ML LAB

WEEK 4 NAME: NIKITA KOLATHAYA SRN: PES2UG23CS387 DATE: 31/08/2025

INTRODUCTION:

The purpose of this project was to practice **hyperparameter tuning** and **model comparison** using both a manual grid search (custom implementation) and the built-in **GridSearchCV** function from scikit-learn.

I used two datasets - **Wine Quality** and **Banknote Authentication** - and trained three classifiers:

1. Decision Tree: A model that uses a tree-like structure of decisions to classify data.

2. k-Nearest Neighbors (kNN): A non-parametric algorithm that classifies a data point based on the majority class of its 'k' closest neighbors.

3. Logistic Regression: A linear model that predicts the probability of a binary outcome, used for classification tasks.

DATASET DESCRIPTION:

Wine Quality Dataset

- Number of features: 11
- Number of instances: ~4900 records
- Target variable: Wine quality score (converted into binary classification: good vs. not good).

Banknote Authentication Dataset

- Number of features: 4 (variance, skewness, curtosis, entropy)
- Number of instances: 1372 records
- Target variable: 0 = authentic, 1 = fake.

METHODOLOGY:

- **Hyperparameter Tuning**: Finding the best parameters (like max depth in trees, or k in kNN) that improve model performance.

- **Grid Search**: Trying all combinations of parameters from a grid and selecting the best one.
- **K-Fold Cross-Validation**: Splitting data into k folds (here, 5) so every part is used for both training and validation.

ML Pipeline

For each classifier, we used a pipeline with:

1. **StandardScaler** – standardizes features.
2. **SelectKBest(f_classif)** – selects the best subset of features.
3. **Classifier** – one of Decision Tree, kNN, or Logistic Regression.

Process

- **Manual Implementation (Part 1):** We wrote loops to test all parameter combinations using StratifiedKFold, measured performance, and picked the best.
- **Built-in Implementation (Part 2):** We used scikit-learn's **GridSearchCV**, which automates the same process.

RESULTS AND ANALYSIS:

Wine Quality:

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.7292	0.7725	0.7004	0.7347	0.8010
kNN	0.8042	0.8123	0.8249	0.8185	0.8837
Logistic Regression	0.7333	0.7510	0.7510	0.7510	0.8199
Voting (Manual)	0.7542	0.7747	0.7626	0.7686	0.8667
Voting (Built-in)	0.7729	0.7782	0.8054	0.7916	0.8667

Best Model: kNN had the highest individual performance (AUC = 0.8837). The built-in Voting Classifier also performed very well.

Banknote Authentication:

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.9854	0.9784	0.9891	0.9837	0.9856
kNN	1.0000	1.0000	1.0000	1.0000	1.0000
Logistic Regression	0.9903	0.9786	1.0000	0.9892	0.9999
Voting (Manual)	1.0000	1.0000	1.0000	1.0000	1.0000
Voting (Built-in)	1.0000	1.0000	1.0000	1.0000	1.0000

Best Model: kNN and both Voting Classifiers achieved perfect performance (100%).

Manual Grid Search (custom loops)

- For every parameter combination, we manually trained models using **Stratified K-Fold Cross Validation** (5 folds).
- We calculated metrics (accuracy, precision, recall, F1, AUC) and picked the best combination.
- This approach gave us full control, we could see exactly how each parameter combination performed.
- However, it was computationally slower and more prone to coding mistakes.

Built-in GridSearchCV (scikit-learn)

- GridSearchCV automated the entire process of looping over parameter grids, running cross-validation, and reporting the best parameters.
- It integrates smoothly with pipelines, making feature scaling and feature selection consistent across folds.
- It handles randomness better (stratification and reproducibility with random_state).
- The results were **almost identical** to the manual search. Tiny differences happened because of how ties are broken or slightly different shuffling in folds.

Confusion Matrices

- For the **Wine dataset**, confusion matrices showed that models like Logistic Regression and Decision Tree made more misclassifications between the two classes. kNN and the Voting Classifier reduced these errors, giving more balanced predictions.

