CARTE ML Workshop

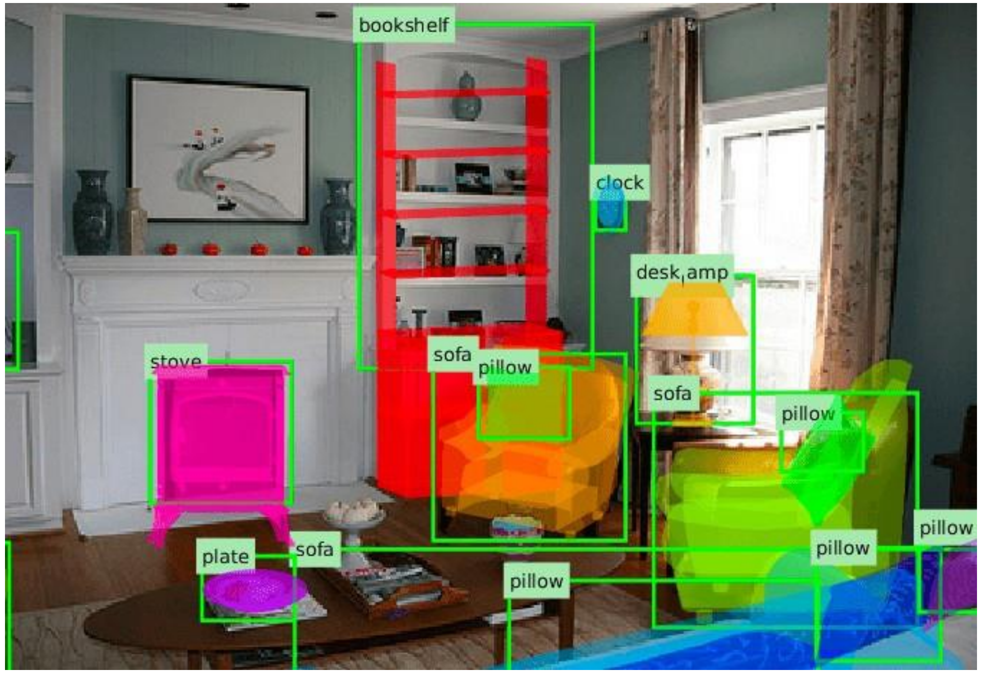
Lecture 1-1: Introduction to ML

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Centre for Analytics and Artificial Intelligence Engineering

Artificial Intelligence

- Getting computers to behave intelligently:
  - Perform non-trivial tasks as well as humans do
  - Perform tasks that even humans struggle with

- Many sub-goals:
  - Perception
  - Reasoning
  - Control
  - Planning



My poker face: AI wins multiplayer game for first time

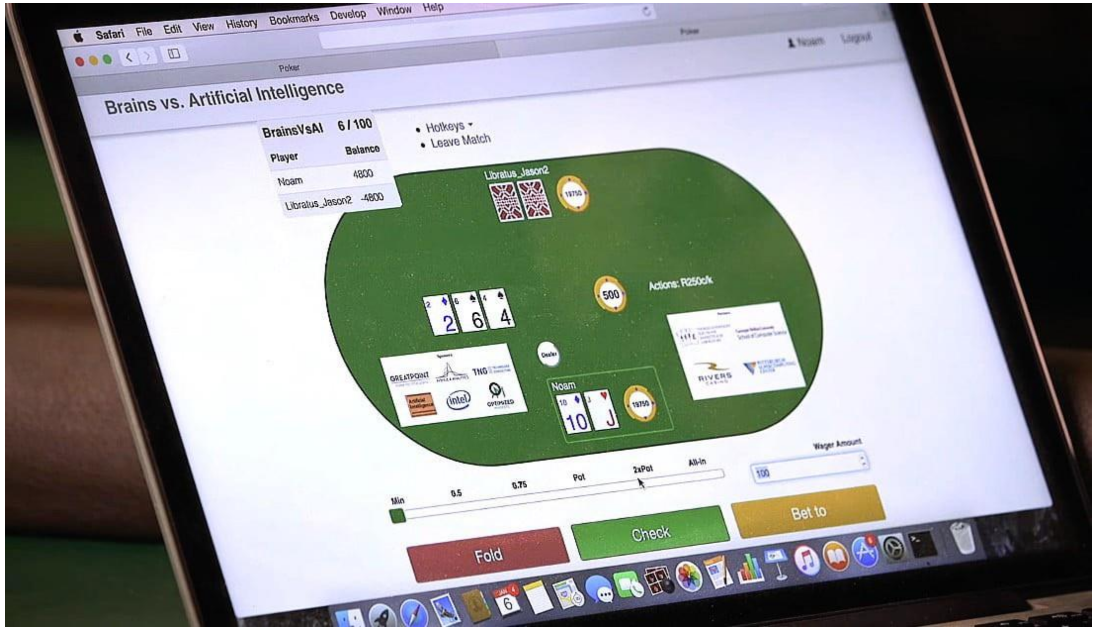
Pluribus wins 12-day session of Texas hold'em against some of the world's best human players

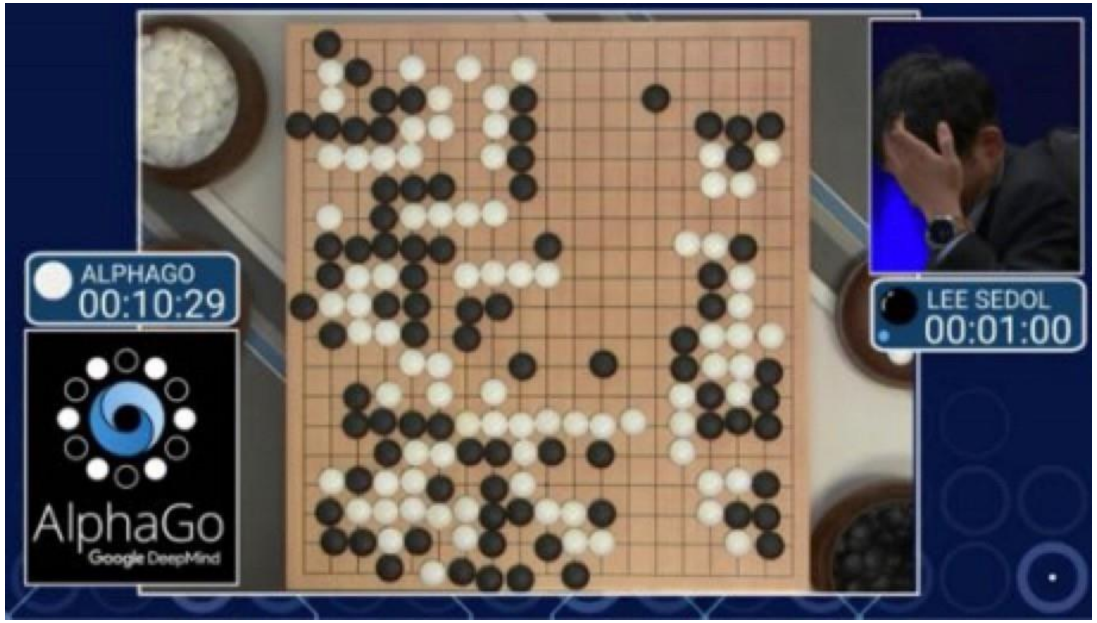
Speech Recognition: Perception + Reasoning

