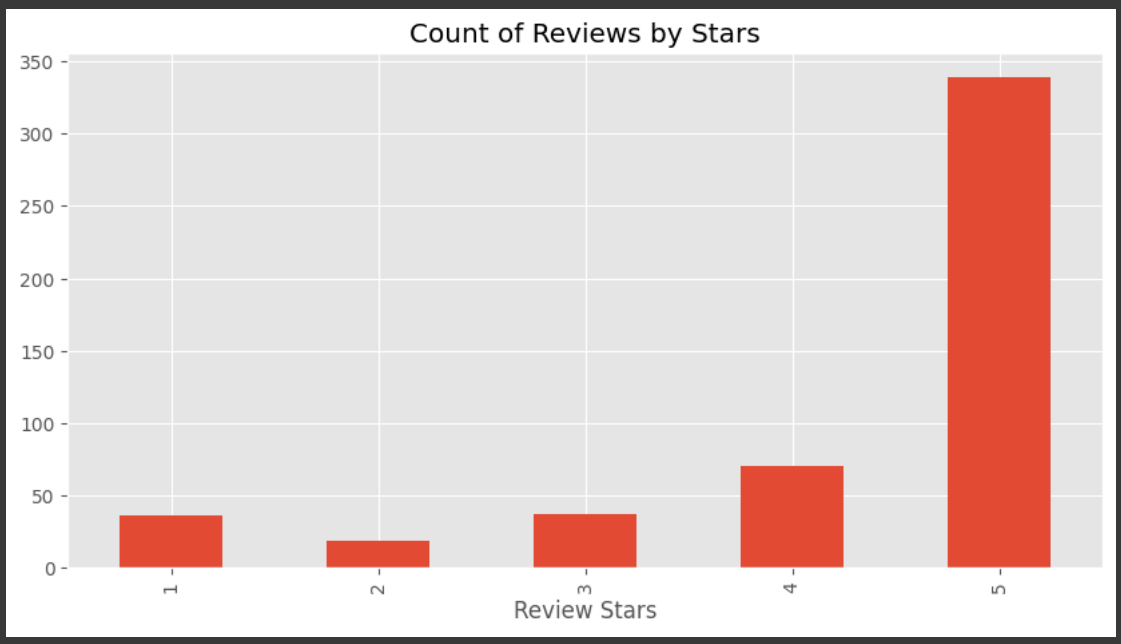
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
plt.style.use('ggplot')
import nltk
```

```
# Read in data
df = pd.read_csv('/content/drive/MyDrive/Colab Notebooks/Dataset/Sentimental Analysis_Amazon Reviews.csv')
print(df.shape)
df = df.head(500)
print(df.shape)
```

```
(40437, 10)
(500, 10)
```

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




```
example = df['Text'][50]
print(example)
```

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

```
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('maxent_ne_chunker')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package maxent_ne_chunker is already up-to-date!
True
```

```
import nltk
example = "370 chars 'Got a free package …"
tokens = nltk.word_tokenize(example)
tokens[:10]
```

```
['370', 'chars', "'Got", 'a', 'free', 'package', '…']
```

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

```
[('370', 'CD'),
 ('chars', 'NNS'),
 ("'Got", 'VBD'),
 ('a', 'DT'),
 ('free', 'JJ'),
 ('package', 'NN'),
 ('…', 'NN')]
```

```
nltk.download('words')
entities = nltk.chunk.ne_chunk(tagged)
entities.pprint()
```

```
(S 370/CD chars/NNS 'Got/VBD a/DT free/JJ package/NN …/NN)
[nltk_data] Downloading package words to /root/nltk_data...
[nltk_data]   Package words is already up-to-date!
```

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

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

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

