THE UNIVERSITY of TENNESSEE KNOXVILLE

# Lecture 05: Decision Trees, Forests, and Flowers

Instructor: Sergei V. Kalinin

- **Classification** is the process of identifying and grouping objects or ideas into predetermined categories

- **Classifiers** are the algorithms or models used to perform this task, assigning items to their appropriate categories based on learned patterns or rules

- These **patterns or rules** can be learned from **data**, or they may be derived from **domain knowledge**, **expert input**, or pre-existing **theoretical frameworks**

- **Decision trees** are a type of classifier that uses a tree-like model of decisions and their possible consequences, where each internal node represents a test on a feature, each branch represents the outcome of that test, and each leaf node represents a class label

https://www.techopedia.com/definition/13779/classification

# Not a good classification:

Animals are divided into

1. those that belong to the Emperor,
2. embalmed ones,
3. those that are trained,
4. suckling pigs,
5. mermaids,
6. fabulous ones,
7. stray dogs,
8. those that are included in this classification,
9. those that tremble as if they were mad,
10. innumerable ones,
11. those drawn with a very fine camel's hair brush,
12. others,
13. those that have just broken a flower vase,
14. those that resemble flies from a distance.

**Celestial Emporium of Benevolent Knowledge** – Jorge Luis Borges's fictional taxonomy of animals from his 1942 short story *The Analytical Language of John Wilkins*.

# Tennis Player Example

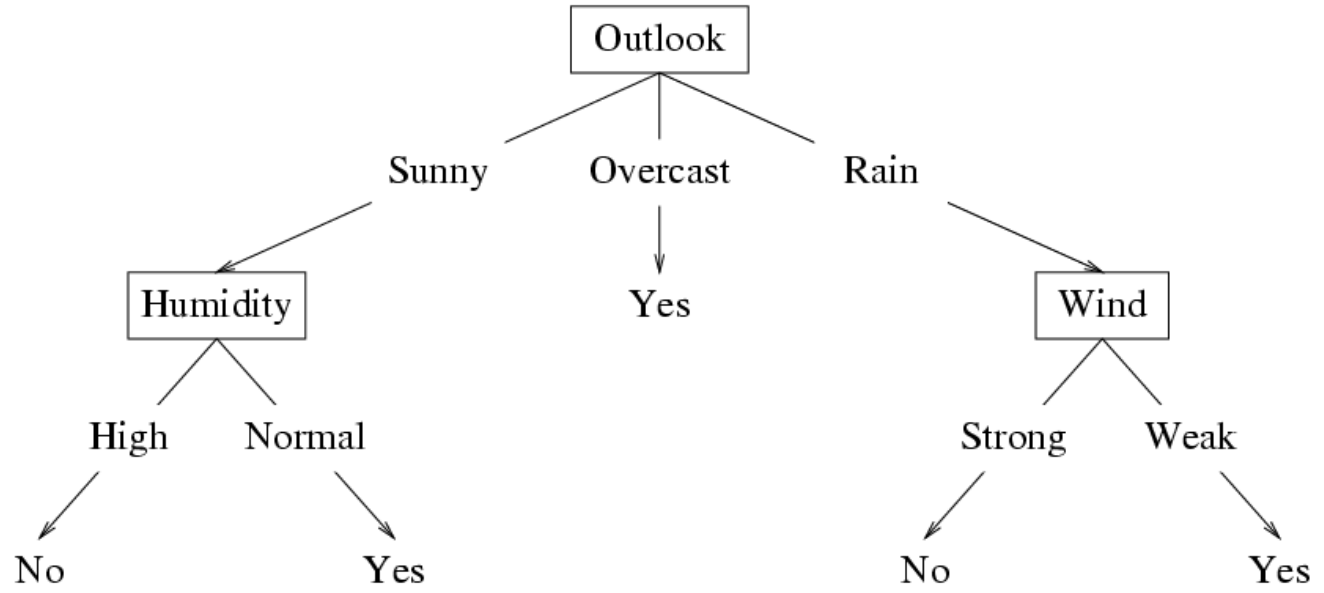
*PlayTennis*: training examples

| Day | Outlook | Temperature | Humidity | Wind | PlayTennis |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Decision Tree Hypothesis Space

- **Internal nodes** test the value of particular features $x_j$ and branch according to the results of the test.
- **Leaf nodes** specify the class $h(\mathbf{x})$.




Suppose the features are **Outlook** ($x_1$), **Temperature** ($x_2$), **Humidity** ($x_3$), and **Wind** ($x_4$). Then the feature vector $\mathbf{x} = (Sunny, Hot, High, Strong)$ will be classified as **No**. The **Temperature** feature is irrelevant.

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Classification and Regression Tree (CART)

- Create a set of questions that consists of all possible questions about the measured variables (define features)

- Select a splitting criterion (likelihood):

  - **Initialization:** create a tree with one node containing all the training data.

  - **Splitting:** find the **best question** for splitting each terminal node. Split the one terminal node that results in the greatest increase in the likelihood.

  - **Stopping:** if each leaf node contains data samples from the same class, or some pre-set threshold is not satisfied, stop. Otherwise, continue splitting.

  - **Pruning:** use an independent test set or cross-validation to prune the tree.

