sentimental-analysis

June 28, 2024

[2]:
```
import numpy as np
import pandas as pd
import os
import tensorflow as tf
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.models import Sequential
```

[4]:
```
df = pd.read_csv("/content/drive/MyDrive/archive (20)/amazon_reviews.csv")
```

[5]:
```
df.head()
```

[5]:

	Unnamed: 0	reviewerName	overall	reviewText	reviewTime	day_diff	helpful_yes	helpful_no	total_vote	score_pos_neg_diff	score_average_rating	wilson_lower_bound
0	0	NaN	4.0	No issues.	2014-07-23	138	0	0	0	0	0.0	0.0
1	1	Omie	5.0	Purchased this for my device, it worked as adv…	2013-10-25	409	0	0	0	0	0.0	0.0
2	2	1K3	4.0	it works as expected. I should have sprung for…	2012-12-23	715	0	0	0	0	0.0	0.0
3	3	1m2	5.0	This think has worked out great.Had a diff. br…	2013-11-21	382	0	0	0	0	0.0	0.0
4	4	2&1/2Men	5.0	Bought it with Retail Packaging, arrived legit…	2013-07-13	513	0	0	0	0	0.0	0.0

```
3                   0.0                 0.0
4                   0.0                 0.0
```

[6]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4915 entries, 0 to 4914
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Unnamed: 0            4915 non-null   int64
 1   reviewerName          4914 non-null   object
 2   overall               4915 non-null   float64
 3   reviewText            4914 non-null   object
 4   reviewTime            4915 non-null   object
 5   day_diff              4915 non-null   int64
 6   helpful_yes           4915 non-null   int64
 7   helpful_no            4915 non-null   int64
 8   total_vote            4915 non-null   int64
 9   score_pos_neg_diff    4915 non-null   int64
 10  score_average_rating  4915 non-null   float64
 11  wilson_lower_bound    4915 non-null   float64
dtypes: float64(3), int64(6), object(3)
memory usage: 460.9+ KB
```

[7]:
```
null_values = df.isnull().sum()
print("Null values in the entire Data:")
print(null_values)
```

```
Null values in the entire Data:
Unnamed: 0              0
reviewerName            1
overall                 0
reviewText              1
reviewTime              0
day_diff                0
helpful_yes             0
helpful_no              0
total_vote              0
score_pos_neg_diff      0
score_average_rating    0
wilson_lower_bound      0
dtype: int64
```

[8]:
```
df.dropna(inplace=True)
```

[9]:
```
null_values = df.isnull().sum()
null_values
```

[9]:
```
Unnamed: 0              0
reviewerName            0
overall                 0
reviewText              0
reviewTime              0
day_diff                0
helpful_yes             0
helpful_no              0
total_vote              0
score_pos_neg_diff      0
score_average_rating    0
wilson_lower_bound      0
dtype: int64
```

[10]:
```
df.drop_duplicates(inplace=True)
```

[11]:
```
import string
df['reviewText'] = df['reviewText'].apply(lambda x: x.lower())
df['reviewText'] = df['reviewText'].apply(lambda x: x.translate(str.
 ↪maketrans('', '', string.punctuation)))
```

[12]:
```
df['reviewText']
```

[12]:
```
1       purchased this for my device it worked as adve…
2       it works as expected i should have sprung for …
3       this think has worked out greathad a diff bran…
4       bought it with retail packaging arrived legit …
5       its mini storage  it doesnt do anything else a…
                              …
4910    i bought this sandisk 16gb class 10 to use wit…
4911    used this for extending the capabilities of my…
4912    great card that is very fast and reliable it c…
4913    good amount of space for the stuff i want to d…
4914    ive heard bad things about this 64gb micro sd …
Name: reviewText, Length: 4913, dtype: object
```

[13]:
```
from sklearn.feature_extraction.text import CountVectorizer
# Assuming 'df' is your Data containing text data
text_data = df['reviewText']
vectorizer = CountVectorizer()
feature_matrix = vectorizer.fit_transform(text_data)
feature_names = vectorizer.get_feature_names_out()
```

[14]:
```
feature_names
```

[14]:
```
array(['00', '001191', '00148', …, 'zumo', 'zune', 'zunehd'],
      dtype=object)
```

[15]:
```
import sklearn.feature_extraction.text as text
count_vectorizer = text.CountVectorizer()
```

[16]:
```
count_vectorizer.fit(df.reviewText)
```

[16]:
```
CountVectorizer()
```

[17]:
```
data_features = count_vectorizer.transform(df.reviewText)
```

[18]:
```
density = (data_features.getnnz() * 100) / (data_features.shape[0] *
                                            data_features.shape[1])
print("Density of the matrix: ", density)
```

