Sentiment Analysis in Python

June 8, 2024

1 Sentiment Analysis in Python

In this notebook we will be doing some sentiment analysis in python using two different techniques: 1.VADER (Valence Aware Dictionary and sentiment Reasoner) - Bag of words approach 2. Roberta Pretrained Model from 3. Hugging face Pipeline

2 Step 0. Read in Data and NLTK Basics

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

plt.style.use('ggplot')

import nltk
```

[2]:
```
# Read in data
df =
pd.read_csv('C:\Users\DhanuN\Documents\GitHub\Review_Data\Revie.csv')
print(df.shape)
df = df.head(500)
```

```
(568454, 10)
(500, 10)
```

[3]:
```
df.head()
```

[3]:

	Id	ProductId	UserId	ProfileName	\
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	\
0	1	1	5	1303862400	
1	0	0	1	1346976000	
2	1	1	4	1219017600	
3	3	3	2	1307923200	
4	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…

2.1 Quick EDA

[4]:
```
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()
```

2.2 Basic NLTK

[5]:
```
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.
```

[6]:
```
tokens = nltk.word_tokenize(example)
tokens[:10]
```

[6]:
```
['This', 'oatmeal', 'is', 'not', 'good', '.', 'Its', 'mushy', ',', 'soft']
```

[7]:
```
tagged = nltk.pos_tag(tokens)
tagged[:10]
```

[7]:
```
[('This', 'DT'),
 ('oatmeal', 'NN'),
 ('is', 'VBZ'),
 ('not', 'RB'),
 ('good', 'JJ'),
 ('.', '.'),
 ('Its', 'PRP$'),
 ('mushy', 'NN'),
 (',', ','),
 ('soft', 'JJ')]
```

[8]:
```
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
./.)
```

3 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:
  1. Stop words are removed
  2. each word is scored and combined to a total score.

[9]:
```
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm

sia = SentimentIntensityAnalyzer()
```

