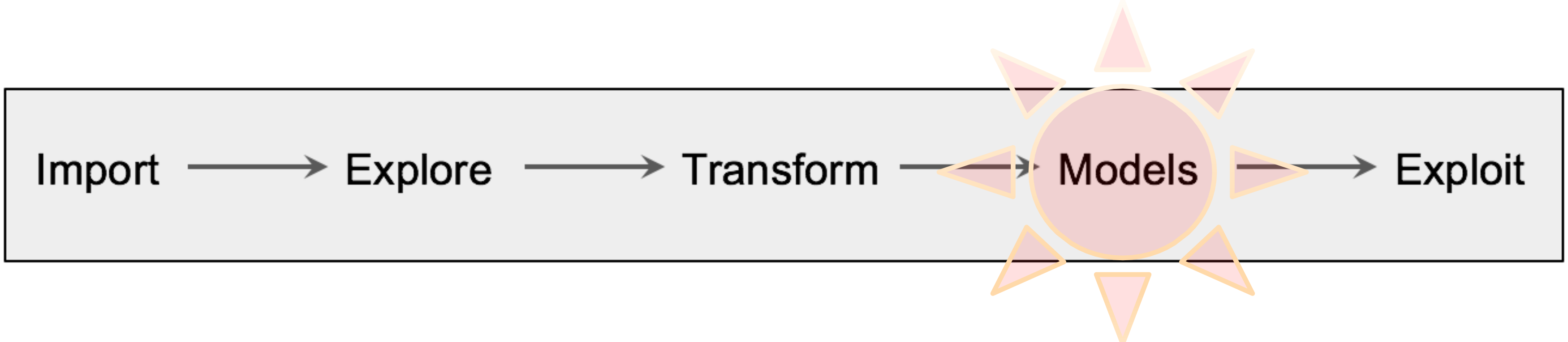
The Pipeline



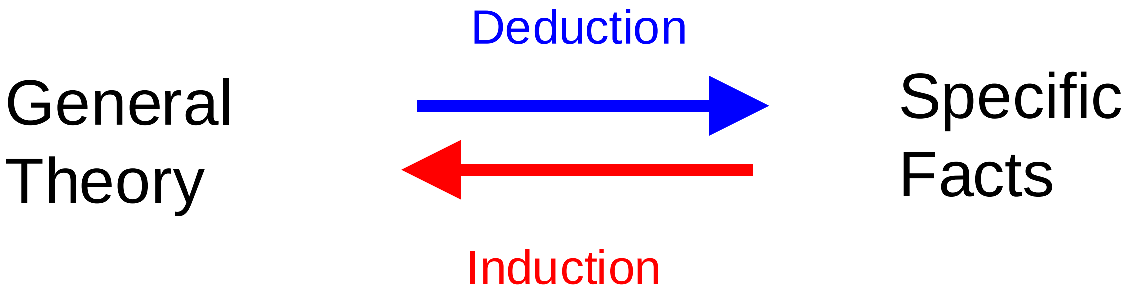
- We have looked at importing, exploring and transforming data.
- Now it is time to do something with our data: **MODELS**!
- Here we discuss one of the most straight forward machine learning models: **the decision tree**.

Machine Learning

- What is Machine Learning?
  - Programs that get better with *experience* given a *task* and some *performance measure.*
    - Learning to classify news articles
    - Learning to recognize spoken words
    - Learning to play board games
    - Learning to navigate (e.g. self-driving cars)
- Usually involves some sort of <u>inductive reasoning</u> step.

Inductive Reasoning

- Deductive reasoning (rule based reasoning)
  - From the general to the specific
- Inductive reasoning
  - From the specific to the general



Note: not to be confused with mathematical induction!

Example - Deduction

- Rules:
  - If Betty wears a white dress then it is Sunday.
  - Betty wears a white dress.
- Deductive step:
  - You infer or *deduce* that today is Sunday.

If Betty wears a white dress then it is Sunday.
Betty wears a white dress.

$\xrightarrow{\text{Deduction}}$

Today is Sunday.

