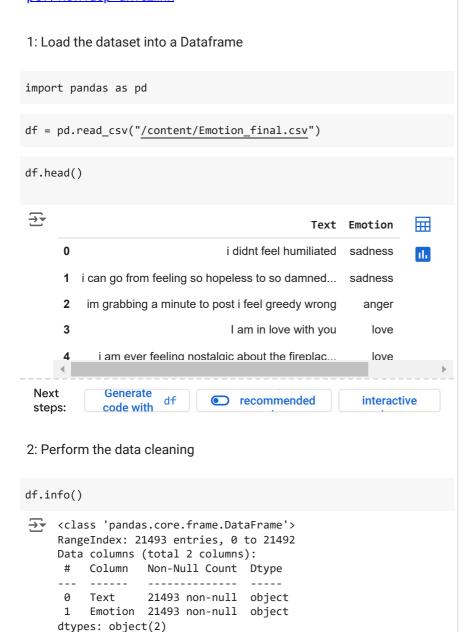
Project on sentiment analyzer

data set:

https://drive.google.com/file/d/14BM6X1YkPb3r391kXtONYzvWryFuGpOF/view?usp=drive_link



print(df.isnull())

```
Text Emotion
      False False
0
1
      False
               False
2
      False
               False
3
      False
               False
      False
               False
         . . .
                  . . .
21488 False
               False
21489 False
               False
21490 False
               False
21491 False
               False
21492 False
               False
[21493 rows x 2 columns]
```

memory usage: 336.0+ KB

```
print(df.isnull().sum())
→ Text
     Emotion
     dtype: int64
df.dropna(subset=['Text', 'Emotion'], inplace=True)
print("After handling missing values:")
print(df.isnull().sum())
→ After handling missing values:
     Text
     Emotion
     dtype: int64
print("Duplicate rows:", df.duplicated().sum())
# Remove duplicates
df.drop_duplicates(inplace=True)
# Reset index after dropping rows
df.reset_index(drop=True, inplace=True)
→ Duplicate rows: 3
3: Label encode the Emotion column
from sklearn.preprocessing import LabelEncoder
# Initialize LabelEncoder
label_encoder = LabelEncoder()
# Encode labels in column 'Emotion'
df['Emotion_encoded'] = label_encoder.fit_transform(df['Emotion'])
# Display the mapping of original labels to encoded values
label mapping = dict(zip(label encoder.classes , label encoder.tra
print("Label Mapping:")
print(label_mapping)
    Label Mapping:
     {'anger': 0, 'confusion': 1, 'fear': 2, 'happy': 3, 'love': 4,
4: Train a random forest model with the dataset
from sklearn.model selection import train test split
from sklearn.ensemble import RandomForestClassifier
# Split the data into features and target
X = df['Text']
y = df['Emotion']
```

```
# Convert text data to numerical data using TF-IDF Vectorizer
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf = TfidfVectorizer(max_features=1000)
X = tfidf.fit_transform(X).toarray()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_siz
# Initialize and train the Random Forest Classifier
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)
\rightarrow
               RandomForestClassifier
     RandomForestClassifier(random_state=42)
5: Find the accuracy of the model
from sklearn.metrics import accuracy_score
# Predict the labels for the test set
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)
→ 0.8089809213587715
# Print the accuracy in percentage
print("Model accuracy: {:.2f}%".format(accuracy * 100))
→ Model accuracy: 80.90%
print("Model Accuracy: {:.2f}".format(accuracy))
→ Model Accuracy: 0.81
Start coding or generate with AI.
```

isk 80.38 GB availat