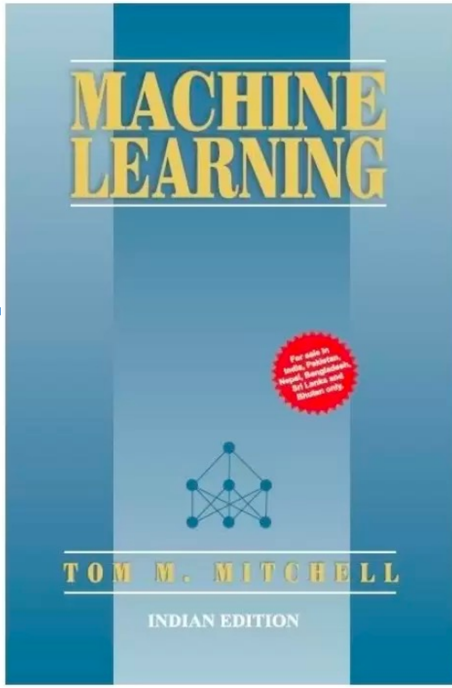
Course material at:
https://github.com/lyeskhalil/mlbootcamp2022



Introduction to Machine Learning

Monday, Lecture 1

DSI-CARTE ML Bootcamp

Based on material from Polo Chau, Tom Mitchell, Roni Rosenfeld, Martial Hebert, Hal Daumé III, David Sontag

A bit about myself…

ekhalil.com

- Joined **UofT MIE** starting July 2020.
- IVADO Postdoc at Polytechnique Montreal
  Canada Excellence Chair in Data Science for Real-Time Decision-Making
- PhD in Computational Science & Engineering, 2019
- Research Interests
  - Machine Learning for Discrete Optimization and Operations Research
  - Principled Optimization Methods for ML
  - Healthcare, City planning, Supply chain applicartions

Teaching Assistants

Rahul Patel

Alex Olson

- 2^{nd} year Industrial Engineering PhD student
- Research in Machine Learning for Mathematical Optimization

- CARTE Research Associate
- UofT MASc MIE graduate
- Broad experience in machine learning, consulting with faculty on applied data science research

Teaching Assistants

Jacob Mosseri

- 2nd year Industrial Engineering MASc student
- Research in Machine Learning and Operations Research for Orthopaedic Surgery Scheduling

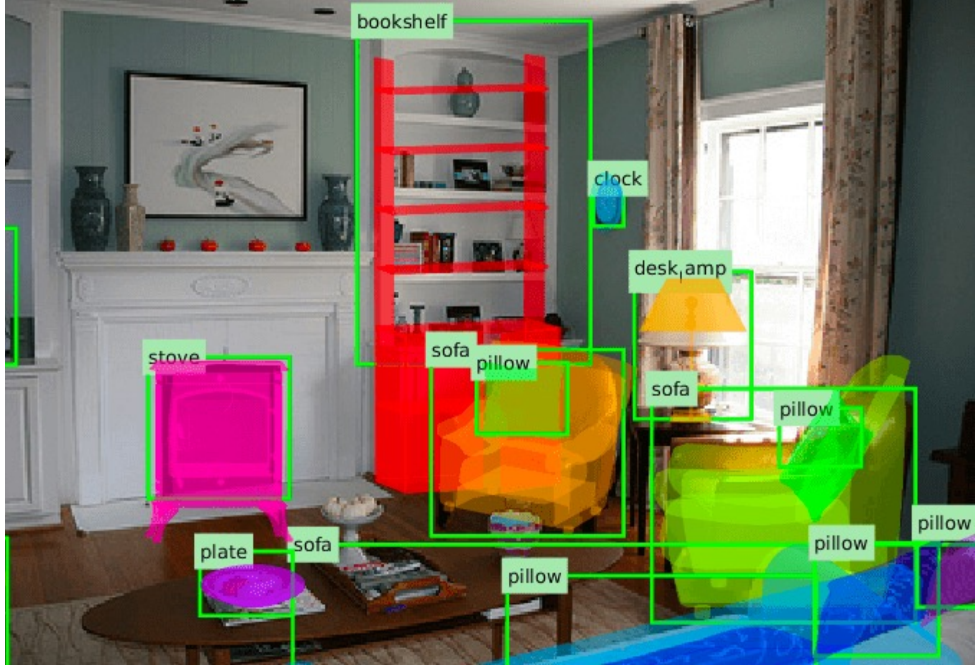
Artificial Intelligence

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**

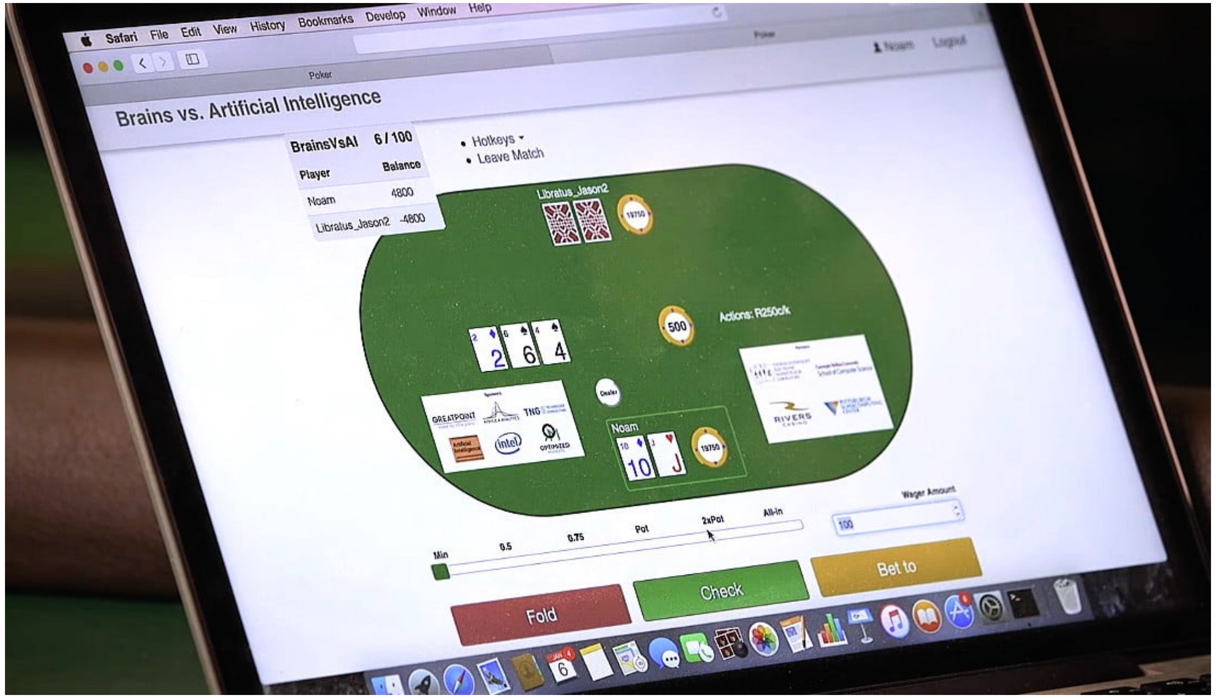
https://medium.com/@joelgarciajr84/creating-an-application-that-uses-speech-recognition-76117a396b7d

Autonomous Driving

Perception + Reasoning
Control + Planning

https://www.pymnts.com/news/partnerships-acquisitions/2019/ups-invests-in-autonomous-driving-firm-tusimple/

Game Playing Reasoning + Planning



https://www.digitaltrends.com/computing/texas-holdem-libratus-ai-defeats-humans/

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)

Knowledge representation via logic

Knowledge base : Set of formulae $\{f_1, f_2, \ldots, f_n\}$
M(KB) = All possible models for $f_1 \wedge f_2 \wedge \ldots \wedge f_n$

Formulae = "known facts"
Models = all possible "worlds" where all these facts hold
(Adding more facts to KB can only shrink set of possible worlds.)

