Russia Ukraine Invasion Sentiment Analysis

Data 602: Final Project

Russia Ukraine invasion took place on 24th Feburary'2022. In this notebook we have tried to analye the sentiment of the people around the world by focusing on the news which is broadcasted on this subject across the globe. The project proceeds with the flow mentioned below:

1. Installing and Importing the required libraries.
2. Data scraping through pygoogle news.
3. Data Cleaning and Data Wrangling.
4. Basic analysis on the data.
5. Sentiment Analysis on all the 3 datasets.
6. Model training through pipeline.
7. Model Training through Deep learning approach.
8. Model Training through Transformer using BERT model
9. Conclusion.
10. Future Work.

Installing libraries

```
!pip install pygooglenews
!pip install snorkel
!pip install textblob
!pip install transformers
```

```
Collecting pygooglenews
  Downloading pygooglenews-0.1.2-py3-none-any.whl (10 kB)
Collecting requests<3.0.0,>=2.24.0
  Downloading requests-2.27.1-py2.py3-none-any.whl (63 kB)
     |████████████████████████████████| 63 kB 652 kB/s
Collecting dateparser<0.8.0,>=0.7.6
  Downloading dateparser-0.7.6-py2.py3-none-any.whl (362 kB)
     |████████████████████████████████| 362 kB 10.0 MB/s
Collecting feedparser<6.0.0,>=5.2.1
  Downloading feedparser-5.2.1.zip (1.2 MB)
     |████████████████████████████████| 1.2 MB 42.3 MB/s
Collecting beautifulsoup4<5.0.0,>=4.9.1
  Downloading beautifulsoup4-4.11.1-py3-none-any.whl (128 kB)
     |████████████████████████████████| 128 kB 63.2 MB/s
Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: pytz in /usr/local/lib/python3.7/dist-packages (from d
Requirement already satisfied: tzlocal in /usr/local/lib/python3.7/dist-packages (fro
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: regex!=2019.02.19 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.7/
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (fr
Building wheels for collected packages: feedparser
  Building wheel for feedparser (setup.py) ... done
  Created wheel for feedparser: filename=feedparser-5.2.1-py3-none-any.whl size=44952
  Stored in directory: /root/.cache/pip/wheels/29/bf/46/b4a597d435d3aee6c2fa583824897
Successfully built feedparser
Installing collected packages: requests, feedparser, dateparser, beautifulsoup4, pygo
  Attempting uninstall: requests
    Found existing installation: requests 2.23.0
    Uninstalling requests-2.23.0:
      Successfully uninstalled requests-2.23.0
  Attempting uninstall: beautifulsoup4
    Found existing installation: beautifulsoup4 4.6.3
    Uninstalling beautifulsoup4-4.6.3:
      Successfully uninstalled beautifulsoup4-4.6.3
ERROR: pip's dependency resolver does not currently take into account all the package
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.27.1 which is i
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompa
Successfully installed beautifulsoup4-4.11.1 dateparser-0.7.6 feedparser-5.2.1 pygoog
Collecting snorkel
  Downloading snorkel-0.9.8-py3-none-any.whl (103 kB)
     |████████████████████████████████| 103 kB 5.4 MB/s
Requirement already satisfied: networkx<2.7,>=2.2 in /usr/local/lib/python3.7/dist-pa
Collecting scikit-learn<0.25.0,>=0.20.2
  Downloading scikit_learn-0.24.2-cp37-cp37m-manylinux2010_x86_64.whl (22.3 MB)
     |████████████████████████████████| 22.3 MB 19.5 MB/s
Collecting tensorboard<2.7.0,>=2.0.0
  Downloading tensorboard-2.6.0-py3-none-any.whl (5.6 MB)
     |████████████████████████████████| 5.6 MB 38.1 MB/s
Collecting munkres>=1.0.6
  Downloading munkres-1.1.4-py2.py3-none-any.whl (7.0 kB)
Requirement already satisfied: tqdm<5.0.0,>=4.33.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: scipy<2.0.0,>=1.2.0 in /usr/local/lib/python3.7/dist-p
```

```
Requirement already satisfied: torch<2.0.0,>=1.2.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: pandas<2.0.0,>=1.0.0 in /usr/local/lib/python3.7/dist-
Collecting numpy<1.20.0,>=1.16.5
  Downloading numpy-1.19.5-cp37-cp37m-manylinux2010_x86_64.whl (14.8 MB)
     |████████████████████████████████| 14.8 MB 42.6 MB/s
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dis
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: google-auth<2,>=1.6.3 in /usr/local/lib/python3.7/dist
Requirement already satisfied: absl-py>=0.4 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: protobuf>=3.6.0 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: grpcio>=1.24.3 in /usr/local/lib/python3.7/dist-packag
Requirement already satisfied: requests<3,>=2.21.0 in /usr/local/lib/python3.7/dist-p
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: setuptools>=41.0.0 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: wheel>=0.26 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: google-auth-oauthlib<0.5,>=0.4.1 in /usr/local/lib/pyt
Requirement already satisfied: werkzeug>=0.11.15 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: tensorboard-plugin-wit>=1.6.0 in /usr/local/lib/python
Requirement already satisfied: tensorboard-data-server<0.7.0,>=0.6.0 in /usr/local/li
Requirement already satisfied: rsa<5,>=3.1.4 in /usr/local/lib/python3.7/dist-package
Requirement already satisfied: cachetools<5.0,>=2.0.0 in /usr/local/lib/python3.7/dis
Requirement already satisfied: pyasn1-modules>=0.2.1 in /usr/local/lib/python3.7/dist
Requirement already satisfied: requests-oauthlib>=0.7.0 in /usr/local/lib/python3.7/d
Requirement already satisfied: importlib-metadata>=4.4 in /usr/local/lib/python3.7/di
Requirement already satisfied: typing-extensions>=3.6.4 in /usr/local/lib/python3.7/d
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: pyasn1<0.5.0,>=0.4.6 in /usr/local/lib/python3.7/dist-
Requirement already satisfied: charset-normalizer~=2.0.0 in /usr/local/lib/python3.7/
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.7/dist-packages
Requirement already satisfied: urllib3<1.27,>=1.21.1 in /usr/local/lib/python3.7/dist
Requirement already satisfied: oauthlib>=3.0.0 in /usr/local/lib/python3.7/dist-packa
Installing collected packages: numpy, tensorboard, scikit-learn, munkres, snorkel
  Attempting uninstall: numpy
    Found existing installation: numpy 1.21.6
    Uninstalling numpy-1.21.6:
      Successfully uninstalled numpy-1.21.6
  Attempting uninstall: tensorboard
    Found existing installation: tensorboard 2.8.0
    Uninstalling tensorboard-2.8.0:
      Successfully uninstalled tensorboard-2.8.0
  Attempting uninstall: scikit-learn
    Found existing installation: scikit-learn 1.0.2
    Uninstalling scikit-learn-1.0.2:
      Successfully uninstalled scikit-learn-1.0.2
ERROR: pip's dependency resolver does not currently take into account all the package
tensorflow 2.8.0 requires tf-estimator-nightly==2.8.0.dev2021122109, which is not ins
yellowbrick 1.4 requires scikit-learn>=1.0.0, but you have scikit-learn 0.24.2 which
tensorflow 2.8.0 requires numpy>=1.20, but you have numpy 1.19.5 which is incompatibl
tensorflow 2.8.0 requires tensorboard<2.9,>=2.8, but you have tensorboard 2.6.0 which
google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.27.1 which is i
datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompa
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 which
Successfully installed munkres-1.1.4 numpy-1.19.5 scikit-learn-0.24.2 snorkel-0.9.8
```

```
Successfully installed munkres-1.1.4 numpy-1.19.5 scikit-learn-0.24.2 snorkel-0.9.8 t
Requirement already satisfied: textblob in /usr/local/lib/python3.7/dist-packages (0.
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.7/dist-packages (f
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nl
Collecting transformers
  Downloading transformers-4.19.0-py3-none-any.whl (4.2 MB)
     |████████████████████████████████| 4.2 MB 5.2 MB/s
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.7/dist-packages (
Collecting tokenizers!=0.11.3,<0.13,>=0.11.1
  Downloading tokenizers-0.12.1-cp37-cp37m-manylinux_2_12_x86_64.manylinux2010_x86_64
     |████████████████████████████████| 6.6 MB 42.2 MB/s
Collecting pyyaml>=5.1
  Downloading PyYAML-6.0-cp37-cp37m-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_
     |████████████████████████████████| 596 kB 68.6 MB/s
Requirement already satisfied: importlib-metadata in /usr/local/lib/python3.7/dist-pa
Requirement already satisfied: filelock in /usr/local/lib/python3.7/dist-packages (fr
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.7/dist-packa
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.7/dist-pac
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.7/dist-packages
Collecting huggingface-hub<1.0,>=0.1.0
  Downloading huggingface_hub-0.6.0-py3-none-any.whl (84 kB)
```

Importing Libraries

```
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /usr/local/lib/python3.7/d
```

```
#Core
import pandas as pd
import numpy as np
import io
#Web Scraping library
from pygooglenews import GoogleNews
#HTML
from IPython.display import HTML, Image
import datetime
# Data Visualization
import seaborn as sns
sns.set_style("darkgrid")
import matplotlib.pyplot as plt
#Word Cloud
from wordcloud import WordCloud
from wordcloud import STOPWORDS
#Snorkel
from snorkel.labeling import LabelingFunction
import re
from snorkel.preprocess import preprocessor
from textblob import TextBlob
from snorkel.labeling import PandasLFApplier
from snorkel.labeling.model import LabelModel
from snorkel.labeling import LFAnalysis
from snorkel.labeling import filter_unlabeled_dataframe
from snorkel.labeling import labeling_function
#NLP packages
import spacy
```

```
from nltk.corpus import stopwords
import string
import nltk
import nltk.tokenize
punc = string.punctuation
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
#Supervised learning
from tqdm import tqdm_notebook as tqdm
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report
from sklearn import preprocessing
from imblearn.over_sampling import RandomOverSampler

# Torch ML libraries
import transformers
from transformers import BertModel, BertTokenizer, AdamW, get_linear_schedule_with_warmup
import torch
from torch import nn, optim
from torch.utils.data import Dataset, DataLoader

#metrics
from sklearn.metrics import accuracy_score, f1_score
from sklearn.metrics import classification_report, confusion_matrix

#set style for plots
sns.set_style("whitegrid")
sns.despine()
plt.style.use("seaborn-whitegrid")
plt.rc("figure", autolayout=True)
plt.rc("axes", labelweight="bold", labelsize="large", titleweight="bold", titlepad=10)


#transformers
from transformers import BertTokenizerFast
from transformers import TFBertModel
from transformers import RobertaTokenizerFast
from transformers import TFRobertaModel

#Naive Bayes
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB

##Deep learning libraries and APIs
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
```

```
from tensorflow.keras.preprocessing.sequence import pad_sequences
from unicodedata import digit

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

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
<Figure size 576x504 with 0 Axes>
```

Data Scraping from Web using pygoogle news

Scraping the 3 months news prior to the invasion

```
#Scraping the 3 months news prior to the war
gn = GoogleNews()

def get_news(search):
    stories = []
    start_date = datetime.date(2021,11,22)
    end_date = datetime.date(2022,2,22)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()

    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
                'published':item.published
            }
            stories.append(story)

    return stories
get_news('Russia-Ukraine')
# before_war = pd.DataFrame(get_news('Russia-Ukraine'))
```

```
[{'link': 'https://www.cnn.com/2021/11/22/politics/us-considering-weaponry-ukraine/in
  'published': 'Tue, 23 Nov 2021 08:00:00 GMT',
  'title': 'US considering sending extra weaponry to Ukraine as fears mount over pote
 {'link': 'https://abcnews.go.com/International/wireStory/russia-rejects-western-conc
  'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
  'title': 'Russia rejects Western concerns about Ukraine as smokescreen - ABC News'}
 {'link': 'https://nypost.com/2021/11/22/ukraine-holds-drills-after-general-warns-of-
```

```
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Ukraine holds new drills after top general warns of Russian attack in 202
{'link': 'https://www.express.co.uk/news/science/1522904/russia-ukraine-invasion-put
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Russia’s Ukraine invasion date predicted as Putin ‘knows Biden will back
{'link': 'https://www.rferl.org/a/bulgaria-washington-concern-ukraine-crimea-russia/
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': "U.S. 'Deeply Concerned' After Bulgarian President Refers To Crimea As 'Ru
{'link': 'https://www.ft.com/content/f7f90eb3-01ef-4cf3-9bae-9f94ea8133ce',
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'IMF releases funds for Ukraine as Russian troop build-up continues - Fina
{'link': 'https://www.politico.eu/article/ukraine-nato-georgia-europe-european-union
 'published': 'Tue, 23 Nov 2021 08:00:00 GMT',
 'title': 'Ukraine: NATO’s original sin - POLITICO Europe'},
{'link': 'https://www.nytimes.com/2021/11/23/opinion/russia-putin-west.html',
 'published': 'Tue, 23 Nov 2021 08:00:00 GMT',
 'title': 'Opinion | Russia’s Foreign Policy Isn’t All About Hurting the West - The
{'link': 'https://ukranews.com/en/news/816072-putin-and-biden-will-discuss-ukraine-a
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Putin And Biden Will Discuss Ukraine At Planned Negotiations - Russian Fo
{'link': 'https://www.thecipherbrief.com/column_article/talk-between-russia-and-the-
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'The Talk Between Russia and the US - The Cipher Brief'},
{'link': 'https://www.chess.com/news/view/ukraine-russia-win-2021-european-team-ches
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Ukraine, Russia Winners At 2021 European Team Chess Championships - Chess
{'link': 'https://www.defenseone.com/threats/2021/11/ukraine-wants-more-exercises-tr
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Ukraine Wants More Exercises, Training with US - Defense One'},
{'link': 'https://www.reuters.com/world/europe/italys-draghi-russias-putin-discuss-m
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': "Italy's Draghi, Russia's Putin discuss migrant crisis, Ukraine-statement
{'link': 'https://www.pressenza.com/2021/11/why-us-pressure-on-ukraine-and-taiwan-is
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Why US Pressure on Ukraine and Taiwan Is Driving China and Russia Closer
{'link': 'https://riponadvance.com/stories/mccaul-fitzpatrick-call-on-administration
 'published': 'Mon, 22 Nov 2021 18:59:33 GMT',
 'title': 'McCaul, Fitzpatrick call on administration to fully back Ukraine against
{'link': 'https://www.rferl.org/a/russia-ukraine-pipeline-sanctions/31574380.html',
 'published': 'Tue, 23 Nov 2021 08:00:00 GMT',
 'title': 'U.S. Imposes Sanctions On Russian-Linked Transadria Over Nord Stream 2 Pr
{'link': 'https://www.bloomberg.com/news/audio/2021-11-22/the-economic-impact-of-a-u
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'The Economic Impact Of A Ukraine-Russia Conflict (Radio) - Bloomberg'},
{'link': 'https://www.thedrive.com/the-war-zone/43239/ukrainian-troops-have-been-fir
 'published': 'Mon, 22 Nov 2021 08:00:00 GMT',
 'title': 'Ukrainian Troops Have Been Firing American-Made Javelin Missiles At Russi
{'link': 'https://euobserver.com/world/153595',
 'published': 'Tue, 23 Nov 2021 08:00:00 GMT',
 'title': "Russia threatens to cut off Moldova's gas again - EUobserver"}
```


