Machine Learning

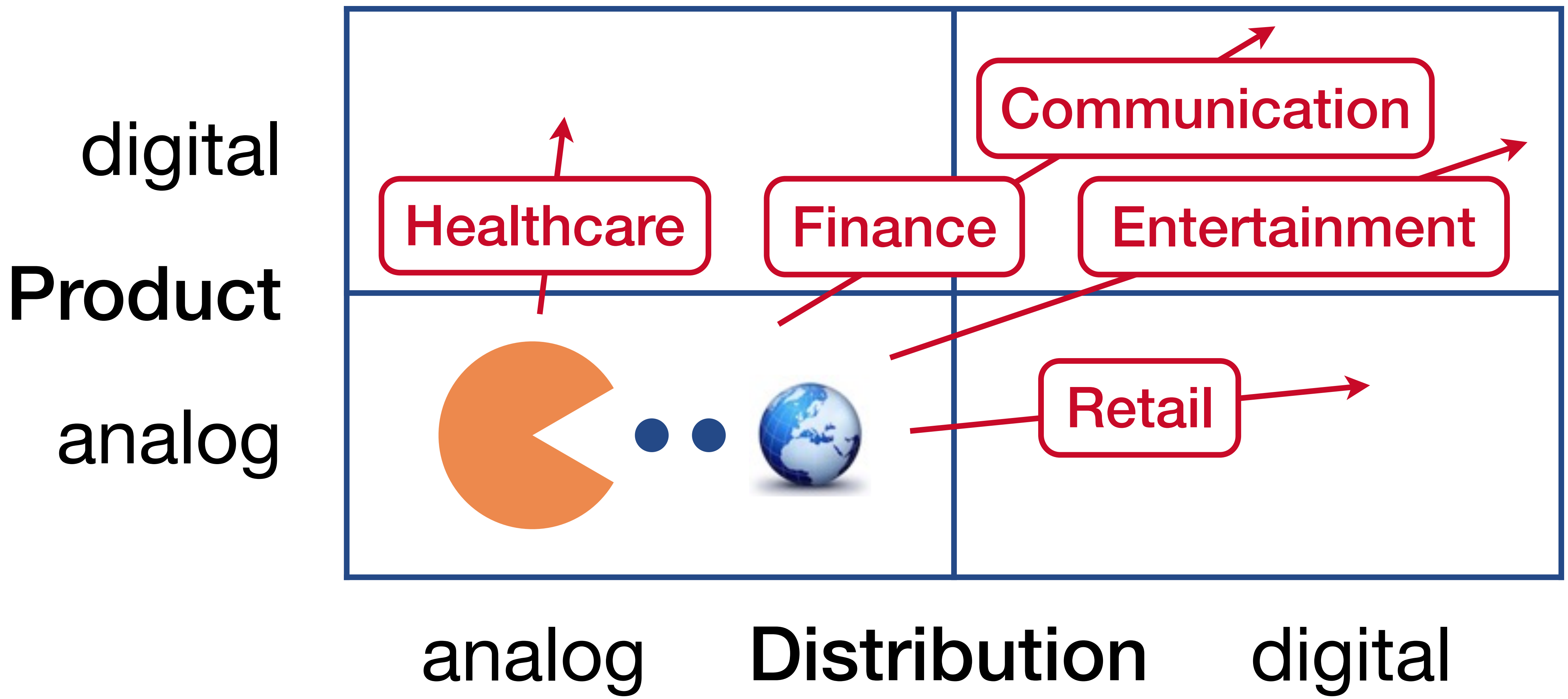
A Short Introduction

Kaspar Etter, MSc ETH in CS
kaspar.etter@gbs-schweiz.org

Basel, Switzerland
15 April 2015

Software is eating the world

… because it is more productive!



The bigger wave is just coming…

Machine learning will eat the world like nothing before it.

Machine Intelligence LANDSCAPE

CORE TECHNOLOGIES

ARTIFICIAL INTELLIGENCE
IBM WATSON, MetaMind, Numenta, ai-one, Cycorp, Microsoft Research, nara, Reactor, SI Scaled Inference

DEEP LEARNING
vicarious, Vision Factory, facebook, Google, Baidu IDL, SKYMIND, ersatz, SignalSense

MACHINE LEARNING
rapidminer, context relevant, 0xdata H2O, DATARPM, Liftigniter, SparkBeyond, Azure ML, yhat, Wise.io, Sense, GraphLab, Alpine, Nutonian

NLP PLATFORMS
cortical.io, idibon, LUMINOSO, wit.ai, Maluuba

PREDICTIVE APIS
AlchemyAPI, MINDOPS, Google, bigml, indico, ALGORITHMIA, Expect Labs, PredictionIO

IMAGE RECOGNITION
clarifai, MADBITS, DNNresearch, DEXTRO, VISENZE, lookflow

SPEECH RECOGNITION
GRIDSPACE, popup archive, NUANCE

RETHINKING ENTERPRISE

SALES
Preact, AVISO, RelateIQ, NG DATA, Help, FRAMED, CLARABRIDGE, infer, ATTENSITY, causata

SECURITY / AUTHENTICATION
CROSSMATCH, conjur, EYEVERIFY, BITSIGHT, CYLANCE, AREA 1 SECURITY, bionym

FRAUD DETECTION
sift science, socure, ThreatMetrix, feedzai, Brighterion, Verafin

HR / RECRUITING
TalentBin, entelo, predikt, Connectifier, gild, hiQ, CONCEPT NODE

MARKETING
brightfunnel, bloomreach, CommandIQ, AIRPR, RADIUS, TellApart, people pattern, Freshplum

PERSONAL ASSISTANT
Siri, x.ai, Google now, Cortana, cleversense, tempo, Robinlabs, KASISTO, fuse|machines, VIV, CLARA LABS

INTELLIGENCE TOOLS
ADATAO, Palantir, Quid, Digital Reasoning, FirstRain

RETHINKING INDUSTRIES

ADTECH
METAMARKETS, dstillery, rocketfuel, YieldMo, ADBRAIN

AGRICULTURE
BLUE RIVER, TerrAvion, ceres imaging, HONEYCOMB, THE CLIMATE CORPORATION, tule

EDUCATION
declara, coursera, KNEWTON, kidaptive

FINANCE
Bloomberg, FinGenius, alphasense, KENSHO, minetta brook, Dataminr, BINATIX

LEGAL
Lex Machina, brightleaf, COUNSELYTICS, RAVEL, JUDICATA, Brevia, DiligenceEngine

MANUFACTURING
SIGHT MACHINE, MICROSCAN, BOULDER IMAGING, IVISYS

MEDICAL
Parzival, transcriptic, ZEPHYR HEALTH, Genescient, grand round table, bina, TUTE GENOMICS

OIL AND GAS
kaggle, AYASDI, TACHYUS, biota, Flutura

MEDIA / CONTENT
Outbrain, newsle, ARRIA, SAILTHRU, wavii, Owlin, NarrativeScience, Yseop, Summly, Prismatic, ai AUTOMATED INSIGHTS

CONSUMER FINANCE
Affirm, inVenture, zest finance, BILLGUARD, LendUp, LendingClub, Kabbage

PHILANTHROPIES
DataKind, thorn, DATA GUILD

AUTOMOTIVE
Google, Continental, TESLA, MOBILEYE, CRUISE

DIAGNOSTICS
enlitic, 3SCAN, lumiata, ENTOPSIS

RETAIL
BAY SENSORS, PRISM SKYLABS, celect, euclid

RETHINKING HUMANS / HCI

AUGMENTED REALITY
wearable intelligence, APX, blippar, META, layar

GESTURAL COMPUTING
THALMIC LABS, omek, FLUTTER, LEAP MOTION, eyeSight, 3Gear Systems, GestureTek, nod

ROBOTICS
intel, LIQUID ROBOTICS, SoftBank, iRobot, Boston Dynamics, jibo, anki, evolution robotics

EMOTIONAL RECOGNITION
affectiva, BEYOND VERBAL, emotient, cogito

SUPPORTING TECHNOLOGIES

HARDWARE
NVIDIA, XILINX, QUALCOMM, NERVANA SYSTEMS, TERADEEP, Artificial Learning, rigetti

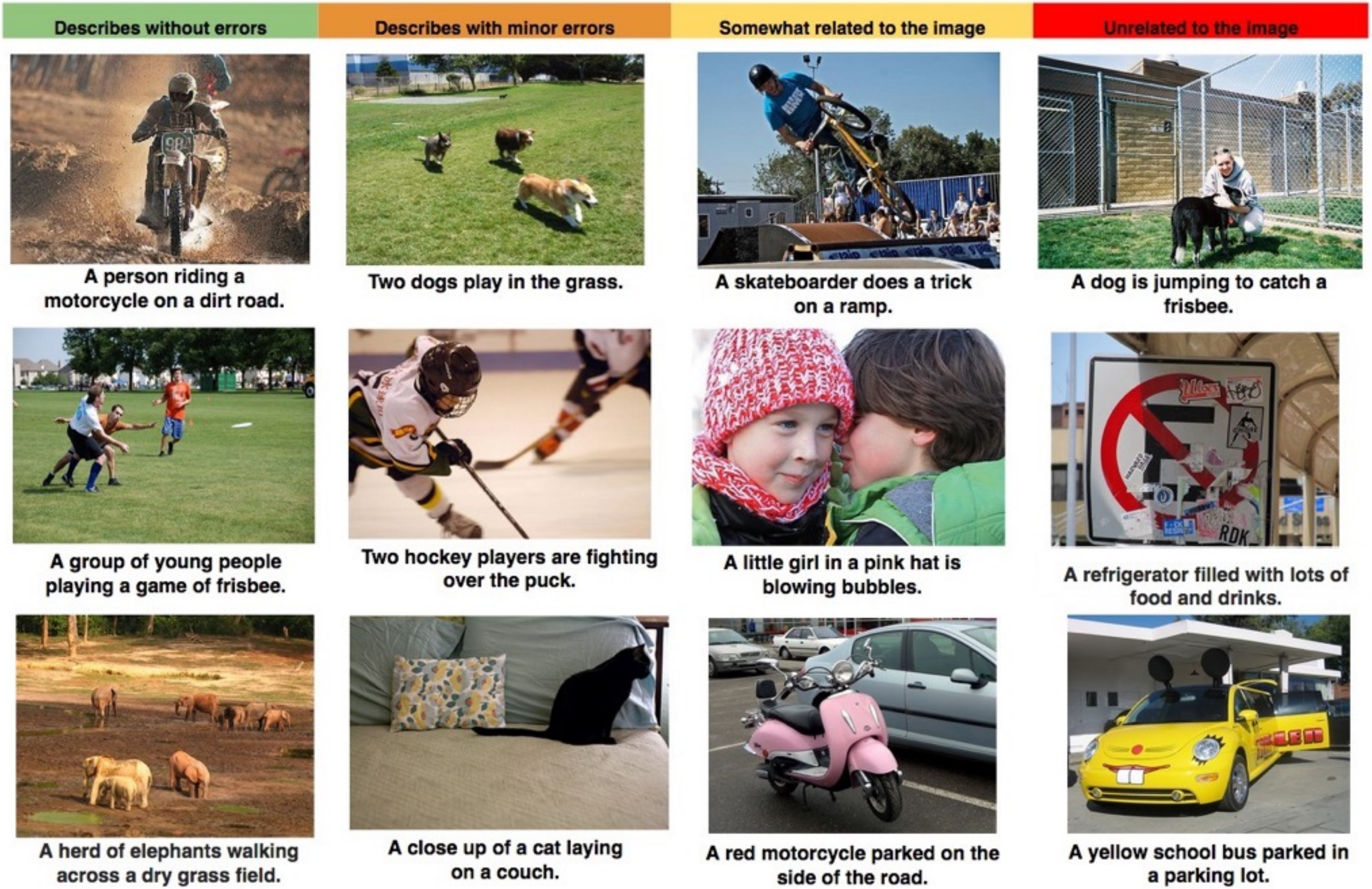
DATA PREP
TRIFACTA, Paxata, tamr, Alation

DATA COLLECTION
diffbot, kimono, CrowdFlower, Connotate, WorkFusion, import io

Computers have just learned …

… how to see, read, write and classify.

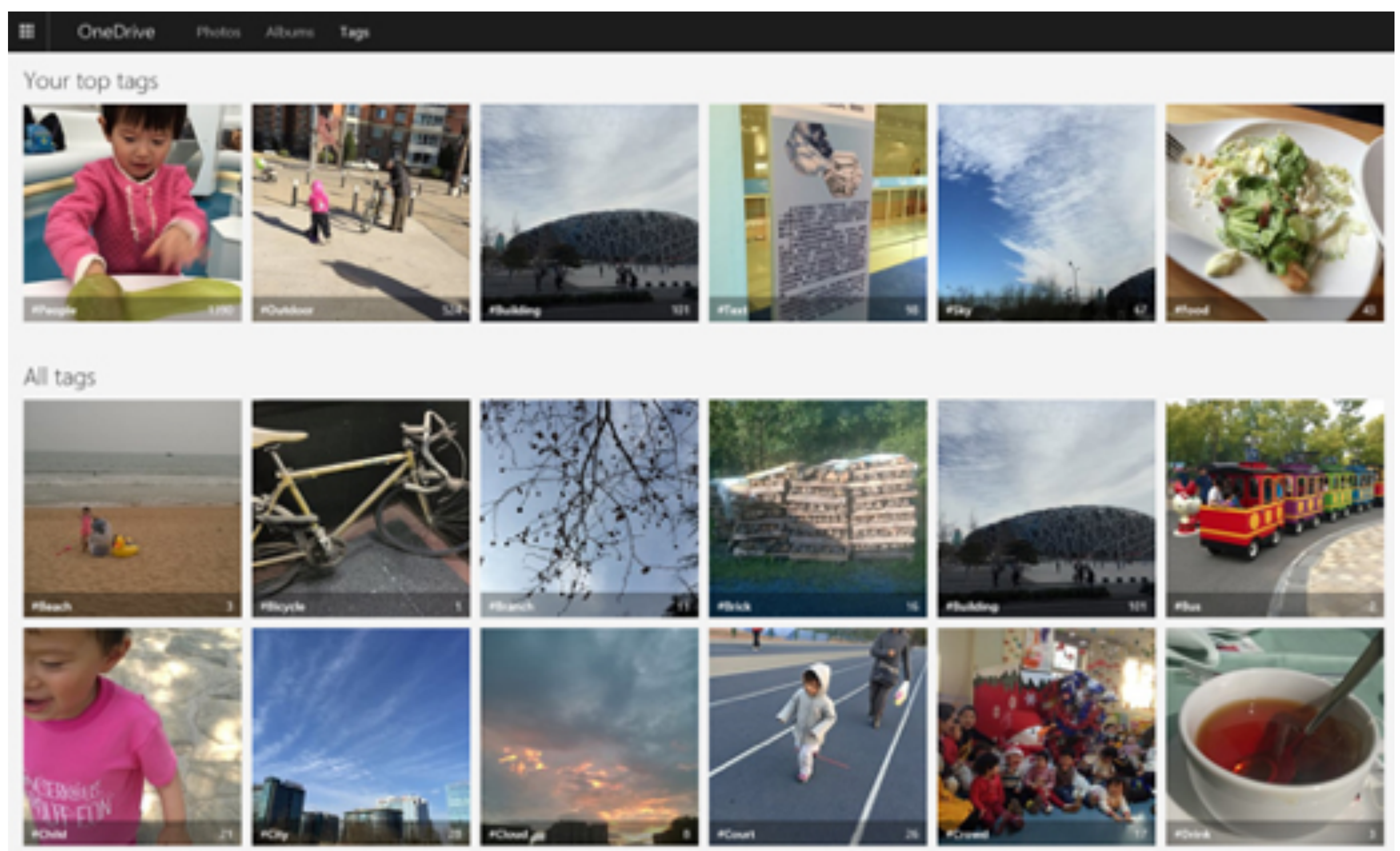
Jeremy Howard
go.ted.com/bbZC



Google Research Blog, November 2014
googleresearch.blogspot.ch/2014/11/[…]

Superhuman Image Recognition

- With convolutional neural networks
- 1.2 m training images, ~ 30 layers



Inside Microsoft Research, February 2015
blogs.technet.com/b/inside_microsoft_research/[…]

