```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('ggplot')
import nltk
nltk.download('punkt_tab')
nltk.download('averaged_perceptron_tagger_eng')
nltk.download('maxent_ne_chunker_tab')
nltk.download('words')
nltk.download('wordnet')
nltk.download('stopwords')
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt_tab.zip.
[nltk_data] Downloading package averaged_perceptron_tagger_eng to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger_eng.zip.
[nltk_data] Downloading package maxent_ne_chunker_tab to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping chunkers/maxent_ne_chunker_tab.zip.
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Unzipping corpora/words.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
True
```

```
dfs = pd.read_csv('/content/Reviews.csv')
```

```
dfs.head()
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Tex
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I hav bough several c th Vitalit canne d.
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Produc arrive labeled a Jumb Salte

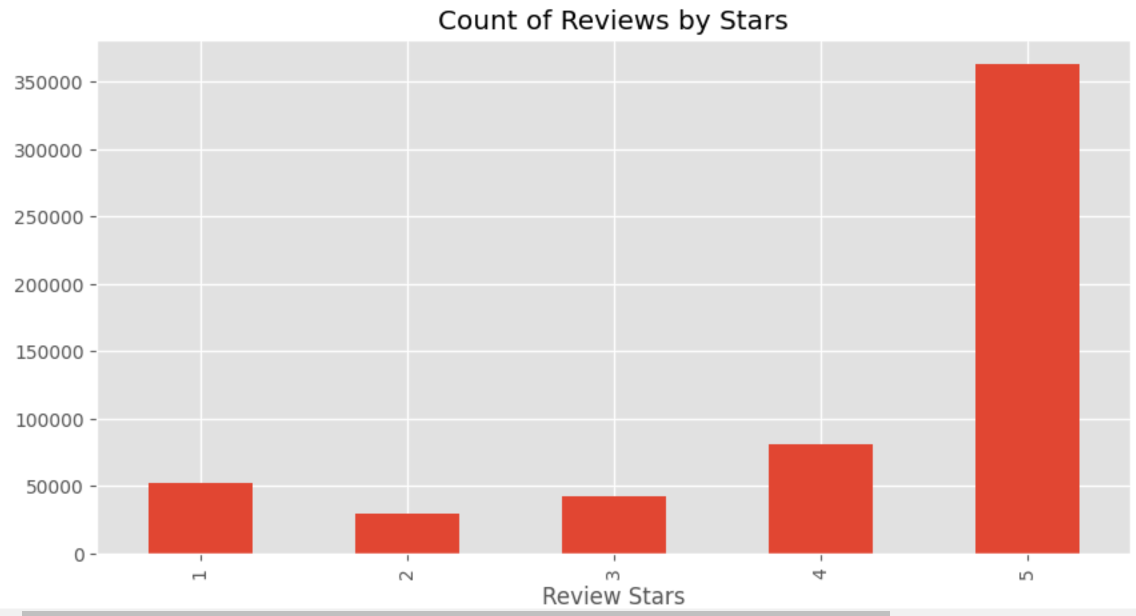
```
dfs['Text'].values[1]
```

```
'Product arrived labeled as Jumbo Salted Peanuts...the peanuts were actually small sized unsalted. Not sure if this was an error or if
the vendor intended to represent the product as "Jumbo".'
```

```
dfs.shape
```

```
(568454, 10)
```

```
#EDA
ax=dfs['Score'].value_counts().sort_index().plot(kind='bar', title='Count of Reviews by Stars', figsize=(10,5))
ax.set_xlabel('Review Stars')
plt.show()
```



```
ex =dfs['Text'].values[50]
print(ex)
```

```
This oatmeal is not good. Its mushy, soft, I don't like it. Quaker Oats is the way to go.
```

```
tokens=nltk.word_tokenize(ex)
tokens[:10]
```

```
['This', 'oatmeal', 'is', 'not', 'good', '.', 'Its', 'mushy', ',', 'soft']
```

```
tagged = nltk.pos_tag(tokens)
tagged[:10]
```

```
[('This', 'DT'),
 ('oatmeal', 'NN'),
 ('is', 'VBZ'),
 ('not', 'RB'),
 ('good', 'JJ'),
 ('.', '.'),
 ('Its', 'PRP$'),
 ('mushy', 'NN'),
 (',', ','),
 ('soft', 'JJ')]
```

```
entities=nltk.chunk.ne_chunk(tagged)
entities.pprint()
```

```
(S
  This/DT
  oatmeal/NN
  is/VBZ
  not/RB
  good/JJ
  ./.
  Its/PRP$
  mushy/NN
  ,/,
  soft/JJ
  ,/,
  I/PRP
  do/VBP
  n't/RB
  like/VB
  it/PRP
  ./.
  (ORGANIZATION Quaker/NNP Oats/NNPS)
  is/VBZ
  the/DT
  way/NN
  to/TO
  go/VB
```

