ECO447 · WEEK 8

Decision Trees

If-then-else rules for credit decisions, housing prices, and recession calls

CART · Gini & Entropy · Pruning · Feature Importance · Economic Applications

Assoc. Prof. Dr. Onur Polat

Hacettepe University · Department of Economics · Machine Learning for Economists

Fall 2026-2027 · 2-hour lecture

Today's Agenda

8 blocks · 2 hours · laptops open throughout

01 **Why Trees for Economists?**

Where CART fits next to OLS & logit · 10 min

02 **The CART Algorithm**

Splits, Gini, Entropy, impurity · 15 min

03 **Live Demo: Credit Default Risk**

German-credit classification tree · 15 min

04 **Regression Trees: Housing Prices**

California Housing: price-by-location rules · 15 min

05 **Overfitting & Pruning**

Cost-complexity pruning, cross-validation · 15 min

06 **Feature Importance for Policy**

Gini vs Permutation, economic interpretation · 15 min

07 **Hands-on: Recession-Risk Tree**

FRED yield-curve signal · 15 min

08 **Bridge to Random Forests**

Why trees + Week-10 ensembles · 10 min

BLOCK 1 · 10 MIN

Why Trees for Economists?

Where CART fits next to OLS and logit

The Toolbox So Far

ECO447 Weeks 1-7 recap

What we know:

- **OLS regression** (W4) — linear mean model; coefficients have causal/structural meaning when assumptions hold
- **Logistic regression** (W5) — linear decision boundary for binary outcomes; log-odds interpretation
- **Ridge & LASSO** (W6) — same linear models with shrinkage, feature selection for high-dimensional data

But — what if the relationship is NOT linear?

Real economic problems with interaction effects:

- **Credit default:** income matters, BUT only above a credit-score threshold; term matters, BUT only for high LTV ratios
- **Housing prices:** square footage × neighborhood × school district — three-way interaction
- **Recession prediction:** yield-curve slope matters only when unemployment is already rising

Decision Trees discover these conditional rules *without you specifying them in advance.*

Decision Trees at a Glance

Breiman et al. (1984) — CART

✓ Why Economists Love Them

- Interpretable "if-then" rules — can be defended before a committee
- Handle categorical + numeric data natively (no one-hot drama)
- Capture non-linearities and interactions automatically
- No distributional assumptions (no need for normal residuals)
- Good for policy audits — rules are visible, auditable

✗ Where They Struggle

- High variance — small data change → very different tree
- A single tree rarely beats a tuned linear model on smooth relationships
- Not causal — rules are correlation-based unless you add identification
- Axis-parallel splits — can be awkward for rotated boundaries
- Fully grown trees overfit aggressively

Econometric Perspective

Think of CART as a **piecewise-constant regression** on hierarchically defined subgroups. Each leaf = a treatment cell with its own prediction.

In causal-inference terms (W12 preview): Causal Trees (Athey & Imbens, 2016) extend CART to discover *heterogeneous treatment effects* — which subgroups benefit from a policy and which don't. Honest Random Forests (Wager & Athey, 2018) do this at scale.

BLOCK 2 · 15 MIN

The CART Algorithm

Splits, impurity measures, information gain

Recursive Partitioning — Step by Step

Grow a tree one split at a time

Given data A, find the best binary split:

1. For each predictor variable X_j:

a. For every candidate value s_j of X_j:

i. Split records into $\{ X_j < s_j \}$ and $\{ X_j \geq s_j \}$

ii. Measure class homogeneity (or variance for regression) in each subgroup

b. Pick the s_j that maximises the within-group homogeneity gain

2. Across all predictors, pick the best (X_j*, s_j*) pair

3. Recurse on each new subgroup until a stopping rule kicks in

Economic intuition

Each split answers: "Among the variables in front of me, which one — and at which threshold — best separates the outcome I care about?" For credit scoring, the first split often lands on income, FICO, or debt-to-income — exactly the variables an underwriter would check first.

Impurity Measures

How do we score a split's quality?

Gini Impurity

$$I(A) = \Sigma\ p_k(1 - p_k) = 1 - \Sigma\ p_k^2$$

- Range: [0, 0.5] for binary, [0, (K−1)/K] for K classes
- Minimum (0) when pure (all one class)
- Maximum when classes are equally mixed
- Default in scikit-learn (criterion="gini")
- Faster than entropy — no logarithm

Entropy (Information)

$$H(A) = -\ \Sigma\ p_k \cdot \log_2(p_k)$$

- Range: [0, $\log_2(K)$]
- Max information: uniform distribution
- Rooted in Shannon's information theory
- scikit-learn: criterion="entropy"
- Produces nearly identical trees to Gini on most data

Information Gain — the selection rule

$$IG(A, s) = I(A) - [\ |A_L| / |A| \cdot I(A_L) + |A_R| / |A| \cdot I(A_R)\ ]$$

The algorithm picks the split (X_j*, s_j*) that MAXIMISES Information Gain — equivalent to minimising the weighted-average impurity of the children. For regression trees, replace impurity with variance (mean squared error).