Inference is the act or process of drawing a conclusion based solely on what one already knows.

Example - Induction

- Facts: every time you see a swan you notice that the swan is white.
- Inductive step: you infer that all swans are white.

All Swans are white. $\xleftarrow{\text{Induction}}$ Observed Swans are white.

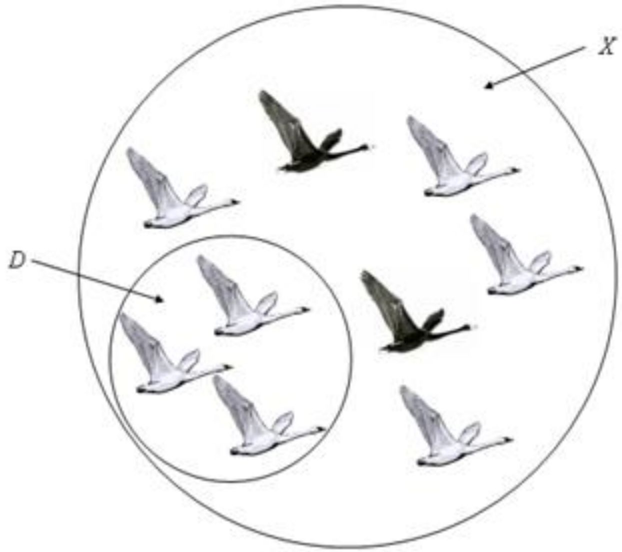
Inference is the act or process of drawing a conclusion based solely on what one already knows.

Observation

- Deduction is “truth preserving”
  - If the rules employed in the deductive reasoning process are sound, then, what holds in the theory will hold for the deduced facts.
- Induction is NOT “truth preserving”
  - It is more of a statistical argument
  - The more swans you see that are white, the more probable it is that all swans are white.. **But this does not exclude the existence of black swans**

Observation

D ≡ observations
X ≡ universe of all swans



This is called the Black Swan Problem and is the classic example posed by the philosopher of science Karl Popper in the early twentieth century. It roughly states that learning/induction is always a probabilistic argument since we can only learn from a limited number of observations (D) and make generalization from those on the universe at large (X). On a more technical level it argues this point based on *falsifiability of a hypothesis*.

Different Styles of Machine Learning

- Supervised Learning
  - The learning needs explicit examples of the concept to be learned (e.g. white swans, playing tennis, *etc*)
- Unsupervised Learning
  - The learner discovers autonomously any structure in a domain that might represent an interesting concept

Knowledge - Representing what has been learned

- Symbolic Learners (transparent models)
  - If-then-else rules
  - Decision trees
  - Association rules
- Sub-Symbolic Learners (non-transparent models)
  - (Deep) Neural Networks
  - Clustering (Self-Organizing Maps, k-Means)
  - Support Vector Machines

Decision Trees

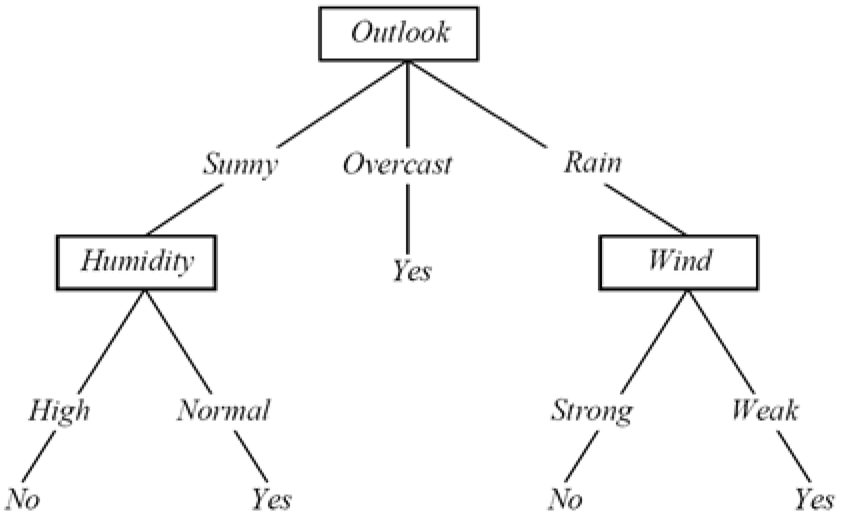
- Learn from labeled observations - supervised learning
- Represent the knowledge learned in form of a tree

Example: learning when to play tennis.

