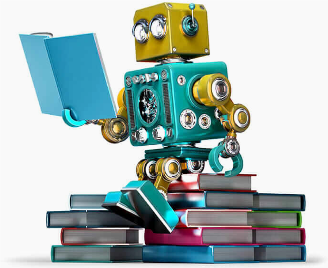
ECE 285

Machine Learning for Image Processing

Chapter I – Introduction

Charles Deledalle

July 9, 2019



(Source: Jeff Walsh)

Who am I?

- A visiting scholar from University of Bordeaux (France).
- Visiting UCSD since Jan 2017.
- PhD in signal processing (2011).
- Research in image processing / applied maths.
- Affiliated with CNRS (French scientific research institute).

- Email: `cdeledalle@ucsd.edu`
- `www.charles-deledalle.fr`

What is it about?

Machine learning / Deep learning

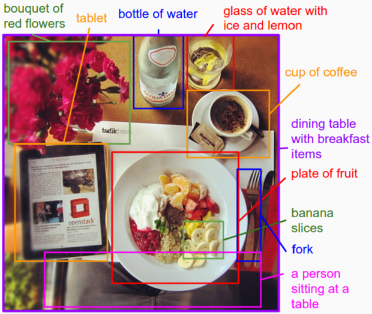
applied to

Image processing / Computer vision

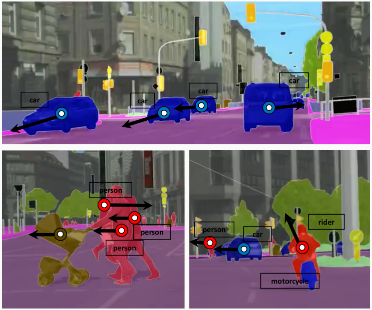
- A bit of theory (but not exhaustive), a bit of math (but not too much),
- Mainly: concepts, vocabulary, recent successful models and applications.

What?

What is it about? – Two examples



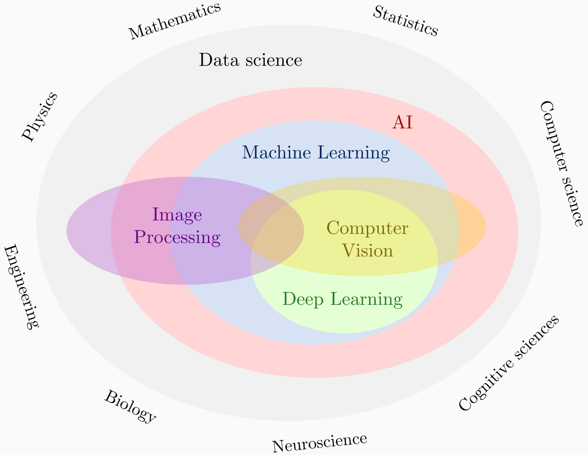
(Source: Luc et al., 2017)



(Karpathy & Fei-Fei, 2015)

(CV:) **Automatic extraction of high level information from images/videos**,
(ML:) **by learning from tons of (annotated) examples**.

What is it about? – A multidisciplinary field



What? Syllabus

- **Introduction to image sciences and machine learning**
  - Examples of image processing and computer vision tasks,
  - Overview of learning problems, approaches and workflow.

- **Preliminaries to deep learning**
  - Perceptron, Artificial Neural Networks (NNs),
  - Backpropagation, Support Vector Machines.

- **Basics of deep learning**
  - Representation learning, auto-encoders, algorithmic recipes.

- **Applications**
  - Image classification
  - Object detection
  - Image captioning
  - Image generation
  - Super resolution
  - Style transfer

  $\Rightarrow$ Convolutional NNs, Recurrent NNs, Generative adversarial networks.

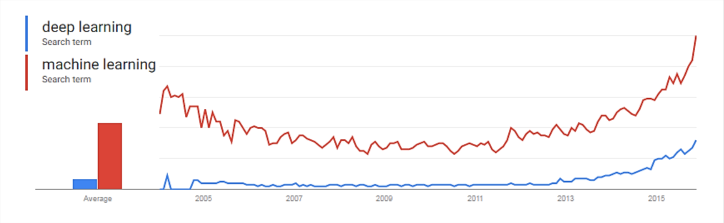
- **Labs and project using Python & PyTorch**.

Why machine learning / deep learning?

- In the past 10 years, machine learning and artificial intelligence have shown tremendous progress.
- The recent success can be attributed to:
  - Explosion of data,
  - Cheap computing cost – CPUs and GPUs,
  - Improvements of machine learning models.
- Much of the current excitement concerns a subfield of it called "deep learning".

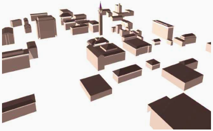
(Source: Poo Kuan Hoong)

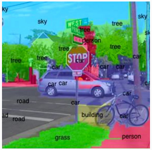
Google trends



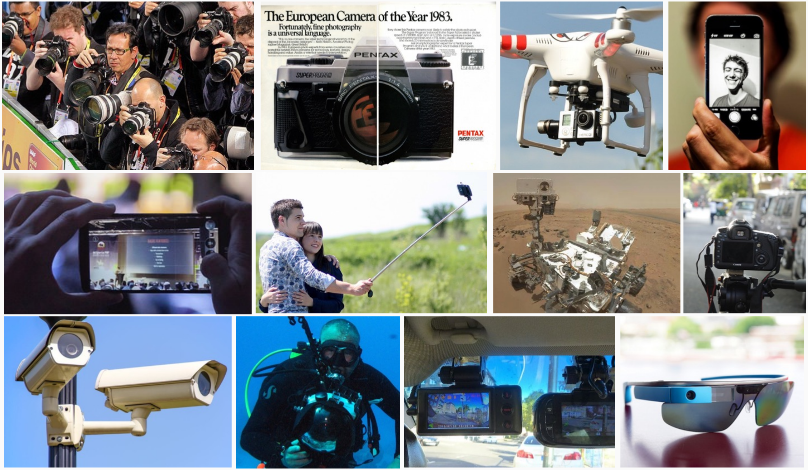
Why image processing / computer vision?

- Images become a major communication media.
- Images need to be analyzed automatically
  - Reduce the burden of human operators by teaching a computer to see.
- To produce images with artistic effect.
- Many applications: robotic, medical, video games, sport, smart cars, . . .





Why? More examples...



Top row, left to right:
Image by Augustas Didžgalvis; licensed under CC BY-SA 3.0; changes made
Image by Nesster is licensed under CC BY-SA 2.0
Image is CC0 1.0 public domain
Image is CC0 1.0 public domain

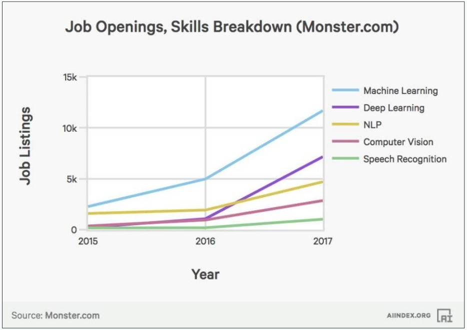
Middle row, left to right
Image by BGPHP Conference is licensed under CC BY 2.0; changes made
Image is CC0 1.0 public domain
Image by NASA is licensed under CC BY 2.0; chang
Image is CC0 1.0 public domain

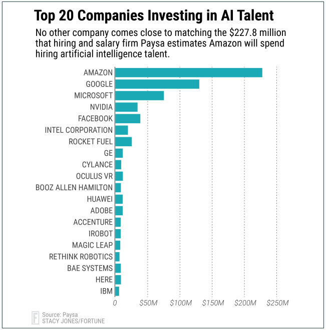
Bottom row, left to right
Image is CC0 1.0 public domain
Image by Derek Keats is licensed under CC BY 2.0; changes made
Image is public domain
Image by Ted Eytan is licensed under CC BY-SA 2.0; changes made

(Source: Stanford 2017'CS231n class)

What for?

- **Industry:** be able to use or implement latest machine learning techniques to solve image processing and computer vision tasks.

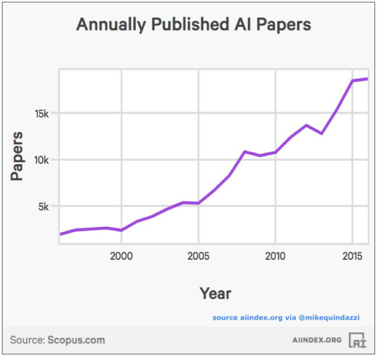


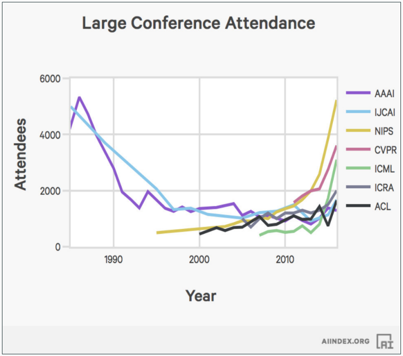


- **Big actors:** Amazon, Google, Microsoft, Facebook, . . .

What for?

- **Academic:** be able to read and understand latest research papers, and possibly publish new ones.





- **Big actors:** Stanford, New York U., U. of Montreal, U. of Toronto, . . .
- **Main conferences:** NIPS, CVPR, ICML, . . .

How? – Teaching staff

Instructor

Charles Deledalle

Teaching assistants

Sneha Gupta

Abhilash Kasarla

Anurag Paul

Inderjot Singh Saggu

How? – Schedule

- $30\times$ **50 min lectures** (10 weeks)
  - Mon/Wed/Fri 3:00-3:50pm
  - Room ~~CENTR 115~~ Ledden Auditorium (LEDDN)
- $5\times$ **2 hour optional labs every two weeks** (refer to Google's calendar)
  - Group 1: Fri 10am-12pm (lastnames from A to Kan)
  - Group 2: Tues 2-4pm (lastnames from Kar to Ra)
  - Group 3: Thurs 10am-12pm (lastnames from Ro to Z)
  - Jacobs Hall, Room 4309

Please, coordinate with your classmates to switch groups.

- **Office hours**
  - Charles Deledalle, Weekly on Tues 10am-12pm, Jacobs Hall 4808.
  - TAs, every two other weeks, TBA
- **Google calendar:** `https://tinyurl.com/y2gltvzs`

How? – Assignments / Project / Evaluation

- **4 assignments** in Python/Pytorch (individual) 40%
  - Don't wait for the lectures to start,
  - You can start doing them all now.
- **1 project** open-ended or to choose among 3 proposed subjects 30%
  - In groups of 3 or 4 (start looking for a group now),
  - Details to be announced in a couple of weeks.
- **3 quizzes** (∼45 mins each) 30%
  - Multiple choice on the topics of **all** previous lectures,
  - Dates are: April 24, May 17, June ~~10~~ 12,
  - No documents allowed.

How? – What assignments?

Assignment 1 (Backpropagation): Create from scratch a simple machine learning technique to recognize hand-written digits from 0 to 9.

$\longrightarrow$ **96% success**

Assignment 2 (CNNs and PyTorch): Develop a deep learning technique and learn how to use GPUs with PyTorch.

Improve your results to 98%!

How? – What assignments?

Assignment 3 (Transfer learning): Teach a program how to recognize bird species when only a small dataset is available.

