BP301T · B.PHARM THEORY

BP301T Introduction to Machine Learning in Pharmaceutical Sciences

Unit IV — Tree-Based Models & Ensemble Learning

Pills & Python Notes
From-zero teaching for pharmacy students.

PILLS & PYTHON
Pharmacy • Data • AI • Healthcare
Instagram @pillsandpython · pillsandpython.com · YouTube @PillsAndPython

CHAPTER 4 — UNIT IV: TREE-BASED MODELS & ENSEMBLE LEARNING

1. Learning Objectives

After studying this chapter, you will be able to:

- Draw and explain the structure of a decision tree (root, branches, leaves).
- Explain splitting criteria in simple words (which question to ask first).
- Trace a decision path for a new patient and interpret feature importance.
- Explain the ensemble concept and how a random forest works.
- List advantages and limitations of tree-based models.
- Apply trees and forests to ADR risk stratification, patient classification and treatment-outcome prediction, and demonstrate them in sklearn.

2. Definitions

Decision tree. A decision tree is a flowchart-like model that asks a series of yes/no questions about the features, following branches down to a leaf that gives the prediction. It is supervised learning — it learns the questions and their order from labelled training data.

Splitting criterion. The rule the algorithm uses to choose the best question at each node — the question that most cleanly separates the classes (e.g., "is age > 60?" separates ADR cases better than "is age > 20?").

Decision path. The unique route from the root to a leaf for one particular case — the chain of questions and answers that produced its prediction. It makes the model's reasoning fully visible.

Feature importance. A score (0 to 1) telling how much each feature contributed to the tree's decisions overall. Higher = the feature was more useful for splitting.

Ensemble. An ensemble is a committee of models whose predictions are combined (usually by majority vote). A single tree can be unstable; a committee of trees is steadier — like asking several doctors instead of one.

Random forest. A random forest is an ensemble of many decision trees, each trained on a random sample of data and features; the final prediction is the majority vote of all trees.

3. Terminology

Term	Plain meaning
Root node	The very first question at the top of the tree.
Internal/decision node	A middle question with branches for each answer.
Leaf node	An end point with no further questions — carries the final prediction.
Branch	One answer-path from a question (e.g., "yes" goes left, "no" goes right).
Depth	The longest chain of questions from root to leaf — how "tall" the tree is.
Overfitting (trees)	A tree grown too deep memorises training quirks — needs pruning or forests.

Term	Plain meaning
Pruning	Cutting off weak branches to keep the tree simple and general.
Majority vote	The ensemble's decision = the class most trees voted for.
Stratification	Dividing patients into risk layers (low/medium/high) — a key tree application.

4. Explanation: decision trees — structure and splitting

Technical concept. The algorithm examines every feature and every possible split point, picks the split that best separates the classes (maximising purity of the resulting groups), and repeats this recursively for each branch until a stopping rule (e.g., maximum depth, minimum samples per leaf).

Simple explanation. Building a tree is like playing "20 questions" with the data. At each step the computer asks: "Which single question divides the patients into the purest groups?" — e.g., first "dose > 350 mg?", then within each group "age > 60?" — until each final group is mostly one class.

Example tree (ADR prediction).

```
Is dose > 350 mg?
├── YES → Is age > 60?
│           ├── YES → LEAF: ADR likely (high risk)
│           └── NO  → LEAF: ADR possible (medium risk)
└── NO  → LEAF: ADR unlikely (low risk)
```

A new patient (age 70, dose 420 mg): dose > 350 → yes; age > 60 → yes → **"ADR likely (high risk)."** The decision path is fully readable — a doctor can see exactly why.

Diagram (draw this). The tree above as boxes and arrows: diamond boxes for questions, rectangles for leaves, with the example patient's path highlighted in colour.

Table: parts of a tree.

Part	Symbol	Role	Example
Root node	Top diamond	First, most informative question	"Dose > 350 mg?"
Decision node	Middle diamond	Follow-up question	"Age > 60?"
Branch	Arrow	One answer	"Yes" / "No"
Leaf	Rectangle	Final prediction	"ADR likely"

Key takeaway. A decision tree is a learned flowchart: the best questions, in the best order, ending in predictions you can read and explain.

