THE UNIVERSITY of TENNESSEE KNOXVILLE

Lecture 06: Representing the World, Classification, and Decision Trees

Instructor: Sergei V. Kalinin

What are our objects?

Quantitative Variables:
- 3 apples, 2 oranges

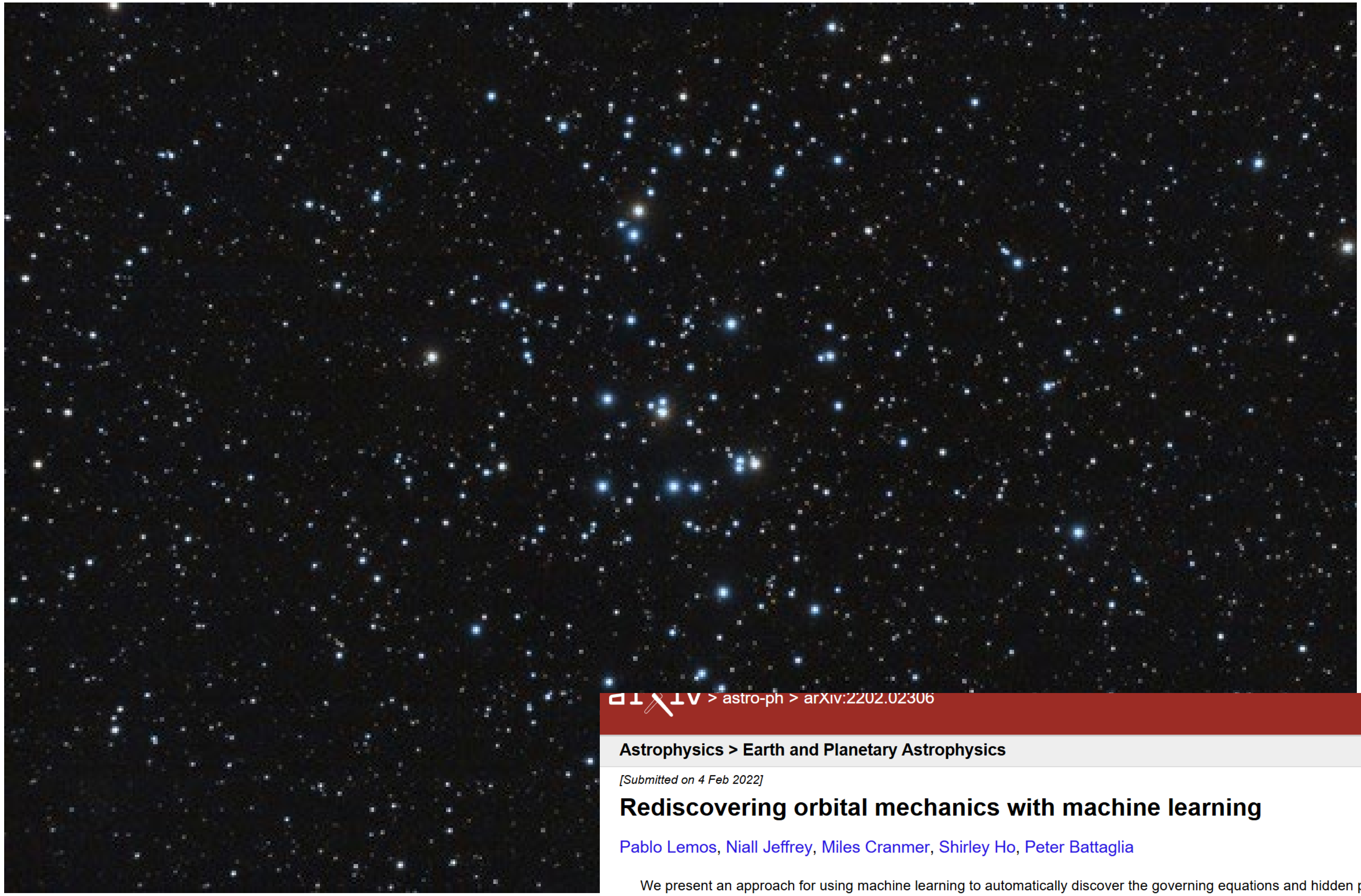
Categorical Variables:
- Apple and Orange

Schemas:
- "Fruit," "Type," and "Quantity"

Ontologies:
- "Apple is a type of Fruit"

Knowledge Graphs:
- Connection between "Apple," "Fruit," "Vitamin C content," and "Growth Regions."

Why does it matter?



arXiv > astro-ph > arXiv:2202.02306 Help |

Astrophysics > Earth and Planetary Astrophysics

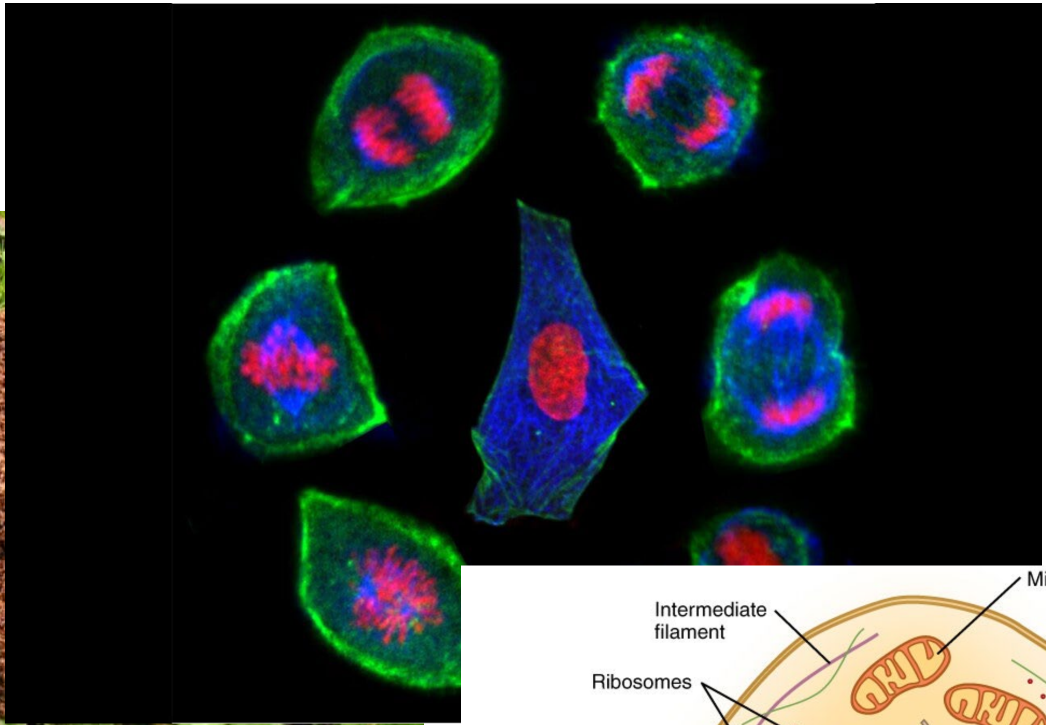
[Submitted on 4 Feb 2022]

