INTRODUCTION À L'APPRENTISSAGE AUTOMATIQUE

Lecture 1 - Polytech

Caio Corro

QUELQUES INFORMATIONS UTILES

Note

- Examen sur table (50%)
- TP/Projet avec rapport (50%)

Examen

- Sans document
- Portera sur les TDs + les contenus des TPs non notés

TP/Projet avec rapport

La note portera principalement sur votre capacité à démontrer que vous avez compris ce que vous avez fait dans le rapport.

Site internet

Tous les documents en lien avec le cours (slides, énoncés de TDs, de TPs, etc) seront disponibles sur mon site web : https://caio-corro.fr/

BACKGROUND & RÉFÉRENCES

Que faut-il maîtriser pour réussir dans ce cours ?

- Savoir calculer des dérivés partielles et des gradients
- Savoir coder un peu en Python
- Maîtriser un peu les thèmes suivants :
  - Algèbre linéaire
  - Probabilités
  - (et c'est tout ?)

Références

- Machine Learning: A First Course for Engineers and Scientists
  (Lindholm, Wahlström, Lindsten, Schön)
  http://smlbook.org/
- A course in Machine Learning
  (Hal Daumé III)
  http://ciml.info/

PLAN TRÈS APPROXIMATIF

Ce qu'on va étudier

- Tentative de définition de ce qu'est l'apprentissage automatique
- Les problèmes standards
  (régression, classification binaire et classification multiclasse)
- Quelques modèles
  - k-plus proches voisins
  - Arbres de décision
  - Modèles linéaires
- Les fonctions de perte
- Les questions de généralisation
- Peut être autre chose ?

TENTATIVE DE DÉFINITION DE CE QU'EST L'APPRENTISSAGE AUTOMATIQUE

Intelligence artificielle

Méthodes informatiques pour répliquer certaines capacités humaines (?)

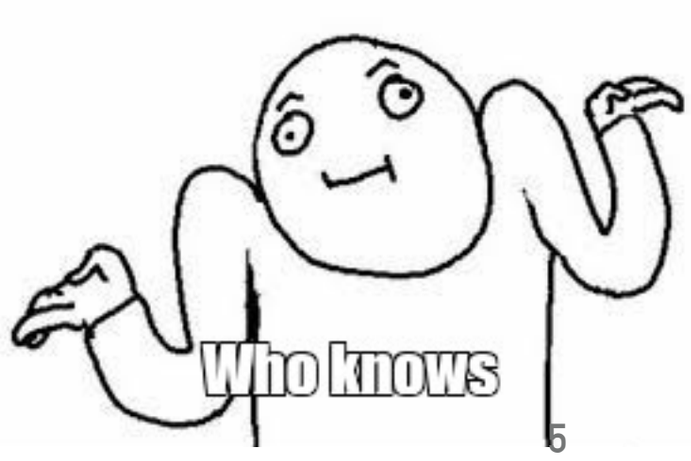
Apprentissage automatique

Approche pour l'intelligence artificielle fondée sur "l'apprentissage" de modèles à partir de données ou de résultats d'expérience (?)

Définitions satisfaisantes ?

- Je ne suis pas épistémologue
- On peut résoudre des problèmes extrêmement difficiles pour des êtres humains avec de l'apprentissage automatique

A-t-on vraiment besoin d'une bonne définition ?



TENTATIVE DE DÉFINITION DE CE QU'EST L'APPRENTISSAGE AUTOMATIQUE

SQL parsing

- Entrée: une phrase en langue naturelle
- Sortie: requête SQL

```
I want to book a flight from Paris to Rome.
```

⇓

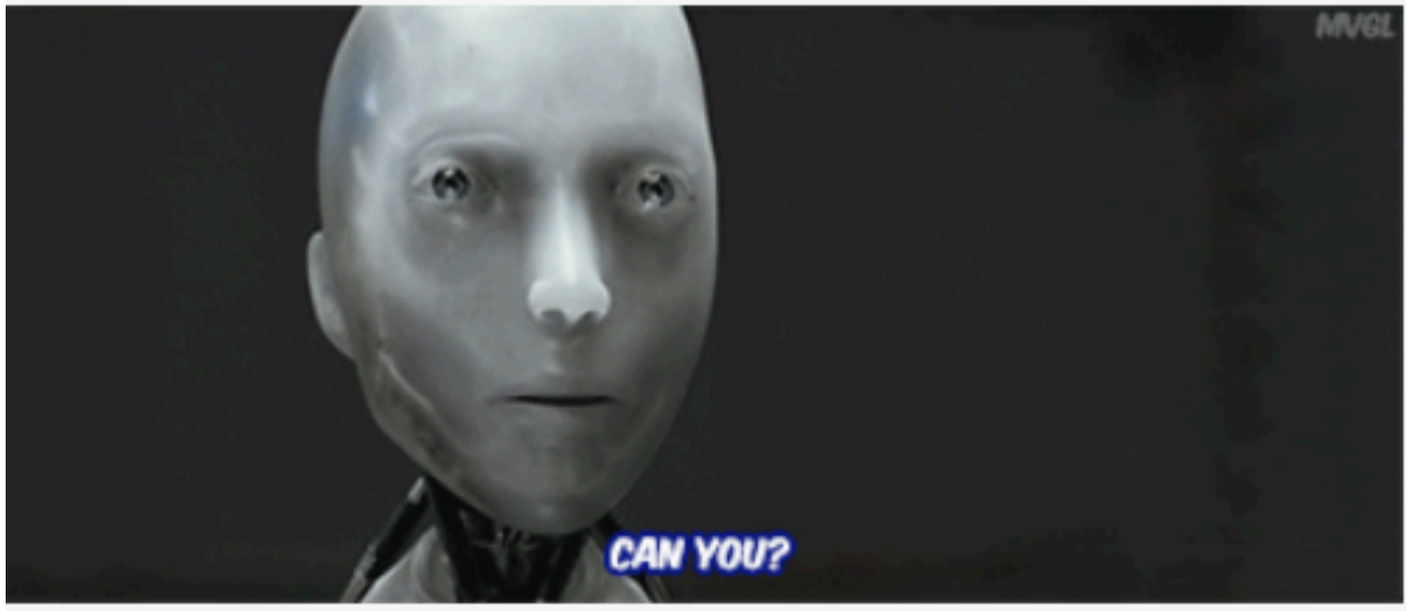
```
SELECT * FROM  flight WHERE  from = "paris" AND to = "rome"
```

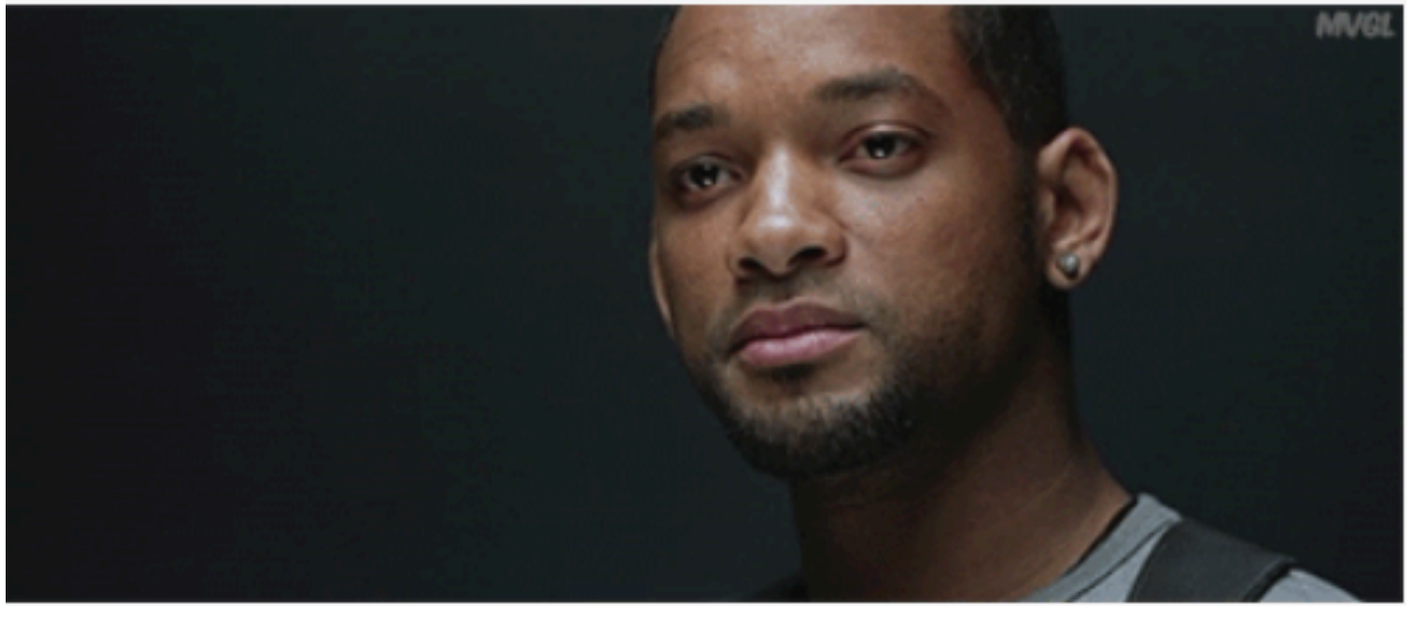
Intelligence artificielle

Méthodes informatiques pour répliquer des capacités humaines (?)

- Écrire des requêtes SQL, est-ce vraiment quelque chose de caractéristique d'une capacité humaine ? Qui fait ça ?
- Pourtant c'est un problème extrêmement important en apprentissage automatique avec beaucoup d'applications industrielles !


MVGL
CAN A ROBOT WRITE A SYMPHONY?
CAN A ROBOT TURN A CANVAS INTO A BEAUTIFUL MASTERPIECE?


MVGL
CAN YOU?


MVGL

TENTATIVE DE DÉFINITION DE CE QU'EST L'APPRENTISSAGE AUTOMATIQUE

Problème de définition assez courant en sciences

- Définir la sciences, les sciences et parfois même ce qu'est la pratique d'une science est extrêmement compliqué
- Toutes les définitions "simplistes" peuvent facilement être mises à mal
- Il y a des cours de philosophie pour ça, pas l'objet du cours

TENTATIVE DE DÉFINITION DE CE QU'EST L'APPRENTISSAGE AUTOMATIQUE

Modèle de prédiction

Modèle qui fait une prédiction sur une entrée

Problèmes intéressants pour des modèles de prédiction

Comment écrire un modèle capable de reconnaitre si une image contient un chat ?

- Intuitivement on pourrait arriver à avoir une idée
- Impossible à formaliser et à implémenter sous forme d'un algorithme

On veut s'intéresser à ce genre de problèmes.

TENTATIVE DE DÉFINITION DE CE QU'EST L'APPRENTISSAGE AUTOMATIQUE

L'apprentissage automatique en pratique

Un algorithme d'apprentissage automatique est un algorithme :

- qui prends en entrées :
  - des données ou un environnement
  - des hyper-paramètres (i.e. valeurs qui configurent l'algorithme)
- qui donne en sortie un modèle de prédiction

Pleins de documents

num_iter=100,
num_layers=10,
seed=1864568153,
init_distrib=normal,
...

Algorithme d'apprentissage automatique

Modèle de prédiction

GÉNÉRALISATION

Données d'apprentissage
(ou d'entrainement)

Pleins de documents

num_iter=100,
num_layers=10,
seed=1864568153,
init_distrib=normal,
...

Algorithme d'apprentissage automatique

Modèle de prédiction

Le problème de la généralisation

On veut que notre modèle marche sur des données qu'on n'a pas vu à l'apprentissage
=> généralisation (il faut généraliser à de nouvelles données)

≠ de la compression (?)

EXAMPLE : URL EXTRACTION

Problem

Task: given a large corpus a text files, **count the number of different URLs**.

How to do you do that?

Easy solution

1. Find a way to extract URLs:
   - ➤ regular expression?
   - ➤ library?
2. Collect all URLs and unify them
3. Count

Downside

- ➤ Does the regex/library actually extract **all** URLs?
- ➤ How do you know if the answer is correct?

URL REGEX

https://gist.github.com/gruber/249502

```
(?xi)
\b
(                                   # Capture 1: entire matched URL
  (?:
    [a-z][\w-]+:                    # URL protocol and colon
    (?:
      /{1,3}                        # 1-3 slashes
      |                             #   or
      [a-z0-9%]                     # Single letter or digit or '%'
                                    # (Trying not to match e.g. "URI::Escape")
    )
    |                               #   or
    www\d{0,3}[.]                   # "www.", "www1.", "www2." … "www999."
    |                               #   or
    [a-z0-9.\-]+[.][a-z]{2,4}/      # looks like domain name followed by a slash
  )
  (?:                               # One or more:
    [^\s()<>]+                      # Run of non-space, non-()<>
    |                               #   or
    \(([^\s()<>]+|(\([^\s()<>]+\)))*\) # balanced parens, up to 2 levels
  )+
  (?:                               # End with:
    \(([^\s()<>]+|(\([^\s()<>]+\)))*\) # balanced parens, up to 2 levels
    |                               #   or
    [^\s`!()\[\]{};:'".,<>?«»“”‘’]  # not a space or one of these punct char
  )
)
```

EXAMPLE 2: NAMED ENTITY RECOGNITION

Problem

Task: given a large corpus a text files, extract all named entities:

- Person names
- Place names
- Dates
- Locations
- ...

