SENTIMENT CLASSIFICATION WITH CUSTOM NAMED ENTITY RECOGNITION

Problem Statement: To build a Machine Learning model for Custom NER using deep neural network to deal with information extraction from the real-world data and to build a classifier based on polarity of the movie reviews. In order to get insightful information, we approach the machine and deep learning techniques for sentiment analysis and entity recognition over labelled and un processed data.

Build a classifier based on reviews into positive / negative , with use of NER to extract data and provide insights

Data Preparation

Webscaping _ Spacy

```
Data = 'http://www.cs.cornell.edu/people/pabo/movie-review-data/polarity_html.zip'
```

```
%cd '/content/drive/MyDrive/Colab Notebooks/CapstoneGL'
```

```
/content/drive/MyDrive/Colab Notebooks/CapstoneGL
```

```
!pwd
```

```
/content/drive/MyDrive/Colab Notebooks/CapstoneGL
```

```
!unzip polarity_html.zip
```

```
Archive:  polarity_html.zip
```

```
!ls
```

```
23408.html  imdbgbprep.csv     SCCNER-v2.ipynb
26597.html  imdbgbpreplen.csv  sentiment_dictionary.csv
27515.html  movie              word2vec.model
28402.html  polarity_html.zip  word2vec.model.trainables.syn1neg.npy
imdbgb.csv  sample             word2vec.model.wv.vectors.npy
```

```
data_path ="/content/drive/MyDrive/Colab Notebooks/CapstoneGL/movie"
```

```
%cd '/content/drive/MyDrive/Colab Notebooks/CapstoneGL/movie/'
```

```
/content/drive/MyDrive/Colab Notebooks/CapstoneGL/movie
```

```
import sys
sys.getdefaultencoding()
```

```
'utf-8'
```

```
#pip install paddleocr
```

```
import requests
from bs4 import BeautifulSoup
import pandas as pd
import codecs
```

```
import os
directory = os.chdir("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/movie")
#sdirectory = os.chdir("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample")
```

```
!ls
```

```
0002.html   13678.html  18172.html  22542.html  27128.html  5076.html
0003.html   13679.html  18173.html  22543.html  27129.html  5077.html
0004.html   1367.html   18174.html  22544.html  2712.html   5078.html
0005.html   13680.html  18175.html  22545.html  27130.html  5079.html
0006.html   13681.html  18176.html  22546.html  27131.html  5080.html
0007.html   13682.html  18177.html  22547.html  27132.html  5081.html
0008.html   13683.html  18178.html  22548.html  27133.html  5082.html
0009.html   13684.html  18179.html  22549.html  27134.html  5084.html
0010.html   13685.html  1817.html   2254.html   27135.html  5085.html
0011.html   13688.html  18180.html  22550.html  27136.html  5086.html
0012.html   13689.html  18181.html  22552.html  27137.html  5087.html
0013.html   1368.html   18182.html  22555.html  27138.html  5088.html
0014.html   13690.html  18183.html  22557.html  27139.html  5089.html
0016.html   13691.html  18184.html  22558.html  2713.html   5090.html
0017.html   13692.html  18185.html  22559.html  27140.html  5091.html
0018.html   13693.html  18186.html  2255.html   27141.html  5092.html
0019.html   13694.html  18187.html  22560.html  27142.html  5093.html
0020.html   13695.html  18188.html  22561.html  27143.html  5094.html
0021.html   13696.html  18189.html  22562.html  27144.html  5095.html
0022.html   13697.html  1818.html   22563.html  27145.html  5096.html
0023.html   13698.html  18190.html  22564.html  27146.html  5097.html
0024.html   13699.html  18191.html  22565.html  27147.html  5098.html
0025.html   1369.html   18192.html  22566.html  27148.html  5099.html
0026.html   13700.html  18193.html  22567.html  27149.html  5100.html
0027.html   13701.html  18194.html  22568.html  2714.html   5101.html
0028.html   13702.html  18195.html  22569.html  27150.html  5102.html
```

```
0030.html   13703.html  18196.html  2256.html   27151.html  5103.html
0031.html   13704.html  18197.html  22570.html  27152.html  5104.html
0032.html   13705.html  18198.html  22571.html  27153.html  5105.html
0033.html   13706.html  18199.html  22572.html  27154.html  5106.html
0034.html   13708.html  1819.html   22573.html  27155.html  5107.html
0035.html   13709.html  18200.html  22574.html  27156.html  5108.html
0036.html   1370.html   18201.html  22575.html  27157.html  5109.html
0037.html   13710.html  18202.html  22576.html  27158.html  5110.html
0038.html   13711.html  18207.html  22577.html  27159.html  5111.html
0039.html   13712.html  18208.html  22578.html  2715.html   5112.html
0040.html   13713.html  18209.html  22579.html  27160.html  5113.html
0042.html   13715.html  1820.html   2257.html   27161.html  5114.html
0043.html   13717.html  18210.html  22580.html  27162.html  5115.html
0044.html   13718.html  18211.html  22581.html  27163.html  5116.html
0045.html   1371.html   1821.html   22582.html  27164.html  5118.html
0046.html   13720.html  18222.html  22583.html  27165.html  5119.html
0047.html   13721.html  18223.html  22584.html  27166.html  5120.html
0048.html   13722.html  18224.html  22585.html  27167.html  5121.html
0049.html   13723.html  18225.html  22586.html  27168.html  5122.html
0050.html   13725.html  18226.html  22587.html  27169.html  5123.html
0052.html   13726.html  18227.html  22588.html  2716.html   5124.html
0053.html   13727.html  18228.html  22589.html  27170.html  5125.html
0054.html   13728.html  18229.html  2258.html   27171.html  5126.html
0055.html   13729.html  1822.html   22590.html  27172.html  5127.html
0056.html   1372.html   18230.html  22591.html  27173.html  5128.html
0057.html   13730.html  18231.html  22592.html  27174.html  5129.html
0058.html   13731.html  18232.html  22593.html  27175.html  5130.html
0060.html   13732.html  18233.html  22594.html  27176.html  5131.html
0061.html   13733.html  18234.html  22595.html  27177.html  5132.html
0062.html   13734.html  18235.html  22596.html  27178.html  5133.html
0063.html   13735.html  18236.html  22597.html  27179.html  5134.html
0064.html   13738.html  18237.html  22598.html  2717.html   5136.html
0065.html   13739.html  18238.html  22599.html  27180.html  5137.html
```

```
sdirectory = "/content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample"

for filename in os.listdir(sdirectory):
  if filename.endswith('.html'):
    fname = os.path.join(sdirectory, filename)
    print("Current file name ..", os.path.abspath(fname))
    with open(fname, 'r') as file:
      beautifulSoupText = BeautifulSoup(file.read(), 'html.parser')

      for tag in beautifulSoupText.findAll(True):
        print(tag.name, " : ", len(beautifulSoupText.find(tag.name).text))
```

```
br  :  0
a  :  19
hr  :  0
p  :  303
pre  :  153
p  :  303
p  :  303
p  :  303
p  :  303
```

```
pre  :  153
a  :  19
hr  :  0
p  :  303
small  :  525
a  :  19
a  :  19
br  :  0
br  :  0
br  :  0
br  :  0
p  :  303
a  :  19
Current file name .. /content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample/29014.h
html  :  8167
head  :  34
title  :  31
link  :  0
body  :  8132
h1  :  20
a  :  20
h3  :  25
br  :  0
a  :  20
hr  :  0
pre  :  64
p  :  724
p  :  724
p  :  724
p  :  724
p  :  724
p  :  724
p  :  724
pre  :  64
pre  :  64
p  :  724
a  :  20
a  :  20
pre  :  64
hr  :  0
p  :  724
small  :  525
a  :  20
a  :  20
br  :  0
br  :  0
br  :  0
br  :  0
```

```
movies = []
print('Scraping in Progress... ')

directory = "/content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample"

for filename in os.listdir(directory):
```

```
    if filename.endswith('.html'):
      fname = os.path.join(directory, filename)
      print("Current file name ..", os.path.abspath(fname))
      with codecs.open(fname, 'r', encoding="utf-8",errors='ignore') as file:
        soup = BeautifulSoup(file.read(), 'html.parser')
        temp = soup.findAll('html')

        for i in temp:
          d = dict()
          d['Title'] = i.find('h1', class_= 'title').find('a').text
          movies.append(d)

print(len(movies))
print((movies))
```

```
Scraping in Progress...
Current file name .. /content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample/0002.html
Current file name .. /content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample/29014.html
2
[{'Title': 'Hitcher, The (1986)'}, {'Title': 'Soylent Green (1973)'}]
```

```
movies = []
print('Scraping in Progress... ')

directory = "/content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample"

for filename in os.listdir(directory):
  if filename.endswith('.html'):
    fname = os.path.join(directory, filename)
    print("Current file name ..", os.path.abspath(fname))
    with codecs.open(fname, 'r', encoding="utf-8",errors='ignore') as file:
      soup = BeautifulSoup(file.read(), 'html.parser')
      temp = soup.findAll('html')

      for i in temp:
        d = dict()
        try:
          d['Title'] = i.find('h1', class_= 'title').find('a').text.replace('\n', '')
        except AttributeError as e:
          break
        d['reviewed_by'] = i.find('h3', href='').find('a').text.replace('\n', '')
        [p.extract() for p in i.find_all('p',class_=["flush"])]
        [p.extract() for p in i.find_all('p',align=["CENTER"])]
        d['reviews'] = i.find_all('p') #ok

        movies.append(d)

df =  pd.DataFrame(movies)
```

```
Scraping in Progress...
```

```
Current file name .. /content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample/0002.html
Current file name .. /content/drive/MyDrive/Colab Notebooks/CapstoneGL/sample/29014.html
```

```
df.head()
```

	Title	reviewed_by	reviews
0	Hitcher, The (1986)	Mark R. Leeper	[[[Editor's note: Sites running 2.10 netnews w...
1	Soylent Green (1973)	Dragan Antulov	[[Science fiction as a genre was often belittl...

```
df.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgb.csv', index = True)
```

```
df.to_csv('imdbgb.csv', index = True)
```

```
import warnings
warnings.filterwarnings('ignore')
```

```
movies = []
print('Scraping in Progress... ')

directory = os.getcwd()

for filename in os.listdir(directory):
  if filename.endswith('.html'):
    fname = os.path.join(directory, filename)
    print("Current file name ..", os.path.abspath(fname))
    with codecs.open(fname, 'r', encoding="utf-8",errors='ignore') as file:
      soup = BeautifulSoup(file.read(), 'html.parser')
      temp = soup.findAll('html')

      for i in temp:
        d = dict()
        try:
          d['Title'] = i.find('h1', class_= 'title').find('a').text.replace('\n', '')
        except AttributeError as e:
          break
        d['reviewed_by'] = i.find('h3', href='').find('a').text.replace('\n', '')
        [p.extract() for p in i.find_all('p',class_=["flush"])]
        [p.extract() for p in i.find_all('p',align=["CENTER"])]
        d['reviews'] = i.find_all('p') #ok

        movies.append(d)

df =  pd.DataFrame(movies)

print(len(movies))
print((movies))
```

```
print((movies))
```

```
df.head()
```

```
df.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgb.csv', index = True)
```

```
df.to_csv('imdbgb.csv', index = True)
```

Data Preparation Done :)

Data Preprocessing

```
import pandas as pd
df= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgb.csv', encoding='ISO
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2 entries, 0 to 1
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   Unnamed: 0   2 non-null      int64 
 1   Title        2 non-null      object
 2   reviewed_by  2 non-null      object
 3   reviews      2 non-null      object
dtypes: int64(1), object(3)
memory usage: 192.0+ bytes
```

```
df.shape
```

```
(2, 4)
```

```
df.columns
```

```
Index(['Unnamed: 0', 'Title', 'reviewed_by', 'reviews'], dtype='object')
```

```
df.drop('Unnamed: 0', axis=1, inplace=True)
```

```
df.shape
```

```
(2, 3)
```

```
df.columns
```

```
Index(['Title', 'reviewed_by', 'reviews'], dtype='object')
```

```
n = len(pd.unique(df['reviewed_by']))
print("No.of.unique values :",n)
```

```
No.of.unique values : 2
```

```
pd.set_option('display.max_rows', 1610)
```

```
print(df['reviewed_by'].unique())
```

```
['Mark R. Leeper' 'Dragan Antulov']
```

```
df.reviewed_by.unique()
```

```
array(['Mark R. Leeper', 'Dragan Antulov'], dtype=object)
```

```
print(df['reviewed_by'].value_counts())
```

```
Dragan Antulov    1
Mark R. Leeper    1
Name: reviewed_by, dtype: int64
```

```
print(df['reviewed_by'].nunique())
```

```
df.isna().sum()
```

```
Title          0
reviewed_by    0
reviews        0
dtype: int64
```

```
df.isna().sum()/len(df) * 100
```

```
df.dropna(inplace=True)
```

```
df.isna().sum()
```

```
df.isna().sum()/len(df) * 100
```

```
df.shape
```

```
import numpy as np
```

```
df = df.replace(np.NAN,"") # replace the null values with space
```

```
df.shape
```

```
df.describe(include='all')
```

