Artificial intelligence in data science

Unsupervised learning

Janos Török

Department of Theoretical Physics

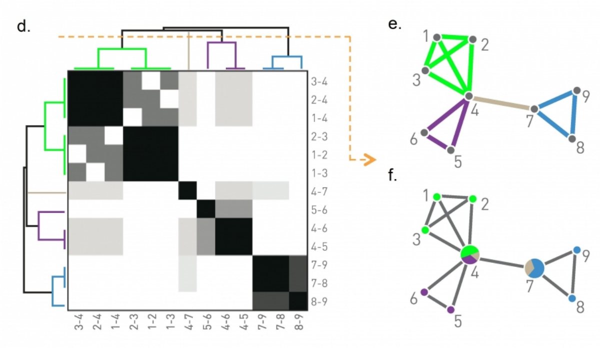
December 7, 2023

Unsupervised learning

- Items do not have class associated with them
- If we have distance
  - k-means clustering
  - Hierarchical clustering
  - etc.
- If we have graph structure
  - Modularity maximization (nodes have more links towards other nodes in the modeule than elsewhere)
  - Cut links which belong to the most minimal path (Girvan-Neumann)
  - Any other graph partition method

Distance ↔ Graph

- Distance to graph
  - Tresholding
  - Similarity
  - Weighted graph
- Graph to distance
  - Graph distance
  - Node similarity (zeleons of measures)

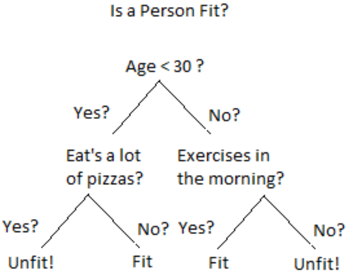


Decision tree, random forest, hierarchical clustering

Why?

- Decision tree
- Random forest
- Importance of parameters
- Unsupervised learning

Decision tree



- Build a tree
- Nodes are yes-no questions
- Links are answers (yes/no)
- Leaves are classification statements

Decision tree

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

- Which parameter to pick first?
- The one which classifies the data best
- What is *best*? $\rightarrow$ information gain or Gini index

Information entropy

- Set of possible outcomes C
- Possible outcomes $c_i \in C$
- The number of experiments is N and the respective events happend n_i times $\sum_i n_i = N$
- The probability with which the above outcome may have happend $P \propto \frac{N!}{n_1! \cdots n_k!}$
- Probability of two independent events $P(1)P(2)$
- Entorpy for independent system is additive so let us use log and of course Stirling's formula for the factorial: $S \equiv \log(P) \simeq -\sum_i p_i \log(p_i)$, with $p_i = n_i/N$
- So for events with probability p_i:

$$H(s) = \sum_i -p_i \log_2 p_i$$

Information entropy

- $H(s) = \sum_{c \in C} -p(c) \log_2 p(c)$, $C = \{\text{yes}, \text{no}\}$
- For the full set:
- 9 out of 14 are yes:

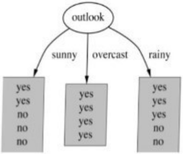
$$H(s) = -\frac{9}{14}\log_2\frac{9}{14} - \frac{5}{14}\log_2\frac{5}{14} = 0.41 + 0.53 = 0.94$$

- Information entropy for perfectly separated $H = 0$, information entropy of perfectly mixed system $H = 1$

outlook	temp.	humidity	windy	play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Information gain, for every feature:

Information entropy of the original minus the one of the divided



E (Outlook=sunny) = $-\frac{2}{5}\log\left(\frac{2}{5}\right)-\frac{3}{5}\log\left(\frac{3}{5}\right)=0.971$

E (Outlook=overcast) = $-1\log(1)-0\log(0)=0$

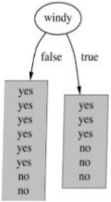
E (Outlook=rainy) = $-\frac{3}{5}\log\left(\frac{3}{5}\right)-\frac{2}{5}\log\left(\frac{2}{5}\right)=0.971$

} H(S,Outlook)

