Dimensionality Reduction

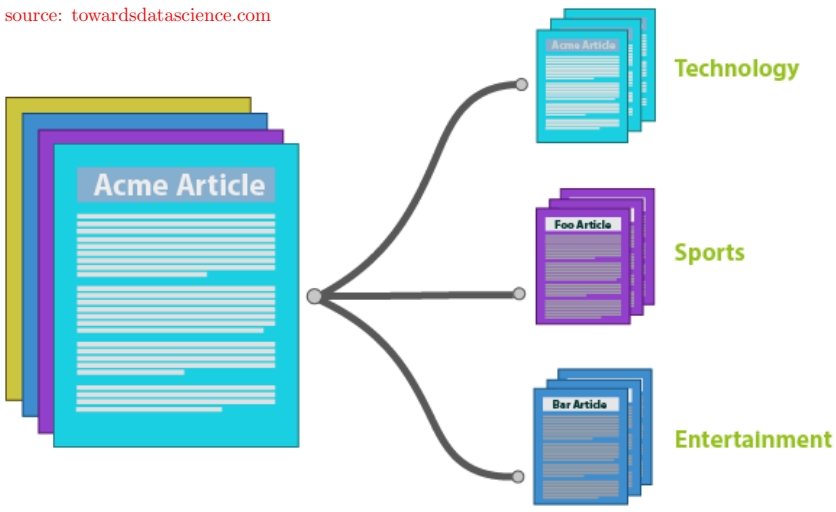
Random Projection and Johnson-Lindenstrauss Lemma

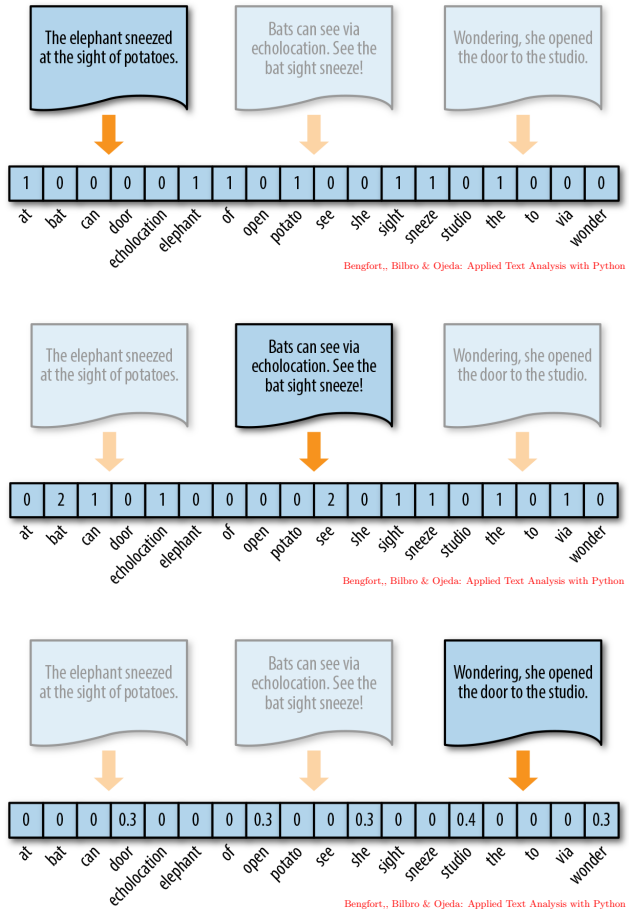
- Imdadullah Khan

High Dimensional Data

High Dimensional Data is common in many applications

- Text represented as set or bag or TF-IDF of words
- 1000's of unigram, millions of bigrams plus contextual attributes





