**Big Home Assignment**

Habibur Rahman Habib

16 December, 2023

**Contents:**

1. Introduction
2. Lazy classification with pattern structures
3. Parameter tuning and decision functions
4. Heart Data
5. Diabetes Data
6. Lung Data
7. Conclusion
8. **Introduction**

Lazy Classification using Formal Concept Analysis (FCA) is a powerful technique in machine learning, particularly adept at handling complex datasets with inherent patterns. In this study, we aim to employ Lazy FCA on selected datasets, foregoing the need for data binarization, which simplifies the preprocessing steps while retaining the integrity of the dataset's structures. The primary focus of this investigation is to conduct lazy classification with pattern structures and optimize the decision function parameters using cross-validation procedures.

Datasets: [Heart Disease, Lung Cancer, Diabetes]

Dataset Links:

Heart Disease: <https://www.kaggle.com/datasets/arezaei81/heartcsv>

Diabetere: <https://www.kaggle.com/code/mvanshika/diabetes-prediction/input>

Lung Cancer: <https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer>

All necessary files are available on my github repository:

<https://github.com/habibrahman13/Lazy-Fca>

1. **Lazy classification with pattern structures**

Lazy Classification: It utilizes LazyClassifier from Lazypredict to automatically fit a suite of models to the data and generate predictions.

Output: The code prints the models generated by LazyClassifier and presumably predicts the target variable for the test set.

Confusion Matrix: Prints the confusion matrix for the KNN model.

Learning Curve: Generates a learning curve for the KNN model using learning\_curve from Scikit-learn.

ROC Curve: Plots the ROC curve for the KNN model.

Model Evaluation: Trains a Decision Tree Classifier with the best max\_depth found and evaluates its performance on the test set.

Learning Curve and ROC Curve: Similar to the KNN model, it generates learning and ROC curves for the Decision Tree Classifier.

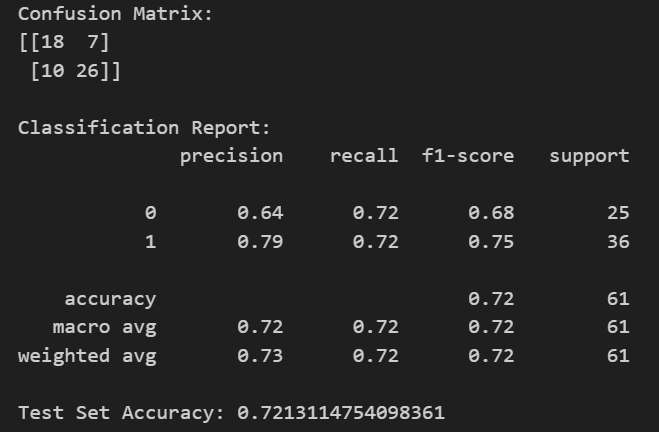
1. **Parameter tuning and decision functions**

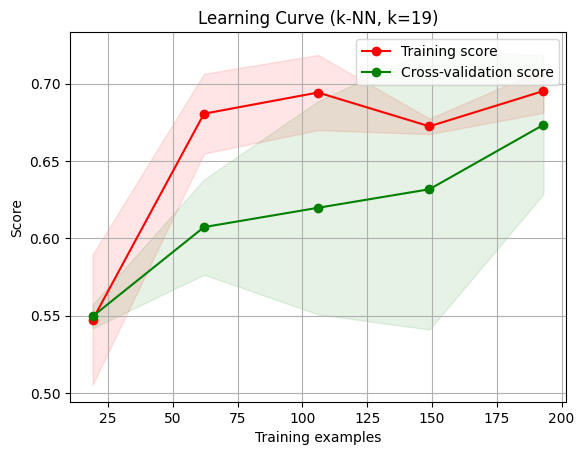
Parameter Tuning: The function tune\_knn\_parameters attempts to tune the k parameter for the K-Nearest Neighbors (KNN) classifier.

