Indian Institute of Technology, Dharwad

॥ सा विद्या या विमुक्तये ॥
ಭಾ.ತಂ.ಸಂ. ಧಾರವಾಡ
भा. प्रौ. सं. धारवाड
I.I.T. DHARWAD

CS209 : Artificial Intelligence
And
CS214 : Artificial Intelligence Laboratory
Project Report :**Classification of music tracks based on their genre**

Course Instructor:
Dr. Dileep A.D.

Mentor Name:
Vikas Kumar

Submitted by:

1. Dontamsetti Kedar Subrahmanya Rishi (MC23BT003)
2. Diyyala Manasa (MC23BT019)
3. Aditya Prabhakar (CS23BT017)
4. Ruthwik Chetan Naik (CS23BT056)

Contents

List of Figures

1 Introduction

Classifying music by genre plays a vital role in enhancing the user experience and organizing vast music collections. It allows streaming platforms to offer personalized recommendations, helps listeners select songs that match their moods, and makes it easier to discover new artists. For music libraries and databases, genre labels support efficient cataloging and retrieval. This classification also aids the music industry in marketing, targeting specific audiences, and structuring award categories based on stylistic differences.

Beyond consumer and industry use, genre classification is essential in fields like artificial intelligence and cultural research. It powers recommendation algorithms, mood detection, and playlist generation in music information retrieval systems. Academically, genres help scholars understand the cultural, historical, and social roots of different music styles and track how they evolve. In essence, music genre classification serves as both a practical tool and a lens for deeper exploration of music's role in society.

In this project, various machine learning models, such as Random Forest and Xgboost, and performance metrics, including accuracy score and F1 score, are also utilised.

2 Dataset Description

The dataset contains **17796** training samples and **7713** test samples with 16 key attributes and 1 target attribute.

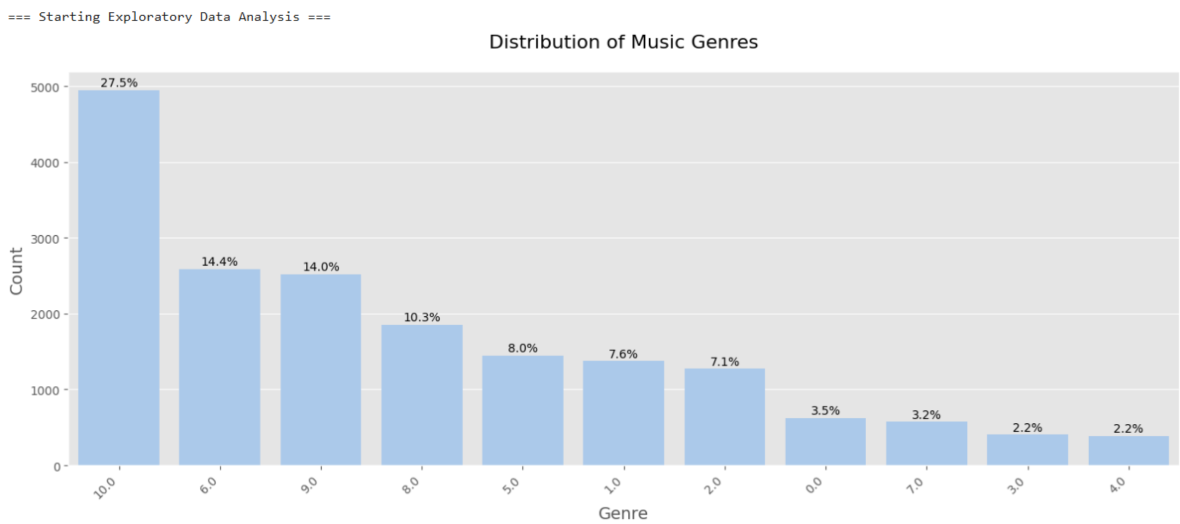

Figure 2.1: Distribution of Music Genre

3 Methodology

3.1 Model Architectures

We used nine model architectures:

- Logistic Regression

- Bayes Classification
- RandomForest Classification
- Neural Networks
- SVM with kernel = 'rbf'
- KNN
- Perceptron with Polynomial Features
- Decision Trees

3.2 Evaluation Metrics

Following were the evaluation metrics used:

- Accuracy Score
- F1 Score
- Recall Score
- Precision Score
- Confusion Matrices

4 Model Implementations

4.1 Logistic Regression

4.1.1 About the Model

Logistic Regression is a supervised machine learning algorithm used for classification problems. Despite the name "regression", it is used to predict categorical outcomes, most commonly binary classification (e.g., yes/no, 0/1, spam/not spam).

