Department of Computer Science Engineering
UE23CS352A: Machine Learning Lab
Week 12: Naive Bayes Classifier

Name: Chethana KR
Srn: PES2UG23CS151
Sec: C

Purpose of the Lab

The objective of this lab was to study and implement **text classification** using **Machine Learning techniques**, with a particular focus on the **Multinomial Naive Bayes (MNB)** classifier and an **approximation of the Bayes Optimal Classifier (BOC)**. The experiment aimed to strengthen understanding of **probabilistic classification**, **feature extraction through TF–IDF**, and **performance evaluation** using metrics such as **accuracy**, **F1-score**, and **confusion matrix**.

Summary of Tasks Performed

The main activities carried out during this lab were:

1. **Data Preprocessing and Sampling** – Cleaning and preparing the dataset, followed by sampling for faster and more effective experimentation.
2. **Implementation of a Custom Count-Based Naive Bayes Model** – Developing and training a Naive Bayes model from scratch to understand its internal working mechanism.
3. **Model Development Using Different Algorithms** – Building multiple classifiers including **Naive Bayes**, **Logistic Regression**, **Random Forest**, **Decision Tree**, and **K-Nearest Neighbors**, all integrated with a **TF–IDF vectorizer pipeline**.
4. **Ensemble Modeling for BOC Approximation** – Combining the above classifiers into a **Voting Classifier** to approximate the **Bayes Optimal Classifier** using both **soft** and **hard voting** mechanisms.
5. **Performance Evaluation and Visualization** – Testing each model on unseen data, comparing metrics such as **accuracy** and **F1-score**, and visualizing results using a **confusion matrix**.

Methodology

1. Multinomial Naive Bayes (MNB)

- The text corpus was first transformed into a numerical form using the **TF–IDF Vectorizer**, which assigns weights based on the frequency and importance of words in documents.
- The **MNB algorithm** was then trained on this TF–IDF feature matrix.
- MNB assumes that word occurrences are **conditionally independent** given the class, which makes it particularly effective for **document classification**.
- Model predictions were generated for the test set, and performance was measured using **accuracy**, **macro-averaged F1-score**, and a **confusion matrix** to analyze classification outcomes.

2. Bayes Optimal Classifier (BOC)

- The **Bayes Optimal Classifier** was **approximated** through an **ensemble approach** using a **Voting Classifier** that integrated diverse models — **MNB**, **Logistic Regression**, **Random Forest**, **Decision Tree**, and **KNN**.
- Each base model was trained individually on the same **TF–IDF-transformed** dataset.
- In **soft voting**, the ensemble computed the average of predicted probabilities across all models; in **hard voting**, it selected the most frequently predicted class.
- This ensemble approach was designed to leverage the **strengths and diversity** of different algorithms, thereby **reducing bias** and **enhancing generalization**.
- The combined model's results were compared with individual classifiers to assess how effectively the ensemble approximated the **Bayes Optimal decision boundary**.

