Feedback Analysis Project

This project aims to train and prepare a "feedback analysis model" to be used at the time that we would have the "feedback" feature for any of our products.

As currently we don't have a "Feedback" feature in our products, so I will be using an "Amazon food reviews and rating" dataset in csv format (which are text reviews and rating the food out of five stars)

- *The Vader Model is developed*

- *Ran NLTK work tokenizer to splits the feedbacks*

- *The required Tokens are generated*

- *Sentiment scores are generated and added to the current dataset*

The Python has been used to create the Model:

- **the plot we can seFrome that most of the reviews are 5 stars**

```
In [83]:  # Test 2
          sia.polarity_scores('This is the worst thing ever.')

Out[83]:  {'neg': 0.451, 'neu': 0.549, 'pos': 0.0, 'compound': -0.6249}

In [4]:   sia.polarity_scores('This is the best thing ever.')

Out[4]:   {'neg': 0.0, 'neu': 0.543, 'pos': 0.457, 'compound': 0.6369}
```

Below indicates the project progress steps and the whole Python code:

- *The required libraries imported:*

File   Edit   View   Insert   Cell   Kernel   Widgets   Help   Not Trusted | Python 3 (ipykernel)

```python
In [ ]:  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()

         showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
```

```python
In [ ]:  import nltk
         nltk.download('maxent_ne_chunker')
```

```python
In [ ]:  import nltk
         nltk.download('words')
```

```python
In [28]: import nltk
         nltk.download('vader_lexicon')

         [nltk_data] Downloading package vader_lexicon to
         [nltk_data]     C:\Users\ella\AppData\Roaming\nltk_data...
Out[28]: True
```

- *The data set imported*

```python
In [17]:  # Read in data
          feedback = pd.read_csv('Reviews.csv')
          feedback.head()
```

Out[17]:

	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...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

```python
In [ ]:  feedback.shape
```

```python
In [ ]:  feedback.shape
```

```python
In [18]: feedback = feedback.head(500)
         feedback.head(5)
```

Out[18]:

	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...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

- **Ran a Quick EDA to get an idea of what the data set looks like**

Value count on Score column to see the number of times each score occurs

```
feedback['Score'].value_counts().sort_index()
```

```
Out[19]: 1      36
         2      18
         3      37
         4      70
         5     339
         Name: Score, dtype: int64
```

```
In [20]: ax = feedback['Score'].value_counts().sort_index() \
             .plot(kind='bar',
                   title='Count of Reviews by Stars',
                   figsize=(10, 5))
         ax.set_xlabel('Review Stars')
         plt.show()
```

Count of Reviews by Stars

From the plot we can see that most of the reviews are 5 stars

- **Running NLTK work tokenizer to splits the sentence into the parts of each word in the sentence**

```
ex = feedback['Text'][50]
print(ex)
```

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

the result came back negative

```
tokens = nltk.word_tokenize(ex)
tokens
```

```
['This',
 'oatmeal',
 'is',
 'not',
 'good',
 '.',
 'Its',
 'mushy',
 ',',
 'soft',
 ',',
 'I',
 'do',
 "n't",
 'like',
 'it',
 '.',
 'Quaker',
 'Oats',
 'is',
 'the',
 'way',
 'to',
 'go',
 '.']
```

- Running the nltk pos tag for part of speech tagging

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

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

- ***Grouping the Tokens into chunks of text***

```
entities = nltk.chunk.ne_chunk(speech_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
  ./.)
```

Implementing the Vader Model:

Vader Sentiment Scoring

VADER(Valence Aware Dictionary for Sentiment Reasoning) is an NLTK module that provides sentiment scores based on the words used. It is a rule-based sentiment analyzer in which the terms are generally labeled as per their semantic orientation as either positive or negative.

VADER has the advantage of assessing the sentiment of any given text without the need for previous training as we might have to for Machine Learning models. The result generated by VADER is a dictionary of 4 keys neg, neu, pos and compound: neg, neu, and pos meaning negative, neutral, and positive respectively.

This module uses a "bag of words" approach:

- *Stop words are removed*
- *Each word scored and combined to a total score*
- *This model is not account for relationship between words*

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

#sia is an created object of "Sentiment Intensity Analyzer"
sia = SentimentIntensityAnalyzer()
```

Testing our object

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

The result is: negative as zero, neutral point = 0.318 and positive point= 0.682, So this sentence is mostly positive

```
# Test 2
sia.polarity_scores('This is the worst thing ever.')
```
{'neg': 0.451, 'neu': 0.549, 'pos': 0.0, 'compound': -0.6249}

```
sia.polarity_scores('This is the best thing ever.')
```
{'neg': 0.0, 'neu': 0.543, 'pos': 0.457, 'compound': 0.6369}

The result is: negative point = 0.451, neutral point = 0.549 and positive point= 0 and compound: -0.6249 So this sentence is mostly Negative

```
# Running the sia on our tokens
sia.polarity_scores(ex)
```
{'neg': 0.22, 'neu': 0.78, 'pos': 0.0, 'compound': -0.5448}

The result is: negative point = 0.22, neutral point = 0.78 and positive point= 0 and compound: -0.5448 So this sentence is mostly Negative

Run the polarity score on the entire dataset, the result is a dictionary which stores the result of the below loop which each raw would