```
/opt/conda/lib/python3.7/site-packages/nltk/twitter/__init__.py:20: UserWarning:
The twython library has not been installed. Some functionality from the twitter
package will not be available.
  warnings.warn("The twython library has not been installed. "
```

[10]:
```
sia.polarity_scores('I am so happy!')
```

[10]:
```
{'neg': 0.0, 'neu': 0.318, 'pos': 0.682, 'compound': 0.6468}
```

[11]:
```
sia.polarity_scores('This is the worst thing ever.')
```

[11]:
```
{'neg': 0.451, 'neu': 0.549, 'pos': 0.0, 'compound': -0.6249}
```

[12]:
```
sia.polarity_scores(example)
```

[12]:
```
{'neg': 0.22, 'neu': 0.78, 'pos': 0.0, 'compound': -0.5448}
```

[13]:
```
# 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]
```

[14]:
```
vaders = pd.DataFrame(res).T
vaders = vaders.reset_index().rename(columns={'index': 'Id'})
vaders = vaders.merge(df, how='left')
```

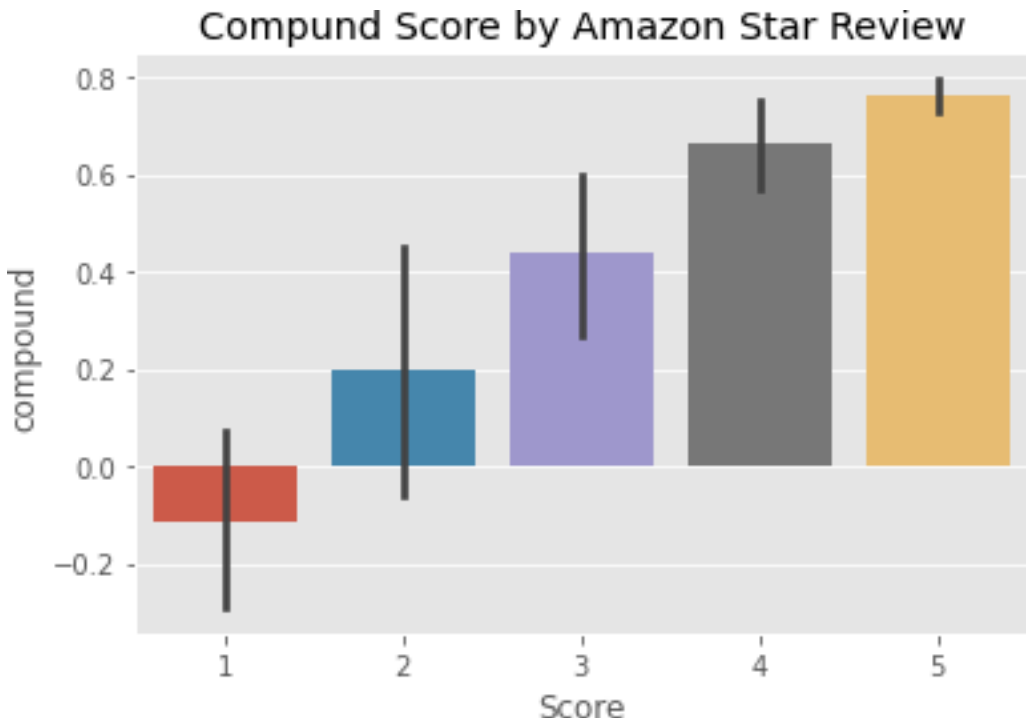
[15]:
```
# Now we have sentiment score and metadata
vaders.head()
```

[15]:

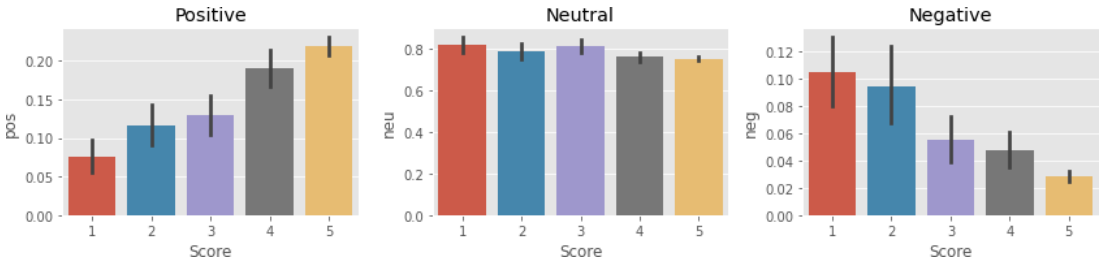
	Id	neg	neu	pos	compound	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	0.000	0.695	0.305	0.9441	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d…
1	2	0.079	0.853	0.068	-0.1027	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…

3.1 Plot VADER results

[16]:
```
ax = sns.barplot(data=vaders, x='Score', y='compound')
ax.set_title('Compund Score by Amazon Star Review')
plt.show()
```



[17]:
```
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()
```



4 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.

[18]:
```
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
```

[19]:
```
MODEL = f"cardiffnlp/twitter-roberta-base-sentiment"
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
```