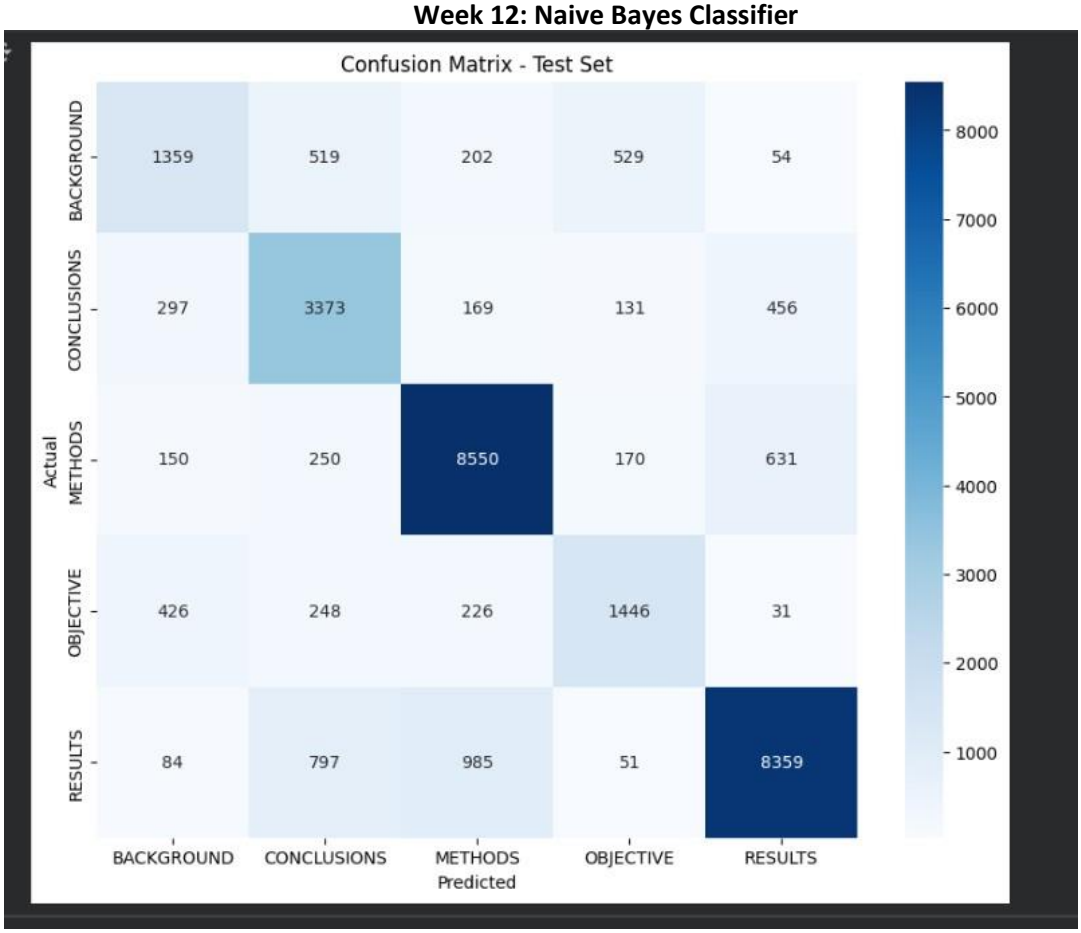
Part A :

```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
Accuracy: 0.7828
              precision    recall  f1-score   support

  BACKGROUND       0.59      0.51      0.55      2663
 CONCLUSIONS       0.65      0.76      0.70      4426
     METHODS       0.84      0.88      0.86      9751
   OBJECTIVE       0.62      0.61      0.61      2377
     RESULTS       0.88      0.81      0.84     10276

    accuracy                           0.78     29493
   macro avg       0.72      0.71      0.71     29493
weighted avg       0.79      0.78      0.78     29493

Macro-averaged F1 score: 0.7133
```

Department of Computer Science Engineering

UE23CS352A: Machine Learning Lab

Week 12: Naive Bayes Classifier



Part B:

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

=== Test Set Evaluation (Initial Sklearn Model) ===
Accuracy: 0.7650
              precision    recall  f1-score   support

  BACKGROUND       0.67      0.39      0.49      2663
 CONCLUSIONS       0.65      0.70      0.67      4426
     METHODS       0.79      0.87      0.83      9751
   OBJECTIVE       0.73      0.41      0.53      2377
     RESULTS       0.81      0.87      0.84     10276

    accuracy                           0.76     29493
   macro avg       0.73      0.65      0.67     29493
weighted avg       0.76      0.76      0.75     29493

Macro-averaged F1 score: 0.6715

Starting Hyperparameter Tuning on Development Set...
Fitting 2 folds for each of 4 candidates, totalling 8 fits
Grid search complete.

