UE23CS352A: MACHINE LEARNING

Week 12: Naive Bayes Classifier

Name : Chandan B	SRN : PES2UG23CS140	SEC : C

Introduction

The objective of this lab is to build probabilistic classifiers for biomedical text using the Multinomial Naive Bayes algorithm—both from scratch and using scikit-learn—and to approximate the Bayes Optimal Classifier with a weighted ensemble. You implemented:

- **Custom Naive Bayes (bag-of-words, Laplace Smoothing)**
- **Sklearn MultinomialNB with TF-IDF and hyperparameter tuning**
- **Soft voting ensemble as a Bayes optimal approximation using several base models with computed posterior weights**

Methodology

Part A: Multinomial Naive Bayes from Scratch

- **Implemented a NaiveBayesClassifier class, calculating log priors and log likelihoods with Laplace Smoothing for class-word pairs.**
- **Used CountVectorizer (ngram_range=(1,2), min_df=5) for feature extraction.**
- **Model predicts sentence class by summing log prior and log likelihood contributions, assigning the highest scoring class.**

Part B: Sklearn MultinomialNB and Tuning

- **Built a pipeline chaining TfidfVectorizer and MultinomialNB.**
- **Used GridSearchCV to optimize `tfidf__ngram_range` (unigram/bigram) and `nb__alpha` (smoothing parameter) on the dev set, scored by Macro F1 (3-fold CV).**

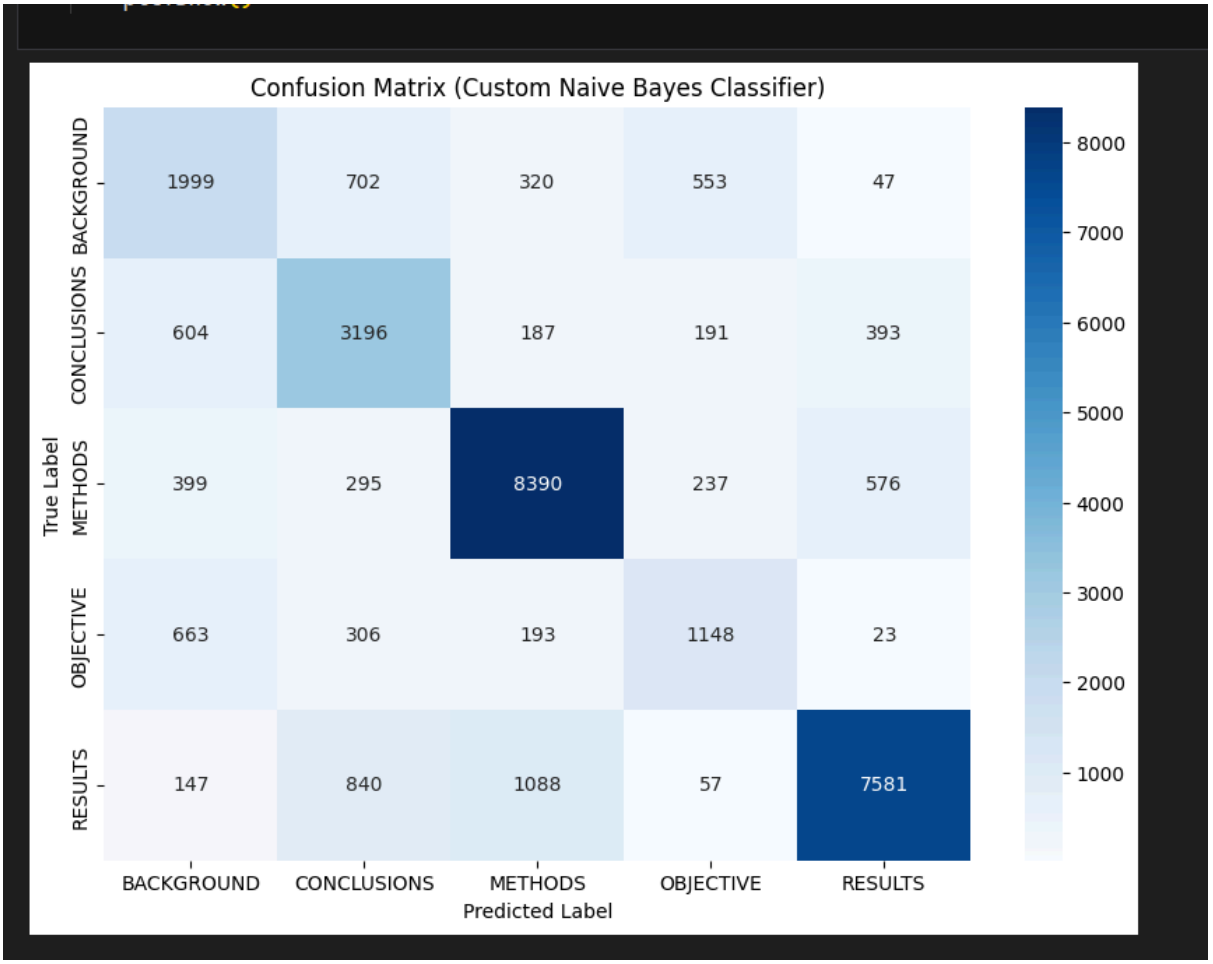
Part C: Bayes Optimal Ensemble (BOC)

