Installing requirements & Imports, Connecting with Drive

```
import gc
gc.collect()
```

```
111
```

```
!pip install keras-tuner
!pip install scikit-learn
!pip install transformers
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting keras-tuner
  Downloading keras_tuner-1.3.5-py3-none-any.whl (176 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 176.1/176.1 kB 1.7 MB/s eta 0:00:00
Collecting kt-legacy
  Downloading kt_legacy-1.0.5-py3-none-any.whl (9.6 kB)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from keras-tuner) (23.1)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from keras-tuner) (2.27.1)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->keras-tuner) (3.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests->keras-tu
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->keras-tuner)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->keras-tuner) (2
Installing collected packages: kt-legacy, keras-tuner
Successfully installed keras-tuner-1.3.5 kt-legacy-1.0.5
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: scikit-learn in /usr/local/lib/python3.10/dist-packages (1.2.2)
Requirement already satisfied: scipy>=1.3.2 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.10.1)
Requirement already satisfied: joblib>=1.1.1 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.2.0)
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (3.1.0)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.10/dist-packages (from scikit-learn) (1.22.4)
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting transformers
  Downloading transformers-4.28.1-py3-none-any.whl (7.0 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.0/7.0 MB 32.7 MB/s eta 0:00:00
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0)
Collecting tokenizers!=0.11.3,<0.14,>=0.11.1
  Downloading tokenizers-0.13.3-cp310-cp310-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (7.8 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 106.8 MB/s eta 0:00:00
Collecting huggingface-hub<1.0,>=0.11.0
  Downloading huggingface_hub-0.14.1-py3-none-any.whl (224 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 224.5/224.5 kB 30.3 MB/s eta 0:00:00
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.12.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.22.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (23.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.65.0)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.27.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2022.10.31)
Requirement already satisfied: fsspec in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.11.0->transfo
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.10/dist-packages (from requests->transfor
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4)
Installing collected packages: tokenizers, huggingface-hub, transformers
Successfully installed huggingface-hub-0.14.1 tokenizers-0.13.3 transformers-4.28.1
```

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
import numpy as np
import pandas as pd
import os
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import tensorflow as tf
from tensorflow.keras.layers import Embedding, LSTM, Dense, Dropout, Bidirectional, GRU, Conv1D, GlobalMaxPooling1D
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
```

```
from tensorflow.keras.preprocessing.text import Tokenizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from keras_tuner import RandomSearch

nltk.download('stopwords')
```

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

```
!unzip /content/gdrive/MyDrive/True.csv.zip
!unzip /content/gdrive/MyDrive/Fake.csv.zip
```

```
Archive:  /content/gdrive/MyDrive/True.csv.zip
  inflating: True.csv
Archive:  /content/gdrive/MyDrive/Fake.csv.zip
  inflating: Fake.csv
```

```
!unzip /content/gdrive/MyDrive/AllTheNews.zip
```

```
Archive:  /content/gdrive/MyDrive/AllTheNews.zip
  inflating: articles1.csv
  inflating: articles2.csv
  inflating: articles3.csv
```

```
!unzip /content/gdrive/MyDrive/archive.zip
```

```
Archive:  /content/gdrive/MyDrive/archive.zip
  inflating: fake.csv
```

```
!unzip /content/gdrive/MyDrive/liar_dataset.zip
```

```
Archive:  /content/gdrive/MyDrive/liar_dataset.zip
  inflating: README
  inflating: test.tsv
  inflating: train.tsv
  inflating: valid.tsv
```

```
!unzip /content/gdrive/MyDrive/FakeNewsNet-master.zip
```

```
Archive:  /content/gdrive/MyDrive/FakeNewsNet-master.zip
654361e1c8d5baa751baf1dac5032df621652280
   creating: FakeNewsNet-master/
  inflating: FakeNewsNet-master/README.md
   creating: FakeNewsNet-master/code/
  inflating: FakeNewsNet-master/code/config.json
  inflating: FakeNewsNet-master/code/main.py
  inflating: FakeNewsNet-master/code/news_content_collection.py
   creating: FakeNewsNet-master/code/resource_server/
  inflating: FakeNewsNet-master/code/resource_server/ResourceAllocator.py
 extracting: FakeNewsNet-master/code/resource_server/__init__.py
  inflating: FakeNewsNet-master/code/resource_server/app.py
   creating: FakeNewsNet-master/code/resources/
  inflating: FakeNewsNet-master/code/resources/tweet_keys_file.json
  inflating: FakeNewsNet-master/code/retweet_collection.py
  inflating: FakeNewsNet-master/code/tweet_collection.py
  inflating: FakeNewsNet-master/code/user_profile_collection.py
   creating: FakeNewsNet-master/code/util/
  inflating: FakeNewsNet-master/code/util/Constants.py
  inflating: FakeNewsNet-master/code/util/TwythonConnector.py
  inflating: FakeNewsNet-master/code/util/util.py
   creating: FakeNewsNet-master/dataset/
  inflating: FakeNewsNet-master/dataset/gossipcop_fake.csv
  inflating: FakeNewsNet-master/dataset/gossipcop_real.csv
  inflating: FakeNewsNet-master/dataset/politifact_fake.csv
  inflating: FakeNewsNet-master/dataset/politifact_real.csv
  inflating: FakeNewsNet-master/requirements.txt
```

Load and preprocess datasets

```
# https://www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset
# Load the initial datasets
true_data = pd.read_csv('/content/True.csv')
```

```
fake_data = pd.read_csv('/content/Fake.csv')

# Assign labels and combine datasets
true_data['label'] = 1
fake_data['label'] = 0
data = pd.concat([true_data, fake_data], ignore_index=True)
from dateutil.parser import parse
from dateutil.parser import ParserError

def parse_date(date_string):
    try:
        return parse(date_string)
    except (ParserError, TypeError):
        return None

data['date'] = data['date'].apply(parse_date)

data
```

	title	text	subject	date	label
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	2017-12-31	1
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	2017-12-29	1
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	2017-12-31	1
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	2017-12-30	1
4	Trump wants Postal Service to charge 'much mor...	SEATTLE/WASHINGTON (Reuters) - President Donal...	politicsNews	2017-12-29	1
...	...	...	...	...	...
	McPain: John				

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 44898 entries, 0 to 44897
Data columns (total 5 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   title    44898 non-null  object
 1   text     44898 non-null  object
 2   subject  44898 non-null  object
 3   date     44888 non-null  datetime64[ns]
 4   label    44898 non-null  int64
dtypes: datetime64[ns](1), int64(1), object(3)
memory usage: 1.7+ MB
```

```
# https://www.kaggle.com/datasets/snapcrack/all-the-news
# Load additional datasets
all_news_1 = pd.read_csv('/content/articles1.csv')
all_news_2 = pd.read_csv('/content/articles2.csv')
all_news_3 = pd.read_csv('/content/articles3.csv')

# Combine the additional datasets
all_news = pd.concat([all_news_1, all_news_2, all_news_3], ignore_index=True)
all_news
```