```
        ./.)
```

```
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
sia = SentimentIntensityAnalyzer()
```

```
sia.polarity_scores('I am so not sad')
```

```
{'neg': 0.0, 'neu': 0.514, 'pos': 0.486, 'compound': 0.4277}
```

```
sia.polarity_scores('I am so happy')
```

```
{'neg': 0.0, 'neu': 0.334, 'pos': 0.666, 'compound': 0.6115}
```

```
 sia.polarity_scores(ex)
```

```
{'neg': 0.22, 'neu': 0.78, 'pos': 0.0, 'compound': -0.5448}
```

```
from tqdm import tqdm

# Ensure 'Text' column has valid strings
dfs['Text'] = dfs['Text'].fillna('').astype(str)

# Initialize results dictionary
results = {}

# Iterate through DataFrame rows with progress bar
for i, row in tqdm(dfs.iterrows(), total=len(dfs)):
    try:
        text = row['Text']
        myid = row['Id']

        # Compute VADER sentiment scores
        results[myid] = sia.polarity_scores(text)
    except Exception as e:
        print(f"Error processing row with Id {row['Id']}: {e}")
```

```
100%|██████████| 568454/568454 [12:11<00:00, 777.29it/s]
```

```
results
```

```
 976: {'neg': 0.0, 'neu': 0.846, 'pos': 0.154, 'compound': 0.6369},
 977: {'neg': 0.056, 'neu': 0.876, 'pos': 0.069, 'compound': -0.0676},
 978: {'neg': 0.036, 'neu': 0.836, 'pos': 0.129, 'compound': 0.9554},
 979: {'neg': 0.089, 'neu': 0.791, 'pos': 0.12, 'compound': 0.5264},
 980: {'neg': 0.0, 'neu': 0.661, 'pos': 0.339, 'compound': 0.9233},
 981: {'neg': 0.0, 'neu': 0.484, 'pos': 0.516, 'compound': 0.9595},
 982: {'neg': 0.0, 'neu': 0.767, 'pos': 0.233, 'compound': 0.9552},
 983: {'neg': 0.0, 'neu': 0.883, 'pos': 0.117, 'compound': 0.6249},
 984: {'neg': 0.0, 'neu': 0.781, 'pos': 0.219, 'compound': 0.9663},
 985: {'neg': 0.0, 'neu': 0.846, 'pos': 0.154, 'compound': 0.7181},
 986: {'neg': 0.022, 'neu': 0.805, 'pos': 0.172, 'compound': 0.9401},
 987: {'neg': 0.059, 'neu': 0.778, 'pos': 0.163, 'compound': 0.5984},
 988: {'neg': 0.0, 'neu': 0.805, 'pos': 0.195, 'compound': 0.902},
 989: {'neg': 0.0, 'neu': 0.785, 'pos': 0.215, 'compound': 0.9586},
 990: {'neg': 0.0, 'neu': 0.674, 'pos': 0.326, 'compound': 0.9631},
 991: {'neg': 0.032, 'neu': 0.667, 'pos': 0.301, 'compound': 0.973},
 992: {'neg': 0.0, 'neu': 0.633, 'pos': 0.367, 'compound': 0.9749},
 993: {'neg': 0.0, 'neu': 0.662, 'pos': 0.338, 'compound': 0.9636},
 994: {'neg': 0.0, 'neu': 0.886, 'pos': 0.114, 'compound': 0.8858},
 995: {'neg': 0.0, 'neu': 0.828, 'pos': 0.172, 'compound': 0.7552},
 996: {'neg': 0.026, 'neu': 0.721, 'pos': 0.253, 'compound': 0.9788},
 997: {'neg': 0.0, 'neu': 0.786, 'pos': 0.214, 'compound': 0.9309},
 998: {'neg': 0.0, 'neu': 0.673, 'pos': 0.327, 'compound': 0.9634},
 999: {'neg': 0.063, 'neu': 0.874, 'pos': 0.062, 'compound': -0.0129},
 1000: {'neg': 0.027, 'neu': 0.939, 'pos': 0.034, 'compound': -0.1027},
 ...}
```

