

A
PROJECT REPORT ON
**PHISHING WEBSITE DETECTION USING
MACHINE LEARNING**

Submitted in partial fulfilment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

By

KOPPULA MAHESH

(21BF1F0046)

Under the esteemed guidance of

DR E. SREEDEVI Professor

HOD OF MCA



DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS
SRI VENKATESWARA COLLEGE OF ENGINEERING
(AUTONOMOUS)

**(Approved by AICTE, New Delhi & Affiliated to JNTUA,
Anantapuramu)**

Accredited by NAAC with 'A' Grade

Opp.LIC Training Center, Karakambadi Road, TIRUPATHI-517507

2021-2023

SRI VENKATESWARA COLLEGE OF ENGINEERING
(AUTONOMOUS)

(Approved by AICTE, New Delhi & Affiliated to JNTUA, Anantapuramu)

Accredited by NAAC with 'A' Grade

Opp.LIC Training Center, Karakambadi Road, TIRUPATHI-517507

DEPARTMENT OF MASTER OF COMPUTER APPLICATIONS

2021-2023



CERTIFICATE

*This is to certify that the project report entitled " **PHISHING WEBSITE DETECTION USING MACHINE LEARNING** " is a bonafide record of the project work done and submitted by*

KOPPULA MAHESH

(21BF1F0046)

*For the partial fulfilment of the requirements for award of **MASTER OF COMPUTER APPLICATIONS** degree from **SRI VENKATESWARA COLLEGE OF ENGINEERING(AUTONOMOUS)** Affiliated to JNT University Anantapur, Anantapuramu.*

GUIDE

HEAD OF DEPARTMENT

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

I am thankful to my guide **Dr . E . Sreedevi** professor for her valuable guidance and encouragement. Her helping attitude and suggestions have helped me in the successful completion of the seminar.

I would like to express my sincere thank to **Dr . E . Sreedevi** Professor , Head of the Department of MCA, for her kind help and encouragement during the course of my study and in the successful completion of the project work.

I express my sincere thanks to **Dr . N. Sudhakar Reddy** Principal S.V College of Engineering. Successful completion of any project cannot be done without proper support and encouragement . My sincere thanks to the Management for providing all the necessary during the Course of my study.

I would like to thank my parents and friends, who have greatest contributions in all my achievements, for the great care and blessings in making me successful in all endeavor,

i would like to express my deep gratitude to all those who helped me directly or transform an idea into my working project.

K.MAHESH
(21BF1F0046)

DECLARATION

I hereby declare that the project entitled “**PHISHING WEBSITE DETECTION USING MACHINE LEARNING**” submitted to the Department of MASTER OF COMPUTER APPLICATIONS. **SRI VENKATESWARA COLLEGE OF ENGINEERING (AUTONOMOUS), TIRUPATI** in partial fulfilment of requirements for the award of the degree of **MASTER OF COMPUTER APPLICATIONS**

This project is the result of my own effort and it has not been submitted to any other University or Institution for the awards of any degree other then specified above.

Signature of the Student

K.MAHESH

(21BF1F0046)

ABSTRACT

Phishing is a common attack on credulous people by making them to disclose their unique information using counterfeit websites. The objective of phishing website URLs is to purloin the personal information like user name, passwords and online banking transactions. Phishers use the websites which are visually and semantically similar to those real websites. As technology continues to grow, phishing techniques started to progress rapidly and this needs to be prevented by using anti phishing mechanisms to detect phishing. Machine learning is a powerful tool used to strive against phishing attacks. This project the features used for detection and detection techniques using machine learning.

Keywords : Decision Tree, Random forest, XGBoost , performance Anlysis.

S.NO	CONTENT	PAGES
1.	Introduction	1
2.	Literature Survey	2-4
3.	System Analysis	5-7
	3.1 Existing System	
	3.2 Proposed System	
4.	Methodology and Algorithms	8-13
	4.1 Decision Tree	
	4.2 XG Boost	
	4.3 Random Forest Classifier	
5.	Software Development Life Cycle	14-15
6.	Feasibility Study	16-17
7.	System Requirements and specifications	17-18
8.	System Design	19-29
	8.1 Input Design	
	8.2 Output Design	
	8.3 Modules	
	8.4 UML Diagrams	
	8.5 ER Diagrams	
	8.6 DF Diagrams	
9	Testing	30-33
	9.1 System Testing	
	9.2 Types of Testing	
10.	Output and Screenshots with Description	34-36

11.	Conclusion	37
12.	Future Scope	37
13.	References and Bibliography	38-41

LIST OF FIGURES

S No	Name Of Figure	Pg No
1	BLOCK DIAGRAM	6
2	ARCHITECTURE	7
3	DECISION TREE	8-9
4	RANDOM FOREST CLASSIFIER	11
5	WATERFALL MODEL	14
6	USE CASE DIAGRAM	23
7	CLASSSS DIAGRAM	23
8	SEQUENCE DIAGRAM	24
9	COLLABORATION DIAGRAM	25
10	ACTIVITY DIAGRAM	26
11	DFD DIAGRAM	29

1. INTRODUCTION

Phishing is the most unsafe criminal exercises in cyber space. Since most of the users go online to access the services provided by government and financial institutions, there has been a significant increase in phishing attacks for the past few years. Phishers started to earn money and they are doing this as a successful business. Various methods are used by phishers to attack the vulnerable users such as messaging, VOIP, spoofed link and counterfeit websites. It is very easy to create counterfeit websites, which looks like a genuine website in terms of layout and content. Even, the content of these websites would be identical to their legitimate websites. The reason for creating these websites is to get private data from users like account numbers, login id, passwords of debit and credit card, etc. Moreover, attackers ask security questions to answer to posing as a high level security measure providing to users. When users respond to those questions, they get easily trapped into phishing attacks. Many researches have been going on to prevent phishing attacks by different communities around the world. Phishing attacks can be prevented by detecting the websites and creating awareness to users to identify the phishing websites. Machine learning algorithms have been one of the powerful techniques in detecting phishing websites. In this , various methods of detecting phishing websites have been discussed.

2. LITERATURE SURVEY

1. J. Shad and S. Sharma, “A Novel Machine Learning Approach to Detect Phishing Websites Jaypee Institute of Information Technology”.

In the last few years, many fake websites have developed on the World Wide Web to harm users by stealing their confidential information such as account ID, user name, password, etc. Phishing is the social engineering attacks and currently attacks on mobile devices. That might result in the form of financial loses. In this paper, we described many detection techniques using URL, Hyperlinks features that can be used to differentiate between the defective and non-defective website. There are six main approaches such as heuristic, blacklist, Fuzzy Rule, machine learning, image processing, and CANTINA based approach. It delivers a good consideration of the phishing issue, a present machine learning solution, and future study about Phishing threats by using machine learning Approach.

2. Y. Sönmez, T. Tuncer, H. Gökal, and E. Avci, “Phishing web sites features classification based on extreme learning machine,” 6th Int. Symp. Digit. Forensic Secur. ISDFS - Proceeding.

Phishing is a common attack on credulous people by making them to disclose their unique information using counterfeit websites. The objective of phishing website URLs is to purloin the personal information like user name, passwords and online banking transactions. Phishers use the websites which are visually and semantically similar to those real websites. As technology continues to grow, phishing techniques started to progress rapidly and this needs to be prevented by using anti-phishing mechanisms to detect phishing. Machine learning is a powerful tool used to strive against phishing attacks. This paper surveys the features used for detection and detection techniques using machine learning.

3. **T. Peng, I. Harris, and Y. Sawa, “Detecting Phishing Attacks Using Natural Language Processing and Machine Learning,” Proc. - 12th IEEE Int. Conf. Semant. Comput.**

Phishing attacks are one of the most common and least defended security threats today. We present an approach which uses natural language processing techniques to analyse text and detect inappropriate statements which are indicative of phishing attacks. Our approach is novel compared to previous work because it focuses on the natural language text contained in the attack, performing semantic analysis of the text to detect malicious intent. To demonstrate the effectiveness of our approach, we have evaluated it using a large benchmark set of phishing emails.

