M508A-GH1019718-NLP assessment -Final1

June 29, 2022

1 IMDB Movie Reviews Classification Assessment

[1]:
```
## Importing image using IPython.image code:
from IPython.display import Image
from IPython.core.display import HTML
Image(url= "https://editor.analyticsvidhya.com/uploads/61727sentiment-fig-1-689.
 ↪jpeg")
```

[1]:
```
<IPython.core.display.Image object>
```

1.0.1 About the data set

In this DataSet 50K reviews about movies have been collected in order to do NLP task analysis. It is worth mentioning that the reviews are separated into **Negative**, **Positive** Reviews. In order to do the sentimental analysis, NLP methods have been used and the results will be shown at the end of Notebook. The Dataset has been downloaded from the Kaggle website with the link below: (https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews)

[2]:
```
## Importing image using IPython.image code:
from IPython.display import Image
from IPython.core.display import HTML
Image(url= "https://raw.githubusercontent.com/Shubham-Trivedi/
 ↪Sentiment-Analysis-on-IMDB-dataset/main/imdb%20logo1.jpg")
```

[2]:
```
<IPython.core.display.Image object>
```

1.0.2 Problem Statement :

The goal is to create an algorithm that can predict the category of a comment or review about a twitter post with high accuracy.

1.0.3 what is sentimental analysis

The point that I would like to begin with is that Sentiment analysis which refers to commonly referred to as opinion mining or emotion AI, uses computational linguistics, biometrics, natural

language processing, and text analysis to systematically identify, extract, measure, and analyze affective states and subjective data (source: Wikipedia).

[3]:
```
## Importing image using IPython.image code:
from IPython.display import Image
from IPython.core.display import HTML
Image(url= "https://cdn-images-1.medium.com/max/361/0*ga5rNPmVYBsCm-lz.")
```

[3]:
```
<IPython.core.display.Image object>
```

1.0.4 Introduction :

These days, the number of people who use social media has increased. They make a different kinds of media like writing about their daily life, sharing images and videos, and sending private and public messages to each other. Twitter is one of the most popular social networks. With 330 million monthly active users and 500 million tweets posted each day, Twitter is one of the most widely used social networking sites in the world. To understand how individuals approach a particular topic, it is crucial to analyse tweets for a variety of purposes, including corporate marketing, politics, the research of public behavior, and information collecting, it is crucial to comprehend the emotion of tweets. Political parties and marketers can both benefit from sentiment analysis of Twitter data to better understand how consumers feel about new products and marketing initiatives.

1.0.5 objective:

Give a brief to explain the algorithm models, In order to get over the challenges of figuring out the feelings of the tweets, we are attempting to apply a Twitter sentiment analysis model in this project. We want to examine the sentiment of the tweets provided from the Sentiment140 dataset by building a machine learning pipeline that uses three classifiers:

- Logistic Regression.
- MultinominalNB.
- Support Vector Machine.

1.0.6 Project Pipeline :

1 Import Necessary Libraries.

2 Read and Load the Dataset.

3 Exploratory Data Analysis.

4 Data Visualization of Target Variables.

5 Data Preprocessing.

6 Data Visualization after Preprocessing.

7 Splitting our data into Train and Test Subset.

1.1 1 Import Necessary libraries.

- **NLTK**— the Natural Language Toolkit —there are quite a range of functions that are provided by importing NLTK library such as categorization, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries
- **pandas**:There is quite an extensive diversity of methods which is provided by pandas for users, for instance, read/write datasets in a variety of formats like TEXT, CSV, XLS, JSON, SQL, HTML, and many more. Concerning the matter of performance, It gives high performance for data mining, reshaping, sub-setting, data alignment, slicing, indexing, and merging/joining data sets. (https://www.educba.com/machine-learning-libraries/)
- with regard to data pre-processing I import **Numpy** library.
- The other library that I like to add here is **re** which I used in order to clean the data from tags, spaces, numbers, and punctuations
- with the aim of splitting the data into train and test split I used **sklearn**.
- Due to the fact that I wanted to plot the history of the model I used **matplotlib** visualization library
- we need a **seaborn** visualization library in order to plot histograms to the positive and negative data.

[4]:
```
## Importing image using IPython.image code:
from IPython.display import Image
from IPython.core.display import HTML
Image(url= "https://miro.medium.com/max/1200/1*qsRJFHCxC0edtLuZeoUi4g.png")
```

[4]:
```
<IPython.core.display.Image object>
```

[1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import tensorflow as tf
from sklearn import preprocessing
import sklearn.model_selection
import sklearn.feature_extraction
from sklearn.model_selection import train_test_split
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
from wordcloud import WordCloud
import sklearn.model_selection
import simpletransformers.classification
```

```
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from nltk.stem.porter import PorterStemmer
from gensim.parsing.preprocessing import remove_stopwords
from nltk.tokenize import word_tokenize # Tokenizaion

import nltk
# Plotting libraries :
import seaborn as sns
from wordcloud import WordCloud
import matplotlib.pyplot as plt
# sklearn :
import sklearn
from sklearn.svm import LinearSVC
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.model_selection import train_test_split # Import train_test_split␣
 ↪function
from sklearn import metrics #Import scikit-learn metrics module for accuracy␣
 ↪calculation
from sklearn.metrics import roc_curve, auc
```

```
2022-06-29 10:27:03.743911: W
tensorflow/stream_executor/platform/default/dso_loader.cc:64] Could not load
dynamic library 'libcudart.so.11.0'; dlerror: libcudart.so.11.0: cannot open
shared object file: No such file or directory
2022-06-29 10:27:03.743950: I tensorflow/stream_executor/cuda/cudart_stub.cc:29]
Ignore above cudart dlerror if you do not have a GPU set up on your machine.
```

1.2 2 Read and Load the Dataset.

• The first point I would to mention here is that In order to build our classifier model, we need a dataset that contains a huge number of tweets that include corresponding feelings being expressed on tweeter by users. I took this Data Set from the kaggle website .(https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews) • Another point that I like to add here is that I named my data fram: Moviedf • As a result of this In the first step I download the data set from Kaggle, save it then upload it on my jupyterhub then use the read_csv function to load this dataset in CSV format.

[7]:
```
Moviedf = pd.read_csv("./Dataset/IMDB Dataset.csv")
Moviedf.head()
```

[7]:

	review	sentiment
0	One of the other reviewers has mentioned that …	positive
1	A wonderful little production. The…	positive

```
2  I thought this was a wonderful way to spend ti…  positive
3  Basically there's a family where a little boy …  negative
4  Petter Mattei's "Love in the Time of Money" is…  positive
```

[8]:
```
Moviedf.head()
```

[8]:
```
                                              review sentiment
0  One of the other reviewers has mentioned that …  positive
1  A wonderful little production. <br /><br />The…  positive
2  I thought this was a wonderful way to spend ti…  positive
3  Basically there's a family where a little boy …  negative
4  Petter Mattei's "Love in the Time of Money" is…  positive
```

1.3 3 Exploratory Data Analysis :

In exploratory section I tried to explore the Data Set as much as possible . I tried to display: * The column names * Get the info about the data set * Shape of the Data * Type of the Data * Null values * Numbers of columns and rows

[9]:
```
# Main names of our dataset columns :
Moviedf.columns
```

[9]:
```
Index(['review', 'sentiment'], dtype='object')
```

[10]:
```
# Getting information about our dataset :
Moviedf.info
```

[10]:
```
<bound method DataFrame.info of
review sentiment
0      One of the other reviewers has mentioned that …  positive
1      A wonderful little production. <br /><br />The…  positive
2      I thought this was a wonderful way to spend ti…  positive
3      Basically there's a family where a little boy …  negative
4      Petter Mattei's "Love in the Time of Money" is…  positive
…                                                    …       …
49995  I thought this movie did a down right good job…  positive
49996  Bad plot, bad dialogue, bad acting, idiotic di…  negative
49997  I am a Catholic taught in parochial elementary…  negative
49998  I'm going to have to disagree with the previou…  negative
49999  No one expects the Star Trek movies to be high…  negative