	Unnamed: 0	id	title	publication	author	date	year	mo
0	0	17283	House Republicans Fret About Winning Their Hea...	New York Times	Carl Hulse	2016-12-31	2016.0	
1	1	17284	Rift Between Officers and Residents as Killi...	New York Times	Benjamin Mueller and Al Baker	2017-06-19	2017.0	

```
print(all_news.loc[3, 'title'])
```

```
Among Deaths in 2016, a Heavy Toll in Pop Music - The New York Times
```

```
all_news.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 142570 entries, 0 to 142569
Data columns (total 10 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   Unnamed: 0   142570 non-null  int64  
 1   id           142570 non-null  int64  
 2   title        142568 non-null  object 
 3   publication  142570 non-null  object 
 4   author       126694 non-null  object 
 5   date         139929 non-null  object 
 6   year         139929 non-null  float64
 7   month        139929 non-null  float64
 8   url          85559 non-null   object 
 9   content      142570 non-null  object 
dtypes: float64(2), int64(2), object(6)
memory usage: 10.9+ MB
```

```
# Drop unwanted columns
all_news = all_news.drop(columns=['Unnamed: 0', 'id', 'year', 'month'])

# Change 'date' column to datetime dtype
all_news['date'] = pd.to_datetime(all_news['date'])

all_news['label'] = 1  # Assuming all news articles in this dataset are legitimate

# Rename 'content' column in all_news to 'text'
all_news.rename(columns={'content': 'text'}, inplace=True)

# Combine with the existing dataset
data = pd.concat([data, all_news], ignore_index=True)
data
```

	title	text	subject	date	label	publicatio
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	2017-12-31	1	Na
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	2017-12-29	1	Na
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	2017-12-31	1	Na
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	2017-12-30	1	Na
	Trump wants					

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 187468 entries, 0 to 187467
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   title        187466 non-null  object
 1   text         187468 non-null  object
 2   subject      44898 non-null   object
 3   date         184817 non-null  datetime64[ns]
 4   label        187468 non-null  int64
 5   publication  142570 non-null  object
 6   author       126694 non-null  object
 7   url          85559 non-null   object
dtypes: datetime64[ns](1), int64(1), object(6)
memory usage: 11.4+ MB
```

```
# https://www.kaggle.com/datasets/mrisdal/fake-news
# Load additional fake news dataset
fake_news_additional = pd.read_csv('/content/fake.csv')
fake_news_additional
```

uuid ord in thread author

```
fake_news_additional.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12999 entries, 0 to 12998
Data columns (total 20 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   uuid                12999 non-null  object 
 1   ord_in_thread       12999 non-null  int64  
 2   author              10575 non-null  object 
 3   published           12999 non-null  object 
 4   title               12319 non-null  object 
 5   text                12953 non-null  object 
 6   language            12999 non-null  object 
 7   crawled             12999 non-null  object 
 8   site_url            12999 non-null  object 
 9   country             12823 non-null  object 
 10  domain_rank         8776 non-null   float64
 11  thread_title        12987 non-null  object 
 12  spam_score          12999 non-null  float64
 13  main_img_url        9356 non-null   object 
 14  replies_count       12999 non-null  int64  
 15  participants_count  12999 non-null  int64  
 16  likes               12999 non-null  int64  
 17  comments            12999 non-null  int64  
 18  shares              12999 non-null  int64  
 19  type                12999 non-null  object 
dtypes: float64(2), int64(6), object(12)
memory usage: 2.0+ MB
```

```
# Keep the 'author', 'site_url', 'published', 'title', and 'text' columns
columns_to_keep = ['author', 'site_url', 'published', 'title', 'text']
fake_news_additional_cleaned = fake_news_additional[columns_to_keep]

# Rename 'published' column to 'date' and 'site_url' column to 'url'
fake_news_additional_cleaned = fake_news_additional_cleaned.rename(columns={'published': 'date', 'site_url': 'url'})

# Add 'label' column and set it to 0 (assuming this dataset contains fake news)
fake_news_additional_cleaned['label'] = 0
```

```
from pytz import utc

# Convert string dates to datetime objects with timezone information
fake_news_additional_cleaned['date'] = pd.to_datetime(fake_news_additional_cleaned['date'], format='%Y-%m-%dT%H:%M:%S.%f%z', error

# Convert timezone-aware datetime objects to UTC timezone
fake_news_additional_cleaned['date'] = fake_news_additional_cleaned['date'].apply(lambda x: x.astimezone(utc) if hasattr(x, 'tzinf

# Drop the time component and convert to string dtype
fake_news_additional_cleaned['date'] = pd.to_datetime(fake_news_additional_cleaned['date']).dt.date
```

```
fake_news_additional_cleaned['date'] = pd.to_datetime(fake_news_additional_cleaned['date'])
fake_news_additional_cleaned.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12999 entries, 0 to 12998
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype         
---  ------  --------------  -----         
 0   author  10575 non-null  object        
 1   url     12999 non-null  object        
 2   date    12999 non-null  datetime64[ns]
 3   title   12319 non-null  object        
 4   text    12953 non-null  object        
 5   label   12999 non-null  int64         
dtypes: datetime64[ns](1), int64(1), object(4)
memory usage: 609.5+ KB
```

