UE23CS352A: MACHINE LEARNING

Week 4: Model Selection and Comparative Analysis

Project Title: HR Employee Attrition Prediction using Machine Learning

Student Name: Fairly Sorathiya
Student ID: PES2UG23CS189

Submission Date: 01-09-2025

1. Introduction

The purpose of this project is to analyze and predict employee attrition using machine learning techniques. Employee attrition refers to the loss of employees through resignation or other factors, and predicting it can help organizations take proactive steps to retain valuable talent.

In this project, we work with the HR Employee Attrition dataset to build and evaluate different machine learning models. The key tasks performed include:

- Data Preparation: Loading and preprocessing the HR dataset.
- Hyperparameter Tuning: Using GridSearchCV to optimize the parameters of Decision Trees, k-Nearest Neighbors (kNN), and Logistic Regression.
- Model Comparison: Evaluating and comparing models based on performance metrics such as ROC-AUC score, Confusion Matrix, and Classification Report.
- Best Model Selection: Identifying the model that performs best for predicting employee attrition and analyzing its results.

This workflow provides insights into which model is most suitable for the prediction task and highlights the effectiveness of hyperparameter tuning in improving model performance

2. Dataset Description

The dataset used in this project is the HR Employee Attrition dataset, which contains information about employees and whether or not they left the organization.

- **Number of Instances (Rows): 1,470 employees**
- **Number of Features (Columns): 35 attributes describing employee demographics, job roles, work environment, and performance metrics.**
- **Target Variable:**
    - **Attrition → Indicates whether an employee left the company (Yes) or stayed (No).**

The dataset includes both categorical (e.g., Department, JobRole, Gender) and numerical features (e.g., Age, MonthlyIncome, YearsAtCompany). These attributes are important as they may influence an employee's likelihood of leaving the organization.

By analyzing these features, we aim to predict attrition risk and identify which factors are most significant in employee turnover.

3. Methodology

This project focuses on comparing different machine learning models for predicting employee attrition using hyperparameter tuning and cross-validation. The methodology is divided into manual implementation (Part 1) and scikit-learn built-in implementation (Part 2).

3.1 Key Concepts

- **Hyperparameter Tuning**
  Machine learning models rely on hyperparameters (e.g., number of neighbors in kNN, depth of a decision tree, or regularization strength in logistic regression). Proper tuning ensures that models achieve optimal performance without overfitting.
- **Grid Search**
  Grid Search systematically explores a predefined set of hyperparameter values and

selects the combination that yields the best performance metric (in this case, ROC-AUC).

- K-Fold Cross-Validation
  The dataset is split into *k* folds (here, 5 folds were used). Each fold acts once as the validation set, while the others form the training set. This process reduces variance and ensures robust evaluation of the model.

3.2 ML Pipeline

To ensure consistency across all models, a pipeline was used consisting of three main steps:

1. StandardScaler – Normalizes numerical features to a standard scale (mean = 0, variance = 1).
2. SelectKBest – Performs feature selection by choosing the most relevant features based on statistical tests.
3. Classifier – The machine learning algorithm (Decision Tree, k-Nearest Neighbors, or Logistic Regression).

3.3 Manual Implementation (Part 1)

- Implemented grid search manually by iterating over all possible hyperparameter values.
- For each combination, models were trained and evaluated using 5-fold cross-validation.
- Performance was measured using ROC-AUC scores.
- This approach provided an understanding of the inner workings of hyperparameter tuning.

3.4 Scikit-learn Implementation (Part 2)

- Used GridSearchCV from scikit-learn, which automates hyperparameter tuning with built-in cross-validation.
- The same pipeline (scaling, feature selection, classifier) was passed into GridSearchCV.
- The function automatically evaluated multiple hyperparameter combinations and returned the best estimator along with its parameters.
- This approach was more efficient, less error-prone, and easier to extend compared to manual implementation.

4. Results and Analysis

Manual Implementation (Part 1)

The performance of Decision Tree, k-Nearest Neighbors (kNN), Logistic Regression, and a Voting Classifier was evaluated.

