



# A short introduction to Machine Learning

Martino Sorbaro

Slides also by: Maria Astefanoaei

# Acknowledgements

- Workshop design: Benson Muite, Claudia Beleites, Martino Sorbaro (for ESCAPE 2021); Michaël Dell'Aiera
- Previously existing materials:
  - introductory slides partly by Maria Astefanoei (ITU Copenhagen)
  - machine learning and statistics class by David Kirkby (UC Irvine)  
<https://github.com/dkirkby/MachineLearningStatistics>

# Outline

- Today
  - Introduction ← *YOU ARE HERE*
  - Clustering and dimensionality reduction tutorial
  - Coffee break*
  - Model evaluation and optimization
  - Classification tutorial
- Tomorrow
  - More hands-on work
  - Deep learning

What is machine learning?

# How has computing changed?

**Data:** terabytes, petabytes, exabytes

**Speed:** gigahertz, teraflops, megabits/second

**Algorithms:** new methods that enable us to find new ways to create, manipulate and analyse data

All three aspects were needed for the current boom of machine learning usage.



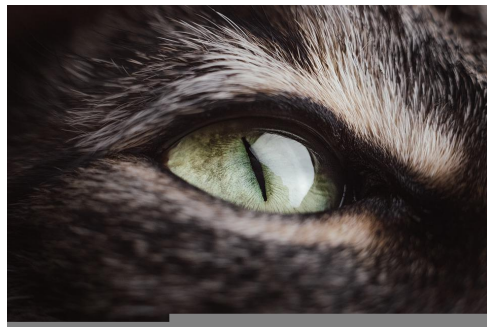
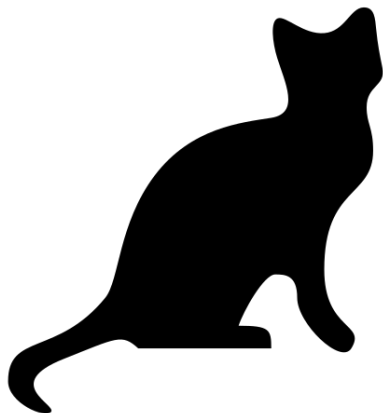
Data center (cybrain; iStock by Getty Images)

# Machine learning

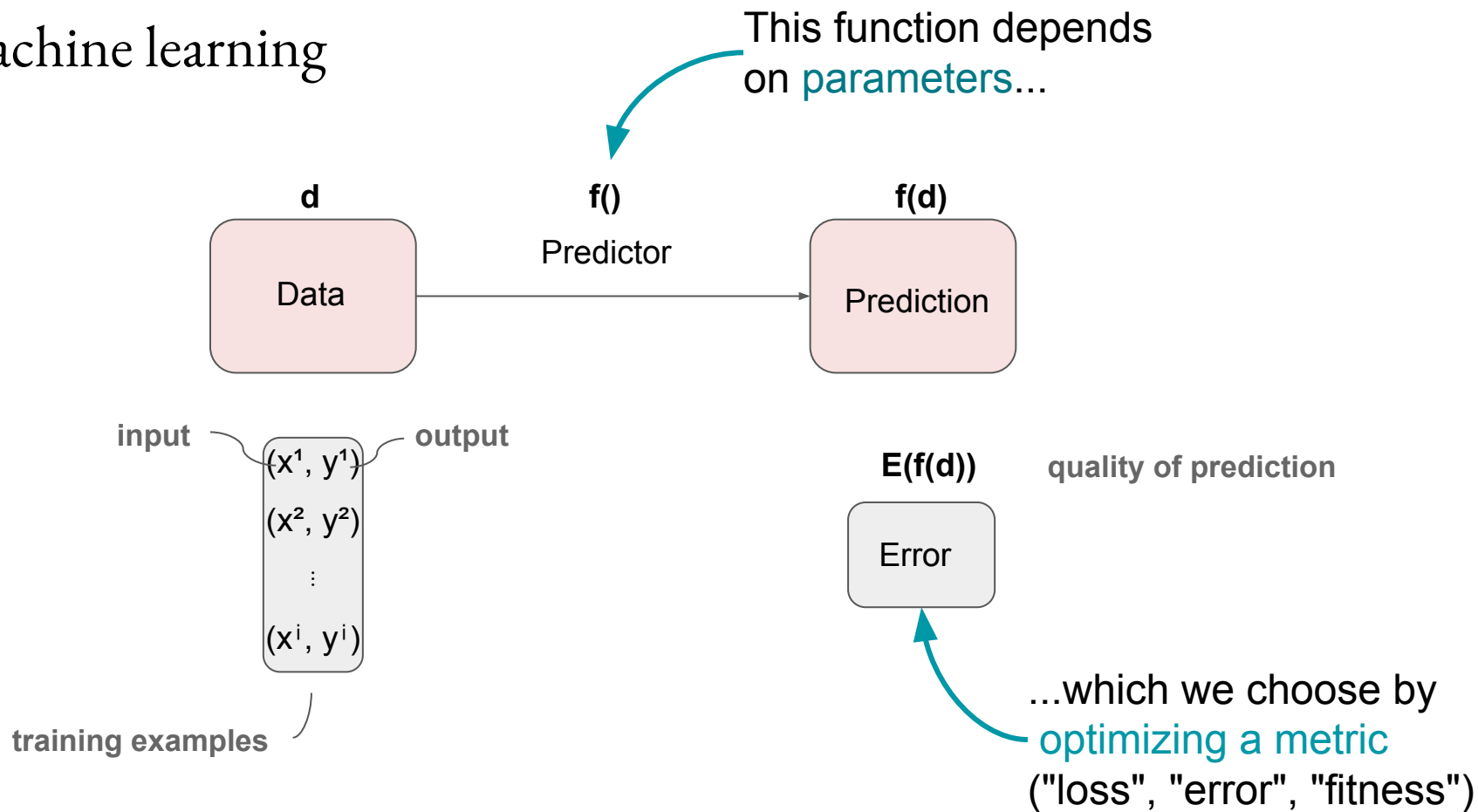
Arthur Samuel: "the field of study that gives computers the ability to learn without being explicitly programmed."

Tom Mitchell: "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  $E$ ."