```
# Merge with the existing 'data' DataFrame
data = pd.concat([data, fake_news_additional_cleaned], ignore_index=True)
data
```

	title	text	subject	date	label	publication
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	2017-12-31	1	NaN
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	2017-12-29	1	NaN
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	2017-12-31	1	NaN
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	2017-12-30	1	NaN

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 200467 entries, 0 to 200466
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   title        199785 non-null  object
 1   text         200421 non-null  object
 2   subject      44898 non-null   object
 3   date         197816 non-null  datetime64[ns]
 4   label        200467 non-null  int64
 5   publication  142570 non-null  object
 6   author       137269 non-null  object
 7   url          98558 non-null   object
dtypes: datetime64[ns](1), int64(1), object(6)
memory usage: 12.2+ MB
```

```
# https://www.cs.ucsb.edu/~william/data/liar_dataset.zip
# Load and preprocess LIAR dataset
liar_train = pd.read_csv('/content/train.tsv', sep='\t', header=None)
liar_test = pd.read_csv('/content/test.tsv', sep='\t', header=None)
liar_valid = pd.read_csv('/content/valid.tsv', sep='\t', header=None)

# Combine LIAR train, test, and valid datasets
liar_data = pd.concat([liar_train, liar_test, liar_valid], ignore_index=True)
liar_data
```

	0	1	2	3	4	5
0	2635.json	false	Says the Annies List political group supports ...	abortion	dwayne-bohac	State representative
1	10540.json	half-true	When did the decline of coal start? It started...	energy,history,job-accomplishments	scott-surovell	State delegate
2	324.json	mostly-true	Hillary Clinton agrees with John McCain "by vo...	foreign-policy	barack-obama	President
3	1123.json	false	Health care reform legislation is	health-care	blog-	NaN

```
# Assign column names
liar_data.columns = ['id', 'label', 'text', 'subject', 'speaker', 'job_title', 'state', 'party', 'barely_true', 'false', 'half_tru

# Rename the columns to match the existing data
```

```
liar_data_cleaned = liar_data.rename(columns={'context': 'url', 'speaker': 'author', 'job_title': 'publication'})

# Note that the Liar dataset does not have a 'title' or 'date' column
# We can create empty columns for these with None values
liar_data_cleaned['title'] = None
liar_data_cleaned['date'] = None

# Select the columns to keep
columns_to_keep = ['title', 'text', 'subject', 'date', 'label', 'publication', 'author', 'url']
liar_data_cleaned = liar_data_cleaned[columns_to_keep]

# Convert label to binary
liar_data_cleaned['label'] = liar_data_cleaned['label'].map({'true': 1, 'mostly-true': 1, 'half-true': 1, 'barely-true': 0, 'false
# Merge with the existing 'data' DataFrame
data = pd.concat([data, liar_data_cleaned], ignore_index=True)
data
```

	title	text	subject	date	label	pu
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	2017-12-31 00:00:00	1	
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	2017-12-29 00:00:00	1	
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	2017-12-31 00:00:00	1	
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	2017-12-30 00:00:00	1	

```
data['date'] = pd.to_datetime(data['date'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 213258 entries, 0 to 213257
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype
---  ------       --------------   -----
 0   title        199785 non-null  object
 1   text         213212 non-null  object
 2   subject      57687 non-null   object
 3   date         197816 non-null  datetime64[ns]
 4   label        213258 non-null  int64
 5   publication  151794 non-null  object
 6   author       150058 non-null  object
 7   url          111218 non-null  object
dtypes: datetime64[ns](1), int64(1), object(6)
memory usage: 13.0+ MB
```

```
# https://github.com/KaiDMML/FakeNewsNet
# Load FakeNewsNet datasets
gossipcop_fake = pd.read_csv('/content/FakeNewsNet-master/dataset/gossipcop_fake.csv')
gossipcop_real = pd.read_csv('/content/FakeNewsNet-master/dataset/gossipcop_real.csv')
politifact_fake = pd.read_csv('/content/FakeNewsNet-master/dataset/politifact_fake.csv')
politifact_real = pd.read_csv('/content/FakeNewsNet-master/dataset/politifact_real.csv')

# Rename 'news_url' column to 'url'
gossipcop_fake = gossipcop_fake.rename(columns={'news_url': 'url'})
gossipcop_real = gossipcop_real.rename(columns={'news_url': 'url'})
politifact_fake = politifact_fake.rename(columns={'news_url': 'url'})
politifact_real = politifact_real.rename(columns={'news_url': 'url'})

# Assign labels to FakeNewsNet datasets
gossipcop_fake['label'] = 0
gossipcop_real['label'] = 1
politifact_fake['label'] = 0
```

```
politifact_real['label'] = 1

# Select relevant columns in FakeNewsNet datasets
gossipcop_fake = gossipcop_fake[['title', 'label', 'url']]
gossipcop_real = gossipcop_real[['title', 'label', 'url']]
politifact_fake = politifact_fake[['title', 'label', 'url']]
politifact_real = politifact_real[['title', 'label', 'url']]

# Combine all datasets into a single dataframe
all_datasets = [data, gossipcop_fake, gossipcop_real, politifact_fake, politifact_real]
data = pd.concat(all_datasets, ignore_index=True)
data
```

	title	text	subject	date	label	publication
0	As U.S. budget fight looms, Republicans flip t...	WASHINGTON (Reuters) - The head of a conservat...	politicsNews	2017-12-31	1	NaN
1	U.S. military to accept transgender recruits o...	WASHINGTON (Reuters) - Transgender people will...	politicsNews	2017-12-29	1	NaN
2	Senior U.S. Republican senator: 'Let Mr. Muell...	WASHINGTON (Reuters) - The special counsel inv...	politicsNews	2017-12-31	1	NaN
3	FBI Russia probe helped by Australian diplomat...	WASHINGTON (Reuters) - Trump campaign adviser ...	politicsNews	2017-12-30	1	NaN

```
gossipcop_fake
```

	title	label	url
0	Did Miley Cyrus and Liam Hemsworth secretly ge...	0	www.dailymail.co.uk/tvshowbiz/article-5874213/...
1	Paris Jackson & Cara Delevingne Enjoy Night Ou...	0	hollywoodlife.com/2018/05/05/paris-jackson-car...
2	Celebrities Join Tax March in Protest of Donal...	0	variety.com/2017/biz/news/tax-march-donald-tru...
3	Cindy Crawford's daughter Kaia Gerber wears a ...	0	www.dailymail.co.uk/femail/article-3499192/Do-...
4	Full List of 2018 Oscar Nominations – Variety	0	variety.com/2018/film/news/list-2018-oscar-nom...
...	...	...	...
5318	September 11: Celebrities Remember 9/11 (TWEETS)	0	www.huffingtonpost.com/2012/09/11/september-11...
5319	NASCAR owners threaten to fire drivers who pro...	0	www.dailymail.co.uk/news/article-4915674/NASCA...

```
# Shuffle the combined data
data = data.sample(frac=1).reset_index(drop=True)
labels = data.label
data
```

	title	text	subject	date	label	publication	aut
0	Kim Kardashian Gets Hysterical Over Lost Earri...	NaN	NaN	NaT	0	NaN	
1	None	Barack Obama "extended health care for	military	NaT	1	President	bar ob

```
data['text'] = data['text'].fillna('')  # Fill missing values with an empty string
data['title'] = data['title'].fillna('')  # Fill missing values with an empty string
```