Scraping the news from Feburary 24th, when the invasion started to till date

```
#Scraping the news from Feburary 24th, when the war started to till date
import datetime

gn = GoogleNews()

def get_war_news(search):
    stories = []
    start_date = datetime.date(2022,2,24)
    end_date = datetime.date(2022,5,11)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()

    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
                'published':item.published
            }
            stories.append(story)

    return stories
get_war_news('Russia-Ukraine')
# war = pd.DataFrame(get_war_news('Russia-Ukraine'))
```

```
[{'link': 'https://apnews.com/article/russia-ukraine-putin-attack-a05e7c4563ac94b9631
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia presses invasion to outskirts of Ukrainian capital - The Associate
 {'link': 'https://www.nbcnews.com/news/world/russia-launches-attacks-key-ukrainian-c
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia invades Ukraine on multiple fronts; U.S. and allies hit back with
 {'link': 'https://www.usnews.com/news/best-countries/articles/2022-02-24/explainer-w
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'EXPLAINER: Why Did Russia Invade Ukraine? - U.S. News & World Report'},
 {'link': 'https://www.nytimes.com/live/2022/02/24/world/russia-ukraine-putin',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Putin’s Forces Attack Ukraine - The New York Times'},
 {'link': 'https://www.newyorker.com/books/double-take/russias-war-on-ukraine-in-cont
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia’s War on Ukraine, in Context - The New Yorker'},
 {'link': 'https://techcrunch.com/2022/02/24/russia-ukraine/',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'How the tech industry is responding to Russia’s invasion of Ukraine - Tec
 {'link': 'https://news.un.org/en/story/2022/02/1112592',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "As Security Council meets on Ukraine crisis, Russia announces start of 's
 {'link': 'https://www.wilsoncenter.org/article/world-reaction-invasion-ukraine',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'World Reaction to the Invasion of Ukraine - Wilson Center'},
 {'link': 'https://www.theguardian.com/world/2022/feb/24/we-dont-want-this-russians-r
```

```
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Thousands join anti-war protests in Russia after Ukraine invasion - The G
{'link': 'https://reliefweb.int/report/ukraine/war-europe-responding-russia-s-invasi
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "War in Europe: Responding to Russia's Invasion of Ukraine - Ukraine - Rel
{'link': 'https://www.edweek.org/teaching-learning/how-to-talk-with-students-about-t
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'How to Talk With Students About the Russia-Ukraine War: 5 Tips - EdWeek'}
{'link': 'https://www.nytimes.com/2022/02/24/business/fox-news-russia-ukraine.html',
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Fox News Hosts Play Down Russia’s Attack on Ukraine - The New York Times'
{'link': 'https://www.republicworld.com/world-news/russia-ukraine-crisis/russia-ukra
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Russia-Ukraine war: Which country is on which side? US, Germany, China, I
{'link': 'https://www.nytimes.com/2022/02/25/opinion/ukraine-russia-kyiv.html',
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Opinion | Russia’s Ukraine Invasion Is Terrifying - The New York Times'},
{'link': 'https://apnews.com/article/russia-ukraine-business-asia-europe-united-nati
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'World leaders slap sanctions on the Kremlin over invasion - The Associate
{'link': 'https://theconversation.com/how-russia-ukraine-conflict-could-influence-af
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "How Russia-Ukraine conflict could influence Africa's food supplies - The
{'link': 'https://www.ft.com/content/77ab8dcf-cb02-4e57-aff0-85c8a84f5a1f',
 'published': 'Thu, 24 Feb 2022 11:25:53 GMT',
 'title': 'War in Ukraine: free to read - Financial Times'},
{'link': 'https://www.cnn.com/2022/02/23/investing/dow-futures-global-markets-russia
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Global stocks plunge as Russia attacks Ukraine - CNN'},
{'link': 'https://www.foxnews.com/politics/john-kerry-russia-ukraine-war-climate-cha
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
```

Country wise News

Here we are zooming into 4 countries like US, UK, Canada and India. We are gathering the news from these four countries and analysing their input on the Russia -Ukraine war and how the business relationships are impacted due to the war with Russia.

United States News, since the invasion took place till date

```
gn = GoogleNews(lang='en', country = 'US')

def get_US_news(search):
    stories = []
    start_date = datetime.date(2022,2,24)
    end_date = datetime.date(2022,5,13)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()
```

```
    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
                'published':item.published
            }
            stories.append(story)

    return stories
get_US_news('Russia-Ukraine-US relationship')
```

```
[{'link': 'https://www.cfr.org/in-brief/why-putins-war-ukraine-miscalculation',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Why Putin’s War With Ukraine Is a Miscalculation - Council on Foreign Rel
 {'link': 'https://www.usatoday.com/story/news/world/2022/02/24/vladimir-putin-russia
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia allies and neighboring nations remain quiet during Ukraine war - U
 {'link': 'https://www.aljazeera.com/news/2022/2/24/ukraine-breaks-diplomatic-ties-wi
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Ukraine cuts diplomatic ties with Russia after invasion - Al Jazeera Engl
 {'link': 'https://www.businessinsider.com/biden-invasion-of-ukraine-marks-complete-r
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "Biden: invasion of Ukraine marks 'complete rupture' in US-Russia relation
 {'link': 'https://www.latimes.com/politics/story/2022-02-24/beijing-may-be-tempted-t
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'China tempted to side with Russia in the Ukraine conflict - Los Angeles T
 {'link': 'https://thediplomat.com/2022/02/indias-neutrality-on-the-ukraine-conflict-
  'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
  'title': "India's 'Neutrality' on the Ukraine Conflict Could Hurt It in the Long Ru
 {'link': 'https://ecfr.eu/article/russias-escalation-in-ukraine-views-from-asia/',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia’s escalation in Ukraine: Views from Asia - European Council on For
 {'link': 'https://www.wilsoncenter.org/article/world-reaction-invasion-ukraine',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'World Reaction to the Invasion of Ukraine - Wilson Center'},
 {'link': 'https://newrepublic.com/article/165489/right-wing-media-putin-ukraine',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "Putin's Right Wing Allies Support His Invasion of Ukraine - The New Repub
 {'link': 'https://asia.nikkei.com/Politics/Ukraine-war/Just-26-of-Americans-say-U.S.
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Just 26% of Americans say U.S. should have major role in Ukraine - Nikkei
 {'link': 'https://abcnews.go.com/International/nato-factors-ukraine-russia-conflict/
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'How NATO factors into the Ukraine-Russia conflict - ABC News'},
 {'link': 'https://www.whitehouse.gov/briefing-room/statements-releases/2022/02/24/fa
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'FACT SHEET: Joined by Allies and Partners, the United States Imposes Deva
 {'link': 'https://www.politico.com/news/2022/02/24/ukraine-trump-impeachment-0001140
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "How the Ukraine invasion connects to Trump's first impeachment — and wher
 {'link': 'https://www.nytimes.com/live/2022/02/24/world/russia-ukraine-putin',
```

```
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Putin’s Forces Attack Ukraine - The New York Times'},
{'link': 'https://www.reuters.com/world/us/republicans-target-biden-blame-over-putin
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': "Republicans target Biden for blame over Putin's Ukraine invasion - Reuter
{'link': 'https://www.cnn.com/2022/02/24/politics/biden-russia-ukraine-poll/index.ht
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Joe Biden doesn't have a strong political hand to play on Russia - CNN"},
{'link': 'https://www.as-coa.org/articles/latin-american-leaders-react-russias-invas
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Latin American Leaders React to Russia's Invasion of Ukraine - AS/COA Onl
{'link': 'https://www.space.com/russia-ukraine-invasion-us-space-partnership-impacts
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'What does the Ukraine invasion mean for US-Russian partnership in space?
{'link': 'https://www.mytwintiers.com/news-cat/world-news/why-is-russia-invading-ukr
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Why is Russia invading Ukraine? History of this complicated relationship
```

United Kingdom News, since the invasion took place till date

```
#country = UK

gn = GoogleNews(lang='en', country = 'UK')

def get_UK_news(search):
    stories = []
    start_date = datetime.date(2022,2,24)
    end_date = datetime.date(2022,5,13)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()

    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
                'published':item.published
            }
            stories.append(story)

    return stories
get_UK_news('Russia-Ukraine-United Kigdom relationship')
```

```
[{'link': 'https://www.politico.eu/podcast/russia-and-britain-a-brief-history-from-na
  'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
  'title': 'Russia and Britain — A brief history, from Navarino to Ukraine - POLITICO
 {'link': 'https://www.telegraph.co.uk/politics/2022/02/24/boris-johnson-russia-ukrai
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
```

```
 'title': "PM unveils 'largest and most severe economic sanctions Russia has ever se
{'link': 'https://www.washingtonpost.com/world/2022/02/24/world-reaction-russia-ukra
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Global leaders from EU and NATO to China react to Russia's attack on Ukra
{'link': 'https://www.cnbc.com/2022/02/24/china-refuses-to-call-attack-on-ukraine-an
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'China refuses to call Russian attack on Ukraine an ‘invasion,’ deflects b
{'link': 'https://www.bbc.com/pidgin/world-60512048',
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Why is Russia attacking Ukraine?: Why Ukraine be so important for Putin R
{'link': 'https://www.politico.eu/article/ukraine-swift-rejection-eu-leaders-sanctio
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Ukraine gets SWIFT rejection from EU leaders - POLITICO Europe'},
{'link': 'https://www.forbes.com/sites/stevebanker/2022/02/24/the-russia-ukraine-war
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'The Russia Ukraine War Could Have Dire Impacts On Global Supply Chains -
{'link': 'https://www.freepressjournal.in/world/russia-ukraine-crisis-united-kingdom
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Russia-Ukraine Crisis: United Kingdom bans Russian airline Aeroflot from
{'link': 'https://www.whitehouse.gov/briefing-room/speeches-remarks/2022/02/24/remar
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Remarks by President Biden on Russia's Unprovoked and Unjustified Attack
{'link': 'https://www.atlanticcouncil.org/blogs/new-atlanticist/russian-hybrid-war-r
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Russian Hybrid War Report: Belarus joins conflict against Ukraine - Atlan
{'link': 'https://www.msnbc.com/opinion/msnbc-opinion/ukraine-s-u-n-ambassador-calls
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Ukraine's U.N. ambassador calls Russia's veto into question - MSNBC"},
{'link': 'https://theconversation.com/a-historian-corrects-misunderstandings-about-u
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'A historian corrects misunderstandings about Ukrainian and Russian histor
{'link': 'https://www.opendemocracy.net/en/dark-money-investigations/russia-controll
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "Exclusive: 623 new 'British' companies are actually controlled from Russi
{'link': 'https://www.whitehouse.gov/briefing-room/statements-releases/2022/02/24/fa
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'FACT SHEET: Joined by Allies and Partners, the United States Imposes Deva
{'link': 'https://theconversation.com/ukraine-invasion-what-the-west-needs-to-do-now
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Ukraine invasion: what the west needs to do now – expert view - The Conve
{'link': 'https://unric.org/en/statement-by-the-secretary-general-on-ukraine/',
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': 'Statement by the Secretary-General – on Ukraine - United Nations Western
{'link': 'https://www.express.co.uk/sport/football/1571507/Man-Utd-news-Boris-Johnso
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Man Utd sponsor banned from UK as part of Russia sanctions in huge £8m-a-
{'link': 'https://www.timesofisrael.com/why-israel-is-having-trouble-picking-sides-i
 'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
 'title': 'Why Israel is having trouble picking sides in Russia’s war on Ukraine - T
{'link': 'https://www.thelocal.dk/20220224/explained-how-denmark-could-be-impacted-b
 'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
 'title': "EXPLAINED: How Denmark could be impacted by Russia's invasion of Ukraine
```

