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

Group Members:

Rochak Shrivastav (20BCE1814)

Vansh Shan (20BCE1533)

Pranay Pratik (20BCE1751)

Full Design:

Predicting personality is a trending topic since many recruiters also have preferences for candidates with particular personality types. Our system would help them as well as other people to know the personality type according to the latest trending and most trusted MBTI personality type. We propose an intelligent system for predicting personality types based on the self-description of a person and their social media posts. For this, we combine two types of personality models namely the HEXACO model and the MBTI (Myers - Briggs Personality Type Indicator) type. We use various machine learning algorithms for prediction along with state-of-the-art models of NLP which help us in dealing with text data. We have used dummy datasets for training our models. For each input, our system generates the HEXACO scores solely based on the text entered by the user with the help of the training provided to the NLP models and the six scores are stored. These six numbers now act as input to a machine learning model which is a classification model. This classification model predicts the personality of the user according to the MBTI personality.

A. System Architecture

Our system has been separated into two different modules, each of which assists us in efficiently predicting the personality of a user.

As shown in figure 1 the system 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.

Now let us look at the working of the two different modules which will help our system to correctly identify the personality type of any user.

a) 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.

b) 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.

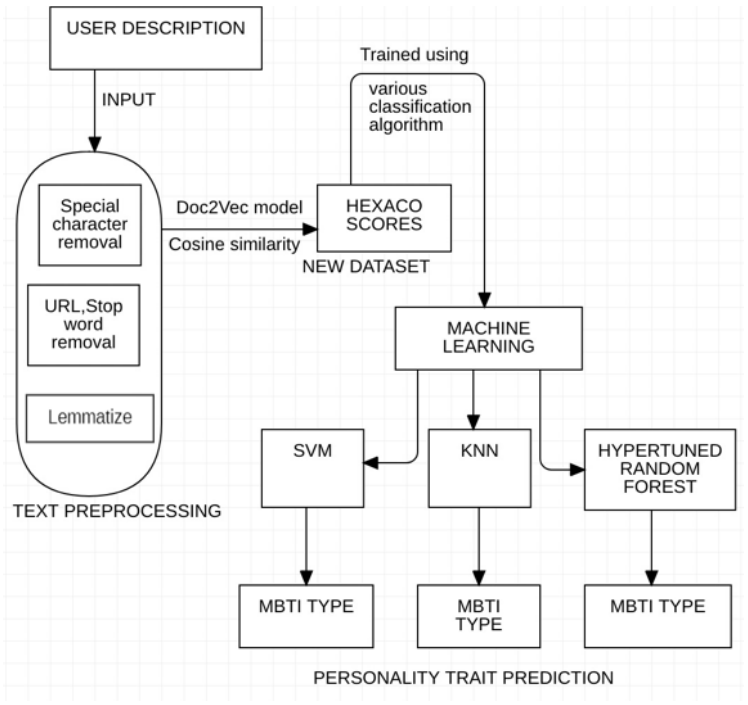


Figure 1: System Architecture

B. Background

a) Gensim library: The Gensim library is an open-source Python package that may be used in several fields, including Natural Language Processing (NLP), Topic Modelling, and Semantic Topic Extraction from Documents. In addition to scikit-learn, it is one of the numerous text processing software that can handle massive text corpora.[33] Using the unsupervised methods available in gensim, we can automate the process of finding the semantic structure of text data. We used the gensim package to develop a Doc2Vec model for personality prediction that is an integral feature of our approach.

b) Text8 Dataset: This dataset contains the first one billion Wikipedia characters.[34] In general, it was designed to make testing models easy and efficient. It was also developed to make output prediction efficient. This text8 dataset was used to create and train the Doc2vec model for our system, which will eventually aid in personality prediction from applicant resumes.

c) Doc2Vec Model: The Doc2Vec model is used to generate a representation of a set of words extracted from a document. It is one of the natural language processing (NLP) methods used to represent texts as vectors. This model provides a way for extracting word embeddings from corpus texts using unsupervised learning techniques. We have developed the Doc2vec model for our system using the Python gensim library.

d) Cosine Similarity: is a statistic that may be used to assess how similar data items are, regardless of size.[35] Cosine similarity treats each data item in a dataset as a vector. Equation 1 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’.

e) KNN Model: KNN is one of the supervised learning approaches that determines the probability that a data point will belong to a specific group, based on the categorization of all data points. This model is applicable for regression and classification. Here, we have attempted to utilise the KNN model as a classification model to categorise the designation matching to each candidate's résumé.[36] This model typically attempts to accurately categorise the testing data by determining the distance, often Euclidean distance or Manhattan distance, between the testing data and training data points, as seen in Equation 2.

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

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

Equation 2

where,

$$a = \text{coordinate point } (x_1, y_1)$$

$$b = \text{coordinate point } (x_2, y_2)$$

Next, K points that are most comparable to the testing data are selected. The KNN model then calculates which classes of the "K" training set the test set will belong to, and eventually selects the class with the greatest probability.

f) SVM Model: SVM (Support Vector Machines) is one of the most effective supervised machine learning models that can be used as both a regression and classification model. SVM facilitates classification by selecting a correct decision boundary line, known as the hyperplane, that maximises the separation between the data points seen for each class label.[37]

The kernels are a collection of mathematical operations that the SVM model employs. The basic function of a kernel is to accept data as input and transform it into the desired format. Various SVM methods use distinct kernel functions. RBF (radial basis function kernel) and polynomial kernel are the two most popular kernels among those chosen for this application. The kernel functions' return of the scalar product between two points in an extraordinarily adequate feature space.

Equation 3 shows the polynomial kernel formula:

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

Equation 3

Equation 4 shows the Gaussian radial basis formula:

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

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.

g) Random Forest Model: Random forest model is a supervised learning model that, similar to KNN and SVM, may be used as both a regression and classification model. It is a collection of trees constructed into a learning-based model. This classifier consists of a set of decision trees trained on a particular portion of the initial training data chosen at random. In order to effectively classify the test item, it mixes the judgments from numerous decision trees and casts the result.[6]

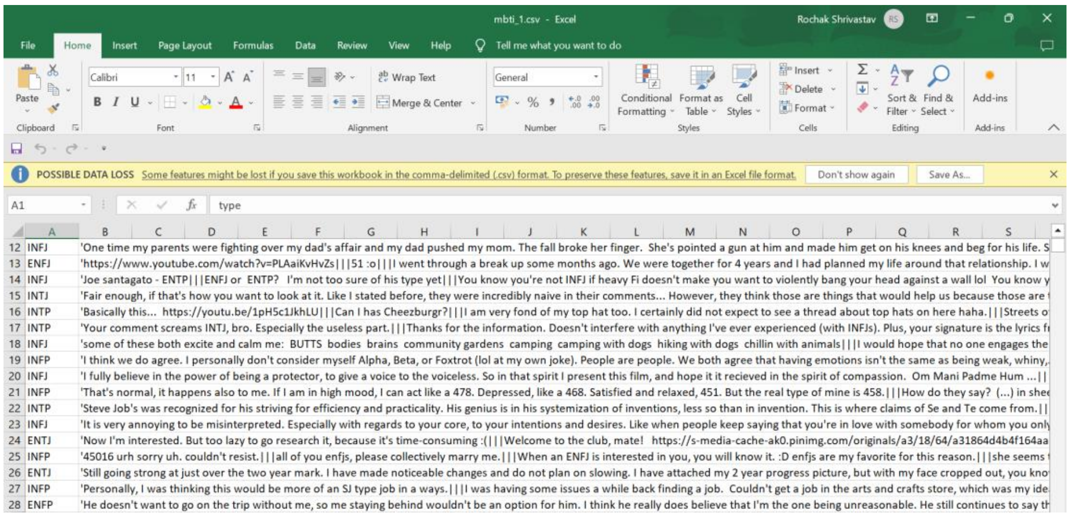
Random forest is really referred to as an ensemble algorithm since it mixes many algorithms, decision tree in this example. This model employs the bagging procedure, an ensemble meta approach that improves the accuracy of ML models. Consequently, the random forest model progressively removes or eliminates the flaws that may occur in a single decision tree classification model. This model aids in boosting classification accuracy and reducing dataset overfitting.

h) Hyperparameter tuning: This strategy is crucial when dealing with machine learning models since it determines the general behaviour of a machine learning model.[7] These are the model parameters that progressively summarise the main aspects of the model, and they are gradually adjusted to provide the optimal model output. GridSearchCV and RandomizedSearchCV are the two most efficient hyperparameter tuning strategies. Here for our system, we have done hyperparameter tweaking on the Random Forest model, which has steadily increased the Random Forest model's efficiency and precision. The model that uses machine learning is tested for a variety of hyperparameter parameters using the GridSearchCV method. This technique is called GridSearchCV because it scans a collection of hyperparameter parameters in order to find the best combination of hyperparameters. Suppose, for instance, that two distinct sets of values are desired to define the Logistic Regression Classification algorithm's hyperparameters C and Alpha. Using a variety of hyperparameter combinations, the grid search method will generate the optimal model.

GridSearchCV has limitations due to the fact that it only evaluates a limited number of hyperparameter values. RandomizedSearchCV circumvents these restrictions. To identify the optimal set of hyperparameters, it randomly traverses the grid. This method reduces inefficient calculation.

<u>Implementation:</u>

Initial Dataset:



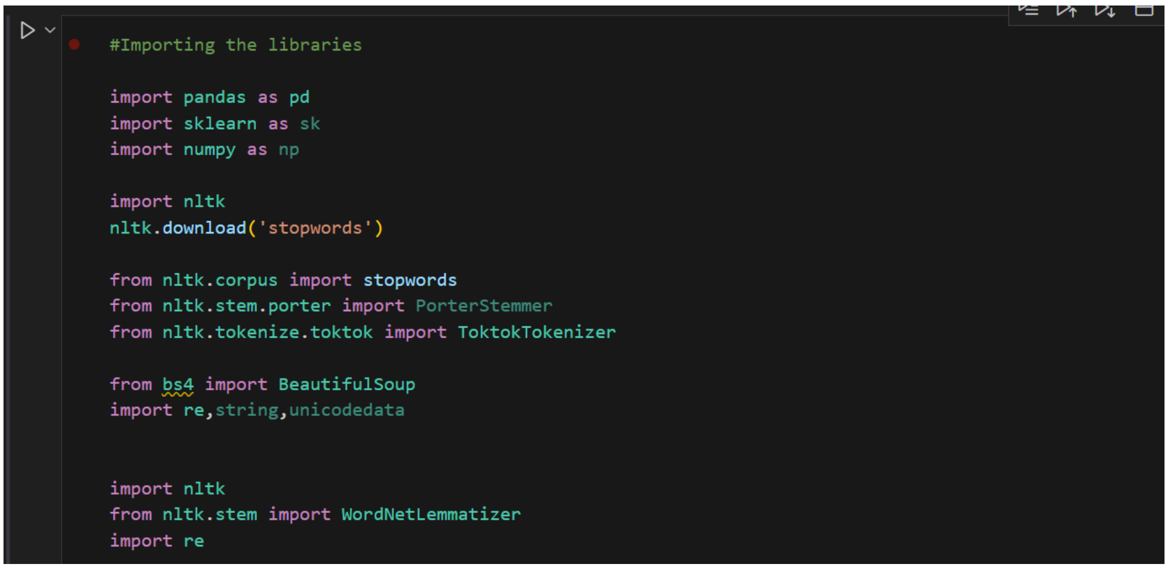
POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

