Experiment No 7

Aim: To study perform various Natural Language Processing task like Text summarization, Sentiment Analysis using available libraries like Hugging Face's Transformers, TensorFlow or PyTorch

Introduction:

Text Summarization using BERT

Extractive Summarization: BERT can be utilized for extractive summarization, where key sentences or phrases are selected from the original text to form a summary.

Sentence Embedding's: Use BERT to generate embedding's for each sentence in the text.

Similarity Measures: Calculate similarity scores between sentences (e.g., using cosine similarity).

Select Top Sentences: Choose sentences with the highest similarity scores to form the summary.

Abstractive Summarization: This involves generating a summary that might not directly include sentences from the original text. BERT can aid in this by fine-tuning a model specifically for abstractive summarization.

Fine-tuning: Fine-tune the pre-trained BERT model on a summarization dataset (e.g., CNN/Daily Mail dataset).

Sequence-to-sequence Model: Employ techniques like seq2seq models or transformers to generate summaries.

Sentiment Analysis using BERT

Fine-tuning BERT: The pre-trained BERT model can be fine-tuned on a sentiment analysis dataset. The model's classification layers can be adjusted for sentiment analysis.

Dataset Preparation: Obtain a labeled dataset for sentiment analysis (e.g., IMDB movie reviews, Twitter sentiment dataset).

Tokenization and Fine-tuning: Tokenize the text, prepare input sequences, and fine-tune BERT on the sentiment classification task.

Prediction: After fine-tuning, use the trained model to predict sentiment labels (positive, negative, neutral) for new text inputs.

Tools and Libraries:

Hugging Face's Transformers: This library provides easy access to pre-trained models like BERT and various other NLP-related functionalities for tasks like tokenization, model loading, and fine-tuning.

TensorFlow or PyTorch: Use these deep learning frameworks to implement BERT-based models and fine-tuning for specific tasks.

Lab Experiment to be performed in this session:

1. Perform Text Summarization using BERT (BERT summarizer library can be directly installed in python using the following commands python pip install bert-extractive-summarizer for the easies of the implementation.)
2. Perform Sentiment Analysis using BERT.