```
{'neg': 0.0, 'neu': 0.318, 'pos': 0.682, 'compound': 0.6468}
```

```
sia.polarity_scores('This is the worst thing ever.')
```

```
{'neg': 0.451, 'neu': 0.549, 'pos': 0.0, 'compound': -0.6249}
```

```
sia.polarity_scores(example)
```

```
{'neg': 0.0, 'neu': 0.548, 'pos': 0.452, 'compound': 0.5106}
```

```
# Run the polarity score on the entire dataset
res = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
    text = row['Text']
    myid = row['Id']
    res[myid] = sia.polarity_scores(text)
```

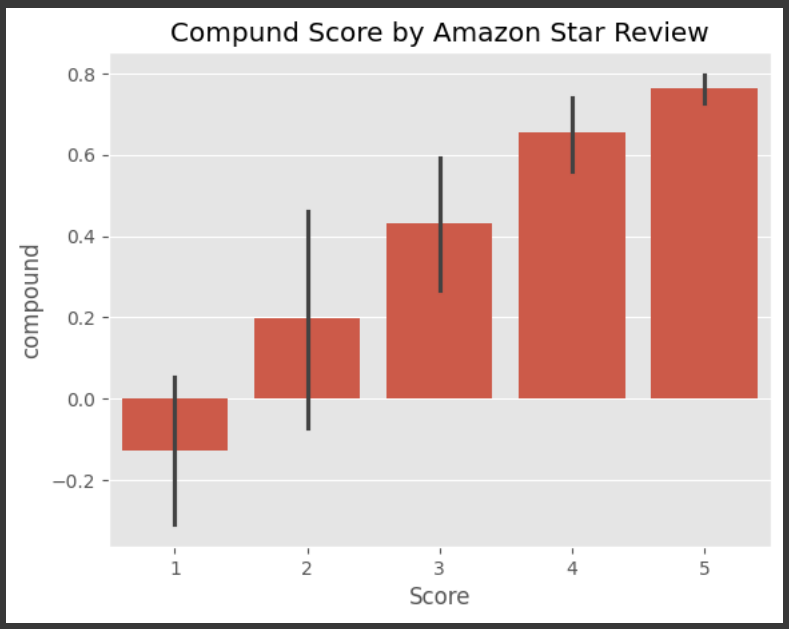
```
100% 500/500 [00:00<00:00, 1610.24it/s]
```

```
vaders = pd.DataFrame(res).T
vaders = vaders.reset_index().rename(columns={'index': 'Id'})
vaders = vaders.merge(df, how='left')
```

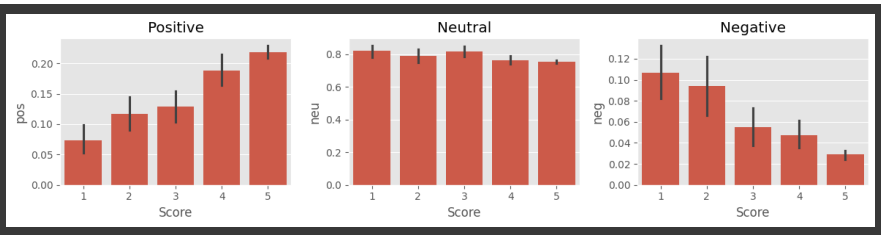
```
# Now we have sentiment score and metadata
vaders.head()
```

| | Id | neg | neu | pos | compound | ProductId | UserId | ProfileName | Help |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0.000 | 0.695 | 0.305 | 0.9441 | B001E4KFG0 | A3SGXH7AUHU8GW | delmartian | |
| 1 | 2 | 0.138 | 0.862 | 0.000 | -0.5664 | B00813GRG4 | A1D87F6ZCVE5NK | dll pa | |
| 2 | 3 | 0.091 | 0.754 | 0.155 | 0.8265 | B000LQOCH0 | ABXLMWJIXXAIN | Natalia Corres "Natalia Corres" | |
| 3 | 4 | 0.000 | 1.000 | 0.000 | 0.0000 | B000UA0QIQ | A395BORC6FGVXV | Karl | |
| 4 | 5 | 0.000 | 0.552 | 0.448 | 0.9468 | B006K2ZZ7K | A1UQRSCLF8GW1T | Michael D. Bigham "M. Wassir" | |

```
1 ax = sns.barplot(data=vaders, x='Score', y='compound')
2 ax.set_title('Compund Score by Amazon Star Review')
3 plt.show()
```




```
1 fig, axs = plt.subplots(1, 3, figsize=(12, 3))
2 sns.barplot(data=vaders, x='Score', y='pos', ax=axs[0])
3 sns.barplot(data=vaders, x='Score', y='neu', ax=axs[1])
4 sns.barplot(data=vaders, x='Score', y='neg', ax=axs[2])
5 axs[0].set_title('Positive')
6 axs[1].set_title('Neutral')
7 axs[2].set_title('Negative')
8 plt.tight_layout()
9 plt.show()
```




```
# Pre Trained Model
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
```

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

```
# VADER results on example
print(example)
sia.polarity_scores(example)
```

```
370 chars 'Got a free package …
{'neg': 0.0, 'neu': 0.548, 'pos': 0.452, 'compound': 0.5106}
```

```
# Run for Roberta Model
encoded_text = tokenizer(example, return_tensors='pt')
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': 0.015122626, 'roberta_neu': 0.7379595, 'roberta_pos': 0.24691787}
```

