MACHINE LEARNING WEEK 4 LAB

PROJECT TITLE: Week 4 Model Selection and Comparative Analysis
Name: DHRUV HEGDE
Student ID: PES2UG23CS172
Course Name: MACHINE LEARNING
Submission Date: 1st SEPTEMBER 2025

1. Introduction

Purpose of the project is as follows:

- Understand hyperparameter tuning through grid search
- Compare manual implementation with built-in functions
- Learn to create and evaluate voting classifiers
- Work with multiple real-world datasets
- Visualize model performance using ROC curves and confusion matrices

Tasks performed:

- Implement manual grid search for hyperparameter tuning.
- Built-in Grid Search Implementation

2. Dataset Description

The dataset is used to predict **employee attrition (churn)** using employee demographics, job-related features, and satisfaction metrics.

Number of Instances (Rows): 1,470 employee

Number of Features (Columns): 35 features (26 numeric, 9 categorical)
Missing Values: None (clean dataset)

Target variable

Attrition

"Yes" → Employee left the company

"No" → Employee stayed in the company

3. Methodology

Hyperparameter Tuning: Systematically searching for the best model settings (e.g., max_depth for trees, C for logistic regression) to maximize validation AUC.

Grid Search: Evaluate all combinations from a predefined hyperparameter grid.

K-Fold Stratified Cross-Validation (K=5): Split data into 5 folds keeping class balance; train on 4 folds and validate on the remaining one; average the metric (AUC).

ML pipeline used

1. **StandardScaler** – z-score scaling of features.
2. **SelectKBest(f_classif)** – univariate feature selection; k tuned (5, 10, or 'all').
3. **Classifier** – one of:
    - Decision Tree (max_depth, min_samples_split, min_samples_leaf)
    - K-Nearest Neighbors (n_neighbors, weights, p)
    - Logistic Regression (C, solver=liblinear)

Process followed

- **Part 1 (Manual):**
  For each model, build the pipeline → iterate over all hyperparameter combos → 5-fold stratified CV → compute AUC per fold → keep the

combo with the highest mean AUC → refit on full training set.

- **Part 2 (scikit-learn):**
  Use GridSearchCV with the same pipeline, parameter grids, scoring='roc_auc', and StratifiedKFold(n_splits=5). Retrieve best_params_, best_estimator_, and best_score_, then refit on full training data.

4. Results and Analysis

Classifier	Mean CV AUC
Decision Tree	0.7217
K-Nearest Neighbors	0.7226
Logistic Regression	0.8328

Test performance -Manual Implementation

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8050	0.3077	0.1690	0.2182	0.7036
KNN	0.8186	0.3784	0.1972	0.2593	0.7236
LR	0.8798	0.7368	0.3944	0.5138	0.8177

Manual Voting Classifier:

Accuracy 0.8345, Precision 0.4643, Recall 0.1831, F1 0.2626, AUC 0.7959.

Test performance- Scikit-learn (GridSearchCV)

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8050	0.3077	0.1690	0.2182	0.7036
KNN	0.8186	0.3784	0.1972	0.2593	0.7236
LR	0.8798	0.7368	0.3944	0.5138	0.8177

Built-in Voting Classifier: Accuracy 0.8390, Precision 0.5000, Recall 0.2394, F1 0.3238, AUC 0.7959.

- **Manual vs GridSearchCV:** Results match (within rounding), indicating your manual search and scikit-learn pipeline are aligned (same CV strategy, grids, and scoring).

