FAKE NEWS DETECTION USING NLP

Building a fake news detection project using Natural Language Processing (NLP) involves several steps, including loading and preprocessing the dataset, feature extraction, model training, and evaluation

Step 1: Install necessary libraries

→pip install pandas scikit-learn nltk

Step 2: Import libraries

```
import pandas as pd
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, confusion_matrix, classification_report
import nltk
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
```

Step 3: Load and explore the dataset

```
# Load the dataset
df = pd.read_csv('fake_news_dataset.csv')
# Explore the dataset
print(df.head())
```

Step 4: Preprocess the text data

```
# Remove NaN values
df = df.dropna()

# Combine title and text for analysis
df['total_text'] = df['title'] + ' ' + df['text']

# Remove stopwords and perform stemming
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
ps = PorterStemmer()

def preprocess_text(text):
    words = nltk.word_tokenize(text)
    words = [ps.stem(word) for word in words if word.isalpha() and word.lower()
not in stop_words]
    return ' '.join(words)

df['processed_text'] = df['total_text'].apply(preprocess_text)
```

Step 5: Split the dataset into training and testing sets

```
X_train, X_test, y_train, y_test = train_test_split(df['processed_text'], df['label'],
test_size=0.2, random_state=42)
```

Step 6: Feature extraction using TF-IDF

```
tfidf_vectorizer = TfidfVectorizer(max_features=5000)
X_train_tfidf = tfidf_vectorizer.fit_transform(X_train)
X_test_tfidf = tfidf_vectorizer.transform(X_test)
```

Step 7: Train a classifier (e.g., Naive Bayes)

```
classifier = MultinomialNB()
classifier.fit(X_train_tfidf, y_train)
```

Step 8: Make predictions and evaluate the model

```
predictions = classifier.predict(X_test_tfidf)

accuracy = accuracy_score(y_test, predictions)
conf_matrix = confusion_matrix(y_test, predictions)
classification_rep = classification_report(y_test, predictions)

print(f'Accuracy: {accuracy}')
print(f'Confusion Matrix:\n{conf_matrix}')
print(f'Classification Report:\n{classification_rep}')
```