2	Manning	Feb. 29, 1996,	NaN	2016-	1	New York Post	Mar

```
from textblob import TextBlob

# Add sentiment analysis as a feature
def get_sentiment(text):
    sentiment = TextBlob(text).sentiment.polarity
    return sentiment

data['sentiment'] = data['text'].apply(get_sentiment)
data['title_sentiment'] = data['title'].apply(get_sentiment)
data
```

```
# Add sentiment analysis as a feature
def get_sentiment(text):
    sentiment = TextBlob(text).sentiment.polarity
    return sentiment

data['title_sentiment'] = data['title'].apply(get_sentiment)
data
```

	title	text	subject	date	label	publication	aut
0	Kim Kardashian Gets Hysterical Over Lost Earri...		NaN	NaT	0	NaN	
1		Barack Obama "extended health care for wounded...	military	NaT	1	President	bar ob
2	Perfect Peyton Manning witness emerges in sex ...	The haze over Feb. 29, 1996, may never be comp...	NaN	2016-03-02	1	New York Post	Mar San
3		The $18.8 billion in funding for K-12 educatio	education	NaT	1	Governor	rick-

```
data[(data['text'].isna() & data['title'].isna())]
```

	title	text	subject	date	label	publication	author	url	sentiment	titl

```
data
```

	title	text	subject	date	label	publication	aut
0	Kim Kardashian Gets Hysterical Over Lost Earri...		NaN	NaT	0	NaN	
1		Barack Obama "extended health care for wounded...	military	NaT	1	President	bar ob
	Perfect						

```
# Preprocess the data using NLTK and regular expressions
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer

# Download NLTK stopwords
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
stemmer = PorterStemmer()

# Define a text preprocessing function
def preprocess_text(text):
    text = str(text)
    text = text.lower()
    text = re.sub(r'\W', ' ', text)  # Remove non-word characters
    text = re.sub(r'\s+', ' ', text)  # Remove extra spaces
    text = ' '.join([stemmer.stem(word) for word in text.split() if word not in stop_words])  # Stemming and stopword removal
    return text

# Apply the preprocessing function to the 'text' column
data['text'] = data['text'].apply(preprocess_text)

# Tokenize the text and split the dataset into training and testing sets
max_words = 10000
tokenizer = Tokenizer(num_words=max_words)
tokenizer.fit_on_texts(data['text'])
sequences = tokenizer.texts_to_sequences(data['text'])
word_index = tokenizer.word_index

# Split the data into training and testing sets
x_train, x_test, y_train, y_test = train_test_split(sequences, data['label'], test_size=0.2, random_state=42)

# Pad the sequences to have equal length
max_sequence_length = 500
x_train = pad_sequences(x_train, maxlen=max_sequence_length)
x_test = pad_sequences(x_test, maxlen=max_sequence_length)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
x_train
```

```
array([[   0,    0,    0, ..., 1103,   34, 1883],
       [   0,    0,    0, ..., 1438,  435,  872],
       [   0,    0,    0, ..., 1949, 1467, 1535],
       ...,
       [   0,    0,    0, ..., 3270,  949, 1555],
       [   0,    0,    0, ...,    0,    0,    0],
       [   0,    0,    0, ..., 5267,  575,  605]], dtype=int32)
```

```
# Calculate the size of the dataset in megabytes
total_size_in_bytes = data.memory_usage(index=True, deep=True).sum()
total_size_in_megabytes = total_size_in_bytes / (1024 * 1024)
print("Size of the whole dataset in megabytes (MB):", total_size_in_megabytes)
```

```
Size of the whole dataset in megabytes (MB): 568.5186910629272
```

```
# Save the data DataFrame to a CSV file in Google Drive
data.to_csv('/content/gdrive/MyDrive/data.csv', index=False)
```

```
# Save the x_train and x_test arrays to numpy binary files in Google Drive
np.save('/content/gdrive/MyDrive/x_train.npy', x_train)
np.save('/content/gdrive/MyDrive/x_test.npy', x_test)
```

