A Project Report for Natural Language Processing

Twitter Sentiment Analysis

Submitted to Manipal University, Jaipur

Towards the partial fulfillment for the Award of the Degree of

BACHELOR OF TECHNOLOGY

In Data Science

2022-2023

By

Perumalla Thushara Meher Siva Mani 209302296

MANIPAL UNIVERSITY JAIPUR

Under the Guidance of

Dr. Vivek Kumar Verma

Department of Information Technology

School of Information Technology

Manipal University Jaipur

Jaipur, Rajasthan

(Nov-2022)

CERTIFICATE

Date: 10/11/2022

This is to certify that the project work entitled **"Twitter Sentiment Analysis"** submitted by **Perumalla Thushara Meher Siva Mani(209302296)** in fulfillment for the requirements of the award of Bachelor of Technology Degree in Data Science at Manipal University Jaipur is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge, the matter embodied in the project has not been submitted to any other University /Institute for the award of any Degree.

Signature of the mentor

Mr.Vivek Verma
Assosciate Professor &
Project Guide
Department of Information
Technology
Manipal University Jaipur

Signature of the HoD

Dr. Pankaj Vyas
Head of the Department
Department of Information
Technology
Manipal University Jaipur

ABSTRACT

Sentiment analysis also called as opinion mining is a (NLP)natural language processing technique which is used to determine whether the given data is negative, positive or neutral. Sentiment analysis is mostly performed on data of text format. It helps companies monitor product and brand sentiment in customer reviews and feedback. It also helps them understand needs of customers.

I have used NLTK, a python library with a ton of utilities, corpora, pre-processing, and trained models for common NLP tasks. Here, the goal is to classify twitter data into sentiments positive or negative(racist). It can be used on all the social media platforms like Twitter, Facebook and Instagram, etc., by their management companies to filter hate speech, abusive words and racist comments to take action against them. It can also be used by any company providing services or involved in direct product sales like Tata Cliq, Amazon, Flipkart, etc., to classify their customer reviews into positive and negative classes using Sentiment Analysis.

I used string.punctuation module to compare and remove punctuation from all the tweets. I used stopwords.words(‘english’) provided by nltk(Natural Language Tool kit) to compare and remove all the stopwords from the tweets. After removing the punctuations and stopwords, I used CountVectorizer( ) from the scikit-learn library to tokenize all the words in tweets into a sparse matrix which is later converted into an array form. I used this tokenized data to classify all the tweets in twitter.csv dataset by using a Multinomial Naïve Bayes Classifier. As this dataset is of text classification, it is better to use Multinomial Naive Bayes Classifier.

CONTENTS

1. INTRODUCTION

Sentiment analysis is contextual mining of text which identifies and extracts subjective information in source material, and helping a business to understand the social sentiment of their brand, product or service while monitoring online conversations. It is used to determine whether the given data is negative, positive or neutral.

Example:-

- "I am so excited to learn Blockchain Technology" is considered to be a sentence with positive sentiment.
- “The customer service is so bad that I could not even solve my problem even after days” is considered to be a sentence with negative sentiment.

In the olden days people used to read and analyze reviews of customers manually but as time goes by, technology has developed, and the data produced is humongous making it impossible to analyze them manually. So, we need an independent, automatic and unbiased solution to extract the summary and emotions from the tweets, reviews and texts which can be solved by Sentiment Analysis on all these texts.

We used tokenizationSo here we will be building a Naïve Bayes Classifier to classify the tweets into negative and positive classes.

1.1 Motivation

We used twitter dataset since it has huge data for classifying public sentiment as compared to normal web blogs and internet articles. Twitter is a social media networking platform used by lakhs of people whereas blogs and articles are written by very few people. We need huge diversified data to get more reliable and accurate predictions. So it is better to use twitter dataset.

1.2 Objectives

- Plot the word cloud to know most frequently used words of both positive and negative dataframes.
- Removal of stop words and punctuations from all the tweets.
- Tokenization of tweets.
- To classify the tweets into positive and negative classes using Naïve Bayes Classifier.

2. Literature Review

Natural language refers to the way we, humans, communicate with each other. Namely, speech and text. Given the importance of this type of data, we must have methods to understand and reason about natural language, just like we do for other types of data. Natural Language Processing, or NLP for short, is broadly defined as the automatic manipulation of natural language, like speech and text, by software. The study of natural language processing has been around for more than 50 years and grew out of the field of linguistics with the rise of computers. Natural language processing is one of the fields in programming where the natural language is processed by the software. This has many applications like sentiment analysis, language translation, fake news detection, grammatical error detection etc.