- Examples/observations are days with their observed characteristics and whether we played tennis or not

Play Tennis Example

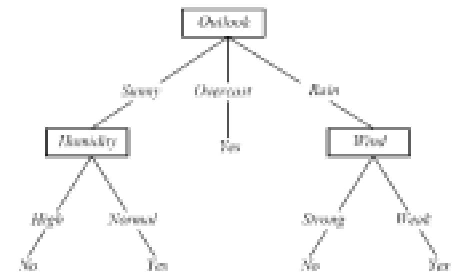
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No



Decision Tree Learning

Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Induction



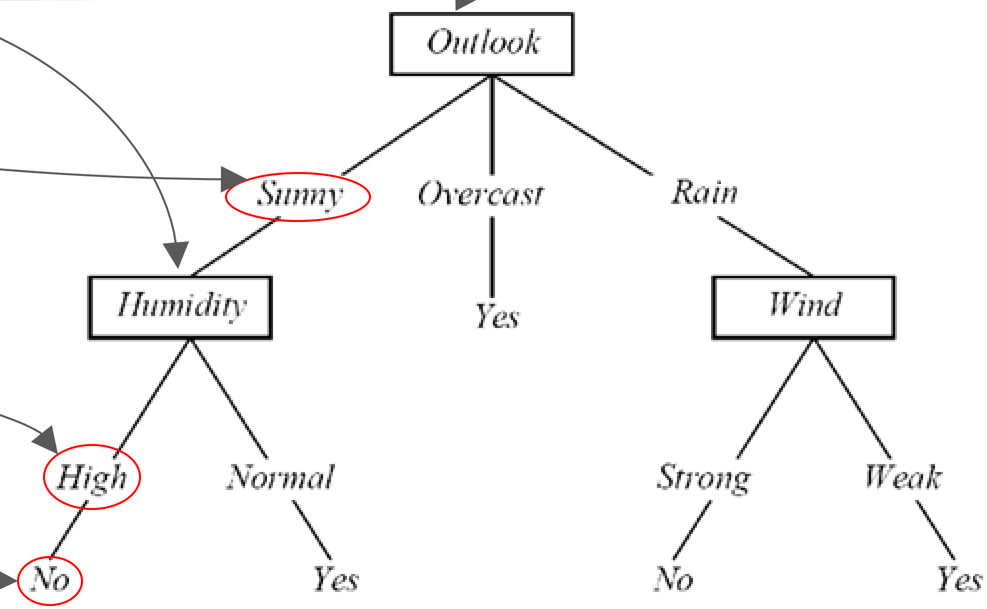
Facts or Observations

Theory

Interpreting a DT

DT ≡ Decision Tree

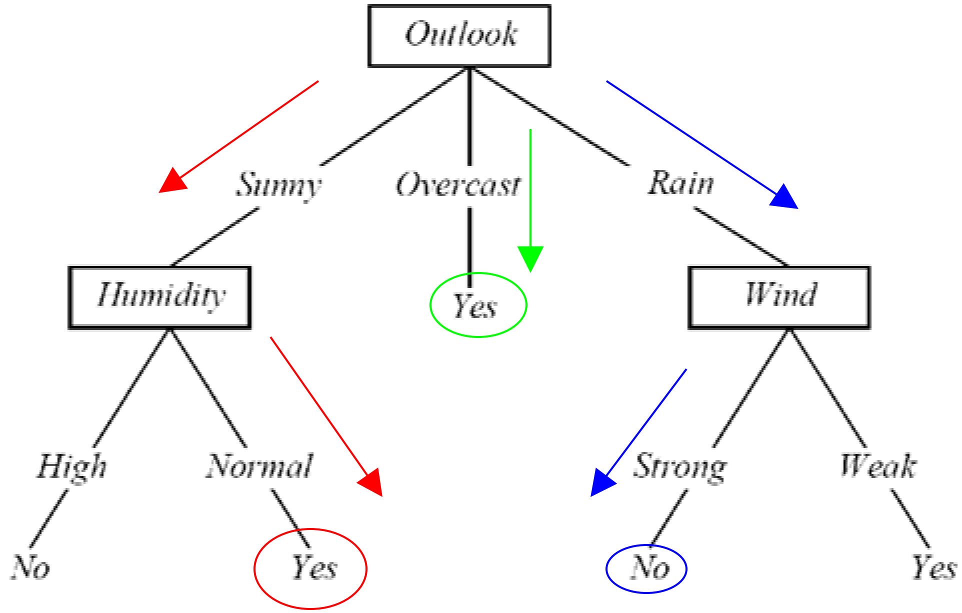
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No



