**IDS594-PROJECT:**

SARCASM DETECTION IN NEWS HEADLINES

**Team Members:**

# LIPSA MISHRA (668886120)

# JIVISHA (663685107)

# SHALINI SINGH (663877768)

# BHOMIK PATHAK (676349569)

# **CONTENTS:**

# **INTRODUCTION PROBLEM STATEMENT**

# **DATASET STATISTICS**

# **EXPLORATORY DATA ANALYSIS & CLEANING**

# **MODEL FLOWCHART**

# **MODEL DEVELOPMENT & PERFORMANCE**

# **CONCLUSIONS**

# **REFERENCES**

# **INTRODUCTION**

# Sarcasm is "a sharp, bitter, or cutting expression or remark; a bitter gibe or taunt". Sarcasm may employ ambivalence, although it is not necessarily ironic. Most noticeable in spoken word, sarcasm is mainly distinguished by the inflection with which it is spoken and is largely context-dependent.

# Sarcasm Detection using Machine Learning has always interested in the research community. Given the nature of sarcasm, correctly identifying it in a sentence and clustering it is what makes it challenging according to research scientists.

# Sarcasm detection, when done on a twitter dataset collected using hashtag-based supervision, is prone to too much noise creation in terms of numerous labels and contextual language. To overcome this issue the dataset which we have selected is a collection created from two real news websites, the Onion, and the HuffPost. The Onion consists of news headlines with a sarcastic edge to it while the HuffPost follows a more traditional and formal path in terms of news headlines.

# Advantages of using news dataset over Twitter Dataset are:

# The Onion is a news website with a sole purpose of publishing sarcastic news, this helps us gain high-quality labels with very low noise compared to the twitter dataset

# The News Headlines for both the websites are written by professionals in a formal manner, hence there are no spelling mistakes and informal usage of words. Such language helps in reducing the sparsity and increases the chances of finding pre-trained embeddings.

# The news headlines we obtained are self-contained. It helps us in teasing apart the real sarcastic element.

# Tweets and replies to the tweets are not self-contained and understanding the sarcastic element is based on the context known by a few

# **DATASET PROPERTIES**

# The Headlines Dataset consists of three attributes:

# Article\_link: link to the original news article

# Headline: the headline of the news article

# is\_sarcastic: 1=sarcastic; 0=non-sarcastic

# The Headlines dataset has 28619 records in total with 13634 sarcastic records and 14985 non-sarcastic records.

# 

# **EXPLORATORY DATA ANALYSIS AND CLEANING**

# In order to clean the dataset, we first clean the headlines so as to enable the extraction of features and to tokenize the words. This is also so as to reduce the noise from the tags and punctuation present in the headlines. The steps are as follows:

# **Removing non-ASCII characters:** This removes all characters other than the 128 characters including A-Z, a-z, 0-9, punctuation, spaces and other control codes found on a standard English keyboard.

# **Lemmatization for tags:** This enables grouping together the different forms of a word with a speech parameter like a noun, adjective, verb, etc. This is to enable them being analyzed as a single item.

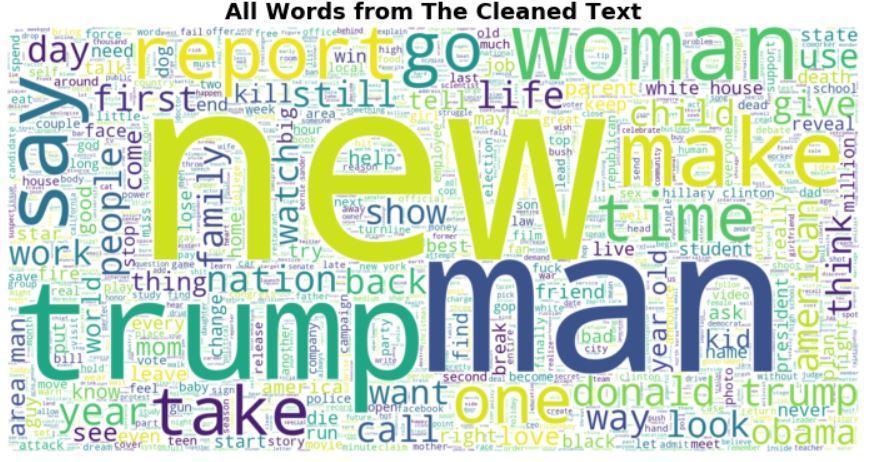
# **Counting and Removal of Punctuation:** This counts the number of URL links, punctuation, numbers, alphanumeric and mention tags in the headlines and removes them.

