Sentiment Analysis

April 24, 2025

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import nltk

plt.style.use('ggplot')
```

[40]:
```
import nltk
nltk.download('punkt')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\essie\AppData\Roaming\nltk_data…
[nltk_data]   Unzipping tokenizers\punkt.zip.
```

[40]: True

[46]:
```
nltk.download('averaged_perceptron_tagger')
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\essie\AppData\Roaming\nltk_data…
[nltk_data]   Unzipping taggers\averaged_perceptron_tagger.zip.
```

[46]: True

[50]:
```
nltk.download('maxent_ne_chunker')
```

```
[nltk_data] Downloading package maxent_ne_chunker to
[nltk_data]     C:\Users\essie\AppData\Roaming\nltk_data…
[nltk_data]   Unzipping chunkers\maxent_ne_chunker.zip.
```

[50]: True

[52]:
```
nltk.download('words')
```

```
[nltk_data] Downloading package words to
[nltk_data]     C:\Users\essie\AppData\Roaming\nltk_data…
[nltk_data]   Unzipping corpora\words.zip.
```

[52]: True

[21]:
```
#Read in the data
df = pd.read_csv(r"C:\Users\essie\Desktop\data analysis␣
↪projects\Python\uber_reviews_without_reviewid.csv")
```

[22]:
```
df.head()
```

[22]:

	userName	userImage	content	score	thumbsUpCount	reviewCreatedVersion	at	replyContent	repliedAt	appVersion
0	User_0	NaN	Good	5	0	4.556.10005	2024-12-18 17:17:19	NaN	NaN	4.556.10005
1	User_1	NaN	Nice	5	0	4.556.10005	2024-12-18 17:17:17	NaN	NaN	4.556.10005
2	User_2	NaN	Very convenient	5	0	4.532.10001	2024-12-18 17:09:42	NaN	NaN	4.532.10001
3	User_3	NaN	Good	4	0	4.556.10005	2024-12-18 17:08:27	NaN	NaN	4.556.10005
4	User_4	NaN	exllence	5	0	4.556.10005	2024-12-18 17:08:16	NaN	NaN	4.556.10005

[23]:
```
print(df.shape)
```

```
(12000, 10)
```

[24]:
```
#we can decide to reduce the number of rows
#df = df.head(700)
```

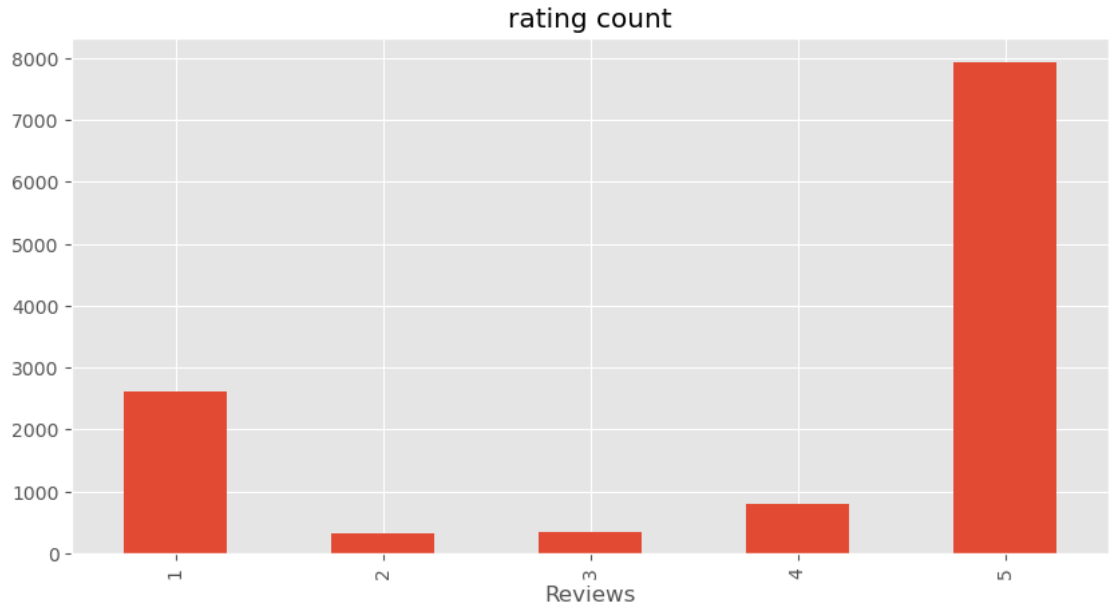
Simple EDA

[27]:
```
df['score'].value_counts()
```

[27]:
```
score
5    7926
1    2618
4     806
3     333
2     317
Name: count, dtype: int64
```

[31]:
```
#lets plot and see how it looks
```

