Machine Learning
V01: Introduction

Logistics of this module
History and breadth of Machine Learning
Inductive supervised learning
What is learnable? (CLT)

Prerequisites (in this order):
- Ch. 1.2–1.4 from [Murphy, ML-APP, 2012]
- Ch. 1.1–1.2 from [Mitchell, ML, 1997]
- Ch. 18.4 from [Russell & Norvig, AIMA, 2010]

With material from Javier Béjar, BarcelonaTech

Zurich University of Applied Sciences and Arts
InIT Institute of Applied Information Technology (stdm)

Zurich University
of Applied Sciences

0. LOGISTICS OF THIS MODULE

About me

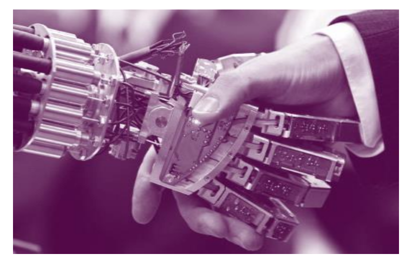
Thilo Stadelmann

- Born 1980, married
- Studied computer science in Giessen & Marburg, then doctorate (AI, voice recognition)
- Passion for programming & artificial intelligence (>20 years experience)

At ZHAW

- Email: stdm@zhaw.ch, office: TD 03.16 (Obere Kirchgasse 2)
- Tel.: 058 934 72 08, web: https://stdm.github.io/
- Professor for AI/ML at InIT/School of Engineering, scientific director of ZHAW digital

Interests



About You

Logistics

Lecture

- Theory & some practice intertwined
- Break between 45min blocks?

Self-study

- Read & experiment as much as possible at home (→ see literature & exercises)

Material

- Find everything on the course e-learning platform
- Video recording, if done, is best-effort only and is no replacement for your presence in the lectures

Labs & grading

- See terms & conditions on the course platform / next slide

Terms & conditions
MSE – TSM_MachLe

Course platform: https://moodle.msengineering.ch/course/view.php?id=1076 or
https://stdm.github.io/ml-course/ (video-lectures)

Dates
- 90 min lecture, 45 min lab per date (in 2 groups, sequentially)
- 19.02. – 19.03.: 5x Thilo Stadelmann (stdm@zhaw.ch)
- 26.03. – 07.05.: 6x Christoph Würsch (christoph.wuersch@ntb.ch)
- 21.05. – 28.05.: 2x Thilo Stadelmann

Labs
- Please **split autonomously** into 2 groups
- As long as there is space / resources, **both groups can be present** at any lab
- Time serves as a starter to the task → material suffices to go on during self study time

Grading
- 120 min written exam, pen & paper (no electronic devices used/allowed)
- Closed book, but a **2-sided A4 sheet** with **handwritten** notes is allowed (not copied / printed)

Module schedule
MSE – TSM_MachLe

Week	Date	Topic	Content	Practice	Self study	Lecturer
		Preparation			P01.1-5	
1	19.02.	Introduction	V01: Introduction	V01/P01: Discuss ML fundamentals	P01.6-7	stdm
2	26.02.		V02: Formulating learning problems	P02.1: Linear regression from scratch		stdm
3	05.03.	Supervised learning	V03: Model assessment & selection	P04.1 Analyzing cross validation	P03	stdm
4	12.03.		V04: SVMs	P04.2: SVM in IPython		stdm
5	19.03.		V05: Ensembles	P04.3: Ensembles in practice	P05	stdm
6	26.03.		V06.3: Debugging ML algorithms V07.1: System development - what to give priority?	P06.2 Applying learning curves	Raschka ch. 6; P07	würc
7	02.04.		V08: Feature engineering	Lab08		würc
8	09.04.		V09a: Probabilistic reasoning, Gaussian distribution and Bayes' therom	Lab09a	09a reader	würc
9	16.04.		V09b: Gaussian processes	Lab09b	Do, 2017	würc
10	23.04.		*holiday (Easer)*			
11	30.04.	Unsupervised learning	V10: Dimensionality reduction	Lab10	Raschka ch. 5	würc
12	07.05.		V11: Clustering	Lab11	Raschka ch. 11; Ng	würc
13	21.05.	Selected chapters	V12a: Learning games from selfplay	P12: Selfplay for Tic-Tac-Toe		stdm
14	28.05.		Open / FAQ	Open / FAQ		stdm

Superior educational objectives

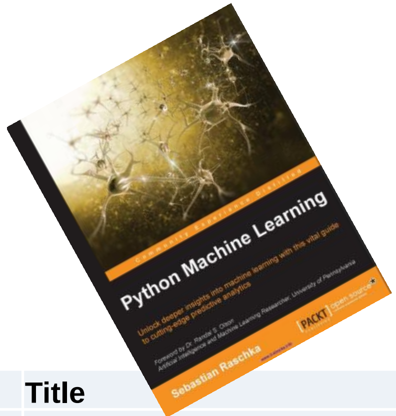
- You **have** a **solid foundation** and **best practices** for the **application of ML**

- You **can go on** from here **increasing** your foundation **through self study**

- You **apply** **ML algorithms** using Python and state-of-the-art libraries
- You **are able to** **select** a **suitable learning algorithm** and **prepare** respective **features** for a given data set

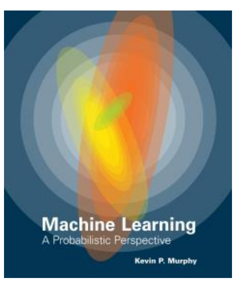
➔ We focus on **overarching principles** & **practical advice**
➔ **Read** the literature to know **more algorithms**

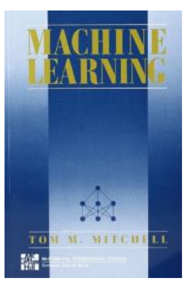
Literature

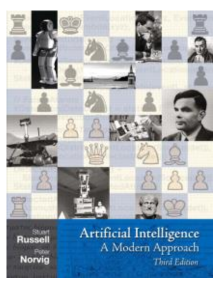


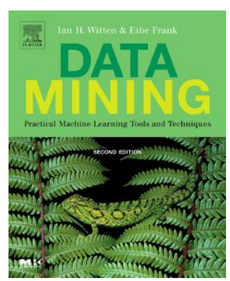
See
literature-guide.xlsx
on the course site (/Material)

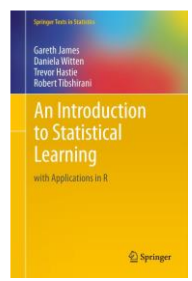
Tag	Authors	Title	Year	Subjective impression
Murphy	Murphy	Machine Learning - A Probabilistic Perspective	2012	Very comprehensive, lots of details, quite academic → required reading for overview of the field (Ch. 1)
Mitchell	Mitchell	Machine Learning	1997	Very concise, rigorous thoughts but explanatory & accessible → required reading for background on general learning (Ch. 1)
AIMA	Russel, Norvig	Artificial Intelligence - A Modern Approach, 3rd Edition	2010	Concise overview from an AI perspective in 2 chapters → required reading for learnability & model selection (Ch. 18)
WEKA	Witten, Frank	Data Mining - Practical Machine Learning Tools &Techniques, 2nd Ed.	2005	Very readable introduction from a data mining perspective → suitable for self study, companion to Java WEKA toolkit
ISL	James, Witten, Hastie, Tibshirani	An Introduction to Statistical Learning with Applications in R, 4th Printing	2014	Great concise introduction for using statistical learning → great read and application-relevant, R & python code
ESL	Hastie, Tibshirani, Friedman	The Elements of Statistical Learning, 2nd Edition	2009	Very comprehensive, lots of mathematical details → big brother of ISL when more details are needed
Duda et al.	Duda, Hart, Stork	Pattern Classification, 2nd Edition	2001	Focus on Pattern Recognition applications → great overview when dealing with pattern recognition data
Dlbook	Goodfellow, Bengio, Courville	Deep Learning	2016	Principled compendium to all things dl, good intro to math → complementary to others, see www.deeplearningbook.com

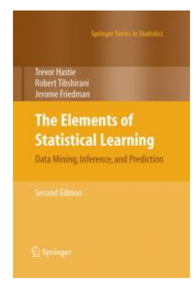


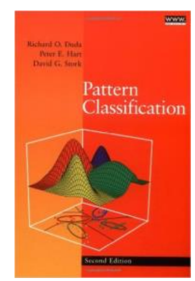


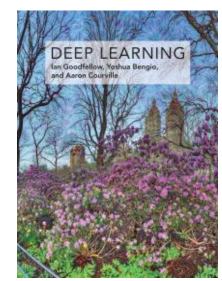












Educational objectives for today

- **Know** the **history** and **breadth** of the discipline **of Machine Learning** to categorize material
- **Understand what is** (machine) **learnable** under the paradigm of inductive (supervised) learning
- **Comfortably tap into** (scientific) machine learning **literature**

1. HISTORY AND BREADTH OF MACHINE LEARNING

What is Machine Learning?
…and how does it relate to learning in general?

Wikipedia on «Learning», 2015:

«...the act of **acquiring** new, **or modifying** and reinforcing, existing **knowledge**, behaviors, **skills**, **values**, or preferences and **may involve synthesizing** different types of information.»

“do something”

A. Samuel, 1959:

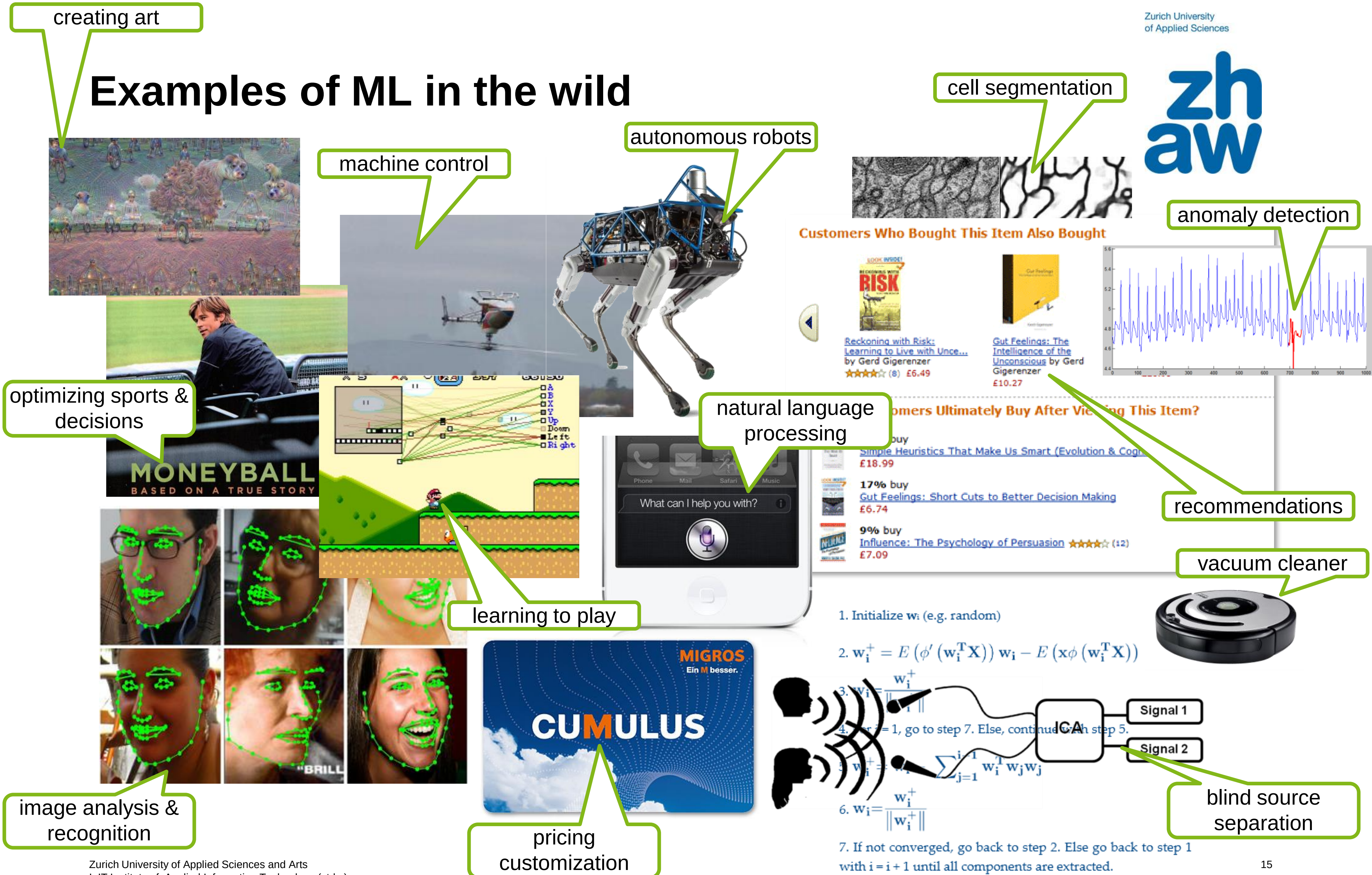
«…gives computers the **ability** to learn **without being explicitly programmed**.»

T.M. Mitchell, 1997:

«…if its **performance** at tasks in T, as measured by P,
improves with experience E.»