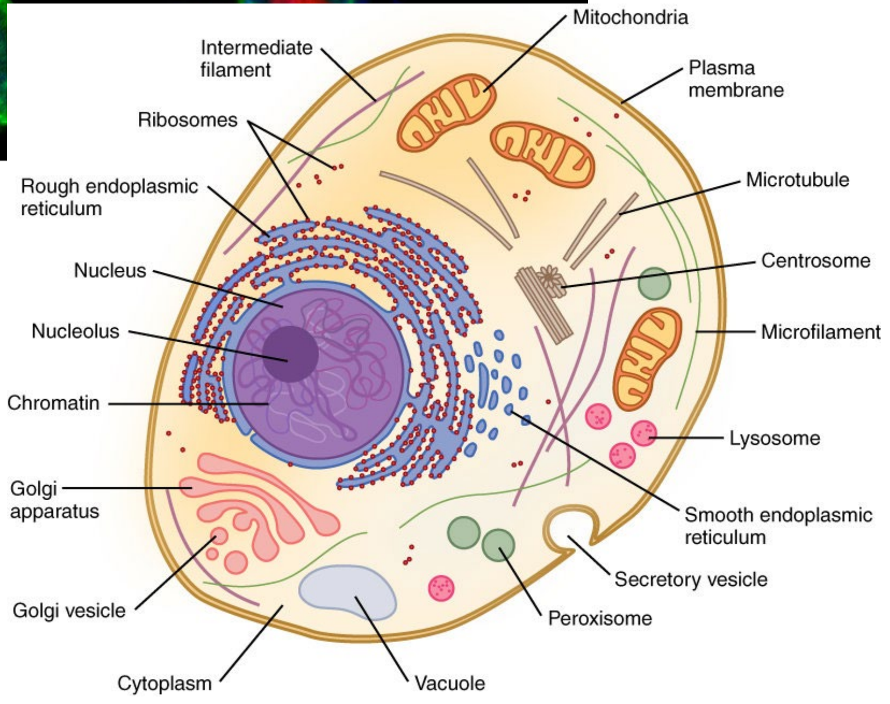
Rediscovering orbital mechanics with machine learning

Pablo Lemos, Niall Jeffrey, Miles Cranmer, Shirley Ho, Peter Battaglia

We present an approach for using machine learning to automatically discover the governing equations and hidden properties of real physical systems from observations. We train a "graph neural network" to simulate the dynamics of our solar system's Sun, planets, and large moons from 30 years of trajectory data. We then use symbolic regression to discover an analytical expression for the force law implicitly learned by the neural network, which our results showed is equivalent to Newton's law of gravitation. The key assumptions that were required were translational and rotational equivariance, and Newton's second and third laws of motion. Our approach correctly discovered the form of the symbolic force law. Furthermore, our approach did not require any assumptions about the masses of planets and moons or physical constants. They, too, were accurately inferred through our methods. Though, of course, the classical law of gravitation has been known since Isaac Newton, our result serves as a validation that our method can discover unknown laws and hidden properties from observed data. More broadly this work represents a key step toward realizing the potential of machine learning for accelerating scientific discovery.

No equations (or ML) without representation!





https://phys.org/news/2021-10-cell-life-optical-tweezers.html

https://doi.org/10.1080/00150193.2020.1722012

https://www.pointepestcontrol.net/inside-the-ant-hill/

https://www.researchgate.net/figure/Schematic-representation-of-a-cell-20-with-organelles-delimited-or-made-of-lipid_fig5_350133282

What are our objects?

Haiku by Matsuo Bashō reading "*Quietly, quietly, / yellow mountain roses fall – / sound of the rapids*"

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

Schemas and SQL

A **schema** is a blueprint that defines the structure of a database, outlining how data is organized into tables, fields, and relationships. A schema might define table entries, showing how they relate to each other.

SQL (Structured Query Language) is a standard language used to manage and query data in relational databases. SQL relies on schemas to define and enforce the structure of data, ensuring consistency and integrity.

