4222-SURYA GROUP OF INSTITUTIONS

VIKRAVANDI.

Prepared by,

R.HARIHARAN,

422221106009,

ECE-DEP,

3RD YEAR.

AI_PHASE 3:

Data preprocessing is the process of cleaning our data set. There might be missing values or outliers in the dataset.

PREPROCESSING THE GIVEN DATASET:

df = pd.read_csv('../input/covid-19-nlp-text-classification/Corona_NLP_train.csv',encoding='ISO-8859-1')

```
df_test = pd.read_csv('../input/covid-19-nlp-text-classification/Corona_NLP_test.csv')
```

	UserName	ScreenName	Location	TweetAt	OriginalTweet	Sentiment
0	3799	48751	London	16-03-2020	@MeNyrbie @Phil_Gahan @Chrisitv https://t.co/i...	Neutral
1	3800	48752	UK	16-03-2020	advice Talk to your neighbours family to excha...	Positive
2	3801	48753	Vagabonds	16-03-2020	Coronavirus Australia: Woolworths to give elde...	Positive
3	3802	48754	NaN	16-03-2020	My food stock is not the only one which is emp...	Positive
4	3803	48755	NaN	16-03-2020	Me, ready to go at supermarket during the #COV...	Extremely Negative

DATA INFO:

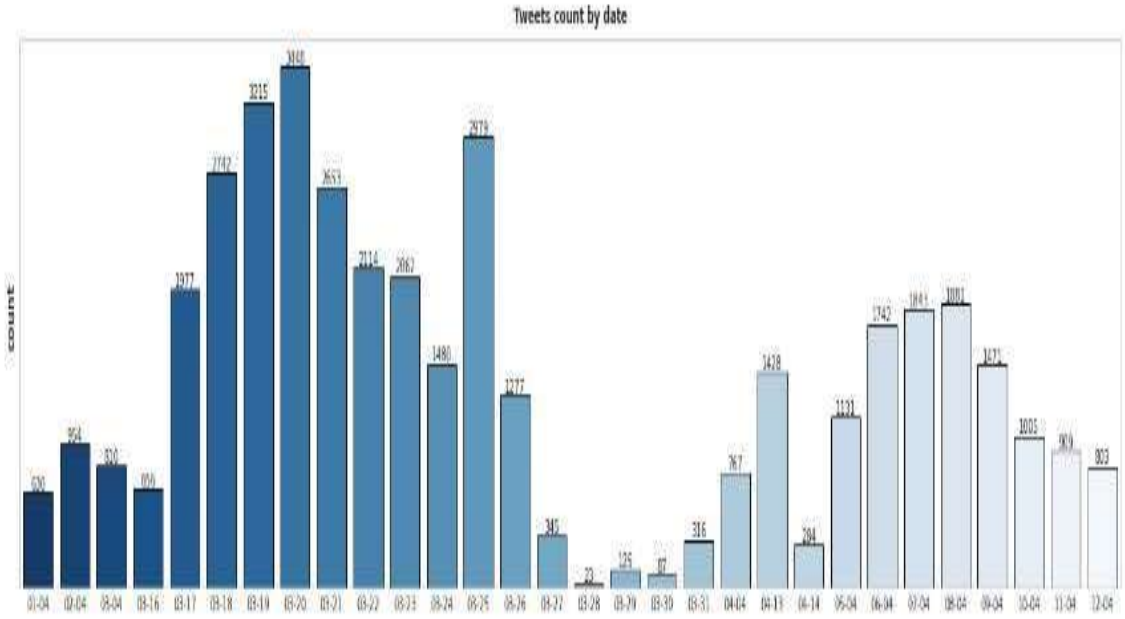
<class 'pandas.core.frame.DataFrame'> RangeIndex:
41157 entries, 0 to 41156 Data columns (total 6
columns):
 # Column Non-Null Count Dtype
--- ------ -------------- -----
0 UserName 41157 non-null int64
1 ScreenName 41157 non-null int64
2 Location 32567 non-null object
3 TweetAt 41157 non-null object
4 OriginalTweet 41157 non-null object 5 Sentiment 41157 non-null object dtypes: int64(2), object(4) memory usage: 1.9+ MB

