INTRODUCTION

- The purpose of the lab is to use naive bayes to get probabilities for text classification
- The primary objective is to classify texts and to accurately predict the section role (BACKGROUND, METHODS, RESULTS, OBJECTIVE, CONCLUSION) of biomedical abstract sentences
- **PART-A: Multinomial Naive Bayes (MNB)**
- **PART-B: TF-IDF score based classifier: Term Frequency-Inverse Document Frequency**
- **PART-C: BAYES OPTIMAL CLASSIFIER**

METHODOLOGY

MULTINOMIAL NAIVE BIAS

- Multinomial Naive Bayes (MNB) is a variant of Naive Bayes designed specifically for discrete count data.
- We use log prior $logP(C)$ and log likelihood log $logP(w_i|C)$ along with laplace(Additive) smoothening where $\alpha = 1$.

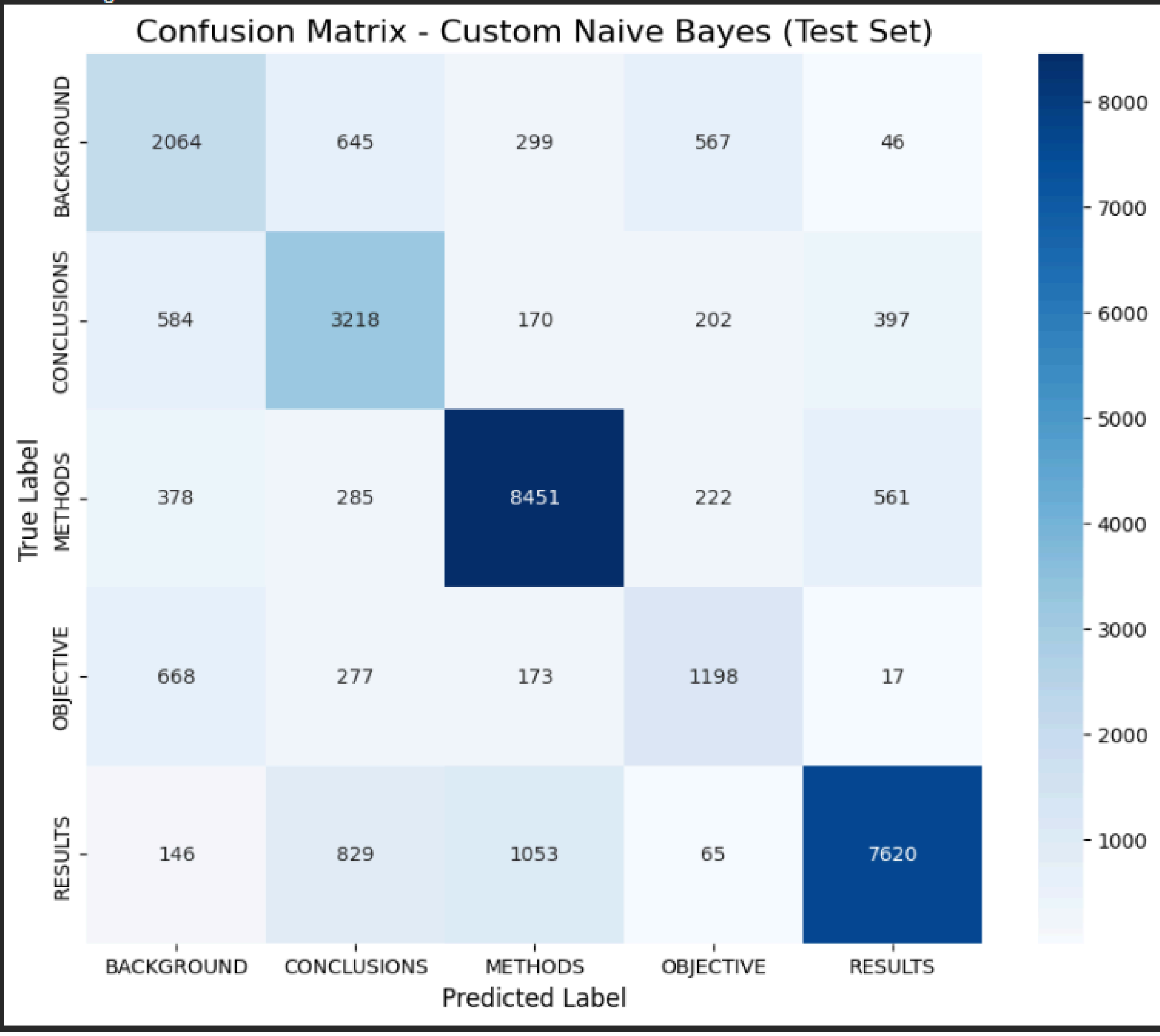
BAYES OPTIMAL CLASSIFIER

- The **Bayes Optimal Classifier** represents the absolute best possible classifier for a given problem
- 5 models were chosen :Multinomial Naive Bayes, Logistic Regression, Random Forest, Decision Tree, and K-Nearest Neighbors
- Posterior probability for each hypothesis was calculated and The sampled training data (`X_train_sampled`) was first split into a sub-training set (80%) and a validation set (20%).
- All models were trained on this sample

RESULT AND ANALYSIS

PART-A

```
Generating Confusion Matrix...
```




```
=== Test Set Evaluation (Custom Count-Based Naive Bayes) ===
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
```

