Machine Learning (ML) 101
Methods in AI research

Roxana Rădulescu
September 2025

Utrecht University

Credit: Dong Nguyen

Practicalities

- **Literature for today:** Hal Daumé III, A Course in Machine Learning, Chapter 1 (Decision Trees; http://ciml.info/dl/v0_99/ciml-v0_99-ch01.pdf) and Chapter 2 (Limits of Learning; http://ciml.info/dl/v0_99/ciml-v0_99-ch02.pdf).

RECAP!

Last time

- **Dialog systems**
  - Chatbots vs. goal-based dialogue systems
  - We came across approaches for which we needed to:
    - Classify domain, intent, slot for frame-based approaches
    - Classify dialog acts
  - Rule-based vs. **machine learning** approaches

Natural Language 1

Egesiel Magalhães S.	**Loan Offer** - Do you need a Loan @ 2% PA? Mail us your: Names,Home Add,Mob No,Email id,Amount Needed,Lo...
Mr. Karim Zongo	**PLEASE THIS IS VERY URGENT.** - Compliment of the day, I am Mr. Karim Zongo Have a Business Proposal of $5...
CITIBANK OF NEW YORK	**NEW MESSAGE FROM CITIBANK NEW YORK** - CITIBANK INTERNATIONAL NEW YORK DIRECTOR, FOREIGN OPE...
MRS. CHRISTY MCCOOL	**MY DONATION OF 4 MILLION DOLLARS ARE YOU INTERESTED ?** - I am writing to seek your consent to conduct...

Spam classification

EN ←→ NL

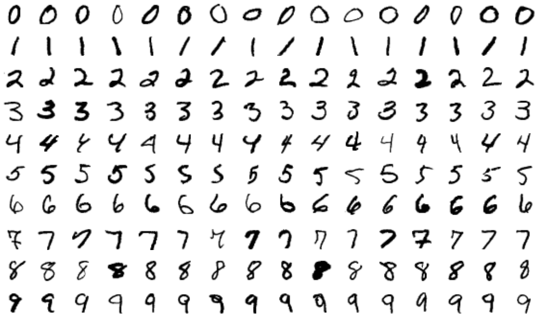
Machine translation

Intent: **SHOWFLIGHT**

`I want to fly to San Francisco on Monday afternoon please`

Intent classification

Image classification

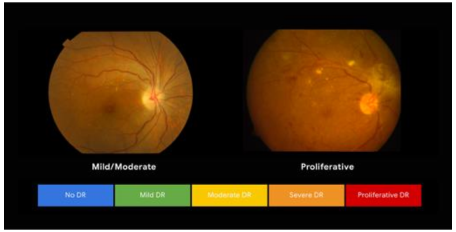


Digit recognition
MNIST dataset

IMAGENET

ImageNet has **21841** classes

http://image-net.org/explore



Diagnosing Diabetic Eye Disease
https://ai.google/healthcare/

What is Machine Learning?

There are many definitions, here is a useful one:

A computer program is said to ***learn*** *from experience* ***E*** *with respect to some class of tasks* ***T*** *and performance measure* ***P****, if its performance at tasks in* ***T****, as measured by* ***P****, improved with experience* ***E****.*

Tom Mitchell, Machine Learning 1997

What is Machine Learning?

Three components:

- Task T
- Experience E
- Performance measure P

Detect the dialog act of an utterance

T: Classify the dialog act of an utterance

P: The fraction of utterances correctly classified

E: A set of utterances labeled with their dialog acts

What is Machine Learning?

Three components:

- Task T
- Experience E
- Performance measure P

Self-driving cars

T: Drive on public highways using vision sensors

P: Average distance traveled before an error

E: Sequence of images and steering commands from human drivers

Experience

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

Experience

The focus of our lectures!

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

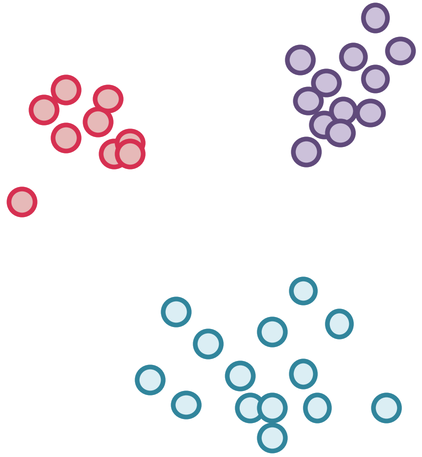
spam **not spam** **not spam**

Learn a model using **labelled** instances

Example: image classification, dialog act classification.

Experience

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning



Learn a model using **unlabelled** data

Example: community detection

Experience

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

Agent: conversational agent
Environment: reward model
Reward: score (e.g., helpful response)
Action: generated tokens

Supervised learning

RECAP!

Hand crafted rules: dialog systems

To recognize SET-ALARM intent:

```
wake me (up) | set (the|an) alarm | get me up
```

Hand crafted rules: spam classification

```
Spam list => spam
‘Buy’ AND (‘cheap’ OR ‘free’) => spam
```

Hand crafted rules: spam classification

```
Spam list => spam
'Buy' AND ('cheap' OR 'free') => spam
```

Very precise. Sometimes easier to fix mistakes.

Hand crafted rules: spam classification

```
Spam list => spam
'Buy' AND ('cheap' OR 'free') => spam
```

Very precise. Sometimes easier to fix mistakes.

Manually crafting rules takes a **lot** of time and is **difficult** to do.

High maintenance cost (e.g. need to adapt to changing language use)

Hand crafted rules: time-consuming!

- 232 industry categories and 504 occupation categories
- Manual rules
  - Development time = **192** person-months
- Machine learning
  - Development time = **4** person-months
  - More accurate!

COMMERCIAL APPLICATIONS OF MASSIVELY PARALLEL SUPERCOMPUTERS FOR THE 90'S
Waltz 1991.

Hand crafting rules for some tasks would be *really* difficult!

For example: author identification of texts

- It's (usually) not about the use of *specific* words, but about small differences between (relative) frequencies of words and grammatical constructions.

But collecting labels is easy…

Supervised learning

- Learn a machine learning model using **labeled example instances**

- Need to define **features**, characteristics of the instances that the model uses for predictions (words in a document, movie ratings, etc..)

Supervised learning

- Learn a machine learning model using **labeled example instances**

- Need to define **features**, characteristics of the instances that the model uses for predictions (words in a document, movie ratings, etc..)

Domain classification for dialog systems

```
I want to fly to San
Francisco on Monday
afternoon please

Domain: AIRLINE
```

Features: words

Supervised learning

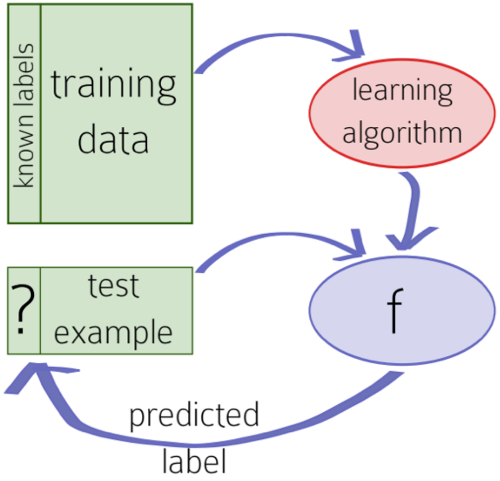
- Learn a machine learning model using **labeled example instances**

- Need to define **features**, characteristics of the instances that the model uses for predictions (words in a document, movie ratings, etc..)

Features for house price prediction:

- Overall condition of the house
- Neighborhood
- Condition of the basement
- Number of bedrooms
- Construction date
- First floor square meters
- Number of schools in within 2 km
- Condition of the kitchen
- ..

Supervised learning



CIML, figure 1.1

Setting:
X: input space (set of possible instances)
Y: output space

H = {f|f : X → Y}: set of hypotheses (the set of all possible classifiers we consider)

Learning:
i: current training example
Input: <$x^{(i)}$, $y^{(i)}$>: training examples
Learning algorithm: Defines a data-driven search over the hypothesis space

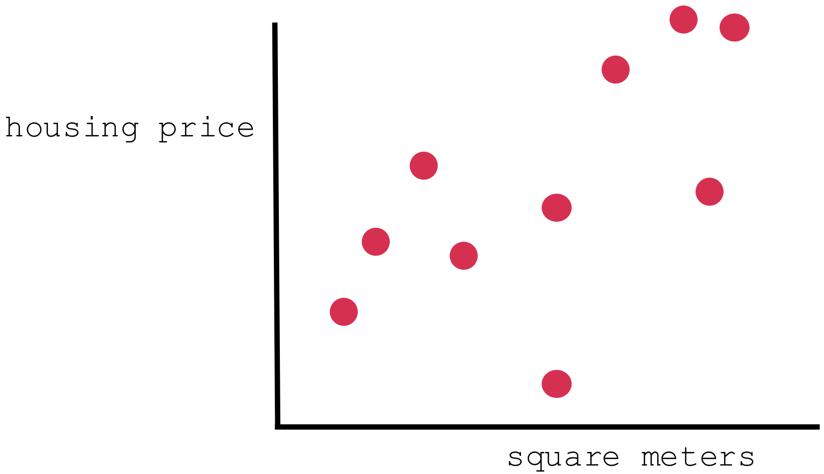
Output:
f ∈ F: hypothesis that approximates the target function

Tasks & data

features target

Input: $\{<x^{(1)}, y^{(1)}>, \ldots, <x^{(N)}, y^{(N)}>\}$

Goal: Predict the target using the features



Housing price prediction:
This is a *regression* problem (target is a real number)

What are the dimensions of the features and the target?

$x^{(i)} \in \mathbb{R}$ (one)
$y^{(i)} \in \mathbb{R}$ (one)

Tasks & data

features target

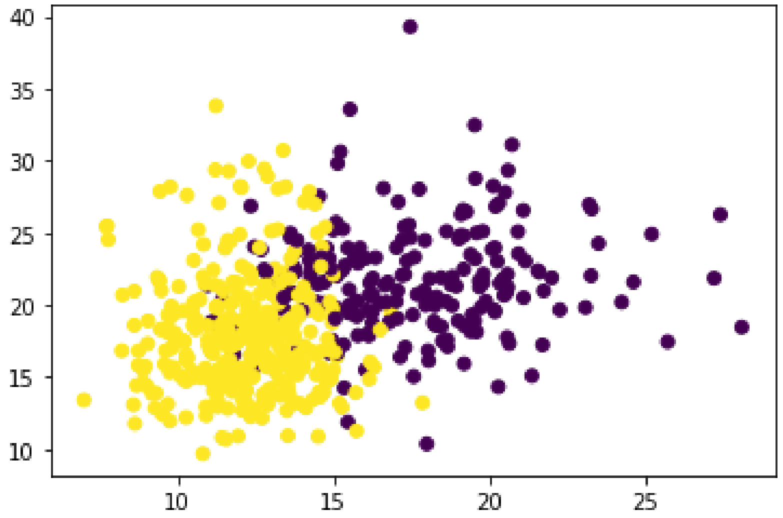
Input: $\{<x^{(1)}, y^{(1)}>, \ldots, <x^{(N)}, y^{(N)}>\}$

Goal: Predict the target using the features

Breast cancer diagnosis (malignant or benign):
This is a *classification* problem (target is a category)

What are the dimensions of the features and the target?

