VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

DECLARATION

We hereby declare that the project report entitled "***Predicting MBTI personality types and calculating HEXACO scores using Hyperparameter tuning Random Forest Algorithm***" submitted by us, for the award of the degree of Computer Science and Engineering, VIT is a record of bonafide work carried out by us under the supervision of Dr. Manjula D.

We further declare that the work reported in this project report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai
Date: 8/11/23

Signature of the Candidates:

Vansh Shan	Rochak Shrivastav	Pranay Pratik
20BCE1533	20BCE1814	20BCE1751

A project report on

Predicting MBTI personality types and calculating HEXACO scores using Hyperparameter tuning Random Forest Algorithm

Submitted in partial fulfillment for the award of the degree of

Computer Science and Engineering

for

CSE4022 Natural Language Processing

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

By

VANSH SHAN 20BCE1533

ROCHAK SHRIVASTAV 20BCE1814

PRANAY PRATIK 20BCE1751

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

November, 2023

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

DECLARATION

We hereby declare that the project report entitled "***Predicting MBTI personality types and calculating HEXACO scores using Hyperparameter tuning Random Forest Algorithm***" submitted by us, for the award of the degree of Computer Science and Engineering, VIT is a record of bonafide work carried out by us under the supervision of Dr. Manjula D.

We further declare that the work reported in this project report has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Place: Chennai

Date:

Signature of the Candidates:

Vansh Shan	Rochak Shrivastav	Pranay Pratik
20BCE1533	20BCE1814	20BCE1751

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the report **entitled "Predicting MBTI personality types and calculating HEXACO scores using Hyperparameter tuning Random Forest Algorithm"** is prepared and submitted by **Vansh Shan, Rochak Shrivastav and Pranay Pratik** to VIT Chennai, in partial fulfillment of the requirement for 'J' component of CSE4022 – Natural Language Processing subject is a bonafide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission.

Signature of the Guide:

Name: Dr. Manjula D

Date:

ABSTRACT

It is commonly recognized that a person's personality serves as a predictor of job success, job happiness, and tenure intention. The capability to assess a candidate's personality throughout the recruiting process benefits the candidate, hiring managers, and recruiters in improving hiring decisions. Our research demonstrates that personality characteristics may be accurately inferred from the textual content of replies to common interview questions about previous behavior and situational judgment. The base paper finds the HEXACO traits using Random Forest Algorithm. It has an accuracy of 87.83%. Our project aims to build a website which uses NLP and machine learning algorithms to predict the HEXACO personality traits and MBTI personality type based on the input given by the user. We will be using the Doc2Vec method for training the dataset and calculating the HEXACO scores. Along with this we will be using Hyperparameter tuning to improve the performance of the Random Forest classification model.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, **Dr. Manjula D**, School of Computer Science and Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. GANESAN R**, Dean of School of Electronics Engineering, VIT Chennai, for extending the facilities of the school towards our project and for her unstinting support.

We express our thanks to our Head of the Department **Dr. NITHYANANDAM P** for his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

Place: Chennai

Date:

Name of the students

Table of Contents

LIST OF FIGURES

LIST OF ACRONYMS

MBTI - Myers-Briggs Type Indicator

HEXACO - honesty- humility, emotionality, extraversion, agreeableness, conscientiousness, and openness to experience

KNN - K nearest neighbors

SVM - Support Vector Machine

Chapter 1

INTRODUCTION

Individual differences in behaviour or emotions brought on by environmental or biological variables are referred to as one's personality. It illustrates the individual's divergent ways of thinking, acting, and feeling. Personality characteristics are continuous in nature rather than displaying a separate personality since they represent the highs and lows of certain attributes in a person on a continuous basis. The Latin word "persona," which means "mask," is where the word "personality" first appeared. Three factors are used to define personality traits: consistency across contexts, constancy over time, and individual differences, which indicates that different people behave in different ways. Personality psychology is the subject of study that examines human personality and how it varies from person to person and from group to group.

The use of artificial intelligence and data science is revolutionising how technology is changing the world. Although machine learning has many applications that we may see in our daily lives, one of its most significant purposes is to classify individuals according to the traits that make up their personalities. Each person in this world is unique and bears a unique personality. The availability of vast amounts of high-dimensional data has cleared the path for boosting the effectiveness of marketing campaigns by targeting specific consumers. Such personality-based communications are extremely effective in improving the appeal and attractiveness of products and services. This can lead to increased usage, increased customer satisfaction, and broader acceptability among users.

Researching and observing human behaviour is made far too convenient by the availability of high-dimensional and fine-grained data regarding human behaviour. The methods that psychologists use to conduct study and evaluate people's personalities have been revolutionised by mobile sensing studies, which collect data from the activities that we participate in on a daily basis. Researchers will find that employing machine learning models to discover very complicated correlations, as well as test the generalizability and robustness of those associations using the resampling method, will be a boon to their work. It possesses the potential to revolutionise both research and evaluation within the field of personality psychology. It is possible for algorithms to process enormous datasets that contain thousands of attributes without experiencing any collinearity problems. In addition, machine learning algorithms are extremely effective in recognising patterns in datasets that humans are unable to even recognise themselves. The application of these ML models can result in personality evaluations that are superior, more objective, and fully automated.

The usage of social networking sites has expanded with the development of technology. People utilise it as a forum to communicate and express their thoughts, hopes, and other things. Along with this, users often provide details about themselves, like their employment, preferences, and so on. It is possible to retrieve this information. Businesses may be able to interact with their clients, comprehend their demands, and then enhance the quality of their products or services by using the extracted data. It is used to discover connection patterns, how things are related, and how things are similar.

To forecast the person's personality scores, we use the HEXACO model. The HEXACO factors were first discovered in factor analyses of personality adjective evaluations, as previously reported. Individuals assessed their own levels of a number of personality characteristics in these research. The HEXACO Personality survey was developed to enable a more precise examination of the HEXACO traits. In the self-report version of the survey, the respondent or another individual, and the survey consists of a series of observations about the target person's personality. The responder expresses how much he or she agrees or disagrees with each assertion.

The six broad HEXACO personality variables, each of which has four "facets" or more specific personality traits, are assessed using the HEXACO Personality Inventory-Revised (HEXACO-PI-R). (An extra, 25th narrow aspect, termed Altruism, which combines the Honesty-Humility, Emotionality, and Agreeableness components, is also included.) Each component has four aspects, which are as follows:

Honesty-Humility (H): Sincerity, Fairness, Avoidance of Greed, and Modesty Emotionality (E): Sentimentality, Dependence, Anxiety, and Fearfulness Social Self-Esteem, Social Boldness, Sociability, and Liveliness are all examples of extraversion (X). Forgiveness, gentleness, flexibility, and patience comprise the trait of agreeability (A). Organisation, diligence, perfectionism, and caution are all examples of conscientiousness (C). Openness to Experience (O): Appreciation of the aesthetics, curiosity, creativity, and unconventionality

The HEXACO-PI-R assigns a level or score to each of the variables and aspects when a person's personality is evaluated. These results are distributed over a continuum from extremely low to very high.

In respect to the HEXACO personality traits, the possible adaptive trade-offs between various levels of each dimension during the stages of human development have been investigated. Higher levels of the components Agreeableness, Honesty-Humility, and are thought to indicate two separate elements of a predisposition toward reciprocally altruistic conduct, whereas higher levels of Emotionality are thought to show a tendency toward kin altruistic behaviour (and toward personal and kin survival more generally). Greater involvement

or effort in three distinct domains—social, task-related, and idea-related—is thought to be represented by higher degrees of extraversion, conscientiousness, and openness to experience characteristics.

The Myers-Briggs Type Indicator (MBTI), a self-report inventory that is based on Jungian theory, is one of the most well-known self-report inventories. The MBTI is a four-factor model that enables individuals to identify their unique type using four letters (such as ENTJ or ISFP). Eight scores are obtained from the scale (one for each type), which may be compared to four typological opposites.

It consists of 94 forced-choice questions with scores on each of the eight criteria and the well-known four dimensions. Sensation-Intuition, Thinking-Feeling, Introversion-Extraversion, Judging-Perceiving, and Sensation-Intuition. Based on the greatest score attained for each bipolar scale, respondents are divided into one of 16 personality types.

The linear scores on each dimension provided by the MBTI-Form G (Briggs-Myers & Briggs, 1985) are often addressed in terms of types based on cutoff values. As a result, the Extraversion-Introversion dimension has a normal distribution, with high scores indicating extraversion and low values indicating introversion.

During the training phase, the random forests ensemble learning technique, that is employed for classification, regression, as well as other tasks, constructs a large number of decision trees. The outcome of a random forest for classification problems is the class selected by the majority of trees. For regression tasks, the mean or average prediction of each individual tree is presented. Random choice forests counteract the propensity of decision trees to overfit their training set. Gradient-enhanced trees are more precise than random forests, while they often perform better than choice trees. However, data anomalies may reduce their usefulness.

A hyperparameter is a setting that is built into an algorithm but cannot be learned from the typical data we feed it. Every algorithm comes with its own one-of-a-kind set of hyperparameters that have been determined in advance. It is an ensemble method that combines several different decision trees in order to work through difficult problems and arrive at the solution. When trying to improve the accuracy of a model, it is common practise to alter the hyperparameters.

The process of hyperparameter tuning involves determining the ideal hyperparameter values for a learning algorithm and applying this improved algorithm to any data collection. This hyperparameter combinations optimises the performance of the model by greatly reducing a preset loss function, resulting in more accurate outcomes with minimal errors. Notably, the training algorithm tries to find the optimal solution within the

limits and enhances the loss based on the input data. However, this configuration is precisely described by hyperparamete

Chapter 2

BACKGROUND

INTRODUCTION

We referred to sufficient journal papers to gain knowledge on developing our project.

SURVEY

A research article suggests ways to predict the personality of a person using his/her facial expressions mainly during online interviews. It describes the use of facial width-to-height ratio (fWHR) extracted from the image of a person using CNN algorithm in order to find the personality of that person among the big five personality traits. Additionally, this work uses a questionnaire based analysis using K-Means clustering Algorithm to support this personality prediction. An accuracy of about 92% was obtained, but the accuracy drastically decreased after the 14th epoch due to some training parameters and the dataset for CNN algorithm was too small. [1]

A paper on personality classification applied cass balancing techniques and experimented different machine learning classifiers, namely, KNN, Decision Tree, Random Forest, MLP, Logistic Regression (LR), SVM, XGBoost, MNB and Stochastic Gradient Descent (SGD) to identify the personality traits. Then it uses accuracy, precision, recall and Ƒ- score, to analyse and examine the overall efficiency of the predictive model. An accuracy of 99% for I/E and S/N traits were achieved using XGBoost classifier. However, KNN classifier resulted in overall lower performance and less weightage was given to feature extraction in classification of text. [2]

A research article describes a multi model deep learning architecture with a pre-trained language model BERT, RoBERTa, and XLNet; along with additional NLP Features (sentiment analysis, TF-IGM, NRC emotion lexicon database) as features extraction method for personality prediction system. It shows the use of combination of multiple sources of social media data to increase the number of datasets for better classification and produces the highest accuracy of 86.17% and f1 measure score 0.912 on Facebook dataset and 88.49% accuracy and 0.882 f1 measure score on the Twitter dataset. However, accuracy measurement used was only good for class distribution on balanced data. [3]

A research articles show a technique that is specifically designed to assess personality traits as objectively and transparently as possible, and then applies it to job interviews. The authors built the technique based on existing closed-vocabulary techniques with the aim to improve their validity by not relying on subjective ratings and using more complete word lists. By using this the quality of the text-to-personality ratings improves when interview questions activate the personality trait under investigation. But the interviews were from either a low-stakes or simulated selection situation, but none were from a true personnel selection context. [4]

A survey gives an overview of different strategies used to predict personality and behaviour using the content available on social sites. This paper gives the summary of the work done for predicting the personality on text from Social media sites and to summarise the future trends. However this paper doesn't have a new unique approach to predict personality and it merely states in brief the approach used by several authors in this field. [5]

