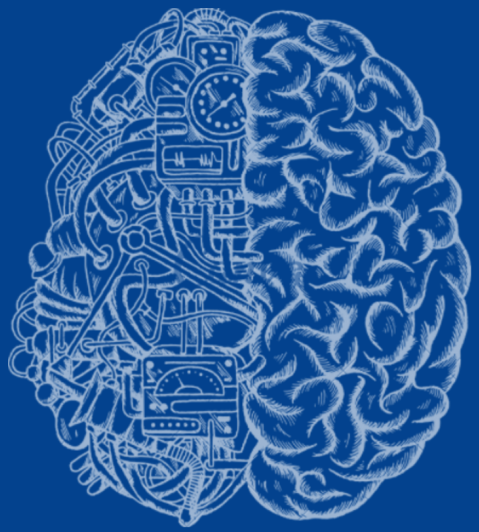
AARHUS
UNIVERSITY
SCHOOL OF ENGINEERING

LESSON 1: Introduction

CARSTEN EIE FRIGAARD
PETER AHRENDT

AUTUMN 2020



Undervisere

Carsten Eie Frigaard:
kursusholder,
rum: E311,
email: `cef@ase.au.dk`

Peter Ahrendt
kursusholder,
rum: E308,
email: `pah@ase.au.dk`

Henrik Daniel Kjeldsen
hjælpelærer,
email: `hdk@ase.au.dk`

Eksamen

Formel beskrivelse fra kursuskataloget

Prøveform

- Undervisningsdeltagelse.

Bedømmelse

- Godkendt/Ikke godkendt, ingen censur.

Eksamen

Formel beskrivelse fra kursuskataloget

Prøveform

- Undervisningsdeltagelse.

Bedømmelse

- Godkendt/Ikke godkendt, ingen censur.

Bemærkninger

- " I løbet af kurset skal et antal obligatoriske opgaver afleveres og der skal deltages i et antal obligatoriske præsentationer.

  Bedømmelsen af kurset sker på baggrund af én samlet vurdering af de afleverede opgaver og præsentationer, hvor der vil blive lagt vægt på, om den studerende opfylder punkterne i kvalifikationsbeskrivelsen.

  Bedømmelsen foretages kun af eksaminator (underviser). "

Eksamen

Formel beskrivelse fra kursuskataloget

Prøveform

- Undervisningsdeltagelse.

Bedømmelse

- Godkendt/Ikke godkendt, ingen censur.

Bemærkninger

- " I løbet af kurset skal et antal obligatoriske opgaver afleveres og der skal deltages i et antal obligatoriske præsentationer.

  Bedømmelsen af kurset sker på baggrund af én samlet vurdering af de afleverede opgaver og præsentationer, hvor der vil blive lagt vægt på, om den studerende opfylder punkterne i kvalifikationsbeskrivelsen.

  Bedømmelsen foretages kun af eksaminator (underviser). "

Reeksamen

- " Reeksamen: Næste ordinære eksamenstermin. Der skal afleveres nye opgaver og præsentationer. "

Eksamen

Afleveringer og evalueringer

Eksamensform, godkendelsesfag via:

- et sæt obligatoriske skriftlige gruppe-opgaver med afleveringsdeadlines (O1/O2/O3/O4),
- en poster-session, med aflevering af poster og mundtlig præsentation af poster,
- en mundtlig gennemgang af den sidste afleveringsopgave (O4/slut-journal) med alle medlemmer i ITMAL gruppen, samt evaluering af hver gruppemedlems bidrag.

=> Endelig godkendelse af kurset sker på én samlet vurdering af de tre punkter ovenfor.

Opgaveafleveringer: O1, O2, O3 og O4

- O1: Opgavesæt fra L01+L02+.. (se Blackboard)
- O2: Opgavesæt fra ..
- O3: Opgavesæt fra ..

Opgaveafleveringer: O1, O2, O3 og O4

- O1: Opgavesæt fra L01+L02+.. (se Blackboard)
- O2: Opgavesæt fra ..
- O3: Opgavesæt fra ..
- O4: slut-journal, et mini-projekt:
  - For the final journal, you must design and implement a full machine learning system. You have relative free hands...

    Criterions [extract]:
    - Data must be split in a training-test set...
    - Your machine learning algorithm must be described in depth...
    - The system must be evaluated via a suitable performance metric...

NOTE$_0$: Afleveringsformat i PDF.
NOTE$_1$: O4 vil blive specificeret på BB.
NOTE$_2$: O4 konflikt med BA projekter, BA deadline 16/12?

Læringsmål

- ITMAL generelt:
  - **Redegøre** for de væsentligste begreber i machine learning terminologi samt principperne i en machine learning pipeline.
  - **Anvende** metoder til analyse af data, bl.a. med henblik på valg af machine learning model.

Læringsmål

- ITMAL generelt:
  - **Redegøre** for de væsentligste begreber i machine learning terminologi samt principperne i en machine learning pipeline.
  - **Anvende** metoder til analyse af data, bl.a. med henblik på valg af machine learning model.
- ITMAL i relation til praktiske projekter:
  - **Anvende** udvalgte machine learning teknikker i praktiske opgaver og projekter.
  - **Anvende** udvalgte kodebiblioteker (frameworks) og udviklingsværktøjer til machine learning.

Læringsmål

- ITMAL generelt:
  - **Redegøre** for de væsentligste begreber i machine learning terminologi samt principperne i en machine learning pipeline.
  - **Anvende** metoder til analyse af data, bl.a. med henblik på valg af machine learning model.
- ITMAL i relation til praktiske projekter:
  - **Anvende** udvalgte machine learning teknikker i praktiske opgaver og projekter.
  - **Anvende** udvalgte kodebiblioteker (frameworks) og udviklingsværktøjer til machine learning.
- ML Data og algoritmer:
  - **Beskrive** betydningen af datakvalitet i machine learning, samt anvende udvalgte databehandlings-teknikker til at forbedre kvaliteten af datagrundlaget.
  - **Sammenligne og vurdere** forskellige algoritmer og teknikkers anvendelighed i forbindelse med praktiske projekter.

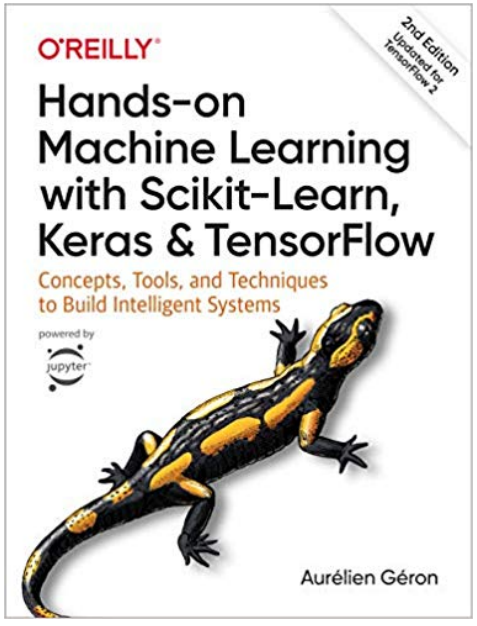
Læringsmål

Indhold på kursushjemmesiden

- Generelle machine learning koncepter.
- Machine learning frameworks (biblioteker til Python).
- Introduktion til diverse specifikke machine learning algoritmer, herunder f.eks. lineær/logistisk regression og neurale netværk.
- Dimensionsreduktion og visualisering.
- Clustering
- Datakvalitet og Big data.

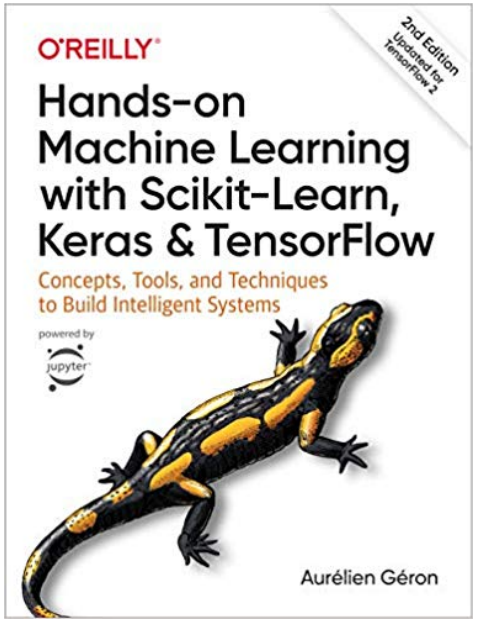
Litteratur

”**Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow**”, Aurélien Géron, O’Reilly, 2019, (Second Edition)



Litteratur

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow",
Aurélien Géron, O'Reilly, 2019, (Second Edition)



Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow
by Aurélien Géron

Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 9547

O'Reilly books may be purchased for educational, business, or sales promotional use. Onl
also available for most titles (*http://oreilly.com*). For more information, contact our corpor
sales department: 800-998-9938 or *corporate@oreilly.com*.

Editors: Rachel Roumeliotis and Nicole Tache
Production Editor: Kristen Brown
Copyeditor: Amanda Kersey
Proofreader: Rachel Head
Indexer: Judith McConville
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

September 2019: Second Edition

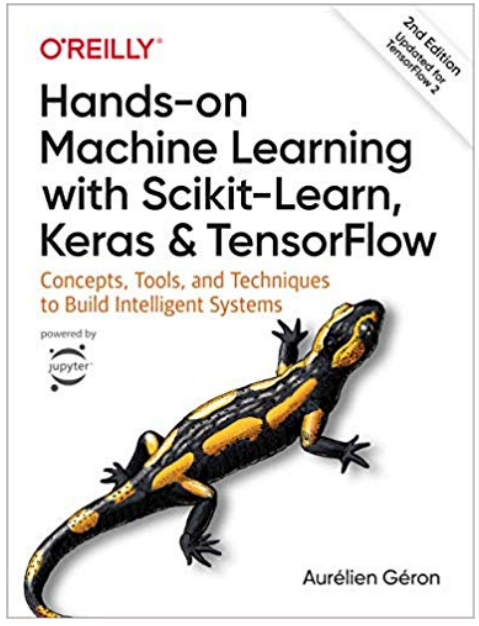
Revision History for the Second Edition
2019-09-05: First Release
2019-10-11: Second Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781492032649* for release details.

- [HOML]→ udtales som (Brian) Holm!
- Ref. til sidetal er for 2.Ed / *Second Release*.

Litteratur

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow", Aurélien Géron, O'Reilly, 2019, (Second Edition)



Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow
by Aurélien Géron

Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 9547

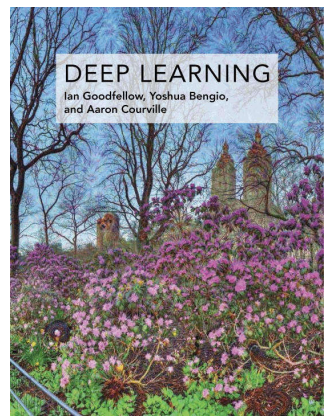
O'Reilly books may be purchased for educational, business, or sales promotional use. Onl
also available for most titles (*http://oreilly.com*). For more information, contact our corpor
sales department: 800-998-9938 or *corporate@oreilly.com*.

Editors: Rachel Roumeliotis and Nicole Tache
Production Editor: Kristen Brown
Copyeditor: Amanda Kersey
Proofreader: Rachel Head
Indexer: Judith McConville
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

September 2019: Second Edition

Revision History for the Second Edition
2019-09-05: First Release
2019-10-11: Second Release

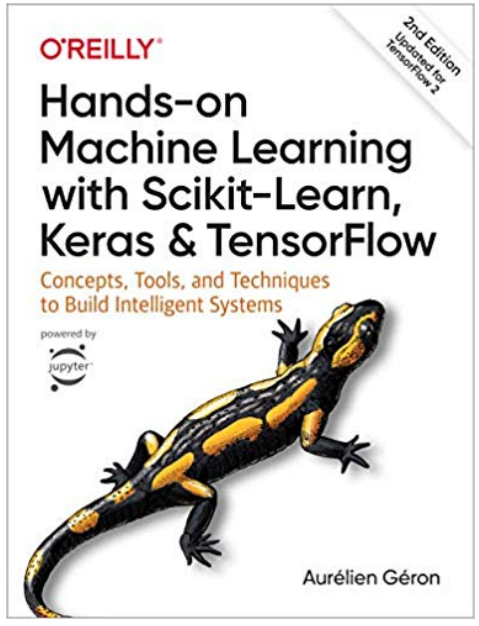
See *http://oreilly.com/catalog/errata.csp?isbn=9781492032649* for release details.



- [HOML]→ udtales som (Brian) Holm!
- Ref. til sidetal er for 2.Ed / *Second Release*.
- Plus yderligere materiale (brug links i BB).

Litteratur

"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow", Aurélien Géron, O'Reilly, 2019, (Second Edition)



Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow
by Aurélien Géron

Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 9547

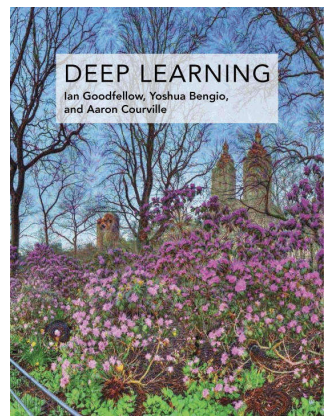
O'Reilly books may be purchased for educational, business, or sales promotional use. Onl
also available for most titles (*http://oreilly.com*). For more information, contact our corpor
sales department: 800-998-9938 or *corporate@oreilly.com*.

Editors: Rachel Roumeliotis and Nicole Tache
Production Editor: Kristen Brown
Copyeditor: Amanda Kersey
Proofreader: Rachel Head
Indexer: Judith McConville
Interior Designer: David Futato
Cover Designer: Karen Montgomery
Illustrator: Rebecca Demarest

September 2019: Second Edition

Revision History for the Second Edition
2019-09-05: First Release
2019-10-11: Second Release

See *http://oreilly.com/catalog/errata.csp?isbn=9781492032649* for release details.



- [HOML]→ udtales som (Brian) Holm!
- Ref. til sidetal er for 2.Ed / *Second Release*.
- Plus yderligere materiale (brug links i BB).

Lektionsplan

Uge	Dato	Lektion	Emne	Opgave	Underviser	Kommentar
36	03/09-2020	L01	Intro		CEF	
37	10/09-2020	L02	Klassifikation		CEF	
38	17/09-2020	L03	End-to-end ML		CEF	
39	24/09-2020	L04	Regression	O1 (25/9)	PAH	
40	01/10-2020	L05	Data analyse		PAH	
41	08/10-2020	L06	Neurale netværk (NN)		PAH+CEF	
42	15/10-2020					Efterårsferie (ingen undervisning.)
43	22/10-2020	L07	Træning og generalisering	O2 (23/10)	CEF	
44	29/10-2020	L08	Regularisering og søgning		CEF	
45	05/11-2020	L09	Deep learning (CNN)		CEF	
46	12/11-2020	L10	Probabilistiske modeller	O3 (13/11)	PAH	
47	19/11-2020	L11	Unsupervised learning I (PCA)		PAH	
48	26/11-2020	L12	Unsupervised learning II (Kmeans,GMM)		PAH	
49	03/12-2020	L13	O4 projekt			
50	10/12-2020	L14	O4 projekt Poster-session	O4 (11/12) Poster(9/12)		
51						Mundtlige præsentationer af O4

ITMAL Nomenklatur

[HOML]:	Hands-On Machine Learning bog, aka (B.)Holm.
[GITHOML]:	Git repository for [HOML].
[GITMAL]:	Git repository for ITMAL kursus opgaver, (bruges evt. slet ikke)
[G]:	ITMAL gruppe, med tre studerende, (evt. to/fire).
[SG]:	ITMAL super-gruppe, ved nogle af opgaverne.
[O1]:	opgavesæt 1, osv. (O2/O3/O4).
[L01]:	Lektion 1, osv.

NOTE: se fuld liste på *'BB | Kursusinfo | Kursusforkortelser'.*

COVID-19 Quiz

Skriv i Blackboard Chat'en
(*BB | Online undervisning | ITMAL Chat*)

END Kursus intro/
BEGIN Python intro

python Introduction

- Python is an **interpreted** high-level programming language for general-purpose programming. Created by **Guido van Rossum** and first released in 1991, Python has a design philosophy that emphasizes **code readability**, notably using significant whitespace. It provides constructs that enable clear programming on both small and large scales.
- Python features a **dynamic type system and automatic memory management**. It supports multiple programming paradigms, including **object-oriented**, **imperative**, **functional and procedural**, and has a large and comprehensive standard library.
- Python interpreters are available for many operating systems.

[https://en.wikipedia.org/wiki/Python_(programming_language)]

Anaconda and Jupyter Introduction

- **Anaconda**: a python distribution [`https://www.anaconda.com`].
- **Jupyter notebook**: interactive python development environment (GUI IDE), distributed with the Anaconda package.
- Jupyter is an anagram of: Julia, Python, and R.

Anaconda and Jupyter Introduction

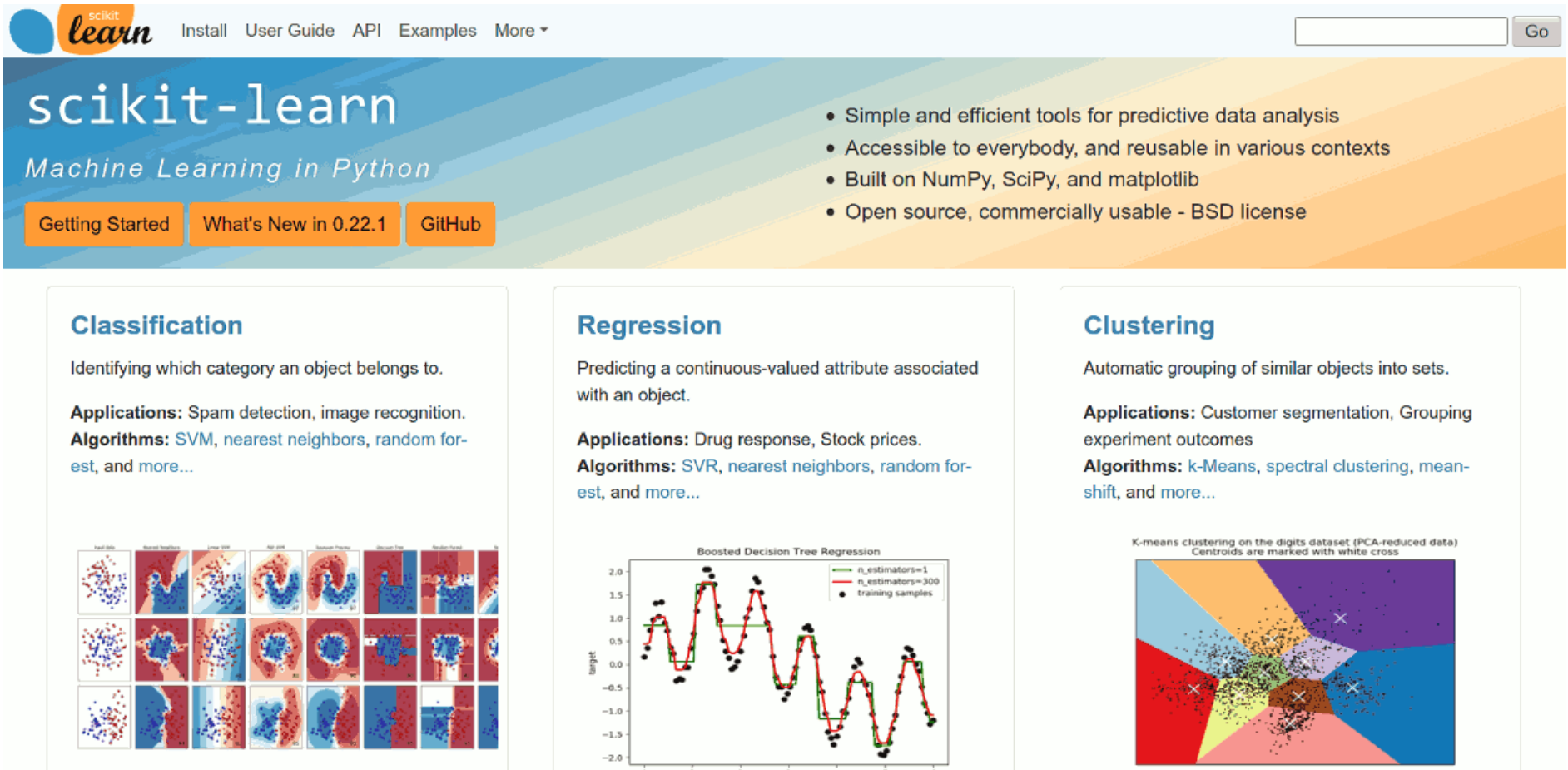
- **Anaconda**: a python distribution [https://www.anaconda.com].
- **Jupyter notebook**: interactive python development environment (GUI IDE), distributed with the Anaconda package.
- Jupyter is an anagram of: Julia, Python, and R.
- Jupyter notebook method:
  - √ polyglot environment, mixing source code, markdown test and formulas (LaTeX),
  - √ interactive/explorativt/trial-and-error environment,
  - ÷ not good at source-code level debugging.

Anaconda and Jupyter Introduction

- **Anaconda**: a python distribution [https://www.anaconda.com].
- **Jupyter notebook**: interactive python development environment (GUI IDE), distributed with the Anaconda package.
- Jupyter is an anagram of: Julia, Python, and R.
- Jupyter notebook method:
  - √ polyglot environment, mixing source code, markdown test and formulas (LaTeX),
  - √ interactive/explorativt/trial-and-error environment,
  - ÷ not good at source-code level debugging.
- Other IDE's:
  - Spyder (Anaconda),
  - VSCode (Microsoft),
  - and many others...

