News Group Classification Report

|  |  |
| --- | --- |
| 2022170473 | نوران هيثم عثمان عثمان |
| 2022170432 | ملك خالد محمد بخيت محمود الحادي |
| 2022170393 | محمد وائل مروان البكري |
| 2022170402 | مروان إبراهيم إبراهيم اليماني |
| 2022170355 | محمد أكرم محمد كامل |
| 2022170311 | كريم حاتم أحمد محمد رضوان |

## 1. Overview

In this project, we aim to build a text classification model that can automatically categorize news articles into their respective topics. This involves applying Natural Language Processing (NLP) techniques and training a machine learning model on labeled news data.

## 2. Dataset Overview

Our dataset consists of news articles with the following columns:

* **category**: The target label indicating the topic of the article (e.g., sports, tech, politics).
* **filename**: The file name or path associated with each article.
* **content**: The full text of the news article, which will serve as our main input for training the classification model.

We will use the content column as the input feature for NLP processing, and the category column as the target for model training.

## 3. Preprocessing

To ensure the text data is clean and consistent for training, we apply the following preprocessing steps:

a) Expand Contractions

* Convert common contractions into their full forms to maintain consistency.
* Examples:
* "don't" → "do not"
* "it's" → "it is"

### b) Lowercase the Text

* Normalize all text to lowercase to reduce the vocabulary size and avoid case-sensitive duplicates.

### c) Remove Metadata

* Strip away unnecessary elements like headers, footers, or email signatures that do not contribute meaningful information.

### d) Remove Numbers and Punctuation

* These elements are often noise in classification tasks and do not contribute to meaning.

### e) Remove Extra Whitespace

* Clean up tabs, multiple spaces, and newline characters to ensure uniform formatting.

### f) Tokenize Text

* Break each cleaned text into individual words (tokens) using NLTK's word\_tokenize. This enables more granular analysis and further NLP processing.
* Example:
* “the quick brown fox” → [“the”, “quick”, “brown”, “fox”]

### g) Remove Stop words

* Eliminate common English stopwords ("the", "is", "and", e.g.) using NLTK’s predefined list. Thesewords typically carry less semantic meaning and can introduce noise in text classification tasks.
* Example:
* [“the”, “quick”, “brown”, “fox”] → [ “quick”, “brown”, “fox”]

### h) Lemmatization

* Reduce inflected or variant word forms to their **dictionary headword** form.
* Example:
  + Am,is,are -> be
  + Car,cars,car’s,cars’ -> car

### i) TF–IDF Vectorization

* To convert our cleaned and lemmatized text into features suitable for machine learning, we apply TF–IDF vectorization. TF–IDF not only counts how often a term appears in a document , but also down‐weights terms that appear in many documents, thereby emphasizing words that are more discriminative for classification.

### j) Train / Validation / Test Split

* To ensure that our model is trained and evaluated in an unbiased manner, we split the TF–IDF feature matrix and labels into three disjoint sets:
  + Test set (20% of data)
  + Validation set (20% of data)
  + Training set (60% of data)

## 4. Modelling

We evaluated several classification algorithms to identify the best fit for our dataset and

for each model we did the following steps:

1) Reason:

* Why we chose this specific model for our dataset.

2) Hyper Tuning:

* **Grid Search with k‑fold Cross‑Validation**: Generate all combinations of selected parameters values via k‑fold cross‑validation, then choosing the set that yields the highest average validation score.
* **Trial and error**:Manually try all combinations of selected parameter values and compare their accuracy scores.

3) Confusion Matrix:

* We visualized the confusion matrix of the model to gain more insight into the model’s strengths and weaknesses.

4) Model Metrics:

* Using the best hyperparameters, we trained the model with them and looked at its metrics to compare it with the other models.

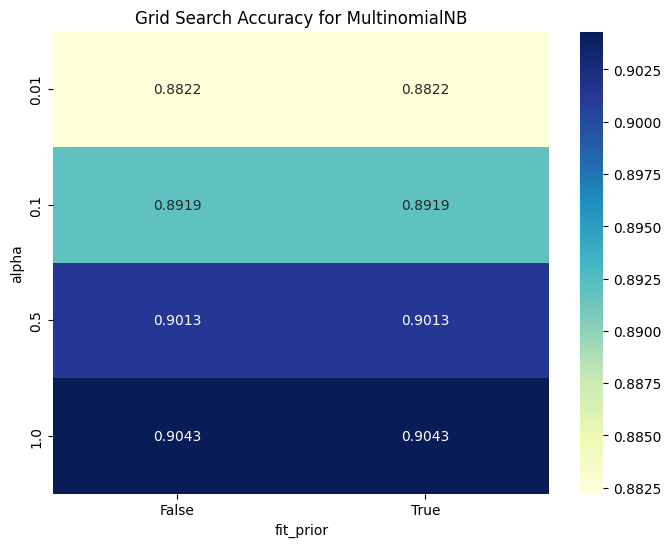
a) Multinomial Naïve Bayes

* **Reason:**

We chose multinomial because it naturally handles TF–IDF features and tends to work best with test classification.

