A project report on

METIER – 7th Sem Project

Submitted in partial fulfillment of the requirements for the Degree of

B. Tech in Computer Science and Engineering

by

ABHISH KUMAR ANAND (1705005)

SHUBHAM KUMAR MAURYA (1705074)

SIDHARTH PUROHIT (1705078)

SURAJ KUMAR MISHRA (1805951)

under the guidance of

Prof. Ajay Anand

School of Computer Engineering
Kalinga Institute of Industrial Technology Deemed to be University
Bhubaneswar

April 2021

CERTIFICATE

This is to certify that the project report entitled **"METIER"** submitted by

ABHISH KUMAR ANAND	**1705005**
SHUBHAM KUMAR MAURYA	**1705074**
SIDHARTH PUROHIT	**1705078**
SURAJ KUMAR MISHRA	**1805951**

in partial fulfillment of the requirements for the award of the **Degree of Bachelor of Technology** in Discipline **of Engineering** is a Bonafede record of the work carried out under my(our) guidance and supervision at School of Computer Engineering, Kalinga Institute of Industrial Technology, deemed to be University.

Prof. Ajay Anand
Academic affiliation
Organization

ACKNOWLEDGEMENTS

We are profoundly grateful to Prof. AJAY ANAND for his expert guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion.

ABHISH KUMAR ANAND

SHUBHAM KUMAR MAURYA

SIDHARTH PUROHIT

SURAJ KUMAR MISHRA

ABSTRACT

Our focus for this project is using machine learning to build a classifier capable of sorting people into their Myers-Briggs Type Index (MBTI) personality type based on text samples from their social media posts. The motivations for building such a classifier are twofold. First, the pervasiveness of social media means that such a classifier would have ample data on which to run personality assessments, allowing more people to gain access to their MBTI personality type, and perhaps far more reliably and more quickly. There is significant interest in this area within the academic realm of psychology as well as the private sector. For example, many students wish to know more about their personalities and the career path(s) that suits them. Thus, our classifier could serve as a verification system for career counseling as a means of allowing people to have more confidence in their choice(s) of stream (or métier). Indeed, a text-based classifier would be able to operate on a far larger amount of data than that given in a single personality test.

TABLE OF CONTENTS

ABSTRACT	3
TABLE OF CONTENTS	4
CHAPTER 1: INTRODUCTION	5-7
CHAPTER 2: LITERATURE SURVEY	8-9
CHAPTER 3: PROJECT ANALYSIS / PROJECT IMPLEMENTATION	10-15
3.1 INTRODUCTION TO MACHINE LEARNING	10-11
3.2 NATURAL LANGUAGE PROCESSING (NLP)	12-13
3.3 MYERS-BRIGGS TYPE INDICATOR AND NATURAL LANGUAGE PROCESSING	14-15
CHAPTER 4: IMPLEMENTATION AND OBSERVATION	16-27
CHAPTER 5: WEB TECHNOLOGIES USED	28-29
CHAPTER 6: WEB APP IMPLEMENTATION	30-35
CHAPTER 7: CONCLUSION AND FUTURE WORK	36-37
INDIVIDUAL CONTRIBUTION	38
REFERENCES	39

CHAPTER 1

INTRODUCTION

The Myers Briggs Type Indicator (or MBTI for short) is a personality type system that divides everyone into 16 distinct personality types across 4 axes:

(I) Introversion – (E) Extroversion

(N) Intuition – (S) Sensing

(T) Thinking – (F) Feeling

(J) Judging – (P) Perceiving

Favorite world: Do you prefer to focus on the outer world or on your own inner world? This is called Extraversion (E) or Introversion (I).

Information: Do you prefer to focus on the basic information you take in or do you prefer to interpret and add meaning? This is called Sensing (S) or Intuition (N).

Decisions: When making decisions, do you prefer to first look at logic and consistency or first look at the people and special circumstances? This is called Thinking (T) or Feeling (F).

Structure: In dealing with the outside world, do you prefer to get things decided or do you prefer to stay open to new information and options? This is called Judging (J) or Perceiving (P).

Your Personality Type: When you decide on your preference in each category, you have your own personality type, which can be expressed as a code with four letters.

- **INTJ-A / INTJ-T**

  Imaginative and strategic thinkers, with a plan for everything.

- **INTP-A / INTP-T**

  Innovative inventors with an unquenchable thirst for knowledge.

- **ENTJ-A / ENTJ-T**

  Bold, imaginative and strong-willed leaders, always finding a way – or making one.

- **ENTP-A / ENTP-T**

  Smart and curious thinkers who cannot resist an intellectual challenge.

- **INFJ-A / INFJ-T**

  Quiet and mystical, yet very inspiring and tireless idealists.

- **INFP-A / INFP-T**

  Poetic, kind and altruistic people, always eager to help a good cause.

- **ENFJ-A / ENFJ-T**

  Charismatic and inspiring leaders, able to mesmerize their listeners.

- **ENFP-A / ENFP-T**

  Enthusiastic, creative and sociable free spirits, who can always find a reason to smile.

- **ISTJ-A / ISTJ-T**

  Practical and fact-minded individuals, whose reliability cannot be doubted.

- **ISFJ-A / ISFJ-T**

  Very dedicated and warm protectors, always ready to defend their loved ones.

- **ESTJ-A / ESTJ-T**

  Excellent administrators, unsurpassed at managing things – or people.

- **ESFJ-A / ESFJ-T**

  Extraordinarily caring, social and popular people, always eager to help.

- **ISTP-A / ISTP-T**

  Bold and practical experimenters, masters of all kinds of tools.

- **ISFP-A / ISFP-T**

  Flexible and charming artists, always ready to explore and experience something new.

- **ESTP-A / ESTP-T**

  Smart, energetic and very perceptive people, who truly enjoy living on the edge.

- **ESFP-A / ESFP-T**

  Spontaneous, energetic and enthusiastic people – life is never boring around them.

Overall, personality classification has many complexities because there are countless factors involved. Furthermore, even a human being may not be able to accurately classify a personality given a text. However, we hypothesize that using pre-trained language models might allow us to pick up on various subtleties in how different personality types use language.

CHAPTER 2

LITERATURE SURVEY

There is significant growing interest in automated personality prediction using social media among researchers in both the Natural Language Processing and Social Science fields [1]. So far, the application of traditional personality tests has mostly been limited to clinical psychology, counselling and human resource management by various organizations to personalize their job analysis needs. However, automated personality prediction from social media has a wider application, such as social media marketing or dating applications and websites to study and predict the behavior of users.

Most studies on personality prediction have focused on the Big Five or MBTI personality models, which are the two most used personality models in the world. According to Soto [2] "A personality trait is a characteristic pattern of thinking, feeling, or behaving that tends to be consistent over time and across relevant situations". The Big Five personality traits are openness, conscientiousness, extraversion, agreeableness, and neuroticism. These five factors are assumed to represent the basic structure behind all personality traits. They were defined and described by several different researchers during multiple periods of research.

Research on personality type prediction from textual data is very scarce and rare. However, important steps and have been taken in this field and through machine learning. Classic machine learning techniques and neural networks have been used successfully for predicting MBTI personality types.

One of the earliest studies on personality prediction using machine learning techniques was by Golbeck et al. [3]. They could accurately predict a user's personality type based on MBTI personality type indicator and by considering the information presented on their Twitter. This paper attempts to bridge the gap between social media and personality research by using the information people reveal in their online profiles. It poses a question and asks whether social media profiles can predict personality traits. If so, then there is an opportunity to integrate the many results on the implications of personality factors and behavior into the users' online experiences and to use social media profiles as a source of information to better understand individuals. For Example, the friend suggestion system could be tailored to a user based on whether they are more introverted or extraverted.

In another study, Komisin and Guinn [4] used the Naïve Bayes and Support Vector Machine (SVM) techniques to predict an individual's personality type based on their word choice. Their database was built based on in-class writing samples that were taken from 40 graduate students along with their MBTI personality type. They compared the performance of these two techniques and discovered that the Naïve Bayes technique performs better than SVM on their small dataset. In this paper they hypothesize that word choice in personal essays is not independent of Myers-Briggs personality type. They tried to explore this hypothesis in two ways: (1) by examining the specific word choices used in corpus of essays of subjects whose MBTI scores are known; and (2) by examining the semantic categories of these words.

Wan et al. [5] used a machine learning method to predict the Big Five personality type of users through their texts in Weibo, a Chinese social network, and they were able to successfully predict the personality type of the users.

Li, Wan and Wang [6] used the grey prediction model, the multiple regression model and the multi- tasking model to predict the user personality type based on the Big Five model and their text samples. They compared the performance of these three models and found that the grey prediction model performs better than the two other models.

In another study, Tandera et al. [7] used the Big Five personality model and some deep learning architecture to predict a person’s personality based on the user’s information on their Facebook. They compared the performance of their method with other previous studies that used classical machine learning methods and the results showed that their model successfully outperformed the accuracy of previous similar studies. Studies in the field of psychology showed that there is a correlation between personality and the linguistic behavior of a person. This correlation can be effectively analyzed and illustrated using natural language processing approach. Therefore, the goal of their research is to build a prediction system that can automatically predict user personality based on their activities in Facebook.

Furthermore, in another study, Hernandez and Knight [8] used various types of recurrent neural networks (RNNs) such as

simple RNN, gated recurrent unit (GRU) which is gating mechanism in recurrent neural networks,

long short-term memory (LSTM) which is an artificial recurrent neural network architecture used in the field of deep learning, and Bidirectional LSTM to build a classifier capable of predicting people’s MBTI personality type based on text samples from their social media posts. The Myers–Briggs Personality Type Dataset from Kaggle was used in their research. They experimented with various types of RNN like Simple RNN, GRU, LSTM. They compared the results and found that LSTM gave the best results.

CHAPTER 3

PROJECT ANALYSIS / PROJECT IMPLEMENTATION

3.1 INTRODUCTION TO MACHINE LEARNING

What is Machine Learning?

Two definitions of Machine Learning are offered. Arthur Samuel described it as: "the field of study that gives computers the ability to learn without being explicitly programmed." This is an older, informal definition.
Tom Mitchell provides a more modern definition: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."
Example: playing checkers.

E = the experience of playing process many confession statements
T = the task of processing statements
P = the probability that the program will give the right output.

In general, any machine learning problem can be assigned to one of two broad classifications: Supervised learning and Unsupervised learning.

Supervised Learning

In supervised learning, we are given a data set and already know what our correct output should look like, having the idea that there is a relationship between the input and the output. Supervised learning problems are categorized into "regression" and "classification" problems. In a regression problem, we are trying to predict results within a continuous output, meaning that we are trying to map input variables to some continuous function. In a classification problem, we are instead trying to predict results in a discrete output. In other words, we are trying to map input variables into discrete categories.

The problem we're going to discuss is also a supervised learning problem where we give our model a label to learn. It's basically a binary classification problem.

Unsupervised Learning

In unsupervised learning, the model is trained on unlabeled data i.e., data which is not given any labels. Unsupervised learning focuses on generating patterns in the data and remembering them or learning from them. One of the most common approach in unsupervised learning is “clustering” or grouping of data through means of some kind of analysis. New Data is then assigned to one of those newly found groups.

Reinforcement Learning

In this type of Learning, every good prediction done by the model includes a reward and every wrong prediction includes a penalty. In this way a feedback mechanism is incorporated with the learning process.

