

MACHINE LEARNING PROJECT

LITERATURE SURVEY

SI.NO.	TITLE	AUTHOR NAME	PUBLISHED YEAR	REMARKS
1	Machine Learning and IoT for Prediction and Detection of Stress	Mr.Purnendu Shekhar Pandey	2017	The study focuses on using heart rate as a measure to identify stress in people and to forecast a person's state in order to let them know how their health is doing. Internet of Things (IoT) and machine learning are the methods the authors suggest adopting to achieve this. While ML is used to foretell a person's condition, IoT is utilised to inform that individual of their health status. In order to prevent an acute disease from developing, it is important for people to learn how to detect when they are in an unhealthy state and take steps to get better.
2	Automatic Stress Detection Using Wearable Sensors and Machine Learning: A Review	Shruti Gedam, Sanchita Paul	2020	The paper provides a review of existing research on the use of wearable sensors and machine learning for automatic stress detection. The authors survey the different types of physiological and behavioural signals that have been used for stress detection, including heart rate, skin conductance, and movement. They also go over the different machine learning algorithms and techniques used for stress detection, such as decision trees, support vector machines, and deep learning. The authors conclude by discussing the potential benefits and limitations of using wearable sensors and machine learning for stress detection, as well as future research directions.
3	Stress Detection with Machine Learning and Deep Learning using Multimodal Physiological Data	Pramod Bobade, Vani M.	2020	The paper discusses the use of machine learning and deep learning algorithms in the detection of stress. The authors train the algorithms and detect stress in individuals using multimodal physiological data such as heart rate, electrodermal activity, and respiration. The paper discusses the efficacy of various machine learning and deep learning methods and compares the results obtained by each. The authors conclude that using multimodal physiological data for stress detection is effective, and that deep learning algorithms easily surpass traditional machine learning methods.

4	A Decision Tree Optimised SVM Model for Stress Detection using Biosignals	Alana Paul Cruz, Aravind Pradeep, Kavali Riya Sivasankar and Krishnaveni K.S	2020	<p>The main topic of the research is the application of decision tree optimization to enhance the performance of support vector machine (SVM) models in biosignal-based stress detection. The authors suggest pre-processing the biosignal data with decision trees and then training the SVM model with the refined data. In comparison to standard SVM models, the performance of the optimised SVM model is assessed. The results show that in stress detection utilising biosignals, the proposed decision tree optimised SVM model outperforms traditional SVM models.</p>
5	Stress detection using deep neural networks	Russell Li and Zhandong Liu	2020	<p>The paper focuses on detecting stress using deep neural networks (DNNs). In order to train DNNs for stress detection, the scientists collect physiological and behavioural data from volunteers who complete stressful tasks. The effectiveness of the DNNs is assessed and contrasted with that of more established machine learning techniques. The findings demonstrate that DNNs perform better in stress detection than conventional machine learning algorithms. The authors also examine the DNNs' learnt features and discover that they can recognise intricate patterns in physiological and behavioural data that point to stress. The authors come to the conclusion that DNNs are useful for spotting stress and may help us understand its underlying causes better.</p>

SCREENSHOTS

The screenshot shows a Jupyter Notebook interface with the following components:

- Browser Tab:** Home Page - Select or create a notebook, Untitled - Jupyter Notebook.
- Address Bar:** localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
- Page Header:** Jupyter Untitled Last Checkpoint: an hour ago (autosaved) Logout
- Menu Bar:** File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)
- Code Cell:**

```
In [1]: import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.naive_bayes import BernoulliNB
from sklearn.feature_extraction.text import CountVectorizer

In [2]: df=pd.read_csv("C:/Users/SAKIRAN PETA/OneDrive/Desktop/stress.csv",encoding="latin-1")
df.head()
```
- Output Cell:** A table with 11 columns: subreddit, post_id, sentence_range, text, id, label, confidence, social_timestamp, social_karma, syntax_ari, lex_dal_min_pleasantness, le. The table contains 3 rows of data.
- System Tray:** 32°C Mostly sunny, Search, ENG IN, 15:52 30-01-2023.