[50000 rows x 2 columns]>
```

[11]:
```
# The shape of our data :
Moviedf.shape
```

[11]: (50000, 2)

[12]:
```
# Type of our data
print(Moviedf.dtypes)
```

```
review       object
sentiment    object
dtype: object
```

The data type of columns in our dataset is object, which means we still have to process our data before getting into machine learning stuff and do the lable encoding in following steps

[13]:
```
print (Moviedf.applymap(type))
```

```
                review      sentiment
0      <class 'str'>  <class 'str'>
1      <class 'str'>  <class 'str'>
2      <class 'str'>  <class 'str'>
3      <class 'str'>  <class 'str'>
4      <class 'str'>  <class 'str'>
…               …              …
49995  <class 'str'>  <class 'str'>
49996  <class 'str'>  <class 'str'>
49997  <class 'str'>  <class 'str'>
49998  <class 'str'>  <class 'str'>
49999  <class 'str'>  <class 'str'>

[50000 rows x 2 columns]
```

It is necessary to check if we have null values or not,if there are any empty value we need to use methods to replce them or delete them.As it is obvious here we have no null article

[14]:
```
(Moviedf.isnull() | Moviedf.empty).sum()
```

[14]:
```
review       0
sentiment    0
dtype: int64
```

number of missing values in the dataframe is 0

continuing the process of data exploration I tried to display the number of columns and rows of my data set.The function shows that in this dataset there are 2 columns and 5000 rows

[15]:
```
print('Count of columns in the data is:  ', len(Moviedf.columns))
print('Count of rows in the data is:  ', len(Moviedf))
```

```
Count of columns in the data is:   2
Count of rows in the data is:   50000
```

[16]:
```
# Replacing the values to ease understanding :
Moviedf['review'] = Moviedf['review'].replace(4,1)
```

Count of columns in the data is: 2 Count of rows in the data is: 5000

[17]:
```
Moviedf.head()
```

[17]:

	review	sentiment
0	One of the other reviewers has mentioned that …	positive
1	A wonderful little production. The…	positive
2	I thought this was a wonderful way to spend ti…	positive
3	Basically there's a family where a little boy …	negative
4	Petter Mattei's "Love in the Time of Money" is…	positive

1.4 4 Data Visualization of Target Variables :

The 4th step is to visualize our data using mathematical plots after processing our data and focusing on the columns we are interested in. We use plots because they shed light on the data and makes the data more understandable.

[18]:
```
Moviedf.groupby('sentiment').count()
```

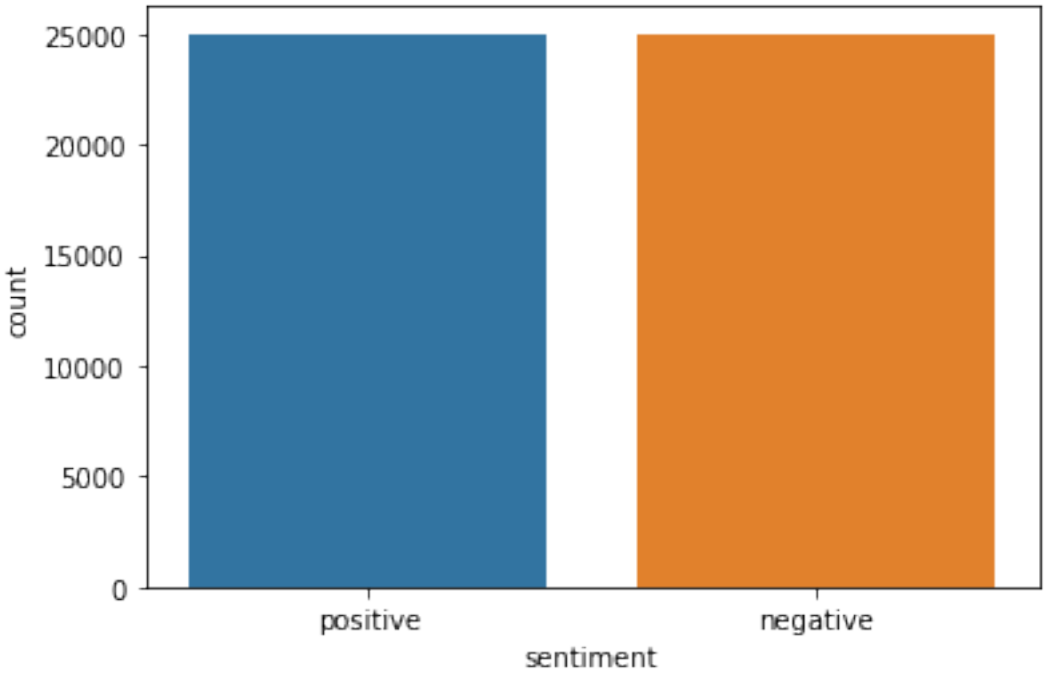
[18]:

sentiment	review
negative	25000
positive	25000

As a result of **groupby** we can conclude that we have 25000 negative reviews and 25000 positive reviews on our tweet.

[19]:
```
import seaborn as sns

sns.countplot(x=Moviedf['sentiment'])
```

[19]:
```
<AxesSubplot:xlabel='sentiment', ylabel='count'>
```



countplot visualize our negative and positive views on a bar chart.

1.5 5 Data Preprocessing

Our data is typically extracted from a wide range of sources and is frequently in a wide range of forms full of punctuations, links, emojis, expressions, hashtags, lower and upper case letters, numbers, and different formats of words. As a result, it is crucial to do the Data Pre-processing step to make our data ready for modeling But there is a catch here as textual material frequently contains redundant and/or repeating words, and cleansing is not an easy operation. We will pre-process the dataset in a number of ways before training the model, including:

1. Lowering Case.
2. Removing Space Charachters.
3. Removal of Stop words.
4. Stopword Removal using Gensim.
5. Removal of Links/URLs.
6. Removal of numbers.
7. Tokenizing the text feature.
8. Stemming and Lemmitizing

5-1 Lowering Case: The lower case option helps us to equalize words of similar value. The size of our vocabulary will be greatly reduced by doing this.

[20]:
```
#Lowering Case :

print("=========== Before Lowering case ============\n")
print("\t" + Moviedf.loc[10, "review"])
print("\n=========== After Lowering case ===========\n")
Moviedf['review'] = Moviedf['review'].str.lower()
print("\t" + Moviedf.loc[10, "review"])
```

```
=========== Before Lowering case ============

        Phil the Alien is one of those quirky films where the humour is based
around the oddness of everything rather than actual punchlines.<br /><br />At
first it was very odd and pretty funny but as the movie progressed I didn't find
the jokes or oddness funny anymore.<br /><br />Its a low budget film (thats
never a problem in itself), there were some pretty interesting characters, but
eventually I just lost interest.<br /><br />I imagine this film would appeal to
a stoner who is currently partaking.<br /><br />For something similar but better
try "Brother from another planet"

=========== After Lowering case ===========

        phil the alien is one of those quirky films where the humour is based
around the oddness of everything rather than actual punchlines.<br /><br />at
first it was very odd and pretty funny but as the movie progressed i didn't find
the jokes or oddness funny anymore.<br /><br />its a low budget film (thats
never a problem in itself), there were some pretty interesting characters, but
eventually i just lost interest.<br /><br />i imagine this film would appeal to
a stoner who is currently partaking.<br /><br />for something similar but better
try "brother from another planet"
```

1.5.1 5-2 Removal of Stop words:

In any Natural language, the most common words are called Stopwords. In doing sentimental analysis these stop words do not value a lot in our process. as a result of this, I remove these words using **Gensim** method.

[21]:
```
Moviedf.loc[12]
```

[21]:
```
review       so im not a big fan of boll's work but then ag…
sentiment                                           negative
Name: 12, dtype: object
```

[22]:
```
Moviedf.loc[12]
```

[22]:
```
review       so im not a big fan of boll's work but then ag…
sentiment                                           negative
Name: 12, dtype: object
```

[23]:
```
## Applying the fucntion to all rows
import gensim
print("============ Before Removing Stop words with Gensim =======\n")
print("\t" + Moviedf.loc[12, "review"])
print("\n============ After Removing Stop words with Gensim =======\n")
Moviedf['review'] = Moviedf['review'].apply(lambda x: gensim.parsing.
 ↪preprocessing.remove_stopwords(x))
print("\t" + Moviedf.loc[12, "review"])
```

