2_model

June 9, 2022

0.0.1 Import Libraries

[ ]:
```
# Data processing
import pandas as pd
import numpy as np

# Data visualization
import seaborn as sns
import matplotlib.pyplot as plt
import matplotlib as mpl
sns.set_theme(style="whitegrid") # all charts will have a light grid
# from wordcloud import WordCloud, STOPWORDS
from nltk.probability import FreqDist

# Deeo learning
import tensorflow as tf

# Text valuation/detection
import langid
from nltk.sentiment.vader import SentimentIntensityAnalyzer as SIA

# Text Preprocessing
import re
import string
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
nltk.download('words')
words = set(nltk.corpus.words.words())
import spacy
from spacy import displacy
# Text stemming
from nltk.stem.porter import *
# Text Vectorization
from tensorflow.keras.preprocessing.text import Tokenizer
# padding sequences
from tensorflow.keras.preprocessing.sequence import pad_sequences
from sklearn.feature_extraction.text import CountVectorizer
```

```
# model featuring
from sklearn.model_selection import train_test_split
from tensorflow.keras.optimizers import SGD, RMSprop
from tensorflow.keras.callbacks import ModelCheckpoint, LearningRateScheduler,␣
 ↪History
from tensorflow.keras.models import load_model

# model
from tensorflow.keras.models import Sequential, Model
from tensorflow.keras.layers import Embedding, Conv1D, MaxPooling1D,␣
 ↪Bidirectional, LSTM, Dense, Dropout, Flatten, SpatialDropout1D, GRU, Input
# Unsupervised
from sklearn.decomposition import LatentDirichletAllocation


# Analyze results
from sklearn.metrics import confusion_matrix
from keras.metrics import Precision, Recall

# Misc
import pickle
import os
import glob
import warnings
warnings.filterwarnings("ignore")
import datetime as dt
```

```
---------------------------------------------------------------------------
KeyboardInterrupt                         Traceback (most recent call last)
c:\Users\Gumo\Desktop\Git\Class\CIS63_NLPTimeSeries\2_model.ipynb Cell 2' in␣
 ↪<module>
      <a href='vscode-notebook-cell:/c%3A/Users/Gumo/Desktop/Git/Class/
 ↪CIS63_NLPTimeSeries/2_model.ipynb#ch0000001?line=2'>3</a> import numpy as np
      <a href='vscode-notebook-cell:/c%3A/Users/Gumo/Desktop/Git/Class/
 ↪CIS63_NLPTimeSeries/2_model.ipynb#ch0000001?line=4'>5</a> # Data visualizatio
----> <a href='vscode-notebook-cell:/c%3A/Users/Gumo/Desktop/Git/Class/
 ↪CIS63_NLPTimeSeries/2_model.ipynb#ch0000001?line=5'>6</a> import seaborn as s
      <a href='vscode-notebook-cell:/c%3A/Users/Gumo/Desktop/Git/Class/
 ↪CIS63_NLPTimeSeries/2_model.ipynb#ch0000001?line=6'>7</a> import matplotlib.
 ↪pyplot as plt
      <a href='vscode-notebook-cell:/c%3A/Users/Gumo/Desktop/Git/Class/
 ↪CIS63_NLPTimeSeries/2_model.ipynb#ch0000001?line=7'>8</a> import matplotlib a
 ↪mpl
```

```
File c:\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\seaborn\__init__.py:
↪2, in <module>
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/__init__.py?line=0'>1</a> # Import seaborn objects
----> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/__init__.py?line=1'>2</a> from .rcmod import *  # noqa: F401,F403
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/__init__.py?line=2'>3</a> from .utils import *  # noqa: F401,F403
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/__init__.py?line=3'>4</a> from .palettes import *  # noqa: F401,F403

File c:\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\seaborn\rcmod.py:7 ␣
↪in <module>
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/rcmod.py?line=4'>5</a> import matplotlib as mpl
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/rcmod.py?line=5'>6</a> from cycler import cycler
----> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/rcmod.py?line=6'>7</a> from . import palettes
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=9'>10</a> __all__ = ["set_theme", "set",␣
↪"reset_defaults", "reset_orig",
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=10'>11</a>            "axes_style", "set_style",␣
↪"plotting_context", "set_context",
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=11'>12</a>            "set_palette"]
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=14'>15</a> _style_keys = [
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=15'>16</a>
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=16'>17</a>     "axes.facecolor",
  (…)
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=51'>52</a>
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/rcmod.py?line=52'>53</a> ]

File c:\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\seaborn\palettes.py:
↪9, in <module>
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/palettes.py?line=4'>5</a> import matplotlib as mpl
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/palettes.py?line=6'>7</a> from .external import husl
----> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪seaborn/palettes.py?line=8'>9</a> from .utils import desaturate,␣
↪get_color_cycle
```

```
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/palettes.py?line=9'>10</a> from .colors import xkcd_rgb, crayons
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/palettes.py?line=12'>13</a> __all__ = ["color_palette", "hls_palette"␣
↪"husl_palette", "mpl_palette",
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/palettes.py?line=13'>14</a>             "dark_palette",␣
↪"light_palette", "diverging_palette",
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/palettes.py?line=14'>15</a>             "blend_palette",␣
↪"xkcd_palette", "crayon_palette",
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/palettes.py?line=15'>16</a>             "cubehelix_palette",␣
↪"set_color_codes"]

File c:\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\seaborn\utils.py:
↪10, in <module>
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-package
↪seaborn/utils.py?line=6'>7</a> from urllib.request import urlopen, urlretriev
     <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-package
↪seaborn/utils.py?line=8'>9</a> import numpy as np
---> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/utils.py?line=9'>10</a> from scipy import stats
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/utils.py?line=10'>11</a> import pandas as pd
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
↪seaborn/utils.py?line=11'>12</a> import matplotlib as mpl

File c:\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\scipy\__init__.py:
↪136, in <module>
   <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=132'>133</a> from scipy.version import version as␣
↪__version__
   <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=134'>135</a> # Allow distributors to run custom init␣
↪code
--> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=135'>136</a> from . import _distributor_init
   <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=137'>138</a> from scipy._lib import _pep440
   <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=138'>139</a> # In maintenance branch, change to␣
↪np_maxversion N+3 if numpy is at N
   <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages/
↪scipy/__init__.py?line=139'>140</a> # See setup.py for more details

File c:
↪\Users\Gumo\Desktop\Git\desktop_env\lib\site-packages\scipy\_distributor_init
↪py:64, in <module>
```

```
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
 ↪scipy/_distributor_init.py?line=61'>62</a>     os.chdir(libs_path)
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
 ↪scipy/_distributor_init.py?line=62'>63</a>     for filename in glob.glob(os.
 ↪path.join(libs_path, '*dll')):
---> <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
 ↪scipy/_distributor_init.py?line=63'>64</a>         WinDLL(os.path.
 ↪abspath(filename))
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
 ↪scipy/_distributor_init.py?line=64'>65</a> finally:
    <a href='file:///c%3A/Users/Gumo/Desktop/Git/desktop_env/lib/site-packages
 ↪scipy/_distributor_init.py?line=65'>66</a>     os.chdir(owd)

File ~\AppData\Local\Programs\Python\Python310\lib\ctypes\__init__.py:374, in␣
 ↪CDLL.__init__(self, name, mode, handle, use_errno, use_last_error, winmode)
   <a href='file:///c%3A/Users/Gumo/AppData/Local/Programs/Python/Python310/li
 ↪ctypes/__init__.py?line=370'>371</a> self._FuncPtr = _FuncPtr
   <a href='file:///c%3A/Users/Gumo/AppData/Local/Programs/Python/Python310/li
 ↪ctypes/__init__.py?line=372'>373</a> if handle is None:
--> <a href='file:///c%3A/Users/Gumo/AppData/Local/Programs/Python/Python310/li
 ↪ctypes/__init__.py?line=373'>374</a>     self._handle = _dlopen(self._name,␣
 ↪mode)
   <a href='file:///c%3A/Users/Gumo/AppData/Local/Programs/Python/Python310/li
 ↪ctypes/__init__.py?line=374'>375</a> else:
   <a href='file:///c%3A/Users/Gumo/AppData/Local/Programs/Python/Python310/li
 ↪ctypes/__init__.py?line=375'>376</a>     self._handle = handle

KeyboardInterrupt:
```

[ ]:
```
# read encoding of the file, so we can import it.
import chardet
with open('data_model/all-data.csv', 'rb') as rawdata:
    result = chardet.detect(rawdata.read(100000))
result
```

[ ]:
```
{'encoding': 'Windows-1252', 'confidence': 0.73, 'language': ''}
```

0.0.2 Functions

[ ]:
```
def tweet_to_words(tweet):
    ''' Convert tweet text into a sequence of words '''

    # convert to lowercase
    text = tweet.lower()
    # remove non letters
    text = re.sub(r"[^a-zA-Z0-9]", " ", text)
    # tokenize
    words = text.split()
```

```
    wds=stopwords.words("english")
    wds.remove('not')
    # remove stopwords
    words = [w for w in words if w not in wds]
    # apply stemming
    words = [PorterStemmer().stem(w) for w in words]
    # return list
    return words

# remove url
def remove_url(txt):
    return " ".join(re.sub("([-a-zA-Z0-9@:%_\+.~#?&//=]{2,256}\.[a-z]{2,4}\b(\/
 ↪[-a-zA-Z0-9@:%_\+.~#?&//=]*)?)", "", txt).split())
#remove hashtag #
def remove_hashtag(txt):
    return " ".join(re.sub("([#]+)([0-9A-Z_]*[A-Z_]+[a-z0-9_üÀ-ÖØ-öø-ÿ]*)", "",␣
 ↪txt).split())
# remove mention @
def remove_at(txt):
    return " ".join(re.sub("(\@[a-zA-Z0-9_%]*)", "", txt).split())

# remove stopwords and puncturations
def get_text_processing(text):
    stpword = stopwords.words('english')
    stpword.remove('not')
    no_punctuation = [char for char in text if char not in string.punctuation]
    no_punctuation = ''.join(no_punctuation)
    return ' '.join([word for word in no_punctuation.split() if word.lower()␣
 ↪not in stpword])



# remove #, @, url, stopwords, punctuations, stemming
def removeFunc(a):
    x = remove_at(a)
    x = remove_hashtag(x)
    x = remove_url(x)
    x= get_text_processing(x.lower())
    return x

# Max length of review
def get_max_length(x):
    review_length = []
    for review in x:
        review_length.append(len(review))
```

```
    return int(np.ceil(np.mean(review_length)))
```

0.1 Import Data/Cleaning

0.1.1 Data Loading

[ ]:
```
df1 = pd.read_csv('data_model/Twitter_Data.csv')
df2 = pd.read_csv('data_model/apple-twitter-sentiment-texts.csv')
df3 = pd.read_csv('data_model/Reddit_Data.csv')
df4 = pd.read_csv('data_model/all-data.
 ↪csv',encoding='Windows-1252',usecols=[0,1],names=['category','text'])
```

[ ]:
```
df1.head(1)
```

[ ]:
```
                                          clean_text  category
0  when modi promised "minimum government maximum…        -1.0
```

[ ]:
```
# rename columns
df2 = df2.rename(columns={'text': 'clean_text', 'sentiment':'category'})
# change values
df2['category'] = df2['category'].map({-1: -1.0, 0: 0.0, 1:1.0})
df2.head(1)
```