Outline

- Introduction to Topic
- Regression Analysis
- Reputation Systems
- Dimensionality Reduction
- Artificial Neural Networks

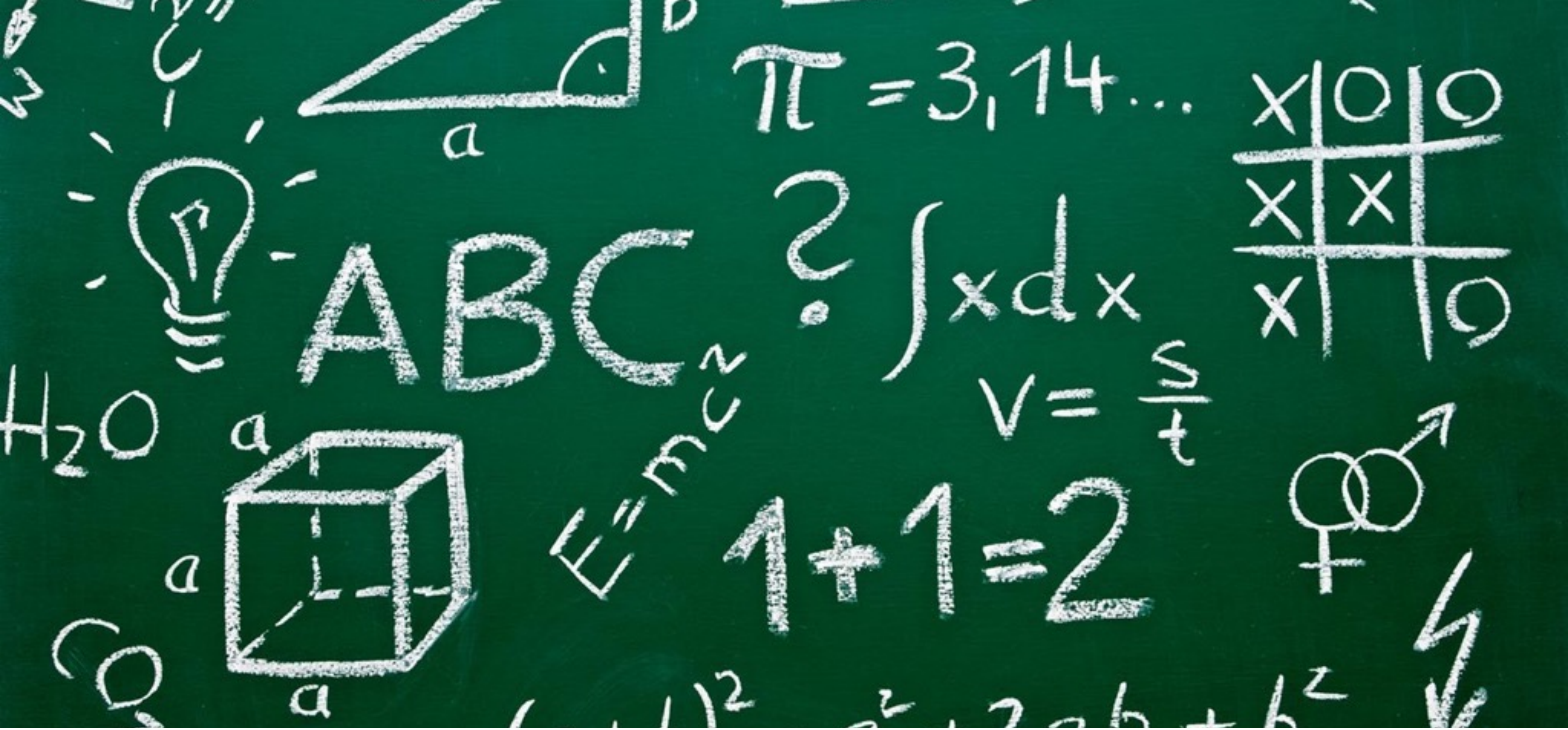
About Myself

- MSc ETH Zurich in Computer Science
- Focus on cryptography and compilers
- Thus I’m not an expert on these topics

Synapps

gbs switzerland

More information on:
www.kasparetter.com



Introduction to Topic

Definitions and Mathematical Prerequisites

Machine

A **machine** runs an **algorithm** which is

- a procedure for
- solving a specified problem
- in a finite number of steps
- (i.e. eventually producing an output)

Turing Machine
en.wikipedia.org/wiki/Turing_machine

Learning

- Building **predictive** models from data
- Algorithms improve with “experience”
- Useful if you don’t know the solution

- **Machine Learning**: Learn and predict
- **Data Mining**: Discover new properties

Machine Learning
en.wikipedia.org/wiki/Machine_learning

Feedback Mechanism

Categorization based on feedback:

- **Supervised Learning**: Given a set of inputs & outputs, learn a general rule
- **Unsupervised Learning**: Learn the structure in input without training set
- **Reinforcement Learning**: Achieve a certain goal in dynamic environment

Supervised Learning
en.wikipedia.org/wiki/Supervised_learning

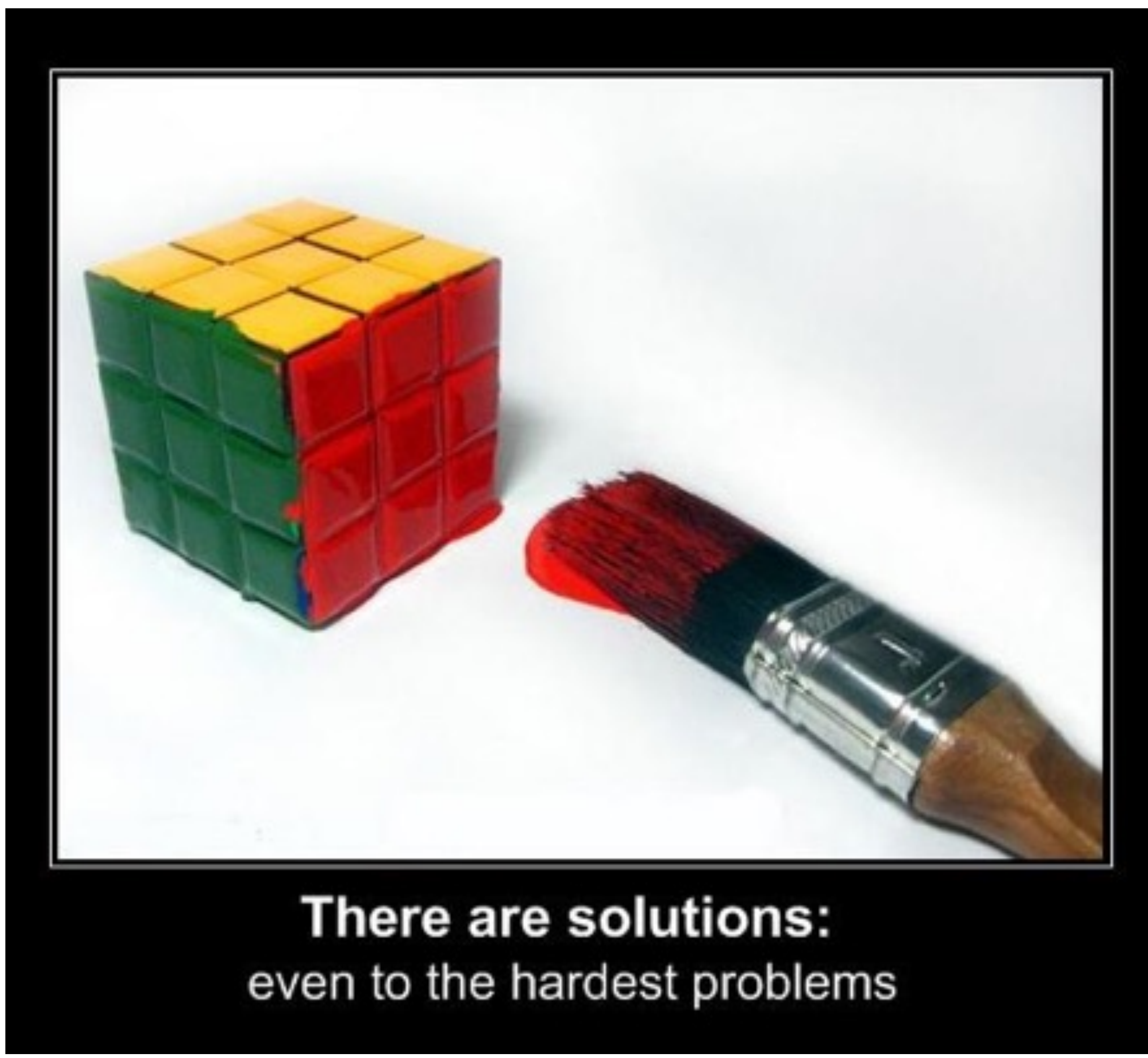
Tasks and Models

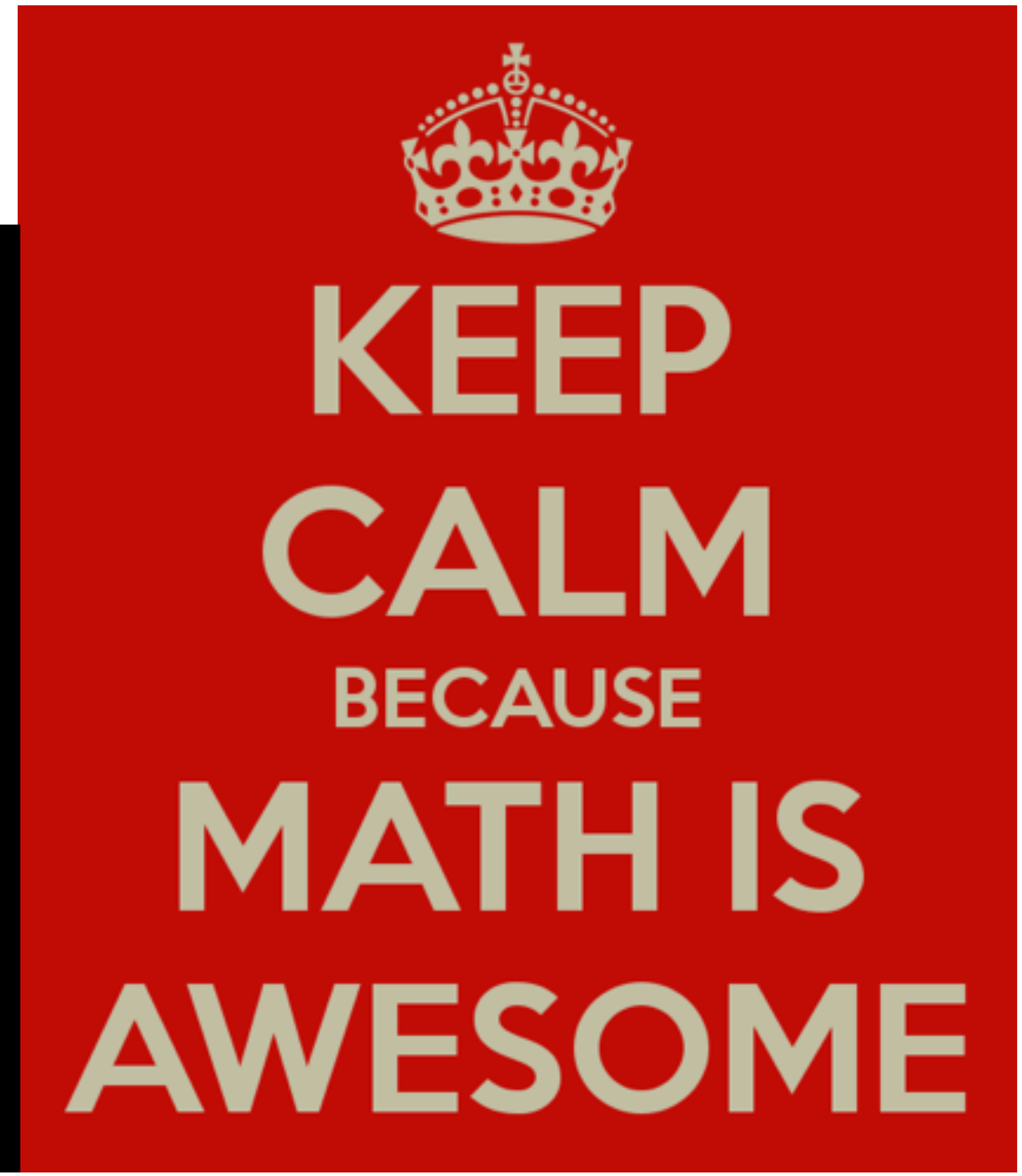
- **Classification**: Assign unseen input to descrete categories (supervised)
- **Regression**: Map the unseen input to continuous values (supervised)
- **Clustering**: Divide set of inputs into a number of groups (unsupervised)
- **Dimensionality Reduction**: Simplify

Statistical Classification
en.wikipedia.org/wiki/Statistical_classification

Math is your Friend

– not your enemy!

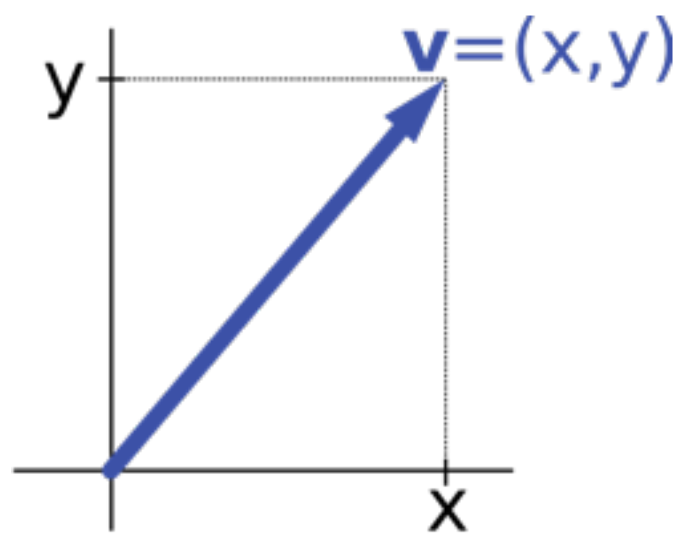


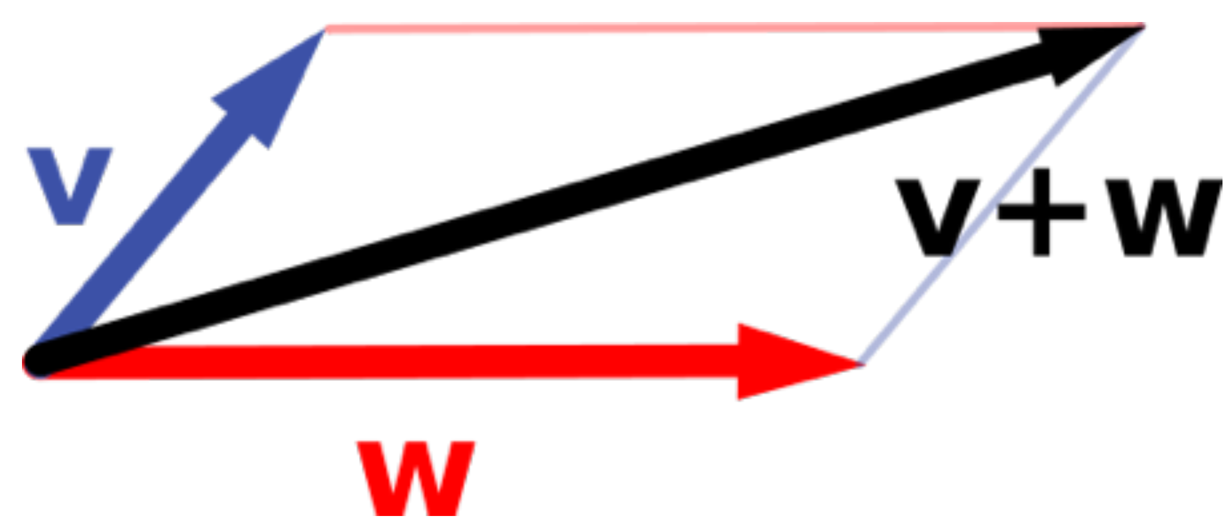


Numberphile Youtube Channel
www.youtube.com/user/numberphile

Dimensions and Vectors

- **Dimension:** A measurable extent of a space (e.g. a direction in 3D space)
- **Vector:** A point in space defined by a scalar value for each dimension

