Sentimental

April 23, 2024

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from google.colab import drive
plt.style.use('ggplot')
```

[2]:
```
# Mount Google Drive
drive.mount('/content/drive')
```

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

[3]:
```
# Read in data

dataset = r'/content/drive/My Drive/ML/NLP/Sentimental_analysis/Reviews.csv'
df = pd.read_csv(dataset)
df = df.head(500)
print(df.shape)
```

```
(500, 10)
```

[4]:
```
df.head()
```

[4]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600

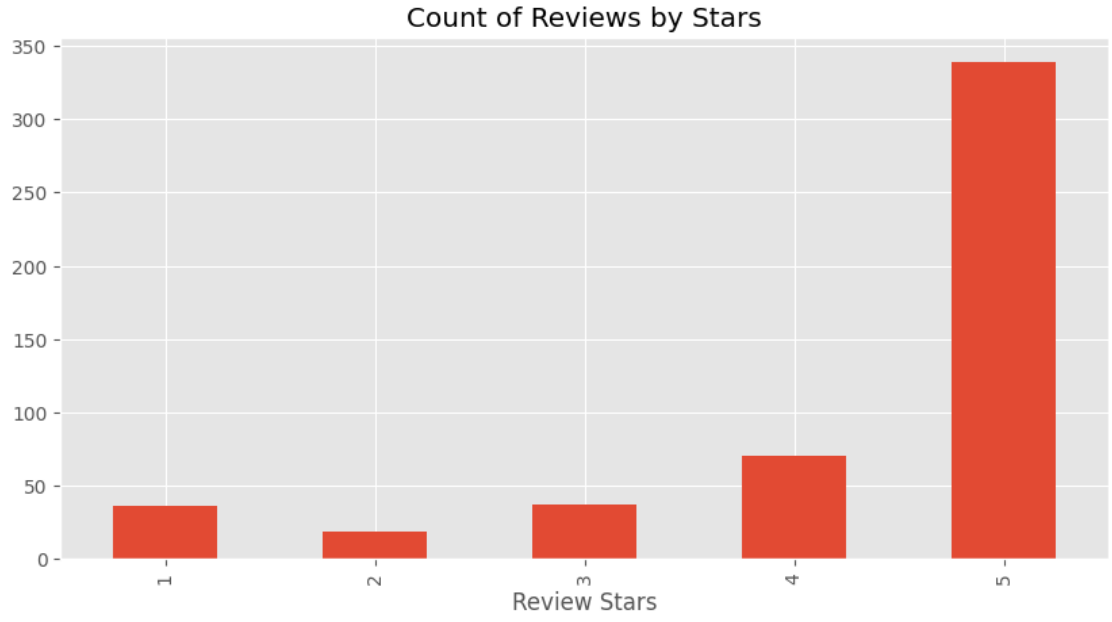
```
                 Summary                                                Text
0  Good Quality Dog Food  I have bought several of the Vitality canned d…
1      Not as Advertised  Product arrived labeled as Jumbo Salted Peanut…
2  "Delight" says it all  This is a confection that has been around a fe…
3         Cough Medicine  If you are looking for the secret ingredient i…
4            Great taffy  Great taffy at a great price.  There was a wid…
```

[5]:
```
#checking the Text column
df['Text'].values[2]
```

[5]:
```
'This is a confection that has been around a few centuries.  It is a light,
pillowy citrus gelatin with nuts - in this case Filberts. And it is cut into
tiny squares and then liberally coated with powdered sugar.  And it is a tiny
mouthful of heaven.  Not too chewy, and very flavorful.  I highly recommend this
yummy treat.  If you are familiar with the story of C.S. Lewis\' "The Lion, The
Witch, and The Wardrobe" - this is the treat that seduces Edmund into selling
out his Brother and Sisters to the Witch.'
```

##EDA

[6]:
```
#plot the frequency of the lables in the data set
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()
```



##Basic NLTK

[7]:
```
example = df['Text'][10]
print(example)
```

