

```
Out[10]: comment_id      0
score                0
self_text            0
subreddit            0
created_time         0
dtype: int64
```

```
In [11]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 189631 entries, 0 to 189630
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype
---  -
0   comment_id      189631 non-null  object
1   score           189631 non-null  int64
2   self_text       189631 non-null  object
3   subreddit       189631 non-null  object
4   created_time    189631 non-null  object
dtypes: int64(1), object(4)
memory usage: 7.2+ MB
```

```
In [12]: df.describe()
```

Out[12]:

	score
count	189631.000000
mean	28.583607
std	179.946085
min	-934.000000
25%	1.000000
50%	2.000000
75%	10.000000
max	16463.000000

```
In [13]: df.nunique()
```

Out[13]:

comment_id	189631
score	1761
self_text	186338
subreddit	14
created_time	13562

dtype: int64

```
In [14]: object_columns = df.select_dtypes(include=['object']).columns
print("Object type columns:")
print(object_columns)

numerical_columns = df.select_dtypes(include=['int', 'float']).columns
print("\nNumerical type columns:")
print(numerical_columns)

Object type columns:
Index(['comment_id', 'self_text', 'subreddit', 'created_time'], dtype='object')

Numerical type columns:
Index(['score'], dtype='object')
```

```
In [15]: df['subreddit'].unique()
```

Out[15]:

```
array(['worldnews', 'Palestine', 'IsraelPalestine', 'TerrifyingAsFuck',
       'worldnewsvideo', 'AskMiddleEast', 'CombatFootage',
       'PublicFreakout', 'NonCredibleDefense', 'IsrealPalestineWar_23',
       'CrazyFuckingVideos', 'AbruptChaos', 'NoahGetTheBoat',
       'ActualPublicFreakouts'], dtype=object)
```

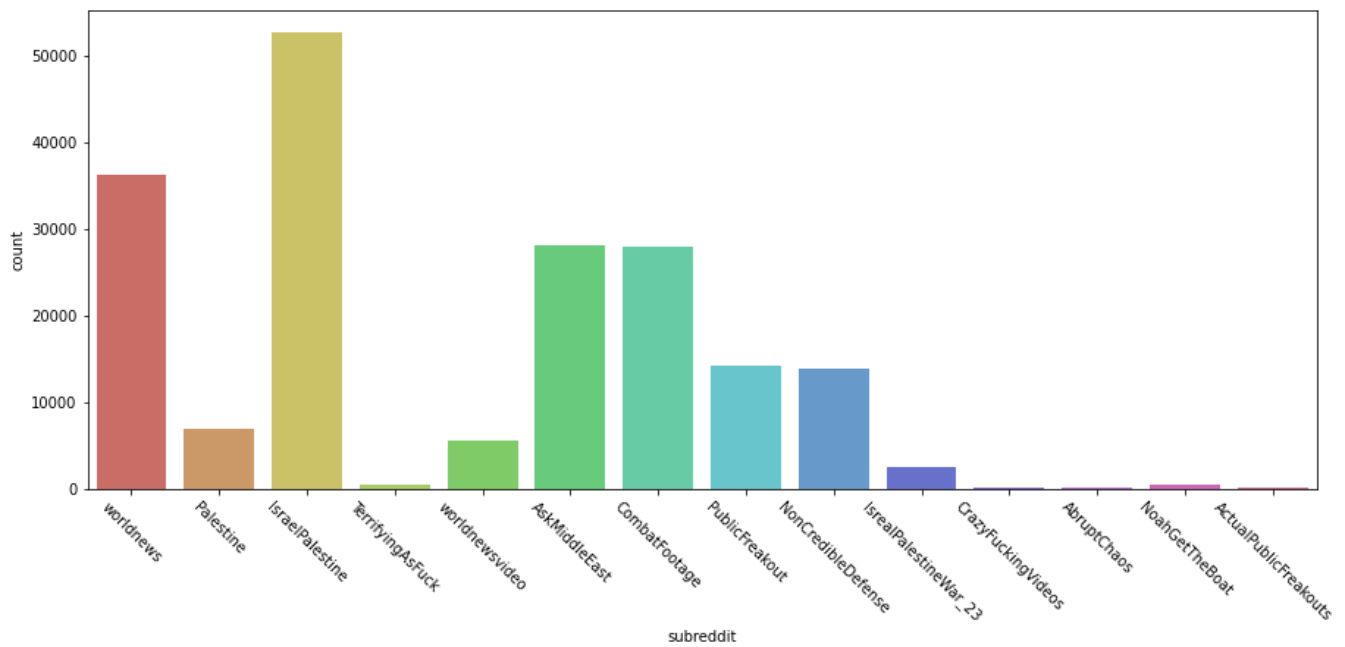
```
In [16]: df['subreddit'].value_counts()
```

Out[16]:

IsraelPalestine	52622
worldnews	36204
AskMiddleEast	28107
CombatFootage	27901
PublicFreakout	14255
NonCredibleDefense	13865
Palestine	6968
worldnewsvideo	5598
IsrealPalestineWar_23	2537
TerrifyingAsFuck	546
NoahGetTheBoat	498
AbruptChaos	200
CrazyFuckingVideos	197
ActualPublicFreakouts	133