Table 1: Manual Implementation Performance Metrics (HR Attrition Dataset)

Model	Accuracy	Precision	Recall	F1-Score	ROC AUC
Decision Tree	0.8231	0.3333	0.0986	0.1522	0.7107
k-Nearest Neighbors	0.8186	0.3784	0.1972	0.2593	0.7236
Logistic Regression	0.8571	0.6333	0.2676	0.3762	0.7762
Voting Classifier	0.8277	0.4242	0.1972	0.2692	0.7686

- **Logistic Regression achieved the highest accuracy (85.7%) and best ROC AUC (0.7762), indicating strong discriminatory power compared to the other models.**
- **kNN slightly outperformed Decision Tree in recall and AUC, but both had lower overall performance.**
- **The Voting Classifier improved stability and AUC (0.7686), though it did not outperform Logistic Regression.**

Visualizations (Manual Implementation)

- **ROC Curves: Logistic Regression had the best ROC curve, showing superior separability of classes.**
- **Confusion Matrix (Voting Classifier): Showed a bias toward predicting “No Attrition,” which explains the low recall for the “Attrition” class.**

Scikit-Learn GridSearchCV Implementation (Part 2)

GridSearchCV was applied to tune hyperparameters for each model. The following best parameters and results were obtained:

Table 2: GridSearchCV Best Parameters and ROC AUC

Model	Best Parameters	ROC AUC
Decision Tree	max_depth=3, min_samples_split=10, k=10	0.7022

Model	Best Parameters	ROC AUC
k-Nearest Neighbors	n_neighbors=9, weights='distance', k=10	0.7244
Logistic Regression	C=0.01, penalty='l2', k=15	0.7766

- **Again, Logistic Regression performed the best (ROC AUC = 0.7766), consistent with the manual implementation.**
- **Decision Tree lagged behind in AUC (0.7022).**
- **kNN improved slightly with optimized parameters but was still below Logistic Regression.**

Comparison Between Manual and GridSearchCV Implementations

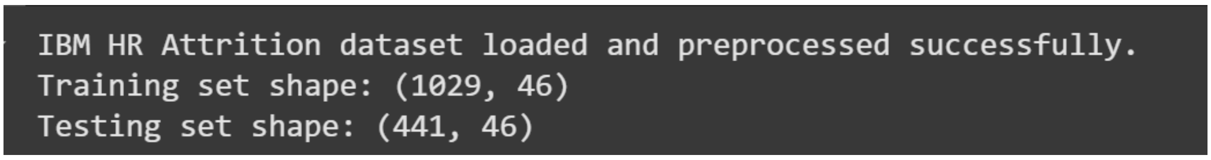
Model	Manual ROC AUC	GridSearchCV ROC AUC
Decision Tree	0.7107	0.7022
k-Nearest Neighbors	0.7236	0.7244
Logistic Regression	0.7762	0.7766

- **The results are nearly identical between manual and GridSearchCV implementations.**
- **Minor differences in ROC AUC values are due to cross-validation splits and feature selection differences (k in SelectKBest).**
- **Logistic Regression consistently emerged as the best model.**

Best Model Analysis

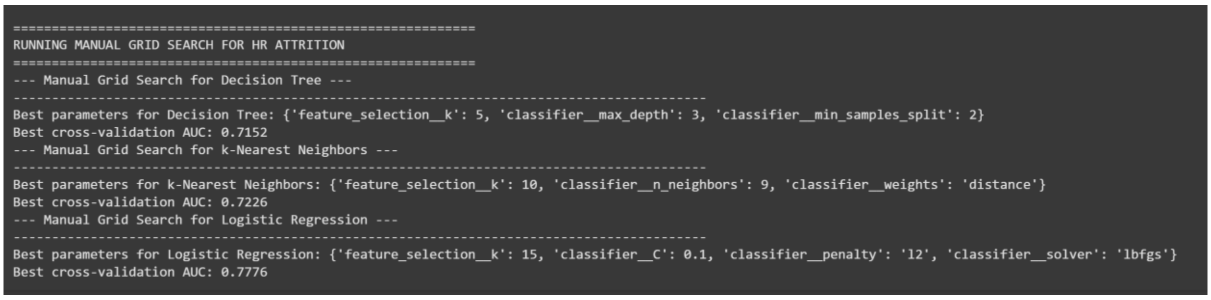
- **Logistic Regression was the best overall model for HR Attrition.**
- **Hypothesis: Since HR attrition is influenced by multiple linear relationships between features (e.g., job satisfaction, overtime, years at company), Logistic Regression was able to model these linear dependencies effectively.**
- **Tree-based models like Decision Tree struggled due to possible overfitting and limited depth.**