$x^{(i)} \in \mathbb{R}^2$ (two)
$y^{(i)} \in \{0,1\}$ (one)



Tasks & data

features target

Input: $\{<x^{(1)}, y^{(1)}>, \ldots, <x^{(N)}, y^{(N)}>\}$

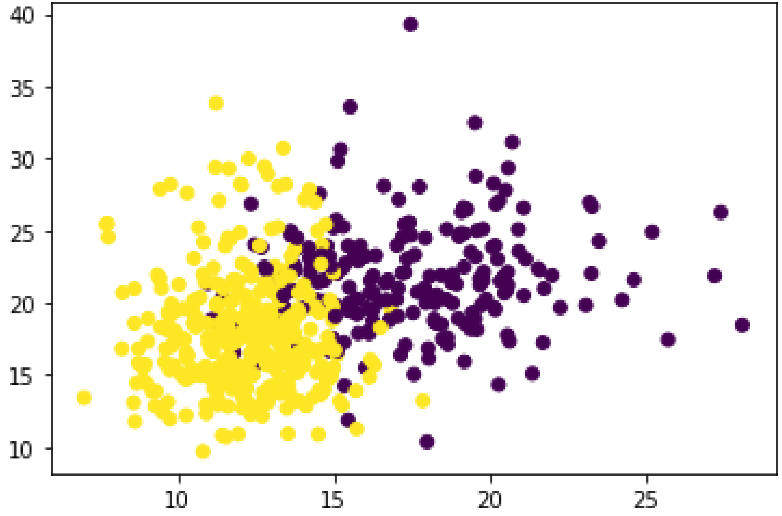
Goal: Predict the target using the features

Breast cancer diagnosis (malignant or benign):
This is a *classification* problem (target is a category)

The focus of our lectures!

What are the dimensions of the features and the target?

$x^{(i)} \in \mathbb{R}^2$ (two)
$y^{(i)} \in \{0,1\}$ (one)



Learning

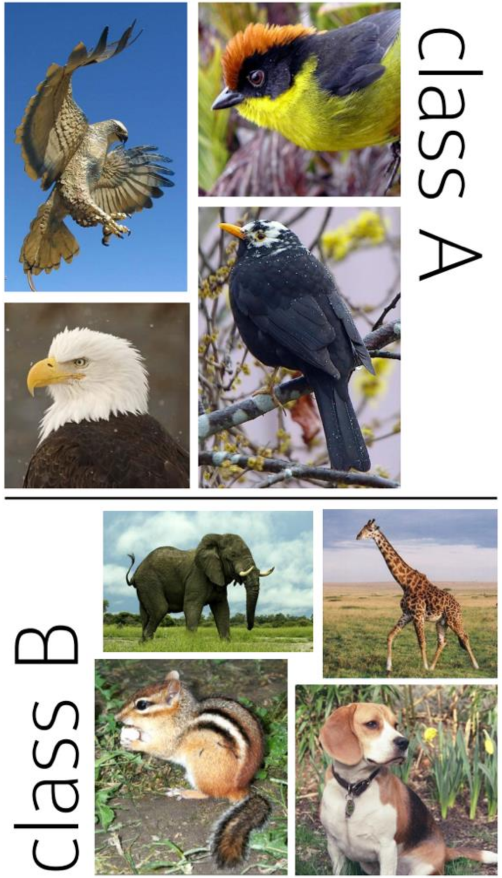
- **Generalization**
  - Training versus test examples
  - *Memorization is not enough!*

Training set

Test set

- **Inductive bias**
  - Allows a learning algorithm to prioritize one solution (or interpretation) over another, independent of the observed data (Battaglia et al. 2018, Mitchell 1980)

Training data



Inductive bias

Test data

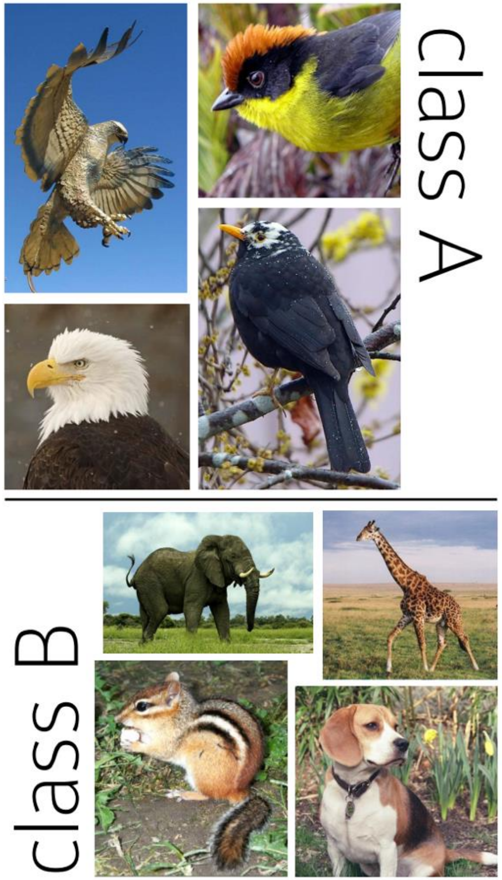
Question: How would you label the test data?

Underlying assumptions to generalize to new input!

[CIML 2.1 and 2.2]

Inductive bias

Training data



[CIML 2.1 and 2.2]

Test data

ABBA:
bird vs. non-bird

AABB:
Fly vs. no-fly

Underlying assumptions to generalize to new input!

Supervised machine learning for classification

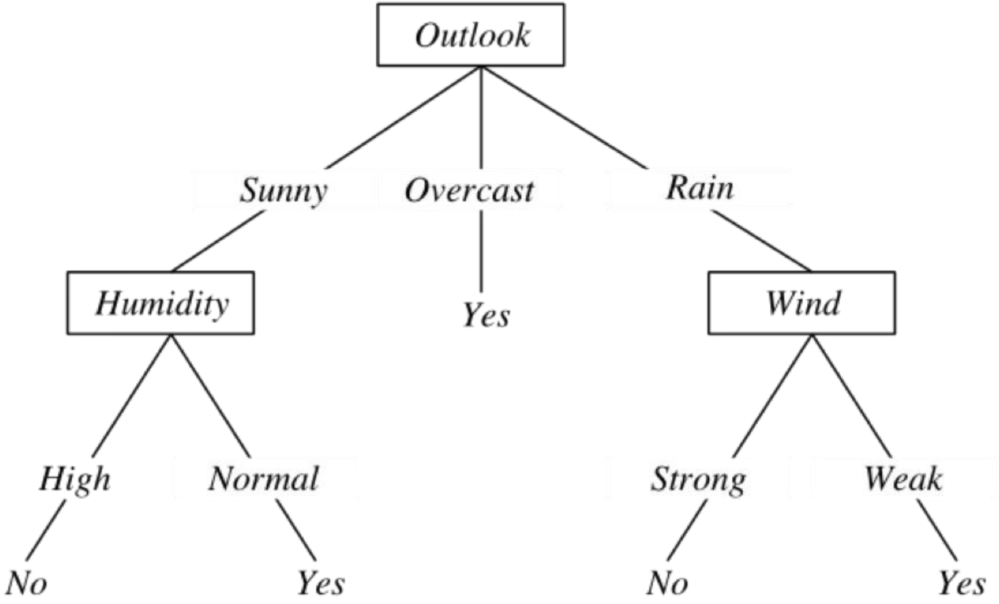
- Naive Bayes
- Logistic Regression
- Support Vector Machines (SVM)
- Neural networks
- Decision trees
- K-nearest neighbors
- And many more…

Supervised machine learning for classification

- Naive Bayes
- **Logistic Regression**
- Support Vector Machines (SVM)
- **Neural networks**
- **Decision trees**
- **K-nearest neighbors**
- And many more…

Decision Trees

Example



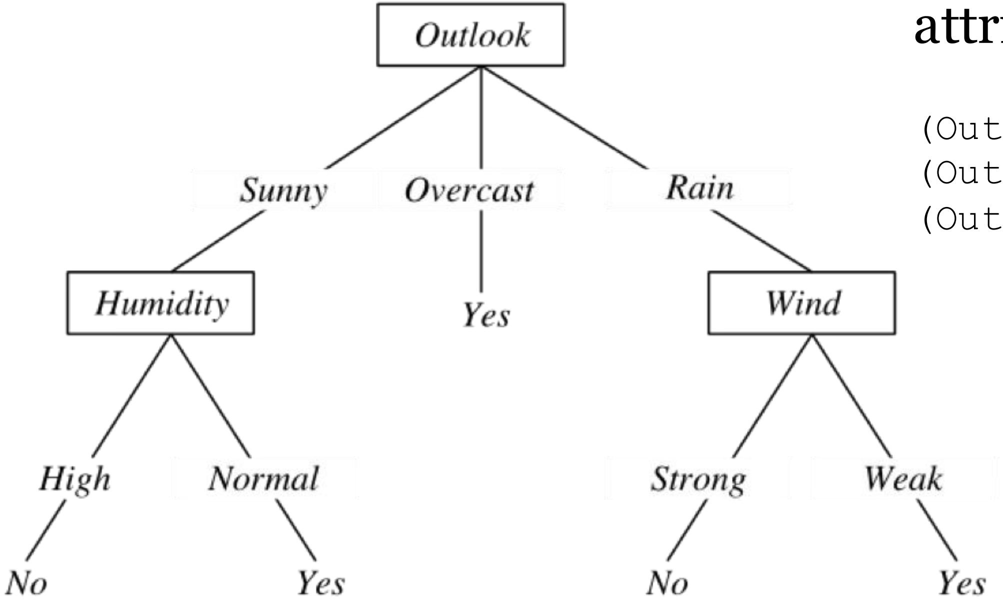
Is it a good time to play tennis?

`(Outlook = Sunny,`
`Temperature = Hot,`
`Humidity = High,`
`Wind = Strong)`

Answer: **No**

Machine Learning, Tom Mitchell (p. 53)

Example



Decision trees represent disjunction of conjunctions of constraints on the attribute values

```
(Outlook = Sunny ∧ Humidity = Normal) V
(Outlook = Overcast) V
(Outlook = Rain ∧ Wind = Weak)
```

Decision trees can be represented as if-then rules (helps interpretability ☺)

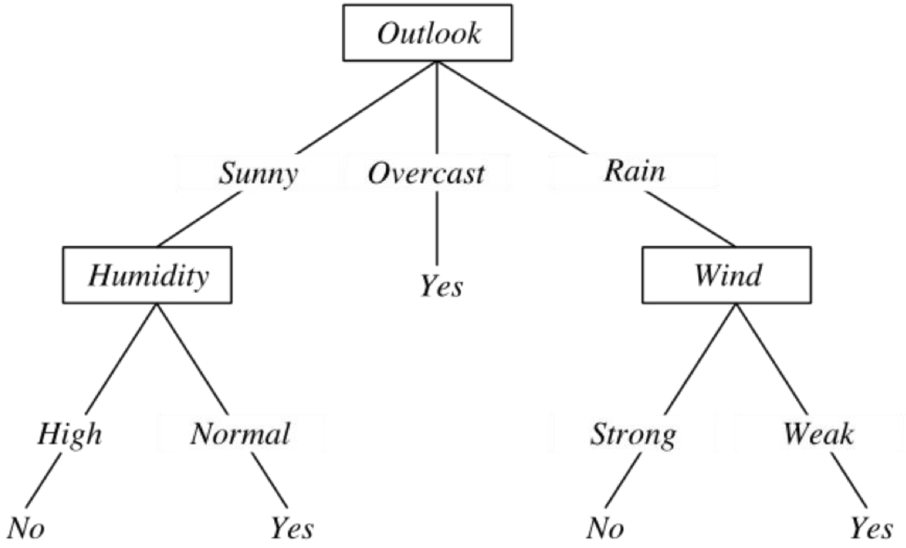
Machine Learning, Tom Mitchell (p. 53)

Decision trees - Representation

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

How would we represent:

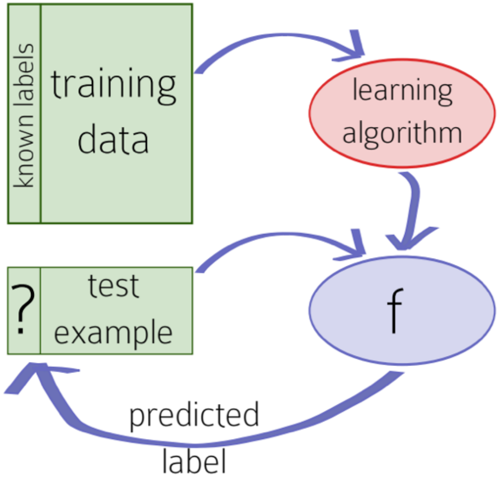
- $\wedge$, $\vee$, XOR
- $(A \wedge B) \vee (C \wedge \neg D \wedge E)$



Decision trees - Representation

$(A \wedge B) \vee (C \wedge \neg D \wedge E)$

Supervised learning



CIML, figure 1.1

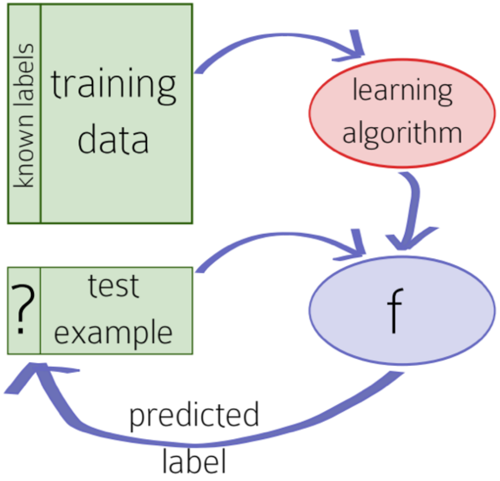
Setting:
X: input space (set of possible instances)
Y: output space

H = {f|f : X → Y}: set of hypotheses (the set of all possible classifiers we consider)

Learning:
i: current training example
Input: <$x^{(i)}$, $y^{(i)}$>: training examples
Learning algorithm: Defines a data-driven search over the hypothesis space

Output:
f ∈ F: hypothesis that approximates the target function

Supervised learning



CIML, figure 1.1

Setting:

X: input space (set of possible instances)

Y: output space

y=1: good time to play tennis 0: not a good time

H = {f|f : X → Y}: set of hypotheses (the set of all possible classifiers we consider) Set of all possible decision trees

Learning:

i: current training example

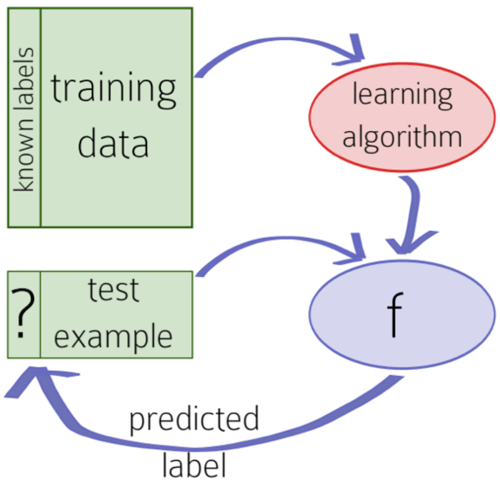
Input: <$x^{(i)}$, $y^{(i)}$>: training examples

Learning algorithm: Defines a data-driven search over the hypothesis space

Output:

f ∈ F: hypothesis that approximates the target function

Supervised learning



CIML, figure 1.1

Setting:

X: input space (set of possible instances)

Y: output space

y=1: good time to play tennis 0: not a good time

H = {f|f : *X* → Y}: set of hypotheses (the set of all possible classifiers we consider) Set of all possible decision trees

Learning:

i: current training example

Input: <$x^{(i)}$, $y^{(i)}$>: training examples

Learning algorithm: Defines a data-driven search over the hypothesis space

Output:

f ∈ F: hypothesis that approximates the target function

Learning decision trees

Find the 'best' tree h ∈ H, i.e. the tree that minimizes training error, or maximizes training accuracy

What about doing an exhaustive search?
Computationally infeasible

Instead: We use a greedy search

Learning decision trees

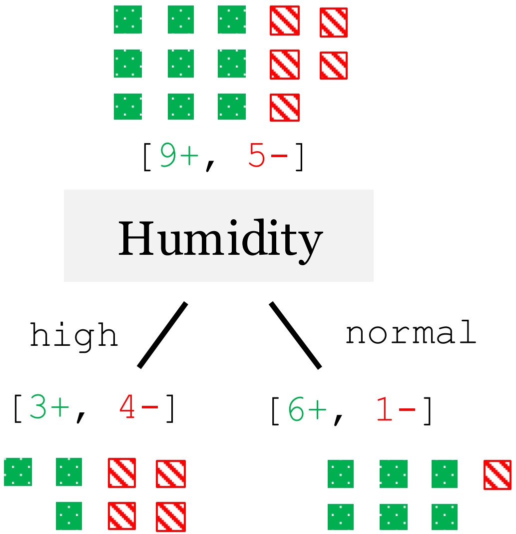
```
Start with empty tree

Base cases:
If all instances have the same label →
   create a leaf with that label and exit
If no features left to split →
    create a leaf with the majority label

Else:
Select the best test to split the data on
Split the data according to the test
Recurse on each subset of the data
```

Learning decision trees

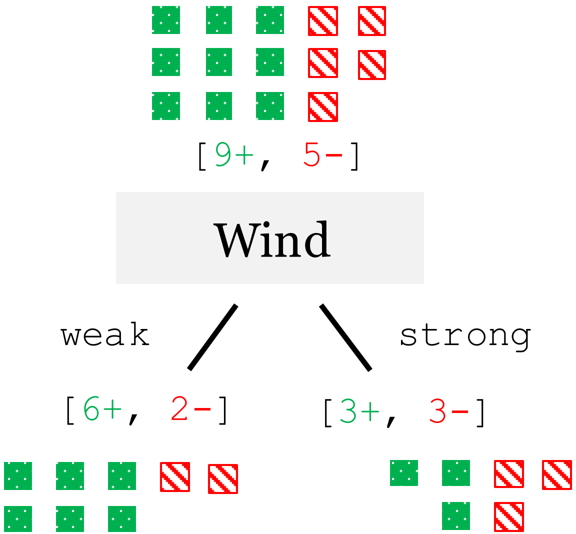
```
Start with empty tree

Base cases:
If all instances have the same label →
   create a leaf with that label and exit
If no features left to split →
    create a leaf with the majority label

Else:
Select the best test to split the data on
Split the data according to the test
Recurse on each subset of the data
```

Selecting attributes to split

Is it a good time to play tennis?



Question:
which one would you choose?



Example from Machine Learning, Tom Mitchell

Selecting attributes to split

We want to be more certain about the label after splitting:

After split:

All instances have the same label 😃

Uniform distribution over labels 😔

Selecting attributes to split

We want to be more certain about the label after splitting:

After split:

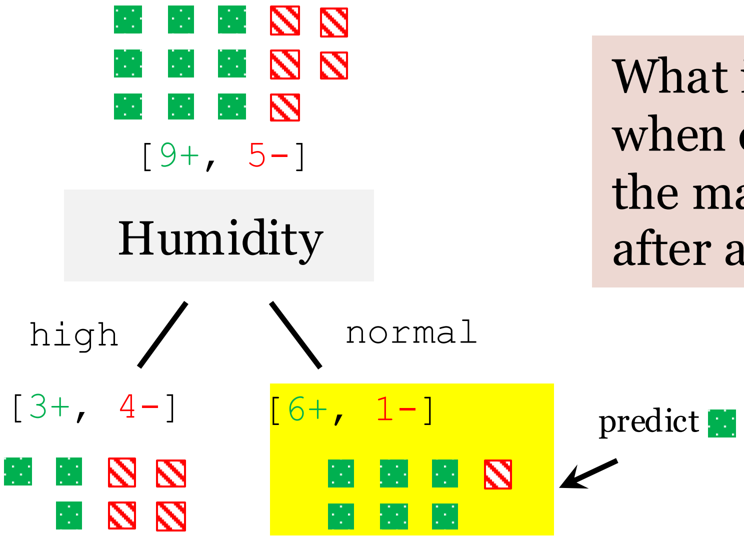
All instances have the same label 😃

Uniform distribution over labels 😔

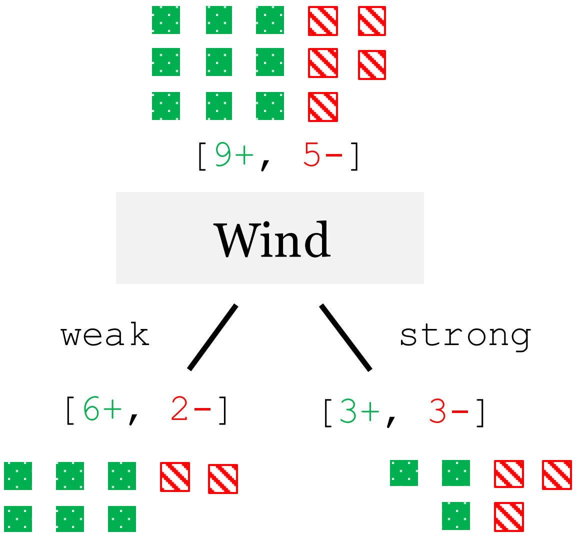
How can we quantify this intuition?

Selecting attributes to split: misclassification rate

Is it a good time to play tennis?



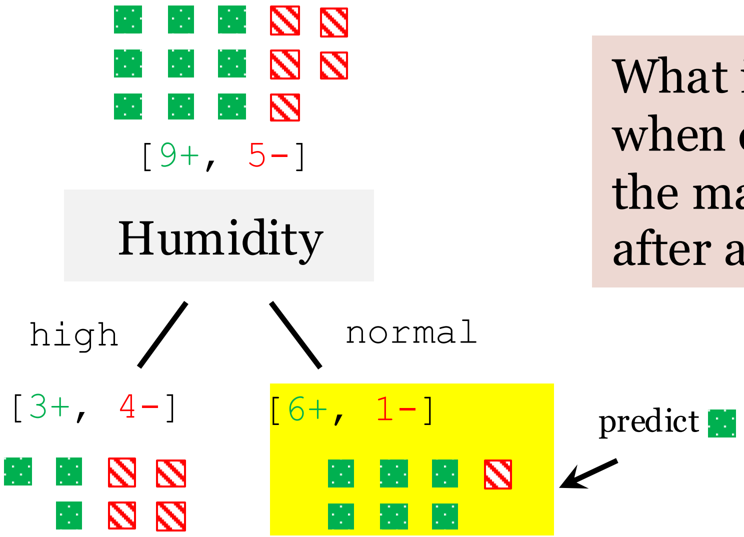
What is the error when choosing the majority label after a split?



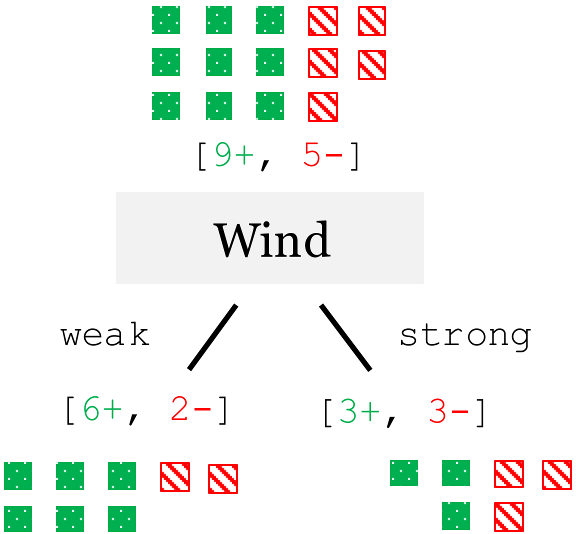
Example from Machine Learning, Tom Mitchell

Selecting attributes to split: misclassification rate

Is it a good time to play tennis?