3.2 NATURAL LANGUAGE PROCESSING (NLP)

WHAT IS NATURAL LANGUAGE PROCESSING?

Natural Language Processing, or NLP for short, is broadly defined as the automatic manipulation of natural language, like speech and text, by software. The study of natural language processing has been around for more than 50 years and grew out of the field of linguistics with the rise of computers.
Natural language refers to the way we, humans, communicate with each other, namely, speech and text. We are surrounded by text. Think about how much text you see each day:

3.2.1 Confessions
3.2.2 Symbols
3.2.3 Literature texts
3.2.4 Signs
3.2.5 Menus
3.2.6 Email
3.2.7 SMS
3.2.8 Web Pages
3.2.9 and so much more…

Now think about speech. We may speak to each other, as a species, more than we write. It may even be easier to learn to speak than to write, voice and text are how we communicate with each other.

Given the importance of this type of data, we must have methods to understand and reason about natural language, just like we do for other types of data.

What is Linguistics?

Linguistics is the scientific study of language, including its grammar, semantics, and phonetics. Classical linguistics involved devising and evaluating rules of language. Great progress was made on formal methods for syntax and semantics, but for the most part, the interesting problems in natural language understanding resist clean mathematical formalism. Broadly, a linguist is anyone who studies language, but perhaps more colloquially, a self-defining linguist may be more focused on being out in the field.

Mathematics is the tool of science. Mathematicians working on natural language may refer to their study as mathematical linguistics, focusing exclusively on the use of discrete mathematical formalism and theory for natural language (e.g. formal languages and automata theory).

Computational Linguistics

Computational Linguistics is the modern study of linguistics using the tools of computer science. Yesterday’s linguistics may be today’s computational linguist as the use of computational tools and thinking has overtaken most fields of study. It is the study of computer systems for understanding and generating natural language

One natural function for computational linguistics would be the testing of grammars proposed by theoretical linguists, or human sentiment analysis, Twitter’s data being very famously explored with the same. We have used this application of Computer Science in the field of Linguistics to generate a machine which predicts the human emotions among the two binary classes, that is happy or sad.

> Large data and fast computers mean that new and different things can be discovered from large datasets of text by writing and running software.
>
> In the 1990s, statistical methods and statistical machine learning began to and eventually replaced the classical top-down rule-based approaches to language, primarily because of their better results, speed, and robustness. The statistical approach to studying natural language now dominates the field; it may define the field.
>
> Data-Drive methods and recommendation engines based on natural language processing have now become so popular that they must be considered mainstream approaches to computational linguistics. A strong contributing factor to this development is undoubtedly the increase amount of available electronically stored data to which these methods can be applied; another factor might be a certain disenchantment with approaches relying exclusively on hand-crafted rules, due to their observed brittleness.

3.3 Myers-Briggs Type Indicatorr and Natural Language Processing

The purpose of the Myers-Briggs Type Indicator personality inventory is to make the theory of psychological types described by C. G. Jung understandable and useful in people's lives. The essence of the theory is that much seemingly random variation in the behaviour is actually quite orderly and consistent, being due to basic differences in the way’s individuals prefer to use their perception and judgment. “Perception involves all the ways of becoming aware of things, people, happenings, or ideas. Judgment involves all the ways of coming to conclusions about what has been perceived. If people differ systematically in what they perceive and in how they reach conclusions, then it is only reasonable for them to differ correspondingly in their interests, reactions, values, motivations, and skills. “In developing the Myers-Briggs Type Indicator [instrument], the aim of Isabel Briggs Myers, and her mother, Katharine Briggs, was to make the insights of type theory accessible to individuals and groups.

They addressed the two related goals in the developments and application of the MBTI instrument:

The identification of basic preferences of each of the four dichotomies specified or implicit in Jung's theory. The identification and description of the 16 distinctive personality types that result from the interactions among the preferences.

Favourite world: Do you prefer to focus on the outer world or on your own inner world? This is called Extroversion(E) or Introversion(I).

Information: Do you prefer to focus on the basic information you take in or do you prefer to interpret and add meaning? This is called Sensing(S) or Intuition(I).

Decisions: When making decisions, do you prefer to first look at logic and consistency or first look at the people and special circumstances? This is called Thinking(T) or Feeling(F).

Structure: In dealing with the outside world, do you prefer to get things decided or do you prefer to stay open to new information and options? This is called Judging(J) or Perceiving(P).

Your Personality Type: When you decide on your preference in each category, you have your own personality type, which can be expressed as a code with four letters.

Tweets from various accounts on Twitter (all with an accompanying personality type) were gathered together; each account has a personality type, along with the last 50 tweets for each user.

At first, my goal was to tokenize the data (turn into individual words or parts) remove ALL the noise (stop words, hyperlinks, hashtags, etc.), and use the tokenized inputs try and predict the personality type of the user.

However, we realized that we were leaving a lot to be desired with respect to exploratory analysis. Instead of simply just getting word and type frequency, we realized we could be looking for so much more (hashtag/mention/retweet/URL frequency, n-gram occurrences, etc.). Thus, we have revamped the project to include a lot more on the exploratory analysis side.

We also have learned a lot about data clean-up, manipulation, storing data in different ways, and more that we hope becomes clear in my code.

We utilized three different Machine Learning models known for success in Natural Language Processing (NLP):

- Multinomial Naive Bayes
- Linear Support Vector Machine
- Neural Network

The project is split up into different sections that you may look at individually:

- Data Extraction/Cleanup
- Exploratory Analysis
- Naive Bayes
- Linear Support Vector Machine
- Neural Network

CHAPTER 4

IMPLEMENTATION & OBSERVATION

Model Part

Data Analysis
First of all, we will analyse the data and see what can be infrared from it

```
dataset = pd.read_csv('../input/mbti-type/mbti_1.csv')
dataset.head()
dataset.describe()
```

	type	posts
count	8675	8675
unique	16	8675
top	INFP	'oh boy- started this thread a while back, jus...
freq	1832	1

Here we can see that the data consist of 8675 values and the top is the most common type of value (INFP) with a frequency of 1832 times.

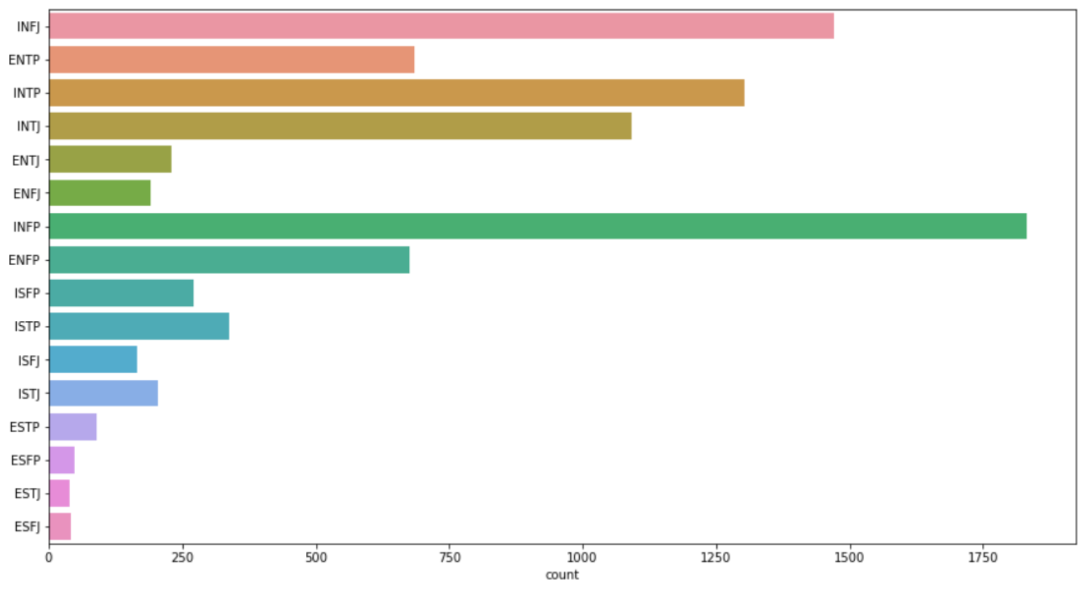
Now we see how many NULL values or empty cells are there in there in the data. As we can see there is no empty cells and no need to bother about filling those.

```
dataset.isnull().sum()
```

```
type     0
posts    0
dtype: int64
```

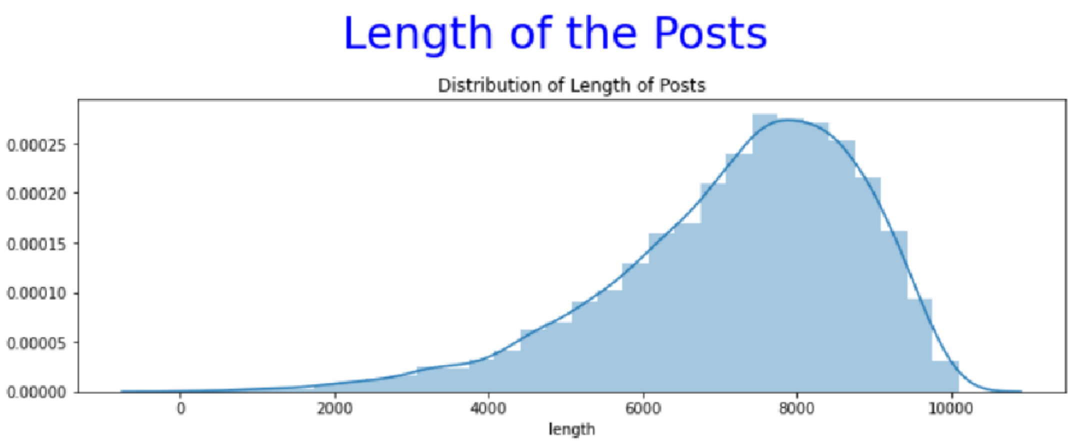
No of records in each category

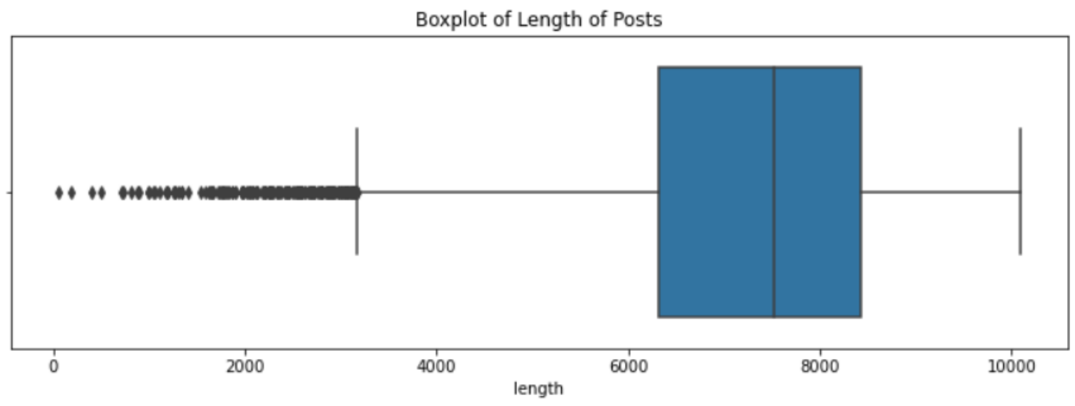
```
plt.figure(figsize = (15,8))
sns.countplot(y = 'type' , data = dataset)
```