4. **M. Karabatak and T. Mustafa, “Performance comparison of classifiers on reduced phishing website dataset,” 6th Int. Symp. Digit. Forensic Secur. ISDFS - Proceeding.**

These days, numerous enemy of phishing frameworks are being created to recognize phishing substance in online correspondence frameworks. In spite of the accessibility of hordes hostile to phishing frameworks, phishing proceeds with unabated because of lacking recognition of a zero-day assault, pointless computational overhead and high bogus rates. In spite of the fact that Machine Learning approaches have accomplished promising exactness rate, the decision and the exhibition of the component vector limit their successful location. Phishing is a typical assault on guileless individuals by making them to unveil their one of a kind data utilizing fake sites. In this work, an upgraded AI based prescient model is proposed to improve the effectiveness of against phishing plans. The prescient model comprises of Feature Selection Module which is utilized for the development of a successful element vector. These highlights are removed from the URL, website page properties and site page conduct utilizing the gradual segment-based framework to introduce the resultant component vector to the prescient model. The proposed framework utilizes CNN, KNN AND SVM which have been prepared on a 30-dimensional list of capabilities. AI is an incredible asset used to endeavor against phishing assaults

5. K. Shima et al., “Classification of URL bitstreams using bag of bytes,” in 21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN).

In present days ,websites are main responsible for the rapid growth of criminal activities in the internet and corresponding activities which results in the many illegal things. So there are many preventive steps to be taken to stop these kind of activities. Here we propose a model which will classify the given URL into any of the three possible classes ,i.e. Benign, spam and malware. Our model will the detect the classification of the URL without using any websites content.

3. SYSTEM ANALYSIS & FEASIBILITY STUDY

3.1 EXISTING SYSTEM:

Where as in the case of the existing system means that what is previous system says a Manual human intervention is not that much applicable and error-prone. Legacy and Conventional Data Mining Algorithms can't deal with huge volumes of data, slower and inaccurate.

Disadvantages:

1. Late process
2. Its more time
3. No accurate result

3.2 PROPOSED SYSTEM:

Machine Learning is cutting edge and trending for different kinds of diverse application in the society where it can deal with tons of data, refined and revised algorithms, available heavy processing power in terms of GPU.

Advantages:

1. Fast process
2. Less time
3. Accurate result

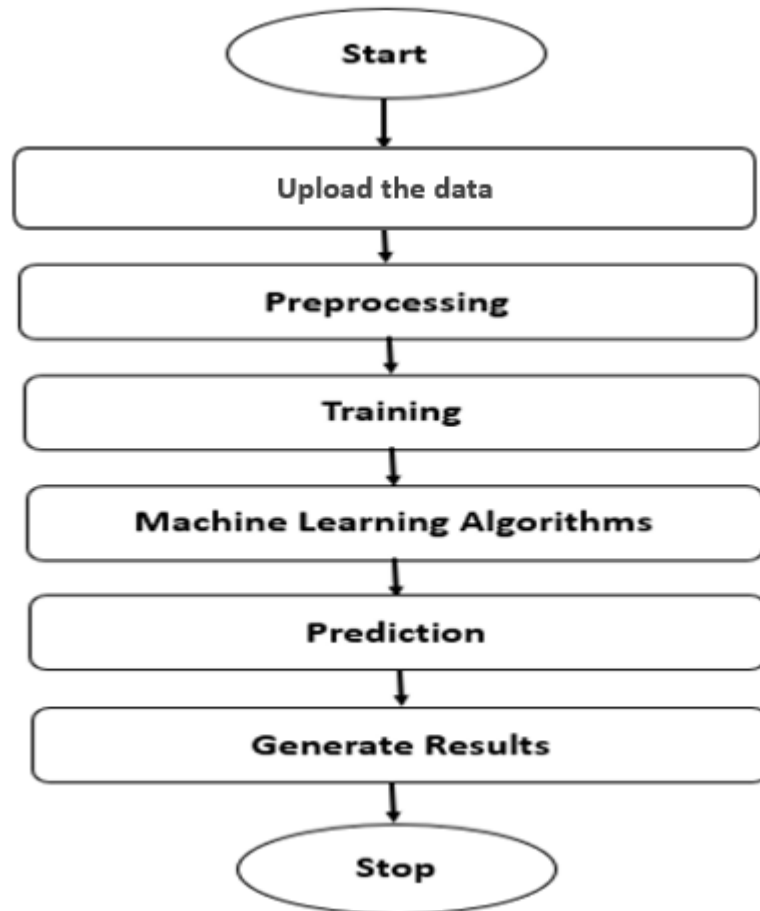
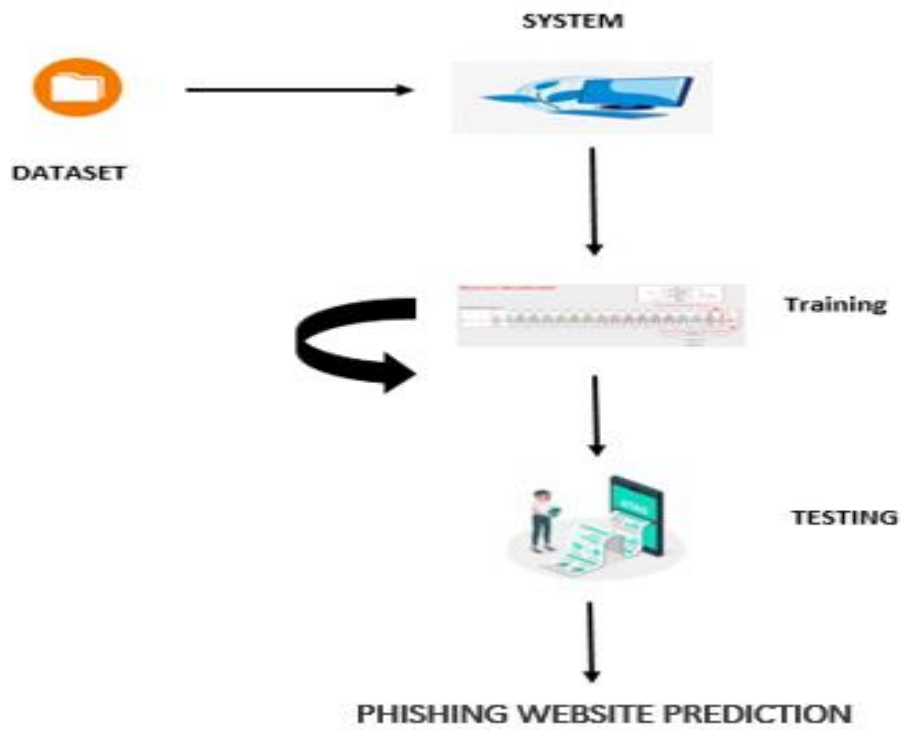
Block Diagram:

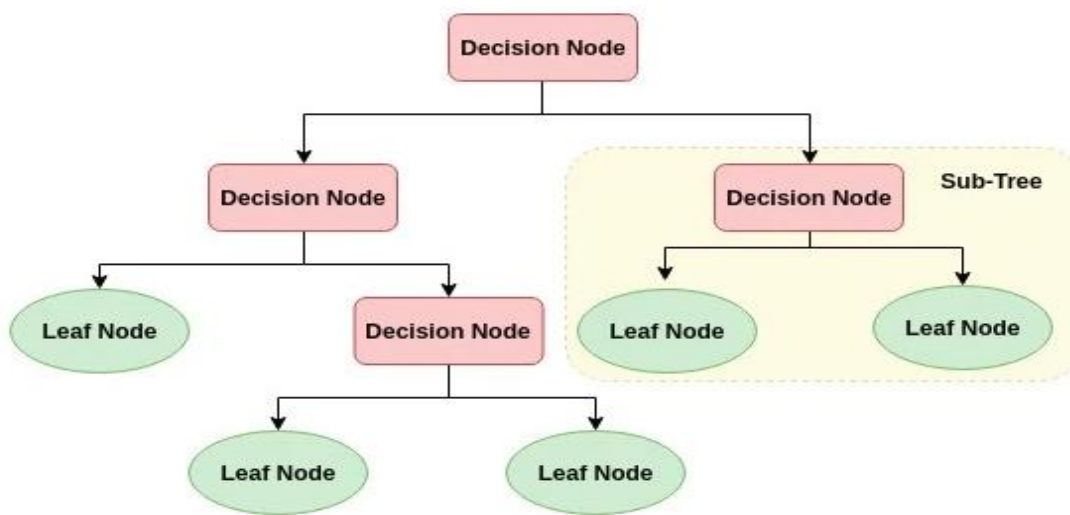
Fig: Block Diagram

Architecture:

4. METHODOLOGY AND ALGORITHMS:

4.1 DECISION TREE:

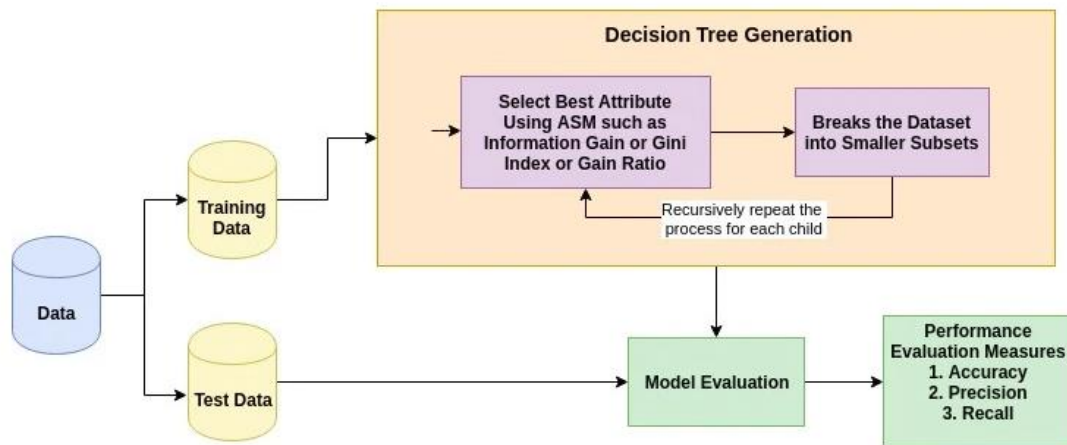
Decision tree is a flowchart-like tree structure where an internal node represents feature(or attribute), the branch represents a decision rule, and each leaf node represents the outcome. The topmost node in a decision tree is known as the root node. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps you in decision making. It's visualization like a flowchart diagram which easily mimics the human level thinking. That is why decision trees are easy to understand and interpret.



The basic idea behind any decision tree algorithm is as follows:

1. Select the best attribute using Attribute Selection Measures (ASM) to split the records.
2. Make that attribute a decision node and breaks the dataset into smaller subsets.
3. Starts tree building by repeating this process recursively for each child until one of the conditions will match:
 - All the tuples belong to the same attribute value.
 - There are no more remaining attributes.

- There are no more instances.



4.2 XGBOOST:

XGBoost stands for “Extreme Gradient Boosting”. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements Machine Learning algorithms under the Gradient Boosting framework. It provides a parallel tree boosting to solve many data science problems in a fast and accurate way.

Boosting

Boosting is an ensemble learning technique to build a strong classifier from several weak classifiers in series. Boosting algorithms play a crucial role in dealing with bias-variance trade-off. Unlike bagging algorithms, which only controls for high variance in a model, boosting controls both the aspects (bias & variance) and is considered to be more effective.

Below are the few types of boosting algorithms:

- AdaBoost (Adaptive Boosting)
- Gradient Boosting
- XGBoost

- CatBoost
- Light GBM

XGBoost:

XGBoost stands for eXtreme Gradient Boosting. It became popular in the recent days and is dominating applied machine learning and Kaggle competitions for structured data because of its scalability.

XGBoost is an extension to gradient boosted decision trees (GBM) and specially designed to improve speed and performance.

4.3 Random Forest Classifier:

A random forest is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.

A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

A random forest eradicates the limitations of a decision tree algorithm. It reduces the over fitting of datasets and increases precision. It generates predictions without requiring many configurations in packages (like Scikit-learn).

Features of a Random Forest Algorithm:

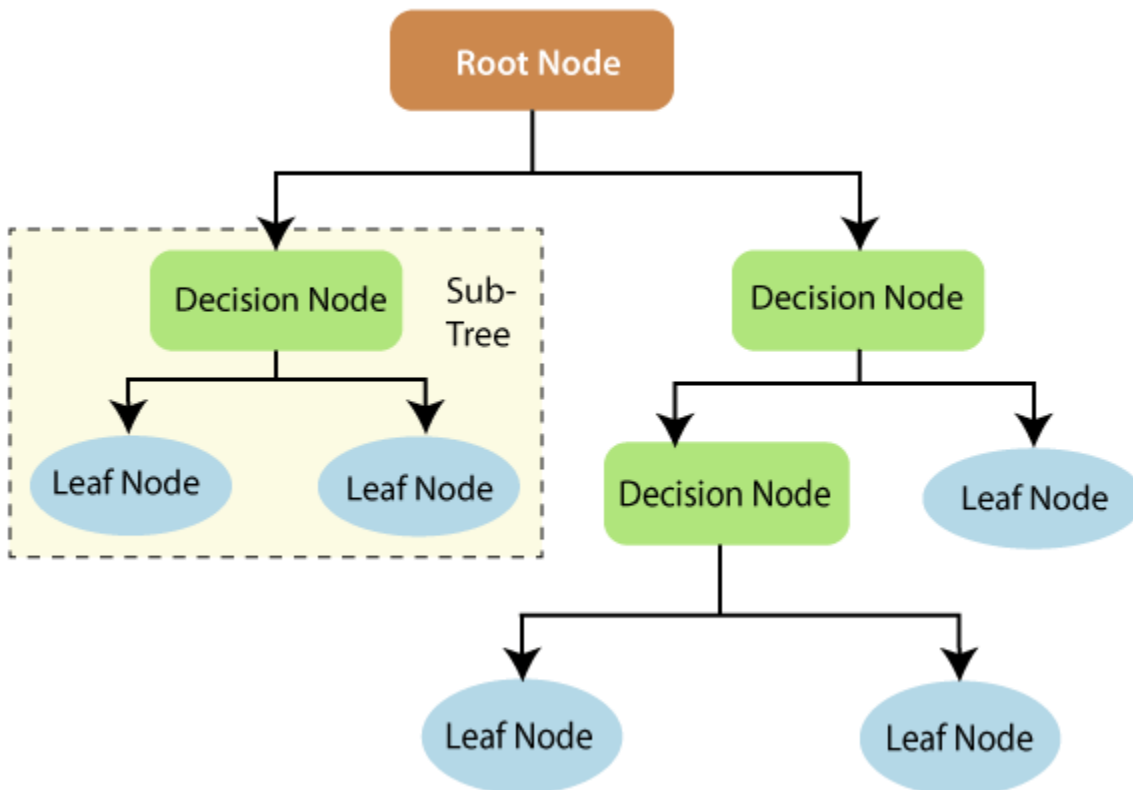
- It's more accurate than the decision tree algorithm.
- It provides an effective way of handling missing data.
- It can produce a reasonable prediction without hyper-parameter tuning.
- It solves the issue of over fitting in decision trees.

- In every random forest tree, a subset of features is selected randomly at the node's splitting point.

Decision trees are the building blocks of a random forest algorithm. A decision tree is a decision support technique that forms a tree-like structure. An overview of decision trees will help us understand how random forest algorithms work.

A decision tree consists of three components: decision nodes, leaf nodes, and a root node. A decision tree algorithm divides a training dataset into branches, which further segregate into other branches. This sequence continues until a leaf node is attained. The leaf node cannot be segregated further.

The nodes in the decision tree represent attributes that are used for predicting the outcome. Decision nodes provide a link to the leaves. The following diagram shows the three types of nodes in a decision tree.



The information theory can provide more information on how decision trees work. Entropy and information gain are the building blocks of decision trees. An overview of these fundamental concepts will improve our understanding of how decision trees are built.