High Dimensional Data

High Dimensional Data is common in many applications

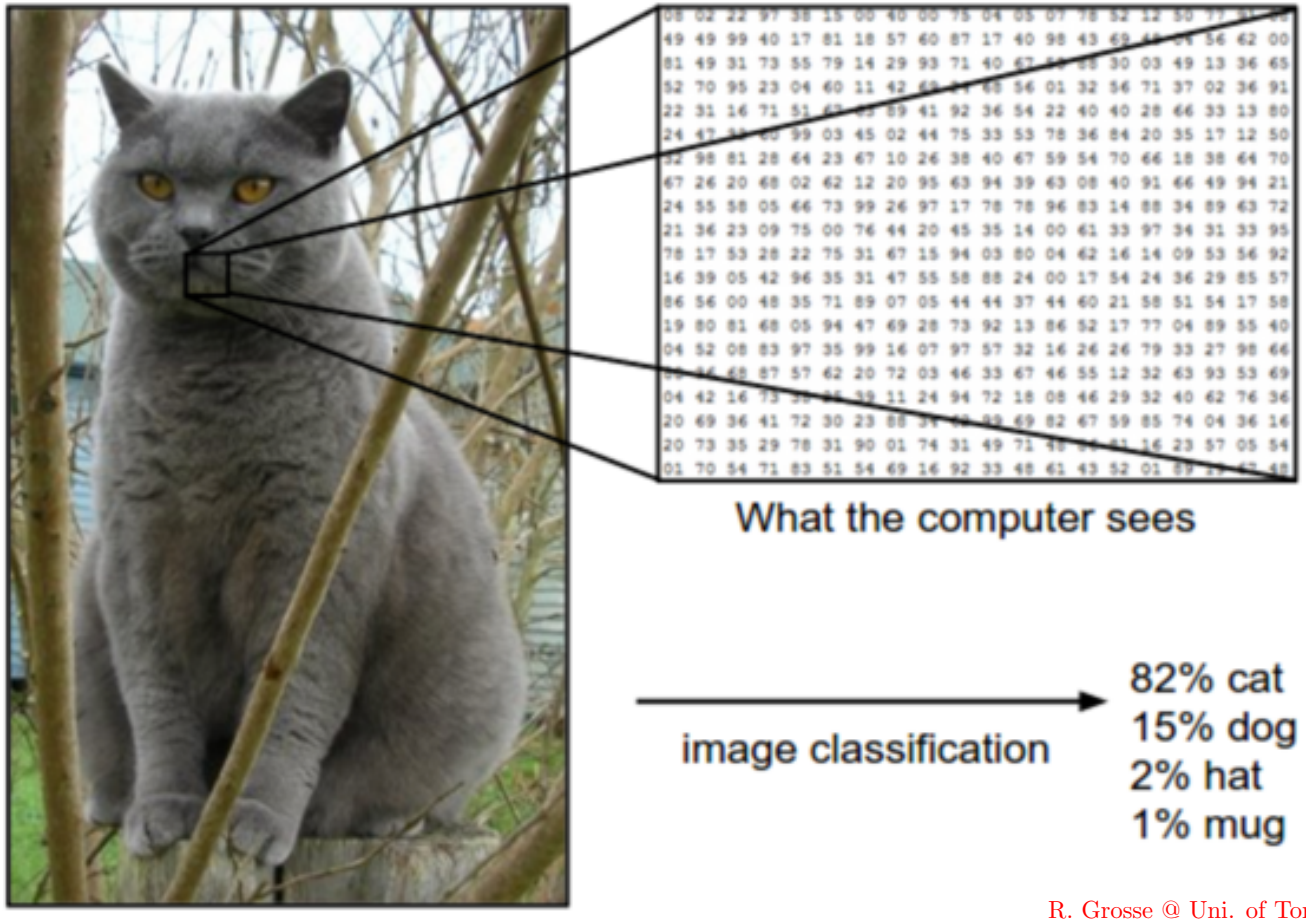
- Utility matrix for recommenders (Amazon product catalogue)
- The netflix prize training set: $\sim 1M$ ratings of the form $\langle$user, movie, date of grade, grade$\rangle$
- 480,189 users, 17,770 movies

	p_1	p_2	p_3						p_j									p_m
u_1	1		2	1		4			2		3	2		5				2
u_2		1				2		1			2			1				3
u_3		1	1	2				1							1		2	
				3		2			5			2			3	4		
		1			2									5				
u_i			3	2	1		4	5	?	1		3	1		2			1
			4													4		
			5			1										5		
		1		4							1	3		5		1	2	
u_n			3		1	1		2	1					4				5

