A

Major Project Report

## On

**“Fake Job Post Detecting System Using Machine Learning”**

Submitted in partial fulfillment of the Requirements for the award of the degree of

# Bachelor of Technology In

**Computer Science & Engineering – Cyber Security By**

## A BHUVAN REDDY – 20R21A6201

Under the guidance of

## Mr. Vaziuddin Mohammed Assistant Professor

**Department of Data Science & Cyber Security**



**2024**



**Department of Data Science & Cyber Security**

# CERTIFICATE

This is to certify that the project entitled **“Fake Job Post Detecting System Using Machine Learning”** has been submitted by **A Bhuvan Reddy (20R21A6201)** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering – Cyber Security from Jawaharlal Nehru Technological University, Hyderabad. The results embodied in this project have not been submitted to any other University or Institution for the award of any degree or diploma.

#### Internal Guide Head of the Department

**External Examiner**



**Department of Data Science & Cyber Security**

# DECLARATION

I hereby declare that the project entitled **“Fake Job Post Detecting System Using Machine Learning”** is the work done during the period from **July 2023 to April 2024** and is submitted in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering – Cyber Security from Jawaharlal Nehru Technology University, Hyderabad. The results embodied in this project have not been submitted to any other university or Institution for the award of any degree or diploma.

#### A Bhuvan Reddy – 20R21A6201



**Department of Data Science & Cyber Security**

# ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned my efforts with success. It is a pleasant aspect that I now have the opportunity to express my gratitude to all of them.

First of all, I would like to express my deep gratitude towards our internal guide **Mr**. **VAZZIUDDIN MOHAMMED**, Assistant Professor, Dept. of Data Science & Cyber Security for his support in the completion of my dissertation. I wish to express my sincere thanks to **Dr. CHIRNAJEEVI MANIKE**, HOD, Dept. of Data Science & Cyber Security, and Principal **Dr. K. SRINIVAS RAO** for providing the facilities to complete the dissertation.

I would like to thank all my faculty and friends for their help and constructive criticism during the project period. Finally, I’m very much indebted to my parents for their moral support and encouragement to achieve my goals.

#### A Bhuvan Reddy – 20R21A6201



**Department of Data Science & Cyber Security**

# ABSTRACT

The proliferation of fake news and fraudulent activities, particularly in the realm of online job postings, poses significant challenges to both individuals and organizations. In response to this pressing issue, this project conducts a comprehensive analysis of research related to fake news detection and explores the application of traditional machine learning models for identifying and mitigating fraudulent job postings on the internet.

Through the utilization of machine learning-based classification techniques, an automated tool is proposed to detect and filter out fraudulent job postings from a vast pool of online listings. The project evaluates the efficacy of various classifiers, including both single classifiers and ensemble classifiers, in discerning fraudulent job postings. Experimental results underscore the superiority of ensemble classifiers in detecting employment scams over their single-classifier counterparts.

The proposed approach leverages supervised machine learning algorithms, with a focus on utilizing Python's sci-kit-learn library and natural language processing (NLP) techniques for textual analysis. Feature extraction and vectorization are performed to facilitate the classification process, with tools such as Count Vectorizer and TfidfVectorizer from sci-kit-learn being employed for tokenization and feature extraction of textual data. The development of an effective detection model holds the potential to mitigate the social and economic damages caused by fraudulent job postings, thereby safeguarding individuals and organizations from potential exploitation and harm.

# APPENDIX-I LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **Figure Number** | **Name of the Figure** | **Page Number** |
| 1 | Machine Learning Process | 2 |
| 2 | Python Installation Steps | 13 |
| 3 | Architecture Diagram | 23 |
| 4 | Entity Relationship Model | 24 |
| 5 | Data Flow Diagram | 25 |
| 6 | Use Case Diagram | 27 |
| 7 | Fake Job Prediction Interface | 53 |

INDEX

CHAPTER 1 1

INTRODUCTION 1-2

CHAPTER 2 3

LITERATURE SURVEY 3-4

[CHAPTER 3 5](#_bookmark0)

[SOFTWARE REQUIREMENT ANALYSIS 5](#_bookmark1)

* 1. [MODULES AND FUNCTIONALITIES 5](#_bookmark2)
  2. [FUNCTIONAL REQUIREMENTS 5-6](#_bookmark3)
  3. [NON-FUNCTIONAL REQUIREMENTS 6-8](#_bookmark4)
  4. FEASIBILITY STUDY 8-9
  5. [ALGORITHMS USED 9-11](#_bookmark5)

[CHAPTER 4 12](#_bookmark6)

SOFTWARE REQUIREMENTS SPECIFICATION 12

* 1. [HARDWARE REQUIREMENTS 12](#_bookmark7)
  2. [SOFTWARE REQUIREMENTS 12](#_bookmark8)
  3. SELECTED SOFTWARE 12-23

[CHAPTER 5 23](#_bookmark9)

[SOFTWARE DESIGN 23](#_bookmark10)

* 1. [ARCHITECTURE DIAGRAM 23](#_bookmark11)
  2. [ENTITY RELATIONSHIP DIAGRAM 24](#_bookmark12)
  3. [DATA FLOW DIAGRAM 25](#_bookmark13)
  4. UML DIAGRAM 26-31

[CHAPTER 6 32](#_bookmark14)

[CODING AND IMPLEMENTATION 32](#_bookmark15)

* 1. [PROJECT METHODOLOGY 32-33](#_bookmark16)
  2. [SAMPLE CODE 33-49](#_bookmark17)

### CHAPTER 7 50

SYSTEM TESTING 50

* 1. TESTING STRATEGIES 50-53
  2. SAMPLE TEST CASES 53-54

### CHAPTER 8 55

OUTPUT SCREENS 55-59

### CHAPTER 9 60

CONCLUSION 60

### CHAPTER 10 61

FUTURE ENHANCEMENTS 61

### CHAPTER 11 62

REFERENCES 62-64

**vii**

# CHAPTER 1 INTRODUCTION

Employment scams are one of the serious issues in recent times addressed in the domain of Online Recruitment Frauds (ORF). In recent days, many companies prefer to post their vacancies online so that they can be accessed easily and timely by job-seekers. However, this intention may be one type of scam by the fraud people because they offer employment to job- seekers in terms of taking money from them. Fraudulent job advertisements can be posted against a reputed company for violating its credibility. This fraudulent job post detection draws good attention to obtaining an automated tool for identifying fake jobs and reporting them to people to avoid applications for such jobs.

For this purpose, a machine learning approach is applied which employs several classification algorithms for recognizing fake posts. In this case, a classification tool isolates fake job posts from a larger set of job advertisements and alerts the user. To address the problem of identifying scams on job postings, supervised learning algorithms as classification techniques are considered initially. A classifier maps input variables to target classes by considering training data. Classifiers addressed in the Project for identifying fake job posts from others are described briefly. This classifiers-based prediction may be broadly categorized into –Single Classifier based Prediction and Ensemble Classifiers based Prediction. And real Job Recommendation system.

#### WHY MACHINE LEARNING

To better understand the uses of machine learning, consider some of the instances where machine learning is applied: the self-driving Google car, cyber fraud detection, online showcasing the movies and shows you might like, and “more items to

consider” and “get yourself a little something” on Amazon—are all examples of applied machine learning. recommendation engines—like friend suggestions on Facebook, Netflix

All these examples echo the vital role machine learning has begun to take in today’s data- rich world. Machines can aid in filtering useful pieces of information that help in major advancements, and we are already seeing how this technology is being implemented in a wide variety of industries.

The process flow depicted here represents how machine learning works:

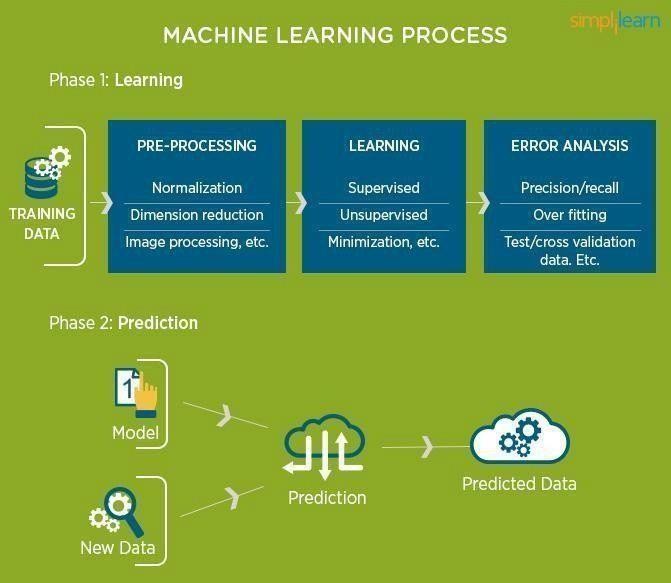


Fig 1: Machine learning process

With the constant evolution of the field, there has been a subsequent rise in the uses, demands, and importance of machine learning. Big data has become quite a buzzwordin the last few years; that’s in part due to the increased sophistication of machine learning, which helps analyze those big chunks of big data. Machine learning has also changed the way data extraction, and interpretation is done by involving automatic sets of generic methods that have replaced traditional statistical techniques.

# CHAPTER 2 LITERATURE SURVEY

1. **"Detecting Fake Job Postings on the Internet" AUTHOR:** Guangyu Zhou, Yuxin Su, and Zongjie Ma.

