```
from google.colab import files
import pandas as pd

# Upload the file
uploaded = files.upload()
```

Choose Files 4 files

- **Sports.csv**(text/csv) - 4974 bytes, last modified: 8/7/2024 - 100% done
- **Politics.csv**(text/csv) - 5529 bytes, last modified: 8/7/2024 - 100% done
- **Education.csv**(text/csv) - 5083 bytes, last modified: 8/7/2024 - 100% done
- **Finance.csv**(text/csv) - 5467 bytes, last modified: 8/7/2024 - 100% done

```
Saving Sports.csv to Sports.csv
Saving Politics.csv to Politics.csv
Saving Education.csv to Education.csv
Saving Finance.csv to Finance.csv
```

```
education=pd.read_csv("Education.csv")
finance=pd.read_csv("Finance.csv")
politics=pd.read_csv("Politics.csv")
sports=pd.read_csv("Sports.csv")
```

```
df = pd.concat([education, finance, politics, sports])
df
```

	Text	Label
0	The impact of educational reforms remains unce...	positive
1	Critics argue that recent improvements in the ...	negative
2	Innovative teaching methods have led to unexpe...	positive
3	Despite budget constraints, the school has man...	positive
4	The true effectiveness of online learning plat...	negative
...	...	...
51	Sports fandom can foster a sense of community ...	positive
52	Sports events offer a platform for showcasing ...	positive
53	The pressure to win in sports can overshadow t...	negative
54	Sports programs in schools play a crucial role...	positive
55	The commercialization of sports has led to exp...	negative

209 rows × 2 columns

Next steps: Generate code with df View recommended plots New interactive sheet

```
df['Label'].unique()
```

```
array(['positive', 'negative'], dtype=object)
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn.naive_bayes import MultinomialNB, BernoulliNB, GaussianNB
from sklearn.metrics import accuracy_score, classification_report
import nltk
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
import string
import re
import joblib
from joblib import load
from joblib import dump
```

```
# Download NLTK data files (only the first time)
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
[nltk_data] Downloading package wordnet to /root/nltk_data...
True
```

```
# Preprocess the text data
def preprocess_text(text):
    # Convert to lowercase
    text = text.lower()
    # Remove punctuation
    text = text.translate(str.maketrans('', '', string.punctuation))
    # Remove numbers
    text = re.sub(r'\d+', '', text)
    # Remove stopwords
    stop_words = set(stopwords.words('english'))
    words = text.split()
    words = [word for word in words if word not in stop_words]
    # Lemmatize words
    lemmatizer = WordNetLemmatizer()
    words = [lemmatizer.lemmatize(word) for word in words]
    return ' '.join(words)

df['cleaned_text'] = df['Text'].apply(preprocess_text)
```

```
# Cleaned Text
df
```

	Text	Label	cleaned_text
0	The impact of educational reforms remains unce...	positive	impact educational reform remains uncertain de...
1	Critics argue that recent improvements in the ...	negative	critic argue recent improvement school system ...
2	Innovative teaching methods have led to unexpe...	positive	innovative teaching method led unexpected chal...
3	Despite budget constraints, the school has man...	positive	despite budget constraint school managed maint...
4	The true effectiveness of online learning plat...	negative	true effectiveness online learning platform st...
...	...	...	...
51	Sports fandom can foster a sense of community ...	positive	sport fandom foster sense community belonging ...
52	Sports events offer a platform for showcasing ...	positive	sport event offer platform showcasing cultural...
53	The pressure to win in sports can overshadow t...	negative	pressure win sport overshadow enjoyment playin...
54	Sports programs in schools play a crucial role...	positive	sport program school play crucial role charact...
55	The commercialization of sports has led to exp...	negative	commercialization sport led exploitation commo...