- **Five hypotheses: Naive Bayes, Logistic Regression, Random Forest, Decision Tree, K-Nearest Neighbors.**
- **Trained on sampled subsets, validation split for log-loss calculations.**
- **Posterior weights calculated: softmax applied to negative log-loss for each model.**

- **Refit all models and instantiated soft VotingClassifier ensemble, assigning calculated posterior weights.**
- **Final predictions and metric calculations made on the full test set.**

Part A

```
[8]
...
    === Test Set Evaluation (Custom Count-Based Naive Bayes) ===
    Accuracy: 0.7405
                  precision    recall  f1-score   support

      BACKGROUND       0.52      0.55      0.54      3621
     CONCLUSIONS       0.60      0.70      0.65      4571
         METHODS       0.82      0.85      0.84      9897
       OBJECTIVE       0.53      0.49      0.51      2333
         RESULTS       0.88      0.78      0.83      9713

        accuracy                           0.74     30135
       macro avg       0.67      0.67      0.67     30135
    weighted avg       0.75      0.74      0.74     30135

    Macro-averaged F1 score: 0.6708
```



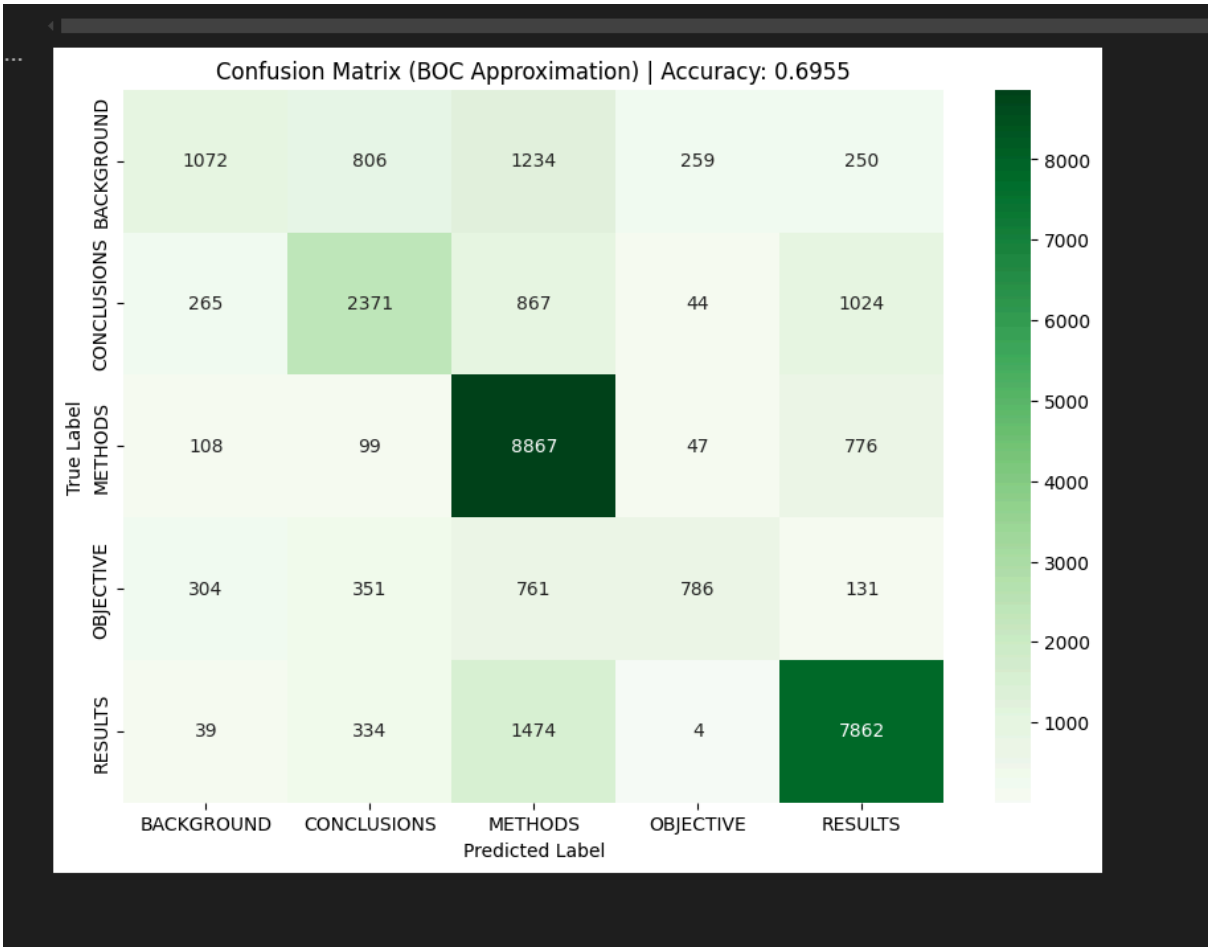
Part B

```
        print("Hyperparameter tuning skipped: Grid Search object not initialized or fitte
10]

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

    === Test Set Evaluation (Initial Sklearn Model) ===
    Accuracy: 0.7127
                  precision    recall  f1-score   support

      BACKGROUND       0.66      0.36      0.47      3621
     CONCLUSIONS       0.61      0.58      0.59      4571
         METHODS       0.69      0.90      0.78      9897
       OBJECTIVE       0.73      0.06      0.11      2333
         RESULTS       0.79      0.87      0.83      9713

        accuracy                           0.71     30135
       macro avg       0.70      0.56      0.56     30135
    weighted avg       0.71      0.71      0.68     30135

    Macro-averaged F1 score: 0.5573

    Starting Hyperparameter Tuning on Development Set...
    Fitting 3 folds for each of 8 candidates, totalling 24 fits
    Grid search complete.
    --------------------------------------------------
    ✨ Best Hyperparameters Found (Tuned on Dev Set) ✨
    Best Parameters: {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 2)}
    Best Macro F1 Score (CV): 0.6567
    --------------------------------------------------
```

