In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
from tqdm.notebook import tqdm
from transformers import pipeline
from transformers import AutoTokenizer
from transformers import AutoModelForSequenceClassification
from scipy.special import softmax
import warnings
```

In [2]:
```
warnings.filterwarnings("ignore")
plt.style.use('ggplot')
```

In [3]:
```
df = pd.read_csv('Reviews.csv')
df
```

Out[3]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Sun
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Qualit
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Adv
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"D say
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Me
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Grea
...	...	...	...	...	...	...	...	...	
568449	568450	B001EO7N10	A28KG5XORO54AY	Lettie D. Carter	0	0	5	1299628800	Will w
568450	568451	B003S1WTCU	A3I8AFVPEE8KI5	R. Sawyer	0	0	2	1331251200	disapp
568451	568452	B004I613EE	A121AA1GQV751Z	pksd "pk_007"	2	2	5	1329782400	Perf our ma
568452	568453	B004I613EE	A3IBEVCTXKNOH	Kathy A. Welch "katwel"	1	1	5	1331596800	Fa Trainir rewar
568453	568454	B001LR2CU2	A3LGQPJCZVL9UC	srfell17	0	0	5	1338422400	Great

568454 rows × 10 columns

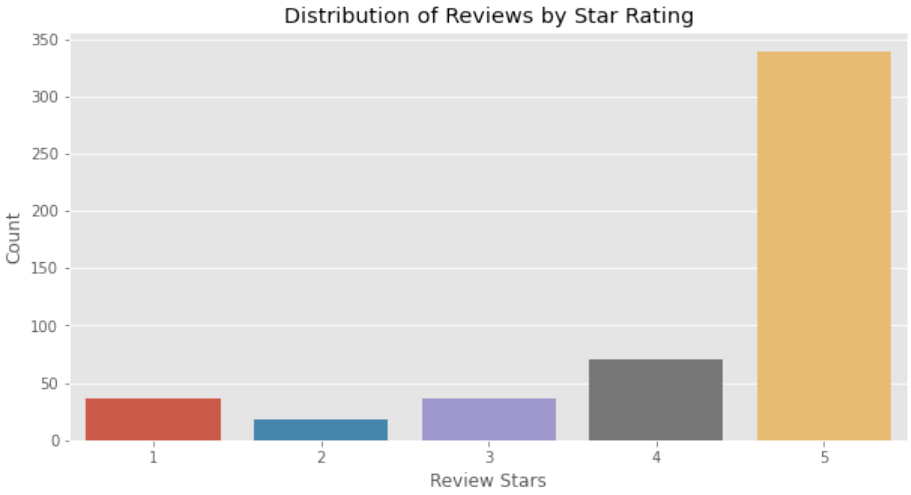
In [4]:
```
df = df.head(500)
df
```

Out[4]:

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy
...	...	...	...	...	...	...	...	...	...
495	496	B000G6RYNE	APGAA43E3WPN7	Darren	0	0	5	1201392000	amazing chips
496	497	B000G6RYNE	ABR7HU5H1KNE	Keith	0	0	5	1196726400	Best Chip Ever
497	498	B000G6RYNE	AJQD2WWJYOYFQ	bubbles	0	0	4	1186617600	Tangy, spicy, and sweet- oh my!
498	499	B000G6RYNE	A16YH487W9ZYO0	Bruce G. Lindsay	0	0	4	1184198400	An indulgence with a bite
499	500	B000G6RYNE	A83YQC1XOU4CS	J. Baker	0	0	5	1183420800	The best I've had

500 rows × 10 columns

In [5]:
```
plt.figure(figsize=(10, 5))
sns.countplot(data=df, x='Score')
plt.title('Distribution of Reviews by Star Rating')
plt.xlabel('Review Stars')
plt.ylabel('Count')
plt.show()
```



In [6]:
```
nltk.download('vader_lexicon')
```

```
[nltk_data] Downloading package vader_lexicon to
[nltk_data]     C:\Users\aryar\AppData\Roaming\nltk_data...