Example: Variables: R, S, C ("Rainy", "Sunny," "Cloudy")

KB: $R \vee S \vee C$; ("It is either Rainy or Sunny or Cloudy.")
$R \rightarrow C \wedge \neg S$; ("If it is Rainy then it is Cloudy and not Sunny.")
$C \leftrightarrow \neg S$ ("If it is Cloudy then it is not Sunny, and vice versa")

Based on slide by Arora and Hazan at Princeton

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

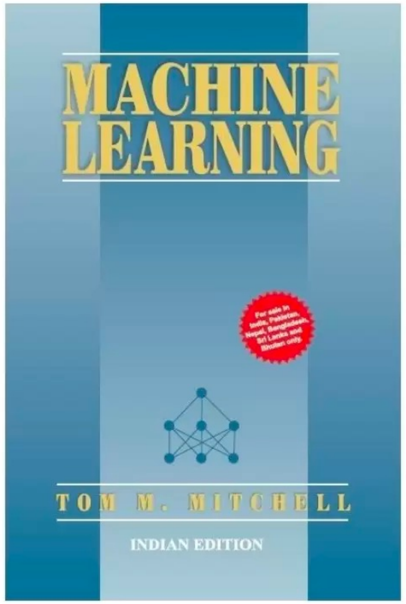
❌ May need **many examples**!
❌ May not understand how the program works!

Machine Learning:

Study of algorithms that

- improve their performance P
- at some task T
- with experience E

well-defined learning task: <P,T,E>



Tom Mitchell, CMU 10-601 slides

The Machine Learning Process

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

Machine Learning:

Study of algorithms that

- improve their performance P
- at some task T
- with experience E

well-defined learning task: <P,T,E>

Tom Mitchell, CMU 10-601 slides

Choices in ML Problem Formulation

Experience
- Examples of the form (input, correct output)

Loan Applications

- What historical examples do I have? What is a correct output?

Task
- Mapping from input to output

- Predict probability of default? Loan decision? Credit score?

Performance
- "Loss function" that measures error w.r.t. desired outcome

- Do I care more about minimizing False Positives? False negatives?

This course

The main algorithms (models)
Focus on Deep Neural Networks

Evaluation
Optimization

Machine Learning:

Study of algorithms that

- improve their performance P
- at some task T
- with experience E

well-defined learning task: <P,T,E>

Classification
Regression
Clustering

Tabular
Image
Sequence

How will I rate "Chopin's 5th Symphony"?

Songs	**Like?**
Some nights	
Skyfall	
Comfortably numb	
We are young	
...	...
...	...
Chopin's 5th	???

Classification: Three Elements

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

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

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

How will I rate "Chopin's 5th Symphony"?

Songs	Like?
Some nights	
Skyfall	
Comfortably numb	
We are young	
...	...
...	...
Chopin's 5th	???

Polo Chau, Georgia Tech CSE 6242

Terminology Explanation

data example = data instance
attribute = feature = dimension
label = target attribute

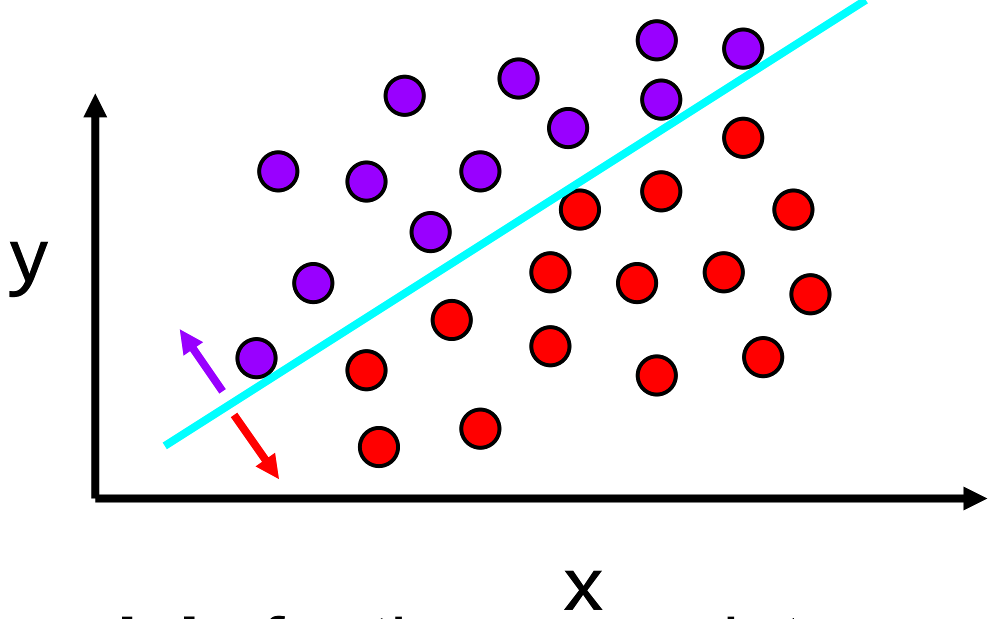
Data $S = \{(x_i, y_i)\}_{i = 1,\ldots,n}$

- x_i : data example with d attributes $x_i = (x_{i1}, \ldots, x_{id})$
- y_i : label of example

Song name	Artist	Length	...	Like?
Some nights	Fun	4:23	...	
Skyfall	Adele	4:00	...	
Comf. numb	Pink Fl.	6:13	...	
We are young	Fun	3:50	...	
...	...	...	...	...
...	...	...	...	...
Chopin's 5th	Chopin	5:32	...	??

Polo Chau
Georgia Tech CSE 6242

What is a “model”?



