In [24]:

```
from sklearn.svm import LinearSVC
import pandas as pd
import numpy as np
import seaborn as sns
import re
import string
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics import accuracy_score, classification_report, confusion_mat
```

In [2]:

```
dp=pd.read_csv(r"C:\Users\SENAPATHI REDDY.K\Downloads\_MConverter.eu_imdb_review
```

In [3]:

```
dp.head()
```

Out[3]:

	review	sentiment
0	One of the other reviewers has mentioned that ...	positive
1	A wonderful little production. <br /><br />The...	positive
2	I thought this was a wonderful way to spend ti...	positive
3	Basically there's a family where a little boy ...	negative
4	Petter Mattei's "Love in the Time of Money" is...	positive

In [4]:

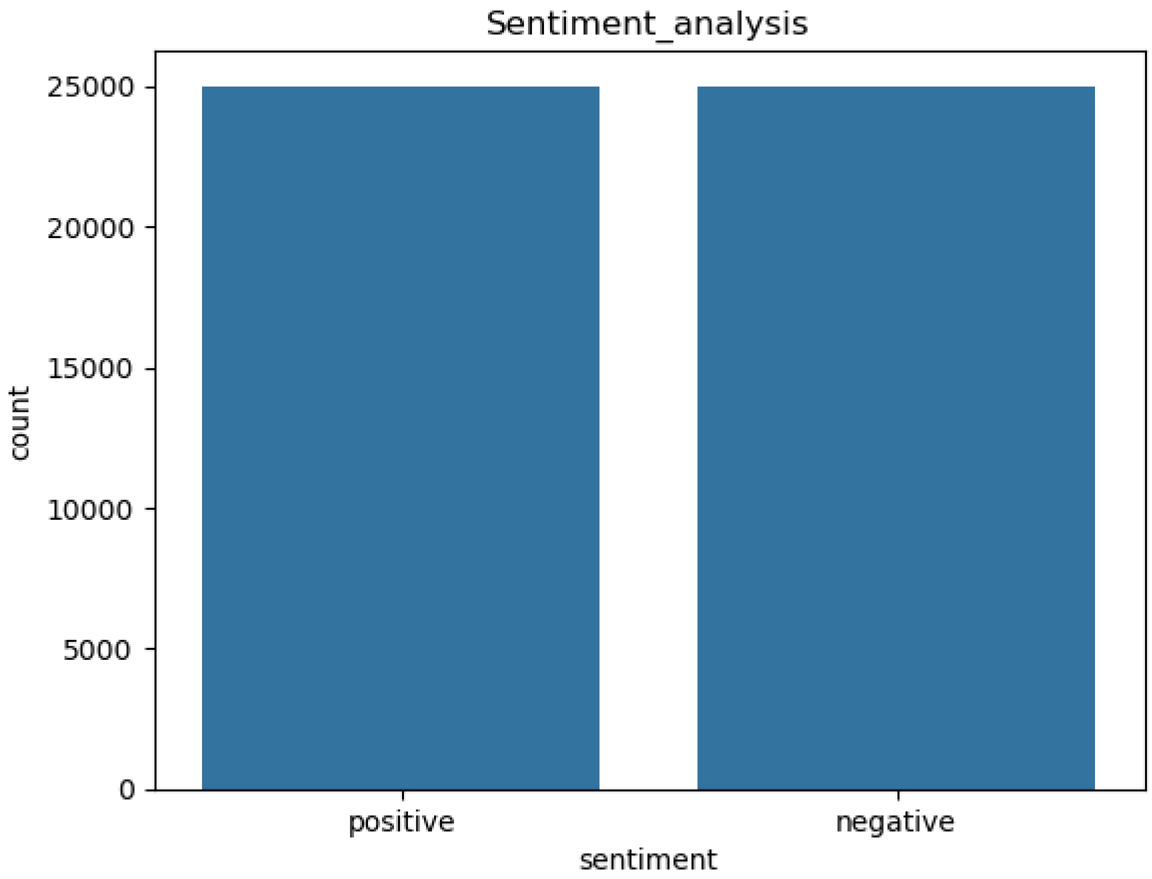
```
dp.describe()
```

Out[4]:

	review	sentiment
count	50000	50000
unique	49582	2
top	Loved today's show!!! It was a variety and not...	positive
freq	5	25000