[nltk_data]   Package vader_lexicon is already up-to-date!
```

Out[6]:
```
True
```

In [7]:
```
sia = SentimentIntensityAnalyzer()
sia
```

Out[7]:
```
<nltk.sentiment.vader.SentimentIntensityAnalyzer at 0x29e463ea140>
```

In [8]:
```
vader_results = {}
for i, row in tqdm(df.iterrows(), total=len(df)):
    text = row['Text']
    myid = row['Id']
    vader_results[myid] = sia.polarity_scores(text)
```

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

In [9]:
```
vaders = pd.DataFrame(vader_results).T.reset_index().rename(columns={'index': 'Id'})
vaders = vaders.merge(df, how='left')
vaders
```

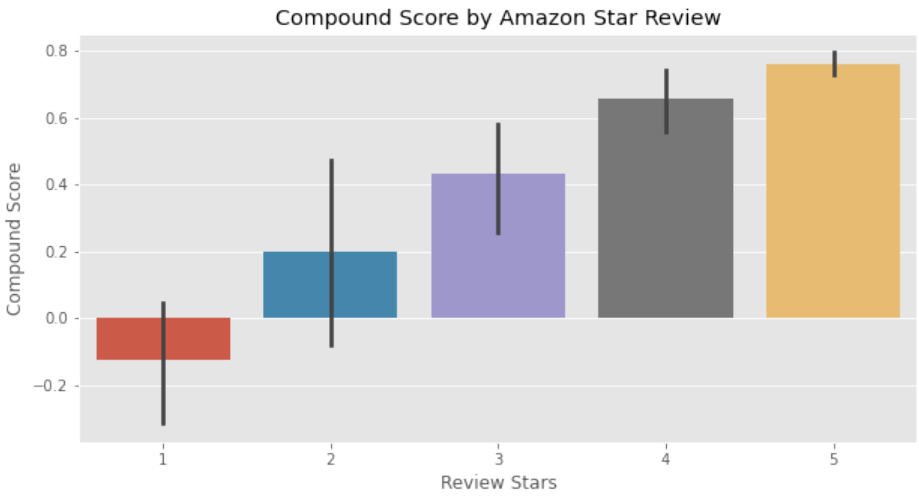
Out[9]:

	Id	neg	neu	pos	compound	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	S
0	1	0.000	0.695	0.305	0.9441	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	
1	2	0.138	0.862	0.000	-0.5664	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	
2	3	0.091	0.754	0.155	0.8265	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	
3	4	0.000	1.000	0.000	0.0000	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	
4	5	0.000	0.552	0.448	0.9468	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	
...	...	...	...	...	...	...	...	...	...	...	...