Scikit-learn Introduction

- Scikit-learn: a framework (API + website) for machine Learning in python.
- `http://scikit-learn.org`
- `git@github.com:scikit-learn/scikit-learn.git`



Vores videnskabelige framework

Sat sammen...

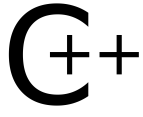
Gode hjælpe og dokumentations-systemer..

Vores videnskabelige framework

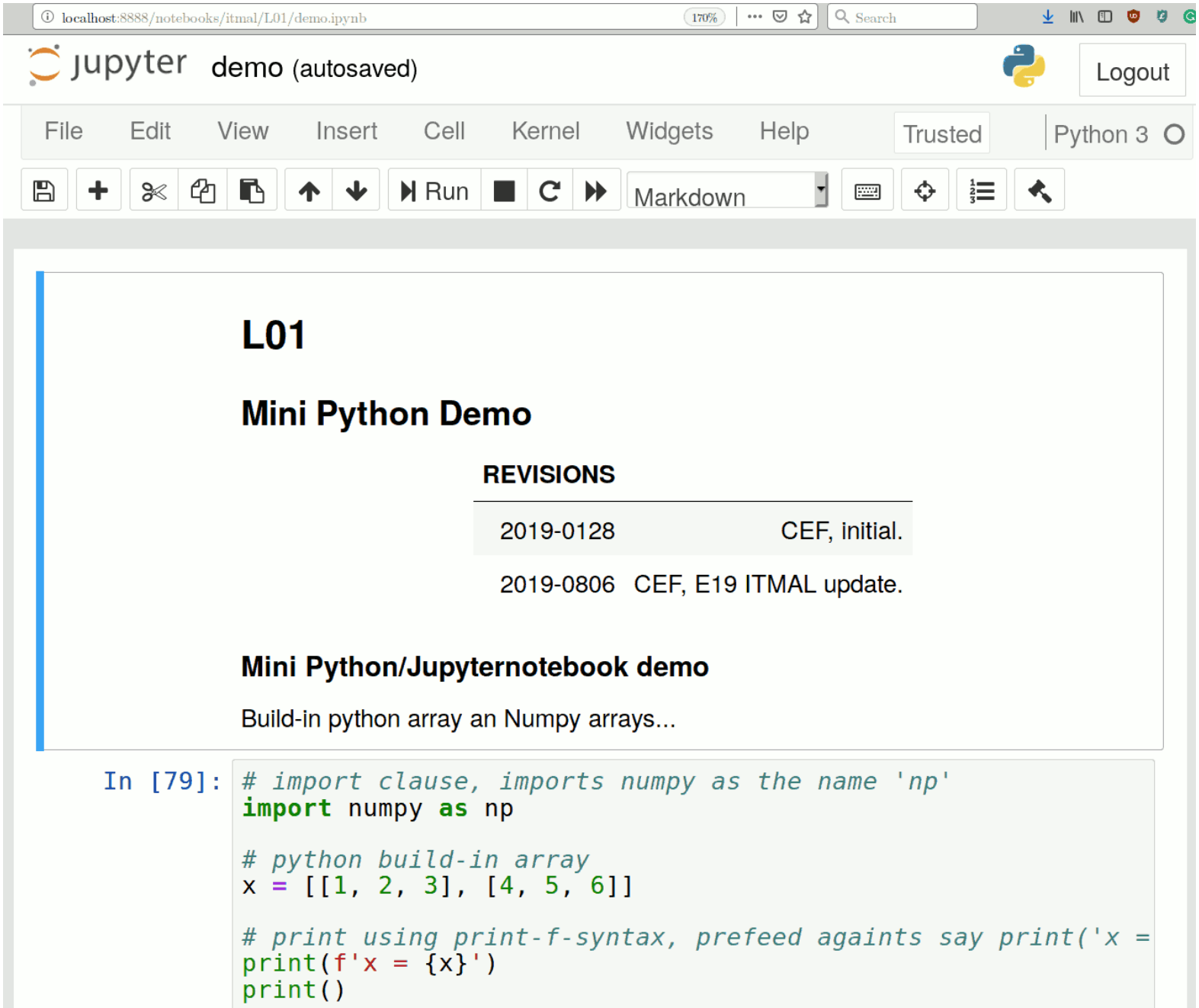
Sat sammen...

Gode hjælpe og dokumentations-systemer..

Alternativer kunne være...



Anaconda and Jupyter Demo



Anaconda and Jupyter Demo: Highlights...

- Polyglot miljø:
  - lidt ala Matlab IDE,
  - markdown (HTML+LaTeX)-og-Python-i-een = polyglot,
  - alt kører i browser, lokalt eller på server.

Anaconda and Jupyter Demo: Highlights...

- Polyglot miljø:
  - lidt ala Matlab IDE,
  - markdown (HTML+LaTeX)-og-Python-i-een = polyglot,
  - alt kører i browser, lokalt eller på server.
- Quickstart:
  - åbn via `http://localhost:8888` (efter launch),
  - ENTER på celle: editer celle,
  - CTRL+ENTER: kør celle,
  - SHIFT+TAB: hjælp på funktion,
  - TAB: tab-completion.

Anaconda and Jupyter Demo: Highlights...

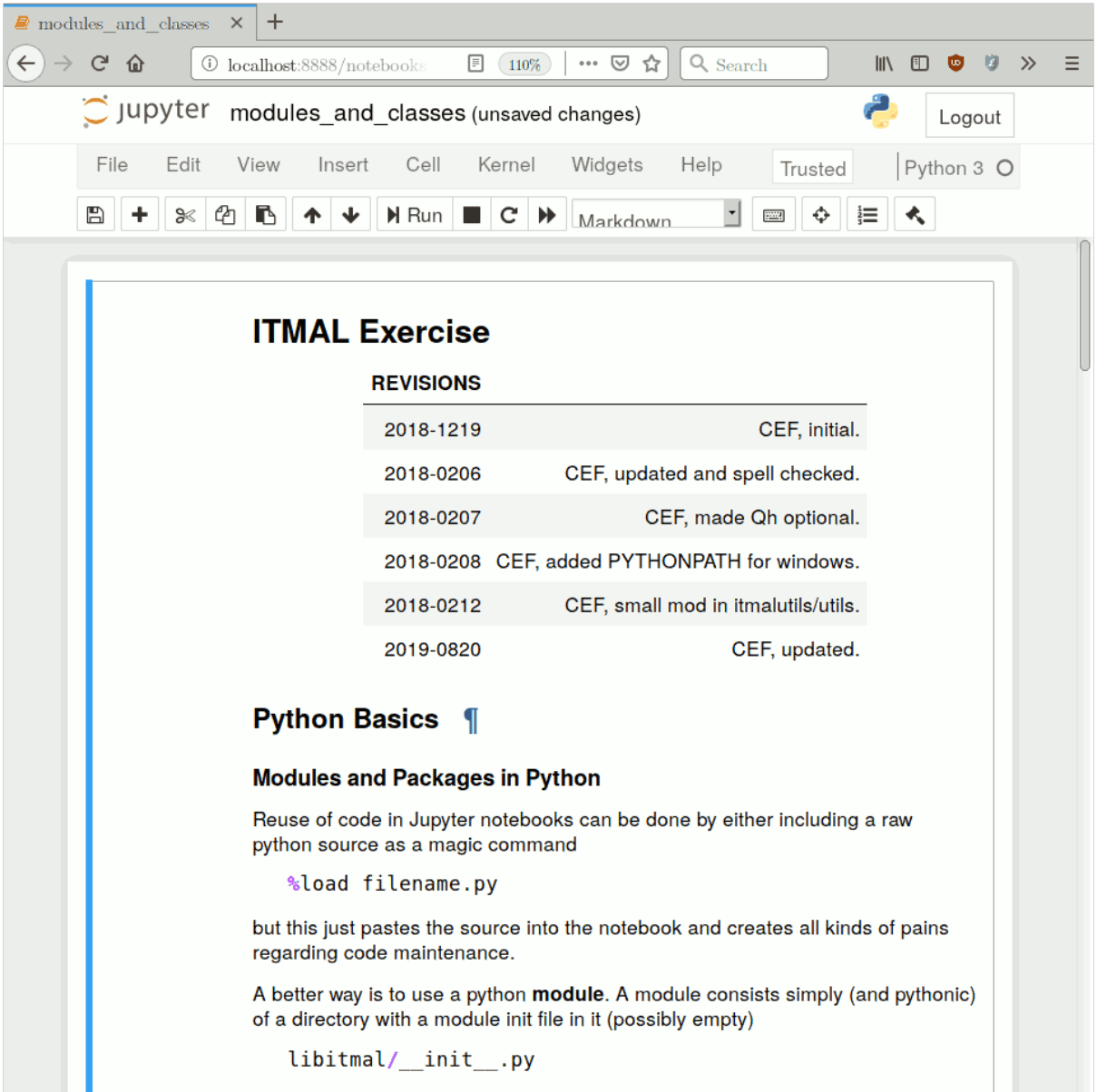
- Polyglot miljø:
  - lidt ala Matlab IDE,
  - markdown (HTML+LaTeX)-og-Python-i-een = polyglot,
  - alt kører i browser, lokalt eller på server.
- Quickstart:
  - åbn via `http://localhost:8888` (efter launch),
  - ENTER på celle: editer celle,
  - CTRL+ENTER: kør celle,
  - SHIFT+TAB: hjælp på funktion,
  - TAB: tab-completion.
- Magics:
  - nulstil vars: `%reset -f`,
  - inline plots: `%matplotlib inline`.

Anaconda and Jupyter Demo: Highlights...

- Polyglot miljø:
  - lidt ala Matlab IDE,
  - markdown (HTML+LaTeX)-og-Python-i-een = polyglot,
  - alt kører i browser, lokalt eller på server.
- Quickstart:
  - åbn via `http://localhost:8888` (efter launch),
  - ENTER på celle: editer celle,
  - CTRL+ENTER: kør celle,
  - SHIFT+TAB: hjælp på funktion,
  - TAB: tab-completion.
- Magics:
  - nulstil vars: `%reset -f`,
  - inline plots: `%matplotlib inline`.
- Hints:
  - Pas på globale vars (igen scopes ml. `.ipynb` celler),
  - Brug menu *'Help'* og
    find shortcuts i *'open command palette'*n,
  - Hvis du er C++ haj: alt er anderledes!

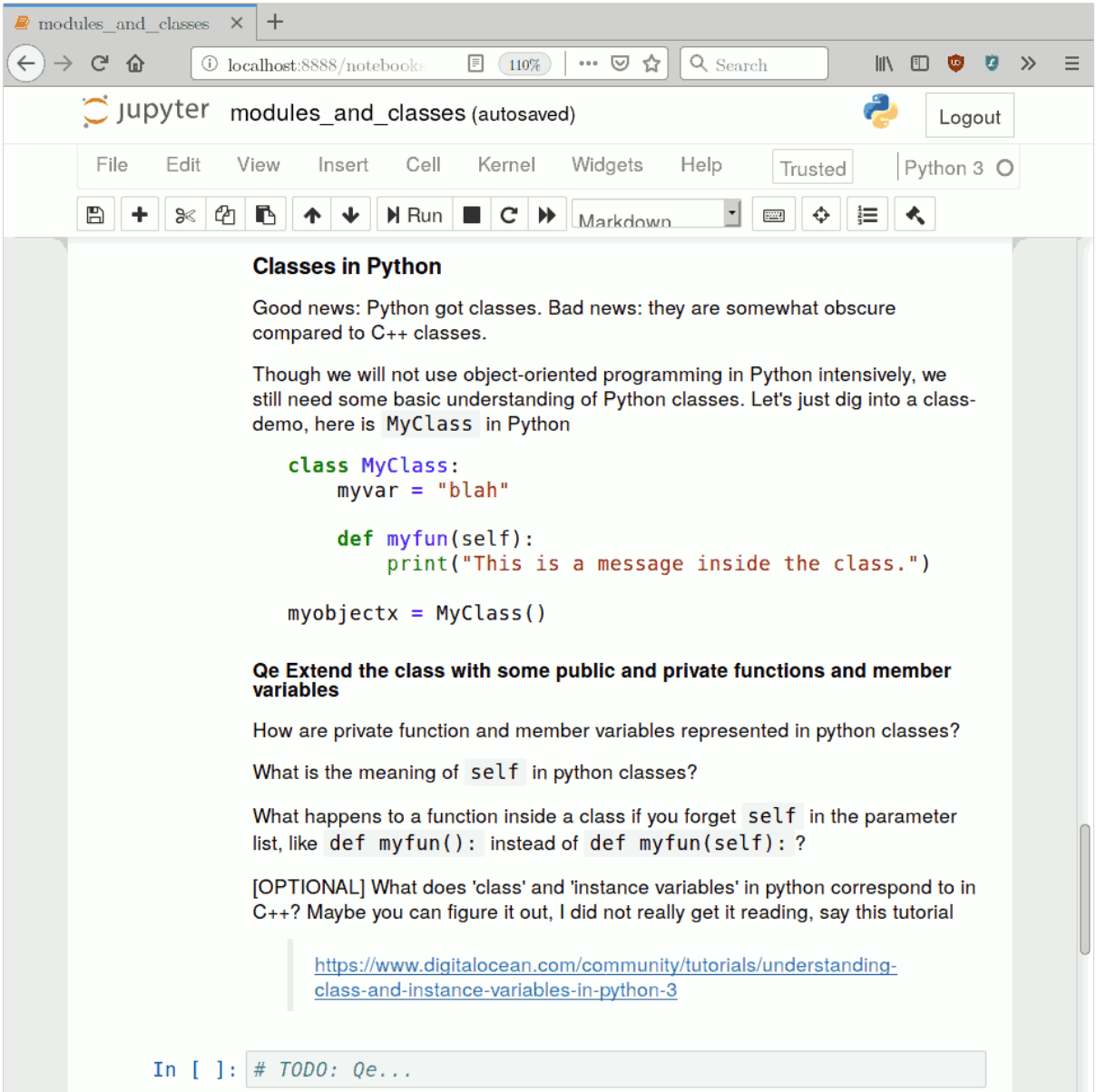
Q: L01/modules_and_classes.ipynb

Modules and Packages...

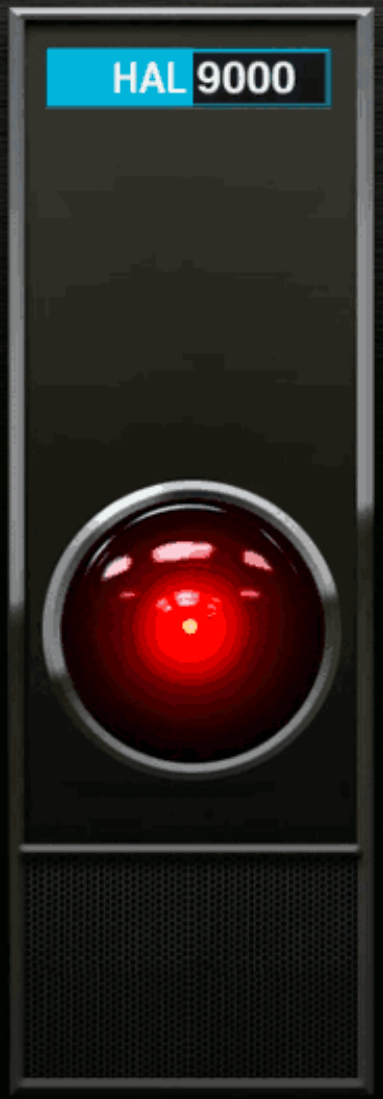


Q: L01/modules_and_classes.ipynb

Python classes...

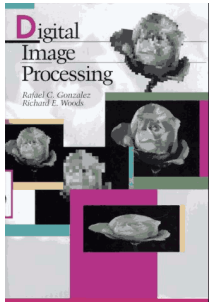


END Python intro/BEGIN ML intro



Klassisk maskinlæring årgang 1992

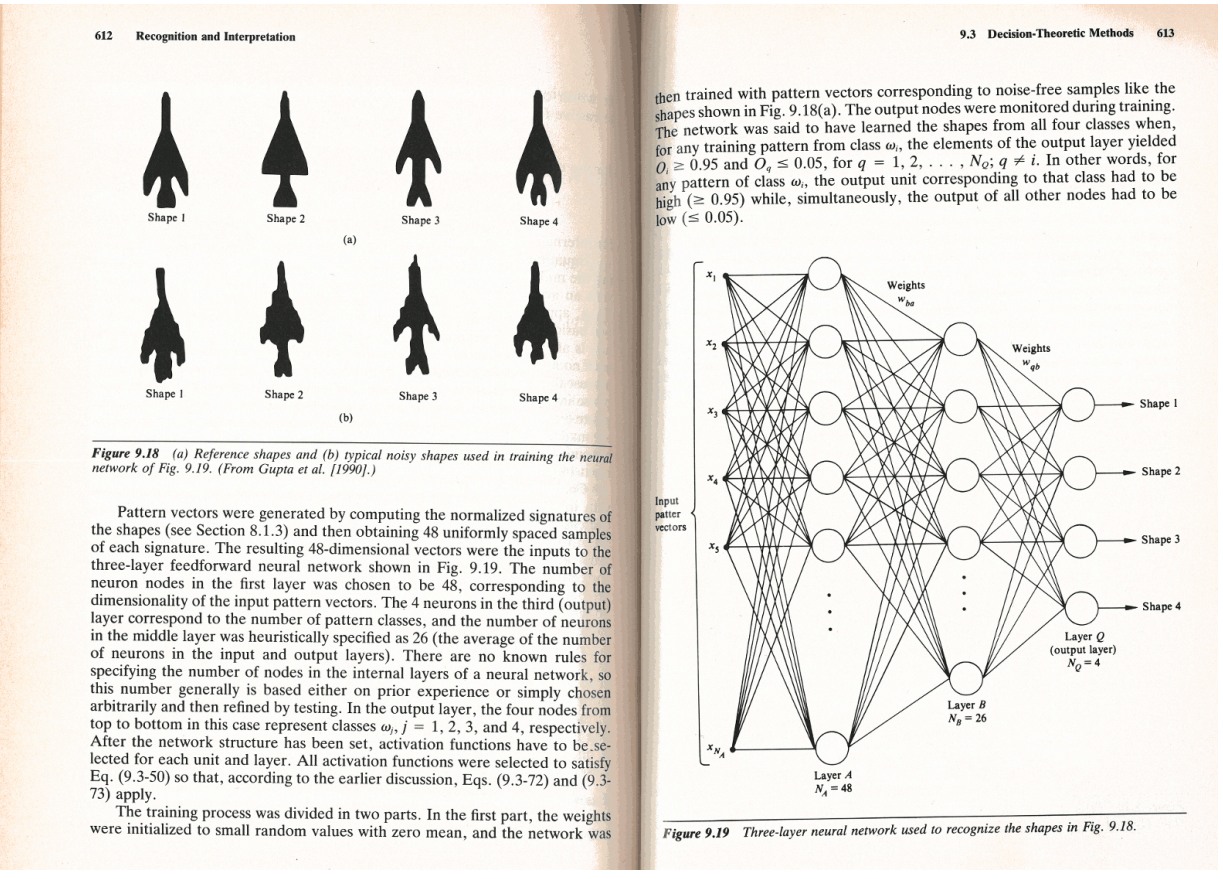
Pattern recognition, machine vision, neural networks...



Digital Image Processing, Gonzalez and Woods, 1992

Klassisk maskinlæring årgang 1992

Pattern recognition, machine vision, neural networks...



612 Recognition and Interpretation

Figure 9.18 *(a) Reference shapes and (b) typical noisy shapes used in training the neural network of Fig. 9.19. (From Gupta et al. [1990].)*

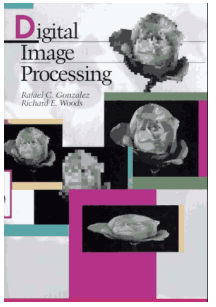
Pattern vectors were generated by computing the normalized signatures of the shapes (see Section 8.1.3) and then obtaining 48 uniformly spaced samples of each signature. The resulting 48-dimensional vectors were the inputs to the three-layer feedforward neural network shown in Fig. 9.19. The number of neuron nodes in the first layer was chosen to be 48, corresponding to the dimensionality of the input pattern vectors. The 4 neurons in the third (output) layer correspond to the number of pattern classes, and the number of neurons in the middle layer was heuristically specified as 26 (the average of the number of neurons in the input and output layers). There are no known rules for specifying the number of nodes in the internal layers of a neural network, so this number generally is based either on prior experience or simply chosen arbitrarily and then refined by testing. In the output layer, the four nodes from top to bottom in this case represent classes ω_j, $j = 1, 2, 3$, and 4, respectively. After the network structure has been set, activation functions have to be selected for each unit and layer. All activation functions were selected to satisfy Eq. (9.3-50) so that, according to the earlier discussion, Eqs. (9.3-72) and (9.3-73) apply.

The training process was divided in two parts. In the first part, the weights were initialized to small random values with zero mean, and the network was

9.3 Decision-Theoretic Methods 613

then trained with pattern vectors corresponding to noise-free samples like the shapes shown in Fig. 9.18(a). The output nodes were monitored during training. The network was said to have learned the shapes from all four classes when, for any training pattern from class ω_i, the elements of the output layer yielded $O_i \geq 0.95$ and $O_q \leq 0.05$, for $q = 1, 2, \ldots, N_Q$; $q \neq i$. In other words, for any pattern of class ω_i, the output unit corresponding to that class had to be high (≥ 0.95) while, simultaneously, the output of all other nodes had to be low (≤ 0.05).

