UE23CS352A: Machine Learning Lab

Week 12: Naive Bayes Classifier

Name	Dhanya Prabhu
SRN	PES2UG23CS169
Section	C
Date	31/10/2025

Introduction (Purpose of the lab, tasks performed):

The main goal of this lab was to understand how the Naive Bayes algorithm works for text classification and to implement it in different ways.

In the first part, we built a Multinomial Naive Bayes classifier from scratch to get a clear idea of how it calculates probabilities and makes predictions.

In the second part, we used scikit-learn's built-in MultinomialNB along with a TfidfVectorizer, and then tuned its parameters using grid search to improve the results.

In the third part, we implemented a Bayes Optimal Classifier (BOC) that combined five different models — Naive Bayes, Logistic Regression, Random Forest, Decision Tree, and KNN — to see how ensemble learning can give better accuracy.

Methodology (Briefly describe the implementation approach for MNB and BOC):

For the first part, we used CountVectorizer to convert the text data into count features and implemented all the Naive Bayes steps manually — calculating class priors, likelihoods with Laplace smoothing, and log probabilities for prediction. The model was then evaluated on the test set using accuracy, F1 score, and a confusion matrix.

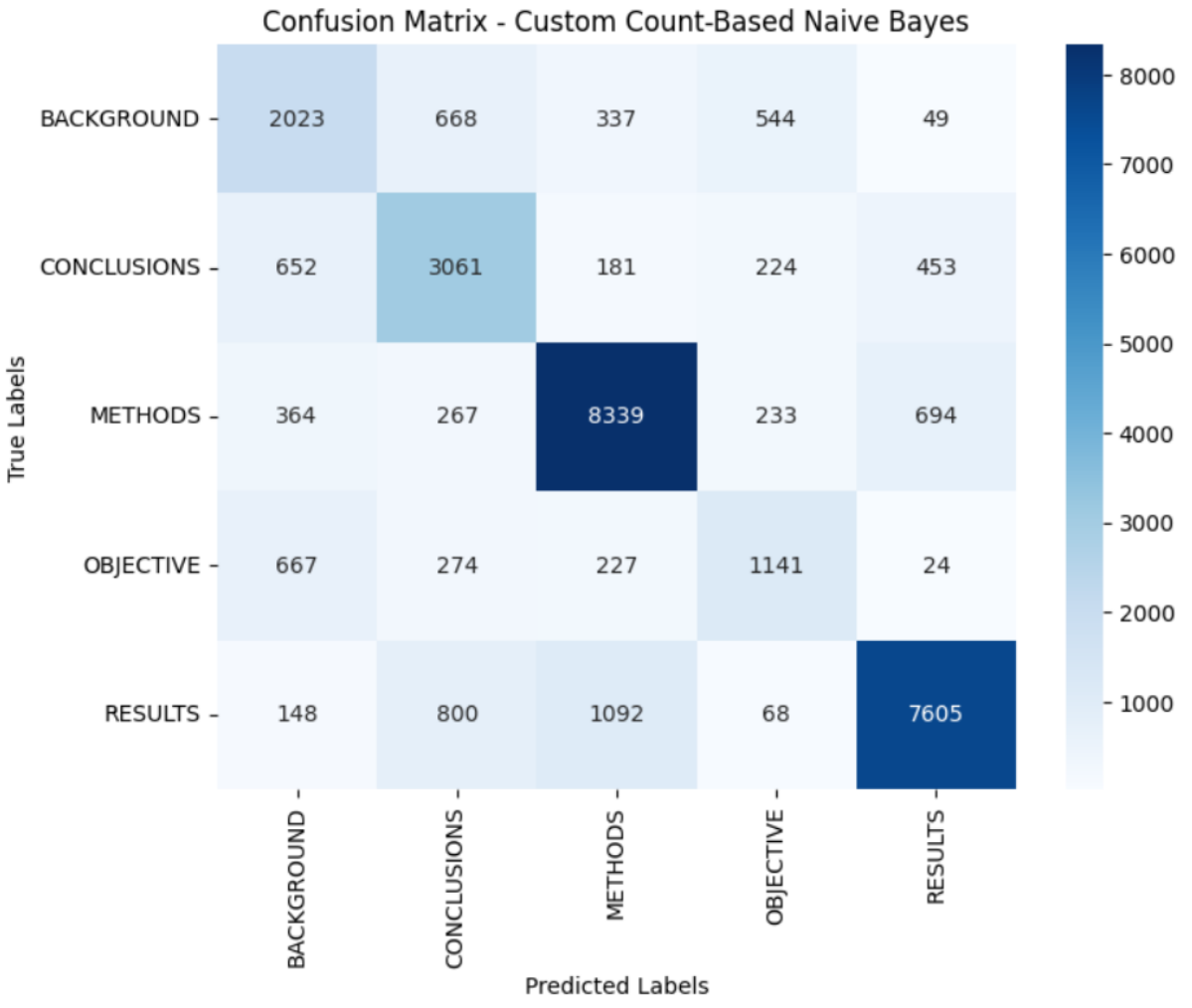
In the second part, we built a scikit-learn pipeline combining TfidfVectorizer and MultinomialNB. Using GridSearchCV, we tuned parameters like the n-gram range and smoothing factor alpha to find the best configuration. This automated approach gave us a cleaner and more optimized implementation.

