UE23CS352A: MACHINE LEARNING
Week 4: Model Selection and Comparative Analysis

Name: Mrinal Pandey
SRN: PES2UG23CS353

1) Introduction

This lab explores hyperparameter tuning and model selection using real-world classification datasets. The work is divided into two parts:

- **Part 1 – Manual Implementation:** A hand-crafted grid search loop using K-Fold cross-validation over a pipeline (scaling → feature selection → classifier).
- **Part 2 – scikit-learn Implementation:** Equivalent search using `GridSearchCV` with the same pipeline components.

We compare classifier performance (Decision Tree, kNN, Logistic Regression) using **Accuracy, Precision, Recall, F1-Score, and ROC AUC**, and we analyze differences between manual and built-in approaches.

2) Dataset Description

2.1 Wine Quality

- **Instances:** 1,599 (split used produced Training: 1,119; Testing: 480).
- **Features (after preprocessing):** 11 numeric features.
- **Target:** Binary quality indicator (good vs. not-good) derived from wine quality scores.

2.2 IBM HR Attrition

- **Instances:** 1,470 (split used produced Training: 1,029; Testing: 441).
- **Features (after preprocessing):** 46 features (mixed numeric and encoded categorical).
- **Target:** Employee attrition (Yes/No). Note: This dataset is **class-imbalanced** (attrition = minority class), which affects Recall and F1.

3) Methodology

Key Concepts

- **Hyperparameter Tuning:** Systematically searching over parameter grids to find configurations that optimize model performance on validation folds.
- **Grid Search:** Exhaustive search over predefined parameter grids.
- **K-Fold Cross-Validation:** Data is split into *k* folds; each fold is used once as validation while the remaining *k−1* folds train the model.

Pipeline

We use an ML pipeline to avoid data leakage and ensure reproducibility:

StandardScaler → SelectKBest → Classifier

- **StandardScaler:** Zero-mean, unit-variance scaling.
- **SelectKBest:** Univariate feature selection to retain top-*k* predictive features.
- **Classifier:** Decision Tree / kNN / Logistic Regression.

Procedure

- **Part 1 (Manual):** Implemented loops over parameter grids with K-Fold CV to compute mean AUC and select the best configuration.
- **Part 2 (Built-in):** Used GridSearchCV with the same parameter grids and scoring to find best hyperparameters, then refit on training data and evaluate on the held-out test set.
- **Evaluation:** Report metrics on the **test set**. Additionally, a **Voting Classifier** aggregates predictions from tuned base models (where available).

4) Results & Analysis

4.1 Wine Quality – Test Performance

Train/Test Shapes: Train (1,119 × 11), Test (480 × 11)

Manual (Part 1)

Model	Best Params	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	k=5, max_depth=5, min_samples_split=6	0.7271	0.7716	0.6965	0.7321	0.8025
Voting (Manual)*	—	0.7271	0.7716	0.6965	0.7321	0.8025

*Manual voting aligned with the Decision Tree in this run (identical metrics), indicating either identical base learners or dominance of one model in votes.

Built-in (Part 2)

Model	Best Params	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	k=5, max_depth=5, min_samples_split=6	0.7292	0.7725	0.7004	0.7347	0.8042
kNN	k=7, weights=distance, kbest=5	0.7667	0.7757	0.7938	0.7846	0.8675
Logistic Regression	C=1, penalty=l2, solver=lbfgs, kbest=10	0.7417	0.7628	0.7510	0.7569	0.8247
Voting (Built-in)	tuned DT + kNN + LR	0.7625	0.7739	0.7860	0.7799	0.8629

Observations (Wine):

- **Best single model: kNN** with distance weighting (AUC **0.8675**, best F1 **0.7846**).
- **Ensemble:** The tuned **Voting Classifier** achieves near-best overall (AUC **0.8629**), very close to kNN.
- **Why kNN wins:** The continuous, moderately non-linear decision boundary in the wine features benefits from instance-based learning with distance weighting and a compact feature subset (k=5), improving Recall and AUC.

