```
# Importing libraries
!pip install -q scikit-learn transformers torch

import pandas as pd
import numpy as np
import torch
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
from transformers import pipeline
```

```
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 3.6 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 38.5 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 29.5 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 29.4 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 3.0 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 5.7 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 13.9 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 7.6 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.2 MB/s eta 0:00:00
 ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 106.5 MB/s eta 0:00:00
```

```
# Load csv file
df = pd.read_csv('/content/IMDB Dataset.csv')

df.head()
```

	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

```
# Map sentiment to binary labels
df['label'] = df['sentiment'].map({'positive': 1, 'negative': 0})

# Defining features and labels
X = df['review'].values
y = df['label'].values

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
```

```
# TF-IDF Vectorization
tfidf = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf.fit_transform(X_train)
X_test_tfidf = tfidf.transform(X_test)

# Training Logistic Regression
lr_model = LogisticRegression(max_iter=1000)
lr_model.fit(X_train_tfidf, y_train)

# Predict
y_pred_lr = lr_model.predict(X_test_tfidf)

# Evaluation
print("=== Logistic Regression Performance ===")
print(classification_report(y_test, y_pred_lr))
print("Accuracy:", accuracy_score(y_test, y_pred_lr))
```

```
=== Logistic Regression Performance ===
              precision    recall  f1-score   support

           0       0.90      0.89      0.89      5000
           1       0.89      0.90      0.89      5000
```

```
    accuracy                           0.89     10000
   macro avg       0.89      0.89      0.89     10000
weighted avg       0.89      0.89      0.89     10000

Accuracy: 0.8924
```

```
# Loading BERT pipeline
bert_classifier = pipeline(
    "sentiment-analysis",
    model="textattack/bert-base-uncased-imdb",
    truncation=True,
    device=0
)

# Predicting
bert_preds = []
for text in X_test:
    if len(text) > 1000:
        text = text[:1000]

    output = bert_classifier(text)[0]['label']
    label = 1 if output == 'LABEL_1' else 0
    bert_preds.append(label)

# Evaluating
print("\n=== API-based BERT Performance ===")
print(classification_report(y_test, bert_preds))
print("Accuracy:", accuracy_score(y_test, bert_preds))
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
config.json: 100%                                   511/511 [00:00<00:00, 10.3kB/s]
pytorch_model.bin: 100%                             438M/438M [00:02<00:00, 224MB/s]
tokenizer_config.json: 100%                         48.0/48.0 [00:00<00:00, 5.35kB/s]
model.safetensors: 100%                             438M/438M [00:02<00:00, 152MB/s]
vocab.txt: 100%                                     232k/232k [00:00<00:00, 4.08MB/s]
special_tokens_map.json: 100%                       112/112 [00:00<00:00, 7.59kB/s]
Device set to use cuda:0
You seem to be using the pipelines sequentially on GPU. In order to maximize efficiency please use a dataset

=== BERT Performance ===
              precision    recall  f1-score   support

           0       0.95      0.96      0.95      5000
           1       0.96      0.95      0.95      5000

    accuracy                           0.95     10000
   macro avg       0.95      0.95      0.95     10000
weighted avg       0.95      0.95      0.95     10000

Accuracy: 0.9523
```

```
# Comparison summary
results = {
    'Model': ['Logistic Regression', 'API-based BERT'],
    'Accuracy': [
        accuracy_score(y_test, y_pred_lr),
        accuracy_score(y_test, bert_preds)
    ]
}

results_df = pd.DataFrame(results)
print(results_df)
```

```
                 Model  Accuracy
0  Logistic Regression    0.8924
1       API-based BERT    0.9523
```