Instance-based learning

The curse of dimensionality and the blessing of non-uniformity

Vlad Gladkikh

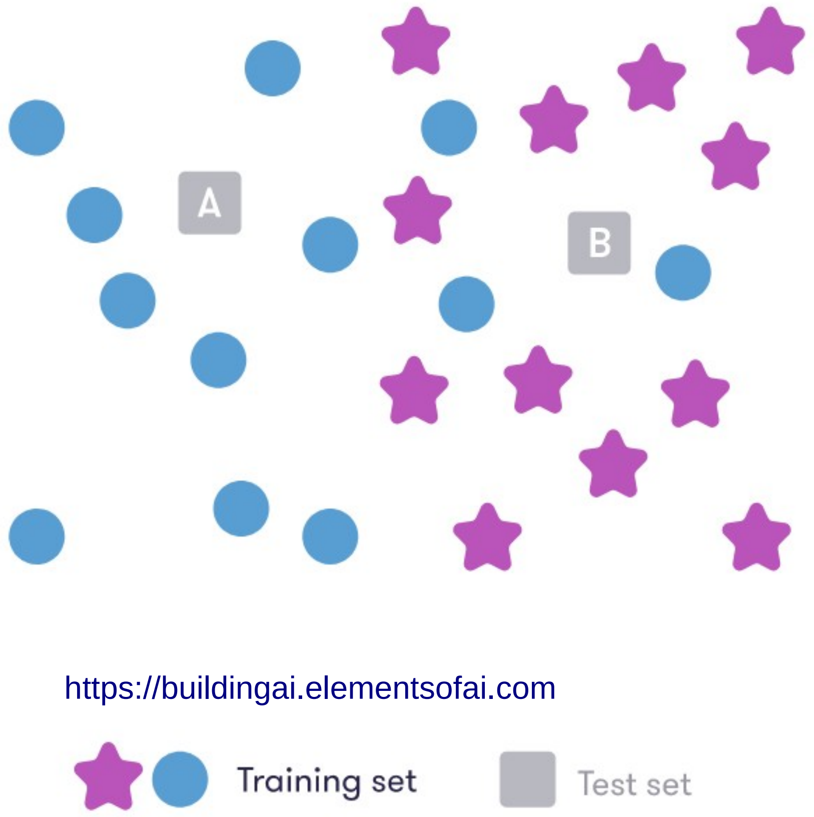
IBS CMCM

Assumption: Similar instances should have similar class labels (in classification) or similar target values (regression).

A very simple algorithm that often beats the most sophisticated ones.

Non-linear classification boundary

No training



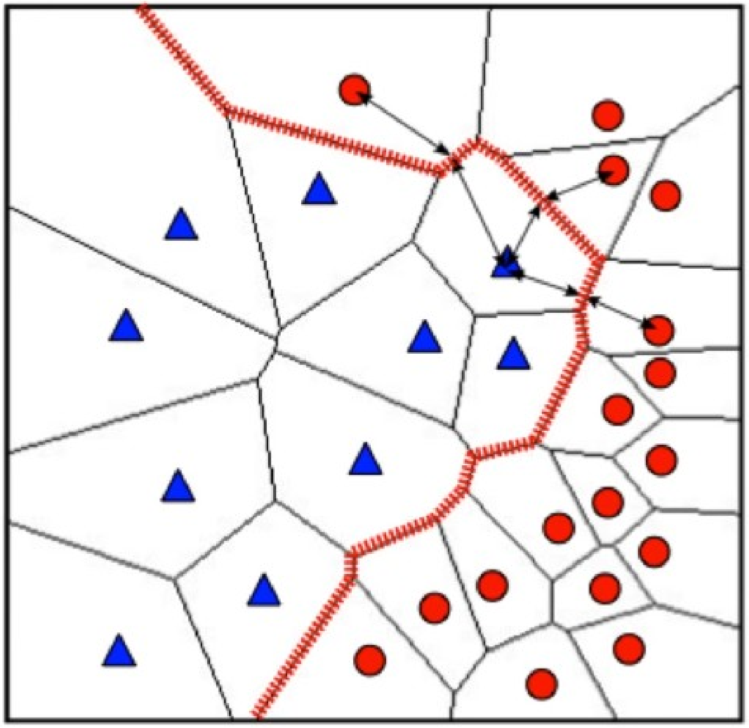
https://buildingai.elementsofai.com

Downside:
the algorithm is computationally expensive, and is prone to overfitting.

The definition of 'distance' can make a big difference to the accuracy of the method.