In [7]:

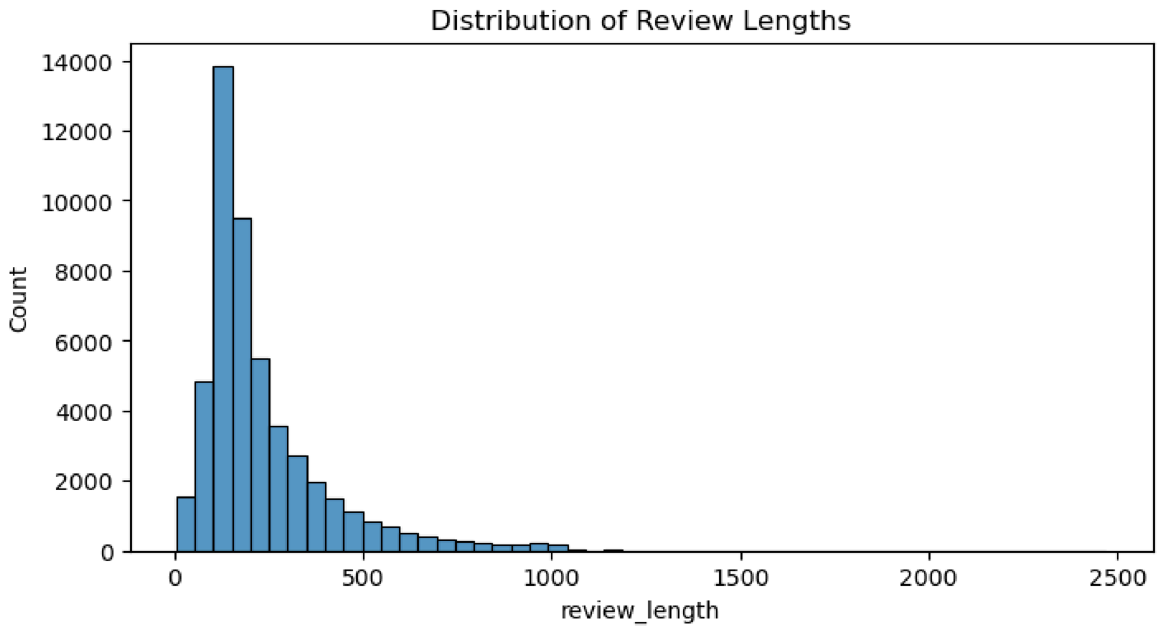
```
sns.countplot(x="sentiment",data=dp)
plt.title("Sentiment_analysis")
plt.show()
```