News Classification with Pre-trained GloVe Embeddings, LSTM, and Hyperparameter Tuning using Keras Tuner

```
# Load pre-trained word embeddings (GloVe)
!wget http://nlp.stanford.edu/data/glove.6B.zip
!unzip glove.6B.zip
```

```
--2023-05-06 16:34:43--  http://nlp.stanford.edu/data/glove.6B.zip
Resolving nlp.stanford.edu (nlp.stanford.edu)... 171.64.67.140
Connecting to nlp.stanford.edu (nlp.stanford.edu)|171.64.67.140|:80... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://nlp.stanford.edu/data/glove.6B.zip [following]
--2023-05-06 16:34:43--  https://nlp.stanford.edu/data/glove.6B.zip
Connecting to nlp.stanford.edu (nlp.stanford.edu)|171.64.67.140|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://downloads.cs.stanford.edu/nlp/data/glove.6B.zip [following]
--2023-05-06 16:34:44--  https://downloads.cs.stanford.edu/nlp/data/glove.6B.zip
Resolving downloads.cs.stanford.edu (downloads.cs.stanford.edu)... 171.64.64.22
Connecting to downloads.cs.stanford.edu (downloads.cs.stanford.edu)|171.64.64.22|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 862182613 (822M) [application/zip]
Saving to: ‘glove.6B.zip’

glove.6B.zip        100%[===================>] 822.24M  5.01MB/s    in 2m 38s

2023-05-06 16:37:23 (5.19 MB/s) - ‘glove.6B.zip’ saved [862182613/862182613]

Archive:  glove.6B.zip
  inflating: glove.6B.50d.txt
  inflating: glove.6B.100d.txt
  inflating: glove.6B.300d.txt
```

```
# Define embedding dimensions and create the embeddings index
embedding_dim = 100
embeddings_index = {}
with open('glove.6B.100d.txt') as f:
    for line in f:
        values = line.split()
        word = values[0]
        coefs = np.asarray(values[1:], dtype='float32')
        embeddings_index[word] = coefs

# Create the embedding matrix
embedding_matrix = np.zeros((max_words, embedding_dim))
for word, i in word_index.items():
    if i < max_words:
        embedding_vector = embeddings_index.get(word)
        if embedding_vector is not None:
            embedding_matrix[i] = embedding_vector
```

```
# Import necessary layers and functions from Keras
from tensorflow.keras.layers import BatchNormalization
from tensorflow.keras.layers import Reshape
```

```
# Define the model-building function for hyperparameter tuning
def build_model(hp):
    model = Sequential()
    model.add(Embedding(max_words, embedding_dim, input_length=max_sequence_length, weights=[embedding_matrix], trainable=False))
    model.add(Dropout(hp.Float('dropout_1', 0.1, 0.5, step=0.1)))

    # Add convolutional layers
    for i in range(hp.Int('num_conv_layers', 1, 3)):
        model.add(Conv1D(hp.Int(f'conv_{i+1}_filters', 32, 128, step=32),
                         hp.Int(f'conv_{i+1}_kernel_size', 3, 7, step=2),
                         activation='relu', padding='same'))
        model.add(BatchNormalization())
```

```
    model.add(GlobalMaxPooling1D())

    # Add dense layers
    for i in range(hp.Int('num_dense_layers', 1, 3)):
        model.add(Dense(hp.Int(f'dense_{i+1}_units', 64, 256, step=64), activation='relu'))
        model.add(Dropout(hp.Float(f'dropout_{i+2}', 0.1, 0.5, step=0.1)))

    # Add a Reshape layer to add the extra dimension
    model.add(Reshape((-1, 1)))

    # Add bidirectional LSTM and GRU layers
    model.add(Bidirectional(LSTM(hp.Int('lstm_units', 32, 128, step=32), return_sequences=True)))
    model.add(GRU(hp.Int('gru_units', 16, 64, step=16)))

    # Add output layer
    model.add(Dense(1, activation='sigmoid'))

    # Compile the model
    model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model

# Set up the random search tuner
tuner = RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=10,
    executions_per_trial=1,
    directory='hyperparameter_tuning',
    project_name='news_classification'
)

# Display the tuner's search space summary
tuner.search_space_summary()

# Run the hyperparameter search
tuner.search(x_train, y_train, epochs=6, validation_data=(x_test, y_test), batch_size=128, verbose=1)
```

```
tuner = RandomSearch(
    build_model,
    objective='val_accuracy',
    max_trials=10,
    executions_per_trial=1,
    directory='hyperparameter_tuning',
    project_name='news_classification'
)

tuner.search_space_summary()

tuner.search(x_train, y_train, epochs=6, validation_data=(x_test, y_test), batch_size=128, verbose=1)
```

```
Trial 10 Complete [00h 04m 17s]
val_accuracy: 0.8970205783843994

Best val_accuracy So Far: 0.8975915312767029
Total elapsed time: 00h 30m 20s
```

```
# Retrieve the best model
best_model = tuner.get_best_models(num_models=1)[0]

# Train the best model with the full dataset
best_model.fit(x_train, y_train, validation_data=(x_test, y_test), epochs=10, batch_size=128)

# Make predictions on the testing set
y_pred_raw = best_model.predict(x_test)
y_pred = [round(pred[0]) for pred in y_pred_raw]

# Calculate accuracy, confusion matrix, and classification report
score = accuracy_score(y_test, y_pred)
print(f'Accuracy: {round(score*100, 2)}%')
```

```
confusion_matrix = confusion_matrix(y_test, y_pred)
print(f'Confusion Matrix:\n{confusion_matrix}')

classification_report = classification_report(y_test, y_pred)
print(f'Classification Report:\n{classification_report}')
```

```
Epoch 1/10
1478/1478 [==============================] - 30s 15ms/step - loss: 0.2942 - accuracy: 0.8981 - val_loss: 0.2963 - val_accurac
Epoch 2/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2936 - accuracy: 0.8986 - val_loss: 0.2943 - val_accurac
Epoch 3/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2929 - accuracy: 0.8991 - val_loss: 0.2956 - val_accurac
Epoch 4/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2928 - accuracy: 0.8993 - val_loss: 0.2946 - val_accurac
Epoch 5/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2918 - accuracy: 0.8999 - val_loss: 0.2958 - val_accurac
Epoch 6/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2917 - accuracy: 0.9003 - val_loss: 0.2947 - val_accurac
Epoch 7/10
1478/1478 [==============================] - 22s 15ms/step - loss: 0.2908 - accuracy: 0.9008 - val_loss: 0.2949 - val_accurac
Epoch 8/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2907 - accuracy: 0.9014 - val_loss: 0.2943 - val_accurac
Epoch 9/10
1478/1478 [==============================] - 21s 14ms/step - loss: 0.2906 - accuracy: 0.9017 - val_loss: 0.2968 - val_accurac
Epoch 10/10
1478/1478 [==============================] - 22s 15ms/step - loss: 0.2896 - accuracy: 0.9023 - val_loss: 0.2960 - val_accurac
1478/1478 [==============================] - 7s 4ms/step
Accuracy: 89.73%
Confusion Matrix:
[[ 5093  4492]
 [  363 37343]]
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.53      0.68      9585
           1       0.89      0.99      0.94     37706

    accuracy                           0.90     47291
   macro avg       0.91      0.76      0.81     47291
weighted avg       0.90      0.90      0.89     47291
```

```
from keras.models import Sequential
from keras.layers import Embedding, Dropout, Conv1D, BatchNormalization, GlobalMaxPooling1D, Dense, Reshape, Bidirectional, LSTM,
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ModelCheckpoint

def build_model(hp):
    model = Sequential()
    model.add(Embedding(max_words, embedding_dim, input_length=max_sequence_length, weights=[embedding_matrix], trainable=False))
    model.add(Dropout(hp.Float('dropout_1', 0.1, 0.5, step=0.1)))

    for i in range(hp.Int('num_conv_layers', 1, 3)):
        model.add(Conv1D(hp.Int(f'conv_{i+1}_filters', 32, 128, step=32),
                         hp.Int(f'conv_{i+1}_kernel_size', 3, 7, step=2),
                         activation='relu', padding='same'))
        model.add(BatchNormalization())

    model.add(GlobalMaxPooling1D())

    for i in range(hp.Int('num_dense_layers', 1, 3)):
        model.add(Dense(hp.Int(f'dense_{i+1}_units', 64, 256, step=64), activation='relu'))
        model.add(Dropout(hp.Float(f'dropout_{i+2}', 0.1, 0.5, step=0.1)))

    # Add a Reshape layer to add the extra dimension
    model.add(Reshape((-1, 1)))

    model.add(Bidirectional(LSTM(hp.Int('lstm_units', 32, 128, step=32), return_sequences=True)))
    model.add(GRU(hp.Int('gru_units', 16, 64, step=16)))
    model.add(Dense(1, activation='sigmoid'))

    optimizer = Adam(learning_rate=hp.Float('learning_rate', 1e-4, 1e-2, sampling='log'))
    model.compile(loss='binary_crossentropy', optimizer=optimizer, metrics=['accuracy'])
    return model
```

```
tuner = RandomSearch(
    build_model,
```

```
    objective='val_accuracy',
    max_trials=5,
    executions_per_trial=3,
    directory='random_search_dir',
    project_name='text_classification'
)

early_stopping = EarlyStopping(monitor='val_loss', patience=3)
model_checkpoint = ModelCheckpoint('best_model.h5', monitor='val_loss', save_best_only=True)
callbacks = [early_stopping, model_checkpoint]

tuner.search(x_train, y_train, epochs=6, validation_data=(x_test, y_test), batch_size=128, callbacks=callbacks, verbose=1)
```

```
Trial 5 Complete [00h 09m 16s]
val_accuracy: 0.8952443401018778

Best val_accuracy So Far: 0.8964707851409912
Total elapsed time: 01h 11m 25s
```

