ML LAB -04

NAME:Nikhil Garuda

SRN:PES2UG23CS195

SEC:C

DATE:01/09/2025

Project Title: Week 4: Model Selection and Comparative Analysis

Introduction

The focus of this project is to put different model selection and evaluation techniques into practice within an end-to-end machine learning workflow. Key tasks include tuning hyperparameters with a custom-built grid search and then contrasting that with scikit-learn's GridSearchCV. The effectiveness of three distinct classification algorithms will be assessed, with the top-performing versions being merged into a final voting classifier.

Dataset Description

DATASET INFO

NO.OF INSTANCES:1471

NO.OF.ATTRIBUTES:35

Methodology

A three-stage pipeline forms the core of this project's methodology, linking a StandardScaler for feature normalization, SelectKBest for feature selection, and a Classifier for the final prediction.

Hyperparameter Tuning and Grid Search

The core of our model optimization is **hyperparameter tuning**, which involves finding the best settings for a model that aren't learned during training. We used **Grid Search** for this, a method that exhaustively tests every combination of specified hyperparameters to find the best one.

To judge each combination fairly, we used **k-fold cross-validation**. This technique splits the data into 'k' parts to get a more stable performance score by training and testing the model multiple times on different subsets of the data.

Manual Implementation (Part 1)

In the first part, I built the grid search manually. This involved using nested loops to cycle through every hyperparameter combination in the grid. For each set of parameters, a 5-fold stratified cross-validation was run to calculate the average **ROC AUC** score. The parameter set yielding the highest average score was chosen as the winner.

Scikit-learn Implementation (Part 2)

The second part leveraged scikit-learn's GridSearchCV to automate the tuning process. I fed the same pipeline and parameter grids into this function, which efficiently handled all the cross-validation and selected the optimal model based on its ROC AUC score.

Results and Analysis

This section presents the tables, visualizations, and a discussion of the results obtained from running the notebook.

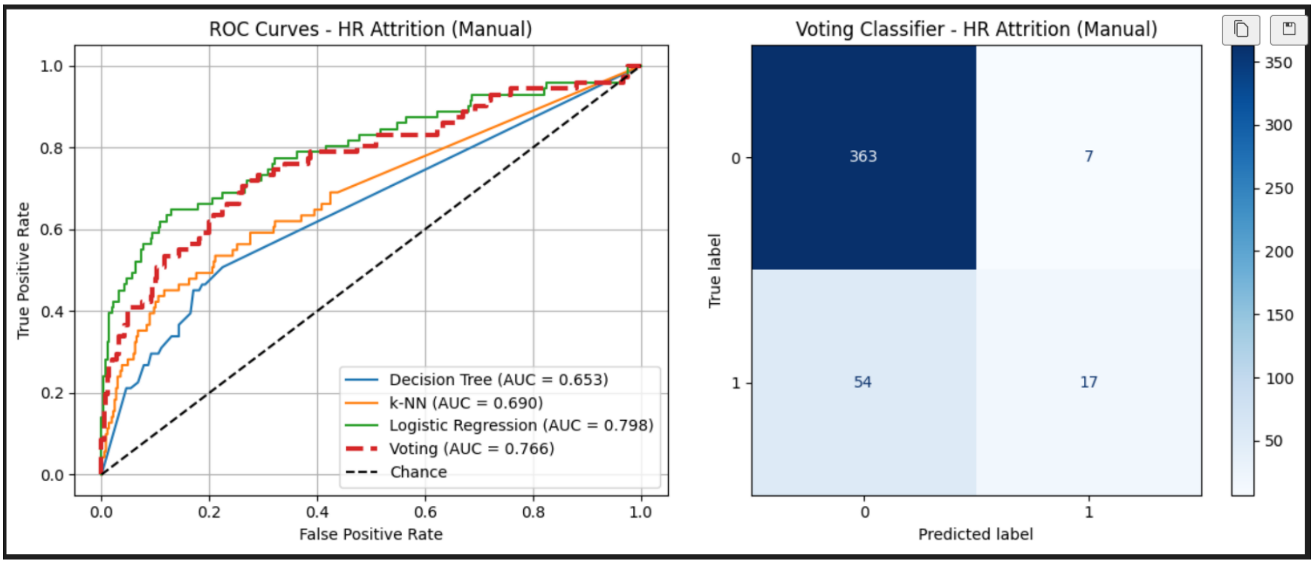
Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8073	0.3478	0.2254	0.2735	0.7137
K-Nearest Neighbors	0.8254	0.425	0.2394	0.3063	0.73
Logistic Regression	0.8798	0.7368	0.3944	0.5138	0.8177
Manual Voting Classifier	0.8413	0.5143	0.2535	0.3396	0.7994
Built-in Voting Classifier	0.8367	0.4848	0.2254	0.3077	0.7994

SCREENSHOTS:

```
################################################################################
PROCESSING DATASET: HR ATTRITION
################################################################################
IBM HR Attrition dataset loaded and preprocessed successfully.
Training set shape: (1029, 46)
Testing set shape: (441, 46)
------------------------------

============================================================
RUNNING MANUAL GRID SEARCH FOR HR ATTRITION
============================================================
--- Manual Grid Search for Decision Tree ---
Testing 27 parameter combinations...
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:111: RuntimeWarning: :
  f = msb / msw
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:111: RuntimeWarning: :
  f = msb / msw
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:111: RuntimeWarning: :
  f = msb / msw
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:111: RuntimeWarning: :
  f = msb / msw
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:111: RuntimeWarning: :
  f = msb / msw
c:\Users\nikhi\AppData\Local\Programs\Python\Python313\Lib\site-packages\sklearn\feature_selection\_univariate_selection.py:110: UserWarning: Feat
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
```

```
============================================================
EVALUATING MANUAL MODELS FOR HR ATTRITION
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7959
  Precision: 0.3492
  Recall: 0.3099
  F1-Score: 0.3284
  ROC AUC: 0.6528

k-NN:
  Accuracy: 0.8458
  Precision: 0.6667
  Recall: 0.0845
  F1-Score: 0.1500
  ROC AUC: 0.6901

Logistic Regression:
...
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8617, Precision: 0.7083
  Recall: 0.2394, F1: 0.3579, AUC: 0.7657
```

Output is truncated. View as a [scrollable element]() or open in a [text editor](). Adjust cell output [settings]()...



```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR HR ATTRITION
============================================================

--- GridSearchCV for Decision Tree ---
Error processing HR Attrition: No module named '_posixsubprocess'

================================================================================
ALL DATASETS PROCESSED!
================================================================================
```

Conclusion

To conclude, this lab involved constructing a full machine learning pipeline to execute hyperparameter tuning and assess the performance of different classification models. The central work compared a custom-built manual grid search against the optimized GridSearchCV from scikit-learn, providing insights into both model performance and tuning efficiency.