[ ]:
```
                                          clean_text  category
0  Wow. Yall needa step it up @Apple RT @heynyla:…        -1.0
```

[ ]:
```
# rename columns
df3 = df3.rename(columns={'clean_comment': 'clean_text'})
# change values
df3['category'] = df3['category'].map({-1: -1.0, 0: 0.0, 1:1.0})
df3.head(1)
```

[ ]:
```
                                          clean_text  category
0   family mormon have never tried explain them t…         1.0
```

[ ]:
```
# rename columns
df4 = df4.rename(columns={'text': 'clean_text'})
# reorder columns
df4 = df4[['clean_text', 'category']]
# change values
df4['category'] = df4['category'].map({'negative': -1.0, 'neutral': 0.0,␣
 ↪'positive':1.0})
df4.head(1)
```

[ ]:
```
                                          clean_text  category
0  According to Gran , the company has no plans t…         0.0
```

[ ]:
```
# merging the 4 dfs
df = pd.concat([df1, df2, df3, df4], ignore_index=True)
df.shape
```

[ ]:
```
(206705, 2)
```

[ ]:
```
# check null value
df.isnull().sum()
```

[ ]:
```
clean_text    104
category        7
dtype: int64
```

[ ]:
```
# drop null
cleandf = df.dropna(axis=0)
cleandf.shape
```

[ ]:
```
(206594, 2)
```

[ ]:
```
# check category unique value
cleandf.category.unique()
```

[ ]:
```
array([-1.,  0.,  1.])
```

[ ]:
```
# rename columns of cleaned df
cleandf = cleandf.rename(columns={'clean_text':'text','category': 'score'})
cleandf.head(1)
```

[ ]:
```
                                                text  score
0  when modi promised "minimum government maximum…   -1.0
```

[ ]:
```
# save to csv
cleandf.to_csv('data_model/merged_data.csv')
```

0.1.2 Text Cleaning

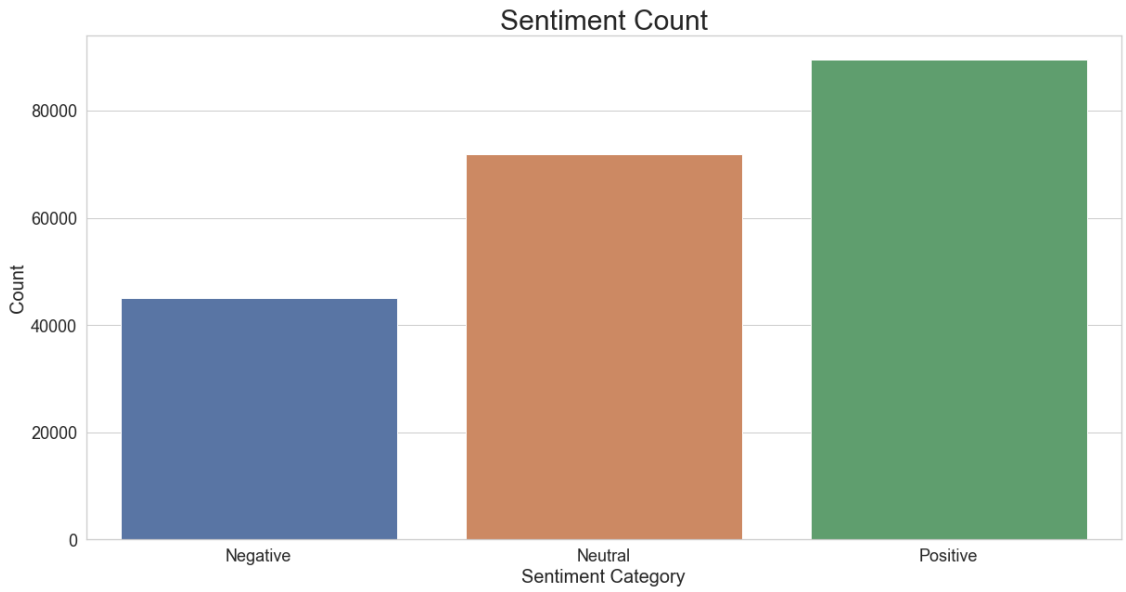
[ ]:
```
# load the csv
df=pd.read_csv('data_model/merged_data.csv',index_col=0)

# create category column
df['category'] = df['score'].map({-1.0:'Negative', 0.0:'Neutral', 1.0:
 ↪'Positive'})
df.head()
```

[ ]:
```
                                                text  score  category
0  when modi promised "minimum government maximum…   -1.0  Negative
1  talk all the nonsense and continue all the dra…    0.0   Neutral
2  what did just say vote for modi  welcome bjp t…    1.0  Positive
```

```
3  asking his supporters prefix chowkidar their n…    1.0  Positive
4  answer who among these the most powerful world…    1.0  Positive
```

[ ]:
```
plt.figure(figsize=(20,10))
# set theme
sns.set_theme(style="whitegrid")
#countplot ploarity
sns.countplot(x=df.category)

plt.xlabel('Sentiment Category', fontsize=20)
plt.ylabel('Count', fontsize = 20)
plt.yticks(fontsize=18)
plt.xticks(fontsize=18)
plt.title('Sentiment Count', fontsize=30)
```

[ ]:
```
Text(0.5, 1.0, 'Sentiment Count')
```



[ ]:
```
# assign new df
df2 = df
# filter text column using removeFunc - see function for more notes
df2['filter_text'] = df.text.apply(removeFunc)
df2.head()
```

[ ]:
```
                                              text  score  category  \
0  when modi promised "minimum government maximum…   -1.0  Negative
1  talk all the nonsense and continue all the dra…    0.0   Neutral
2  what did just say vote for modi  welcome bjp t…    1.0  Positive
3  asking his supporters prefix chowkidar their n…    1.0  Positive
```

```
4  answer who among these the most powerful world…    1.0  Positive

                                          filter_text
0  modi promised "minimum government maximum gove…
1             talk nonsense continue drama vote modi
2  say vote modi welcome bjp told rahul main camp…
3  asking supporters prefix chowkidar names modi …
4  answer among powerful world leader today trump…
```

[ ]:
```
# check na again
df2.isna().sum()
```

[ ]:
```
text           0
score          0
category       0
filter_text    0
dtype: int64
```

[ ]:
```
# assing split text column
df2['split_text'] = df2.filter_text.apply(tweet_to_words)
df2.isna().sum()
```

[ ]:
```
text           0
score          0
category       0
filter_text    0
split_text     0
dtype: int64
```

[ ]:
```
# save to csv
df2.to_csv('data_model/df.csv')
```

0.2 Deep Learning

0.2.1 Preproessing Text Vectorization

[ ]:
```
# load csv
df2= pd.read_csv('data_model/df.csv',index_col=0)
df2.head()
```

[ ]:
```
                                                text  score  category  \
0  when modi promised "minimum government maximum…   -1.0  Negative
1  talk all the nonsense and continue all the dra…    0.0   Neutral
2  what did just say vote for modi  welcome bjp t…    1.0  Positive
3  asking his supporters prefix chowkidar their n…    1.0  Positive
4  answer who among these the most powerful world…    1.0  Positive

                                          filter_text  \
```

```
0  modi promised "minimum government maximum gove…
1             talk nonsense continue drama vote modi
2  say vote modi welcome bjp told rahul main camp…
3  asking supporters prefix chowkidar names modi …
4  answer among powerful world leader today trump…

                                          split_text
0  ['modi', 'promis', 'minimum', 'govern', 'maxim…
1  ['talk', 'nonsens', 'continu', 'drama', 'vote'…
2  ['say', 'vote', 'modi', 'welcom', 'bjp', 'told…
3  ['ask', 'support', 'prefix', 'chowkidar', 'nam…
4  ['answer', 'among', 'power', 'world', 'leader'…
```

[ ]:
```
df2['len'] = df2.split_text.apply(len)

df2.head()
```

[ ]:
```
                                                text  score  category  \
0  when modi promised "minimum government maximum…   -1.0  Negative
1  talk all the nonsense and continue all the dra…    0.0   Neutral
2  what did just say vote for modi  welcome bjp t…    1.0  Positive
3  asking his supporters prefix chowkidar their n…    1.0  Positive
4  answer who among these the most powerful world…    1.0  Positive

                                         filter_text  \
0  modi promised "minimum government maximum gove…
1             talk nonsense continue drama vote modi
2  say vote modi welcome bjp told rahul main camp…
3  asking supporters prefix chowkidar names modi …
4  answer among powerful world leader today trump…

                                          split_text  len
0  [modi, promis, minimum, govern, maximum, gover…   22
1        [talk, nonsens, continu, drama, vote, modi]    6
2  [say, vote, modi, welcom, bjp, told, rahul, ma…   13
3  [ask, support, prefix, chowkidar, name, modi, …   20
4  [answer, among, power, world, leader, today, t…   10
```

[ ]:
```
# choose x_data
x_data = df2.split_text

# make y_data dummies
y_data = pd.get_dummies(df2.category)

#obtain max x length
max_x_len = get_max_length(x_data)
```

```
# embed dimension
EMBED_DIM = 32
max_x_len
```

[ ]: 16

[ ]:
```
# ENCODE REVIEW
token = Tokenizer(lower=False)    # False becuase we already did it.

# find formula to vectorize the text
token.fit_on_texts(x_data)

# converting text into vector
x_token = token.texts_to_sequences(x_data)

### this is the code to add the pad to x_train
x_token_pad = pad_sequences(x_token, maxlen=max_x_len, padding='post',␣
 ↪truncating='post')

total_words = len(token.word_index) + 1   # we need to add 1 because of 0␣
 ↪padding
```

