Personality Traits Analysis

Submitted by

Sanjeev Pratap (13000120063)

Sudakshina Majumdar (13000120078)

Suman Raj (13000120080)

Sumana Sen (13000120081)

7th Semester

**September, 2023**

Submitted for the partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering



Techno Main Salt Lake

EM 4/1, Salt Lake, Sector – V, Kolkata – 700 091.

**ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to Prof. Nairanjana Chowdhury of the department of Computer Science and Engineering, whose role as project guide was invaluable for the project. We are extremely thankful for the keen interest he / she took in advising us, for the books and reference materials provided for the moral support extended to us.

Last but not the least we convey our gratitude to all the teachers for providing us the technical skill that will always remain as our asset and to all non-teaching staff for the gracious hospitality they offered us.

Place: Techno Main Salt Lake

Date: 25 September, 2023

Sanjeev Pratap (13000120063)

Sudakshina Majumdar (13000120078)

Suman Raj (13000120080)

Sumana Sen (13000120081)

Department of Computer Science and Engineering

Techno Main Salt Lake

Kolkata – 700 091

West Bengal, India.

**APPROVAL**

This is to certify that the project report entitled “Personality Traits Analysis” prepared under my supervision by **Sanjeev Pratap** (13000120063), **Sudakshina Majumdar** (13000120078), **Suman Raj** (13000120080) and **Sumana Sen** (13000120081) be accepted in partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering.

It is to be understood that by this approval, the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn thereof, but approves the report only for the purpose for which it has been submitted.

|  |  |
| --- | --- |
| ………………………………………..…  Signature, Name & Designation of  Internal Guide(s) | …………………………………………  Signature, Name & Designation  of the HOD |
|  |  |
| ………………………………………..…  Signature of External Examiner |  |

Table of Contents

[1 Introduction 2](#_Toc147329646)

[1.1 Abstract 2](#_Toc147329647)

[1.2 Problem Domain 3](#_Toc147329648)

[1.3 Glossary 3](#_Toc147329649)

[2 Problem Definition 4](#_Toc147329650)

[2.1 Scope 4](#_Toc147329651)

[2.2 Exclusions 4](#_Toc147329652)

[2.3 Assumptions 4](#_Toc147329653)

[3 Related Studies 5](#_Toc147329654)

[4 Project Planning 5](#_Toc147329655)

[4.1 Software Life Cycle Model 5](#_Toc147329656)

[4.2 Scheduling 6](#_Toc147329657)

[4.3 Cost Analysis 8](#_Toc147329658)

[5 Requirement Analysis 10](#_Toc147329659)

[5.1 Requirement Matrix 10](#_Toc147329660)

[5.2 Requirement Elaboration 10](#_Toc147329661)

[5.2.3 Prediction dashboard 14](#_Toc147329662)

[6 Design 17](#_Toc147329663)

[6.1 Technical Environment 17](#_Toc147329664)

[6.2 Hierarchy of Modules 17](#_Toc147329668)

[6.3 Detailed Design 19](#_Toc147329669)

[7 Test Plan 23](#_Toc147329670)

[8 Conclusion 26](#_Toc147329671)

[9 References / Bibliography 27](#_Toc147329672)