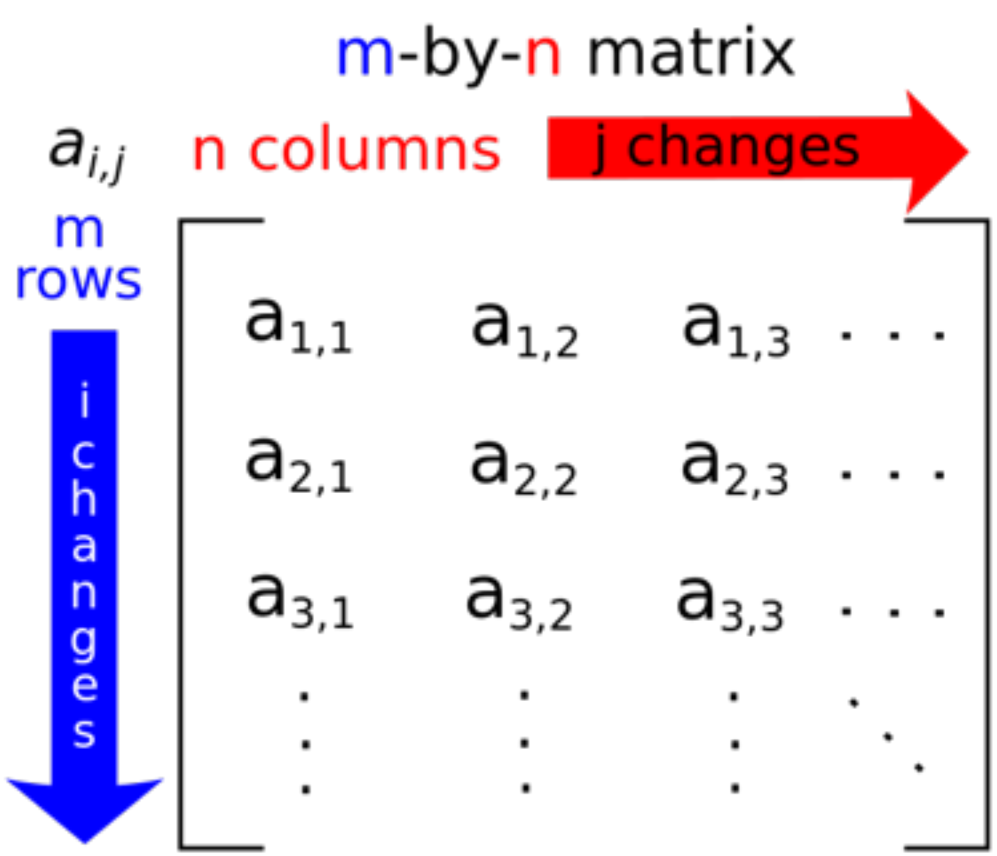




Vector Space with Certain Axioms
en.wikipedia.org/wiki/Vector_space

Matrices and Operations

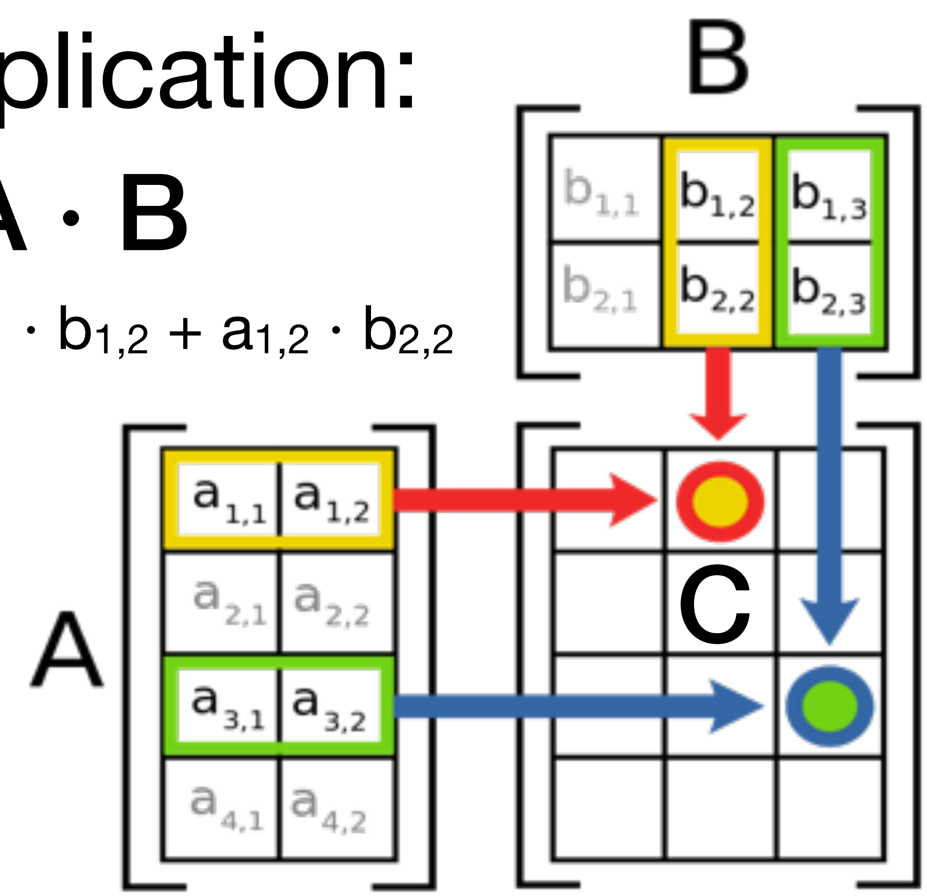
Matrix: 2-dimensional array of values
(a vector of vectors, one per column)



Multiplication:

$$C = A \cdot B$$

$$c_{1,2} = a_{1,1} \cdot b_{1,2} + a_{1,2} \cdot b_{2,2}$$



Matrix
en.wikipedia.org/wiki/Matrix_(mathematics)

Scalar Product

- **Algebraic Definition:** Element-wise multiplication and then summation, i.e. $\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + \ldots + a_n b_n$
- **Geometric Definition:** Product of the magnitudes and the cosine of angle between them: $\mathbf{a} \cdot \mathbf{b} = ||\mathbf{a}|| \; ||\mathbf{b}|| \cos\theta$
- $\mathbf{a} \perp \mathbf{b} \Rightarrow \mathbf{a} \cdot \mathbf{b} = 0$ and $\mathbf{a} \cdot \mathbf{a} = ||\mathbf{a}||^2$

Scalar Product a.k.a. Dot Product
en.wikipedia.org/wiki/Dot_product

Transpose

The **transpose** of a matrix **A** is another matrix $\mathbf{A}^T$ created by swapping the rows and columns: $[\mathbf{A}^T]_{ij} = [\mathbf{A}]_{ji}$

- $(\mathbf{A}^T)^T = \mathbf{A}$
- $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
- $(\mathbf{AB})^T = \mathbf{B}^T\mathbf{A}^T$

Thus: $\mathbf{a} \cdot \mathbf{b} = \mathbf{a}^T\mathbf{b}$

$$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

Transpose
en.wikipedia.org/wiki/Transpose

Transformations

A matrix with n columns and m rows maps vectors from R^n to R^m: $\mathbf{y} = \mathbf{A}\mathbf{x}$

This is a **linear transformation** as

- $\mathbf{A}(\mathbf{x} + \mathbf{y}) = \mathbf{A}\mathbf{x} + \mathbf{A}\mathbf{y}$ (additivity)
- $\mathbf{A}(\alpha\mathbf{x}) = \alpha\mathbf{A}\mathbf{x}$ (homogeneity)

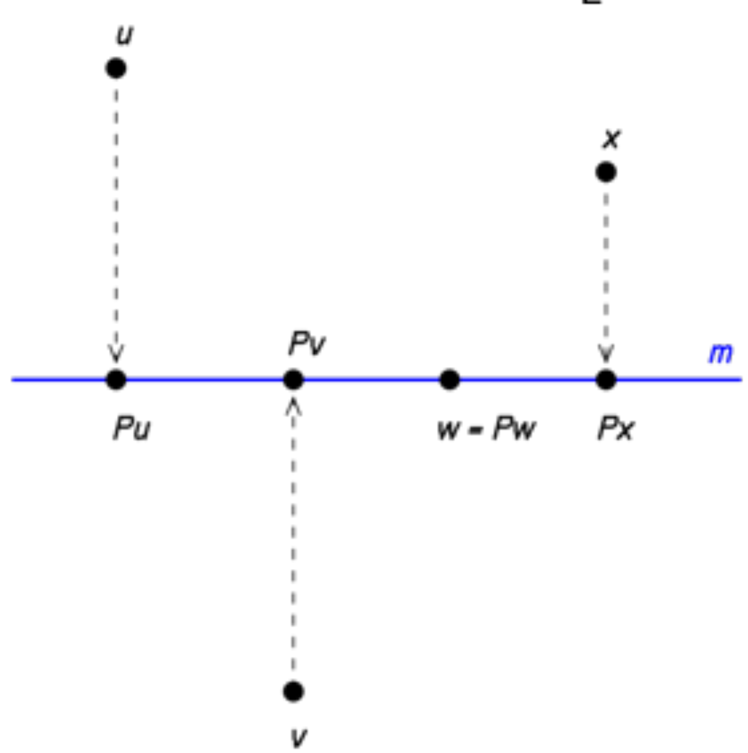
hold (with α being a scalar).

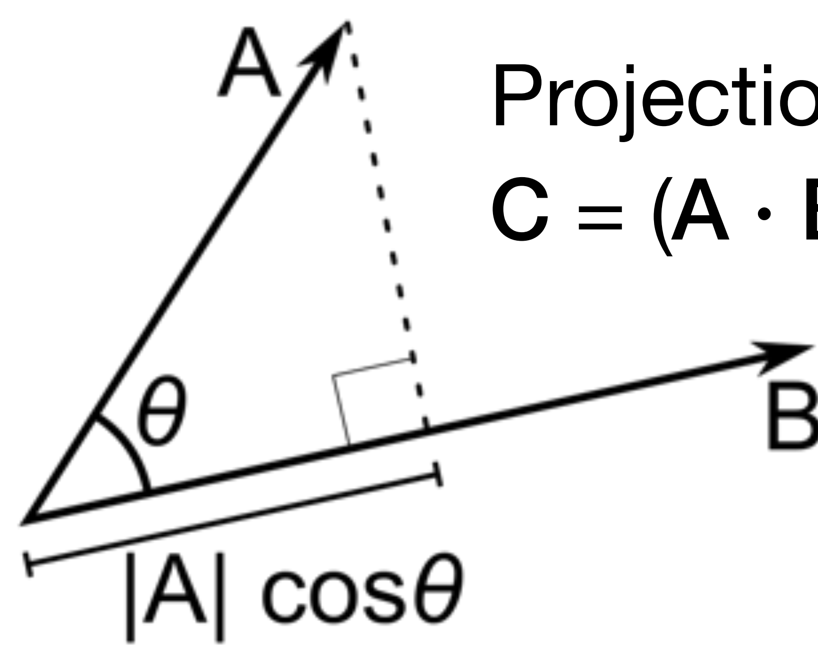
Transformation Matrix
en.wikipedia.org/wiki/Transformation_matrix

Projections

A projection **P** is a linear transformation with $\mathbf{P}^2 = \mathbf{P}$ (thus **P** is a square matrix).

Example: $P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}.\quad P\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x \\ y \\ 0 \end{pmatrix}.\quad P^2\begin{pmatrix} x \\ y \\ z \end{pmatrix} = P\begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = P\begin{pmatrix} x \\ y \\ z \end{pmatrix}.$

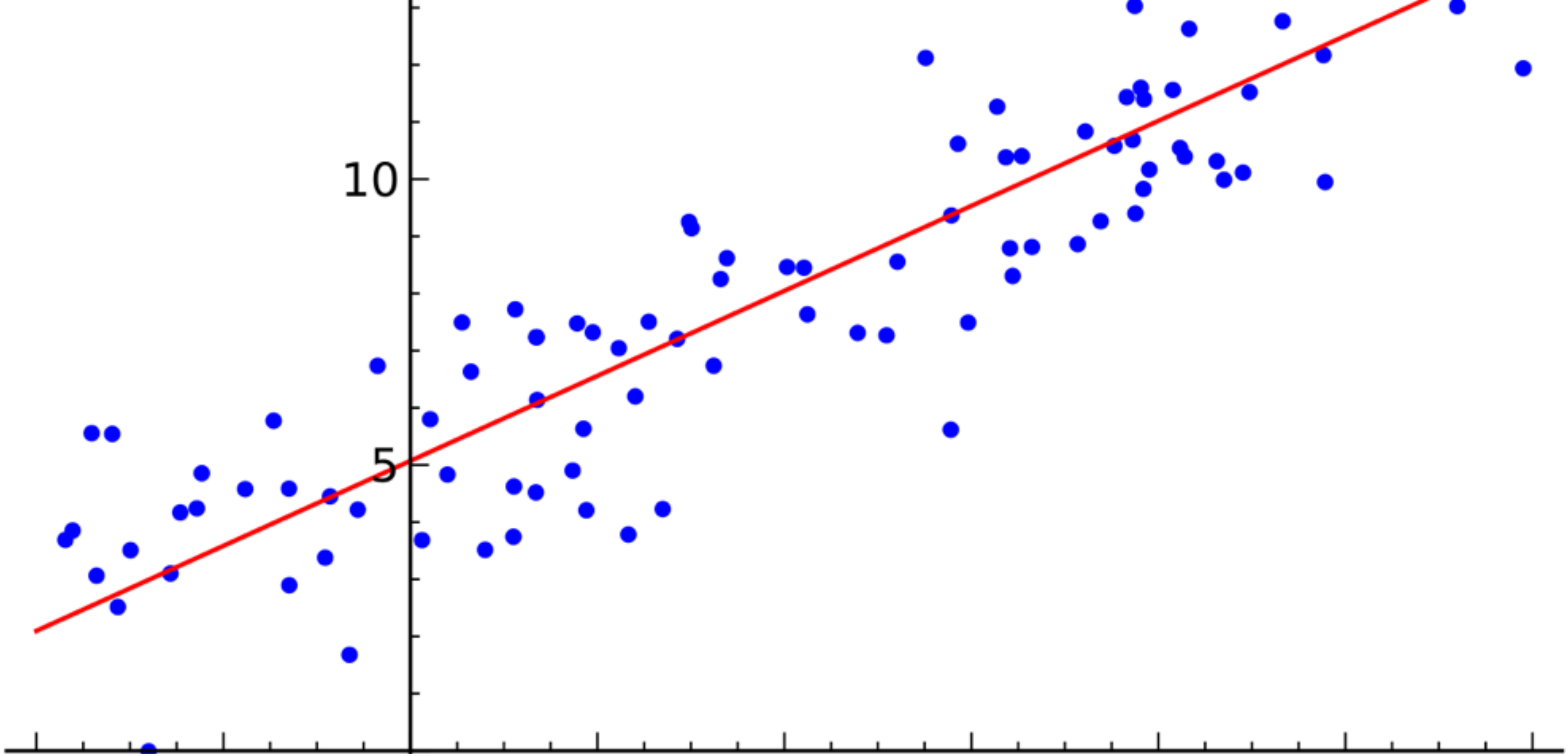




Projection of **A** on **B**:

$\mathbf{C} = (\mathbf{A} \cdot \mathbf{B})\,\mathbf{B} \,/\, \underbrace{\|\mathbf{B}\|^2}_{\text{Normalization}}$

