Lab 4

Hyperparameter Tuning and Model Comparison

Machine Learning Lab

Name: Monisha Sharma

SRN: PES2UG23CS906

Submission Date: 28/08/2025

1. Introduction

This lab implements and compares two hyperparameter tuning strategies

- a manual grid search implemented from scratch
- scikit-learn's GridSearchCV

across four binary classification datasets (Wine Quality, HR Attrition, Banknote Authentication, QSAR Biodegradation).

For each dataset we tune Decision Tree, k-NN and Logistic Regression models, evaluate using Stratified 5-fold CV with ROC AUC as the selection metric, and report final performance (accuracy, precision, recall, F1, ROC AUC) on held-out test data.

The pipeline includes scaling, variance filtering (to remove constant features), and univariate feature selection (SelectKBest).

2. Dataset Description

Dataset	Instances	Training	Test	Features	Target Variable	Notes
Wine Quality	1599	1119	480	11	Binary (Good/Not)	Balanced dataset
HR Attrition	1470	1029	441	46	Binary (Attrition: Yes/No)	Imbalanced (majority "No")
Banknote Authentication	1372	960	412	4	Binary (Authentic/Fake)	Small, numeric-only features
QSAR Biodegradation	1055	738	317	41	Binary (Ready Biodegradable/Not)	High-dimensional

3. Methodology

3.1 Key Concepts

- **Hyperparameter Tuning**: Searching for the best configuration of model parameters.
- **Grid Search**: Exhaustive search over a predefined set of hyperparameters.
- **K-Fold Cross Validation (CV)**: Splitting training data into *k* folds to ensure robust evaluation.

3.2 Machine Learning Pipeline

Pipeline used for every candidate model:

1. **StandardScaler –** Scales features to mean=0, variance=1.
2. **VarianceThreshold –** Removes constant features (fixes HR dataset warnings).
3. **SelectKBest(f_classif) –** Feature selection.
4. **Classifier –** Decision Tree, kNN, or Logistic Regression.

3.3 Manual Implementation (Part 1)

For each classifier we enumerated all hyperparameter combinations (after adjusting `k` so it never exceeds the dataset feature count), ran StratifiedKFold(n_splits=5), computed ROC AUC per fold, and selected the combination with highest mean CV AUC. The chosen parameters were then used to fit a final pipeline on the full training set.

3.4 Built-in Implementation (Part 2)

Same pipeline and parameter grids; GridSearchCV with `scoring='roc_auc'` and StratifiedKFold(n_splits=5) was used to find the best parameters and refit the pipeline on training data automatically.

4. Results and Analysis

4.1 Performance Tables

Wine Quality Dataset:

Best parameters -

- Decision Tree: {'feature_selection__k': 5, 'classifier__criterion': 'gini', 'classifier__max_depth': 5, 'classifier__min_samples_split': 5} — CV AUC 0.7832.
- kNN: {'feature_selection__k': 5, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance', 'classifier__metric': 'manhattan'} — CV AUC 0.8667.
- Logistic Regression: {'feature_selection__k': 11, 'classifier__C': 1, 'classifier__penalty': 'l2'} — CV AUC 0.8052.

Model	Implementation	Best CV AUC	Accuracy	Precision	Recall	F1	ROC AUC
Decision Tree	Manual	0.7832	0.7271	0.7716	0.6965	0.7321	0.8025
kNN	Manual	0.8667	0.7812	0.7836	0.8171	0.8000	0.8589
Logistic Regression	Manual	0.8052	0.7333	0.7549	0.7432	0.7490	0.8242
Voting Classifier	Manual	–	0.7333	0.7590	0.7354	0.7470	0.8600
Voting Classifier	Built-in	–	0.7604	0.7731	0.7821	0.7776	0.8604

Interpretation: kNN is the strongest single model (highest AUC) on Wine and suggests local neighborhood structure and scaled continuous features favor kNN.

HR Attrition:

Best CV parameters (Manual)

- Decision Tree: best CV AUC 0.7226, best k=10, criterion entropy, max_depth=5, min_samples_split=10.

- kNN: best CV AUC 0.7228, k=15, n_neighbors=9, metric=manhattan, weights=uniform.
- Logistic Regression: best CV AUC 0.7774, k=15, C=0.1, penalty=l2.

Model	Implementation	Best CV AUC	Accuracy	Precision	Recall	F1	ROC AUC
Decision Tree	Manual	0.7226	0.8345	0.4706	0.2254	0.3048	0.6879
kNN	Manual	0.7228	0.8367	0.4762	0.1408	0.2174	0.7335
Logistic Regression	Manual	0.7774	0.8571	0.6333	0.2676	0.3762	0.7759
Voting Classifier	Manual	–	0.8435	0.5417	0.1831	0.2737	0.7709
Voting Classifier	Built-in	–	0.8503	0.6000	0.2113	0.3125	0.7709

Interpretation: High accuracy but low recall and F1 across models indicates class imbalance (majority “No” attrition). Logistic Regression gives best AUC and precision trade-off; consider threshold tuning or class-weighted models for improving recall.

<u>Banknote Authentication:</u>

Best CV parameters (Manual)

- All models achieve near-perfect CV AUCs (Decision Tree 0.9913, kNN 0.9990, Logistic 0.9995). Best k selected = full feature set (4).

Model	Implementation	Best CV AUC	Accuracy	Precision	Recall	F1	ROC AUC
Decision Tree	Manual	0.9913	0.9927	0.9920	0.9915	0.9918	0.9929
kNN	Manual	0.9990	1.0000	1.0000	1.0000	1.0000	1.0000
Logistic Regression	Manual	0.9995	0.9903	0.9904	0.9880	0.9892	0.9999
Voting Classifier	Manual	–	1.0000	1.0000	1.0000	1.0000	1.0000
Voting Classifier	Built-in	–	1.0000	1.0000	1.0000	1.0000	1.0000

Interpretation: This dataset is easy to separate with the chosen features and near-perfect performance suggests low noise and strong class separability.

