Assignment 2: Text classification for Sentimental analysis using KNN. (Refer any dataset like Titanic, Twitter, etc.)

Code =>

```
import pandas as pd
import numpy as np
import re
import nltk
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report

# Load Dataset (with correct encoding and column names)
df = pd.read_csv(r"C:\Users\Shreyash Musmade\Desktop\Practical\MIDS_Prac-2\training.1600000.processed.noemoticon.csv",
                 encoding="ISO-8859-1",
                 names=["target", "id", "date", "flag", "user", "text"])  # Column names

# Keep only relevant columns
df = df[['text', 'target']]

# Convert sentiment labels (target: 4 → positive, 0 → negative)
df['sentiment'] = df['target'].apply(lambda x: 1 if x == 4 else 0)

# Reduce dataset size (Optional: Take a subset for testing)
df = df.sample(n=50000, random_state=42)  # Reduce to 50,000 samples

# Preprocessing
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))

def preprocess_text(text):
    text = str(text).lower()
    text = re.sub(r'\W', ' ', text)  # Remove special characters
    text = re.sub(r'\s+', ' ', text)  # Remove extra spaces
    return ' '.join([word for word in text.split() if word not in stop_words])

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

```python
# Convert text into numerical features (TF-IDF) - Keep as Sparse Matrix
vectorizer = TfidfVectorizer(max_features=5000)
X = vectorizer.fit_transform(df['cleaned_text'])  # No `.toarray()` here
y = df['sentiment']

# Split Dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Train KNN Model (Increase `n_neighbors` to handle sparse data)
knn = KNeighborsClassifier(n_neighbors=7, metric="cosine")
knn.fit(X_train, y_train)

# Predictions
y_pred = knn.predict(X_test)

# Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

Output=>

```
[Running] python -u "c:\Users\Shreyash Musmade\Desktop\Practical\MIDS_Prac-
2\Practical.py"

[nltk_data] Downloading package stopwords to C:\Users\Shreyash
[nltk_data]     Musmade\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Accuracy: 0.6775
Classification Report:
              precision    recall  f1-score   support

           0       0.69      0.64      0.66      4977
           1       0.67      0.72      0.69      5023

    accuracy                           0.68     10000
   macro avg       0.68      0.68      0.68     10000
weighted avg       0.68      0.68      0.68     10000
```