# How do we split?

**Information gain:**

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

- $f$ is the feature to perform the split
- $D_p$ and $D_j$ are the dataset of the parent and $j$th child node
- $I$ is our **impurity** measure
- $N_p$ is the total number of training examples at the parent node
- $N_j$ is the number of examples in the $j$th child node

**Binary split:**

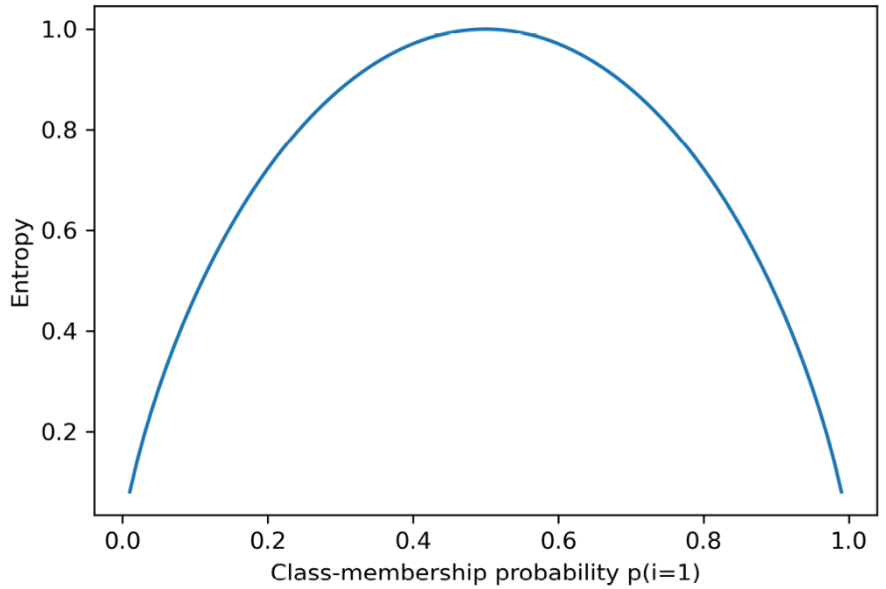
$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# What are impurity measures?

Entropy:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t) \log_2 p(i|t)$$




- $p(i|t)$ is the proportion of the examples that belong to class $i$ for a particular node, $t$.
- The entropy is 0 if all examples at a node belong to the same class, and entropy is maximal if we have a uniform class distribution.
- For binary class setting, the entropy is 0 if $p(i=1|t) = 1$ or $p(i=0|t) = 0$. If the classes are distributed uniformly with $p(i=1|t) = 0.5$ and $p(i=0|t) = 0.5$, the entropy is 1.

From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# What are impurity measures?

**Gini impurity:**

$$I_G(t) = \sum_{i=1}^{c} p(i|t)\,(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2$$

**Classification error:**

$$I_E(t) = 1 - \max\{p(i|t)\}$$

From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# Computing Information Gain

- Let's begin with the root node of the DT and compute $IG$ of each feature
- Consider feature "wind" $\in$ {weak,strong} and its $IG$ w.r.t. the root node

| day | outlook | temperature | humidity | wind | play |
|---|---|---|---|---|---|
| 1 | sunny | hot | high | weak | no |
| 2 | sunny | hot | high | strong | no |
| 3 | overcast | hot | high | weak | yes |
| 4 | rain | mild | high | weak | yes |
| 5 | rain | cool | normal | weak | yes |
| 6 | rain | cool | normal | strong | no |
| 7 | overcast | cool | normal | strong | yes |
| 8 | sunny | mild | high | weak | no |
| 9 | sunny | cool | normal | weak | yes |
| 10 | rain | mild | normal | weak | yes |
| 11 | sunny | mild | normal | strong | yes |
| 12 | overcast | mild | high | strong | yes |
| 13 | overcast | hot | normal | weak | yes |
| 14 | rain | mild | high | strong | no |

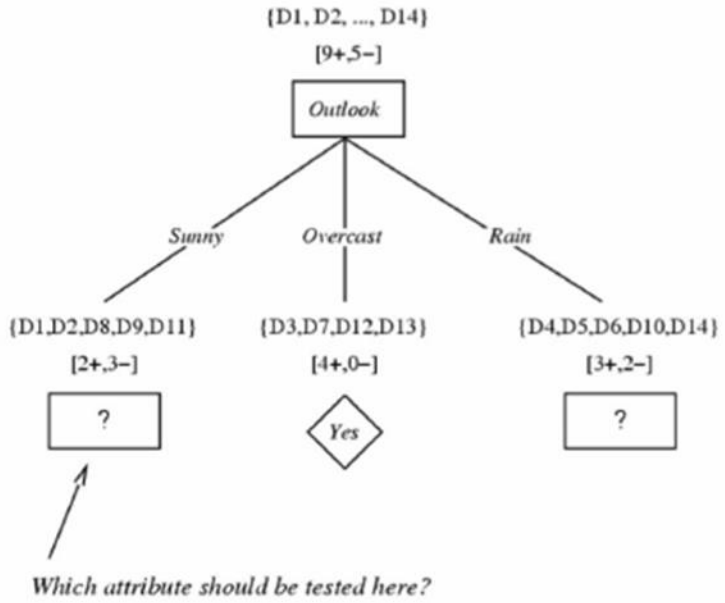
- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \Longrightarrow H(S_{weak}) = 0.811$
- $S_{strong} = [3+, 3-] \Longrightarrow H(S_{strong}) = 1$

$$\begin{aligned} IG(S, \text{wind}) &= H(S) - \frac{|S_{weak}|}{|S|} H(S_{weak}) - \frac{|S_{strong}|}{|S|} H(S_{strong}) \\ &= 0.94 - 8/14 * 0.811 - 6/14 * 1 \\ &= 0.048 \end{aligned}$$

