Stress Analysis of Social Media Post by Machine learning

SmartInternz

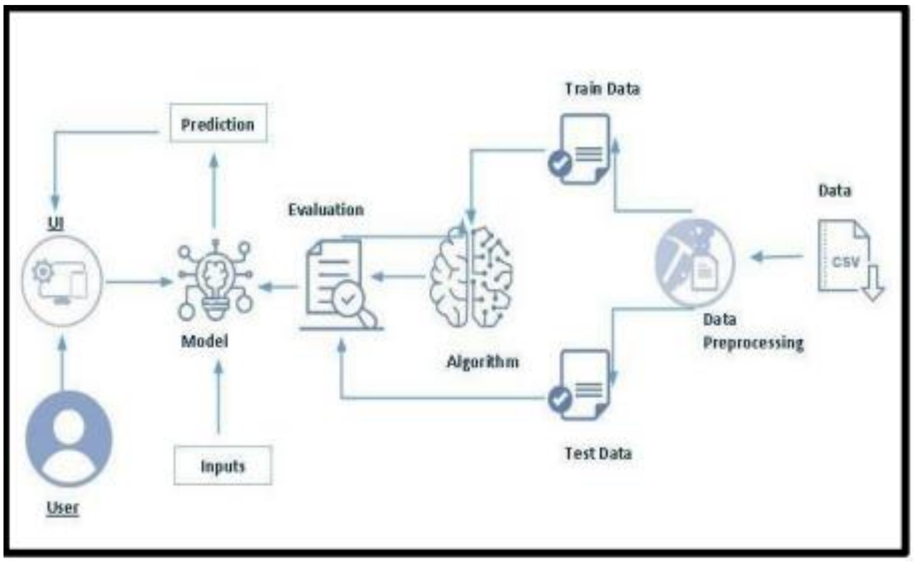
www.smartinternz.com

Stress Analysis of Social Media Post by Machine learning :

Context - We live in an era where there is a surplus of information flowing in every second. Sometimes this leads to stress. Too much stress can negatively impact our health and may lead to headaches, high blood pressure, heart problems, diabetes, skin conditions, asthma, arthritis, depression, and anxiety

Content - A data-set of lengthy multi-domain social media data for identifying stress from five different categories of Reedit communities

Technical Architecture:



Project Flow:

- User interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once model analyses the input the prediction is showcased on the

UITo accomplish this, we have to complete all the activities listed below,

- Define Problem / Problem Understanding
  - Specify the Missing-Value, Migrants problem
  - Literature Survey
- Data Collection & Preparation
  - Collect the data-set
  - Data Preparation
- Exploratory Data Analysis
  - Visual Analysis
- Model Building

    - Performing data Prepossessing
    - Performing data Vectorizer Techniques
    - Apply the Machine learning algorithms
- Performance Testing & Hyper-parameter Tuning
    - Testing model with multiple evaluation metrics
    - Comparing model accuracy
- Model Deployment
    - Save the best model
    - Integrate with Web Framework

- Project Demonstration & Documentation
    - Project Documentation-Step by step project development procedure

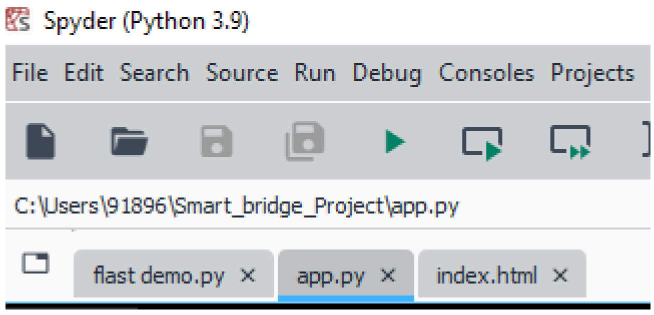
Prior Knowledge:

You must have prior knowledge of following topics to complete this project.

- ML Concepts
    - Supervised learning: https://www.javatpoint.com/supervised-machine-learning
    - Unsupervised learning: https://www.javatpoint.com/unsupervised-machine-learning
- Decision tree: https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm
- Random forest: https://www.javatpoint.com/machine-learning-random-forest-algorithm
- Xgboost: https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/
- Evaluation metrics: https://www.analyticsvidhya.com/blog/2019/08/11-important-model-evaluation-error-metrics/
- Flask Basics : https://www.youtube.com/watch?v=lj4I_CvBnt0

Project Structure:

- **Create the Project folder which contains files as shown below:**


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects
C:\Users\91896\Smart_bridge_Project\app.py
flast demo.py
app.py
index.html

We are building a flask application which needs HTML pages stored in the templates folder and a python script app.py for scripting.

- My_model.pkl is our saved model. Further we will use this model for flask integration.
- Data Folder contains the Data-set used
- The Notebook file contains procedure for building the model.

Milestone 1: Define Problem / Problem Understanding:

Activity 1: Specify the Classification problem : We live in an era where there is a surplus of information flowing in every second. Sometimes this leads to stress so here we have to Check the sentiment of the post and Comment of their posts and we have to predict or Identify He or She is In Stress or No Stress.

Conflicting classification has been assigned to the LABLE column. It is a binary representation of whether Based on his/her Comment he or she is in Stress or not, where 0 NOT Stress classifications and 1 represents Stress classifications.

Activity 2: Literature Survey :

The objective is to predict The Person is In stress or not Stress To find the Solution why he or she is in Stress what are the Reason And how we can help them so that in future it won't repeat for them based on there social media post and and comment we will fetch information and then we will do sentimental Analysis on the text and then we will able to predict that he or she is in Stress or not Stress.

Milestone 2: Data Collection & Preparation:

Activity 1: Collect the data-set

There are many popular open sources for collecting the data. Eg: kaggle.com, UCI repository, etc.

