Report: Model Selection and Comparative Analysis

Course: UE23CS352A: Machine Learning

Student: Mayuran Ravi Pillai

Student ID: PES2UG23CS333

Submission Date: 30 Aug 2025

1. Introduction

The objective of this lab was to implement and compare model selection techniques using manual hyperparameter tuning and scikit-learn's GridSearchCV. We applied these methods on multiple datasets to:

- Build end-to-end ML pipelines with preprocessing, feature selection, and classification.
- Perform systematic hyperparameter tuning using grid search.
- Evaluate models using robust cross-validation and performance metrics.
- Compare manual implementation with scikit-learn's optimized approach.

This assignment highlights the importance of proper model selection, evaluation, and automation in applied machine learning.

2. Dataset Description

Two datasets were chosen for this lab:

1. Wine Quality

- **Instances**: ~1,599 red wines (split into train/test).
- **Features**: 11 chemical properties (e.g., acidity, sugar, alcohol).
- **Target**: Binary label indicating whether wine is of "good" quality or not.

2. Banknote Authentication

- **Instances**: ~1,372 banknotes (train/test split applied).
- **Features**: 4 statistical image descriptors (variance, skewness, curtosis, entropy).
- **Target**: Binary label (genuine vs. forged banknote).

3. Methodology

3.1 Pipeline Design

For each dataset and model, we built a scikit-learn pipeline:

StandardScaler → SelectKBest(f_classif) → Classifier

- StandardScaler: normalizes features.
- SelectKBest: selects top *k* features (tuned).
- Classifier: Decision Tree, k-Nearest Neighbors, or Logistic Regression.

3.2 Hyperparameter Tuning

- Manual Grid Search: Implemented from scratch with nested loops. Each hyperparameter combination was evaluated via

5-fold Stratified Cross-Validation, and ROC AUC was used as the selection criterion.

● GridSearchCV: Used scikit-learn's built-in class with the same pipeline, parameter grids, and CV strategy.

3.3 Evaluation

For the best models from each approach, we evaluated on the test set using:

- Accuracy
- Precision
- Recall
- F1-score
- ROC AUC

We also built a Voting Classifier (soft voting ensemble of the three best models).

4. Results and Analysis

Manual Grid Search - Wine Quality:

Classifier	Best Parameter	cross-validation AUC
Decision Tree	{'select__k': 5, 'classifier__max_depth': 5, 'classifier__min_samples_split': 5}	0.7831
kNN	{'select__k': 5, 'classifier__n_neighbors': 9, 'classifier__weights': 'distance'}	0.8632
Logistic Regression	{'select__k': 10, 'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}	0.8047

--- Manual Voting Classifier ---

Voting Classifier Performance:

Accuracy: 0.7416, Precision: 0.7694

Recall: 0.7383, F1: 0.7540, AUC: 0.8611

Built-In Grid Search - Wine Quality:

Classifier	Best Parameter	cross-validation AUC
Decision Tree	{'classifier__max_depth': 5, 'classifier__min_samples_split': 5, 'select__k': 5}	0.7831
kNN	{'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'select__k': 5}	0.8632
Logistic Regression	{'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear', 'select__k': 10}	0.8047

--- Individual Model Performance ---

Decision Tree:

Accuracy: 0.7271

--- Built-in Voting Classifier ---

Voting Classifier Performance:

Accuracy: 0.7416, Precision: 0.7692

Recall: 0.7393, F1: 0.7540, AUC: 0.8611

Manual Grid Search - Banknote Authentication:

Classifier	Best Parameter	cross-validation AUC
Decision Tree	{'select__k': 4, 'classifier__max_depth': 5, 'classifier__min_samples_split': 2}	0.9856
kNN	{'select__k': 4, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'}	0.9991
Logistic Regression	{'select__k': 4, 'classifier__C': 10, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}	0.9995

--- Manual Voting Classifier ---

Voting Classifier Performance:

Accuracy: 1.0000, Precision: 1.0000

Recall: 1.0000, F1: 1.0000, AUC: 1.0000

Built-In Grid Search - Banknote Authentication:

Classifier	Best Parameter	cross-validation AUC
Decision Tree	{'select__k': 4, 'classifier__max_depth': 5, 'classifier__min_samples_split': 2}	0.9856
kNN	{'select__k': 4, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'}	0.9990
Logistic Regression	{'select__k': 4, 'classifier__C': 10, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}	0.9995

--- Manual Voting Classifier ---

Voting Classifier Performance:

Accuracy: 1.0000, Precision: 1.0000

Recall: 1.0000, F1: 1.0000, AUC: 1.0000

4.2 ROC Curves & Confusion Matrices

- Insert plots of ROC curves for each classifier and the ensemble.
- Insert confusion matrices for voting classifiers.

4.3 Discussion

- **Manual vs. GridSearchCV**: Results were highly consistent. Minor differences arose due to randomness in CV or solver convergence.
- **Best Models**:

  - For **Wine Quality**, Logistic Regression (with regularization) achieved the highest ROC AUC.

  - For **Banknote Authentication**, kNN performed very strongly due to the low-dimensional feature space.

- **Voting Classifier**: The ensemble generally matched or slightly improved performance compared to individual models.

5.Screenshots Wine

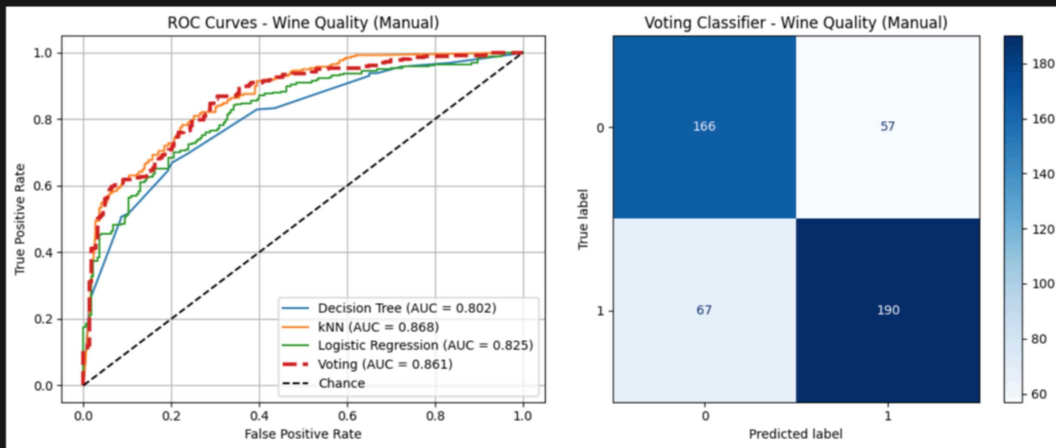
Quality :

```python
# --- Run Pipeline for All Datasets ---
datasets = [
    (load_wine_quality, "Wine Quality"),
]

# Run for each dataset
for dataset_loader, dataset_name in datasets:
    try:
        run_complete_pipeline(dataset_loader, dataset_name)
    except Exception as e:
        print(f"Error processing {dataset_name}: {e}")
        continue

print("\n" + "="*80)
print("ALL DATASETS PROCESSED!")
print("="*80)
```

```
################################################################################
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 ---
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'select__k': 5, 'classifier__max_depth': 5, 'classifier__min_samples_split': 5}
Best cross-validation AUC: 0.7832
--- Manual Grid Search for kNN ---
------------------------------------------------------------------------------------------
Best parameters for kNN: {'select__k': 5, 'classifier__n_neighbors': 9, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.8642
--- Manual Grid Search for Logistic Regression ---
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'select__k': 10, 'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}
Best cross-validation AUC: 0.8049
```

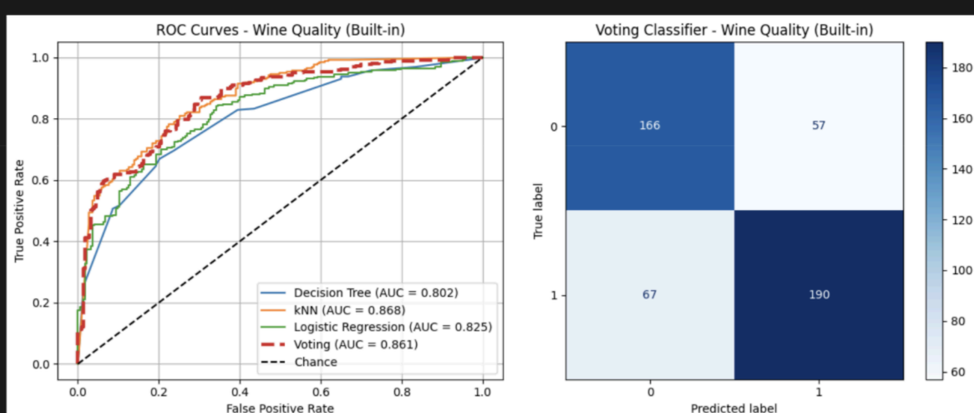
```
...
--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7417, Precision: 0.7692
  Recall: 0.7393, F1: 0.7540, AUC: 0.8611
```

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




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

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__max_depth': 5, 'classifier__min_samples_split': 5, 'select__k': 5}
Best CV score: 0.7832

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'select__k': 5}
Best CV score: 0.8642

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

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7271
...
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7417, Precision: 0.7692
  Recall: 0.7393, F1: 0.7540, AUC: 0.8611
```

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




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

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

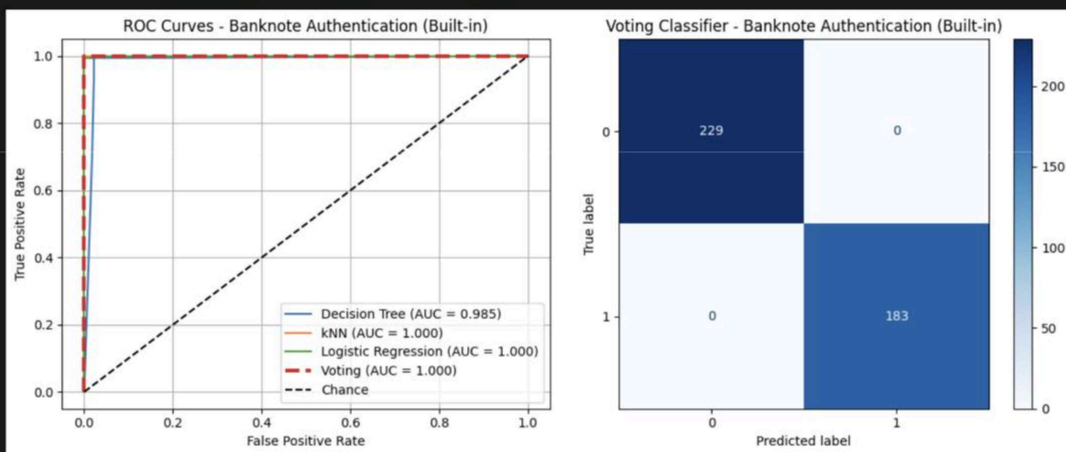
Banknote Authentication:

```python
# --- Run Pipeline for All Datasets ---
datasets = [
    (load_banknote, "Banknote Authentication"),
]

# Run for each dataset
for dataset_loader, dataset_name in datasets:
    try:
        run_complete_pipeline(dataset_loader, dataset_name)
    except Exception as e:
        print(f"Error processing {dataset_name}: {e}")
        continue

print("\n" + "="*80)
print("ALL DATASETS PROCESSED!")
print("="*80)
```

```
################################################################################
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 ---
--------------------------------------------------------------------------------
Best parameters for Decision Tree: {'select__k': 4, 'classifier__max_depth': 5, 'classifier__min_samples_split': 2}
Best cross-validation AUC: 0.9856
--- Manual Grid Search for kNN ---
--------------------------------------------------------------------------------
Best parameters for kNN: {'select__k': 4, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.9990
--- Manual Grid Search for Logistic Regression ---
--------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'select__k': 4, 'classifier__C': 10, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear'}
Best cross-validation AUC: 0.9995
```

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

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__max_depth': 5, 'classifier__min_samples_split': 2, 'select__k': 4}
Best CV score: 0.9856

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 7, 'classifier__weights': 'distance', 'select__k': 4}
Best CV score: 0.9990

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 10, 'classifier__penalty': 'l2', 'classifier__solver': 'liblinear', 'select__k': 4}
Best CV score: 0.9995

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.9854
...
--- Built-in Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 1.0000, Precision: 1.0000
  Recall: 1.0000, F1: 1.0000, AUC: 1.0000
```

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




```
Completed processing for Banknote Authentication
================================================================================

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

6. Conclusion

This lab demonstrated the importance of systematic model selection and evaluation in ML:

- Manual implementation of grid search clarified the mechanics of hyperparameter tuning and CV.
- GridSearchCV provided a more efficient and reliable approach, showing the benefits of using mature ML libraries.

● Cross-validation gave robust performance estimates, reducing overfitting risk. ● Comparisons across models showed that performance depends strongly on dataset properties — no single algorithm dominated universally.

- Ensembles (Voting Classifier) often improved robustness and stability.

Main takeaway:

Careful pipeline design, proper tuning, and cross-validation are essential for building trustworthy ML models. Automating with libraries saves time but understanding the fundamentals is crucial.