

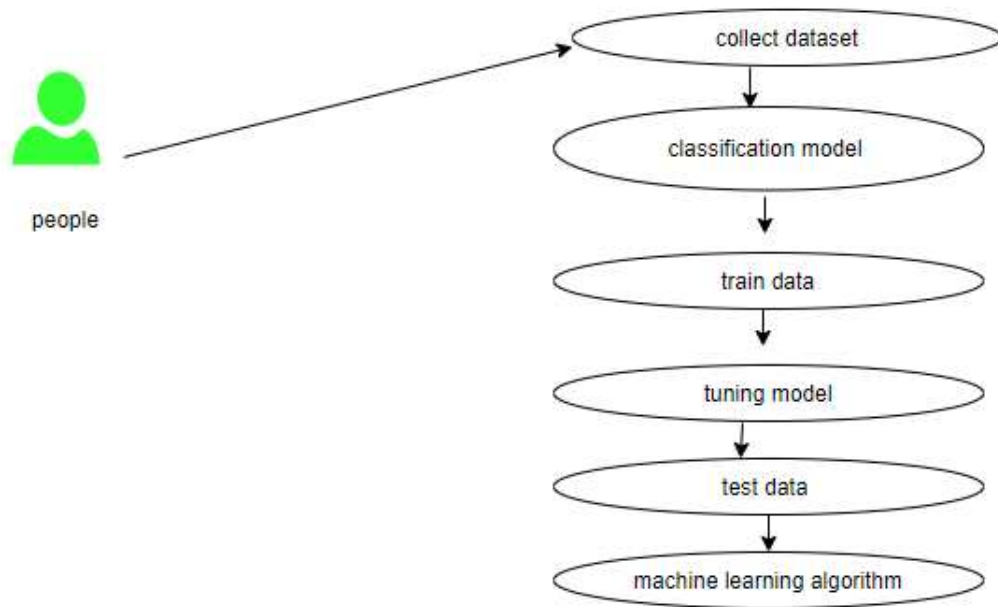


Fig 4.3 USE CASE DIAGRAM

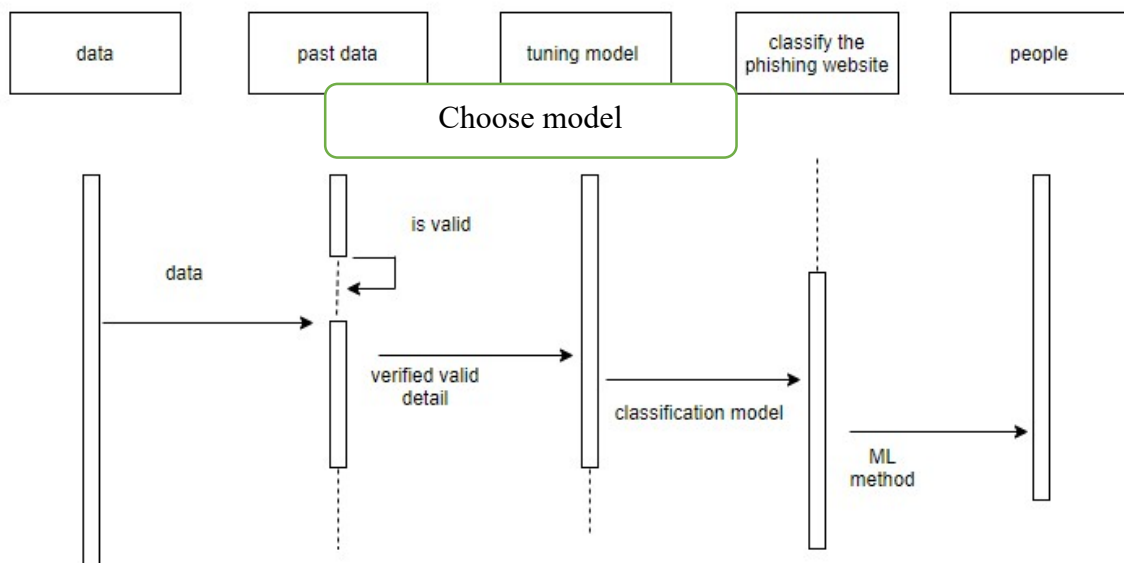


Fig 4.3.1 SEQUENCE DIAGRAM

CHAPTER 5

MODULE DESCRIPTION

CHAPTER – 5

MODULE DESCRIPTION

5.1 MODULE

- 1.Pre-Processing
- 2.Visulation
- 3.Application Of Algorithm On Dataset

5.1.1 DATA PREPROCESSING IN MACHINE LEARNING

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

5.1.2 WHY DO WE NEED DATA PREPROCESSING?

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model. Some specified Machine Learning model needs information in a specified format, for example, Random Forest algorithm does not support null values. Therefore, to execute random forest algorithm null values have to be managed from the original raw data set. And another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in given dataset.