A1 type

	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
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 fi
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 shee
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 only
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
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

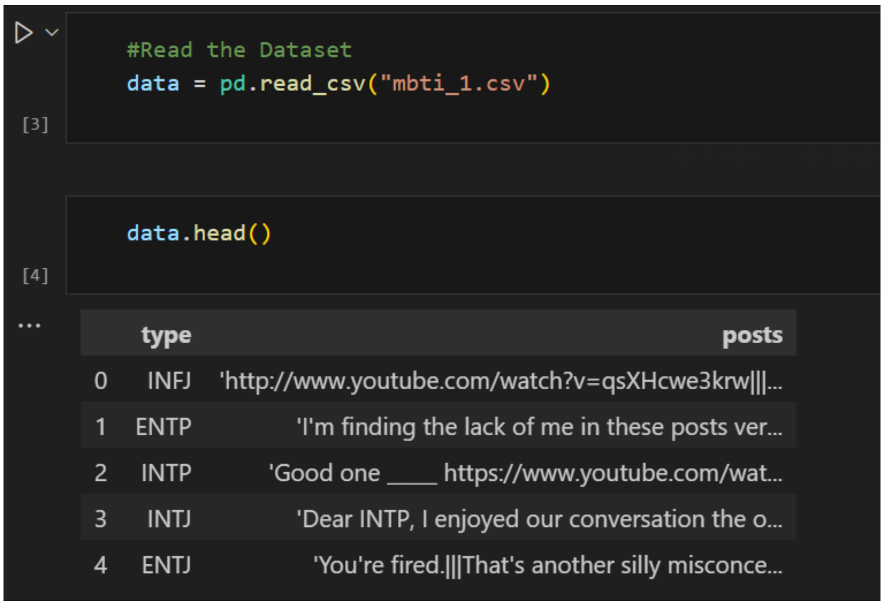
Preprocessing:

- 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
```

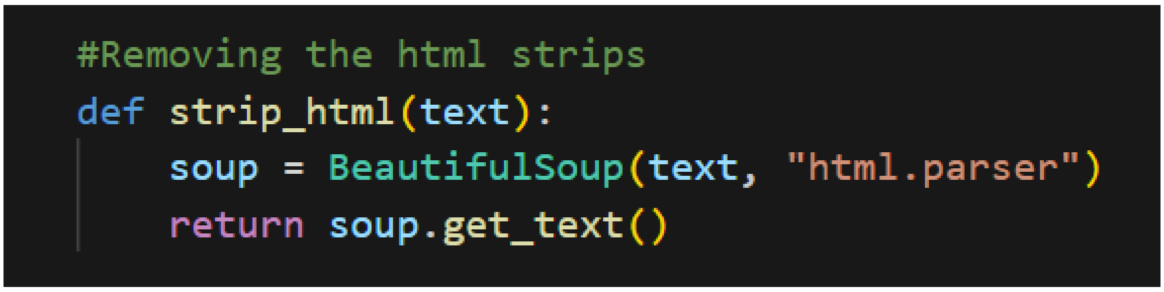
- Reading Dataset:



```
#Read the Dataset
data = pd.read_csv("mbti_1.csv")
[3]

data.head()
[4]
```

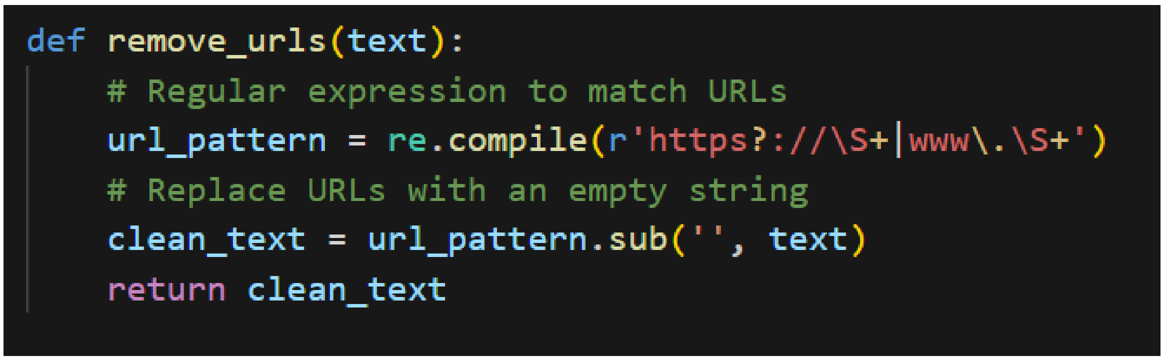
```
   type                                              posts
0  INFJ  'http://www.youtube.com/watch?v=qsXHcwe3krw|||...
1  ENTP        'I'm finding the lack of me in these posts ver...
2  INTP     'Good one _____ https://www.youtube.com/wat...
3  INTJ        'Dear INTP, I enjoyed our conversation the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

- Removing HTML Tags based content:



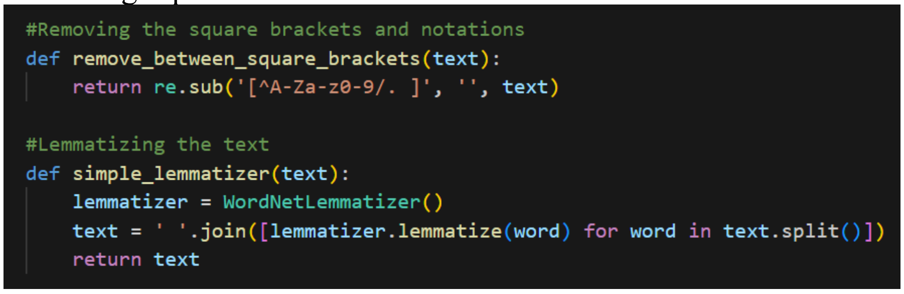
```
#Removing the html strips
def strip_html(text):
    soup = BeautifulSoup(text, "html.parser")
    return soup.get_text()
```

- Replacing URL's:



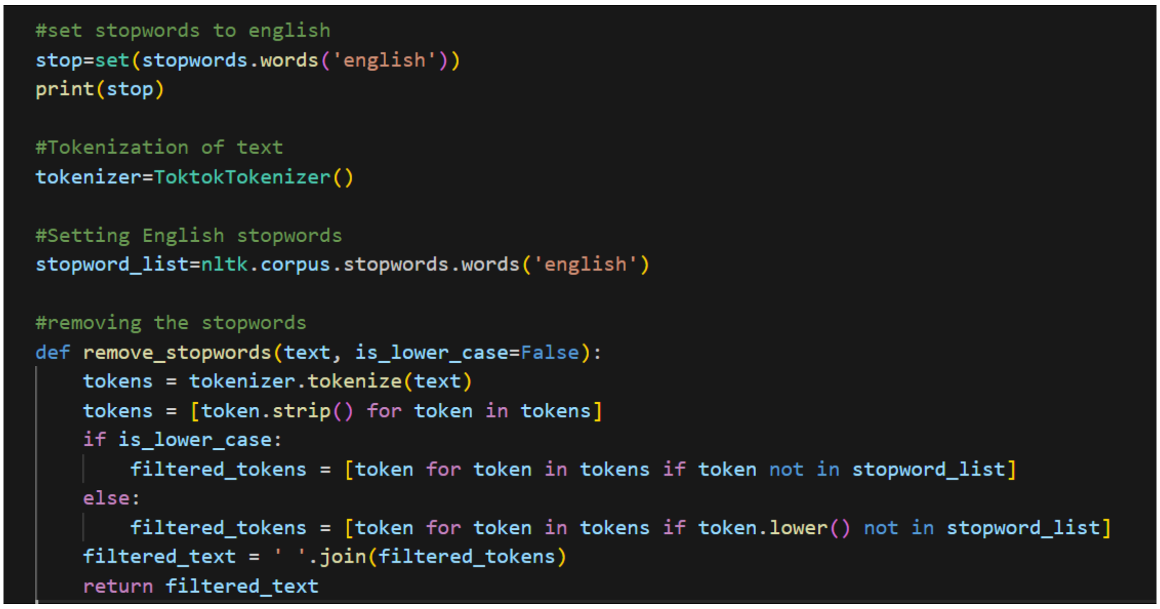
```
def remove_urls(text):
    # Regular expression to match URLs
    url_pattern = re.compile(r'https?://\S+|www\.\S+')
    # Replace URLs with an empty string
    clean_text = url_pattern.sub('', text)
    return clean_text
```

- Removing Square brackets and lemmatization:



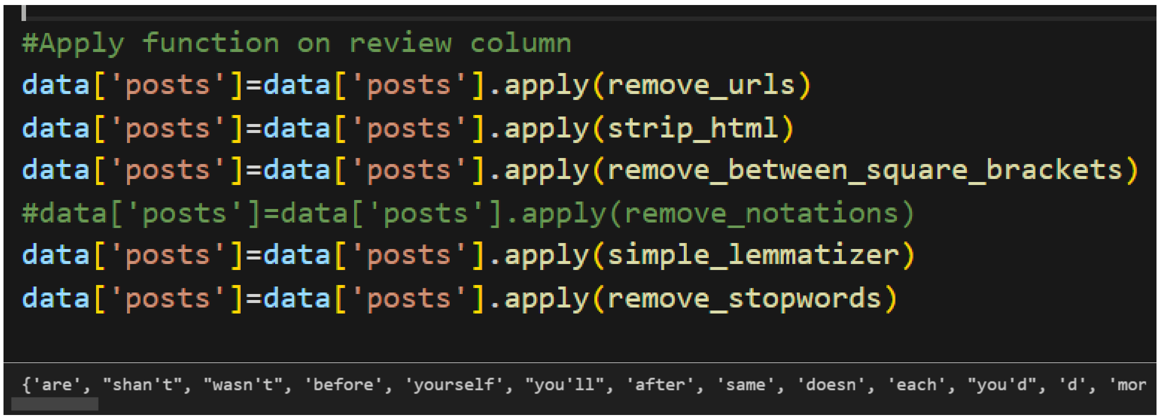
```
#Removing the square brackets and notations
def remove_between_square_brackets(text):
    return re.sub('[^A-Za-z0-9/. ]', '', text)

#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
```