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

We can represent data with schemas in Python using:

- **Dictionaries:** Flexible, key-value pairs.

```
student = {"StudentID": 1, "Name": "Alice", "Major": "Physics"}
```

- **Pandas DataFrame:** Powerful structure for handling tabular data

```
import pandas as pd
df = pd.DataFrame([{"StudentID": 1, "Name": "Alice", "Major": "Physics"}])
```

- **JSON:** For storing data in a structured text format.

```
import json json_data = json.dumps(student)
```

Ontologies

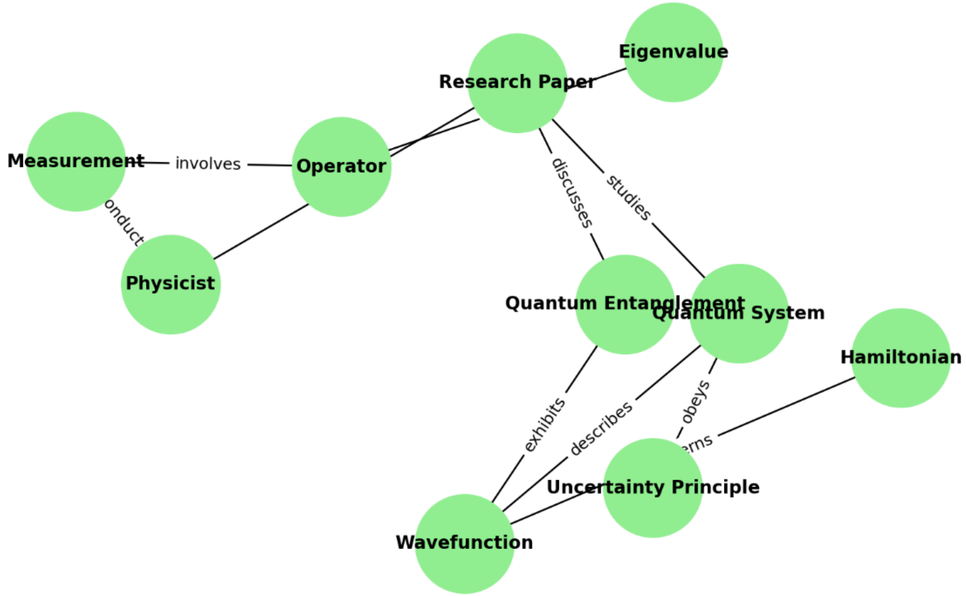
- A structured framework that defines a set of concepts and categories within a domain, along with the relationships between them.
- **Vocabulary:** A set of standardized terms used to describe concepts and relationships within an ontology. Terms like "Material," "Property," and "Process" in a materials science ontology.
- **Semantics:** The meaning and interpretation of the terms and relationships in an ontology. "Material has Property" describes the relationship between a material and its attributes.
- **Syntax:** The formal structure or rules that govern how information is represented and encoded in an ontology. OWL (web ontology language) syntax defines how classes and properties are written, RDF (resource description framework) syntax uses triples (subject, predicate, object).
- **Classes**: Categories or types of things within an ontology. "Mammal" as a class in a biological ontology.
- **Instances:** Specific examples or members of a class. "Dog" as an instance of the class "Mammal."
- **Relations:** The ways in which concepts or instances are connected within an ontology. "Material undergoes Process" to describe how a material is involved in a manufacturing step.

Knowledge Graphs

A knowledge graph is a network of interconnected entities (nodes) and their relationships (edges), often built upon ontologies.

- Nodes represent entities or concepts (e.g., "Person," "Material," "Document").
- Edges represent relationships between entities (e.g., "authored," "relatedTo," "hasProperty").
- Knowledge graphs are often constructed using ontologies to define the types of nodes and edges, ensuring consistency and meaning across the graph.

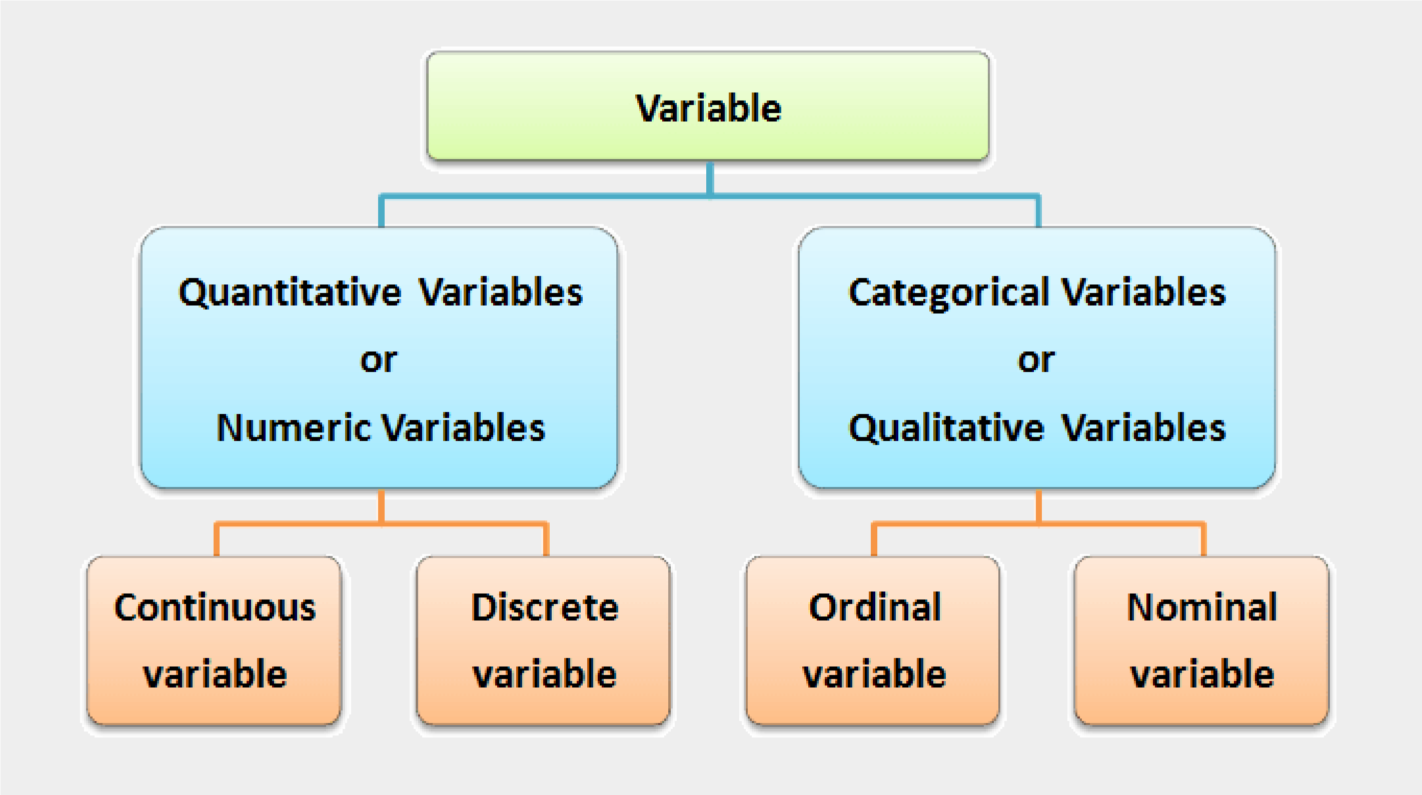
Example of a Knowledge Graph in Quantum Mechanics