209 rows × 3 columns

Next steps: Generate code with df | View recommended plots | New interactive sheet

```
# Function to apply Gaussian Naive Bayes
def apply_gaussian_nb(X_train, X_test, y_train, y_test):
    gnb = GaussianNB()
    gnb.fit(X_train.toarray(), y_train)
    y_pred = gnb.predict(X_test.toarray())
    accuracy = accuracy_score(y_test, y_pred)
    report = classification_report(y_test, y_pred)
    return accuracy, report
```

```
# TF-IDF Vectorizer
tfidf = TfidfVectorizer(max_features=5000)
X_tfidf = tfidf.fit_transform(df['cleaned_text'])
y = df['Label']
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=42)
```

```
# Multinomial Naive Bayes with TF-IDF
nb_tfidf = MultinomialNB()
nb_tfidf.fit(X_train_tfidf, y_train)
y_pred_tfidf = nb_tfidf.predict(X_test_tfidf)
accuracy_tfidf = accuracy_score(y_test, y_pred_tfidf)
report_tfidf = classification_report(y_test, y_pred_tfidf)
print(f'TF-IDF Vectorizer with MultinomialNB Accuracy: {accuracy_tfidf}')
print(f'TF-IDF Vectorizer with MultinomialNB Classification Report:\n{report_tfidf}')
```

```
TF-IDF Vectorizer with MultinomialNB Accuracy: 0.7619047619047619
TF-IDF Vectorizer with MultinomialNB Classification Report:
              precision    recall  f1-score   support

    negative       0.87      0.62      0.72        21
    positive       0.70      0.90      0.79        21

    accuracy                           0.76        42
   macro avg       0.79      0.76      0.76        42
weighted avg       0.79      0.76      0.76        42
```

```
# Bernoulli Naive Bayes with TF-IDF
nb_bernoulli_tfidf = BernoulliNB()
nb_bernoulli_tfidf.fit(X_train_tfidf, y_train)
y_pred_bernoulli_tfidf = nb_bernoulli_tfidf.predict(X_test_tfidf)
accuracy_bernoulli_tfidf = accuracy_score(y_test, y_pred_bernoulli_tfidf)
report_bernoulli_tfidf = classification_report(y_test, y_pred_bernoulli_tfidf)
print(f'TF-IDF Vectorizer with BernoulliNB Accuracy: {accuracy_bernoulli_tfidf}')
print(f'TF-IDF Vectorizer with BernoulliNB Classification Report:\n{report_bernoulli_tfidf}')
```

```
TF-IDF Vectorizer with BernoulliNB Accuracy: 0.7380952380952381
TF-IDF Vectorizer with BernoulliNB Classification Report:
              precision    recall  f1-score   support

    negative       0.81      0.62      0.70        21
    positive       0.69      0.86      0.77        21

    accuracy                           0.74        42
   macro avg       0.75      0.74      0.73        42
weighted avg       0.75      0.74      0.73        42
```

```
# Gaussian Naive Bayes with TF-IDF
accuracy_gaussian_tfidf, report_gaussian_tfidf = apply_gaussian_nb(X_train_tfidf, X_test_tfidf, y_train, y_test)
print(f'TF-IDF Vectorizer with GaussianNB Accuracy: {accuracy_gaussian_tfidf}')
print(f'TF-IDF Vectorizer with GaussianNB Classification Report:\n{report_gaussian_tfidf}')
```

```
TF-IDF Vectorizer with GaussianNB Accuracy: 0.7380952380952381
TF-IDF Vectorizer with GaussianNB Classification Report:
              precision    recall  f1-score   support

    negative       0.73      0.76      0.74        21
    positive       0.75      0.71      0.73        21

    accuracy                           0.74        42
   macro avg       0.74      0.74      0.74        42
weighted avg       0.74      0.74      0.74        42
```

```
# Count Vectorizer
count_vectorizer = CountVectorizer(max_features=5000)
X_count = count_vectorizer.fit_transform(df['cleaned_text'])
X_train_count, X_test_count, y_train, y_test = train_test_split(X_count, y, test_size=0.2, random_state=42)
```

```
# Multinomial Naive Bayes with Count Vectorizer
nb_count = MultinomialNB()