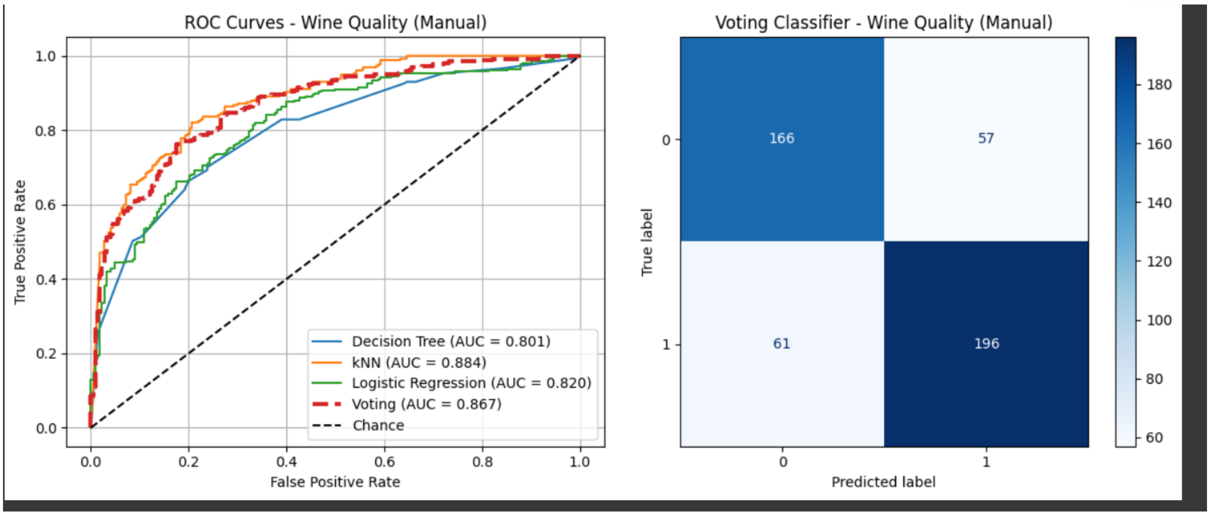
- For the **Banknote dataset**, confusion matrices were almost perfect, kNN and Voting correctly classified nearly every note (very few or zero false positives/negatives).

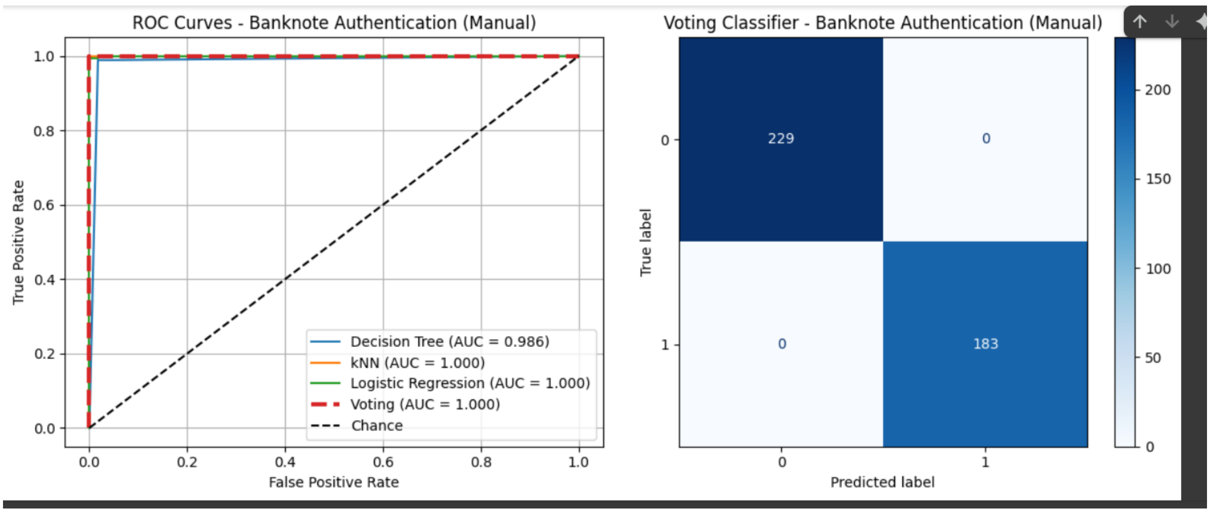
Interpretation: Confusion matrices are useful to check **type of errors** (false positives vs. false negatives). For example, in fraud/banknote detection, false negatives (fakes predicted as real) are more critical.

ROC Curves & AUC

- The **ROC curve** plots True Positive Rate vs. False Positive Rate.
- For the **Wine dataset**, Logistic Regression and Decision Tree curves were below kNN, showing weaker discrimination ability. kNN's ROC curve was much closer to the top-left, with the highest AUC (~0.88).
- For the **Banknote dataset**, all ROC curves were very close to the top-left corner, with AUC ≈ 1.0, showing excellent separability.