After Sherborne, Turing studied as an undergraduate from 1931 to 1934 at King's College, Cambridge, where he was awarded first-class honours in mathematics. In 1935, at the age of 22, he was elected a fellow of King's on the strength of a dissertation in which he proved the central limit theorem. Unknown to the committee, the theorem had already been proven, in 1922, by Jarl Waldemar Lindeberg. A blue plaque at the college was unveiled on the centenary of his birth on 23 June 2012 and is now installed at the college's Keynes Building on King's Parade

Potential tags:
LOCATION
ORGANIZATION
DATE
MONEY
PERSON
PERCENT
TIME

Extract of https://en.wikipedia.org/wiki/Alan_Turing tagged with http://nlp.stanford.edu:8080/ner/

EXAMPLE 2: NAMED ENTITY RECOGNITION

Easy solution

Use a gazetteer:

Class	Gazetteer excerpt
Locations	sunderland, sundsvall, sundsvl, sunnyvale, sunrise, sunset beach, sunshine coast, suourland, suouroy
Organizations	xyratex, y e data, ya, yadkin, yahoo, yahoo maps, yahoo personal email, yahoo personals, yahoo sbc
Person Names	jerrine, jerrod, jerrol duane, jerrold, jerrome, jerry, jerry ben, jerry brown, jerry claude, jerry coleman
Other	january, february, sunday, monday, toyota prius, ford focus, apple ipod, microsoft zune, tylenol, advil

Gazetteer from [Carlson et al., 2009]

Downsides

- Language is ambiguous, e.g. « Tim Cook »
- Language is evolving, e.g. new people, new places, …
- Typos, « non-standard » writing (e.g. tweets)

In general for natural language processing

- More or less intuitive for a human who speaks a given language
- Fuzzy decision, lots of « weak » contradictory signals

Even for syntactic analysis!

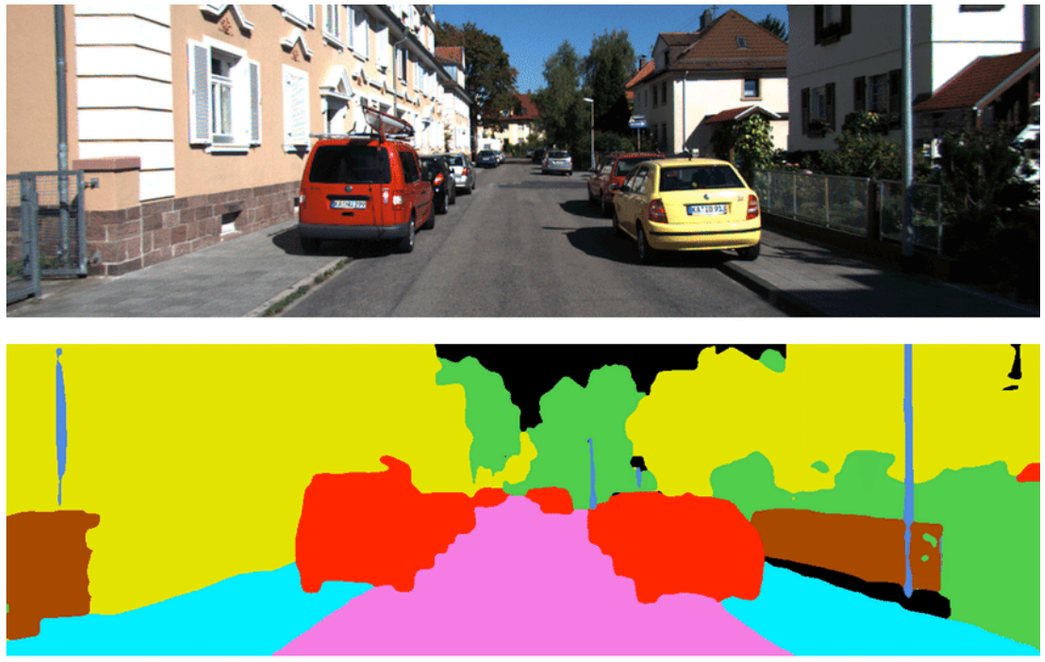
*« The spokesperson, **Μιχάλης Χατζόπουλος**, has declared that… »*

EXAMPLE 3: COMPUTER VISION

Which one of these animals is a cat?

- Obvious!
- How do you know?
  Fuzzy decision, several criteria,…
- Can you build an algorithm out of your intuitions?

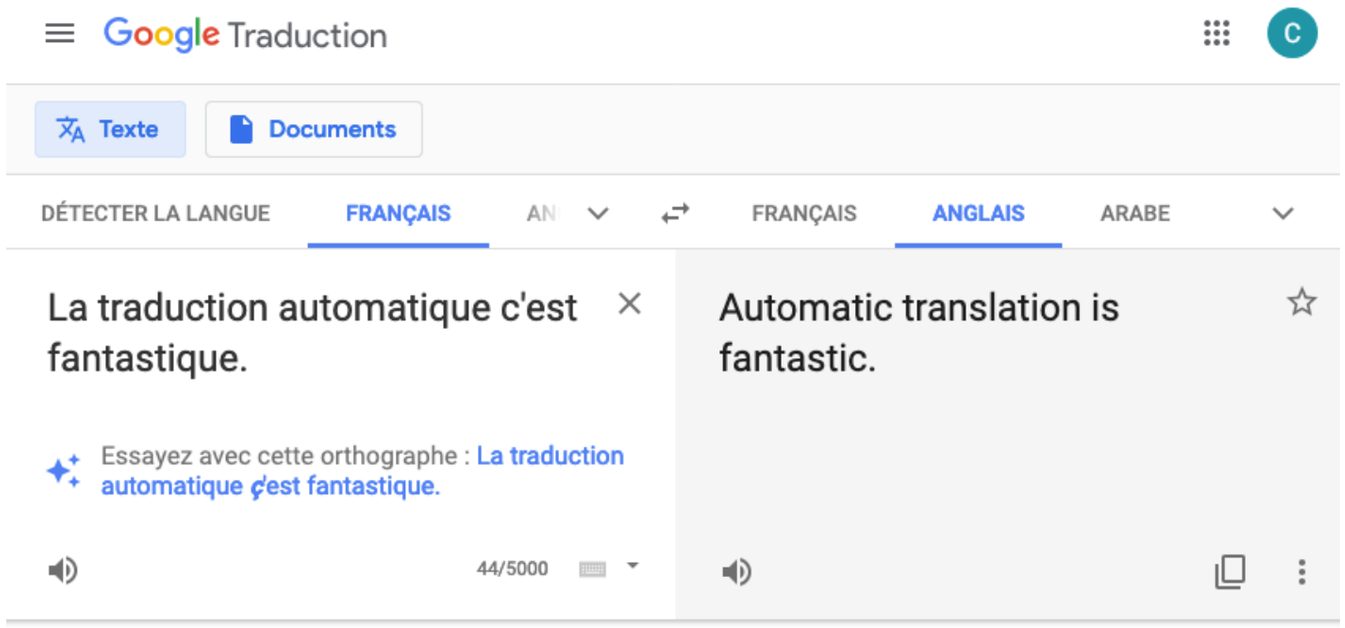
EXEMPLES DE PROBLÈME



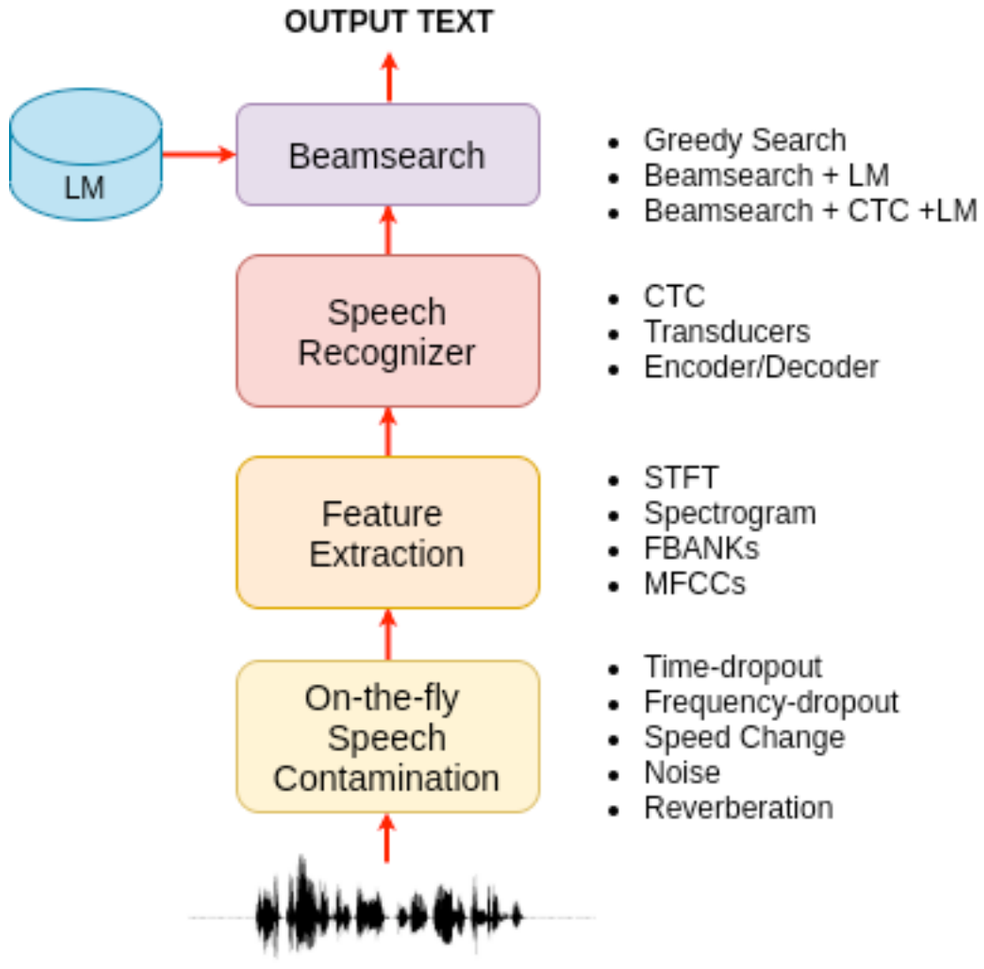
https://www.researchgate.net/figure/Example-of-2D-semantic-segmentation-Top-input-image-Bottom-prediction_fig3_326875064

Self-driving car

EXEMPLES DE PROBLÈME



Machine translation, Google



Speech recognition (picture from Speechbrain)

SENTENCE CLASSIFICATION

Sentiment analysis

- Input: sentence
- Output: Positive? Neutral? Negative?

Inputs are of different length!

```
This movie is great!
I saw that movie some years ago.
The food was disgusting…
```

=> The "true" answer can be ambiguous, even for a human it can be difficult! (irony, ...)

Natural language inference

- Input: premise and hypothesis (2 sentences)
- Output: Entailment? Neutral? Contradiction?

```
John likes Baltimore a lot.
Johne likes Baltimore.
```

TAGGING

Part-of-speech tagging

- Input: sentence
- Output: grammatical category for each word of the sentence

PRP	VB	DET	NN
They	walk	the	dog

Named entity recognition with BIO tags

- Input: sentence
- Output: BIO tags + 6 classes
  (person, location, group, creative work, product, corporation)

B-Per	I-Per	O	O	B-Loc
Neil	Armstrong	visited	the	moon

CHUNKING

Named entity recognition

- Input: sentence
- Output: chunks (person, location, group, creative work, product, corporation)

Nested named entity recognition

- Input: sentence
- Output: nested chunks

SEQUENCE GENERATION

Machine translation

- Input: sentence in a source language
- Output: sentence in a target language

`They walk the dog` ⇒ `Ils promènent le chien`

« Elles » ?

Image captioning

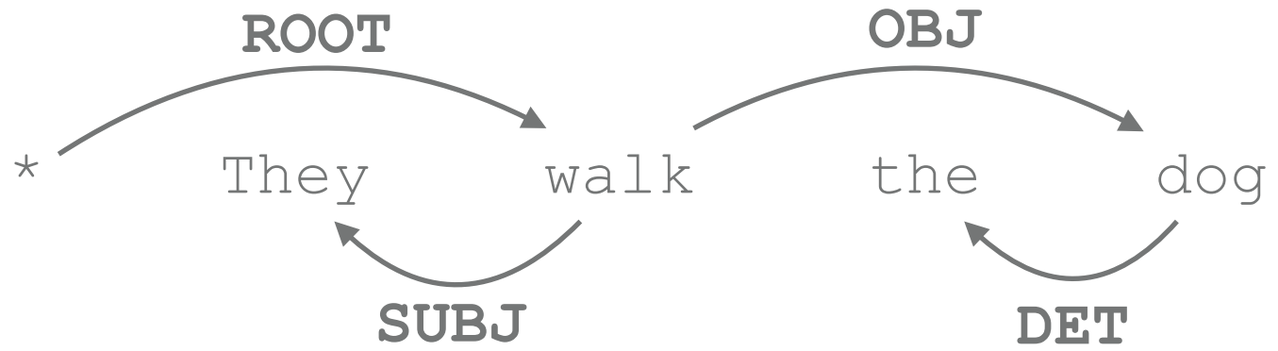
- Input: image
- Output: sentence describing the image

⇒ `Spongebob is cooking burgers`

SYNTACTIC PARSING

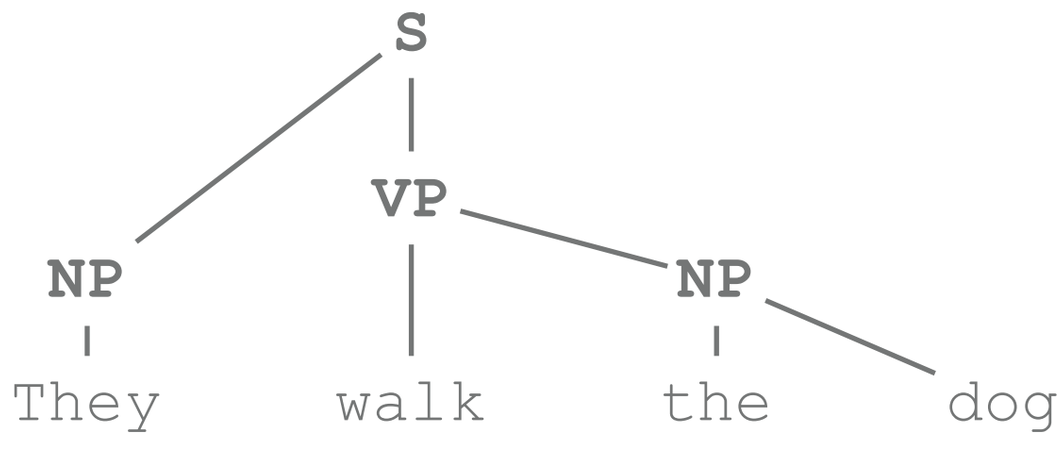
Syntactic dependency parsing

- Input: sentence
- Output: bi-lexical dependencies between words



Constituency parsing

- Input: sentence
- Output: hierarchical phrase-structure



SEMANTIC PARSING

SQL parsing

- Input: sentence
- Output: SQL query

```
I want to book a flight from Paris to Rome.
```

