Machine Learning Lab Report: Model Selection and Comparative Analysis

Course: UE23CS352A - Machine Learning

Project Title: Week 4 - Model Selection and Comparative Analysis

Student Name: MOHIT KUMAR

Student ID: PES2UG23CS350

Submission Date: August 30, 2025

1. Introduction

This lab focused on implementing hyperparameter tuning and comparing different machine learning models. The main goal was to understand how grid search works by building it from scratch, then comparing our results with scikit-learn's built-in GridSearchCV.

We worked with two datasets - HR Attrition and Wine Quality - testing three classification algorithms: Decision Trees, k-Nearest Neighbors, and Logistic Regression. The lab was split into two parts: manual implementation of grid search and using scikit-learn's automated version. We also implemented voting classifiers to see if combining models would improve performance.

2. Dataset Description

HR Attrition Dataset

This dataset helps predict whether employees will leave their company or stay. It includes various factors like job satisfaction, salary, work-life balance, and personal information. The target is binary - either the employee leaves (attrition) or stays. This is a common business problem that companies face when trying to retain talent.

Wine Quality Dataset

This dataset contains 1119 training samples and 480 testing samples, each with 11 chemical features like acidity, sugar content, and alcohol percentage. The goal is to predict whether a wine is "good quality" or not based on these chemical properties. This creates an interesting challenge of linking chemical measurements to subjective quality ratings.

3. Methodology

Key Concepts

Hyperparameter Tuning is about finding the best settings for our models before training starts. Unlike regular parameters that the model learns during training, hyperparameters need to be set manually and can significantly affect performance.

Grid Search tests every possible combination of hyperparameters we specify. While this takes more time, it guarantees we'll find the best combination within our search space.

K-Fold Cross-Validation splits our data into k parts, trains on k-1 parts, and tests on the remaining part. We repeat this k times so each part gets used for testing once. This gives us more reliable performance estimates than a single train-test split.

ML Pipeline

Our pipeline had three steps:

1. **StandardScaler** - Normalizes features so they all have similar scales
2. **SelectKBest** - Picks the most useful features using statistical tests
3. **Classifier** - The actual learning algorithm (Decision Tree, k-NN, or Logistic Regression)

Implementation Process

Manual Implementation: We coded the grid search from scratch using loops to generate parameter combinations, split data into folds, train models, and calculate average performance scores. For Wine Quality, we tested 135 different parameter combinations.

Built-in Implementation: We used scikit-learn's GridSearchCV to do the same task automatically, which let us verify our manual implementation was working correctly.