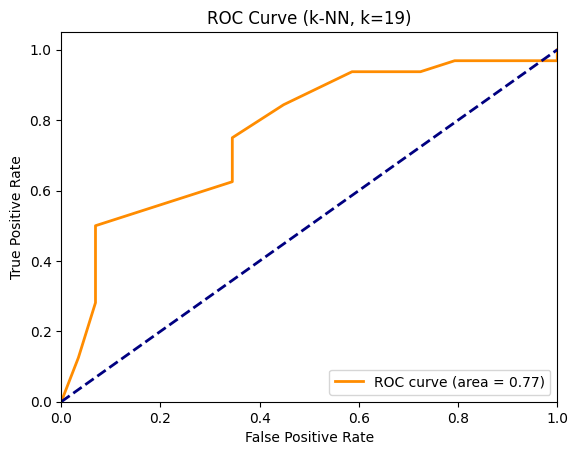
Cross-Validation: It uses cross-validation to evaluate different k values and prints the mean accuracy for each k value.

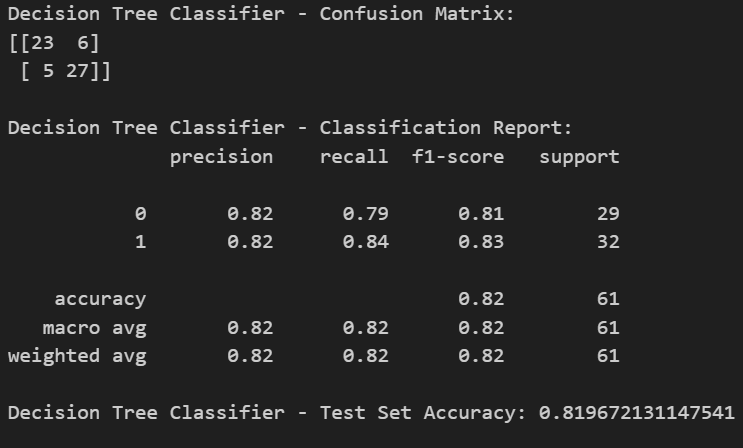
Decision Tree Tuning: This section attempts to find the best max\_depth parameter for a Decision Tree Classifier using cross-validation.

