Stress Detection with Machine Learning

Stress, anxiety, and depression are threatening the mental health of people. Every person has a reason for having a stressful life. People often share their feelings on social media platforms like on Instagram in the form of posts and stories, and on Reddit in the form of asking for suggestions about their life on subreddits. In the past few years, many content creators have come forward to create content to help people with their mental health. Many organizations can use stress detection to find which social media users are stressed to help them quickly. So if you want to learn how to use machine learning to detect stress on social media posts, this article is for you. In this article, I will take you through the task of stress detection with machine learning using Python.

Stress Detection with Machine Learning

Stress detection is a challenging task, as there are so many words that can be used by people on their posts that can show whether a person is having psychological stress or not. While looking for datasets that I can use to train a machine learning model for stress detection, I found a dataset on Kaggle with 116 columns. We only need to use the text and label column for this task.

The dataset I am using for this task contains data posted on subreddits related to mental health. This dataset contains various mental health problems shared by people about their life. Fortunately, this dataset is labelled as 0 and 1, where 0 indicates no stress and 1 indicates stress. So in the section below, I will take you through the task of stress detection in social media posts using Python.

Stress Detection using Python

Now let's start the task of stress detection with machine learning. I will start this task by importing the necessary Python libraries and the **dataset** that we need for this task:

```
import pandas as pd

import numpy as np

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

print(data.head())
```

```
          subreddit post_id  ... syntax_fk_grade sentiment
0              ptsd  8601tu  ...        3.253573 -0.002742
1        assistance  81brx9  ...        8.828316  0.292857
2              ptsd  9ch1zh  ...        7.841667  0.011894
3     relationships  7rorpp  ...        4.104027  0.141671
4  survivorsofabuse  9p2gbc  ...        7.910952 -0.204167

[5 rows x 116 columns]
```

Let's have a look at whether this dataset contains any null values or not:

print(data.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
```

So this dataset does not have any null values. Now let's prepare the text column of this dataset to clean the text column with stopwords, links, special symbols and language errors:

import nltk

import re

nltk.download('stopwords')

stemmer = nltk.SnowballStemmer("english")

from nltk.corpus import stopwords

import string

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

```
def clean(text):
    text = str(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)
    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
  data["text"] = data["text"].apply(clean)
```

Now let's have a look at the most used words by the people sharing about their life problems on social media by visualizing a **word cloud** of the text column:

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator
text = " ".join(i for i in data.text)
stopwords = set(STOPWORDS)
wordcloud = WordCloud(stopwords=stopwords,
                      background_color="white").generate(text)
plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.show()
```

Stress Detection Model

The label column in this dataset contains labels as 0 and 1. 0 means no stress, and 1 means stress. I will use Stress and No stress labels instead of 1 and 0. So let's prepare this column accordingly and select the text and label columns for the process of training a machine learning model:

```
data["label"] = data["label"].map({0: "No Stress", 1: "Stress"})

data = data[["text", "label"]]

print(data.head())
```

```
                                                text       label
0  said felt way sugget go rest trigger ahead you...      Stress
1  hey rassist sure right place post goe  im curr...   No Stress
2  mom hit newspap shock would know dont like pla...      Stress
3  met new boyfriend amaz kind sweet good student...      Stress
4  octob domest violenc awar month domest violenc...      Stress
```

Now I will split this dataset into training and test sets:

```
from sklearn.feature_extraction.text import CountVectorizer

from sklearn.model_selection import train_test_split
```

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

cv = CountVectorizer()
X = cv.fit_transform(x)
xtrain, xtest, ytrain, ytest = train_test_split(X, y,
                                test_size=0.33,
                                random_state=42)
```

As this task is based on the problem of binary classification, I will be using the Bernoulli Naive Bayes algorithm, which is one of the best algorithms for binary classification problems. So let's train the stress detection model:

```
from sklearn.naive_bayes import BernoulliNB
model = BernoulliNB()
model.fit(xtrain, ytrain)
```

Now let's test the performance of our model on some random sentences based on mental health:

```
user = input("Enter a Text: ")
data = cv.transform([user]).toarray()
output = model.predict(data)
print(output)
```

```
Enter a Text: People need to take care of their mental health
['No Stress']
```

```
user = input("Enter a Text: ")
data = cv.transform([user]).toarray()
output = model.predict(data)
print(output)
```

```
Enter a Text: Sometime I feel like I need some help
['Stress']
```

So as you can see, we can see good results from our machine learning model. This is how you can train a stress detection model to detect stress from social media posts. This machine learning model can be improved by feeding it with more data.

Summary

So this is how you can train a machine learning model to detect stress from social media posts. People often share their feelings on social media platforms. Many organizations can use stress detection to find which social media users are stressed to help them quickly. I hope you liked this article on stress detection with machine learning using Python. Feel free to ask your valuable questions in the comments section below.