```
============ Before Removing Stop words with Gensim =======

        so im not a big fan of boll's work but then again not many are. i
enjoyed his movie postal (maybe im the only one). boll apparently bought the
rights to use far cry long ago even before the game itself was even finsished.
<br /><br />people who have enjoyed killing mercs and infiltrating secret
research labs located on a tropical island should be warned, that this is not
far cry… this is something mr boll have schemed together along with his legion
of schmucks.. feeling loneley on the set mr boll invites three of his countrymen
to play with. these players go by the names of til schweiger, udo kier and ralf
moeller.<br /><br />three names that actually have made them selfs pretty big in
the movie biz. so the tale goes like this, jack carver played by til schweiger
(yes carver is german all hail the bratwurst eating dudes!!) however i find that
tils acting in this movie is pretty badass.. people have complained about how
he's not really staying true to the whole carver agenda but we only saw carver
in a first person perspective so we don't really know what he looked like when
he was kicking a**.. <br /><br />however, the storyline in this film is beyond
demented. we see the evil mad scientist dr. krieger played by udo kier, making
genetically-mutated-soldiers or gms as they are called. performing his top-
secret research on an island that reminds me of "spoiler" vancouver for some
reason. thats right no palm trees here. instead we got some nice rich
lumberjack-woods. we haven't even gone far before i started to cry (mehehe) i
cannot go on any more.. if you wanna stay true to bolls shenanigans then go and
see this movie you will not be disappointed it delivers the true boll
experience, meaning most of it will suck.<br /><br />there are some things worth
mentioning that would imply that boll did a good work on some areas of the film
such as some nice boat and fighting scenes. until the whole cromed/albino gms
squad enters the scene and everything just makes me laugh.. the movie far cry
reeks of scheisse (that's poop for you simpletons) from a fa,r if you wanna take
a wiff go ahead.. btw carver gets a very annoying sidekick who makes you wanna
shoot him the first three minutes he's on screen.

============ After Removing Stop words with Gensim =======

        im big fan boll's work are. enjoyed movie postal (maybe im one). boll
apparently bought rights use far long ago game finsished. <br /><br />people
enjoyed killing mercs infiltrating secret research labs located tropical island
warned, far cry… mr boll schemed legion schmucks.. feeling loneley set mr boll
```

```
invites countrymen play with. players names til schweiger, udo kier ralf
moeller.<br /><br />three names actually selfs pretty big movie biz. tale goes
like this, jack carver played til schweiger (yes carver german hail bratwurst
eating dudes!!) tils acting movie pretty badass.. people complained he's staying
true carver agenda saw carver person perspective don't know looked like kicking
a**.. <br /><br />however, storyline film demented. evil mad scientist dr.
krieger played udo kier, making genetically-mutated-soldiers gms called.
performing top-secret research island reminds "spoiler" vancouver reason. thats
right palm trees here. instead got nice rich lumberjack-woods. haven't gone far
started (mehehe) more.. wanna stay true bolls shenanigans movie disappointed
delivers true boll experience, meaning suck.<br /><br />there things worth
mentioning imply boll good work areas film nice boat fighting scenes.
cromed/albino gms squad enters scene makes laugh.. movie far reeks scheisse
(that's poop simpletons) fa,r wanna wiff ahead.. btw carver gets annoying
sidekick makes wanna shoot minutes he's screen.
```

5-4 Deleting Numbers

I removed the number due to the fact that my concern is to clarify the twitter comment using the negative and positive reviews and I don't need numbers mentioned in comments.

[24]:
```
## Creating a fucntion that will be applied to our datset :
import re
def RemoveNumbers(text):
    return re.sub(r"[0-9]+", "", text)

## Applying the fucntion to all rows
print("=========== Before Removing Numbers =======\n")
print("\t" + Moviedf.loc[2,"review"])  #let's see for example the thirs row,␣
 ↪which contains an number 50
print("\n=========== After Removing Numbers ========\n")
Moviedf['review'] = Moviedf['review'].apply(RemoveNumbers)
print("\t" + Moviedf.loc[2,"review"])
```

```
=========== Before Removing Numbers =======

        thought wonderful way spend time hot summer weekend, sitting air
conditioned theater watching light-hearted comedy. plot simplistic, dialogue
witty characters likable (even bread suspected serial killer). disappointed
realize match point 2: risk addiction, thought proof woody allen fully control
style grown love.<br /><br />this i'd laughed woody's comedies years (dare
decade?). i've impressed scarlet johanson, managed tone "sexy" image jumped
right average, spirited young woman.<br /><br />this crown jewel career, wittier
"devil wears prada" interesting "superman" great comedy friends.

=========== After Removing Numbers ========

        thought wonderful way spend time hot summer weekend, sitting air
conditioned theater watching light-hearted comedy. plot simplistic, dialogue
```

witty characters likable (even bread suspected serial killer). disappointed realize match point : risk addiction, thought proof woody allen fully control style grown love.

this i'd laughed woody's comedies years (dare decade?). i've impressed scarlet johanson, managed tone "sexy" image jumped right average, spirited young woman.

this crown jewel career, wittier "devil wears prada" interesting "superman" great comedy friends.

5-6 Removal of Links/URLs:* As we dont need Links and URLs in our Data set,i remove them using RemoveLinks .

[25]:
```
## Creating a fucntion that will be applied to our datset :
def RemoveLinks(text):
    return re.sub(r"http\S+", "", text)


## Applying the fucntion to all rows of our dataset :
print("============ Before Removing Hyperlinks =======\n")
print("\t" + Moviedf.loc[0, "review"])  # let's see for example the first row,␣
 ↪which contains an hyperlink.
print("\n============ After Removing Hyperlinks =======\n")
Moviedf['review'] = Moviedf['review'].apply(RemoveLinks)
print("\t" + Moviedf.loc[0, "review"])
```

```
============ Before Removing Hyperlinks =======

        reviewers mentioned watching  oz episode you'll hooked. right, exactly
happened me.<br /><br />the thing struck oz brutality unflinching scenes
violence, set right word go. trust me, faint hearted timid. pulls punches
regards drugs, sex violence. hardcore, classic use word.<br /><br />it called oz
nickname given oswald maximum security state penitentary. focuses mainly emerald
city, experimental section prison cells glass fronts face inwards, privacy high
agenda. em city home many..aryans, muslims, gangstas, latinos, christians,
italians, irish more…so scuffles, death stares, dodgy dealings shady
agreements far away.<br /><br />i main appeal fact goes shows wouldn't dare.
forget pretty pictures painted mainstream audiences, forget charm, forget
romance…oz doesn't mess around. episode saw struck nasty surreal, couldn't
ready it, watched more, developed taste oz, got accustomed high levels graphic
violence. violence, injustice (crooked guards who'll sold nickel, inmates who'll
kill order away it, mannered, middle class inmates turned prison bitches lack
street skills prison experience) watching oz, comfortable uncomfortable
viewing…thats touch darker side.