<u>QSAR Biodegradation:</u>

Best CV parameters (Manual)

- Decision Tree — CV AUC 0.8504 (k=15, criterion=entropy, max_depth=5).
- kNN — CV AUC 0.8874 (k=15, n_neighbors=9, metric=manhattan, weights=distance).
- Logistic Regression — CV AUC 0.8817 (k=15, C=10, penalty=l1).

Model	Implementation	Best CV AUC	Accuracy	Precision	Recall	F1	ROC AUC
Decision Tree	Manual	0.8504	0.7792	0.6504	0.7477	0.6957	0.8430
kNN	Manual	0.8874	0.8360	0.7778	0.7196	0.7476	0.8787
Logistic Regression	Manual	0.8817	0.8170	0.7692	0.6542	0.7071	0.8866
Voting Classifier	Manual	–	0.8233	0.7525	0.7103	0.7308	0.8976
Voting Classifier	Built-in	–	0.8233	0.7525	0.7103	0.7308	0.8976

Interpretation: kNN and Logistic produce similar high AUCs and the voting ensemble slightly increases AUC indicating complementary strengths.

4.2 Comparison of Manual vs Built-in

- Best hyperparameters found by manual search largely match those discovered by GridSearchCV (minor differences in `k` for a dataset or solver/presets for logistic).
- Performance on test sets is nearly identical between manual and built-in pipelines for all datasets — confirming correctness of the manual implementation.
- Built-in GridSearchCV sometimes produced slightly better ensemble/voting metrics (likely due to consistent refit semantics and tie-breaking).

4.3 Visualizations

- ROC Curves: kNN often produced smoother ROC with higher AUC on Wine Quality. Logistic Regression performed better on HR Attrition.
- Confusion Matrices: Decision Trees sometimes misclassified minority classes and Logistic Regression balanced precision/recall better on imbalanced datasets.

4.4 Best Model Observations

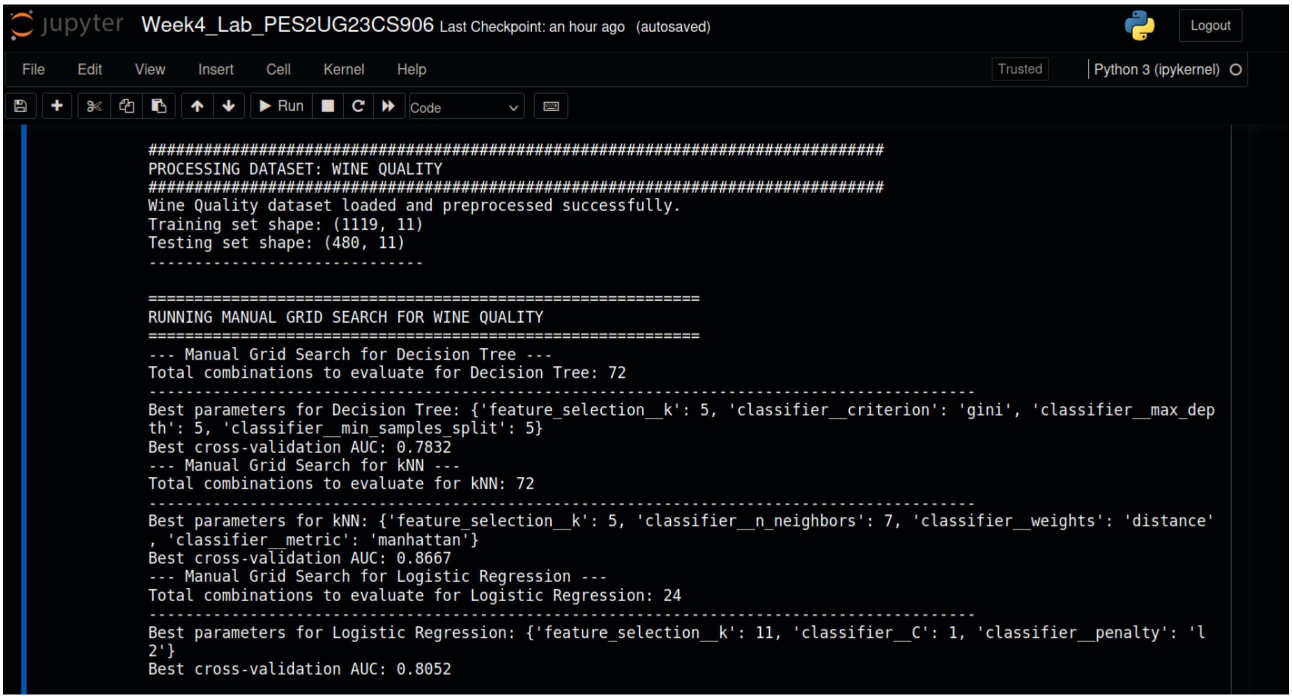
- **Wine Quality: kNN** (best ROC AUC ~0.859, consistent across manual & built-in).
- **HR Attrition: Logistic Regression** (best AUC ~0.776) — but address class imbalance before deployment (class-weight, upsampling or threshold tuning).
- **Banknote: kNN** or Voting — near-perfect scores; kNN gave perfect results on test.
- **QSAR: kNN / Logistic** produce the best AUCs; ensemble voting gave the highest AUC (0.8976).