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

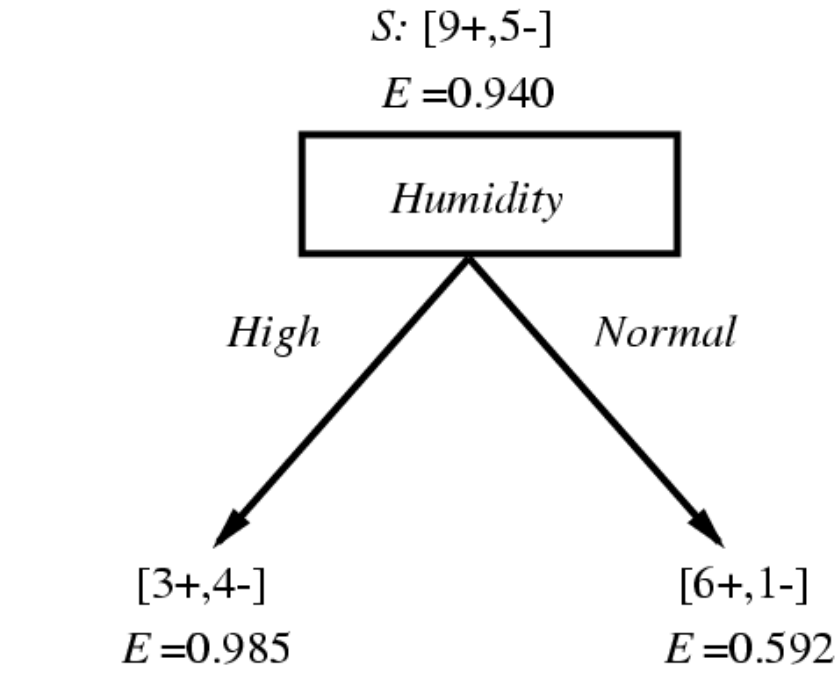
# Choosing the Most Informative Feature

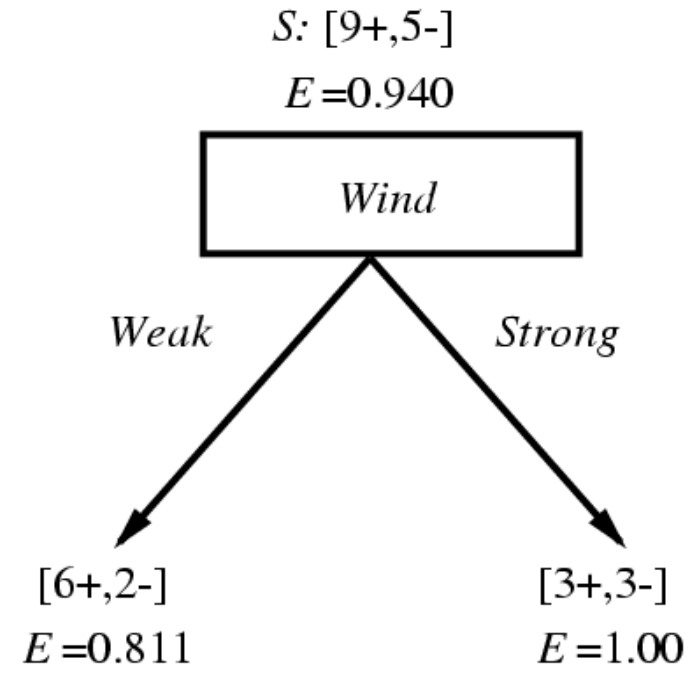
- At the root node, the information gains are:
  - $IG(S, \text{wind}) = 0.048$ (we already saw)
  - $IG(S, \text{outlook}) = 0.246$
  - $IG(S, \text{humidity}) = 0.151$
  - $IG(S, \text{temperature}) = 0.029$
- "outlook" has the maximum $IG \Longrightarrow$ chosen as the root node
- Growing the tree:
  - Iteratively select the feature with the highest information gain for each child of the previous node




Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Selecting the Next Attribute