495	496	0.000	0.554	0.446	0.9725	B000G6RYNE	APGAA43E3WPN7	Darren	0	0	
496	497	0.059	0.799	0.142	0.7833	B000G6RYNE	ABR7HU5H1KNE	Keith	0	0	
497	498	0.025	0.762	0.212	0.9848	B000G6RYNE	AJQD2WWJYOYFQ	bubbles	0	0	
498	499	0.041	0.904	0.055	0.1280	B000G6RYNE	A16YH487W9ZYO0	Bruce G. Lindsay	0	0	
499	500	0.000	0.678	0.322	0.9811	B000G6RYNE	A83YQC1XOU4CS	J. Baker	0	0	

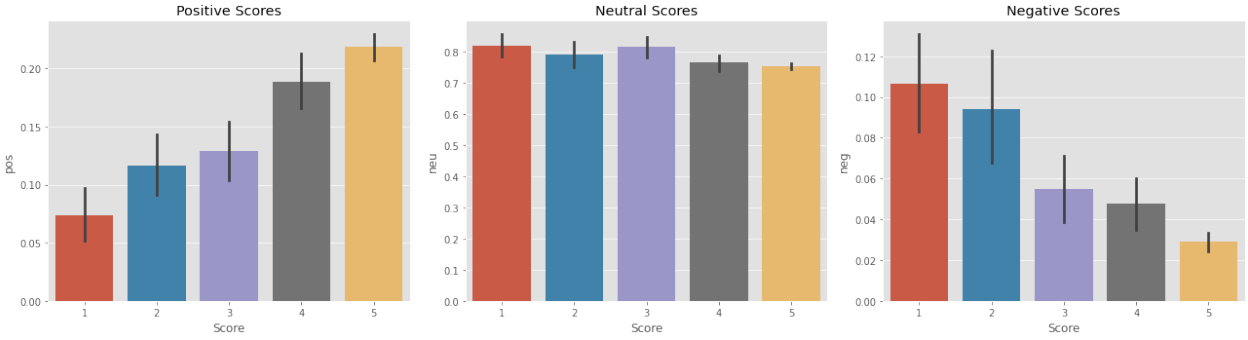
500 rows × 14 columns

In [10]:

```
plt.figure(figsize=(10, 5))
sns.barplot(data=vaders, x='Score', y='compound')
plt.title('Compound Score by Amazon Star Review')
plt.xlabel('Review Stars')
plt.ylabel('Compound Score')
plt.show()
```



In [11]:
```
fig, axes = plt.subplots(1, 3, figsize=(18, 5))
sns.barplot(data=vaders, x='Score', y='pos', ax=axes[0])
axes[0].set_title('Positive Scores')
sns.barplot(data=vaders, x='Score', y='neu', ax=axes[1])
axes[1].set_title('Neutral Scores')
sns.barplot(data=vaders, x='Score', y='neg', ax=axes[2])
axes[2].set_title('Negative Scores')
plt.tight_layout()
plt.show()
```



In [12]:
```
MODEL = f'cardiffnlp/twitter-roberta-base-sentiment'
tokenizer = AutoTokenizer.from_pretrained(MODEL)
model = AutoModelForSequenceClassification.from_pretrained(MODEL)
```

In [13]:
```
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
```

In [14]:
```
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
```

In [15]:
```
results_df = pd.DataFrame(res).T
results_df = results_df.reset_index().rename(columns={'index': 'Id'})
results_df = results_df.merge(df, how='left')
results_df
```

Out[15]:

	Id	vader_neg	vader_neu	vader_pos	vader_compound	roberta_neg	roberta_neu	roberta_pos	ProductId	UserId	P
0	1	0.000	0.695	0.305	0.9441	0.009624	0.049980	0.940395	B001E4KFG0	A3SGXH7AUHU8GW	
1	2	0.138	0.862	0.000	-0.5664	0.508986	0.452414	0.038600	B00813GRG4	A1D87F6ZCVE5NK	
2	3	0.091	0.754	0.155	0.8265	0.003229	0.098067	0.898704	B000LQOCH0	ABXLMWJIXXAIN	
3	4	0.000	1.000	0.000	0.0000	0.002295	0.090219	0.907486	B000UA0QIQ	A395BORC6FGVXV	
4	5	0.000	0.552	0.448	0.9468	0.001635	0.010302	0.988063	B006K2ZZ7K	A1UQRSCLF8GW1T	I
...	...	...	...	...	...	...	...	...	...	...	...