Sentiment analysis also called as opinion mining is a (NLP)natural language processing technique which is used to determine whether the given data is negative, positive or neutral. Sentiment analysis is mostly performed on data of text format. It helps companies monitor product and brand sentiment in customer reviews and feedback. It also helps them understand needs of customers.

We plotted word cloud in this project. Word cloud is an image of cloud shape containing most frequently used words. Size of word cloud is directly proportional to frequency of the word used in the dataset. We used word cloud to get all the most frequently used words in tweets.

We have to ensure that the data is cleaned whenever we train or feed a model. So,we preprocessed the data by removing all the punctuations and stop words. Punctuations are symbols like ('!"#$%&\'()*+,-./:;<=>?@[\\]^_`{|}~') which are used in languages to divide words and sentences to make the user interpretation of the context easier. Stop words are words which doesn’t have any significant meaning but are used frequently in any language. Some of the examples of stop words are 'i','me','my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself' and 'yourselves',etc.,

We performed tokenization on all the tweets and convert them to tokens. Tokenization is the process of tokenizing or splitting a string, text into a list of tokens.These tokens help in understanding the context or developing the model for the NLP.

We used Naïve Bayes Classifier in this project to classify the tweets into positive and negative classes. Naive Bayes classifiers are a collection of classification algorithms based on Bayes Theorem.It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.It is mainly used in text classification that includes a high-dimensional training dataset.Bayes Theorem finds the probability of an event occurring given the probability of another event that has already occurred. Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

3. Design and Implementation

As this is a text classification problem we can use any of the classifiers like Naïve Bayes, Random Forest, XGBoost, etc., We are using Naïve Bayes Classifier in this project to reach our ultimate goal to classify tweets into positive and negative classes.

Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.It is mainly used in text classification that includes a high-dimensional training dataset.It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

Step by step procedure to build a Naïve Bayes Classifier to classify tweets into positive and negative classes:-

Step 1: Import Libraries

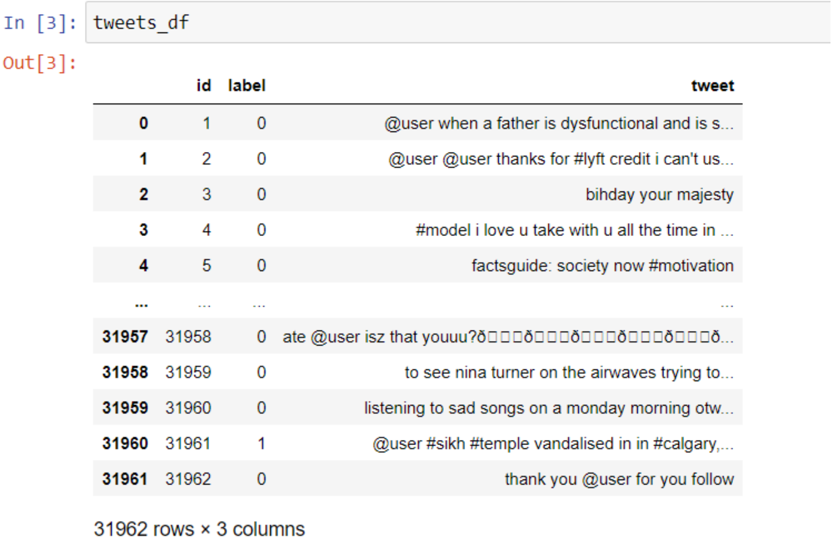
- Import all the necessary libraries such as pandas, numpy, seaborn, matplotlib, etc.,

Step 2: Dataset

2.1 Load the data: We are using twitter dataset to get tweets with their labels mentioned together.

2.2 Perform Data Exploration:

- We have done the data exploration to get the overview of data.