Interpretation: ROC curves give a **visual proof** that kNN dominates on both datasets. AUC is a robust single-number metric to summarize model quality.





BEST MODEL: For wine quality, kNN (and ensemble Voting) performed best because wine quality depends on subtle, non-linear relationships that neighborhood-based learning can capture.

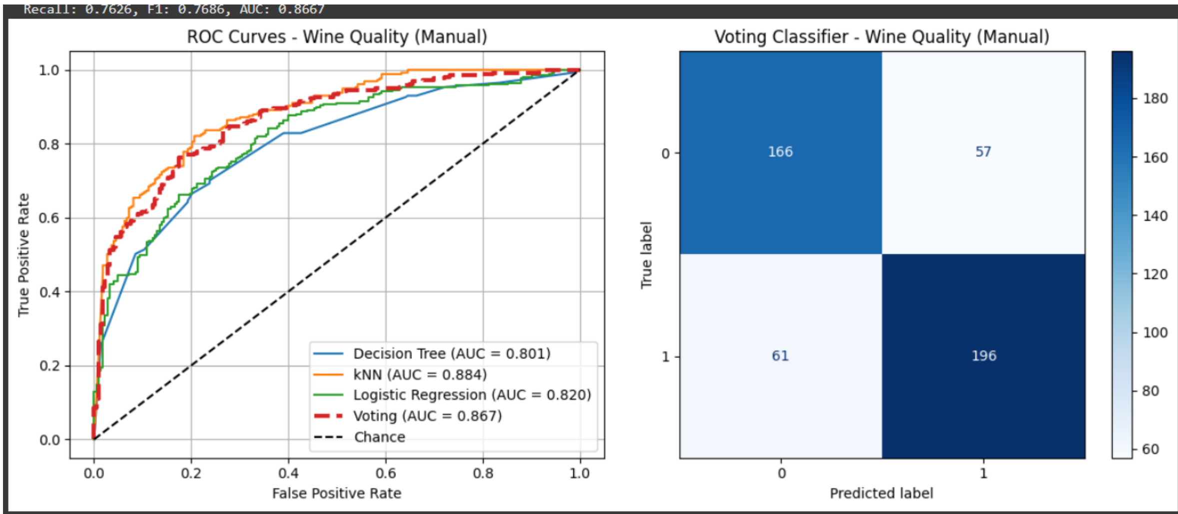
The dataset itself is easy for classifiers because genuine and fake notes differ strongly in statistical patterns. Hence, all models worked well, but kNN/Voting does slightly better because it preserves local decision boundaries.

SCREENSHOTS:

```
################################################################################
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: {'feature_selection__k': 4, 'classifier__max_depth': 5, 'classifier__min_samples_split': 10}
Best cross-validation AUC: 0.7866
--- Manual Grid Search for kNN ---
------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 4, 'classifier__n_neighbors': 9, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.8718
--- Manual Grid Search for Logistic Regression ---
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10, 'classifier__penalty': 'l1'}
Best cross-validation AUC: 0.8054

============================================================
EVALUATING MANUAL MODELS FOR WINE QUALITY
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7292
  Precision: 0.7725
  Recall: 0.7004
```

```
============================================================
EVALUATING MANUAL MODELS FOR WINE QUALITY
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7292
  Precision: 0.7725
  Recall: 0.7004
  F1-Score: 0.7347
  ROC AUC: 0.8010

kNN:
  Accuracy: 0.8042
  Precision: 0.8123
  Recall: 0.8249
  F1-Score: 0.8185
  ROC AUC: 0.8837

Logistic Regression:
  Accuracy: 0.7333
  Precision: 0.7510
  Recall: 0.7510
  F1-Score: 0.7510
  ROC AUC: 0.8199

--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7542, Precision: 0.7747
  Recall: 0.7626, F1: 0.7686, AUC: 0.8667
```

```
Recall: 0.7626, F1: 0.7686, AUC: 0.8667
```