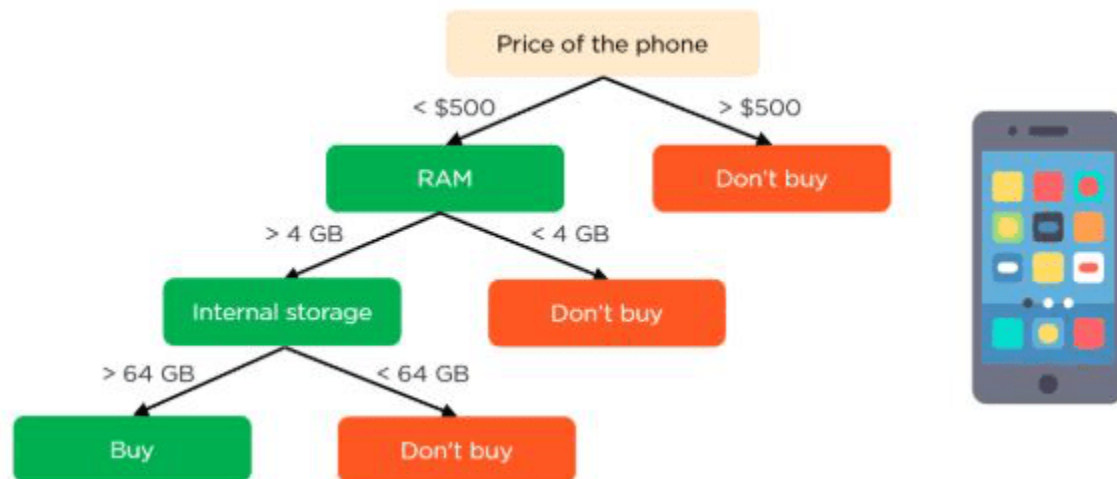
Entropy is a metric for calculating uncertainty. Information gain is a measure of how uncertainty in the target variable is reduced, given a set of independent variables.

The information gain concept involves using independent variables (features) to gain information about a target variable (class). The entropy of the target variable (Y) and the conditional entropy of Y (given X) are used to estimate the information gain. In this case, the conditional entropy is subtracted from the entropy of Y.

Information gain is used in the training of decision trees. It helps in reducing uncertainty in these trees. A high information gain means that a high degree of uncertainty (information entropy) has been removed. Entropy and information gain are important in splitting branches, which is an important activity in the construction of decision trees.

Let's take a simple example of how a decision tree works. Suppose we want to predict if a customer will purchase a mobile phone or not. The features of the phone form the basis of his decision. This analysis can be presented in a decision tree diagram.

The root node and decision nodes of the decision represent the features of the phone mentioned above. The leaf node represents the final output, either *buying* or *not buying*. The main features that determine the choice include the price, internal storage, and Random Access Memory (RAM). The decision tree will appear as follows.



Applying decision trees in random forest

The main difference between the decision tree algorithm and the random forest algorithm is that establishing root nodes and segregating nodes is done randomly in the latter. The random forest employs the bagging method to generate the required prediction.

Bagging involves using different samples of data (training data) rather than just one sample. A training dataset comprises observations and features that are used for making predictions. The decision trees produce different outputs, depending on the training data fed to the random forest algorithm. These outputs will be ranked, and the highest will be selected as the final output.

Our first example can still be used to explain how random forests work. Instead of having a single decision tree, the random forest will have many decision trees. Let's assume we have only four decision trees. In this case, the training data comprising the phone's observations and features will be divided into four root nodes.

The root nodes could represent four features that could influence the customer's choice (price, internal storage, camera, and RAM). The random forest will split the nodes by selecting features randomly. The final prediction will be selected based on the outcome of the four trees.

The outcome chosen by most decision trees will be the final choice. If three trees predict *buying*, and one tree predicts *not buying*, then the final prediction will be *buying*. In this case, it's predicted that the customer will buy the phone.

SOFTWARE DEVELOPMENT LIFE CYCLE – SDLC:

In our project we use waterfall model as our software development cycle because of its step-by-step procedure while implementing.

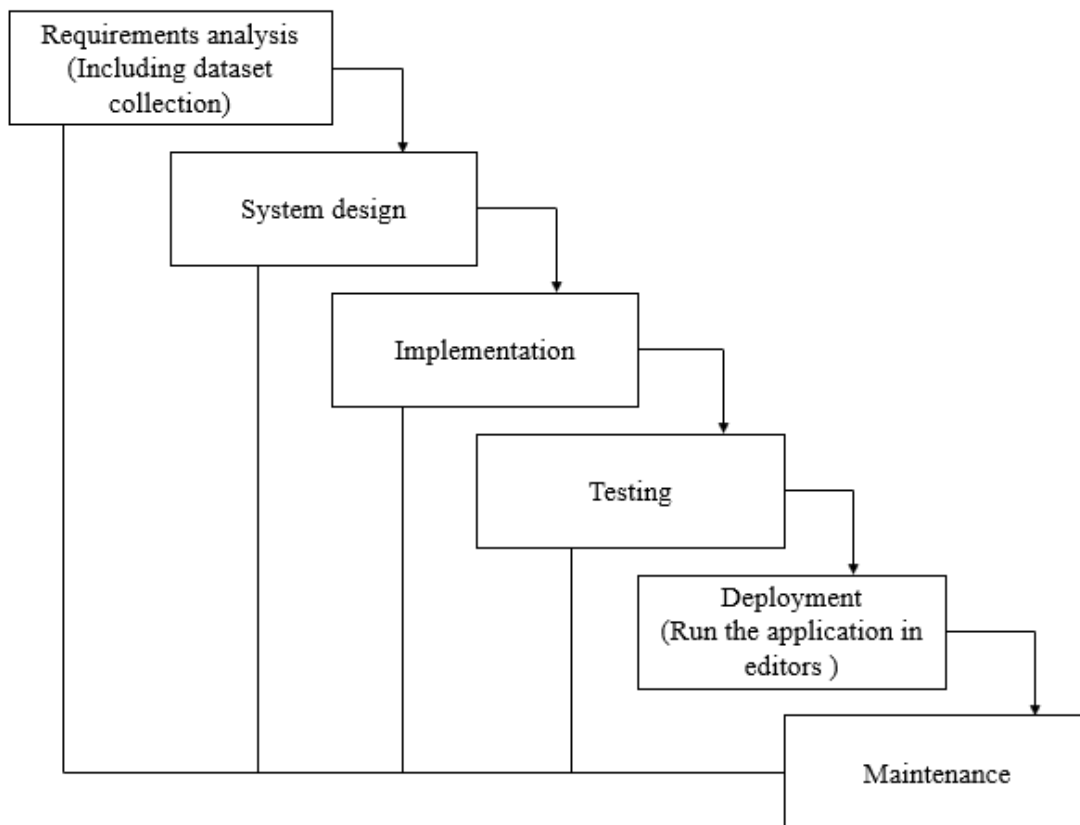


Fig1: Waterfall Model

- **Requirement Gathering and analysis** – All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** – the requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** – with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** – All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** – Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** – There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ ECONOMICAL FEASIBILITY
- ◆ TECHNICAL FEASIBILITY
- ◆ SOCIAL FEASIBILITY

Economic feasibility:

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical feasibility:

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends

on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM REQUIREMENTS SPECIFICATION

Functional and non-functional requirements:

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements: These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

- 1) Authentication of user whenever he/she logs into the system
- 2) System shutdown in case of a cyber-attack
- 3) A verification email is sent to user whenever he/she register for the first time on some software system.

Non-functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability

- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

- 1) Emails should be sent with a latency of no greater than 12 hours from such an activity.
- 2) The processing of each request should be done within 10 seconds
- 3) The site should load in 3 seconds whenever of simultaneous users are > 10000

SOFTWARE AND HARDWARE REQUIREMENTS:

Hardware:

Operating system	: Windows 7 or 7+
RAM	: 8 GB
Hard disc or SSD	: More than 500 GB
Processor	: Intel 3rd generation or high or Ryzen with 8 GB Ram

Software:

Software's	: Python 3.6 or high version
IDE	: PyCharm.
Framework	: Flask

8. SYSTEM DESIGN:

8.1 Input Design:

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

Therefore, the quality of system input determines the quality of system output. Well-designed input forms and screens have following properties –

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- It ensures proper completion with accuracy.
- It should be easy to fill and straightforward.
- It should focus on user's attention, consistency, and simplicity.
- All these objectives are obtained using the knowledge of basic design principles regarding –
 - What are the inputs needed for the system?
 - How end users respond to different elements of forms and screens.

Objectives for Input Design:

The objectives of input design are –

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.

- To use validation checks and develop effective input controls.

8.2 Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of input design are:

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end user's requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

8.3 MODULES:

1. User:

1.1 View Home page:

Here user view the home page of the phishing website prediction web application.

1.2 View Upload page:

In the about page, users can learn more about the phishing prediction.

1.3 Input Model:

The user must provide input values for the certain fields in order to get results.

1.4 View Results:

User view's the generated results from the model.

1.5 View score:

Here user have ability to view the score in %

2. System

2.1 Working on dataset:

System checks for data whether it is available or not and load the data in csv files.

2.2 Pre-processing:

Data need to be pre-processed according the models it helps to increase the accuracy of the model and better information about the data.