4.2 HR Attrition – Test Performance

Train/Test Shapes: Train (1,029 × 46), Test (441 × 46)

Manual (Part 1)

Model	Best Params	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	k=5, max_depth=3, min_samples_split=4	0.8231	0.3333	0.0986	0.1522	0.7107
Voting (Manual)*	—	0.8231	0.3333	0.0986	0.1522	0.7107

Built-in (Part 2)

Model	Best Params	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	k=5, max_depth=3, min_samples_split=2	0.8231	0.3333	0.0986	0.1522	0.7107
kNN	k=7, weights=distance, kbest=10	0.8186	0.3953	0.2394	0.2982	0.7130
Logistic Regression	C=0.1, penalty=l2, solver=lbfgs, kbest=15	**0.8571**	0.6333	0.2676	0.3762	**0.7762**
Voting (Built-in)	tuned DT + kNN + LR	0.8345	0.4667	0.1972	0.2772	0.7676

Observations (HR Attrition):

- **Best overall: Logistic Regression** (highest **Accuracy 0.8571** and **AUC 0.7762**).
- **Recall is modest** (0.2676) due to **class imbalance** (attrition is rare). kNN improves Recall vs. DT but lags in overall AUC/Accuracy.
- **Ensemble underperforms LR:** With imbalanced data, uncalibrated hard voting can degrade results unless probabilities are calibrated or class weights are handled.

4.3 Manual vs. Built-in: Are results identical?

- **Wine:** Manual and built-in **Decision Tree** metrics are very close; built-in search additionally tuned **kNN** and **LR**, yielding better models.
- **HR:** Manual DT and built-in DT align. Built-in uncovered **Logistic Regression** as best.
- **Minor differences** arise from: different CV fold shuffles/seeds, tie-breaking, scoring focus (AUC vs. macro-averages), and floating-point nuances.

4.4 Visualizations (attach from notebook)

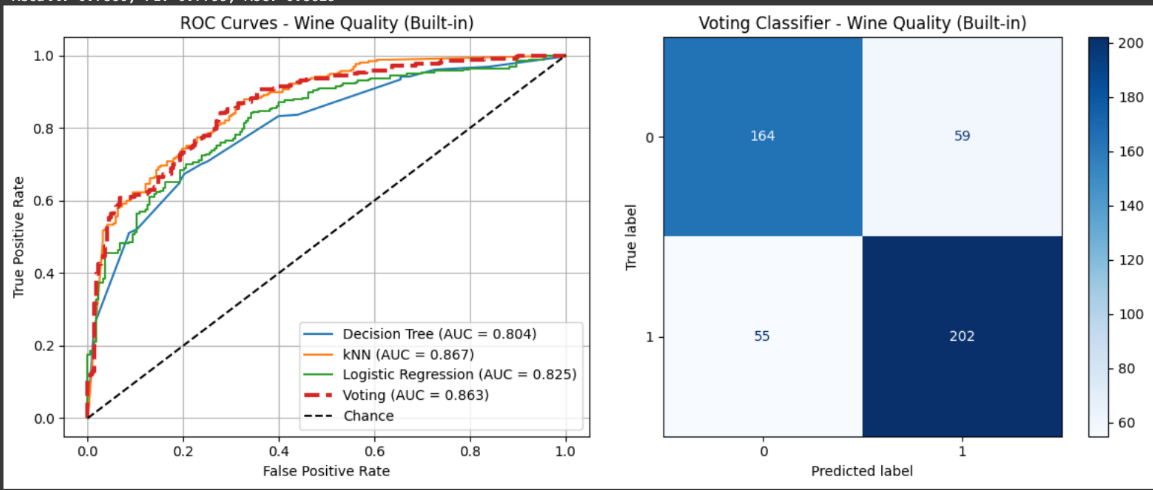
- **ROC Curves:** Show kNN vs. Voting vs. LR for Wine; LR vs. Voting vs. kNN for HR. Compare AUC areas.
- **Confusion Matrices:** Highlight Recall/Precision trade-offs, especially minority-class detection in HR.