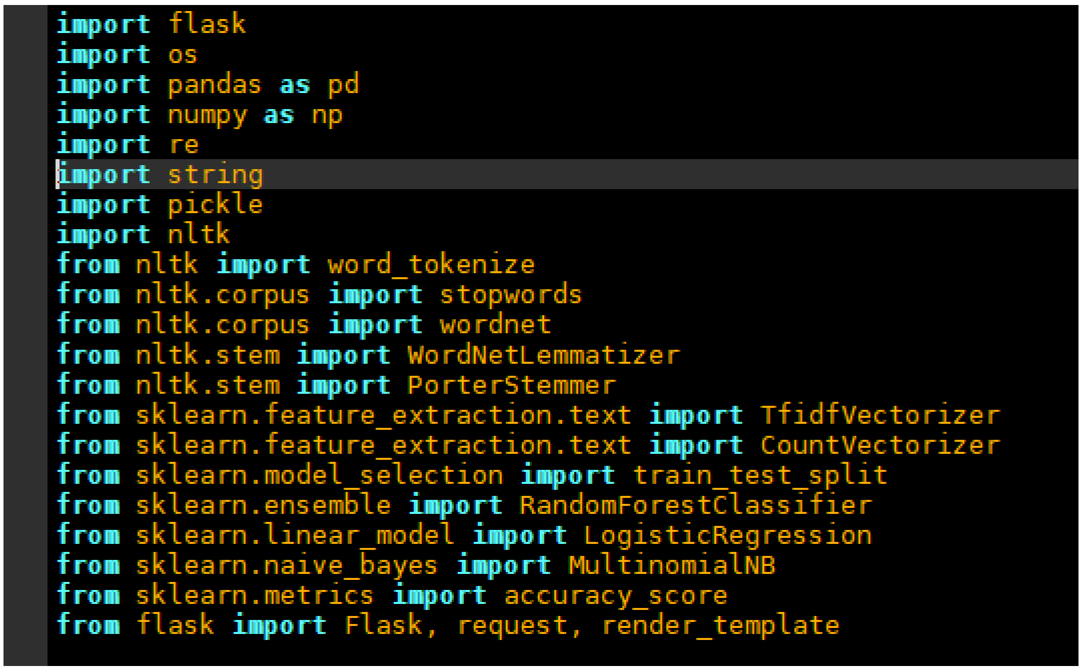
In this project we have used .csv data. This data is downloaded from kaggle.com. Please refer to the link given below to download the data-set.

Link: https://www.kaggle.com/datasets/ruchi798/stress-analysis-in-social-media?resource=download

As the data-set is downloaded. Let us read and understand the data properly with the help of some visualization techniques and some analyzing techniques.

Activity 1.1: Importing the libraries:Import the necessary libraries

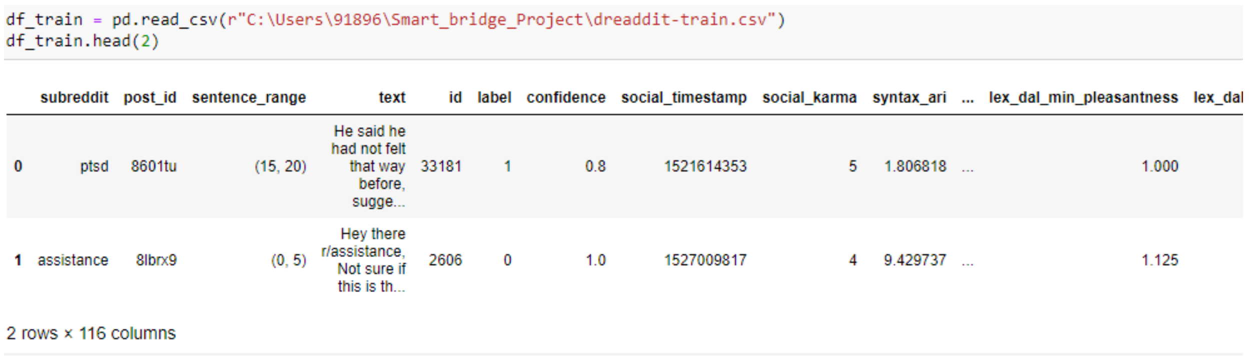
as shown in the image. (optional) Here we have used visualization style asfive thirtyeight.



```
import flask
import os
import pandas as pd
import numpy as np
import re
import string
import pickle
import nltk
from nltk import word_tokenize
from nltk.corpus import stopwords
from nltk.corpus import wordnet
from nltk.stem import WordNetLemmatizer
from nltk.stem import PorterStemmer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score
from flask import Flask, request, render_template
```

Activity 1.2: Read the Dataset :

Our data-set format might be in .csv, excel files, .txt, .json, etc. We can read the dataset with the help of pandas. In pandas we have a function called read_csv() to read the data-set. As a parameter we have to give the directory of the csv file.



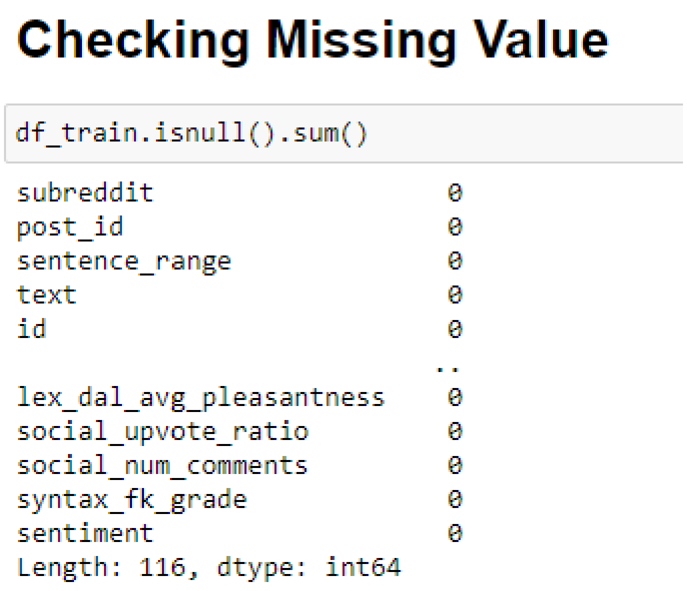
```
df_train = pd.read_csv(r"C:\Users\91896\Smart_bridge_Project\dreaddit-train.csv")
df_train.head(2)
```

	subreddit	post_id	sentence_range	text	id	label	confidence	social_timestamp	social_karma	syntax_ari	...	lex_dal_min_pleasantness	lex_dal
0	ptsd	8601tu	(15, 20)	He said he had not felt that way before, sugge...	33181	1	0.8	1521614353	5	1.806818	...	1.000	
1	assistance	8lbrx9	(0, 5)	Hey there r/assistance, Not sure if this is th...	2606	0	1.0	1527009817	4	9.429737	...	1.125	