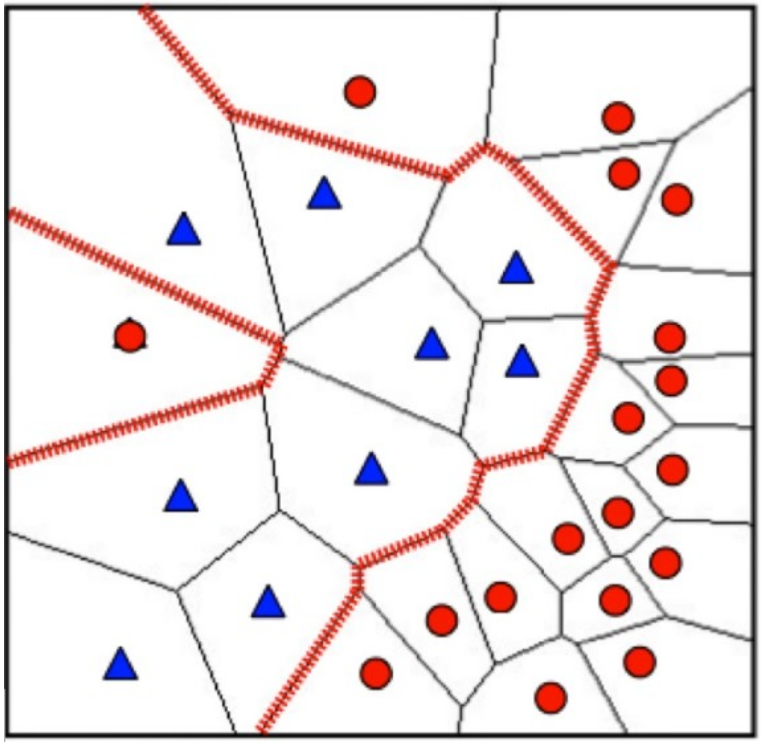
Houle et al (2010) Can Shared-Neighbor Distances Defeat the Curse of Dimensionality?. In: Scientific and Statistical Database Management. SSDBM 2010. https://doi.org/10.1007/978-3-642-13818-8_34

https://towardsdatascience.com/the-proper-way-of-handling-mixed-type-data-state-of-the-art-distance-metrics-505eda236400 https://www.youtube.com/playlist?list=PLBv09BD7ez_48heon5Az-TsyoXVYOJtDZ



The algorithm assigns a label not to a single point, rather to a region in space.

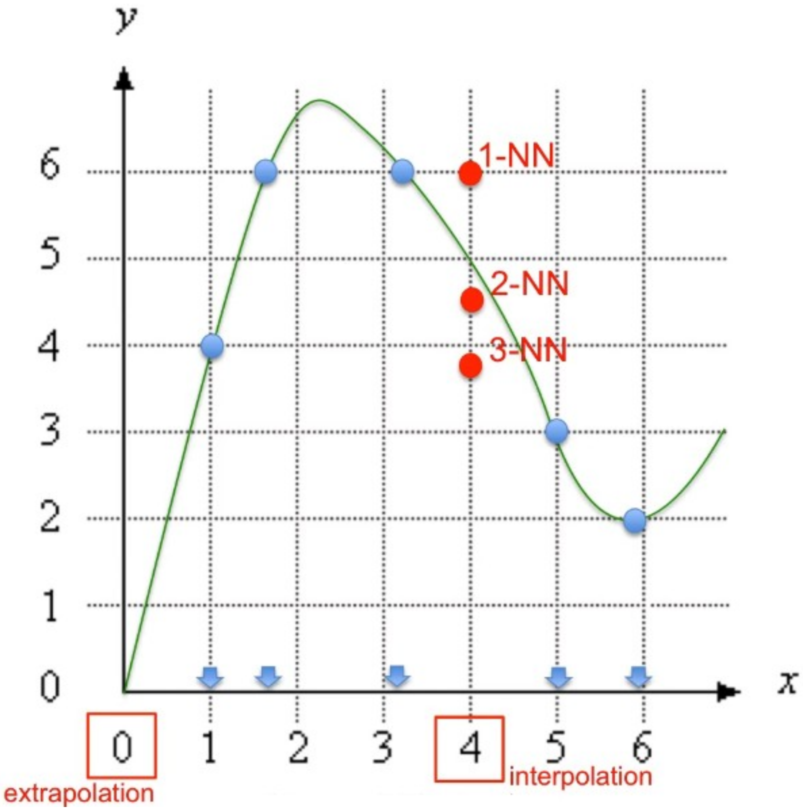
Voronoi tesselation



Problem: outliers

One point can change the boundary drastically

Solution: use more than one nearest neighbor (an odd number) to make a decision https://www.youtube.com/playlist?list=PLBv09BD7ez_48heon5Az-TsyoXVYOJtDZ



There is an optimal number of neighbors

All neighbors just give you the mean

Does not work outside the training set

$$f(x) = \sum_i 1_{x_i \in N(x)} \frac{k(x, x_i)}{|N|}$$

looks like a kernel method

https://www.analyticsvidhya.com/blog/2018/08/k-nearest-neighbor-introduction-regression-python/

A drawback of the "majority voting" occurs when the class distribution is skewed.

Examples of a more frequent class dominate the prediction of the new example, because they are common among the *k* nearest neighbors due to their large number.

One way to overcome skew is by abstraction in data representation.

For example, in a **self-organizing map** (**SOM**), each node is a representative (a center) of a cluster of similar points, regardless of their density in the original training data.

k-NN can then be applied to the SOM.

Self-Organizing Maps

aka Kohonen Networks

Teuvo Kohonen (1982)

