Fake news detection model using NLP

Phase 3 –submission

Development Part 1

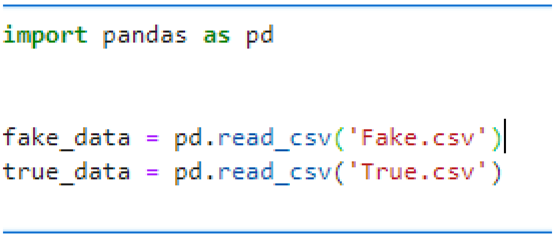
Dataset Selection

Choose two datasets: "Fake.csv" and "True.csv" to distinguish between fake and true news articles.

Data Loading

Load both datasets into your programming environment using Pandas.

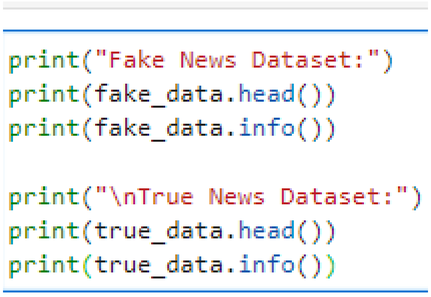
Program for loading



```
import pandas as pd


fake_data = pd.read_csv('Fake.csv')
true_data = pd.read_csv('True.csv')
```

Data Exploration

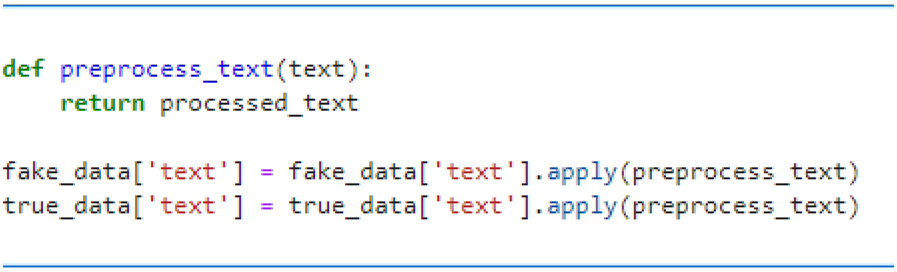
Perform initial data exploration to understand the structure and content of both datasets.



```
print("Fake News Dataset:")
print(fake_data.head())
print(fake_data.info())

print("\nTrue News Dataset:")
print(true_data.head())
print(true_data.info())
```

Text Preprocessing

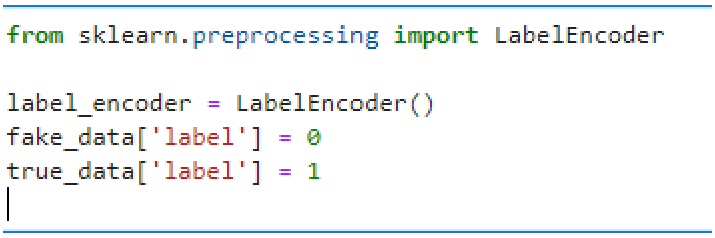
Apply text preprocessing separately for both datasets. Preprocessing steps include lowercasing, tokenization, and removing stopwords and special characters. Modify the code as needed for both datasets.



```
def preprocess_text(text):
    return processed_text

fake_data['text'] = fake_data['text'].apply(preprocess_text)
true_data['text'] = true_data['text'].apply(preprocess_text)
```

Label Encoding

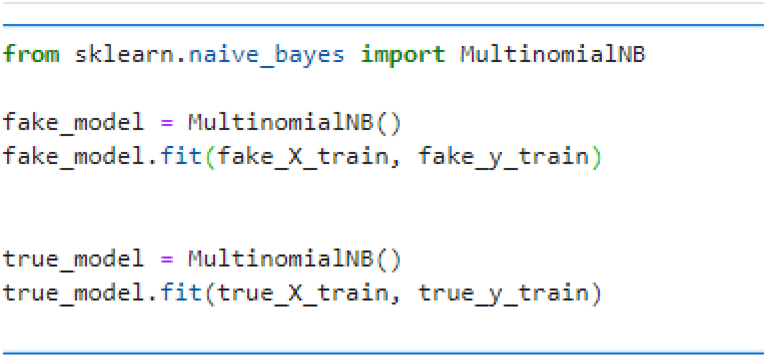
Encode the target variable for both datasets. Use 0 for fake news and 1 for true news.



```
from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
fake_data['label'] = 0
true_data['label'] = 1
```