5. Screenshots

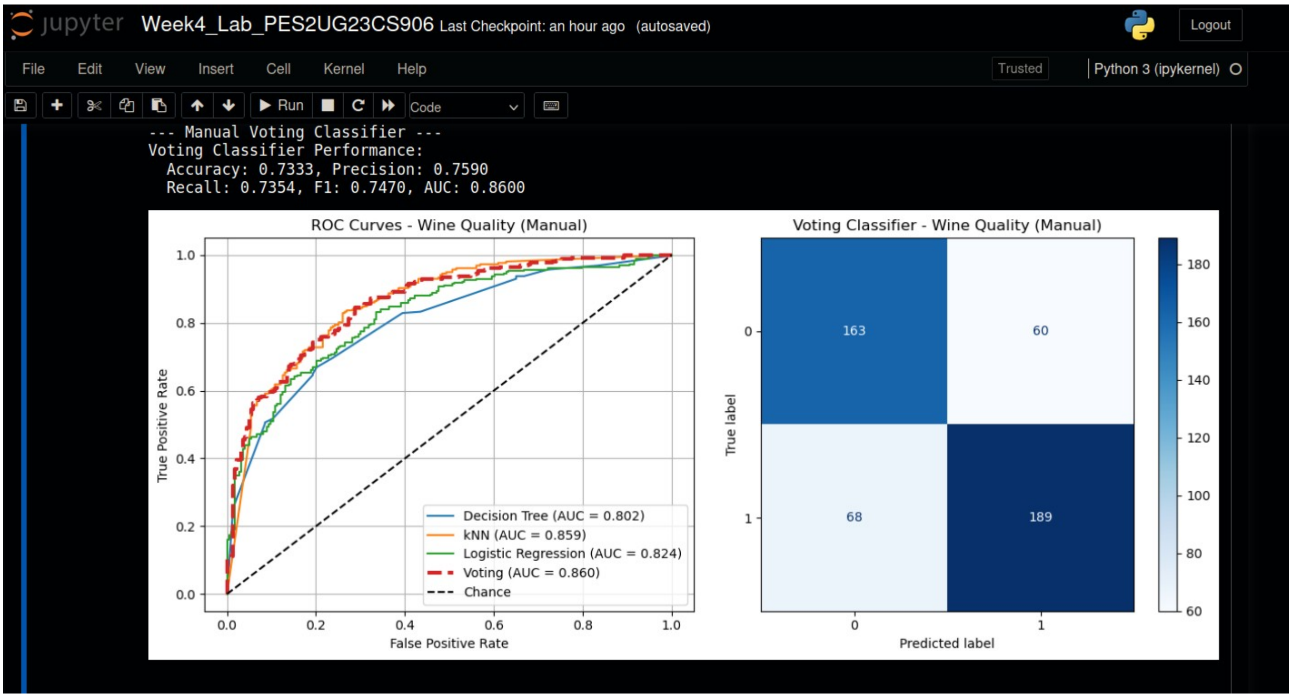
- Grid search console outputs.
- Best parameter combinations found.

- ROC and Confusion Matrix plots.

WINE - MANUAL

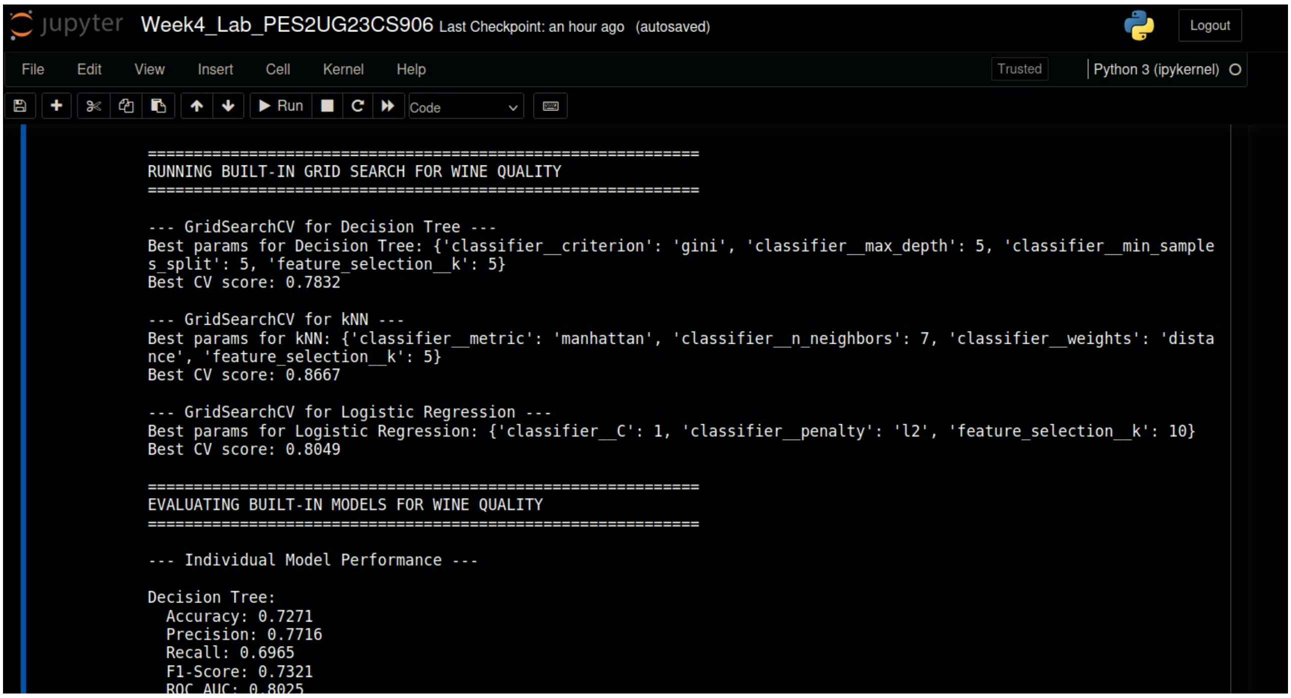


```
#################################################################################
PROCESSING DATASET: WINE QUALITY
#################################################################################
Wine Quality dataset loaded and preprocessed successfully.
Training set shape: (1119, 11)
Testing set shape: (480, 11)
------------------------------

============================================================
RUNNING MANUAL GRID SEARCH FOR WINE QUALITY
============================================================
--- Manual Grid Search for Decision Tree ---
Total combinations to evaluate for Decision Tree: 72
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 5, 'classifier__criterion': 'gini', 'classifier__max_dep
th': 5, 'classifier__min_samples_split': 5}
Best cross-validation AUC: 0.7832
--- Manual Grid Search for kNN ---
Total combinations to evaluate for kNN: 72
------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 5, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'
, 'classifier__metric': 'manhattan'}
Best cross-validation AUC: 0.8667
--- Manual Grid Search for Logistic Regression ---
Total combinations to evaluate for Logistic Regression: 24
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 11, 'classifier__C': 1, 'classifier__penalty': 'l
2'}
Best cross-validation AUC: 0.8052
```