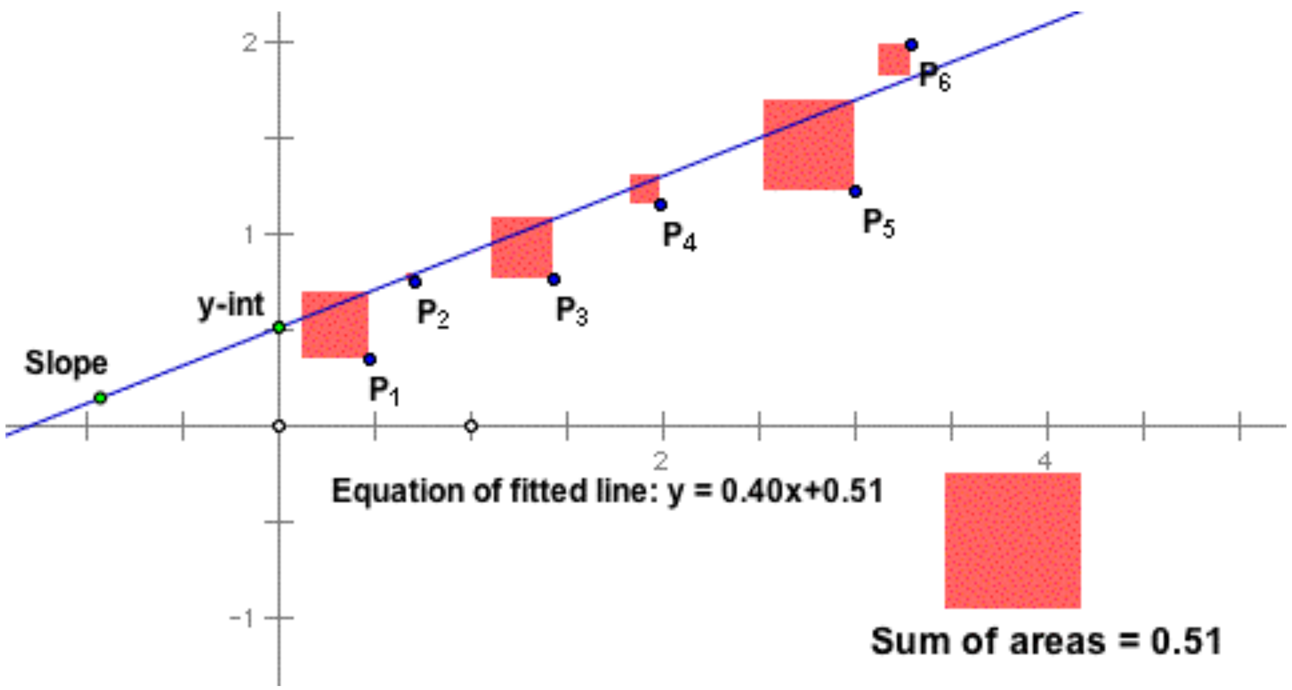
Projection
en.wikipedia.org/wiki/Projection_(linear_algebra)



Regression Analysis

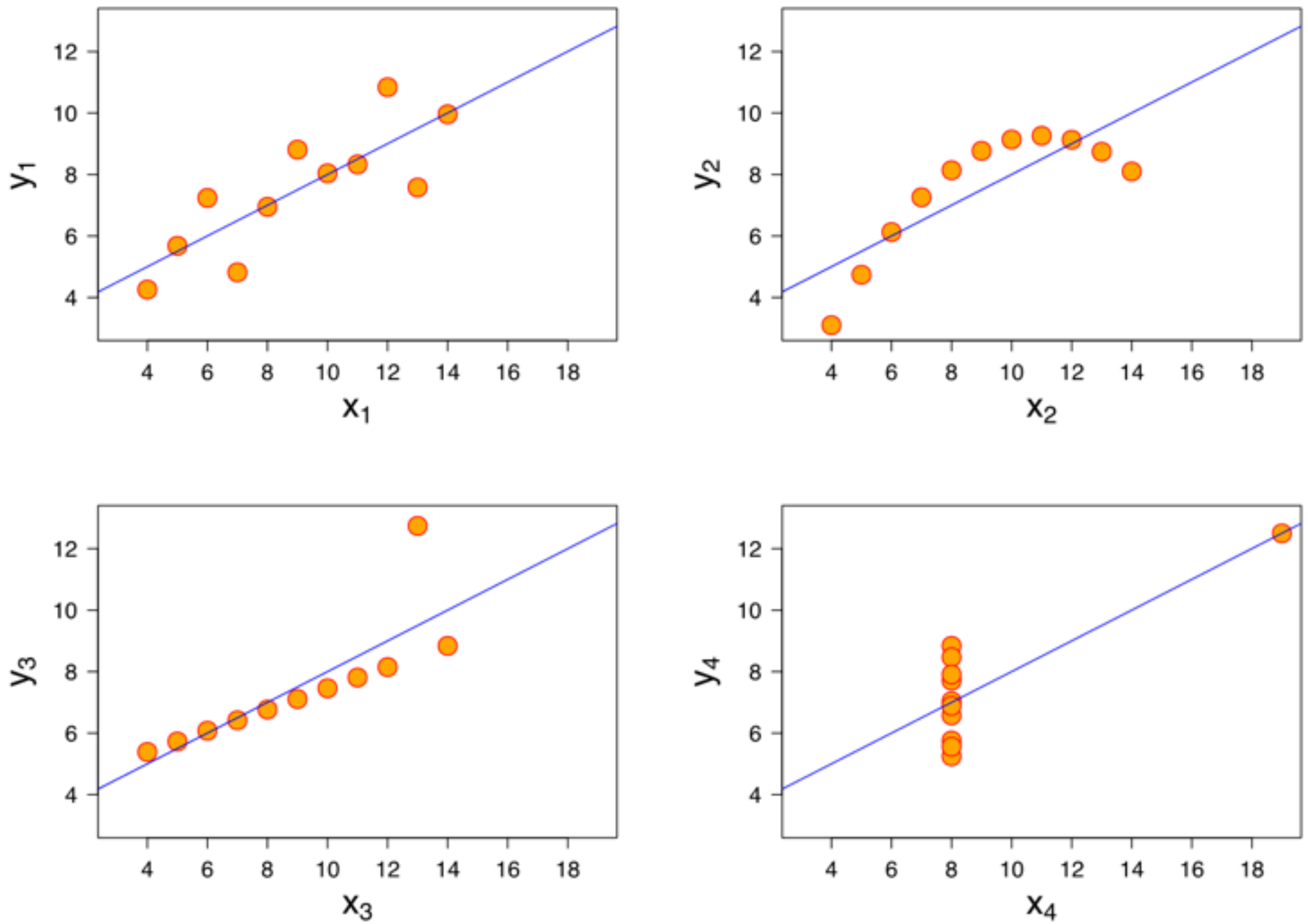
Solving Overdetermined Equations

Problem: Finding Trend Lines



Minimize the area of the squares!

Anscombe’s Quartet



Anscombe’s Quartet
en.wikipedia.org/wiki/Anscombe%27s_quartet

System of Equations

m **linear equations** with *n* **unknowns:**

- System notation:

$$\begin{aligned} a_{11}x_1 + \; & a_{12}x_2 + \cdots + \; a_{1n}x_n = b_1 \\ a_{21}x_1 + \; & a_{22}x_2 + \cdots + \; a_{2n}x_n = b_2 \\ \vdots \quad\; & \quad \vdots \qquad\qquad\quad \vdots \qquad\; \vdots \\ a_{m1}x_1 + \; & a_{m2}x_2 + \cdots + a_{mn}x_n = b_m . \end{aligned}$$

- Vector notation:

$$x_1 \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} + x_2 \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} + \cdots + x_n \begin{bmatrix} a_{1n} \\ a_{2n} \\ \vdots \\ a_{mn} \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

- Matrix notation:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

$$A\mathbf{x} = \mathbf{b}$$

Gaussian Elimination

Solve **Ax** = **b** by multiplying and adding rows to one another and substitute back

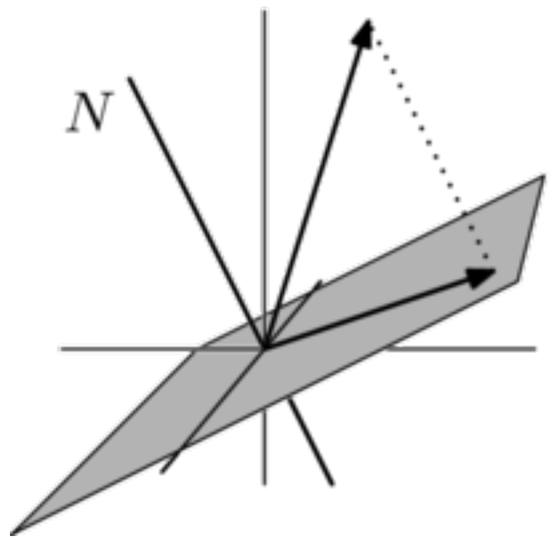
System of equations	Row operations	Augmented matrix
$\begin{aligned} 2x + y - z &= 8 \\ -3x - y + 2z &= -11 \\ -2x + y + 2z &= -3 \end{aligned}$		$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ -3 & -1 & 2 & -11 \\ -2 & 1 & 2 & -3 \end{array}\right]$
$\begin{aligned} 2x + y - z &= 8 \\ \frac{1}{2}y + \frac{1}{2}z &= 1 \\ 2y + z &= 5 \end{aligned}$	$L_2 + \frac{3}{2}L_1 \to L_2$ $L_3 + L_1 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 2 & 1 & 5 \end{array}\right]$
$\begin{aligned} 2x + y - z &= 8 \\ \frac{1}{2}y + \frac{1}{2}z &= 1 \\ -z &= 1 \end{aligned}$	$L_3 + -4L_2 \to L_3$	$\left[\begin{array}{ccc\|c} 2 & 1 & -1 & 8 \\ 0 & 1/2 & 1/2 & 1 \\ 0 & 0 & -1 & 1 \end{array}\right]$

Gaussian Elimination
en.wikipedia.org/wiki/Gaussian_elimination

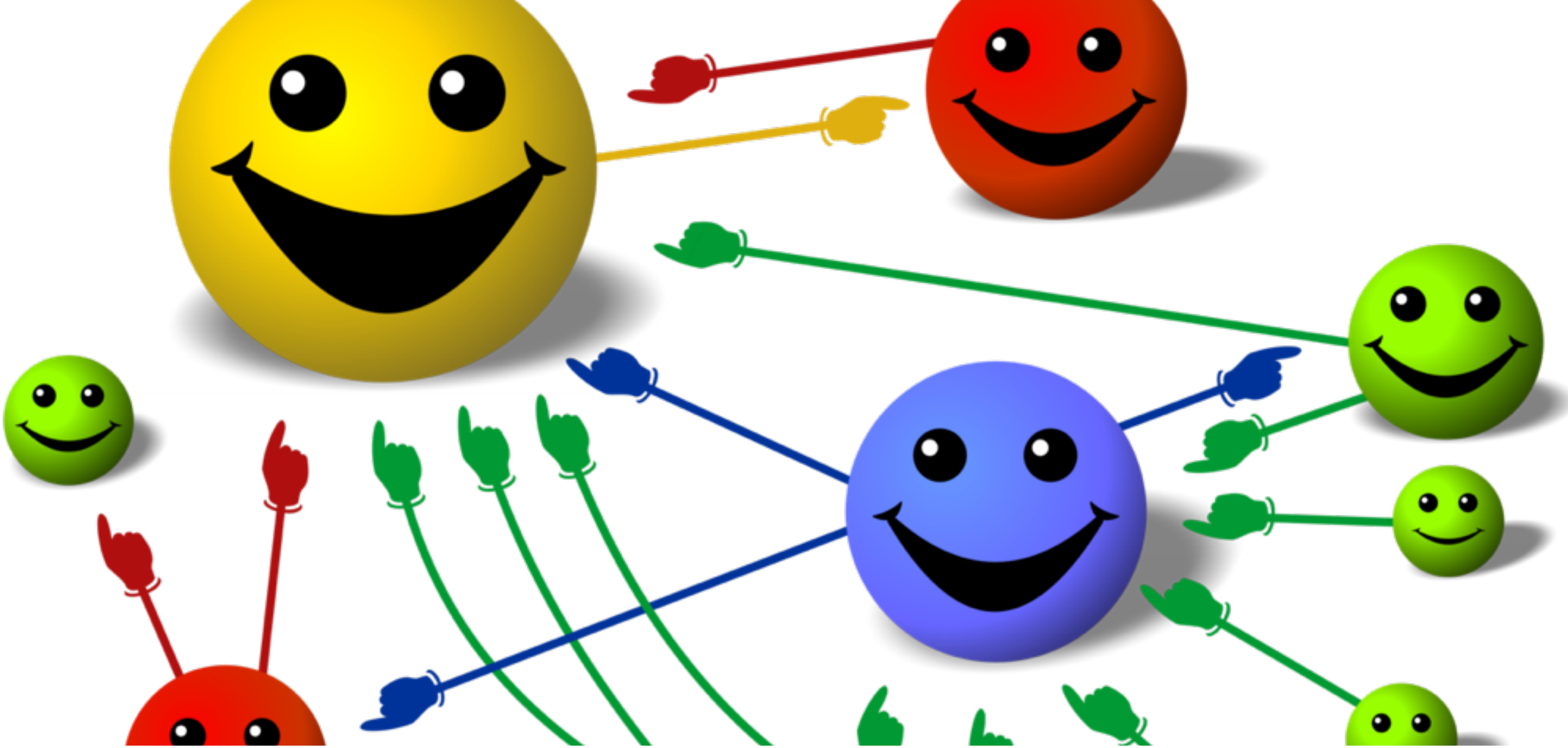
Solution: Orthogonal Projection

Solve overdetermined linear system $\mathbf{Ax} = \mathbf{y}$ so that the residual vector $\mathbf{r} = \mathbf{Ax} - \mathbf{y}$ is as small as possible.

Project $\mathbf{y}$ onto the columns of $\mathbf{A}$:
$\mathbf{A}^T\mathbf{Ax} = \mathbf{A}^T\mathbf{y}$ and $\mathbf{x} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{y}$.



Works for polynomials of all degrees.



Reputation Systems

Eigenvectors of Adjacency Matrix

Problem: Determine Reputation

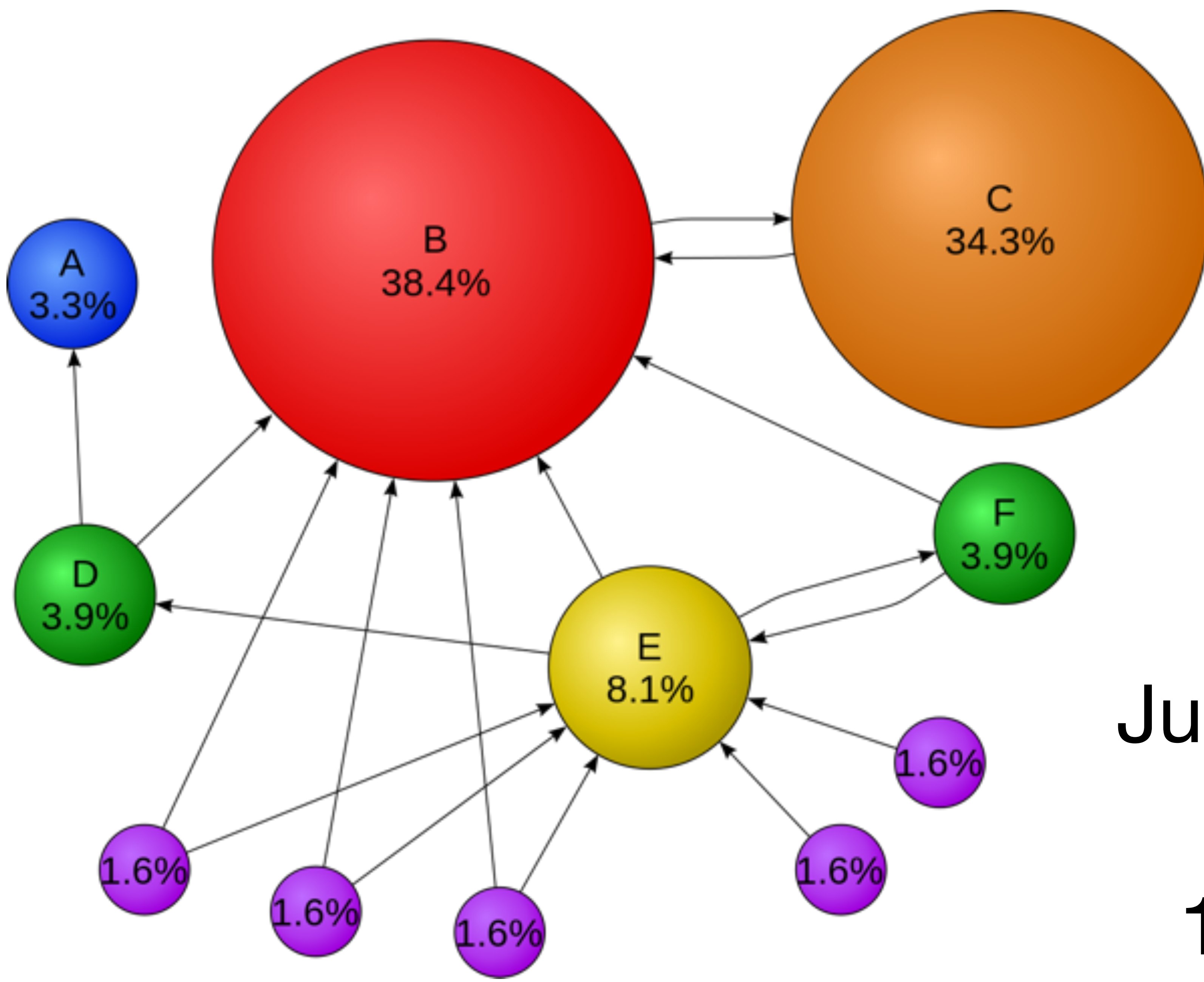
Compute a **reputation score** based on opinions by other entities about subject

Basic idea: Opinions of more reputable entities should count more than others

Applications: PageRank by Google Inc.