2 rows × 116 columns

Activity 2: Data Preparation: As we have understood how the data is, let's per-process the collected data. Thedownload data set is not suitable for training the machine learning model as it might have so much randomness so we need to clean the data-set properly in order to fetch good results. This activity includes the following steps.

- Handling missing values
- Note: These are the general steps of per-processing the data before using it for machine learning. Depending on the condition of your data-set, you may or may not have to go through all these steps.



Checking Missing Value

```
df_train.isnull().sum()
```

```
subreddit                    0
post_id                      0
sentence_range               0
text                         0
id                           0
                            ..
lex_dal_avg_pleasantness     0
social_upvote_ratio          0
social_num_comments          0
syntax_fk_grade              0
sentiment                    0
Length: 116, dtype: int64
```

We have nearly 116 columns ,for that we cant do flask so we have to take the most important take the most important columns that we will do in feature Engineering

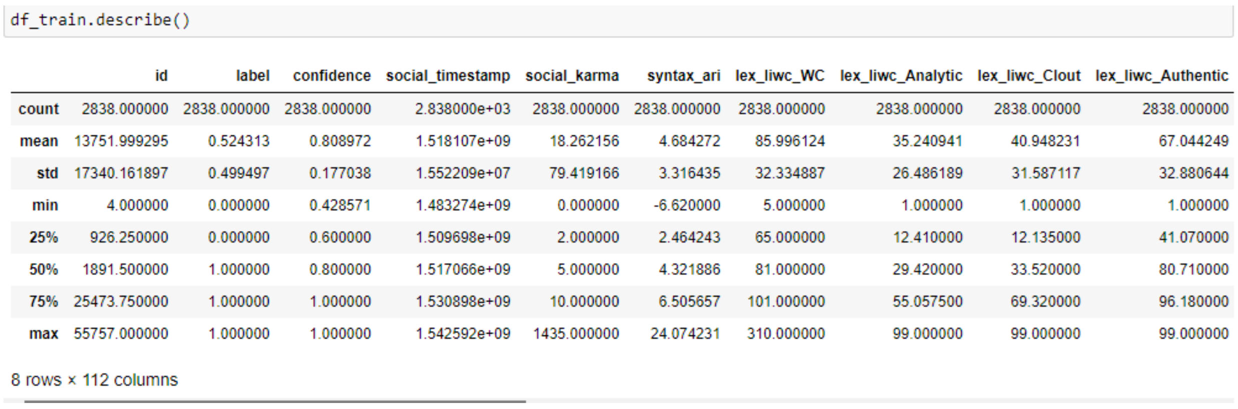
Activity 2.1: Handling missing values:

- For checking the null values, df.isna().any( ) function is used. To sum those null values we use .sum() function. From the below image we found that there are no null values present in our dataset. So we can skip handling the missing values step.

Milestone 3: Exploratory Data Analysis

Activity 1: Descriptive statistical

Descriptive analysis is to study the basic features of data with the statistical process. Here pandas has a worthy function called describe. With this describe function we can understand the unique, top and frequent values of categorical features. And we can find mean, std, min, max and percentile values of continuous features.



```
df_train.describe()
```

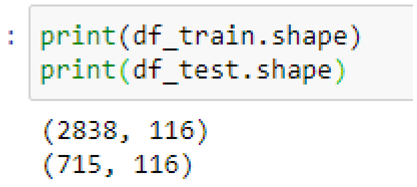
	id	label	confidence	social_timestamp	social_karma	syntax_ari	lex_liwc_WC	lex_liwc_Analytic	lex_liwc_Clout	lex_liwc_Authentic
count	2838.000000	2838.000000	2838.000000	2.838000e+03	2838.000000	2838.000000	2838.000000	2838.000000	2838.000000	2838.000000
mean	13751.999295	0.524313	0.808972	1.518107e+09	18.262156	4.684272	85.996124	35.240941	40.948231	67.044249
std	17340.161897	0.499497	0.177038	1.552209e+07	79.419166	3.316435	32.334887	26.486189	31.587117	32.880644
min	4.000000	0.000000	0.428571	1.483274e+09	0.000000	-6.620000	5.000000	1.000000	1.000000	1.000000
25%	926.250000	0.000000	0.600000	1.509698e+09	2.000000	2.464243	65.000000	12.410000	12.135000	41.070000
50%	1891.500000	1.000000	0.800000	1.517066e+09	5.000000	4.321886	81.000000	29.420000	33.520000	80.710000
75%	25473.750000	1.000000	1.000000	1.530898e+09	10.000000	6.505657	101.000000	55.057500	69.320000	96.180000
max	55757.000000	1.000000	1.000000	1.542592e+09	1435.000000	24.074231	310.000000	99.000000	99.000000	99.000000

8 rows × 112 columns

Activity 2: Visual analysis

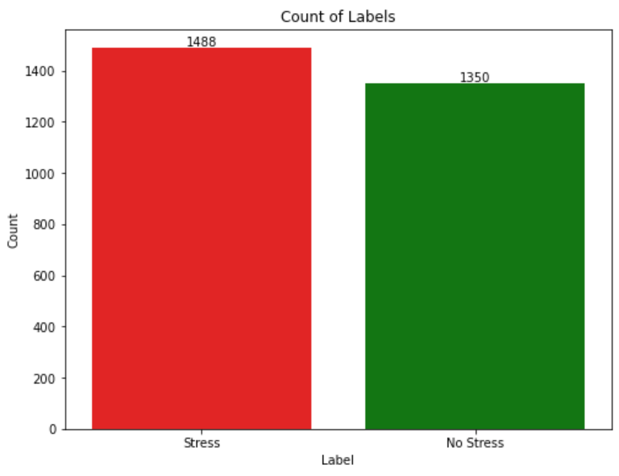
Visual analysis is the process of using visual representations, such as charts, plots, and graphs, to explore and understand data. It is a way to quickly identify patterns, trends, and outliers in the data, which can help to gain insights and make informed decisions

In the below plot ,it will indicate that in which "Region " having high number of missing.