It models the probability that a given input belongs to a particular category using the logistic (sigmoid) function.

4.1.2 Advantages

- Efficient for linearly separable data
- Lightweight and faster to train
- Less prone to overfitting

4.1.3 Limitations

- Assumes linear relationship
- Sensitive to outliers
- Requires large sample size

4.2 Bayes Classification

4.2.1 About the model

Bayes Classification is a probabilistic classifier based on Bayes' Theorem. It predicts the class of a given data point by calculating the posterior probability for each class and picking the most probable one.

The most common form is the Naive Bayes Classifier, which makes a "naive" assumption that the features are independent given the class.

4.2.2 Advantages

- Training and prediction are super quick
- Works well with High-Dimensional Data
- Simple and Interpretable

4.2.3 Limitations

- Strong Independence Assumption
- If a feature never appears in training data for a class, it assigns zero probability
- If features are dependent, performance drops.

4.3 Random Forest Classification

4.3.1 About the model

Random Forest is an ensemble learning method that combines multiple decision trees to produce a more accurate and stable prediction. It creates many decision trees using bootstrapped samples of the data (i.e., random subsets with replacement). At each split in a tree, it considers only a random subset of features (not all).

4.3.2 Advantages

- High Accuracy
- Robust to Overfitting
- Can Handle Missing Values

4.3.3 Limitations

- Slower Predictions
- Less Interpretable
- Bias Toward Features with More Levels

4.4 Decision Trees

4.4.1 About the model

A Decision Tree is a supervised learning model used for both classification and regression.

It works by splitting the data into subsets based on feature values, asking "yes/no" questions at each node until it reaches a final decision (a leaf node).

4.4.2 Advantages

- Easy to understand and visualize
- Can handle both numerical and categorical data
- Works for classification and regression
- Handles missing values and irrelevant features fairly well

4.4.3 Limitations

- Can easily overfit if not pruned or regularized
- Small changes in data $\rightarrow$ big changes in the tree (high variance)
- Often less accurate than ensemble methods (like Random Forest or XGBoost)

4.5 Neural Networks

4.5.1 About the model

A Neural Network is a computational model inspired by how the human brain works. It's made up of layers of interconnected neurons (also called nodes or units) that learn to map inputs to outputs. They're especially powerful for recognizing patterns in complex and high-dimensional data.

Input layer takes in the features, the hidden layer processes inputs using weights, biases and activation function and the output layer gives the final prediction.

4.5.2 Advantages

- Can learn complex patterns
- Scales well with big data
- Powers modern AI applications like ChatGPT, Alexa, and Netflix recommendations

4.5.3 Limitations

- Computationally expensive (needs GPUs/TPUs)
- Takes time to train
- Hard to interpret (black box models)

4.6 SVM

4.6.1 About the model

SVM is a supervised learning algorithm used for classification and sometimes regression. It is used to find the best hyperplane that separates classes in the data with the maximum margin. We used the RBF kernel:

RBF (Radial Basis Function) or Gaussian Kernel: Most commonly used, handles non-linear relationships.

4.6.2 Advantages

- High accuracy
- Works well in high-dimensional spaces (e.g. text classification)
- Effective when margin of separation is clear

4.6.3 Limitations

- Slow on large datasets
- Harder to tune (kernel, C, gamma... all need proper selection)
- Not great for datasets with a lot of noise or overlapping classes

4.7 KNN

4.7.1 About the model

KNN is a lazy, instance-based, and non-parametric learning algorithm used for classification and regression. It predicts the output based on the K closest points in the training dataset.