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

GET THE DATASET :

To create a machine learning model, the first thing we required is a dataset as a machine learning model completely works on data. The collected data for a particular problem in a proper format is known as the dataset.

Dataset may be of different formats for different purposes, such as, if we want to create a machine learning model for business purpose, then dataset will be different with the dataset required for a liver patient. So each dataset is different from another dataset. To use the dataset in our code, we usually put it into a CSV file. However, sometimes, we may also need to use an HTML or xlsx file

WHAT IS A CSV FILE?

CSV stands for "Comma-Separated Values" files; it is a file format which allows us to save the tabular data, such as spreadsheets. It is useful for huge datasets and can use these datasets in programs. Here we will use a demo dataset for data preprocessing, and for practice, For real-world problems, we can download datasets online from various sources such as <https://www.kaggle.com/uciml/datasets> We can also create our dataset by gathering data using various API with Python and put that data into a .csv file

IMPORTING LIBRARIES :

In order to perform data preprocessing using Python, we need to import some predefined Python libraries. These libraries are used to perform some specific jobs. There are three specific libraries that we will use for data preprocessing, which are:

NUMPY:

NumPy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices. So, in Python, we can import it as:

MATPLOTLIB:

The second library is Matplotlib, which is a Python 2D plotting library, and with this library, we need to import a sub-library pyplot. This library is used to plot any type of charts in Python for the code.

PANDAS:

The last library is the Pandas library, which is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library

IMPORTING THE DATASETS

Now we need to import the datasets which we have collected for our machine learning project. But before importing a dataset, we need to set the current directory as a working directory. To set a working directory in Spyder IDE

1. Save your Python file in the directory which contains dataset.
2. Go to File explorer option in Spyder IDE, and select the required directory.
3. Click on F5 button or run option to execute the file.

READ_CSV() FUNCTION:

Now to import the dataset, we will use `read_csv()` function of pandas library, which is used to read a csv file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.

HANDLING MISSING DATA:

The next step of data pre-processing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

WAYS TO HANDLE MISSING DATA:

There are mainly two ways to handle missing data, which are:

By Deleting The Particular Row: The first way is used to commonly deal with null values. In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

By Calculating The Mean: In this way, we will calculate the mean of that column or row which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc. Here, we will use this approach. To handle missing values, we will use Scikit-learn library in our code, which contains various libraries for building machine learning models. Here we will use Imputer class of sklearn. Pre-processing library. Below is the code for it:

SPLITTING THE DATASET INTO THE TRAINING SET AND TEST SET

In machine learning data preprocessing, we divide our dataset into a training set and test set. This is one of the crucial steps of data preprocessing as by doing this, we can enhance the performance of our machine learning model.

Suppose, if we have given training to our machine learning model by a dataset and we test it by a completely different dataset. Then, it will create difficulties for our model to understand the correlations between the models.

If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:

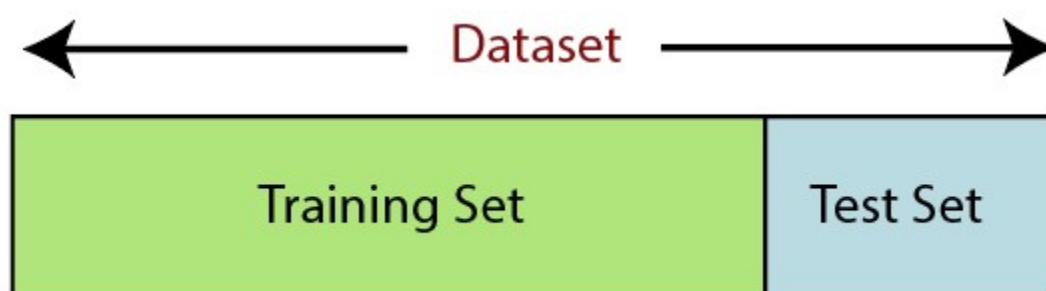


Fig 5.1.3 Data splitting

TRAINING THE DATASET

- The first line imports the given data set which is already predefined in sklearn module and raw data set is basically a table which contains information about various varieties.
- For example, to import any algorithm and `train_test_split` class from sklearn and numpy module for use in this program.

- To encapsulate `load_data ()` method in `data_dataset` variable. Further divide the dataset into training data and test data using `train_test_split` method. The `X` prefix in variable denotes the feature values and `y` prefix denotes target values.
- This method divides dataset into training and test data randomly in ratio of 67:33 / 70:30. Then we encapsulate any algorithm.
- In the next line, we fit our training data into this algorithm so that computer can get trained using this data. Now the training part is complete.
- A subset of dataset to train the machine learning model, and we already know the output.