A paper shows an online application for the registration of candidate's details and analysis of candidate's personalities through an online Multiple-Choice Question (MCQ) test containing personality quizzes.the options have a score range of 1-8. Then the system analyses professional eligibility by comparing the uploaded Curriculum Vitae trained datasets. This system employs a machine learning algorithm namely "Logistic Regression" and Random Forest Classifier which helps to choose fair decisions to recruit a suitable candidate. The proposed system is fully automated with more transparency and legibility, thus helping in reducing the burden of HR, time consumption, and more productivity. However, Mcq response based prediction has smaller deviation and hence lesser accuracy than textual data. [6]

A research paper shows a comparative study of various strategies and algorithms used in The Big Five test. They have inspected ways to use the new technology for future development in this area and also understand human personality and contribute to the development of personality theory. They have shown different algorithms used and the performances for predicting personality. Though they have mentioned that they will develop a new model in future, this paper doesn't have a new unique detailed approach to predict personality. [7]

A paper proposes an intelligent personality prediction system to predict the personality of candidates based on Twitter data. The proposed method is a machine learning technique using Random Forest classifier based on Myers–Briggs Type Indicator. It mines user characteristics and learns patterns from large amounts of personal behavioural data. This system can automatically evaluate candidates' personality traits by processing various attributes and eliminating the time-consuming process required in the conventional

approach. Experimental evaluation demonstrates that the Random Forest classifier performs better than the different traditional machine learning algorithms in terms of accuracy. It cancels out biases by taking the average of all predictions. However, This research only studied people with particular social media, namely Twitter, and this algorithm requires multiple decision trees resulting in a slow prediction generation process. [8]

A paper presents a model that predicts which cloud services a client will like to purchase based on a variety of variables, including their past purchasing behaviour, a sequence of adverts watched, customer locality, etc. Using a Training Dataset, they developed a Random Forest Model for determining the most desirable service a customer can purchase. This proposed model predicts client purchasing behaviour with an approximate accuracy of 87.02%. The experimental results demonstrate that Random Forest Model method is practicable and enables real-time customer behaviour analysis. However, some more classification models could be used to analyse the accuracy score of different models with respect to this dataset. [9]

A research paper proposes a Applicant Personality Prediction System Using Machine Learning which predicts the personality of the candidate based on the candidate's resume details and also by using some personality tests. This method quickly identifies and selects the best candidate for the desired job profile by evaluating the resume based on certain criteria such as experience, skills, etc. It uses the Spacy module to analyse, summarise, and compare resumes and job descriptions to generate a similarity score. Factors such as conscientiousness, openness, and agreeableness, among others, are considered to determine the user's personality traits score. These two scores are used to shortlist the candidates. The accuracy of this model is 79.47%, which is significantly higher than that of the KNN and Random Forest Algorithm models. However, they have used traditional questions to evaluate personality score instead of using user's text and behaviour in social media which would have given larger realistic dataset and hence better accuracy [10]

A paper applies Cronbach Alpha test to a questionnaire made using the HEXACO model before implementing the machine learning algorithm to verify the reliability of the factor and variables considered during the study of the personality trials from the dataset. This research is made to improve personality prediction, the dataset is constructed and utilised based on HEXACO Model. It showed that contrary to popular opinion, random reactions can inflate the alpha of Cronbach when their mean differs from that of the true reactions. However HEXACO is an outdated technique as the data can be manipulated. [11]

A research paper proposed an advanced personality classification and professional profile prediction system that extracts personality patterns through data mining. It used the Random Forest Classifier algorithm to predict user personality based on personality data that was saved via classification of prior user data. This

system gives its users a wider and more precise understanding of how they might improve their professional lives. However the proposed application is not interactive. [12]

A research paper proposes an Automated Personality Classification system that uses the Naive Bayes Algorithm and Support Vector Machine to analyse a user's personality based on specific parameters. This method will be useful for businesses and other agencies that recruit individuals based on their personality as opposed to their technical competence. With an accuracy of approximately 60%, this Personality Prediction System examines the entire personality of candidates and might be valuable for firms that hire individuals for management and sales positions based on their personality. A limitation of Naïve Bayes method is it supposes that all the linguistic features are conditionally independent. [13]

A research paper proposes using an Attention Recurrent Neural Network (AttRNN) model that takes into consideration the sequence of digital footprints to solve the challenge of predicting personality traits. The AttRNN model predicts openness, conscientiousness, and neuroticism with much greater accuracy than linear regression and the BiGRU model. The experimental findings demonstrated the efficiency of the AttRNN model for predicting personality characteristics. However, training datasets using RNN is a very difficult task and it cannot process large datasets. [14]

A research paper proposes a flight delay prediction model based on the random forest model, which by analysing the departure flight data of Guangzhou Baiyun International Airport in June 2020, and by selecting the data of ten landing airports, analyses the distribution of delayed, punctual, and early arrived flights. The accuracy of the prediction model was close to 90 percent. However, Random forests are found to be biassed while dealing with categorical variables and it has a time consuming training rate. As flight information is fast-paced changing, slow training of datasets can cause low delivery rate. [15]

A research paper shows Stacked Random Forests (SRFs) successively apply numerous Random Forests (RFs), with each instance using the estimate of the previous as extra input to refine the forecast. The author demonstrates that through stacking, classification accuracy steadily improves and then soon reaches a halt after only a few levels of stacking. Through stacking the performance gain quickly peaks and increases calibration quality marginally. One point that was neglected in this paper is that probabilistic predictions can not only be checked regarding their calibration quality. [16]

A research paper proposes the weighted random forest and ant colony algorithm as a means of resolving this issue and successfully analysing and studying the financial market to increase the predicted income. The proposed weighted random forest model has a lower prediction error than both the standard random forest approach and the regression procedure. On the data sets of four stocks, the algorithm has produced the most

accurate and robust outcomes, achieving the best results in terms of accuracy and robustness. However due to the limited amount of data in this data collection, theoretical statistics are unavailable. [17]

A research paper deals with the Random Forest (RF) algorithm to detect four types of attack like DOS, probe, U2R and R2L.They have adopted 10 cross validation applications for classification. Feature selection is applied on the data set to reduce dimensionality and to remove redundant and irrelevant features. They applied symmetrical uncertainty of attributes which overcomes the problems of information gain. The proposed approach is evaluated using the NSL KDD dataset. It yielded high DR and low FAR to classify the attacks. For DOS attack an accuracy of 99.67% was achieved which is 7% more than J48 algorithm. However they have used simple computation as a feature selection measure, which decreased their accuracy. [18]

In this paper the datasets consist of observation vectors *Vt.*After that, the datasets are injected with two faults at different rates. Following that, six classifiers are applied on outdoor data collected from multi-hop WSNs. The classifiers are then evaluated on the basis of four different performance metrics.The extensive simulation results show that RF outperformed in terms of DA and TPR.Random forest classifier have beaten all other classifiers like SVM, CNN, MLP, SGD, RF, and PNN in terms of DA, TPR, MCC, and F1-score.The dataset used is large and mostly raw.They have used few number of sensors to obtain data for their datasets. The robustness of the RF algorithm can also be checked by increasing the number of sensors. [19]

A research paper tests and examines classification algorithms and personality predictability using the survey results. The research demonstrates the different methods and templates that were used.The experimental research demonstrates that, although each algorithm exhibits a different accuracy rate for the various personality traits in the data collection, Logical Regression and Naïve Bayes algorithm gives the best accuracy contrasting different parameters with less error rate. The paper has many graphs to show the comparative study of accuracy of different algorithms. An unique way of evaluation was done through a confusion matrix. However, the research does't show comparative study of results of this algorithm on the basis of time. [20]

A research paper took a dataset from 2 sources, "myPersonality" and manually using Facebook API graph uses several features to see the comparison of the results and capabilities between them. They have used linguistic features such as LIWC and SPLICE with a closed-vocabulary approach, which performs an automatic exploration of the dataset to find relationships between words with personality. However it is complex to implement deep learning algorithms to obtain results for prediction. [21]

A paper on personality prediction provides preliminary work on personality prediction by using various algorithms. Then it discusses various algorithms to predict personality from social media users. The proposed

hybrid algorithm can predict the personality of the person with comparatively higher accuracy. However if the job applicants are non-social media networking users, then the proposed framework cannot be used. [22]

This paper proposed an ensemble random forest algorithm based on Apache Spark which can be used in the large scaled imbalanced classification of insurance business data, the experiment result showed that the ensemble random forest algorithm is more suitable in the insurance product recommendation than SVM and Logistic Regression.They used the heuristic under-sampling algorithm which will find a better sample subset by searching for security samples near the classification boundary.The computational complexity of serial ensemble random forest algorithm is unpredictable in the situation of large-scale business data. [23]

The purpose of this paper was to determine if a data mining approach could offer new insights that can explain sugarcane productivity in the Wet Tropics, Australia.The approach outlined in this paper can easily be extended to other sugarcane-growing regions.The model was not able to take account of the amount of damage to sugarcane due to wet weather harvesting.The model doesn't consider external parameter which can have an negative effect on actual result.[24]

The objective of this study was to investigate whether the RF approach could successfully simulate the complicated relationship between the predictors and predictands. The daily mean temperature observations and the NCEP reanalysis daily data were selected in order to compare the results of the RF model with those of the MLR, ANN, and SVM models.The built-in variable importance evaluation process and the OOB samples in the RF made predictor selection convenient. The OOB samples gave unbiased estimation of the model efficiency. There was room for improvement for the prediction accuracy in mountainous areas.[25]

Chapter 3

PROPOSED SYSTEM

3.1 SYSTEM ARCHITECTURE

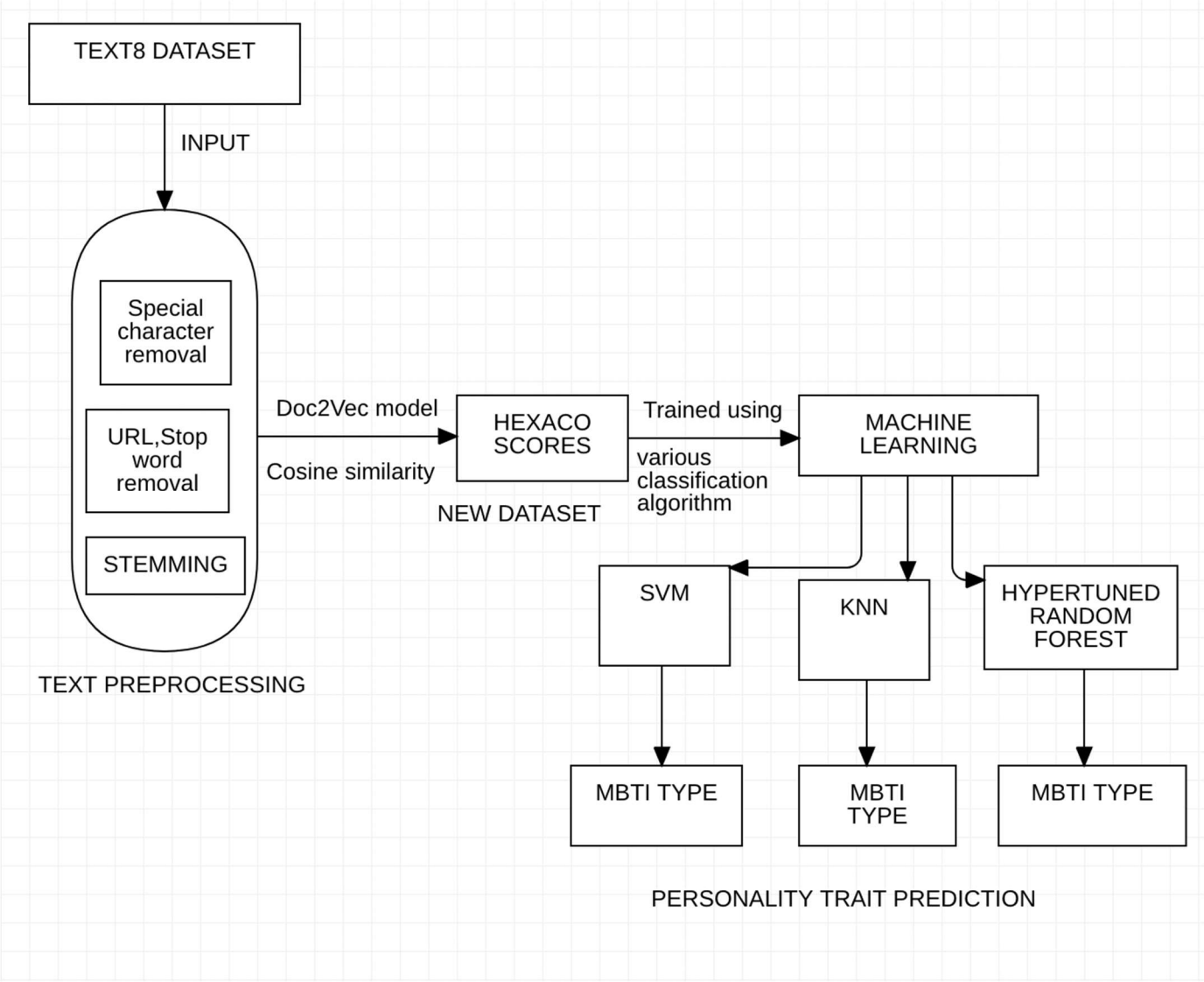


Figure 1: System Architecture

3.2 MODULES

Our system has been separated into two different modules, each of which assists us in efficiently predicting the personality of a user. We will take an input in the form of text which will be preprocessed using various NLP techniques. The steps in pre-processing includes:

- Special characters removal
- URL removal
- Stop word removal
- Stemming

After this, the first module of our system runs to generate HEXACO scores which act as an input to our second module which is a classification algorithm. Finally we have a personality type as a result which is given as an output by the second module.