4.7.2 Advantages

- Very intuitive and simple
- No training required (lazy learning)
- Naturally handles multi-class problems

4.7.3 Limitations

- Slow prediction time (since it compares to every point in the dataset)
- Sensitive to irrelevant features or scaling (standardization is usually required)
- Poor performance on imbalanced data

Classification Report:

	precision	recall	f1-score	support
0	0.98	0.93	0.96	208
1	0.50	0.59	0.54	458
2	0.68	0.78	0.73	424
3	0.98	0.90	0.94	134
4	0.82	0.95	0.88	129
5	0.72	0.76	0.74	483
6	0.59	0.54	0.56	863
7	0.99	0.99	0.99	192
8	0.72	0.83	0.77	618
9	0.70	0.75	0.73	841
10	0.82	0.70	0.75	1649
accuracy			0.73	5999
macro avg	0.77	0.79	0.78	5999
weighted avg	0.73	0.73	0.73	5999

Figure 4.1: Classification Report - Logistic Regression

4.8 Perceptron with Polynomial Features

4.8.1 About the model

The Perceptron is the simplest type of neural network, proposed by Frank Rosenblatt in 1958. It's a binary classifier that maps input features to an output using a linear decision boundary.

If we expand the feature space using polynomial features, we can make non-linear data linearly separable in that higher-dimensional space.

4.8.2 Advantages

- Lets simple models handle complex decision boundaries
- Easy to implement
- Works well on small, clean datasets

4.8.3 Limitations

- Can lead to overfitting (too many features)
- Becomes computationally expensive for high degrees / high dimensions

5 Conclusion

According to figure 5.1 (at the last page), we see that the Random Forest Classifier gave the highest accuracy amongst all the models used to train the data.

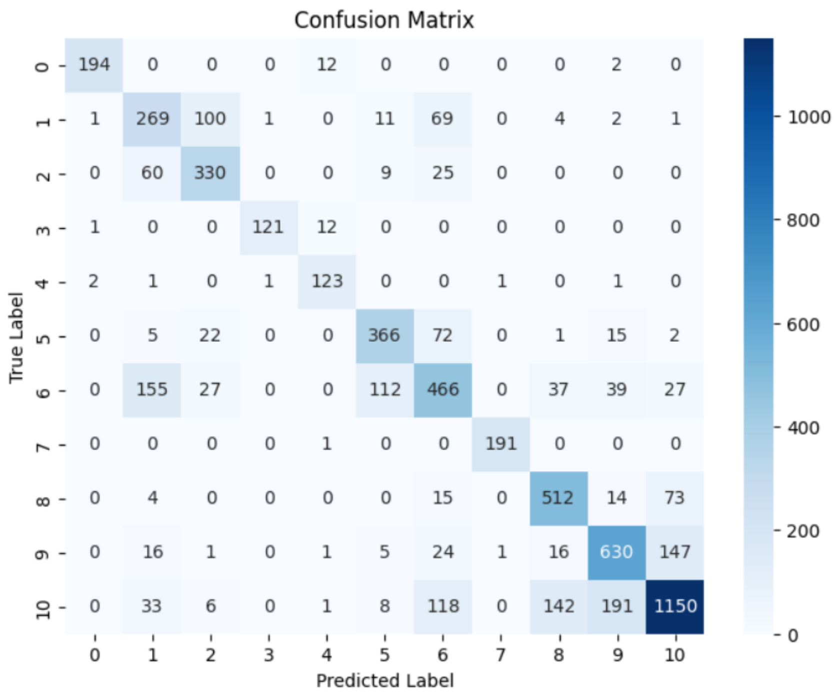

Figure 4.2: Confusion Matrix - Logistic Regression

```
===== General Bayes Classifier =====

--- Fold 1 ---
Fold 1 | Validation Accuracy: 0.6976

--- Fold 2 ---
Fold 2 | Validation Accuracy: 0.6849

--- Fold 3 ---
Fold 3 | Validation Accuracy: 0.6854

Best Validation Accuracy: 0.6976

Classification Report:
              precision    recall  f1-score   support

           0       0.95      0.94      0.94       208
           1       0.49      0.52      0.51       458
           2       0.66      0.79      0.72       424
           3       0.99      0.93      0.96       134
           4       0.89      0.91      0.90       129
           5       0.65      0.81      0.72       483
           6       0.62      0.54      0.57       863
           7       1.00      0.97      0.98       192
           8       0.70      0.84      0.76       618
           9       0.59      0.81      0.68       841
          10       0.84      0.57      0.68      1649

    accuracy                           0.70      5999
   macro avg       0.76      0.78      0.77      5999
weighted avg       0.72      0.70      0.70      5999
```