Text Preprocessing

```
df.head()
```

	Title	reviewed_by	reviews
0	Final Fantasy: The Spirits Within (2001)	Evelyn C. Leeper	[<p> CAPSULE: This very dark sci-fi fantasy...
1	Sexy Beast (2000)	Mark R. Leeper	[<p>Roger Ebert asks in his review of SEXY BEA...
2	Final Fantasy: The Spirits Within (2001)	Robin Clifford	[<p>Aliens beings have taken over the Earth. T...
3	Jurassic Park III (2001)	Susan	[<p>Susan Granger's review of "JURASSIC

```
print(df.iloc[0]['reviews'])
```

```
[<p>    CAPSULE: This very dark sci-fi fantasy is magnificent visually
    but it has a nearly incoherent plot.  FINAL FANTASY is a
    Japanese-American co-production entirely animated but with a
    very real three-dimensional look and with very real-looking
    characters.  In the year 2065 aliens that appear to us as
    translucent images, but still very deadly creatures, have
    invaded Earth.  Saving the Earth requires resorting to
    semi-mystical means to understand and halt the enemy.  If this
    film had been done in live-action the scenes more spectacular
    than those of BLADERUNNER would have been hailed as a triumph.
    Rating: 6 (0 to 10), high +1 (-4 to +4)</p>, <p>The art of the animated film cont
at an incredible rate.  It seems that one animated film after
another is released and advances the art of animation.  I
personally was very impressed with the visual images created in
TITAN A.E.  But there are images in FINAL FANTASY that go well
beyond the power of that film's animation.  The one problem is
that if I applaud this film it will have to be mostly on the
imagination of the concepts and on the visuals.  I don't think the
story was a very good one.  And the uncertain terms in which I say
that are intentional.  The telling of the story and the
explanation of what is going on lies somewhere in the range
between terse and incoherent.  I frequently had no idea what was
happening in the plot, thought FINAL FANTASY was never failed to
be an enjoyable film to watch.</p>, <p>The greatest part of what was remarkable about
animation work.  The entire film is done in a three-dimensional
technique.  Every single image is as three-dimensional as a live
```

```
action film.  Of course, I am afraid one could always distinguish
the images from real live action.  And that is (intentionally)
praising the animation with faint criticism.  The computer-
generated images were almost photographic.  And what images they
were!  There were planet-scapes and futuristic battlefields.
There were alien monsters of towering height.  There were things
that cannot be described; they have to be seen.</p>, <p>The story opens in 2065, with
and conquered by a diaphanous life form from space.  Well, not
just one diaphanous life form, but a whole class of gossamer life
forms.  There are things that are insect-like and things that look
like floating dragons.  It is like a whole planet of creatures are
cooperating and taking part in the invasion.  Why?  Dr. Sid
(voiced by Donald Sutherland) and his protege Dr. Aki Ross (Ming-
Na) want to find out.  The creatures seem to burrow into the
ground then attack with deadly potency.  Humans have reacted by
retreating to force-field protected cities.  A guard of power-
suited soldiers protects these cities and what is left of the
human race.  Dr. Sid believes in the Gaia theory that planets are
like a living organism with self-protection mechanisms.  Perhaps
they can be triggered to protect the planet.  But Sid and Aki have
to act fast.  Aki's body has been invaded by one form of the
aliens' essence.  AIDS-like it will prove deadly if the nature of
the aliens is not better understood soon.  Hironobu Sakaguchi, who
is connected with the Final Fantasy video games wrote the story
for this film as well as directed and acted as executive producer.
Jeff Vintar and Al Reinert wrote the screenplay.
Generally in an animated film of this sort, I complain that any
starving actor could have gotten a good job doing the voice of an
animated character.  It usually seems wasteful and useless to give
these voice roles to established and successful actors.  In this
film it really did serve a purpose.  The animation technique makes
the characters realistic and even gives them some marvelous facial
```

```
from lxml import html
from lxml.html.clean import clean_html

tree = html.fromstring("""</p> I kept pineapple expecting to see Dr. Sid with the Sutherland
 <p>This film from Square Pictures (whose logo is a rectangle) is
animated pineapple to be just one step from live action. I rate the film a 6 on the 0
to 10 scale and a high +1 on the -4 to +4 scale.</p>""")

## text only
print(clean_html(tree).text_content().strip())
```

```
I kept pineapple expecting to see Dr. Sid with the Sutherland face.,
 This film from Square Pictures (whose logo is a rectangle) is
animated pineapple to be just one step from live action. I rate the film a 6 on the 0
to 10 scale and a high +1 on the -4 to +4 scale.
```

```
df['reviews'] = df[['reviews']].applymap(lambda s: html.fromstring(s))
```

```
df.head()
```

	Title	reviewed_by	reviews
0	Final Fantasy: The Spirits Within (2001)	Evelyn C. Leeper	[[], [], [], [], [], []]
1	Sexy Beast (2000)	Mark R. Leeper	[[], [], [], [], [], [], [], []]
2	Final Fantasy: The Spirits Within (2001)	Robin Clifford	[[], [], [], [], [], [], [], [], [], []]
3	Jurassic Park III (2001)	Susan Granger	[[], [], []]
4	Final Fantasy: The Spirits Within (2001)	Susan Granger	[[], [], []]

```
df['reviews'] = df[['reviews']].applymap(lambda s: clean_html(s).text_content())
```

```
print(df.iloc[0]['reviews'])
```

```
[    CAPSULE: This very dark sci-fi fantasy is magnificent visually
    but it has a nearly incoherent plot.  FINAL FANTASY is a
    Japanese-American co-production entirely animated but with a
    very real three-dimensional look and with very real-looking
    characters.  In the year 2065 aliens that appear to us as
    translucent images, but still very deadly creatures, have
    invaded Earth.  Saving the Earth requires resorting to
    semi-mystical means to understand and halt the enemy.  If this
    film had been done in live-action the scenes more spectacular
    than those of BLADERUNNER would have been hailed as a triumph.
    Rating: 6 (0 to 10), high +1 (-4 to +4), The art of the animated film continues to
at an incredible rate.  It seems that one animated film after
another is released and advances the art of animation.  I
personally was very impressed with the visual images created in
TITAN A.E.  But there are images in FINAL FANTASY that go well
beyond the power of that film's animation.  The one problem is
that if I applaud this film it will have to be mostly on the
imagination of the concepts and on the visuals.  I don't think the
story was a very good one.  And the uncertain terms in which I say
that are intentional.  The telling of the story and the
explanation of what is going on lies somewhere in the range
between terse and incoherent.  I frequently had no idea what was
happening in the plot, thought FINAL FANTASY was never failed to
be an enjoyable film to watch., The greatest part of what was remarkable about the fi
animation work.  The entire film is done in a three-dimensional
technique.  Every single image is as three-dimensional as a live
action film.  Of course, I am afraid one could always distinguish
the images from real live action.  And that is (intentionally)
praising the animation with faint criticism.  The computer-
generated images were almost photographic.  And what images they
were!  There were planet-scapes and futuristic battlefields.
There were alien monsters of towering height.  There were things
that cannot be described; they have to be seen., The story opens in 2065, with the Ea
and conquered by a diaphanous life form from space.  Well, not
just one diaphanous life form, but a whole class of gossamer life
forms.  There are things that are insect-like and things that look
like floating dragons.  It is like a whole planet of creatures are
cooperating and taking part in the invasion.  Why?  Dr. Sid
(voiced by Donald Sutherland) and his protege Dr. Aki Ross (Ming-
```

```
Na) want to find out.  The creatures seem to burrow into the
ground then attack with deadly potency.  Humans have reacted by
retreating to force-field protected cities.  A guard of power-
suited soldiers protects these cities and what is left of the
human race.  Dr. Sid believes in the Gaia theory that planets are
like a living organism with self-protection mechanisms.  Perhaps
they can be triggered to protect the planet.  But Sid and Aki have
to act fast.  Aki's body has been invaded by one form of the
aliens' essence.  AIDS-like it will prove deadly if the nature of
the aliens is not better understood soon.  Hironobu Sakaguchi, who
is connected with the Final Fantasy video games wrote the story
for this film as well as directed and acted as executive producer.
Jeff Vintar and Al Reinert wrote the screenplay.
Generally in an animated film of this sort, I complain that any
starving actor could have gotten a good job doing the voice of an
animated character.  It usually seems wasteful and useless to give
these voice roles to established and successful actors.  In this
film it really did serve a purpose.  The animation technique makes
the characters realistic and even gives them some marvelous facial
```

```
df.head()
```

	Title	reviewed_by	reviews
0	Final Fantasy: The Spirits Within (2001)	Evelyn C. Leeper	[CAPSULE: This very dark sci-fi fantasy is...
1	Sexy Beast (2000)	Mark R. Leeper	[Roger Ebert asks in his review of SEXY BEAST,...
2	Final Fantasy: The Spirits Within (2001)	Robin Clifford	[Aliens beings have taken over the Earth. The ...
3	Jurassic Park III (2001)	Susan Granger	[Susan Granger's review of "JURASSIC PARK

```
import re;
df= df.applymap(lambda s: s.lower())
df = df.applymap(lambda s: re.sub('[^0-9a-z #+_]','',s))


df = df.applymap(lambda x: x.strip() if isinstance(x, str) else x)


df.head()
```

```
print(df['reviewed_by'].value_counts())
```

```
steve rhodes                    1784
james berardinelli              1396
dennis schwartz                 1046
mark r leeper                    918
scott renshaw                    911
harvey s karten                  896
christopher null                 517
michael dequina                  512
jon popick                       508
susan granger                    455
brian koller                     431
edwin jahiel                     424
berge garabedian                 412
dustin putman                    403
michael j legeros                347
walter frith                     333
edward johnsonott                326
ben hoffman                      325
dragan antulov                   325
jerry saravia                    302
chad polenz                      295
tim voon                         290
james sanford                    290
ted prigge                       274
bob bloom                        271
andrew hicks                     264
ross anthony                     260
david n butterworth              247
james brundage                   246
homer yen                        242
michael redman                   229
pedro sena                       224
matt williams                    222
greg king                        218
steve kong                       212
laura clifford                   203
chuck dowling                    195
nathaniel r atcheson             192
shane burridge                   180
eugene novikov                   179
david sunga                      174
serdar yegulalp                  166
frank maloney                    164
marty mapes                      163
luke buckmaster                  148
jamie peck                       148
brian l johnson                  147
frankie paiva                    147
robin clifford                   142
mark ohara                       139
jamey hughton                    137
jeff meyer                       133
john beachem                     133
```

```
david wilcock                          130
shannon patrick sullivan               120
akiva gottlieb                         119
rose bams cooper                       118
seth bookey                            113
chuck schwartz                         110
```

```
df.shape
```

```
(27867, 3)
```

```
df.head()
```

	Title	reviewed_by	reviews
0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...
1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...
2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...
3	jurassic park iii 2001	susan granger	susan grangers review of jurassic park iii uni...
4	final fantasy the spirits within 2001	susan granger	susan grangers review of final fantasy spirits...

```
#df.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgbprep.csv', index = True)
```

```
rwfc = df['reviews']

from collections import Counter
p = Counter(" ".join(rwfc).split()).most_common(10)
rsltrwf = pd.DataFrame(p, columns=['word','Frequency'])
print(rsltrwf)
```

```
   word  Frequency
0   the     954910
1     a     480344
2    of     446971
3   and     441496
4    to     398791
5    is     323146
6    in     286782
7  that     183949
8    as     148279
9    it     146776
```

```
df.isna().sum()
```

```
Title          0
reviewed_by    0
reviews        0
dtype: int64
```

```
lendf = df
lendf['len'] = lendf['reviews'].apply(lambda x: len(x.split()))
```

```
lendf.head()
```

	Title	reviewed_by	reviews	len
0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...	860
1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...	752
2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...	706
3	jurassic park iii 2001	susan	susan grangers review of jurassic park iii	870

```
#lendf.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgbpreplen.csv', index =
```

```
df.drop('len', axis=1, inplace=True)
```

```
df.head()
```

	Title	reviewed_by	reviews
0	Hitcher, The (1986)	Mark R. Leeper	[<p>[Editor's note: Sites running 2.10 netnews...
1	Soylent Green (1973)	Dragan Antulov	[<p>Science fiction as a genre was often belit...

```
df.shape
```

```
(27867, 3)
```

1st Methos Word2Vec & KMeans Cluster

https://www.kaggle.com/pierremegret/gensim-word2vec-tutorial

Bigrams:

```
from gensim.models.phrases import Phrases, Phraser
```

```
sent = [row.split() for row in df['reviews']]
```

```
phrases = Phrases(sent, min_count=30, progress_per=10000)
```

```
bigram = Phraser(phrases)
```

```
sentences = bigram[sent]
```

Most Frequent Words:

```
from collections import defaultdict
```

```
word_freq = defaultdict(int)
for sent in sentences:
  for i in sent:
    word_freq[i] += 1
len(word_freq)
```

```
834
```

```
sorted(word_freq, key=word_freq.get, reverse=True)[:10]
```

```
['the', 'a', 'and', 'of', 'to', 'is', 'in', 'that', 'it', 'as']
```

Training the model

Gensim Word2Vec Implementation: We use Gensim implementation of word2vec: https://radimrehurek.com/gensim/models/word2vec.html

```
import multiprocessing
from gensim.models import Word2Vec
```

```
cores = multiprocessing.cpu_count() # Count the number of cores in a computer
```

```
print(cores)
```

```
2
```

```
w2v_model = Word2Vec(min_count=20,
                     window=2,
                     size=300,
                     sample=6e-5,
                     alpha=0.03,
                     min_alpha=0.0007,
```

```
                     negative=20,
                     workers=cores-1)
```

Building the Vocabulary Table:

```
from time import time  # To time our operation
```

```
t = time()
w2v_model.build_vocab(sentences, progress_per=10000)
print('Time to build vocab: {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time to build vocab: 0.96 mins
```

Training of the model:

```
w2v_model.train(sentences, total_examples=w2v_model.corpus_count, epochs=30, report_delay=1)
```

```
(230992529, 487783530)
```

```
w2v_model.init_sims(replace=True)
```

```
print(df.iloc[0]['reviews'])
```

```
capsule this very dark scifi fantasy is magnificent visually     but it has a nearly inc
```

```
w2v_model.wv.most_similar(positive=["fantasy"])
```

```
[('fantasies', 0.49274423718452454),
 ('fairy_tale', 0.3920763432979584),
 ('drama', 0.37106359004974365),
 ('adventure', 0.36621296405792236),
 ('dream', 0.3415670692920685),
 ('fiction', 0.3363228440284729),
 ('sciencefiction', 0.33136099576950073),
 ('comedy', 0.330048143863678),
 ('science_fiction', 0.32756027579307556),
 ('escapism', 0.3267386555671692)]
```

```
w2v_model.wv.most_similar(positive=["familiar"])
```

```
[('recognizable', 0.4428706467151642),
 ('unfamiliar_with', 0.4244775176048279),
 ('familiarwith', 0.42044976353645325),
 ('unfamiliar', 0.4111863374710083),
 ('wellworn', 0.3923879861831665),
 ('rehashed', 0.33976733684539795),
```

```
      ('similar', 0.3333616256713867),
      ('interesting', 0.3314129710197449),
      ('wellknown', 0.32752725481987),
      ('alltoofamiliar', 0.3267640769481659)]
```

```
from gensim.models import Word2Vec, KeyedVectors
w2v_model.wv.save_word2vec_format('model.bin', binary=True)
```

```
w2v_model.save("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/word2vec.model")
```