This study intends a means for detecting fake task postings by resolving various countenance to a degree task writing, company facts, and consumer date versification. The authors employ machine intelligence methods, containing theme mining and emotion study, to categorize task postings as genuine or fake.

1. **"A Hybrid Approach for Fake Job Detection on Online Recruitment Platforms" AUTHOR:** Md Shafiqul Islam and Asif Bin Anwar.

This research presents a composite approach joining machine intelligence algorithms and rule- located techniques to label fake task postings. The study survey the influence of features to a degree task writing distance, payroll range, and geographic neighbourhood in distinctive 'tween legal and deceptive task record.

1. **"Job Recommendation System Based on Machine Learning Algorithms" AUTHOR:** Ravindra Pratap Singh, Alok Singh Chauhan, and Harshit Aggarwal.

Focusing on the approval facet, this paper proposes a task approval scheme that resorts to machine learning algorithms to couple task applicants accompanying relevant freedom. The authors survey cooperative refining and content-based procedures to embody task pieces of advice based on consumer priorities and classical interplays.

1. **"Fake Job Posting Detection on Social Media Using Machine Learning Techniques" AUTHOR:** Srinivas Mukkamala and Sagar Samtani.

This study investigates the detection of fake task postings on public news floors, leveraging machine learning methods to a degree support heading machines (SVM) and haphazard jungles. The authors analyse countenance elicited from task postings, consumer profiles, and date patterns to change 'tween real and fraudulent record.

#### "Enhanced Fake Job Detection Using Machine Learning and Natural Language Processing"

**AUTHOR:** Elena-Simona Apostol, Mihaela Mocanu, and Cristian-Dragos Vizitiu.

Focusing on the unification of robotics (NLP) methods, this research proposes an embellished approach for fake task discovery. The study survey the use of word embeddings and pertaining to syntax study to extract significant looks from job postings, reconstructing the veracity of machine intelligence models in recognizing deceptive record.

1. **"Personalized Job Recommendation System Using Machine Learning Algorithms" AUTHOR:** Rajesh Kanna R and Vengattaraman T.

This paper presents a embodied task approval system that applies machine intelligence algorithms to couple task applicants with acceptable positions. The authors examine the influence of cooperative filtering, model factorization, and deep knowledge methods in produce accurate and appropriate task pieces of advice established user options and background.

# CHAPTER 3

# SOFTWARE REQUIREMENT ANALYSIS

## MODULES AND FUNCTIONALITIES

#### Service Provider

In this module, the service provider has to login by using valid user name and password. After login successful he can do some operations such as train and test data sets, view trained and tested accuracy in bar chart, view trained and tested accuracy results, predict job post type details, find job post type prediction ratio, download trained data sets, view job post type prediction ratio results, view all remote users.

#### View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user’s details such as, user name, email, address and admin authorises the users.

#### Remote User

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once login is successful user will do some operations like register and login, post job post data sets, predict job post prediction, view your profile.

## FUNCTIONAL REQUIREMENTS

* The system should be able to identify and extract relevant data from job posts.
* The system should be able to identify fake job posts with a high degree of accuracy.
* The system should be able to handle large amounts of data efficiency.
* The system should be able to categorize job posts based on various parameters such as industry, location, job type, etc.
* The system should be able to perform data analysis and generate informative reports for users.
* The system should have a user-friendly interface for easy navigation and understanding of the results.
* The system should ensure data security and privacy for users and job seekers.
* The system should be scalable to accommodate future additions and modifications to the system.
* The system should be accessible from multiple devices and platforms.
* The system should be able to integrate with other applications or system if required.

## NON-FUNCTIONAL REQUIREMENTS

Systems engineering and requirements engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviours. They are contrasted with functional requirements that define specific behaviour or functions. Non-functional requirements add tremendous value to business analysis. It is commonly misunderstood by a lot of people. It is important for business stakeholders, and Clients to clearly explain the requirements and their expectations in measurable terms. If the nonfunctional requirements are not measurable then they should be

revised or rewritten to gain better clarity. For example, User stories help in mitigating the gap between developers and the user community in Agile Methodology.

## Usabilty

Prioritize the important functions of the system based on usage patterns. Frequently, used functions should be tested for usability, as should complex and critical functions. Be sure to create a requirement for this.

## Reliability

* Reliability defines the trust in the system that is developed after using it for a period of time. It defines the likeability of the software to work without failure for a given time period.
* The number of bugs in the code, hardware failures, and problems can reduce there liability of the software.
* Your goal should be a long MTBF (mean time between failures). It is defined as the average period of time the system runs before failing.
* Create a requirement that data created in the system will be retained for a number of years without the data being changed by the system.

## Performance

What should system response times be, as measured from any point, under what circumstances?

## Supportability

* The system needs to be cost-effective to maintain.
* Maintainability requirements may cover diverse levels of documentation, such as system documentation, as well as test documentation, e.g. which test cases and test plans will accompany the system.

## FEASIBILTY STUDY

The feasibility of the project is analyzed 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

## Economical 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.

## ALGORITHMS USED

#### Logistic Regression Algorithm

Logistic regression is a supervised machine learning algorithm mainly used for classification tasks where the goal is to predict the probability that an instance of belonging to a given class. It is used for classification algorithms its name is logistic regression.

Here are some common terms involved in logistic regression:

* Independent variables: The input characteristics or predictor factors applied to the dependent variable’s predictions.
* Dependent variable: The target variable in a logistic regression model, which we are trying to predict.

#### Support Vector Machine Algorithm

Support Vector Machine (SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well it’s best suited for classification. The objective of the SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. If the number of input features is two, then the hyperplane is just a line. If the number of input features is three, then the hyperplane becomes a 2-D plane. It becomes difficult to imagine when the number of features exceeds three.

#### Naive Bayes

* Naive Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
* It is mainly used in text classification that includes a high-dimensional training dataset.
* Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
* It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
* Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.
* It is mainly used in text classification that includes a high-dimensional training dataset.

#### Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains several decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, it predicts the final output.

# CHAPTER 4

**SOFTWARE REQUIREMENT SPECIFICATION**

## HARDWARE REQUIREMENTS

Processor : Pentium –IV

RAM : 4 GB (min)

Hard Disk : 200 GB

#### SOFTWARE REQUIREMENTS

Operating system : Windows 7 Ultimate. Coding Language : Python.

Front-End : Python.

Back-End : Django-ORM

Designing : Html, Css, Javascript.

Data Base : MySQL (WAMP Server)

#### SELECTED SOFTWARE Introduction To Python:

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



Fig 2: Python Installation

* **Python is Interpreted:** Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERLand PHP.
* **Python is Interactive:** You can sit at a Python prompt and interact with the interpreter directly to write your programs.
* **Python is Object-Oriented:** Python supports an Object-Oriented style or technique of programming that encapsulates code within objects.
* **Python is a Beginner's Language:** Python is a great language for beginner- level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

## Installation of Python

**Step 1:** Go to Download and Open the downloaded Python version to carry out the installation process.

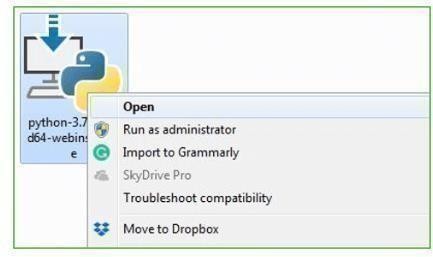


Fig 2.1

**Step 2:** Before you click on Install Now, make sure to put a tick on Add Python 3.7 to PATH.



Fig 2.2

**Step 3:** Click on Install NOW After the installation is successful. Click on Close.

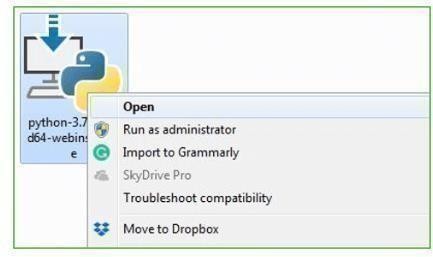


Fig 2.3

With these above three steps on Python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