```
I don't know if it's the cactus or the tequila or just the unique combination of
ingredients, but the flavour of this hot sauce makes it one of a kind!  We
picked up a bottle once on a trip we were on and brought it back home with us
and were totally blown away!  When we realized that we simply couldn't find it
anywhere in our city we were bummed.<br /><br />Now, because of the magic of the
internet, we have a case of the sauce and are ecstatic because of it.<br /><br
/>If you love hot sauce..I mean really love hot sauce, but don't want a sauce
that tastelessly burns your throat, grab a bottle of Tequila Picante Gourmet de
Inclan.  Just realize that once you taste it, you will never want to use any
other sauce.<br /><br />Thank you for the personal, incredible service!
```

[8]:
```
#download additional resources
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to /root/nltk_data…
[nltk_data]   Unzipping tokenizers/punkt.zip.
```

[8]:
```
True
```

[9]:
```
#tokenize the sentence
tokens = nltk.word_tokenize(example)
tokens[:10]
```

[9]:
```
['I', 'do', "n't", 'know', 'if', 'it', "'s", 'the', 'cactus', 'or']
```

[10]:
```
#download additional resources
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.
```

[10]:
```
True
```

[11]:
```
#view the word tags (Type of the words)
tagged = nltk.pos_tag(tokens)
tagged[:10]
```

[11]:
```
[('I', 'PRP'),
 ('do', 'VBP'),
 ("n't", 'RB'),
 ('know', 'VB'),
 ('if', 'IN'),
```

```
 ('it', 'PRP'),
 ("'s", 'VBZ'),
 ('the', 'DT'),
 ('cactus', 'NN'),
 ('or', 'CC')]
```

[12]:
```
nltk.download('maxent_ne_chunker')
nltk.download('words')
```

```
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     /root/nltk_data…
[nltk_data]   Unzipping chunkers/maxent_ne_chunker.zip.
[nltk_data] Downloading package words to /root/nltk_data…
[nltk_data]   Unzipping corpora/words.zip.
```

[12]:
```
True
```

[13]:
```
entities = nltk.chunk.ne_chunk(tagged)
entities.pprint()
```

```
(S
  I/PRP
  do/VBP
  n't/RB
  know/VB
  if/IN
  it/PRP
  's/VBZ
  the/DT
  cactus/NN
  or/CC
  the/DT
  tequila/NN
  or/CC
  just/RB
  the/DT
  unique/JJ
  combination/NN
  of/IN
  ingredients/NNS
  ,/,
  but/CC
  the/DT
  flavour/NN
  of/IN
  this/DT
  hot/JJ
  sauce/NN
```

makes/VBZ
it/PRP
one/CD
of/IN
a/DT
kind/NN
!/.
We/PRP
picked/VBD
up/RP
a/DT
bottle/NN
once/RB
on/IN
a/DT
trip/NN
we/PRP
were/VBD
on/IN
and/CC
brought/VBD
it/PRP
back/RP
home/NN
with/IN
us/PRP
and/CC
were/VBD
totally/RB
blown/VBN
away/RB
!/.
When/WRB
we/PRP
realized/VBD
that/IN
we/PRP
simply/RB
could/MD
n't/RB
find/VB
it/PRP
anywhere/RB
in/IN
our/PRP$
city/NN
we/PRP
were/VBD

bummed./JJ
</NNP
br/NN
//NNP
>/NNP
</NNP
br/NN
//NNP
>/NNP
Now/RB
,/,
because/IN
of/IN
the/DT
magic/NN
of/IN
the/DT
internet/NN
,/,
we/PRP
have/VBP
a/DT
case/NN
of/IN
the/DT
sauce/NN
and/CC
are/VBP
ecstatic/JJ
because/IN
of/IN
it./JJ
</NNP
br/NN
//NNP
>/NNP
</NNP
br/NN
//NNP
>/NNP
If/IN
you/PRP
love/VBP
hot/JJ
sauce/NN
../NN
I/PRP
mean/VBP