Figure 4.3: Classification Report - Bayes Classifier

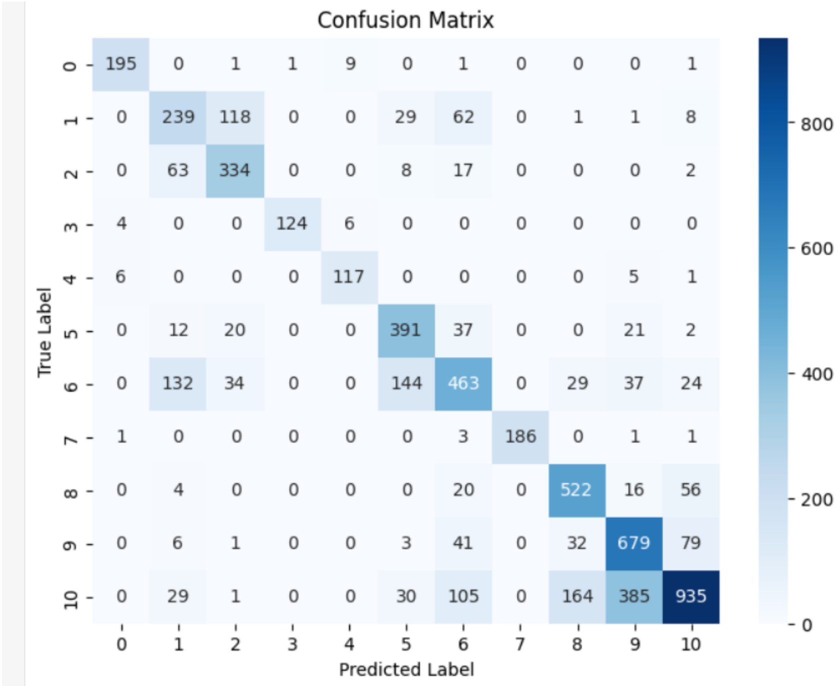

Figure 4.4: Confusion Matrix - Bayes Classifier

```
--- Fold 1 ---
Validation Accuracy for Fold 1: 0.8255

--- Fold 2 ---
Validation Accuracy for Fold 2: 0.8185

--- Fold 3 ---
Validation Accuracy for Fold 3: 0.8191

Best Validation Accuracy across all folds: 0.8255

Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       208
           1       0.45      0.59      0.51       458
           2       0.92      0.93      0.92       424
           3       1.00      1.00      1.00       134
           4       0.99      0.97      0.98       129
           5       0.96      0.95      0.95       483
           6       0.72      0.68      0.70       863
           7       1.00      1.00      1.00       192
           8       0.84      0.90      0.87       618
           9       0.85      0.87      0.86       841
          10       0.86      0.79      0.82      1649

    accuracy                           0.83      5999
   macro avg       0.87      0.88      0.87      5999
weighted avg       0.83      0.83      0.83      5999
```