https://link.springer.com/article/10.1007/BF00337288

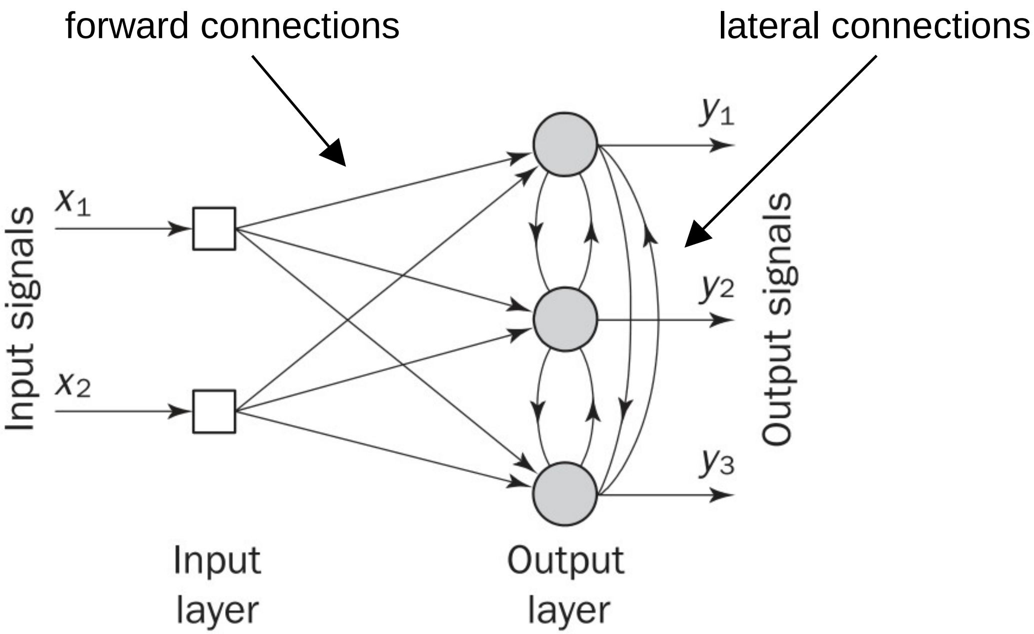
A **self-organizing map** (**SOM**) is a grid of neurons which adapt to the topological shape of a dataset, allowing us to visualize large datasets and identify potential clusters.

A SOM learns the shape of a dataset by repeatedly moving its neurons closer to the data points.

This is an unsupervised learning.

More specifically: competitive learning

In competitive learning, neurons compete among themselves to be activated.



forward connections

lateral connections

Michael Negnevitsky. Artificial Intelligence. A Guide to Intelligent Systems. (3rd Edition) (2011)

Raul Rojas. Neural Networks. A Systematic Introduction

https://algobeans.com/2017/11/02/self-organizing-map/

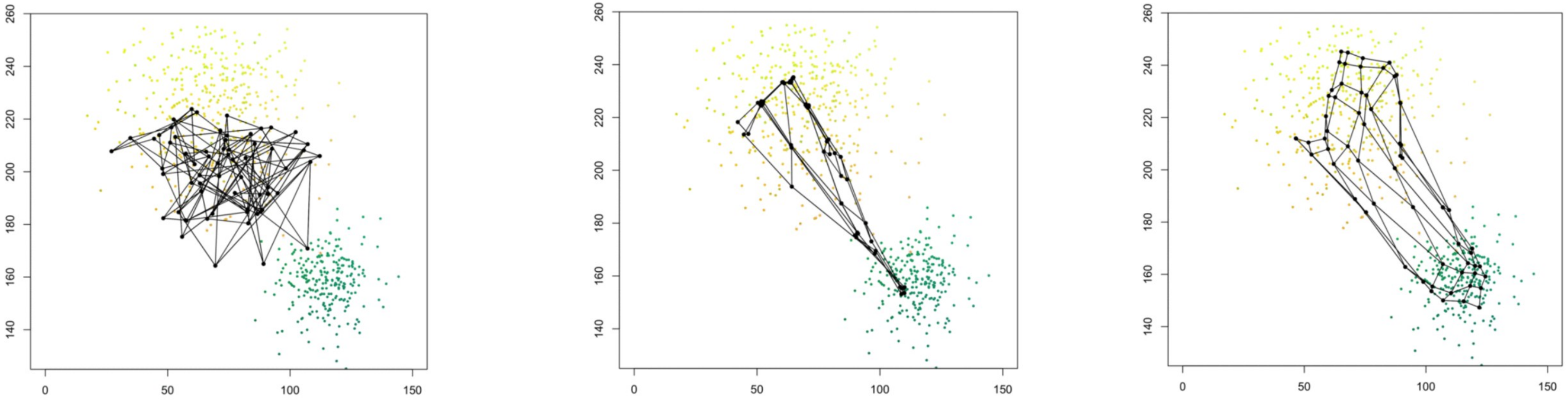
https://datascience.stackexchange.com/questions/32023/about-neural-network-called-self-organizing-maps/32115

https://datascience.stackexchange.com/questions/32599/why-self-organizing-map-som-units-are-called-neurons/32656

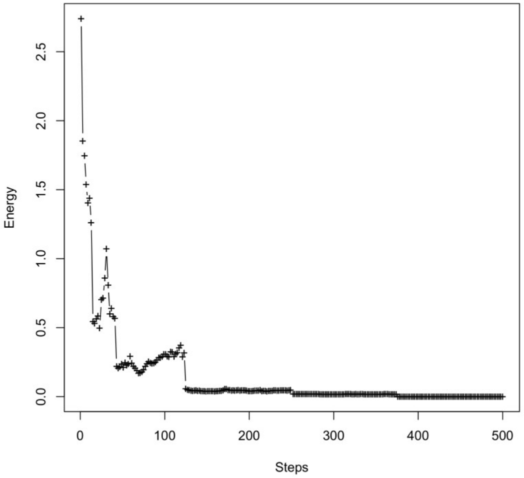
https://ai.stackexchange.com/questions/15624/what-is-the-impact-of-using-multiple-bmus-for-self-organizing-maps

Self-Organizing Maps

Initially, neurons in the SOM grid start out in random positions, but they gradually move into a mould outlining the shape of the data.