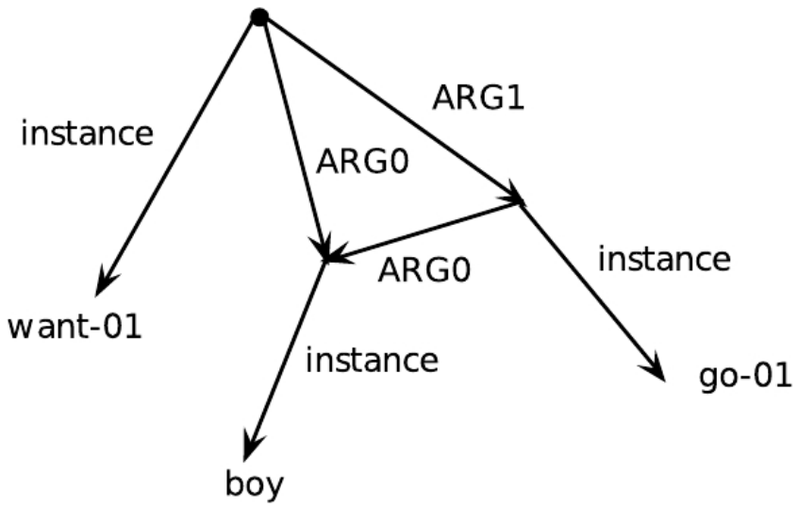
⇓

```
SELECT * FROM  flight WHERE  from = "paris" AND to = "rome"
```

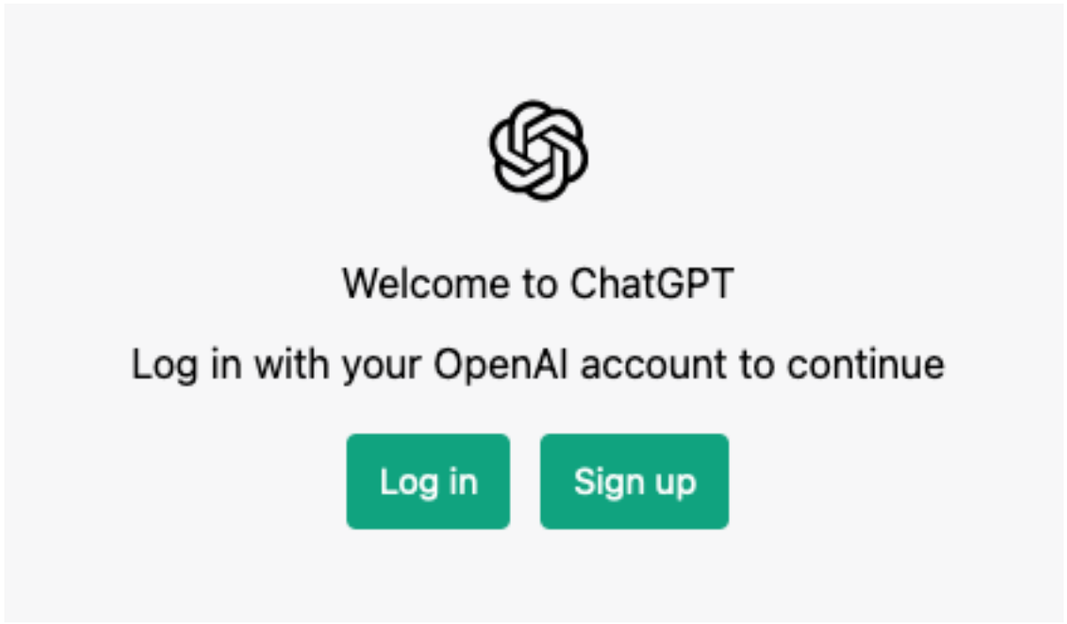
Abstract Meaning Representation (AMR) parsing

- Input: sentence
- Output: graph

The boy want to go. ⇒



EXEMPLE : LANGUAGE MODELS



Les modèles de langue comme ChatGPT

SIZE OF LANGUAGE MODELS

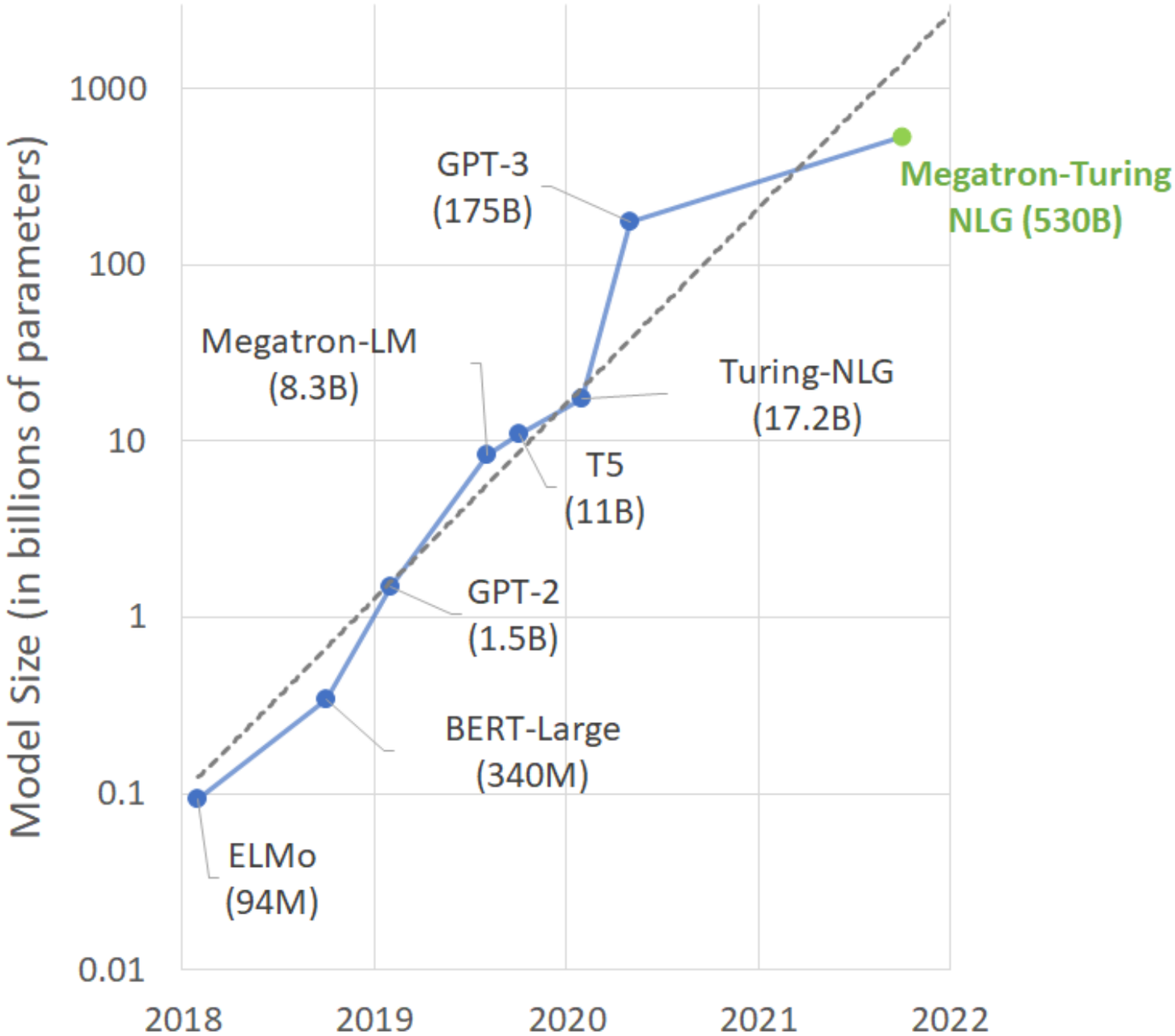
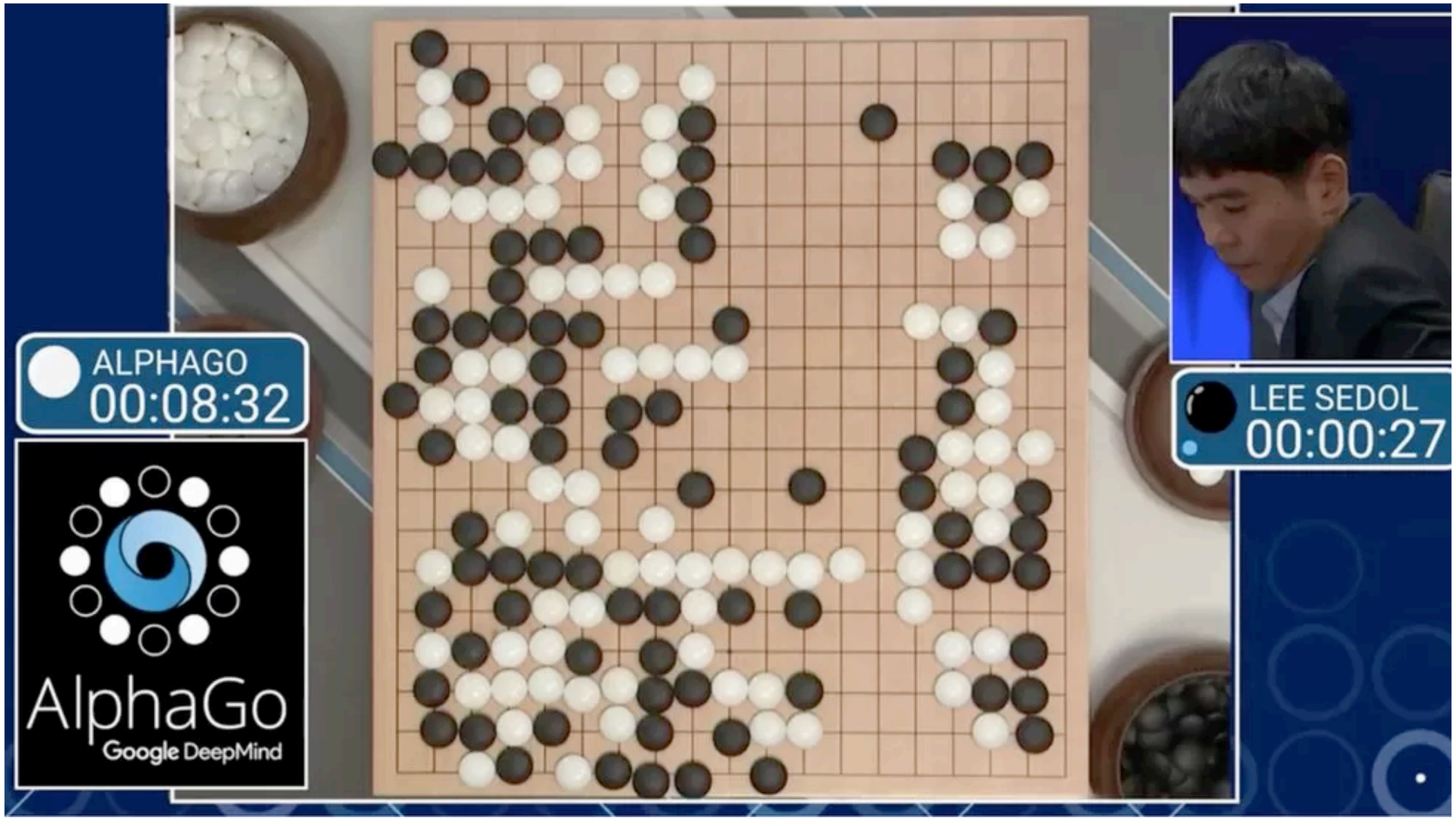
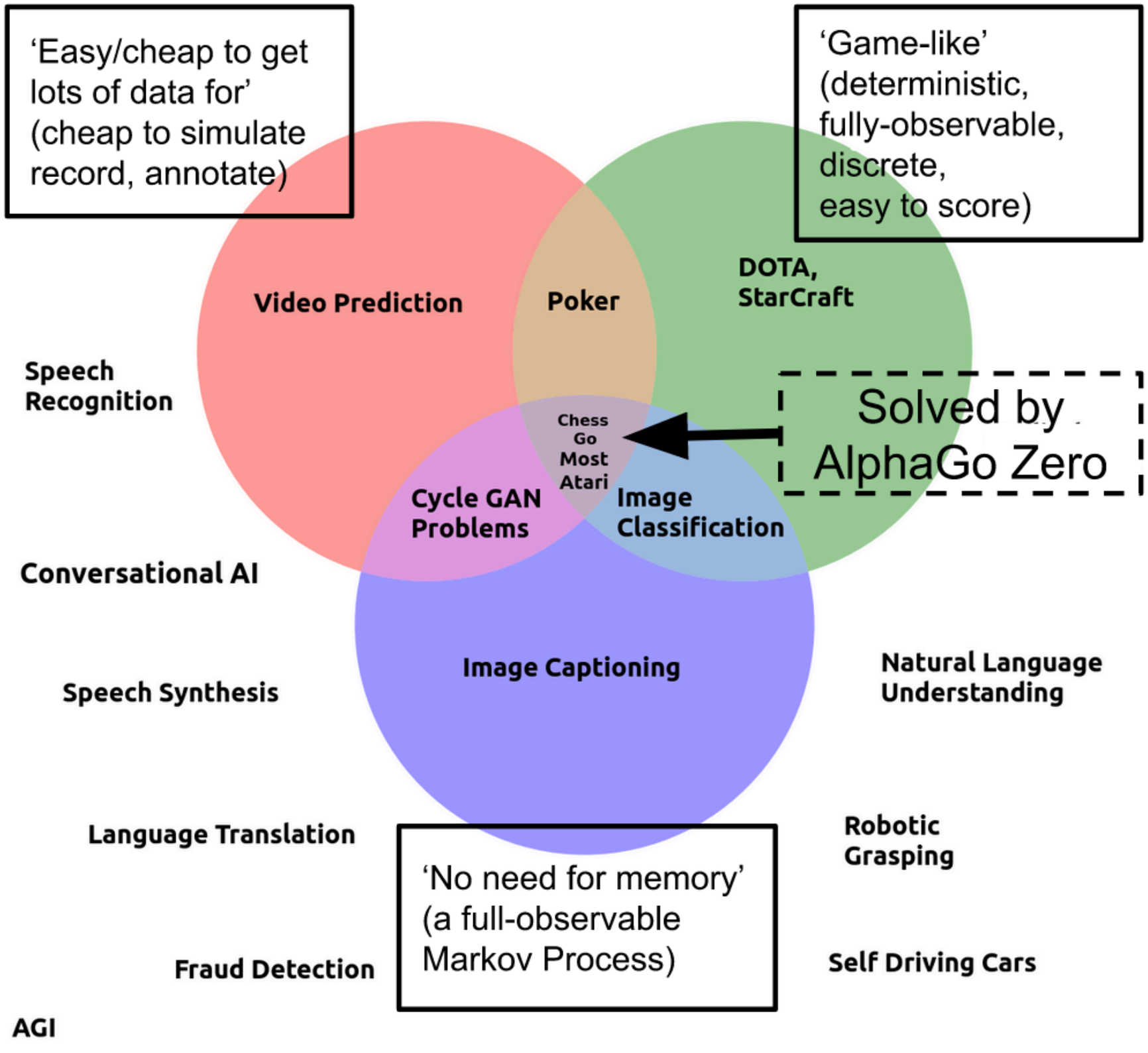