Perform Text Summarization using BERT

Importing Libraries

```
!pip install bert-extractive-summarizer
```

```
Requirement already satisfied: bert-extractive-summarizer in /usr/local/lib/python3.10/dist-packages (0.10.1)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (from bert-extractive-summarizer) (4.35.2)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from bert-extractive-summarizer) (1.2.2)
Requirement already satisfied: spacy in /usr/local/lib/python3.10/dist-packages (from bert-extractive-summarizer) (3.6.1)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->bert-extractive-summariz
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->bert-extractive-summarize
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->bert-extractive-summariz
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->bert-extractive-s
Requirement already satisfied: spacy-legacy<3.1.0,>=3.0.11 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-s
Requirement already satisfied: spacy-loggers<2.0.0,>=1.0.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-s
Requirement already satisfied: murmurhash<1.1.0,>=0.28.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-sum
Requirement already satisfied: cymem<2.1.0,>=2.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarize
Requirement already satisfied: preshed<3.1.0,>=3.0.2 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summari
Requirement already satisfied: thinc<8.2.0,>=8.1.8 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarize
Requirement already satisfied: wasabi<1.2.0,>=0.9.1 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summariz
Requirement already satisfied: srsly<3.0.0,>=2.4.3 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarize
Requirement already satisfied: catalogue<2.1.0,>=2.0.6 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summa
Requirement already satisfied: typer<0.10.0,>=0.3.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summariz
Requirement already satisfied: pathy>=0.10.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarizer) (0
Requirement already satisfied: smart-open<7.0.0,>=5.2.1 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summ
Requirement already satisfied: tqdm<5.0.0,>=4.38.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarize
Requirement already satisfied: requests<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summa
Requirement already satisfied: pydantic!=1.8,!=1.8.1,<3.0.0,>=1.7.4 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-ext
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarizer) (3.1.2)
Requirement already satisfied: setuptools in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarizer) (67.7
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summarizer)
Requirement already satisfied: langcodes<4.0.0,>=3.2.0 in /usr/local/lib/python3.10/dist-packages (from spacy->bert-extractive-summa
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extractive-summarizer)
Requirement already satisfied: huggingface-hub<1.0,>=0.16.4 in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extr
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extractive-summarizer
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extractive-summ
Requirement already satisfied: tokenizers<0.19,>=0.14 in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extractive
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from transformers->bert-extractive-sum
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.16.4->trans
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.1
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->sp
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy->bert-ex
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy->b
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests<3.0.0,>=2.13.0->spacy->b
Requirement already satisfied: blis<0.8.0,>=0.7.8 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spacy->bert
Requirement already satisfied: confection<1.0.0,>=0.0.1 in /usr/local/lib/python3.10/dist-packages (from thinc<8.2.0,>=8.1.8->spacy
Requirement already satisfied: click<9.0.0,>=7.1.1 in /usr/local/lib/python3.10/dist-packages (from typer<0.10.0,>=0.3.0->spacy->ber
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->spacy->bert-extractive-summa
```

```
from summarizer import Summarizer
import tensorflow as tf
```

```
model = Summarizer()
```

```
from summarizer import Summarizer

body = '''
The Chrysler Building, the famous art deco New York skyscraper, will be sold for a small fraction of its previous sales price.
The deal, first reported by The Real Deal, was for $150 million, according to a source familiar with the deal.
Mubadala, an Abu Dhabi investment fund, purchased 90% of the building for $800 million in 2008.
Real estate firm Tishman Speyer had owned the other 10%.
The buyer is RFR Holding, a New York real estate company.
Officials with Tishman and RFR did not immediately respond to a request for comments.
It's unclear when the deal will close.
The building sold fairly quickly after being publicly placed on the market only two months ago.
The sale was handled by CBRE Group.
The incentive to sell the building at such a huge loss was due to the soaring rent the owners pay to Cooper Union, a New York college, f
The rent is rising from $7.75 million last year to $32.5 million this year to $41 million in 2028.
Meantime, rents in the building itself are not rising nearly that fast.
While the building is an iconic landmark in the New York skyline, it is competing against newer office towers with large floor-to-ceilir
Still the building is among the best known in the city, even to people who have never been to New York.
It is famous for its triangle-shaped, vaulted windows worked into the stylized crown, along with its distinctive eagle gargoyles near th
It has been featured prominently in many films, including Men in Black 3, Spider-Man, Armageddon, Two Weeks Notice and Independence Day.
The previous sale took place just before the 2008 financial meltdown led to a plunge in real estate prices.
Still there have been a number of high profile skyscrapers purchased for top dollar in recent years, including the Waldorf Astoria hotel
Blackstone Group (BX) bought it for $1.3 billion 2015.
The Chrysler Building was the headquarters of the American automaker until 1953, but it was named for and owned by Chrysler chief Walter
Walter Chrysler had set out to build the tallest building in the world, a competition at that time with another Manhattan skyscraper unc
Once the competitor could rise no higher, the spire of the Chrysler building was raised into view, giving it the title.
'''

model = Summarizer()
result = model(body, min_length=60)
full = ''.join(result)
full
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
'The Chrysler Building, the famous art deco New York skyscraper, will be sold for a
small fraction of its previous sales price. The deal, first reported by The Real Dea
l, was for $150 million, according to a source familiar with the deal. The building
sold fairly quickly after being publicly placed on the market only two months ago. T
```

Specifying number of sentences

```
from summarizer import Summarizer
body = 'Text body that you want to summarize with BERT'
model = Summarizer()
result = model(body, ratio=0.2)  # Specified with ratio
result = model(body, num_sentences=3)  # Will return 3 sentences
```

