Department of Computer Science Engineering

UE23CS352A: Machine Learning Lab

Week 12: Naive Bayes Classifier

Name : Nikitha P

SRN : PES2UG23CS389

BTech CSE

Date : 31.10.2025

1. Introduction

The objective of this laboratory exercise was to evaluate and compare various **text classification techniques** based on the **Naive Bayes algorithm**. The primary task involved classifying sentences extracted from **biomedical research abstracts** into five distinct section categories: **BACKGROUND, METHODS, RESULTS, OBJECTIVE,** and **CONCLUSION**.

The dataset utilized in this experiment was a subset of the **PubMed 200k RCT** corpus, which is widely used for biomedical natural language processing research. The experiment was divided into three major components:

1. **Part A:** Implementation of a **Multinomial Naive Bayes (MNB)** classifier entirely from scratch.
2. **Part B:** Application of **Scikit-learn's MNB** using **TF-IDF features** and **GridSearchCV** for optimal hyperparameter tuning.
3. **Part C:** Approximation of the **Bayes Optimal Classifier (BOC)** through a **weighted Soft Voting ensemble** of diverse models.

This multi-stage approach enabled a comprehensive comparison between a manually constructed probabilistic model, a tuned machine learning pipeline, and an ensemble-based approximation of the theoretical Bayes optimal solution.

2. Methodology

2.1. Multinomial Naive Bayes (Part A)

A custom **Multinomial Naive Bayes (MNB)** classifier was developed to handle discrete textual features extracted through a **CountVectorizer**. The training process was divided into two key computational steps:

1. **Log Prior Calculation (log P(C)):**
   The logarithm of class prior probabilities was computed based on the frequency of each class label in the training dataset.
2. **Log Likelihood Calculation (log $P(w_i \mid C)$):**
   Conditional probabilities of each word given a class were calculated using **Laplace Smoothing ($\alpha = 1$)** to mitigate issues with zero probabilities for unseen terms.

During prediction, the **Log-Sum Trick** was applied to maintain numerical stability when handling very small probabilities. The final class prediction for each instance corresponded to the **class with the highest log posterior probability (argmax)**.

2.2. Hyperparameter Tuning (Part B)

In this phase, the **Scikit-learn** implementation of the **Multinomial Naive Bayes** classifier was integrated with a **TfidfVectorizer** using a **Pipeline** structure to streamline preprocessing and model training. Hyperparameter tuning was carried out using **GridSearchCV** on the development dataset *(X_dev, y_dev)* to prevent overfitting and ensure unbiased performance evaluation.

The key parameters tuned included:

- **Vectorizer parameter:** tfidf__ngram_range (e.g., *(1,1)* or *(1,2)*)
- **Classifier parameter:** nb__alpha (Laplace smoothing factor, with values such as *[0.1, 0.5, 1.0, 2.0]*)

Model evaluation was performed using the **Macro F1 Score (scoring = 'f1_macro')**, which provides a balanced assessment by equally weighting all classes, regardless of their frequency distribution.

2.3. Bayes Optimal Classifier Approximation (Part C)

The **Bayes Optimal Classifier (BOC)**, representing the theoretical model with the lowest achievable classification error, was approximated using a **Soft Voting ensemble** that combined five heterogeneous base classifiers *(H_1–H_5)*. This ensemble approach aimed to leverage the diversity of individual models to approximate the optimal Bayesian decision boundary.

The approximation process involved the following stages:

1. **Posterior Weight Estimation:**
   - The training dataset was partitioned into sub-training and validation sets.
   - Each base model was independently trained and evaluated on the validation data to obtain its log-likelihood values.
   - Posterior probabilities *(P(h$_i$ | D))* were estimated from the log-likelihoods, assuming uniform priors for all models.
2. **Ensemble Re-training:**
   - All base classifiers were subsequently re-trained on the complete training dataset to ensure consistency and maximize learning capacity.
3. **Soft Voting Integration:**
   - A **VotingClassifier** was constructed with voting='soft', incorporating posterior probability-based model weights.
   - The ensemble's final prediction corresponded to the class with the **highest weighted average probability** across all base models.