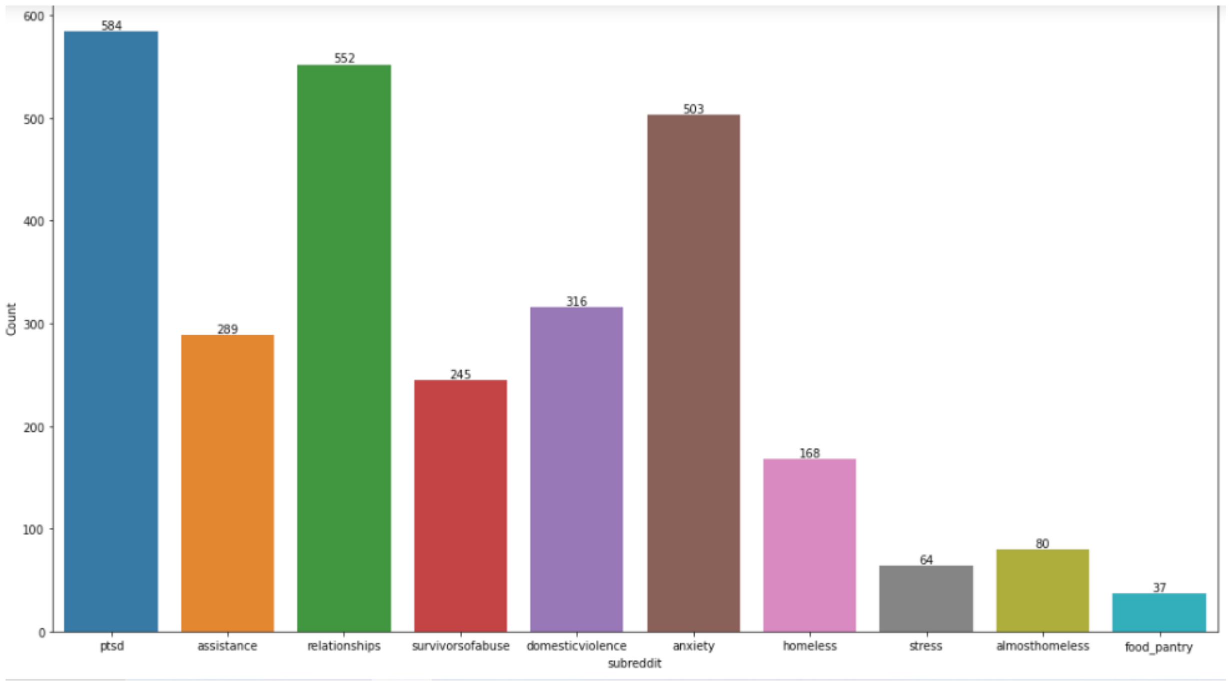


```
print(df_train.shape)
print(df_test.shape)
```

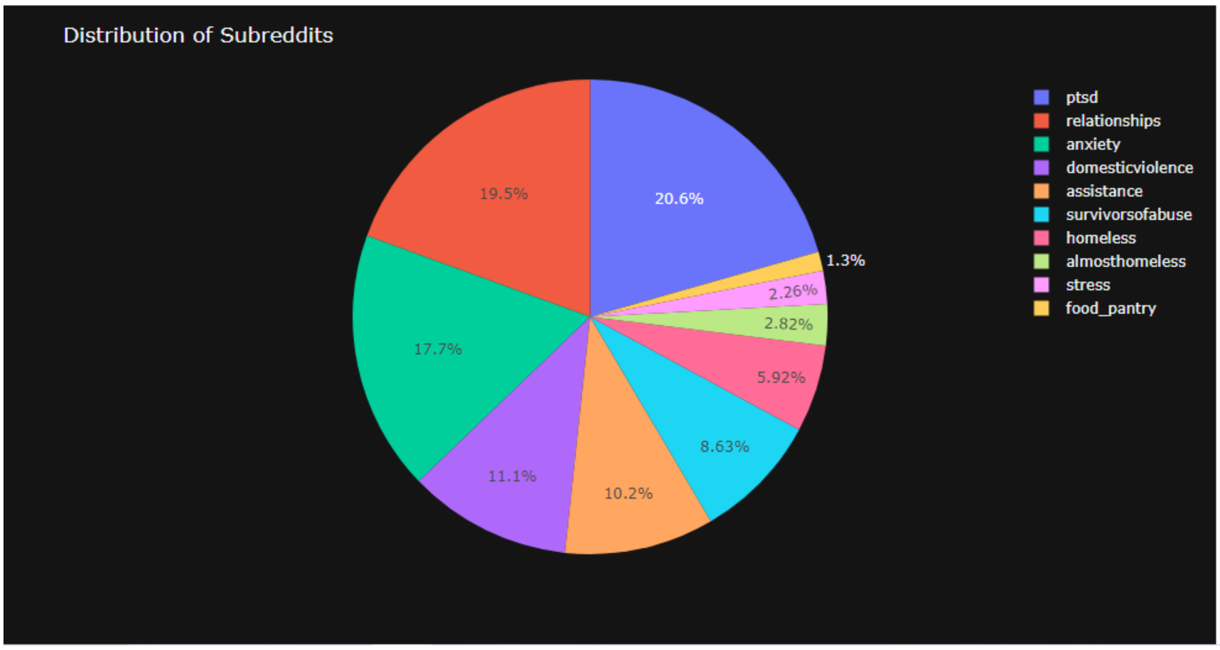
```
(2838, 116)
(715, 116)
```



Here we can see the we have 2838 data point where 1488 are stress and 1350 are not Stress so we can say that we have almost equall data so it is not baies dataset and in my point of view says that there are more then 50% of people are Stress in the time which it very hard to bilive



