

Principles of Data Science

Prof. Tias Guns
tias.guns@kuleuven.be

DTAI lab, KU Leuven 
Data Analytics lab, VUB 

General outline

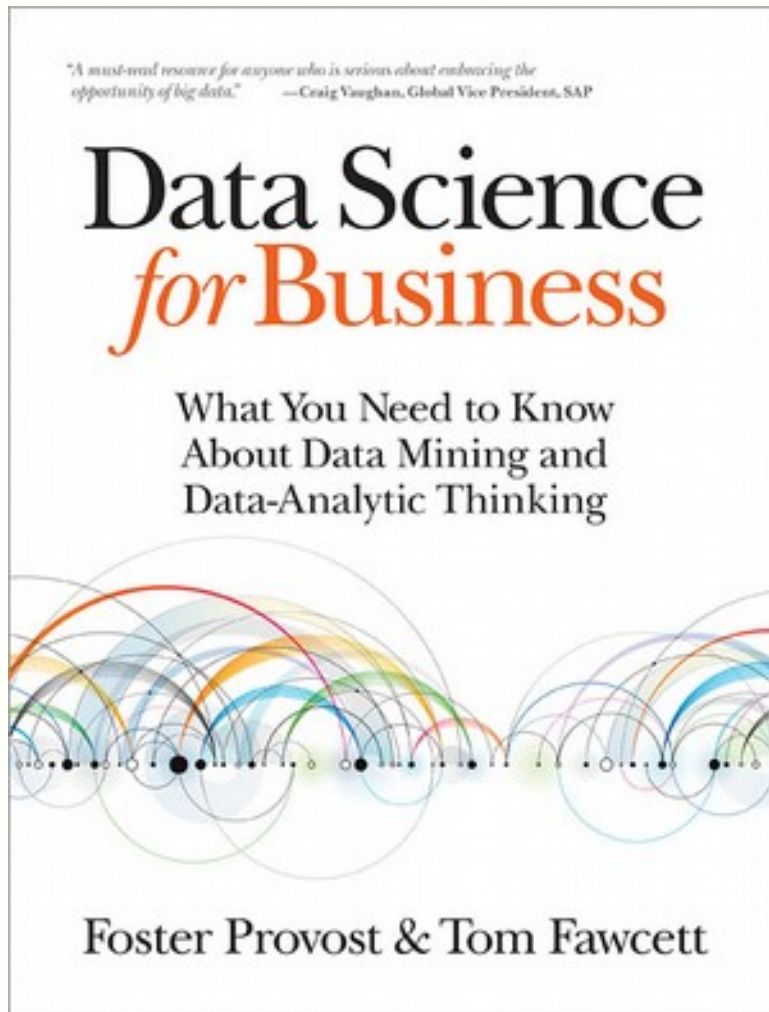
This session:

- ~60 min Principles of Data Science
- ~30 min Learning from user in CP
- ~30 min Practical (python notebook)

Afternoon session:

- ~20 min Learning from vision in CP
- ~40 min Learning from environment in CP
- ~30 min Practical (other python notebook)

Sources



Includes slides from
P. Adamopoulos (NYU) and
E. Ricci (Perugia)

and colleagues and
collaborators, including:

Hendrik Blockeel and
Luc De Raedt, KU Leuven

Paulo Frasconi, Uni Florence

Wouter Verbeke, VUB

What is data science?

“The previous new hype?”

Statistics...

Big data...

Machine Learning...

Data Analytics...

Data Mining...

Deep Learning...

Prescriptive analytics...

A.I...

Data Science...

What is Data Science?

No single definition

Components:

- Data-driven (the more the better: big data)
- Interdisciplinary (math, stat, CS, ...)
- Extract knowledge from observed data

Success stories 1/3

automatic image captioning



"man in black shirt is playing guitar."



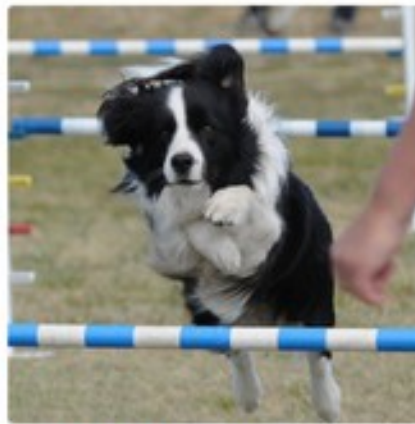
"construction worker in orange safety vest is working on road."



"two young girls are playing with lego toy."



"girl in pink dress is jumping in air."



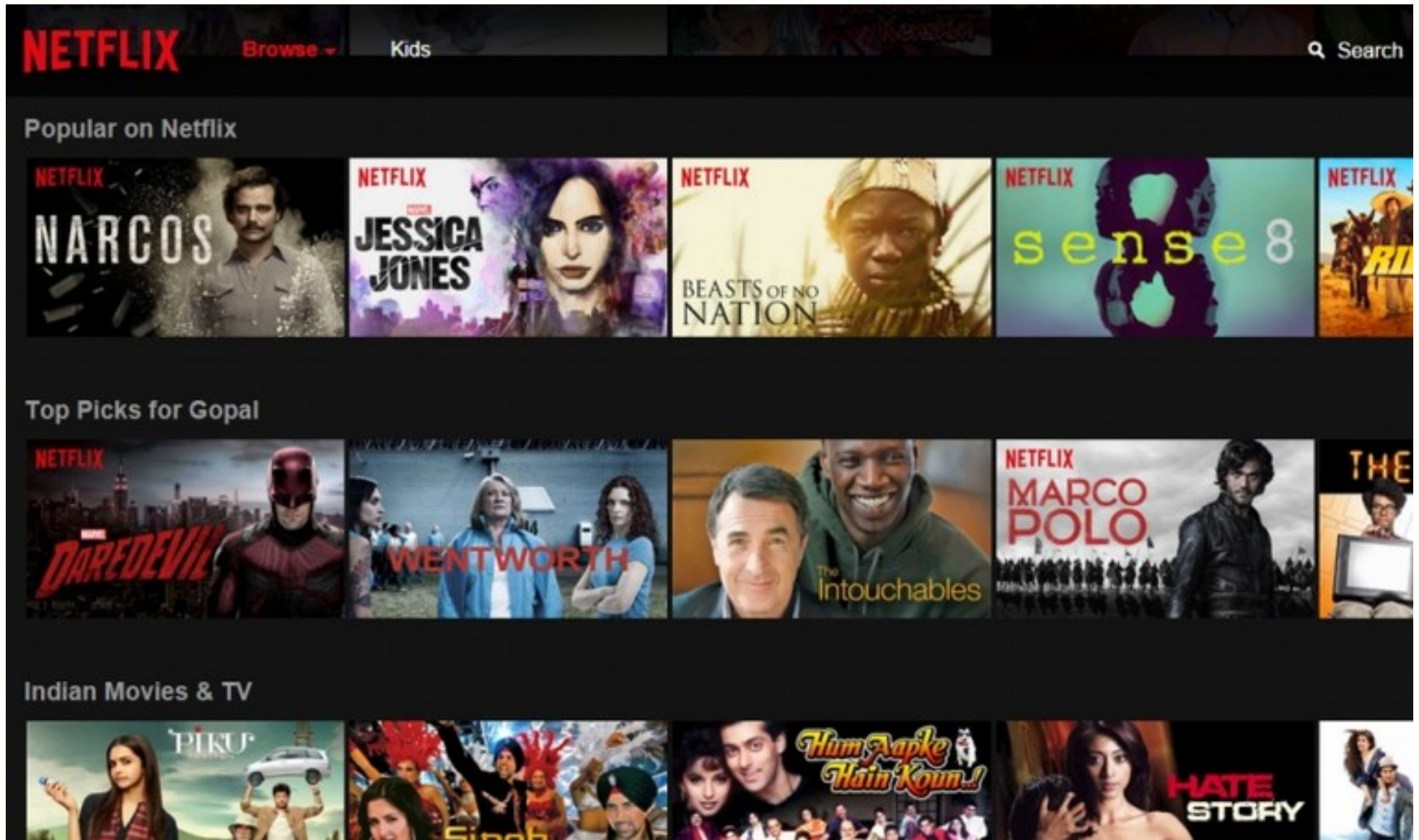
"black and white dog jumps over bar."



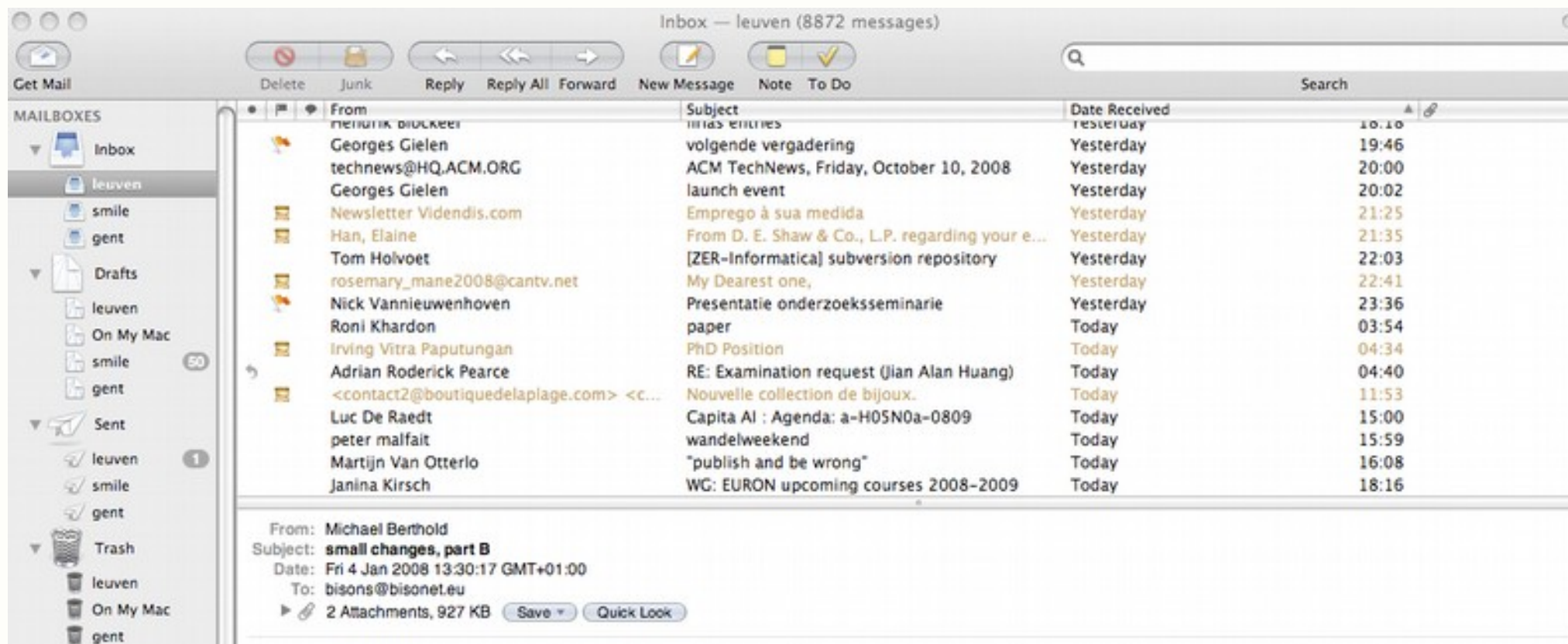
"young girl in pink shirt is swinging on swing."