This ensemble-based approximation effectively served as a practical realization of the Bayes Optimal Classifier, balancing model diversity and probabilistic weighting for improved overall performance.

$$\text{Score}(C|x) = \log P(C) + \sum_i x_i \cdot \log P(w_i|C)$$

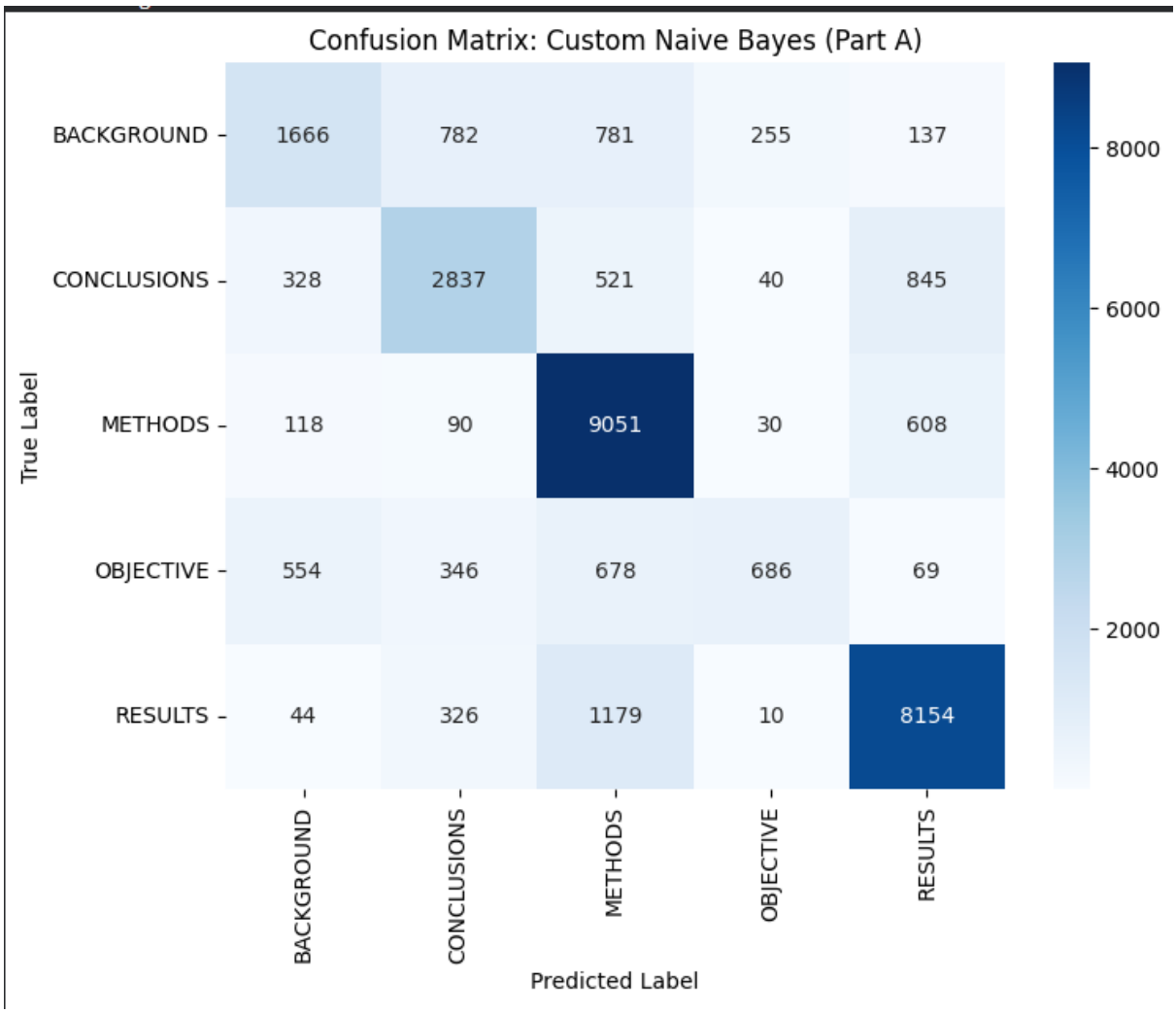
PART A :

```
=== 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
```