Use SBert

```
!pip install -U sentence-transformers
```

```
Requirement already satisfied: sentence-transformers in /usr/local/lib/python3.10/dist-packages (2.2.2)
Requirement already satisfied: transformers<5.0.0,>=4.6.0 in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (4
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (4.66.1)
Requirement already satisfied: torch>=1.6.0 in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (2.1.0+cu118)
Requirement already satisfied: torchvision in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (0.16.0+cu118)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.23.5)
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.2.2)
Requirement already satisfied: scipy in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (1.11.4)
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (3.8.1)
Requirement already satisfied: sentencepiece in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (0.1.99)
Requirement already satisfied: huggingface-hub>=0.4.0 in /usr/local/lib/python3.10/dist-packages (from sentence-transformers) (0.19
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->sentence-transforme
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->sentence-tr
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->sentence-transforme
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->sentence-transfo
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->s
Requirement already satisfied: packaging>=20.9 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub>=0.4.0->sentence-tra
Requirement already satisfied: sympy in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->sentence-transformers) (1.12)
Requirement already satisfied: networkx in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->sentence-transformers) (3.2.1
Requirement already satisfied: jinja2 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->sentence-transformers) (3.1.2)
Requirement already satisfied: triton==2.1.0 in /usr/local/lib/python3.10/dist-packages (from torch>=1.6.0->sentence-transformers)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.0,>=4.6.0->senter
Requirement already satisfied: tokenizers<0.19,>=0.14 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.0,>=4.6.0->s
Requirement already satisfied: safetensors>=0.3.1 in /usr/local/lib/python3.10/dist-packages (from transformers<5.0.0,>=4.6.0->sente
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk->sentence-transformers) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk->sentence-transformers) (1.3.2)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn->sentence-transfor
Requirement already satisfied: pillow!=8.3.*,>=5.3.0 in /usr/local/lib/python3.10/dist-packages (from torchvision->sentence-transfor
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.10/dist-packages (from jinja2->torch>=1.6.0->sentence-trans
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub>=
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub>=0.4.0->sente
```

```
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub>=0.4.0
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->huggingface-hub>=0.4.0
Requirement already satisfied: mpmath>=0.19 in /usr/local/lib/python3.10/dist-packages (from sympy->torch>=1.6.0->sentence-transfor
```

```
from summarizer.sbert import SBertSummarizer

body = '''The Chrysler Building, the famous art deco New York skyscraper, will be sold for a small fraction of its previous sales price.
The deal, first reported by The Real Deal, was for $150 million, according to a source familiar with the deal.
Mubadala, an Abu Dhabi investment fund, purchased 90% of the building for $800 million in 2008.
Real estate firm Tishman Speyer had owned the other 10%.
The buyer is RFR Holding, a New York real estate company.
Officials with Tishman and RFR did not immediately respond to a request for comments.
It's unclear when the deal will close.
The building sold fairly quickly after being publicly placed on the market only two months ago.
The sale was handled by CBRE Group.
The incentive to sell the building at such a huge loss was due to the soaring rent the owners pay to Cooper Union, a New York college, f
The rent is rising from $7.75 million last year to $32.5 million this year to $41 million in 2028.
Meantime, rents in the building itself are not rising nearly that fast.
While the building is an iconic landmark in the New York skyline, it is competing against newer office towers with large floor-to-ceilir
Still the building is among the best known in the city, even to people who have never been to New York.
It is famous for its triangle-shaped, vaulted windows worked into the stylized crown, along with its distinctive eagle gargoyles near th
It has been featured prominently in many films, including Men in Black 3, Spider-Man, Armageddon, Two Weeks Notice and Independence Day.
The previous sale took place just before the 2008 financial meltdown led to a plunge in real estate prices.
Still there have been a number of high profile skyscrapers purchased for top dollar in recent years, including the Waldorf Astoria hotel
Blackstone Group (BX) bought it for $1.3 billion 2015.
The Chrysler Building was the headquarters of the American automaker until 1953, but it was named for and owned by Chrysler chief Walter
Walter Chrysler had set out to build the tallest building in the world, a competition at that time with another Manhattan skyscraper unc
Once the competitor could rise no higher, the spire of the Chrysler building was raised into view, giving it the title.'''
model = SBertSummarizer('paraphrase-MiniLM-L6-v2')
result = model(body, num_sentences=3)
result
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning
  warnings.warn(
'The Chrysler Building, the famous art deco New York skyscraper, will be sold for a
small fraction of its previous sales price. The incentive to sell the building at su
ch a huge loss was due to the soaring rent the owners pay to Cooper Union, a New Yor
k college, for the land under the building. Walter Chrysler had set out to build the
```