[ ]:
```
# word counts of the tokenizer
token.word_counts
```

[ ]:
```
OrderedDict([('modi', 168391),
             ('promis', 4740),
             ('minimum', 638),
             ('govern', 9319),
             ('maximum', 284),
             ('expect', 2154),
             ('begin', 651),
             ('difficult', 505),
             ('job', 5229),
             ('reform', 517),
             ('state', 4804),
             ('take', 8312),
             ('year', 11682),
             ('get', 11836),
             ('justic', 549),
             ('not', 38231),
             ('busi', 2229),
             ('exit', 210),
             ('psu', 115),
             ('templ', 646),
             ('talk', 4521),
```

```
('nonsens', 486),
('continu', 1601),
('drama', 657),
('vote', 14589),
('say', 13879),
('welcom', 717),
('bjp', 20208),
('told', 1218),
('rahul', 7949),
('main', 1303),
('campaign', 3406),
('think', 8440),
('relax', 124),
('ask', 6292),
('support', 8982),
('prefix', 86),
('chowkidar', 4815),
('name', 3822),
('great', 4511),
('servic', 1017),
('confus', 450),
('read', 2872),
('crustal', 1),
('clear', 1408),
('crass', 51),
('filthi', 99),
('see', 7663),
('abus', 1300),
('come', 8406),
('answer', 2030),
('among', 756),
('power', 7738),
('world', 4908),
('leader', 6435),
('today', 4923),
('trump', 1277),
('putin', 124),
('may', 4346),
('kiya', 441),
('tho', 171),
('refresh', 46),
('maarkefir', 1),
('comment', 1822),
('karo', 236),
('surat', 44),
('women', 1342),
('perform', 942),
```

```
('yagna', 11),
('seek', 679),
('divin', 68),
('grace', 123),
('narendra', 11783),
('becom', 4654),
('cabinet', 480),
('scholar', 64),
('like', 18945),
('smriti', 231),
('hema', 102),
('time', 12190),
('introspect', 70),
('upcom', 479),
('elect', 14592),
('india', 36328),
('saga', 26),
('go', 4498),
('import', 2038),
('pair', 79),
('look', 4578),
('current', 1245),
('lead', 1295),
('govt', 11284),
('deal', 1332),
('brexit', 49),
('combin', 285),
('weekli', 52),
('juici', 9),
('bear', 229),
('imho', 35),
('gandhi', 6537),
('gay', 283),
('thing', 6279),
('demonetis', 917),
('gst', 892),
('good', 8720),
('tax', 2226),
('upper', 222),
('cast', 1106),
('would', 8374),
('sort', 387),
('either', 1091),
('view', 1138),
('favour', 634),
('need', 7652),
('give', 9581),
```

```
('dalit', 409),
('muslim', 4010),
('constituency2', 1),
('hope', 3049),
('tuthukudi', 1),
('peopl', 20093),
('prefer', 450),
('honest', 903),
('well', 5172),
('behav', 294),
('nationalist', 716),
('courag', 806),
('likli', 1),
('minist', 6742),
('benifit', 53),
('thuthukudi', 1),
('calm', 172),
('water', 711),
('where', 43),
('wave', 1178),
('one', 13827),
('make', 10312),
('differ', 3130),
('anil', 301),
('kapoor', 30),
('2019', 4509),
('clarion', 17),
('call', 5041),
('extend', 247),
('kar', 809),
('parti', 8708),
('leadershipwho', 1),
('fast', 406),
('firm', 256),
('action', 1689),
('none', 664),
('damodarda', 92),
('creat', 2684),
('ensur', 862),
('deserv', 1136),
('anupam', 38),
('kher', 46),
('respond', 443),
('appeal', 554),
('dont', 8816),
('play', 2060),
('word', 2040),
```

```
('swami', 491),
('relat', 987),
('guru', 194),
('protect', 833),
('mind', 1817),
('tweet', 2715),
('dark', 225),
('side', 1472),
('terror', 1577),
('brighter', 28),
('better', 4239),
('know', 9909),
('write', 1247),
('mean', 3349),
('anti', 4320),
('tri', 4552),
('visit', 1218),
('plz', 649),
('use', 6318),
('recent', 872),
('said', 7032),
('nation', 12129),
('put', 2367),
('gen', 105),
('hooda', 33),
('congress', 15151),
('jawan', 660),
('hear', 1151),
('belief', 207),
('leadership', 1541),
('shri', 1407),
('enter', 561),
('polit', 7559),
('given', 3129),
('form', 1198),
('file', 746),
('nomin', 449),
('khammam', 5),
('parliamentari', 149),
('seat', 3472),
('proceed', 71),
('crush', 125),
('jaw', 26),
('shoutmodimodi', 5),
('jd', 239),
('mla', 625),
('incit', 76),
```

```
('murder', 616),
('sultanpur', 7),
('uttar', 312),
('pradesh', 813),
('loksabha', 239),
('candid', 2177),
('select', 514),
('pawan', 29),
('kumar', 726),
('pandey', 41),
('actual', 2384),
('public', 3258),
('want', 10071),
('condid', 4),
('popular', 808),
('district', 299),
('bsp', 365),
('sonbhadra', 4),
('singh', 1611),
('thiugh', 1),
('nehru', 3634),
('aliv', 225),
('still', 4552),
('heart', 709),
('everi', 5590),
('failur', 1269),
('respons', 2237),
('develop', 4233),
('mass', 603),
('movement', 411),
('econom', 1746),
('social', 1507),
('empower', 134),
('life', 2212),
('wit', 439),
('posit', 1309),
('paradigm', 43),
('shift', 386),
('new', 5358),
('alreadi', 2686),
('taken', 1659),
('notic', 774),
('order', 1306),
('probe', 217),
('famili', 4066),
('harass', 153),
('beaten', 128),
```

```
('extremist', 219),
('hindu', 4135),
('suggest', 696),
('leav', 1211),
('move', 1517),
('pakistan', 6119),
('wait', 2366),
('also', 8913),
('varanasi', 1542),
('accord', 930),
('yogi', 840),
('imran', 843),
('masood', 313),
('kin', 31),
('azhar', 245),
('logic', 951),
('nirav', 3294),
('lalit', 212),
('brother', 814),
('mother', 661),
('agre', 1757),
('tenur', 573),
('modiganga', 1),
('rejuven', 23),
('work', 8236),
('start', 4423),
('three', 720),
('code', 1481),
('crack', 133),
('huge', 1335),
('foreign', 1104),
('polici', 2181),
('jumpstart', 8),
('via', 4348),
('slash', 23),
('educ', 2029),
('budget', 524),
('indic', 334),
('care', 1965),
('futur', 1624),
('presid', 1302),
('hand', 2102),
('increas', 1671),
('gdp', 594),
('born', 469),
('religion', 1251),
('femal', 100),
```

```
('deiti', 33),
('worship', 191),
('misogynist', 37),
('sadist', 34),
('tradit', 228),
('total', 1532),
('point', 2539),
('isit', 4),
('man', 3850),
('made', 4705),
('written', 472),
('religi', 591),
('lunat', 47),
('repress', 9),
('amazedn', 1),
('fear', 1489),
('frustat', 8),
('result', 1672),
('sir', 4767),
('wast', 841),
('ministerdisgrac', 1),
('entir', 1586),
('check', 1304),
('latest', 435),
('articl', 1334),
('premier', 40),
('archeri', 3),
('leagu', 317),
('second', 1306),
('optimist', 54),
('global', 673),
('execut', 397),
('growth', 1175),
('show', 4797),
('survey', 417),
('senior', 454),
('number', 1553),
('role', 669),
('wish', 1587),
('vision', 1247),
('least', 1579),
('interest', 1870),
('person', 4512),
('enmiti', 23),
('other', 1758),
('problem', 2467),
('handl', 619),
```

```
('join', 1921),
('dirti', 344),
('fight', 2522),
('tell', 3090),
('etern', 61),
('wrong', 2341),
('dear', 1548),
('sirji', 73),
('perfectli', 126),
('fine', 556),
('indian', 12015),
('impress', 331),
('godrej', 7),
('tata', 104),
('compliment', 39),
('term', 1982),
('maid', 57),
('keep', 3359),
('kalla', 4),
('yet', 1320),
('goe', 1675),
('hug', 258),
('wink', 24),
('magand', 1),
('idu', 1),
('bekagittu', 1),
('pleas', 4586),
('divid', 680),
('ye', 3033),
('highli', 331),
('insensitivearrog', 1),
('incompet', 320),
('ploar', 1),
('defeat', 1451),
('costnobodi', 1),
('arrog', 310),
('gave', 2258),
('ticketh', 1),
('touch', 416),
('grounddespit', 1),
('3month', 3),
('upsc', 16),
('protest', 501),
('nvr', 49),
('met', 378),
('2014', 4133),
('hindustan', 404),
```

```
('seen', 1918),
('worst', 743),
('maj', 5),
('rashtra', 102),
('thrash', 73),
('rascal', 55),
('face', 2251),
('politiciansantin', 1),
('urban', 336),
('naxal', 170),
('watch', 3459),
('win', 5773),
('mein', 765),
('bhi', 1950),
('hona', 54),
('garv', 13),
('baat', 334),
('hogi', 64),
('higher', 433),
('turnout', 34),
('directli', 539),
('proport', 89),
('victori', 717),
('wonder', 1405),
('launch', 1767),
('sit', 817),
('home', 1167),
('everyon', 2397),
('friend', 1874),
('rel', 301),
('never', 5229),
('done', 5633),
('remark', 372),
('corrupt', 4534),
('free', 2690),
('ultim', 216),
('success', 3439),
('shall', 355),
('achiev', 4396),
('jail', 1034),
('corruptionfre', 23),
('loot', 1438),
('countri', 11104),
('law', 1494),
('namo', 2618),
('app', 1637),
('beg', 371),
```

```
('welfar', 364),
('deliveri', 183),
('ibc', 47),
('feo', 1),
('place', 2039),
('2nd', 354),
('money', 6070),
('appoint', 297),
('judg', 661),
('polic', 838),
('forens', 24),
('lab', 27),
('fasttrack', 2),
('healthcar', 216),
('citizen', 1624),
('invest', 640),
('defenc', 933),
('build', 1393),
('live', 4901),
('deplor', 30),
('charact', 322),
('overpromis', 4),
('underdeliveri', 1),
('pithi', 3),
('summari', 82),
('outcom', 149),
('last', 4572),
('five', 1072),
('approach', 347),
('gener', 2052),
('heal', 38),
('surgeri', 52),
('remov', 1474),
('cancer', 155),
('spread', 1145),
('rss', 1770),
('farmer', 1884),
('474', 2),
('instal', 206),
('next', 3292),
('month', 1708),
('centr', 551),
('announc', 4588),
('75000crore', 2),
('scheme', 3342),
('mistri', 3),
('drag', 186),
```

```
('nri', 242),
('follow', 2962),
('hatr', 1119),
('agenda', 1355),
('condemn', 326),
('crimin', 631),
('activ', 758),
('fyi', 107),
('forgot', 329),
('dollar', 216),
('except', 985),
('diplomat', 187),
('shrewd', 33),
('alway', 3213),
('undermin', 146),
('prone', 18),
('criticis', 667),
('even', 9776),
('without', 2751),
('consid', 1026),
('aspect', 173),
('entrepreneur', 91),
('rise', 732),
('system', 1465),
('took', 1612),
('concern', 913),
('infra', 122),
('incub', 6),
('happen', 4551),
('guy', 3668),
('shit', 1666),
('simpl', 753),
('noth', 3396),
('els', 1809),
('phobia', 89),
('itna', 188),
('fark', 15),
('effort', 844),
('institutionalis', 12),
('honesti', 202),
('way', 4869),
('institut', 1189),
('design', 384),
('inculc', 17),
('inspir', 588),
('jai', 2331),
('hind', 1291),
```

```
('muje', 15),
('puri', 98),
('bharat', 1741),
('janta', 230),
('par', 383),
('vishwa', 32),
('hai', 5880),
('aap', 1774),
('hamr', 4),
('prime', 4922),
('hong', 63),
('must', 3224),
('anchor', 233),
('canva', 31),
('fit', 313),
('journal', 366),
('slam', 348),
('maker', 472),
('biopic', 973),
('deliber', 168),
('credit', 5856),
('channel', 1543),
('scare', 884),
('contest', 2395),
('two', 2364),
('propoganda', 128),
('littl', 853),
('decenc', 61),
('centuri', 279),
('yuva', 50),
('shakti', 2056),
('height', 310),
('stand', 2045),
('firmli', 90),
('100', 1458),
('sure', 3010),
('inform', 1069),
('record', 821),
('thank', 4492),
('loos', 688),
('exist', 949),
('rafel', 38),
('scam', 1923),
('natur', 488),
('scammer', 29),
('nautanki', 102),
('baj', 6),
```

```
('concierg', 1),
('super', 1126),
('rich', 845),
('unusu', 34),
('sight', 93),
('intellectu', 474),
('decid', 1327),
('question', 4830),
('similarli', 115),
('disagre', 218),
('anyth', 2946),
('learn', 1038),
('treat', 443),
('minor', 1225),
('cheat', 436),
('impact', 465),
('lot', 3041),
('economi', 2090),
('brotherhood', 23),
('could', 3371),
('throw', 514),
('light', 443),
('behind', 1266),
('bjpnda', 22),
('lose', 1772),
('bengaluru', 218),
('south', 1331),
('opposit', 4804),
('intent', 494),
('understand', 2977),
('fail', 2405),
('attempt', 486),
('oppon', 310),
('selfish', 133),
('idiot', 1082),
('attack', 2949),
('skill', 553),
('3040', 8),
('yr', 1386),
('oppo', 26),
('rule', 3213),
('stupid', 1321),
('dishonest', 65),
('pit', 60),
('pictur', 767),
('video', 2257),
('crime', 551),
```

```
('coz', 620),
('ppl', 1787),
('storm', 77),
('tweetstorm', 46),
('khan', 939),
('kill', 2294),
('paid', 948),
('loan', 1649),
('vijay', 823),
('maalya', 5),
('whose', 546),
('list', 1278),
('doesnt', 2089),
('fals', 797),
('deliv', 1024),
('whatev', 1033),
('track', 443),
('count', 756),
('kitna', 95),
('jalt', 3),
('tum', 267),
('jealousi', 49),
('toward', 1455),
('togeth', 845),
('tbfree', 22),
('2025', 51),
('ever', 2274),
('rgi', 2),
('length', 95),
('breadth', 13),
('recept', 33),
('love', 3255),
('match', 581),
('mad', 314),
('sandip', 36),
('sens', 1093),
('jave', 98),
('akhtar', 67),
('produc', 656),
('ssingh', 31),
('row', 153),
('entertain', 223),
('news', 4493),
('express', 936),
('porn', 80),
('site', 356),
('ban', 1312),
```

```
('eat', 726),
('beef', 464),
('biryani', 290),
('sleep', 514),
('asaduddin', 97),
('owaisi', 302),
('screw', 228),
('hate', 3545),
('ideolog', 663),
('propag', 128),
('week', 937),
('major', 2490),
('propaganda', 1072),
('media', 5101),
('fed', 175),
('terrorist', 2195),
('bullet', 259),
('bomb', 449),
('adityanath', 180),
('open', 1635),
('eye', 665),
('critic', 1700),
('cong', 1518),
('exculd', 1),
('monger', 215),
('vast', 83),
('indai', 8),
('secur', 2195),
('prais', 1053),
('worthi', 112),
('autocrat', 56),
('norm', 152),
('deni', 906),
('alleg', 596),
('kimjong', 3),
('pain', 396),
('sorrow', 14),
('regard', 619),
('request', 1037),
('stop', 3637),
('favor', 293),
('big', 3030),
('malya', 176),
('nivav', 1),
('nonscen', 1),
('charg', 667),
('haunt', 62),
```

```
('abduct', 37),
('girl', 664),
('wel', 21),
('messag', 1322),
('unlik', 672),
('bank', 2850),
('sharam', 38),
('karodesh', 1),
('ghotal', 4),
('bech', 32),
('khaega', 1),
('tumhara', 24),
('youth', 1311),
('issu', 3798),
('right', 5927),
('forget', 1350),
('petrol', 240),
('price', 1027),
('risen', 54),
('gulf', 43),
('thought', 1811),
('petta', 3),
('antibjp', 37),
('movi', 1899),
('blog', 162),
('sapna', 47),
('choudhari', 24),
('yesterday', 653),
('manoj', 65),
('tiwari', 51),
('superbl', 8),
('summarizedjai', 1),
('vand', 164),
('mataramagain', 1),
('sarkar', 1528),
('prosper', 354),
('forefront', 25),
('endeavour', 55),
('focuss', 62),
('enabl', 226),
('empow', 336),
('sabbash', 2),
('mera', 159),
('peppermit', 1),
('abvp', 58),
('hold', 1057),
('ralli', 2244),
```

```
('amit', 1477),
('shah', 2720),
('beginningmodi', 1),
('wada', 14),
('faramoshi', 1),
('believ', 2984),
('exampl', 823),
('lakh', 3075),
('fulfil', 589),
('modiji', 1435),
('enemi', 688),
('destroy', 2027),
('camp', 346),
('hatedont', 1),
('defam', 230),
('human', 894),
('kind', 1531),
('pure', 501),
('limit', 573),
('twitter', 1228),
('perhap', 315),
('accur', 93),
('captur', 271),
('dig', 223),
('saharanpur', 32),
('soninlaw', 18),
('speak', 2474),
('languag', 902),
('prot', 4),
('g', 13),
('tread', 20),
('path', 285),
('compani', 1921),
('head', 1288),
('chowkidhar', 106),
('tsunami', 119),
('frenzi', 34),
('duti', 394),
('mallaya', 106),
('wouldnt', 254),
('lootedsonia', 1),
('bcoz', 504),
('mm', 555),
('weak', 491),
('blatant', 99),
('misus', 289),
('rampant', 68),
```

```
('misgovern', 24),
('overt', 17),
('lie', 2380),
('damag', 450),
('mob', 303),
('worri', 1422),
('send', 1055),
('bar', 620),
('dream', 959),
('looter', 221),
('includ', 1388),
('kejriw', 820),
('paper', 615),
('irani', 253),
('career', 279),
('rememb', 1641),
('feed', 299),
('left', 1819),
('old', 1609),
('wife', 565),
('return', 1079),
('walay', 6),
('sab', 533),
('hain', 818),
('shi', 81),
('kha', 52),
('yadav', 334),
('mere', 388),
('bhai', 968),
('mai', 321),
('2010', 393),
('graduat', 161),
('hucongress', 1),
('kon', 42),
('rojgaar', 9),
('faila', 8),
('rakha', 37),
('thabhai', 1),
('chor', 1861),
('strong', 1928),
('came', 1701),
('ago', 827),
('takenin', 1),
('jobsyr', 9),
('forward', 784),
('present', 998),
('lost', 1683),
```

```
('45yrhigh', 7),
('unemploy', 1262),
('whoa', 28),
('actor', 671),
('statement', 1594),
('band', 153),
('fake', 2431),
('bloodthirsti', 10),
('taliban', 72),
('mindset', 261),
('brahminnon', 1),
('brahmin', 338),
('baad', 111),
('karna', 155),
('chahiyeour', 1),
('wise', 415),
('antihindu', 48),
('persecut', 89),
('anyway', 682),
('varda', 4),
('donnot', 2),
('focu', 576),
('unrequir', 1),
('kinda', 135),
('uncomfort', 57),
('unexpect', 51),
('idk', 57),
('shown', 640),
('enough', 1863),
('solidar', 60),
('back', 5740),
('normal', 345),
('someon', 1959),
('offer', 867),
('lol', 1294),
('funni', 452),
('dow', 8),
('plane', 351),
('endian', 21),
('air', 1926),
('chief', 1755),
('replac', 449),
('detect', 87),
('sub', 396),
('near', 454),
('pakistani', 1074),
('admir', 242),
```

```
('mirror', 113),
('agent', 296),
('adnan', 3),
('sami', 14),
('endia', 42),
('within', 934),
('versu', 99),
('swarajya', 14),
('idea', 1570),
('gentleman', 69),
('convict', 246),
('clariti', 62),
('trust', 1230),
('anyon', 2007),
('cover', 679),
('blood', 343),
('solut', 554),
('first', 4649),
('hey', 644),
('pledg', 291),
('link', 1201),
('task', 184),
('independ', 957),
('communist', 257),
('assert', 122),
('abandon', 112),
('liquid', 58),
('japan', 162),
('singapor', 57),
('usa', 484),
('tdp', 148),
('spend', 701),
('moneyexp', 1),
('detail', 682),
('experi', 508),
('pension', 160),
('dwacra', 2),
('yuvanestam', 1),
('song', 418),
('sung', 17),
('hurrah', 5),
('chinook', 21),
('multi', 51),
('helicopt', 163),
('realli', 3504),
('proud', 2633),
('step', 1072),
```

```
('urg', 357),
('walk', 398),
('disciplin', 88),
('fauji', 27),
('fli', 370),
('tiranga', 14),
('planet', 111),
('kunal', 50),
('kamra', 49),
('pretenti', 10),
('chutiya', 339),
('braincel', 2),
('sinc', 2972),
('birth', 225),
('smoke', 141),
('weed', 91),
('cow', 689),
('graffiti', 13),
('mock', 497),
('appear', 473),
('find', 1856),
('bengal', 448),
('shabana', 25),
('azmi', 25),
('mislead', 242),
('priminist', 49),
('halala', 17),
('whr', 38),
('whole', 1747),
('fuck', 2206),
('poor', 4244),
('woman', 405),
('den', 103),
('physic', 171),
('mental', 506),
('handicap', 19),
('kid', 714),
('due', 2062),
('genet', 24),
…])
```