To check that the algorithm has converged, we can plot the evolution of the SOM’s energy



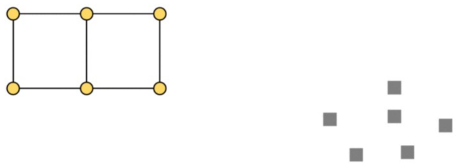
Initially, the SOM evolves rapidly, but as it reaches the approximate shape of the data, the rate of change slows down.

https://algobeans.com/2017/11/02/self-organizing-map/

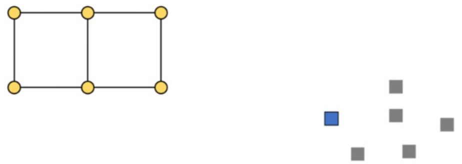
Self-Organizing Maps

https://algobeans.com/2017/11/02/self-organizing-map/

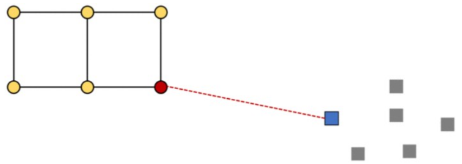
Step 0: Randomly position the grid’s neurons in the data space.



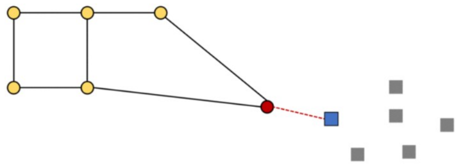
Step 1: Select one data point, either randomly or systematically cycling through the dataset in order



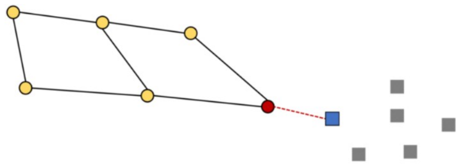
Step 2: Find the neuron that is closest to the chosen data point. This neuron is called the Best Matching Unit (BMU).



Step 3: Move the BMU closer to that data point. The distance moved by the BMU is determined by a learning rate, which decreases after each iteration.



Step 4: Move the BMU’s neighbors closer to that data point as well, with farther away neighbors moving less. Neighbors are identified using a radius around the BMU, and the value for this radius decreases after each iteration.



Step 5: Update the learning rate and BMU radius, before repeating Steps 1 to 4. Iterate these steps until positions of neurons have been stabilized.

Learning rate: $\gamma(t)=\gamma_0 e^{-t/a}$

BMU radius: $r(t)=r_0 e^{-t/d}$

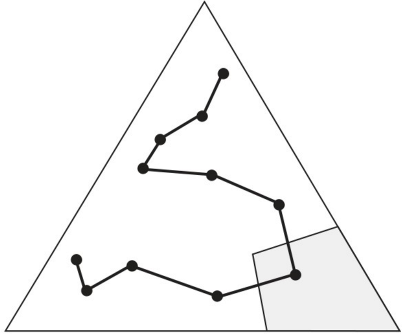
If values for both the learning rate and BMU radius are too high, neurons will be shoved around constantly without settling down.

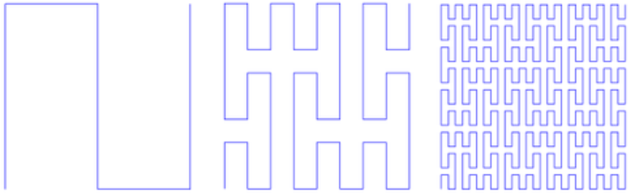
If these values are too low, the analysis will take too long as neurons inch towards their optimal positions.

https://datascience.stackexchange.com/questions/32031/neural-network-self-organizing-maps/32118

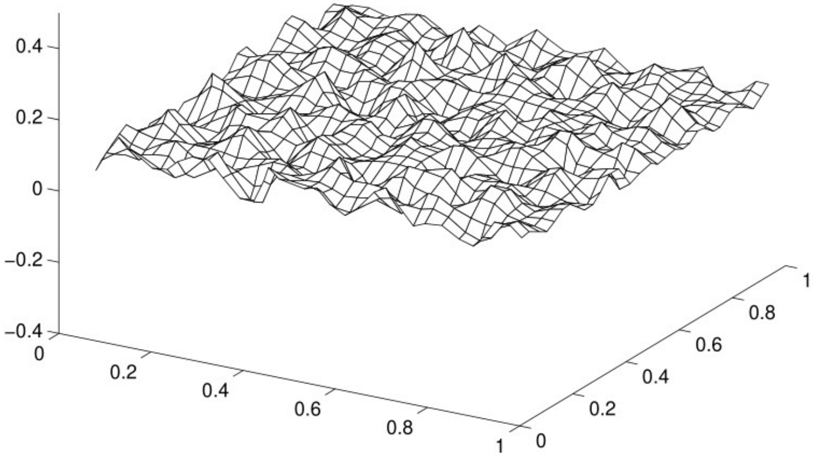
Another feature that we need to validate is the optimal number of neurons in the grid.

The chain of Kohonen units adopts the form of a Peano curve.





2D map of a 3D region



Curse of dimensionality

Richard Bellman (1961)

Many algorithms that work fine in low dimensions don't work when the input is high-*D*.

Generalizing correctly becomes harder as the number of features of the examples grows.

In high dimensional space the data becomes sparse.

A fixed-size training set covers a fraction of the input space.

larger dimensionality => smaller fraction

The noise from irrelevant features swamps the signal from the relevant features.

The similarity-based reasoning breaks down in high dimensions.

In high dimensions all examples look alike.