UNIVERSITY OF TORONTO
FACULTY OF APPLIED SCIENCE & ENGINEERING
Centre for Analytics and Artificial Intelligence Engineering

Autonomous Driving: Perception + Reasoning
Control + Planning

Game Playing: Reasoning + Planning



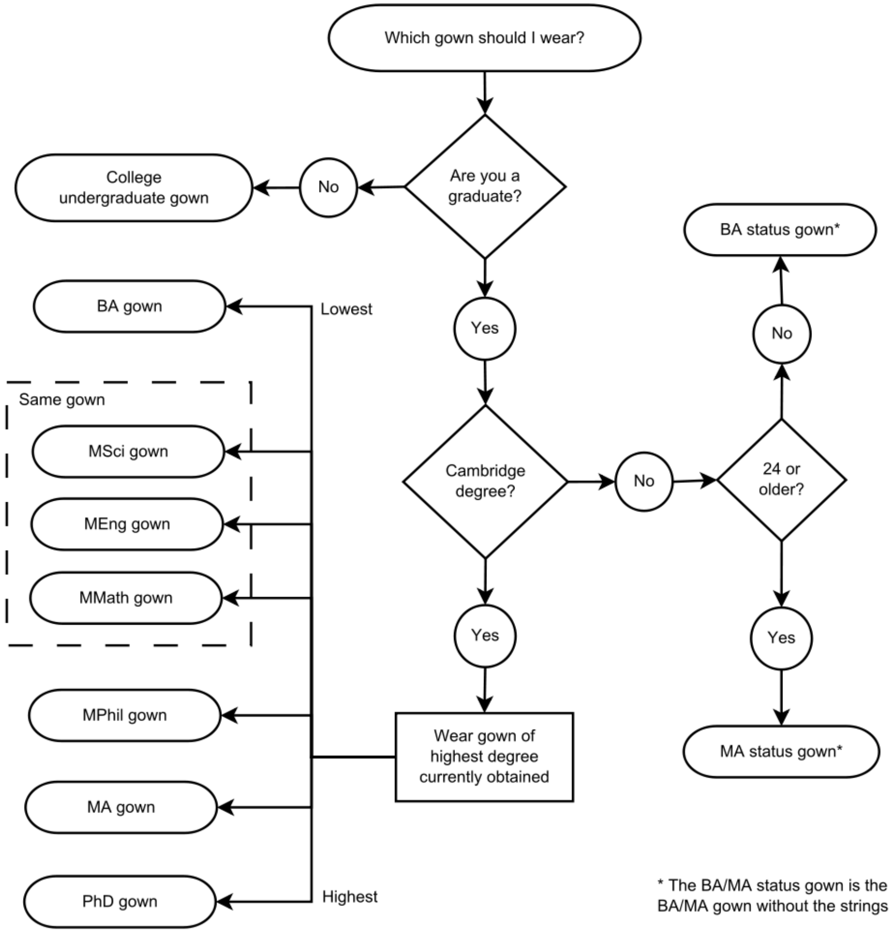


Knowledge-Based AI

Write programs that simulate how people solve the problem

Fundamental limitations:

- Will never get better than a person
- Requires deep domain knowledge
- We don't know how we do some things (e.g., riding a bicycle)



Data-Based AI = Machine Learning

Write programs that learn the task from examples

✅ No need to know how we do it as humans
✅ Performance should improve with more examples

❌ May need many examples!
❌ May not understand how the program works!

Machine Learning

- Study of algorithms that
  - Improve their performance P
  - At some task T
  - With experience E
- Well defined learning task: <P,T,E>

The Machine Learning Process

- Study of algorithms that
  - Improve their <u>performance</u> P
  - At some <u>task</u> T
  - With <u>experience</u> E
- Well defined learning task: <P,T,E>

- Experience
  - Examples of the form

    (input, correct output)
- Task
  - Mapping from input to output
- Performance
  - ”Loss function” that measures error w.r.t. desired outcome

Choices in ML Problem Formulation

- Experience
  - Examples of the form (input, correct output)
- Task
  - Mapping from input to output
- Performance
  - ”Loss function” that measures error w.r.t. desired outcome

Loan Applications

- What historical examples do I have? What is a correct output?
- Predict probability of default? Loan decision? Credit score?
- Do I care more about minimizing False Positives? False negatives?

How will I rate “Chopin’s 5th Symphony”?

Song	Rating
Some nights	⭐⭐⭐⭐⭐
Skyfall	⭐
Comfortably numb	⭐⭐⭐
We are young	⭐⭐⭐⭐
…	…
…	…
Chopin’s 5th	???

Classification: Three Elements

1. Data:
   - x: data example with d attributes
   - y: label of example (what you care about)

2. Classification model: a function $f_{(a,b,c,..}$
   - Maps from X to Y
   - (a,b,c,…) are the parameters

3. Loss function:
   - Penalizes the model's mistakes

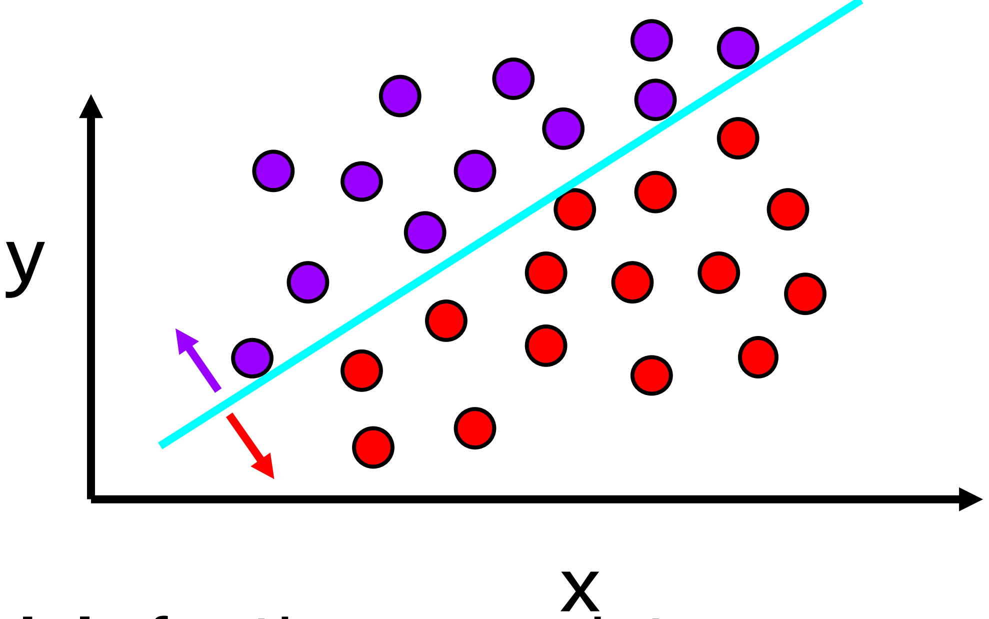
Song	Rating
Some nights	⭐⭐⭐⭐⭐
Skyfall	⭐
Comfortably numb	⭐⭐⭐
We are young	⭐⭐⭐⭐
…	…
…	…
Chopin's 5th	???

Terminology Explanation

Song	Artist	Length	…	Rating
Some nights	Fun	4:23	…	⭐⭐⭐⭐⭐
Skyfall	Adele	4:00	…	⭐
Comf. Numb	Pink Floyd	6:13	…	⭐⭐⭐
We are young	Fun	3:50	…	⭐⭐⭐⭐
…	…	…	…	…
…	…	…	…	…
Chopin's 5^{th}	Chopin	5:32	…	???