Figure from: https://developer.nvidia.com/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/

EXEMPLE



A (rough) Venn Diagram of AI Problem Complexity



https://www.skynettoday.com/editorials/is-alphago-zero-overrated

EXAMPLE FROM PHYSICS 1/2

Map of mass distribution in the universe

If we can learn a generative model of these maps, we can inspect its parameters to hope to better understand the universe

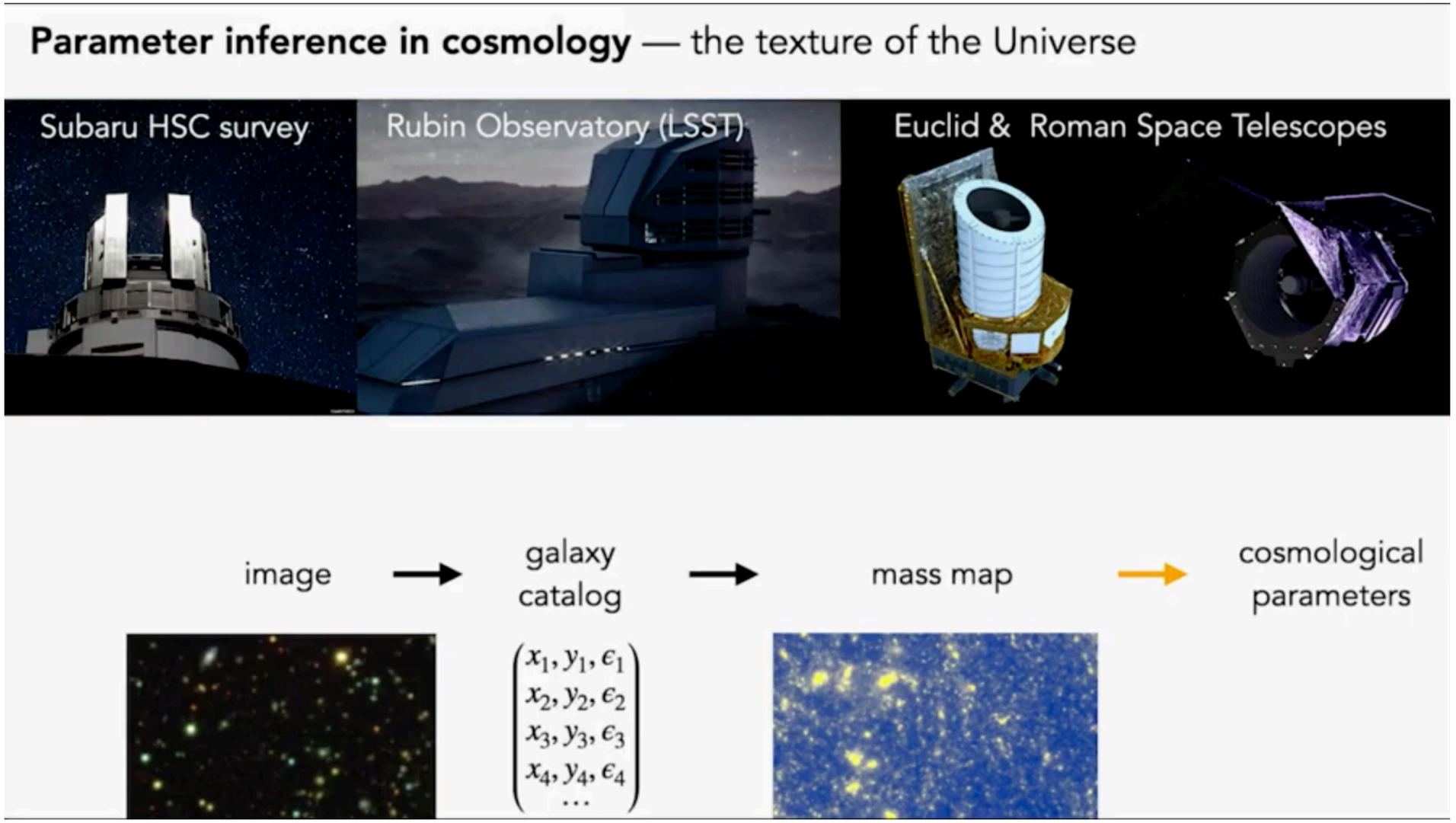
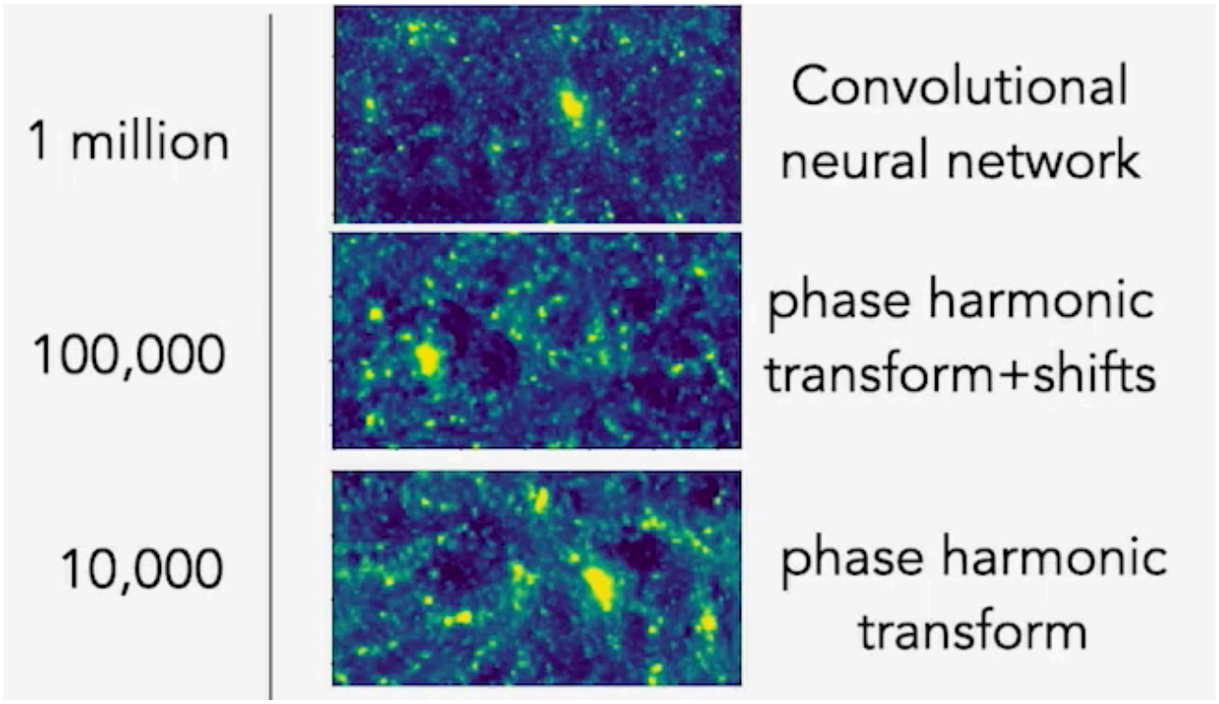


Image by Brice Ménard, seminar at Collège de France https://www.youtube.com/watch?v=xGM5Fv6kAoU

EXAMPLE FROM PHYSICS 2/2

Modeling via machine learning

- CNNs : can generate very good pictures, but too many parameters
- Scattering Transform: architectures with very good mathematical foundations, can generate good maps with way fewer parameters than CNNs



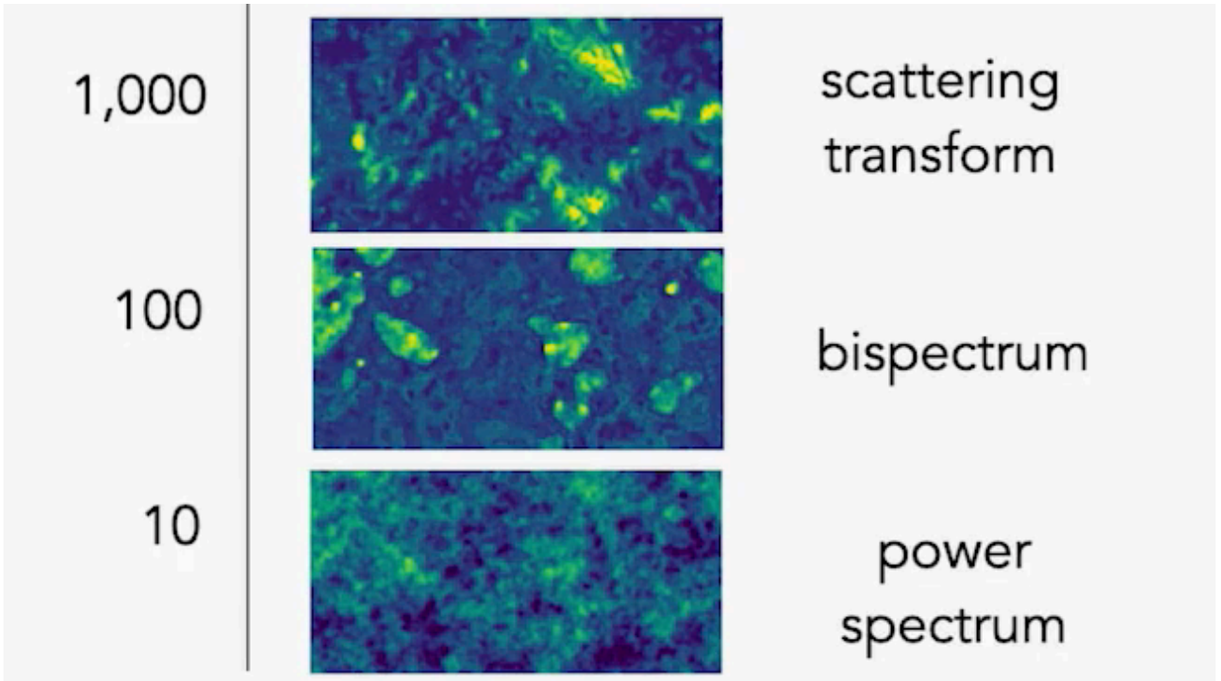


Image by Brice Ménard, seminar at Collège de France https://www.youtube.com/watch?v=xGM5Fv6kAoU

Scattering transform:

- Group invariant scattering (Mallat, 2012) https://arxiv.org/abs/1101.2286
- How to quantify fields or textures? A guide to the scattering transform (Sihao and Ménard, 2021) https://arxiv.org/abs/2112.01288

TYPOLOGIE DES PROBLÈMES ET MODÈLES D'APPRENTISSAGE AUTOMATIQUE

TYPES DE PROBLÈME

Notations

- $\mathbf{x} \in \mathbb{R}^d$: entrée d'un modèle de prédiction, vecteurs de valeurs
- $\mathbf{y} \in Y$: sortie d'un modèle de prédiction

Exemples

- Régression : $Y = \mathbb{R}$
- Classification binaire : $Y = \{0,1\}$ ou $Y = \{-1,1\}$
- Classification multiclasse : $Y = \{1,2,...,k\}$ ou $Y = E(k)$
- Classification multilabel (possibilité de prédire plusieurs classes pour une entrée)
- Prédiction structurée
- ...

Base canonique de dim. k (ou ensemble des vecteurs one-hot de dim. k)

TYPES D'APPRENTISSAGE

Apprentissage supervisé

Accès à un ensemble de couples entrées/sorties de forme $(\mathbf{x}, y) \in D$
pour apprendre le modèle, où D dénote l'ensemble des données d'entrainement.
Un couple $(\mathbf{x}, y) \in D$ est un point de donnée annoté avec une annotation de référence.

Apprentissage semi-supervisé

Accès à un ensemble de données annotées $(\mathbf{x}, y) \in D$
ainsi qu'un ensemble de données supplémentaires non annotés $\mathbf{x} \in D'$

Apprentissage faiblement supervisé

Accès à un ensemble de données non annotées $\mathbf{x} \in D'$
ainsi que de connaissance sur la tâche

TYPES D'APPRENTISSAGE

Apprentissage non supervisé

Simplement accès à des données non annotés

En général, dans ce cas on ne cherche pas à prédire une sortie, mais plutôt à apprendre un modèle des données, i.e. un modèle capable de dire si un nouveau point a été généré de la même distribution que les données vues à l'entraînement.