2.3 Training the data:

After pre-processing the data will split into two parts as train and test data before training with the given algorithms.

2.4 Model Building

To create a model that predicts the personality with better accuracy, this module will help user.

2.5 Generated Score:

2.6 Here user view the score in %

2.7 Generate Results:

We train the machine learning algorithm and calculate the personality prediction.

8.5 UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artefacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

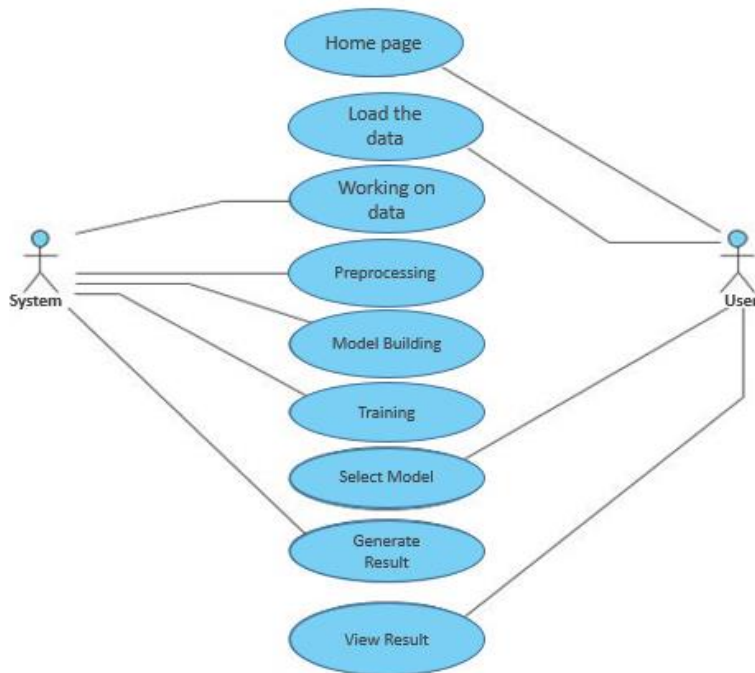
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

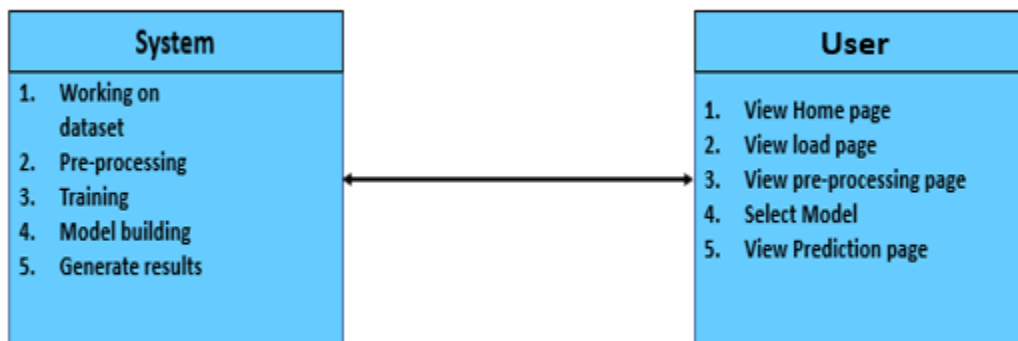
USE CASE DIAGRAM

- ▶ A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.
- ▶ Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.
- ▶ The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



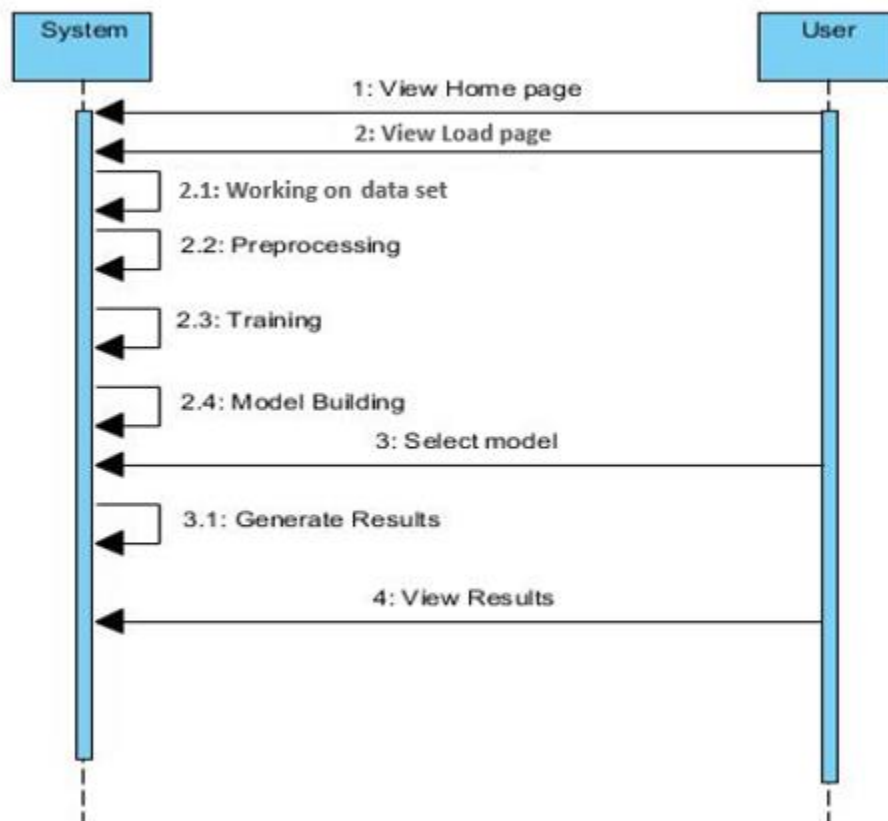
CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains informatio



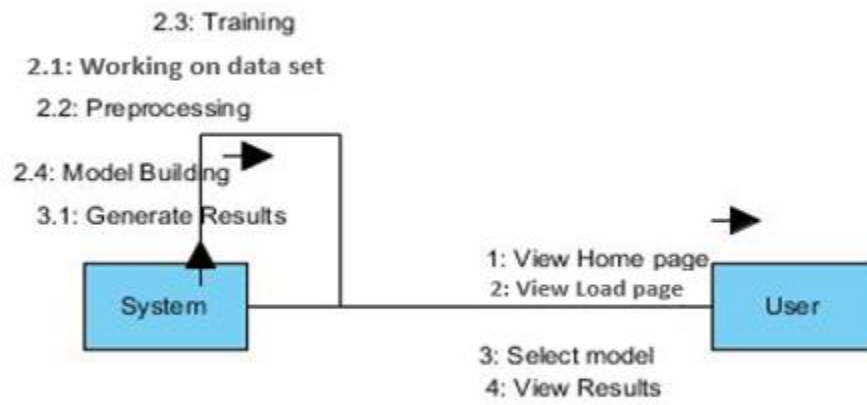
SEQUENCE DIAGRAM

- ▶ A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.
- ▶ It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams



COLLABORATION DIAGRAM:

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.



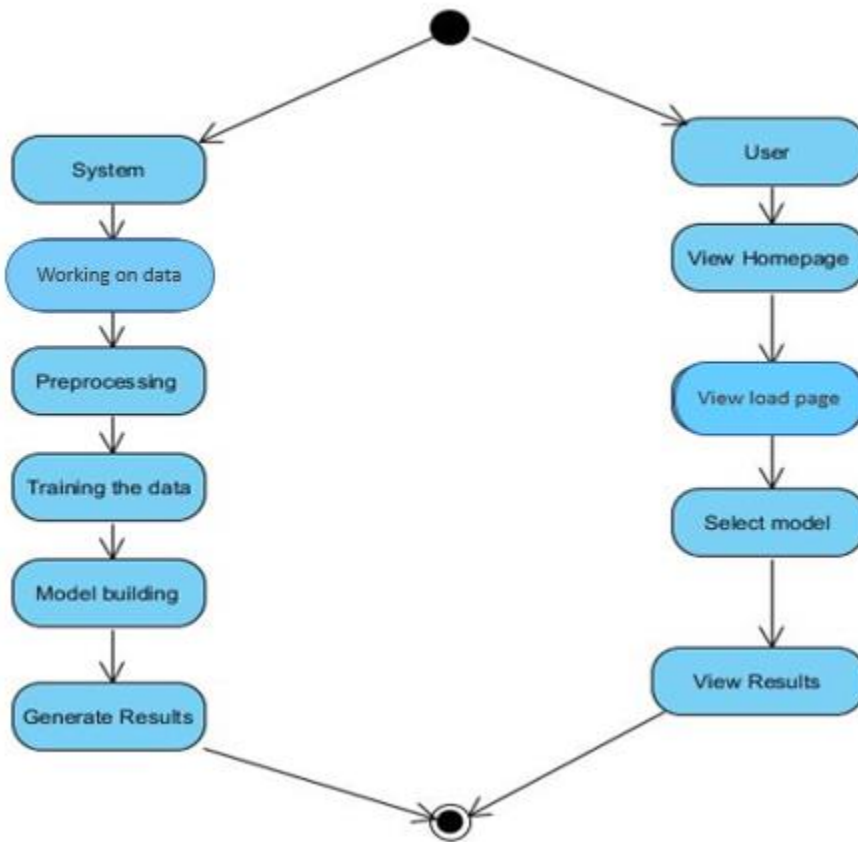
DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware's used to deploy the application.