Reputation System
en.wikipedia.org/wiki/Reputation_system

Graph with Nodes and Edges

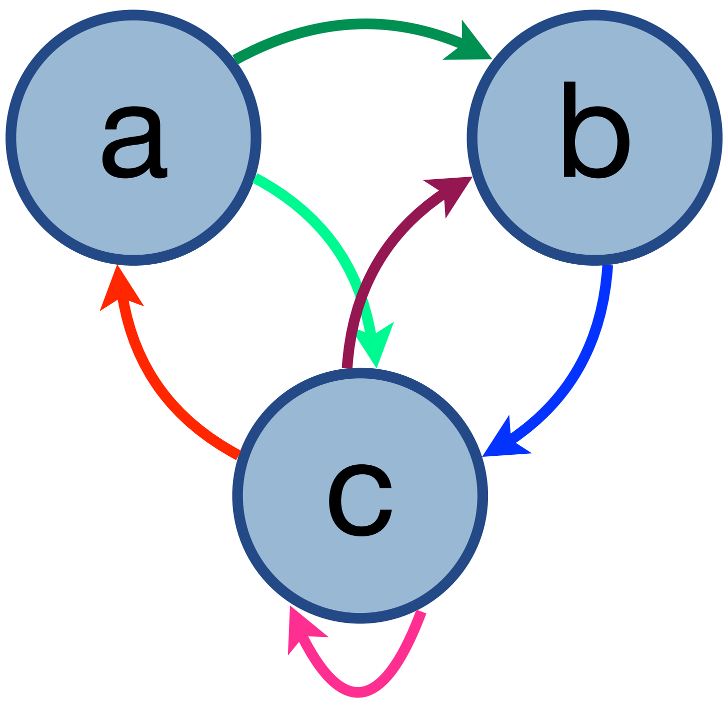


Abstract & rethink!

Prevent sinks:
Jump to arbitrary page with e.g. 15% likelihood!

PageRank
en.wikipedia.org/wiki/PageRank

Adjacency Matrix



A =

	a	b	c
a	0	0	1
b	1	0	1
c	1	1	1
∑	2	1	3

p =

a	1
b	0
c	0

Normalize A: A' =

	a	b	c
a	0	0	0.3
b	0.5	0	0.3
c	0.5	1	0.3
∑	1	1	1

Transition:

$$p' = A' \cdot p$$

Adjacency Matrix
en.wikipedia.org/wiki/Adjacency_matrix

Probabilistic Walk

- Make columns of **A** add up to 1
- Start at an arbitrary node
- Follow all outgoing edges
- Thus $\mathbf{p}_{new} = \mathbf{A}\,\mathbf{p}_{old}$
- Repeat indefinitely
- Converges to $\mathbf{p} = \mathbf{A}\,\mathbf{p}$

$\mathbf{p} =$

a	0.18
b	0.27
c	0.55

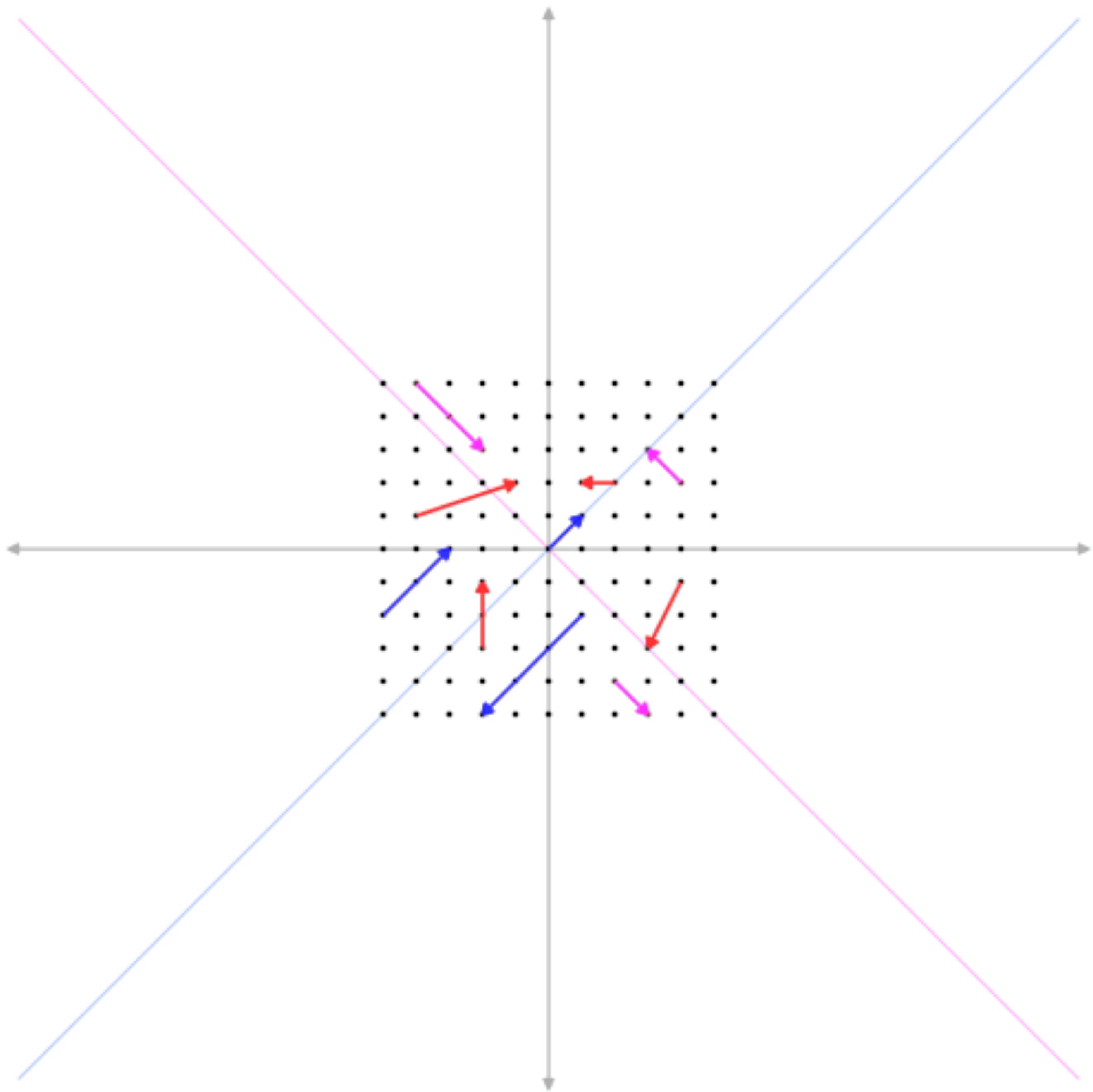
Solution: Compute Eigenvector

Solve equation $Av = \lambda v$

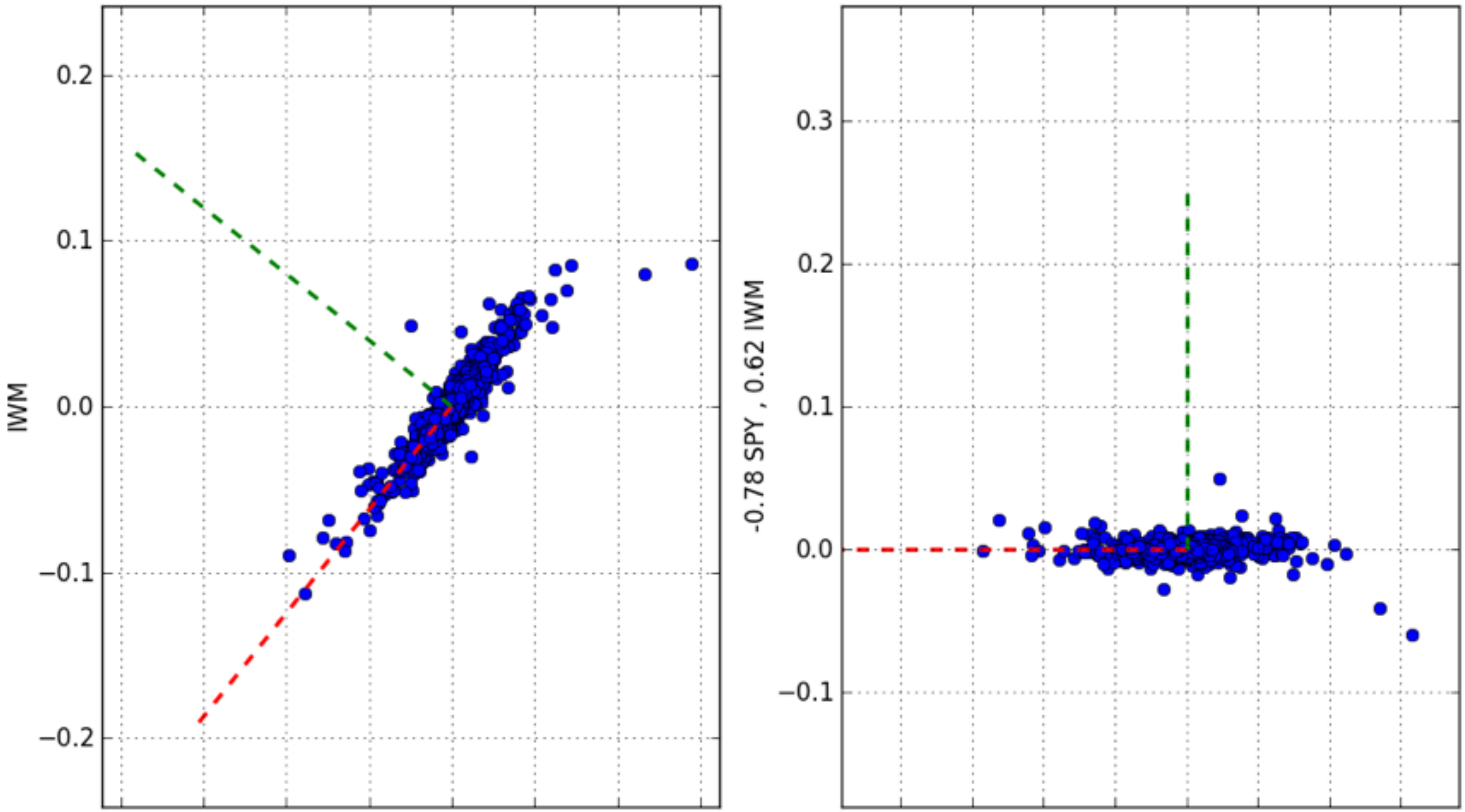
- **v** is called an **eigenvector** of **A**
- λ is its corresponding **eigenvalue**
- Calculate with **characteristic polynomial** of square matrix **A**

Eigenvalues and Eigenvectors
en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors

Eigenvector Visualization



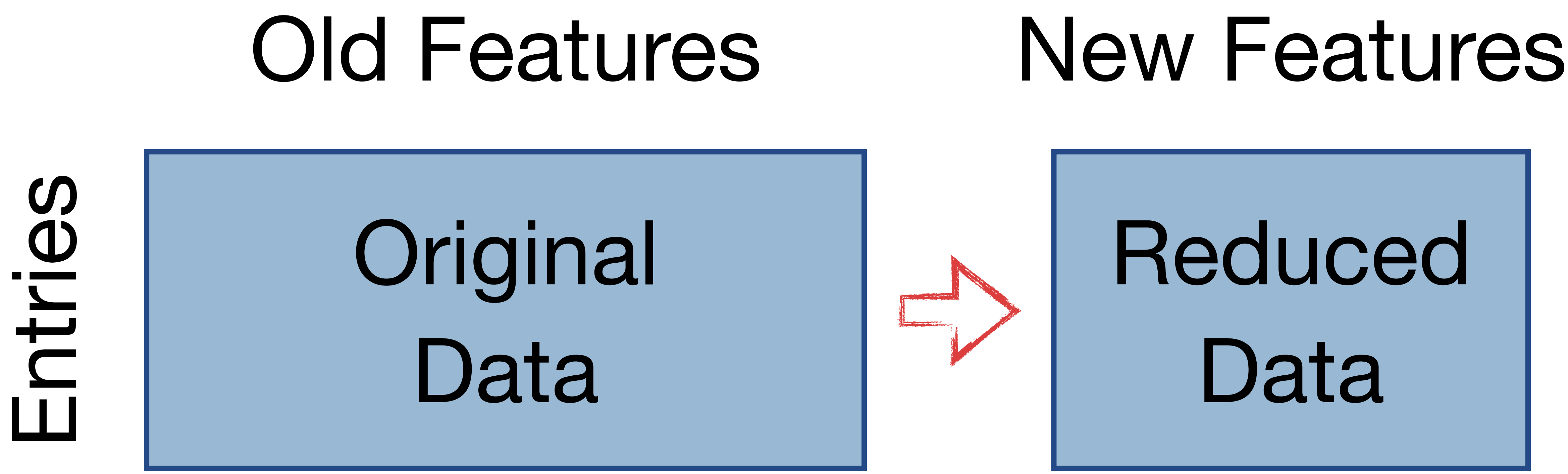
Eigenvalues and Eigenvectors
en.wikipedia.org/wiki/Eigenvalues_and_eigenvectors