BLOCK 3 · 15 MIN · LIVE DEMO

Classification: Credit Default Risk

German-credit dataset — 1 000 borrowers, 20 features

Case: Approving a Personal Loan

What rules would YOU write as an underwriter?

The business problem

A bank receives 1 000 loan applications. 300 historically defaulted. For a new applicant, predict default risk and decide whether to approve. Model must be explainable to regulators and to the borrower (adverse-action notices).

Features in the German-credit dataset:

Numeric: duration (months), credit_amount, installment_commitment, age, existing_credits, num_dependents

Categorical: checking_status, credit_history, purpose, savings_status, employment, personal_status, property_magnitude, other_payment_plans, housing, job, telephone, foreign_worker

Target: good (paid off) vs bad (default) — class imbalance ~70/30

What we'll fit live: *DecisionTreeClassifier(max_depth=3, min_samples_leaf=20). Deep trees overfit — we demo pruning in Block 5.*

Reading the Credit Tree

How to interpret a trained tree in plain English

Sample path (a leaf rule):

If `checking_status ∈ {no checking, <0}` **AND** `duration ≥ 36 months` **AND** `credit_amount ≥ 7 500` → **predict**

BAD (78% of 73 training records in this leaf defaulted)

This is the single biggest selling point of trees

- **Regulators** (e.g., Fed SR 11-7, ECB TRIM) require every credit model to be *documentable and challengeable* — a rule-based tree passes trivially.
- **Adverse-action notices** (US Reg. B / FCRA) can cite the specific rule that triggered a decline.
- **An underwriter** can look at the tree and say "yes, this matches how we actually think" — building organizational trust.

Compare: explaining a 256-dimension logistic regression with 40 interaction terms to a regulator.

BLOCK 4 · 15 MIN

Regression Trees: Housing Prices

California Housing — 20 640 census blocks

From Classification to Regression

Same algorithm, different criterion

Key change: impurity → variance

For a regression tree, each leaf predicts the **mean of y** of the training records that land there. The split criterion becomes:

$$\mathrm{MSE}(A) = (1/|A|) \cdot \Sigma_{i \in A} (y_i - \bar{y}_A)^2$$

The algorithm picks the split that maximally reduces this MSE across children. (scikit-learn: DecisionTreeRegressor, default criterion="squared_error".)

California Housing — our regression demo

- **20 640 census blocks** from the 1990 US Census
- **Target:** median house value in $100 000s
- **Predictors:** median income, house age, average rooms/bedrooms, population, households, latitude, longitude
- *What economic rules do you expect to emerge? ("Income × coastal proximity", "age penalty at edges of SF Bay"...)*

Lab (your Colab): DecisionTreeRegressor at depths 2, 4, 6, 8, 10, 15, None. Watch R^2 rise on train, peak then fall on test.

The Regression Tree in Economic Terms

Each leaf is a sub-market

Illustrative leaf rules (depth-4 tree):

```
Leaf 1: median_income ≥ 5.0 AND longitude < -118.0 → predict $340 k (coastal SF / SV rich blocks)
Leaf 2: median_income ≥ 5.0 AND longitude ≥ -118.0 → predict $265 k (inland affluent)
Leaf 3: median_income < 2.5 AND ocean_prox = INLAND → predict $82 k (poor Central Valley blocks)
```

Why economists care about these rules

1. **Automatic market segmentation —** the tree finds the sub-markets without you specifying neighborhood dummies
2. **Interaction discovery —** income × location × house-age interactions emerge naturally; no need to pre-specify them
3. **Policy targeting —** if a housing subsidy should go to blocks where price < $150 k, the leaves give you an exact definition
4. **Baseline for hedonic models —** compare tree R^2 to your log-linear hedonic specification. If the tree wins, your log-linear mis-specifies.

Caveat: *tree rules describe the historical joint distribution of features and price — they are NOT causal effects of any single variable.*

BLOCK 5 · 15 MIN

Overfitting and Pruning

Cost-complexity pruning + cross-validation

The Overfitting Problem

A fully grown tree memorises noise

⚠ Fully grown trees are deadly

An unconstrained CART can grow until every training record sits in its own pure leaf → 100% training accuracy. But: those leaves also encode every random blip in the data. On fresh data, the tree collapses to essentially majority-class accuracy.

Two families of defense:

Pre-pruning (stop early)

- max_depth — hard cap on tree depth
- min_samples_split — don't split a node with few samples
- min_samples_leaf — each leaf must have ≥ k observations
- min_impurity_decrease — only split if gain exceeds threshold

Fast, but you have to guess the hyperparameter.

Post-pruning (grow then cut)

- Grow the full tree first
- Cost-complexity pruning: ccp_alpha
  — Penalty α × (# leaves) added to error
  — Larger α → simpler tree
- Scan α values, pick the one that minimises CV error

Principled, built into sklearn.

Cost-Complexity Pruning Formally

Breiman's α-pruning (CART, 1984)

The penalised objective

$$R_\alpha(T) = R(T) + \alpha \cdot |T|, \quad \text{where}$$

- $R(T)$ = training error of tree T
- $|T|$ = number of leaves (complexity)
- $\alpha \geq 0$ = complexity cost per leaf

$\alpha = 0 \rightarrow$ full tree. $\alpha \rightarrow \infty \rightarrow$ stump (single leaf). Sweet spot: cross-validation.