```
def polarity_scores_roberta(example):
    encoded_text = tokenizer(example, return_tensors='pt')
    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]
    }
    return scores_dict
```

```
res = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
    try:
        text = row['Text']
        myid = row['Id']
        vader_result = sia.polarity_scores(text)
        vader_result_rename = {}
        for key, value in vader_result.items():
            vader_result_rename[f"vader_{key}"] = value
        roberta_result = polarity_scores_roberta(text)
        both = {**vader_result_rename, **roberta_result}
        res[myid] = both
    except RuntimeError:
        print(f'Broke for id {myid}')
```

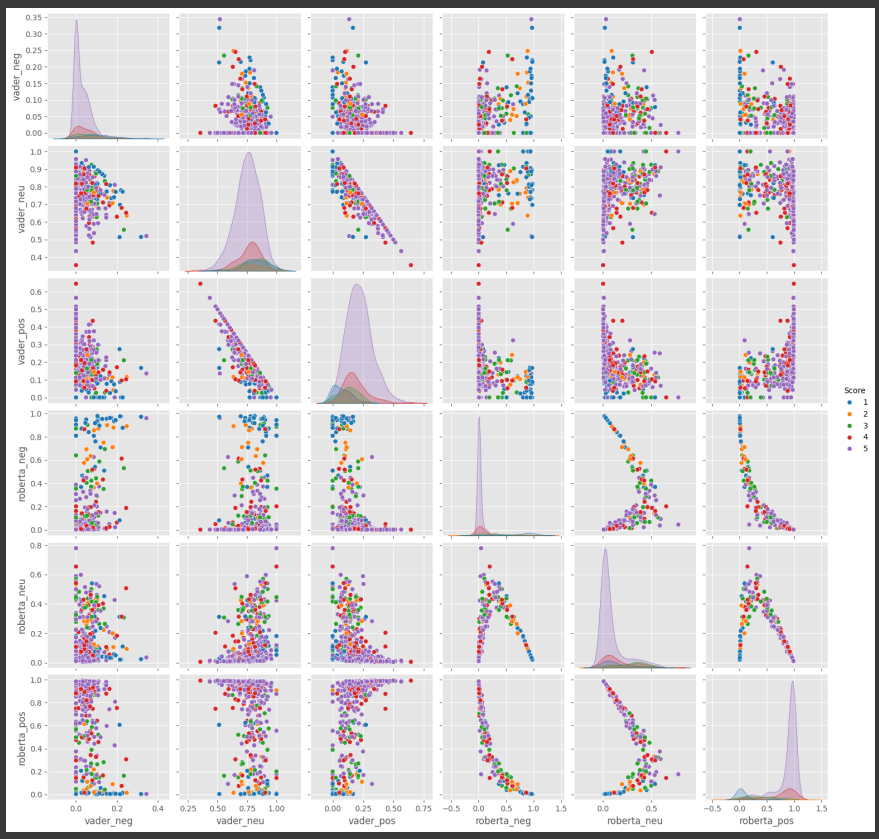
```
100% 500/500 [02:12<00:00, 4.01it/s]
Broke for id 83
Broke for id 187
```

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

```
# Compare Scores between Models
results_df.columns
```

```
Index(['Id', 'vader_neg', 'vader_neu', 'vader_pos', 'vader_compound',
       'roberta_neg', 'roberta_neu', 'roberta_pos', 'ProductId', 'UserId',
       'ProfileName', 'HelpfulnessNumerator', 'HelpfulnessDenominator',
       'Score', 'Time', 'Summary', 'Text'],
      dtype='object')
```

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



```
#Review Examples:Positive 1-Star and Negative 5-Star Reviews Lets look at some examples where the model scoring and review score dif
results_df.query('Score == 1') \
    .sort_values('roberta_pos', ascending=False)['Text'].values[0]
```

```
'I felt energized within five minutes, but it lasted for about 45 minutes. I paid
$3.99 for this drink. I could have just drunk a cup of coffee and saved my money.'
```

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

```
'So we cancelled the order.  It was cancelled without any problem.  That is a positi
ve note...'
```

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

```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my faul
+'
```

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

```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my faul
+'
```

```
from transformers import pipeline

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

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english a
Using a pipeline without specifying a model name and revision in production is not re
config.json: 100% 629/629 [00:00<00:00, 19.9kB/s]
model.safetensors: 100% 268M/268M [00:03<00:00, 27.8MB/s]
tokenizer_config.json: 48.0/48.0 [00:00<00:00,
```