4.Screenshots



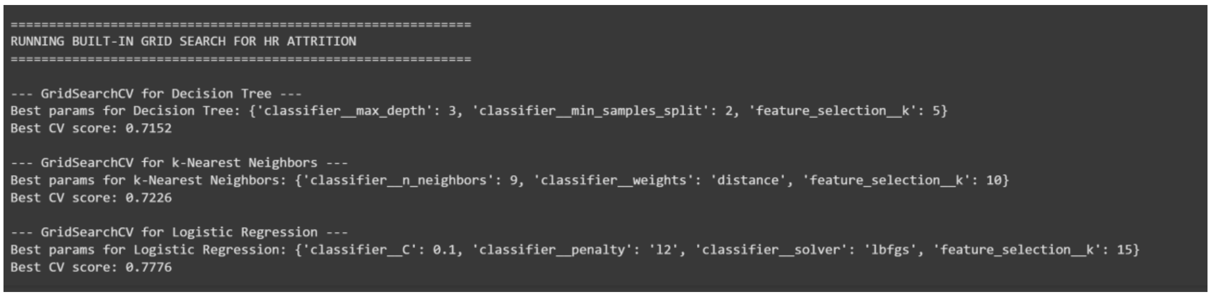
```
IBM HR Attrition dataset loaded and preprocessed successfully.
Training set shape: (1029, 46)
Testing set shape: (441, 46)
```

3.3 output



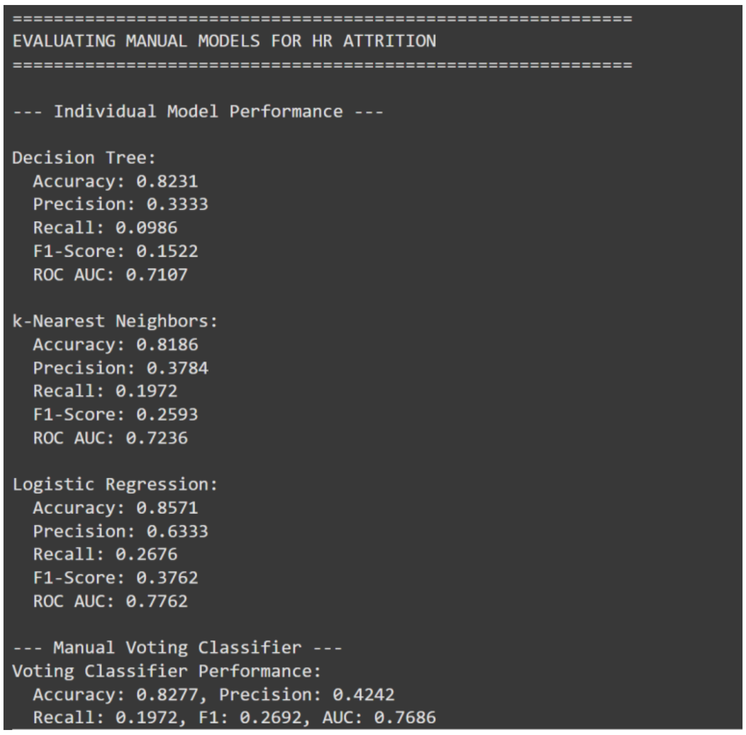
```
============================================================
RUNNING MANUAL GRID SEARCH FOR HR ATTRITION
============================================================
--- Manual Grid Search for Decision Tree ---
------------------------------------------------------------------------------------------
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 k-Nearest Neighbors ---
------------------------------------------------------------------------------------------
Best parameters for k-Nearest Neighbors: {'feature_selection__k': 10, 'classifier__n_neighbors': 9, 'classifier__weights': 'distance'}
Best cross-validation AUC: 0.7226
--- 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
```