# Example: object recognition in images



# Machine learning





# Types of learning

# Types of learning

- **Supervised** learning

- explicit examples of output, (input, desired output) pairs
- accuracy computed directly
- classification, regression

- **Unsupervised** learning

- letting the computer learn by itself
- model the underlying structure; does not require labeled data
- evaluation is indirect or qualitative
- clustering

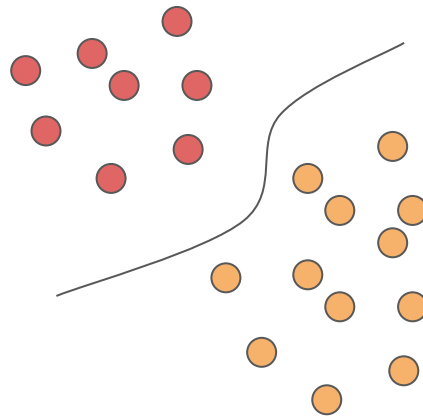
# Types of learning

- Supervised learning

- explicit examples of output, (input, desired output) pairs
- accuracy computed directly
- **classification**, regression

- Unsupervised learning

- letting the computer learn by itself
- model the underlying structure; does not require labeled data
- evaluation is indirect or qualitative
- clustering



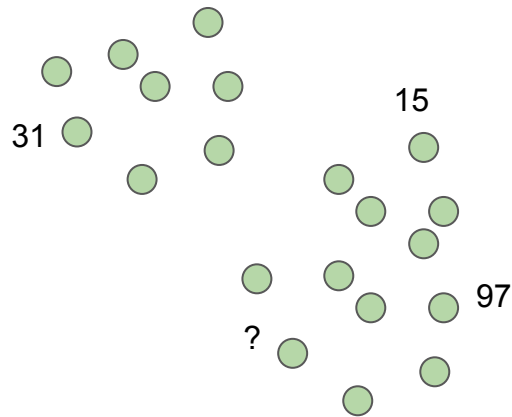
# Types of learning

- Supervised learning

- explicit examples of output, (input, desired output) pairs
- accuracy computed directly
- classification, regression

- Unsupervised learning

- letting the computer learn by itself
- model the underlying structure; does not require labeled data
- evaluation is indirect or qualitative
- clustering



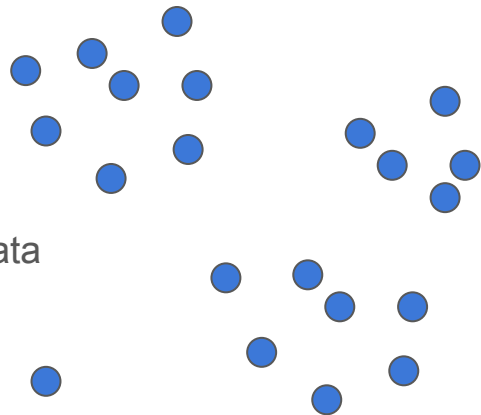
# Types of learning

- Supervised learning

- explicit examples of output, (input, desired output) pairs
- accuracy computed directly
- classification, regression

- Unsupervised learning

- letting the computer learn by itself
- model the underlying structure; does not require labeled data
- evaluation is indirect or qualitative
- clustering



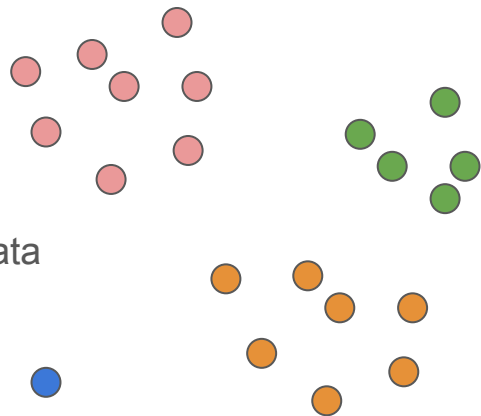
# Types of learning

- Supervised learning

- explicit examples of output, (input, desired output) pairs
- accuracy computed directly
- classification, regression

- Unsupervised learning

- letting the computer learn by itself
- model the underlying structure; does not require labeled data
- evaluation is indirect or qualitative
- clustering



## Supervised classification: training phase

*Training set*

ბ ა ბ ლ  
ც ვ ზ თ  
ი კ ლ მ

Georgian


ཁ ག ཀ ང  
ཅ ཆ ཇ ཉ  
ཏ ཐ ད ན

Tibetan

*labels*

## Supervised classification: test phase

Which alphabet does each of these belong to?

						<i>Test set</i>
Labels	T	G	T	G	G	
Your guess	T	T	T	G	G	Evaluation: accuracy 80%



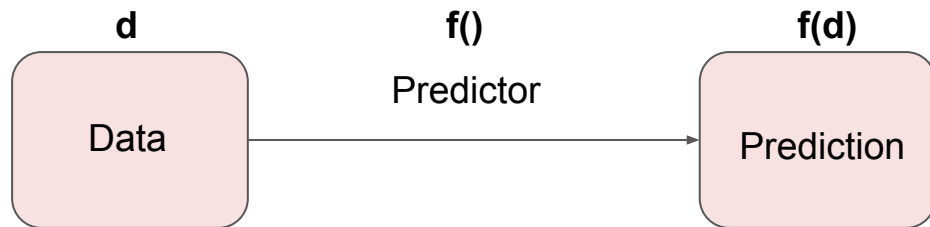
# Reinforcement learning

- An "agent" exploring an environment
- Learning by **trial and error** instead of having examples



# Types of data and representations

# Representing data mathematically



Attribute-value pairs:

- **categorical**: cat, dog, giraffe, elephant
  - mutually exclusive
  - encoded as numbers, but mathematical operations not meaningful
- **ordinal**: poor, satisfactory, good, excellent
  - encoded as numbers, meaningful to compare
- **numeric**: 0/1, -54, 32
  - meaningful to do mathematical operations
  - usually good practice to normalize values

## Example: handwritten digits

Bitmap images with a lot of variation in style,  
pressure, pen type



## Example: handwritten digits

Bitmap images with a lot of variation in style, pressure, pen type

Representation:

- each pixel is a separate attribute
- 400 attributes for a 20x20 bitmap
- real number (degree of blackness) or binary
- preprocessing such that a pixel in different images has the same meaning (eg. centering)



A 10x10 grid of handwritten digits from 0 to 9. Each row contains 10 variations of a single digit, showing a wide range of styles, pen pressures, and orientations. The digits are written in black ink on a white background.



## Example: labels in multiclass classification

<b>Class</b>	<b>Numerical label</b>	<b>One-hot encoding</b>
Cat	0	[1, 0, 0, 0]
Dog	1	[0, 1, 0, 0]
Horse	2	[0, 0, 1, 0]
Parrot	3	[0, 0, 0, 1]

# Example: music/speech

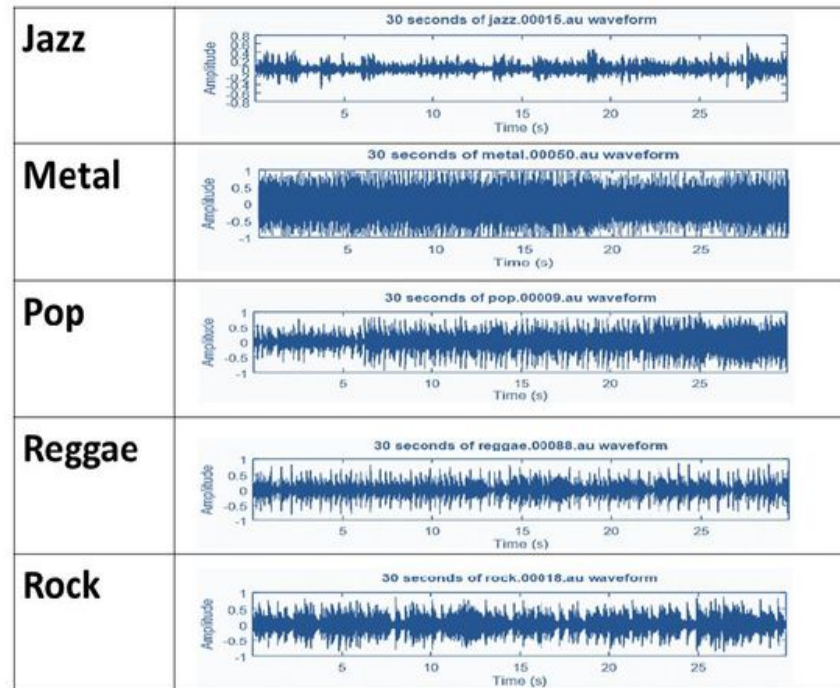
Time series — waveforms

Naïve representation:

- sample at regular intervals

Better representation:

- Fourier spectrum



# Summary: end-to-end ML

1. Define the task
2. Get the data
3. Choose the right representation
4. Get an overview
  - summary statistics (location, scale, shape, multivariate analysis)
  - visualizations
5. Prepare the data
  - data cleaning (missing values)
  - standardisation, outlier detection, transformations (polynomial, logarithmic)
6. Select a model
  - based on volume, complexity of data, assumptions about distributions and shape
7. Train and evaluate
  - on training set (cross-validation)
  - fine-tune model - optimize hyperparameters (grid search, randomized search)
8. Evaluate on test set
9. Launch, monitor, maintain



# Reproducibility in machine learning

# Reproducibility in ML

Computational reproducibility can be defined as the process of obtaining consistent results using the same input data, computational methods, and conditions of analysis.

Sources of stochasticity:

- Random initialization of layer weights
- Shuffling of datasets
- Changes in machine learning frameworks
- Noisy hidden layers

Tools for reproducible ML:

- keeping track of model trainings (hyperparameters, etc)
- dataset versioning
- model sharing and management tools
- version control (git)
- model deployment tools

E.g. *wandb.ai*, *guild.ai*, ...

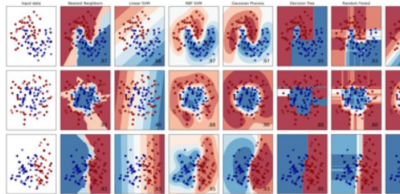
# Introduction to scikit-learn

## Classification

Identifying which category an object belongs to.

**Applications:** Spam detection, image recognition.

**Algorithms:** SVM, nearest neighbors, random forest, and more...



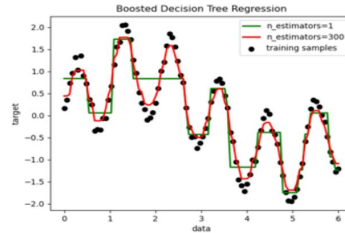
Examples

## Regression

Predicting a continuous-valued attribute associated with an object.

**Applications:** Drug response, Stock prices.

**Algorithms:** SVR, nearest neighbors, random forest, and more...



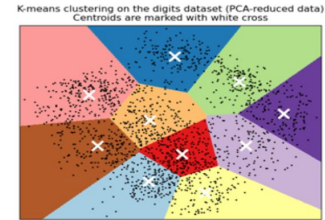
Examples

## Clustering

Automatic grouping of similar objects into sets.

**Applications:** Customer segmentation, Grouping experiment outcomes

**Algorithms:** k-Means, spectral clustering, mean-shift, and more...



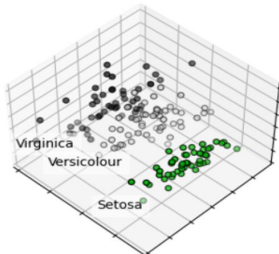
Examples

## Dimensionality reduction

Reducing the number of random variables to consider.

**Applications:** Visualization, Increased efficiency

**Algorithms:** k-Means, feature selection, non-negative matrix factorization, and more...

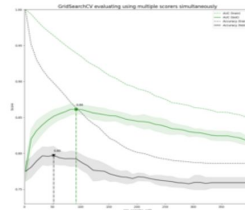


## Model selection

Comparing, validating and choosing parameters and models.

**Applications:** Improved accuracy via parameter tuning

**Algorithms:** grid search, cross validation, metrics, and more...

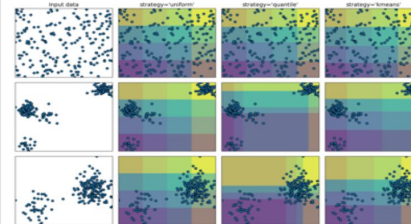


## Preprocessing

Feature extraction and normalization.

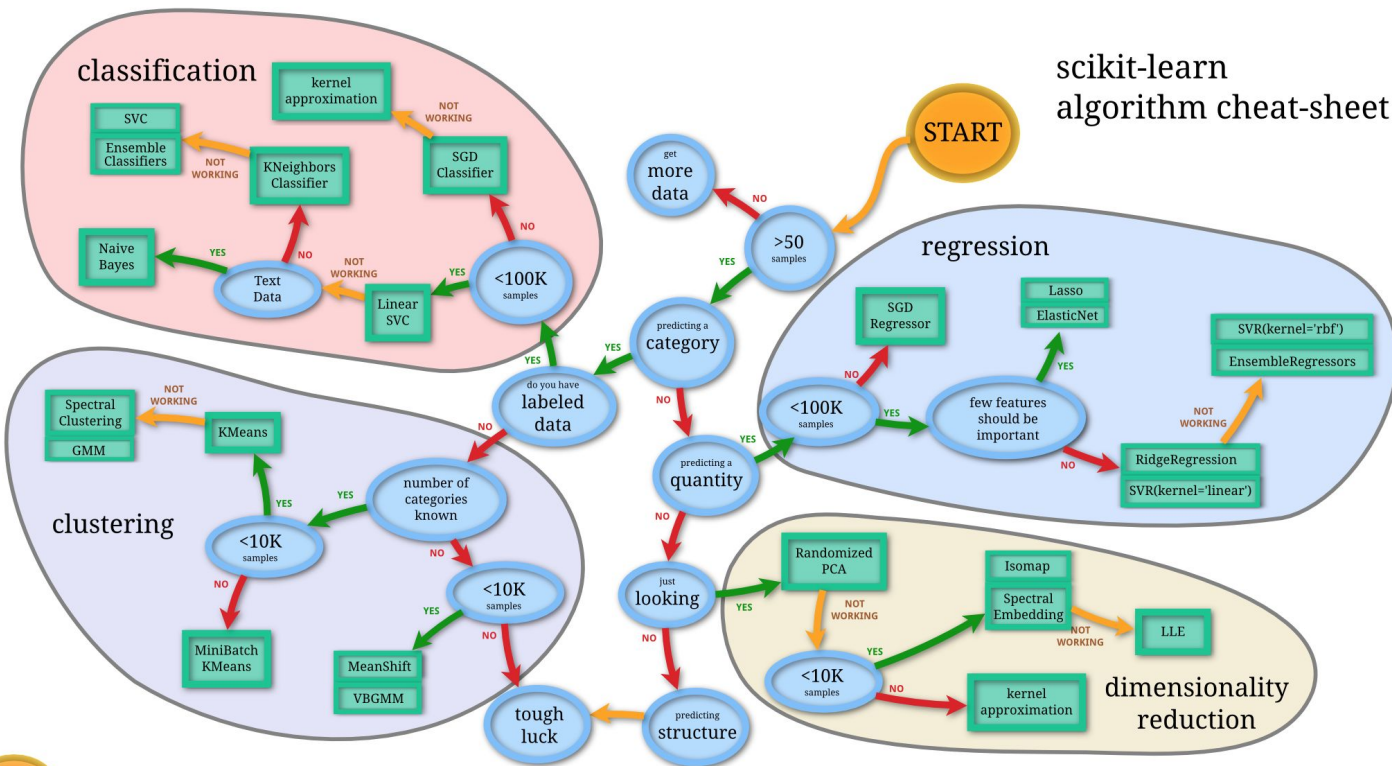
**Applications:** Transforming input data such as text for use with machine learning algorithms.