[ ]:
```
# train test split
X_train, X_test, y_train, y_test = train_test_split(x_token_pad, y_data,␣
↪test_size=0.25, random_state=42)
```

[ ]:
```
X_train.shape
```

[ ]: (154945, 16)

[ ]:
```
y_train.shape
```

[ ]: (154945, 3)

[ ]:
```
# saving to pickle file
with open('output/token_model1.pickle', 'wb') as handle:
    pickle.dump(token, handle, protocol=pickle.HIGHEST_PROTOCOL)

# loading from pickle file
with open('output/token_model1.pickle', 'rb') as handle:
    token = pickle.load(handle)
```

0.2.2 Embedding with LSTM

[ ]:
```
#### add the model here:
lstm = Sequential()
# Embedding - common application is for text processing, changing integer into␣
 ↪vectors
lstm.add(Embedding(total_words,EMBED_DIM,
                   input_length=max_x_len))
# lstm.add(Dropout(0.4))
lstm.add(LSTM(units=32, dropout=0.3, recurrent_dropout=0.3))
# lstm.add(Dropout(0.2))
# lstm.add(Dense(27, activation='sigmoid'))
# lstm.add(Dropout(0.2))
# lstm.add(Dense(9, activation='sigmoid'))
# lstm.add(Dropout(0.2))
lstm.add(Dense(3, activation='softmax'))

# show a graph of model
tf.keras.utils.plot_model(lstm, show_shapes=True)

### compile the model using: optimizer = 'adam', loss = 'binary_crossentropy',␣
 ↪metrics = ['accuracy']
lstm.compile(optimizer='SGD', loss='binary_crossentropy', metrics=['accuracy',␣
 ↪Precision(), Recall()])
lstm.summary()
# Save model checkpoint and save best only
checkpoint = ModelCheckpoint('output/LSTM.h5', monitor='accuracy',␣
 ↪save_best_only=True, verbose=1)
# fit and train model, call back based on checkpoint(best model)
history = lstm.fit(X_train, y_train, epochs=10, batch_size=32,␣
 ↪validation_data=(X_test, y_test), callbacks=[checkpoint])
```