============ After Removing Hyperlinks =======

        reviewers mentioned watching  oz episode you'll hooked. right, exactly
happened me.<br /><br />the thing struck oz brutality unflinching scenes
violence, set right word go. trust me, faint hearted timid. pulls punches
regards drugs, sex violence. hardcore, classic use word.<br /><br />it called oz
```

```
nickname given oswald maximum security state penitentary. focuses mainly emerald
city, experimental section prison cells glass fronts face inwards, privacy high
agenda. em city home many..aryans, muslims, gangstas, latinos, christians,
italians, irish more…so scuffles, death stares, dodgy dealings shady
agreements far away.<br /><br />i main appeal fact goes shows wouldn't dare.
forget pretty pictures painted mainstream audiences, forget charm, forget
romance…oz doesn't mess around. episode saw struck nasty surreal, couldn't
ready it, watched more, developed taste oz, got accustomed high levels graphic
violence. violence, injustice (crooked guards who'll sold nickel, inmates who'll
kill order away it, mannered, middle class inmates turned prison bitches lack
street skills prison experience) watching oz, comfortable uncomfortable
viewing…thats touch darker side.
```

[ ]:

5-7 Tokenizing the text feature: Give a brief to explain Tokenization, tokenization is the process of breaking down a character sequence into smaller units, which is known as tokens, sometimes removing some characters, like punctuation can be considered as tokenization.

It can be noted that the most popular tokenization algorithm is word tokenization. To shed light on this algorithm, it separates a text block into distinct words, and then as a result of this Different word-level tokens are created depending on the delimiters. Here is a tokenization illustration: This is an essential process due to the fact that the meaning of the text can be interpreted through analysis of the words present in the text. .(https://medium.com/analytics-vidhya/spacy-basics-the-importance-of-tokens-in-natural-language-processing-89a698d8a76#:~:text=Before%20processing%20a%20natural%20language%2C%20we%20want%20to,analysis%20

I used NLTK in order to do word tokenization

[47]:
```
# NLTK (Natural Language Toolkit) provides a utility function for tokenizing␣
↪data.
nltk.download()

Moviedf['tokenized_tweets'] = Moviedf["review"].apply(word_tokenize)
Moviedf.head()
```

```
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
```

[47]:

	review	sentiment	\
0	reviewers mentioned watching oz episode you'l…	positive	
1	wonderful little production. the f…	positive	
2	thought wonderful way spend time hot summer we…	positive	
3	basically there's family little boy (jake) thi…	negative	
4	petter mattei's "love time money" visually stu…	positive	

	tokenized_tweets
0	[reviewers, mentioned, watching, oz, episode, …
1	[wonderful, little, production, ., <, br, /, >…

```
2  [thought, wonderful, way, spend, time, hot, su…
3  [basically, there, 's, family, little, boy, (,…
4  [petter, mattei, 's, ``, love, time, money, ''…
```

Tokenizer was successfully applied to our data

5-8 Stemming and lemmatization Both stemming and lemmatizations main aim are to reduce a word's inflectional forms and occasionally related derivational forms to a basic form by removing a portion of a word or distilling a word down to its stem or root.

Stemming

[48]:
```
#
# Creating an instance of the stemmer :
stemmer = PorterStemmer()

## Creating a fucntion that will be applied to our datset :
def Stemmer(text):
    return " ".join([stemmer.stem(word) for word in text])

## Applying the fucntion to all rows :
Moviedf['tokenized_tweets_stemmed'] = Moviedf['tokenized_tweets'].apply(lambda␣
 ↪text: Stemmer(text))
```

[49]:
```
# Checking the results :
Moviedf.head(10)
```

[49]:
```
                                              review sentiment  \
0  reviewers mentioned watching  oz episode you'l…  positive
1  wonderful little production. <br /><br />the f…  positive
2  thought wonderful way spend time hot summer we…  positive
3  basically there's family little boy (jake) thi…  negative
4  petter mattei's "love time money" visually stu…  positive
5  probably all-time favorite movie, story selfle…  positive
6  sure like resurrection dated seahunt series te…  positive
7  amazing, fresh & innovative idea 's aired.   y…  negative
8  encouraged positive comments film looking forw…  negative
9  like original gut wrenching laughter like movi…  positive

                                    tokenized_tweets  \
0  [reviewers, mentioned, watching, oz, episode, …
1  [wonderful, little, production, ., <, br, /, >…
2  [thought, wonderful, way, spend, time, hot, su…
3  [basically, there, 's, family, little, boy, (,…
4  [petter, mattei, 's, ``, love, time, money, ''…
5  [probably, all-time, favorite, movie, ,, story…
6  [sure, like, resurrection, dated, seahunt, ser…
```

```
7  [amazing, ,, fresh, &, innovative, idea, 's, a…
8  [encouraged, positive, comments, film, looking…
9  [like, original, gut, wrenching, laughter, lik…

                          tokenized_tweets_stemmed
0  review mention watch oz episod you 'll hook . …
1  wonder littl product . < br / > < br / > the f…
2  thought wonder way spend time hot summer weeke…
3  basic there 's famili littl boy ( jake ) think…
4  petter mattei 's `` love time money '' visual …
5  probabl all-tim favorit movi , stori selfless …
6  sure like resurrect date seahunt seri tech tod…
7  amaz , fresh & innov idea 's air . year brilli…
8  encourag posit comment film look forward watch…
9  like origin gut wrench laughter like movi . yo…
```

1.5.2 Lemmatizing the text feature:

I imported the libraries we need again

[50]:
```
import sklearn
from sklearn.svm import LinearSVC
from sklearn.naive_bayes import BernoulliNB
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.tree import DecisionTreeClassifier # Import Decision Tree␣
 ↪Classifier
from sklearn.model_selection import train_test_split # Import train_test_split␣
 ↪function
from sklearn import metrics #Import scikit-learn metrics module for accuracy␣
 ↪calculation
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import roc_curve, auc
from nltk.stem import WordNetLemmatizer
```

[51]:
```
# Creating an instance of the limmatizer :
import nltk
#nltk.download('wordnet')

wordnet_lemmatizer = WordNetLemmatizer()

# Applying the limmatizer to all rows:
Moviedf['tokenized_tweets_stemmed_lemmatized'] =␣
 ↪Moviedf['tokenized_tweets_stemmed'].apply(
```

```
    lambda text: wordnet_lemmatizer.lemmatize(text, pos="v"))
```

[53]:
```
Moviedf.head(50)
```

[53]:

	review	sentiment
0	reviewers mentioned watching oz episode you'l…	positive
1	wonderful little production. the f…	positive
2	thought wonderful way spend time hot summer we…	positive
3	basically there's family little boy (jake) thi…	negative
4	petter mattei's "love time money" visually stu…	positive
5	probably all-time favorite movie, story selfle…	positive
6	sure like resurrection dated seahunt series te…	positive
7	amazing, fresh & innovative idea 's aired. y…	negative
8	encouraged positive comments film looking forw…	negative
9	like original gut wrenching laughter like movi…	positive
10	phil alien quirky films humour based oddness a…	negative
11	saw movie came out. recall scariest scene big…	negative
12	im big fan boll's work are. enjoyed movie post…	negative
13	cast played shakespeare. shakespear…	negative
14	fantastic movie prisoners famous. actors georg…	positive
15	kind drawn erotic scenes, realize amateurish u…	negative
16	films simply remade. them. bad film. fails cap…	positive
17	movie awful movies. horrible. the…	negative
18	remember film,it film watched cinema picture d…	positive
19	awful film! real stinkers nominated golden glo…	negative
20	success die hard it's sequels it's surprise s,…	positive
21	terrible misfortune having view "b-movie" it's…	negative
22	absolutely stunning movie, . hrs kill, watch i…	positive
23	all, let's things straight here: a) anime fan-…	negative
24	worst movie saw worldfest received applause af…	negative
25	karen carpenter story shows little singer kare…	positive
26	"the cell" exotic masterpiece, dizzying trip v…	positive
27	film tried things once: stinging political sat…	negative
28	movie frustrating. energetic totally prepared …	negative
29	'war movie' hollywood genre redone times clich…	positive
30	taut organically gripping, edward dmytryk's cr…	positive
31	"ardh satya" finest film indian cinema. direct…	positive
32	exposure templarios & good one. excited title …	negative
33	significant quotes entire film pronounced half…	positive
34	watched film expecting much, got pack films, …	negative
35	bought film blockbuster $., sounded interestin…	negative
36	plot death little children. hopper investigate…	negative
37	watched movie lost plot? well, didn't begin wi…	negative
38	okay, series kind takes route 'here again!' we…	positive
39	sitting pile dung, husband wondered actually p…	negative
40	clichés movies type substance. plot went end m…	negative
41	movie based book, "a splendored thing" han suy…	positive