Name: subreddit, dtype: int64

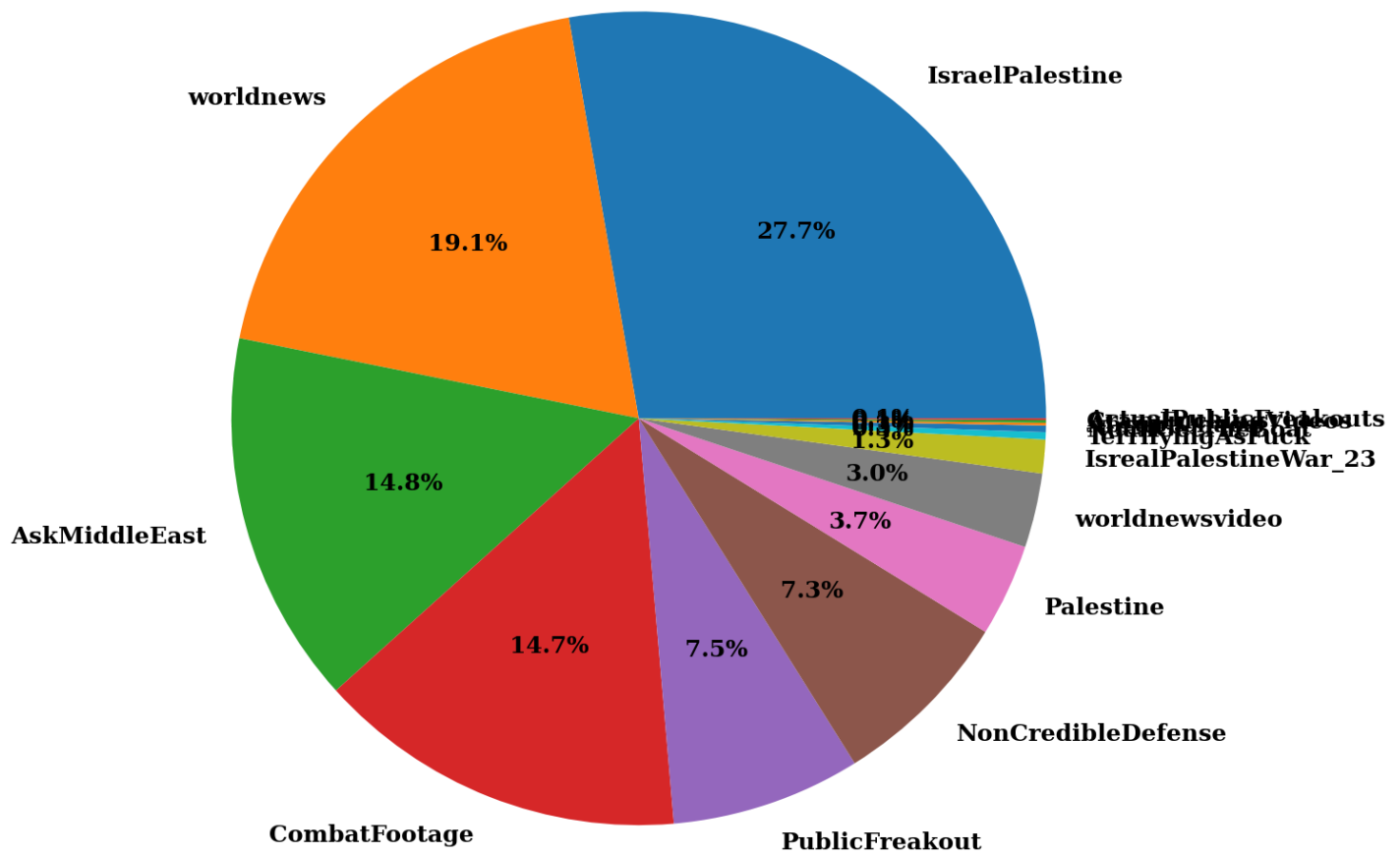
```
In [17]: plt.figure(figsize=(15,6))
sns.countplot(df['subreddit'], data = df, palette = 'hls')
plt.xticks(rotation = -45)
plt.show()
```



```
In [18]: plt.figure(figsize=(30,20))
plt.pie(df['subreddit'].value_counts(), labels=df['subreddit'].value_counts().index,
        autopct='%1.1f%%', textprops={ 'fontsize': 25,
                                         'color': 'black',
                                         'weight': 'bold',
                                         'family': 'serif' })

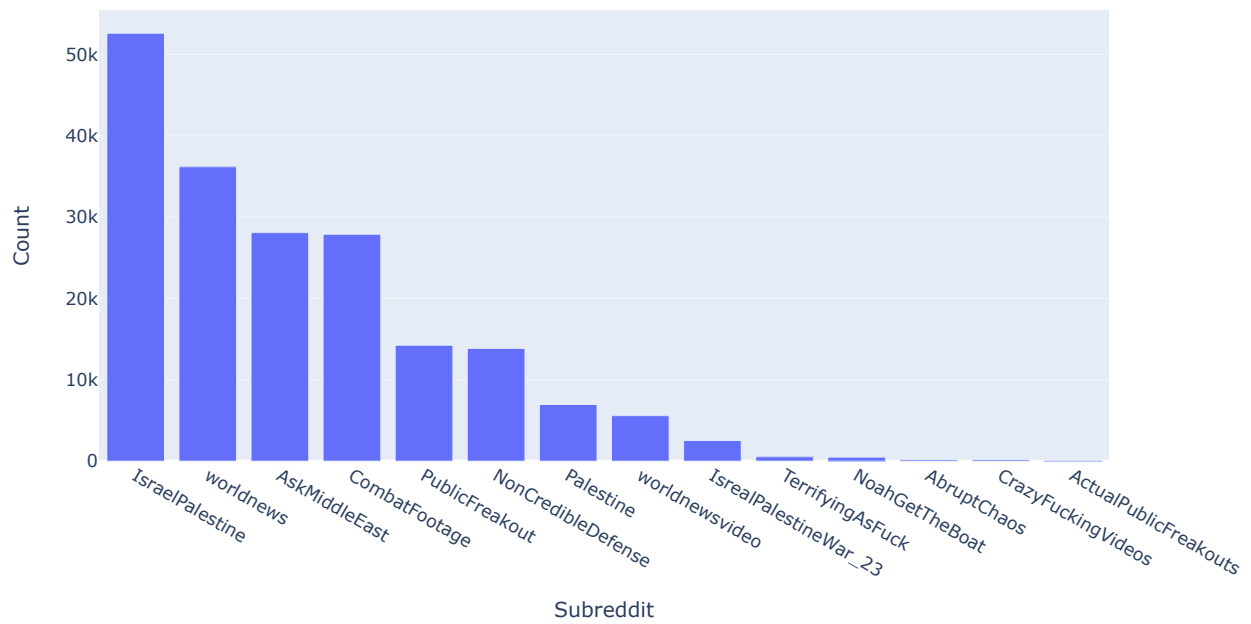
hfont = {'fontname': 'serif', 'weight': 'bold'}
plt.title('Subreddit', size=20, **hfont)
plt.show()
```

Subreddit



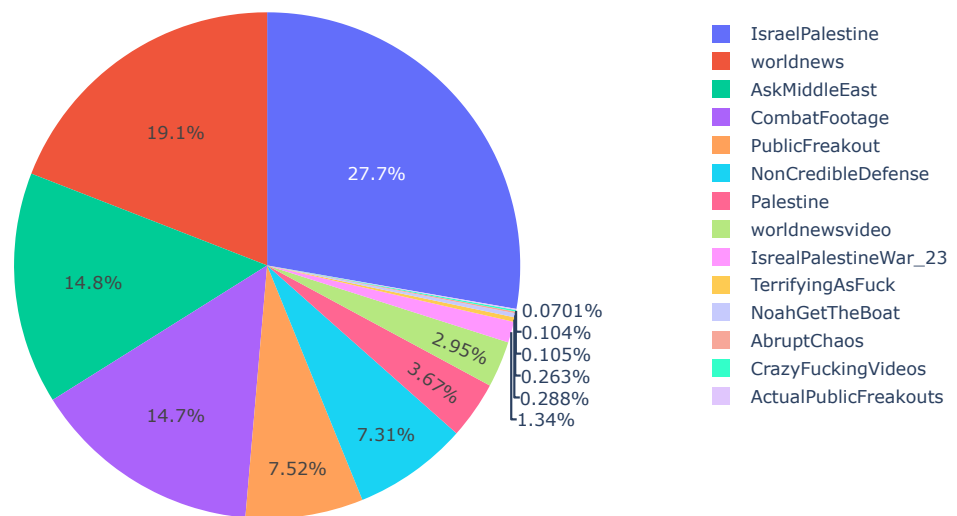
```
In [19]: fig = go.Figure(data=[go.Bar(x=df['subreddit'].value_counts().index,
                                       y=df['subreddit'].value_counts())])
fig.update_layout(title='Subreddit', xaxis_title='Subreddit', yaxis_title="Count")
fig.show()
```