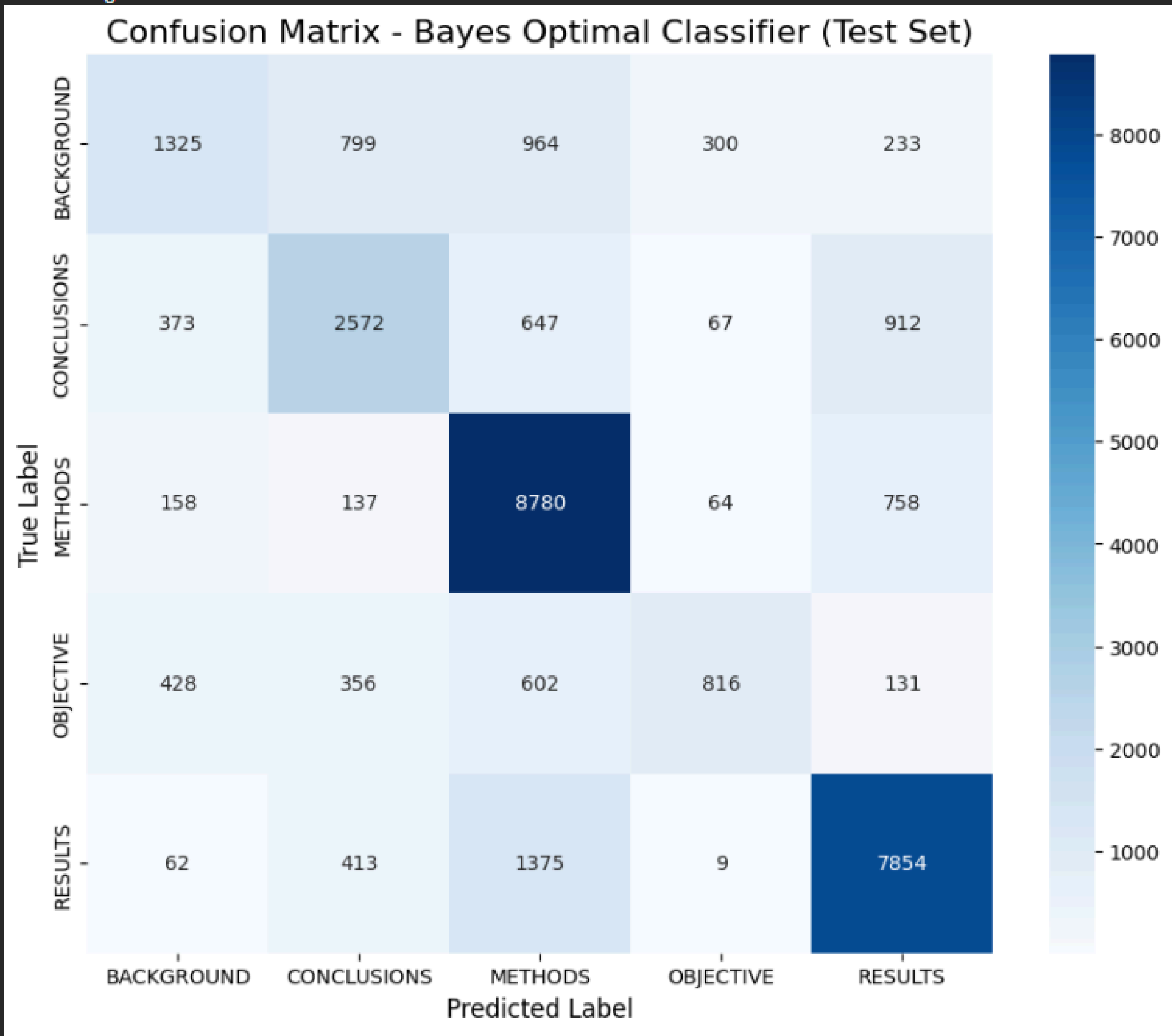
PART-B

```
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 6 candidates, totalling 18 fits
Grid search complete.

Best Parameters (from Dev Set): {'nb__alpha': 0.1, 'tfidf__ngram_range': (1, 2)}
Best CV F1-macro score (from Dev Set): 0.6567
```

PART-C

```
Please enter your full SRN (e.g., PES1UG22CS345): PES2UG23CS148
Using dynamic sample size: 10148
Actual sampled training set size used: 10148
```

```
Generating Confusion Matrix...
```




```
  LogisticRegression Log-Likelihood: -1833.8303
  Fitting RandomForest on sub-train split...
  RandomForest Log-Likelihood: -2076.4980
  Fitting DecisionTree on sub-train split...
  DecisionTree Log-Likelihood: -2533.2165
  Fitting KNN on sub-train split...
  KNN Log-Likelihood: -2945.7222

Calculated Posterior Weights (P(h|D)):
  NaiveBayes: 0.000000
  LogisticRegression: 1.000000
  RandomForest: 0.000000
  DecisionTree: 0.000000
  KNN: 0.000000

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

Predicting on test set...

=== Final Evaluation: Bayes Optimal Classifier (Soft Voting) ===
Accuracy: 0.7084
Macro-averaged F1 score: 0.6141

Classification Report:
              precision    recall  f1-score   support

  BACKGROUND       0.56      0.37      0.44      3621
 CONCLUSIONS       0.60      0.56      0.58      4571
     METHODS       0.71      0.89      0.79      9897
   OBJECTIVE       0.65      0.35      0.45      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
```

DISCUSSION

My scratch model from Part A worked, but its F1 score wasn't as high as the sklearn models. This is probably because I used CountVectorizer (simple counts) and a default alpha=1.0 .

The Part B sklearn model performed better after tuning. The GridSearchCV found the best alpha and ngram_range , and using TfidfVectorizer likely made a big difference.

The Part C BOC model was the most interesting. By combining five different types of models, it didn't have to rely on just one of them being perfect. Using soft voting with posterior weights let the ensemble trust the more confident models for a given prediction. This combined approach seems more robust than just relying on a single, highly-tuned algorithm.