Count
subreddit
ptsd
assistance
relationships
survivorsofabuse
domesticviolence
anxiety
homeless
stress
almosthomeless
food_pantry
584
289
552
245
316
503
168
64
80
37



Distribution of Subreddits
20.6%
19.5%
17.7%
11.1%
10.2%
8.63%
5.92%
2.82%
2.26%
1.3%
ptsd
relationships
anxiety
domesticviolence
assistance
survivorsofabuse
homeless
almosthomeless
stress
food_pantry

```
df_final["text"] = df_final["text"].apply(pre_porccess)
df_final.sample(10)
```

```
                                                  text      label  sentiment
  64     in class, im always on edge, i cant focus on o...     Stress   0.015000
 724   however, this was months ago and i have heard...     Stress   0.487500
 290     things between us were amicable until that poi...  No Stress   0.137500
 232     i'm pretty broke and almost all of my money fo...  No Stress   0.140000
 105   do you huddle up in the corner of some obscure...     Stress   0.326263
1625        i was very excited to go back to live in the s...  No Stress   0.383763
2095      any help would be appreciated, after this comi...  No Stress   0.233333
2049    she was meeting her boyfriend (the one before ...     Stress   0.125000
2801       i was in a rabbit hole of youtube videos over ...     Stress  -0.052143
1045       they love you and are very loyal to you. they ...  No Stress   0.229167
```

Final Data – Set

Spliting of the data

```
]: #Assign variables; x = features & y = target
   x = df_final.text
   y = df_final.label
```

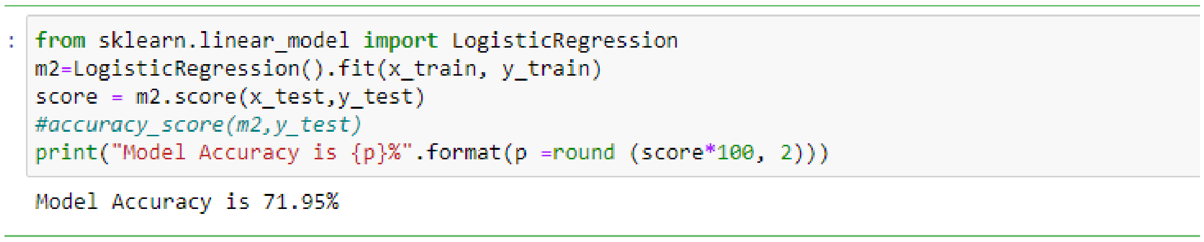
```
]: from sklearn.feature_extraction.text import CountVectorizer
   vect=CountVectorizer(stop_words="english")
   x=vect.fit_transform(x)
```

```
]: from sklearn.model_selection import train_test_split
   x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,random_state=99)
```

Milestone 4: Model Building

Activity 1: Training the model in multiple algorithms

Now our data is cleaned and it’s time to build the model. We can train our data on different algorithms. For this project we are applying three classification algorithms. The best model is saved based on its performance.

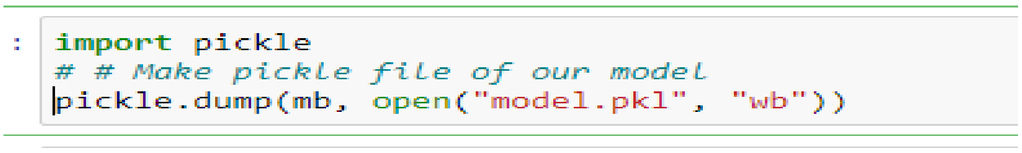


```
: from sklearn.linear_model import LogisticRegression
  m2=LogisticRegression().fit(x_train, y_train)
  score = m2.score(x_test,y_test)
  #accuracy_score(m2,y_test)
  print("Model Accuracy is {p}%".format(p =round (score*100, 2)))

  Model Accuracy is 71.95%
```

Milestone 6: Model Deployment

Activity 1: Save the best model

Saving the best model after comparing its performance using different evaluation metrics means selecting the model with the highest performance. This can be useful in avoiding the need to retrain the model every time it is needed and also to be able to use it in the future.



```
: import pickle
  # # Make pickle file of our model
  pickle.dump(mb, open("model.pkl", "wb"))
```

Activity 2: Integrate with Web Framework:In this section, we will be building a web application that is integrated to the model we built. A UI is provided for the uses where he has to enter the values for predictions. The enter values are given to the saved model and prediction is showcased on the UI.

This section has the following tasks

Building HTML Pages

- Building server-side script
- Run the web application

Activity 2.1: Building Html Page:

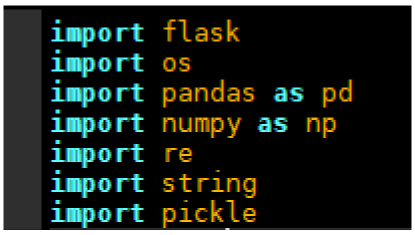
For this project create HTML file namely

 Index10.html

and save them in the templates folder.