Hyperparameter Optimization and Evaluation of Machine Learning Models using RandomizedSearchCV

```
import pandas as pd
import numpy as np

# Read the data DataFrame from the CSV file
data = pd.read_csv('/content/gdrive/MyDrive/data.csv')

# Read the x_train and x_test arrays from the numpy binary files
x_train = np.load('/content/gdrive/MyDrive/x_train.npy')
x_test = np.load('/content/gdrive/MyDrive/x_test.npy')
# Replace missing values with empty strings
data['text'].fillna('', inplace=True)
```

```
data
```

	title	text	subject	date	label	publication	author
0	Kim Kardashian Gets Hysterical Over Lost Earri...		NaN	NaN	0	NaN	NaN
1	NaN	barack obama extend health care wound troop ne...	military	NaN	1	President	barack-obama
2	Perfect Peyton Manning witness emerges in sex ...	haze feb 29 1996 may never complet clear anoth...	NaN	2016-03-02	1	New York Post	Mark W. Sanchez
3	NaN	18 8 billion fund k 12 educ fund highest flori...	education	NaN	1	Governor	rick-scott

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.svm import LinearSVC
from sklearn.ensemble import RandomForestClassifier
from scipy.sparse import hstack
from sklearn.model_selection import RandomizedSearchCV
```

```
# Function to optimize and evaluate a given model
def optimize_and_evaluate(model_name, estimator, params, X_train, y_train, X_test, y_test):
    print(f"Optimizing {model_name}...")  # Indicate the model being optimized
    # Perform randomized search for hyperparameter optimization
    randomized_search = RandomizedSearchCV(estimator=estimator, param_distributions=params, n_iter=5, scoring='accuracy', cv=3, ra
    randomized_search.fit(X_train, y_train)

    # Get the best parameters, score, and estimator from the randomized search
    best_params = randomized_search.best_params_
    best_score = randomized_search.best_score_
    best_estimator = randomized_search.best_estimator_

    # Evaluate the optimized model on the test set
    y_pred = best_estimator.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    confusion_mat = confusion_matrix(y_test, y_pred)
    class_report = classification_report(y_test, y_pred)

    # Print the results
    print(f"Best parameters: {best_params}")
    print(f"Best cross-validated accuracy: {best_score:.2f}")
    print(f"Test set accuracy: {accuracy:.2f}")
    print(f"Confusion Matrix:\n{confusion_mat}")
    print(f"Classification Report:\n{class_report}")
    print("="*50)
```

```
# Use TfidfVectorizer for feature extraction
vectorizer = TfidfVectorizer(stop_words='english', ngram_range=(1, 2), max_features=5000)
X = vectorizer.fit_transform(data['text'])

# Combine the sparse matrix with the sentiment column
X = hstack((X, data['sentiment'].values[:, None])).tocsr()

y = data['label']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.linear_model import LogisticRegression
# Logistic Regression
lr_params = {
    'C': [0.1, 1, 10],
    'penalty': ['l1', 'l2'],
    'solver': ['liblinear', 'saga'],
    'max_iter': [1000, 2000, 5000]
}

optimize_and_evaluate('Logistic Regression', LogisticRegression(), lr_params, X_train, y_train, X_test, y_test)
```

```
Optimizing Logistic Regression...
Best parameters: {'solver': 'saga', 'penalty': 'l1', 'max_iter': 1000, 'C': 1}
Best cross-validated accuracy: 0.90
Test set accuracy: 0.90
Confusion Matrix:
[[ 5522  3987]
 [  843 36939]]
Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.58      0.70      9509
           1       0.90      0.98      0.94     37782

    accuracy                           0.90     47291
   macro avg       0.89      0.78      0.82     47291
weighted avg       0.90      0.90      0.89     47291

==================================================
```

```
# Linear Support Vector Machine
lsvc_params = {
    'C': [0.1, 1, 10]
}
optimize_and_evaluate('Linear Support Vector Machine', LinearSVC(), lsvc_params, X_train, y_train, X_test, y_test)
```

```
Optimizing Linear Support Vector Machine...
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_search.py:305: UserWarning: The total space of parameters 3
  warnings.warn(
Best parameters: {'C': 1}
Best cross-validated accuracy: 0.90
Test set accuracy: 0.90
Confusion Matrix:
[[ 5555  3954]
 [  917 36865]]
Classification Report:
              precision    recall  f1-score   support

           0       0.86      0.58      0.70      9509
           1       0.90      0.98      0.94     37782

    accuracy                           0.90     47291
   macro avg       0.88      0.78      0.82     47291
weighted avg       0.89      0.90      0.89     47291