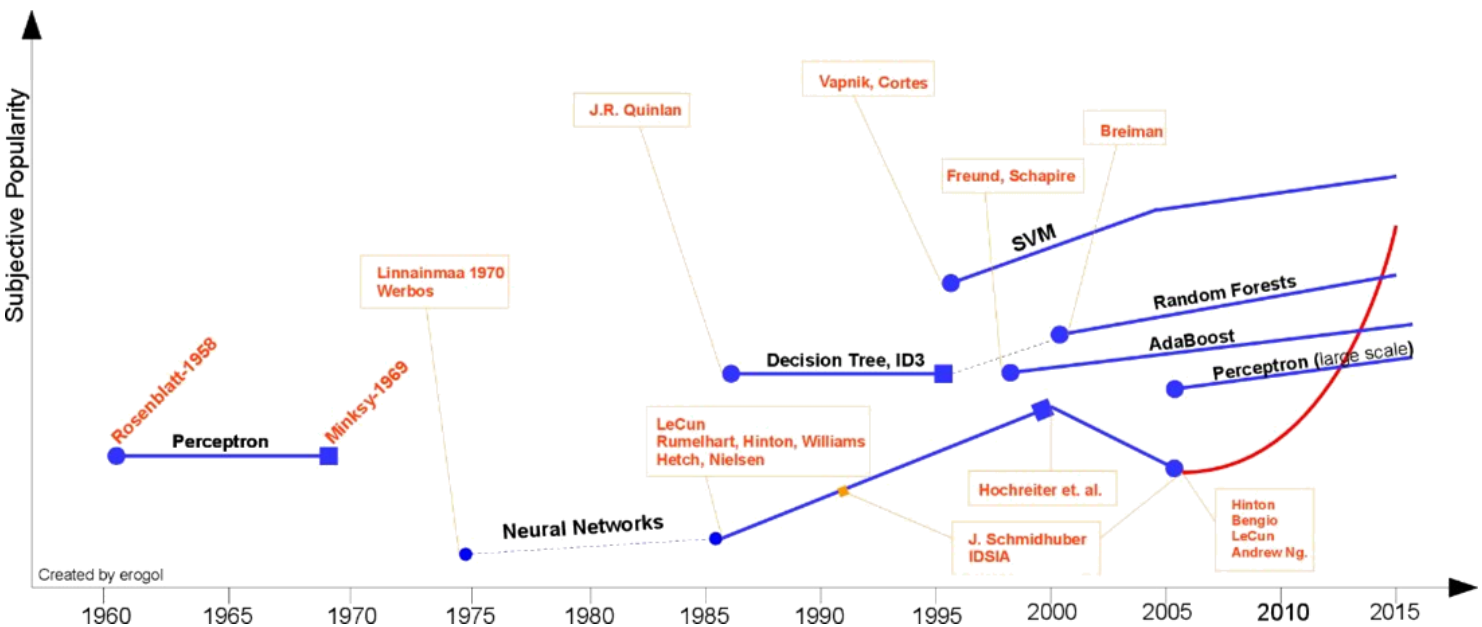
➔ In practice: Fitting parameters of a function to a set of data
(data usually handcrafted, function chosen heuristically)

Examples of ML in the wild



A simplified history of Machine Learning

- Discipline has its roots in AI
- Many methods have roots in statistics
- Two cultures: model-driven vs. «algorithmic»
  → see [Breiman, 2001]

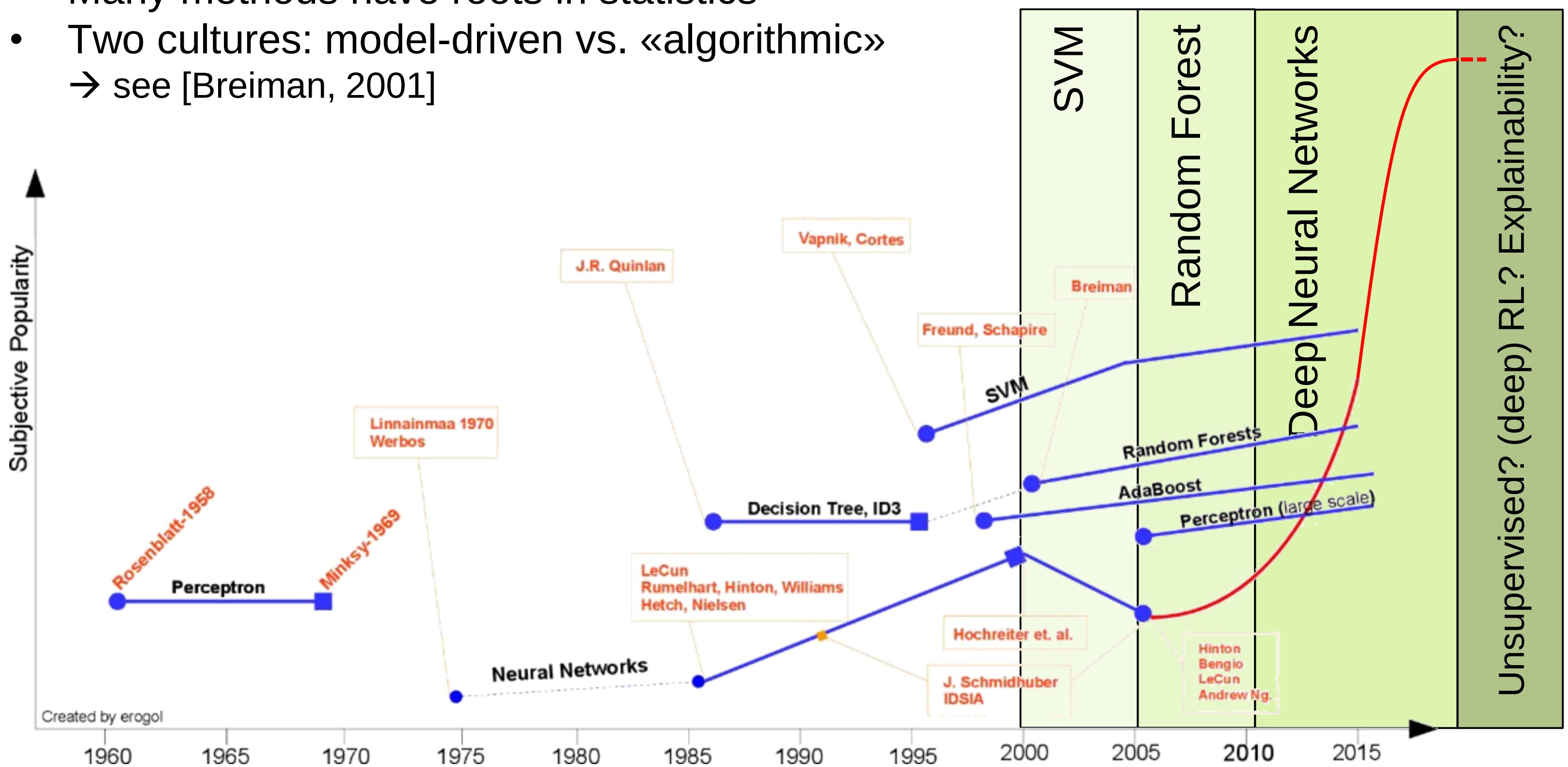


Original chart by Diego Marinho de Oliveira, https://www.linkedin.com/pulse/20141024101110-52688293-brief-history-of-machine-learning

A simplified history of Machine Learning

- Discipline has its roots in AI
- Many methods have roots in statistics
- Two cultures: model-driven vs. «algorithmic»
  → see [Breiman, 2001]

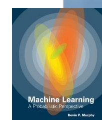
Trends in research-oriented practice
(subjective view)

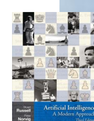


Original chart by Diego Marinho de Oliveira, https://www.linkedin.com/pulse/20141024101110-52688293-brief-history-of-machine-learning

Exercise: Recap from P01 reading assignment

➔ Your 2 facts & 3 questions?





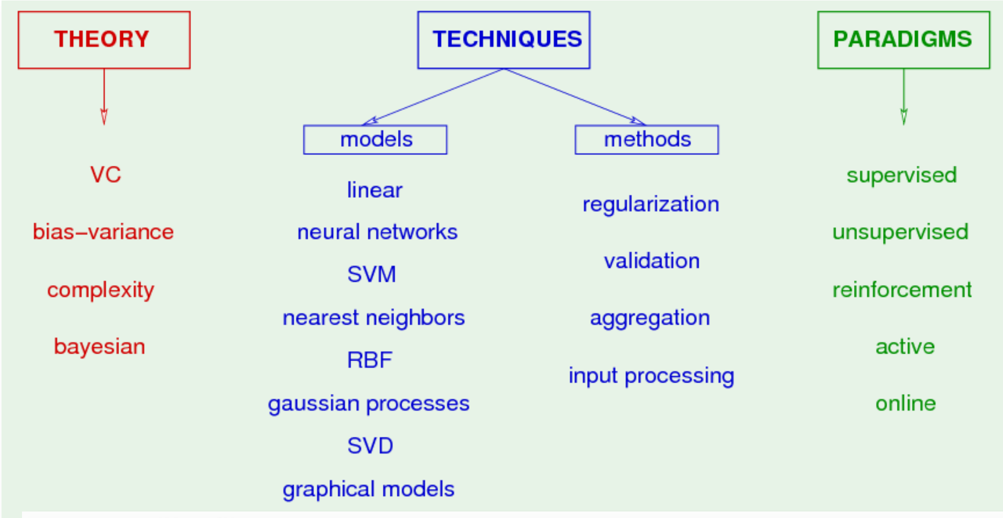
Important points

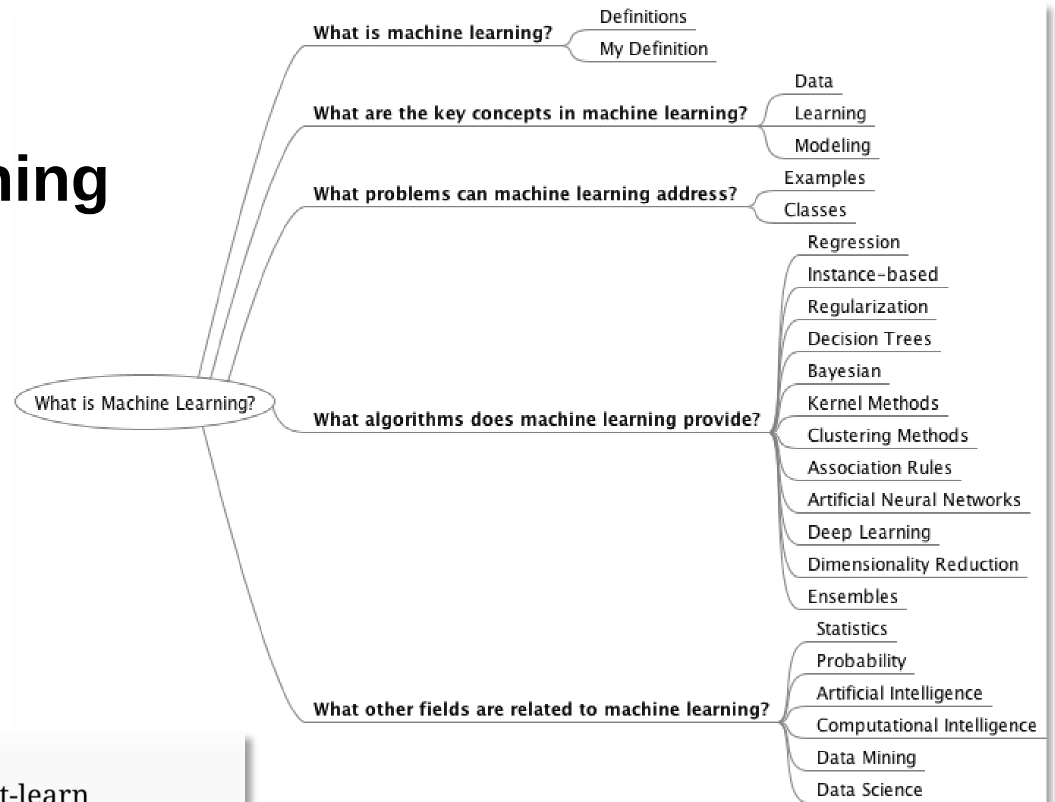
- Brief examples for supervised and unsupervised learning [Murphy, 1.3–1.4]
- Basic idea of few algorithms: kNN, linear regression & logistic regression [Murphy, 1.4.1–1.4.6]
- What is a well posed learning problem? [Mitchell, 1.1]
- One detailed example of formulating a learning problem [Mitchell, 1.2]
- Model flexibility and the bias-variance trade-off
  → the need for evaluation
  [Murphy, 1.4.7–1.4.9] [Russell & Norvig, 18.4]