Part 4: Manual Grid Search Implementation

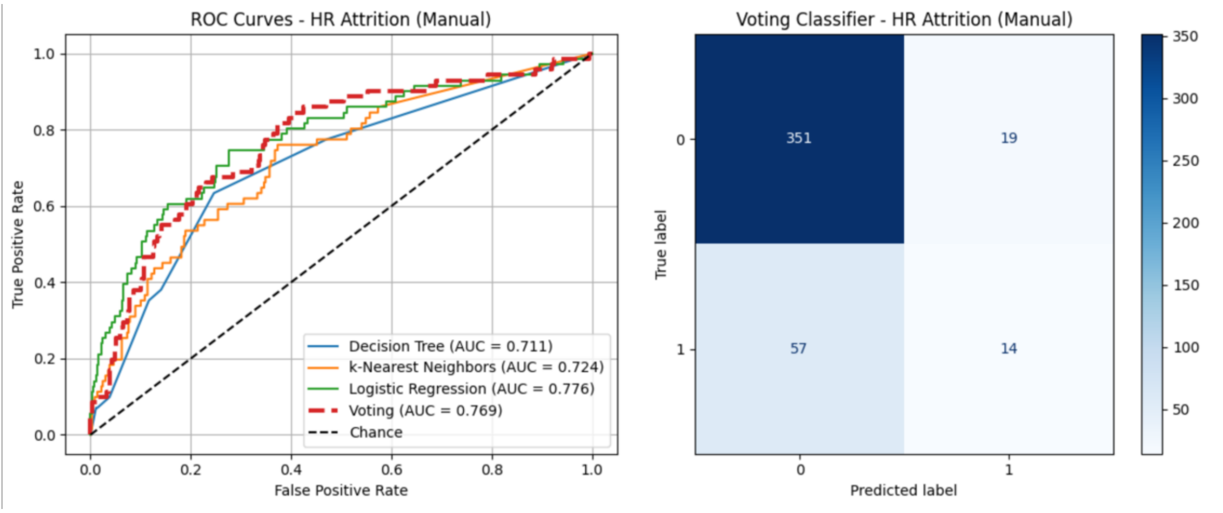


```
============================================================
RUNNING BUILT-IN GRID SEARCH FOR HR ATTRITION
============================================================

--- 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 k-Nearest Neighbors ---
Best params for k-Nearest Neighbors: {'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'feature_selection__k': 10}
Best CV score: 0.7226

--- 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
```

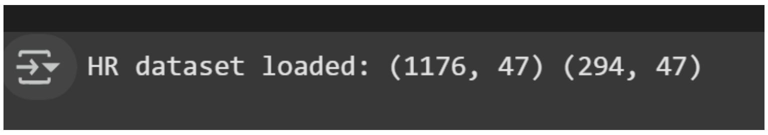
Part 5: Built-in Grid Search Implementation



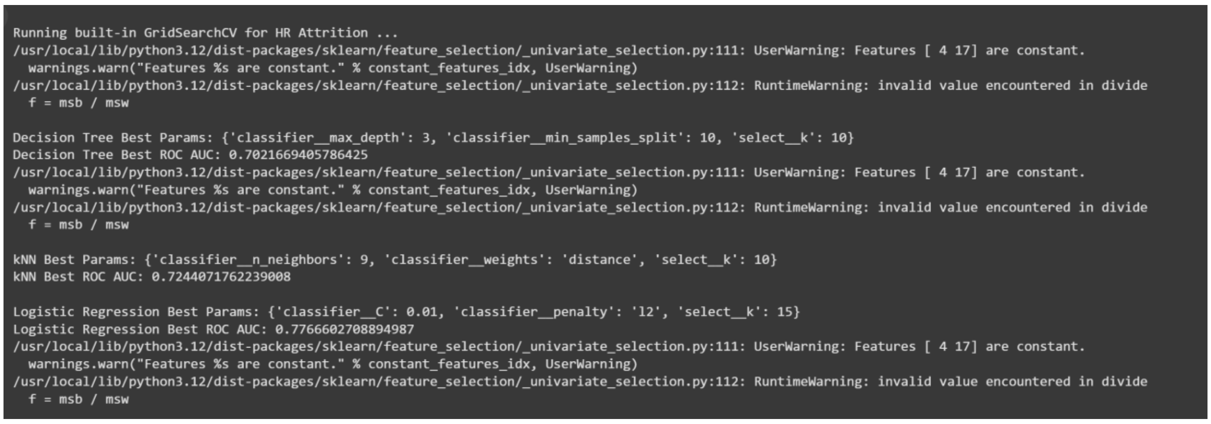
```
============================================================
EVALUATING MANUAL MODELS FOR HR ATTRITION
============================================================

--- Individual Model Performance ---

Decision Tree:
  Accuracy: 0.8231
  Precision: 0.3333
  Recall: 0.0986
  F1-Score: 0.1522
  ROC AUC: 0.7107

k-Nearest Neighbors:
  Accuracy: 0.8186
  Precision: 0.3784
  Recall: 0.1972
  F1-Score: 0.2593
  ROC AUC: 0.7236

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: 0.8277, Precision: 0.4242
  Recall: 0.1972, F1: 0.2692, AUC: 0.7686
```