```
42  films seen, one, rage, got worst yet. directio…  negative
43  heard good things "states grace" came open min…  negative
44  movie struck home me. , remember ''s father wo…  positive
45  disclaimer, i've seen movie - times  years, sa…  positive
46  protocol implausible movie saving grace stars …  negative
47  film classified drama, idea. john voight mary …  negative
48  preston sturgis' power glory unseen public nea…  positive
49  average (and surprisingly tame) fulci giallo m…  negative

                                    tokenized_tweets  \
0   [reviewers, mentioned, watching, oz, episode, …
1   [wonderful, little, production, ., <, br, /, >…
2   [thought, wonderful, way, spend, time, hot, su…
3   [basically, there, 's, family, little, boy, (,…
4   [petter, mattei, 's, ``, love, time, money, ''…
5   [probably, all-time, favorite, movie, ,, story…
6   [sure, like, resurrection, dated, seahunt, ser…
7   [amazing, ,, fresh, &, innovative, idea, 's, a…
8   [encouraged, positive, comments, film, looking…
9   [like, original, gut, wrenching, laughter, lik…
10  [phil, alien, quirky, films, humour, based, od…
11  [saw, movie, came, out, ., recall, scariest, s…
12  [im, big, fan, boll, 's, work, are, ., enjoyed…
13  [cast, played, shakespeare., <, br, /, >, <, b…
14  [fantastic, movie, prisoners, famous, ., actor…
15  [kind, drawn, erotic, scenes, ,, realize, amat…
16  [films, simply, remade, ., them, ., bad, film,…
17  [movie, awful, movies, ., horrible, ., <, br, …
18  [remember, film, ,, it, film, watched, cinema,…
19  [awful, film, !, real, stinkers, nominated, go…
20  [success, die, hard, it, 's, sequels, it, 's, …
21  [terrible, misfortune, having, view, ``, b-mov…
22  [absolutely, stunning, movie, ,, ., hrs, kill,…
23  [all, ,, let, 's, things, straight, here, :, a…
24  [worst, movie, saw, worldfest, received, appla…
25  [karen, carpenter, story, shows, little, singe…
26  [``, the, cell, '', exotic, masterpiece, ,, di…
27  [film, tried, things, once, :, stinging, polit…
28  [movie, frustrating, ., energetic, totally, pr…
29  ['war, movie, ', hollywood, genre, redone, tim…
30  [taut, organically, gripping, ,, edward, dmytr…
31  [``, ardh, satya, '', finest, film, indian, ci…
32  [exposure, templarios, &, good, one, ., excite…
33  [significant, quotes, entire, film, pronounced…
34  [watched, film, expecting, much, ,, got, pack,…
35  [bought, film, blockbuster, $, ., ,, sounded, …
36  [plot, death, little, children, ., hopper, inv…
```

```
37  [watched, movie, lost, plot, ?, well, ,, did, …
38  [okay, ,, series, kind, takes, route, 'here, a…
39  [sitting, pile, dung, ,, husband, wondered, ac…
40  [clichés, movies, type, substance, ., plot, we…
41  [movie, based, book, ,, ``, a, splendored, thi…
42  [films, seen, ,, one, ,, rage, ,, got, worst, …
43  [heard, good, things, ``, states, grace, '', c…
44  [movie, struck, home, me, ., ,, remember, ``, …
45  [disclaimer, ,, i, 've, seen, movie, -, times,…
46  [protocol, implausible, movie, saving, grace, …
47  [film, classified, drama, ,, idea, ., john, vo…
48  [preston, sturgis, ', power, glory, unseen, pu…
49  [average, (, and, surprisingly, tame, ), fulci…

                            tokenized_tweets_stemmed  \
0   review mention watch oz episod you 'll hook . …
1   wonder littl product . < br / > < br / > the f…
2   thought wonder way spend time hot summer weeke…
3   basic there 's famili littl boy ( jake ) think…
4   petter mattei 's `` love time money '' visual …
5   probabl all-tim favorit movi , stori selfless …
6   sure like resurrect date seahunt seri tech tod…
7   amaz , fresh & innov idea 's air . year brilli…
8   encourag posit comment film look forward watch…
9   like origin gut wrench laughter like movi . yo…
10  phil alien quirki film humour base odd actual …
11  saw movi came out . recal scariest scene big b…
12  im big fan boll 's work are . enjoy movi posta…
13  cast play shakespeare. < br / > < br / > shake…
14  fantast movi prison famou . actor georg cloone…
15  kind drawn erot scene , realiz amateurish unbe…
16  film simpli remad . them . bad film . fail cap…
17  movi aw movi . horribl . < br / > < br / > the…
18  rememb film , it film watch cinema pictur dark…
19  aw film ! real stinker nomin golden globe . th…
20  success die hard it 's sequel it 's surpris s …
21  terribl misfortun have view `` b-movi '' it 's…
22  absolut stun movi , . hr kill , watch it , wo …
23  all , let 's thing straight here : a ) anim fa…
24  worst movi saw worldfest receiv applaus afterw…
25  karen carpent stori show littl singer karen ca…
26  `` the cell '' exot masterpiec , dizzi trip va…
27  film tri thing onc : sting polit satir , holly…
28  movi frustrat . energet total prepar good time…
29  'war movi ' hollywood genr redon time clichéd …
30  taut organ grip , edward dmytryk 's crossfir d…
31  `` ardh satya '' finest film indian cinema . d…
```

32	exposur templario & good one . excit titl offe…
33	signific quot entir film pronounc halfway prot…
34	watch film expect much , got pack film , prett…
35	bought film blockbust $. , sound interest (a…
36	plot death littl children . hopper investig ki…
37	watch movi lost plot ? well , did n't begin wi…
38	okay , seri kind take rout 'here again ! ' wee…
39	sit pile dung , husband wonder actual product …
40	cliché movi type substanc . plot went end movi…
41	movi base book , `` a splendor thing '' han su…
42	film seen , one , rage , got worst yet . direc…
43	heard good thing `` state grace '' came open m…
44	movi struck home me . , rememb `` s father wor…
45	disclaim , i 've seen movi - time year , saw m…
46	protocol implaus movi save grace star goldi ha…
47	film classifi drama , idea . john voight mari …
48	preston sturgi ' power glori unseen public nea…
49	averag (and surprisingli tame) fulci giallo …

	tokenized_tweets_stemmed_lemmatized
0	review mention watch oz episod you 'll hook . …
1	wonder littl product . < br / > < br / > the f…
2	thought wonder way spend time hot summer weeke…
3	basic there 's famili littl boy (jake) think…
4	petter mattei 's `` love time money '' visual …
5	probabl all-tim favorit movi , stori selfless …
6	sure like resurrect date seahunt seri tech tod…
7	amaz , fresh & innov idea 's air . year brilli…
8	encourag posit comment film look forward watch…
9	like origin gut wrench laughter like movi . yo…
10	phil alien quirki film humour base odd actual …
11	saw movi came out . recal scariest scene big b…
12	im big fan boll 's work are . enjoy movi posta…
13	cast play shakespeare. < br / > < br / > shake…
14	fantast movi prison famou . actor georg cloone…
15	kind drawn erot scene , realiz amateurish unbe…
16	film simpli remad . them . bad film . fail cap…
17	movi aw movi . horribl . < br / > < br / > the…
18	rememb film , it film watch cinema pictur dark…
19	aw film ! real stinker nomin golden globe . th…
20	success die hard it 's sequel it 's surpris s …
21	terribl misfortun have view `` b-movi '' it 's…
22	absolut stun movi , . hr kill , watch it , wo …
23	all , let 's thing straight here : a) anim fa…
24	worst movi saw worldfest receiv applaus afterw…
25	karen carpent stori show littl singer karen ca…
26	`` the cell '' exot masterpiec , dizzi trip va…