4.5 Best Models & Why

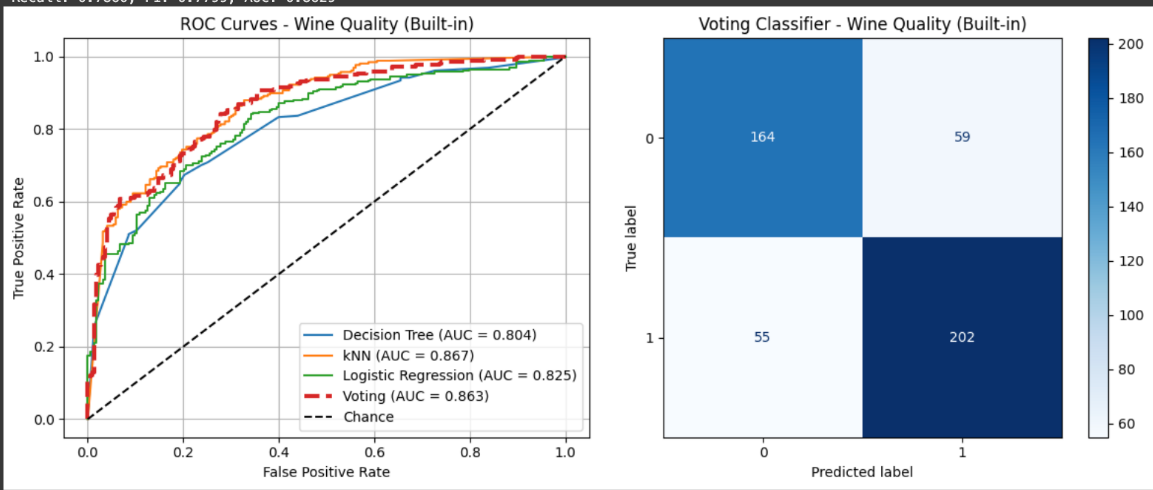
- **Wine Quality: kNN (k=7, distance weights, kbest=5)** — excels on smooth, non-linear boundaries with local structure and benefits from scaling + compact feature set.
- **HR Attrition: Logistic Regression (C=0.1, kbest=15)** — linear decision surface generalizes well across many encoded features; stronger regularization (C=0.1) combats overfitting and class noise.

5) Screenshots to Attach

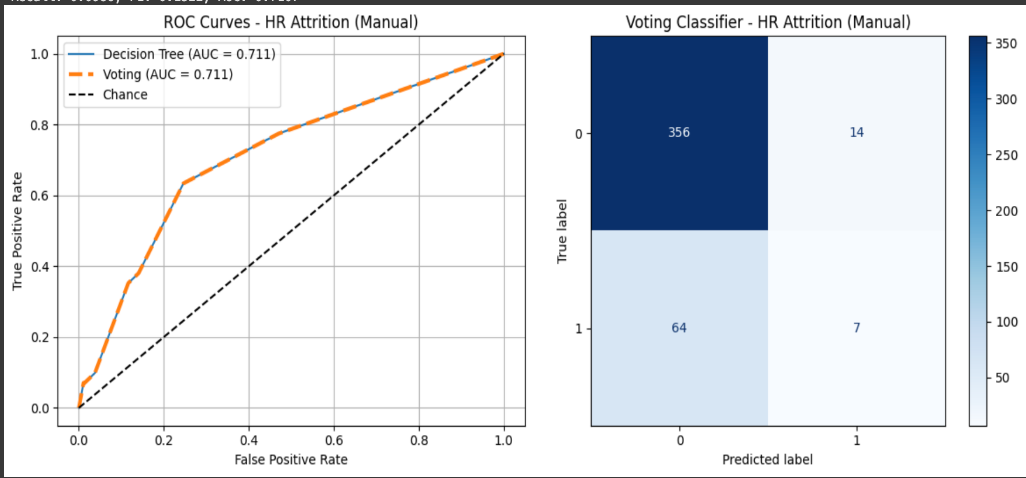
```
============================================================
EVALUATING BUILT-IN MODELS FOR WINE QUALITY
============================================================

Decision Tree:
 Accuracy: 0.7292
 Precision: 0.7725
 Recall: 0.7004
 F1-Score: 0.7347
 ROC AUC: 0.8042

kNN:
 Accuracy: 0.7667
 Precision: 0.7757
 Recall: 0.7938
 F1-Score: 0.7846
 ROC AUC: 0.8675

Logistic Regression:
 Accuracy: 0.7417
 Precision: 0.7628
 Recall: 0.7510
 F1-Score: 0.7569
 ROC AUC: 0.8247

--- Built-in Voting Classifier ---
Voting Classifier Performance:
 Accuracy: 0.7625, Precision: 0.7739
 Recall: 0.7860, F1: 0.7799, AUC: 0.8629
```



```
============================================================
EVALUATING BUILT-IN MODELS FOR WINE QUALITY
============================================================

Decision Tree:
 Accuracy: 0.7292
 Precision: 0.7725
 Recall: 0.7004
 F1-Score: 0.7347
 ROC AUC: 0.8042

kNN:
 Accuracy: 0.7667
 Precision: 0.7757
 Recall: 0.7938
 F1-Score: 0.7846
 ROC AUC: 0.8675

Logistic Regression:
 Accuracy: 0.7417
 Precision: 0.7628
 Recall: 0.7510
 F1-Score: 0.7569
 ROC AUC: 0.8247

--- Built-in Voting Classifier ---
Voting Classifier Performance:
 Accuracy: 0.7625, Precision: 0.7739
 Recall: 0.7860, F1: 0.7799, AUC: 0.8629
```