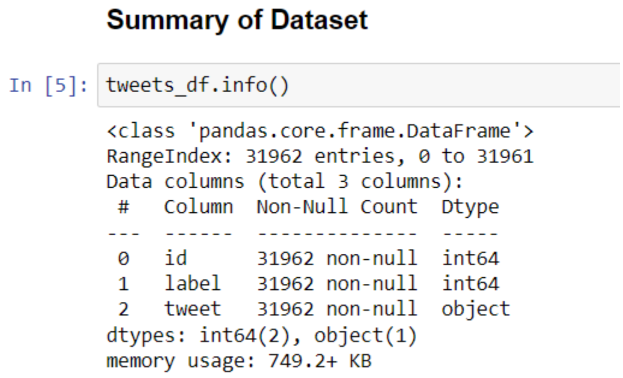
```
In [3]: tweets_df
Out[3]:
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
...	...	...	...
31957	31958	0	ate @user isz that youuu?ð□□□ð□□□ð□□□ð□□□ð□□□ð...
31958	31959	0	to see nina turner on the airwaves trying to...
31959	31960	0	listening to sad songs on a monday morning otw...
31960	31961	1	@user #sikh #temple vandalised in in #calgary,...
31961	31962	0	thank you @user for you follow

31962 rows × 3 columns

Fig 3.1 Overview of twitter.csv dataset

- A simple summary of twitter dataset is:-



Summary of Dataset

```
In [5]: tweets_df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31962 entries, 0 to 31961
Data columns (total 3 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   id      31962 non-null  int64 
 1   label   31962 non-null  int64 
 2   tweet   31962 non-null  object
dtypes: int64(2), object(1)
memory usage: 749.2+ KB
```

Fig 3.2 Summary of twitter.csv dataset

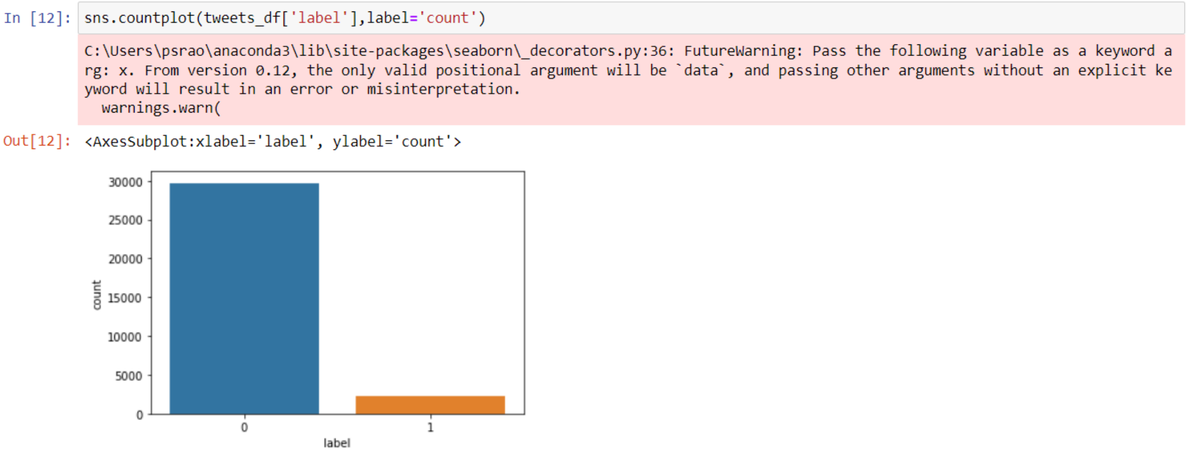
Step 3: Preprocess the data and further data exploration

3.1 Dropping unnecessary columns:

- Dropping 'id' coloumn completely

3.2 Visualizing the classes in dataset

- We have two labels 0(positive tweets) and 1(negative tweets). This is a unbalanced dataset.



```
In [12]: sns.countplot(tweets_df['label'],label='count')

C:\Users\psrao\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword a
rg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit ke
yword will result in an error or misinterpretation.
  warnings.warn(