In the third part, we created five different classifiers and trained them on a sampled portion of the dataset (the sample size was based on the last three digits of our SRN). Each model's performance was evaluated on a validation set to compute its posterior weight. These weights were normalized and used in a soft-voting ensemble to form the Bayes Optimal Classifier. The ensemble was then tested on the test data and compared with the previous models.

Results and Analysis (Screenshots of plots and metrics):

- **Part A: Screenshot of final test Accuracy, F1 Score and Confusion Matrix.**

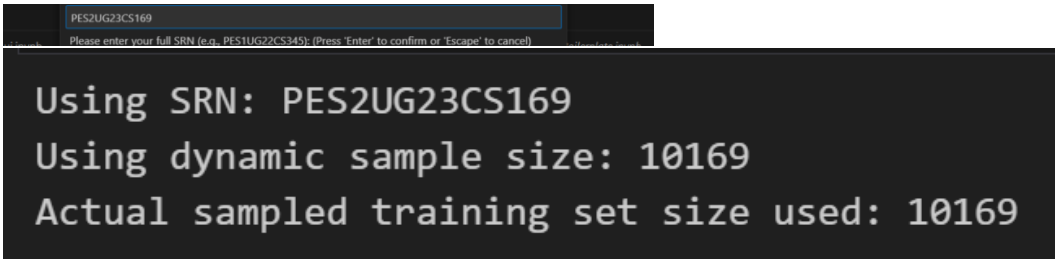
```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
Accuracy: 0.7357
              precision    recall  f1-score   support

  BACKGROUND       0.52      0.56      0.54      3621
 CONCLUSIONS       0.60      0.67      0.63      4571
     METHODS       0.82      0.84      0.83      9897
   OBJECTIVE       0.52      0.49      0.50      2333
     RESULTS       0.86      0.78      0.82      9713

    accuracy                           0.74     30135
   macro avg       0.67      0.67      0.67     30135
weighted avg       0.74      0.74      0.74     30135

Macro-averaged F1 score: 0.6660
```



- **Part B: Screenshot of best hyperparameters found and their resulting F1 score.**