nb_count.fit(X_train_count, y_train)
y_pred_count = nb_count.predict(X_test_count)
accuracy_count = accuracy_score(y_test, y_pred_count)
report_count = classification_report(y_test, y_pred_count)
print(f'Count Vectorizer with MultinomialNB Accuracy: {accuracy_count}')
print(f'Count Vectorizer with MultinomialNB Classification Report:\n{report_count}')
```

```
Count Vectorizer with MultinomialNB Accuracy: 0.6904761904761905
Count Vectorizer with MultinomialNB Classification Report:
              precision    recall  f1-score   support

    negative       0.70      0.67      0.68        21
    positive       0.68      0.71      0.70        21

    accuracy                           0.69        42
   macro avg       0.69      0.69      0.69        42
weighted avg       0.69      0.69      0.69        42
```

```
# Bernoulli Naive Bayes with Count Vectorizer
nb_bernoulli_count = BernoulliNB()
nb_bernoulli_count.fit(X_train_count, y_train)
y_pred_bernoulli_count = nb_bernoulli_count.predict(X_test_count)
accuracy_bernoulli_count = accuracy_score(y_test, y_pred_bernoulli_count)
report_bernoulli_count = classification_report(y_test, y_pred_bernoulli_count)
print(f'Count Vectorizer with BernoulliNB Accuracy: {accuracy_bernoulli_count}')
print(f'Count Vectorizer with BernoulliNB Classification Report:\n{report_bernoulli_count}')
```

```
Count Vectorizer with BernoulliNB Accuracy: 0.7380952380952381
Count Vectorizer with BernoulliNB Classification Report:
              precision    recall  f1-score   support

    negative       0.81      0.62      0.70        21
    positive       0.69      0.86      0.77        21

    accuracy                           0.74        42
```

```
   macro avg       0.75      0.74      0.73        42
weighted avg       0.75      0.74      0.73        42
```

```
# Gaussian Naive Bayes with Count Vectorizer
accuracy_gaussian_count, report_gaussian_count = apply_gaussian_nb(X_train_count, X_test_count, y_train, y_test)
print(f'Count Vectorizer with GaussianNB Accuracy: {accuracy_gaussian_count}')
print(f'Count Vectorizer with GaussianNB Classification Report:\n{report_gaussian_count}')
```

```
Count Vectorizer with GaussianNB Accuracy: 0.7142857142857143
Count Vectorizer with GaussianNB Classification Report:
              precision    recall  f1-score   support

    negative       0.70      0.76      0.73        21
    positive       0.74      0.67      0.70        21

    accuracy                           0.71        42
   macro avg       0.72      0.71      0.71        42
weighted avg       0.72      0.71      0.71        42
```

```
# Binary Vectorizer
binary_vectorizer = CountVectorizer(binary=True, max_features=5000)
X_binary = binary_vectorizer.fit_transform(df['cleaned_text'])
X_train_binary, X_test_binary, y_train, y_test = train_test_split(X_binary, y, test_size=0.2, random_state=42)
```

```
# Multinomial Naive Bayes with Binary Vectorizer
nb_binary = MultinomialNB()
nb_binary.fit(X_train_binary, y_train)
y_pred_binary = nb_binary.predict(X_test_binary)
accuracy_binary = accuracy_score(y_test, y_pred_binary)
report_binary = classification_report(y_test, y_pred_binary)
print(f'Binary Vectorizer with MultinomialNB Accuracy: {accuracy_binary}')
print(f'Binary Vectorizer with MultinomialNB Classification Report:\n{report_binary}')
```

```
Binary Vectorizer with MultinomialNB Accuracy: 0.6904761904761905
Binary Vectorizer with MultinomialNB Classification Report:
              precision    recall  f1-score   support

    negative       0.70      0.67      0.68        21
