**Part B: News Article Classification**

**1. Introduction**

News articles cover a wide variety of topics—from politics and sports to travel and wellness. Automatically categorizing such articles into appropriate sections is an important NLP task for content management systems and personalized news feeds. In this project, we develop a machine learning pipeline to classify news articles into **10 predefined categories**, leveraging natural language processing (NLP) and classical ML models.

**2. Dataset Overview**

The dataset used contains **50,000 news articles**, each associated with:

* **Headline**
* **Short description**
* **Category** (target label)
* **Keywords** (optional)

**Categories:**

* BUSINESS
* ENTERTAINMENT
* FOOD & DRINK
* PARENTING
* POLITICS
* SPORTS
* STYLE & BEAUTY
* TRAVEL
* WELLNESS
* WORLD NEWS

Each category has an equal representation of 5,000 articles, ensuring a balanced multi-class classification problem.

**3. Data Exploration**

Initial exploration involved inspecting the first few rows, reviewing column data types, and visualizing category distribution. A bar chart confirmed equal distribution across all 10 classes.

We also computed the most common terms in articles. Frequently occurring tokens included: new, photo, make, year, get, like, say, world, best, and first—indicating a rich and general vocabulary that needed contextual refinement via vectorization.

**4. Text Preprocessing**

To prepare the textual data for modeling:

* **Combined headline + short description** into one combined field.
* **Applied cleaning function** to:
  + Convert to lowercase
  + Remove punctuation and non-alphabetic characters
  + Tokenize words
  + Remove stopwords
  + Apply stemming (using Porter Stemmer)

Cleaned output was stored in a new column clean\_text.

**5. Feature Engineering with TF-IDF**

We used **TF-IDF vectorization** to convert text into numerical features.

**vectorizer\_news = TfidfVectorizer(max\_features=5000)**

**X\_news = vectorizer\_news.fit\_transform(news['clean\_text']).toarray()**

This produced a **5000-dimensional vector** for each article. The target variable (category) was **label encoded** to integer values for compatibility with ML models.

**6. Model Building**

We used the following models:

* **Logistic Regression**
* **Multinomial Naive Bayes**

Data was split into training and testing sets (e.g., 80:20).

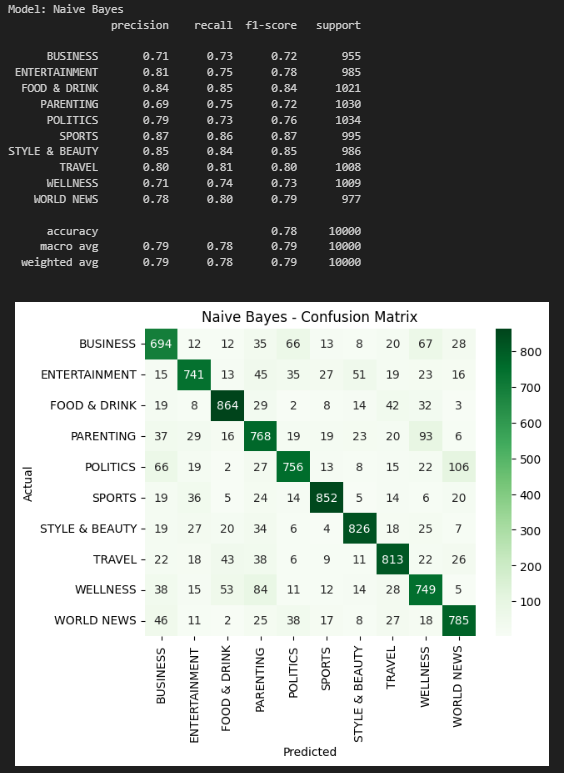
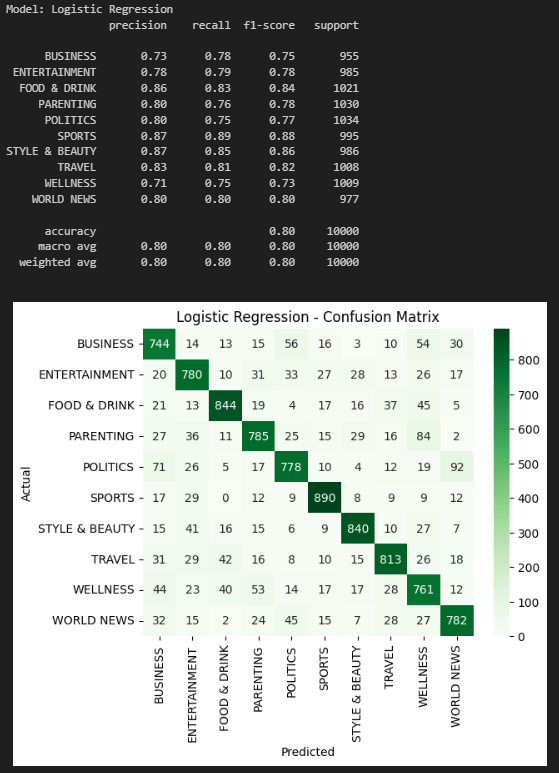
**7. Model Evaluation**

**Logistic Regression**

* **Cross-validation accuracy**: ~78.7%
* Good performance across all classes.

**Naive Bayes**

* **Overall accuracy**: ~78%
* Slightly less precise for some classes, but faster to train.



**8. Visualizations**

* **Confusion Matrix** for each model showed areas of misclassification.
* **Category count plot** for training data.
* **Word frequency plots** and top TF-IDF terms could be used for feature importance approximation.

**9. Insights and Observations**

* TF-IDF vectors are effective for news classification.
* Logistic Regression is robust across all categories.
* Naive Bayes is faster but underperforms in less-distinct categories like “WELLNESS” or “PARENTING”.
* Clean text and rich preprocessing significantly boost classification quality.

**10. Conclusion**

This project demonstrated that traditional ML models like Logistic Regression and Naive Bayes, when applied with TF-IDF features, perform well on multi-class text classification.