HEXACO SCORES: This module accepts input as the self-description uploaded by the user along with their mbti personality types. This module uses genism library and text8 corpus, the doc2vec model to train the dataset. Using cosine-similarity the HEXACO scores of each user is calculated and a new dataset is hence created which will be for the further project.

MBTI PERSONALITY TYPE: The input for this module will be the HEXACO scores which came as an output of the previous module. This module is trained using a supervised classification algorithm and it predicts the mbti personality type on the basis of the training given to the machine learning model.

3.3 ALGORITHM/ TECHNIQUE USED

ALGORITHM:

```
function main()
{
        user_description=getUserDescription();
        processedData=preProcessData(user_description);
        HEXACO_SCORES=trainModel(pro Data)
        MODEL=RandomForestClassifier(Training_Data);
        result=MODEL.predictType(HEXACO_SCORES);
        return result;
}



function preProcessData(UserDescription)
{
        strip_html(UserDescription);
        removeBetweenSquareBrackets(UserDescription);
        stemmer(UserDescription);
        removeUrl(UserDescription);
        removeStopWord(UserDescription);
```

```
        return UserDescription;
}

function TrainModel(preProcessedData)
{
        Similarity=cosineSimilarity(preProcessedData);
        return Similarity;
}
```

TECHNIQUES:

GENSIM LIBRARY: The Gensim library is a free, open-source Python package that may be used in a variety of domains, including Natural Language Processing (NLP), Topic Modelling, and extracting Semantic Topics from Documents.It is one of the many packages, besides scikit-learn, that greatly aid in text processing and can manage large text corpora.We can automate the process of determining the semantic structure of text data by using the unsupervised algorithms present in gensim.For our system we have used the gensim library to create a Doc2Vec model for personality prediction which forms an essential part of our system.

DOC2VEC MODEL: Doc2Vec model is used to create a representation of a group of words taken present in a document. It is one of the NLP tools which is used for representing documents as a vector. Using unsupervised learning methods, this model offers a method for obtaining word embeddings from documents that are included in the corpus. For our system, we have implemented the Doc2vec model using the gensim package in Python.

COSINE SIMILARITY: is a statistic that may be used to assess how similar data items are, regardless of size. Cosine similarity treats each data item in a dataset as a vector. The following is the formula to get the-

$$cos(x, y) = \frac{xy}{||x|||y||}$$

Equation 1

where,

x . y = product (dot) of the vectors ‘x’ and ‘y’.

||x|| and ||y|| = length of the two vectors ‘x’ and ‘y’.

||x|| * ||y|| = cross product of the two vectors ‘x’ and ‘y’.

KNN MODEL: The k-nearest neighbours (KNN) is one of the supervised learning techniques that calculates the chance that a data point will belong to one group or another based on which grouping of all the data points occurs. This model can be used for both regression and classification. Here we have tried to use KNN model as a classification model for classifying the designation corresponding to the respective resumes of candidates. This model generally tries to classify the testing data correctly by computing the distance , mostly euclidean distance or Manhattan distance, between the testing data and training data points.

$$Euclidean\ distance\ (a, b) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

$$Manhattan\ distance\ (a, b) = |x2 - x1| + |y2 - y1|$$

Equation 2

where,

$a = coordinate\ point\ (x_1, y_1)$

$b = coordinate\ point\ (x_2, y_2)$

The next step is to choose K points that are most similar to the testing data. Finally, the KNN model determines which classes of the "K" training data the test data will belong to, and the class with the highest probability is gradually chosen.

SVM MODEL: SVM (Support Vector Machines) is one of the best supervised machine learning model which can be used both as a regression model as well as a classification model. SVM helps in

classification by choosing a proper decision boundary line called the hyperplane that maximizes the separation of the data points seen in each class label.

The kernels are a set of mathematical operations which are used by the SVM model. A kernel's primary task is to take data as input and change it into the required form. Different kernel functions are used by various SVM algorithms. RBF (radial basis function kernel) and polynomial kernel are the two most well-known kernels among them that we have picked for this application. The scalar product between two points in an incredibly appropriate feature space returned by the kernel functions.

1) *Polynomial Kernel Formula*

$$f(y, y_j) = (y \,.\, y_j + 1)^d$$

Equation 3

2) *Gaussian Radial Basis Formula*

$$f(y, y_j) = e^{\left(-\gamma * \left|\left|y - y_j\right|\right|^2\right)}$$

Equation 4

Where,

y and y_j refers to the data that has to be classified.

'.' shows the dot product of both the values.

d denotes the degree of the values.

f(y, y_j) indicating the decision boundary (hyperplane) to divide the specified classes.

The value of γ varies from 0 to 1 and the most preferred value for γ is 0.1.

RANDOM FOREST MODEL: Random forest model is a supervised learning model and , just like KNN and SVM, this model can also be used both as a regression model as well as a classification model. It is a learning based model which is built on an ensemble of trees. A group of decision trees

from a randomly chosen subset of the training data combines to form this classifier and in order to accurately classify the test object , it combines the decisions taken from multiple decision trees and casts the result. Random forest is actually termed as an ensemble algorithm because it combines more than one algorithm of which in this case, is decision tree. This model takes advantage of bagging algorithm, which is an ensemble meta technique that raises the precision of ML models.. As a result the shortcomings which might arise in a single decision tree classification model are gradually removed or eliminated by the random forest model. This model helps in improving the accuracy of the classifications and lowers dataset overfitting.

HYPERPARAMETER TUNING: This method is essential while we are working with machine learning models as they govern the general behaviour of a machine learning model. These are basically the model parameters which gradually summarise the important properties of the model and these parameters are gradually tuned to get the best output from the model. GridSearchCV and RandomizedSearchCV are the two most effective methods for hyperparameter tuning. Here for our system, we have performed hyperparameter tuning on the Random forest model and this gradually resulted in increasing the efficiency and accuracy of the Random Forest model. The machine learning model is assessed for a variety of hyperparameter values in the GridSearchCV technique. This method is known as GridSearchCV because it analyses a grid of hyperparameter values to find the optimum set of hyperparameters. For instance, let's say we wish to define two alternative sets of values for the Logistic Regression Classifier model's hyperparameters C and Alpha. The optimal model will be constructed using the grid search approach using a variety of different hyperparameter combinations. GridSearchCV has limitations since it only considers a finite set of hyperparameter parameters. RandomizedSearchCV addresses these limitations. To locate the ideal collection of hyperparameters, it travels randomly inside the grid. This strategy eliminates wasteful computation.

Chapter 4

EXPERIMENTAL SETUP

4.1 SOFTWARE REQUIREMENTS

- Windows 11
- python 3.9
- jupyter notebook
- Anaconda 3.0
- gensim package
- nltk library

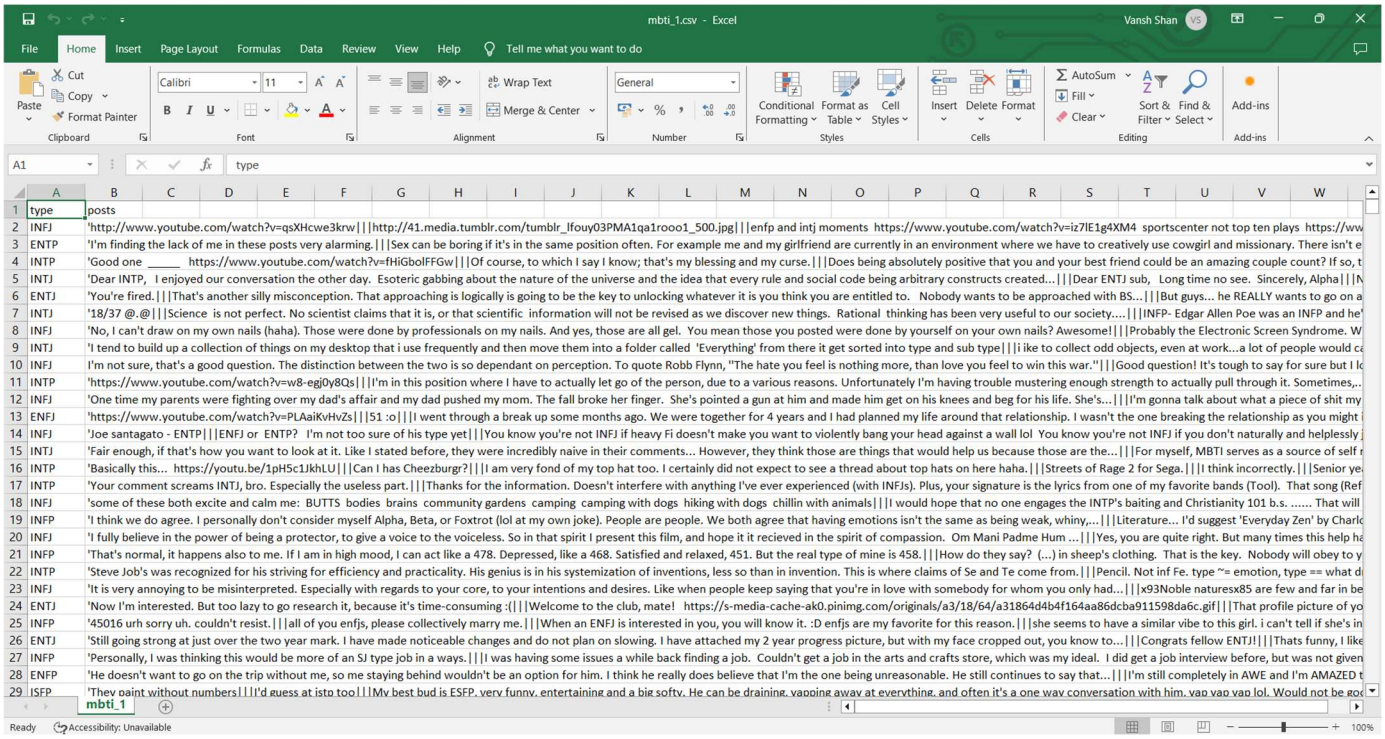
4.2 HARDWARE REQUIREMENTS

- i5 11th gen processor
- 8gb ram
- 512 ssd

Chapter 5

DETAILED CORPUS

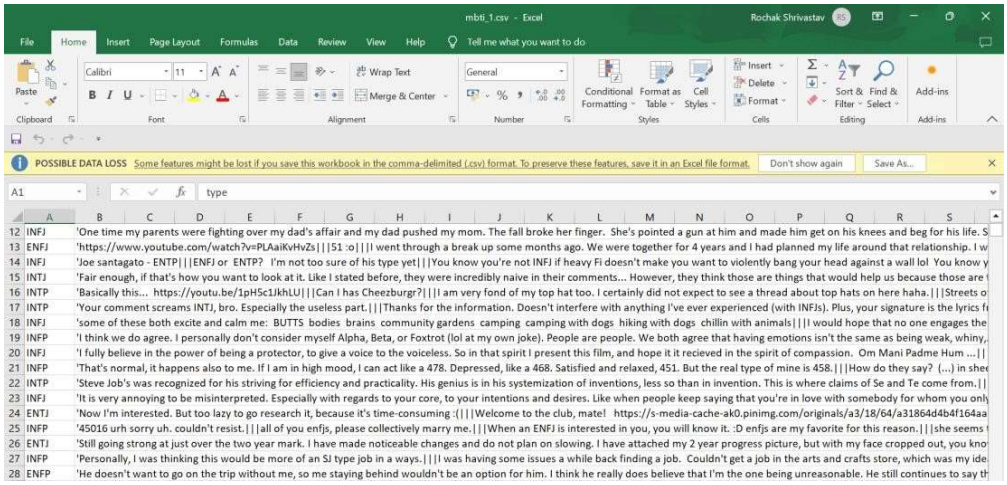
It is a 2X8676 corpus



Chapter 6

IMPLEMENTATION

Initial Dataset:



	A	B
12	INFJ	'One time my parents were fighting over my dad's affair and my dad pushed my mom. The fall broke her finger. She's pointed a gun at him and made him get on his knees and beg for his life. S
13	ENFJ	'https://www.youtube.com/watch?v=PLAaiKvHvZs|||51 :o|||I went through a break up some months ago. We were together for 4 years and I had planned my life around that relationship. I w
14	INFJ	'Joe santagato - ENTP|||ENFJ or ENTP? I'm not too sure of his type yet|||You know you're not INFJ if heavy Fi doesn't make you want to violently bang your head against a wall lol You know y
15	INTJ	'Fair enough, if that's how you want to look at it. Like I stated before, they were incredibly naive in their comments... However, they think those are things that would help us because those are t
16	INTP	'Basically this... https://youtu.be/1pH5c1JkhLU|||Can I has Cheezburgr?|||I am very fond of my top hat too. I certainly did not expect to see a thread about top hats on here haha.|||Streets o
17	INTP	'Your comment screams INTJ, bro. Especially the useless part.|||Thanks for the information. Doesn't interfere with anything I've ever experienced (with INFJs). Plus, your signature is the lyrics fr
18	INFJ	'some of these both excite and calm me: BUTTS bodies brains community gardens camping camping with dogs hiking with dogs chillin with animals|||I would hope that no one engages the
19	INFP	'I think we do agree. I personally don't consider myself Alpha, Beta, or Foxtrot (lol at my own joke). People are people. We both agree that having emotions isn't the same as being weak, whiny,
20	INFJ	'I fully believe in the power of being a protector, to give a voice to the voiceless. So in that spirit I present this film, and hope it it recieved in the spirit of compassion. Om Mani Padme Hum ...||
21	INFP	'That's normal, it happens also to me. If I am in high mood, I can act like a 478. Depressed, like a 468. Satisfied and relaxed, 451. But the real type of mine is 458.|||How do they say? (...) in shec
22	INTP	'Steve Job's was recognized for his striving for efficiency and practicality. His genius is in his systemization of inventions, less so than in invention. This is where claims of Se and Te come from.||
23	INFJ	'It is very annoying to be misinterpreted. Especially with regards to your core, to your intentions and desires. Like when people keep saying that you're in love with somebody for whom you onl
24	ENTJ	'Now I'm interested. But too lazy to go research it, because it's time-consuming :(|||Welcome to the club, mate! https://s-media-cache-ak0.pinimg.com/originals/a3/18/64/a31864d4b4f164aa
25	INFP	'45016 urh sorry uh. couldn't resist.|||all of you enfjs, please collectively marry me.|||When an ENFJ is interested in you, you will know it. :D enfjs are my favorite for this reason.||she seems t
26	ENTJ	'Still going strong at just over the two year mark. I have made noticeable changes and do not plan on slowing. I have attached my 2 year progress picture, but with my face cropped out, you kno
27	INFP	'Personally, I was thinking this would be more of an SJ type job in a ways.|||I was having some issues a while back finding a job. Couldn't get a job in the arts and crafts store, which was my ide
28	ENFP	'He doesn't want to go on the trip without me, so me staying behind wouldn't be an option for him. I think he really does believe that I'm the one being unreasonable. He still continues to say th

Preprocessing:

- Importing libraries:

```
#Importing the libraries

import pandas as pd
import sklearn as sk
import numpy as np

import nltk
nltk.download('stopwords')

from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
from nltk.tokenize.toktok import ToktokTokenizer

from bs4 import BeautifulSoup
import re,string,unicodedata


import nltk
from nltk.stem import WordNetLemmatizer
import re
```

Reading Dataset:

```
#Read the Dataset
data = pd.read_csv("mbti_1.csv")
```

[3]

```
data.head()
```

[4]

	type	posts
0	INFJ	'http://www.youtube.com/watch?v=qsXHcwe3krw|||...
1	ENTP	'I'm finding the lack of me in these posts ver...
2	INTP	'Good one _____ https://www.youtube.com/wat...
3	INTJ	'Dear INTP, I enjoyed our conversation the o...
4	ENTJ	'You're fired.|||That's another silly misconce...

- Removing HTML Tags based content:

```
#Removing the html strips
def strip_html(text):
    soup = BeautifulSoup(text, "html.parser")
    return soup.get_text()
```

- Replacing URL's:

```
def remove_urls(text):
    # Regular expression to match URLs
    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    # Replace URLs with an empty string
    clean_text = url_pattern.sub('', text)
    return clean_text
```

- Removing Square brackets and lemmatization:

```
#Removing the square brackets and notations
def remove_between_square_brackets(text):
    return re.sub('[^A-Za-z0-9/. ]', '', text)

#Lemmatizing the text
def simple_lemmatizer(text):
    lemmatizer = WordNetLemmatizer()
    text = ' '.join([lemmatizer.lemmatize(word) for word in text.split()])
    return text
```

- Setting and stopwords:

```
#set stopwords to english
stop=set(stopwords.words('english'))
print(stop)

#Tokenization of text
tokenizer=ToktokTokenizer()

#Setting English stopwords
stopword_list=nltk.corpus.stopwords.words('english')

#removing the stopwords
def remove_stopwords(text, is_lower_case=False):
    tokens = tokenizer.tokenize(text)
    tokens = [token.strip() for token in tokens]
    if is_lower_case:
        filtered_tokens = [token for token in tokens if token not in stopword_list]
    else:
        filtered_tokens = [token for token in tokens if token.lower() not in stopword_list]
    filtered_text = ' '.join(filtered_tokens)
    return filtered_text
```

- Applying function to review on column:

```
#Apply function on review column
data['posts']=data['posts'].apply(remove_urls)
data['posts']=data['posts'].apply(strip_html)
data['posts']=data['posts'].apply(remove_between_square_brackets)
#data['posts']=data['posts'].apply(remove_notations)
data['posts']=data['posts'].apply(simple_lemmatizer)
data['posts']=data['posts'].apply(remove_stopwords)
```

```
{'are', "shan't", "wasn't", 'before', 'yourself', "you'll", 'after', 'same', 'doesn', 'each', "you'd", 'd', 'mor
```

Printing Data:

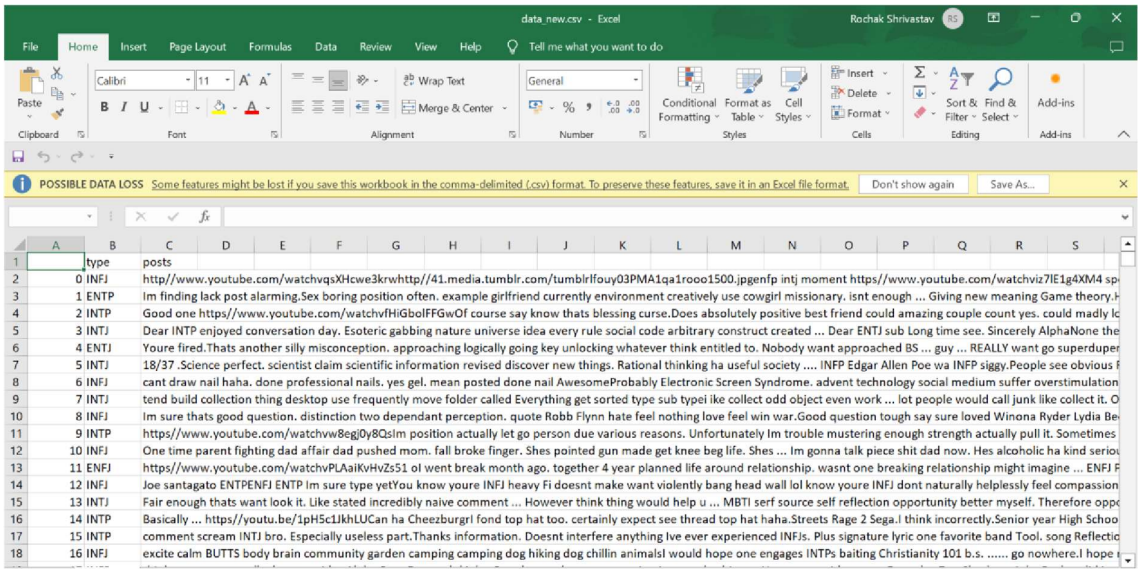
```
print(data.posts)
```

```
0       intj moment sportscenter top ten play pranksWh...
1       Im finding lack post alarming.Sex boring posit...
2       Good one course say know thats blessing curse....
3       Dear INTP enjoyed conversation day. Esoteric g...
4       Youre fired.Thats another silly misconception....
                              ...
8670    always think cat Fi doms reason. website becom...
8671    ... thread already exists someplace else doe h...
8672    many question things. would take purple pill. ...
8673    conflicted right come wanting children. honest...
8674    ha long since personalitycafe although doesnt ...
Name: posts, Length: 8675, dtype: object
```

- Saving data:

```
#Save the data
data.to_csv("data_new.csv")
```

[10]



Training Model:

- Importing Libraries:

```
[1]  import gensim

[2]  import joblib

[3]  import pandas as pd
     from sklearn.metrics.pairwise import cosine_similarity

[4]  import gensim.downloader as api
     dataset = api.load("text8")
     data = [i for i in dataset]

...  [==================================================] 100.0% 31.6/31.6MB downloaded
```

- Building model:

```
def tagged_document(list_of_list_of_words):
    for i, list_of_words in enumerate(list_of_list_of_words):
        yield gensim.models.doc2vec.TaggedDocument(list_of_words, [i])

training_data = list(tagged_document(data))
model = gensim.models.doc2vec.Doc2Vec(vector_size=40, min_count=2, epochs=30)

model.build_vocab(training_data)
model.train(training_data, total_examples=model.corpus_count, epochs=model.epochs)
```

```
model
```

```
<gensim.models.doc2vec.Doc2Vec at 0x2791f247f10>
```

- Saving Model:

```
joblib.dump(model, 'model_2.pkl')
```

```
['model_2.pkl']
```

Creating dataset of cosine similarity values based on HEXACO scores:

```
my_model = joblib.load('model_2.pkl')
```
Python

```
sentences =["honesty", "emotionality", "extraversion", "agreeableness", "conscientiousness","openness to exper
```
Python

```
data =pd.read_csv("data_new.csv")
```
Python