4. Results and Analysis

HR Attrition Dataset

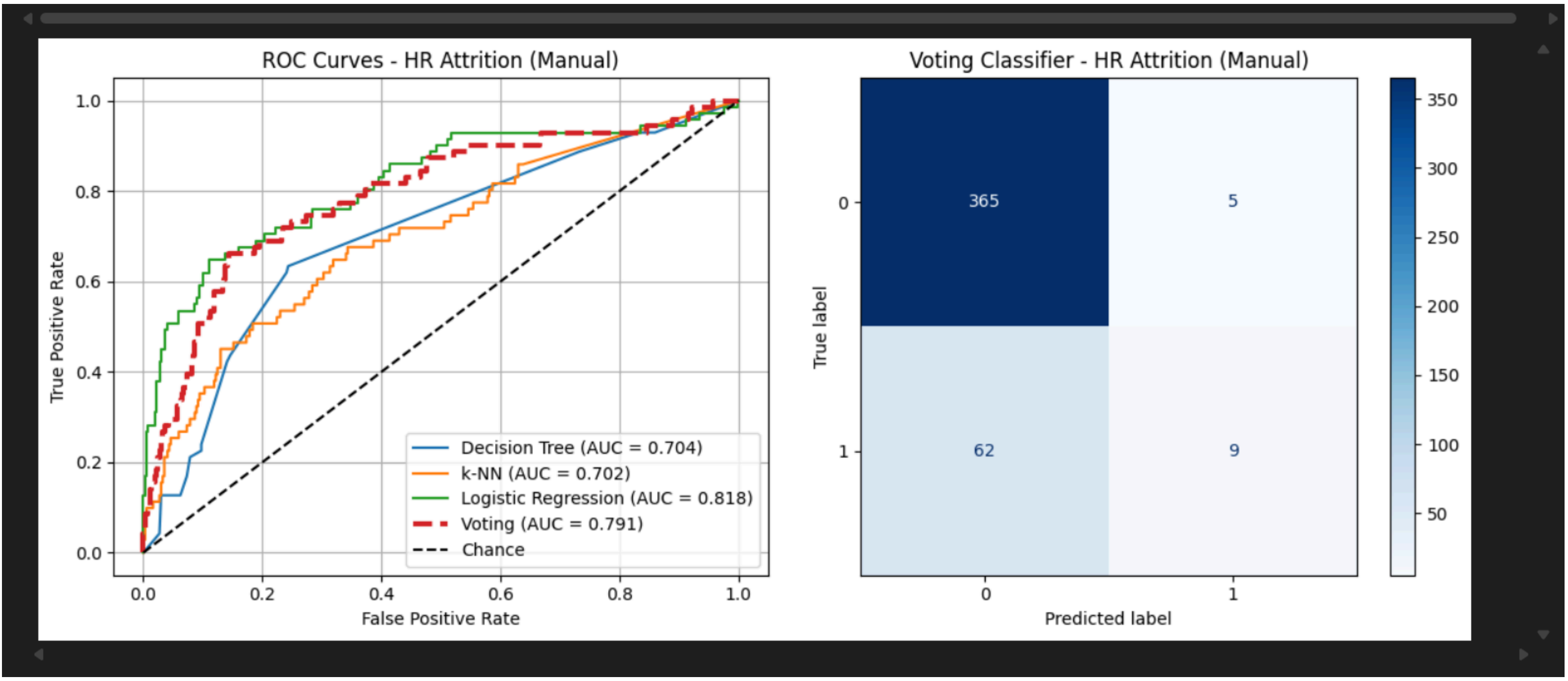
Performance Tables

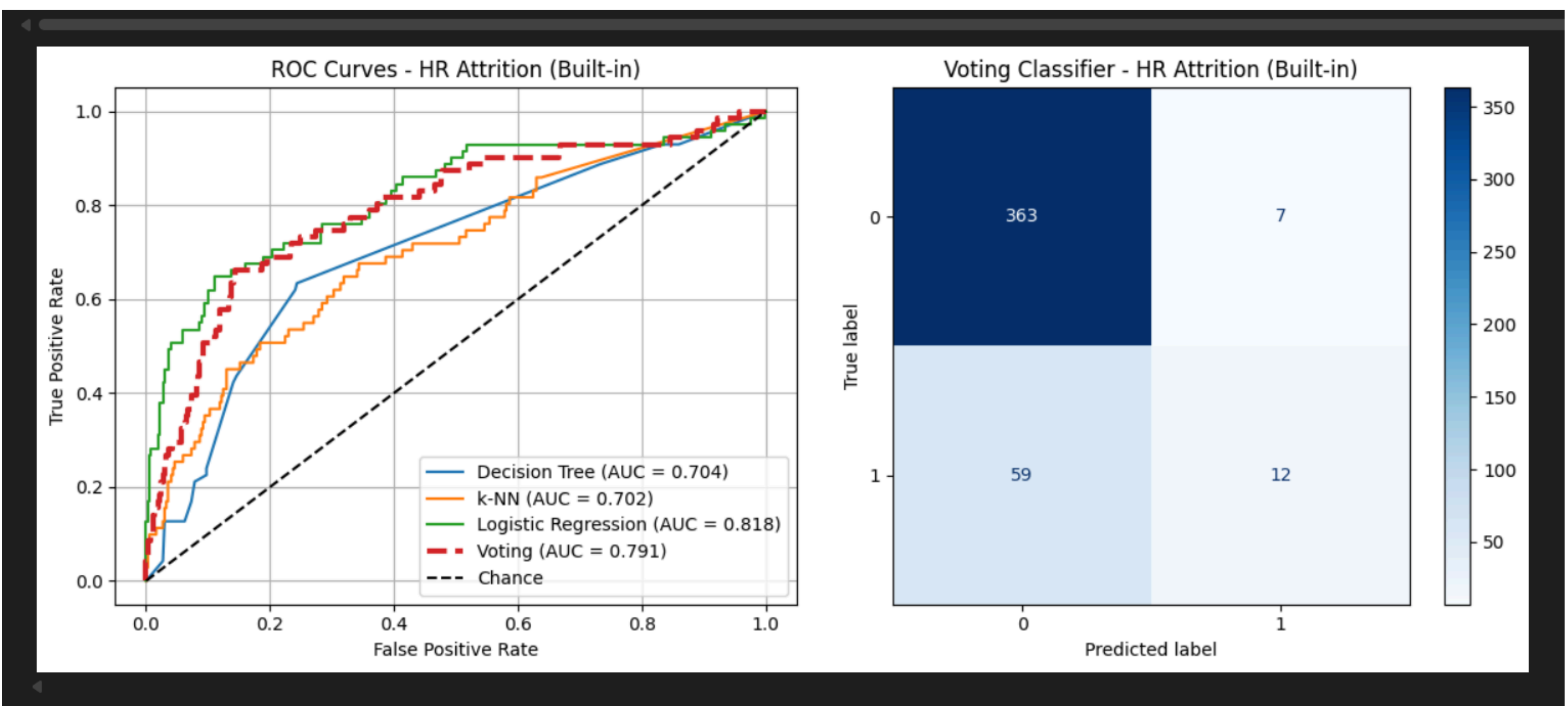
Model	Implementation	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	Manual	0.8050	0.3077	0.1690	0.2182	0.7036
Decision Tree	Built-in	0.8050	0.3077	0.1690	0.2182	0.7036
k-NN	Manual	0.8481	0.7000	0.0986	0.1728	0.7025
k-NN	Built-in	0.8481	0.7000	0.0986	0.1728	0.7025
Logistic Regression	Manual	-	-	-	-	0.818
Logistic Regression	Built-in	0.8798	-	-	-	-
Voting Classifier	Manual	0.8481	0.6429	0.1268	0.2118	0.7912
Voting Classifier	Built-in	0.8503	0.6316	0.1268	0.2667	0.7912