```
Density of the matrix:  0.32925356540915807
```

[19]:
```
feature_counts = df['reviewText'].value_counts()
feature_counts
```

[19]:
```
reviewText
works fine my virgin mobile htc evo v the phone didnt recognize it at first but
after formatting its all good
2
yes
2
brand speaks for itselfrecommend to a size 64 gbprice is a bit highsandisk ultra
64 gb microsdxc class 10 uhs1 memory card
1
with the small 16 g onboard memory in this tablet the added 32 makes it
pleasurable for textbooks music and pictures
1
this is one of those products that either works or it doesnt which rarely
happens so what can i say  it works great  32 gb allows plenty of room for
storage in my tablet and with the high rate transfer speed apps run as quickly
as if they were using internal memory storage
1
                    ..
fast and safe very good quality and has an excellent performacea bit expensive
comparing to other models in the store
1
i bought this to put into a wifi enabled usb flash drive it works fine i can get
my entire cd collection on it which was my goal
1
i took advantage of an irresistible offer from amazon and this 32gb card
performed like charm in my rokuim very satisfied and suggest to anyone looking
```

```
for fast micro cards give a look to this one
1
dont even waste your time buying this memory card 64gb is written on it but its
actually even less than 3gb thats in iti have never experienced such deceptioni
had tried loading my pics and music and other data on it from day one but no
matter what i did i wont take more than 3gb of data i didnt realise till i read
other stories of similar memory cards that will state 64gb but would contain a
smaller memory capacity often times even less than 10gbi bought this card for
49what a big wastenow i am so scared of buying memory cards on amazon or online
elsewherei wouldnt dear it againit was in a sealed  sanddisk packet so i was
thinking it must be originalalasit was a big  wastei have lost confidence in
buying memory card on amazon    1
ive heard bad things about this 64gb micro sd card crapping out after a few
weeks but sk far so good transfer speeds are normal but i like to be space for
the price it was a stealwould recommend
1
Name: count, Length: 4911, dtype: int64
```

[20]:
```
features = vectorizer.get_feature_names_out()  # Replace with the variable that␣
↪holds feature names
features_counts = np.sum(data_features.toarray(), axis=0)
features_counts_df = pd.DataFrame({'features': features, 'counts':␣
↪features_counts})
```

[21]:
```
count_of_single_occurrences =␣
↪len(features_counts_df[features_counts_df['counts'] == 1])
count_of_single_occurrences
```

[21]:
```
6624
```

[22]:
```
count_vectorizer = CountVectorizer(max_features=10000)
feature_vector = count_vectorizer.fit_transform(df['reviewText'])
features = count_vectorizer.get_feature_names_out()
data_features = feature_vector.toarray()
features_counts = np.sum(data_features, axis=0)
feature_counts = pd.DataFrame({'features': features, 'counts': features_counts})
```

[24]:
```
top_features_counts = feature_counts.sort_values(by='counts', ascending=False).
↪head(15)
```

[25]:
```
top_features_counts
```

[25]:

	features	counts
8723	the	9077
1444	and	7585
5096	it	6836
8900	to	6135

