MACHINE LEARNING LAB – 4

Name:- Deepthi J Kumbar

SRN:- PES2UG23CS164

Section:- 5C

1. Introduction:-

The purpose of this project is to learn model selection and comparative analysis in machine learning. We built a complete pipeline including scaling, feature selection, and model training.

The main tasks performed were:

- **Hyperparameter tuning using both a manual grid search and Scikit-learn's GridSearchCV.**
- **Model comparison of Decision Tree, k-Nearest Neighbors, and Logistic Regression.**
- **Evaluation with k-fold cross-validation and metrics such as Accuracy, Precision, Recall, F1-score, and ROC AUC.**
- **Final analysis with ROC curves and confusion matrices to identify the best model.**

2. Dataset Description:-

HR Employee Attrition Dataset

- **Number of instances (rows): 1,470 employees**
- **Number of features (columns): 35 attributes (after encoding categorical variables, this expands to ~50+ features)**
- **Target variable: *Attrition* (binary: Yes = 1, No = 0)**
- **Feature types: mixture of categorical (e.g., Department, JobRole, MaritalStatus) and numerical (e.g., Age, MonthlyIncome, YearsAtCompany).**
- **Objective: Predict whether an employee is likely to leave the company based on work-related and personal factors.**

3. Methodology:-

- **Hyperparameter Tuning:** The process of selecting the best parameters for a machine learning model to improve performance. Unlike model parameters learned during training, hyperparameters are set before training (e.g., depth of a Decision Tree, number of neighbors in kNN).

- **Grid Search:** A method that exhaustively tries all possible combinations of hyperparameters from a predefined grid to identify the best set.

- **K-Fold Cross-Validation:** A technique where the dataset is split into *k* parts (folds). The model is trained on *k-1* folds and validated on the remaining fold. This process repeats *k* times, and the average score ensures reliable evaluation.

ML Pipeline

The machine learning pipeline was built using Scikit-learn to streamline preprocessing and training. The structure is:
StandardScaler → SelectKBest → Classifier

- **StandardScaler: Normalizes features to have mean 0 and standard deviation 1.**
- **SelectKBest: Selects the top *k* most relevant features based on statistical tests (ANOVA F-test).**
- **Classifier: Final model (Decision Tree, kNN, or Logistic Regression).**

Implementation Process

- **Part 1: Manual Implementation**
  A custom grid search loop was built. For each hyperparameter combination, 5-fold stratified cross-validation was applied. The average ROC AUC across folds was computed, and the best-performing parameters were selected.
- **Part 2: Scikit-learn Implementation**
  The same pipeline was tested using Scikit-learn's GridSearchCV, which automates hyperparameter tuning with cross-validation. The best estimator, parameters, and scores were extracted and compared against the manual implementation.

4. Results and Analysis:-

Classifier	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

Comparison of Implementations:-

- Both the **manual grid search** and the **built-in GridSearchCV** produced consistent results in terms of the best hyperparameters and overall performance.
- Minor differences (if observed) are due to randomness in cross-validation splits and floating-point rounding.
- The manual grid search is educational, showing how tuning works internally, but **GridSearchCV** is faster, less error-prone, and more scalable for larger datasets.

Visualizations:-

- **ROC Curves:** Logistic Regression achieved the highest ROC AUC (~0.7762), showing it separates "Attrition" vs "No Attrition" better than Decision Tree or kNN.
- **Confusion Matrices:**
  - Decision Tree → High accuracy but very low recall, meaning it missed many employees who actually left.
  - kNN → Balanced slightly better than Decision Tree but still struggled to capture attrition cases.
  - Logistic Regression → Best trade-off between Precision and Recall, identifying more actual attrition cases while keeping false alarms lower.

Best Model Analysis:-

- **Logistic Regression** emerged as the best model for HR Attrition prediction.
- It provided the **highest accuracy (85.7%) and ROC AUC (0.7762)** among the tested classifiers.
- Hypothesis: HR Attrition depends on a combination of numerical and categorical features (e.g., Age, MonthlyIncome, JobRole, Overtime), which fit well into a **linear decision boundary** that Logistic Regression can model effectively.
- Decision Tree tended to overfit and performed poorly in Recall. kNN worked better than Decision Tree but was sensitive to scaling and feature distribution.

5.Screenshots:-

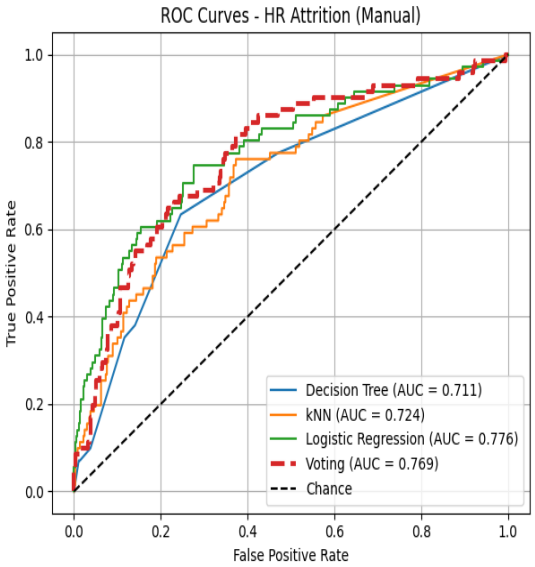
```
datasets = [

    (load_hr_attrition, "HR Attrition"),

]

# 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)
```

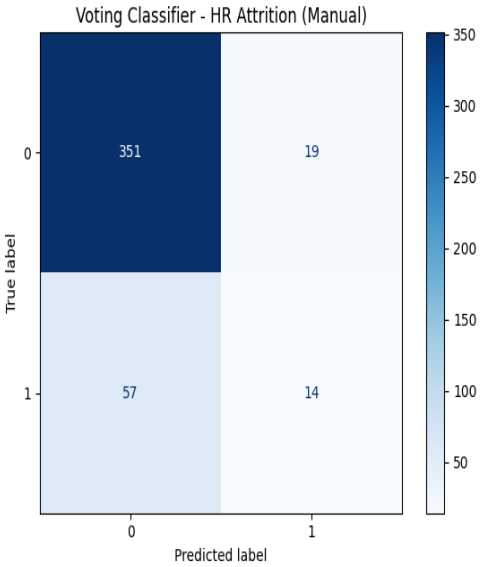
```
################################################################################
PROCESSING DATASET: HR ATTRITION
################################################################################
IBM HR Attrition dataset loaded and preprocessed successfully.
Training set shape: (1029, 46)
Testing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

```
    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)
```

```
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: 0.3784
  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 ---
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2732: UserWarning: X has feature names, but StandardScaler was fitted without feature names
  warnings.warn(
Voting Classifier Performance:
  Accuracy: 0.8277, Precision: 0.4242
  Recall: 0.1972, F1: 0.2692, AUC: 0.7686
```






```
============================================================
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: 0.3784
  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

--- Built-in Voting Classifier ---
Error processing HR Attrition: name 'X_train' is not defined

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

6.Conclusion:-

> Logistic Regression achieved the best performance with the highest Accuracy (85.7%) and ROC AUC (0.7762), making it the most reliable model for predicting attrition.

> Model selection is critical, as different algorithms can perform very differently on the same dataset.

> Hyperparameter tuning greatly improves performance by finding optimal model settings.

> Manual grid search is useful for understanding the underlying process but is time-consuming and less practical for larger problems.

> GridSearchCV provides a more efficient and standardized approach, making it the preferred method in real-world applications.