Dimensionality Reduction

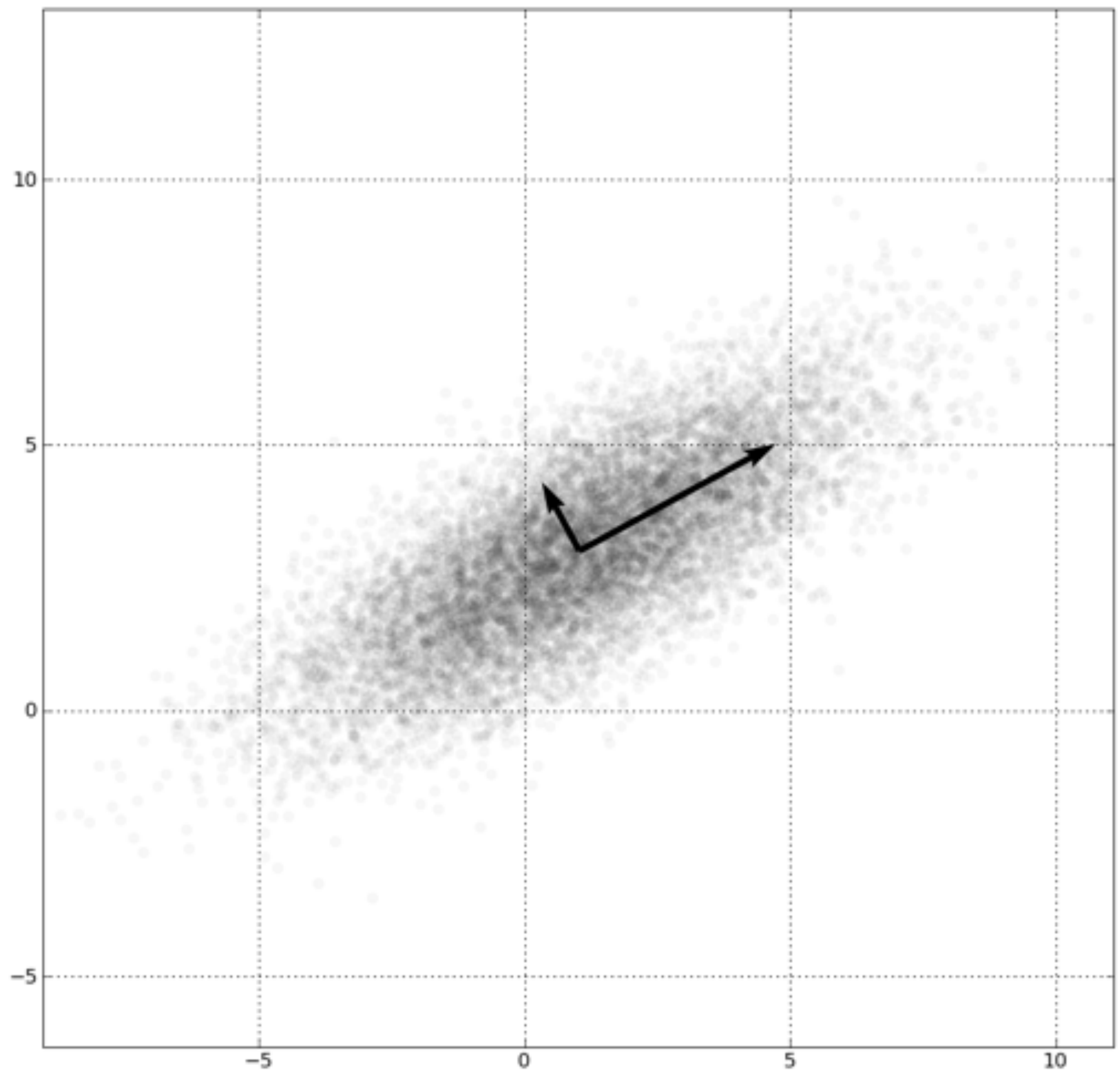
Projecting Data with Principal Component Analysis

Goal: Reduce Dimensionality



retain as much information as possible
with as little information as possible …

Principal Component Analysis



Covariance Matrix

- Calculate mean of *n* data vectors
  $\bar{\mathbf{x}} = (\mathbf{x}_1 + \mathbf{x}_2 + \ldots + \mathbf{x}_n) / n, \; \bar{\mathbf{X}} = [\bar{\mathbf{x}}, \ldots, \bar{\mathbf{x}}]$
- Calculate the covariance matrix
  $\mathbf{C} = \hat{\mathbf{X}} \cdot \hat{\mathbf{X}}^T / n$ with $\hat{\mathbf{X}} = \mathbf{X} - \bar{\mathbf{X}}$
- Compute the matrix of eigenvectors
  $\mathbf{CV} = \mathbf{VD}$; $\mathbf{D}$ contains eigenvalues $\ddots$
- Rearrange: Decreasing eigenvalues

Covariance Matrix
en.wikipedia.org/wiki/Covariance_matrix

Optimal Transformation

- **Goal:** Find unit vector **u** that maximiz. *variance* of data when projected on **u**
- **Variance:** $(\mathbf{u}^T \cdot \hat{\mathbf{X}}) \cdot (\mathbf{u}^T \cdot \hat{\mathbf{X}})^T / n = \mathbf{u}^T \mathbf{C}\, \mathbf{u}$
- Set $\lambda = \mathbf{u}^T \mathbf{C}\, \mathbf{u}$, thus $\mathbf{C}\, \mathbf{u} = \lambda\, \mathbf{u}$ $(\mathbf{U}^{-1} = \mathbf{U}^T)$
- The eigenvector **u** with the largest eigenvalue is the principal component!

Recommendation Systems

Example: If each song is a dimension, a user's taste can be viewed as vector.

Observation: As groups of users rate the same kind of songs high and low, try to distill the tastes as base vectors.

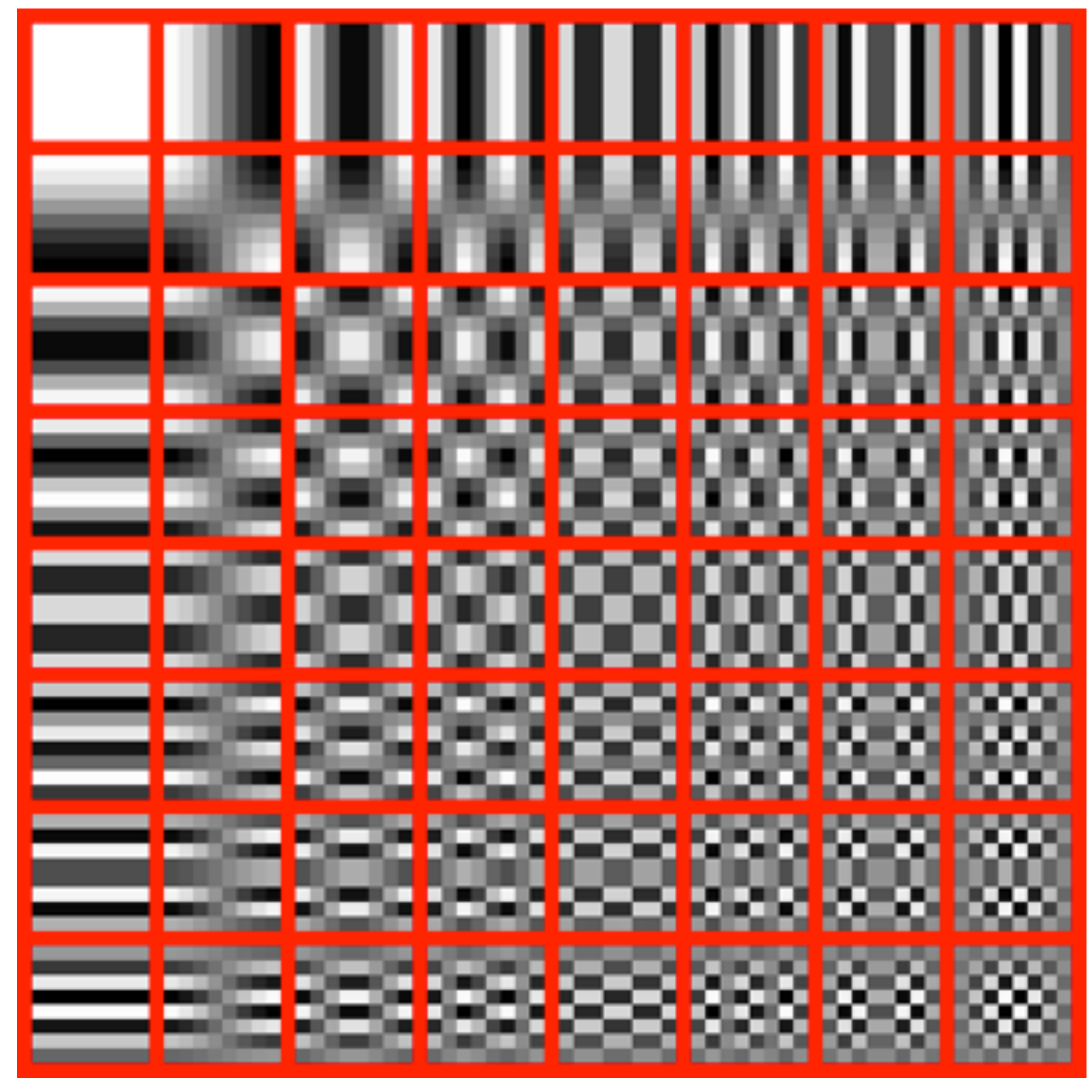
Apply **Principal Component Analysis!**

Recommender System
en.wikipedia.org/wiki/Recommender_system

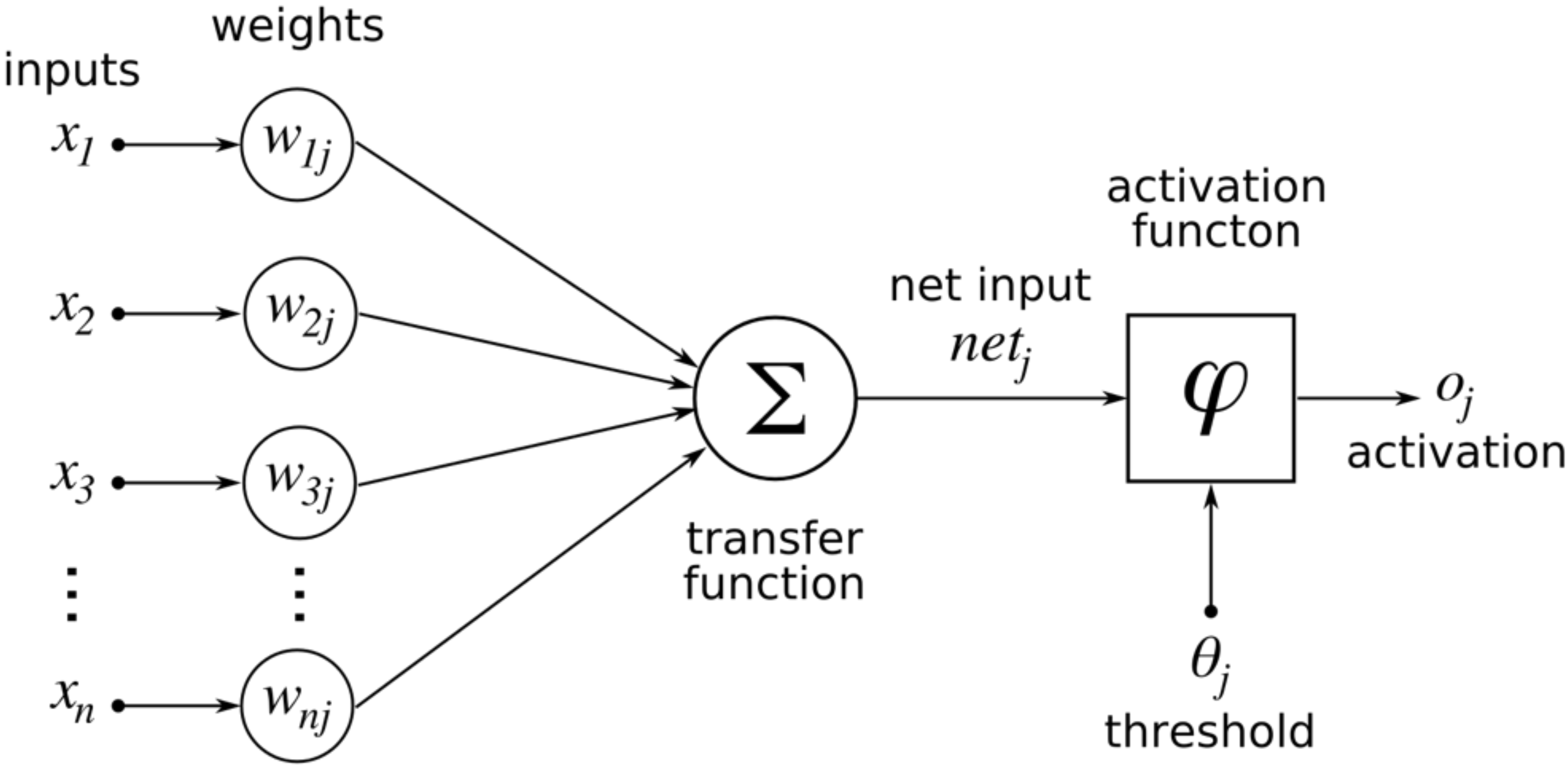
Lossy Compression

– Downside: New base vectors need to be stored to recover the data

– Solution: Choose appropriate base vectors (in case of JPEG: DCT)



Data Compression
en.wikipedia.org/wiki/Data_compression



Artificial Neural Networks

Optimizing Functions with Gradient Descent

Popularized: Deep Learning

Artificial Neural Networks (ANNs) are statistical learning algorithms inspired by biological neural networks.

ANNs are often used to approximate unknown functions that can depend on a large number of inputs and are hard to solve with handmade rules.

Deep Learning
en.wikipedia.org/wiki/Deep_learning

Types of Artificial Neural Nets

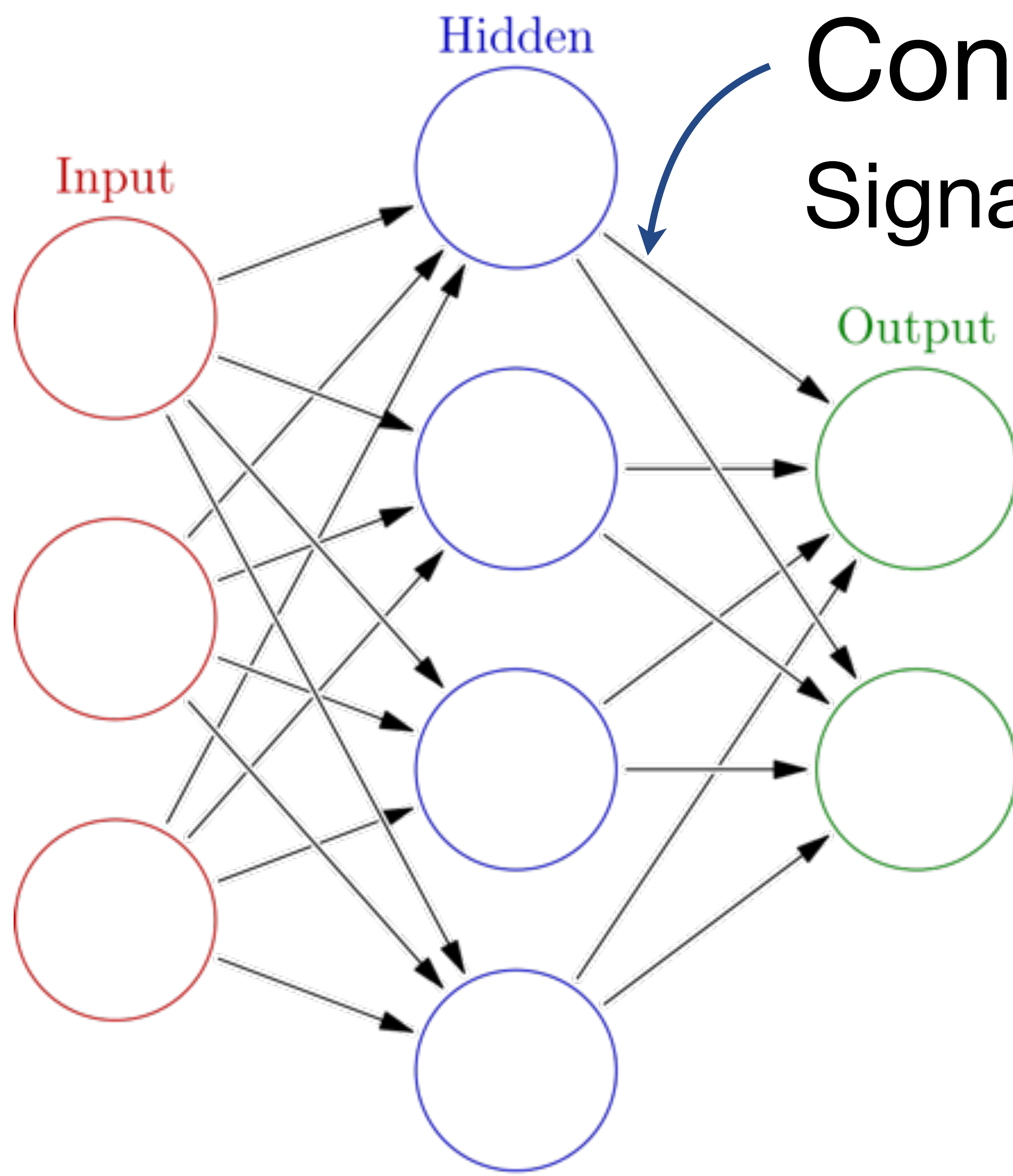
- **Feedforward Neural Networks:**
  Connections do not form any cycles
- **Convolutional Neural Networks:**
  Also acyclic but with shared weights
- **Recurrent Neural Networks:**
  With cycles to learn temporal pattern