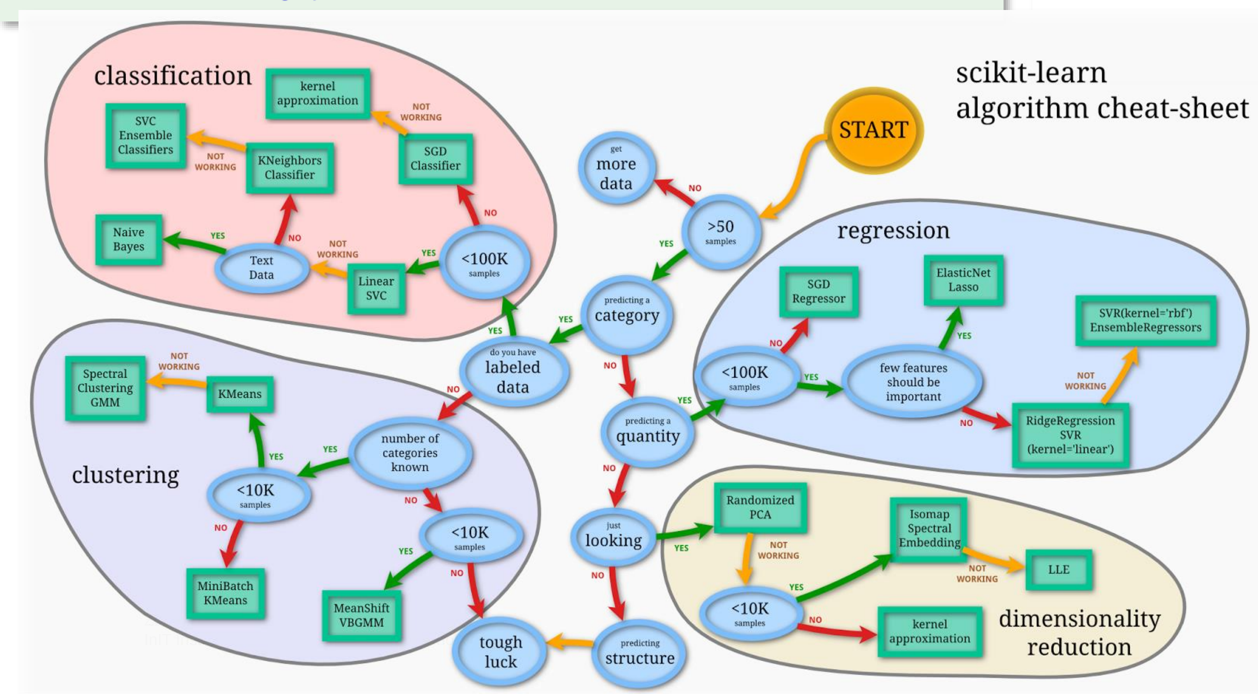
Not covered here specifically

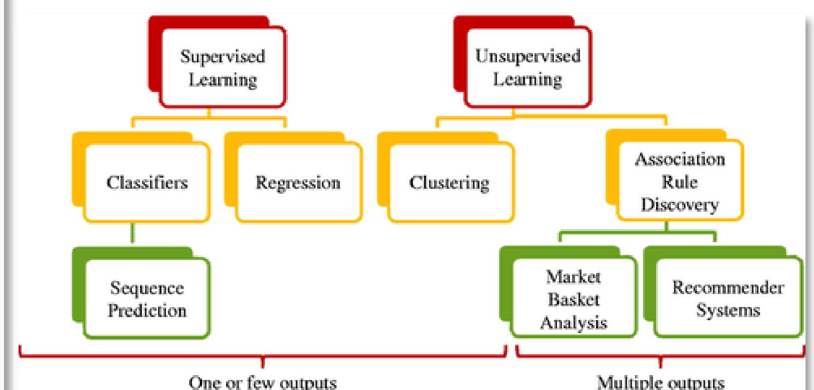
- Statistical perspective on hierarchical clustering
- Statistical perspective on kNN, CART, Random Forest