Success stories 2/3

product recommendation



Success stories 3/3 spam detection



Common Data Mining Tasks

- Classification and class probability estimation
 - How likely is this consumer to respond to our campaign?
- Regression
 - How much will she use the service?
- Similarity Matching
 - Can we find consumers similar to my best customers?
- Clustering
 - Do my customers form natural groups?

Common Data Mining Tasks

Task	Supervised Methods	Unsupervised Methods
*Classification	✓	
*Regression	✓	
Causal Modeling	✓	
Similarity Matching	✓	✓
Link Prediction	✓	✓
Data Reduction	✓	✓
Clustering		✓
Co-occurrence Grouping		✓
Profiling		✓

Supervised = *labelled* data, target attribute (e.g. SPAM or not)

Unsupervised = *no labels* (e.g. customer records for profiling)

Terminology

Name	Balance	Age	Employed	Write-off
Mike	\$200,000	42	no	yes
Mary	\$35,000	33	yes	no
Claudio	\$115,000	40	no	no
Robert	\$29,000	23	yes	yes
Dora	\$72,000	31	no	no

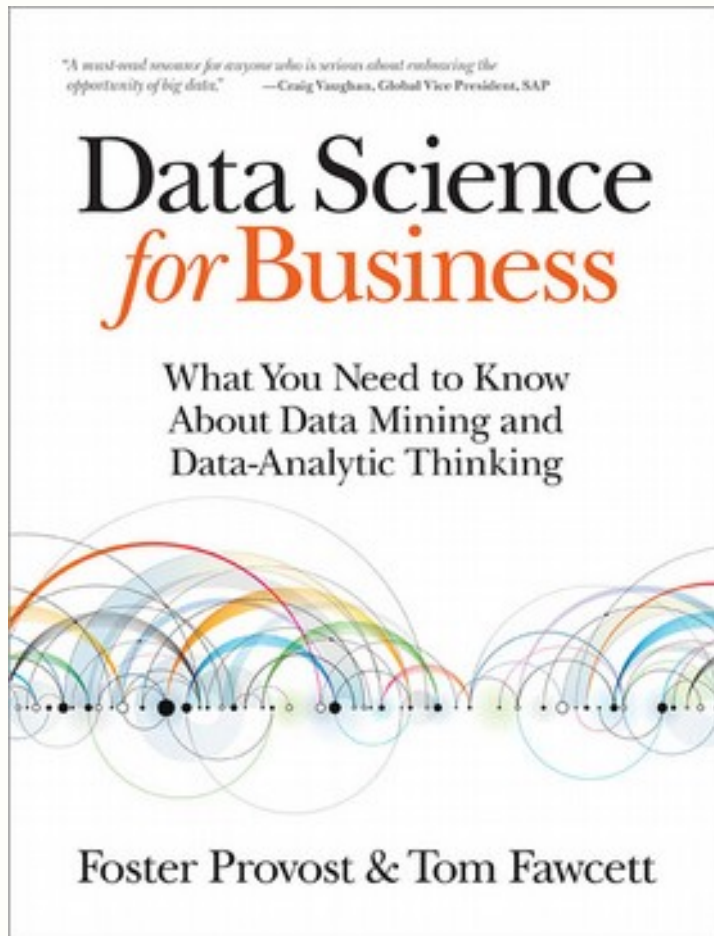
This is one row (example).

Feature vector is: **<Claudio,115000,40,no>**

Class label (value of Target attribute) is **no**

Books

Most books: algorithmic or statistical focus



Focus on general principles

“In ten years’ time, technologies will likely have changed such that today’s choices seem quaint.”

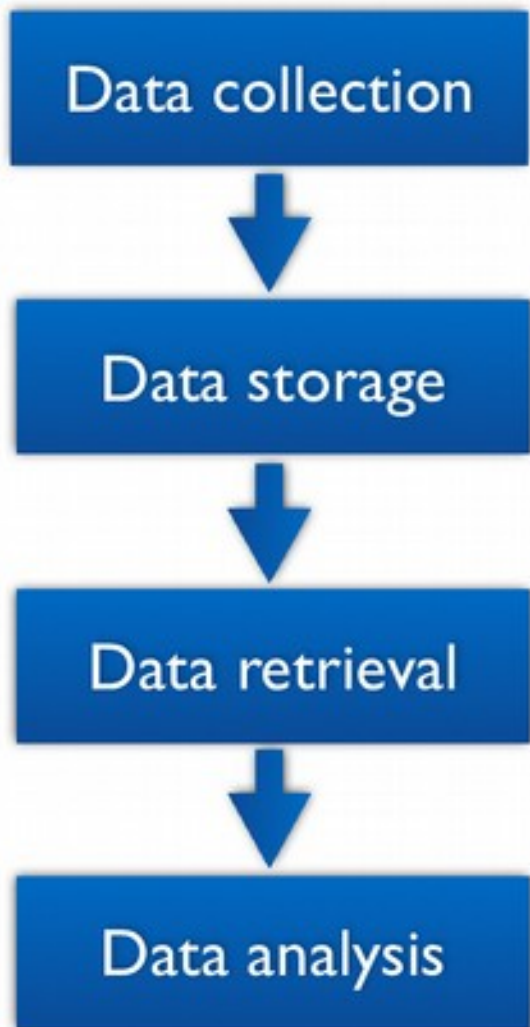
...

“general principles same for 20 years”

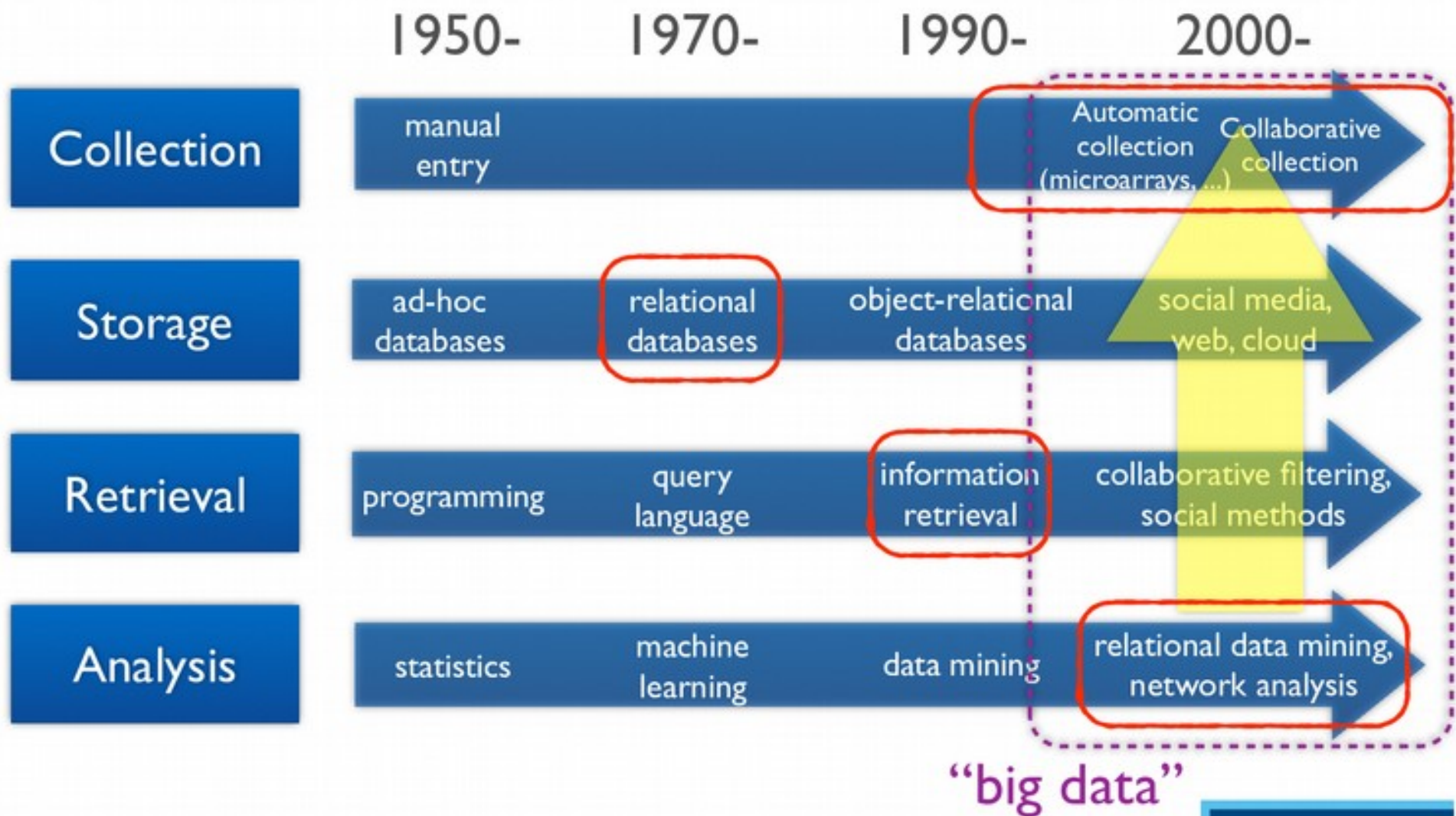
Principle 1:

Data Science is a process

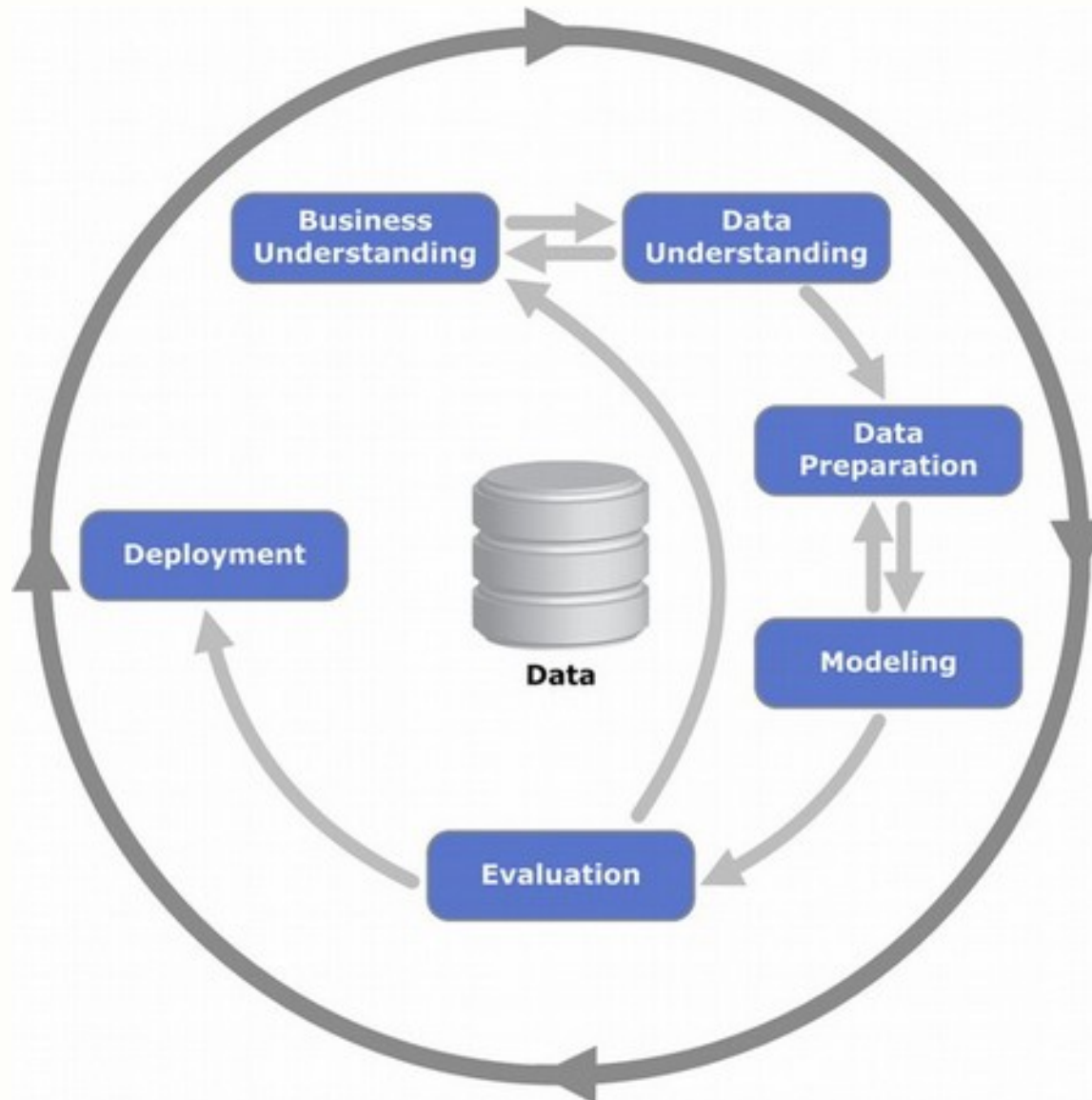
From collection to utilization



Chronology

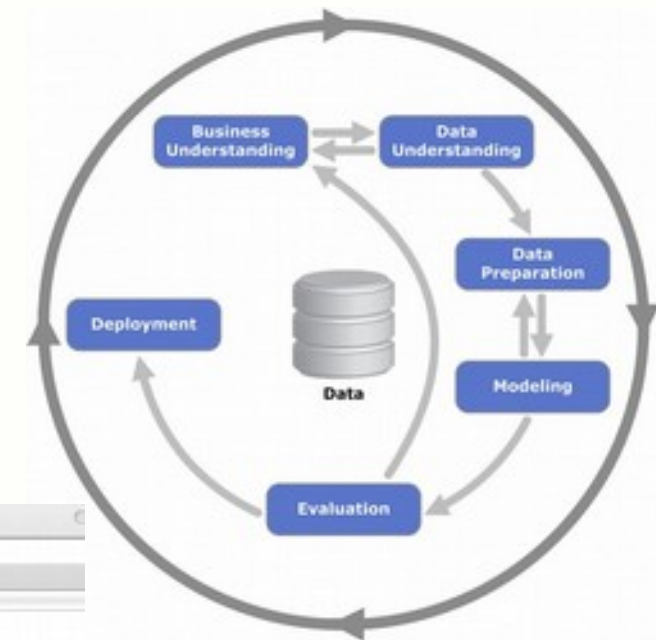
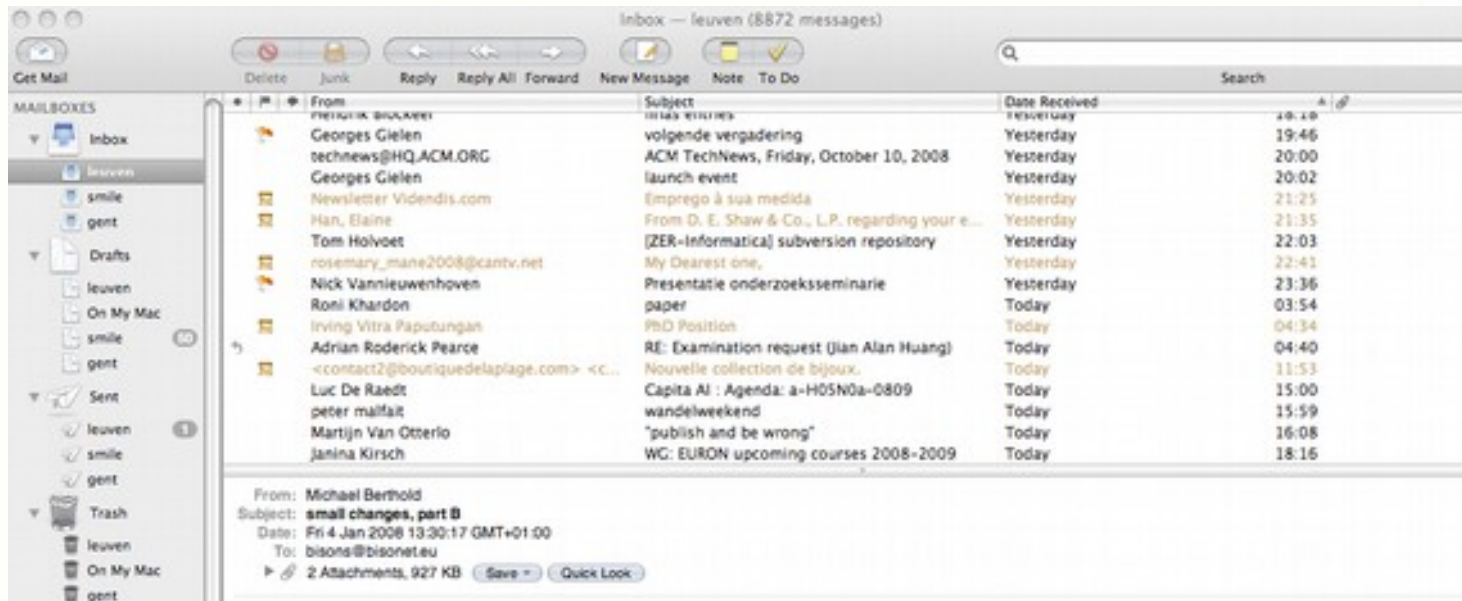


CRISP-DM process



An example

Business understanding



SPAM email reduces productivity,
automatically remove it

An example

Data understanding



Given a text message, predict whether it is spam or not

- text categorization, useful in general
- we want a function from message to $\{0,1\}$
- is called binary classification problem

An example

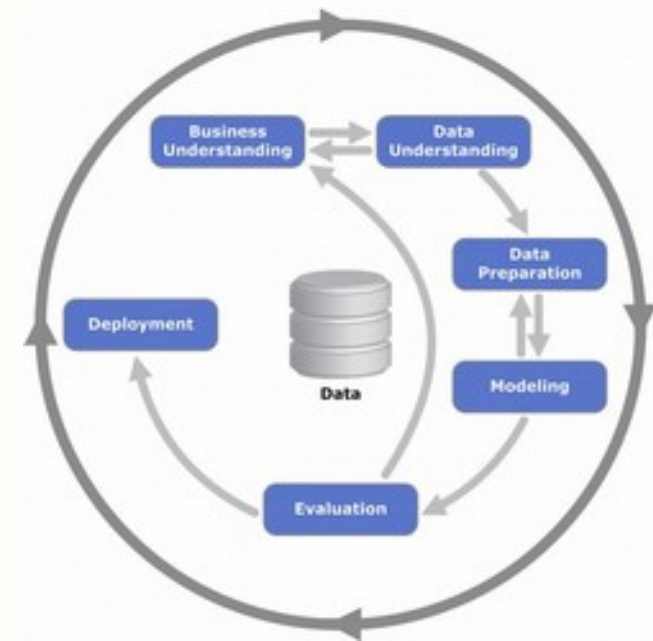
Data preparation: raw text

Modeling:

We could write a rule-based system, such as

```
if Title.contains("YOU HAVE WON!!!") then  
    return Spam
```

Does it work well? → evaluate



An example

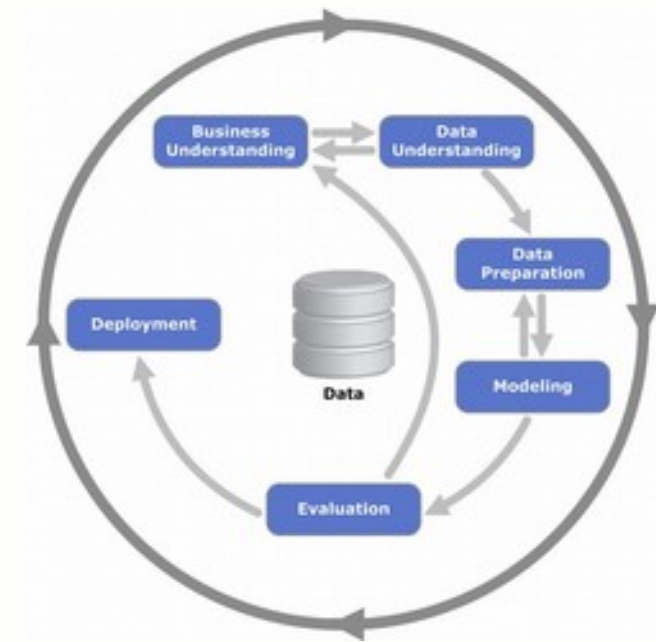
Evaluation



		Truth		
		Spam	Legitimate	
Predicted as	Spam	150	30	False positives
	Legitimate	200	720	False negatives