Data example = data instance
Attribute = feature = dimension
Label = target attribute

What is a "model"?



A **useful approximation** of the world

Typically, there are **many reasonable models** for the same data

Training a model = finding appropriate values for (a,b,c,…)

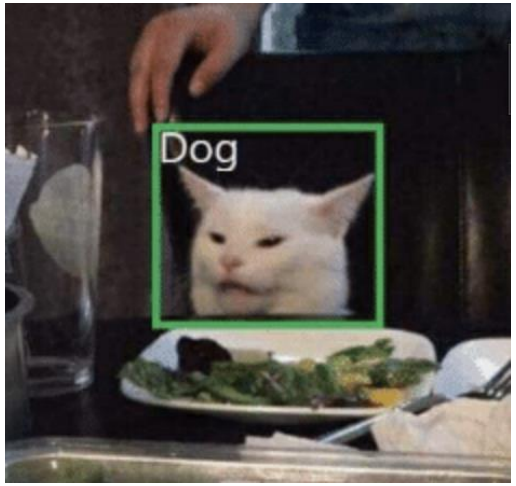
- An **optimization** problem
- "appropriate" = **minimizes the Loss (cost)** function
- We will focus on a common training algorithm later on

Classification Loss Function

- How unhappy are you with the answer that the model gave?

- $\mathbf{L_{0\text{-}1}(y, f(x))} = 1 \qquad$ if: $\mathbf{y \neq f(x)}$

  $\qquad\qquad\qquad\quad 0 \qquad$ otherwise



- **0-1 loss** function: intuitive but hard to optimize = train

- In practice, we use **approximations** of the 0-1 loss – getting warmer or getting colder

Why should this work at all?

The main theoretical basis of ML:

With a **sufficient amount of "similar" data**

+

an **expressive model class**:

Minimizing the loss function on the training data yields a highly accurate model on **unseen test data**, with high probability

1. Data: $S = \{(x_i, y_i)\}_{i = 1,...,n}$
   - x_i: data example with d attributes
   - y_i: label of example (what you care about)

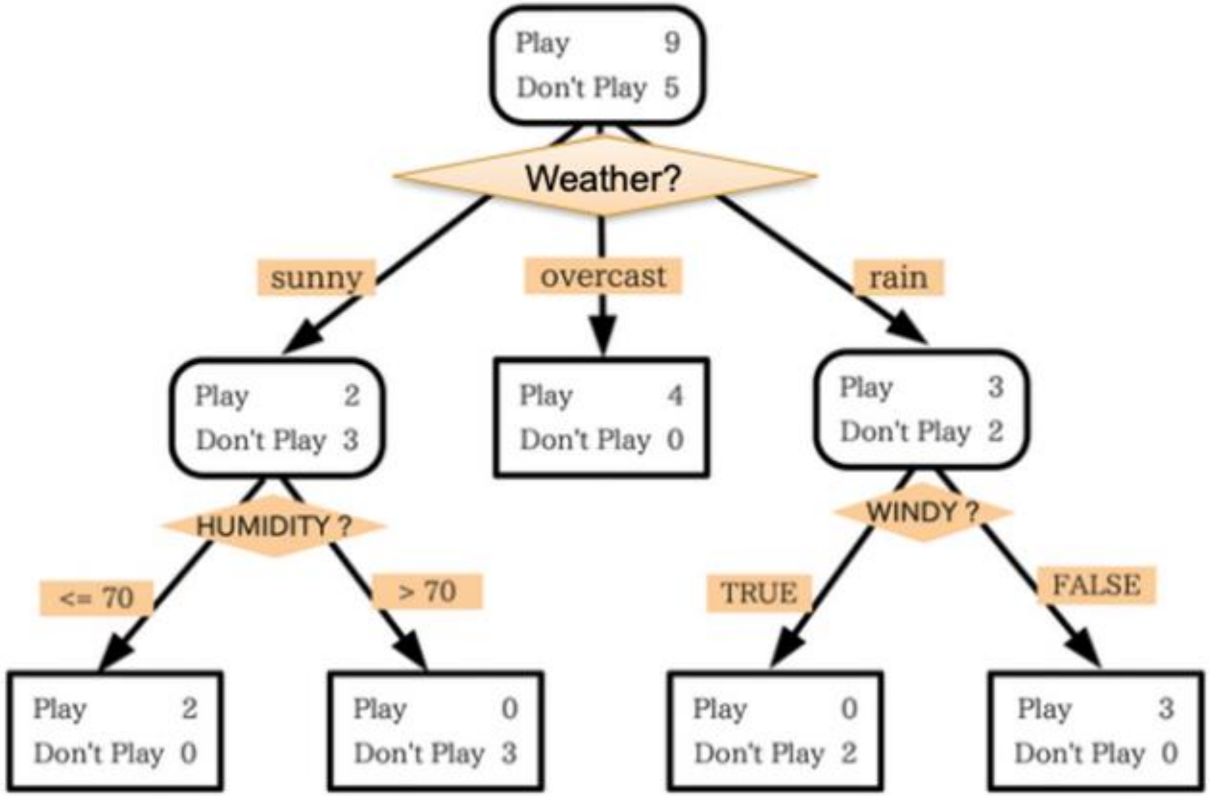
2. Classification model: a function $f_{(a,b,c,...)}$
   - Maps from X to Y
   - (a,b,c,...) are the parameters

3. Loss function: $L(y, f(x))$
   - Penalizes the model's mistakes

Decision Trees: To play tennis or not to?

- **Data**: attributes describing the weather; (sunny? humidity level, …)
- **Target**: 1if it's good to **Play**, 0 otherwise

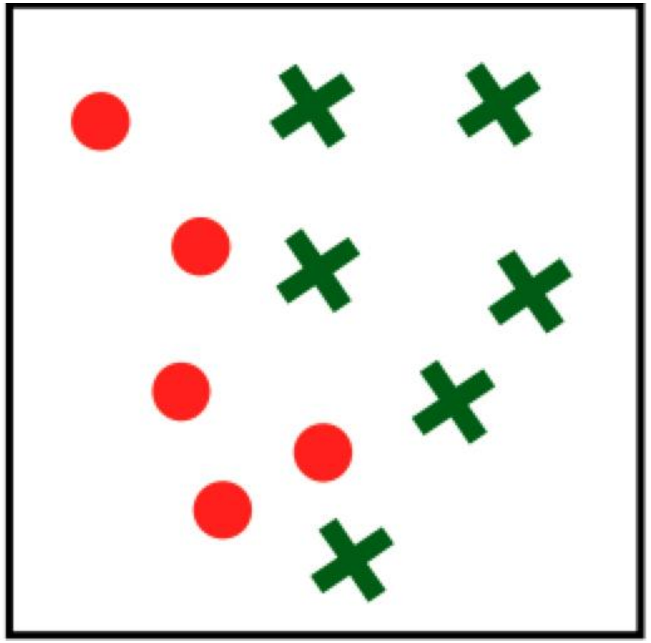
- **Model:** $f_T(x)$
- **Model parameters**: T, the tree structure (and size)