*Gain (S, Humidity )*

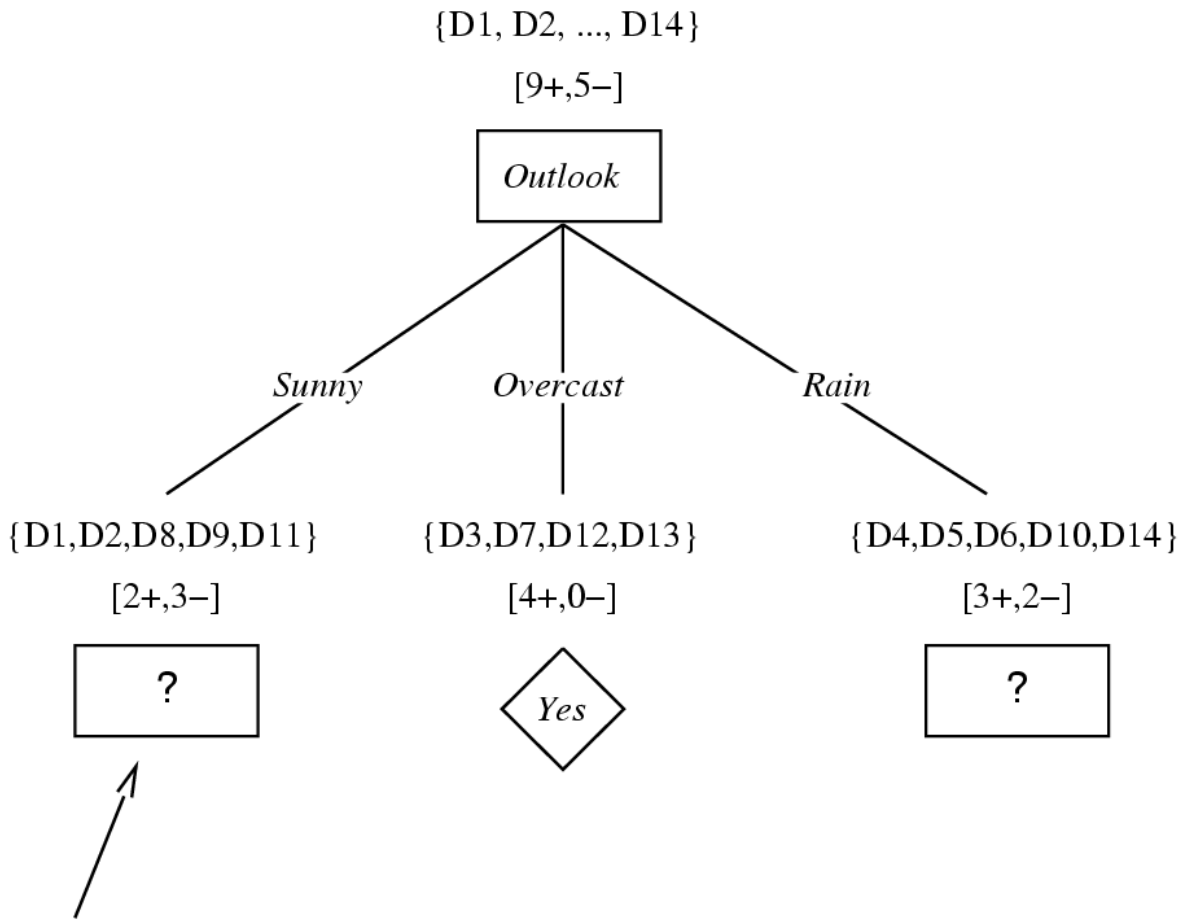
= .940 - (7/14).985 - (7/14).592
= .151

*Gain (S, Wind)*

= .940 - (8/14).811 - (6/14)1.0
= .048

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# And so on...




*Which attribute should be tested here?*

$S_{sunny}$ = {D1,D2,D8,D9,D11}

*Gain* ($S_{sunny}$, *Humidity*) = .970 − (3/5) 0.0 − (2/5) 0.0 = .970

*Gain* ($S_{sunny}$, *Temperature*) = .970 − (2/5) 0.0 − (2/5) 1.0 − (1/5) 0.0 = .570

*Gain* ($S_{sunny}$, *Wind*) = .970 − (2/5) 1.0 − (3/5) .918 = .019

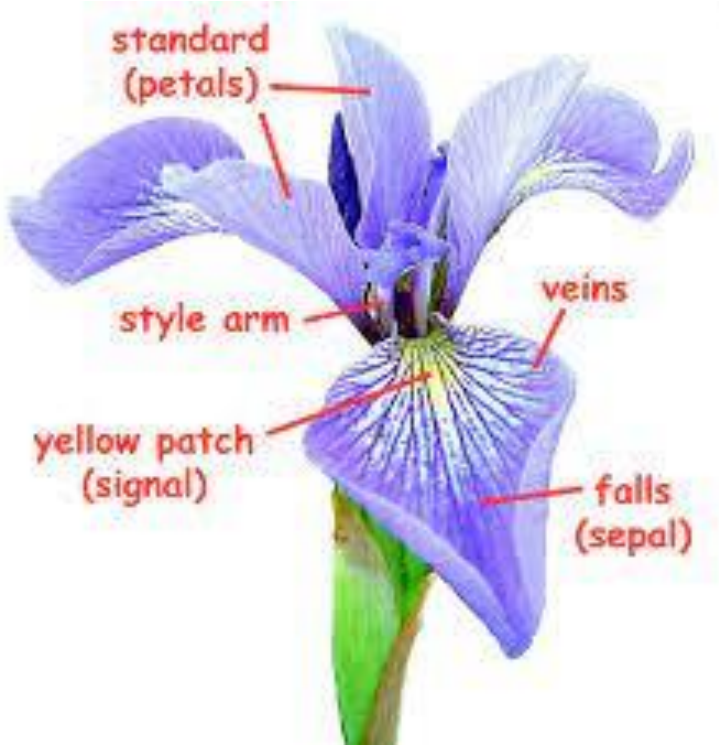
Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Let’s do some classification!


http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html

# Let's do some classification!

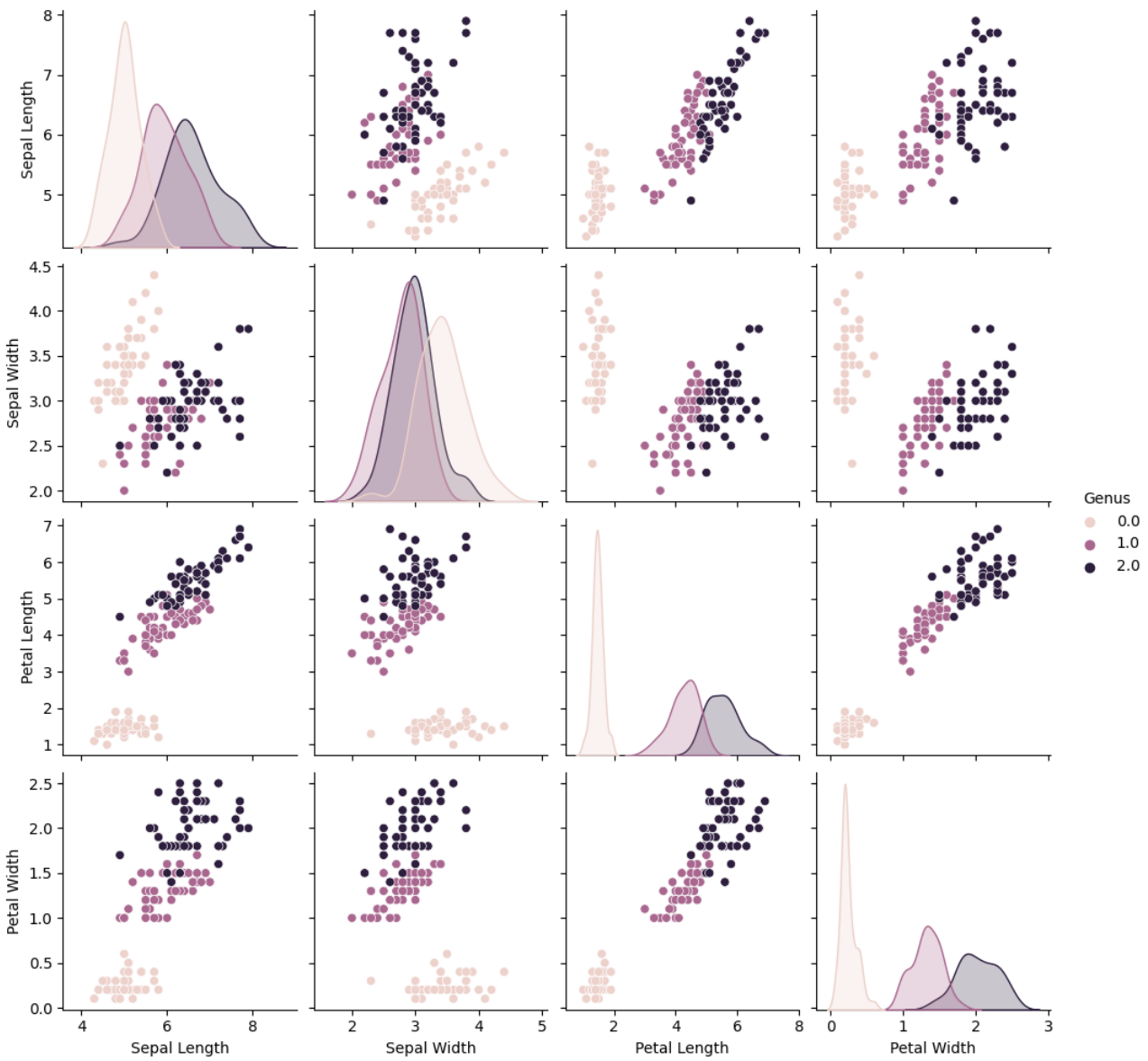
| | Sepal length | Sepal width | Petal length | Petal width | |
|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 |
| 145 | 6.7 | 3.0 | 5.2 | 2.3 | Iris-virginica |
| 146 | 6.3 | 2.5 | 5.0 | 1.9 | Iris-virginica |
| 147 | 6.5 | 3.0 | 5.2 | 2.0 | Iris-virginica |
| 148 | 6.2 | 3.4 | 5.4 | 2.3 | Iris-virginica |
| 149 | 5.9 | 3.0 | 5.1 | 1.8 | Iris-virginica |



