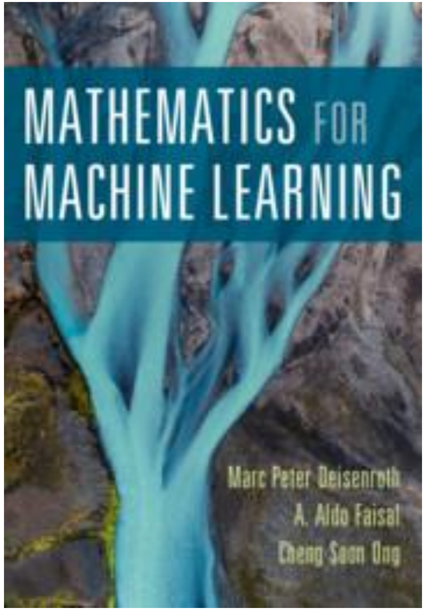
Living in a high dimensional world

Hello there

- Michael Garcia Ortiz – michael.garcia-ortiz@city.ac.uk
    - New Lecturer
    - Former Research Scientist in Robotics company
    - Former consultant in ML (finance, startups)

- Interests: Robotics
  Artificial General Intelligence
  (I have UG – PG project ideas)

Goal: give you an overview of what you can do with data.

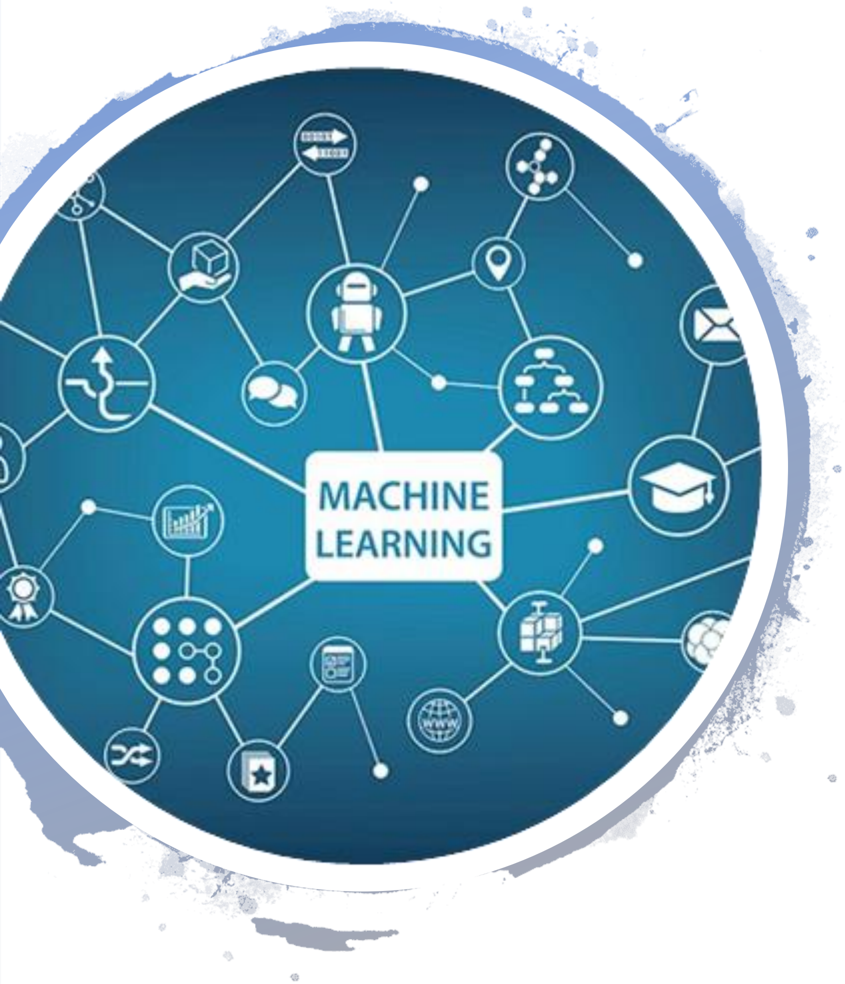


What you need for this module

- Understanding how things work:
  Mathematics https://mml-book.com
- Making things work:
  git, python, sklearn, matplotlib with **Jupyter Lab**
- Having a feeling about what might work:
  Practice, trial and error, gaining insights and intuition

Do the thing, and you shall have the power.

(insert thunder)



What we will see together

- Introduction: Living in a high dimensional world
    - The effect of dimensionality
    - Solving problems with data
    - Feature engineering
    - Projection and visualization
- Finding Intrinsic Structure in Data with Unsupervised Learning
- Predicting with Supervised Learning

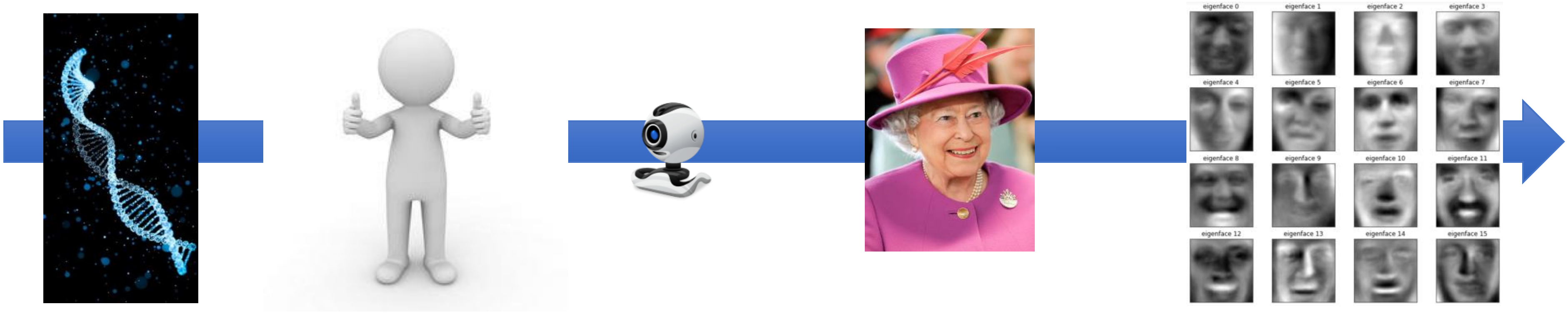
What is high dimensionality?

- High dimensional data is ubiquitous
- High dimension is a double-edge sword
- Transforming raw data
- Projecting and visualizing your dataset

High dimensionality is everywhere

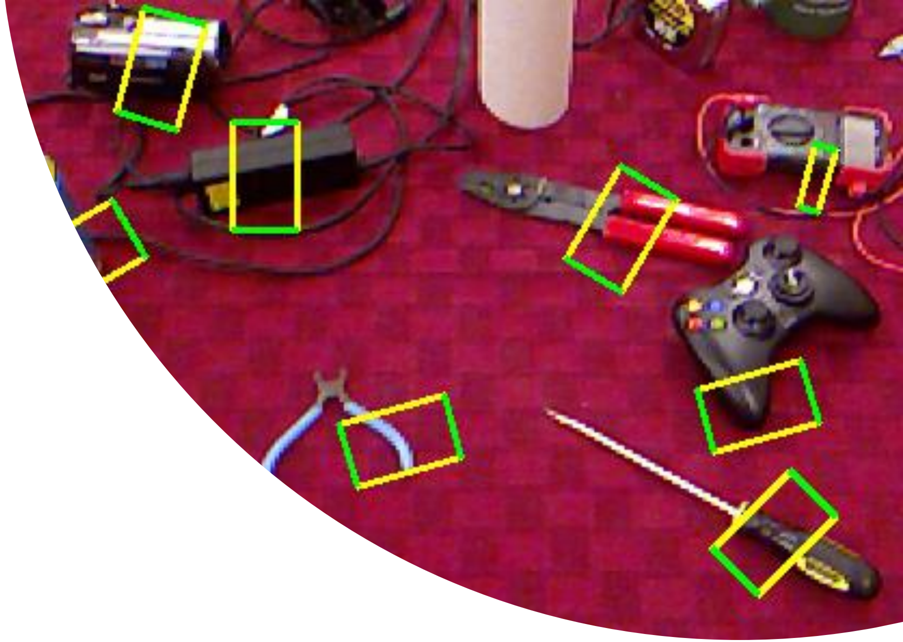
- Facebook: 350 M pictures
  every day (in 2013)
- Starbucks: 90 M transactions
  every week (in 2018)
- Universities: Course Signals at Purdue
- Big data in governments, Finance, Health ...

What is dimensionality?



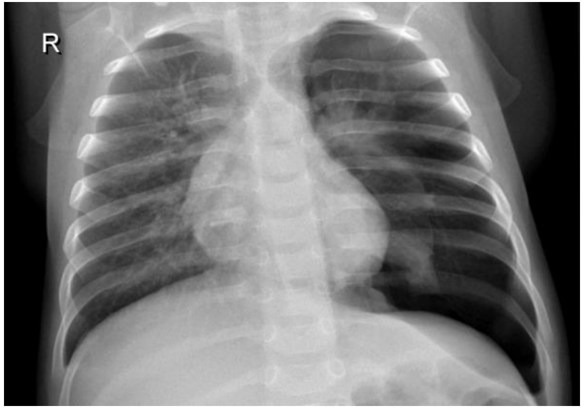
DNA? Human? VGA: 10^6 EigenFace: 100

Dimensionality always **depends on your level of description**.
Often, you don't control the dimensionality of the data you obtain.



- Data you observe is the result of physical process:
    - It is the projection of this process on your sensor (image, sound, csv file)
- Example: Relevant objects, relevant object features for grasping
- Assumption: data can be manipulated, compressed, abstracted, vectorized. Made more explicit, easier to handle

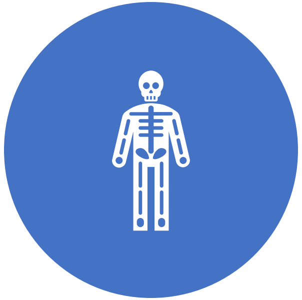
Intrinsic dimension



Data dimensionality? Examples from Kaggle

- Los Angeles Parking Citations : 19
  - Make, color
  - Location
- Chest X-Ray Images (Pneumonia) : ~ 1400x1000
- New York City Airbnb Open Data : 16
  - Location
  - Room type
  - Price
  - Name

Exploitable dimensionality?