KMeans_clustering

```
import pandas as pd
import numpy as np
from gensim.models import Word2Vec
from sklearn.cluster import KMeans
```

```
word_vectors = Word2Vec.load("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/word2vec.mode
```

```
t = time()
model = KMeans(n_clusters=2, max_iter=1000, random_state=True, n_init=50).fit(X=word_vectors.
print('Time to build vocab: {} mins'.format(round((time() - t) / 60, 2)))
```

```
     Time to build vocab: 0.74 mins
```

```
word_vectors.similar_by_vector(model.cluster_centers_[1], topn=10, restrict_vocab=None)
```

```
     [('hrvatskommovie_reviews', 0.714034914970398),
      ('played_by', 0.6211074590682983),
      ('recenzije_na', 0.58077472448349),
      ('named', 0.5669833421707153),
      ('inhe_can', 0.5497649312019348),
      ('frithhttphomenetinccawfrithmovieshtm', 0.5487039089202881),
      ('orcbloomiquestnet', 0.5481616258621216),
      ('httpwwwcinemareviewcomon_icq', 0.5419092774391174),
      ('at_httpmailyahoocom', 0.5410147905349731),
      ('httpwelcometomrbrowncinemareview_magazine', 0.5407020449638367)]
```

```
positive_cluster_index = 1
positive_cluster_center = model.cluster_centers_[positive_cluster_index]
negative_cluster_center = model.cluster_centers_[1-positive_cluster_index]
```

```
words = pd.DataFrame(word_vectors.vocab.keys())
words.columns = ['words']
words['vectors'] = words.words.apply(lambda x: word_vectors[f'{x}'])
words['cluster'] = words.vectors.apply(lambda x: model.predict([np.array(x)]))
words.cluster = words.cluster.apply(lambda x: x[0])
```

```
words['cluster_value'] = [1 if i==positive_cluster_index else -1 for i in words.cluster]
words['closeness_score'] = words.apply(lambda x: 1/(model.transform([x.vectors]).min()), axis
words['sentiment_coeff'] = words.closeness_score * words.cluster_value
```

```
words.head(10)
```

	words	vectors	cluster	cluster_value	closeness_score	sentiment_coeff
0	capsule	[0.00054656726, 0.050772376, -0.039495766, -0....	0	-1	1.009721	-1.009721
1	this	[-0.01395587, 0.013400842, 0.013489433, 0.0434...	0	-1	1.057334	-1.057334
2	very	[-0.01736871, -0.022394864, -0.046146654, -0.0...	0	-1	1.027217	-1.027217
3	dark	[-0.0710048, -0.055463348, 0.048182193, -0.060...	0	-1	1.021196	-1.021196
4	scifi	[0.017985985, 0.043503035, -0.039599366, -0.04...	0	-1	1.027799	-1.027799
5	fantasy	[-0.012766158, -0.043640926, -0.055291694, -0....	0	-1	1.019102	-1.019102

```
words[['words', 'sentiment_coeff']].to_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL
```

```
words = pd.DataFrame(word_vectors.vocab.keys())
words.columns = ['words']
words['vectors'] = words.words.apply(lambda x: word_vectors.wv[f'{x}'])
words['cluster'] = words.vectors.apply(lambda x: model.predict([np.array(x)]))
words.cluster = words.cluster.apply(lambda x: x[0])
words['cluster_value'] = [1 if i==0 else -1 for i in words.cluster]
words['closeness_score'] = words.apply(lambda x: 1/(model.transform([x.vectors]).min()), axis
words['sentiment_coeff'] = words.closeness_score * words.cluster_value
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: DeprecationWarning: Call
  This is separate from the ipykernel package so we can avoid doing imports until
```

2nd Method Word2vec Gensim

Load Gensim Library

```
import gensim
import logging
logging.basicConfig(format='%(asctime)s : %(levelname)s : %(message)s',
                    level=logging.INFO)
```

Load Text Data

```
df= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgbprep.csv', encoding=
```

```
df.head()
```

	Unnamed: 0	Title	reviewed_by	reviews
0	0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...
1	1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...
2	2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...

```
df.loc[0, 'reviews']
```

```
'capsule this very dark scifi fantasy is magnificent visually     but it has a nearly i
ncoherent plot  final fantasy is a     japaneseamerican coproduction entirely animated
but with a     very real threedimensional look and with very reallooking     characters
in the year 2065 aliens that appear to us as     translucent images but still very dead
ly creatures have     invaded earth  saving the earth requires resorting to     semimys
tical means to understand and halt the enemy  if this     film had been done in liveact
ion the scenes more spectacular     than those of bladerunner would have been hailed as
a triumph    rating 6 0 to 10 high +1 4 to +4 the art of the animated film continues to
evolve before our eyes at an incredible rate  it seems that one animated film after ano
ther is released and advances the art of animation  i personally was very impressed wit
h the visual images created in titan ae  but there are images in final fantasy that go
```

Function to Clean up dat

```
import re, string

def clean_str(string):
```

```
def clean_str(string):
  """
  String cleaning before vectorization
  """
  try:
    string = re.sub(r'^https?:\/\/<>.*[\r\n]*', '', string, flags=re.MULTILINE)
    string = re.sub(r"[^A-Za-z]", " ", string)
    words = string.strip().lower().split()
    words = [w for w in words if len(w)>=1]
    return " ".join(words)
  except:
    return ""
```

Clean the Data using routine above

```
df['clean_reviews'] = df['reviews'].apply(clean_str)
df.head()
```

	Unnamed: 0	Title	reviewed_by	reviews	clean_reviews
0	0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...	capsule this very dark scifi fantasy is magnif...
1	1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...	roger ebert asks in his review of sexy beast w...
2	2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...	aliens beings have taken over the earth the gr...
3	3	jurassic park iii 2001	susan granger	susan grangers review of jurassic park iii uni...	susan grangers review of jurassic park iii uni...
4	4	final fantasy the spirits within 2001	susan granger	susan grangers review of final fantasy spirits...	susan grangers review of final fantasy spirits...

```
df.loc[0, 'clean_reviews']
```

```
'capsule this very dark scifi fantasy is magnificent visually but it has a nearly incoh
erent plot final fantasy is a japaneseamerican coproduction entirely animated but with
a very real threedimensional look and with very reallooking characters in the year alie
ns that appear to us as translucent images but still very deadly creatures have invaded
earth saving the earth requires resorting to semimystical means to understand and halt
the enemy if this film had been done in liveaction the scenes more spectacular than tho
se of bladerunner would have been hailed as a triumph rating to high to the art of the
animated film continues to evolve before our eyes at an incredible rate it seems that o
ne animated film after another is released and advances the art of animation i personal
ly was very impressed with the visual images created in titan ae but there are images i
n final fantasy that go well beyond the power of that films animation the one problem i
```

insights : 2nd method ensures additional spaces are removed rather than 1st Method

Convert 'clean_reviews' to a Word List

```
#List to hold all words in each review
documents = []

#Iterate over each review
for doc in df['clean_reviews']:
  documents.append(doc.split(' '))


print(len(documents))
```

```
27867
```

```
print(documents[0])
```

```
['capsule', 'this', 'very', 'dark', 'scifi', 'fantasy', 'is', 'magnificent', 'visually',
```

Build the Model

```
import warnings
warnings.filterwarnings('ignore')


model = gensim.models.Word2Vec(documents, #Word list
                               min_count=10, #Ignore all words with total frequency lower tha
                               workers=4, #Number of CPU Cores
                               size=50,  #Embedding size
                               window=5, #Neighbours on the left and right
                               iter=10   #Number of iterations over the text corpus
                              )
```

```
2021-06-26 13:24:39,786 : WARNING : consider setting layer size to a multiple of 4 fo
2021-06-26 13:24:39,790 : INFO : collecting all words and their counts
2021-06-26 13:24:39,793 : INFO : PROGRESS: at sentence #0, processed 0 words, keeping
2021-06-26 13:24:41,515 : INFO : PROGRESS: at sentence #10000, processed 6457624 word
2021-06-26 13:24:43,186 : INFO : PROGRESS: at sentence #20000, processed 12615184 wor
2021-06-26 13:24:44,509 : INFO : collected 725324 word types from a corpus of 1728567
2021-06-26 13:24:44,514 : INFO : Loading a fresh vocabulary
2021-06-26 13:24:45,751 : INFO : effective_min_count=10 retains 52256 unique words (7
2021-06-26 13:24:45,753 : INFO : effective_min_count=10 leaves 16236975 word corpus (
2021-06-26 13:24:45,926 : INFO : deleting the raw counts dictionary of 725324 items
2021-06-26 13:24:45,944 : INFO : sample=0.001 downsamples 39 most-common words
2021-06-26 13:24:45,946 : INFO : downsampling leaves estimated 12568082 word corpus (
2021-06-26 13:24:46,163 : INFO : estimated required memory for 52256 words and 50 dim
2021-06-26 13:24:46,165 : INFO : resetting layer weights
2021-06-26 13:24:56,667 : INFO : training model with 4 workers on 52256 vocabulary an
2021-06-26 13:24:57,706 : INFO : EPOCH 1 - PROGRESS: at 4.06% examples, 477735 words/
2021-06-26 13:24:58,748 : INFO : EPOCH 1 - PROGRESS: at 8.07% examples, 488883 words/
```

```
2021-06-26 13:24:59,749 : INFO : EPOCH 1 - PROGRESS: at 11.75% examples, 490805 words
2021-06-26 13:25:00,752 : INFO : EPOCH 1 - PROGRESS: at 15.28% examples, 491291 words
2021-06-26 13:25:01,797 : INFO : EPOCH 1 - PROGRESS: at 18.97% examples, 491250 words
2021-06-26 13:25:02,800 : INFO : EPOCH 1 - PROGRESS: at 22.69% examples, 491919 words
2021-06-26 13:25:03,802 : INFO : EPOCH 1 - PROGRESS: at 26.33% examples, 492289 words
2021-06-26 13:25:04,834 : INFO : EPOCH 1 - PROGRESS: at 30.22% examples, 490742 words
2021-06-26 13:25:05,847 : INFO : EPOCH 1 - PROGRESS: at 34.26% examples, 490348 words
2021-06-26 13:25:06,860 : INFO : EPOCH 1 - PROGRESS: at 38.15% examples, 489912 words
2021-06-26 13:25:07,865 : INFO : EPOCH 1 - PROGRESS: at 42.23% examples, 491681 words
2021-06-26 13:25:08,866 : INFO : EPOCH 1 - PROGRESS: at 46.22% examples, 490440 words
2021-06-26 13:25:09,868 : INFO : EPOCH 1 - PROGRESS: at 50.23% examples, 490741 words
2021-06-26 13:25:10,878 : INFO : EPOCH 1 - PROGRESS: at 54.26% examples, 490008 words
2021-06-26 13:25:11,881 : INFO : EPOCH 1 - PROGRESS: at 58.01% examples, 490696 words
2021-06-26 13:25:12,904 : INFO : EPOCH 1 - PROGRESS: at 61.94% examples, 490595 words
2021-06-26 13:25:13,947 : INFO : EPOCH 1 - PROGRESS: at 65.98% examples, 490475 words
2021-06-26 13:25:14,972 : INFO : EPOCH 1 - PROGRESS: at 69.99% examples, 490572 words
2021-06-26 13:25:15,977 : INFO : EPOCH 1 - PROGRESS: at 74.13% examples, 490345 words
2021-06-26 13:25:16,998 : INFO : EPOCH 1 - PROGRESS: at 78.13% examples, 490151 words
2021-06-26 13:25:18,001 : INFO : EPOCH 1 - PROGRESS: at 82.11% examples, 490295 words
2021-06-26 13:25:19,010 : INFO : EPOCH 1 - PROGRESS: at 86.10% examples, 490281 words
2021-06-26 13:25:20,023 : INFO : EPOCH 1 - PROGRESS: at 90.37% examples, 490598 words
2021-06-26 13:25:21,049 : INFO : EPOCH 1 - PROGRESS: at 94.66% examples, 490651 words
2021-06-26 13:25:22,053 : INFO : EPOCH 1 - PROGRESS: at 99.13% examples, 490998 words
2021-06-26 13:25:22,250 : INFO : worker thread finished; awaiting finish of 3 more th
2021-06-26 13:25:22,260 : INFO : worker thread finished; awaiting finish of 2 more th
2021-06-26 13:25:22,269 : INFO : worker thread finished; awaiting finish of 1 more th
2021-06-26 13:25:22,286 : INFO : worker thread finished; awaiting finish of 0 more th
2021-06-26 13:25:22,287 : INFO : EPOCH - 1 : training on 17285675 raw words (12568423
2021-06-26 13:25:23,310 : INFO : EPOCH 2 - PROGRESS: at 3.88% examples, 470730 words/
2021-06-26 13:25:24,321 : INFO : EPOCH 2 - PROGRESS: at 7.84% examples, 487240 words/
2021-06-26 13:25:25,323 : INFO : EPOCH 2 - PROGRESS: at 11.40% examples, 486615 words
2021-06-26 13:25:26,358 : INFO : EPOCH 2 - PROGRESS: at 14.91% examples, 484743 words
2021-06-26 13:25:27,344 : INFO : EPOCH 2 - PROGRESS: at 18.37% examples, 484491 words
2021-06-26 13:25:28,358 : INFO : EPOCH 2 - PROGRESS: at 22.13% examples, 485362 words
2021-06-26 13:25:29,374 : INFO : EPOCH 2 - PROGRESS: at 25.74% examples, 485711 words
2021-06-26 13:25:30,390 : INFO : EPOCH 2 - PROGRESS: at 29.50% examples, 485139 words
2021-06-26 13:25:31,420 : INFO : EPOCH 2 - PROGRESS: at 33.65% examples, 485361 words
2021-06-26 13:25:32,434 : INFO : EPOCH 2 - PROGRESS: at 37.63% examples, 486610 words
2021-06-26 13:25:33,439 : INFO : EPOCH 2 - PROGRESS: at 41.60% examples, 487483 words
2021-06-26 13:25:34,480 : INFO : EPOCH 2 - PROGRESS: at 45.81% examples, 487260 words
2021-06-26 13:25:35,520 : INFO : EPOCH 2 - PROGRESS: at 49.94% examples, 487537 words
```