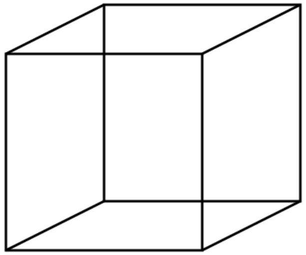
As the *D* increases, more and more examples become nearest neighbors of a certain example, until the choice of nearest neighbor is effectively random.

https://stats.stackexchange.com/questions/99171/why-is-euclidean-distance-not-a-good-metric-in-high-dimensions

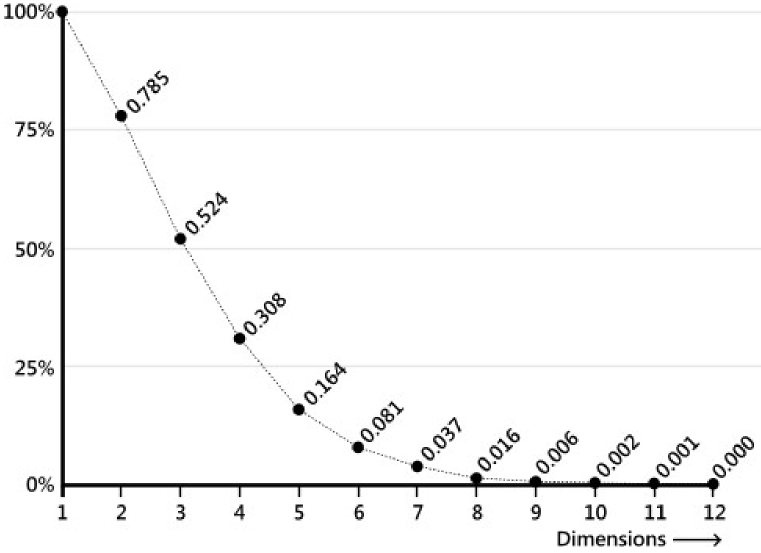
Pedro Domingos. A few useful things to know about machine learning. https://dl.acm.org/citation.cfm?id=2347755

Our intuitions, which come from a 3*D* world, often do not apply in high-*D*.

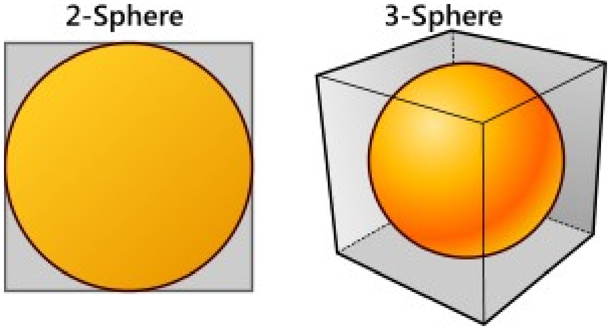
Most of the volume of a high-dimensional orange is in the skin, not the pulp.



If a constant number of examples is distributed uniformly in a high-D hypercube, beyond some dimensionality most examples are closer to a face of the hypercube than to their nearest neighbor.



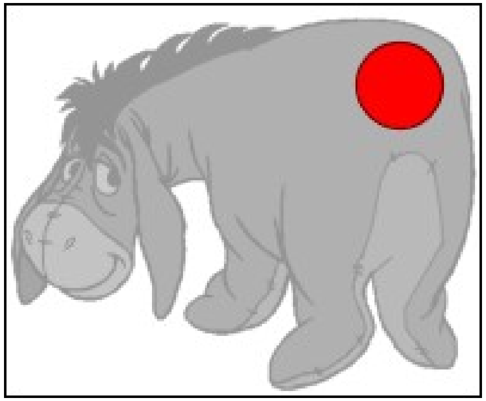
In high *D* almost all the volume of the hypercube is outside the hyper-sphere.



$$V_{2k}(R)=\frac{\pi^k}{k!}R^{2k}$$

$$V_{2k+1}(R)=\frac{2(k!)(4\pi)^k}{(2k+1)!}R^{2k+1}$$

It is difficult to pin the tail on the donkey in high-*D*



https://ieatbugsforbreakfast.wordpress.com/2011/07/31/on-getting-lucky-in-higher-dimensions/

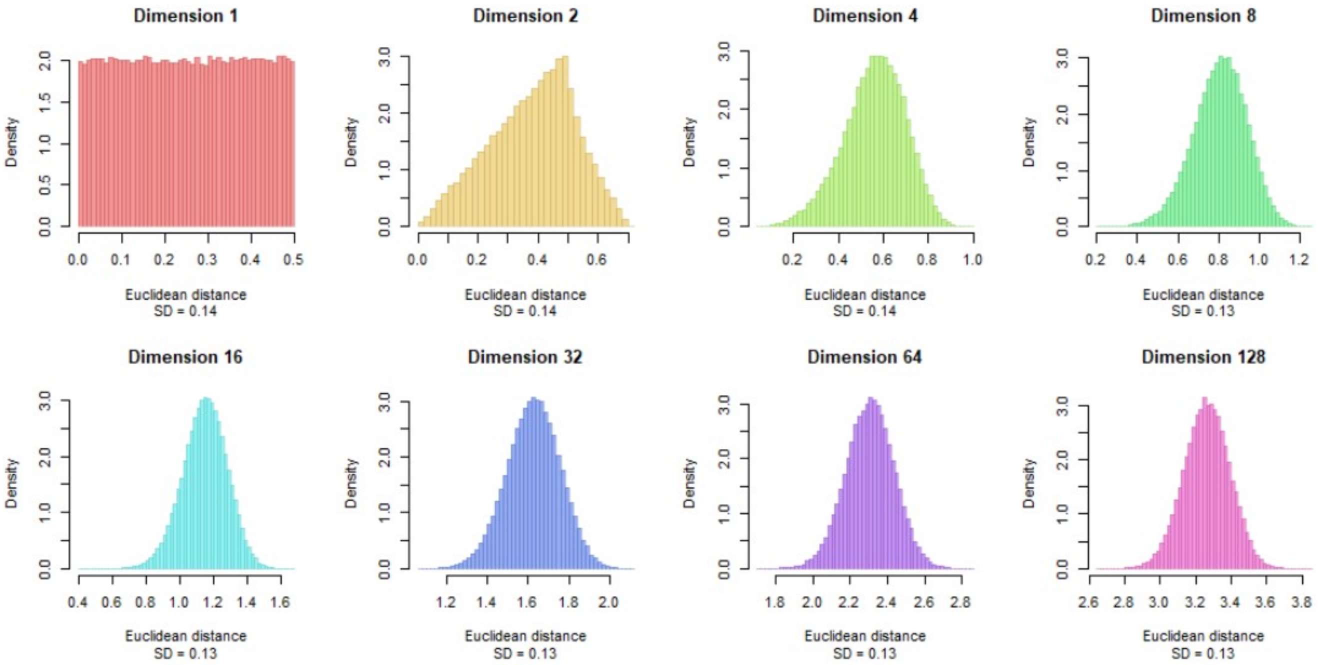
https://stats.stackexchange.com/questions/451027/mathematical-demonstration-of-the-distance-concentration-in-high-dimensions