**Verify the Python Installation Step 1:** Click on Start.

**Step 2:** In the Windows Run Command, type “cmd”.

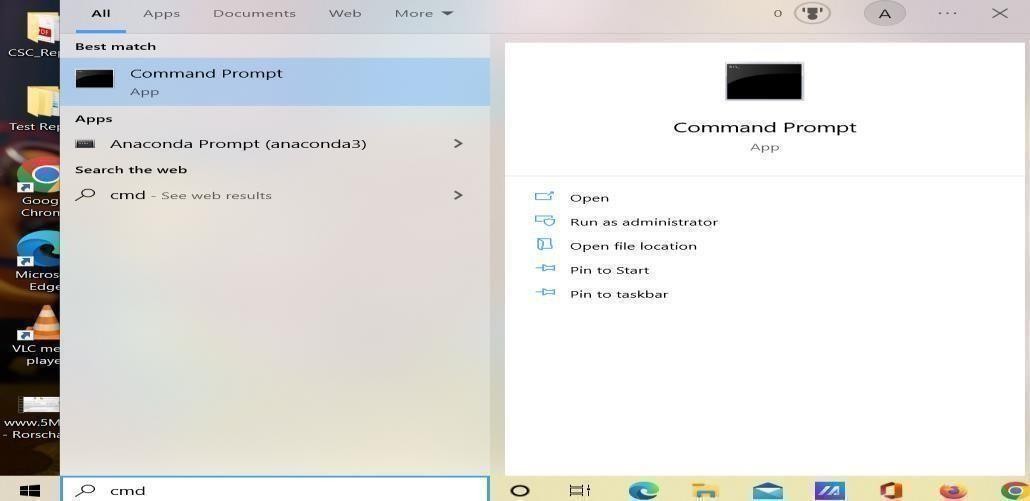


Fig 2.4

**Step 3:** Open the Command prompt option.

**Step 4:** Let us test whether the python is correctly installed. Type python –V and press Enter.

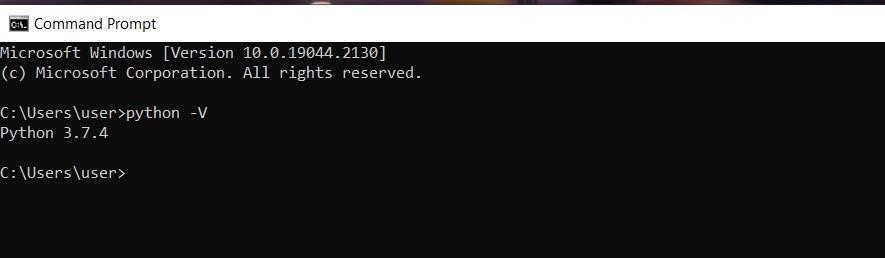


Fig 2.5

**Step 5:** You will get the answer as 3.7.4 **Check how the Python IDLE worksStep 1:** Click on Start.

**Step 2:** In the Windows Run command, type “python idle”.



Fig 2.6

**Step 3:** Click on IDLE (Python 3.7 64-bit) and launch the program.

**Step 4:** To go ahead with working in IDLE you must first save the file. Click on File > Click

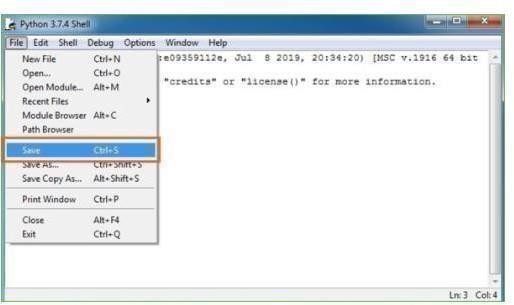
on save .

Fig 2.7

**Step 5:** Name the file and save it as the type that should be a Python files. Click on SAVE. Here I have named the files as Hey World. Step 6: Now e.g., enter print.

## Features

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

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

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

**Databases:** Python provides interfaces to all major commercial databases

**GUI Programming:** Python supports GUI applications that can be created andported to many system calls, libraries and windows systems, such as WindowsMFC, Macintosh, and the X Window system of Unix.

**Scalable:** Python provides a better structure and support for large programsthan shell scripting.

#### Module

A module allows you to logically organize your Python code. Grouping related code into a module makes the code easier to understand and use. A module is a Python object with arbitrarily named attributes that you can bind and reference. Simply, a module is a file consisting of Python code. A module can define functions, classes and variables. A module can also include runnable code.

#### Example:

The python code for a module named a name normally resides in a file named aname.py. Here's an example of a simple module, support.py

def print\_func( par ): print "Hello : ", par return

#### Exception

An exception is an event, which occurs during the execution of a program that disrupts the normal flow of the program's instructions. In general, when a Python script encounters a situation that it cannot cope with, it raises an exception. An exception is a Python object that represents an error.

When a Python script raises an exception, it must either handle the exception immediately otherwise it terminates and quits.

#### Handling an exception

If you have some suspicious code that may raise an exception, you can defend your program by placing the suspicious code in a try: block. After the try: block, include an except: statement, followed by a block of code which handles the problem as elegantly as possible.

The Python standard for database interfaces is the Python DB-API. Most Python database interfaces adhere to this standard.

You can choose the right database for your application. Python Database API supports a wide range of database servers such as

* GadFly
* mSQL
* MySQL
* PostgreSQL
* Microsoft SQL Server 2000
* Informix
* Inter-base
* Oracle
* Sybas

The DB API provides a minimal standard for working with databases using Python structures and syntax wherever possible. This API includes the following:

Importing the API module.

Acquiring a connection with the database. Issuing SQL statements and stored procedures. Closing the connection.

#### Class Library

Normally, a library is a collection of books or a room or place where many books are stored to be used later. Similarly, in the programming world, a library is a collection of precompiled codes that can be used later on in a program for some specific well-defined operations. Other than precompiled codes, a library may contain documentation, configuration data, message templates, classes, values, etc.

A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and more convenient for the programmer. As we don’t need to write the same code again and again for different programs. Python libraries play a very vital role in the fields of Machine Learning, Data Science, Data Visualization, etc.

**Matplotlib:** This library is responsible for plotting numerical data. And that’s why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.

**Pandas:** Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools.

It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.

**Numpy:** The name “Numpy” stands for “Numerical Python”. It is the commonly used library. It is a popular machine-learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. The Array Interface is one of the key features of this library.

#### Versions

Some previous versions of the documentation remain available online. Use the list below to select a version to view.

For unreleased (in development) documentation, see In Development Versions. Python 3.11.0, documentation was released on 24 October 2022.

Python 3.10.7, documentation released on 6 September 2022.

Python 3.10.6, documentation released on 8 August 2022.

Python 3.10.5, documentation released on 6 June 2022.

Python 3.10.4, documentation released on 24 March 2022.

Python 3.10.3, documentation released on 16 March 2022.

Python 3.10.2, documentation released on 14 January 2022.

Python 3.10.1, documentation released on 6 December 2021.



Fig 2.8

## Advantages

Python's features include:

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

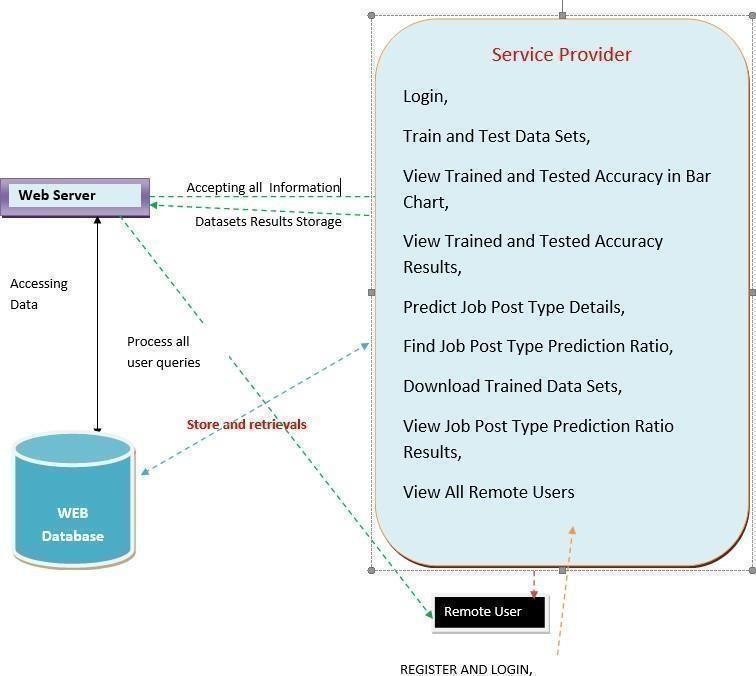
**Easy-to-maintain:** Python's source code is fairly easy to maintain.

# CHAPTER 5

# SOFTWARE DESIGN

## ARCHITECTURE DIAGRAM

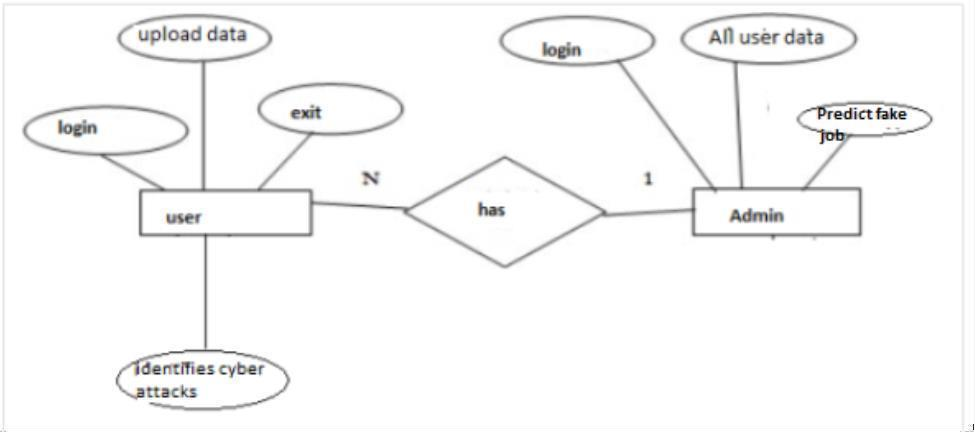
An architectural diagram is a visual representation that maps out the physical implementation of components of a software system. It shows the general structure of the software system and the associations, limitations, and boundaries between each element.