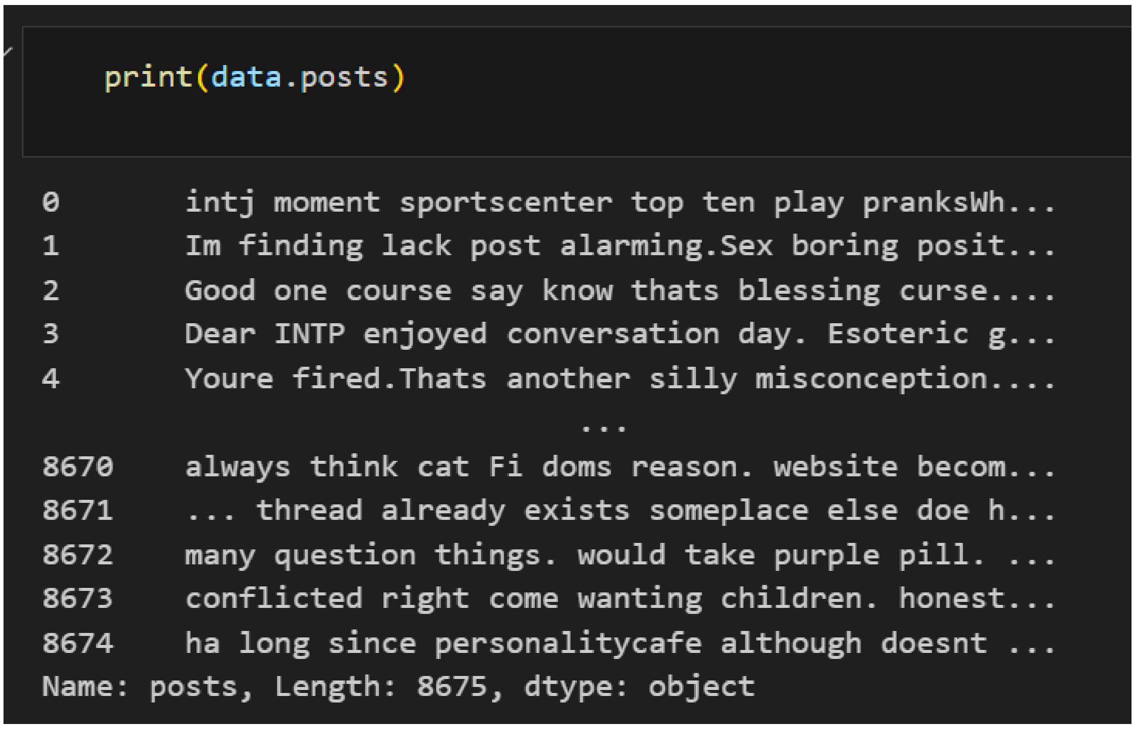
- Applying function to review on column:



```
#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
```

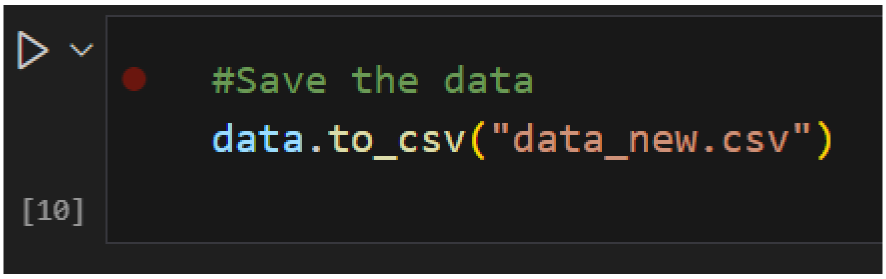
- Printing Data:



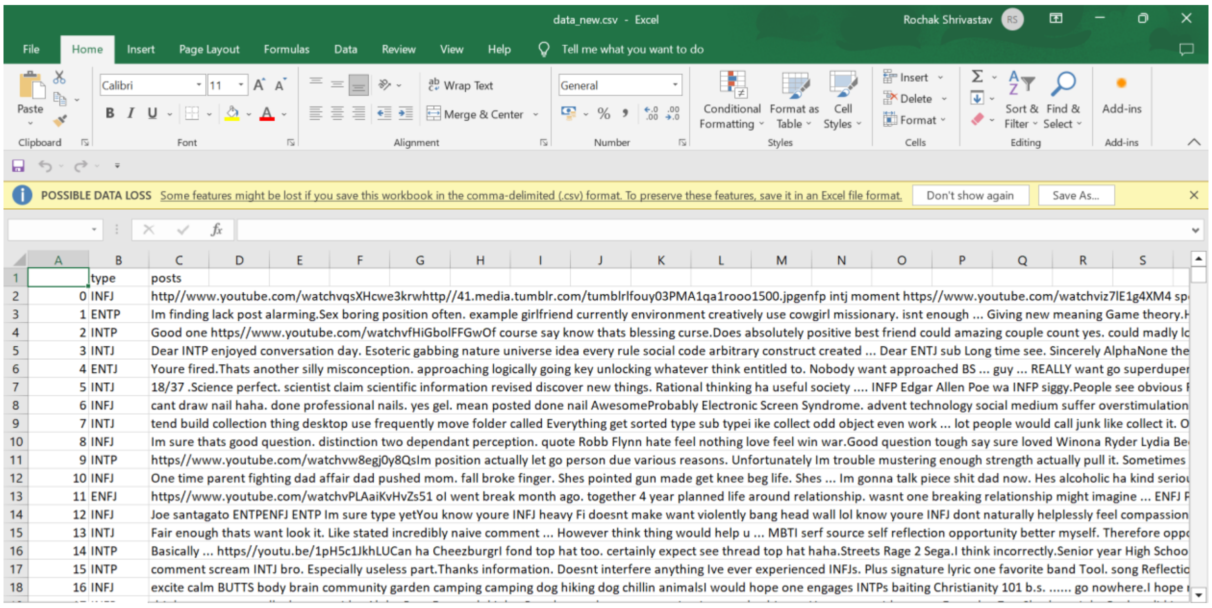
```
print(data.posts)
```

```
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")
```

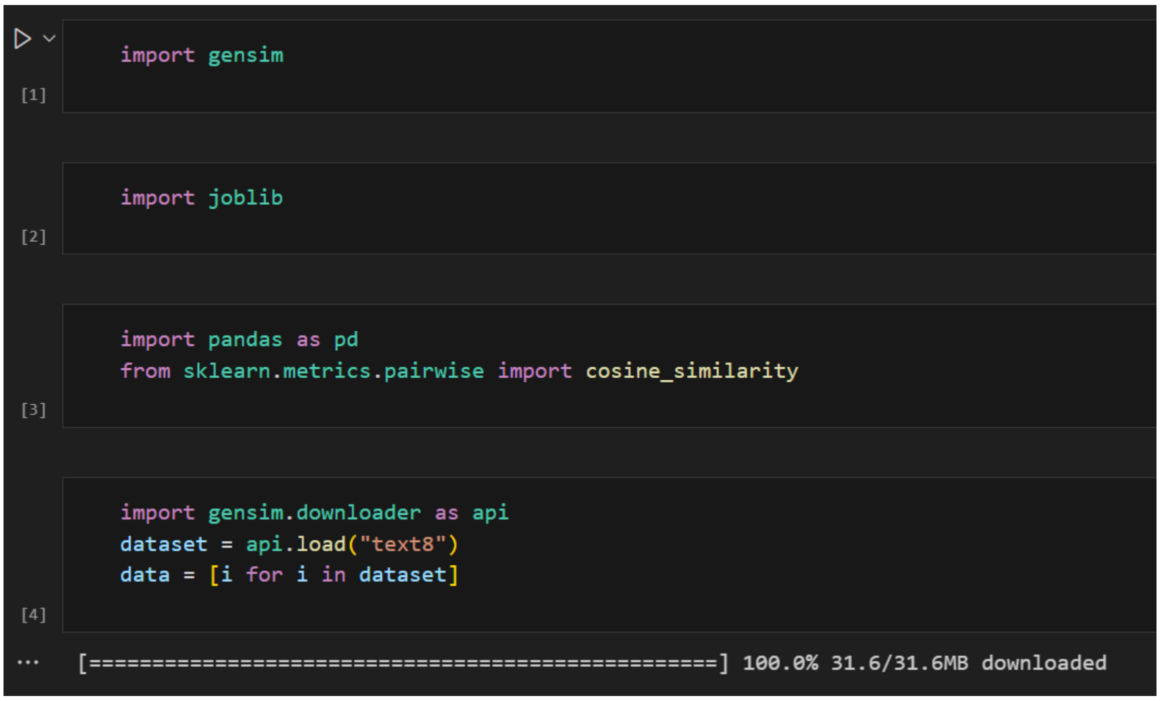


POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these features, save it in an Excel file format.

	A	B	C
1		type	posts
2	0	INFJ	http//www.youtube.com/watchvqsXHcwe3krwhttp//41.media.tumblr.com/tumblrlfouy03PMA1qa1rooo1500.jpgenfp intj moment https//www.youtube.com/watchviz7lE1g4XM4 sp
3	1	ENTP	Im finding lack post alarming.Sex boring position often. example girlfriend currently environment creatively use cowgirl missionary. isnt enough ... Giving new meaning Game theory.H
4	2	INTP	Good one https//www.youtube.com/watchvfHiGbolFFGwOf course say know thats blessing curse.Does absolutely positive best friend could amazing couple count yes. could madly lo
5	3	INTJ	Dear INTP enjoyed conversation day. Esoteric gabbing nature universe idea every rule social code arbitrary construct created ... Dear ENTJ sub Long time see. Sincerely AlphaNone the
6	4	ENTJ	Youre fired.Thats another silly misconception. approaching logically going key unlocking whatever think entitled to. Nobody want approached BS ... guy ... REALLY want go superduper
7	5	INTJ	18/37 .Science perfect. scientist claim scientific information revised discover new things. Rational thinking ha useful society .... INFP Edgar Allen Poe wa INFP siggy.People see obvious F
8	6	INFJ	cant draw nail haha. done professional nails. yes gel. mean posted done nail AwesomeProbably Electronic Screen Syndrome. advent technology social medium suffer overstimulation
9	7	INTJ	tend build collection thing desktop use frequently move folder called Everything get sorted type sub typei ike collect odd object even work ... lot people would call junk like collect it. O
10	8	INFJ	Im sure thats good question. distinction two dependant perception. quote Robb Flynn hate feel nothing love feel win war.Good question tough say sure loved Winona Ryder Lydia Be
11	9	INTP	https//www.youtube.com/watchvw8egj0y8QsIm position actually let go person due various reasons. Unfortunately Im trouble mustering enough strength actually pull it. Sometimes
12	10	INFJ	One time parent fighting dad affair dad pushed mom. fall broke finger. Shes pointed gun made get knee beg life. Shes ... Im gonna talk piece shit dad now. Hes alcoholic ha kind seriou
13	11	ENFJ	https//www.youtube.com/watchvPLAaiKvHvZs51 ol went break month ago. together 4 year planned life around relationship. wasnt one breaking relationship might imagine ... ENFJ F
14	12	INFJ	Joe santagato ENTPENFJ ENTP Im sure type yetYou know youre INFJ heavy Fi doesnt make want violently bang head wall lol know youre INFJ dont naturally helplessly feel compassion
15	13	INTJ	Fair enough thats want look it. Like stated incredibly naive comment ... However think thing would help u ... MBTI serf source self reflection opportunity better myself. Therefore oppo
16	14	INTP	Basically ... https//youtu.be/1pH5c1JkhLUCan ha CheezburgrI fond top hat too. certainly expect see thread top hat haha.Streets Rage 2 Sega.I think incorrectly.Senior year High Schoo
17	15	INTP	comment scream INTJ bro. Especially useless part.Thanks information. Doesnt interfere anything Ive ever experienced INFJs. Plus signature lyric one favorite band Tool. song Reflectio
18	16	INFJ	excite calm BUTTS body brain community garden camping camping dog hiking dog chillin animalsI would hope one engages INTPs baiting Christianity 101 b.s. ...... go nowhere.I hope

Training Model:

- Importing Libraries:



```
[1]
import gensim
```

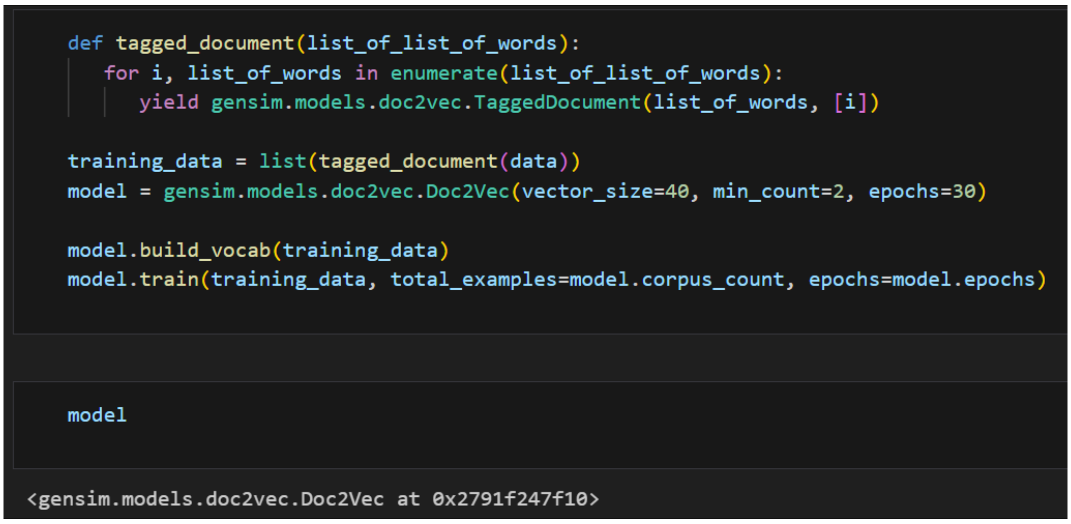
```
[2]
import joblib
```

```
[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:

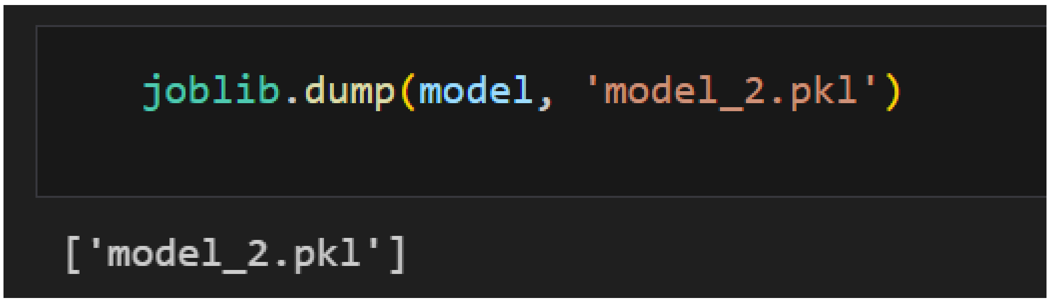


```
def tagged_document(list_of_list_of_words):
    for i, list_of_words in enumerate(list_of_list_of_words):
        yield gensim.models.doc2vec.TaggedDocument(list_of_words, [i])

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>
```

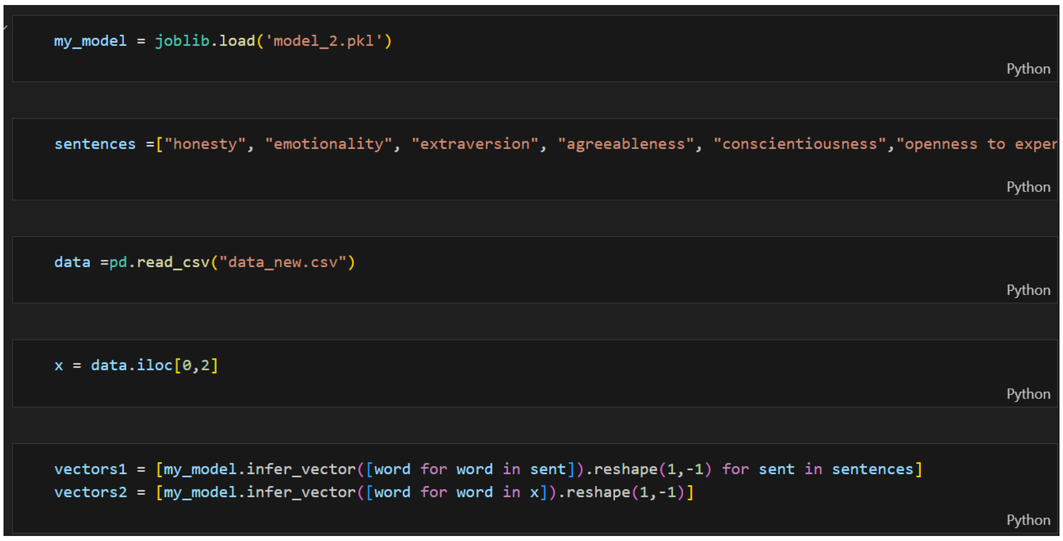
- Saving Model:



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

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

- Creating dataset of cosine similarity values based on HEXACO scores:



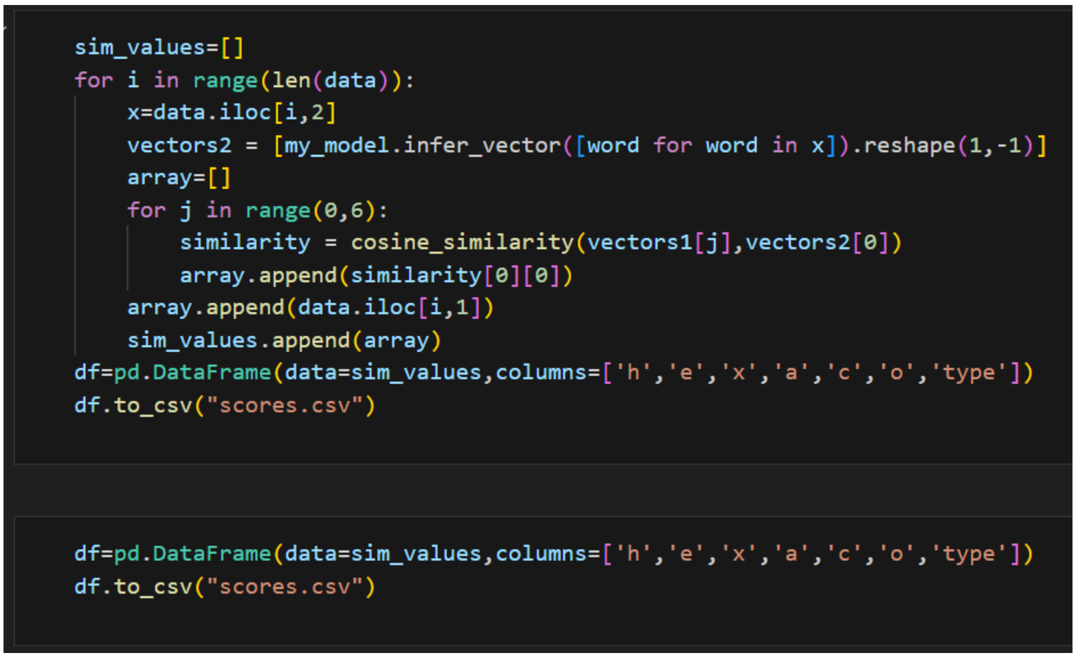
```
my_model = joblib.load('model_2.pkl')
```
Python

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

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

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

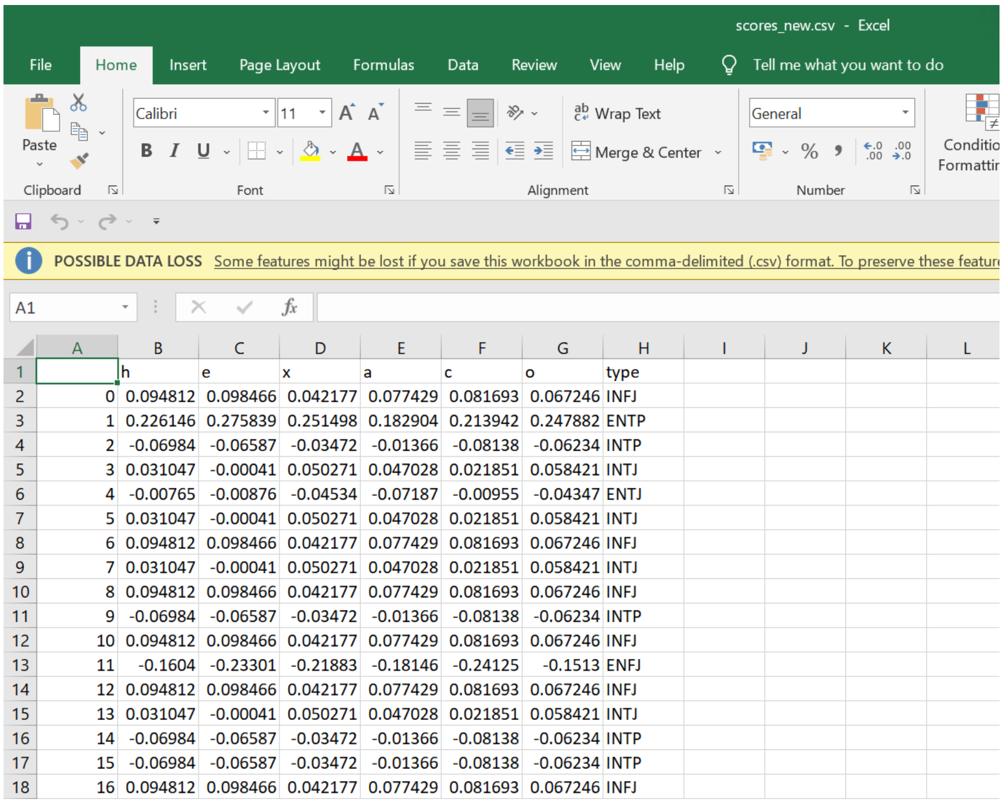
```
vectors1 = [my_model.infer_vector([word for word in sent]).reshape(1,-1) for sent in sentences]
vectors2 = [my_model.infer_vector([word for word in x]).reshape(1,-1)]
```
Python



```
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:

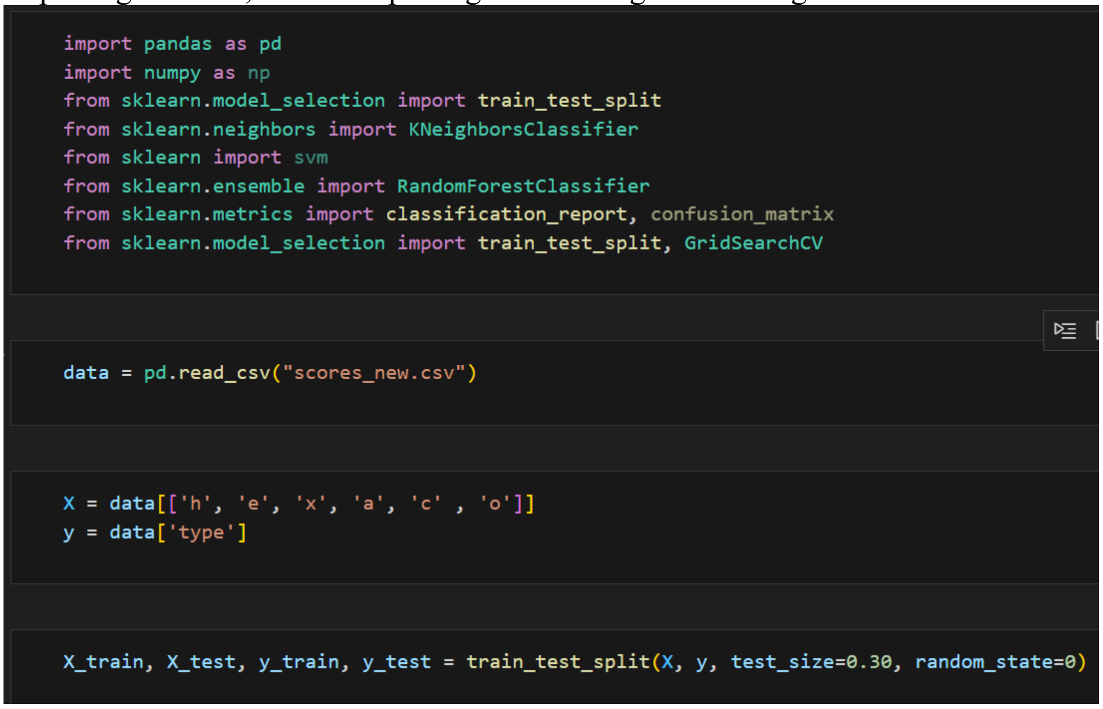


POSSIBLE DATA LOSS Some features might be lost if you save this workbook in the comma-delimited (.csv) format. To preserve these featur