Exploring the model

How many words in the model

```
#Model size
model.wv.vectors.shape
```

```
(52256, 50)
```

```
# Vocablury of the model
model.wv.vocab
```

```
model.wv.vocab
```

```
{'capsule': <gensim.models.keyedvectors.Vocab at 0x7ffabe6ed3d0>,
 'this': <gensim.models.keyedvectors.Vocab at 0x7ffabe5faed0>,
 'very': <gensim.models.keyedvectors.Vocab at 0x7ffabe600290>,
 'dark': <gensim.models.keyedvectors.Vocab at 0x7ffabe600510>,
 'scifi': <gensim.models.keyedvectors.Vocab at 0x7ffabe600110>,
 'fantasy': <gensim.models.keyedvectors.Vocab at 0x7ffabe600190>,
 'is': <gensim.models.keyedvectors.Vocab at 0x7ffabe6002d0>,
 'magnificent': <gensim.models.keyedvectors.Vocab at 0x7ffabe600250>,
 'visually': <gensim.models.keyedvectors.Vocab at 0x7ffabe6004d0>,
 'but': <gensim.models.keyedvectors.Vocab at 0x7ffabe600090>,
 'it': <gensim.models.keyedvectors.Vocab at 0x7ffabe590150>,
 'has': <gensim.models.keyedvectors.Vocab at 0x7ffabe590550>,
 'a': <gensim.models.keyedvectors.Vocab at 0x7ffabe590510>,
 'nearly': <gensim.models.keyedvectors.Vocab at 0x7ffabe590790>,
 'incoherent': <gensim.models.keyedvectors.Vocab at 0x7ffabe590210>,
 'plot': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c250>,
 'final': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cb90>,
 'japaneseamerican': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c050>,
 'coproduction': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cb10>,
 'entirely': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c850>,
 'animated': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cf90>,
 'with': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cd50>,
 'real': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c990>,
 'threedimensional': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c090>,
 'look': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cd90>,
 'and': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c150>,
 'characters': <gensim.models.keyedvectors.Vocab at 0x7ffabe59cf10>,
 'in': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c390>,
 'the': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c810>,
 'year': <gensim.models.keyedvectors.Vocab at 0x7ffabe59c410>,
 'aliens': <gensim.models.keyedvectors.Vocab at 0x7ffabe59ccd0>,
 'that': <gensim.models.keyedvectors.Vocab at 0x7ffabe881c10>,
 'appear': <gensim.models.keyedvectors.Vocab at 0x7ffabe881a90>,
 'to': <gensim.models.keyedvectors.Vocab at 0x7ffabe881a50>,
 'us': <gensim.models.keyedvectors.Vocab at 0x7ffabe881f10>,
 'as': <gensim.models.keyedvectors.Vocab at 0x7ffabe881fd0>,
 'translucent': <gensim.models.keyedvectors.Vocab at 0x7ffabe881f90>,
 'images': <gensim.models.keyedvectors.Vocab at 0x7ffabe596d10>,
 'still': <gensim.models.keyedvectors.Vocab at 0x7ffabe596cd0>,
 'deadly': <gensim.models.keyedvectors.Vocab at 0x7ffabe596c50>,
 'creatures': <gensim.models.keyedvectors.Vocab at 0x7ffabe596d90>,
 'have': <gensim.models.keyedvectors.Vocab at 0x7ffabe596e90>,
 'invaded': <gensim.models.keyedvectors.Vocab at 0x7ffabe596fd0>,
 'earth': <gensim.models.keyedvectors.Vocab at 0x7ffabe596b90>,
 'saving': <gensim.models.keyedvectors.Vocab at 0x7ffabe596ed0>,
 'requires': <gensim.models.keyedvectors.Vocab at 0x7ffabe596e50>,
 'resorting': <gensim.models.keyedvectors.Vocab at 0x7ffabe596f90>,
 'means': <gensim.models.keyedvectors.Vocab at 0x7ffabe5963d0>,
 'understand': <gensim.models.keyedvectors.Vocab at 0x7ffabe596b50>,
 'halt': <gensim.models.keyedvectors.Vocab at 0x7ffabe596bd0>,
 'enemy': <gensim.models.keyedvectors.Vocab at 0x7ffabe596b10>,
 'if': <gensim.models.keyedvectors.Vocab at 0x7ffabe596f10>,
 'film': <gensim.models.keyedvectors.Vocab at 0x7ffabe596ad0>,
 'had': <gensim.models.keyedvectors.Vocab at 0x7ffabe596e10>,
 'been': <gensim.models.keyedvectors.Vocab at 0x7ffabe6c5050>,
 'done': <gensim.models.keyedvectors.Vocab at 0x7ffabe6c5090>,
```

```
 'liveaction': <gensim.models.keyedvectors.Vocab at 0x7ffabe6c50d0>,
 'scenes': <gensim.models.keyedvectors.Vocab at 0x7ffabe6c5110>,
 'more': <gensim.models.keyedvectors.Vocab at 0x7ffabe6c5150>,
```

Get an embedding for a word

```
model.wv['corners']
```

```
array([-8.39342996e-02, -4.26457584e-01,  1.81552255e+00, -5.26911259e-01,
       -9.99832153e-03, -3.55538309e-01, -3.66164953e-01,  5.56809902e-01,
        1.36004376e+00,  1.03032553e+00,  2.11882278e-01,  4.12856400e-01,
       -5.68003297e-01, -1.96252203e+00, -5.61692476e-01, -8.65697324e-01,
       -1.25565541e+00, -3.94889563e-01,  1.75256729e+00,  8.68612111e-01,
        1.10259891e+00, -4.23175216e-01, -3.23956788e-01,  7.92814493e-01,
        2.75305480e-01,  1.59892142e+00, -6.17634177e-01, -3.05563897e-01,
       -9.01678264e-01, -4.73075420e-01, -4.05977309e-01, -1.06974137e+00,
       -1.03561807e+00, -1.17415988e+00,  2.42346928e-01,  1.48605490e+00,
        9.26962197e-01,  3.83212984e-01,  2.18699202e-01, -2.37989798e-03,
       -3.60730052e-01, -7.82768190e-01,  1.08098276e-01, -2.45743799e+00,
        3.43371868e-01, -7.20070064e-01,  3.44390690e-01, -2.80719697e-01,
        1.29473209e+00, -1.27883005e+00], dtype=float32)
```

Finding Words which have similar meaning

```
model.wv.most_similar('great', topn=15)
```

```
2021-06-26 13:31:39,267 : INFO : precomputing L2-norms of word weight vectors
[('good', 0.8438466191291809),
 ('wonderful', 0.8068698048591614),
 ('fantastic', 0.7926812171936035),
 ('terrific', 0.7724218964576721),
 ('decent', 0.7537559270858765),
 ('nice', 0.7367268800735474),
 ('agreat', 0.7194309234619141),
 ('fine', 0.7161422967910767),
 ('terrible', 0.7130523324012756),
 ('remarkable', 0.6968520879745483),
 ('topnotch', 0.6830755472183228),
 ('marvelous', 0.6671568155288696),
 ('perfect', 0.6640931963920593),
 ('superb', 0.6625052690505981),
 ('amazing', 0.6588210463523865)]
```

Find the word which is not like others

```
model.doesnt_match("man woman child kitchen".split())
```

```
'kitchen'
```

Saving the model

```
model.save('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/word2vec2ndappropach')
```

```
2021-06-26 13:31:56,386 : INFO : saving Word2Vec object under /content/drive/MyDrive/Col
2021-06-26 13:31:56,389 : INFO : not storing attribute vectors_norm
2021-06-26 13:31:56,391 : INFO : not storing attribute cum_table
2021-06-26 13:31:56,755 : INFO : saved /content/drive/MyDrive/Colab Notebooks/CapstoneGL
```

```
model.save('word2vec-movie-50')
```

```
2021-06-26 13:32:20,678 : INFO : saving Word2Vec object under word2vec-movie-50, separat
2021-06-26 13:32:20,684 : INFO : not storing attribute vectors_norm
2021-06-26 13:32:20,685 : INFO : not storing attribute cum_table
2021-06-26 13:32:21,037 : INFO : saved word2vec-movie-50
```

```
#Load model from memory
model = gensim.models.Word2Vec.load('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/word2v
```

```
2021-06-26 13:33:12,489 : INFO : loading Word2Vec object from /content/drive/MyDrive/Col
2021-06-26 13:33:12,781 : INFO : loading wv recursively from /content/drive/MyDrive/Cola
2021-06-26 13:33:12,783 : INFO : setting ignored attribute vectors_norm to None
2021-06-26 13:33:12,786 : INFO : loading vocabulary recursively from /content/drive/MyDr
2021-06-26 13:33:12,788 : INFO : loading trainables recursively from /content/drive/MyDr
2021-06-26 13:33:12,790 : INFO : setting ignored attribute cum_table to None
2021-06-26 13:33:12,792 : INFO : loaded /content/drive/MyDrive/Colab Notebooks/CapstoneG
```

1.Equation king + man = queen + ?

2.In this case there may not be enough data for this equation

```
model.most_similar(positive=['king','man'], negative=['queen'])
```

```
2021-06-26 13:33:57,436 : INFO : precomputing L2-norms of word weight vectors
[('filmmaker', 0.574756383895874),
 ('master', 0.5427805185317993),
 ('apostle', 0.5274704694747925),
 ('boy', 0.5216406583786011),
 ('ghost', 0.511692225933075),
 ('writer', 0.502508282661438),
 ('director', 0.5004320740699768),
 ('brasco', 0.49977269768714905),
 ('carpenter', 0.4951419234275818),
 ('soldier', 0.4941391944885254)]
```

```
model.wv['king'] + model.wv['man'] - model.wv['queen']
```

```
array([ 2.8859878 , -2.3124592 ,  0.1574229 , -0.8344816 , -3.0550926 ,
```

```
 9.331563  , -1.029568  , -2.0700016 ,  1.9540011 ,  4.6654706 ,
-0.33849454,  2.0218792 ,  2.0902374 , -4.142296  ,  4.60834   ,
 6.2659535 , -3.0033653 , -3.216746  , -4.361799  , -0.28582048,
 1.6542425 ,  1.7576241 , -1.998185  , -4.9835024 ,  1.544697  ,
-2.1282747 , -4.003945  ,  2.9368503 , -1.5102228 , -0.74828696,
 5.5744596 ,  3.574868  ,  1.7522917 , -1.6300452 ,  3.7428474 ,
 0.9770286 ,  0.9006014 , -3.6642966 , -4.0001473 ,  4.563144  ,
 1.3659697 ,  7.2905073 , -0.13535929,  0.529544  ,  0.9734591 ,
-0.40028113,  4.834721  ,  1.1169889 , -0.94091415, -2.0536413 ],
dtype=float32)
```

Rule-Based Sentiment Analysis

Tokenization

https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/

```
from nltk.tokenize import word_tokenize
```

```
import nltk
nltk.download('punkt')
from nltk.corpus import words
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

```
text = "this is to check word tokenize working"
tokens = word_tokenize(text)
tokens
```

```
['this', 'is', 'to', 'check', 'word', 'tokenize', 'working']
```

Enrichment – POS tagging

```
from nltk.tag import pos_tag
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
True
```

```
pos = nltk.pos_tag(tokens)
pos
```

```
[('this', 'DT'),
 ('is', 'VBZ'),
 ('to', 'TO'),
 ('check', 'VB'),
 ('word', 'NN'),
 ('tokenize', 'NN'),
 ('working', 'VBG')]
```

Stopwords removal

```
nltk.download('stopwords')
from nltk.corpus import stopwords
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
```

```
print(stopwords.words('english'))
```

```
['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've",
```

```
text = "this is to check word for an tokenize working and whether it has stop words or not"
tokens = word_tokenize(text)

new_text = (" ").join(ele for ele in tokens if ele.lower() not in stopwords.words('english'))
new_text
```

```
'check word tokenize working whether stop words'
```

Obtaining the stem words

```
# Stemming
from nltk.stem import PorterStemmer

# Lemmatization
from nltk.stem import WordNetLemmatizer
```

```
text = "He glanced up from his computer when she came into his office"
text
```

```
'He glanced up from his computer when she came into his office'
```

Stemming

```
# Instantiate PosterStemmer()
stemmer = PorterStemmer()
```

```
tokens = word_tokenize(text)
stem = []
for ele in tokens:
    if ele.lower() not in stopwords.words('english'):
        stem.append(stemmer.stem(ele))
```

Lemmatization

```
# to map pos tags to wordnet tags
nltk.download('wordnet')
from nltk.corpus import wordnet
```

```
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Unzipping corpora/wordnet.zip.
```

```
# Lemmatization
# POS tagger dictionary
pos_dict = {'J':wordnet.ADJ, 'V':wordnet.VERB, 'N':wordnet.NOUN, 'R':wordnet.ADV}

#Instantiate WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()

lemma = []
pos = pos_tag(word_tokenize(text))
for ele, tag in pos:
    tag = pos_dict.get(tag[0])
    if ele.lower() not in stopwords.words('english'):
        if not tag:
            lemma.append(ele)
        else:
            lemma.append(wordnet_lemmatizer.lemmatize(ele, tag))
```

```
print("Text:", text)
print("Stem:", stem)
print("Lemma:", lemma)
```

```
Text: He glanced up from his computer when she came into his office
Stem: ['glanc', 'comput', 'came', 'offic']
Lemma: ['glance', 'computer', 'come', 'office']
```

```
import pandas as pd
```

```
df= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbgbprep.csv', encoding=
```

```
df.drop('Unnamed: 0', axis=1, inplace=True)
```

```
df.head()
```

	Title	reviewed_by	reviews
0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...
1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...
2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...
3	jurassic park iii 2001	susan granger	susan grangers review of jurassic park iii uni...
4	final fantasy the spirits within 2001	susan granger	susan grangers review of final fantasy spirits...

```
import re, string

def clean_str(string):
  """
  String cleaning before vectorization
  """
  try:
    string = re.sub(r'^https?:\/\/<>.*[\r\n]*', '', string, flags=re.MULTILINE)
    string = re.sub(r"[^A-Za-z]", " ", string)
    words = string.strip().lower().split()
    words = [w for w in words if len(w)>=1]
    return " ".join(words)
  except:
    return ""


df['clean_reviews'] = df['reviews'].apply(clean_str)
df.head()
```

	Title	reviewed_by	reviews	clean_reviews
0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...	capsule this very dark scifi fantasy is magnif...
1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...	roger ebert asks in his review of sexy beast w...
2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...	aliens beings have taken over the earth the gr...
3	jurassic park iii 2001	susan granger	susan grangers review of jurassic park iii uni...	susan grangers review of jurassic park iii uni...
4	final fantasy the spirits within 2001	susan granger	susan grangers review of final fantasy spirits...	susan grangers review of final fantasy spirits...

POS Tag on data

```
from time import time  # to time our operation


# POS tagger dictionary
t = time()

pos_dict = {'J':wordnet.ADJ, 'V':wordnet.VERB, 'N':wordnet.NOUN, 'R':wordnet.ADV}
def token_stop_pos(text):
    tags = pos_tag(word_tokenize(text))
    newlist = []
    for word, tag in tags:
        if word.lower() not in set(stopwords.words('english')):
          newlist.append(tuple([word, pos_dict.get(tag[0])]))
    return newlist

df['POS tagged'] = df['clean_reviews'].apply(token_stop_pos)

print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time taken to build : 50.85 mins
```

```
df.head()
```

	Title	reviewed_by	reviews	clean_reviews	POS tagged
0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...	capsule this very dark scifi fantasy is magnif...	[(capsule, n), (dark, a), (scifi, n), (fantasy...
1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...	roger ebert asks in his review of sexy beast w...	[(roger, n), (ebert, n), (asks, v), (review, n...
2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...	aliens beings have taken over the earth the gr...	[(aliens, n), (beings, n), (taken, v), (earth,...
3	jurassic park iii 2001	susan granger	susan grangers review of jurassic park iii uni...	susan grangers review of jurassic park iii uni...	[(susan, a), (grangers, n), (review, n), (jura...