d-torus: no boundaries => all points are geometrically equivalent.

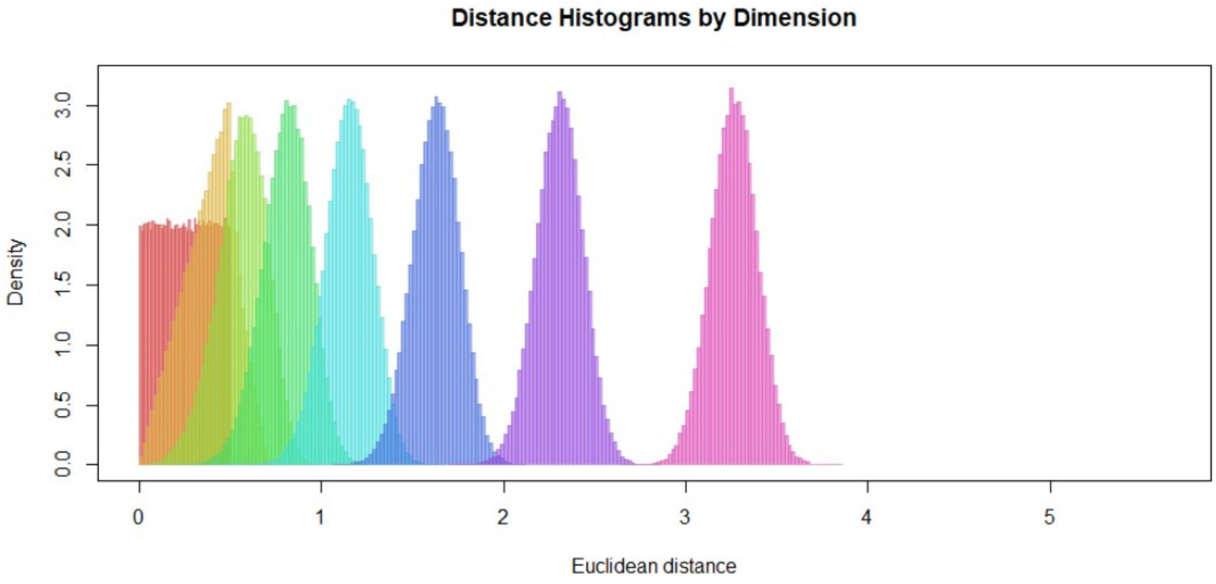
500 points

The distributions of distances tend to a Gaussian as the dimension increases.

The spreads of these histograms are nearly constant.



The average distances increase with dimension.



The greatest possible distance in the *d*-torus is achieved by pairs of points whose coordinates all differ by 1/2 (because you cannot get any further apart than that along a loop).