	A	B	C	D	E	F	G	H
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:



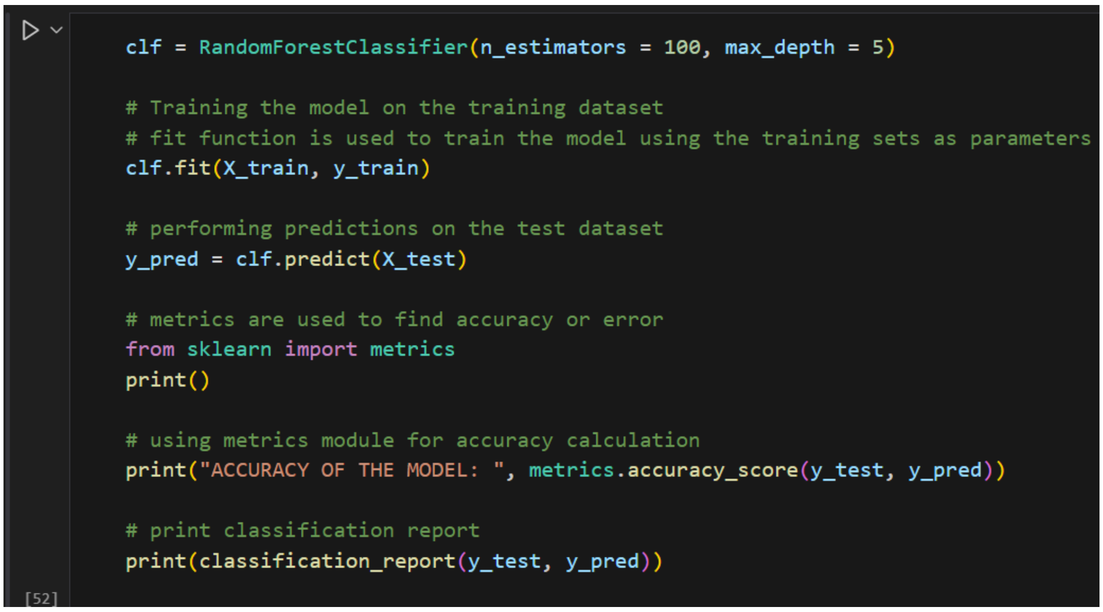
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn import svm
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection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

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

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

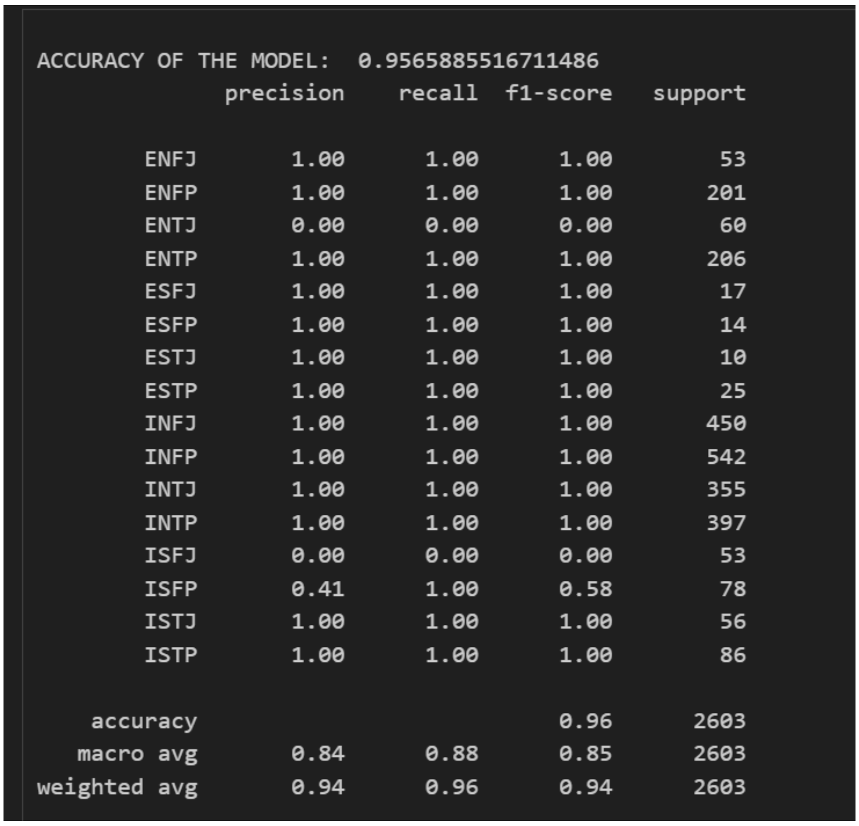
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30, random_state=0)
```

- Training Random forest model:



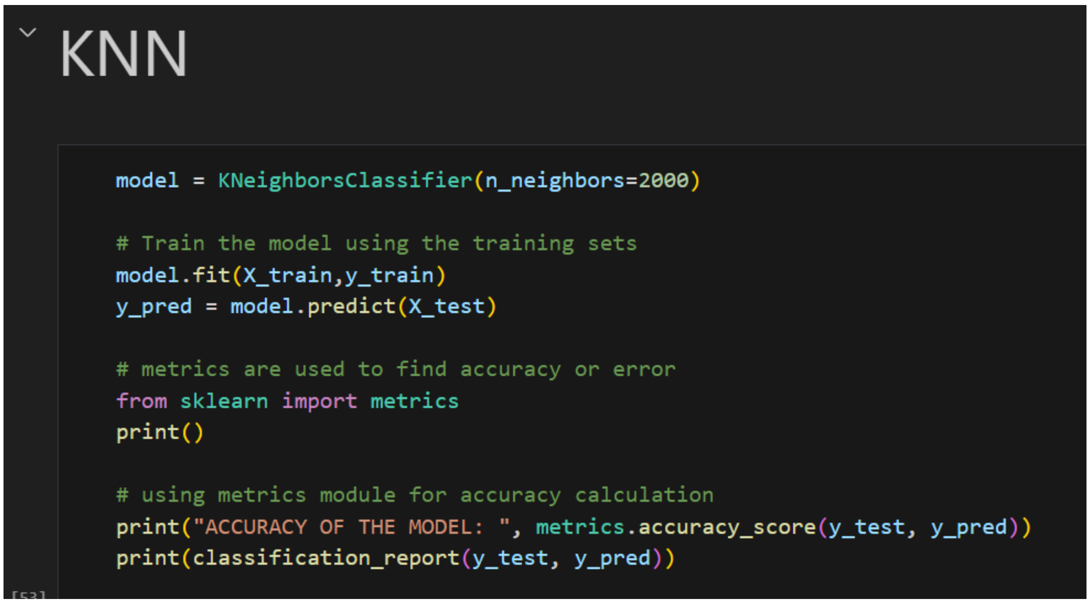
```
clf = RandomForestClassifier(n_estimators = 100, max_depth = 5)

# 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]



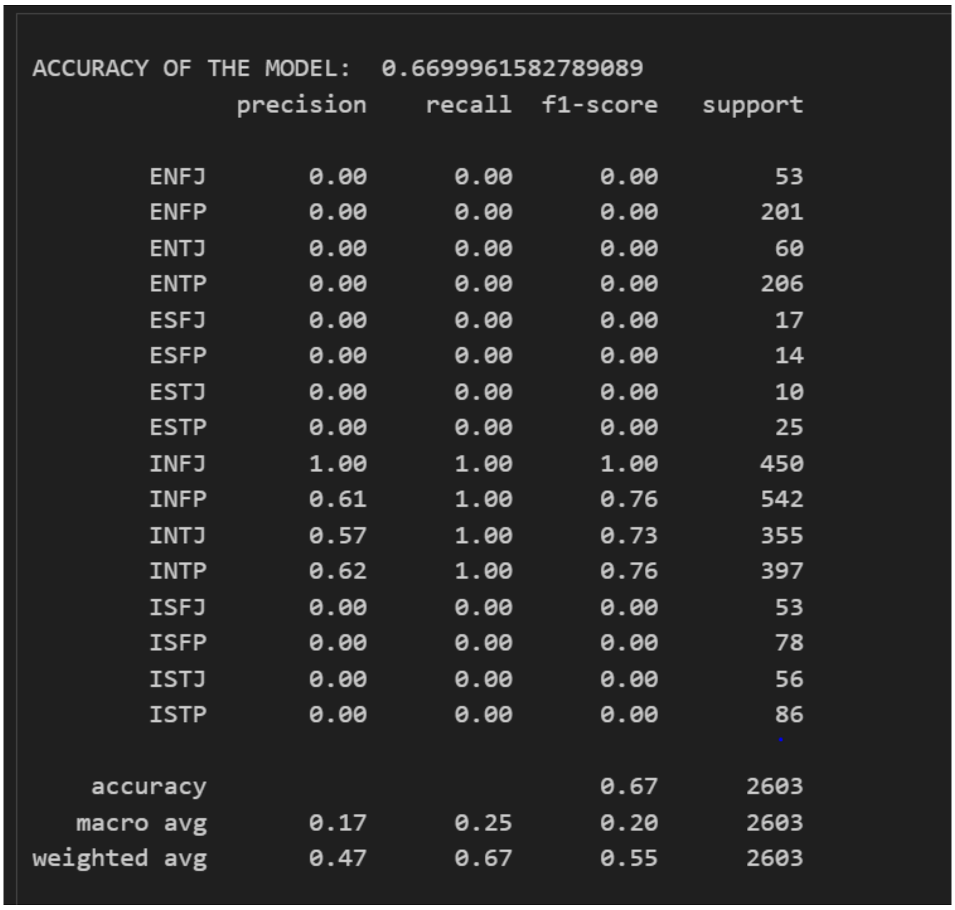
```
ACCURACY OF THE MODEL:  0.9565885516711486
              precision    recall  f1-score   support

        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:



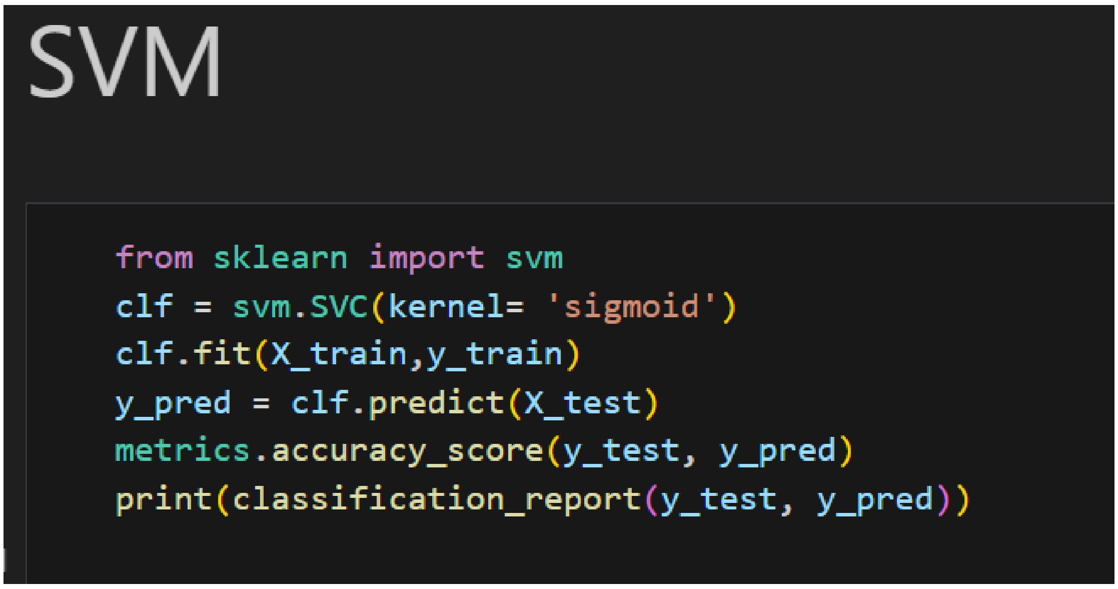
KNN