* **Hyper tuning plots:**

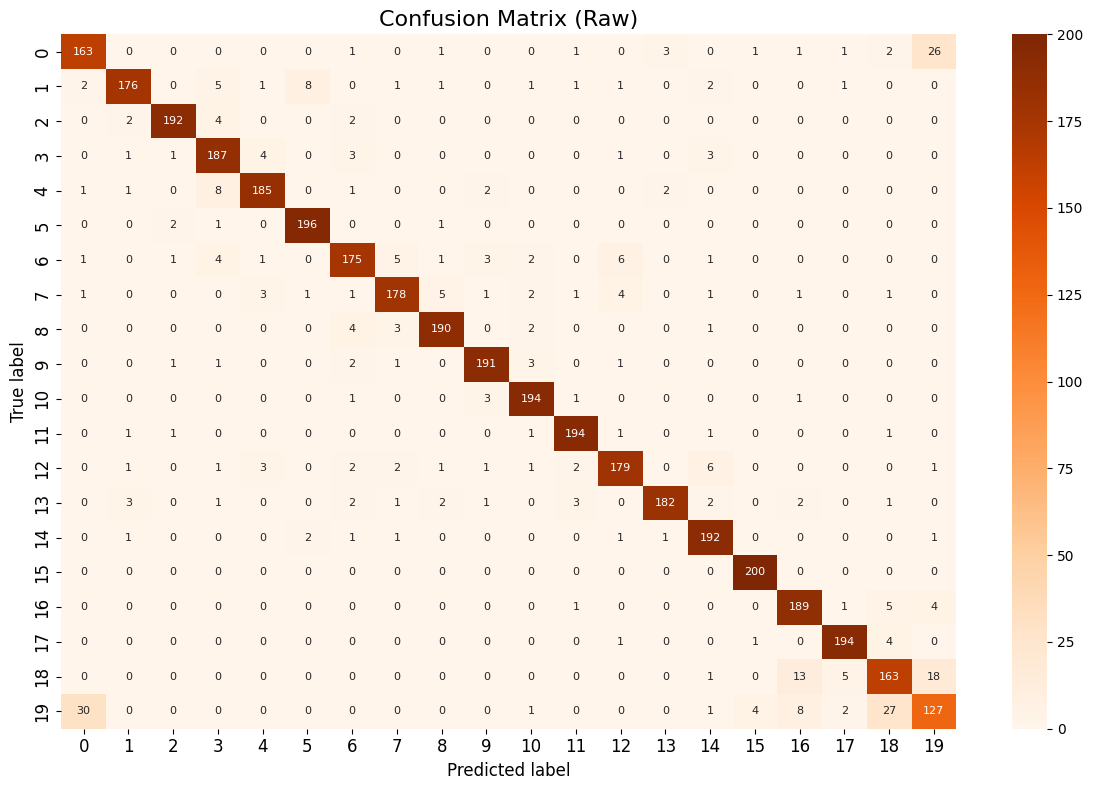
Looking at the plots we can see that the best hyperparameters are alpha = 1 and any value of fit\_prior



A graph of error

AI-generated content may be incorrect.

* **Confusion matrix plot:**



* **Model metrics results:**

A black background with white numbers

AI-generated content may be incorrect.

b) Random Forest Classifier

* **Reason:**

We chose Random Forest because it can handle high‑dimensional sparse TF–IDF features and capture non‑linear interactions between term patterns.

* **Hyper tuning plots:**

Looking at the plots we can see that the best hyperparameters are n\_estimators = 200 and max\_depth = 10.

A blue squares with white text

AI-generated content may be incorrect.

A screenshot of a graph

AI-generated content may be incorrect.

* **Confusion matrix plot:**

A graph of numbers and symbols

AI-generated content may be incorrect.

* **Model metrics results:**