- ➔ A DT uses the <u>features</u> of an observation table as nodes and the <u>feature values</u> as links.
- ➔ <u>All</u> feature values of a particular feature need to be represented as links.
- ➔ The target feature is special - its values show up as <u>leaf nodes</u> in the DT.

Interpreting a DT

Each path from the root of the DT to a leaf can be interpreted as a decision rule.



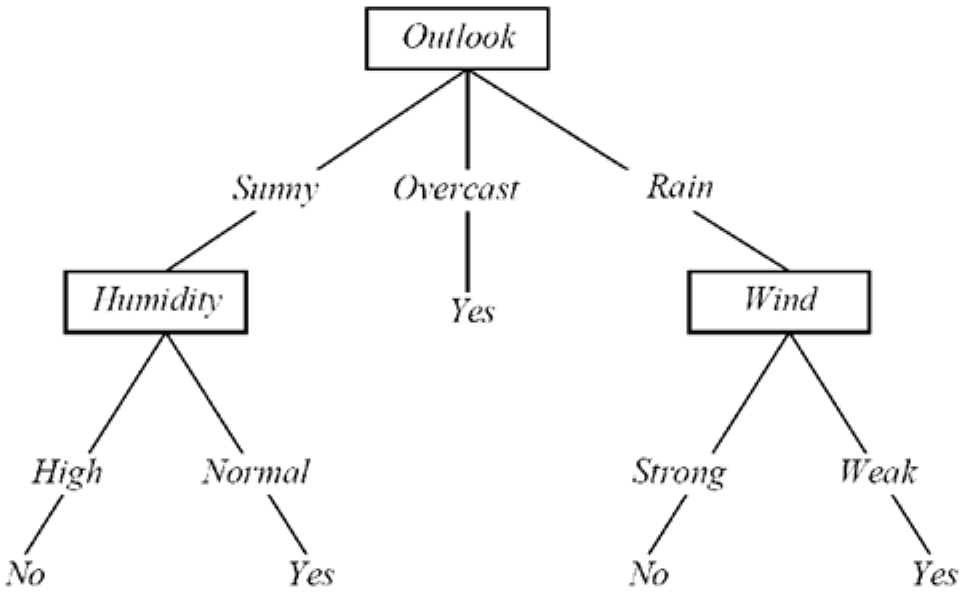
IF *Outlook = Sunny* AND *Humidity = Normal* THEN *Playtennis = Yes*

IF *Outlook = Overcast* THEN *Playtennis =Yes*

IF *Outlook = Rain* AND *Wind = Strong* THEN *Playtennis = No*

DT: Explanation & Prediction

Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No



Explanation: the DT summarizes (explains) all the observations in the table perfectly ⇒ 100% Accuracy

Prediction: once we have a DT (or model) we can use it to make predictions on observations that are not in the original training table, consider:

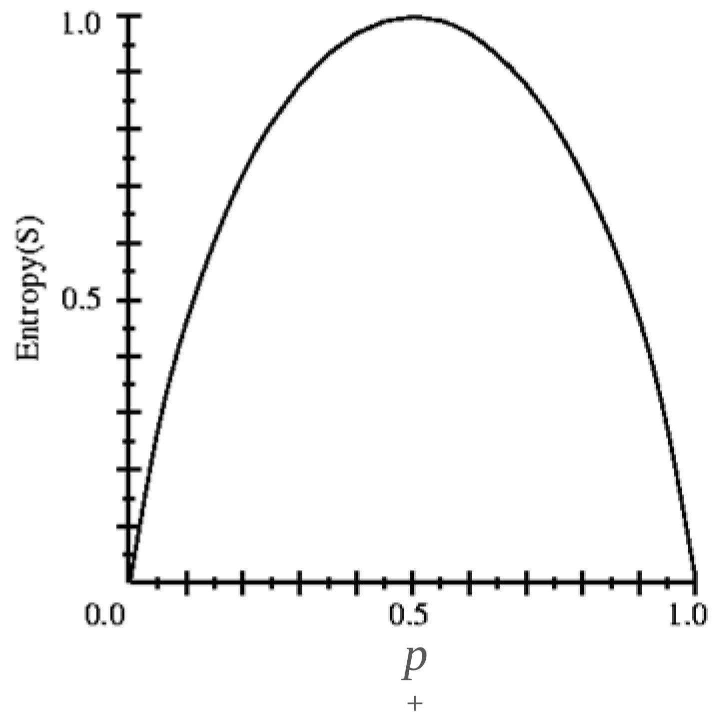
Outlook = Sunny, Temperature = Mild, Humidity = Normal, Windy = Weak, Playtennis = ?

Constructing DTs

- How do we choose the attributes and the order in which they appear in a DT?
  - Recursive partitioning of the original data table
  - Heuristic - each generated partition has to be “less random” (entropy reduction) than previously generated partitions

Entropy

- S is a sample of training examples
- p^+ is the proportion of positive examples in S
- p^- is the proportion of negative examples in S
- Entropy measures the impurity (randomness) of S



S

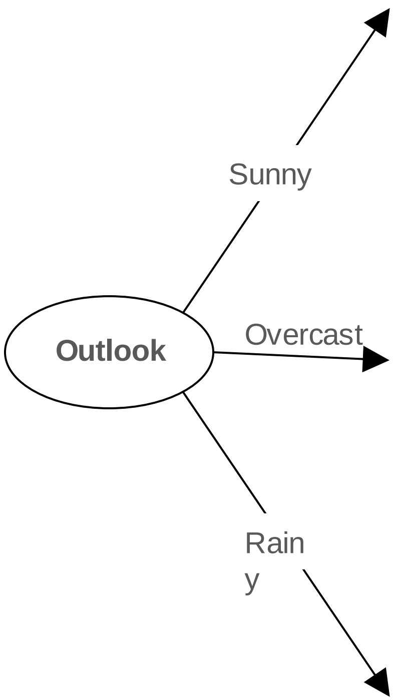
Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

$$\boxed{Entropy(S) \equiv - p^+ \log_2 p^+ - p^- \log_2 p^-}$$

$Entropy(S) = Entropy([9+,5-]) = .94$

$$\text{AvgEntropy}(S,A) = \sum_{v \in Values(A)} \frac{|S_v|}{|S|} E(S_v) \quad \text{(weighted average)}$$

Partitioning the Data Set



Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

E = .97

Outlook	Temperature	Humidity	Windy	PlayTennis
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

E = 0

Outlook	Temperature	Humidity	Windy	PlayTennis
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Rainy	Mild	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