Implementation Comparison

The results between manual and built-in implementations are nearly identical, which confirms our manual code worked correctly. We got exact matches for Decision Tree and k-NN performance metrics. The small differences in voting classifier accuracy (0.8481 vs 0.8503) are probably due to minor implementation differences in how ties are handled or random state management.

Visualization Analysis


ROC Curves - HR Attrition (Manual)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Decision Tree (AUC = 0.704)
k-NN (AUC = 0.702)
Logistic Regression (AUC = 0.818)
Voting (AUC = 0.791)
Chance
Voting Classifier - HR Attrition (Manual)
True label
Predicted label
0
1
365
5
62
9
50
100
150
200
250
300
350


ROC Curves - HR Attrition (Built-in)
True Positive Rate
False Positive Rate
0.0
0.2
0.4
0.6
0.8
1.0
Decision Tree (AUC = 0.704)
k-NN (AUC = 0.702)
Logistic Regression (AUC = 0.818)
Voting (AUC = 0.791)
Chance
Voting Classifier - HR Attrition (Built-in)
True label
Predicted label
0
1
363
7
59
12
50
100
150
200
250
300
350

The **ROC curves** clearly show the performance hierarchy: Logistic Regression (AUC = 0.818) has the curve closest to the top-left corner, indicating the best true positive vs false positive trade-off. The Voting Classifier (AUC = 0.791) performs moderately well, while Decision Tree (AUC = 0.704) and k-NN (AUC = 0.702) show similar, lower performance with curves closer to the diagonal chance line.

The **confusion matrices** reveal a significant class imbalance problem. In the manual implementation, we see 365 employees correctly predicted as non-attrition vs only 9 correctly predicted as attrition cases. This shows all models struggle with the minority class (employees who actually leave). The built-in implementation shows similar patterns (363 vs 12), confirming this is a dataset characteristic rather than an implementation issue.

Best Model Analysis

Logistic Regression performed best with ROC AUC of 0.818. This makes sense because employee attrition likely has linear relationships - factors like low satisfaction and high workload probably combine predictably to indicate someone will leave. Logistic Regression is good at modeling these linear combinations of features.