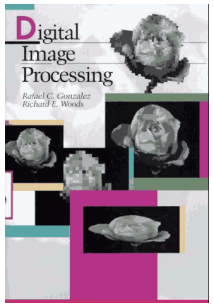
Figure 9.19 *Three-layer neural network used to recognize the shapes in Fig. 9.18.*



Digital Image Processing, Gonzalez and Woods, 1992

Klassisk maskinlæring årgang 1992

Pattern recognition, machine vision, neural networks...



Digital Image Processing, Gonzalez and Woods, 1992

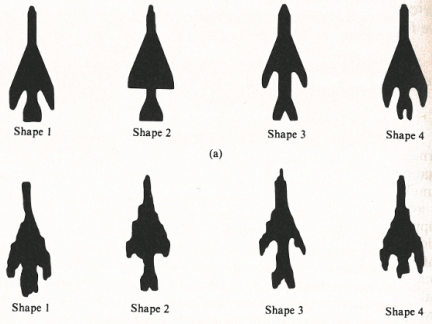


Figure 9.18 *(a) Reference shapes and (b) typical noisy shapes used in training the neural network of Fig. 9.19. (From Gupta et al. [1990].)*

Pattern vectors were generated by computing the normalized signatures of the shapes (see Section 8.1.3) and then obtaining 48 uniformly spaced samples of each signature. The resulting 48-dimensional vectors were the inputs to the three-layer feedforward neural network shown in Fig. 9.19. The number of neuron nodes in the first layer was chosen to be 48, corresponding to the dimensionality of the input pattern vectors. The 4 neurons in the third (output) layer correspond to the number of pattern classes, and the number of neurons in the middle layer was heuristically specified as 26 (the average of the number of neurons in the input and output layers). There are no known rules for specifying the number of nodes in the internal layers of a neural network, so this number generally is based either on prior experience or simply chosen arbitrarily and then refined by testing. In the output layer, the four nodes from top to bottom in this case represent classes ω_j, $j = 1, 2, 3$, and 4, respectively. After the network structure has been set, activation functions have to be selected for each unit and layer. All activation functions were selected to satisfy Eq. (9.3-50) so that, according to the earlier discussion, Eqs. (9.3-72) and (9.3-73) apply.

The training process was divided in two parts. In the first part, the weights were initialized to small random values with zero mean, and the network was

then trained with pattern vectors corresponding to noise-free samples like the shapes shown in Fig. 9.18(a). The output nodes were monitored during training. The network was said to have learned the shapes from all four classes when, for any training pattern from class ω_i, the elements of the output layer yielded $O_i \geq 0.95$ and $O_q \leq 0.05$, for $q = 1, 2, \ldots, N_Q$; $q \neq i$. In other words, for any pattern of class ω_i, the output unit corresponding to that class had to be high (≥ 0.95) while, simultaneously, the output of all other nodes had to be low (≤ 0.05).

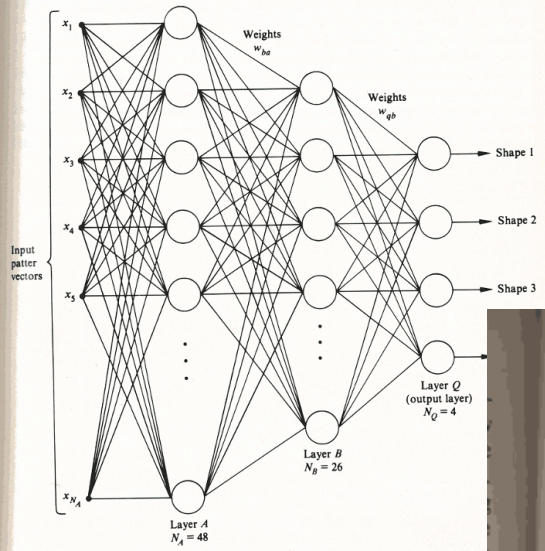


Figure 9.19 *Three-layer neural network used to recognize the shapes in Fig. 9.18*

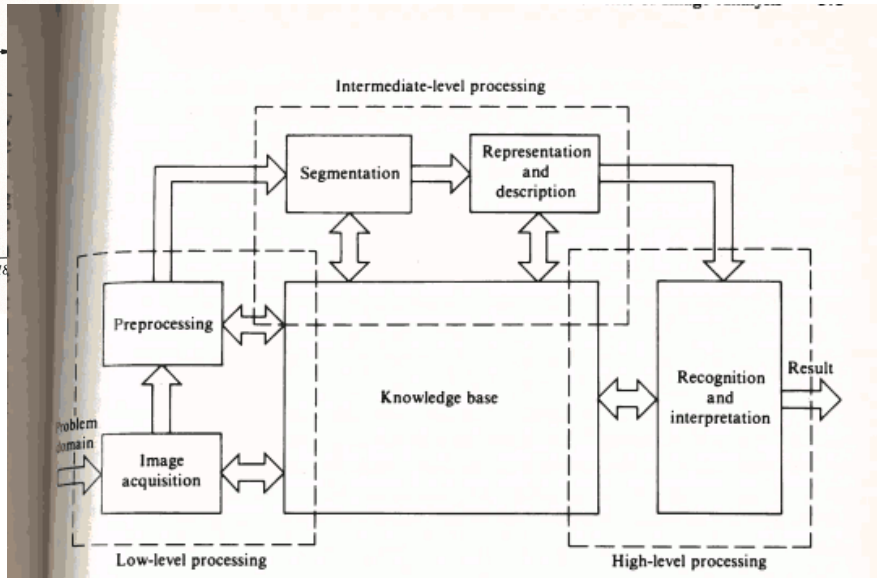
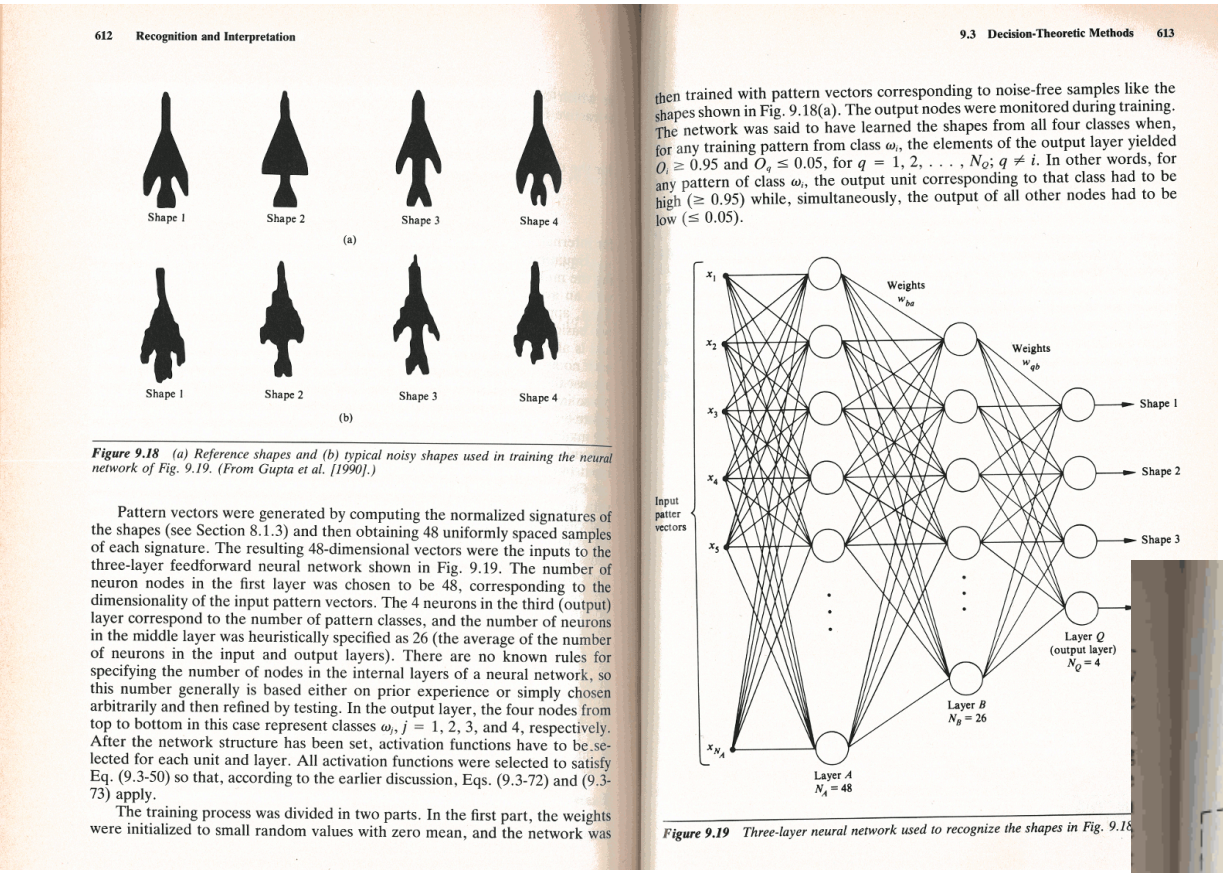


Figure 9.1 *Elements of image analysis.*

dark theater from bright sunlight. The (intelligent) process of finding an unoccupied seat cannot begin until a suitable image is available. The process

Klassisk maskinlæring årgang 1992

Pattern recognition, machine vision, neural networks...



612 Recognition and Interpretation

Figure 9.18 *(a) Reference shapes and (b) typical noisy shapes used in training the neural network of Fig. 9.19. (From Gupta et al. [1990].)*

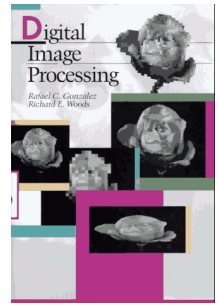
Pattern vectors were generated by computing the normalized signatures of the shapes (see Section 8.1.3) and then obtaining 48 uniformly spaced samples of each signature. The resulting 48-dimensional vectors were the inputs to the three-layer feedforward neural network shown in Fig. 9.19. The number of neuron nodes in the first layer was chosen to be 48, corresponding to the dimensionality of the input pattern vectors. The 4 neurons in the third (output) layer correspond to the number of pattern classes, and the number of neurons in the middle layer was heuristically specified as 26 (the average of the number of neurons in the input and output layers). There are no known rules for specifying the number of nodes in the internal layers of a neural network, so this number generally is based either on prior experience or simply chosen arbitrarily and then refined by testing. In the output layer, the four nodes from top to bottom in this case represent classes ω_j, $j = 1, 2, 3$, and 4, respectively. After the network structure has been set, activation functions have to be selected for each unit and layer. All activation functions were selected to satisfy Eq. (9.3-50) so that, according to the earlier discussion, Eqs. (9.3-72) and (9.3-73) apply.

The training process was divided in two parts. In the first part, the weights were initialized to small random values with zero mean, and the network was

9.3 Decision-Theoretic Methods 613

then trained with pattern vectors corresponding to noise-free samples like the shapes shown in Fig. 9.18(a). The output nodes were monitored during training. The network was said to have learned the shapes from all four classes when, for any training pattern from class ω_i, the elements of the output layer yielded $O_i \geq 0.95$ and $O_q \leq 0.05$, for $q = 1, 2, \ldots, N_Q$; $q \neq i$. In other words, for any pattern of class ω_i, the output unit corresponding to that class had to be high (≥ 0.95) while, simultaneously, the output of all other nodes had to be low (≤ 0.05).

Figure 9.19 *Three-layer neural network used to recognize the shapes in Fig. 9.18*



Digital Image Processing, Gonzalez and Woods, 1992

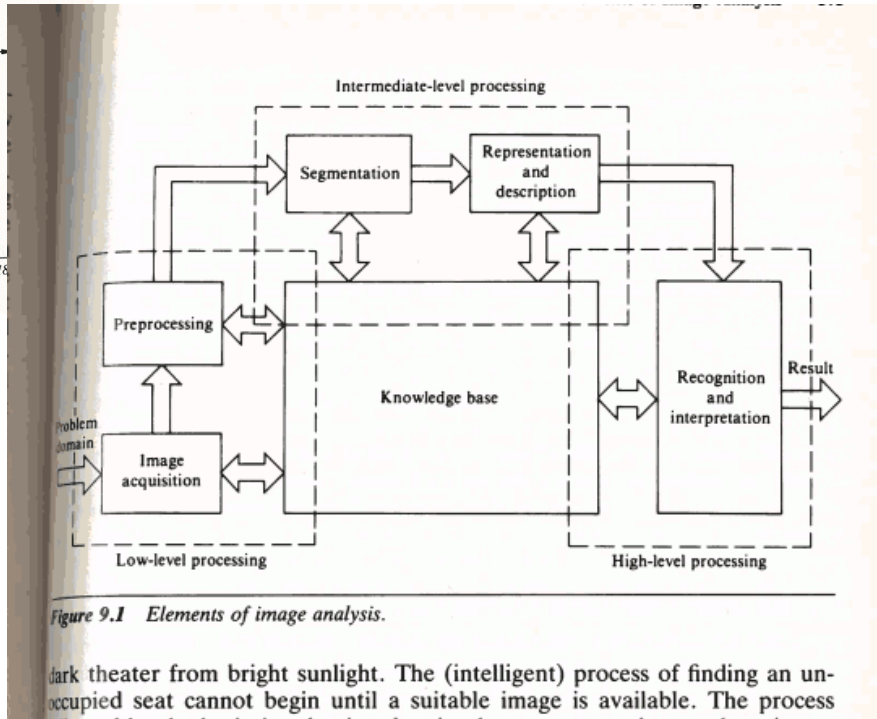


Figure 9.1 *Elements of image analysis.*

dark theater from bright sunlight. The (intelligent) process of finding an unoccupied seat cannot begin until a suitable image is available. The process

- Indeholder allerede det 'meste' ML,
- ML "vintre og somre":
  90'erne=sommer,
  00'erne=vinter

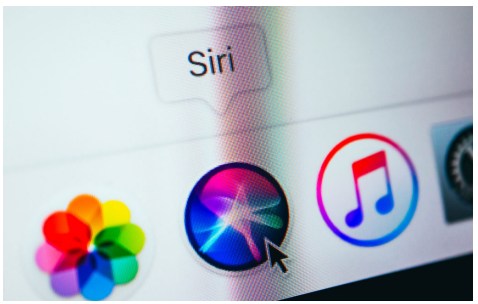
Moderne maskinlæring, renæssancen

SciFi:

Moderne maskinlæring, renæssancen

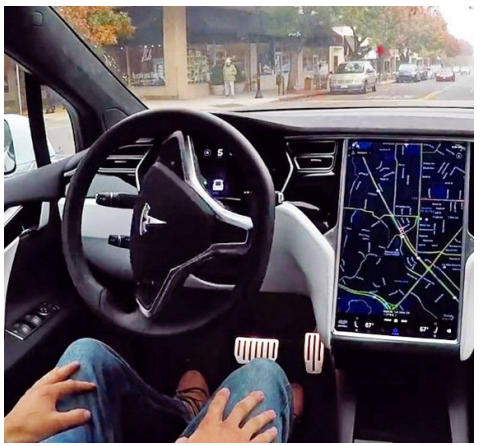
SciFi:

Real:



Apple's Siri

IBM's Watson



Tesla's selvkørende bil

Moderne maskinlæring

Diskussion: hvad er ML for Jer?

- hvad kender I af ML systemer?
- hvilke ML systemer anvender I allerede nu?
- ...og andre ML relaterede kommentarer!

Moderne maskinlæring

Eksponentiel udvikling og Moore's Lov

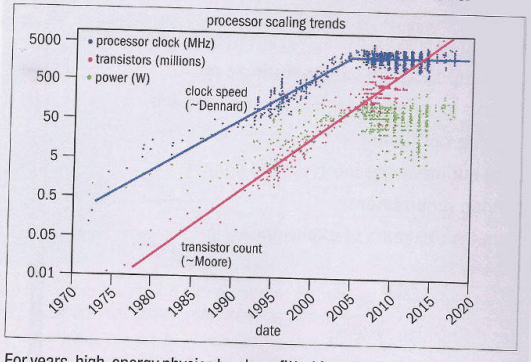
more resources at the problem and hope things will work out. A more radical approach for improvements is needed. Fortunately, this comes at a time when other fields have started to tackle data-mining problems of a comparable scale to those in high-energy physics – today's commercial data centres crunch data at prodigious rates and exceed the size of our biggest Tier-1 WLCG centres by a large margin. Our efforts in software and computing therefore naturally fit into and can benefit from the emerging field of data science.

A new way to approach the high-energy physics (HEP) computing problem began in 2014, when the HEP Software Foundation (HSF) was founded. Its aim was to bring the HEP software community together and find common solutions to the challenges ahead, beginning with a number of workshops organised by a dedicated startup team. In the summer of 2016 the fledgling HSF body was charged by WLCG leaders to produce a roadmap for HEP software and computing. With help from a planning grant from the US National Science Foundation, at a meeting in San Diego in January 2017, the HSF brought community and non-HEP experts together to gather ideas in a world much changed from the time when the first LHC software was created. The outcome of this process was summarised in a 90-page-long community white paper released in December last year.

The report doesn't just look at the LHC but considers common problems across HEP, including neutrino and other "intensity-frontier" experiments, Belle II at KEK, and future linear and circular colliders. In addition to improving the performance of our software and optimising the computing infrastructure itself, the report also explores new approaches that would extend our physics reach as well as ways to improve the sustainability of our software to match the multi-decade lifespan of the experiments.

Almost every aspect of HEP software and computing is presented in the white paper, detailing the R&D programmes necessary to deliver the improvements the community needs. HSF members looked at all steps from event generation and data taking up to final analysis, each of which presents specific challenges and opportunities.

CPU scaling comes to the end of an era

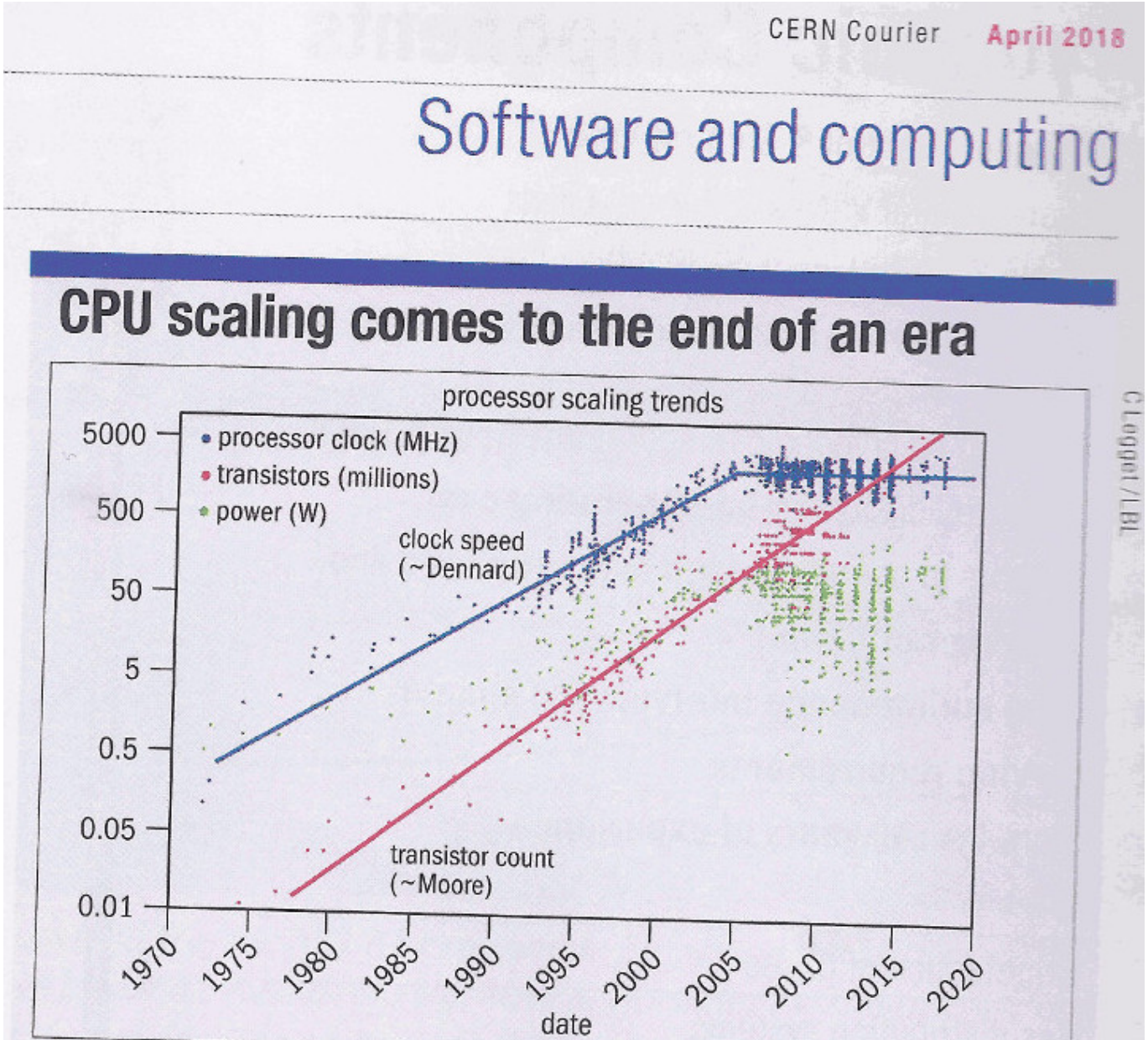


C Leggett / LBL

For years, high-energy physics has benefitted from a wave of improvements in computing hardware. Every couple of years the performance of code more than doubled because as transistors became smaller, they could be run faster. Many problems that were faced could be solved simply by waiting for computers to get faster, and this trend was a key part of the plan for LHC computing. However, as can be seen from the plot above, in the mid-2000s this "Dennard scaling" ground to an abrupt halt: while chip makers continued to pack more and more smaller transistors onto silicon following Moore's law (red points), it became impossible to increase the clock speed of the CPU (blue) further due to the increased heat load. With the performance of single CPU cores stalling, manufacturers started to add multiple independent CPU cores to their machines. Instead of doing one thing faster, such architectures could at least do many things together at the same speed. This trend continues today, and is one key change to which high-energy physicists need to adapt to find practical solutions to the big-data challenges of the 2020s.

