```
import nltk
import string
import pandas as pd
import matplotlib.pyplot as plt

from nltk.corpus import movie_reviews, stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
```

```
nltk.download('movie_reviews')
nltk.download('stopwords')
nltk.download('punkt')
```

```
[nltk_data] Downloading package movie_reviews to /root/nltk_data...
[nltk_data]   Unzipping corpora/movie_reviews.zip.
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Unzipping tokenizers/punkt.zip.
True
```

```
positive_reviews = [
    movie_reviews.raw(fileid)
    for fileid in movie_reviews.fileids('pos')
]

negative_reviews = [
    movie_reviews.raw(fileid)
    for fileid in movie_reviews.fileids('neg')
]
```

```
stop_words = set(stopwords.words('english'))

def preprocess(text):
    tokens = nltk.word_tokenize(text.lower())
    tokens = [
        word for word in tokens
        if word not in stop_words and word not in string.punctuation
    ]
    return " ".join(tokens)
```

```
nltk.download('punkt_tab')
positive_clean = [preprocess(review) for review in positive_reviews]
negative_clean = [preprocess(review) for review in negative_reviews]
```

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

```
tfidf_pos = TfidfVectorizer(max_features=3000)
tfidf_neg = TfidfVectorizer(max_features=3000)

pos_matrix = tfidf_pos.fit_transform(positive_clean)
neg_matrix = tfidf_neg.fit_transform(negative_clean)
```

```
def get_top_terms(tfidf_matrix, vectorizer, top_n=15):
    scores = tfidf_matrix.mean(axis=0).A1
    terms = vectorizer.get_feature_names_out()
    tfidf_scores = pd.DataFrame(
        {'term': terms, 'score': scores}
    ).sort_values(by='score', ascending=False)
    return tfidf_scores.head(top_n)
```

```
top_pos = get_top_terms(pos_matrix, tfidf_pos)
top_neg = get_top_terms(neg_matrix, tfidf_neg)

print("Top TF-IDF Terms in Positive Reviews")
print(top_pos)

print("\nTop TF-IDF Terms in Negative Reviews")
print(top_neg)
```

```
Top TF-IDF Terms in Positive Reviews
            term     score
1020        film  0.069678
1771       movie  0.044395
1898         one  0.041486
1566        like  0.029298
2537       story  0.023930
1173        good  0.023619
1562        life  0.022879
2711        time  0.022603
105         also  0.022031
2918        well  0.021969
445    character  0.021070
901         even  0.021064
2982       would  0.020983
447   characters  0.020831
2794         two  0.020259

Top TF-IDF Terms in Negative Reviews
            term     score
1009        film  0.066423
1766       movie  0.054195
1879         one  0.043028
1560        like  0.032026
884         even  0.025852
1156        good  0.024383
2978       would  0.024224
223          bad  0.023346
2699        time  0.023289
2526       story  0.022397
1129         get  0.022161
```

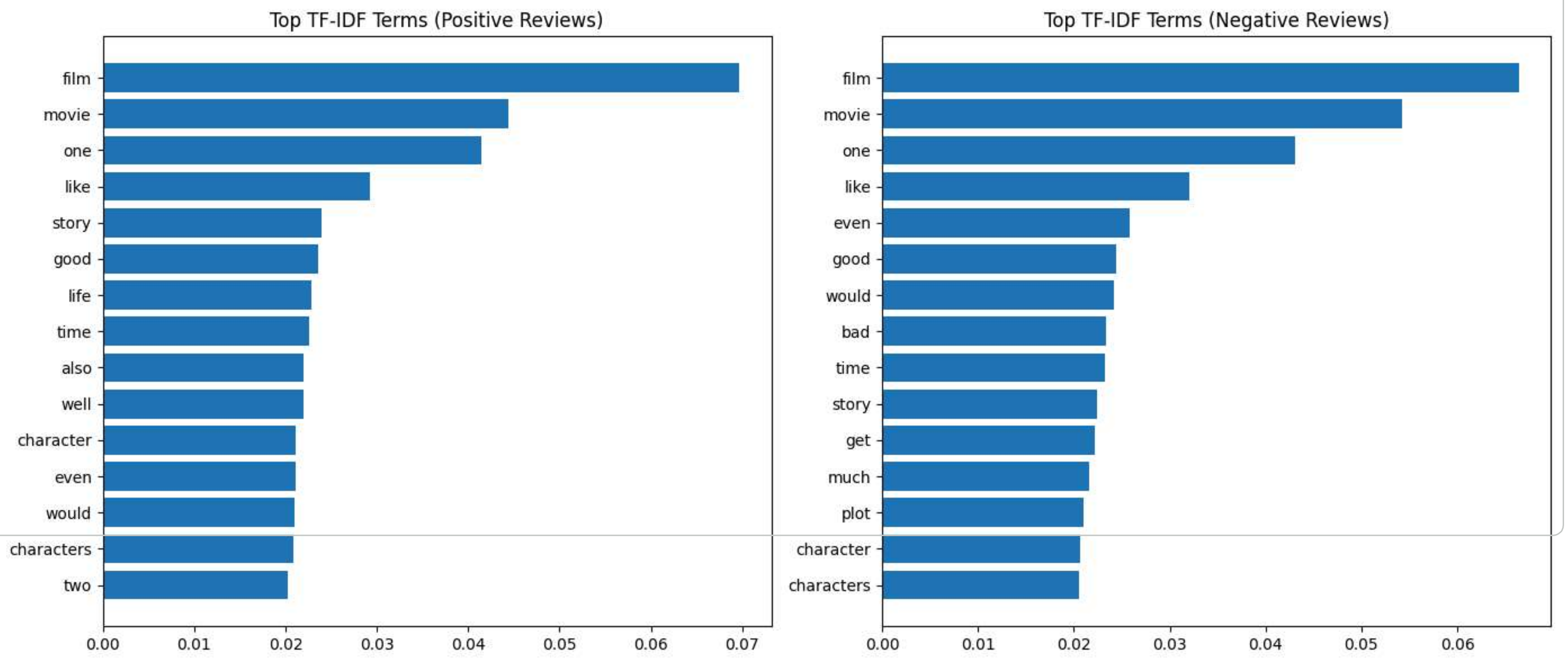
```
1773        much  0.021639
1999        plot  0.020999
447    character  0.020664
449   characters  0.020563
```

```
fig, axes = plt.subplots(1, 2, figsize=(14, 6))

# Positive Reviews
axes[0].barh(top_pos['term'], top_pos['score'])
axes[0].set_title("Top TF-IDF Terms (Positive Reviews)")
axes[0].invert_yaxis()

# Negative Reviews
axes[1].barh(top_neg['term'], top_neg['score'])
axes[1].set_title("Top TF-IDF Terms (Negative Reviews)")
axes[1].invert_yaxis()

plt.tight_layout()
plt.show()
```


Top TF-IDF Terms (Positive Reviews)
film
movie
one
like
story
good
life
time
also
well
character
even
would
characters
two
Top TF-IDF Terms (Negative Reviews)
film
movie
one
like
even
good
would
bad
time
story
get
much
plot
character
characters