```
#df.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbLdata.csv', index = True)
```

```
df.loc[0, 'clean_reviews']
```

```
'capsule this very dark scifi fantasy is magnificent visually but it has a nearly incoh
erent plot final fantasy is a japaneseamerican coproduction entirely animated but with
```

```
df.loc[0, 'POS tagged']
```

```
[('capsule', 'n'),
 ('dark', 'a'),
 ('scifi', 'n'),
 ('fantasy', 'n'),
 ('magnificent', 'a'),
 ('visually', 'r'),
 ('nearly', 'r'),
 ('incoherent', 'a'),
 ('plot', 'n'),
 ('final', 'a'),
 ('fantasy', 'n'),
 ('japaneseamerican', 'a'),
 ('coproduction', 'n'),
 ('entirely', 'r'),
 ('animated', 'v'),
 ('real', 'a'),
 ('threedimensional', 'a'),
 ('look', 'n'),
 ('reallooking', 'v'),
 ('characters', 'n'),
 ('year', 'n'),
 ('aliens', 'v'),
 ('appear', 'v'),
 ('us', None),
 ('translucent', 'n'),
 ('images', 'n'),
 ('still', 'r'),
 ('deadly', 'r'),
 ('creatures', 'n'),
 ('invaded', 'v'),
 ('earth', 'n'),
 ('saving', 'v'),
 ('earth', 'n'),
 ('requires', 'v'),
 ('resorting', 'v'),
 ('semimystical', 'a'),
 ('means', 'n'),
 ('understand', 'v'),
 ('halt', 'v'),
 ('enemy', 'n'),
 ('film', 'n'),
 ('done', 'v'),
 ('liveaction', 'n'),
 ('scenes', 'n'),
 ('spectacular', 'a'),
 ('bladerunner', 'n'),
 ('would', None),
 ('hailed', 'v'),
 ('triumph', 'n'),
 ('rating', 'n'),
 ('high', 'v'),
 ('art', 'n'),
```

```
      ('animated', 'a'),
      ('film', 'n'),
      ('continues', 'v'),
      ('evolve', 'v'),
      ('eyes', 'n'),
      ('incredible', 'a'),
      ('rate', 'n'),
```

Obtaining the stem words – Lemmatization

```
t = time()
from nltk.stem import WordNetLemmatizer
wordnet_lemmatizer = WordNetLemmatizer()

def lemmatize(pos_data):
    lemma_rew = " "
    for word, pos in pos_data:
        if not pos:
            lemma = word
            lemma_rew = lemma_rew + " " + lemma
        else:
            lemma = wordnet_lemmatizer.lemmatize(word, pos=pos)
            lemma_rew = lemma_rew + " " + lemma
    return lemma_rew

df['Lemma'] = df['POS tagged'].apply(lemmatize)

print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))
```

```
 Time taken to build : 0.8 mins
```

```
df.head()
```

Title	reviewed_by	reviews	clean_reviews	POS tagged	Lemma

```
df.loc[0, 'Lemma']
```

```
'  capsule dark scifi fantasy magnificent visually nearly incoherent plot final fantasy japaneseamerican coproduction entirely animate real threedimensional look reallooking character year alien appear us translucent image still deadly creature invade earth save earth require resort semimystical mean understand halt enemy film do liveaction scene spectacular bladerunner would hail triumph rating high art animated film continue evolve eye incredible rate seem one animate film another release advance art animation personally impressed visual image create titan ae image final fantasy go well beyond power film animation one problem applaud film mostly imagination concept visuals dont think story good one uncertain term say intentional telling story explanation go lie somewhere range terse incoherent frequently idea happen plot think final fantasy never fail enjoyable film watch great part remarkable film animation work entire film do threedimensional
```

				gr...	gr...	(earth,...	
3	jurassic park	susan	review of	review of	(grangers, n),		review jurassic

```
#df.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbLdata.csv', index = True)
```

```
df= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbLdata.csv')
```

```
df.head()
```

	Unnamed: 0	Title	reviewed_by	reviews	clean_reviews	POS tagged	Lemma
0	0	final fantasy the spirits within 2001	evelyn c leeper	capsule this very dark scifi fantasy is magnif...	capsule this very dark scifi fantasy is magnif...	[('capsule', 'n'), ('dark', 'a'), ('scifi', 'n...	capsule dark scifi fantasy magnificent visua...
1	1	sexy beast 2000	mark r leeper	roger ebert asks in his review of sexy beast w...	roger ebert asks in his review of sexy beast w...	[('roger', 'n'), ('ebert', 'n'), ('asks', 'v')...	roger ebert ask review sexy beast would gues...
2	2	final fantasy the spirits within 2001	robin clifford	aliens beings have taken over the earth the gr...	aliens beings have taken over the earth the gr...	[('aliens', 'n'), ('beings', 'n'), ('taken', '...	alien being take earth great city desert and...
		jurassic		susan	susan grangers	[('susan',	susan granger

```
df.drop('Unnamed: 0', axis=1, inplace=True)
```

```
df.drop(['Title','reviewed_by'], axis=1, inplace=True)
```

```
df.head()
```

	reviews	clean_reviews	POS tagged	Lemma
0	capsule this very dark scifi fantasy is magnif...	capsule this very dark scifi fantasy is magnif...	[('capsule', 'n'), ('dark', 'a'), ('scifi', 'n...	capsule dark scifi fantasy magnificent visua...
1	roger ebert asks in his review of sexy beast w...	roger ebert asks in his review of sexy beast w...	[('roger', 'n'), ('ebert', 'n'), ('asks', 'v')...	roger ebert ask review sexy beast would gues...
2	aliens beings have taken over the earth the gr...	aliens beings have taken over the earth the gr...	[('aliens', 'n'), ('beings', 'n'), ('taken', '...	alien being take earth great city desert and...
3	susan grangers review of jurassic park iii uni...	susan grangers review of jurassic park iii uni...	[('susan', 'a'), ('grangers', 'n'), ('review',...	susan granger review jurassic park iii unive...

```
df[['reviews','Lemma']]
```

	reviews	Lemma
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...
1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...
2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...
3	susan grangers review of jurassic park iii uni...	susan granger review jurassic park iii unive...
4	susan grangers review of final fantasy spirits...	susan granger review final fantasy spirit wi...
...	...	...
27862	ay carmela is a film by carlos saura and stars...	ay carmela film carlos saura star spains fir...
27863	synopsisin 1920s china an old man the owner of...	synopsisin china old man owner dye factory b...
27864	closet land is a movie written and directed by...	closet land movie write direct radha bharadw...
27865	an artist painting a picture has the option of...	artist paint picture option reproduce exactl...
27866	i wrote this title after seeing the doors in e...	write title see door europe documentaryof or...

```
df.shape
```

```
(27867, 4)
```

Sentiment Analysis using TextBlob:

```
from textblob import TextBlob
res = TextBlob("I love horror films")
res
```

```
TextBlob("I love horror films")
```

```
from textblob import TextBlob

# function to calculate subjectivity
def getSubjectivity(review):
    return TextBlob(review).sentiment.subjectivity

# function to calculate polarity
def getPolarity(review):
    return TextBlob(review).sentiment.polarity

# function to analyze the reviews
def analysis(score):
    if score < 0:
        return 'Negative'
    elif score == 0:
        return 'Neutral'
    else:
        return 'Positive'


fin_data = pd.DataFrame(df[['reviews','Lemma']])


# fin_data['Subjectivity'] = fin_data['Lemma'].apply(getSubjectivity)
#t = time()
fin_data['Polarity'] = fin_data['Lemma'].apply(getPolarity)
fin_data['Analysis'] = fin_data['Polarity'].apply(analysis)
#print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))


fin_data.dtypes
```

```
reviews      object
Lemma        object
Polarity    float64
Analysis     object
dtype: object
```

```
fin_data.head()
```

	reviews	Lemma	Polarity	Analysis

```
fin_data.sample()
```

	reviews	Lemma	Polarity	Analysis
25166	somewhere between bigbudget action films and s...	somewhere bigbudget action film smallbudgetm...	0.0925	Neutral

Count the number of positive, negative, neutral reviews.

```
tb_counts = fin_data.Analysis.value_counts()

tb_counts
```

```
Neutral     27856
Positive       11
Name: Analysis, dtype: int64
```

```
import matplotlib.pyplot as plt
%matplotlib inline

tb_count= fin_data.Analysis.value_counts()
plt.figure(figsize=(10, 7))
plt.pie(tb_counts.values, labels = tb_counts.index, explode = (0, 0, 0.25), autopct='%1.1f%%'
# plt.legend()
```

```
([<matplotlib.patches.Wedge at 0x7efbcb730a50>,
  <matplotlib.patches.Wedge at 0x7efbcb753910>,
  <matplotlib.patches.Wedge at 0x7efbcb71f410>],
 [Text(-1.0294569226326988, 0.3875802425867622, 'Positive'),
  Text(1.0275217244958899, -0.3926819396012413, 'Negative'),
  Text(1.3499833913558095, -0.006696496357586787, 'Neutral')],
 [Text(-0.5615219577996539, 0.21140740504732483, '88.5%'),
  Text(0.5604663951795762, -0.21419014887340432, '11.3%'),
```

Sentiment Analysis using VADER

```
pip install vaderSentiment==3.3.2
```

```
Collecting vaderSentiment==3.3.2
  Downloading https://files.pythonhosted.org/packages/76/fc/310e16254683c1ed35eeb9738698
     |████████████████████████████████| 133kB 5.1MB/s
Requirement already satisfied: requests in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1 in /usr/local/lib
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.7/dist-packag
Installing collected packages: vaderSentiment
Successfully installed vaderSentiment-3.3.2
```

```
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
analyzer = SentimentIntensityAnalyzer()

# function to calculate vader sentiment
def vadersentimentanalysis(review):
    vs = analyzer.polarity_scores(review)
    return vs['compound']

fin_data['Vader Sentiment'] = fin_data['Lemma'].apply(vadersentimentanalysis)


# function to analyse
def vader_analysis(compound):
    if compound >= 0.5:
        return 'Positive'
    elif compound <= -0.5 :
        return 'Negative'
    else:
        return 'Neutral'
t = time()
fin_data['Vader Analysis'] = fin_data['Vader Sentiment'].apply(vader_analysis)
print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time taken to build : 0.0 mins
```

```
fin_data.head()
```

	reviews	Lemma	Polarity	Analysis	Vader Sentiment	Vader Analysis
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	Positive
1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	Positive
2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	Positive
3	susan grangers review of jurassic park iii uni...	susan granger review jurassic park iii unive...	-0.028190	Negative	0.8074	Positive
4	susan grangers review	susan granger review	0.056250	Positive	-0.1027	Neutral

```
vader_counts = fin_data['Vader Analysis'].value_counts()
vader_counts
```

```
Positive    20732
Negative     5340
Neutral      1795
Name: Vader Analysis, dtype: int64
```

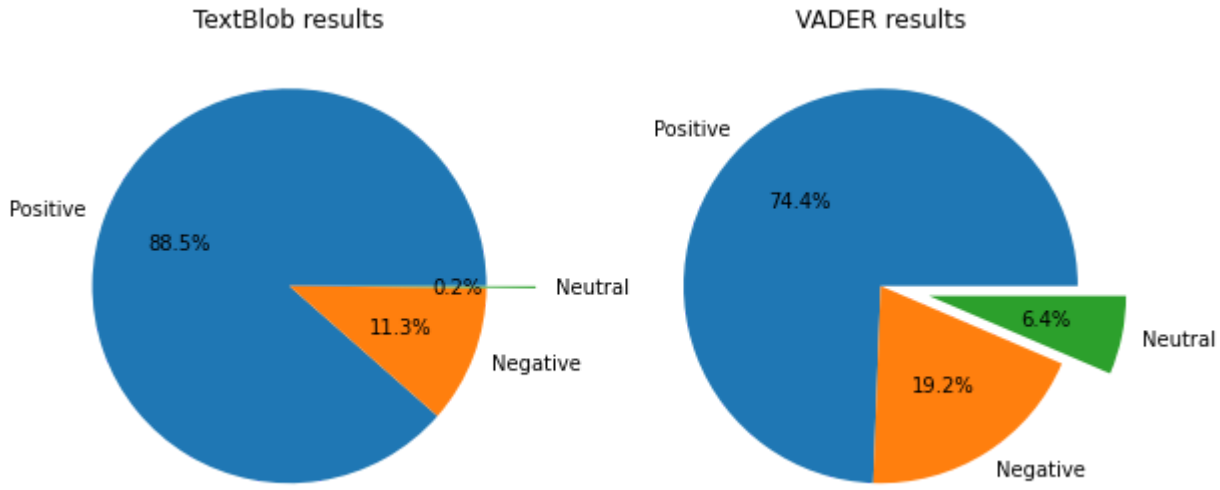
```
vader_counts= fin_data['Vader Analysis'].value_counts()
plt.figure(figsize=(10, 7))
plt.pie(vader_counts.values, labels = vader_counts.index, explode = (0.1, 0, 0), autopct='%1.
# plt.legend()
```