```
f = pd.DataFrame(results)
print(f)
```

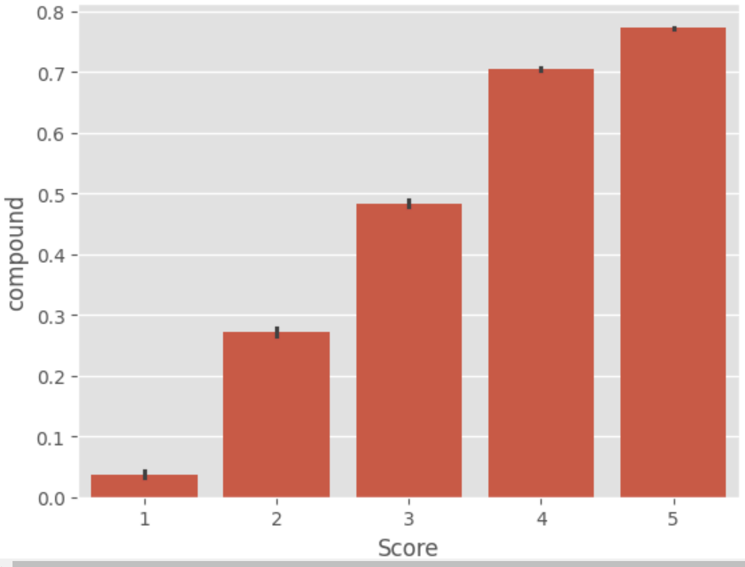
```
               1       2       3       4       5       6       7       8       \
neg       0.0000  0.1380  0.0910     0.0  0.0000   0.029  0.0340  0.0000   
neu       0.6950  0.8620  0.7540     1.0  0.5520   0.809  0.6930  0.5200   
pos       0.3050  0.0000  0.1550     0.0  0.4480   0.163  0.2730  0.4800   
compound  0.9441 -0.5664  0.8265     0.0  0.9468   0.883  0.9346  0.9487   

               9       10      ...  568445  568446  568447  568448  568449  568450  \
neg       0.0000  0.0000  ...  0.0000  0.0560  0.0740   0.031  0.1480  0.0720   
neu       0.8510  0.7050  ...  0.8570  0.8290  0.8220   0.845  0.7480  0.6000   
pos       0.1490  0.2950  ...  0.1430  0.1150  0.1040   0.124  0.1030  0.3270   
compound  0.6369  0.8313  ...  0.6892  0.5251  0.1655   0.902 -0.0675  0.8589   

          568451  568452  568453  568454  
neg       0.1900  0.0370  0.0410  0.0000  
neu       0.6970  0.8840  0.5060  0.8460  
pos       0.1140  0.0780  0.4520  0.1540  
compound -0.4848  0.4352  0.9717  0.4754  

[4 rows x 568454 columns]
```

```
# Create the df DataFrame with sentiment scores as columns
vaders = pd.DataFrame(results).T  # Transpose to have sentiment scores as columns
vaders = vaders.reset_index().rename(columns={'index': 'Id'})  # Rename index column to 'Id'

# Ensure both 'Id' columns are the same data type
dfs['Id'] = dfs['Id'].astype(str)
vaders['Id'] = vaders['Id'].astype(str)

# Now, merge the two DataFrames on the 'Id' column
vaders = vaders.merge(dfs, how='left', on='Id')  # Use 'on' to specify the column to merge on

# Verify the result
print(vaders.head())
```

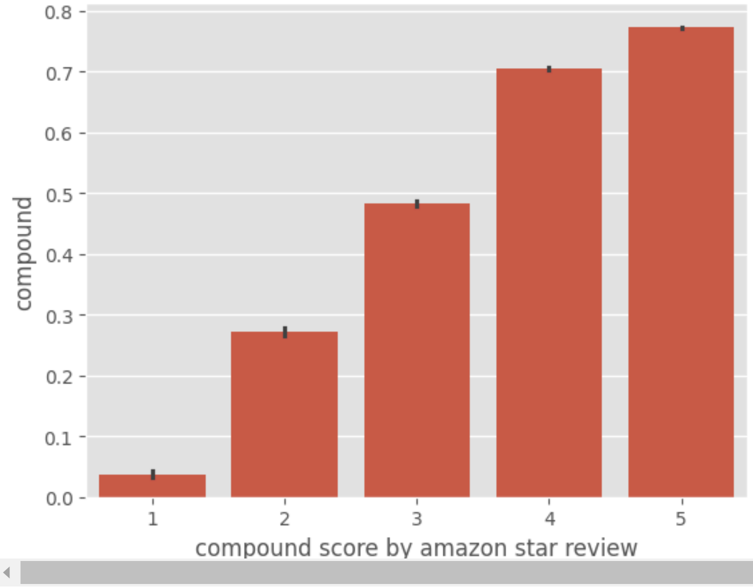
```
  Id    neg    neu    pos  compound   ProductId          UserId  \
0  1  0.000  0.695  0.305    0.9441  B001E4KFG0  A3SGXH7AUHU8GW   
1  2  0.138  0.862  0.000   -0.5664  B00813GRG4  A1D87F6ZCVE5NK   
2  3  0.091  0.754  0.155    0.8265  B000LQOCH0   ABXLMWJIXXAIN   
3  4  0.000  1.000  0.000    0.0000  B000UA0QIQ  A395BORC6FGVXV   
4  5  0.000  0.552  0.448    0.9468  B006K2ZZ7K  A1UQRSCLF8GW1T   