# **Removing Stop Words:** This removes the stopwords by referencing the words supplied by the function stopwords.words(“English”).

# The data created is then appended together and word frequency and WordClouds are created to establish the word distributions.

# 

*Fig: Most Frequently occurring Words*



*Fig: WordCloud for All Words*

# 

*Fig: WordCloud for Sarcastic & Non-Sarcastic words*

# **FEATURE EXTRACTION**

# The feature extraction is carried out on the Cleaned Headline column and appended to the dataset. These features enable us to carry out diverse model creations.

# The features created using this column are listed as below:

# Number of words used

# Number of unique words used

# Total number of characters used

# Total number of unique words used

# Total number of punctuations used

# Total number of stop words

# The average length of words

# Remove the punctuations and calculate the number of most frequent words

# Frequency of the most common 100 words in the headlines

# Average of frequency terms with total words used

# **MODEL CREATION FLOWCHART:**

# 

*Fig: Flowchart for Model Creation*

# **DATASET PREPARATION FLOWCHART:**

# 

# **MODEL DEVELOPMENT & PERFORMANCE:**

The dataset for Extracted Features and Text features is analyzed and models are applied separately first with the vision to get better predictions.

Model is created and validated using Train and Validation Dataset

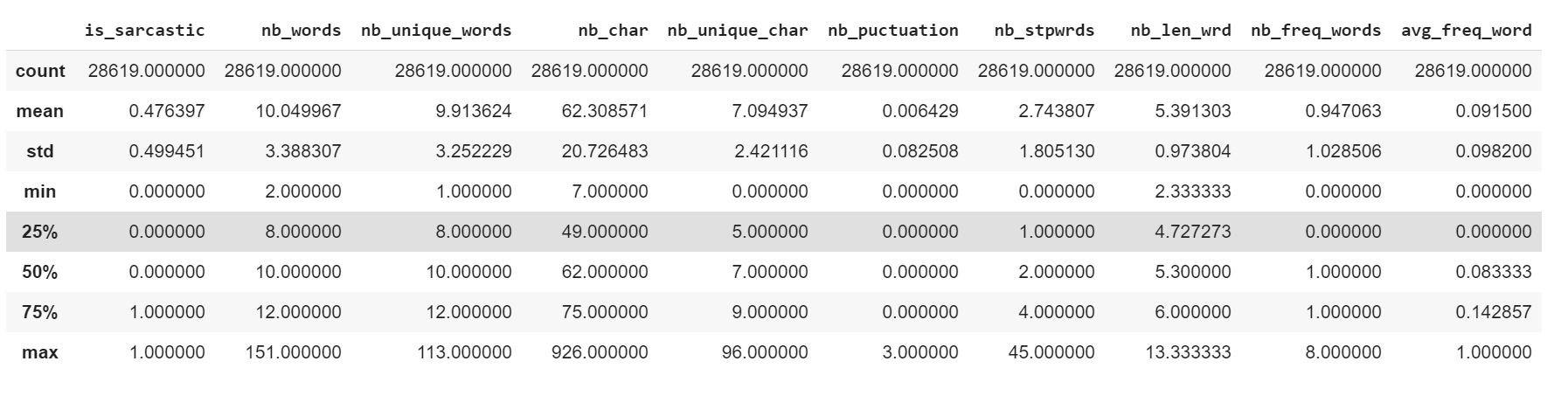
Best Model is tested using Test Dataset

**MODEL CREATION FOR EXTRACTED FEATURES DATASET:**

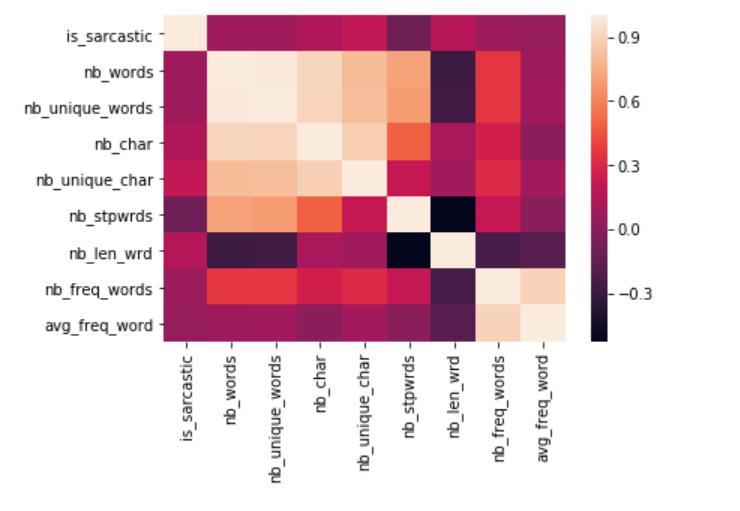
**Feature Extracted –**

* Number of **words** in headline
* Number of **unique words** in headline
* Number of **characters** used in headline
* **Unique words** in headline
* Number of **Punctuations** used
* Number of **Stop Words** used
* Average length of words
* Number of words contain out of most common 100 words
* Average of Frequent terms with total words in headline