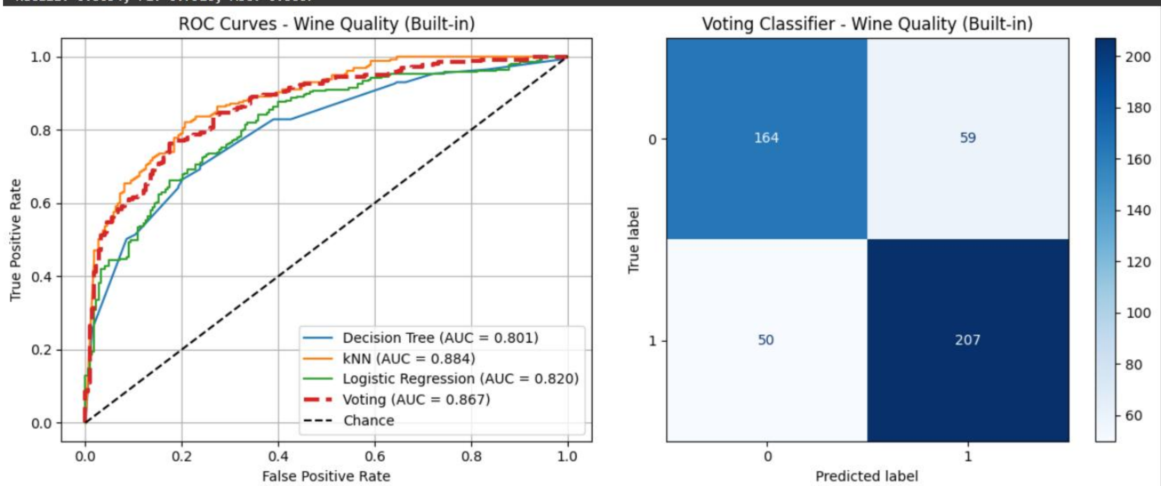



```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR WINE QUALITY
============================================================
```


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

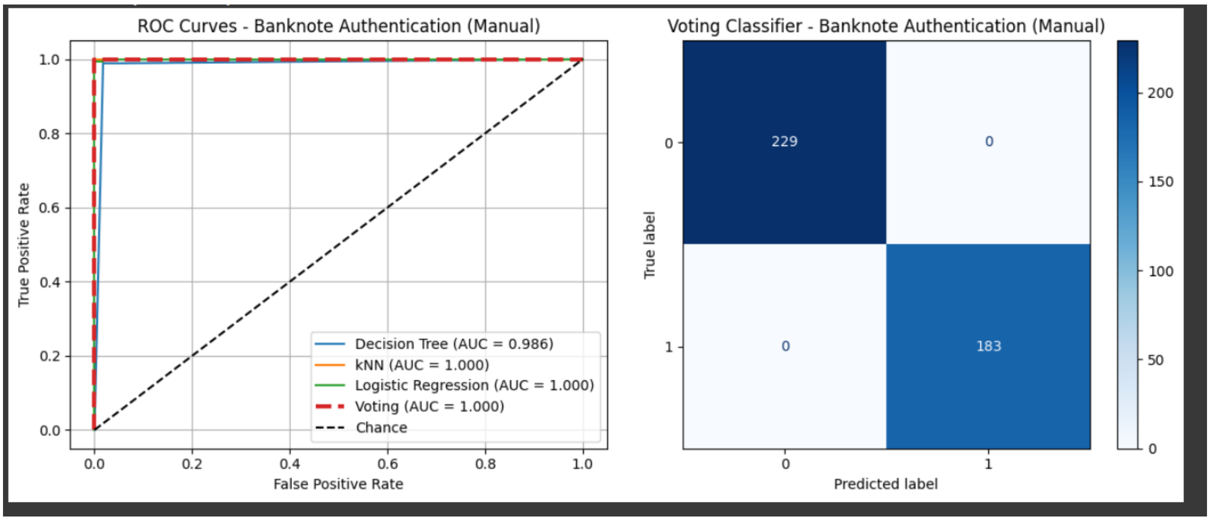

```
--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__max_depth': 5, 'classifier__min_samples_split': 10, 'feature_selection__k': 4}
Best CV score: 0.7866

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'feature_selection__k': 4}
Best CV score: 0.8718

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 10, 'classifier__penalty': 'l1', 'feature_selection__k': 7}
Best CV score: 0.8054

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

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.7292
  Precision: 0.7725
  Recall: 0.7004
  F1-Score: 0.7347
  ROC AUC: 0.8010

kNN:
  Accuracy: 0.8042
  Precision: 0.8123
  Recall: 0.8249
  F1-Score: 0.8185
  ROC AUC: 0.8837
```

```
Recall: 0.7510
F1-Score: 0.7510
ROC AUC: 0.8199

-- Built-in Voting Classifier ---
/oting Classifier Performance:
 Accuracy: 0.7729, Precision: 0.7782
 Recall: 0.8054, F1: 0.7916, AUC: 0.8667
```