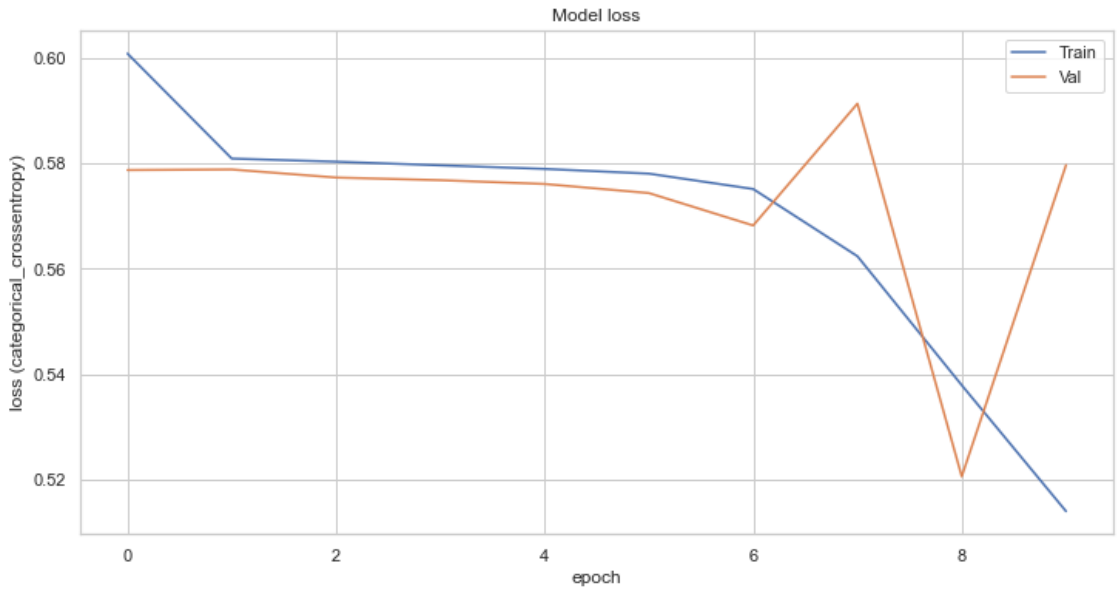
```
You must install pydot (`pip install pydot`) and install graphviz (see
instructions at https://graphviz.gitlab.io/download/) for
```

```
plot_model/model_to_dot to work.
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_3 (Embedding)     (None, 16, 32)            3611296

 dropout_3 (Dropout)         (None, 16, 32)            0

 lstm_3 (LSTM)               (None, 32)                8320

 dense_3 (Dense)             (None, 3)                 99

=================================================================
Total params: 3,619,715
Trainable params: 3,619,715
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
4841/4843 [============================>.] - ETA: 0s - loss: 0.6008 - accuracy:
0.4899 - precision_3: 0.5496 - recall_3: 0.2421
Epoch 1: accuracy improved from -inf to 0.48992, saving model to output\LSTM.h5
4843/4843 [==============================] - 56s 11ms/step - loss: 0.6008 -
accuracy: 0.4899 - precision_3: 0.5496 - recall_3: 0.2422 - val_loss: 0.5787 -
val_accuracy: 0.5309 - val_precision_3: 0.5438 - val_recall_3: 0.4907
Epoch 2/10
4841/4843 [============================>.] - ETA: 0s - loss: 0.5809 - accuracy:
0.5276 - precision_3: 0.5406 - recall_3: 0.4841
Epoch 2: accuracy improved from 0.48992 to 0.52764, saving model to
output\LSTM.h5
4843/4843 [==============================] - 52s 11ms/step - loss: 0.5809 -
accuracy: 0.5276 - precision_3: 0.5406 - recall_3: 0.4841 - val_loss: 0.5788 -
val_accuracy: 0.5291 - val_precision_3: 0.5414 - val_recall_3: 0.4935
Epoch 3/10
4839/4843 [============================>.] - ETA: 0s - loss: 0.5803 - accuracy:
0.5275 - precision_3: 0.5416 - recall_3: 0.4833
Epoch 3: accuracy did not improve from 0.52764
4843/4843 [==============================] - 51s 10ms/step - loss: 0.5803 -
accuracy: 0.5275 - precision_3: 0.5416 - recall_3: 0.4833 - val_loss: 0.5773 -
val_accuracy: 0.5314 - val_precision_3: 0.5436 - val_recall_3: 0.4911
Epoch 4/10
4843/4843 [==============================] - ETA: 0s - loss: 0.5796 - accuracy:
0.5292 - precision_3: 0.5428 - recall_3: 0.4829
Epoch 4: accuracy improved from 0.52764 to 0.52919, saving model to
output\LSTM.h5
4843/4843 [==============================] - 51s 10ms/step - loss: 0.5796 -
accuracy: 0.5292 - precision_3: 0.5428 - recall_3: 0.4829 - val_loss: 0.5768 -
val_accuracy: 0.5344 - val_precision_3: 0.5495 - val_recall_3: 0.4817
```

```
Epoch 5/10
4843/4843 [==============================] - ETA: 0s - loss: 0.5790 - accuracy:
0.5290 - precision_3: 0.5431 - recall_3: 0.4827
Epoch 5: accuracy did not improve from 0.52919
4843/4843 [==============================] - 54s 11ms/step - loss: 0.5790 -
accuracy: 0.5290 - precision_3: 0.5431 - recall_3: 0.4827 - val_loss: 0.5761 -
val_accuracy: 0.5321 - val_precision_3: 0.5464 - val_recall_3: 0.4893
Epoch 6/10
4839/4843 [============================>.] - ETA: 0s - loss: 0.5781 - accuracy:
0.5300 - precision_3: 0.5448 - recall_3: 0.4836
Epoch 6: accuracy improved from 0.52919 to 0.53001, saving model to
output\LSTM.h5
4843/4843 [==============================] - 57s 12ms/step - loss: 0.5781 -
accuracy: 0.5300 - precision_3: 0.5448 - recall_3: 0.4836 - val_loss: 0.5744 -
val_accuracy: 0.5363 - val_precision_3: 0.5500 - val_recall_3: 0.4884
Epoch 7/10
4839/4843 [============================>.] - ETA: 0s - loss: 0.5752 - accuracy:
0.5347 - precision_3: 0.5502 - recall_3: 0.4878
Epoch 7: accuracy improved from 0.53001 to 0.53470, saving model to
output\LSTM.h5
4843/4843 [==============================] - 59s 12ms/step - loss: 0.5751 -
accuracy: 0.5347 - precision_3: 0.5502 - recall_3: 0.4878 - val_loss: 0.5682 -
val_accuracy: 0.5437 - val_precision_3: 0.5573 - val_recall_3: 0.5019
Epoch 8/10
4840/4843 [============================>.] - ETA: 0s - loss: 0.5624 - accuracy:
0.5548 - precision_3: 0.5763 - recall_3: 0.4995
Epoch 8: accuracy improved from 0.53470 to 0.55484, saving model to
output\LSTM.h5
4843/4843 [==============================] - 63s 13ms/step - loss: 0.5624 -
accuracy: 0.5548 - precision_3: 0.5763 - recall_3: 0.4995 - val_loss: 0.5913 -
val_accuracy: 0.5231 - val_precision_3: 0.5295 - val_recall_3: 0.5042
Epoch 9/10
4842/4843 [============================>.] - ETA: 0s - loss: 0.5380 - accuracy:
0.5857 - precision_3: 0.6164 - recall_3: 0.5284
Epoch 9: accuracy improved from 0.55484 to 0.58574, saving model to
output\LSTM.h5
4843/4843 [==============================] - 51s 11ms/step - loss: 0.5380 -
accuracy: 0.5857 - precision_3: 0.6164 - recall_3: 0.5284 - val_loss: 0.5207 -
val_accuracy: 0.5987 - val_precision_3: 0.6150 - val_recall_3: 0.5668
Epoch 10/10
4843/4843 [==============================] - ETA: 0s - loss: 0.5141 - accuracy:
0.6097 - precision_3: 0.6502 - recall_3: 0.5502
Epoch 10: accuracy improved from 0.58574 to 0.60969, saving model to
output\LSTM.h5
4843/4843 [==============================] - 51s 10ms/step - loss: 0.5141 -
accuracy: 0.6097 - precision_3: 0.6502 - recall_3: 0.5502 - val_loss: 0.5797 -
val_accuracy: 0.5109 - val_precision_3: 0.5171 - val_recall_3: 0.4864
```

[ ]:
```
plt.figure(figsize=(12,6))
plt.plot(lstm.history.history['loss'][:])
plt.plot(lstm.history.history['val_loss'][:])
plt.title('Model loss')
plt.xlabel('epoch')
plt.ylabel('loss (categorical_crossentropy)')
plt.legend(['Train', 'Val'], loc='upper right')
```

[ ]:
```
<matplotlib.legend.Legend at 0x2776f9a4580>
```



0.2.3 Embedding, Conv1D with Bidirectional LSTM

[ ]:
```
#### add the model here:
epochs=20
learning_rate = 0.1
decay_rate = learning_rate / epochs
momentum = 0.8

sgd = SGD(lr=learning_rate, momentum=momentum, decay=decay_rate, nesterov=False)

model = Sequential()
model.add(Embedding(total_words,EMBED_DIM, input_length=max_x_len))
# Conv1D is for regression, Conv2D is for images, Conv3D
model.add(Conv1D(filters=32, kernel_size=3, padding='same',activation='relu'))
# model.add(SpatialDropout1D(0.3))
model.add(MaxPooling1D(pool_size=2))
```

```
# since maxpooling returns 3dimension, we dont need reshape, if we connect to␣
 ↪regular NN, we should flatten to change shape into 2D
# model.add(Flatten)
model.add(Bidirectional(LSTM(32)))
model.add(Dropout(0.5))
# model.add(Dense(16, activation='sigmoid'))
# model.add(Dropout(0.3))
# dense 3 output is based on y dimension, this y dimension is (_, 3)
model.add(Dense(3, activation='softmax'))
# model.add(Dropout(0.2))
# model.add(Dense(1, activation='softmax'))

# show a graph of model
tf.keras.utils.plot_model(model, show_shapes=True)

### compile the model using: optimizer = 'adam', loss = 'binary_crossentropy',␣
 ↪metrics = ['accuracy']
model.compile(optimizer=sgd, loss='binary_crossentropy', metrics=['accuracy',␣
 ↪Precision(), Recall()])
model.summary()
```

```
You must install pydot (`pip install pydot`) and install graphviz (see
instructions at https://graphviz.gitlab.io/download/) for
plot_model/model_to_dot to work.
Model: "sequential_4"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_4 (Embedding)     (None, 16, 32)            3611296

 conv1d (Conv1D)             (None, 16, 32)            3104

 max_pooling1d (MaxPooling1D  (None, 8, 32)            0
 )

 bidirectional (Bidirectiona  (None, 64)               16640
 l)

 dropout_4 (Dropout)         (None, 64)                0

 dense_4 (Dense)             (None, 3)                 195

=================================================================
Total params: 3,631,235
Trainable params: 3,631,235
Non-trainable params: 0
_________________________________________________________________
```