Moderne maskinlæring

Eksponentiel udvikling og Moore's Lov



Moderne maskinlæring

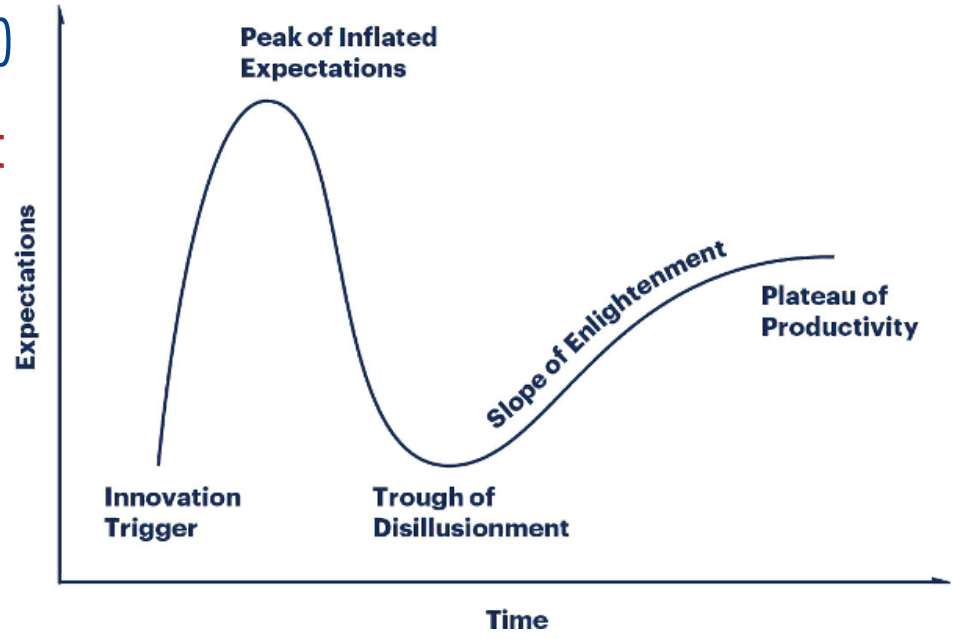
Forskel på ML og AI (artificial intelligence)

- Er det skrevet i PowerPoint
  så er det AI.
- Er det skrevet i Python
  så er det ML.

Moderne maskinlæring

Forskel på ML og AI (artificial intelligence)

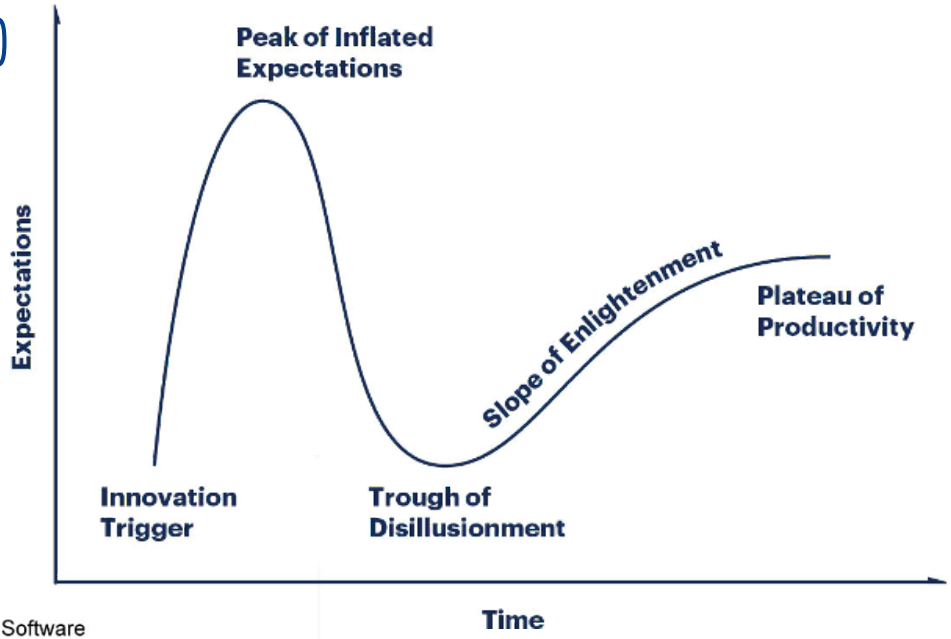
- Er det skrevet i PowerPoint
  så er det AI.
- Er det skrevet i Python
  så er det ML.

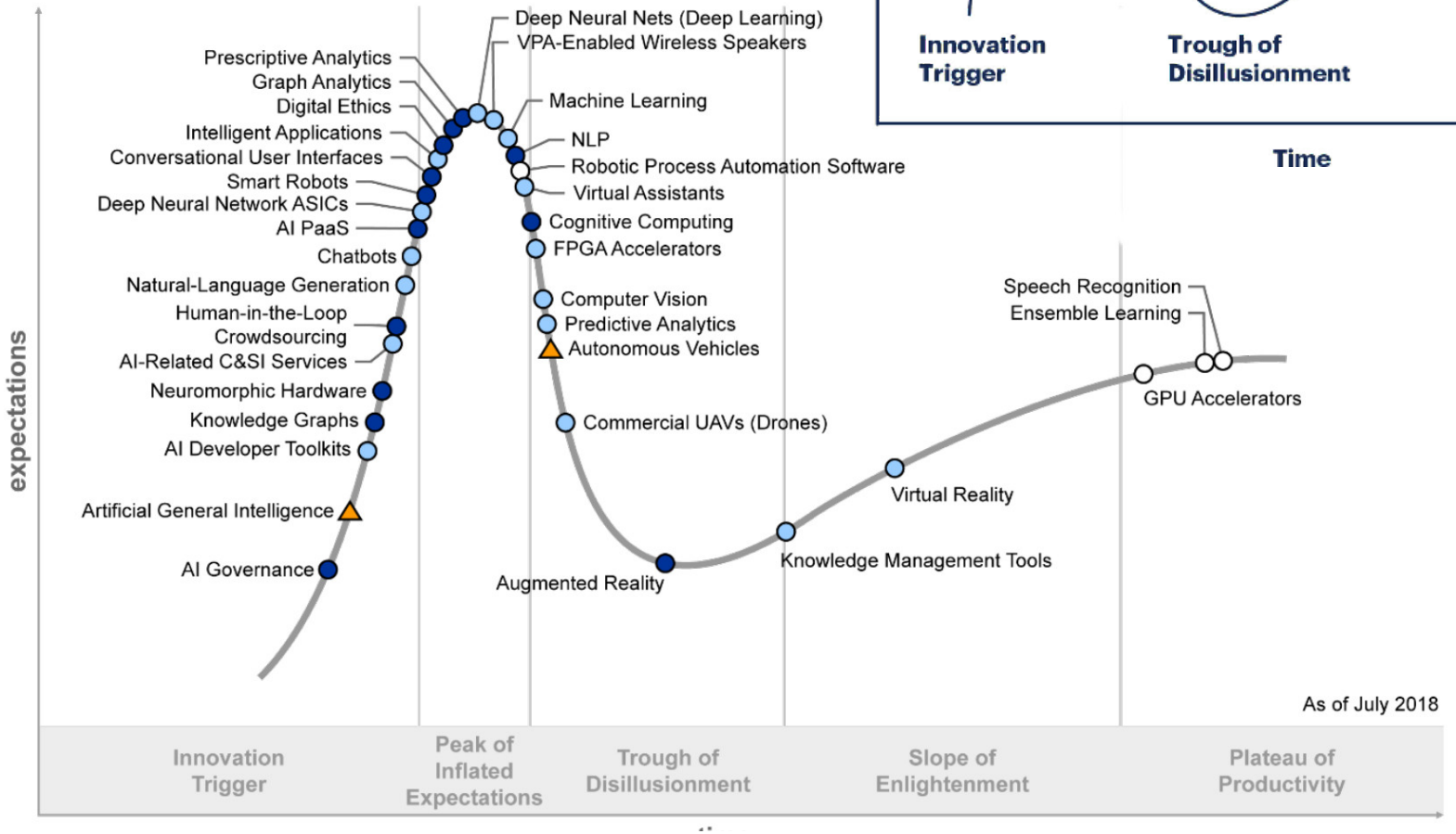


Moderne maskinlæring

Forskel på ML og AI (artificial intelligence)

- Er det skrevet i PowerPoint
  så er det AI.
- Er det skrevet i Python
  så er det ML.

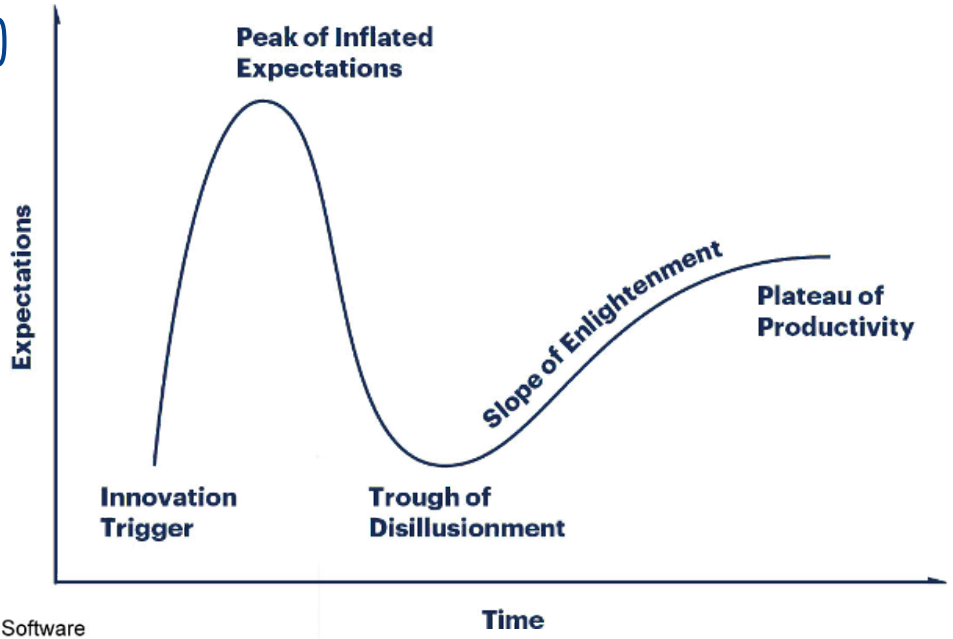


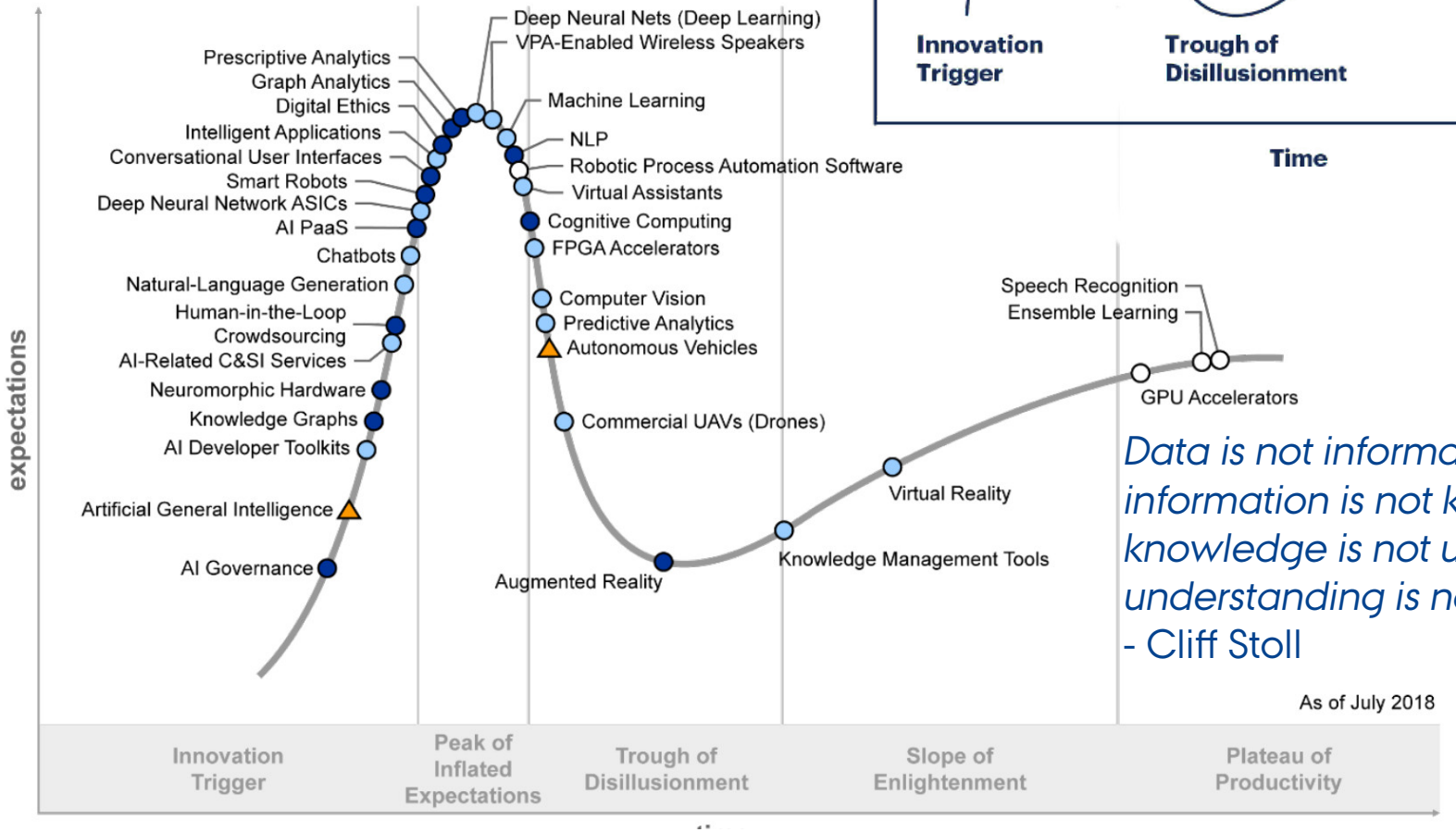


Moderne maskinlæring

Forskel på ML og AI (artificial intelligence)

- Er det skrevet i PowerPoint
  så er det AI.
- Er det skrevet i Python
  så er det ML.





Data is not information,
information is not knowledge,
knowledge is not understanding,
understanding is not wisdom
- Cliff Stoll

Moderne maskinlæring

Forskel på Sci-Fi og Science

- Har seks årtiers forskning i AI led til ingenting?
- Forstår vi hjernen, cognition, menneskelig intelligens?

Moderne maskinlæring

Forskel på Sci-Fi og Science

- Har seks årtiers forskning i AI led til ingenting?
- Forstår vi hjernen, cognition, menneskelig intelligens?
- Vil eksponentiel udvikling give os ægte AI?

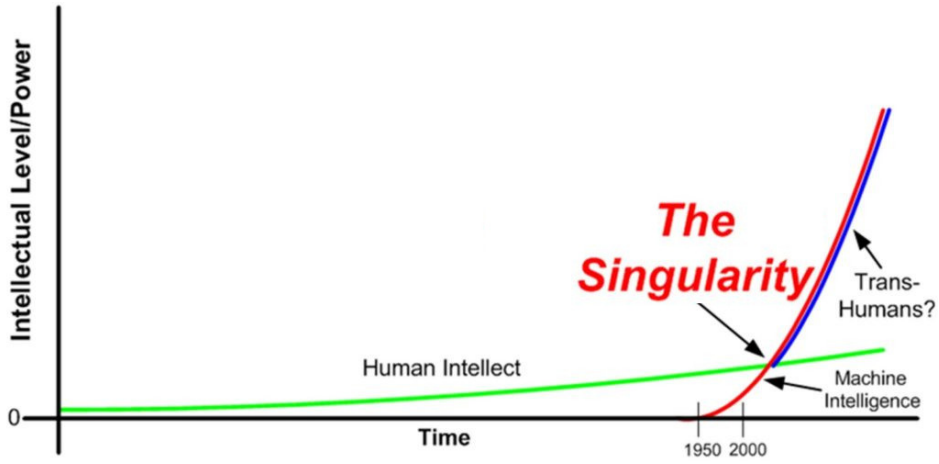
  (Ray Kurzweil, Singularity)
- Hvor er alle android, cyborg eller HAL-9000'erne i vores hverdag?

Moderne maskinlæring

Forskel på Sci-Fi og Science

- Har seks årtiers forskning i AI led til ingenting?
- Forstår vi hjernen, cognition, menneskelig intelligens?
- Vil eksponentiel udvikling give os ægte AI?

  (Ray Kurzweil, Singularity)
- Hvor er alle android, cyborg eller HAL-9000'erne i vores hverdag?

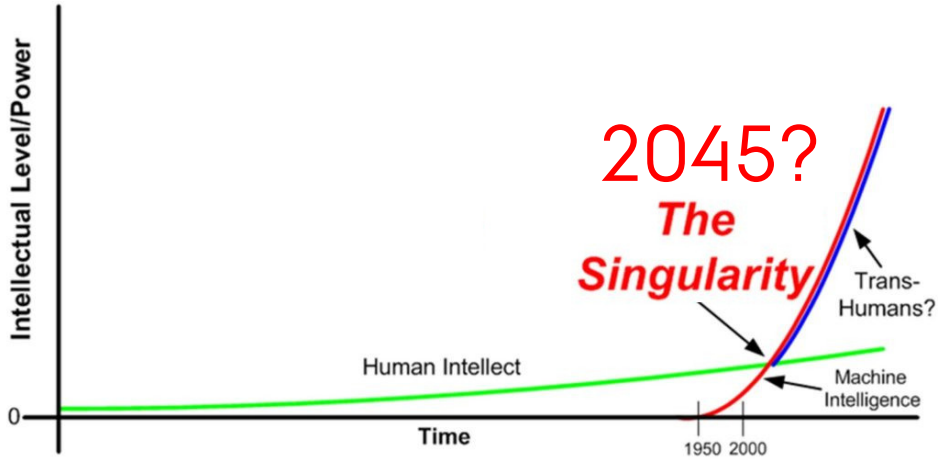


Moderne maskinlæring

Forskel på Sci-Fi og Science

- Har seks årtiers forskning i AI led til ingenting?
- Forstår vi hjernen, cognition, menneskelig intelligens?
- Vil eksponentiel udvikling give os ægte AI?

  (Ray Kurzweil, Singularity)
- Hvor er alle android, cyborg eller HAL-9000'erne i vores hverdag?

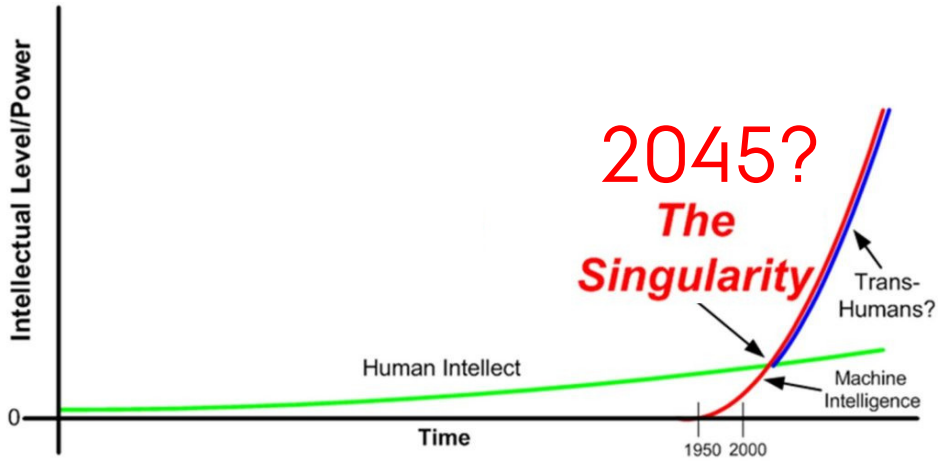


Moderne maskinlæring

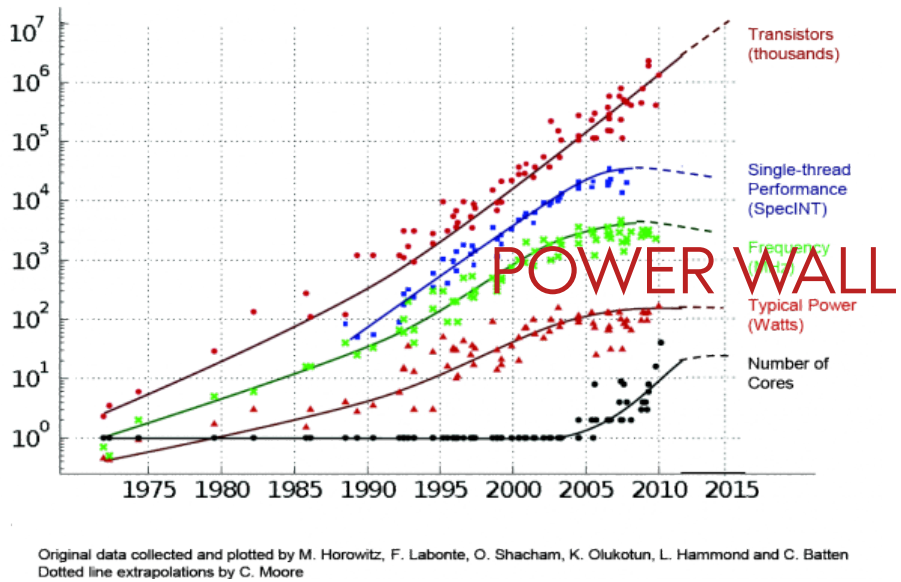
Forskel på Sci-Fi og Science

- Har seks årtiers forskning i AI led til ingenting?
- Forstår vi hjernen, cognition, menneskelig intelligens?
- Vil eksponentiel udvikling give os ægte AI?

