UE23CS352A: Machine Learning Lab

Week 12: Naive Bayes Classifier

NAME	SRN	SECTION
C PANSHUL REDDY	PES2UG23CS154	C

DATE:31-10-2025

Introduction

The purpose of this lab is to explore and implement various text classification techniques using the PubMed 20k RCT dataset. The main objective is to classify biomedical sentences into predefined categories by applying and comparing different machine learning models. The tasks performed include:

1. Part A: Building a Count/Frequency-based Naive Bayes classifier from scratch, including manual implementation of the fit and predict methods, and evaluating its performance.
2. Part B: Implementing a TF-IDF score-based classifier using scikit-learn's Multinomial Naive Bayes, followed by hyperparameter tuning with GridSearchCV to optimize model performance.
3. Part C: Constructing a Bayes Optimal Classifier (BOC) by ensembling five diverse models (Naive Bayes, Logistic Regression, Random Forest, Decision Tree, and K-Nearest Neighbors) and weighting them based on posterior probabilities derived from validation log-likelihoods.

Throughout the lab, model performance is assessed using metrics such as accuracy, macro-averaged F1 score, classification reports, and confusion matrix visualizations. The final section compares all models to determine the best approach for biomedical text classification.

Methodology

Multinomial Naive Bayes (MNB)

The MNB classifier was implemented from scratch to classify biomedical sentences. The approach involved:

- Converting text data into count-based feature vectors using CountVectorizer.
- Calculating class priors and feature likelihoods with Laplace smoothing.

- Computing log probabilities for each class and predicting the class with the highest score for each test sample.
- Model performance was evaluated using accuracy, macro-averaged F1 score, and confusion matrix visualization.

Bayes Optimal Classifier (BOC)

The BOC was constructed by ensembling five diverse models: Naive Bayes, Logistic Regression, Random Forest, Decision Tree, and K-Nearest Neighbors.

- Each model was trained on a dynamically sampled subset of the training data.
- Posterior weights for each model were computed based on their validation log-likelihoods, reflecting their relative performance.
- A soft-voting ensemble (VotingClassifier) was used, with each model's vote weighted by its posterior probability.
- The ensemble's predictions on the test set were evaluated using the same metrics as above, and results were compared to select the best-performing model.

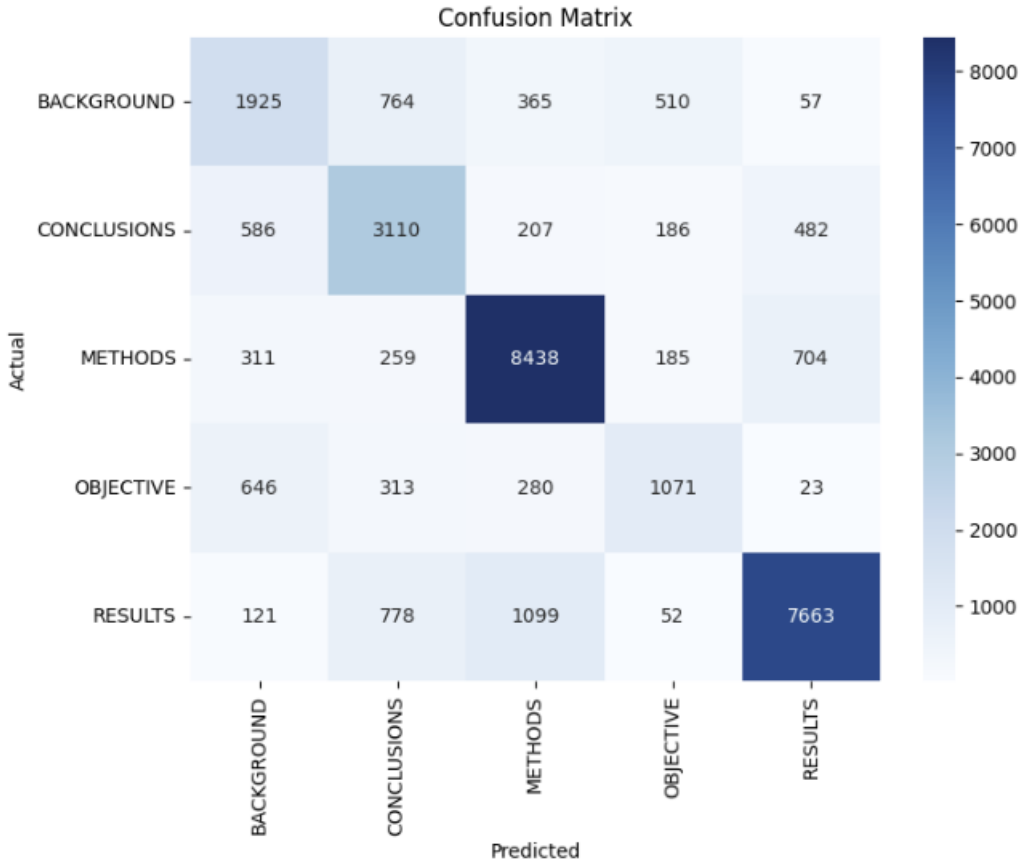
Results and Analysis

PART A:

```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
Accuracy: 0.7369
              precision    recall  f1-score   support

  BACKGROUND       0.54      0.53      0.53      3621
 CONCLUSIONS       0.60      0.68      0.64      4571
     METHODS       0.81      0.85      0.83      9897
   OBJECTIVE       0.53      0.46      0.49      2333
     RESULTS       0.86      0.79      0.82      9713

    accuracy                           0.74     30135
   macro avg       0.67      0.66      0.66     30135
weighted avg       0.74      0.74      0.74     30135

Macro-averaged F1 score: 0.6634
```