$\longrightarrow$ **Mocking bird!**

How? – What assignments?

Assignment 4 (Image Denoising): Teach a program how to remove noise.

$\longrightarrow$

How? – Assignments and Project Deadlines

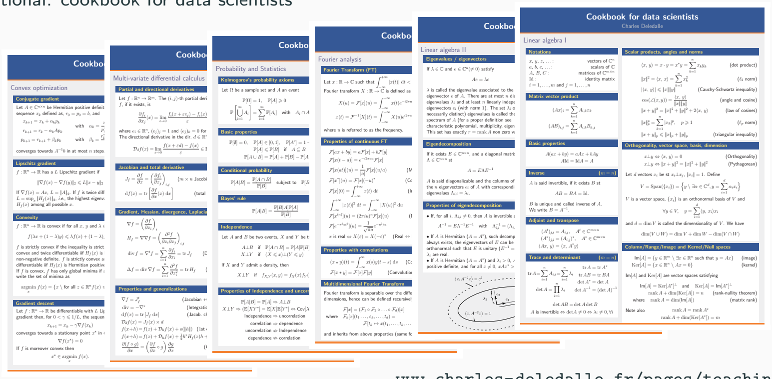
Calendar	Deadline
1. Assignment 0 – Python/Numpy/Matplotlib (Prereq)	optional
2. Assignment 1 – Backpropagation	April 17
3. Assignment 2 – CNNs and PyTorch	May 1
4. Assignment 3 – Transfer Learning	May 15
5. Assignment 4 – Image Denoising	May 29
6. Project	June ~~7~~ 9

Refer to the Google calendar: `https://tinyurl.com/y2gltvzs`

How? – Prerequisites

- Linear algebra + Differential calculus + Basics of optimization + Statistics/Probabilities
- Python programming (at least Assignment 0)

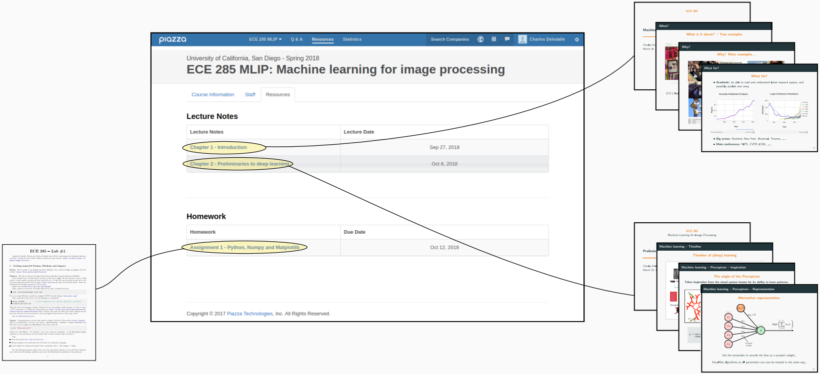
Optional: cookbook for data scientists



www.charles-deledalle.fr/pages/teaching/

How? – Piazza

https://piazza.com/ucsd/spring2019/ece285mlip



If you cannot get access to it contact me asap

at cdeledalle@ucsd.edu

(title: "[ECE285-MLIP][Piazza] Access issues").

Misc

Programming environment: Python/PyTorch/Jupyter

- We will use UCSD's DSMLP cluster with GPU/CUDA. Great but busy.
- We recommend you to install Conda/Python 3/Jupyter on your laptop.
- Please refer to additional documentations on Piazza.

Communication:

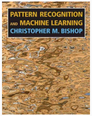
- All your emails **must have** a title starting with "`[ECE285-MLIP]`"
  $\rightarrow$ or it will end up in my spam/trash.

  Note: "`[ECE 285-MLIP]`", "`[ece285 MLIP]`", "`(ECE285MLIP)`" are invalid!

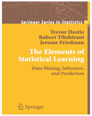
- But avoid emails, use Piazza to communicate instead.
- For questions that may interest everyone else, post on Piazza forums.

Some references

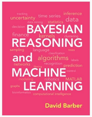
Reference books



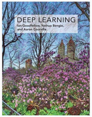
C. Bishop
Pattern recognition and Machine Learning
Springer, 2006



T. Hastie, R. Tibshirani, J. Friedman
The Elements of Statistical Learning: Data Mining, Inference, and Prediction
Springer, 2009
`http://web.stanford.edu/~hastie/ElemStatLearn/`

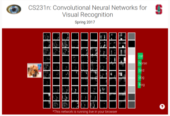


D. Barber
Bayesian Reasoning and Machine Learning
Cambridge University Press, 2012
`http://www.cs.ucl.ac.uk/staff/d.barber/brml/`

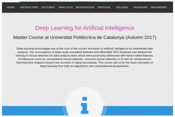


I. Goodfellow, Y. Bengio and A. Courville.
Deep Learning
MIT Press book, 2017
`http://www.deeplearningbook.org/`

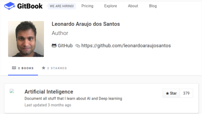
Reference online classes



Fei-Fei Li, Justin Johnson and Serena Yeung, 2017 (Stanford)
CS231n: Convolutional Neural Networks for Visual Recognition
`http://cs231n.stanford.edu`



Giró et al, 2017 (Catalonia)
Deep Learning for Artificial Intelligence
`https://telecombcn-dl.github.io/2017-dlai/`



Leonardo Araujo dos Santos.
Artificial Intelligence
`https://www.gitbook.com/@leonardoaraujosantos`

Image sciences

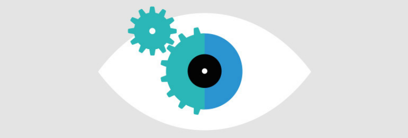
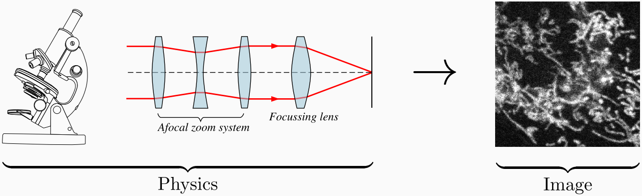


Image sciences

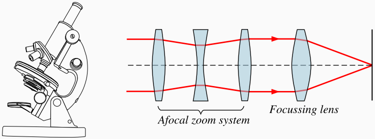
- Imaging:



Modeling the image formation process

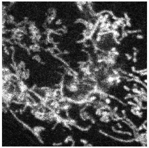
Image sciences

- Imaging:



Physics

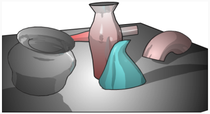
→



Image

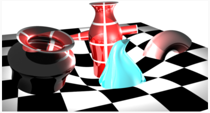
Modeling the image formation process

- Computer graphics:



Symbols

→



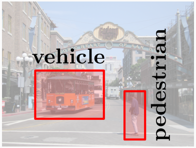
Image

Rendering images/videos from symbolic representation

Image sciences

- Computer vision:

$\underbrace{\qquad\qquad\qquad}_{\text{Image}}$ $\longrightarrow$



$\underbrace{\qquad\qquad\qquad}_{\text{Symbols}}$

Extracting information from images/videos

Image sciences

- Computer vision:

→

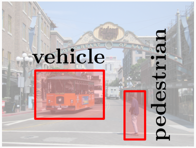


Image — Symbols

Extracting information from images/videos

- Image/Video processing:

→

Image — Image

Producing new images/videos from input images/videos

Spectrum from image processing to computer vision

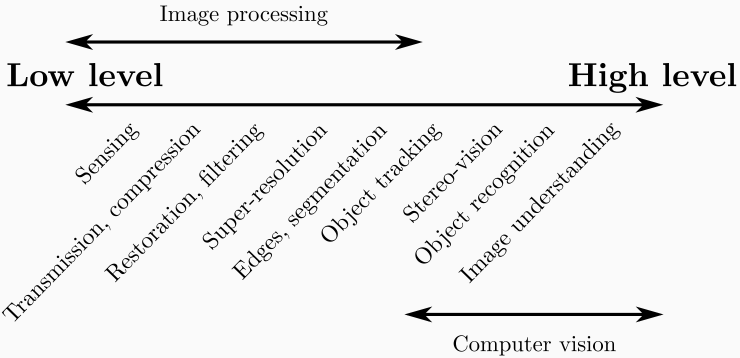


Image processing

Denoising

Enhancement

Compression

Feature detection

Inpainting

Super-resolution

(Source: Iasonas Kokkinos)

- Image processing: define a new image from an existing one
- Video processing: same problems + motion information

Image processing

Denoising

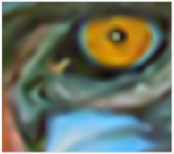
Enhancement

Compression

Feature detection

Inpainting

Super-resolution



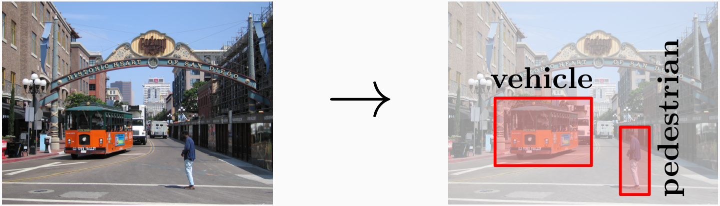
(Source: Iasonas Kokkinos)

- Image processing: define a new image from an existing one
- Video processing: same problems + motion information

Computer vision

Definition (The British Machine Vision Association)

Computer vision (CV) is concerned with the automatic extraction, analysis and understanding of useful information from a single image or a sequence of images.



$\underbrace{\hspace{6em}}_{\text{Image}}$ $\underbrace{\hspace{6em}}_{\text{Symbols}}$

CV is a subfield of Artificial Intelligence.

Computer vision – Artificial Intelligence (AI)

Definition (Collins dictionary)

artificial intelligence, *noun*: type of computer technology which is concerned with making machines work in an intelligent way, similar to the way that the human mind works.

Definition (Oxford dictionary)

artificial intelligence, *noun*: the theory and development of computer systems able to perform tasks normally requiring human intelligence, such as visual perception, speech recognition, decision-making, and translation.

Remark:

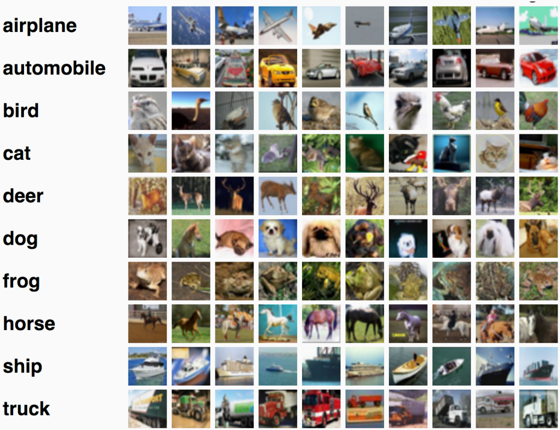
CV is a subfield of AI,

CV's new very best friend is machine learning (ML),

ML is also a subfield of AI,

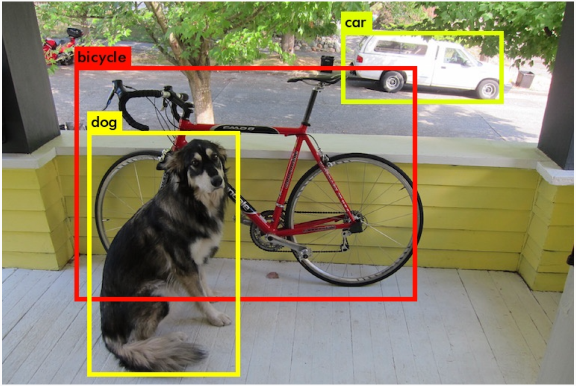
but not all computer vision algorithms are ML.