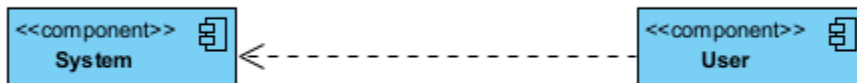
ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



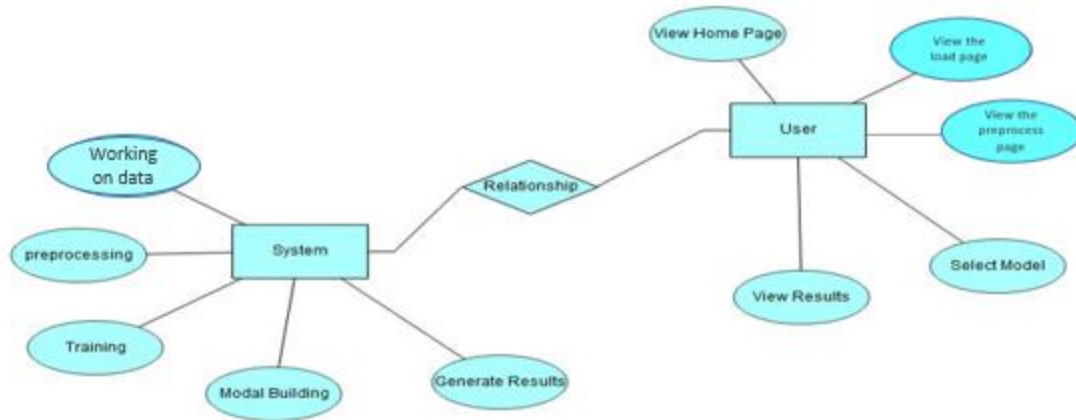
COMPONENT DIAGRAM:

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

**8.5 ER DIAGRAM:**

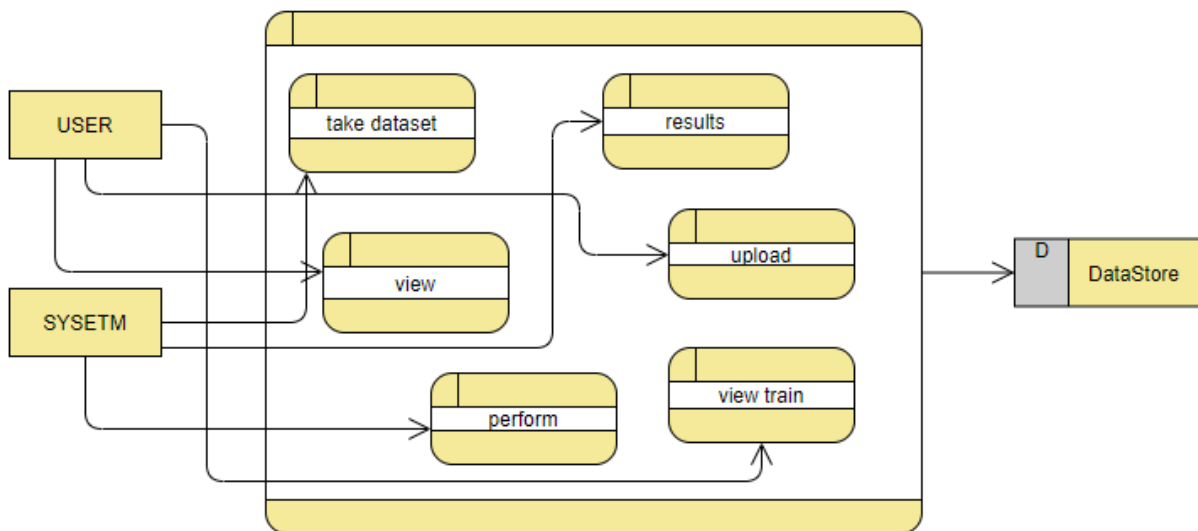
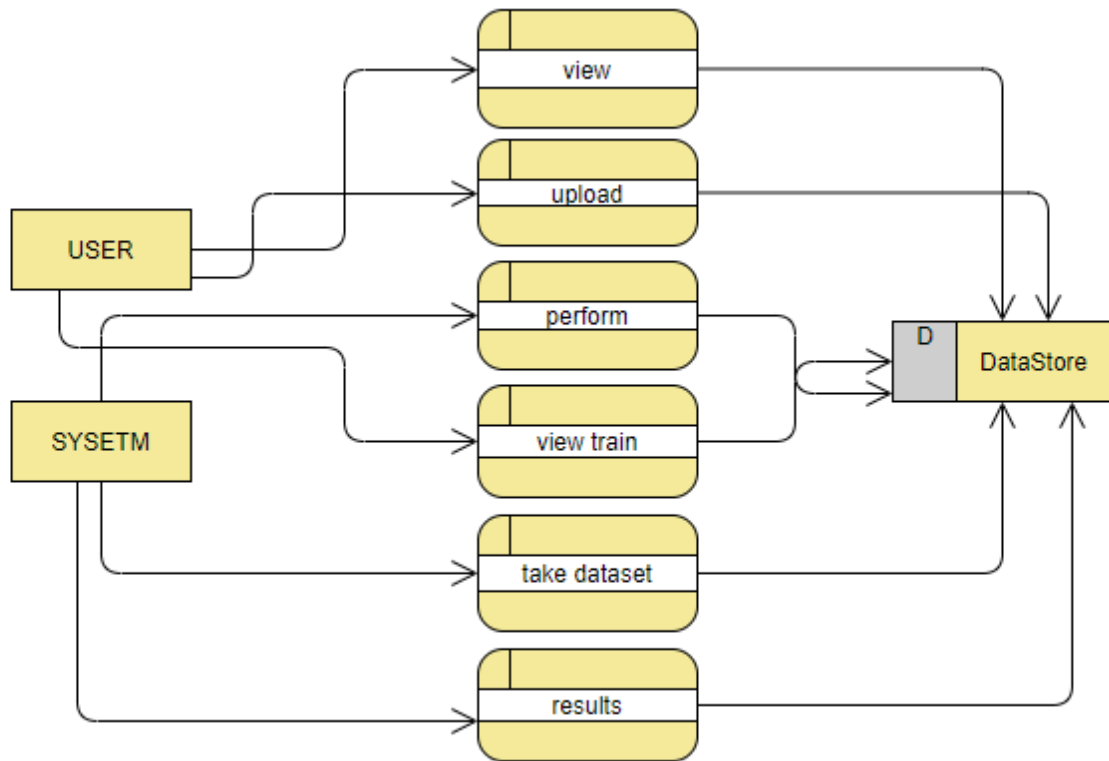
An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram). An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.



8.6 DFD DIAGRAM:

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.



9. TESTING

9.1 TESTING PROCEDURE

Software Testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is defect free.

These are the benefits of software testing.

- Cost-effectiveness
- Customer Satisfaction
- Security
- Product Quality

Testing is the process of attempting to find every possible flaw or error in a work product. It allows you to test the functionality of parts, subassemblies, assemblies, and/or finished products. It is a method of writing code with the goal of ensuring that the software meets its requirements and user expectations and does not fail in an unfavorable way. There are numerous types of tests. Each check sort reports a specific testing demand.