**Algorithms:** preprocessing, feature extraction, and more...



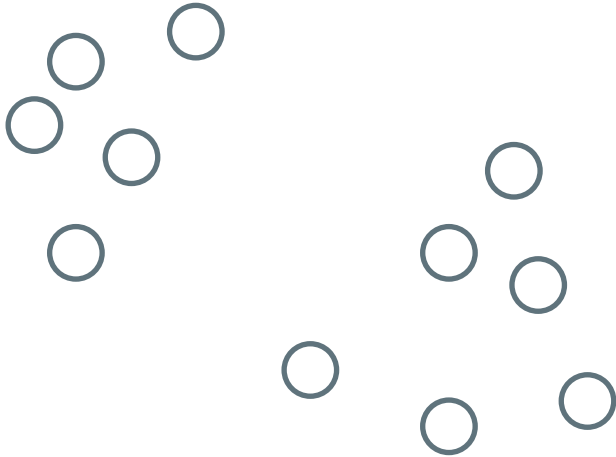
# Choosing the right model

scikit-learn  
algorithm cheat-sheet

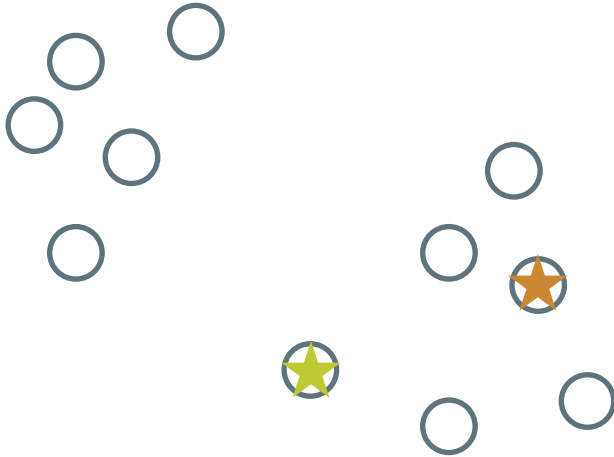


# Clustering

# How k-means works



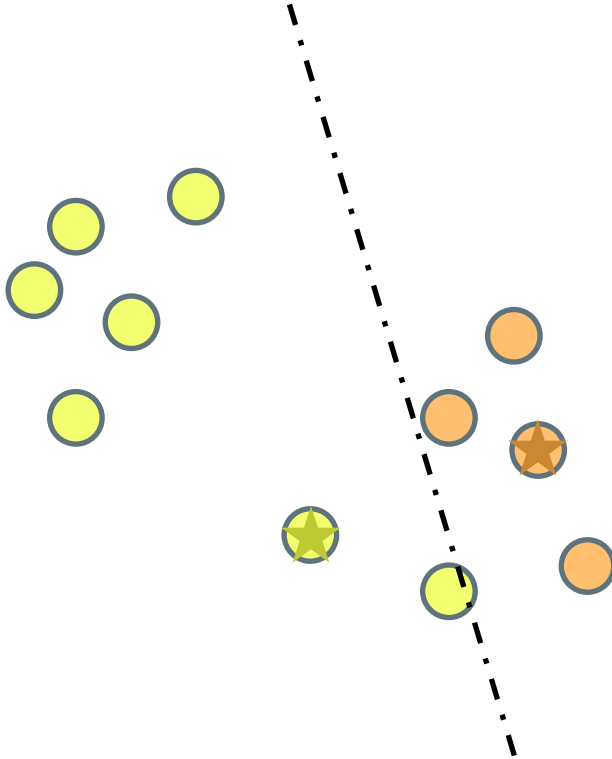
# How k-means works



Place  $k$  centroids in  
correspondence with  
 $k$  random data points

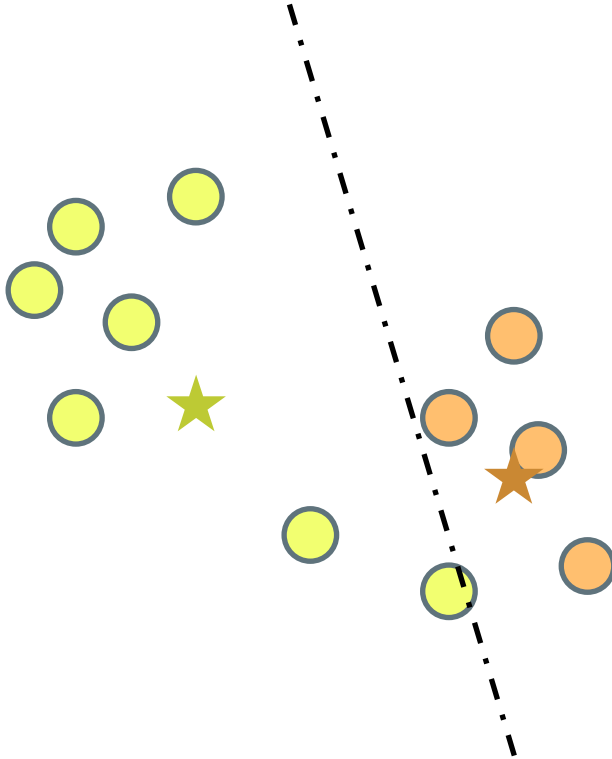


# How k-means works



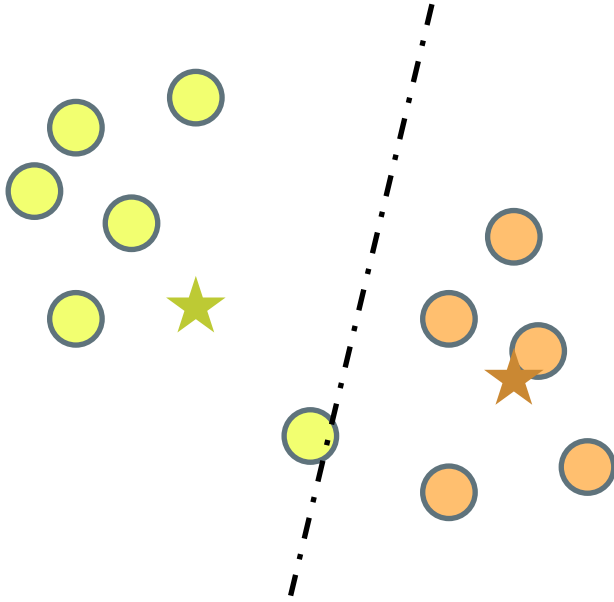
Assign each point to  
the nearest centroid