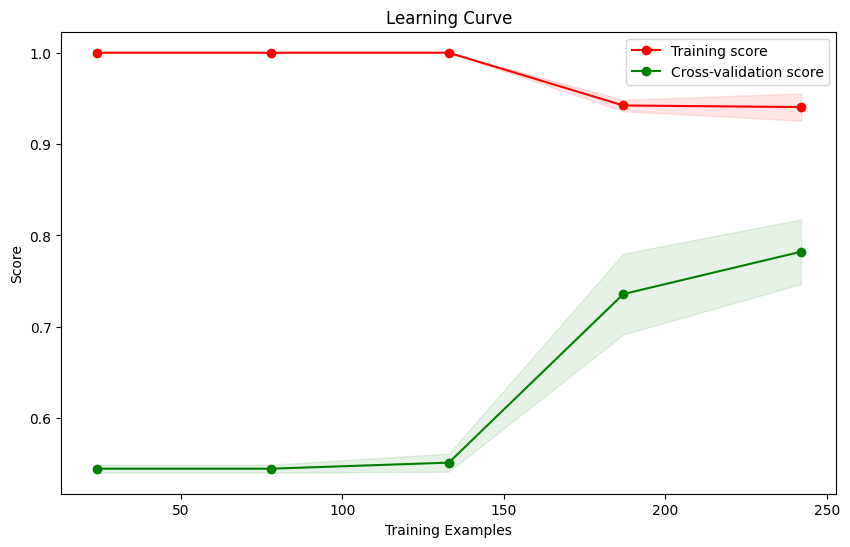
1. **Heart Data**

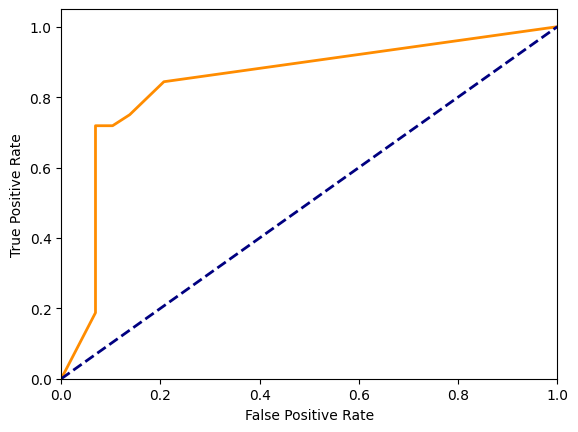


****

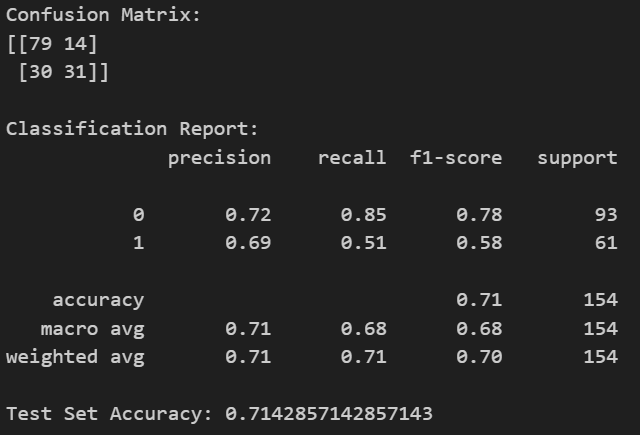
****

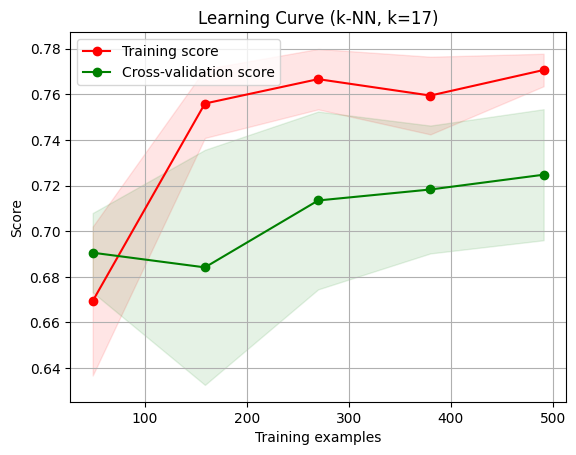


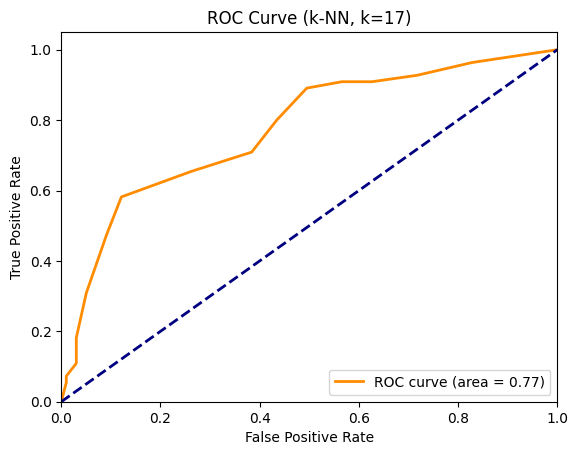
****

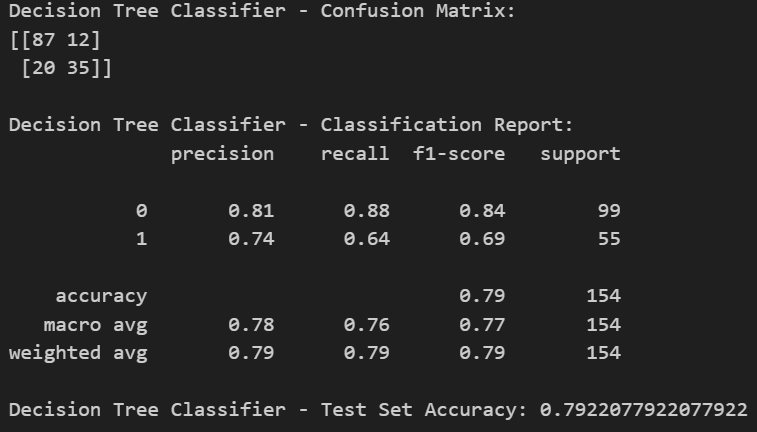
****

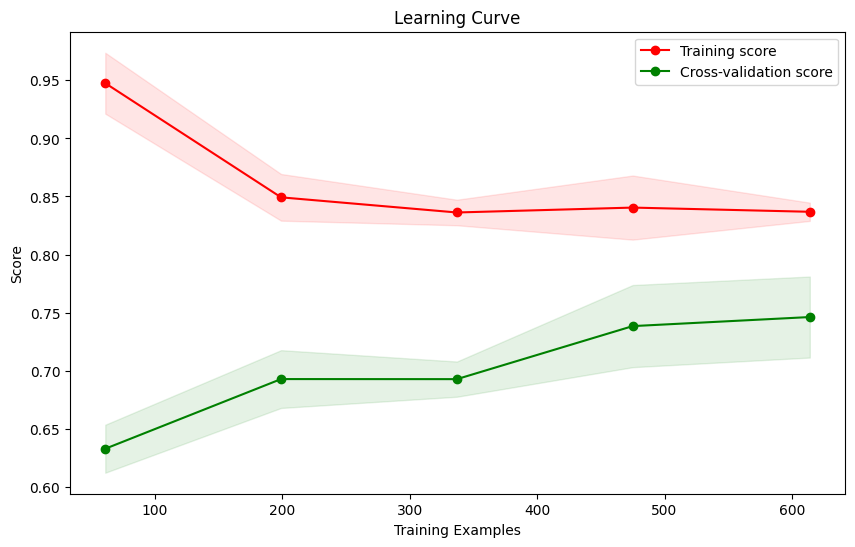
1. **Diabetes Data**