                      ProfileName  HelpfulnessNumerator  \
0                      delmartian                     1   
1                          dll pa                     0   
2  Natalia Corres "Natalia Corres"                     1   
3                            Karl                     3   
4    Michael D. Bigham "M. Wassir"                     0   

   HelpfulnessDenominator  Score        Time                Summary  \
0                       1      5  1303862400  Good Quality Dog Food   
```

```
1                       0      1  1346976000      Not as Advertised
2                       1      4  1219017600  "Delight" says it all
3                       3      2  1307923200         Cough Medicine
4                       0      5  1350777600            Great taffy

                                                Text
0  I have bought several of the Vitality canned d...
1  Product arrived labeled as Jumbo Salted Peanut...
2  This is a confection that has been around a fe...
3  If you are looking for the secret ingredient i...
4  Great taffy at a great price.  There was a wid...
```

```
sns.barplot (data=vaders,x='Score',y='compound')
ax.set_xlabel('Review Stars')
plt.show()
```



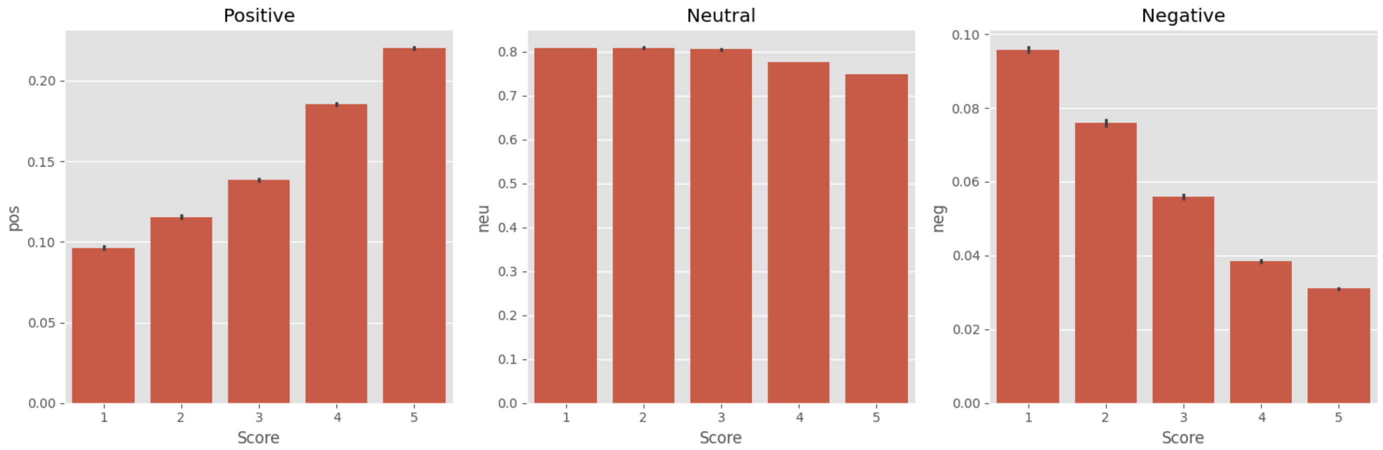
```
ax=sns.barplot (data=vaders,x='Score',y='compound')
ax.set_xlabel('compound score by amazon star review')
plt.show()
```



Suggested code may be subject to a license |

```
# Ensure columns exist and have no missing values
required_columns = ['Score', 'pos', 'neu', 'neg']
vaders = vaders.dropna(subset=required_columns)

# Initialize subplots
fig, axs = plt.subplots(1, 3, figsize=(15, 5))
```

```
# Plot the data
sns.barplot(data=vaders, x='Score', y='pos', ax=axs[0])
sns.barplot(data=vaders, x='Score', y='neu', ax=axs[1])
sns.barplot(data=vaders, x='Score', y='neg', ax=axs[2])

# Add titles
axs[0].set_title('Positive')
axs[1].set_title('Neutral')
axs[2].set_title('Negative')

# Adjust layout and show plot
plt.tight_layout()
plt.show()
```



Suggested code may be subject to a license | vthz/Natural_Language_Processing | maenbos3/Sentiment-Analysis-on-Harry-Potter-Hogwart-Legacy-Review

```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
```

Suggested code may be subject to a license | vthz/Natural_Language_Processing | maenbos3/Sentiment-Analysis-on-Harry-Potter-Hogwart-Legacy-Review

```
MODEL=f'cardiffnlp/twitter-roberta-base-sentiment'
tokenizer=AutoTokenizer.from_pretrained(MODEL)
model=AutoModelForSequenceClassification.from_pretrained(MODEL)
```

```
/usr/local/lib/python3.10/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secre
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
```