```
27  film tri thing onc : sting polit satir , holly…
28  movi frustrat . energet total prepar good time…
29  'war movi ' hollywood genr redon time clichéd …
30  taut organ grip , edward dmytryk 's crossfir d…
31  `` ardh satya '' finest film indian cinema . d…
32  exposur templario & good one . excit titl offe…
33  signific quot entir film pronounc halfway prot…
34  watch film expect much , got pack film , prett…
35  bought film blockbust $ . , sound interest ( a…
36  plot death littl children . hopper investig ki…
37  watch movi lost plot ? well , did n't begin wi…
38  okay , seri kind take rout 'here again ! ' wee…
39  sit pile dung , husband wonder actual product …
40  cliché movi type substanc . plot went end movi…
41  movi base book , `` a splendor thing '' han su…
42  film seen , one , rage , got worst yet . direc…
43  heard good thing `` state grace '' came open m…
44  movi struck home me . , rememb `` s father wor…
45  disclaim , i 've seen movi - time year , saw m…
46  protocol implaus movi save grace star goldi ha…
47  film classifi drama , idea . john voight mari …
48  preston sturgi ' power glori unseen public nea…
49  averag ( and surprisingli tame ) fulci giallo …
```

After preprocessing our data, we are a step away from using our dataframe for our machine/deep learning models, but before we'll save the new preprocessed dataframe as a csv file that we will use later.

[54]:
```
# Saving our dataframe :
Moviedf.to_csv("cleaned.csv")
```

1.6 6 Data Visualization after Preprocessing :

Before doing the other steps of Machine Learning we need to get the basic accuracy of our Data. To do this, we'll utilize a word cloud, which is a collection of words that are represented in various sizes. The more frequently and prominently a term is used in a certain text, the more significant it is. In our case, we predict that a word cloud will include a sample of terms that correspond to the category that we are plotting. As the main aim of this sentimental analysis is to classify these tweets as negative and positive I would like To determine which words are most frequently used for each type of tweet, we will create word clouds for positive and negative tweets, respectively.

[55]:
```
Moviedf.head(2)
```

[55]:
```
                                              review sentiment  \
0  reviewers mentioned watching  oz episode you'l…  positive
1  wonderful little production. <br /><br />the f…  positive
```

```
                                        tokenized_tweets  \
0  [reviewers, mentioned, watching, oz, episode, …
1  [wonderful, little, production, ., <, br, /, >…

                                tokenized_tweets_stemmed  \
0  review mention watch oz episod you 'll hook . …
1  wonder littl product . < br / > < br / > the f…

                     tokenized_tweets_stemmed_lemmatized
0  review mention watch oz episod you 'll hook . …
1  wonder littl product . < br / > < br / > the f…
```

[56]:
```
text = " ".join(i for i in Moviedf[Moviedf['sentiment']=='positive']['review'])
wordcloud = WordCloud( background_color="white").generate(text)

plt.figure( figsize=(15,10))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for positive review')
plt.show()
```

[57]:
```
text = " ".join(i for i in Moviedf[Moviedf['sentiment']=='negative']['review'])
#stopwords = set(STOPWORDS)
wordcloud = WordCloud( background_color="white").generate(text)
#wordcloud = WordCloud(stopwords=stopwords, background_color="white").
  ↪generate(text)
plt.figure( figsize=(15,10))
```

```
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title('wordcloud for negative review')
plt.show()
```

1.6.1 7 Splitting our data into Train and Test Subset

Creating a new variable df_reduced that contains a shuffled sample of the dataset : I put test size into 0.2 which meand I select 20% my data set to be trained a.

[66]:
```
#splitting into train and test
train, test= train_test_split(Moviedf, test_size=0.2, random_state=42)
X_train, y_train = train['review'], train['sentiment']
X_test, y_test = test['review'], test['sentiment']
```

```
                                                  review sentiment  \
33553  liked summerslam look arena, curtains look ove…  positive
9427   television shows appeal different kinds fans l…  positive
199    film quickly gets major chase scene increasing…  negative
12447  jane austen definitely approve one!<br /><br /…  positive
39489  expectations somewhat high went movie, thought…  negative
…                                                    …       …
28567  casper van dien michael rooker generally releg…  negative
25079  liked movie. wasn't sure started watching it, …  positive
18707  yes non-singaporean's can't what's big deal fi…  positive
15200  far films go, likable enough. entertaining cha…  negative
5857   saw anatomy years ago -- dubbed friends house …  positive
```

```
                                          tokenized_tweets  \
33553  [liked, summerslam, look, arena, ,, curtains, …
9427   [television, shows, appeal, different, kinds, …
199    [film, quickly, gets, major, chase, scene, inc…
12447  [jane, austen, definitely, approve, one, !, <,…
39489  [expectations, somewhat, high, went, movie, ,,…
…                                                     …
28567  [casper, van, dien, michael, rooker, generally…
25079  [liked, movie, ., was, n't, sure, started, wat…
18707  [yes, non-singaporean, 's, ca, n't, what, 's, …
15200  [far, films, go, ,, likable, enough, ., entert…
5857   [saw, anatomy, years, ago, --, dubbed, friends…

                                tokenized_tweets_stemmed  \
33553  like summerslam look arena , curtain look over…
9427   televis show appeal differ kind fan like farsc…
199    film quickli get major chase scene increas des…
12447  jane austen definit approv one ! < br / > < br…
39489  expect somewhat high went movi , thought steve…
…                                                     …
28567  casper van dien michael rooker gener releg b m…
25079  like movi . wa n't sure start watch it , enjoy…
18707  ye non-singaporean 's ca n't what 's big deal …
15200  far film go , likabl enough . entertain charac…
5857   saw anatomi year ago -- dub friend hous do n't…

                     tokenized_tweets_stemmed_lemmatized
33553  like summerslam look arena , curtain look over…
9427   televis show appeal differ kind fan like farsc…
199    film quickli get major chase scene increas des…
12447  jane austen definit approv one ! < br / > < br…
39489  expect somewhat high went movi , thought steve…
…                                                     …
28567  casper van dien michael rooker gener releg b m…
25079  like movi . wa n't sure start watch it , enjoy…
18707  ye non-singaporean 's ca n't what 's big deal …
15200  far film go , likabl enough . entertain charac…
5857   saw anatomi year ago -- dub friend hous do n't…

[10000 rows x 5 columns]
```

1.7 8 Word Embedding and Transforming Dataset using TF-IDF Vectorizer.

[75]:
```
#Vectorizing data
import nltk
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer

tfidf = TfidfVectorizer() #tfidfVectorizer
X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)




count_vect = CountVectorizer() # CountVectorizer
X_train_count = count_vect.fit_transform(X_train)
X_test_count = count_vect.transform(X_test)
```

