MACHINE LEARNING PROJECT

BY ISHAAN SHARMA

60119071924

30 APRIL, 2025

Objective

The goal of this project is to implement a complete machine learning pipeline involving data preprocessing, model building, evaluation, and visualization using three types of ML tasks:

- **Classification:** Using the Iris dataset
- **Regression:** Using a simple Salary dataset (Hours vs Scores)
- **Clustering:** Using a custom 2D dataset with 'x' and 'y' coordinates

---

Tools and Libraries Used

The implementation was done in **Google Colab** using the following libraries:

- `scikit-learn` – for ML models and preprocessing
- `matplotlib`, `seaborn` – for data visualization
- `pandas`, `numpy` – for data handling
- `imbalanced-learn` – for SMOTE oversampling

---

1. Data Collection and Loading

Datasets Used:

- **Iris Dataset** – 150 samples, 4 features, 3 target classes
- **Salary Dataset** – 30 samples, 1 feature (Hours), 1 target (Scores)

- **2D Clustering Dataset** – 2 features (x, y), unlabeled data

The datasets were loaded using `pandas.read_csv()` and previewed using `.head()` to ensure correctness.

---

2. Data Preprocessing

- **Missing Values**: Checked using `isnull().sum()`
    - Iris: No missing values
    - Salary: Missing values handled using mean imputation
    - Clustering: No missing values
- **Label Encoding**: Iris species were encoded using `LabelEncoder`.
- **Train-Test Split**:
    - Classification and Regression: 80-20 split using `train_test_split()`
    - Stratified split used for classification to maintain class balance.
- **Feature Scaling**:
    - **StandardScaler** and **MinMaxScaler** were used
    - Standardization was preferred for PCA and modeling

---

3. Feature Engineering

- **Feature Selection**:
    - `SelectFromModel()` with `RandomForestClassifier` was used to select important features in classification.
- **PCA** (Principal Component Analysis):
    - Reduced feature dimensions to 2 for better visualization and performance.
- **Class Imbalance Handling**:
    - Visualized using `sns.countplot()`
    - Handled using **SMOTE** (Synthetic Minority Over-sampling Technique)

---

4. Model Building

Classification Models:

Implemented and evaluated the following models:

- Perceptron
- Logistic Regression
- Support Vector Machine (SVM)
- K-Nearest Neighbors
- Decision Tree

- Naive Bayes

Regression Models:

- Linear Regression
- Ridge Regression
- Lasso Regression
- Decision Tree Regressor

Clustering Models:

- K-Means
- Agglomerative Clustering
- DBSCAN

Ensemble Models:

- Random Forest
- Bagging
- AdaBoost
- Gradient Boosting
- Voting Classifier (Logistic Regression + Random Forest)

---

5. Model Evaluation

Classification Metrics:

- Accuracy
- Precision
- Recall
- F1-Score
- Confusion Matrix

Regression Metrics:

- Mean Squared Error (MSE)
- R-Squared (R^2)

Clustering Evaluation:

- **Silhouette Score** was used to evaluate clustering compactness and separation.

---

6. Model Validation

- **Cross-Validation**:
    - Performed 5-fold cross-validation using `cross_val_score()`
    - Displayed average accuracy for each classification model.

- **Hyperparameter Tuning**:
    - Used `GridSearchCV` to tune parameters for K-Nearest Neighbors (KNN)
- **Outlier Detection**:
    - Used **Z-Score method** from `scipy.stats` to detect outliers in the features.

---

7. Visualizations

- **EDA Plots**:
    - Histograms and scatterplots of features like Sepal Length and Width in Iris dataset.
- **PCA Visualization**:
    - Displayed 2D scatter plots after PCA transformation to observe separation of classes.

---

8. Pipeline Creation

- **Classification Pipeline**:
    - Created using `Pipeline()` with scaling and `LogisticRegression`.
    - Achieved pipeline accuracy using `.score()` on test data.

