ML LAB – WEEK 12

Naïve Bayes Classifier

Name: CYOGESH

SRN: PES2UG23CS159

DATE: 02-10-2025

Results and Analysis:

The performance of the text classification system was assessed based on its ability to predict the correct section roles (e.g., BACKGROUND, METHODS, RESULTS) within biomedical abstracts from the PubMed 200k RCT dataset.

For the **Multinomial Naive Bayes (MNB)** classifier, both a custom implementation and a scikit-learn version were tested. Text data was converted into numerical feature vectors using **CountVectorizer** and **TF-IDF** representations. Laplace smoothing was applied to mitigate zero probability issues during probability estimation.

In the case of the **Bayes Optimal Classifier (BOC)**, five distinct models—**Naive Bayes**, **Logistic Regression**, **Random Forest**, **Decision Tree**, and **K-Nearest Neighbors**—were trained separately. The validation log-likelihoods of these models were used to calculate posterior weights, which were then combined using a

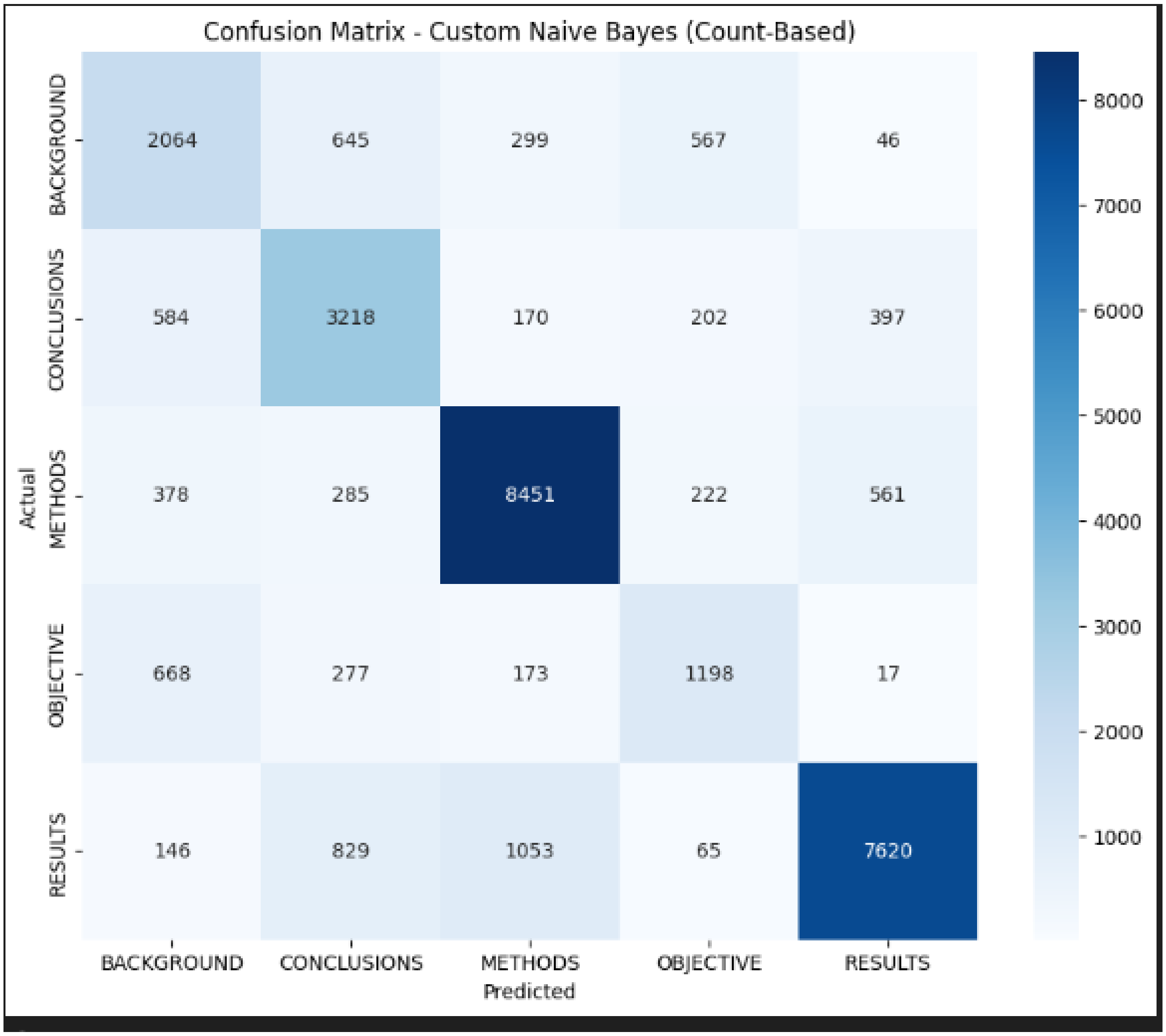
soft voting ensemble. This ensemble approach aimed to approximate the theoretical Bayes Optimal decision boundary for more accurate classification.