```python
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))
```



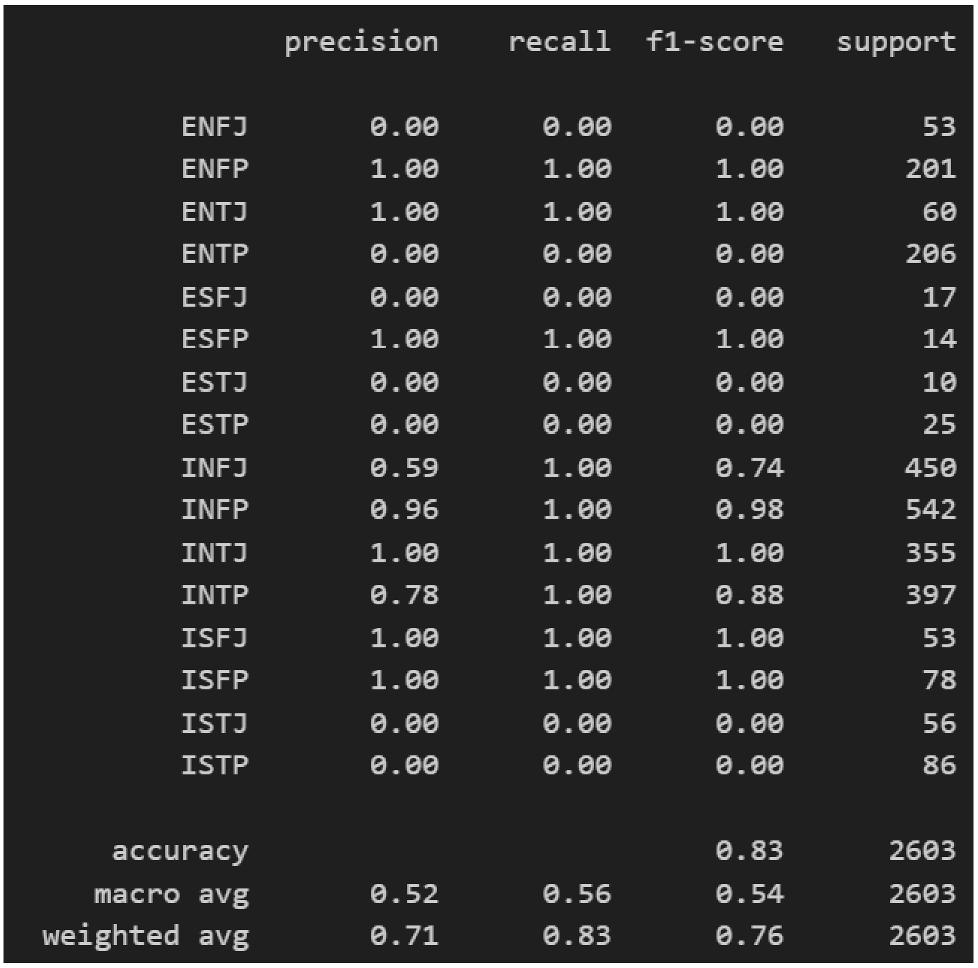
```
ACCURACY OF THE MODEL:  0.6699961582789089
              precision    recall  f1-score   support

        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:



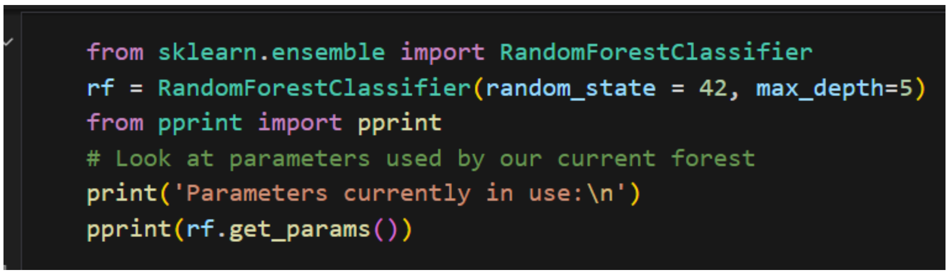
SVM

```
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))
```

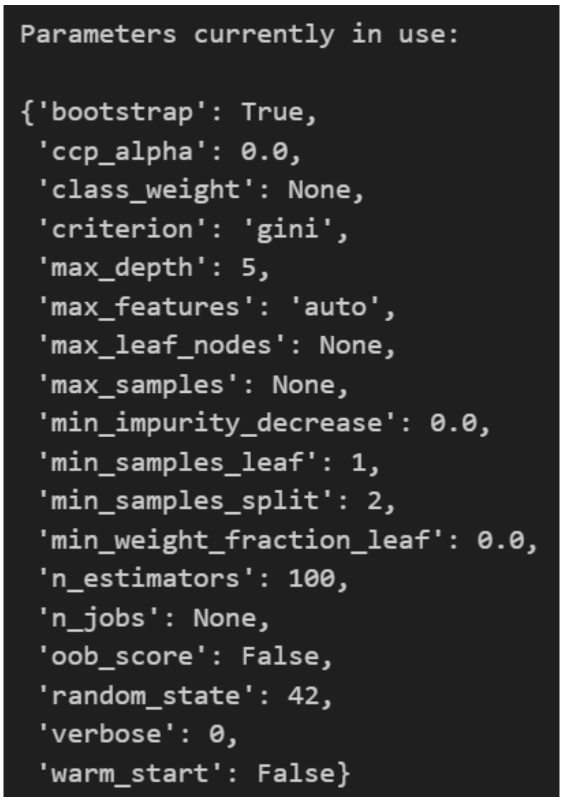


```
              precision    recall  f1-score   support

        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
```

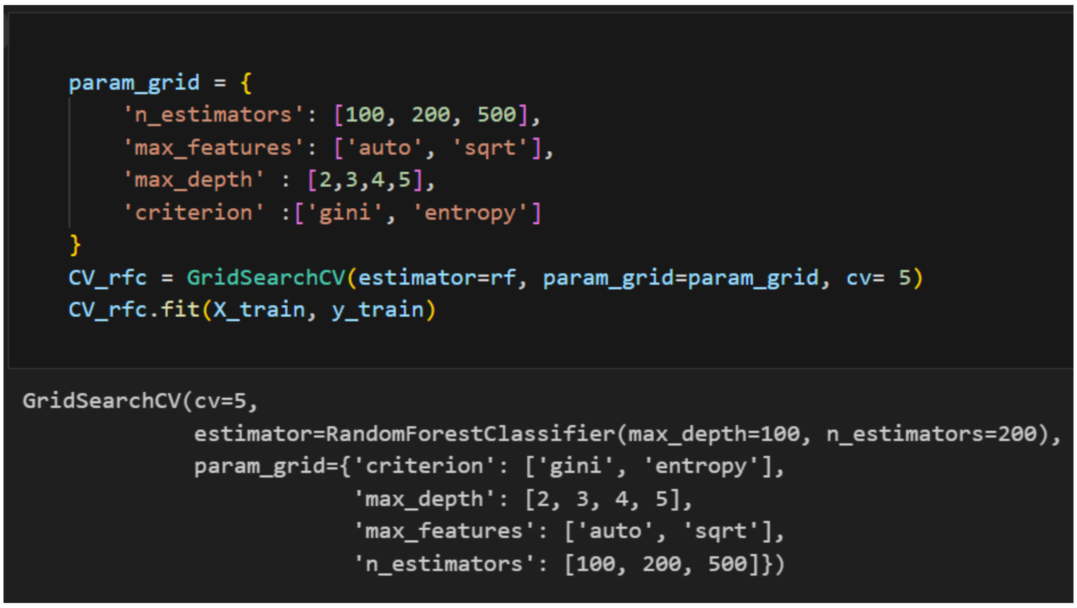
- Hyper-Parameter tuning:



```
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())
```

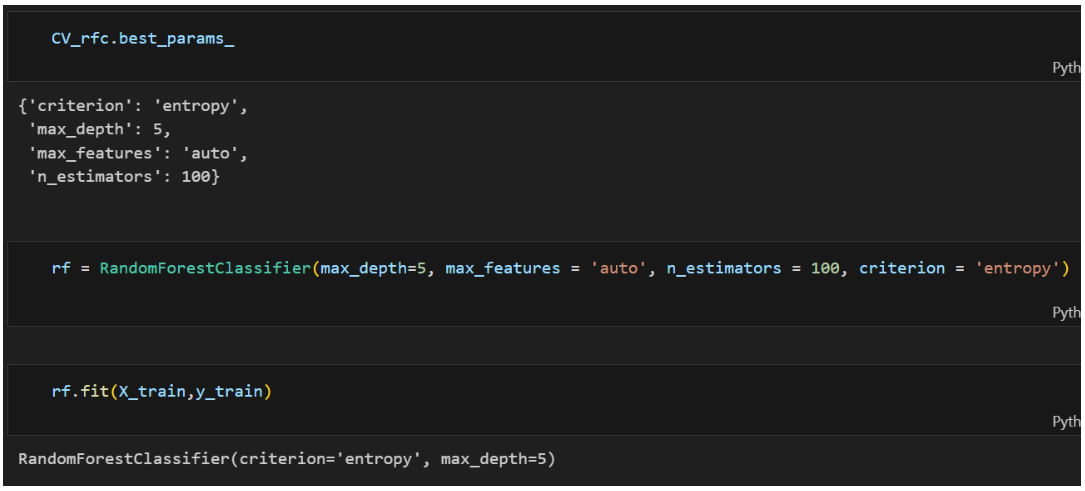


```
Parameters currently in use:

{'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_
```