```
([<matplotlib.patches.Wedge at 0x7efbcb8d9810>,
  <matplotlib.patches.Wedge at 0x7efbcba6c750>,
  <matplotlib.patches.Wedge at 0x7efbcba938d0>],
 [Text(-0.8322818776440963, 0.8644691296658416, 'Positive'),
  Text(0.5880938999866443, -0.929594301186544, 'Negative'),
  Text(1.0775545753583702, -0.22107948146366485, 'Neutral')],
 [Text(-0.4854977619590562, 0.5042736589717408, '74.4%'),
  Text(0.32077849090180593, -0.5070514370108421, '19.2%'),
  Text(0.5877570411045655, -0.1205888080710899, '6.4%')])
```

Visual representation of TextBlob, VADER, SentiWordNet results

```
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(15,7))
plt.subplot(1,3,1)
plt.title("TextBlob results")
plt.pie(tb_counts.values, labels = tb_counts.index, explode = (0, 0, 0.25), autopct='%1.1f%%'
plt.subplot(1,3,2)
plt.title("VADER results")
plt.pie(vader_counts.values, labels = vader_counts.index, explode = (0, 0, 0.25), autopct='%1
```

```
([<matplotlib.patches.Wedge at 0x7efbcb8dd910>,
  <matplotlib.patches.Wedge at 0x7efbcba41450>,
  <matplotlib.patches.Wedge at 0x7efbcba41310>],
 [Text(-0.7629250545070884, 0.7924300355270214, 'Positive'),
  Text(0.5880938999866443, -0.929594301186544, 'Negative'),
  Text(1.3224533424852725, -0.2713248181599523, 'Neutral')],
 [Text(-0.41614093882204817, 0.4322345648329207, '74.4%'),
  Text(0.32077849090180593, -0.5070514370108421, '19.2%'),
  Text(0.8326558082314678, -0.17083414476737735, '6.4%')])
```




```
fin_data.head()
```

	reviews	Lemma	Polarity	Analysis	Vader Sentiment	Vader Analysis
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	Positive
1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	Positive
2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	Positive
3	susan grangers review of jurassic park iii uni...	susan granger review jurassic park iii unive...	-0.028190	Negative	0.8074	Positive

```
fin_data.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbdatawithlabelsfromvad
```

```
sadf= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbdatawithlabelsfromva
```

```
sadf.drop('Unnamed: 0', axis=1, inplace=True)
```

```
sadf.head()
```

	reviews	Lemma	Polarity	Analysis	Vader Sentiment	Vader Analysis
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	Positive
1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	Positive
2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	Positive
3	susan grangers review of jurassic park iii uni...	susan granger review jurassic park iii unive...	-0.028190	Negative	0.8074	Positive
4	susan grangers review	susan granger review	0.056250	Positive	-0.1027	Neutral

```
sadf.rename(columns={'Polarity': 'TextBlobPolarity', 'Analysis': 'TextBlobSentiment', 'Vader
```

```
sadf.head()
```

	reviews	Lemma	TextBlobPolarity	TextBlobSentiment	VaderPolarity	VaderSentime
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	Posit
1	roger ebert asks in his review of sexy beast	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	Posit

```
sadf.loc[3, 'reviews']
```

```
'susan grangers review of jurassic park iii universal pictures   this third installment
continues the story of a paleontologist dralan grant sam neill who eight years ago acce
pted an invitation from ingenindustrialist john hammond to preview a new tourist attrac
tion featuringgenetically engineered dinosaurs on an island near costa rica after barel
yescaping from that nightmare he vowed never to return  but he didnt count onhis resear
ch money becoming extinct so when a wealthy adventurer william hmacy and his exwife tea
leoni offer to fund his new theory on velociraptorintelligence if he will accompany the
m on an aerial tour of isla sorna a newingen site that has become both a dinosaur breed
ing ground and a magnet forthrillseekers he agrees to go bringing his gungho protg ales
sandronivola but when their plane unexpectedly lands he discovers that the couples14 ye
arold son trevor morgan is lost in the dense jungle in a paraglidingaccident meanwhile
```

```
sadf.loc[3, 'Lemma']
```

```
'  susan granger review jurassic park iii universal picture third installment continue
story paleontologist dralan grant sam neill eight year ago accept invitation ingenindus
trialist john hammond preview new tourist attraction featuringgenetically engineer dino
saur island near costa rica barelyescaping nightmare vow never return didnt count onhis
research money become extinct wealthy adventurer william hmacy exwife tea leoni offer f
und new theory velociraptorintelligence accompany aerial tour isla sorna newingen site
become dinosaur breeding ground magnet forthrillseekers agree go bring gungho protg ale
ssandronivola plane unexpectedly land discover couple yearold son trevor morgan lose de
nse jungle paraglidingaccident meanwhile must fight survival attack byrampaging reptile
particularly massive menacing spinosaurus brieflybattles tyrannosaurus rex flock fly pt
eranodons ofcourse wily velociraptor dr grant still crusty curmudgeon thebrutal behemot
```

```
sadf.dtypes
```

```
reviews              object
Lemma                object
TextBlobPolarity    float64
TextBlobSentiment    object
VaderPolarity       float64
VaderSentiment       object
dtype: object
```

```
sadf.describe()
```

	TextBlobPolarity	VaderPolarity
count	27867.000000	27867.000000
mean	0.098207	0.539097
std	0.084974	0.743563
min	-0.385937	-0.999500
25%	0.044561	0.429600
50%	0.098217	0.970700
75%	0.151744	0.992400
max	0.633535	0.999900

```
sadf.isnull().sum()
```

```
reviews             26
Lemma                0
TextBlobPolarity     0
TextBlobSentiment    0
VaderPolarity        0
VaderSentiment       0
dtype: int64
```

```
sadf.isna().sum()/len(sadf) * 100
```

```
reviews             0.0933
Lemma               0.0000
TextBlobPolarity    0.0000
TextBlobSentiment   0.0000
VaderPolarity       0.0000
VaderSentiment      0.0000
dtype: float64
```

```
sadf.dropna(inplace=True)
```

```
sadf.isnull().sum()
```

```
reviews             0
Lemma               0
TextBlobPolarity    0
TextBlobSentiment   0
VaderPolarity       0
VaderSentiment      0
dtype: int64
```

```
tb_counts = sadf['TextBlobSentiment'].value_counts()
tb_counts
```

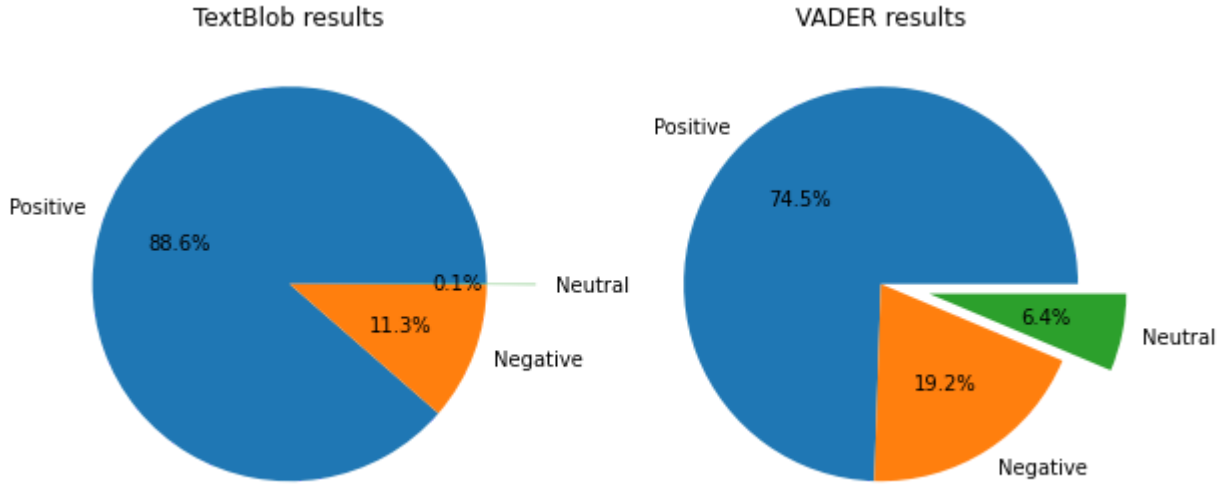
```
Positive    24673
Negative     3150
Neutral        18
Name: TextBlobSentiment, dtype: int64
```

```
vader_counts = sadf['VaderSentiment'].value_counts()
vader_counts
```

```
Positive    20732
Negative     5340
Neutral      1769
Name: VaderSentiment, dtype: int64
```

```
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(15,7))
plt.subplot(1,3,1)
plt.title("TextBlob results")
plt.pie(tb_counts.values, labels = tb_counts.index, explode = (0, 0, 0.25), autopct='%1.1f%%'

plt.subplot(1,3,2)
plt.title("VADER results")
plt.pie(vader_counts.values, labels = vader_counts.index, explode = (0, 0, 0.25), autopct='%1
```

```
([<matplotlib.patches.Wedge at 0x7f0747201e10>,
  <matplotlib.patches.Wedge at 0x7f07472103d0>,
  <matplotlib.patches.Wedge at 0x7f0747210d50>],
 [Text(-0.7646528150604763, 0.7907629685437281, 'Positive'),
  Text(0.5926672583412124, -0.9266852329083004, 'Negative'),
  Text(1.3231930758452324, -0.26769401195251474, 'Neutral')],
 [Text(-0.41708335366935073, 0.4313252555693062, '74.5%'),
  Text(0.3232730500042976, -0.5054646724954365, '19.2%'),
  Text(0.833121566272924, -0.16854808159973148, '6.4%')])
```




```
sadf.to_csv(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbsadsgb.csv', index = True
```

Testing - obtainging the best polarity

```
a = [['0.115716', '0.9955'], [ '0.8074','-0.028190'], ['-0.056250', '-0.1027'], ['-0.56250',
dumdf = pd.DataFrame(a, columns=['x', 'y'])
```

```
dumdf
```

	x	y
0	0.115716	0.9955
1	0.8074	-0.028190
2	-0.056250	-0.1027
3	-0.56250	-0.028190

```
conditions = [
    (dumdf['x'] >= dumdf['y']),
    dumdf['x'] < dumdf['y']]

choices = [dumdf['x'], dumdf['y']]

dumdf['Best'] = np.select(conditions, choices, default=np.nan)
```

```
dumdf
```

	x	y	Best
0	0.115716	0.9955	0.9955
1	0.8074	-0.028190	0.8074
2	-0.056250	-0.1027	-0.1027
3	-0.56250	-0.028190	-0.56250

```
dumdf.dtypes
```

```
x       object
y       object
Best    object
dtype: object
```

```
dumdf['Best'] = dumdf['Best'].astype(float)
```

```
dumdf.dtypes
```

```
x        object
y        object
Best    float64
dtype: object
```

```
def analysis(score):
  if score >= 0.5:
    return 'Positive'
  elif score <= -0.5:
    return 'Negative'
  else:
    return 'Neutral'
```

```
dumdf['Analysis'] = dumdf['Best'].apply(analysis)
```

```
dumdf
```

	x	y	Best	Analysis
0	0.115716	0.9955	0.9955	Positive
1	0.8074	-0.028190	0.8074	Positive
2	-0.056250	-0.1027	-0.1027	Neutral
3	-0.56250	-0.028190	-0.5625	Negative

Selection of Best Polarities from TextBlob & Vader

```
import numpy as np
import pandas as pd
from time import time
```

```
sadf2= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbsadsgb.csv')
```

```
sadf2.head()
```

	Unnamed: 0	reviews	Lemma	TextBlobPolarity	TextBlobSentiment	VaderPolarity	Va
0	0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	
1	1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	
2	2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	
		susan	susan				
		jurassic	jurassic		Negative		

```
sadf2.drop('Unnamed: 0', axis=1, inplace=True)
```

```
conditions = [
    (sadf2['TextBlobPolarity'] >= sadf2['VaderPolarity']),
    sadf2['TextBlobPolarity'] < sadf2['VaderPolarity']]

choices = [sadf2['TextBlobPolarity'], sadf2['VaderPolarity']]
t = time()
sadf2['BestPolarity'] = np.select(conditions, choices, default=np.nan)
print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time taken to build : 0.0 mins
```

```
def analysis(score):
  if 0.5 <= score <= 1:
    return 'Positive'
  elif -0.5 <= score <= 0.5:
    return 'Neutral'
  else:
    return 'Negative'
t = time()
sadf2['OptimisedSentiment'] = sadf2['BestPolarity'].apply(analysis)
print('Time taken to build : {} mins'.format(round((time() - t) / 60, 2)))
```

```
Time taken to build : 0.0 mins
```

```
sadf2.head()
```

	reviews	Lemma	TextBlobPolarity	TextBlobSentiment	VaderPolarity	VaderSentime
0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	Posit
1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	Posit
2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	Posit
3	susan grangers review of jurassic park iii uni...	susan granger review jurassic park iii unive...	-0.028190	Negative	0.8074	Posit
4	susan grangers review of final fantasy spirits...	susan granger review final fantasy spirit wi...	0.056250	Positive	-0.1027	Neu

```
tb_counts = sadf2['TextBlobSentiment'].value_counts()
tb_counts
```

```
Positive    24673
Negative     3150
Neutral        18
Name: TextBlobSentiment, dtype: int64
```

```
vader_counts = sadf2['VaderSentiment'].value_counts()
```

```
vader_counts
```

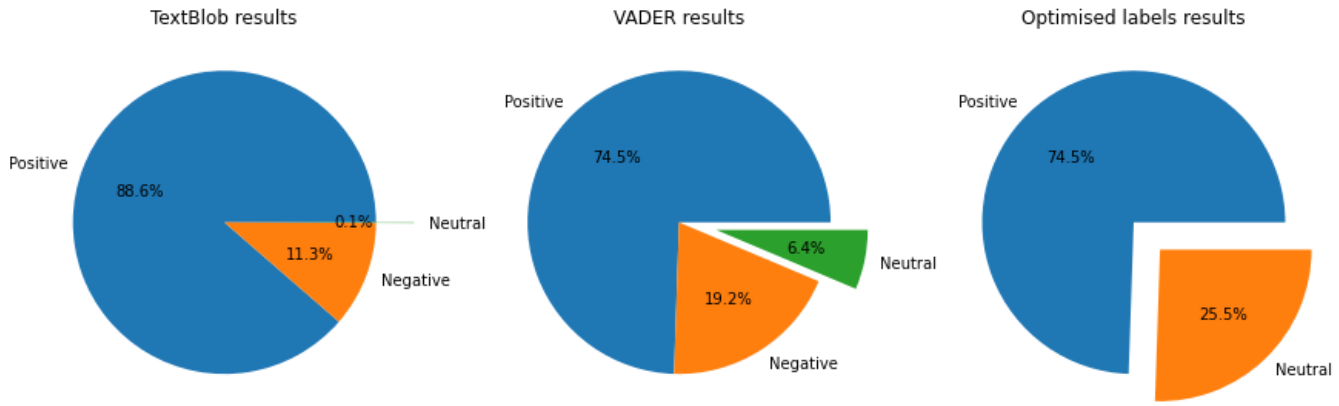
```
Positive    20732
Negative     5340
Neutral      1769
Name: VaderSentiment, dtype: int64
```