Figure 4.5: Classification Report - Random Forest

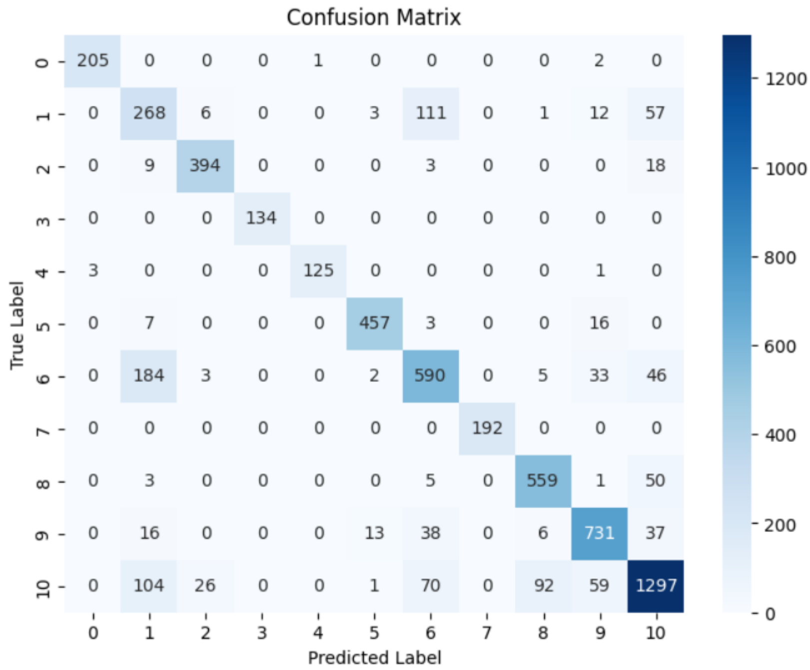

Figure 4.6: Confusion Matrix - Random Forest

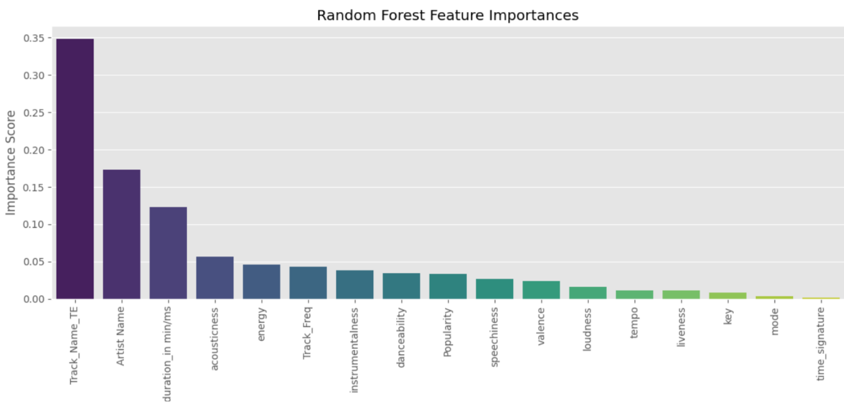

Figure 4.7: Feature Importance - Random Forest

```
Best Validation Accuracy across all folds: 0.8095

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.99      0.97       209
           1       0.50      0.81      0.62       457
           2       0.82      0.83      0.83       424
           3       0.96      0.99      0.97       134
           4       0.90      0.94      0.92       129
           5       0.88      0.89      0.89       482
           6       0.81      0.62      0.70       862
           7       0.96      0.99      0.98       192
           8       0.81      0.89      0.85       618
           9       0.81      0.81      0.81       842
          10       0.88      0.78      0.83      1650

    accuracy                           0.81      5999
   macro avg       0.85      0.87      0.85      5999
weighted avg       0.83      0.81      0.81      5999
```