TESTING THE DATASET

- Now, the dimensions of new features in a NumPy array called '`n`' and we want to predict the species of this features and to do using the `predict` method which takes this array as input and spits out predicted target value as output.
- So, the predicted target value comes out to be 0. Finally to find the test score which is the ratio of no. of predictions found correct and total predictions made and finding accuracy score method which basically compares the actual values of the test set with the predicted values. A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

FEATURE SCALING

Feature scaling is the final step of data preprocessing in machine learning. It is a technique to standardize the independent variables of the dataset in a specific range. In feature scaling, we put our variables in the same range and in the same scale so that no any variable dominate the other variable.

5.2 EXPLORATION DATA ANALYSIS OF VISUALIZATION

Data visualization is an important skill in applied statistics and machine learning. Statistics does indeed focus on quantitative descriptions and estimations of data. Data visualization provides an important suite of tools for gaining a qualitative understanding. This can be helpful when exploring and getting to know a dataset and can help with identifying patterns, corrupt data, outliers, and much more. With a little domain knowledge, data visualizations can be used

to express and demonstrate key relationships in plots and charts that are more visceral and stakeholders than measures of association or significance. Data visualization and exploratory data analysis are whole fields themselves and it will recommend a deeper dive into some the books mentioned at the end. Sometimes data does not make sense until it can look at in a visual form, such as with charts and plots. Being able to quickly visualize of data samples and others is an important skill both in applied statistics and in applied machine learning. It will discover the many types of plots that you will need to know when visualizing data in Python and how to use them to better understand your own data.

- How to chart time series data with line plots and categorical quantities with bar charts.
- How to summarize data distributions with histograms and box plots.

5.2.1 USED PYTHON PACKAGES:

sklearn:

- In python, sklearn is a machine learning package which include a lot of ML algorithms.
- Here, we are using some of its modules like `train_test_split`, Decision Tree Classifier or Logistic Regression and accuracy score.

NumPy:

- It is a numeric python module which provides fast maths functions for calculations.
- It is used to read data in numpy arrays and for manipulation purpose.

Pandas:

- Used to read and write different files.
- Data manipulation can be done easily with data frames.

Matplotlib:

- Data visualization is a useful way to help with identify the patterns from given dataset.
- Data manipulation can be done easily with data frames.

5.1.3 APPLICATION OF ALGORITHM ON DATASET

Therefore, to execute random forest algorithm null values have to be managed from the original raw data set. And another aspect is that data set should be formatted in such a way that more than one Machine Learning and Deep Learning algorithms are executed in given dataset.

False Positives (FP):

A person who will pay predicted as defaulter. When actual class is no and predicted class is yes. E.g. if actual class says this passenger did not survive but predicted class tells you that this passenger will survive.

False Negatives (FN): A person who default predicted as payer. When actual class is yes but predicted class is no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.

True Positives (TP): A person who will not pay predicted as defaulter. These are the correctly predicted positive values which means that the value of actual class is yes and the value of predicted class is also yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you the same thing.

True Negatives (TN): A person who default predicted as payer. These are the correctly predicted negative values which means that the value of actual class is no and value of predicted class is also no. E.g. if actual class says this passenger did not survive and predicted class tells you the same thing.

Precision: The proportion of positive predictions that are actually correct.

$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$ Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. The question that this metric answer is of all passengers that labeled as survived, how many actually survived? High precision relates to the low false positive rate. We have got 0.788 precision which is pretty good.

Recall: The proportion of positive observed values correctly predicted. (The proportion of actual defaulters that the model will correctly predict)

$\text{Recall} = \text{TP} / (\text{TP} + \text{FN})$ Recall (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes.

F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost. If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall.

General Formula:

$$F\text{- Measure} = 2TP / (2TP + FP + FN)$$

F1-Score Formula:

$$F1 \text{ Score} = 2 * (\text{Recall} * \text{Precision}) / (\text{Recall} + \text{Precision})$$

CHAPTER 6

ALGORITHM AND

TESTING

CHAPTER 6

6. ALGORITHM AND TESTING

6.1 ALGORITHM EXPLANATION:

In machine learning and statistics, classification is a supervised learning approach in which the computer program learns from the data input given to it and then uses this learning to classify new observation. This data set may simply be bi-class (like identifying whether the person is male or female or that the mail is spam or non-spam) or it may be multi-class too. Some examples of classification problems are: speech recognition, handwriting recognition, biometric identification, document classification etc. In Supervised Learning, algorithms learn from labelled data. After understanding the data, the algorithm determines which label should be given to new data based on pattern and associating the patterns to the unlabelled new data.

