A PROJECT REPORT ON

# "Comparative Analysis of Feature Selection Algorithms for Computational Personality Prediction from Social Media"

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

**BACHELOR OF ENGINEERING**
**IN**
**COMPUTER ENGINEERING**
**BY,**

| | |
|---|---|
| MR.Bhushan Nishane | ( B151044300 ) |
| MR.Suhas Dhole | ( B151044242 ) |
| MR.Atharv Sahare | ( B151044325 ) |
| MS.Aishwarya Kale | ( B151044206 ) |

**Under The Guidance of**
**Prof.P.D.HALLE**

**DEPARTMENT OF COMPUTER ENGINEERING**

**SKN Sinhgad Institute Of Technology and Science**

**Kusgaon(BK),Lonavala PUNE 410401**

**SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE 2021 - 22**

SINHGAD TECHNICAL EDUCATION SOCIETY, PUNE

तमसो मा ज्योतिर्गमय

Sinhgad Institutes

# CERTIFICATE

This is to certify that the Project Entitled

**"Comparative Analysis of Feature Selection Algorithms for Computational Personality Prediction from Social Media"**

Submitted by

| | |
|---|---|
| MR.Bhushan Nishane | (B151044300) |
| MR.Suhas Dhole | (B151044242) |
| MR.Atharv Sahare | (B151044325) |
| MS.Aishwarya Kale | (B151044206) |

is a bonafide work carried out by them under the supervision of **Prof. P. D. Halle** and it is approved for the partial fulfillment of the requirement of Savtribai Phule Pune university, Pune for the award of the degree of Bachelor of Engineering (Computer Engineering).

**Prof. P.D.HALLE**
Internal Guide
Dept. of Computer Engineering.

**Prof. G.M.KADAM**
Head Of Department
Dept.Computer Engineering

External Examiner

**Dr. M. S. Rohokale**
Principal
SKNSITS, LONAVALA

Place:
Date:

## PROJECT APPROVAL SHEET

A Project Report Titled as

**"Comparative Analysis of Feature Selection Algorithms for Computational Personality Prediction From Social Media"**

Is verified for its originality in documentation, problem statement, proposed work and implementation successfully completed

By

| | |
|---|---|
| MR.Bhushan Nishane | (B151044300) |
| MR.Suhas Dhole | (B151044242) |
| MR.Atharv Sahare | (B151044325) |
| MS.Aishwarya Kale | (B151044206) |

At

DEPARTMENT OF COMPUTER ENGINEERING

SKN Sinhgad Institute of Technology & Science (SKNSITS), Lonavala,

SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE

ACADEMIC YEAR 2021-2022

| | |
|---|---|
| Prof. P.D.HALLE | Prof. G.M.Kadam |
| Guide | H.O.D |
| Dept.of Computer Engg. | Dept.of Computer Engg. |

## Acknowledgments

It gives us great pleasure in presenting the preliminary project report on **'Comparative Analysis of Feature Selection Algorithms for Computational Personality Prediction From Social Media'.**

I would like to take this opportunity to thank my internal guide **Prof. P.D.HALLE** for giving me all the help and guidance I needed. I am really grateful to them for their kind support. Their valuable suggestions were very helpful.

I am also grateful to **Prof. G. M. KADAM**, Head of Computer Engineering Department, **SKNSITS LONAVALA** for his indispensable support, suggestions.

MR. Bhushan Nishane

MR. Suhas Dhole

MR. Atharv Sahare

MS. Aishwarya kale

(B.E.Computer Engineering)

## Abstract

With the rapid growth of social media, users are getting involved in virtual socialism, generating a huge volume of textual and image contents. Considering the contents such as status updates/tweets and shared posts/retweets, liking other posts is reflecting the online behaviour of the users. Predicting personality of a user from these digital footprints has become a computationally challenging problem. In a profile-based approach, utilizing the user-generated textual contents could be useful to reflect the personality in social media. Using huge number of features of different categories, such as traditional linguistic features (character-level, word-level, structural, and so on), psycholinguistic features (emotional affects, perceptions, social relationships, and so on) or social network features (network size, betweenness, and so on) could be useful to predict personality traits from social media. According to a widely popular personality model, namely, big-five-factor model (BFFM), the five factors are openness-to experience, conscientiousness, extraversion, agreeableness, and neuroticism. Predicting personality is redefined as predicting each of these traits separately from the extracted features. Traditionally, it takes huge number of features to get better accuracy on any prediction task although applying feature selection algorithms may improve the performance of the model. In this article, we have compared the performance of five feature selection algorithms, namely the Pearson correlation coefficient (PCC), correlation-based feature subset (CFS), information gain (IG), symmetric uncertainly (SU) evaluator, and chi-squared (CHI) method. The performance is evaluated using the classic metrics, namely, precision, recall, f-measure, and accuracy as evaluation matrices.

**Keywords:** Personality, Prediction, Socialism, Footprints.

# INDEX

# LIST OF FIGURE

# CHAPTER 1

# INTRODUCTION

## 1.1 OVERVIEW

Social media platforms such as Facebook, Twitter, Google, and Instagram have gained popularity due to ease access throughout the world and user-friendly interfaces to start communicating with others within a short period of time. Each user in these social networking sites (SNSs) is considered as an entity, and each entity is connected with other entities as friends, connections, or followers. While using these SNS's, users are facilitated by many activities, such as posting statuses/tweets, sharing others' posts/retweets, liking others' posts, commenting on others' posts, chatting directly with the friends, and playing online games with the friends. It is evident that from the activities performed by users, online behaviour could be depicted. Understanding users' behaviour may help to identify personality traits. Predicting users' personalities from digital footprints of social media is a challenging task as the context of identifying personality traits in social media is not trivial. Positive or negative reviews/ opinions about a political party. These types of statuses may have contextual trends, as other friends of the users may also be involved in posting similar statuses. Considering the trend, user may post his/her political views. Users behave differently in social media and real life.

### 1.1.1 MOTIVATION

The analysis of the data revealed that almost all respondents used social media. Based on factor analysis results, their motivations for doing so are entertainment, information seeking, personal utility and convenience.

### 1.1.2 OBJECTIVE

The objective of this project is to analyse and classify personalities of a given set of people or an individual using advanced data mining concepts.

### 1.1.3 PROBLEM DEFINIATION

We have extracted over 150 features to analyse the predictive system over different types of features, such as traditional linguistic, psycholinguistic, and SN features. In the literature, many researchers have used few features to predict personality, but the overall outcome of those approaches is not quite satisfactory. We have considered several scenarios/cases of feature combinations based on psycholinguistic features to find the best subset of features to predict each personality trait differently.

# CHAPTER 2

# LITERATURE SURVEY

**Analysis Literature Survey:**

1. Social media usage has been on an ever increasing exponential rise. Usage of social media sites, such as Twitter and Facebook, for social interaction has also become a popular trend. It is estimated that on an average, around 6,000 tweets are tweeted on Twitter every second. With people spending on an average 35 minutes on Facebook each day, it is also estimated that there are about 317,000 status updates on Facebook per minute. These vast volumes of data have powerful information locked within them. This data can be analysed and several purposes. The use of such social media data for predicting user personality is common. Prediction models have been successfully built that can predict several user attributes age, gender, personality traits, occupation, political orientation etc. Standards in personality models such as the Big Five model, DISC and the Myers-Briggs Type Indicator have been the basis for all such personality prediction. A user's social media data can thus be used to predict his/her personality. The main objective of this work is to review the work carried out for personality prediction using social media data.

2. Speaker Trait Prediction for Automatic Personality Perception using Frequency Domain Linear Prediction features the aim of automatic personality perception is to predict the personality of the speaker perceived by the listener from nonverbal behaviour. Extroversion, Conscientiousness, Agreeableness, Neuroticism and Openness are the speaker traits used for personality evaluation. In this work, a speaker trait prediction approach for automatic personality assessment is proposed to model the relationship between speech signal and personality traits using frequency domain linear prediction (FDLP) technique. Among several feature extraction techniques, FDLP features render increased performance. SSPNet Speaker Personality Corpus is used for experiments and evaluation. The proposed method predicts the speaker traits with 90-99classification accuracy.

3. Prediction of Social Network Users Through weibo users, we extract social data and questionnaire, and focus on how to use the user text information to

predict their personality characteristics. We use the correlation analysis and principal component analysis to select the user information, and then use the multiple regression model, the gray prediction model and the multitasking model to predict and analyse the results. It is found that MAE values of the gray prediction are better than the multiple regression model Multitask model, the overall effect of the prediction between 0.8 and 0.9, the overall accuracy of good prediction. This shows that gray prediction in the user's personality prediction shows a good generalization and non-linear ability.

4. Traits Prediction Based on Users' Digital Footprints in Social Networks via Attention RNN Shipeng Wang, Lizhen Cui†, Lei Liu†, Xudong Lu†, Qingzhong Li With the increasing popularity of social networks, massive digital footprints of individuals in online service platforms are generated. As a result, an emerging technology namely personality trait analysis has drawn much attention. The prediction and analysis of personality trait is an efficient way to voting prediction, review analysis, decision analysis and marketing. The existing studies generally employ classification models while ignore the temporal property of digital footprints, which may lead to unsatisfactory results. To make an improvement, this paper proposes an effective method to predict the personality traits by taking the temporal factors into account through the use of Attention Recurrent Neural Network (AttRNN). The experimental results based on the dataset of 19000 Facebook volunteers suggest the proposed method is effective for predicting personality traits.

# CHAPTER 3

# PROJECT REQUIREMEN SPECIFICATION

## 3.1 EXTERNAL INTERFACE REQUIREMENT

### 3.1.1 User Interface

Application Based Personality Prediction

### 3.1.2 Hardware Interfaces:

**RAM:** 8 GB

As we are using Machine Learning Algorithm and Various High Level Libraries Laptop RAM minimum required is 8 GB.

**Hard Disk:** 40 GB

Data Set of CT Scan images is to be used hence minimum 40 GB Hard Disk memory is required.

**Processor:** Intel i5 Processor

Pycharm IDE that Integrated Development Environment is to be used and data loading should be fast hence Fast Processor is required.

**IDE:** Pycharm

Best Integrated Development Environment as it gives possible suggestions at the time of typing code snippets that makes typing feasible and fast.

Coding Language: Python Version 3.5

Highly specified Programming Language for Machine Learning because of availability of High Performance Libraries.

**Operating System:** Windows 10, Windows 11

Latest Operating System that supports all type of installation and development Environment

### 3.1.3 Software Interfaces

**Operating System:** Windows 10, Windows 11

**IDE:** Pycharm, Spyder

**Programming Language:** Python

## 3.2 NON FUNCTIONAL REQUIREMENT

### 3.2.1 Performance Requirements

The performance of the functions and every module must be well. The overall performance of the software will enable the users to work recently. Performance of encryption of data should be fast. Performance of the providing virtual environment should be fast Safety Requirement The application is designed in modules where errors can be detected and xed easily. This makes it easier to install and update new functionality if required.

### 3.2.2 Safety Requirement

The application is designed in modules where errors can be detected and fixed easily.

This makes it easier to install and update new functionality if required.

### 3.2.3 Software Quality Attributes

Our software has many quality attribute that are given below:-

**Adaptability:** This software is adaptable by all users.

**Availability:** This software is freely available to all users. The availability of the software is easy for everyone.

**Maintainability:** After the deployment of the project if any error occurs then it can be easily maintained by the software developer.

**Reliability:** The performance of the software is better which will increase the reliabilityof the Software.

**User Friendliness:** Since, the software is a GUI application; the output generated is much user friendly in its behaviour.

**Integrity:** Integrity refers to the extent to which access to software or data by unauthorized persons can be controlled.

**Security:** Users are authenticated using many security phases so reliable security is provided.

**Testability:** The software will be tested considering all the aspects.

# CHAPTER 4

# SYSTEM DESIGN

## 4.1 SYSTEM ARCHITECTURE

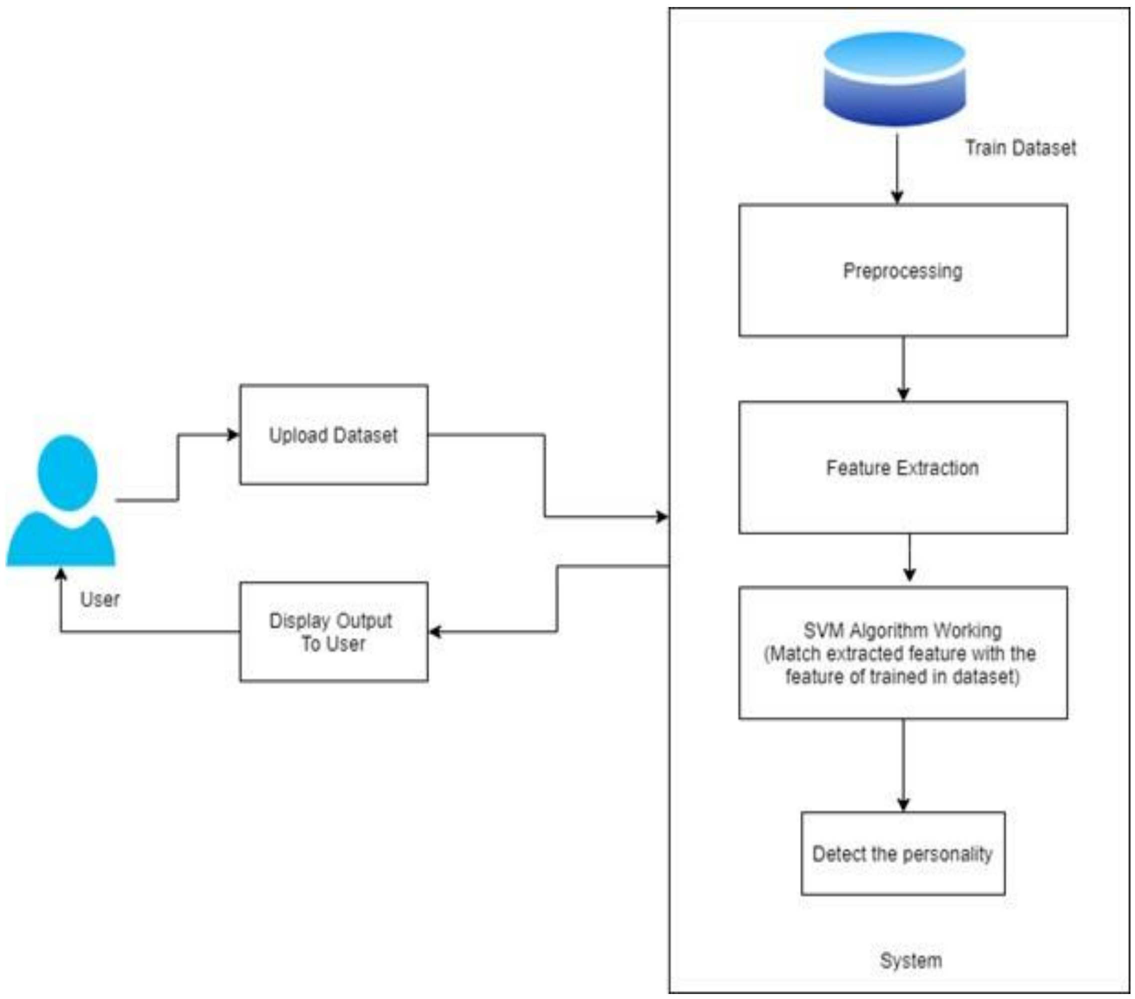


*Figure 4.1: System Architectural*

### 4.1.1 Module

**Admin**

In this module, the Admin has to log in by using valid user name and password.