How Knowledge Graphs are Used:

- Data Integration: Combine and link data from multiple sources, making it easier to query and analyze complex information.
- Semantic Search: Enable more precise and context-aware search capabilities by understanding the relationships between entities.
- Inference and Reasoning: Use the relationships in the graph to infer new knowledge or validate existing relationships.

Type of variables



Nominal variable: no order

Colors: blue, red, yellow, white

Dog breed: Labrador, German shepherd, poodle

Ordinal variable: there is order

Letter grades: A, B, C, …

Severity of the software bug: mild, serious, critical

https://prinsli.com/categorical-variables/

We have got the data… Now what?

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

- In data management, classification enables the separation and sorting of data according to set requirements for various business or personal objectives.

- In machine learning (ML), classification is used in predictive modeling to assign input data with a class label.

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

- **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

Classification Workflow

1. Selecting features and collecting labeled training examples
2. Choosing a performance metric
3. Choosing a learning algorithm and training a model
    - Can we understand how it works?
    - Does it have any hyperparameters
    - How well does it generalize to new data?
    - How expensive is it?
4. Evaluating the performance of the model
5. Changing the settings of the algorithm and tuning the model.

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

There are many classifiers:

Choosing an appropriate classification algorithm for a particular problem task requires practice and experience; each algorithm has its own quirks and is based on certain assumptions.

To paraphrase the **no free lunch theorem** by David H. Wolpert, no single classifier works best across all possible scenarios (*The Lack of A Priori Distinctions Between Learning Algorithms*, *Wolpert, David H*, *Neural Computation 8.7* (1996): 1341-1390).

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

Hello, I am Akinator

Think about a real or fictional character.
I will try to guess who it is

akinator®

Inactive
Sensitive content

Download on the App Store

PLAY

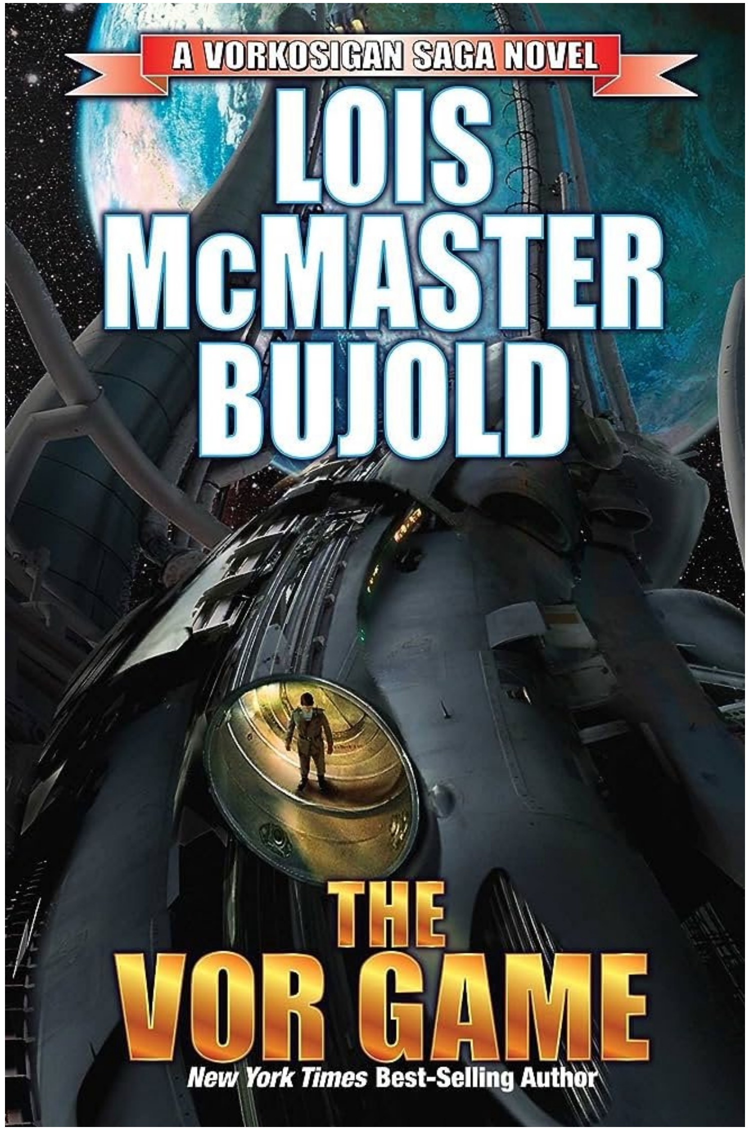
GET IT ON Google Play

824 people are playing right now.
631563588 games played 30599 today.