Average Entropy information for Outlook

I (Outlook) = $\frac{5}{14}*0.971+\frac{4}{14}*0+\frac{5}{14}*0.971=0.693$ } $\sum_{t\in T} p(t)H(t)$

Gain (Outlook) = E(S) – I (outlook) = 0.94-.693 = 0.247 ⇨ $IG(A,S)=H(S)-\sum_{t\in T} p(t)H(t)$



E (Windy=false) = $-\frac{6}{8}\log\left(\frac{6}{8}\right)-\frac{2}{8}\log\left(\frac{2}{8}\right)=0.811$

E (Windy=true) = $-\frac{3}{6}\log\left(\frac{3}{6}\right)-\frac{3}{6}\log\left(\frac{3}{6}\right)=1$

Average entropy information for Windy

I (Windy) = $\frac{8}{14}*0.811+\frac{6}{14}*1=0.892$

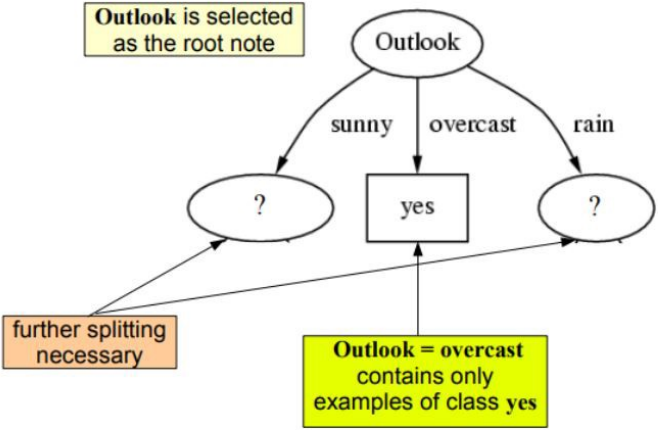
Gain (Windy) = E(S) – I (Windy) = 0.94-0.892=0.048

Information gain, for every feature, pick the highest:

Outlook		Temperature	
Info:	0.693	Info:	0.911
Gain: 0.940-0.693	0.247	Gain: 0.940-0.911	0.029
Humidity		**Windy**	
Info:	0.788	Info:	0.892
Gain: 0.940-0.788	0.152	Gain: 0.940-0.892	0.048

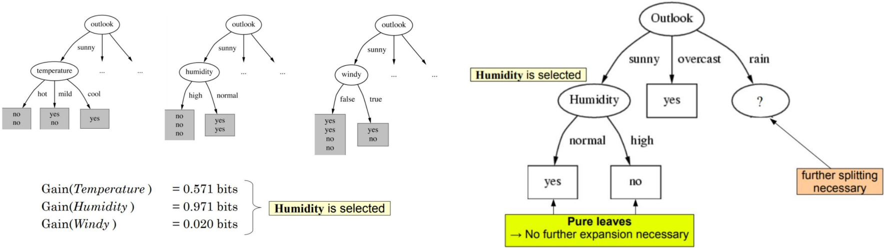
- So root node is Outlook.

Decision tree: First level



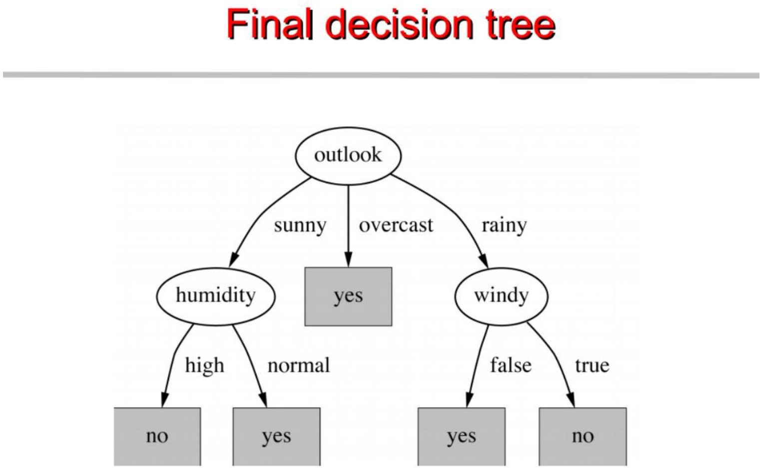
- So root node is Outlook.