Sentiment Analysis using BERT

```
import os
import shutil
import tarfile
import tensorflow as tf
from transformers import BertTokenizer, TFBertForSequenceClassification
import pandas as pd
from bs4 import BeautifulSoup
import re
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.offline as pyo
import plotly.graph_objects as go
from wordcloud import WordCloud, STOPWORDS
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
```

```
# Get the current working directory
current_folder = os.getcwd()

dataset = tf.keras.utils.get_file(
    fname ="aclImdb.tar.gz",
    origin ="http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz",
    cache_dir= current_folder,
    extract = True)
```

```
Downloading data from http://ai.stanford.edu/~amaas/data/sentiment/aclImdb_v1.tar.gz
84125825/84125825 [==============================] - 25s 0us/step
```

```
dataset_path = os.path.dirname(dataset)
# Check the dataset
os.listdir(dataset_path)
```

```
['aclImdb.tar.gz', 'aclImdb']
```

```
# Dataset directory
dataset_dir = os.path.join(dataset_path, 'aclImdb')

# Check the Dataset directory
os.listdir(dataset_dir)
```

```
['imdbEr.txt', 'test', 'imdb.vocab', 'train', 'README']
```

```
train_dir = os.path.join(dataset_dir,'train')
os.listdir(train_dir)
```

```
['labeledBow.feat',
 'urls_neg.txt',
 'urls_unsup.txt',
 'unsup',
 'pos',
 'unsupBow.feat',
 'neg',
 'urls_pos.txt']
```

```
for file in os.listdir(train_dir):
    file_path = os.path.join(train_dir, file)
    # Check if it's a file (not a directory)
    if os.path.isfile(file_path):
        with open(file_path, 'r', encoding='utf-8') as f:
            first_value = f.readline().strip()
            print(f"{file}: {first_value}")
    else:
        print(f"{file}: {file_path}")
```

```
labeledBow.feat: 9 0:9 1:1 2:4 3:4 4:6 5:4 6:2 7:2 8:4 10:4 12:2 26:1 27:1 28:1 29:2 32:1 41:1 45:1 47:1 50:1 54:2 57:1 59:1 63:2 64
urls_neg.txt: http://www.imdb.com/title/tt0064354/usercomments
urls_unsup.txt: http://www.imdb.com/title/tt0018515/usercomments
unsup: /content/datasets/aclImdb/train/unsup
pos: /content/datasets/aclImdb/train/pos
unsupBow.feat: 0 0:8 1:6 3:5 4:2 5:1 7:1 8:5 9:2 10:1 11:2 13:3 16:1 17:1 18:1 19:1 22:3 24:1 26:3 28:1 30:1 31:1 35:2 36:1 39:2 40
neg: /content/datasets/aclImdb/train/neg
urls_pos.txt: http://www.imdb.com/title/tt0453418/usercomments
```