**Data Describe –**



*Fig: Correlation Values for extracted features*

**Correlation Matrix between Variables –**

**Building Models**

1. **XGBoost Classifier:** e**X**treme **G**radient **B**oosting is designed for speed and performance with the learning rate.

**Parameters for Model:**

Learning Rate = 0.1

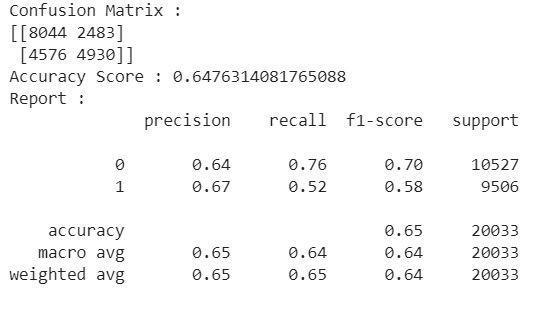
# of estimators = 100

Objective = 'binary:logistic'

**Output of Model –**

Accuracy obtained on Train Dataset = 64.76%.

The confusion matrix is shown below:



1. **XGBoost Classifier with Standardization & Cross-Validation**: This requires the standardization of the dataset using **StandardScaler** before applying the **XGBoostClassifier** as the estimator on the dataset.

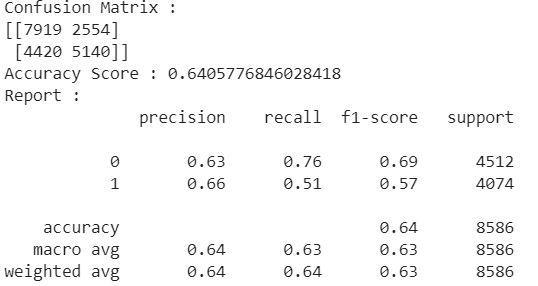
This keeps the accuracy of the model same as before. The accuracy takes a dip when cross-validation is applied to the standardized model.

**Parameters of Model** -

Cross-validation of 10-fold.

**Results –**

Accuracy obtained on Train Dataset= 64.10%.

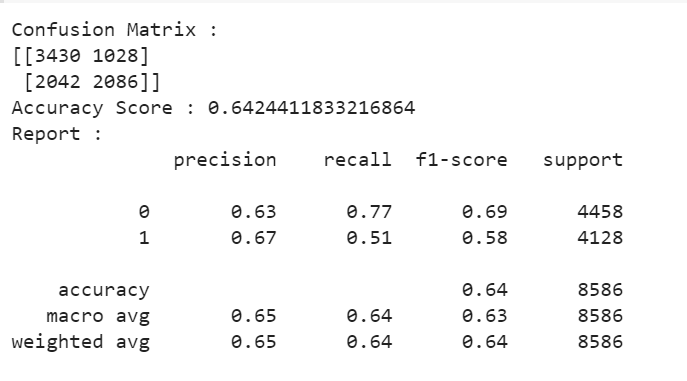
The confusion matrix is as shown below:

**Testing Best model on Test Extracted Features Dataset:**

The best model is XGBoost classifier without applying the cross-validation on the dataset.

Accuracy of Model on Test Dataset= 64.24%

The confusion matrix for the test dataset is as shown below:



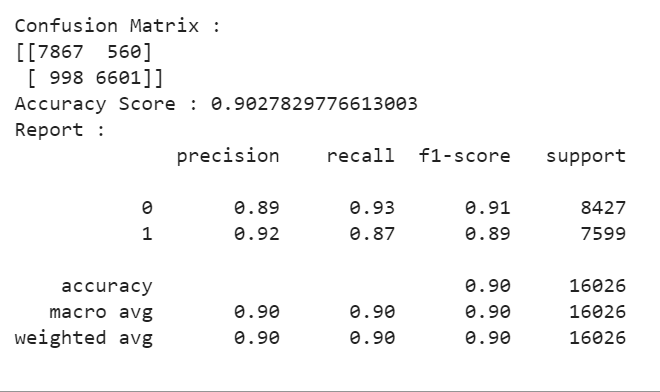
**MODEL CREATION FOR TEXT FEATURES DATASET:**

1. **NAÏVE BAYES CLASSIFIER:** This classifier is used so as to get the probabilistic mean for sarcastic headlines in the text features dataset. It is applied after applying count vectorizer with the analyzer=word and tf-idf transformer with the smooth\_idf=True on the dataset.

