LABORATORY 4
GROUP 5

By Uhma Pawel and Maurizio Andrés Carrasquero

Introduction

The task involves training and evaluating two machine learning models — a Decision Tree and a Random Forest classifier — using the Wine dataset, a classical multi-class classification dataset. It contains 178 samples of wine, each described by 13 numerical features. The objective is to predict the wine type (class 0, 1, or 2) based on its chemical properties.

Algorithm Description

1. Decision Tree Classifier:
   A tree-based model that splits the dataset recursively based on feature thresholds, creating a structure that allows fast and interpretable predictions.
   - Advantages: Interpretable, non-parametric, handles feature interactions well.
   - Disadvantages: Prone to overfitting, especially with limited depth control.

2. Random Forest Classifier:
   An ensemble of decision trees where each tree is trained on a subset of data and features, and the final decision is based on majority voting.
   - Advantages: Reduces overfitting, high accuracy, robust to noise.
   - Disadvantages: Less interpretable, higher computational cost.

Implementation

Data Preprocessing and PCA Visualization

The dataset is loaded using a custom function from datasets.py. Feature values are standardized using StandardScaler to have zero mean and unit variance. This is crucial as models like PCA are sensitive to feature scaling:

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

To visualize the separability of the classes, we apply PCA to reduce the dimensionality from 13 to 2, enabling 2D plotting:

```
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)
```

The PCA projection shows how well classes can be visually separated before training.

Train/Test Split

We divide the dataset using ¨train_test_split¨, allocating 20% of the data for testing. A fixed random_state ensures reproducibility:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
```

Model Training

Two classifiers are initialized with chosen hyperparameters:

```
model_dt = DecisionTreeClassifier(max_depth=4, random_state=0)
model_rf = RandomForestClassifier(n_estimators=100, random_state=0)
```

We evaluate the models using 4-fold cross-validation on the training data:

```
scores_dt = cross_val_score(model_dt, X_train, y_train, cv=4, scoring='accuracy')
scores_rf = cross_val_score(model_rf, X_train, y_train, cv=4, scoring='accuracy')
```

Then, both models are fit to the entire training set:

```
final_model_dt = model_dt.fit(X_train, y_train)
final_model_rf = model_rf.fit(X_train, y_train)
```

Feature importances are extracted from both models and plotted. The following code sorts and plots them:

```
plot_feature_importances(final_model_dt.feature_importances_, feature_names,
                         "Decision Tree Feature Importances",
                         "DecisionTree_feature_importances.png")
```

Finally, test set predictions are generated and evaluated. The confusion matrix is plotted to visually assess model performance:

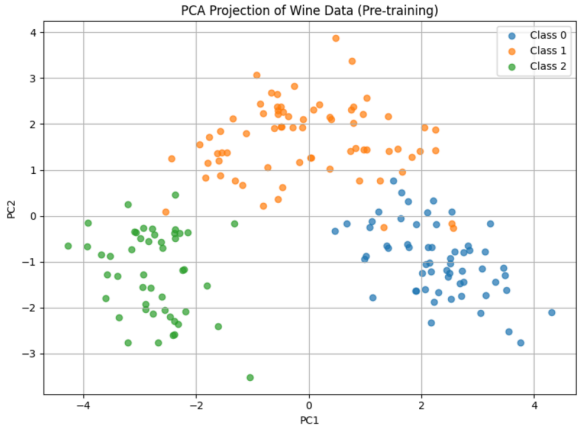
acc, predictions = evaluate_model(model, X_test, y_test)

The best model (based on test accuracy) is used to plot the PCA projection with predicted and true labels:

