ML and NLP PBL 2023

DEPRESSION DETECTION ANALYSIS IN TWEETS USING NAÏVE BAYES

Name	Enrollment	Batch
Aditi Mahabole	20103023	B1
Molshree Sharma	20103060	B2
Arushi Sharma	20103010	B1

Submitted to: Ankit Vidhyarthi Sir.

Problem Statement:

- In our project, we're working on a special task, finding out if people are feeling sad or dealing with depression based on what they say in their tweets.
- We're using machine learning (ML) and natural language processing (NLP), to teach our system to understand the language in tweets.
- The goal is to detect and analyse a large number of tweets and tell us if there's a possibility that someone might be going through a tough time.
- This could be really helpful for supporting people online and connecting them with the right resources when they need it most.

Model:

Naïve Bayes

Naive Bayes is a classification algorithm based on Bayes' theorem. It is considered "naive" because it makes an assumption that the features used to describe an observation are conditionally independent, given the class label.

Bayes' Theorem

The Naive Bayes algorithm relies on Bayes' theorem, which is a mathematical formula that calculates the probability of an event based on prior knowledge of conditions that might be related to the event. The formula is as follows:

$$P(A|B) = P(B|A) \times P(A) / P(B)$$

- P(A|B) is the probability of event A occurring given that event B has occurred.
- P(B|A) is the probability of event B occurring given that event A has occurred.
- P(A) is the prior probability of event A.
- P(B) is the prior probability of event B.

Naive Bayes Classification

The Naive Bayes algorithm calculates the probability of each class given the observation and assigns the class with the highest probability to the observation.

$$P\,(C_i \mid X) = P\,(X \mid C_i) \times P\,(C_i) / P(X)$$

- **<u>Prior Probability P(Ci):</u>** This is the probability of **observing class Ci** without considering any features. It represents our belief about the likelihood of each class before observing the data.

- **<u>Likelihood P (X | Ci):</u>** This term represents the probability of observing the **features X** given that the **class is Ci**. The likelihood is often estimated from the training data.

- **<u>Evidence P(X):</u>** This term is the probability of observing the features X across all possible classes. It acts as a normalizing factor and ensures that the sum of posterior probabilities over all classes is equal to 1.

- **<u>Posterior Probability P (Ci |X):</u>** This is the probability of class Ci given the observed features X. It's what we're trying to calculate.

- **<u>Decision Rule:</u>** In practical terms, the decision rule for classification often involves selecting the class with the highest posterior probability

$$\mathbf{argmax}_{C_i}\{P(C_i)\times P(X \mid C_i)\}$$

$$\hat{P}(c_j) = \frac{doccount(C = c_j)}{N_{doc}}$$

$$\hat{P}(w_i \mid c_j) = \frac{count(w_i, c_j)}{\sum_{w \in V} count(w, c_j)}$$

fraction of times word w_i appears among all words in documents of topic c_j

➢ *Laplace smoothing*

Laplace smoothing is a way to prevent these zero probabilities. It involves adding a small constant (usually 1) to the count of each word in each class. This "smoothing" ensures that no word has a zero probability of occurrence in any class, even if it didn't appear in the training data.

$$\hat{P}(w_i \mid c) = \frac{count(w_i, c) + 1}{\sum_{w \in V} \left(count(w, c) + 1\right)} = \frac{count(w_i, c) + 1}{\left(\sum_{w \in V} count(w, c)\right) + |V|}$$

➢ Process

Data Collection

We gathered tweets from Kaggle that were classified as showing moderate, severe, or no depression. Divided dataset into train and test and saved into csv files.

Data Cleaning

We used stopwords from NLTK library to remove useless words and converted words to lowercase, removed special characters using regular expression library.

Data Preprocessing

Used Lemmatization to convert words to meaningful base form

Fit Data using Naïve Bayes Classifier

Build Naïve Bayes classifier from scratch, used dictionaries to store Likelihood of each word with respect to each class.

Predict Test Data using Naïve Bayes Classifier

Detected tweets in Test dataset as moderate, sever or no depression

Data Visualization

Used Matplotlib to show percentage of each class in test data set.

➢ Tech Stack

✓ Python

Pandas Library : for making dataframe

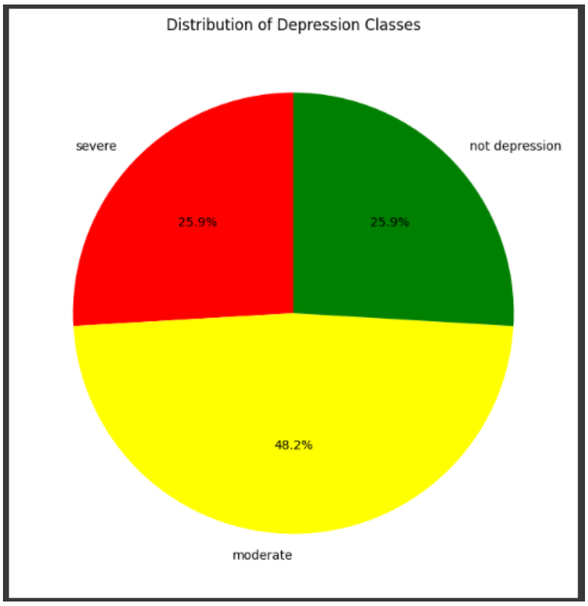
NLTK (Natural Language ToolKit) Library : lemmatization , removing stop words

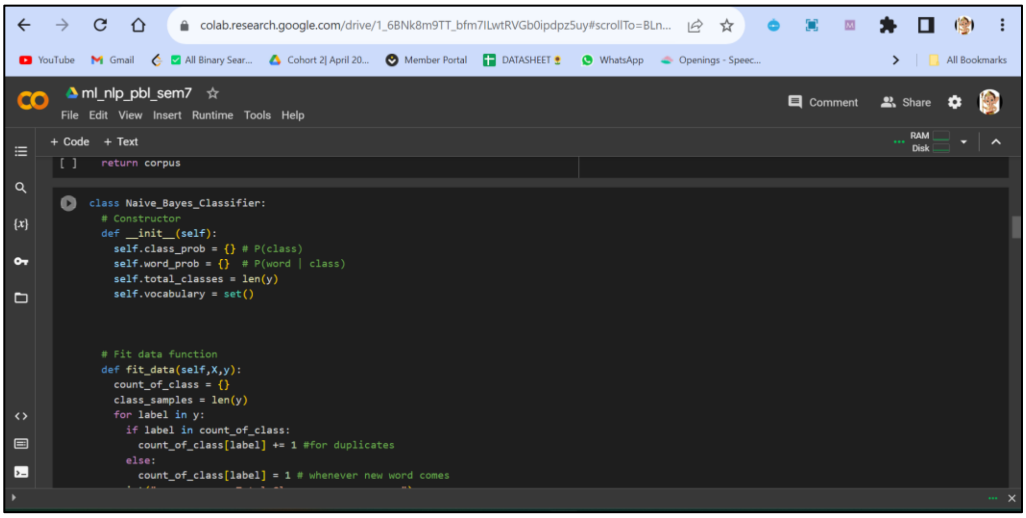
Regular Expression Library : removing special characters

Matplotlib Library : showing pie chart for percentage of people present in each class

➢ Results

48.2% were in moderate depression, 25.9% were in severe depression and remaining were in no depression.





➢ Link: https://github.com/aditimahabole/ML_NLP_Depression_Prediction_Tweets

➢ Refrences

1.https://www.analyticsvidhya.com/blog/2022/03/building-naive-bayes-classifier-from-scratch-to-perform-sentiment-analysis/

2.Class teacher Notes