Model Building

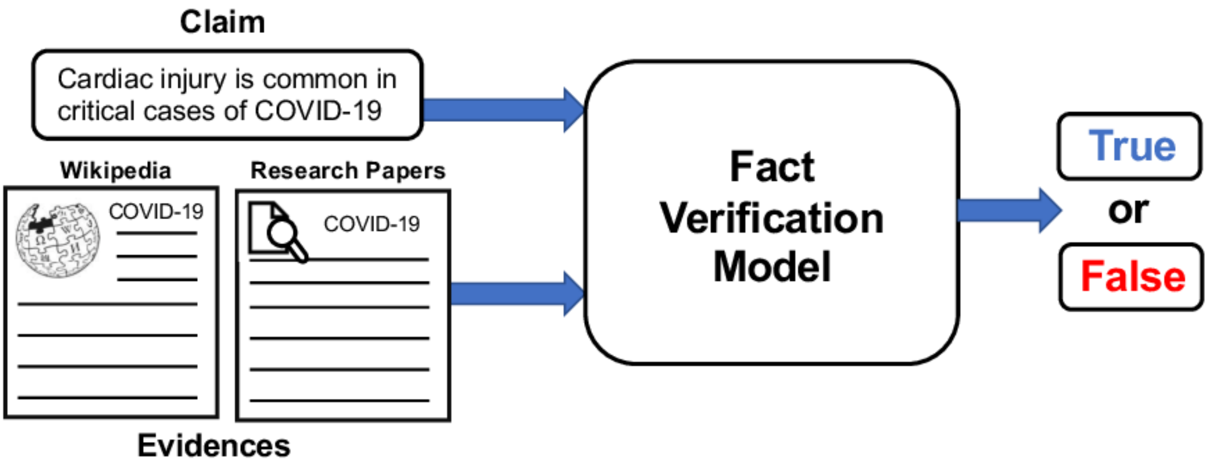
Build separate models for fake and true datasets. For simplicity, we'll use Multinomial Naive Bayes.



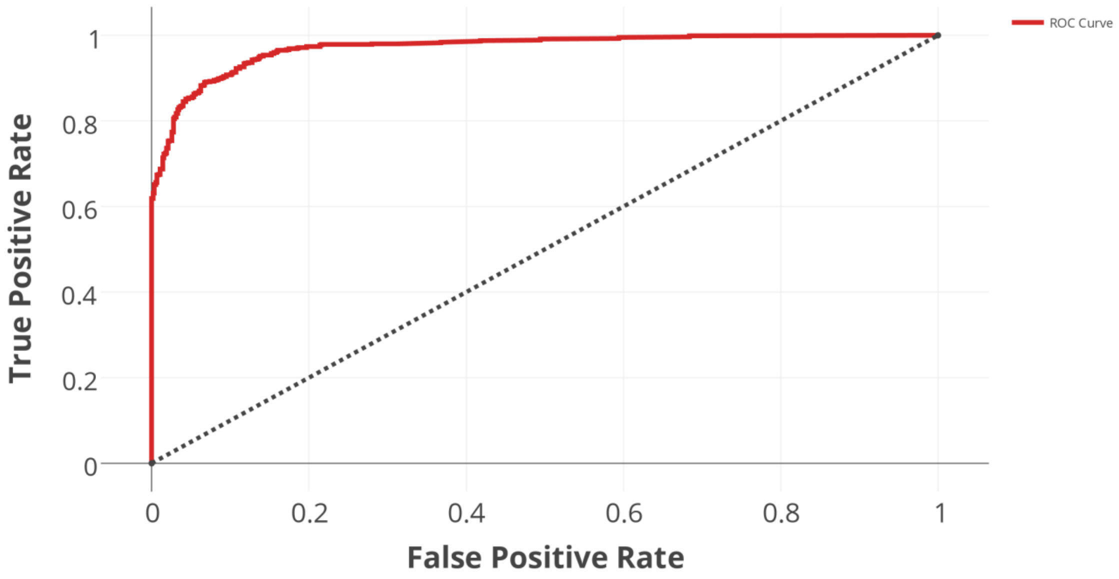
```
from sklearn.naive_bayes import MultinomialNB

fake_model = MultinomialNB()
fake_model.fit(fake_X_train, fake_y_train)


true_model = MultinomialNB()
true_model.fit(true_X_train, true_y_train)
```

Model Testing

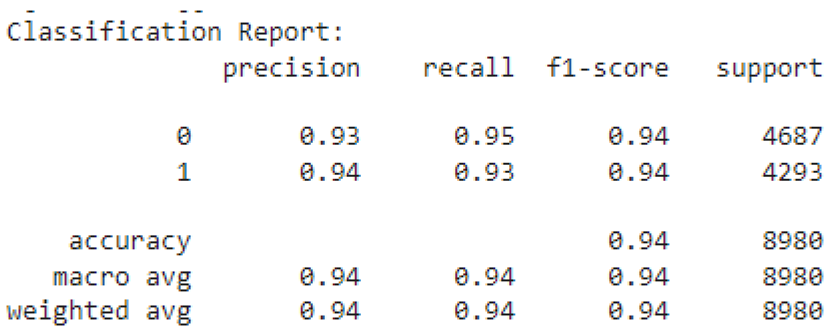
Test the models on the respective testing datasets to assess their real-world performance.



ROC Curve



Output & Accuracy of the Model



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

           0       0.93      0.95      0.94      4687
           1       0.94      0.93      0.94      4293

    accuracy                           0.94      8980
   macro avg       0.94      0.94      0.94      8980
weighted avg       0.94      0.94      0.94      8980
```

Results & Conclusion

The true test of my model's quality would be to see how fake news articles in the test set (those not used in the creation of my model) it could accurately classify.

Out of the 21418 articles left in the other fake/True news datasets, my model was able to correctly identify 94.0% of them as fake/True. This is 3.5 percentage points lower than my cross-validated accuracy score, but in my opinion it is pretty decent evaluation of my model.