```
Downloading:   0%|          | 0.00/747 [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/878k [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/446k [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/150 [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/476M [00:00<?, ?B/s]
```

[20]:
```
# VADER results on example
print(example)
sia.polarity_scores(example)
```

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

[20]:
```
{'neg': 0.22, 'neu': 0.78, 'pos': 0.0, 'compound': -0.5448}
```

[21]:
```
# 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': 0.9763551, 'roberta_neu': 0.020687457, 'roberta_pos':
0.0029573673}
```

[22]:
```
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
```

[23]:
```
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
```

[24]:
```
results_df = pd.DataFrame(res).T
results_df = results_df.reset_index().rename(columns={'index': 'Id'})
results_df = results_df.merge(df, how='left')
```

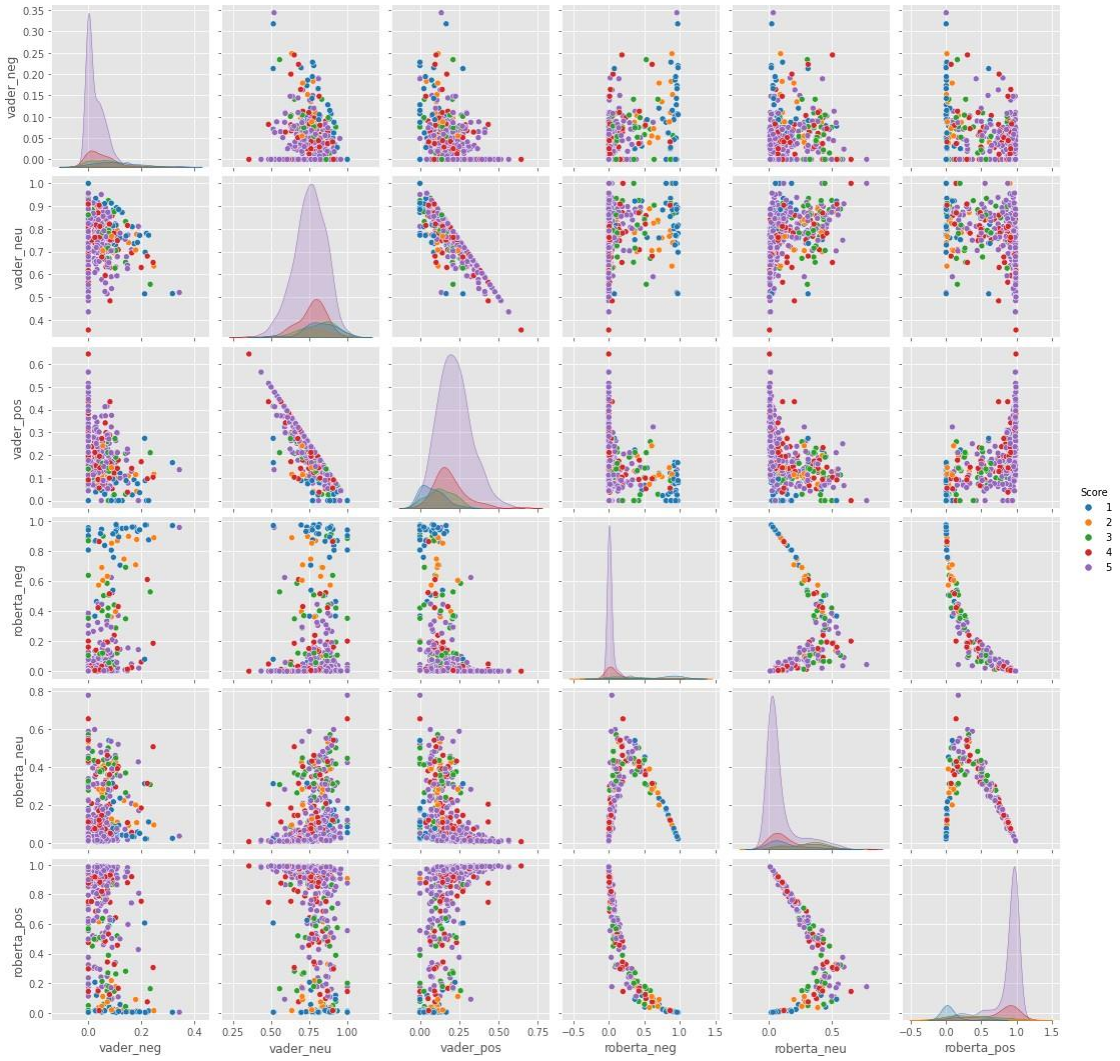
4.1 Compare Scores between models

[25]:
```
results_df.columns
```

[25]:
```
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')
```

5 Step 3. Combine and compare

[26]:
```
sns.pairplot(data=results_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                   'roberta_neg', 'roberta_neu', 'roberta_pos'],
             hue='Score',
             palette='tab10')
plt.show()
```



6 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.

[27]:
```
results_df.query('Score == 1') \
    .sort_values('roberta_pos', ascending=False)['Text'].values[0]
```

[27]:
```
'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.'
```

[28]:
```
results_df.query('Score == 1') \
    .sort_values('vader_pos', ascending=False)['Text'].values[0]
```

[28]:
```
'So we cancelled the order.  It was cancelled without any problem.  That is a positive note…'
```

[29]:
```
# nevative sentiment 5-Star view
```

[30]:
```
results_df.query('Score == 5') \
    .sort_values('roberta_neg', ascending=False)['Text'].values[0]
```

[30]:
```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my fault'
```

[31]:
```
results_df.query('Score == 5') \
    .sort_values('vader_neg', ascending=False)['Text'].values[0]
```

[31]:
```
'this was sooooo deliscious but too bad i ate em too fast and gained 2 pds! my fault'
```

7 Extra: The Transformers Pipeline

- Quick & easy way to run sentiment predictions

[32]:
```
from transformers import pipeline

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

```
No model was supplied, defaulted to distilbert-base-uncased-finetuned-sst-2-english (https://huggingface.co/distilbert-base-uncased-finetuned-sst-2-english)
Downloading:   0%|          | 0.00/629 [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/255M [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/48.0 [00:00<?, ?B/s]
Downloading:   0%|          | 0.00/226k [00:00<?, ?B/s]
```

[33]:
```
sent_pipeline('I love sentiment analysis!')
```

[33]: [{'label': 'POSITIVE', 'score': 0.9997853636741638}]

[34]:
```
sent_pipeline('Make sure to like and subscribe!')
```

[34]: [{'label': 'POSITIVE', 'score': 0.9991742968559265}]

[35]:
```
sent_pipeline('booo')
```

[35]: [{'label': 'NEGATIVE', 'score': 0.9936267137527466}]

8 The End