6.1.1 RANDOM FOREST CLASSIFIER:

Random forests or random decision forests are an ensemble learning method for classification, regression and other tasks, that operate by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random decision forests correct for decision trees' habit of over fitting to their training set. Random forest is a type of supervised machine learning algorithm based on ensemble learning. Ensemble learning is a type of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest algorithm combines multiple algorithm of the same type i.e. multiple decision trees, resulting in a forest of trees, hence the name

"Random Forest". The random forest algorithm can be used for both regression and classification tasks.

The following are the basic steps involved in performing the random forest algorithm:

- Pick N random records from the dataset.
- Build a decision tree based on these N records.
- Choose the number of trees you want in your algorithm and repeat steps 1 and 2.

- In case of a regression problem, for a new record, each tree in the forest predicts a value for Y (output). The final value can be calculated by taking the average of all the values predicted by all the trees in forest. Or, in case of a classification problem, each tree in
- the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the majority vote.
- And we implementing confusion matrix to our algorithm .A Confusion matrix is an $N \times N$ matrix used for evaluating the performance of a classification model, where N is the number of target classes. The matrix compares the actual target values with those predicted by the machine learning model. This gives us a holistic view of how well our classification model is performing and what kinds of errors it is making.

6.1.2 K-NEAREST NEIGHBORS:

The k-nearest neighbour (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems. The KNN algorithm assumes that similar things exist in close proximity. In other words, similar things are near to each other. The KNN algorithm hinges on this assumption being true enough for the algorithm to be useful. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph.

1. Here we do not learn weights from training data to predict output (as in model-based algorithms) but use entire training instances to predict output for unseen data.
2. Lazy Learning: Model is not learned using training data prior and the learning process is postponed to a time when prediction is requested on the new instance.
3. Non -Parametric: In KNN, there is no predefined form of the mapping function.

6.1.2 K-NEAREST NEIGHBORS:

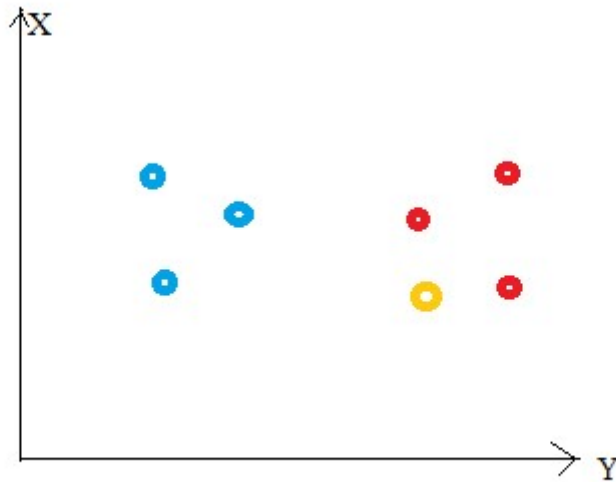


Fig 6.1.2 K-NEAREST NEIGHBORS:

Yes, this is the principle behind K Nearest Neighbors. Here, nearest neighbors are those data points that have minimum distance in feature space from our new data point. And K is the number of such data points we consider in our implementation of the algorithm. Therefore, distance metric and K value are two important considerations while using the KNN algorithm. Euclidean distance is the most popular distance metric. You can also use Hamming distance, Manhattan distance, Minkowski distance as per your need. For predicting class/ continuous value for a new data point, it considers all the data points in the training dataset. Finds new data point's 'K' Nearest Neighbors (Data points) from feature space and their class labels or continuous values. For classification: A class label assigned to the majority of K Nearest Neighbors from the training dataset is considered as a predicted class for the new data point. For regression: Mean or median of continuous values assigned to K Nearest Neighbors from training dataset is a predicted continuous value for our new data point

6.2 TESTING :

Software testing is the process of evaluation a software item to detect differences between given input and expected output. Also to assess the feature of A software item. Testing assesses the quality of the product.

Software testing is a process that should be done during the development process. In other words software testing is a verification and validation process.

6.2.1 VERIFICATION:

Verification is the process to make sure the product satisfies the conditions imposed at the start of the development phase. In other words, to make sure the product behaves the way we want it to.

6.2.2 TYPES OF TESTING:

There are many types of testing like

- Unit Testing
- System Testing
- Performance Testing
- Integration Testing
- Functional Testing

UNIT TESTING

Unit testing is the testing of an individual unit or group of related units. It falls under the class of white box testing. It is often done by the programmer to test that the unit he/she has implemented is producing expected output against given input.

INTEGRATION TESTING

Integration testing is testing in which a group of components are combined to produce output. Also, the interaction between software and hardware is tested in integration testing if software and hardware components have any relation. It may fall under both white box testing and black box testing.

FUNCTIONAL TESTING

Functional testing is the testing to ensure that the specified functionality required in the system requirements works. It falls under the class of black box testing.

SYSTEM TESTING

System testing is the testing to ensure that by putting the software in different environments (e.g., Operating Systems) it still works. System testing is done with full system implementation and environment. It falls under the class of black box testing.

