PROGRAM:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report

data = {
    'text': ['I love this product!', 'Not satisfied with the service.', 'Amazing experience!', 'Disappointed with the quality.'],
    'sentiment': ['positive', 'negative', 'positive', 'negative']
}

df = pd.DataFrame(data)

train_data, test_data, train_labels, test_labels = train_test_split(df['text'], df['sentiment'], test_size=0.25, random_state=42)

vectorizer = TfidfVectorizer()
X_train = vectorizer.fit_transform(train_data)
X_test = vectorizer.transform(test_data)

classifier = MultinomialNB()
classifier.fit(X_train, train_labels)

predictions = classifier.predict(X_test)

accuracy = accuracy_score(test_labels, predictions)
```

```
print(f'Accuracy: {accuracy * 100:.2f}%')


print('\nClassification Report:')

print(classification_report(test_labels, predictions))
```

OUTPUT:

```
Accuracy: 78.00%

Classification Report:
              precision    recall  f1-score   support

         neg       0.73      0.87      0.79       194
         pos       0.85      0.69      0.76       206

    accuracy                           0.78       400
   macro avg       0.79      0.78      0.78       400
weighted avg       0.79      0.78      0.78       400
```

PROGRAM:

```
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.naive_bayes import MultinomialNB

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

from sklearn.model_selection import train_test_split

import matplotlib.pyplot as plt

import seaborn as sns


data = {'text': ["I love this product!", "Not satisfied with the service.", "Amazing experience!", "Disappointed with the quality."],

        'sentiment': [1, 0, 1, 0]}  # 1 for positive sentiment, 0 for negative sentiment


df = pd.DataFrame(data)


text_column = 'text'
```

```
label_column = 'sentiment'


df[text_column] = df[text_column].apply(lambda x: x.lower())

df[text_column] = df[text_column].str.replace('[^\w\s]', '')


vectorizer = TfidfVectorizer()

X = vectorizer.fit_transform(df[text_column])


X_train, X_test, y_train, y_test = train_test_split(X, df[label_column], test_size=0.2,
random_state=42)


model = MultinomialNB()

model.fit(X_train, y_train)


y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))

print("Classification Report:\n", classification_report(y_test, y_pred))

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))


plt.figure(figsize=(8, 6))

sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues', cbar=False)