5. Interpretation of decision paths and feature importance

Technical concept. Because each prediction follows one visible path, trees are *interpretable*: you can state the exact rule used. Feature importance aggregates, over all splits, how much each feature reduced impurity — sklearn reports it as numbers summing to 1.

Simple explanation. The decision path is the model's "reasoning diary" for one patient. Feature importance is the model's "report card" saying which measurements mattered most overall.

Worked example (verified by Python, hypothetical data). A tree trained on [age, severity score, dose] to predict cured (1) / not cured (0) reported feature importance **[1.0, 0.0, 0.0]** with depth 1. Interpretation: this tiny tree asked only *one* question — about age — and ignored the other features. That is honest but also a warning: a single shallow tree can be crude, which is why we use forests.

Pharmacy reading. If "kidney function" gets importance 0.6 and "age" 0.3 in an ADR tree, the pharmacist learns: kidney function drives the model's ADR calls most — a clinically sensible, checkable insight.

Key takeaway. Decision paths explain single predictions; feature importance ranks what the model relies on overall.

6. Random forest and the ensemble concept

Technical concept. A random forest builds many trees (e.g., 100), each on a random bootstrap sample of patients and considering a random subset of features at each split. Each tree votes; the majority wins. Randomness makes the trees diverse, and voting cancels individual trees' errors.

Simple explanation — the committee analogy. One junior doctor may misjudge a case. Ask 100 doctors, each shown a slightly different sample of past cases, and take the majority opinion — the committee is far more reliable than any single member. A random forest is exactly this committee of trees.

Why randomness helps. If every tree saw the same data, they would all make the same mistakes. Different samples + different features at each split = different mistakes, which cancel out in the vote. (This directly fights the high variance of single trees from Unit I.)

Worked example (verified by Python, hypothetical data). On the same treatment-outcome data: single tree accuracy = **1.0** (100%), random forest (10 trees) accuracy = **1.0** (100%). Feature importance from the forest: **[0.5, 0.3, 0.2]** — age 0.5, severity 0.3, dose 0.2. Interpretation: the forest uses *all three* features in balanced proportions, unlike the single tree that used only age — the committee sees the fuller picture.

Diagram (draw this). Three small trees side by side, each with slightly different questions, all pointing arrows into a ballot box labelled "majority vote" with one output arrow: "final prediction."

Key takeaway. Random forest = many diverse trees voting. More stable and usually more accurate than one tree, while still reporting feature importance.

7. Advantages and limitations of tree-based models

Table.

Advantages	Limitations
Easy to understand and explain to doctors/patients (white-box)	Single trees overfit easily — deep trees memorise noise
Handles both numerical and categorical features naturally	Small data changes can reshape the tree (unstable alone)
Needs little data preparation (no scaling required)	Biased toward features with many possible splits
Captures non-linear patterns automatically	Forests lose the single-tree readability (many trees to inspect)
Gives feature importance for clinical insight	Forests are slower to train than one tree
Works for both classification and regression	Very deep forests can still overfit tiny datasets

Pharmacy angle. The biggest advantage in healthcare is *explainability*: a doctor can follow the decision path and defend the decision to a patient — unlike "black-box" models. The biggest limitation is overfitting of single trees, fixed by pruning or by using a random forest.

Key takeaway. Trees: beautifully explainable but individually wobbly. Forests: steadier and stronger, slightly less transparent.

8. Pharma and clinical demonstrations

ADR risk stratification. A decision tree sorts patients into low/medium/high ADR-risk leaves using age, dose, kidney function and co-medications. Each leaf becomes a risk stratum with a clear rule ("age > 60 AND dose > 350 → high risk"), which the pharmacist can pin on the ward wall.

Patient classification. Classifying patients into treatment groups (e.g., "standard dose" vs "needs dose adjustment") from lab values — the tree's path documents exactly why each patient was classified so.