```
os_counts = sadf2['OptimisedSentiment'].value_counts()
os_counts
```

```
Positive    20733
Neutral      7108
Name: OptimisedSentiment, dtype: int64
```

```
import matplotlib.pyplot as plt
%matplotlib inline

plt.figure(figsize=(15,7))

plt.subplot(1,3,1)
plt.title("TextBlob results")
plt.pie(tb_counts.values, labels = tb_counts.index, explode = (0, 0, 0.25), autopct='%1.1f%%'

plt.subplot(1,3,2)
plt.title("VADER results")
plt.pie(vader_counts.values, labels = vader_counts.index, explode = (0, 0, 0.25), autopct='%1

plt.subplot(1,3,3)
plt.title("Optimised labels results")
plt.pie(os_counts.values, labels = os_counts.index, explode = (0, 0.25), autopct='%1.1f%%', s
```

```
([<matplotlib.patches.Wedge at 0x7f30f02e8450>,
  <matplotlib.patches.Wedge at 0x7f30f02f1250>],
 [Text(-0.7647420982920643, 0.7906766235951654, 'Positive'),
  Text(0.9385472114843095, -0.9703757683573032, 'Neutral')],
 [Text(-0.4171320536138532, 0.4312781583246356, '74.5%'),
  Text(0.5909371331567875, -0.610977335632376, '25.5%')])
```



```
sadf2.loc[5695, 'reviews']
```

```
'father of the bride director vincente minnelli screenwriters francesgoodrichalbert hac
kettfrom a novel by edward streetercinamatographer john alton editor ferris webster cas
t spencer tracystanley banks joan bennett ellie banks elizabeth taylor kaybanks don tay
lor buckley dunstan billie burke doris dunstanmoroni olsen herbert dunstan leo g carrol
l mr massoula mariettacanty delilah tom irish ben banks paul harvey rev aigalsworthy ru
ss tamblyn tommy banks  1950 this is one of the great dark comedies of the 1950s the sa
tire evolvesfrom the nightmare a middleaged suburban father has when his beautiful20yea
rold daughter his only daughter and the one he is most partialto of his three children
announces to him that she intends to getmarried the crises for the father becomes one o
f his own making as heis jealous of the groom fearful that he has lost his daughter to
astranger overly concerned about the high cost of the wedding and isinsecure about gett
```

Insight:

Seems obatang the best Polarity among TextBlob & Vader doesn't provide good labels. So sitcking back only to VADER Labels

Sentiment Clasification

For faster execution connect the runtime to GPU # Change Runtime Type & below

snipet to avoid session time out on colab on inspect element console step1

function ClickConnect() { console.log('Working') document .querySelector('#top-toolbar > colab-connect-button') .shadowRoot.querySelector('#connect') .click() }

step 2 setInterval(ClickConnect, 60000)

```
!pip install ktrain
```

```
#Import libraries

import numpy as np
import pandas as pd
import tensorflow as tf
import seaborn as sns
import ktrain
from ktrain import text
from sklearn.feature_extraction.text import CountVectorizer
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
```

```
from keras.utils.np_utils import to_categorical
import re

scdf= pd.read_csv('/content/drive/MyDrive/Colab Notebooks/CapstoneGL/imdbsadsgb.csv')

scdf.head()
```

	Unnamed: 0	reviews	Lemma	TextBlobPolarity	TextBlobSentiment	VaderPolarity	Va
0	0	capsule this very dark scifi fantasy is magnif...	capsule dark scifi fantasy magnificent visua...	0.115716	Positive	0.9955	
1	1	roger ebert asks in his review of sexy beast w...	roger ebert ask review sexy beast would gues...	0.079123	Positive	0.9929	
2	2	aliens beings have taken over the earth the gr...	alien being take earth great city desert and...	0.061461	Positive	0.9857	
3	3	susan grangers review of jurassic	susan granger review jurassic	-0.028190	Negative	0.8074	

```
scdf.drop(['Unnamed: 0','Lemma','TextBlobPolarity','TextBlobSentiment'], axis=1, inplace=True

scdf.rename(columns={'VaderPolarity': 'Polarity', 'VaderSentiment': 'Sentiment'}, inplace=Tru

scdf.head()
```

	reviews	Polarity	Sentiment
0	capsule this very dark scifi fantasy is magnif	0.9955	Positive

```
scdf.tail()
```

	reviews	Polarity	Sentiment
27836	ay carmela is a film by carlos saura and stars...	-0.0863	Neutral
27837	synopsisin 1920s china an old man the owner of...	0.9829	Positive
27838	closet land is a movie written and directed by...	0.9897	Positive
27839	an artist painting a picture has the option of...	-0.9697	Negative
27840	i wrote this title after seeing the doors in e...	0.9884	Positive

```
s_counts = scdf['Sentiment'].value_counts()
s_counts
```

```
Positive    20732
Negative     5340
Neutral      1769
Name: Sentiment, dtype: int64
```

```
TRAIN_SIZE = 20000
TEST_SIZE = 7840

data_train = scdf[:TRAIN_SIZE]
data_test = scdf[TRAIN_SIZE:].reset_index(drop=True)
```

```
data_train.head()
```

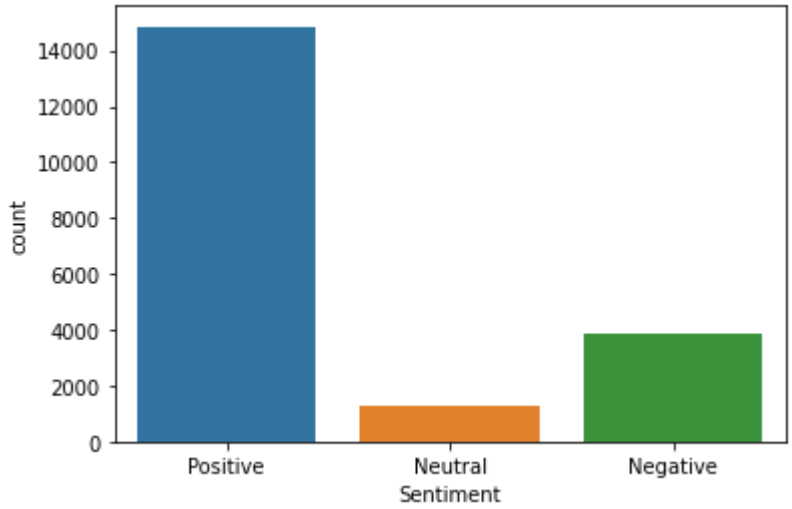
	reviews	Polarity	Sentiment
0	capsule this very dark scifi fantasy is magnif...	0.9955	Positive
1	roger ebert asks in his review of sexy beast w...	0.9929	Positive
2	aliens beings have taken over the earth the gr...	0.9857	Positive
3	susan grangers review of jurassic park iii uni...	0.8074	Positive
4	susan grangers review of final fantasy spirits...	-0.1027	Neutral

```
data_train['Sentiment'].value_counts()
```

```
Positive    14856
Negative     3884
Neutral      1260
Name: Sentiment, dtype: int64
```

```
sns.countplot(data_train["Sentiment"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa11662e4d0>
```




```
data_train.isna().sum()/len(data_train) * 100
```

```
reviews      0.0
Polarity     0.0
Sentiment    0.0
dtype: float64
```

```
data_test.head()
```

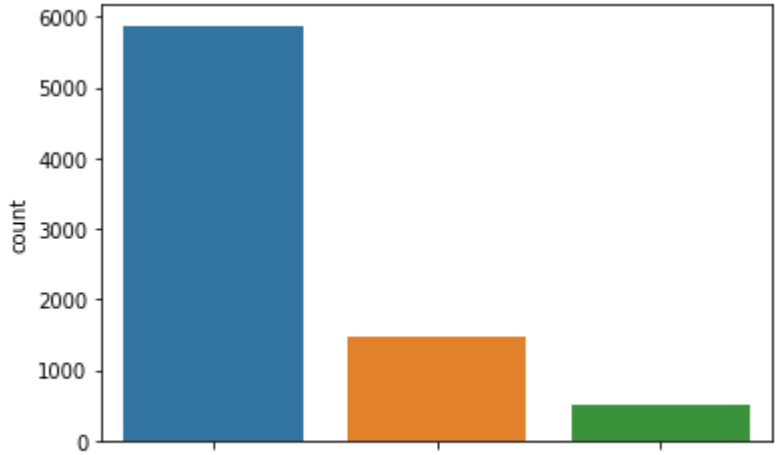
	reviews	Polarity	Sentiment
0	the fact that the film is based on a true stor...	0.9521	Positive
1	when director noyce undertook sic this version...	0.9360	Positive
2	a twentieth century foxfilm corporation releas...	0.9986	Positive
3	the honeymoon is a short one for young america...	0.9970	Positive
4	films live or die on the strength of their scr...	0.9986	Positive

```
data_test['Sentiment'].value_counts()
```

```
Positive    5876
Negative    1456
Neutral      509
Name: Sentiment, dtype: int64
```

```
sns.countplot(data_test["Sentiment"])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa115ebc590>
```




```
data_test.isna().sum()/len(data_test) * 100
```

```
reviews      0.0
Polarity     0.0
Sentiment    0.0
dtype: float64
```

```
#dimension of the dataset

print("Size of train dataset: ",data_train.shape)
print("Size of test dataset: ",data_test.shape)
```

```
Size of train dataset:  (20000, 3)
Size of test dataset:  (7841, 3)
```

```
max_features = 10000
embedding_size = 50


# maxlen means it is considering that much words and rest are getting trucated
# preprocess_mode means tokenizing, embedding and transformation of text corpus(here it is co


(X_train, y_train), (X_test, y_test), preproc = text.texts_from_df(train_df=data_train,
                                                                   text_column = 'reviews',
                                                                   label_columns = 'Sentiment
                                                                   val_df = data_test,
                                                                   maxlen = 500,
                                                                   ngram_range=2,
                                                                   preprocess_mode = 'bert')
```

```
['Negative', 'Neutral', 'Positive']
   Negative  Neutral  Positive
0       0.0      0.0       1.0
1       0.0      0.0       1.0
2       0.0      0.0       1.0
3       0.0      0.0       1.0
4       0.0      1.0       0.0
['Negative', 'Neutral', 'Positive']
   Negative  Neutral  Positive
0       0.0      0.0       1.0
1       0.0      0.0       1.0
2       0.0      0.0       1.0
3       0.0      0.0       1.0
4       0.0      0.0       1.0
downloading pretrained BERT model (uncased_L-12_H-768_A-12.zip)...
[████████████████████████████████████████████████████████████]
extracting pretrained BERT model...
done.

cleanup downloaded zip...
done.

preprocessing train...
```

```
len(X_train[1])
```

```
20000
language: en
```

```
X_train[0].shape
```

```
(20000, 500)
```

```
print('review: \n', X_train[0])
print('label: \n', y_train[0])
```

```
review:
 [[  101 18269  2023 ...  2046  1996   102]
 [  101  5074 22660 ... 19104  1037   102]
 [  101 12114  9552 ... 24951 23805   102]
 ...
 [  101  3313  2136 ...  2802  1996   102]
 [  101  9617  8663 ... 18743  2015   102]
 [  101  2122  2024 ...  4948  7556   102]]
label:
 [0. 0. 1.]
```

BERT Model Building

```
# name = "bert" means, here we are using BERT model.

model = text.text_classifier(name = 'bert',
                             train_data = (X_train, y_train),
                             preproc = preproc)
```

```
Is Multi-Label? False
maxlen is 500
done.
```

```
model.summary()
```

```
_________________________________________________________________
Encoder-10-MultiHeadSelfAttenti (None, 500, 768)     0           Encoder-9-FeedForwar
                                                                 Encoder-10-MultiHead
_________________________________________________________________
Encoder-10-MultiHeadSelfAttenti (None, 500, 768)     1536        Encoder-10-MultiHead
_________________________________________________________________
Encoder-10-FeedForward (FeedFor (None, 500, 768)     4722432     Encoder-10-MultiHead
_________________________________________________________________
Encoder-10-FeedForward-Dropout  (None, 500, 768)     0           Encoder-10-FeedForwa
_________________________________________________________________
Encoder-10-FeedForward-Add (Add (None, 500, 768)     0           Encoder-10-MultiHead
                                                                 Encoder-10-FeedForwa
_________________________________________________________________
Encoder-10-FeedForward-Norm (La (None, 500, 768)     1536        Encoder-10-FeedForwa
_________________________________________________________________
Encoder-11-MultiHeadSelfAttenti (None, 500, 768)     2362368     Encoder-10-FeedForwa
_________________________________________________________________
Encoder-11-MultiHeadSelfAttenti (None, 500, 768)     0           Encoder-11-MultiHead
_________________________________________________________________
Encoder-11-MultiHeadSelfAttenti (None, 500, 768)     0           Encoder-10-FeedForwa
                                                                 Encoder-11-MultiHead
_________________________________________________________________
Encoder-11-MultiHeadSelfAttenti (None, 500, 768)     1536        Encoder-11-MultiHead
_________________________________________________________________
Encoder-11-FeedForward (FeedFor (None, 500, 768)     4722432     Encoder-11-MultiHead
_________________________________________________________________
Encoder-11-FeedForward-Dropout  (None, 500, 768)     0           Encoder-11-FeedForwa
_________________________________________________________________
Encoder-11-FeedForward-Add (Add (None, 500, 768)     0           Encoder-11-MultiHead
                                                                 Encoder-11-FeedForwa
_________________________________________________________________
Encoder-11-FeedForward-Norm (La (None, 500, 768)     1536        Encoder-11-FeedForwa
_________________________________________________________________
Encoder-12-MultiHeadSelfAttenti (None, 500, 768)     2362368     Encoder-11-FeedForwa
_________________________________________________________________
Encoder-12-MultiHeadSelfAttenti (None, 500, 768)     0           Encoder-12-MultiHead
_________________________________________________________________
Encoder-12-MultiHeadSelfAttenti (None, 500, 768)     0           Encoder-11-FeedForwa
                                                                 Encoder-12-MultiHead
_________________________________________________________________
Encoder-12-MultiHeadSelfAttenti (None, 500, 768)     1536        Encoder-12-MultiHead
_________________________________________________________________
Encoder-12-FeedForward (FeedFor (None, 500, 768)     4722432     Encoder-12-MultiHead
_________________________________________________________________
Encoder-12-FeedForward-Dropout  (None, 500, 768)     0           Encoder-12-FeedForwa
_________________________________________________________________
Encoder-12-FeedForward-Add (Add (None, 500, 768)     0           Encoder-12-MultiHead
                                                                 Encoder-12-FeedForwa
_________________________________________________________________
```