Figure 4.8: Classification Report - Neural Network

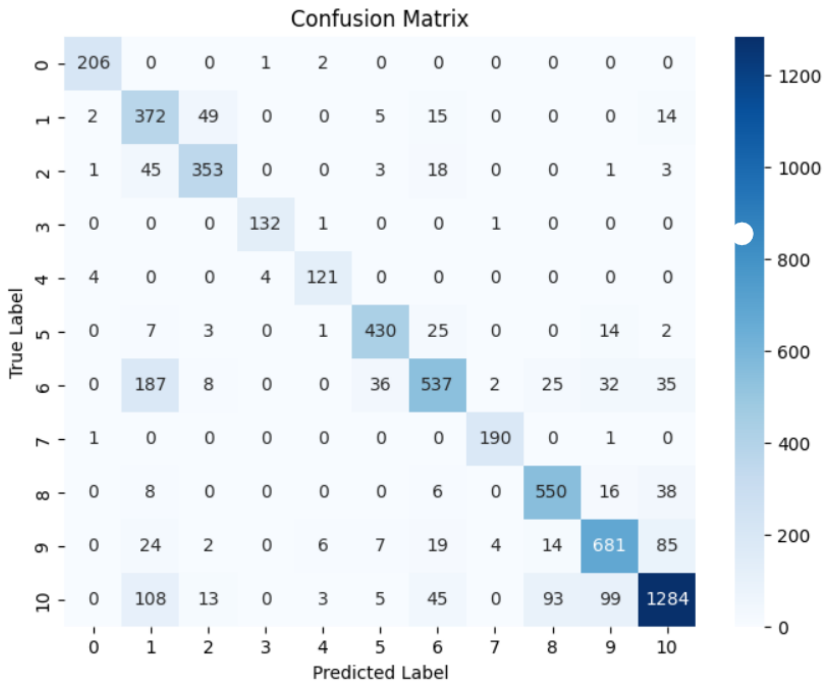

Figure 4.9: Confusion Matrix - Neural Network

```
--- Fold 1 ---
Validation Accuracy for Fold 1: 0.8220

--- Fold 2 ---
Validation Accuracy for Fold 2: 0.8155

--- Fold 3 ---
Validation Accuracy for Fold 3: 0.8126

Best Validation Accuracy across all folds: 0.822

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       208
           1       0.49      0.88      0.63       458
           2       0.90      0.85      0.88       424
           3       1.00      0.98      0.99       134
           4       0.95      0.95      0.95       129
           5       0.87      0.87      0.87       483
           6       0.78      0.64      0.70       863
           7       0.99      0.99      0.99       192
           8       0.80      0.87      0.83       618
           9       0.86      0.83      0.84       841
          10       0.91      0.80      0.85      1649

    accuracy                           0.82      5999
   macro avg       0.87      0.87      0.86      5999
weighted avg       0.84      0.82      0.83      5999
```

Figure 4.10: Classification Report - SVM

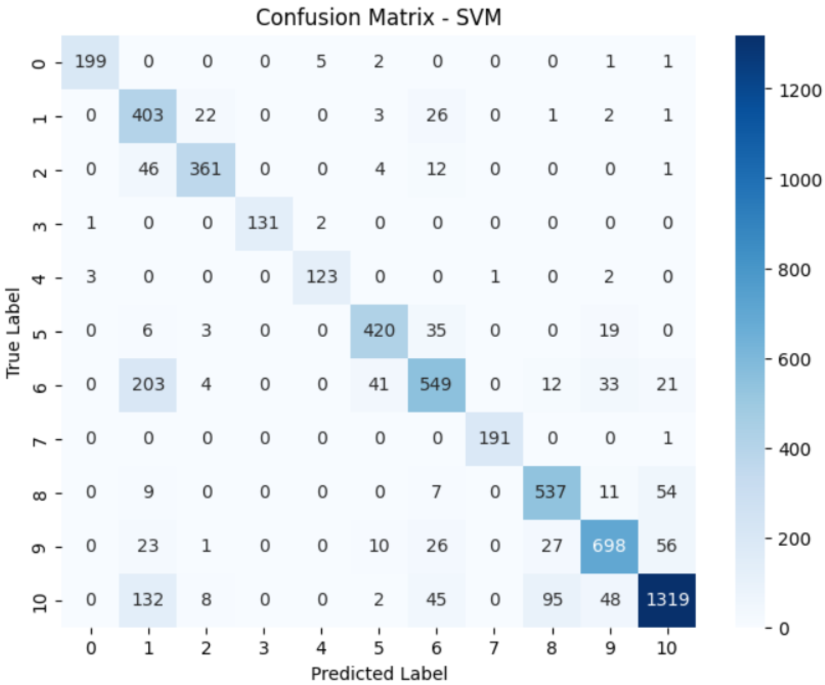

Figure 4.11: Confusion Matrix - SVM

```
KNN Model Evaluation:
Best k: 13
Best Validation Accuracy: 0.6131
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.89      0.90       208
           1       0.37      0.57      0.45       458
           2       0.64      0.75      0.69       424
           3       0.76      0.80      0.78       134
           4       0.72      0.85      0.78       129
           5       0.67      0.79      0.72       483
           6       0.57      0.42      0.48       863
           7       0.88      0.96      0.92       192
           8       0.47      0.84      0.60       618
           9       0.59      0.71      0.65       841
          10       0.85      0.39      0.54      1649

    accuracy                           0.61      5999
   macro avg       0.68      0.73      0.68      5999
weighted avg       0.67      0.61      0.61      5999
```