High Dimensional Data

High Dimensional Data is common in many applications

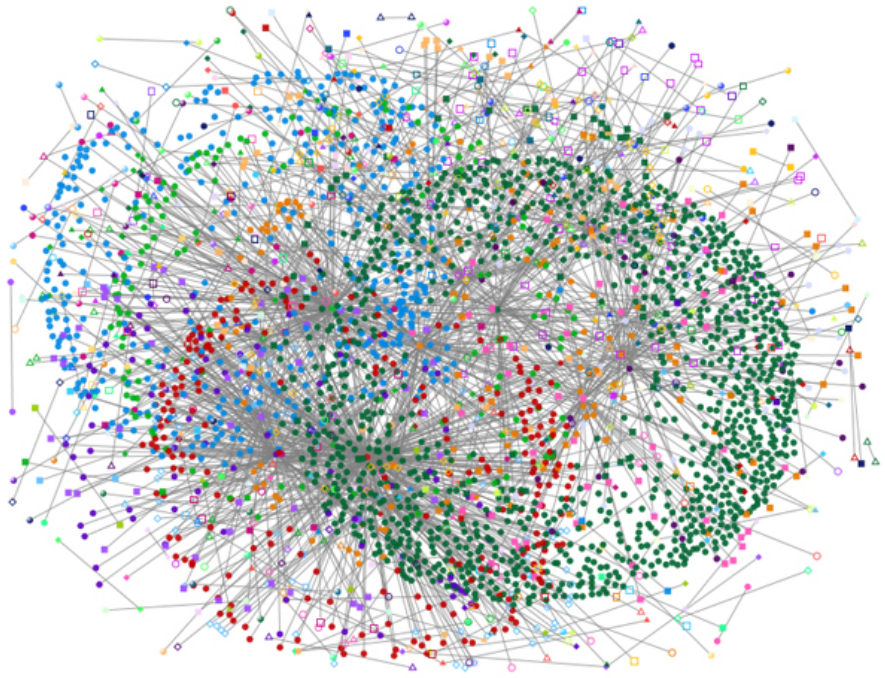
- Images and videos from multi-mega pixels digital cameras



High Dimensional Data

High Dimensional Data is common in many applications

- Social networks as adjacency matrix
- A row of facebook graph's adjacency matrix contains more than a billion dimensions



Curse of dimensionality

- In general as features increase redundancy also increases
  - more noise added to data than signal
- High dimensional data is hard to visualize and interpret
- Computationally challenging
  - Processing time
  - Storage capacity
  - Communication bandwidth

Dimensionality Reduction

- We discussed many issues with large dimensions
- We focus on computational aspect of the curse
  - Processing time
  - Storage capacity
  - Communication bandwidth
- Our goal is to reduce dimensionality of the dataset, while preserving pairwise distance

Data Compression

Data Compression deals with large volumes of data

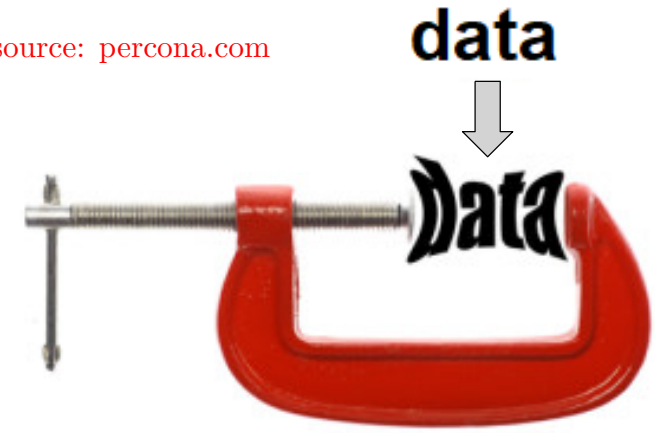
- Given a point set $X = \{x_1, x_2, x_n\}$. Find
  - a compression scheme $f : X \mapsto X'$ ▷ encoder
  - a decompressor $g : X' \mapsto X$ ▷ decoder
  - objective is to minimize