Computer vision – Image classification



Goal: to assign a given image into one of the predefined classes.

Computer vision – Object detection



(Source: Joseph Redmon)

Goal: to detect instances of objects of a certain class (such as human).

Computer vision – Image segmentation

(Source: Abhijit Kundu)

Goal: to partition an image into multiple segments such that pixels in a same segment share certain characteristics (color, texture or semantic).

Computer vision – Image captioning

"girl in pink dress is jumping in air."

"black and white dog jumps over bar."

"young girl in pink shirt is swinging on swing."

"man in blue wetsuit is surfing on wave."

"little girl is eating piece of cake."

"baseball player is throwing ball in game."

"woman is holding bunch of bananas."

"black cat is sitting on top of suitcase."

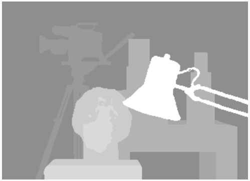
(Karpathy, Fei-Fei, CVPR, 2015)

Goal: to write a sentence that describes what is happening.

Computer vision – Depth estimation

→

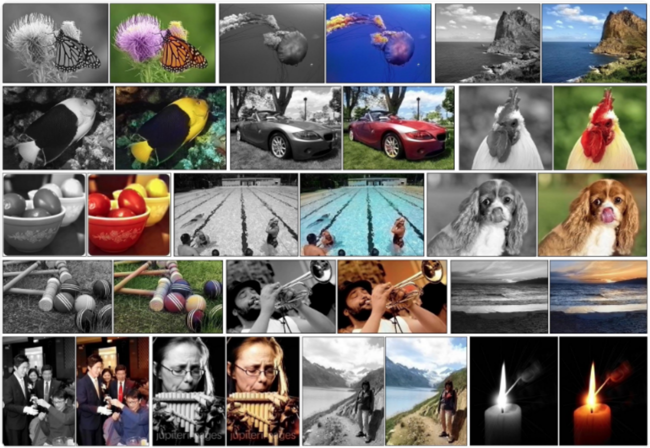
→



(Stereo-vision: from two images acquired with different views.)

Goal: to estimate a depth map from one, two or several frames.

Image colorization



(Source: Richard Zhang, Phillip Isola and Alexei A. Efros, 2016)

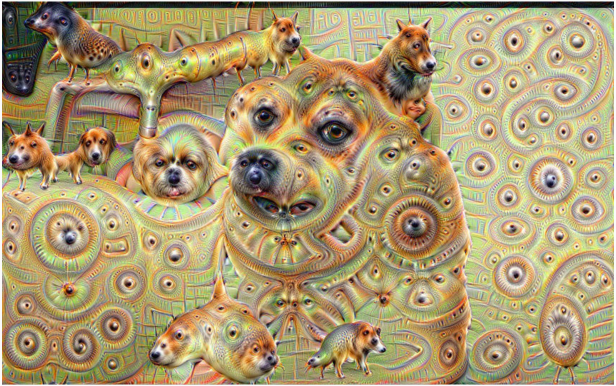
Goal: to add color to grayscale photographs.

Image generation

Generated images of bedrooms (Source: Alec Radford, Luke Metz, Soumith Chintala, 2015)

Goal: to automatically create realistic pictures of a given category.

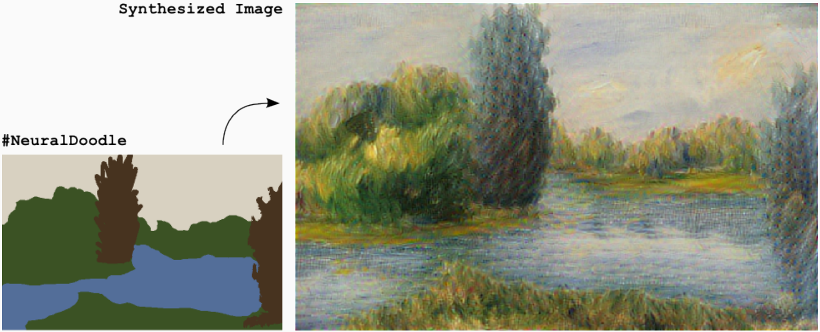
Image generation – DeepDream



(Source: Google Deep Dream, Mordvintsev et al., 2016)

Goal: to generate arbitrary ~~photo-realistic~~ artistic images, and understand/visualizing deep networks.

Image stylization



(Source: Neural Doodle, Champandard, 2016)

Goal: to create stylized images from rough sketches.

Style transfer

(Source: Gatys, Ecker and Bethge, 2015)

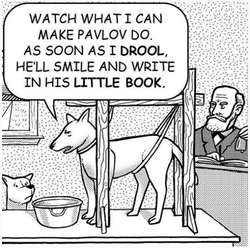
Goal: transfer the style of an image into another one.

Machine learning

What is learning?

Herbert Simon (Psychologist, 1916–2001):

Learning is any process by which a system improves performance from experience.



Pavlov's dog (Mark Stivers, 2003)

Tom Mitchell (Computer Scientist):

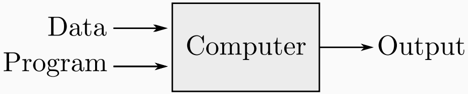
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Machine learning (ML)

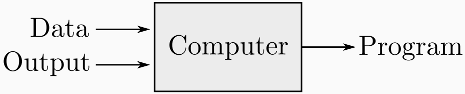
Definition

machine learning, *noun*: type of Artificial Intelligence that provides computers with the ability to learn without being explicitly programmed.

Traditional Programming



Machine Learning

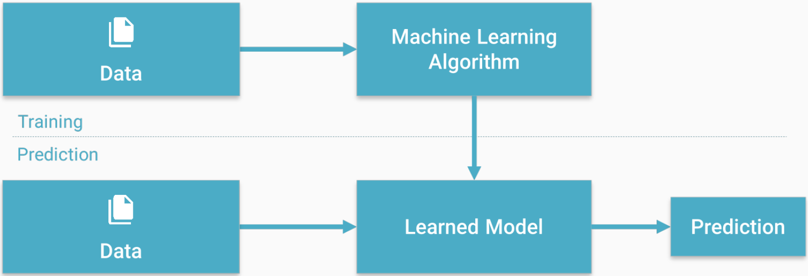


(Source: Pedro Domingos)

Machine learning (ML)

Provides various techniques that can learn from and make predictions on data.

Most of them follow the same general structure:



(Source: Lucas Masuch)

Learning from examples

3 main ingredients

1. Training set / examples:

$$\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$$

2. Machine or model:

$$\boldsymbol{x} \rightarrow \underbrace{f(\boldsymbol{x}; \theta)}_{\text{function / algorithm}} \rightarrow \underbrace{\boldsymbol{y}}_{\text{prediction}}$$

θ: parameters of the model

3. Loss, cost, objective function / energy:

$$\underset{\theta}{\operatorname{argmin}}\ E(\theta; \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$$

Learning from examples

Tools: $\begin{cases} \text{Data} & \leftrightarrow & \text{Statistics} \\ \text{Loss} & \leftrightarrow & \text{Optimization} \end{cases}$

Goal: to extract information from the training set

- relevant for the given task,
- relevant for other data of the same kind.

Can we learn everything? *i.e.*, **to be relevant for all problems?**

Terminology

Sample (Observation or Data): item to process (*e.g.*, classify). *Example: an individual, a document, a picture, a sound, a video. . .*

Features (Input): set of distinct traits that can be used to describe each sample in a quantitative manner. Represented as a multi-dimensional vector usually denoted by $\boldsymbol{x}$. *Example: size, weight, citizenship, . . .*

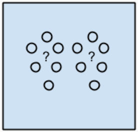
Training set: Set of data used to discover potentially predictive relationships.

Validation set: Set used to adjust the model hyperparameters.

Testing set: Set used to assess the performance of a model.

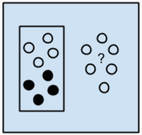
Label (Output): The class or outcome assigned to a sample. The actual prediction is often denoted by $\boldsymbol{y}$ and the desired/targeted class by $\boldsymbol{d}$ or $\boldsymbol{t}$. *Example: man/woman, wealth, education level, . . .*

Learning approaches



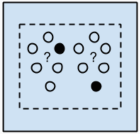
Unsupervised Learning Algorithms

Unsupervised learning: Discovering patterns in unlabeled data. *Example: cluster similar documents based on the text content.*



Supervised Learning Algorithms

Supervised learning: Learning with a labeled training set. *Example: email spam detector with training set of already labeled emails.*



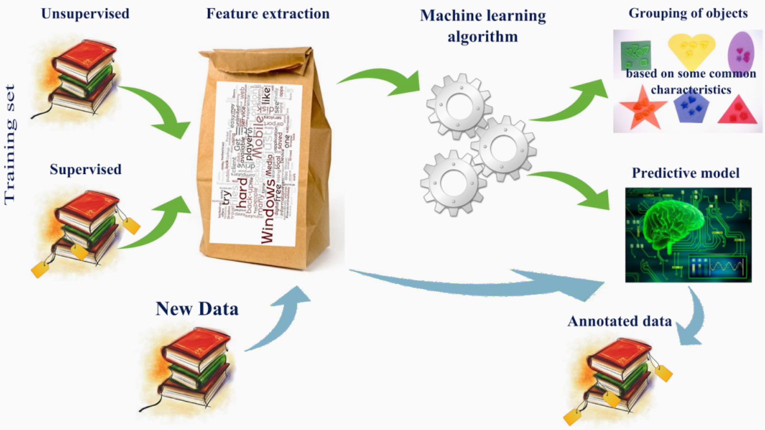
Semi-supervised Learning Algorithms

Semisupervised learning: Learning with a small amount of labeled data and a large amount of unlabeled data. *Example: web content and protein sequence classifications.*

Reinforcement learning: Learning based on feedback or reward. *Example: learn to play chess by winning or losing.*

(Source: Jason Brownlee and Lucas Masuch)

Machine learning workflow

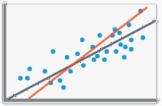


(Source: Michael Walker)

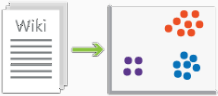
Machine learning – Problem types

Problem types

Classification
(supervised – predictive)



Regression
(supervised – predictive)



Clustering
(unsupervised – descriptive)

Anomaly Detection
(unsupervised– descriptive)

(Source: Lucas Masuch)

Unsupervised learning

Unsupervised learning

- **Training set:** $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)$ where $\boldsymbol{x}_i \in \mathbb{R}^d$.
- **Goal:** to find interesting structures in the data $\boldsymbol{X}$.

Examples:
- clustering,
- quantile estimation,
- outlier detection,
- dimensionality reduction.