Trying to map Machine Learning

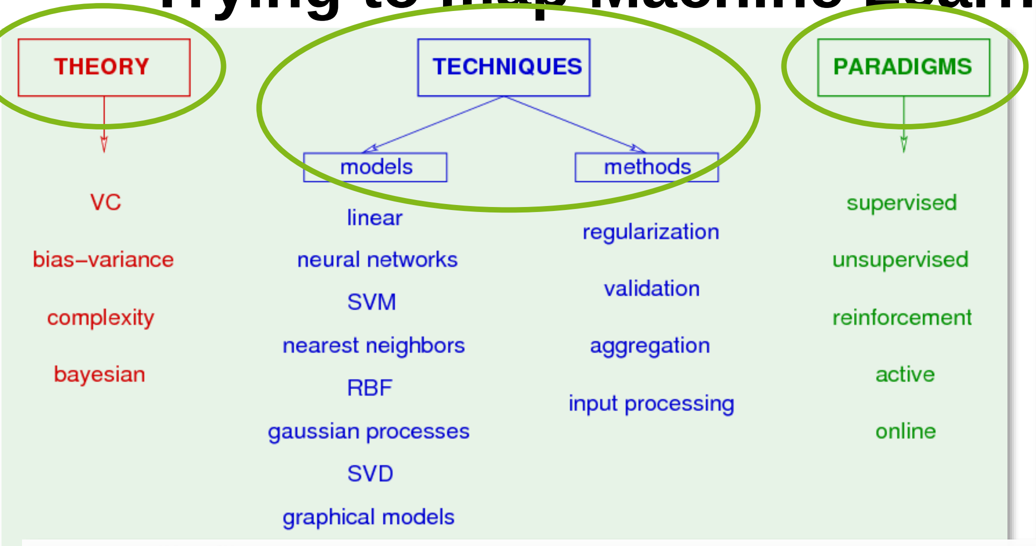


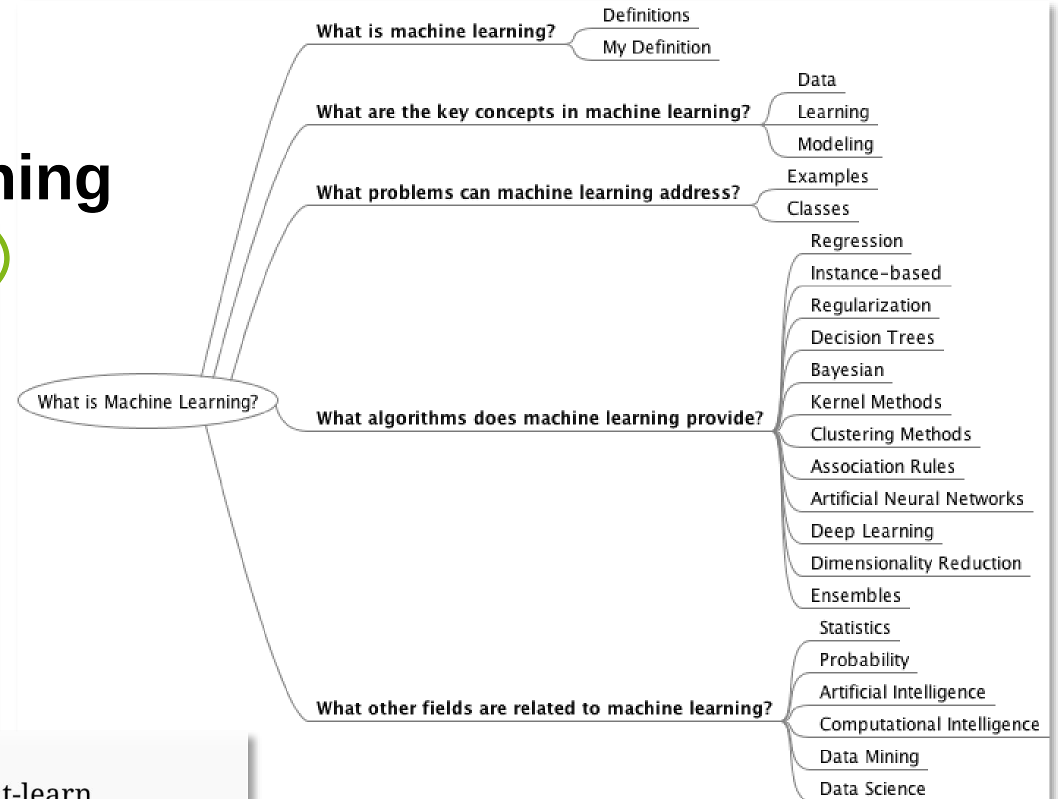


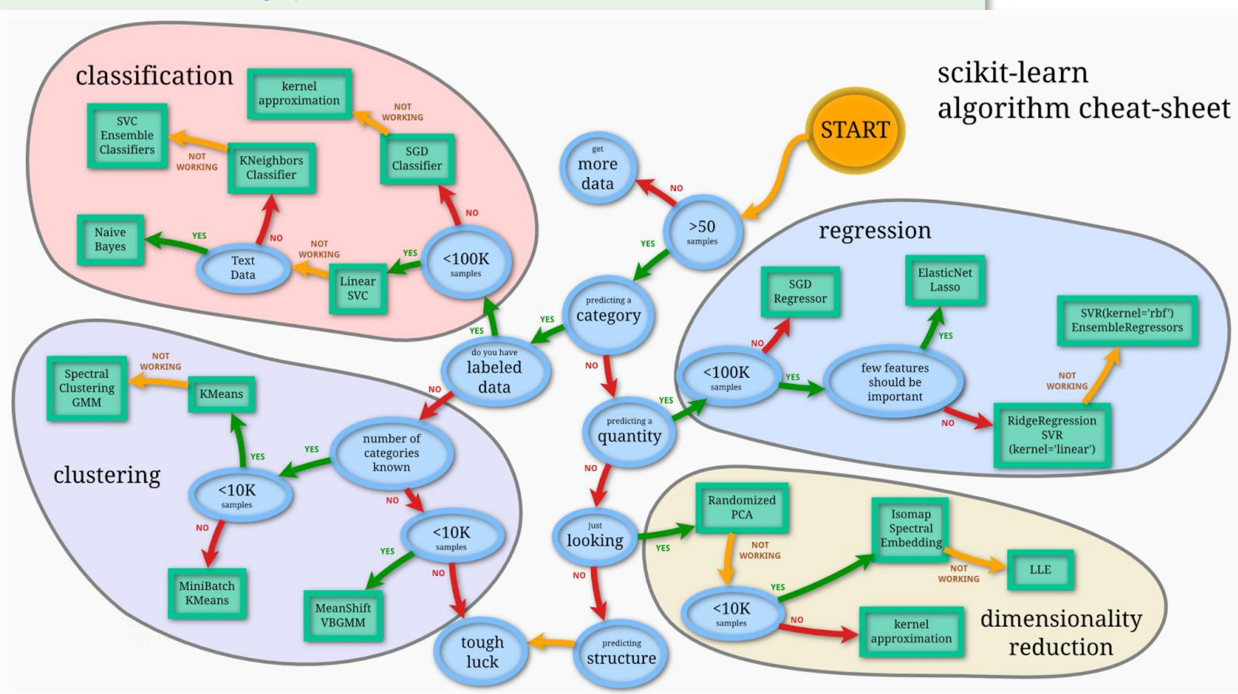


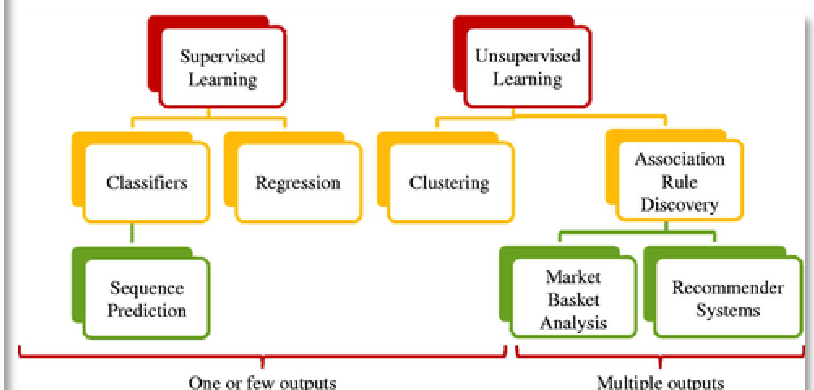


Trying to map Machine Learning

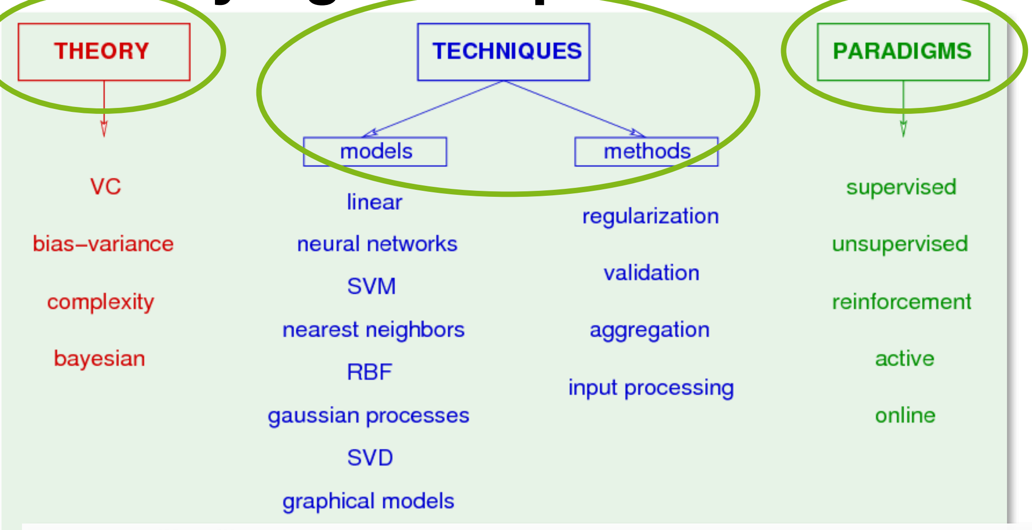


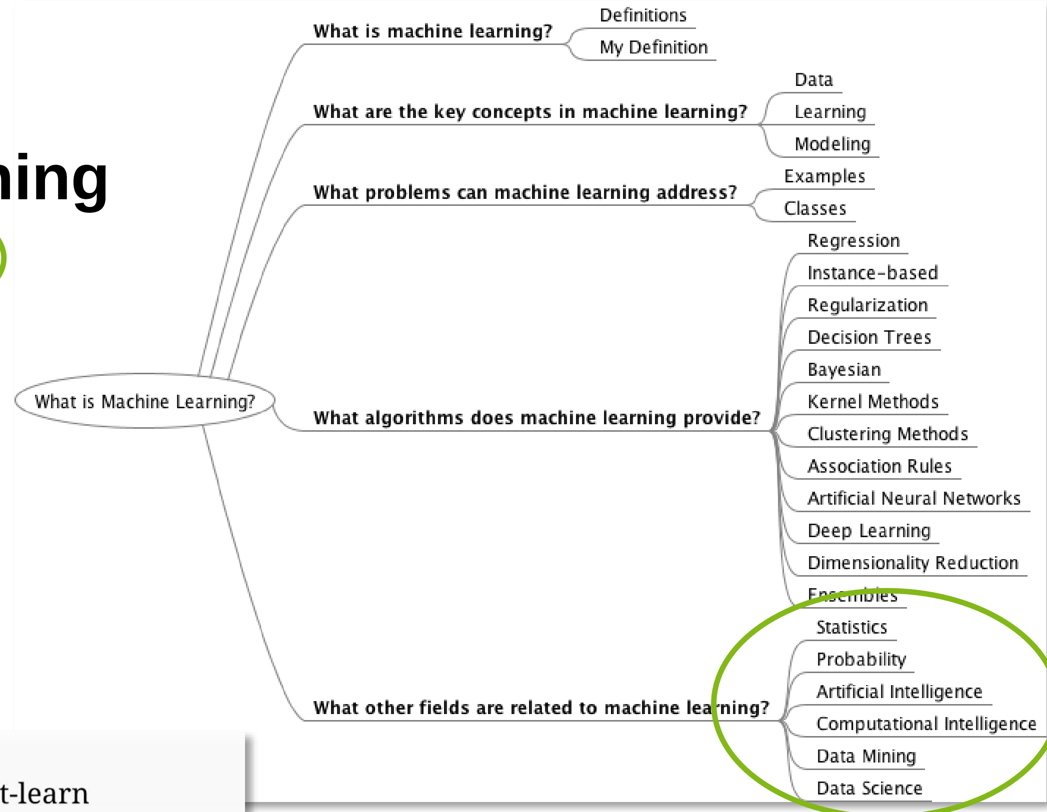


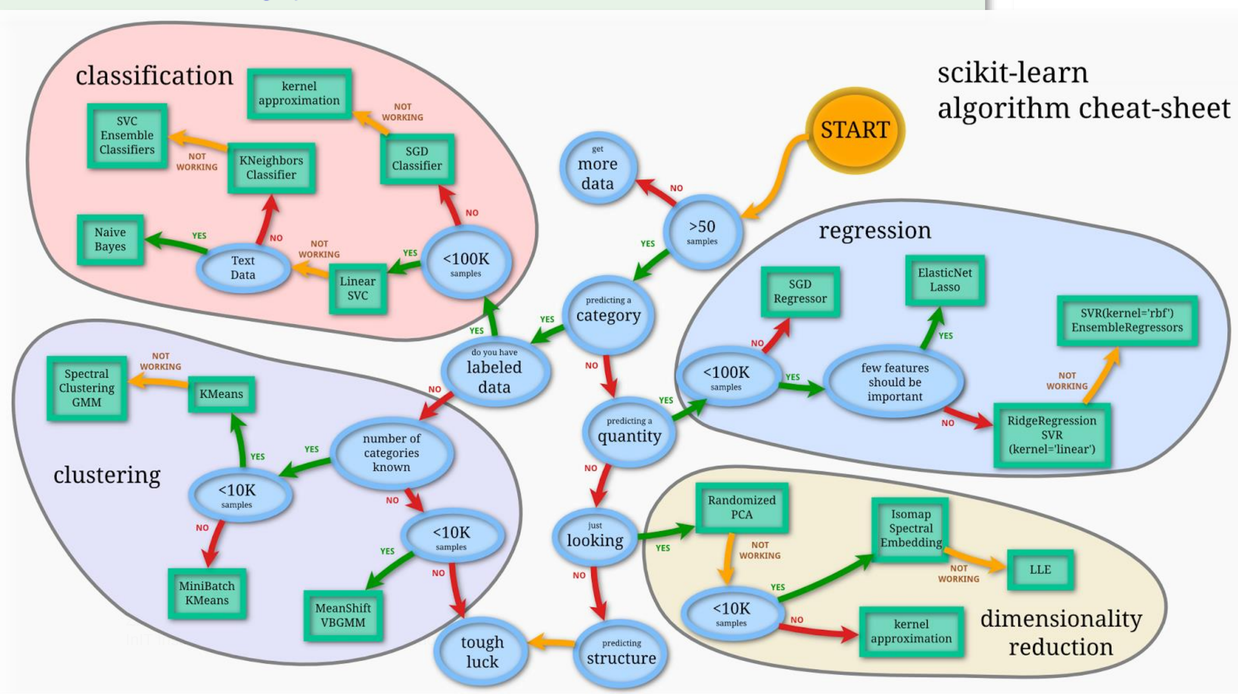


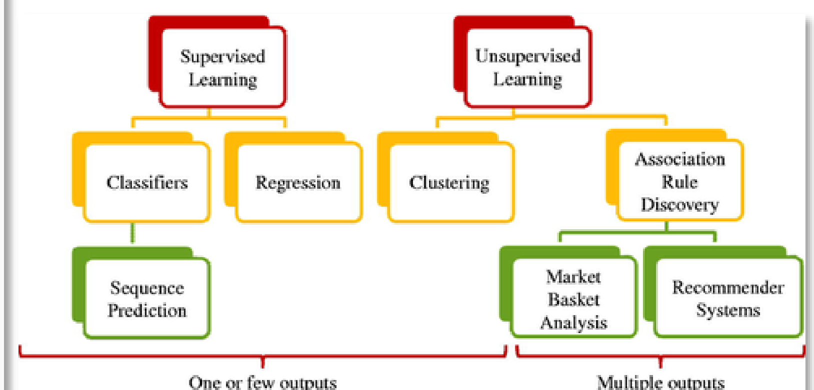


Trying to map Machine Learning

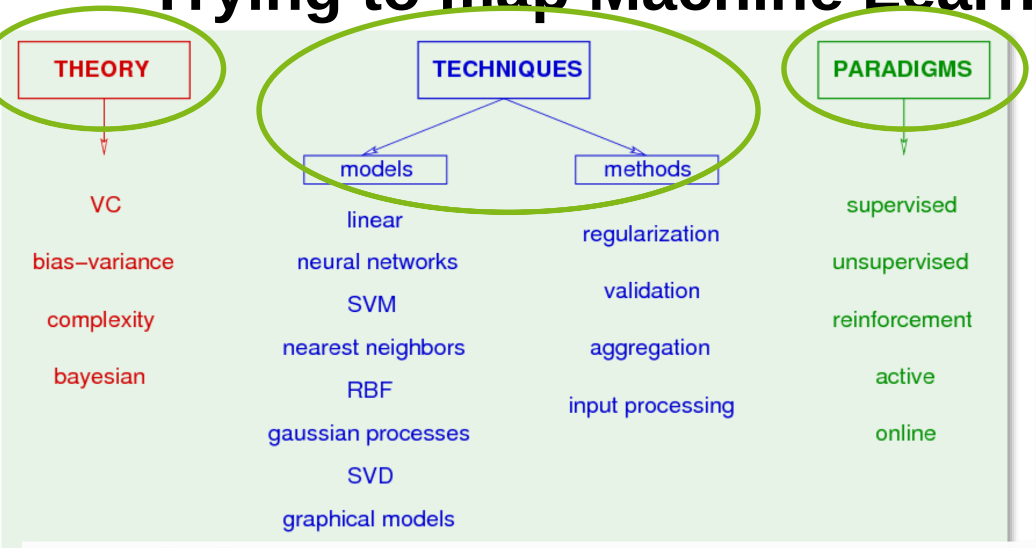


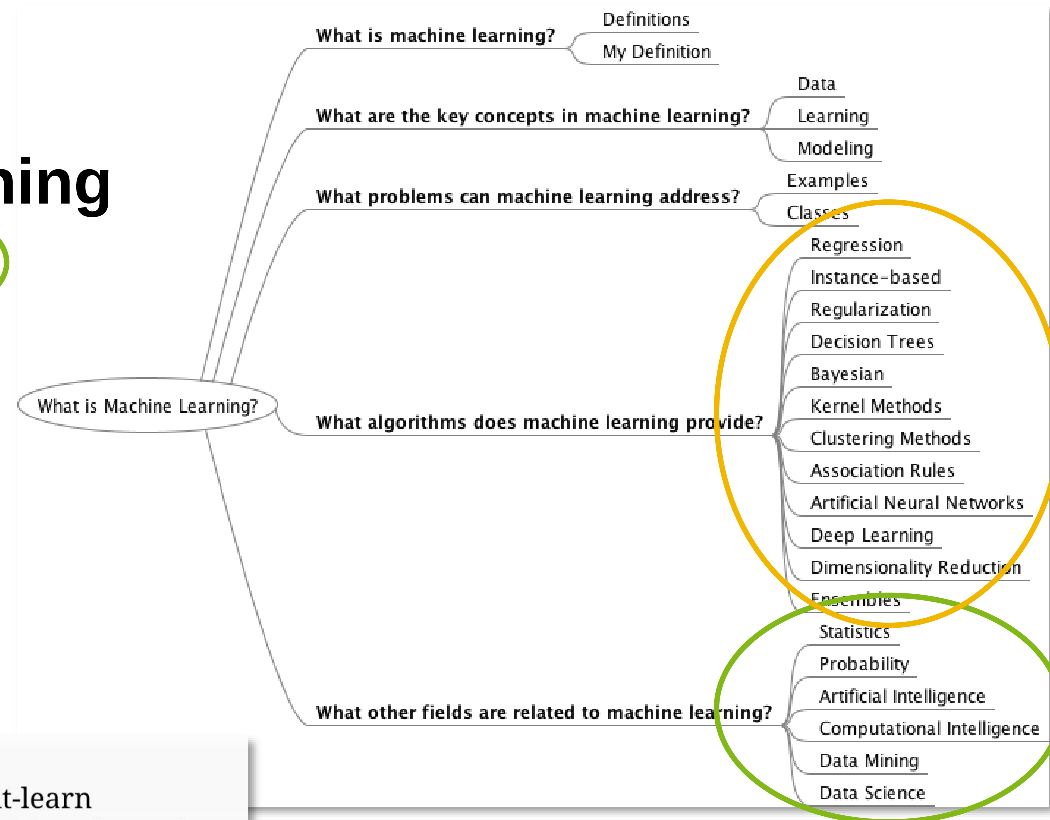


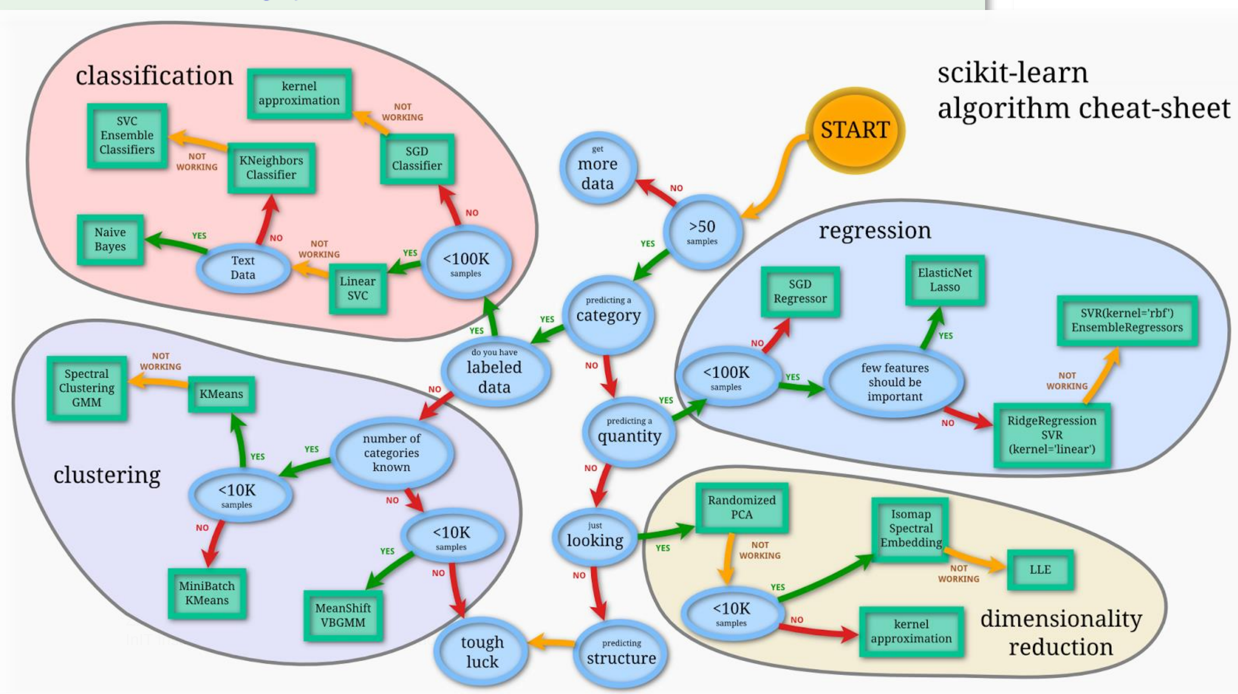


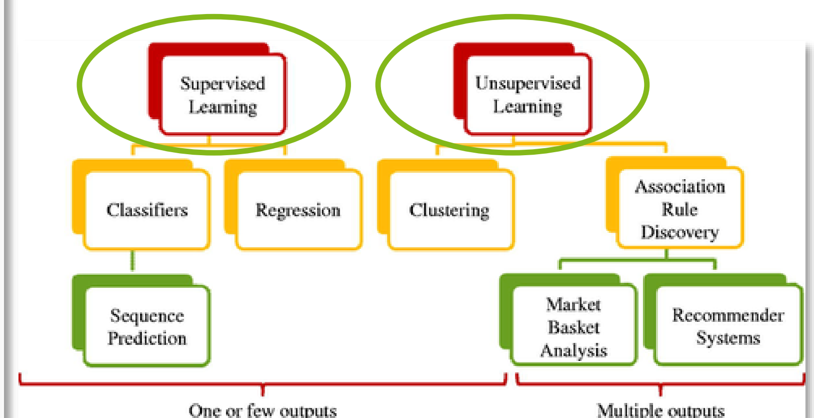


Trying to map Machine Learning

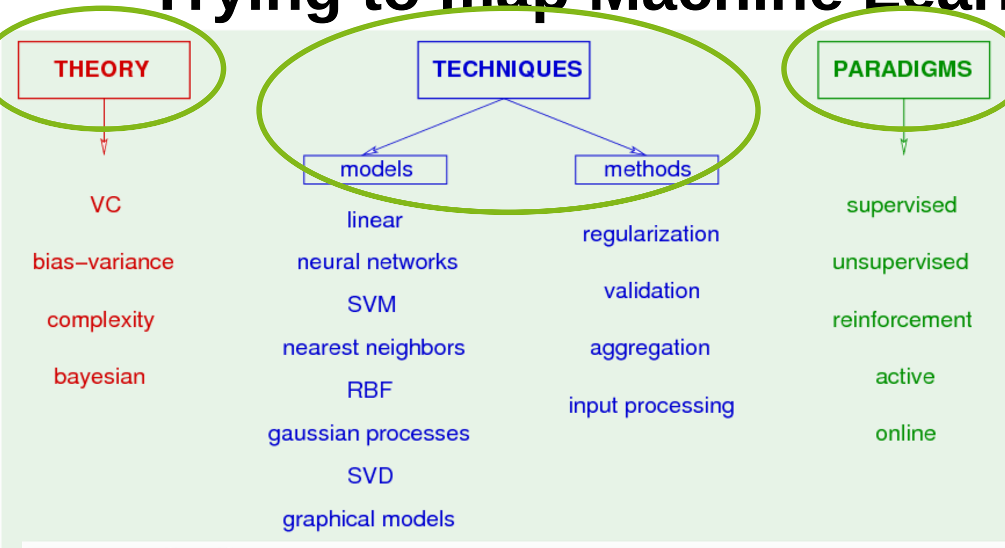


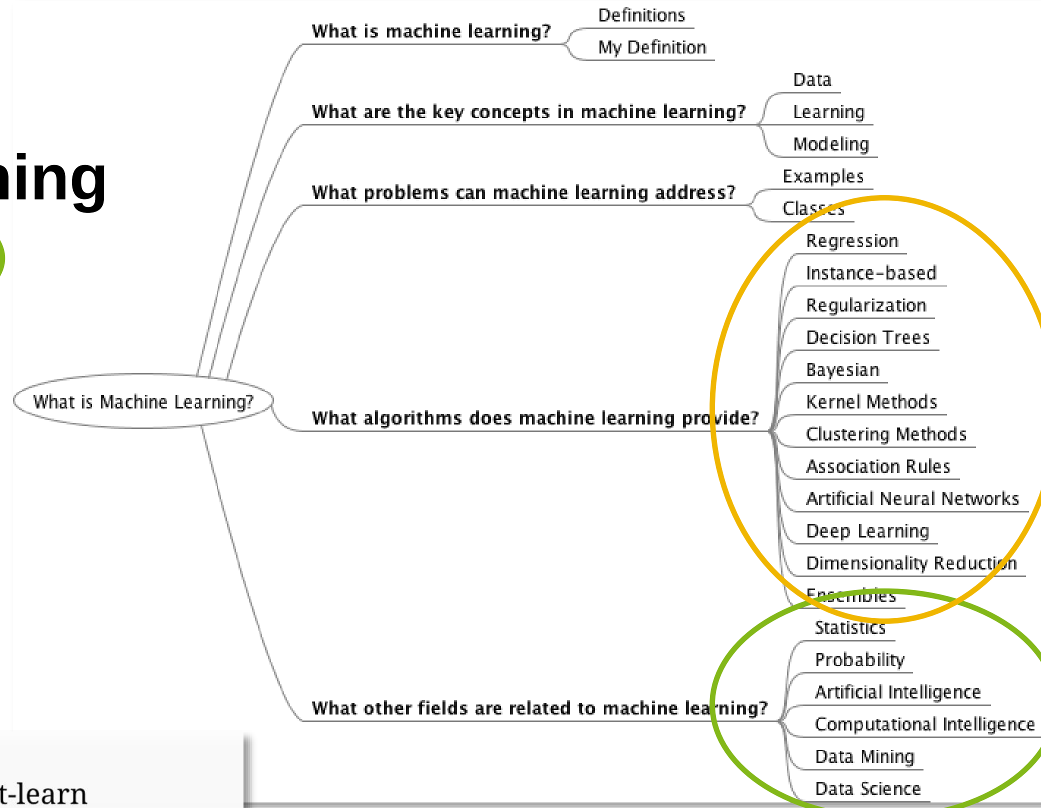


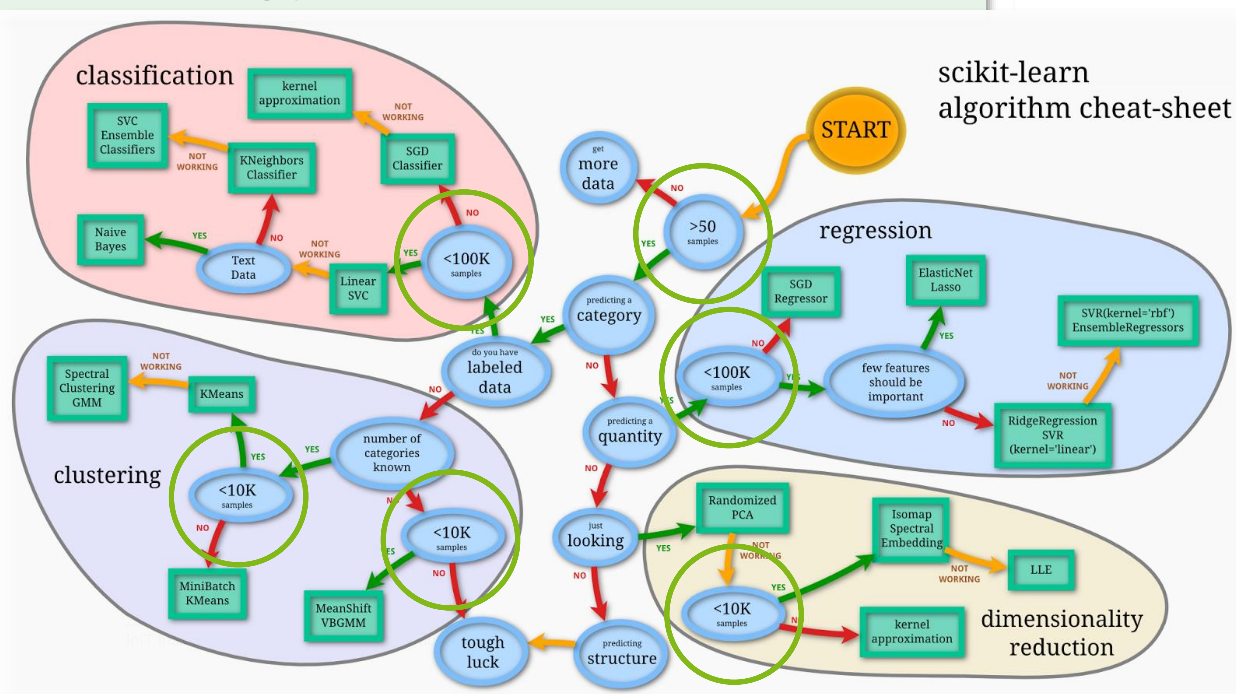


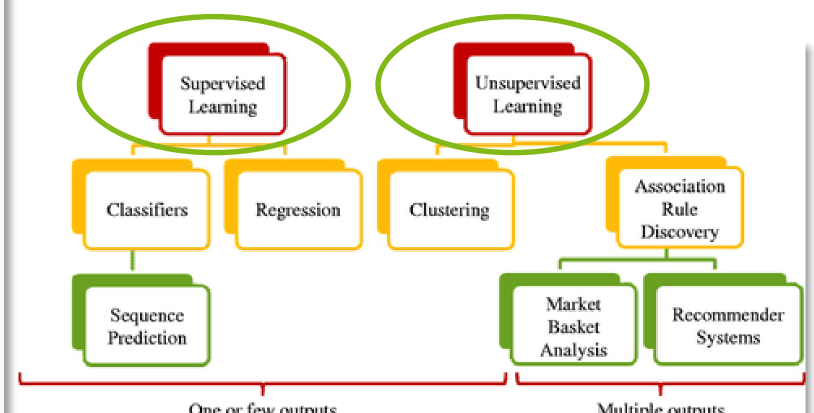


Trying to map Machine Learning



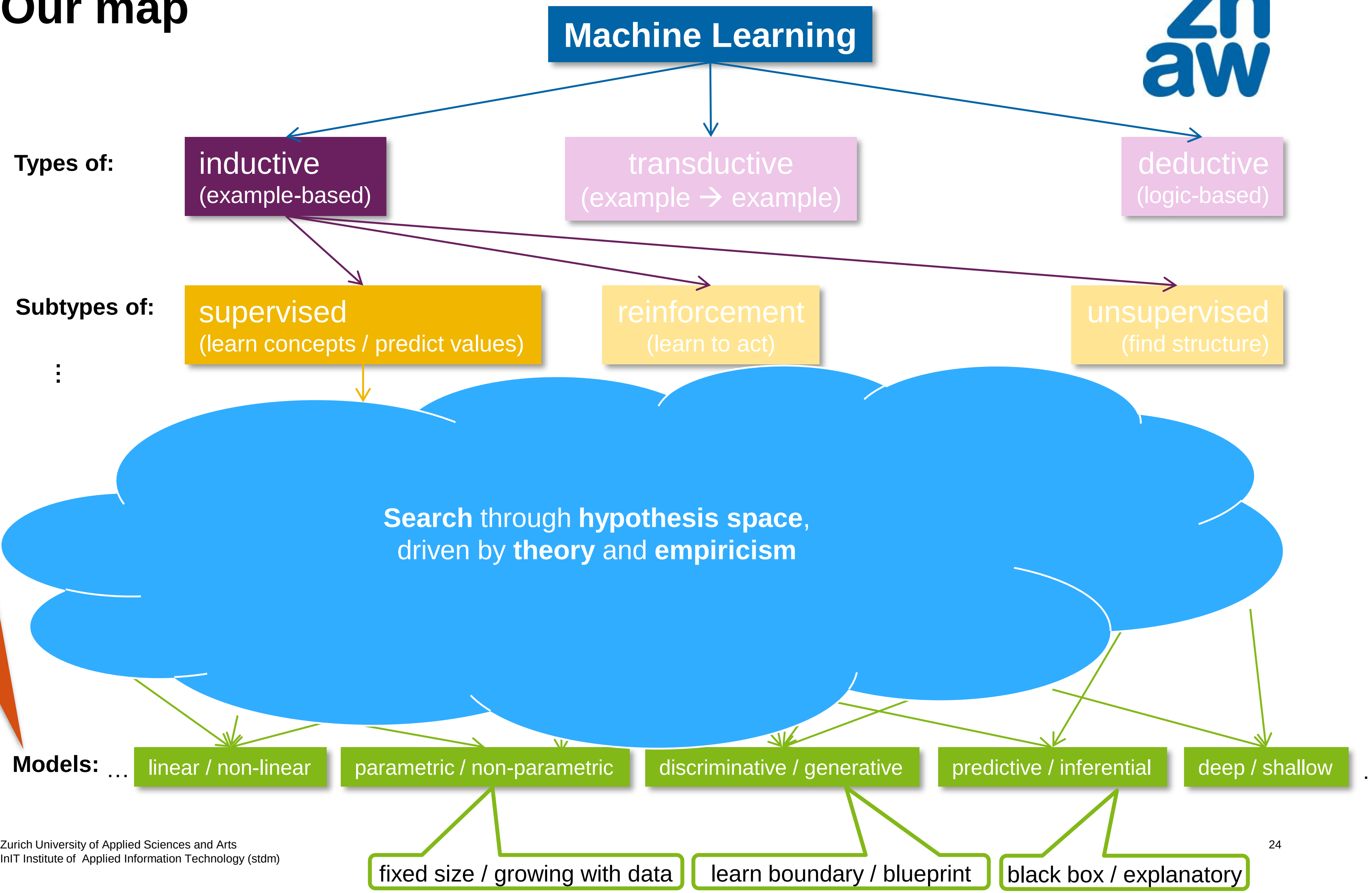






Our map