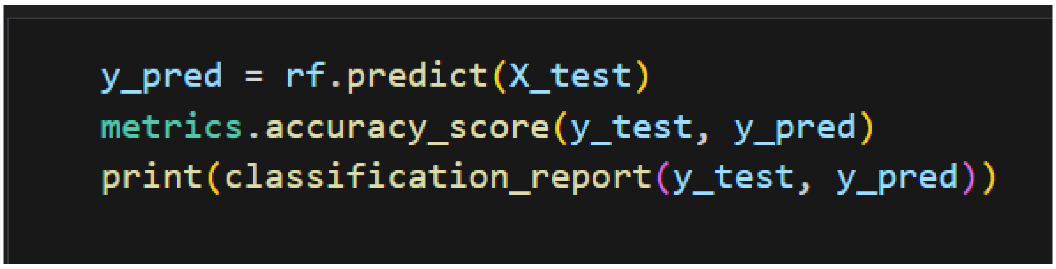
```
{'criterion': 'entropy',
 'max_depth': 5,
 'max_features': 'auto',
 'n_estimators': 100}
```

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

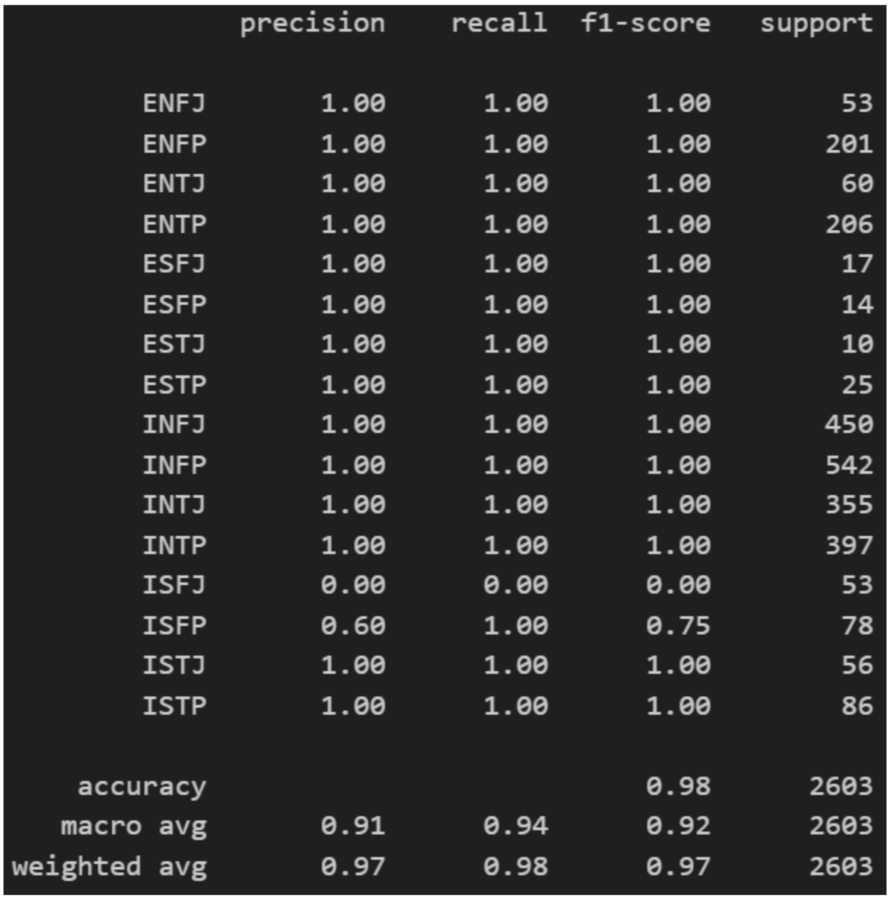
```
rf.fit(X_train,y_train)
```

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

Making Prediction for out text:

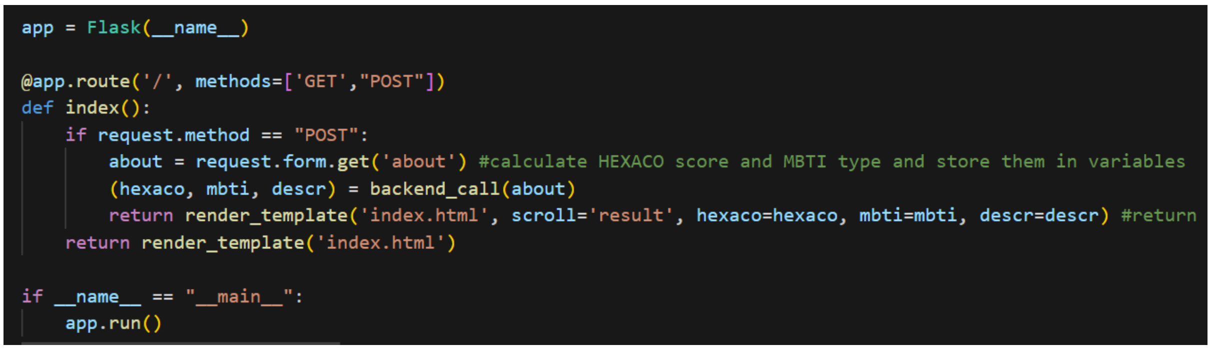


```
y_pred = rf.predict(X_test)
metrics.accuracy_score(y_test, y_pred)
print(classification_report(y_test, y_pred))
```

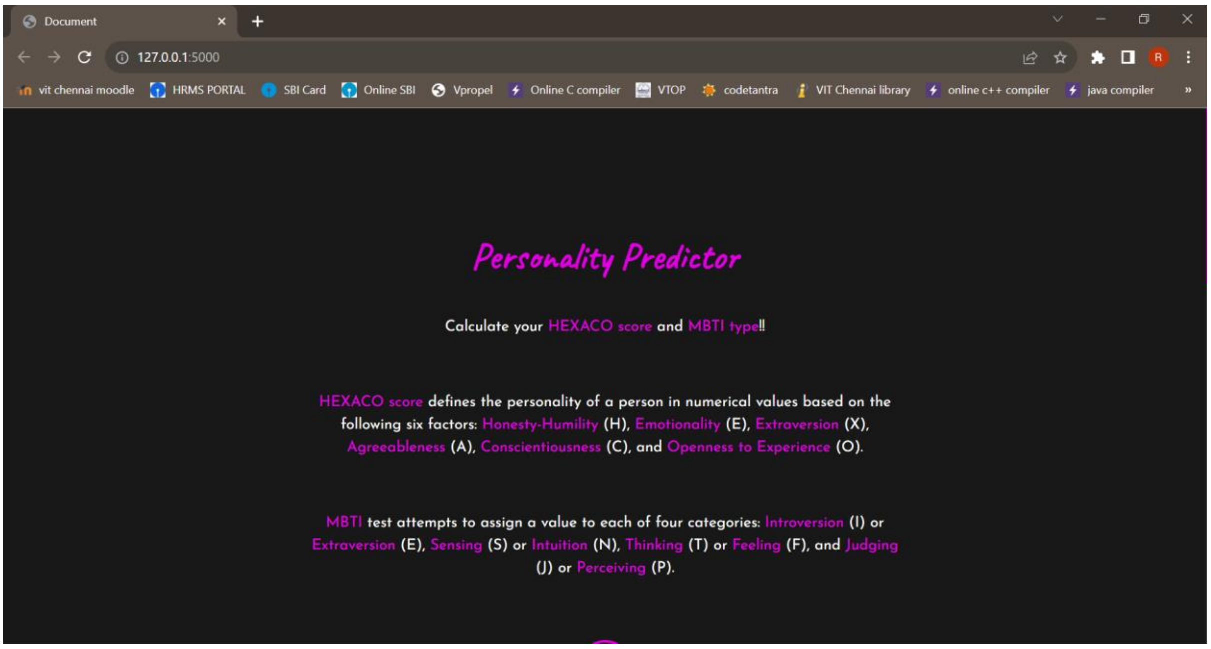


```
              precision    recall  f1-score   support

        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()
```

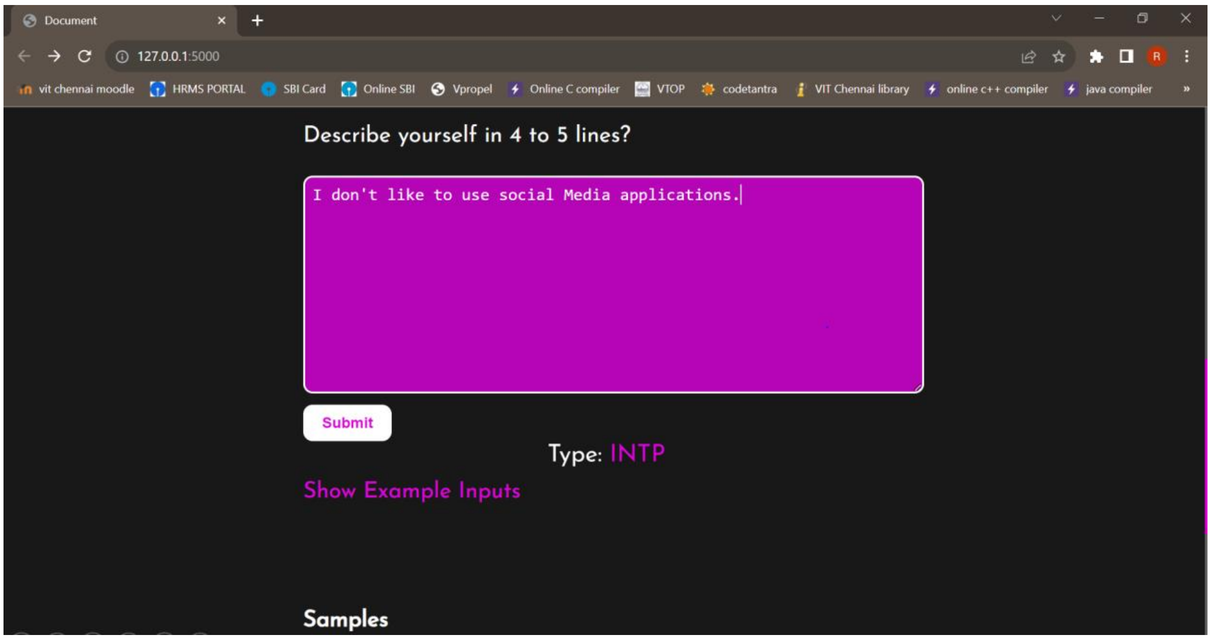


Personality Predictor

Calculate your HEXACO score and MBTI type!!

HEXACO score defines the personality of a person in numerical values based on the following six factors: Honesty-Humility (H), Emotionality (E), Extraversion (X), Agreeableness (A), Conscientiousness (C), and Openness to Experience (O).

MBTI test attempts to assign a value to each of four categories: Introversion (I) or Extraversion (E), Sensing (S) or Intuition (N), Thinking (T) or Feeling (F), and Judging (J) or Perceiving (P).