What is the error when choosing the majority label after a split?



(3 + 1) / 14 = 0.286

(2 + 3) / 14 = 0.357

Example from Machine Learning, Tom Mitchell

Selecting attributes to split: Information Gain

Gain(S,A) = expected reduction in **entropy** due to sorting on *A*

Machine Learning, Tom Mitchell

Selecting attributes to split: Information Gain

Gain(S,A) = expected reduction in **entropy** due to sorting on *A*

Decision tree method: **ID3** (**Iterative Dichotomiser 3**)

- selects the attribute that **maximises** information gain

Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81–106

Machine Learning, Tom Mitchell

Entropy

Entropy(S) = expected number of bits needed to encode class (⊕ or ⊖) of randomly drawn member of S (under the optimal, shortest-length code)

Why?

Information theory: optimal length code assigns $-\log_2 p$ bits to message having probability p

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

Entropy

Entropy(S) = expected number of bits needed to encode class ($\oplus$ or $\ominus$) of randomly drawn member of S (under the optimal, shortest-length code)

Why?

Information theory: optimal length code assigns $-\log_2 p$ bits to message having probability p

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

So, expected number of bits to encode $\oplus$ or $\ominus$ of random member of S:

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Entropy

Entropy(S) = expected number of bits needed to encode class ($\oplus$ or $\ominus$) of randomly drawn member of S (under the optimal, shortest-length code)

Why?

Information theory: optimal length code assigns - $\log_2 p$ bits to message having probability p

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

So, expected number of bits to encode $\oplus$ or $\ominus$ of random member of S:

$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

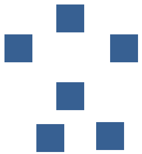
with c-wise classification, we get

$$Entropy(S) = \sum_{i=1}^{c} -p_i \log_2 p_i$$

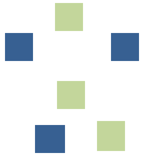
Entropy

Entropy:

$$Entropy(S) = -\sum_i p_i \log_2 p_i$$



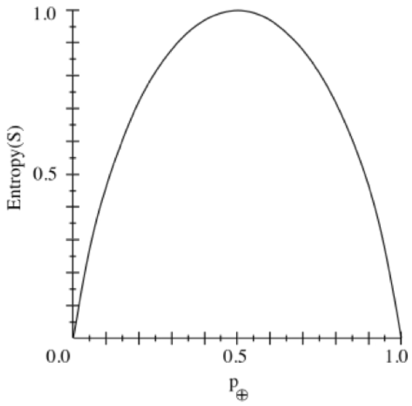
Entropy = 0



Entropy = 1

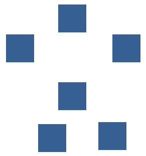
p_i: the probability of class i (i.e. the fraction of instances of class i in S)

"the amount of randomness"



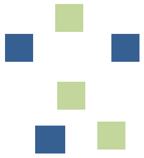
Entropy

Entropy: $$Entropy(S) = -\sum_i p_i \log_2 p_i$$



Entropy = 0

p_i: the probability of class i (i.e. the fraction of instances of class i in S)



Entropy = 1

"the amount of randomness"

[9+, 5-] $-(9/14) \log_2 (9/14) - (5/14) \log_2 (5/14) = 0.940$
[7+, 7-] $-(7/14) \log_2 (7/14) - (7/14) \log_2 (7/14) = 1$
[14+, 0-] $-(14/14) \log_2 (14/14) - (0/14) \log_2 (0/14) = 0$

Entropy

Entropy:

$$Entropy(S) = -\sum_i p_i \log_2 p_i$$

p_i: the probability of class i (i.e. the fraction of instances of class i in S)

"the amount of randomness"

"the average number of yes/no questions to guess a draw from S"

	p
A	0.5
B	0.25
C	0.25

What strategy would use you to guess my draw? (A,B, or C).

Entropy

Entropy:

$$Entropy(S) = -\sum_i p_i \log_2 p_i$$

p_i: the probability of class i (i.e. the fraction of instances of class i in S)

"the amount of randomness"

"the average number of yes/no questions to guess a draw from S"

	p
A	0.5
B	0.25
C	0.25

A? yes → A				1 * 0.5 = 0.5
no	→ B? yes	→ B		2 * 0.25 = 0.5
	no	→ C		2 * 0.25 = 0.5

On average we need 1.5 questions

-0.5*log2(0.5)-0.25*log2(0.25)
-0.25*log2(0.25) = 1.5

Entropy

Entropy:

$$Entropy(S) = -\sum_i p_i \log_2 p_i$$

p_i: the probability of class i (i.e. the fraction of instances of class i in S)

"the amount of randomness"

"the average number of yes/no questions to guess a draw from S"

	p
A	0.5
B	0.25
C	0.25

A? yes → A $1 * 0.5 = 0.5$
no → B? yes → B $2 * 0.25 = 0.5$
no → C $2 * 0.25 = 0.5$

On average we need 1.5 questions

$-0.5*\log2(0.5)-0.25*\log2(0.25)$
$-0.25*\log2(0.25) = 1.5$

Entropy

Entropy:

$$Entropy(S) = -\sum_i p_i \log_2 p_i$$

	p
A	0.5
B	0.25
C	0.25

p_i: the probability of class i (i.e. the fraction of instances of class i in S)

"the amount of randomness"

"the average number of yes/no questions to guess a draw from S"

A? yes → A 1 * 0.5 = 0.5
no → B? yes → B 2 * 0.25 = 0.5
no → C 2 * 0.25 = 0.5

On average we need 1.5 questions

-0.5*log2(0.5)-0.25*log2(0.25)
-0.25*log2(0.25) = 1.5

Information Gain

Information Gain:
Entropy before you split – entropy after split
(weighted by probability of following each branch)

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Selecting attributes to split: Information Gain

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

Which attribute is the best next node?

Selecting attributes to split: Information Gain

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

[9+,5-]

$p_\oplus = 9/14$

$$Entropy(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

$p_\ominus = 5/14$

Selecting attributes to split: Information Gain

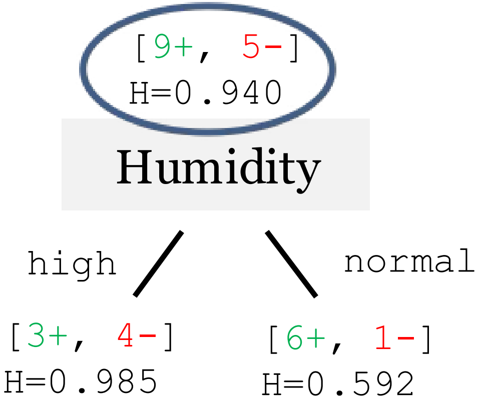
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

[9+,5-]

$p_{\oplus} = 9/14$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

$p_{\ominus} = 5/14$



Selecting attributes to split: Information Gain

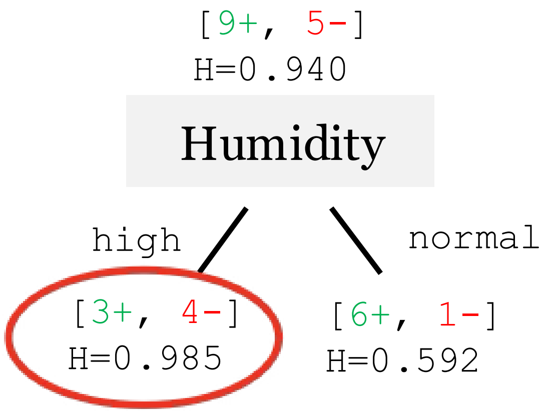
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

[3+,4-]

$p_{\oplus}$ = 3/7

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

$p_{\ominus}$ = 4/7



Selecting attributes to split: Information Gain

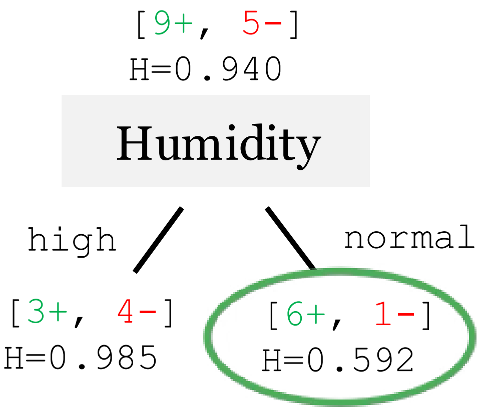
Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	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

[6+,1-]

$p_\oplus = 6/7$

$$Entropy(S) \equiv -p_\oplus \log_2 p_\oplus - p_\ominus \log_2 p_\ominus$$

$p_\ominus = 1/7$



Selecting attributes to split: Information Gain

[9+, 5-]
H=0.940

Humidity

high / \ normal

[3+, 4-] H=0.985 [6+, 1-] H=0.592

Gain(S, Humidity)
= 0.940 -
(7/14) * 0.985 -
(7/14) * 0.592 =
0.1515

Information Gain:
Entropy before you split – entropy after split (weighted by probability of following each branch)

[9+, 5-]
E=0.940

Wind

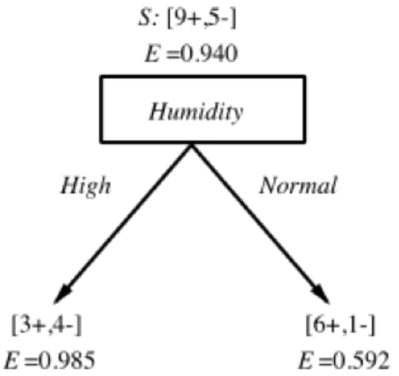
weak / \ strong

[6+, 2-] H=0.811 [3+, 3-] H=1.0

Gain(S, Wind) = 0.940 -
(8/14) * 0.811
- (6/14) * 1 = 0.048

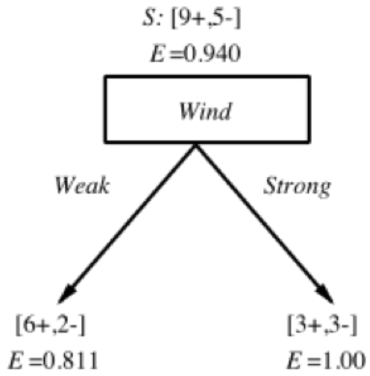
Selecting attributes to split: Information Gain

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



Gain (S, Humidity)

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



Gain (S, Wind)

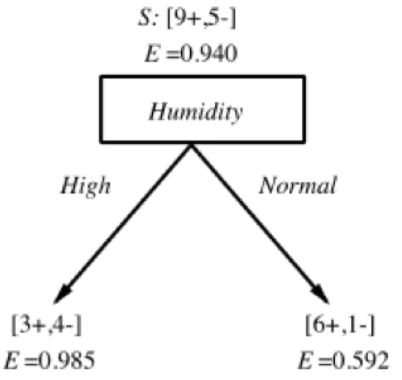
= .940 - (8/14).811 - (6/14)1.0
= .048

$$Gain(s, Outlook) = 0.246$$

$$Gain(s, Temperature) = 0.029$$

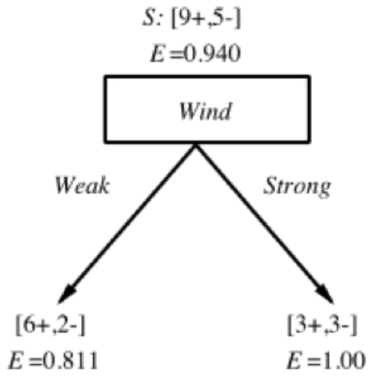
Selecting attributes to split: Information Gain