Model := "**a (abstract) system** which captures the characteristics of some (real-world) system for easier processing." → see appendix



2. INDUCTIVE SUPERVISED LEARNING

Inductive learning

Goal

- Discover **general** concepts **from** a **limited** set of **examples** (experience)

Methods are based on **inductive reasoning**:

- It obtains general knowledge (a **model**) from specific information
- The **knowledge** obtained **is new** (i.e., not implicitly present in a logical theory)
- Its **not truth preserving** (new information can invalidate the knowledge obtained)
- It is **heuristic** in nature (i.e., no well-founded theory)

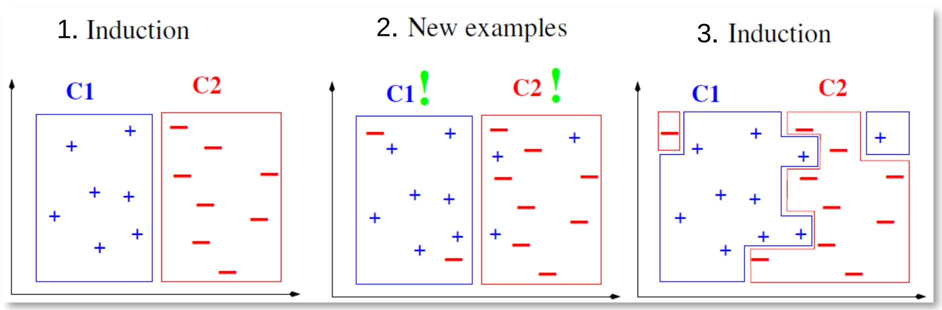
➔ **Assumption**: A model fitted to sufficiently large example set will **generalize** to unseen data

called the «inductive learning hypothesis»

What is sufficient? Basic ML research question!

Only one counterexample invalidates the result

➔ But, **most of the human learning is inductive**!