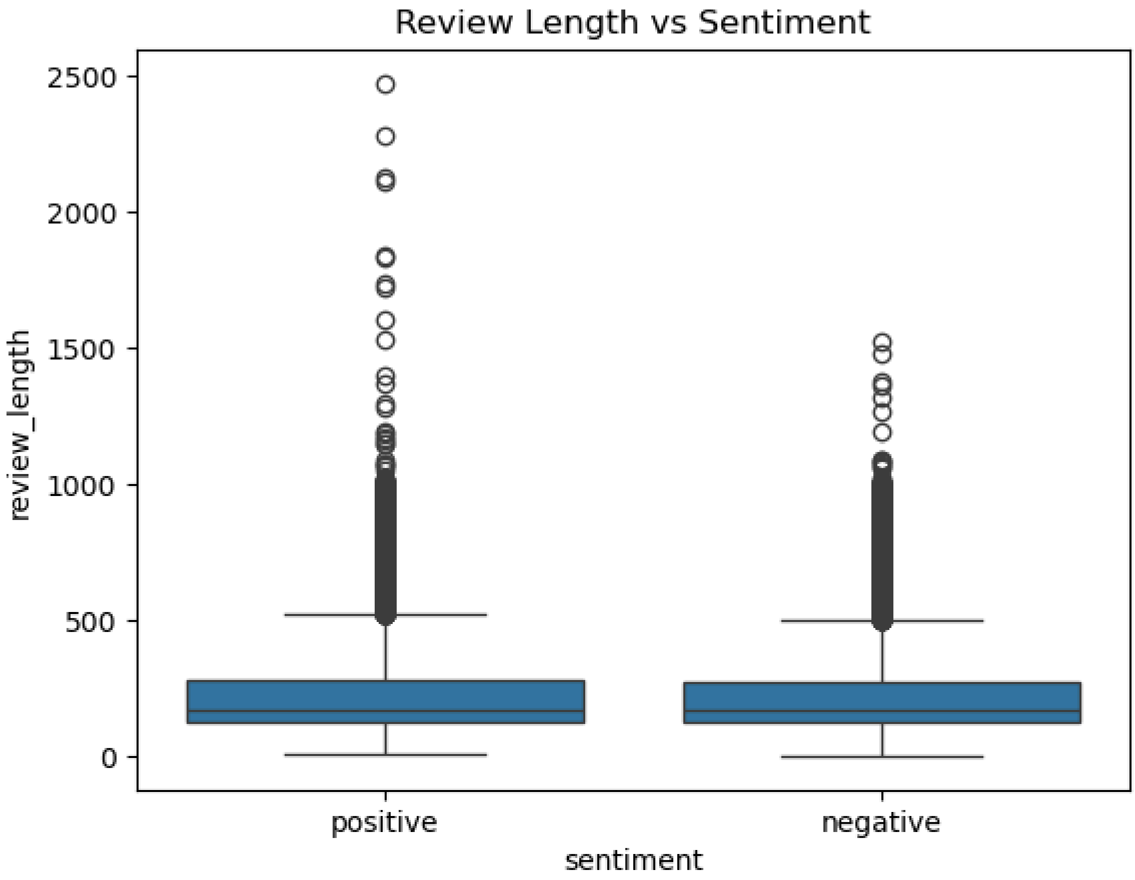
In [9]:
```
dp_encoded = pd.get_dummies(
    dp,
    columns=["sentiment"],
    drop_first=True,
    dtype=int
)
```

In [13]:
```
dp["review_length"]=dp["review"].apply((lambda x: len(x.split())))
```

In [14]:
```
plt.figure(figsize=(8,4))
sns.histplot(dp['review_length'], bins=50)
plt.title("Distribution of Review Lengths")
plt.show()
```



```
In [16]: sns.boxplot(x='sentiment', y='review_length', data=dp)
         plt.title("Review Length vs Sentiment")
         plt.show()
```



```
In [17]: def clean_text(text):
             text=text.lower()
             text = re.sub(r"<.*?>", "", text)
             text = re.sub(r"[^\w\s]", "", text)
             text = re.sub(r"\d+", "", text)
             return text
```

In [20]:

```
dp['clean_review'] = dp['review'].apply(clean_text)
```

In [22]:

```
x_train, x_test, y_train, y_test = train_test_split(
    dp['clean_review'],
    dp['sentiment'],
    test_size=0.2,
    random_state=42
)
```

In [29]:

```
vectorizer = TfidfVectorizer(
    max_features=5000,
    stop_words='english'
)
```

In [32]:

```
x_train_vec = vectorizer.fit_transform(x_train)
x_test_vec = vectorizer.transform(x_test)
```

In [33]:

```
svm_model = LinearSVC()
svm_model.fit(x_train_vec, y_train)
```

Out[33]:

```
▾ LinearSVC ⓘ ?
LinearSVC()
```

In [34]:

```
y_pred_svm = svm_model.predict(X_test_vec)

print("SVM Accuracy:", accuracy_score(y_test, y_pred_svm))
print("\nClassification Report:\n", classification_report(y_test, y_pred_svm))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred_svm))
```

```
SVM Accuracy: 0.8724

Classification Report:
               precision    recall  f1-score   support

    negative       0.88      0.86      0.87      4961
    positive       0.86      0.89      0.87      5039

    accuracy                           0.87     10000
   macro avg       0.87      0.87      0.87     10000
weighted avg       0.87      0.87      0.87     10000


Confusion Matrix:
 [[4262  699]
 [ 577 4462]]
```

In [ ]: