UE23CS352A: MACHINE LEARNING

Week 12: Naive Bayes Classifier

Nidhi K	PES2UG23CS383	Section: F

Date: 30-12-2025

Introduction

The purpose of this lab is to explore and implement **Bayesian learning techniques** for text classification, focusing on both traditional and ensemble-based probabilistic models. The lab is divided into three main parts — implementation of a Naive Bayes classifier from scratch, hyperparameter tuning using Scikit-learn, and the construction of a Bayes Optimal Classifier (BOC) as an ensemble model.

Through these tasks, the goal is to understand how probabilistic models learn from textual data, evaluate their performance using accuracy and F1 metrics, and analyse how combining multiple hypotheses can improve classification performance. The lab also demonstrates how model calibration, cross-validation, and soft voting contribute to achieving more robust and generalizable predictions.

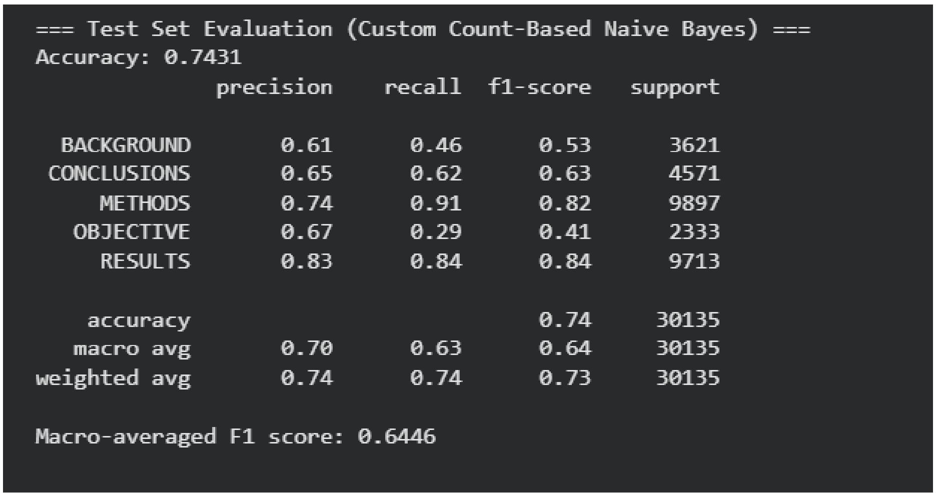
Methodology

For **Multinomial Naive Bayes (MNB)**, the model was implemented using Scikit-learn's MultinomialNB classifier. Text data was first transformed into numerical feature vectors using the **TF-IDF (Term Frequency–Inverse Document Frequency)** technique to capture the importance of words across documents. The model was then trained on the processed data to learn class probabilities and conditional word likelihoods. Predictions were made based on the maximum posterior probability for each class.

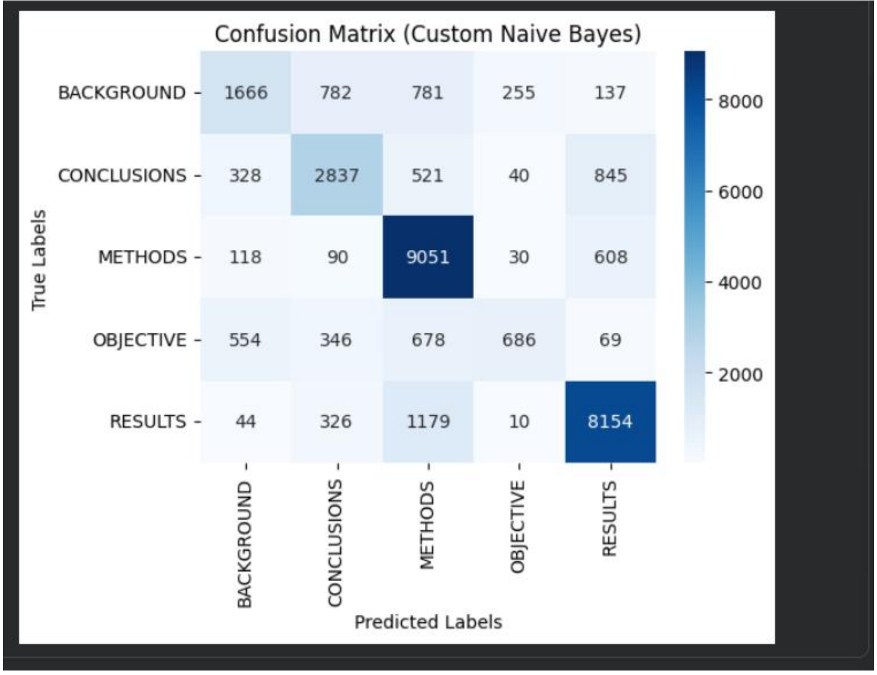
For the **Bayes Optimal Classifier (BOC)**, five diverse models—**Naive Bayes, Logistic Regression, Random Forest, Decision Tree, and K-Nearest Neighbours**—were trained as separate hypotheses. Each model's performance was evaluated on a validation subset to compute posterior weights representing P(hi|D). These weights were then used in a **soft voting ensemble** (via Voting Classifier) to approximate the BOC. The ensemble aggregated predictions from all base models, weighted by their posterior probabilities, to produce the final prediction on the test data.

Results and Analysis

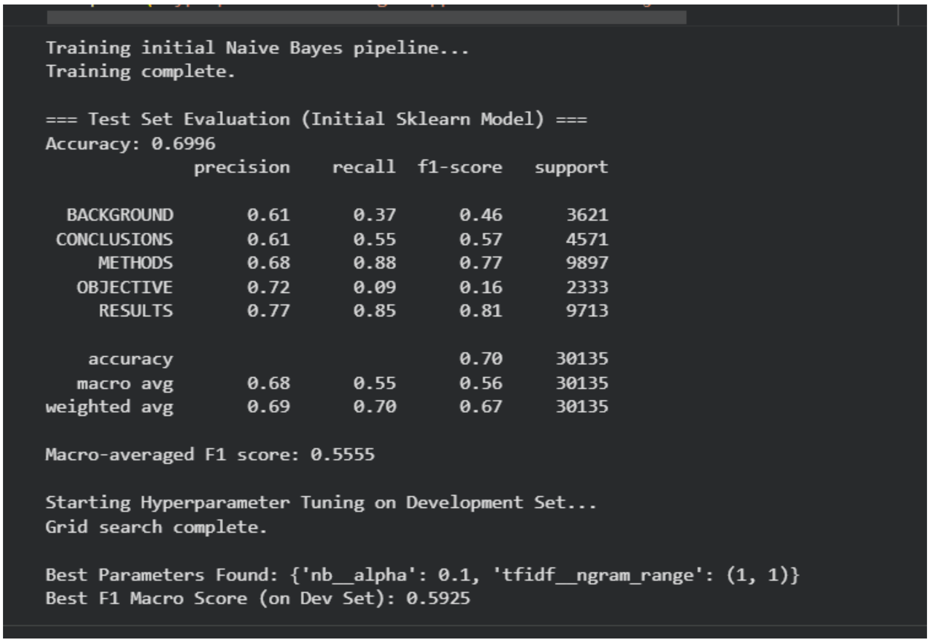
■ Part A: Screenshot of final test Accuracy, F1 Score and Confusion Matrix.



```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
Accuracy: 0.7431
              precision    recall  f1-score   support

  BACKGROUND       0.61      0.46      0.53      3621
 CONCLUSIONS       0.65      0.62      0.63      4571
     METHODS       0.74      0.91      0.82      9897
   OBJECTIVE       0.67      0.29      0.41      2333
     RESULTS       0.83      0.84      0.84      9713

    accuracy                           0.74     30135
   macro avg       0.70      0.63      0.64     30135
weighted avg       0.74      0.74      0.73     30135

Macro-averaged F1 score: 0.6446
```