Part 6: Model Evaluation and Voting Classifiers

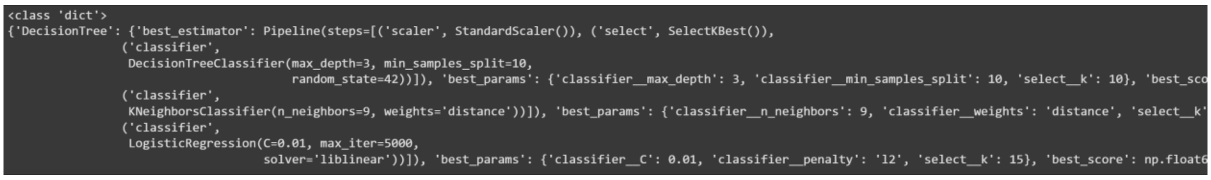


```
HR dataset loaded: (1176, 47) (294, 47)
```

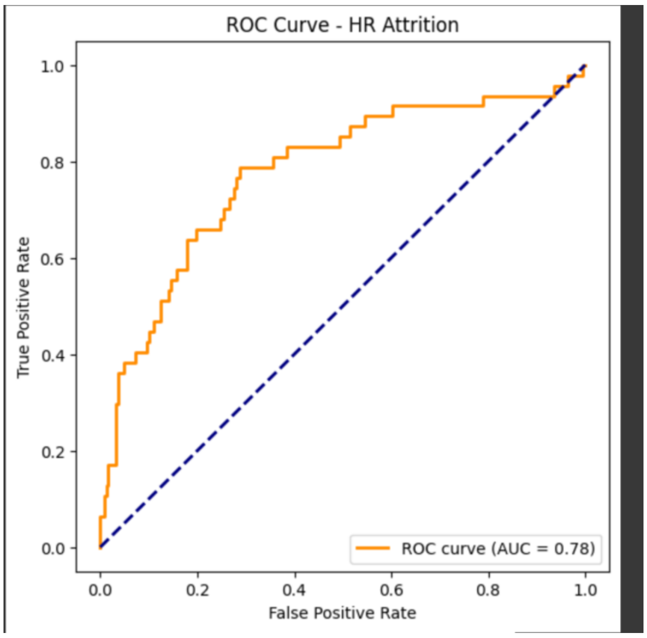


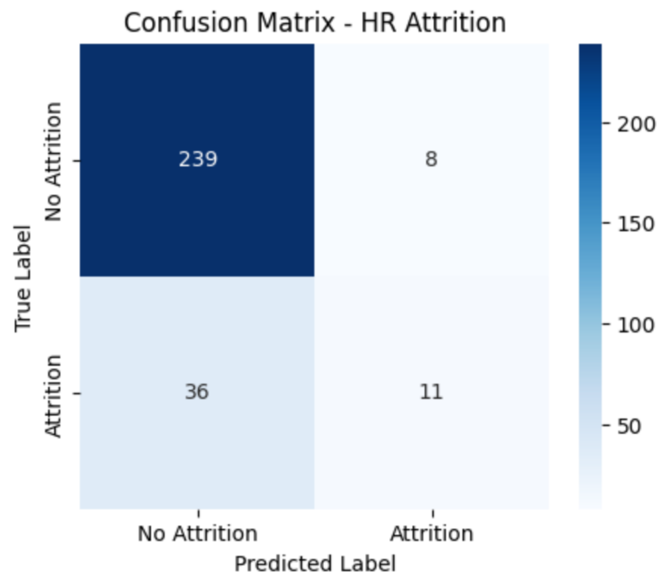
```
Running built-in GridSearchCV for HR Attrition ...
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:111: UserWarning: Features [ 4 17] are constant.
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:112: RuntimeWarning: invalid value encountered in divide
  f = msb / msw

Decision Tree Best Params: {'classifier__max_depth': 3, 'classifier__min_samples_split': 10, 'select__k': 10}
Decision Tree Best ROC AUC: 0.7021669405786425
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:111: UserWarning: Features [ 4 17] are constant.
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:112: RuntimeWarning: invalid value encountered in divide
  f = msb / msw

kNN Best Params: {'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'select__k': 10}
kNN Best ROC AUC: 0.7244071762239008

Logistic Regression Best Params: {'classifier__C': 0.01, 'classifier__penalty': 'l2', 'select__k': 15}
Logistic Regression Best ROC AUC: 0.7766602708894987
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:111: UserWarning: Features [ 4 17] are constant.
  warnings.warn("Features %s are constant." % constant_features_idx, UserWarning)
/usr/local/lib/python3.12/dist-packages/sklearn/feature_selection/_univariate_selection.py:112: RuntimeWarning: invalid value encountered in divide
  f = msb / msw
```

