```
import pandas as pd
import numpy as np
import re
import string

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, confusion_matrix
```

```
# Load dataset
df = pd.read_csv("/content/news.csv", encoding='latin-1')

# Keep only necessary columns
# The 'news.csv' dataset has columns like 'title', 'text', 'label'.
# Assuming 'label' is the existing label column and 'text' is the message content.
df = df[['label','text']] # Select 'label' as the first column, 'text' as the second.
df.columns = ['label','message'] # Rename them to 'label' and 'message' respectively.

# Display first 5 rows
print(df.head())

# Dataset size
print("Dataset shape:", df.shape)

# Class distribution
print(df['label'].value_counts())
```

```
  label                                            message
0  FAKE  Daniel Greenfield, a Shillman Journalism Fello...
1  FAKE  Google Pinterest Digg Linkedin Reddit Stumbleu...
2  REAL  U.S. Secretary of State John F. Kerry said Mon...
3  FAKE  â�� Kaydee King (@KaydeeKing) November 9, 2016...
4  REAL  It's primary day in New York and front-runners...
Dataset shape: (6335, 2)
label
REAL    3171
FAKE    3164
Name: count, dtype: int64
```

```
# Convert to lowercase
df['message'] = df['message'].str.lower()

# Remove punctuation & numbers
def clean_text(text):
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    return text

df['message'] = df['message'].apply(clean_text)
```

```
vectorizer = TfidfVectorizer(
    stop_words='english',
    ngram_range=(1,2),     # Unigram + Bigram (IMPORTANT)
    max_df=0.9,
    min_df=2
)

X = vectorizer.fit_transform(df['message'])
y = df['label']

print("Feature Matrix Shape:", X.shape)
```

```
Feature Matrix Shape: (6335, 320245)
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42,
    stratify=y
)
```

```
model = MultinomialNB(alpha=0.5)
model.fit(X_train, y_train)
```

```
▾ MultinomialNB
MultinomialNB(alpha=0.5)
```

```
y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.8808208366219415

Confusion Matrix:
 [[493 140]
 [ 11 623]]

Classification Report:
               precision    recall  f1-score   support

        FAKE       0.98      0.78      0.87       633
        REAL       0.82      0.98      0.89       634

    accuracy                           0.88      1267
   macro avg       0.90      0.88      0.88      1267
weighted avg       0.90      0.88      0.88      1267
```

Start coding or generate with AI.