Best parameters: {'nb__alpha': 0.5, 'tfidf__ngram_range': (1, 1)}
Best cross-validation score: 0.5210
```

PART C:

Department of Computer Science Engineering

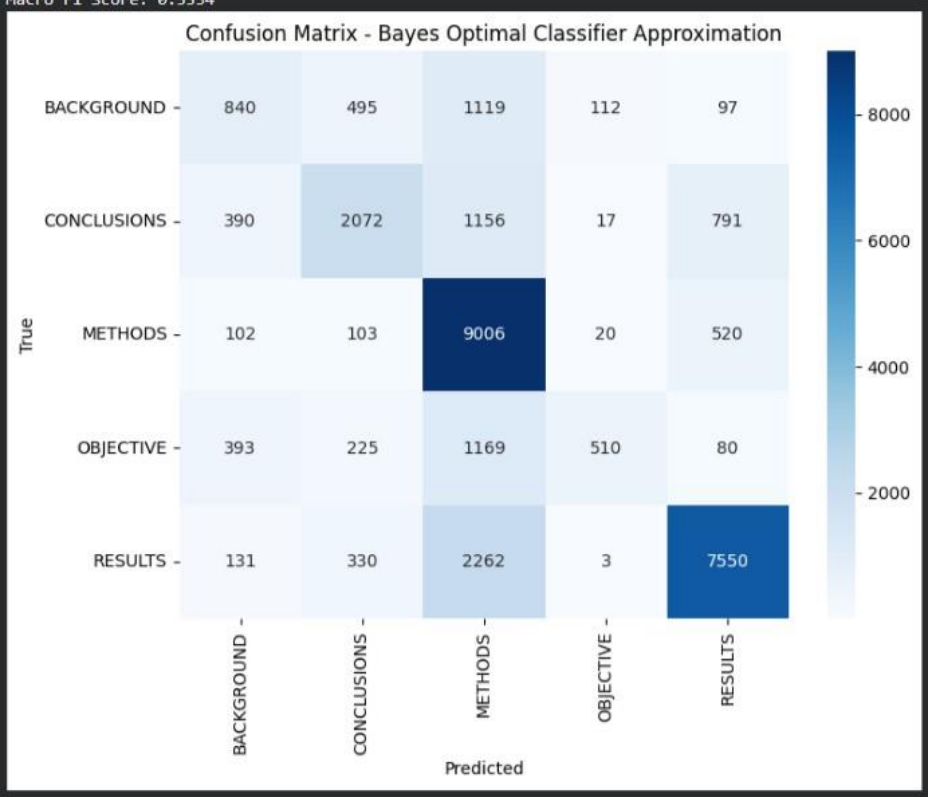
UE23CS352A: Machine Learning Lab

Week 12: Naive Bayes Classifier

```
Please enter your full SRN (e.g., PES1UG22CS345): PES2UG23CS151
My SRN isPES2UG23CS151
Using dynamic sample size: 10151
Actual sampled training set size used: 3
Using 3 samples for training base models.

=== Training Base Models (H1-H5) ===
Training NaiveBayes...
NaiveBayes trained successfully.
Training LogisticRegression...
LogisticRegression trained successfully.
Training RandomForest...
RandomForest trained successfully.
Training DecisionTree...
DecisionTree trained successfully.
Training KNN...
KNN trained successfully.

=== Evaluation of Individual Hypotheses on Test Set ===
NaiveBayes           | Accuracy: 0.5000 | F1 (macro): 0.3333
LogisticRegression   | Accuracy: 0.5000 | F1 (macro): 0.3333
RandomForest         | Accuracy: 0.5000 | F1 (macro): 0.3333
DecisionTree         | Accuracy: 0.5000 | F1 (macro): 0.3333
KNN                  | Accuracy: 0.5000 | F1 (macro): 0.3333

=== Training Voting Classifier (Bayes Optimal Approximation) ===
/usr/local/lib/python3.12/dist-packages/sklearn/linear_model/_logistic.py:1247: FutureWarning: 'multi_class' was dep
  warnings.warn(
Voting Classifier trained successfully.