Before releasing or deploying any software application the testers use various testing processes and techniques for testing and validation. This process is known as software testing procedure. It can be done either manually or by using automated tools. Software testing is essential because it allows any faults or errors in the software to be found early and fixed before the software product is delivered. Reliability, security, and high performance are all provided by properly tested software, which also leads to time savings, cost effectiveness, and customer pleasure.

9.2 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.

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

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. 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.

6.1 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.

6.2 Integration Testing

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.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

6.3 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.

10. OUTPUT SCREEN SHOTS WITH DESCRIPTION.

Home Page:

Here user view the home page of phishing website prediction web application.

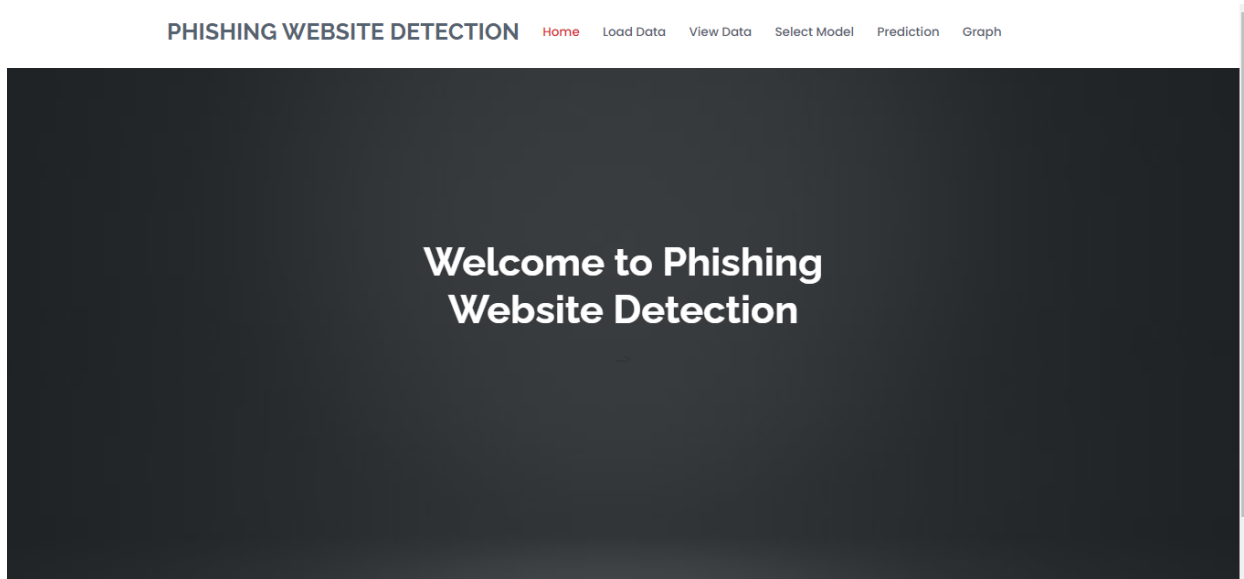
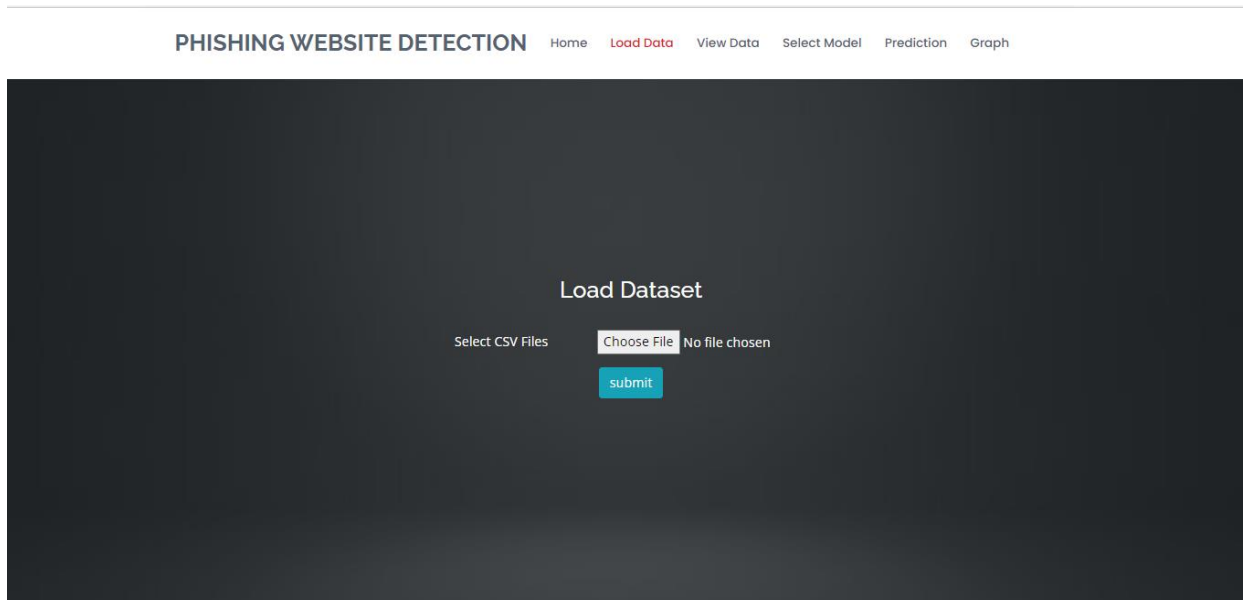


Fig1: Home Page

Load:

In the load page, users can load the website dataset.



View:

Here we can see the uploaded data set.

PHISHING WEBSITE DETECTION					
Home Load Data View Data Select Model Prediction Graph					
S/N	Domain	Have_IP	Have_At	URL_Len	
1	graphicriver.net	0	0	1	
2	ecnavi.jp	0	0	1	
3	hubpages.com	0	0	1	
4	extratorrent.cc	0	0	1	
5	icicibank.com	0	0	1	
6	nypost.com	0	0	1	
7	kienthuc.net.vn	0	0	1	
8	thenextweb.com	0	0	1	
9	tobogo.net	0	0	1	
10	akhbarelom.com	0	0	1	
11	tunein.com	0	0	1	
12	tune.pk	0	0	1	
13	sfglobe.com	0	0	1	
14	mic.com	0	0	1	
15	thenextweb.com	0	0	1	

Model:

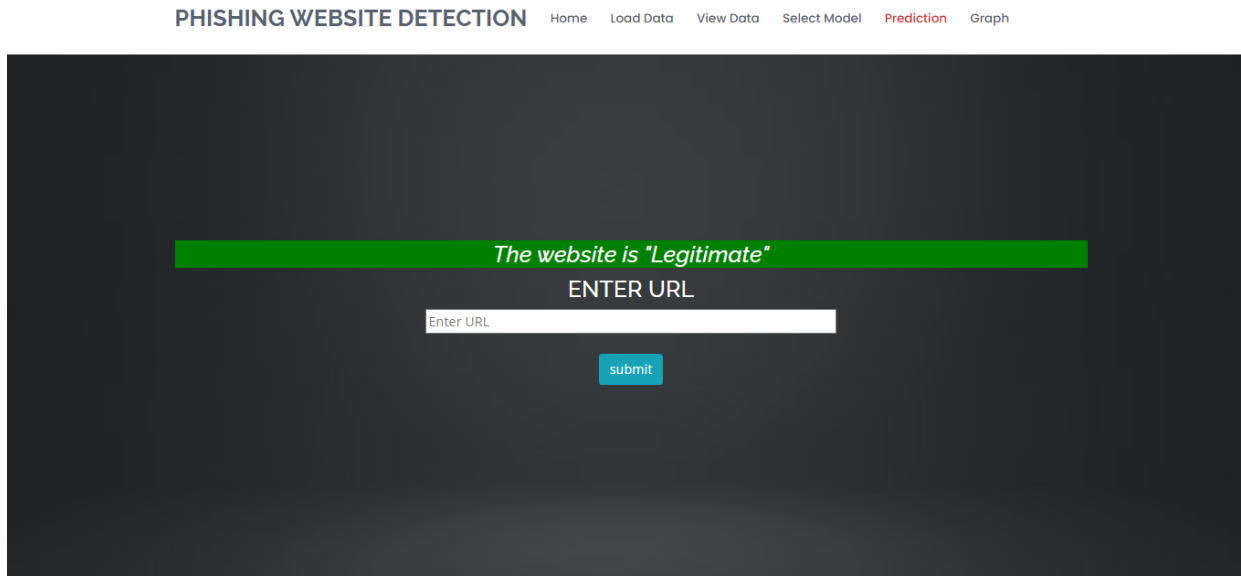
Here we can train our data using different algorithm.