The screenshot shows a Jupyter Notebook interface with the following components:

- Browser Tab:** Home Page - Select or create a notebook, Untitled - Jupyter Notebook.
- Address Bar:** localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
- Page Header:** Jupyter Untitled Last Checkpoint: an hour ago (autosaved) Logout
- Menu Bar:** File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)
- Code Cell:**

```
In [3]: df.describe()
```
- Output Cell:** A table with 11 columns: id, label, confidence, social_timestamp, social_karma, syntax_ari, lex_lw_wc, lex_lw_wc_Analytic, lex_lw_wc_Clout, lex_lw_wc_Authentic. The table contains 5 rows of data.
- System Tray:** 32°C Mostly sunny, Search, ENG IN, 15:52 30-01-2023.

Home Page - Select or create a notebook x x x
Untitled - Jupyter Notebook x +

localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3

jupyter Untitled Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

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 x 112 columns

In [4]: df.isnull()

Out[4]:

	subreddit	post_id	sentence_range	text	id	label	confidence	social_timestamp	social_karma	syntax_ari	...	lex_dal_min_pleasantness	lex_dal_mi
0	False	False	False	False	False	False	False	False	False	False	...	False	False
1	False	False	False	False	False	False	False	False	False	False	...	False	False
2	False	False	False	False	False	False	False	False	False	False	...	False	False
3	False	False	False	False	False	False	False	False	False	False	...	False	False
4	False	False	False	False	False	False	False	False	False	False	...	False	False
...
2833	False	False	False	False	False	False	False	False	False	False	...	False	False
2834	False	False	False	False	False	False	False	False	False	False	...	False	False

32°C Mostly sunny

Search

ENG IN 15:53 30-01-2023

Home Page - Select or create a notebook x x x
Untitled - Jupyter Notebook x +

localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3

jupyter Untitled Last Checkpoint: an hour ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

2835 False False False False False False False False False False False ... False False
2836 False False False False False False False False False False False ... False False
2837 False False False False False False False False False False False ... False False

2838 rows x 116 columns

In [5]: df.isnull().sum()

Out[5]:

```
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
```

In [8]: import nltk
import re
from nltk.corpus import stopwords
import string

32°C Mostly sunny

Search

ENG IN 15:54 30-01-2023

Home Page - Select or create a notebook x x x + x
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Python 3 (ipykernel)
File Edit View Insert Cell Kernel Widgets Help
Run Code

```
import string
nltk.download('stopwords')
stemmer=nltk.SnowballStemmer("english")
stopword=set (stopwords.words('english'))

def clean(text):
    text=str(text).lower()
    text=re.sub('[\.\?!\]', ' ',text)
    text=re.sub('https?://\S+/\S+\. \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)
    text=[word for word in text.split(' ') if word not in stopword]
    text=" ".join(text)
    text=[stemmer.stem(word) for word in text.split(' ')]
    text=" ".join(text)
    return text
df["text"]=df["text"].apply(clean)
```

[nltk_data] Downloading package stopwords to C:\Users\SAIKIRAN
[nltk_data] PETA\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!

32°C Mostly sunny 15:54 30-01-2023

Home Page - Select or create a notebook x x x + x
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
Python 3 (ipykernel)
File Edit View Insert Cell Kernel Widgets Help
Run Code

```
In [11]: from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split

x=np.array(df["text"])
y=np.array(df["label"])

cv=CountVectorizer()
X=cv.fit_transform(x)
print(X)
xtrain, xtest, ytrain, ytest =train_test_split(X,y,test_size=0.30,random_state=42)
```

```
(0, 7517) 1
(0, 3321) 1
(0, 9603) 1
(0, 872) 1
(0, 8486) 1
(0, 3802) 1
(0, 7323) 1
(0, 9054) 1
(0, 303) 1
(0, 9912) 1
(0, 4366) 1
(0, 5109) 1
(0, 5408) 1
(0, 2221) 1
(0, 5196) 1
(0, 3308) 1
(0, 2630) 3
```

32°C Mostly sunny 15:54 30-01-2023

Home Page - Select or create a notebook x x + x
Untitled - Jupyter Notebook x + x
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run Code
(0, 2630) 3
(0, 4249) 1
(0, 5399) 1
(0, 3748) 1
(0, 8466) 1
(0, 6968) 1
(0, 4211) 1
(0, 5253) 1
(0, 1858) 1
:
(2836, 889) 1
(2836, 4620) 1
(2836, 2967) 1
(2836, 4680) 1
(2836, 4856) 1
(2836, 4576) 1
(2837, 7517) 2
(2837, 3057) 1
(2837, 5619) 2
(2837, 8926) 1
(2837, 8632) 1
(2837, 6876) 1
(2837, 4381) 1
(2837, 9828) 1
(2837, 5657) 1
(2837, 9024) 1
(2837, 5805) 1
(2837, 2623) 1
(2837, 7580) 1
32°C Mostly sunny
Search
ENG IN
15:54 30-01-2023