PERFORMANCE TESTING

Performance testing is the testing to assess the speed and effectiveness of the system and to make sure it is generating results within a specified time as in performance requirements. It falls under the class of black box testing.

CHAPTER 7

CODING AND

SCREENSHOTS

CHAPTER 7

7. CODING AND SCREENSHOTS

7.1 SCREEN SHOTS :

The screenshot shows a Jupyter Notebook interface in a web browser. The notebook is titled 'Module 1 - Jupyter Notebook' and is running on 'localhost:8888/notebooks/Desktop/phising/Module%201.ipynb'. The notebook content is titled 'Data Validation Process' and contains the following code and output:

```
In [2]: import pandas as p
```

```
In [3]: data= p.read_csv("phishing_data.csv")
```

```
In [4]: data.head()
```

Out[4]:

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Dom
0	graphonver.net	0	0	1	1	0	0	0	0	0	1	1	
1	eonavi.jp	0	0	1	1	1	0	0	0	0	1	1	
2	hubpages.com	0	0	1	1	0	0	0	0	0	1	0	
3	extratorrent.cc	0	0	1	3	0	0	0	0	0	1	0	
4	icibank.com	0	0	1	3	0	0	0	0	0	1	0	

```
In [5]: data.tail()
```

Out[5]:

	Domain	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain_Age	Dom
9995	wvk12-my.sharepoint.com	0	0	1	5	0	0	1	1	0	1	1	
9996	adpfile.com	0	0	1	4	0	0	0	0	0	1	0	
9997	kuroitnoye.com.ua	0	1	1	3	0	0	1	0	0	0	1	
9998	norcalto-my.sharepoint.com	0	0	1	5	0	0	1	1	0	1	1	
9999	sieok-kuehlssysteme.de	0	1	1	4	0	0	1	1	0	1	1	

Fig 7.1.1 data validation 1

Jupyter Module 1 Last Checkpoint: 02/19/2021 (unsaved changes)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run

```

12      4
13      2
20      1
18      1
17      1
Name: URL_Depth, dtype: int64

In [14]: df['Web_Traffic'].value_counts()
Out[14]: 1    8457
         0    1543
         Name: Web_Traffic, dtype: int64

In [15]: df['Domain_Age'].value_counts()
Out[15]: 0    5863
         1    4137
         Name: Domain_Age, dtype: int64

In [16]: df.describe()
Out[16]:
```

	Have_IP	Have_At	URL_Length	URL_Depth	Redirection	https_Domain	TinyURL	Prefix/Suffix	DNS_Record	Web_Traffic	Domain
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	0.005500	0.022600	0.773400	3.072000	0.013500	0.000200	0.090300	0.093200	0.100800	0.845700	0.41
std	0.073961	0.148632	0.418653	2.128631	0.115408	0.014141	0.286625	0.290727	0.301079	0.361254	0.49