After login successful he can do some operations such as View All Users and Authorize, View All E-Commerce Website and Authorize, View All Products and Reviews, View All Products Early Reviews, View All Keyword Search Details, View All Products Search Ratio, View All Keyword Search Results, View All Product Review Rank Results

**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 authorizes the users.

**View Charts Results**

View All Products Search Ratio, View All Keyword Search Results, View All Product Review Rank Results.

**Ecommerce 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 Add Products, View All Products with reviews, View All Early Product's reviews, View All Purchased Transactions.

**End 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 best or 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 Manage Account, Search Products by keyword and Purchase, View Your Search Transactions, View.

### 4.1.2 Mathematical Model:

Let S be the Whole system S= I, P, O

I-input

P-procedure

O-output

Input (I)

I=No of user, reviews, likes, dislikes, total, trusted reviews

Where,

Users =: upload +ve, -ve review,

Trusted review =: Friend circles recommended review

Procedure (P),

P=I, LDA algorithm, Sentiment Analysis Algorithm, suggestion

, total review count

For LDA Algorithm:

**Input:** words w belongs to documents d

Where,

w be the corpus of words.

d is the set of documents.

n be the number of words.

k be the number of words in the document.

Alpha and beta are LDA constants.

**Output:** topic assignments z and counts n(d,k) , n(k,w) and n(k)

Where,

n(d,k) the number of words assigned to topic k in document d.

n(k,w) the number of times word w is assigned to topic k.

College Short Form Name, Department of Computer Engineering 2021 51

Output (O) - O=effective reviews, total count, Search history.

### 4.1.2 Data Flow Diagram

In Data Flow Diagram, we Show that flow of data in our system in DFD0 we show that base DFD in which rectangle present input as well as output and circle show our system, In DFD1 we show actual input and actual output of system input of our system is text or image and output is rumour detected likewise in DFD 2 we present operation of user as well as admin.

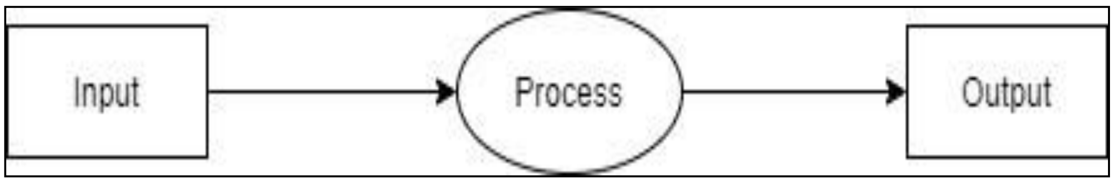


*Figure 4.2: Data Flow (1) diagram*

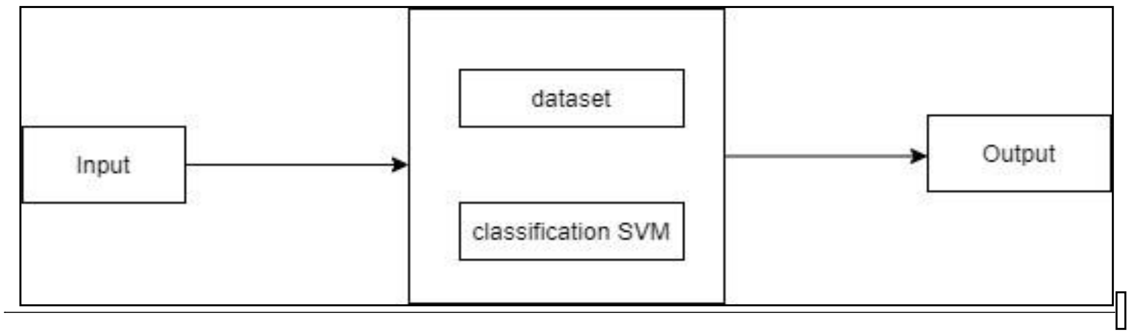


*Figure 4.2: Data Flow (2) diagram*

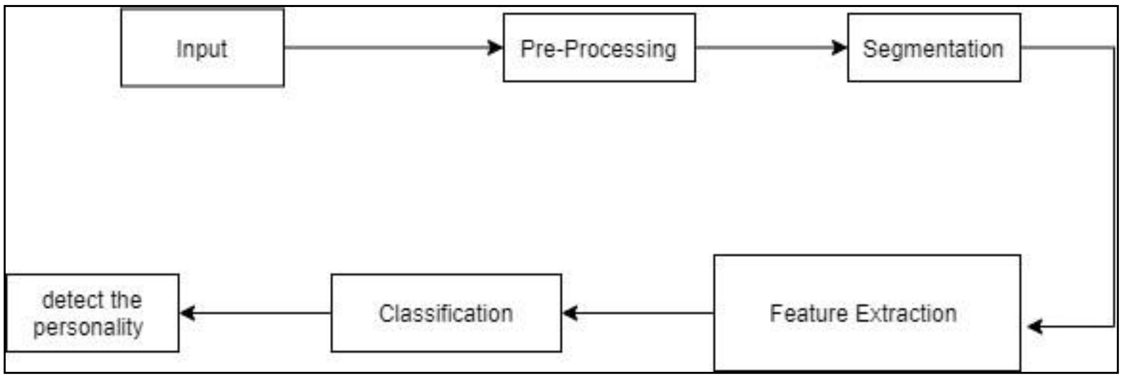


*Figure 4.2: Data Flow (3) diagram*

## 4.2 UML DIAGRAMS

Unified Modelling Language is a standard language for writing software blueprints. The UML may be used to visualize, specify, construct and document the artefacts of a software intensive system. UML is process independent, although optimally it should be used in process that is use case driven, architecture-centric, iterative, and incremental. The Number of UML Diagram is available.

### Class Diagram

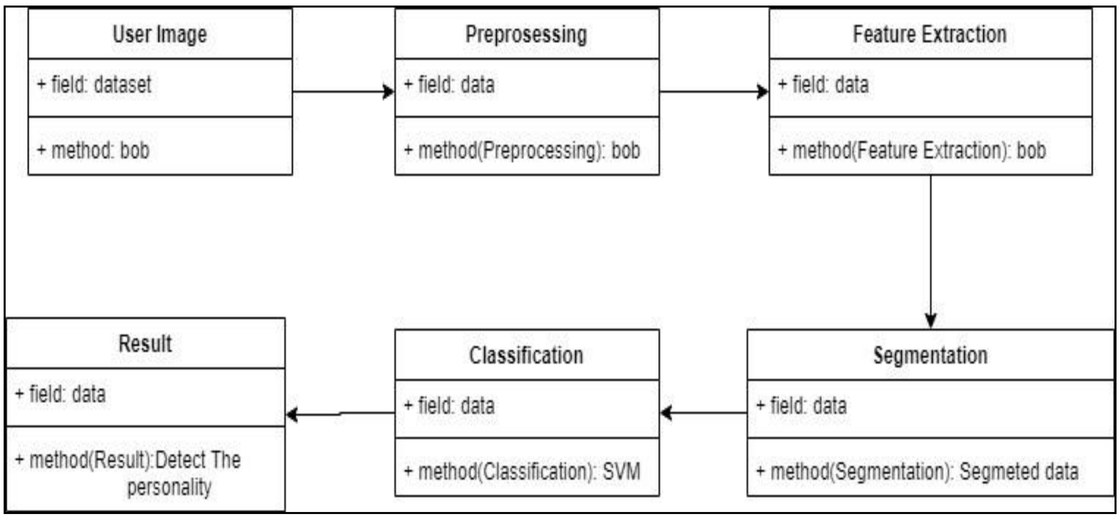


*Figure 4.3: Class Diagram*

**Use case Diagram**

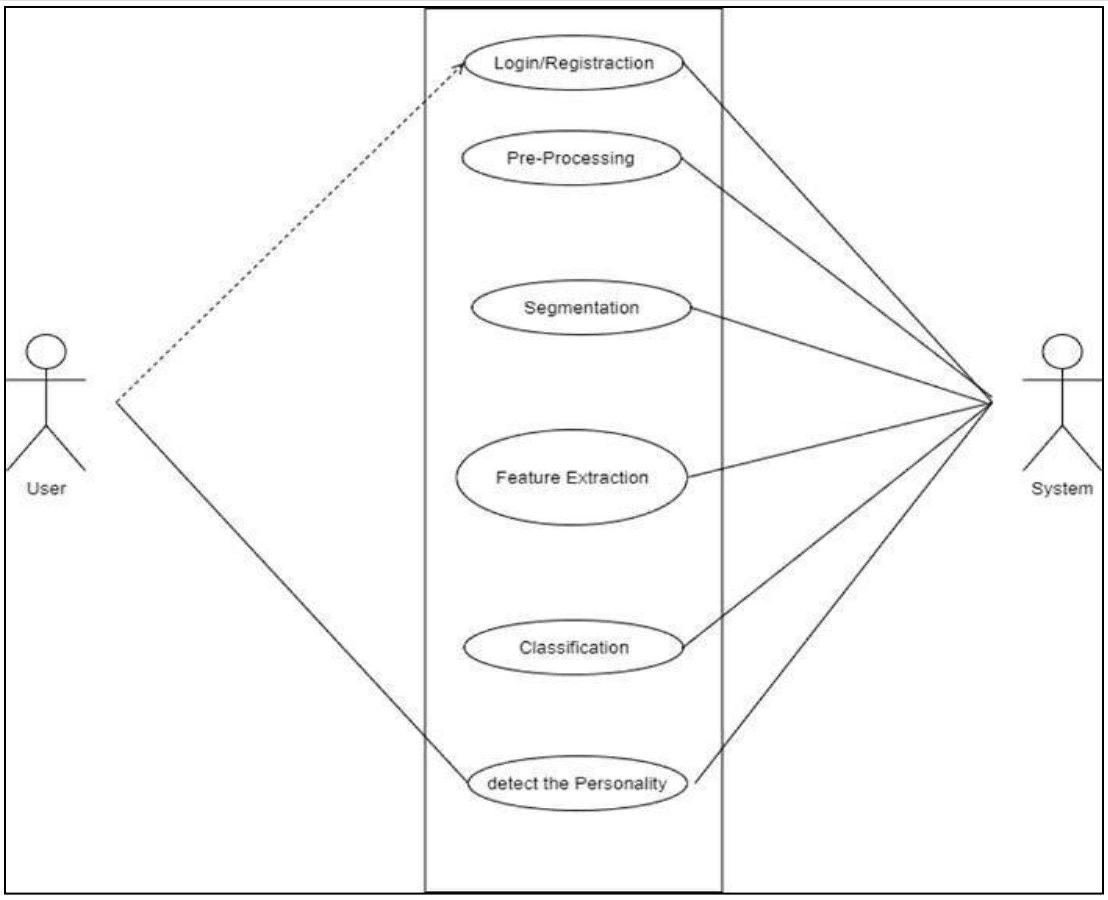


*Figure 4.4: Use case Diagram*

## Activity Diagram

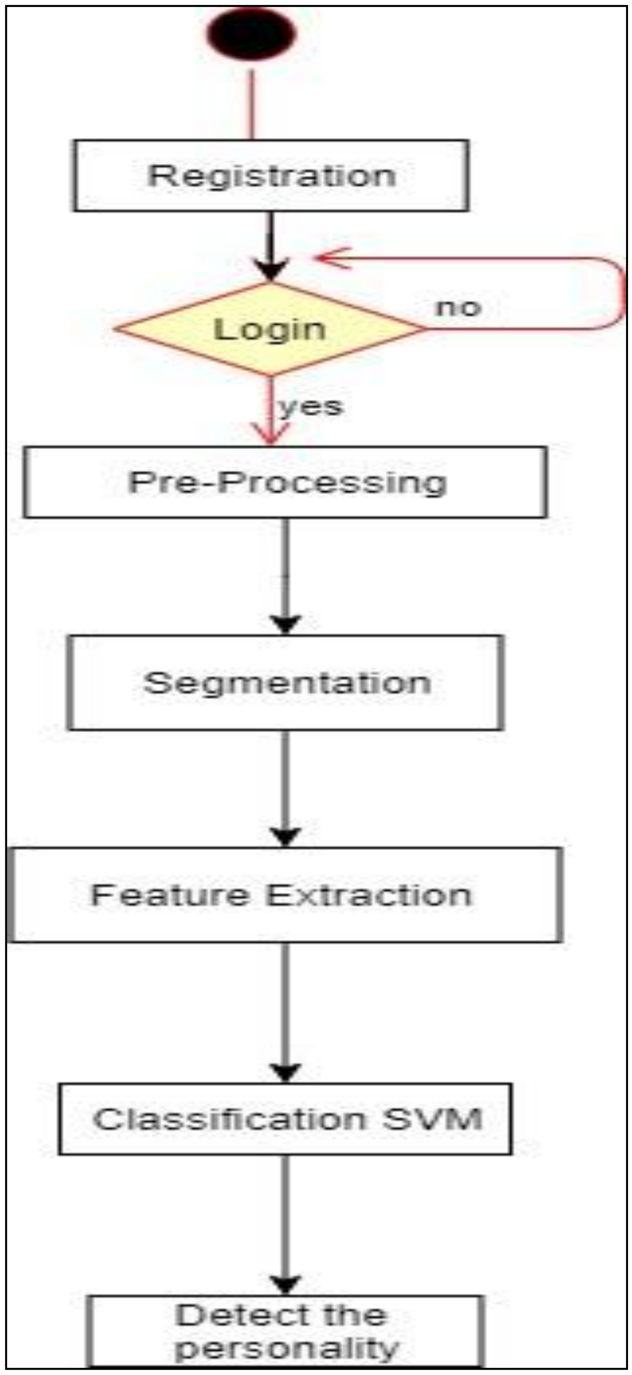


*Figure 4.5: Activity Diagram*

## Sequence Diagram

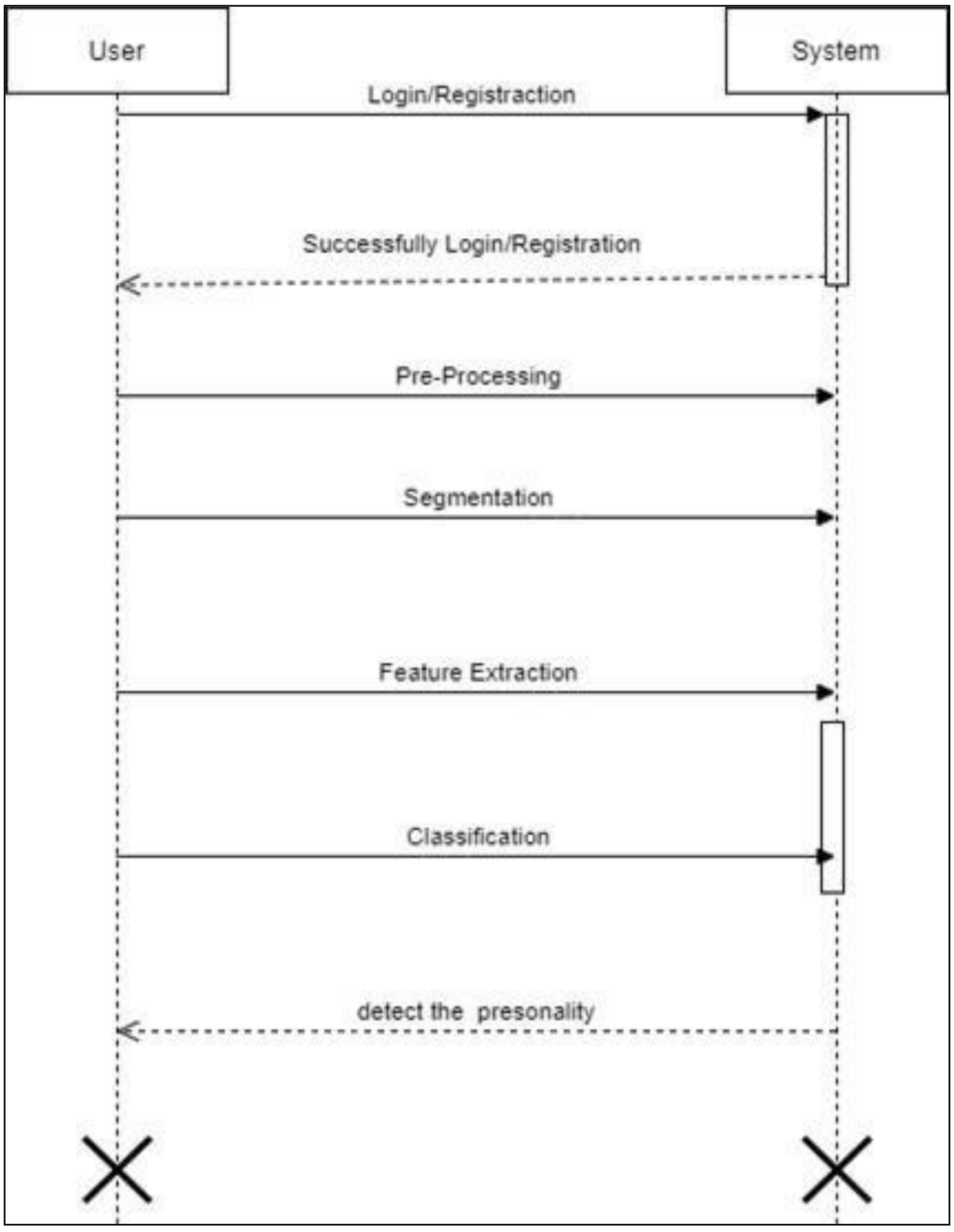


*Figure 4.6: Sequence Diagram*

# CHAPTER 5

# SOFTWARE INFORMATION

Python is an interpreted, high-level and general-purpose programming language.