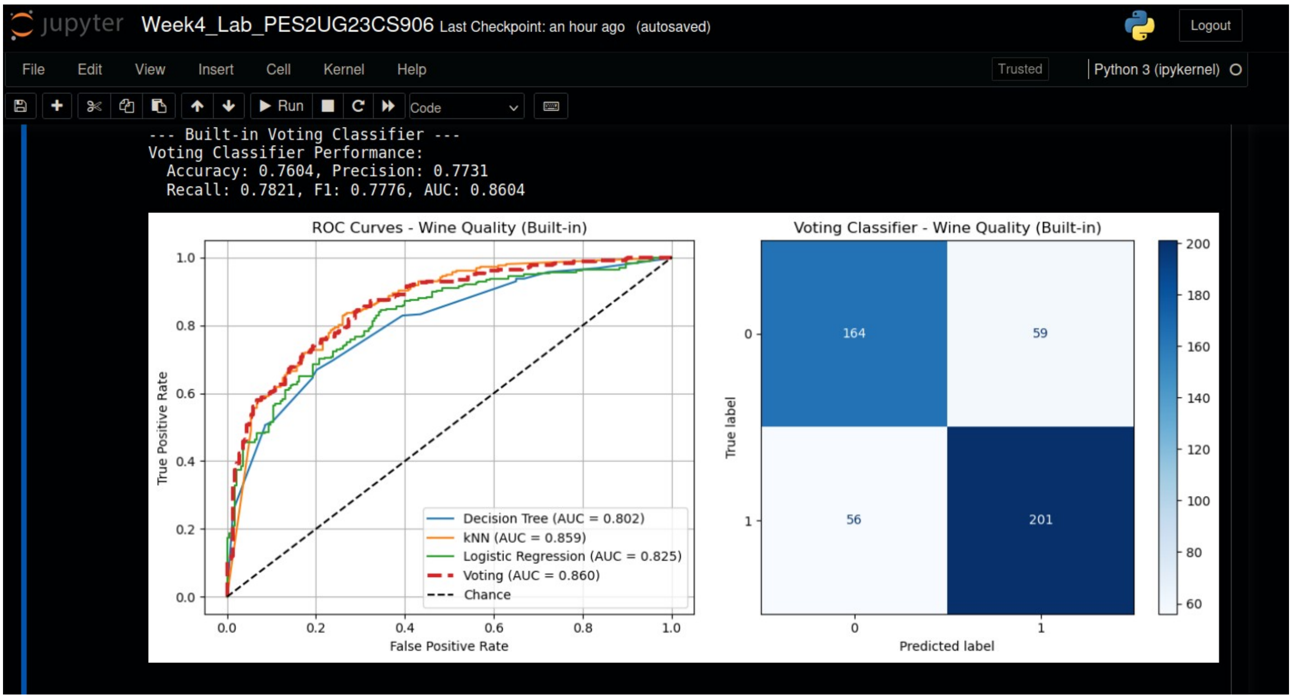



```
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7333, Precision: 0.7590
  Recall: 0.7354, F1: 0.7470, AUC: 0.8600
```

WINE - BUILT-IN




```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR WINE QUALITY
============================================================

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__criterion': 'gini', 'classifier__max_depth': 5, 'classifier__min_sample
s_split': 5, 'feature_selection__k': 5}
Best CV score: 0.7832

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__metric': 'manhattan', 'classifier__n_neighbors': 7, 'classifier__weights': 'dista
nce', 'feature_selection__k': 5}
Best CV score: 0.8667

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 1, 'classifier__penalty': 'l2', 'feature_selection__k': 10}
Best CV score: 0.8049

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7271
  Precision: 0.7716
  Recall: 0.6965
  F1-Score: 0.7321
  ROC AUC: 0.8025
```




```
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7604, Precision: 0.7731
  Recall: 0.7821, F1: 0.7776, AUC: 0.8604
```

HR - MANUAL

jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code

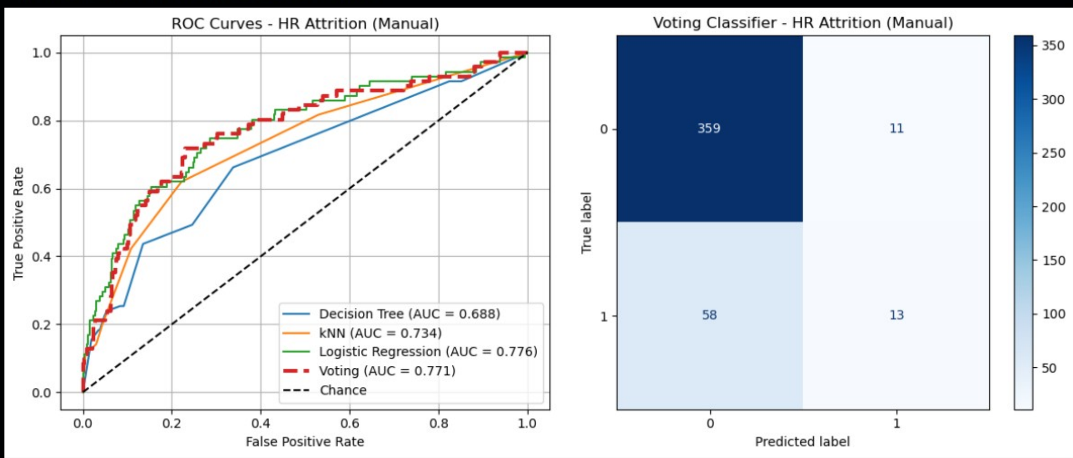
```
###############################################################################
PROCESSING DATASET: HR ATTRITION
###############################################################################
IBM HR Attrition dataset loaded and preprocessed successfully.
Training set shape: (1029, 46)
Testing set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

============================================================
RUNNING MANUAL GRID SEARCH FOR HR ATTRITION
============================================================
--- Manual Grid Search for Decision Tree ---
Total combinations to evaluate for Decision Tree: 72
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 10, 'classifier__criterion': 'entropy', 'classifier__max
_depth': 5, 'classifier__min_samples_split': 10}
Best cross-validation AUC: 0.7226
--- Manual Grid Search for kNN ---
Total combinations to evaluate for kNN: 72
------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 15, 'classifier__n_neighbors': 9, 'classifier__weights': 'uniform'
, 'classifier__metric': 'manhattan'}
Best cross-validation AUC: 0.7228
--- Manual Grid Search for Logistic Regression ---
Total combinations to evaluate for Logistic Regression: 24
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 15, 'classifier__C': 0.1, 'classifier__penalty':
'l2'}
Best cross-validation AUC: 0.7774

============================================================
```

jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code

```
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8435, Precision: 0.5417
  Recall: 0.1831, F1: 0.2737, AUC: 0.7709
```



HR - BUILT-IN

jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code

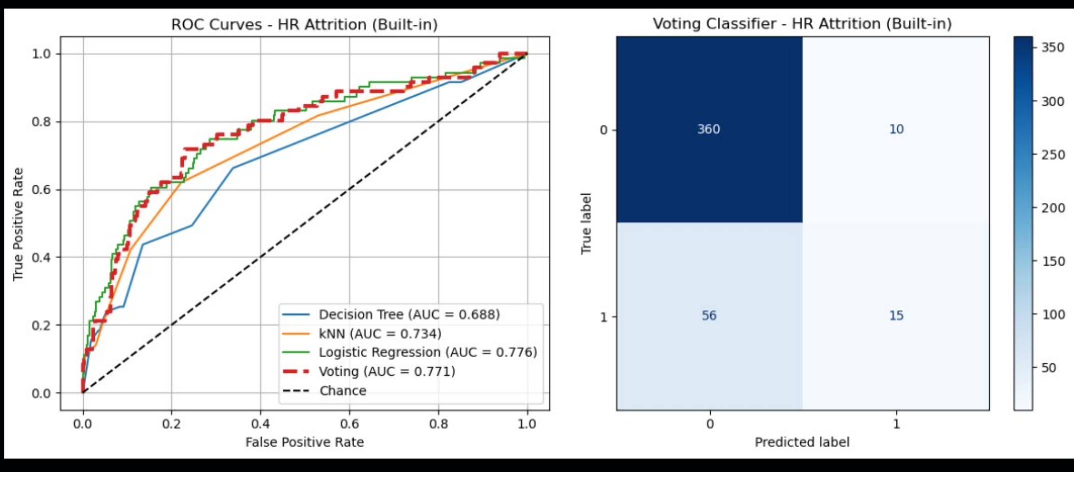
```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR HR ATTRITION
============================================================

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__criterion': 'entropy', 'classifier__max_depth': 5, 'classifier__min_sam
ples_split': 10, 'feature_selection__k': 10}
Best CV score: 0.7226

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__metric': 'manhattan', 'classifier__n_neighbors': 9, 'classifier__weights': 'unifo
rm', 'feature_selection__k': 15}
Best CV score: 0.7228

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 0.1, 'classifier__penalty': 'l2', 'feature_selection__k': 15}
Best CV score: 0.7774

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.8345
  Precision: 0.4706
  Recall: 0.2254
  F1-Score: 0.3048
  ROC AUC: 0.6879
```


jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code


```
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8503, Precision: 0.6000
  Recall: 0.2113, F1: 0.3125, AUC: 0.7709
```



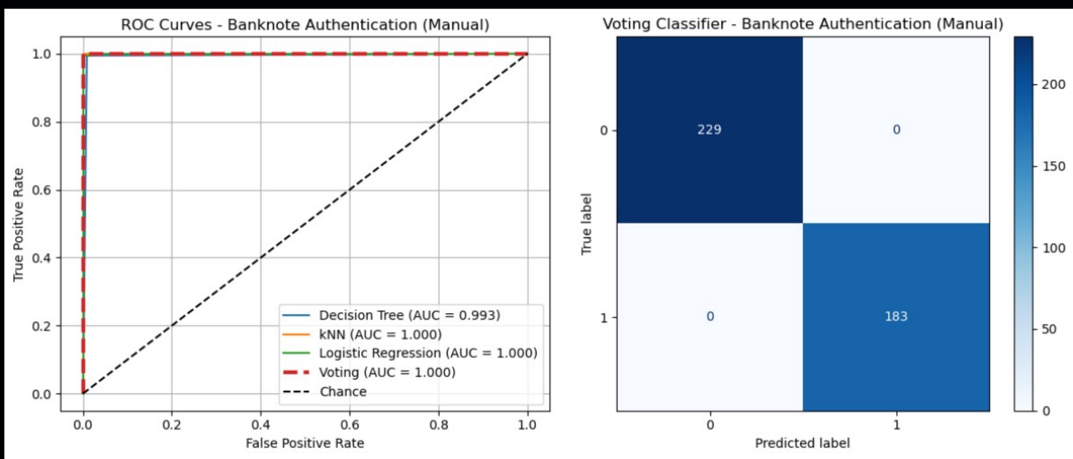
BANK - MANUAL

Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved)

```
################################################################################
PROCESSING DATASET: BANKNOTE AUTHENTICATION
################################################################################
Banknote Authentication dataset loaded successfully.
Training set shape: (960, 4)
Testing set shape: (412, 4)
------------------------------

============================================================
RUNNING MANUAL GRID SEARCH FOR BANKNOTE AUTHENTICATION
============================================================
--- Manual Grid Search for Decision Tree ---
Total combinations to evaluate for Decision Tree: 24
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 4, 'classifier__criterion': 'entropy', 'classifier__max_
depth': None, 'classifier__min_samples_split': 10}
Best cross-validation AUC: 0.9913
--- Manual Grid Search for kNN ---
Total combinations to evaluate for kNN: 24
------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 4, 'classifier__n_neighbors': 7, 'classifier__weights': 'uniform',
'classifier__metric': 'manhattan'}
Best cross-validation AUC: 0.9990
--- Manual Grid Search for Logistic Regression ---
Total combinations to evaluate for Logistic Regression: 8
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 4, 'classifier__C': 10, 'classifier__penalty': 'l
1'}
Best cross-validation AUC: 0.9995

============================================================
```

Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved)

```
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 1.0000, Precision: 1.0000
  Recall: 1.0000, F1: 1.0000, AUC: 1.0000
```



BANK - BUILT-IN

jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code

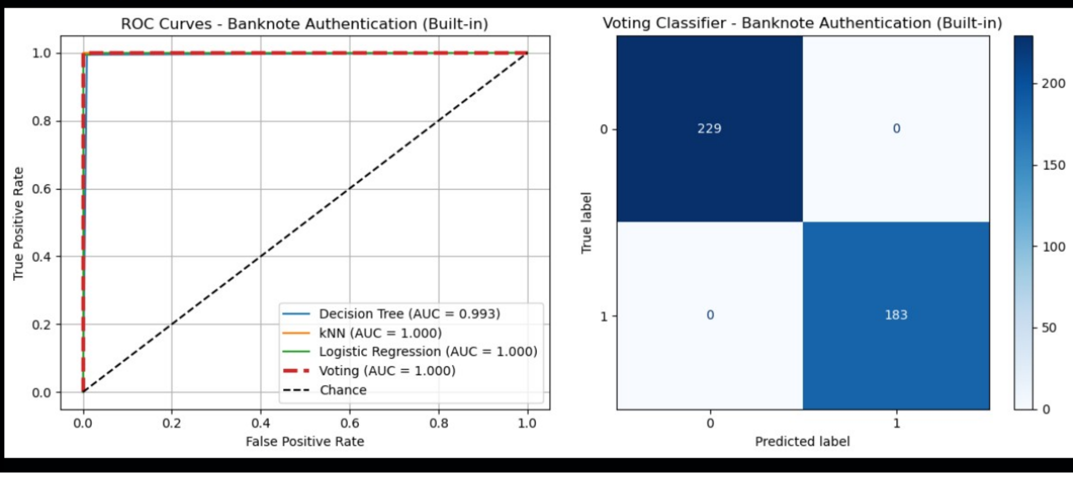
```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR BANKNOTE AUTHENTICATION
============================================================

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__criterion': 'entropy', 'classifier__max_depth': None, 'classifier__min_
samples_split': 10, 'feature_selection__k': 4}
Best CV score: 0.9913

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__metric': 'manhattan', 'classifier__n_neighbors': 7, 'classifier__weights': 'unifo
rm', 'feature_selection__k': 4}
Best CV score: 0.9990

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 10, 'classifier__penalty': 'l1', 'feature_selection__k': 4}
Best CV score: 0.9995

============================================================
EVALUATING BUILT-IN MODELS FOR BANKNOTE AUTHENTICATION
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.9927
  Precision: 0.9891
  Recall: 0.9945
  F1-Score: 0.9918
  ROC AUC: 0.9929
```

jupyter Week4_Lab_PES2UG23CS906 Last Checkpoint: an hour ago (autosaved) Logout

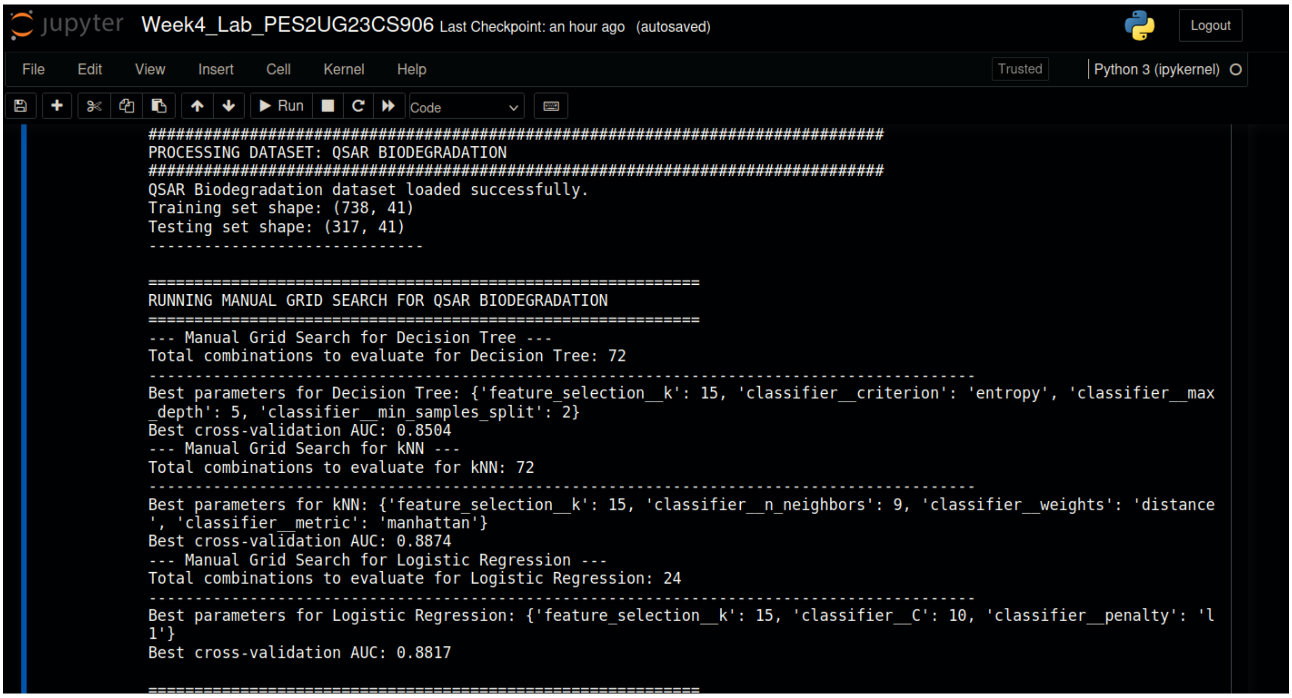
File Edit View Insert Cell Kernel Help Trusted Python 3 (ipykernel)