Famous sportsmen Akinator can guess

Playing a classifier game:

1. Is your character a pokemon: No
2. Does your character have legs: Yes
3. Is your character a girl: No
4. Is your character famous youtuber: No
5. Is your character real: No
6. Does your character have human head: Yes
7. Is your character from Japanese anime: No
8. Wear a mask: No
9. Originally from video game: No
10. Animated: No
11. Played in superhero movie: No
12. From a book: Yes
13. Have been in movie: No
14. Have powers: No
15. Popular tv show: No
16. Have phone: yes
17. Adult: Yes
18. Killed people: Yes
19. Live in future: Yes
20. Been in space: Yes
21. Wife dead: No
22. Linked with metal suit: No
23. Short: Yes
24. Father famous: Yes



Answer: Miles Vorkosigan

Tennis Player Example

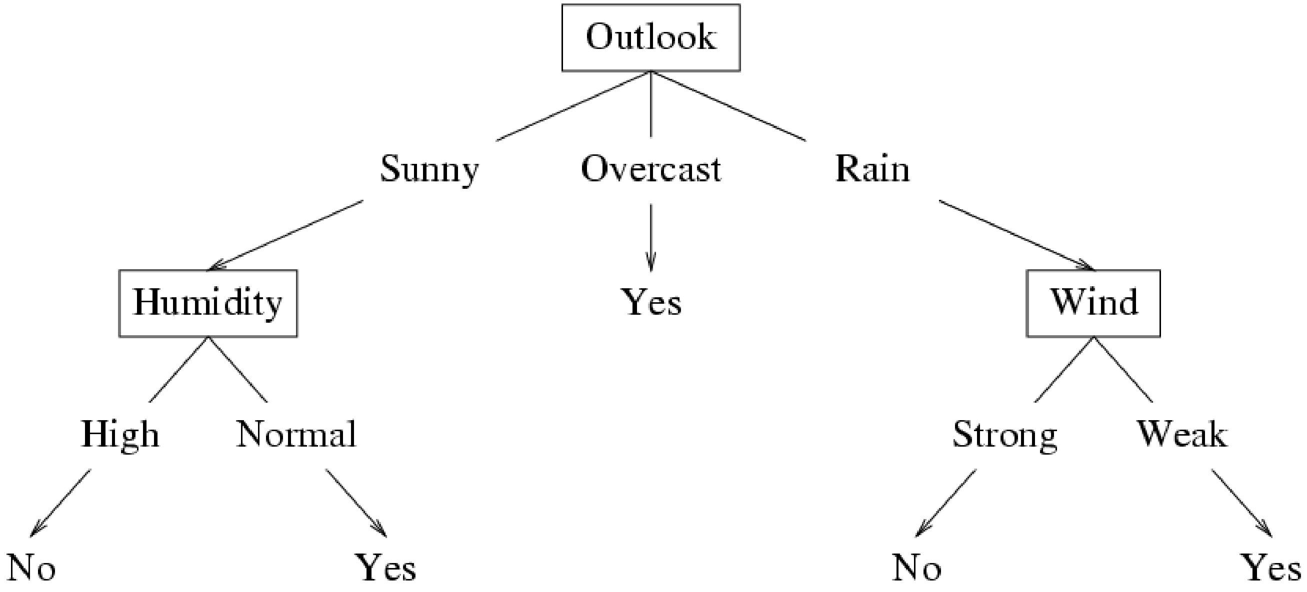
PlayTennis: training examples

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 $h(\mathbf{x})$.