A **useful approximation** of the world

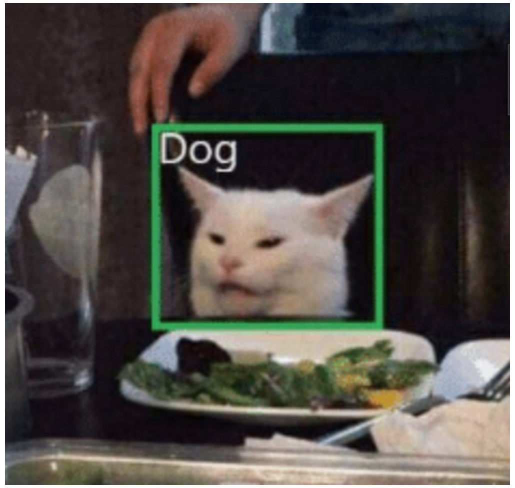
Typically, there are **many reasonable models** for the same data

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

- An **optimization** problem
- “appropriate” = **minimizes the Loss** function
- We will focus on a common training algorithm later on

Classification Loss Function

- How unhappy you would be if your model **f(.)** predict label **y'** on input **x** when **y** is the correct output



- $\mathbf{L_{0\text{-}1}(y, f(x))} = \begin{cases} 1 & \text{if: } \mathbf{y \neq f(x)} \\ 0 & \text{otherwise} \end{cases}$

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

- In practice, we use **approximations** of the 0-1 loss

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

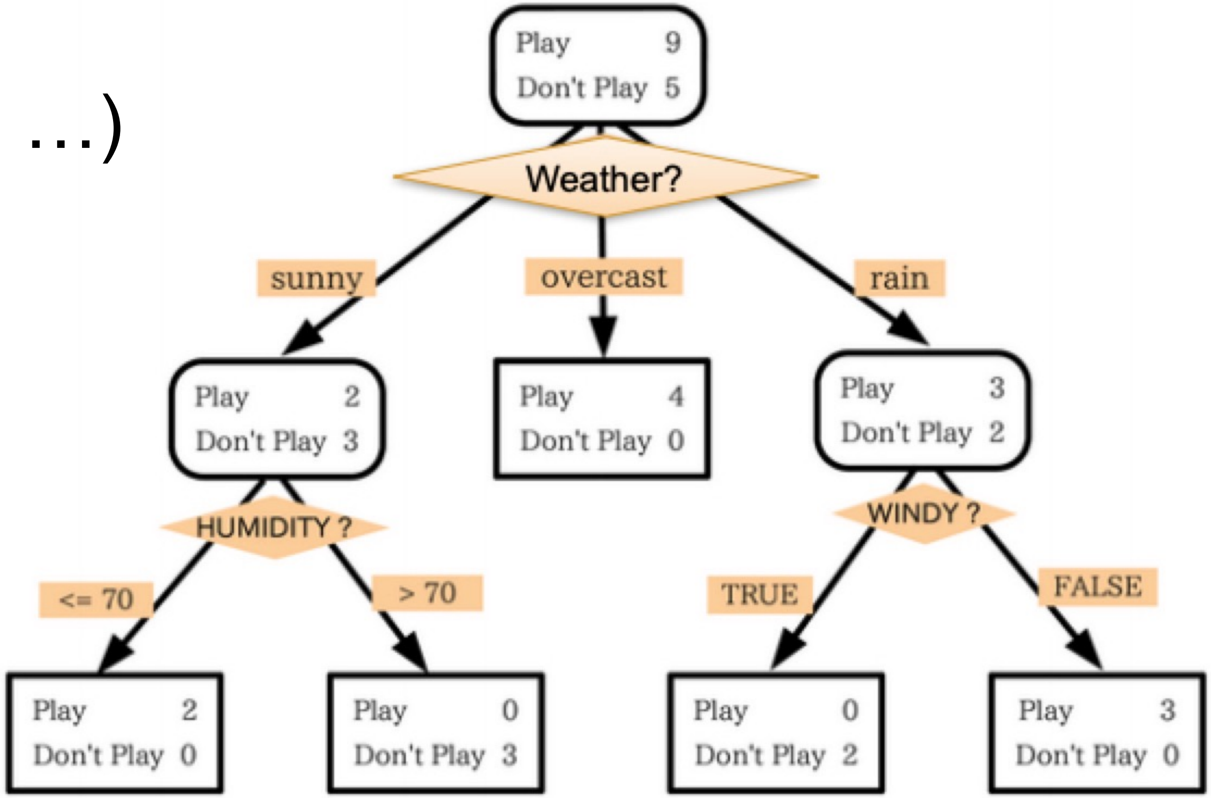
1. **Data**: $S = \{(x_i, y_i)\}_{i = 1,\dots,n}$
   - x_i: data example with d attributes
   - y_i: label of example (what you care about)

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

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

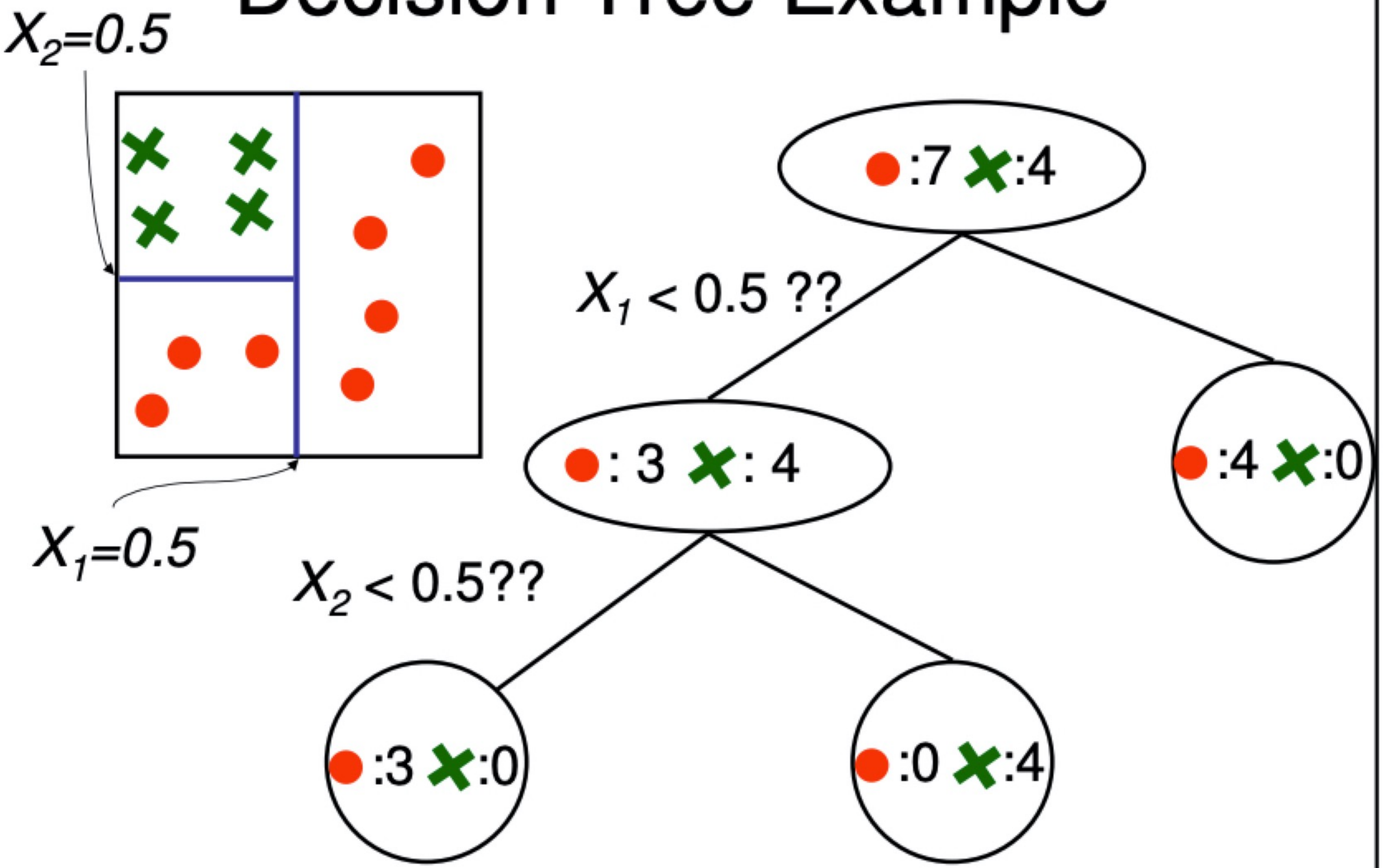
Decision Trees: To play tennis or not to?