Out[12]: <AxesSubplot:xlabel='label', ylabel='count'>
```

Fig 3.3 Displaying labels/classes in the dataset

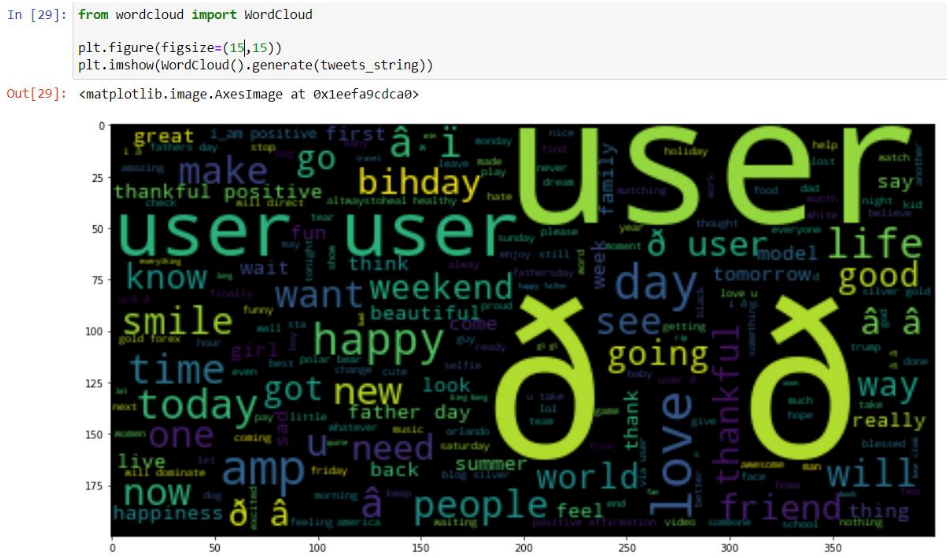
Step 4: Plot the word cloud

4.1 Preparing positive and negative dataframes to plot word clouds seperately

- We are now going to create two new dataframes named 'positive' and 'negative'. We will store all the positive tweets(label==0) in 'positive' dataframe and all the negative tweets(label==1) in 'negative' dataframe.
- Creating seperate lists containing all the positive tweets and negative tweets from positive and negative dataframes respectively.
- Joining all the elements/tweets in the lists to one string such that a word cloud can be plotted.

4.2 Plotting Word cloud

- Pip install word cloud
- Plotting word cloud for positive tweets in positive data frame



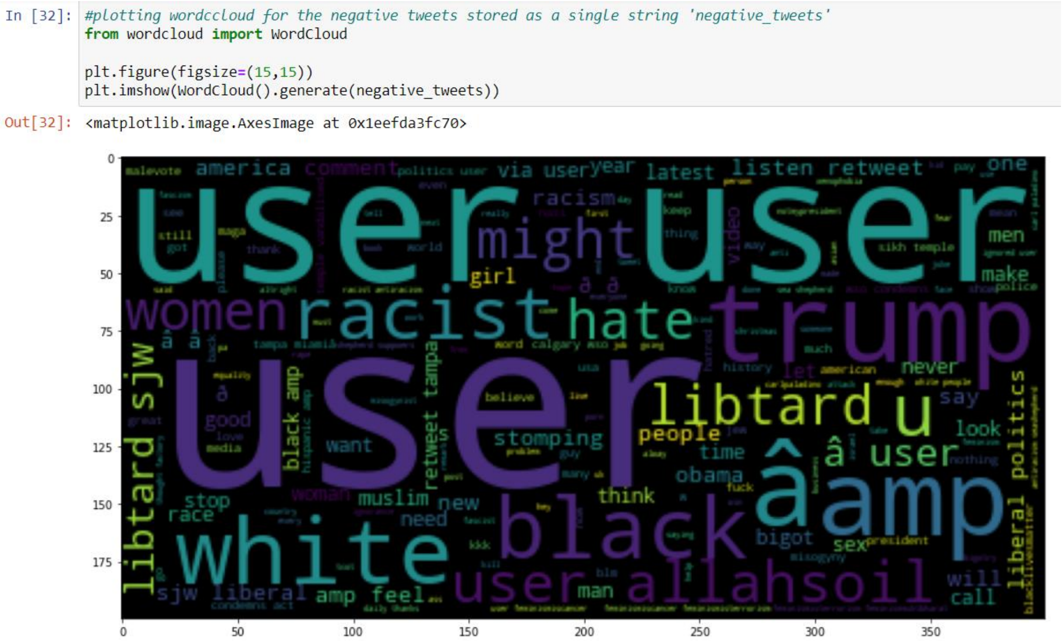
```
In [29]: from wordcloud import WordCloud

         plt.figure(figsize=(15,15))
         plt.imshow(WordCloud().generate(tweets_string))

Out[29]: <matplotlib.image.AxesImage at 0x1eefa9cdca0>
```

Fig 3.4 Word cloud of positive label tweets

- Plotting wordcloud for negative tweets in negative data frame



```
In [32]: #plotting wordccloud for the negative tweets stored as a single string 'negative_tweets'
         from wordcloud import WordCloud

         plt.figure(figsize=(15,15))
         plt.imshow(WordCloud().generate(negative_tweets))