Training (fitting) a Decision Tree

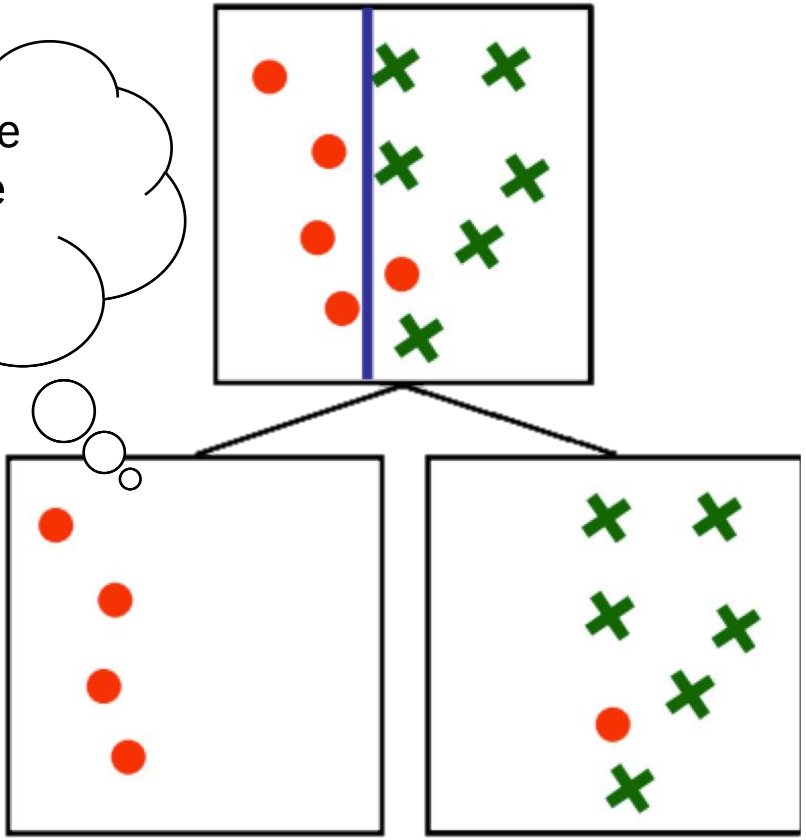
How to choose the attribute/value to split on at each level of the tree?

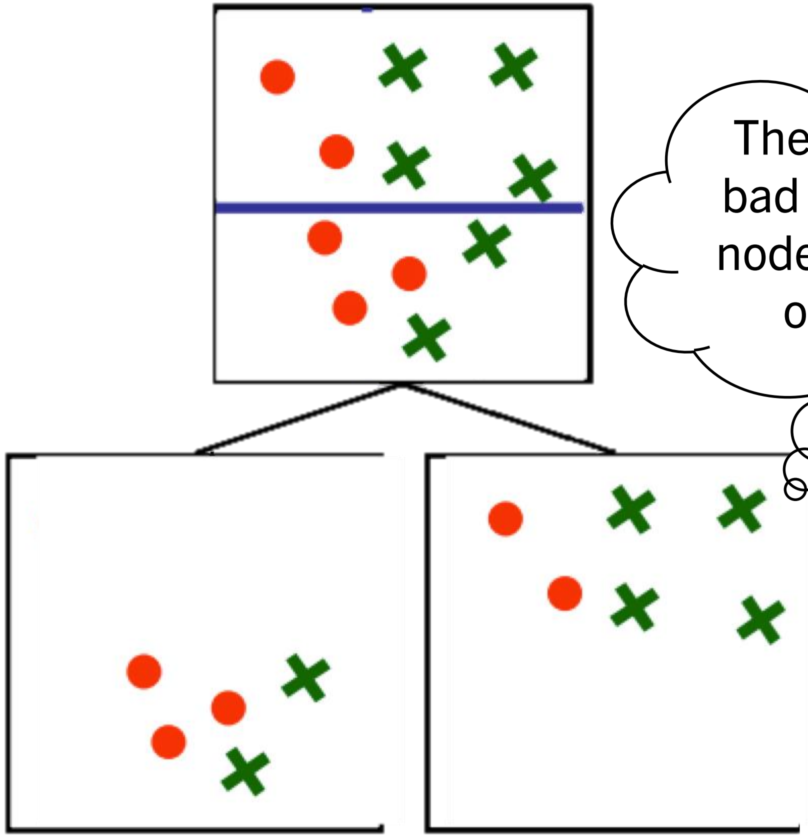
- Two classes (red circles / green crosses)
- Two attributes: X and Y
- 11 points in training data
- Idea: construct a decision tree such that the leaf nodes correctly predict the class for all the training examples



Training (fitting) a Decision Tree

These splits are great because the nodes are "pure"

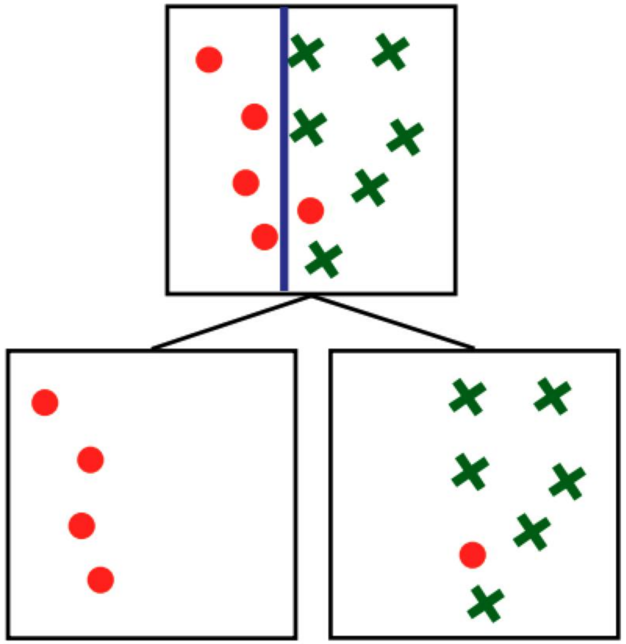




These splits are bad because the nodes have a mix of samples

Training (fitting) a Decision Tree

1. Find the best attribute to split on
2. Find the **best split** on the chosen attribute
3. Repeat 1 & 2 until **stopping criterion** is met



Common **stopping criteria**:

- Node contains very few data points
- Node is pure: most training data in node have same label