An example

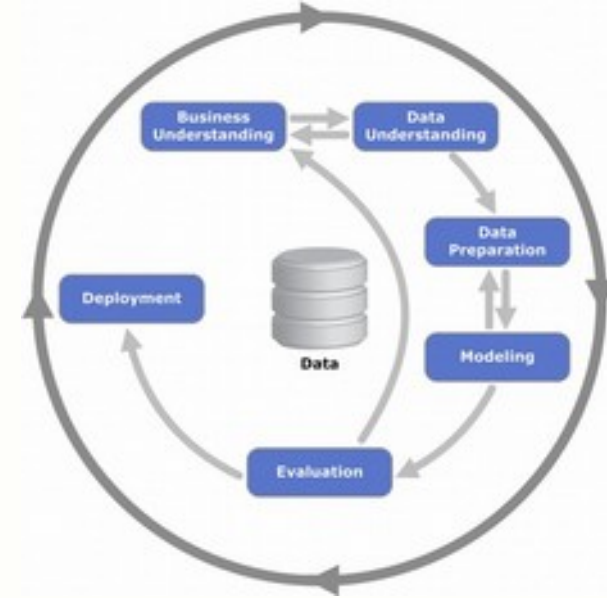


Evaluation to Business understanding:

- How do we find good rules?
Knowledge elicitation or formalization may be difficult
- How do we define good? Will depend on user?

We need a system that can adapt: self-learning

An example, classifier-based



- Data understanding: collect messages, in general and from the user, that are spam (negative) and legitimate (positive)
- Data preparation: bag-of-words representation
- Modeling: train a classifier (e.g. naïve bayes)
- Evaluate: on unseen emails
- Deploy: predict for new emails, retrain when user disagrees

Principle 2a:

Machine Learning is optimisation,

it optimises loss functions

A formal task description

Function approximation!!!

•Given:

- a space of possible instances X
- an unknown target function $f: X \rightarrow Y$
- a hypothesis space L containing functions $X \rightarrow Y$
- **a set of examples $E = \{ (x, f(x)) \mid x \in X \}$**
- a loss function $loss(h,E) \rightarrow \mathbb{R}$

•Find: $h \in L$ that minimizes $loss(h,E)$

↑
model

↑
“supervised”

Matches many (not all) tasks

Attributes				Target attribute
Name	Balance	Age	Employed	Write-off
Mike	\$200,000	42	no	yes
Mary	\$35,000	33	yes	no
Claudio	\$115,000	40	no	no
Robert	\$29,000	23	yes	yes

Linear Regression

Notations:

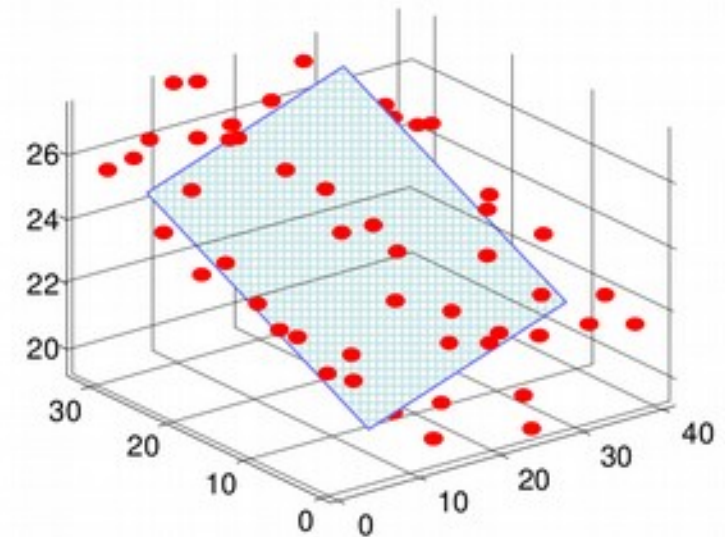
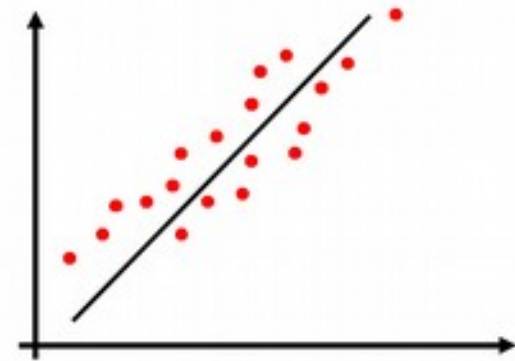
- Datapoints $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$ $\mathbf{x}_i \in R^d$

- Labels $\mathbf{y} = \{y_1, y_2, \dots, y_n\}$ $\mathbf{y} \in R$

- Linear decision function $f(\cdot): R^d \rightarrow R$

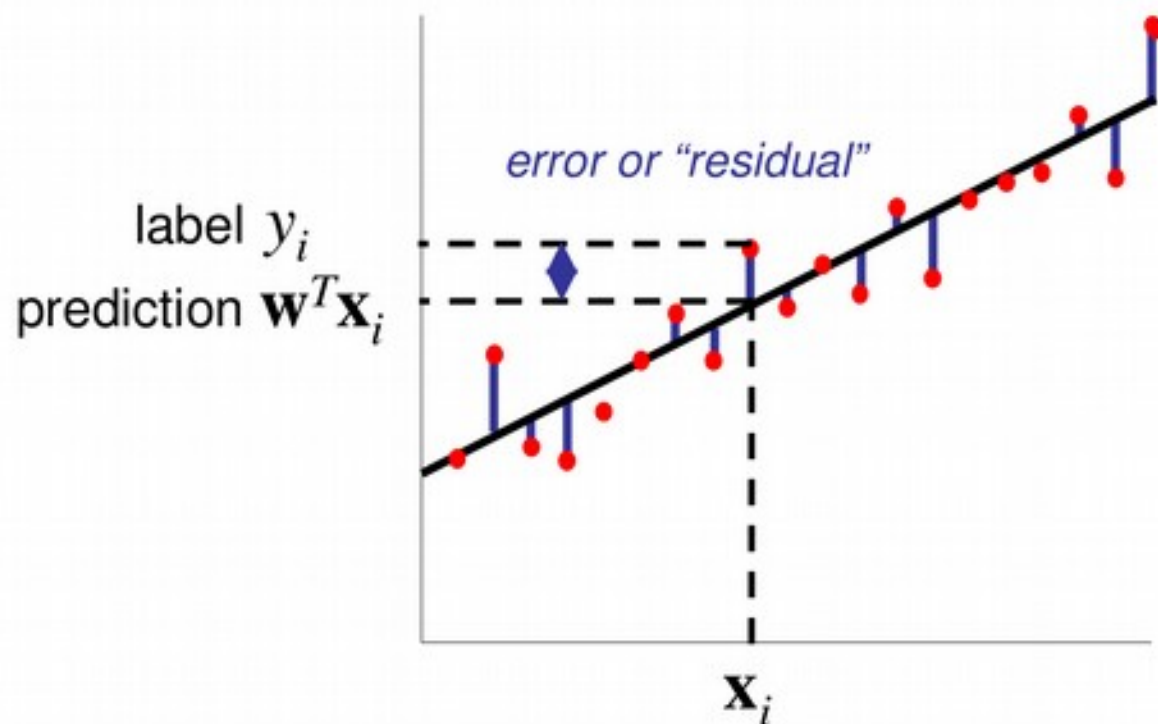
$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x}$$

- Parameter vector \mathbf{w}



Linear Regression

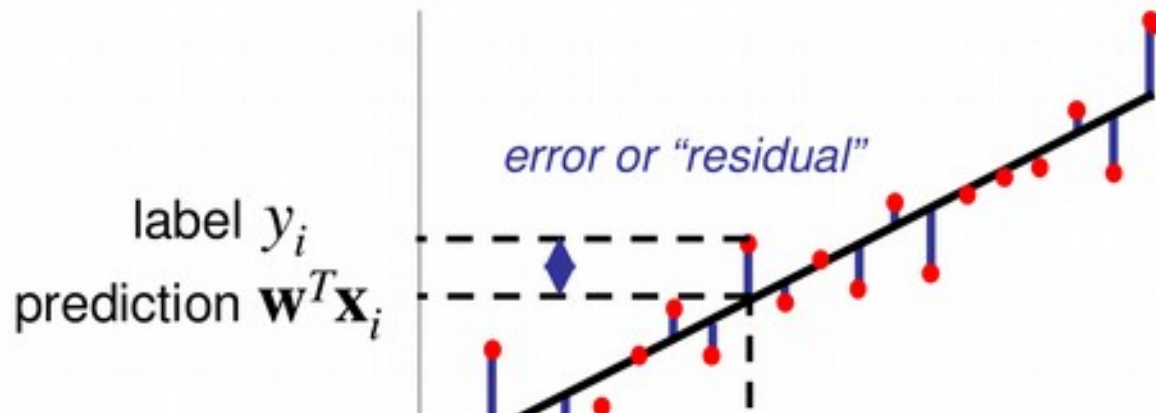
- Goal: find a linear function $\mathbf{X}\mathbf{w}$ that approximates the labels \mathbf{y} .
- For a new test point \mathbf{x} the label y can be estimated as $\mathbf{w}^T\mathbf{x}$.



Sum Squared Error
$$E = \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 = \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Linear Regression

- Goal: find a linear function $\mathbf{X}\mathbf{w}$ that approximates the labels \mathbf{y} .
- For a new test point \mathbf{x} the label y can be estimated as $\mathbf{w}^T\mathbf{x}$.