DUPLICATE TWEET:

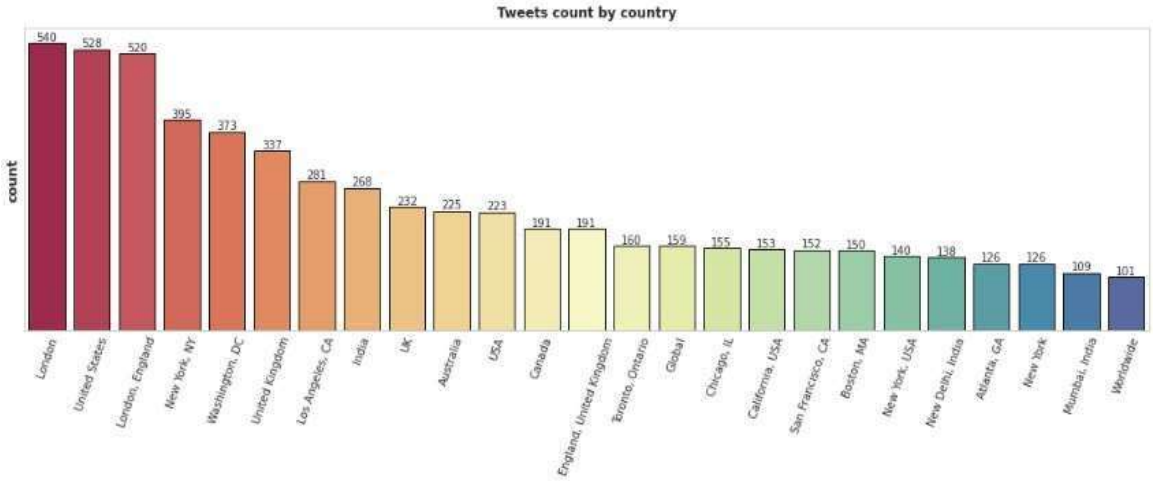
df.drop_duplicates(subset='OriginalTweet',inplace=True)

In [93]: df.info()
<class 'pandas.core.frame.DataFrame'> Int64Index:
41157 entries, 0 to 41156 Data columns (total 6
columns):
 # Column Non-Null Count Dtype
--- ------ -------------- -----
0 UserName 41157 non-null int64
1 ScreenName 41157 non-null int64
2 Location 32567 non-null object
3 TweetAt 41157 non-null datetime64[ns]
4 OriginalTweet 41157 non-null object 5 Sentiment 41157 non-null object
dtypes: datetime64[ns](1), int64(2), object(3)

TWEETS COUNT BY DATA



TWEETS PER COUNTRY AND CITY



TWEETS DEEP CLEANING

df = df[['OriginalTweet','Sentiment']] In

[99]:

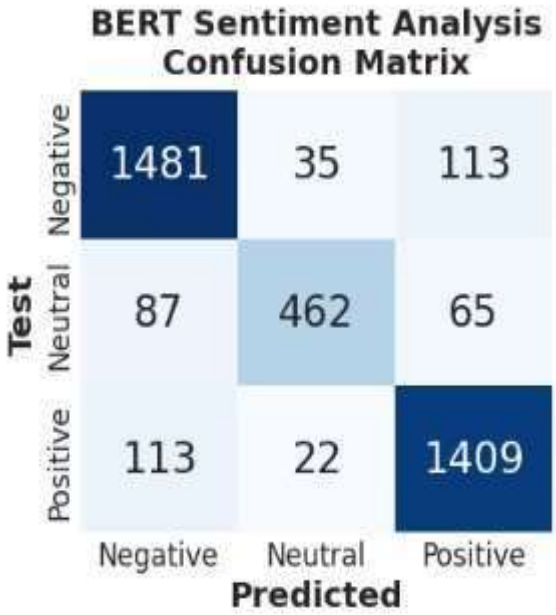
df_test = df_test[['OriginalTweet','Sentiment']]

Then we define custom functions to clean the text of the tweets.