**Fig 3 Architecture Diagram**

## ENTITY RELATIONSHIP DIAGRAM

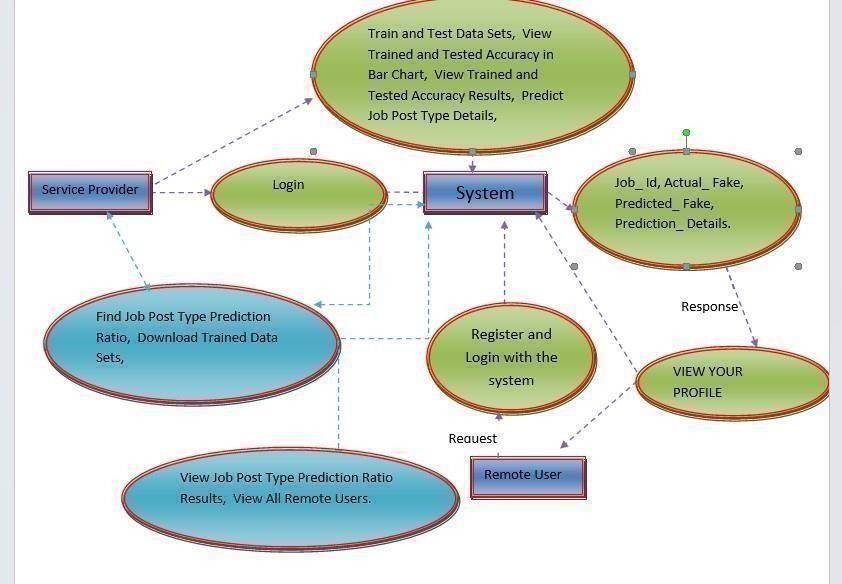
ER Diagram stands for Entity Relationship Diagram, also known as ERD is a diagram that displays the relationship of entity sets stored in a database.



**Fig 4 Entity Relationship Model**

# DATA FLOW DIAGRAM

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.



**Fig 5 Data Flow Diagram**

# UML DIAGRAMS

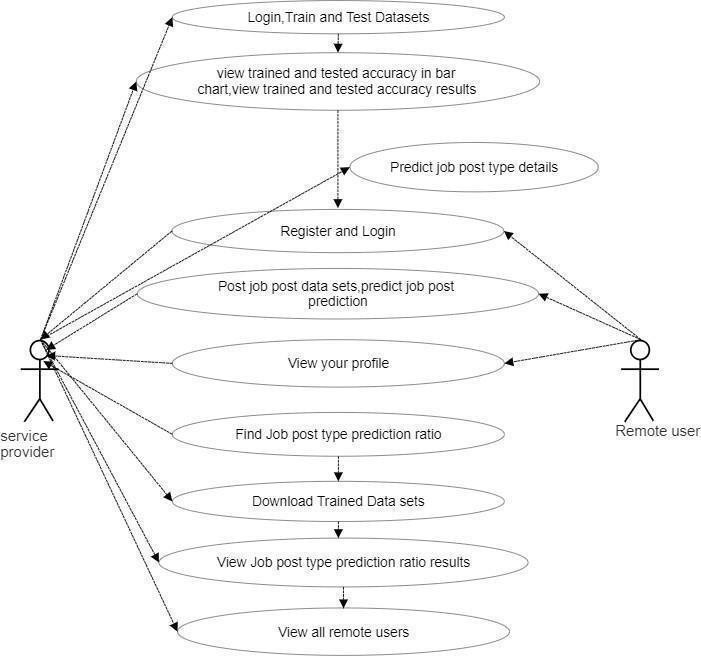
* UML represents Unified Modeling Language. UML is an institutionalized universally useful showing dialect in the subject of article-situated programming designing. The fashionable is overseen, and made by way of, the Object Management Group.
* The goal is for UML to become a regular dialect for making fashions of item- arranged PC programming. In its gift frame, UML has contained two noteworthy components: a Metashow and documentation. Later on, a few types of methods or systems can also likewise be brought to; or related to, UML.
* The Unified Modeling Language is a popular dialect for indicating, Visualization, Constructing, and archiving the curios of programming frameworks, and for business demonstrating and different non-programming frameworks.
* The UML speaks to an accumulation of first-rate building practices that have been verified fruitful in the showing of full-size and complicated frameworks.
* The UML is an essential piece of creating gadgets located in programming and the product development method. The UML makes use of commonly graphical documentation to specify the plan of programming ventures.

## GOALS

* The Primary goals inside the plan of the UML are as in step with the subsequent:
* Provide clients with a prepared-to-utilize, expressive visual showing Language on the way to create and change massive models.
* Provide extensibility and specialization units to make bigger the middle ideas.
* Be free of specific programming dialects and advancement manner.

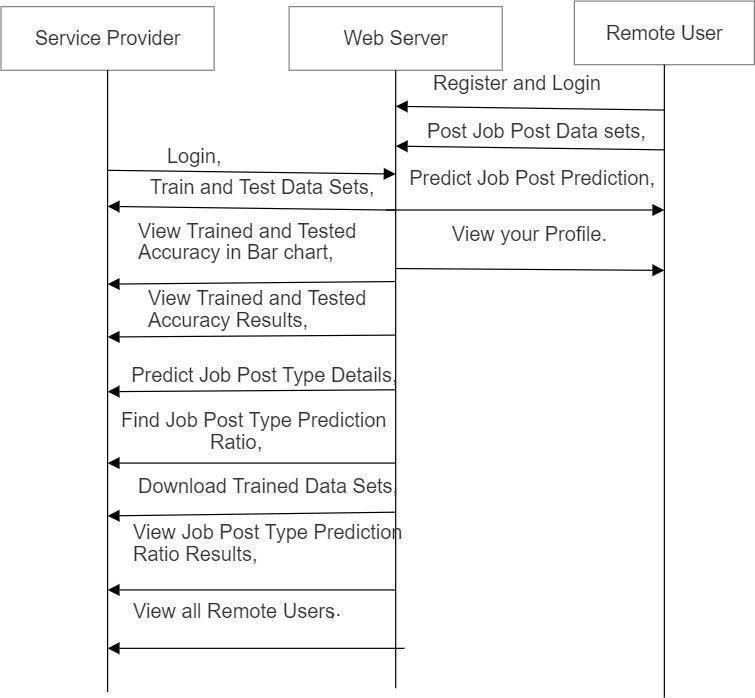
## USE CASE

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. The roles of the actors in the system can be depicted.



**Fig 6 Use Case Diagram**

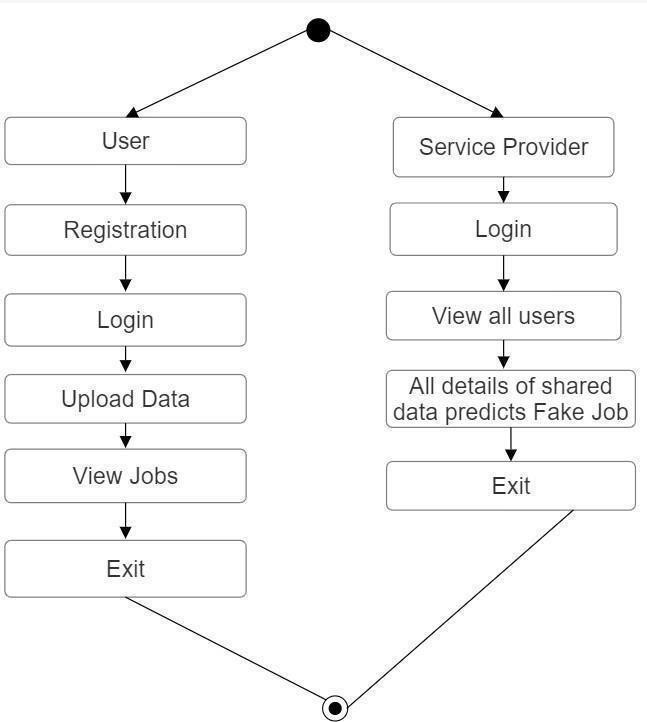
## SEQUENCE



**Fig 6.1 Sequence Diagram**

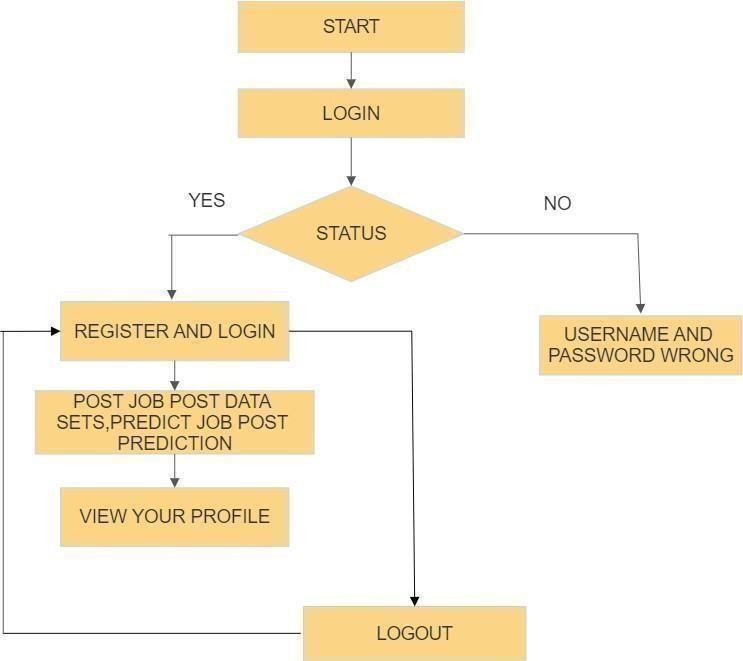
## ACTIVITY

Activity diagram is another important behavioural diagram in [UML](https://en.wikipedia.org/wiki/Unified_Modeling_Language) diagram to describe dynamic aspects of the system. Activity diagram is essentially an advanced version of flow chart that modeling the flow from one activity to another activity