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



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

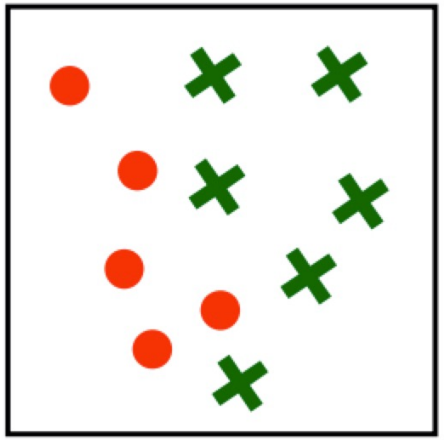
Decision Tree Example



Martial Hebert, CMU 15-381

Training (fitting) a Decision Tree

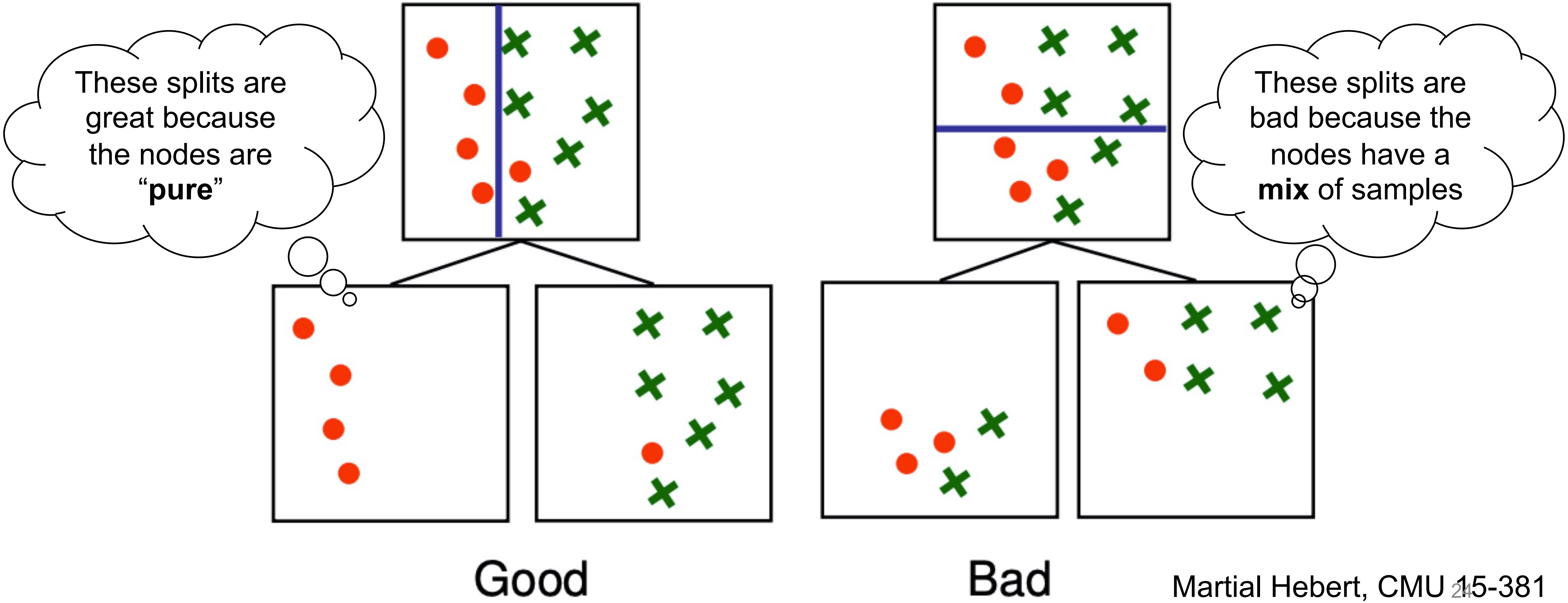
How to choose the attribute/value to split on at each level of the tree?



- Two classes (red circles/green crosses)
- Two attributes: X_1 and X_2
- 11 points in training data
- Idea → Construct a decision tree such that the leaf nodes predict correctly the class for all the training examples

Training (fitting) a Decision Tree

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

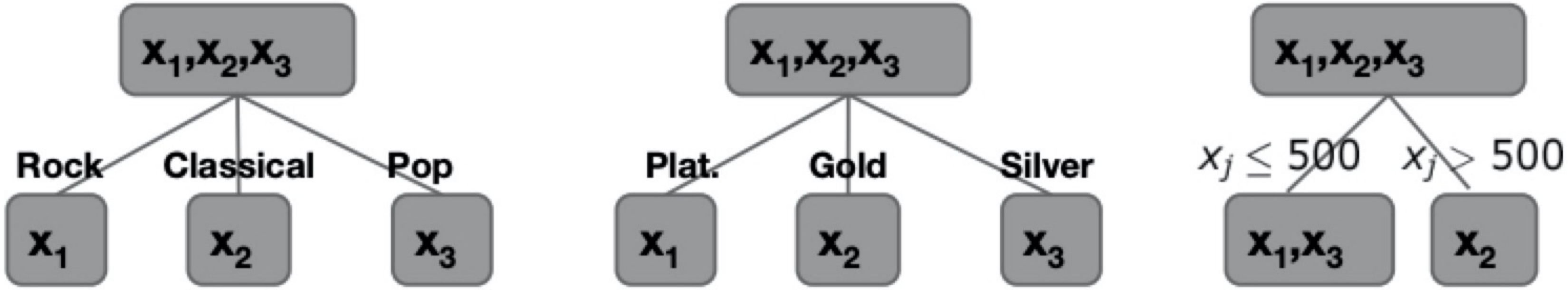


$$x_i = (x_{i1}, \dots, x_{id}); y_i = \{1, \dots, m\}$$

Choosing the split point

Split types for a selected attribute j:

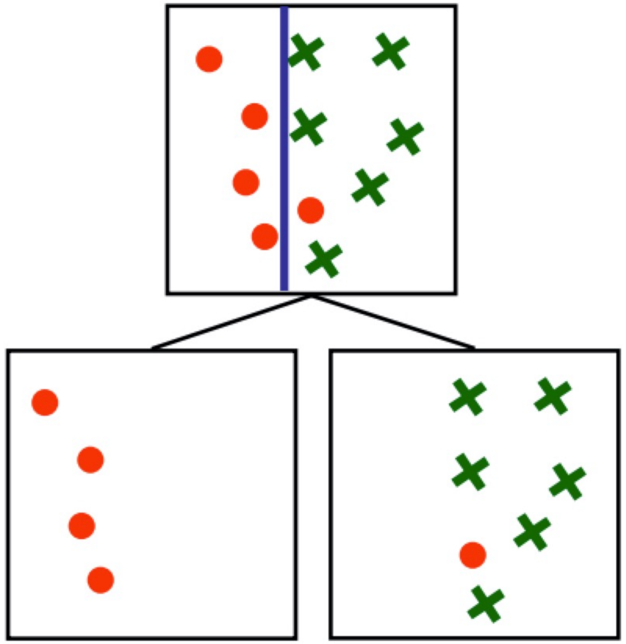
1. Categorical attribute (e.g. "genre")
   x_{1j} = Rock, x_{2j} = Classical, x_{3j} = Pop
2. Ordinal attribute (e.g., "achievement")
   x_{1j}=Platinum, x_{2j}=Gold, x_{3j}=Silver
3. Continuous attribute (e.g., song duration)
   $x_{1j} = 235$, $x_{2j} = 543$, $x_{3j} = 378$



Split on genre   Split on achievement   Split on duration

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

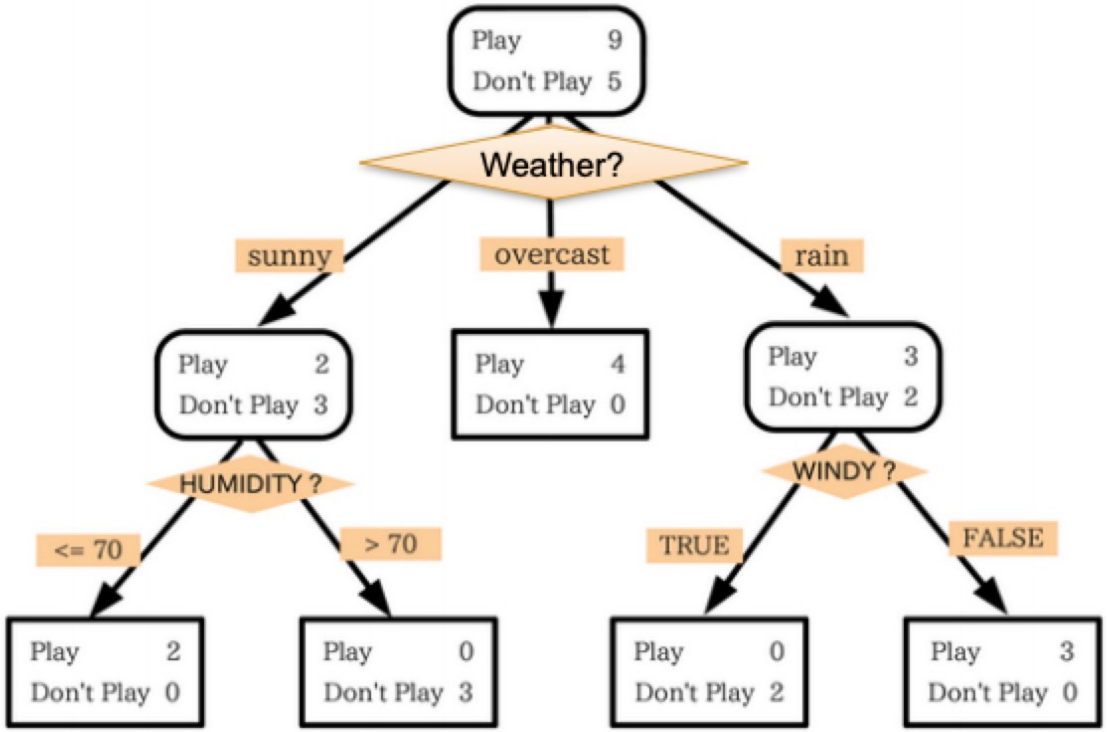
Final words on Decision Trees

Advantages

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

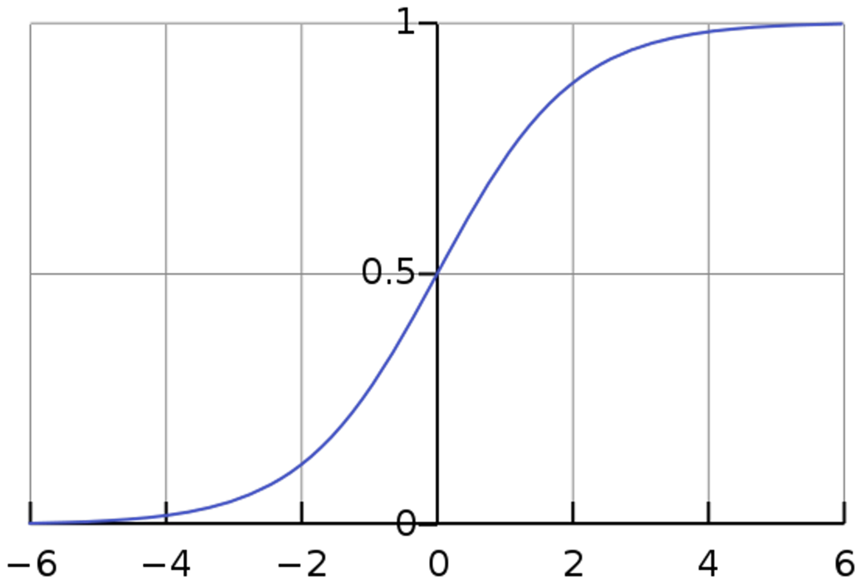
Caveats

- 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, #pregnancies=3, smoking=True, …)

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

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

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

This is the model!

Logistic Regression: Training

$$p(x;\boldsymbol{\beta}) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_d x_d)}}$$

Data: $S = \{(x_i, y_i)\}_{i=1,\ldots,n}$

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 $\boldsymbol{\beta}$:

$$\prod_{i=1}^{n} p(x_i;\boldsymbol{\beta})^{y_i} \times \left(1 - p(x_i;\boldsymbol{\beta})\right)^{1-y_i}$$

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

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

Logistic Regression: Properties

$$p(x;\boldsymbol{\beta}) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \cdots + \beta_d x_d)}}$$

Data: $S = \{(x_i, y_i)\}_{i=1,\ldots,n}$

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

1- Find $\boldsymbol{\beta}^*$ by Maximum Likelihood Estimation

2- For a new example x', compute $p(x';\boldsymbol{\beta}^*)$ and **threshold at 0.5**

Note that LR is a **linear model**!