[10 APPENDIX A – Prototypes 27](#_Toc147329673)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| IntroductionAbstract Introduction: Social networking sites plays an important role of our everyday life and users are now free to choose from various social networking platforms. In 2017, more than half of the globe uses the world wide web network with 2.7 billion live social media accounts world widely [1] and every user of that social platforms leaves a mark as digital footprint and tend to present themselves in a type of behavior usually determined by their personality traits. With images getting more popularity especially among younger people [2] and recent social networks are focusing on visual content such as Snapchat or Instagram, personality dimensions in this case can be calculated by running content analysis which is based solely on images as presented in previous researches [3]. Images, in general, contain multiple variables such as scenes, compositions, colors, emotions, facial presentations, and facial expressions where these properties can be fetched by leveraging various computer vision algorithms such as [4].  In this project, we investigate how social media profile pictures differ based on the personality of the users posting them at their social networking sites. We use profile images from Twitter, Facebook platform whose personality we predicted based on data points. We conducted our analysis on users faces by extracting unique facial features in order to examine the relationship between personality and profile picture. Finally, various machine learning approaches were investigated to test the effectiveness of these facial features in predicting users’ psychological traits. In our project, Big Five Factor and Myers Briggs Type Indicator (MBTI) personality prediction analysis are used to train the model.  Problem Statement: Personality traits analysis revolves around understanding and categorizing the unique characteristics, behaviors, and tendencies exhibited by individuals across various contexts. This analysis aims to analyze the impact of these traits on decision-making, behavior, relationships, and overall personal development.  Solution:  1. Data Collection: Gather data from social media platforms, such as Twitter, Facebook, or Instagram. This data should include users' posts, comments, likes, and other interactions.  Utilize APIs or web scraping tools to collect historical social media data, ensuring compliance with platform terms of service and privacy regulations.  2. Data Pre-processing: Clean and preprocess the collected data to remove noise, handle missing values, and standardize text data.  3. Extract relevant features from social media activities include Linguistic features from text data, such as sentiment analysis, word frequency, and topic modeling.  4. Personality Traits prediction Models: Develop machine learning models capable of predicting personality traits based on the engineered features. Choose appropriate algorithms, such as regression, decision trees, random forests, or neural networks, for personality prediction.  Excepted Result:  1. Insights into User Behavior: Gain a deeper understanding of how individuals manifest their personality traits through social media interactions, including the content they post, the language they use, and their online engagement patterns. Uncover correlations and patterns between specific personality traits and social media behaviors.  2. Model Performance: Evaluate model performance using appropriate metrics, such as mean squared error or classification accuracy. Fine-tune models to enhance prediction accuracy and generalizability.  3. Applications and Insights: Explore potential applications of personality trait analysis derived from social media data, such as targeted content recommendations, mental health assessments, personalized marketing, and social science research.  Project Benefits:  This project holds immense usefulness like Personal growth and development, Interpersonal relationships, Organizational productivity and Effective treatment planning and emotional support. Problem Domain This project lies within the domain of Web-development using Machine learning and Deep learning. The personality traits are observed through social media activities. Analysing the personality patterns, their perspectives, opinions, sensitivity, and judgement can be predicted that helps in fields like interviews, new product launch or even preventing suicide and self-harm. Glossary  |  |  |  | | --- | --- | --- | | S No. | Term | Description | | 1 | Social Networking Sites | Online platforms that allow users to create profiles, connect with others, and share content and interactions. | | 2 | Digital Footprint | The trail of data and information left by a person's online activities, including social media interactions. | | 3 | Personality Traits | Unique psychological characteristics, behaviors, and tendencies that define an individual's personality. | | 4 | Linguistic Features | Characteristics related to language and text, such as sentiment analysis, word frequency, and topic modeling. | | 5 | Mean Squared Error (MSE) | A metric used to measure the average squared difference between predicted and actual values in regression tasks. | | 6 | Real-time Analysis | Continuous monitoring and analysis of data to provide insights and adapt to changing behaviors and circumstances. | | 8 | Classification Accuracy | A metric used to measure the percentage of correctly classified instances in classification tasks. |  Problem DefinitionScope  * Corporate Sectors are prioritizing personality traits analysis for optimizing team dynamics, developing leadership to deliver good products, and employee satisfaction by providing them with what they need. * Education Industries like schools, colleges, and other educational platforms can use personality traits analysis to provide personalized learning experiences. * Therapists and counselors use personality traits analysis for more effective treatment planning and improvement of mental health. * Companies can use personality traits analysis in recruitment and HR processes to match candidates with job roles, leading to improved employee retention. * Businesses can tap into personality traits analysis for targeted marketing strategies, consumer insights and product development.  