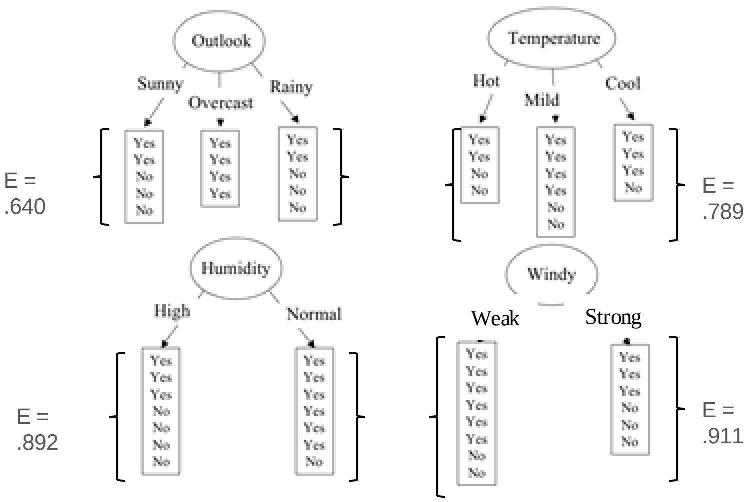
E = .97

Average Entropy = .64

(weighted .69)

Partitioning in Action

Outlook	Temperature	Humidity	Windy	PlayTennis
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No



The ID3 Algorithm

Function ID3 (S:Dataset) return T:Tree

1. Calculate the entropy of every variable in S
2. Partition ("split") the S into subsets using the variable for which the resulting entropy after splitting is minimized.
3. Make a decision tree node containing that variable.
4. Create a branch for each label in the variable.
5. Recurse on subsets using the *remaining variables*.
6. Return the resulting tree.

Recursive Partitioning

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Outlook

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Outlook	Temperature	Humidity	Windy	Play
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

Outlook	Temperature	Humidity	Windy	Play
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Rainy	Mild	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Recursive Partitioning

Outlook

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Outlook	Temperature	Humidity	Windy	Play
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

Outlook	Temperature	Humidity	Windy	Play
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Rainy	Mild	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Recursive Partitioning

Outlook

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Outlook	Temperature	Humidity	Windy	Play
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

Outlook	Temperature	Humidity	Windy	Play
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Rainy	Mild	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Humidity

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No

Outlook	Temperature	Humidity	Windy	Play
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Recursive Partitioning

Outlook

Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Outlook	Temperature	Humidity	Windy	Play
Overcast	Hot	High	Weak	Yes
Overcast	Cool	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes

Outlook	Temperature	Humidity	Windy	Play
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Cool	Normal	Strong	No
Rainy	Mild	Normal	Weak	Yes
Rainy	Mild	High	Strong	No

Humidity

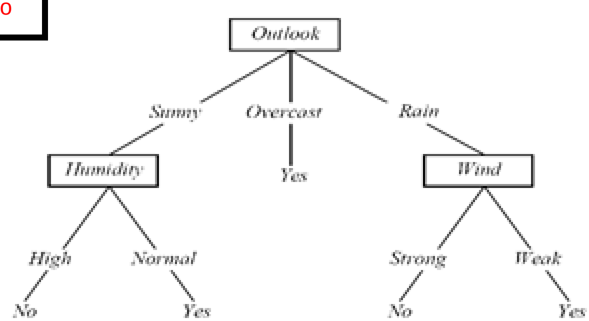
Outlook	Temperature	Humidity	Windy	Play
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Sunny	Mild	High	Weak	No

Outlook	Temperature	Humidity	Windy	Play
Sunny	Cool	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes

Windy

Outlook	Temperature	Humidity	Windy	Play
Rainy	Cool	Normal	Strong	No
Rainy	Mild	High	Strong	No

Outlook	Temperature	Humidity	Windy	Play
Rainy	Mild	High	Weak	Yes
Rainy	Cool	Normal	Weak	Yes
Rainy	Mild	Normal	Weak	Yes



Recursive Partioning

- Our hand-simulated algorithm created exactly the same tree that we have shown before for the tennis dataset.