Figure 4.12: Classification Report - KNN

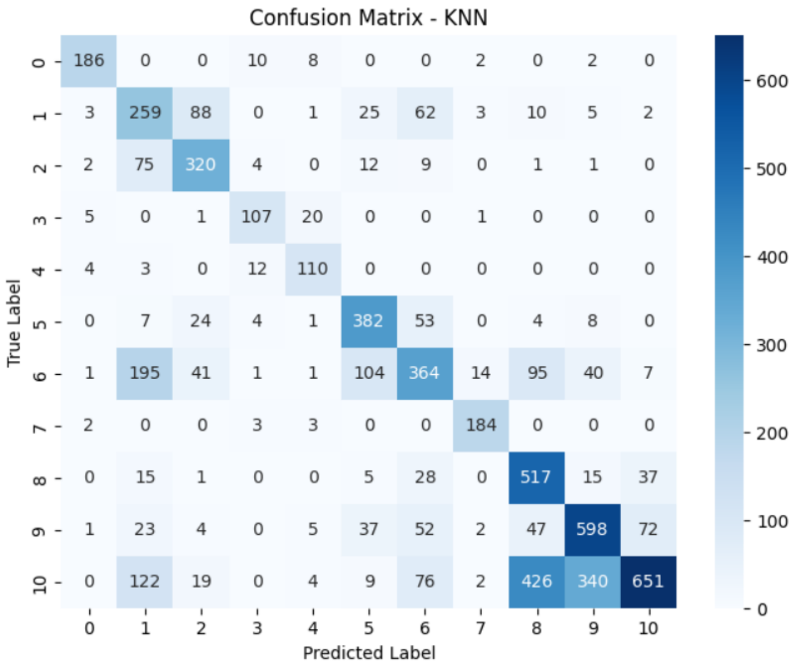

Figure 4.13: Confusion Matrix - KNN

Best Validation Accuracy across all folds: 0.6811

Classification Report:

	precision	recall	f1-score	support
0	0.88	0.91	0.90	208
1	0.42	0.59	0.49	458
2	0.67	0.73	0.70	424
3	0.79	0.80	0.80	134
4	0.74	0.86	0.80	129
5	0.68	0.70	0.69	483
6	0.56	0.49	0.52	863
7	0.94	0.96	0.95	192
8	0.62	0.79	0.70	618
9	0.67	0.69	0.68	841
10	0.84	0.66	0.74	1649
accuracy			0.68	5999
macro avg	0.71	0.74	0.72	5999
weighted avg	0.70	0.68	0.68	5999