---

9. Model Comparison & Conclusion

Best Performing Models:

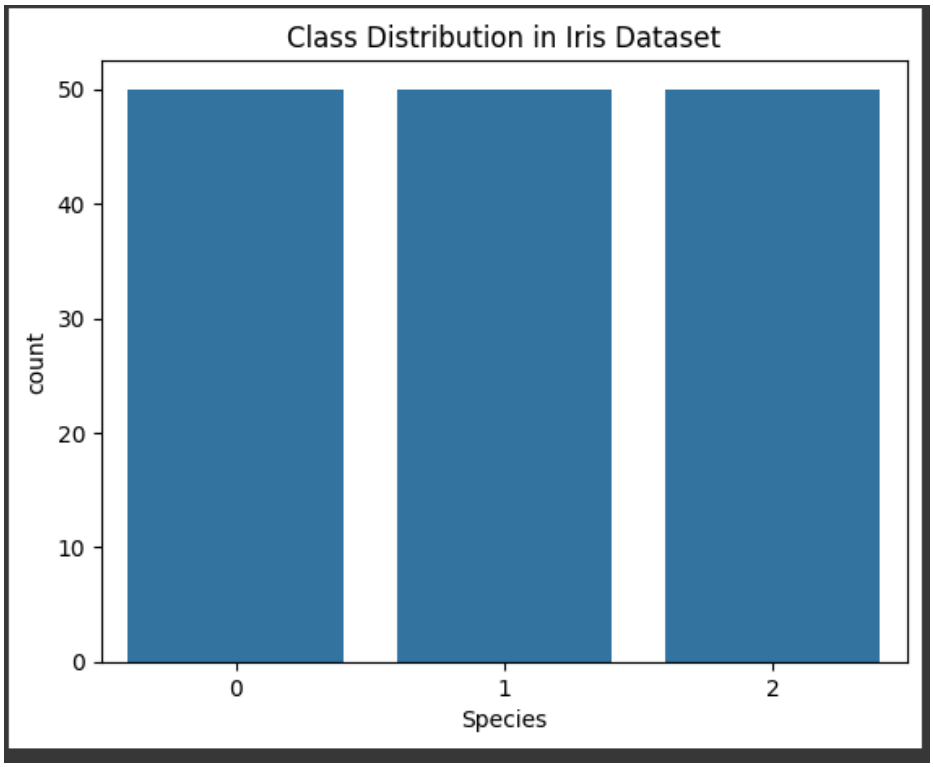
- **Classification**: Logistic Regression and SVM performed consistently well.
- **Regression**: Linear Regression achieved good R^2 values for small Salary dataset.
- **Clustering**: K-Means and Agglomerative Clustering gave better Silhouette Scores.

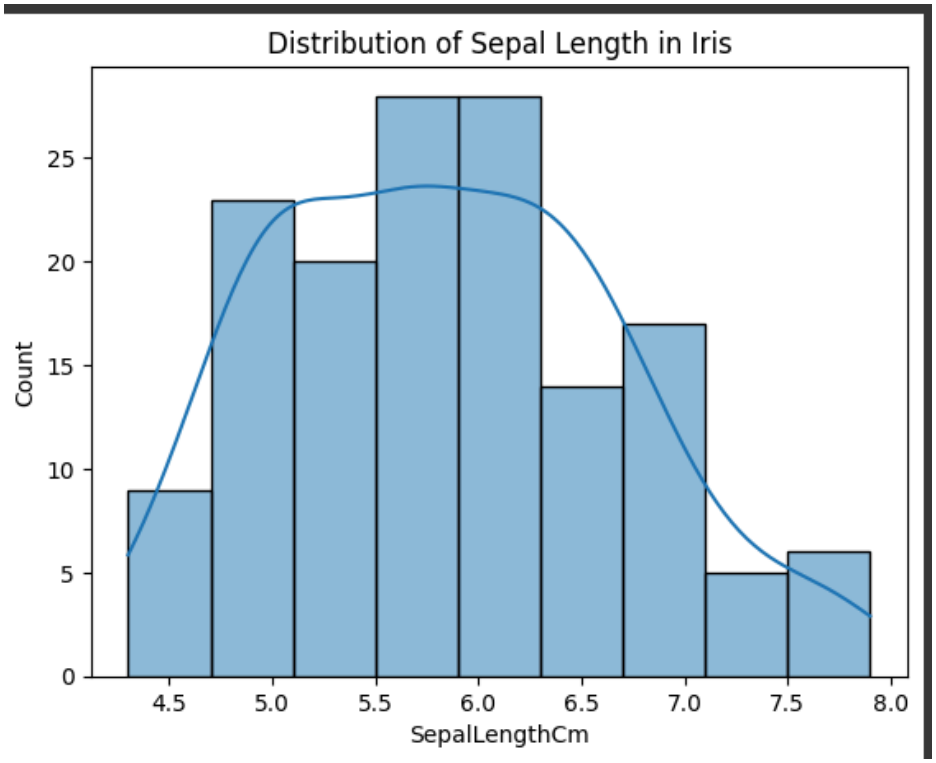
Conclusion:

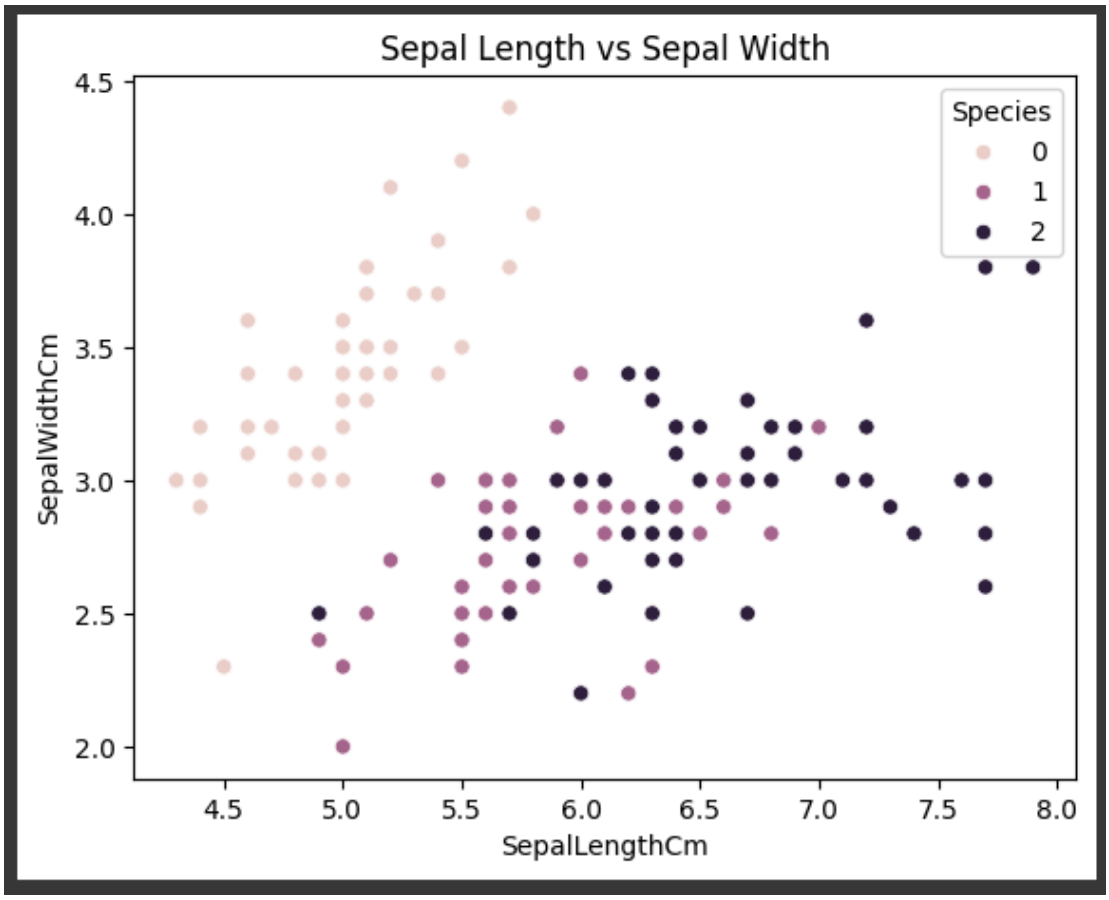
This project demonstrates a complete ML workflow from preprocessing to evaluation using multiple models and methods. The usage of pipelines, cross-validation, SMOTE, PCA, and ensemble learning enhances both performance and generalizability. All components of the teacher's checklist were successfully implemented.