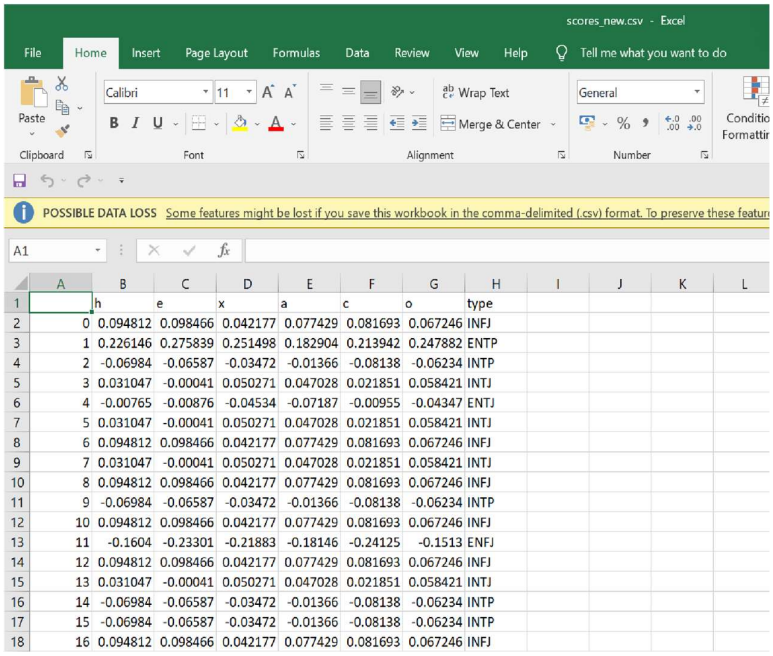
```
x = data.iloc[0,2]
```
Python

```
vectors1 = [my_model.infer_vector([word for word in sent]).reshape(1,-1) for sent in sentences]
vectors2 = [my_model.infer_vector([word for word in x]).reshape(1,-1)]
```
Python

```
sim_values=[]
for i in range(len(data)):
    x=data.iloc[i,2]
    vectors2 = [my_model.infer_vector([word for word in x]).reshape(1,-1)]
    array=[]
    for j in range(0,6):
        similarity = cosine_similarity(vectors1[j],vectors2[0])
        array.append(similarity[0][0])
    array.append(data.iloc[i,1])
    sim_values.append(array)
df=pd.DataFrame(data=sim_values,columns=['h','e','x','a','c','o','type'])
df.to_csv("scores.csv")
```

```
df=pd.DataFrame(data=sim_values,columns=['h','e','x','a','c','o','type'])
df.to_csv("scores.csv")
```

HEXACO scores CSV file:



scores_new.csv - Excel

File Home Insert Page Layout Formulas Data Review View Help Tell me what you want to do

POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these featur

A1

	A	B	C	D	E	F	G	H	I	J	K	L
1		h	e	x	a	c	o	type				
2	0	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				
3	1	0.226146	0.275839	0.251498	0.182904	0.213942	0.247882	ENTP				
4	2	-0.06984	-0.06587	-0.03472	-0.01366	-0.08138	-0.06234	INTP				
5	3	0.031047	-0.00041	0.050271	0.047028	0.021851	0.058421	INTJ				
6	4	-0.00765	-0.00876	-0.04534	-0.07187	-0.00955	-0.04347	ENTJ				
7	5	0.031047	-0.00041	0.050271	0.047028	0.021851	0.058421	INTJ				
8	6	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				
9	7	0.031047	-0.00041	0.050271	0.047028	0.021851	0.058421	INTJ				
10	8	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				
11	9	-0.06984	-0.06587	-0.03472	-0.01366	-0.08138	-0.06234	INTP				
12	10	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				
13	11	-0.1604	-0.23301	-0.21883	-0.18146	-0.24125	-0.1513	ENFJ				
14	12	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				
15	13	0.031047	-0.00041	0.050271	0.047028	0.021851	0.058421	INTJ				
16	14	-0.06984	-0.06587	-0.03472	-0.01366	-0.08138	-0.06234	INTP				
17	15	-0.06984	-0.06587	-0.03472	-0.01366	-0.08138	-0.06234	INTP				
18	16	0.094812	0.098466	0.042177	0.077429	0.081693	0.067246	INFJ				

Training ML Models based on HEXACO scores:

- Importing libraries, data and splitting it for testing and training:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split, GridSearchCV
```

```
data = pd.read_csv("scores_new.csv")
```

```
X = data[['h', 'e', 'x', 'a', 'c' , 'o']]
y = data['type']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=0)
```

Training Random forest model:

```
clf = RandomForestClassifier(n_estimators = 100, max_depth = 5)

# Training the model on the training dataset
# fit function is used to train the model using the training sets as parameters
clf.fit(X_train, y_train)

# performing predictions on the test dataset
y_pred = clf.predict(X_test)

# metrics are used to find accuracy or error
from sklearn import metrics
print()

# using metrics module for accuracy calculation
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))

# print classification report
print(classification_report(y_test, y_pred))
```

[52]

```
ACCURACY OF THE MODEL:  0.9565885516711486
              precision    recall  f1-score   support

        ENFJ       1.00      1.00      1.00        53
        ENFP       1.00      1.00      1.00       201
        ENTJ       0.00      0.00      0.00        60
        ENTP       1.00      1.00      1.00       206
        ESFJ       1.00      1.00      1.00        17
        ESFP       1.00      1.00      1.00        14
        ESTJ       1.00      1.00      1.00        10
        ESTP       1.00      1.00      1.00        25
        INFJ       1.00      1.00      1.00       450
        INFP       1.00      1.00      1.00       542
        INTJ       1.00      1.00      1.00       355
        INTP       1.00      1.00      1.00       397
        ISFJ       0.00      0.00      0.00        53
        ISFP       0.41      1.00      0.58        78
        ISTJ       1.00      1.00      1.00        56
        ISTP       1.00      1.00      1.00        86

    accuracy                           0.96      2603
   macro avg       0.84      0.88      0.85      2603
weighted avg       0.94      0.96      0.94      2603
```

- Training KNN Model:

KNN

```
model = KNeighborsClassifier(n_neighbors=2000)

# Train the model using the training sets
model.fit(X_train,y_train)
y_pred = model.predict(X_test)

# metrics are used to find accuracy or error
from sklearn import metrics
print()

# using metrics module for accuracy calculation
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
ACCURACY OF THE MODEL:  0.6699961582789089
              precision    recall  f1-score   support

        ENFJ       0.00      0.00      0.00        53
        ENFP       0.00      0.00      0.00       201
        ENTJ       0.00      0.00      0.00        60
        ENTP       0.00      0.00      0.00       206
        ESFJ       0.00      0.00      0.00        17
        ESFP       0.00      0.00      0.00        14
        ESTJ       0.00      0.00      0.00        10
        ESTP       0.00      0.00      0.00        25
        INFJ       1.00      1.00      1.00       450
        INFP       0.61      1.00      0.76       542
        INTJ       0.57      1.00      0.73       355
        INTP       0.62      1.00      0.76       397
        ISFJ       0.00      0.00      0.00        53
        ISFP       0.00      0.00      0.00        78
        ISTJ       0.00      0.00      0.00        56
        ISTP       0.00      0.00      0.00        86

    accuracy                           0.67      2603
   macro avg       0.17      0.25      0.20      2603
weighted avg       0.47      0.67      0.55      2603
```

- Training SVM Model:

•

SVM

```
from sklearn import svm
clf = svm.SVC(kernel= 'sigmoid')
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
print(classification_report(y_test, y_pred))
```

	precision	recall	f1-score	support
ENFJ	0.00	0.00	0.00	53
ENFP	1.00	1.00	1.00	201
ENTJ	1.00	1.00	1.00	60
ENTP	0.00	0.00	0.00	206
ESFJ	0.00	0.00	0.00	17
ESFP	1.00	1.00	1.00	14
ESTJ	0.00	0.00	0.00	10
ESTP	0.00	0.00	0.00	25
INFJ	0.59	1.00	0.74	450
INFP	0.96	1.00	0.98	542
INTJ	1.00	1.00	1.00	355
INTP	0.78	1.00	0.88	397
ISFJ	1.00	1.00	1.00	53
ISFP	1.00	1.00	1.00	78
ISTJ	0.00	0.00	0.00	56
ISTP	0.00	0.00	0.00	86
accuracy			0.83	2603
macro avg	0.52	0.56	0.54	2603
weighted avg	0.71	0.83	0.76	2603

Hyper-Parameter tuning:

•

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(random_state = 42, max_depth=5)
from pprint import pprint
# Look at parameters used by our current forest
print('Parameters currently in use:\n')
pprint(rf.get_params())
```

```
Parameters currently in use:

{'bootstrap': True,
 'ccp_alpha': 0.0,
 'class_weight': None,
 'criterion': 'gini',
 'max_depth': 5,
 'max_features': 'auto',
 'max_leaf_nodes': None,
 'max_samples': None,
 'min_impurity_decrease': 0.0,
 'min_samples_leaf': 1,
 'min_samples_split': 2,
 'min_weight_fraction_leaf': 0.0,
 'n_estimators': 100,
 'n_jobs': None,
 'oob_score': False,
 'random_state': 42,
 'verbose': 0,
 'warm_start': False}
```

```
param_grid = {
    'n_estimators': [100, 200, 500],
    'max_features': ['auto', 'sqrt'],
    'max_depth' : [2,3,4,5],
    'criterion' :['gini', 'entropy']
}
CV_rfc = GridSearchCV(estimator=rf, param_grid=param_grid, cv= 5)
CV_rfc.fit(X_train, y_train)
```

```
GridSearchCV(cv=5,
             estimator=RandomForestClassifier(max_depth=100, n_estimators=200),
             param_grid={'criterion': ['gini', 'entropy'],
                         'max_depth': [2, 3, 4, 5],
                         'max_features': ['auto', 'sqrt'],
                         'n_estimators': [100, 200, 500]})
```

```
CV_rfc.best_params_
```

Pyth

```
{'criterion': 'entropy',
 'max_depth': 5,
 'max_features': 'auto',
 'n_estimators': 100}
```

```
rf = RandomForestClassifier(max_depth=5, max_features = 'auto', n_estimators = 100, criterion = 'entropy')
```

Pyth

```
rf.fit(X_train,y_train)
```

Pyth

```
RandomForestClassifier(criterion='entropy', max_depth=5)
```

Making Prediction for out text:

```
y_pred = rf.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
print(classification_report(y_test, y_pred))
```

•

	precision	recall	f1-score	support
ENFJ	1.00	1.00	1.00	53
ENFP	1.00	1.00	1.00	201
ENTJ	1.00	1.00	1.00	60
ENTP	1.00	1.00	1.00	206
ESFJ	1.00	1.00	1.00	17
ESFP	1.00	1.00	1.00	14
ESTJ	1.00	1.00	1.00	10
ESTP	1.00	1.00	1.00	25
INFJ	1.00	1.00	1.00	450
INFP	1.00	1.00	1.00	542
INTJ	1.00	1.00	1.00	355
INTP	1.00	1.00	1.00	397
ISFJ	0.00	0.00	0.00	53
ISFP	0.60	1.00	0.75	78
ISTJ	1.00	1.00	1.00	56
ISTP	1.00	1.00	1.00	86
accuracy			0.98	2603
macro avg	0.91	0.94	0.92	2603
weighted avg	0.97	0.98	0.97	2603