Canada News, since the invasion took place till date

```
#Country: Canada
gn = GoogleNews(lang='en', country = 'Canada')

def get_CA_news(search):
    stories = []
    start_date = datetime.date(2022,2,24)
    end_date = datetime.date(2022,5,13)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()

    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
                'published':item.published
            }
            stories.append(story)

    return stories
get_CA_news('Russia-Ukraine-canada-relationship')
```

```
[{'link': 'https://pm.gc.ca/en/news/news-releases/2022/02/24/canada-announces-additio
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Canada announces additional measures to support Ukraine - Prime Minister
 {'link': 'https://www.thestar.com/news/world/2022/02/24/russia-ukraine-updates-canad
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Latest on Russia-Ukraine: Russian forces advance to outskirts of Ukrainia
 {'link': 'https://www.wilsoncenter.org/article/world-reaction-invasion-ukraine',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'World Reaction to the Invasion of Ukraine - Wilson Center'},
 {'link': 'https://www.whitehouse.gov/briefing-room/statements-releases/2022/02/24/fa
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'FACT SHEET: Joined by Allies and Partners, the United States Imposes Deva
 {'link': 'https://www.aljazeera.com/news/2022/2/24/ukraine-breaks-diplomatic-ties-wi
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Ukraine cuts diplomatic ties with Russia after invasion - Al Jazeera Engl
 {'link': 'https://www.cbc.ca/news/canada/saskatchewan/ukraine-russia-attack-sask-rea
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Sask. residents fear for loved ones in Ukraine after Russian invasion - C
 {'link': 'https://www.nytimes.com/live/2022/02/24/world/russia-ukraine-putin',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Putin’s Forces Attack Ukraine - The New York Times'},
 {'link': 'https://www.elpasotimes.com/story/news/2022/02/24/us-mexico-border-summit-
  'published': 'Thu, 24 Feb 2022 23:01:56 GMT',
  'title': 'Russian invasion of Ukraine looms large at US-Mexico Border Summit - El P
 {'link': 'https://www.space.com/russia-ukraine-invasion-us-space-partnership-impacts
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'What does the Ukraine invasion mean for US-Russian partnership in space?
```

```
 {'link': 'https://www.aljazeera.com/news/2022/2/24/rooted-in-nato-inside-irans-respo
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': '‘Rooted in NATO’: Iran responds to Russia’s Ukraine attack - Al Jazeera E
 {'link': 'https://globalnews.ca/news/8642615/quebec-legault-ukraine-russia-invasion/
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Quebec Premier François Legault condemns ‘Russian aggression’ in Ukraine
 {'link': 'https://newrepublic.com/article/165489/right-wing-media-putin-ukraine',
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "Putin's Right Wing Allies Support His Invasion of Ukraine - The New Repub
 {'link': 'https://www.ctvnews.ca/world/world-leaders-slap-sanctions-on-the-kremlin-o
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russia attacks Ukraine: World reacts in outrage - CTV News'},
 {'link': 'https://www.justsecurity.org/80343/russias-new-assault-on-ukraine-is-not-e
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "Russia's Invasion of Ukraine Is Essentially Not About NATO - Just Securit
 {'link': 'https://theconversation.com/a-historian-corrects-misunderstandings-about-u
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'A historian corrects misunderstandings about Ukrainian and Russian histor
 {'link': 'https://www.atlanticcouncil.org/blogs/new-atlanticist/russian-hybrid-war-r
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Russian Hybrid War Report: Belarus joins conflict against Ukraine - Atlan
 {'link': 'https://www.aljazeera.com/economy/2022/2/25/ukraine-russia-crisis-will-chi
  'published': 'Fri, 25 Feb 2022 08:00:00 GMT',
  'title': 'Ukraine-Russia crisis: Will China be Putin’s economic lifeline? - Al Jaze
 {'link': 'https://www.aljazeera.com/news/2022/2/24/ukraine-asked-turkey-to-close-bla
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': 'Ukraine asks Turkey to shut Black Sea waterways to Russian ships - Al Jaz
 {'link': 'https://theconversation.com/how-russias-invasion-of-ukraine-will-ripple-th
  'published': 'Thu, 24 Feb 2022 08:00:00 GMT',
  'title': "How Russia's invasion of Ukraine will ripple through the global economy a
```

India News, since the invasion took place till date

```
#country = India
gn = GoogleNews(lang='en', country = 'India')

def get_Ind_news(search):
    stories = []
    start_date = datetime.date(2022,2,24)
    end_date = datetime.date(2022,5,13)
    delta = datetime.timedelta(days=1)
    date_list = pd.date_range(start_date, end_date).tolist()

    for date in date_list[:-1]:
        result = gn.search(search, from_=date.strftime('%Y-%m-%d'), to_=(date+delta).strftime
        newsitem = result['entries']

        for item in newsitem:
            story = {
                'title':item.title,
                'link':item.link,
```

```
                'published':item.published
            }
            stories.append(story)

    return stories
get_Ind_news('Russia-Ukrain-india-relationship')
```

Creating the dataframes of the results obtained from the Web Scraping from all over the world and country wise individually

```
#dataset for before war
before_war = pd.DataFrame(get_news('Russia-Ukraine'))
before_war
```

	title	link	published
0	US considering sending extra weaponry to Ukrai...	https://www.cnn.com/2021/11/22/politics/us-con...	Tue, 23 Nov 2021 08:00:00 GMT
1	Russia rejects Western concerns about Ukraine ...	https://abcnews.go.com/International/wireStory...	Mon, 22 Nov 2021 08:00:00 GMT
2	Ukraine holds new drills after top general war...	https://nypost.com/2021/11/22/ukraine-holds-dr...	Mon, 22 Nov 2021 08:00:00 GMT
3	Russia’s Ukraine invasion date predicted as Pu...	https://www.express.co.uk/news/science/1522904...	Mon, 22 Nov 2021 08:00:00 GMT
4	U.S. 'Deeply Concerned' After	https://www.rferl.org/a/bulgaria-washington-co...	Mon, 22 Nov 2021 08:00:00

```
#During War
war = pd.DataFrame(get_war_news('Russia-Ukraine'))
war
```

	title	link	published
0	Russia presses invasion to outskirts of Ukrain...	https://apnews.com/article/russia-ukraine-puti...	Thu, 24 Feb 2022 08:00:00 GMT
1	Russia invades Ukraine on multiple fronts; U.S...	https://www.nbcnews.com/news/world/russia-laun...	Thu, 24 Feb 2022 08:00:00 GMT
2	EXPLAINER: Why Did Russia Invade Ukraine? - U....	https://www.usnews.com/news/best-countries/art...	Thu, 24 Feb 2022 08:00:00 GMT

```
#countrywise creation of dfs during the war
df_US = pd.DataFrame(get_US_news('Russia-Ukraine-US relationship'))
df_UK = pd.DataFrame(get_UK_news('Russia-Ukraine-United Kingdom relationship'))
df_CA = pd.DataFrame(get_CA_news('Russia-Ukraine-Canada-relationship'))
df_IN = pd.DataFrame(get_Ind_news('Russia-Ukrain-India-relationship'))
```

.	Russia’s War on Ukraine.	https://www.newyorker.com/books/double-	Feb 2022

Combining the country wise dataframes into one and then will analyse the sentiments of all 4 countries together

```
# merging the all four countries dataframe into one for further analysis
combined_df =df_US.append(df_UK,ignore_index=True).append(df_CA,ignore_index=True).append(df_
combined_df
```

	title	link	published

Data Cleaning and Wrangling

```
# We will be using df1 as our dataset of before war
df1 = before_war.copy()
#We will be using df2 as our dataset of during the war
df2 = war.copy()
#We will be using df3 as our dataset of combined 4 countries which we have taken above
df3 = combined_df.copy()
```

```
df1.isnull().sum()
```

```
title        0
link         0
published    0
dtype: int64
```

```
df2.isnull().sum()
```

```
title        0
link         0
published    0
dtype: int64
```

```
df3.isnull().sum()
```

```
title        0
link         0
published    0
dtype: int64
```

As we can see that there are no null values in our datasets. Hence the datasets are almost clean.

Checking Duplicates

```
df1.drop_duplicates()
```

	title	link	published
0	US considering sending extra weaponry to Ukrai...	https://www.cnn.com/2021/11/22/politics/us-con...	Tue, 23 Nov 2021 08:00:00 GMT
1	Russia rejects Western concerns about Ukraine ...	https://abcnews.go.com/International/wireStory...	Mon, 22 Nov 2021 08:00:00 GMT
2	Ukraine holds new drills after top general war...	https://nypost.com/2021/11/22/ukraine-holds-dr...	Mon, 22 Nov 2021 08:00:00 GMT
3	Russia's Ukraine invasion date predicted as Pu...	https://www.express.co.uk/news/science/1522904...	Mon, 22 Nov 2021 08:00:00 GMT

```
df2.drop_duplicates()
```

	title	link	published
0	Russia presses invasion to outskirts of Ukrain...	https://apnews.com/article/russia-ukraine-puti...	Thu, 24 Feb 2022 08:00:00 GMT
1	Russia invades Ukraine on multiple fronts; U.S...	https://www.nbcnews.com/news/world/russia-laun...	Thu, 24 Feb 2022 08:00:00 GMT
2	EXPLAINER: Why Did Russia Invade Ukraine? - U....	https://www.usnews.com/news/best-countries/art...	Thu, 24 Feb 2022 08:00:00 GMT
3	Putin's Forces Attack Ukraine - The New York T...	https://www.nytimes.com/live/2022/02/24/world/...	Thu, 24 Feb 2022 08:00:00 GMT
4	Russia's War on Ukraine, in Context - The New ...	https://www.newyorker.com/books/double-take/ru...	Thu, 24 Feb 2022 08:00:00

```
df3.drop_duplicates()
```

	title	link	published
0	Why Putin's War With Ukraine Is a Miscalculati...	https://www.cfr.org/in-brief/why-putins-war-uk...	Thu, 24 Feb 2022 08:00:00 GMT
1	Russia allies and neighboring nations remain q...	https://www.usatoday.com/story/news/world/2022...	Thu, 24 Feb 2022 08:00:00 GMT
2	Ukraine cuts diplomatic ties with Russia after...	https://www.aljazeera.com/news/2022/2/24/ukrai...	Thu, 24 Feb 2022 08:00:00 GMT
3	Biden: invasion of Ukraine marks	https://www.businessinsider.com/biden-invasion...	Thu, 24 Feb 2022
	China tempted to side		

Extracting only date from published column and renaming it as date.

```
# extracting dates and dropping published.
df1['published'] = pd.to_datetime(df1.published)
df1['date'] = df1['published'].dt.date
df1.drop(columns = 'published',inplace = True)

df2['published'] = pd.to_datetime(df2.published)
df2['date'] = df2['published'].dt.date
df2.drop(columns = 'published',inplace = True)

df3['published'] = pd.to_datetime(df3.published)
df3['date'] = df3['published'].dt.date
df3.drop(columns = 'published',inplace = True)
```

We have dropped the duplicate values from all the datasets. Now, we are good to use our unique data for further analysis.

Showing the General Information about the available data.Table 1,2 and 3 summarizes the information.

```
# General Info
display(HTML('<span style="font-weight:bold">' + 'Table 1 - General Dataset Information - Bef
            + '</span>'),df1.head(3))
print(f"Number of Samples: {df1.shape[0]}")
```

```
print(f"Number of Features: {df1.shape[1]}")
# Missing Values
aux = df1.isnull().sum()/df1.shape[0] * 100.00
if sum(aux) == 0:
    print("No Missing Data!")
else:
    aux = aux[aux > 0]
    aux = pd.DataFrame({'Feature': aux.index,
                        'Percent_Missing': aux.values})
    aux.sort_values('Percent_Missing', inplace=True)
    display(aux)
```

Table 1 - General Dataset Information - Before War

	title	link	date
0	US considering sending extra weaponry to Ukrai...	https://www.cnn.com/2021/11/22/politics/us-con...	2021-11-23
1	Russia rejects Western concerns about Ukraine ...	https://abcnews.go.com/International/wireStory...	2021-11-22
2	Ukraine holds new drills after top general war...	https://nypost.com/2021/11/22/ukraine-holds-dr...	2021-11-22

```
# General Info
display(HTML('<span style="font-weight:bold">' + 'Table 2 - General Dataset Information - Dur
             + '</span>'),df2.head(3))
print(f"Number of Samples: {df2.shape[0]}")
print(f"Number of Features: {df2.shape[1]}")
# Missing Values
aux = df2.isnull().sum()/df2.shape[0] * 100.00
if sum(aux) == 0:
    print("No Missing Data!")
else:
    aux = aux[aux > 0]
    aux = pd.DataFrame({'Feature': aux.index,
                        'Percent_Missing': aux.values})
    aux.sort_values('Percent_Missing', inplace=True)
    display(aux)
```

Table 2 - General Dataset Information - During War

	title	link	date
0	Russia presses invasion to outskirts of Ukrain...	https://apnews.com/article/russia-ukraine-puti...	2022-02-24
1	Russia invades Ukraine on multiple fronts; U.S...	https://www.nbcnews.com/news/world/russia-laun...	2022-02-24
2	EXPLAINER: Why Did Russia Invade Ukraine? - U....	https://www.usnews.com/news/best-countries/art...	2022-02-24

```
Number of Samples: 5963
```

```
# General Info
display(HTML('<span style="font-weight:bold">' + 'Table 3 - General Dataset Information - Com
            + '</span>'),df3.head(3))
print(f"Number of Samples: {df3.shape[0]}")
print(f"Number of Features: {df3.shape[1]}")
# Missing Values
aux = df3.isnull().sum()/df3.shape[0] * 100.00
if sum(aux) == 0:
    print("No Missing Data!")
else:
    aux = aux[aux > 0]
    aux = pd.DataFrame({'Feature': aux.index,
                        'Percent_Missing': aux.values})
    aux.sort_values('Percent_Missing', inplace=True)
    display(aux)
```