```
config.json:  100%                              747/747  [00:00<00:00,  18.4kB/s]
vocab.json:  100%                               899k/899k  [00:00<00:00,  13.1MB/s]
merges.txt:  100%                               456k/456k  [00:00<00:00,  17.1MB/s]
special_tokens_map.json:  100%                           150/150  [00:00<00:00,  7.34kB/s]
pytorch_model.bin:  100%                            499M/499M  [00:02<00:00,  208MB/s]
```

```
print(ex)
sia.polarity_scores(ex)
```

```
This oatmeal is not good. Its mushy, soft, I don't like it. Quaker Oats is the way to go.
{'neg': 0.22, 'neu': 0.78, 'pos': 0.0, 'compound': -0.5448}
```

Suggested code may be subject to a license | AyambaSumaila/NLP-Sentiment-Analysis-Project

```
encoded_text=tokenizer(ex,return_tensors='pt')
output = model(**encoded_text)
output = model(**encoded_text)
scores=output[0][0].detach().numpy()
scores=softmax(scores)
scores_dict={
    'roberta_neg':scores[0],
    'roberta_neu':scores[1],
    'roberta_pos':scores[2]
}
print(scores_dict)
```

```
{'roberta_neg': 0.97635514, 'roberta_neu': 0.020687465, 'roberta_pos': 0.0029573706}
```

Suggested code may be subject to a license | AnnNaserNabil/DS_ML_in_Bangla

```
def polarity_scores_roberta(example):
    encoded_text=tokenizer(example,return_tensors='pt')
    output = model(**encoded_text)
```

```
from tqdm import tqdm

# Initialize an empty dictionary to store the results
results = {}

# Iterate through the DataFrame rows
for i, row in tqdm(dfs.iterrows(), total=len(dfs)):
    try:
        # Extract text and ID
        text = row['Text']
        myid = row['Id']

        # Compute VADER sentiment scores
        vader_result = sia.polarity_scores(text)
        vader_result_rename = {f"vader_{key}": value for key, value in vader_result.items()}

        # Compute RoBERTa sentiment scores
        roberta_result = polarity_scores_roberta(text)

        # Validate RoBERTa result
        if not isinstance(roberta_result, dict):
            roberta_result = {}  # Fallback to empty dictionary

        # Combine VADER and RoBERTa results into a single dictionary
        combined_result = {**vader_result_rename, **roberta_result}