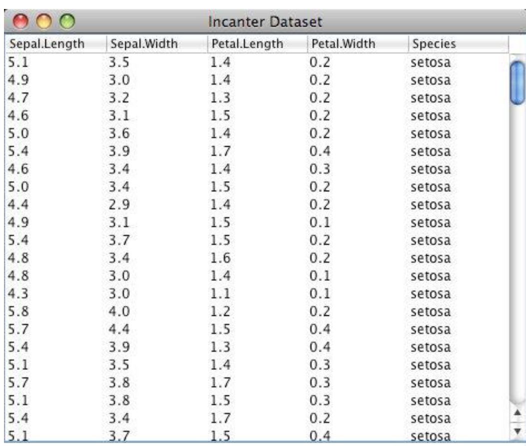


Zurich University
of Applied Sciences

Inductive supervised learning
Classification & regression

Semi-formal representation

- N Examples are usually described by **attribute-label pairs** $(\vec{x}_i, y_i), \mathrm{i} = 1..\mathrm{N}$
- Labels usually denote concepts, e.g. $y_i = 0$ for “red”, $y_i = 1$ for “blue”
- Examples have been generated by some unknown function $f(x) = y$, and noise



Incanter Dataset

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa

instances / records, $i = 1..N$

p-dimensional **feature vectors**, x **labels**, y

we usually drop the vector notation

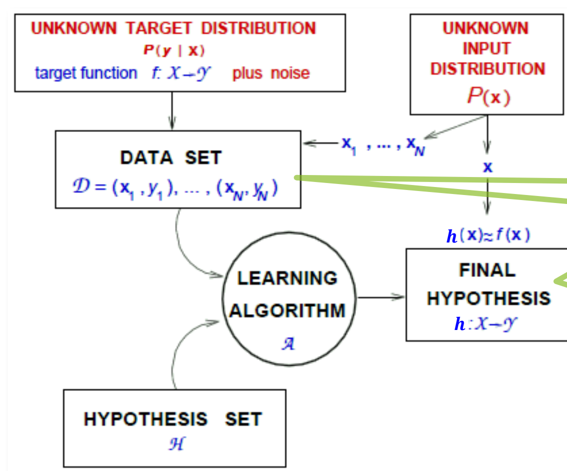
real-valued labels are also possible
➔ regression instead of classification

Goal

- Approximate the mapping function from example x to label y with a **hypothesis** $h(x) = \hat{y} \approx f(x)$
- …such that it **generalizes** well!

Basic ML research questions!

➔ Which is the **best approximation**, what are the **candidates**, how to **search** them?



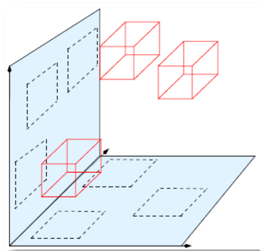
training data
(X, Y)

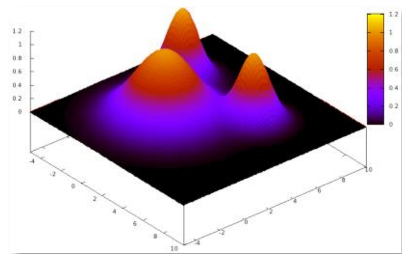
applicable to **test data** (just X)

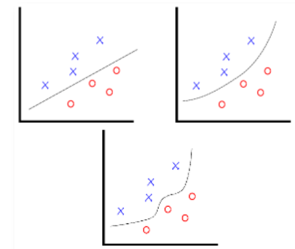
Zurich University
of Applied Sciences

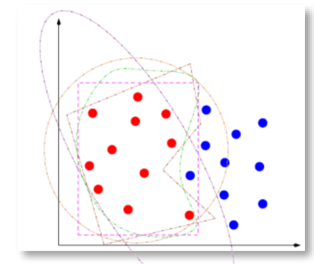
Learning as search
…through a hypothesis space $\mathcal{H}$

Hypothesis spaces


Logic formulas in DNF


Probabilistic models (PDF)


Linear/non-linear functions


Ellipse, rectangle, …

- $\mathcal{H}$ contains all possible hypothesis that can be built with the chosen representation

Formal goal

explicit reference to parameters θ

- Find the hypothesis $h^*(x, \theta) = \hat{y}$ that *best fits the training data*…
  - …according to a **loss function** $L(h(x, \theta), y)$…
  - …by searching the hypothesis space $\mathcal{H} = \{h(x, \theta) | \theta \in P\}$ (P is the set of all possible parameters)
- That is:
  - find $h^* = \arg\min_{h \in \mathcal{H}} E_{emp}(h)$… minimizing average loss
  - …by minimizing the **empirical error** $E_{emp}(h) = \frac{1}{N}\sum_{i=1}^{N} L(h(x_i, \theta), y_i)$, with e.g. $L(\hat{y}, y) = \begin{cases} 0 \; if \; y = \hat{y} \\ 1 \; else \end{cases}$

0/1 loss is often used in classification tasks

➔ Very useful decomposition of inductive bias for practice:
language bias | search bias | overfitting-avoidance bias

Zurich University of Applied Sciences

Inductive bias

Guiding the search through $\mathcal{H}$

*«A learner that makes **no a priori assumptions** regarding the identity of the target concept has **no rational basis for classifying** any unseen instances»*
[Mitchell, 1997, Ch. 2.7.3]

No free lunch theorem regarding the general equivalence of learners

- When all functions f are equally likely, the probability of observing an arbitrary sequence of cost values during training does not depend upon the learning algorithm $\mathcal{L}$ [Wolpert, 1996]

➔ All learning algorithms have advantages & disadvantages, depending on the current data

Inductive bias of a learning algorithm $\mathcal{L}$ for instances in X

- Any **minimal set of assertions** B that, together with $\mathcal{L}$ and the training set, $D = \{(x_i, y_i)\}, i = 1..N$, **allows for deductively inferring** the y' for a new $x' \in X$
- That is: Make all assumptions **explicit** in B such that $\forall x' \in X$: $(B, \mathcal{L}, D, x') \Rightarrow y'$ is provable

i.e.: based on a priori knowledge

➔ Ultimately, ML depends on intelligent choice of the class of $\mathcal{H}$; $\mathcal{L}$ then optimizes the details
➔ We can characterize ML algorithms by (the strength of) their inductive bias

We see more optimistic scenarios in the area of (deep) feature learning

Inductive unsupervised learning

Clustering and beyond

Usual task: Clustering

- N Examples are described by feature vectors $\vec{x}_i, \mathrm{i} = 1..\mathrm{N}$ *without any labels*
- The examples naturally fall into K groups; K and the group membership function $f(x) = y, y \in 1..K$ are unknown

Challenges

a form of inductive bias!

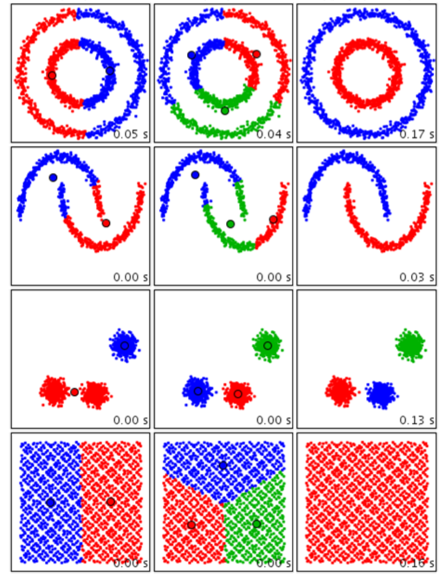
- Similarity by **distance** and/or **density**?
- Choice of **parameters** (i.e., range of K)

Other tasks

Also called «latent factors» or «hidden variables»

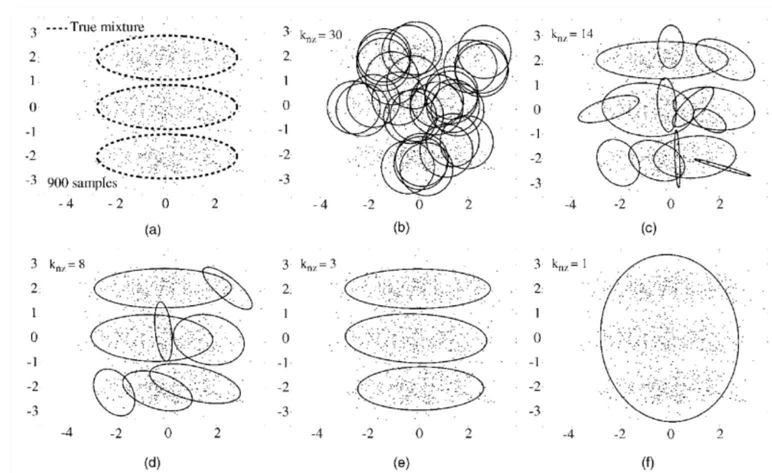
- Discovery of unobserved variables
- Dimensionality reduction
- Feature learning (e.g. autoencoders)
- Matrix completion (e.g. recommend, inpaint)
- Discovery of dependency structure in features (graph analysis)

not graph**ics**!



Left: Effect of density- vs. distance-based similarity. From left to right: K-Means ($K = 2$), K-Means ($K = 3$), DBSCAN ($eps = .1$, $\min = 3$)

Bottom: Problem of parameter choice in fitting a number of Gaussians to data. Top left to bottom right: True mixture (3), $K = 30, 14, 8, 3, 1$



3. WHAT IS LEARNABLE? (COMPUTATIONAL LEARNING THEORY)

What is learnable?

Previous findings

- **Any** target **function** f over an instance set X **is learnable**
- …given an expressive enough (deterministic) hypothesis space $\mathcal{H}$,
- …a large enough training set D_{train},
- …and stationarity of the distribution over X (i.e., instances in D_{train} and D_{test} are i.i.d.)

Better questions

- What size of $\boldsymbol{D_{train}}$ is **large enough**?
- Given that large enough training set, how well does the **training error** (empirical error over D_{train}) **predict generalizability**?

➔ This is the domain of computational learning theory (CLT)

PAC Learnability and VC Complexity

Measuring the complexity of infinite hypothesis spaces

Theoretical results (→ see appendix): Unrealistic but helpful

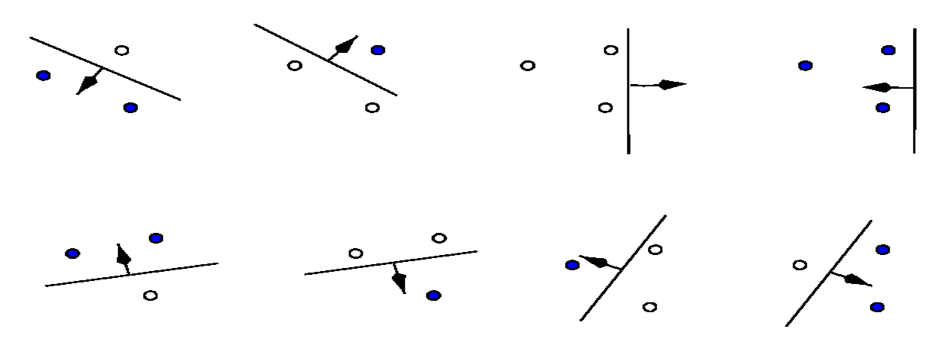
- **Sample complexity** bounds using $|\mathcal{H}|$ usually do a **substantial overestimate**
- The assumption that $\boldsymbol{f} \in \boldsymbol{\mathcal{H}}$ **is unrealistic** (would often need an unbiased $\mathcal{H}$)
- **But**: **Characterizing** learning problem **complexity** and **generalization** improvement **with** $\boldsymbol{N}$ **rocks**!

«[VC] dimension is a way of measuring the complexity of a class of functions by assessing **how wiggly** its members can be.»

http://www.svms.org/vc-dimension/

Measuring Vapnik-Chervonenkis (VC) dimension for infinite $\mathcal{H}$

- $h \in \mathcal{H}$ is "**shattering a set of instances** $S \in X$" $iif\ h$ can **partition** S **in any way** possible
- $VC(\mathcal{H}) := |\{S \in X | S\ is\ the\ \boldsymbol{largest\ subset\ of\ X\ shattered\ by}\ any\ \boldsymbol{h} \in \mathcal{H}\}|$
- $VC(\mathcal{H})$ can be used as an alternative measure of $|\mathcal{H}|$ to compute sample complexity



A 2d linear classifier (straight line) can shatter 3 points
➔ $VC(2D\ straight\ lines)\ =\ 3$ (but $|\mathcal{H}| = \infty$) .

VC Complexity contd.

What VC theory guarantees:
*«The **size of training set** required to ensure good generalisation **scales linearly with [VC dimension]** in the case of a consistent hypothesis».* [Cristianini and Shawe-Taylor, 2000]

Zurich University of Applied Sciences

Examples

- Provable: $VC(p - \dim \mathit{linear\ decision\ surface}) = p + 1$
- Provable: $VC(\mathit{conjunction\ of\ n\ Boolean\ literals}) = n$
- Provable: $VC(\mathit{multilayer\ neural\ net\ of\ n\ perceptrons}) = 2(p+1) \cdot n \cdot \log(e \cdot n)$
  - ➔ Doesn't hold for backpropagation training (sigmoidal units, inductive bias for small weights)
  - ➔ Sample complexity for NN should consider number **and** numerical size of weights!
  - ➔ Fits "Uncle Bernie's rule": $\mathrm{N} \approx 10w$ (w is the number of weights; other estimate: $N = w \log w$)

VC dimension and sample complexity of a learning problem

- Learning problem defined by concept C and hypothesis space $\mathcal{H}$, ε and δ as in appendix
- $\underbrace{max\left(\frac{1}{\varepsilon}\log\frac{1}{\delta}, \frac{VC(C)-1}{32\varepsilon}\right)}_{\text{Lower bound} \rightarrow \text{necessary } N} \le N \le \underbrace{\frac{1}{\varepsilon}\left(4\log_2\frac{2}{\delta} + 8VC(\mathcal{H}) \cdot \log_2\frac{13}{\varepsilon}\right)}_{\text{Upper bound} \rightarrow \text{sufficient } N}$

rather tight: only difference to order of lower bound is a factor of $\log\frac{1}{\varepsilon}$

required: $VC(C) \ge 2$, $\varepsilon < \frac{1}{8}$, $\delta < \frac{1}{100}$

➔ We **can characterize** ML **algorithms** regarding complexity **by** their **VC dimension**

Exercise: What's the point of CL theory?