Decision tree: Next levels, same procedure



- Next question is about Humidity.

Final decision tree



Gini index

- $Gini = 1 - \sum_{c \in C} p(c)^2,\ C = \{\text{yes}, \text{no}\}$
- For the full set:
- 9 out of 14 are yes:

$$Gini = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.46$$

- For perfectly separated sample Gini index is zero.

Gini index, for two groups

- Fraction weighted sum of respective Gini indices
- Example:

Class	A	A	A	A	A	B	B	B	B	B
v	0	0	0	0	1	1	1	0	1	0

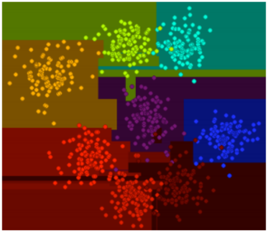
- v=1: $Gini(1) = 1 - (1/4)^2 - (3/4)^2$
- v=0: $Gini(0) = 1 - (4/6)^2 - (2/6)^2$
- Combined Gini:

$$Gini = \frac{4}{10} Gini(1) + \frac{6}{10} Gini(0)$$

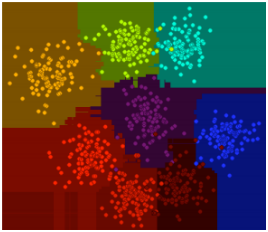
Decision tree

- Advantages
  - Fast
  - Easy to interpret
  - Can be combined with other techniques
- Disadvantages
  - Very unstable (small change in the data, enormous change in the tree)
  - Very inaccurate
  - Separation lines parallel to axes

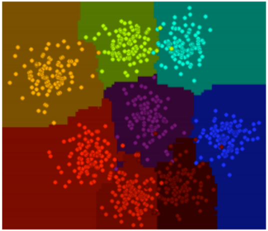
Unsupervised random forest: Illustration



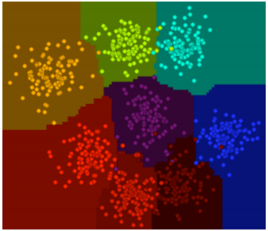
1 rCART



10 rCARTs



100 rCARTs



500 rCARTs

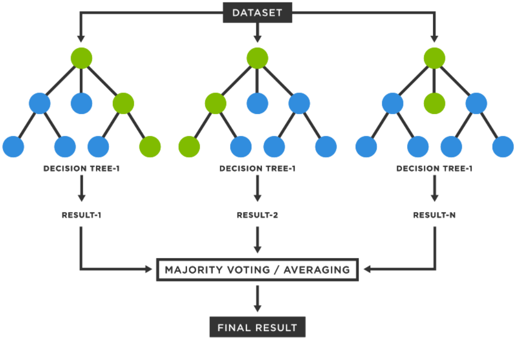
From: Eric Debreuve / Team Morpheme University Nice Sophia Antipolis

Random forest

- Bagging trees (Bootstrap Aggregating)
  - Bagging: Average a given procedure over many samples to reduce the variance
  - Draw bootstrap samples from the the original sample and to the training. Original dataset: `x = c(x1, x2, ..., x100)` Bootstrap samples: `boot1 = sample(x, 100, replace = True),`
  - Average the results
- Random forest
  - When selecting the random sample fewer data is used
  - Average the prediction of each tree
  - Much more stable than decision tree (indeed the forest looks more impressive and stable than a single tree!)