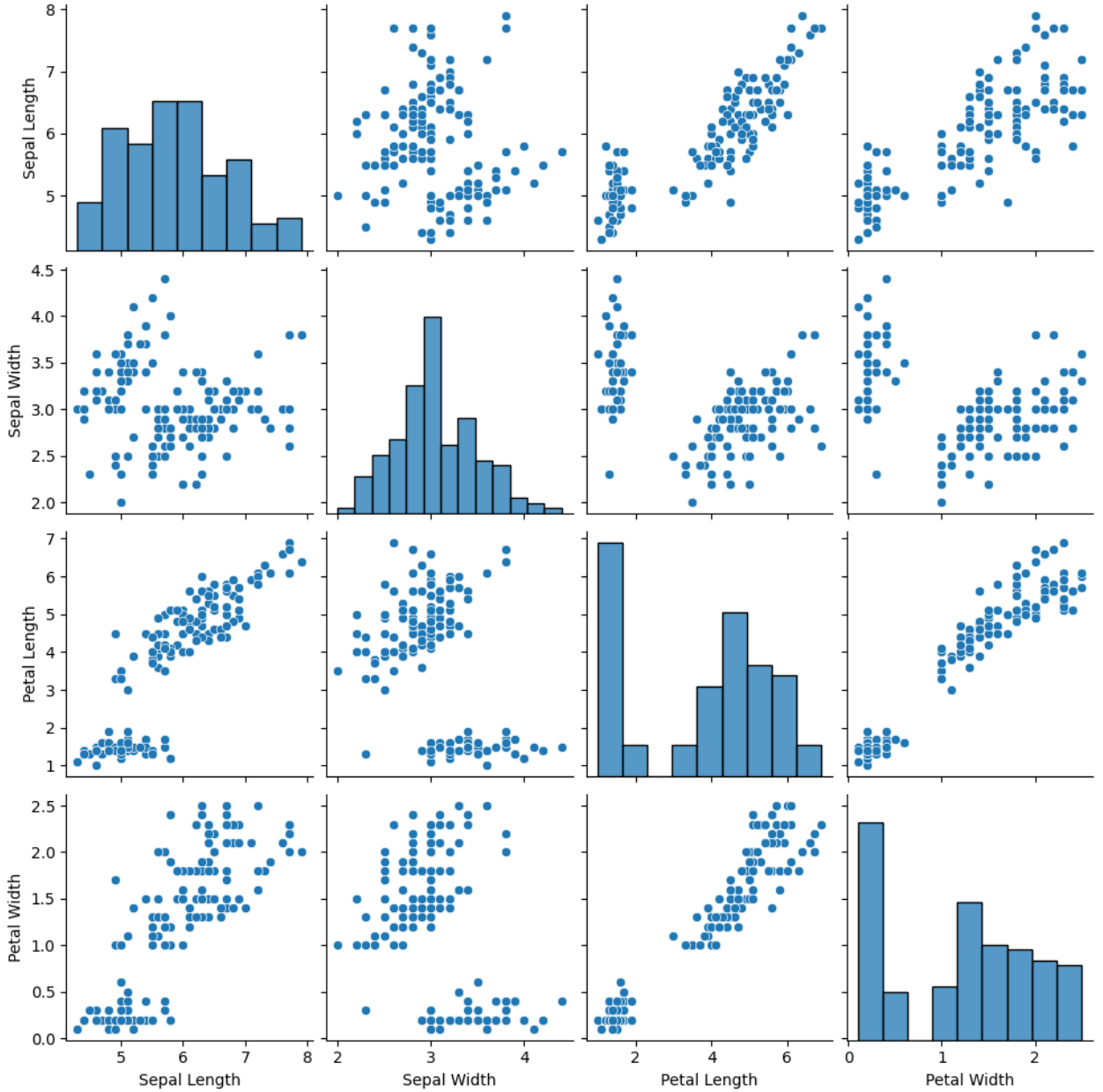

In scikit-learn, Iris-setosa, Iris-versicolor, and Iris-virginica, are already stored as integers