Subreddit

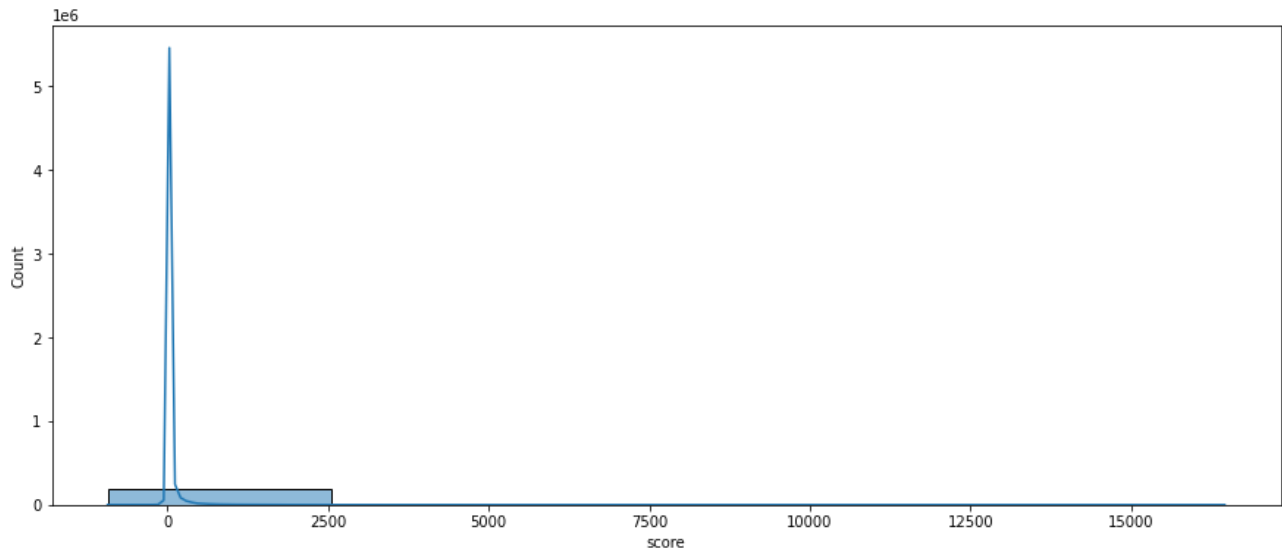


```
In [20]: fig = px.pie(df, names='subreddit', title = 'Subreddit')
fig.show()
```

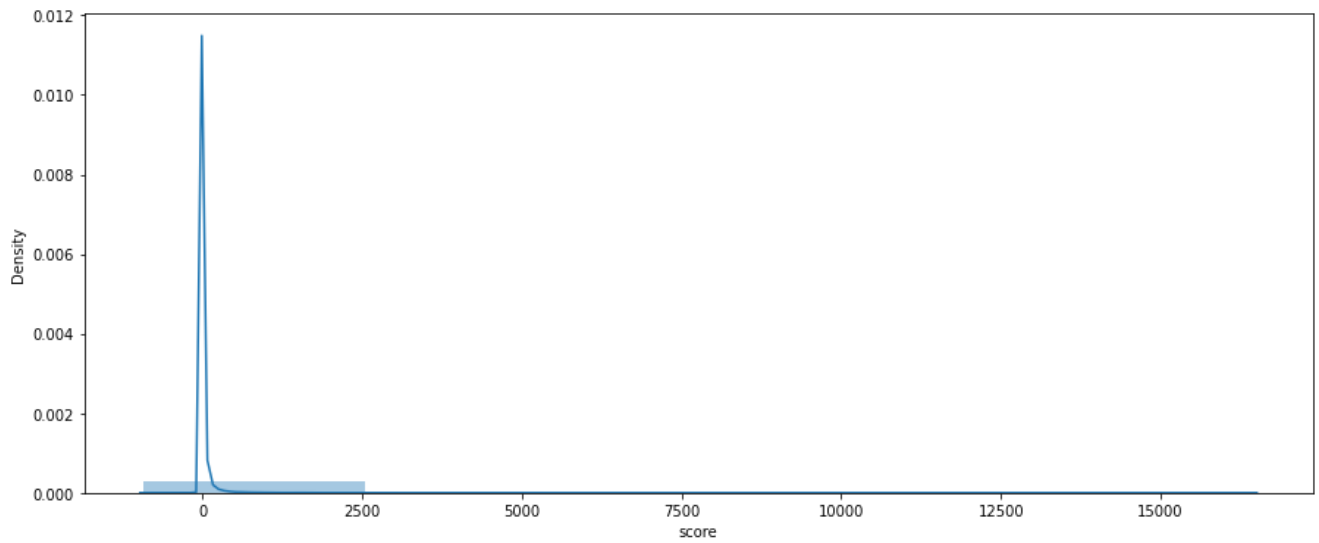
Subreddit



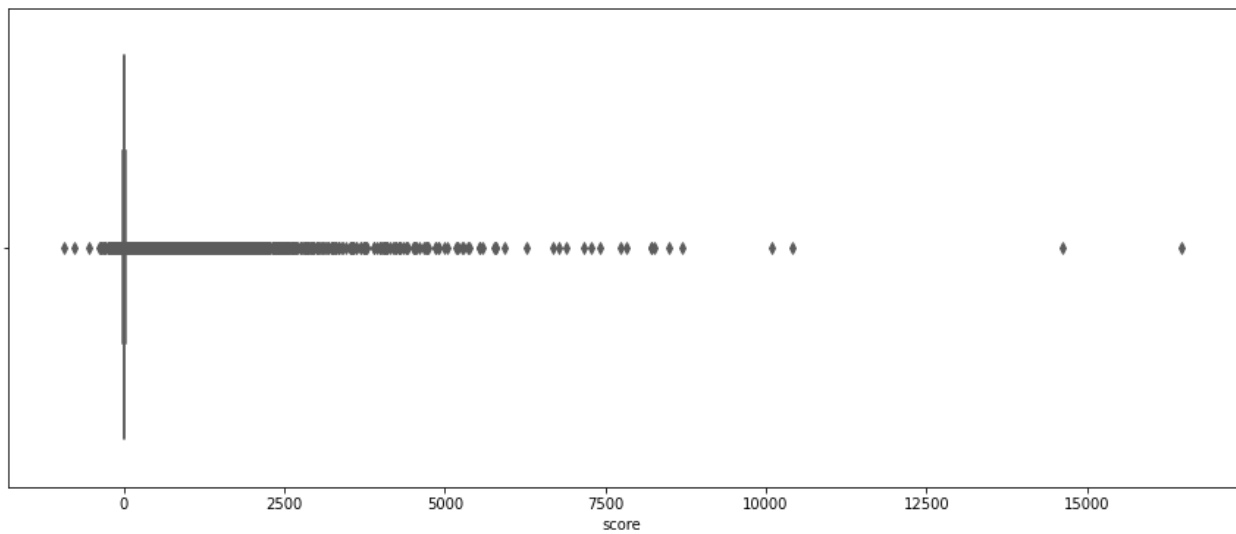
```
In [21]: plt.figure(figsize=(15,6))
sns.histplot(df['score'], kde = True, bins = 5, palette = 'hls')
plt.show()
```



