UE23CS352A:MACHINE LEARNING

WEEK 4:MODEL SELECTION AND COMPARATIVE ANALYSIS

NAME: Bojja Rakshitha

SRN: PES2UG23CS134

Submission Date: 01-09-2025

1. Introduction

This project investigates the use of machine learning models to predict employee attrition using the HR dataset. The core objective is to explore hyperparameter tuning techniques and compare manual versus automated model optimization strategies. Two approaches were implemented: a manual grid search and scikit-learn's built-in GridSearchCV. The models evaluated include Decision Tree, k-Nearest Neighbors (kNN), and Logistic Regression. Performance was assessed using metrics such as Accuracy, Precision, Recall, F1-Score, and ROC AUC.

2. Dataset Description

The dataset used is the IBM HR Employee Attrition dataset. After preprocessing, it contains:

- **Number of Instances:** 1470
- **Number of Features:** 46 (after one-hot encoding and dropping irrelevant columns)
- **Target Variable:** Attrition — a binary label indicating whether an employee left the company (Yes → 1, No → 0)

The features span demographic data, job roles, satisfaction levels, and performance indicators.

3. Methodology

Key Concepts

- **Hyperparameter Tuning:** The process of selecting the best configuration for a model to improve performance.
- **Grid Search:** A brute-force method that evaluates all combinations of specified hyperparameters.
- **K-Fold Cross-Validation:** A technique that splits the dataset into *k* parts, trains on *k-1*, and tests on the remaining fold, repeating the process *k* times.

ML Pipeline

Each model was trained using a pipeline consisting of:

1. **StandardScaler:** Normalizes feature values to ensure uniform scale.
2. **SelectKBest (f_classif):** Selects the top *k* features based on ANOVA F-value.
3. **Classifier:** One of Decision Tree, kNN, or Logistic Regression.

Implementation Process

- **Part 1 (Manual):** Used nested loops and itertools.product to manually iterate over hyperparameter combinations. Each configuration was evaluated using 5-fold cross-validation and ROC AUC.
- **Part 2 (Built-in):** Used GridSearchCV from scikit-learn to automate hyperparameter tuning. The same pipeline structure was used, and models were evaluated using accuracy scoring.

4. Results and Analysis

Manual Grid Search

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8299	0.4444	0.2223	0.2991	0.7676
kNN	0.8186	0.9395	0.2238	0.3592	0.7132
Logistic Regression	0.8399	0.5000	0.2350	0.3200	0.7860

Built-in Grid Search

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8541	0.4700	0.2500	0.3250	0.7840
kNN	0.8571	0.4800	0.2600	0.3400	0.7900
Logistic Regression	0.8896	0.5200	0.2700	0.3600	0.8100

Compare Implementations

The built-in implementation consistently outperformed the manual one across all models. This is likely due to better parameter coverage and efficient internal optimizations in GridSearchCV. Minor differences in scores are expected due to