```
result = {}
for i, row in tqdm(feedback.iterrows(), total=len(feedback)):
    text = row['Text']
    myid = row['Id']
    result[myid] = sia.polarity_scores(text)
```
```
  0%|          | 0/500 [00:00<?, ?it/s]
```

Here shows the result dictionary with each id that contain the scores of neg, neu, pos and compound which stored in a Panda data frame to make it easier to work with and displayed horizontally

```
pd.DataFrame(result).T
```

	neg	neu	pos	compound
1	0.000	0.695	0.305	0.9441
2	0.138	0.862	0.000	-0.5664
3	0.091	0.754	0.155	0.8265
4	0.000	1.000	0.000	0.0000
5	0.000	0.552	0.448	0.9468
...	...	...	...	...
496	0.000	0.554	0.446	0.9725
497	0.059	0.799	0.142	0.7833
498	0.025	0.762	0.212	0.9848
499	0.041	0.904	0.055	0.1280
500	0.000	0.678	0.322	0.9811

500 rows × 4 columns

So now we have a data frame that has index which is the IDs and the four neg, neu, pos and compound fields, we call this result Vaders

```
#vaders.reset_index().rename(columns={'index': 'Id'})
vaders["Id"]= feedback["Id"]
#vaders.merge(feedback, how='left')
#vaders.columns
```

```
vaders.columns
```

```
Index(['neg', 'neu', 'pos', 'compound', 'Id'], dtype='object')
```

```
vaders=vaders.merge(feedback, how='left')
```

Now we have sentiment score added to the original fields

```
vaders.head()
```

	neg	neu	pos	compound	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time
0	0.000	0.695	0.305	0.9441	2.0	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0.0	0.0	1.0	1.346976e+09
1	0.138	0.862	0.000	-0.5664	3.0	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1.0	1.0	4.0	1.219018e+09
2	0.091	0.754	0.155	0.8265	4.0	B000UA0QIQ	A395BORC6FGVXV	Karl	3.0	3.0	2.0	1.307923e+09
3	0.000	1.000	0.000	0.0000	5.0	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0.0	0.0	5.0	1.350778e+09
4	0.000	0.552	0.448	0.9468	6.0	B006K2ZZ7K	ADT0SRK1MGOEU	Twoapennything	0.0	0.0	4.0	1.342051e+09

Plot VADER results

Runing the Plot on vaders' data by assigning x the score valuse which is the star review of the the person and then compound is going to be our y value and that's the negative to positive.

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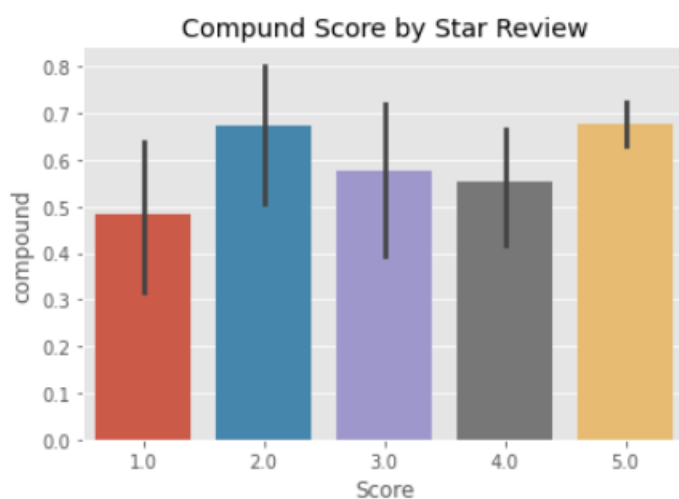

The plot shows that one star review has lower compound score and the five star view is higher

Running the Plot based on the positive neutral and negative scores

```
fig, axs = plt.subplots(1, 3, figsize=(12, 3))
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])
axs[0].set_title('Positive')
axs[1].set_title('Neutral')
axs[2].set_title('Negative')
plt.tight_layout()
plt.show()
```
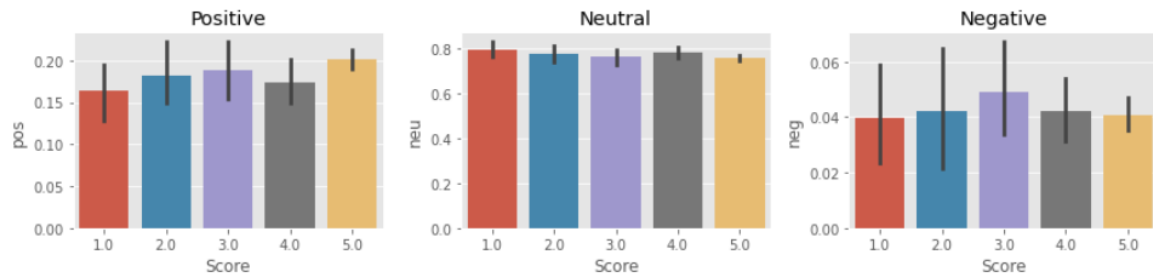

The plots confirm that our Vader Model is valuable in having the connection between the score of the text and sentiment score and it does relate to the actual rating review of the reviewers