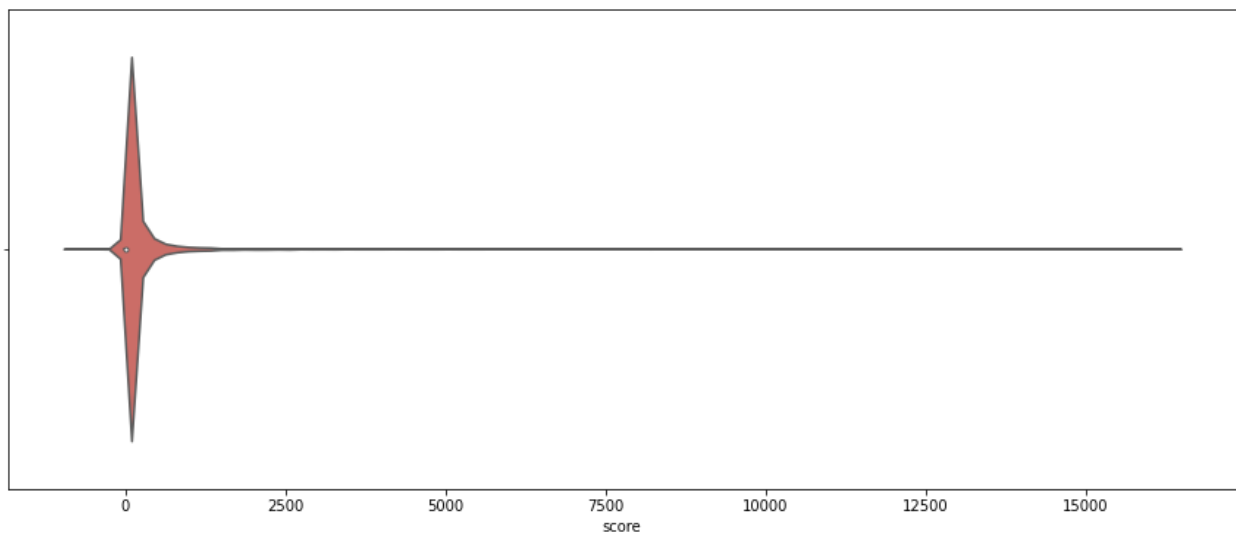
```
In [22]: plt.figure(figsize=(15,6))
sns.distplot(df['score'], kde = True, bins = 5)
plt.show()
```



```
In [23]: plt.figure(figsize=(15,6))
sns.boxplot(df['score'], data = df, palette = 'hls')
plt.show()
```

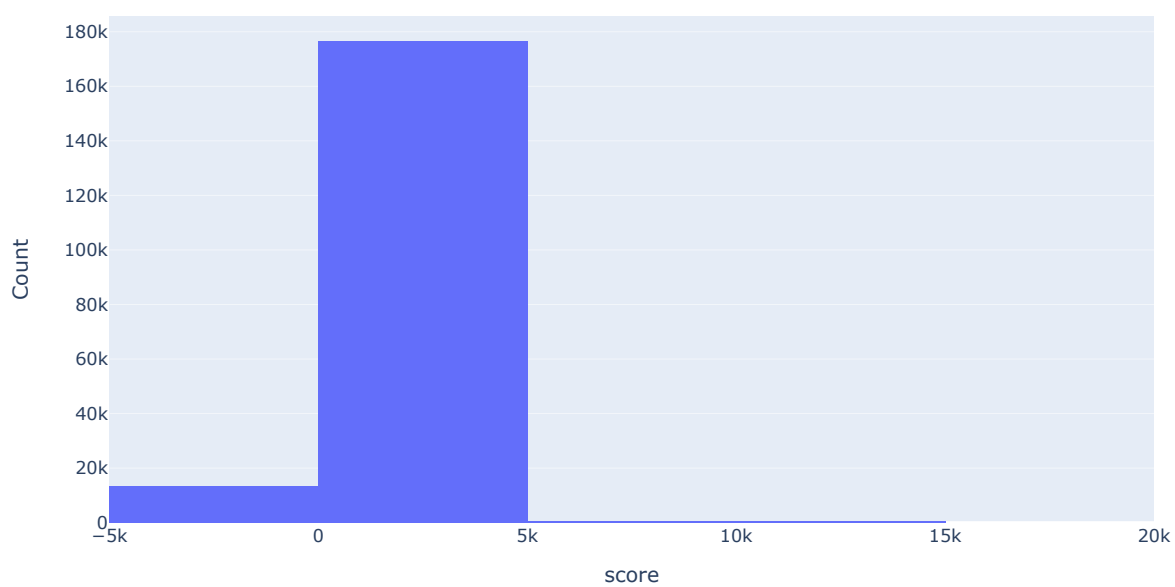


```
In [24]: plt.figure(figsize=(15,6))
sns.violinplot(df['score'], data = df, palette = 'hls')
plt.show()
```



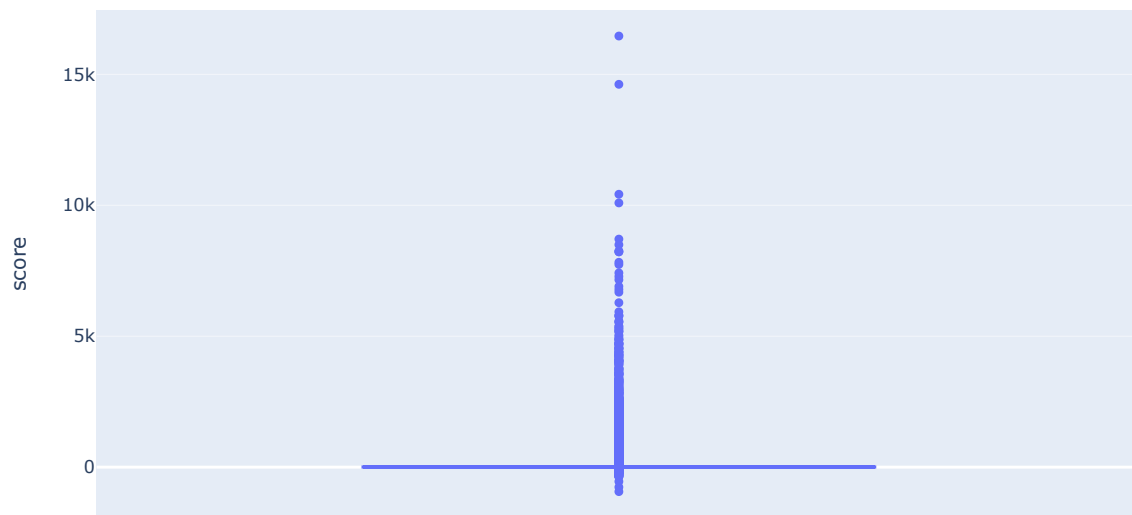
```
In [25]: fig = go.Figure(data=[go.Histogram(x=df['score'], nbinsx=5)])  
fig.update_layout(title='Histogram of score', xaxis_title='score', yaxis_title='Count')  
fig.show()
```

Histogram of score



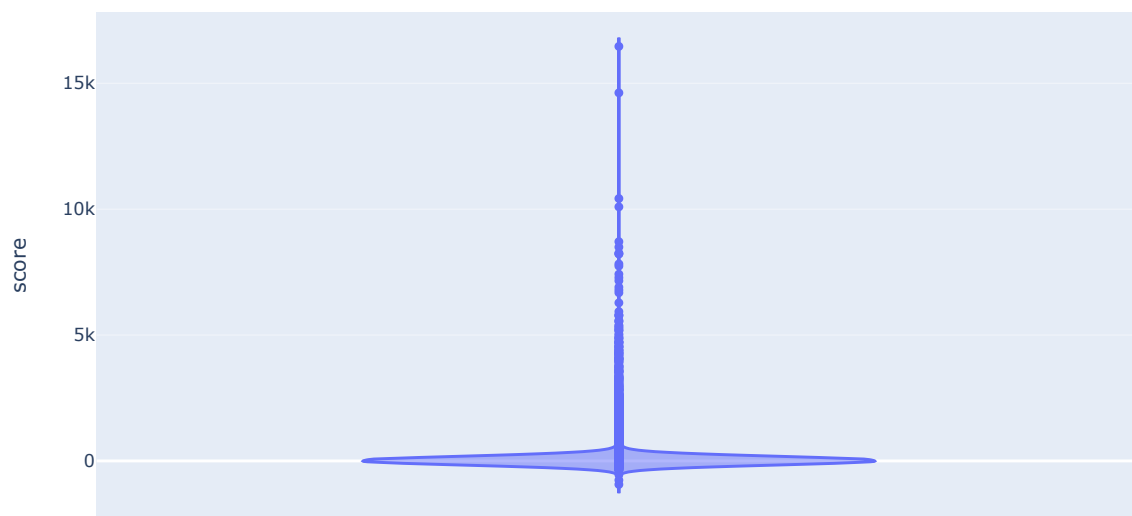
```
In [26]: fig = px.box(df, y='score', title=f'Box Plot Score')  
fig.show()
```