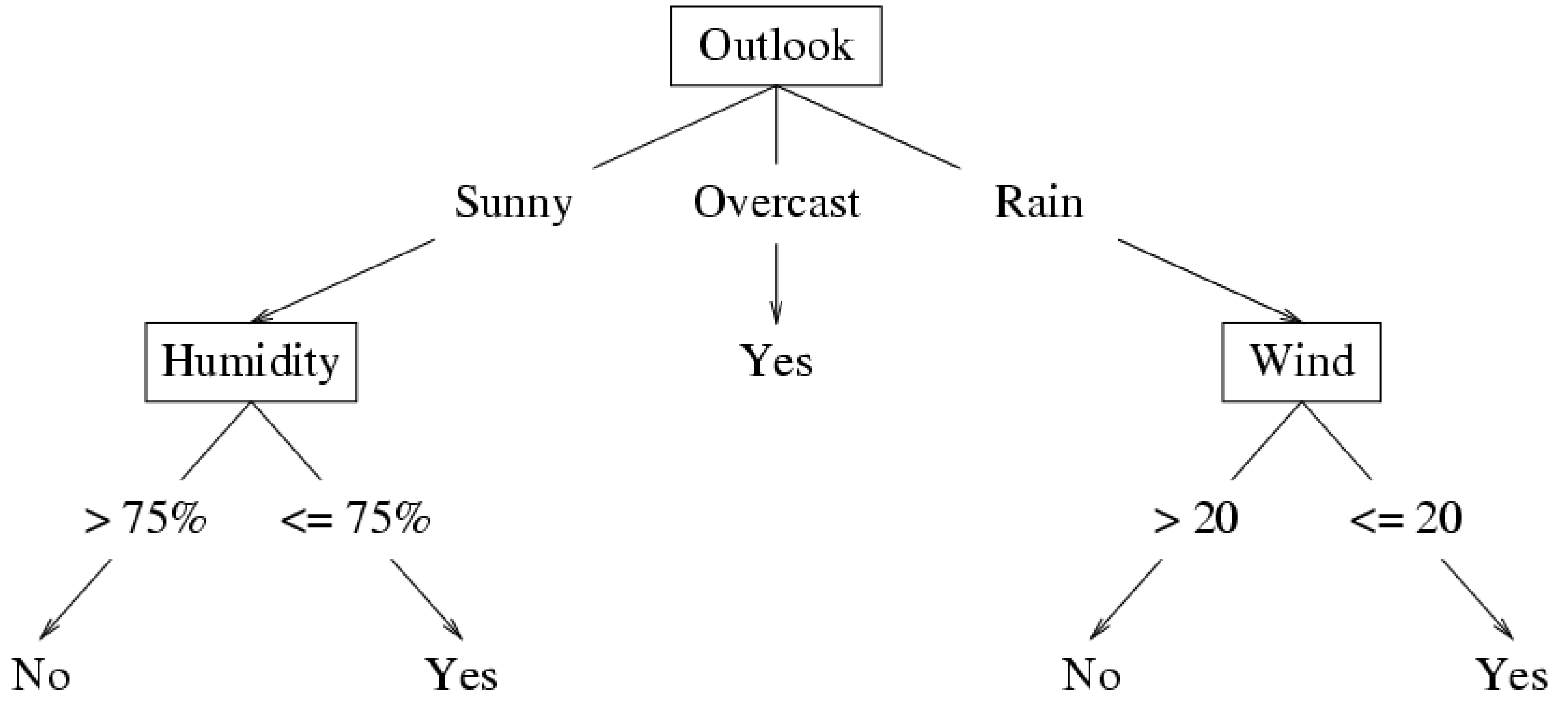


Suppose the features are **Outlook** (x_1), **Temperature** (x_2), **Humidity** (x_3), and **Wind** (x_4). Then the feature vector $\mathbf{x} = (Sunny, Hot, High, Strong)$ will be classified as **No**. The **Temperature** feature is irrelevant.

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

What if we have continuous variables?

If the features are continuous, internal nodes may test the value of a feature against a threshold.

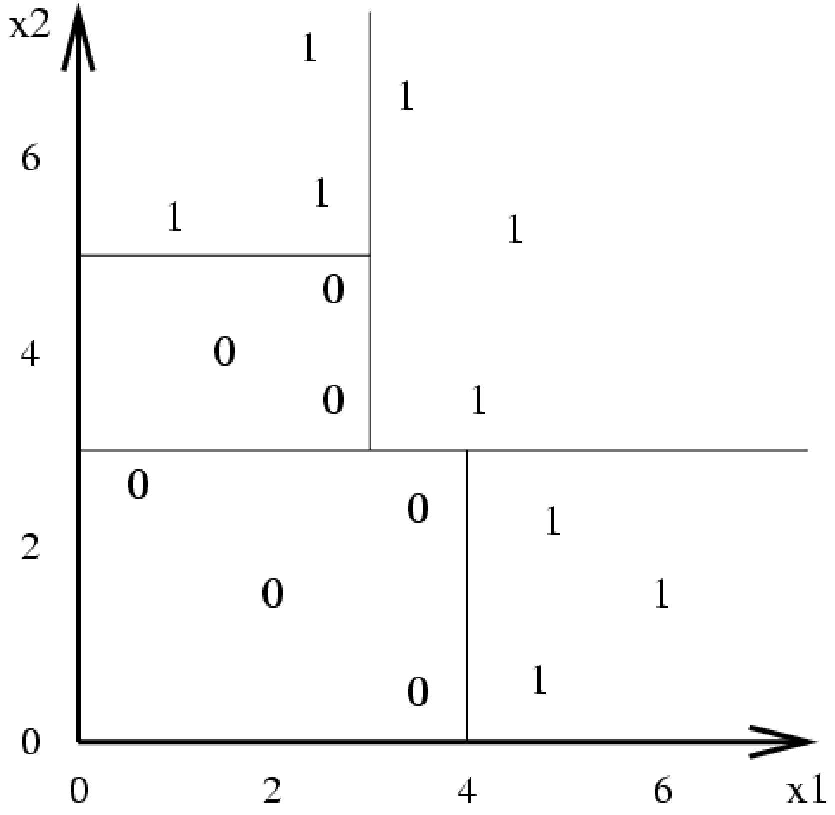


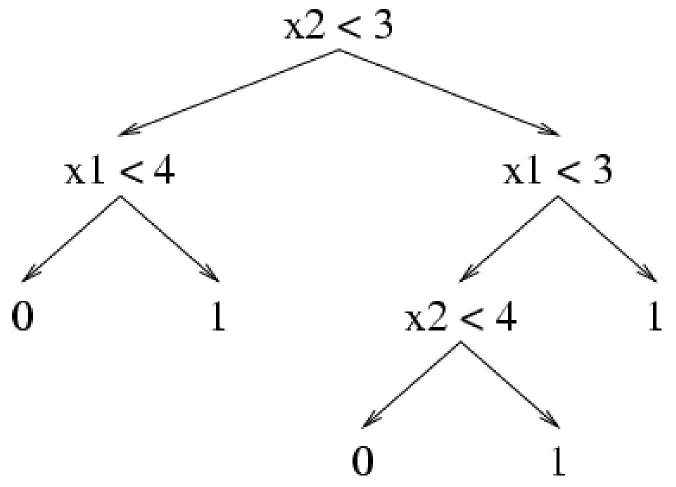
Also: regression trees

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

Decision Tree Boundaries

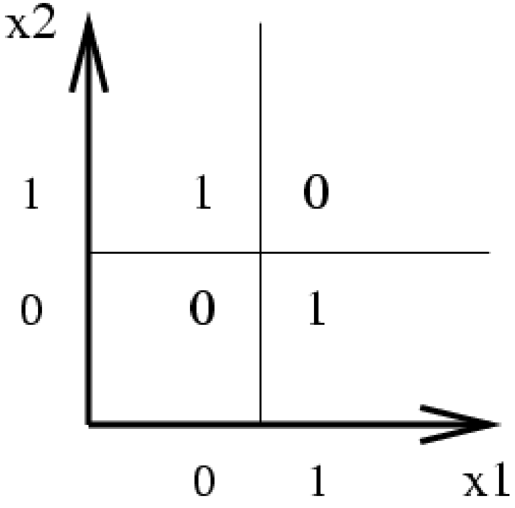
Decision trees divide the feature space into axis-parallel rectangles, and label each rectangle with one of the K classes.