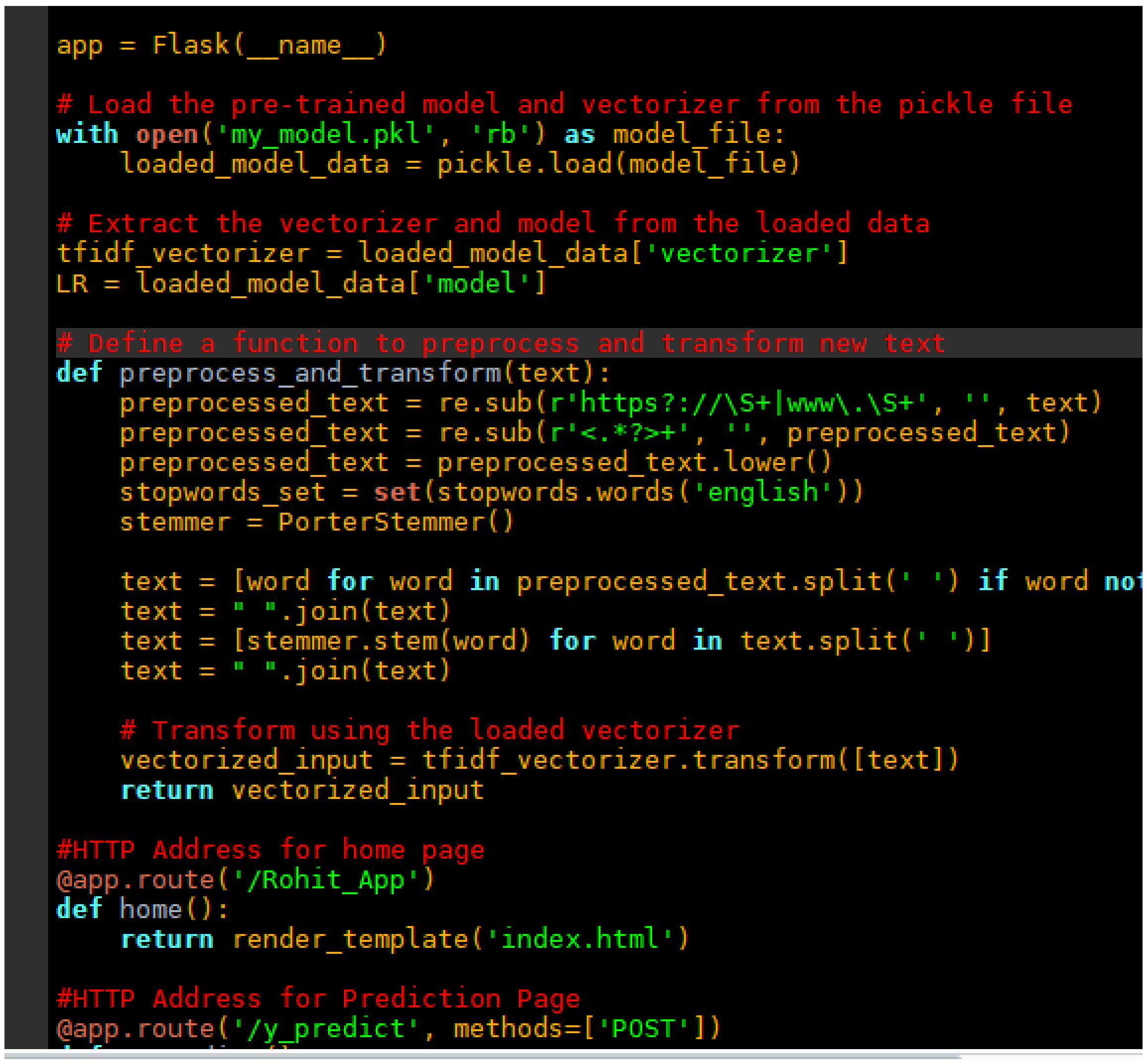
Activity 2.2: Build Python code:

Import the libraries



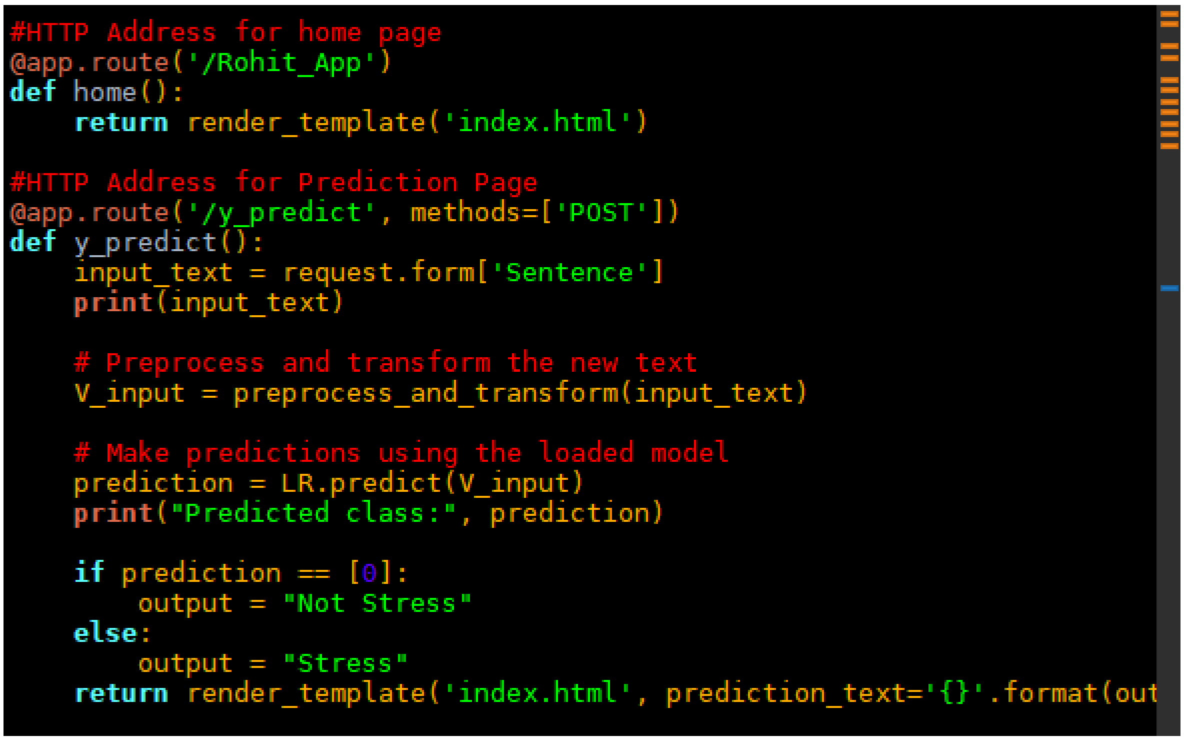
```python
import flask
import os
import pandas as pd
import numpy as np
import re
import string
import pickle
```