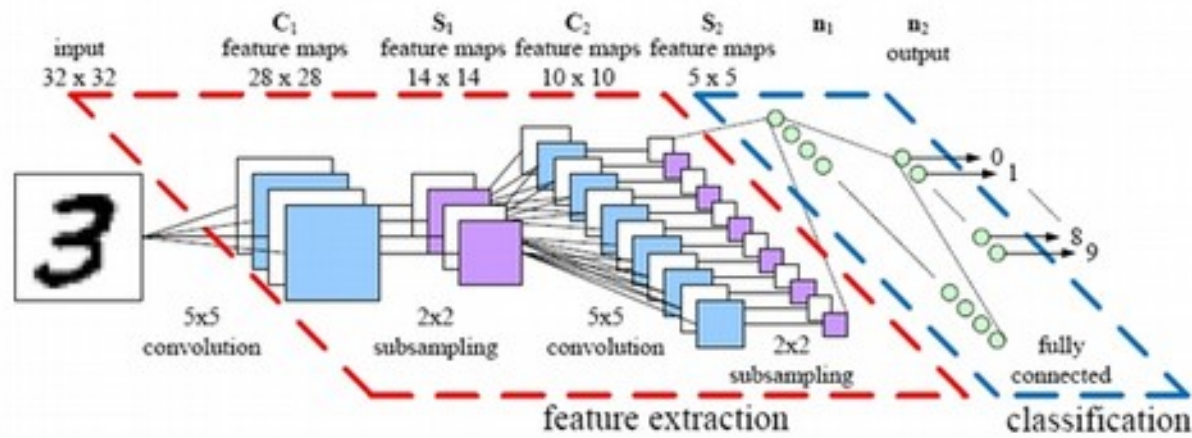


Loss function is Sum of Squared errors (L2 norm)
Is convex \rightarrow optimise it (least squares regr.)

$$\text{Sum Squared Error } E = \sum_{i=1}^n (y_i - \mathbf{w}^T \mathbf{x}_i)^2 = \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$$

Deep learning

handwritten number recognition:



learning: (stochastic) gradient descent

Deep learning

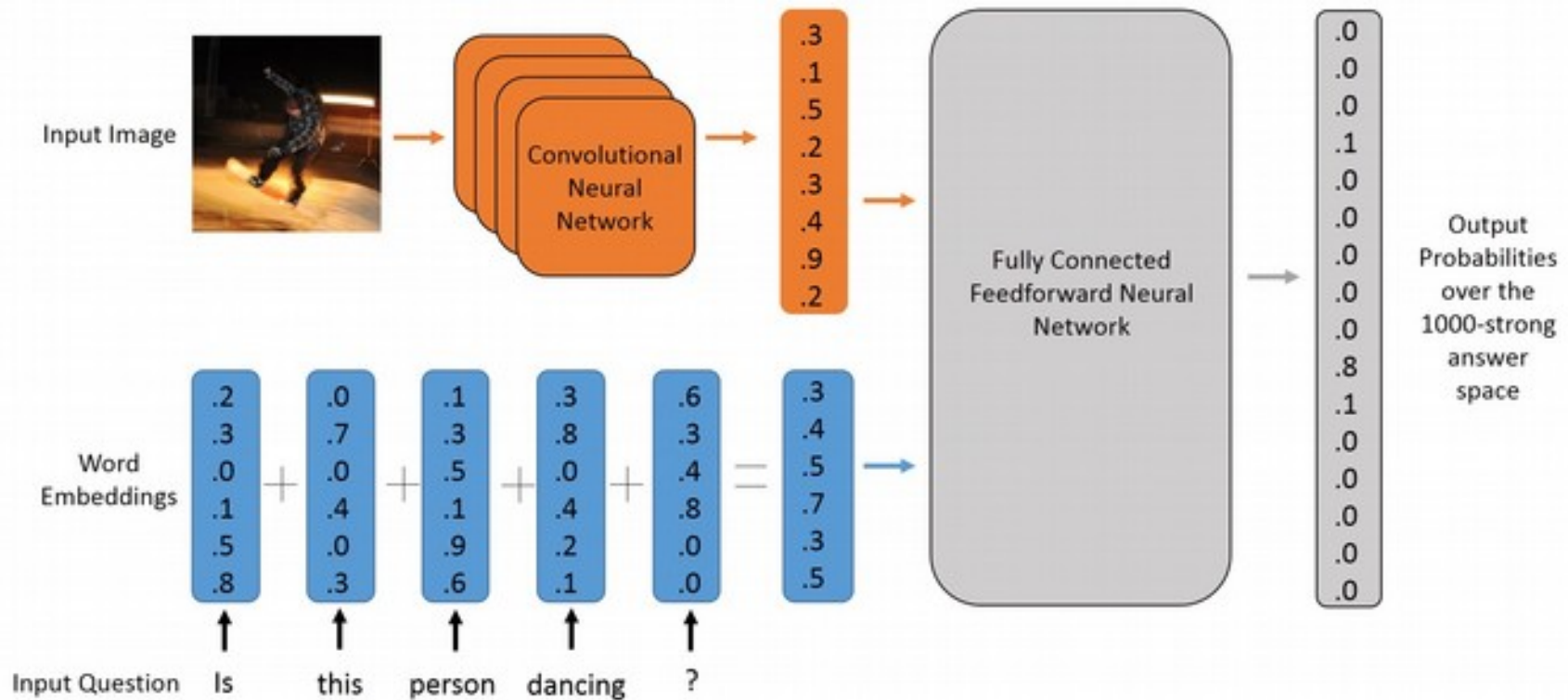
Optimisation through stochastic gradient descent

Algorithm 1: Stochastic gradient descent

Input : training data $\mathcal{D} = \{X, y\}_{i=1}^n$, learning rate γ

- 1 initialize θ (neural network weights)
- 2 **for** *epochs* **do**
- 3 **for** *batches* **do**
- 4 sample batch $(X, y) \sim \mathcal{D}$
- 5 $\hat{y} \leftarrow g(z, \theta)$ (forward pass: compute predictions)
- 6 Compute loss $L(y, \hat{y})$ and gradient $\frac{\partial L}{\partial \theta}$
- 7 Update $\theta = \theta - \gamma \frac{\partial L}{\partial \theta}$ through backpropagation (backward pass)
- 8 **end**
- 9 **end**

Deep learning



learning: (stochastic) gradient descent

<https://playground.tensorflow.org/>

Deep learning: importance layers, number neurons?

The screenshot displays the TensorFlow Playground interface with several key elements highlighted by red circles:

- Problem type:** Set to "Classification".
- Activation:** Set to "Tanh".
- Number of hidden layers:** Set to "2 HIDDEN LAYERS".
- Dataset:** The "MNIST" dataset is selected.

The interface also shows the following details:

- Epochs:** 000,000
- Learning rate:** 0.03
- Regularization:** None
- Regularization rate:** 0
- DATA:** Ratio of training to test data: 50%, Noise: 0, Batch size: 10.
- FEATURES:** Includes X_1 , X_2 , X_1^2 , X_2^2 , $X_1 X_2$, $\sin(X_1)$, and $\sin(X_2)$.
- OUTPUT:** Test loss 0.525, Training loss 0.517.
- Diagram:** A neural network diagram with 4 neurons in the first hidden layer and 2 neurons in the second hidden layer. A color scale at the bottom indicates values from -1 to 1.

Principle 2b:

If you look too hard at a dataset,
you'll find things that don't generalize
to unseen data

Principle 2b:

If you look too hard at a dataset,
you'll find things that don't generalize
to unseen data

Overfitting

A formal task description

Function approximation!!!

•Given:

- a space of possible instances X
- an unknown target function $f: X \rightarrow Y$
- a hypothesis space L containing functions $X \rightarrow Y$
- **a set of examples $E = \{ (x, f(x)) \mid x \in X \}$**
- a loss function $loss(h, E) \rightarrow \mathbb{R}$

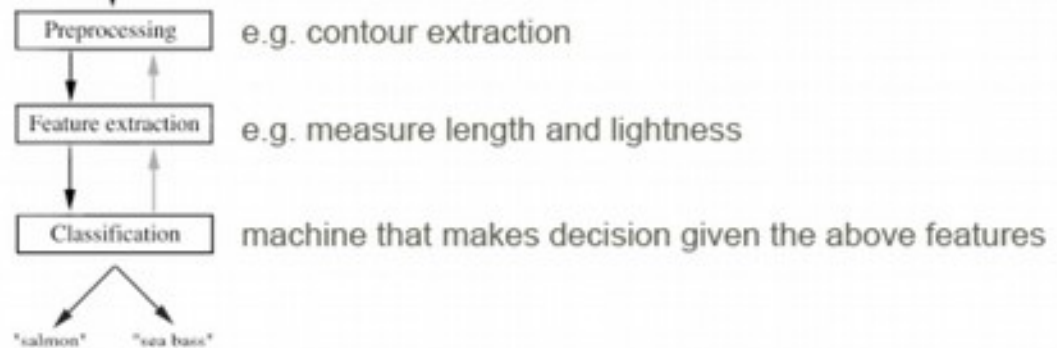
•Find: $h \in L$ that minimizes $loss(h, E)$

 model

 “supervised”