$$\sum_{i=1}^{n} \|x_i - g(f(x_i))\|^p$$

- called ℓ_p reconstruction error
- g is not necessarily f^{-1}
- If $g = f^{-1}$, compression is called Lossless otherwise it is Lossy

Data Compression

source: percona.com



Lossless Compression

Huffman
Shannon Fano

= original data

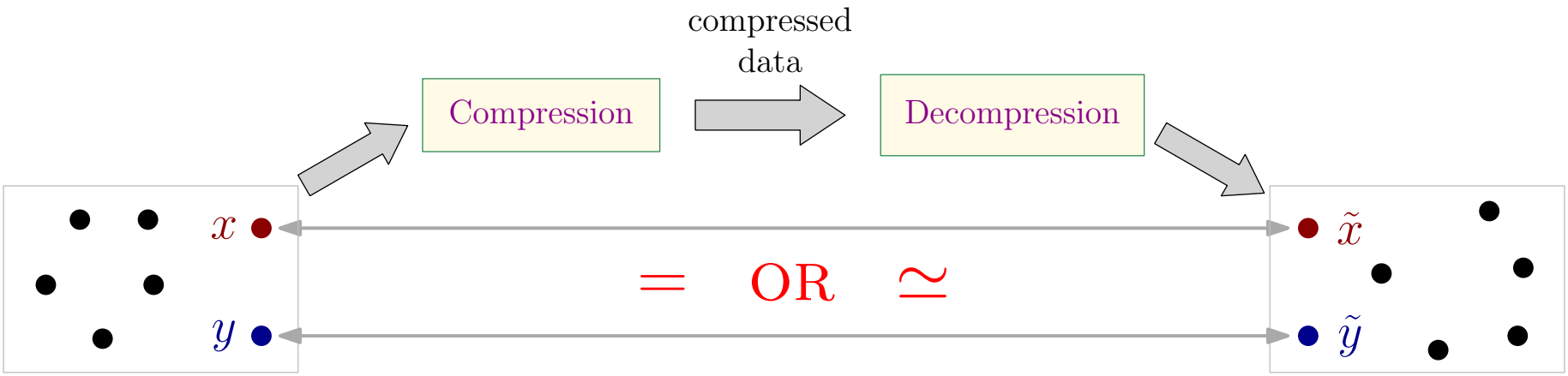
Lossy Compression

JPEG
MPEG

≃ original data

Data Compression

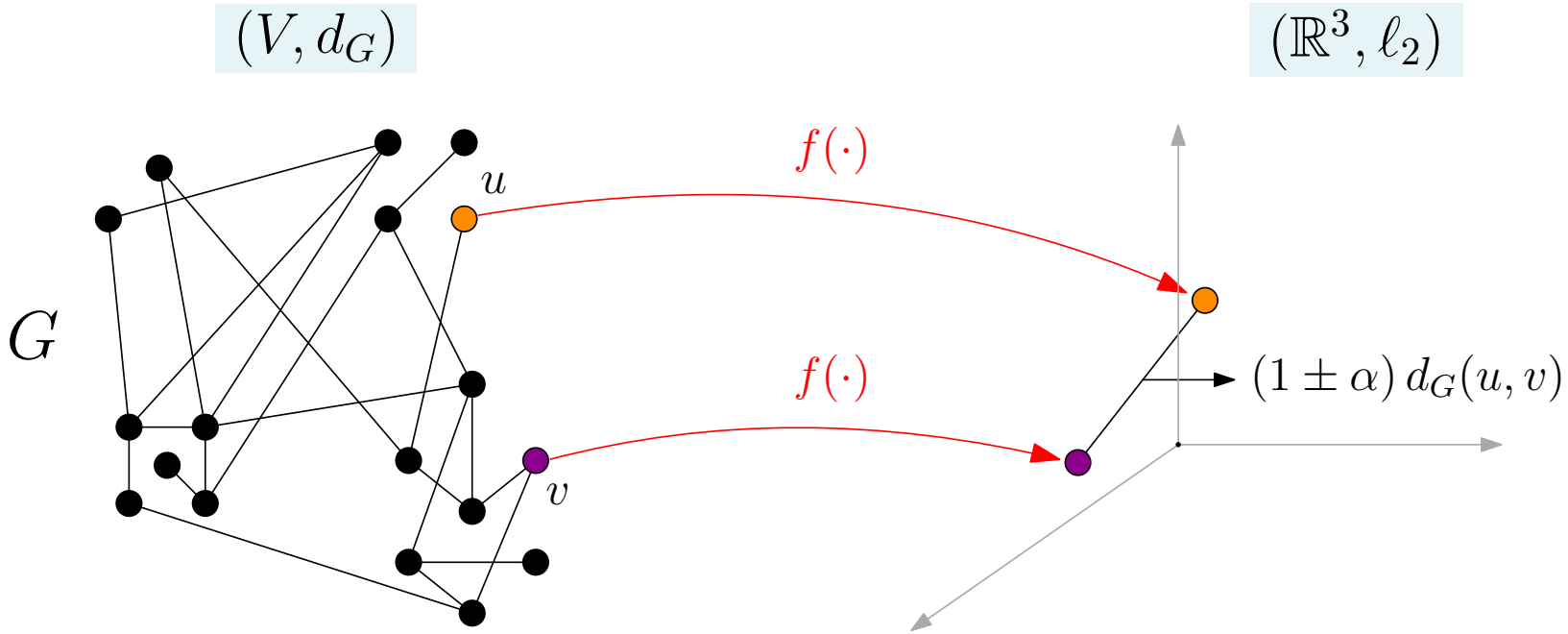
Data Compression deals with large volumes of data



Low Distortion Embedding

- Given two metric spaces (X, d) and (Y, d') and a real $\alpha > 0$, Find
- an embedding function $f : X \mapsto Y$ such that

$$\forall\, x_i, x_j \in X \quad \frac{1}{\alpha} d(x_i, x_j) \;\leq\; d'(f(x_i), f(x_j)) \;\leq\; d(x_i, x_j)$$



Low Distortion Embedding

- Given two metric spaces (X, d) and (Y, d') and a real $\alpha > 0$, Find
- an embedding function $f : X \mapsto Y$ such that

$$\forall\, x_i, x_j \in X \quad \frac{1}{\alpha} d(x_i, x_j) \;\leq\; d'(f(x_i), f(x_j)) \;\leq\; d(x_i, x_j)$$

- Points in X embedded into Y almost preserving pairwise distances
- The space Y may be easy to work with
- The distance metric d' may be computationally nicer
- Graph vertices with shortest paths distances embedded to $(\mathbb{R}^k, \ell_2)$
- Sequences with edit distance embedded into Euclidean space