Apprentissage par renforcement

Accès à un environnement dans lequel un modèle peu faire des actions puis recevoir une récompense (e.g. l'environnement est un jeu vidéo, le modèle prédit l'action suivante pour un joueur)

TYPES DE MODÈLE

Modèle paramétrique vs. modèle non paramétrique

- Paramétrique : la "taille" (nombre de paramètres) du modèle ne dépends pas des données d'apprentissage
- Non-paramétrique : le nombre de paramètres (voir même la structure de ceux-ci) dépends des données d'apprentissage

Modèle discriminant vs. modèle génératif

- Discriminant : modèle simplement capable de faire une prédiction
- Génératif : modèle capable de générer de nouvelles données et/ou d'estimer si un point vient de la même distribution que les données d'entraînement

$$p_\theta(y \mid x) \qquad p_\theta(x, y)$$

Distribution modélisée par un modèle discriminant

Distribution modélisée par un modèle génératif

TYPES D'ENTRÉE

Données

Nos données seront (la plupart du temps) des vecteurs de valeurs

$$\mathbf{x} = \begin{bmatrix} 1.4 \\ 3 \\ -13.3 \\ \dots \end{bmatrix}$$

Chaque élément du vecteur correspond à une caractéristique / feature
(l'ordre des features doit toujours être le même !)

Variables numériques et variables catégoriques

- numérique : notion d'ordre (i.e. $3 < 4$)
- catégorique : pas de notion d'ordre (maison ??? route)

Variable type	Example	Handled as
Number (continuous)	32.23 km/h, 12.50 km/h, 42.85 km/h	Numerical
Number (discrete) with natural ordering	0 children, 1 child, 2 children	Numerical
Number (discrete) without natural ordering	1 = Sweden, 2 = Denmark, 3 = Norway	Categorical
Text string	Hello, Goodbye, Welcome	Categorical

(Lindholm et al.)

VECTORS

Vector of features

- We represent features of the input as a **fixed size vector**
- Each cell is the value of a feature

Feature extractor

$$x = \begin{bmatrix} 1 \\ 0 \\ 0.34 \\ -5 \end{bmatrix}$$

No semantic, just values!

Number of features

- Low dim: 20
- Medium dim: 1000
- High dim: 100000

DISTANCE

Comparing inputs

Inputs can be compared by measuring a distance between their input vectors:

- Euclidean distance
- Radian distance
- Manhattan distance
- Non-euclidean geometry: Poincaré disk, …

Sometimes we can use functions that do not formally define a distance:

- Cosinus/sinus

Euclidean distance

$$dist(x, x') = \sqrt{\sum_i (x_i - x'_i)^2}$$

CLASS FEATURES

Example:

- What is the first word in the sentence?
- Is the dominant color blue, red or yellow?

Integer encoding:

- The=0, A=1, An=2, I=3, …
- blue=0, red=1, yellow=2

No! Problematic w.r.t. to distance measure!

One-hot vector

- Binary vector
- One at the index of the selected feature value
- Concatenate the other feature values

$$x = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{matrix} The \\ A \\ An \\ I \end{matrix}$$

GEOMETRIC INTERPRETATION

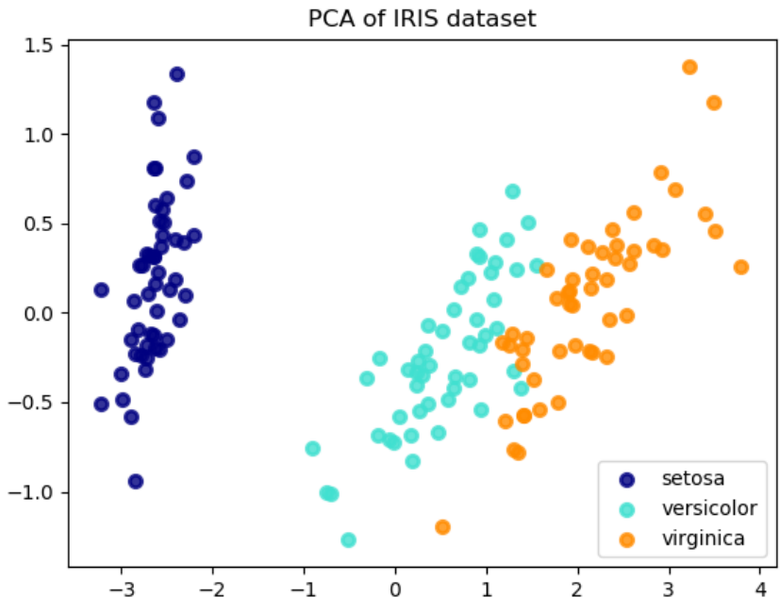
Feature analysis

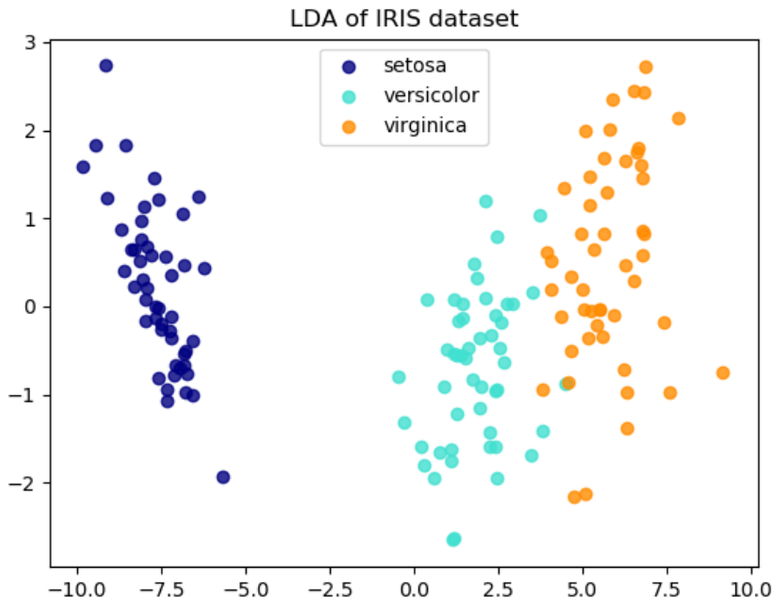
- Transform feature space (scale, truncate, …)
- Is distance meaningful? (Let's hope it is the case…)

Projection

It is often useful to project the feature space into 2D:

- Principal Component Analysis (PCA)
- Linear Discriminant Analysis (LDA)





https://scikit-learn.org/stable/auto_examples/decomposition/plot_pca_vs_lda.html

TEXT CLASSIFICATION

Task

Predict if a movie review is good or bad.

« This movie is great! » *« Worst movie ever… »*

Features

1. Uni-gram features:
   - ➤ One feature per word in the vocabulary
   - ➤ The feature is equal to one if the word is in the review
2. Bi-gram features:
   - ➤ One feature per couple of words in the vocabulary
   - ➤ The feature is equal to one if the words appear consecutively in the review

FEATURE SET

Let V be the vocabulary.

Unigram features

- Number of features: |V|
- We can use a dictionary to map a word to an index in a vector

$$x = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ \dots \\ 0 \end{bmatrix} \begin{matrix} bad \\ movie \\ great \\ worst \\ \\ house \end{matrix}$$

Bigram features

- Number of features: |V|x|V|
- Similarly, use a dict!

(vector should also include all other unigram features!)

$$x = \begin{bmatrix} 1 \\ 0 \\ \dots \\ 1 \end{bmatrix} \begin{matrix} This\ movie \\ The\ movie \\ \\ is\ great \end{matrix}$$

Problem

- The feature vector is sparse
- The dot product can be unnecessarily expensive!
- There is no inductive bias: a sentence is not bag-of-words, it is a sequence!

IMAGE CLASSIFICATION

Problem

Does the image contain a fire truck?



Which features?

- Pixel value at each position — Curse of dimensionality!
- Number of red pixels — Position invariant!

Central idea

- Design features that are central for the task
- We need the « reduce the dimension » of examples

THE PROBLEM OF FEATURE EXTRACTION

A lot of effort has been put into designing « good » features

- Images: Scale-invariant feature transform (SIFT features)
- Sounds: Mel-frequency cepstral coefficients (MFCC features)
- Text: list of relevant words, grammars

What if a classifier does not work?

- Is it because of the classification algorithm?
- Is it because of bad features?

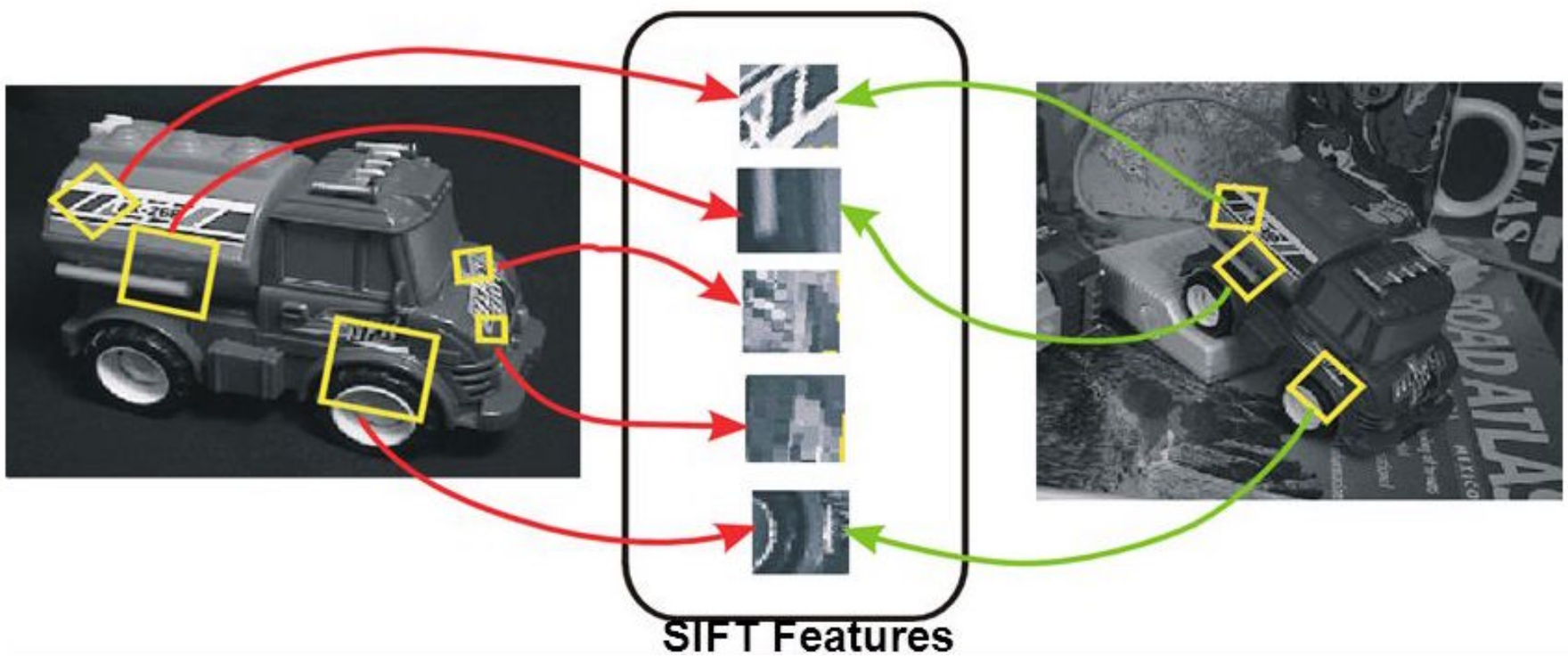
SCALE-INVARIANT FEATURE TRANSFORM (SIFT)

Main idea

Compute image features that are:

- Translation-invariant
- Scale-invariant
- Rotation-invariant
- Illumination-invariant
- Robust to noise

If a picture contains similar objects, it will contains similar features!



DECISION TREES

Based on the course of Hal Daumé III

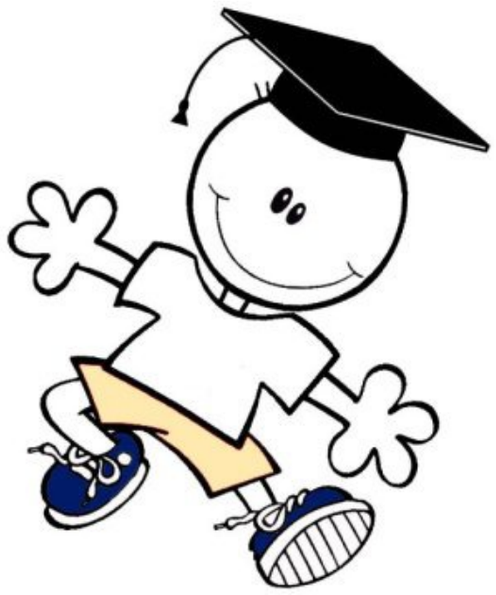
EXAMPLE 1/3

Task: recommandation system

Predict if a student is going to like a lecture (e.g. Machine Learning) or not

(Training) data

- Set of lectures and set of students
- Each student has taken and evaluated (a subset of) the lectures



EXAMPLE 2/3

Data

Each student rated the course (s)he took:

- Yes: liked the course
- No: disliked the course

Data can be sparse: students may not rate all the course

	Algorithms	OOP	Machine Learning	Graphs
Maryam	yes	no	no	yes
Nadi	no	yes	yes	no
Diaa	yes	no	no	yes
Oana	yes	no	yes	no
		...		

Generalization

No learning

- Predict a if a student will like a course that (s)he evaluated:
  Trivial, look at the data (memoization)

Need to learn something from the data

- Predict if a student will like a course the (s)he did not evaluate:
  Need to **generalize** from the student and other students evaluations!