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



Gain (S, Humidity)

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



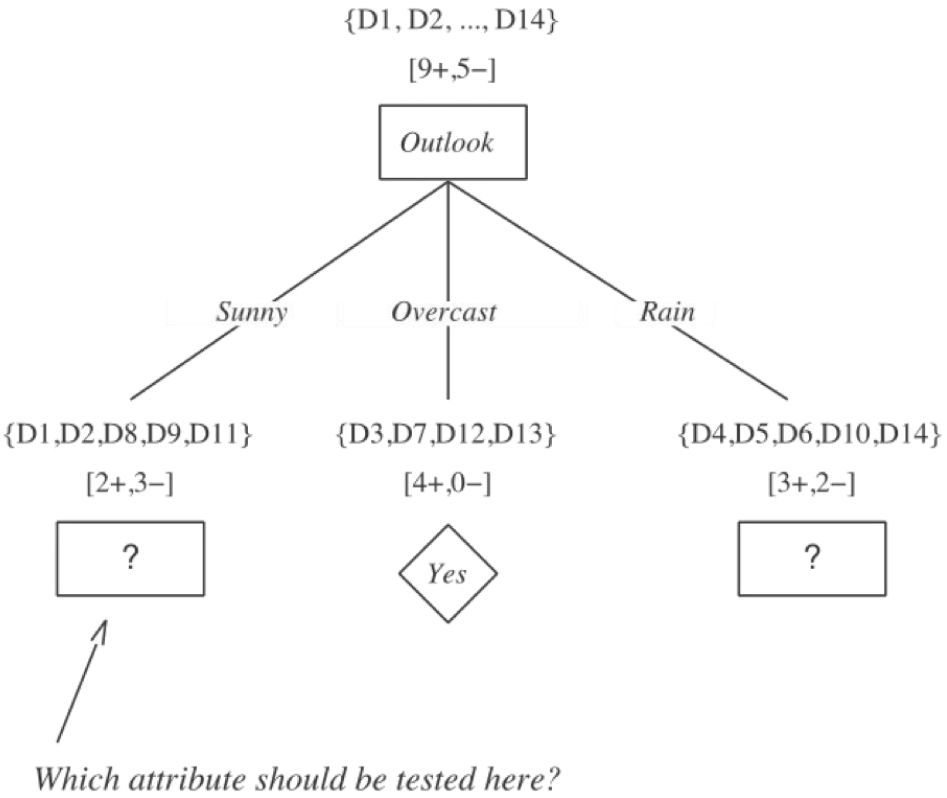
Gain (S, Wind)

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

$$Gain(s, Outlook) = 0.246$$

$$Gain(s, Temperature) = 0.029$$

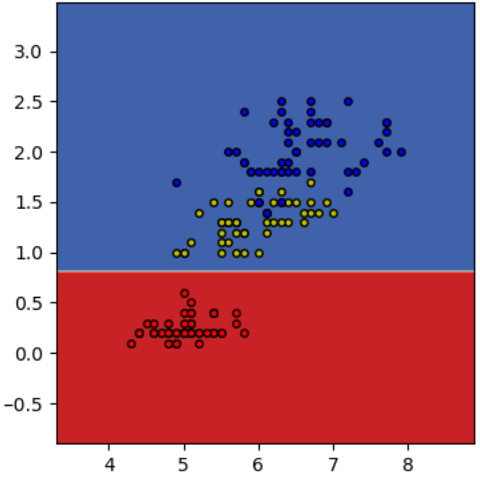
Selecting attributes to split: Information Gain



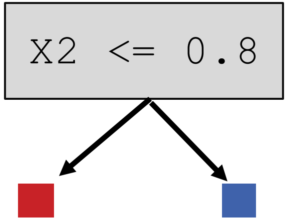
Decision boundary

max depth = 1

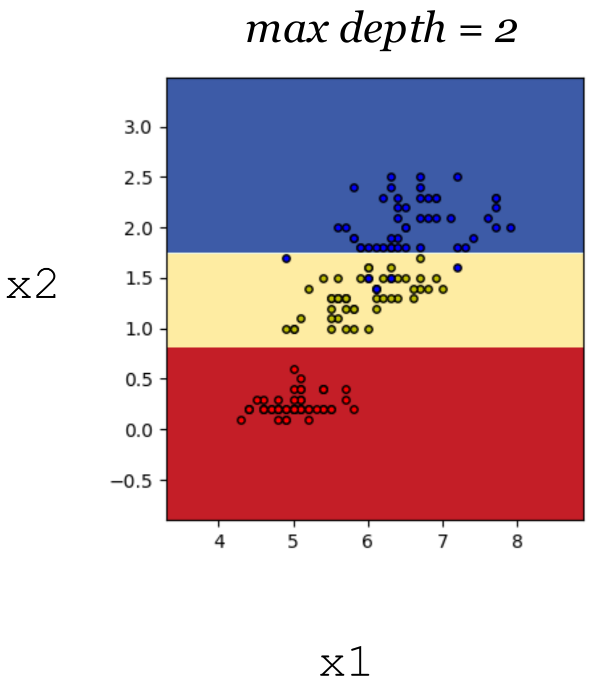
x2

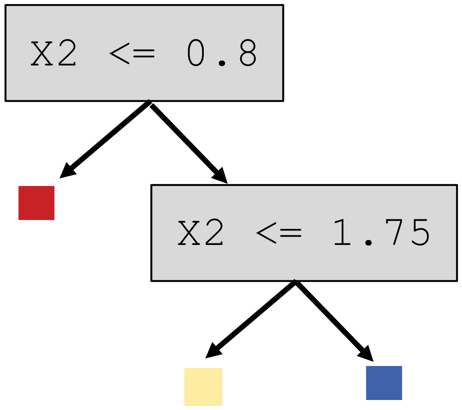


x1



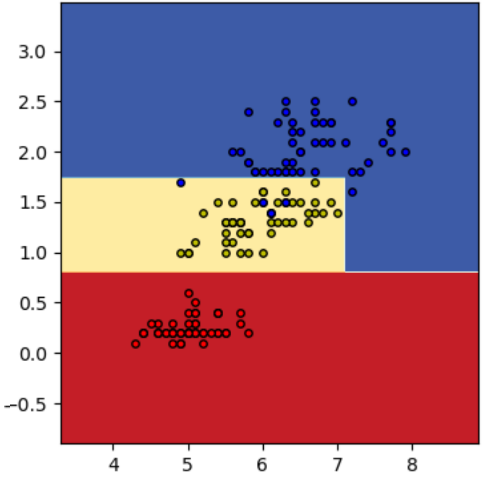
Decision boundary



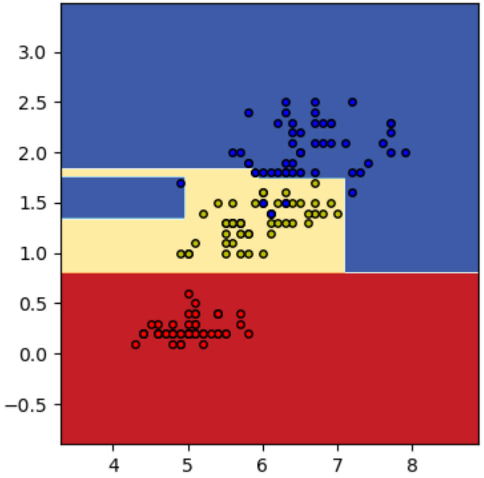


Decision boundary

max depth = 3



max depth = 25

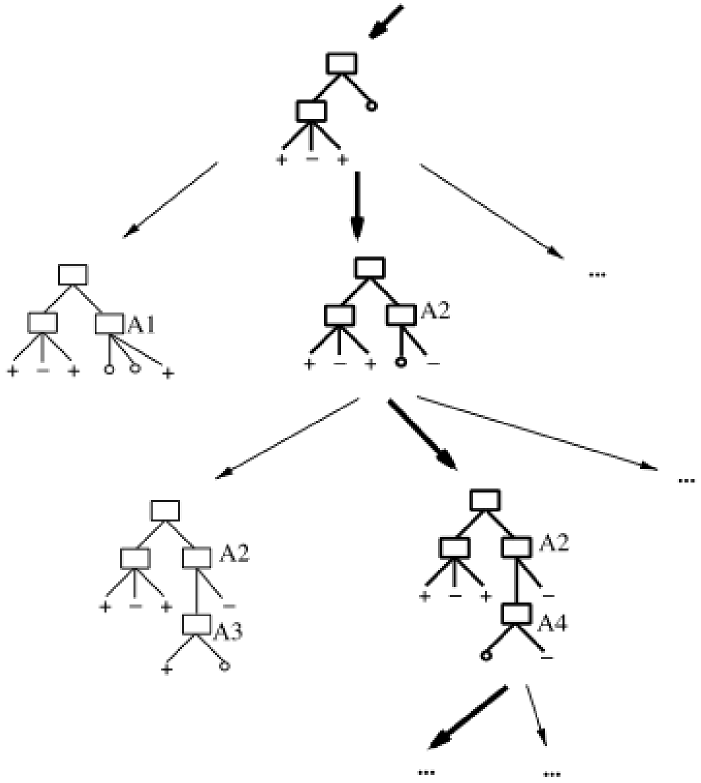


Hypothesis Space Search by ID3

ID3 algorithms perform a hill-climbing search

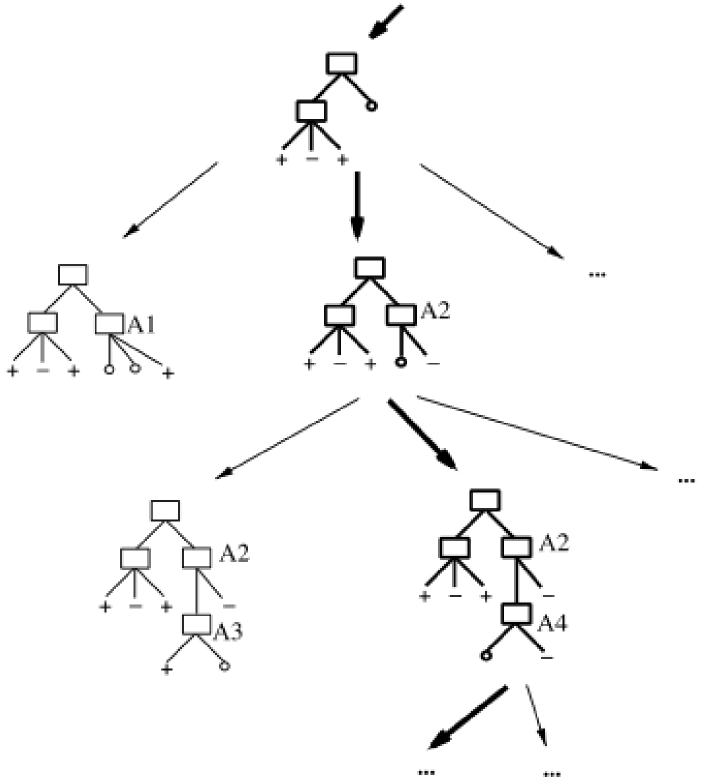
- The nodes of the tree are partial decision trees

- The evaluation function = information gain



Hypothesis Space Search by ID3

- Hypothesis space is complete!
  - Target function surely in there…
- Outputs a single hypothesis (which one?)
  - The one that leads us to the answer in few questions
- No back tracking
  - Local minima
- Statistically-based search choices
  - Robust to noisy data



Inductive bias

Underlying assumptions to generalize to new input!
What type of solutions are we more likely to prefer?