        # Store the combined result in the results dictionary
        results[myid] = combined_result

    # Handle runtime errors
    except RuntimeError as e:
        print(f"Error processing row with Id {myid}: {e}")
```

```
  0%|          | 188/568454 [01:07<43:15:35,  3.65it/s]Error processing row with Id 187: The expanded size of the tensor (546) must m
  0%|          | 528/568454 [03:15<72:15:00,  2.18it/s]Error processing row with Id 529: The expanded size of the tensor (639) must m
  0%|          | 539/568454 [03:18<38:37:41,  4.08it/s]Error processing row with Id 540: The expanded size of the tensor (1386) must 
  0%|          | 745/568454 [04:42<128:10:04,  1.23it/s]Error processing row with Id 746: The expanded size of the tensor (705) must 
  0%|          | 862/568454 [05:25<84:27:52,  1.87it/s]Error processing row with Id 863: The expanded size of the tensor (605) must m
  0%|          | 1052/568454 [06:37<45:53:06,  3.43it/s]Error processing row with Id 1053: The expanded size of the tensor (789) must
  0%|          | 1071/568454 [06:45<45:55:43,  3.43it/s]Error processing row with Id 1070: The expanded size of the tensor (585) must
  0%|          | 1157/568454 [07:17<29:33:42,  5.33it/s]Error processing row with Id 1156: The expanded size of the tensor (597) must
  0%|          | 1320/568454 [08:14<78:57:16,  2.00it/s]Error processing row with Id 1321: The expanded size of the tensor (1246) mus
  0%|          | 1374/568454 [08:32<119:50:09,  1.31it/s]Error processing row with Id 1375: The expanded size of the tensor (569) mus
  0%|          | 1497/568454 [09:20<38:04:50,  4.14it/s]Error processing row with Id 1498: The expanded size of the tensor (1195) mus
  0%|          | 1574/568454 [09:51<108:36:22,  1.45it/s]Error processing row with Id 1575: The expanded size of the tensor (555) mus
  0%|          | 1797/568454 [11:10<22:21:56,  7.04it/s]Error processing row with Id 1796: The expanded size of the tensor (527) must
  0%|          | 1827/568454 [11:22<42:05:00,  3.74it/s]Error processing row with Id 1826: The expanded size of the tensor (976) must
  0%|          | 2168/568454 [13:43<49:44:25,  3.16it/s]Error processing row with Id 2169: The expanded size of the tensor (585) must
  0%|          | 2247/568454 [14:13<42:05:06,  3.74it/s]Error processing row with Id 2248: The expanded size of the tensor (1043) mus
  0%|          | 2475/568454 [15:41<70:08:48,  2.24it/s]Error processing row with Id 2476: The expanded size of the tensor (550) must
  0%|          | 2493/568454 [15:49<35:00:58,  4.49it/s]Error processing row with Id 2492: The expanded size of the tensor (803) must
  0%|          | 2583/568454 [16:25<62:33:15,  2.51it/s]Error processing row with Id 2584: The expanded size of the tensor (879) must
  0%|          | 2609/568454 [16:33<68:53:34,  2.28it/s]Error processing row with Id 2610: The expanded size of the tensor (779) must
  1%|          | 2896/568454 [18:23<59:34:49,  2.64it/s]Error processing row with Id 2897: The expanded size of the tensor (603) must
Error processing row with Id 2898: The expanded size of the tensor (740) must match the existing size (514) at non-singleton dimensio
  1%|          | 2901/568454 [18:24<39:59:59,  3.93it/s]Error processing row with Id 2902: The expanded size of the tensor (538) must
  1%|          | 2929/568454 [18:34<46:53:46,  3.35it/s]Error processing row with Id 2928: The expanded size of the tensor (791) must
  1%|          | 2941/568454 [18:38<94:18:47,  1.67it/s]Error processing row with Id 2942: The expanded size of the tensor (1073) mus
  1%|          | 2943/568454 [18:39<59:14:02,  2.65it/s]Error processing row with Id 2944: The expanded size of the tensor (524) must
  1%|          | 2946/568454 [18:40<63:00:50,  2.49it/s]Error processing row with Id 2947: The expanded size of the tensor (524) must
Error processing row with Id 2948: The expanded size of the tensor (1073) must match the existing size (514) at non-singleton dimensi
  1%|          | 3021/568454 [19:08<51:45:05,  3.03it/s]Error processing row with Id 3022: The expanded size of the tensor (945) must
Error processing row with Id 3023: The expanded size of the tensor (940) must match the existing size (514) at non-singleton dimensio
  1%|          | 3024/568454 [19:09<40:40:33,  3.86it/s]Error processing row with Id 3025: The expanded size of the tensor (826) must
  1%|          | 3305/568454 [20:52<50:28:41,  3.11it/s]Error processing row with Id 3306: The expanded size of the tensor (860) must
  1%|          | 3787/568454 [23:44<60:46:21,  2.58it/s]Error processing row with Id 3788: The expanded size of the tensor (587) must
  1%|          | 3968/568454 [24:43<37:17:53,  4.20it/s]Error processing row with Id 3969: The expanded size of the tensor (551) must
  1%|          | 4108/568454 [25:41<39:28:36,  3.97it/s]Error processing row with Id 4107: The expanded size of the tensor (585) must
  1%|          | 4109/568454 [25:42<45:42:56,  3.43it/s]Error processing row with Id 4110: The expanded size of the tensor (593) must
  1%|          | 4306/568454 [26:58<61:01:12,  2.57it/s]Error processing row with Id 4307: The expanded size of the tensor (886) must
  1%|          | 4315/568454 [27:03<121:20:20,  1.29it/s]Error processing row with Id 4316: The expanded size of the tensor (928) mus
  1%|          | 4407/568454 [27:36<87:38:01,  1.79it/s]Error processing row with Id 4408: The expanded size of the tensor (793) must
  1%|          | 4484/568454 [28:00<26:39:12,  5.88it/s]Error processing row with Id 4483: The expanded size of the tensor (635) must
  1%|          | 4511/568454 [28:13<48:13:02,  3.25it/s]Error processing row with Id 4512: The expanded size of the tensor (540) must
  1%|          | 4552/568454 [28:32<58:46:39,  2.66it/s]Error processing row with Id 4553: The expanded size of the tensor (543) must
  1%|          | 4582/568454 [28:52<156:10:45,  1.00it/s]Error processing row with Id 4583: The expanded size of the tensor (826) mus
  1%|          | 5039/568454 [31:57<73:46:20,  2.12it/s]Error processing row with Id 5040: The expanded size of the tensor (572) must
  1%|          | 5181/568454 [32:50<41:47:23,  3.74it/s]Error processing row with Id 5182: The expanded size of the tensor (521) must
  1%|          | 5193/568454 [32:57<79:54:08,  1.96it/s] Error processing row with Id 5192: The expanded size of the tensor (521) mus
  1%|          | 5364/568454 [33:59<67:05:18,  2.33it/s]Error processing row with Id 5365: The expanded size of the tensor (658) must
  1%|          | 5441/568454 [34:35<61:04:09,  2.56it/s]Error processing row with Id 5442: The expanded size of the tensor (682) must
  1%|          | 5708/568454 [36:18<44:46:56,  3.49it/s]Error processing row with Id 5709: The expanded size of the tensor (754) must
  1%|          | 5997/568454 [38:11<38:49:48,  4.02it/s]Error processing row with Id 5998: The expanded size of the tensor (1077) mus