EXAMPLE 3/3

Prediction by asking a series of binary questions

- **Model:** Is the course a machine learning course?
- **User:** Yes
- **Model:** Has this student taken any other machine learning courses?
- **User:** Yes
- **Model:** Has this student liked most previous machine learning courses?
- **User:** No
- **Model:** **I predict this student will not like this course.**

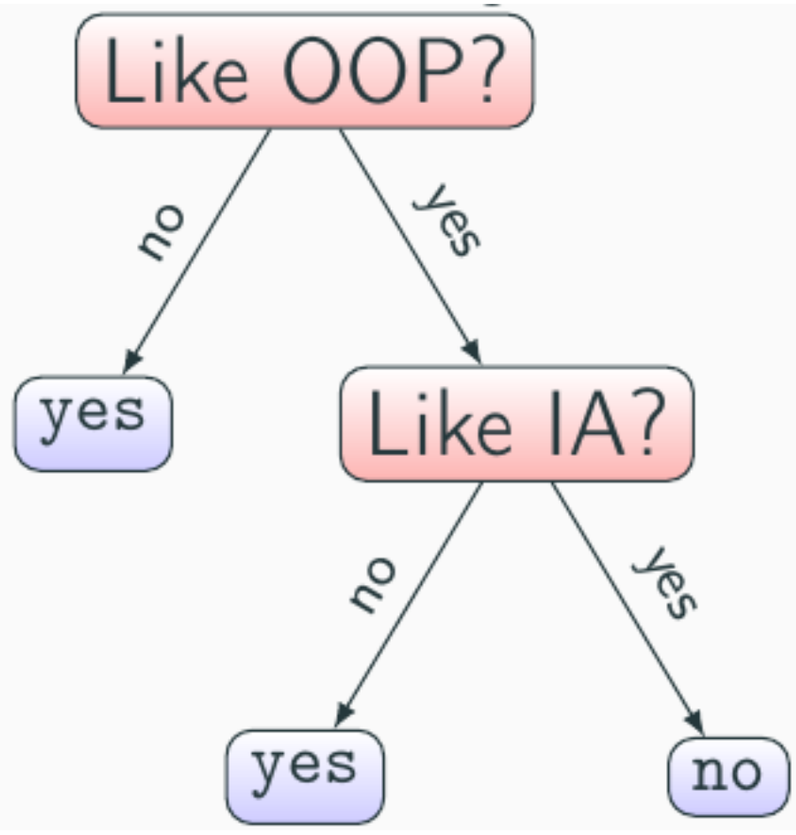
Learning a decision tree

- Which question to ask?
- In which order?
- Which answer?

DECISION TREE

Tree representation

- Internal nodes: questions
- Arcs: answer to questions
- Leaf nodes: prediction

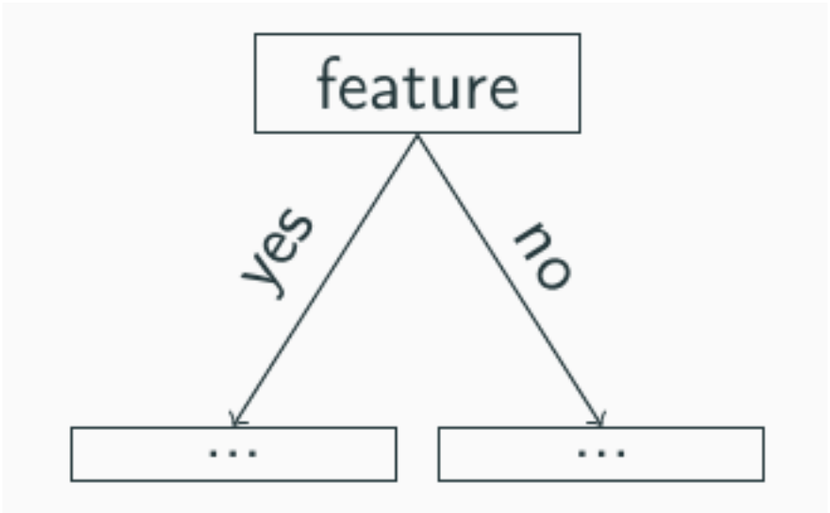


Features

- Questions look at features of the input
- Answer look at the **value** of a given feature

	Algorithms	OOP	Machine Learning	Graphs
Maryam	yes	no	no	yes
Nadi	no	yes	yes	no
Diaa	yes	no	no	yes
Oana	yes	no	yes	no
		...		

MODELING A DECISION TREE



Nodes

- Internal nodes: a triplet (left, left node, right node)
- Leaf nodes: label/value, i.e. output prediction

Tree

- Set of nodes

```python
def is_leave(node):
    return type(node) != tuple

def get_label(node):
    return node[0]

def get_left(node):
    return node[1]

def get_right(node):
    return node[2]
```

Internal nodes are represented as tuples

First element is the feature name (course name)

Second is the node to go if the student liked the course

Second is the node to go if the student disliked the course

```python
def predict(node, student):
    if is_leave(node):
        return node
    if get_label(node) in student:
        return predict(get_left(node), example)
    else:
        return predict(get_right(node), example)
```

If a leaf node, just return the prediction!

Take decision with respect to the student feature values

```python
tree = ("Algorithms", 0, ("Probabilities", 1, 0))

student1 = ["Graphs", "Probabilities"]
print(predict(tree, student1))  # output is 1

student2 = ["Algorithms", "Graphs", "Probabilities"]
print(predict(tree, student2))  # output is 0
```

Tree represented as a tuple

List of courses the student liked

TRAINING A DECISION TREE

Learning task

Find the decision tree that best models the training data

Problem

We cannot enumerate all trees except in trivial cases!
(i.e. except when the number of features is very small)

Approximate learning

- If we restrict the tree to a single feature,
  it is easy to find the best solution
- Find a decent solution for the full tree via a **greedy algorithm**

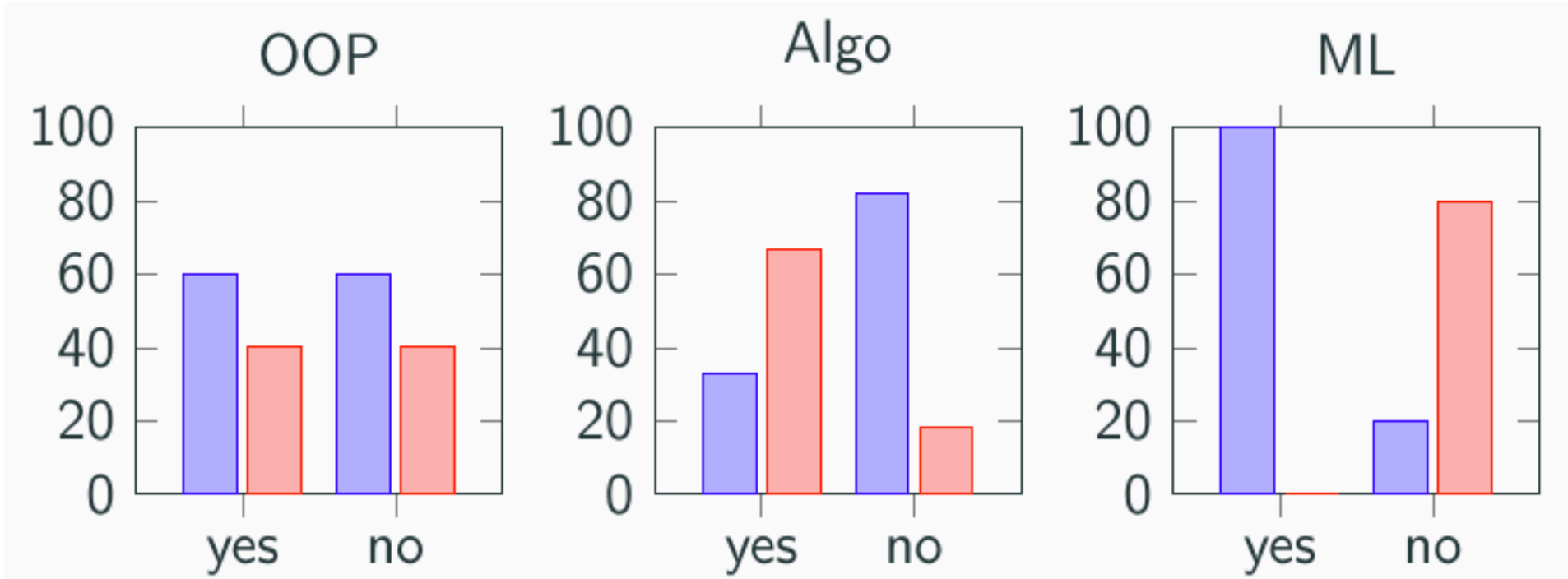
HOW TO SELECT A FEATURE? 1/3

Example

Task: will a student like the **Deep Learning** class?

Data

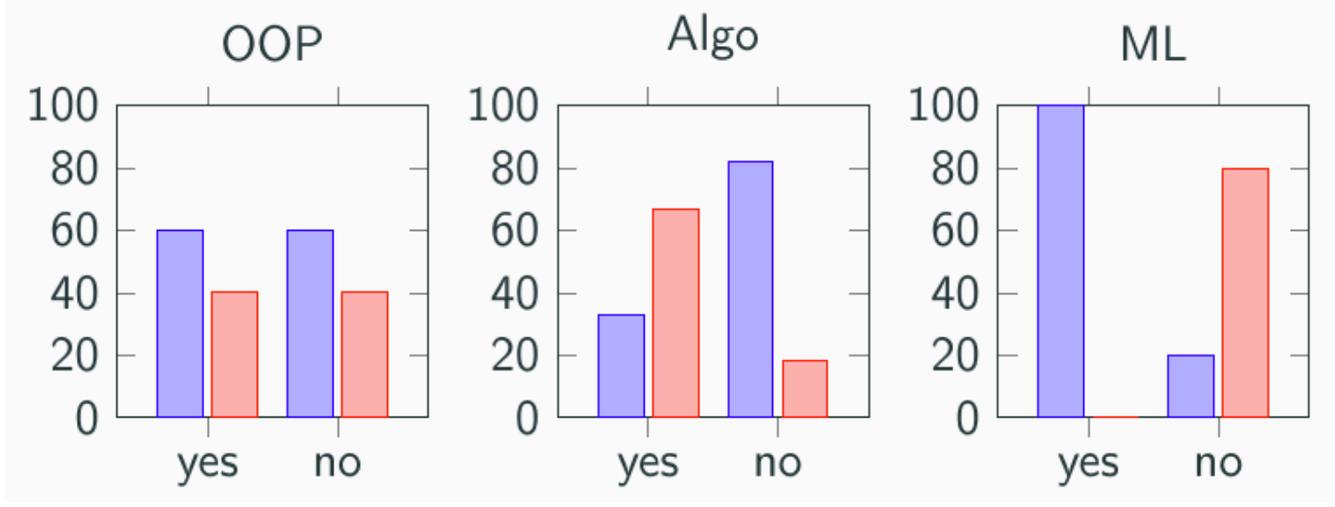
- Features: OOP, Algoritms, Machine Learning (i.e. other classes)
- We also know if students in the training data like the deep learning class or not



Blue : % of students that liked deep learning (red => didn't like), depending on whether they liked or not another course (OOP, ...)

For example: 60% of students that like OOP also liked the deep learning class

HOW TO SELECT A FEATURE? 2/3



Feature OOP

- Students that liked OOP: 60% also liked Deep Learning
- Students that disliked OOP: 60% also liked Deep Learning

This feature does not give any information!

Feature Algorithms

- Students that liked OOP: 35% also liked Deep Learning
- Students that disliked OOP: 80% also liked Deep Learning

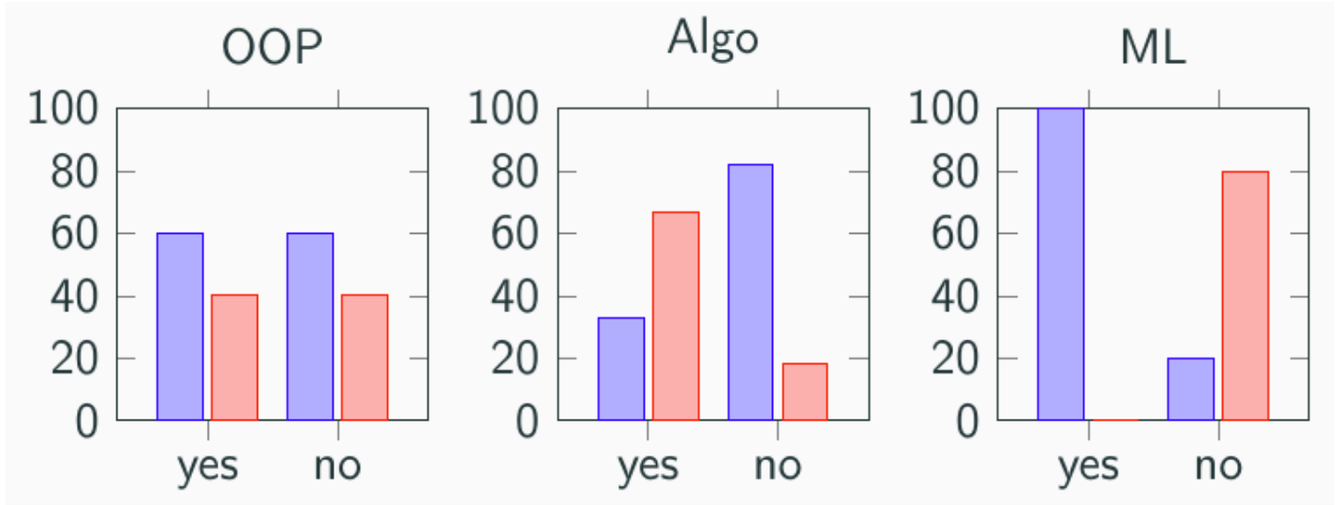
Better

Feature Machine learning

- Students that liked ML: 100% also liked Deep Learning
- Students that disliked ML: 20% also liked Deep Learning

Best discriminative feature!