Home Page - Select or create a notebook x x + x
Untitled - Jupyter Notebook x + x
localhost:8888/notebooks/Untitled.ipynb?kernel_name=python3
jupyter Untitled Last Checkpoint: an hour ago (autosaved)
File Edit View Insert Cell Kernel Widgets Help
Run Code
(2837, 7580) 1
(2837, 2385) 1
(2837, 7926) 1
(2837, 2796) 1
(2837, 9023) 1
(2837, 5544) 1
(2837, 3059) 1
In [13]: from sklearn.naive_bayes import BernoulliNB
model=BernoulliNB()
model.fit(xtrain,ytrain)
Out[13]: BernoulliNB()
In [15]: user=input("enter the text")
data=cv.transform([user]).toarray()
output=model.predict(data)
print(output)
enter the textsravanthi is happy
[0]
32°C Mostly sunny
Search
ENG IN
15:54 30-01-2023

CODE

```
[{"metadata":{"trusted":true},"cell_type":"code","source":"import numpy as
np\nimport pandas as pd\nimport seaborn as sns\nfrom matplotlib import pyplot
as plt\nfrom sklearn.naive_bayes import BernoulliNB\nfrom
sklearn.feature_extraction.text import
CountVectorizer","execution_count":1,"outputs":[{}],"metadata":{"trusted":tru
e},"cell_type":"code","source":"df=pd.read_csv(\"C:/Users/SAIKIRAN
PETA/OneDrive/Desktop/stress.csv\",encoding=\"latin-
1\")\nndf.head()","execution_count":2,"outputs":[{"output_type":"execute_resul
t","execution_count":2,"data":{"text/plain":
sentence_range \\n0          PTSD 8601tu          (15, 20) \\n1
assistance 8lbrx9          (0, 5) \\n2          PTSD 9chlzh          (15,
20) \\n3          relationships 7rorpp          [5, 10] \\n4          survivorssofabuse
9p2gbc          [0, 5] \\n\\n
text      id label \\n0 He said he had not felt that way before, sugge...
33181      1 \\n1 Hey there r/assistance, Not sure if this is th... 2606
0 \\n2 My mom then hit me with the newspaper and it s... 38816      1
\\n3 until i met my new boyfriend, he is amazing, h... 239      1 \\n4
October is Domestic Violence Awareness Month a... 1421      1 \\n\\n
confidence social_timestamp social_karma syntax_ari ... \\n0
0.8          1521614353          5          1.806818 ... \\n1          1.0
1527009817          4          9.429737 ... \\n2          0.8          1535935605
2          7.769821 ... \\n3          0.6          1516429555          0
2.667798 ... \\n4          0.8          1539809005          24          7.554238
... \\n\\n lex_dal_min_pleasantness lex_dal_min_activation
lex_dal_min_imagery \\n0          1.000          1.1250
1.0 \\n1          1.125          1.0000
1.0 \\n2          1.000          1.1429
1.0 \\n3          1.000          1.1250
1.0 \\n4          1.000          1.1250
1.0 \\n\\n lex_dal_avg_activation lex_dal_avg_imagery
lex_dal_avg_pleasantness \\n0          1.77000          1.52211
1.89556 \\n1          1.69586          1.62045
1.88919 \\n2          1.83088          1.58108
1.85828 \\n3          1.75356          1.52114
1.98848 \\n4          1.77644          1.64872
1.81456 \\n\\n social_upvote_ratio social_num_comments syntax_fk_grade
sentiment \\n0          0.86          1          3.253573 -
0.002742 \\n1          0.65          2          8.828316
0.292857 \\n2          0.67          0          7.841667
0.011894 \\n3          0.50          5          4.104027
0.141671 \\n4          1.00          1          7.910952 -
0.204167 \\n\\n[5 rows x 116 columns]","text/html":
\\n\\n \\n \\n subreddit\\n post_id\\n sentence_range\\n
text\\n id\\n label\\n confidence\\n social_timestamp\\n
social_karma\\n syntax_ari\\n ...\\n lex_dal_min_pleasantness\\n
lex_dal_min_activation\\n lex_dal_min_imagery\\n
lex_dal_avg_activation\\n lex_dal_avg_imagery\\n
lex_dal_avg_pleasantness\\n social_upvote_ratio\\n
social_num_comments\\n syntax_fk_grade\\n sentiment\\n \\n \\n \\n
\\n 0\\n PTSD\\n 8601tu\\n (15, 20)\\n He said he had not
felt that way before, sugge...\\n 33181\\n 1\\n 0.8\\n
1521614353\\n 5\\n 1.806818\\n ...\\n 1.000\\n 1.1250\\n
1.0\\n 1.77000\\n 1.52211\\n 1.89556\\n 0.86\\n 1\\n
3.253573\\n -0.002742\\n \\n \\n 1\\n assistance\\n
```

```

8lbrx9\n      (0, 5)\n      Hey there r/assistance, Not sure if this is
th...\n      2606\n      0\n      1.0\n      1527009817\n      4\n
9.429737\n      ...\n      1.125\n      1.0000\n      1.0\n      1.69586\n
1.62045\n      1.88919\n      0.65\n      2\n      8.828316\n      0.292857\n
\n      \n      2\n      ptsd\n      9chlzh\n      (15, 20)\n      My mom then
hit me with the newspaper and it s...\n      38816\n      1\n      0.8\n
1535935605\n      2\n      7.769821\n      ...\n      1.000\n      1.1429\n
1.0\n      1.83088\n      1.58108\n      1.85828\n      0.67\n      0\n
7.841667\n      0.011894\n      \n      \n      3\n      relationships\n
7rorpp\n      [5, 10]\n      until i met my new boyfriend, he is amazing,
h...\n      239\n      1\n      0.6\n      1516429555\n      0\n
2.667798\n      ...\n      1.000\n      1.1250\n      1.0\n      1.75356\n
1.52114\n      1.98848\n      0.50\n      5\n      4.104027\n      0.141671\n
\n      \n      4\n      survivorsofabuse\n      9p2gbc\n      [0, 5]\n
October is Domestic Violence Awareness Month a...\n      1421\n      1\n
0.8\n      1539809005\n      24\n      7.554238\n      ...\n      1.000\n
1.1250\n      1.0\n      1.77644\n      1.64872\n      1.81456\n      1.00\n
1\n      7.910952\n      -0.204167\n      \n      \n\n