Fig 7.1.2 data validation 2

7.1.2 VISUALIZATION

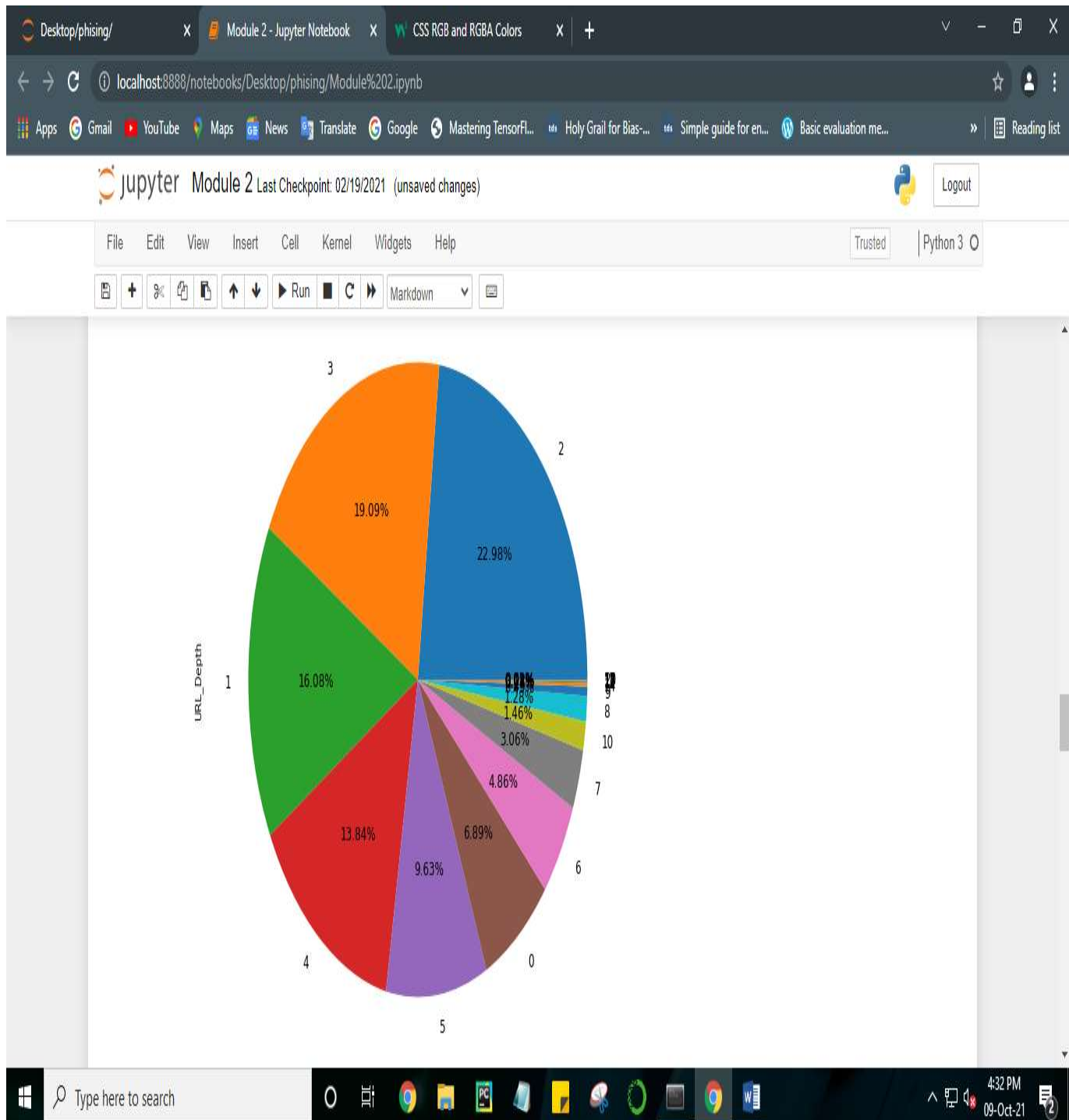