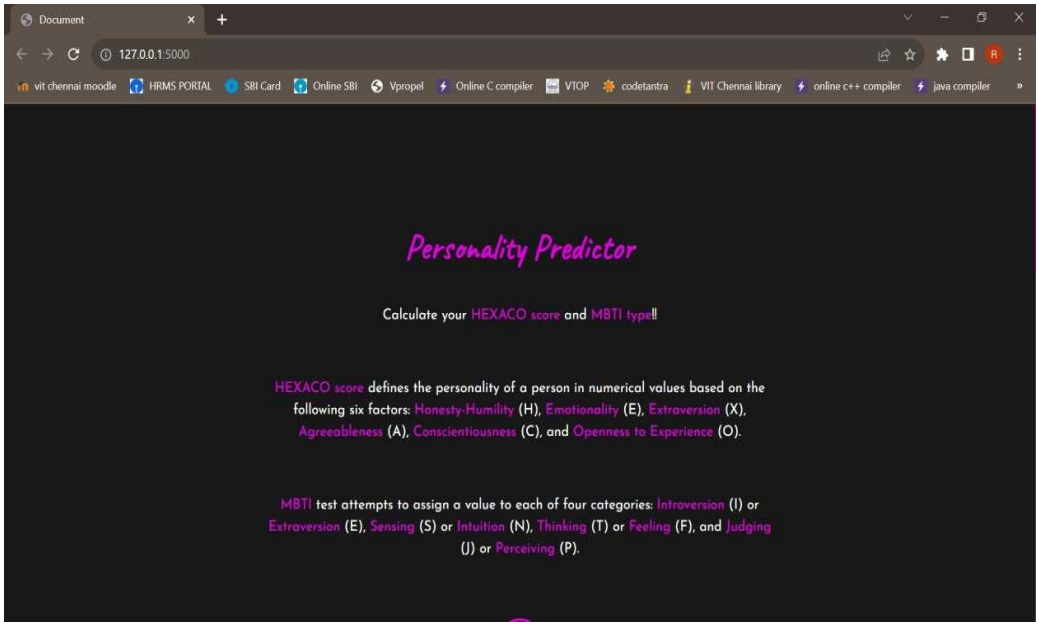
•

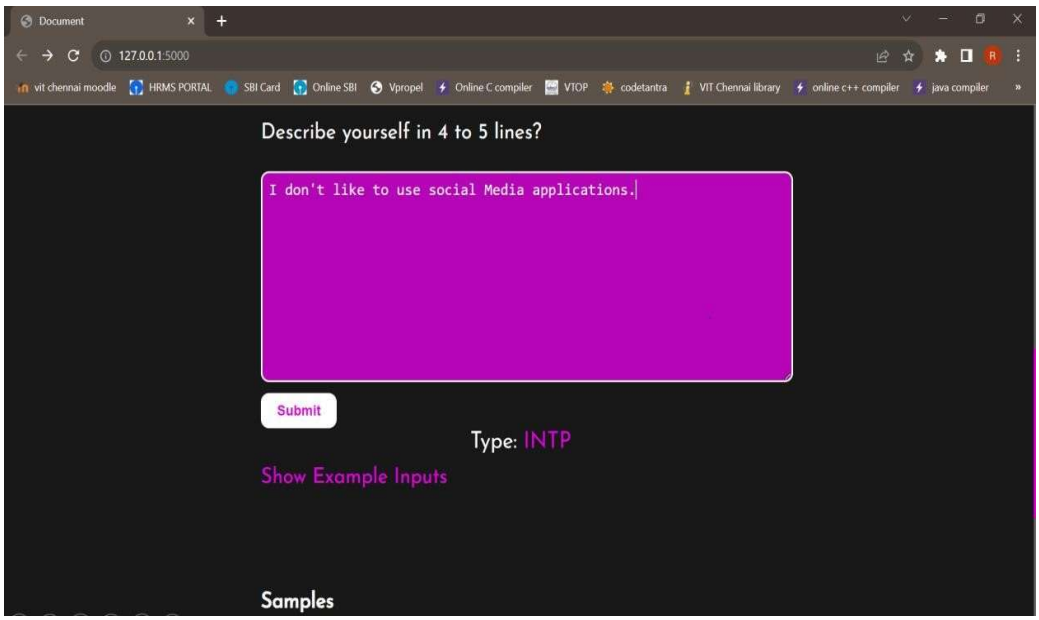
Testing the model using user interface:

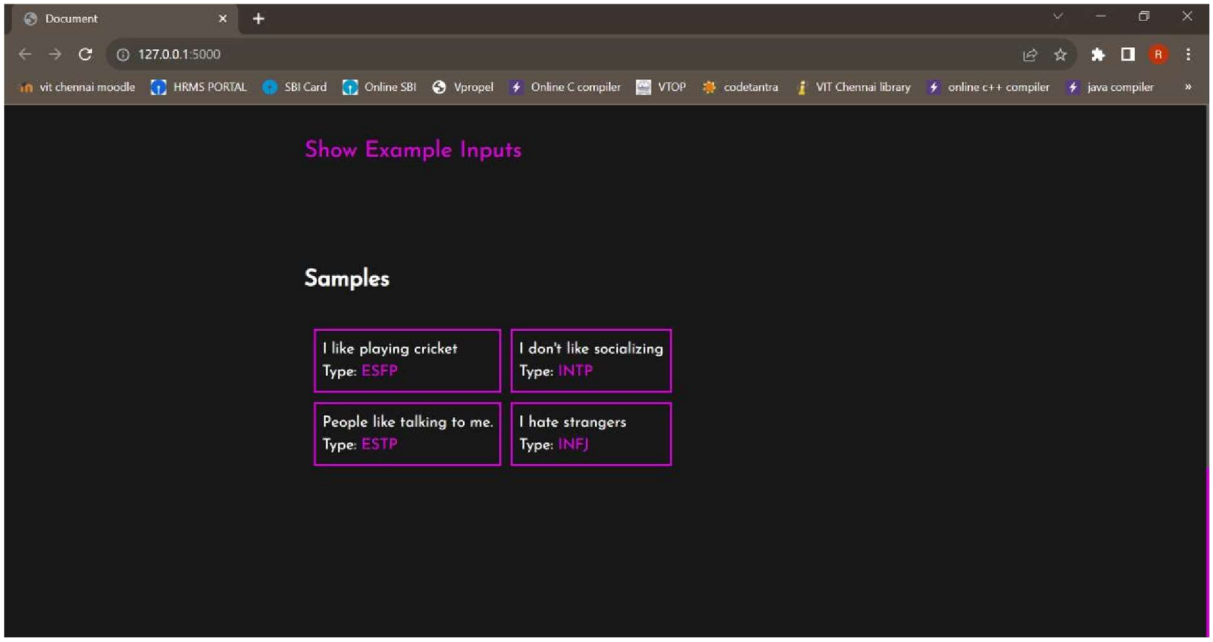
```
app = Flask(__name__)

@app.route('/', methods=['GET',"POST"])
def index():
    if request.method == "POST":
        about = request.form.get('about') #calculate HEXACO score and MBTI type and store them in variables
        (hexaco, mbti, descr) = backend_call(about)
        return render_template('index.html', scroll='result', hexaco=hexaco, mbti=mbti, descr=descr) #return
    return render_template('index.html')

if __name__ == "__main__":
    app.run()
```







Lexical Processing

```
import spacy
nlp = spacy.load('en_core_web_sm')    #we are loading english ('en') core with size small ('sm')
from nltk import word_tokenize
from nltk.corpus import stopwords
import string
punctuations = string.punctuation
```

[1]

+ Code + Markdown

```
# NLTK implementation
sent = "The main challenge, is to start!"
stop = stopwords.words('english') + list(punctuations)    #removing unwanted punctuations also along with stopwords
print("NLTK implementation Result: ", [i for i in word_tokenize(sent) if i not in stop])

# Spacy implementation
doc = nlp(sent)    # Create a Doc object
spacy_stopwords = spacy.lang.en.stop_words.STOP_WORDS
print("Spacy implementation Result: ",[token.text for token in doc if token.text not in spacy_stopwords and token.text not in punctuations])
```

[2]

```
NLTK implementation Result:  ['The', 'main', 'challenge', 'start']
Spacy implementation Result:  ['The', 'main', 'challenge', 'start']
```

Stemming

```
# NLTK implementation
from nltk.stem import PorterStemmer
porter = PorterStemmer()
print("NLTK implementation result: ",{"running": porter.stem("running"),"saw": porter.stem("saw"),"troubling": porter.stem("troubling")})
```

[3]

```
NLTK implementation result:  {'running': 'run', 'saw': 'saw', 'troubling': 'troubl'}
```

Lemmatization

```
#NLTK implementation
from nltk.stem import WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()
print("NLTK implementation result: ",wordnet_lemmatizer.lemmatize('saw',pos='v'))
#Spacy implementation
from spacy.lemmatizer import Lemmatizer
from spacy.lang.en import LEMMA_INDEX, LEMMA_EXC, LEMMA_RULES
lemmatizer = Lemmatizer(LEMMA_INDEX, LEMMA_EXC, LEMMA_RULES)
lemmas = lemmatizer(u'saw', u'VERB')
print("Spacy implementation result: ", lemmas[0])
```

[4]

```
NLTK implementation result:  saw
Spacy implementation result:  see
```

TF-IDF

```
from sklearn.feature_extraction.text import TfidfVectorizer
corpus = [
    'This is the first document.',
    'This document is the second document.',
    'And this is the third one.',
    'Is this the first document?',
]
vectorizer = TfidfVectorizer(stop_words=stop) #stop was defined initially using stopwords from NLTK
X = vectorizer.fit_transform(corpus)
print(vectorizer.get_feature_names())
print(X)
```

[5]

```
['document', 'first', 'one', 'second', 'third']
  (0, 1)        0.7772211620785797
  (0, 0)        0.6292275146695526
  (1, 0)        0.78722297610404
  (1, 3)        0.6166684570284895
  (2, 4)        0.7071067811865476
  (2, 2)        0.7071067811865476
  (3, 1)        0.7772211620785797
  (3, 0)        0.6292275146695526
```

Edit Distance

		m	o	n	k	e	y
	0	1	2	3	4	5	6
m	1	0	1	2	3	4	5
o	2	1	0	1	2	3	4
n	3	2	1	0	1	2	3
e	4	3	2	1	1	1	2
y	5	4	3	2	2	2	1

```
import re                                   #regular expression
from collections import Counter             #creating frequency count dict
import heapq                                #for selecting n largest
import os
```

```
os.listdir("../input/")
```

```
['big.txt']
```

```
def words(text): return re.findall(r'\w+', text.lower())

WORDS = Counter(words(open('../input/big.txt').read()))
WORDS.most_common(10)
```

```
[('the', 79809),
 ('of', 40024),
 ('and', 38312),
 ('to', 28765),
 ('in', 22023),
 ('a', 21124),
 ('that', 12512),
 ('he', 12401),
 ('was', 11410),
 ('it', 10681)]
```

```
def P(word, N=sum(WORDS.values())):
    "Probability of `word`."
    return WORDS[word] / N

def correction(word):
    "Most probable spelling correction for word."
    listProb = {word: P(word) for word in candidates(word)}
    return listProb, max(candidates(word), key=P)

def candidates(word):
    "Generate possible spelling corrections for word."
    return (known([word]) or known(edits1(word)) or known(edits2(word)) or [word])

def known(words):
    "The subset of `words` that appear in the dictionary of WORDS."
    return set(w for w in words if w in WORDS)
```

```
def edits1(word):
    "All edits that are one edit away from `word`."
    letters    = 'abcdefghijklmnopqrstuvwxyz'
    splits     = [(word[:i], word[i:])    for i in range(len(word) + 1)]
    deletes    = [L + R[1:]               for L, R in splits if R]
    transposes = [L + R[1] + R[0] + R[2:] for L, R in splits if len(R)>1]
    replaces   = [L + c + R[1:]           for L, R in splits if R for c in letters]
    inserts    = [L + c + R               for L, R in splits for c in letters]
    return set(deletes + transposes + replaces + inserts)

def edits2(word):
    "All edits that are two edits away from `word`."
    return (e2 for e1 in edits1(word) for e2 in edits1(e1))
```

```
    def get_correct_word(word):
        corrected_word = next(iter(correction(word)[0]))
        print("Word passed: ", word, " Word corrected To: ", corrected_word)
        return corrected_word

    print(get_correct_word('speling'))
```

```
Word passed:  speling  Word corrected To:  spelling
spelling
```

Chapter 7

PERFORMANCE EVALUATION

For evaluating the job classification module, we have identified four parameters for the multiclass-classification. The four parameters include precision, recall, F-measure and accuracy. Precision can be defined as the ratio of the numbers of records correctly classified as true to the number of records classified as true. Recall can be defined as the ratio of the number of records correctly classified as true to the number of records which are actually true. Now, for multiclass classification all precision and recall for every class is calculated individually and then the average is taken as macro or simple average or weighted average.F-measure can be defined as the harmonic mean of the precision and the recall value. It is often used as a very significant metric. Accuracy can be defined as the total number of correct classifications to the total number of records.

a) *Precision:* It is an evaluation metric which determines how many times our model predicted positive values correctly.

$$\boldsymbol{Precision} = \frac{\boldsymbol{True\ Positive\ (TP)}}{\boldsymbol{True\ Positive\ (TP)\ +\ False\ Positive\ (FP)}}$$

Equation 5

b) *Recall:* It is an evaluation metric which determines that out of the total actual positive values, how many of them were predicted correctly as positive.

$$\boldsymbol{Recall} = \frac{\boldsymbol{True\ Positive\ (TP)}}{\boldsymbol{True\ Positive\ (TP)\ +\ False\ Negative\ (FN)}}$$

Equation 6

c) *F-measure:* This evaluation metric is the harmonic mean of precision and recall. As in some cases, both false positive values and false negative values might be very important, so in order to create a balance between false positive and negative values f1 metric might be useful or beneficial.

$$\boldsymbol{F-measure(F1-Score)} = \mathbf{2} * \left[\frac{\boldsymbol{precision\ *\ recall}}{\boldsymbol{precision\ +\ recall}}\right]$$