  1%|          | 6002/568454 [38:13<86:13:36,  1.81it/s]Error processing row with Id 6003: The expanded size of the tensor (603) must
  1%|          | 6006/568454 [38:16<119:46:40,  1.30it/s]Error processing row with Id 6007: The expanded size of the tensor (556) mus
  1%|          | 6107/568454 [38:54<38:17:04,  4.08it/s]Error processing row with Id 6106: The expanded size of the tensor (706) must
  1%|          | 6233/568454 [39:43<91:03:39,  1.72it/s] Error processing row with Id 6234: The expanded size of the tensor (564) mus
  1%|          | 6557/568454 [41:54<72:24:29,  2.16it/s]Error processing row with Id 6558: The expanded size of the tensor (531) must
  1%|          | 6650/568454 [42:33<116:34:59,  1.34it/s]Error processing row with Id 6651: The expanded size of the tensor (924) mus
  1%|          | 6799/568454 [43:57<59:35:08,  2.62it/s]Error processing row with Id 6798: The expanded size of the tensor (629) must
  1%|          | 6837/568454 [44:19<95:18:49,  1.64it/s]Error processing row with Id 6838: The expanded size of the tensor (523) must
  1%|          | 6849/568454 [44:28<112:22:36,  1.39it/s]Error processing row with Id 6850: The expanded size of the tensor (1149) mu
  1%|          | 6885/568454 [44:47<54:23:13,  2.87it/s]Error processing row with Id 6886: The expanded size of the tensor (631) must
  1%|          | 6904/568454 [44:57<122:43:41,  1.27it/s]Error processing row with Id 6905: The expanded size of the tensor (544) mus
  1%|          | 6954/568454 [45:21<77:13:33,  2.02it/s]Error processing row with Id 6955: The expanded size of the tensor (617) must
  1%|          | 7076/568454 [46:08<89:25:32,  1.74it/s]Error processing row with Id 7077: The expanded size of the tensor (717) must
  1%|          | 7087/568454 [46:13<54:59:27,  2.84it/s]Error processing row with Id 7088: The expanded size of the tensor (603) must
  1%|▏         | 7200/568454 [46:56<66:46:30,  2.33it/s]Error processing row with Id 7201: The expanded size of the tensor (705) mus
  1%|▏         | 7605/568454 [49:22<39:47:55,  3.91it/s]Error processing row with Id 7604: The expanded size of the tensor (531) mus
  1%|▏         | 7607/568454 [49:22<45:37:38,  3.41it/s]Error processing row with Id 7608: The expanded size of the tensor (710) mus
  1%|▏         | 7670/568454 [49:45<82:13:51,  1.89it/s]Error processing row with Id 7671: The expanded size of the tensor (617) mus
  1%|▏         | 7700/568454 [50:02<83:20:05,  1.87it/s]Error processing row with Id 7701: The expanded size of the tensor (628) mus
  1%|▏         | 7809/568454 [50:42<62:45:20,  2.48it/s]Error processing row with Id 7810: The expanded size of the tensor (788) mus
  1%|▏         | 7908/568454 [51:18<70:16:50,  2.22it/s] Error processing row with Id 7907: The expanded size of the tensor (672) mu
  1%|▏         | 8076/568454 [52:30<55:07:37,  2.82it/s]Error processing row with Id 8077: The expanded size of the tensor (1259) mu
  1%|▏         | 8130/568454 [52:48<73:02:32,  2.13it/s]Error processing row with Id 8131: The expanded size of the tensor (584) mus
  1%|▏         | 8265/568454 [53:40<64:48:44,  2.40it/s]Error processing row with Id 8266: The expanded size of the tensor (524) mus
  2%|▏         | 8739/568454 [56:25<65:13:19,  2.38it/s]Error processing row with Id 8740: The expanded size of the tensor (600) mus
  2%|▏         | 8756/568454 [56:32<46:53:17,  3.32it/s]Error processing row with Id 8757: The expanded size of the tensor (2017) mu
  2%|▏         | 8934/568454 [57:41<33:03:14,  4.70it/s]Error processing row with Id 8935: The expanded size of the tensor (587) mus
  2%|▏         | 9589/568454 [1:01:30<90:56:37,  1.71it/s]Error processing row with Id 9590: The expanded size of the tensor (530) m
  2%|▏         | 9849/568454 [1:03:21<88:56:53,  1.74it/s] Error processing row with Id 9850: The expanded size of the tensor (2149)
  2%|▏         | 9880/568454 [1:03:31<61:58:04,  2.50it/s]Error processing row with Id 9881: The expanded size of the tensor (612) m
  2%|▏         | 9909/568454 [1:03:44<156:47:57,  1.01s/it]Error processing row with Id 9910: The expanded size of the tensor (636) 
  2%|▏         | 9922/568454 [1:03:51<74:41:18,  2.08it/s]Error processing row with Id 9923: The expanded size of the tensor (572) m
  2%|▏         | 10001/568454 [1:04:23<72:30:35,  2.14it/s]Error processing row with Id 10002: The expanded size of the tensor (790)
  2%|▏         | 10004/568454 [1:04:24<60:25:44,  2.57it/s]Error processing row with Id 10005: The expanded size of the tensor (2365
  2%|▏         | 10030/568454 [1:04:34<37:41:39,  4.12it/s]Error processing row with Id 10031: The expanded size of the tensor (638)
  2%|▏         | 10549/568454 [1:07:45<84:08:28,  1.84it/s]Error processing row with Id 10550: The expanded size of the tensor (542)
  2%|▏         | 10836/568454 [1:09:32<84:18:00,  1.84it/s]Error processing row with Id 10837: The expanded size of the tensor (725)
  2%|▏         | 10865/568454 [1:09:42<27:57:56,  5.54it/s]Error processing row with Id 10861: The expanded size of the tensor (725)
Error processing row with Id 10862: The expanded size of the tensor (725) must match the existing size (514) at non-singleton dimensi
Error processing row with Id 10863: The expanded size of the tensor (725) must match the existing size (514) at non-singleton dimensi
Error processing row with Id 10864: The expanded size of the tensor (706) must match the existing size (514) at non-singleton dimensi
  2%|▏         | 10884/568454 [1:09:48<85:56:00,  1.80it/s]Error processing row with Id 10885: The expanded size of the tensor (725)
  2%|▏         | 10923/568454 [1:10:05<59:06:08,  2.62it/s]Error processing row with Id 10924: The expanded size of the tensor (1034
  2%|▏         | 10947/568454 [1:10:11<27:19:37,  5.67it/s]Error processing row with Id 10946: The expanded size of the tensor (577)
  2%|▏         | 10968/568454 [1:10:24<75:37:43,  2.05it/s]Error processing row with Id 10969: The expanded size of the tensor (584)
  2%|▏         | 11113/568454 [1:11:18<67:33:58,  2.29it/s]Error processing row with Id 11114: The expanded size of the tensor (564)
  2%|▏         | 11240/568454 [1:12:11<21:24:26,  7.23it/s]Error processing row with Id 11238: The expanded size of the tensor (725)
Error processing row with Id 11239: The expanded size of the tensor (725) must match the existing size (514) at non-singleton dimensi
  2%|▏         | 11244/568454 [1:12:12<48:34:03,  3.19it/s]Error processing row with Id 11245: The expanded size of the tensor (680)
  2%|▏         | 11301/568454 [1:12:32<29:23:35,  5.27it/s]Error processing row with Id 11302: The expanded size of the tensor (535)
  2%|▏         | 11375/568454 [1:13:01<59:36:07,  2.60it/s]
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-27-6102ee3867b1> in <cell line: 7>()
     16 
     17         # Compute RoBERTa sentiment scores
---> 18         roberta_result = polarity_scores_roberta(text)
     19 
     20         # Validate RoBERTa result