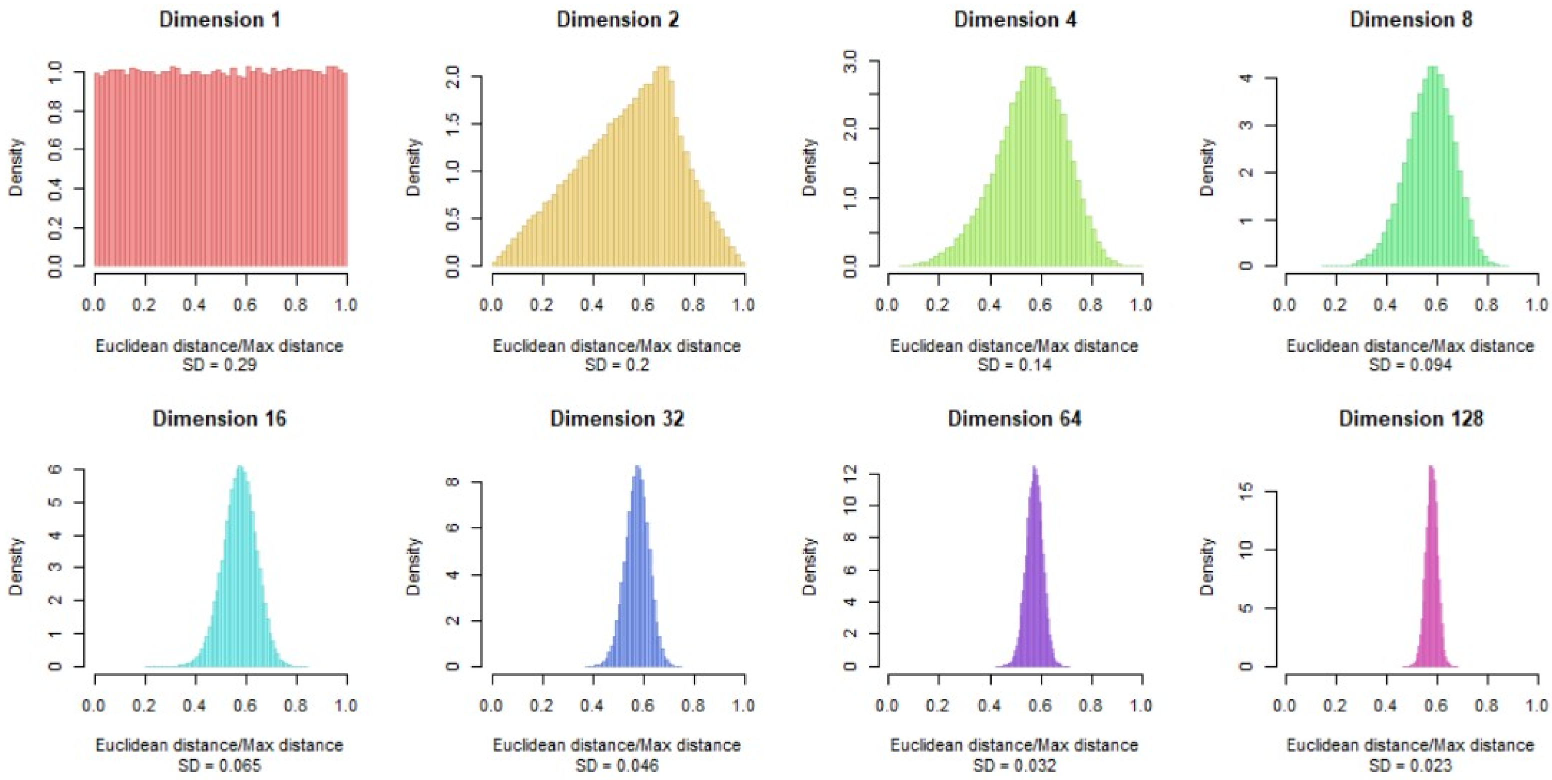
That distance is $\sqrt{(d)}/2$

https://stats.stackexchange.com/questions/451027/mathematical-demonstration-of-the-distance-concentration-in-high-dimensions

Compare the relative distances in each dimension.

The distances from the previous slide divided by $\sqrt{(d)}/2$

As the dimension increases the distances concentrate more closely around a central value.



Around any given point on a high-dimensional torus (and all points are geometrically the same, so it doesn't matter which point), nearly all other points on the torus are nearly the same distance away.

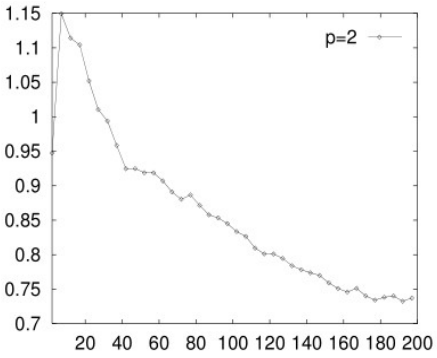
C. Aggarwal et al. On the Surprising Behavior of Distance Metrics in High Dimensional Space
https://link.springer.com/chapter/10.1007/3-540-44503-X_27

$$L_k(x_1, x_2) = \left[\sum_{i=1}^{d} (x_1^i - x_2^i)^k \right]^{1/k}$$

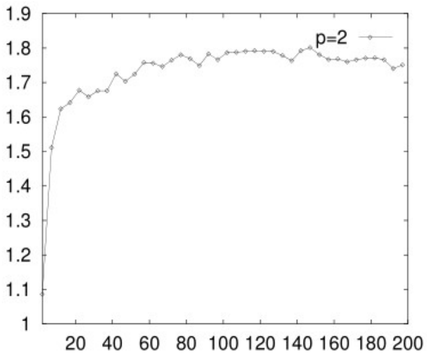
Let F be the arbitrary distribution of n points. Then,

$$C_k \leq \lim_{d \to \infty} E\left[\frac{Dmax_d^k - Dmin_d^k}{d^{1/k-1/2}} \right] \leq (n-1) C_k$$

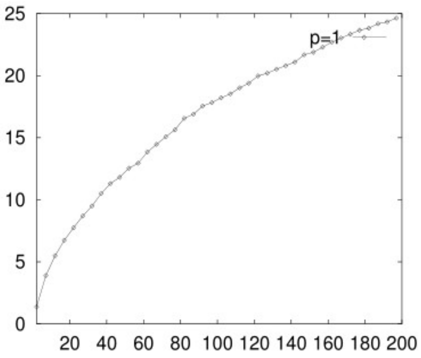
$$Dmax_d^k - Dmin_d^k \sim d^{1/k-1/2}$$