Created by Guido van Rossum and first released in 1991, Python's design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

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

Python was created in the late 1980s as a successor to the ABC language. Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting.

Python 3.0, released in 2008, was a major revision of the language that is not completely backward-compatible, and much Python 2 code does not run unmodified on Python 3.

The Python 2 language was officially discontinued in 2020 (first planned for

2015), and "Python 2.7.18 is the last Python 2.7 release and therefore the last Python 2 release."[30] No more security patches or other improvements will be released for it. With Python 2's end-of-life, only Python 3.6.x and later are supported.

Python interpreters are available for many operating systems. A global community of programmers develops and maintains CPython, a free and open-source reference implementation. A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

Python was conceived in the late 1980s by Guido van Rossum at Centrum Wiskunde Informatics (CWI) in the Netherlands as a successor to the ABC language (itself inspired by SETL), capable of exception handling and interfacing with the Amoeba operating system. Its implementation began in December 1989. Van Rossum shouldered sole responsibility for the project, as the lead developer, until 12

July 2018, when he announced his "permanent vacation" from his responsibilities as

Python's Benevolent Dictator For Life, a title the Python community bestowed

Upon him to reflect his long-term commitment as the project’s chief decision-maker. He now shares his leadership as a member of a five-person steering council. In January 2019, active Python core developers elected Brett Cannon, Nick Coughlan, and Barry

Warsaw, Carol Willing and Van Rossum to a five-member “Steering Council” to lead the project.

Anaconda: Anaconda is a free and open-source distribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive analytics, etc.), that aims to simplify package management and deployment. The distribution includes data-science packages suitable for Windows, Linux, and macOS. It is developed and maintained by Anaconda, Inc., which was founded by Peter Wang and Travis Oliphant in 2012. As an Anaconda, Inc. product, it is also known as Anaconda Distribution or Anaconda Individual Edition, while other products from the company are Anaconda Team Edition and Anaconda Enterprise Edition, both of which are not free.

Package versions in Anaconda are managed by the package management system conda. This package manager was spun out as a separate open-source package as it ended up being useful on its own and for other things than Python. There is also a small, bootstrap version of Anaconda called Miniconda, which includes only; conda, Python, the packages they depend on, and a small number of other packages. Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages [citation needed]. It will install a package and any of its dependencies regardless of the state of the existing installation [citation needed]. Because of this, a user with a working installation of, for example, Google Tensor flow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensor flow. In some cases, the package may appear to work but produce different results in detail.

In contrast, conda analyses the current environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have Tensor flow version 2,0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Open source packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with conda

# CHAPTER 6

# SOFTWARE TESTING

## 6.1 INTRODUCTION

### 6.1.1 Purpose

Testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing also provides an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs. Software testing can also be stated as the process of validating and verifying that a software program or application or product:

1. Meets the business and technical requirements that guided its design and development;

2. Works as expected; and

3. Can be implemented with the same characteristics

### 6.1.2 Test Plan

To test this application we are going with proper sequencing of testing like unit, integration, validation, GUI, Low level and High level test cases, major scenarios likewise. We will go with the GUI testing first and then integration testing. After integration testing performs the high level test cases and major scenarios which can affect the working on the application. We will perform the testing on the data transmitted using the various inputs and outputs and validate the results. It also intends to cover any deviations that the project might take from the initially agreed Test Strategy in terms of scope, testing methodology, tools, etc... This test plan covers details of testing activities for this project and scope.

### 6.1.3 Software To Be Tested

**1. Edraw Max:**

It enables students, teachers and business professional store liable create and publish various kinds of diagram store present any ideas. With this application users can easily create professional- looking flow charts, organizational charts, network diagrams, business presentations, building plans, mind maps, science illustration, fashion deCollege Short Form Name, Department of Computer Engineering 2021 35 signs, UML diagrams and much more.

**2. Star UML:**

Star UML is a fully fledged, open source, UML modeling tool thats supports the ability to create software designs, from basic concepts, through to the coded solution. The user should be aware that this tool is more complex than a simple UML diagram editing tool, in that, through the use of the model Drive Architecture (MDA) standard, the tool supports complex modeling which is realizable in code. College Short Form Name, Department of Computer Engineering 2021 36

## 6.2 TEST CASES

### 6.2.1 GUI Testing

Graphical User Interface (GUI) testing is the one of the mechanism in which user interface developed System Under some graphical rules. GUI testing includes checking various controls- menus, buttons, icons, dialog boxes and windows etc. Proposed system is tested for user inputs against different modules, validations are done. GUI is tested for appearance of different controls, visibility graphs is tested. GUI testing involves following actions:

1. Check all elements for size, position, width, length and acceptance of characters or numbers. For instance, you must be able to provide inputs to the input fields.

2. Overall functionality related with performance of user’s graphical interface are checked.

3. Check Error Messages are displayed correctly.

4. Check the font, layout details, style and display warning messages if it is false.

5. Check the positioning of GUI elements.

### 6.2.2 Unit Testing

It is the testing of individual software units of the application .it is done after the complexion of an individual unit before integration. 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. 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.

### 6.2.3 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.

# CHAPTER 7

# SYSTEM IMPLEMENTATION

## 7.1 GUI MAIN

```
from tkinter import *
import tkinter as tk

from PIL import Image ,ImageTk

from tkinter.ttk import *
from pymsgbox import *

root=tk.Tk()

root.title("Personality Prediction")
w,h = root.winfo_screenwidth(),root.winfo_screenheight()

bg = Image.open("C:/Users/suhas/Desktop/personality_Prediction/bg1.jpg")
bg.resize((1800,800),Image.ANTIALIAS)
print(w,h)
bg_img = ImageTk.PhotoImage(bg)
bg_lbl = tk.Label(root,image=bg_img)
bg_lbl.place(x=0,y=93)
#, relwidth=1, relheight=1)

w = tk.Label(root, text="Personality
Prediction",width=40,background="#00BFFF",height=2,font=("Times new
roman",23,"bold"))
w.place(x=0,y=15)

w,h = root.winfo_screenwidth(),root.winfo_screenheight()
root.geometry("%dx%d+0+0"%(w,h))
root.configure(background="#00BFFF")
```

```
from tkinter import messagebox as ms



def Login():
    from subprocess import call
    call(["python","login1.py"])
def Register():
    from subprocess import call
    call(["python","registration.py"])


wlcm=tk.Label(root,text="......Welcome to Personality Prediction System ......",width=90,height=3,background="#00BFFF",foreground="black",font=("Times new roman",22,"bold"))
wlcm.place(x=0,y=620)


d2=tk.Button(root,text="Login",command=Login,width=9,height=2,bd=0,background="#00BFFF",foreground="black",font=("times new roman",18,"bold"))
d2.place(x=1200,y=18)


d3=tk.Button(root,text="Register",command=Register,width=9,height=2,bd=0,background="#00BFFF",foreground="black",font=("times new roman",18,"bold"))
d3.place(x=1300,y=18)


root.mainloop()
```

## 7.2 REGISTRATION

```
import tkinter as tk
# from tkinter import *
from tkinter import messagebox as ms
import sqlite3
from PIL import Image, ImageTk
import re
import random
import os
import cv2

window = tk.Tk()
w,h = window.winfo_screenwidth(),window.winfo_screenheight()
window.geometry("%dx%d+0+0"%(w,h))
window.title("REGISTRATION FORM")
window.configure(background="skyblue")

Fullname = tk.StringVar()
address = tk.StringVar()
username = tk.StringVar()
Email = tk.StringVar()
Phoneno = tk.IntVar()
var = tk.IntVar()
age = tk.IntVar()
password = tk.StringVar()
password1 = tk.StringVar()

value = random.randint(1, 1000)
print(value)
```

```
# database code
db = sqlite3.connect('evaluation.db')
cursor = db.cursor()
cursor.execute("CREATE TABLE IF NOT EXISTS registration"
          "(Fullname TEXT, address TEXT, username TEXT, Email TEXT, Phoneno
TEXT,Gender TEXT,age TEXT , password TEXT)")
db.commit()

def password_check(passwd):

    SpecialSym =['$', '@', '#', '%']
    val = True

    if len(passwd) < 6:
        print('length should be at least 6')
        val = False

    if len(passwd) > 20:
        print('length should be not be greater than 8')
        val = False

    if not any(char.isdigit() for char in passwd):
        print('Password should have at least one numeral')
        val = False

    if not any(char.isupper() for char in passwd):
        print('Password should have at least one uppercase letter')
        val = False

    if not any(char.islower() for char in passwd):
        print('Password should have at least one lowercase letter')
```

```
            val = False

        if not any(char in SpecialSym for char in passwd):
            print('Password should have at least one of the symbols $@#')
            val = False
        if val:
            return val

def insert():
    fname = Fullname.get()
    addr = address.get()
    un = username.get()
    email = Email.get()
    mobile = Phoneno.get()
    gender = var.get()
    time = age.get()
    pwd = password.get()
    cnpwd = password1.get()

    with sqlite3.connect('evaluation.db') as db:
        c = db.cursor()

    # Find Existing username if any take proper action
    find_user = ('SELECT * FROM registration WHERE username = ?')
    c.execute(find_user, [(username.get())])

    # else:
    #   ms.showinfo('Success!', 'Account Created Successfully !')

    # to check mail
    #regex = '^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$'
    regex='^[a-z0-9]+[\._]?[a-z0-9]+[@]\w+[.]\w{2,3}$'
    if (re.search(regex, email)):
        a = True
    else:
        a = False
    # validation
    if (fname.isdigit() or (fname == "")):
        ms.showinfo("Message", "please enter valid name")
    elif (addr == ""):
        ms.showinfo("Message", "Please Enter Address")
    elif (email == "") or (a == False):
        ms.showinfo("Message", "Please Enter valid email")
    elif((len(str(mobile)))<10 or len(str((mobile)))>10):
        ms.showinfo("Message", "Please Enter 10 digit mobile number")
    elif ((time > 100) or (time == 0)):
        ms.showinfo("Message", "Please Enter valid age")
    elif (c.fetchall()):
        ms.showerror('Error!', 'Username Taken Try a Diffrent One.')
    elif (pwd == ""):
        ms.showinfo("Message", "Please Enter valid password")
    elif (var == False):
        ms.showinfo("Message", "Please Enter gender")
    elif(pwd=="")or(password_check(pwd))!=True:
        ms.showinfo("Message", "password must contain atleast 1 Uppercase letter,1 
symbol,1 number")
    elif (pwd != cnpwd):
        ms.showinfo("Message", "Password Confirm password must be same")
    else:
        conn = sqlite3.connect('evaluation.db')
        with conn:
            cursor = conn.cursor()
        cursor.execute(
            'INSERT INTO registration(Fullname, address, username, Email, Phoneno,
Gender, age , password) VALUES(?,?,?,?,?,?,?,?)',
            (fname, addr, un, email, mobile, gender, time, pwd))

        conn.commit()
        db.close()
        ms.showinfo('Success!', 'Account Created Successfully !')
        # window.destroy()
        window.destroy()

##############################################################################################################################################################

#from subprocess import call
#call(["python", "lecture_login.py"])



# assign and define variable
# def login():

#####For background Image
image2 =Image.open('r1.jpg')
image2 =image2.resize((1800,1000), Image.ANTIALIAS)

background_image=ImageTk.PhotoImage(image2)

background_label = tk.Label(window, image=background_image)

background_label.image = background_image
background_label.place(x=0, y=0) #, relwidth=1, relheight=1)


frame = tk.LabelFrame(window, text="", width=600, height=570, bd=10,
font=('times', 14, ' bold '),bg="#ffffb3")

frame.grid(row=0, column=0, sticky='nw')

frame.place(x=370, y=120)


lbl = tk.Label(window, text="Registration Form", font=('times', 35,' bold '), height=1,
width=65,bg="#A74AC7",fg="black")

lbl.place(x=0, y=0)


#l1 = tk.Label(window, text="Registration Form", font=("Times new roman", 30,
"bold"), bg="blue4", fg="red")

#l1.place(x=490, y=40)


# that is for label1 registration


l2 = tk.Label(frame, text="Full Name :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2",bd=5)

l2.place(x=30, y=30)

t1 = tk.Entry(frame, textvar=Fullname, width=20, font=('', 15),bd=5)

t1.place(x=230, y=30)

# that is for label 2 (full name)


l3 = tk.Label(frame, text="Address :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2",bd=5)
```

```
l3.place(x=30, y=80)

t2 = tk.Entry(frame, textvar=address, width=20, font=('', 15),bd=5)

t2.place(x=230, y=80)

# that is for label 3(address)



# that is for label 4(blood group)


l5 = tk.Label(frame, text="E-mail :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2")

l5.place(x=30, y=130)

t4 = tk.Entry(frame, textvar=Email, width=20, font=('', 15),bd=5)

t4.place(x=230, y=130)

# that is for email address


l6 = tk.Label(frame, text="Phone number :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2")

l6.place(x=30, y=180)

t5 = tk.Entry(frame, textvar=Phoneno, width=20, font=('', 15),bd=5)

t5.place(x=230, y=180)

# phone number

l7 = tk.Label(frame, text="Gender :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2")

l7.place(x=30, y=230)

# gender

tk.Radiobutton(frame, text="Male", padx=5, width=5, bg="antiquewhite2",
font=("bold", 15), variable=var, value=1).place(x=230,

                                                          y=230)

tk.Radiobutton(frame, text="Female", padx=20, width=4, bg="antiquewhite2",
font=("bold", 15), variable=var, value=2).place(

    x=340, y=230)
```

```
l8 = tk.Label(frame, text="Age :", width=12, font=("Times new roman", 15, "bold"),
bg="antiquewhite2")

l8.place(x=30, y=280)

t6 = tk.Entry(frame, textvar=age, width=20, font=('', 15),bd=5)

t6.place(x=230, y=280)


l4 = tk.Label(frame, text="User Name :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2")

l4.place(x=30, y=330)

t3 = tk.Entry(frame, textvar=username, width=20, font=('', 15),bd=5)

t3.place(x=230, y=330)


l9 = tk.Label(frame, text="Password :", width=12, font=("Times new roman", 15,
"bold"), bg="antiquewhite2")

l9.place(x=30, y=380)

t9 = tk.Entry(frame, textvar=password, width=20, font=('', 15), show="*",bd=5)

t9.place(x=230, y=380)


l10 = tk.Label(frame, text="Confirm Password:", width=13, font=("Times new
roman", 15, "bold"), bg="antiquewhite2")

l10.place(x=30, y=430)


t10 = tk.Entry(frame, textvar=password1, width=20, font=('', 15), show="*",bd=5)

t10.place(x=230, y=430)


btn = tk.Button(frame, text="Register", bg="red",font=("",20),fg="white", width=9,
height=1, command=insert)

btn.place(x=230, y=480)

# tologin=tk.Button(window , text="Go To Login", bg ="dark green", fg = "white",
width=15, height=2, command=login)

# tologin.place(x=330, y=600)

window.mainloop()
```