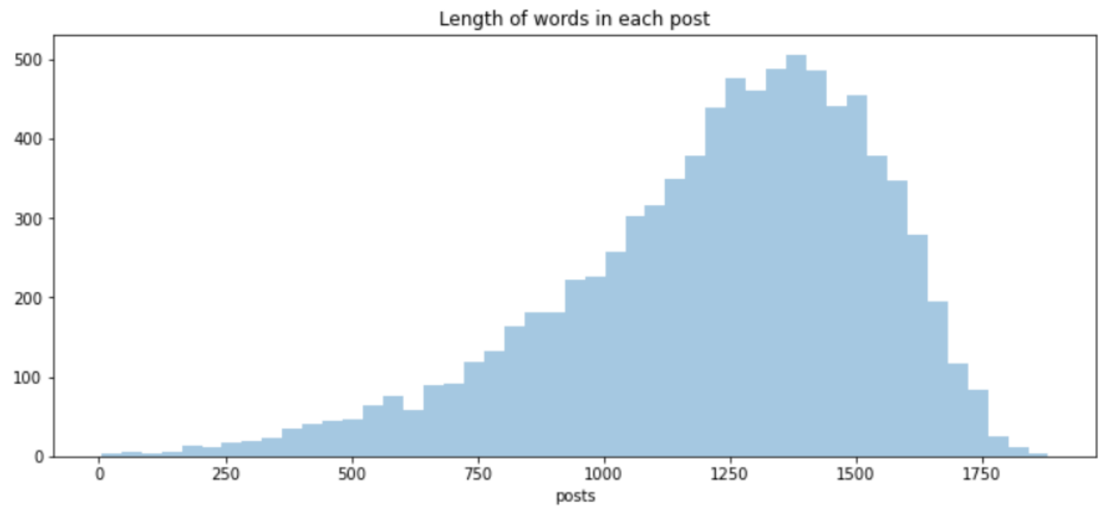
The distribution of length of posts is skewed right, centered around 7800 with most lengths between 4000 to 10000, a range of roughly 10000, some outliers are present in extreme left

```
# Length of the text
plt.figure(figsize = (12,8))
plt.subplot(2,1,1)
dataset['length'] = dataset.posts.apply(lambda x:len(x))
plt.suptitle('Length of the Posts',fontsize = 30 , color = 'blue')
sns.distplot(dataset.length , kde=True , bins = 30 ).set(title = 'Distribution of Length of Posts')
plt.figure(figsize = (12,8))
plt.subplot(2,1,2)
sns.boxplot(dataset.length).set(title = 'Boxplot of Length of Posts')
plt.show()
```





For the Length of words in each posts vs posts



Here y- axis contains length of posts

Confidence interval Calculation

A confidence interval is how _much uncertainty there is with any particular statistic. Confidence intervals are often _used with a margin of error. It tells you how confident you can be that the results from a poll or survey reflect what you would expect to find if it were possible to **survey the entire population**. The Formula for calculating Confidence interval is given below

$$CI = \bar{x} \pm z\frac{s}{\sqrt{n}}$$

CI = confidence interval

$\bar{x}$ = sample mean

z = confidence level value

s = sample standard deviation

n = sample size

```
def CI(words):
    unbiased_point_estimate = np.mean(words)
    std = np.std(words)
    z_star = 1.96
    estimated_se = std/len(words)**0.5

    lcb = np.around(unbiased_point_estimate - z_star*estimated_se,decimals = 2)
    ucb = np.around(unbiased_point_estimate + z_star*estimated_se,decimals = 2)
    return (lcb,ucb)

CI(words)

mean_length_word = {}
confidence_interval = {}
def category_length(data , category):
    dx = dataset[dataset.type == category]
    words = dx['posts'].str.split().map(lambda x:len(x))
    mean_length_word[category] = np.around(np.mean(words),decimals = 2)
    confidence_interval[category] = CI(words)
    sns.distplot(words).set(title = 'Length of word in ' + category)
    plt.show()
```

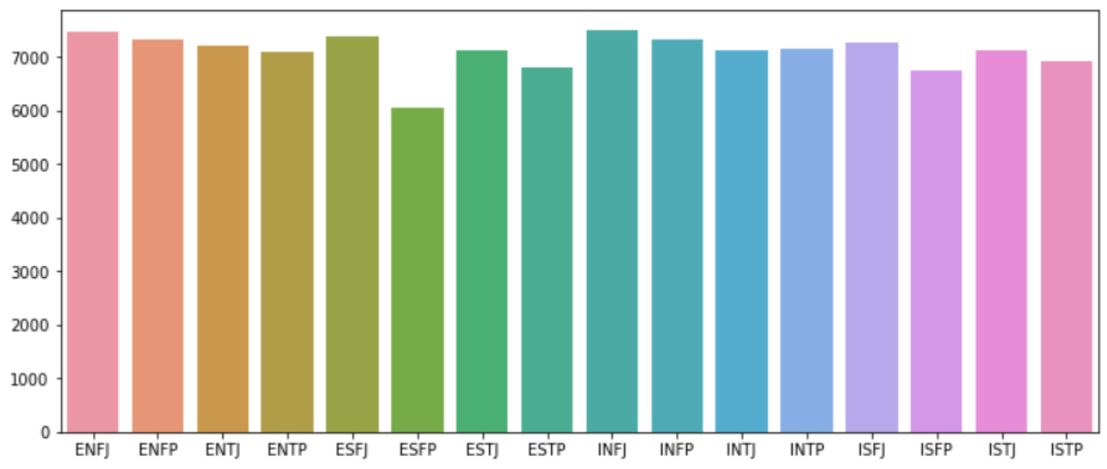
```
confidence_interval
```

```
{'INFJ': (1263.06, 1293.8),
 'ENTP': (1184.99, 1227.0),
 'INTP': (1181.19, 1214.33),
 'INTJ': (1175.73, 1213.42),
 'ENTJ': (1180.78, 1255.39),
 'ENFJ': (1242.47, 1330.7),
 'INFP': (1230.41, 1258.7),
 'ENFP': (1238.19, 1283.35),
 'ISFP': (1096.06, 1176.71),
 'ISTP': (1131.1, 1200.02),
 'ISFJ': (1187.11, 1295.48),
 'ISTJ': (1168.25, 1258.19),
 'ESTP': (1095.78, 1229.41),
 'ESFP': (912.8, 1131.45),
 'ESTJ': (1136.8, 1322.28),
 'ESFJ': (1185.29, 1395.66)}
```

With 95% confidence, the length of word in each category is estimated to be in between 1095 to 1323. There is not a significant difference but there are some categories which shows clear difference, like INFJ and (ENTP, INTP, INTJ, ENTJ, INFP, ISFP, ISTP, ISTJ, ESTP, ESFP)

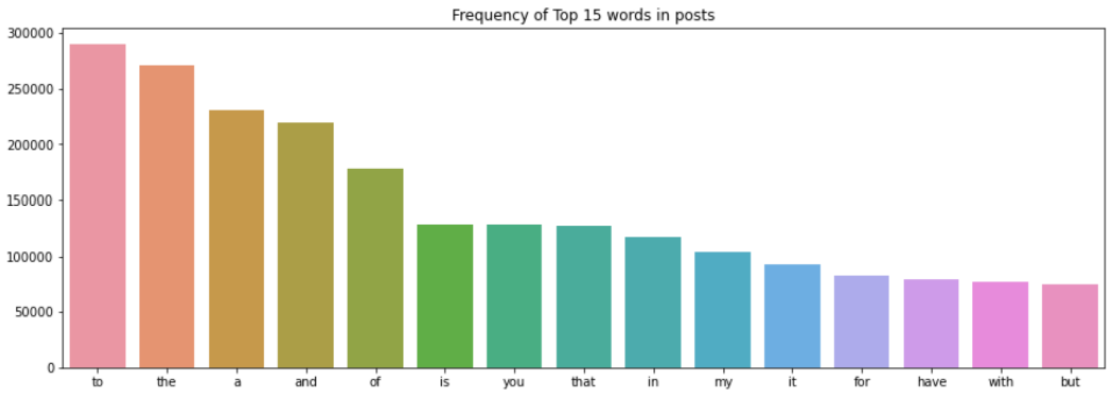
Also, Average Length of posts in each category is almost same

```
dx = dataset.groupby(['type'])['length'].apply(lambda x: np.mean(x))
dx
plt.figure(figsize = (12,5))
sns.barplot(x = list(dx.index) , y = dx.values)
```



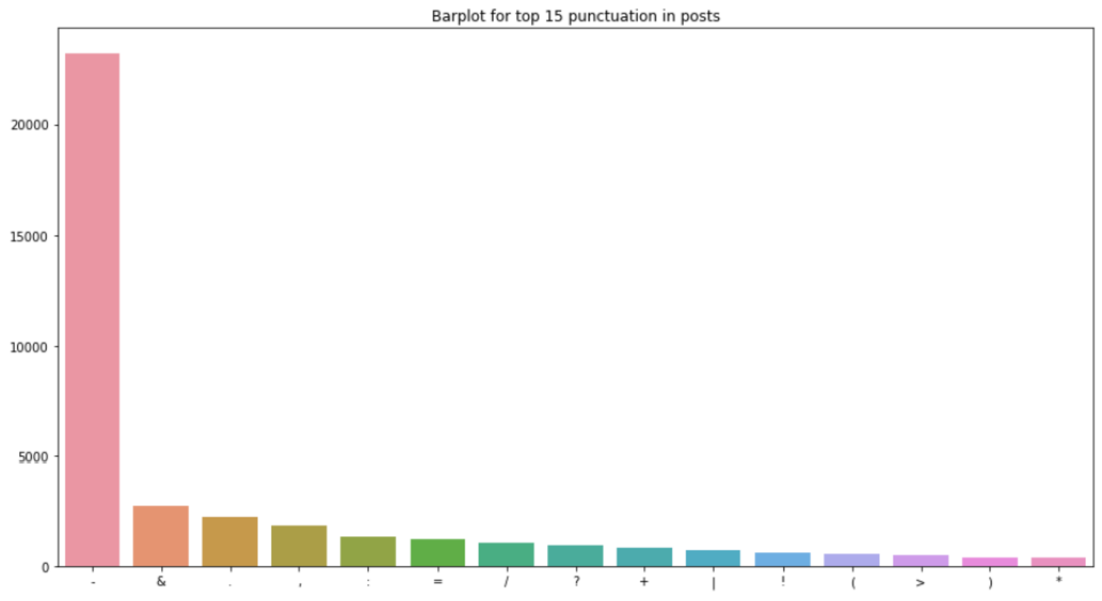
Now we see the Frequency of Top 15 words in posts so that we can remove stopwords and keep only those words which are of importance to our model.

```
corpus = create_corpus(dataset)

dic = defaultdict(int)
for word in corpus:
    if word in stop:
        dic[word] += 1
    else:
        dic[word] = 1

top = sorted(dic.items() , key = lambda x:x[1] , reverse = True)[:15]

plt.figure(figsize = (15,5))
x , y = zip(*top)
sns.barplot(list(x) , list(y)).set(title = 'Frequency of Top 15 words in posts')
```



Also, we will analyse the top punctuation marks in our data (which can be conveniently cleaned to make data cleaner and more balanced).

```
plt.figure(figsize = (15,8))
corpus = create_corpus(dataset)
dic = defaultdict(int)
punctuation = string.punctuation

for word in corpus:
    if word in punctuation:
        dic[word] += 1

top = sorted(dic.items() , key = lambda x:x[1] , reverse = True)[:15]

x,y = zip(*top)
sns.barplot(list(x) , list(y)).set(title = 'Barplot for top 15 punctuation in posts')
```



We find that - (hyphen) is the most occurring punctuation mark.