```
5762       my    5048
2180     card    4591
8789     this    4490
3921      for    4033
4829       in    3372
5879       of    3369
5063       is    3111
9744     with    3056
5921       on    2220
4484     have    2153
8713     that    1932
```

[26]:
```
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')
english_stop_words = stopwords.words('english')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data…
[nltk_data]   Package stopwords is already up-to-date!
```

[27]:
```
df['reviewText'][0:10]
```

[27]:
```
1     purchased this for my device it worked as adve…
2     it works as expected i should have sprung for …
3     this think has worked out greathad a diff bran…
4     bought it with retail packaging arrived legit …
5     its mini storage  it doesnt do anything else a…
6     i have it in my phone and it never skips a bea…
7     its hard to believe how affordable digital has…
8     works in a htc rezound  was running short of s…
9     in my galaxy s4 super fast card and am totally…
10    i like this sd card because it can take music …
Name: reviewText, dtype: object
```

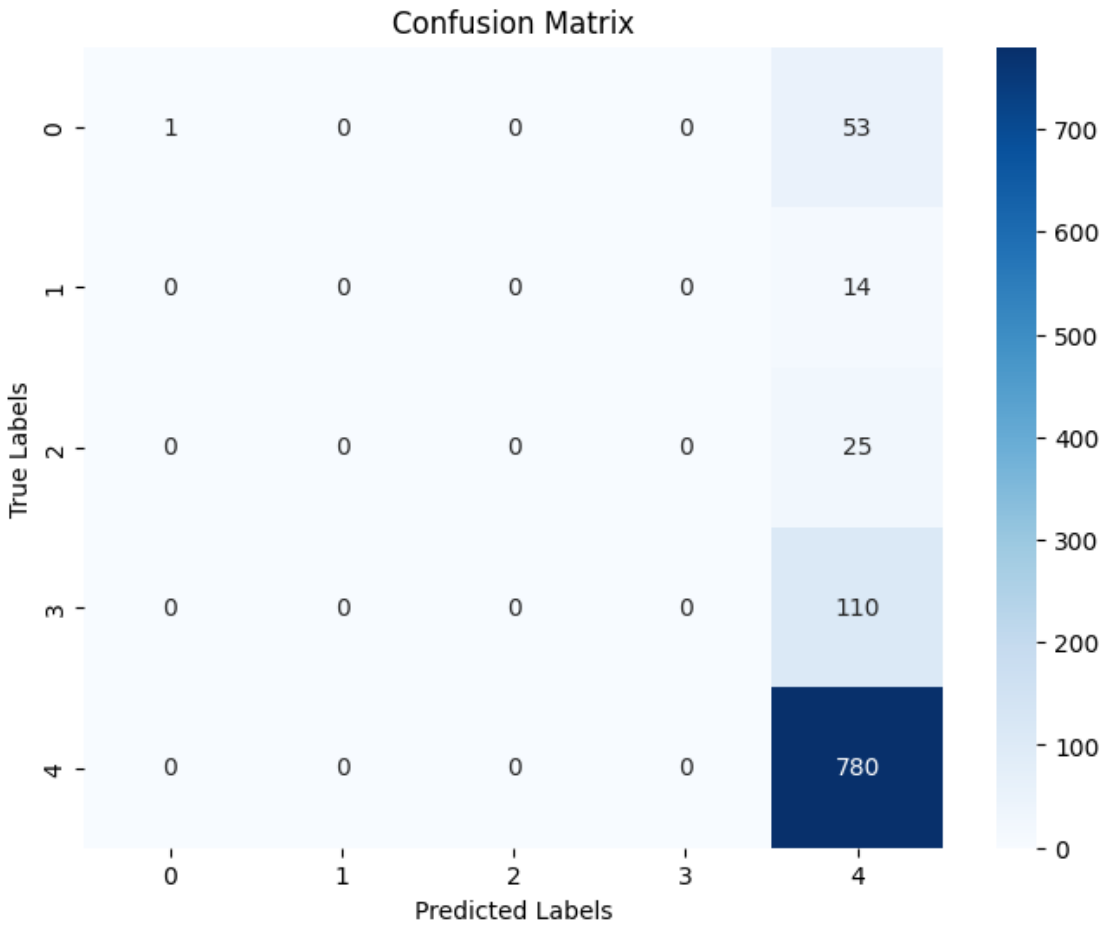
[29]:
```
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(df['reviewText'],
                                                    df['overall'], test_size=0.
 ↪2, random_state=42)
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)
model = SVC()
model.fit(X_train_vectorized, y_train)
```