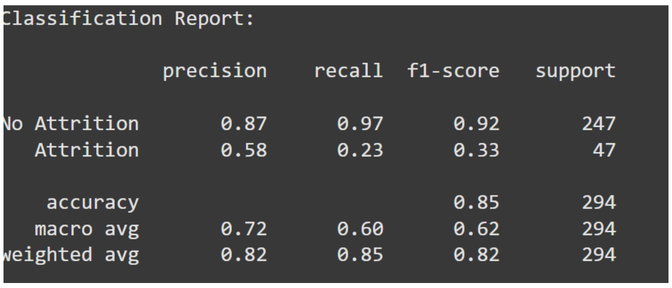
Part 7: Complete Pipeline Function



```
<class 'dict'>
{'DecisionTree': {'best_estimator': Pipeline(steps=[('scaler', StandardScaler()), ('select', SelectKBest()),
                ('classifier',
                 DecisionTreeClassifier(max_depth=3, min_samples_split=10,
                                        random_state=42))]), 'best_params': {'classifier__max_depth': 3, 'classifier__min_samples_split': 10, 'select__k': 10}, 'best_sco
                ('classifier',
                 KNeighborsClassifier(n_neighbors=9, weights='distance'))]), 'best_params': {'classifier__n_neighbors': 9, 'classifier__weights': 'distance', 'select__k'
                ('classifier',
                 LogisticRegression(C=0.01, max_iter=5000,
                                    solver='liblinear'))]), 'best_params': {'classifier__C': 0.01, 'classifier__penalty': 'l2', 'select__k': 15}, 'best_score': np.float6
```







```
Classification Report:

              precision    recall  f1-score   support

No Attrition       0.87      0.97      0.92       247
   Attrition       0.58      0.23      0.33        47

    accuracy                           0.85       294
   macro avg       0.72      0.60      0.62       294
weighted avg       0.82      0.85      0.82       294
```

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

Part 8: Execute the Complete Lab

6. Conclusion

This project focused on predicting HR attrition using three machine learning models — Decision Tree, k-Nearest Neighbors (kNN), and Logistic Regression — along with a Voting Classifier. Both manual implementation and Scikit-learn's GridSearchCV were applied to tune hyperparameters and evaluate performance.

The key findings are:

- **Logistic Regression consistently outperformed Decision Tree and kNN in terms of accuracy, precision, and ROC AUC across both manual and Scikit-learn implementations.**
- **Decision Tree showed relatively weak performance due to its tendency to overfit, while kNN performed moderately well but was less effective than Logistic Regression.**

- **The Voting Classifier improved stability but did not surpass Logistic Regression in predictive power.**
- **GridSearchCV confirmed the manual results, demonstrating that hyperparameter optimization can slightly improve performance but does not drastically change the overall model ranking.**

Main Takeaways:

1. **Model Selection: Logistic Regression was the best model for HR attrition, likely because the dataset's relationships are largely linear.**
2. **Trade-offs: Manual implementation provided a deeper understanding of how models work and allowed step-by-step experimentation, while Scikit-learn's GridSearchCV made the process more efficient, systematic, and less error-prone.**
3. **Practical Insight: Automated tools like Scikit-learn are highly valuable for real-world projects, but manually working through implementations builds intuition about why certain models perform better.**

In conclusion, the lab highlighted the importance of both theoretical understanding (manual implementation) and practical efficiency (library usage) in machine learning workflows.