# How k-means works



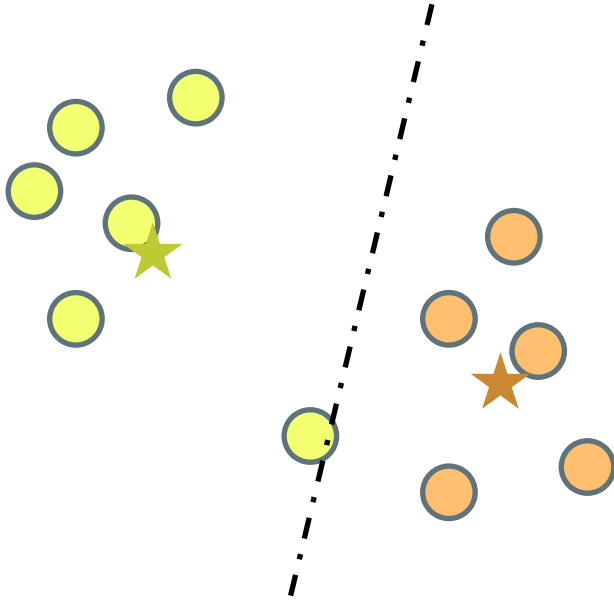
Move centroid to the center of mass of the datapoints assigned to it

# How k-means works



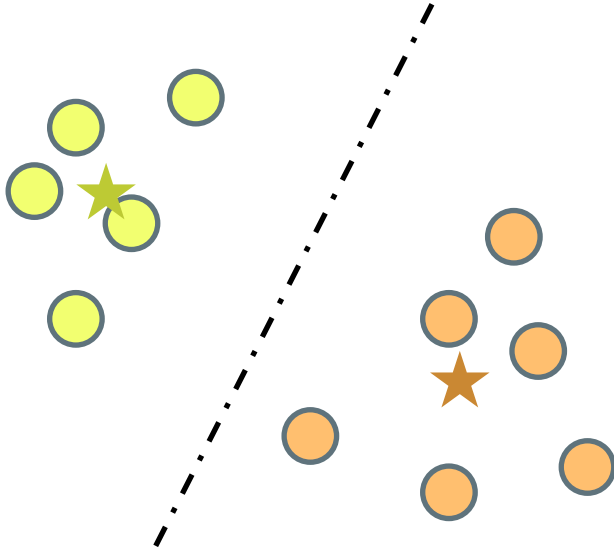
Repeat assignment  
to nearest centroids

# How k-means works



Repeat moving  
centroids to center of  
mass

# How k-means works

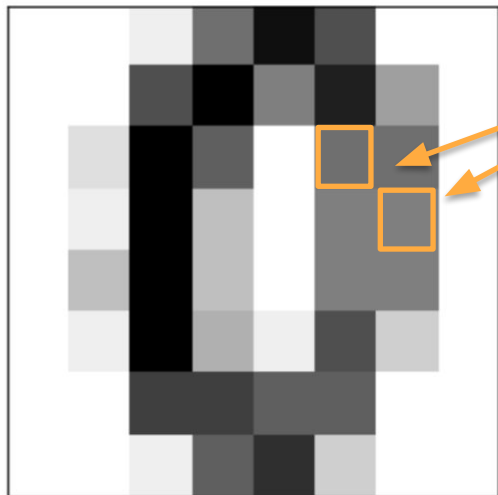


The algorithm  
converges

# Dimensionality Reduction

# How does PCA work?

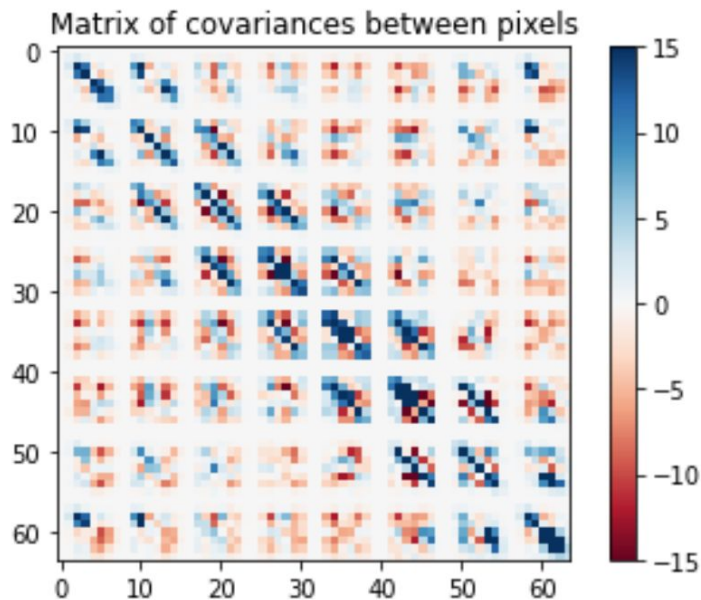
In all "natural" datasets, feature variables are normally correlated to each other.



Across the dataset, these two pixels are likely to be of similar colour.

# How does PCA work?

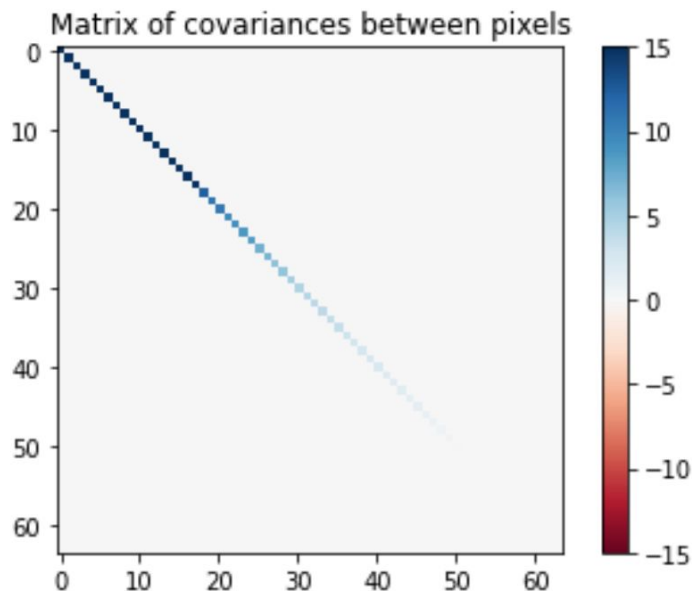
The matrix of covariances is symmetric and positive-definite.



We can therefore **diagonalise** it...