OFTEN NEED TO **MANIPULATE DATA**: BINNING, ONE-HOT ENCODING, FEATURE EXTRACTION, COMPRESSION, ABSTRACTION, REPRESENTATION...

TABULAR DATA: ACTUAL DIMENSION IS PROBABLY HIGHER THAN THE DIMENSION OF YOUR SPREADSHEET

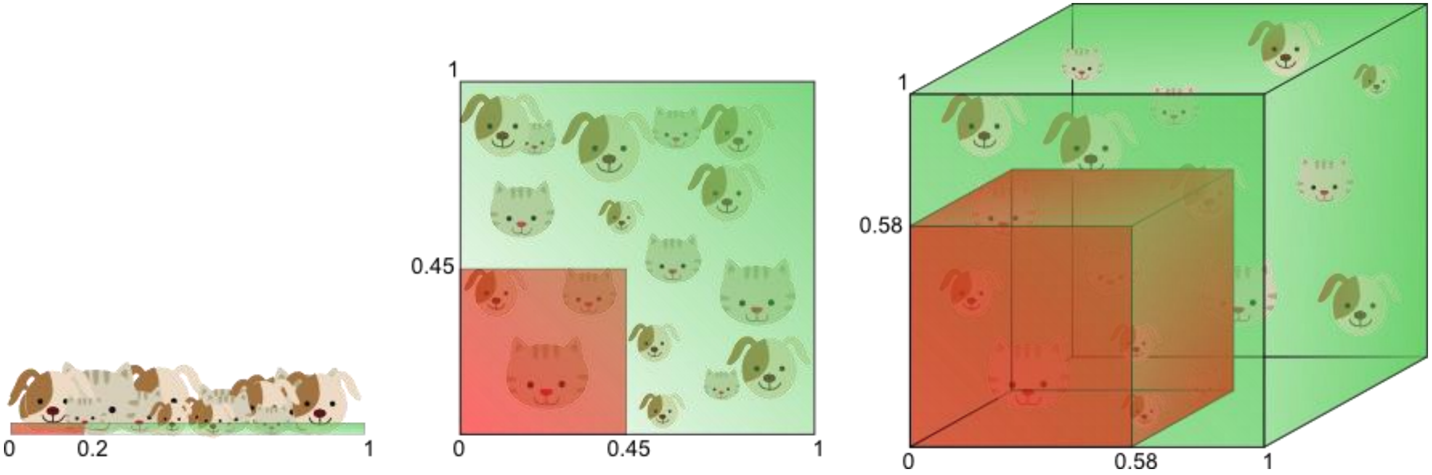
IMAGE DATA: ACTUAL DIMENSION IS PROBABLY LOWER THAN THE DIMENSION OF YOUR IMAGE

Why is high dimensionality a problem?

- **Visualization:** How do you communicate your findings? Top management, non-technical people: We need very **communicative** visualization.
- **Interpretation:** Hypothesis about patterns in data you observe. How do you **confirm your insights?**
- **Difficult to process**: memory, compute. Multiplied by the number of models you want to try out, size of database, ...

The curse of dimensionality

- Higher dimensionality requires more data



High dimensionality is very beneficial

- More dimensions to express your problem

- Example: describe an animal!
  - Number of legs
  - Carnivorous?
  - Predator?
  - Size?
  - Size of ears?

- It is far easier when the number of informative features is high.

- But, high dimensionality is not always equal to more information

How to describe a problem?

- It is the role of a Data Scientist to decide how to describe a problem
- Extract relevant information
- Visualize the data
- Find patterns to solve problems

Approaches of Datascience

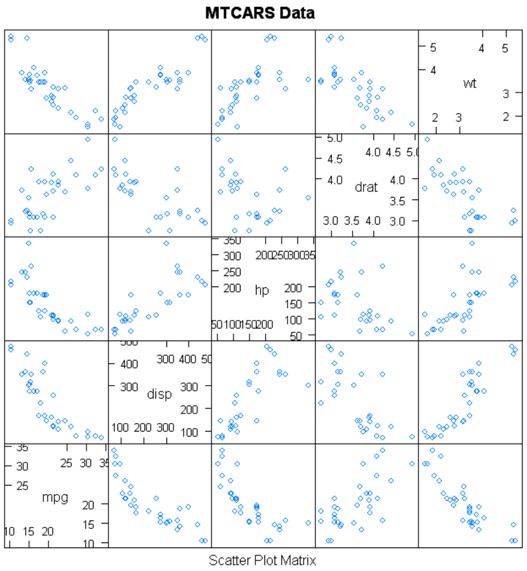
- Once we have data, what do we do with it?
- Visualization
- Compression
- Find patterns
- Solve problems

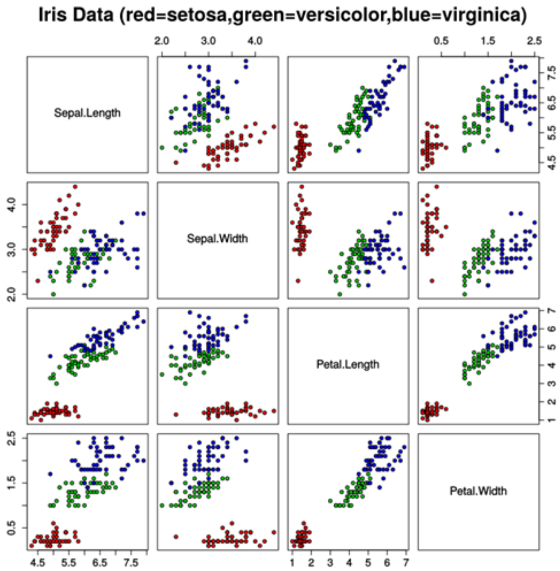
Dimensionality Reduction

- Storage and transfer
- Describing your problem correctly
- Abstraction to improve machine learning
- Other challenges: visualization, anonymity, transfer, ...

Visualization

- Why is it hard in high dimension?



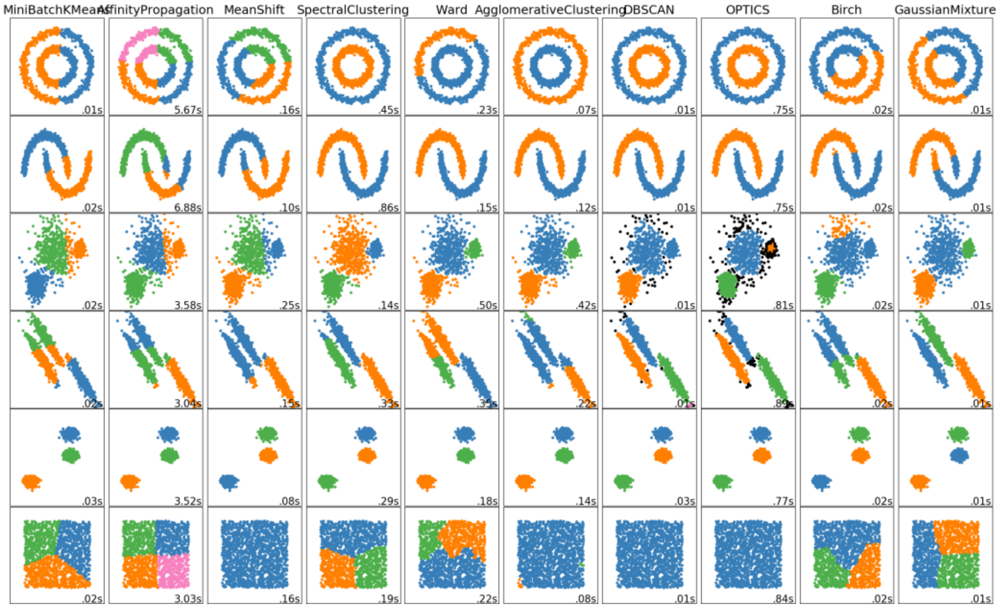


Find patterns

- Relations between dimensions of data
- Correlation, covariance, prediction
- Clusters, similarity / distance between data points
- Can data be used to predict things?

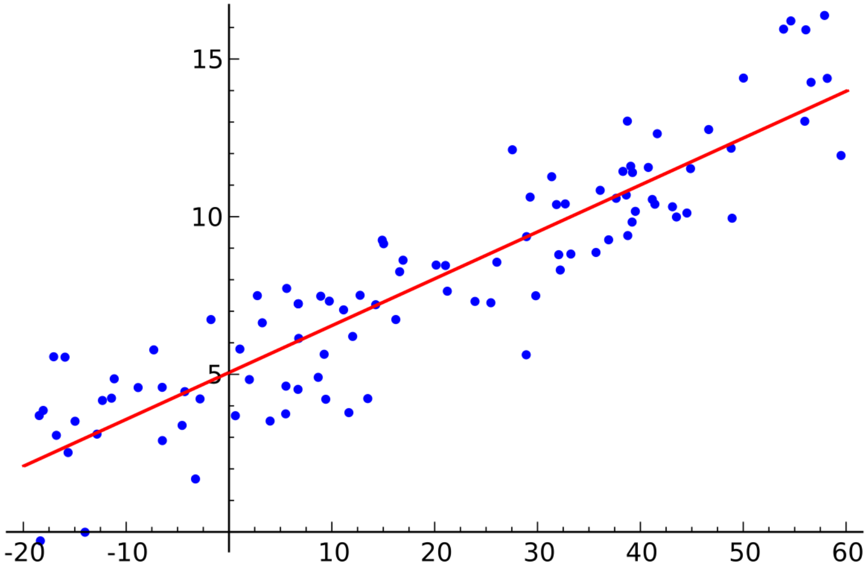
Unsupervised Learning: Clustering

- Grouping data points together by similarity
- What is similarity?
- How do you group?
- Assumptions?