E.g., prefer smaller models with similar training accuracy (e.g. shallow decision trees), i.e. decisions can be made by only looking at a small number of features.

When to consider Decision Trees

- Instances describable by attribute – value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data

Model selection

Model selection

- **Features:** Words, user profile, etc.

- **Model:** Decision trees, or maybe something different?

Model selection

- **Features:** Words, user profile, etc.
- **Model:** Decision trees, or maybe something different?

We are interested in how well the model **generalizes**!

i.e. how does it perform on data it hasn't seen before?

Classification: Accuracy

$$\frac{\texttt{\#correctly labeled instances}}{\texttt{\#total instances}}$$

Confusion Matrix:

	Truth: A	Truth: B
Predicted: A	70	40
Predicted: B	30	60

Classification: Accuracy

$$\frac{\texttt{\#correctly labeled instances}}{\texttt{\#total instances}}$$

Confusion Matrix:

	Truth: A	Truth: B
Predicted: A	70	40
Predicted: B	30	60

Accuracy: 130/200 = 0.65

Train & test data

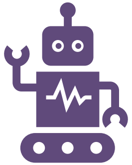
Dataset

Training set

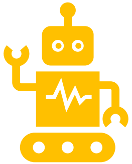
Test set

Train your model on this data!

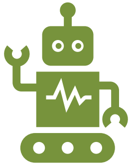
Test your model on this data



***80%** accurate*
on the test data



***85%** accurate*
on the test data



***64%** accurate*
on the test data

Train & test data

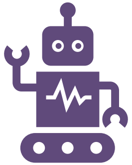
Dataset

Training set | Test set

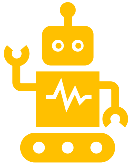
system X:
83% accuracy
on the test set

Train your model on this data!

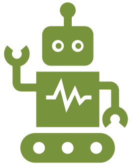
Test your model on this data



80% *accurate*
on the test data



85% *accurate*
on the test data



64% *accurate*
on the test data

Train & test data

My model is better!! It is 85% accurate

Hold on...

Dataset

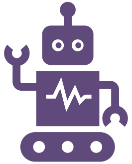
Training set

Test set

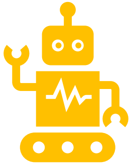
system X:
83% accuracy
on the test set

Train your model on this data!

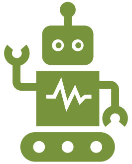
Test your model on this data



***80%** accurate*
on the test data



***85%** accurate*
on the test data



***64%** accurate*
on the test data

Warning!!

Training error is an optimistic estimate of your system's true error. So evaluate on a holdout test set. But…

Make sure no knowledge about the test data leaks into your model. So, you should **NEVER** do **ANY** learning on the test set.

- Feature development
- Selecting the model
- …

Train & dev & test data

Dataset

Training set | Dev set | Test set

Train your model on this data!

Tune your model on this data

When you're done with experimenting with features, tuning parameters etc. test your final model using this data.

Train & dev & test data

Dataset

Training set

Dev set

Test set

Train your model on this data!

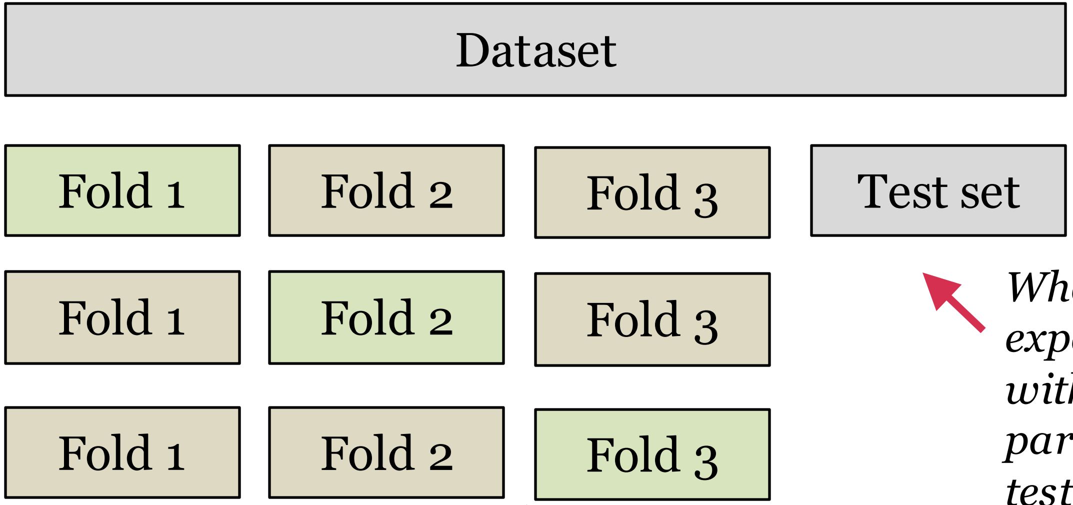
Tune your model on this data

When you're done with experimenting with features, tuning parameters etc. test your final model using this data.

Example:

1. Train five different decision trees (max depth = 2, 5, 10, 15, 20) on the *training* set.
2. Evaluate their performance on the *dev.* set.
3. Select the best one and run in on the *test* set.

Cross validation



When you're done with experimenting with features, tuning parameters etc. test your final model using this data.

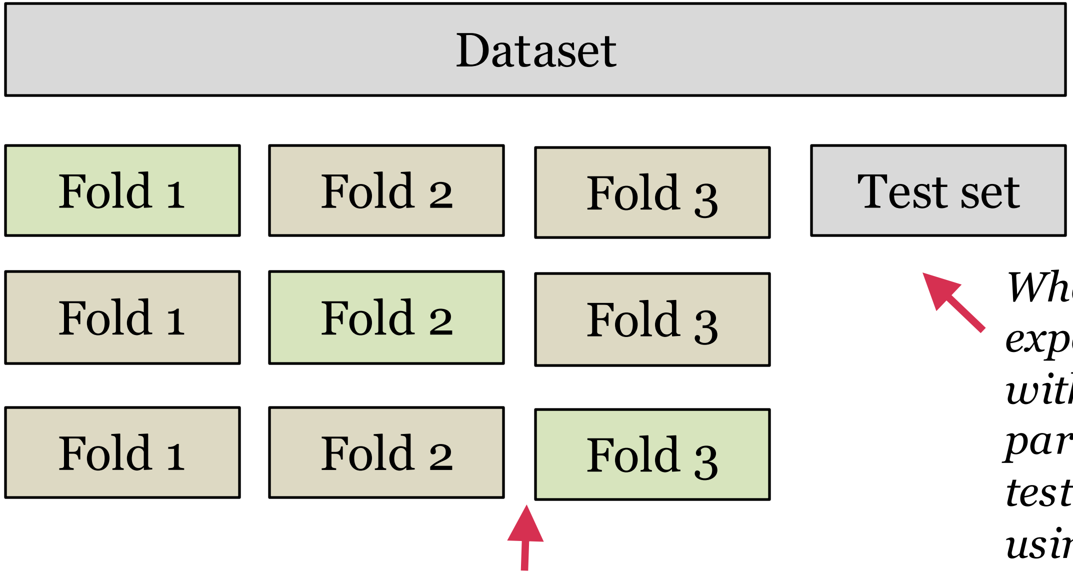
Train and tune your parameters on folds 1-3

E.g. train on folds 2 and 3, test on fold 1.
Usually 10 folds (i.e. 10-fold cross validation), but depends on the data

Cross validation

leave-one-out cross validation:

number of folds = number of data points



When you're done with experimenting with features, tuning parameters etc. test your final model using this data.

Train and tune your parameters on folds 1-3

E.g. train on folds 2 and 3, test on fold 1.
Usually 10 folds (i.e. 10-fold cross validation), but depends on the data

Overfitting and underfitting

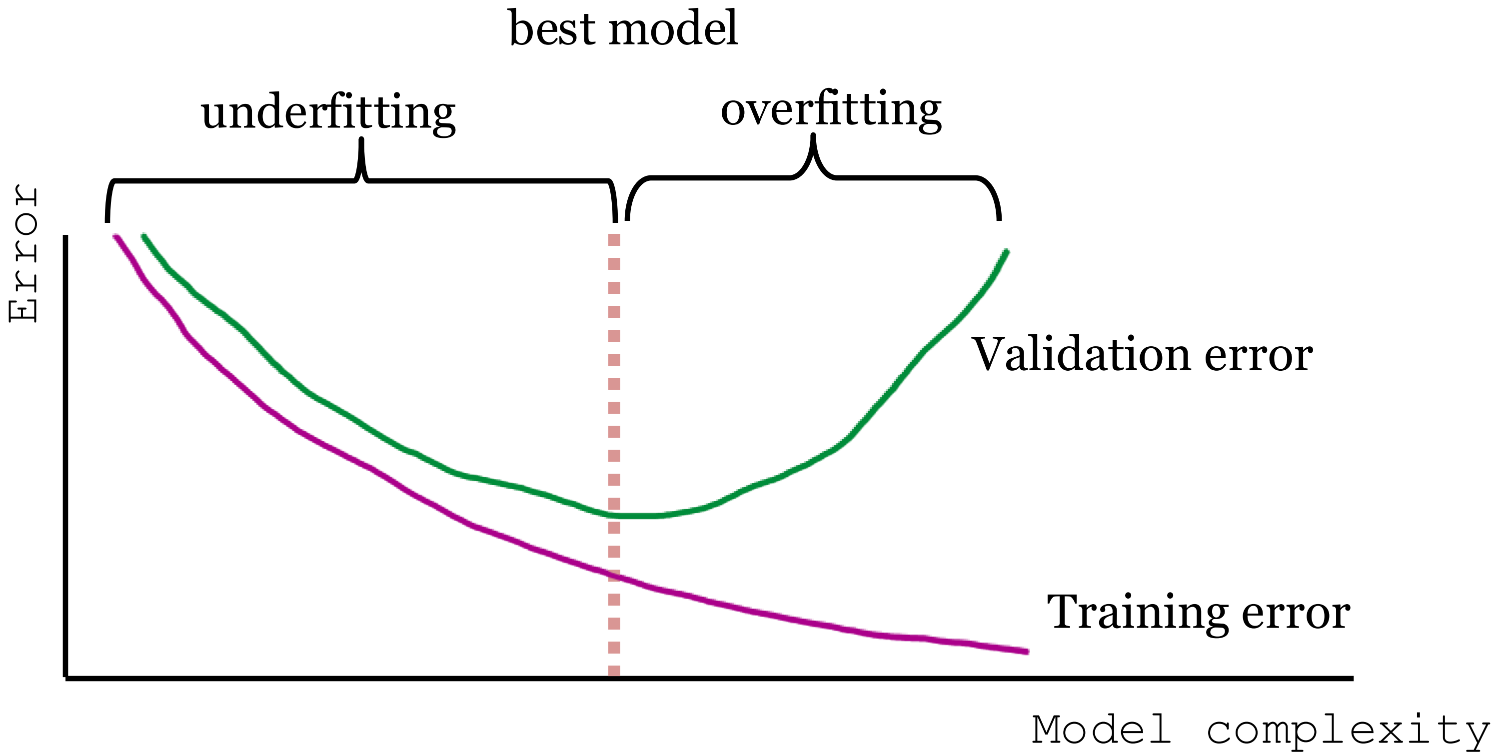
Underfitting: The model is too simple. It could have learned something but didn't.
Example: A decision tree which always predicts the same label (majority class)

Overfitting: The model pays too much attention to idiosyncrasies of the training data.
Example: a leaf for each instance in your training data (training error will be zero!) .