# How does PCA work?

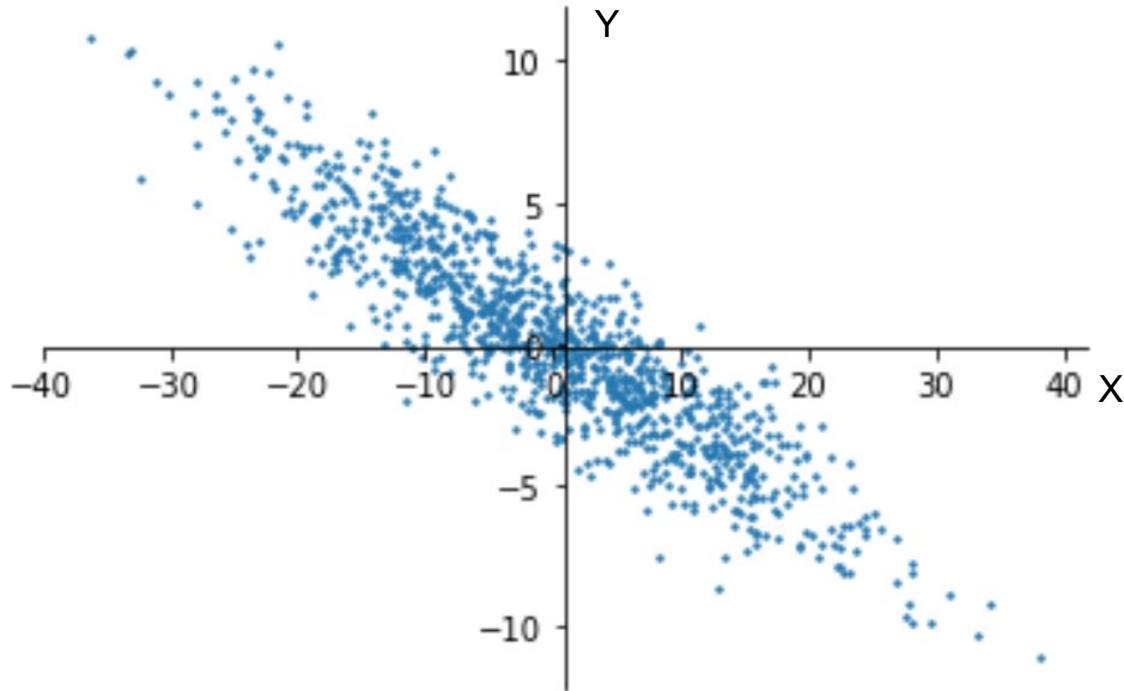


And move to a space of **statistically independent** feature variables.

The dimensions corresponding to **larger eigenvalues** have larger variance and thus describe the variability within the dataset better.

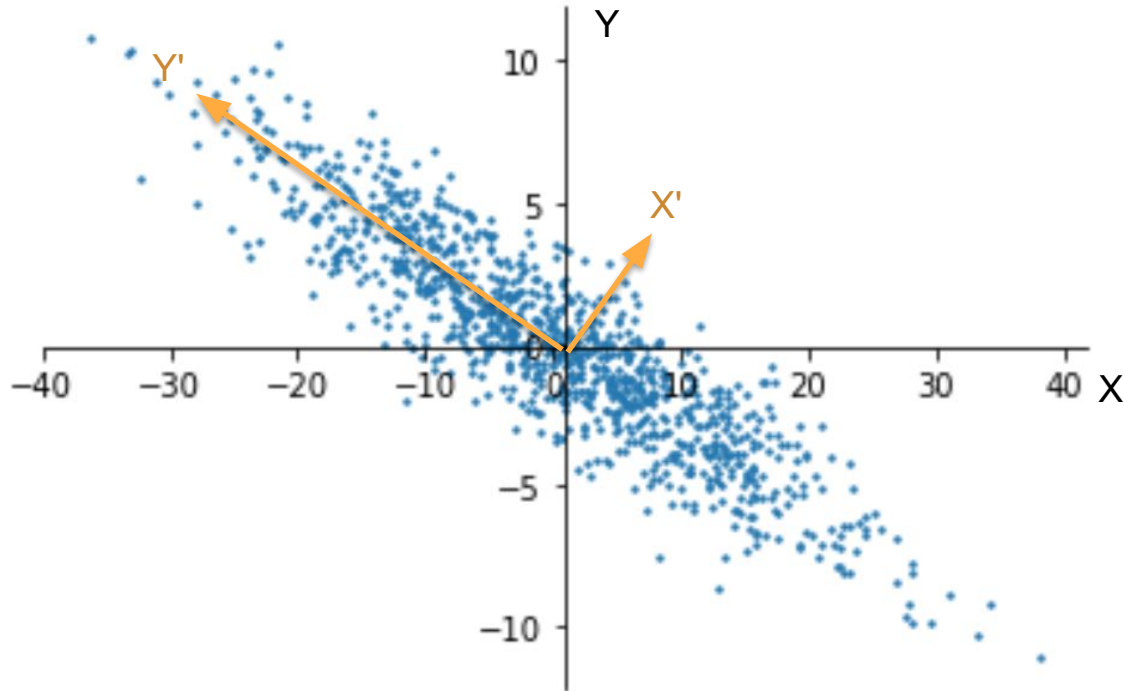
# Wait... what?

Suppose we have data described by **2** variables X and Y:



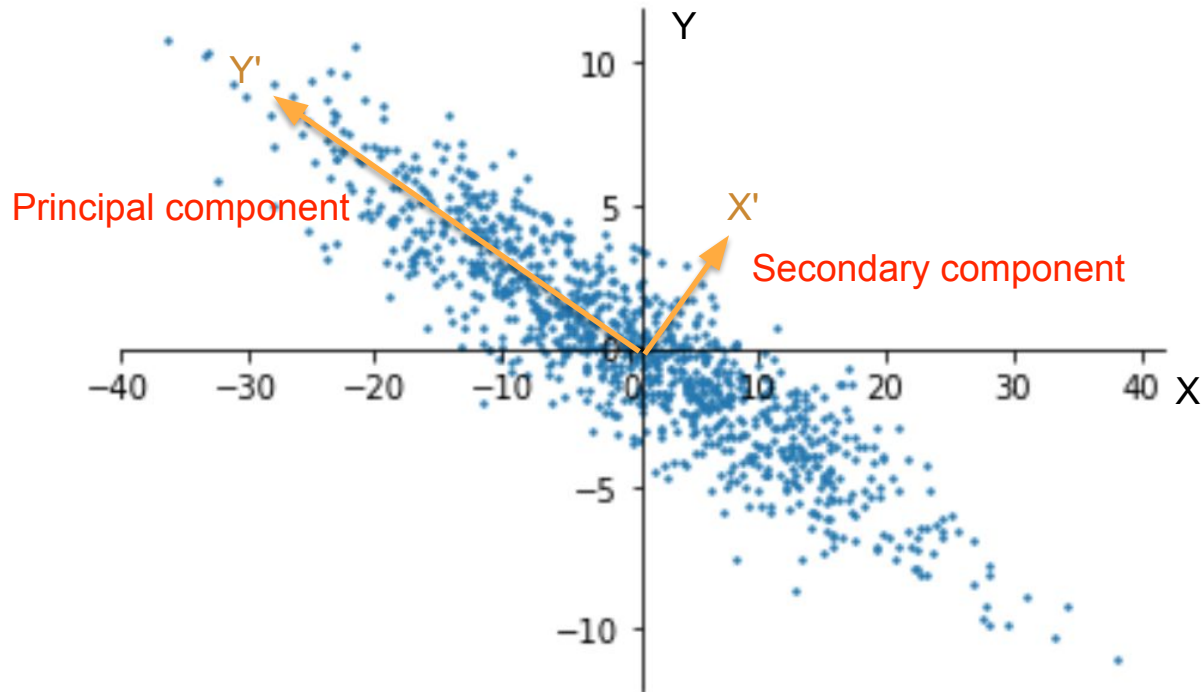
# Wait... what?