```
Encoder-12-FeedForward-Norm (La (None, 500, 768)     1536        Encoder-12-FeedForwa
__________________________________________________________________________________
Extract (Extract)               (None, 768)          0           Encoder-12-FeedForwa
__________________________________________________________________________________
NSP-Dense (Dense)               (None, 768)          590592      Extract[0][0]
__________________________________________________________________________________
dense (Dense)                   (None, 3)            2307        NSP-Dense[0][0]
==================================================================================
Total params: 109,475,331
```

```
#here we have taken batch size as 6 as from the documentation it is recommend to use this wit

learner = ktrain.get_learner(model=model, train_data=(X_train, y_train),
                  val_data = (X_test, y_test),
                  batch_size = 6)


# To find the best lr, use below code, takes a day to train
#learner.lr_find()
#learner.lr_plot()
```

Fitting The Model

```
#Essentially fit is a very basic training loop, where as fit one cycle uses the one cycle pol

learner.fit_onecycle(lr = 2e-5, epochs = 1)
```

```
begin training using onecycle policy with max lr of 2e-05...
3334/3334 [==============================] - 3727s 1s/step - loss: 0.5209 - accuracy: 0
<tensorflow.python.keras.callbacks.History at 0x7f1b59d74f10>
```

Saving Model

```
predictor = ktrain.get_predictor(learner.model, preproc)
predictor.save("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/Model")
```

```
/usr/local/lib/python3.7/dist-packages/tensorflow/python/keras/utils/generic_utils.py:49
  category=CustomMaskWarning)
```

```
scdf.loc[4, 'Sentiment']
```

```
'Neutral'
```

```
#sample dataset to test on
```

```
data = ['movie was half good watchable but not great','this movie was horrible, the plot was
        'the fild is really sucked. there is not plot and acting was bad',
        'what a beautiful movie. great plot. acting was good. will see it again',]


data= ['The villain is wasted worse than. Walton Goggin character in Ant Man 2 and Karl Urban


data = ['A little disappointed I was hoping for a more serious spy and deserved more, espiona


predictor.predict(data)
```

```
['Positive', 'Negative', 'Negative', 'Positive']
```

```
#return_proba = True means it will give the prediction probabilty for each class

predictor.predict(data, return_proba=True)
```

```
array([[0.02747514, 0.07506157, 0.89746326],
       [0.80216634, 0.08307405, 0.11475966],
       [0.90544456, 0.04071458, 0.05384094],
       [0.0032457 , 0.00654709, 0.99020725]], dtype=float32)
```

```
#classes available

predictor.get_classes()
```

```
['Negative', 'Neutral', 'Positive']
```

SCPrediction

```
!pip install ktrain


#Import libraries

import numpy as np
import pandas as pd
import tensorflow as tf
import seaborn as sns
import ktrain
from ktrain import text
from sklearn.feature_extraction.text import CountVectorizer
from keras.preprocessing.text import Tokenizer
from keras.preprocessing.sequence import pad_sequences
from keras.models import Sequential
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from sklearn.model_selection import train_test_split
```

```
from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
import re


import os
os.chdir(r'/content/drive/MyDrive/Colab Notebooks/CapstoneGL/Model')


os.listdir()
```

```
['tf_model.h5', 'tf_model.preproc']
```

Model File Size

```
for file in os.listdir():
    print(f"{file}: {round(os.path.getsize(file)/1e+6,2)} MB")
```

```
tf_model.h5: 1314.45 MB
tf_model.preproc: 1.08 MB
```

Load the model

```
#loading the model

predictor_load = ktrain.load_predictor("/content/drive/MyDrive/Colab Notebooks/CapstoneGL/Mod


predictor_load.get_classes()
```

```
['Negative', 'Neutral', 'Positive']
```

```
#sample dataset to test on

data = ['The public went berserk for “Psycho” in 1960, but critics were not as crazy about Al
        'movie was half good watchable but not great','this movie was horrible, the plot was
        'the fild is really sucked. there is not plot and acting was bad',
        'what a beautiful movie. great plot. acting was good. will see it again',]


predictor_load.predict(data)
```

```
['Negative', 'Positive', 'Negative', 'Negative', 'Positive']
```

```
new_data = ["this movie is shit, feels like i have wasted my time", "best movie i have seen",
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred) == 0:
        print(f"{new_data[i]} => {pred} => negative")
```

```
        print(f"{new_data[i]} => {pred} => negative")
    else:
        print(f"{new_data[i]} => {pred}=> positive")
```

```
 this movie is shit, feels like i have wasted my time => [0.8843597  0.04596698 0.069673
 best movie i have seen => [0.02159161 0.0292159  0.94919246]=> positive
 you are a kind man => [0.01797667 0.02001    0.9620133 ]=> positive
```

```
new_data = ["this movie is shit, feels like i have wasted my time", "best movie i have seen",
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred) -1 <= -0.5:
        print(f"{new_data[i]} => {pred} => negative")
    elif np.argmax(pred) -0.5 <= 0.5:
        print(f"{new_data[i]} => {pred} => neutral")
    else:
        print(f"{new_data[i]} => {pred}=> positive")
```

```
 this movie is shit, feels like i have wasted my time => [0.8843597  0.04596698 0.069673
 best movie i have seen => [0.02159162 0.02921591 0.94919246]=> positive
 i will rate this movie as average => [0.19612424 0.20894141 0.5949344 ]=> positive
 you are a kind man => [0.01797673 0.02001002 0.9620132 ]=> positive
 worst kind of movie ever created in MCU => [0.8491575  0.07222106 0.07862142] => negativ
 I have seen this movie => [0.23983234 0.21232134 0.5478463 ]=> positive
```

```
new_data = ["The public went berserk for “Psycho” in 1960, but critics were not as crazy abou
            "this movie is shit, feels like i have wasted my time", "best movie i have seen",
            "you are a kind man", "worst kind of movie ever created in MCU","I have seen this
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred):
        print(f"{new_data[i]} => {pred} => positive")
    elif np.argmax(pred):
        print(f"{new_data[i]} => {pred} => neutral")
    else:
        print(f"{new_data[i]} => {pred}=> negative")
```

```
 The public went berserk for “Psycho” in 1960, but critics were not as crazy about Alfred
 this movie is shit, feels like i have wasted my time => [0.8843597  0.04596698 0.069673
 best movie i have seen => [0.02159158 0.02921588 0.9491926 ] => positive
 i will rate this movie as average => [0.1961243  0.20894152 0.5949342 ] => positive
 you are a kind man => [0.0179767  0.02000999 0.9620133 ] => positive
 worst kind of movie ever created in MCU => [0.84915733 0.07222116 0.07862155]=> negative
 I have seen this movie => [0.23983237 0.21232131 0.54784626] => positive
```

'negative' | 'neutral' | 'positive'

```
#new_data = ["this movie is shit, feels like i have wasted my time", "best movie i have seen"
new_data = ["The public went berserk for “Psycho” in 1960, but critics were not as crazy abou
            "this movie is shit, feels like i have wasted my time", "best movie i have seen",
            "you are a kind man", "worst kind of movie ever created in MCU","I have seen this
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred):
        print(f"{new_data[i]} => {pred} => positive")
    elif np.argmax(pred):
        print(f"{new_data[i]} => {pred} => neutral")
    else:
        print(f"{new_data[i]} => {pred}=> negative")
```

```
The public went berserk for “Psycho” in 1960, but critics were not as crazy about Alfred
this movie is shit, feels like i have wasted my time => [0.8843597  0.04596698 0.069673
best movie i have seen => [0.02159158 0.02921588 0.9491926 ] => positive
i will rate this movie as average => [0.1961243  0.20894152 0.5949342 ] => positive
you are a kind man => [0.0179767  0.02000999 0.9620133 ] => positive
worst kind of movie ever created in MCU => [0.84915733 0.07222116 0.07862155]=> negative
I have seen this movie => [0.23983237 0.21232131 0.54784626] => positive
```

```
new_data = ["The public went berserk for “Psycho” in 1960, but critics were not as crazy abou
            "this movie is shit, feels like i have wasted my time",
            "best movie i have seen",
            "i will rate this movie as average",
            "you are a kind man",
            "worst kind of movie ever created in MCU",
            "I have seen this movie"]

new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred):
        print(f"{pred} => positive")
    elif np.argmax(pred):
        print(f"{pred} => neutral")
    else:
        print(f"{pred}=> negative")
```

```
[0.79927886 0.07960716 0.12111395]=> negative
[0.8843597  0.04596698 0.06967334]=> negative
[0.02159158 0.02921588 0.9491926 ] => positive
[0.1961243  0.20894152 0.5949342 ] => positive
[0.0179767  0.02000999 0.9620133 ] => positive
[0.84915733 0.07222116 0.07862155]=> negative
[0.23983237 0.21232131 0.54784626] => positive
```

```
#new_data = ["this movie is shit, feels like i have wasted my time", "best movie i have seen"
new_data = ["The public went berserk for “Psycho” in 1960, but critics were not as crazy abou
```

```
            "this movie is shit, feels like i have wasted my time",
            "best movie i have seen",
            "i will rate this movie as average",
            "you are a kind man",
            "worst kind of movie ever created in MCU",
            "I have seen this movie"
            ]
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred):
        print(f"{new_data[i]} => \n {pred} => Positive")
    elif np.argmax(pred):
        print(f"{new_data[i]} => \n {pred} => Neutral")
    else:
        print(f"{new_data[i]} => \n {pred} => Negative")
```

```
 The public went berserk for “Psycho” in 1960, but critics were not as crazy about Alfred
  [0.79927886 0.07960716 0.12111395] => Negative
 this movie is shit, feels like i have wasted my time =>
  [0.8843597  0.04596698 0.06967334] => Negative
 best movie i have seen =>
  [0.02159158 0.02921588 0.9491926 ] => Positive
 i will rate this movie as average =>
  [0.1961243  0.20894152 0.5949342 ] => Positive
 you are a kind man =>
  [0.0179767  0.02000999 0.9620133 ] => Positive
 worst kind of movie ever created in MCU =>
  [0.84915733 0.07222116 0.07862155] => Negative
 I have seen this movie =>
  [0.23983237 0.21232131 0.54784626] => Positive
```

```
new_data = ["The public went berserk for “Psycho” in 1960, but critics were not as crazy abou
            "this movie is shit, feels like i have wasted my time",
            "best movie i have seen",
            "i will rate this movie as average",
            "you are a kind man",
            "worst kind of movie ever created in MCU",
            "I have seen this movie"
            ]
new_prediction = predictor_load.predict(new_data, return_proba=True)


for i, pred in enumerate(new_prediction):
    if np.argmax(pred):
        print(f"{pred} => {np.argmax(pred)} => Positive")
    elif np.argmax(pred):
        print(f"{pred} => {np.argmax(pred)} => Neutral")
    else:
        print(f"{pred} => {np.argmax(pred)} => Negative")
```

```
[0.79927886 0.07960716 0.12111395] => 0 => Negative
[0.8843597  0.04596698 0.06967334] => 0 => Negative
[0.02159158 0.02921588 0.9491926 ] => 2 => Positive
[0.1961243  0.20894152 0.5949342 ] => 2 => Positive
[0.0179767  0.02000999 0.9620133 ] => 2 => Positive
[0.84915733 0.07222116 0.07862155] => 0 => Negative
[0.23983237 0.21232131 0.54784626] => 2 => Positive
```

```
np.argmax(pred)
```

```
2
```

```
pred = new_prediction
if np.argmax(pred[index_max]) < 0.5:
  print("Neutral")
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-184-f1647d0e5b1b> in <module>()
      1 pred = new_prediction
----> 2 if np.argmax(pred[index_max]) < 0.5:
      3   print("Neutral")

IndexError: arrays used as indices must be of integer (or boolean) type
```

SEARCH STACK OVERFLOW

```
index_max = np.argmax(pred)
if new_prediction[index_max] < 0.5:
  print("Neutral")
```

```
Pred = new_data[5]
new_prediction = predictor_load.predict(new_data, return_proba=True)
for i, pred in enumerate(new_prediction):
  print(np.argmax(pred))
```

```
0
0
2
2
2
0
```

```
0.23983234+0.21232134+0.5478463
```

```
0.99999998
```

```
#return_proba = True means it will give the prediction probabilty for each class
```

```
predictor_load.predict(new_data, return_proba=True)
```

```
array([[0.8843597 , 0.04596698, 0.06967334],
       [0.02159158, 0.02921588, 0.9491926 ],
       [0.1961243 , 0.20894152, 0.5949342 ],
       [0.0179767 , 0.02000999, 0.9620133 ],
       [0.84915733, 0.07222116, 0.07862155]], dtype=float32)
```

```
0.8843597+0.04596698+0.06967336
```

```
1.00000004
```

```
0.02159163+0.02921593+0.94919246
```

```
1.00000002
```

```
0.1961243+0.2089415+0.5949342
```

```
1.0
```

```
0.01797669+0.02000999+0.9620133
```

```
0.9999999799999999
```

```
new_data = ["this movie is shit, feels like i have wasted my time", "best movie i have seen",
new_prediction = predictor_load.predict(new_data, return_proba=True)

for i, pred in enumerate(new_prediction):
    if np.argmax(pred) -1 <= -0.5:
        print(f"{new_data[i]} => {pred} => negative")
    elif np.argmax(pred) -0.5 <= 0.5:
        print(f"{new_data[i]} => {pred} => neutral")
    else:
        print(f"{new_data[i]} => {pred}=> positive")
```

```
this movie is shit, feels like i have wasted my time => [0.8843597  0.04596698 0.069673
best movie i have seen => [0.02159158 0.02921588 0.9491926 ]=> positive
i will rate this movie as average => [0.1961243  0.20894152 0.5949342 ]=> positive
you are a kind man => [0.0179767  0.02000999 0.9620133 ]=> positive
worst kind of movie ever created in MCU => [0.84915733 0.07222116 0.07862155] => negati
```