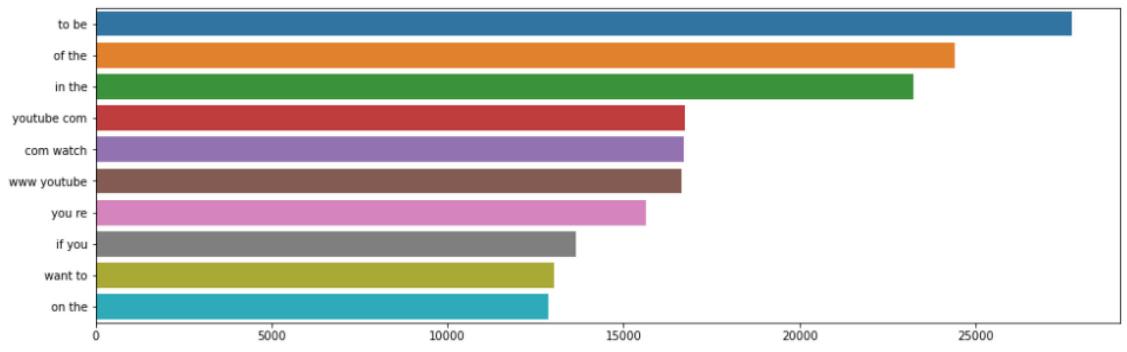
After that we find the combination of words which are found together frequently. For this we generate two-word combinations called bigrams. A **bigram** or **diagram** is a sequence of two adjacent elements from a string of tokens, which are typically letters, syllables, or words.

Now, we perform one hot encoding or Count vectorisation to Convert a collection of text documents to a matrix of token counts for easy processing. Then applying fit() method on the object returned by count vectorizer. fit() method "learn(s) a vocabulary dictionary of all tokens in the raw documents", i.e. it creates a dictionary of tokens (by default the tokens are words separated by spaces and punctuation) that maps each single token to a position in the output matrix. Here we use the corpus generated earlier.

Whenever we apply any algorithm in NLP, it works on numbers. We cannot directly feed our text into that algorithm. Hence, Bag of Words model is used to pre-process the text by converting it into a *bag of words*, which keeps a count of the total occurrences of most frequently used words.

```
def get_top_tweet_bigrams(corpus,n = None):
    vec = CountVectorizer(ngram_range=(2,2)).fit(corpus)
    bag_of_words = vec.transform(corpus)
    sum_of_words = bag_of_words.sum(axis = 0)
    words_freq = [(word,sum_of_words[0,idx]) for word,idx in vec.vocabulary_.items()]
    words_freq = sorted(words_freq , key = lambda x:x[1] , reverse = True)
    return words_freq[:n]
```

```
plt.figure(figsize=(16,5))
top_tweet_bigrams=get_top_tweet_bigrams(dataset.posts)[:10]
x,y=map(list,zip(*top_tweet_bigrams))
sns.barplot(x=y,y=x)
```



Cleaning the Data
Cleaning of Dataset is performed which includes activities such as conversion into lower case, removal of text in square brackets, removing links, unnecessary punctuation marks and words containing numbers. Next step is the pre processing of Data which consist of tokenizing the data and removing stopwords. **Stopwords** are the **words** in any language which does not add much meaning to a sentence like the, is, at, which etc.

```python
def clean_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuation
    and remove words containing numbers.'''
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text

def combine_text(text):
    return ' '.join(text)

def text_preprocessing(text):
    tokenizer = nltk.tokenize.RegexpTokenizer(r'\w+')
    no_punc = clean_text(text)
    tokenized_text = tokenizer.tokenize(no_punc)
    remove_stopwords = [w for w in tokenized_text if w not in stopwords.words('english')]
    combined_text = combine_text(remove_stopwords)

    return combined_text
```

Labelling of Data according to myers briggs personality indicator is Done . Also we add two columns called ‘description’ and ‘clean text length’ to store additional information about the data.

```python
dataset['clean_posts'] = dataset['posts'].apply(lambda x: text_preprocessing(x))
```

```python
mbti = {'I':'Introversion', 'E':'Extroversion', 'N':'Intuition',
        'S':'Sensing', 'T':'Thinking', 'F': 'Feeling',
        'J':'Judging', 'P': 'Perceiving'}
dataset['description'] = dataset.type.apply(lambda x:' '.join([mbti[l] for l in list(x)]))
dataset['clean_text_length'] = dataset.clean_posts.apply(lambda x:len(x))
```

Split training data and test data in 80-20 ratio

```python
from sklearn.preprocessing import LabelEncoder
dx = dataset[['clean_posts','type']]
encoder = LabelEncoder()
dx['type_enc'] = encoder.fit_transform(dx.type)
dx.head()
```

	clean_posts	type	type_enc
0	intj moments sportscenter top ten plays pranks...	INFJ	8
1	im finding lack posts alarmingsex boring posit...	ENTP	3
2	good one course say know thats blessing cursed...	INTP	11
3	dear intp enjoyed conversation day esoteric ga...	INTJ	10
4	youre firedthats another silly misconception a...	ENTJ	2

```
count_vectorizer = CountVectorizer()
train_vectors = count_vectorizer.fit_transform(train_data.clean_posts)
test_vectors = count_vectorizer.transform(test_data.clean_posts)
```

```
tfidf_vectorizer = TfidfVectorizer(min_df = 2, max_df = 0.5, ngram_range = (1 , 2))
train_tfidf = tfidf_vectorizer.fit_transform(train_data.clean_posts)
test_tfidf = tfidf_vectorizer.transform(test_data.clean_posts)
```

Here Two models are used for Classification - Logistic Regression and XGBoost Classifer. **XGBoost** is an implementation of gradient boosted decision trees designed for speed and performance in which we use

```
from sklearn.linear_model import LogisticRegression
clf = LogisticRegression(class_weight='balanced' , C = 0.005)
score = model_selection.cross_val_score(clf , train_vectors , train_data['type_enc'] , cv
= 5 , scoring = 'f1_micro')
score
```

```
clf_xgb = xgb.XGBClassifier(max_depth = 7 , n_estimators = 200 , colsample_bytree = 0.8 ,
subsample = 0.8 , nthread = 10 , learning_rate = 0.1)
scores = model_selection.cross_val_score(clf_xgb , train_vectors , train_data['type_enc']
, cv = 5 , scoring = 'f1_micro')
scores
```

```
array([0.63904899, 0.67002882, 0.65057637, 0.66570605, 0.63904899])
```

```
clf_xgb.fit(train_vectors , train_data.type_enc)
```

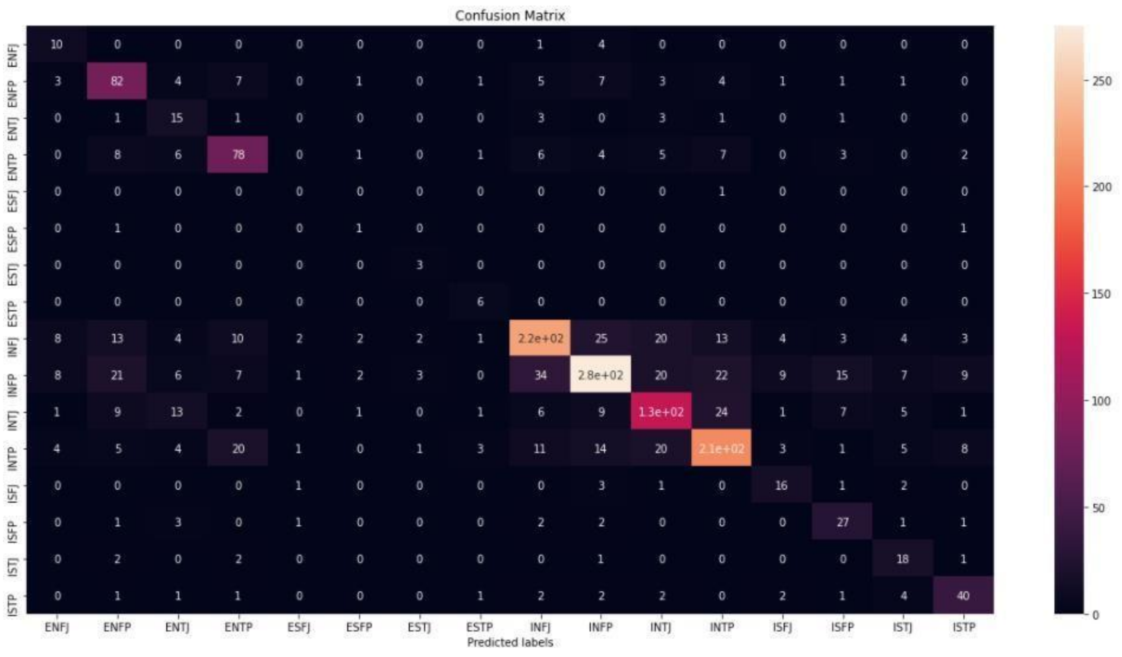
```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
              colsample_bynode=1, colsample_bytree=0.8, gamma=0, gpu_id=-1,
              importance_type='gain', interaction_constraints='',
              learning_rate=0.1, max_delta_step=0, max_depth=7,
              min_child_weight=1, missing=nan, monotone_constraints='()',
              n_estimators=200, n_jobs=10, nthread=10, num_parallel_tree=1,
              objective='multi:softprob', random_state=0, reg_alpha=0,
              reg_lambda=1, scale_pos_weight=None, subsample=0.8,
              tree_method='exact', validate_parameters=1, verbosity=None)
```

```
y_pred = clf_xgb.predict(test_vectors)
y_test = test_data.type_enc
```

```
from sklearn.metrics import accuracy_score
accuracy_score(y_test,y_pred)
```

```
0.6547550432276658
```

```
clf_xgb.predict(test_vectors)
cm = confusion_matrix(y_pred,y_test)

plt.figure(figsize = (20,10))
ax= plt.subplot()
sns.heatmap(cm, annot=True, ax = ax); #annot=True to annotate cells

# labels, title and ticks
ax.set_xlabel('Predicted labels');ax.set_ylabel('True labels');
ax.set_title('Confusion Matrix');
ax.xaxis.set_ticklabels(category); ax.yaxis.set_ticklabels(category);
```



	ENFJ	ENFP	ENTJ	ENTP	ESFJ	ESFP	ESTJ	ESTP	INFJ	INFP	INTJ	INTP	ISFJ	ISFP	ISTJ	ISTP
ENFJ	10	0	0	0	0	0	0	0	1	4	0	0	0	0	0	0
ENFP	3	82	4	7	0	1	0	1	5	7	3	4	1	1	1	0
ENTJ	0	1	15	1	0	0	0	0	3	0	3	1	0	1	0	0
ENTP	0	8	6	78	0	1	0	1	6	4	5	7	0	3	0	2
ESFJ	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
ESFP	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	1
ESTJ	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	0
ESTP	0	0	0	0	0	0	0	6	0	0	0	0	0	0	0	0
INFJ	8	13	4	10	2	2	2	1	2.2e+02	25	20	13	4	3	4	3
INFP	8	21	6	7	1	2	3	0	34	2.8e+02	20	22	9	15	7	9
INTJ	1	9	13	2	0	1	0	1	6	9	1.3e+02	24	1	7	5	1
INTP	4	5	4	20	1	0	1	3	11	14	20	2.1e+02	3	1	5	8
ISFJ	0	0	0	0	1	0	0	0	0	3	1	0	16	1	2	0
ISFP	0	1	3	0	1	0	0	0	2	2	0	0	0	27	1	1
ISTJ	0	2	0	2	0	0	0	0	0	1	0	0	0	0	18	1
ISTP	0	1	1	1	0	0	0	1	2	2	2	0	2	1	4	40

CHAPTER-5

WEB TECHNOLOGIES USED

FRONTEND TECHNOLOGIES USED

For front end, we have used HTML, CSS and JAVASCRIPT with BOOTSTRAP as a support.