```
x =df['score'].value_counts().sort_index().plot(kind='bar',title = 'rating␣
 ↪count', figsize= (10,5))
x.set_xlabel('Reviews')
plt.show()
```



Basic Nltk operations

[36]:
```
# lets have an idea of how nltk works
example = df['content'][500]
print(example)
```

```
Awesome
```

[44]:
```
#lets tokonize the example, spliting the part of each word in a sentence
tokens = nltk.word_tokenize(example)
tokens[:2]
```

[44]:
```
['Awesome', '  ']
```

[48]:
```
#nltk can also find a part of speech for the word
tagged=nltk.pos_tag(tokens)
tagged[:2]
```

[48]:
```
[('Awesome', 'NNP'), ('  ', 'NN')]
```

[53]:
```
#grouping into chunk of text, must be saved and pprint means pretty print
entities = nltk.chunk.ne_chunk(tagged)
entities.pprint()
```

```
(S (GPE Awesome/NNP)  /NN)
```

Veader sentiment score, Nltk sentimentInternsityAnalyzer

[56]:
```
# sentiment intensityAnalyzer is use to get negative, neutral, and positve text.
  ↪tqdm is to keep track of progress on the notebook
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
```

[59]:
```
Analyzer = SentimentIntensityAnalyzer()
```

[60]:
```
Analyzer
```

[60]:
```
<nltk.sentiment.vader.SentimentIntensityAnalyzer at 0x1bd37a63b10>
```

[69]:
```
#Running the polarity Score for the entire dataset
store = {}
for i, row in tqdm(df.iterrows(), total = len(df)):
    text = row['content']
    my_id = row['userName']
    store[my_id] = Analyzer.polarity_scores(text)
```

```
  0%|          | 0/12000 [00:00<?, ?it/s]
```

[96]:
```
veder_score =pd.DataFrame(store).T
```

[98]:
```
# Drop any existing unique_id columns to avoid duplicates
if 'unique_id' in veder_score.columns:
    veder_score = veder_score.drop(columns='unique_id')
if 'unique_id' in df.columns:
    df = df.drop(columns='unique_id')

# Reset index to create a unique_id column
veder_score = veder_score.reset_index().rename(columns={'index': 'unique_id'})
df = df.reset_index().rename(columns={'index': 'unique_id'})

# Select only the required columns from veder_score
veder_score = veder_score[['unique_id', 'neg', 'neu', 'pos', 'compound']]

# Merge the DataFrames using unique_id
merged_df = df.merge(veder_score, how='left', on='unique_id')

# Drop the unique_id column from the final DataFrame
merged_df = merged_df.drop(columns='unique_id')

# Verify the resulting columns
print("Merged DataFrame columns:", merged_df.columns)
```

```
Merged DataFrame columns: Index(['userName', 'userImage', 'content', 'score',
```

```
      'thumbsUpCount',
             'reviewCreatedVersion', 'at', 'replyContent', 'repliedAt', 'appVersion',
             'neg', 'neu', 'pos', 'compound'],
            dtype='object')
```

[100]:
```
merged_df
```

[100]:
```
          userName  userImage  \
0          User_0        NaN
1          User_1        NaN
2          User_2        NaN
3          User_3        NaN
4          User_4        NaN
…            …         …
11995  User_11995        NaN
11996  User_11996        NaN
11997  User_11997        NaN
11998  User_11998        NaN
11999  User_11999        NaN