Artificial Neural Network
en.wikipedia.org/wiki/Artificial_neural_network

Basic Setup of Neurons

Connection for Signal Forwarding

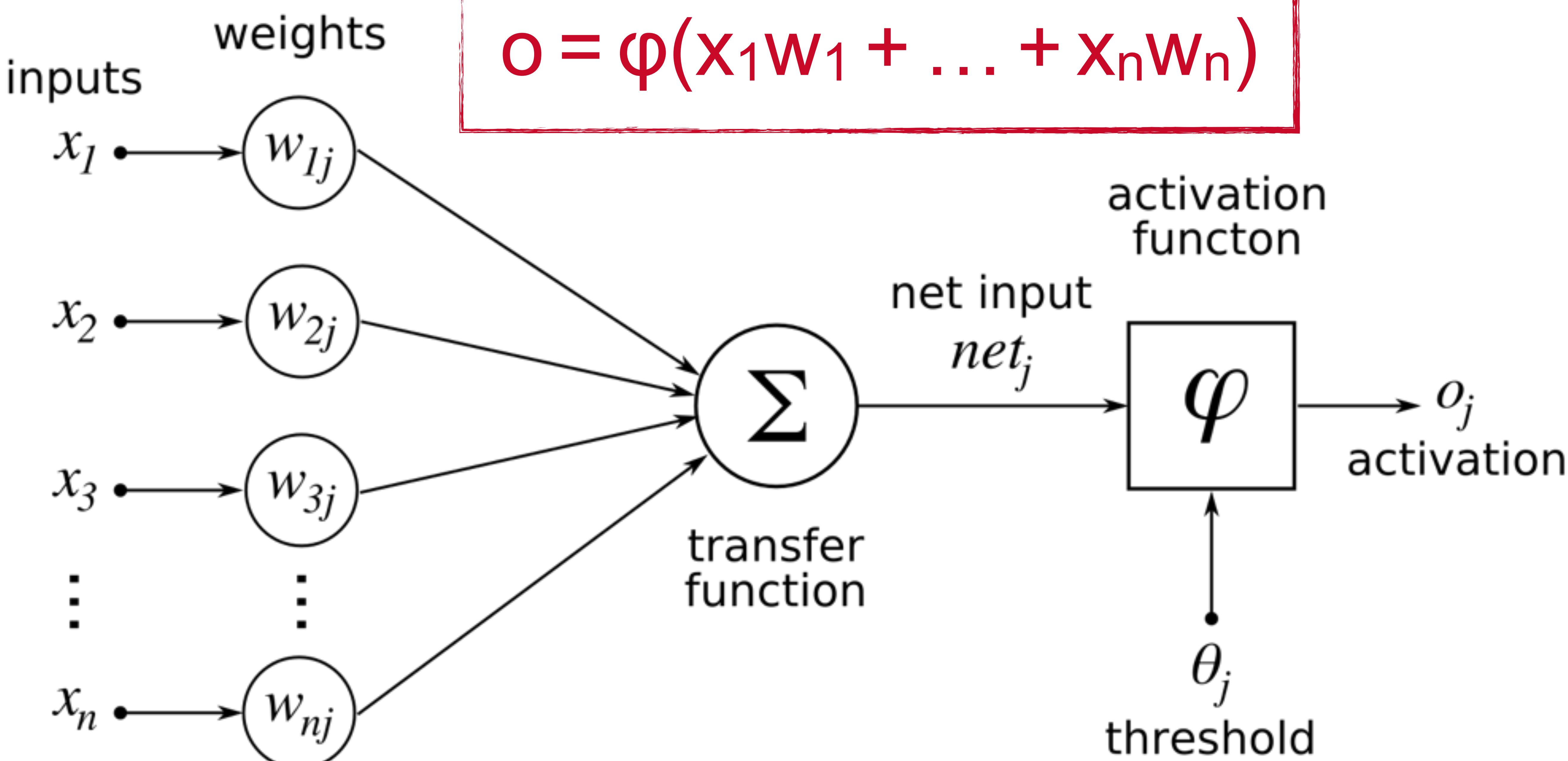
Neuron:



Feedforward Neural Network
en.wikipedia.org/wiki/Feedforward_neural_network

Activation Function

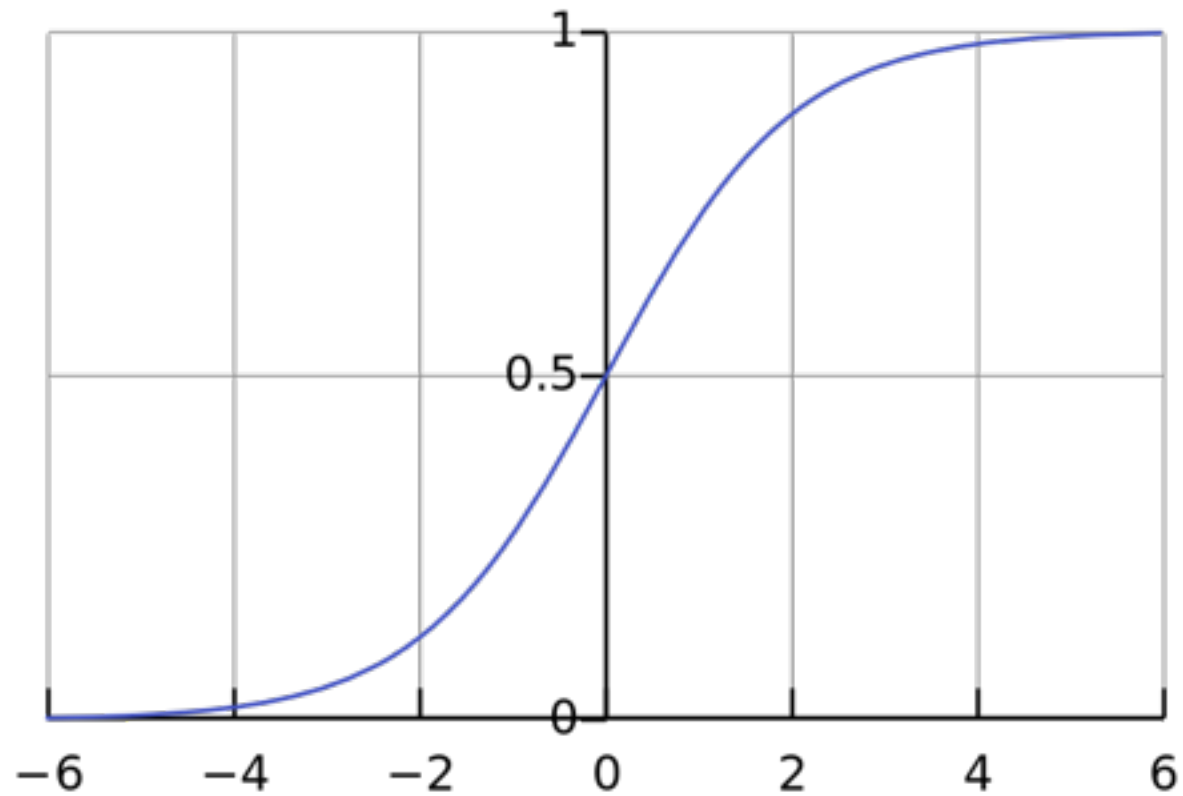
$$o = \varphi(x_1 w_1 + \ldots + x_n w_n)$$



Activation Functions
en.wikibooks.org/wiki/[…]/Activation_Functions

Activation Functions

- May **not be linear** or layers collapse
- **Sigmoid functions** like $f(x) = (1 + e^{-x})^{-1}$
- Also popular: **rectified linear** $f(x) = \max(0, x)$ with threshold



Optimization Problem

Goal: Find weights that minimize the mean squared error $E = ½ (f(\mathbf{x}) - \mathbf{y})^2$

Given search space and **cost function**, apply ordinary optimization algorithms:

- Evolutionary Algorithm
- Gradient Descent

Mathematical Optimization
en.wikipedia.org/wiki/Mathematical_optimization

Evolutionary Algorithm

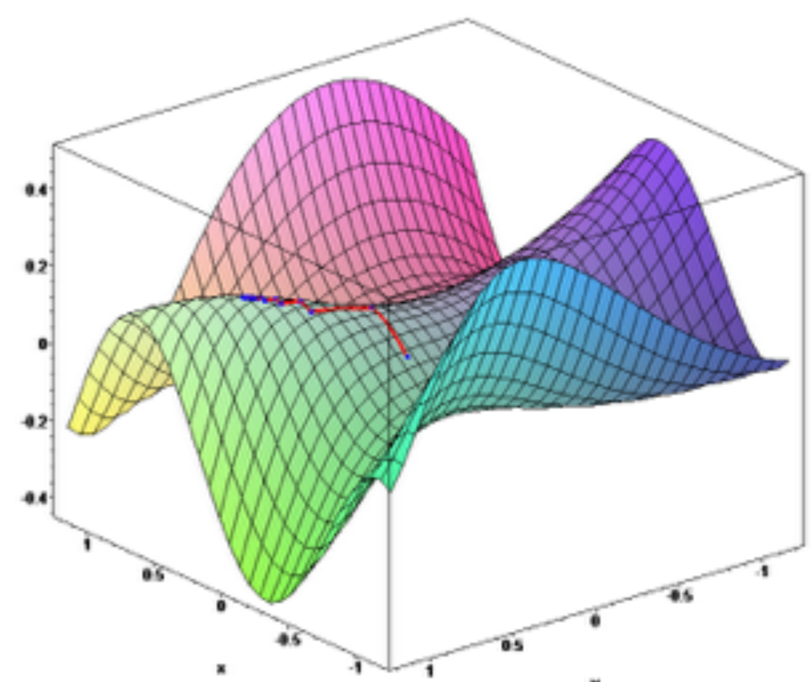
Inspired by biological evolution, apply **reproduction, mutation, recombination** and **selection** with the mean squared error as **fitness function** on parameters.

- No assumptions made about space
- Difficulty in finding a good crossover

Evolutionary Algorithm
en.wikipedia.org/wiki/Evolutionary_algorithm

Gradient Descent

- **Goal**: Find minimum of function F(**x**)
- Take steps proportional to negative of the **gradient**: $\mathbf{x}_2 = \mathbf{x}_1 - \gamma \nabla F(\mathbf{x}_1)$
- Only **local minimum**
- Function needs to be **differentiable**



Partial Derivative

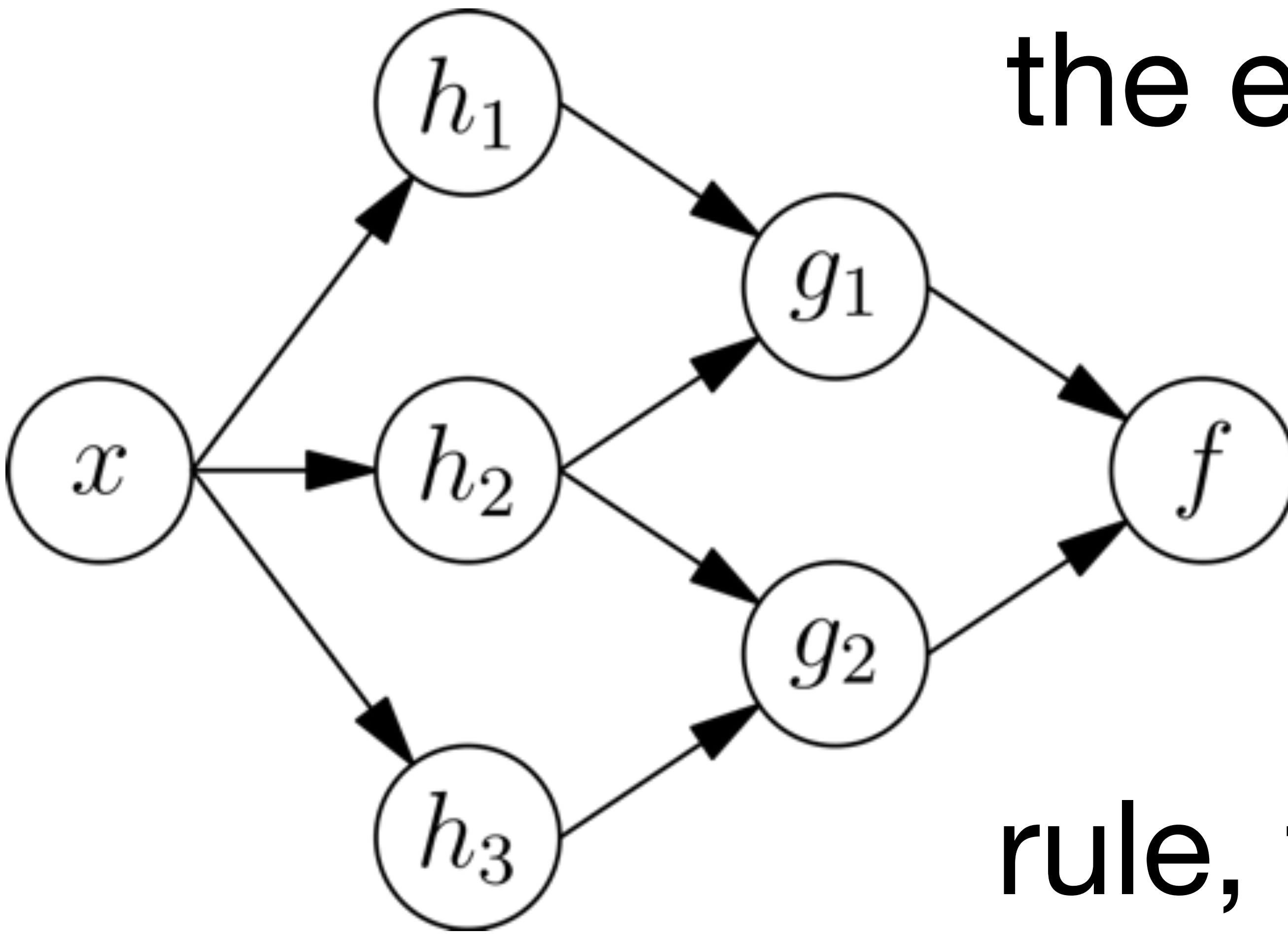
Derivative with respect to one variable

Example: $f(x, y) = x^2 + xy + y^2$, then the partial derivative wrt. x is $\partial f/\partial x = 2x + y$

Nabla symbol ∇ denotes the **gradient:**
$\nabla f(x_1, \ldots, x_n)$ = vector $(\partial f/\partial x_1, \ldots, \partial f/\partial x_n)$

Chain rule: $(f(g(x)))' = f'(g(x)) \cdot g'(x)$

Backpropagation



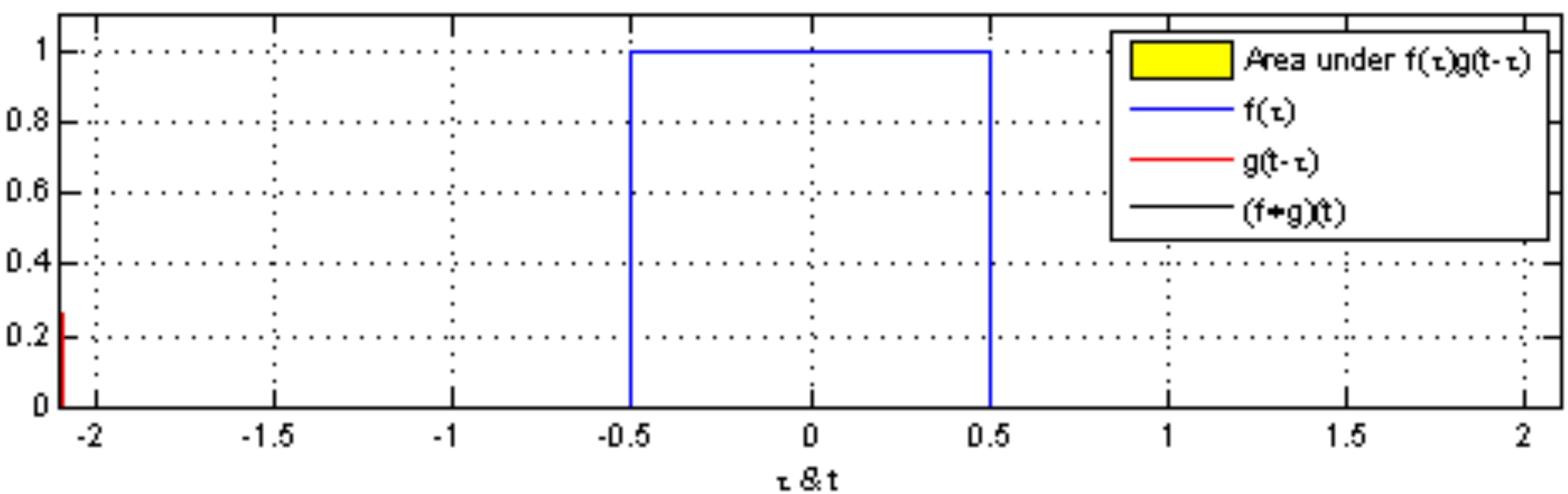
Calculate the gradient of the error with respect to each weight.