In [100]: linkcode
##CUSTOM DEFINED FUNCTIONS TO CLEAN THE TWEETS

#Clean emojis from text def strip_emoji(text): return re.sub(emoji.get_emoji_regexp(), r"", text) *#remove emoji*

#Remove punctuations, links, mentions and \r\n new line characters def strip_all_entities(text): text = text.replace('**r**', '').replace('**n**', ' ').replace('**n**', ' ').lower() *#remove \n and \r and lowercase* text = re.sub(r"(?:\@|https?\://)\S+", "", text) *#remove links and mentions*
text = re.sub(r'[^\x00-\x7f]',r'', text) *#remove non utf8/ascii characters such as '\x9a\x91\x97\x9a\x97'*
banned_list= string.punctuation + 'Ã'+'±'+'ã'+'¼'+'â'+'»'+'§'
table = str.maketrans('', '', banned_list)
text = text.translate(table) return text

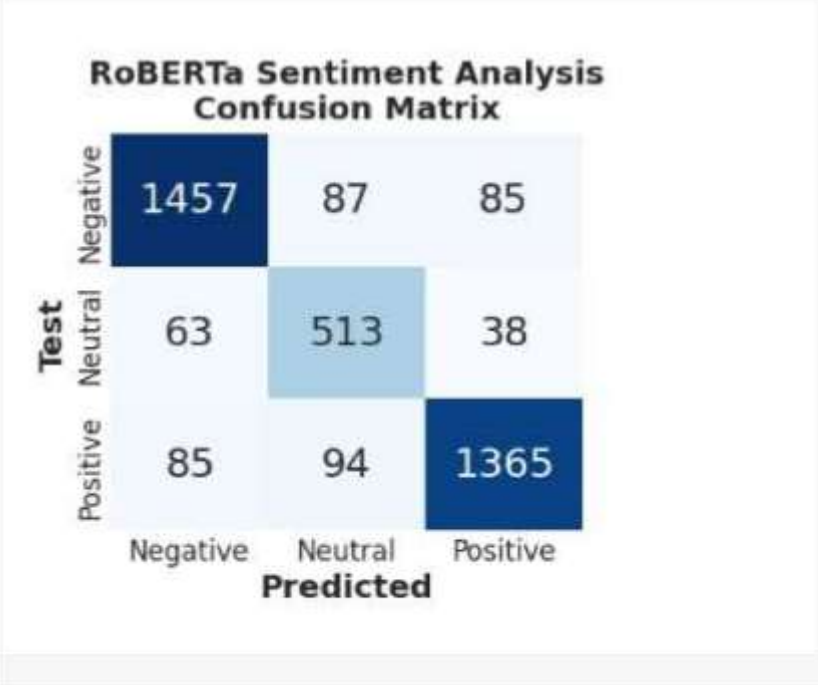


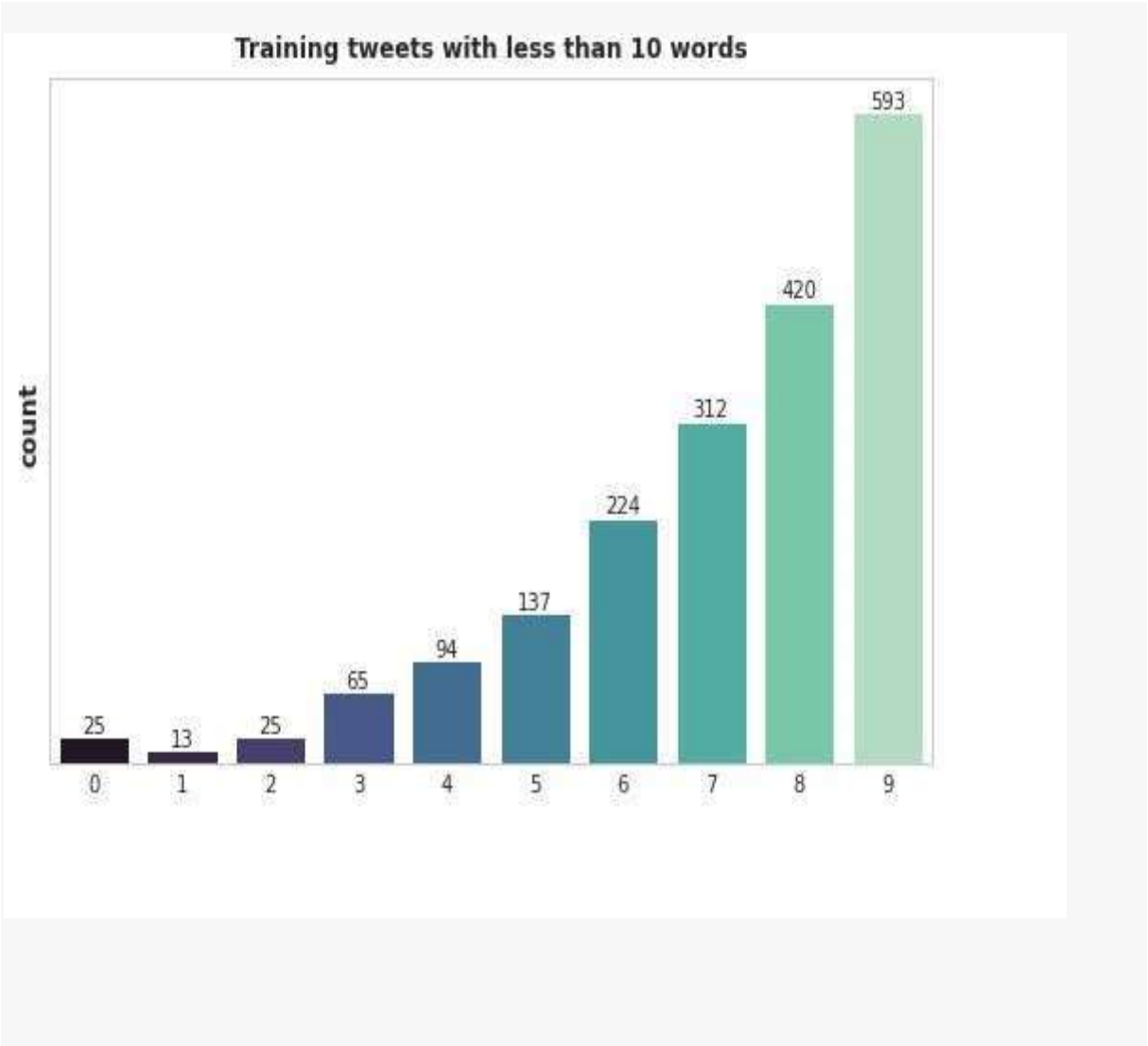
#clean hashtags at the end of the sentence, and keep those in the middle of the sentence by removing just the # symbol def clean_hashtags(tweet):
new_tweet = " ".join(word.strip() for word **in** re.split('#(?!(?:hashtag)**b**)[\w-]+(?=(?:\s+#[\w-]+)*\s*$)', tweet)) *#remove last hashtags*
new_tweet2 = " ".join(word.strip() for word **in** re.split('#|_', new_tweet)) *#remove hashtags symbol from words in the middle of the sentence* return new_tweet2

TWEETS ROBERT CONFUSION:

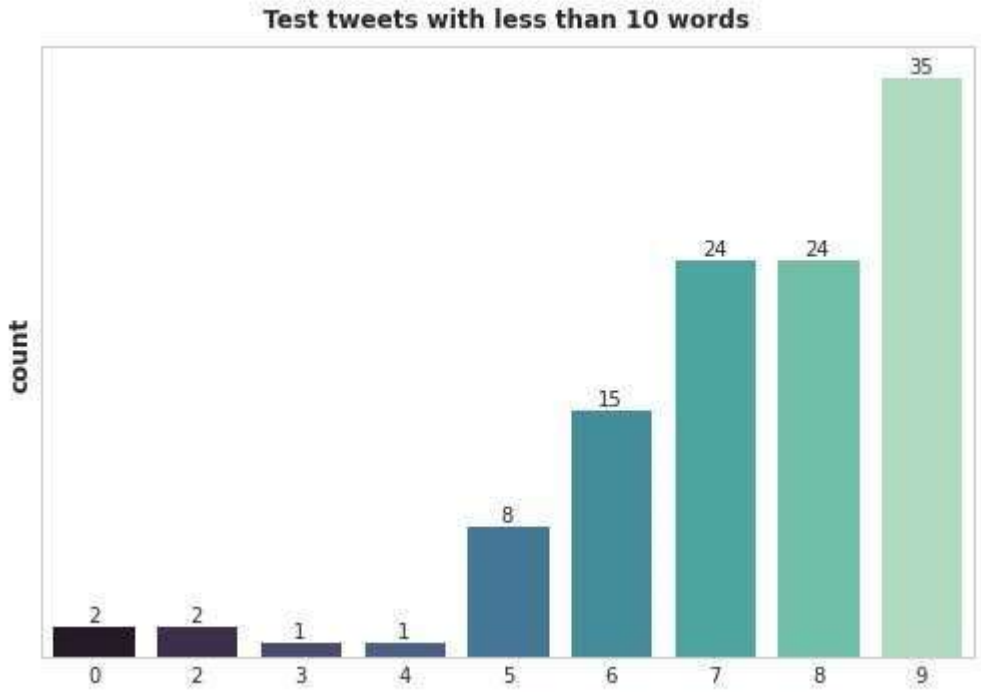
```
#Filter special characters such as & and $ present in some words def
filter_chars(a):      sent = []     for word in a.split('
'):         if ('$' in word) | ('&' in word):
            sent.append('')
else:
sent.append(word)
return ' '.join(sent)

def remove_mult_spaces(text): # remove multiple spaces
return re.sub("\s\s+" , " ", text)
```

```
text_len_test = []
for text in df_test.text_clean:
    tweet_len = len(text.split())
    text_len_test.append(tweet_len)
```



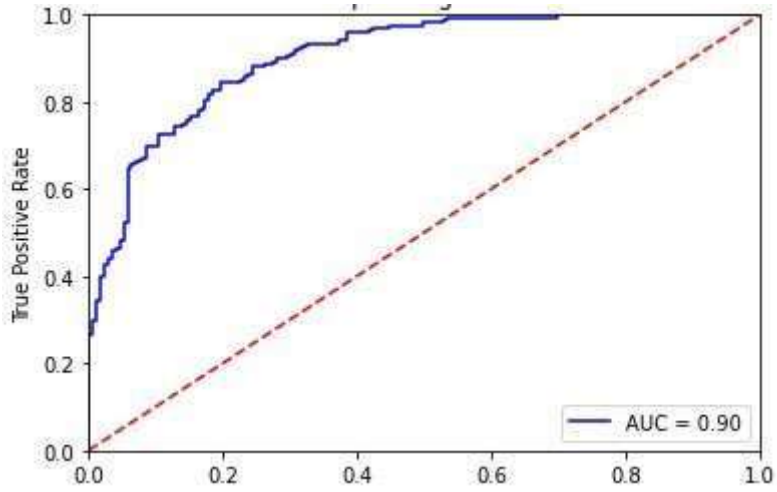



```
plt.figure(figsize=(7,5))
ax = sns.countplot(x='text_len', data=df_test[df_test['text_len']<10], palette='mako') plt.title('Test tweets with less than 10 words')
plt.yticks([])
ax.bar_label(ax.containers[0])
plt.ylabel('count')
plt.xlabel('') plt.show()
```



SENTIMENT COLUMN ANAYSIS

Positive 11381
Negative 9889
Neutral 7560
Extremely Positive 6618
Extremely Negative 5475
Name: Sentiment, dtype: int64



CONCLUSION:

In this project we tried to show the basic way of classifying tweets into positive or negative category using Naive Bayes as baseline and how language models are related to the Naive Bayes and can produce better results.