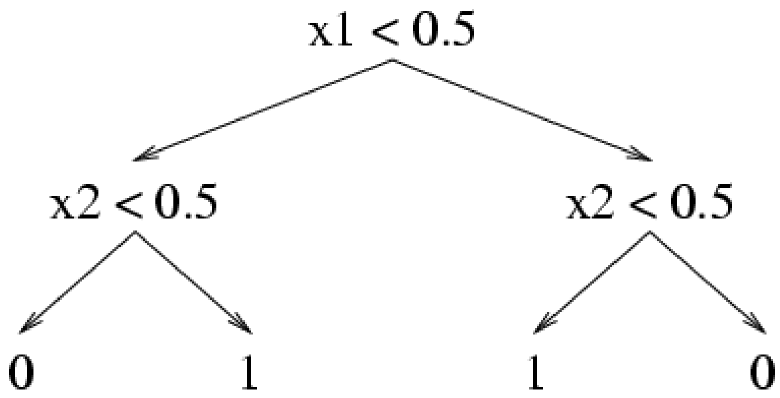




Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

Can Represent Any Boolean Function





The tree will in the worst case require exponentially many nodes, however.

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.

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*.

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 jth 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 jth child node

Binary split:

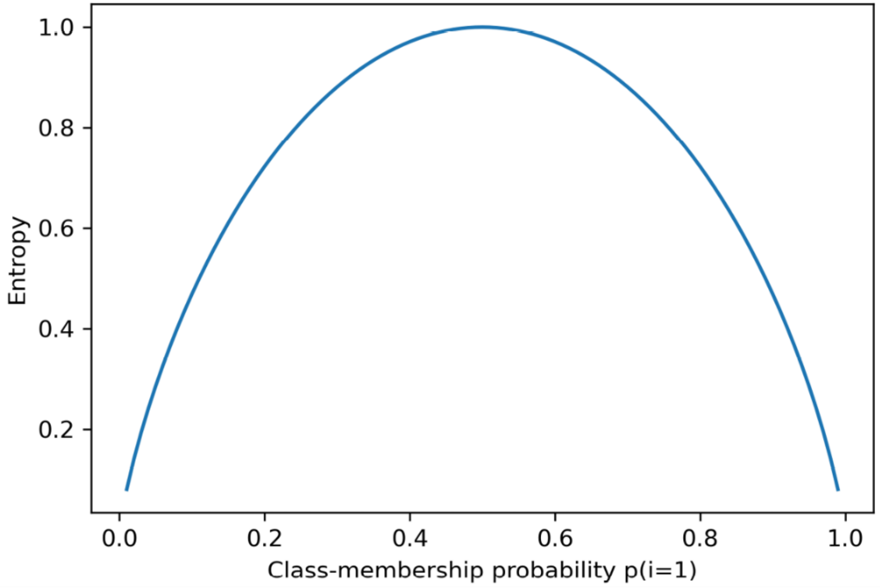
$$IG(D_p, f) = I(D_p) - \frac{N_{left}}{N_p} I(D_{left}) - \frac{N_{right}}{N_p} I(D_{right})$$

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

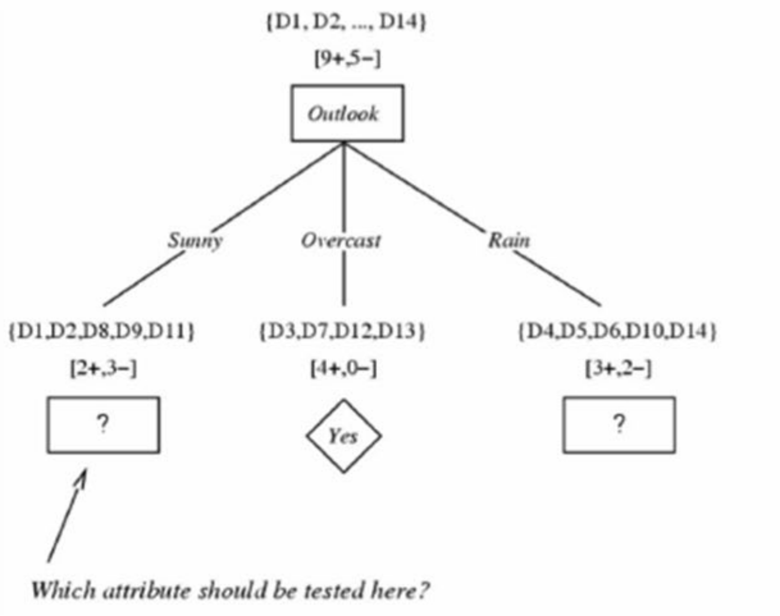
- Root node: $S = [9+, 5-]$ (all training data: 9 play, 5 no-play)
- Entropy: $H(S) = -(9/14)\log_2(9/14) - (5/14)\log_2(5/14) = 0.94$
- $S_{weak} = [6+, 2-] \Longrightarrow H(S_{weak}) = 0.811$
- $S_{strong} = [3+, 3-] \Longrightarrow H(S_{strong}) = 1$