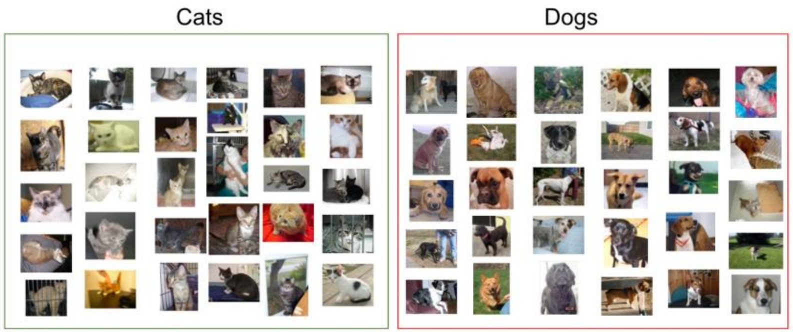


Supervised Learning: Regression

- Classic example: linear regression
- Input data, model, target
- But can be much more complex!



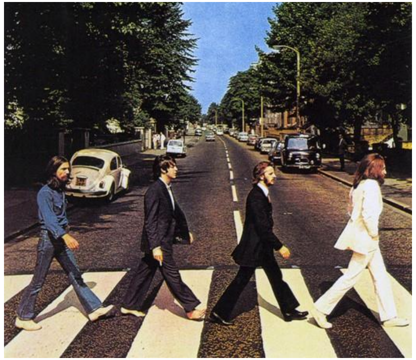
Supervised Learning: Classification



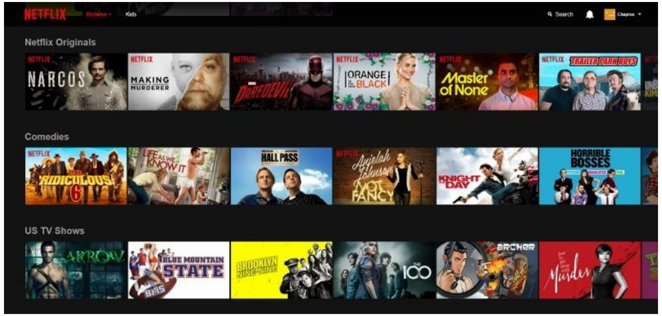
Sample of cats & dogs images from Kaggle Dataset

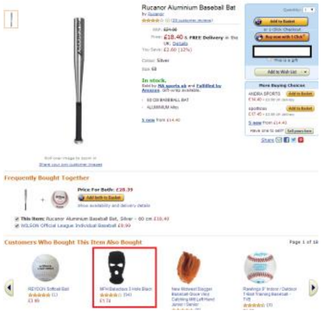
FRANCE RUGBY

England Rugby



Anomaly detection, recommender systems, ...





You need to represent your data with numbers

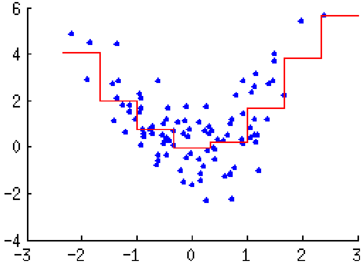
- Create features
- Input data is a curated list of features
- Algorithms will perform analysis based on the features you choose.

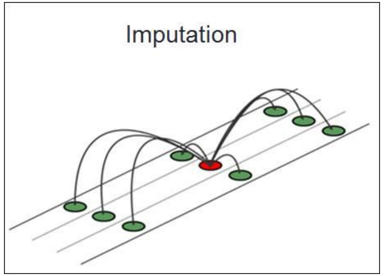
Feature engineering

- Manage the dimensionality of the data
- Bring your knowledge and intuitions to the problem:
  - Facilitates Machine Learning
  - Knowledge comes with bias
- Dimensionality reduction through compression
- Feature selection

Basic data preparation

- Continuous data: binning
- Categorical data: one-hot encoding
- Ordinal data: numerical
- Missing data: imputation
- Temporal data?
- Scaling, normalization, whitening...





Label Encoding

Food Name	Categorical #	Calories
Apple	1	95
Chicken	2	231
Broccoli	3	50

→

One Hot Encoding

Apple	Chicken	Broccoli	Calories
1	0	0	95
0	1	0	231
0	0	1	50

scikit learn

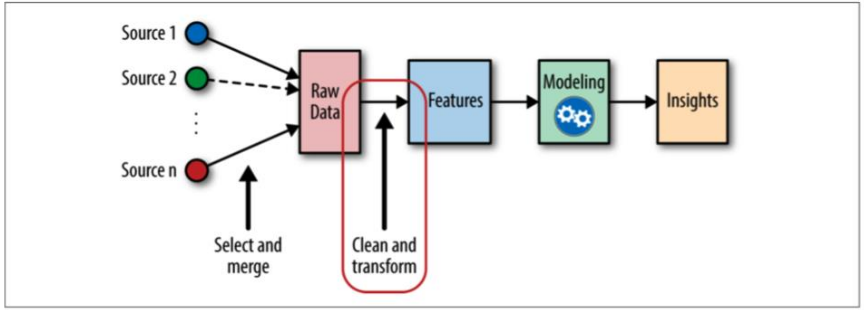
Why creating features?

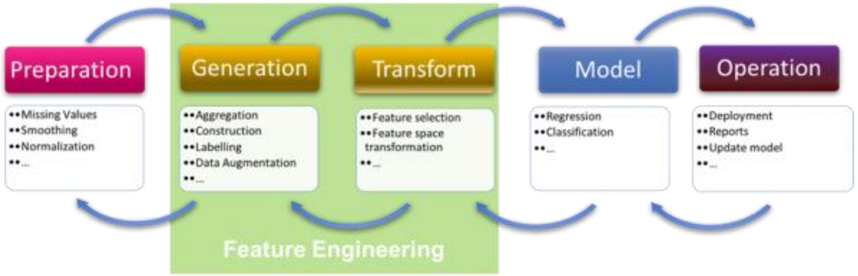
Importance of having indicators for change of trend, price drop, sudden price increase. Automatic alert? Autonomous trading?

Technical reasons

- Some ML algorithms depend on dimensionality of input data.
- Dimensionality too low: not expressive enough
- Dimensionality too high: can't compute anymore
- Benefit from more descriptive, easily separable features

Feature engineering



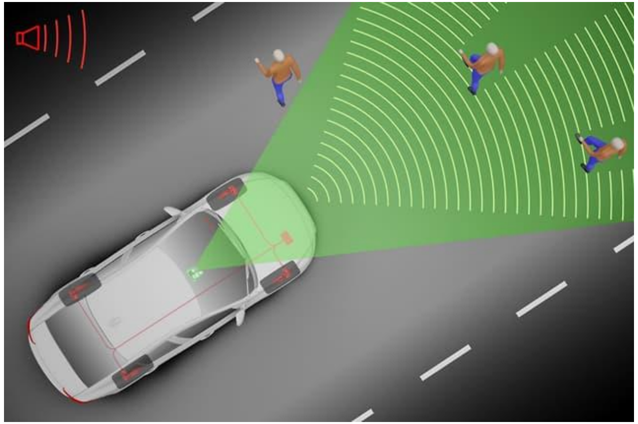


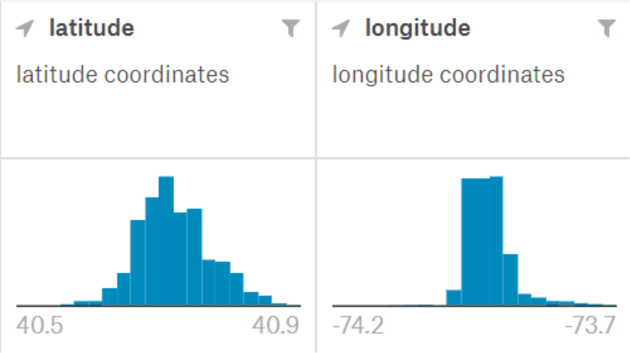
Bring domain knowledge to the table

- GPS coordinates on AirBnB listings: 2 real numbers
  - Why not distance to closest subway station? 1 number
    Distance to landmarks? N numbers

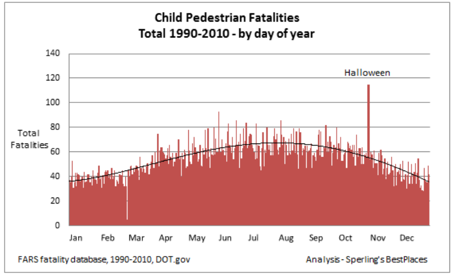
- Sentiment analysis in political tweets
  - Political jargon can be very country-specific (Ex: US / UK)

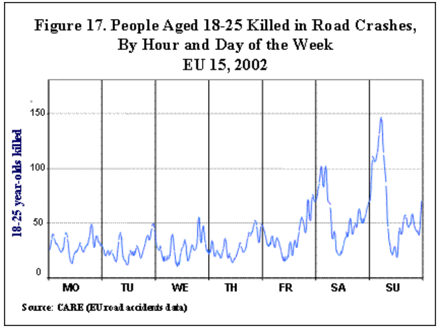
- Predicting driver behavior:
  - distance to the next car
  - time to collision