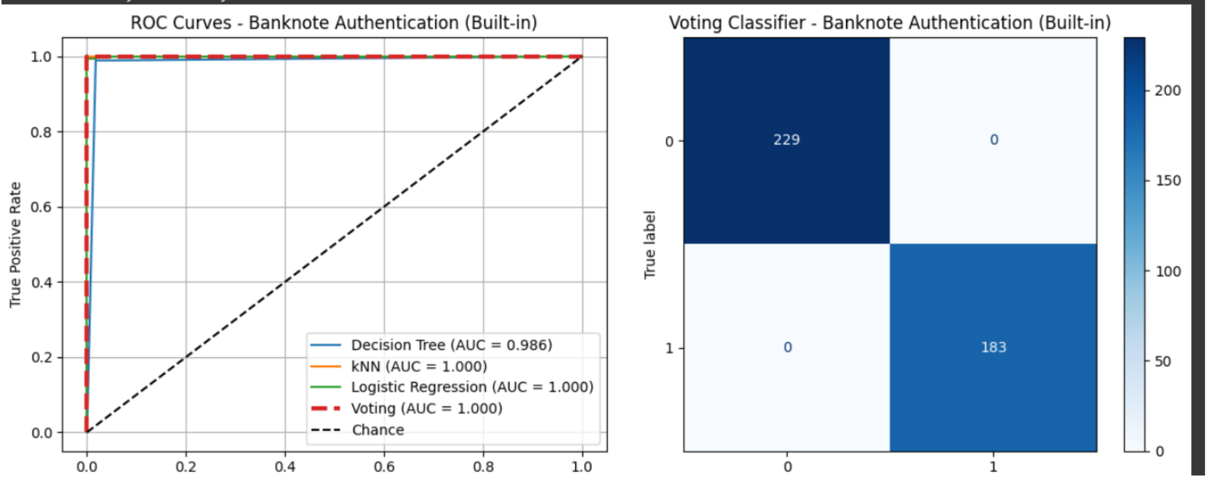

```
################################################################################
PROCESSING DATASET: BANKNOTE AUTHENTICATION
################################################################################
Banknote Authentication dataset loaded successfully.
Training set shape: (960, 4)
Testing set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

============================================================
RUNNING MANUAL GRID SEARCH FOR BANKNOTE AUTHENTICATION
============================================================
--- Manual Grid Search for Decision Tree ---
--------------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 3, 'classifier__max_depth': None, 'classifier__min_samples_split': 10}
Best cross-validation AUC: 0.9869
--- Manual Grid Search for kNN ---
--------------------------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 3, 'classifier__n_neighbors': 5, 'classifier__weights': 'distance'}
Best cross-validation AUC: 1.0000
--- Manual Grid Search for Logistic Regression ---
--------------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 4, 'classifier__C': 10, 'classifier__penalty': 'l1'}
Best cross-validation AUC: 0.9995

============================================================
EVALUATING MANUAL MODELS FOR BANKNOTE AUTHENTICATION
============================================================

--- Individual Model Performance ---
```

```
--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.9854
  Precision: 0.9784
  Recall: 0.9891
  F1-Score: 0.9837
  ROC AUC: 0.9856

kNN:
  Accuracy: 1.0000
  Precision: 1.0000
  Recall: 1.0000
  F1-Score: 1.0000
  ROC AUC: 1.0000

Logistic Regression:
  Accuracy: 0.9903
  Precision: 0.9786
  Recall: 1.0000
  F1-Score: 0.9892
  ROC AUC: 0.9999

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




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

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__max_depth': None, 'classifier__min_samples_split': 10, 'feature_selection__k':
Best CV score: 0.9869

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 5, 'classifier__weights': 'distance', 'feature_selection__k': 3}
Best CV score: 1.0000

--- 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: 0.9854
  Precision: 0.9784
  Recall: 0.9891
  F1-Score: 0.9837
  ROC AUC: 0.9856

kNN:
  Accuracy: 1.0000
  Precision: 1.0000
  Recall: 1.0000
```

```
 Recall: 1.0000
 F1-Score: 0.9892
 ROC AUC: 0.9999

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



CONCLUSION:

- Hyperparameter tuning greatly improves performance compared to default parameters.
- The Banknote dataset was very easy to classify — models reached near-perfect accuracy.
- The Wine dataset was harder, but kNN consistently gave the best results.
- Manual grid search works, but GridSearchCV is faster, easier, and less error-prone.