```
really/RB
love/VB
hot/JJ
sauce/NN
,/,
but/CC
do/VBP
n't/RB
want/VB
a/DT
sauce/NN
that/WDT
tastelessly/RB
burns/VBZ
your/PRP$
throat/NN
,/,
grab/VB
a/DT
bottle/NN
of/IN
(PERSON Tequila/NNP Picante/NNP Gourmet/NNP)
de/FW
(PERSON Inclan/NNP)
./.
Just/NNP
realize/VB
that/DT
once/RB
you/PRP
taste/VBP
it/PRP
,/,
you/PRP
will/MD
never/RB
want/VB
to/TO
use/VB
any/DT
other/JJ
sauce./NN
</NNP
br/NN
//NNP
>/NNP
</NNP
br/NN
```

```
//NNP
>/NNP
Thank/NNP
you/PRP
for/IN
the/DT
personal/JJ
,/,
incredible/JJ
service/NN
!/.)
```

#Step 1. VADER Seniment Scoring

We will use NLTK's SentimentIntensityAnalyzer to get the neg/neu/pos scores of the text.

This uses a "bag of words" approach:

- Stop words are removed
- each word is scored and combined to a total score

[14]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
nltk.download('vader_lexicon')
sia = SentimentIntensityAnalyzer()
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data…
```

[15]:
```
sia.polarity_scores('i love you')
```

[15]:
```
{'neg': 0.0, 'neu': 0.192, 'pos': 0.808, 'compound': 0.6369}
```

[16]:
```
sia.polarity_scores('This is the worst thing ever.')
```

[16]:
```
{'neg': 0.451, 'neu': 0.549, 'pos': 0.0, 'compound': -0.6249}
```

[17]:
```
example
```

[17]:
```
"I don't know if it's the cactus or the tequila or just the unique combination
of ingredients, but the flavour of this hot sauce makes it one of a kind!  We
picked up a bottle once on a trip we were on and brought it back home with us
and were totally blown away!  When we realized that we simply couldn't find it
anywhere in our city we were bummed.<br /><br />Now, because of the magic of the
internet, we have a case of the sauce and are ecstatic because of it.<br /><br
/>If you love hot sauce..I mean really love hot sauce, but don't want a sauce
that tastelessly burns your throat, grab a bottle of Tequila Picante Gourmet de
Inclan.  Just realize that once you taste it, you will never want to use any
other sauce.<br /><br />Thank you for the personal, incredible service!"
```

[18]:
```
sia.polarity_scores(example)
```

[18]:
```
{'neg': 0.017, 'neu': 0.846, 'pos': 0.137, 'compound': 0.9746}
```

[19]:
```
# 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)
```

```
  0%|          | 0/500 [00:00<?, ?it/s]
```

[20]:
```
vaders = pd.DataFrame(res).T
vaders = vaders.reset_index().rename(columns={'index': 'Id'})
vaders = vaders.merge(df, how='left')
```