                                                content  score  \
0                                                  Good      5
1                                                  Nice      5
2                                       Very convenient      5
3                                                  Good      4
4                                              exllence      5
…                                                    …    …
11995                                       Excellent!!!      5
11996  Worst experience after 10pm in Hyde cityno aut…      5
11997                                        Exceptional      5
11998                                      Good Service.      5
11999  Very bad experience with this app, booked a sh…      1

       thumbsUpCount reviewCreatedVersion                   at replyContent  \
0                  0          4.556.10005  2024-12-18 17:17:19          NaN
1                  0          4.556.10005  2024-12-18 17:17:17          NaN
2                  0          4.532.10001  2024-12-18 17:09:42          NaN
3                  0          4.556.10005  2024-12-18 17:08:27          NaN
4                  0          4.556.10005  2024-12-18 17:08:16          NaN
…                …                    …                    …          …
11995              0          4.553.10000  2024-11-24 21:59:16          NaN
11996              0          4.552.10000  2024-11-24 21:56:10          NaN
11997              0          4.552.10000  2024-11-24 21:52:21          NaN
11998              0          4.553.10000  2024-11-24 21:50:30          NaN
11999              0                  NaN  2024-11-24 21:44:44          NaN

      repliedAt   appVersion    neg    neu    pos  compound
```

```
0           NaN  4.556.10005  0.000  0.000  1.000    0.4404
1           NaN  4.556.10005  0.000  0.000  1.000    0.4215
2           NaN  4.532.10001  0.000  1.000  0.000    0.0000
3           NaN  4.556.10005  0.000  0.000  1.000    0.4404
4           NaN  4.556.10005  0.000  1.000  0.000    0.0000
…           …          …     …      …      …        …
11995       NaN  4.553.10000  0.000  0.000  1.000    0.6784
11996       NaN  4.552.10000  0.291  0.709  0.000   -0.6249
11997       NaN  4.552.10000  0.000  1.000  0.000    0.0000
11998       NaN  4.553.10000  0.000  0.256  0.744    0.4404
11999       NaN          NaN  0.350  0.576  0.074   -0.9000

[12000 rows x 14 columns]
```

[101]:
```
print(merged_df.shape)
```

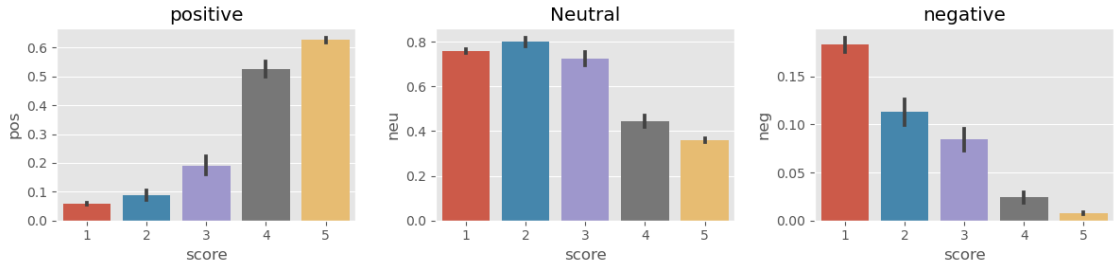
```
(12000, 14)
```

[109]:
```
# lets plot the vader_score

Xyz= sns.barplot(data=merged_df, x = 'score', y = 'compound')
Xyz.set_title('Compound score of Uber reviews')
plt.show()
```

Comparing all (negative, Positve and neutral)

[118]:
```
fig,axs = plt.subplots(1,3, figsize=(12,3))
sns.barplot(data=merged_df, x ='score', y= 'pos', ax = axs[0])
sns.barplot(data=merged_df, x= 'score', y = 'neu', ax = axs[1])
sns.barplot(data=merged_df, 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()# to Avoid overlap
plt.show()
```



To use a trianed model,Transformer model account for the words but also the context related to other words.

[120]:
```
!pip install transformers
```

```
Requirement already satisfied: transformers in
c:\users\essie\anaconda3\lib\site-packages (4.32.1)
Requirement already satisfied: filelock in c:\users\essie\anaconda3\lib\site-
packages (from transformers) (3.9.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.15.1 in
c:\users\essie\anaconda3\lib\site-packages (from transformers) (0.15.1)
Requirement already satisfied: numpy>=1.17 in c:\users\essie\anaconda3\lib\site-
packages (from transformers) (1.24.3)
Requirement already satisfied: packaging>=20.0 in
c:\users\essie\anaconda3\lib\site-packages (from transformers) (23.1)
Requirement already satisfied: pyyaml>=5.1 in c:\users\essie\anaconda3\lib\site-
packages (from transformers) (6.0)
Requirement already satisfied: regex!=2019.12.17 in
c:\users\essie\anaconda3\lib\site-packages (from transformers) (2022.7.9)
Requirement already satisfied: requests in c:\users\essie\anaconda3\lib\site-
packages (from transformers) (2.31.0)
Requirement already satisfied: tokenizers!=0.11.3,<0.14,>=0.11.1 in
c:\users\essie\anaconda3\lib\site-packages (from transformers) (0.13.2)
```