```

5 rows × 116 columns

```

\n
{"", "metadata": {}}, {"metadata": {"trusted": true}, "cell_type": "code", "source":
"df.describe()", "execution_count": 3, "outputs": [{"output_type": "execute_result",
"execution_count": 3, "data": {"text/plain": "
id label
confidence social_timestamp social_karma \\n\ncount 2838.000000
2838.000000 2838.000000 2.838000e+03 2838.000000 \nmean
13751.999295 0.524313 0.808972 1.518107e+09 18.262156
\nstd 17340.161897 0.499497 0.177038 1.552209e+07
79.419166 \nmin 4.000000 0.000000 0.428571 1.483274e+09
0.000000 \n25% 926.250000 0.000000 0.600000 1.509698e+09
2.000000 \n50% 1891.500000 1.000000 0.800000 1.517066e+09
5.000000 \n75% 25473.750000 1.000000 1.000000 1.530898e+09
10.000000 \nmax 55757.000000 1.000000 1.000000 1.542592e+09
1435.000000 \n\n syntax_ari lex_liwc_WC lex_liwc_Analytic
lex_liwc_Clout \\n\ncount 2838.000000 2838.000000 2838.000000
2838.000000 \nmean 4.684272 85.996124 35.240941
40.948231 \nstd 3.316435 32.334887 26.486189
31.587117 \nmin -6.620000 5.000000 1.000000
1.000000 \n25% 2.464243 65.000000 12.410000
12.135000 \n50% 4.321886 81.000000 29.420000
33.520000 \n75% 6.505657 101.000000 55.057500
69.320000 \nmax 24.074231 310.000000 99.000000
99.000000 \n\n lex_liwc_Authentic ... lex_dal_min_pleasantness
\\n\ncount 2838.000000 ... 2838.000000 \nmean
67.044249 ... 1.088001 \nstd 32.880644 ...
0.117159 \nmin 1.000000 ... 1.000000 \n25%
41.070000 ... 1.000000 \n50% 80.710000 ...
1.000000 \n75% 96.180000 ... 1.142900 \nmax
99.000000 ... 1.900000 \n\n lex_dal_min_activation
lex_dal_min_imagery lex_dal_avg_activation \\n\ncount
2838.000000 2838.000000 2838.000000 \nmean
1.120099 1.000211 1.722759 \nstd
0.085227 0.006500 0.047835 \nmin
1.000000 1.000000 1.485400 \n25%
1.000000 1.000000 1.691430 \n50%
1.142900 1.000000 1.721430 \n75%
1.142900 1.000000 1.751760 \nmax