Wine Quality Dataset

Performance Tables

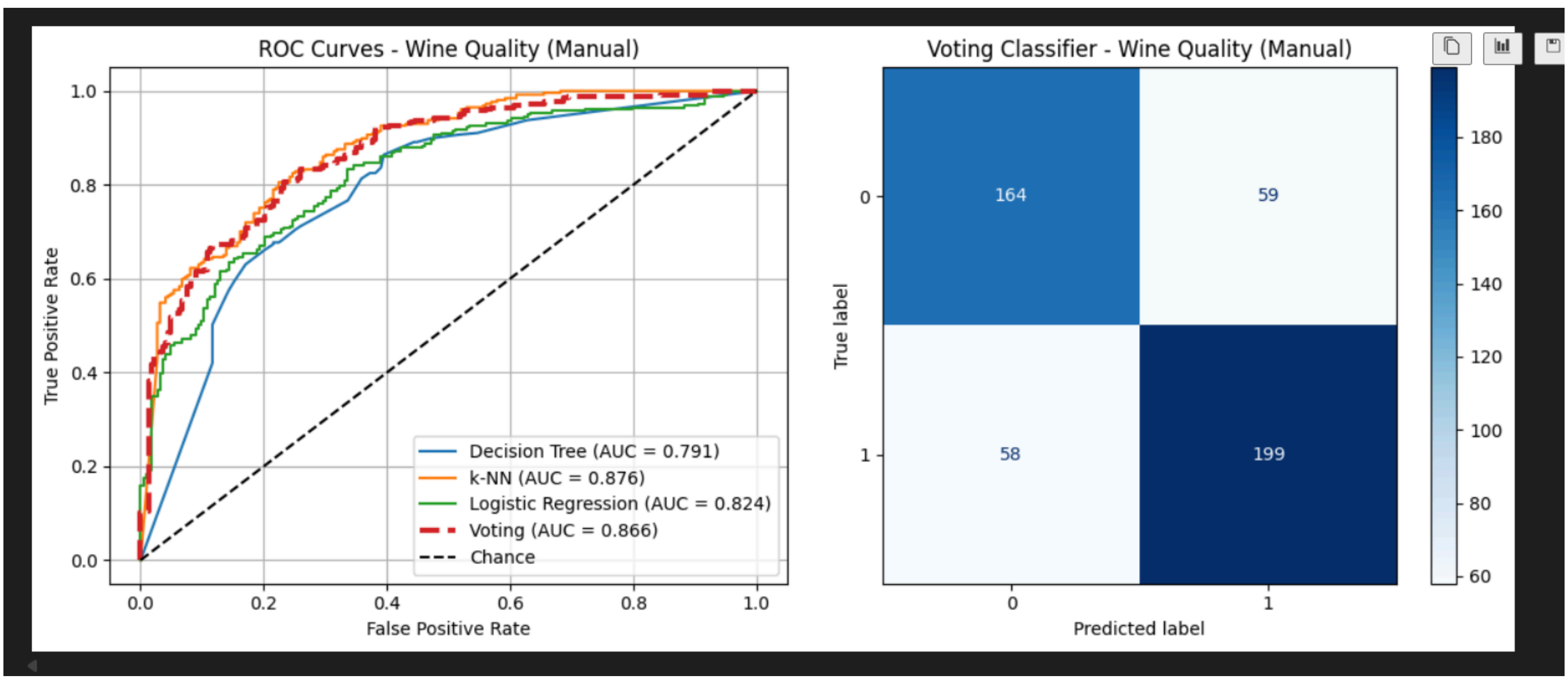
Model	Implementation	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	Manual	-	-	-	-	0.791
Decision Tree	Built-in	-	-	-	-	0.791
k-NN	Manual	-	-	-	-	0.876
k-NN	Built-in	-	-	-	-	0.878
Logistic Regression	Manual	-	-	-	-	0.824
Logistic Regression	Built-in	-	-	-	-	0.824