Can you guess why?

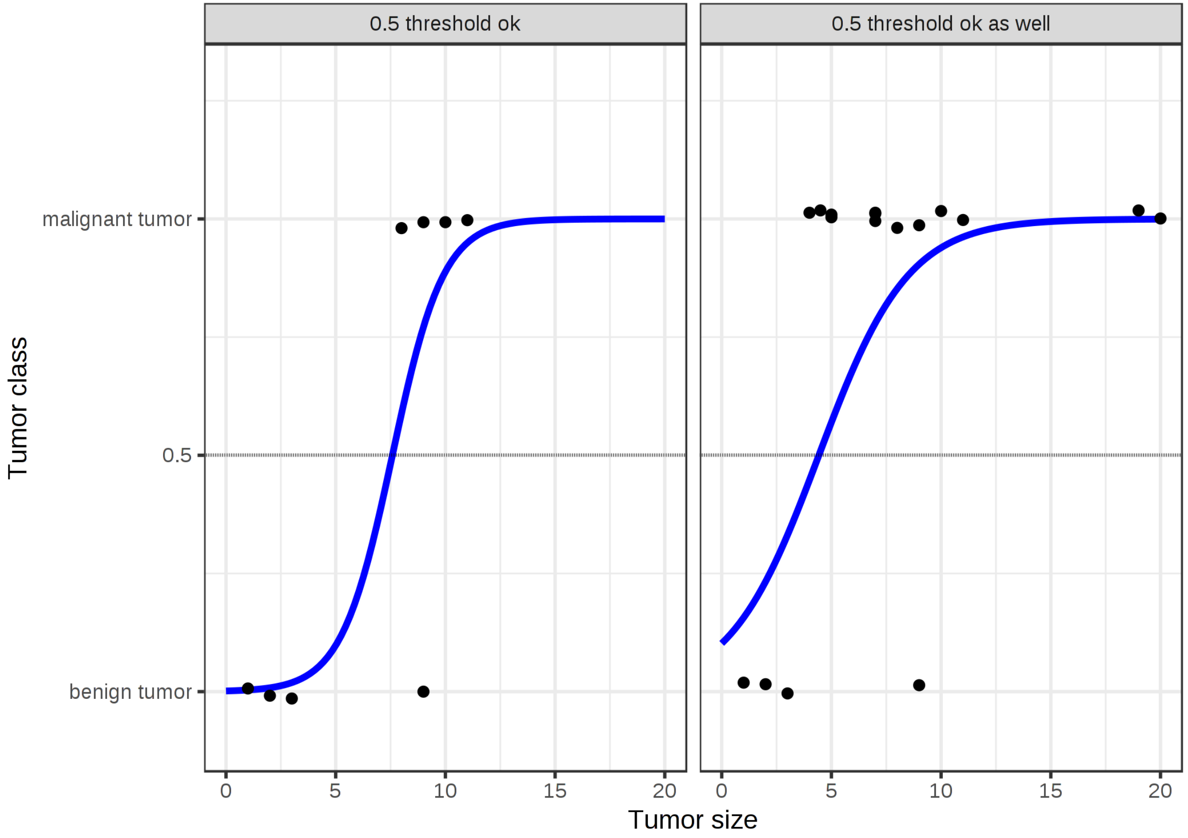
Final words on Logistic Regression

Advantages

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

Caveats

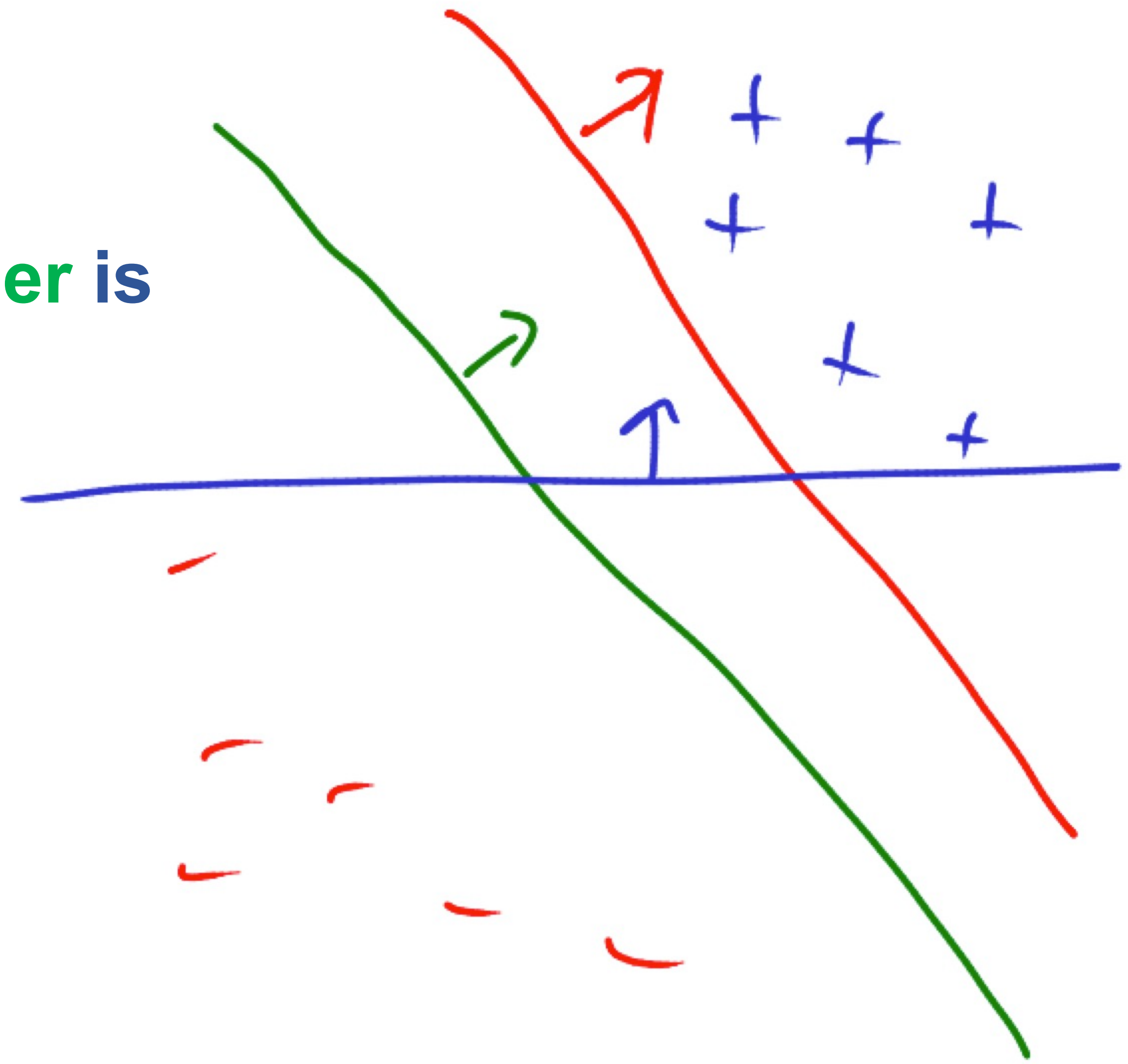
- A low-capacity, linear model



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

Support Vector Machines (SVM)

Which classifier is the best?