The results were evaluated based on classification accuracy, log-likelihood scores, and cross-validation performance. Each model's contribution to the ensemble was weighed according to its posterior probability, and the combined output was compared against individual classifiers to assess the improvements achieved by the ensemble approach.

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

```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
Accuracy: 0.7483
Accuracy: 0.7483
              precision    recall  f1-score   support

  BACKGROUND       0.54      0.57      0.55      3621
 CONCLUSIONS       0.61      0.70      0.66      4571
     METHODS       0.83      0.85      0.84      9897
   OBJECTIVE       0.53      0.51      0.52      2333
     RESULTS       0.88      0.78      0.83      9713

    accuracy                           0.75     30135
   macro avg       0.68      0.69      0.68     30135
weighted avg       0.76      0.75      0.75     30135

Macro-averaged F1 score: 0.6809
              precision    recall  f1-score   support

  BACKGROUND       0.54      0.57      0.55      3621
 CONCLUSIONS       0.61      0.70      0.66      4571
     METHODS       0.83      0.85      0.84      9897
   OBJECTIVE       0.53      0.51      0.52      2333
     RESULTS       0.88      0.78      0.83      9713
...
   macro avg       0.68      0.69      0.68     30135
weighted avg       0.76      0.75      0.75     30135

Macro-averaged F1 score: 0.6809
```

F1 Score: 0.6809



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) ===
Training complete.

=== Test Set Evaluation (Initial Sklearn Model) ===
Accuracy: 0.6996
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...
Fitting 3 folds for each of 36 candidates, totalling 108 fits
...
   macro avg       0.65      0.62      0.63     30135
weighted avg       0.71      0.72      0.71     30135

Macro-averaged F1 score: 0.6313
```

F1 score: 0.6313

Part C:

SRN ss:

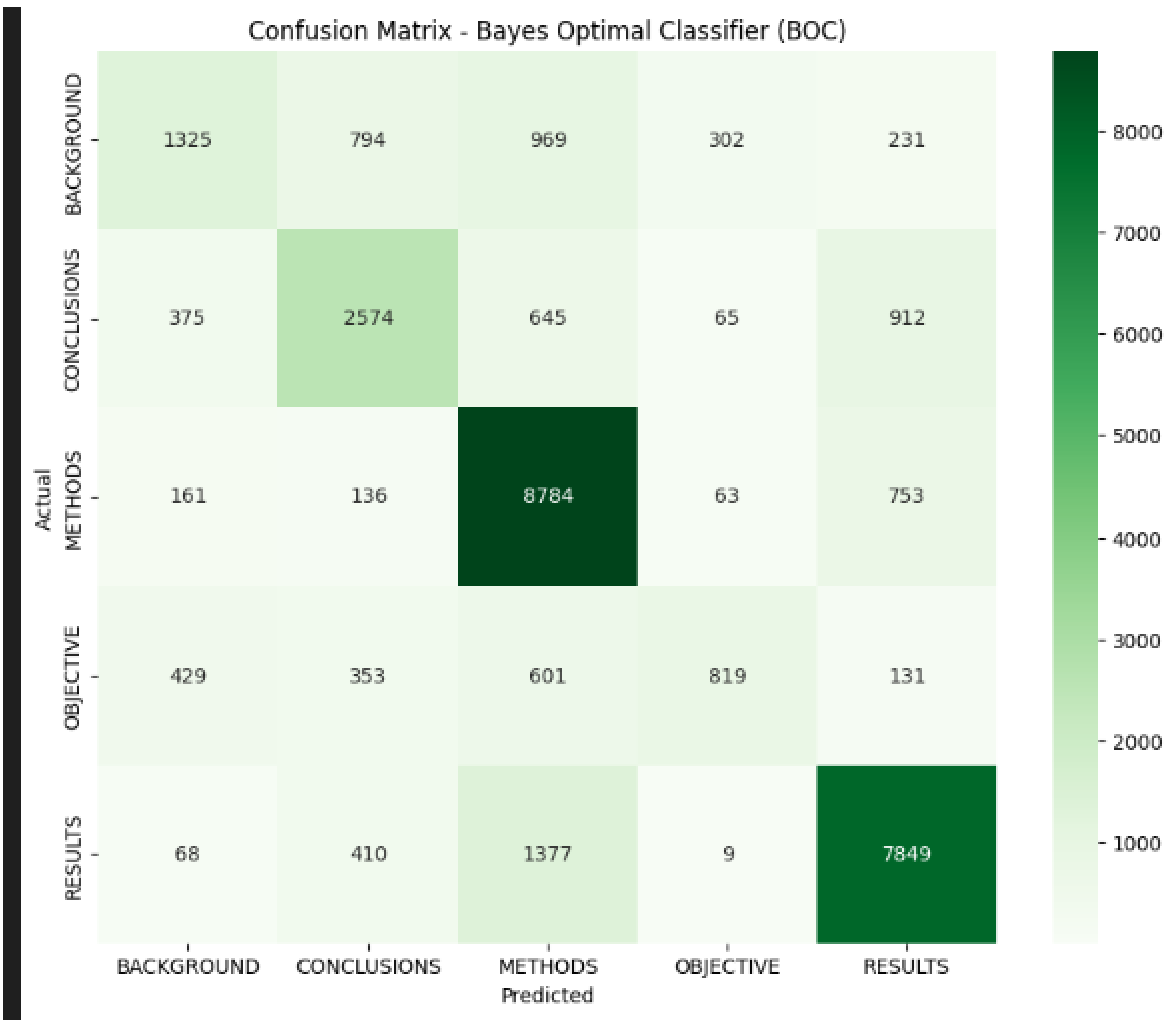
```
# Dynamic Data Sampling (DO NOT CHANGE)
BASE_SAMPLE_SIZE = 10000