PCA finds two new variables that are **uncorrelated**:



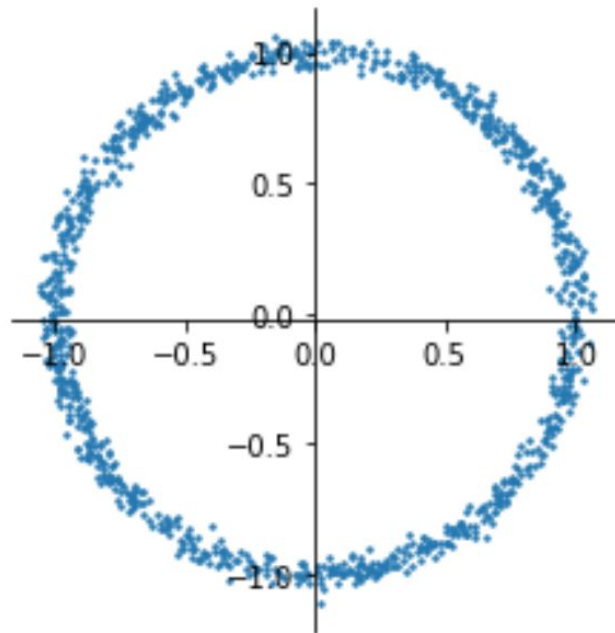
# Wait... what?

... then selects the ones with **highest variance**, which capture the data variability:



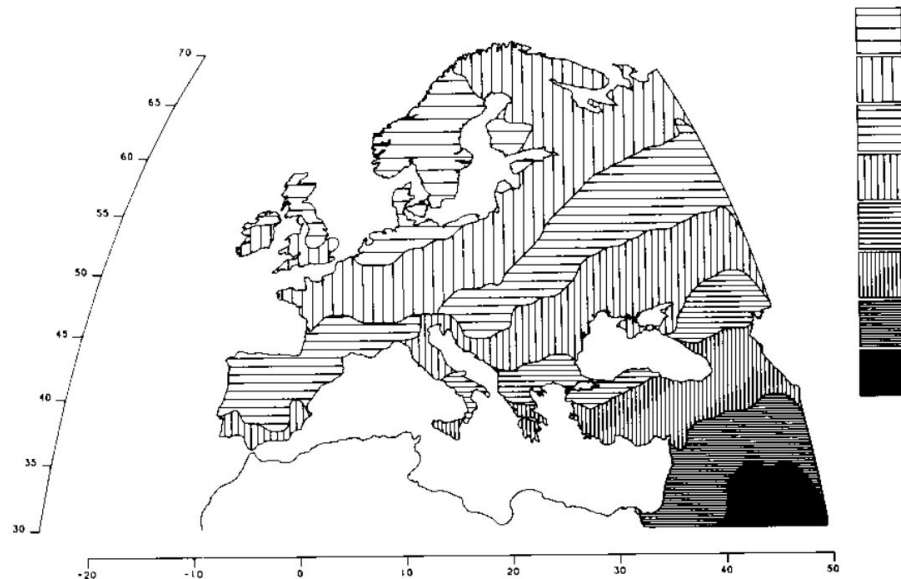
# Failure case

- Data is not linearly correlated
- $\theta$  is a "principal" component,  $r$  a "secondary" component
- PCA will never find them.



# Cool uses of dimensionality reduction

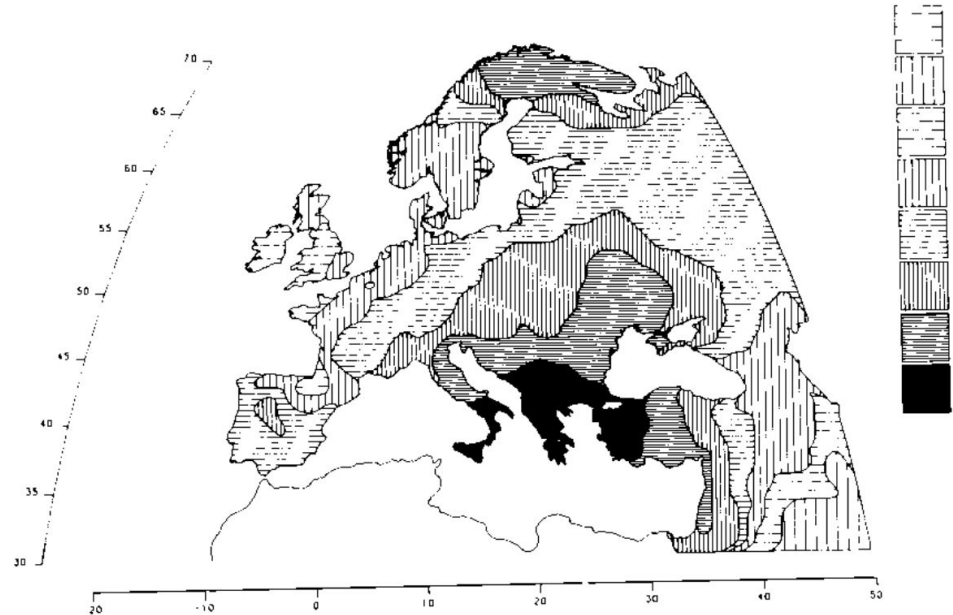
- DNA dimensionality is huge
- First PC of genetic variability in Europe: spread of agriculture



Cavalli Sforza, 1997

# Cool uses of dimensionality reduction

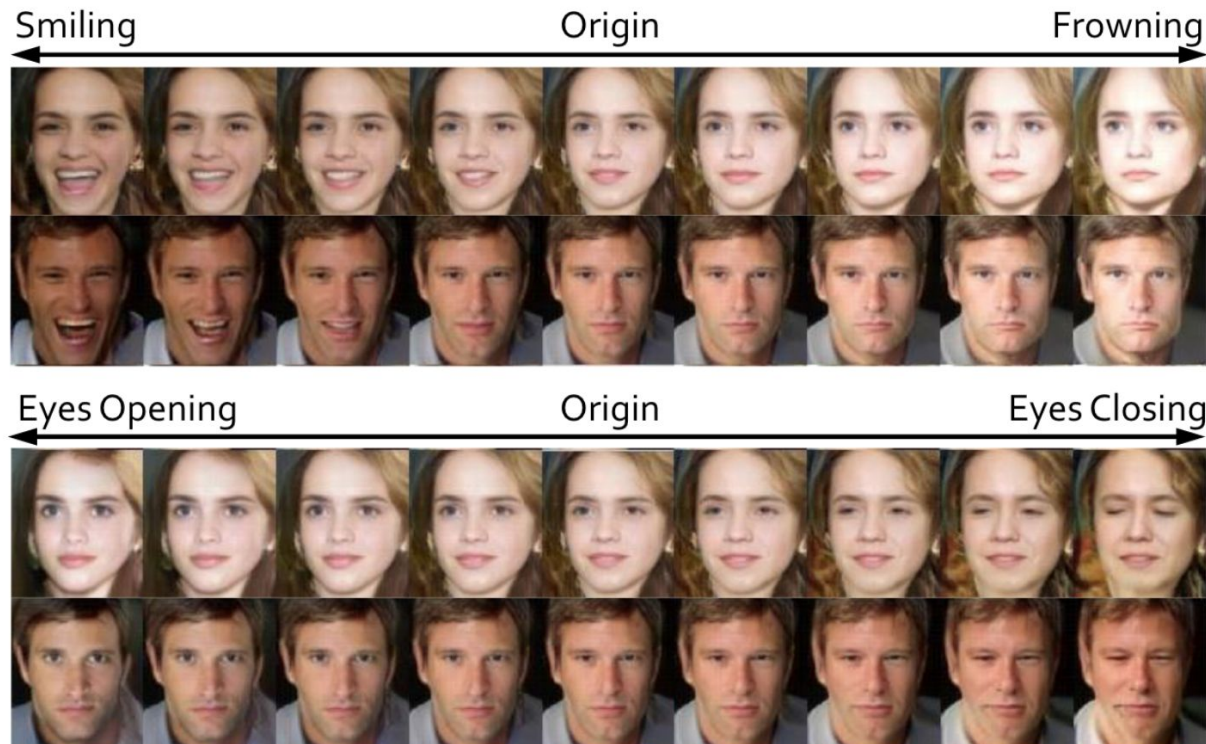
- DNA dimensionality is huge
- Third PC of genetic variability in Europe: Greek colonisation