Figure 4.14: Classification Report - Perceptron

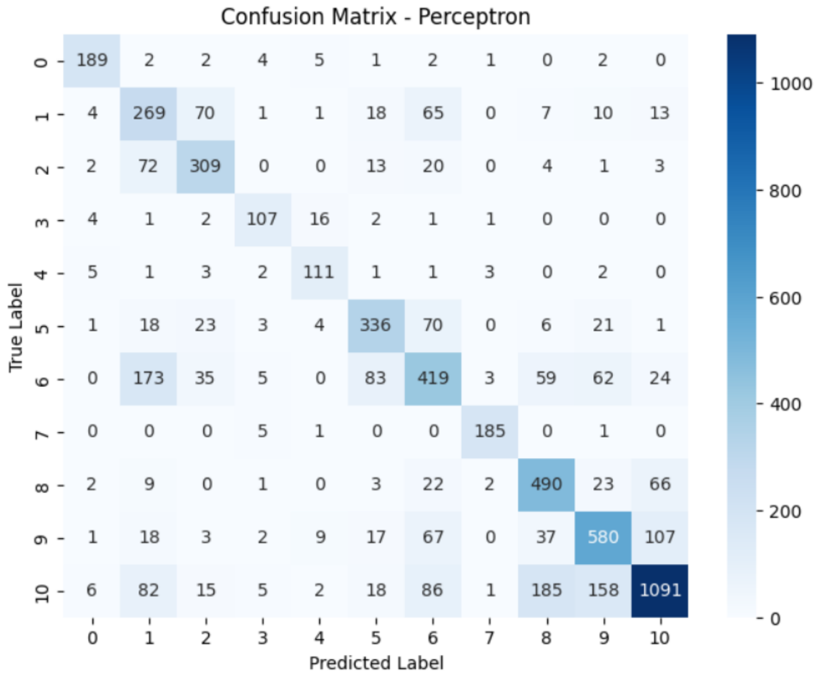

Figure 4.15: Confusion Matrix - Perceptron

```
--- Fold 1 ---
Validation Accuracy for Fold 1: 0.8046

--- Fold 2 ---
Validation Accuracy for Fold 2: 0.7965

--- Fold 3 ---
Validation Accuracy for Fold 3: 0.8003

Best Validation Accuracy across all folds: 0.8046

Classification Report:
              precision    recall  f1-score   support

           0       0.98      0.94      0.96       208
           1       0.43      0.52      0.47       458
           2       0.87      0.91      0.89       424
           3       0.98      0.99      0.99       134
           4       0.98      0.93      0.95       129
           5       0.96      0.92      0.94       483
           6       0.67      0.70      0.69       863
           7       0.95      0.98      0.97       192
           8       0.82      0.88      0.85       618
           9       0.83      0.84      0.84       841
          10       0.86      0.77      0.81      1649

    accuracy                           0.80      5999
   macro avg       0.85      0.85      0.85      5999
weighted avg       0.81      0.80      0.81      5999
```

Figure 4.16: Classification Report - Decision Trees

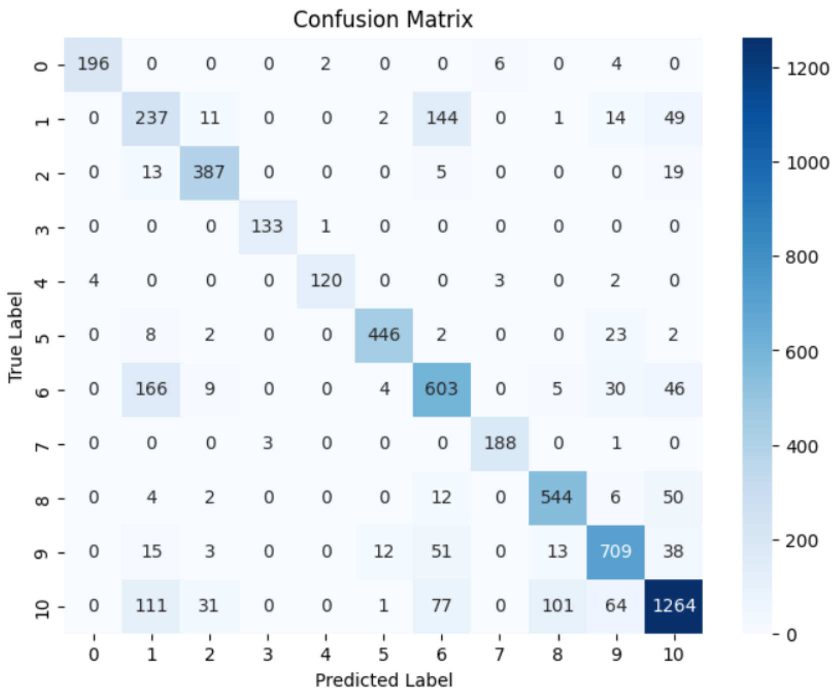

Figure 4.17: Confusion Matrix - Decision Trees

	Model	Best Validation Accuracy	precision	recall	f1_score
1	Random Forest	0.8255	0.8717	0.8781	0.8738
2	SVM Classifier rbf kernel	0.8220	0.8669	0.8744	0.8643
3	NeuralNetwork (sigmoid)	0.8095	0.8453	0.8670	0.8510
4	Decision Tree	0.8046	0.8482	0.8533	0.8498
5	Logistic Regression	0.7255	0.7727	0.7929	0.7805
6	Naive Bayes	0.6956	0.7583	0.7757	0.7605
7	Perceptron PolynomialDeg=3	0.6811	0.7111	0.7430	0.7231
8	KNN Model with k=13	0.6131	0.6752	0.7254	0.6833

Figure 5.1: Final Comparison of all the methods