**7.3 LOGIN**

```
from tkinter import *
from tkinter import messagebox as ms
from PIL import ImageTk
import sqlite3

class login_system:
    def __init__(self,root):
        self.root = root
        self.root.title("log in ")
        self.root.geometry("1350x700+0+0")

        ##______________________All Images ___________________________##

    #=====Variable======#

    # Some Usefull variables
    # Create Widgets

        # Some Usefull variables
        self.username = StringVar()
        self.password = StringVar()
        self.n_username = StringVar()
        self.n_password = StringVar()


self.bg1_icon=ImageTk.PhotoImage(file=r"C:/Users/suhas/Desktop/personality_Predi
ction/L.jpg")
```

```
self.bg_icon=ImageTk.PhotoImage(file=r"C:/Users/suhas/Desktop/personality_Predict
ion/L.jpg")

self.user_icon=ImageTk.PhotoImage(file=r"C:/Users/suhas/Desktop/personality_Predi
ction/u1.png")

self.pass_icon=ImageTk.PhotoImage(file=r"C:/Users/suhas/Desktop/personality_Predi
ction/L.jpg")

        bg_lbl=Label(self.root,image=self.bg1_icon).pack()

        title=Label(self.root, text="LOGIN", font=("Times new roman", 40, "bold"),
bg="#900C3F",fg="white",bd=5,relief=RAISED)
        title.place(x=0, y=0,relwidth=1)

        Login_frame=Frame(self.root,bg="#ff9999")
        Login_frame.place(x=400,y=150)

logolbl=Label(Login_frame,image=self.bg_icon,bd=0).grid(row=0,columnspan=2,pad
y=20)

lbluser=Label(Login_frame,text="Username",image=self.user_icon,compound=LEFT,
font=("Times new roman", 20,
"bold"),bg="white").grid(row=1,column=0,padx=20,pady=10)
        txtuser=Entry(Login_frame,bd=5,textvariable=self.username
,relief=GROOVE,font=("",15)).grid(row=1,column=1,padx=20)

lblpass=Label(Login_frame,text="Password",image=self.pass_icon,compound=LEFT,
font=("Times new roman", 20,
"bold"),bg="white").grid(row=2,column=0,padx=20,pady=10)

txtpass=Entry(Login_frame,bd=5,textvariable=self.password,relief=GROOVE,font=("
",15),show='*').grid(row=2,column=1,padx=20)
```

```
btn_log=Button(Login_frame,text="Login",command=self.login,width=15,font=("Tim
es new roman", 14,
"bold"),bg="#900C3F",fg="white").grid(row=3,column=1,pady=10)


        btn_log=Button(Login_frame,text="Create
Account",command=self.login,width=15,font=("Times new roman", 14,
"bold"),bg="#900C3F",fg="white").grid(row=3,column=2,pady=10)


        # Login Function
    def login(self):
        # Establish Connection


        with sqlite3.connect('evaluation.db') as db:
            c = db.cursor()


        # Find user If there is any take proper action
        db = sqlite3.connect('evaluation.db')
        cursor = db.cursor()
        cursor.execute("CREATE TABLE IF NOT EXISTS registration"
                       "(Fullname TEXT, address TEXT, username TEXT, Email TEXT,
Phoneno TEXT,Gender TEXT,age TEXT , password TEXT)")
        db.commit()
        find_entry = ('SELECT * FROM registration WHERE username = ? and
password = ?')
        c.execute(find_entry, [(self.username.get()), (self.password.get())])
        result = c.fetchall()


        if result:
            msg = ""
            # self.logf.pack_forget()
            # self.head['text'] = self.username.get() + '\n Loged In'
            # msg = self.head['text']
            #           self.head['pady'] = 150
            print(msg)
            ms.showinfo("messege", "LogIn sucessfully")
            # ===========================================
            root.destroy()

            from subprocess import call
            call(['python','Personality_Detector.py'])

            # ================================================
        else:
            ms.showerror('Oops!', 'Username Or Password Did Not Found/Match.')

    def new_user(self):
        # Establish Connection
        with sqlite3.connect('evaluation.db') as db:
            c = db.cursor()

        # Find Existing username if any take proper action
        find_user = ('SELECT * FROM user WHERE username = ?')
        c.execute(find_user, [(self.username.get())])
        if c.fetchall():
            ms.showerror('Error!', 'Username Taken Try a Diffrent One.')
        else:
            ms.showinfo('Success!', 'Account Created Successfully !')
            self.log()
        # Create New Account
        insert = 'INSERT INTO user(username,password) VALUES(?,?)'
        c.execute(insert, [(self.n_username.get()), (self.n_password.get())])
        db.commit()
        # Frame Packing Methords

    def registration(self):
        root.destroy()
        from subprocess import call
        call(["python", "registration.py"])

        # mainloop(root)

    def log(self):
        self.username.set('')
        self.password.set('')
        self.crf.pack_forget()
        self.head['text'] = 'LOGIN'
        self.logf.pack()

    def cr(self):
        self.n_username.set('')
        self.n_password.set('')
        self.logf.pack_forget()
        self.head['text'] = 'Create Account'
        self.crf.pack()



root = Tk()
obj = login_system(root)
root.mainloop()
```

## 7.4 PREPROCESSING PAGE

```
import pandas as pd
import numpy as np
import re

# plotting
import seaborn as sns
import matplotlib.pyplot as plt
import nltk
from nltk.stem import PorterStemmer, WordNetLemmatizer
from nltk.corpus import stopwords
from nltk import word_tokenize
nltk.download('stopwords')
nltk.download('wordnet')

# read data
data = pd.read_csv('E:/personality_Prediction/mbti_1.csv')
data.head(10)

[p.split('|||') for p in data.head(2).posts.values]

cnt_types = data['type'].value_counts()

plt.figure(figsize=(12,4))
sns.barplot(cnt_types.index, cnt_types.values, alpha=0.8)
plt.ylabel('Number of Occurrences', fontsize=12)
plt.xlabel('Types', fontsize=12)
plt.show()

def get_types(row):
    t=row['type']
    I = 0; N = 0
    T = 0; J = 0

    if t[0] == 'I': I = 1
    elif t[0] == 'E': I = 0
    else: print('I-E incorrect')

    if t[1] == 'N': N = 1
    elif t[1] == 'S': N = 0
    else: print('N-S incorrect')

    if t[2] == 'T': T = 1
    elif t[2] == 'F': T = 0
    else: print('T-F incorrect')

    if t[3] == 'J': J = 1
    elif t[3] == 'P': J = 0
    else: print('J-P incorrect')
    return pd.Series( {'IE':I, 'NS':N , 'TF': T, 'JP': J })

data = data.join(data.apply (lambda row: get_types (row),axis=1))
data.head(5)

print ("Introversion (I) /  Extroversion (E):\t", data['IE'].value_counts()[0], " / ",
data['IE'].value_counts()[1])
print ("Intuition (N) – Sensing (S):\t\t", data['NS'].value_counts()[0], " / ",
data['NS'].value_counts()[1])
print ("Thinking (T) – Feeling (F):\t\t", data['TF'].value_counts()[0], " / ",
data['TF'].value_counts()[1])
print ("Judging (J) – Perceiving (P):\t\t", data['JP'].value_counts()[0], " / ",
data['JP'].value_counts()[1])
```

```
N = 4

but = (data['IE'].value_counts()[0], data['NS'].value_counts()[0],
data['TF'].value_counts()[0], data['JP'].value_counts()[0])

top = (data['IE'].value_counts()[1], data['NS'].value_counts()[1],
data['TF'].value_counts()[1], data['JP'].value_counts()[1])


ind = np.arange(N)    # the x locations for the groups

width = 0.7      # the width of the bars: can also be len(x) sequence


p1 = plt.bar(ind, but, width)

p2 = plt.bar(ind, top, width, bottom=but)


plt.ylabel('Count')

plt.title('Distribution accoss types indicators')

plt.xticks(ind, ('I/E',  'N/S', 'T/F', 'J/P',))


plt.show()


data[['IE','NS','TF','JP']].corr()


cmap = plt.cm.RdBu

corr = data[['IE','NS','TF','JP']].corr()

plt.figure(figsize=(12,10))

plt.title('Pearson Features Correlation', size=15)

sns.heatmap(corr, cmap=cmap,  annot=True, linewidths=1)


b_Pers = {'I':0, 'E':1, 'N':0, 'S':1, 'F':0, 'T':1, 'J':0, 'P':1}

b_Pers_list = [{0:'I', 1:'E'}, {0:'N', 1:'S'}, {0:'F', 1:'T'}, {0:'J', 1:'P'}]


def translate_personality(personality):
    # transform mbti to binary vector

    return [b_Pers[l] for l in personality]

def translate_back(personality):
    # transform binary vector to mbti personality

    s = ""
    for i, l in enumerate(personality):
        s += b_Pers_list[i][l]
    return s

# Check ...
d = data.head(4)
list_personality_bin = np.array([translate_personality(p) for p in d.type])
print("Binarize MBTI list: \n%s" % list_personality_bin)


# We want to remove these from the psosts
unique_type_list = ['INFJ', 'ENTP', 'INTP', 'INTJ', 'ENTJ', 'ENFJ', 'INFP', 'ENFP',
       'ISFP', 'ISTP', 'ISFJ', 'ISTJ', 'ESTP', 'ESFP', 'ESTJ', 'ESFJ']

unique_type_list = [x.lower() for x in unique_type_list]

# Lemmatize
stemmer = PorterStemmer()
lemmatiser = WordNetLemmatizer()

# Cache the stop words for speed
cachedStopWords = stopwords.words("english")
```

```
def pre_process_data(data, remove_stop_words=True, remove_mbti_profiles=True):

    list_personality = []
    list_posts = []
    len_data = len(data)
    i=0

    for row in data.iterrows():
        i+=1
        if (i % 500 == 0 or i == 1 or i == len_data):
            print("%s of %s rows" % (i, len_data))

        ##### Remove and clean comments
        posts = row[1].posts
        temp = re.sub('http[s]?://(?:[a-zA-Z]|[0-9]|[$-_@.&+]|(?:%[0-9a-fA-F][0-9a-fA-
F]))+', ' ', posts)
        temp = re.sub("[^a-zA-Z]", " ", temp)
        temp = re.sub(' +', ' ', temp).lower()
        if remove_stop_words:
            temp = " ".join([lemmatiser.lemmatize(w) for w in temp.split(' ') if w not in
cachedStopWords])
        else:
            temp = " ".join([lemmatiser.lemmatize(w) for w in temp.split(' ')])

        if remove_mbti_profiles:
            for t in unique_type_list:
                temp = temp.replace(t,"")

        type_labelized = translate_personality(row[1].type)
        list_personality.append(type_labelized)
        list_posts.append(temp)
    list_posts = np.array(list_posts)
    list_personality = np.array(list_personality)
    return list_posts, list_personality

list_posts, list_personality  = pre_process_data(data, remove_stop_words=True)

print("Num posts and personalities: ",  list_posts.shape, list_personality.shape)

list_posts[0]
list_personality[0]

from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.manifold import TSNE

# Posts to a matrix of token counts
cntizer = CountVectorizer(analyzer="word",
                             max_features=1500,
                             tokenizer=None,
                             preprocessor=None,
                             stop_words=None,
                             max_df=0.7,
                             min_df=0.1)

# Learn the vocabulary dictionary and return term-document matrix
print("CountVectorizer...")
X_cnt = cntizer.fit_transform(list_posts)

# Transform the count matrix to a normalized tf or tf-idf representation
tfizer = TfidfTransformer()
```

```
print("Tf-idf...")
# Learn the idf vector (fit) and transform a count matrix to a tf-idf representation
X_tfidf =  tfizer.fit_transform(X_cnt).toarray()

feature_names = list(enumerate(cntizer.get_feature_names()))
feature_names

X_tfidf.shape
print("X: Posts in tf-idf representation \n* 1st row:\n%s" % X_tfidf[0])
type_indicators = [ "IE: Introversion (I) / Extroversion (E)", "NS: Intuition (N) – Sensing (S)",
                  "FT: Feeling (F) - Thinking (T)", "JP: Judging (J) – Perceiving (P)"  ]

for l in range(len(type_indicators)):
    print(type_indicators[l])

print("MBTI 1st row: %s" % translate_back(list_personality[0,:]))
print("Y: Binarized MBTI 1st row: %s" % list_personality[0,:])
from numpy import loadtxt
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Posts in tf-idf representation
X = X_tfidf

# Let's train type indicator individually
for l in range(len(type_indicators)):
    print("%s ..." % (type_indicators[l]))
    # Let's train type indicator individually
    Y = list_personality[:,l]

    # split data into train and test sets
    seed = 7
    test_size = 0.33
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size,
random_state=seed)

    # fit model on training data
    model = XGBClassifier()
    model.fit(X_train, y_train)

    # make predictions for test data
    y_pred = model.predict(X_test)
    predictions = [round(value) for value in y_pred]
    # evaluate predictions
    accuracy = accuracy_score(y_test, predictions)
    print("* %s Accuracy: %.2f%%" % (type_indicators[l], accuracy * 100.0))

for l in range(len(type_indicators)):
    print("%s ..." % (type_indicators[l]))

    Y = list_personality[:,l]

    # split data into train and test sets
    seed = 7
    test_size = 0.33
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size,
random_state=seed)
```

```
    # fit model on training data
    model = XGBClassifier()
    eval_set = [(X_test, y_test)]
    model.fit(X_train, y_train, early_stopping_rounds=10, eval_metric="logloss",
eval_set=eval_set, verbose=True)

    # make predictions for test data
    y_pred = model.predict(X_test)
    predictions = [round(value) for value in y_pred]
    # evaluate predictions
    accuracy = accuracy_score(y_test, predictions)
    print("* %s Accuracy: %.2f%%" % (type_indicators[l], accuracy * 100.0))

from xgboost import plot_importance

# Only the 1st indicator
y = list_personality[:,0]
# fit model on training data
model = XGBClassifier()
model.fit(X, y)
# plot feature importance
ax = plot_importance(model, max_num_features=25)

fig = ax.figure
fig.set_size_inches(15, 20)

plt.show()
features = sorted(list(enumerate(model.feature_importances_)), key=lambda x: x[1],
reverse=True)
for f in features[0:25]:
    print("%d\t%f\t%s" % (f[0],f[1],cntizer.get_feature_names()[f[0]]))
```

```
# Save xgb_params for late discussuin
default_get_xgb_params = model.get_xgb_params()

default_get_xgb_params = model.get_xgb_params()
print (default_get_xgb_params)

param = {}

param['n_estimators'] = 200
param['max_depth'] = 2
param['nthread'] = 8
param['learning_rate'] = 0.2

# Let's train type indicator individually
for l in range(len(type_indicators)):
    print("%s ..." % (type_indicators[l]))

    Y = list_personality[:,l]

    # split data into train and test sets
    seed = 7
    test_size = 0.33
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size, 
random_state=seed)

    # fit model on training data
    model = XGBClassifier(**param)
    model.fit(X_train, y_train)
    # make predictions for test data
    y_pred = model.predict(X_test)
    predictions = [round(value) for value in y_pred]
    # evaluate predictions
    accuracy = accuracy_score(y_test, predictions)
    print("* %s Accuracy: %.2f%%" % (type_indicators[l], accuracy * 100.0))

# from numpy import loadtxt
# from xgboost import XGBClassifier
# from sklearn.model_selection import GridSearchCV
# from sklearn.model_selection import StratifiedKFold

# # Posts in tf-idf representation
# X = X_tfidf

# # setup parameters for xgboost
# param = {}
# param['n_estimators'] = 200
# param['max_depth'] = 2
# param['nthread'] = 8
# param['learning_rate'] = 0.2



# # Let's train type indicator individually
# for l in range(len(type_indicators)):
#     print("%s ..." % (type_indicators[l]))

#     Y = list_personality[:,l]
#     model = XGBClassifier(**param)
#     # learning_rate = [0.0001, 0.001, 0.01, 0.1, 0.2, 0.3]
#     # param_grid = dict(learning_rate=learning_rate)

#     param_grid = {
```

```
#       'n_estimators' : [ 200, 300],
#       'learning_rate': [ 0.2, 0.3]
#       # 'learning_rate': [ 0.01, 0.1, 0.2, 0.3],
#       # 'max_depth': [2,3,4],
#   }


#   kfold = StratifiedKFold(n_splits=10, shuffle=True, random_state=7)
#   grid_search = GridSearchCV(model, param_grid, scoring="neg_log_loss",
n_jobs=-1, cv=kfold)
#   grid_result = grid_search.fit(X, Y)

#   # summarize results
#   print("* Best: %f using %s" % (grid_result.best_score_,
grid_result.best_params_))
#   means = grid_result.cv_results_['mean_test_score']
#   stds = grid_result.cv_results_['std_test_score']
#   params = grid_result.cv_results_['params']
#   for mean, stdev, param in zip(means, stds, params):
#       print("* %f (%f) with: %r" % (mean, stdev, param))
```