  (Ray Kurzweil, Singularity)
- Hvor er alle android, cyborg eller HAL-9000'erne i vores hverdag?



35 YEARS OF MICROPROCESSOR TREND DATA

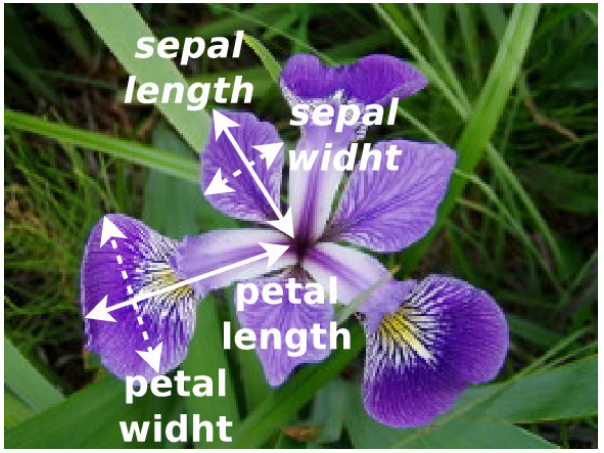


Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

Sci: Dette kursus handler om Hands-on ML+python—ikke om Sci-Fi.

Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...



Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...

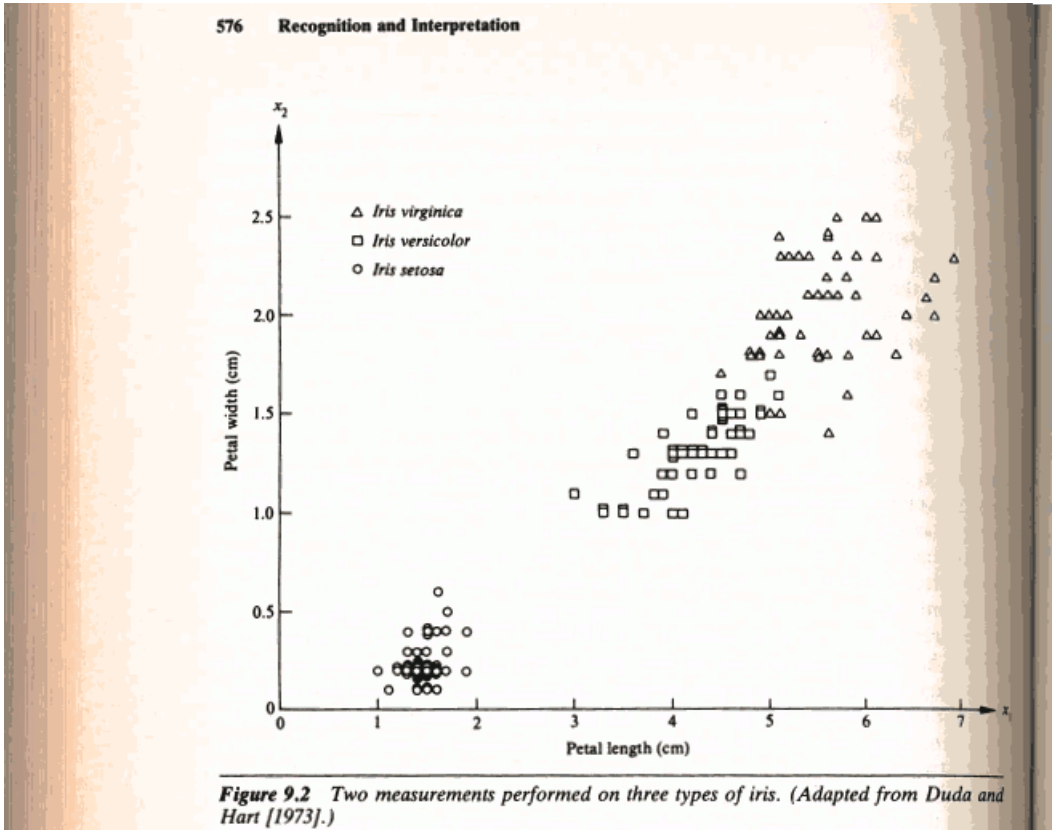
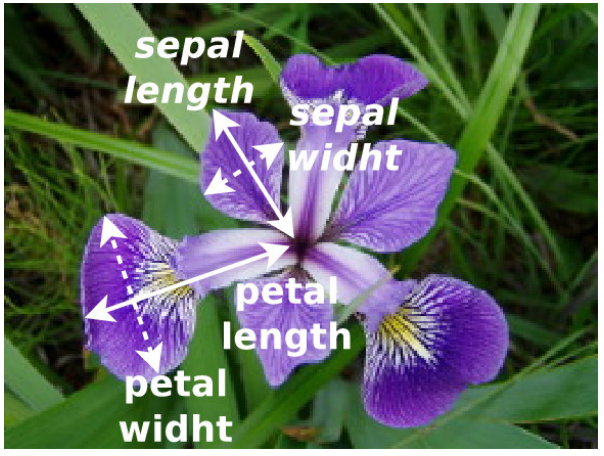


Figure 9.2 *Two measurements performed on three types of iris. (Adapted from Duda and Hart [1973].)*



Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...

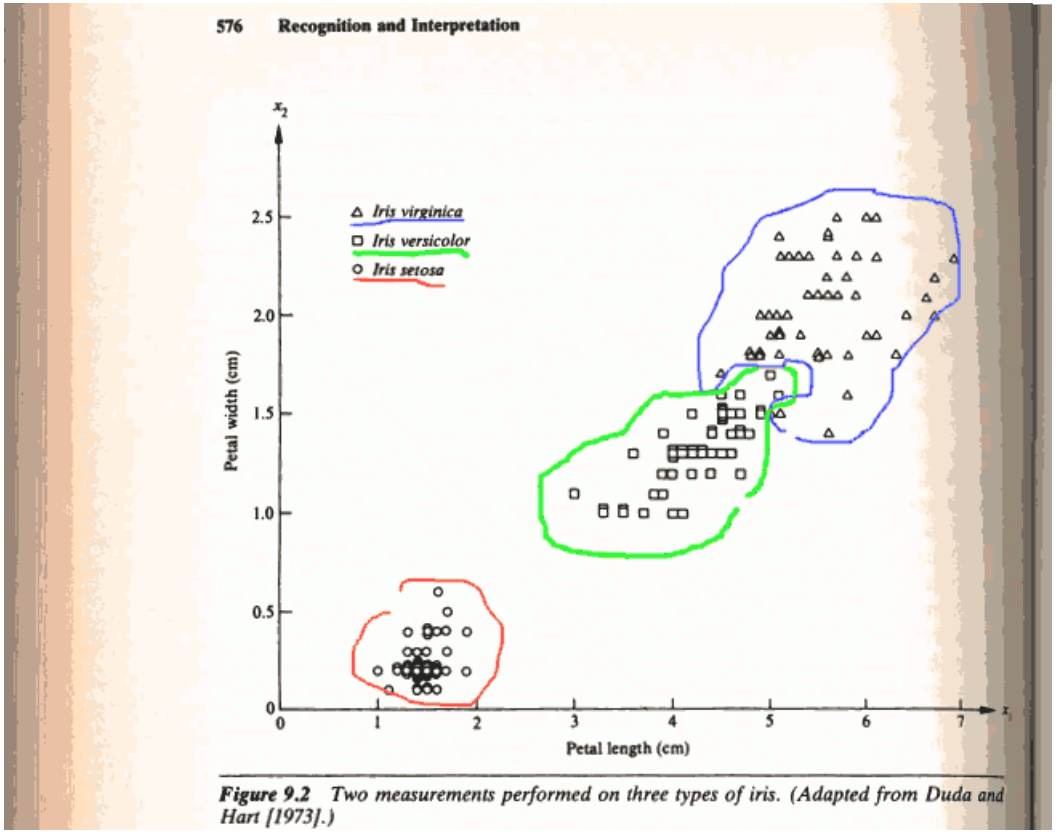
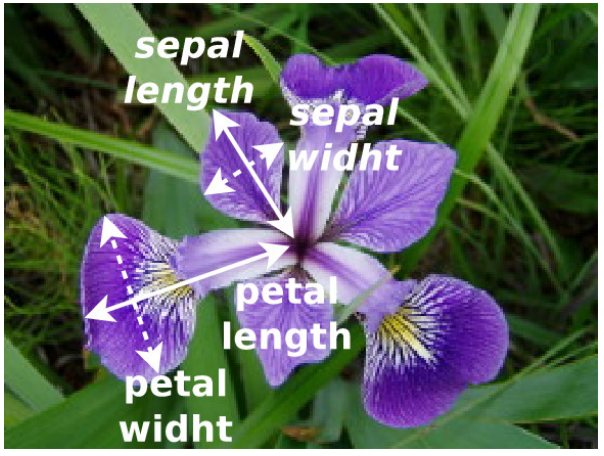


Figure 9.2 *Two measurements performed on three types of iris. (Adapted from Duda and Hart [1973].)*



Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...

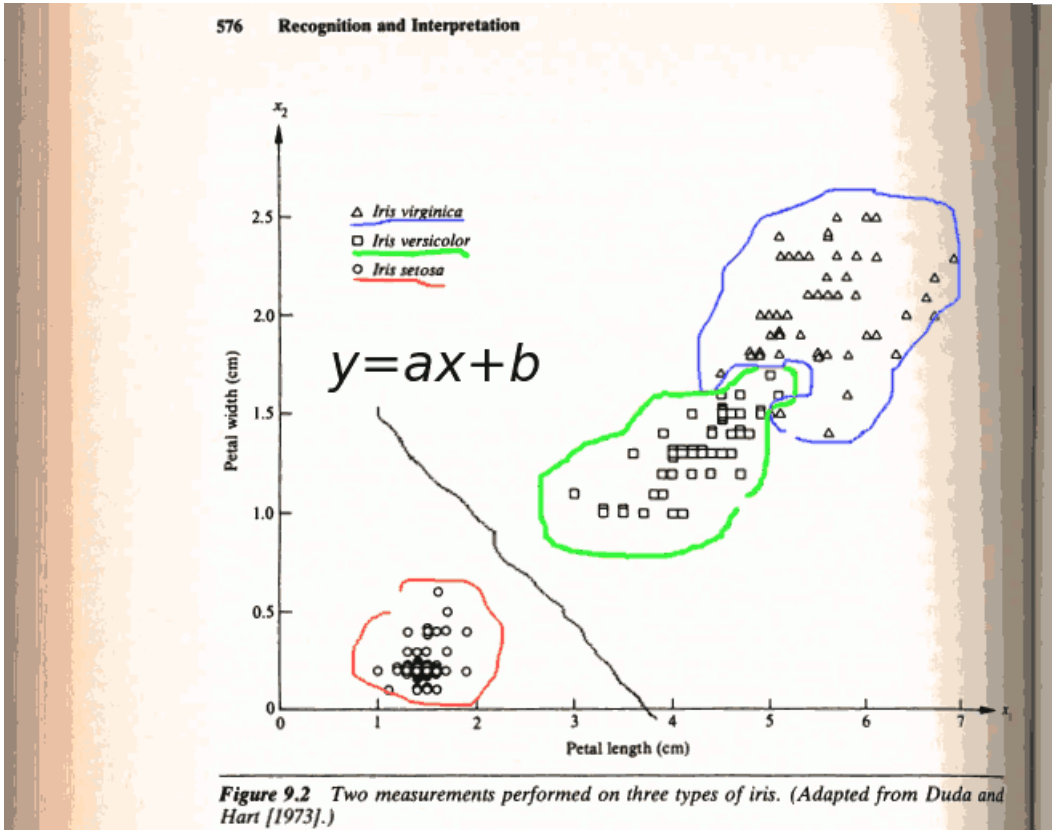
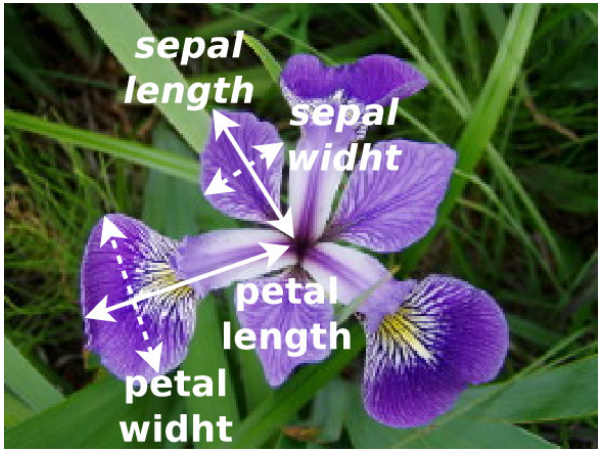
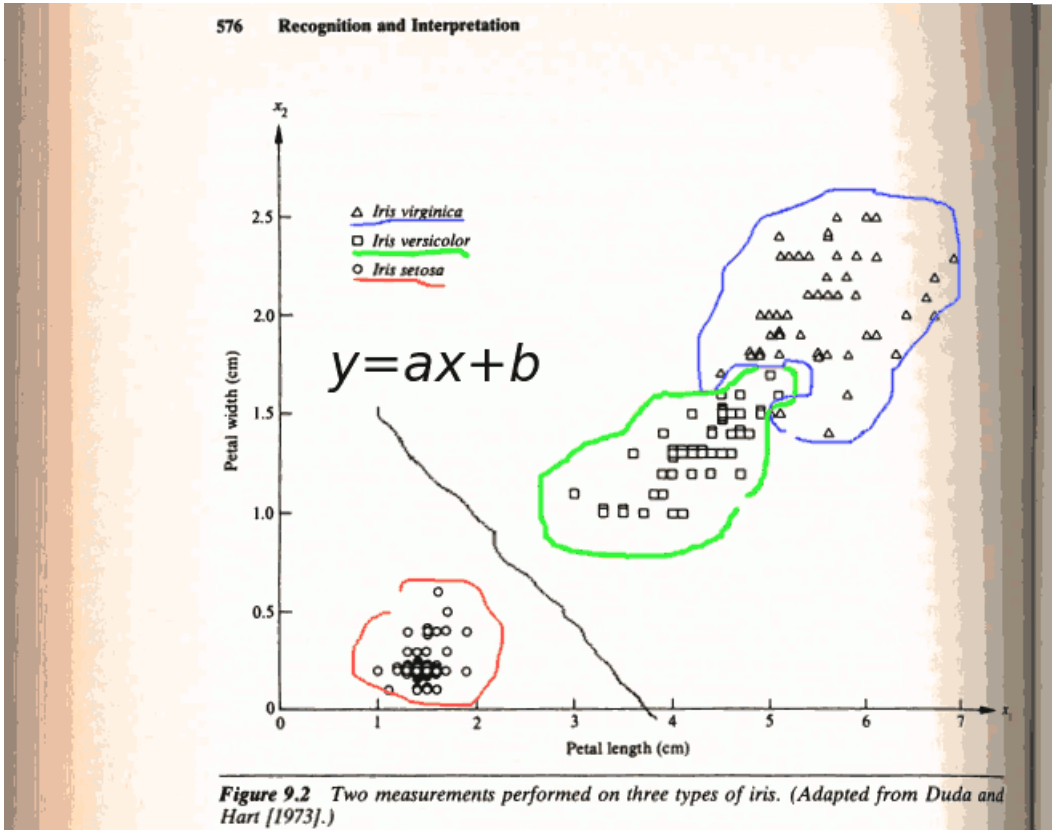


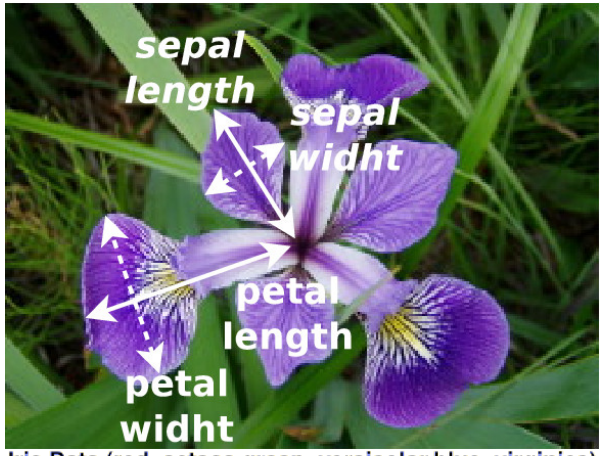
Figure 9.2 *Two measurements performed on three types of iris. (Adapted from Duda and Hart [1973].)*



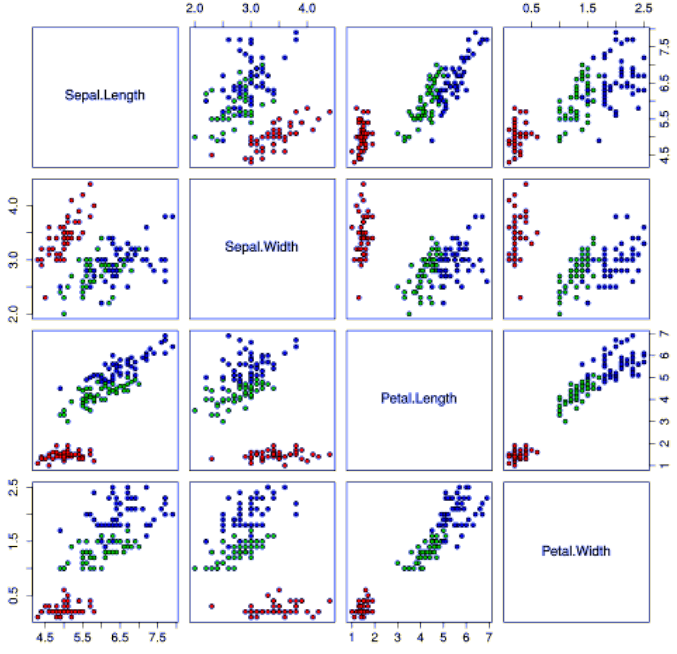
Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...





Iris Data (red=setosa,green=versicolor,blue=virginica)



Fra klassisk til moderne maskinlæring

Stadig pattern recognition, machine vision, neural networks...

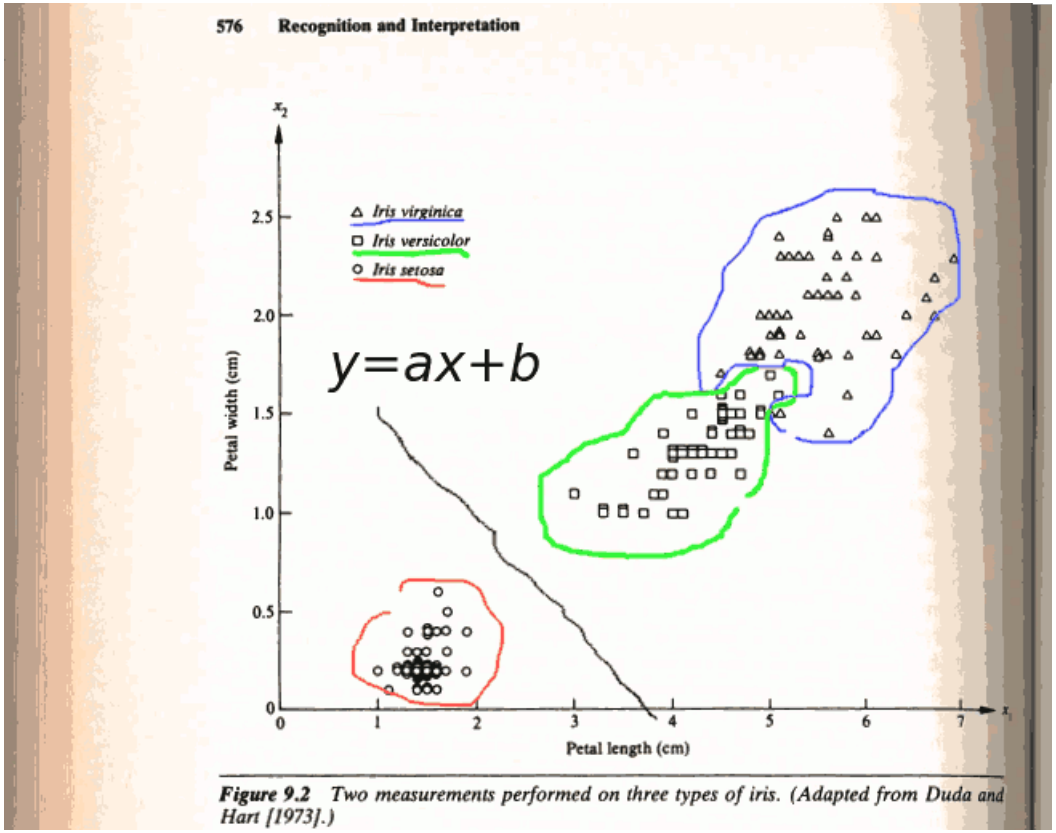
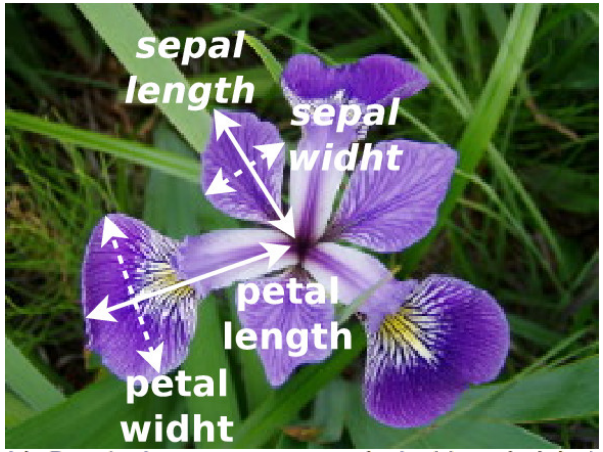
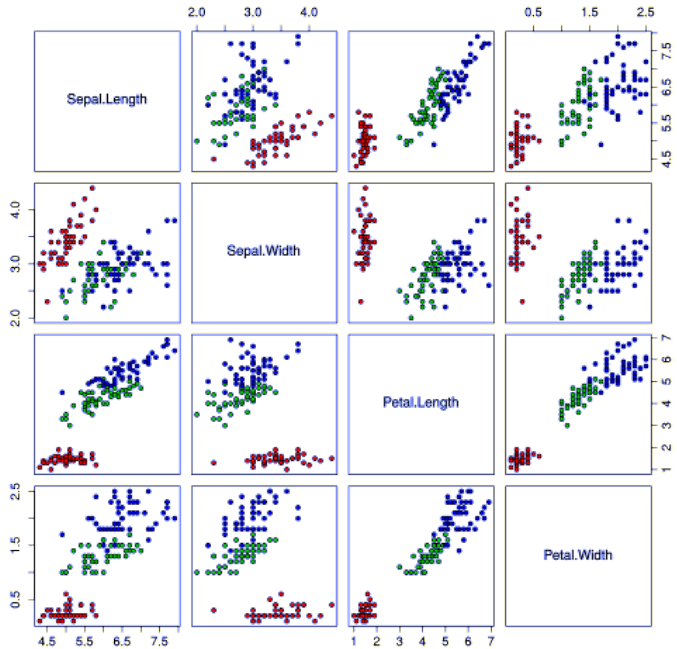


Figure 9.2 *Two measurements performed on three types of iris. (Adapted from Duda and Hart [1973].)*



Iris Data (red=setosa,green=versicolor,blue=virginica)



- ikke-nyt$_1$: matematik,
- ikke-nyt$_2$: algoritmer,
- nyt$_1$: meget mere data og flere dimensioner, f.eks. 4D til 784D,
- nyt$_2$: hurtigere hardware (og parallelitet).