Statistical point of view

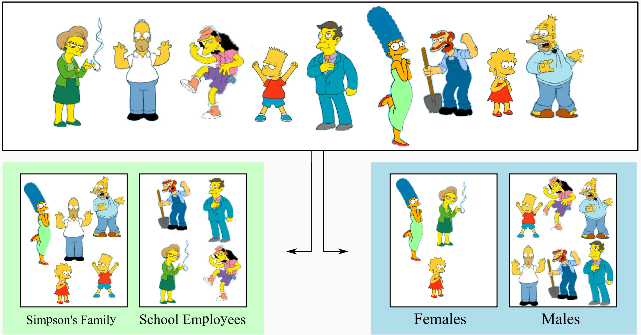
To estimate a density p which is likely to have generated $\boldsymbol{X}$, *i.e.*, such that

$$\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N \overset{\text{i.i.d}}{\sim} p$$

(i.i.d = identically and independently distributed).

Clustering

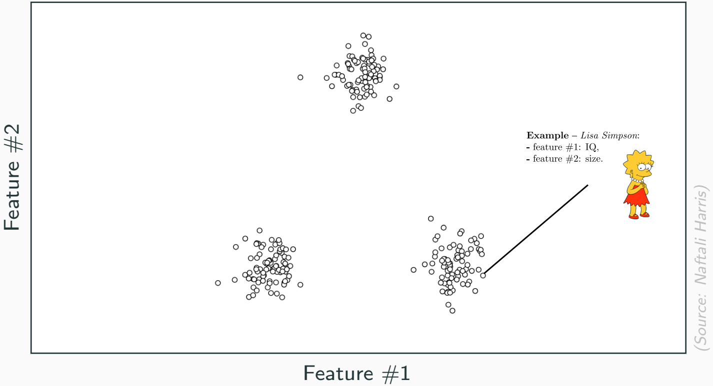
Clustering: group observations into "meaningful" groups.



(Source: Kasun Ranga Wijeweera)

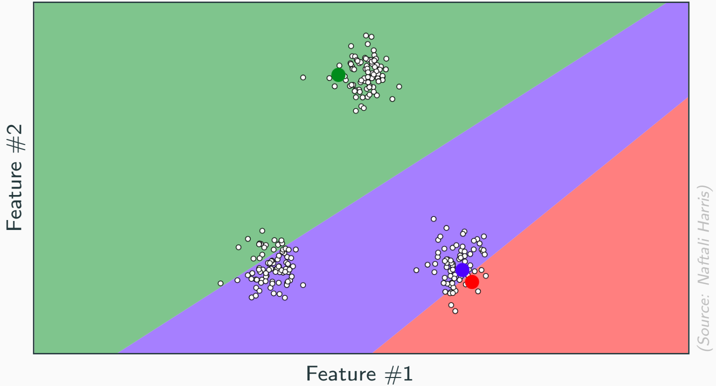
- Task of grouping a set of objects in such a way that objects in the same group (called a cluster) are more similar to each other.
- Popular ones are K-means clustering and Hierarchical clustering.

Clustering – K-means



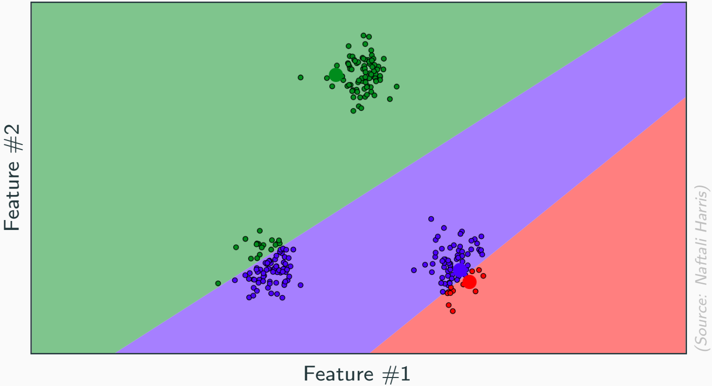
❶ Consider data in $\mathbb{R}^2$ spread on three different clusters,

Clustering – K-means



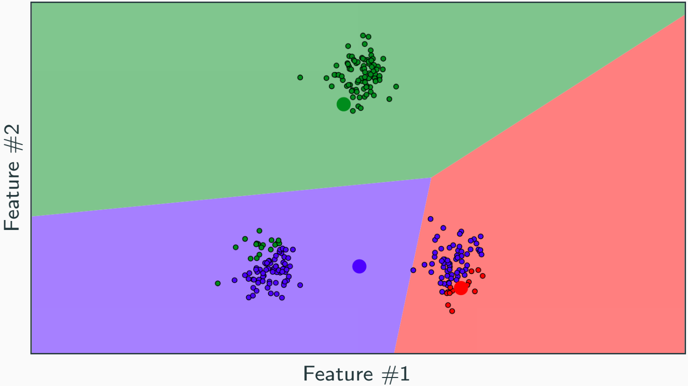
1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,

Clustering – K-means



1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,

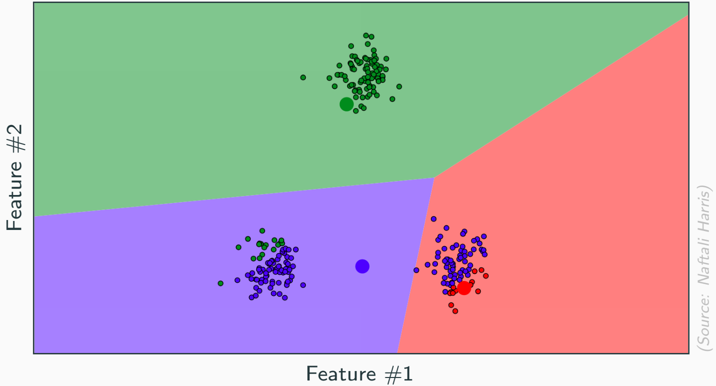
Clustering – K-means



(Source: Naftali Harris)

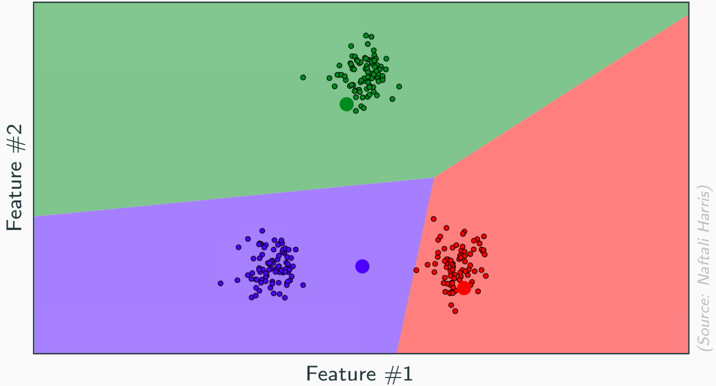
1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,
4. Update the centroids by taking the means within the clusters,

Clustering – K-means



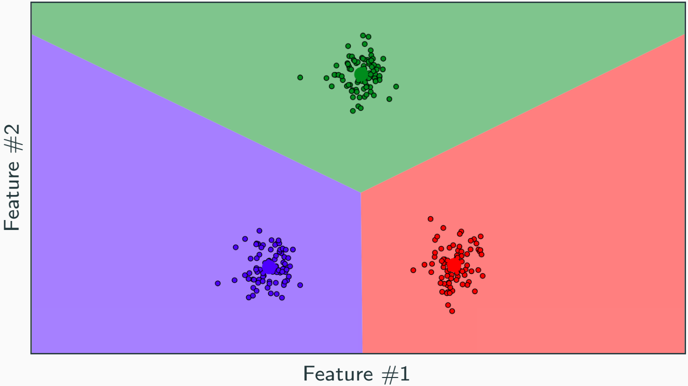
1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,
4. Update the centroids by taking the means within the clusters,
5. Go back to 3 until no more changes.

Clustering – K-means



1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,
4. Update the centroids by taking the means within the clusters,
5. Go back to 3 until no more changes.

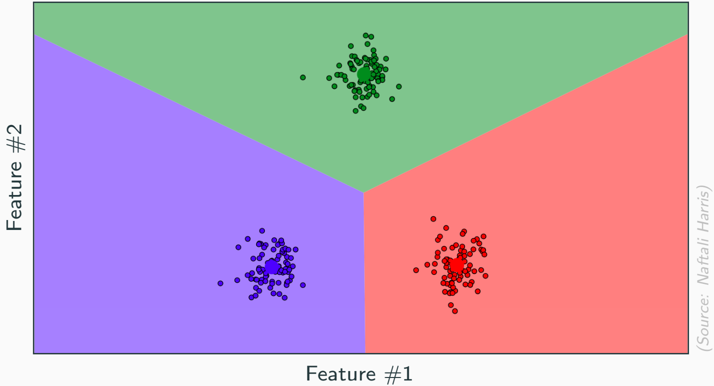
Clustering – K-means



(Source: Naftali Harris)

1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,
4. Update the centroids by taking the means within the clusters,
5. Go back to 3 until no more changes.

Clustering – K-means



1. Consider data in $\mathbb{R}^2$ spread on three different clusters,
2. Pick randomly $K = 3$ data points as cluster centroids,
3. Assign each data point to the class with closest centroid,
4. Update the centroids by taking the means within the clusters,
5. Go back to 3 until no more changes.

Clustering – K-means

- Optimal in terms of inter- and extra-class variability (loss),
- In practice, it requires much more iterations,
- Solutions **strongly depend on the initialization**,

  $\rightarrow$ Good initializations can be obtained by K-means++ strategy.
- The number of class K **is often unknown**:
  - usually found by trial and error,
  - or by cross-validation, AIC, BIC, ...
  - K too small/large $\Rightarrow$ under/overfitting. *(we will come back to this)*
- The data dimension d is often much larger than 2,

  $\rightarrow$ subject to the curse of dimensionality. *(we will also come back to this)*
- Vector quantization (VQ): the centroid substitutes all vectors of its class.

Supervised learning

Supervised learning

- **A training labeled set:** $(\boldsymbol{x}_1, d_1), (\boldsymbol{x}_2, d_2), \ldots, (\boldsymbol{x}_N, d_N)$.
- **Goal:** to learn a relevant mapping f st

$$y_i = f(\boldsymbol{x}_i; \theta) \approx d_i$$

Examples:
- classification (d is a categorical variable [a]),
- regression (d is a real variable),

a. can take one of a limited, and usually fixed, number of possible values.

Statistical point of view

- Discriminative models: to estimate the posterior distribution $p(d|\boldsymbol{x})$.
- Generative models: to estimate the likelihood $p(\boldsymbol{x}|d)$, or the joint distribution $p(\boldsymbol{x}, d)$.

Supervised learning – Bayesian inference

Bayes rule

In the case of a categorical variable d and a real vector $\boldsymbol{x}$

$$\mathbb{P}(d|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, d)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|d)\mathbb{P}(d)}{p(\boldsymbol{x})} = \frac{p(\boldsymbol{x}|d)\mathbb{P}(d)}{\sum_d p(\boldsymbol{x}|d)\mathbb{P}(d)}$$

- $\mathbb{P}(d|\boldsymbol{x})$: probability that $\boldsymbol{x}$ is of class d,
- $p(\boldsymbol{x}|d)$: distribution of $\boldsymbol{x}$ within class d,
- $\mathbb{P}(d)$: frequency of class d.