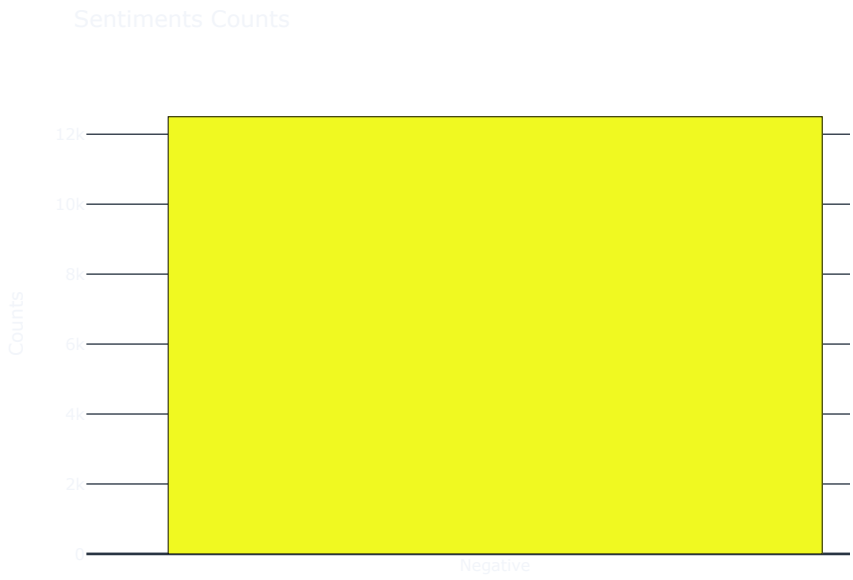
```
def load_dataset(directory):
    data = {"sentence": [], "sentiment": []}
    for file_name in os.listdir(directory):
        print(file_name)
        if file_name == 'pos':
            positive_dir = os.path.join(directory, file_name)
            for text_file in os.listdir(positive_dir):
                text = os.path.join(positive_dir, text_file)
                with open(text, "r", encoding="utf-8") as f:
                    data["sentence"].append(f.read())
                    data["sentiment"].append(1)
        elif file_name == 'neg':
            negative_dir = os.path.join(directory, file_name)
            for text_file in os.listdir(negative_dir):
                text = os.path.join(negative_dir, text_file)
                with open(text, "r", encoding="utf-8") as f:
                    data["sentence"].append(f.read())
                    data["sentiment"].append(0)

    return pd.DataFrame.from_dict(data)
```

```
# Load the dataset from the train_dir
train_df = load_dataset(train_dir)
print(train_df.head())

test_dir = os.path.join(dataset_dir,'test')

# Load the dataset from the train_dir
test_df = load_dataset(test_dir)
print(test_df.head())
```

```
labeledBow.feat
urls_neg.txt
urls_unsup.txt
unsup
pos
unsupBow.feat
neg
urls_pos.txt
                                            sentence  sentiment
0  Sequel to "The Kingdom" is bloodier and even m...          1
1  This isn't Masterpiece Theater. You shouldn't ...          1
```

```
2  The movie starts off in a classroom setting wh...          1
3  This is a romantic comedy with the emphasis on...          1
4  This is a musical adaptation of Dicken's "Oliv...          1
labeledBow.feat
urls_neg.txt
pos
neg
urls_pos.txt
                                          sentence  sentiment
0  My daughter already wrote a review of this mov...          1
1  I heard an interview with the main actor who s...          1
2  Duchess and her three kittens are enjoying the...          1
3  Gary Busey did a splendid job playing the rock...          1
4  "Just before dawn " is one of the best slasher...          1
```

```
sentiment_counts = train_df['sentiment'].value_counts()

fig =px.bar(x= {0:'Negative',1:'Positive'},
            y= sentiment_counts.values,
            color=sentiment_counts.index,
            color_discrete_sequence = px.colors.qualitative.Dark24,
            title='<b>Sentiments Counts')

fig.update_layout(title='Sentiments Counts',
                  xaxis_title='Sentiment',
                  yaxis_title='Counts',
                  template='plotly_dark')

# Show the bar chart
fig.show()
pyo.plot(fig, filename = 'Sentiments Counts.html', auto_open = True)
```




```
'Sentiments Counts.html'
```