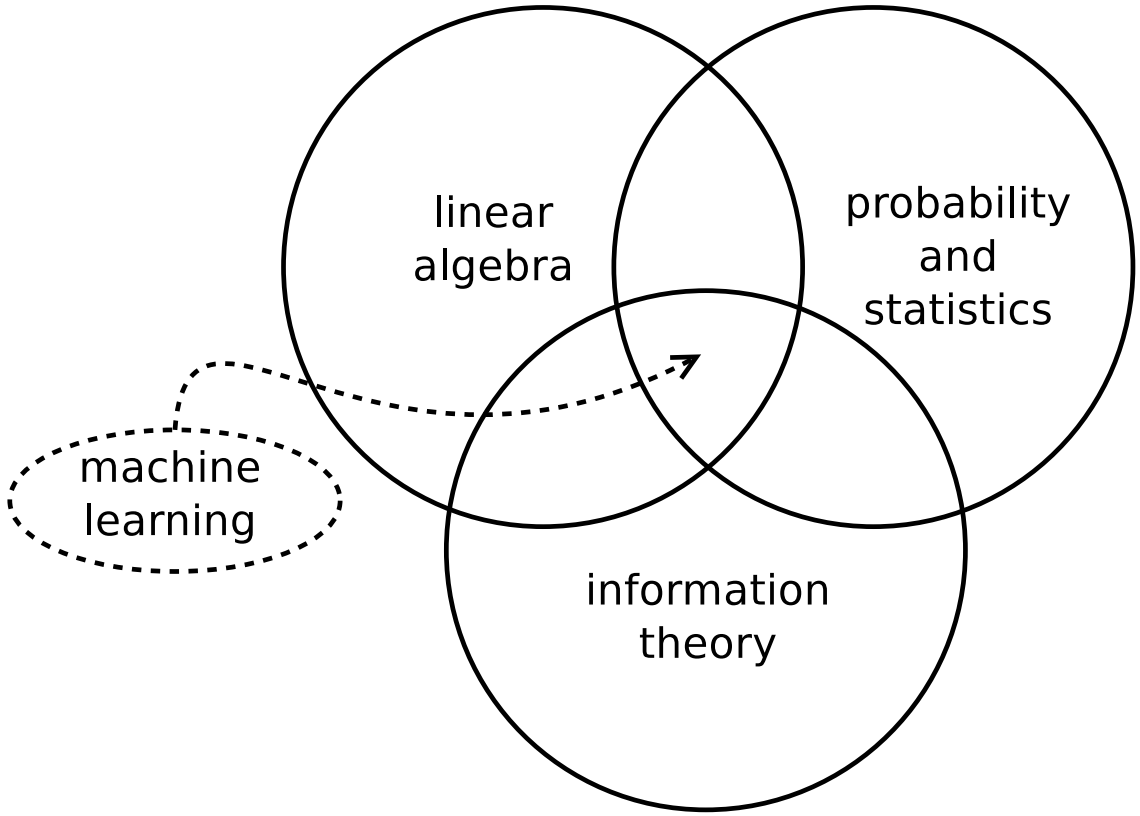
Fra MMLS (1.semester) til ITMAL

Matrix notation genopfriskning og intro til Design Matrix'en...

(Se noter..)

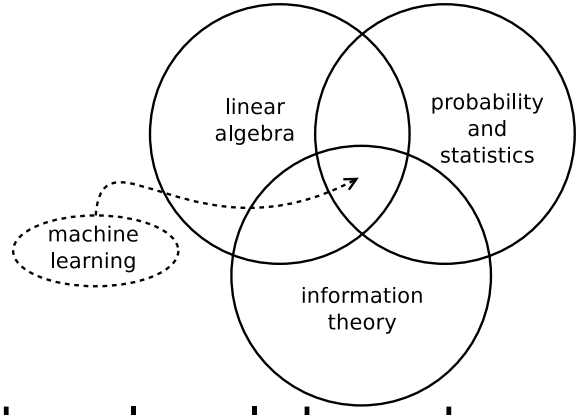
Machine learning baggrund

Machine learning: matematisk baggrund



Machine learning baggrund

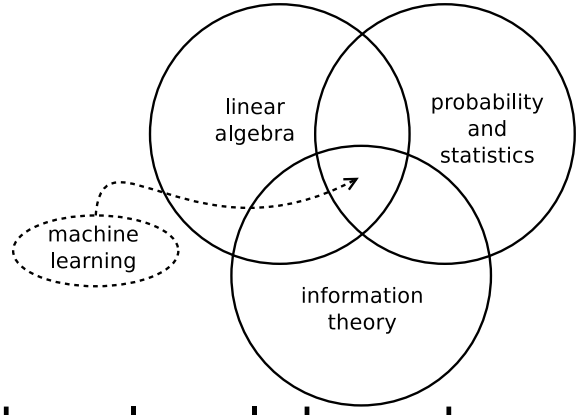
Machine learning: matematisk baggrund



- Lineær algebra, læs løbende op:
  - norm (afstand),
  - matrix algebra (mest multiplikation),
  - least-square closed solution, $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$,
  - nabla operator, $\nabla_\mathbf{w} = [\frac{\partial}{\partial\, w_1}, \frac{\partial}{\partial\, w_2}..]$,

Machine learning baggrund

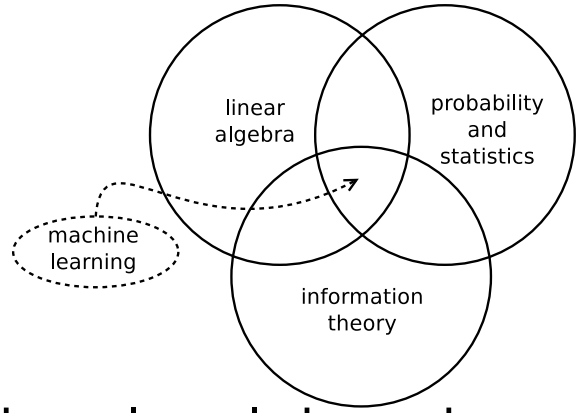
Machine learning: matematisk baggrund



- Lineær algebra, læs løbende op:
  - norm (afstand),
  - matrix algebra (mest multiplikation),
  - least-square closed solution, $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$,
  - nabla operator, $\nabla_\mathbf{w} = [\frac{\partial}{\partial\, w_1}, \frac{\partial}{\partial\, w_2}..]$,
- Sandsynlighedsregning, læs løbende op:
  - multivariate mean, variance,
  - multivariate Gaussisk distribution,
  - (Bayes',)

Machine learning baggrund

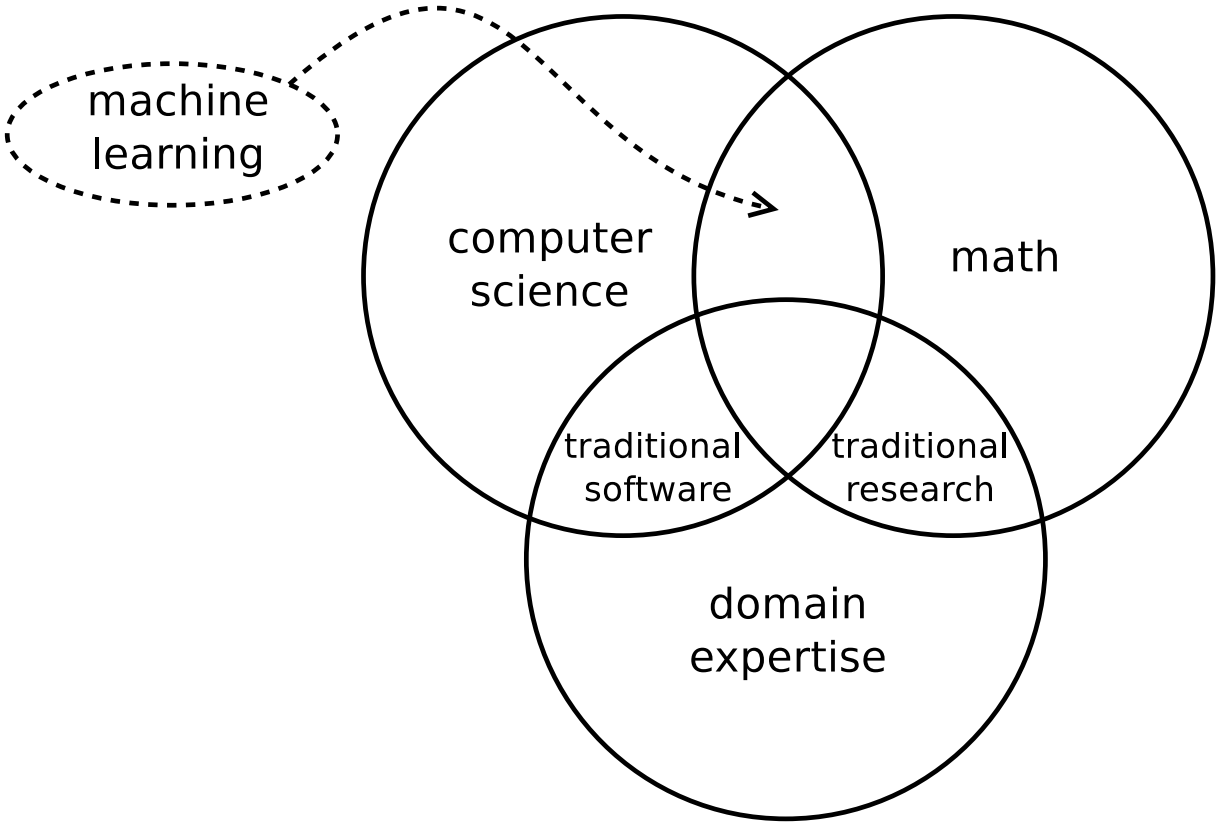
Machine learning: matematisk baggrund



- Lineær algebra, læs løbende op:
  - norm (afstand),
  - matrix algebra (mest multiplikation),
  - least-square closed solution, $(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top\mathbf{y}$,
  - nabla operator, $\nabla_{\mathbf{w}} = [\frac{\partial}{\partial\, w_1}, \frac{\partial}{\partial\, w_2}..]$,
- Sandsynlighedsregning, læs løbende op:
  - multivariate mean, variance,
  - multivariate Gaussisk distribution,
  - (Bayes',)
- Informationsteori: vi navigere (mest) udenom entropi og andre informations-teori elementer i dette kursus.

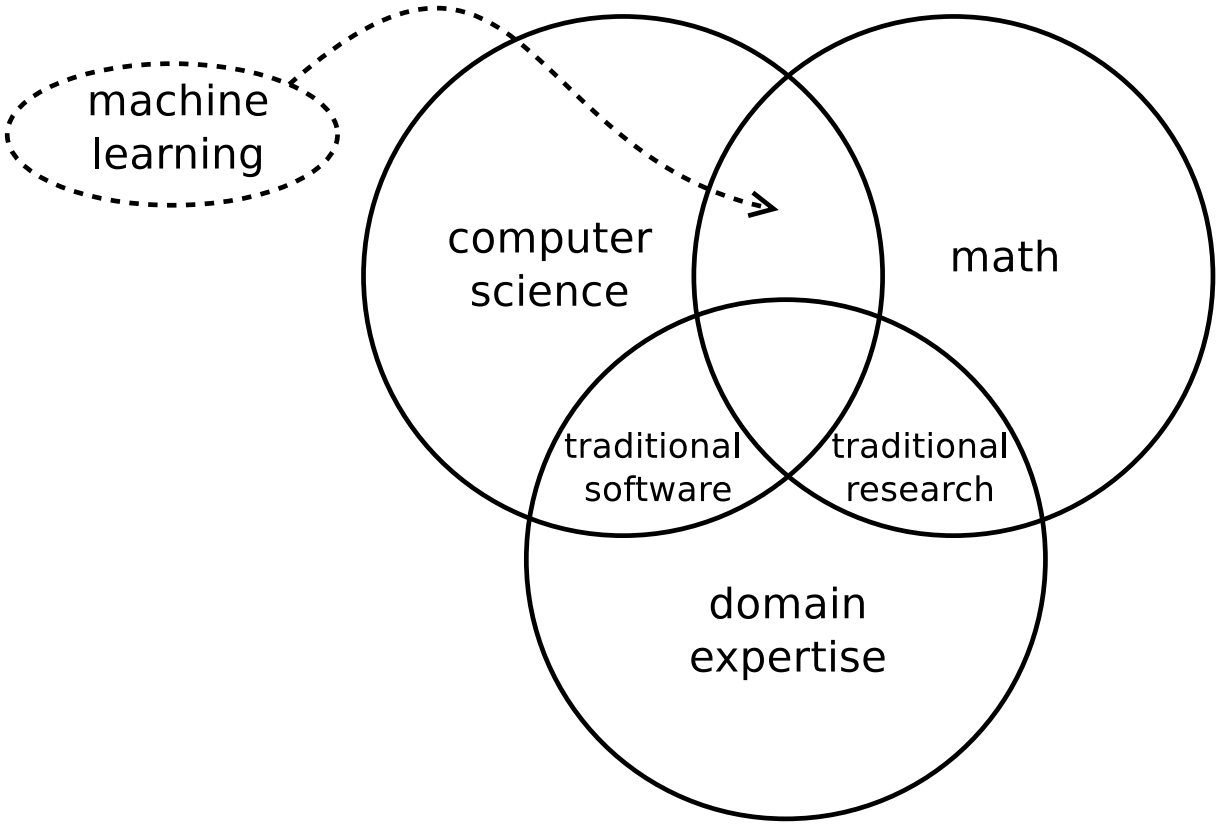
Machine learning baggrund

Machine learning: ekspertise



Machine learning baggrund

Machine learning: ekspertise

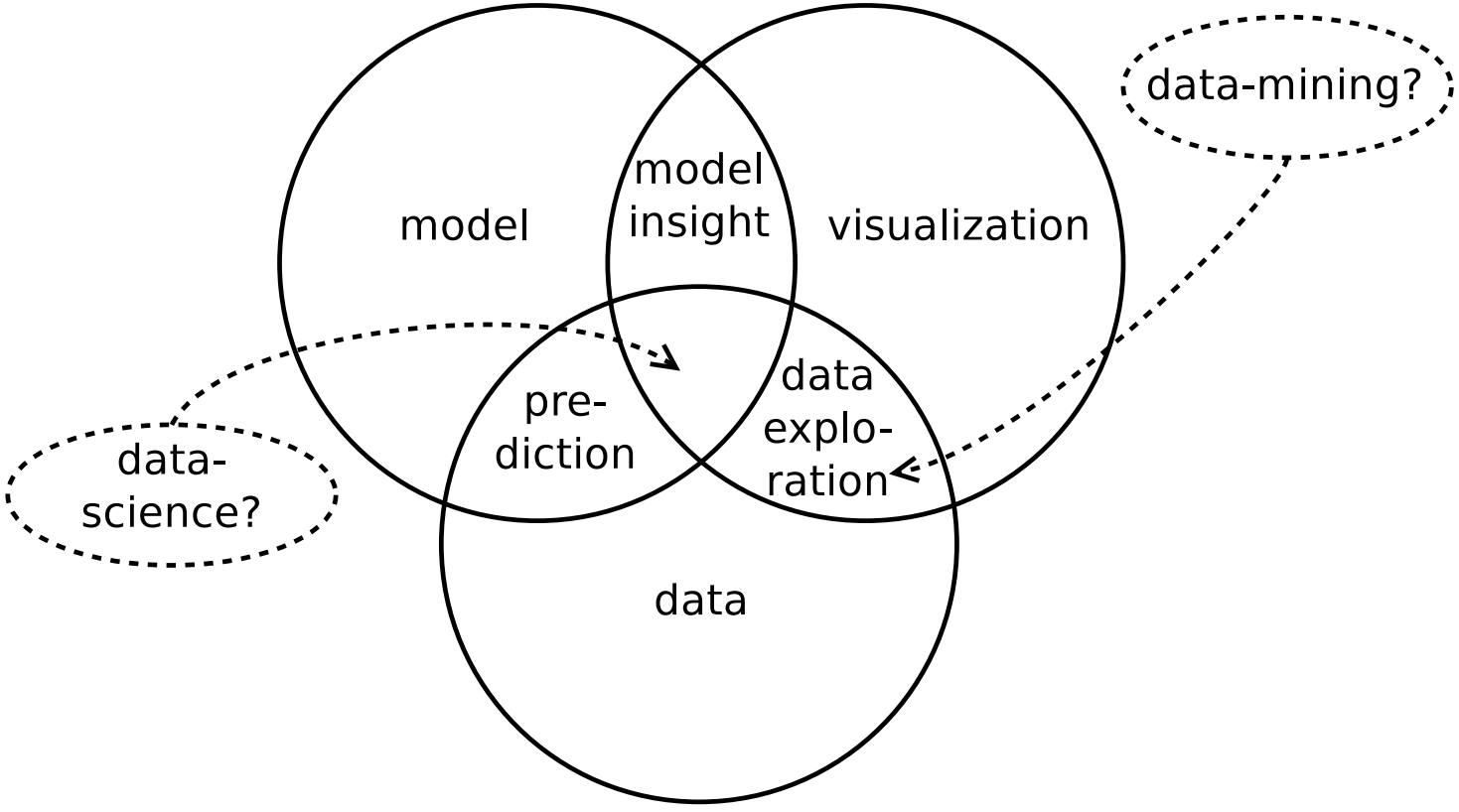


- ML ekspert er computer science og matematik ekspert.
- ML ekspert er IKKE (nødvendigvis) domæne ekspert!