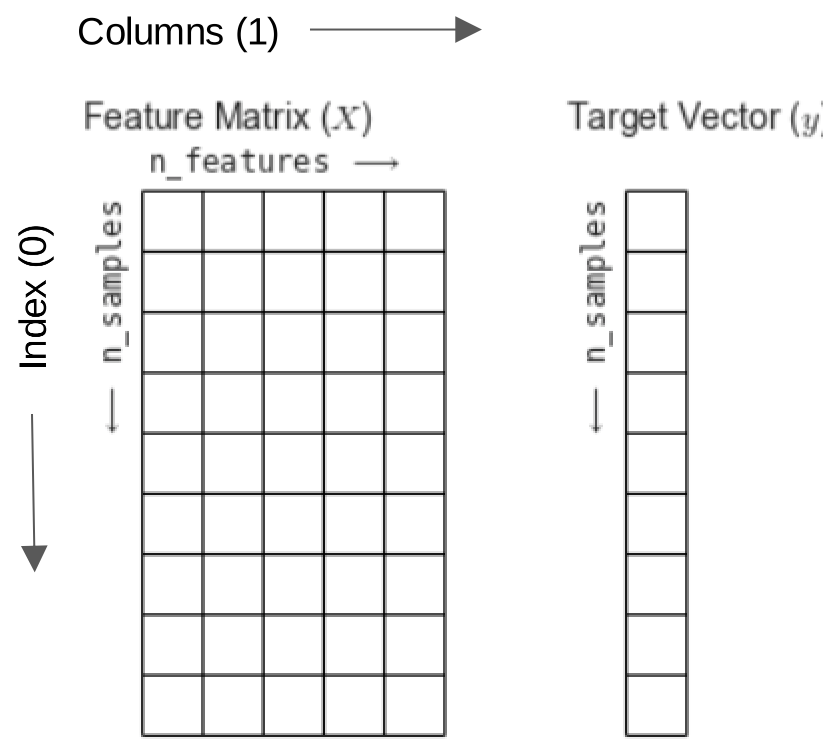
SKlearn Decision Tree Basics

Training data needs to be structured into a *feature matrix* and a *target vector*.

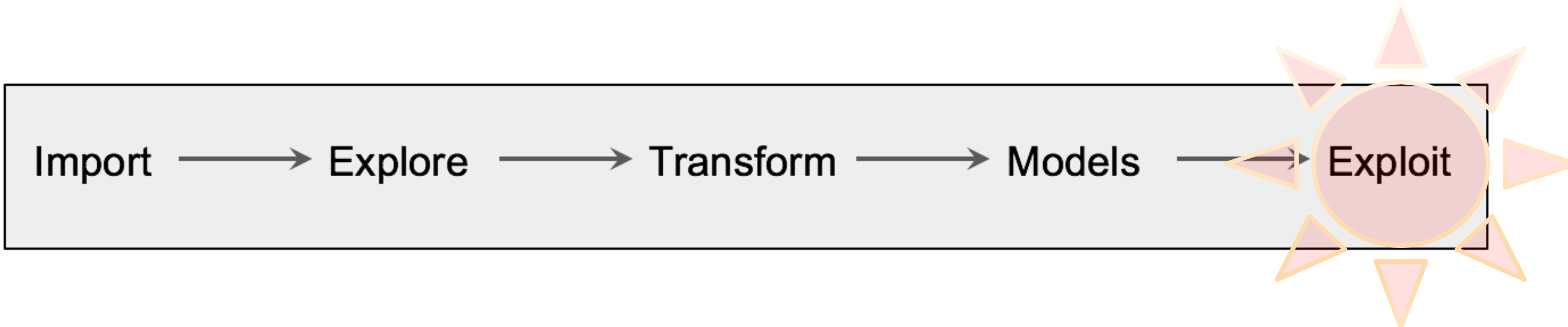
In the feature matrix one row for each observations.

In the target vector one entry for each observation.

NOTE: rows and vector entries have to be consistent!



The Pipeline



- The last step in our pipeline is “Exploit”, meaning taking advantage of the knowledge gained through the models.
- In this course we will not really delve into this because it is highly domain depended, e.g.
  - For the scientist, using the raw model to predict future research outcomes might be enough
  - For a banking application, like a loan scoring program, the model would have to be embedded in some sort of application that allows the bank employee to effectively take advantage of the model
  - For a network intrusion detection system, the model would have to be embedded in the network with appropriate access to the network and defensive systems

Models

- Let’s continue to look at machine learning basics such as tree model **building**, **evaluation**, and **visualization**