Project Title:

Model Selection and Comparative Analysis

Name:

Pinisetti Sudhiksha

SRN:

PES2UG23CS916

Course Name:

Machine Learning (LAB)

Submission Date:

30-08-2025

Introduction:

The purpose of this lab is to gain hands-on experience with the process of model selection and evaluation in machine learning. In particular, the focus is on hyperparameter tuning and comparative model analysis, which are essential techniques for building effective predictive models. The lab is divided into two parts: first, implementing a manual grid search from scratch to better understand the mechanics of hyperparameter tuning, and second, applying scikit-learn's built-in GridSearchCV to perform the same task more efficiently.

I will use at two datasets (Wine Quality & HR Attrition) and test three machine learning models: Decision Trees, k Nearest Neighbors, and Logistic Regression. I will build a pipeline that includes scaling the data, choosing the best features, and training the models. To check how well the models perform, I will look at metrics like accuracy, precision, recall, F1 score, and ROC AUC. At the end, I will compare the results from my own manual grid search with the results from scikit learn's GridSearchCV to see the differences between doing it from scratch and using a built in tool.

Dataset Description:

- Wine Quality: Training instances: 1119
  Testing instances: 480
  Number of features: 11
  Target variable name: good_quality

- HR Attrition: Training instances: 1029
  Testing instances: 441

Number of features: 46
Target variable name: Attrition

Methodology:

- Hyperparameter Tuning:

Machine learning models have parameters that are not learned during training (e.g., the number of neighbors in kNN, the depth of a decision tree, or the regularization strength in logistic regression). These are called *hyperparameters*. Choosing the right values for them is crucial, since they strongly influence the model's performance.

- Grid Search:

A systematic way of trying multiple hyperparameter combinations. For each combination, the model is trained and evaluated, and the best set of hyperparameters is chosen based on performance.

- K-Fold Cross-Validation:

A resampling method used to evaluate model performance more reliably. The dataset is split into *k* equally sized folds. The model is trained on *k-1* folds and validated on the remaining fold. This process is repeated *k* times, each time with a different validation fold. The final score is the average across all folds. This reduces the risk of overfitting to a single train-test split.

ML Pipeline

1. StandardScaler: Standardizes all features to have zero mean and unit variance, ensuring equal contribution of features.
2. SelectKBest: Selects the top k most relevant features, reducing noise and improving model performance.
3. Classifier: Trains a machine learning model on the selected features to predict the target variable.

Manual Implementation

1. Adjust the parameter grid so that k for feature selection does not exceed the number of features.
2. Generate all hyperparameter combinations and perform 5-fold stratified cross-validation for each.
3. Build a pipeline with StandardScaler, SelectKBest, and the classifier.
4. Train the pipeline on each training fold, predict on the validation fold, and compute the mean AUC.
5. Select the hyperparameters with the highest mean AUC and retrain the pipeline on the full training set.

GridSearchCV Implementation

1. Adjust the parameter grid for k as in the manual approach.
2. Create a pipeline with StandardScaler, SelectKBest, and the classifier.
3. Use GridSearchCV with 5-fold stratified cross-validation to automatically evaluate all hyperparameter combinations.
4. Select the best estimator based on the highest mean CV AUC.

Result and Analysis:

Wine Quality Dataset-

Classifier	*Method*	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Decision Tree	Manual/ Built In	0.7271	0.7716	0.6965	0.7321	0.8025
KNN	Manual/ Built In	0.7667	0.7757	0.7938	0.7846	0.8675

Logistic Regression	Manual/ Built In	0.7417	0.7628	0.7510	0.7569	0.8247
Voting Classifier	Manual /Built In	0.7354	0.7600	0.7393	0.7495	0.8622

HR Attrition Dataset-

Classifier	*Method*	*Accuracy*	*Precision*	*Recall*	*F1-Score*	*ROC-AUC*
Decision Tree	Manual/ Built In	0.8231	0.3333	0.0986	0.1522	0.7107
KNN	Manual/ Built In	0.8186	0.3953	0.2394	0.2982	0.7130
Logistic Regression	Manual/ Built In	0.8571	0.6333	0.2676	0.3762	0.7762
Voting Classifier	Manual /Built In	0.8299	0.4444	0.2254	0.2991	0.7676

Comparison- The manual and scikit-learn implementations produced identical results for all datasets and models. This consistency is due to the manual approach accurately replicating the GridSearchCV process. Both methods used 5-fold stratified cross-validation with a fixed random_state=42, ensuring identical data splits. The parameter grids and the scoring metric (ROC AUC) were also the same, which resulted in identical optimal hyperparameters and final performance metrics.