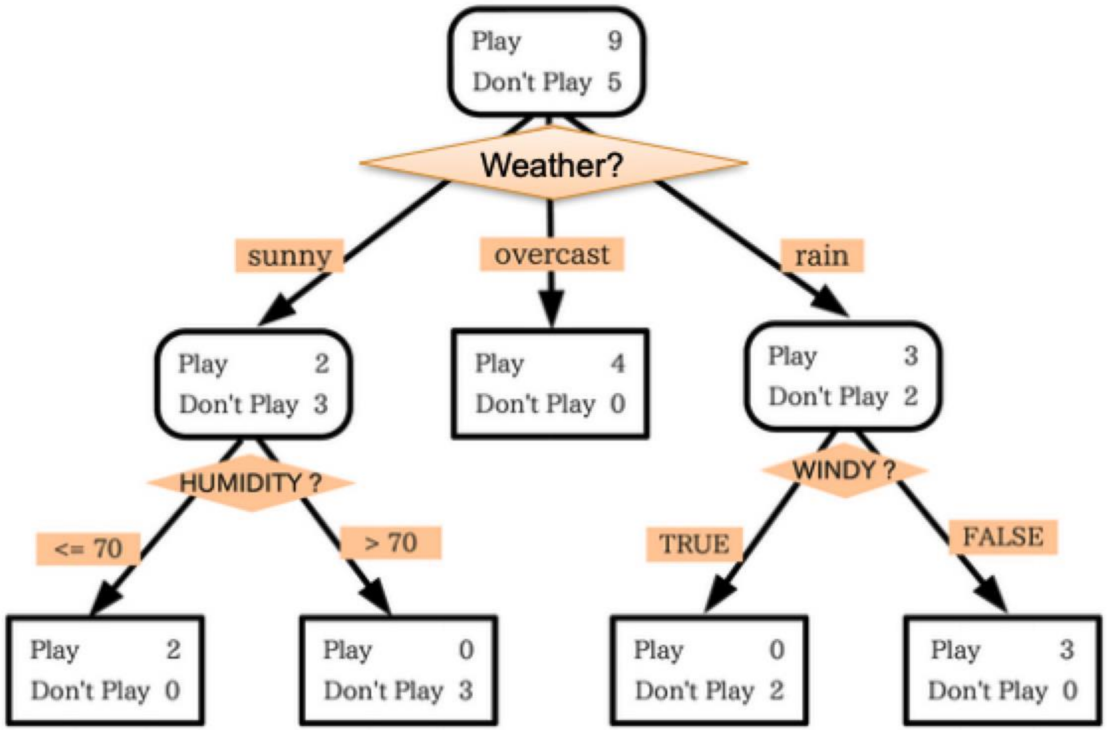
Final words on Decision Trees

Advantages

- Simple interpretation
- Fast predictions
- Handles mixed-type attributes

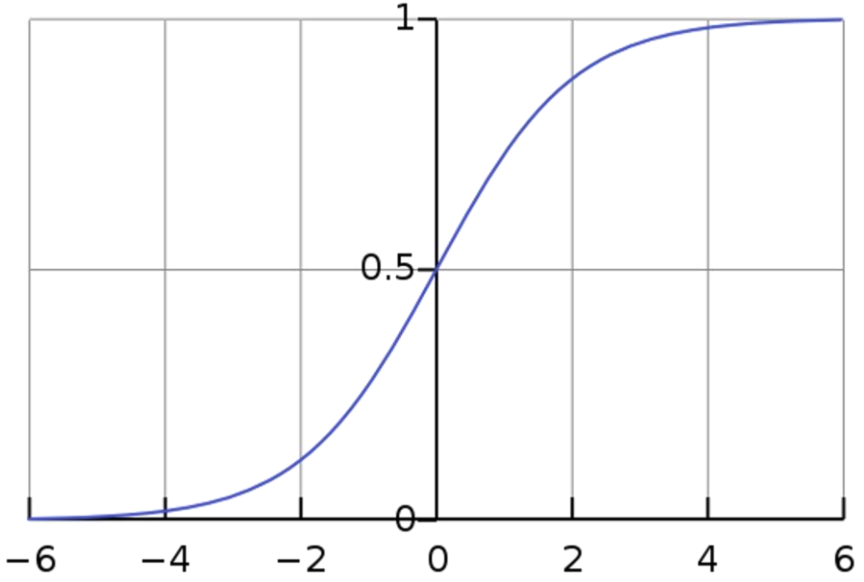
Caveats

- May be too simple for complex data
- Hard to figure out the right depth, stopping criterion, especially at the node level



Logistic Regression (LR)

$\sigma(x) = 1/(1 + e^{-x})$



Decision Trees predict discrete outcomes

LR predicts probabilities of outcomes

- Probabilities give a notion of certainty
- Model can still be used as a classifier

Probability of getting cervical cancer, p(x):

p(age=42, #pregnancies=3, smoking=True, …)

Logistic Regression: Assumptions

Probability of getting cervical cancer, $p(x)$:

$$p(age = 42, pregnancancies = 3, smoking = True \ldots)$$

LR Parameters: $\beta_0, \beta_1, \ldots, \beta_d$

$$\log \frac{p(x)}{1 - p(x)} = \beta_0 + \beta_1 x_1 + \ \ldots + \beta_d x_d$$

This is the model!

$$\Rightarrow \boxed{p(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \ldots + \beta_d x_d)}}}$$

Logistic Regression: Training

Data: $S = \{(x_i, y_i)\}$

x_i: example with d attributes (age, #pregnancies, …)
y_i: cervical cancer diagnosis (0 or 1)

Maximum Likelihood Estimation (MLE)

Likelihood of observing the data for a given $\boldsymbol{\beta}$

MLE seeks parameters $\boldsymbol{\beta}$ that maximize the likelihood

The optimal parameters, $\boldsymbol{\beta}^*$, can be found by optimization

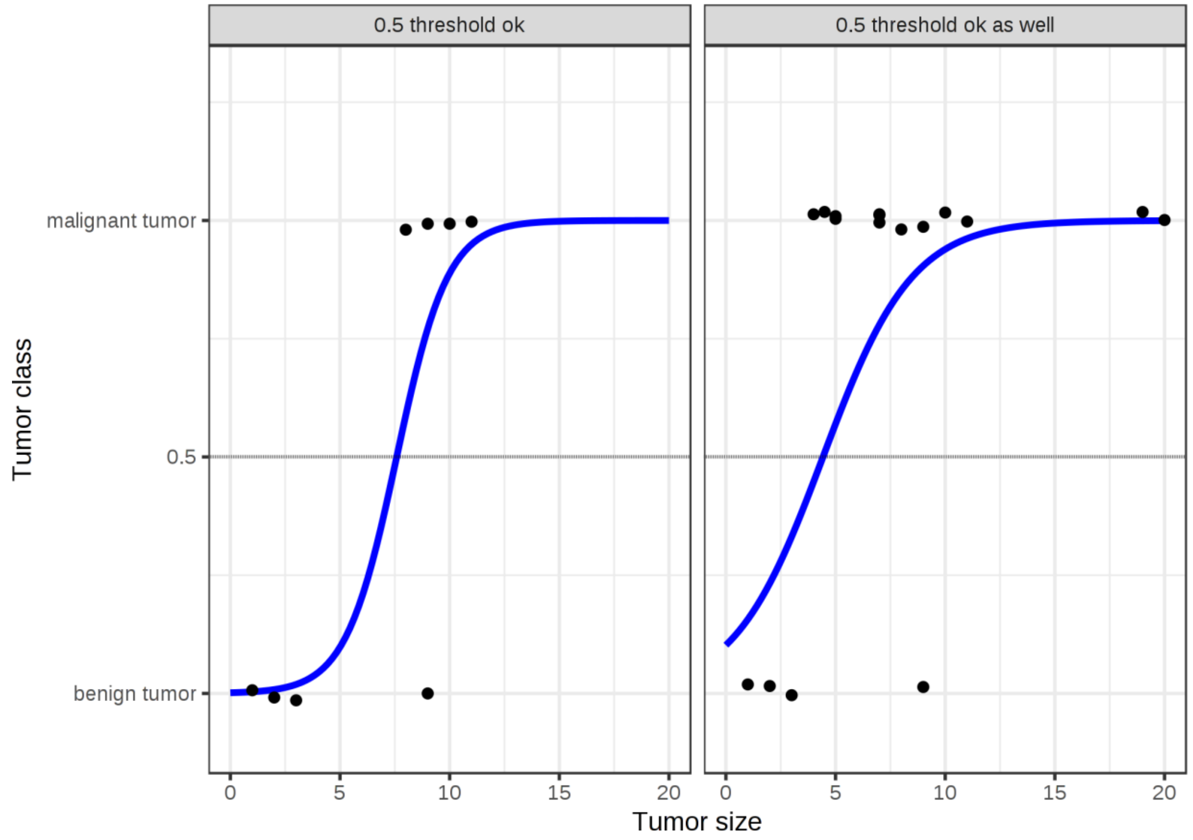
Final words on Logistic Regression

Advantages

- Simple interpretation
- Fast training (convex optimization)
- Fast predictions
- Handles mixed-type attributes

Caveats

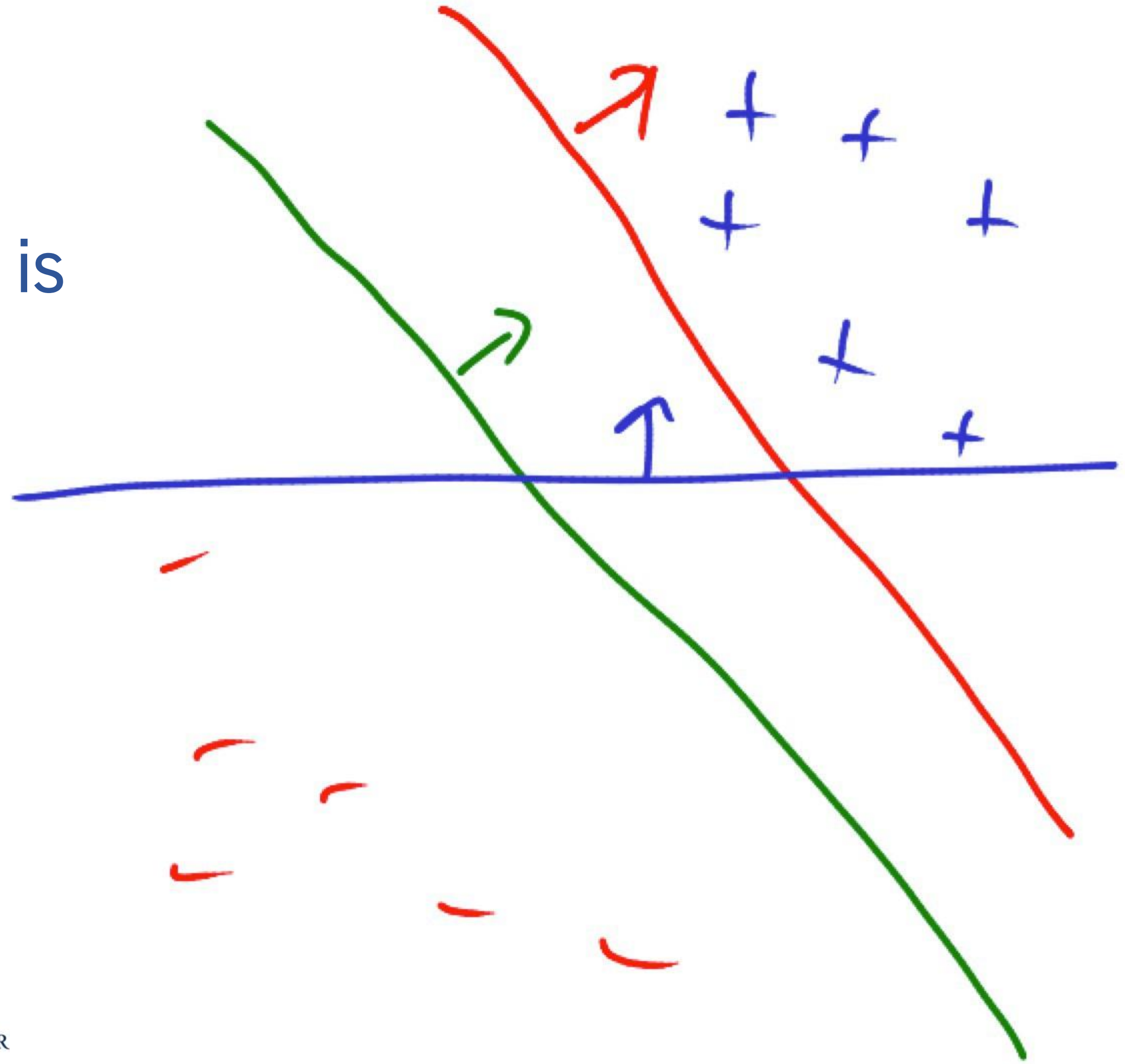
- A low-capacity, linear model



https://christophm.github.io/interpretable-ml-book/logistic.html

Support Vector Machines (SVM)

Which classifier is the best?

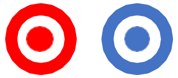


A Course in Machine Learning by Hal Daumé III

SVM: The Maximum-Margin Principle

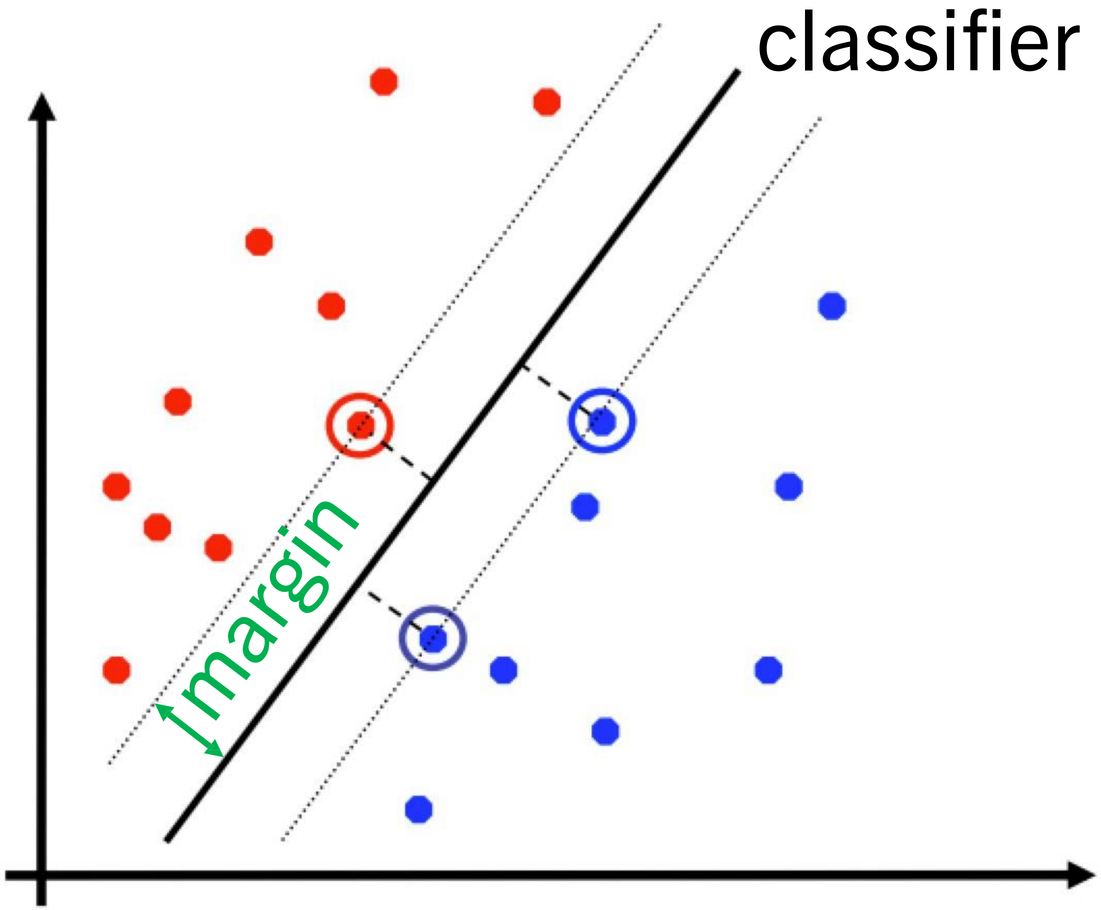
Vapnik (1990) derived the SVM as an “optimal” classifier