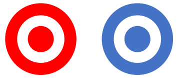


A Course in Machine Learning
by Hal Daumé III

SVM: The Maximum-Margin Principle

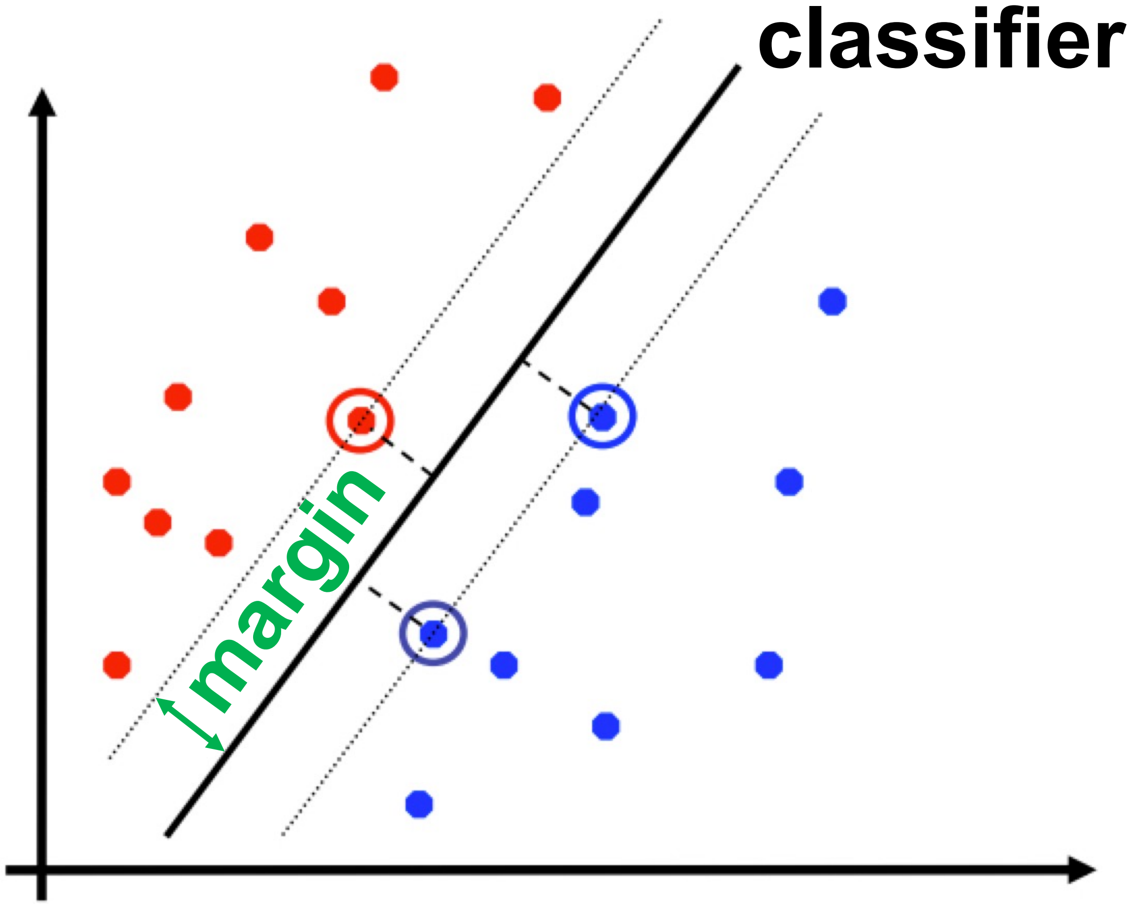
Vapnik (1990) derived the SVM as an “optimal” classifier

Intuitively, **robust** to outliers



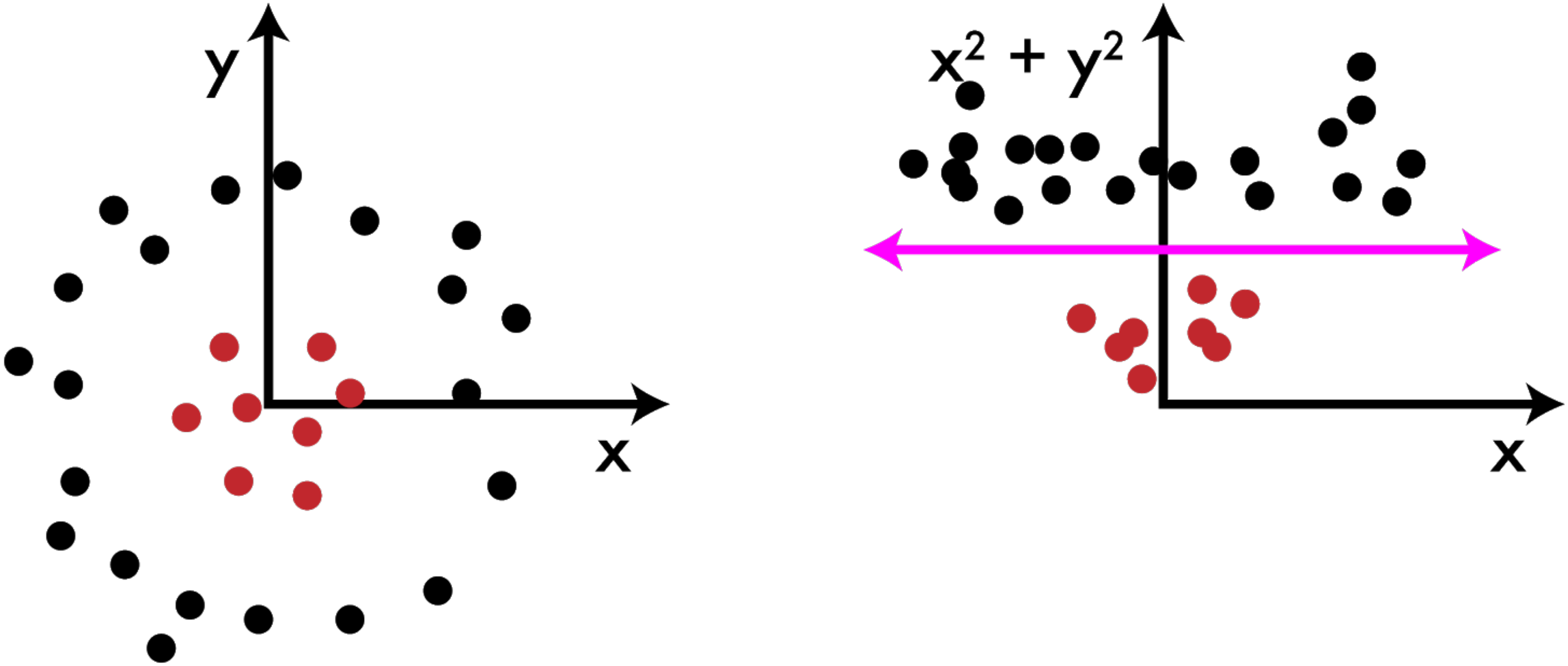
Support vectors: subset of data closest to classifier