==================================================
```

```
from sklearn.ensemble import RandomForestClassifier

# Random Forest
rf_params = {
    'n_estimators': [100, 200, 300],
    'max_depth': [None, 10, 20]
}
optimize_and_evaluate('Random Forest', RandomForestClassifier(), rf_params, X_train, y_train, X_test, y_test)
```

```
Optimizing Random Forest...
Best parameters: {'n_estimators': 100, 'max_depth': None}
Best cross-validated accuracy: 0.89
Test set accuracy: 0.90
Confusion Matrix:
[[ 5388  4121]
 [  462 37320]]
Classification Report:
              precision    recall  f1-score   support

           0       0.92      0.57      0.70      9509
           1       0.90      0.99      0.94     37782

    accuracy                           0.90     47291
   macro avg       0.91      0.78      0.82     47291
weighted avg       0.90      0.90      0.89     47291

==================================================
```

Fine-tuning Pretrained Language Models for News Classification

```
import pandas as pd
import numpy as np

# Read the data DataFrame from the CSV file
data = pd.read_csv('/content/gdrive/MyDrive/data.csv')

# Read the x_train and x_test arrays from the numpy binary files
x_train = np.load('/content/gdrive/MyDrive/x_train.npy')
x_test = np.load('/content/gdrive/MyDrive/x_test.npy')
# Replace missing values with empty strings
data['text'].fillna('', inplace=True)
```

```
import gc
gc.collect()
```

```
45
```

```
x_train, x_test, y_train, y_test = train_test_split(data['text'], data['label'], test_size=0.2, random_state=42)
```

```
# Convert the dataset to BERT input format
def convert_data_to_examples(data, labels):
    InputExamples = []
    for i in range(len(data)):
        example = InputExample(guid=None, text_a=data.iloc[i], text_b=None, label=labels.iloc[i])
        InputExamples.append(example)
    return InputExamples
```

```
# Updated `convert_examples_to_tf_dataset` function
def convert_examples_to_tf_dataset(examples, tokenizer, max_length=128):
    features = []
    for e in examples:
        input_dict = tokenizer.encode_plus(e.text_a, add_special_tokens=True, max_length=max_length, return_token_type_ids=True, r
        input_ids, token_type_ids, attention_mask = (input_dict["input_ids"], input_dict["token_type_ids"], input_dict["attention_
        features.append(InputFeatures(input_ids=input_ids, attention_mask=attention_mask, token_type_ids=token_type_ids, label=e.l

    def gen():
        for f in features:
            yield ({"input_ids": f.input_ids, "attention_mask": f.attention_mask, "token_type_ids": f.token_type_ids}, f.label)

    return tf.data.Dataset.from_generator(gen, ({"input_ids": tf.int32, "attention_mask": tf.int32, "token_type_ids": tf.int32}, t
```

```
from transformers import BertTokenizer, TFBertForSequenceClassification
from transformers import InputExample, InputFeatures
import tensorflow as tf

tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
model = TFBertForSequenceClassification.from_pretrained("bert-base-uncased", num_labels=2)
```

```
Downloading (…)solve/main/vocab.txt: 100%  232k/232k [00:00<00:00, 314kB/s]
Downloading (…)okenizer_config.json: 100%  28.0/28.0 [00:00<00:00, 2.25kB/s]
Downloading (…)lve/main/config.json: 100%  570/570 [00:00<00:00, 46.1kB/s]
```

```
import pandas as pd

# Replace NaN values in input data
x_train = x_train.fillna("")
x_test = x_test.fillna("")


train_examples = convert_data_to_examples(x_train, y_train)
test_examples = convert_data_to_examples(x_test, y_test)

train_dataset = convert_examples_to_tf_dataset(train_examples, tokenizer)
test_dataset = convert_examples_to_tf_dataset(test_examples, tokenizer)
```

```
# Freeze all layers except the last few layers
for layer in model.layers[:-4]:
    layer.trainable = False