HTML (Hyper Text Mark-up Language)

HTML provides the essential structure for web pages. A properly written HTML document will not only be readable to the user, but will also convey the structure of the document, the relationship of its content to each other, and allow the user to link to other pages and sites.

HTML can do all of this because it's a mark-up language. That means that HTML is used to mark up content in order to explain what that content is and how it relates to other content on the page. It does this through the use of tags, which are used to identify specific types of content. A p tag, for example, identifies the content as a paragraph, while the nav tag would identify its content as being part of the page's navigation. By knowing the rules of how tags work together, and which tags to use, it's incredibly simple to author an HTML page.

CSS (Cascading Style Sheet)

The standout advantage of CSS is the added design flexibility and interactivity it brings to web development. Developers have greater control over the layout allowing them to make precise section-wise changes.CSS works by creating rules. These rules are simultaneously applied to multiple elements within the site. Eliminating the repetitive coding style of HTML makes development work faster and less monotonous. Errors are also reduced considerably. Since the content is completely separated from the design, changes across the website can be implemented all at once. This reduces delivery times and costs of future edits.

JavaScript

JavaScript, an object scripting language which is used in web pages along with mark-up language HTML. JavaScript allows dynamic content to get execute in a webpage.

JavaScript can make the website more interactive and user-friendliness of JavaScript helps easy navigation of the website and helps designers to guide the visitors with additional information or guide them through walkthroughs. Visual effects can also be achieved with JavaScript.
JavaScript can be used effectively to create special effects like rollover for images.

One of the most important features of JavaScript is DOM manipulation. The DOM (Document Object Model) is an API (Application Programming Interface) for HTML and XML documents. It represents a web document or a web page, that allows developers a way of representing everything on a web document so that it is accessible via a common set of properties and methods. The DOM is browser independent as well as platform independent and it achieves this independence by representing the contents of the web document as a generic tree structure.

BACKEND TECHNOLOGIES USED

FLASK API (Python)

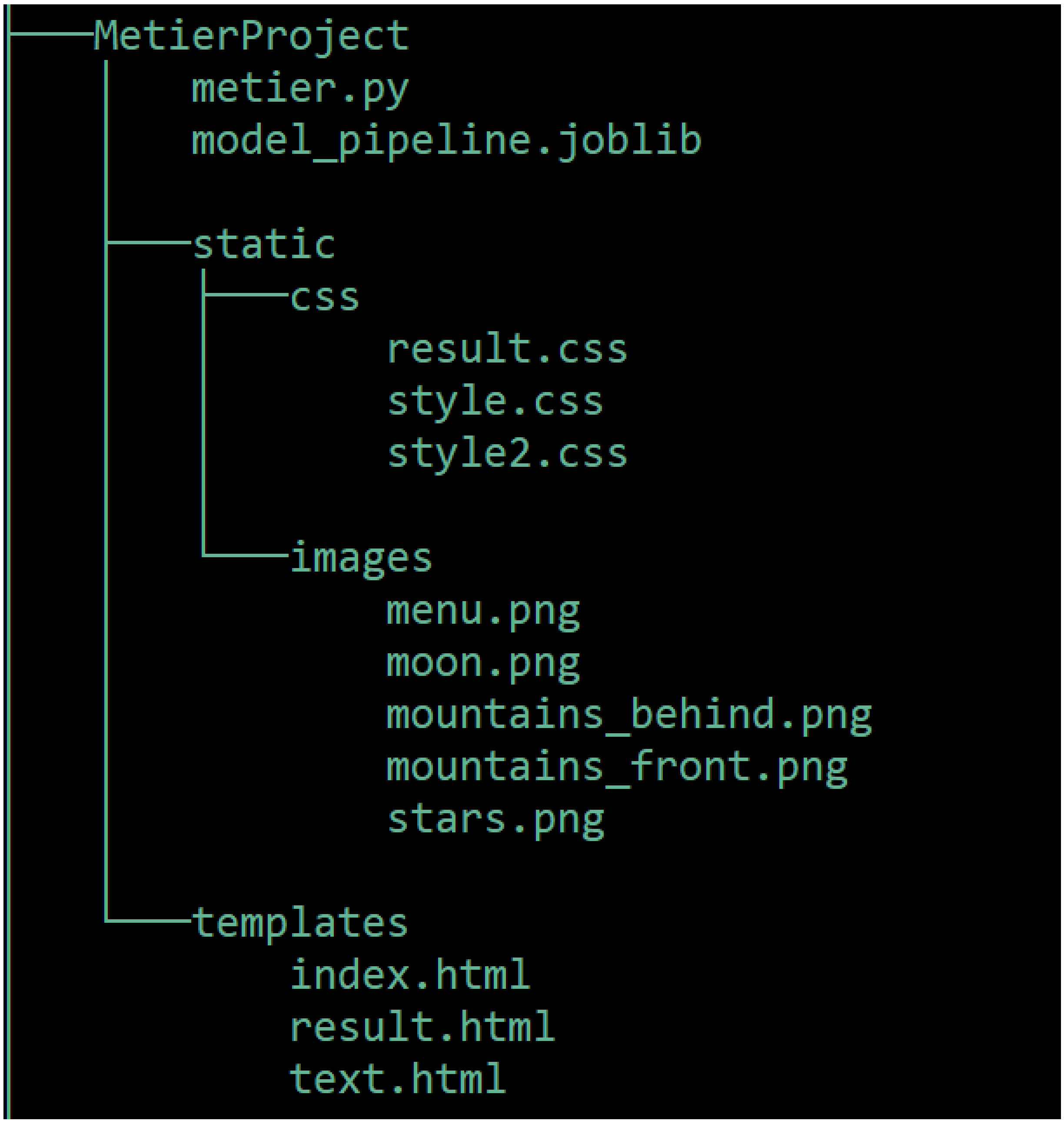
Flask is an API of Python that allows us to build up web-applications. It was developed by Armin Ronacher. Flask’s framework is more explicit than Django’s framework and is also easier to learn because it has less base code to implement a simple web-Application. A Web-Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc. Flask is based on WSGI (Web Server Gateway Interface) toolkit and Jinja2 template engine.

One of the main advantages of using Flask as a framework is that it is really easy to learn and it also supports a built in development server which can be used for quick prototyping of applications and faster and easier debugging .Flask also supports modular design which is really helpful in phase containment of errors. It does not restrict the user for certain features like Database unlike Django which has more strict requirements for its usage of features.
Flask also follows Model- Template -View-Pattern (MTV) pattern . There are three types of components – Models, Templates and views and each of them share their own responsibilities .Model refers to the storage unit of flask from where data can be accessed by writing business logic. Templates are static and dynamic resources that gets rendered by the flask server whenever a request is being made . It is imperative to know that FLASK server only responds to requests made and it does not start the execution of any logic on its own . Views are the code or modules or function which call the templates to be rendered. Flask uses patterns to match the incoming request URL to the view that should handle it.