Out[32]: <matplotlib.image.AxesImage at 0x1eefda3fc70>
```

Fig 3.5 Word cloud of negative label tweets

Step 4: Create a pipeline to remove punctuations, stopwords and perform count vectorization

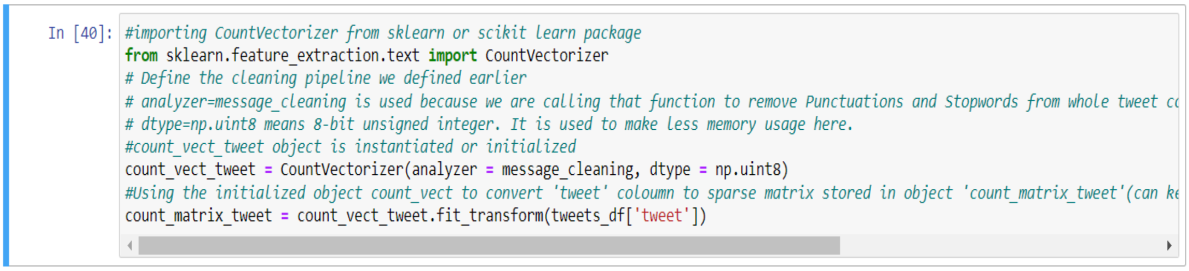
4.1 Create a function to remove punctuations and stopwords

- Import all necessary libraries and packages such as string and nltk to get punctuations and stop words in English Language.

- We created a function called as message_cleaning to remove all the punctuations and stop words

4.2 Perform Tokenization/count vectorization

- As We find that there are no punctuations and Stopwords in cleaned up version compared to the original version.So,we can proceed to Count Vectorization(Tokenization) by calling the function 'message_cleaning'.



```
In [40]: #importing CountVectorizer from sklearn or scikit learn package
         from sklearn.feature_extraction.text import CountVectorizer
         # Define the cleaning pipeline we defined earlier
         # analyzer=message_cleaning is used because we are calling that function to remove Punctuations and Stopwords from whole tweet co
         # dtype=np.uint8 means 8-bit unsigned integer. It is used to make less memory usage here.
         #count_vect_tweet object is instantiated or initialized
         count_vect_tweet = CountVectorizer(analyzer = message_cleaning, dtype = np.uint8)
         #Using the initialized object count_vect to convert 'tweet' coloumn to sparse matrix stored in object 'count_matrix_tweet'(can ke
         count_matrix_tweet = count_vect_tweet.fit_transform(tweets_df['tweet'])
```

Fig 3.6 Code snippet of tokenization

- Assigning variable ‘X’ to sparse matrix obtained after tokenization. Assigning variable ‘y’ to labels feature.

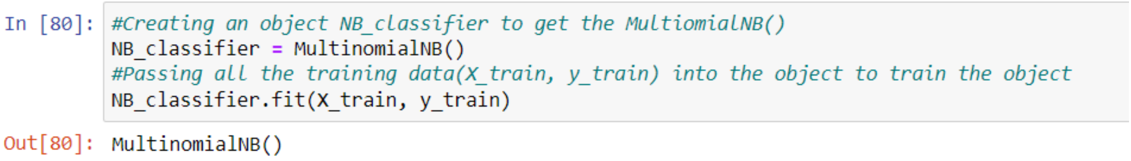
Step 5: Train a Multinomial Naive Bayes classifier model

5.1 Splitting dataset into train and test sets

- Split all the data(X and y) with criteria training data as 80% and testing data as 20% using train_test_split() function.
- We split data(X and y) into X_train, X_test, y_train and y_test.

5.2 Training a Multinomial Naïve Bayes classifier model

- Import all necessary libraries like MultinomialNB(Multinomial Naive bayes Classifier) from sklearn.naive_bayes library, import precision_recall_curve, classification_report and confusion_report from sklearn.metrics,etc.,
- There are 3types of Naive Bayes Classifiers such as Gaussian,Multinomial and Bernoulli but it is better to use Multinomial as it is good for text classification and document classification problems.
- Create a Naïve Bayes classifier object and train it by passing all the training data(X_train, y_train)



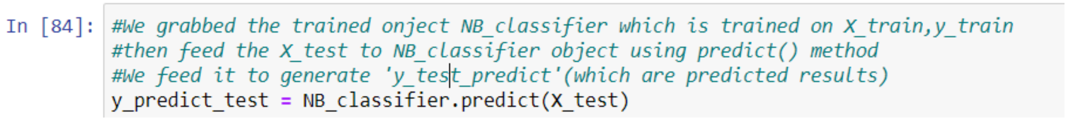
```
In [80]: #Creating an object NB_classifier to get the MultiomialNB()
         NB_classifier = MultinomialNB()
         #Passing all the training data(X_train, y_train) into the object to train the object
         NB_classifier.fit(X_train, y_train)

Out[80]: MultinomialNB()
```

Fig 3.7 Code snippet of training a Naïve Bayes Classifier

Step 6: Predicting the test results

- Feed the X_test to NB_classifier object using predict() method to generate 'y_test_predict'(which are predicted results).