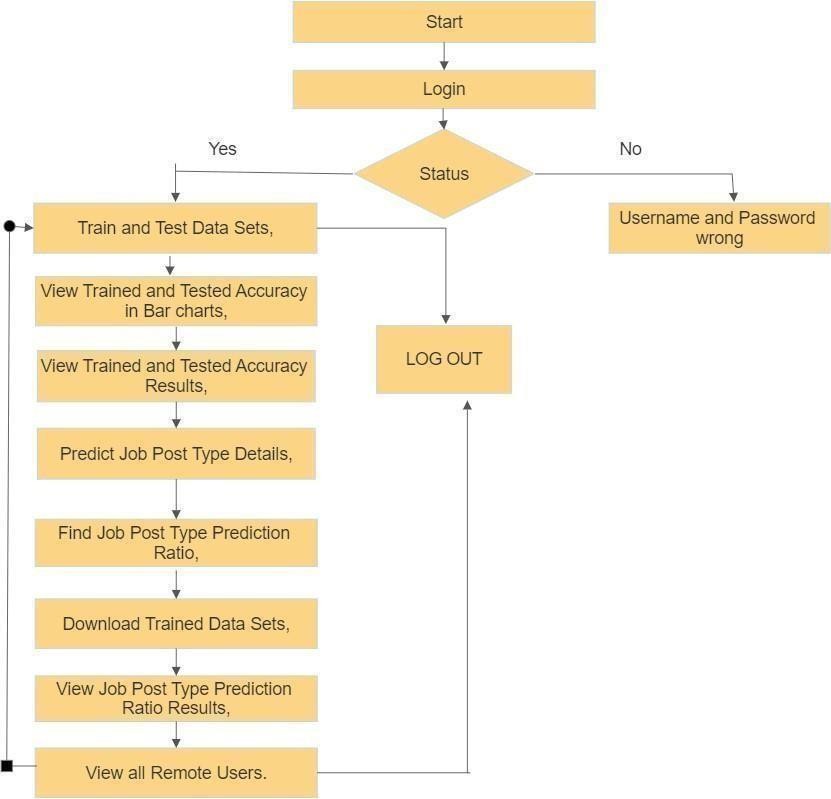


**Fig 6.2 Activity Diagram**

# FLOW CHART



#### Fig 6.3 Flow chart diagram of the remote user



**Fig 6.4 Flow Chart Diagram of Service Provider**

# CHAPTER 6

# CODING AND IMPLEMENTATION

## PROJECT METHODOLOGY

1. **Data Collection and Preparation:** Identify suitable datasets for your study. These datasets should consist of both genuine and fake job posts. You can collect data from various online job platforms or utilize publicly available datasets. Preprocess the data by cleaning and transforming it as per your analysis requirements.
2. **Feature Selection and Extraction:** Analyze the collected data to identify relevant features that can help in distinguishing between genuine and fake job posts. Extract these features from the dataset and create a feature set for further analysis.
3. **Algorithm Selection:** Identify different data mining techniques that can be used for fake job post prediction. Some potential techniques include logistic regression, SVM, naïve bayes, random forest classifier. Research and select appropriate algorithms based on their performance, suitability for the problem, and availability of implementations
4. **Experimental Design:** Design an experimental framework to evaluate and compare the performance of different data mining techniques. Divide your dataset into training and testing sets using appropriate sampling techniques to ensure unbiased evaluation.
5. **Prediction:** Implement the prediction functionality using the trained ensemble model. Allow users to input details about the job id, and job description details.
6. **Evaluation:** Implement and train the selected data mining algorithms on the training dataset. Evaluate their performance using appropriate evaluation metrics such as accuracy, precision, recall, F1 score, and area under the receiver operating characteristic curve. Compare the results of different techniques to identify the most effective approach.
7. **Statistical Analysis:** Perform statistical analysis to validate the significance of the results obtained. Use appropriate statistical tests to compare the performance of different techniques.
8. **Deployment:** Deploy the trained ensemble model in a suitable environment, such as a web-based dashboard or application, to make it accessible to users.

Ensure the deployment environment can handle real-time or near-real-time predictions based on input data related to disruptions.

## SAMPLE CODE

#### REMOTE USER

from django.db.models import Count from django.db.models import Q

from django.shortcuts import render, redirect, get\_object\_or\_404 import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report from sklearn.metrics import accuracy\_score

from sklearn.tree import DecisionTreeClassifier from sklearn.ensemble import VotingClassifier # Create your views here.

from Remote\_User.models import ClientRegister\_Model,attack\_prediction,detection\_ratio,detection\_accuracy

def login(request):

if request.method == "POST" and 'submit1' in request.POST: username = request.POST.get('username')

password = request.POST.get('password') try:

enter =

ClientRegister\_Model.objects.get(username=username,password=password) request.session["userid"] = enter.id

return redirect('ViewYourProfile') except:

pass

return render(request,'RUser/login.html')

def index(request):

return render(request, 'RUser/index.html') def Add\_DataSet\_Details(request):

return render(request, 'RUser/Add\_DataSet\_Details.html', {"excel\_data": ''}) def Register1(request):

if request.method == "POST":

username = request.POST.get('username') email = request.POST.get('email') password = request.POST.get('password') phoneno = request.POST.get('phoneno') country = request.POST.get('country') state = request.POST.get('state')

city = request.POST.get('city') address = request.POST.get('address') gender = request.POST.get('gender')

ClientRegister\_Model.objects.create(username=username, email=email, password=password, phoneno=phoneno,

country=country, state=state,

city=city,address=address,gender=gender)

obj = "Registered Successfully"

return render(request, 'RUser/Register1.html',{'object':obj}) else:

return render(request,'RUser/Register1.html')

def ViewYourProfile(request): userid = request.session['userid']

obj = ClientRegister\_Model.objects.get(id= userid)

return render(request,'RUser/ViewYourProfile.html',{'object':obj}) def Predict\_Web\_Spoofing\_Attack\_Type(request):

if request.method == "POST":

review = request.POST.get('keyword') if request.method == "POST":

url= request.POST.get('url') df = pd.read\_csv('Datasets.csv') def apply\_response(Label):

if (Label == 0):

return 0 # No Attack elif (Label == 1):

return 1 # Phishing Attack

df['Label'] = df['Label'].apply(apply\_response) cv = CountVectorizer()

X = df['URL']

y = df['Label']

print("RID") print(X) print("Results") print(y

cv = CountVectorizer() X = cv.fit\_transform(X) models = []

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20)

X\_train.shape, X\_test.shape, y\_train.shape print("Naive Bayes")

from sklearn.naive\_bayes import MultinomialNB NB = MultinomialNB()

NB.fit(X\_train, y\_train) predict\_nb = NB.predict(X\_test)

naivebayes = accuracy\_score(y\_test, predict\_nb) \* 100 print("ACCURACY")

print(naivebayes) print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_nb)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_nb)) models.append(('naive\_bayes', NB))

# SVM Model print("SVM")

from sklearn import svm lin\_clf = svm.LinearSVC() lin\_clf.fit(X\_train, y\_train)

predict\_svm = lin\_clf.predict(X\_test)

svm\_acc = accuracy\_score(y\_test, predict\_svm) \* 100 print("ACCURACY")

print(svm\_acc) print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_svm)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_svm)) models.append(('svm', lin\_clf)) print("Logistic Regression")

from sklearn.linear\_model import LogisticRegression

reg = LogisticRegression(random\_state=0, solver='lbfgs').fit(X\_train, y\_train) y\_pred = reg.predict(X\_test)

print("ACCURACY")

print(accuracy\_score(y\_test, y\_pred) \* 100)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, y\_pred)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, y\_pred)) models.append(('logistic', reg)) print("Decision Tree Classifier")

dtc = DecisionTreeClassifier() dtc.fit(X\_train, y\_train) dtcpredict = dtc.predict(X\_test) print("ACCURACY")

print(accuracy\_score(y\_test, dtcpredict) \* 100) print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, dtcpredict)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, dtcpredict)) models.append(('DecisionTreeClassifier', dtc))

classifier = VotingClassifier(models) classifier.fit(X\_train, y\_train) y\_pred = classifier.predict(X\_test) url1 = [url]

vector1 = cv.transform(url1).toarray() predict\_text = classifier.predict(vector1)

pred = str(predict\_text).replace("[", "") pred1 = pred.replace("]", "") prediction = int(pred1)

if (prediction == 0):

val = 'Genuine Job URL Detected' elif (prediction == 1):

val = 'Fake Job Url Detected' print(val)

print(pred1)

attack\_prediction.objects.create(url=url,Prediction=val)

return render(request, 'RUser/Predict\_Web\_Spoofing\_Attack\_Type.html',{'objs': val})

return render(request, 'RUser/Predict\_Web\_Spoofing\_Attack\_Type.html')