    13 frames

/usr/local/lib/python3.10/dist-packages/torch/nn/functional.py in embedding(input, weight, padding_idx, max_norm, norm_type,
scale_grad_by_freq, sparse)
   2549         # remove once script supports set_grad_enabled
   2550         _no_grad_embedding_renorm_(weight, input, max_norm, norm_type)
-> 2551     return torch.embedding(weight, input, padding_idx, scale_grad_by_freq, sparse)
   2552 
   2553 

IndexError: index out of range in self
```

Next steps: Explain error

```
results_df = pd.DataFrame(results).T
results_df = results_df.reset_index().rename(columns={'index': 'Id'})
results_df = results_df.merge(dfs, how='left', on='Id')
```

```
results_df.head()
```

```
results_df.columns
```

```
sns.pairplot(data=results_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                  'roberta_neg', 'roberta_neu', 'roberta_pos'],
            hue='Score',
            palette='tab10')
plt.show()
```

```
results_df.query('Score == 1') \
    .sort_values('roberta_pos', ascending=False)['Text'].values[0]
```

```
results_df.query('Score == 1') \
    .sort_values('vader_pos', ascending=False)['Text'].values[0]
```

```
results_df.query('Score == 5') \
    .sort_values('roberta_neg', ascending=False)['Text'].values[0]
```

```
from transformers import pipeline

sent_pipeline = pipeline("sentiment-analysis")
```