```
plot_test_pca_with_predictions(X_test, y_test, predictions, best_model_name):
```

Discussion

Plot 1: pre_training_pca.png

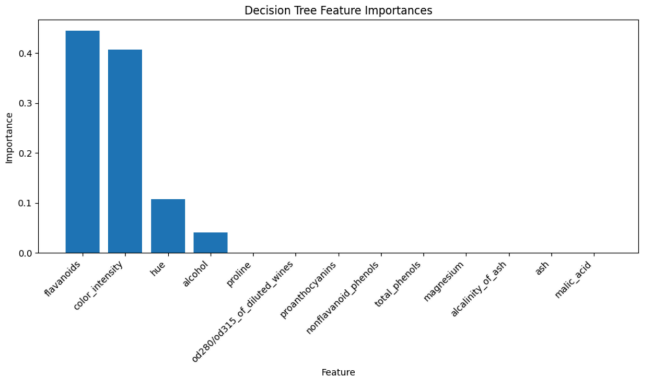


This PCA projection shows the dataset in two dimensions before training:

- The three classes are mostly well-separated in the reduced space.
- Class 0 appears the most isolated, suggesting strong feature discriminability.
- Classes 1 and 2 show minor overlap, indicating potential classification difficulty between these two.

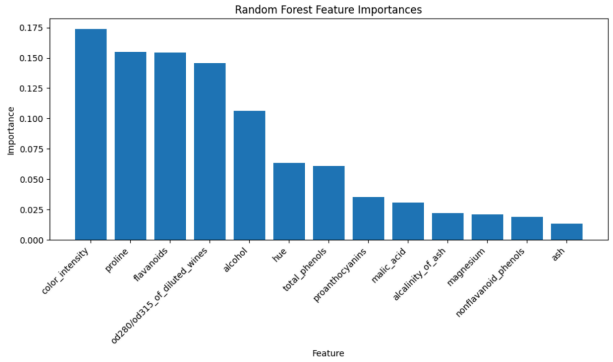
This separation implies the dataset has good structure for classification.

Plot 2: DecisionTree_feature_importances.png



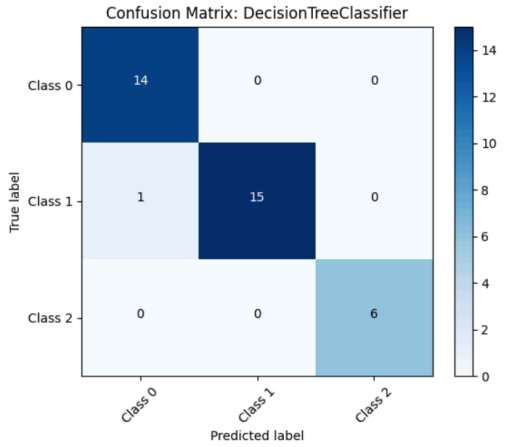
- The most important features identified are: proline, od280/od315_of_diluted_wines, and flavanoids.
- The decision tree relies heavily on a few strong features, while many others are unused.
- This reflects the model's simplicity and tendency to overfit or underfit if not tuned properly.

Plot 3: RandomForest_feature_importances.png



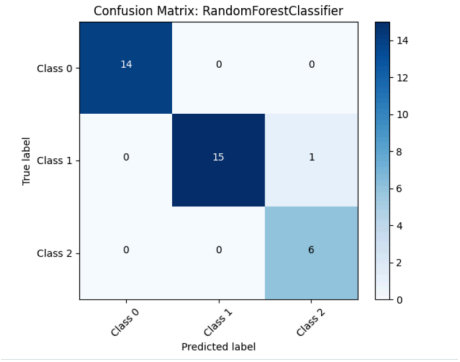
- Shows a more distributed importance across features.
- While proline and flavanoids remain important, other features like alcohol, color_intensity, and hue contribute significantly.
- This spread indicates a more robust model that utilizes more of the dataset's structure.

Plot 4: DecisionTreeClassifier_confusion.png



- The decision tree performs well overall, correctly classifying most samples.
- Misclassifications occur mainly between class 1 and class 2.
- Class 0 is perfectly classified, which aligns with its separability seen in the PCA plot.

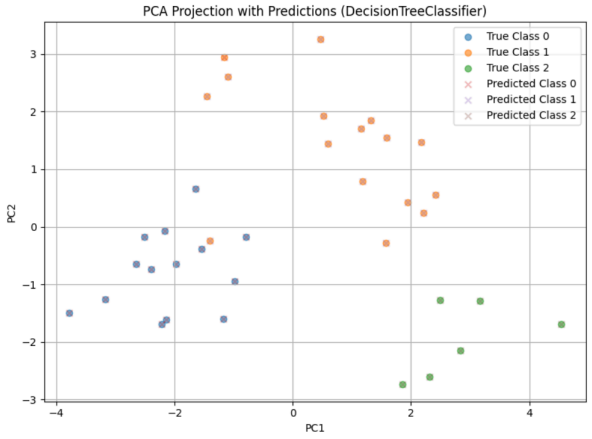
Plot 5: RandomForestClassifier_confusion.png



- Almost perfect classification with very few errors.
- Only one or two misclassified samples, indicating a strong performance on unseen data.
- The confusion matrix is close to perfectly diagonal.

Plot 6: RandomForestClassifier_test_pca.png

- This PCA plot compares true test labels (circles) with predictions (crosses).
- Almost all predicted labels align with true labels, confirming the high accuracy of the random forest.
- Slight mismatches appear in overlapping regions, mostly between class 1 and 2.



Conclusion

We successfully implemented and compared two classification models on the Wine dataset. Random Forest outperformed the Decision Tree in both accuracy and generalization.

Key learnings:

- Feature selection and normalization are crucial for high performance.
- PCA is an effective tool for understanding data structure.
- Ensemble methods like Random Forest significantly improve classification robustness.

Future Work:

- Experiment with hyperparameter tuning (e.g., grid search).
- Try additional classifiers like SVM or logistic regression.
- Use other dimensionality reduction techniques like t-SNE or UMAP for deeper insights.