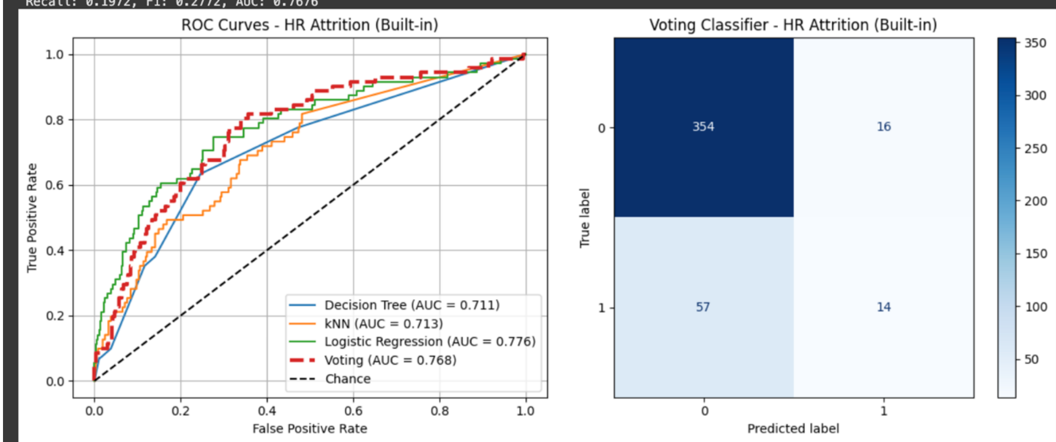


```
============================================================
EVALUATING MANUAL MODELS FOR HR ATTRITION
============================================================

Decision Tree:
 Accuracy: 0.8231
 Precision: 0.3333
 Recall: 0.0986
 F1-Score: 0.1522
 ROC AUC: 0.7107

--- Manual Voting Classifier ---
Voting Classifier Performance:
 Accuracy: 0.8231, Precision: 0.3333
 Recall: 0.0986, F1: 0.1522, AUC: 0.7107
```




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

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

Decision Tree:
 Accuracy: 0.8231
 Precision: 0.3333
 Recall: 0.0986
 F1-Score: 0.1522
 ROC AUC: 0.7107

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

Logistic Regression:
 Accuracy: 0.8571
 Precision: 0.6333
 Recall: 0.2676
 F1-Score: 0.3762
 ROC AUC: 0.7762

--- Built-in Voting Classifier ---
Voting Classifier Performance:
 Accuracy: 0.8345, Precision: 0.4667
 Recall: 0.1972, F1: 0.2772, AUC: 0.7676
```



6) Conclusion

- **Key Findings:**
  - On **Wine**, **kNN** (distance-weighted) achieved the best performance; an ensemble was competitive.
  - On **HR**, **Logistic Regression** outperformed others in Accuracy and AUC but had modest Recall due to class imbalance.
- **Takeaways on Model Selection:**
  - Pipelines with scaling + feature selection consistently help.
  - **Built-in GridSearchCV** is reliable and uncovers better models more quickly than manual loops, while reducing implementation error.
  - **Imbalanced data** requires targeted strategies (class weights, resampling, threshold tuning, probability calibration) to lift Recall without sacrificing too much Precision/AUC.