- Intuitively, robust to outliers



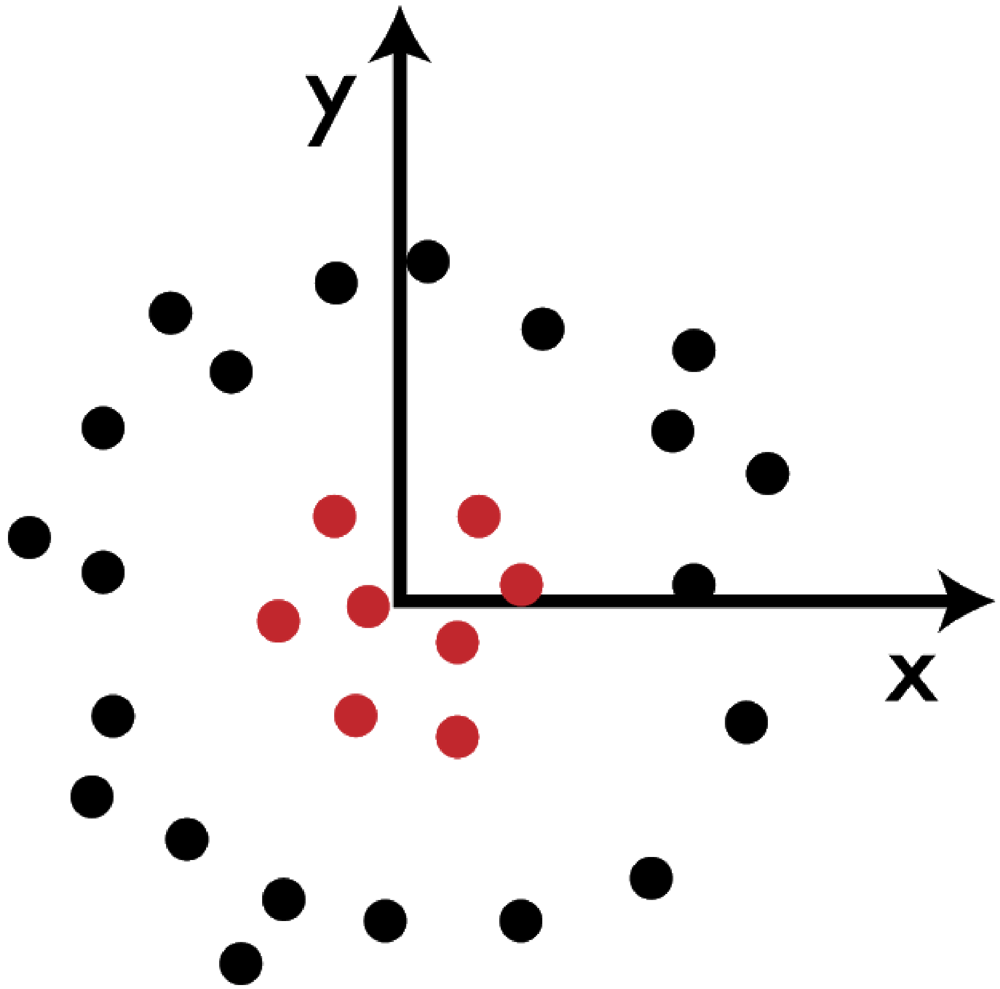
- Support vectors: subset of data closest to classifier

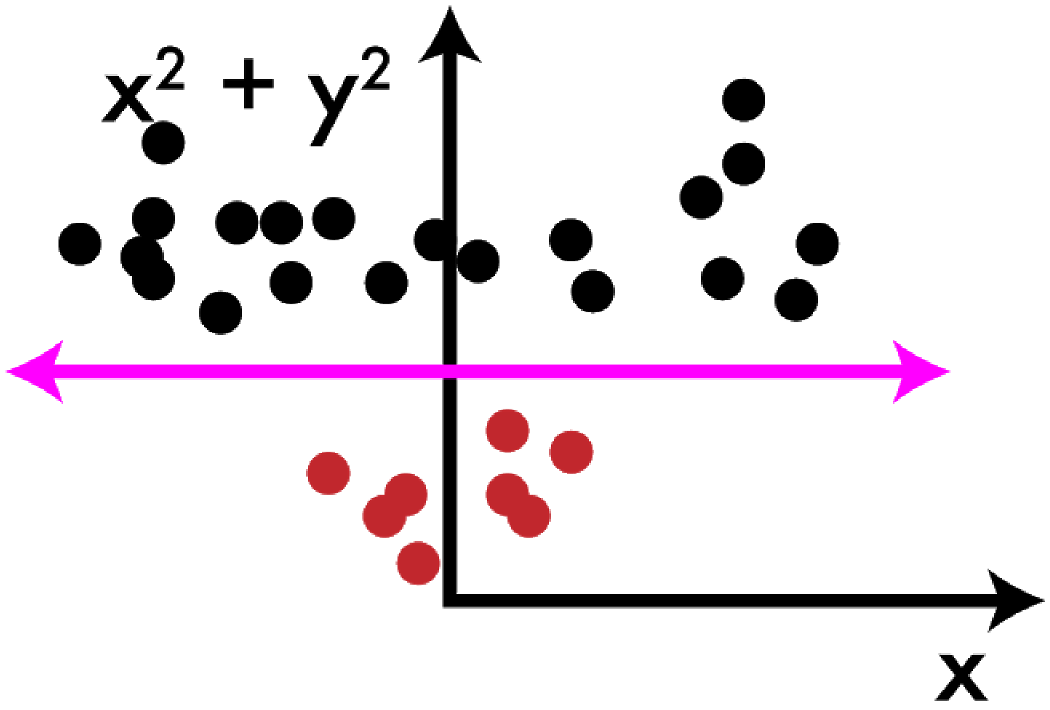
- Great empirical success in the 90s – early 2000s