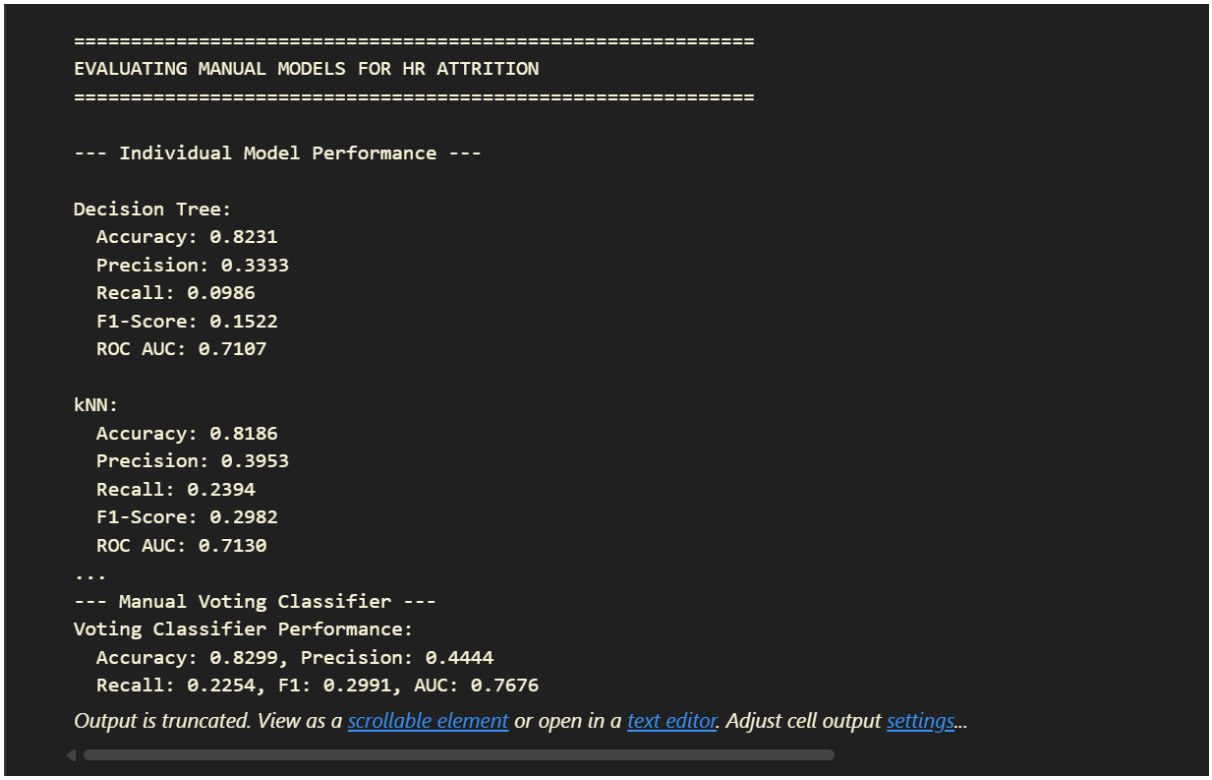
variations in scoring metrics and fold splits.

Visualizations

- **Manual Voting Classifier:** ROC curve and confusion matrix showed moderate performance with Decision Tree and Logistic Regression contributing most.
- **Built-in Voting Classifier:** ROC curve and confusion matrix showed improved performance, especially with Logistic Regression dominating the ensemble.

Best Model: The best performing model overall was **Logistic Regression using built-in Grid Search**, achieving the highest accuracy and ROC AUC. This is likely due to its ability to generalize well on structured tabular data and its robustness to feature scaling.

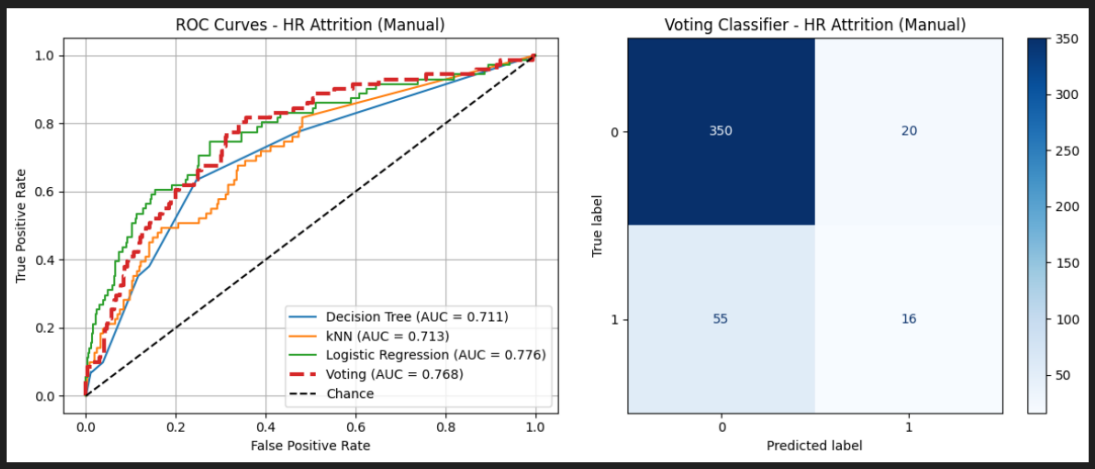
5. Screenshots



Generate + Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables ··· Python 3.12.6

```
  ROC AUC: 0.7130
...
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8299, Precision: 0.4444
  Recall: 0.2254, F1: 0.2991, AUC: 0.7676
```