```
In [84]: #We grabbed the trained onject NB_classifier which is trained on X_train,y_train
         #then feed the X_test to NB_classifier object using predict() method
         #We feed it to generate 'y_test_predict'(which are predicted results)
         y_predict_test = NB_classifier.predict(X_test)
```

Fig 3.8 Code snippet of predicting test results

Step 7: Evaluation of Naïve Bayes Classifier model

- Plot confusion matrix

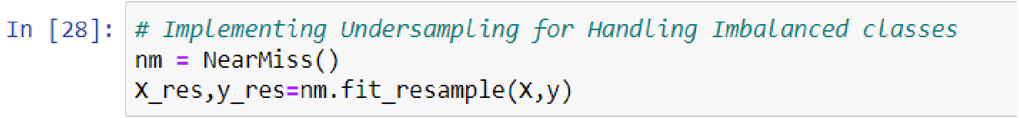
- Classification report for y_test(which is a True class) and y_predict_test(which are Predictions) to get precision,recall anf f1-score.

Step 8: Multinomial Naïve Bayes Classifier after removing class imbalance:

We can handle class imbalance by many methods like undersampling, oversampling, Bagging Based techniques, Boosting-Based techniques, etc., But here we are using undersampling method

8.1 Undersampling to handle unbalanced datasets

- There are many techniques in undersampling like Near-Miss, random undersampling, condensed nearest neighbour rule, etc., but here we are using Near-Miss to perform undersampling.
- Import NearMiss from imblearn.under_sampling
- Perform undersampling using NearMiss



```
In [28]: # Implementing Undersampling for Handling Imbalanced classes
         nm = NearMiss()
         X_res,y_res=nm.fit_resample(X,y)
```

Fig 3.9 Code snippet of Near-Miss Undersampling

8.2 Visualizing the balanced data

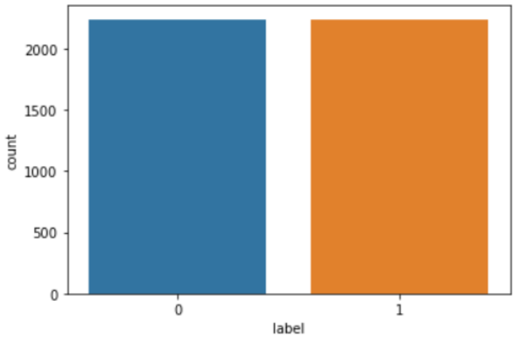
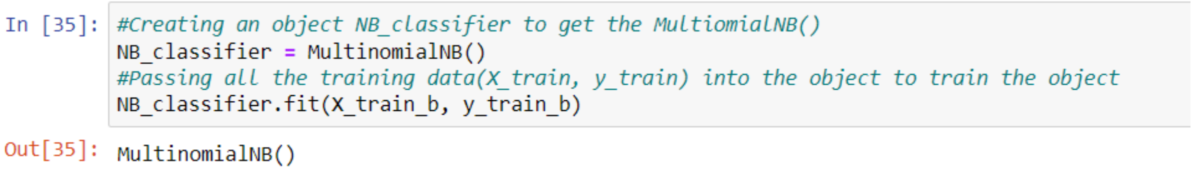


Fig 3.10 Balanced classes/labels

8.3 Training a Multinomial Naïve Bayes classifier model for balanced data

- Split the balanced data to test and train sets
    - Create a Naïve Bayes classifier object and train it by passing all the training data(X_train_b, y_train_

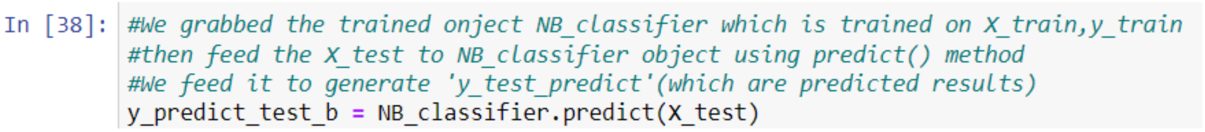


```
In [35]: #Creating an object NB_classifier to get the MultiomialNB()
         NB_classifier = MultinomialNB()
         #Passing all the training data(X_train, y_train) into the object to train the object
         NB_classifier.fit(X_train_b, y_train_b)

Out[35]: MultinomialNB()
```

Fig 3.11 Code snippet of training a Naïve Bayes Classifier for balanced data

- Feed the X_test to NB_classifier object using predict() method to generate 'y_test_predict'(which are predicted results).
- Feed the X_test to NB_classifier object using predict() method to generate 'y_test_predict'(which are predicted results).