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

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

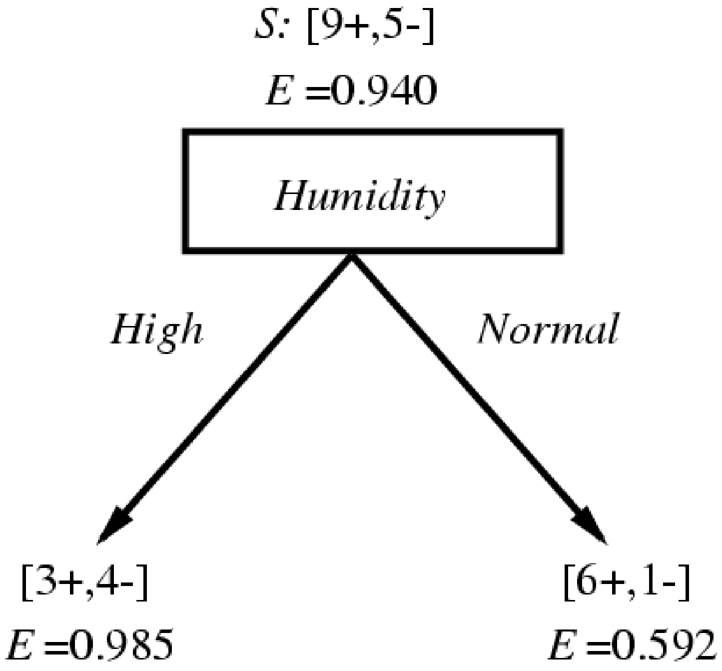
Choosing the Most Informative Feature

- 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



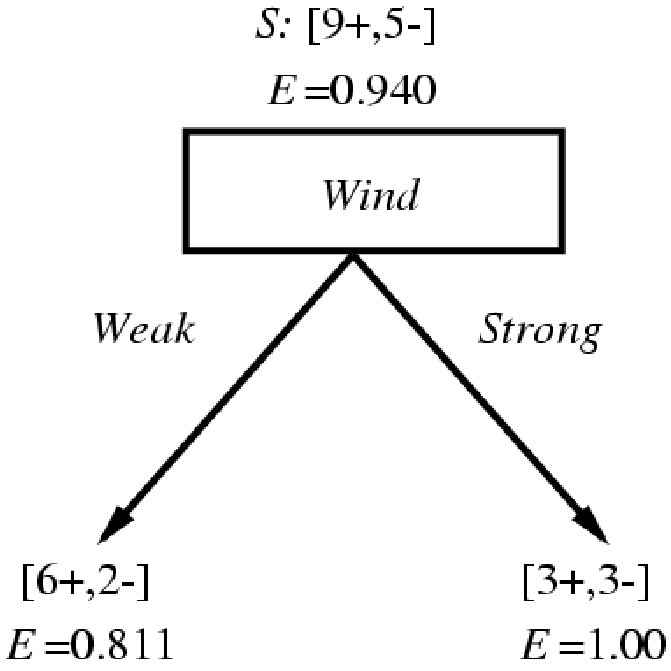
Adapted from Greg Grudic, Decision Trees (Notes borrowed from Thomas G. Dietterich and Tom Mitchell)

Selecting the Next Attribute



Gain (S, Humidity)

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

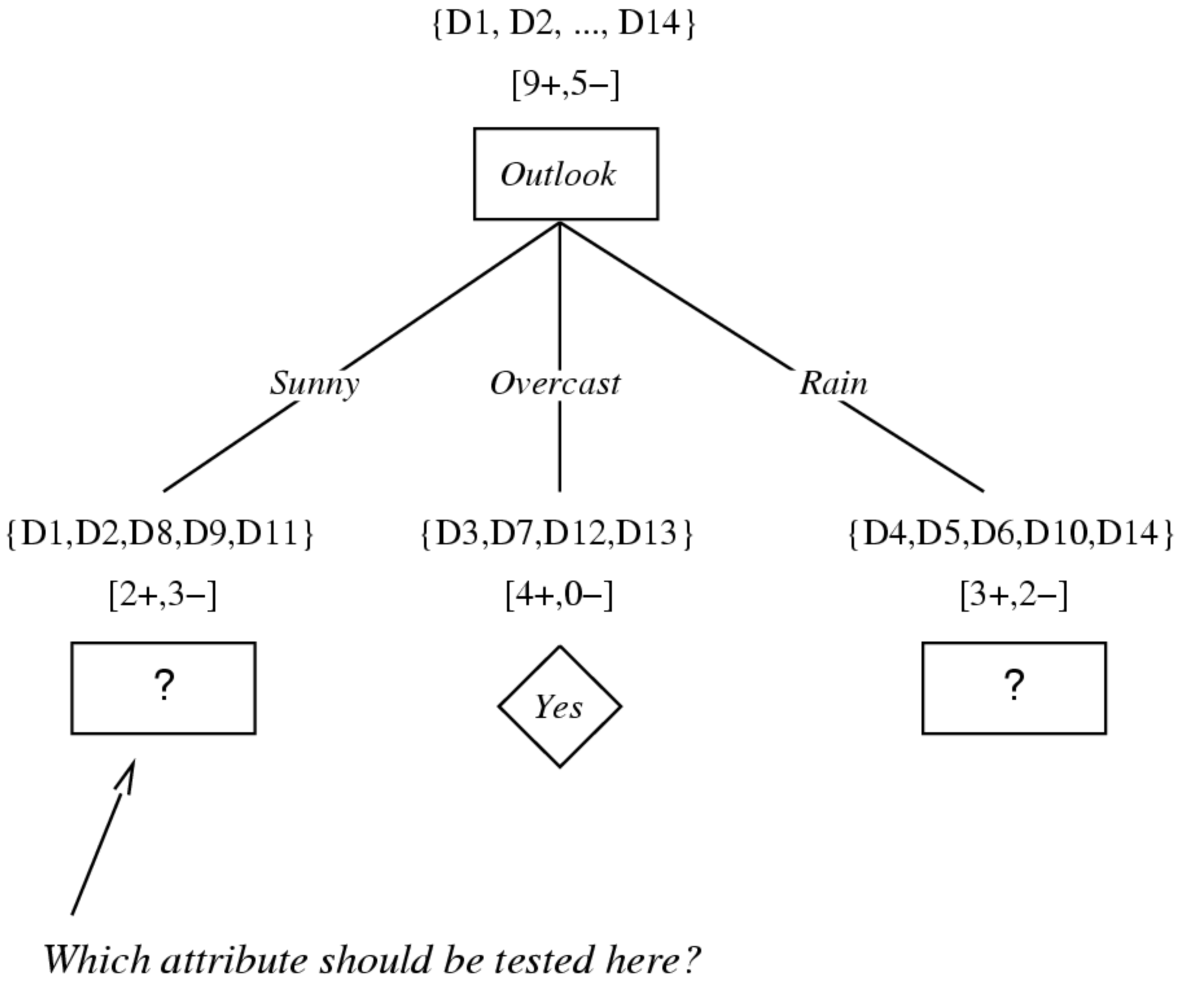


Gain (S, Wind)

= .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...



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

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

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

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

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