# my_posts  = """Getting started with data science and applying machine learning has never been as simple as it is now. There are many free and paid online tutorials and courses out there to help you to get started. I’ve recently started to learn, play, and work on Data Science & Machine Learning on Kaggle.com. In this brief post, I’d like to share my experience with the Kaggle Python Docker image, which simplifies the Data Scientist’s life.

# Awesome #AWS monitoring introduction.

# HPE Software (now @MicroFocusSW) won the platinum reader's choice #ITAWARDS 2017 in the new category #CloudMonitoring

# Certified as AWS Certified Solutions Architect

# Hi, please have a look at my Udacity interview about online learning and machine learning,

# Very interesting to see the  lessons learnt during the HP Operations Orchestration to CloudSlang journey. http://bit.ly/1Xo41ci

```
# I came across a post on devopsdigest.com and need your input: “70% DevOps organizations Unhappy with DevOps Monitoring Tools”

# In a similar investigation I found out that many DevOps organizations use several monitoring tools in parallel. Senu, Nagios, LogStach and SaaS offerings such as DataDog or SignalFX to name a few. However, one element is missing: Consolidation of alerts and status in a single pane of glass, which enables fast remediation of application and infrastructure uptime and performance issues.

# Sure, there are commercial tools on the market for exactly this use case but these tools are not necessarily optimized for DevOps.

# So, here my question to you: In your DevOps project, have you encountered that the lack of consolidation of alerts and status is a real issue? If yes, how did you approach the problem? Or is an ChatOps approach just right?

# You will probably hear more and more about ChatOps - at conferences, DevOps meet-ups or simply from your co-worker at the coffee station. ChatOps is a term and concept coined by GitHub. It's about the conversation-driven development, automation, and operations.

# Now the question is: why and how would I, as an ops-focused engineer, implement and use ChatOps in my organization? The next question then is: How to include my tools into the chat conversation?

# Let’s begin by having a look at a use case. The Closed Looped Incidents Process (CLIP) can be rejuvenated with ChatOps. The work from the incident detection runs through monitoring until the resolution of issues in your application or infrastructure can be accelerated with improved, cross-team communication and collaboration.

# In this blog post, I am going to describe and share my experience with deploying HP Operations Manager i 10.0 (OMi) on HP Helion Public Cloud. An Infrastructure as a Service platform such as HP Helion Public Cloud Compute is a great place to quickly spin-up a Linux server and install HP Operations Manager i for various use scenarios. An example of a good use case is monitoring workloads across public clouds such as AWS and Azure.

# """


# # The type is just a dummy so that the data prep fucntion can be reused

# mydata = pd.DataFrame(data={'type': ['INFJ'], 'posts': [my_posts]})


# my_posts, dummy  = pre_process_data(mydata, remove_stop_words=True)


# my_X_cnt = cntizer.transform(my_posts)

# my_X_tfidf =  tfizer.transform(my_X_cnt).toarray()
# param = {}
# param['n_estimators'] = 200
# param['max_depth'] = 2
# param['nthread'] = 8
# param['learning_rate'] = 0.2

# result = []
# # Let's train type indicator individually
# for l in range(len(type_indicators)):
#     print("%s ..." % (type_indicators[l]))

#     Y = list_personality[:,l]

#     # split data into train and test sets
#     seed = 7
#     test_size = 0.33
#     X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=test_size,
random_state=seed)

#     # fit model on training data
#     model = XGBClassifier(**param)
#     model.fit(X_train, y_train)

#     # make predictions for my  data
#     y_pred = model.predict(my_X_tfidf)
#     result.append(y_pred[0])

# print("The result is: ", translate_back(result))
```

## 7.5 OUTPUT PAGE

```
import pandas as pd, numpy as np, re

from sklearn.metrics import classification_report, accuracy_score , confusion_matrix
from sklearn.model_selection import train_test_split
import tkinter as tk
from sklearn import svm
from PIL import Image, ImageTk
from tkinter import ttk
from joblib import dump , load
from sklearn.feature_extraction.text import TfidfVectorizer
from textblob import TextBlob
from nltk.corpus import stopwords
from sklearn import metrics
from sklearn.model_selection import GridSearchCV
import pickle
import nltk

nltk.download('stopwords')
stop = stopwords.words('english')
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')



root = tk.Tk()
root.title("Personality Prediction using Twitter")
w, h = root.winfo_screenwidth(), root.winfo_screenheight()
root.geometry("%dx%d+0+0" % (w, h))
# image2 =Image.open(r'E:/personality_Prediction/BG.jpg')
# image2 =image2.resize((w,h), Image.ANTIALIAS)
# background_image=ImageTk.PhotoImage(image2)

# background_label = tk.Label(root, image=background_image)

# background_label.image = background_image

# background_label.place(x=0, y=0)

img=ImageTk.PhotoImage(Image.open("s1.jpg"))

img2=ImageTk.PhotoImage(Image.open("s2.jpg"))

img3=ImageTk.PhotoImage(Image.open("s3.jpg"))



logo_label=tk.Label()
logo_label.place(x=0,y=100)

x = 1

# function to change to next image
def move():
	global x
	if x == 4:
		x = 1
	if x == 1:
		logo_label.config(image=img)
	elif x == 2:
		logo_label.config(image=img2)
	elif x == 3:
		logo_label.config(image=img3)
```

```
        x = x+1
        root.after(4000, move)

# calling the function
move()

w = tk.Label(root, text="Personality
Prediction",width=40,background="#7D0552",height=2,font=("Times new
roman",28,"bold"))
w.place(x=500,y=10)

w,h = root.winfo_screenwidth(),root.winfo_screenheight()
root.geometry("%dx%d+0+0"%(w,h))
root.configure(background="#7D0552")

def Train():

    result =
pd.read_csv(r"C:/Users/suhas/Desktop/personality_Prediction/new.csv",encoding =
'unicode_escape')

    result.head()

    result['posts'] = result['posts'].apply(lambda x: ' '.join([word for word in x.split() if
word not in (stop)]))

    def pos(review_without_stopwords):
        return TextBlob(review_without_stopwords).tags

    os = result.posts.apply(pos)
    os1 = pd.DataFrame(os)
    #
```

```
os1.head()


os1['pos'] = os1['posts'].map(lambda x: " ".join(["/".join(x) for x in x]))


result = result = pd.merge(result, os1, right_index=True, left_index=True)
result.head()
result['pos']
review_train, review_test, label_train, label_test = train_test_split(result['pos'],
result['type'],
                                                                       test_size=0.2, random_state=13)


tf_vect = TfidfVectorizer(lowercase=True, use_idf=True, smooth_idf=True,
sublinear_tf=False)


X_train_tf = tf_vect.fit_transform(review_train)
X_test_tf = tf_vect.transform(review_test)



def svc_param_selection(X, y, nfolds):
    Cs = [0.001, 0.01, 0.1, 1, 10]
    gammas = [0.001, 0.01, 0.1, 1]
    param_grid = {'C': Cs, 'gamma': gammas}
    grid_search = GridSearchCV(svm.SVC(kernel='linear'), param_grid, cv=nfolds)
    grid_search.fit(X, y)
    return grid_search.best_params_



svc_param_selection(X_train_tf, label_train, 2)
#


clf = svm.SVC(C=10, gamma=0.001, kernel='linear')
```

```
clf.fit(X_train_tf, label_train)
pred = clf.predict(X_test_tf)

with open('vectorizer.pickle', 'wb') as fin:
    pickle.dump(tf_vect, fin)
with open('mlmodel.pickle', 'wb') as f:
    pickle.dump(clf, f)

pkl = open('mlmodel.pickle', 'rb')
clf = pickle.load(pkl)
vec = open('vectorizer.pickle', 'rb')
tf_vect = pickle.load(vec)

X_test_tf = tf_vect.transform(review_test)
pred = clf.predict(X_test_tf)

print(metrics.accuracy_score(label_test, pred))


print(confusion_matrix(label_test, pred))


print(classification_report(label_test, pred))



print("=" * 40)
print("==========")
print("Classification Report : ",(classification_report(label_test, pred)))
print("Accuracy : ",accuracy_score(label_test, pred)*100)
accuracy = accuracy_score(label_test, pred)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
ACC = (accuracy_score(label_test, pred) * 100)
repo = (classification_report(label_test, pred))
```

```
    label4 = tk.Label(root,text
=str(repo),width=35,height=10,bg='khaki',fg='black',font=("Tempus Sanc ITC",14))

    label4.place(x=205,y=100)


    label5 = tk.Label(root,text ="Accracy : "+str(ACC)+"%\nModel saved as
SVM_MODEL.joblib",width=35,height=3,bg='khaki',fg='black',font=("Tempus Sanc
ITC",14))

    label5.place(x=205,y=320)


    dump (clf,"SVM_MODEL.joblib")

    print("Model saved as SVM_MODEL.joblib")


frame = tk.LabelFrame(root,text="Control
Panel",width=600,height=400,bd=3,background="#F67280",font=("Tempus Sanc
ITC",15,"bold"))

frame.place(x=600,y=200)


entry = tk.Entry(frame,width=50)

entry.insert(0,"Enter text here...")

entry.place(x=25,y=150)


def Test():

    predictor = load("SVM_MODEL.joblib")

    Given_text = entry.get()

    #Given_text = "the 'roseanne' revival catches up to our thorny po..."

    vec = open('vectorizer.pickle', 'rb')

    tf_vect = pickle.load(vec)

    X_test_tf = tf_vect.transform([Given_text])

    y_predict = predictor.predict(X_test_tf)

    print(y_predict[0])

    if y_predict[0]==0:
        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Feeling,Judging",width=70,height=2,bg='Green',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==1:

        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Intuition,Thinking,Perceiving",width=70,height=2,bg='Red',fg='black',fo
nt=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==2:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Thinking,Perceiving",width=70,height=2,bg='Red',fg='black',font
=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==3:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==4:

        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Intuition,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font
=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==5:

        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Intuition,Feeling,Judging",width=70,height=2,bg='Red',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==6:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Feeling,Perceiving",width=70,height=2,bg='Red',fg='black',font=
("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==7:
        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Intuition,Feeling,Perceiving",width=70,eight=2,bg='Red',fg='black',font=
("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==8:

        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Intuition,Feeling,Perceiving",width=70,height=2,bg='Red',fg='black',font
=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==9:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Sensing,Thinking,Perceiving",width=70,height=2,bg='Red',fg='black',font
=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==10:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Sensing,Feeling,Judging",width=70,height=2,bg='Red',fg='black',font=("T
empus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==11:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Sensing,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font=("
Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==12:

        label4 = tk.Label(root,text ="Personality Prediction is
Extroversion,Sensing,Thinking,Perceiving",width=70,height=2,bg='Red',fg='black',fo
nt=("Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==13:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==14:
        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)

    elif y_predict[0]==15:

        label4 = tk.Label(root,text ="Personality Prediction is
Introvrsion,Intuition,Thinking,Judging",width=70,height=2,bg='Red',fg='black',font=(
"Tempus Sanc ITC",14))

        label4.place(x=600,y=800)



def window():

    root.destroy()




# =
tk.Button(frame,command=Train,text="Train",bg="red",fg="black",width=15,font=("
Times New Roman",15,"italic"))

#button2.place(x=5,y=100)


button3 =
tk.Button(frame,command=Test,text="Test",bg="green",fg="black",width=15,font=("
Times New Roman",20,"italic"))

button3.place(x=50,y=250)


exit = tk.Button(root, text="Exit", command=window, width=15, height=2,
font=('times', 15, ' bold '),bg="red",fg="white")

exit.place(x=800, y=650)



root.mainloop()
```