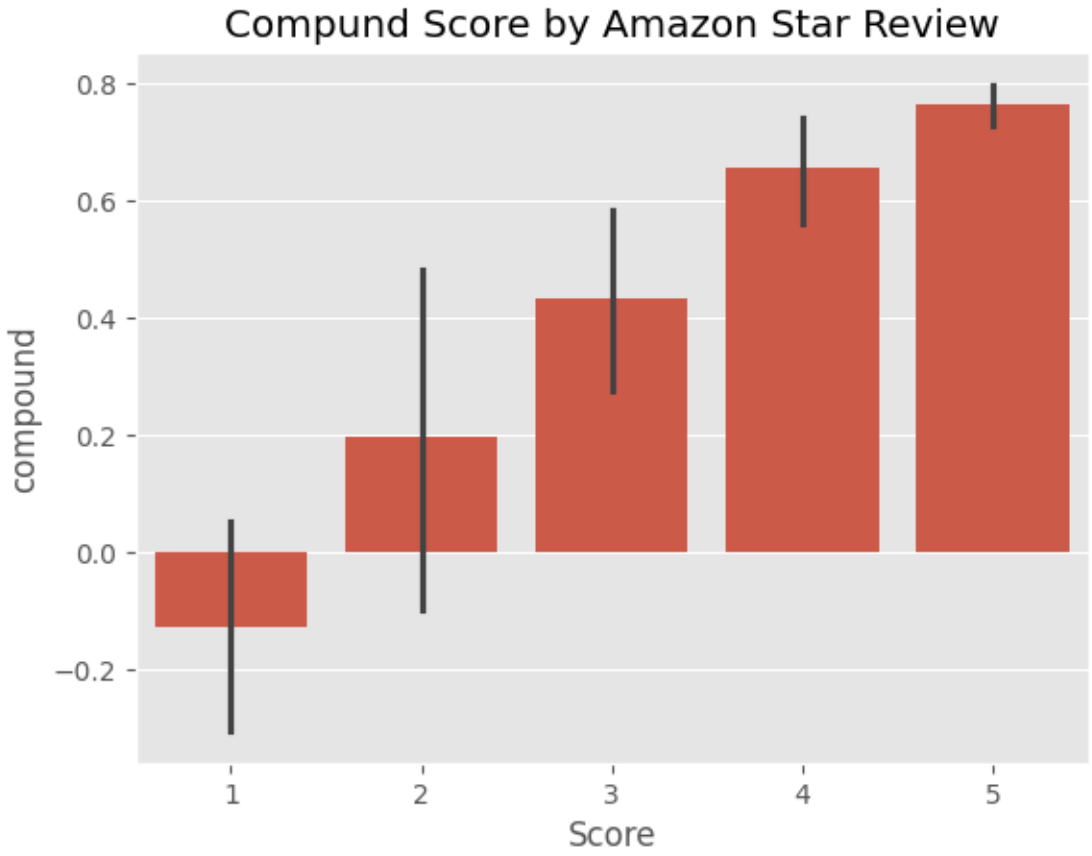
[21]:
```
# Now we have sentiment score and metadata
vaders.head()
```

[21]:

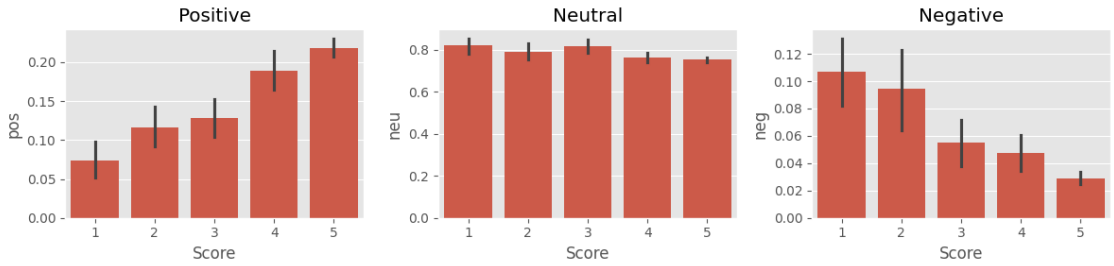
	Id	neg	neu	pos	compound	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	0.000	0.695	0.305	0.9441	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d…
1	2	0.138	0.862	0.000	-0.5664	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut…
2	3	0.091	0.754	0.155	0.8265	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	0.000	1.000	0.000	0.0000	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i…
4	5	0.000	0.552	0.448	0.9468	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid…

##Plot VADER results

[22]:
```
ax = sns.barplot(data=vaders, x='Score', y='compound')
ax.set_title('Compund Score by Amazon Star Review')
plt.show()
```



[23]:
```
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()
```



##Step 3. Roberta Pretrained Model

- Use a model trained of a large corpus of data.
- Transformer model accounts for the words but also the context related to other words.