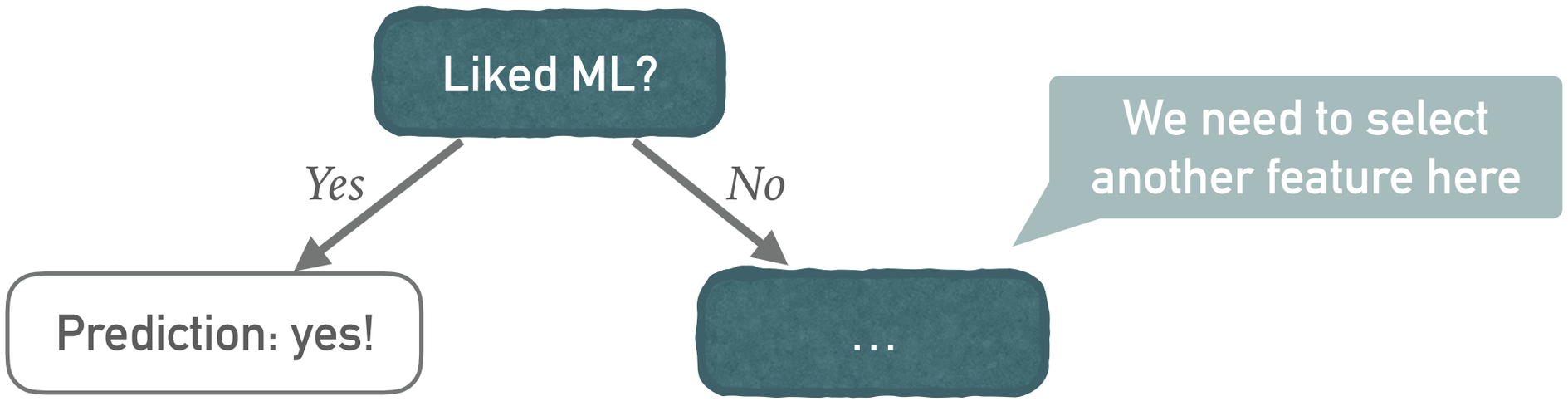
HOW TO SELECT A FEATURE? 3/3



Feature Machine learning

- Students that liked Machine Learning: 100% also liked ML
- Students that disliked Machine Learning: 20% also liked ML

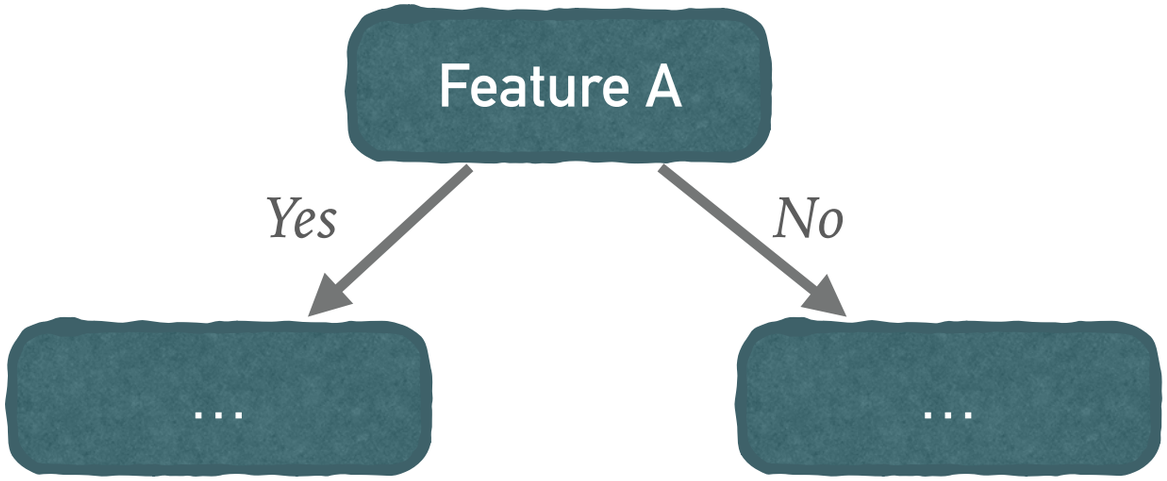
Best discriminative feature!

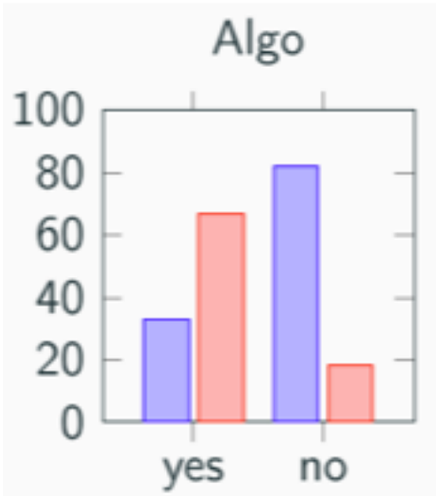


RECURSIVE CONSTRUCTION

Passing features to child nodes

- Yes branch: pass only example that **contains** Feature A
- No branch: pass only example that **does not contains** Feature A
- Remove the Feature A from example
  (i.e. it cannot be selected as a feature anymore)





What to do when we have only a single feature left?

- Predict the most probable output, no matter if the remaining examples contains the feature or not

Algorithm 1 DECISIONTREETRAIN(*data*, *remaining features*)

```
guess ← most frequent answer in data                    // default answer for this data
if the labels in data are unambiguous then
    return LEAF(guess)                                  // base case: no need to split further
else if remaining features is empty then
    return LEAF(guess)                                  // base case: cannot split further
else                                                    // we need to query more features
    for all f ∈ remaining features do
        NO ← the subset of data on which f=no
        YES ← the subset of data on which f=yes
        score[f] ← # of majority vote answers in NO
                 + # of majority vote answers in YES
                                                        // the accuracy we would get if we only queried on f
    end for
    f ← the feature with maximal score(f)
    NO ← the subset of data on which f=no
    YES ← the subset of data on which f=yes
    left ← DECISIONTREETRAIN(NO, remaining features \ {f})
    right ← DECISIONTREETRAIN(YES, remaining features \ {f})
    return NODE(f, left, right)
end if
```

DECISION TREE CONCLUSION

Training algorithm

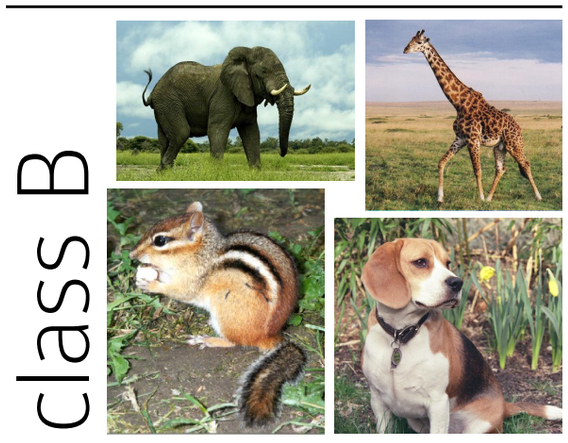
- This a « simplified » training algorithm, there are many variants (ID3, C4.5, …)
- Pruning methods
- We need to "bound" the maximum depth of the tree

Explainability

The decision tree can actually give an explanation of the output,
i.e. **justify why** it decided to output a given prediction

INDUCTIVE BIAS 1/4

Train data



Test data

What are the label on the test data?

Not a bird, but can fly

Bird that cannot fly

There several « correct » solutions:

- A=fly, B=no-fly
- A=bird, B=no-bird

We need to **bias** toward one!

INDUCTIVE BIAS 2/4

INDUCTIVE BIAS 3/4

[McCoy et al., 2019]

Natural language inference

Predict the relation between a premise and a hypothesis: Implication, neutral, Contradiction

John likes Baltimore a lot.

John likes Baltimore.

Implication

Datasets

- **Stanford NLI:** 570.000 pairs of sentences
- **Multi NLI:** 433.000 pairs of sentences

We must prevent the learning algorithm to do that!

Bad heuristic learned that can be learned

- If the premise is a subsequence of the hypothesis, relation is: **implication**

John likes Baltimore a lot.

John likes Baltimore.

Alice believes Mary is lying.

Alice believes Mary.

Roses are red, and **violets are blue.**

Violets are blue.

The book on the table **is blue.**

The table is blue.

INDUCTIVE BIAS 4/4

Definition

- Set of assumption that the training algorithm must know about the data
- No generalization without inductive bias
  (i.e. no inductive bias = algorithm will just learn by heart the data)
- Main difference about different couple of prediction function and training algorithm is the bias that they (implicitly) produce

Problem specific

- Natural language processing: sequence, hierarchical structures
- Computer vision: translation and rotation invariant, color or black and white, …
- …

Machine learning is not a black box toolbox, **one solution does not fit all problems**

Even with deep neural networks!

INDUCTIVE BIAS OF DECISION TREES

Shallow decision trees

- Require that a tree cannot query more than T features
- T is a **hyper-parameter**,
  i.e. a non-learned value that parameterize the learning algorithm

Inductive bias of shallow decision tree

- Decision can be made by looking at a small number of features
- Not able learn something like:
  « A student likes ML only if has liked an even number of lectures »

Could be implicitly learned even if it is not a given feature

UNDERFITTING/OVERFITTING

Underfitting

- Do not « learn / extract » information available in the data
- High train and test error

Overfitting

- Pay too much attention to idiosyncrasies of data (i.e. learned by heart)
- No training error, high testing error

Do not generalize to unseen data!

Extreme decision trees

- Empty tree (no question asked): arbitrary training error
- Full decision tree (query all features): can easily overfit

K-NEAREST NEIGHBORS

EXAMPLE: PREDICTING THE STATE OF WATER

Problem

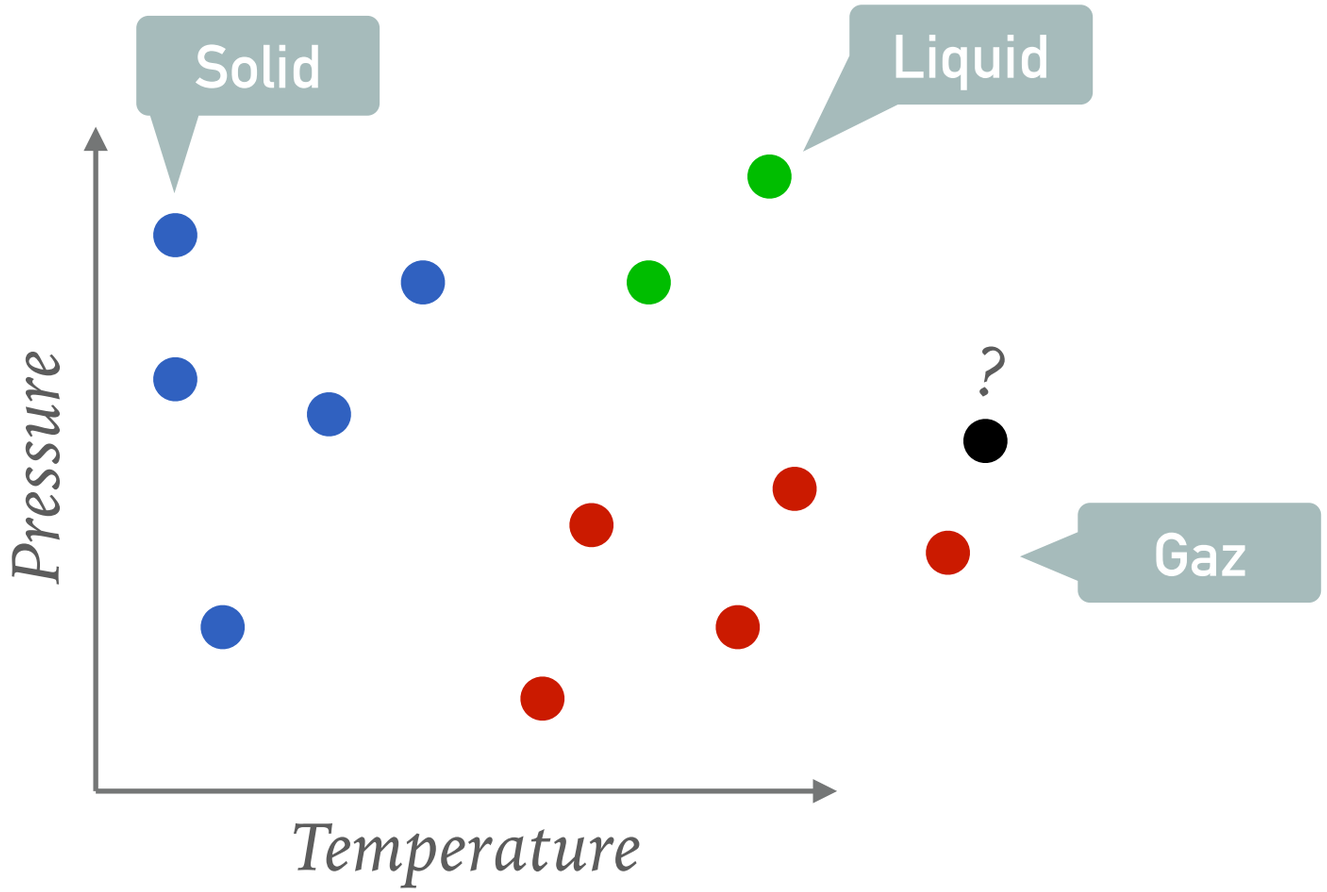
- Water can be in 3 states: **liquid**, **solid**, **gas**, ~~plasma~~
- The state depends on **temperature** and **pressure**

Can we predict the state from the temperature and pressure?

Obtaining data

1. Manipulate (e.g. change the fire level)
2. Observe temperature/pressure
3. Observe water state
4. Record data (e.g. add point on a plot)

EXAMPLE: PLOT



First measure

- Temperature: x
- Pressure: y
- State: solid

Feature engineering

Observing temperature and pressure is feature extraction!

Prediction

Given a new point, can we guess its state?