493	496	0.000	0.554	0.446	0.9725	0.001906	0.009862	0.988232	B000G6RYNE	APGAA43E3WPN7	
494	497	0.059	0.799	0.142	0.7833	0.004415	0.034215	0.961369	B000G6RYNE	ABR7HU5H1KNE	
495	498	0.025	0.762	0.212	0.9848	0.006427	0.074537	0.919036	B000G6RYNE	AJQD2WWJYOYFQ	
496	499	0.041	0.904	0.055	0.1280	0.865614	0.119366	0.015020	B000G6RYNE	A16YH487W9ZYO0	
497	500	0.000	0.678	0.322	0.9811	0.002440	0.011327	0.986233	B000G6RYNE	A83YQC1XOU4CS	

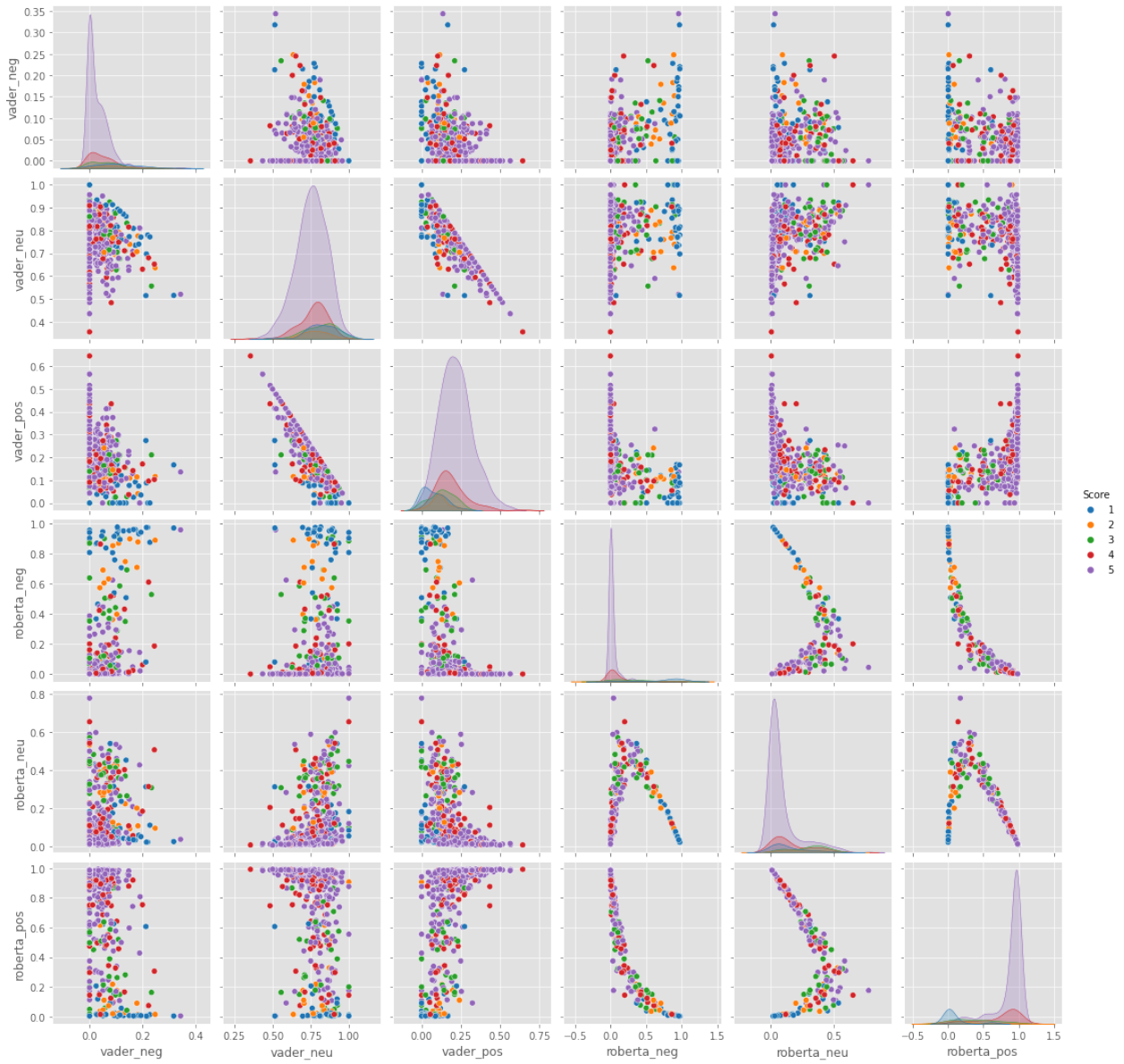
498 rows × 17 columns

In [16]:
```
results_df.columns
```

Out[16]:
```
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')
```

In [17]:
```
sns.pairplot(data=results_df,
             vars=['vader_neg', 'vader_neu', 'vader_pos',
                   'roberta_neg', 'roberta_neu', 'roberta_pos'],
```

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



In [18]:
```
sent_pipeline = pipeline("sentiment-analysis")
sent_pipeline
```

```
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.