```
y_pred = model.predict(X_test_vectorized)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
print("Accuracy: ", accuracy)
print(report)
print("Classification Report:\n", report)
```

```
Accuracy:  0.7945066124109867
              precision    recall  f1-score   support

         1.0       1.00      0.02      0.04        54
         2.0       0.00      0.00      0.00        14
         3.0       0.00      0.00      0.00        25
         4.0       0.00      0.00      0.00       110
         5.0       0.79      1.00      0.89       780

    accuracy                           0.79       983
   macro avg       0.36      0.20      0.18       983
weighted avg       0.69      0.79      0.70       983

Classification Report:
               precision    recall  f1-score   support

         1.0       1.00      0.02      0.04        54
         2.0       0.00      0.00      0.00        14
         3.0       0.00      0.00      0.00        25
         4.0       0.00      0.00      0.00       110
         5.0       0.79      1.00      0.89       780

    accuracy                           0.79       983
   macro avg       0.36      0.20      0.18       983
weighted avg       0.69      0.79      0.70       983


/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
```

```
  _warn_prf(average, modifier, msg_start, len(result))
```

[30]:
```
import seaborn as sns
from sklearn.metrics import confusion_matrix
import matplotlib.pyplot as plt
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, cmap='Blues', fmt='d')
plt.title('Confusion Matrix')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
```



[39]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer

X_train, X_test, y_train, y_test = train_test_split(df['reviewText'],
```

```
                                                    df['overall'],
                                                         test_size=0.2,
                                                         random_state=42)
vectorizer = CountVectorizer()
X_train_vectorized = vectorizer.fit_transform(X_train)
X_test_vectorized = vectorizer.transform(X_test)
model = RandomForestClassifier()
model.fit(X_train_vectorized, y_train)
y_pred = model.predict(X_test_vectorized)
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
print("Accuracy: ", accuracy)
print("Classification Report:\n", report)
```

