Train

April 18, 2025

[1]:
```
# Import necessary libraries
import pandas as pd
import re
import joblib
from io import StringIO
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
```

[2]:
```
# Step 1: Load and Explore Data
# Load the dataset
df = pd.read_csv('Dataset.csv')
print('Dataset shape:', df.shape)
print(df.head())
```

```
Dataset shape: (50000, 2)
                                              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
```

[3]:
```
# Step 2: Text Preprocessing (Cleaning)
# Define a function to clean text
def clean_text(text):
    text = re.sub(r'<.*?>', '', text)  # Remove HTML tags
    text = re.sub(r'[^a-zA-Z\s]', '', text)  # Remove non-letter characters
    return text.lower()  # Convert to lowercase
```

[4]:
```
# Apply cleaning to the 'review' column
df['clean_review'] = df['review'].apply(clean_text)
print(df[['review', 'clean_review']].head())
```

```
                                              review  \
0  One of the other reviewers has mentioned that …
1  A wonderful little production. <br /><br />The…
```

```
2  I thought this was a wonderful way to spend ti…
3  Basically there's a family where a little boy …
4  Petter Mattei's "Love in the Time of Money" is…
```

```
                                        clean_review
0  one of the other reviewers has mentioned that …
1  a wonderful little production the filming tech…
2  i thought this was a wonderful way to spend ti…
3  basically theres a family where a little boy j…
4  petter matteis love in the time of money is a …
```

[5]:
```
# Step 3: Feature Engineering (TF-IDF Vectorization)
# Initialize TF-IDF vectorizer
vectorizer = TfidfVectorizer(max_features=9000)

# Fit and transform the cleaned text into numerical features
X_features = vectorizer.fit_transform(df['clean_review'])

# Map sentiment labels to binary values
df['sentiment_val'] = df['sentiment'].map({'positive': 1, 'negative': 0})
print('TF-IDF matrix shape:', X_features.shape)
```

```
TF-IDF matrix shape: (50000, 9000)
```

[7]:
```
# Step 4: Model Training (Logistic Regression)
# Define features (X) and labels (y)
X = X_features
y = df['sentiment_val']

# Split data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Train Logistic Regression model
model = LogisticRegression(max_iter=2000)
model.fit(X_train, y_train)
```

[7]:
```
LogisticRegression(max_iter=2000)
```

[8]:
```
# Step 5: Model Evaluation
# Generate predictions on test data
y_pred = model.predict(X_test)

# Calculate accuracy
accuracy = accuracy_score(y_test, y_pred)
print('Test Accuracy:', accuracy)
```

```
# Classification report for detailed evaluation
print('\nClassification Report:\n', classification_report(y_test, y_pred))
```

```
Test Accuracy: 0.8952

Classification Report:
              precision    recall  f1-score   support

           0       0.90      0.88      0.89      4961
           1       0.89      0.91      0.90      5039

    accuracy                           0.90     10000
   macro avg       0.90      0.90      0.90     10000
weighted avg       0.90      0.90      0.90     10000
```

[9]:
```
# Step 6: Testing on New Examples
# Define a sample new review
sample_review = "I absolutely loved this movie, it was fantastically directed␣
 ↪and brilliantly acted."

# Clean the sample review
clean_sample = clean_text(sample_review)

# Transform the review using the TF-IDF vectorizer
sample_features = vectorizer.transform([clean_sample])
```

[10]:
```
# Predict sentiment
predicted_sentiment = model.predict(sample_features)[0]
sentiment_label = "positive" if predicted_sentiment == 1 else "negative"

print('Sample Review:', sample_review)
print('Cleaned Review:', clean_sample)
print('Predicted Sentiment:', sentiment_label)
```

```
Sample Review: I absolutely loved this movie, it was fantastically directed and
brilliantly acted.
Cleaned Review: i absolutely loved this movie it was fantastically directed and
brilliantly acted
Predicted Sentiment: positive
```

[11]:
```
#Saving Model
joblib.dump(model, "logistic_regression_model.pkl")
```

[11]:
```
['logistic_regression_model.pkl']
```