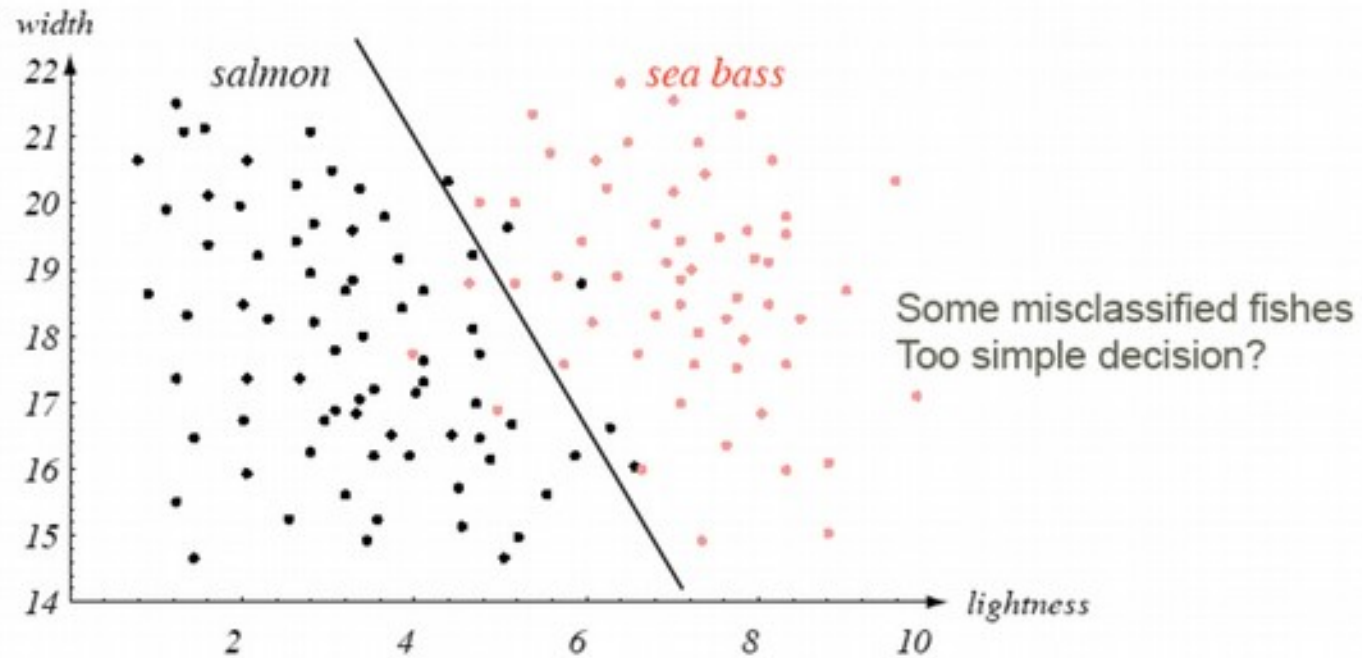
Matches many (not all) tasks

A toy example

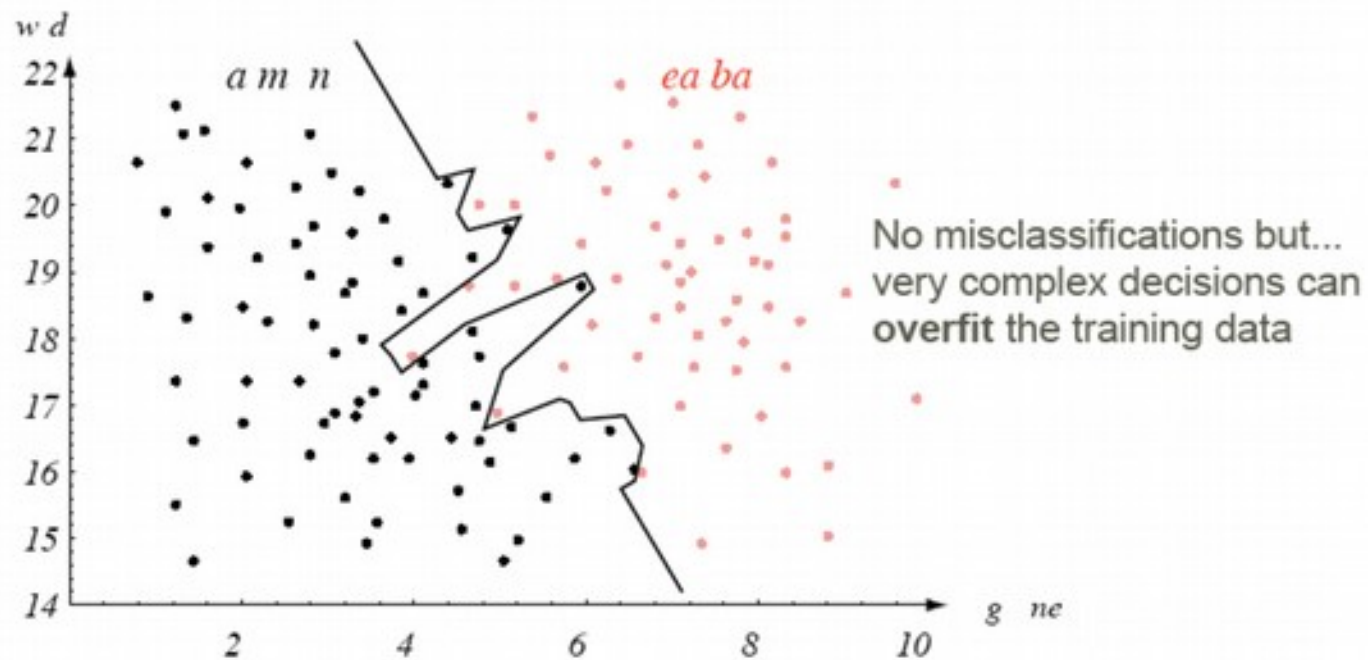


Example from: Pattern Classification (2nd ed) by R. O. Duda, P. E. Hart and D. G. Stork, John Wiley & Sons, 2000

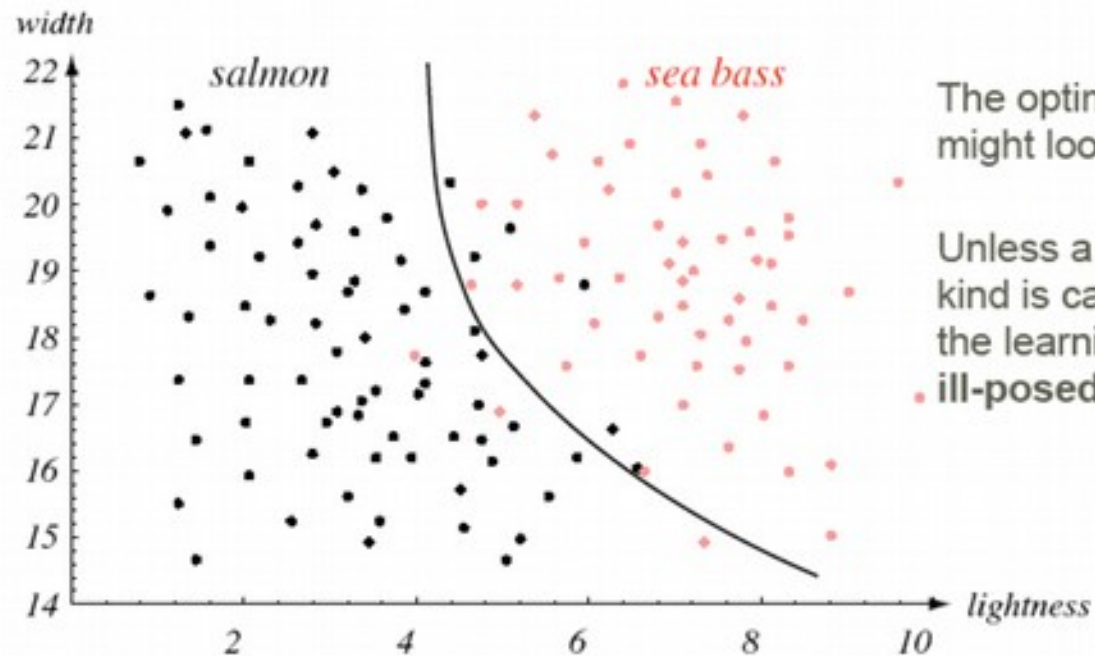
Linear Separation



Overfitting



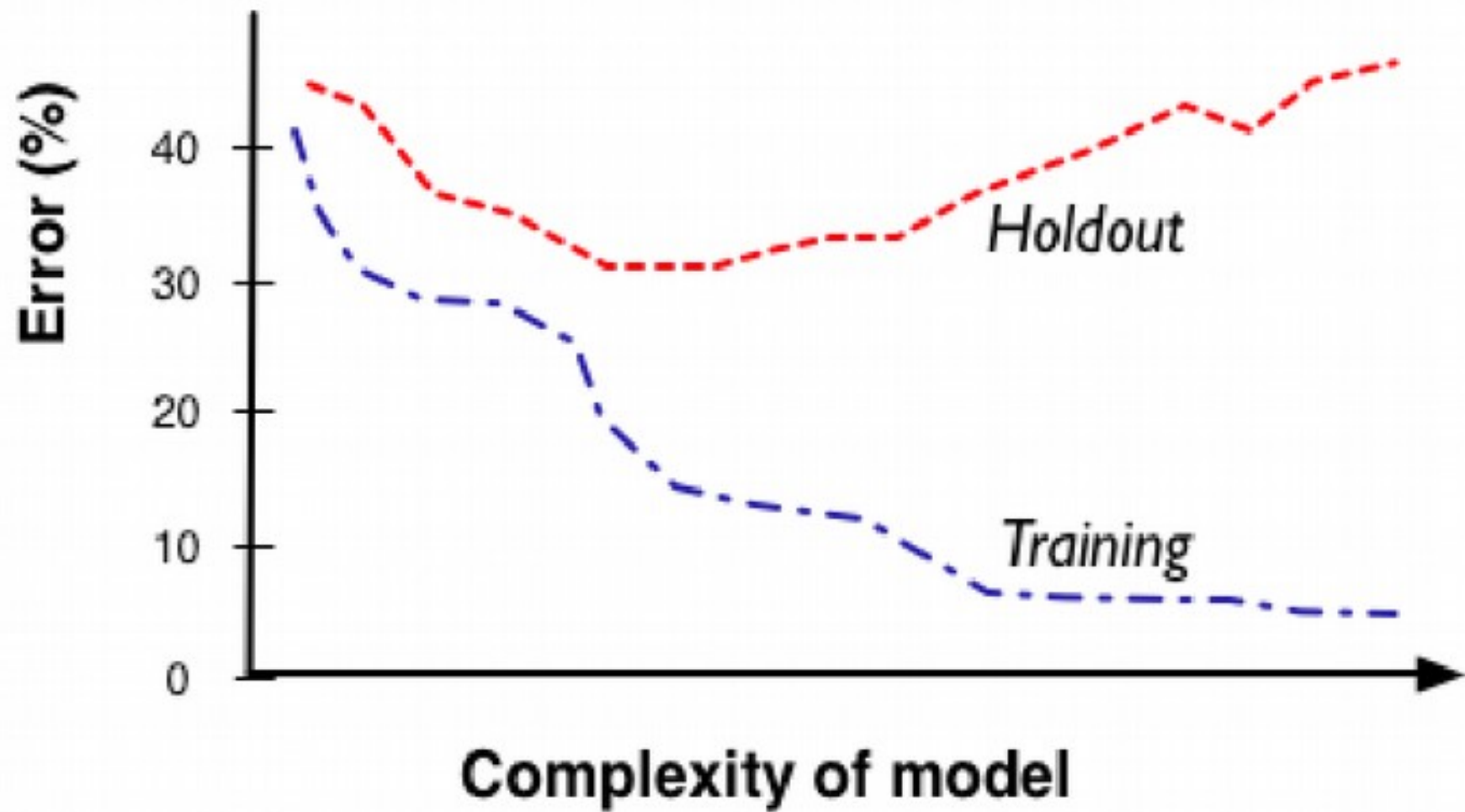
Reasonable Solution



The optimal tradeoff might look like this

Unless a tradeoff of this kind is carefully defined the learning problem is **ill-posed!**