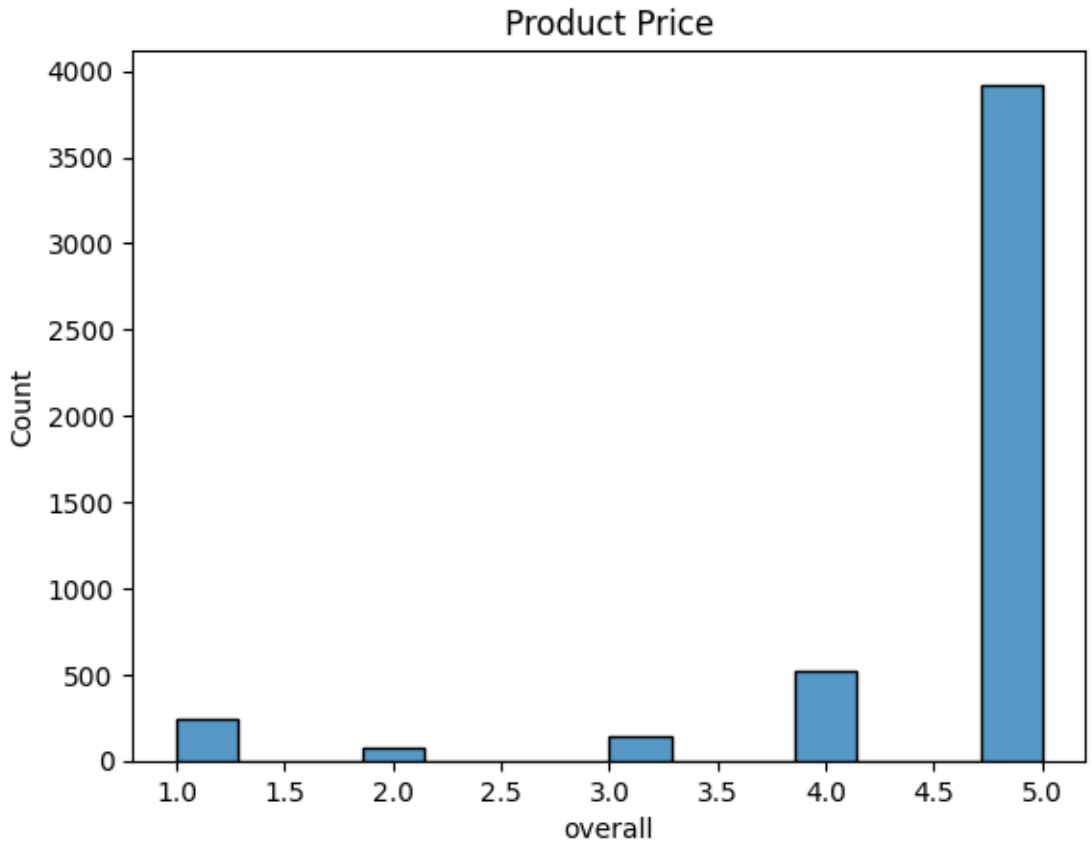
```
Accuracy:  0.7965412004069176
Classification Report:
               precision    recall  f1-score   support

         1.0       1.00      0.06      0.11        54
         2.0       0.00      0.00      0.00        14
         3.0       0.00      0.00      0.00        25
         4.0       0.00      0.00      0.00       110
         5.0       0.80      1.00      0.89       780

    accuracy                           0.80       983
   macro avg       0.36      0.21      0.20       983
weighted avg       0.69      0.80      0.71       983


/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

[42]:

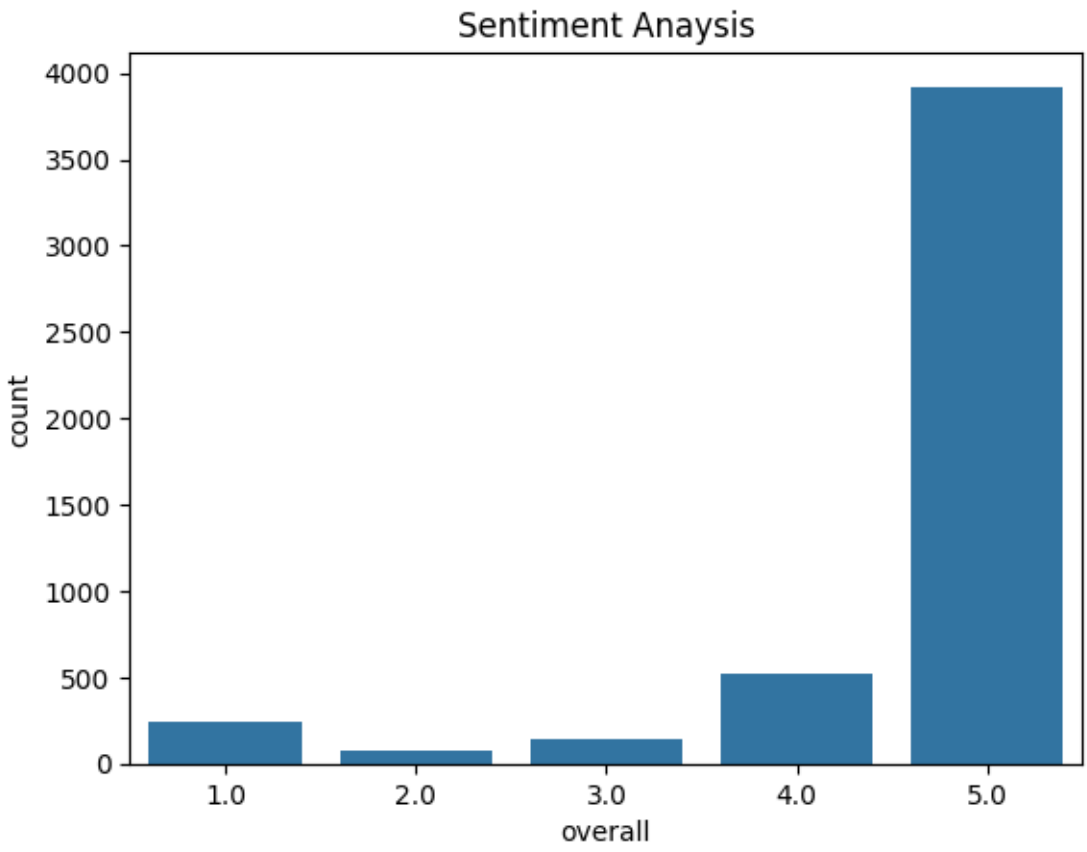
```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Verify the available columns in your DataFrame
print(df.columns)