Fig 7.1.3 VISUALIZATION 1

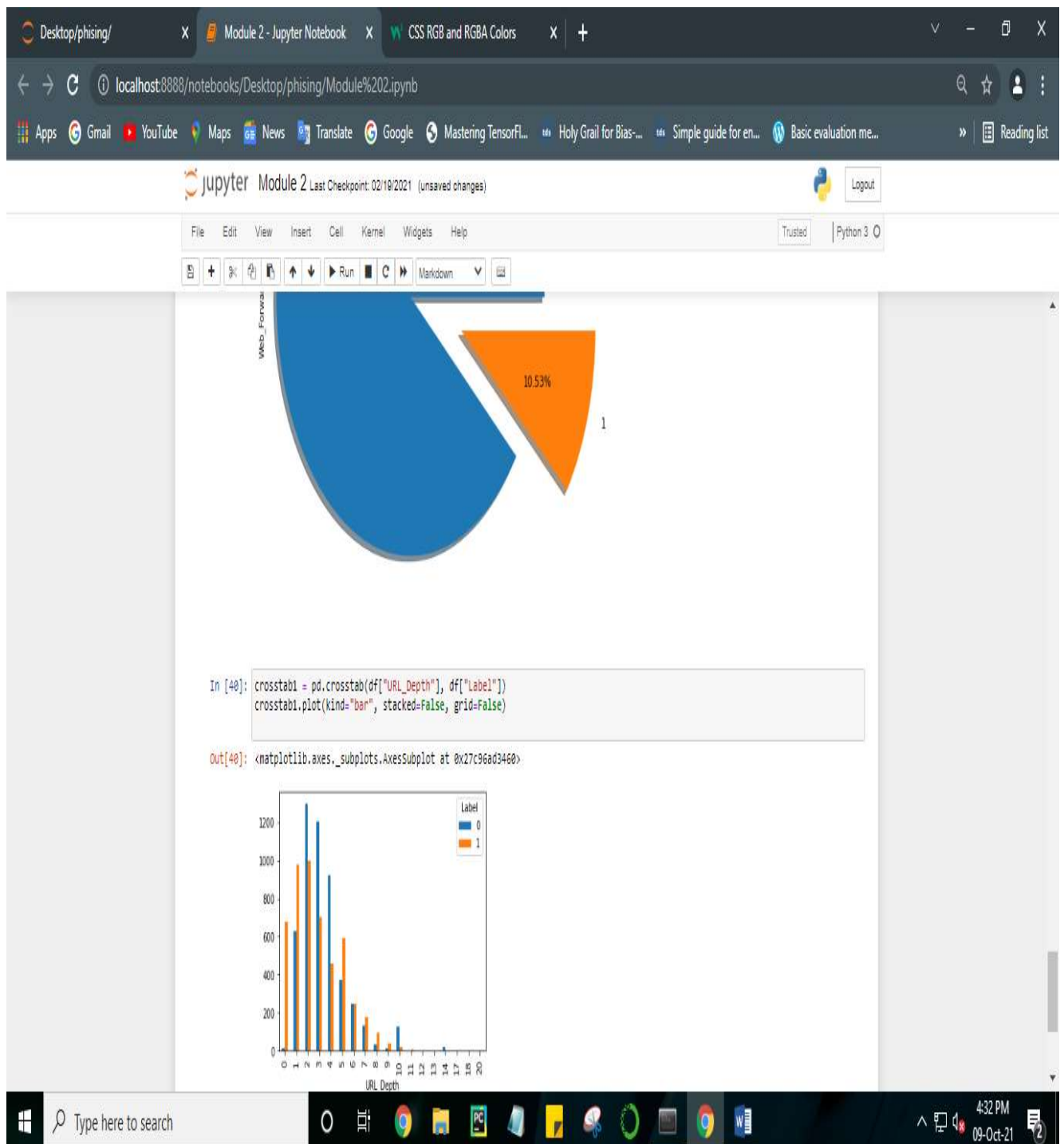
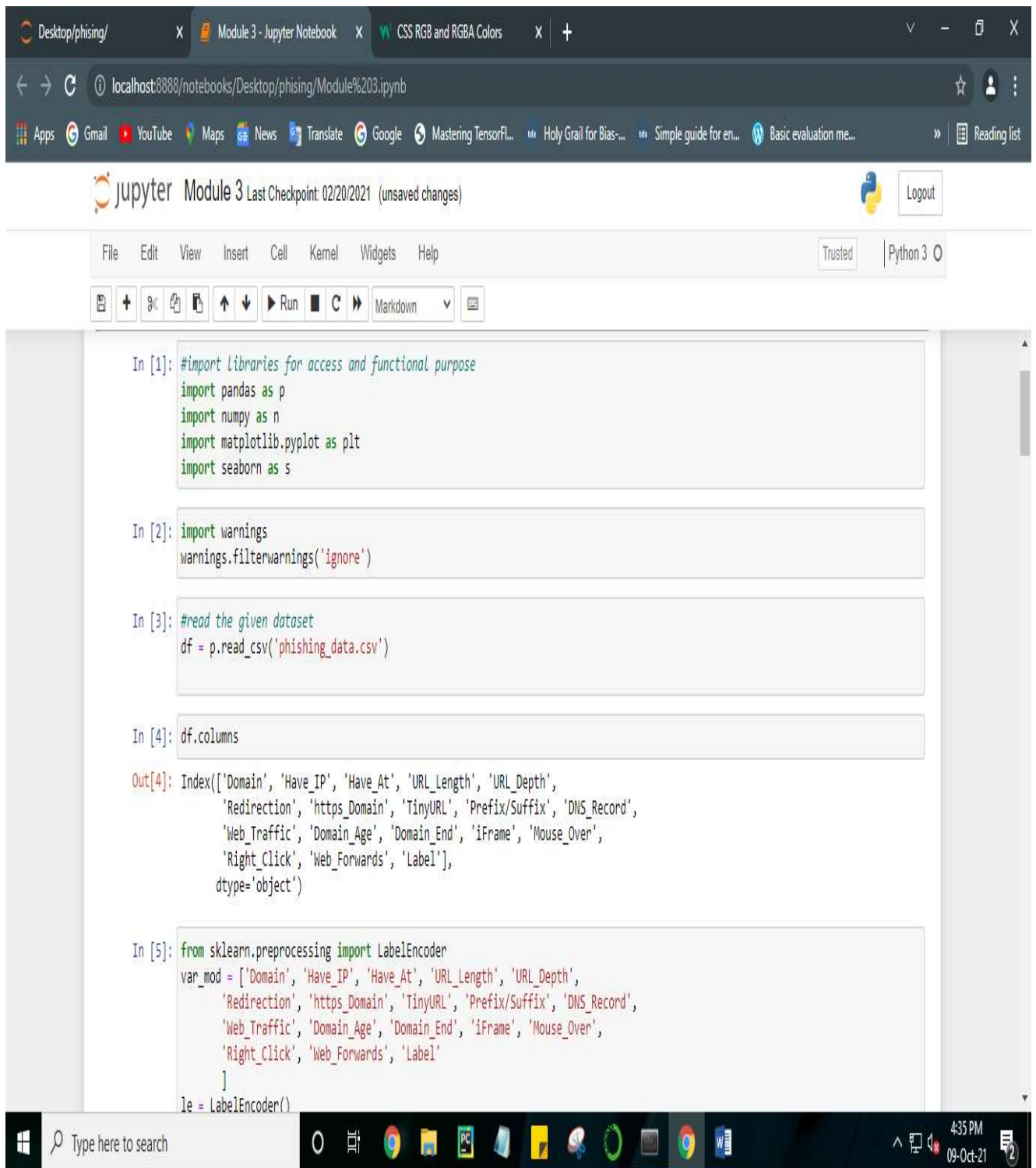