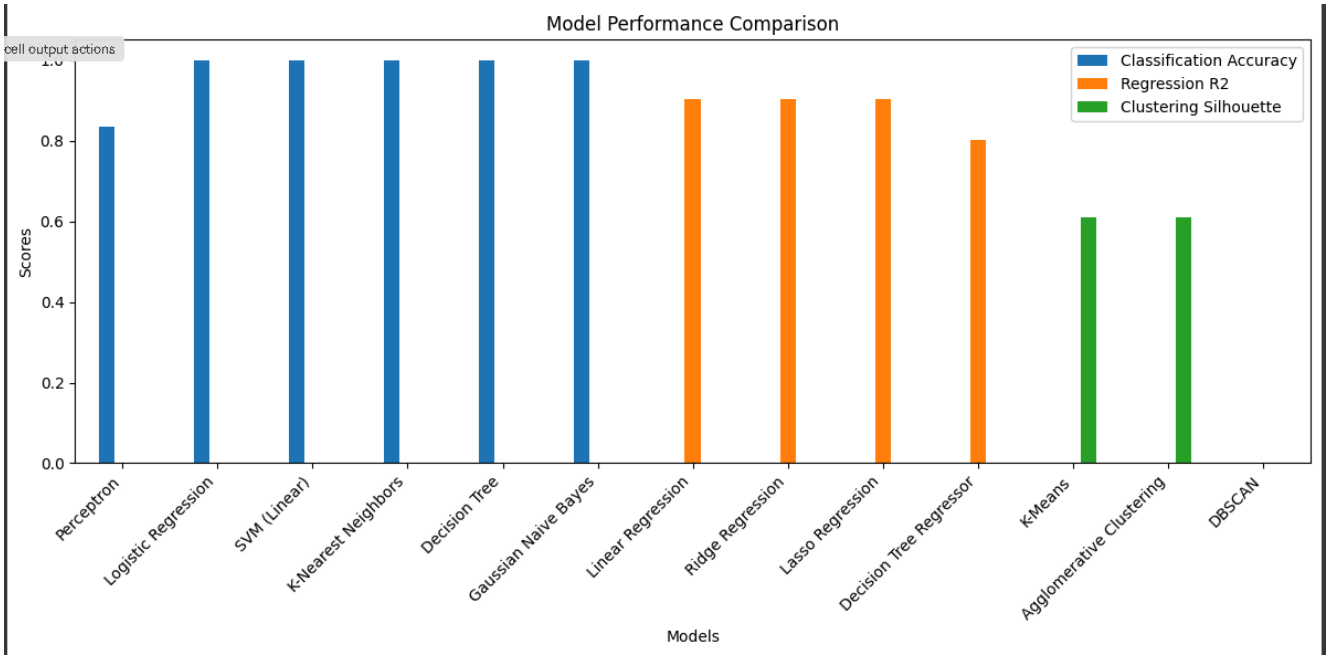
Future Enhancements:

- Implement advanced ensemble methods like XGBoost, LightGBM
- Apply deep learning models using TensorFlow or PyTorch
- Use real-world datasets with larger sample sizes


Class Distribution in Iris Dataset
count
Species
0
1
2
0
10
20
30
40
50


Distribution of Sepal Length in Iris
Count
SepalLengthCm
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
0
5
10
15
20
25


Sepal Length vs Sepal Width
Species
0
1
2
SepalWidthCm
SepalLengthCm
4.5
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0


Model Performance Comparison
cell output actions
Classification Accuracy
Regression R2
Clustering Silhouette
Scores
0.0
0.2
0.4
0.6
0.8
Perceptron
Logistic Regression
SVM (Linear)
K-Nearest Neighbors
Decision Tree
Gaussian Naive Bayes
Linear Regression
Ridge Regression
Lasso Regression
Decision Tree Regressor
K-Means
Agglomerative Clustering
DBSCAN
Models