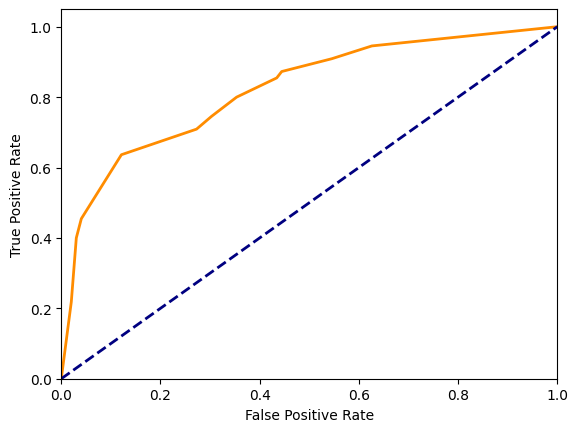


****

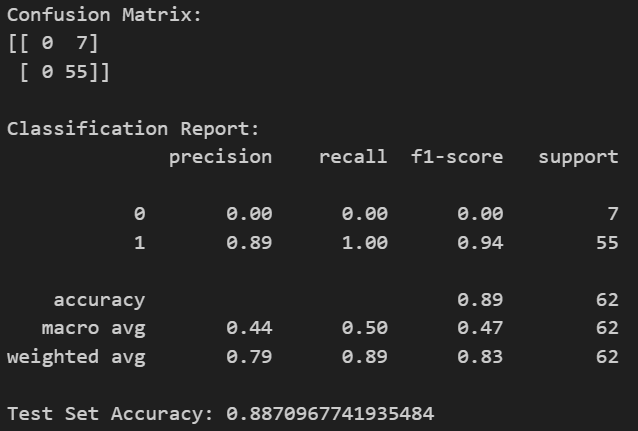
****

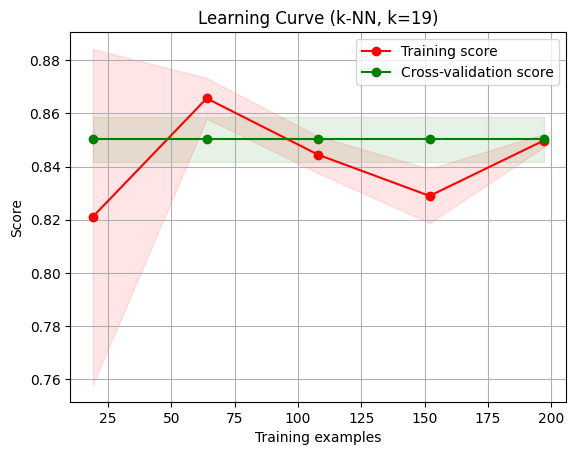


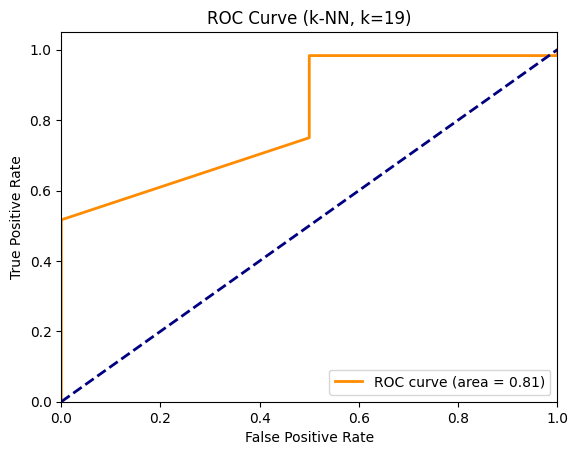
****

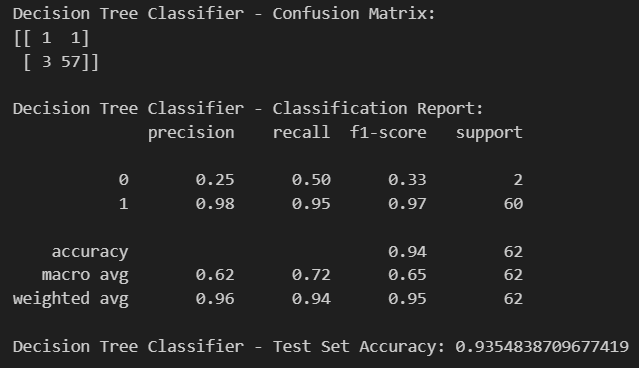
****

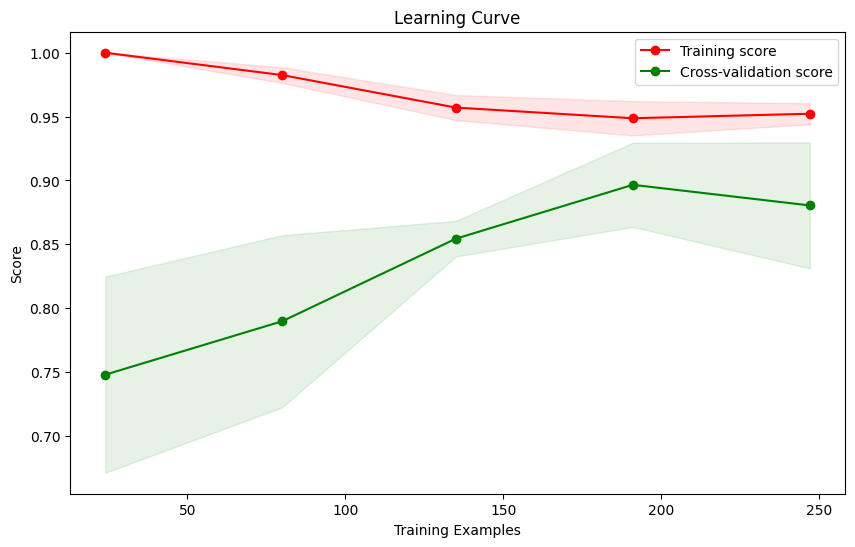
**6 Lung Data**

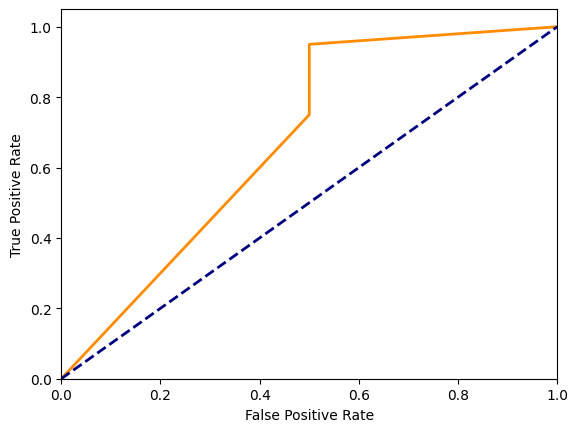


****

****







**Conclusion**

By implementing Lazy FCA and optimizing decision function parameters, we anticipate obtaining refined classification models that accurately predict outcomes within the selected datasets. This approach aims to achieve robust and efficient classification without the computational burden of extensive preprocessing, making it suitable for complex, real-world datasets.