Equation 7

d) *Accuracy:* It is an evaluation metric or parameter which defines the relationship of a measured value to a true value.It helps in determining the number of correct predictions made by the model.

$$\boldsymbol{Accuracy} = \frac{\boldsymbol{TP\ +\ TN}}{\boldsymbol{TP\ +\ TN\ +\ FP\ +\ FN}}$$

Equation 8

•

where,

TP - True Positive

TN - True Negative

FP - False Positive

FN - False Negative

Chapter 8

RESULTS AND DISCUSSION

Initially we tried to perform data analysis on the mbti data to check the quality of the dataset. Here we plotted the distribution of different mbti personality types present in the dataset using line graph. Applying the line-graph we go the following result:

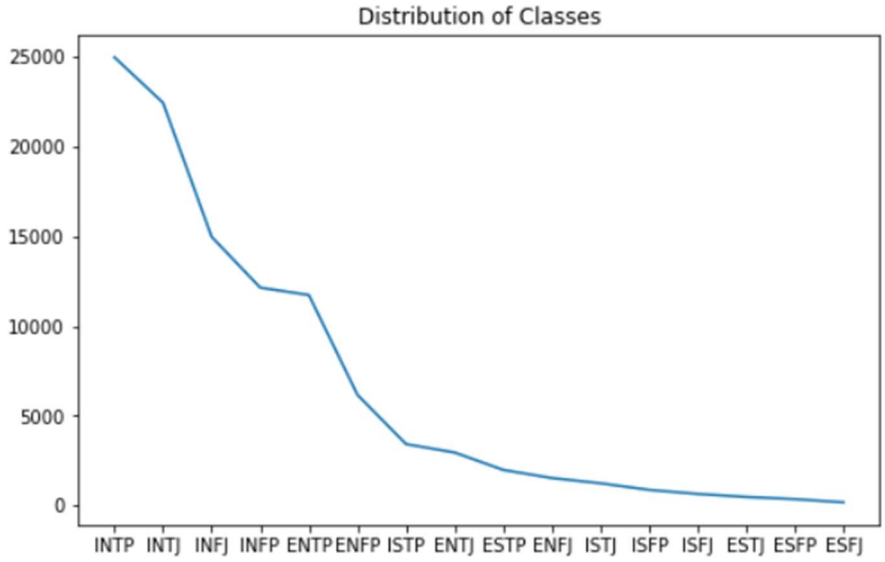


Figure 2: Line graph between the MBTI types and number of rows

Figure 2 shows that the data is highly imbalanced. But the minimum value for each mbti type is 500.

Also, once after data pre-processing and dataset training, we converted the textual data into vector-like representation, which is actually the HEXACO score for that text. After calculating the HEXACO scores for each of the 8675 rows we calculated the correlation among each of the HEXACO traits to observe how dependent the attributes of our new data is. After evaluation we got the following result:

Figure 3: Correlation matrix between HEXACO traits

Figure 3 shows that each of the six attributes are highly correlated with a correlation value of about 0.99(approx.).

We tried applying four algorithms including KNN Classifier, SVM (with Sigmoid kernel), Random Forest and then hyperparameter tuned random forest algorithms. Table 1 shows the results of the experiments conducted.

Sl. No.	Algorithm	Precision	Recall	F-measure	Accuracy
1	KNN Classifier	0.96	0.98	0.96	0.97
2	SVM	0.53	0.56	0.54	0.69
3	Random Forest	0.97	0.98	0.98	0.98
4	Hyperparameter Tuned Random Forest	0.99	0.99	0.99	0.99

Table 1: Comparison table for different classification models

Table 1 shows that the random forest was giving better results compared to KNN classifier and Support Vector Machine. And performing hyperparameter tunning in the random forest model we get even better result (99% accuracy). Thus, we will be using the hyperparameter tunned random forest model for the deployment of our model.

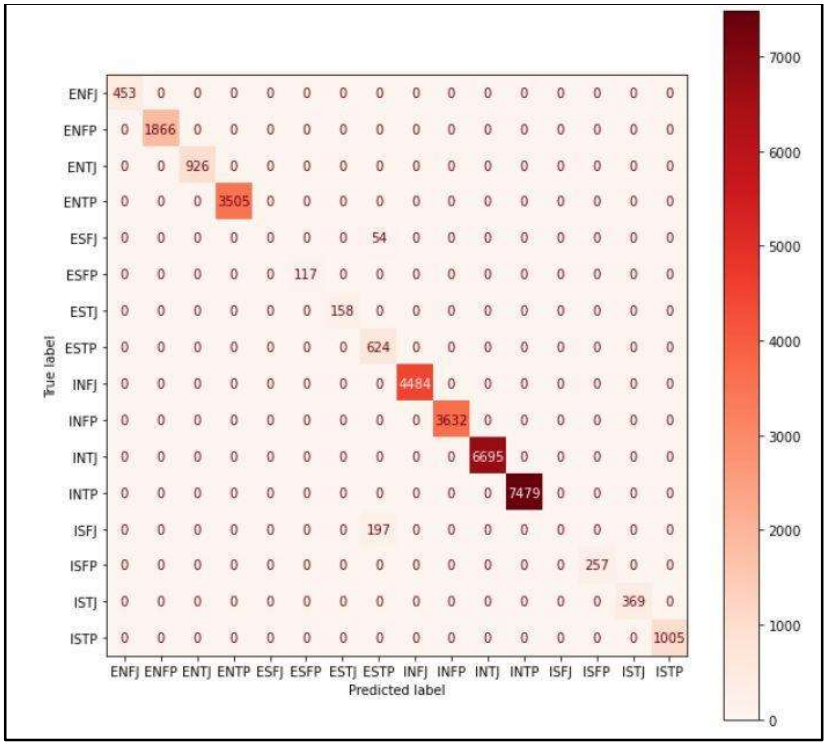


Figure 4: Confusion matrix for the dataset

Figure 4 shows the confusion matrix for the hypertuned random forest model. The right diagonal of the matrix show the true positive value for the predicted and actual value. The heat map shows the density of the values for that particular MBTI type.

	precision	recall	f1-score	support
ENFJ	1.00	1.00	1.00	453
ENFP	1.00	1.00	1.00	1866
ENTJ	1.00	1.00	1.00	926
ENTP	1.00	1.00	1.00	3505
ESFJ	0.00	0.00	0.00	54
ESFP	1.00	1.00	1.00	117
ESTJ	1.00	1.00	1.00	158
ESTP	0.71	1.00	0.83	624
INFJ	1.00	1.00	1.00	4484
INFP	1.00	1.00	1.00	3632
INTJ	1.00	1.00	1.00	6695
INTP	1.00	1.00	1.00	7479
ISFJ	0.00	0.00	0.00	197
ISFP	1.00	1.00	1.00	257
ISTJ	1.00	1.00	1.00	369
ISTP	1.00	1.00	1.00	1005
accuracy			0.99	31821
macro avg	0.86	0.88	0.86	31821
weighted avg	0.99	0.99	0.99	31821

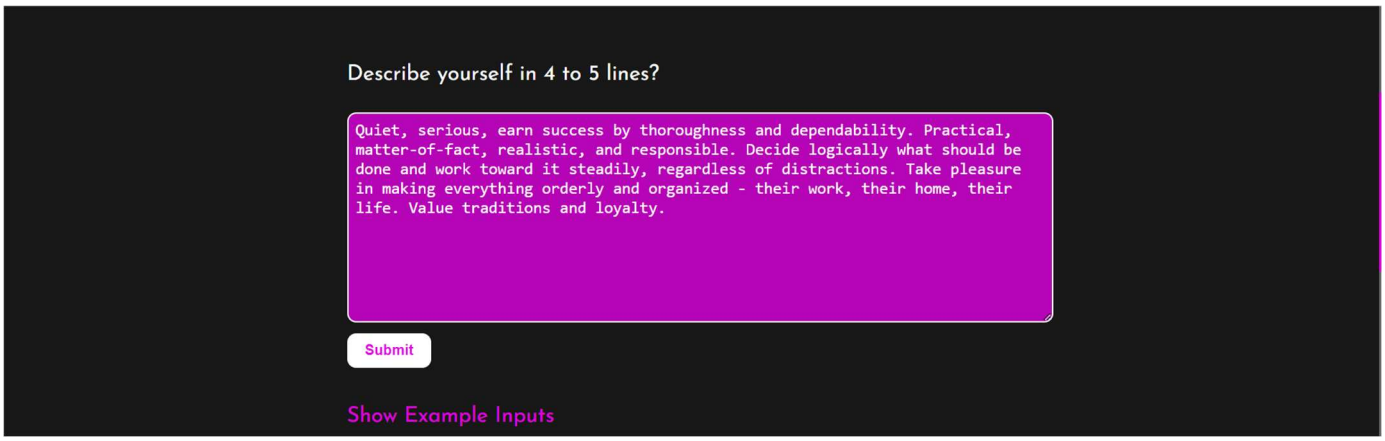
Table 2: Classification Report for Hypertuned Random Forest model

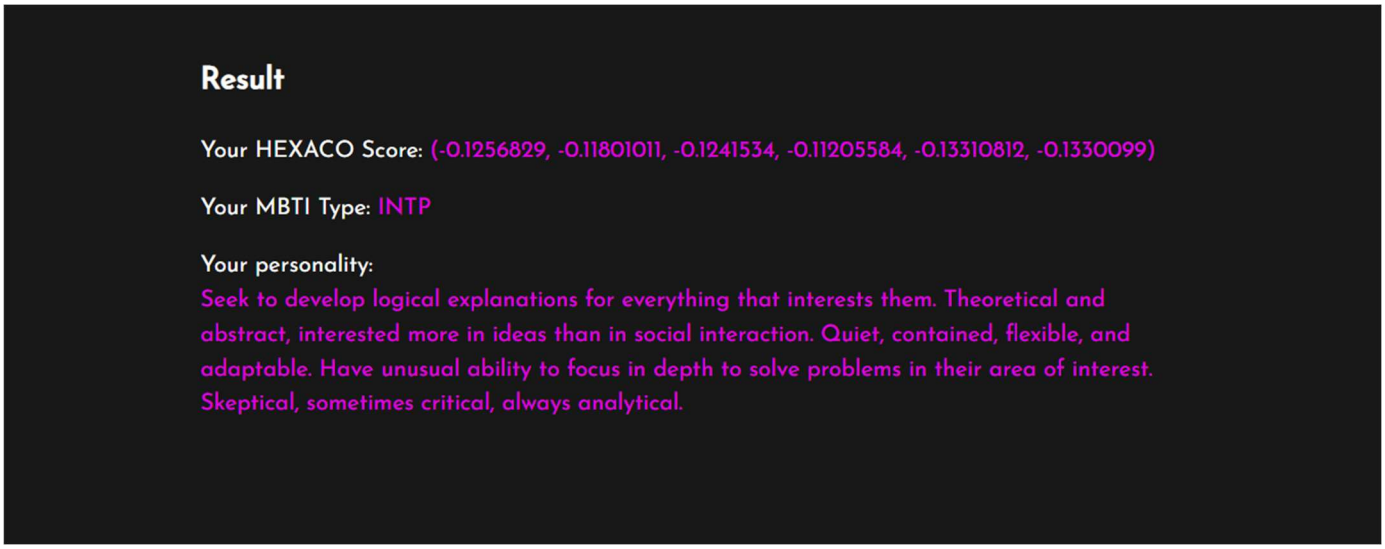
Table 2 shows the classification report of the hypertuned random forest model for the training dataset. The comparison is done on the basis of accuracy, precision, recall and f1-score for each of the MBTI type. The weighted average is also calculated for the dataset. The accuracy for this training dataset is approximately 99%.

Chapter 9

TEST CASES AND VALIDATION

Test case 1:





Test case 2:

Describe yourself in 4 to 5 lines?

A good person is characterized by their kindness and compassion, always ready to lend a helping hand to those in need. They exhibit empathy, understanding, and respect for others, fostering a sense of community and unity. They act with integrity, upholding moral and ethical values in their interactions. A good person seeks to make the world a better place through their positive influence and selfless actions, inspiring others to follow their example. Their genuine and selfless nature shines through in their daily deeds, promoting harmony and goodwill.