Great empirical success in the 90s – early 2000s



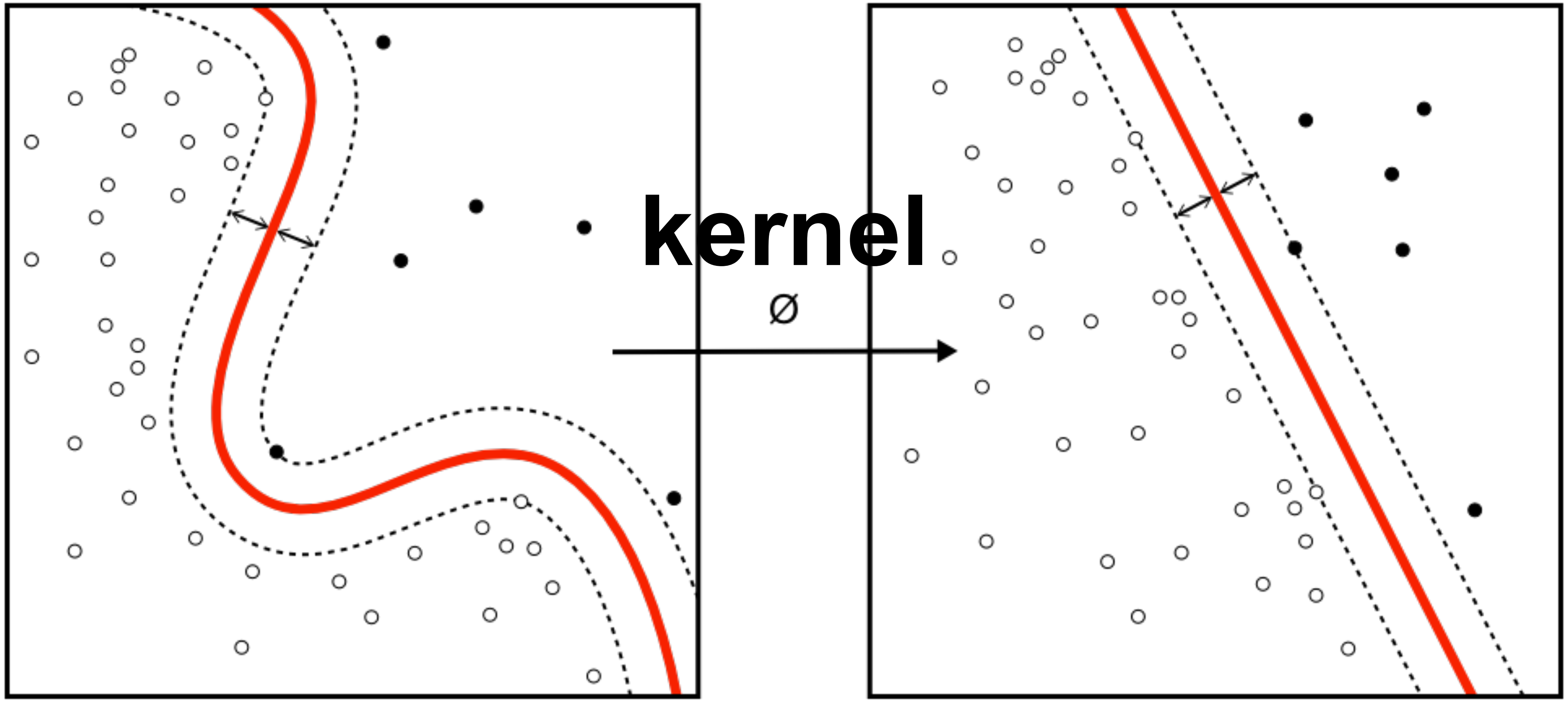
David Sontag, NYU ML class

What about **non-linearly** separable data?



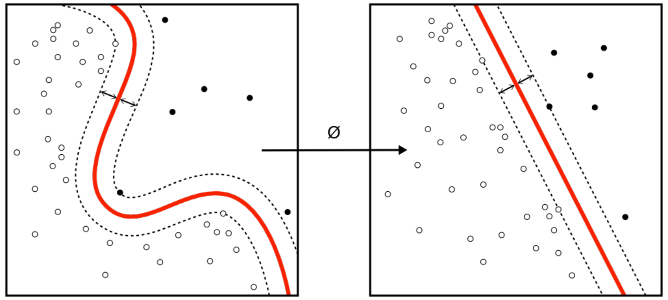
SVM for **non-linearly** separable data

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



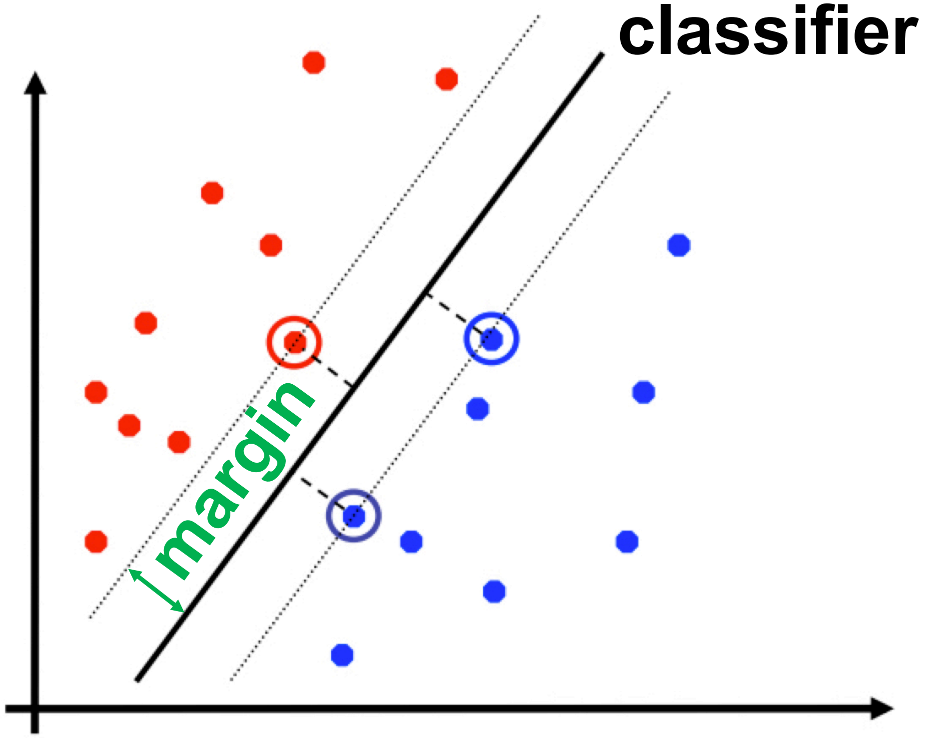
Wikipedia SVM article

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