Random forest

- Data importance measure
  - How much the accuracy decreases when the variable is excluded
  - The decrease of Gini impurity when a variable is chosen to split a node



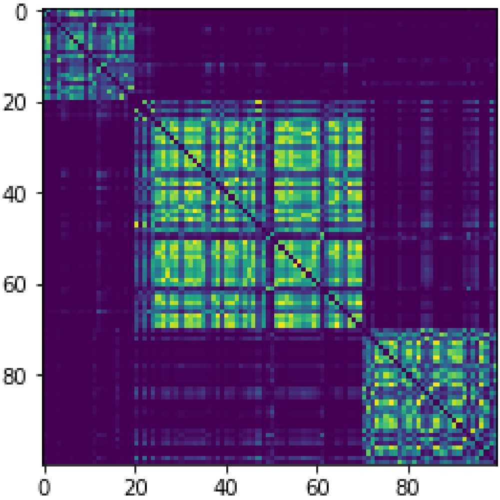
Unsupervised learning

- Cluster methods: k-means, hierarchical clustering, etc.
- Principal component analysis
- Anomaly detection
- **Teach a method to distinguish between the real and a synthetic data**

Random forest unsupervised

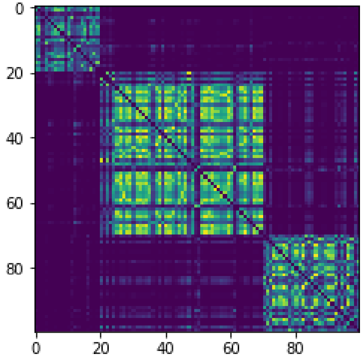
- How to make a decision tree without target?
  - Create a synthetic data set
  - Mark the original dataset with target 1 and the synthetic with target 0
  - Use random forest to find dissimilarity between the random and the real data.
- After each decision tree is trained, fit the original dataset
- Points ending up in the same leaf are related.
- Aggregating this events creates a similarity matrix.
- Can use other methods to cut them into pieces

Unsupervised random forest similarity matrix



Unsupervised random forest

- The algorithm results in a distance matrix
- Norms and distances in the mixed original data can be misleading



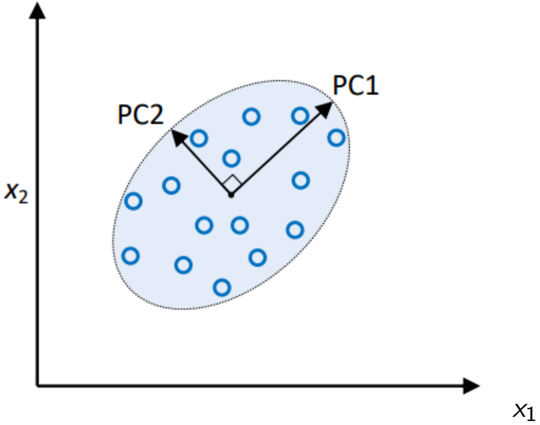
Dimension reduction

- Images contain too much data compared to output, (e.g. VGG16, input $228 \times 228 \times 3 = 155952$, output 1000.)
- Methods to retrieve the relevant information
  - Eigen decomposition
  - Principal component analysis
  - Autoencoder
  - All fall in the unsupervised cathegory

Figures from Sebastian Raschka

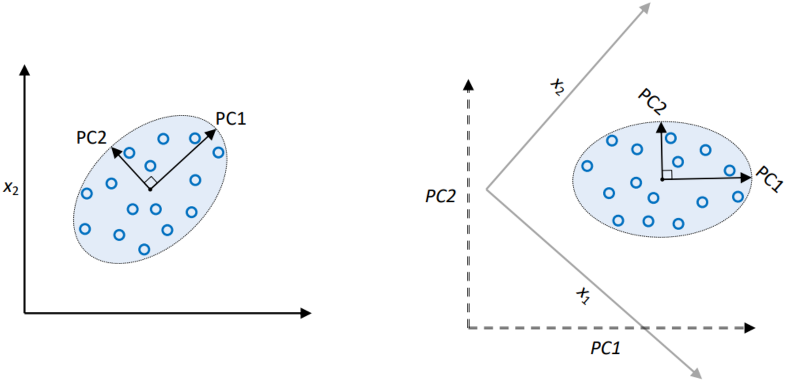
Principal component analysis

- PCA :Find directions of maximum variance
- Eigen decomposition: Consider data $N \times P$ as a matrix. Consider the eigen vectors with the largest absolute eigen values.