# CHAPTER 8

# RESULT

## 8.1 SCREENSOTS

### 8.1.1 Pre-processing

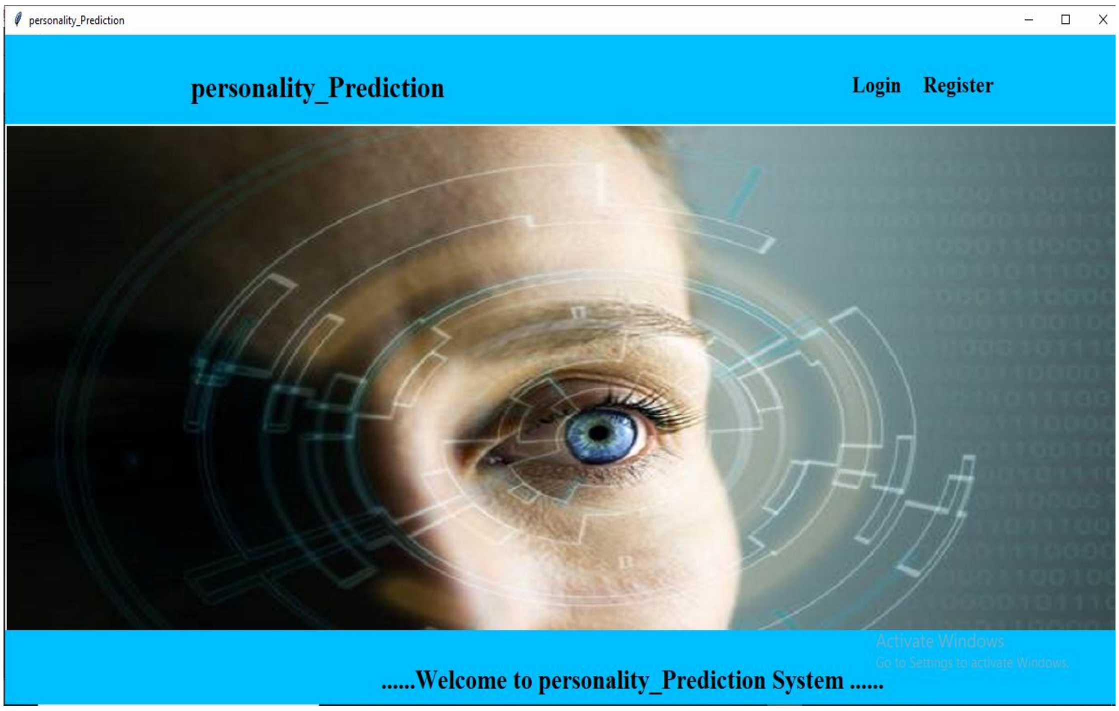
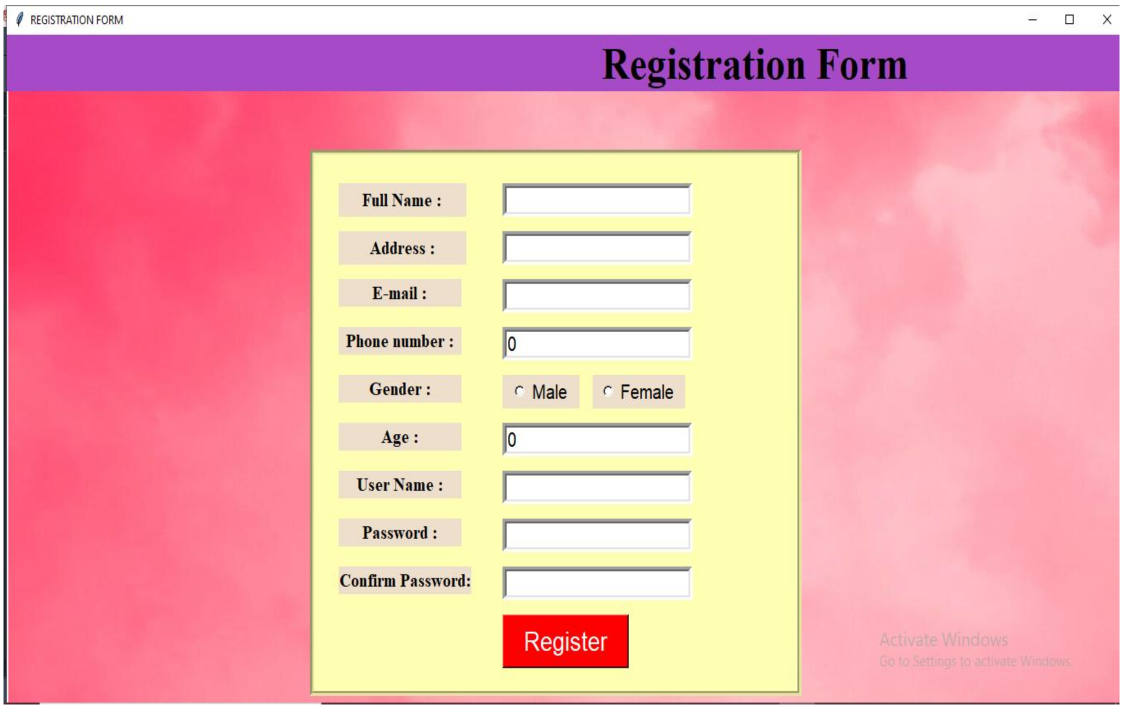

personality_Prediction
personality_Prediction
Login
Register
......Welcome to personality_Prediction System ......
Activate Windows
Go to Settings to activate Windows.

*Figure 8.1: First page*



| | |
|---|---|
| Full Name : | |
| Address : | |
| E-mail : | |
| Phone number : | 0 |
| Gender : | ○ Male ○ Female |
| Age : | 0 |
| User Name : | |
| Password : | |
| Confirm Password: | |

*Figure 8.2: Registration Form*

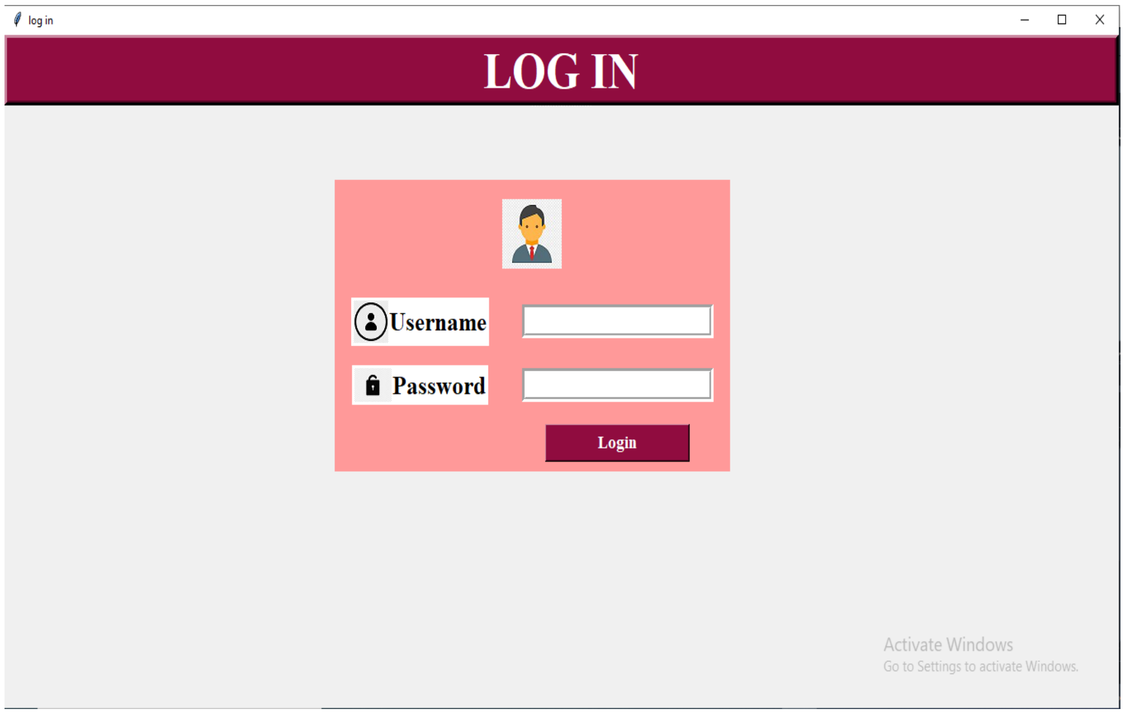


*Figure 8.3: Login Form*

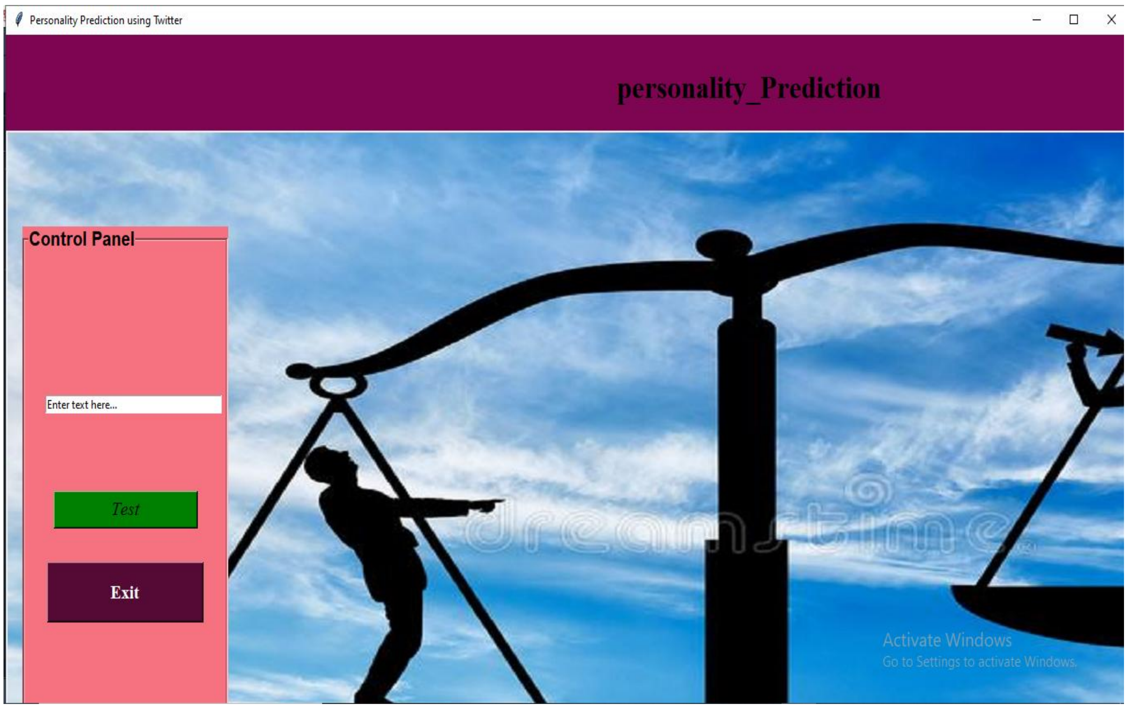
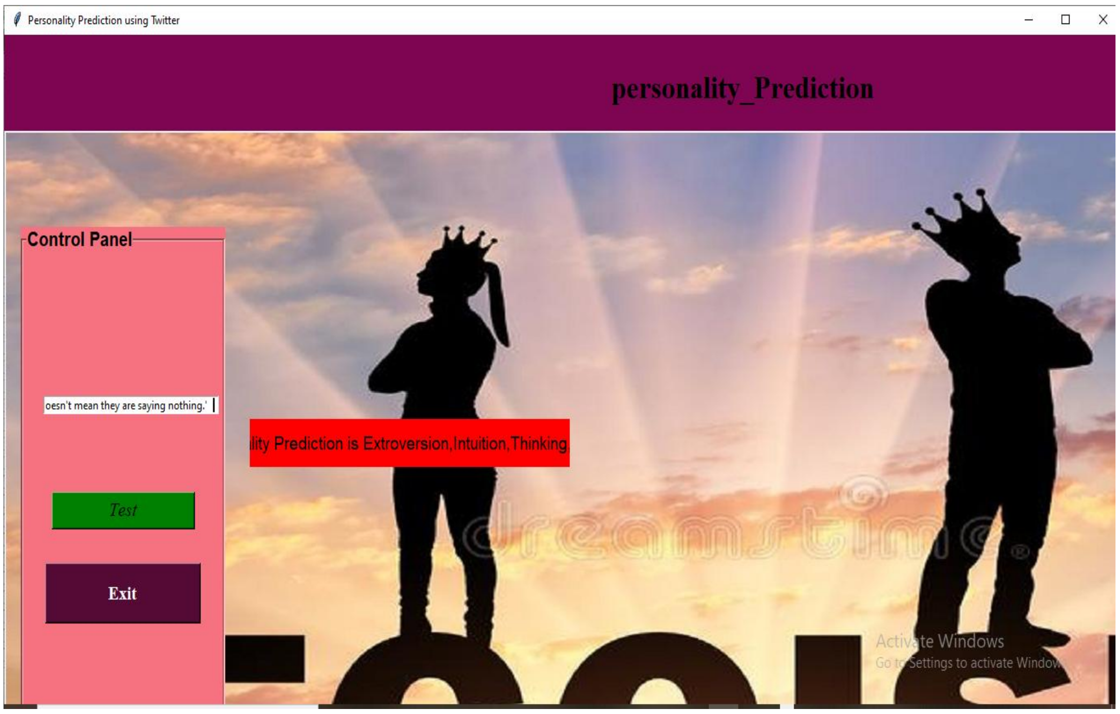


*Figure 8.4: Preprocessing Page*


Personality Prediction using Twitter
personality_Prediction
Control Panel
oesn't mean they are saying nothing.'
Test
Exit
Activate Windows
Go to Settings to activate Windows

lity Prediction is Extroversion,Intuition,Thinking,

*Figure 8.5: Output page*

# CHAPTER 9

# CONCLUSION

As we know, the base of social media is basically a graph. The connections between the nodes and impact on them due to social media interactions could be reflected through the SN features.

We have designed and performed experiments utilizing the linguistic features as well as SN features. To the best of our knowledge, we have used the most number of features to compare the performance of the feature selection algorithms to predict personality traits.

# CHAPTER 10

# REFERENCES

M. R. Barrick and M. K. Mount, “The big five personality dimensions and job performance: A meta-analysis,” Personnel Psychology, vol. 44, no. 1, pp.1–26, 1991. [Online].Available:https://onlinelibrary.wiley.com/doi/abs/10.1111/j.17446570.1991.tb00688.x[1]

S. Rothmann and E. P. Coetzer, “The big five personality dimensions and job performance,” SA Journal of Industrial Psychology, vol. 29, no. 1, pp. 68–74,2003. [Online]. Available: https://journals.co.za/content/psyc/29/1/EJC88938[2]

J. F. Salgado, “The big five personality dimensions and counterproductive behaviors,” International Journal of Selection and Assessment, vol. 10, no. 1, pp.117–125, 2002. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/1468-2389.00198[3]

J. W. Lounsbury, R. P. Steel, L. W. Gibson, and A. W. Drost, “Personality traits and career satisfaction of human resource professionals,” Human Resource Development International, vol. 11, no. 4, pp. 351–366, 2008. [Online]. Available: https://doi.org/10.1080/13678860802261215[4]

J. W. Lounsbury, N. Foster, H. Patel, P. Carmody, L. W. Gibson, and D. R. Stairs, “An investigation of the personality traits of scientists versus non-scientists and their relationship with career satisfaction,” RD Management, vol. 42,no. 1, pp. 47–59, 2012.[Online].Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/9310.2011.00665.x[5]

V. Ariyabuddhiphongs and S. Marican, “Big five personality traits and turnover intention among thai hotel employees,” International Journal of Hospitality Tourism Administration, vol. 16, no. 4, pp. 355–374, 2015. [Online]. Available: https://doi.org/10.1080/15256480.2015.1090257[6]

T. A. Timmerman, “Predicting turnover with broad and narrow personality traits,” International Journal of Selection and Assessment, vol. 14, no. 4, pp.392–399, 2006. [Online]. https://onlinelibrary.wiley.com/doi/abs/10.1111/j.14682389.2006.00361.x[7]

P. T. Costa Jr. and R. R. McCrae, “The Revised NEO Personality Inventory

(NEO-PI-R).” in The SAGE handbook of personality theory and assessment,

Vol 2: Personality measurement and testing. Thousand Oak, CA, US: Sage Publications, Inc, 2008, pp. 179–198.[8]