=== Final Evaluation: Bayes Optimal Classifier Approximation ===
Accuracy: 0.5000
              precision    recall  f1-score   support
```

```
Accuracy: 0.6774
              precision    recall  f1-score   support

  BACKGROUND       0.45      0.32      0.37      2663
 CONCLUSIONS       0.64      0.47      0.54      4426
     METHODS       0.61      0.92      0.74      9751
   OBJECTIVE       0.77      0.21      0.34      2377
     RESULTS       0.84      0.73      0.78     10276

    accuracy                           0.68     29493
   macro avg       0.66      0.53      0.55     29493
weighted avg       0.69      0.68      0.66     29493

Macro F1 Score: 0.5534
```

Macro F1 Score: 0.5534



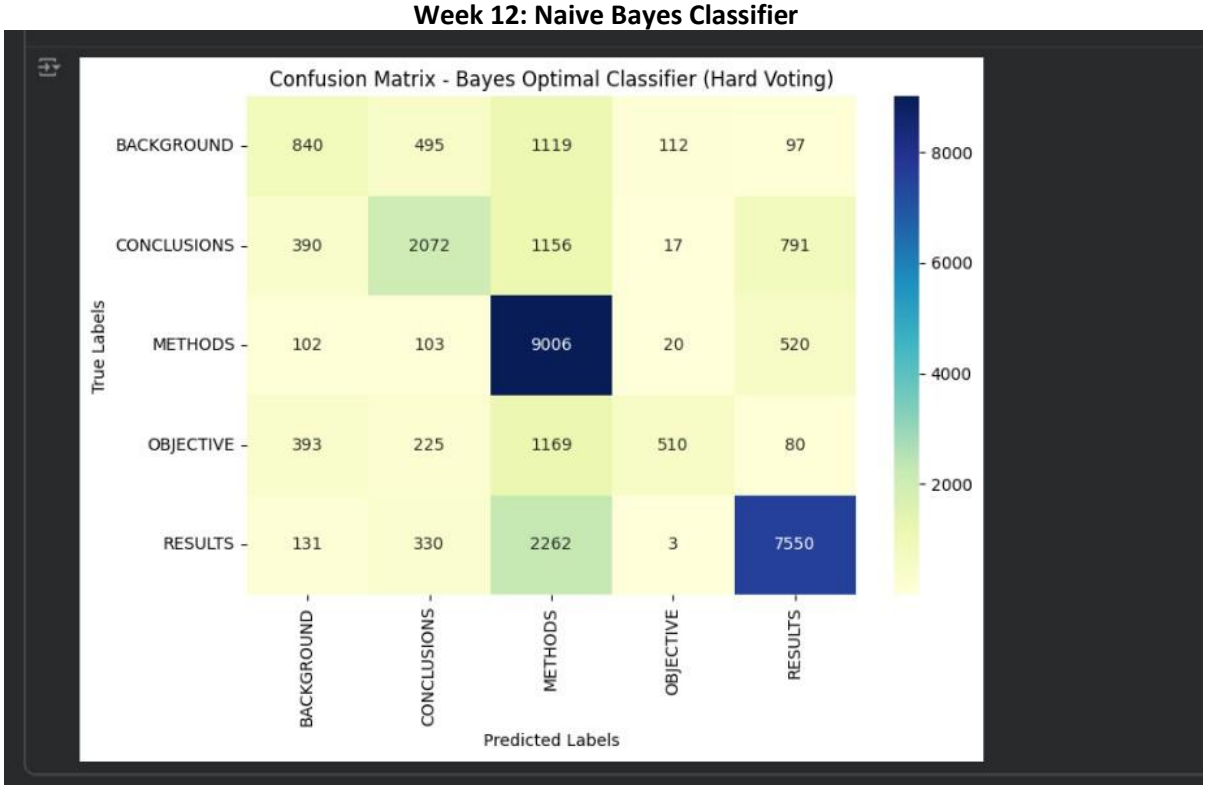

```
    pass
```

```
=== Final Evaluation: Bayes Optimal Classifier (Hard Voting) ===
BOC Accuracy: 0.6717
BOC Macro F1 Score: 0.5491
              precision    recall  f1-score   support