```
Requirement already satisfied: safetensors>=0.3.1 in
c:\users\essie\anaconda3\lib\site-packages (from transformers) (0.3.2)
Requirement already satisfied: tqdm>=4.27 in c:\users\essie\anaconda3\lib\site-
packages (from transformers) (4.65.0)
Requirement already satisfied: fsspec in c:\users\essie\anaconda3\lib\site-
packages (from huggingface-hub<1.0,>=0.15.1->transformers) (2023.4.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
c:\users\essie\anaconda3\lib\site-packages (from huggingface-
hub<1.0,>=0.15.1->transformers) (4.12.2)
Requirement already satisfied: colorama in c:\users\essie\anaconda3\lib\site-
packages (from tqdm>=4.27->transformers) (0.4.6)
Requirement already satisfied: charset-normalizer<4,>=2 in
c:\users\essie\anaconda3\lib\site-packages (from requests->transformers) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in
c:\users\essie\anaconda3\lib\site-packages (from requests->transformers) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in
c:\users\essie\anaconda3\lib\site-packages (from requests->transformers)
(1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in
c:\users\essie\anaconda3\lib\site-packages (from requests->transformers)
(2024.2.2)
```

[121]:
```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax #to smooth out between 0 and 1
```

the model below has already been trained on a bunch of data based on sentiment (roberta Model)

[122]:
```
MODEL = f"cardiffnlp/twitter-roberta-base-sentiment"
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
```

```
Downloading config.json:   0%|          | 0.00/747 [00:00<?, ?B/s]

C:\Users\essie\anaconda3\Lib\site-packages\huggingface_hub\file_download.py:133:
UserWarning: `huggingface_hub` cache-system uses symlinks by default to
efficiently store duplicated files but your machine does not support them in
C:\Users\essie\.cache\huggingface\hub. Caching files will still work but in a
degraded version that might require more space on your disk. This warning can be
disabled by setting the `HF_HUB_DISABLE_SYMLINKS_WARNING` environment variable.
For more details, see https://huggingface.co/docs/huggingface_hub/how-to-
cache#limitations.
To support symlinks on Windows, you either need to activate Developer Mode or to
run Python as an administrator. In order to see activate developer mode, see
this article: https://docs.microsoft.com/en-us/windows/apps/get-started/enable-
your-device-for-development
  warnings.warn(message)

Downloading vocab.json:   0%|          | 0.00/899k [00:00<?, ?B/s]
```

```
Downloading merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]
```

```
Downloading (…)cial_tokens_map.json:   0%|          | 0.00/150 [00:00<?, ?B/s]
```

```
C:\Users\essie\anaconda3\Lib\site-packages\transformers\utils\generic.py:260:
FutureWarning: `torch.utils._pytree._register_pytree_node` is deprecated. Please
use `torch.utils._pytree.register_pytree_node` instead.
  torch.utils._pytree._register_pytree_node(
```

```
Downloading pytorch_model.bin:   0%|          | 0.00/499M [00:00<?, ?B/s]
```

between roberta and Veder sentiment scoring system

[124]:
```
#Running with roberta
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)
```

```
[0.00291451 0.0322172  0.96486825]
```

[126]:
```
# VADER results on example
print(example)
Analyzer.polarity_scores(example)
```

```
Awesome
```

[126]:
```
{'neg': 0.0, 'neu': 0.196, 'pos': 0.804, 'compound': 0.6249}
```

Creating a function

[127]:
```
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
```

[137]:
```
res = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
```

```
    try:
        text = row['content']
        myid = row['userName']
        vader_result = Analyzer.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%|          | 0/12000 [00:00<?, ?it/s]
```