```
              review
0  I love this movie
1  I hate this movie
```

1.8 9 Function for Model Evaluation.

1.8.1 1. Model 1:LogisticRegression

An algorithm for classifying data is logistic regression. On the basis of a number of independent factors, it is used to forecast a binary outcome.

It is worth mentioning that there are only two outcomes that can occur in a binary situation: either the event occurs (1) or it doesn't (0). Independent variables are those variables or elements that could have an impact on the result (or dependent variable). Therefore, while working with binary data, you should utilize logistic regression as your type of analysis. When the output or dependent variable is dichotomous or categorical in nature, or if it falls into one of two categories (such as "yes" or "no," "pass" or "fail," and so on), you know you are working with binary data. ref:https://careerfoundry.com/en/blog/data-analytics/what-is-logistic-regression/

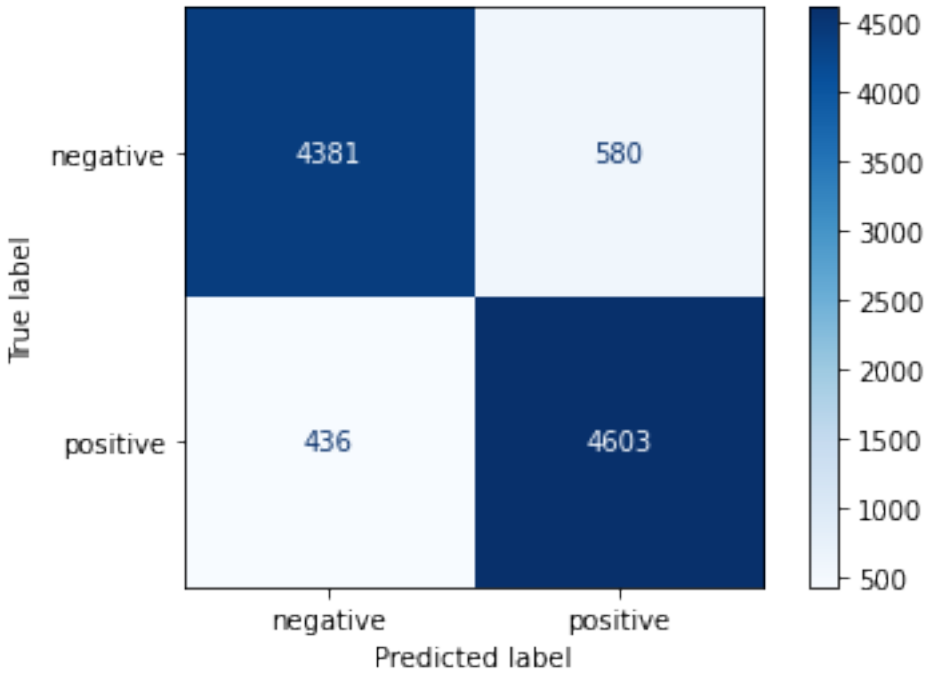
In my notebook in the first step I imported the necessary libraries then train and fit and predict the result with logistic regression, draw a confusion matrix and try to predict a sentence with this algorithm. As the results show this algorithm can predict the sentence as a negative one with 90% accuracy.

[81]:
```
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report
from sklearn.metrics import classification_report, accuracy_score,␣
 ↪confusion_matrix, plot_confusion_matrix, plot_roc_curve,␣
 ↪plot_precision_recall_curve
logicreg = LogisticRegression()
```

```
logicreg.fit(X_train_tfidf, y_train)
pred_logicreg = logicreg.predict(X_test_tfidf)
score_logicreg = accuracy_score(y_test, pred_logicreg)
logicreg_cvScore = cross_val_score(logicreg, X_train_tfidf, y_train, cv=5)
print("Logistic Regression Accuracy: ", "{:.2f}%".format(100*score_logicreg))
print("\nLogistic Regresion cross validation score: ","{:.2f}%".
  ↪format(round(logicreg_cvScore.mean(),2)*100))
print("\n")
plot_confusion_matrix(logicreg, X_test_tfidf, y_test, cmap='Blues')
plt.grid(False)
print("Classification Report:\n")
print(classification_report(pred_logicreg,y_test))


#Sentence Prediction:
Sentence = {"review":["This is a terrible movie"]}
Sdf = pd.DataFrame(Sentence)
Xsdf = tfidf.transform(Sdf)
print(Sdf)
print("Classification Results for the test sentence")
results=logicreg.predict(Xsdf)
print(results)
```

```
Logistic Regression Accuracy:  89.84%

Logistic Regresion cross validation score:  89.00%


Classification Report:


c:\Users\nikva\AppData\Local\Programs\Python\Python39\lib\site-
packages\sklearn\utils\deprecation.py:87: FutureWarning: Function
plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is
deprecated in 1.0 and will be removed in 1.2. Use one of the class methods:
ConfusionMatrixDisplay.from_predictions or
ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
              precision    recall  f1-score   support

    negative       0.88      0.91      0.90      4817
    positive       0.91      0.89      0.90      5183

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000
```

```
                    review
0  This is a terrible movie
Classification Results for the test sentence
['negative']
```



1.8.2 2. Model 2: MultinomialNB

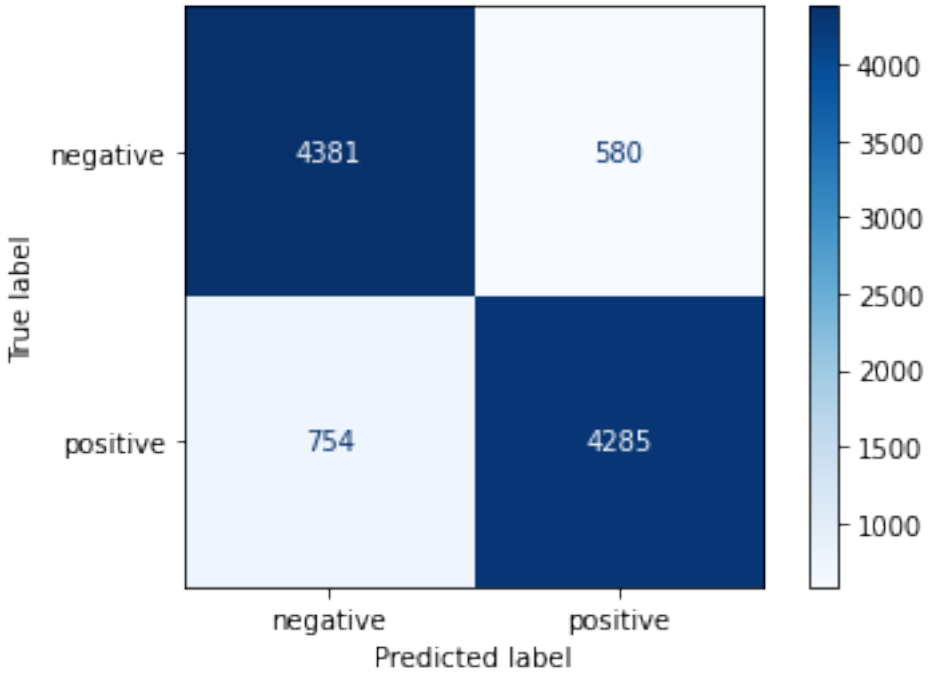
Similar to Bernoulli NB, the multinomial NB classifier is appropriate for discrete features (e.g. word count for text classification). When a multinomial distribution calls for integer feature counts, multinomial NB is employed. However, in actuality, fractional counts like tf—idf may also be effective. Gaussian NB: This technique is helpful when working with continuous data. When data has a Gaussian distribution, it is employed. ref:https://medium.com/@akshayc123/naive-bayes-classifier-nb-7429a1bdb2c0#:~:text=Multinomial%20NB%20is%20used%20for%20multinomial%20distribution%20that,with%2

I define my MultinominalNB as MultiNB in my notebook,I import the necessary libraries,train it,test it and fit it.I got the accuracy of 87% .

[83]:
```
from sklearn.naive_bayes import GaussianNB, MultinomialNB
MultiNB = MultinomialNB()
MultiNB.fit(X_train_tfidf, y_train)
pred_MultiNB = MultiNB.predict(X_test_tfidf)
score_MultiNB = accuracy_score(y_test, pred_MultiNB)
MultiNB_cvScore = cross_val_score(MultiNB, X_train_tfidf, y_train, cv=5)
print("Multinomial Naive Bayes Accuracy: ", "{:.2f}%".format(100*score_MultiNB))
```