David Sontag, NYU ML class

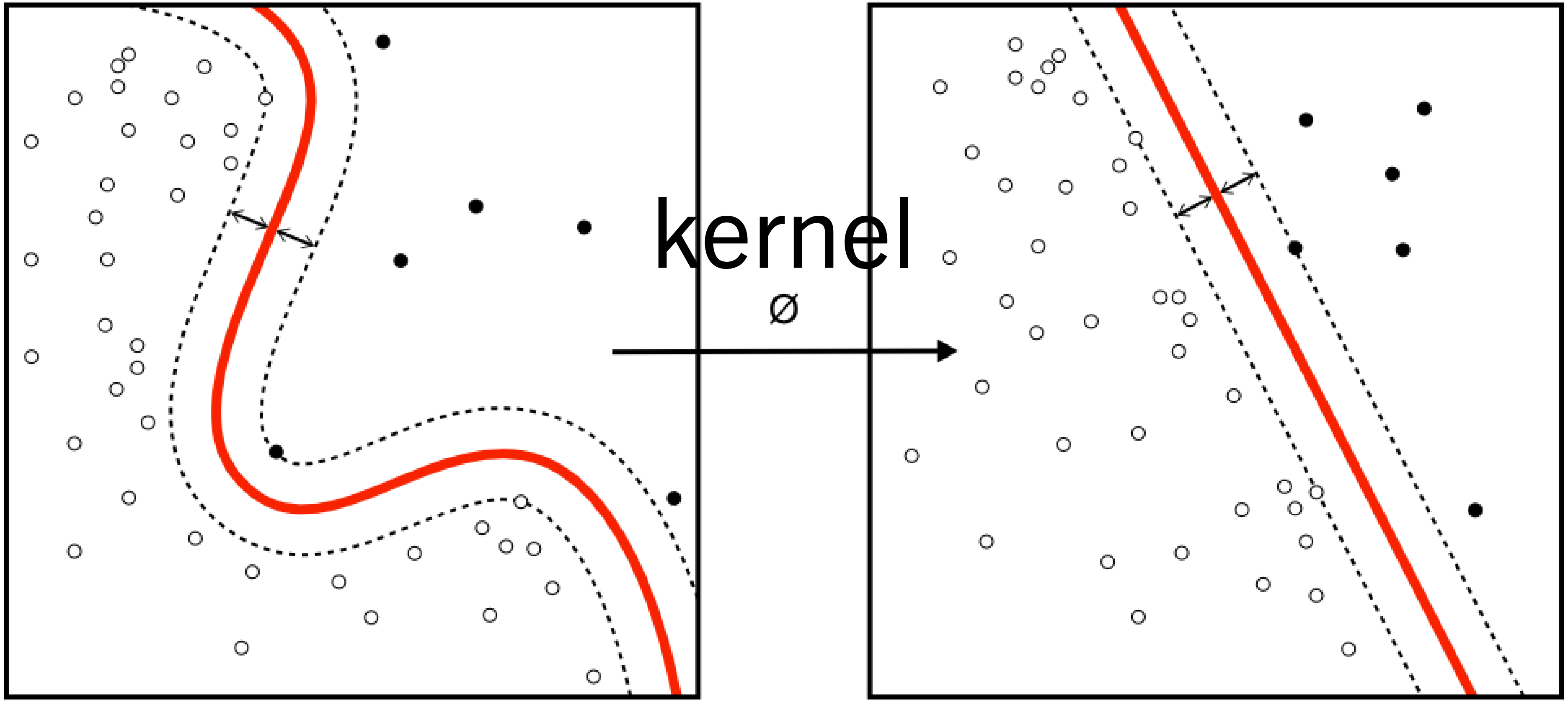
What about non-linearly separable data?





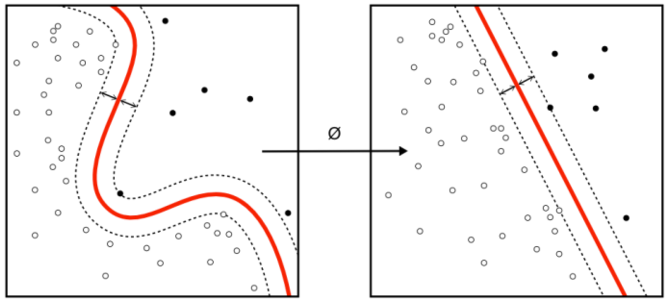
SVM for non-linearly separable data

SVM can do this "lifting" at a relatively small additional cost in computation



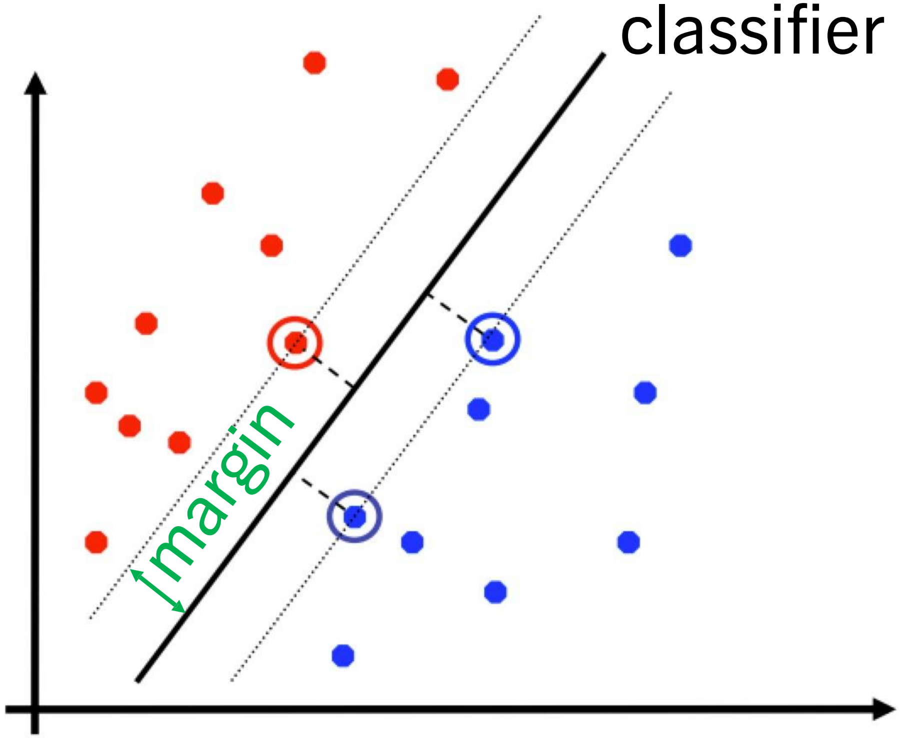
Wikipedia SVM article

Final words on SVMs



Advantages

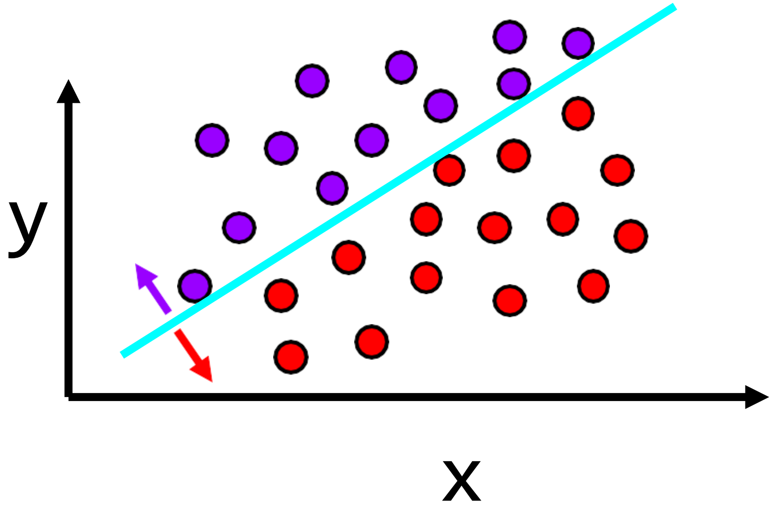
- Strong theoretical basis
- Easy to train linear SVMs
- Typically a strong baseline



Caveats

- Non-linear SVM slow to train
- Hard to specify a good kernel in advance

Recap



- ML vs Knowledge-Based AI
- The ML mindset
- Classification: definition and assumptions
- Classifiers:
    - Decision Trees
    - Logisitic Regression
    - Support Vector Machines