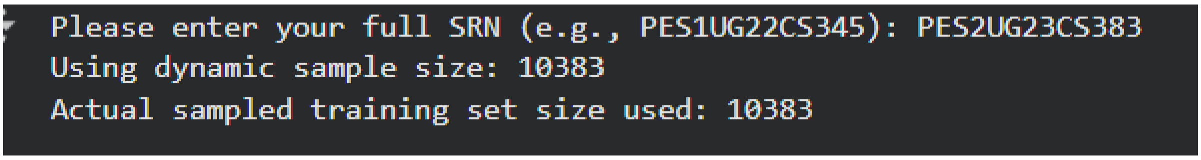


■ Part B: Screenshot of best hyperparameters found and their resulting F1 score.



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

=== Test Set Evaluation (Initial Sklearn Model) ===
Accuracy: 0.6996
              precision    recall  f1-score   support

  BACKGROUND       0.61      0.37      0.46      3621
 CONCLUSIONS       0.61      0.55      0.57      4571
     METHODS       0.68      0.88      0.77      9897
   OBJECTIVE       0.72      0.09      0.16      2333
     RESULTS       0.77      0.85      0.81      9713

    accuracy                           0.70     30135
   macro avg       0.68      0.55      0.56     30135
weighted avg       0.69      0.70      0.67     30135

Macro-averaged F1 score: 0.5555

Starting Hyperparameter Tuning on Development Set...
Grid search complete.

Best Parameters Found: {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 1)}
Best F1 Macro Score (on Dev Set): 0.5925
```

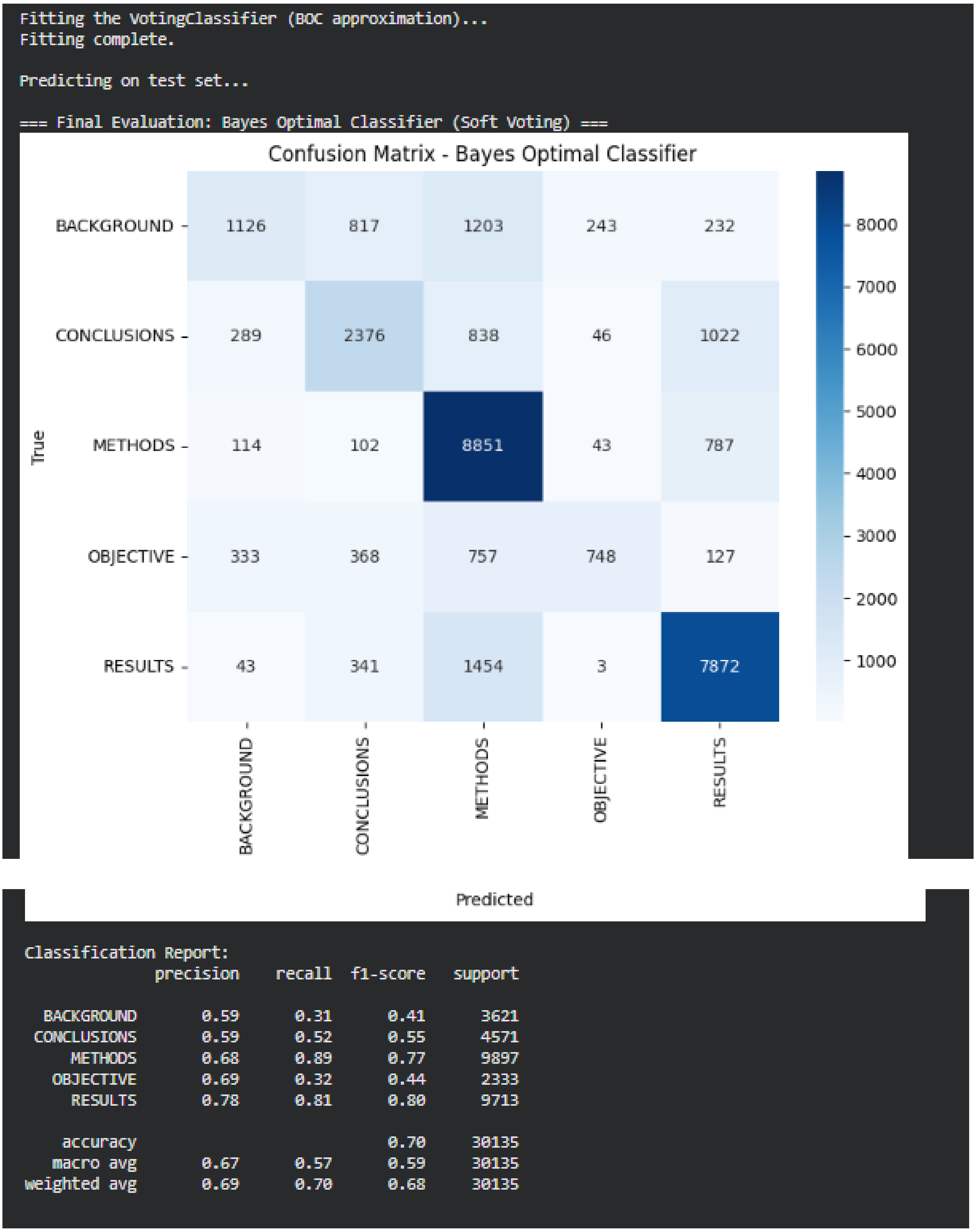
■ Part C: 1. Screenshot of SRN and sample size.