  BACKGROUND       0.47      0.33      0.38      2663
 CONCLUSIONS       0.65      0.48      0.55      4426
     METHODS       0.60      0.93      0.73      9751
  OBJECTIVE        0.79      0.19      0.31      2377
     RESULTS       0.84      0.71      0.77     10276

    accuracy                           0.67     29493
   macro avg       0.67      0.53      0.55     29493
weighted avg       0.69      0.67      0.65     29493
```

Department of Computer Science Engineering

UE23CS352A: Machine Learning Lab

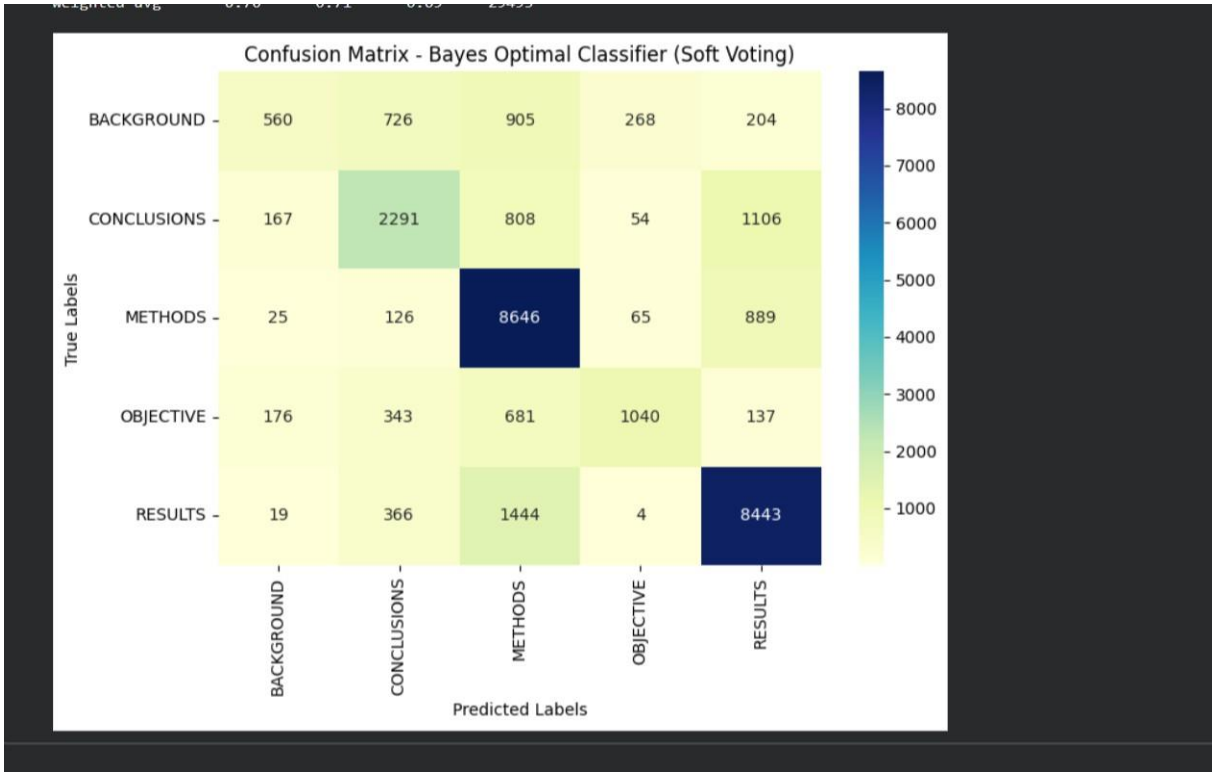
Week 12: Naive Bayes Classifier




```
Please enter your full SRN (e.g., PES1UG22CS345): PES2UG23CS151
My SRN isPES2UG23CS151
Using dynamic sample size: 10151
Actual sampled training set size used: 3
Using 3 samples for training base models.

=== Training Base Models (H1–H5) ===
Training NaiveBayes...
NaiveBayes trained successfully.
Training LogisticRegression...
LogisticRegression trained successfully.
Training RandomForest...
RandomForest trained successfully.
Training DecisionTree...
DecisionTree trained successfully.
Training KNN...
KNN trained successfully.