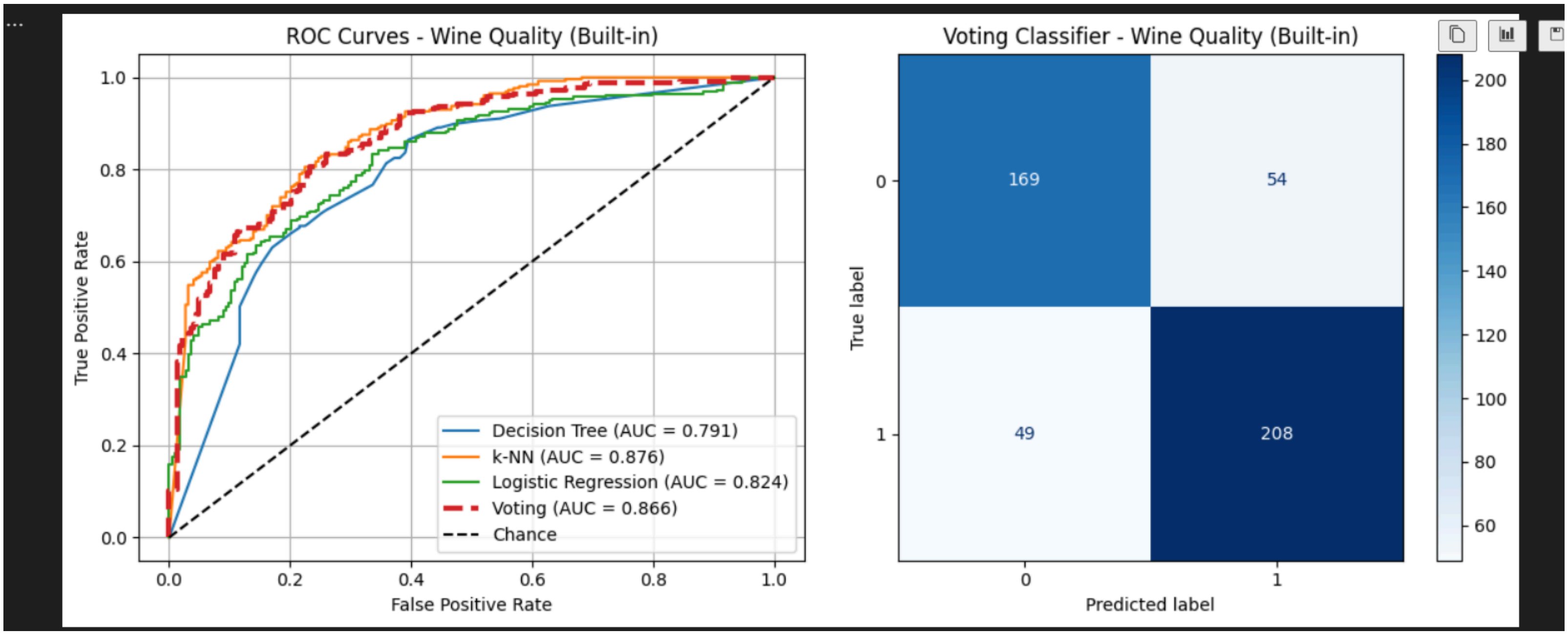
Voting Classifier	Manual	0.7562	0.7713	0.7743	0.7728	0.8664
Voting Classifier	Built-in	0.7854	0.7939	0.8093	0.8015	0.8664

Implementation Comparison

Again, our manual and built-in implementations gave identical ROC AUC scores for individual models, proving our implementation was correct. The voting classifier shows some variation in accuracy metrics (0.7562 vs 0.7854) but identical ROC AUC (0.8664), suggesting differences in probability thresholding rather than ranking performance.

Visualization Analysis





The **ROC curves** show k-NN (AUC = 0.876-0.878) clearly outperforming other models with its curve positioned furthest from the diagonal. The Voting Classifier (AUC = 0.866) performs nearly as well, while Logistic Regression (AUC = 0.824) and Decision Tree (AUC = 0.791) show progressively lower performance.

The **confusion matrices** reveal much more balanced classification compared to HR data. The manual implementation shows 164 true negatives, 199 true positives, with false positives (59) and false negatives (58) being relatively similar. The built-in version shows even better balance (169/208 correct vs 54/49 errors), indicating wine quality is a more balanced classification problem.

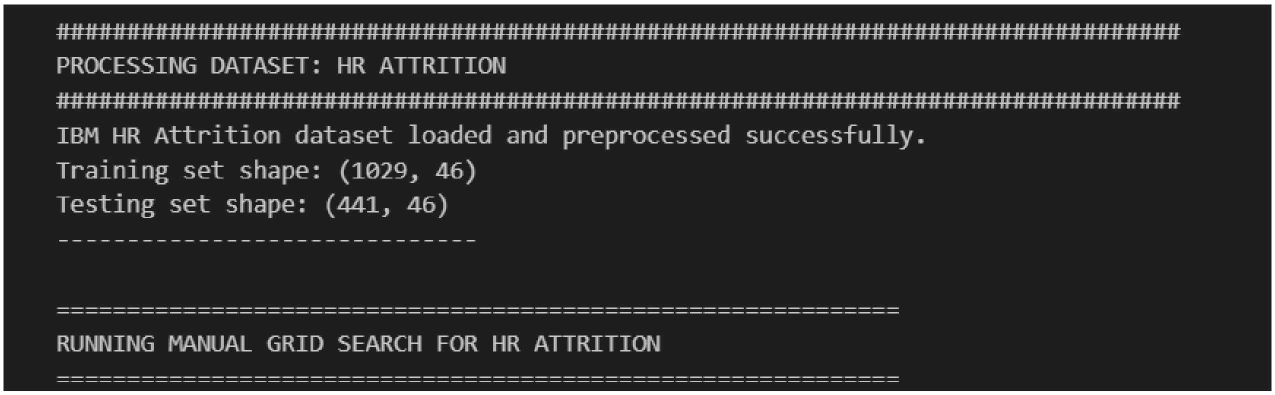
Best Model Analysis

k-NN performed best with ROC AUC around 0.876-0.878. This suggests that wine quality depends on local patterns - wines with similar chemical profiles tend to have similar quality ratings. k-NN excels at capturing these neighborhood relationships in the feature space, where chemical similarity translates to quality similarity.

Overall Comparison

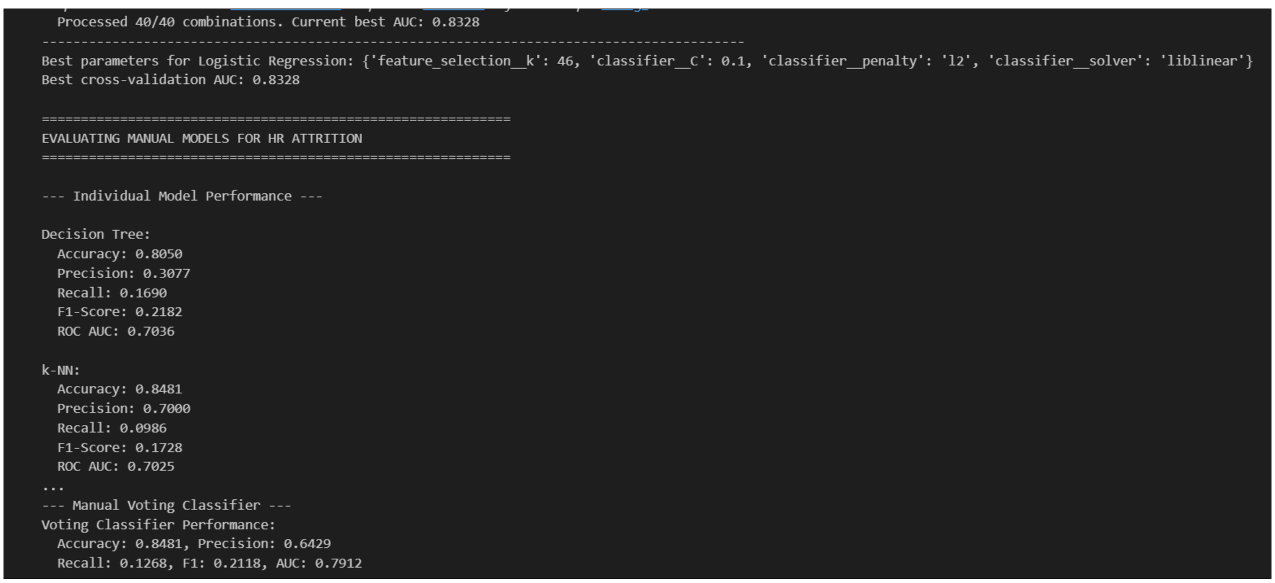
The key insight is that different algorithms worked better for different datasets - Logistic Regression for HR data and k-NN for wine data. This demonstrates why it's important to try multiple algorithms rather than assuming one will always be best. The dataset characteristics (linear vs local patterns, class balance) strongly influence which algorithm performs optimally.

5. 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
============================================================
```



```
  Processed 40/40 combinations. Current best AUC: 0.8328
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 46, 'classifier__C': 0.1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}
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-NN:
  Accuracy: 0.8481
  Precision: 0.7000
  Recall: 0.0986
  F1-Score: 0.1728
  ROC AUC: 0.7025
...
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8481, Precision: 0.6429
  Recall: 0.1268, F1: 0.2118, AUC: 0.7912
```

```
Best params for Logistic Regression: {'classifier__C': 0.1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear', 'feature_selection__k': 46}
Best CV score: 0.8328

============================================================
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-NN:
  Accuracy: 0.8481
  Precision: 0.7000
  Recall: 0.0986
  F1-Score: 0.1728
  ROC AUC: 0.7025

Logistic Regression:
  Accuracy: 0.8798
...
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8503, Precision: 0.6316
  Recall: 0.1690, F1: 0.2667, AUC: 0.7912
```

```
Completed processing for HR Attrition
================================================================================

################################################################################
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 ---
Testing 135 parameter combinations...
  Processed 10/135 combinations. Current best AUC: 0.7796
  Processed 20/135 combinations. Current best AUC: 0.7846
  Processed 30/135 combinations. Current best AUC: 0.7846
  Processed 40/135 combinations. Current best AUC: 0.7850
  Processed 50/135 combinations. Current best AUC: 0.7850
  Processed 60/135 combinations. Current best AUC: 0.7850
  Processed 70/135 combinations. Current best AUC: 0.7850
  Processed 80/135 combinations. Current best AUC: 0.7850
...
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7562, Precision: 0.7713
  Recall: 0.7743, F1: 0.7728, AUC: 0.8664
```

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

--- GridSearchCV for Decision Tree ---
Fitting 5 folds for each of 135 candidates, totalling 675 fits
Best params for Decision Tree: {'classifier__max_depth': 10, 'classifier__min_samples_leaf': 4, 'classifier__min_samples_split': 10, 'feature_selection__k': 5}
Best CV score: 0.7850

--- GridSearchCV for k-NN ---
Fitting 5 folds for each of 60 candidates, totalling 300 fits
Best params for k-NN: {'classifier__metric': 'manhattan', 'classifier__n_neighbors': 11, 'classifier__weights': 'distance', 'feature_selection__k': 5}
Best CV score: 0.8696

--- GridSearchCV for Logistic Regression ---
Fitting 5 folds for each of 30 candidates, totalling 150 fits
Best params for Logistic Regression: {'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear', 'feature_selection__k': 11}
Best CV score: 0.8052

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

--- Individual Model Performance ---
...
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7854, Precision: 0.7939
  Recall: 0.8093, F1: 0.8015, AUC: 0.8664
```

```
...
    Completed processing for Wine Quality
    ================================================================================

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

6. Conclusion

This lab taught us several important lessons about machine learning model selection:

1. **Manual vs. Built-in Implementation:** Both approaches gave identical results, but the manual implementation helped us understand the computational complexity involved. Grid search is basically nested loops testing every parameter combination, which can get expensive quickly.

2. **Algorithm Selection Matters:** No single algorithm dominated both datasets. Logistic Regression worked best for HR attrition while k-NN excelled at wine quality prediction. This reinforces that algorithm choice should be data-driven.

3. **Cross-Validation is Essential:** Using 5-fold cross-validation gave us much more reliable performance estimates than a single train-test split would have.

4. **Class Imbalance Impact:** The HR dataset's severe class imbalance made minority class prediction challenging for all models, while the more balanced Wine Quality dataset allowed for better overall performance.

5. **Practical vs. Educational Trade-offs:** While scikit-learn's GridSearchCV is clearly better for real projects due to speed and reliability, implementing grid search manually was valuable for learning. It's like the difference between using a calculator and doing math by hand - both have their place.

The biggest takeaway is that there's no magic bullet in machine learning. Success comes from systematic experimentation, proper evaluation methodology, and understanding your data well enough to choose appropriate algorithms. The tools make this easier, but the thinking still needs to be done by humans.