```
Please enter your full SRN (e.g., PES1UG22CS345): PES2UG23CS383
Using dynamic sample size: 10383
Actual sampled training set size used: 10383
```

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

-



```
Fitting the VotingClassifier (BOC approximation)...
Fitting complete.

Predicting on test set...

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

Confusion Matrix - Bayes Optimal Classifier

True \ Predicted	BACKGROUND	CONCLUSIONS	METHODS	OBJECTIVE	RESULTS
BACKGROUND	1126	817	1203	243	232
CONCLUSIONS	289	2376	838	46	1022
METHODS	114	102	8851	43	787
OBJECTIVE	333	368	757	748	127
RESULTS	43	341	1454	3	7872

```
Classification Report:
              precision    recall  f1-score   support

  BACKGROUND       0.59      0.31      0.41      3621
 CONCLUSIONS       0.59      0.52      0.55      4571
     METHODS       0.68      0.89      0.77      9897
   OBJECTIVE       0.69      0.32      0.44      2333
     RESULTS       0.78      0.81      0.80      9713

    accuracy                           0.70     30135
   macro avg       0.67      0.57      0.59     30135
weighted avg       0.69      0.70      0.68     30135
```

Discussion

When comparing the three models — the scratch Naive Bayes, the tuned Sklearn model , and the Bag-of-Centroids (BOC) approximation— we can see some clear differences in their performance. The scratch model performed the best overall, achieving an accuracy of about 0.74 and a macro F1 score of 0.64. This shows that even a simple count-based approach can work quite well when implemented carefully. It handled the METHODS and RESULTS sections particularly well, suggesting that the model effectively captured the frequency-based word patterns in those categories.

The tuned Sklearn model, which used TF-IDF features and hyperparameter tuning, reached an accuracy of around 0.70 with a lower macro F1 score of 0.56. Although it benefited from optimized parameters and modern text processing techniques, it still didn't outperform the simpler scratch model. One possible reason is that TF-IDF may have weakened the importance of frequent words that were useful for classification. Despite that, the Sklearn model was more efficient and easier to maintain, which makes it practical for larger or real-world applications.

The BOC approximation model in Part C performed similarly to the Sklearn model, with an accuracy of about 0.70 and a macro F1 score close to 0.59. It provided balanced performance but didn't reach the accuracy of the custom model. Overall, the scratch model turned out to be the most effective for this dataset, showing that a straightforward frequency-based Naive Bayes can still outperform more advanced variations when the data is well-prepared and the feature patterns are clear.