NOTE:
[https://imarticus.org/what-are-the-skills-you-need-to-become-a-machine-learning-engineer/]

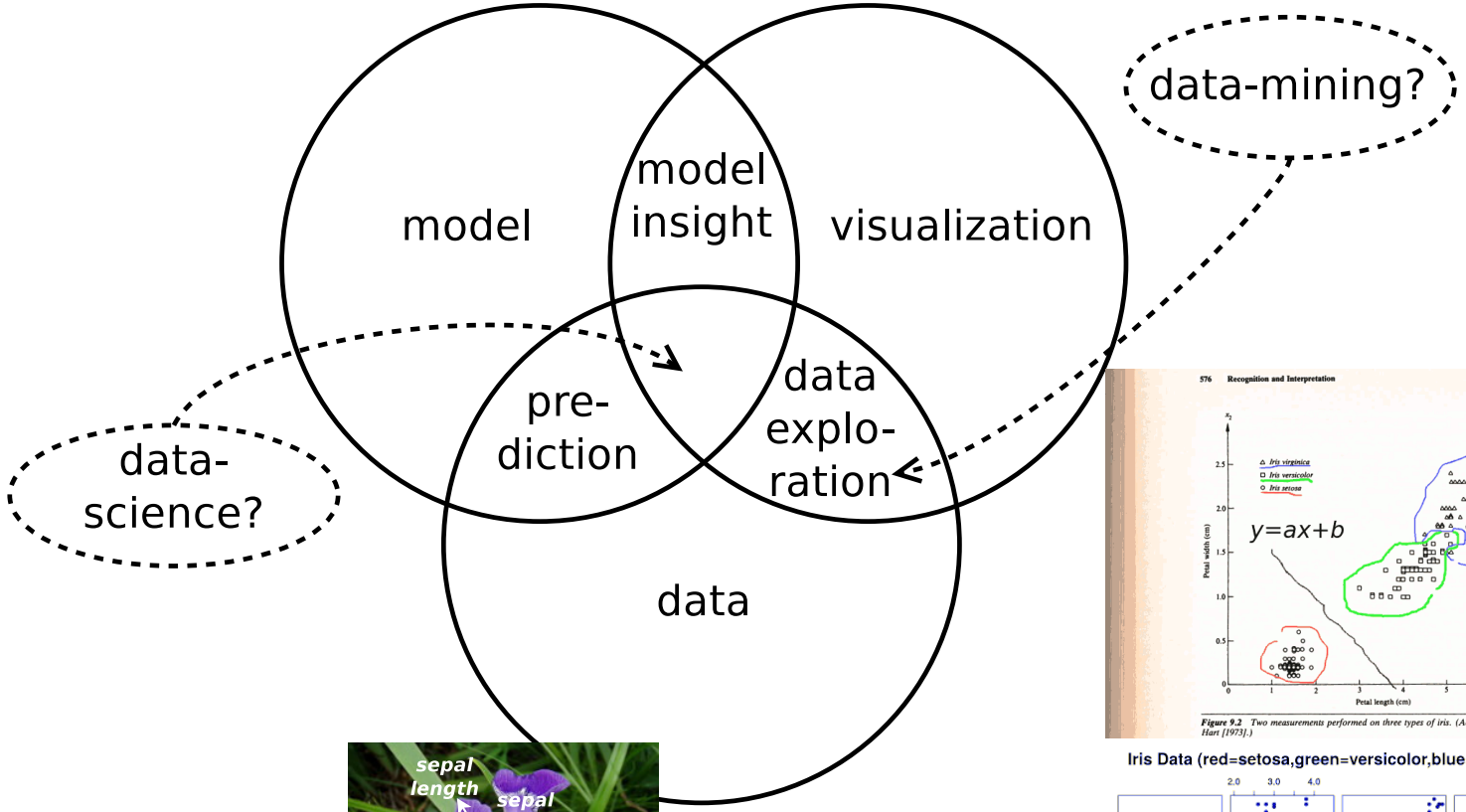
Machine learning baggrund

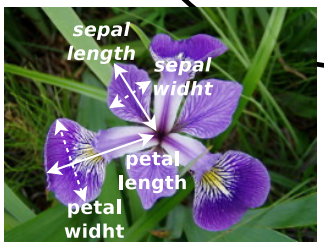
Machine learning: data science ekspert

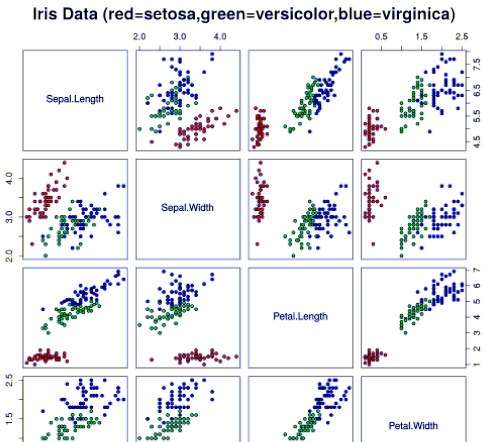


Machine learning baggrund

Machine learning: data science ekspert

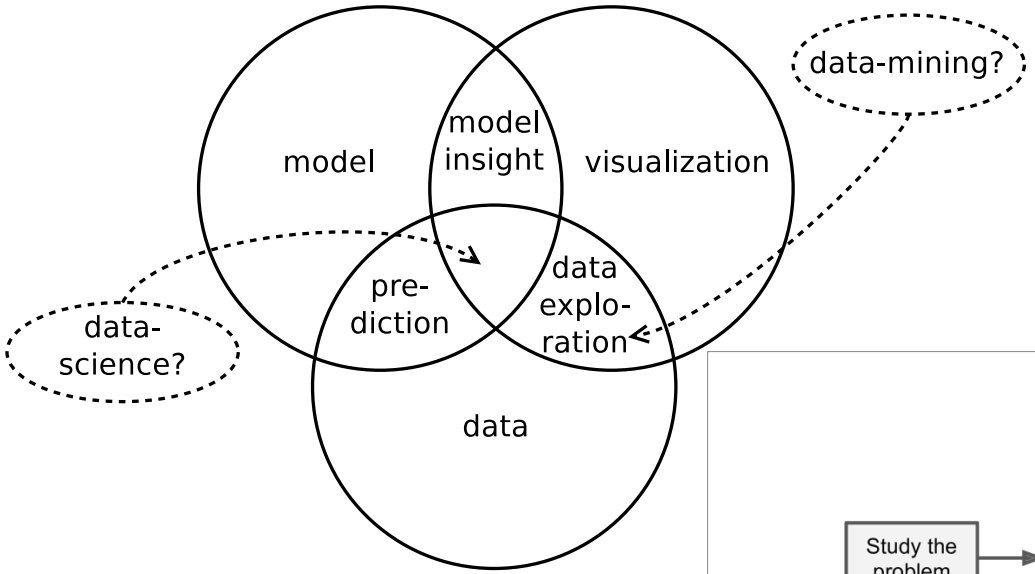






Machine learning baggrund

Machine learning: data science ekspert



- fra white-box domæne ekspert til black-box ML data scientist,
- stadig polytekniker:
  - math- og computer science,
  - pattern-recognition,
  - neurocompution,
  - datamining,
  - visualization,
  - etc..

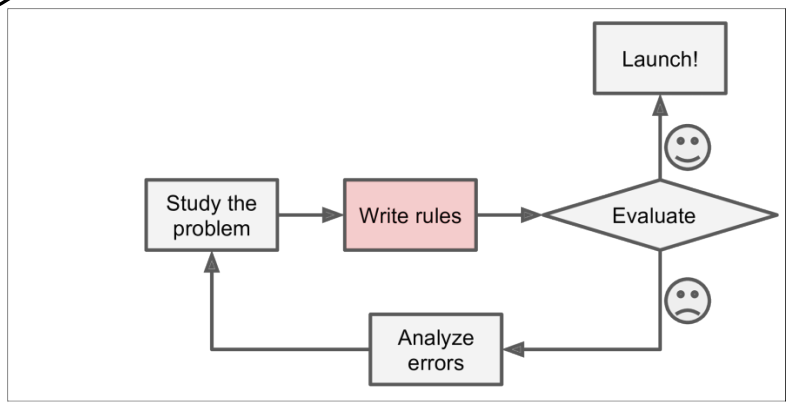


Figure 1-1. The traditional approach

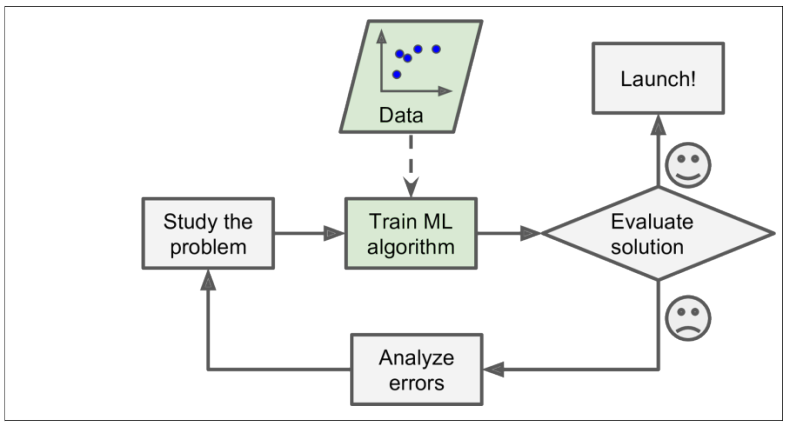
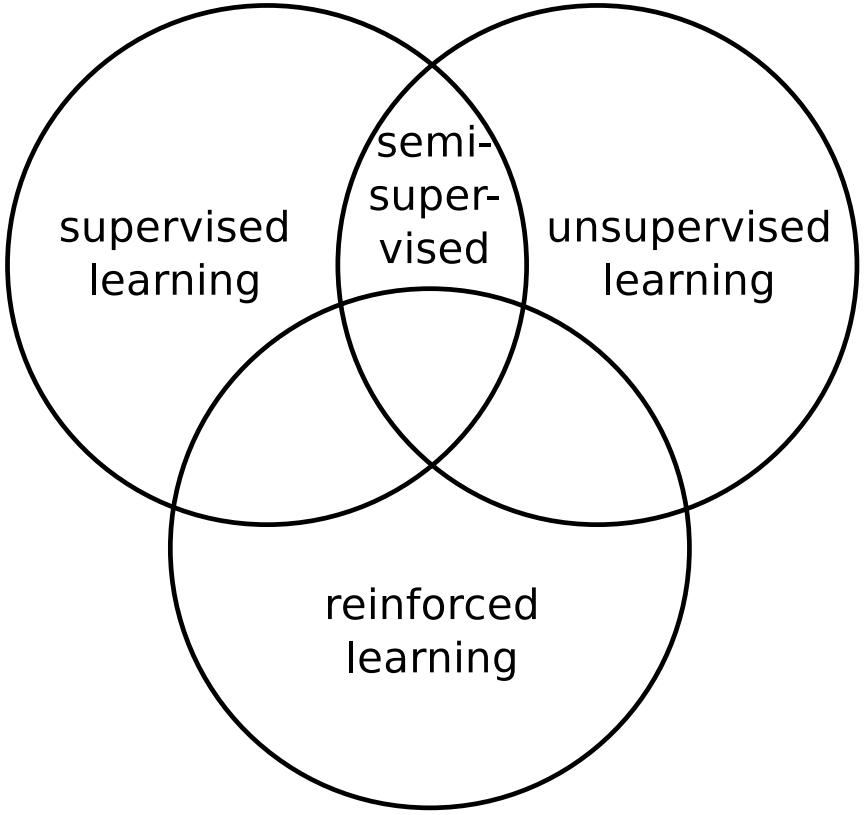


Figure 1-2. Machine Learning approach

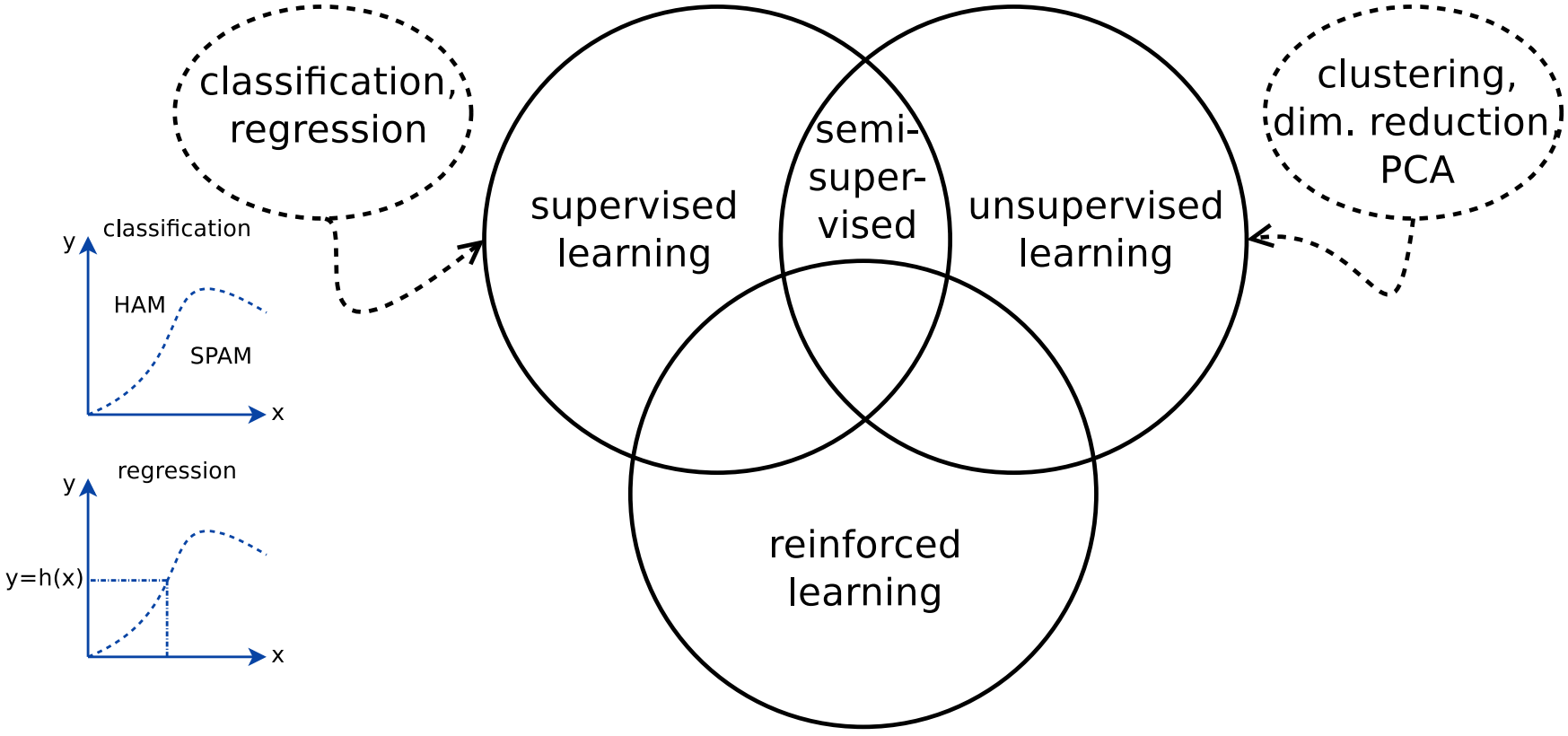
Machine learning taksonomi

Machine learning læringstyper



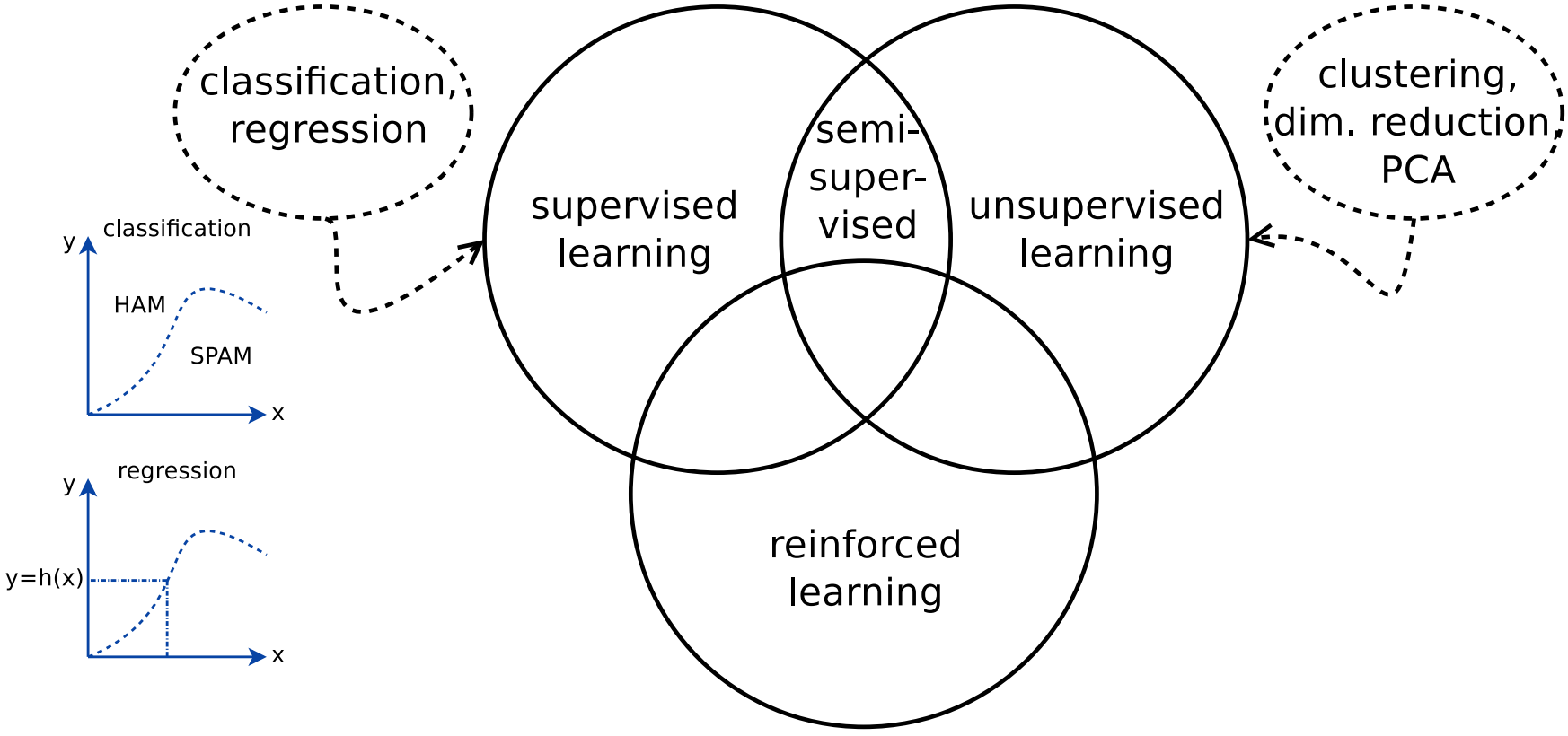
Machine learning taksonomi

Machine learning læringstyper



Machine learning taksonomi

Machine learning læringstyper



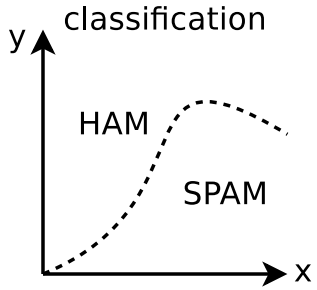
- I dette kursus:

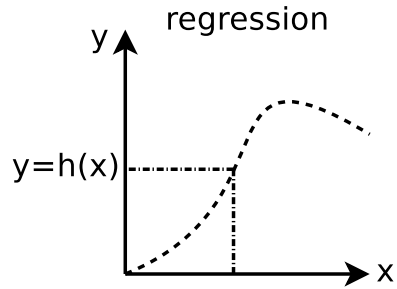
  kun om supervised- og unsupervised-learning.

Et ML end-to-end projekt

Emner fra [HOML] §2 'End-to-end Machine Learning'

- Læringstyper:
  - supervised (mest om dette i ITMAL),
  - unsupervised, [semisupervised], [reinforced learning].
- Output klasser:
  - classification (ham/spam),
  - regression ($h(x) = y$),





- Læring via data:
  - batch læring (al data),
  - [inkrementel læring (on-the-fly)].
- Prediktions/generaliserings model:
  - model-based (pattern-detection, byg intern model),
  - [instance-based (lær al data udenad)],
- Typiske ML fejl klasser:
  - for lidt trænings data (small-data, brug cross-validation),
  - sampling noise, sampling bias (ved manglende stratificering),
  - outliers og dårlig data (i big-data),
  - model og algoritme fejl: underfitting/overfitting.

Machine learning terminologi

X, **x**: input data matrix og vektor,

y, y: output data vektor og skalar,

θ: model parametre,

h: hypothesis funktion; typer af ML algos:

Bayes classifier, k-Nearest Neighbors, Linear Reg., Logistic Reg., SVM, Decision Trees, Random Forest, Neural Networks, k-Means, ...

$\mathbf{y}_{\text{true}}$: ground truth, til supervised learning,

$\mathbf{y}_{\text{pred}}$: predikteret værdi, aka $\hat{\mathbf{y}}$,

attribut: data type, f.eks. salgspris, dog anvendes 'feature' typisk i stedet for attribut!

λ, feature: data attribut plus value, f.eks. $\lambda_{\text{salgspris}} = \$42$,

J, L, loss fun.: loss/cost/error/objective funktion, som minimeres via fitting, jo lavere jo bedre et fit,

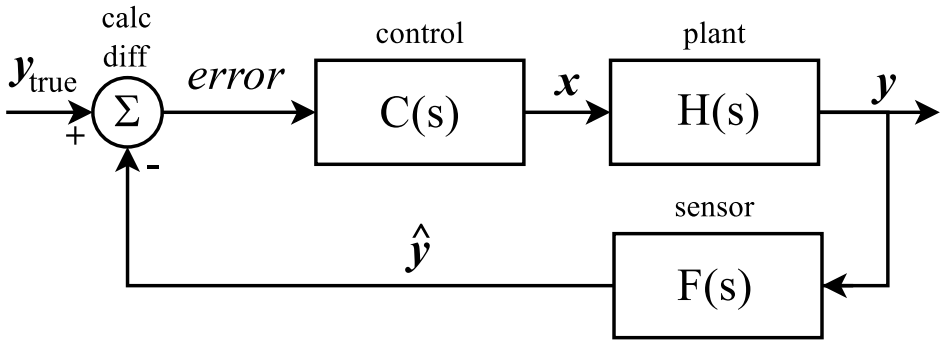
score fun.: score/fitness/goodness funktion, jo højere

performance-metric bruges typisk efter fit-minimeringingen til model inspektion og eftervalidering.