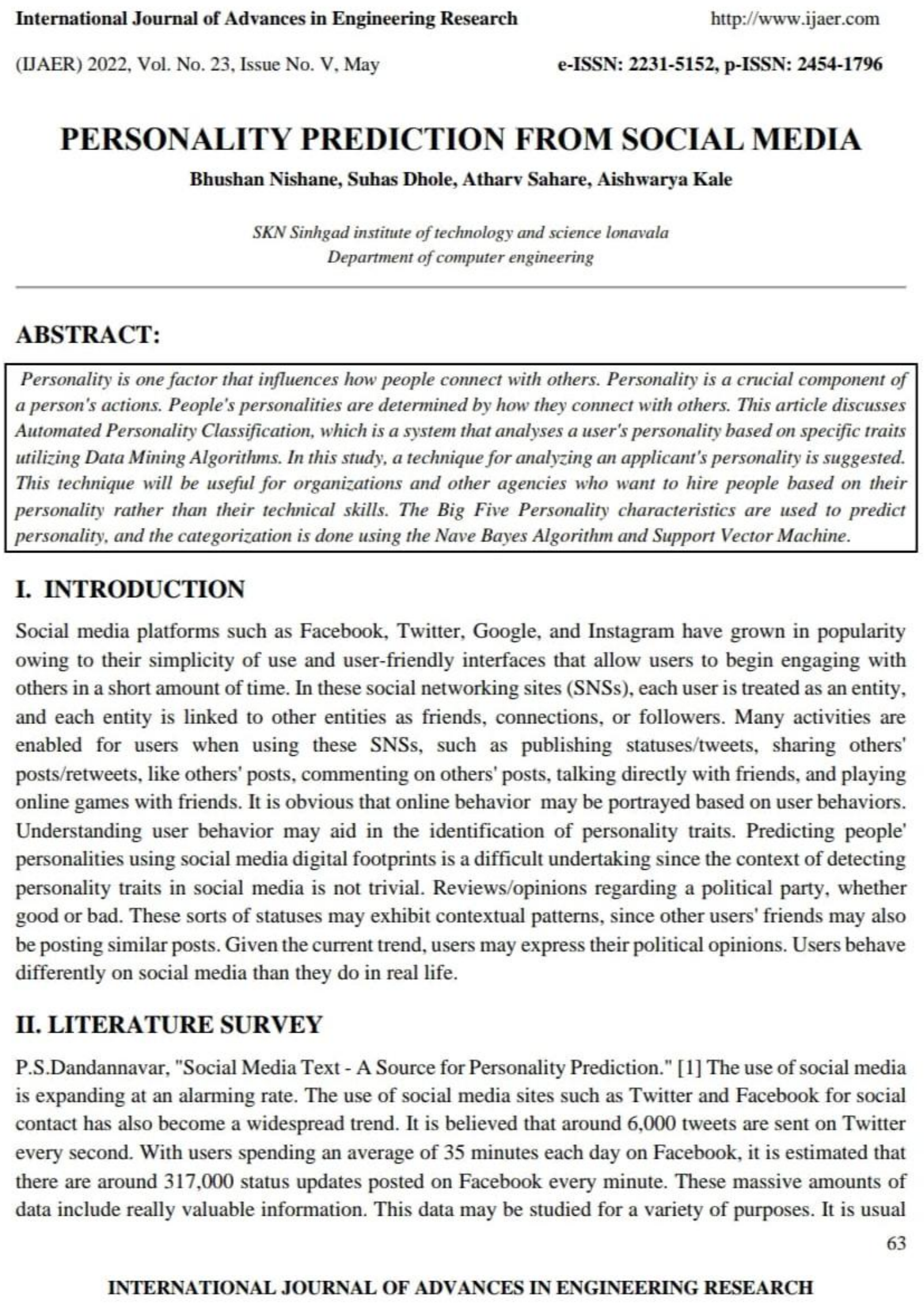
K. Lee and M. C. Ashton, “Psychometric properties of the hexaco-100,” Assessment, vol. 25, no. 5, pp. 543–556, 2018.

[Online]. Available: https://doi.org/10.1177/107[9]

N. Anderson, J. Salgado, and U. Hu¨lsheger, “Applicant reactions in selection: Comprehensive meta-analysis into reaction generalization versus situational specificity,” International Journal of Selection and Assessment, vol. 19, pp. 291–304, 08 2010[10]

# CHAPTER 11

# Published paper copy and certificate



International Journal of Advances in Engineering Research http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. V, May **e-ISSN: 2231-5152, p-ISSN: 2454-1796**

# PERSONALITY PREDICTION FROM SOCIAL MEDIA

**Bhushan Nishane, Suhas Dhole, Atharv Sahare, Aishwarya Kale**

*SKN Sinhgad institute of technology and science lonavala*
*Department of computer engineering*

## ABSTRACT:

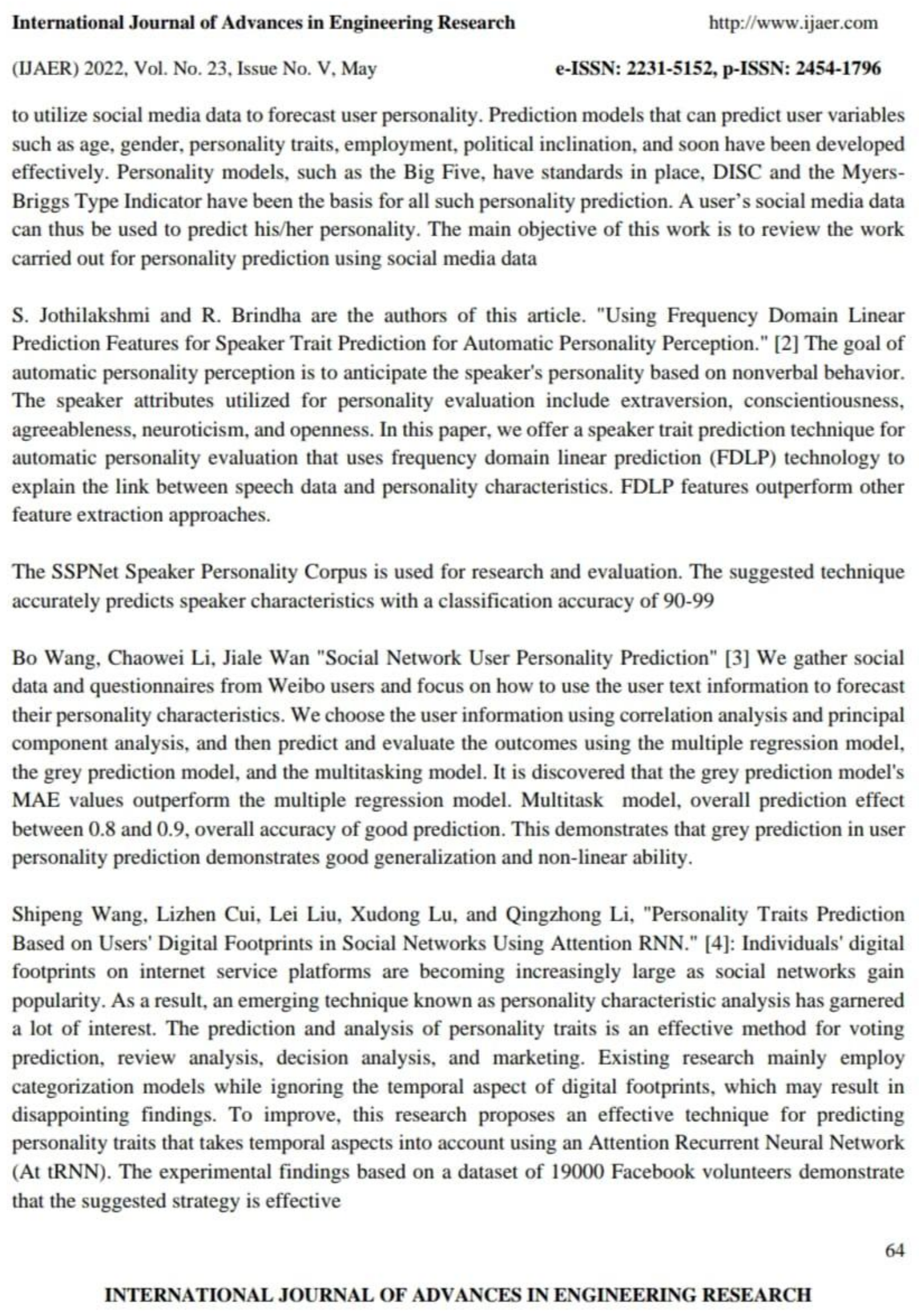
*Personality is one factor that influences how people connect with others. Personality is a crucial component of a person's actions. People's personalities are determined by how they connect with others. This article discusses Automated Personality Classification, which is a system that analyses a user's personality based on specific traits utilizing Data Mining Algorithms. In this study, a technique for analyzing an applicant's personality is suggested. This technique will be useful for organizations and other agencies who want to hire people based on their personality rather than their technical skills. The Big Five Personality characteristics are used to predict personality, and the categorization is done using the Nave Bayes Algorithm and Support Vector Machine.*

## I. INTRODUCTION

Social media platforms such as Facebook, Twitter, Google, and Instagram have grown in popularity owing to their simplicity of use and user-friendly interfaces that allow users to begin engaging with others in a short amount of time. In these social networking sites (SNSs), each user is treated as an entity, and each entity is linked to other entities as friends, connections, or followers. Many activities are enabled for users when using these SNSs, such as publishing statuses/tweets, sharing others' posts/retweets, like others' posts, commenting on others' posts, talking directly with friends, and playing online games with friends. It is obvious that online behavior may be portrayed based on user behaviors. Understanding user behavior may aid in the identification of personality traits. Predicting people' personalities using social media digital footprints is a difficult undertaking since the context of detecting personality traits in social media is not trivial. Reviews/opinions regarding a political party, whether good or bad. These sorts of statuses may exhibit contextual patterns, since other users' friends may also be posting similar posts. Given the current trend, users may express their political opinions. Users behave differently on social media than they do in real life.

## II. LITERATURE SURVEY

P.S.Dandannavar, "Social Media Text - A Source for Personality Prediction." [1] The use of social media is expanding at an alarming rate. The use of social media sites such as Twitter and Facebook for social contact has also become a widespread trend. It is believed that around 6,000 tweets are sent on Twitter every second. With users spending an average of 35 minutes each day on Facebook, it is estimated that there are around 317,000 status updates posted on Facebook every minute. These massive amounts of data include really valuable information. This data may be studied for a variety of purposes. It is usual



**International Journal of Advances in Engineering Research** http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. V, May **e-ISSN: 2231-5152, p-ISSN: 2454-1796**

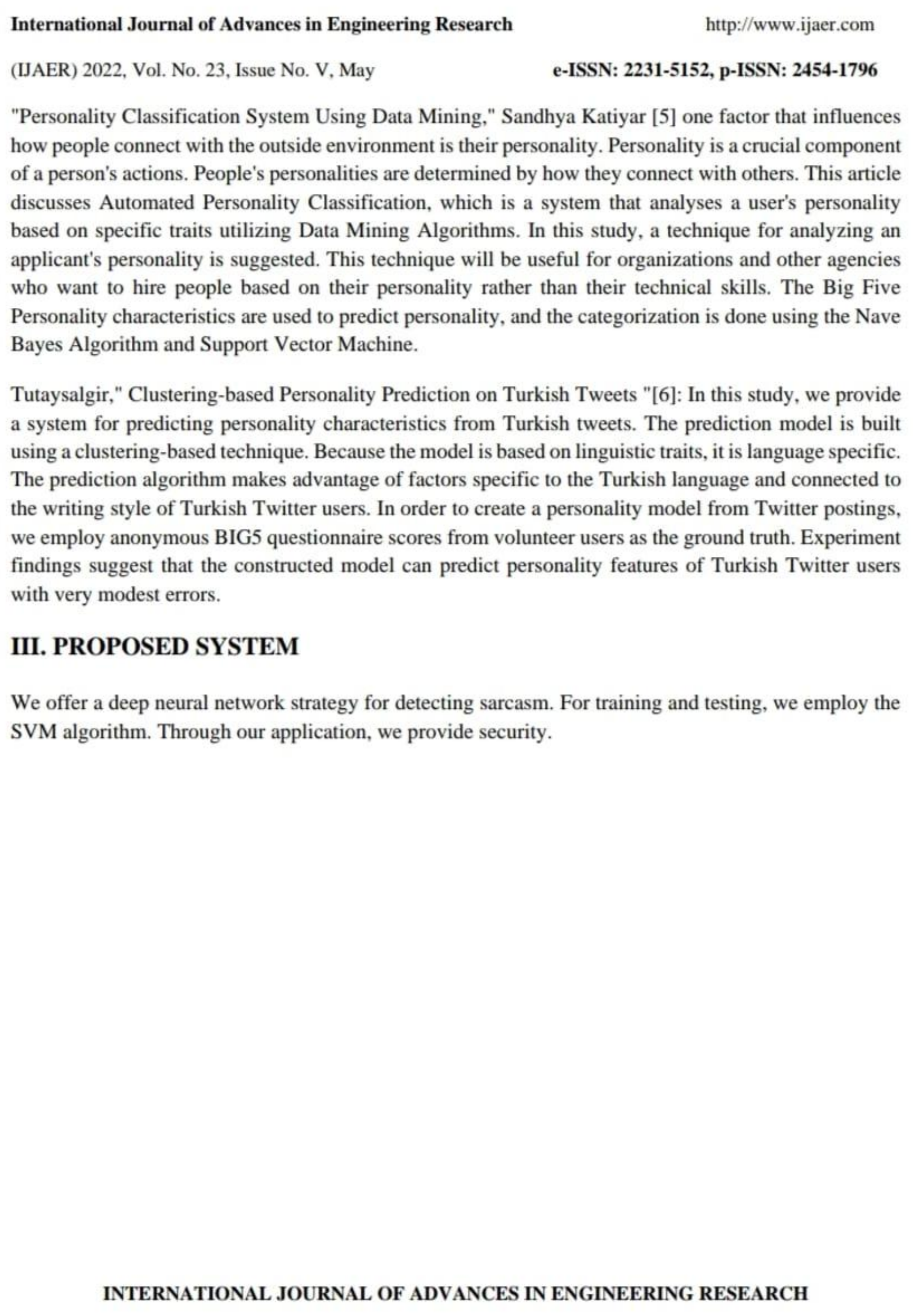
to utilize social media data to forecast user personality. Prediction models that can predict user variables such as age, gender, personality traits, employment, political inclination, and soon have been developed effectively. Personality models, such as the Big Five, have standards in place, DISC and the Myers-Briggs Type Indicator have been the basis for all such personality prediction. A user's social media data can thus be used to predict his/her personality. The main objective of this work is to review the work carried out for personality prediction using social media data

S. Jothilakshmi and R. Brindha are the authors of this article. "Using Frequency Domain Linear Prediction Features for Speaker Trait Prediction for Automatic Personality Perception." [2] The goal of automatic personality perception is to anticipate the speaker's personality based on nonverbal behavior. The speaker attributes utilized for personality evaluation include extraversion, conscientiousness, agreeableness, neuroticism, and openness. In this paper, we offer a speaker trait prediction technique for automatic personality evaluation that uses frequency domain linear prediction (FDLP) technology to explain the link between speech data and personality characteristics. FDLP features outperform other feature extraction approaches.

The SSPNet Speaker Personality Corpus is used for research and evaluation. The suggested technique accurately predicts speaker characteristics with a classification accuracy of 90-99

Bo Wang, Chaowei Li, Jiale Wan "Social Network User Personality Prediction" [3] We gather social data and questionnaires from Weibo users and focus on how to use the user text information to forecast their personality characteristics. We choose the user information using correlation analysis and principal component analysis, and then predict and evaluate the outcomes using the multiple regression model, the grey prediction model, and the multitasking model. It is discovered that the grey prediction model's MAE values outperform the multiple regression model. Multitask model, overall prediction effect between 0.8 and 0.9, overall accuracy of good prediction. This demonstrates that grey prediction in user personality prediction demonstrates good generalization and non-linear ability.