CHAPTER-6

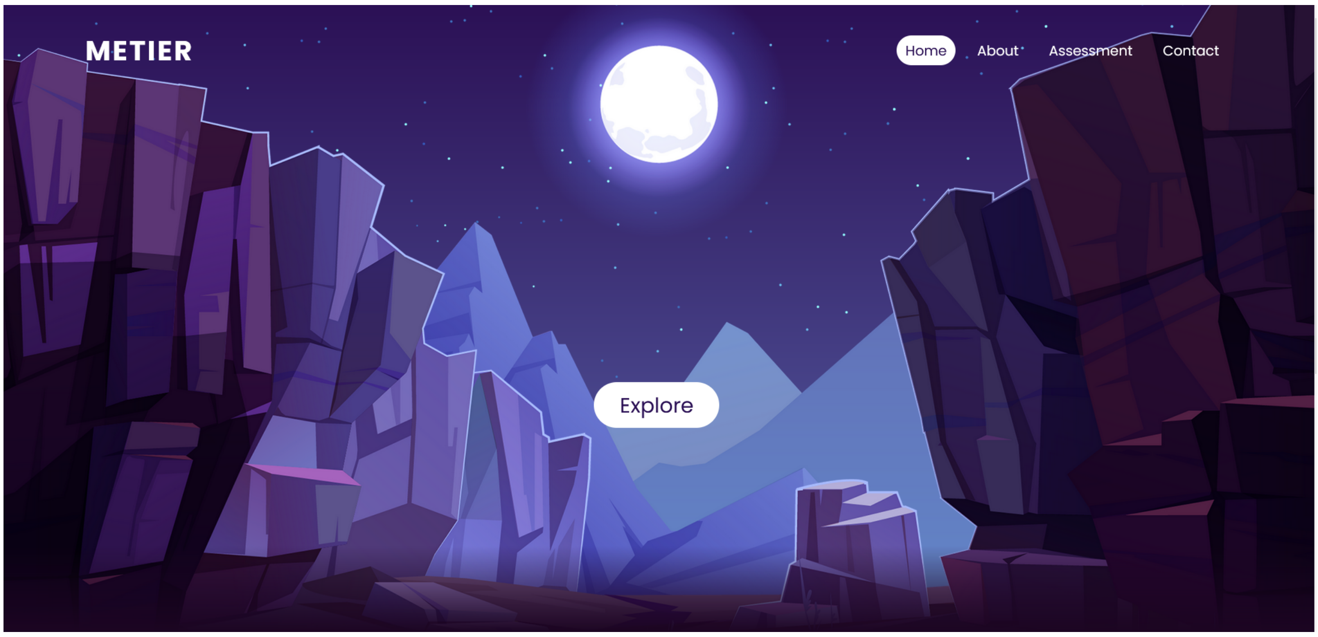
WEB APP IMPLEMENTATION

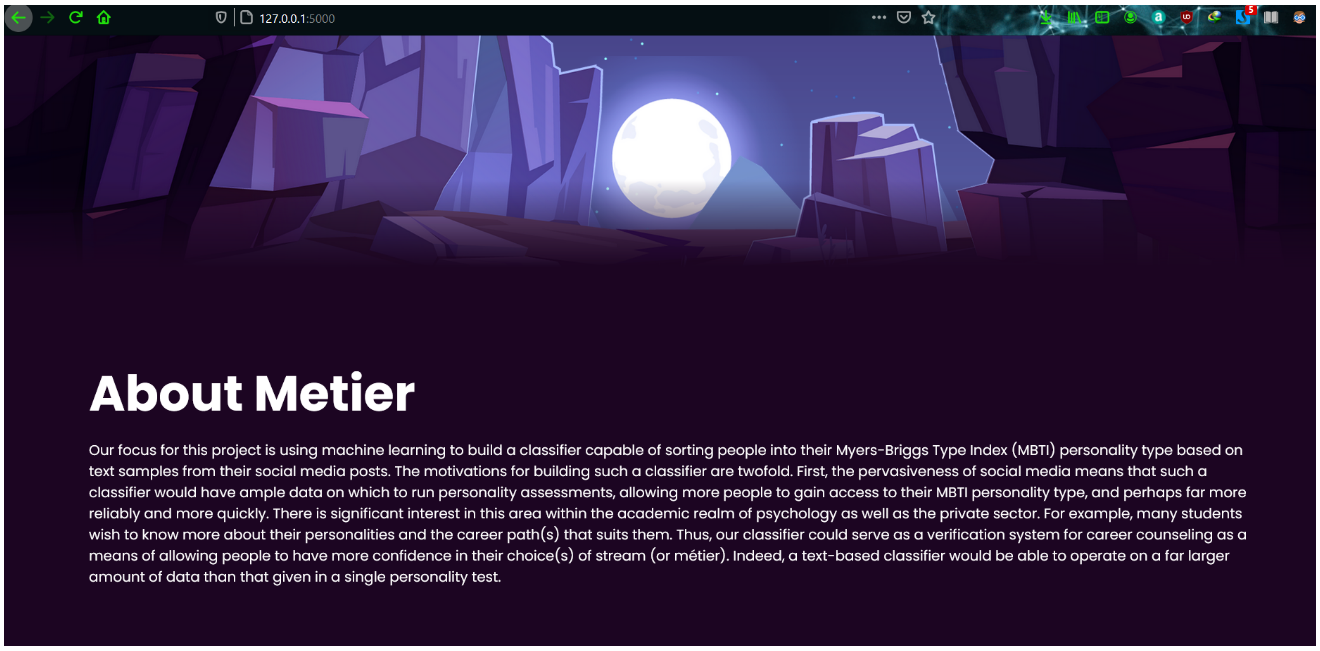
The project was designed using HTML,CSS and Javascript as Frontend with FLASK API serving as a backend to handle requests.
The project tree of the application is as follows



```
──MetierProject
  │   metier.py
  │   model_pipeline.joblib
  │
  ├──static
  │   ├──css
  │   │       result.css
  │   │       style.css
  │   │       style2.css
  │   │
  │   └──images
  │           menu.png
  │           moon.png
  │           mountains_behind.png
  │           mountains_front.png
  │           stars.png
  │
  └──templates
          index.html
          result.html
          text.html
```

Templates – This directory is a default location used by the Flask application for all the templates (html files) used by the application. It contains three files –
<u>Index.html </u>– This webpage serves as a homepage for the application . It gets called as soon as the server is started at the given port.