Exclusions  * Private Data: The project will not access or analyze any private or restricted data from social media platforms. Only publicly available data will be considered. * Real-time Analysis: The project will not focus on real-time personality analysis or adaptability to changing behaviors on social media. It will primarily analyze historical data. * In-depth Sentiment Analysis: While linguistic features will be considered, in-depth sentiment analysis, including sentiment towards specific topics or entities, is outside the scope of this project. * Ethical and Privacy Considerations: The project will not address the ethical and privacy implications of personality analysis, such as the potential for bias or invasions of privacy. These considerations should be addressed separately.  Assumptions Many factors are taken into account in the project assumption. The list below indicates the major milestones that have been scheduled:   * The project deadline of 24th May, 2024. * The final presentation is on 23rd May, 2024. * The peer evaluation deadline is on 22nd May, 2024. * The project assumes that the results of personality trait predictions can be interpreted and translated into actionable insights, even though the complexity of human personality may limit the precision of such predictions. * The project assumes that machine learning models can achieve a reasonable level of accuracy in predicting personality traits based on social media activities.   NOTE: Since the deadline of project is 24thMay, 2024, running out of time will have its reflection on the product, and submission schedule can’t be revised. All possible measures will be taken to finish the project as per schedule. Related Studies The prevalence of social platforms triggered multiple research endeavors in human personality estimation and prediction. In this part, we summarize recent research efforts in  predicting individuals’ personality from appearance. Liu et al. [16] presented a large-scale analysis of profile images and personality at Twitter microblogging platform. They used a wider range of interpretable aesthetic and facial features to capture correlations with the personality that is in line and complement psychological research. Skowron et al. [20] proposed a unique technique that combines multiple inputs as text, image, meta-features and integrates it out of two different social networking sites which are Instagram and Twitter. The superior outcomes for every personality trait are achieved by blending engineered features derived from jointly social networking sites.  Cristani et al. [22] investigated the rising size of multimedia information users generate and engage online and consider it as a probable contributing factor to our what so-called online appearance. The paper also confers that visual patterns correlates with personality score and can be used to predict personality where also they found that the favorite images users assign in his/her profile can be used eventually to build prediction models to estimate their preserved online personality. Project PlanningSoftware Life Cycle Model The spiral model process will be used for this project.  Spiral Model in Software Life Cycle Model (SDLC)  **Spiral Model** is a type of **Software Development Model** in which activities are created in a spiral and carried out in the order in which they are chosen. The Spiral Model focuses on risk assessment. As a result, any team looking to use this model must have people with knowledge and skills in this area.  This model has four phases dividing the model into quadrants: planning and determining objectives, analysing and resolving risks, engineering and testing, evaluation and plan next iteration. The number of loops in the spiral depends on the specific project and the project manager’s discretion.  Here, we are using spiral model for the following reasons:   * It is desirable to have frequent software releases. * Prototyping is used. * The requirement criteria are ambiguous and difficult to understand. * There is a lot of change going on, and it may happen at any time. * Whether for economic or other reasons, the long-term project commitment is compromised.  Scheduling        Cost Analysis   As the team size is small, the problem is well understood and also the team members have a nominal experience regarding the problem. That’s why we have chosen the BASIC COCOMO MODEL (ORGANIC).  Effort =a\*(KLOC)^ b PM  T dev =c\*(efforts)^d Months  Where, KLOC is the estimated size of the software product indicate in Kilo Lines of Code, a1, a2, b1, b2 are constants for each group of software products, T dev is the estimated time to develop the software, expressed in months, Effort is the total effort required to develop the software product, expressed in person months (PMs).   Requirement AnalysisRequirement Matrix  Requirement Elaboration DFD Level-0    DFD Level-1     * + 1. Verify user credentials * User Input: The system will receive user input, including a username and password. * Database Query: It will query a database where user credentials are stored. * Credential Validation: The system will compare the entered credentials with the stored data to verify their correctness. * Authentication: If the credentials match, the system will grant access to the user, allowing them to proceed. * Access Denied: If the credentials do not match or are invalid, the system will deny access and may display an error message.      * + - 1. Validate encrypted password * User Input: The system will receive the user's entered password. * Retrieve Stored Password: It will retrieve the previously stored encrypted password associated with the user's account. * Decryption: The system will decrypt the stored encrypted password. * Comparison: It will compare the decrypted stored password with the entered password. * Validation: If the entered password matches the stored password, the system will validate it as correct.   + - 1. User registration functionality * User Registration Form: The system will provide a registration form where users can enter their details. * User Information Input: Users will input their personal information such as name, email, password, and any other required details. * Data Validation: The system will validate the entered data, ensuring that all required fields are filled correctly. * Unique Email Check: It will check if the email provided by the user is unique and not already registered in the system. * Password Encryption: The system will securely encrypt the user's password to protect it. * User Profile Creation: Upon successful validation, the system will create a user profile with the provided information.   + 1. Create user profile * User Registration: When a user successfully registers on the system, a basic user profile is automatically created for them. * Default Profile: Initially, the user's profile will contain default information, such as a placeholder profile picture and basic personal details provided during registration (e.g., name, email). * Profile Completion: After registration, the user can choose to complete their profile by adding more information. This can include: * Uploading a profile picture or avatar. * Adding additional personal information, such as a bio, location, or contact details. Customizing profile settings, such as privacy preferences.      * + - 1. Options for manage and update profile * Profile Management Access: Users will have access to a "Profile Management" section within their account settings. * Edit Profile Information: Within the "Profile Management" section, users can choose to edit various aspects of their profile. This may include: * Basic Information: Users can update basic information such as their name, email address, or contact details. * Profile Picture: Users can change or update their profile picture or avatar. * Bio and Description: Users can modify their bio or add a description that provides more information about themselves. * Privacy Settings: Users can adjust privacy settings related to their profile, controlling who can view specific information. * Change Password: Users may have the option to change their account password within the profile management section for security purposes. * Save and Apply Changes: Once users make desired changes, they can save and apply them, making the updated profile information visible to others, if applicable.   .   * + - 1. user profile customization features * Public: Everyone can see their profile. * Connections: Only connected users can see it. * Private: No one else can see it.  Prediction dashboard  * Input Fields: The dashboard will have input fields for users to enter text-based data and upload image data for personality prediction. * Prediction Choice: Users can choose between text-based or image-based prediction by selecting the appropriate option. * Predict Button: A "Predict" button triggers the personality prediction process based on the user's input and choice. * Result Display: The dashboard displays the predicted personality traits clearly and concisely. * Additional Visualization: Users can click a "More Visualization" button to access additional visual representations of the personality predictions if needed.   + - 1. Interface for input text data * CSV File Upload: Users will have the option to upload a CSV file containing text data instead of manually entering text. This feature is useful for processing larger datasets. * File Format Validation: The system will first check if the uploaded file is in CSV format. If the file format is not CSV, it will display an error message to the user, indicating that only CSV files are accepted. * Data Extraction: Once a valid CSV file is uploaded, the system will extract the text data from the file. This may involve parsing the CSV structure and identifying the column(s) containing text data. * Data Preprocessing: Similar to manual text input, the extracted text data will undergo preprocessing steps, including stemming, stop words removal, and text cleaning, to prepare it for analysis. * Data Type Verification: The system will ensure that the extracted data is in text format and not any other data type. * Clear Submission: Users will have an option to proceed with the processed text data for personality prediction once the CSV file is successfully uploaded and processed.   + - 1. Interface for input image data * Image Upload: Users will have the option to upload image files directly for personality prediction. This feature allows users to provide images for analysis. * File Format Validation: The system will check if the uploaded files are in common image formats such as JPEG, PNG, or GIF. It will display an error message if the uploaded file is not in a supported image format. * Image Preprocessing: Once a valid image file is uploaded, the system may perform preprocessing steps on the image, such as resizing, normalization, or converting it to a suitable format for feature extraction. * Feature Extraction: After preprocessing, the system will extract relevant features from the uploaded image. This process may involve using a Convolutional Neural Network (CNN) or other image processing techniques. * Data Type Verification: The system will ensure that the extracted data represents image features and is compatible with the chosen prediction model. * Clear Submission: Users will have the option to proceed with the processed image data for personality prediction once the image is successfully uploaded and features are extracted.   + - 1. Choice to choose data type for prediction * Data Type Selection: Users will be presented with a clear and user-friendly interface that allows them to choose the data type they want to use for personality prediction. The available options will include: * Text Data * Image Data * Both Text and Image Data   + - 1. Implement personality prediction based on input data * Multimodal Capability: The system will support the implementation of personality prediction based on both text and image data. Users can choose to input either text, image, or both for the prediction process. * Data Preprocessing: Before prediction, the system will perform necessary data preprocessing steps. For text data, this may include tasks such as text cleaning, tokenization, stemming, and removing stop words. For image data, feature extraction techniques will be applied. * Model Selection: The system will employ machine learning classification algorithms for personality prediction. models including: * Naive Bayes (for text data) * BERT (for text data) * Convolutional Neural Network (CNN) (for image data) * Training and Testing: The selected model will undergo training using labeled data. The system will reserve a portion of the dataset for testing and evaluation to ensure the model's accuracy. * Multimodal Fusion: In the case of multimodal input (both text and image), the system will employ fusion techniques to combine the predictions from text and image models. This fusion may involve techniques such as concatenation, weighted averaging, or more advanced methods based on model performance.   + 1. Provide analysis of personality predictions in an easily understandable format * Prediction Visualization: The system will generate visual representations of the personality predictions to make them easily understandable to users. These visualizations may include graphs, charts, or other graphical elements. * Trait Descriptions: Each personality trait prediction will be accompanied by a brief description to clarify its meaning and relevance. Users can quickly grasp the significance of the predicted traits. * Trait Scores: The system will display numerical scores or ratings for each predicted personality trait. These scores indicate the strength or degree of expression of each trait based on the input data. * Overall Personality Summary: In addition to individual trait predictions, the system will provide users with an overall personality summary. This summary will consolidate the trait predictions into a coherent personality profile. * Comparison and Trends: Users may have the option to compare their current personality prediction with previous predictions if applicable. This allows users to track changes or trends in their personality traits over time. * Interactive Elements: The interface will incorporate interactive elements, such as hover-over tooltips, that provide additional information or context when users interact with specific parts of the visualization. * Customization: Users may have the ability to customize the format of the personality analysis. This could include choosing different types of visualizations or adjusting the level of detail displayed.      * + - 1. Offer insights or recommendations based on the analysis * Insightful Interpretations: The system will go beyond presenting raw data by providing insightful interpretations of the personality predictions. Users will receive explanations of what the predictions mean for them. * Recommendations: In some cases, the system may offer recommendations or insights based on the personality analysis. For example, it may suggest strategies for personal development or provide advice on improving certain traits. * User-Friendly Interface: The presentation of personality predictions and analysis will be designed with user-friendliness in mind. It will be intuitive, ensuring that users can easily access and understand the information. * Feedback Mechanism: The system may include a feedback mechanism, allowing users to provide feedback on the accuracy or relevance of the personality predictions and analysis. This feedback can be used for continuous improvement  DesignTechnical Environment Hardware Requirements:   * CPU * RAM 2 GB * HDD 200 GB * Network adapter * UPS   Operating System Requirements:   * Windows OS – latest version * Any Linux OS – latest version * MAC OS   Software Requirements:   * Jupyter Notebooks * Keras, TensorFlow * Spyder * Numpy and Pandas  Hierarchy of Modules  |  | | --- | | Authentication | | Verify Credentials | |  | | Password Encryption  Registration | |  |  |  | | --- | |  | | User Profile | | Manage Profile | |  | | Customization | |  |  |  | | --- | | Text-based Personality Prediction | | Data Collection and Processing | |  | | Feature Extraction | |  | | Model Training | |  | |  | | Image-based Personality Prediction | | Image Collection and Feature Extraction | |  | | Deep Learning Models | |  | | Model Training | |  | |  | | Multimodal Fusion | | Multimodal Fusion Technique | |  | | Comprehensive Analysis | |  | |  | | User Interface & Interaction | | User Interface Development | |  | | Data Input Handling | |  | | Predicted Information Display | |  | |  | | Model Evaluation & Improvement | | Model Evaluation Metrics | |  | | Model Fine Tuning |  Detailed Design **Activity Diagram**    **Interaction Overview Diagram**    **Sequence -Diagram**     * + 1. Authentication System: * **Description:** The Authentication System ensures secure user access to the application. It verifies user credentials, handles user sessions, and enforces security measures. * **Implementation:** Utilizes Django's built-in authentication system, including user models, views, forms, and middleware. It employs encryption for password storage and implements Two-Factor Authentication (2FA) for enhanced security.   + - 1. Password Validation Module: * **Description:** This module enforces strict password policies to enhance system security. It validates passwords during registration and password changes. * **Implementation:** Customizes Django's password validation settings to enforce rules like minimum length, complexity, and prevents common passwords. Employs client-side validation for immediate feedback.   + - 1. Registration System: * **Description:** The Registration System facilitates new user account creation. It collects user details, verifies email addresses, and creates user profiles. * **Implementation:** Develops registration views and forms, integrates email verification, and stores user data using Django's Object-Relational Mapping (ORM) to ensure data integrity.   + 1. Profile Management: * **Description:** Profile Management covers user profile-related operations, including creation, viewing, and updates. Users can access and edit their profiles. * **Implementation:** Implements Django views, forms, and templates for profile creation and editing. Uses Django's ORM to link user accounts to profiles.   + - 1. Profile Editing: * **Description:** This component focuses on enabling users to edit specific profile details, such as contact information and profile pictures. * **Implementation:** Develops forms for editing profile information. Ensures proper validation and secure storage of updated data.   + - 1. Profile Customization: * **Description:** Profile Customization allows users to personalize their profile appearance and settings. It includes options like selecting profile pictures and themes. * **Implementation:** Creates a user-friendly interface for customizing profile settings. Stores customization preferences in the database.   + 1. Prediction Dashboard: * **Description:** The Prediction Dashboard serves as the central hub for personality prediction. It provides a user-friendly interface for initiating predictions and viewing results. * **Implementation:** Designs a responsive and intuitive dashboard using Django templates and frontend technologies like HTML, CSS, and JavaScript. Integrates form handling and result visualization.   + - 1. Text Input Interface: * **Description:** This interface handles text-based input for personality prediction. Users can input or paste text data for analysis. * **Implementation:** Develops input forms and utilizes Django views for processing text data. Includes support for text preprocessing, such as stemming and stop words removal. * Data Collection: Incorporates a mechanism for users to provide text data, possibly from various sources, and gather user-labeled data to train and improve prediction models.   + - 1. Image Input Interface: * **Description:** The Image Input Interface manages image submissions for personality prediction. It allows users to upload images for analysis. * **Implementation**: Implements file upload functionality using Django's file handling capabilities. Utilizes libraries like OpenCV for image preprocessing. * Data Collection: Enables users to upload images relevant to the personality prediction task, while also providing the opportunity to collect labeled data for training image-based prediction models. Including data collection and labeled data gathering as part of these interfaces ensures that your system can collect valuable data from users to enhance the accuracy and effectiveness of personality prediction models. This user-generated data can be used for model training, validation, and improvement.   + - 1. Data Type Selection Module: * **Description:** This module offers users the flexibility to choose their preferred data type for personality prediction, whether it's text, images, or a combination. * **Implementation:** Provides an interactive component (e.g., dropdown menu) for users to select the desired data type. Adjusts data processing pipelines accordingly.   + - 1. Prediction Engine: * **Description:** The Prediction Engine is the core of personality prediction. It processes input data, applies machine learning models (e.g., Naive Bayes, BERT), and generates predictions. * **Implementation:** Develops machine learning pipelines using libraries like scikit-learn and Hugging Face Transformers. Fine-tunes models and integrates them into the application.   + 1. Analysis Module: * **Description:** The Analysis Module receives prediction results and generates human-readable reports. It may include visualizations and insights for users. * **Implementation:** Utilizes data visualization libraries (e.g., Matplotlib, Seaborn) and reporting tools (e.g., Jupyter Notebook) to create informative analysis outputs.   + - 1. Insights and Recommendations: * **Description:** This component derives actionable insights and recommendations based on the analysis of personality predictions. It helps users understand and apply the results. * **Implementation:** Implements algorithms to extract insights and suggestions. Presents these findings within the user interface, providing explanations and actionable advice.  Test Plan  |  |  |  |  | | --- | --- | --- | --- | | **Test Case Number** | **Test Case Description** | **Expected Outcome** | **Status** | | T-LGN-1 | User Credential verification | Completed | Completed | | T-LGN-1.1 | Validate encrypted Password | Completed | Completed | | T-LGN-1.2 | User Registration Functionality | Completed | Completed | | T-LGN-1.3 | Password recovery option | Completed | In Progress | | T-PROF-1 | Create user profile | Completed | Completed | | T-PROF-1.1 | Option for manage and update profile | Completed | Completed | | T-PROF-1.2 | User profile customization features | Completed | Completed | | T-PRED-1 | Prediction dashboard | Completed | Completed | | T-PRED-1.1 | Interface for input text data | Completed | Completed | | T-PRED-1.2 | Interface for input image data | Completed | Completed | | T-PRED-1.3 | Choice to choose data type for prediction | Completed | Completed | | T-PRED-1.4 | Implement personality prediction based on input data | Completed | Completed | | T-ANAL-1 | Provide analysis of personality prediction in an easily understandable format | Completed | In Progress | | T-ANAL-1.1 | Offer insights or recommendation based on the analysis | Completed | In Progress |   **Test Case 1**    **Test Case 2**    **Test Case 3**    **Test Case 4**    **Test Case 5**   Conclusion  * 1. Project Benefits   This project holds immense usefulness across diverse fields:   * Personal Growth and Development: Individuals can gain self-awareness, identify strengths, and work on areas for improvement, leading to personal growth and better life decisions. * Interpersonal Relationships: Better understanding of one's own and others traits enhances communication, conflict resolution, and relationship-building skills. * Organizational Productivity: Businesses can optimize team dynamics, job-role fit, and leadership development for enhanced productivity and employee satisfaction. * Education: Educational institutions can adapt teaching methods to students learning styles, leading to improved engagement and academic outcomes.   1. Future Scope for improvements * **Real-time Analysis**: Develop tools that provide real-time personality traits analysis, adapting to individuals changing behaviors and life circumstances. Like we will observe their social media profiles where they like, share, comments, post and from there we analyze their personality. * **Platform Independent:** Make the system capable of working on Androids and iOS.   1. Potential Market Demand * **Corporate Sector**: Businesses are increasingly recognizing the value of personality traits analysis for optimizing team dynamics, leadership development, and employee satisfaction. * **Education Industry:** Schools, colleges, and educational platforms can use personality traits analysis to personalize learning experiences and improve student outcomes. * **Mental Health Services:** Therapists and counsellors can gain useful insights from personality traits analysis for more effective treatment planning and emotional support. * **Recruitment and HR:** Companies can use personality traits analysis in recruitment processes to match candidates with job roles, leading to improved employee retention. * **Marketing and Consumer Insights:** Businesses can tap into personality traits analysis for targeted marketing strategies and product development.  References / Bibliography  * <https://cynoteck.com/blog-post/top-software-development-models-to-choose-from/> * <https://www.researchgate.net/publication/359303511_A_Machine_Learning_Approach_to_Identify_Personality_Traits_from_Social_Media> * <https://grantkim94.medium.com/personality-prediction-system-based-on-graphology-using-machine-learning-929fdbae0dee> * <https://www.enjoyalgorithms.com/blog/personality-prediction-using-ml>   [1] We are Social and Hootsuite, “Digital in 2017 Global Overview”, In slides, 2018-7- 7, <https://www.slideshare.net/wearesocialsg/digital-in-2017-global-overview>.  [2] Richter, F. (2017). Snapchat crowned number 1 by American teens. Statista. Retrieved from <https://www.statista.com/chart/4823/teenagersfavorite-> social-networks  [3] F. Bruce, M. Schedl, M. Tkalcic,” Predicting personality traits with Instagram pictures.” In Proceedings of the 3rd Workshop on Emotions and Personality in Personalized Systems 2015, pp. 7-10. ACM, 2015.  [4] Megvii, I. “Face++ research toolkit.” (2013). APPENDIX A – Prototypes ## HERE DATA SET ARE TRAINED  import csv  import array  import pandas  import pickle  import os  import sys  import numpy as np  from sklearn.feature\_extraction.text import CountVectorizer  from sklearn.feature\_extraction.text import TfidfVectorizer  from sklearn.naive\_bayes import GaussianNB  from sklearn import svm  csvFile=open('newfrequency300.csv', 'rt')  csvReader=csv.reader(csvFile)  mydict={row[1]: int(row[0]) for row in csvReader}  # print(mydict)  y=[]  with open ('PJFinaltest.csv', 'rt') as f:      reader=csv.reader(f)      corpus=[rows[0] for rows in reader]  # print(corpus)  with open ('PJFinaltest.csv', 'rt') as f:      csvReader1=csv.reader(f)      for rows in csvReader1:          y.append([int(rows[1])])  # print(y);  vectorizer=TfidfVectorizer(vocabulary=mydict,min\_df=1)  x=vectorizer.fit\_transform(corpus).toarray()  # print(x)  result=np.append(x,y,axis=1)  # print(result)  X=pandas.DataFrame(result)  # print(X)  model=GaussianNB()  train = X.sample(frac=0.8, random\_state=1)  test=X.drop(train.index)  y\_train=train[301]  y\_test=test[301]  print(train)  print(train.shape)  print(test.shape)  xtrain=train.drop(301,axis=1)  xtest=test.drop(301,axis=1)  model.fit(xtrain,y\_train)  pickle.dump(model, open('BNPJFinal.sav', 'wb'))  del result  y=[]  with open ('IEFinaltest.csv', 'rt') as f:      reader=csv.reader(f)      corpus=[rows[0] for rows in reader]  with open ('IEFinaltest.csv', 'rt') as f:      csvReader1=csv.reader(f)      for rows in csvReader1:          y.append([int(rows[1])])  vectorizer=TfidfVectorizer(vocabulary=mydict,min\_df=1)  x=vectorizer.fit\_transform(corpus).toarray()  result=np.append(x,y,axis=1)  X=pandas.DataFrame(result)  model=GaussianNB()  train = X.sample(frac=0.8, random\_state=1)  test=X.drop(train.index)  y\_train=train[301]  y\_test=test[301]  print(train.shape)  print(test.shape)  xtrain=train.drop(301,axis=1)  xtest=test.drop(301,axis=1)  model.fit(xtrain,y\_train)  pickle.dump(model, open('BNIEFinal.sav', 'wb'))  del result  y=[]  with open ('TFFinaltest.csv', 'rt') as f:      reader=csv.reader(f)      corpus=[rows[0] for rows in reader]  with open ('TFFinaltest.csv', 'rt') as f:      csvReader1=csv.reader(f)      for rows in csvReader1:          y.append([int(rows[1])])  vectorizer=TfidfVectorizer(vocabulary=mydict,min\_df=1)  x=vectorizer.fit\_transform(corpus).toarray()  result=np.append(x,y,axis=1)  X=pandas.DataFrame(result)  model=GaussianNB()  train = X.sample(frac=0.8, random\_state=1)  test=X.drop(train.index)  y\_train=train[301]  y\_test=test[301]  print(train.shape)  print(test.shape)  xtrain=train.drop(301,axis=1)  xtest=test.drop(301,axis=1)  model.fit(xtrain,y\_train)  pickle.dump(model, open('BNTFFinal.sav', 'wb'))  del result  y=[]  with open ('SNFinaltest.csv', 'rt') as f:      reader=csv.reader(f)      corpus=[rows[0] for rows in reader]  with open ('SNFinaltest.csv', 'rt') as f:      csvReader1=csv.reader(f)      for rows in csvReader1:          y.append([int(rows[1])])  vectorizer=TfidfVectorizer(vocabulary=mydict,min\_df=1,lowercase=False)  x=vectorizer.fit\_transform(corpus).toarray()  result=np.append(x,y,axis=1)  X=pandas.DataFrame(result)  model=GaussianNB()  train = X.sample(frac=0.8, random\_state=1)  test=X.drop(train.index)  y\_train=train[301]  y\_test=test[301]  print(train.shape)  print(test.shape)  xtrain=train.drop(301,axis=1)  xtest=test.drop(301,axis=1)  model.fit(xtrain,y\_train)  pickle.dump(model, open('BNSNFinal.sav', 'wb'))  # ## HERE REAL DATA ARE TESTED  import tweepy  from nltk.corpus import stopwords  from nltk.tokenize import word\_tokenize  from nltk.stem import \*  from nltk.stem.snowball import SnowballStemmer  from nltk.stem import SnowballStemmer  import sys  import os  import nltk  import re  import numpy as np  import string  from unidecode import unidecode  import csv  from itertools import islice  import pandas as pd  import pickle  from sklearn.feature\_extraction.text import CountVectorizer  from sklearn.feature\_extraction.text import TfidfVectorizer  from collections import Counter  import nltk  nltk.download('stopwords')  import nltk  nltk.download('punkt')    import re  emoticons\_str = r"""      (?:          [:=;] # Eyes          [oO\-]? # Nose (optional)          [D\)\]\(\]/\\OpP] # Mouth      )"""  emoji\_pattern = re.compile("["          u"\U0001F600-\U0001F64F"  # emoticons          u"\U0001F300-\U0001F5FF"  # symbols & pictographs          u"\U0001F680-\U0001F6FF"  # transport & map symbols          u"\U0001F1E0-\U0001F1FF"  # flags (iOS)                             "]+", flags=re.UNICODE)  regex\_str = [      emoticons\_str,      r'<[^>]+>',  # HTML tags      r'(?:@[\w\_]+)',  # @-mentions      r"(?:\#+[\w\_]+[\w\'\_\-]\*[\w\_]+)",  # hash-tags      r'http[s]?://(?:[a-z]|[0-9]|[$-\_@.&amp;+]|[!\*\(\),]|(?:%[0-9a-f][0-9a-f]))+',  # URLs      r'(?:(?:\d+,?)+(?:\.?\d+)?)',  # numbers      r"(?:[a-z][a-z'\-\_]+[a-z])",  # words with - and '      r'(?:[\w\_]+)',  # other words      r'(?:\S)'  # anything else  ]  tokens\_re = re.compile(r'(' + '|'.join(regex\_str) + ')', re.VERBOSE | re.IGNORECASE)  emoticon\_re = re.compile(r'^' + emoticons\_str + '$', re.VERBOSE | re.IGNORECASE)  def tokenize(s):      return tokens\_re.findall(s)  def preprocess(s, lowercase=False):      tokens = tokenize(s)      if lowercase:          tokens = [token if emoticon\_re.search(token) else token.lower() for token in tokens]      return tokens  def preproc(s):      #s=emoji\_pattern.sub(r'', s) # no emoji      s= unidecode(s)      POSTagger=preprocess(s)      #print(POSTagger)      tweet=' '.join(POSTagger)      stop\_words = set(stopwords.words('english'))      word\_tokens = word\_tokenize(tweet)      #filtered\_sentence = [w for w in word\_tokens if not w in stop\_words]      filtered\_sentence = []      for w in POSTagger:          if w not in stop\_words:              filtered\_sentence.append(w)      #print(word\_tokens)      #print(filtered\_sentence)      stemmed\_sentence=[]      stemmer2 = SnowballStemmer("english", ignore\_stopwords=True)      for w in filtered\_sentence:          stemmed\_sentence.append(stemmer2.stem(w))      #print(stemmed\_sentence)      temp = ' '.join(c for c in stemmed\_sentence if c not in string.punctuation)      preProcessed=temp.split(" ")      final=[]      for i in preProcessed:          if i not in final:              if i.isdigit():                  pass              else:                  if 'http' not in i:                      final.append(i)      temp1=' '.join(c for c in final)      #print(preProcessed)      return temp1  print("hello")  with open('user.csv','rt') as f:      csvReader=csv.reader(f)      tweetList=[rows[0] for rows in csvReader]  # print(tweetList,".....................")  