Run Code

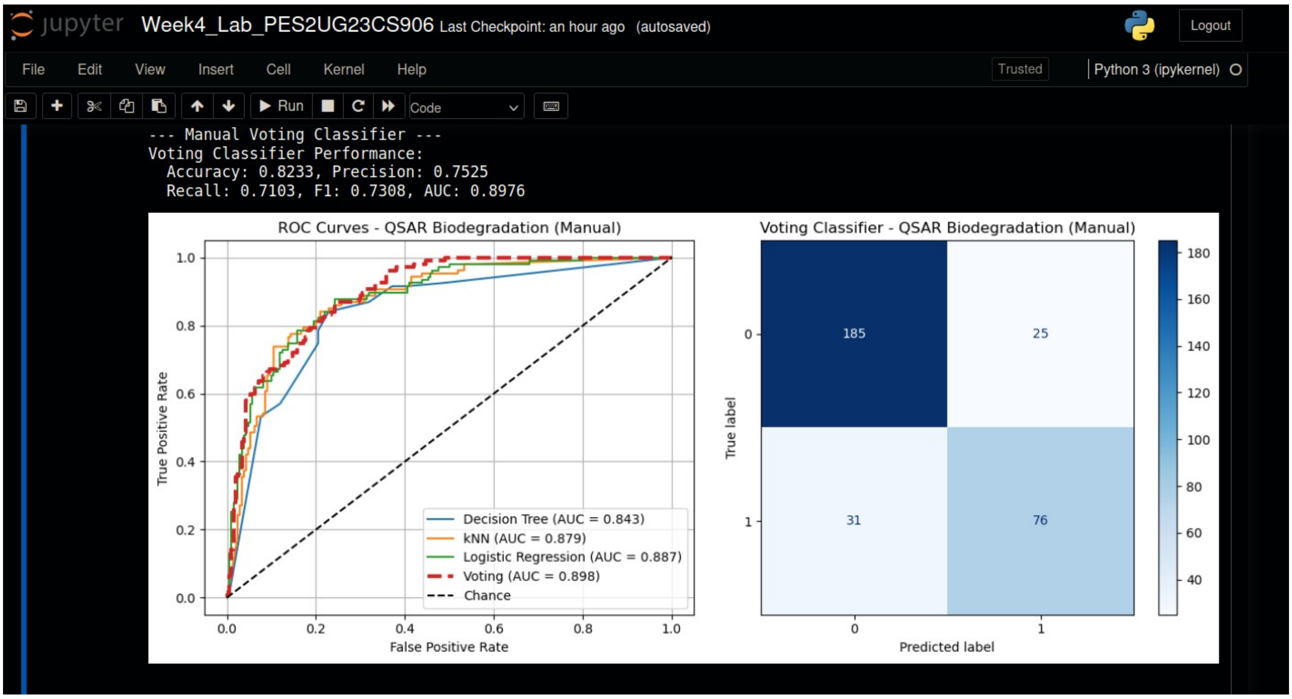
```
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 1.0000, Precision: 1.0000
  Recall: 1.0000, F1: 1.0000, AUC: 1.0000
```