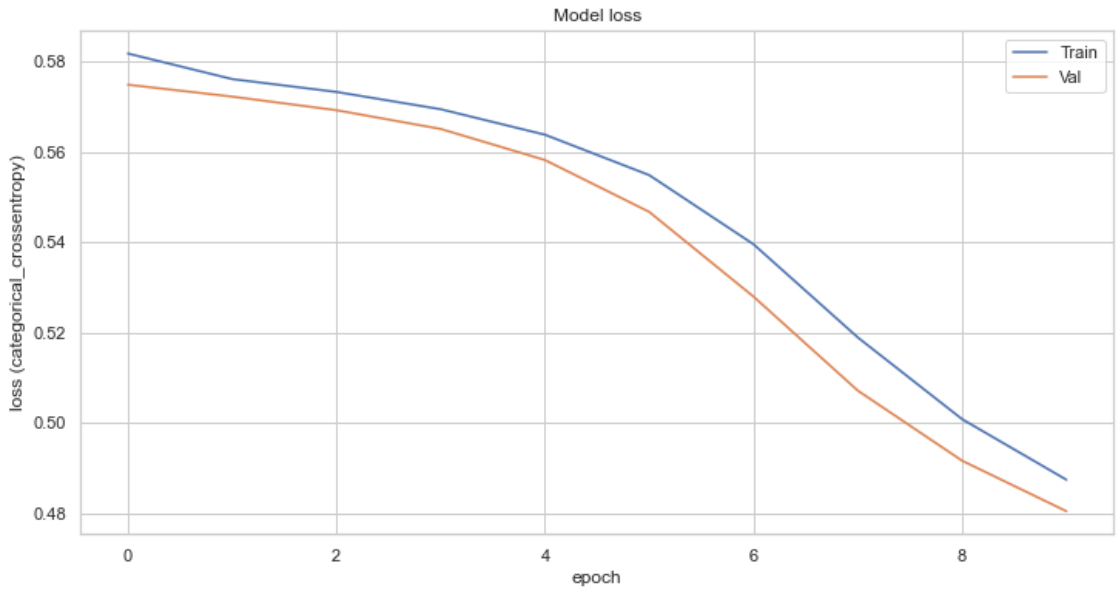
[ ]:
```
# tensor board log
log_dir = "logs/fit/" + dt.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,␣
 ↪histogram_freq=1)

# Save model checkpoint and save best only
checkpoint = ModelCheckpoint('output/convBiLSTM.h5', monitor='accuracy',␣
 ↪save_best_only=True, verbose=1)
# fit and train model, call back based on checkpoint(best model)
history = model.fit(X_train, y_train, epochs=10, batch_size=64,␣
 ↪validation_data=(X_test, y_test),␣
 ↪callbacks=[checkpoint,tensorboard_callback])
```

```
Epoch 1/10
2418/2422 [============================>.] - ETA: 0s - loss: 0.5818 - accuracy:
0.5266 - precision_4: 0.5472 - recall_4: 0.4572
Epoch 1: accuracy improved from -inf to 0.52651, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 25s 8ms/step - loss: 0.5818 -
accuracy: 0.5265 - precision_4: 0.5472 - recall_4: 0.4572 - val_loss: 0.5749 -
val_accuracy: 0.5360 - val_precision_4: 0.5510 - val_recall_4: 0.4734
Epoch 2/10
2422/2422 [==============================] - ETA: 0s - loss: 0.5762 - accuracy:
0.5335 - precision_4: 0.5519 - recall_4: 0.4769
Epoch 2: accuracy improved from 0.52651 to 0.53349, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 19s 8ms/step - loss: 0.5762 -
accuracy: 0.5335 - precision_4: 0.5519 - recall_4: 0.4769 - val_loss: 0.5723 -
val_accuracy: 0.5377 - val_precision_4: 0.5566 - val_recall_4: 0.4782
Epoch 3/10
2415/2422 [============================>.] - ETA: 0s - loss: 0.5733 - accuracy:
0.5367 - precision_4: 0.5560 - recall_4: 0.4808
Epoch 3: accuracy improved from 0.53349 to 0.53676, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.5733 -
accuracy: 0.5368 - precision_4: 0.5560 - recall_4: 0.4808 - val_loss: 0.5693 -
val_accuracy: 0.5419 - val_precision_4: 0.5609 - val_recall_4: 0.4840
Epoch 4/10
2422/2422 [==============================] - ETA: 0s - loss: 0.5695 - accuracy:
0.5423 - precision_4: 0.5620 - recall_4: 0.4863
Epoch 4: accuracy improved from 0.53676 to 0.54229, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5695 -
accuracy: 0.5423 - precision_4: 0.5620 - recall_4: 0.4863 - val_loss: 0.5651 -
val_accuracy: 0.5478 - val_precision_4: 0.5682 - val_recall_4: 0.4904
Epoch 5/10
2415/2422 [============================>.] - ETA: 0s - loss: 0.5639 - accuracy:
0.5499 - precision_4: 0.5708 - recall_4: 0.4941
```

```
Epoch 5: accuracy improved from 0.54229 to 0.54999, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.5638 -
accuracy: 0.5500 - precision_4: 0.5709 - recall_4: 0.4942 - val_loss: 0.5582 -
val_accuracy: 0.5577 - val_precision_4: 0.5782 - val_recall_4: 0.5036
Epoch 6/10
2420/2422 [============================>.] - ETA: 0s - loss: 0.5549 - accuracy:
0.5635 - precision_4: 0.5865 - recall_4: 0.5074
Epoch 6: accuracy improved from 0.54999 to 0.56354, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.5549 -
accuracy: 0.5635 - precision_4: 0.5866 - recall_4: 0.5074 - val_loss: 0.5467 -
val_accuracy: 0.5768 - val_precision_4: 0.6007 - val_recall_4: 0.5196
Epoch 7/10
2417/2422 [============================>.] - ETA: 0s - loss: 0.5396 - accuracy:
0.5876 - precision_4: 0.6167 - recall_4: 0.5279
Epoch 7: accuracy improved from 0.56354 to 0.58768, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 19s 8ms/step - loss: 0.5396 -
accuracy: 0.5877 - precision_4: 0.6167 - recall_4: 0.5278 - val_loss: 0.5280 -
val_accuracy: 0.6031 - val_precision_4: 0.6350 - val_recall_4: 0.5450
Epoch 8/10
2416/2422 [============================>.] - ETA: 0s - loss: 0.5190 - accuracy:
0.6155 - precision_4: 0.6515 - recall_4: 0.5587
Epoch 8: accuracy improved from 0.58768 to 0.61557, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5189 -
accuracy: 0.6156 - precision_4: 0.6516 - recall_4: 0.5588 - val_loss: 0.5072 -
val_accuracy: 0.6249 - val_precision_4: 0.6611 - val_recall_4: 0.5728
Epoch 9/10
2416/2422 [============================>.] - ETA: 0s - loss: 0.5009 - accuracy:
0.6341 - precision_4: 0.6711 - recall_4: 0.5829
Epoch 9: accuracy improved from 0.61557 to 0.63415, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5008 -
accuracy: 0.6342 - precision_4: 0.6712 - recall_4: 0.5830 - val_loss: 0.4916 -
val_accuracy: 0.6393 - val_precision_4: 0.6758 - val_recall_4: 0.5915
Epoch 10/10
2417/2422 [============================>.] - ETA: 0s - loss: 0.4874 - accuracy:
0.6461 - precision_4: 0.6849 - recall_4: 0.5996
Epoch 10: accuracy improved from 0.63415 to 0.64603, saving model to
output\convBiLSTM.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.4874 -
accuracy: 0.6460 - precision_4: 0.6848 - recall_4: 0.5996 - val_loss: 0.4804 -
val_accuracy: 0.6485 - val_precision_4: 0.6865 - val_recall_4: 0.6031
```

[ ]:
```
plt.figure(figsize=(12,6))
plt.plot(model.history.history['loss'][:])
plt.plot(model.history.history['val_loss'][:])
plt.title('Model loss')
plt.xlabel('epoch')
plt.ylabel('loss (categorical_crossentropy)')
plt.legend(['Train', 'Val'], loc='upper right')
```

[ ]:
```
<matplotlib.legend.Legend at 0x2777567d430>
```



0.2.4 GRU

[ ]:
```
gru = Sequential()
gru.add(Embedding(total_words,EMBED_DIM, input_length=max_x_len))
gru.add(GRU(32))
gru.add(Dropout(0.4))
# time took too long
# gru.add(Dense(9))
# gru.add(Dropout(0.25))
gru.add(Dense(3, activation='softmax'))

gru.compile(loss='binary_crossentropy', optimizer='SGD',metrics=['accuracy',␣
 ↪Precision(), Recall()])

gru.summary()
```

```
Model: "sequential_5"
_________________________________________________________________
```

```
 Layer (type)                 Output Shape              Param #
=================================================================
 embedding_5 (Embedding)      (None, 16, 32)            3611296

 gru (GRU)                    (None, 32)                6336

 dropout_5 (Dropout)          (None, 32)                0

 dense_5 (Dense)              (None, 3)                 99

=================================================================
Total params: 3,617,731
Trainable params: 3,617,731
Non-trainable params: 0
_________________________________________________________________
```

[ ]:
```
X_train.shape
```

[ ]:
```
(154945, 16)
```

[ ]:
```
# tensor board log
# log_dir = "logs/fit/" + dt.datetime.now().strftime("%Y%m%d-%H%M%S")
# tensorboard_callback = tf.keras.callbacks.TensorBoard(log_dir=log_dir,␣
 ↪histogram_freq=1)

# Save model checkpoint and save best only
checkpoint = ModelCheckpoint('output/gru.h5', monitor='accuracy',␣
 ↪save_best_only=True, verbose=1)
# fit and train model, call back based on checkpoint(best model)
history = gru.fit(X_train, y_train, epochs=10, batch_size=64,␣
 ↪validation_data=(X_test, y_test), callbacks=[checkpoint], verbose=1)
```