NEAREST NEIGHBORS CLASSIFIER

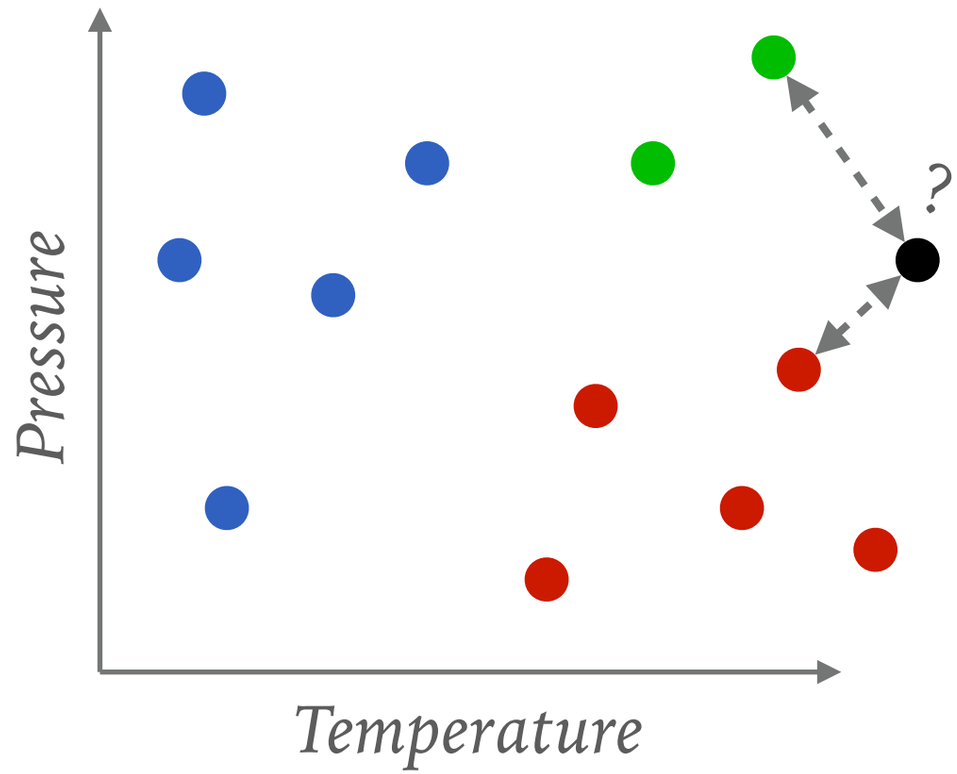
You need to define what « closest » means!

Principle

The label of a new point is the label of the closest point in the data

Euclidean distance

$$dist(x, x') = \sqrt{\sum_i (x_i - x'_i)^2}$$



Prediction function

$$f(x'; \theta = \mathcal{D}) = \text{argmin}_{x,y \in \mathcal{D}}\, dist(x, x')$$

The function is directly parameterized by the training set

Model

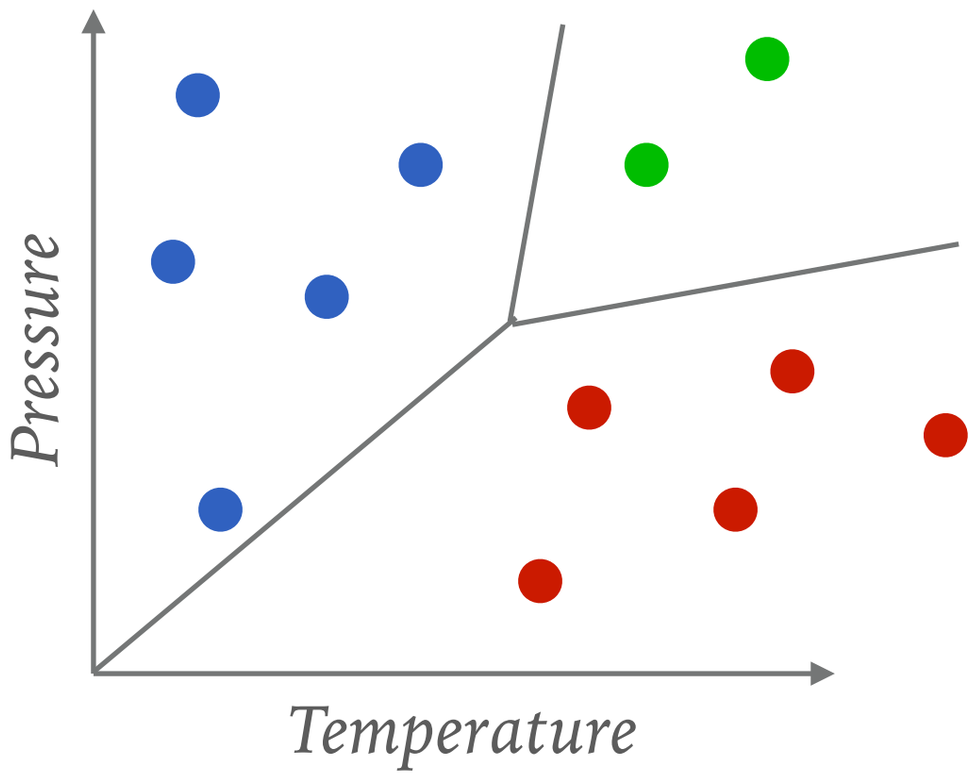
- Hyper-parameters: topological space
  (e.g. euclidean space/distance, Poincaré disk, hamming distance…)
- Complexity of prediction: $\mathcal{O}(n \times m)$
  n: number of features, m: number of training points

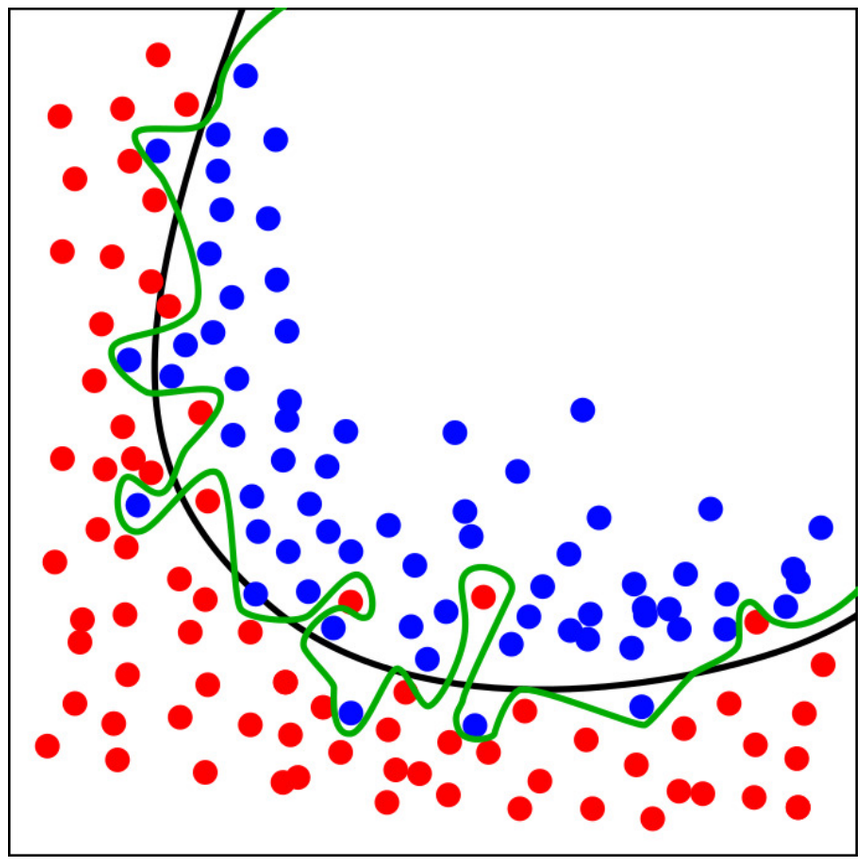
DECISION BOUNDARY

- Good way to visualize the model:
  - Are we robust to noize?
  - Are we overfitting?

Not a formal check!

- For Nearest Neighbor classification: points that are equidistant





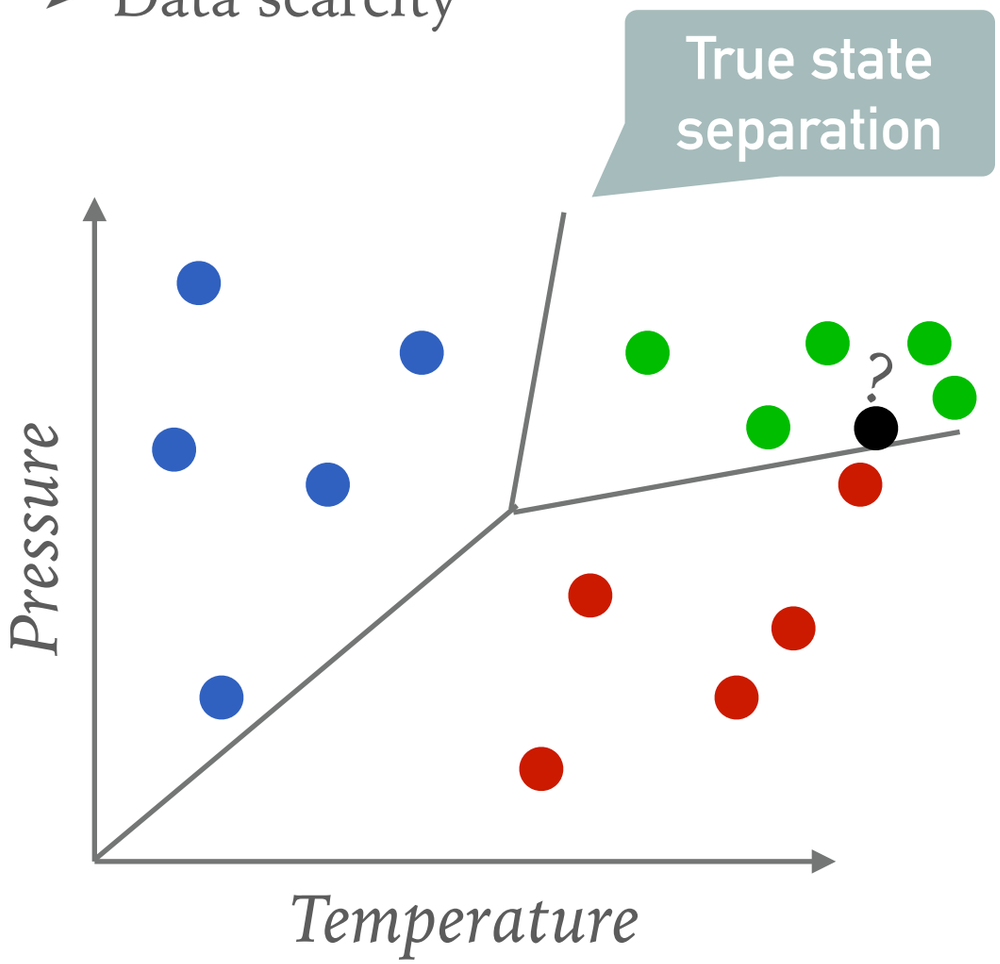
K-NEAREST NEIGHBORS

Principle

- Majority vote between the k-nearest points
- k is an hyper-parameter

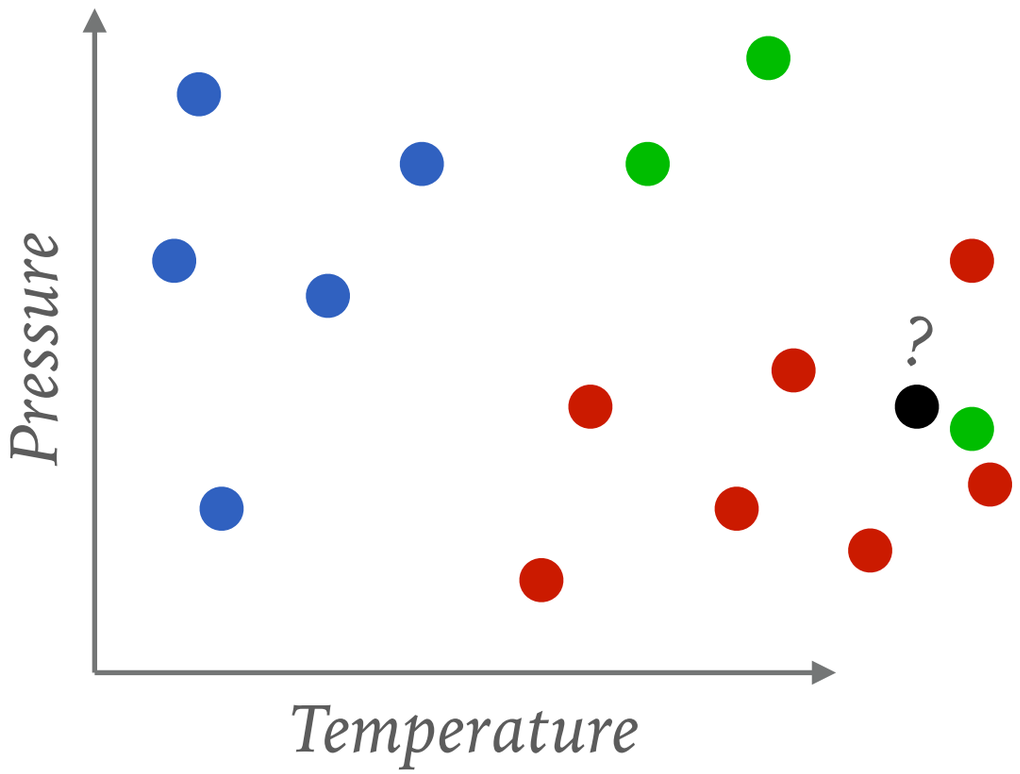
Strongest predictor

- Data scarcity



Noise robustness

- Outliers
- Incorrect data



INDUCTIVE BIAS

Inductive bias of k-NN

- Nearby points should have the same label
- All features are equally important

Downsides

- We need to carefully select the features!
  Not the case of decision trees which choose the best features to use
- The scale of features is really important! (and the notion of distance)