Take some time with your neighbor and discuss:

- How can you determine the VC dimension of a practical ML method?
- How is $VC(\mathcal{H})$ related to the complexity of the learning *task*?
- Can you make statements on the complexity of a learning problem based on CLT results?
- Do you find concrete results on the web concerning sample complexity of algorithms you already know/use?

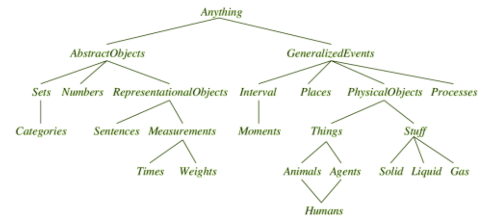
Zurich University
of Applied Sciences

Rehabilitation of ML
Relativizing previous pessimistic statements

"Fitting parameters of a function to a set of data
(data usually handcrafted, function chosen heuristically)"

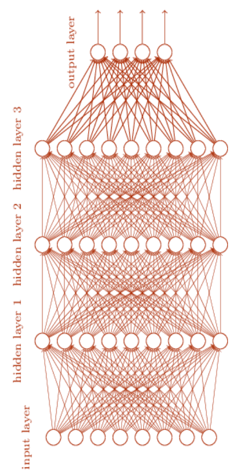
- Pure function fitting could be extended to a **feedback loop** (**active learning**): A "critic" reviews the result of a learner and operates a **simulation** to generate exactly the next data needed to enlighten current "blind spots"
- A model for practical "AI" (inspired by E. Mogenet, Google Research Europe):

AI Knowledge engineering (symbolic):
- ↓ Ontologies
- ↓ Logical inference



Gap to be filled by**: common sense DB**, **NLP**

Machine Learning (sub-symbolic):
- ↑ Hierarchical unsupervised learning
- ↑ Solid computer vision stack
- ↑ Images of the world



Rehabilitation of ML (contd.)
Relativizing previous pessimistic statements

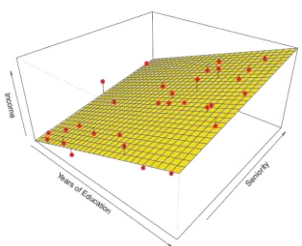
“Search through hypothesis space, driven by theory and empiricism (ultimately, ML depends on intelligent choice of the class of $\mathcal{H}$; $\mathcal{L}$ then optimizes the details)”

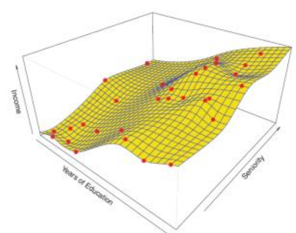
- Even if NFL states the *general* equivalence of all learners, **there might be a** single well-suited **learner for** the **subclass** of ***all practical* problems** encountered on earth
  ➔ Deep learning has shown some progress on the subclass of pattern recognition problems
- Two facts give hope: (a) bias-free learning is futile and (b) **good general learners** for all practical problems **do exist** (biological learners, especially humans)
  ➔ We **might discover general inductive biases** (i.e., learning algorithms) that are less domain-/problem-specific

application level

fundamental level

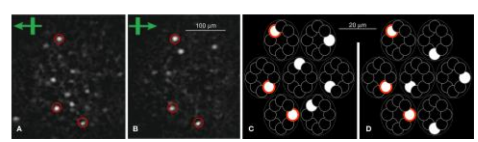
there’s a linear relationship between inputs & outputs





the hypothesis space is **smooth**

learn **sparse**, **distributed** representations



Review

- Remember what you've read for **P01** (in particular algorithm examples)!

- Classic (inductive supervised) ML: **approximate a function** of your choice using given tabulated data X with labels Y → **training**
- Long-term goal: (artificial) **intelligence → learn representation** (of data & function) automatically

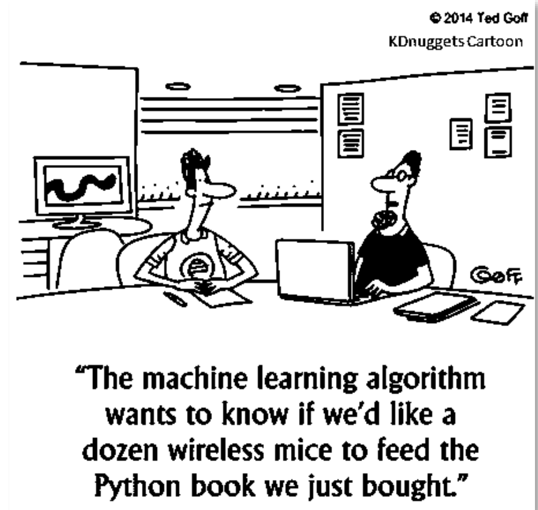
- The **inductive bias** guides the **search** through the chosen **hypothesis space**

- No single learner is best for all occasions (**no free lunch theorem**)

- **Computational learning theory** guarantees that the number of **needed training data** grows linearly with **VC dimension**

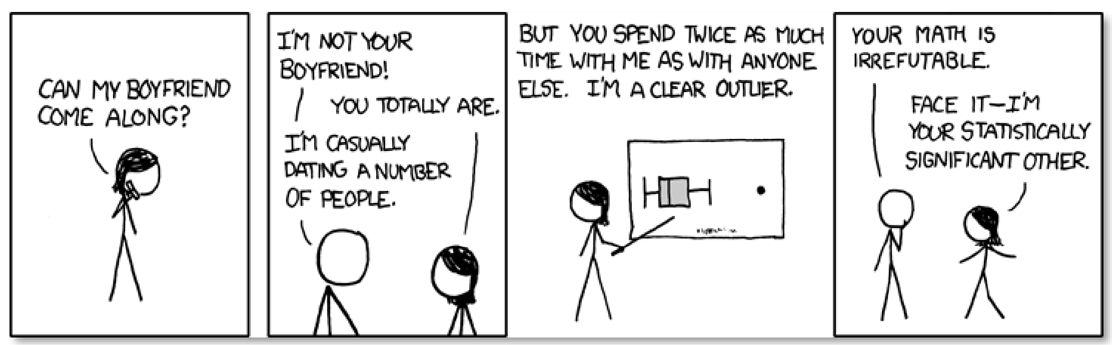
Review II

what it is



what it does

Machine Learning



how it works

APPENDIX

Further reading

See also: https://stdm.github.io/Some-places-to-start-learning-ai-ml/

Authors	Title	Year	Category	Focus
Lipton	A Critical Review of Recurrent Neural Networks for Sequence Learning	2015	Algorithms	Sequential supervised learning
Chandola et al.	Anomaly Detection - A Survey	2009	Algorithms	Anomaly detection
Mitchell	Machine Learning, Chapter 6	1997	Algorithms	Bayesian Methods
James, Witten, Hastie, Tibshirani	Introduction to Statistical Learning, 4th Printing, Chapter 8	2014	Algorithms	Tree-based methods
Duda, Hart, Stork	Pattern Classification, 2nd Edition, Chapter 9	2001	Fundamentals	More ML principles, classifier evaluation
James, Witten, Hastie, Tibshirani	Introduction to Statistical Learning, 4th Printing, Chapter 10	2014	Algorithms	Unsupervised learning
Mitchell	Machine Learning, Chapter 11+12	1997	Algorithms	Analytical (deductive) learning
Oza, Tumer	Classifier Ensembles - Select Real-World Applications	2008	Algorithms	Ensemble learning
LeCun, Bengio, Hinton	Deep Learning	2015	Algorithms	Deep Learning
Stanley, Miikulainen	Evolving Neural Networks through Augmenting Topologies	2002	Algorithms	Genetic algorithms train weights & structure of neural nets
LeCun et al.	Gradient-Based Learning Applied to Document Recognition	1998	Algorithms	Learning end to end
Hyärinen, Oja	Independent Component Analysis - A Tutorial	1999	Algorithms	Independent component analysis
Kaelbling et al.	Reinforcement Learning - A Survey	1996	Algorithms	Reinforcement learning
Breiman	Statistical Modeling - The Two Cultures	2001	Algorithms	Debating different approaches in statistics and machine learning (computer science)
Ke, Hoiem, Sukthankar	Computer Vision for Music Identification	2005	Applications	Audio fingerprinting
Ciresan et al.	Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images	2012	Applications	Medical image segmentation
Yu et al.	Feature engineering and classifier ensemble for KDD cup 2010	2010	Applications	Feature extraction & ensemble building
Viola, Jones	Robust Real-Time Face Detection	2004	Applications	Face detection
Reynolds, Rose	Robust Text-Independent Speaker Identification Using Gaussian Mixture Speaker Models	1995	Applications	Speaker recognition
Mordvintsev et al.	Inceptionism: Going Deeper into Neural Networks	2015	Applications	Synthesizing psychedelic images from Neural Networks
Jimmy Ba, Volodymyr Mnih, Koray Kavukcuoglu	Multiple Object Recognition with Visual Attention	2015	Applications	Automatic creation of image captions using deep learning
Chung … Hinton	Gated Feedback Recurrent Neural Network	2015	Algorithms	RNN architecture which learns
Dieleman et al.	Classifying plankton with deep neural nets	2015	Applications	Application of CNN to classify images from unbalanced small data sets
Bauckhage, Sifa	k-Maxoids Clustering	2015	Algorithms	Clustering
Doersch	A Tutorial on Variational Autoencoders	2016	Algorithms	Unsupervised learning
Goodfellow et al.	Generative Adversarial Nets	2014	Algorithms	Unsupervised learning

Zurich University
of Applied Sciences

Secrets of success

Do **program**, do **take notes** (yourself)!

Formulate goals, cross-connect knowledge

Know your learning style [Kolb]: do you need additional material to our deductive (general theory → example) approach to ML?

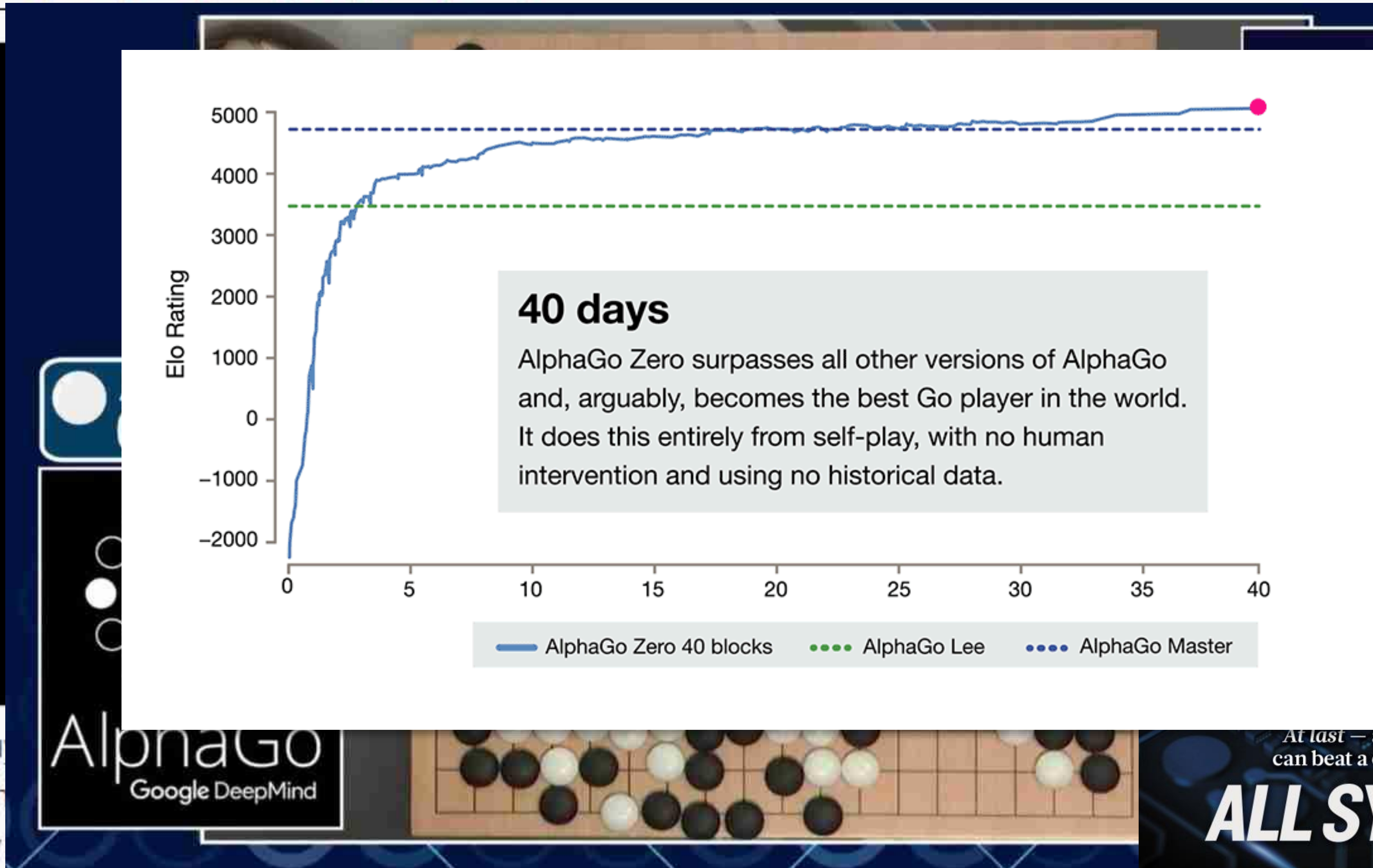
«Most of the things you *need* will *be brought* to you; most of the things you *want* you have to *go get*.» (Bill Johnson)

Use self study possibilities

Zurich University
of Applied Sciences

Google Acquires Artificial Intelligence Startup DeepMind For More Than $500M

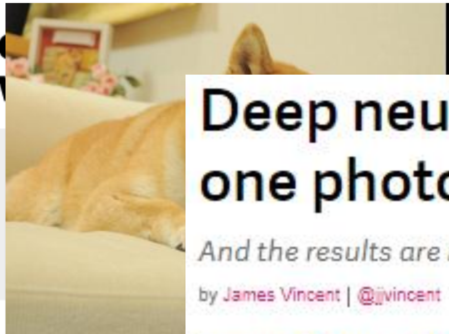
Posted Jan 26, 2014 by Catherine Shu (@catherineshu)



Google will bu… reports that th… in talks to buy… couldn't disclose deal terms.