# Unfreeze the last few layers
for layer in model.layers[-4:]:
    layer.trainable = True

# Compile and train the model with a smaller learning rate
optimizer = tf.keras.optimizers.Adam(learning_rate=1e-6)
model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

```
model.fit(train_dataset.shuffle(100).batch(16), epochs=10, batch_size=16, validation_data=test_dataset.shuffle(100).batch(16))
```

```
Epoch 1/10
11823/11823 [==============================] - 986s 79ms/step - loss: 0.5791 - accuracy: 0.7725 - val_loss: 0.4170 - val_accu
Epoch 2/10
11823/11823 [==============================] - 863s 73ms/step - loss: 0.4237 - accuracy: 0.8199 - val_loss: 0.3930 - val_accu
Epoch 3/10
 4059/11823 [=========>....................] - ETA: 8:34 - loss: 0.3923 - accuracy: 0.8356
```

```
# Freeze all layers except the last few layers
for layer in model.layers[:-4]:
    layer.trainable = False

# Unfreeze the last few layers
for layer in model.layers[-4:]:
    layer.trainable = True

# Compile and train the model with a smaller learning rate
optimizer = tf.keras.optimizers.Adam(learning_rate=1e-6)
model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])
model.fit(train_dataset.shuffle(100).batch(16), epochs=4, batch_size=16, validation_data=test_dataset.shuffle(100).batch(16))
```

```
Epoch 1/4
11823/11823 [==============================] - 959s 77ms/step - loss: 0.4700 - accuracy: 0.8044 - val_loss: 0.3940 - val_accu
Epoch 2/4
11823/11823 [==============================] - 817s 69ms/step - loss: 0.3869 - accuracy: 0.8446 - val_loss: 0.3415 - val_accu
Epoch 3/4
11823/11823 [==============================] - 812s 69ms/step - loss: 0.3475 - accuracy: 0.8658 - val_loss: 0.3005 - val_accu
Epoch 4/4
11823/11823 [==============================] - 815s 69ms/step - loss: 0.3246 - accuracy: 0.8717 - val_loss: 0.3045 - val_accu
<keras.callbacks.History at 0x7fd4a56849d0>
```

```
import pandas as pd

def convert_examples_to_tf_dataset(examples, tokenizer, max_length=128):
    features = []
    for e in examples:
        input_dict = tokenizer.encode_plus(e.text_a, add_special_tokens=True, max_length=max_length, return_attention_mask=True, p
        input_ids, attention_mask = (input_dict["input_ids"], input_dict["attention_mask"])
        features.append(InputFeatures(input_ids=input_ids, attention_mask=attention_mask, label=e.label))

    def gen():
        for f in features:
            yield ({"input_ids": f.input_ids, "attention_mask": f.attention_mask}, f.label)

    return tf.data.Dataset.from_generator(gen, ({"input_ids": tf.int32, "attention_mask": tf.int32}, tf.int64), ({"input_ids": tf.
```

```
from transformers import BertTokenizer, TFBertForSequenceClassification, AutoTokenizer, TFAutoModelForSequenceClassification
import tensorflow as tf

MODEL_NAME = "roberta-base"

tokenizer = AutoTokenizer.from_pretrained(MODEL_NAME)
model = TFAutoModelForSequenceClassification.from_pretrained(MODEL_NAME, num_labels=2)

train_dataset = convert_examples_to_tf_dataset(train_examples, tokenizer)
test_dataset = convert_examples_to_tf_dataset(test_examples, tokenizer)

optimizer = tf.keras.optimizers.Adam(learning_rate=1e-5)

model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])

model.fit(train_dataset.shuffle(100).batch(32), epochs=8, batch_size=32, validation_data=test_dataset.shuffle(100).batch(32))
```

```
Downloading                                                481/481 [00:00<00:00,
(…)lve/main/config.json: 100%                              41.0kB/s]
Downloading                                                899k/899k [00:00<00:00,
(…)olve/main/vocab.json: 100%                              13.7MB/s]
Downloading                                                456k/456k [00:00<00:00,
(…)olve/main/merges.txt: 100%                              6.45MB/s]
Downloading                                                1.36M/1.36M [00:00<00:00,
(…)/main/tokenizer.json: 100%                              40.3MB/s]
Downloading tf_model.h5:                                   657M/657M [00:03<00:00,
100%                                                       216MB/s]
All model checkpoint layers were used when initializing TFRobertaForSequenc

Some layers of TFRobertaForSequenceClassification were not initialized from
You should probably TRAIN this model on a down-stream task to be able to us
Epoch 1/8
5912/5912 [==============================] - 762s 121ms/step - loss: 0.6948
Epoch 2/8
5912/5912 [==============================] - 633s 107ms/step - loss: 0.6931
Epoch 3/8
5912/5912 [==============================] - 628s 106ms/step - loss: 0.6931
Epoch 4/8
 341/5912 [>.............................] - ETA: 8:53 - loss: 0.6931 - acc
---------------------------------------------------------------------------
-
KeyboardInterrupt                         Traceback (most recent call
last)
```

```
from transformers import BertTokenizer, TFBertForSequenceClassification
from transformers import InputExample, InputFeatures
import tensorflow as tf

# Use the base pre-trained model
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
model = TFBertForSequenceClassification.from_pretrained("bert-base-uncased", num_labels=2)

# Increase the maximum length of input sequences
max_length = 256

train_examples = convert_data_to_examples(x_train, y_train)
test_examples = convert_data_to_examples(x_test, y_test)

train_dataset = convert_examples_to_tf_dataset(train_examples, tokenizer, max_length)
test_dataset = convert_examples_to_tf_dataset(test_examples, tokenizer, max_length)

# Learning rate scheduler
lr_scheduler = tf.keras.optimizers.schedules.ExponentialDecay(
    initial_learning_rate=5e-5,
    decay_steps=10000,
    decay_rate=0.9)

# Compile and train the model with a smaller learning rate
optimizer = tf.keras.optimizers.Adam(learning_rate=lr_scheduler)
model.compile(optimizer=optimizer, loss='sparse_categorical_crossentropy', metrics=['accuracy'])

# Early stopping
early_stopping = tf.keras.callbacks.EarlyStopping(monitor='val_loss', patience=2, restore_best_weights=True)

# Train for more epochs and monitor the validation loss
model.fit(train_dataset.shuffle(100).batch(16),
          epochs=16,
          batch_size=16,
          validation_data=test_dataset.shuffle(100).batch(16),
          callbacks=[early_stopping])
```

```
Downloading                                    232k/232k [00:00<00:00,
(…)solve/main/vocab.txt: 100%                  939kB/s]
Downloading                                    28.0/28.0 [00:00<00:00,
(…)okenizer_config.json: 100%                  2.16kB/s]
Downloading                                    570/570 [00:00<00:00,
(…)lve/main/config.json: 100%                  50.4kB/s]
Downloading tf_model.h5:                       536M/536M [00:01<00:00,
100%                                           474MB/s]
All model checkpoint layers were used when initializing TFBertForSequenceCl

Some layers of TFBertForSequenceClassification were not initialized from th
You should probably TRAIN this model on a down-stream task to be able to us
Epoch 1/16
11823/11823 [==============================] - 1567s 128ms/step - loss: 0.6
Epoch 2/16
11823/11823 [==============================] - 1400s 118ms/step - loss: 0.6
Epoch 3/16
 3507/11823 [=======>......................] - ETA: 14:54 - loss: 0.6931 -
---------------------------------------------------------------------------
-
KeyboardInterrupt                         Traceback (most recent call
last)
<ipython-input-12-72d92ce3f405> in <cell line: 32>()
     30
```

```
from sklearn.utils.class_weight import compute_class_weight

unique_classes = np.unique(y_train)
class_weights = compute_class_weight('balanced', classes=unique_classes, y=y_train)
class_weights = dict(enumerate(class_weights))

# Train for more epochs, apply class weights, and monitor the validation loss
model.fit(train_dataset.shuffle(100).batch(16),
          epochs=20,
          batch_size=32,
          validation_data=test_dataset.shuffle(100).batch(16),
          callbacks=[early_stopping],
          class_weight=class_weights)
```

```
Epoch 1/20
10024/10024 [==============================] - 1194s 118ms/step - loss: 0.6931 - accuracy: 0.7722 - val_loss: 0.6931 - val_ac
Epoch 2/20
10024/10024 [==============================] - 1186s 118ms/step - loss: 0.6931 - accuracy: 0.7736 - val_loss: 0.6931 - val_ac
Epoch 3/20
10024/10024 [==============================] - 1181s 118ms/step - loss: 0.6931 - accuracy: 0.7728 - val_loss: 0.6931 - val_ac
<keras.callbacks.History at 0x7fcc74512920>
```

Executing (34m 43s) <cell line: 12> › error_handler() › fit() › error_handler() › __call__() › _call() › __call__() › _call_flat() › call() › quick_execute()