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (name) as argument.



```python
app = Flask(__name__)

# Load the pre-trained model and vectorizer from the pickle file
with open('my_model.pkl', 'rb') as model_file:
    loaded_model_data = pickle.load(model_file)

# Extract the vectorizer and model from the loaded data
tfidf_vectorizer = loaded_model_data['vectorizer']
LR = loaded_model_data['model']

# Define a function to preprocess and transform new text
def preprocess_and_transform(text):
    preprocessed_text = re.sub(r'https?://\S+|www\.\S+', '', text)
    preprocessed_text = re.sub(r'<.*?>+', '', preprocessed_text)
    preprocessed_text = preprocessed_text.lower()
    stopwords_set = set(stopwords.words('english'))
    stemmer = PorterStemmer()

    text = [word for word in preprocessed_text.split(' ') if word no
    text = " ".join(text)
    text = [stemmer.stem(word) for word in text.split(' ')]
    text = " ".join(text)

    # Transform using the loaded vectorizer
    vectorized_input = tfidf_vectorizer.transform([text])
    return vectorized_input

#HTTP Address for home page
@app.route('/Rohit_App')
def home():
    return render_template('index.html')

#HTTP Address for Prediction Page
@app.route('/y_predict', methods=['POST'])
```

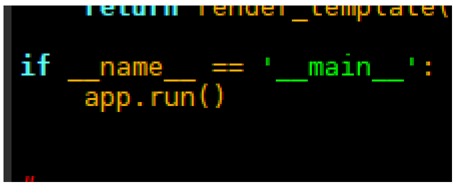
Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the



```python
#HTTP Address for home page
@app.route('/Rohit_App')
def home():
    return render_template('index.html')

#HTTP Address for Prediction Page
@app.route('/y_predict', methods=['POST'])
def y_predict():
    input_text = request.form['Sentence']
    print(input_text)

    # Preprocess and transform the new text
    V_input = preprocess_and_transform(input_text)

    # Make predictions using the loaded model
    prediction = LR.predict(V_input)
    print("Predicted class:", prediction)

    if prediction == [0]:
        output = "Not Stress"
    else:
        output = "Stress"
    return render_template('index.html', prediction_text='{}'.format(out
```

model.predict() function. This function returns the prediction. And this prediction value will be rendered to the text that we have mentioned in the Index.html page earlier.