Submit

Show Example Inputs

Result

Your HEXACO Score: (0.20385557, 0.22459145, 0.25097504, 0.22429337, 0.21156734, 0.20873353)

Your MBTI Type: ENTP

Your personality:
Quick, ingenious, stimulating, alert, and outspoken. Resourceful in solving new and challenging problems. Adept at generating conceptual possibilities and then analyzing them strategically. Good at reading other people. Bored by routine, will seldom do the same thing the same way, apt to turn to one new interest after another.

Chapter 10

CONCLUSION & FUTURE WORK

As we can see from the above results, with hyperparameter tuning random forest models we are able to achieve a higher accuracy along with precision and recall and so we have used the corresponding model for predicting the job designation of the candidates based on the provided resumes. So we can finally conclude from the above results that we have been able to successfully build and design a system which, with the help of various models like the Doc2Vec model, the hyperparameter tuned random forest model and the regression score equation, will gradually help in in predicting the accurate MBTI personality type of any person based on how they define themself.

In the near future, the system which we have built may be transformed into a fully workable web application or a mobile application and thus our system can be then utilised on a wider range by different users who are curious to know their personality type. Also this can be used by the interviewers and different organisations for finding the best candidate based on the interview questions answered by that person. In addition to that, presently our system performs prediction and analysis by utilising only the mbti dataset and not the data present as url links, like youtube, medium or github links, in the dataset. In future there can be some modifications to our system where along with the mbti data we can make use of BeautifulSoup and Selenium packages in python package which will help in scraping data from the corresponding instagram accounts and linkedin accounts and combine the data along with the mbti data and provide it as input to our system. This way we can get more proper and accurate results or predictions from our system and thus the efficiency of the system will be enhanced.

REFERENCES

[1] S. K. Nivetha, M. Geetha, R. S. Latha, K. Sneha, S. Sobika and C. Yamuna, "Personality Prediction for Online Interview," 2022 International Conference on Computer Communication and Informatics (ICCCI), 2022, pp. 1-4, doi: 10.1109/ICCCI54379.2022.9740980.

[2] Khan, Alam Sher, et al. "Personality classification from online text using machine learning approach." International Journal of Advanced Computer Science and Applications 11.3 (2020).

[3] Christian, H., Suhartono, D., Chowanda, A. et al. Text based personality prediction from multiple social media data sources using pre-trained language model and model averaging. J Big Data 8, 68 (2021). https://doi.org/10.1186/s40537-021-00459-1

[4] Holtrop, Djurre, et al. "Exploring the application of a text-to-personality technique in job interviews." European Journal of Work and Organizational Psychology (2022): 1-18.

[5] Vora, Hetal, Mamta Bhamare, and Dr K. Ashok Kumar. "Personality prediction from social media text: An overview." Int. J. Eng. Res 9.05 (2020): 352-357.

[6] Sudha, G., et al. "Personality Prediction Through CV Analysis using Machine Learning Algorithms for Automated E-Recruitment Process." 2021 4th International Conference on Computing and Communications Technologies (ICCCT). IEEE, 2021.

[7] Al Maruf, Abdullah, et al. "A Survey on Personality Prediction." Proceedings of the 2nd International Conference on Computing Advancements. 2022.

[8] Abidin, Nur Haziqah Zainal, et al. "Improving Intelligent Personality Prediction using Myers-Briggs Type Indicator and Random Forest Classifier." International Journal of Advanced Computer Science and Applications 11.11 (2020).

[9] Ghosh, Soumi, and Chandan Banerjee. "A predictive analysis model of customer purchase behavior using modified random forest algorithm in cloud environment." 2020 IEEE 1st International conference for convergence in engineering (ICCE). IEEE, 2020.

[10] Karnakar, M., et al. "Applicant Personality Prediction System Using Machine Learning." 2021 2nd Global Conference for Advancement in Technology (GCAT). IEEE, 2021.

[11] William, P., and Abhishek Badholia. "Analysis of personality traits from text based answers using HEXACO model." 2021 international conference on innovative computing, intelligent communication and smart electrical systems (ICSES). IEEE, 2021.

[12] Sharma, Mridul, et al. "Data Mining Classification Techniques to Assign Individual Personality Type and Predict Job Profile." 2021 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO). IEEE, 2021.

[13] Katiyar, Sandhya, Himdweep Walia, and Sanjay Kumar. "Personality Classification System using Data Mining." 2020 8th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions)(ICRITO). IEEE, 2020.

[14] Wang, Shipeng, et al. "Personality traits prediction based on users' digital footprints in social networks via attention RNN." 2020 IEEE International Conference on Services Computing (SCC). IEEE, 2020.

[15] Hu, Peng, Jianping Zhang, and Ning Li. "Research on Flight Delay Prediction Based on Random Forest." 2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT). IEEE, 2021.

[16] Hänsch, Ronny. "Stacked Random Forests: More accurate and better calibrated." IGARSS 2020-2020 IEEE International Geoscience and Remote Sensing Symposium. IEEE, 2020.

[17] Zi, Ren, et al. "Stock price prediction based on optimized random forest model." 2022 Asia Conference on Algorithms, Computing and Machine Learning (CACML). IEEE, 2022.

[18] Farnaaz, Nabila, and M. A. Jabbar. "Random forest modeling for network intrusion detection system." Procedia Computer Science 89 (2016): 213-217.

[19] Noshad, Zainib, et al. "Fault detection in wireless sensor networks through the random forest classifier." Sensors 19.7 (2019): 1568.

[20] William, P., and Abhishek Badholia. "Evaluating Efficacy of Classification Algorithms on Personality Prediction Dataset." Ilkogretim Online 19.4 (2020): 3362-3375.

[21] Tandera, Tommy, et al. "Personality prediction system from facebook users." Procedia computer science 116 (2017): 604-611.

[22] Valanarasu, Mr R. "Comparative analysis for personality prediction by digital footprints in social media." Journal of Information Technology 3.02 (2021): 77-91.

[23] Lin, Weiwei, et al. "An ensemble random forest algorithm for insurance big data analysis." Ieee access 5 (2017): 16568-16575.

[24] Everingham, Yvette, et al. "Accurate prediction of sugarcane yield using a random forest algorithm." Agronomy for sustainable development 36.2 (2016): 1-9.

[25] Pang, Bo, et al. "Statistical downscaling of temperature with the random forest model." Advances in Meteorology 2017 (2017)

APPENDICES

PREPROCESSING:

```
#Importing the libraries

import pandas as pd

import sklearn as sk

import numpy as np

import nltk

nltk.download('stopwords')

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer

from nltk.tokenize.toktok import ToktokTokenizer

from bs4 import BeautifulSoup

import re,string,unicodedata

# In[31]:

#Read the Dataset

data = pd.read_csv("Dataset.csv")

# In[32]:

data.head()
```

```
# In[33]:

#Removing the html strips

def strip_html(text):

    soup = BeautifulSoup(text, "html.parser")

    return soup.get_text()

#Removing the square brackets and notations

def remove_between_square_brackets(text):

    return re.sub('[^A-Za-z0-9/. ]', '', text)

#Stemming the text

def simple_stemmer(text):

    ps=nltk.porter.PorterStemmer()

    text= ' '.join([ps.stem(word) for word in text.split()])

    return text

#set stopwords to english

stop=set(stopwords.words('english'))

print(stop)

#Tokenization of text

tokenizer=ToktokTokenizer()

#Setting English stopwords

stopword_list=nltk.corpus.stopwords.words('english')
```

```
#removing the stopwords
def remove_stopwords(text, is_lower_case=False):
    tokens = tokenizer.tokenize(text)
    tokens = [token.strip() for token in tokens]
    if is_lower_case:
        filtered_tokens = [token for token in tokens if token not in stopword_list]
    else:
        filtered_tokens = [token for token in tokens if token.lower() not in stopword_list]
    filtered_text = ' '.join(filtered_tokens)
    return filtered_text
#Apply function on review column
data['Resume']=data['Resume'].apply(strip_html)
data['Resume']=data['Resume'].apply(remove_between_square_brackets)
#data['Resume']=data['Resume'].apply(remove_notations)
data['Resume']=data['Resume'].apply(simple_stemmer)
data['Resume']=data['Resume'].apply(remove_stopwords)
print(data.Resume)
# In[34]:
#Save the data
data.to_csv("data_new.csv")
```

•

In[ ]:

TRAINING:

In[1]:

```
import gensim
```

In[2]:

```
import joblib
```

In[3]:

```
import pandas as pd
from sklearn.metrics.pairwise import cosine_similarity
```

In[4]:

```
import gensim.downloader as api
dataset = api.load("text8")
data = [i for i in dataset]
```

In[5]:

```
my_model = joblib.load('model_2.pkl')
```

In[6]:

```
sentences =["honesty", "emotionality", "extraversion", "agreeableness", "conscientiousness","openness to experience"]
```

In[7]:

```
data = pd.read_csv("MBTI_500.csv")
```

```
# In[9]:

x = data.iloc[0,1]

# In[10]:

vectors1 = [my_model.infer_vector([word for word in sent]).reshape(1,-1) for sent in sentences]

vectors2 = [my_model.infer_vector([word for word in x]).reshape(1,-1)]

# In[13]:

sim_values=[]

for i in range(len(data)):

    x=data.iloc[i,1]

    vectors2 = [my_model.infer_vector([word for word in x]).reshape(1,-1)]

    array=[]

    for j in range(0,6):

        similarity = cosine_similarity(vectors1[j],vectors2[0])

        array.append(similarity[0][0])

    array.append(data.iloc[i,1])

    sim_values.append(array)

df=pd.DataFrame(data=sim_values,columns=['h','e','x','a','c','o','type'])

df.to_csv("scores2.csv")

# In[ ]:
```

•

MODELS:

```
# In[11]:

import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn import svm

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification_report, confusion_matrix

from sklearn.model_selection import train_test_split, GridSearchCV

# In[2]:

data = pd.read_csv("scores2.csv")

# In[3]:

X = data[['h', 'e', 'x', 'a', 'c' , 'o']]

y = data['type']

# In[4]:

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=0)

# # RANDOM FOREST
```

```
# In[9]:

clf = RandomForestClassifier(n_estimators = 100, max_depth = 6)

# Training the model on the training dataset

# fit function is used to train the model using the training sets as parameters

clf.fit(X_train, y_train)

# performing predictions on the test dataset

y_pred = clf.predict(X_test)

# metrics are used to find accuracy or error

from sklearn import metrics

print()

 # using metrics module for accuracy calculation

print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))

# print classification report

print(classification_report(y_test, y_pred))

# # KNN

# In[10]:

model = KNeighborsClassifier(n_neighbors=2000)

# Train the model using the training sets

model.fit(X_train,y_train)

y_pred = model.predict(X_test)
```

```
# metrics are used to find accuracy or error
from sklearn import metrics
print()
# using metrics module for accuracy calculation
print("ACCURACY OF THE MODEL: ", metrics.accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred)
# # SVM
# In[12]:
from sklearn import svm
clf = svm.SVC(kernel= 'sigmoid')
clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
print(classification_report(y_test, y_pred))
# # HYPERPARAMETER TUNNING
# In[21]:
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(random_state = 42, max_depth=6)
from pprint import pprint
# Look at parameters used by our current forest
```

```
print('Parameters currently in use:\n')
pprint(rf.get_params())
# In[52]:
param_grid = {
    'n_estimators': [200,500],
    'max_features': ['auto', 'sqrt'],
    'max_depth' : [2,3,4,5,6],
    'criterion' :['gini', 'entropy']
}
CV_rfc = GridSearchCV(estimator=rf, param_grid=param_grid, cv= 5)
CV_rfc.fit(X_train, y_train)
# In[53]:
y_pred = CV_rfc.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
from sklearn.model_selection import train_test_split, GridSearchCV
# In[54]:
CV_rfc.best_params_
# In[25]:
rf = RandomForestClassifier(max_depth=5, max_features = 'auto', n_estimators = 200, criterion = 'entropy')
# In[26]:
```

```
rf.fit(X_train,y_train)

y_pred = rf.predict(X_test)

metrics.accuracy_score(y_test, y_pred)

print(classification_report(y_test, y_pred))

# In[ ]:
```