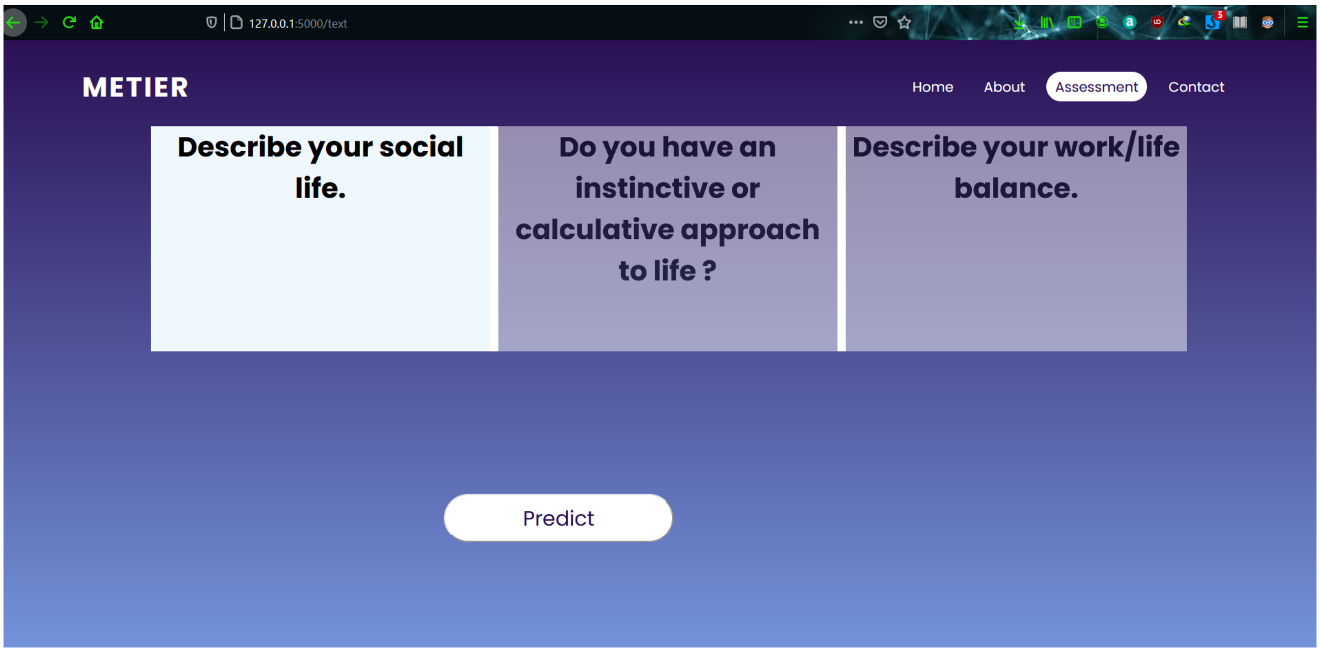


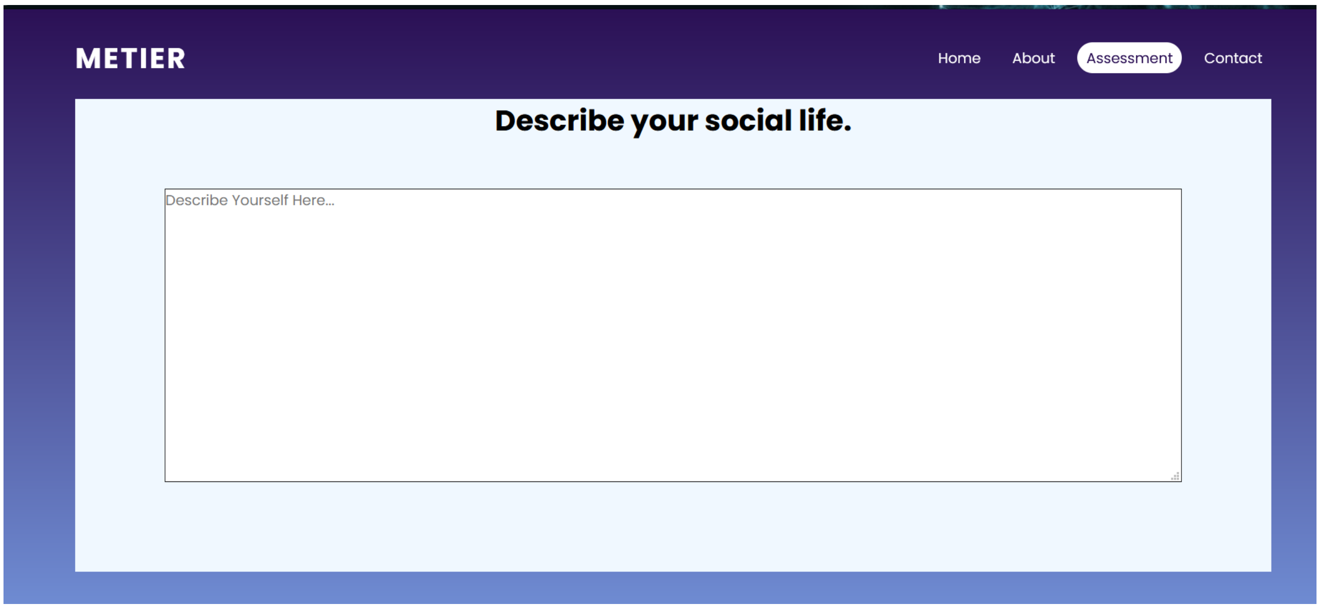


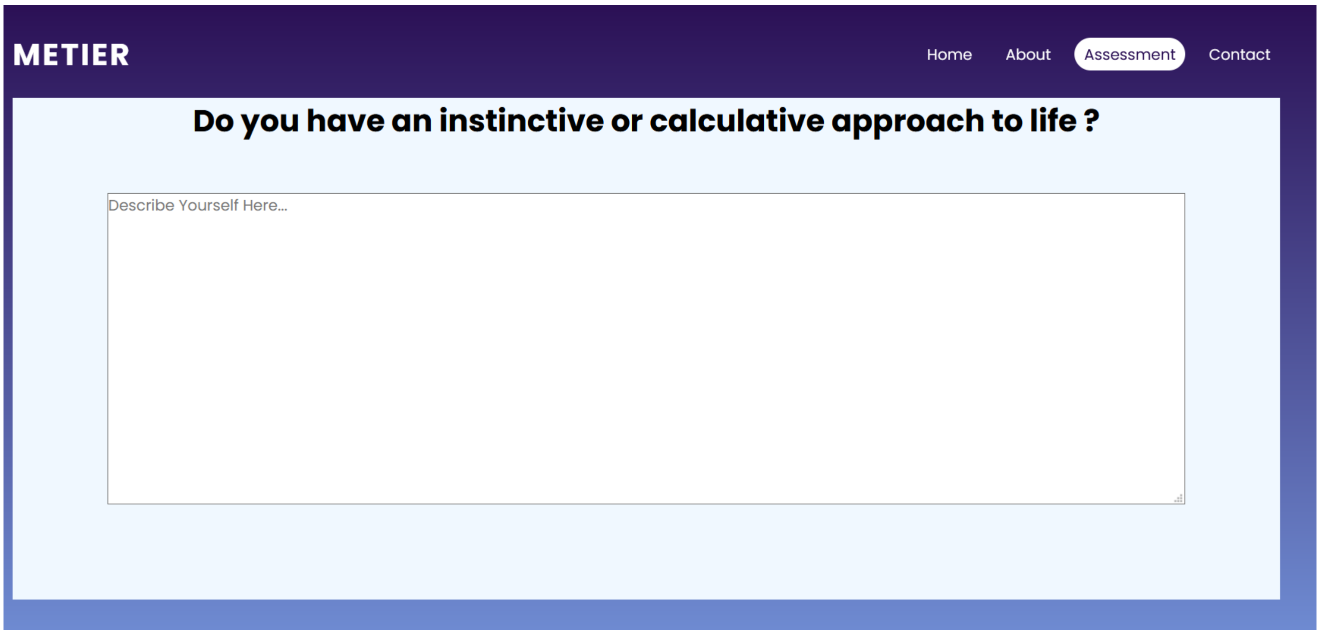
About Metier

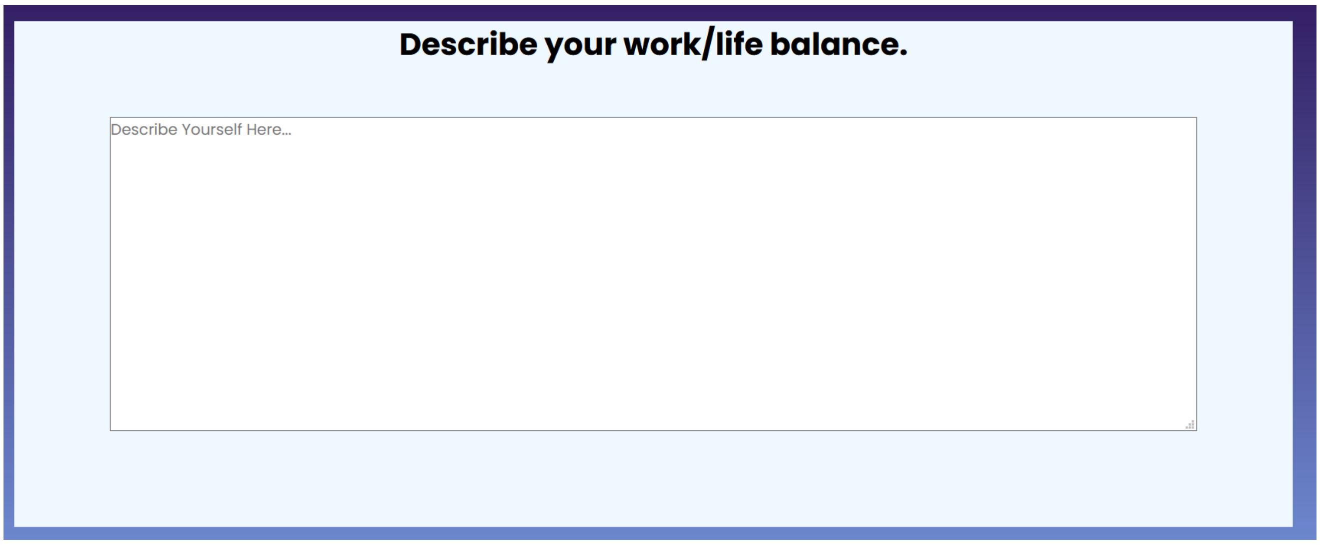
Our focus for this project is using machine learning to build a classifier capable of sorting people into their Myers-Briggs Type Index (MBTI) personality type based on text samples from their social media posts. The motivations for building such a classifier are twofold. First, the pervasiveness of social media means that such a classifier would have ample data on which to run personality assessments, allowing more people to gain access to their MBTI personality type, and perhaps far more reliably and more quickly. There is significant interest in this area within the academic realm of psychology as well as the private sector. For example, many students wish to know more about their personalities and the career path(s) that suits them. Thus, our classifier could serve as a verification system for career counseling as a means of allowing people to have more confidence in their choice(s) of stream (or métier). Indeed, a text-based classifier would be able to operate on a far larger amount of data than that given in a single personality test.

Text.html- This file contains the source code for assessment in which user has to answer three question in order to gather input for personality detection.

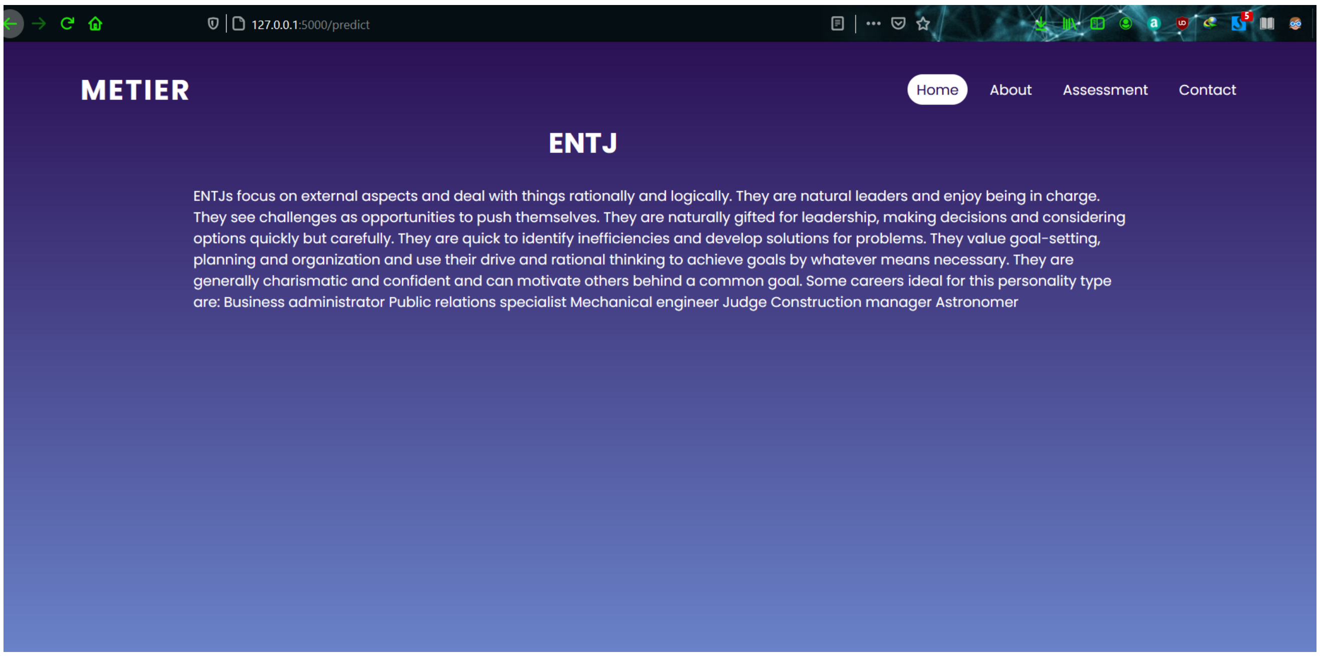

127.0.0.1:5000/text
METIER
Home
About
Assessment
Contact
Describe your social life.
Do you have an instinctive or calculative approach to life ?
Describe your work/life balance.
Predict


METIER
Home
About
Assessment
Contact
Describe your social life.
Describe Yourself Here...


METIER
Home
About
Assessment
Contact
Do you have an instinctive or calculative approach to life ?
Describe Yourself Here...



Result.html- This page shows the resulting class for our prediction.

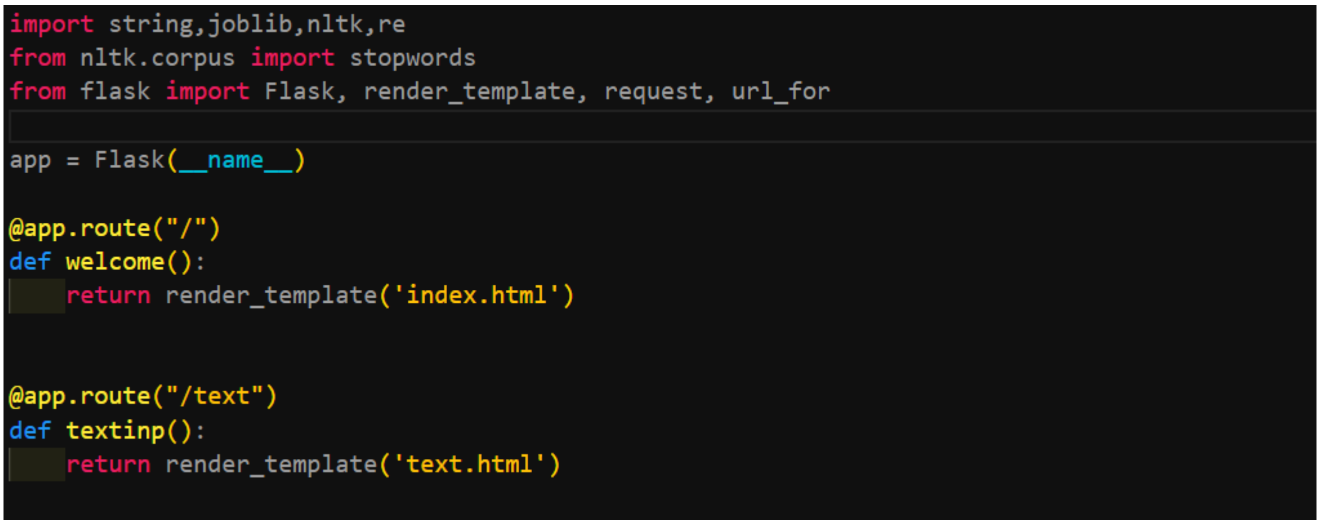


ENTJ

ENTJs focus on external aspects and deal with things rationally and logically. They are natural leaders and enjoy being in charge. They see challenges as opportunities to push themselves. They are naturally gifted for leadership, making decisions and considering options quickly but carefully. They are quick to identify inefficiencies and develop solutions for problems. They value goal-setting, planning and organization and use their drive and rational thinking to achieve goals by whatever means necessary. They are generally charismatic and confident and can motivate others behind a common goal. Some careers ideal for this personality type are: Business administrator Public relations specialist Mechanical engineer Judge Construction manager Astronomer

Static- This folder contains all the CSS files and images used in the project which can be referenced as necessary.
We use a.py file to call the necessary files and define the business logic behind the server .