Supervised learning, blok diagram

Fra white-box til black-box

Almindelig white-box negativt feedback control block diagram, som for lineære og tids-uafhængige funktioner kan Laplace analyseres 'i det uendelige':



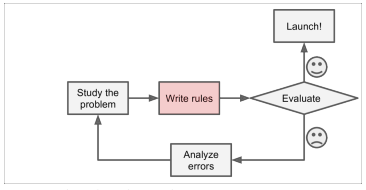


Figure 1-1. The traditional approach

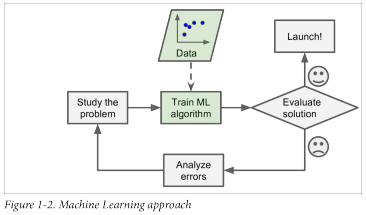
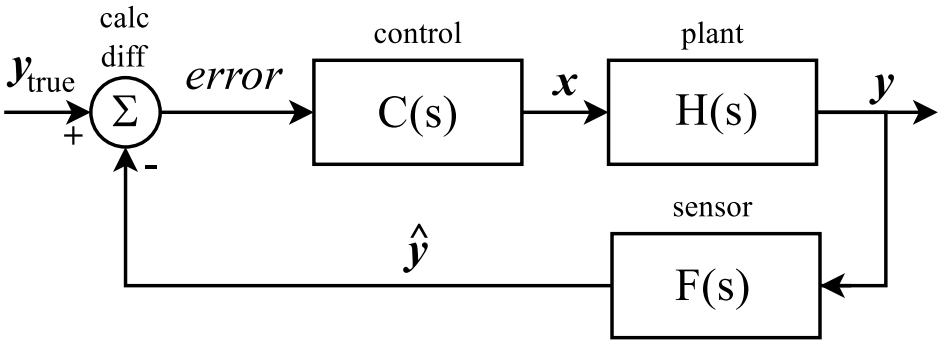


Figure 1-2. Machine Learning approach

Supervised learning, blok diagram

Fra white-box til black-box

Almindelig white-box negativt feedback control block diagram, som for lineære og tids-uafhængige funktioner kan Laplace analyseres 'i det uendelige':



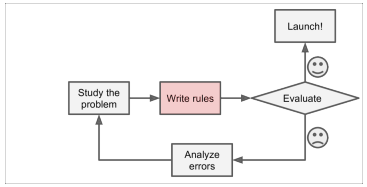


Figure 1-1. The traditional approach

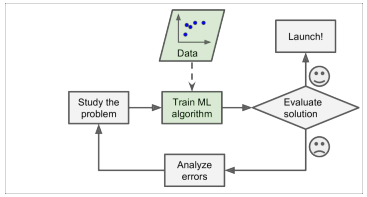
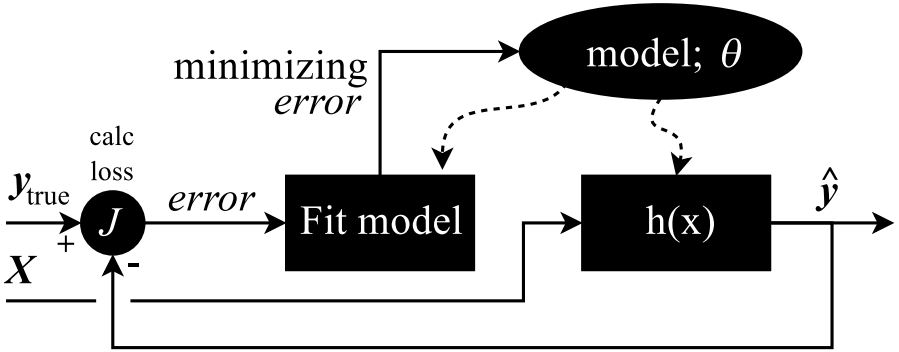


Figure 1-2. Machine Learning approach

Supervised machine learning block diagram:



Valg af: model/hypothesis funktion, *h*, that's is!

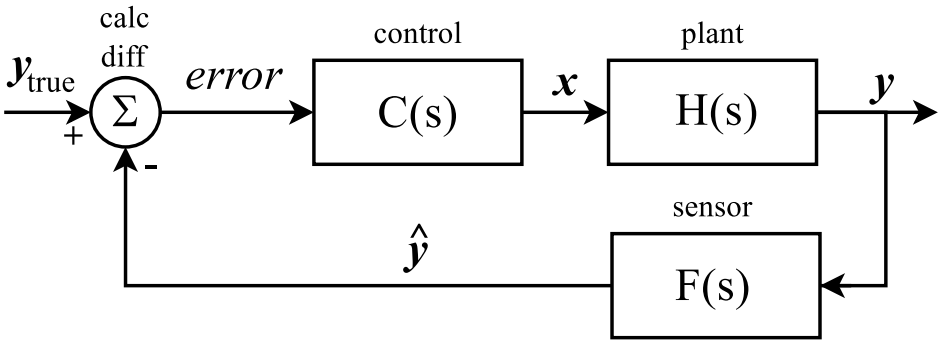
(excl. hyperparametre og valg af loss fun.)

Alt er nu black-box.

Supervised learning, blok diagram

Fra white-box til black-box

Almindelig white-box negativt feedback control block diagram, som for lineære og tids-uafhængige funktioner kan Laplace analyseres 'i det uendelige':



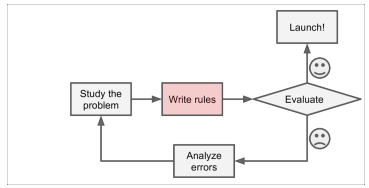


Figure 1-1. The traditional approach

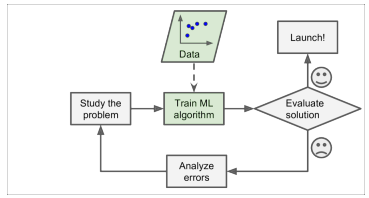
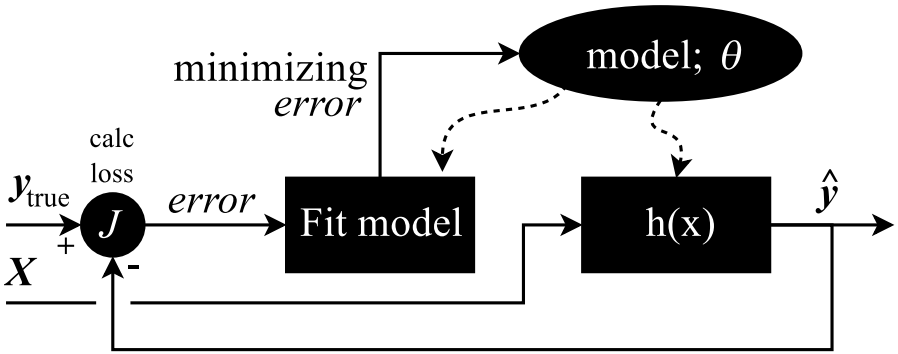
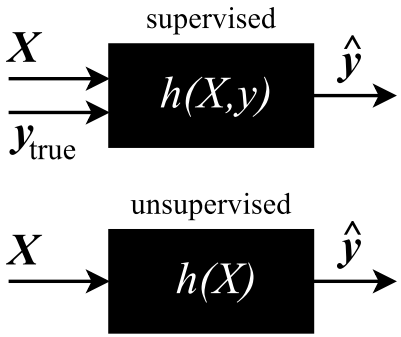


Figure 1-2. Machine Learning approach

Supervised machine learning block diagram:





Valg af: model/hypothesis funktion, *h*, that's is!

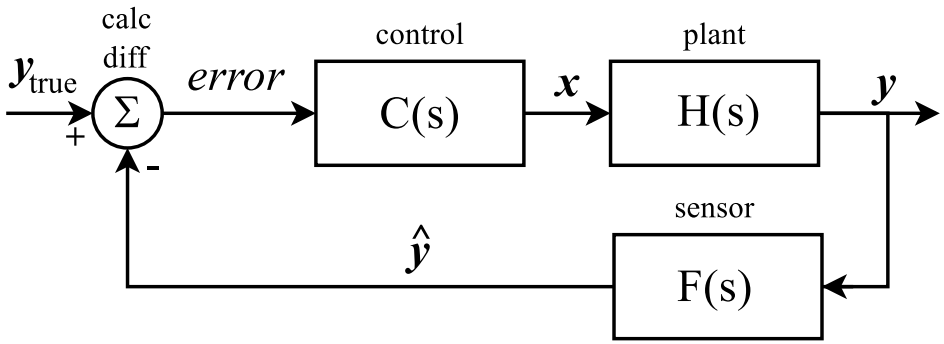
(excl. hyperparametre og valg af loss fun.)

Alt er nu black-box.

Supervised learning, blok diagram

Fra white-box til black-box

Almindelig white-box negativt feedback control block diagram, som for lineære og tids-uafhængige funktioner kan Laplace analyseres 'i det uendelige':



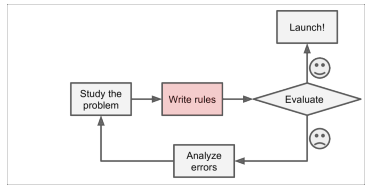


Figure 1-1. The traditional approach

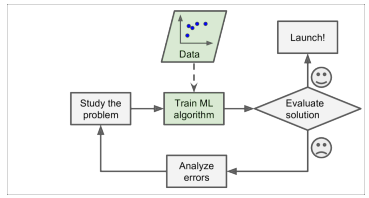
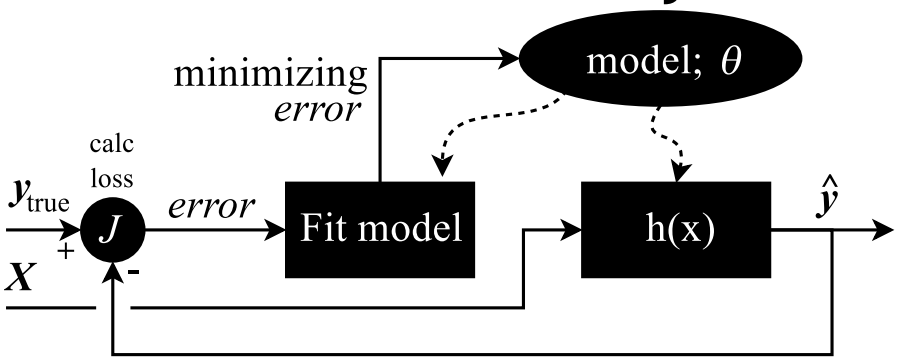
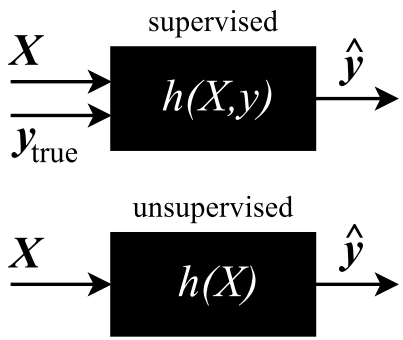


Figure 1-2. Machine Learning approach

Supervised machine learning block diagram:





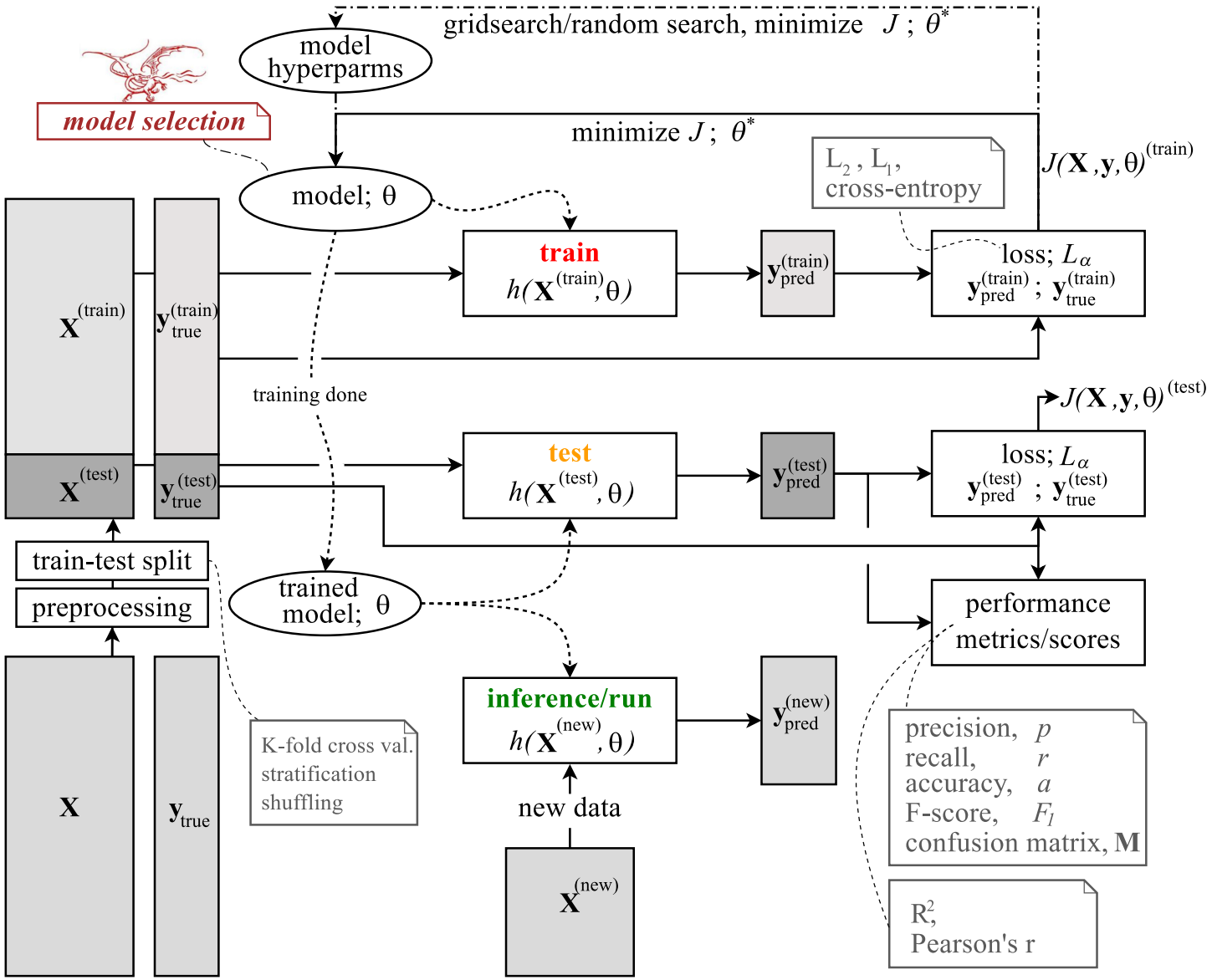
Valg af: model/hypothesis funktion, *h*, that's is!

(excl. hyperparametre og valg af loss fun.)

Alt er nu black-box. ⇒ **Holder ikke, kendskab til ML algo osv. nødvendig!**

Supervised learning, blok diagram

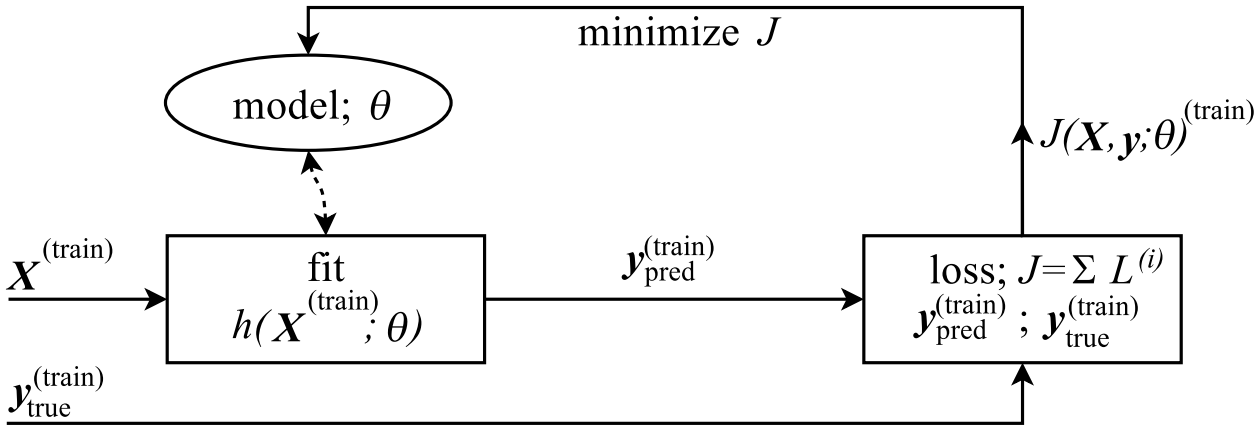
Sneak-preview af 'the full monty'...



NOTE: Kun et preview; vi går igennem detaljerne i figuren i de følgende lektioner.

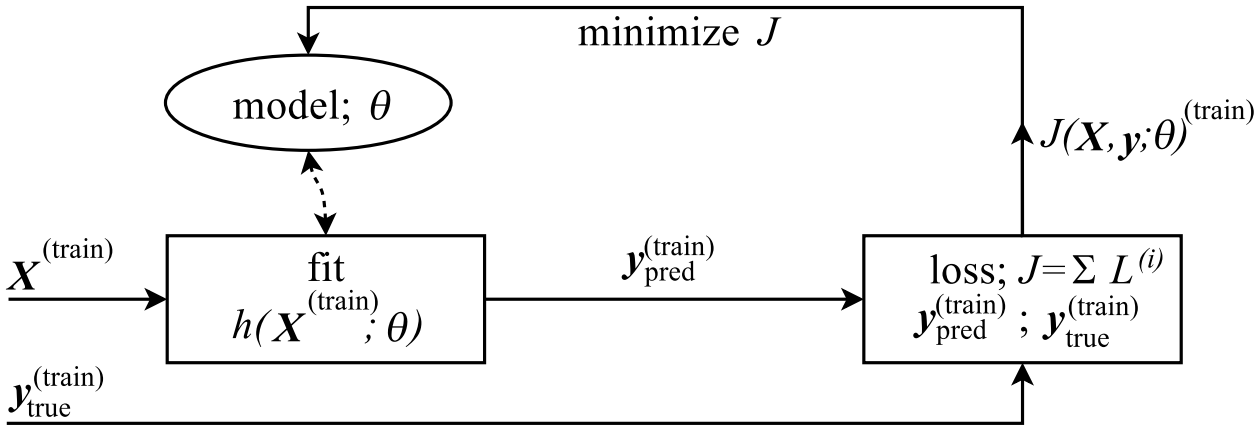
Q: `L01/intro.ipynb`

ML supervised learning data flow model: Training (fit).



Q: `L01/intro.ipynb`

ML supervised learning data flow model: Training (fit).

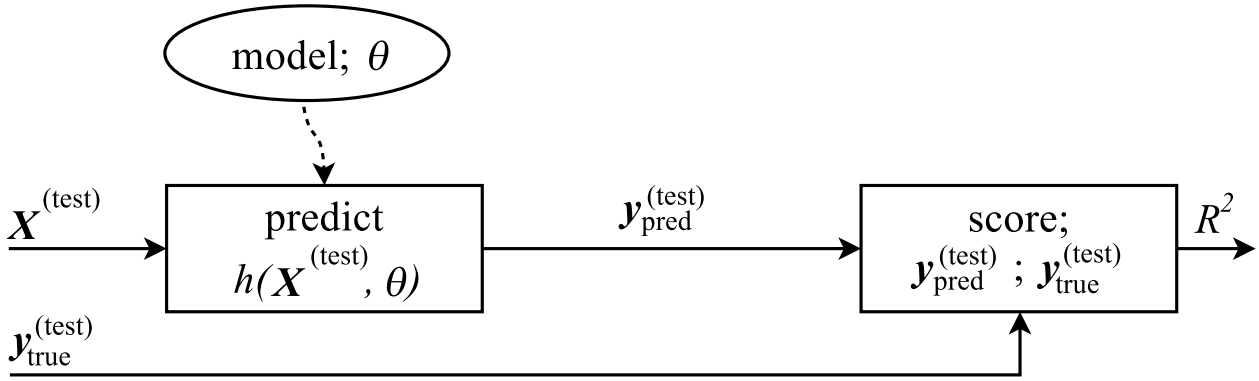


$\mathbf{X}^{(\text{train})}$	:	trænings matrix input data,
$\mathbf{x}^{(\text{train})}$	:	data input vector; $\mathbf{x} = [x_1, x_2, \cdots, x_d]$,
$\mathbf{y}_{\text{true}}^{(\text{train})}$	:	trænings input ground truth vektor,
$\mathbf{y}_{\text{pred}}^{(\text{train})}$	:	predikteret værdi for y, aka $\hat{y}$
$\boldsymbol{\theta}$	:	model parametre,
h	:	hypothesis funktion, aka. ML algoritmen,
$L^{(i)}$	:	loss funktion (individuel), $L^{(i)}(y_{\text{pred}}^{(i)}, y_{\text{true}}^{(i)})$
J	:	loss funktion (summeret), $J = \frac{1}{n}\sum_i L^{(i)}$.

NOTE: med **x** havende dimensionalitet d... mere om denne og loss funktioner i L02.

Q: L01/intro.ipynb

ML supervised learning data flow model: Regression and prediction



Øvelse:

- træn en lineær regression model,
  (Scikit-learn fit-predict interface),
- gå i detaljen med R^2 score funktionen,
  (NOTE: test data er lig train data for denne øvelse),
- check k-Nearest Neighbors modellen ud på data,
  sammenlign kNN-score med lineær regression-score.
- prøv en neuralt netværks-model på data
  (NOTE: den performer ekstrem dårligt!).

Q: `L01/intro.ipynb`

Opstart med Python, Scikit-learn og lidt matematik...

- Jupyter notebook: `intro.ipynb` [GITMAL].
- Scikit-learn `fit-predict` interface
- ML Algoritmer:
  - mange forskellige ML algoritmer, vi går pt. ikke i detaljen,
  - for denne opgave:
    - Linear Regression.
    - k-Nearest Neighbors.
    - Neural-network (virker dårligt til data!).
  - fokuserer på det overordnede ML flow.
- Loss og Scores funktioner
  - Loss: funktion, som ML algoritmen forsøger at minimere under `fit`.
  - Score: funktion, der fortæller noget om hvor godt et `predict` er, her afprøver vi R^2 (Coefficient of determination).