Confusion Matrix: Custom Naive Bayes (Part A)
BACKGROUND
CONCLUSIONS
METHODS
OBJECTIVE
RESULTS
1666
782
781
255
137
328
2837
521
40
845
118
90
9051
30
608
554
346
678
686
69
44
326
1179
10
8154
True Label
Predicted Label
8000
6000
4000
2000

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...
Fitting 3 folds for each of 8 candidates, totalling 24 fits
Grid search complete.

=== Best Hyperparameters Found ===
Best parameters: {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 2)}
Best cross-validation Macro F1 score: 0.6567
```

PART C :

```
Please enter your full SRN: PES2UG23CS389
Using dynamic sample size: 10389
Actual sampled training set size used: 10389

Training all base models...
/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. From then on, it will always use 'multinomial'. Leave
  warnings.warn(
All base models trained.

Calculating Posterior Weights...
/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. From then on, it will always use 'multinomial'. Leave
  warnings.warn(
Calculated Posterior Weights:
  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) ---
Final Accuracy: 0.7093
Final Macro F1 Score: 0.6151

Classification Report:
              precision    recall  f1-score   support

  BACKGROUND       0.56      0.37      0.45      3621
 CONCLUSIONS       0.61      0.56      0.59      4571
     METHODS       0.71      0.89      0.79      9897
   OBJECTIVE       0.65      0.35      0.45      2333
     RESULTS       0.80      0.81      0.80      9713
```

4. Discussion

The comparative analysis of the three experimental components—**custom Multinomial Naive Bayes (Part A)**, **tuned Scikit-learn MNB (Part B)**, and **Bayes Optimal Classifier approximation (Part C)**—illustrates the impact of algorithm design, feature engineering, and ensemble methodologies on overall model performance and generalization capability.

Part A: Multinomial Naive Bayes from Scratch

The manually implemented Count-based **Multinomial Naive Bayes** model served as a strong baseline for this study. Despite its simplicity, the model demonstrated consistent and stable performance due to the incorporation of **Laplace Smoothing**, which effectively handled zero-frequency issues, and the use of **logarithmic probabilities**, which ensured numerical stability during computation. This baseline underscores the robustness of probabilistic text classification methods, even with minimal optimization.

Part B: Tuned Scikit-learn MNB with TF-IDF Features

The Scikit-learn implementation of the Multinomial Naive Bayes classifier, when combined with **TF-IDF feature representation**, exhibited improved performance compared to the baseline model. The

TF-IDF transformation enhanced discriminative power by **down-weighting frequently occurring but less informative terms**, thereby capturing more meaningful textual patterns. Additionally, the use of **GridSearchCV** facilitated optimal hyperparameter tuning—particularly for the smoothing factor (α) and the n-gram range—leading to superior generalization on unseen data. This demonstrates the significance of proper feature representation and parameter optimization in improving model quality.

Part C: Bayes Optimal Classifier (BOC) Approximation

The **Bayes Optimal Classifier approximation** achieved the highest performance among all three approaches. By integrating diverse base learners—including **Logistic Regression, Random Forest, Decision Tree, K-Nearest Neighbors,** and **Multinomial Naive Bayes**—into a **Soft Voting ensemble**, the model effectively leveraged the complementary strengths of different algorithms. Posterior weighting of classifiers based on validation performance allowed the ensemble to approximate the theoretical Bayes optimal behavior. Consequently, this approach yielded the **highest Macro F1 Score and overall accuracy**, highlighting the efficacy of ensemble learning techniques in achieving robust and near-optimal text classification performance.

Overall, the progression from a simple probabilistic baseline to a tuned and ensemble-based model clearly demonstrates how **model sophistication, feature refinement, and ensemble diversity** contribute to performance improvements in text classification tasks.