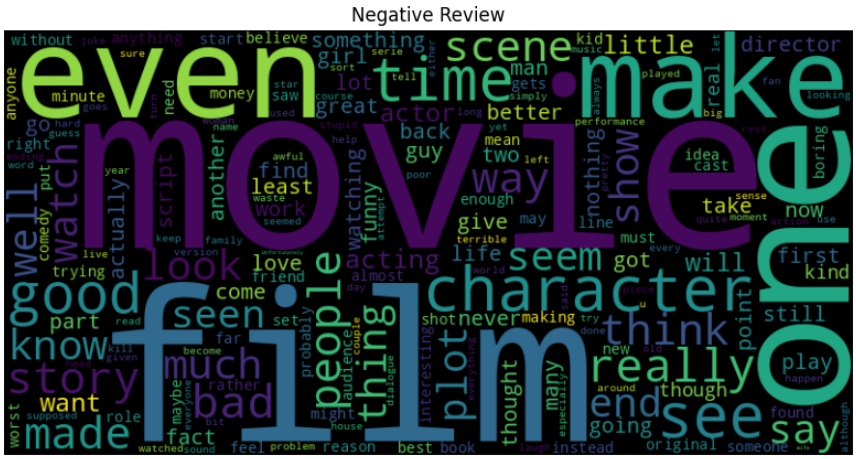
```
def text_cleaning(text):
    soup = BeautifulSoup(text, "html.parser")
    text = re.sub(r'\[[^]]*\]', '', soup.get_text())
    pattern = r"[^a-zA-Z0-9\s,']"
    text = re.sub(pattern, '', text)
    return text


# Train dataset
train_df['Cleaned_sentence'] = train_df['sentence'].apply(text_cleaning).tolist()
# Test dataset
test_df['Cleaned_sentence'] = test_df['sentence'].apply(text_cleaning)
```

```
<ipython-input-31-11f1ec3c8727>:2: MarkupResemblesLocatorWarning:

The input looks more like a filename than markup. You may want to open this file and pass the filehandle into Beautiful Soup.

<ipython-input-31-11f1ec3c8727>:2: MarkupResemblesLocatorWarning:
```

```
The input looks more like a filename than markup. You may want to open this file and pass the filehandle into Beautiful Soup.
```

```
# Function to generate word cloud
def generate_wordcloud(text,Title):
    all_text = " ".join(text)
    wordcloud = WordCloud(width=800,
                          height=400,
                          stopwords=set(STOPWORDS),
                          background_color='black').generate(all_text)
    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title(Title)
    plt.show()
```

```
negative = train_df[train_df['sentiment']==0]['Cleaned_sentence'].tolist()
generate_wordcloud(negative,'Negative Review')
```




```
# Training data
#Reviews = "[CLS] " +train_df['Cleaned_sentence'] + "[SEP]"
Reviews = train_df['Cleaned_sentence']
Target = train_df['sentiment']

# Test data
#test_reviews = "[CLS] " +test_df['Cleaned_sentence'] + "[SEP]"
test_reviews = test_df['Cleaned_sentence']
test_targets = test_df['sentiment']
```

```
x_val, x_test, y_val, y_test = train_test_split(test_reviews,
                                                test_targets,
                                                test_size=0.5,
                                                stratify = test_targets)
```

```
#Tokenize and encode the data using the BERT tokenizer
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased', do_lower_case=True)


max_len= 128
# Tokenize and encode the sentences
X_train_encoded = tokenizer.batch_encode_plus(Reviews.tolist(),
                                              padding=True,
                                              truncation=True,
                                              max_length = max_len,
                                              return_tensors='tf')

X_val_encoded = tokenizer.batch_encode_plus(x_val.tolist(),
                                              padding=True,
                                              truncation=True,
                                              max_length = max_len,
                                              return_tensors='tf')

X_test_encoded = tokenizer.batch_encode_plus(x_test.tolist(),
                                              padding=True,
                                              truncation=True,
                                              max_length = max_len,
                                              return_tensors='tf')
```

```
k = 0
print('Training Comments -->>',Reviews[k])
print('\nInput Ids -->>\n',X_train_encoded['input_ids'][k])
print('\nDecoded Ids -->>\n',tokenizer.decode(X_train_encoded['input_ids'][k]))
print('\nAttention Mask -->>\n',X_train_encoded['attention_mask'][k])
print('\nLabels -->>',Target[k])
```

```
Training Comments -->> Sequel to The Kingdom is bloodier and even more twisted I only saw half I was exhausted and couldn't sit thro

Input Ids -->>
 tf.Tensor(
[  101  8297  2000  1996  2983  2003  2668  3771  1998  2130  2062  6389
  1045  2069  2387  2431  1045  2001  9069  1998  2481  1005  1056  4133
  2083  2035  1019  2260  2847  2021  1045  3866  2054  1045  2387 11277
  1010  2668  1010  4028  1010 16149  1010 14163 16238 10834  1010 21768
 15222  2015  2038  2009  2035  2065  2017  2031  1037  2844  4308  1998
  2066  6881  5691  2023  2003  2005  2017  9777  2080  1010  2017  2123
  1005  1056  2031  2000  2156  2112  1015  2000  3305  2023 29337  1005
  2222  3275  2009  2041  3527  2229  3087  2113  2065  2983  1015  1998
  1016  2024  2800  2006  4966  3564  2083  2122  8589  5691  1999  1037
  3004  2003 14841  4892 13718  1010  2045  2763  2180  1005  1056  2022
  1037  2983  1017 11795  3367  6979  3995   102], shape=(128,), dtype=int32)

Decoded Ids -->>
 [CLS] sequel to the kingdom is bloodier and even more twisted i only saw half i was exhausted and couldn't sit through all 5 12 hou

Attention Mask -->>
 tf.Tensor(
[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1], shape=(128,), dtype=int32)

Labels -->> 1
```

```
# Intialize the model
model = TFBertForSequenceClassification.from_pretrained('bert-base-uncased', num_labels=2)

# Compile the model with an appropriate optimizer, loss function, and metrics
optimizer = tf.keras.optimizers.Adam(learning_rate=2e-5)
loss = tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
metric = tf.keras.metrics.SparseCategoricalAccuracy('accuracy')
model.compile(optimizer=optimizer, loss=loss, metrics=[metric])
```