Cavalli Sforza, 1997

# Cool uses of dimensionality reduction

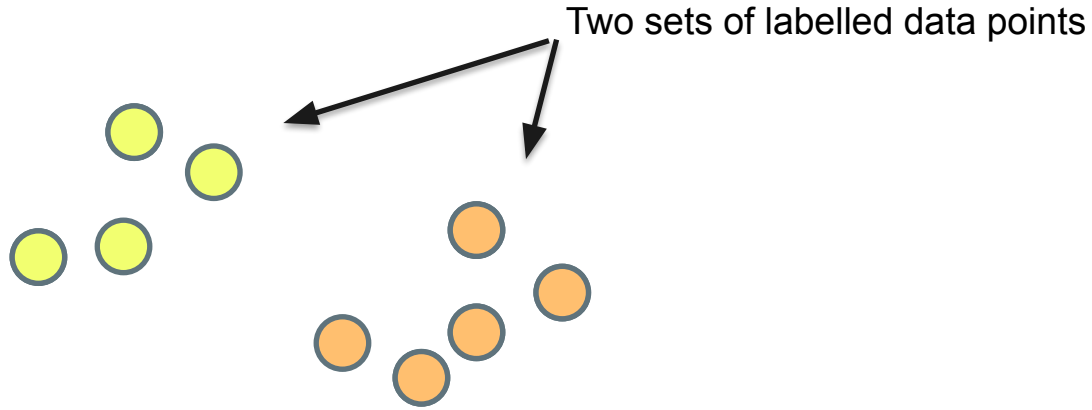
Finding semantically  
meaningful dimensions  
and interpolating



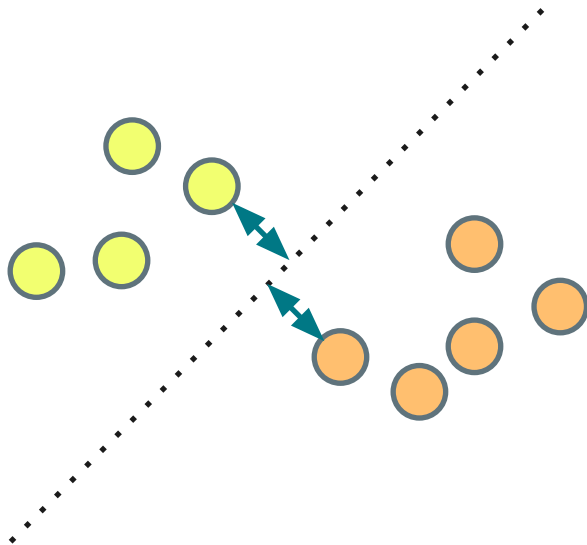


# Support Vector Machines

# Intuitive understanding of SVM



# Intuitive understanding of SVM

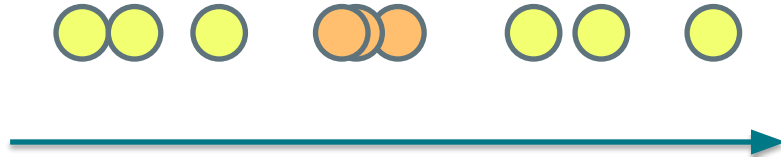


## Linear SVM

Find the hyperplane that separates the two sets by the largest **margin** between the "support vectors".

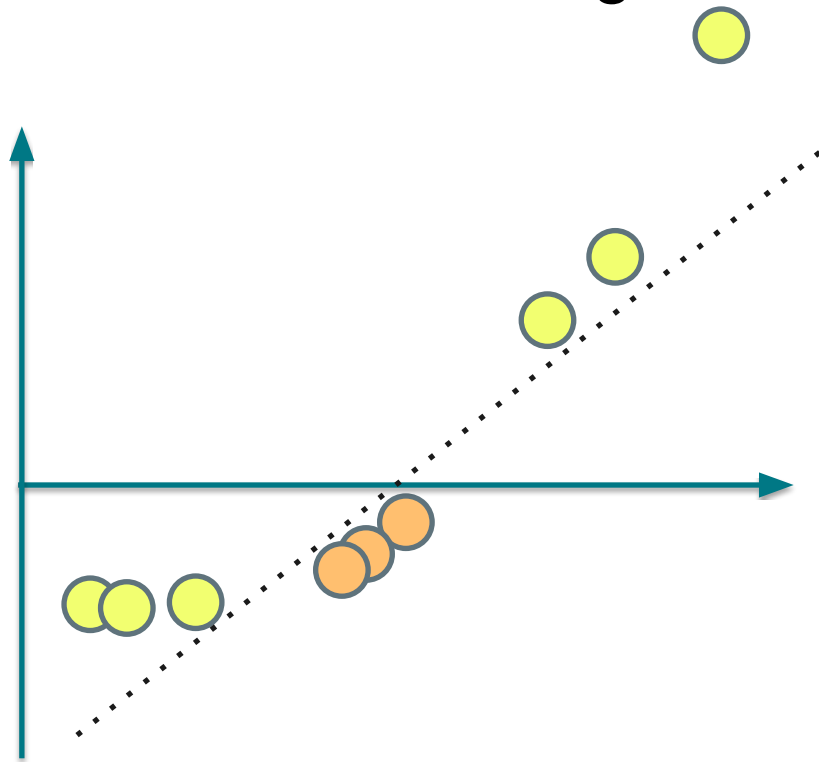
This is the line that separates the classes.

# Intuitive understanding of SVM



If the data is **not linearly separable**...

# Intuitive understanding of SVM



If the data is **not linearly separable**...

we project to more dimensions which are nonlinear functions of the original ones, so that we can now separate linearly.

The actual mathematical implementation is more efficient!