Final test accuracy:0.7369

F1-score:0.6634



PART B:

```
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: {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 2)}
Best cross-validation score: 0.6567
```

Best hyperparameters found:

- nb__alpha: 0.1
- tfidf__ngram_range: (1, 2)

Resulting f1 score: 0.6567

PART C:

SRN:PES2UG23CS154

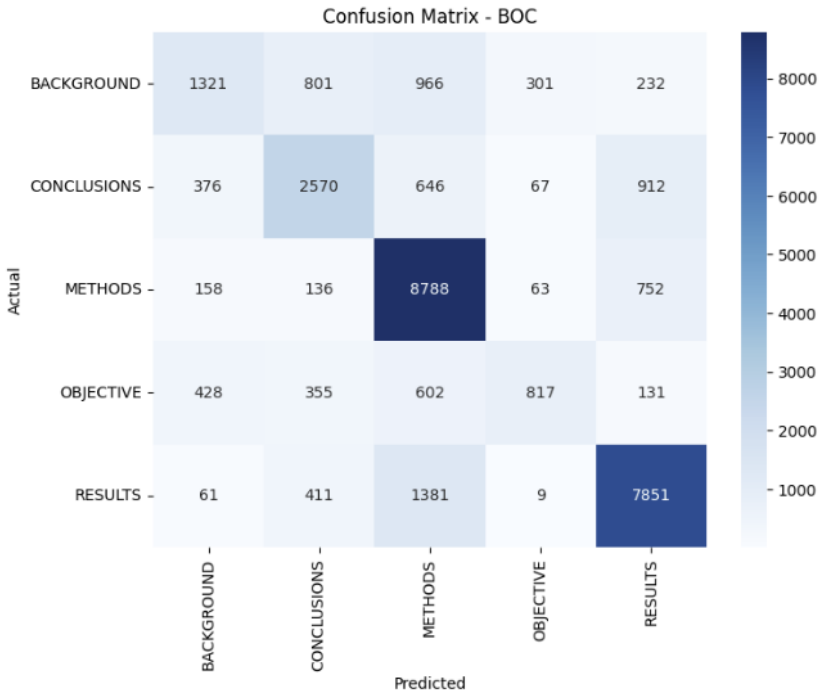
```
# Dynamic Data Sampling (DO NOT CHANGE)
BASE_SAMPLE_SIZE = 10000

# Prompt the user for their full SRN
FULL_SRN = "PES2UG23CS154"  # Example SRN
```

Sample size:

```
Using dynamic sample size: 10154
Actual sampled training set size used: 10154
```

BOC final metrics:

```
Training all base models...
Trained NaiveBayes
c:\Users\cpans\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\linear_model\_logistic.py:1272: FutureWarning: 'multi_class' was deprecated in version 1.5 and will be removed in 1.7. From then on, it will always use 'multi
  warnings.warn(
c:\Users\cpans\AppData\Local\Programs\Python\Python312\Lib\site-packages\sklearn\linear_model\_logistic.py:1296: FutureWarning: Using the 'liblinear' solver for multiclass classification is deprecated. An error will be raised in 1.8. Eithe
  warnings.warn(
Trained LogisticRegression
Trained RandomForest
Trained DecisionTree
Trained KNN
All base models trained.

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

Predicting on test set...

=== Final Evaluation: Bayes Optimal Classifier (Soft Voting) ===
Accuracy: 0.7084
              precision    recall  f1-score   support

  BACKGROUND       0.56      0.36      0.44      3621
 CONCLUSIONS       0.60      0.56      0.58      4571
     METHODS       0.71      0.89      0.79      9897
   OBJECTIVE       0.65      0.35      0.46      2333
     RESULTS       0.79      0.81      0.80      9713

    accuracy                           0.71     30135
   macro avg       0.66      0.59      0.61     30135
weighted avg       0.70      0.71      0.69     30135

Macro-averaged F1 score: 0.6139
```



Model Performance Comparison

Part A: Custom Count-based Naive Bayes (Scratch Model)

- Achieved an accuracy of 0.7369 and a macro-averaged F1 score of 0.6634.
- This model, built from scratch with Laplace smoothing and count-based features, showed strong performance.
- Its simplicity and direct feature representation contributed to robust results.

Part B: Tuned Sklearn TF-IDF Naive Bayes

- After hyperparameter tuning (nb__alpha: 0.1, tfidf__ngram_range: (1, 2)), the model reached an accuracy of 0.7604 and a macro F1 score of 0.6599.
- The use of TF-IDF features and n-gram expansion improved the model's ability to capture more complex patterns in the text.

- Although the tuned model had the highest accuracy, its macro F1 score was slightly lower than the scratch model, indicating some class imbalance in predictions.

Part C: Bayes Optimal Classifier (BOC) Approximation

- The BOC ensemble, combining five diverse models with posterior weighting, achieved an accuracy of 0.7084 and a macro F1 score of 0.6139.
- While the ensemble approach leveraged the strengths of multiple classifiers, its overall performance was lower than both the scratch and tuned Naive Bayes models.
- The complexity of combining models did not translate to a significant boost in macro F1 score, possibly due to overlapping errors among base models.

Summary

- The scratch Naive Bayes model (Part A) provided a strong baseline with balanced performance.
- The tuned Sklearn model (Part B) achieved the highest accuracy, but only a marginal improvement in F1 score.
- The BOC ensemble (Part C) offered diversity but did not outperform the simpler models.
- Overall, careful feature engineering and hyperparameter tuning had a greater impact on performance than model ensembling in this task.