```



```

1.500000          1.200000          2.007400  \n\n
lex_dal_avg_imagery lex_dal_avg_pleasantness social_upvote_ratio  \\\ncount
2838.000000          2838.000000          2838.000000  \nmean
1.536400          1.879385          0.843517  \nstd
0.102971          0.058932          0.174794  \nmin
1.200000          1.561150          0.140000  \n25%
1.469745          1.841782          0.750000  \n50%
1.530295          1.878250          0.890000  \n75%
1.596030          1.916243          1.000000  \nmax
2.066670          2.158490          1.000000  \n\n
social_num_comments syntax_fk_grade sentiment  \ncount
2838.000000          2838.000000  2838.000000  \nmean          9.948555
5.448836          0.040740  \nstd          21.798032          2.535829
0.195490  \nmin          0.000000          -1.918000  -1.000000  \n25%
2.000000          3.729973  -0.072222  \n50%          5.000000
5.210000          0.044821  \n75%          10.000000          6.855217
0.166667  \nmax          416.000000          21.198919          1.000000  \n\n[8
rows x 112 columns]", "text/html": "
\n\n  \n  \n  id\n  label\n  confidence\n
social_timestamp\n  social_karma\n  syntax_ari\n  lex_liwc_WC\n
lex_liwc_Analytic\n  lex_liwc_Clout\n  lex_liwc_Authentic\n
...\n  lex_dal_min_pleasantness\n  lex_dal_min_activation\n
lex_dal_min_imagery\n  lex_dal_avg_activation\n
lex_dal_avg_imagery\n  lex_dal_avg_pleasantness\n
social_upvote_ratio\n  social_num_comments\n  syntax_fk_grade\n
sentiment\n  \n  \n  \n  \n  count\n  2838.000000\n
2838.000000\n  2838.000000\n  2.838000e+03\n  2838.000000\n
2838.000000\n  2838.000000\n  2838.000000\n  2838.000000\n
2838.000000\n  ...\n  2838.000000\n  2838.000000\n
2838.000000\n  2838.000000\n  2838.000000\n  2838.000000\n
2838.000000\n  2838.000000\n  2838.000000\n  2838.000000\n  \n
\n  mean\n  13751.999295\n  0.524313\n  0.808972\n
1.518107e+09\n  18.262156\n  4.684272\n  85.996124\n
35.240941\n  40.948231\n  67.044249\n  ...\n  1.088001\n
1.120099\n  1.000211\n  1.722759\n  1.536400\n  1.879385\n
0.843517\n  9.948555\n  5.448836\n  0.040740\n  \n  \n
std\n  17340.161897\n  0.499497\n  0.177038\n  1.552209e+07\n
79.419166\n  3.316435\n  32.334887\n  26.486189\n
31.587117\n  32.880644\n  ...\n  0.117159\n  0.085227\n
0.006500\n  0.047835\n  0.102971\n  0.058932\n  0.174794\n
21.798032\n  2.535829\n  0.195490\n  \n  \n  min\n
4.000000\n  0.000000\n  0.428571\n  1.483274e+09\n
0.000000\n  -6.620000\n  5.000000\n  1.000000\n  1.000000\n
1.000000\n  ...\n  1.000000\n  1.000000\n  1.000000\n
1.485400\n  1.200000\n  1.561150\n  0.140000\n  0.000000\n
-1.918000\n  -1.000000\n  \n  \n  25%\n  926.250000\n
0.000000\n  0.600000\n  1.509698e+09\n  2.000000\n
2.464243\n  65.000000\n  12.410000\n  12.135000\n
41.070000\n  ...\n  1.000000\n  1.000000\n  1.000000\n
1.691430\n  1.469745\n  1.841782\n  0.750000\n  2.000000\n
3.729973\n  -0.072222\n  \n  \n  50%\n  1891.500000\n
1.000000\n  0.800000\n  1.517066e+09\n  5.000000\n
4.321886\n  81.000000\n  29.420000\n  33.520000\n
80.710000\n  ...\n  1.000000\n  1.142900\n  1.000000\n
1.721430\n  1.530295\n  1.878250\n  0.890000\n  5.000000\n
5.210000\n  0.044821\n  \n  \n  75%\n  25473.750000\n
1.000000\n  1.000000\n  1.530898e+09\n  10.000000\n
6.505657\n  101.000000\n  55.057500\n  69.320000\n

