NLP Project Final Report – Part B: News Article Classification

1. Importing Libraries and Handling Missing Values

We start by importing the essential libraries:

- **pandas** and **numpy**: For handling and manipulating data.
- **re**: Helps with cleaning text using regular expressions.
- **nltk**: A powerful toolkit for natural language processing tasks like tokenizing text and removing stopwords.
- **matplotlib** and **seaborn**: Useful for creating charts and visualizations.
- **sklearn**:
    - For converting text into numerical data (TF-IDF and BoW).
    - For splitting data, training models, and evaluating performance.
    - For implementing models like Logistic Regression, Naïve Bayes, and SVM.

To handle missing data, we use dropna() to eliminate rows with null values. Using inplace=True applies the changes directly to the dataframe.

---

2. Text Preprocessing

We clean and process the text using the following steps:

- **Cleaning**: The text is converted to lowercase, punctuation and numbers are removed, and common stopwords are filtered out.
- **Combining**: Headline and short description fields are merged into a single string for analysis.
- **Tokenizing**: The cleaned text is split into individual words for further processing.

This ensures our data is in a clean, consistent format that’s ready for model training.

---

3. Feature Extraction

Code:

```
# TF-IDF Feature Extraction
tfidf = TfidfVectorizer(max_features=5000)
X_tfidf = tfidf.fit_transform(df['processed_text'])
# Bag of Words (BoW)
bow = CountVectorizer(max_features=5000)
X_bow = bow.fit_transform(df['processed_text'])
y = df['category']
# Train-Test Split
X_train_tfidf, X_test_tfidf, y_train, y_test = train_test_split(X_tfidf, y, test_size=0.2, random_state=42)
X_train_bow, X_test_bow, _, _ = train_test_split(X_bow, y, test_size=0.2, random_state=42)
```

We use two common techniques to convert text into numerical features:

- **TF-IDF (Term Frequency–Inverse Document Frequency)**: Gives weight to words that are more unique and meaningful.
- **BoW (Bag of Words)**: Represents text based on the count of words.

The data is split into training and testing sets using an 80/20 ratio. This allows us to train our models on most of the data and test them on unseen samples for evaluation.

---

4. Model Training & Evaluation

Code:

```
def train_evaluate_model(model, X_train, X_test, y_train, y_test):
 scores = cross_val_score(model, X_train, y_train, cv=5, scoring='accuracy')
 print(f"Model: {model.__class__.__name__}")
 print(f"Cross-Validation Accuracy: {scores.mean():.4f} (+/- {scores.std():.4f})")

 model.fit(X_train, y_train)
 y_pred = model.predict(X_test)
```

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

# Plot Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=model.classes_,
yticklabels=model.classes_)
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title(f"Confusion Matrix - {model.__class__.__name__}")
plt.show()
```

We define a function that trains and evaluates any machine learning model by:

- Using **5-fold cross-validation** to assess the model's performance on different subsets of the training data.
- Training the model and predicting the test results.
- Calculating evaluation metrics such as accuracy, precision, recall, and F1-score.
- Displaying a **confusion matrix** as a heatmap to visually interpret model performance.

We apply this process to two models:

- **Logistic Regression**: A linear model that works well with text data. We increase max_iter to 200 for better convergence.
- **Multinomial Naïve Bayes**: A probabilistic model that performs well on text classification tasks, especially with BoW and TF-IDF features.

---

5. Observations

- **TF-IDF** performs slightly better than BoW in most scenarios.
- **Logistic Regression** outperforms Naïve Bayes when using TF-IDF features.
- With **BoW**, both models produce similar results.

VIDEO EXPLANATION: -

https://drive.google.com/file/d/1HKiMWnLjbZwE5BgvS8AsEJPmooZqY8Ka/view?usp=sharing