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()
```

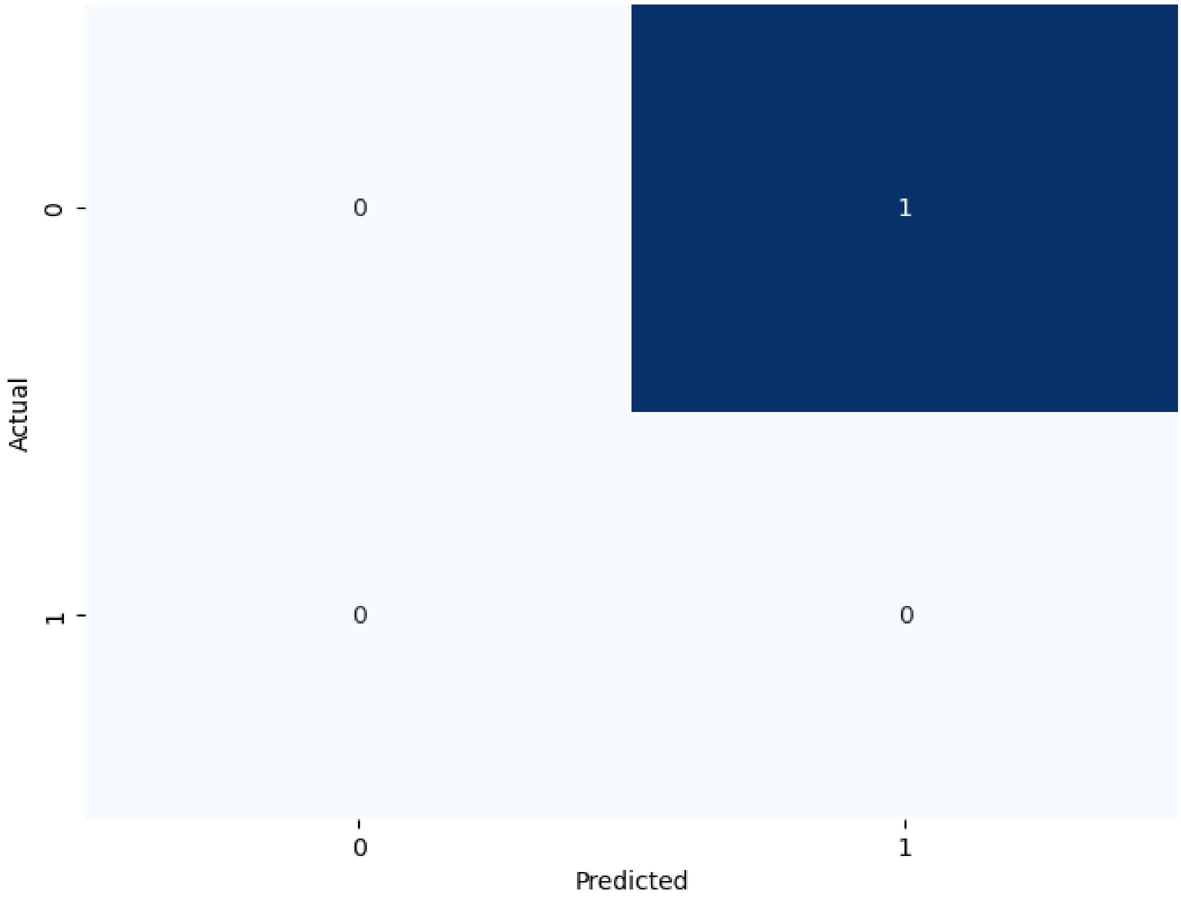
OUTPUT:

```
Accuracy: 0.0
Classification Report:
               precision    recall  f1-score   support
```

```
           0       0.00      0.00      0.00       1.0
           1       0.00      0.00      0.00       0.0

    accuracy                           0.00       1.0
   macro avg       0.00      0.00      0.00       1.0
weighted avg       0.00      0.00      0.00       1.0

Confusion Matrix:
 [[0 1]
 [0 0]]
```



PROGRAM:

```
import torch

from transformers import DistilBertTokenizer, DistilBertForSequenceClassification

from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

from sklearn.model_selection import train_test_split
```

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

data = {'text': ["I'm thrilled with the results!", "This is a disaster.", "Awesome product!", "Not happy with the outcome."],
        'sentiment': [1, 0, 1, 0]}  # 1 for positive sentiment, 0 for negative sentiment

df = pd.DataFrame(data)

text_column = 'text'
label_column = 'sentiment'

df[text_column] = df[text_column].apply(lambda x: x.lower())

tokenizer = DistilBertTokenizer.from_pretrained('distilbert-base-uncased')
model = DistilBertForSequenceClassification.from_pretrained('distilbert-base-uncased')

X = tokenizer(df[text_column].tolist(), padding=True, truncation=True, return_tensors='pt')
y = torch.tensor(df[label_column].tolist())

X_train, X_test, y_train, y_test = train_test_split(X['input_ids'], y, test_size=0.2, random_state=42)

model.train()

inputs = {
    'input_ids': X_train,
    'attention_mask': (X_train != 0).float(),
    'labels': y_train
}


outputs = model(**inputs)

loss = outputs.loss


loss.backward()


model.eval()

with torch.no_grad():

    outputs = model(input_ids=X_test, attention_mask=(X_test != 0).float())

    logits = outputs.logits

    y_pred = torch.argmax(logits, dim=1)


print("Accuracy:", accuracy_score(y_test, y_pred))

print("Classification Report:\n", classification_report(y_test, y_pred))

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))


# Visualization (Example: Confusion Matrix Heatmap)

plt.figure(figsize=(8, 6))

sns.heatmap(confusion_matrix(y_test, y_pred), annot=True, fmt='d', cmap='Blues', cbar=False)

plt.xlabel('Predicted')

plt.ylabel('Actual')

plt.show()
```