```

```

96.180000\n      ... \n      1.142900\n      1.142900\n      1.000000\n1.751760\n      1.596030\n      1.916243\n      1.000000\n      10.000000\n6.855217\n      0.166667\n      \n      \n      max\n      55757.000000\n1.000000\n      1.000000\n      1.542592e+09\n      1435.000000\n24.074231\n      310.000000\n      99.000000\n      99.000000\n99.000000\n      ... \n      1.900000\n      1.500000\n      1.200000\n2.007400\n      2.066670\n      2.158490\n      1.000000\n      416.000000\n21.198919\n      1.000000\n      \n      \n

```

8 rows × 112 columns

```

\n
{"metadata":{}}},{ "metadata":{"trusted":true},"cell_type":"code","source":
"df.isnull()","execution_count":4,"outputs":[{"output_type":"execute_result",
"execution_count":4,"data":{"text/plain": "      subreddit  post_id
sentence_range  text      id  label  confidence  \\\n0      False      False
False  False  False  False      False  \n1      False      False
False  False  False  \n2      False      False
False  False  False  \n3      False      False
False  False  False  \n4      False      False
False  False  False  \n...      ...      ...
...      ...      ...      ...      \n2833      False      False
False  False  False  False      False  \n2834      False      False
False  False  False  \n2835      False      False
False  False  False  \n2836      False      False
False  False  False  \n2837      False      False
False  False  False  \n\n      social_timestamp
social_karma  syntax_ari  ...  \\\n0      False      False
False  ...  \n1      False      False      False  \n2
False  False      False      False  \n3      False      False
False  ...  \n4      False      False      False  \n...
...      ...      ...      \n2833      False      False
False  ...  \n2834      False      False      False  \n2835
False  False      False      False  \n2836      False      False
False  ...  \n2837      False      False      False  \n\n
lex_dal_min_pleasantness  lex_dal_min_activation  lex_dal_min_imagery  \\\n0
False      False      False  \n1
False      False      False  \n2
False      False      False  \n3
False      False      False  \n4
False      False      False  \n...
...      ...      ...      \n2833
False      False      False  \n2834
False      False      False  \n2835
False      False      False  \n2836
False      False      False  \n2837
False      False      False  \n\n
lex_dal_avg_activation  lex_dal_avg_imagery  lex_dal_avg_pleasantness  \\\n0
False      False      False  \n1
False      False      False  \n2
False      False      False  \n3
False      False      False  \n4
False      False      False  \n...
...      ...      ...      \n2833
False      False      False  \n2834
False      False      False  \n2835
False      False      False  \n2836
False      False      False  \n2837
False      False      False

```

[illegible]

```
False\n      False\n      False\n      False\n      \n      \n      2837\nFalse\n      False\n      False\n      False\n      False\n      False\nFalse\n      False\n      False\n      False\n      ... \n      False\nFalse\n      False\n      False\n      False\n      False\n      False\nFalse\n      False\n      False\n      \n      \n
```