```
In [38]: #We grabbed the trained onject NB_classifier which is trained on X_train,y_train
         #then feed the X_test to NB_classifier object using predict() method
         #We feed it to generate 'y_test_predict'(which are predicted results)
         y_predict_test_b = NB_classifier.predict(X_test)
```

Fig 3.12 Code snippet of predicting test results of blanced data

8.4 Evaluation of model trained after balancing the data

- Plot confusion matrix

- Classification report for y_test(which is a True class) and y_predict_test_b(which are Predictions of balanced data) to get precision,recall anf f1-score.

4. RESULT ANALYSIS

- Observations from the confusion matrix of unbalanced data are:-
  - Correctly classified
    - True positive:- 5.8e+03
    - True Negative:- 2.2e+02
  - Misclassified
    - False positive:- 1.5e+02
    - False Negative:- 2.1e+02

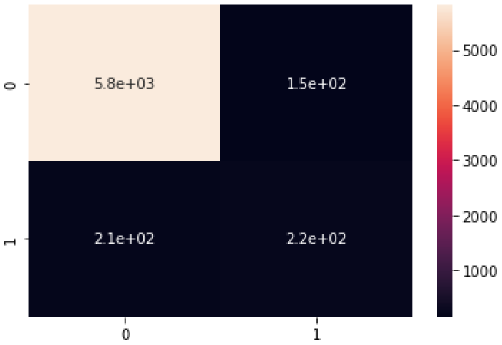


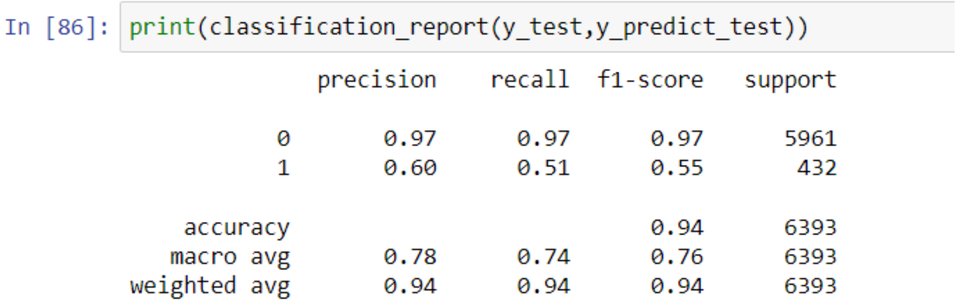
Fig 4.1 Confusion matrix (unbalanced data)

- Explanation of the below Classification report from unbalanced data

  - We have two classes with label0 and label1

  - Class with label0 has precision,recall and f1-score as 97% whereas class with label1 has precision,recall and f1-score as 60%,51% and 55% respectively

  - Overall Accuracy is 94%



```
In [86]: print(classification_report(y_test,y_predict_test))

              precision    recall  f1-score   support

           0       0.97      0.97      0.97      5961
           1       0.60      0.51      0.55       432