#### Service Provider

from django.db.models import Count, Avg from django.shortcuts import render, redirect from django.db.models import Count

from django.db.models import Q import datetime

import xlwt

from django.http import HttpResponse import pandas as pd

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.metrics import accuracy\_score, confusion\_matrix, classification\_report from sklearn.metrics import accuracy\_score

from sklearn.tree import DecisionTreeClassifier

# Create your views here.

from Remote\_User.models import ClientRegister\_Model,attack\_prediction,detection\_ratio,detection\_accuracy

def serviceproviderlogin(request): if request.method == "POST":

admin = request.POST.get('username') password = request.POST.get('password')

if admin == "Admin" and password =="Admin": detection\_accuracy.objects.all().delete() return redirect('View\_Remote\_Users')

return render(request,'SProvider/serviceproviderlogin.html')

def View\_Web\_Spoofing\_Attack\_Status\_Ratio(request): detection\_ratio.objects.all().delete()

ratio = ""

kword = 'No Phishing or Web Spoofing Attack Found' print(kword)

obj = attack\_prediction.objects.all().filter(Q(Prediction=kword)) obj1 = attack\_prediction.objects.all()

count = obj.count(); count1 = obj1.count();

ratio = (count / count1) \* 100 if ratio != 0:

detection\_ratio.objects.create(names=kword, ratio=ratio) ratio12 = ""

kword12 = 'Phishing or Web Spoofing Attack Found' print(kword12)

obj12 = attack\_prediction.objects.all().filter(Q(Prediction=kword12)) obj112 = attack\_prediction.objects.all()

count12 = obj12.count();

count112 = obj112.count();

ratio12 = (count12 / count112) \* 100 if ratio12 != 0:

detection\_ratio.objects.create(names=kword12, ratio=ratio12) obj = detection\_ratio.objects.all()

return render(request, 'SProvider/View\_Web\_Spoofing\_Attack\_Status\_Ratio.html',

{'objs': obj})

def View\_Remote\_Users(request): obj=ClientRegister\_Model.objects.all()

return render(request,'SProvider/View\_Remote\_Users.html',{'objects':obj}) def charts(request,chart\_type):

chart1 = detection\_ratio.objects.values('names').annotate(dcount=Avg('ratio'))

return render(request,"SProvider/charts.html", {'form':chart1, 'chart\_type':chart\_type})

def charts1(request,chart\_type):

chart1 = detection\_accuracy.objects.values('names').annotate(dcount=Avg('ratio')) return render(request,"SProvider/charts1.html", {'form':chart1,

'chart\_type':chart\_type})

def View\_Prediction\_Of\_Web\_Spoofing\_Attack\_Status(request): obj =attack\_prediction.objects.all()

return render(request,

'SProvider/View\_Prediction\_Of\_Web\_Spoofing\_Attack\_Status.html',

{'list\_objects': obj})

def likeschart(request,like\_chart):

charts =detection\_accuracy.objects.values('names').annotate(dcount=Avg('ratio')) return render(request,"SProvider/likeschart.html", {'form':charts,

'like\_chart':like\_chart})

def Download\_Trained\_DataSets(request):

response = HttpResponse(content\_type='application/ms-excel') # decide file name

response['Content-Disposition'] = 'attachment; filename="Predicted\_Datasets.xls"' # creating workbook

wb = xlwt.Workbook(encoding='utf-8') # adding sheet

ws = wb.add\_sheet("sheet1") # Sheet header, first row row\_num = 0

font\_style = xlwt.XFStyle() # headers are bold font\_style.font.bold = True

# writer = csv.writer(response)

obj = attack\_prediction.objects.all()

data = obj # dummy method to fetch data. for my\_row in data:

row\_num = row\_num + 1

ws.write(row\_num, 0, my\_row.url, font\_style) ws.write(row\_num, 1, my\_row.Prediction, font\_style)

wb.save(response) return response

def train\_model(request): detection\_accuracy.objects.all().delete() df = pd.read\_csv('Datasets.csv')

def apply\_response(Label): if (Label == 0):

return 0 # No Attack elif (Label == 1):

return 1 # Web Attack

df['Label'] = df['Label'].apply(apply\_response) cv = CountVectorizer()

X = df['URL']

y = df['Label']

print("URL") print(X) print("Label") print(y)

cv = CountVectorizer() X = cv.fit\_transform(X)

models = []

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.20)

X\_train.shape, X\_test.shape, y\_train.shape print(X\_test)

print("Naive Bayes")

from sklearn.naive\_bayes import MultinomialNB NB = MultinomialNB()

NB.fit(X\_train, y\_train) predict\_nb = NB.predict(X\_test)

naivebayes = accuracy\_score(y\_test, predict\_nb) \* 100 print(naivebayes)

print(confusion\_matrix(y\_test, predict\_nb)) print(classification\_report(y\_test, predict\_nb)) models.append(('naive\_bayes', NB)) detection\_accuracy.objects.create(names="Naive Bayes", ratio=naivebayes)

# SVM Model print("SVM")

from sklearn import svm lin\_clf = svm.LinearSVC() lin\_clf.fit(X\_train, y\_train)

predict\_svm = lin\_clf.predict(X\_test)

svm\_acc = accuracy\_score(y\_test, predict\_svm) \* 100 print(svm\_acc)

print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, predict\_svm)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, predict\_svm)) models.append(('svm', lin\_clf)) detection\_accuracy.objects.create(names="SVM", ratio=svm\_acc) print("Logistic Regression")

from sklearn.linear\_model import LogisticRegression

reg = LogisticRegression(random\_state=0, solver='lbfgs').fit(X\_train, y\_train) y\_pred = reg.predict(X\_test)

print("ACCURACY")

print(accuracy\_score(y\_test, y\_pred) \* 100) print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, y\_pred)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, y\_pred)) models.append(('logistic', reg))

detection\_accuracy.objects.create(names="Logistic Regression", ratio=accuracy\_score(y\_test, y\_pred) \* 100)

print("Decision Tree Classifier") dtc = DecisionTreeClassifier() dtc.fit(X\_train, y\_train) dtcpredict = dtc.predict(X\_test) print("ACCURACY")

print(accuracy\_score(y\_test, dtcpredict) \* 100) print("CLASSIFICATION REPORT")

print(classification\_report(y\_test, dtcpredict)) print("CONFUSION MATRIX")

print(confusion\_matrix(y\_test, dtcpredict)) models.append(('DecisionTreeClassifier', dtc))

detection\_accuracy.objects.create(names="Decision Tree Classifier", ratio=accuracy\_score(y\_test, dtcpredict) \* 100)

csv\_format = 'Results.csv' df.to\_csv(csv\_format, index=False) df.to\_markdown

obj = detection\_accuracy.objects.all()

return render(request,'SProvider/train\_model.html', {'objs': obj})

# CHAPTER 7

**SYSTEM TESTING**

# TESTING STATERGIES

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

## 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 run as one program. Testing is event driven and is more concerned with thebasic 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 theproblems that arise from the combination of components.

#### Functional testing

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 centred on the following items:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Valid Input | : identified classes of valid input must be accepted. | |  |  |
| Invalid Input | : identified classes of invalid input must be rejected. | |  |  |
| Functions | : identified functions must be exercised. |  |  |  |
| Output | : identified classes of application outputs | must | be | exercised. |

Systems/Procedures: interfacing systems or procedures must be invoked.

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

#### System Test

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

#### White Box Testing

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

#### Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner working structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

#### Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

* + strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

* + objectives
  + field entries must work properly.
  + ages must be activated from the identified link.
  + entry screen, messages and responses must not be delayed.
  + features to be tested
  + Verify that the entries are of the correct format
  + duplicate entries should be allowed
  + links should take the user to the correct page.

#### 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.

#### 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.

# SAMPLE TEST CASES

**Login details**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S No** | **Test Case** | **Input** | **Expected**  **output** | **Actual**  **Output** | **Result** |
| 1 | Registration | Personal Information | Creates a User | Create a User | Pass |
| 2 | Login | Username,  Pssword | User page | User page | Pass |
| 3 | Login | Invalid Username,  Password | Login page | Login page | Pass |
| 4 | User Upload Dataset | Choose a Dataset | Dataset Successfully  Selected and Diaplayed | Dataset Successfully  Selected and Diaplayed | Pass |
| 5 | Predict that job  is fake or real | Enter details | Predict job is  fake or real | Predict the job  as fake or real | Pass |