Constrain to simpler trees

- Max. depth
- Max. number of leaves
- Minimum number of instances per leaf

Overfitting and underfitting



Parameters vs. hyper parameters

- **Parameters:** The weights or structure selected by the learning algorithm

- **Hyperparameters:** 'Parameters that control the other parameters'. 'Things' we can tune but are not selected by the learning algorithm.

Parameters vs. hyper parameters

- **Parameters:** The weights or structure selected by the learning algorithm

- **Hyperparameters:**
  'Parameters that control the other parameters'. 'Things' we can tune but are not selected by the learning algorithm.

Decision trees

Parameters:
structure of a specific decision tree

Hyperparameters:
Maximum depth, minimum number of instances per leaf, ..

> *Cannot be naively adjusted using the training data, because increasing max depth will always reduce the training error!*

Evaluation metrics

Y	X1	X2
0	0	1
0	0	0
0	0	1
1	0	1
1	0	1
1	1	1
1	1	0
1	1	0
1	1	0
1	1	0

Accuracy

$$\frac{\texttt{\#correctly labeled instances}}{\texttt{\#total instances}}$$

Question: What is the accuracy of a classifier that would predict the majority label?

Evaluation metrics

Y	X1	X2
0	0	1
0	0	0
0	0	1
1	0	1
1	0	1
1	1	1
1	1	0
1	1	0
1	1	0
1	1	0

Accuracy

$$\frac{\texttt{\#correctly labeled instances}}{\texttt{\#total instances}}$$

Question: What is the accuracy of a classifier that would predict the majority label?

Accuracy is not suitable when the class distribution is (heavily) skewed!

Evaluation metrics

	Truth: A	Truth: B
Predicted: A	True Positive (TP)	False Positive (FP)
Predicted: B	False Negative (FN)	True Negative (TN)

$$accuracy = \frac{\#TP + \#TN}{\#TP + \#FP + \#FN + \#TN}$$

Evaluation metrics

	Truth: A	Truth: B
Predicted: A	70 (TP)	40 (FP)
Predicted: B	30 (FN)	60 (TN)

Precision for class A:
70/110 = 0.64

$$precision = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FP}}$$

What fraction of the ones that you have identified belong to that class?

Of all messages labeled as spam, what fraction is actually spam?

Evaluation metrics

	Truth: A	Truth: B
Predicted: A	70 (TP)	40 (FP)
Predicted: B	30 (FN)	60 (TN)

$$precision = \frac{\#TP}{\#TP + \#FP}$$

Precision for class A:
70/110 = 0.64

Precision for class B:
60/90= 0.67

What fraction of the ones that you have identified belong to that class?

Of all messages labeled as spam, what fraction is actually spam?

Evaluation metrics

	Truth: A	Truth: B
Predicted: A	70 (TP)	40 (FP)
Predicted: B	30 (FN)	60 (TN)

Recall A:
70/100 = 0.7

$$recall = \frac{\#TP}{\#TP + \#FN}$$

What fraction of the ones that belong to the class have you identified?

Of all the messages that are actually spam, what fraction has the system labeled as spam?

Evaluation metrics

	Truth: A	Truth: B
Predicted: A	70 (TP)	40 (FP)
Predicted: B	30 (FN)	60 (TN)

Recall A:
70/100 = 0.7

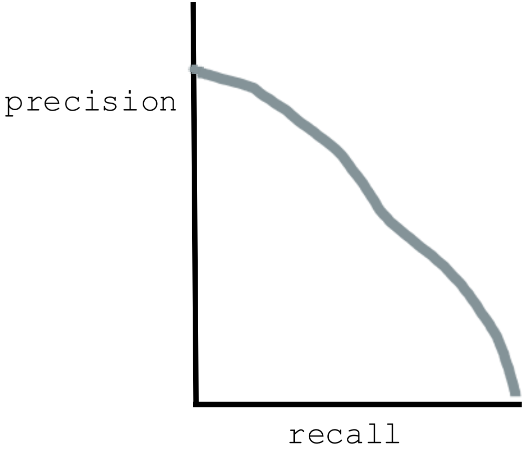
Recall B:
60/100 = 0.6

$$recall = \frac{\#TP}{\#TP + \#FN}$$

What fraction of the ones that belong to the class have you identified?

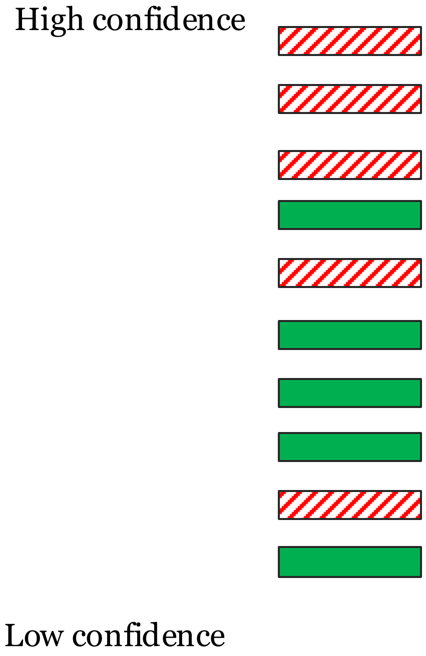
Of all the messages that are actually spam, what fraction has the system labeled as spam?

Precision recall curve

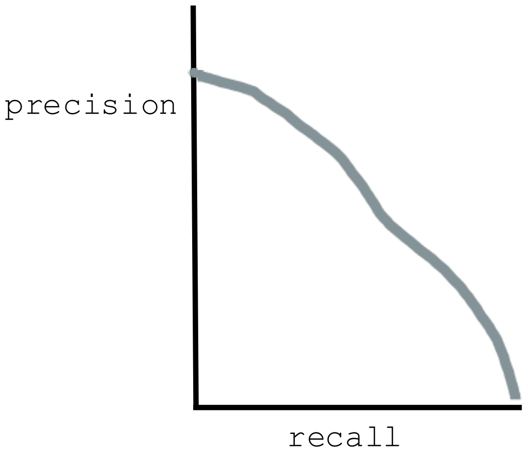


Spam classification
Accidentally labeling a message as spam: BAD
Accidentally labeling a spam message as ok: ANNOYING

Only label messages as spam if we're really *sure*.
Use the "confidence" of the classifier

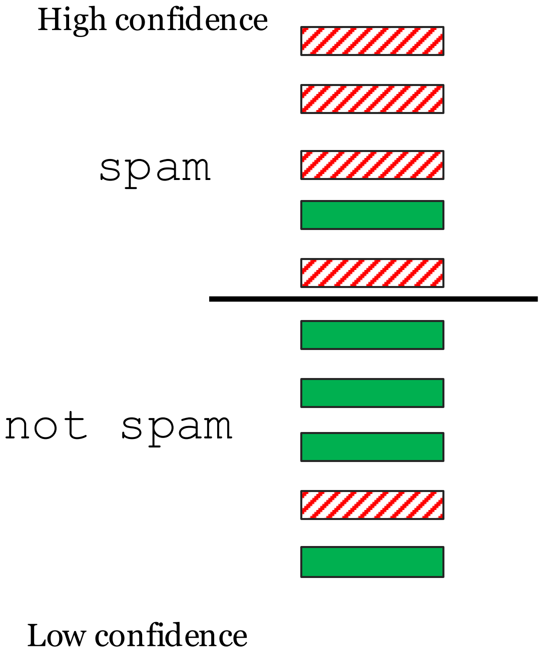


Precision recall curve



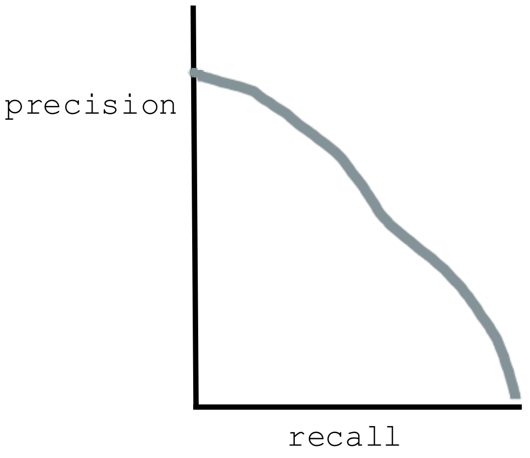
Spam classification
Accidentally labeling a message as spam: BAD
Accidentally labeling a spam message as ok: ANNOYING

Only label messages as spam if we're really *sure*.
Use the "confidence" of the classifier



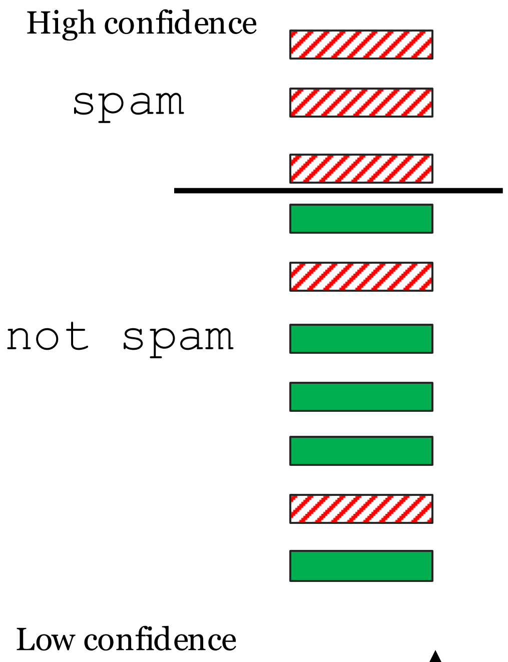
Precision = 4/5
Recall = 4/5

Precision recall curve



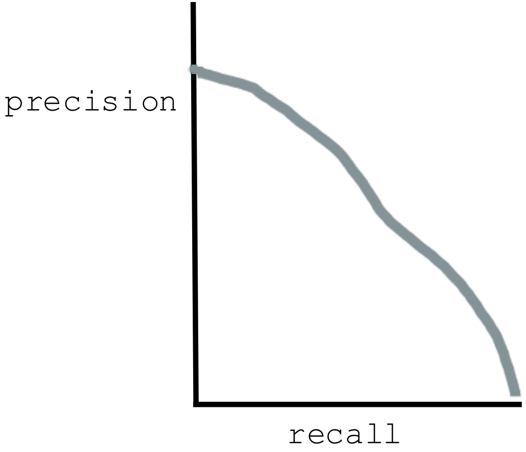
Spam classification
Accidentally labeling a message as spam: BAD
Accidentally labeling a spam message as ok: ANNOYING

Only label messages as spam if we're really *sure*.
Use the "confidence" of the classifier



Precision = 3/3 = 1 ↑
Recall = 3/5 ↓

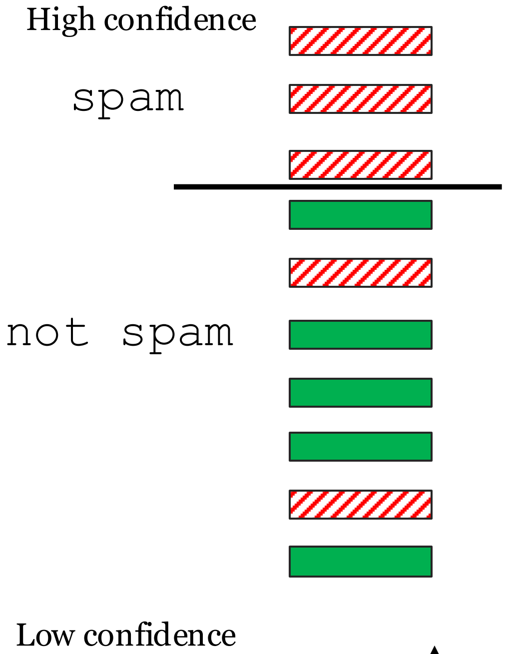
Precision recall curve



Spam classification
Accidentally labeling a message as spam: BAD
Accidentally labeling a spam message as ok: ANNOYING

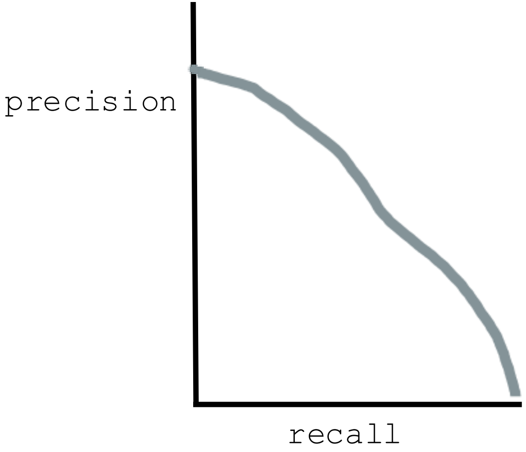
Only label messages as spam if we're really *sure*.
Use the "confidence" of the classifier

How can we compute the confidence of a decision tree?