Example of final classifier: $f(\boldsymbol{x};\theta) = \underset{d}{\operatorname{argmax}}\, \mathbb{P}(d|\boldsymbol{x})$

Generative models carry more information:

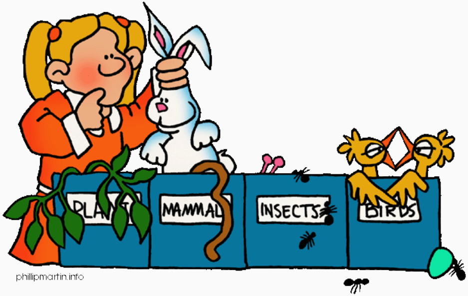
Learning $p(\boldsymbol{x}|d)$ and $\mathbb{P}(d)$ allows to deduce $\mathbb{P}(d|\boldsymbol{x})$.

But they often require many more parameters and more training data.

Discriminative models are usually easier to learn and thus more accurate.

Classification

Classification: predict class d from observation x.

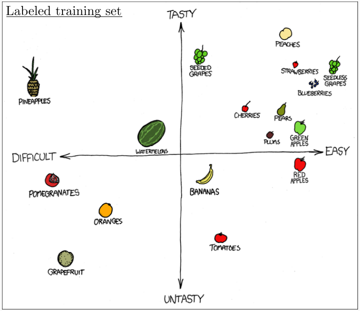


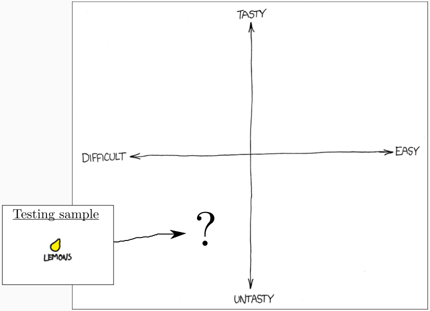
(Source: Philip Martin)

- Classify a document into a predefined category.
- Documents can be text, images, videos...
- Popular ones are Support Vector Machines and Artificial Neural Networks.

Regression

Regression (prediction): predict value(s) from observation.





- Statistical process for estimating the relationships among variables.
- Regression means to predict the output value using training data.
  $\rightarrow$ related to interpolation and extrapolation.
- Popular ones are linear least square and Artificial Neural Networks.

Classification vs Regression

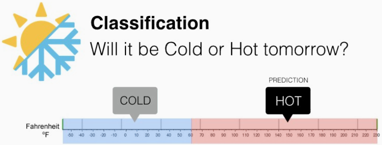
Classification

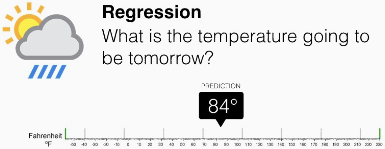
- Assign to a class
- Ex: a type of tumor is harmful or not
- Output is discrete/categorical

v.s

Regression

- Predict one or several output values
- Ex: what will be the house price?
- Output is a real number/continuous





(Source: Ali Reza Kohani)

Quiz, which one is which?

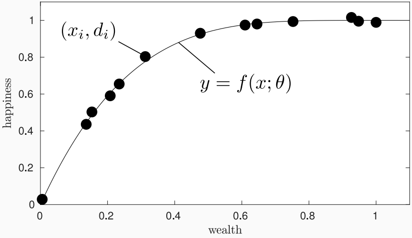
denoising, identification, verification, approximation.

Polynomial curve fitting

- Consider N individuals answering a survey asking for
  - their wealth: x_i
  - level of happiness: d_i
- We want to learn how to predict d_i (the desired output) from x_i as

$$d_i \approx y_i = f(x_i; \theta)$$

where f is the predictor and y_i denotes the predicted output.



Quiz

Supervised or unsupervised?

Classification or regression?

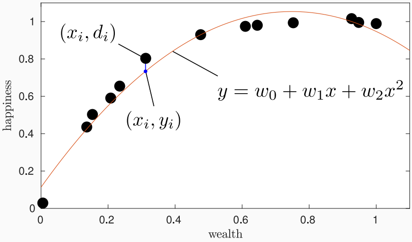
Polynomial curve fitting

- We assume that the relation is M-order polynomial

$$y_i = f(x_i; \boldsymbol{w}) = w_0 + w_1 x_i + w_2 x_i^2 + \ldots + w_M x_i^M = \sum_{j=0}^{M} w_j x_i^j$$

where $\boldsymbol{w} = (w_0, w_1, \ldots, w_M)^T$ are the polynomial coefficients.

- The (multi-dimensional) parameter θ is the vector $\boldsymbol{w}$.



Polynomial curve fitting

- Let $\boldsymbol{y} = (y_1, y_2, \ldots, y_N)^T$ and $\boldsymbol{X} = \begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^M \\ 1 & x_2 & x_2^2 & \ldots & x_2^M \\ \vdots & & & \vdots & \\ 1 & x_N & x_N^2 & \ldots & x_N^M \end{pmatrix}$, then

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{w} \quad \text{with} \quad \boldsymbol{w} = (w_0, w_1, \ldots, w_M)^T$$

- Polynomial curve fitting is linear regression.

  linear regression = linear relation between $\boldsymbol{y}$ and θ, even though f is non-linear.

- Standard procedures involve minimizing the sum of square errors (SSE)

$$E(\boldsymbol{w}) = \sum_{i=1}^{N} (y_i - d_i)^2 = \|\boldsymbol{y} - \boldsymbol{d}\|_2^2 = \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{d}\|_2^2$$

also called sum of square differences (SSD), or mean square error (MSE, when divided by N).

Linear regression + SSE $\longrightarrow$ Linear least square regression

Polynomial curve fitting

Recall: $E(\boldsymbol{w}) = \|\boldsymbol{X}\boldsymbol{w} - \boldsymbol{d}\|_2^2 = (\boldsymbol{X}\boldsymbol{w} - \boldsymbol{d})^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{d})$

Note that: $\nabla \boldsymbol{w}^T\boldsymbol{A}\boldsymbol{w} = (\boldsymbol{A} + \boldsymbol{A}^T)\boldsymbol{w}$ and $\nabla \boldsymbol{b}^T\boldsymbol{w} = \boldsymbol{b}$

- The solution is obtained by canceling the gradient

$$\nabla E(\boldsymbol{w}) = 0 \quad \Rightarrow \quad \underbrace{\boldsymbol{X}^T(\boldsymbol{X}\boldsymbol{w} - \boldsymbol{d}) = 0}_{\text{normal equation}}$$

- As soon as we have $N \geqslant M + 1$ distinct x_i, the solution is unique

$$\boldsymbol{w}^* = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{d}$$

- Otherwise, there is an infinite number of solutions.

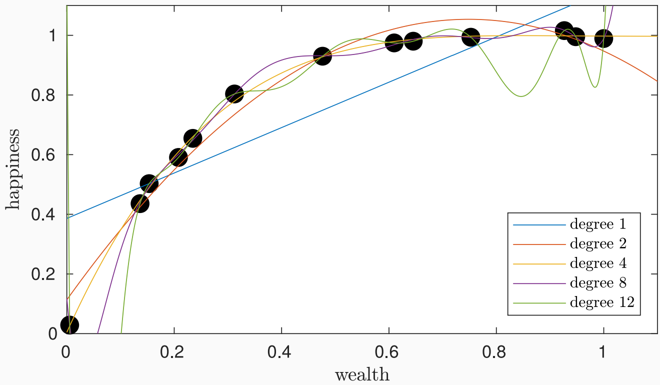
Polynomial curve fitting

• **Training data:**	answers to the survey
• **Model:**	polynomial function of degree M
• **Loss:**	sum of square errors
• **Machine learning algorithm:**	linear least square regression

The methodology for **Deep Learning** will be the exact same one.

The only difference is that the relation between $\boldsymbol{y}$ and θ will be (extremely) non-linear.

Polynomial curve fitting

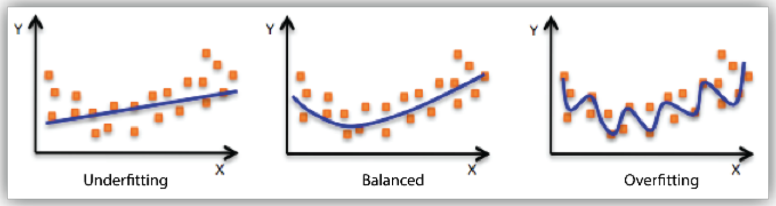


As M increases, unwanted oscillations appear (Runge's phenomenon), even though $N \geqslant M + 1$.

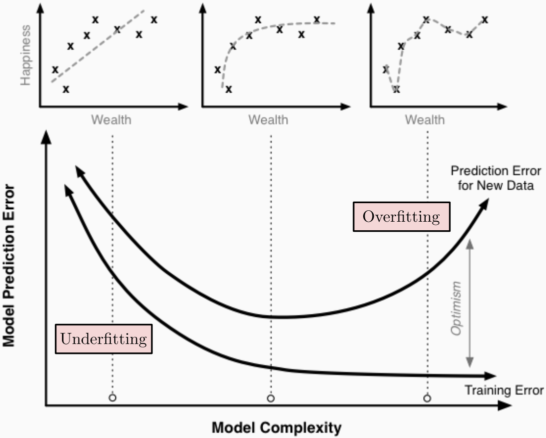
How to choose the degree M?

Difficulty of learning

- **Fit:** to explain the training samples,
  $\rightarrow$ requires some flexibility of the model.
- **Generalization:** to be accurate for samples outside the training dataset.
  $\rightarrow$ requires some rigidity of the model.



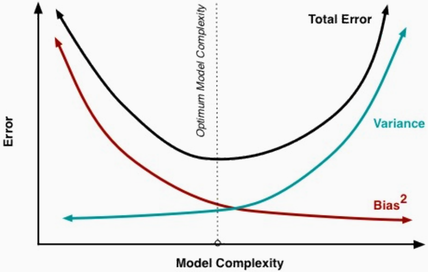
Difficulty of learning



Complexity: number of parameters, degrees of freedom, capacity, richness, flexibility, see also Vapnik–Chervonenkis (VC) dimension.

Difficulty of learning

Tradeoff: Underfitting/Overfitting Bias/Variance Data fit/Complexity



Variance: how much the predictions of my model on unseen data fluctuate if trained over different but similar training sets.

Bias: how off is the average of these predictions.

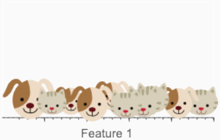
$$\mathrm{MSE} = \mathrm{Bias}^2 + \mathrm{Variance}$$

The tradeoff depends on several factors

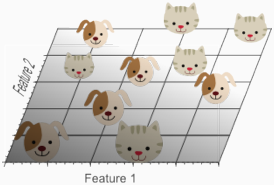
- Intrinsic complexity of the phenomenon to be predicted,
- Size of the training set: the larger the better,
- Size of the feature vectors: larger or smaller?

Curse of dimensionality

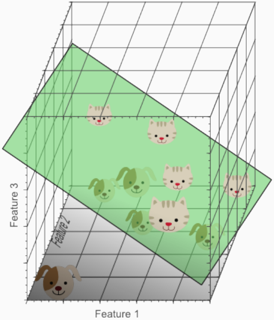
Is there a (hyper)plane that perfectly separates dogs from cats?



