

Name : K.Pardhasaradhi  
 Ht\_no : 2403a52001  
 Batch \_no : 01

Double-click (or enter) to edit

```
df = pd.read_csv('/content/news.csv', engine='python', on_bad_lines='skip')
display(df.head())
```

	Unnamed: 0	title	text	label	
0	8476	You Can Smell Hillary's Fear	Daniel Greenfield, a Shillman Journalism Fello...	FAKE	
1	10294	Watch The Exact Moment Paul Ryan Committed Pol...	Google Pinterest Digg Linkedin Reddit Stumbleu...	FAKE	
2	3608	Kerry to go to Paris in gesture of sympathy	U.S. Secretary of State John F. Kerry said Mon...	REAL	
3	10142	Bernie supporters on Twitter erupt in anger ag...	— Kaydee King (@KaydeeKing) November 9, 2016 T...	FAKE	
4	875	The Battle of New York: Why This Primary Matters	It's primary day in New York and front-runners...	REAL	

```
# Make predictions on the test set
y_pred = mnbc_model.predict(X_test)

# Evaluate the model
print("Model Evaluation Metrics:")
print(f"Accuracy: {accuracy_score(y_test, y_pred):.4f}")
print(f"Precision: {precision_score(y_test, y_pred, pos_label='FAKE'):.4f}")
print(f"Recall: {recall_score(y_test, y_pred, pos_label='FAKE'):.4f}")
print(f"F1 score: {f1_score(y_test, y_pred, pos_label='FAKE'):.4f}")
```

```
print(f"F1-score: {f1_score(y_test, y_pred, pos_label= FAKE ):.4f} ")

print("\nConfusion Matrix:")
display(pd.DataFrame(confusion_matrix(y_test, y_pred), index=['True Real',

print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

Model Evaluation Metrics:

Accuracy: 0.8483  
Precision: 0.8966  
Recall: 0.7723  
F1-score: 0.8298

Confusion Matrix:

	Pred Real	Pred Fake
True Real	78	23
True Fake	9	101



Classification Report:

	precision	recall	f1-score	support
FAKE	0.90	0.77	0.83	101
REAL	0.81	0.92	0.86	110
accuracy			0.85	211
macro avg	0.86	0.85	0.85	211
weighted avg	0.85	0.85	0.85	211

```
# Initialize the Multinomial Naive Bayes model
mnf_model = MultinomialNB()

# Train the model on the training data
mnf_model.fit(X_train, y_train)

# Display model parameters
```

```
print("Multinomial Naive Bayes Model Parameters:")  
display(mnb_model.get_params())
```

```
Multinomial Naive Bayes Model Parameters:  
{'alpha': 1.0, 'class_prior': None, 'fit_prior': True, 'force_alpha': True}
```

```
# Split data into training and testing sets  
X_train, X_test, y_train, y_test = train_test_split(X_vectorized, y, test_size=0.2, random_state=42)
```

```
# Print the shapes of the datasets  
print(f"Shape of X_train: {X_train.shape}")  
print(f"Shape of X_test: {X_test.shape}")  
print(f"Shape of y_train: {y_train.shape}")  
print(f"Shape of y_test: {y_test.shape}")
```

```
Shape of X_train: (843, 5000)  
Shape of X_test: (211, 5000)  
Shape of y_train: (843,)  
Shape of y_test: (211,)
```

```
# Initialize TF-IDF Vectorizer  
tfidf_vectorizer = TfidfVectorizer(max_features=5000) # Limiting to 5000 features for demonstration
```

```
# Fit and transform the processed text  
X_vectorized = tfidf_vectorizer.fit_transform(X_processed)
```

```
# Print feature matrix shape  
print(f"Feature matrix shape: {X_vectorized.shape}")
```

```
# Display sample feature names  
print("\nSample feature names:")  
display(tfidf_vectorizer.get_feature_names_out()[:20]) # Display first 20 feature names
```

```
Feature matrix shape: (1054, 5000)
```

```
Sample feature names:
```

```
array(['abandon', 'abandoned', 'abandoning', 'abbott', 'abc',  
      'abdulazeez', 'abedin', 'ability', 'able', 'aborigine', 'abortion',  
      'abroad', 'absence', 'absent', 'absolute', 'absolutely', 'absurd',  
      'abuse', 'academic', 'academy'], dtype=object)
```

```
# Initialize lemmatizer and stopwords
```

```
lemmatizer = WordNetLemmatizer()
```

```
stop_words = set(stopwords.words('english'))
```

```
def preprocess_text(text):
```

```
    # Convert to lowercase
```

```
    text = text.lower()
```

```
    # Remove punctuation and numbers
```

```
    text = re.sub(r'^a-z]', ' ', text)
```

```
    # Tokenize words
```

```
    words = text.split()
```

```
    # Remove stopwords and lemmatize
```

```
    words = [lemmatizer.lemmatize(word) for word in words if word not in stop_words]
```

```
    return ' '.join(words)
```

```
# Apply preprocessing to the text column
```

```
X_processed = X.apply(preprocess_text)
```

```
print("Original text sample:")
```

```
display(X.head())
```

```
print("\nProcessed text sample:")
```

```
display(X_processed.head())
```

Original text sample:

	text
0	Daniel Greenfield, a Shillman Journalism Fello...
1	Google Pinterest Digg Linkedin Reddit Stumbleu...
2	U.S. Secretary of State John F. Kerry said Mon...
3	— Kaydee King (@KaydeeKing) November 9, 2016 T...
4	It's primary day in New York and front-runners...

**dtype:** object

Processed text sample:

	text
0	daniel greenfield shillman journalism fellow f...
1	google pinterest digg linkedin reddit stumbleu...
2	u secretary state john f kerry said monday sto...
3	kaydee king kaydeeking november lesson tonight...
4	primary day new york front runner hillary clin...

**dtype:** object

```
# Identify input text (X) and target labels (y)
X = df['text']
y = df['label']

print(f"Dataset size (rows, columns): {df.shape}")
print("\nClass distribution of target labels:")
display(y.value_counts())
```

Dataset size (rows, columns): (1054, 4)

Class distribution of target labels:

	count
label	
FAKE	529
REAL	525

**dtype:** int64

```
# Data loading and handling
import pandas as pd
import numpy as np

# Text preprocessing
import nltk
import re
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer

# Feature extraction (vectorization)
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer

# Model building
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

# Model evaluation
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, classif

# Download NLTK data if not already present
```

```
nltk.download('stopwords')  
nltk.download('wordnet')  
nltk.download('omw-1.4')
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...  
[nltk_data]   Unzipping corpora/stopwords.zip.  
[nltk_data] Downloading package wordnet to /root/nltk_data...  
[nltk_data] Downloading package omw-1.4 to /root/nltk_data...  
True
```