Predicting treatment outcomes. Random forest predicts cured/not cured from age, severity and dose; feature importance tells the team which factors drive outcomes, guiding protocol improvements.

9. Python demonstration with sklearn (every number verified)

Hypothetical data: features = [age, severity_score (1–10), dose_mg]; label = 1 (cured) / 0 (not cured); 16 patients.

```
import numpy as np
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Step 1: hypothetical data
X = np.array([
    [25, 2, 100], [30, 3, 150], [65, 8, 400], [70, 9, 450],
    [45, 4, 200], [55, 7, 350], [28, 2, 120], [68, 8, 420],
    [40, 5, 250], [60, 7, 380], [35, 3, 180], [72, 9, 500],
    [50, 6, 300], [33, 4, 220], [58, 7, 360], [48, 5, 280]])
y = np.array([1,1,0,0,1,0,1,0,1,0,1,0,0,1,0,1])

X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=4, random_state=7)
```

```
# Step 2: single decision tree
tree = DecisionTreeClassifier(random_state=7)
tree.fit(X_train, y_train)
print("Tree predictions:", tree.predict(X_test))
print("True labels     :", y_test)
print("Tree accuracy:", accuracy_score(y_test, tree.predict(X_test)))
print("Tree depth:", tree.get_depth())
print("Tree feature importance:", np.round(tree.feature_importances_, 3))
```

Output:

```
Tree predictions: [1 0 1 0]
True labels     : [1 0 1 0]
Tree accuracy: 1.0
Tree depth: 1
Tree feature importance: [1. 0. 0.]
```

Explanation: `DecisionTreeClassifier()` creates the tree; `.fit()` grows it by choosing the best questions. All 4 test predictions correct (accuracy 1.0). Depth 1 = the tree asked just one question. Feature importance [1.0, 0, 0] = it used only age — simple but crude.

```
# Step 3: random forest — the committee
forest = RandomForestClassifier(n_estimators=10, random_state=7)
forest.fit(X_train, y_train)
print("Forest predictions:", forest.predict(X_test))
print("Forest accuracy:", accuracy_score(y_test, forest.predict(X_test)))
print("Forest feature importance:",
      np.round(forest.feature_importances_, 3))
```

Output:

```
Forest predictions: [1 0 1 0]
Forest accuracy: 1.0
Forest feature importance: [0.5 0.3 0.2]
```

Explanation: `RandomForestClassifier(n_estimators=10)` builds 10 trees; each votes, majority wins. Accuracy also 1.0 here, but the importance [0.5, 0.3, 0.2] shows the forest weighs age (0.5), severity (0.3) and dose (0.2) — a fuller, more trustworthy picture than the single tree's age-only view. `random_state=7` keeps the random sampling repeatable.

10. Key points

- A decision tree is a learned flowchart: root → questions → leaves with predictions.
- Splitting criterion picks the question that best separates the classes at each node.
- Decision paths make single predictions fully explainable; feature importance ranks features overall.
- Single trees overfit easily (deep = memorising); prune them or use forests.
- Random forest = many trees on random data/feature samples + majority vote = stable and strong.
- Healthcare loves trees for explainability; forests for reliability.

11. Common confusions

- **Depth vs accuracy:** deeper is not always better — very deep trees overfit.
- **Feature importance = usefulness to the model**, not proof of medical causation.
- **Random forest is still supervised** — it learns from labelled data, just with many trees.
- **Trees don't need feature scaling** (unlike some other methods) — a practical exam point.

12. Common mistakes

- Growing one huge tree on a small dataset and trusting it completely.
- Reading a single tree's path as medical truth without validation on test data.
- Ignoring that forests, while accurate, are harder to explain case-by-case than one tree.
- Forgetting `random_state` — without it, results change on every run.

13. Mnemonic

"R-B-L" — **R**oot question, **B**ranches of answers, **L**eaf prediction. **"Forest = a committee of trees that votes"** — many doctors, one majority decision.