No perfect separation



No perfect separation



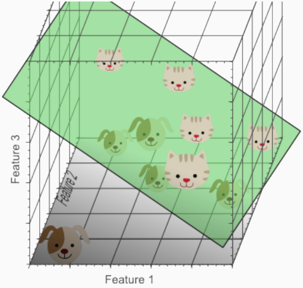
Linearly separable case

Looks like the more features we have, the better it is. But. . .

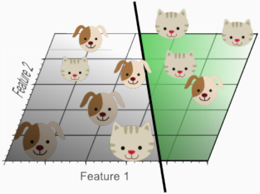
(Source: Vincent Spruyt)

Curse of dimensionality

Is there a (hyper)plane that perfectly separates dogs from cats?



Yes, but overfitting

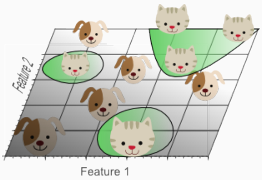


No, but better on unseen data

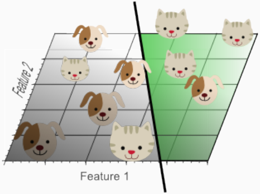
(Source: Vincent Spruyt)

Curse of dimensionality

Is there a (hyper)plane that perfectly separates dogs from cats?



Yes, but overfitting

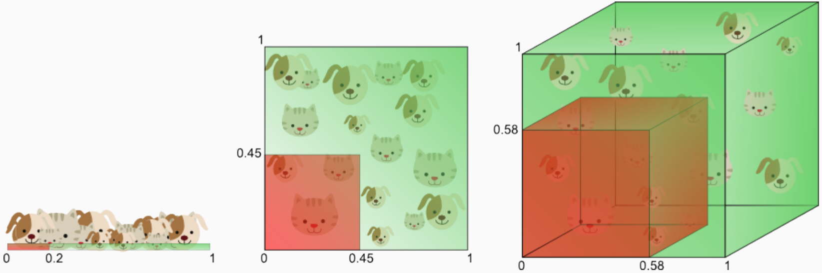


No, but better on unseen data

Why is that?

(Source: Vincent Spruyt)

Curse of dimensionality



The amount of training data needed to cover 20% of the feature range grows exponentially with the number of dimensions.

⇒ **Reducing the feature dimension is often favorable.**

"Many algorithms that work fine in low dimensions become intractable when the input is high-dimensional." Bellman, 1961.

(Source: Vincent Spruyt)

Feature engineering

- **Feature selection:** choice of distinct traits used to describe each sample in a quantitative manner.

  Ex: fruit $\rightarrow$ acidity, bitterness, size, weight, number of seeds, . . .

  Correlations between features: weight vs size, seeds vs bitterness, ...

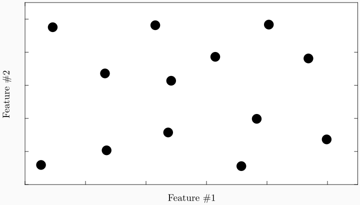
  $\Rightarrow$ **Information is redundant and can be summarized with less but more relevant features.**

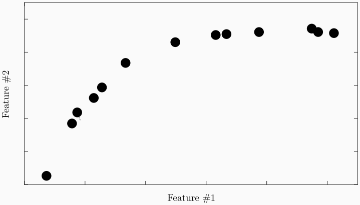
- **Feature extraction:** extract/generate new features from the initial set of features intended to be informative, non-redundant and facilitating the subsequent task.

  $\Rightarrow$ **Common procedure: Principal Component Analysis (PCA)**

Principal Component Analysis (PCA)

In most applications examples are not spread uniformly throughout the example space, but are concentrated on or near a low-dimensional subspace/manifold.

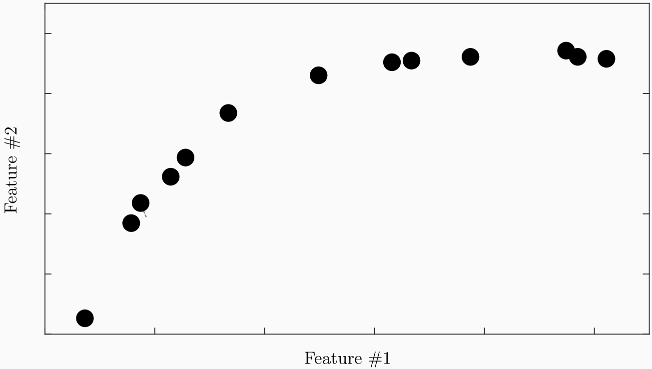




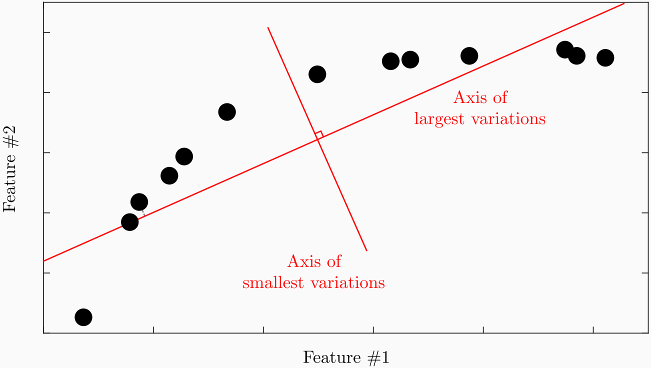
No correlations
$\Rightarrow$ Both features are informative,
$\Rightarrow$ No dimensionality reductions.

Strong correlation
$\Rightarrow$ Features "influence" each other,
$\Rightarrow$ Dimensionality reductions possible.

Principal Component Analysis (PCA)

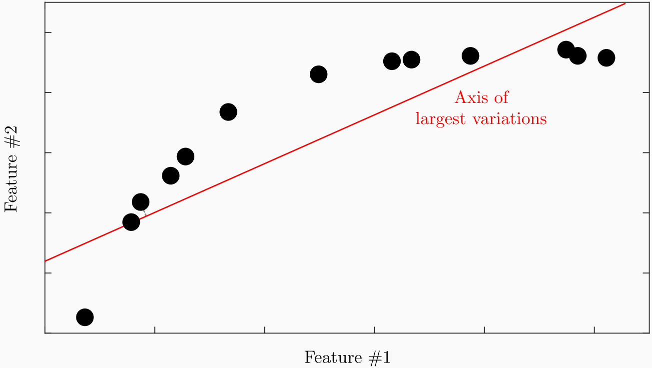


Principal Component Analysis (PCA)



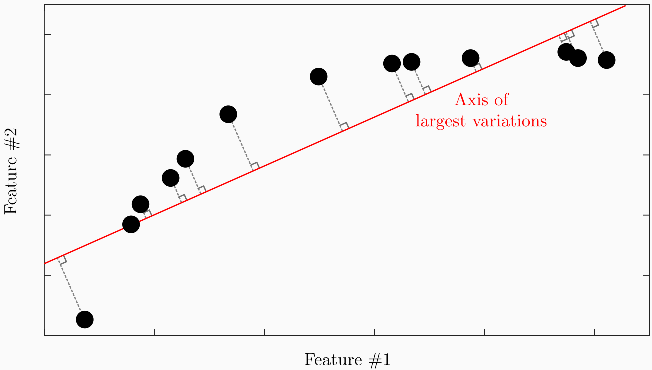
- Find the principal axes (eigenvectors of the covariance matrix),

Principal Component Analysis (PCA)



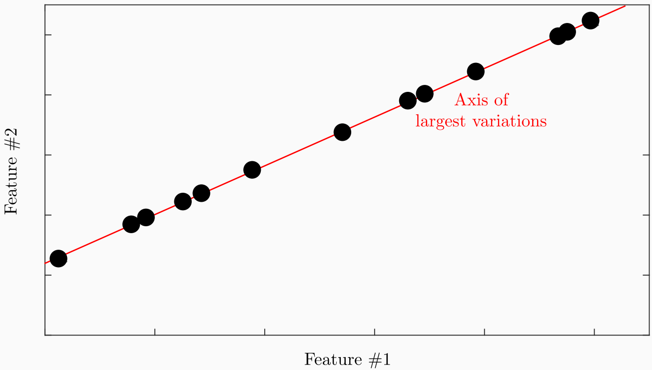
- Find the principal axes (eigenvectors of the covariance matrix),
- Keep the ones with largest variations (largest eigenvalues),

Principal Component Analysis (PCA)



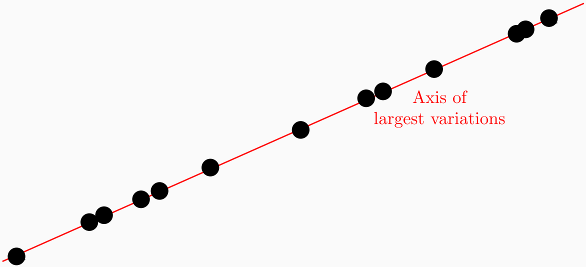
- Find the principal axes (eigenvectors of the covariance matrix),
- Keep the ones with largest variations (largest eigenvalues),
- Project the data on this low-dimensional space,

Principal Component Analysis (PCA)



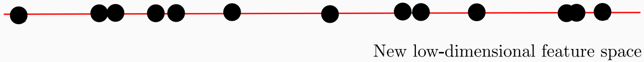
- Find the principal axes (eigenvectors of the covariance matrix),
- Keep the ones with largest variations (largest eigenvalues),
- Project the data on this low-dimensional space,

Principal Component Analysis (PCA)



- Find the principal axes (eigenvectors of the covariance matrix),
- Keep the ones with largest variations (largest eigenvalues),
- Project the data on this low-dimensional space,
- Change system of coordinate to reduce data dimension.

Principal Component Analysis (PCA)



- Find the principal axes (eigenvectors of the covariance matrix),
- Keep the ones with largest variations (largest eigenvalues),
- Project the data on this low-dimensional space,
- Change system of coordinate to reduce data dimension.