[24]:
```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
```

[25]:
```
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/_token.py:88:
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 secret in your Google Colab
and restart your session.
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:   0%|          | 0.00/747 [00:00<?, ?B/s]
vocab.json:   0%|          | 0.00/899k [00:00<?, ?B/s]
merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]
special_tokens_map.json:   0%|          | 0.00/150 [00:00<?, ?B/s]
pytorch_model.bin:   0%|          | 0.00/499M [00:00<?, ?B/s]
```

[26]:
```
# VADER results on example
print(example)
sia.polarity_scores(example)
```

```
I don't know if it's the cactus or the tequila or just the unique combination of
```

```
ingredients, but the flavour of this hot sauce makes it one of a kind!  We
picked up a bottle once on a trip we were on and brought it back home with us
and were totally blown away!  When we realized that we simply couldn't find it
anywhere in our city we were bummed.<br /><br />Now, because of the magic of the
internet, we have a case of the sauce and are ecstatic because of it.<br /><br
/>If you love hot sauce..I mean really love hot sauce, but don't want a sauce
that tastelessly burns your throat, grab a bottle of Tequila Picante Gourmet de
Inclan.  Just realize that once you taste it, you will never want to use any
other sauce.<br /><br />Thank you for the personal, incredible service!
```

[26]:
```
{'neg': 0.017, 'neu': 0.846, 'pos': 0.137, 'compound': 0.9746}
```

[27]:
```
# 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': 0.019134127, 'roberta_neu': 0.0710445, 'roberta_pos': 0.9098214}
```

[28]:
```
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
```

[29]:
```
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}')
```

```
  0%|          | 0/500 [00:00<?, ?it/s]
```

```
Broke for id 83
Broke for id 187
```

[30]:
```
results_df = pd.DataFrame(res).T
results_df = results_df.reset_index().rename(columns={'index': 'Id'})
results_df = results_df.merge(df, how='left')
```

