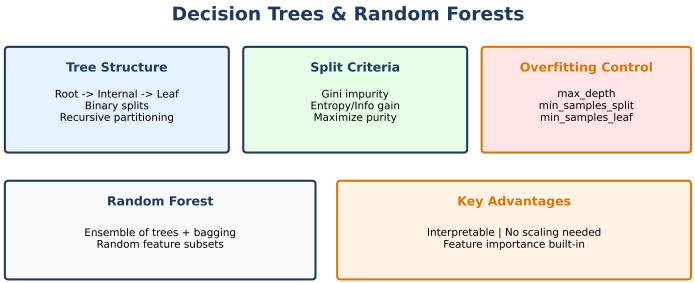
Lesson 26 Summary: Decision Trees

Data Science with Python – Key Concepts

Data Science Program

Decision Trees Overview



Decision trees are intuitive and require no feature scaling

Tree Structure

Components of a decision tree:

- **Root node:** Starting point, first split
- **Internal nodes:** Decision points with splits
- **Leaf nodes:** Final predictions

Each path from root to leaf is a decision rule

Split Criteria

How to choose the best split:

- **Gini impurity:** Probability of misclassification
- **Entropy:** Information gain measure
- **Goal:** Maximize purity of child nodes

Gini and entropy usually give similar results

Controlling Overfitting

Key hyperparameters:

- **max_depth:** Maximum tree depth
- **min_samples_split:** Minimum samples to split
- **min_samples_leaf:** Minimum samples in leaf

Deep trees overfit; shallow trees underfit

Feature Importance

Built-in interpretability:

- **Importance:** Total reduction in impurity
- **Access:** model.feature_importances_
- **Sum:** Always equals 1.0

Feature importance helps understand what drives predictions

Random Forest

Ensemble of trees:

- **Bagging:** Train trees on bootstrap samples
- **Random features:** Subset of features per split
- **Aggregation:** Average (regression) or vote (classification)

Random forests reduce variance compared to single trees

Random Forest Hyperparameters

Key settings:

- **n_estimators:** Number of trees (more = better, slower)
- **max_features:** Features per split (sqrt for classification)
- **max_depth:** Still important to control

100-500 trees is usually sufficient

Advantages and Limitations

Pros:

- No scaling required
- Handles mixed data types
- Interpretable (single tree)

Cons:

- Single trees overfit easily
- Axis-aligned boundaries only

Use random forests for better generalization

Visualization

Plotting decision trees:

- **plot_tree():** sklearn built-in
- **export_graphviz():** For publication quality
- **Limit depth:** For readability

Visual trees are great for explaining models to stakeholders

Quick Reference

Essential Commands:

Task	Code
Decision tree	`DecisionTreeClassifier(max_depth=5)`
Random forest	`RandomForestClassifier(n_estimators=100)`
Feature importance	`model.feature_importances_`
Visualize	`plot_tree(model, filled=True)`

Trees are among the most widely used ML algorithms