PHISHING WEBSITE DETECTION					
Home Load Data View Data Select Model Prediction Graph					
Model Selection					
Select Model <input type="text" value="Select an option"/>					
<input type="submit" value="submit"/>					

Prediction:

This page show the detection result that whether the website is a phishing website or legitimate.

PHISHING WEBSITE DETECTION [Home](#) [Load Data](#) [View Data](#) [Select Model](#) [Prediction](#) [Graph](#)



The website is "Legitimate"

ENTER URL

Enter URL

submit

11. CONCLUSION:

This project presented various algorithms and approaches to detect phishing websites by several researchers in Machine Learning. On reviewing the papers, we came to a conclusion that most of the work done by using familiar machine learning algorithms like XGBoost, Decision Tree and Random Forest and MLP classifier which generates the neural network results. Some authors proposed a new system like Phish Score and Phish Checker for detection. The combinations of features with regards to accuracy, precision, recall etc. were used. As phishing websites increases day by day, some features may be included or replaced with new ones to detect them.

11. FUTURE SCOPE

There are quite a few things that can be polished or be added in the future work. • We have opted to use two data mining classifiers in this project namely the ID3 and Naive Bayes classifier. There are more classifiers such as the Bayesian network classifier, Neural Network classifier and C4.5 classifier. Such classifiers were not included in this project and could be counted in future to give a more data to be compared with.

13. REFERENCES:

1. J. Shad and S. Sharma, "A Novel Machine Learning Approach to Detect Phishing Websites Jaypee Institute of Information Technology," pp. 425–430, 2018.
2. Y. Sönmez, T. Tuncer, H. Gökal, and E. Avci, "Phishing web sites features classification based on extreme learning machine," 6th Int. Symp. Digit. Forensic Secur. ISDFS 2018 - Proceeding, vol. 2018–Janua, pp. 1–5, 2018.
3. T. Peng, I. Harris, and Y. Sawa, "Detecting Phishing Attacks Using Natural Language Processing and Machine Learning," Proc. - 12th IEEE Int. Conf. Semant. Comput. ICSC 2018, vol. 2018–Janua, pp. 300–301, 2018.
4. M. Karabatak and T. Mustafa, "Performance comparison of classifiers on reduced phishing website dataset," 6th Int. Symp. Digit. Forensic Secur. ISDFS 2018 - Proceeding, vol. 2018–Janua, pp. 1–5, 2018.
5. S. Parekh, D. Parikh, S. Kotak, and P. S. Sankhe, "A New Method for Detection of Phishing Websites: URL Detection," in 2018 Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018, vol. 0, no. Iicct, pp. 949–952.
6. K. Shima et al., "Classification of URL bitstreams using bag of bytes," in 2018 21st Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), 2018, vol. 91, pp. 1–5.
7. A. Vazhayil, R. Vinayakumar, and K. Soman, "Comparative Study of the Detection of Malicious URLs Using Shallow and Deep Networks," in 2018 9th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2018, 2018, pp. 1–6.
8. W. Fadheel, M. Abusharkh, and I. Abdel-Qader, "On Feature Selection for the Prediction of Phishing Websites," 2017 IEEE 15th Intl Conf Dependable, Auton. Secur. Comput. 15th Intl Conf Pervasive Intell. Comput. 3rd Intl Conf Big Data Intell. Comput. Cyber Sci. Technol. Congr., pp. 871–876, 2017.
9. X. Zhang, Y. Zeng, X. Jin, Z. Yan, and G. Geng, "Boosting the Phishing Detection Performance by Semantic Analysis," 2017.

10. L. MacHado and J. Gadge, "Phishing Sites Detection Based on C4.5 Decision Tree Algorithm," in 2017 International Conference on Computing, Communication, Control and Automation, ICCUBEA 2017, 2018, pp. 1–5.

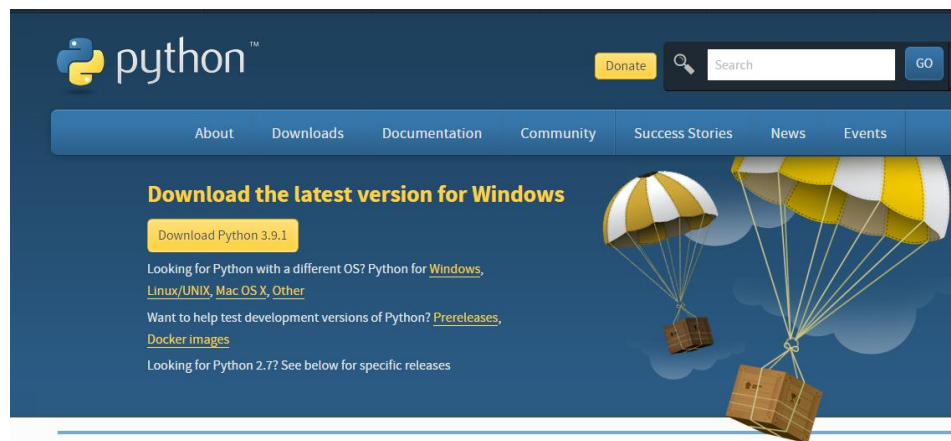
11.

BIBLIOGRAPHY:

SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECTS:

Installing Python:

1. To download and install Python visit the official website of Python <https://www.python.org/downloads/> and choose your version.



2. Once the download is complete, run the exe for install Python. Now click on Install Now.
3. You can see Python installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm:

1. To download PyCharm visit the website <https://www.jetbrains.com/pycharm/download/> and click the "DOWNLOAD" link under the Community Section.

Download PyCharm

[Windows](#)[Mac](#)[Linux](#)

Professional

For both Scientific and Web Python development. With HTML, JS, and SQL support.

[Download](#)

Free trial

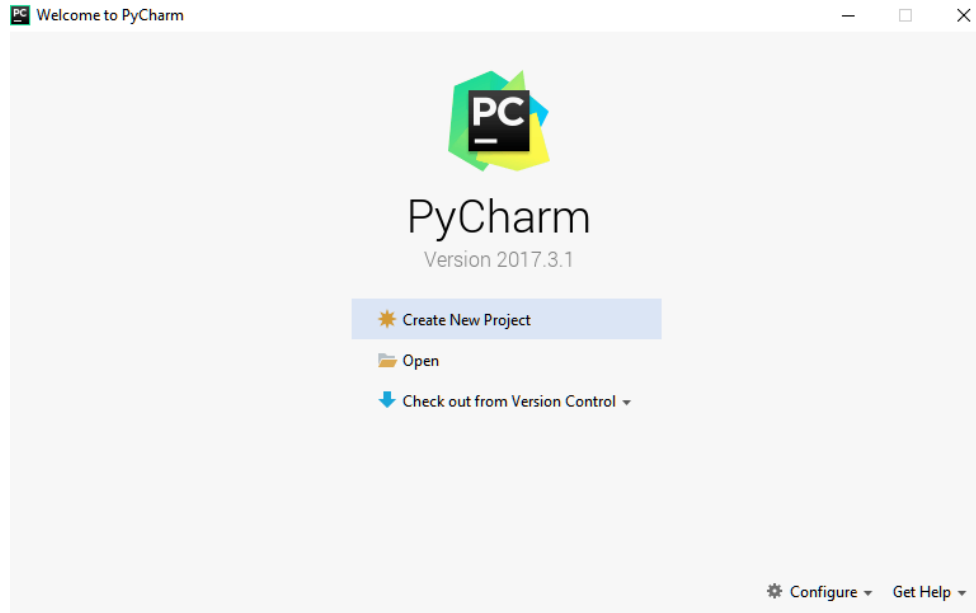
Community

For pure Python development

[Download](#)

Free, open-source

2. Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.
3. On the next screen, Change the installation path if required. Click “Next”.
4. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
5. Choose the start menu folder. Keep selected JetBrains and click on “Install”.
6. Wait for the installation to finish.
7. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
8. After you click on "Finish," the Following screen will appear.



9. You need to install some packages to execute your project in a proper way.
10. Open the command prompt/ anaconda prompt or terminal as administrator.
11. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, sea born, scikit-learn, Matplotlib, Pyplot)

Ex: Pip install NumPy

```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    | 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```