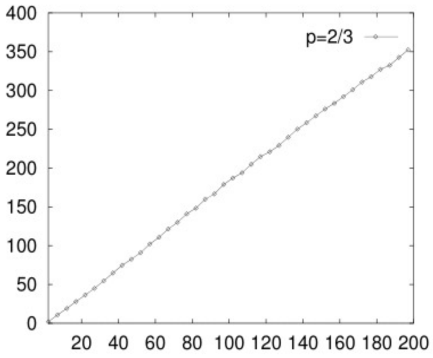
$k = 3$



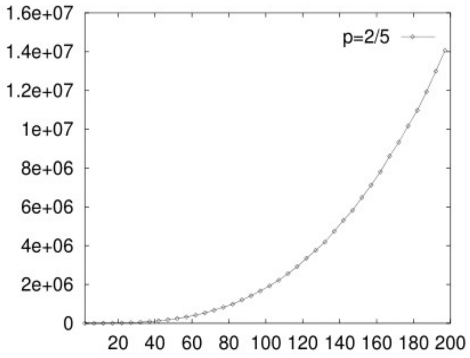
$k = 2$



$k = 1$



$k = 2/3$



$k = 2/5$

d – dimensionality of the data space

E – expected value

$Dmax_d^k$, $Dmin_d^k$ – farthest and nearest distances

n – number of data points

C_k – const., dependent of the data distribution

K. Hansen et al. Assessment and Validation of Machine Learning Methods for Predicting Molecular Atomization Energies
https://pubs.acs.org/doi/10.1021/ct400195d

Methods with Laplacian kernel perform better than methods with Gaussian kernel.

$$k(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|_1}{\sigma}\right)$$

$$k(x_1, x_2) = \exp\left(-\frac{\|x_1 - x_2\|_2^2}{2\sigma^2}\right)$$

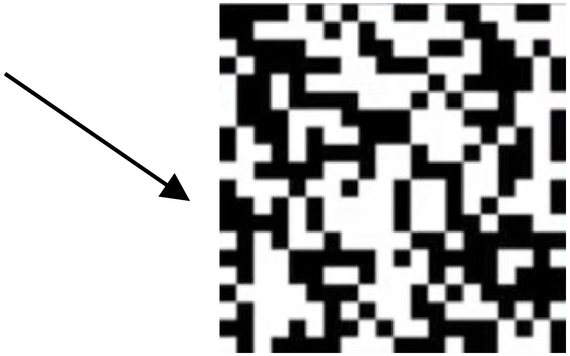
C. Aggarwal et al. On the Surprising Behavior of Distance Metrics in High Dimensional Space
https://link.springer.com/chapter/10.1007/3-540-44503-X_27

For dimensionalities of 20 or higher the manhattan distance metric provides a signicantly higher relative contrast than the Euclidean distancemetric with very high probability.

Blessing of non-uniformity

Data are not spread uniformly throughout the instance space, but are concentrated on or near a lower-dimensional manifold.

k-nearest neighbors works well for handwritten digit recognition even though images of digits have one dimension per pixel, because the space of digit images is much smaller than the space of all possible images.



not a digit

MNIST: 28 × 28 pixels = 784-dimensional

Intrinsic dimensionality estimated between 12 and 14

Matthias Hein, Jean Yves Audibert;Intrinsic dimensionality estimation of submanifolds in Rd (2005). ICML '05: Proceedings of the 22nd international conference on Machine learning 289–296 https://doi.org/10.1145/1102351.1102388

Jose A. Costa, Alfred O. Hero III; Determining Intrinsic Dimension and Entropy of High-Dimensional Shape Spaces (2006) Statistics and Analysis of Shapes pp 231-252 https://doi.org/10.1007/0-8176-4481-4_9

Granata, D., Carnevale, V. Accurate Estimation of the Intrinsic Dimension Using Graph Distances: Unraveling the Geometric Complexity of Datasets. Sci Rep 6, 31377 (2016). https://doi.org/10.1038/srep31377

Zimek, A., Schubert, E. and Kriegel, H.-P. (2012), A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analy Data Mining, 5: 363-387. https://doi.org/10.1002/sam.11161

For points that deviate in every attribute from the usual data distribution, the outlier characteristics just become even stronger and more pronounced with increasing dimensionality.

When the data follows a mixture of distributions, the concentration effect is not always observed. In such cases, distances between members of different distributions may not necessarily tend to the global mean as the dimensionality increases.

Useless attributes are one of the core of the problem of the curse.

Relevant additional dimensions can also increase the contrast.

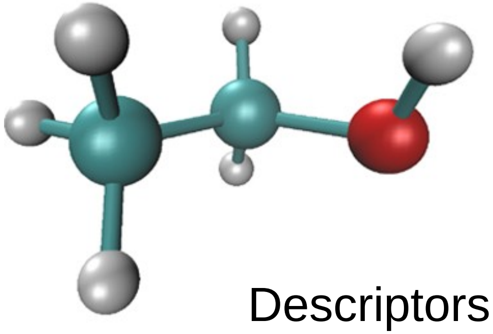
Correlated attributes will result in an intrinsic dimensionality that is considerably lower than the representational dimensionality.

Idea: Instead of a high-*D* point use multisets of low-*D* points

Questions:

How to divide a feature set into points?

Which metric to use?



Descriptors should be invariant w.r.t. permutations of the atoms

Shaohua Kevin Zhou, Rama Chellappa; From sample similarity to ensemble similarity: probabilistic distance measures in reproducing kernel Hilbert space. (2006) IEEE Trans. Pattern Anal. Mach. Intell. 28(6):917-29. doi: 10.1109/TPAMI.2006.120.

Veronika Cheplygina, David M.J. Tax, Marco Loog; On classification with bags, groups and sets. (2015)
Pattern Recognition Letters, V 59, 11-17, https://doi.org/10.1016/j.patrec.2015.03.008

Veronika Cheplygina, Dissimilarity-based multiple instance learning.
http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.852.1126&rep=rep1&type=pdf