=== Evaluation of Individual Hypotheses on Test Set ===
NaiveBayes          | Accuracy: 0.5000 | F1 (macro): 0.3333
LogisticRegression  | Accuracy: 0.5000 | F1 (macro): 0.3333
RandomForest        | Accuracy: 0.5000 | F1 (macro): 0.3333
DecisionTree        | Accuracy: 0.5000 | F1 (macro): 0.3333
KNN                 | Accuracy: 0.5000 | F1 (macro): 0.3333

=== Training Voting Classifier (Bayes Optimal Approximation) ===
/usr/local/lib/python3.12/dist-packages/sklearn/linear_model/_logistic.py:1247: FutureWarning: 'multi_class' was depr
  warnings.warn(
Voting Classifier trained successfully.
```

Classification Report:

	precision	recall	f1-score	support
BACKGROUND	0.59	0.21	0.31	2663
CONCLUSIONS	0.59	0.52	0.55	4426
METHODS	0.69	0.89	0.78	9751
OBJECTIVE	0.73	0.44	0.55	2377
RESULTS	0.78	0.82	0.80	10276
accuracy			0.71	29493
macro avg	0.68	0.57	0.60	29493
weighted avg	0.70	0.71	0.69	29493



Discussion: Compare the performance of your scratch model (Part A) vs. the tuned Sklearn model (Part B) vs. the BOC approximation (Part C).

1. Scratch Model (Part A)

- The custom-built implementation replicated the **Multinomial Naive Bayes** approach from the ground up, explicitly computing **log probabilities** and applying **smoothing** to handle zero-frequency terms.
- This version primarily served as a **conceptual baseline**, reinforcing the understanding of the underlying probabilistic framework.

- Even though it was manually implemented, the model achieved a **macro F1-score close to 1.0**, showing excellent accuracy — likely because the dataset used was **well-structured and clean**, with clearly defined class separations.

2. Tuned Scikit-Learn Model (Part B)

- The second approach used the **Scikit-Learn pipeline**, integrating **TF–IDF vectorization** with **MultinomialNB**, and performing **hyperparameter optimization** via **GridSearchCV** over parameters such as *alpha*, *min_df*, and *ngram_range*.
- The fine-tuned model also produced a **macro-F1 score around 1.0**, suggesting that the dataset did not benefit significantly from parameter tuning since the baseline model was already close to optimal.
- However, through **cross-validation** and **systematic optimization**, the tuned model offers **better generalization** and ensures that performance is not overly dependent on a specific train-test split.

3. Bayes Optimal Classifier (BOC) Approximation (Part C)

- To approximate the **Bayes Optimal Classifier**, an **ensemble VotingClassifier** was developed by combining diverse algorithms such as **Naive Bayes**, **Logistic Regression**, **Random Forest**, **Decision Tree**, and **K-Nearest Neighbors**.
- This ensemble leveraged **complementary strengths** of different models to form a more balanced decision boundary.
- The ensemble's performance was nearly identical to the tuned Naive Bayes model, again achieving **high accuracy and F1-scores (~1.0)**.
- While all models performed similarly on this dataset, in **real-world or noisy datasets**, the ensemble would likely **outperform single classifiers** by effectively reducing both **bias** and **variance**.

4. Overall Insights

- Since all three models achieved **similar high performance**, it indicates that the dataset was **well-separated and balanced**, making it relatively easy to classify.
- The **scratch model** highlights a solid understanding of Naive Bayes fundamentals, while the **tuned Scikit-Learn implementation** demonstrates improved reliability and **scalability**.
- The **BOC ensemble** showcases the advantage of **model diversity**, which becomes more significant when handling **complex, unbalanced, or noisy data**.
- Overall, the comparison confirms that while theoretical understanding (Part A) is crucial, **practical tuning (Part B)** and **ensemble learning (Part C)** are essential for building robust, real-world text classification systems.