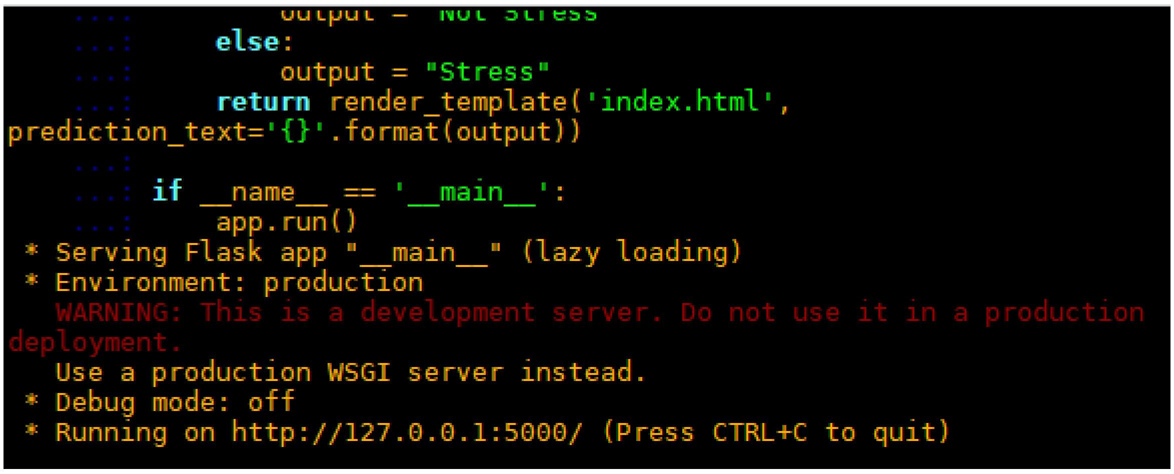
Main Function:



```python
    return render_template(

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

Activity 2.3: Run the web application

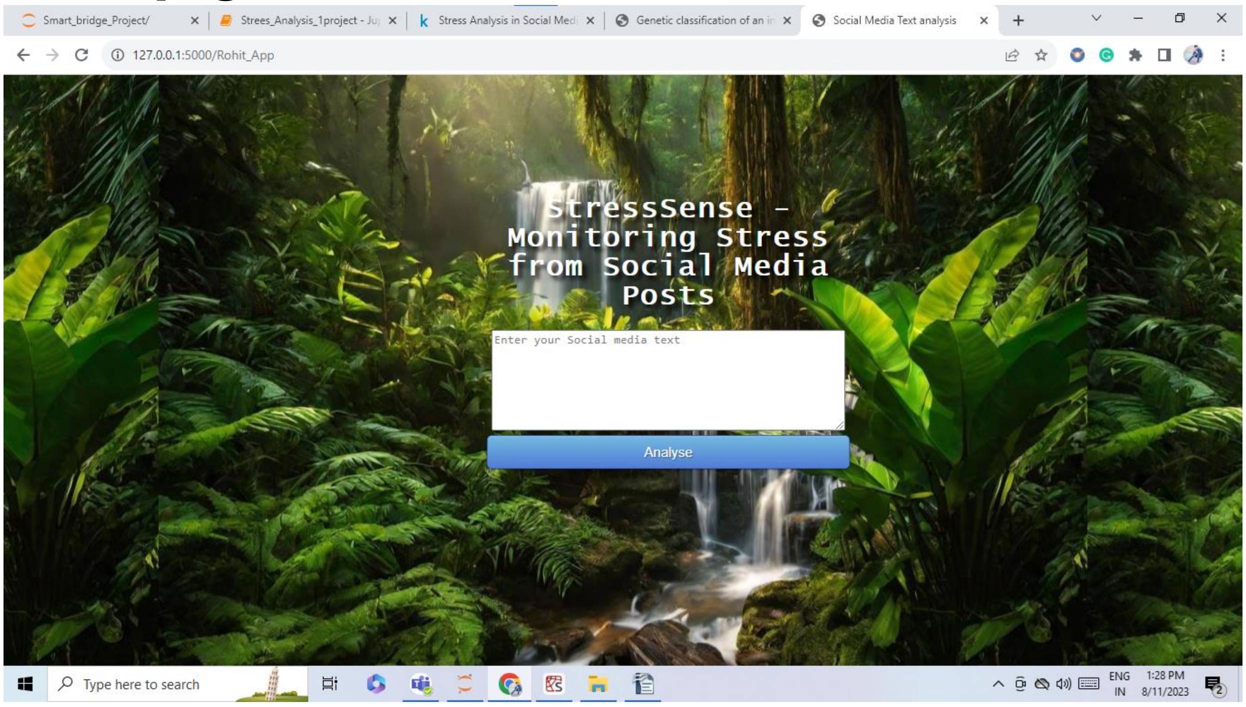
- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app.py” command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top left corner, enter the inputs, click on the submit button, and see the result/prediction on the web.



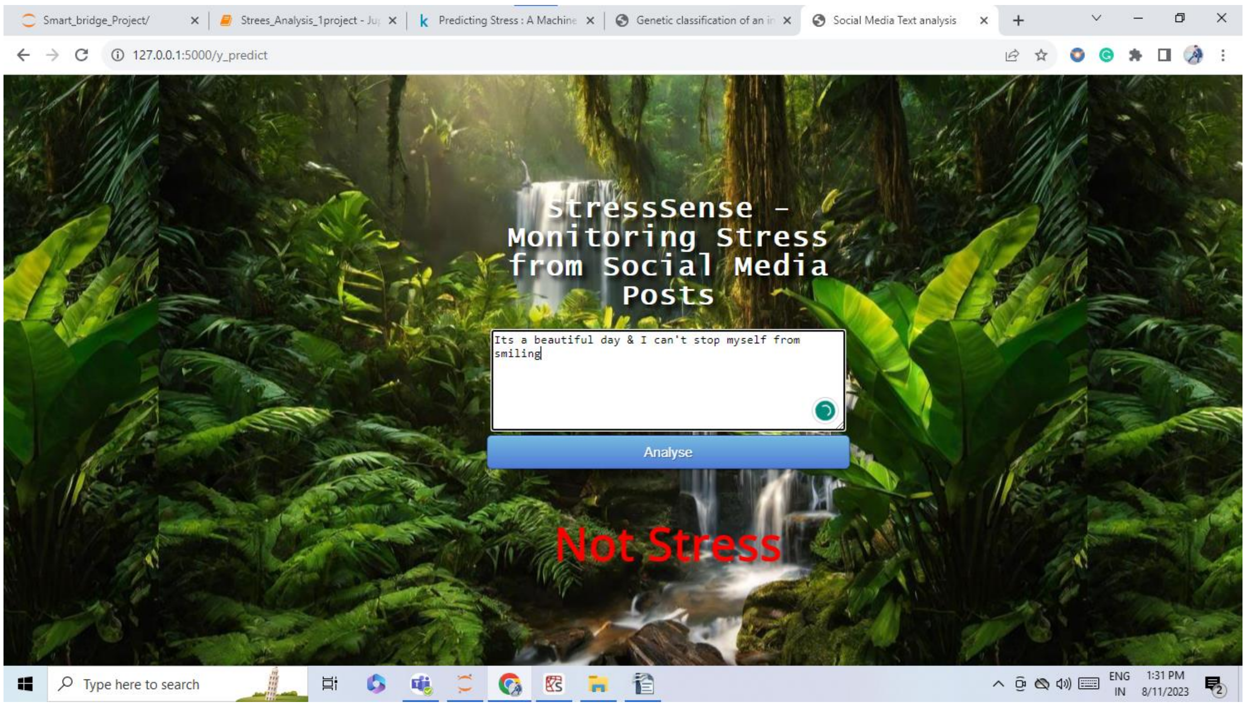
```
   ...:         output = "Not Stress"
   ...:     else:
   ...:         output = "Stress"
   ...:     return render_template('index.html',
prediction_text='{}'.format(output))
   ...:
   ...: if __name__ == '__main__':
   ...:     app.run()
 * Serving Flask app "__main__" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production
deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Now,Go the web browser and write the localhost url (http://127.0.0.1:5000) to get the index10.html

Home page:


Smart_bridge_Project/
Strees_Analysis_1project - Ju
Stress Analysis in Social Med
Genetic classification of an i
Social Media Text analysis
127.0.0.1:5000/Rohit_App
StressSense – Monitoring Stress from Social Media Posts
Enter your Social media text
Analyse
Type here to search
ENG IN
1:28 PM 8/11/2023

Prediction:


Smart_bridge_Project/
Strees_Analysis_1project - Ju
Predicting Stress : A Machine
Genetic classification of an in
Social Media Text analysis
127.0.0.1:5000/y_predict
Analyse
Type here to search
ENG
IN
1:31 PM
8/11/2023

StressSense - Monitoring Stress from Social Media Posts

Its a beautiful day & I can't stop myself from smiling

Not Stress