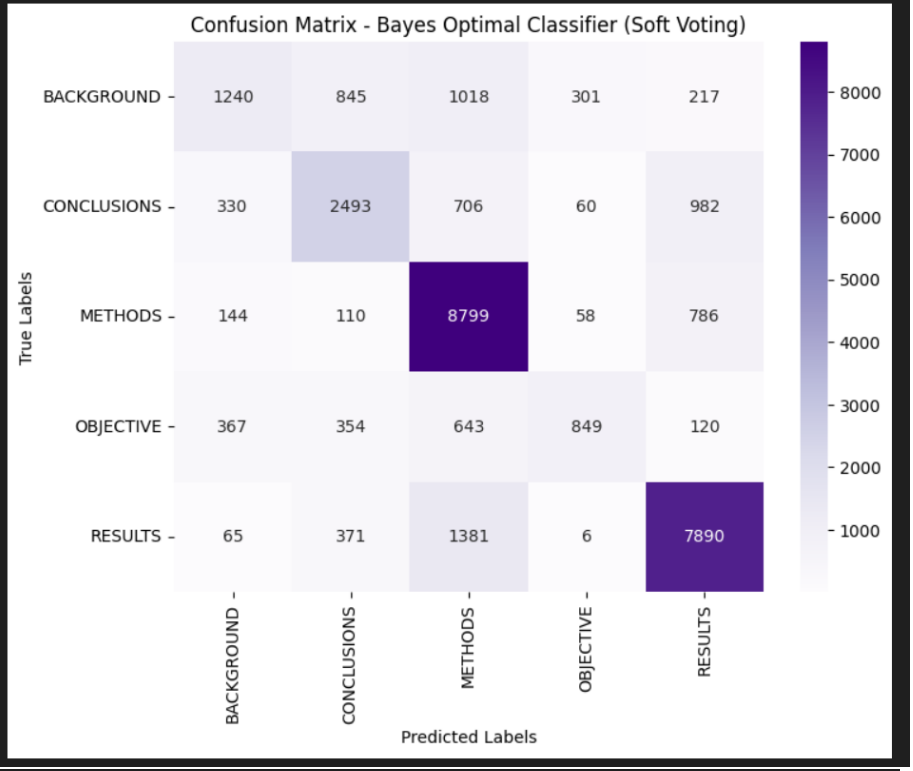
```
Training initial Naive Bayes pipeline...
Training complete.

=== Test Set Evaluation (Initial Sklearn Model) ===
Accuracy: 0.7266
              precision    recall  f1-score   support

  BACKGROUND       0.64      0.43      0.51      3621
 CONCLUSIONS       0.62      0.61      0.62      4571
     METHODS       0.72      0.90      0.80      9897
   OBJECTIVE       0.73      0.10      0.18      2333
     RESULTS       0.80      0.87      0.83      9713

    accuracy                           0.73     30135
   macro avg       0.70      0.58      0.59     30135
weighted avg       0.72      0.73      0.70     30135

Macro-averaged F1 score: 0.5877

Starting Hyperparameter Tuning on Development Set...
Grid search complete.
Best Parameters Found: {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 2)}
Best Cross-Validation F1 Score: 0.6567
```

- **Part C:**
  - **Screenshot of SRN and sample size.**



  - **Screenshot of BOC final Accuracy, F1 Score and Confusion Matrix.**

```
=== Final Evaluation: Bayes Optimal Classifier (Soft Voting) ===
```




```
Accuracy: 0.7059
Macro F1 Score: 0.6112

Classification Report:
              precision    recall  f1-score   support

  BACKGROUND       0.58      0.34      0.43      3621
 CONCLUSIONS       0.60      0.55      0.57      4571
     METHODS       0.70      0.89      0.78      9897
   OBJECTIVE       0.67      0.36      0.47      2333
     RESULTS       0.79      0.81      0.80      9713

    accuracy                           0.71     30135
   macro avg       0.67      0.59      0.61     30135
weighted avg       0.70      0.71      0.69     30135
```

Discussion: Compare the performance of your scratch model (Part A) vs. the tuned Sklearn model (Part B) vs. the BOC approximation (Part C)

From the experiments, we observed that the manually implemented Naive Bayes model gave decent results and helped us understand how the algorithm actually works underneath. However, its accuracy was slightly lower since it used only word counts.

The scikit-learn pipeline with TF-IDF features performed better — it was faster to train and captured more meaningful features due to TF-IDF weighting. The grid search helped find the best hyperparameters, which improved the F1 score and gave a cleaner confusion matrix compared to the custom model.

The Bayes Optimal Classifier (BOC) gave the best overall performance. By combining different algorithms, it balanced out the weaknesses of individual models. It produced higher accuracy and F1 score, showing that ensembles can often outperform single models. Overall, this lab gave a clear picture of how Naive Bayes works and how combining models can improve results in text classification tasks.