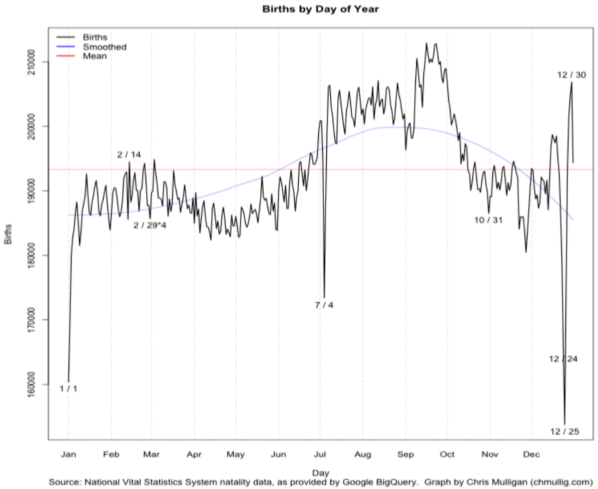


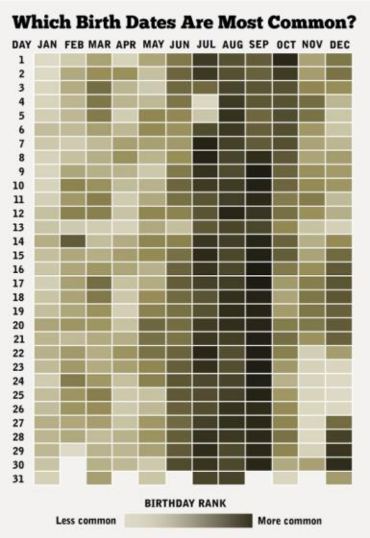


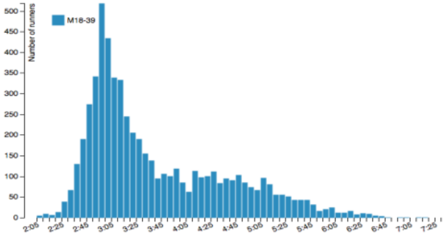
Example: Time







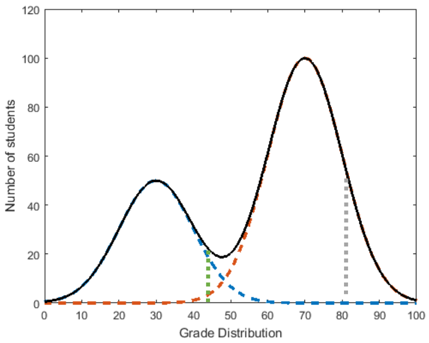




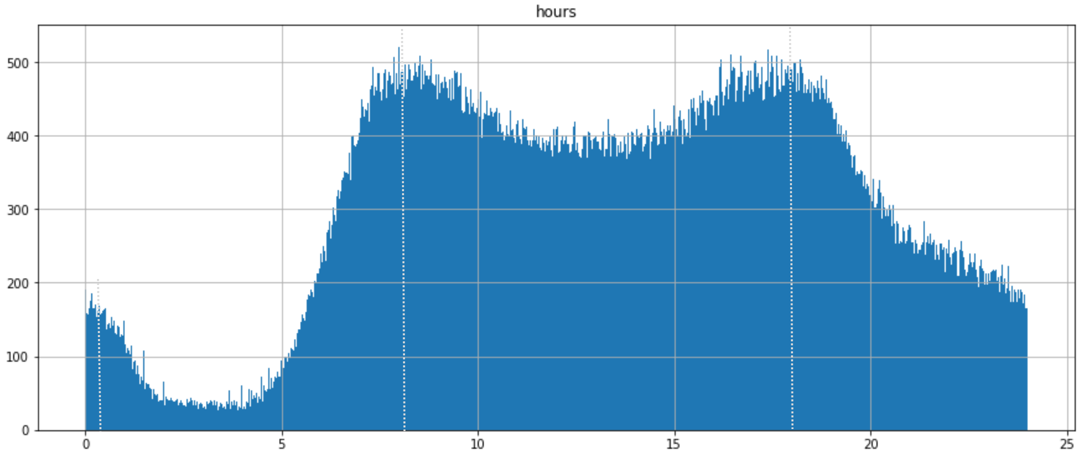
- Timescales

- index

Example: Multimodal data

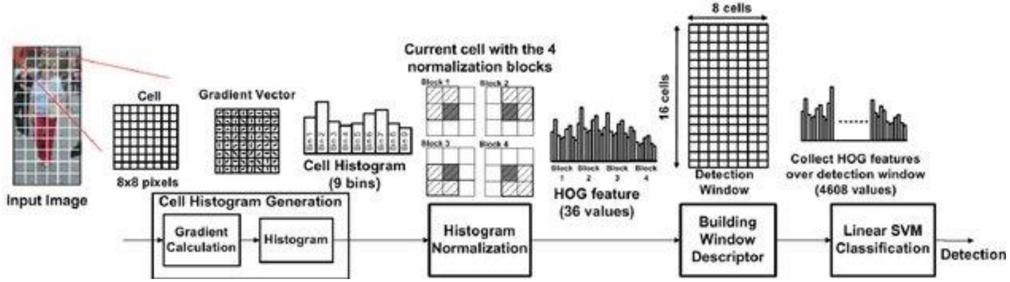


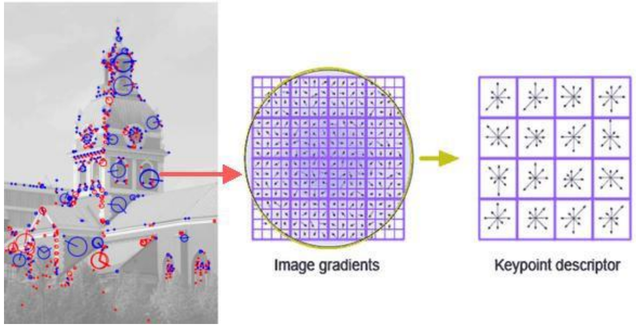
Student 1: 0.4 , -0.1
Student 2: 0.001, 0.5



San Francisco public transport

Example: Image descriptors





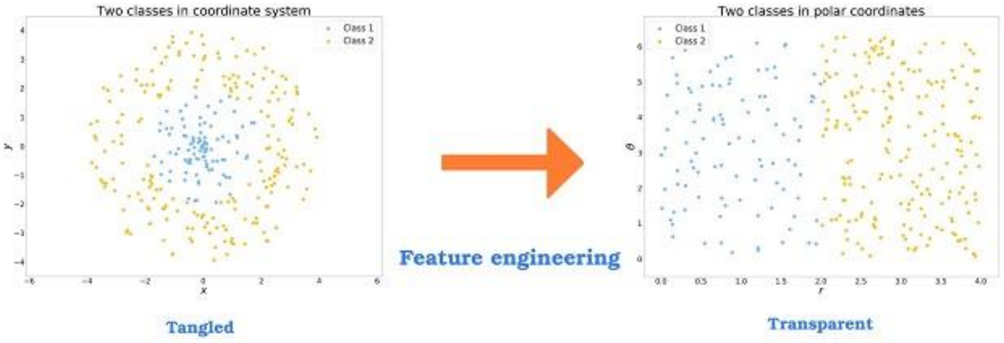
Example: Finance technical indicators

- Different kinds of traders: technical analysis, fundamental analysis...

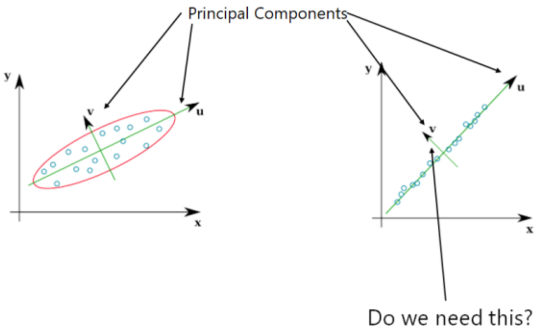
- Technical analysis: look at signals, quantifiable indicators.

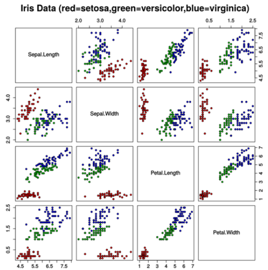
If these traders influence the market, then these signals become important to predict how the market will evolve.

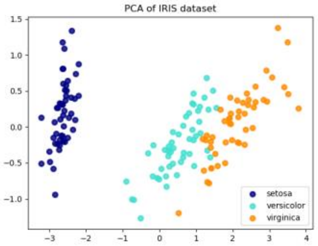
Feature extraction



- Given current data representation, transform into other representation.
- More compact, more robust, more explicit input space
- Or just easier to handle for Machine Learning







Principal Component Analysis

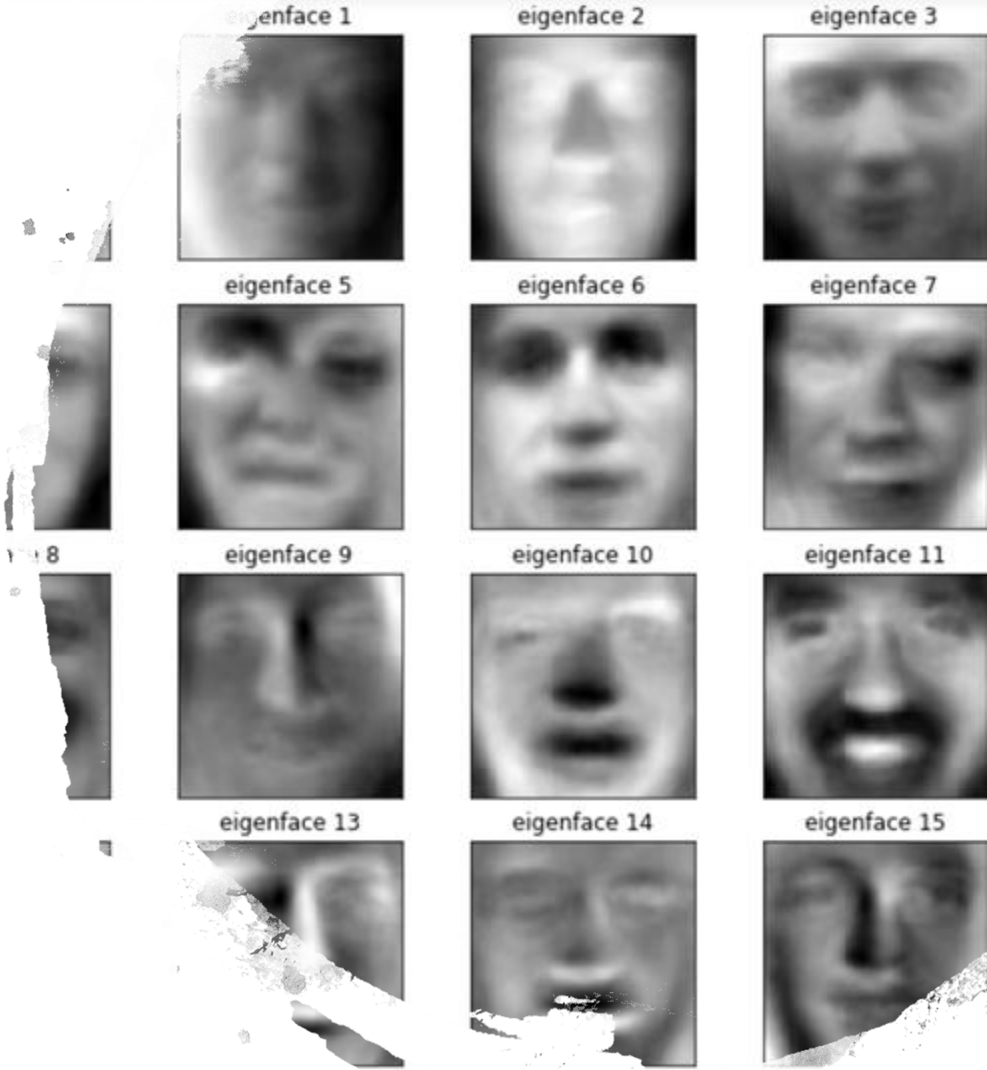
- **Reduce** the dimensionality of a data set by finding a **new set of variables (i.e., principal components)**, smaller than the original set of variables
- Principal components (**PC**s) are **linear combinations** of existing variables
- Ordered by explained variance
- PCs are **uncorrelated**