From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

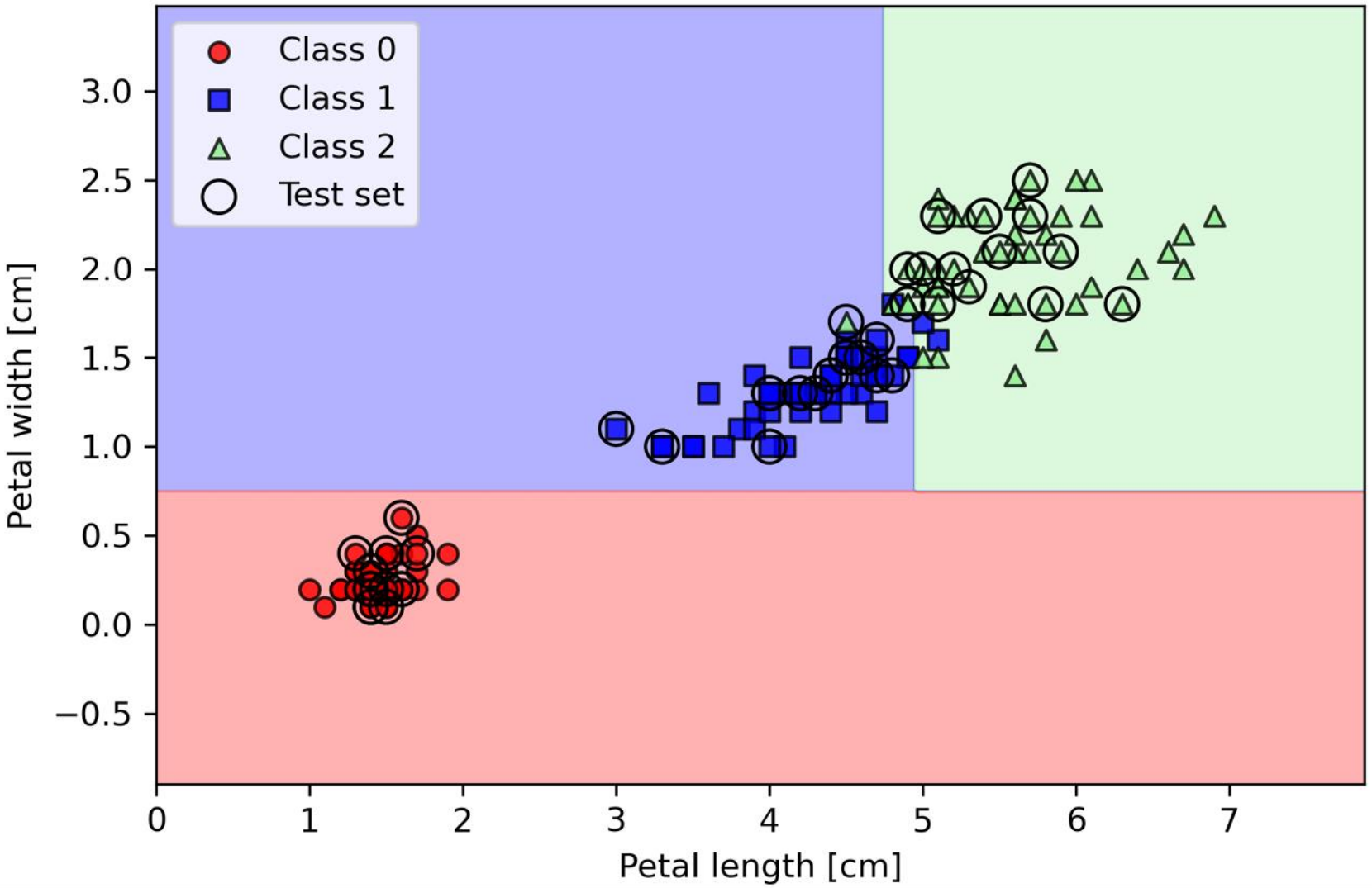
# But first, exploratory data analysis…



# Can we do it if there is no labels?

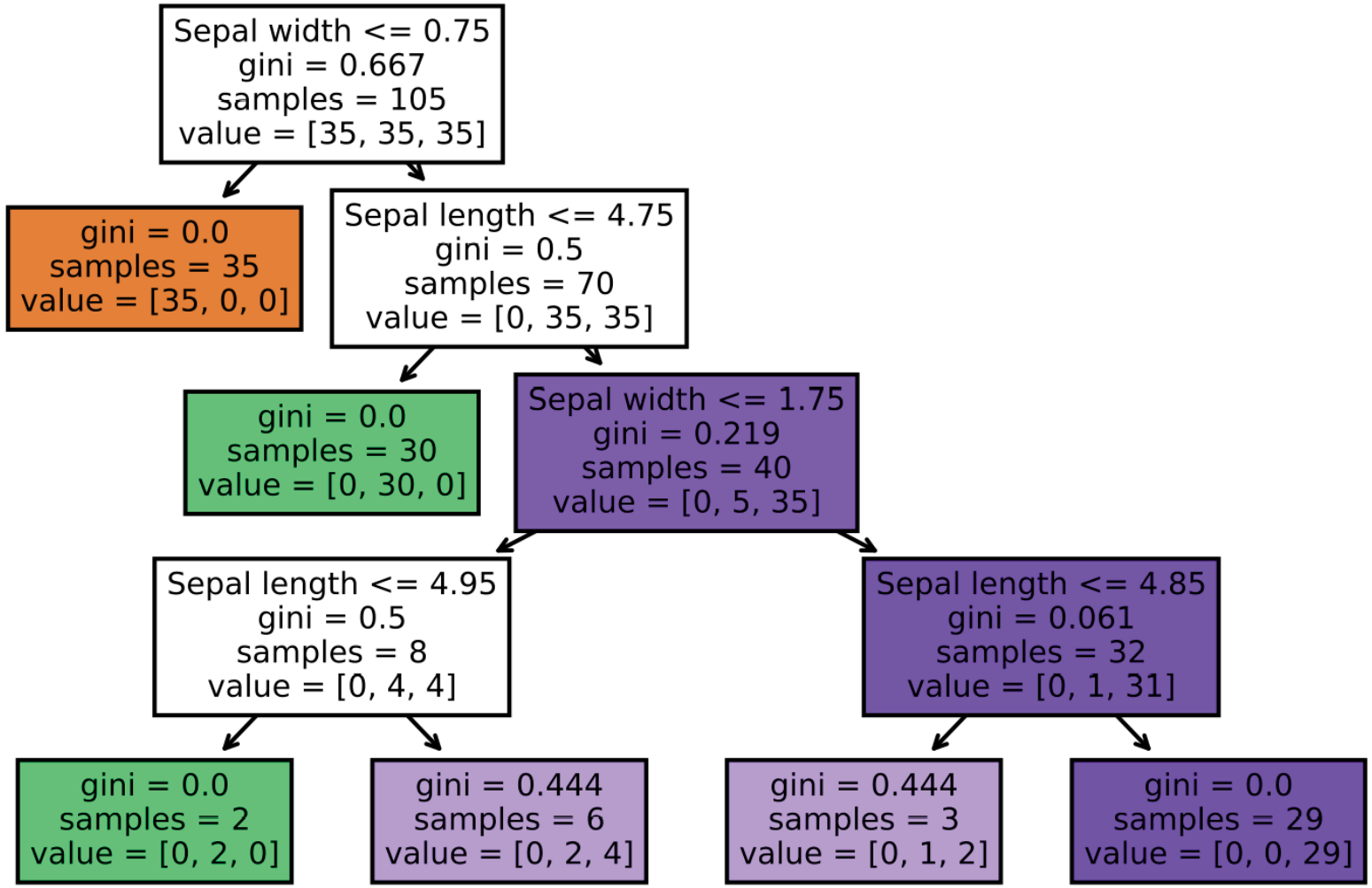


# Visualization




From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# Running Decision Tree Algorithm




From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# Decision Tree are adaptable:

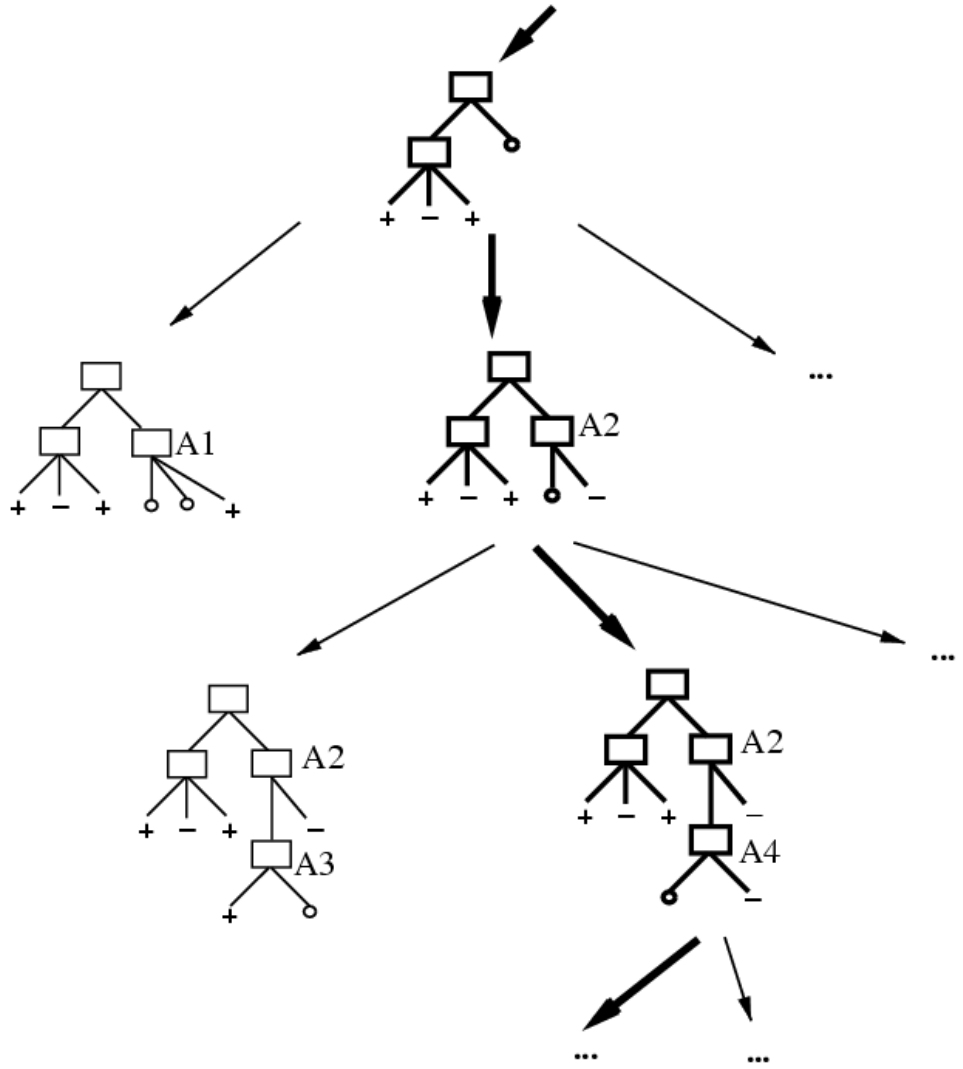
- Features with multiple discrete values
  - Multi-way splits
  - Test for one value versus the rest
  - Group values into disjoint sets

- Real-valued features
  - Use thresholds

- Regression
  - Splits based on mean squared error metric

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Hypothesis Space Search




You do not get the globally optimal tree!

