Pattern Recognition Task Report

Clarance L.G.S.

I. INTRODUCTION

This report is a comprehensive overview of the task which is a continuation of the course ”Pattern Recognition”. The primary objective of the module is to introduce the fundamental concepts and algorithms for machine learning with their applications.

A. *What is Pattern Recognition?*

- Searching for patterns in data is a fundamental task in data analysis, machine learning, and artificial intelligence.
- The goal is to identify regularities, trends, correlations, or any **meaningful structure** within the data that can provide **valuable insights** or be used for predictive modelling. [1]
- Overall, pattern recognition plays a crucial role in enabling machines to learn from data and make informed decisions.
- Applications: Machine Vision, Speech Recognition, Character Recognition, Medical Imaging, Robotics, Speech and Natural Language Processing

II. BACKGROUND

A. *HW1*

- GitHub link
- HW1 focuses primarily on exploring data preprocessing methods within the scikit-learn preprocessing package.
- The main objective involves comparing various feature scaling methods, including:
  - Min-Max Scaling
  - Standard Scaling
  - Robust Scaling
  - Power Transformer
- The analysis delves into the effects of each scaling method on the data distribution and handling of outliers within the data.

B. *HW2*

- GitHub link
- HW2 is the application of the Logistic regression algorithm.
- This presents a detailed discussion on topics, including:
  - Encoding
  - Solver
  - Feature scaling
  - Train and test set split
- Also comparison of some examples within the aforementioned topics.
  - LabelEncoder Vs. OneHotEncoder
  - saga Vs. liblinear
  - MaxAbsScaler Vs. StandardScaler

III. HW3

This is designed as a guided project to solve real-world scenario with step-by-step guidance. Focusing on the multi-class classification problem on an open domain dataset about Spotify songs retrieved through Spotify API. The primary goal is to develop a model which can provide an accurate genre classification. Project Link

The project covers the content of learning from data and related challenges, machine learning algorithms for classification, and neural networks. It is structured as follows:

- Data preprocessing and visualization
- Data cleaning
- Feature engineering
- Data preparation
- Model selection
- Model deployment
- Neural network model

In this project, common libraries such as Pandas, Numpy, Matplotlib, and Seaborn are utilized for data manipulation, computation, and visualization tasks. Primarily, Scikit-Learn [2], a machine learning library, is employed for ML-related implementations, while Keras [3], a deep learning library, is utilized to construct neural network model.

A. *Data preprocessing and visualization*

The dataset undergoes preprocessing to handle null values and duplicate entries, thereby reducing the complexity and redundancy within the dataset. As part of exploratory data analysis, data visualization techniques are employed to identify general patterns within the dataset.

B. *Data cleaning*

To obtain clean data, outliers—data points that significantly deviate from others—are need to be removed from the dataset. The Interquartile Range (IQR) method is utilized for outlier detection, and any detected outliers are subsequently removed.

C. *Feature engineering*

Feature engineering is a pivotal ML technique that involves transforming raw input features into an effective set of features. This process may include removing or linearly combining dependent and correlated variables. A correlation matrix is used to compute the correlation between input features, aiding in the identification of relationships among variables.

D. Data preparation

This step includes encoding, train and test data splitting, and feature scaling, which prepare the dataset for model training. LabelEncoder is chosen for encoding, while MinMaxScaler is utilized for feature scaling. The rationale behind these choices is elaborated in the project.

E. Model selection

Model selection is the process of choosing the best model from a set of candidate models for a specific task. For the multiclass classification problem at hand, the set of candidate algorithms includes:

1) Logistic Regression
2) K-Nearest Neighbors(KNN)
3) Naive Bayes(NB)
4) Support Vector Machine(SVM)
5) Decision Tree
6) Random Forest
7) Extreme Gradient Boosting(XGBoost)
8) Gradient Boosting

Each model is trained using the prepared dataset, and evaluation is based on how well the model generalizes to unseen data. Given that the goal is to develop a model capable of accurate genre classification, the metric of accuracy is used as the criterion for selecting the best model.

Additionally, techniques to enhance the model performance such as hyperparameter tuning also discussed in this section.

F. Model Deployment

Following the evaluation results, the best-performing model is deployed. This deployed model takes user input and predicts the song genre, aligning with the project's ultimate goal.

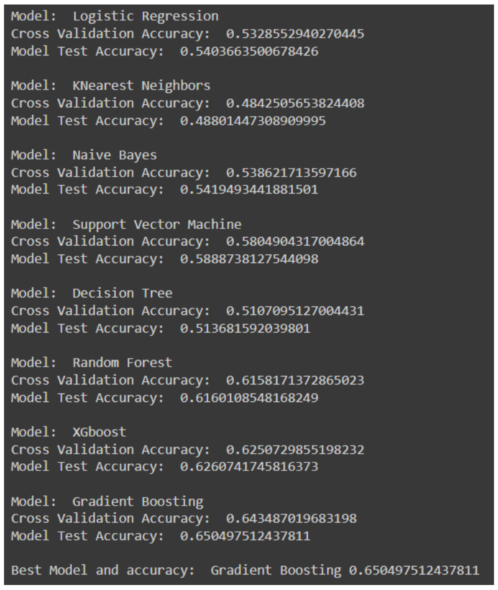
G. Neural network model

In addition to utilizing existing classification algorithms, a neural network using Keras [3] is implemented to perform classification. Layers of the network model are carefully designed to avoid overfitting and stabilize the training of the model.