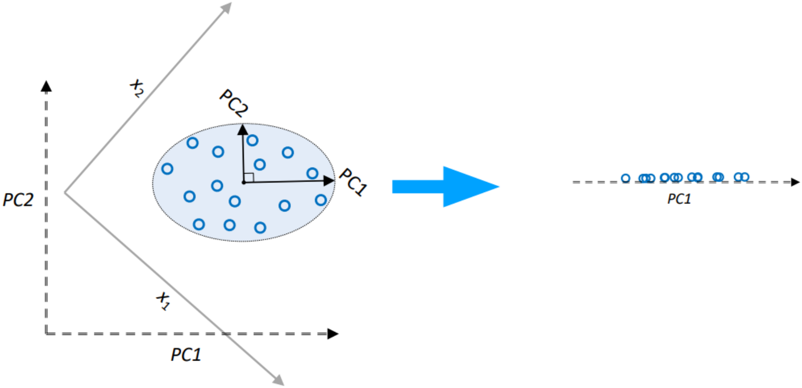
PCA

- Transform data



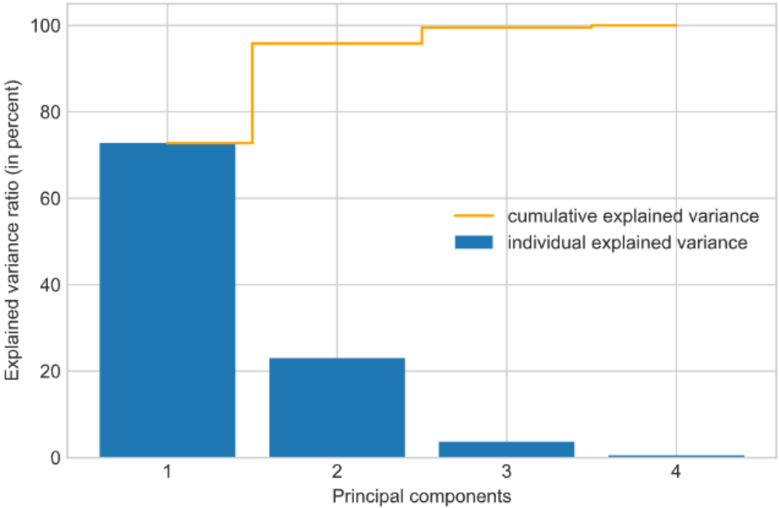
PCA

- Keep relevant dimensions



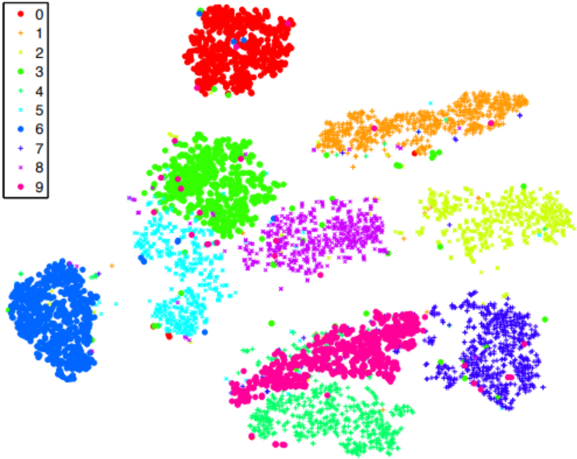
PCA

- Keep relevant dimensions



PCA

- If you are lucky a few dimensions are enough to tell the categories apart.

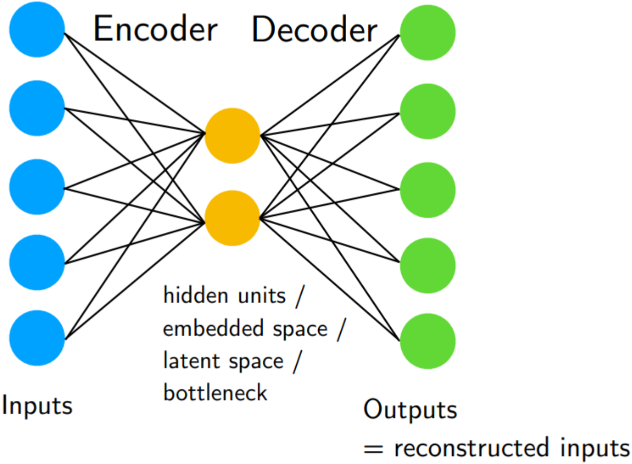


(a) Visualization by t-SNE.