Fig 7.1.3 VISUALIZATION 2

7.1.4 Applying algorithm



The screenshot displays a Jupyter Notebook titled "Module 3" running on a local server at localhost:8888. The notebook contains five input cells with the following code:

```
In [1]: #import libraries for access and functional purpose
import pandas as p
import numpy as n
import matplotlib.pyplot as plt
import seaborn as s

In [2]: import warnings
warnings.filterwarnings('ignore')

In [3]: #read the given dataset
df = p.read_csv('phishing_data.csv')

In [4]: df.columns

Out[4]: Index(['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth',
              'Redirection', 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record',
              'Web_Traffic', 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over',
              'Right_Click', 'Web_Forwards', 'Label'],
              dtype='object')

In [5]: from sklearn.preprocessing import LabelEncoder
var_mod = ['Domain', 'Have_IP', 'Have_At', 'URL_Length', 'URL_Depth',
           'Redirection', 'https_Domain', 'TinyURL', 'Prefix/Suffix', 'DNS_Record',
           'Web_Traffic', 'Domain_Age', 'Domain_End', 'iFrame', 'Mouse_Over',
           'Right_Click', 'Web_Forwards', 'Label']
le = LabelEncoder()
```

The interface includes a top navigation bar with tabs for "Desktop/phishing/", "Module 3 - Jupyter Notebook", and "CSS RGB and RGBA Colors". The Jupyter Notebook header shows "Module 3" and "Last Checkpoint: 02/20/2021 (unsaved changes)". The bottom status bar indicates the time as 4:35 PM on 09-Oct-21.

FIG 7.1.4 APPLYING ALGORITHM

CHAPTER 8

CONCLUSION

CHAPTER 8

8. CONCLUSION

The analytical process started from data cleaning and processing, missing value, exploratory analysis and finally model building and evaluation. The best accuracy on public test set is higher accuracy score is will be find out. Getting the dataset form skykit learn and clean the raw data and split them and using NumPy, pandas we split the data and view the data sent and apply algorithm to find the best result .

This application can help to find the Prediction of Phishing Website and we implement random forest algorithm for better prediction add on that we use KNN algorithm and compare them both to find the best accuracy. This paper aims to enhance detection method to detect phishing websites using machine learning technology. We achieved 97.14% detection accuracy using random forest algorithm with lowest false positive rate. Also result shows that classifiers give better performance when we used more data as training data. In future hybrid technology will be implemented to detect phishing websites more accurately, for which random forest algorithm of machine learning technology and blacklist method will be used.

- Phishing prediction to connect with AI model.
- To automate this process by show the prediction result in web application or desktop application.
- To optimize the work to implement in Artificial Intelligence environment

CHAPTER 9

BIBLIOGRAPHY

CHAPTER 9

9. BIBLIOGRAPHY

- [1] Gunter Ollmann, “The Phishing Guide Understanding & Systems, 2007.
- [2] Feroz, M. N., & Mengel, S. (2015, June). Spoofed URL detection using URL ranking. In 2015 IEEE International Congress on Big Data (pp. 635-638). IEEE.
- [3] Mahmoud Khonji, Youssef Iraqi, "Phishing Detection: A Literature Survey IEEE, and Andrew Jones, 2013
- [4] Mohammad R., Thabtah F. McCluskey L., (2015) Phishing websites dataset. Available: <https://archive.ics.uci.edu/ml/datasets/Phishing+Websites> Accessed January 2016
- [5] <http://dataaspirant.com/2017/01/30/how-decision-tree-algorithm-works/>
- [6] <http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learning/>
- [7] <https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html>
- [8] W. Jing, “Covert redirect vulnerability,” 2017.
- [9] K. Krombholz, H. Hobel, M. Huber, and E. Weippl, “Advanced social engineering attacks,” Journal of Information Security and applications, vol. 22, pp. 113–122, 2015.
- [10] P. Kumaraguru, J. Cranshaw, A. Acquisti, L. Cranor, J. Hong, M. A. Blair, and T. Pham, “School of phish: a real-world evaluation of antiphishing training,” in Proceedings of the 5th Symposium on Usable Privacy and Security, pp. 1–12, 2009.
- [11] R. C. Dodge Jr, C. Carver, and A. J. Ferguson, “Phishing for user security awareness,” computers & security, vol. 26, no. 1, pp. 73–80, 2007.