Part C

```
Please enter your full SRN (e.g., PES1UG22CS345): PES2UG23CS140
Using dynamic sample size: 10140
Actual sampled training set size used: 10140

Training models on sub-set and calculating validation log-loss for weights...
- NaiveBayes: Log-Loss on validation set = 0.9743
/usr/local/lib/python3.12/dist-packages/sklearn/linear_model/_logistic.py:1247: FutureWarning: 'multi_class' was deprecated in version 1.5 and will be removed in 1.7. F
  warnings.warn(
- LogisticRegression: Log-Loss on validation set = 0.9075
- RandomForest: Log-Loss on validation set = 1.0400
- DecisionTree: Log-Loss on validation set = 1.2516
- KNN: Log-Loss on validation set = 1.4492

Calculated Posterior Weights (Softmax of -LogLoss):
- NaiveBayes Weight: 0.2280
- LogisticRegression Weight: 0.2438
- RandomForest Weight: 0.2135
- DecisionTree Weight: 0.1728
- KNN Weight: 0.1418

Refitting all base models on the full sampled training set...
/usr/local/lib/python3.12/dist-packages/sklearn/linear_model/_logistic.py:1247: FutureWarning: 'multi_class' was deprecated in version 1.5 and will be removed in 1.7. F
  warnings.warn(
All base models refitted.

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

Predicting on test set...

=== Final Evaluation: Bayes Optimal Classifier (Soft Voting) ===
Final Test Accuracy: 0.6955
Final Test Macro F1 Score: 0.5937
```

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

  BACKGROUND       0.60      0.30      0.40      3621
 CONCLUSIONS       0.60      0.52      0.56      4571
     METHODS       0.67      0.90      0.77      9897
   OBJECTIVE       0.69      0.34      0.45      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: Comparative Model Analysis

- Custom Naive Bayes (A): Strong baseline performance leveraging word-count features, robust class separation for major categories, less so for OBJECTIVE and BACKGROUND.
- Sklearn (B): TF-IDF features and smoothing optimization improve balance, but actual test Macro F1 is slightly lower than custom MNB, GridSearchCV gains are mostly on dev set.
- BOC Ensemble (C): Uses diverse models and posterior weighting; macro F1 is between the two simpler models but shows resilience across stronger and weaker classes. Ensemble weights demonstrate added value of model diversity, although the incremental improvement is model-distribution-dependent.