with open('newfrequency300.csv','rt') as f:      csvReader=csv.reader(f)      mydict={rows[1]: int(rows[0]) for rows in csvReader}  vectorizer=TfidfVectorizer(vocabulary=mydict,min\_df=1,lowercase=False)  # print(vectorizer)  x=vectorizer.fit\_transform(tweetList).toarray()  # print(x)  df=pd.DataFrame(x)  # print(df)  model\_IE = pickle.load(open("BNIEFinal.sav", 'rb'))  # print(model\_IE)  model\_SN = pickle.load(open("BNSNFinal.sav", 'rb'))  # print(model\_SN)  model\_TF = pickle.load(open('BNTFFinal.sav', 'rb'))  model\_PJ = pickle.load(open('BNPJFinal.sav', 'rb'))  answer=[]  IE=model\_IE.predict(df)  # print(IE)  SN=model\_SN.predict(df)  TF=model\_TF.predict(df)  PJ=model\_PJ.predict(df)  b = Counter(IE)  value=b.most\_common(1)  print(value)  if value[0][0] == 1.0:      answer.append("I")  else:      answer.append("E")  b = Counter(SN)  value=b.most\_common(1)  print(value)  if value[0][0] == 1.0:      answer.append("S")  else:      answer.append("N")  b = Counter(TF)  value=b.most\_common(1)  print(value)  if value[0][0] == 1:      answer.append("T")  else:      answer.append("F")  b = Counter(PJ)  value=b.most\_common(1)  print(value)  if value[0][0] == 1:      answer.append("P")  else:      answer.append("J")  mbti="".join(answer)  # print(mbti)  IMAGE-BASED PERSONALITY  import os  import numpy as np  import tensorflow as tf  from tensorflow import keras  from tensorflow.keras.layers import Conv2D, BatchNormalization, Dropout, Flatten, Input, Dense, MaxPooling2D, Reshape  from tensorflow.keras.optimizers import Adam  from tensorflow.keras.models import Model, model\_from\_json, load\_model  from tensorflow.keras.callbacks import ReduceLROnPlateau  from tensorflow.keras.losses import MeanSquaredError  from tensorflow.keras import backend as K  from PIL import Image, ImageOps  import pickle  import time  import tqdm  import math  X\_train = []  Y\_train = []  base\_path = '/content/gdrive/MyDrive/data\_for\_training&test'  for i in tqdm.tqdm(range(1,4)):    file\_name = 'train\_clselfie\_v4\_' + str(i) + ".pickle"    pic = pickle.load(open(os.path.join(base\_path, file\_name), "rb"))    for i in pic['X']: for i in pic['X']:      x = np.array(i)      X\_train.append(x)    for i in pic['Y']:      y = np.array(i)      Y\_train.append(y)  file\_name = 'test\_clselfie\_v4.pickle'  X\_val = []  Y\_val = []  pic = pickle.load(open(os.path.join(base\_path, file\_name), "rb"))  for i in pic['X']:    x = np.array(i)    X\_val.append(x)  for i in pic['Y']:    y = np.array(i)    Y\_val.append(y)  X\_train = np.array(X\_train)  Y\_train = np.array(Y\_train)  X\_val = np.array(X\_val)  Y\_val = np.array(Y\_val)  print(X\_train.shape)  FEATURE\_EXTRACTION  input\_cla = Input(shape=(208,208,1,))  fm\_size = 16;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(input\_cla)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  fm\_size = fm\_size\*2;  x = Conv2D(fm\_size, (3,3), activation='relu', padding='same')(x)  x = MaxPooling2D((2, 2), padding='same')(x)  em\_size = math.ceil(208/(2\*\*8));  x = Reshape((em\_size\*em\_size\*fm\_size,),input\_shape=(em\_size,em\_size,fm\_size))(x)  x=Dense(50,activation='relu', input\_shape=(em\_size\*em\_size\*fm\_size,))(x)  x=Dense(50,activation='relu')(x)  x=Dense(10,activation='relu')(x)  output=Dense(5)(x)  classifier = Model(input\_cla, output)  adam = Adam(lr=1e-4)  loss = MeanSquaredError()  classifier.compile(optimizer=adam, loss=loss, metrics = [tf.keras.metrics.MeanSquaredError()])  classifier.summary()  callbacks = [ReduceLROnPlateau(factor=0.3, patience=5, verbose = 1)]  BATCH\_SIZE = 64  TRAIN\_STEP\_SIZE = len(X\_train) // BATCH\_SIZE  history = classifier.fit(X\_train, Y\_train, batch\_size = BATCH\_SIZE, epochs = 50, steps\_per\_epoch = TRAIN\_STEP\_SIZE,                      shuffle = True, validation\_data = (X\_val, Y\_val))  PREDICTION  import os  import time  import tensorflow as tf  from tensorflow.keras.models import model\_from\_json  import cv2  import numpy as np  import urllib  import warnings  warnings.filterwarnings("ignore")  # Define the image directory  image\_directory = './images/introvert'  # Change this to the directory containing your images  # Load the model  with open('ipv2.json', 'r') as json\_file:      model = model\_from\_json(json\_file.read())  model.load\_weights('ipv2.h5')  # Define personality traits  personality\_traits = ['Extraversion', 'Agreeableness', 'Conscientiousness', 'Neuroticism', 'Openness']  # Create an output directory if it doesn't exist  output\_dir = 'output'  os.makedirs(output\_dir, exist\_ok=True)  # Process each image in the specified directory  for filename in os.listdir(image\_directory):      if filename.endswith(('.jpg', '.jpeg', '.png', '.jfif')):  # Add more image extensions if needed          image\_path = os.path.join(image\_directory, filename)          print(f'Processing image: {filename}')          # Load and preprocess the image          img = cv2.imread(image\_path, cv2.IMREAD\_GRAYSCALE)          img = cv2.resize(img, (208, 208), cv2.INTER\_CUBIC)          img = np.array(img)          img = np.expand\_dims(img, axis=0)          # Predict personality traits          y = model.predict(img)  #         Create a result file for each image          result\_filename = os.path.splitext(filename)[0] + '\_result.txt'          result\_filepath = os.path.join(output\_dir, result\_filename)          # Write results to the result file          with open(result\_filepath, "w") as f:              for i in range(len(personality\_traits)):                  result = f"{personality\_traits[i]} ----------> {y[0][i]\*100}%\n"                  print(result)                  f.write(result) |
|  |
|  |
|  |
|  |
|  |
|  |