The acquisition was originally confirmed by Google to Re/code.

MIT
Techn
Revie



Zurich University
of Applied Sciences

Deep neural networks can now transfer the style of one photo onto another

And the results are impressive

by James Vincent | @jjvincent | Mar 30, 2017, 1:53pm EDT

SHARE TWEET LINKEDIN

Original photo Reference photo Result

You've probably heard of an AI technique known as "style transfer" — or, if you haven't heard of it, you've seen it. The process uses neural networks to apply the look and feel of one image to another, and appears in apps like Prisma and Facebook. These style transfers, however, are stylistic, not photorealistic. They look good because they look like they've been painted. Now a group of researchers from Cornell University and Adobe have augmented

NOW TRENDING

Computing

Algorith
Artistic
Other In

A deep neural n
other images.

by Emerging Tech

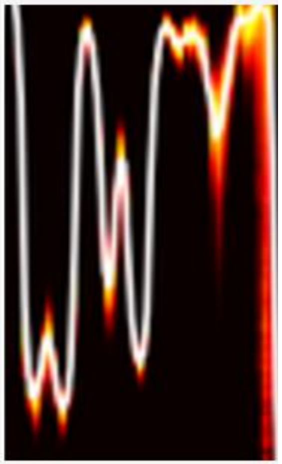
The nature of arti
of Vincent Van G
Edvard Munch's
humans recogni

Künstliche Intelligenz

WaveNet läs[...] Computersprache natürlich klingen

von Henning Steier / 12.9.201[...]

Die Google-Tochter DeepM[...]
macht auch Musik.



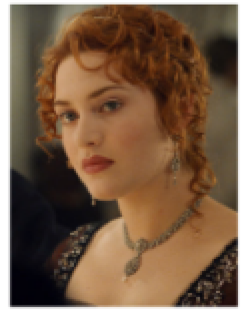
DeepMind lässt WaveNet Spra[...]

Die Google-Tochter Deep[...]
Spiel «Go» Schlagzeilen: [...]
einen der besten mensch[...]
Londoner Unternehmen [...]
erzeugt Sprache, die seh[...]
im Blogeintrag des Unter[...]
Massstab nimmt. Man ha[...]

Intro

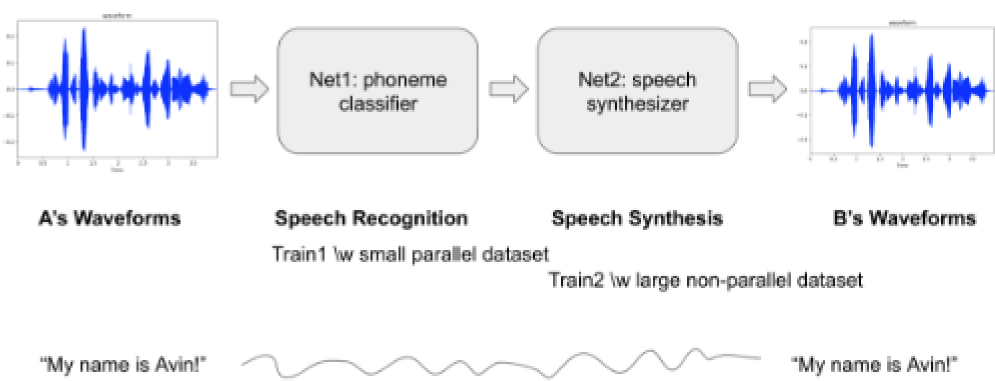
What if you could imitate a famous celebrity's voice or sing like a famous singer? This project started with a goal to convert someone's voice to a specific target voice. So called, it's voice style transfer. We worked on this project that aims to convert someone's voice to a famous English actress Kate Winslet's voice. We implemented a deep neural networks to achieve that and more than 2 hours of audio book sentences read by Kate Winslet are used as a dataset.

? ⇨



Model Architecture

This is a many-to-one voice conversion system. The main significance of this work is that we could generate a target speaker's utterances without parallel data like <source's wav, target's wav>, <wav, text> or <wav, phone>, but only waveforms of the target speaker. (To make these parallel datasets needs a lot of effort.) All we need in this project is a number of waveforms of the target speaker's utterances and only a small set of <wav, phone> pairs from a number of anonymous speakers.

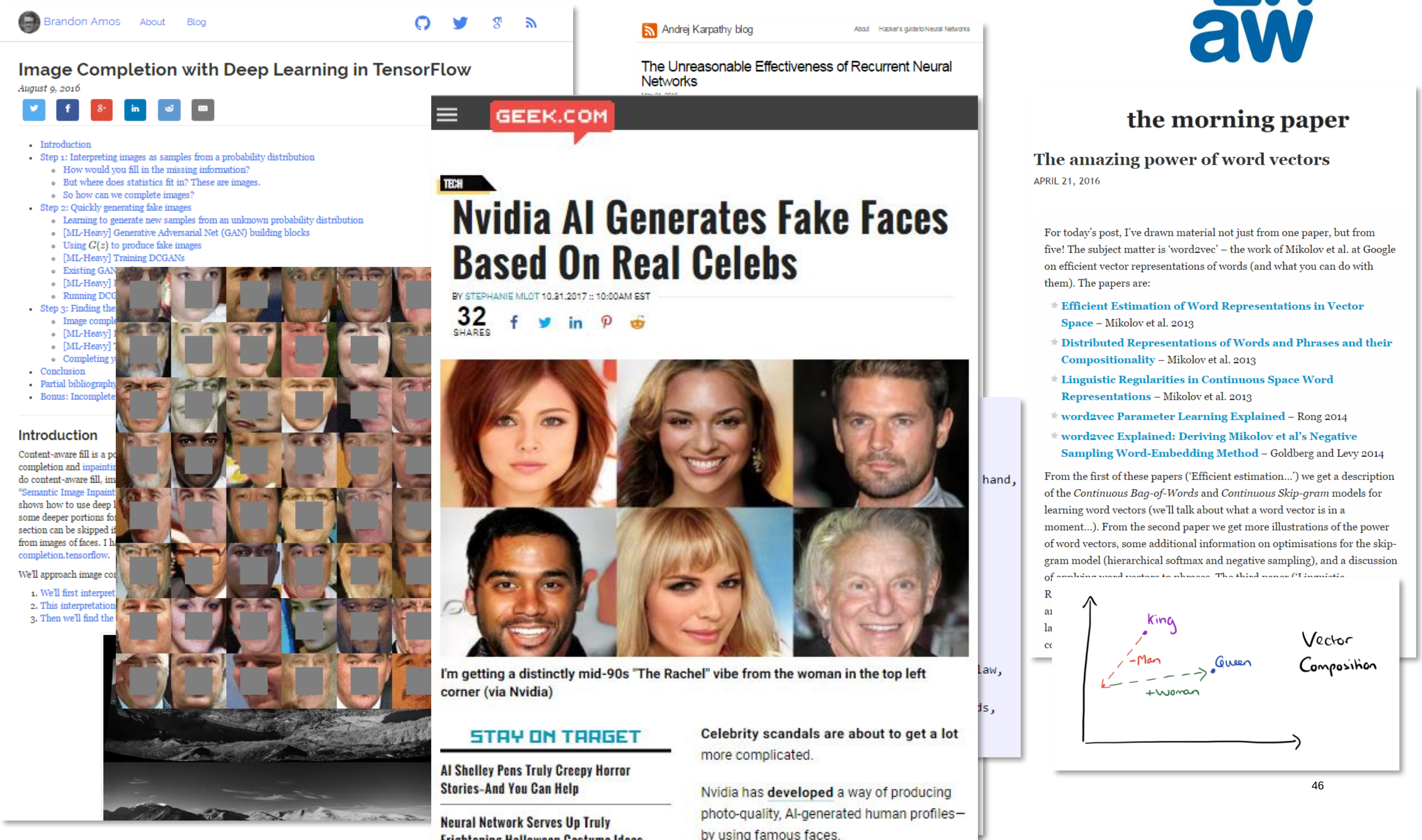


Zurich University of Applied Sciences

[...]nerierte Sprache
[...]s Texteingabe»

[...]nerierte Musik
[...]ne Inhaltsvorgabe»

…und die Liste liesse sich fortsetzen!

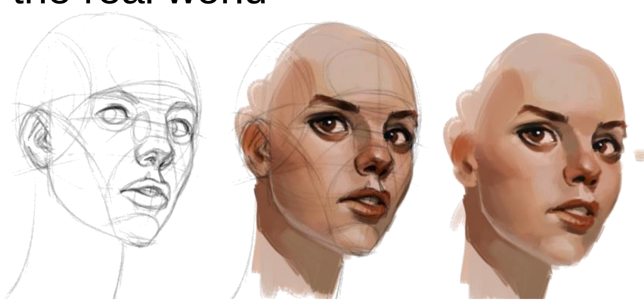


On modeling and abstraction

Quoted from *AIMA*, p. 68-69, sec. 3.1.2

- *A* ***model*** [is] an abstract mathematical description […] and not the real thing
- The process of removing detail from a representation is called **abstraction**
- The abstraction is ***valid*** if we can expand any abstract solution into a solution in the more detailed world
- The abstraction is ***useful*** if carrying out each of the actions in the solution is easier than the original problem
- The choice of a **good abstraction** thus involves **removing as much detail as possible while retaining validity** and **ensuring that the abstract actions are easy** to carry out

➔ Were it not for the ability to construct useful abstractions, machine learning solutions would be completely swamped by the real world



Zurich University
of Applied Sciences

An example
…of the futility of bias-free learning

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Warm	Same	Yes
2	Sunny	Warm	High	Strong	Warm	Same	Yes
3	Rainy	Cold	High	Strong	Warm	Change	No
4	Sunny	Warm	High	Strong	Cool	Change	Yes

Example	*Sky*	*AirTemp*	*Humidity*	*Wind*	*Water*	*Forecast*	*EnjoySport*
1	Sunny	Warm	Normal	Strong	Cool	Change	Yes
2	Cloudy	Warm	Normal	Strong	Cool	Change	Yes
3	Rainy	Warm	Normal	Strong	Cool	Change	No

Training set D_{train} (left) and unseen test set D_{test} (right) for the concept "EnjoySport" (from [Mitchell, 1997, Ch. 2]).

- The CandidateElimination algorithm finds all hypotheses $V = \{h \in \mathcal{H} | h\ consistent\ with\ D_{train}\}$
  - Classifies an unknown instance positive iif it is classified positive by all $h \in V$
  - Searches by enumerating the sets of **most general** and **most specific** consistent hypotheses (V_g, V_s)

- Suppose $\mathcal{H}$ includes **conjunctions of constraints** on all features (specific values & wildcards)
  - ➔ *CandidateElimination* learns $V_s = \{< Sunny, Warm, ?, Strong, ?, ? >\}$, $V_g = \{< Sunny, ?, ?, ?, ?, ? >, < ?, Warm, ?, ?, ?, ? > \}$
  - ➔ No $h \in V$ classifies all 3 instances in D_{test} correctly (disjunction needed)

- Change $\mathcal{H}$ so it allows arbitrary **combinations** of **conjunction, disjunction and negation**
  - ➔ $\mathcal{H}$ now contains all possible concepts → it is unbiased
  - ➔ *CandidateElimination* will just memorize D_{train} ($V_g = V_s =$ instances themselves)
  - ➔ No generalization possible!

Inductive bias of *CandidateElimination*: The concept can be represented in its (limited) $\mathcal{H}$.

PAC Learnability

Framework for characterizing learners over finite $|\mathcal{H}|$

with probability $1-\delta$

with true error < ε

A learning algorithm is said to learn probably approximately correct (PAC) iif

- We can find N such that **after seeing N training examples**, all consistent $h \in \mathcal{H}$ will be approximately correct with high probability after reasonable computational time
- That is: Computational effort & needed training samples grow only polynomial with $\frac{1}{\varepsilon}$ & $\frac{1}{\delta}$

Advantages if one can show an algorithm to be a PAC learner

- *«Any hypothesis that is consistent with a sufficiently large set of training examples is* ***unlikely to be seriously wrong****»* [Russell & Norvig, 2010, Ch. 18.5]
- There are provable upper bounds on the sample complexity of learners over specific $\mathcal{H}$
  - E.g., $N \geq \frac{1}{\varepsilon}\left(\ln\frac{1}{\delta} + \ln|\mathcal{H}|\right)$ for any consistent PAC learning algorithm

"**Consistent**" learners have 0 training error. Replace $\frac{1}{\varepsilon}$ with $\frac{1}{2\varepsilon^2}$ for "**agnostic**" learners ($f \notin \mathcal{H}$); replace $|\mathcal{H}|$ with $c \cdot 2^p$ for an **unbiased** $\mathcal{H}$ with p-dimensional features (c is a constant).

Example: *CandidateElimination* (see appendix)

- 96 distinct instances possible for *EnjoySports* task
- $|\mathcal{H}| = 973$ (just conjunctions), let $\varepsilon = 0.01$, $\delta = 0.95$
  ➔ N should be greater than 693

➔ $|\mathcal{H}|$ needs to be restricted to allow for reasonable N

E.g. via inductive bias, regularization
→ more in V06.