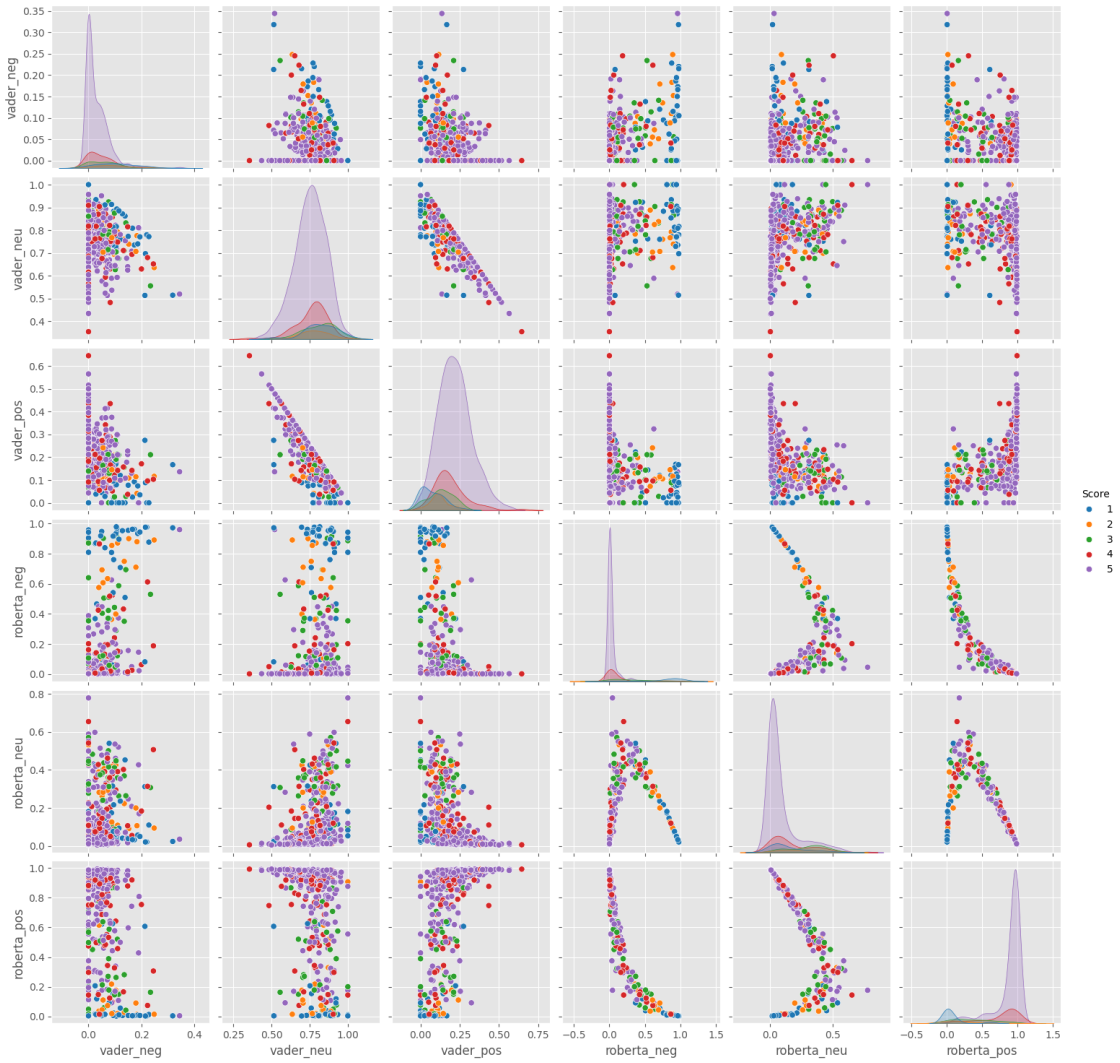
##Compare Scores between models

[31]:
```
results_df.columns
```

[31]:
```
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')
```

##Step 3. Combine and compare

[32]:
```
sns.pairplot(data=results_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                   'roberta_neg', 'roberta_neu', 'roberta_pos'],
             hue='Score',
             palette='tab10')
plt.show()
```



##Step 4: Review Examples:

- Positive 1-Star and Negative 5-Star Reviews

Lets look at some examples where the model scoring and review score differ the most.

[33]:
```
results_df.query('Score == 1') \
    .sort_values('roberta_pos', ascending=False)['Text'].values[0]
```

[33]:
```
'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.'
```

[34]:
```
results_df.query('Score == 1') \
    .sort_values('vader_pos', ascending=False)['Text'].values[0]
```

[34]:
```
'So we cancelled the order.  It was cancelled without any problem.  That is a positive note…'
```

[35]:
```
results_df.query('Score == 5') \
    .sort_values('roberta_neg', ascending=False)['Text'].values[0]
```

[35]:
```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my fault'
```

[36]:
```
results_df.query('Score == 5') \
    .sort_values('vader_neg', ascending=False)['Text'].values[0]
```

[36]:
```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my fault'
```

##The Transformers Pipeline Quick & easy way to run sentiment predictions

[37]:
```
from transformers import pipeline

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

```
No model was supplied, defaulted to distilbert/distilbert-base-uncased-finetuned-sst-2-english and revision af0f99b (https://huggingface.co/distilbert/distilbert-base-uncased-finetuned-sst-2-english).
Using a pipeline without specifying a model name and revision in production is not recommended.
```

```
config.json:   0%|          | 0.00/629 [00:00<?, ?B/s]
```

```
model.safetensors:   0%|          | 0.00/268M [00:00<?, ?B/s]
```

```
tokenizer_config.json:   0%|          | 0.00/48.0 [00:00<?, ?B/s]
```

```
vocab.txt:   0%|          | 0.00/232k [00:00<?, ?B/s]
```

[38]:
```
sent_pipeline('I love sentiment analysis!')
```

[38]:
```
[{'label': 'POSITIVE', 'score': 0.9997853636741638}]
```

[39]:
```
sent_pipeline('I love python')
```

[39]:
```
[{'label': 'POSITIVE', 'score': 0.9997324347496033}]
```

[40]:
```
sent_pipeline('booo')
```

[40]:
```
[{'label': 'NEGATIVE', 'score': 0.9936267137527466}]
```

##The End