Table 3 - General Dataset Information - Combined 4 countries News

	title	link	date
0	Why Putin’s War With Ukraine Is a Miscalculati...	https://www.cfr.org/in-brief/why-putins-war-uk...	2022-02-24
1	Russia allies and neighboring nations remain q...	https://www.usatoday.com/story/news/world/2022...	2022-02-24
2	Ukraine cuts diplomatic ties with Russia after...	https://www.aljazeera.com/news/2022/2/24/ukrai...	2022-02-24

```
Number of Samples: 12598
```

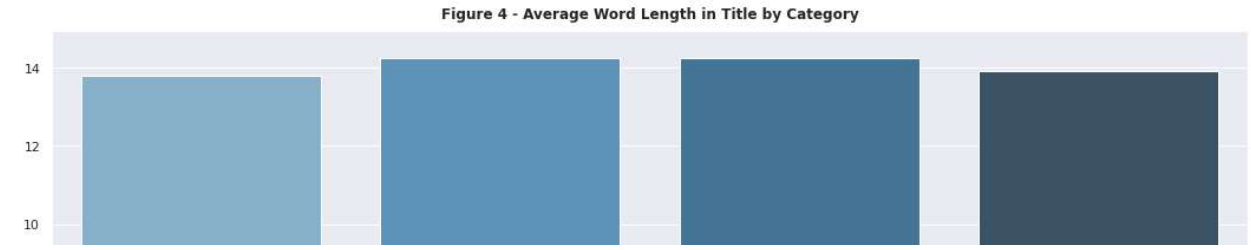
Basic Data Analysis

Average Word Length in Title

```
df_ed2 = df2.copy()
df_ed2['year'] = pd.DatetimeIndex(df_ed2['date']).year


df_ed2['month'] = pd.DatetimeIndex(df_ed2['date']).month


df_ed2.head()
```

	title	link	date	year	month
0	Russia presses invasion to outskirts	https://apnews.com/article/russia-ukraine-puti...	2022-03-24	2022	2

```
categories = df_ed2["month"].unique()
aux = []
j = 0
for i in categories:
    df_aux = df_ed2[df_ed2["month"] == i]
    aux.append(np.mean(df_aux['title'].apply(lambda x : len(x.split()))))
    j = j+1

df_aux = pd.DataFrame({
    "Months" : categories,
    "Average": aux
}
)

df_aux = df_aux.sort_values(['Average'], ascending=False).reset_index(drop=True)

sns.set(rc={'figure.figsize':(15,8)})

ax = sns.barplot(x="Months", y="Average", data=df_aux, palette="Blues_d")
plt.title("Figure 4 - Average Word Length in Title by Category",
          fontweight="bold", size=12)
plt.xticks(rotation=80);
```



Word Cloud

```
stopwords1 = set(STOPWORDS)

wordcloud = WordCloud(background_color='white',
                      max_words=100,
                      width=500,
                      height=500
)

wordcloud.generate(str(df2['title']))
plt.rcParams['figure.figsize'] = (8,7)
plt.axis('off')
plt.suptitle('Figure 7 - Russia-Ukraine War Word Cloud (2022)', fontsize=16, fontweight='bold
plt.imshow(wordcloud, interpolation='bilinear')
plt.show()
```

Figure 7 - Russia-Ukraine War Word Cloud (2022).

Sentiment Analysis

Text Preprocessing for Sentiment Analysis

```
#defining constants to represent the class labels :positive, negative, and abstain
POSITIVE = 1
NEGATIVE = 0
ABSTAIN = -1
#define function which looks into the input words to represent a proper label
def keyword_lookup(x, keywords, label):
    if any(word in x.text.lower() for word in keywords):
        return label
    return ABSTAIN
#define function which assigns a correct label
def make_keyword_lf(keywords, label=POSITIVE):
    return LabelingFunction(
        name=f"keyword_{keywords[0]}",
        f=keyword_lookup,
        resources=dict(keywords=keywords, label=label))

#these two lists can be further extended
"""positive news might contain the following words' """
keyword_positive = make_keyword_lf(keywords=['boosts', 'great', 'develops', 'promising', 'amb
                                             'peace', 'party', 'hope', 'flourish', 'respect',
                                             'perfect', 'complete', 'assured' ])
"""negative news might contain the following words"""
keyword_negative = make_keyword_lf(keywords=['war','solidiers', 'turmoil', 'injur','trouble',
                                             'defeat', 'damage', 'dishonest', 'dead', 'fear',
                                             'fraud', 'dispute', 'destruction', 'battle', 'un
                                             'unhealthy', 'tensions','emergency', 'Accident',
                                             'weaponizing', 'crisis', 'warships', 'pessimisti
                                             'complicate', 'separatists'], label=NEGATIVE)
```

Another set of labeling functions was created using the TextBlob tool, which is a sentiment analyzer that has been pre-trained. We'll make a Pre-processor that runs TextBlob on our headlines before extracting the polarity and subjectivity ratings.

```
#set up a preprocessor function to determine polarity & subjectivity using textlob pretrained
@preprocessor(memoize=True)
def textblob_sentiment(x):
    scores = TextBlob(x.text)
    x.polarity = scores.sentiment.polarity
```

```
    x.subjectivity = scores.sentiment.subjectivity
    return x
#find polarity
@labeling_function(pre=[textblob_sentiment])
def textblob_polarity(x):
    return POSITIVE if x.polarity > 0.6 else ABSTAIN
#find subjectivity
@labeling_function(pre=[textblob_sentiment])
def textblob_subjectivity(x):
    return POSITIVE if x.subjectivity >= 0.5 else ABSTAIN
```

After that, we'll integrate all of the labeling functions and apply them to our dataset. The label model was then used to predict and generate positive and negative classes.

Before Invasion - label_model to predict and generate the positive and negative classes

```
#conduct some data cleaning
df11 = df1.drop(['link', 'date'], axis=1)
df11 = df11.rename(columns = {'title': 'text'})
df11['text'] = df11['text'].astype(str)
df11.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1164 entries, 0 to 1163
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    1164 non-null   object
dtypes: object(1)
memory usage: 9.2+ KB
```

```
#combine all the labeling functions
lfs = [keyword_positive, keyword_negative, textblob_polarity, textblob_subjectivity ]
#apply the lfs on the dataframe
applier = PandasLFApplier(lfs=lfs)
L_snorkel = applier.apply(df=df11)
#apply the label model
label_model = LabelModel(cardinality=2, verbose=True)
#fit on the data
label_model.fit(L_snorkel)
#predict and create the labels
df11["label"] = label_model.predict(L=L_snorkel)
```

```
100%|██████████| 1164/1164 [00:02<00:00, 489.51it/s]
INFO:root:Computing O...
INFO:root:Estimating \mu...
```

```
  0%|          | 0/100 [00:00<?, ?epoch/s]INFO:root:[0 epochs]: TRAIN:[loss=0.187]
INFO:root:[10 epochs]: TRAIN:[loss=0.069]
INFO:root:[20 epochs]: TRAIN:[loss=0.002]
INFO:root:[30 epochs]: TRAIN:[loss=0.006]
INFO:root:[40 epochs]: TRAIN:[loss=0.002]
INFO:root:[50 epochs]: TRAIN:[loss=0.001]
INFO:root:[60 epochs]: TRAIN:[loss=0.000]
 67%|██████▋    | 67/100 [00:00<00:00, 664.81epoch/s]INFO:root:[70 epochs]: TRAIN:[loss=
INFO:root:[80 epochs]: TRAIN:[loss=0.000]
INFO:root:[90 epochs]: TRAIN:[loss=0.000]
100%|██████████| 100/100 [00:00<00:00, 661.44epoch/s]
INFO:root:Finished Training
```

```
#Filtering out unlabeled data points
dfl1= dfl1.loc[dfl1.label.isin([0,1,-1]), :]
#find the label counts
dfl1['label'].value_counts()
```

```
 0    675
-1    389
 1    100
Name: label, dtype: int64
```

```
#Visualization
# Let's have a look at the class balance.
class_names = ['Neutral', 'Negative', 'Positive']
ax = sns.countplot(dfl1.label)
plt.title('Before War')
plt.xlabel('Review sentiment')
ax.set_xticklabels(class_names)
```

```
[Text(0, 0, 'Neutral'), Text(0, 0, 'Negative'), Text(0, 0, 'Positive')]
```

The graph clearly shows that before war the news was negative and neutral and hardly any positive.

During Invasion - label_model to predict and generate the positive and negative classes

```
#conduct some data cleaning
df12 = df2.drop(['link', 'date'], axis=1)
df12 = df12.rename(columns = {'title': 'text'})
df12['text'] = df12['text'].astype(str)
df12.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5963 entries, 0 to 5962
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    5963 non-null   object
dtypes: object(1)
memory usage: 46.7+ KB
```

```
#combine all the labeling functions
lfs = [keyword_positive, keyword_negative, textblob_polarity, textblob_subjectivity ]
#apply the lfs on the dataframe
applier = PandasLFApplier(lfs=lfs)
L_snorkel = applier.apply(df=df12)
#apply the label model
label_model = LabelModel(cardinality=2, verbose=True)
#fit on the data
label_model.fit(L_snorkel)
#predict and create the labels
df12["label"] = label_model.predict(L=L_snorkel)
```

```
100%|██████████| 5963/5963 [00:10<00:00, 556.30it/s]
INFO:root:Computing O...
INFO:root:Estimating \mu...
  0%|          | 0/100 [00:00<?, ?epoch/s]INFO:root:[0 epochs]: TRAIN:[loss=0.198]
INFO:root:[10 epochs]: TRAIN:[loss=0.072]
INFO:root:[20 epochs]: TRAIN:[loss=0.002]
```

```
INFO:root:[30 epochs]: TRAIN:[loss=0.006]
INFO:root:[40 epochs]: TRAIN:[loss=0.003]
INFO:root:[50 epochs]: TRAIN:[loss=0.001]
INFO:root:[60 epochs]: TRAIN:[loss=0.000]
 61%|██████    | 61/100 [00:00<00:00, 595.54epoch/s]INFO:root:[70 epochs]: TRAIN:[loss=0
INFO:root:[80 epochs]: TRAIN:[loss=0.000]
INFO:root:[90 epochs]: TRAIN:[loss=0.000]
100%|██████████| 100/100 [00:00<00:00, 589.97epoch/s]
INFO:root:Finished Training
```

```
#Filtering out unlabeled data points
dfl2= dfl2.loc[dfl2.label.isin([0,1,-1]), :]
#find the label counts
dfl2['label'].value_counts()
```

```
 0    3669
-1    1696
 1     598
Name: label, dtype: int64
```

```
#Visualization
# Let's have a look at the class balance.
class_names = ['Neutral', 'Negative', 'Positive']
ax = sns.countplot(dfl2.label)
plt.title('During War')
plt.xlabel('Review sentiment')
ax.set_xticklabels(class_names)
```

```
[Text(0, 0, 'Neutral'), Text(0, 0, 'Negative'), Text(0, 0, 'Positive')]
```

We can see that the number of negative words in the dataset during the war is significantly higher than the number of positive or neutral words.

UK, US, Canada, India- label_model to predict and generate the positive and negative classes

```
#conduct some data cleaning
df13 = df3.drop(['link', 'date'], axis=1)
df13 = df13.rename(columns = {'title': 'text'})
df13['text'] = df13['text'].astype(str)
df13.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 12598 entries, 0 to 12597
Data columns (total 1 columns):
 #   Column  Non-Null Count  Dtype 
---  ------  --------------  ----- 
 0   text    12598 non-null  object
dtypes: object(1)
memory usage: 98.5+ KB
```

```
# combine all the labeling functions
lfs = [keyword_positive, keyword_negative, textblob_polarity, textblob_subjectivity ]
#apply the lfs on the dataframe
applier = PandasLFApplier(lfs=lfs)
L_snorkel = applier.apply(df=df13)
#apply the label model
label_model = LabelModel(cardinality=2, verbose=True)
#fit on the data
label_model.fit(L_snorkel)
#predict and create the labels
df13["label"] = label_model.predict(L=L_snorkel)
```

```
100%|██████████| 12598/12598 [00:18<00:00, 682.63it/s]
INFO:root:Computing O...
INFO:root:Estimating \mu...
  0%|          | 0/100 [00:00<?, ?epoch/s]INFO:root:[0 epochs]: TRAIN:[loss=0.079]
INFO:root:[10 epochs]: TRAIN:[loss=0.030]
INFO:root:[20 epochs]: TRAIN:[loss=0.001]
INFO:root:[30 epochs]: TRAIN:[loss=0.002]
INFO:root:[40 epochs]: TRAIN:[loss=0.001]
INFO:root:[50 epochs]: TRAIN:[loss=0.000]
```

```
INFO:root:[60 epochs]: TRAIN:[loss=0.000]
INFO:root:[70 epochs]: TRAIN:[loss=0.000]
 75%|███████▌  | 75/100 [00:00<00:00, 735.13epoch/s]INFO:root:[80 epochs]: TRAIN:[loss=0
INFO:root:[90 epochs]: TRAIN:[loss=0.000]
100%|██████████| 100/100 [00:00<00:00, 739.21epoch/s]
INFO:root:Finished Training
```

```
#Filtering out unlabeled data points
dfl3= dfl3.loc[dfl3.label.isin([0,1,-1]), :]
#find the label counts
dfl3['label'].value_counts()
```

```
-1    6280
 0    4582
 1    1736
Name: label, dtype: int64
```

```
# Let's have a look at the class balance.
class_names = ['Neutral', 'Negative', 'Positive']
ax = sns.countplot(dfl3.label)
plt.xlabel('Review sentiment')
plt.title('US, UK, Canada and India Sentiments')
ax.set_xticklabels(class_names)
```

```
[Text(0, 0, 'Neutral'), Text(0, 0, 'Negative'), Text(0, 0, 'Positive')]
```