PCA: Algorithm

- Center and normalize the data
- Compute covariance matrix

$$\Sigma_{ij} = \mathrm{cov}(X_i, X_j) = \mathrm{E}\big[(X_i - \mu_i)(X_j - \mu_j)\big]$$

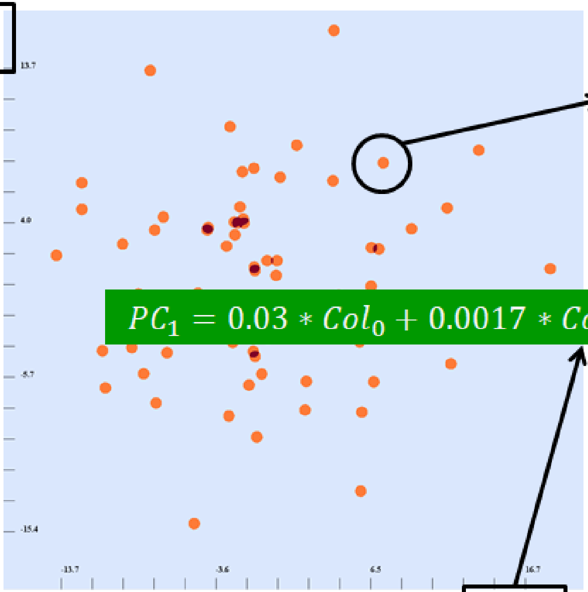
- Calculate eigenvectors of covariance matrix -> PCs
- Project the data on the PCs
- How many dim do you keep?



PCA, details

PC_2

Linear combination of data columns (dimensions)



A row of the data table

$$PC_1 = 0.03 * Col_0 + 0.0017 * Col_1 + \cdots + 0.0001 * Col_{357}$$

PC_1

Loading for a variable

PCA: example with Young People Survey

kaggle

- **Music preferences** (19 items)
- **Movie preferences** (12 items)
- **Hobbies & interests** (32 items)
- **Phobias** (10 items)
- **Health habits** (3 items)
- **Personality traits, views on life, & opinions** (57 items)
- **Spending habits** (7 items)
- **Demographics** (10 items)

```
import pandas as pd
import numpy as np
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt


young_ds = pd.read_csv('data/responses.csv')
```

```
youg_ds_music = young_ds.iloc[:,:19].dropna()
youg_ds_music.head()
```

	Music	Slow songs or fast songs	Dance	Folk	Country	Classical music	Musical	Pop	Rock	Metal or Hardrock	Punk	Hiphop, Rap	Reggae, Ska	Swing, Jazz	Rock n roll	Alternative	Latino	Techno, Trance	Opera
0	5.0	3.0	2.0	1.0	2.0	2.0	1.0	5.0	5.0	1.0	1.0	1.0	1.0	1.0	3.0	1.0	1.0	1.0	1.0
1	4.0	4.0	2.0	1.0	1.0	1.0	2.0	3.0	5.0	4.0	4.0	1.0	3.0	1.0	4.0	4.0	2.0	1.0	1.0
2	5.0	5.0	2.0	2.0	3.0	4.0	5.0	3.0	5.0	3.0	4.0	1.0	4.0	3.0	5.0	5.0	5.0	1.0	3.0
3	5.0	3.0	2.0	1.0	1.0	1.0	1.0	2.0	2.0	1.0	4.0	2.0	2.0	1.0	2.0	5.0	1.0	2.0	1.0
4	5.0	3.0	4.0	3.0	2.0	4.0	3.0	5.0	3.0	1.0	2.0	5.0	3.0	2.0	1.0	2.0	4.0	2.0	2.0

```
data_music = youg_ds_music.to_numpy()
pca = PCA(whiten = True)
X_pca = pca.fit_transform(data_music)
print( pca.explained_variance_ratio_)
```

```
[0.21462856 0.15159296 0.10932881 0.06389559 0.05871546 0.05128242
 0.04683182 0.03940256 0.03736163 0.03154765 0.03065362 0.02501457
 0.02351328 0.02224939 0.0209301  0.02061214 0.01940997 0.0187642
 0.01426525]
```

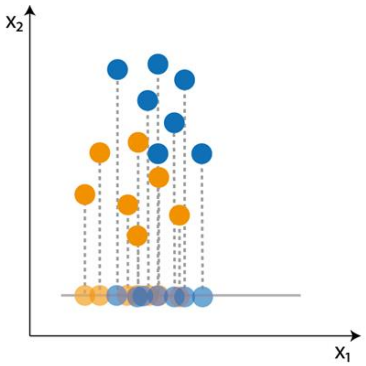
```
youg_ds_hobbies = young_ds.iloc[:,19+12:19+12+32].dropna()
data_hobbies = youg_ds_hobbies.to_numpy()
pca = PCA(whiten = True)
X_pca = pca.fit_transform(data_hobbies)
print( pca.explained_variance_ratio_)
```

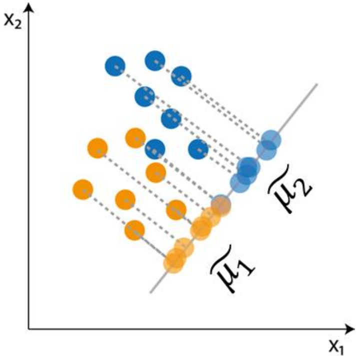
```
[0.13544799 0.10479068 0.08063201 0.06946438 0.05273425 0.04286766
 0.04094587 0.03457081 0.0330224  0.03203322 0.02858578 0.02790116
 0.02637219 0.02556872 0.02287498 0.02230709 0.02090546 0.01967975
 0.01890299 0.01870482 0.01676079 0.01558494 0.01492138 0.01457642
 0.0131472  0.01233027 0.01150779 0.01002716 0.00894522 0.00866793
 0.00825431 0.00696436]
```

```
youg_ds_movies = young_ds.iloc[:,19:19+12].dropna()
data_movies = youg_ds_movies.to_numpy()
pca = PCA(whiten = True)
X_pca = pca.fit_transform(data_movies)
print( pca.explained_variance_ratio_)
```

```
[0.2410899  0.16954119 0.13430844 0.08476939 0.07149352 0.06382178
 0.0593179  0.05306483 0.04439523 0.0290464  0.02663497 0.02251645]
```

Linear Discriminant Analysis





- LDA reduces dimensionality while preserving as much of the **class discriminatory information** as possible

- Suitable when you have class labels and want to find a projection that will help you to separate them

- Tries to **maximise** the difference between **group means** while **minimizing** the **within-class variance**

- **Maximize:**

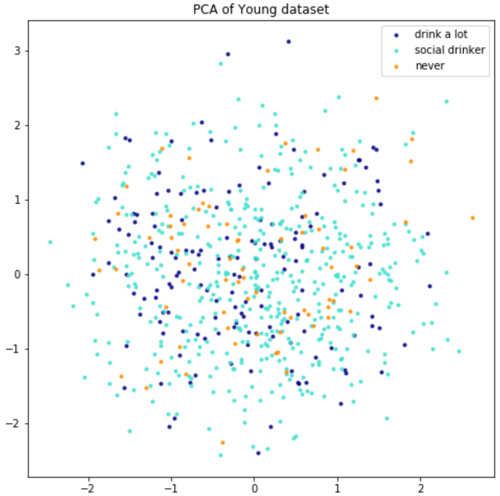
$$J(w) = \frac{|\widetilde{\mu}_1 - \widetilde{\mu}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

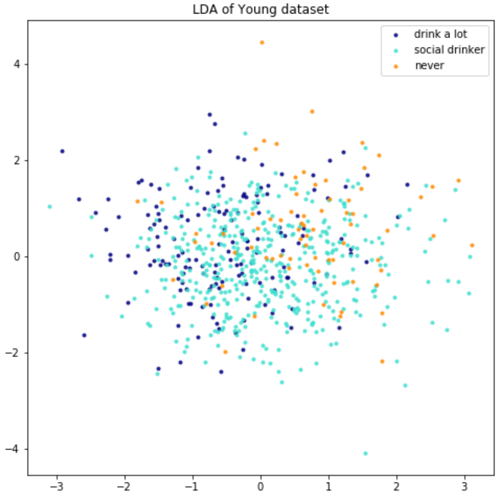
- Can be generalized to high dimension and multi-class

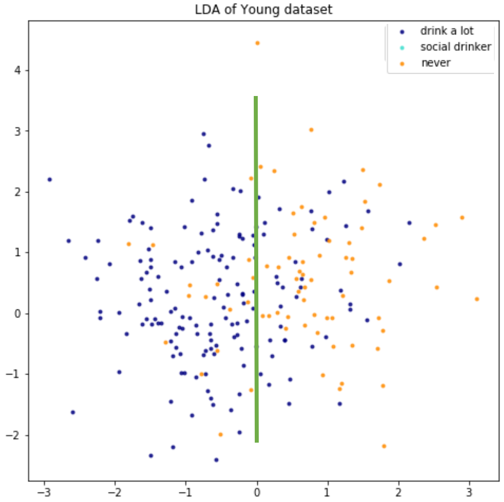
LDA: example

32	History
33	Psychology
34	Politics
35	Mathematics
36	Physics
37	Internet
38	PC Software, Hardware
39	Economy, Management
40	Biology
41	Chemistry
42	Poetry reading
43	Geography
44	Foreign languages
45	Medicine
46	Law
47	Cars
48	Art
49	Religion
50	Outdoor activities
51	Dancing