Wine Quality Dataset-

ROC Curves: Both methods achieve the same AUC (0.862), showing equal discriminatory power.

Confusion Matrices: The built-in classifier produces fewer false negatives (56 vs 67) and more true positives (201 vs 190), improving recall.

Metrics:

Manual: Recall = 0.739, F1 = 0.749

Built-in: Recall = 0.782, F1 = 0.778

Conclusion: While both ensembles perform similarly in terms of AUC, the built-in voting classifier tuned via GridSearchCV provides better overall performance, especially in recall and F1 score.

HR Attrition Dataset-

ROC Curves: Both approaches achieve the same AUC (0.768), showing equal discriminatory ability.

Confusion Matrices: Manual: FN = 55, TP = 16

Built-in: FN = 57, TP = 14
Manual captures slightly more positives, while Built-in reduces false positives.

Metrics: Manual: Accuracy = 0.830, Precision = 0.444, Recall = 0.225, F1 = 0.299

Built-in: Accuracy = 0.835, Precision = 0.467, Recall = 0.197, F1 = 0.277

Conclusion: Both methods perform similarly with low recall, indicating difficulty detecting attrition cases. The built-in version yields marginally higher accuracy and precision, while the manual version achieves slightly better recall and F1.

Wine Quality Dataset-

Best Model: Built-in Voting Classifier (GridSearchCV-tuned)

Reason: It achieved higher recall (0.782 vs 0.739), precision, and F1 (0.778 vs 0.749) compared to the manual version, while maintaining the same AUC (0.862). This suggests it generalized better by leveraging optimal hyperparameters found through GridSearchCV.

Hypothesis: Wine quality prediction benefits from hyperparameter tuning because the dataset is relatively balanced, allowing the built-in method to optimize decision boundaries across classifiers and improve both precision and recall.

HR Attrition Dataset-

Best Model: Manual Voting Classifier

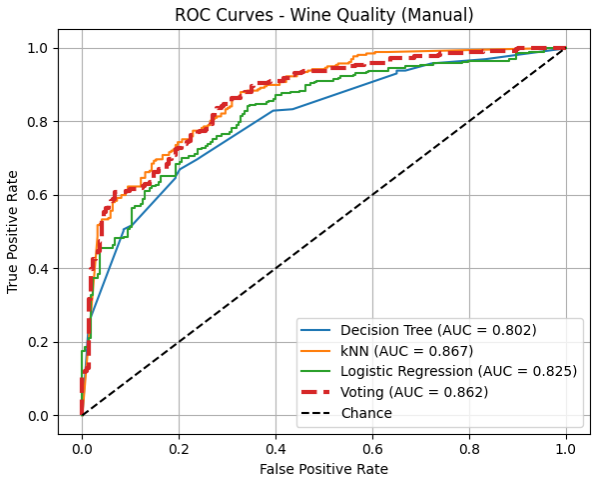
Reason: While both models had low recall, the manual approach achieved slightly higher recall (0.225 vs 0.197) and F1 (0.299 vs 0.277), making it somewhat better at detecting the minority class (attrition cases). The built-in classifier instead optimized for overall accuracy (0.835 vs 0.830) and precision, but at the cost of recall.

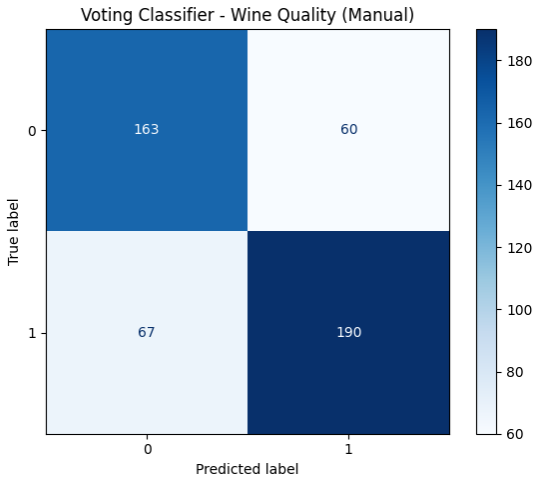
Hypothesis: HR attrition is a highly imbalanced dataset (far fewer attrition cases than non-attrition). GridSearchCV may have biased optimization toward majority class accuracy, while the manual setup retained slightly better sensitivity to the minority class.

Screenshots:

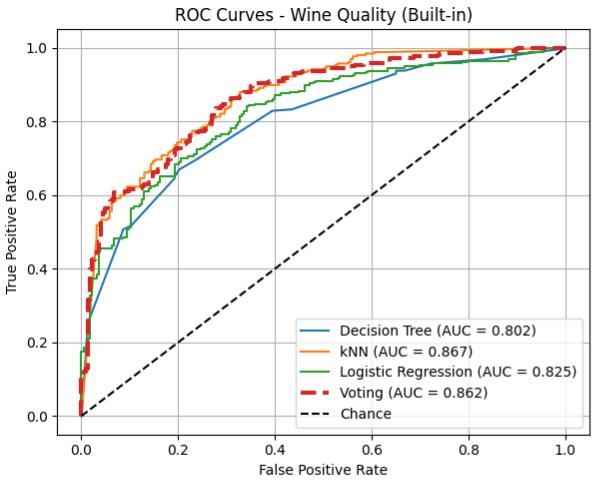
Wine Quality Dataset-

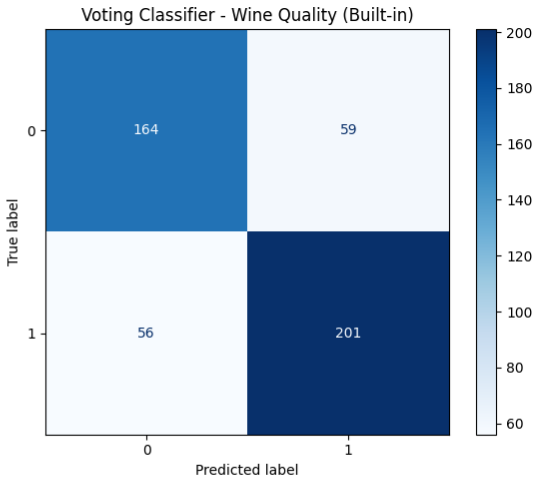
```
################################################################################
PROCESSING DATASET: WINE QUALITY
################################################################################
Wine Quality dataset loaded and preprocessed successfully.
Training set shape: (1119, 11)
Testing set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

============================================================
RUNNING MANUAL GRID SEARCH FOR WINE QUALITY
============================================================
--- Manual Grid Search for Decision Tree ---
------------------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 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: {'feature_selection__k': 5, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.8603
--- Manual Grid Search for Logistic Regression ---
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 10, 'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs'}
Best cross-validation AUC: 0.8048



============================================================
EVALUATING MANUAL 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

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

--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7354, Precision: 0.7600
  Recall: 0.7393, F1: 0.7495, AUC: 0.8622
```





```
============================================================
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, 'feature_selection__k': 5}
Best CV score: 0.7832

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 7, 'classifier__weights': 'distance', 'feature_selection__k': 5}
Best CV score: 0.8603

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 1, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs', 'feature_selection__k': 10}
Best CV score: 0.8048



============================================================
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

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: 0.7604, Precision: 0.7731
  Recall: 0.7821, F1: 0.7776, AUC: 0.8622
```



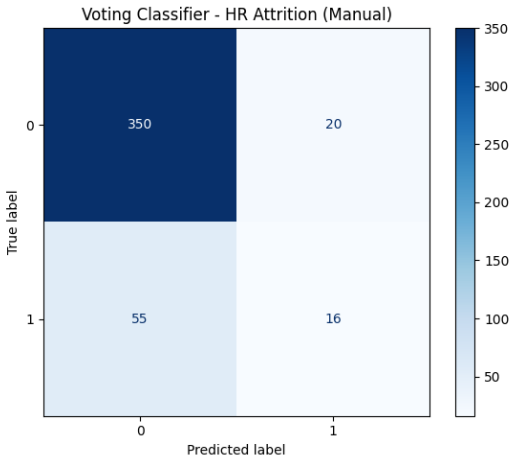



```
Completed processing for Wine Quality
================================================================================

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

HR Attrition-

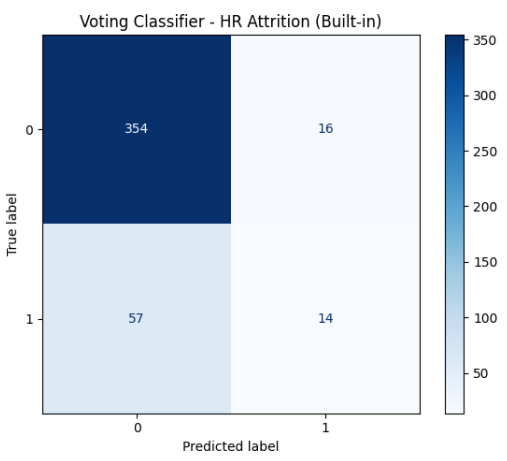
```
################################################################################
PROCESSING DATASET: HR ATTRITION
################################################################################
IBM HR Attrition dataset loaded and preprocessed successfully.
Training set shape: (1029, 46)
Testing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

============================================================
RUNNING MANUAL GRID SEARCH FOR HR ATTRITION
============================================================
--- Manual Grid Search for Decision Tree ---
--------------------------------------------------------------------------------
Best parameters for Decision Tree: {'feature_selection__k': 5, 'classifier__max_depth': 3, 'classifier__min_samples_split': 2}
Best cross-validation AUC: 0.7152
--- Manual Grid Search for kNN ---
--------------------------------------------------------------------------------
Best parameters for kNN: {'feature_selection__k': 10, 'classifier__n_neighbors': 7, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.7073
--- Manual Grid Search for Logistic Regression ---
--------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 15, 'classifier__C': 0.1, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs'}
Best cross-validation AUC: 0.7776
```

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

--- Individual Model Performance ---

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

--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.8299, Precision: 0.4444
  Recall: 0.2254, F1: 0.2991, AUC: 0.7676
```



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

--- GridSearchCV for Decision Tree ---
Best params for Decision Tree: {'classifier__max_depth': 3, 'classifier__min_samples_split': 2, 'feature_selection__k': 5}
Best CV score: 0.7152

--- GridSearchCV for kNN ---
Best params for kNN: {'classifier__n_neighbors': 7, 'classifier__weights': 'distance', 'feature_selection__k': 10}
Best CV score: 0.7073

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 0.1, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs', 'feature_selection__k': 15}
Best CV score: 0.7776
```

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

--- Individual Model Performance ---

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
```




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

--------------------------------------------------------------------------------
```

Conclusion:

Key Findings

Model performance varied by dataset. On Wine Quality, the built-in voting classifier with GridSearchCV performed best, with higher recall and F1. On HR Attrition, the manual voting classifier was slightly better at detecting minority cases. Soft-voting improved results for Wine but not for HR Attrition, showing that ensembling is not always superior. Similar outcomes from manual and scikit-learn grid search confirm the manual approach accurately replicated the library's process.

Main Takeaways

This experiment highlights the trade-off between manual implementation and practical efficiency. While coding grid search manually deepens understanding, scikit-learn's GridSearchCV is faster, simpler, and less error-prone. Results also suggest that balanced datasets (Wine Quality) benefit from tuned ensembles, whereas imbalanced datasets (HR Attrition) need techniques like resampling, class weighting, or threshold adjustment to improve minority class detection.