Principal Component Analysis (PCA)

- Find the principal axes of variations of $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N \in \mathbb{R}^d$:

$$\underbrace{\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}_i,}_{\text{mean (vector)}} \quad \underbrace{\boldsymbol{\Sigma} = \frac{1}{N}\sum_{i=1}^{N} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T,}_{\text{covariance (matrix)}} \qquad \underbrace{\boldsymbol{\Sigma} = \boldsymbol{V}^T \boldsymbol{\Lambda} \boldsymbol{V}}_{\substack{\text{eigen decomposition} \\ (\boldsymbol{V}\boldsymbol{V}^T = \boldsymbol{V}^T\boldsymbol{V} = \mathbf{Id}_d)}}$$

$$\boldsymbol{V} = (\underbrace{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d}_{\text{eigenvectors}}), \quad \boldsymbol{\Lambda} = \mathrm{diag}(\underbrace{\lambda_1, \ldots, \lambda_d}_{\text{eigenvalues}}) \quad \text{and} \quad \lambda_1 \geqslant \cdots \geqslant \lambda_d$$

- Keep the $K < d$ first dimensions: $\boldsymbol{V}_K = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_K) \in \mathbb{R}^{d \times K}$
- Project the data on this low-dimensional space:

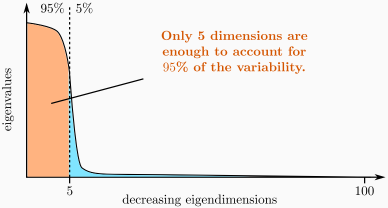
$$\tilde{\boldsymbol{x}}_i = \boldsymbol{\mu} + \sum_{k=1}^{K} \langle \boldsymbol{v}_k, \, \boldsymbol{x}_i - \boldsymbol{\mu} \rangle \boldsymbol{v}_k = \boldsymbol{\mu} + \boldsymbol{V}_K \boldsymbol{V}_K^T (\boldsymbol{x}_i - \boldsymbol{\mu}) \in \mathbb{R}^d$$

- Change system of coordinate to reduce data dimension:

$$\boldsymbol{h}_i = \boldsymbol{V}_K^T (\tilde{\boldsymbol{x}}_i - \boldsymbol{\mu}) = \boldsymbol{V}_K^T (\boldsymbol{x}_i - \boldsymbol{\mu}) \in \mathbb{R}^K$$

Principal Component Analysis (PCA)

- Typically: from hundreds to a few (one to ten) dimensions,
- Number K of dimensions often chosen to cover 95% of the variability:



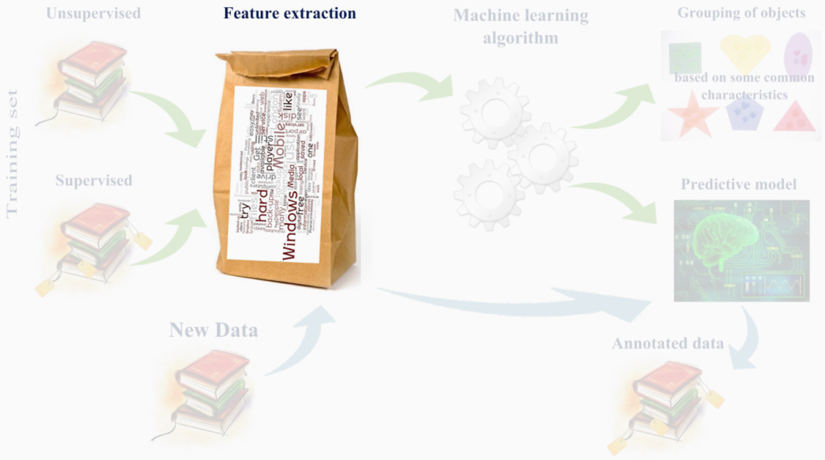
$$K = \min \left\{ K \setminus \frac{\sum_{k=1}^{K} \lambda_k}{\sum_{k=1}^{d} \lambda_k} > .95 \right\}$$

- **PCA is done on training data, not on testing data!**:
  - First, learn the low-dimensional subspace on training data only,
  - Then, project both the training and testing samples on this subspace,
  - It's an affine transform (translation, rotation, projection, rescaling):

$$\boldsymbol{h} = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b} \quad \text{(with} \quad \boldsymbol{W} = \boldsymbol{V}_K^T \quad \text{and} \quad \boldsymbol{b} = -\boldsymbol{V}_K^T \boldsymbol{\mu}\text{)}$$

Deep learning does something similar but in an (extremely) non-linear way.

What features for an image?



(Source: Michael Walker)

Image representation

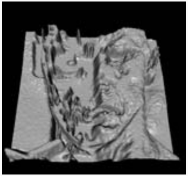
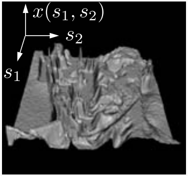
La Trahison des images, René Magritte, 1928
(Los Angeles County Museum of Art)

How do we represent images?

A two dimensional function

- Think of an image as a two dimensional function x.
- $x(s_1, s_2)$ gives the intensity at location (s_1, s_2).



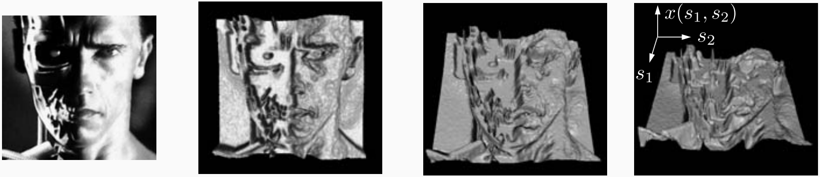
(Source: Steven Seitz)

Convention: larger values correspond to brighter content.

How do we represent images?

A two dimensional function

- Think of an image as a two dimensional function x.
- $x(s_1, s_2)$ gives the intensity at location (s_1, s_2).



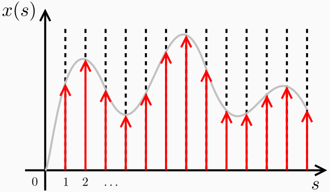
(Source: Steven Seitz)

Convention: larger values correspond to brighter content.

A color image is defined similarly as a 3 component vector-valued function:

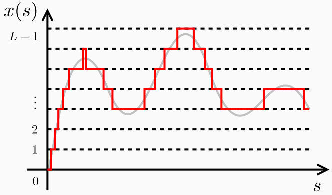
$$x(s_1, s_2) = \begin{pmatrix} r(s_1, s_2) \\ g(s_1, s_2) \\ b(s_1, s_2) \end{pmatrix}.$$

Digital imagery



Raster images

- Sampling: reduce the 2d continuous space to a discrete grid $\Omega \subseteq \mathbb{Z}^2$
- Gray level image: $\Omega \to \mathbb{R}$ (discrete position to gray level)
- Color image: $\Omega \to \mathbb{R}^3$ (discrete position to RGB)

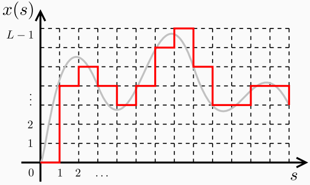


Bitmap image

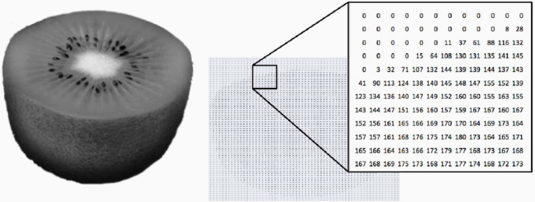
- Quantization: map each value to a discrete set $[0, L-1]$ of L values (*e.g.*, round to nearest integer)
- Often $L = 2^8 = 256$ (8bit images $\equiv$ `unsigned char`)
  - Gray level image: $\Omega \to [0, 255]$ $(255 = 2^8 - 1)$
  - Color image: $\Omega \to [0, 255]^3$
- Optional: assign instead an index to each pixel pointing to a color palette (format: `.png`, `.bmp`)

Digital imagery

- Digital images: sampling + quantization:



$\longrightarrow$ 8bit images can be seen as a matrix of integer values



We will refer to an element $s \in \Omega$ as a pixel location, $x(s)$ as a pixel value, and the pair $(s, x(s))$ as a pixel ("picture element").

Functional representation: $x : \Omega \subseteq \mathbb{Z}^d \to \mathbb{R}^K$

- d: dimension ($d = 2$ for pictures, $d = 3$ for videos, ...)
- K: number of channels ($K = 1$ monochrome, 3 colors, ...)
- $s = (i, j)$: pixel position in Ω
- $x(s) = x(i, j)$: pixel value(s) in $\mathbb{R}^K$

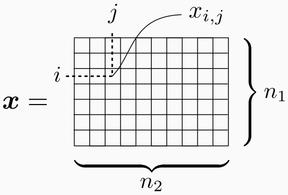
Image representation – Types of images – Digital imagery

Functional representation: $x : \Omega \subseteq \mathbb{Z}^d \to \mathbb{R}^K$

- d: dimension ($d = 2$ for pictures, $d = 3$ for videos, ...)
- K: number of channels ($K = 1$ monochrome, 3 colors, ...)
- $s = (i, j)$: pixel position in Ω
- $x(s) = x(i, j)$: pixel value(s) in $\mathbb{R}^K$

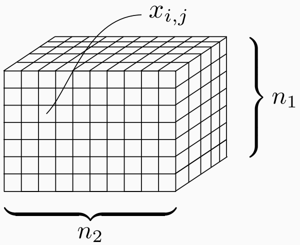
Array representation ($d = 2$): $\boldsymbol{x} \in (\mathbb{R}^K)^{n_1 \times n_2}$

- $n_1 \times n_2$: n_1: image height, and n_2: width
- $x_{i,j} \in \mathbb{R}^K$: pixel value(s) at position $s = (i, j)$: $x_{i,j} = x(i, j)$



For $d > 2$, we speak of multidimensional arrays: $\boldsymbol{x} \in (\mathbb{R}^K)^{n_1 \times \ldots \times n_d}$

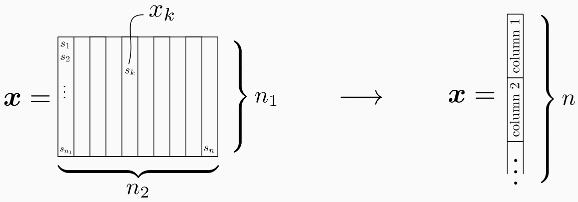
- d is called dimension, rank or order,

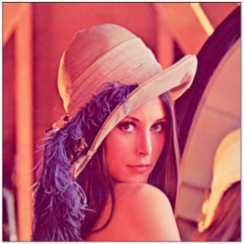
$\boldsymbol{x} =$ 

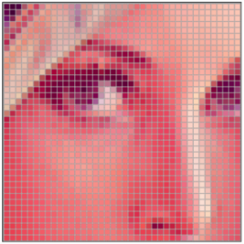
- In the deep learning community: they are referred to as tensors
(not to be confused with tensor fields or tensor imagery).

Vector representation: $\boldsymbol{x} \in (\mathbb{R}^K)^n$

- $n = n_1 \times n_2$: image size (number of pixels)
- $x_k \in \mathbb{R}^K$: value(s) of the k-th pixel at position s_k: $x_k = x(s_k)$







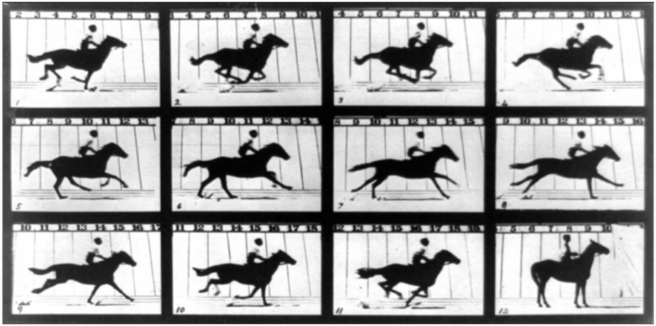
Color 2d image: $\Omega \subseteq \mathbb{Z}^2 \to [0, 255]^3$

- Red, Green, Blue (RGB), $K = 3$
- RGB: Usual colorspace for acquisition and display
- There exist other colorspaces for different purposes:

  HSV (Hue, Saturation, Value), YUV, YCbCr. . .