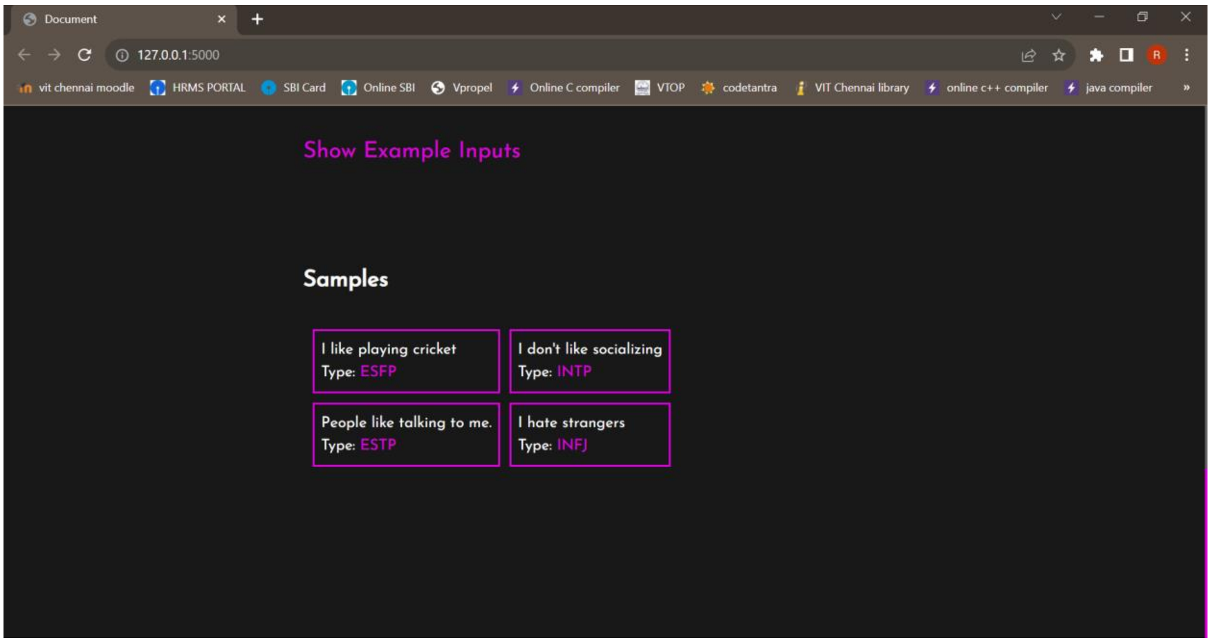
Describe yourself in 4 to 5 lines?

I don't like to use social Media applications.

Type: INTP

Show Example Inputs

Samples



Show Example Inputs

Samples

I like playing cricket Type: ESFP	I don't like socializing Type: INTP
People like talking to me. Type: ESTP	I hate strangers Type: INFJ

Performance Metrices:

- Graph 1:

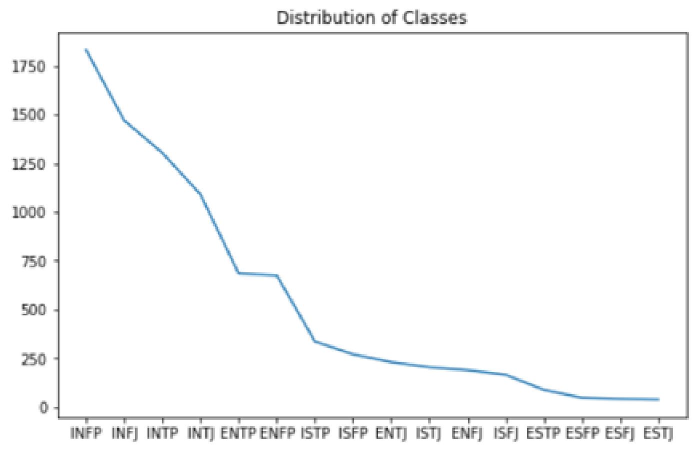


Figure 2: Line graph between the MBTI types and number of rows

Figure 2 shows that the data is highly imbalanced.

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:

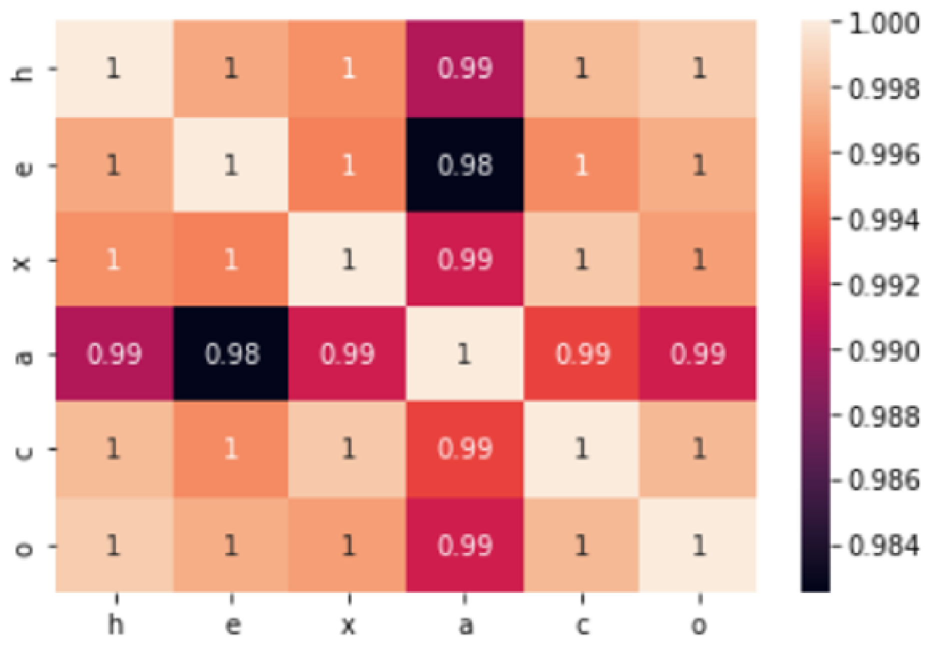


Figure 3: Correlation matrix between HEXACO traits

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

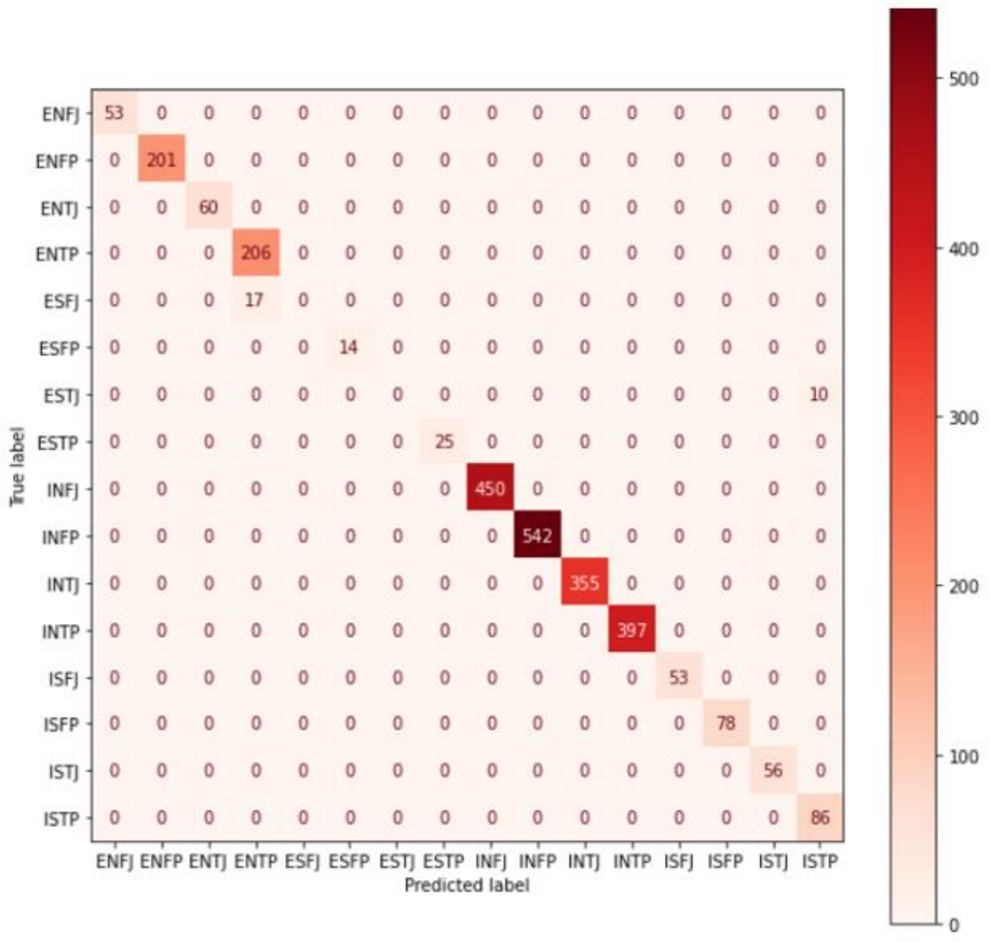
We tried applying four algorithms including KNN Classifier, SVM (with Sigmoid kernel), Random Forest and then hyperparameter tuned random forest algorithms. Table 1 shows the results of the experiments conducted.

- Table 1 :

Table 1: Comparison table for different classification models

Sl. No.	Algorithm	Precision	Recall	F-measure	Accuracy
1	KNN Classifier	0.88	0.93	0.91	0.93
2	SVM	0.71	0.83	0.76	0.83
3	Random Forest	0.94	0.96	0.94	0.96
4	Hyperparameter Tuned Random Forest	0.97	0.98	0.97	0.98

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



	ENFJ	ENFP	ENTJ	ENTP	ESFJ	ESFP	ESTJ	ESTP	INFJ	INFP	INTJ	INTP	ISFJ	ISFP	ISTJ	ISTP
ENFJ	53	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ENFP	0	201	0	0	0	0	0	0	0	0	0	0	0	0	0	0
ENTJ	0	0	60	0	0	0	0	0	0	0	0	0	0	0	0	0
ENTP	0	0	0	206	0	0	0	0	0	0	0	0	0	0	0	0
ESFJ	0	0	0	17	0	0	0	0	0	0	0	0	0	0	0	0
ESFP	0	0	0	0	0	14	0	0	0	0	0	0	0	0	0	0
ESTJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	10
ESTP	0	0	0	0	0	0	0	25	0	0	0	0	0	0	0	0
INFJ	0	0	0	0	0	0	0	0	450	0	0	0	0	0	0	0
INFP	0	0	0	0	0	0	0	0	0	542	0	0	0	0	0	0
INTJ	0	0	0	0	0	0	0	0	0	0	355	0	0	0	0	0
INTP	0	0	0	0	0	0	0	0	0	0	0	397	0	0	0	0
ISFJ	0	0	0	0	0	0	0	0	0	0	0	0	53	0	0	0
ISFP	0	0	0	0	0	0	0	0	0	0	0	0	0	78	0	0
ISTJ	0	0	0	0	0	0	0	0	0	0	0	0	0	0	56	0
ISTP	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	86

Figure 4: The dataset's confusion matrix

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

- Table 2 :

Table 2: Classification Report for Hypertuned Random Forest model

TYPE	PRECISION	RECALL	FI-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	0.92	1.00	0.96	206
ESFJ	1.00	1.00	1.00	17
ESFP	1.00	1.00	1.00	14
ESTJ	0.00	0.00	0.00	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	1.00	1.00	1.00	53
ISFP	1.00	1.00	1.00	78
ISTJ	1.00	1.00	1.00	56
ISTP	0.90	1.00	0.95	86

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 types. The weighted average is also calculated for the dataset. The accuracy for this training dataset is approximately 99%.

Result:

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.