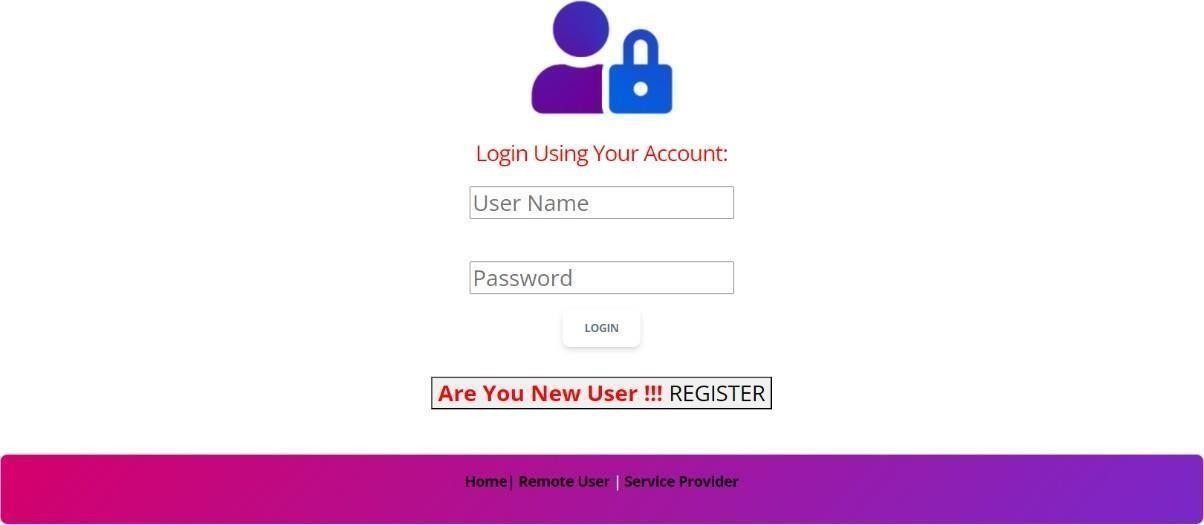
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 6 | Admin/Service Provider Login | Enter name and Password | Service Provider Page | Service Provider page | Pass |
| 7 | View Trained Datasets | Dataset uploaded  By registered users | Display  Trained and Tested Result | Display Trained and Tested Result | Pass |
| 8 | Result Of models | Display Accuracy of our models | Display Accuracy of Our models | Display accuracy of  Our models | Pass |
| 9 | Predictin g jobpost  typ e details | Displaying job post type  Prediction details | Displaying job post type Prediction details | Displaying job post type Prediction details | Pass |
| 10 | Find Job post type predictio n ratio | Displaying Job Post type Prediction ration | Displaying Job Post type Prediction ratio | Displaying Job Post type Prediction ratio | Pass |