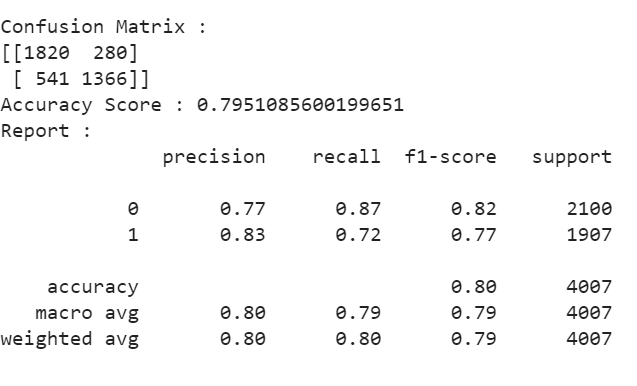
**Results:**

Accuracy Score on Train Dataset = 90.27%

The confusion matrix is as shown below:



Accuracy Score on Validation Dataset = 79.5%

The confusion matrix is as shown below:

1. **XGBOOST CLASSIFIER:** The XGBoost classifier is applied in pipeline form after applying count vectorizer and TF-IDF transformer.

**Parameters of XGBoost:**

Objective='binary:logistic'

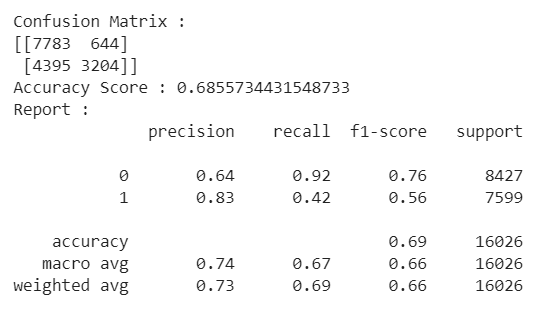
Base\_score = 0.5

Booster = ‘gbtree’

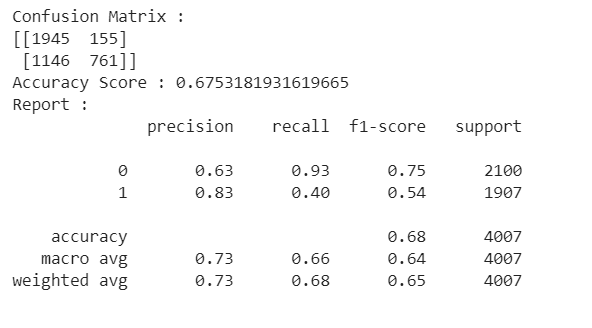
**Results:**

Accuracy Score on Training dataset = 68.5%

The confusion matrix is as shown below:



Accuracy Score on Validation dataset = 67.5%

The confusion matrix is as shown below:

1. **LOGISTIC REGRESSION CLASSIFIER:** This is used in pipeline form after applying count vectorizer and TF-IDF transformer.

**Parameters of Logistic Regression:**

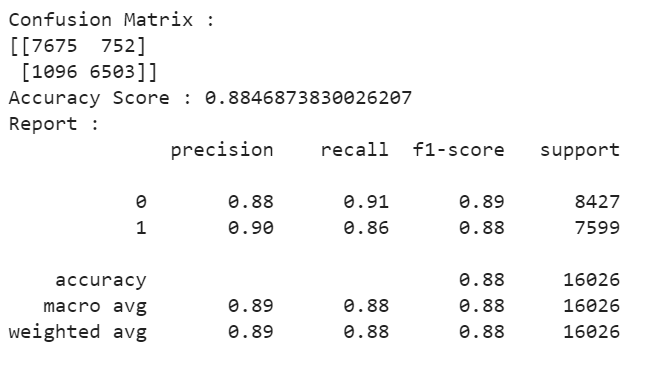
C=1.0

Class\_weight= none

Fit\_intercept = True

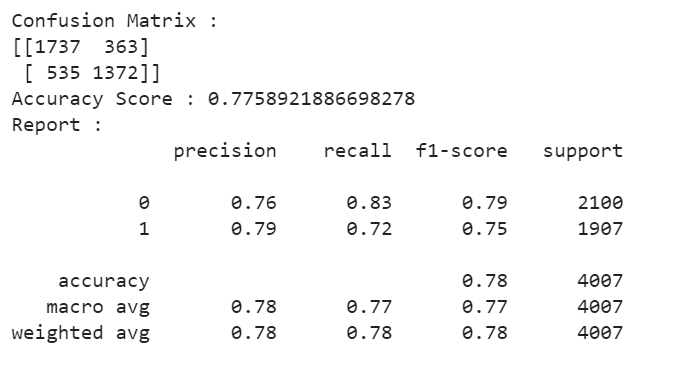
**Results:**

Accuracy Score on Train Dataset = 88.4%

The confusion matrix is as shown below:

Accuracy Score on Validation Dataset = 77.5%

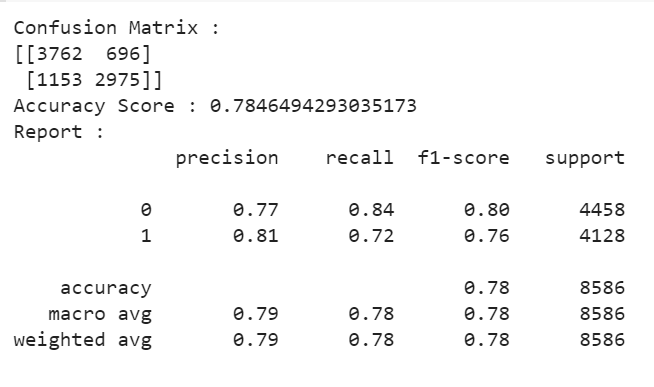
The confusion matrix is as shown below:



**Testing the best model in Test Features Dataset**

The best model is Naïve Bayes Classifier for the text features dataset.

Accuracy Score = 78.4%

The confusion matrix is as shown below: 

**ENSEMBLE MODEL CREATION:**

The goal of using ensemble model is to elevate the robustness of models, accuracy scores, for both types of dataset by combining the predictions of several base estimators created with the current learning algorithms. Here, in this scenario we are going to use voting ensemble model.

**VOTING ENSEMBLE**

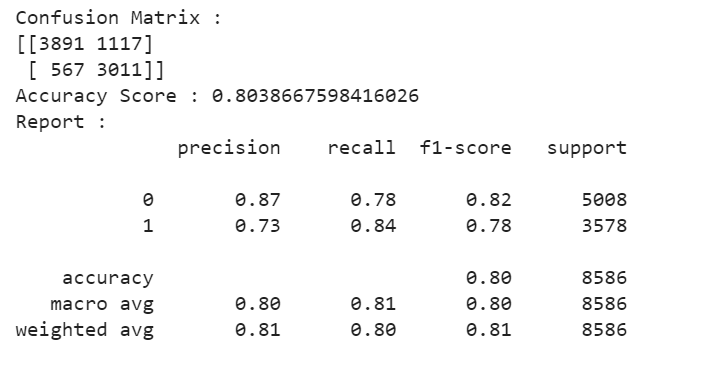
Voting Ensemble is one of the simplest ways of combining the predictions from different machine learning algorithm into one. A Voting classifier is built by wrapping the models and average the predictions of the sub-models.

In our ensemble by combining the probability of is\_sarcastic and not\_sarcastic from Naïve Bayes Model of Text Feature and XGBoost Model of Extracted Feature, we were able to increase the accuracy of the model.

Result of Ensemble Model in Test Dataset:

Accuracy Score = **80.5%**

Confusion Matrix



# **CONCLUSIONS:**

# Following inferences which can be concluded from the project so far:

# Feature extraction from News Headlines holds a significant importance in classifying a Fake News or Legitimate News.

# Text Features using TF-IDF vectorization (term frequency-inverse document frequency) proved to be the best method of classifying the Headlines.

# We saw XGBoost in ensemble method was the best estimator for dataset with Feature variables. Whereas Naïve Bayes was the best estimator for dataset with Text Features.

# From all of the above we found that Ensemble Model using Voting Ensemble method was able to learn both text features and extracted features from our dataset. This model gave the Highest Accuracy of all.

# **FUTURE SCOPE:**

# Accuracy of the model’s can be further analyzed using various different Machine Learning Algorithms like LSTM (Long Short Term Memory), which is an artificial recurrent neural network (RNN).

# Various text analysis can also be done like Word2vec, Count Vectorization for better data exploration and feature extraction.

# **REFERENCES:**

* <https://arxiv.org/pdf/1908.07414.pdf>
* <https://www.kaggle.com/rmisra/news-headlines-dataset-for-sarcasm-detection>
* <https://github.com/rishabhmisra/News-Headlines-Dataset-For-Sarcasm-Detection>