Practical recipe (your Colab Block 5)

1. `path = tree.cost_complexity_pruning_path(X_train, y_train)`
2. For each α in `path.ccp_alphas`: fit a `DecisionTreeClassifier(ccp_alpha=α)` and record CV accuracy
3. Pick $\alpha^* = \text{argmax}$ CV accuracy
4. Refit final model on full training set with `ccp_alpha=α*`

*For the German-credit demo, the optimal α typically yields a tree with 5-10 leaves — compact enough to **write on a whiteboard** and defend before a credit committee.*

BLOCK 6 · 15 MIN

Feature Importance for Policy

Gini vs Permutation — and how to report

Two Ways to Rank Variables

Which features drive our predictions?

① Gini / MDI Importance

The sum of impurity reductions across all splits on each variable, weighted by the share of samples reaching those splits.

Pros:
- Free with the tree — no extra compute
- Available via .feature_importances_

Cons:
- Biased toward high-cardinality features
- Computed on training data — may mislead

② Permutation Importance

For each variable, randomly shuffle its column in the TEST set and measure how much accuracy drops.

Pros:
- Model-agnostic — works for any classifier
- Measured on held-out data → honest
- Interpretable: "if I destroy this signal, how much do I lose?"

Cons:
- Slow — needs many model evaluations
- Can underweight correlated variables

When they disagree — what to report

- **Default to permutation importance** in a policy paper — it answers the question regulators actually care about ("if we had bad data on X, how much would the model break?")
- **If Gini ranks a feature high but permutation ranks it low:** the tree used it as a convenient but redundant splitter.
- **Correlated predictors** will split importance between them in both methods — report them as a group where possible.
- ***Preview of Week 12:*** *for true causal interpretation, use SHAP values or causal-forest partial effects, not raw importances.*

BLOCK 7 · 15 MIN · HANDS-ON

Recession-Risk Tree

Can a tree beat the "yield-curve-only" rule?

The Challenge

Yield-curve inversion as a recession signal

Economic background

Since the 1960s every US recession has been preceded by an **inversion** of the yield curve — 10-year Treasury yield minus 3-month T-Bill rate falling below zero (Estrella & Mishkin 1998, Rudebusch & Williams 2009). The classic rule: ***"inverted for 3+ months → recession in 6-18 months".***

Question: can a decision tree using additional macro signals (unemployment, industrial production, inflation, credit spread) improve on this?

Hands-on in your Colab (Block 7)

1. Download FRED series (free) via pandas_datareader: T10Y3M, UNRATE, INDPRO, CPIAUCSL, BAA-10Y spread, USREC dummy
2. Target: USREC 12 months ahead (NBER recession dummy)
3. Fit a depth-3 tree — read the rules — compare out-of-sample AUC vs the single-variable logit "yield curve only"
4. Discuss: what does your tree say — does recession risk really need multiple variables, or is the yield curve enough?

BLOCK 8 · 10 MIN

Bridge to Random Forests

Why we keep going after single trees

The Weakness that Motivates Everything Else

Tree instability

⚠ High variance

If you bootstrap-resample your 1 000 German-credit applicants 10 times and fit 10 trees, you get 10 noticeably different trees — different top splits, different leaf rules. The prediction for any single applicant can swing by 10-20 percentage points across the ensemble.

How we fix it — Weeks 10 and 11 preview

Week 10 — Bagging and Random Forests. Fit MANY trees on bootstrap samples (and random feature subsets) and average their predictions. Variance ↓ sharply, bias ≈ unchanged. The "wisdom of the crowd" — a 500-tree forest reliably beats a single tree on almost every tabular dataset.

Week 11 — AdaBoost / Gradient Boosting. Fit trees SEQUENTIALLY, each one learning from the residuals of the previous. XGBoost is the industry-standard implementation — it currently wins most Kaggle tabular competitions and is the default for most fintech credit models.

Week 12 — Causal Trees. Trees designed to estimate heterogeneous treatment effects — which subgroups benefit from a policy intervention. The frontier for ML in applied economics.

Takeaways for Today

What to remember and what to practice

🧠 Concepts to remember

1. CART splits data recursively using information gain (Gini or entropy for classification, MSE for regression)
2. A fully grown tree memorises — prune with max_depth, min_samples_leaf, or cost-complexity (ccp_alpha)
3. Report permutation importance — not just Gini — when writing a paper or a regulator submission
4. Trees discover interactions and sub-markets automatically — a complement to linear hedonic models, not a replacement for causal identification

💻 Your Colab tasks tonight

- Rerun the German-credit tree with criterion="entropy" and compare the first 3 splits to the Gini tree
- Grid-search max_depth ∈ {3, 5, 7}, min_samples_leaf ∈ {10, 20, 50} on the California housing regression — report best RMSE
- Train 10 trees on bootstrap samples of the credit data and visualize the 10 "top splits" — feel the instability
- (Challenge) Add the FRED credit-spread series (BAA10Y) to the recession tree and see if it beats the T10Y3M-only model