Metier.py-

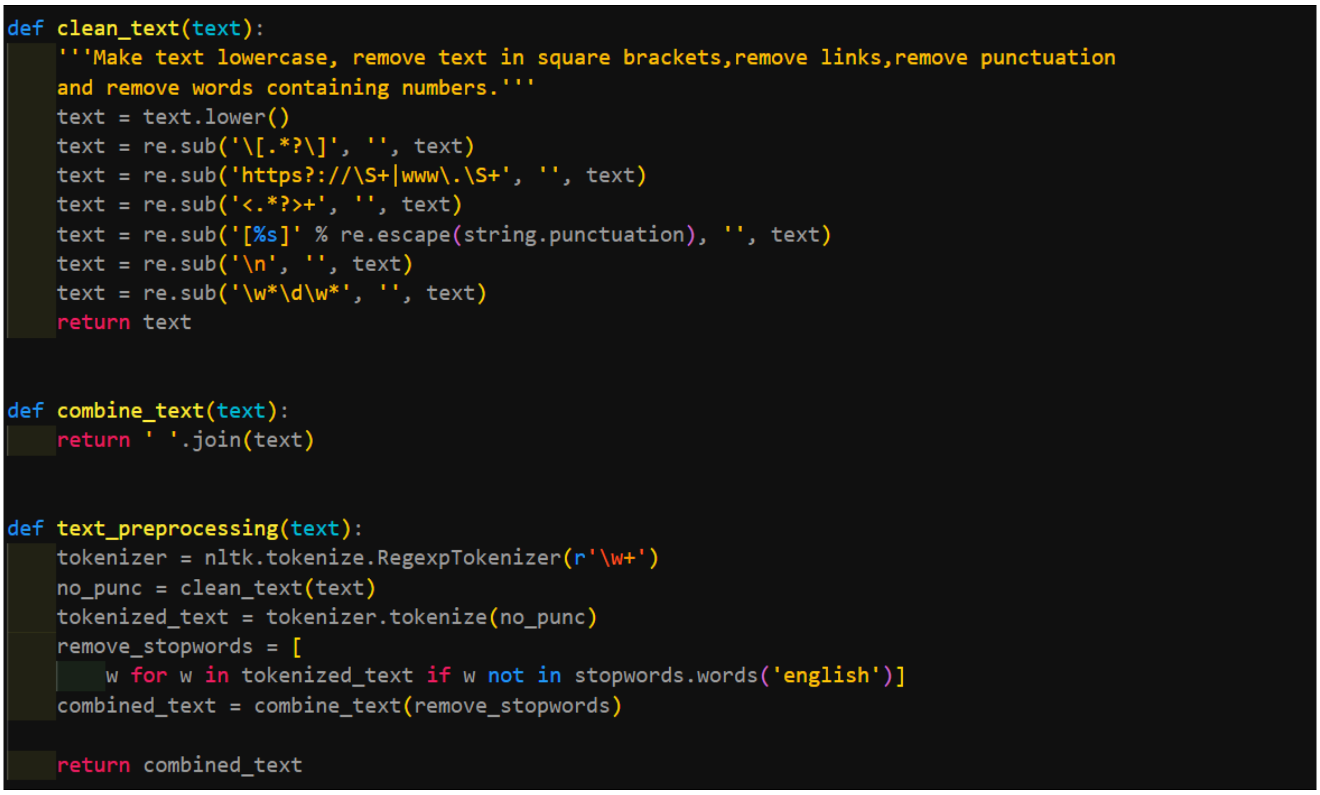


```python
import string,joblib,nltk,re
from nltk.corpus import stopwords
from flask import Flask, render_template, request, url_for

app = Flask(__name__)

@app.route("/")
def welcome():
    return render_template('index.html')


@app.route("/text")
def textinp():
    return render_template('text.html')
```

All the necessary file are imported and then we imported the Flask class. An instance of this class will be our WSGI application.

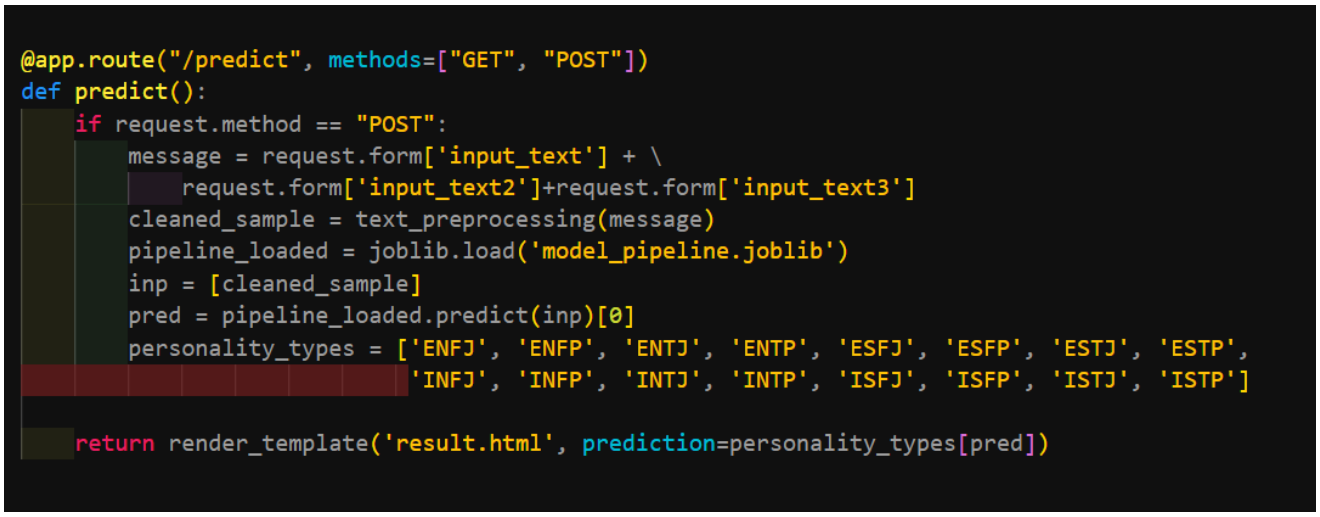
Next we create an instance of this class. The first argument is the name of the application's module or package. Since in this case we are using single module we use __name__ to instantiate it . This is needed so that Flask knows where to look for templates, static files, and so on.

We then use the route() decorator to tell Flask what URL should trigger our function. Here("/") is the home page of the application and we can use the render_template() method to render the webpage in the browser. Flask will look for templates in the templates folder.



```python
def clean_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuation
    and remove words containing numbers.'''
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    return text


def combine_text(text):
    return ' '.join(text)


def text_preprocessing(text):
    tokenizer = nltk.tokenize.RegexpTokenizer(r'\w+')
    no_punc = clean_text(text)
    tokenized_text = tokenizer.tokenize(no_punc)
    remove_stopwords = [
        w for w in tokenized_text if w not in stopwords.words('english')]
    combined_text = combine_text(remove_stopwords)

    return combined_text
```

The three functions shown above is used for preprocessing of the raw input data so that it can serve as an input to pre trained model to make our predictions.



```python
@app.route("/predict", methods=["GET", "POST"])
def predict():
    if request.method == "POST":
        message = request.form['input_text'] + \
            request.form['input_text2']+request.form['input_text3']
        cleaned_sample = text_preprocessing(message)
        pipeline_loaded = joblib.load('model_pipeline.joblib')
        inp = [cleaned_sample]
        pred = pipeline_loaded.predict(inp)[0]
        personality_types = ['ENFJ', 'ENFP', 'ENTJ', 'ENTP', 'ESFJ', 'ESFP', 'ESTJ', 'ESTP',
                            'INFJ', 'INFP', 'INTJ', 'INTP', 'ISFJ', 'ISFP', 'ISTJ', 'ISTP']

    return render_template('result.html', prediction=personality_types[pred])
```

Now we make a new route with methods argument containing the list of *“GET”* and *“POST”* methods. Since Flask only handles GET requests by default so it is important to specify POST inside the list of all the methods. Now request Object stores the request made to server. We check the **request.method==”POST”** to check if a POST request is received. If this condition is true, we extract data from the form by matching the name attribute of the HTML element and then we pass this to cleaning functions defined above to preprocess the data. The model is then loaded and appropriate prediction is passed to render template as an argument to send it to result page for displaying it by the use of Jinja Variables.

CHAPTER 7

CONCLUSION AND FUTURE WORK

5.1 Conclusion

Métier is a ML powered career prediction solution based on NLP text analysis and results are predicted based on Myers-Briggs Personality Type Indicator.

The overall accuracy of our Model which was trained with XGBoost Classifier. XGBoost is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting framework. It is an implementation of gradient boosted decision trees designed for speed and performance. We also cleaned and preprocessed the data by removing unnecessary objects like punctuation marks, articles and helping verbs to help train our model to an optimum result (neither overfitted nor underfitted).

5.2 Future Work

Deep learning can improve the accuracy even if the accuracy is still quite low for some traits. We can do class balancing and Weight balancing which balances our data by altering the weight that each training example carries when computing the loss. Techniques like Oversampling which means that we will create copies of our minority class in order to have the same number of examples as the majority class has.

We can improve our model by manually building a large corpus mapping excerpts (e.g. from famous novels with MBTI types of authors). To gauge the difficulty of identifying MBTI from text, we can cluster text segments based on word embedding similarities to determine whether there exists any non-uniform distribution of personality types (for balanced dataset). This provides a good starting frame of reference to understand how subtle changes to personality traits are when hidden in written data.

We can then implement a bag of words feed-forward neural network as a baseline to understand how simple models in deep-learning can provide insight of hidden personality features by monitoring the outputs of each layer in NN architecture.

Finally, we can go further into a more complex long-short term memory based recurrent neural network and aim to build a more generalizable system that can incorporate context and meaning of writing to determine overall personality types.

For tuning purposes, introducing bidirectionality to our RNNs can incorporate future information along with past information for any given token, which could improve our results even further.

We also hypothesize that a more complex loss function could yield better training. Intuitively, it does not make sense to treat all classifications as disjoint and penalize the same way. In particular, a misclassification of INFJ (Introverted, Intuitive, Feeling, and Judging) as ISFJ (Introverted, Observant, Feeling, and Judging) should not be penalized as much as a misclassification of INFJ (Introverted, Intuitive, Feeling, and Judging) as ESTP (Extraverted, Observant, Thinking, and Prospecting). Thus, we propose exploration of a more suitable weighted loss function by personality dimension and attributes that make up that personality (in comparison how much they are similar) for our task, one that possibly computes cross-entropy (from a probabilistic point of view, the cross-entropy arises as the natural cost function to use if we use a sigmoid or nonlinearity in the output layer of our network, and so as to maximize the likelihood of classifying the input data correctly) and sums up the individual losses.

INDIVIDUAL CONTRIBUTION

MEMBERS		CONTRIBUTION	
ABHISH KUMAR ANAND	1705074	**Report:** Abstract; Literature Survey; Implementation,Future Work; References	**Project:** ML pipeline dev; Back- end; Data collection
SHUBHAM KUMAR MAURYA	1705074	**Report:** Introduction; Conclusion	**Project:** UI/UX Designing; Questionnaire building; Data collection
SIDHARTH PUROHIT	1705078	**Report:** Project Analysis	**Project:** ML engineer (model implementation) ; Questionnaire building; Data collection
SURAJ KUMAR MISHRA	1805951	**Report:** Web Tech Used	**Project:** Web dev (front-end); Data collection

REFERENCES

1. Nguyen, D.; Doǧruöz, A.S.; Rosé, C.P.; Jong, F.D. *Computational sociolinguistics: A survey*. Comput. Linguist. 2016, 42, 537–593
2. Soto, C. J. (2018). *Big Five personality traits*. In M. H. Bornstein, M. E. Arterberry, K. L. Fingerman, & J. E. Lansford (Eds.), The SAGE encyclopedia of lifespan human development (pp. 240-241). Thousand Oaks, CA: Sage
3. Golbeck, J.; Robles, C.; Edmondson, M.; Turner, K. *Predicting personality from Twitter*. In proceedings of IEEE Third International Conference on Privacy, Security, Risk and Trust and IEEE Third International Conference on Social Computing, Boston, USA, 2011.
4. Komisin, M.; Guinn, C. *Identifying personality types using document classification methods*. In Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference, Marco Island, FL, USA, 23–25 May, 2012; pp. 232–237
5. Wan, D.; Zhang, C.; Wu, M.; An, Z. *Personality prediction based on all characters of user social media information*. Chinese national conference on social media processing: Beijing, China, 2014; pp. 220–230.
6. Li, C.; Wan, J.; Wang, B. 16th International Symposium on Distributed Computing and Applications to Business, Engineering and Science, Anyang, China, 2017.
7. Tandera, T.; Suhartono, D.; Wongso, R.; Prasetio, Y. *Personality prediction system from Facebook users*. 2nd International conference on computer science and computational intelligence, Bali, Indonesia, 2017
8. Hernandez, R.; Knight, I.S. *Predicting Myers-Bridge Type Indicator with text classification*. 31st conference on Neural Information processing Systems, 2017.