    accuracy                           0.94      6393
   macro avg       0.78      0.74      0.76      6393
weighted avg       0.94      0.94      0.94      6393
```

Fig 4.2 Classification report (balanced data)

- Observations from the confusion matrix of balanced data are:-
  - Correctly classified
    - True positive:- 2.7e+02
    - True Negative:- 4.7e+02
  - Misclassified
    - False positive:- 1.4e+02
    - False Negative:- 11

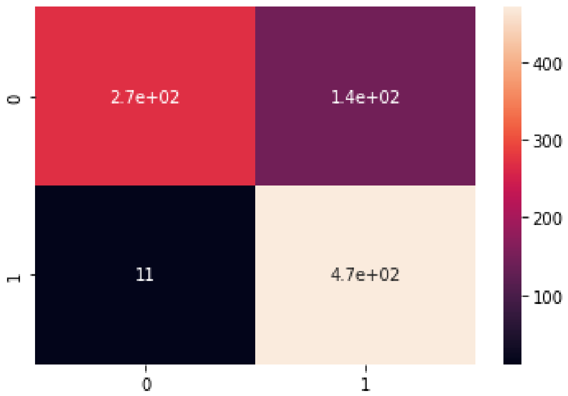
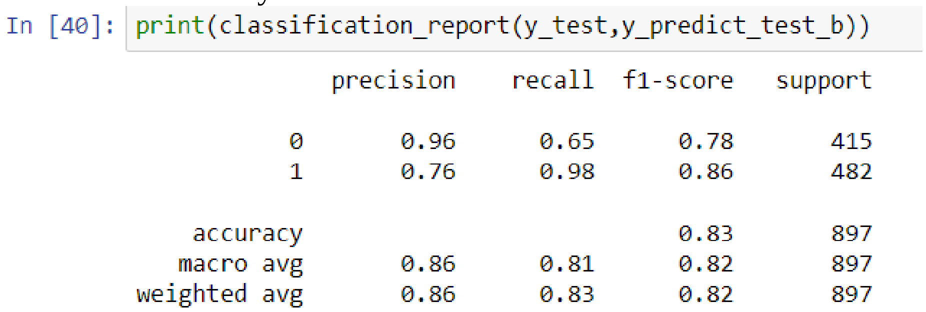


Fig 4.3 Confusion matrix (balanced data)

- Explanation of the below Classification report from balanced data
    - We have two classes with label0 and label1
    - Class with label0 has precision,recall and f1-score as 96%, 67% and 78% respectively whereas class with label1 has precision,recall and f1-score as 76%, 98% and 86% respectively
    - Overall Accuracy is 83%



```
In [40]: print(classification_report(y_test,y_predict_test_b))

                       precision    recall  f1-score   support

                   0        0.96      0.65      0.78       415
                   1        0.76      0.98      0.86       482

            accuracy                            0.83       897
           macro avg        0.86      0.81      0.82       897
        weighted avg        0.86      0.83      0.82       897
```

Fig 4.4 Classification report (balanced data)

5. CONCLUSION

We implemented sentiment analysis on twitter dataset to classify the tweets into positive and negative sentiments. This technique proves to be efficient for sentiment prediction. We have applied Multinomial Naïve Bayes classifier on both unbalanced data and balanced data. Multinomial Naïve Bayes classifier applied on unbalanced data stands out with accuracy 94%.

We used python as a language in this project to do all the necessary steps like loading dataset, data exploration, pre-processing data, cleaning the data, tokenization, splittingg of dataset into training and testing sets, balancing the labels, building Multinomial Naïve Bayes Classifier and evaluation of models, etc.,

There are still some limitations of sentiment analysis as it can only classify the text into positive, negative and neutral classes but human languages have many emotions and tones like joy, surprise, love, sad, angry and hate. Sentiment Analysis generally considers joy, surprise and love emotions as positive label whereas sad, angry and hate as negative label. Inorder to classify the emotions further we need to use Emotion Analysis.

Benefits of sentiment analysis is that it helps companies monitor product and brand sentiment from customer reviews and feedback. It also helps them understand the needs of customers. It can be used on all the social media platforms like Twitter, Facebook and Instagram, etc., by their management companies to filter hate speech, abusive words and racist comments to take action against them. It can also be used by any company providing services or involved in direct product sales like Tata Cliq, Amazon, Flipkart, etc., to classify their customer reviews into positive and negative classes using Sentiment Analysis.

References

- https://towardsdatascience.com/tokenization-for-natural-language-processing-a179a891bad4
- https://www.geeksforgeeks.org/nlp-how-tokenizing-text-sentence-words-works/
- https://www.geeksforgeeks.org/removing-stop-words-nltk-python/
- https://www.geeksforgeeks.org/python-remove-punctuation-from-string/
- https://towardsdatascience.com/text-classification-using-naive-bayes-theory-a-working-example-2ef4b7eb7d5a
- https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17
- https://machinelearningmastery.com/undersampling-algorithms-for-imbalanced-classification/

Acknowledgment

I, Perumalla Thushara Meher Siva Mani, the student of BTech in Data Science and Engineering - 2020 batch (fifth semester) would like to thank all of our teachers and Manipal University Jaipur for providing us with this platform. We would like to thank our mentor Mr.Vivek Verma who helped us to clear any doubt whatsoever we had to encounter while making this project.

This is a wonderful opportunity for us, students, to show our creativity and to get ourselves involved in the work that we love and desire to do. This has enabled us to learn and have fun at the same time. We will try our best to perform as per the expectation put upon us by our parents, family members, teachers, Manipal University Jaipur, and the whole nation.

Lastly, I would like to thank my family for the tremendous amount of support they offered me while making this Project. I also had the invaluable support of my friends and colleagues which helped me in completing this Project.