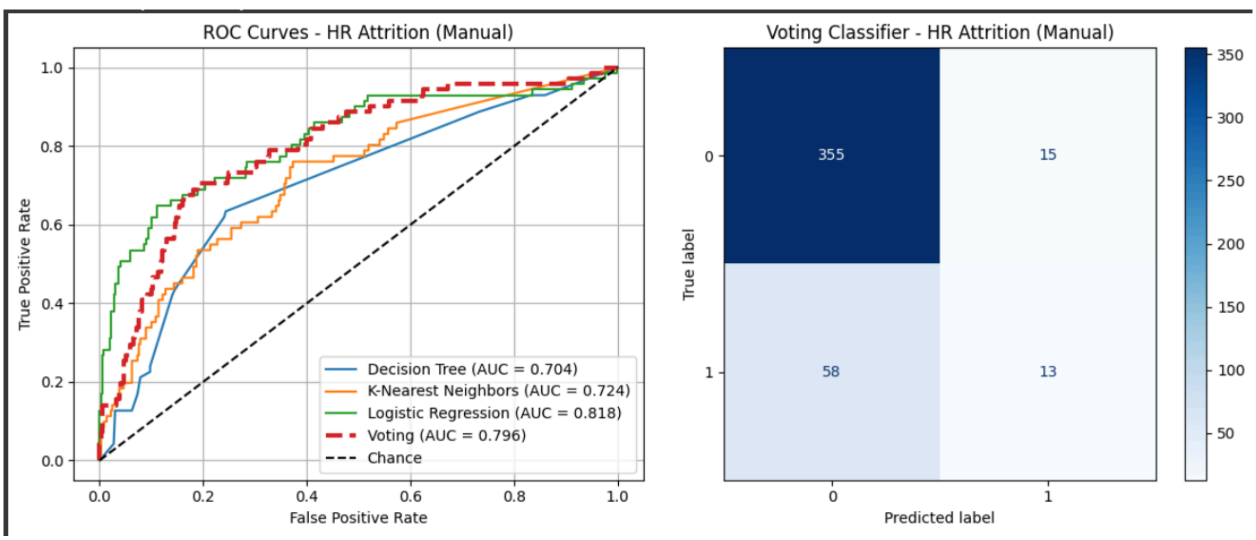
Best model

- **Logistic Regression** is the clear winner on this dataset (highest AUC **0.8177**, highest accuracy **0.8798**, and strongest precision/F1 among single models). With standardized features and a linear decision boundary, LR benefits from the `liblinear` solver and a modest regularization strength (`C=0.1`), which likely balances bias–variance well on this tabular dataset.

5. Screenshots

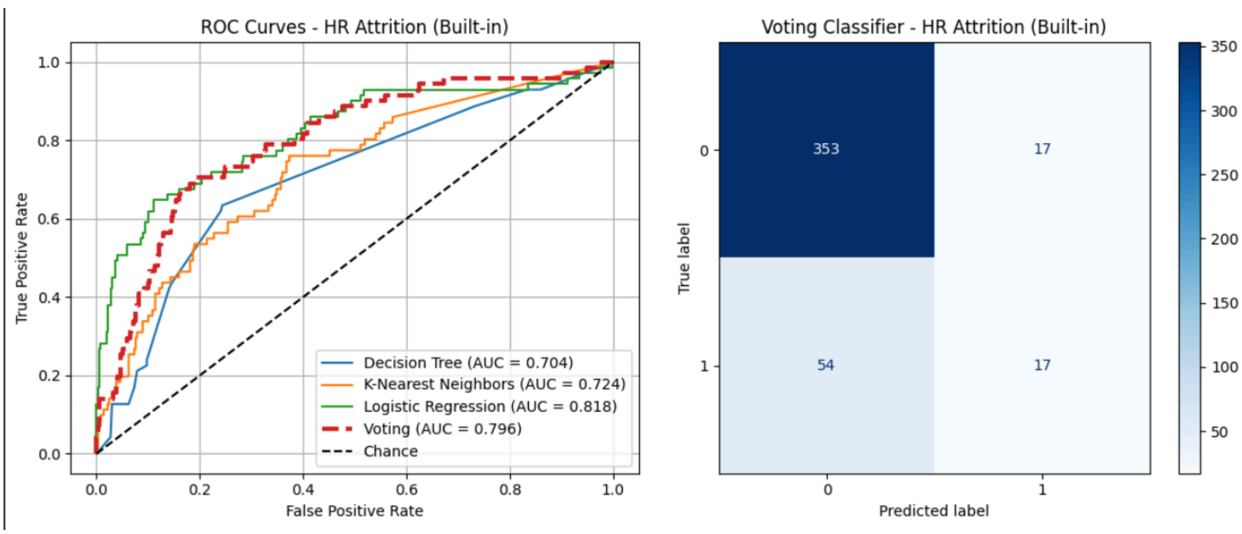
1) MANUAL

```
  f = msb / msw
-------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 'all', 'classifier__C': 0.1}
Best cross-validation AUC: 0.8328

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.8050
  Precision: 0.3077
  Recall: 0.1690
  F1-Score: 0.2182
  ROC AUC: 0.7036

K-Nearest Neighbors:
  Accuracy: 0.8186
  Precision: 0.3784
  Recall: 0.1972
  F1-Score: 0.2593
  ROC AUC: 0.7236

Logistic Regression:
  Accuracy: 0.8798
  Precision: 0.7368
  Recall: 0.3944
  F1-Score: 0.5138
  ROC AUC: 0.8177

--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8345, Precision: 0.4643
  Recall: 0.1831, F1: 0.2626, AUC: 0.7959
```



2)Built in

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.8050
  Precision: 0.3077
  Recall: 0.1690
  F1-Score: 0.2182
  ROC AUC: 0.7036

K-Nearest Neighbors:
  Accuracy: 0.8186
  Precision: 0.3784
  Recall: 0.1972
  F1-Score: 0.2593
  ROC AUC: 0.7236

Logistic Regression:
  Accuracy: 0.8798
  Precision: 0.7368
  Recall: 0.3944
  F1-Score: 0.5138
  ROC AUC: 0.8177

--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8390, Precision: 0.5000
  Recall: 0.2394, F1: 0.3238, AUC: 0.7959
```



6) <u>Conclusion</u>

Key findings

- Logistic Regression with C=0.1 and all features selected delivered the best performance on IBM HR Attrition.

- Tree-based and KNN models trailed LR in ROC AUC and F1.

Takeaways

- Pipelines (scaling → selection → classifier) make experiments reproducible and reduce leakage.

- Grid search with stratified CV gives robust, comparable model selection across classifiers.

- A careful manual implementation can replicate scikit-learn's behavior when the CV splitter, parameter grids, and scoring are identical.

Trade-offs:

- Manual search = full control and transparency, but more code and room for subtle bugs.

- GridSearchCV = concise, reliable, easier to extend (parallelism, cv_results_, callbacks), and less error-prone.