```
Broke for id User_3073
Broke for id User_5048
Broke for id User_6463
Broke for id User_10731
```

[141]:
```
result_df =pd.DataFrame(res).T
```

[143]:
```
result_df.head()
```

[143]:

	unique_id	vader_neg	vader_neu	vader_pos	vader_compound	roberta_neg	roberta_neu	roberta_pos
0	User_0	0.0	0.0	1.0	0.4404	0.060793	0.329428	0.609778
1	User_1	0.0	0.0	1.0	0.4215	0.066038	0.354453	0.579509
2	User_2	0.0	1.0	0.0	0.0000	0.005416	0.075310	0.919274
3	User_3	0.0	0.0	1.0	0.4404	0.060793	0.329428	0.609778
4	User_4	0.0	1.0	0.0	0.0000	0.204142	0.604452	0.191406

[148]:
```
# Create results_df and set userName from index
results_df = pd.DataFrame(res).T.reset_index().rename(columns={'index':␣
 ↪'userName'})

# Select only sentiment columns plus userName from results_df
sentiment_columns = ['userName', 'vader_neg', 'vader_neu', 'vader_pos',␣
 ↪'vader_compound',
                     'roberta_neg', 'roberta_neu', 'roberta_pos']
results_df = results_df[sentiment_columns]
```

```
# Merge with df on userName
merged_df = df.merge(results_df, how='left', on='userName')