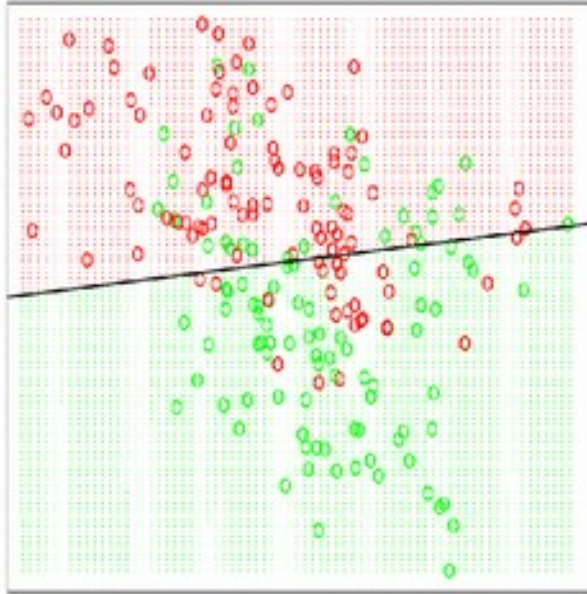
Fitting Graph



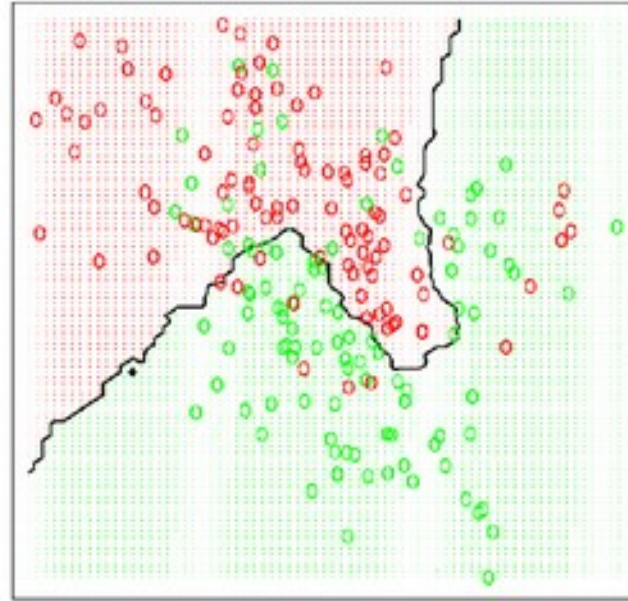
Over-fitting in tree induction



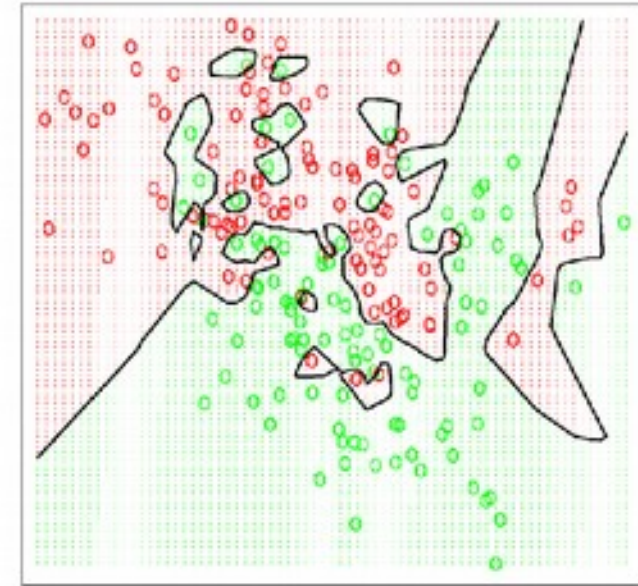
Need for holdout evaluation



Under-fitting



Good



Over-fitting

- In sample evaluation is in favor or “memorizing”
- On the *training data* the right model would be best
- But on *new data* it would be bad

<https://playground.tensorflow.org/>

Overfitting: try different ways to make it happen

The screenshot displays the TensorFlow Playground interface for a neural network. The top navigation bar shows the following settings:

- Epoch: 000,004
- Learning rate: 0.03
- Activation: Tanh
- Regularization: None
- Regularization rate: 0
- Problem type: Classification

The main interface is divided into several sections:

- DATA:** A dropdown menu is circled in red, showing various dataset options. Below it, the "Ratio of training to test data" is set to 50%, and the "Noise" level is set to 50, both of which are also circled in red. A "REGENERATE" button is located at the bottom of this section.
- FEATURES:** A list of input features is shown, including X_1 , X_2 , X_1^2 , X_2^2 , X_1X_2 , $\sin(X_1)$, and $\sin(X_2)$.
- NEURAL NETWORK ARCHITECTURE:** The network consists of an input layer with 5 neurons, three hidden layers each with 5 neurons, and an output layer with 5 neurons. The text "3 HIDDEN LAYERS" is circled in red. A color scale at the bottom indicates that colors represent data, neuron, and weight values, ranging from -1 (blue) to 1 (orange).
- OUTPUT:** A scatter plot shows the training data points (blue and orange) and the model's output. The training loss is 0.492 and the test loss is 0.503. A line graph above the scatter plot shows the loss curves. The output plot and the loss graph are circled in red.

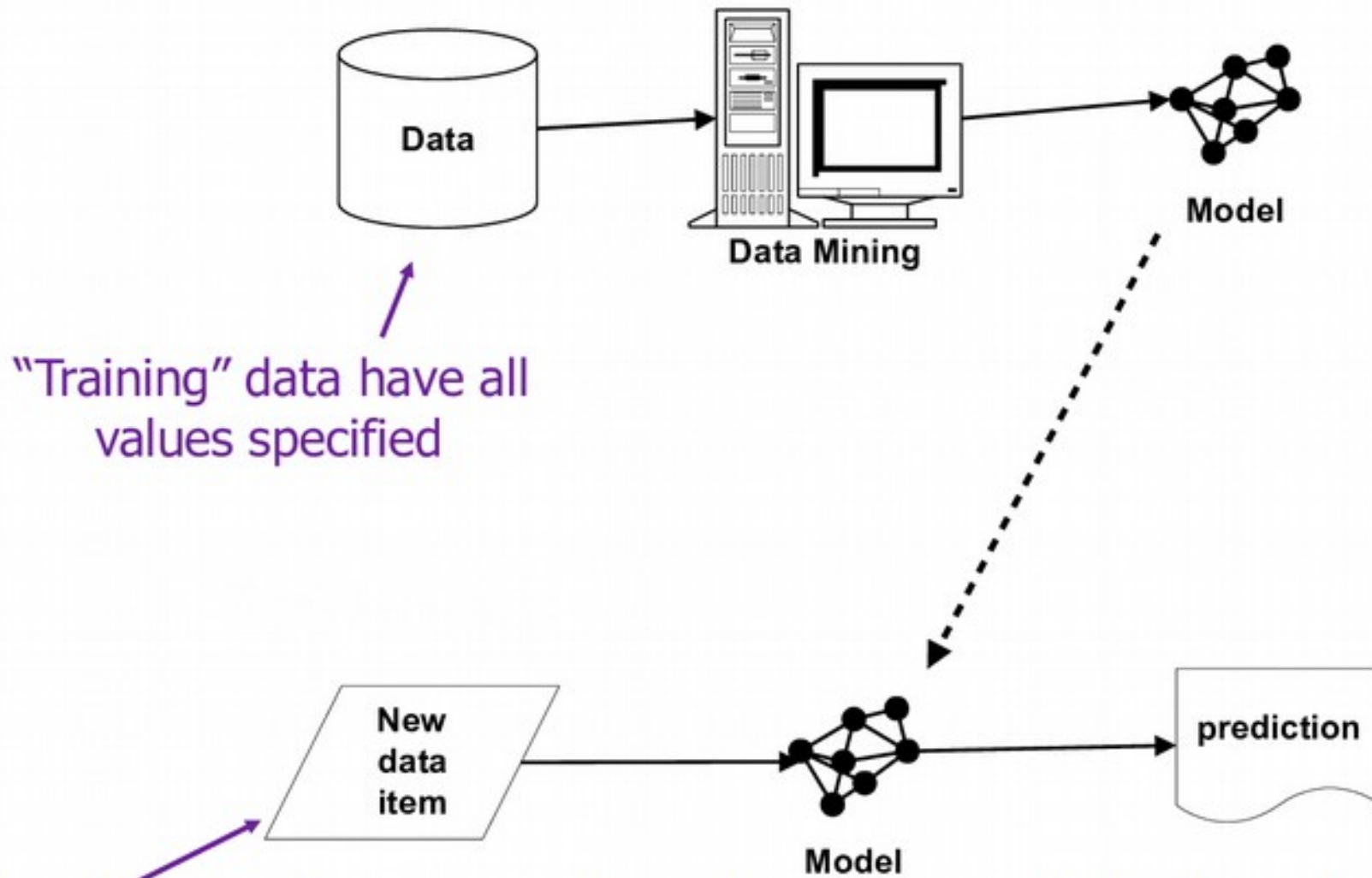
Additional text in the interface includes: "This is the output from one neuron. Hover to see it larger." and "The outputs are mixed with varying weights, shown by the thickness of the lines."

Principle 3:

Data science needs to be evaluated
in the context of operation

Data Mining versus Use of the Model

"Supervised" modeling:



"Training" data have all values specified

New data item has some value unknown (e.g., will she leave?)