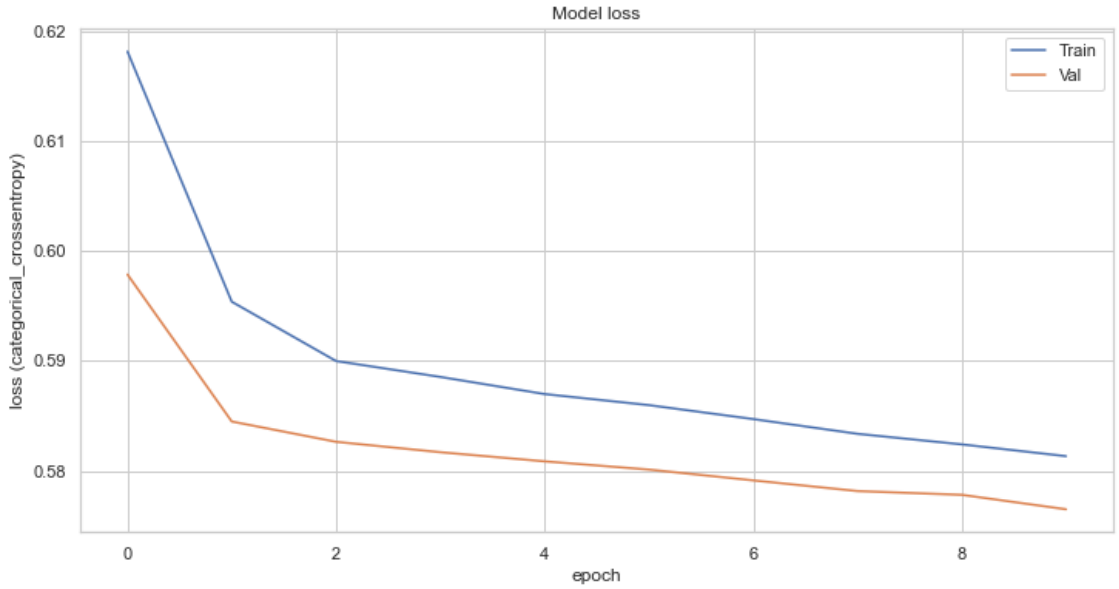
```
Epoch 1/10
2418/2422 [============================>.] - ETA: 0s - loss: 0.6182 - accuracy:
0.4591 - precision_5: 0.4912 - recall_5: 0.1212
Epoch 1: accuracy improved from -inf to 0.45910, saving model to output\gru.h5
2422/2422 [==============================] - 20s 7ms/step - loss: 0.6182 -
accuracy: 0.4591 - precision_5: 0.4912 - recall_5: 0.1214 - val_loss: 0.5979 -
val_accuracy: 0.5289 - val_precision_5: 0.5553 - val_recall_5: 0.2242
Epoch 2/10
2419/2422 [============================>.] - ETA: 0s - loss: 0.5954 - accuracy:
0.5098 - precision_5: 0.5371 - recall_5: 0.3590
Epoch 2: accuracy improved from 0.45910 to 0.50978, saving model to
output\gru.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5954 -
accuracy: 0.5098 - precision_5: 0.5371 - recall_5: 0.3590 - val_loss: 0.5845 -
val_accuracy: 0.5257 - val_precision_5: 0.5471 - val_recall_5: 0.4591
Epoch 3/10
```

```
2418/2422 [============================>.] - ETA: 0s - loss: 0.5900 - accuracy:
0.5203 - precision_5: 0.5375 - recall_5: 0.4325
Epoch 3: accuracy improved from 0.50978 to 0.52029, saving model to
output\gru.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5900 -
accuracy: 0.5203 - precision_5: 0.5375 - recall_5: 0.4325 - val_loss: 0.5827 -
val_accuracy: 0.5257 - val_precision_5: 0.5377 - val_recall_5: 0.4830
Epoch 4/10
2416/2422 [============================>.] - ETA: 0s - loss: 0.5886 - accuracy:
0.5221 - precision_5: 0.5387 - recall_5: 0.4477
Epoch 4: accuracy improved from 0.52029 to 0.52215, saving model to
output\gru.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.5886 -
accuracy: 0.5222 - precision_5: 0.5387 - recall_5: 0.4477 - val_loss: 0.5817 -
val_accuracy: 0.5257 - val_precision_5: 0.5357 - val_recall_5: 0.4923
Epoch 5/10
2422/2422 [==============================] - ETA: 0s - loss: 0.5870 - accuracy:
0.5244 - precision_5: 0.5404 - recall_5: 0.4565
Epoch 5: accuracy improved from 0.52215 to 0.52443, saving model to
output\gru.h5
2422/2422 [==============================] - 16s 7ms/step - loss: 0.5870 -
accuracy: 0.5244 - precision_5: 0.5404 - recall_5: 0.4565 - val_loss: 0.5809 -
val_accuracy: 0.5257 - val_precision_5: 0.5443 - val_recall_5: 0.4744
Epoch 6/10
2419/2422 [============================>.] - ETA: 0s - loss: 0.5860 - accuracy:
0.5248 - precision_5: 0.5418 - recall_5: 0.4612
Epoch 6: accuracy improved from 0.52443 to 0.52476, saving model to
output\gru.h5
2422/2422 [==============================] - 16s 7ms/step - loss: 0.5860 -
accuracy: 0.5248 - precision_5: 0.5419 - recall_5: 0.4612 - val_loss: 0.5801 -
val_accuracy: 0.5290 - val_precision_5: 0.5443 - val_recall_5: 0.4745
Epoch 7/10
2414/2422 [============================>.] - ETA: 0s - loss: 0.5848 - accuracy:
0.5259 - precision_5: 0.5434 - recall_5: 0.4633
Epoch 7: accuracy improved from 0.52476 to 0.52602, saving model to
output\gru.h5
2422/2422 [==============================] - 16s 7ms/step - loss: 0.5847 -
accuracy: 0.5260 - precision_5: 0.5435 - recall_5: 0.4634 - val_loss: 0.5791 -
val_accuracy: 0.5290 - val_precision_5: 0.5443 - val_recall_5: 0.4745
Epoch 8/10
2416/2422 [============================>.] - ETA: 0s - loss: 0.5834 - accuracy:
0.5275 - precision_5: 0.5450 - recall_5: 0.4653
Epoch 8: accuracy improved from 0.52602 to 0.52749, saving model to
output\gru.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5834 -
accuracy: 0.5275 - precision_5: 0.5450 - recall_5: 0.4652 - val_loss: 0.5782 -
val_accuracy: 0.5290 - val_precision_5: 0.5448 - val_recall_5: 0.4897
Epoch 9/10
```

```
2418/2422 [============================>.] - ETA: 0s - loss: 0.5824 - accuracy:
0.5285 - precision_5: 0.5467 - recall_5: 0.4666
Epoch 9: accuracy improved from 0.52749 to 0.52850, saving model to
output\gru.h5
2422/2422 [==============================] - 18s 7ms/step - loss: 0.5824 -
accuracy: 0.5285 - precision_5: 0.5466 - recall_5: 0.4666 - val_loss: 0.5778 -
val_accuracy: 0.5291 - val_precision_5: 0.5445 - val_recall_5: 0.4767
Epoch 10/10
2413/2422 [============================>.] - ETA: 0s - loss: 0.5813 - accuracy:
0.5294 - precision_5: 0.5486 - recall_5: 0.4670
Epoch 10: accuracy improved from 0.52850 to 0.52928, saving model to
output\gru.h5
2422/2422 [==============================] - 17s 7ms/step - loss: 0.5813 -
accuracy: 0.5293 - precision_5: 0.5486 - recall_5: 0.4670 - val_loss: 0.5765 -
val_accuracy: 0.5327 - val_precision_5: 0.5494 - val_recall_5: 0.4738
```

[ ]:
```
plt.figure(figsize=(12,6))
plt.plot(gru.history.history['loss'][:])
plt.plot(gru.history.history['val_loss'][:])
plt.title('Model loss')
plt.xlabel('epoch')
plt.ylabel('loss (categorical_crossentropy)')
plt.legend(['Train', 'Val'], loc='upper right')
```

[ ]:
```
<matplotlib.legend.Legend at 0x27770599730>
```



0.2.5 GRU TF

[ ]:
```
inputs = Input(shape=(max_x_len))
x = Embedding(total_words,EMBED_DIM,
                  input_length=max_x_len)(inputs)
x = GRU(32, recurrent_dropout=0.5, return_sequences=True)(x)
x = GRU(32, recurrent_dropout=0.5)(x)
x = Dropout(0.5)(x)
outputs = Dense(3,activation = 'softmax')(x)
model = Model(inputs, outputs)
model.compile(loss='binary_crossentropy', optimizer='adam',metrics=['accuracy',␣
 ↪Precision(), Recall()])
model.summary()


# Save model checkpoint and save best only
checkpoint = ModelCheckpoint('output/gruTF.h5', monitor='accuracy',␣
 ↪save_best_only=True, verbose=1)
# fit and train model, call back based on checkpoint(best model)
history = model.fit(X_train, y_train, epochs=10, batch_size=32,␣
 ↪validation_data=(X_test, y_test), callbacks=[checkpoint], verbose=1)
```