# Verify columns
print("Merged DataFrame columns:", merged_df.columns)
```

```
Merged DataFrame columns: Index(['unique_id', 'userName', 'userImage',
'content', 'score',
       'thumbsUpCount', 'reviewCreatedVersion', 'at', 'replyContent',
       'repliedAt', 'appVersion', 'vader_neg', 'vader_neu', 'vader_pos',
       'vader_compound', 'roberta_neg', 'roberta_neu', 'roberta_pos'],
      dtype='object')
```

[150]:
```
merged_df.head()
```

[150]:

	unique_id	userName	userImage	content	score	thumbsUpCount	\
0	0	User_0	NaN	Good	5	0	
1	1	User_1	NaN	Nice	5	0	
2	2	User_2	NaN	Very convenient	5	0	
3	3	User_3	NaN	Good	4	0	
4	4	User_4	NaN	exllence	5	0	

	reviewCreatedVersion	at	replyContent	repliedAt	\
0	4.556.10005	2024-12-18 17:17:19	NaN	NaN	
1	4.556.10005	2024-12-18 17:17:17	NaN	NaN	
2	4.532.10001	2024-12-18 17:09:42	NaN	NaN	
3	4.556.10005	2024-12-18 17:08:27	NaN	NaN	
4	4.556.10005	2024-12-18 17:08:16	NaN	NaN	

	appVersion	vader_neg	vader_neu	vader_pos	vader_compound	roberta_neg	\
0	4.556.10005	0.0	0.0	1.0	0.4404	0.060793	
1	4.556.10005	0.0	0.0	1.0	0.4215	0.066038	
2	4.532.10001	0.0	1.0	0.0	0.0000	0.005416	
3	4.556.10005	0.0	0.0	1.0	0.4404	0.060793	
4	4.556.10005	0.0	1.0	0.0	0.0000	0.204142	

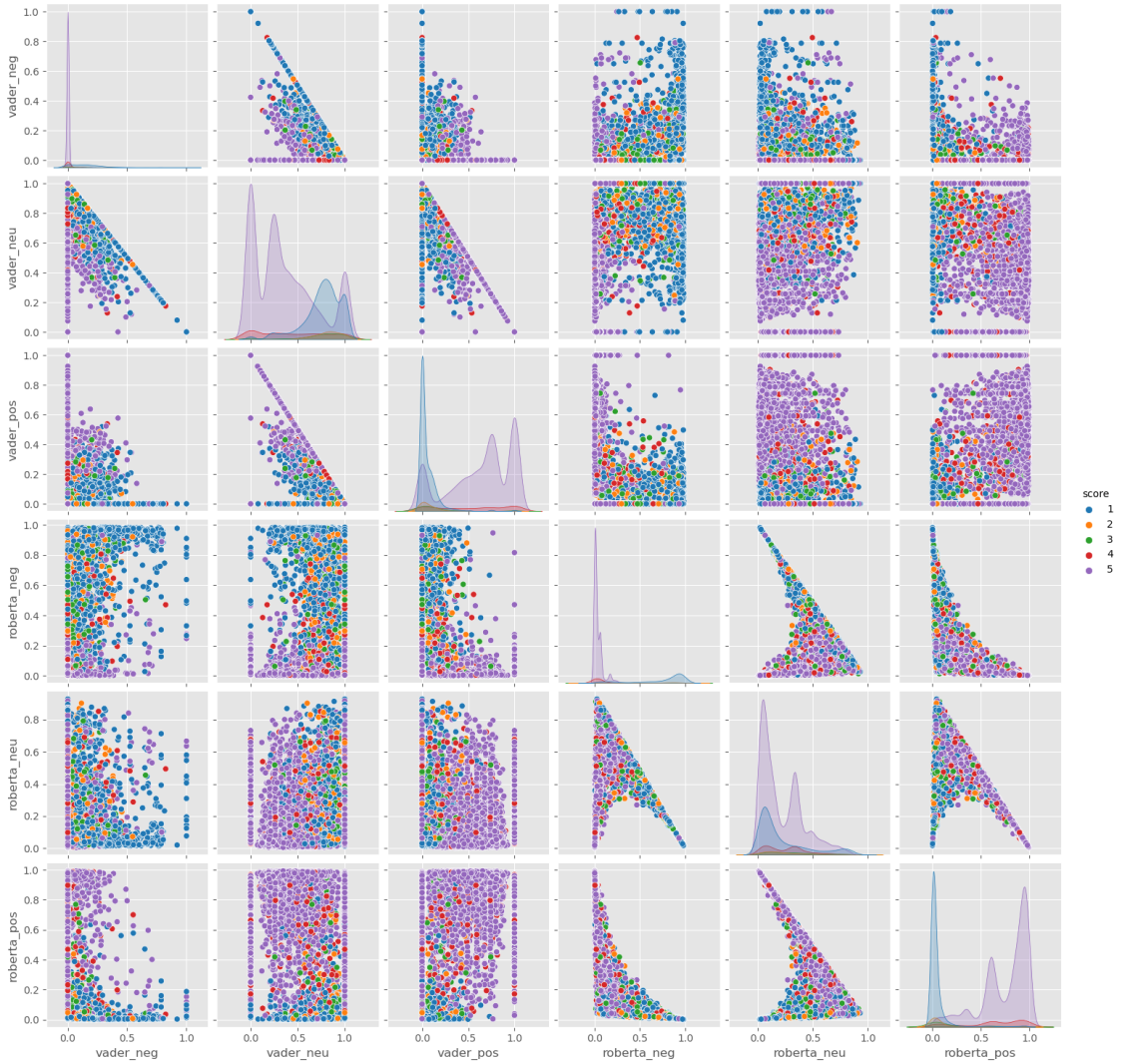
	roberta_neu	roberta_pos
0	0.329428	0.609778
1	0.354453	0.579509
2	0.075310	0.919274
3	0.329428	0.609778
4	0.604452	0.191406

comparing both models

[153]:
```
sns.pairplot(data=merged_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                   'roberta_neg', 'roberta_neu', 'roberta_pos'],
```

```
            hue='score',
            palette='tab10')
plt.show()
```

```
C:\Users\essie\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning:
The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```



Reviews

Text that have high positve score but the reviewer gave a very low rating, Lets look at some examples where the model scoring and review score differ the most.

Positive sentiment but a 1 star review

[156]:
```
merged_df.query('score == 1') \
    .sort_values('roberta_pos', ascending=False)['content'].values[0]
```

[156]:
```
'Rider is very good   '
```

[157]:
```
merged_df.query('score == 1') \
    .sort_values('vader_pos', ascending=False)['content'].values[0]
```

[157]:
```
'Good'
```

Lets do same for negative sentiment but a 5 star review

[160]:
```
merged_df.query('score == 5') \
    .sort_values('roberta_neg', ascending=False)['content'].values[0]
```

[160]:
```
'One of the worst app which is not useful late night I book a cab but it was
cancelled at least 10 times I was not give 1 star also'
```

[161]:
```
merged_df.query('score == 5') \
    .sort_values('vader_pos', ascending=False)['content'].values[0]
```

[161]:
```
'Good'
```

learn transformer pipeline

[ ]:
```

```