Pitfalls in DM

- Training data is not consistent with actual use
- Bad sample
- Bad features

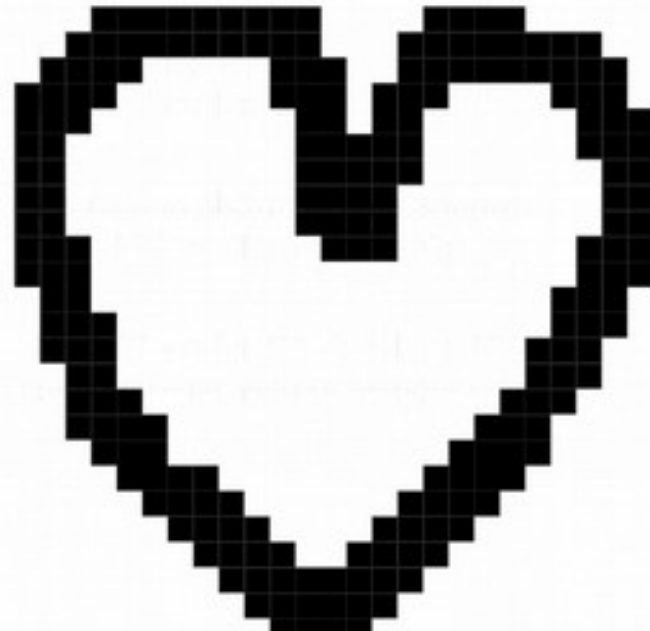
Ex: “survivorship issues”

Lending agency wants to use ML to screen applications and accept/reject them

- data of accepted loans + outcome
- BAD: use this to learn an outcome predictive model

Pitfalls in DM

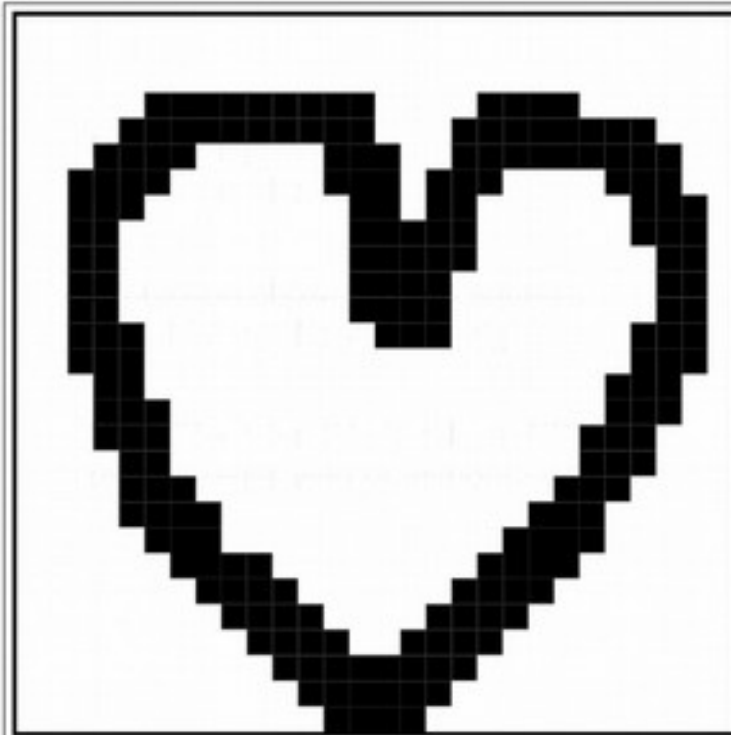
What number is this?



http://www.ccom.ucsd.edu/~cdeotte/ programs/MNIST.html



MNIST Digit Recognizer



Select tool:

Draw

Erase

0

Probability = 0.97

Classify

Clear



Your
History

Download digits as CSV without labels

Download digits as CSV with true labels

Draw a digit between 0 and 9 above and then click classify. A neural network will predict your digit in the blue square above. Your image is 784 pixels (= 28 rows by 28 columns with black=1 and white=0). Those 784 features get fed into a 3 layer neural network; Input:784 - AvgPool:196 - Dense:100 - Softmax:10. The net has 20,600 learned weights hardcoded into this JavaScript webpage. It achieves 98.5% accuracy on the famous MNIST 10k test set and was coded and trained in C. The net is explained [here](#). The best nets are convolutional neural networks and they can achieve 99.8% accuracy. An example coded in Python with Keras and TensorFlow is [here](#).

Additionally this page allows you to download your hand drawn images. Your images get added to your history showing above to the right. Click 'download' to receive a CSV of digits with or without labels. You can import that CSV into your neural network software for training or testing. The format of the CSV is the same as Kaggle's. Each row is a digit with 784 pixels representing a 28x28 image (rows first). If you download

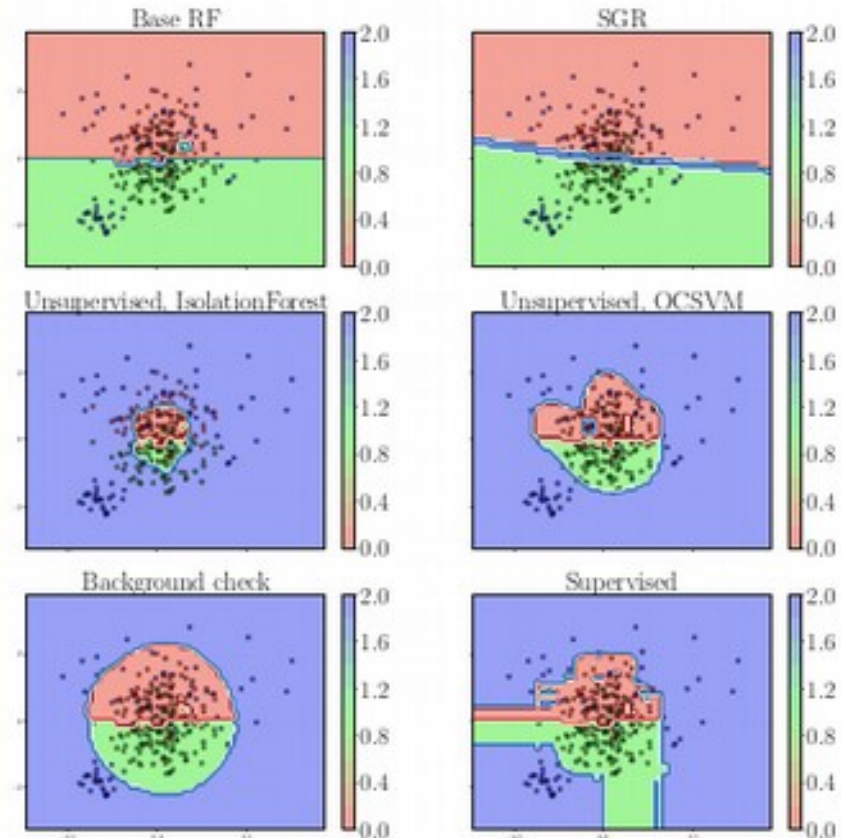
Pitfalls in DM



Can we do something about this?

Highly relevant topic; related to:

- out-of-distribution detection
- classification with a reject option

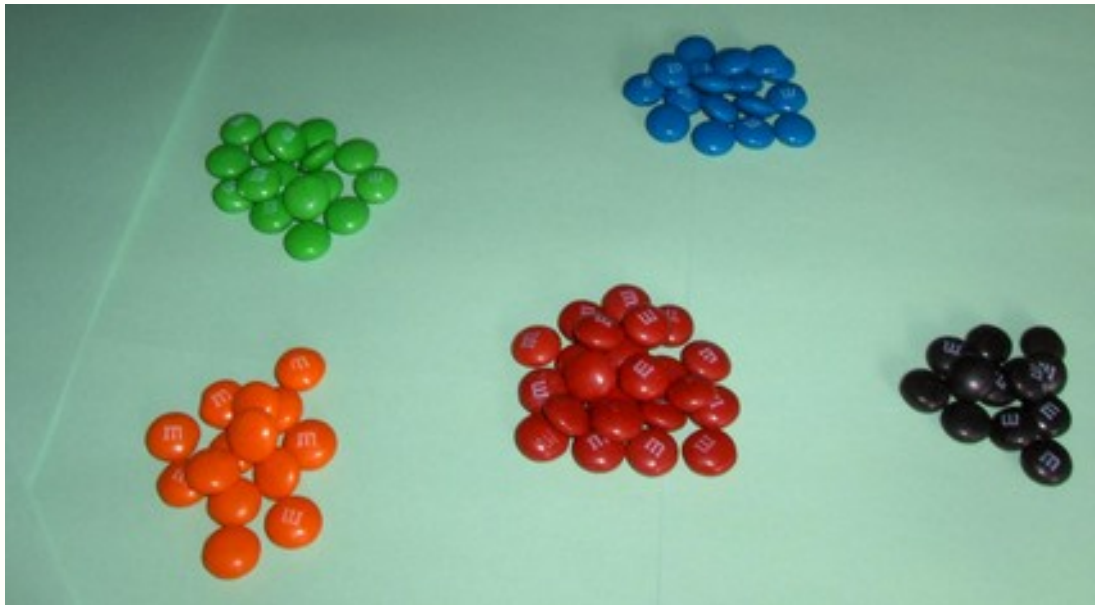


Principle 4:

Entities that are similar on some attributes
often are similar on unseen attributes

Similarity

- Ex: clustering



- Also solved with optimisation, e.g. min. distances to cluster center

Similarity

Key concept: distance between objects

Euclidean, manhattan, edit distances (strings),
dynamic time warping (temporal sequences), ...

Ex. group hand-written letters together

1) based on raw pixels (but: shift, scale)

2) based on learned representation (auto-encoding)

<https://cs.stanford.edu/people/karpaty/convnetjs/demo/autoencoder.html>

Auto-encoding + clustering of representation

of neuron 1 and firing of neuron 2. These two values are enough for the decoder network that follows to reproduce all 784 original numbers. As an example, suppose the 8 activates neurons 1 and 2 to 0.5 and 0.9, we would plot that digit 8 at position (0.5, 0.9) in the visualization.



cycle through visualized neurons at selected layer (if more than 2)

drawing neurons 0 and 1 of layer with index 5 (fc)



Principle 5:

To draw causal conclusions,

one must pay very close attention to the presence of (possibly unseen) confounding factors

Causality?

Machine models exploit correlation, NOT causality

Very tempting to inspect model and see
“what causes things to be true/false”

Causality?

Machine models exploit correlation, NOT causality

Very tempting to inspect model and see
“what causes things to be true/false”

E.g. coefficients of linear regression

$$Y = 20 * X_1 - 12 * X_2 + 300 * X_3 + 99 * X_4 - 299 * X_5$$

Which feature has most impact?

Principles

- 1.Data Science is a process
- 2.ML is optimisation of loss functions
- 3.ML must generalize to unseen data
- 4.Evaluate data science in its operational context
- 5.Similar entities can have similar unseen attribs
- 6.Correlation, not causation

Popular practical tools

- Exploratory analysis of high dim. tabular data:
tableau (web only, not open source)
- Classification and regression on tabular data:
scikitlearn
- Non-linear regression on large tabular data:
xdgboost
- Deep learning on sensory data (images, audio, ...):
pytorch

Questions?

Slides available: <http://homepages.vub.ac.be/~tiasguns/> (soon)

More playgrounds?

<https://cs.stanford.edu/people/karpathy/convnetjs/>