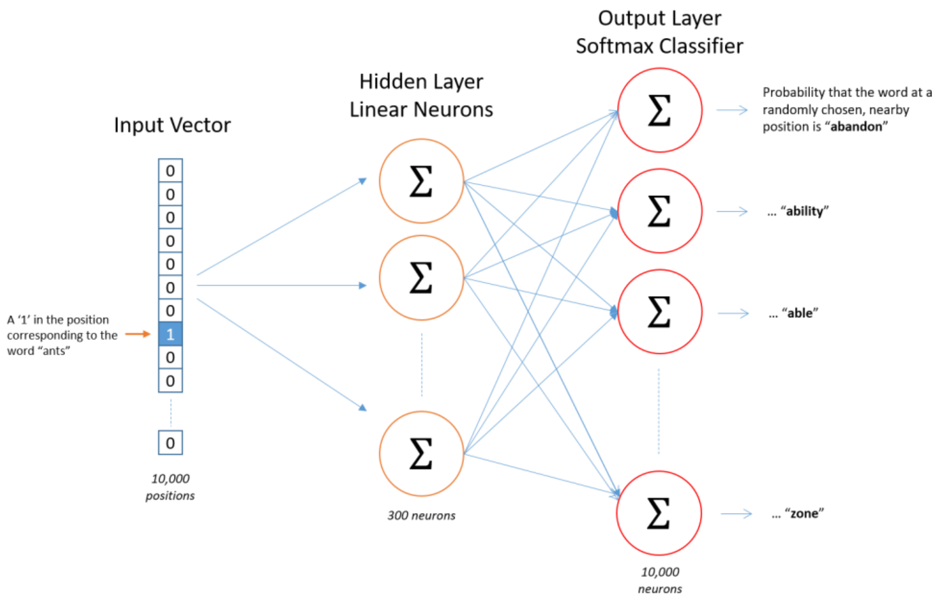
```python
pca = PCA(whiten = True)
X_pca = pca.fit_transform(data_hobbies)

young_ds_drinking = young_ds.iloc[:, 74]
y_target, target_names = young_ds_drinking.factorize()

lda = LinearDiscriminantAnalysis(n_components=2)
lda_projection = lda.fit(data_hobbies, y_target).transform(data_hobbies)
```



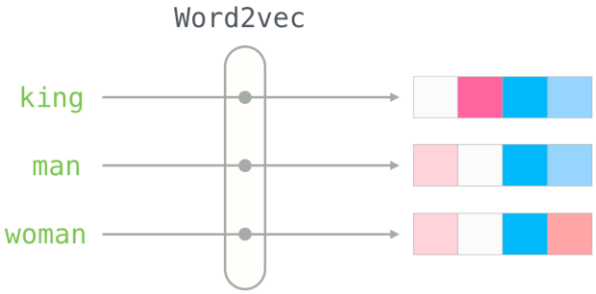




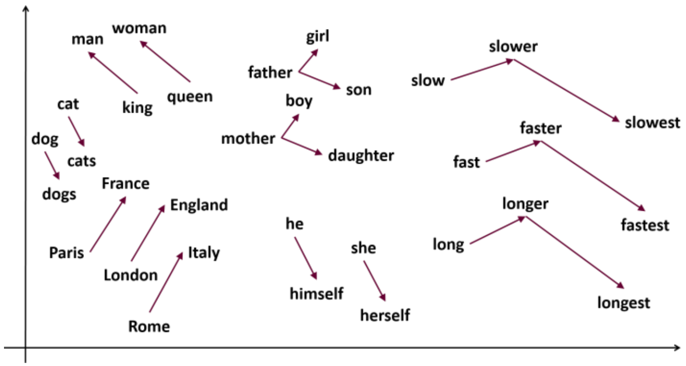
Word2Vec



Input Vector
Hidden Layer
Linear Neurons
Output Layer
Softmax Classifier
0
0
0
0
0
0
0
1
0
0
0
A '1' in the position corresponding to the word "ants"
10,000 positions
Σ
300 neurons
Probability that the word at a randomly chosen, nearby position is "**abandon**"
... "**ability**"
... "**able**"
... "**zone**"
10,000 neurons

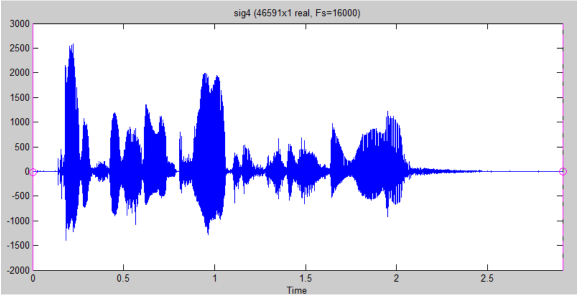


Word2vec
king
man
woman

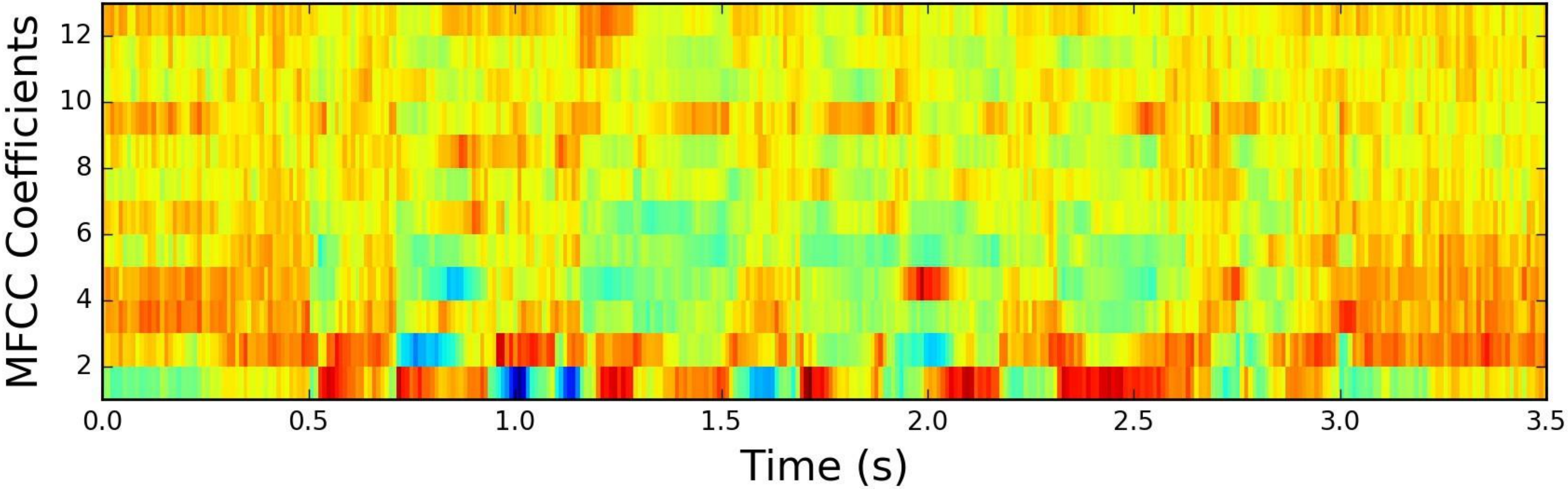


man
woman
king
queen
cat
cats
dog
dogs
France
England
Paris
London
Italy
Rome
girl
father
son
boy
mother
daughter
he
she
himself
herself
slow
slower
slowest
fast
faster
fastest
long
longer
longest

MFCCs



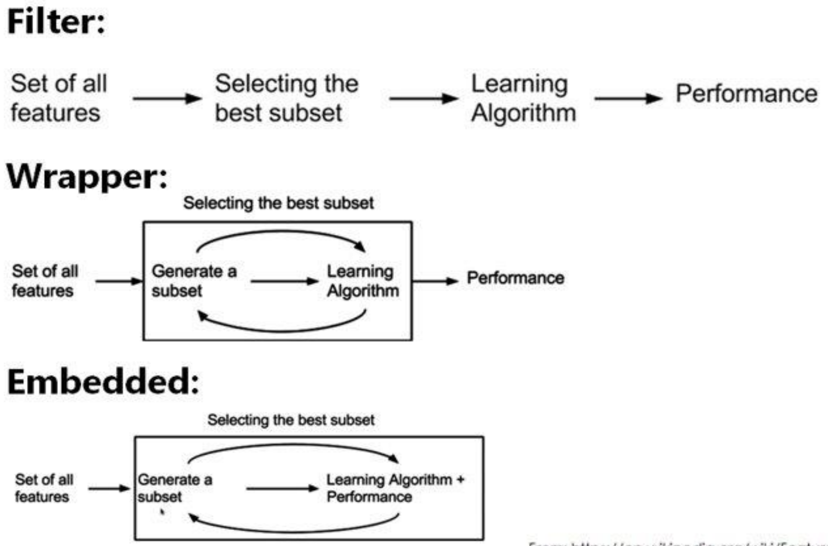
sig4 (46591x1 real, Fs=16000)
3000
2500
2000
1500
1000
500
0
-500
-1000
-1500
-2000
0
0.5
1
1.5
2
2.5
Time



MFCC Coefficients
12
10
8
6
4
2
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Time (s)

Feature selection

- Choose best features
- Limit number to prevent computational issues
- Measure quality of feature
- Filter vs wrapper vs embedded

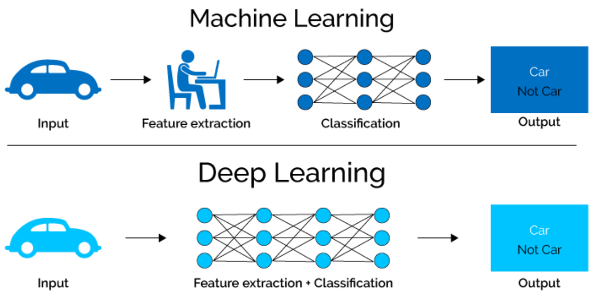


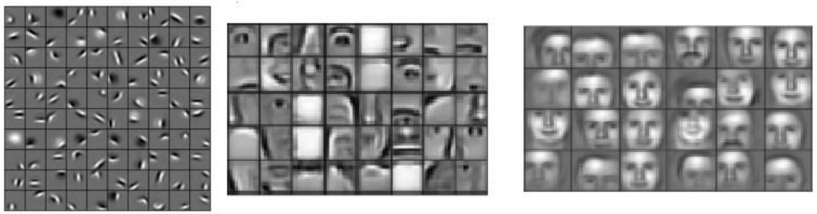
From: https://en.wikipedia.org/wiki/Feature_selection

- Other topics: Feature expansion, Automatic feature generation

Deep Learning?