```
print("\nMultinomial Naive Bayes cross validation score: ","{:.2f}%".
  ↪format(round(MultiNB_cvScore.mean(),2)*100))
print("\n")
plot_confusion_matrix(MultiNB, X_test_tfidf, y_test, cmap='Blues')
plt.grid(False)
print("Classification Report:\n")
print(classification_report(pred_MultiNB,y_test))


#Sentence Prediction:
Sentence = {"review":["This is a fantastic movie"]}
Sdf = pd.DataFrame(Sentence)
Xsdf = tfidf.transform(Sdf)
print(Sdf)
print("Classification Results for the test sentence")
resultsNB=MultiNB.predict(Xsdf)
print(resultsNB)
```

```
Multinomial Naive Bayes Accuracy:  86.66%

Multinomial Naive Bayes cross validation score:  86.00%


Classification Report:


c:\Users\nikva\AppData\Local\Programs\Python\Python39\lib\site-
packages\sklearn\utils\deprecation.py:87: FutureWarning: Function
plot_confusion_matrix is deprecated; Function `plot_confusion_matrix` is
deprecated in 1.0 and will be removed in 1.2. Use one of the class methods:
ConfusionMatrixDisplay.from_predictions or
ConfusionMatrixDisplay.from_estimator.
  warnings.warn(msg, category=FutureWarning)
              precision    recall  f1-score   support

    negative       0.88      0.85      0.87      5135
    positive       0.85      0.88      0.87      4865

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000

                     review
0  This is a fantastic movie
Classification Results for the test sentence
['negative']
```



1.8.3 3. Model 3: SVC

A common Supervised Learning method known as the Support Vector Machine Algorithm, or SVM, can be used to address classification and regression problems. But it's largely used in machine learning to solve classification problems. The goal of the SVM method is to determine the best decision boundary or line for classifying n-dimensional space into groups so that subsequent data points can be quickly assigned to the appropriate category. ref:https://tutorialforbeginner.com/support-vector-machine-algorithm#:~:text=The%20Support%20Vector%20Machine%20Algorithm%2C%20or%20SVM%2C%20is,mostly%

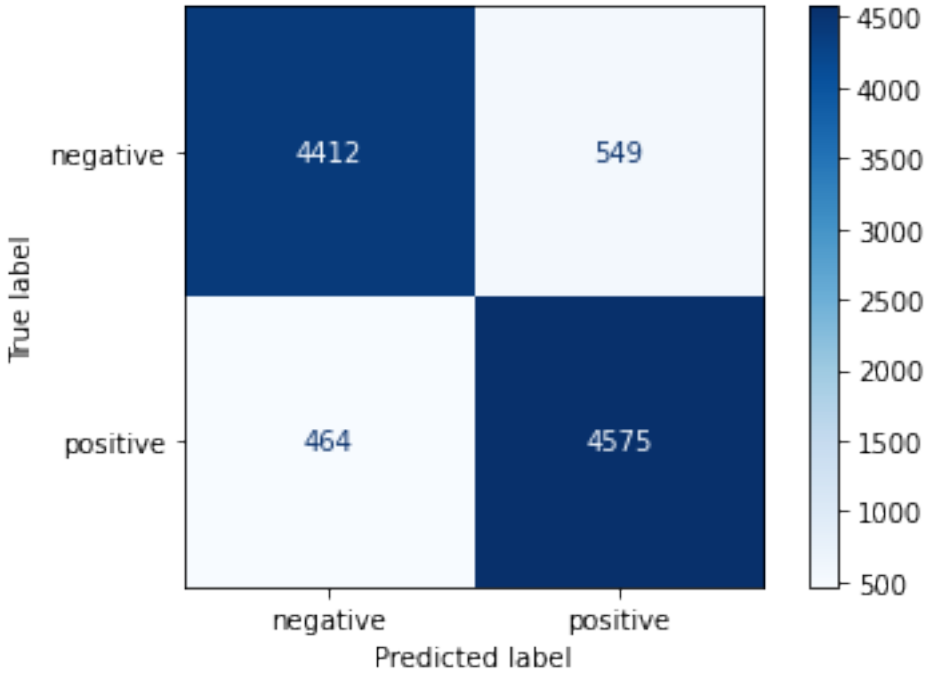
I modify my LinearSVC as Lsvc in my notebook and then try to predict the result with it ,I got 90% accuracy for my dataset.

[63]:
```
from sklearn.svm import LinearSVC
Lsvc = LinearSVC(penalty='l2',loss='hinge')
Lsvc.fit(X_train_tfidf, y_train)
pred_Lsvc = Lsvc.predict(X_test_tfidf)
score_Lsvc = accuracy_score(y_test, pred_Lsvc)
Lsvc_cvScore = cross_val_score(Lsvc, X_train_tfidf, y_train, cv=5)
print("Linear Support Vectpr Classifier Accuracy: ", "{:.2f}%".
  ↪format(100*score_Lsvc))
print("\nLinear Support Vector Classifier cross validation score: ","{:.2f}%".
  ↪format(round(Lsvc_cvScore.mean(),2)*100))
print("\n")
```

```
plot_confusion_matrix(Lsvc, X_test_tfidf, y_test, cmap='Blues')
plt.grid(False)
print("Classification Report:\n")
print(classification_report(pred_Lsvc,y_test))

#Sentence Prediction:
Sentence = {"review":["This is a fantastic movie"]}
Sdf = pd.DataFrame(Sentence)
Xsdf = tfidf.transform(Sdf)
print(Sdf)
print("Classification Results for the test sentence")
resultsLS=Lsvc.predict(Xsdf)
print(resultsLS)
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/svm/_base.py:985:
ConvergenceWarning: Liblinear failed to converge, increase the number of
iterations.
  warnings.warn("Liblinear failed to converge, increase "
```

```
Linear Support Vectpr Classifier Accuracy:  89.87%

Linear Support Vector Classifier cross validation score:  89.00%


Classification Report:

              precision    recall  f1-score   support

    negative       0.89      0.90      0.90      4876
    positive       0.91      0.89      0.90      5124

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000
```



[66]:
```
print(Lsvc.score(X_train_tfidf, y_train))
print(MultiNB.score(X_train_tfidf, y_train))
print(logicreg.score(X_train_tfidf, y_train))
```

```
0.962
0.906475
0.9347
```

[67]:
```
from sklearn.metrics import f1_score

f1_score(y_train,Lsvc.predict(X_train_tfidf),
          labels = ['positive','negative'],average=None)
```

[67]:
```
array([0.96203986, 0.96196006])
```

1.9 1 0 Conclusion

In conclusion, I would like to say that as I mentioned before twitter comments have been so viral and play a vital role in analyzing comments and views about a subject. In my project,t I worked on Movie sentimental analysis, and by these analyses and steps I went through we can conclude that after pre-processing steps and making the data set ready for prediction logistic regression can predict a tweet better than the other two models with higher accuracy and it is predicted that this model can predict other sentences and comments in future

1.10 1 1 Refrence

- https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews)
- https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
- https://medium.com/analytics-vidhya/spacy-basics-the-importance-of-tokens-in-natural-language-processing-89a698d8a76#:~:text=Before%20processing%20a%20natural%20language%2C%20we%
- https://careerfoundry.com/en/blog/data-analytics/what-is-logistic-regression/
- https://medium.com/@akshayc123/naive-bayes-classifier-nb-7429a1bdb2c0#:~:text=Multinomial%20NB%2 https://tutorialforbeginner.com/support-vector-machine--
- https://tutorialforbeginner.com/support-vector-machine-algorithm#:~:text=The%20Support%20Vector%20
- http://ai.stanford.edu/~amaas/data/sentiment
- https://www.educba.com/machine-learning-libraries/
- (https://medium.com/analytics-vidhya/spacy-basics-the-importance-of-tokens-in-natural-language-processing-89a698d8a76#:~:text=Before%20processing%20a%20natural%20language%2C%20we%
- https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews

[ ]: