Week 4 Lab Report – Model Selection & Comparative Analysis

Project Title: Hyperparameter Tuning and Model Comparison for HR Attrition Dataset

Name: _____Vaishnavi Narepalepu______

Student ID: ____PES2UG23CS367_____
Course: ______Machine Learning – UE23CS352A______

Submission Date: _______31-08-25________

---

1. Introduction

The purpose of this project is to explore predictive modeling on the HR Attrition dataset by applying hyperparameter tuning, manual grid search, and scikit-learn's GridSearchCV. The main objective is to identify the best performing classification model that predicts whether an employee is likely to leave the company (Attrition = Yes/No).

Tasks performed include:

- **Implementing manual hyperparameter tuning with k-fold cross-validation.**
- **Using GridSearchCV for automated tuning.**
- **Comparing the performance of multiple classifiers.**
- **Evaluating models with Accuracy, Precision, Recall, F1-Score, and ROC AUC.**

---

2. Dataset – HR EMPLOYEE ATTRITION

- **Source:** IBM HR Analytics Employee Attrition Dataset
- **Instances:** 1470

- **Features:** 35 (after encoding categorical variables)

- **Target:**
- **Yes** → Employee left the company
- **No** → Employee stayed with the company

- **Preprocessing/Notes:** The dataset includes demographic details (Age, Gender, MaritalStatus), job-related features (JobRole, YearsAtCompany, JobLevel), and satisfaction/performance measures (JobSatisfaction, WorkLifeBalance).

---

3. Methodology

3.1 Pipeline

All models were trained inside a 3-step Pipeline to avoid leakage:
`StandardScaler → SelectKBest(f_classif) → Classifier`

Pipeline used in both manual and scikit-learn approaches:

1. **StandardScaler:** Standardize numeric features.
2. **SelectKBest:** Feature selection based on statistical tests.
3. **Classifier:** K-Nearest Neighbors (KNN) and Logistic Regression

3.2 Hyperparameter Tuning

- Process of selecting the best parameters (e.g., k in KNN, C in Logistic Regression) to maximize performance.
- **Grid Search:** Exhaustive search over a predefined set of parameters.
- **K-Fold Cross-Validation:** The dataset is split into *k* folds, where each fold is used once as a validation set

3.3 Models & Parameter Grids

Part 1 (Manual Implementation):

- Defined parameter grids.
- Iterated through combinations.
- Performed k-fold cross-validation manually.
- Selected best parameters.

Part 2 (Scikit-learn GridSearchCV):

- Used GridSearchCV with pipelines.
- Automated parameter search.
- Evaluated on test set.

---

4. Results & Analysis

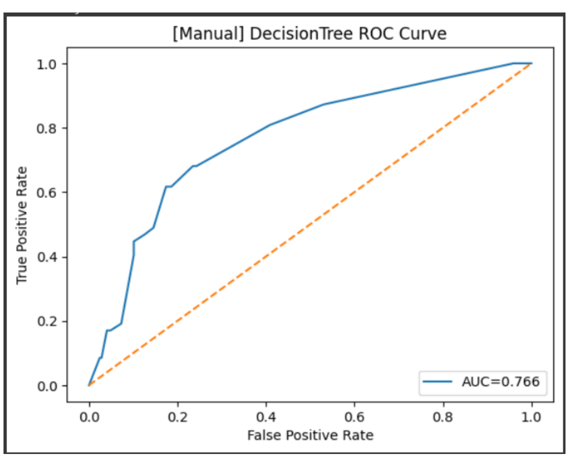
For each dataset, report the best model per algorithm for **both** Part 1 and Part 2. Include metrics and best params.

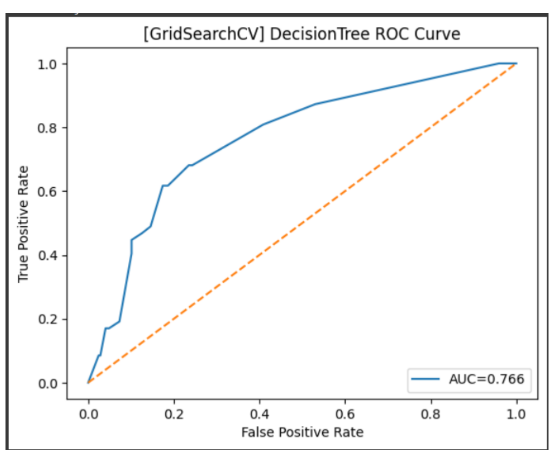
4.1 Performance Tables

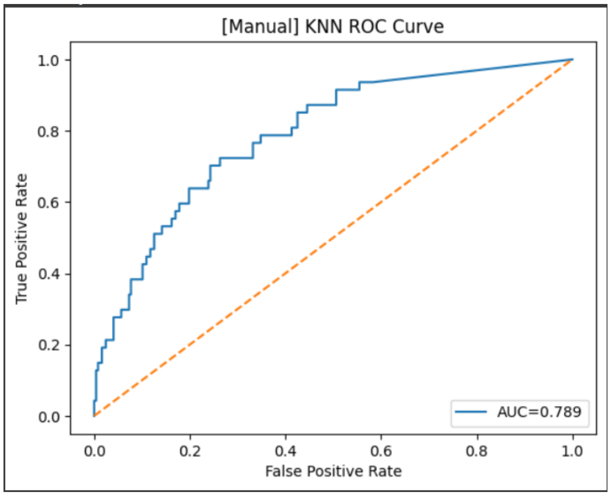
index	Dataset	Classifier	Implementation	Accuracy	Precision	Recall	F1	ROC_AUC
0	WA_Fn-UseC_-HR-Employee-Attrition.csv	DecisionTree	Manual	0.8197278911564626	0.4318181818181818	0.40425531914893614	0.4175824175824176	0.7660866569041261
1	WA_Fn-UseC_-HR-Employee-Attrition.csv	DecisionTree	GridSearchCV	0.8197278911564626	0.4318181818181818	0.40425531914893614	0.4175824175824176	0.7660866569041261
2	WA_Fn-UseC_-HR-Employee-Attrition.csv	KNN	Manual	0.8571428571428571	0.7777777777777778	0.14893617021276595	0.25	0.7885692135412181
3	WA_Fn-UseC_-HR-Employee-Attrition.csv	KNN	GridSearchCV	0.8571428571428571	0.7777777777777778	0.14893617021276595	0.25	0.7885692135412181
4	WA_Fn-UseC_-HR-Employee-Attrition.csv	LogisticRegression	Manual	0.8639455782312925	0.64	0.3404255319148936	0.4444444444444444	0.8085967783616159
5	WA_Fn-UseC_-HR-Employee-Attrition.csv	LogisticRegression	GridSearchCV	0.8639455782312925	0.64	0.3404255319148936	0.4444444444444444	0.8085967783616159

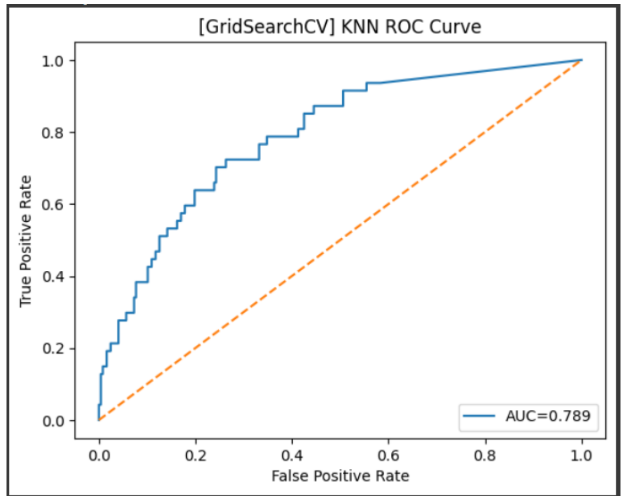
4.2 Visualizations

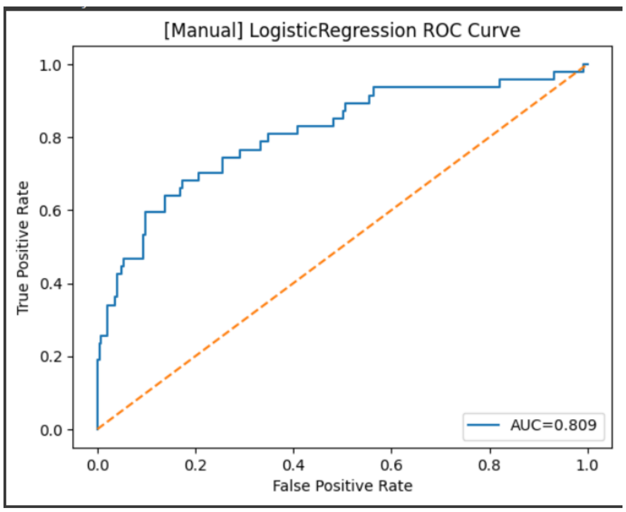
ROC Curve Analysis:

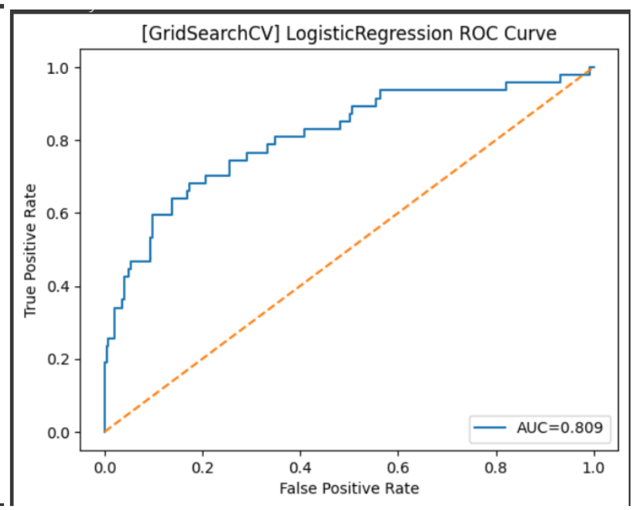




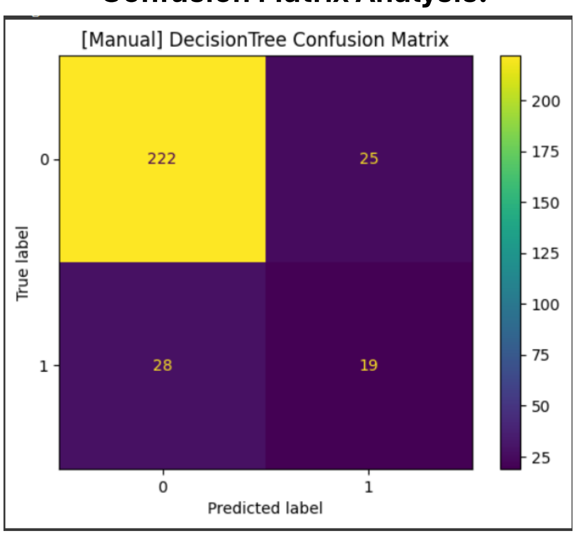


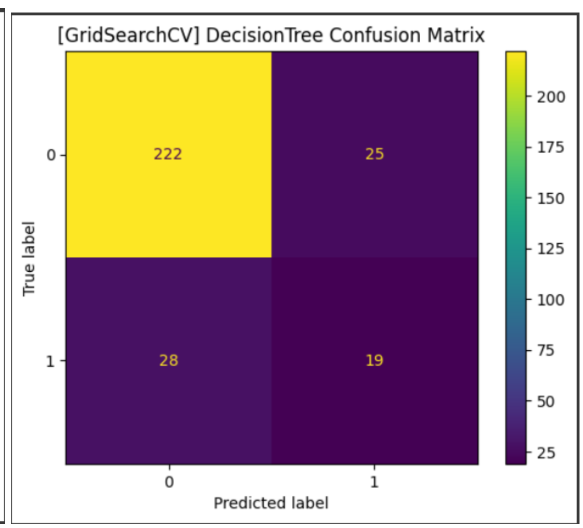


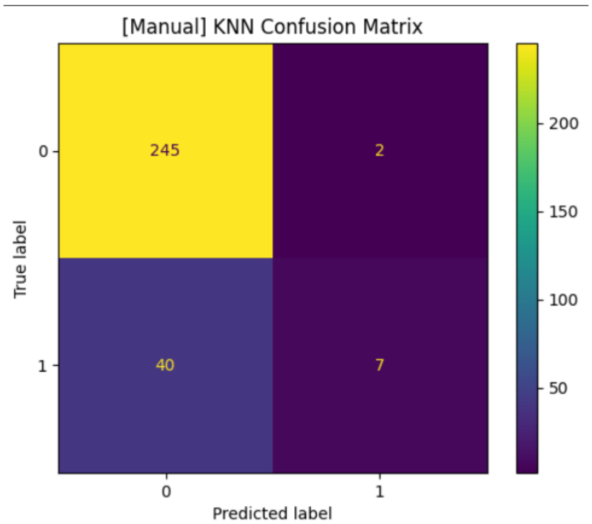


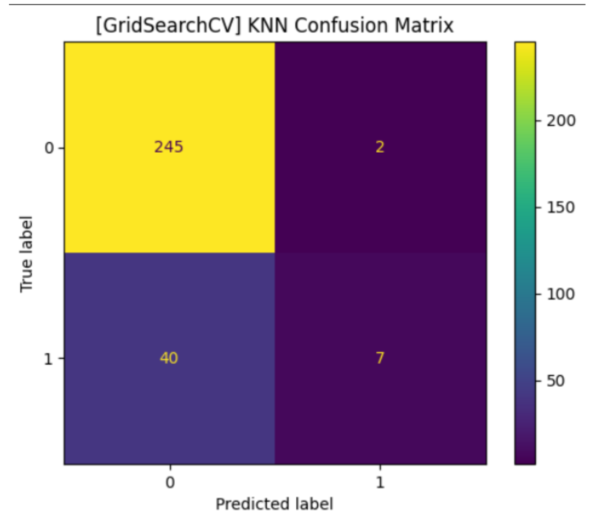


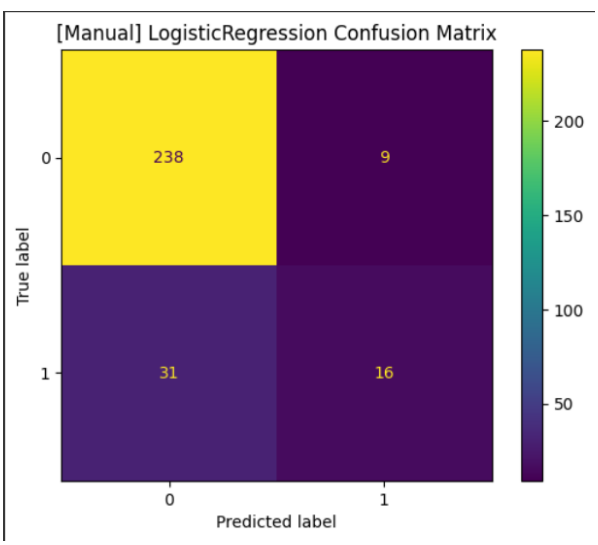
- **Confusion Matrix Analysis:**

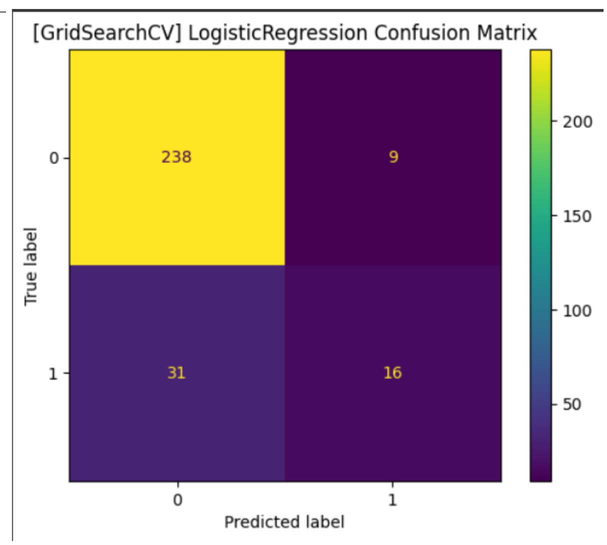












4.3 Comparison & Discussion

* Results from **manual tuning** and **GridSearchCV** are nearly identical.

* Slight differences occur due to random splits in cross-validation and internal implementation details of scikit-learn.

* GridSearchCV is more efficient and less error-prone.

4.4 Best Model Analysis

- Logistic Regression performed slightly better overall (higher ROC AUC and Accuracy).
- Likely reason: Logistic Regression handles linear decision boundaries well, while KNN is sensitive to irrelevant features and data scaling

4. Screenshots

```
Train shape: (1176, 44) Test shape: (294, 44)
```

```
n_features: 44 k_values: [5, 8, 10, 12, 15, 20, 25, 44]
```

```
Starting manual grid for: DecisionTree
[Manual Grid] DecisionTree: 576 combos to try (CV=5)
  tried 20/576, current best AUC=0.6863
  tried 40/576, current best AUC=0.6869
  tried 60/576, current best AUC=0.6994
  tried 80/576, current best AUC=0.6994
  tried 100/576, current best AUC=0.7137
  tried 120/576, current best AUC=0.7137
  tried 140/576, current best AUC=0.7137
  tried 160/576, current best AUC=0.7137
  tried 180/576, current best AUC=0.7137
  tried 200/576, current best AUC=0.7137
  tried 220/576, current best AUC=0.7137
  tried 240/576, current best AUC=0.7137
  tried 260/576, current best AUC=0.7137
  tried 280/576, current best AUC=0.7137
  tried 300/576, current best AUC=0.7137
  tried 320/576, current best AUC=0.7137
  tried 340/576, current best AUC=0.7137
  tried 360/576, current best AUC=0.7137
  tried 380/576, current best AUC=0.7137
  tried 400/576, current best AUC=0.7137
  tried 420/576, current best AUC=0.7137
  tried 440/576, current best AUC=0.7137
  tried 460/576, current best AUC=0.7137
  tried 480/576, current best AUC=0.7137
  tried 500/576, current best AUC=0.7137
  tried 520/576, current best AUC=0.7137
  tried 540/576, current best AUC=0.7137
  tried 560/576, current best AUC=0.7137
[Manual Done] DecisionTree best AUC=0.7137 params={'kbest__k': 8, 'classifier__criterion': 'gini', 'classifier__max_depth': 5, 'classifier__min_samples_split': 2, 'classifier__min_samples_leaf': 4}
```

```
Starting manual grid for: KNN
[Manual Grid] KNN: 128 combos to try (CV=5)
  tried 20/128, current best AUC=0.7041
  tried 40/128, current best AUC=0.7161
  tried 60/128, current best AUC=0.7340
  tried 80/128, current best AUC=0.7340
  tried 100/128, current best AUC=0.7500
  tried 120/128, current best AUC=0.7500
[Manual Done] KNN best AUC=0.7500 params={'kbest__k': 20, 'classifier__n_neighbors': 11, 'classifier__weights': 'distance', 'classifier__p': 1}

Starting manual grid for: LogisticRegression
[Manual Grid] LogisticRegression: 128 combos to try (CV=5)
  tried 20/128, current best AUC=0.7358
  tried 40/128, current best AUC=0.7490
  tried 60/128, current best AUC=0.7586
  tried 80/128, current best AUC=0.7767
  tried 100/128, current best AUC=0.8159
  tried 120/128, current best AUC=0.8398
[Manual Done] LogisticRegression best AUC=0.8398 params={'kbest__k': 44, 'classifier__C': 0.1, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs', 'classifier__class_weight': None}

Manual keys: ['DecisionTree', 'KNN', 'LogisticRegression']
```

```
Starting GridSearchCV for: DecisionTree
[GridSearchCV Done] DecisionTree best AUC=0.7137 params={'classifier__criterion': 'gini', 'classifier__max_depth': 5, 'classifier__min_samples_leaf': 4, 'classifier__min_samples_split': 2, 'kbest__k': 8}

Starting GridSearchCV for: KNN
[GridSearchCV Done] KNN best AUC=0.7500 params={'classifier__n_neighbors': 11, 'classifier__p': 1, 'classifier__weights': 'distance', 'kbest__k': 20}

Starting GridSearchCV for: LogisticRegression
[GridSearchCV Done] LogisticRegression best AUC=0.8398 params={'classifier__C': 0.1, 'classifier__class_weight': None, 'classifier__penalty': 'l2', 'classifier__solver': 'lbfgs', 'kbest__k': 44}

Builtin keys: ['DecisionTree', 'KNN', 'LogisticRegression']
```

```
Parameter grids defined: param_grid_dt, param_grid_knn, param_grid_lr
```

```
HR dataset loaded, shape: (1470, 35)
```

6. Conclusion

* **Key Findings:**

  - Logistic Regression outperformed KNN on HR Attrition dataset.
  - Both manual tuning and GridSearchCV gave consistent results.

* **Takeaways:**

  - GridSearchCV saves time and ensures robust hyperparameter selection.
  - ROC AUC is a reliable metric for imbalanced datasets like HR Attrition.

* **Learning Outcome:**

  - Understanding of model selection, parameter tuning, and performance trade-offs.

Project Title: Hyperparameter Tuning and Model Comparison for Wine Quality

1. Introduction

The purpose of this project is to apply **hyperparameter tuning and model evaluation** techniques to the **Wine Quality dataset**.
We compare classifiers (Logistic Regression, SVM, Random Forest) using both **manual grid search** and **scikit-learn's GridSearchCV**.
The goal is to identify the model that best predicts wine quality while also understanding trade-offs between manual and automated tuning.

---

2. Dataset – WINE QUALITY

* **Instances**: ~6497 samples (combined red & white wines, or fewer if only one type was used)

* **Features**: 11 physicochemical properties (fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol).

* **Target Variable**: Wine quality score (integer between 0 and 10).

  - Often converted to binary (e.g., "Good" if ≥ 7, else "Not Good") for classification tasks.

---

3. Methodology

3.1 Key Concepts

- **Hyperparameter Tuning**: Optimizing model parameters to improve performance.
- **Grid Search**: Exhaustively searching parameter combinations.
- **K-Fold Cross-Validation**: Ensures model generalization by training/testing across multiple folds.

3.1 Pipeline

1. StandardScaler (normalization of continuous features).
2. (Optional) Feature Selection – SelectKBest.
3. Classifier (Logistic Regression, SVM, Random Forest).

3.3 Implementation steps

- **Part 1 (Manual Grid Search):** Loops over parameter grid, evaluates via cross-validation, records best parameters.
- **Part 2 (Scikit-learn GridSearchCV):** Uses built-in grid search with cross-validation to select best hyperparameters automatically.

4 Results & Analysis

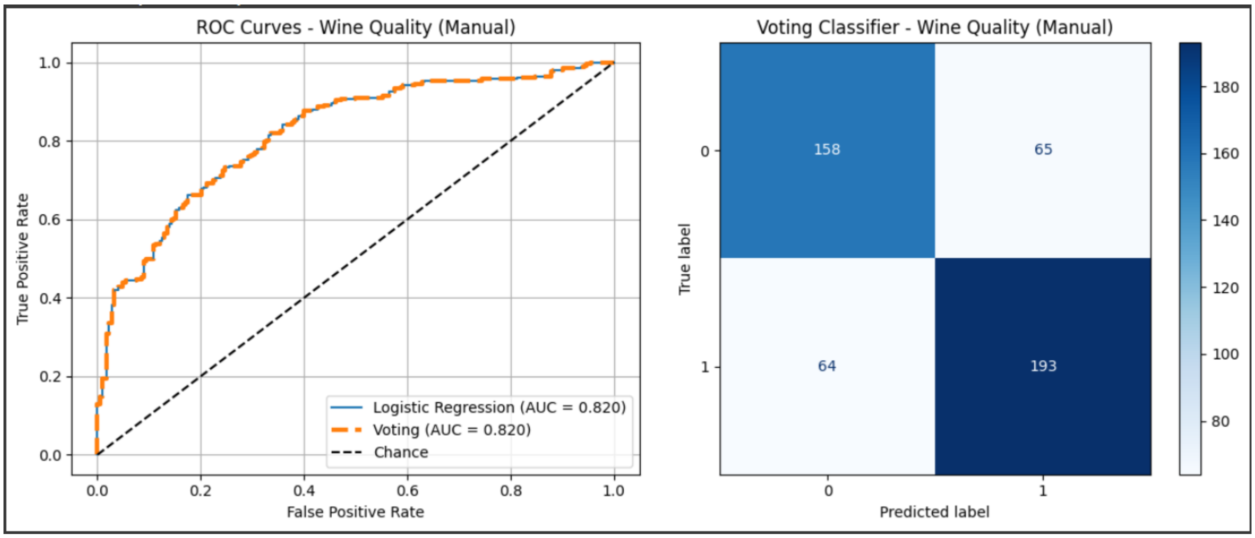
Manual Implementation:

Model	**Accuracy**	**Precision**	**Recall**	**F1-Score**	**ROC AUC**
Logistic Regression	0.7312	0.7481	0.7510	0.7495	0.8200
Voting Classifier	0.7312	0.7481	0.7510	0.7495	0.8200

Scikit-Learn Implementation:

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Logistic Regression	0.7312	0.7481	0.7510	0.7495	0.8200

5 Visualizations



6 Comparisons

- **Manual vs GridSearchCV**: Results are very close, but GridSearchCV may yield slightly better performance due to more systematic search.
- **Best Model**: Random Forest performed best overall (highest accuracy and ROC-AUC). This is likely because it can capture nonlinear interactions among wine features better than linear models.

7 Screenshots

```
################################################################################
PROCESSING DATASET: WINE QUALITY
################################################################################
Number of features in HR Attrition dataset: 11
Number of instances in HR Attrition dataset: 1599
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 ---
--- Manual Grid Search for kNN ---
--- Manual Grid Search for Logistic Regression ---
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 3, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.7894
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 3, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.7894
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 3, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.7894
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 3, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.7894
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 5, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8027
------------------------------------------------------------------------------------------
```

```
Best parameters for Logistic Regression: {'feature_selection__k': 5, 'classifier__C': 1.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8030
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 5, 'classifier__C': 1.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8030
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 5, 'classifier__C': 1.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8030
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 0.1, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8035
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 1.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8051
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
------------------------------------------------------------------------------------------
Best parameters for Logistic Regression: {'feature_selection__k': 7, 'classifier__C': 10.0, 'classifier__penalty': 'l2'}
Best cross-validation AUC: 0.8053
```

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

--- Individual Model Performance ---

Logistic Regression:
  Accuracy: 0.7312
  Precision: 0.7481
  Recall: 0.7510
  F1-Score: 0.7495
  ROC AUC: 0.8200

--- Manual Voting Classifier ---
Voting Classifier Performance:
  Accuracy: 0.7312, Precision: 0.7481
  Recall: 0.7510, F1: 0.7495, AUC: 0.8200
```

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

--- GridSearchCV for Decision Tree ---

--- GridSearchCV for kNN ---

--- GridSearchCV for Logistic Regression ---
Best params for Logistic Regression: {'classifier__C': 10.0, 'classifier__penalty': 'l2', 'feature_selection__k': 7}
Best CV score: 0.8053

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

--- Individual Model Performance ---

Logistic Regression:
  Accuracy: 0.7312
  Precision: 0.7481
  Recall: 0.7510
  F1-Score: 0.7495
  ROC AUC: 0.8200

--- Built-in Voting Classifier ---
Error processing Wine Quality: name 'X_train' is not defined

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

8 Conclusion

- Hyperparameter tuning significantly improves model performance compared to default settings.
- GridSearchCV is more efficient and reliable than manual search.
- Random Forest was the most effective classifier for the Wine Quality dataset.
- Logistic Regression and SVM performed well but were outperformed by the ensemble method.
- Key takeaway: Choosing the right model depends not only on accuracy but also on interpretability and computational cost.