Shipeng Wang, Lizhen Cui, Lei Liu, Xudong Lu, and Qingzhong Li, "Personality Traits Prediction Based on Users' Digital Footprints in Social Networks Using Attention RNN." [4]: Individuals' digital footprints on internet service platforms are becoming increasingly large as social networks gain popularity. As a result, an emerging technique known as personality characteristic analysis has garnered a lot of interest. The prediction and analysis of personality traits is an effective method for voting prediction, review analysis, decision analysis, and marketing. Existing research mainly employ categorization models while ignoring the temporal aspect of digital footprints, which may result in disappointing findings. To improve, this research proposes an effective technique for predicting personality traits that takes temporal aspects into account using an Attention Recurrent Neural Network (At tRNN). The experimental findings based on a dataset of 19000 Facebook volunteers demonstrate that the suggested strategy is effective



**International Journal of Advances in Engineering Research** http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. V, May **e-ISSN: 2231-5152, p-ISSN: 2454-1796**

"Personality Classification System Using Data Mining," Sandhya Katiyar [5] one factor that influences how people connect with the outside environment is their personality. Personality is a crucial component of a person's actions. People's personalities are determined by how they connect with others. This article discusses Automated Personality Classification, which is a system that analyses a user's personality based on specific traits utilizing Data Mining Algorithms. In this study, a technique for analyzing an applicant's personality is suggested. This technique will be useful for organizations and other agencies who want to hire people based on their personality rather than their technical skills. The Big Five Personality characteristics are used to predict personality, and the categorization is done using the Nave Bayes Algorithm and Support Vector Machine.

Tutaysalgir," Clustering-based Personality Prediction on Turkish Tweets "[6]: In this study, we provide a system for predicting personality characteristics from Turkish tweets. The prediction model is built using a clustering-based technique. Because the model is based on linguistic traits, it is language specific. The prediction algorithm makes advantage of factors specific to the Turkish language and connected to the writing style of Turkish Twitter users. In order to create a personality model from Twitter postings, we employ anonymous BIG5 questionnaire scores from volunteer users as the ground truth. Experiment findings suggest that the constructed model can predict personality features of Turkish Twitter users with very modest errors.

## III. PROPOSED SYSTEM

We offer a deep neural network strategy for detecting sarcasm. For training and testing, we employ the SVM algorithm. Through our application, we provide security.

International Journal of Advances in Engineering Research http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. V, May e-ISSN: 2231-5152, p-ISSN: 2454-1796

**SYSTEM ARCHITECTURE**

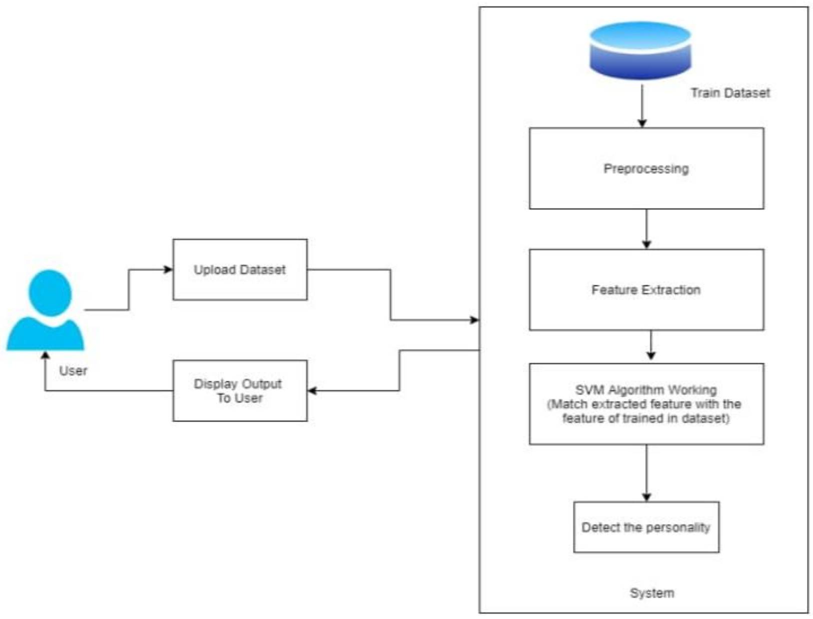
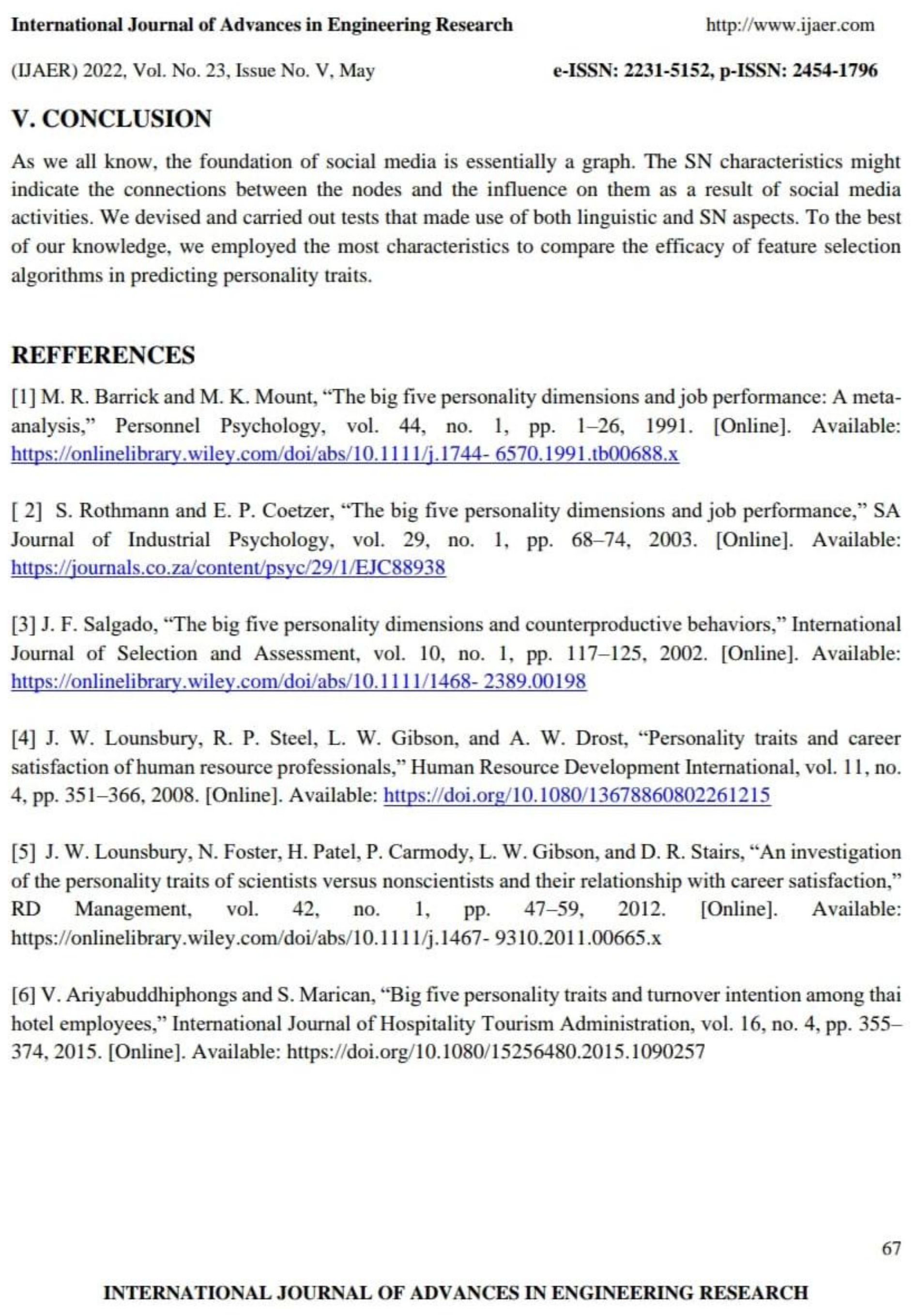


Fig. System Architecture

## IV. ALGORITHM

Support Vector Machine, or SVM, is a prominent Supervised Learning technique that is used for both classification and regression issues. However, it is mostly utilized in Machine Learning for Classification difficulties. The SVM algorithm's purpose is to find the optimum line or decision boundary for categorizing n-dimensional space so that we may simply place fresh data points in the proper category in the future. A hyperplane is the optimal choice boundary. SVM selects the extreme points/vectors that aid in the creation of the hyperplane. These extreme examples are referred to as support vectors, and the technique is known as the Support Vector Machine. Consider the diagram below, which shows two distinct categories that are separated by a decision boundary or hyperplane.



International Journal of Advances in Engineering Research http://www.ijaer.com

(IJAER) 2022, Vol. No. 23, Issue No. V, May e-ISSN: 2231-5152, p-ISSN: 2454-1796

## V. CONCLUSION

As we all know, the foundation of social media is essentially a graph. The SN characteristics might indicate the connections between the nodes and the influence on them as a result of social media activities. We devised and carried out tests that made use of both linguistic and SN aspects. To the best of our knowledge, we employed the most characteristics to compare the efficacy of feature selection algorithms in predicting personality traits.

## REFFERENCES

[1] M. R. Barrick and M. K. Mount, "The big five personality dimensions and job performance: A meta-analysis," Personnel Psychology, vol. 44, no. 1, pp. 1–26, 1991. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1744- 6570.1991.tb00688.x

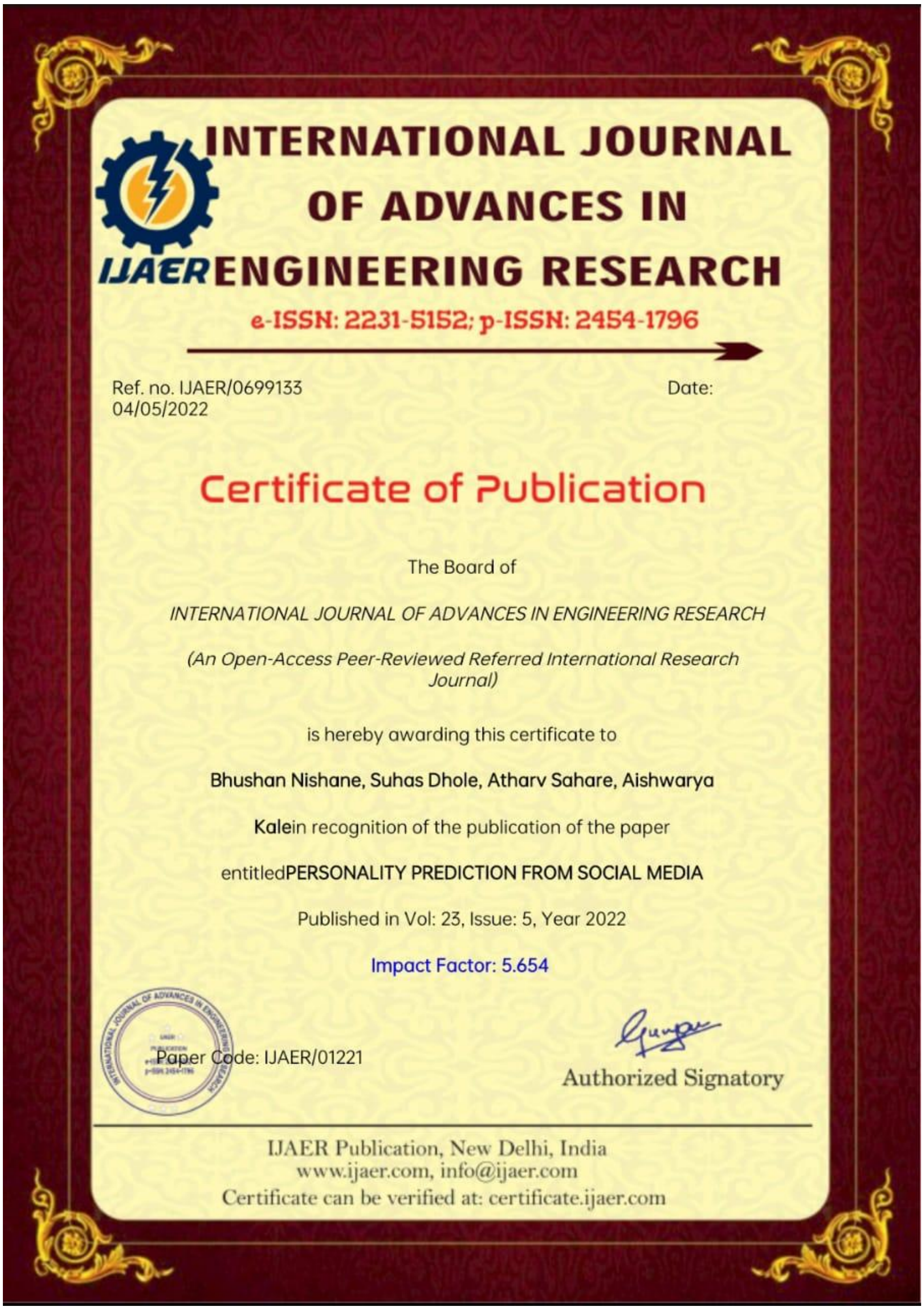
[ 2] S. Rothmann and E. P. Coetzer, "The big five personality dimensions and job performance," SA Journal of Industrial Psychology, vol. 29, no. 1, pp. 68–74, 2003. [Online]. Available: https://journals.co.za/content/psyc/29/1/EJC88938

[3] J. F. Salgado, "The big five personality dimensions and counterproductive behaviors," International Journal of Selection and Assessment, vol. 10, no. 1, pp. 117–125, 2002. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/1468- 2389.00198

[4] J. W. Lounsbury, R. P. Steel, L. W. Gibson, and A. W. Drost, "Personality traits and career satisfaction of human resource professionals," Human Resource Development International, vol. 11, no. 4, pp. 351–366, 2008. [Online]. Available: https://doi.org/10.1080/13678860802261215

[5] J. W. Lounsbury, N. Foster, H. Patel, P. Carmody, L. W. Gibson, and D. R. Stairs, "An investigation of the personality traits of scientists versus nonscientists and their relationship with career satisfaction," RD Management, vol. 42, no. 1, pp. 47–59, 2012. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467- 9310.2011.00665.x

[6] V. Ariyabuddhiphongs and S. Marican, "Big five personality traits and turnover intention among thai hotel employees," International Journal of Hospitality Tourism Administration, vol. 16, no. 4, pp. 355–374, 2015. [Online]. Available: https://doi.org/10.1080/15256480.2015.1090257



INTERNATIONAL JOURNAL OF ADVANCES IN ENGINEERING RESEARCH

IJAER

e-ISSN: 2231-5152; p-ISSN: 2454-1796

Ref. no. IJAER/0699133 Date: 04/05/2022

Certificate of Publication

The Board of

*INTERNATIONAL JOURNAL OF ADVANCES IN ENGINEERING RESEARCH*

*(An Open-Access Peer-Reviewed Referred International Research Journal)*

is hereby awarding this certificate to

**Bhushan Nishane, Suhas Dhole, Atharv Sahare, Aishwarya Kale**in recognition of the publication of the paper

entitled**PERSONALITY PREDICTION FROM SOCIAL MEDIA**

Published in Vol: 23, Issue: 5, Year 2022

Impact Factor: 5.654

INTERNATIONAL JOURNAL OF ADVANCES IN ENGINEERING RESEARCH

Paper Code: IJAER/01221

[illegible]

Authorized Signatory

IJAER Publication, New Delhi, India
www.ijaer.com, info@ijaer.com
Certificate can be verified at: certificate.ijaer.com