Through the graph we can see that the countries like US, UK, India and Canada are neutral about the war

Text Preprocessing

Before Invasion- Dataframe

```
#make a copy of the dataframe
data1 = dfl1.copy()
#define a function which handles the text preprocessing
def preparation_text_data(data1):
    """
    This pipeline prepares the text data, conducting the following steps:
    1) Tokenization
    2) Lemmatization
    4) Removal of stopwords
    5) Removal of punctuation
    """
    # initialize spacy object
    nlp = spacy.load('en_core_web_sm')
    # select raw text
    raw_text = data1.text.values.tolist()
    # tokenize
    tokenized_text = [[nlp(i.lower().strip())] for i in tqdm(raw_text)]
    #define the punctuations and stop words
    punc = string.punctuation
    stop_words = set(stopwords.words('english'))
    #lemmatize, remove stopwords and punctuationd
    corpus = []
    for doc in tqdm(tokenized_text):
        corpus.append([word.lemma_ for word in doc[0] if (word.lemma_ not in stop_words and w
    # add prepared data to df
    data1["text"] = corpus
    return data1
#apply the data preprocessing function
data1 =  preparation_text_data(data1)
```

```
100%                                                1164/1164 [00:15<00:00, 80.00it/s]
100%                                                1164/1164 [00:00<00:00, 13184.93it/s]
```

During Invasion - Dataframe

```
#make a copy of the dataframe
data2 = df12.copy()
#define a function which handles the text preprocessing
def preparation_text_data(data2):
    """
    This pipeline prepares the text data, conducting the following steps:
    1) Tokenization
    2) Lemmatization
    4) Removal of stopwords
    5) Removal of punctuation
    """
    # initialize spacy object
    nlp = spacy.load('en_core_web_sm')
    # select raw text
    raw_text = data2.text.values.tolist()
    # tokenize
    tokenized_text = [[nlp(i.lower().strip())] for i in tqdm(raw_text)]
    #define the punctuations and stop words
    punc = string.punctuation
    stop_words = set(stopwords.words('english'))
    #lemmatize, remove stopwords and punctuationd
    corpus = []
    for doc in tqdm(tokenized_text):
        corpus.append([word.lemma_ for word in doc[0] if (word.lemma_ not in stop_words and w
    # add prepared data to df
    data2["text"] = corpus
    return data2
#apply the data preprocessing function
data2 =  preparation_text_data(data2)
```

```
100%                                                  5963/5963 [01:17<00:00, 79.56it/s]
100%                                                  5963/5963 [00:00<00:00, 30167.86it/s]
```

US, UK, Canada and India - Dataframe

```
#make a copy of the dataframe
data3 = df13.copy()
#define a function which handles the text preprocessing
def preparation_text_data(data3):
    """
    This pipeline prepares the text data, conducting the following steps:
    1) Tokenization
    2) Lemmatization
    4) Removal of stopwords
    5) Removal of punctuation
    """
```

```
    # initialize spacy object
    nlp = spacy.load('en_core_web_sm')
    # select raw text
    raw_text = data3.text.values.tolist()
    # tokenize
    tokenized_text = [[nlp(i.lower().strip())] for i in tqdm(raw_text)]
    #define the punctuations and stop words
    punc = string.punctuation
    stop_words = set(stopwords.words('english'))
    #lemmatize, remove stopwords and punctuationd
    corpus = []
    for doc in tqdm(tokenized_text):
        corpus.append([word.lemma_ for word in doc[0] if (word.lemma_ not in stop_words and w
    # add prepared data to df
    data3["text"] = corpus
    return data3
#apply the data preprocessing function
data3 =  preparation_text_data(data3)
```

```
100%                                                  12598/12598 [03:46<00:00, 62.17it/s]

100%                                                  12598/12598 [00:00<00:00, 21441.83it/s]
```

Text Representation

```
def text_representation(data1):
  tfidf_vect = TfidfVectorizer()
  data1['text'] = data1['text'].apply(lambda text: " ".join(set(text)))
  X_tfidf = tfidf_vect.fit_transform(data1['text'])
  print(X_tfidf.shape)
  print(tfidf_vect.get_feature_names())
  X_tfidf = pd.DataFrame(X_tfidf.toarray())
  return X_tfidf
#apply the TFIDV function
X_tfidf = text_representation(data1)
```

```
(1164, 2538)
['000', '10', '100', '10000', '11', '11th', '12', '120', '13', '130000', '14', '15', '15
```

```
X_tfidf2 = text_representation(data2)
```

```
(5963, 6545)
['000', '037', '04', '06', '07', '10', '100', '100000', '100th', '101', '104', '108', '1
```

```
X_tfidf3 = text_representation(data3)
```

```
(12598, 11130)
['00', '000', '02', '03', '04', '05', '06', '07', '08', '10', '100', '1000', '100000',
```

Model Training through Pipeline

Here we will be using 6 different models in the pipeline to test the accuracy of each dataset.

```
#Creating a list of Pipeline with well-known ML models
from sklearn.pipeline import make_pipeline
from sklearn.naive_bayes import MultinomialNB,ComplementNB
from sklearn.linear_model import LogisticRegression, RidgeClassifier
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.tree import DecisionTreeClassifier

pipelines=[]
for model in [DecisionTreeClassifier(), AdaBoostClassifier(), ComplementNB(),
              LogisticRegression(solver='saga'), RidgeClassifier(solver='auto'), SVC(),Random
    pipeline=make_pipeline(TfidfVectorizer(lowercase=True), model)
    pipelines.append(pipeline)


X= data1.text
y=data1.label
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)


#Training the model
import time
training_time=[]

for pipeline in pipelines:
    start=time.time()
    pipeline.fit(X_train, y_train)
    stop=time.time()
    training_time.append(stop-start)



#Prediction from test dataset
from sklearn.metrics import classification_report, confusion_matrix, f1_score, precision_scor
model_name=[]
precision_array=[]
recall_array=[]
f1_array=[]
test_time=[]
print("Classifiation Report\n")
print("***************************************************")
for i, pipeline in enumerate(pipelines):
```

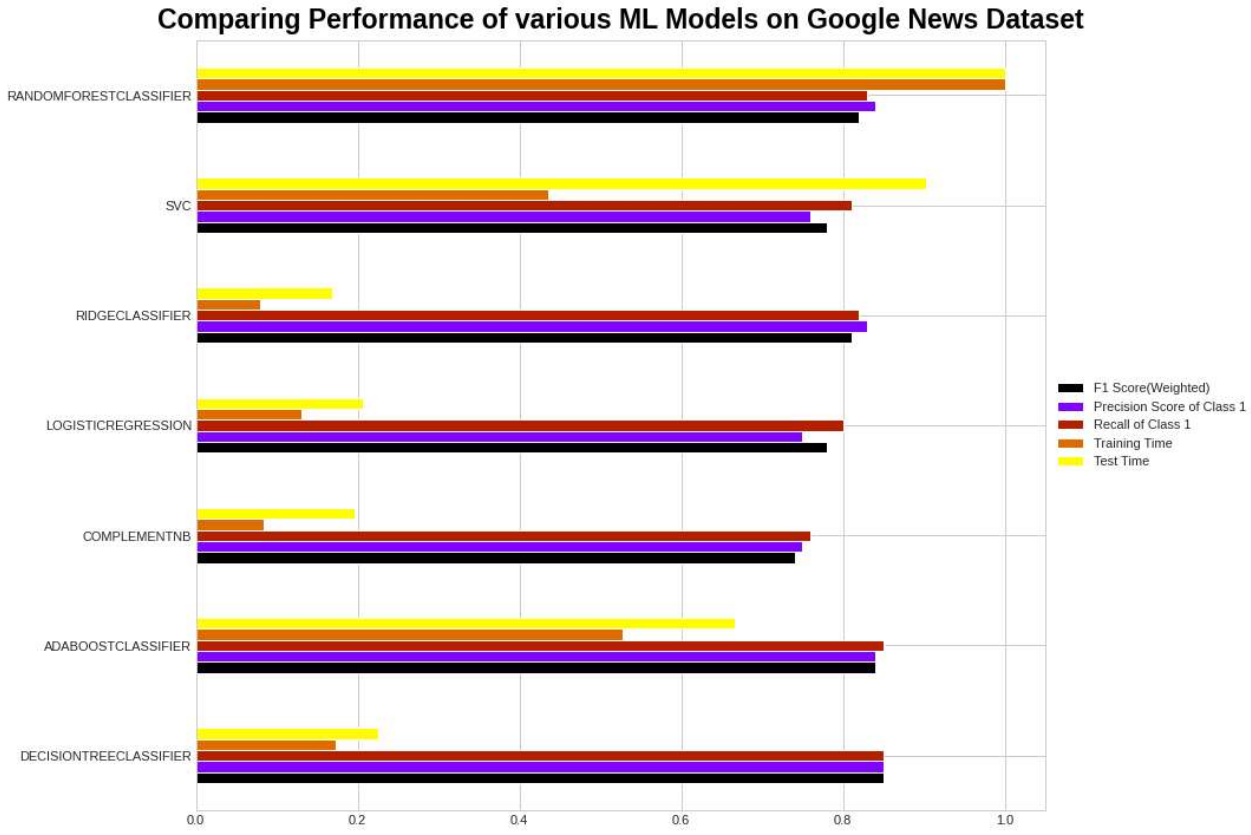
```
start=time.time()
y_pred=pipeline.predict(X_test)
stop=time.time()
test_time.append(stop-start)
print(pipelines[i].steps[1][0].upper())
model_name.append(pipelines[i].steps[1][0].upper())
f1_array.append(round(f1_score(y_test, y_pred, average='weighted'),2))
precision_array.append(round(precision_score(y_test, y_pred, average='weighted'),2))
recall_array.append(round(recall_score(y_test, y_pred, average='weighted'),2))
print("\n",classification_report(y_test, y_pred))
print("*****************************************************")
print("*****************************************************")
```

```
 Classifiation Report

 *****************************************************
 DECISIONTREECLASSIFIER

               precision    recall  f1-score   support

           -1       0.78      0.90      0.84        71
            0       0.96      0.89      0.92       149
            1       0.08      0.08      0.08        13

    accuracy                           0.85       233
   macro avg       0.61      0.62      0.61       233
weighted avg       0.85      0.85      0.85       233

 *****************************************************
 *****************************************************
 ADABOOSTCLASSIFIER

               precision    recall  f1-score   support

           -1       0.70      0.89      0.78        71
            0       0.96      0.89      0.92       149
            1       0.25      0.08      0.12        13

    accuracy                           0.85       233
   macro avg       0.64      0.62      0.61       233
weighted avg       0.84      0.85      0.84       233

 *****************************************************
 *****************************************************
 COMPLEMENTNB

               precision    recall  f1-score   support

           -1       0.77      0.48      0.59        71
            0       0.77      0.95      0.85       149
            1       0.43      0.23      0.30        13

    accuracy                           0.76       233
   macro avg       0.66      0.55      0.58       233
weighted avg       0.75      0.76      0.74       233
```

```
**************************************************
**************************************************
LOGISTICREGRESSION

              precision    recall  f1-score   support

          -1       0.72      0.66      0.69        71
           0       0.83      0.94      0.88       149
           1       0.00      0.00      0.00        13

    accuracy                           0.80       233
   macro avg       0.52      0.53      0.52       233
weighted avg       0.75      0.80      0.78       233

**************************************************
```

Through the pipeline of different models we got to know that the DecisionTreeClassifier model is the best as its accuracy is highest 85%

```
#Plotting the various performance metrix of all models
training_time=np.array(training_time)/np.max(training_time)
test_time=np.array(test_time)/np.max(test_time)
score_df=pd.DataFrame({'F1 Score(Weighted)':f1_array,
                    'Precision Score of Class 1':precision_array,
                    'Recall of Class 1':recall_array,
                     'Training Time': training_time,
                     'Test Time':test_time}, index=model_name)

f=plt.figure(figsize=(15,10))
plt.title('Comparing Performance of various ML Models on Google News Dataset', color='Black',
         fontdict={'fontsize':23})
score_df.plot(kind='barh', ax=f.gca(), cmap='gnuplot')
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
plt.show()
```



Dataset2: During the Invasion

```
X= data2.text
y=data2.label
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)


#Training the model
import time
training_time=[]

for pipeline in pipelines:
    start=time.time()
    pipeline.fit(X_train, y_train)
    stop=time.time()
    training_time.append(stop-start)


#Prediction from test dataset
from sklearn.metrics import classification_report, confusion_matrix, f1_score, precision_scor
model_name=[]
precision_array=[]
recall_array=[]
f1_array=[]
test_time=[]
print("Classifiation Report\n")
```