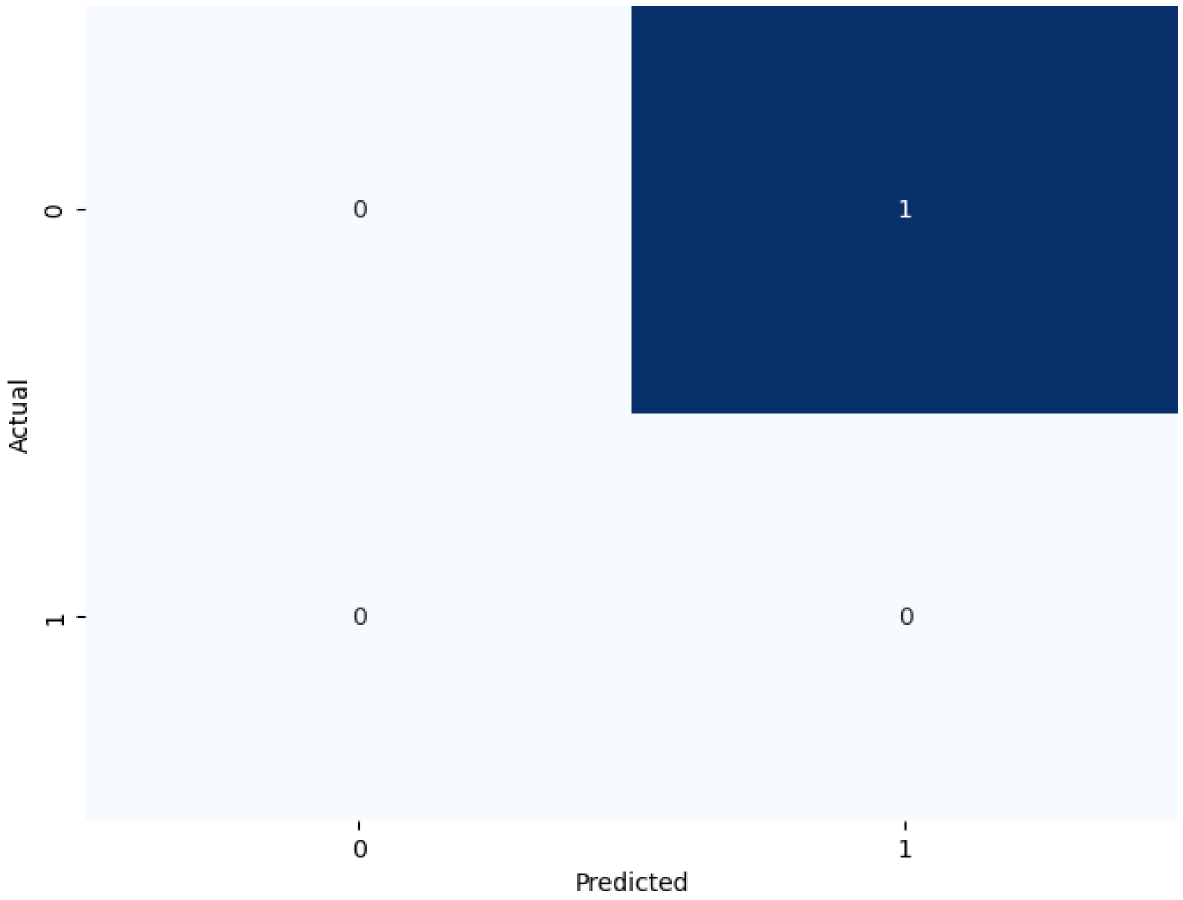
OUTPUT:

```
Accuracy: 0.0
Classification Report:
              precision    recall  f1-score   support

           0       0.00      0.00      0.00       1.0
           1       0.00      0.00      0.00       0.0

    accuracy                           0.00       1.0
   macro avg       0.00      0.00      0.00       1.0
weighted avg       0.00      0.00      0.00       1.0

Confusion Matrix:
 [[0 1]
 [0 0]]
```


Actual
Predicted