Big Data Computing

Master's Degree in Computer Science

2024-2025

Gabriele Tolomei

Department of Computer Science

Sapienza Università di Roma

tolomei@di.uniroma1.it

SAPIENZA
UNIVERSITÀ DI ROMA

Similarity: Why Bother?

- We have seen a number of similiarity/distance metrics

Similarity: Why Bother?

- We have seen a number of similiarity/distance metrics
- Each metric may be suitable for specific task(s) in a particular domain

Similarity: Why Bother?

- We have seen a number of similiarity/distance metrics
- Each metric may be suitable for specific task(s) in a particular domain
- **Clustering** is one of these tasks!

CLUSTERING

What is Clustering?

- A procedure to group a set of objects into classes of **similar** objects

What is Clustering?

- A procedure to group a set of objects into classes of **similar** objects
- A standard problem in many (big) data applications:

What is Clustering?

- A procedure to group a set of objects into classes of **similar** objects
- A standard problem in many (big) data applications:
  - Categorizing documents by their topics
  - Grouping customers by their behaviors
  - …

What is Clustering?

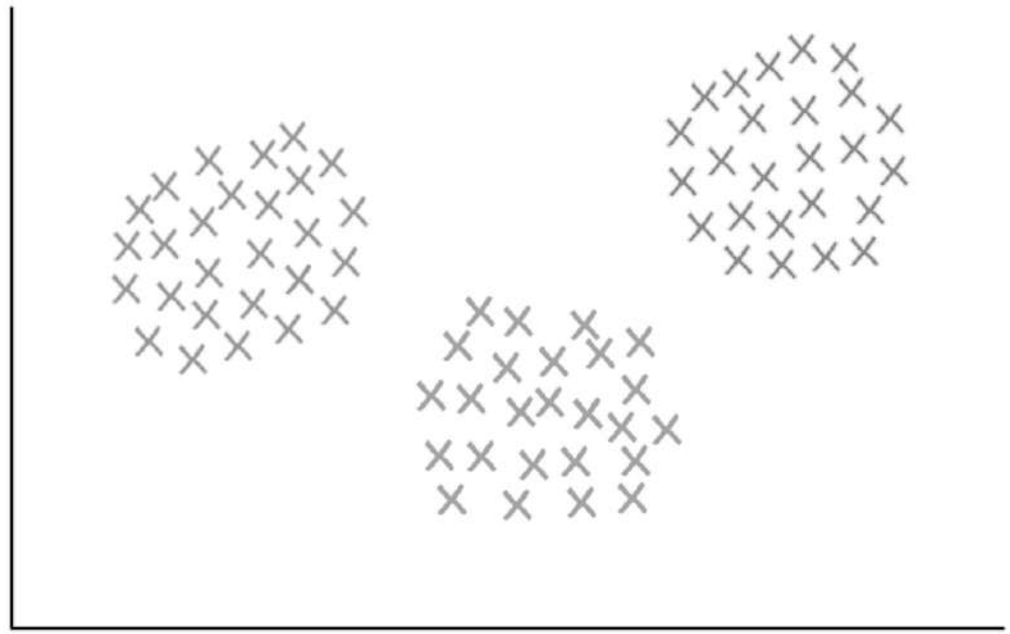
- A typical example of **unsupervised learning** technique

What is Clustering?

- A typical example of **unsupervised learning** technique
- A method of **data exploration**, i.e., a way of looking for patterns of interest in data

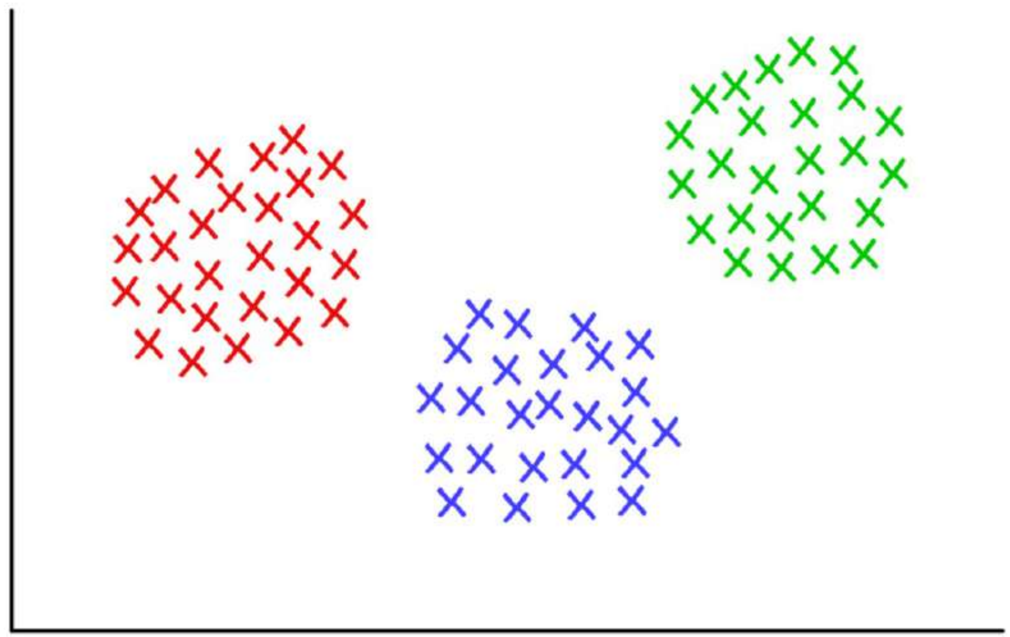
Clustering: Intuition

Given a set of 2-dimensional data points



Clustering: Intuition

We'd like to understand their "structure" to find groups of data points



Clustering: Formal Definition

- Given a set of data points and a notion of **distance** between those

Clustering: Formal Definition

- Given a set of data points and a notion of **distance** between those
- Group the data points into some number of clusters so that:

Clustering: Formal Definition

- Given a set of data points and a notion of **distance** between those
- Group the data points into some number of clusters so that:
  - Members of a cluster are close/similar to each other (i.e., **high intra-cluster similarity**)

Clustering: Formal Definition

- Given a set of data points and a notion of **distance** between those
- Group the data points into some number of clusters so that:
  - Members of a cluster are close/similar to each other (i.e., **high intra-cluster similarity**)
  - Members of different clusters are dissimilar (i.e., **low inter-cluster similarity**)

Clustering: Practical Issues

- Object **representation**
  - Data points may be in very high-dimensional spaces

Clustering: Practical Issues

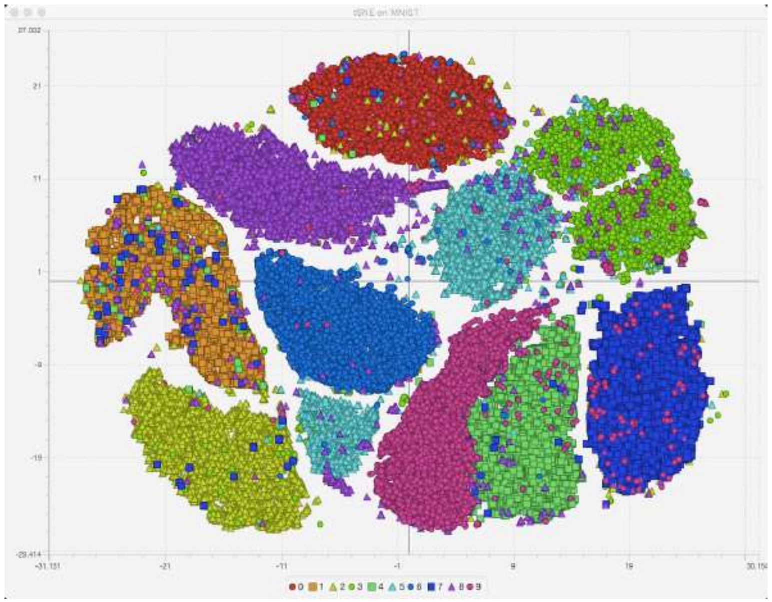
- Object **representation**
  - Data points may be in very high-dimensional spaces
- Notion of **similarity** between objects using a distance measure
  - Euclidean distance, Cosine similarity, Jaccard coefficient, etc.

Clustering: Practical Issues

- Object **representation**
  - Data points may be in very high-dimensional spaces
- Notion of **similarity** between objects using a distance measure
  - Euclidean distance, Cosine similarity, Jaccard coefficient, etc.
- Number of **output clusters**
  - Fixed apriori? Data-driven?

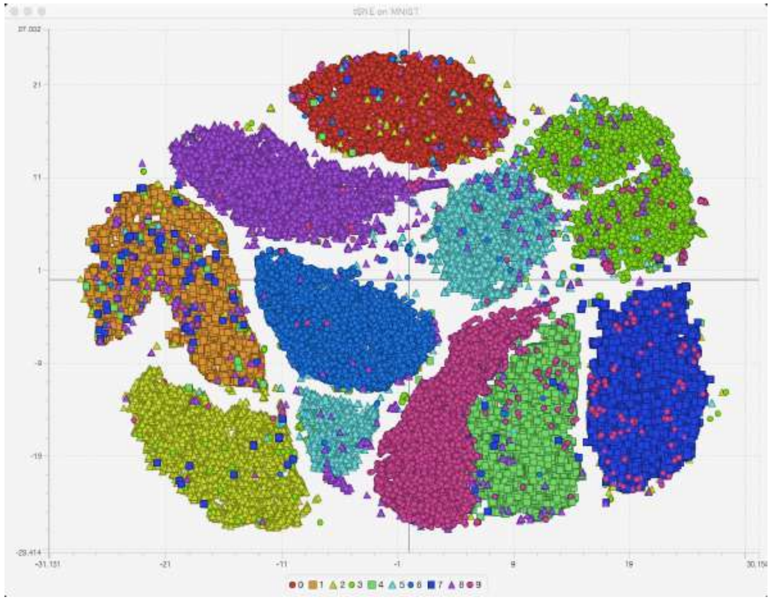
Clustering: A Hard Problem

Data points are not always easily and clearly separable



Clustering: A Hard Problem

Data points are not always easily and clearly separable



Finding a clear boundary between clusters may be hard in the real world

Clustering: A Hard Problem

- Clustering in 2 dimensions looks easy

Clustering: A Hard Problem

- Clustering in 2 dimensions looks easy
- So does clustering of a small number of data points

Clustering: A Hard Problem

- Clustering in 2 dimensions looks easy
- So does clustering of a small number of data points
- What does make things hard, then?

Clustering: A Hard Problem

- Clustering in 2 dimensions looks easy
- So does clustering of a small number of data points
- **What does make things hard, then?**
  - Many real-world applications involve 10s, 100s, or 1,000s of dimensions

Clustering: A Hard Problem

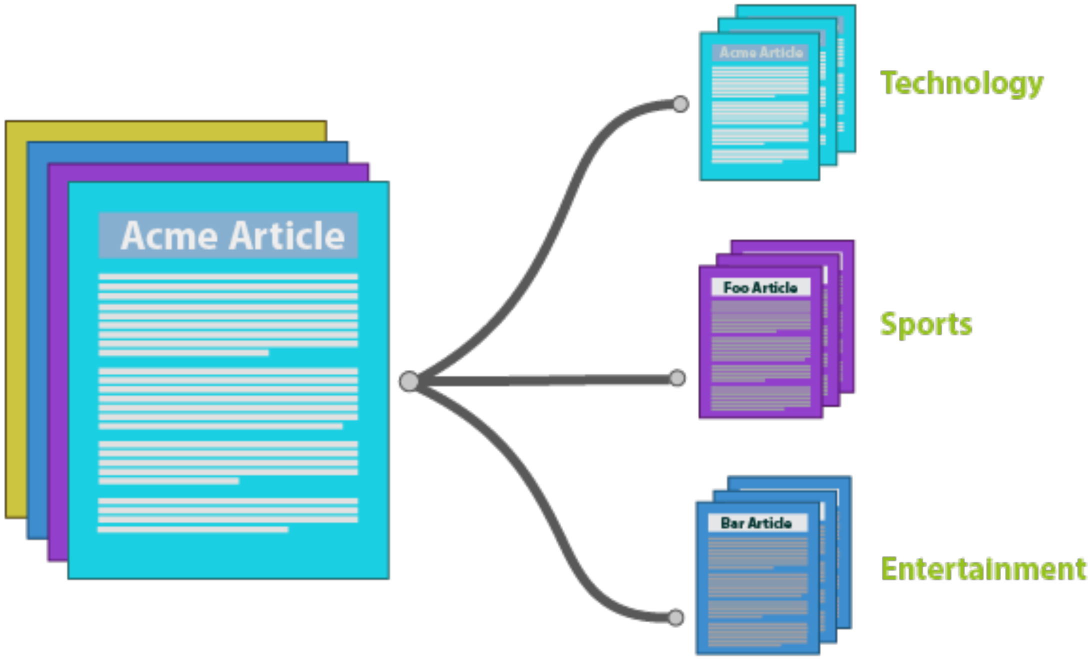
- Clustering in 2 dimensions looks easy
- So does clustering of a small number of data points
- **What does make things hard, then?**

Many real-world applications involve 10s, 100s, or 1,000s of dimensions

⇩

In high-dimensional spaces almost all pairs of points are at the same (large) distance

Example: Text Document Clustering



source: https://towardsdatascience.com/applying-machine-learning-to-classify-an-unsupervised-text-document-e7bb6265f52

Example: Text Document Clustering

- **Problem:** Group together documents on the same **topic**

Example: Text Document Clustering

- **Problem:** Group together documents on the same **topic**
- Documents with similar sets of words may be about the same topic

Example: Text Document Clustering

- **Problem:** Group together documents on the same **topic**
- Documents with similar sets of words may be about the same topic
- **Key Issues:**
  - Representing documents (in the space of words)

Example: Text Document Clustering

- **Problem:** Group together documents on the same **topic**
- Documents with similar sets of words may be about the same topic
- **Key Issues:**
  - Representing documents (in the space of words)
  - Measuring document similarity (in the space of words)

Document Representation

- Different ways of representing documents (in the space of words)

Document Representation

- Different ways of representing documents (in the space of words)
  - As a **set of words** (disregarding the order and multiplicity)

Document Representation

- Different ways of representing documents (in the space of words)
  - As a **set of words** (disregarding the order and multiplicity)
  - As a **bag-of-words** (i.e., a multiset disregarding the order yet keeping multiplicity)

Document Representation

- Different ways of representing documents (in the space of words)
  - As a **set of words** (disregarding the order and multiplicity)
  - As a **bag-of-words** (i.e., a multiset disregarding the order yet keeping multiplicity)
  - As a **bag-of-n-grams** (i.e., the more general case of bag-of-words)

Document Representation

- Different ways of representing documents (in the space of words)
  - As a **set of words** (disregarding the order and multiplicity)
  - As a **bag-of-words** (i.e., a multiset disregarding the order yet keeping multiplicity)
  - As a **bag-of-*n*-grams** (i.e., the more general case of bag-of-words)
  - More advanced representations derived from Neural Language Models (e.g., word2vec, BERT)

Document Representation

- Different ways of representing documents (in the space of words)
  - As a **set of words** (disregarding the order and multiplicity)
  - As a **bag-of-words** (i.e., a multiset disregarding the order yet keeping multiplicity)
  - As a **bag-of-*n*-grams** (i.e., the more general case of bag-of-words)
  - More advanced representations derived from Neural Language Models (e.g., word2vec, BERT)
- The choice of document representation affects the similarity measure

Document Representation: Set of Words

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

Document Representation: Set of Words

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

{John, likes, to, watch, movies, Mary, too}

{Mary, also, likes, to, watch, football, games}

Document Representation: Bag-of-Words

We keep multiplicity

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

Document Representation: Bag-of-Words

We keep multiplicity

doc 1

John likes to watch movies. Mary likes movies too.

{
John:1, likes:2, to:1, watch:1,
movies:2, Mary:1, too:1
}

doc 2

Mary also likes to watch football games.

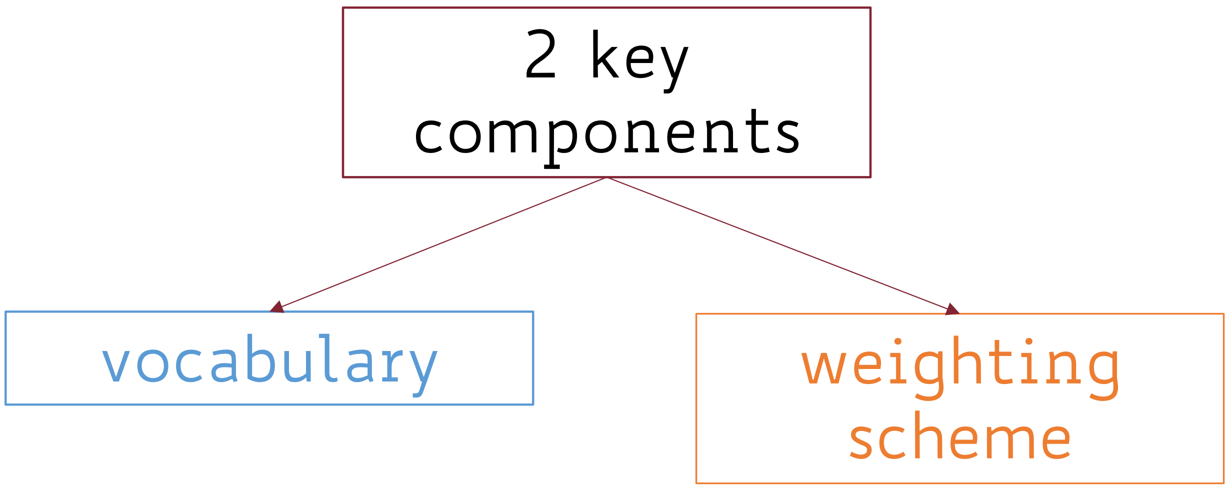
{
Mary:1, also:1, likes:1, to:1,
watch:1, football:1, games:1
}

Document Representation: Bag-of-Words

Bag-of-Words (BoW) model is just a preliminary step for more complex document representations

Document Representation: Bag-of-Words

Bag-of-Words (BoW) model is just a preliminary step for more complex document representations



Bag-of-Words:Vocabulary

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

Bag-of-Words:Vocabulary

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

V = {also, football, games, John, likes, Mary, movies, to, too, watch}

Bag-of-Words: Weighting Scheme

doc 1

John likes to watch movies. Mary likes movies too.

A |V|-dimensional vector, where the i-th component indicates the multiplicity of the i-th word of the vocabulary

V = {also, football, games, John, likes, Mary, movies, to, too, watch}

Bag-of-Words: Weighting Scheme

doc 1

John likes to watch movies. Mary likes movies too.

A |V|-dimensional vector, where the i-th component indicates the multiplicity of the i-th word of the vocabulary

(0, 0, 0, 1, 2, 1, 2, 1, 1, 1)

V = {also, football, games, John, likes, Mary, movies, to, too, watch}

Bag-of-Words: Weighting Scheme

doc 2

Mary also likes to watch football games.

A |V|-dimensional vector, where the i-th component indicates the multiplicity of the i-th word of the vocabulary

V = {also, football, games, John, likes, Mary, movies, to, too, watch}

Bag-of-Words: Weighting Scheme

doc 2

Mary also likes to watch football games.

A |V|-dimensional vector, where the i-th component indicates the multiplicity of the i-th word of the vocabulary

(1, 1, 1, 0, 1, 1, 0, 1, 0, 1)

V = {also, football, games, John, likes, Mary, movies, to, too, watch}

Bag-of-Words: A Formal Perspective

$D = \{d_1, \ldots, d_N\} =$ collection of N documents

$V = \{w_1, \ldots, w_{|V|}\} =$ **vocabulary** of $|V|$ words extracted from D

$\mathbf{d}_i = (f(w_1, i), \ldots, f(w_{|V|}, i)) = |V|$-dimensional vector representing d_i

$f : V \times D \longmapsto \mathbb{R}$ is a function that maps each word of a document to a real value (**weighting scheme**)

Bag-of-Words: A Formal Perspective

$\boxed{\text{One-Hot (binary) weighting scheme}}$

$$f(w_j, i) = \begin{cases} 1 & \text{if } w_j \text{ appears in } d_i \\ 0 & \text{otherwise} \end{cases}$$

Bag-of-Words: A Formal Perspective

Term-Frequency weighting scheme

$$f(w_j, i) = tf(w_j, i)$$

tf computes the number of times word w_j occurs in document d_i

Bag-of-Words: A Formal Perspective

TF-IDF weighting scheme

$$f(w_j, i) = tf(w_j, i) * idf(w_j)$$

$$idf(w_j) = \log\left(\frac{N}{n_j}\right)$$

n_j is the number of documents in D containing the word w_j

Bag-of-Words: A Formal Perspective

TF-IDF weighting scheme

$$f(w_j, i) = tf(w_j, i) * idf(w_j)$$

$$idf(w_j) = \log\left(\frac{N \boxed{+1}}{n_j \boxed{+1}}\right)$$

Any idea why?

n_j is the number of documents in D containing the word w_j

BoW: Limitations and Improvements

- **2 main limitations** of BoW model:
  - High dimensionality → sparseness
  - No sequential information nor semantics included → unigram model

BoW: Limitations and Improvements

- **2 main limitations** of BoW model:
  - High dimensionality → sparseness
  - No sequential information nor semantics included → unigram model
- Possible improvements:
  - Use *n*-grams rather than unigrams to capture sequentiality between consecutive words (i.e., context)
  - Even better, use so-called Neural Language Models like word2vec and BERT

Bag-of-*n*-grams

Example: bigrams (n=2)

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

Bag-of-*n*-grams

Example: bigrams (n=2)

doc 1

John likes to watch movies. Mary likes movies too.

doc 2

Mary also likes to watch football games.

{"John likes", "likes to", "to watch",
"watch movies", "Mary likes",
"likes movies", "movies too"}

{"Mary also", "also likes", "likes to",
"to watch", "watch football", "football games"}

Document Similarity

- We have examined a number of possible document representations

Document Similarity

- We have examined a number of possible document representations
- Depending on those, several similarity measures can be used

Document Similarity

- We have examined a number of possible document representations
- Depending on those, several similarity measures can be used
- For example, if documents are represented as:
  - set of words → Jaccard coefficient
  - one-hot bag-of-words → Euclidean distance
  - tf or tf-idf bag-of-words → Cosine similarity

High-Dimensional Spaces

- It's easy to get to very high-dimensional spaces

High-Dimensional Spaces

- It's easy to get to very high-dimensional spaces
- In the word space, the size of the vocabulary can be very high!

High-Dimensional Spaces

- It’s easy to get to very high-dimensional spaces
- In the word space, the size of the vocabulary can be very high!
- Moreover, only few dimensions are non-zero

High-Dimensional Spaces

- It's easy to get to very high-dimensional spaces
- In the word space, the size of the vocabulary can be very high!
- Moreover, only few dimensions are non-zero
- Other domains like images, audio, etc. suffer from the same issue

High-Dimensional Spaces

- Data in a high-dimensional space tends to be **sparser** than in lower dimensions

High-Dimensional Spaces

- Data in a high-dimensional space tends to be **sparser** than in lower dimensions
- Data points are **more dissimilar** to each other

High-Dimensional Spaces

- In Euclidean space, the distance between two points is large as long as they are far apart along **at least one** dimension

High-Dimensional Spaces

- In Euclidean space, the distance between two points is large as long as they are far apart along **at least one** dimension
- The higher the number of dimensions the higher the chance this happens

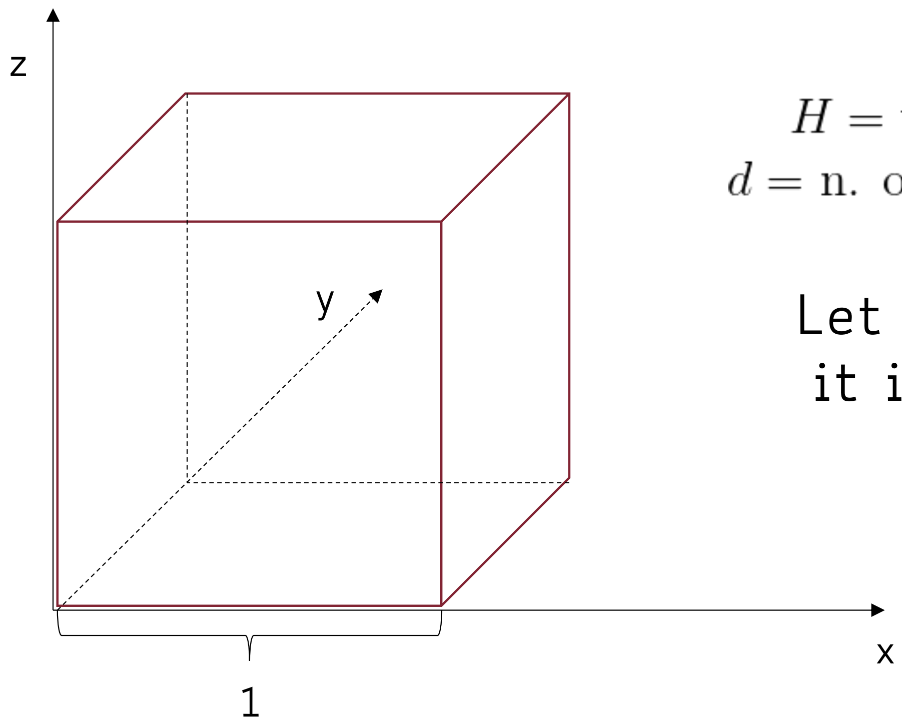
High-Dimensional Spaces

- In Euclidean space, the distance between two points is large as long as they are far apart along **at least one** dimension
- The higher the number of dimensions the higher the chance this happens

⇩

The Curse of Dimensionality

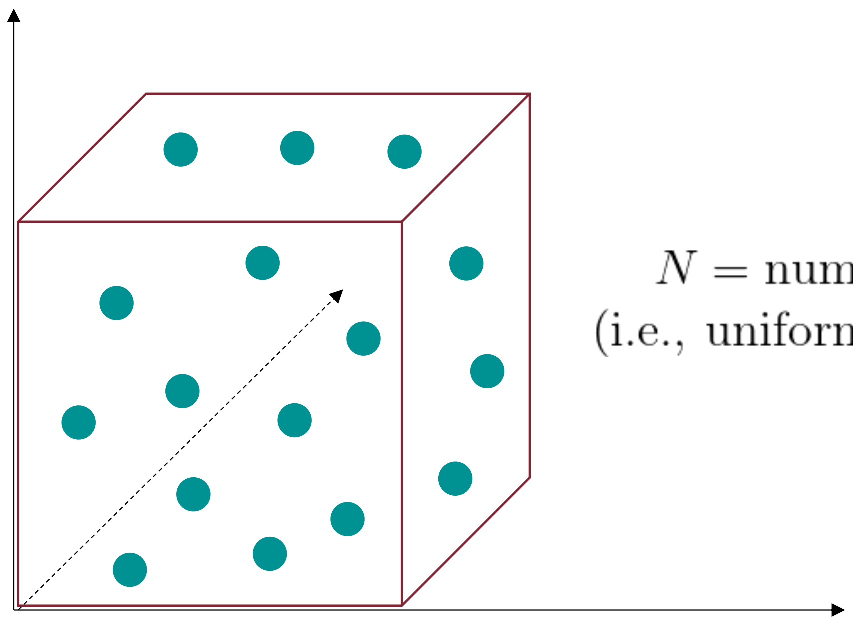
The Curse of Dimensionality



$$H = \text{unit-length hypercube in } \mathbb{R}^d$$
$$d = \text{n. of space dimensions}$$

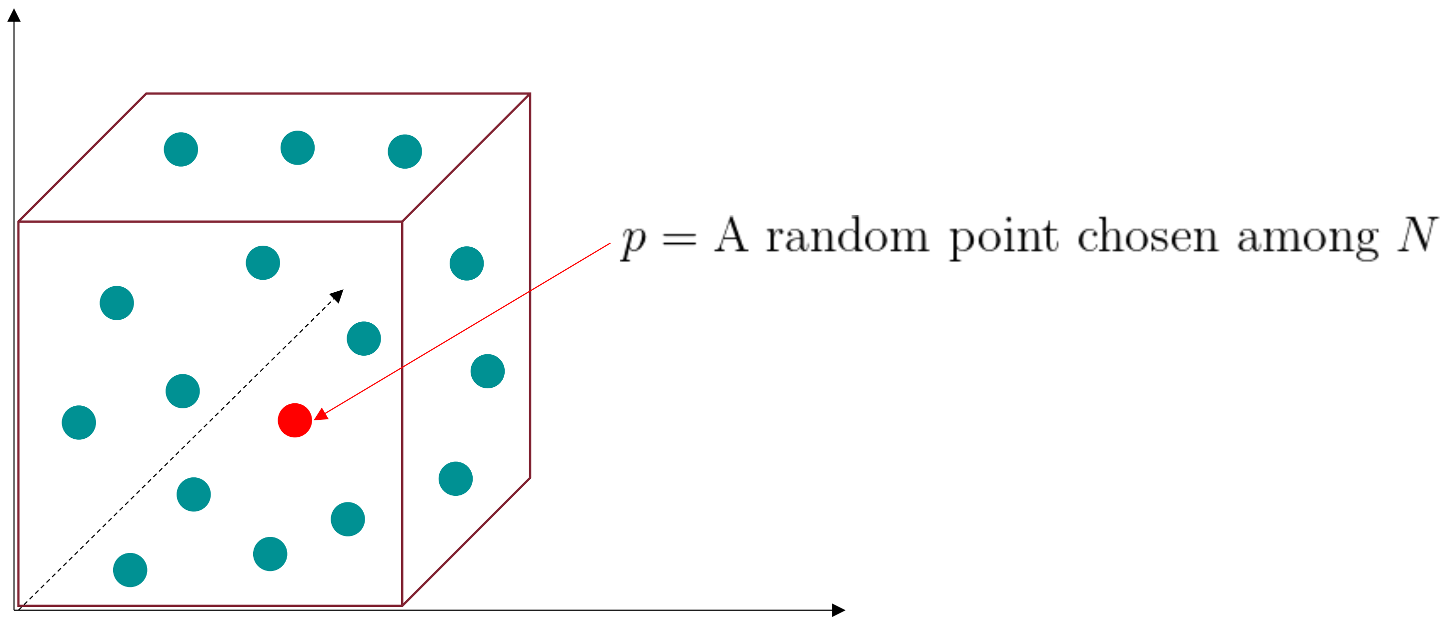
Let d=3 as beyond that it is hard to visualize the space

The Curse of Dimensionality

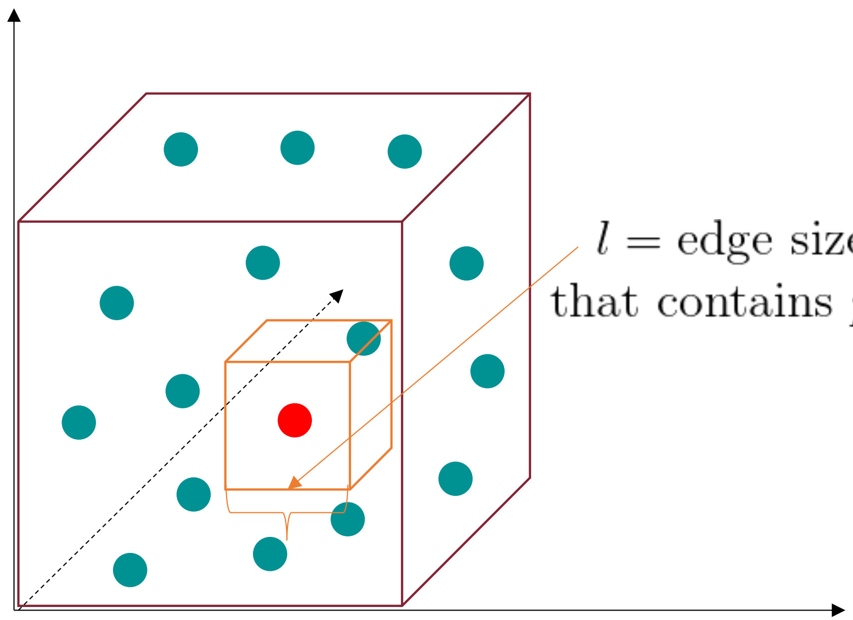


N = number of data points randomly (i.e., uniformly) distributed in H

The Curse of Dimensionality

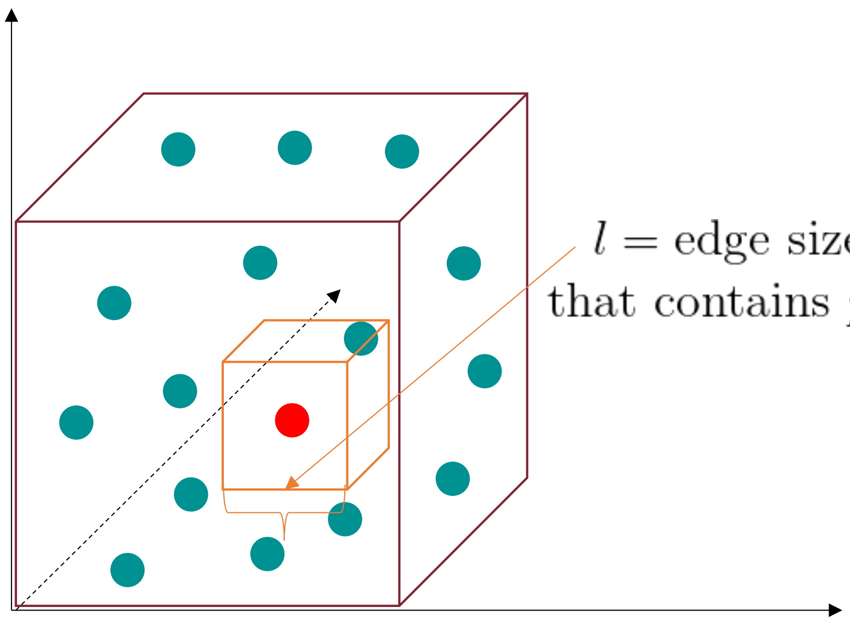


The Curse of Dimensionality



l = edge size of the hypercube h inscribed in H that contains p along with its k nearest points

The Curse of Dimensionality

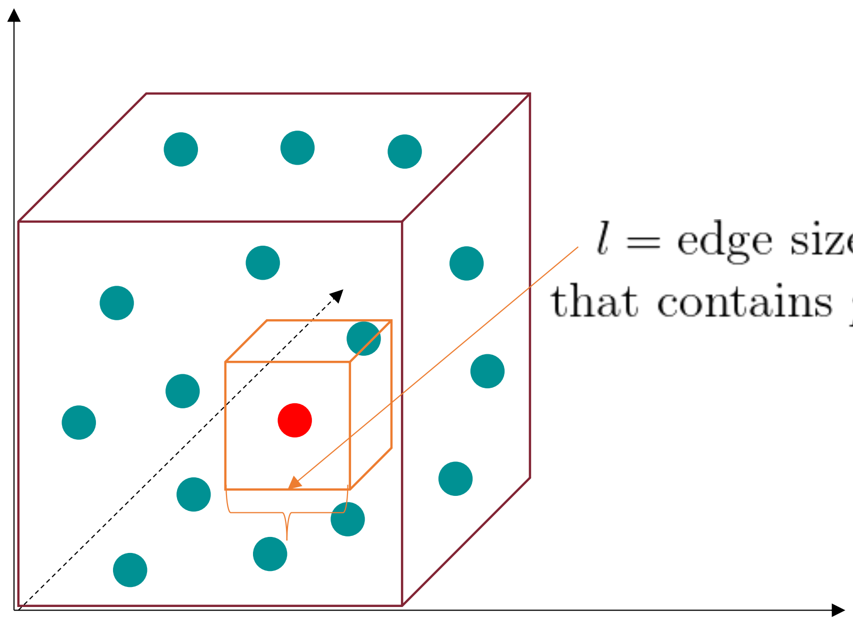


l = edge size of the hypercube h inscribed in H that contains p along with its k nearest points

We consider **edge points** whose distance from p is **at most** $\frac{l}{2}\sqrt{d}$

$$\boxed{\frac{l}{2}\sqrt{3}}$$

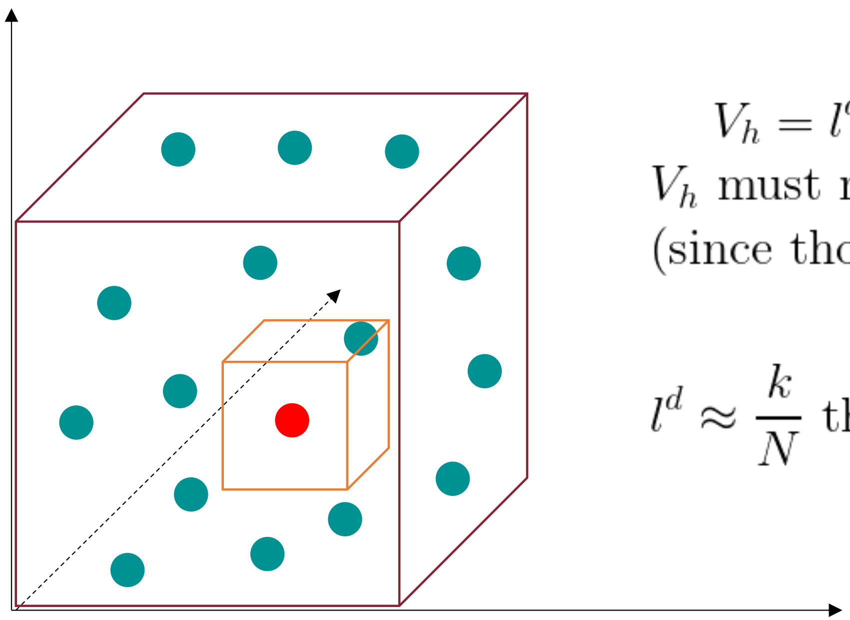
The Curse of Dimensionality



l = edge size of the hypercube h inscribed in H that contains p along with its k nearest points

The same question can be formulated in terms of the radius l of an inscribed hypersphere

The Curse of Dimensionality



$V_h = l^d$ volume of the hypercube h

V_h must roughly contain k/N points

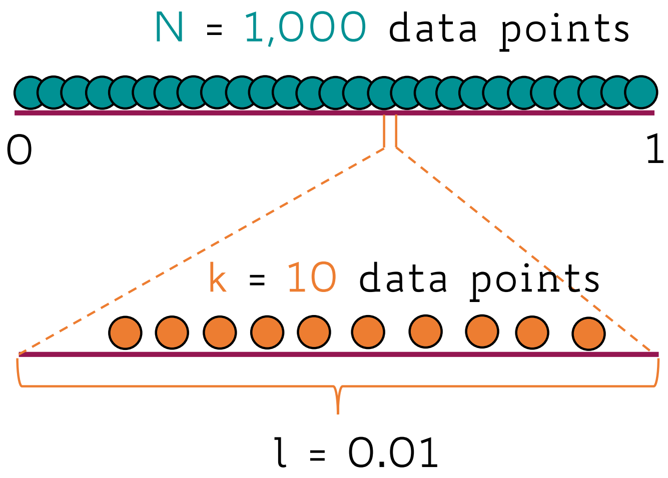
(since those are randomly distributed)

$$l^d \approx \frac{k}{N} \text{ therefore } l \approx \left(\frac{k}{N}\right)^{1/d}$$

The Curse of Dimensionality

A few numbers… $N = 1,000; k = 10 \quad l \approx \left(\frac{10}{1000}\right)^{1/d} = \left(\frac{1}{100}\right)^{1/d}$

d	l
1	0.01

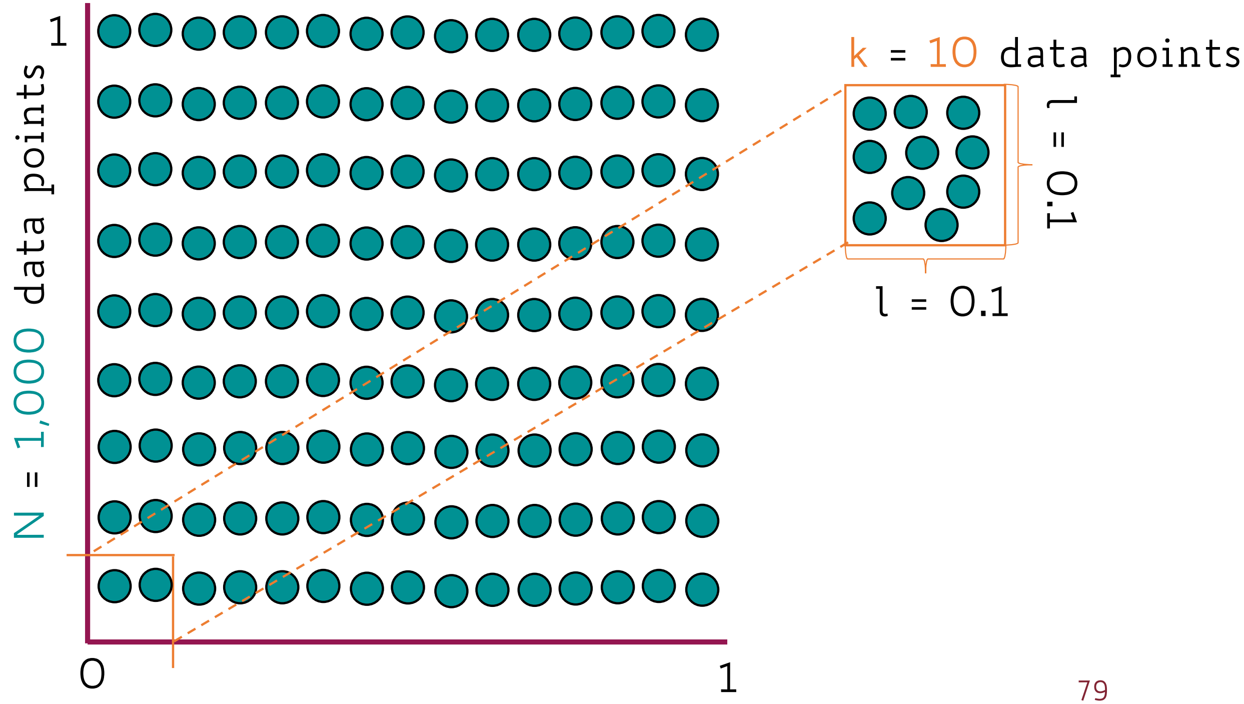


The Curse of Dimensionality

A few numbers…

$$N = 1{,}000; k = 10 \quad l \approx \left(\frac{10}{1000}\right)^{1/d} = \left(\frac{1}{100}\right)^{1/d}$$

d	l
1	0.01
2	0.1



The Curse of Dimensionality

A few numbers... $N = 1,000; k = 10 \quad l \approx \left(\frac{10}{1000}\right)^{1/d} = \left(\frac{1}{100}\right)^{1/d}$

d	l
1	0.01
2	0.1
3	0.215
...	...
10	0.631

When d is equal 10 the length of the edge of the inscribed hypercube is already about 63% of the largest hypercube

The Curse of Dimensionality

A few numbers...

$$N = 1,000; k = 10 \quad l \approx \left(\frac{10}{1000}\right)^{1/d} = \left(\frac{1}{100}\right)^{1/d}$$

d	l
1	0.01
2	0.1
3	0.215
...	...
10	0.631
...	...
1000	0.995

When d is equal 1,000 there is basically no difference between the two hypercubes!

The Curse of Dimensionality: Why Bother?

- Points are more likely to be located at the edges of the region

The Curse of Dimensionality: Why Bother?

- Points are more likely to be located at the edges of the region
- Nearest points are not close at all!

The Curse of Dimensionality: Why Bother?

- Points are more likely to be located at the edges of the region
- Nearest points are not close at all!
- Distance between points indistinguishable (**distance concentration**)
  - Hard to separate between nearest and furthest data points
  - Hard to find clusters among so many pairs that are all at approximately the same distance

The Curse of Dimensionality: The Edge

$\boxed{\text{Let } \varepsilon \text{ define the edge (i.e., border) of our space}}$

The Curse of Dimensionality: The Edge

Let ε define the edge (i.e., border) of our space

See how the probability of picking a data point that is not located at the edge changes as the number of dimensions grow

The Curse of Dimensionality: The Edge

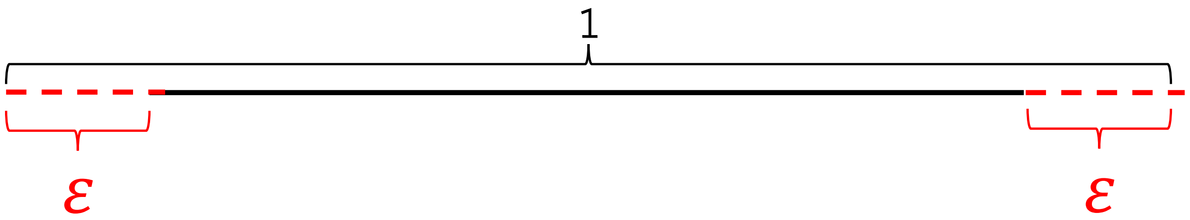
Let ε define the edge (i.e., border) of our space

See how the probability of picking a data point that is not located at the edge changes as the number of dimensions grow

Remember:
We assume data points are uniformly distributed at random on the space

The Curse of Dimensionality: The Edge

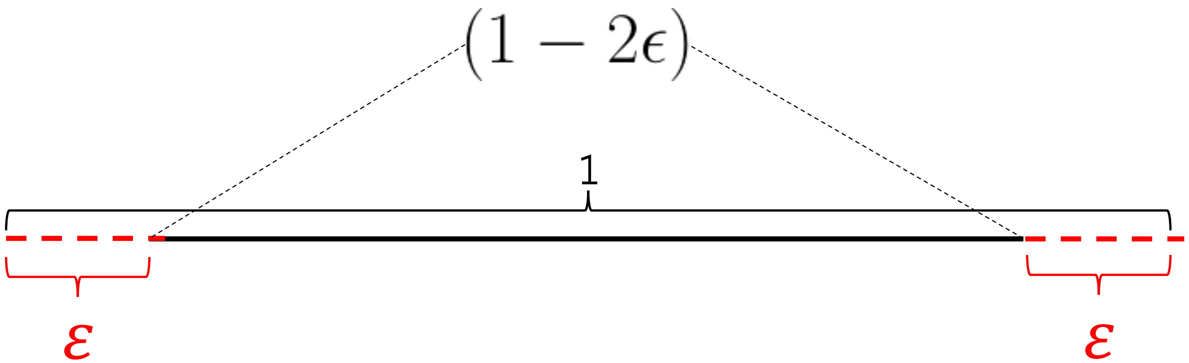
d = 1



The Curse of Dimensionality: The Edge

d = 1

The probability of being not at the edge is just



The Curse of Dimensionality: The Edge

$d > 1$

The probability of being not at the edge is the probability of being not at the edge on every single dimension

The Curse of Dimensionality: The Edge

d > 1

The probability of being not at the edge is the probability of being not at the edge on every single dimension

$$(1 - 2\epsilon)^d$$

assuming each dimension is independent from each other

The Curse of Dimensionality: The Edge

d > 1

The probability of being not at the edge is the probability of being not at the edge on every single dimension

$$(1 - 2\epsilon)^d$$

assuming each dimension is independent from each other

$$\lim_{d \to \infty} (1 - 2\epsilon)^d = 0$$

The Curse of Dimensionality

A Notebook where the Curse of Dimensionality is (visually) explained is available at the following link:

https://github.com/gtolomei/big-data-computing/blob/master/notebooks/The_Curse_of_Dimensionality.ipynb

So What Can We Do?

- If data are really uniformly distributed in a high-dimensional space… nothing!

So What Can We Do?

- If data are really uniformly distributed in a high-dimensional space… nothing!
- Luckily, though, real-world (interesting) data have patterns underneath (i.e., they are **not random**!)

So What Can We Do?

- If data are really uniformly distributed in a high-dimensional space… nothing!
- Luckily, though, real-world (interesting) data have patterns underneath (i.e., they are **not random**!)
- Lower intrinsic dimensionality

So What Can We Do?

The Manifold Hypothesis

- High dimensional data (e.g., images) lie on low-dimensional manifolds (i.e., sub-space) embedded in the high-dimensional space

So What Can We Do?

The Manifold Hypothesis

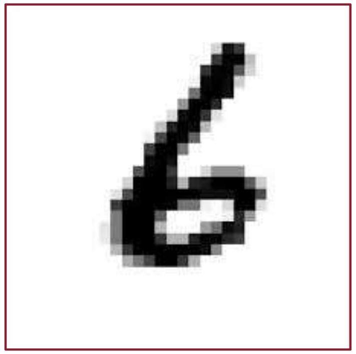
- High dimensional data (e.g., images) lie on low-dimensional manifolds (i.e., sub-space) embedded in the high-dimensional space
- Dimensionality reduction techniques (more on this later…)

Modeled vs. True Dimensionality

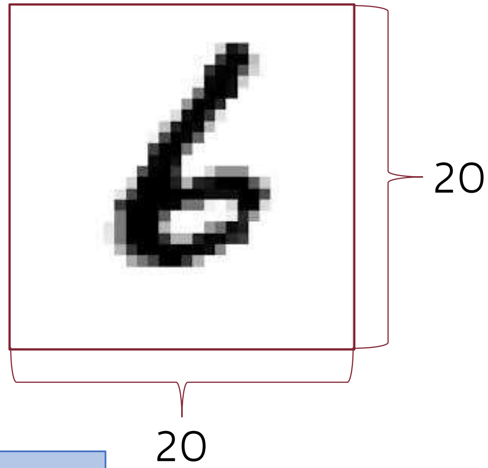
Example

Handwritten digit recognition

Modeled vs. True Dimensionality



Modeled vs. True Dimensionality

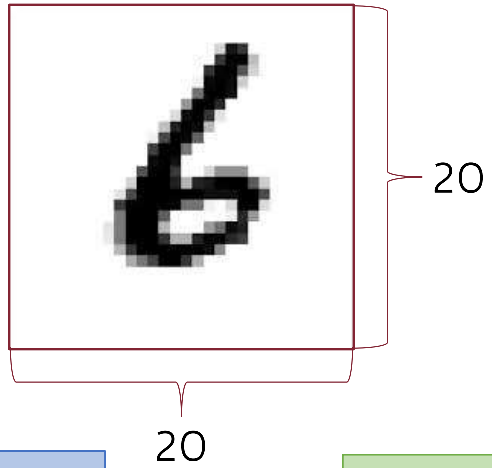


Modeled dimensionality

Each digit represented by
20x20 bitmap

400-dimensional binary vector

Modeled vs. True Dimensionality



Modeled dimensionality

Each digit represented by 20x20 bitmap

400-dimensional binary vector

True dimensionality

Actual digits just cover a tiny fraction of all this huge space

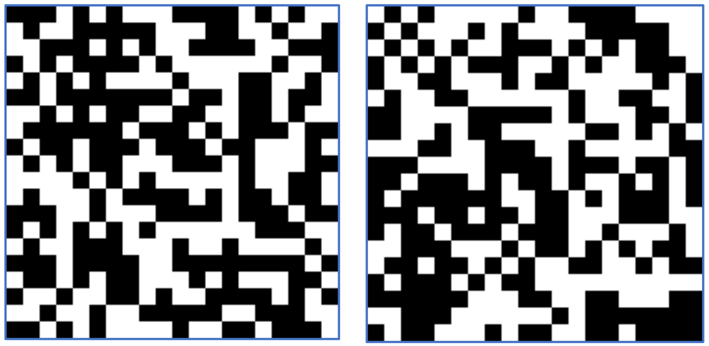
Small variations of the pen-stroke

Modeled vs. True Dimensionality

Random samples from 400-d space

Modeled vs. True Dimensionality

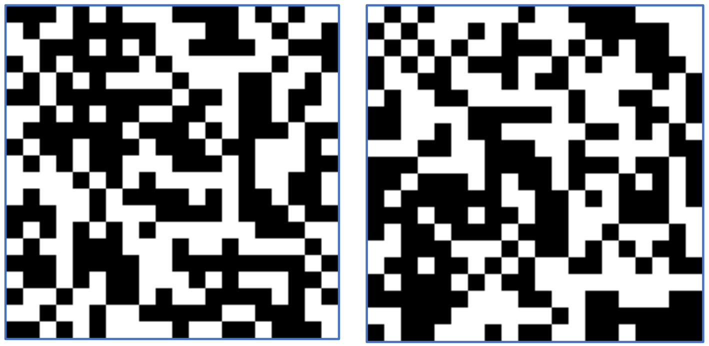
Random samples from 400-d space



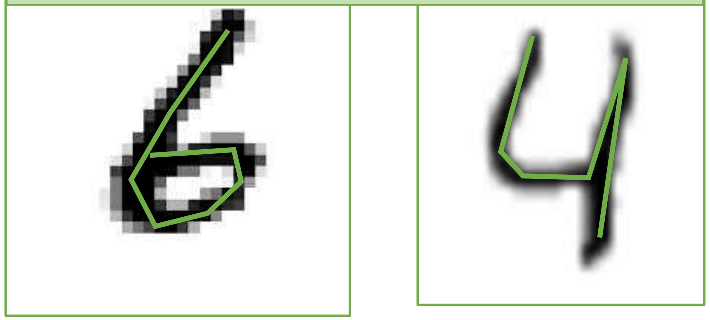
True digits living in a 400-d space

Modeled vs. True Dimensionality

Random samples from 400-d space



True digits living in a 400-d space



We model data (i.e., digits) as very high dimensional…

… In fact, they are not so

Take-Home Message of Today

- Many big data tasks are based on finding (hidden) commonalities between input data points

Take-Home Message of Today

- Many big data tasks are based on finding (hidden) commonalities between input data points
- For example, clustering is an unsupervised learning technique to group "similar" objects together

Take-Home Message of Today

- Many big data tasks are based on finding (hidden) commonalities between input data points
- For example, clustering is an unsupervised learning technique to group "similar" objects together
- Depends on:
  - **object representation**
  - **similarity measure**

Take-Home Message of Today

- In the Euclidean space, when data dimensionality gets large, similarity/distance becomes meaningless!

Take-Home Message of Today

- In the Euclidean space, when data dimensionality gets large, similarity/distance becomes meaningless!
- Any set of random data points are far away from each other → **curse of dimensionality**

Take-Home Message of Today

- In the Euclidean space, when data dimensionality gets large, similarity/distance becomes meaningless!
- Any set of random data points are far away from each other → **curse of dimensionality**
- Luckily, real-data may live in lower-dimensional spaces