Shown are 6000 images from MNIST projected in 2D

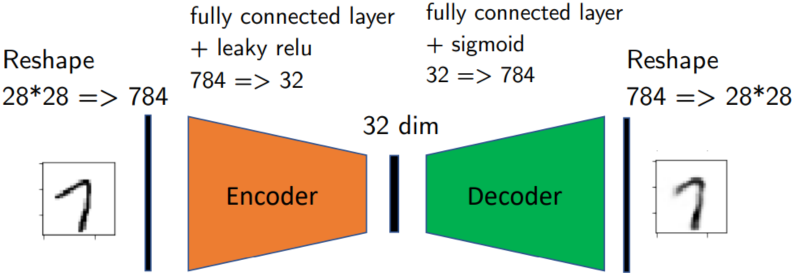
Autoencoder

- Make the machine learn the important components
- Make a bottleneck in the network.
- Teach the network the image itself



Autoencoder

- Make the machine learn the important components
- Make a bottleneck in the network.
- Teach the network the image itself



Autoencoder

- Transposed convolution
- Upscaling

Regular Convolution:

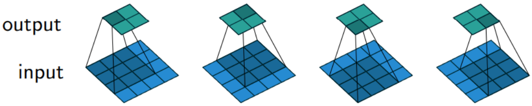
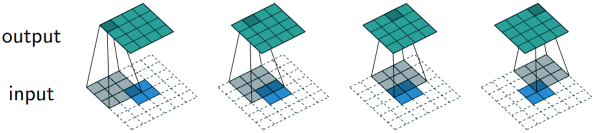


Figure 2.1: (No padding, unit strides) Convolving a 3×3 kernel over a 4×4 input using unit strides (i.e., $i = 4$, $k = 3$, $s = 1$ and $p = 0$).

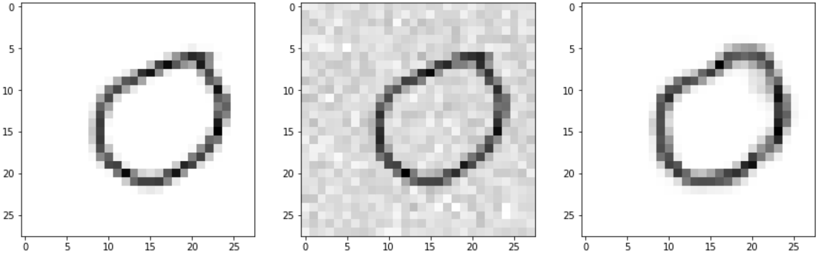
Transposed Convolution (emulated with direct convolution):



Dumoulin, Vincent, and Francesco Visin. "A guide to convolution arithmetic for deep learning." *arXiv preprint arXiv:1603.07285* (2016).

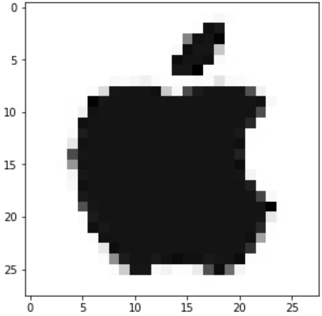
Use cases of Autoencoder

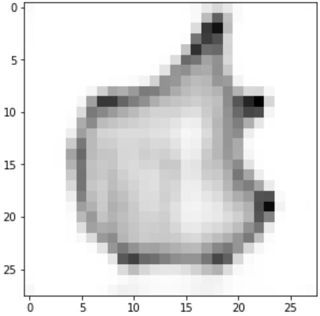
- Noise reduction: noise is a lot of information, since it has no correlation, most of it will be lost at the bottleneck.
- Missing part reconstruction



Use cases of Autoencoder

- Noise reduction
- Missing part reconstruction
- Images in given style





Use cases of Autoencoder

- Noise reduction
- Missing part reconstruction
- Images in given style https://arxiv.org/abs/1508.06576