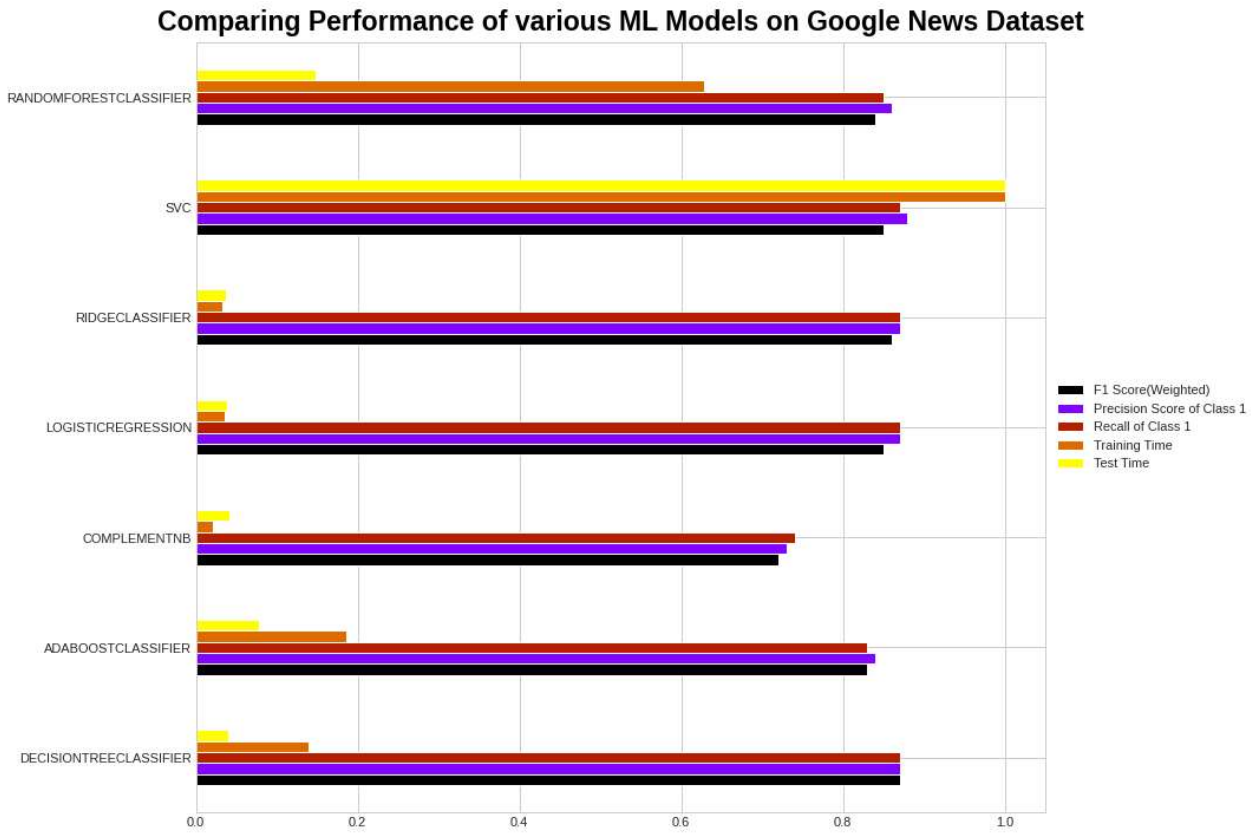
```
print("*****************************************************")
for i, pipeline in enumerate(pipelines):
    start=time.time()
    y_pred=pipeline.predict(X_test)
    stop=time.time()
    test_time.append(stop-start)
    print(pipelines[i].steps[1][0].upper())
    model_name.append(pipelines[i].steps[1][0].upper())
    f1_array.append(round(f1_score(y_test, y_pred, average='weighted'),2))
    precision_array.append(round(precision_score(y_test, y_pred, average='weighted'),2))
    recall_array.append(round(recall_score(y_test, y_pred, average='weighted'),2))
    print("\n",classification_report(y_test, y_pred))
    print("*****************************************************")
    print("*****************************************************")
```

```
 Classifiation Report

 ***************************************************
 DECISIONTREECLASSIFIER

               precision    recall  f1-score   support

           -1       0.81      0.86      0.84       339
            0       0.96      0.94      0.95       725
            1       0.57      0.53      0.55       129

    accuracy                           0.87      1193
   macro avg       0.78      0.78      0.78      1193
weighted avg       0.87      0.87      0.87      1193

 ***************************************************
 ***************************************************
 ADABOOSTCLASSIFIER

               precision    recall  f1-score   support

           -1       0.75      0.94      0.84       339
            0       0.96      0.86      0.91       725
            1       0.38      0.36      0.37       129

    accuracy                           0.83      1193
   macro avg       0.70      0.72      0.71      1193
weighted avg       0.84      0.83      0.83      1193

 ***************************************************
 ***************************************************
 COMPLEMENTNB

               precision    recall  f1-score   support

           -1       0.71      0.48      0.57       339
            0       0.76      0.94      0.84       725
            1       0.61      0.29      0.40       129

    accuracy                           0.74      1193
```

```
           macro avg       0.69      0.57      0.60      1193
        weighted avg       0.73      0.74      0.72      1193

        ****************************************************
        ****************************************************
        LOGISTICREGRESSION

                      precision    recall  f1-score   support

                  -1       0.75      0.90      0.82       339
                   0       0.93      0.96      0.94       725
                   1       0.85      0.27      0.41       129

            accuracy                           0.87      1193
           macro avg       0.84      0.71      0.72      1193
        weighted avg       0.87      0.87      0.85      1193

        ****************************************************
```

Through the pipeline of 7 models we got to know that the DecisionTreeClassifier model is the best as its accuracy is highest 87%

```
#Plotting the various performance metrix of all models
training_time=np.array(training_time)/np.max(training_time)
test_time=np.array(test_time)/np.max(test_time)
score_df=pd.DataFrame({'F1 Score(Weighted)':f1_array,
                       'Precision Score of Class 1':precision_array,
                       'Recall of Class 1':recall_array,
                        'Training Time': training_time,
                        'Test Time':test_time}, index=model_name)

f=plt.figure(figsize=(15,10))
plt.title('Comparing Performance of various ML Models on Google News Dataset', color='Black',
          fontdict={'fontsize':23})
score_df.plot(kind='barh', ax=f.gca(), cmap='gnuplot')
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
plt.show()
```



Model Training od Dataset 3: Combined 4 countries df

```
X= data3.text
y=data3.label
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)


#Training the model
import time
training_time=[]

for pipeline in pipelines:
    start=time.time()
    pipeline.fit(X_train, y_train)
    stop=time.time()
    training_time.append(stop-start)


#Prediction from test dataset
from sklearn.metrics import classification_report, confusion_matrix, f1_score, precision_scor
model_name=[]
precision_array=[]
recall_array=[]
f1_array=[]
```

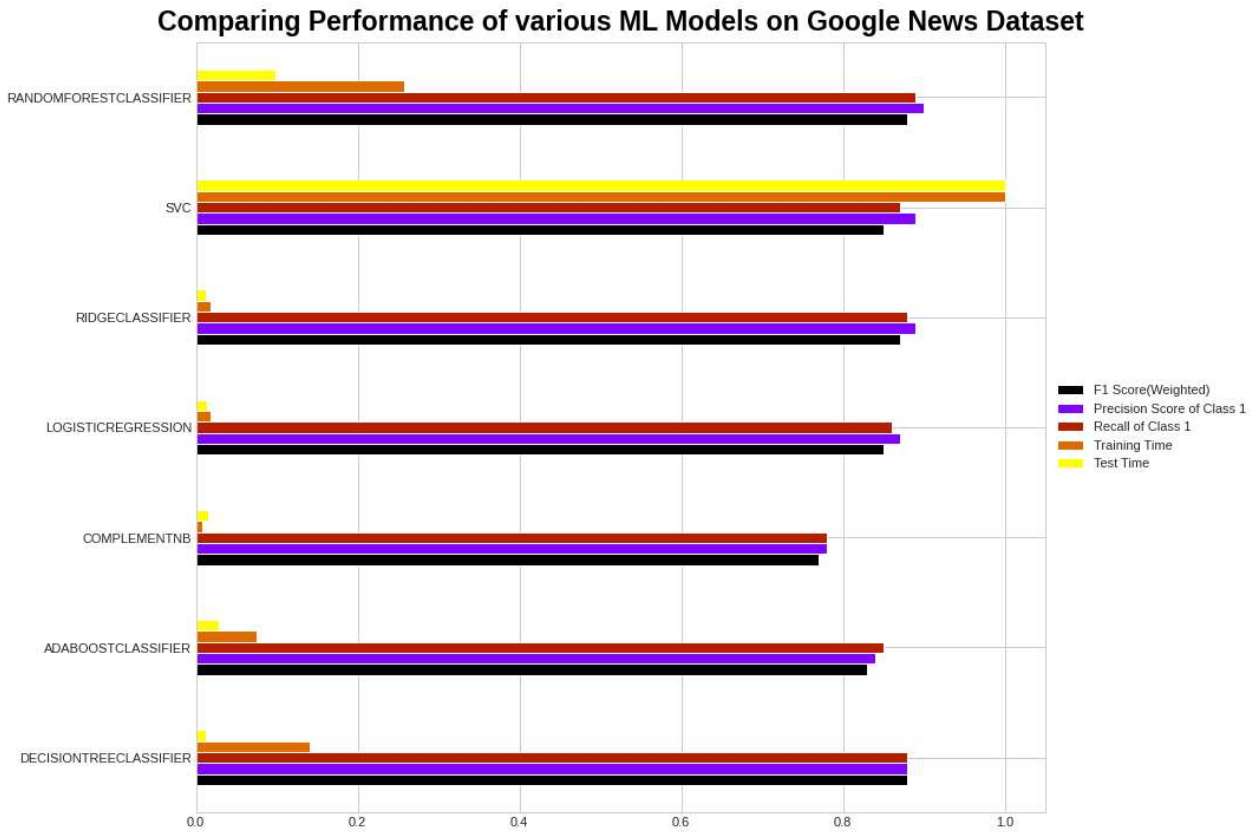
```
test_time=[]
print("Classifiation Report\n")
print("*****************************************************")
for i, pipeline in enumerate(pipelines):
    start=time.time()
    y_pred=pipeline.predict(X_test)
    stop=time.time()
    test_time.append(stop-start)
    print(pipelines[i].steps[1][0].upper())
    model_name.append(pipelines[i].steps[1][0].upper())
    f1_array.append(round(f1_score(y_test, y_pred, average='weighted'),2))
    precision_array.append(round(precision_score(y_test, y_pred, average='weighted'),2))
    recall_array.append(round(recall_score(y_test, y_pred, average='weighted'),2))
    print("\n",classification_report(y_test, y_pred))
    print("*****************************************************")
    print("*****************************************************")
```

```
Classifiation Report

*************************************************
DECISIONTREECLASSIFIER

              precision    recall  f1-score   support

          -1       0.89      0.92      0.90      1266
           0       0.96      0.95      0.95       908
           1       0.66      0.60      0.63       346

    accuracy                           0.88      2520
   macro avg       0.83      0.82      0.83      2520
weighted avg       0.88      0.88      0.88      2520

*************************************************
*************************************************
ADABOOSTCLASSIFIER

              precision    recall  f1-score   support

          -1       0.80      0.97      0.88      1266
           0       0.97      0.89      0.93       908
           1       0.64      0.26      0.37       346

    accuracy                           0.85      2520
   macro avg       0.80      0.71      0.72      2520
weighted avg       0.84      0.85      0.83      2520

*************************************************
*************************************************
COMPLEMENTNB

              precision    recall  f1-score   support

          -1       0.85      0.77      0.81      1266
           0       0.73      0.93      0.82       908
```

```
              1       0.67      0.43      0.52       346

       accuracy                           0.78      2520
      macro avg       0.75      0.71      0.72      2520
   weighted avg       0.78      0.78      0.77      2520

**************************************************
**************************************************
LOGISTICREGRESSION

              precision    recall  f1-score   support

          -1       0.81      0.98      0.89      1266
           0       0.97      0.89      0.93       908
           1       0.83      0.35      0.49       346

    accuracy                           0.86      2520
   macro avg       0.87      0.74      0.77      2520
weighted avg       0.87      0.86      0.85      2520

**************************************************
```

Through the pipeline of different models we got to know that the DecisionTreeClassifier and RandomForest model are the best as their accuracy is highest 89%

```
#Plotting the various performance metrix of all models
training_time=np.array(training_time)/np.max(training_time)
test_time=np.array(test_time)/np.max(test_time)
score_df=pd.DataFrame({'F1 Score(Weighted)':f1_array,
                       'Precision Score of Class 1':precision_array,
                       'Recall of Class 1':recall_array,
                        'Training Time': training_time,
                        'Test Time':test_time}, index=model_name)

f=plt.figure(figsize=(15,10))
plt.title('Comparing Performance of various ML Models on Google News Dataset', color='Black',
          fontdict={'fontsize':23})
score_df.plot(kind='barh', ax=f.gca(), cmap='gnuplot')
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.5))
plt.show()
```



Through the graph we can conclude that the Decision Tree Classifier is more accurate because when compared to Random Forest the time taken by training time and testing time is lesser and precision and recall is also better.