Search space is exponential.

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Overfitting

Consider error of hypothesis $h$ over

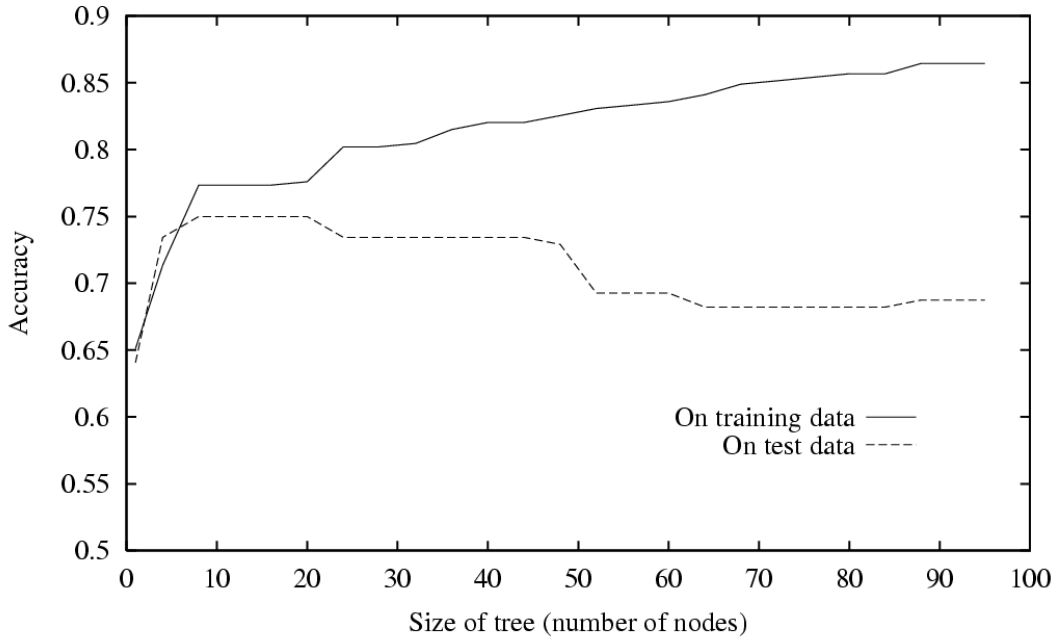
- training data: $error_{train}(h)$
- entire distribution $\mathcal{D}$ of data: $error_{\mathcal{D}}(h)$

Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

$$error_{train}(h) < error_{train}(h')$$

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$




- Prune tree to reduce error on validation set

Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

# Solution: Decision Tree Forest

1. Draw a random **bootstrap** sample of size *n* (randomly choose *n* examples from the training dataset with replacement).

2. Grow a decision tree from the bootstrap sample. At each node:
   a. Randomly select *d* features without replacement.
   b. Split the node using the feature that provides the best split according to the objective function, for instance, maximizing the information gain.

3. Repeat *steps 1-2 k* times.

4. Aggregate the prediction by each tree to assign the class label by **majority vote**.

From S. Raschka, Machine Learning with PyTorch and Scikit-Learn

# Advantages of Decision Trees - I

- Simple to understand and interpret. People are able to understand decision tree models after a brief explanation. Trees can also be displayed graphically in a way that is easy for non-experts to interpret.
- Able to handle both numerical and categorical data.
- Requires little data preparation. Other techniques often require data normalization. Since trees can handle qualitative predictors, there is no need to create dummy variables.
- Uses a white box or open-box model. If a given situation is observable in a model the explanation for the condition is easily explained by boolean logic. By contrast, in a black box model, the explanation for the results is typically difficult to understand, for example with an artificial neural network.
- Possible to validate a model using statistical tests. That makes it possible to account for the reliability of the model.

https://en.wikipedia.org/wiki/Decision_tree_learning

# Advantages of Decision Trees - II

- Non-parametric approach that makes no assumptions of the training data or prediction residuals; e.g., no distributional, independence, or constant variance assumptions
- Performs well with large datasets. Large amounts of data can be analyzed using standard computing resources in reasonable time.
- Mirrors human decision making more closely than other approaches.
- Robust against co-linearity, particularly boosting.
- In built feature selection. Additional irrelevant feature will be less used so that they can be removed on subsequent runs. The hierarchy of attributes in a decision tree reflects the importance of attributes. It means that the features on top are the most informative.
- Decision trees can approximate any Boolean function e.g. XOR.

https://en.wikipedia.org/wiki/Decision_tree_learning

# Disadvantages of Decision Trees

- Trees can be very non-robust. A small change in the training data can result in a large change in the tree and consequently the final predictions.
- The problem of learning an optimal decision tree is known to be NP-complete under several aspects of optimality and even for simple concepts. Consequently, practical decision-tree learning algorithms are based on heuristics such as the greedy algorithm where locally optimal decisions are made at each node. Such algorithms cannot guarantee to return the globally optimal decision tree.
- Decision-tree learners can create over-complex trees that do not generalize well from the training data (overfitting.) Mechanisms such as pruning are necessary to avoid this problem
- The average depth of the tree that is defined by the number of nodes or tests till classification is not guaranteed to be minimal or small under various splitting criteria.
- For data including categorical variables with different numbers of levels, information gain in decision trees is biased in favor of attributes with more levels.

https://en.wikipedia.org/wiki/Decision_tree_learning

# Conclusion:

- Decision trees are the single most popular data mining tool
  - Easy to understand
  - Easy to implement
  - Easy to use
  - Computationally cheap
- It's possible to get in trouble with overfitting
- They do classification: predict a categorical output from categorical and/or real inputs