Box Plot Score



```
In [27]: fig = px.violin(df, y='score', title=f'Box Plot Score')
fig.show()
```

Box Plot Score



```
In [28]: df_new = df.copy()
```

```
In [29]: def clean_text(text):
text = text.lower()
return text.strip()
```

```
In [30]: df_new['self_text'] = df_new['self_text'].apply(lambda x: clean_text(x))
```

```
In [31]: import string
string.punctuation
```

```
Out[31]: '!"#$%&'()*+,-./:;<=>?@[\\]^_`{|}~'
```

```
In [32]: def remove_punctuation(text):
punctuationfree="".join([i for i in text if i not in string.punctuation])
return punctuationfree
```

```
In [33]: df_new['self_text'] = df_new['self_text'].apply(lambda x: remove_punctuation(x))
```

```
In [34]: import re
```

```

In [35]: def tokenization(text):
          tokens = re.split('W+',text)
          return tokens

In [36]: df_new['self_text'] = df_new['self_text'].apply(lambda x: tokenization(x))

In [37]: import nltk
          from wordcloud import WordCloud

In [38]: nltk.download('vader_lexicon')
          nltk.download('stopwords')

[nltk_data] Downloading package vader_lexicon to
[nltk_data] C:\Users\hp5cd\AppData\Roaming\nltk_data...
[nltk_data] Package vader_lexicon is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\hp5cd\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
Out[38]: True

In [39]: stopwords = nltk.corpus.stopwords.words('english')

In [40]: def remove_stopwords(text):
          output = " ".join(i for i in text if i not in stopwords)
          return output

In [41]: df_new['self_text'] = df_new['self_text'].apply(lambda x: remove_stopwords(x))

In [42]: from nltk.stem import WordNetLemmatizer

In [43]: wordnet_lemmatizer = WordNetLemmatizer()

In [44]: def lemmatizer(text):
          lemm_text = " ".join([wordnet_lemmatizer.lemmatize(word) for word in text])
          return lemm_text

In [45]: df_new['self_text'] = df_new['self_text'].apply(lambda x: lemmatizer(x))

In [46]: def clean_text(text):
          text = re.sub('[\.\*]', '', text).strip()
          text = re.sub('\S*\d\S*\s*', '', text).strip()
          return text.strip()

In [47]: df_new['self_text'] = df_new['self_text'].apply(lambda x: clean_text(x))

In [48]: def remove_urls(vTEXT):
          vTEXT = re.sub(r'(https|http)?://\/(\w|\.\|\/|\|?)\=|\&|\%)*\b', '', vTEXT, flags=re.MULTILINE)
          return(vTEXT)

In [49]: df_new['self_text'] = df_new['self_text'].apply(lambda x: remove_urls(x))

In [50]: def remove_digits(text):
          clean_text = re.sub(r"\b[0-9]+\b\s*", "", text)
          return(text)

In [51]: df_new['self_text'] = df_new['self_text'].apply(lambda x: remove_digits(x))

In [52]: def remove_emojis(data):
          emoji_pattern = re.compile("["
                                     u"\U0001F600-\U0001F64F"  # emoticons
                                     u"\U0001F300-\U0001F5FF"  # symbols & pictographs
                                     u"\U0001F680-\U0001F6FF"  # transport & map symbols
                                     u"\U0001F1E0-\U0001F1FF"  # flags (iOS)
                                     "]+", flags=re.UNICODE)
          return re.sub(emoji_pattern, '', data)

In [53]: df_new['self_text'] = df_new['self_text'].apply(lambda x: remove_emojis(x))

In [54]: df_new['self_text'] = df_new['self_text'].apply(lambda x: re.sub(r'\s+[a-zA-Z]\s+', '', x))

In [55]: df_new['self_text'] = df_new['self_text'].apply(lambda x: re.sub(r'\s+', ' ', x, flags=re.I))

In [56]: df_new

```

Out[56]:

	comment_id	score		self_text	subreddit	created_time
	0	k5480sx	1	exactlycan remember the humanitarian aid strea...	worldnews	16-10-2023 19:39
	1	k547q14	1	we are the only part of the world that has fre...	Palestine	16-10-2023 19:36
	2	k547elf	1	i don't make israeli strategy nor amisraeli or...	worldnews	16-10-2023 19:34
	3	k54742r	1	these people didnt vote hamas in or something ...	worldnews	16-10-2023 19:32
	4	k5473zi	1	we dont care what you do we just want to live ...	worldnews	16-10-2023 19:32

	189626	k3sdwfc	42	us this is bullshit	Palestine	07-10-2023 05:20
	189627	k3sdixt	1	i am in the united states and it has the dotte...	Palestine	07-10-2023 05:17
	189628	k3sccp2	54	in which country are you sometimes maps adapt ...	Palestine	07-10-2023 05:08
	189629	k3ritvj	116	you cant give up on something you only pretend...	worldnews	07-10-2023 01:46
	189630	k3riboh	30	gt the head of islamic jihad denounced arab at...	worldnews	07-10-2023 01:42

189631 rows × 5 columns

In [57]:

In [58]:

In [59]:

Out[59]:

	comment_id	score		self_text	subreddit	created_time	sentiment
	0	k5480sx	1	exactlycan remember the humanitarian aid strea...	worldnews	16-10-2023 19:39	0.000000
	1	k547q14	1	we are the only part of the world that has fre...	Palestine	16-10-2023 19:36	0.000000
	2	k547elf	1	i don't make israeli strategy nor amisraeli or...	worldnews	16-10-2023 19:34	0.305159
	3	k54742r	1	these people didnt vote hamas in or something ...	worldnews	16-10-2023 19:32	0.045000
	4	k5473zi	1	we dont care what you do we just want to live ...	worldnews	16-10-2023 19:32	-0.347643

	189626	k3sdwfc	42	us this is bullshit	Palestine	07-10-2023 05:20	0.000000
	189627	k3sdixt	1	i am in the united states and it has the dotte...	Palestine	07-10-2023 05:17	0.000000
	189628	k3sccp2	54	in which country are you sometimes maps adapt ...	Palestine	07-10-2023 05:08	0.000000
	189629	k3ritvj	116	you cant give up on something you only pretend...	worldnews	07-10-2023 01:46	-0.300000
	189630	k3riboh	30	gt the head of islamic jihad denounced arab at...	worldnews	07-10-2023 01:42	-0.032143

189631 rows × 6 columns

In [60]:

Correlation between "score" and sentiment:

	score	sentiment
score	1.00000	-0.00933
sentiment	-0.00933	1.00000

In [61]:

Average score per subreddit:

subreddit	
AbruptChaos	8.715000
ActualPublicFreakouts	60.563910
AskMiddleEast	4.736400
CombatFootage	40.808430
CrazyFuckingVideos	11.223350
IsraelPalestine	1.436205
IsrealPalestineWar_23	1.746945
NoahGetTheBoat	12.160643
NonCredibleDefense	33.659719
Palestine	11.412600
PublicFreakout	47.413750
TerrifyingAsFuck	36.661172
worldnews	76.210336
worldnewsvideo	8.808324
Name: score, dtype: float64	

In [62]:

In [63]:


```

Score trends over time:
created_time
2023-07-10    120.231962
2023-07-11         NaN
2023-07-12         NaN
2023-07-13         NaN
2023-07-14         NaN
...
2023-12-06         NaN
2023-12-07         NaN
2023-12-08         NaN
2023-12-09         NaN
2023-12-10    28.593894
Freq: D, Name: score, Length: 154, dtype: float64

```

```
In [64]: df_new['text_length'] = df_new['self_text'].apply(len)
```

```
In [65]: length_correlation = df_new[['score', 'text_length']].corr()
print('Correlation between "score" and text length:')
print(length_correlation)
```

```

Correlation between "score" and text length:
           score  text_length
score      1.00000      -0.01091
text_length -0.01091      1.00000

```

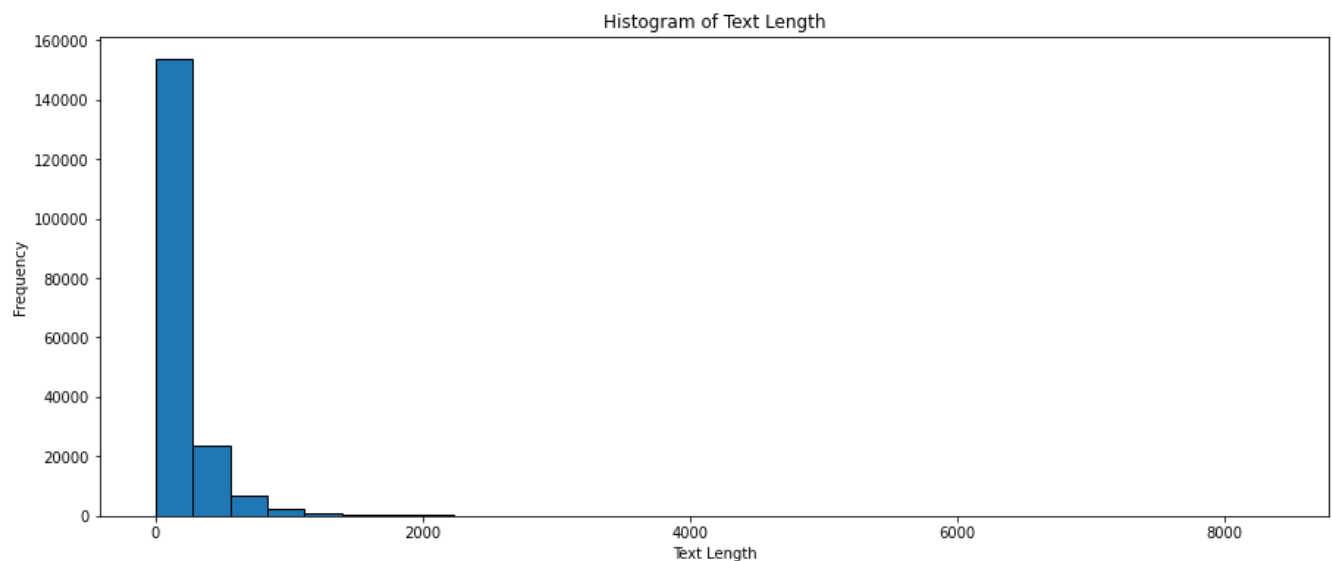
```
In [66]: from collections import Counter
```

```
In [67]: def word_frequency(word):
          return df_new[df_new['self_text'].str.contains(word, case=False, na=False)]['score'].mean()

word_example_frequency = word_frequency('example')
print('Average score for comments containing the word "example":', word_example_frequency)
```

```
Average score for comments containing the word "example": 14.987375415282392
```

```
In [68]: plt.figure(figsize=(15,6))
plt.hist(df_new['text_length'], bins=30, edgecolor='black')
plt.xlabel('Text Length')
plt.ylabel('Frequency')
plt.title('Histogram of Text Length')
plt.show()
```



```
In [69]: from sklearn.feature_extraction.text import CountVectorizer
          from sklearn.decomposition import LatentDirichletAllocation as LDA
```

```
In [70]: text_data = df_new['self_text'].astype(str)
```

```
In [71]: vectorizer = CountVectorizer(max_df=0.85, stop_words='english')
text_vectorized = vectorizer.fit_transform(text_data)
```

```
In [72]: num_topics = 5
lda = LDA(n_components=num_topics, random_state=42)
lda.fit(text_vectorized)
```

```
Out[72]: ▾ LatentDirichletAllocation
          LatentDirichletAllocation(n_components=5, random_state=42)
```

```
In [73]: for topic_idx, topic in enumerate(lda.components_):
          print(f"Topic {topic_idx + 1}:")
          print([vectorizer.get_feature_names()[i] for i in topic.argsort()[::-10 - 1:-1]])
          print()
```

Topic 1:
['iran', 'just', 'russia', 'israel', 'ukraine', 'like', 'dont', 'going', 'weapons', 'time']

Topic 2:
['hamas', 'israel', 'people', 'gaza', 'civilians', 'palestinians', 'just', 'israeli', 'like', 'war']

Topic 3:
['like', 'just', 'people', 'say', 'video', 'news', 'im', 'post', 'good', 'media']

Topic 4:
['comments', 'action', 'questions', 'comment', 'concerns', 'contact', 'based', 'automatically', 'performed', 'moderators']

Topic 5:
['israel', 'jews', 'land', 'palestine', 'people', 'palestinians', 'arab', 'like', 'jewish', 'state']

In [74]: df_new

Out[74]:

	comment_id	score	self_text	subreddit	created_time	sentiment	text_length
0	k5480sx	1	exactlycan remember the humanitarian aid strea...	worldnews	2023-10-16 19:39:00	0.000000	278
1	k547q14	1	we are the only part of the world that has fre...	Palestine	2023-10-16 19:36:00	0.000000	148
2	k547elf	1	i don't make israeli strategy nor amisraeli or...	worldnews	2023-10-16 19:34:00	0.305159	285
3	k54742r	1	these people didnt vote hamas in or something ...	worldnews	2023-10-16 19:32:00	0.045000	630
4	k5473zi	1	we dont care what you do we just want to live ...	worldnews	2023-10-16 19:32:00	-0.347643	293
...
189626	k3sdwfc	42	us this is bullshit	Palestine	2023-07-10 05:20:00	0.000000	19
189627	k3sdixt	1	i am in the united states and it has the dotte...	Palestine	2023-07-10 05:17:00	0.000000	120
189628	k3sccp2	54	in which country are you sometimes maps adapt ...	Palestine	2023-07-10 05:08:00	0.000000	123
189629	k3ritvj	116	you cant give up on something you only pretend...	worldnews	2023-07-10 01:46:00	-0.300000	79
189630	k3riboh	30	gt the head of islamic jihad denounced arab at...	worldnews	2023-07-10 01:42:00	-0.032143	1232

189631 rows × 7 columns

```
In [75]: highest_score_index = df_new['score'].idxmax()
lowest_score_index = df_new['score'].idxmin()

highest_score_text = df_new.loc[highest_score_index, 'self_text']
highest_score = df_new.loc[highest_score_index, 'score']

lowest_score_text = df_new.loc[lowest_score_index, 'self_text']
lowest_score = df_new.loc[lowest_score_index, 'score']

print(f"Comment with the highest score ({highest_score}):")
print(highest_score_text)
print("\n")

print(f"Comment with the lowest score ({lowest_score}):")
print(lowest_score_text)
```

Comment with the highest score (16463):
that's pretty damning for netanyahu and israeli intelligence no