- Leverages huge quantity of data to **learn features**





- But you won't always have big data problems!
  - Dataset size and dimensionality
  - Memory, compute

- Finance: need insurance, cover risks
- Medical: help doctors, not replace them
- Self-driving cars (don't worry, my algorithm was trained in Paris)
- Being able to know why something has gone terribly wrong...

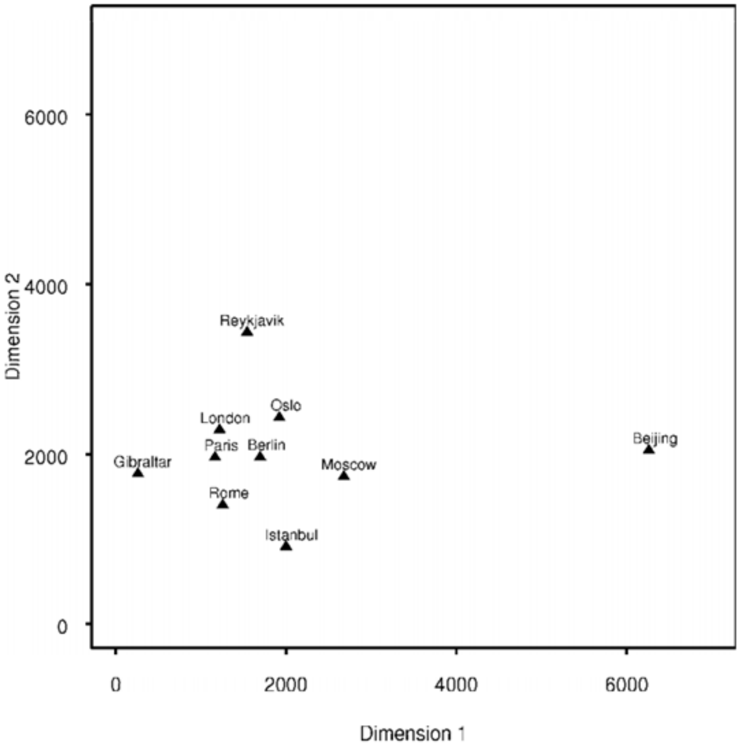
Sometimes, you need explainable features

Projection for visualization

- Get intuition from data
- Goal: project to 2D or 3D
- Focus on non-linear models

Example 1: Multi-dimensional Scaling

	London	Berlin	Oslo	Moscow	Paris	Rome	Beijing	Istanbul	Gibraltar	Reykjavik
London	–									
Berlin	570	–								
Oslo	710	520	–							
Moscow	1550	1000	1020	–						
Paris	210	540	830	1540	–					
Rome	890	730	1240	1470	680	–				
Beijing	5050	4570	4360	3600	5100	5050	–			
Istanbul	1550	1080	1520	1090	1040	850	4380	–		
Gibraltar	1090	1450	1790	2410	960	1030	6010	1870	–	
Reykjavik	1170	1480	1080	2060	1380	2040	4900	2560	2050	–



MDS: algorithm

- Tries to preserve high-dimensional distances in the lower dimensional projection
- For instance, metricMDS (an implementation) tries to minimize:

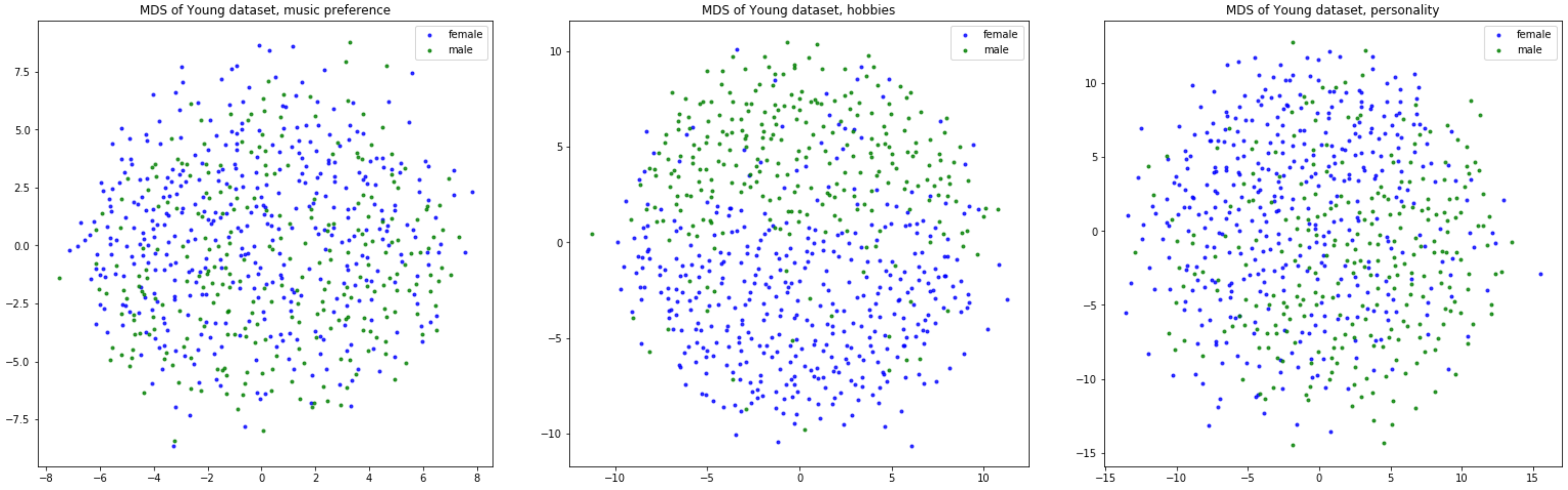
$$\min_{Y} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\boxed{d_{ij}^{(X)}} - \boxed{d_{ij}^{(Y)}} \right)^2$$

Original distances in high-dim.

Distances in reduced dimensionality

- Suitable for visualisation
- Depends on how distance is defined
- Curse of dimensionality (??)
- Meaning in **proximity** but not so much the axes.

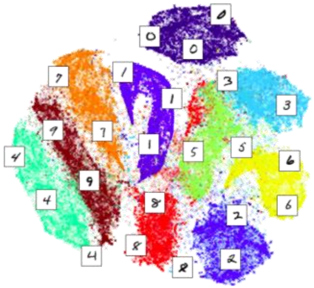
MDS: example


MDS of Young dataset, music preference
female
male
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−8 −6 −4 −2 0 2 4 6 8
MDS of Young dataset, hobbies
female
male
10
5
0
−5
−10
−10 −5 0 5 10
MDS of Young dataset, personality
female
male
10
5
0
−5
−10
−15
−15 −10 −5 0 5 10 15

Example 2: t-SNE

- Applicable to very large datasets
- Very effective to learn local structures
- Well-suited for 2D or 3D visualization

General idea: Match probability distribution of distances in high dimension and low dimension



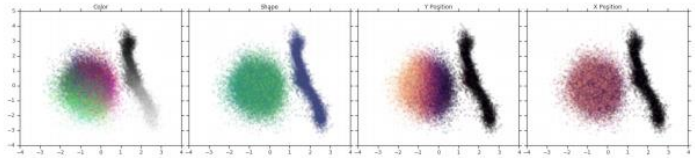


Figure 25. Projection on the first two principal components of the latent distribution for the **Multi-dSprites** dataset. Each dot represents one object latent and is colored according to the corresponding ground truth factor.

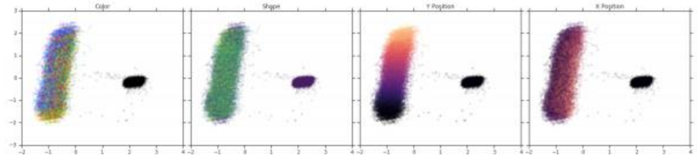


Figure 26. Projection on the first two principal components of the latent distribution for the **Tetris** dataset. Each dot represents one object latent and is colored according to the corresponding ground truth factor.

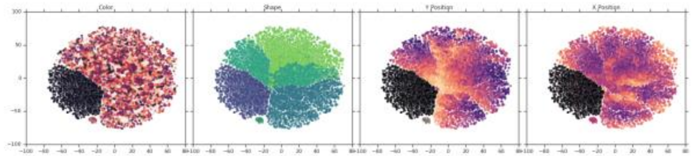


Figure 27. t-SNE of the latent distribution for the **CLEVR6** dataset. Each dot represents one object latent and is colored according to the corresponding ground truth factor

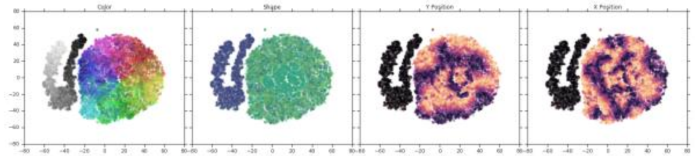
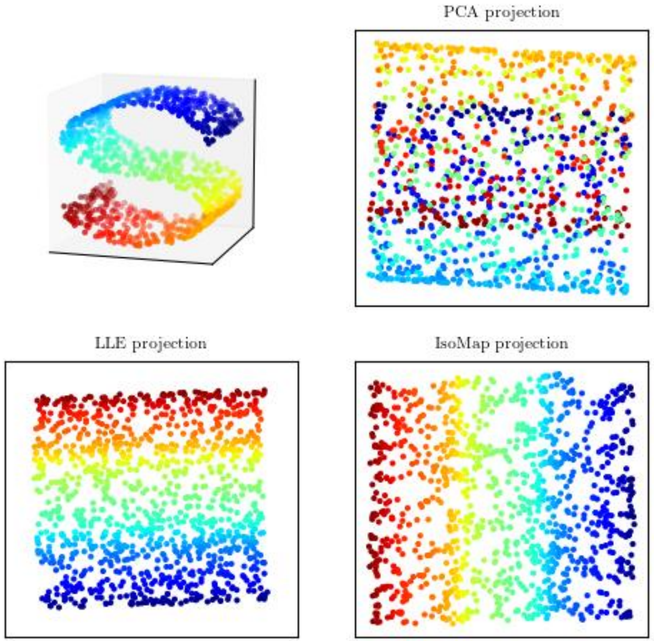
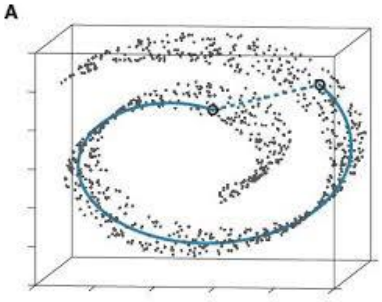