    positive       0.68      0.71      0.70        21

    accuracy                           0.69        42
   macro avg       0.69      0.69      0.69        42
weighted avg       0.69      0.69      0.69        42
```

```
# Bernoulli Naive Bayes with Binary Vectorizer
nb_bernoulli_binary = BernoulliNB()
nb_bernoulli_binary.fit(X_train_binary, y_train)
y_pred_bernoulli_binary = nb_bernoulli_binary.predict(X_test_binary)
accuracy_bernoulli_binary = accuracy_score(y_test, y_pred_bernoulli_binary)
report_bernoulli_binary = classification_report(y_test, y_pred_bernoulli_binary)
print(f'Binary Vectorizer with BernoulliNB Accuracy: {accuracy_bernoulli_binary}')
print(f'Binary Vectorizer with BernoulliNB Classification Report:\n{report_bernoulli_binary}')
```

```
Binary Vectorizer with BernoulliNB Accuracy: 0.7380952380952381
Binary Vectorizer with BernoulliNB Classification Report:
              precision    recall  f1-score   support

    negative       0.81      0.62      0.70        21
    positive       0.69      0.86      0.77        21

    accuracy                           0.74        42
   macro avg       0.75      0.74      0.73        42
weighted avg       0.75      0.74      0.73        42
```

```
# Gaussian Naive Bayes with Binary Vectorizer
accuracy_gaussian_binary, report_gaussian_binary = apply_gaussian_nb(X_train_binary, X_test_binary, y_train, y_test)
print(f'Binary Vectorizer with GaussianNB Accuracy: {accuracy_gaussian_binary}')
print(f'Binary Vectorizer with GaussianNB Classification Report:\n{report_gaussian_binary}')
```

```
Binary Vectorizer with GaussianNB Accuracy: 0.7142857142857143
Binary Vectorizer with GaussianNB Classification Report:
              precision    recall  f1-score   support

    negative       0.70      0.76      0.73        21
    positive       0.74      0.67      0.70        21

    accuracy                           0.71        42
   macro avg       0.72      0.71      0.71        42
weighted avg       0.72      0.71      0.71        42
```

```
# Save the Multinomial Naive Bayes model with TF-IDF
dump(nb_tfidf, 'nb_tfidf_model.joblib')
```

```
['nb_tfidf_model.joblib']
```

```
# Load the saved model
nb_tfidf = load('nb_tfidf_model.joblib')
```

```
# Predict a new sentence
def predict_sentence(model, vectorizer, sentence):
    sentence_vectorized = vectorizer.transform([sentence])
    prediction = model.predict(sentence_vectorized)
    return prediction[0]
```

```
# Get input from the user
user_input = input("Enter a sentence to predict its class: ")

# Predict and display the result
prediction = predict_sentence(nb_tfidf, tfidf, user_input)
print(f"The predicted class for the sentence '{user_input}' is: {prediction}")

# Display the model's accuracy
print(f'The model accuracy is: {accuracy_tfidf}')
```

```
Enter a sentence to predict its class: pressure in sports
The predicted class for the sentence 'pressure in sports' is: positive
The model accuracy is: 0.7619047619047619
```

Conclusion:

TF-IDF Vectorizer

- with MultinomialNB Accuracy: 0.7619047619047619
- with BernoulliNB Accuracy: 0.7380952380952381
- with GaussianNB Accuracy: 0.7380952380952381

Count Vectorizer

- with MultinomialNB Accuracy: 0.6904761904761905
- with BernoulliNB Accuracy: 0.7380952380952381
- with GaussianNB Accuracy: 0.7142857142857143

Binary Vectorizer

- with MultinomialNB Accuracy: 0.6904761904761905
- with BernoulliNB Accuracy: 0.7380952380952381
- with GaussianNB Accuracy: 0.7142857142857143

Highest Accuracy:

TF-IDF Vectorizer with MultinomialNB Accuracy: 0.7619047619047619