Comment with the lowest score (-934):
too bad its pretty much pointless when they fire thousands of rockets

```
In [76]: def categorize_sentiment(polarity):
    if polarity > 0.05:
        return 'Positive'
    elif polarity < -0.05:
        return 'Negative'
    else:
        return 'Neutral'
```

```
In [77]: df_new['sentiment_category'] = df_new['sentiment'].apply(categorize_sentiment)
```

```
In [78]: df_new
```

Out[78]:

	comment_id	score	self_text	subreddit	created_time	sentiment	text_length	sentiment_category	
	0	k5480sx	1	exactlycan remember the humanitarian aid strea...	worldnews	2023-10-16 19:39:00	0.000000	278	Neutral
	1	k547q14	1	we are the only part of the world that has fre...	Palestine	2023-10-16 19:36:00	0.000000	148	Neutral
	2	k547elf	1	i don't make israeli strategy nor amisraeli or...	worldnews	2023-10-16 19:34:00	0.305159	285	Positive
	3	k54742r	1	these people didnt vote hamas in or something ...	worldnews	2023-10-16 19:32:00	0.045000	630	Neutral
	4	k5473zi	1	we dont care what you do we just want to live ...	worldnews	2023-10-16 19:32:00	-0.347643	293	Negative

	189626	k3sdwfc	42	us this is bullshit	Palestine	2023-07-10 05:20:00	0.000000	19	Neutral
	189627	k3sdixt	1	i am in the united states and it has the dotte...	Palestine	2023-07-10 05:17:00	0.000000	120	Neutral
	189628	k3sccp2	54	in which country are you sometimes maps adapt ...	Palestine	2023-07-10 05:08:00	0.000000	123	Neutral
	189629	k3ritvj	116	you cant give up on something you only pretend...	worldnews	2023-07-10 01:46:00	-0.300000	79	Negative
	189630	k3riboh	30	gt the head of islamic jihad denounced arab at...	worldnews	2023-07-10 01:42:00	-0.032143	1232	Neutral

189631 rows × 8 columns

In [79]:

df_new['sentiment_category'].unique()

Out[79]: array(['Neutral', 'Positive', 'Negative'], dtype=object)

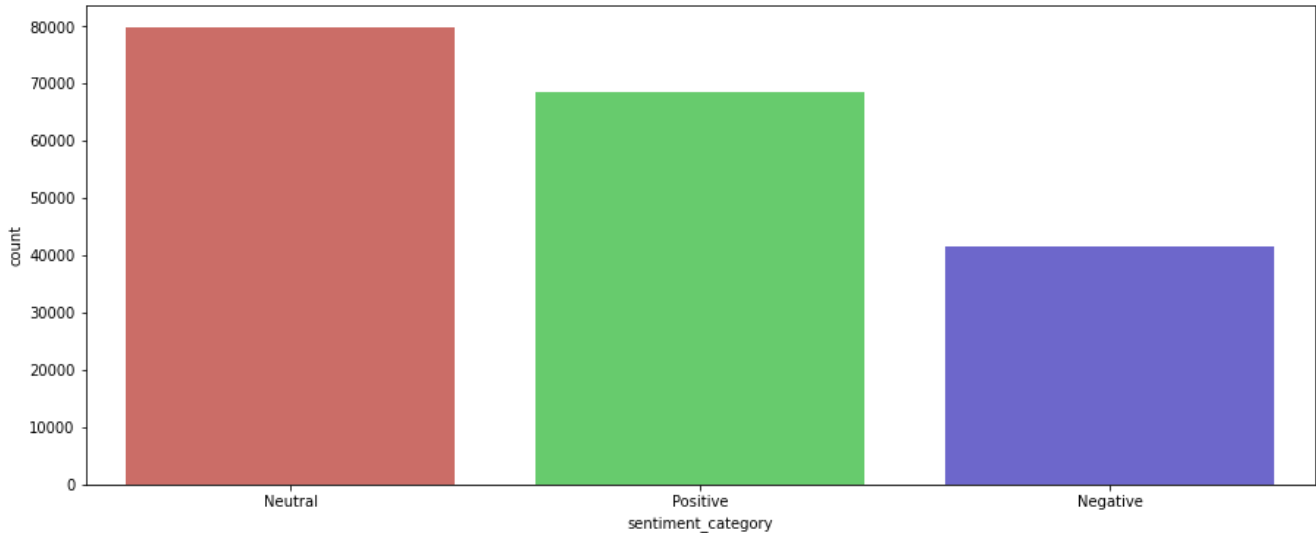
In [80]:

df_new['sentiment_category'].value_counts()

Out[80]: Neutral 79689
Positive 68529
Negative 41413
Name: sentiment_category, dtype: int64

In [81]:

plt.figure(figsize=(15,6))
sns.countplot(df_new['sentiment_category'], data = df_new, palette = 'hls')
plt.show()

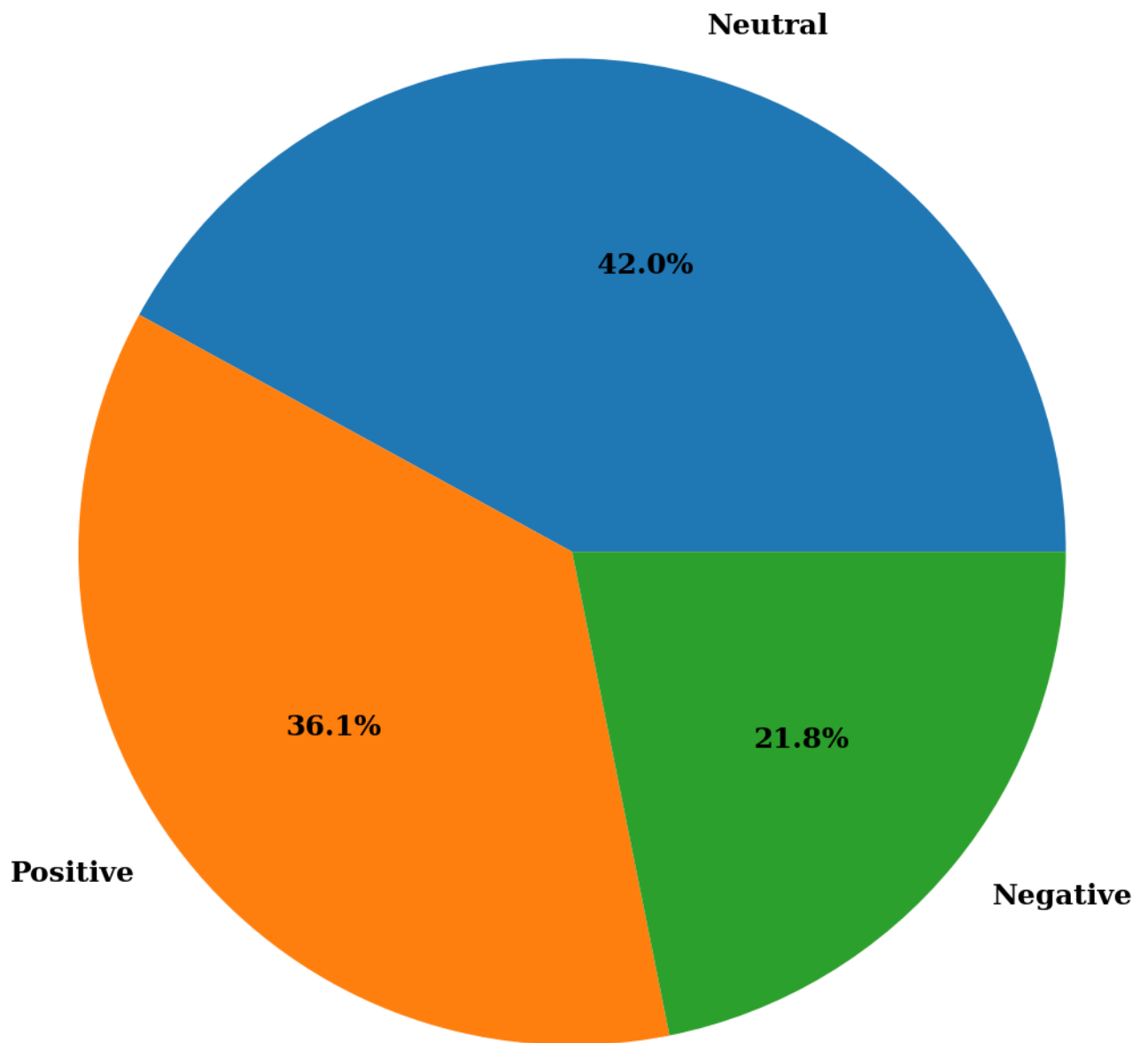


In [82]:

plt.figure(figsize=(30,20))
plt.pie(df_new['sentiment_category'].value_counts(), labels=df_new['sentiment_category'].value_counts().index,
autopct='%1.1f%', textprops={ 'fontsize': 25,
 'color': 'black',
 'weight': 'bold',
 'family': 'serif' })

hfont = {'fontname':'serif', 'weight': 'bold'}
plt.title('Sentiment Category', size=20, **hfont)
plt.show()

Sentiment Category



```
In [83]: def map_sentiment_to_numeric(sentiment_category):  
        if sentiment_category == 'Positive':  
            return 1  
        elif sentiment_category == 'Negative':  
            return -1  
        else:  
            return 0
```

```
In [84]: df_new['sentiment_numeric'] = df_new['sentiment_category'].apply(map_sentiment_to_numeric)
```

```
In [85]: df3 = df_new[['self_text', 'sentiment_numeric']]
```

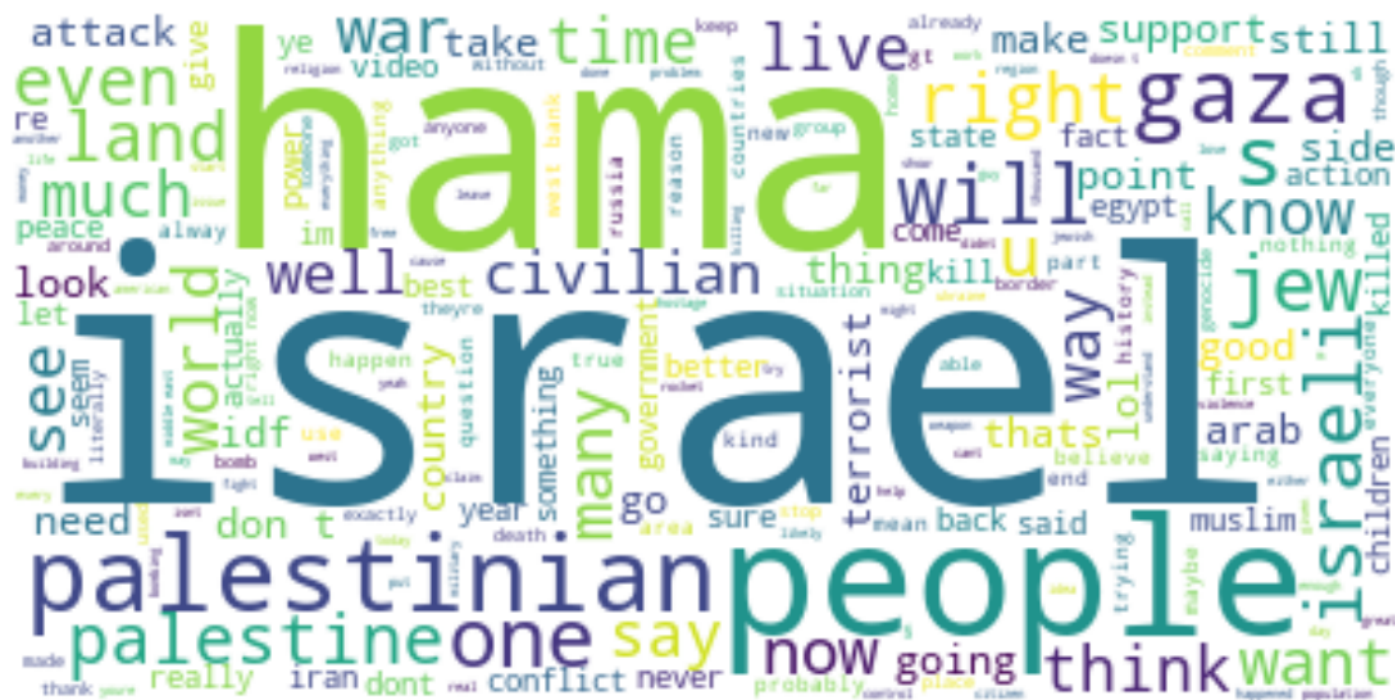
```
In [86]: df3
```

self text sentiment numeric

189631 rows x 2 columns


```
if positive_text_data:
    wordcloud = WordCloud(background_color='white').generate(positive_text_data)

    fig, ax = plt.subplots(figsize=(30, 10))
    ax.imshow(wordcloud, interpolation='bilinear')
    ax.axis('off')
    plt.show()
else:
    print('No positive text data to generate a word cloud.')
```



```
if negative_text_data:
    wordcloud = WordCloud(background_color='white').generate(negative_text_data)

    fig, ax = plt.subplots(figsize=(30, 10))
    ax.imshow(wordcloud, interpolation='bilinear')
    ax.axis('off')
    plt.show()
else:
    print('No negative text data to generate a word cloud.')
```


Out[95]:  LogisticRegression
LogisticRegression()

```
In [96]: y_pred = lr_classifier.predict(X_test)
lr_accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", lr_accuracy)
```


Accuracy: 0.8297255253513328

```
In [97]: precision, recall, f1, _ = precision_recall_fscore_support(y_test, y_pred, average='weighted')
print('Precision:', precision)
print('Recall:', recall)
print('F1 Score:', f1)
```

Precision: 0.8311469792237322
Recall: 0.8297255253513328
F1 Score: 0.8291814267939249

```
In [98]: from sklearn.tree import DecisionTreeClassifier
```

```
In [99]: dt_classifier = DecisionTreeClassifier()
dt_classifier.fit(X_train, y_train)
```

Out[99]:  DecisionTreeClassifier
DecisionTreeClassifier()

```
In [100]: y_pred = dt_classifier.predict(X_test)
dt_accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", dt_accuracy)
```

Accuracy: 0.7498615761858307

```
In [101]: precision, recall, f1, _ = precision_recall_fscore_support(y_test, y_pred, average='weighted')
print('Precision:', precision)
print('Recall:', recall)
print('F1 Score:', f1)
```

Precision: 0.7489914531442408
Recall: 0.7498615761858307
F1 Score: 0.7493180803889777

```
In [102]: accuracies = {
    'Logistic Regression': lr_accuracy,
    'Decision Tree': dt_accuracy,
}
```

```
In [103]: fig = go.Figure(
    data=[
        go.Bar(x=list(accuracies.keys()), y=list(accuracies.values()))
    ],
    layout={
        'title': 'Model Comparison: Accuracy',
        'xaxis': {'title': 'Models'},
        'yaxis': {'title': 'Accuracy'}
    }
)
fig.show()
```

Model Comparison: Accuracy