Figure 28. t-SNE of the latent distribution for the **Multi-dSprites** dataset. Each dot represents one object latent and is colored according to the corresponding ground truth factor

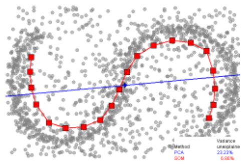
Example 3: IsoMap



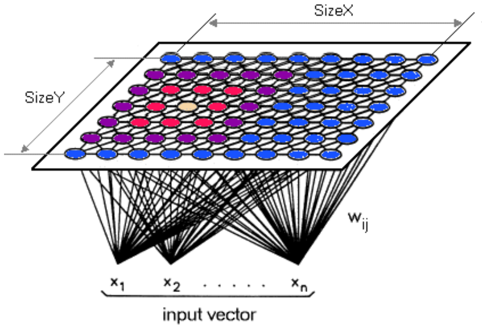
- MDS + geodesic distances
- Determine graph of neighbors
- Measure geodesic distances between nodes
- Compute dimensionality reduction on distances (MDS)

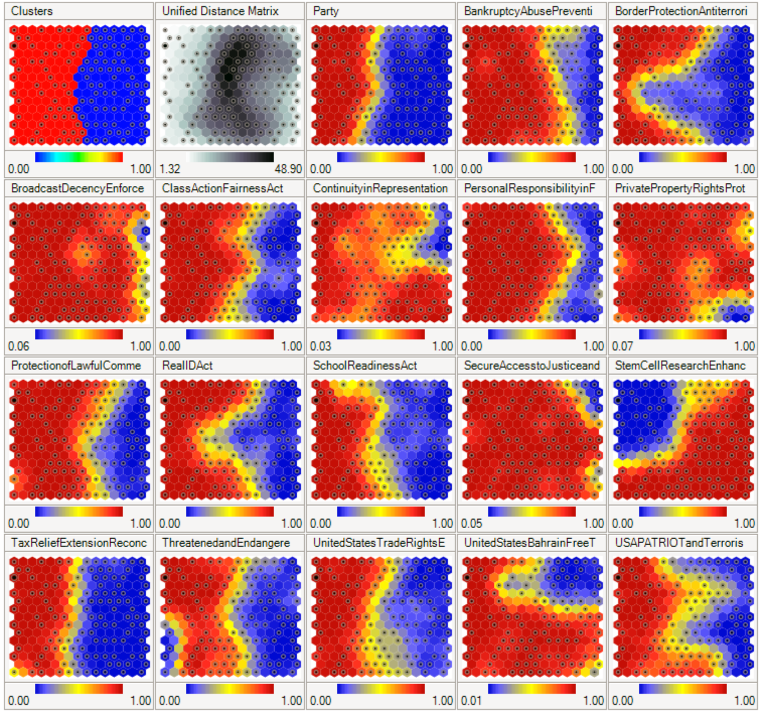


Example 4: SOMs

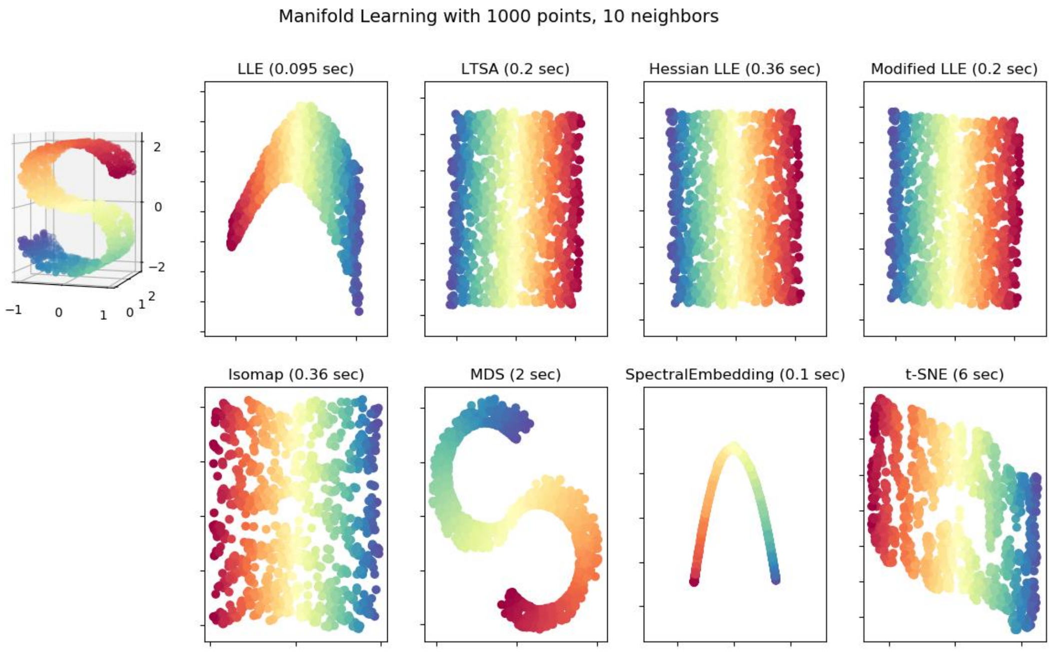


- Projection to 2D or 3D map using Neural Networks
- Iterative competitive learning:
  - Pick new data point
  - Find Best Matching Unit
  - Update BMU and neighbours toward data point





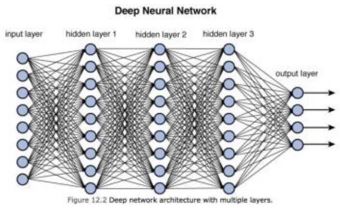
Other examples

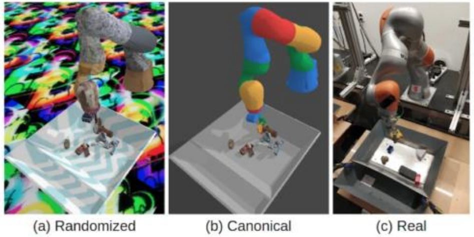


- What are the assumptions on the shape of the manifold?
- Can it deal with the dimensionality of your dataset?

Feature learning and Big data ?

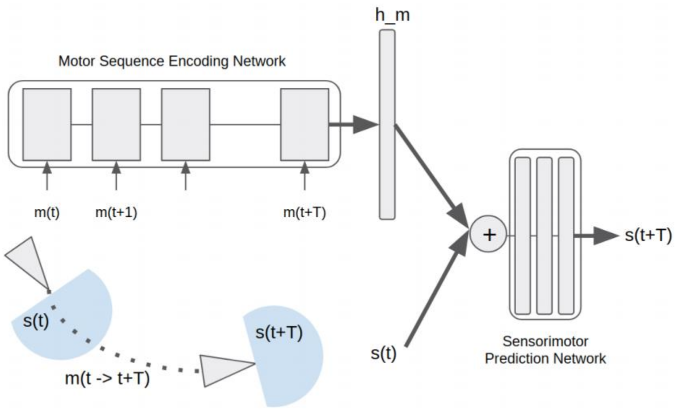
- Use huge quantity of data to learn features:
  - Plenty of labeled data: end-to-end
  - Plenty of unlabeled data: auto-encoder, generative models
  - Transfer: pre-trained network as feature extractor



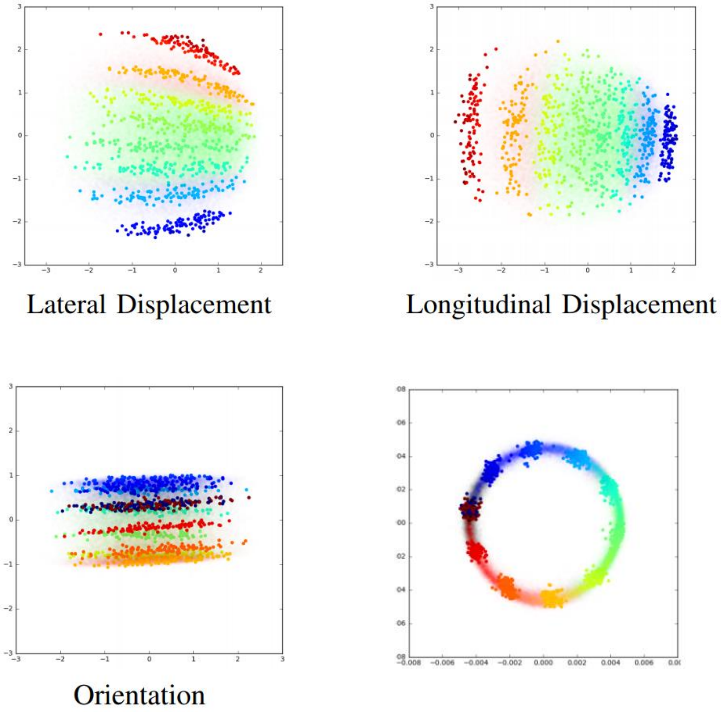


- Difficult to visualize / interpret -> explainability
  Open questions in AI:
  - Human brain is (still) not explainable, but we still rely on it
  - Should we rely on algorithms which perform better, but we don't know why?
  - Who needs explainability? The user ? The engineer?

Questions?



h_m
Motor Sequence Encoding Network
m(t)
m(t+1)
m(t+T)
+
s(t+T)
s(t)
s(t+T)
m(t -> t+T)
s(t)
Sensorimotor
Prediction Network



Lateral Displacement
Longitudinal Displacement
Orientation