WARNING:tensorflow:From C:\Python3\lib\site-packages\tf_keras\src\losses.py:2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
```

Out[18]:
```
<transformers.pipelines.text_classification.TextClassificationPipeline at 0x29e4e1c8d00>
```

In [19]:
```
hf_results = []
for i, row in tqdm(df.iterrows(), total=len(df)):
    try:
        text = row['Text']
        myid = row['Id']
        hf_result = sent_pipeline(text)[0]
        hf_result['Id'] = myid
        hf_results.append(hf_result)
    except RuntimeError:
        print(f'Broke for id {myid}')
```

```
  0%|          | 0/500 [00:00<?, ?it/s]
Token indices sequence length is longer than the specified maximum sequence length for this model (582 > 512). Running this sequence through the model will result in indexing errors
Broke for id 83
Broke for id 187
```

In [20]:
```
hf_df = pd.DataFrame(hf_results)
hf_df
```

Out[20]:

	label	score	Id
0	POSITIVE	0.998385	1
1	NEGATIVE	0.999525	2
2	POSITIVE	0.999765	3
3	POSITIVE	0.999153	4
4	POSITIVE	0.998708	5
...	...	...	...
493	POSITIVE	0.999860	496
494	POSITIVE	0.999754	497
495	POSITIVE	0.999425	498
496	POSITIVE	0.992952	499
497	POSITIVE	0.999796	500

498 rows × 3 columns

In [21]:

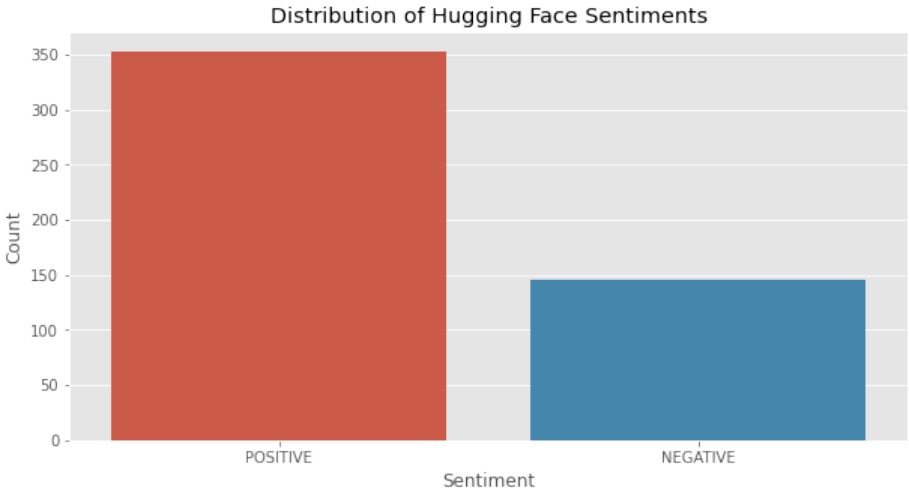
```
hf_df = hf_df.merge(df, on='Id')
hf_df
```

Out[21]:

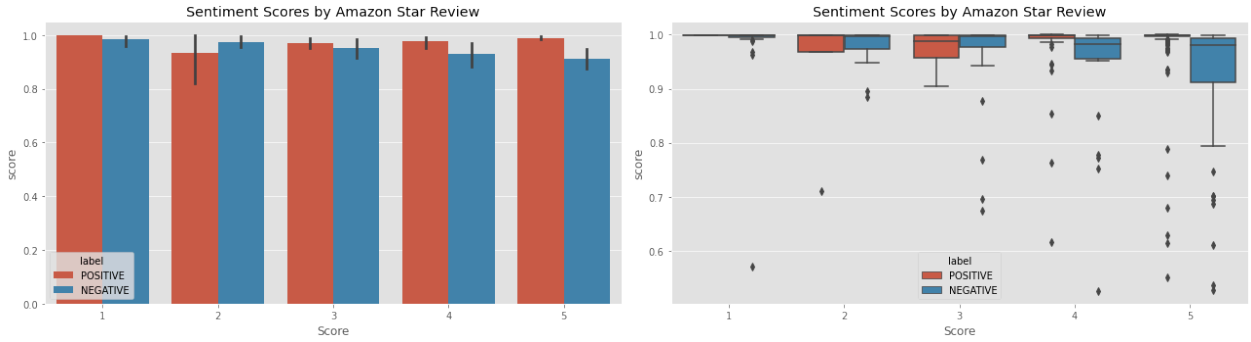
	label	score	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	
0	POSITIVE	0.998385	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303
1	NEGATIVE	0.999525	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346