Deep learning Approach: Tensor Flow and Keras

Train Test Split

```
##store headlines and labels in respective lists
text = list(data2['text'])
labels = list(data2['label'])
##sentences
training_text = text[0:100]
testing_text = text[100:]
##labels
training_labels = labels[0:100]
testing_labels = labels[100:]
```

Set up the tokenizer from Tensor to pre-process the data.

```
#preprocess
tokenizer = Tokenizer(num_words=10000, oov_token= "<OOV>")
tokenizer.fit_on_texts(training_text)
word_index = tokenizer.word_index
training_sequences = tokenizer.texts_to_sequences(training_text)
training_padded = pad_sequences(training_sequences, maxlen=120, padding='post', truncating='p
testing_sequences = tokenizer.texts_to_sequences(testing_text)
testing_padded = pad_sequences(testing_sequences, maxlen=120, padding='post', truncating='pos
# convert lists into numpy arrays to make it work with TensorFlow
training_padded = np.array(training_padded)
training_labels = np.array(training_labels)
testing_padded = np.array(testing_padded)
testing_labels = np.array(testing_labels)
```

Define & train the Sequential model

```
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(10000, 16, input_length=120),
    tf.keras.layers.GlobalAveragePooling1D(),
    tf.keras.layers.Dense(24, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
##compile the model
model.compile(loss='binary_crossentropy',optimizer='adam',metrics=['accuracy'])

model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_2 (Embedding)     (None, 120, 16)           160000

 global_average_pooling1d_2   (None, 16)               0
 (GlobalAveragePooling1D)

 dense_4 (Dense)             (None, 24)                408

 dense_5 (Dense)             (None, 1)                 25

=================================================================
Total params: 160,433
Trainable params: 160,433
Non-trainable params: 0
_________________________________________________________________
```

```
num_epochs = 10
history = model.fit(training_padded,
                    training_labels,
                    epochs=num_epochs,
                    validation_data=(testing_padded, testing_labels),
                    verbose=2)
```

```
Epoch 1/10
4/4 - 1s - loss: 0.6946 - accuracy: 0.2000 - val_loss: 0.6865 - val_accuracy: 0.6162 - 1
Epoch 2/10
4/4 - 0s - loss: 0.6813 - accuracy: 0.5600 - val_loss: 0.6739 - val_accuracy: 0.6162 - 2
Epoch 3/10
4/4 - 0s - loss: 0.6675 - accuracy: 0.5600 - val_loss: 0.6627 - val_accuracy: 0.6162 - 3
Epoch 4/10
4/4 - 0s - loss: 0.6547 - accuracy: 0.5600 - val_loss: 0.6511 - val_accuracy: 0.6162 - 3
Epoch 5/10
4/4 - 0s - loss: 0.6408 - accuracy: 0.5600 - val_loss: 0.6388 - val_accuracy: 0.6162 - 3
Epoch 6/10
4/4 - 0s - loss: 0.6260 - accuracy: 0.5600 - val_loss: 0.6248 - val_accuracy: 0.6162 - 2
Epoch 7/10
4/4 - 0s - loss: 0.6097 - accuracy: 0.5600 - val_loss: 0.6094 - val_accuracy: 0.6162 - 3
Epoch 8/10
4/4 - 0s - loss: 0.5918 - accuracy: 0.5600 - val_loss: 0.5934 - val_accuracy: 0.6162 - 2
Epoch 9/10
4/4 - 0s - loss: 0.5721 - accuracy: 0.5600 - val_loss: 0.5768 - val_accuracy: 0.6162 - 2
Epoch 10/10
4/4 - 0s - loss: 0.5529 - accuracy: 0.5600 - val_loss: 0.5556 - val_accuracy: 0.6162 - 2
```

Model Training through Transformer- BERT (On a section of our Data)

Text Preprocessing for BERT

```
aux = pd.DataFrame(
    {
        "Setting": ["Model Name", "Number of Epochs", "Batch Size",
                    "Max Sequence Length", "Learning Rate", "Accumulation Steps",
                    "Random Seed"],
        "Value": ["bert-base-uncased", 4, 32, 36, 5e-5, 4, 42]
    }
)

display(HTML('<span style="font-weight:bold">' + 'Table 4 - BERT Model Setup'\
             + '</span>'),aux)
```

Table 4 - BERT Model Setup

	Setting	Value
0	Model Name	bert-base-uncased
1	Number of Epochs	4
2	Batch Size	32
3	Max Sequence Length	36
4	Learning Rate	0.00005
5	Accumulation Steps	4

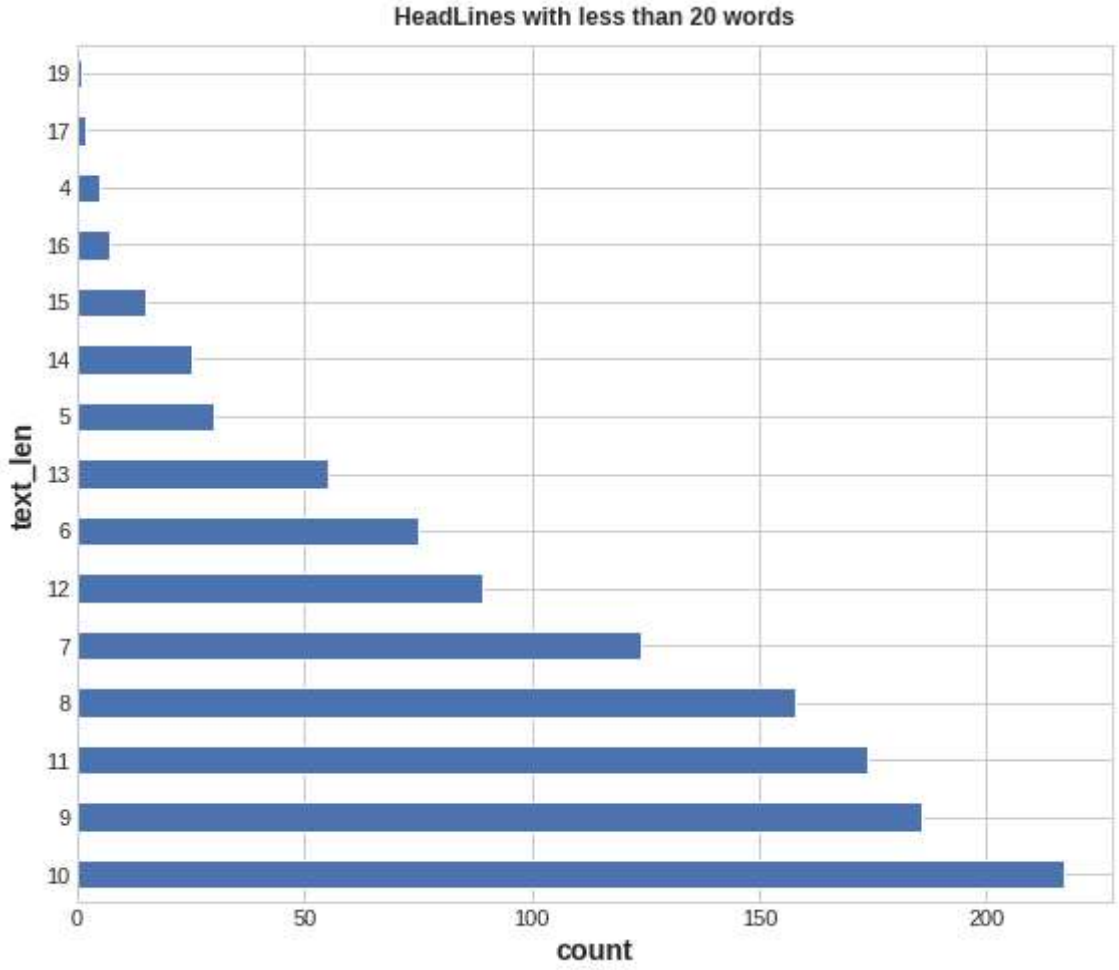
```
#make a copy of the dataframe
df_1 = dfl1.copy()
#define a function which handles the text preprocessing
def clean_text(df_1):
    """
    Text preprocessing:
    tokenize, make lower case,
    Remove Stop word, punctuation, digit
    lemmatize
    """
    nlp = spacy.load('en')
    for i in range(df_1.shape[0]):
        doc = nlp(df_1['text'][i])
        # Word Tokenize
        #token = [w.text for w in doc]

        # Make Lower case
        # Remove Stop word, punctuation, digit and lemmatize
        text = [w.lemma_.lower().strip() for w in doc
               if not (w.is_stop |
                    w.is_punct |
                    w.is_digit)
               ]
        text = " ".join(text)


        df_1['text'][i] = text
    return df_1



df_1 = clean_text(df_1)


text_len = []
for text in df_1.text:
    title_len = len(text.split())
    text_len.append(title_len)
```

```
df_1['text_len'] = text_len
```

Text with less than 20 words

```
plt.figure(figsize=(8,7))
data=df_1[df_1['text_len']<20]
data['text_len'].value_counts().plot(kind='barh', figsize=(8,7))
plt.xlabel("count")
plt.ylabel("text_len")
plt.title("HeadLines with less than 20 words")
```

```
Text(0.5, 1.0, 'HeadLines with less than 20 words')
```




```
print(f" DF SHAPE: {df_1.shape}")
```

```
 DF SHAPE: (1164, 3)
```

Data Processing for BERT

Raw text is incompatible with machine learning models. We'll need to convert text to numerical form. When it comes to its representation, BERT requires even more attention.

Here are the requirements:

1. Add special tokens to separate sentences and do classification
2. Pass sequences of constant length (introduce padding)
3. Create array of 0s (pad token) and 1s (real token) called attention mask.

BERT has a number of model architectures, and I'll be combining one of them with manual preprocessing. I'm using the cased version.

The tokenizer will break the sentence into words and give numerical values to each word.

```
# Set the model name
MODEL_NAME = 'bert-base-uncased'

# Build a BERT based tokenizer
tokenizer = BertTokenizer.from_pretrained(MODEL_NAME)
```

```
Downloading: 100%  226k/226k [00:00<00:00, 2.20MB/s]
Downloading: 100%  28.0/28.0 [00:00<00:00, 727B/s]
Downloading: 100%  570/570 [00:00<00:00, 16.5kB/s]
```

```
#Some of the common BERT tokens
print(tokenizer.sep_token, tokenizer.sep_token_id) # marker for ending of a sentence
print(tokenizer.cls_token, tokenizer.cls_token_id) # start of each sentence, so BERT knows we
print(tokenizer.pad_token, tokenizer.pad_token_id) # special token for padding
print(tokenizer.unk_token, tokenizer.unk_token_id) # tokens not found in training set
```

```
[SEP] 102
[CLS] 101
[PAD] 0
[UNK] 100
```

BERT works with fixed-length sequences. To determine the maximum length, we'll utilize a simple technique. Let's keep track of each review's token length.

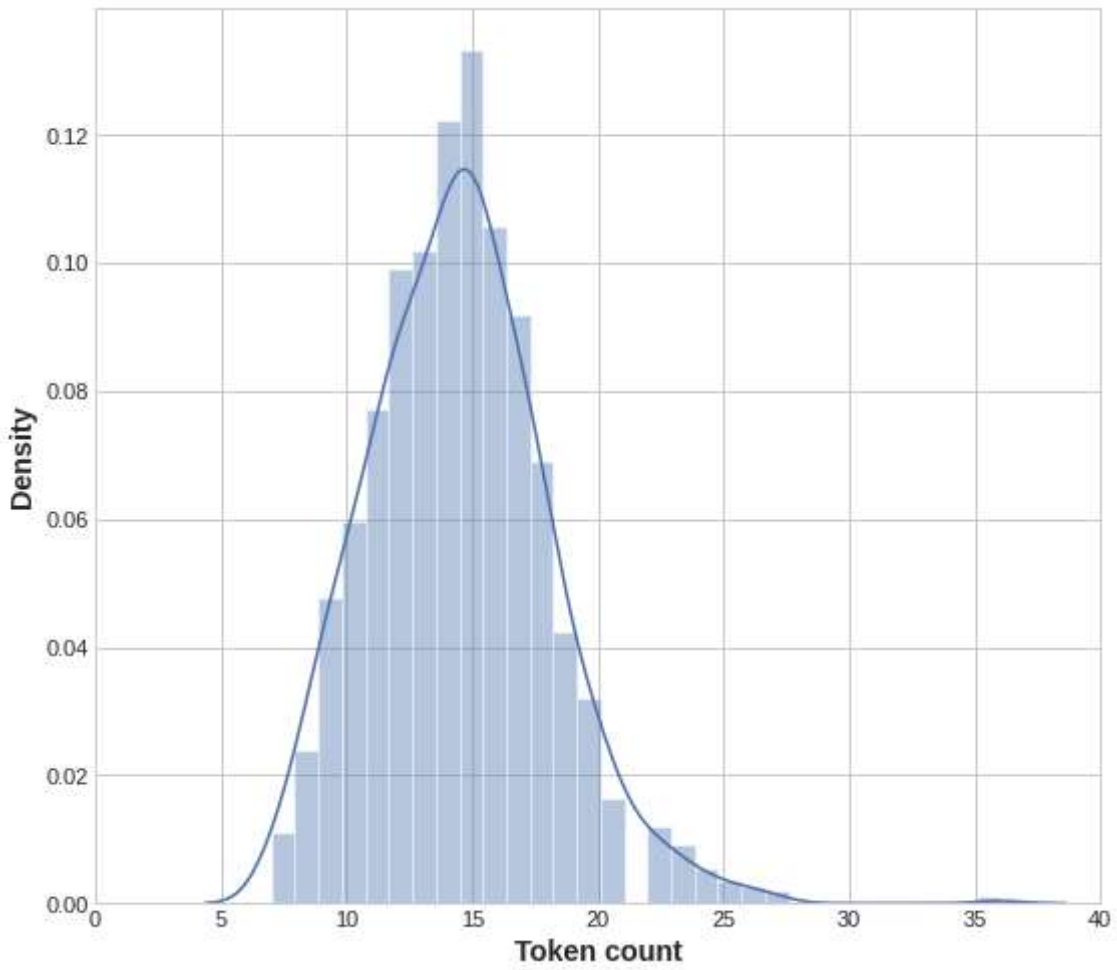
```
# Store length of each review
token_lens = []

# Iterate through the text slide
for txt in data1.text:
    tokens = tokenizer.encode(txt, max_length=36, truncation=True)
    token_lens.append(len(tokens))

max_len=np.max(token_lens)
```

```
print(f"MAX TOKENIZED SENTENCE LENGTH: {max_len}")
```

```
MAX TOKENIZED SENTENCE LENGTH: 36
```

```
# plot the distribution of title lengths
sns.distplot(token_lens)
plt.xlim([0, 40]);
plt.xlabel('Token count')
```

```
Text(0.5, 0, 'Token count')
```



Most of the reviews seem to contain less than 40 tokens, but we'll be on the safe side and choose a maximum length of 20.