Due to the chain rule, the values of the right layers can be reused.

Convolution

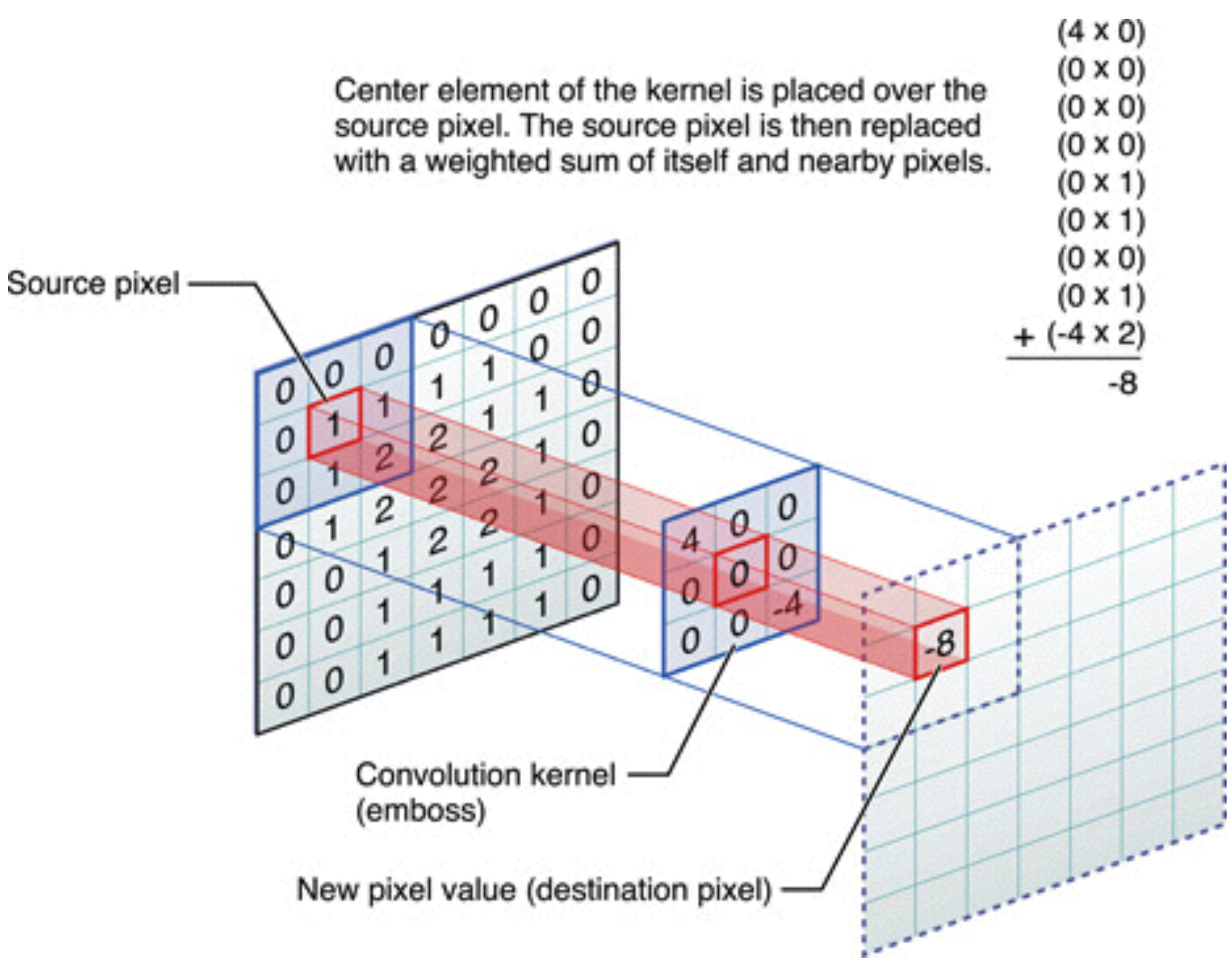
The integral of the product of two functions after one is reversed and shifted:

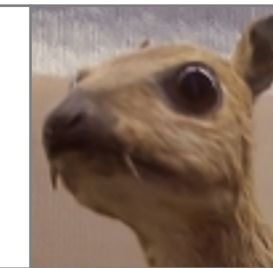
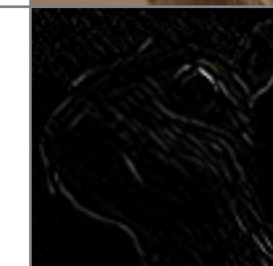
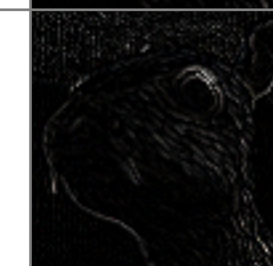
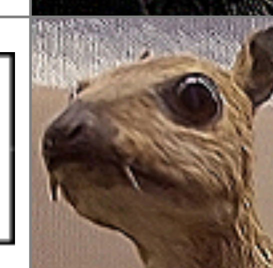
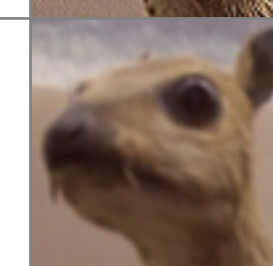
$(f * g)(t) = \int f(\tau)\, g(t - \tau)\, d\tau$ (as a definition)



Feature Detection

Shift kernel across function or image



Identity	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	
Edge detection	$\begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix}$	
	$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$	
	$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$	
Sharpen	$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix}$	
Box blur (normalized)	$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$	

Kernel (Image Processing)
en.wikipedia.org/wiki/Kernel_(image_processing)

Convolutional Neural Networks

- Feedforward network with input tiled such that there are overlapping areas
- Convolution where **kernels also learnt**
- **Weight sharing** reduces free variables
- Often with **rectified linear** units (ReLU)
- **Prevent overfitting** w. cross-validation

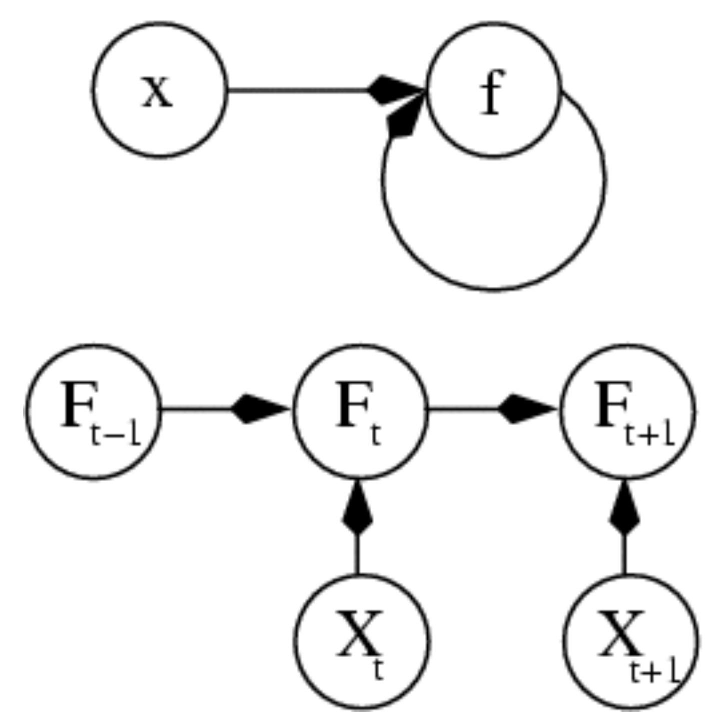
Convolutional Neural Network
en.wikipedia.org/wiki/Convolutional_neural_network

Recurrent Neural Networks

Connections between neurons form a directed **cycle**, creating internal states

Useful for processing **sequences of inputs** (like speech)

Most RNNs can compute anything a conventional computer can compute!

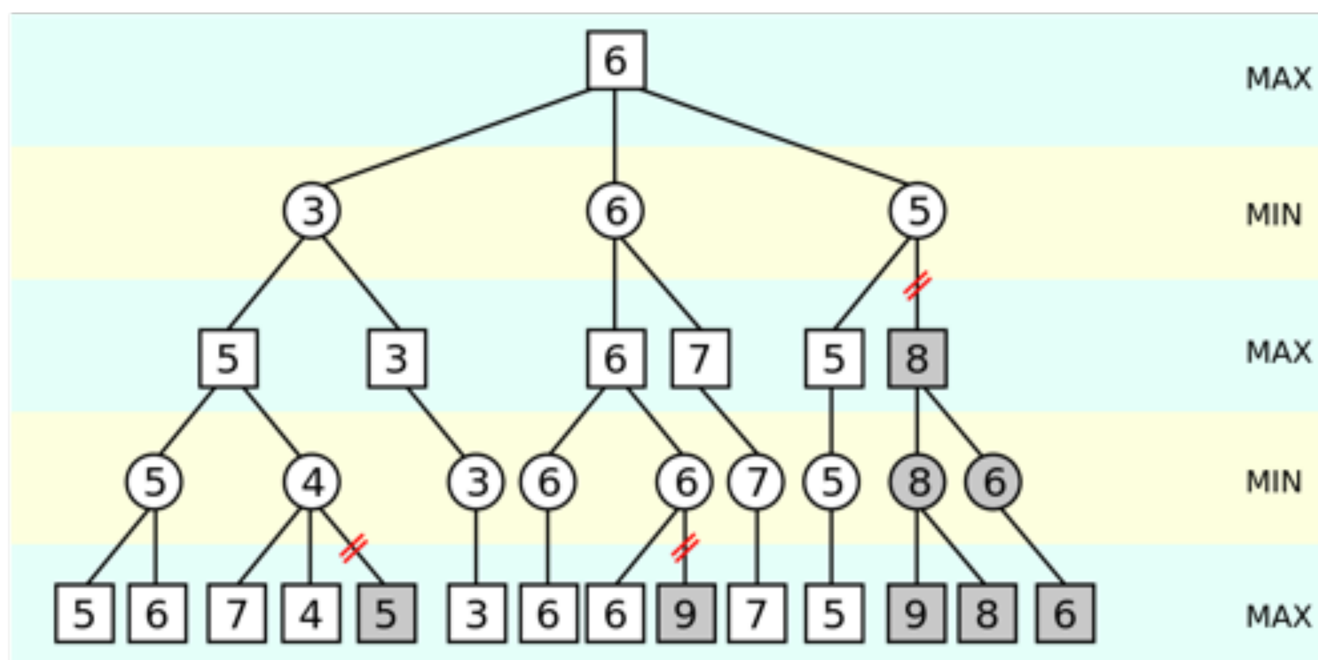


Combinatorial Explosion

Games consist of states & transitions.

Search trees grow very fast but ANNs can be used to learn useful heuristics!

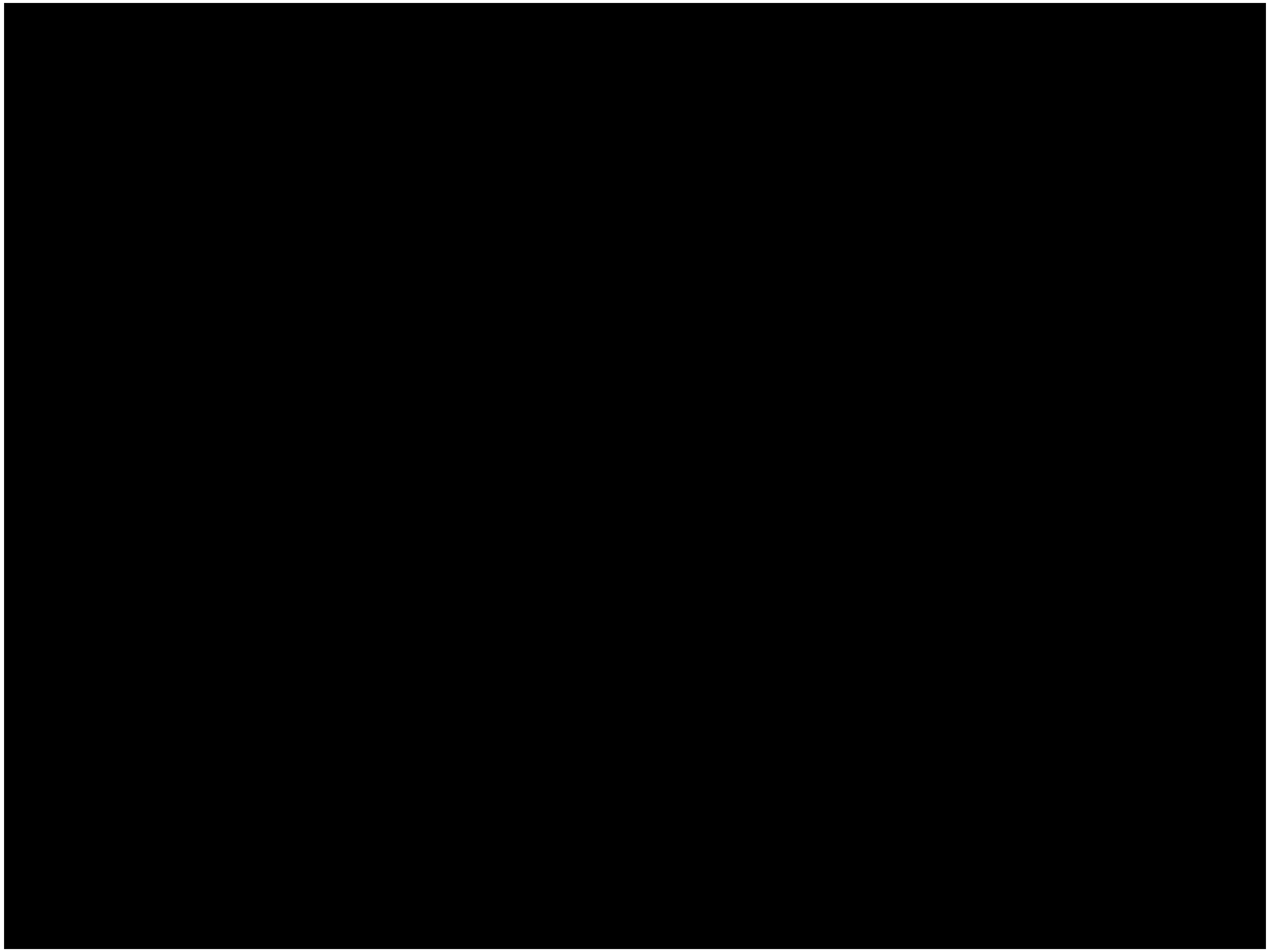
Alpha-beta pruning for **minimax** algorithm:



Reinforcement Learning

- Idea: Train a CNN to learn the goodness of actions given recent frames
- If environment cannot be controlled, adjust current estimate after next estimate (temporal difference learning)
- Favor sooner rewards w. greediness factor because no progress otherw.

Reinforcement Learning (CNN)



DEEPMIND

Google develops self-learning computer program
www.theguardian.com/technology/2015/feb/25/[…]

Outlook: Huge Responsibility

… in the near-term. For the long-term, see www.superintelligence.ch.

Train CNN

Deploy CNN

Campaign to Stop Killer Robots
www.stopkillerrobots.org

Discussion

www.superintelligence.ch

Kaspar Etter, MSc ETH in CS
kaspar.etter@gbs-schweiz.org