# CHAPTER 8 OUTPUT SCREENS

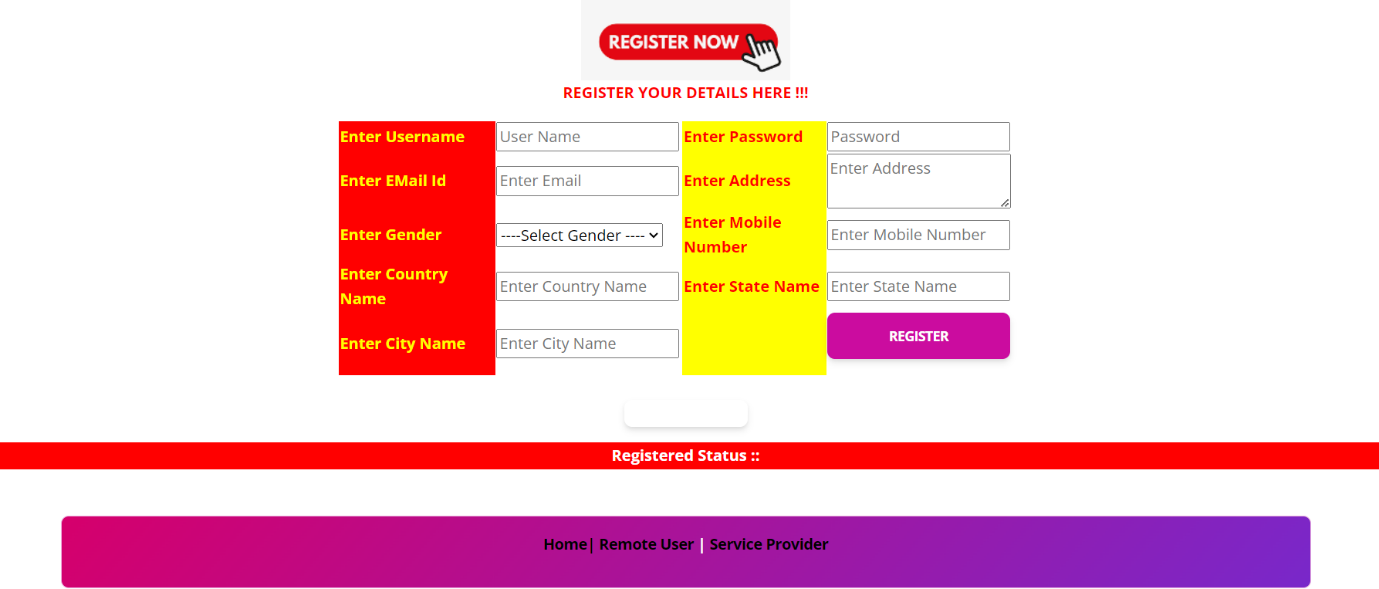




#### Fig 7 Fake Job Prediction Interface



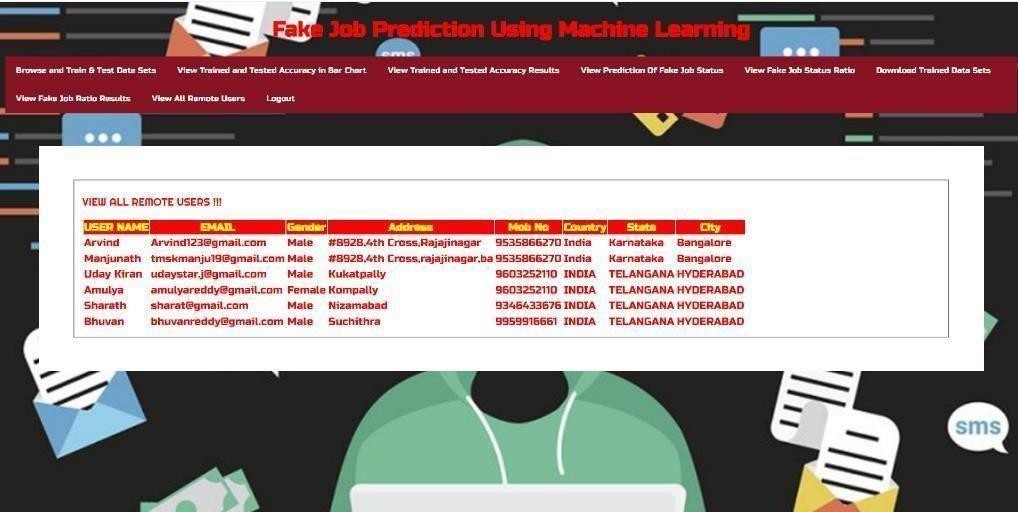
**Fig 7.1 User Login**



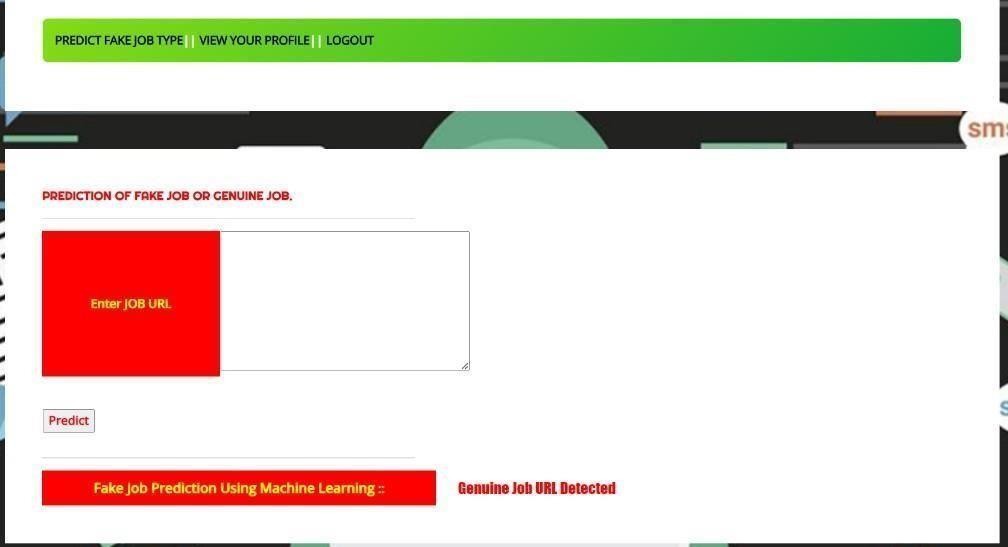
#### Fig 7.2 User Registration



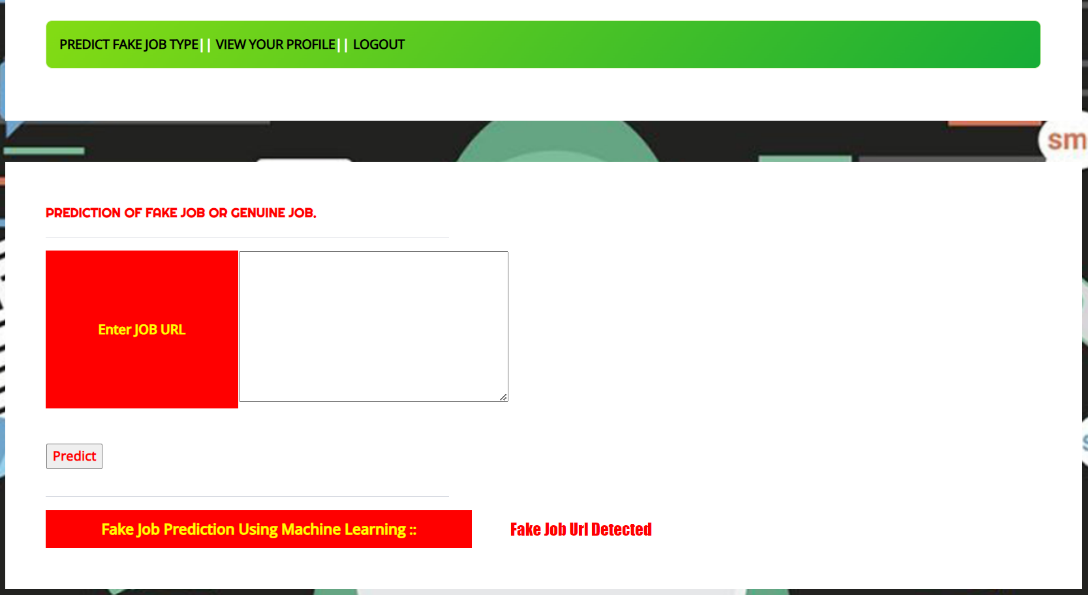
**Fig 7.3 Service Provider Login**



#### Fig 7.4 All remote users' information



**Fig 7.5 After Job Post Detection predicted as Genuine**



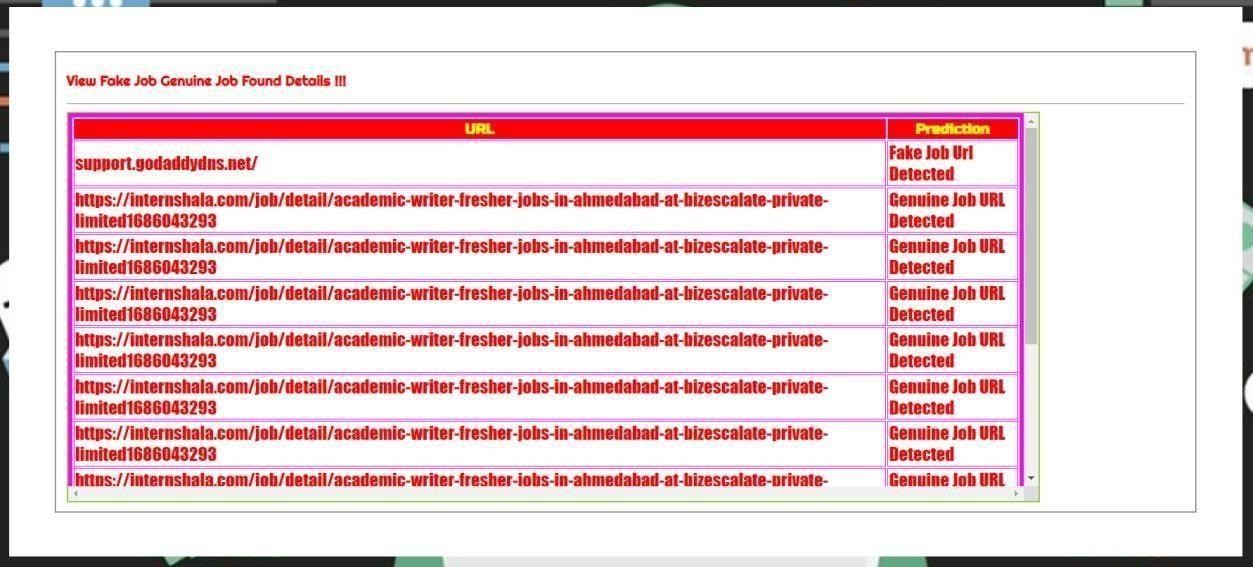
#### Fig 7.6 After Job Post Detection Predicted as Fake



**Fig 7.7 Data sets trained and tested results**



#### Fig 7.8 Graphical representation of true and fake jobs



**Fig 7.9 Details of fake and genuine job posts**

# CHAPTER 9

1. **CONCLUSION**

# CONCLUSION

Job scam detection has become a great concern all over the world at present. In this paper, we have analyzed the impacts of job scam which can be a very prosperous area in research filed creating a lot of challenges to detect fraudulent job posts. We have experimented with EMSCAD dataset which contains real life fake job posts. In this paper we have experimented both machine learning algorithms (SVM, Naive Bayes, Random Forest and logistic classifier) and deep learning model (Deep Neural Network). This work shows a comparative study on the evaluation of traditional machine learning and deep learning based classifiers. We have found highest classification accuracy for logistic regression among traditional machine learning algorithms and 98 % accuracy.

# CHAPTER 10

**FUTURE ENHANCEMENTS**

There are several potential future enhancements for the project "A Comparative study on fake job post prediction using different datamining techniques." Here are a few ideas:

1. Expand the dataset: The study could be enhanced by including more data sources to increase the size and diversity of the dataset. This could involve scraping additional job postings from different job boards, and also including data from social media and other online platforms where fake job postings may be shared.
2. Incorporate more advanced machine learning techniques: While the study already compares different data mining techniques, incorporating more advanced machine learning algorithms, such as deep learning, could yield even better results.
3. Consider additional features: The current study uses a variety of features, such as job title and company name, to predict fake job postings. However, there may be other features that could be useful, such as the language used in the posting, the salary offered, or the location of the job.
4. Evaluate the impact of different algorithms on different types of job postings: It could be useful to evaluate how different algorithms perform on different types of job postings, such as those in different industries or with different levels of seniority.
5. Apply the model to real-world scenarios: Once the model is developed, it could be applied to real-world scenarios to test its efficacy. This could involve partnering with companies or job boards to help identify and remove fake job post.

# CHAPTER 11

**REFERENCES**

1. S. Vidros , C. Kolias , G. Kambourakis ,and L. Akoglu, “Automatic Detection of Online Recruitment Frauds: Characteristics, Methods, and a Public Dataset”, FutureInternet 2017.
2. Alghamdi, F. Alharby, “An Intelligent Model for Online Recruitment Fraud Detection”, Journal of Information Security, 2019 .
3. Tin Van Huynh1, Kiet Van Nguyen, Ngan Luu-Thuy Nguyen1, and Anh Gia-Tuan Nguyen, “Job

Prediction: From Deep Neural Network Models to Applications”, RIVF International Conference on Computing and Communication Technologies (RIVF), 2020.

1. Jiawei Zhang, Bowen Dong, Philip S. Yu, “FAKEDETECTOR: Effective Fake NewsDetection with Deep Diffusive Neural Network”, IEEE 36th International Conferenceon Data Engineering (ICDE), 2020.
2. Scanlon, J.R. and Gerber, M.S., “Automatic Detection of Cyber Recruitment byViolent Extremists”, Security Informatics, 3, 5, 2014.
3. Y. Kim, “Convolutional neural networks for sentence classification,” arXivPrepr. arXiv1408.5882, 2014.
4. T. Van Huynh, V. D. Nguyen, K. Van Nguyen, N. L.-T. Nguyen, and A.G.- T.Nguyen, “Hate Speech Detection on Vietnamese Social Media Text 2019.
5. P. Wang, B. Xu, J. Xu, G. Tian, C.-L. Liu, and H. Hao, “Semantic expansion using word embedding clustering and convolutional neural network for improving short textclassification,” Neurocomputing.
6. Li, G. Zhan, and Z. Li, “News Text Classification Based on Improved BiLSTM- CNN,” in 2018

9th International Conference on Information Technology in Medicine and Education (ITME), 2018, pp. 890-893.

1. K. R. Remya and J. S. Ramya, “Using weighted majority voting classifier combination for relation classification in biomedical texts,” International Conference

on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), 2014, pp. 1205-1209.

1. Yasin, A. and Abuhasan, A. (2016) An Intelligent Classification Model for Phishing Email Detection. International Journal of Network Security& Its Applications, 8, 55- 72. https://doi.org/10.5121/imsa.2016.8405
2. Li, H.; Chen, Z.; Liu, B.; Wei, X.; Shao, J. Spotting fake reviews via collective positive-
3. unlabeled learning. In Proceedings of the 2014 IEEE International Conference on Data Mining (ICDM), Shenzhen, China.
4. Ott, M.; Cardie, C.; Hancock, J. Estimating the prevalence of deception in online review communities. InProceedings of the 21st international conference on World Wide Web, Lyon, France.
5. Nizamani, S., Memon, N., Glasdam, M. and Nguyen, D.D. (2014) Detection of Fraudulent Emails by Employing Advanced Feature Abundance

# TEXT BOOKS

* Machine Learning: A Probabilistic Perspective by Kevin P. Murphy
* The Elements of Statistical Learning: Data Mining, Inference, and Prediction by Trevor Hastie, Robert Tibshirani, and Jerome Friedman.
* Practical Predictive Analytics: Using Python by Vishal Chawla, Shruti Saxena, and Pronab Sen.
* Large-Scale Machine Learning with Python by Bastiaan Sjardin, Luis Pedro Coelho, Vaibhav Saxena, and John W. Fisher III
* High-Dimensional Data Mining: Problems and Solutions edited by Liqing Zhang and Anil Jain

## Websites

1. https:/[/www.w3schools.com/python/](http://www.w3schools.com/python/)
2. https:/[/www.ge](http://www.geeksforgeeks.org/machine-learning/)e[ksforgeeks.org/machine-learning/](http://www.geeksforgeeks.org/machine-learning/)
3. https:/[/www.tutorialspoi](http://www.tutorialspoint.com/sql/index.htm)n[t.com/sql/index.htm](http://www.tutorialspoint.com/sql/index.htm)
4. https:/[/www.ge](http://www.geeksforgeeks.org/css)e[ksforgeeks.org/css](http://www.geeksforgeeks.org/css)
5. https:/[/www.ge](http://www.geeksforgeeks.org/javascript/)e[ksforgeeks.org/javascript/](http://www.geeksforgeeks.org/javascript/)