IV. RESULTS

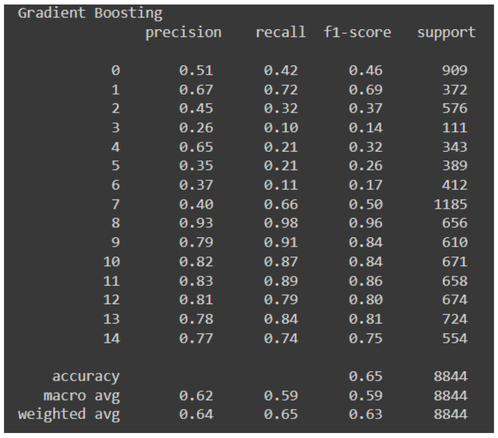
The output of the model selection process is shown in the Fig.1. The result indicates that the best model among algorithm candidates for the song genre classification problem is **Gradient boosting**, achieving an accuracy of 0.651.

Fig. 2 and Fig. 3 are the classification reports of Gradient boosting and Neural network model respectively. These reports offer a detailed breakdown of how effectively both models perform on each class for unseen data.



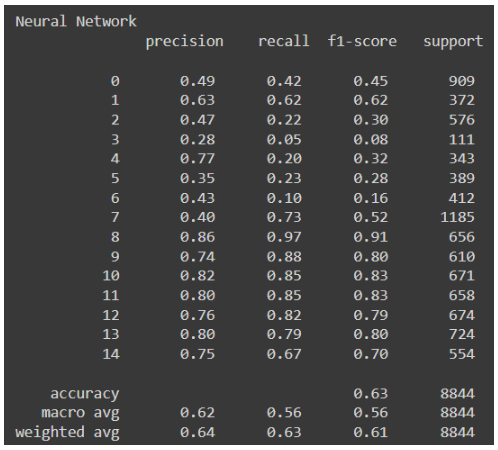
```
Model:  Logistic Regression
Cross Validation Accuracy:  0.5328552940270445
Model Test Accuracy:  0.5403663500678426

Model:  KNearest Neighbors
Cross Validation Accuracy:  0.4842505653824408
Model Test Accuracy:  0.48801447308909995

Model:  Naive Bayes
Cross Validation Accuracy:  0.538621713597166
Model Test Accuracy:  0.5419493441881501

Model:  Support Vector Machine
Cross Validation Accuracy:  0.5804904317004864
Model Test Accuracy:  0.5888738127544098

Model:  Decision Tree
Cross Validation Accuracy:  0.5107095127004431
Model Test Accuracy:  0.513681592039801

Model:  Random Forest
Cross Validation Accuracy:  0.6158171372865023
Model Test Accuracy:  0.6160108548168249

Model:  XGboost
Cross Validation Accuracy:  0.6250729855198232
Model Test Accuracy:  0.6260741745816373

Model:  Gradient Boosting
Cross Validation Accuracy:  0.643487019683198
Model Test Accuracy:  0.650497512437811

Best Model and accuracy:  Gradient Boosting 0.650497512437811
```

Fig. 1. Model Selection Output



```
Gradient Boosting
              precision    recall  f1-score   support

           0       0.51      0.42      0.46       909
           1       0.67      0.72      0.69       372
           2       0.45      0.32      0.37       576
           3       0.26      0.10      0.14       111
           4       0.65      0.21      0.32       343
           5       0.35      0.21      0.26       389
           6       0.37      0.11      0.17       412
           7       0.40      0.66      0.50      1185
           8       0.93      0.98      0.96       656
           9       0.79      0.91      0.84       610
          10       0.82      0.87      0.84       671
          11       0.83      0.89      0.86       658
          12       0.81      0.79      0.80       674
          13       0.78      0.84      0.81       724
          14       0.77      0.74      0.75       554

    accuracy                           0.65      8844
   macro avg       0.62      0.59      0.59      8844
weighted avg       0.64      0.65      0.63      8844
```

Fig. 2. Gradient Boosting Classification Report



```
Neural Network
              precision    recall  f1-score   support

           0       0.49      0.42      0.45       909
           1       0.63      0.62      0.62       372
           2       0.47      0.22      0.30       576
           3       0.28      0.05      0.08       111
           4       0.77      0.20      0.32       343
           5       0.35      0.23      0.28       389
           6       0.43      0.10      0.16       412
           7       0.40      0.73      0.52      1185
           8       0.86      0.97      0.91       656
           9       0.74      0.88      0.80       610
          10       0.82      0.85      0.83       671
          11       0.80      0.85      0.83       658
          12       0.76      0.82      0.79       674
          13       0.80      0.79      0.80       724
          14       0.75      0.67      0.70       554

    accuracy                           0.63      8844
   macro avg       0.62      0.56      0.56      8844
weighted avg       0.64      0.63      0.61      8844
```

Fig. 3. Neural Network Classification Report

V. CONCLUSIONS

In pursuit of a model capable of providing accurate genre classification for a real dataset of Spotify songs, project adhere to the fundamental principles of machine learning techniques. Given the dataset's complexity, the accuracy of the model prediction is reasonable.

However, some F1 scores for predicted classes are not optimal. This discrepancy can be attributed to the class imbalance within the dataset. To address this issue, undersampling of the majority class or generating synthetic samples of the minority class can be employed. To synthesize new samples for minority classes, methods such as SMOTE (Synthetic Minority Oversampling Technique) from the Imbalanced-learn toolbox [4] can be utilized.

REFERENCES

[1] M. T. U. Sampath K. Perera. Motivation and introduction to the course. Lecture slides. EN3150, University of Moratuwa.

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[3] Francois Chollet et al. Keras, 2015.

[4] Guillaume Lemaître, Fernando Nogueira, and Christos K. Aridas. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *Journal of Machine Learning Research*, 18(17):1–5, 2017.