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

```
from google.colab import drive

# Mount Google Drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
import pandas as pd

file_path = '###############Reviews.csv'  # Replace with your actual file path

# Read the CSV file into a DataFrame
df = pd.read_csv(file_path)

# Display the DataFrame
df.head()
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...

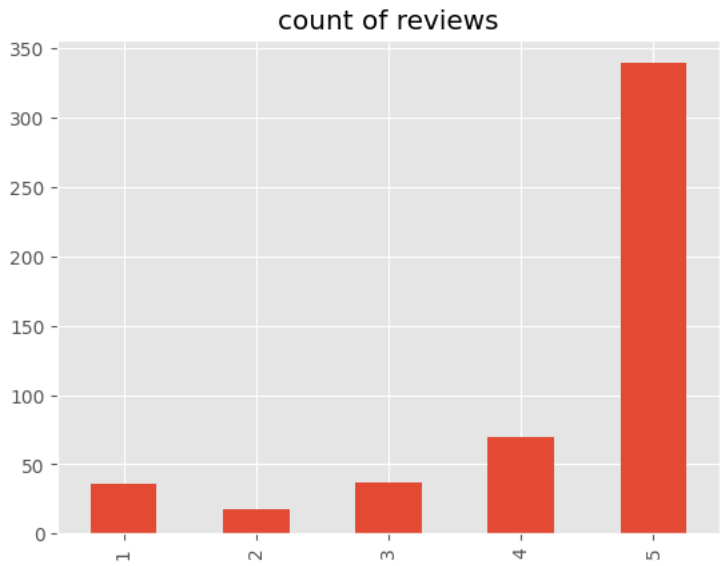
```
df.shape
```

```
(568454, 10)
```

```
## Quick EDA
df=df.head(500)
```

```
df['Score'].value_counts().sort_index().plot(kind='bar',title ="count of reviews")
```

```
<Axes: title={'center': 'count of reviews'}>
```




```
## Example
example = df['Text'][50]
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_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
True
```

```
tokens = nltk.word_tokenize(example)
tokens[0:10]
```

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

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

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Unzipping taggers/averaged_perceptron_tagger.zip.
True
```

```
tagged = nltk.pos_tag(tokens)
```

```
tagged[:10]
```

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

```
!python -m nltk.downloader maxent_ne_chunker
```

```
/usr/lib/python3.10/runpy.py:126: RuntimeWarning: 'nltk.downloader' found in sys.modules after import of package 'nltk', but prior to execution of 'nlt
  warn(RuntimeWarning(msg))
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package maxent_ne_chunker is already up-to-date!
```

```
from nltk import word_tokenize, pos_tag, ne_chunk
```

```
nltk.download('words')
```

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

```
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
  ./.)
```

Vader sentiment scoring

takes all words from our sentences, and will show how positive is statement, uses bag of word approach

```
!pip install tqdm
```

```
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (4.66.1)
```

```
nltk.download('vader_lexicon')
```

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

```
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm

sia = SentimentIntensityAnalyzer()
```

```
sia.polarity_scores("i am so  sad and i am really depressed")
```

```
{'neg': 0.583, 'neu': 0.417, 'pos': 0.0, 'compound': -0.7897}
```

```
sia.polarity_scores("i am so happy")
```

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

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

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

```
## RUN polarity score on 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, 1090.69it/s]
```

```
vaders =pd.DataFrame(res).T
```

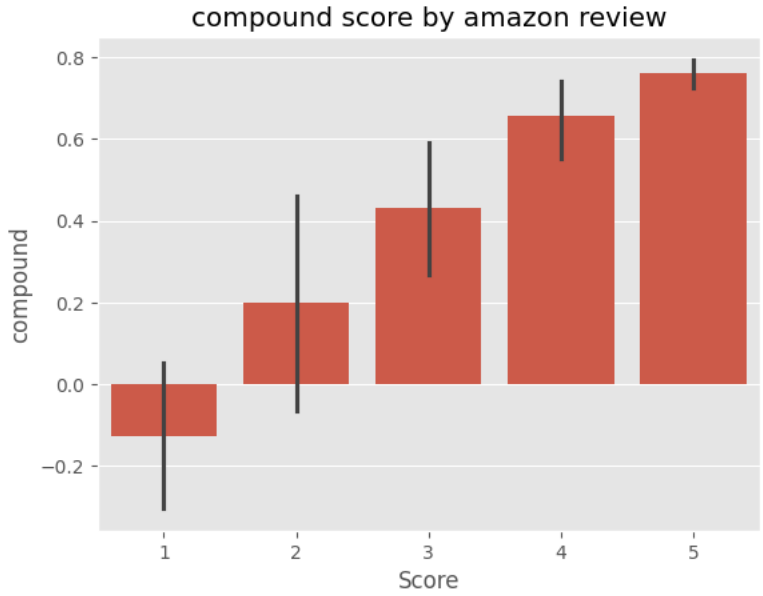
```
vaders = vaders.reset_index().rename(columns={'index':'Id'})
```

```
vaders =  vaders.merge(df,how="left")
```

```
vaders.head()
```

	Id	neg	neu	pos	compound	ProductId	UserId	ProfileName	HelpfulnessNumera
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	

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



##THis shows more higher compound score has better reviews , we can say our model did good

Extra is Transformer pipeline from Huggingface

```
from transformers import pipeline
```

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

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english and revision a
Using a pipeline without specifying a model name and revision in production is not recommended.
config.json: 100%  629/629 [00:00<00:00, 16.6kB/s]
model.safetensors: 100%  268M/268M [00:00<00:00, 315MB/s]
tokenizer_config.json: 100%  48.0/48.0 [00:00<00:00, 2.65kB/s]
vocab.txt: 100%  232k/232k [00:00<00:00, 489kB/s]
```

```
sent_pipeline("i love this product")
```

```
[{'label': 'POSITIVE', 'score': 0.9998788833618164}]
```

Start coding or generate with AI.