2	POSITIVE	0.999765	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219
3	POSITIVE	0.999153	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307
4	POSITIVE	0.998708	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350
...	...	...	...	...	...	...	...	...	...	
493	POSITIVE	0.999860	496	B000G6RYNE	APGAA43E3WPN7	Darren	0	0	5	1201
494	POSITIVE	0.999754	497	B000G6RYNE	ABR7HU5H1KNE	Keith	0	0	5	1196
495	POSITIVE	0.999425	498	B000G6RYNE	AJQD2WWJYOYFQ	bubbles	0	0	4	1186
496	POSITIVE	0.992952	499	B000G6RYNE	A16YH487W9ZYO0	Bruce G. Lindsay	0	0	4	1184
497	POSITIVE	0.999796	500	B000G6RYNE	A83YQC1XOU4CS	J. Baker	0	0	5	1183

498 rows × 12 columns

In [22]:
```
plt.figure(figsize=(10, 5))
sns.countplot(data=hf_df, x='label')
plt.title('Distribution of Hugging Face Sentiments')
plt.xlabel('Sentiment')
plt.ylabel('Count')
plt.show()
```

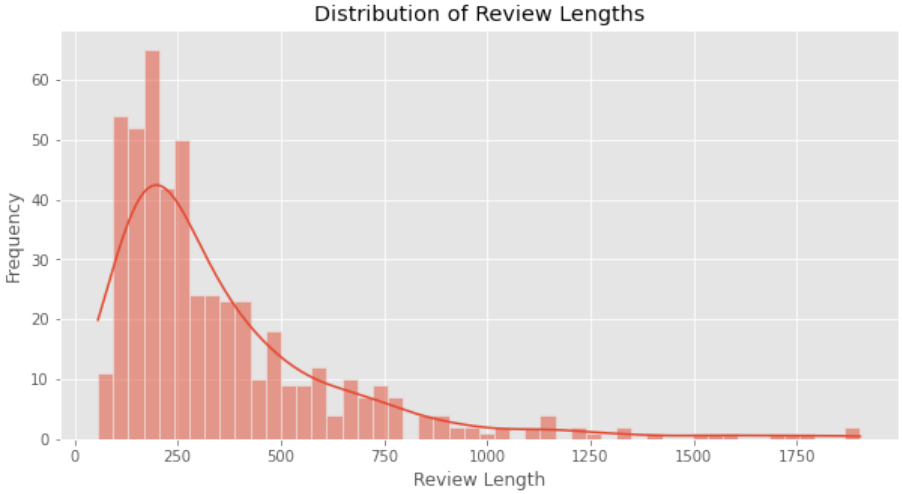


In [23]:
```
fig, axes = plt.subplots(1, 2, figsize=(18, 5))
sns.barplot(data=hf_df, x='Score', y='score', hue='label', ax=axes[0])
axes[0].set_title('Sentiment Scores by Amazon Star Review')
sns.boxplot(data=hf_df, x='Score', y='score', hue='label', ax=axes[1])
axes[1].set_title('Sentiment Scores by Amazon Star Review')
plt.tight_layout()
plt.show()
```



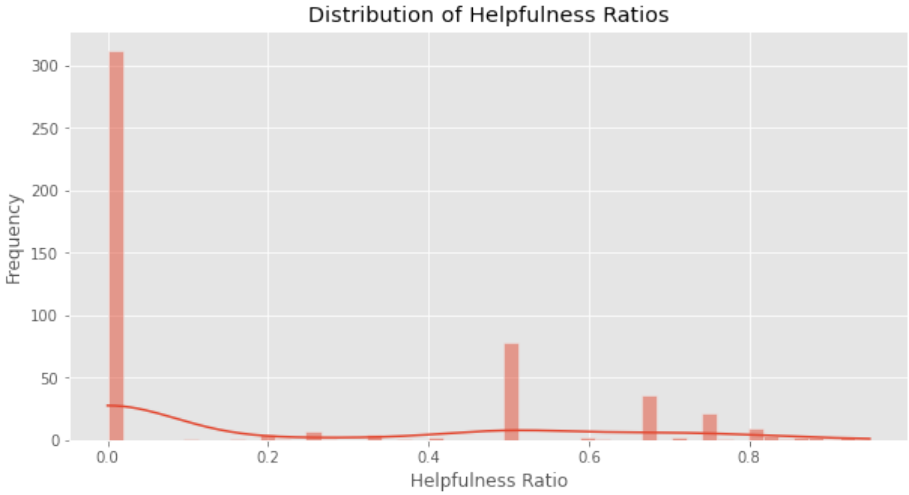
In [24]:
```
print("Review length distribution:")
hf_df['review_length'] = hf_df['Text'].apply(len)
plt.figure(figsize=(10, 5))
sns.histplot(hf_df['review_length'], bins=50, kde=True)
plt.title('Distribution of Review Lengths')
plt.xlabel('Review Length')
plt.ylabel('Frequency')
plt.show()
```