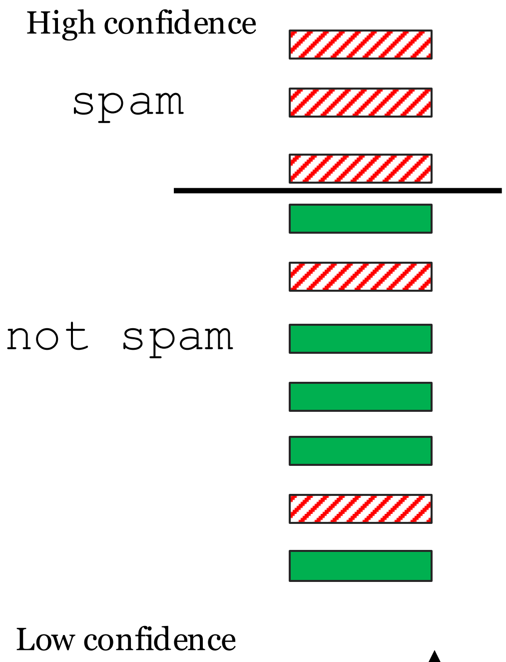


Precision = 3/3 = 1 ↑
Recall = 3/5 ↓

Precision recall curve



Question: Come up with a task for which *precision* is more important, and a task for which *recall* is more important



Precision = 3/3 = 1 ↑

Recall = 3/5 ↓

Evaluation

Combining recall and precision using F-measure

$$F_\beta = (1 + \beta^2)\frac{\text{precision } \times \text{recall}}{(\beta^2\text{precision}) + \text{recall}}$$

Often **β** = 1:

$$F_1 = \frac{2 \cdot \text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Multiclass classification

- Many classification tasks are *binary* (e.g. spam or not spam).
- But… often there are more than 2 classes. This is called **multiclass** classification.

IMAGENET

ImageNet has **21841** classes

Speech act classification
15 speech acts in the project dataset

F1 for multiclass problems

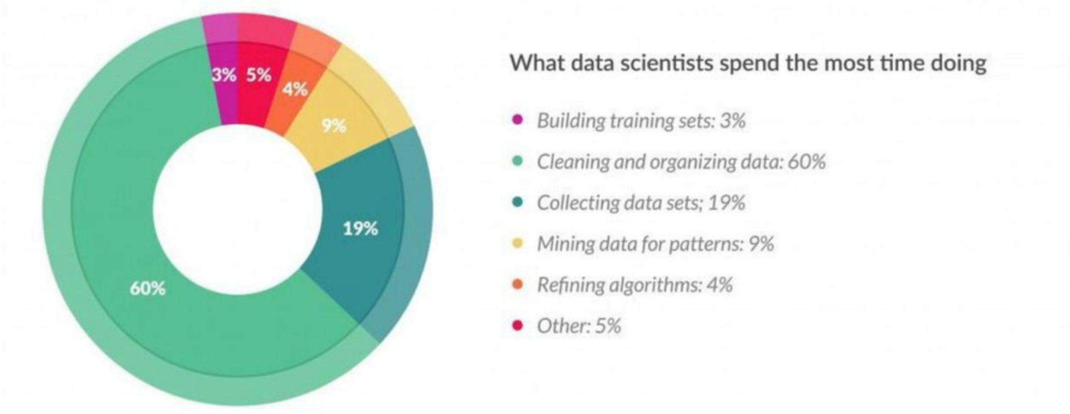
- F1 scores for individual classes
- In addition:
  - Micro F1 average: Calculate F1 by counting total nr of true positives, false negatives and false positives
  - Macro F1 average: Calculate metrics for each class, and aggregate by taking an (unweighted) average

ML process + wrap up

ML process

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

CIML, figure 2.4



Source: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

ML process

Decide *what and how* to collect data!

Some labels can be collected 'automatically'

- E.g. whether user clicked on an ad.

When annotating your own data:

- Develop annotation guidelines (sometimes called code book, especially in the social sciences).
- Calculate inter-annotator agreement and within-annotator agreement.
- If humans can't agree about the right label….

CIML, figure 2.4

ML process

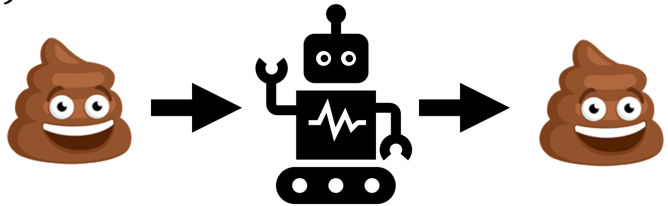
1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

CIML, figure 2.4

Decide *what and how* to collect data!

Be suspicious of 'ground truth' or 'gold labels'!

- Annotator noise
- Annotator bias
- Inherent ambiguity
- Some concepts are very hard to formalize! (hate speech detection)



garbage in, garbage out

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

ML process

Data pre-processing

- Raw data is usually messy (e.g., missing values, outliers, class imbalance)
- Data quality, amount and preparation is a key factor for the success of any ML solution

CIML, figure 2.4

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

ML process

Data pre-processing

Dealing with missing values

- Eliminate entries/attributes with missing values (only works if not too many elements are in this situation)
- Estimate missing values (e.g., interpolation, fill in the mean/median)

Inconsistent values

- Be aware of data types and ranges of the attributes (e.g., you cannot have negative values for 'Age')

CIML, figure 2.4

1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow², ± click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for ± click
12	deploy!	(hope we achieve our goal)

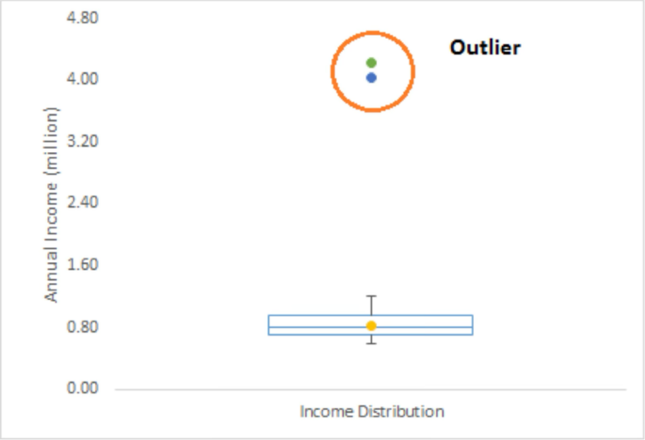
CIML, figure 2.4

ML process

Data pre-processing

Identify and remove outliers

- outliers = unexpected values that can statistically distort the dataset and negatively impact the performance of the ML model
- caused usually by measurement **noise, faulty input** or rare events



1	real world goal	increase revenue
2	real world mechanism	better ad display
3	learning problem	classify click-through
4	data collection	interaction w/ current system
5	collected data	query, ad, click
6	data representation	bow^2, $\pm$ click
7	select model family	decision trees, depth 20
8	select training data	subset from april'16
9	train model & hyperparams	final decision tree
10	predict on test data	subset from may'16
11	evaluate error	zero/one loss for $\pm$ click
12	deploy!	(hope we achieve our goal)

ML process

Usually a combination of multiple evaluation metrics.

Also take into account:

- Cost of errors (e.g. accidently labeling a spam e-mail as 'ok' vs. a self-driving car not detecting a pedestrian crossing a street)
- Biases
- Etc..

Before deciding to deploy it!

CIML, figure 2.4

Quiz

I posted **a short quiz (optional) on Brightspace** for you to practice with the material.

Do the quiz before **Wednesday 4pm**, so I have time to take a look before the next lecture.

What do you need to know

- Pros and cons of rule-based vs. supervised learning
- Differences between supervised, unsupervised, reinforcement learning
- Decision Trees (algorithm, entropy, information gain, error rate)
- Concepts such as decision boundary, overfitting, underfitting, inductive bias, hyperparameters
- How to set up machine learning experiments (cross validation, evaluation metrics, precision recall tradeoff)

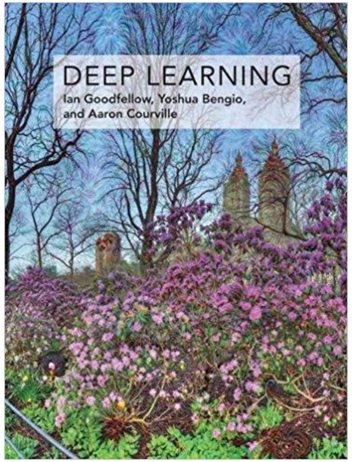
Resources

- **scikit-learn:** https://scikit-learn.org. Python library with many implementations for ML models (incl. decision trees), as well as pre processing and evaluation

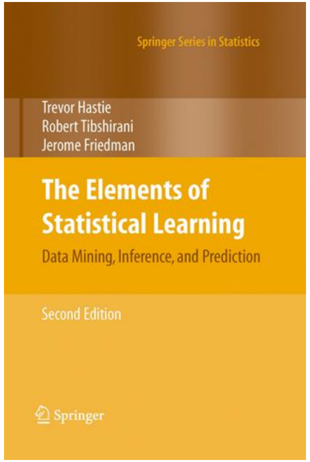
- **Kaggle:** https://www.kaggle.com/. Improve your ML skills by participating in competitions with shared datasets.

- There are many online tutorials and online courses (e.g. ML courses by Andrew Ng, Fast AI, etc.)

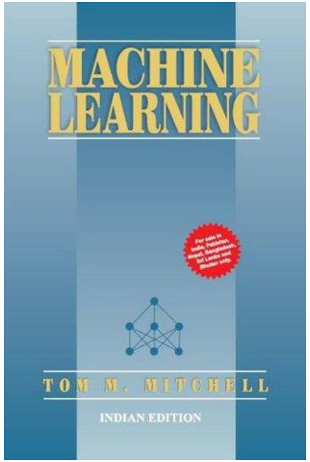
Books



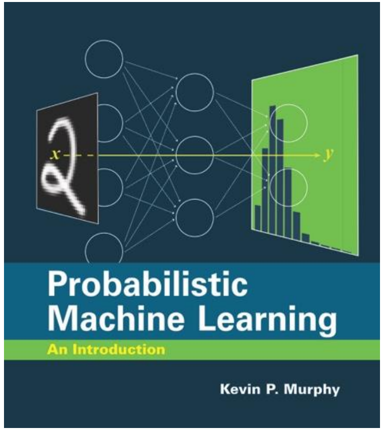
http://www.deeplearningbook.org/



https://web.stanford.edu/~hastie/ElemStatLearn/



https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf



https://probml.github.io/pml-book/book1.html