Spectral image: $\Omega \subseteq \mathbb{Z}^2 \to \mathbb{R}^K$

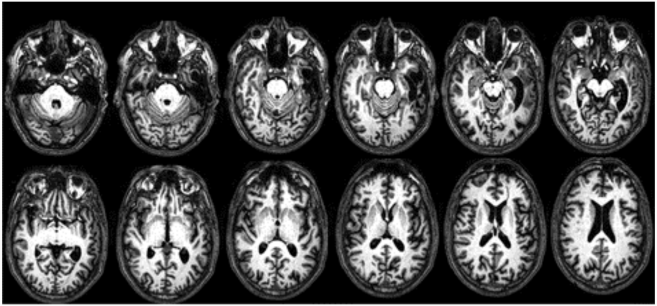
- Each of the K channels is a wavelength band
- For $K \approx 10$: multi-spectral imagery
- For $K \approx 200$: hyper-spectral imagery
- Used in astronomy, surveillance, mineralogy, agriculture, chemistry



The Horse in Motion (1878, Eadweard Muybridge)

Gray level video: $\Omega \subseteq \mathbb{Z}^3 \to \mathbb{R}$

- 2 dimensions for space
- 1 dimension for time



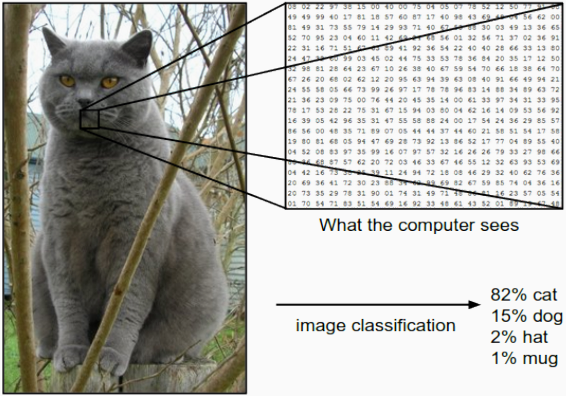
MRI slices at different depths

3d brain scan: $\Omega \subseteq \mathbb{Z}^3 \to \mathbb{C}$

- 3 dimensions for space
- 3d pixels are called voxels ("volume elements")

Image representation – Semantic gap

Semantic gap in CV tasks



Gap between tensor representation and its semantic content.

Old school computer vision

Semantic gap: initial representation of the data is too low-level,

Curse of dimensionality: reducing dimension is necessary for limited datasets,

Instead of considering images as a collection of pixel values (tensor),
we may consider other features/descriptors:

Designed from prior knowledge

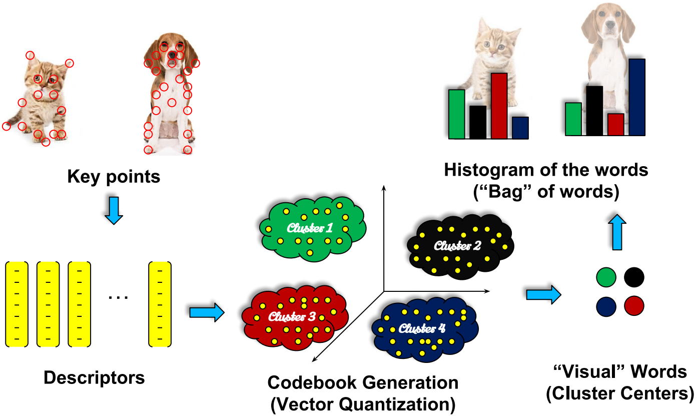
- Image edges,
- Color histogram,
- Local frequencies,
- High-level descriptor (SIFT).

Or learned by unsupervised learning

- Dimensionality reduction (PCA),
- Parameters of density distributions,
- Clustering of image regions,
- Membership to classes (GMM-EM).

Goal: Extract informative features, remove redundancy, reduce dimensionality, facilitating the subsequent learning task.

Example of a classical CV pipeline



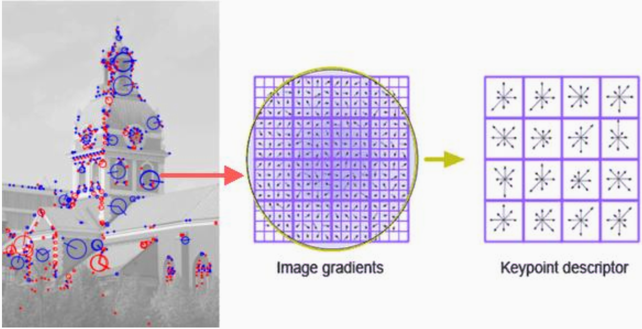
1. Identify "interesting" key points,
2. Extract "descriptors" from the interesting points,
3. Collect the descriptors to "describe" an image.

Key point detector

- Goal: to detect interesting points (without describing them).
- Method: to measure intensity changes in local sliding windows.
- Constraint: to be invariant to illumination, rotation, scale, viewpoint.
- Famous ones: Harris, Canny, DoG, LoG, DoH, . . .

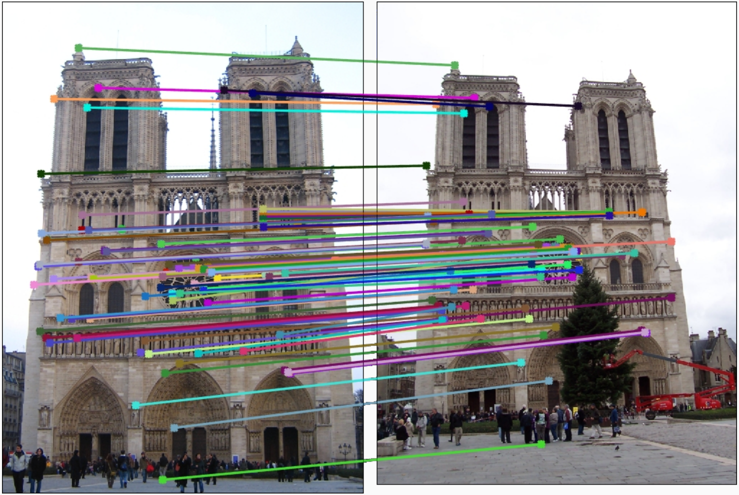
Scale-invariant feature transform (SIFT) (Lowe, 1999)



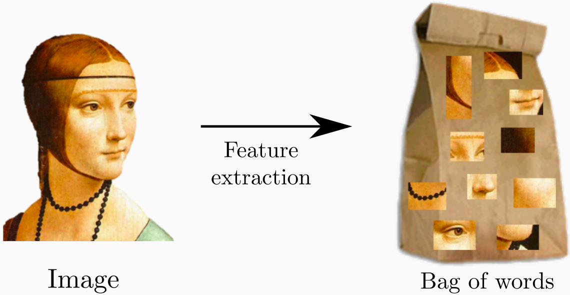
(Source: Ravimal Bandara)

- Goal: to provide a quantitative description at a given image location.
- Based on multi-scale analysis and histograms of local gradients.
- Robust to changes of scales, rotations, viewpoints, illuminations.
- Fast, efficient, very popular in the 2000s.
- Other famous descriptors: HoG, SURF, LBP, ORB, BRIEF, . . .

SIFT – Example: Object matching



Bags of words

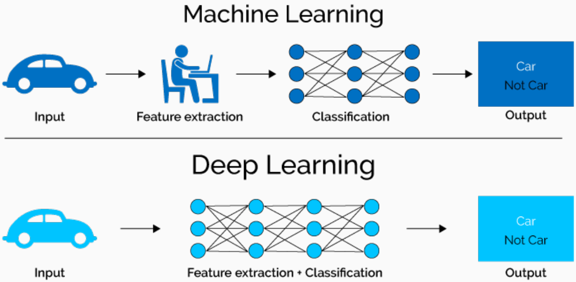


(Source: Rob Fergus & Svetlana Lazebnik)

Bag of words: vector of occurrence count of visual descriptors (often obtained after vector quantization).

Before deep learning: most computer vision tasks were relying on feature engineering and bags of words.

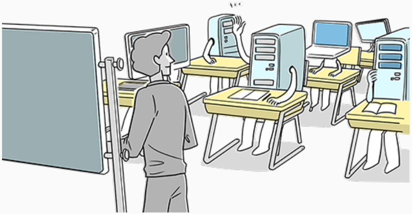
Modern computer vision – Deep learning



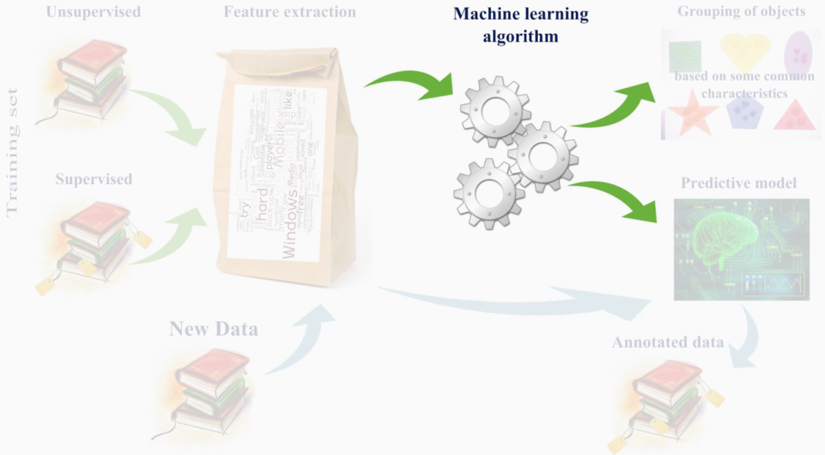
Deep learning is about learning the feature extraction, instead of designing it yourself.

Deep learning requires a lot of data and hacks to fight the curse of dimensionality (*i.e.*, reduce complexity and overfitting).

Quick overview of ML algorithms



What about algorithms?



(Source: Michael Walker)

Quick overview of ML algorithms

In fact, most of statistical tools are machine learning algorithms.

Dimensionality reduction / Manifold learning

- Principal Component Analysis (PCA) / Factor analysis
- Dictionary learning / Matrix factorization
- Kernel-PCA / Self organizing map / Auto-encoders

Linear regression / Variable selection

- Least square regression / Ridge regression / Least absolute deviations
- LASSO / Sparse regression / Matching pursuit / Compressive sensing

Classification and non-linear regression

- K-nearest neighbors
- Naive Bayes / Decision tree / Random forest
- Artificial neural networks / Support vector machines

Quiz: Supervised or unsupervised?

Quick overview of ML algorithms

Clustering

- K-Means / Mixture models
- Hidden Markov Model
- Non-negative matrix factorization

Recommendation

- Association rules
- Low-rank approximation
- Metric learning

Density estimation

- Maximum likelihood / a posteriori
- Parzen windows / Mean shift
- Expectation-Maximization

Simulation / Sampling / Generation

- Variational auto-encoders
- Deep Belief Network
- Generative adversarial network

Often based on tools from **optimization, sampling or operations research**:

- Gradient descent / Quasi-Newton / Proximal methods / Duality
- Simulated annealing / Genetic algorithms
- Gibbs sampling / Metropolis-hasting / MCMC

Questions?

Next class: Preliminaries to deep learning

Sources, images courtesy and acknowledgment

L. Condat
R. Fergus
P. Gallinari
N. Harris
A. Horodniceanu
J. Johnson
A. Karpathy
S. Lazebnik
F.-F. Li
A. Newson
S. Parameswaran
D. C. Pearson
S. Seitz
V. Spruyt
V. Tong Ta
R. Wendell
Wikipedia