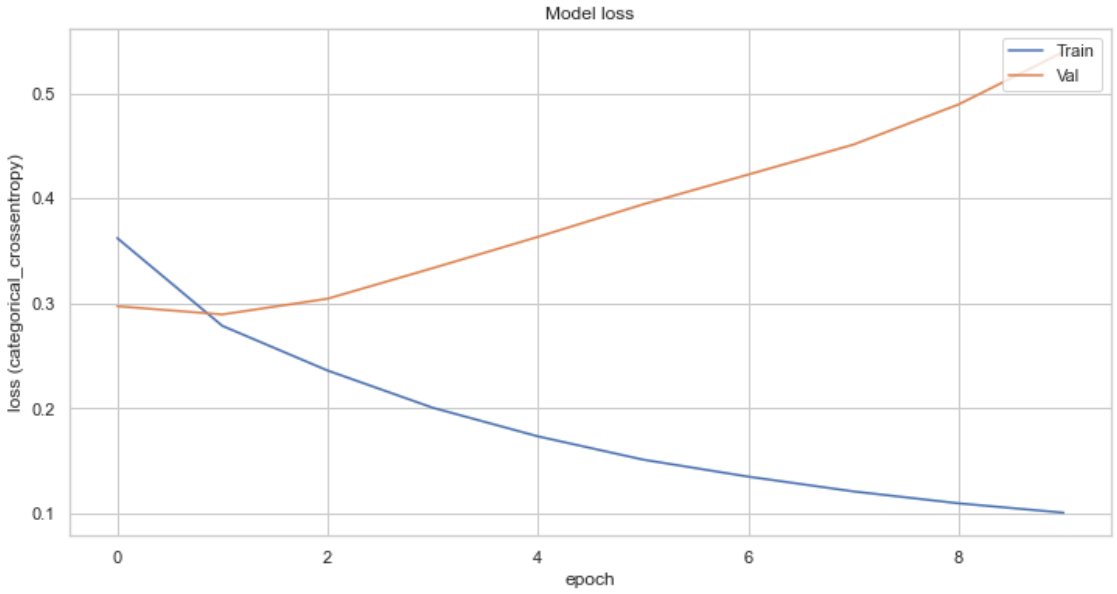
```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 16)]              0

 embedding_1 (Embedding)     (None, 16, 32)            3611296

 gru_2 (GRU)                 (None, 16, 32)            6336

 gru_3 (GRU)                 (None, 32)                6336

 dropout_1 (Dropout)         (None, 32)                0

 dense_1 (Dense)             (None, 3)                 99

=================================================================
Total params: 3,624,067
Trainable params: 3,624,067
Non-trainable params: 0
_________________________________________________________________
Epoch 1/10
4843/4843 [==============================] - ETA: 0s - loss: 0.3623 - accuracy:
0.7676 - precision_1: 0.7946 - recall_1: 0.7358
Epoch 1: accuracy improved from -inf to 0.76761, saving model to output\gruTF.h5
4843/4843 [==============================] - 192s 39ms/step - loss: 0.3623 -
```

```
accuracy: 0.7676 - precision_1: 0.7946 - recall_1: 0.7358 - val_loss: 0.2974 -
val_accuracy: 0.8128 - val_precision_1: 0.8297 - val_recall_1: 0.7944
Epoch 2/10
4843/4843 [==============================] - ETA: 0s - loss: 0.2787 - accuracy:
0.8345 - precision_1: 0.8475 - recall_1: 0.8203
Epoch 2: accuracy improved from 0.76761 to 0.83449, saving model to
output\gruTF.h5
4843/4843 [==============================] - 193s 40ms/step - loss: 0.2787 -
accuracy: 0.8345 - precision_1: 0.8475 - recall_1: 0.8203 - val_loss: 0.2896 -
val_accuracy: 0.8153 - val_precision_1: 0.8275 - val_recall_1: 0.8037
Epoch 3/10
4842/4843 [==============================>.] - ETA: 0s - loss: 0.2363 - accuracy:
0.8647 - precision_1: 0.8744 - recall_1: 0.8550
Epoch 3: accuracy improved from 0.83449 to 0.86472, saving model to
output\gruTF.h5
4843/4843 [==============================] - 196s 41ms/step - loss: 0.2363 -
accuracy: 0.8647 - precision_1: 0.8744 - recall_1: 0.8550 - val_loss: 0.3046 -
val_accuracy: 0.8083 - val_precision_1: 0.8183 - val_recall_1: 0.7981
Epoch 4/10
4843/4843 [==============================] - ETA: 0s - loss: 0.2008 - accuracy:
0.8888 - precision_1: 0.8963 - recall_1: 0.8819
Epoch 4: accuracy improved from 0.86472 to 0.88878, saving model to
output\gruTF.h5
4843/4843 [==============================] - 197s 41ms/step - loss: 0.2008 -
accuracy: 0.8888 - precision_1: 0.8963 - recall_1: 0.8819 - val_loss: 0.3337 -
val_accuracy: 0.7983 - val_precision_1: 0.8074 - val_recall_1: 0.7895
Epoch 5/10
4843/4843 [==============================] - ETA: 0s - loss: 0.1736 - accuracy:
0.9052 - precision_1: 0.9118 - recall_1: 0.8992
Epoch 5: accuracy improved from 0.88878 to 0.90524, saving model to
output\gruTF.h5
4843/4843 [==============================] - 198s 41ms/step - loss: 0.1736 -
accuracy: 0.9052 - precision_1: 0.9118 - recall_1: 0.8992 - val_loss: 0.3634 -
val_accuracy: 0.7933 - val_precision_1: 0.8007 - val_recall_1: 0.7864
Epoch 6/10
4843/4843 [==============================] - ETA: 0s - loss: 0.1514 - accuracy:
0.9180 - precision_1: 0.9241 - recall_1: 0.9130
Epoch 6: accuracy improved from 0.90524 to 0.91797, saving model to
output\gruTF.h5
4843/4843 [==============================] - 199s 41ms/step - loss: 0.1514 -
accuracy: 0.9180 - precision_1: 0.9241 - recall_1: 0.9130 - val_loss: 0.3944 -
val_accuracy: 0.7809 - val_precision_1: 0.7899 - val_recall_1: 0.7714
Epoch 7/10
4843/4843 [==============================] - ETA: 0s - loss: 0.1352 - accuracy:
0.9267 - precision_1: 0.9318 - recall_1: 0.9226
Epoch 7: accuracy improved from 0.91797 to 0.92666, saving model to
output\gruTF.h5
4843/4843 [==============================] - 201s 41ms/step - loss: 0.1352 -
```

```
accuracy: 0.9267 - precision_1: 0.9318 - recall_1: 0.9226 - val_loss: 0.4227 -
val_accuracy: 0.7833 - val_precision_1: 0.7903 - val_recall_1: 0.7771
Epoch 8/10
4842/4843 [============================>.] - ETA: 0s - loss: 0.1212 - accuracy:
0.9346 - precision_1: 0.9392 - recall_1: 0.9308
Epoch 8: accuracy improved from 0.92666 to 0.93456, saving model to
output\gruTF.h5
4843/4843 [==============================] - 201s 42ms/step - loss: 0.1212 -
accuracy: 0.9346 - precision_1: 0.9392 - recall_1: 0.9308 - val_loss: 0.4513 -
val_accuracy: 0.7760 - val_precision_1: 0.7832 - val_recall_1: 0.7690
Epoch 9/10
4843/4843 [==============================] - ETA: 0s - loss: 0.1098 - accuracy:
0.9407 - precision_1: 0.9451 - recall_1: 0.9372
Epoch 9: accuracy improved from 0.93456 to 0.94075, saving model to
output\gruTF.h5
4843/4843 [==============================] - 201s 42ms/step - loss: 0.1098 -
accuracy: 0.9407 - precision_1: 0.9451 - recall_1: 0.9372 - val_loss: 0.4894 -
val_accuracy: 0.7749 - val_precision_1: 0.7804 - val_recall_1: 0.7685
Epoch 10/10
4843/4843 [==============================] - ETA: 0s - loss: 0.1009 - accuracy:
0.9457 - precision_1: 0.9494 - recall_1: 0.9425
Epoch 10: accuracy improved from 0.94075 to 0.94566, saving model to
output\gruTF.h5
4843/4843 [==============================] - 200s 41ms/step - loss: 0.1009 -
accuracy: 0.9457 - precision_1: 0.9494 - recall_1: 0.9425 - val_loss: 0.5395 -
val_accuracy: 0.7696 - val_precision_1: 0.7740 - val_recall_1: 0.7646
```

[ ]:
```
plt.figure(figsize=(12,6))
plt.plot(model.history.history['loss'][:])
plt.plot(model.history.history['val_loss'][:])
plt.title('Model loss')
plt.xlabel('epoch')
plt.ylabel('loss (categorical_crossentropy)')
plt.legend(['Train', 'Val'], loc='upper right')
```

[ ]:
```
<matplotlib.legend.Legend at 0x2094e956fb0>
```



0.2.6 LDA

[ ]:
```
df2.head()
df2['clean_text'] = df2.split_text.apply(lambda x: ' '.join(x))
df2.head()
df2.to_csv('data_model/clean_df.csv')
df2.shape
df = pd.read_csv('data_model/clean_df.csv')
cleandf = df.dropna(axis=0)
cleandf.shape
cleandf = cleandf.iloc[:,2:]
cleandf.to_csv('data_model/clean_df.csv')
cleandf.info()
```

[ ]:
```
(206594, 7)
```

[ ]:
```
df = pd.read_csv('data_model/clean_df.csv',index_col=0)
df.isna().sum()
```

[ ]:
```
Unnamed: 0     0
text           0
score          0
category       0
filter_text    0
split_text     0
clean_text     0
dtype: int64
```

[ ]:
```
# ignore high freq and low freq
cv = CountVectorizer(max_df=0.95, min_df=2)

# document term matrix - dtm
dtm = cv.fit_transform(df['clean_text'])
```

[ ]:
```
# initialize insteace with 4 category, since there was 4 different data sets to␣
 ↪each category.
LDA = LatentDirichletAllocation(n_components=4,random_state=42)
LDA.fit(dtm)
```

[ ]:
```
LatentDirichletAllocation(n_components=4, random_state=42)
```

[ ]:
```
# print top 15 words in each topic
for index,topic in enumerate(LDA.components_):
    print(f'THE TOP 15 WORDS FOR TOPIC #{index}')
    #print([cv.get_feature_names()[i] for i in topic.argsort()[-15:]])
    print([cv.get_feature_names_out()[i] for i in topic.argsort()[-15:]])
    print('\n')
```

```
THE TOP 15 WORDS FOR TOPIC #0
['give', 'rahul', 'time', 'work', 'like', 'bjp', 'good', 'one', 'govt', 'year',
'peopl', 'congress', 'india', 'not', 'modi']


THE TOP 15 WORDS FOR TOPIC #1
['see', 'parti', 'leader', '2019', 'say', 'chowkidar', 'narendra', 'india',
'not', 'like', 'peopl', 'elect', 'vote', 'bjp', 'modi']


THE TOP 15 WORDS FOR TOPIC #2
['announc', 'drdo', 'satellit', 'minist', 'indian', 'scientist', 'say',
'nation', 'space', 'credit', 'not', 'narendra', 'pakistan', 'india', 'modi']


THE TOP 15 WORDS FOR TOPIC #3
['compani', 'team', 'also', 'would', 'india', 'nirav', 'get', 'one', 'hindu',
'use', 'not', 'muslim', 'like', 'modi', 'hai']
```

[ ]:
```
# topic results
topic_results = LDA.transform(dtm)

# append to the df
df['lda_topic']=topic_results.argmax(axis=1)
df.head()
```

[ ]:
```
   Unnamed: 0                                               text  score  \
0           0  when modi promised "minimum government maximum…   -1.0
1           1  talk all the nonsense and continue all the dra…    0.0
2           2  what did just say vote for modi  welcome bjp t…    1.0
3           3  asking his supporters prefix chowkidar their n…    1.0
4           4  answer who among these the most powerful world…    1.0

   category                                        filter_text  \
0  Negative  modi promised "minimum government maximum gove…
1   Neutral             talk nonsense continue drama vote modi
2  Positive  say vote modi welcome bjp told rahul main camp…
3  Positive  asking supporters prefix chowkidar names modi …
4  Positive  answer among powerful world leader today trump…

                                        split_text  \
0  ['modi', 'promis', 'minimum', 'govern', 'maxim…
1  ['talk', 'nonsens', 'continu', 'drama', 'vote'…
2  ['say', 'vote', 'modi', 'welcom', 'bjp', 'told…
3  ['ask', 'support', 'prefix', 'chowkidar', 'nam…
4  ['answer', 'among', 'power', 'world', 'leader'…

                                        clean_text  lda_topic
0  modi promis minimum govern maximum govern expe…          0
1              talk nonsens continu drama vote modi          1
2  say vote modi welcom bjp told rahul main campa…          1
3  ask support prefix chowkidar name modi great s…          1
4  answer among power world leader today trump pu…          1
```

[ ]:
```
topic_results[0].round(2)
topic_results[0].argmax()

# append to the df
df['lda_topic']=topic_results.argmax(axis=1)
df.head()
```

[ ]:
```
   Unnamed: 0                                               text  score  \
0           0  when modi promised "minimum government maximum…   -1.0
1           1  talk all the nonsense and continue all the dra…    0.0
2           2  what did just say vote for modi  welcome bjp t…    1.0
3           3  asking his supporters prefix chowkidar their n…    1.0
4           4  answer who among these the most powerful world…    1.0

   category                                        filter_text  \
0  Negative  modi promised "minimum government maximum gove…
1   Neutral             talk nonsense continue drama vote modi
2  Positive  say vote modi welcome bjp told rahul main camp…
3  Positive  asking supporters prefix chowkidar names modi …
```

```
4  Positive  answer among powerful world leader today trump…

                                          split_text  \
0  ['modi', 'promis', 'minimum', 'govern', 'maxim…
1  ['talk', 'nonsens', 'continu', 'drama', 'vote'…
2  ['say', 'vote', 'modi', 'welcom', 'bjp', 'told…
3  ['ask', 'support', 'prefix', 'chowkidar', 'nam…
4  ['answer', 'among', 'power', 'world', 'leader'…

                                          clean_text  lda_topic
0  modi promis minimum govern maximum govern expe…           0
1              talk nonsens continu drama vote modi           1
2  say vote modi welcom bjp told rahul main campa…           1
3  ask support prefix chowkidar name modi great s…           1
4  answer among power world leader today trump pu…           1
```

0.3 Reference

Data from: https://www.kaggle.com/code/kritanjalijain/twitter-sentiment-analysis-lstm/notebook

GRU from: https://www.kaggle.com/code/tanulsingh077/deep-learning-for-nlp-zero-to-transformers-bert#GRU's