Label counts in each df:

```
df_1['label'].value_counts()
```

```
 0    675
-1    389
 1    100
Name: label, dtype: int64
```

Class Balancing by RandomOverSampler

```
ros = RandomOverSampler()
train_x, train_y = ros.fit_resample(np.array(df_1['text']).reshape(-1, 1), np.array(df_1['lab
train_df1 = pd.DataFrame(list(zip([x[0] for x in train_x], train_y)), columns = ['text', 'lab
```

```
train_df1['label'].value_counts()
```

```
 0    675
-1    675
 1    675
Name: label, dtype: int64
```

Train- Validation-Test split

```
X = train_df1['text'].values
y = train_df1['label'].values
```

```
X_train, X_valid, y_train, y_valid = train_test_split(X, y, test_size=0.1, stratify=y, random
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

One hot Encoding

```
y_train_a = y_train.copy()
y_valid_a = y_valid.copy()
y_test_a = y_test.copy()
```

```
ohe = preprocessing.OneHotEncoder()
y_train = ohe.fit_transform(np.array(y_train).reshape(-1, 1)).toarray()
y_valid = ohe.fit_transform(np.array(y_valid).reshape(-1, 1)).toarray()
y_test = ohe.fit_transform(np.array(y_test).reshape(-1, 1)).toarray()
```

```
print(f"TRAINING DATA: {X_train.shape[0]}\nVALIDATION DATA: {X_valid.shape[0]}\nTESTING DATA:
```

```
TRAINING DATA: 1620
VALIDATION DATA: 203
TESTING DATA: 405
```

Before implementing BERT, we will define a simple Naive Bayes baseline model to classify the sentiments.

Naive Bayes Classifier

First we need to tokenize the text using CountVectorizer

```
clf = CountVectorizer()
X_train_cv =  clf.fit_transform(X_train)
X_test_cv = clf.transform(X_test)
```

Then we create the TF-IDF (term-frequency times inverse document-frequency) versions of the tokenized text.

```
tf_transformer = TfidfTransformer(use_idf=True).fit(X_train_cv)
X_train_tf = tf_transformer.transform(X_train_cv)
X_test_tf = tf_transformer.transform(X_test_cv)
```

Now we can define the Naive Bayes Classifier model

```
nb_clf = MultinomialNB()
```

```
nb_clf.fit(X_train_tf, y_train_a)
```

```
MultinomialNB()
```

```
nb_pred = nb_clf.predict(X_test_tf)
```

```
print('\tClassification Report for Naive Bayes:\n\n',(classification_report(y_test_a, nb_pred
```

```
            Classification Report for Naive Bayes:

                  precision    recall  f1-score   support

              -1       0.90      0.73      0.81       139
               0       0.74      0.85      0.79       124
               1       0.90      0.94      0.92       142

        accuracy                           0.84       405
       macro avg       0.85      0.84      0.84       405
    weighted avg       0.85      0.84      0.84       405
```

Confusion Matrix

```
print('Confusion matrix\n',confusion_matrix(y_test_a,nb_pred))
```

```
Confusion matrix
 [[102  31   6]
 [  9 106   9]
 [  2   7 133]]
```

The algorithm's performance is decent. The F1 score for the more populated classes (Neutral and Positive emotions) is over 75%. The total accuracy, in particular, is 84%.

BERT Sentiment Analysis

```
MAX_LEN=36
```

```
def tokenize(data,max_len=MAX_LEN) :
    input_ids = []
    attention_masks = []
    for i in range(len(data)):
        encoded = tokenizer.encode_plus(
            data[i],
            add_special_tokens=True,
            max_length=MAX_LEN,
            padding='max_length',
            return_attention_mask=True
        )
        input_ids.append(encoded['input_ids'])
        attention_masks.append(encoded['attention_mask'])
    return np.array(input_ids),np.array(attention_masks)
```

Then, we apply the tokenizer function to the train, validation and test sets.

```
train_input_ids, train_attention_masks = tokenize(X_train, MAX_LEN)
val_input_ids, val_attention_masks = tokenize(X_valid, MAX_LEN)
test_input_ids, test_attention_masks = tokenize(X_test, MAX_LEN)
```

BERT Modelling

```
bert_model = TFBertModel.from_pretrained('bert-base-uncased')
```

```
Downloading:                                    511M/511M [00:12<00:00,
100%                                            38.9MB/s]
Some layers from the model checkpoint at bert-base-uncased were not used when init
- This IS expected if you are initializing TFBertModel from the checkpoint of a mo
- This IS NOT expected if you are initializing TFBertModel from the checkpoint of
```

Then, we develop a new function to host the pre-trained BERT model and attach a three-neuron output layer to it, which is required to do classification of the dataset's three separate classes (the 3 emotions).

```
def create_model(bert_model, max_len=MAX_LEN):

    ##params###
    opt = tf.keras.optimizers.Adam(learning_rate=1e-5, decay=1e-7)
    loss = tf.keras.losses.CategoricalCrossentropy()
    accuracy = tf.keras.metrics.CategoricalAccuracy()


    input_ids = tf.keras.Input(shape=(max_len,),dtype='int32')

    attention_masks = tf.keras.Input(shape=(max_len,),dtype='int32')

    embeddings = bert_model([input_ids,attention_masks])[1]

    output = tf.keras.layers.Dense(3, activation="softmax")(embeddings)

    model = tf.keras.models.Model(inputs = [input_ids,attention_masks], outputs = output)

    model.compile(opt, loss=loss, metrics=accuracy)


    return model


model = create_model(bert_model, MAX_LEN)
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==========================================================================================
 input_1 (InputLayer)           [(None, 36)]         0           []

 input_2 (InputLayer)           [(None, 36)]         0           []

 tf_bert_model (TFBertModel)    TFBaseModelOutputWi  109482240   ['input_1[0][0]',
                                thPoolingAndCrossAt               'input_2[0][0]']
                                tentions(last_hidde
```

```
                                n_state=(None, 36,
                                768),
                                 pooler_output=(Non
                                e, 768),
                                 past_key_values=No
                                ne, hidden_states=N
                                one, attentions=Non
                                e, cross_attentions
                                =None)

 dense_6 (Dense)                (None, 3)            2307        ['tf_bert_model[0][1]']

==================================================================================================
Total params: 109,484,547
Trainable params: 109,484,547
Non-trainable params: 0
__________________________________________________________________________________________________
```

Finally we can start fine tuning the BERT transformer !

```
history_bert = model.fit([train_input_ids,train_attention_masks], y_train, validation_data=([
```

```
Epoch 1/4
51/51 [==============================] - 642s 12s/step - loss: 1.0590 - categorical_accu
Epoch 2/4
51/51 [==============================] - 607s 12s/step - loss: 0.6797 - categorical_accu
Epoch 3/4
51/51 [==============================] - 617s 12s/step - loss: 0.3493 - categorical_accu
Epoch 4/4
51/51 [==============================] - 624s 12s/step - loss: 0.2097 - categorical_accu
```

Here, we can see that the accuracy increases with each epoch and loss decreases. The accuracy which we achieved here is 93.58% which is much better than our our base model naive bayes which we have taken earlier.

BERT results

```
result_bert = model.predict([test_input_ids,test_attention_masks])
result_bert
```

```
array([[0.00533346, 0.00960299, 0.98506355],
       [0.0049388 , 0.01658746, 0.9784737 ],
       [0.28811023, 0.69741166, 0.01447817],
       ...,
       [0.00967442, 0.02297704, 0.9673486 ],
```

```
            [0.00513489, 0.02094309, 0.9739221 ],
            [0.00407779, 0.03616557, 0.9597566 ]], dtype=float32)
```

```
y_pred_bert =  np.zeros_like(result_bert)
y_pred_bert[np.arange(len(y_pred_bert)), result_bert.argmax(1)] = 1
```

Confusion Matrix:

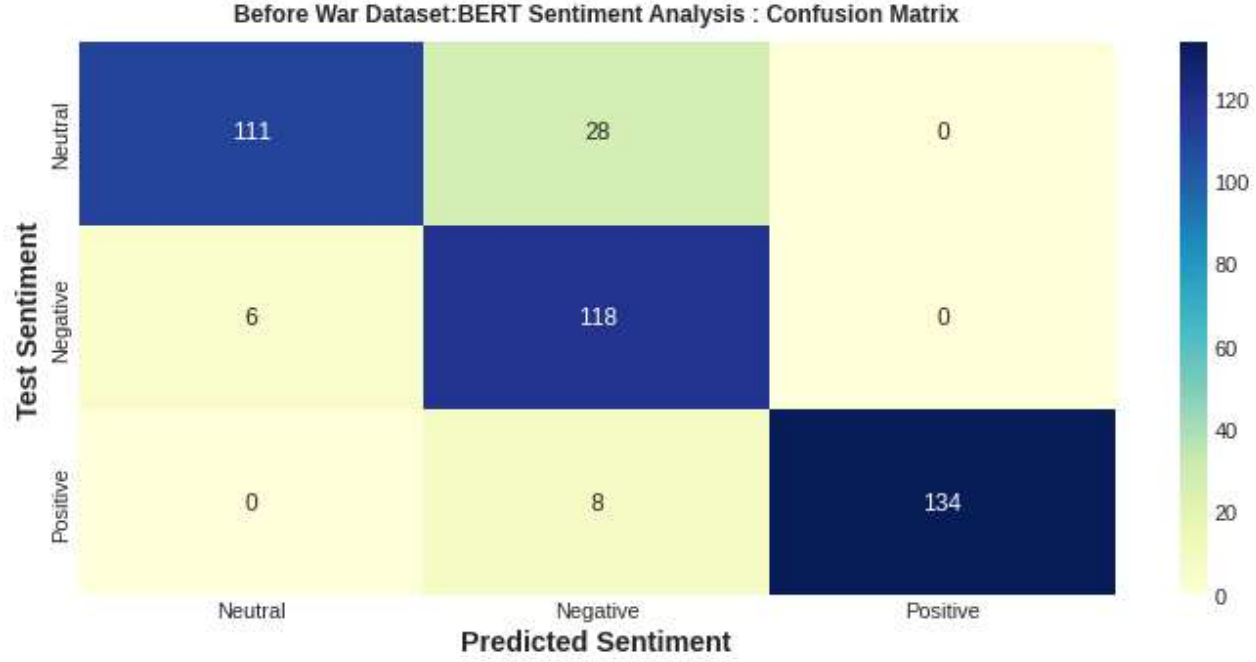
```
cf_matrix  = confusion_matrix(y_test.argmax(1),y_pred_bert.argmax(1))
```

```
print('Before War Dataset:BERT Sentiment Analysis : Confusion Matrix\n', cf_matrix)
```

```
Before War Dataset:BERT Sentiment Analysis : Confusion Matrix
 [[111  28   0]
 [  6 118   0]
 [  0   8 134]]
```

```
cf_matrix = confusion_matrix(y_test.argmax(1),y_pred_bert.argmax(1))
df_cm = pd.DataFrame(cf_matrix, index=class_names, columns=class_names)
plt.figure(figsize=(10, 5))
ax = sns.heatmap(df_cm, annot=True, fmt="d", cmap="YlGnBu")
ax.set_ylabel('Test Sentiment')
ax.set_xlabel('Predicted Sentiment')
ax.set_title("Before War Dataset:BERT Sentiment Analysis : Confusion Matrix", size=12)
```

```
Text(0.5, 1.0, 'Before War Dataset:BERT Sentiment Analysis : Confusion Matrix')
```



Classification Report

```
print('\tClassification Report for BERT:\n\n',classification_report(y_test,y_pred_bert, targe
```

```
            Classification Report for BERT:

                    precision    recall  f1-score   support

           Neutral       0.95      0.80      0.87       139
          Negative       0.77      0.95      0.85       124
          Positive       1.00      0.94      0.97       142

         micro avg       0.90      0.90      0.90       405
         macro avg       0.90      0.90      0.90       405
      weighted avg       0.91      0.90      0.90       405
       samples avg       0.90      0.90      0.90       405
```

After model training through BERT, we found that the positive sentiment count were more befor the war which is true in real scenario. Hence, our BERT model performed well and gave the best results of accuracy.

Conclusion

As a part of the invasion sentiment analysis we have tried ddifferent approaches as follows:

1. After the data processing and labeling of all three datasets we found that:
   a) In before invasion dataset the neutral and negative sentiment counts were more.
   b) Following the same in during invasion dataset, we found that during the war the negative count was far more than the positive and neutral.
   c) In combined dataset the neutral count was more than negative and positive. And as per the study these countries were neutral about the invasion due to the business relations with Russia.
2. After applying pipeline we found of 6 models we found that the Decision Tree Classifier outperformed among every other models with accuracy more than 85% in all the three datasets.
3. To further analysis, we delve into depp learning apprach and tried to find out the sentiments accuracy through tensorflow and keras but couldn't find the better accuracy.
4. At last we applied transformer approach for the sentiment analysis to a section of our data. Surprisingly, we got the best results using transformer. We applied BERT model for model training and got the accuracy more than 93% with 4 epochs which was way better the models we used in pipeline.