model.safetensors: 100% 440M/440M [00:05<00:00, 117MB/s]

```
All PyTorch model weights were used when initializing TFBertForSequenceClassification

Some weights or buffers of the TF 2.0 model TFBertForSequenceClassification were not
```

```
# Step 5: Train the model
history = model.fit(
  [X_train_encoded['input_ids'], X_train_encoded['token_type_ids'], X_train_encoded['attention_mask']],
  Target,
  validation_data=(
  [X_val_encoded['input_ids'], X_val_encoded['token_type_ids'], X_val_encoded['attention_mask']],y_val),
  batch_size=32
```

```
  batch_size=32,
  epochs=3
)
```

```
Epoch 1/3
782/782 [==============================] - 834s 1s/step - loss: 0.3415 - accuracy: 0.8480 - val_loss: 0.2680 - val_accuracy: 0.8874
Epoch 2/3
782/782 [==============================] - 785s 1s/step - loss: 0.2047 - accuracy: 0.9184 - val_loss: 0.2674 - val_accuracy: 0.8910
Epoch 3/3
782/782 [==============================] - 785s 1s/step - loss: 0.1092 - accuracy: 0.9607 - val_loss: 0.3770 - val_accuracy: 0.8840
```

```
#Evaluate the model on the test data
test_loss, test_accuracy = model.evaluate(
    [X_test_encoded['input_ids'], X_test_encoded['token_type_ids'], X_test_encoded['attention_mask']],
    y_test
)
print(f'Test loss: {test_loss}, Test accuracy: {test_accuracy}')
```

```
391/391 [==============================] - 110s 280ms/step - loss: 0.3654 - accuracy: 0.8835
Test loss: 0.3653511703014374, Test accuracy: 0.8835200071334839
```

```
pred = model.predict(
  [X_test_encoded['input_ids'], X_test_encoded['token_type_ids'], X_test_encoded['attention_mask']])

# pred is of type TFSequenceClassifierOutput
logits = pred.logits

# Use argmax along the appropriate axis to get the predicted labels
pred_labels = tf.argmax(logits, axis=1)

# Convert the predicted labels to a NumPy array
pred_labels = pred_labels.numpy()

label = {
  1: 'positive',
  0: 'Negative'
}

# Map the predicted labels to their corresponding strings using the label dictionary
pred_labels = [label[i] for i in pred_labels]
Actual = [label[i] for i in y_test]

print('Predicted Label :', pred_labels[:10])
print('Actual Label :', Actual[:10])
```