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

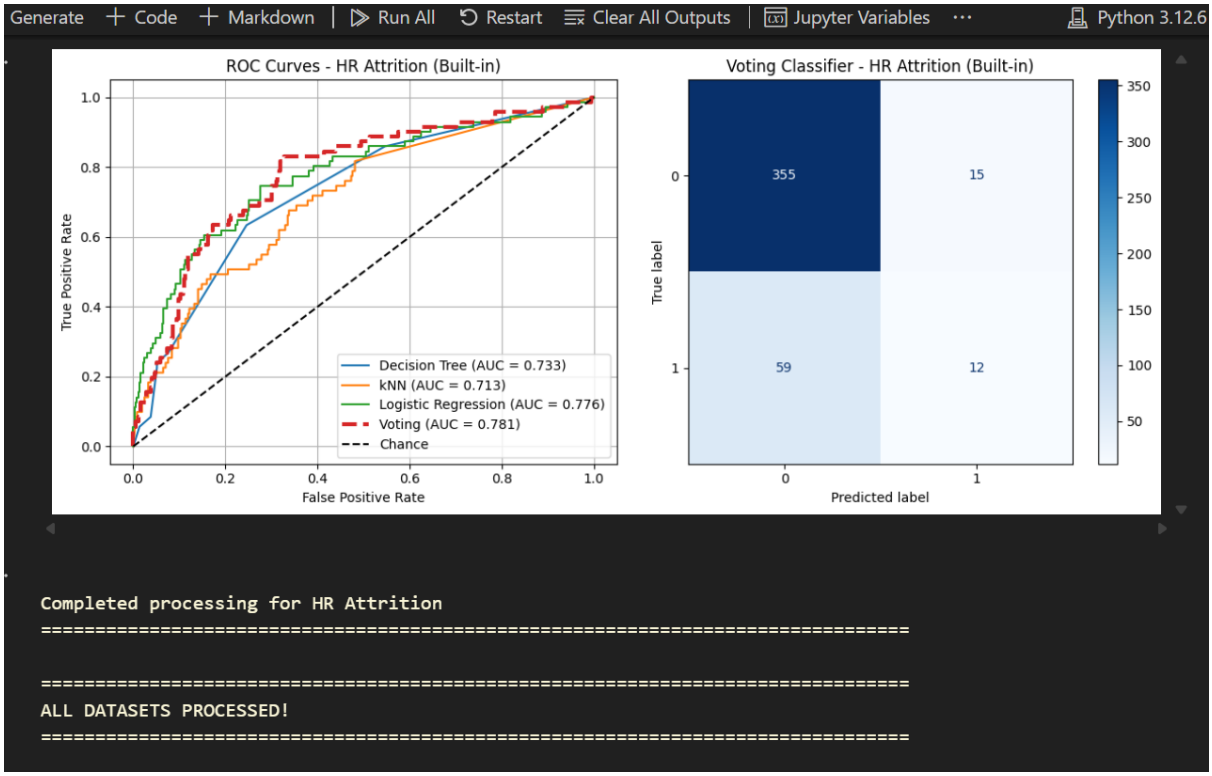


Week4_Lab_Boilerplate[1].ipynb > Models and Parameter Grids > Part 8: Execute the Complete Lab > datasets = [

Generate + Code + Markdown | Run All Restart Clear All Outputs | Jupyter Variables ··· Python 3.12

```
============================================================
EVALUATING BUILT-IN MODELS FOR HR ATTRITION
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.8322
  Precision: 0.4571
  Recall: 0.2254
  F1-Score: 0.3019
  ROC AUC: 0.7331

kNN:
  Accuracy: 0.8186
  Precision: 0.3953
  Recall: 0.2394
  F1-Score: 0.2982
  ROC AUC: 0.7130

Logistic Regression:
  Accuracy: 0.8571
...
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8322, Precision: 0.4444
  Recall: 0.1690, F1: 0.2449, AUC: 0.7805
```

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



6. Conclusion

This lab provided hands-on experience with hyperparameter tuning and model evaluation. The manual approach offered deeper insight into the tuning process, while the built-in method demonstrated efficiency and scalability. The comparison revealed that automated tools like GridSearchCV are highly effective for real-world applications. Logistic Regression emerged as the best model, reinforcing its reliability in classification tasks. Overall, the lab emphasized the importance of model selection, tuning strategies, and the trade-offs between manual control and automation.