Review length distribution:



In [25]:
```
print("Helpfulness ratio distribution:")
hf_df['helpfulness_ratio'] = hf_df['HelpfulnessNumerator'] / (hf_df['HelpfulnessDenominator'] + 1)
plt.figure(figsize=(10, 5))
sns.histplot(hf_df['helpfulness_ratio'], bins=50, kde=True)
plt.title('Distribution of Helpfulness Ratios')
plt.xlabel('Helpfulness Ratio')
plt.ylabel('Frequency')
plt.show()
```

Helpfulness ratio distribution:



In [26]:
```
hf_df.head()
```

Out[26]:

	label	score	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Tim
0	POSITIVE	0.998385	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	13038624
1	NEGATIVE	0.999525	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	13469760
2	POSITIVE	0.999765	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	12190176
3	POSITIVE	0.999153	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	13079232
4	POSITIVE	0.998708	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	13507776

In [27]:
```
print("Most Positive Review with Hugging Face:", hf_df.sort_values(by='score', ascending=False).query('label == "POSITIVE"')['Text'].v
```

Most Positive Review with Hugging Face: good flavor! these came securely packed... they were fresh and delicious! i love these T wizzlers!

In [28]:
```
print("Most Negative Review with Hugging Face:", hf_df.sort_values(by='score', ascending=False).query('label == "NEGATIVE"')['Text
```

Most Negative Review with Hugging Face: Five minutes in, one tentacle was bitten off, ball inside cracked in half. Not durable e nough to be a dog toy. Disappointed :(. So is the dog :(.