# If you have a column representing price, replace 'actual_price_column' with␣
 ↪the correct column name
sns.histplot(df['overall'])

plt.title('Product Price')
plt.show()
```

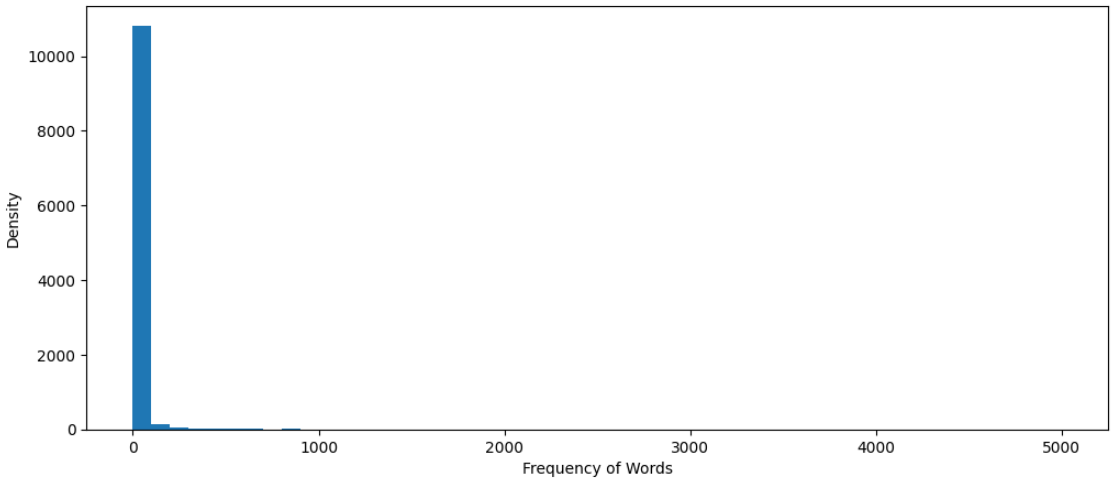
```
Index(['Unnamed: 0', 'reviewerName', 'overall', 'reviewText', 'reviewTime',
       'day_diff', 'helpful_yes', 'helpful_no', 'total_vote',
       'score_pos_neg_diff', 'score_average_rating', 'wilson_lower_bound'],
      dtype='object')
```



[45]:
```
sns.countplot(data=df, x='overall')
plt.title('Sentiment Anaysis')
plt.show()
```



[46]:
```
import matplotlib.pyplot as plt
plt.figure(figsize=(12, 5))
plt.hist(features_counts_df['counts'], bins=50, range=(0, 5000))
plt.xlabel('Frequency of Words')
plt.ylabel('Density')
plt.show()
```


10000
8000
6000
4000
2000
0
Density
0
1000
2000
3000
4000
5000
Frequency of Words