14. Infographic concept box — Unit IV

- Title: "Trees & Forests in One Page"
- Left: a small decision tree (dose → age → risk leaves) with one highlighted decision path.
- Center: feature-importance bar chart (age 0.5, severity 0.3, dose 0.2).
- Right: three mini-trees feeding a ballot box → "majority vote."
- Bottom strip: advantages (explainable ✔, no scaling ✔) vs limitation (single tree overfits ⚠ → use forest).
- Footer: "One tree explains; a forest decides reliably."

15. Exam summary

A decision tree is a supervised model shaped like a flowchart: the root asks the most informative yes/no question, branches carry the answers, and leaves give predictions. The splitting criterion chooses each question to separate classes as cleanly as possible. A decision path traces one case from root to leaf, making predictions explainable, and feature importance scores each feature's overall contribution. Single deep trees overfit, so we prune them or use a random forest — an ensemble of many trees trained on random data and feature samples whose majority vote is more stable and usually more accurate. In pharmacy, trees stratify ADR risk, classify patients and predict treatment outcomes; in sklearn, `DecisionTreeClassifier` and `RandomForestClassifier` implement them.

16. One-minute revision

- Tree = learned flowchart (root → branches → leaves).
- Split = best separating question at each node.
- Decision path explains one prediction; feature importance ranks features.
- One deep tree overfits → prune or use a forest.
- Random forest = many random trees + majority vote.
- Pharmacy: ADR risk strata, patient classification, outcome prediction.

17. Exam questions

Short answers:

1. Draw a tiny decision tree for ADR prediction with one root, one decision node and three leaves.
2. What is a decision path? Why does it matter to a doctor?
3. Define feature importance. What does [0.5, 0.3, 0.2] tell you?
4. What is an ensemble? How does a random forest use it?

Long answers:

5. Explain decision-tree structure and splitting with a pharmacy example. How do you interpret a decision path and feature importance?
6. Describe the random forest algorithm step by step. Why is it more stable than a single tree?
7. Discuss four advantages and four limitations of tree-based models in healthcare.

MCQs:

8. In a decision tree, the final prediction sits at the:

a) root b) branch c) leaf d) trunk *Answer: c*

9. A random forest's final prediction is decided by:

a) the deepest tree b) majority vote of trees c) averaging depths d) the first tree *Answer: b*

10. Feature importances from sklearn sum to:

a) 100 b) 1 c) number of trees d) number of features *Answer: b*

11. The main weakness of a single deep decision tree is:

a) underfitting b) overfitting c) slow prediction d) need for scaling *Answer: b*

Viva:

12. Your tree has depth 12 on 40 patients. What is the danger and the remedy?
13. A doctor asks "why did the model flag my patient as high risk?" — how does a decision tree let you answer?

Application-based:

14. Design an ADR risk-stratification tree: name 3 features, sketch two levels of questions, and state what each leaf means for the pharmacist's action.

18. References (Unit IV)

1. James, G., Witten, D., Hastie, T. and Tibshirani, R. *An Introduction to Statistical Learning with Applications in Python*. Springer. Available at: statlearning.com.
2. Géron, A. *Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow*. O'Reilly Media.
3. Brown, N. *Artificial Intelligence in Drug Discovery*. Royal Society of Chemistry.

COMBINED REFERENCES (ALL UNITS)

As listed in the official BP301T syllabus (preferably latest editions):

1. James, G., Witten, D., Hastie, T. and Tibshirani, R. An Introduction to Statistical Learning with Applications in Python. Springer. Available at: statlearning.com.
2. Simon, G. J. and Aliferis, C. Artificial Intelligence and Machine Learning in Health Care and Medical Sciences: Best Practices and Pitfalls. Springer.
3. Brown, N. Artificial Intelligence in Drug Discovery. Royal Society of Chemistry.
4. Walters, S. J., Campbell, M. J. and Machin, D. Medical Statistics: A Textbook for the Health Sciences. Wiley-Blackwell.
5. Géron, A. Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow. O'Reilly Media.