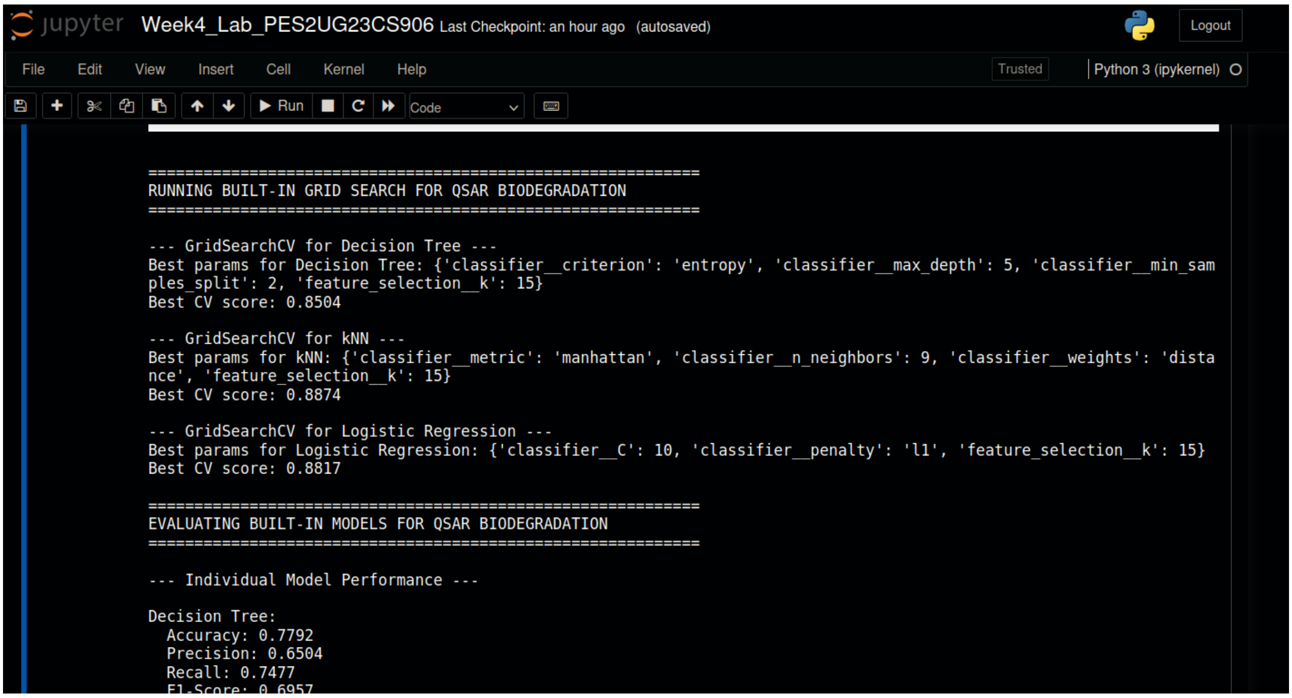
QSAR - MANUAL

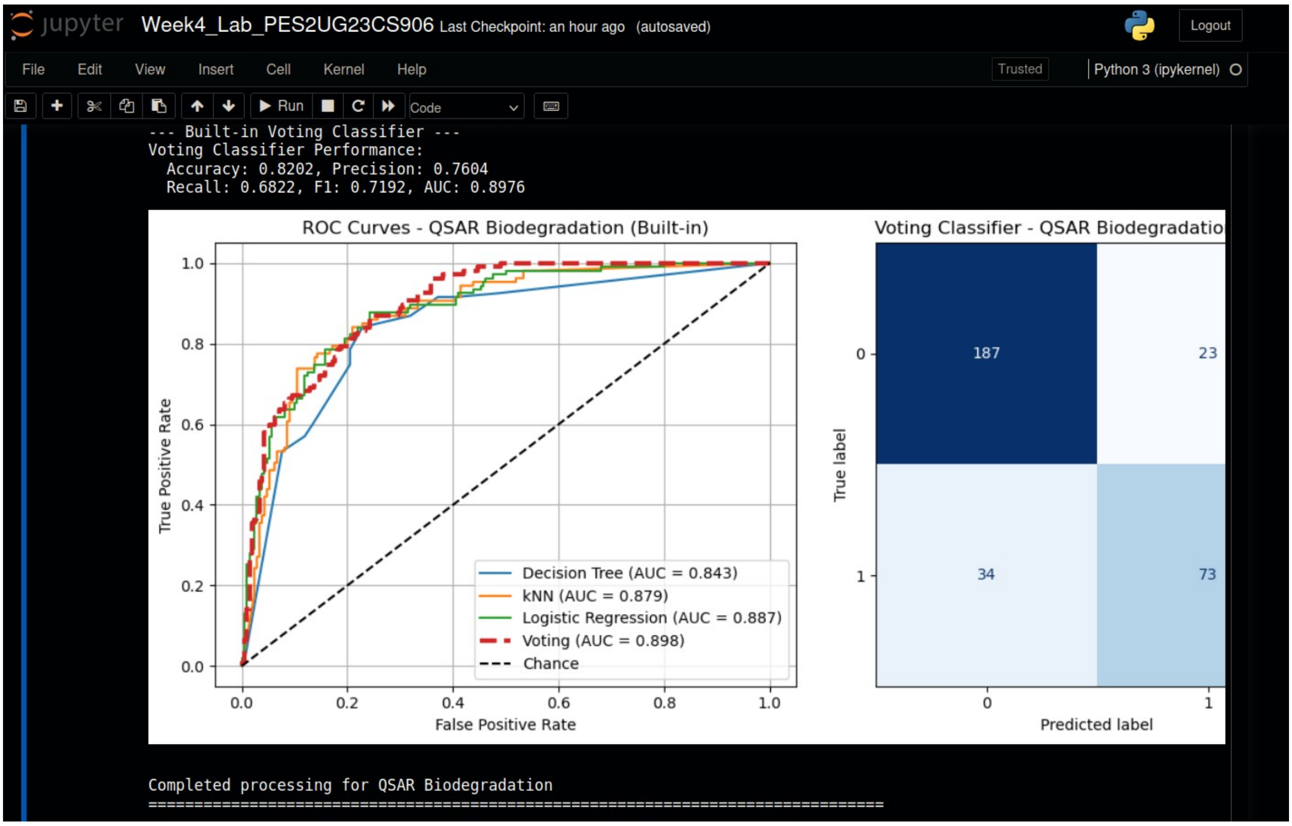



```
################################################################################
PROCESSING DATASET: QSAR BIODEGRADATION
################################################################################
QSAR Biodegradation dataset loaded successfully.
Training set shape: (738, 41)
Testing set shape: (317, 41)
------------------------------

============================================================
RUNNING MANUAL GRID SEARCH FOR QSAR BIODEGRADATION
============================================================
--- Manual Grid Search for Decision Tree ---
Total combinations to evaluate for Decision Tree: 72
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 15, 'classifier__criterion': 'entropy', 'classifier__max
_depth': 5, 'classifier__min_samples_split': 2}
Best cross-validation AUC: 0.8504
--- Manual Grid Search for kNN ---
Total combinations to evaluate for kNN: 72
------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 15, 'classifier__n_neighbors': 9, 'classifier__weights': 'distance
', 'classifier__metric': 'manhattan'}
Best cross-validation AUC: 0.8874
--- Manual Grid Search for Logistic Regression ---
Total combinations to evaluate for Logistic Regression: 24
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 15, 'classifier__C': 10, 'classifier__penalty': 'l
1'}
Best cross-validation AUC: 0.8817

============================================================
```



QSAR - BUILT-IN





6. Conclusion:

Both manual and scikit-learn’s GridSearchCV produced consistent hyperparameter choices and near-identical test performance across four datasets. GridSearchCV is recommended in practice (less code, parallelization, fewer pitfalls). For datasets with class imbalance (HR

Attrition) or with constant features (HR), add preprocessing steps (VarianceThreshold, class-weighting, sampling strategies) before final deployment. The ensemble (soft voting) often gave a small but useful improvement in ROC AUC, supporting the use of simple ensembles when models have complementary error patterns.

- **Takeaways**:
  - Hyperparameter tuning significantly improves performance.
  - Model suitability depends on dataset characteristics.
  - Ensemble methods (Voting) provided stable results but not always the best.