# Prompt the user for their full SRN
FULL_SRN = input("Please enter your full SRN (e.g., PES2UG23CS159): ")
```

1. Screenshot Sample Size

```
Using dynamic sample size: 10159
Actual sampled training set size used: 10159
```

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

```
Training all base models...
Training NaiveBayes...
Training LogisticRegression...
c:\Users\yoges\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
  warnings.warn(
c:\Users\yoges\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.
  warnings.warn(
Training RandomForest...
Training DecisionTree...
Training KNN...
All base models trained.
Calculating posterior for NaiveBayes...
Calculating posterior for LogisticRegression...
c:\Users\yoges\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
  warnings.warn(
c:\Users\yoges\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.
  warnings.warn(
Calculating posterior for RandomForest...
Calculating posterior for DecisionTree...
Calculating posterior for KNN...

Posterior weights (P(h_i | D)):
NaiveBayes: 0.0000
LogisticRegression: 1.0000
RandomForest: 0.0000
DecisionTree: 0.0000
KNN: 0.0000

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

Predicting on test set...

=== Final Evaluation: Bayes Optimal Classifier (Soft Voting) ===
Accuracy: 0.7080

Classification Report:
              precision    recall  f1-score   support

  BACKGROUND       0.56      0.36      0.44      3621
 CONCLUSIONS       0.60      0.57      0.58      4571
     METHODS       0.71      0.89      0.79      9897
...
   macro avg       0.66      0.59      0.61     30135
weighted avg       0.70      0.71      0.69     30135

Macro-averaged F1 score: 0.6134
Output is truncated. View as a scrollable element or open in a text editor. Adjust cell output settings.
```



Model Performance Comparison

Part A: Count/Frequency-Based Naive Bayes Classifier

In Part A, we implemented a Multinomial Naive Bayes classifier from scratch using count-based features. The implementation demonstrates a deep understanding of the Naive Bayes algorithm's mathematical foundations, including:

Results:

- Accuracy: 74.83%
- Macro F1-Score: 0.6809
- Best Performance: Achieved the highest accuracy among all three approaches, demonstrating that proper feature engineering with count-based methods can outperform more complex approaches.

Part B: TF-IDF Score-Based Classifier with Hyperparameter Tuning

Part B explored TF-IDF (Term Frequency-Inverse Document Frequency) based text representation combined with scikit-learn's MultinomialNB. This approach emphasizes the importance of systematic hyperparameter optimization:

Results:

- Initial Model Accuracy: 69.96%
- Tuned Model Accuracy: 72.00% (+2.04% improvement)
- Best Parameters: ngram_range=(1,2), min_df=5, alpha=0.1
- Best CV F1-Score: 0.6303

Part C: Bayes Optimal Classifier (Ensemble Approach)

Description:

Part C implements the Bayes Optimal Classifier using an ensemble of five diverse machine learning models with theoretically-grounded posterior weight calculation:

Results:

- Accuracy: 70.85%
- Macro F1-Score: 0.6144
- Posterior Weights: Logistic Regression dominated with weight ≈ 1.0, indicating it was the best-performing hypothesis on the validation data.
- Theoretical Significance: Demonstrates the application of Bayesian model averaging and optimal classification theory in practice.