2838 rows × 116 columns

```
\n
", "metadata": {} }], {"metadata": {"trusted": true, "cell_type": "code", "source":
"df.isnull().sum()", "execution_count": 5, "outputs": [{"output_type": "execute_re
sult", "execution_count": 5, "data": {"text/plain": "subreddit
0\npost_id                                0\nsentence_range                0\nntext
0\nnid                                    0\n
..\nlex_dal_avg_pleasantness              0\nsocial_upvote_ratio
0\nsocial_num_comments                    0\nsyntax_fk_grade                0\nsentiment
0\nLength: 116, dtype:
int64"}], {"metadata": {"trusted": true, "cell_type": "code", "sou
rce": "import nltk\nimport re\nfrom nltk.corpus import stopwords\nimport
string\nnltk.download('stopwords')\nstemmer=nltk.SnowballStemmer('english')
\nstopword=set (stopwords.words('english'))\n\nndef clean(text):\n
text=str(text).lower()\n      text=re.sub('[\'.\']*\\', ' ', text)\n
text=re.sub('https?://\\S+/www\\. \\S+', ' ', text)\n      text=re.sub('+', '
', text)\n      text=re.sub('[%s]' %re.escape(string.punctuation), ' ', text)\n
text=re.sub(' \\n', ' ', text)\n      text=re.sub(' \\w*\\d\\w*', ' ', text)\n
text=[word for word in text.split(' ') if word not in stopword] \n      text=\\
".join(text)\n      text=[stemmer.stem(word) for word in text.split(' ')]\n
text=\\ " ".join(text)\n      return
text\ndf[["text"]]=df[["text"]].apply(clean)\n\n\n\n
\n
", "execution_count": 8, "outputs": [{"output_type": "stream", "text": "[nltk_data]
Downloading package stopwords to C:\\Users\\SAIKIRAN\\n[nltk_data]
PETA\\AppData\\Roaming\\nltk_data...\\n[nltk_data] Package stopwords is
already up-to-
date!\n", "name": "stderr"} ]], {"metadata": {"trusted": true, "cell_type": "code", "
source": "from sklearn.feature_extraction.text import CountVectorizer\nfrom
sklearn.model_selection import
train_test_split\n\nx=np.array(df[\"text\"])\nny=np.array(df[\"label\"])\n\nncv
=CountVectorizer()\nX=cv.fit_transform(x)\n\nprint(X)\n\nxtrain, xtest, ytrain,
ytest
=train_test_split(X,y,test_size=0.30,random_state=42)", "execution_count": 11, "
outputs": [{"output_type": "stream", "text": " (0, 7517)\t1\n (0, 3321)\t1\n
(0, 9603)\t1\n (0, 872)\t1\n (0, 8486)\t1\n (0, 3802)\t1\n (0, 7323)\t1\n
(0, 9054)\t1\n (0, 303)\t1\n (0, 9912)\t1\n (0, 4366)\t1\n (0, 5109)\t1\n
(0, 5408)\t1\n (0, 2221)\t1\n (0, 5196)\t1\n (0, 3308)\t1\n (0,
2630)\t3\n (0, 4249)\t1\n (0, 5399)\t1\n (0, 3748)\t1\n (0, 8466)\t1\n
(0, 6968)\t1\n (0, 4211)\t1\n (0, 5253)\t1\n (0, 1858)\t1\n :t:\n
(2836, 889)\t1\n (2836, 4620)\t1\n (2836, 2967)\t1\n (2836, 4680)\t1\n
(2836, 4856)\t1\n (2836, 4576)\t1\n (2837, 7517)\t2\n (2837, 3057)\t1\n
(2837, 5619)\t2\n (2837, 8926)\t1\n (2837, 8632)\t1\n (2837, 6876)\t1\n
(2837, 4381)\t1\n (2837, 9828)\t1\n (2837, 5657)\t1\n (2837, 9024)\t1\n
(2837, 5805)\t1\n (2837, 2623)\t1\n (2837, 7580)\t1\n (2837, 2385)\t1\n
(2837, 7926)\t1\n (2837, 2796)\t1\n (2837, 9023)\t1\n (2837, 5544)\t1\n
(2837,
3059)\t1\n", "name": "stdout"} ]], {"metadata": {"trusted": true, "cell_type": "code
", "source": "from sklearn.naive_bayes import
BernoulliNB\n\nmodel=BernoulliNB()\nmodel.fit(xtrain,ytrain)", "execution_count"
: 13, "outputs": [{"output_type": "execute_result", "execution_count": 13, "data": {"
```

```
text/plain":"BernoulliNB()"},"metadata":{}}]],{"metadata":{"trusted":true},"cell_type":"code","source":"user=input(\"enter the text\")\ndata=cv.transform([user]).toarray()\noutput=model.predict(data)\nprint(output)","execution_count":15,"outputs":[{"output_type":"stream","name":"stdout","text":"enter the textsravanthi is happy\n[0]\n"}]}}
```

THE END