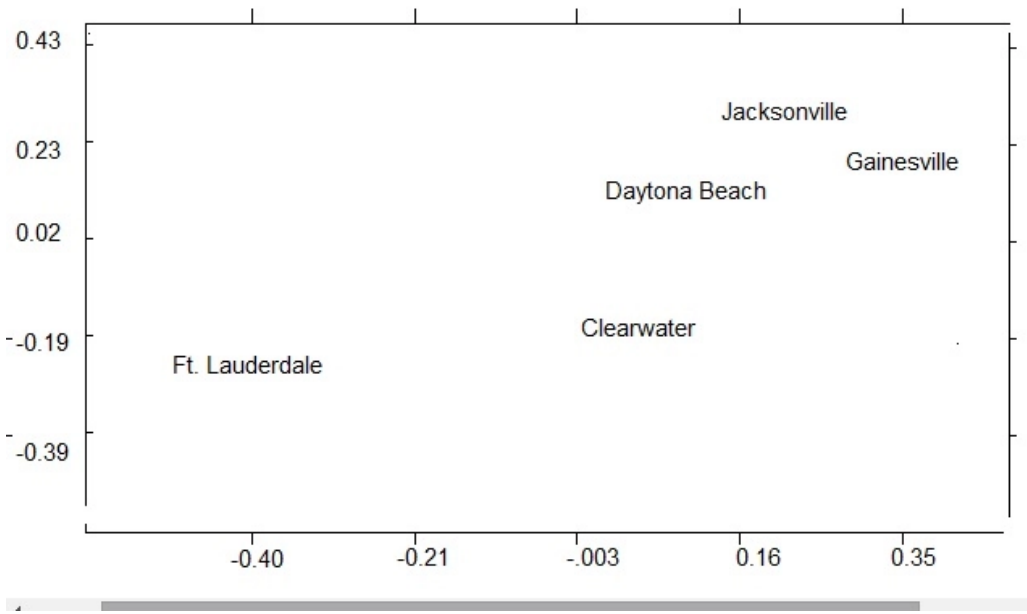
Multi-Dimensional Scaling

- Given $X = \{x_1, \ldots, x_n\}$ and pairwise distance matrix $D = \{d_{ij}\}$, Find
- A k-dimensional representation $\{x'_1, x'_2, \ldots, x'_n\}$ for points in X

$$\forall\, x_i, x_j \in X \quad d(x'_i, x'_j) \sim D(i, j)$$

source: statisticshowto.com

CITY	Clearwater	Daytona Beach	Ft. Lauderdale	Gainesville	Jacksonville
Clearwater	0	159	247	131	197
Daytona Beach	159	0	230	97	89
Ft. Lauderdale	247	230	0	309	317
Gainesville	131	97	309	0	68
Jacksonville	197	89	317	68	0

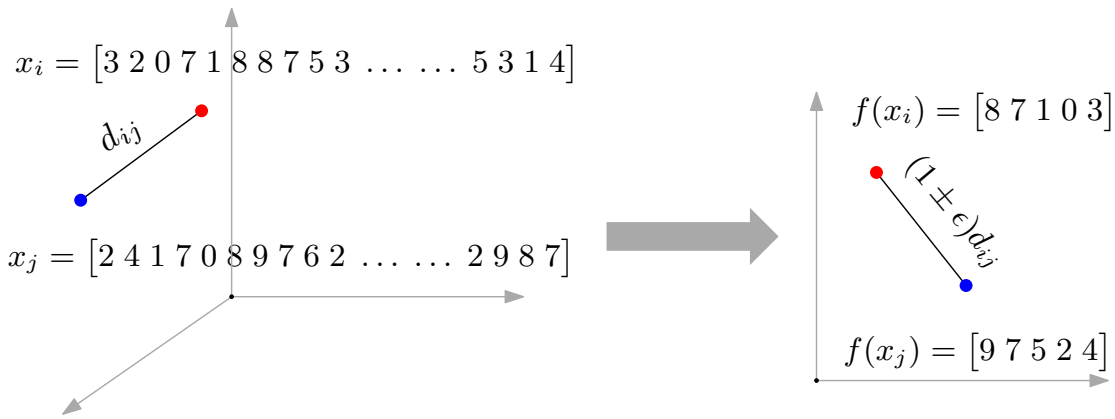


- Many methods depending on whether or not the given and required distance measure is metric or Euclidean

Dimensionality Reduction

- Given a point set $\mathcal{X} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$, Find
- a dimensionality reduction function $f : \mathbb{R}^m \mapsto \mathbb{R}^k$, $k \ll m$ such that

$$\forall\, x_i, x_j \in \mathcal{X} \quad (1-\epsilon)d(x_i, x_j) \;\leq\; d(f(x_i), f(x_j)) \;\leq\; (1+\epsilon)d(x, y)$$

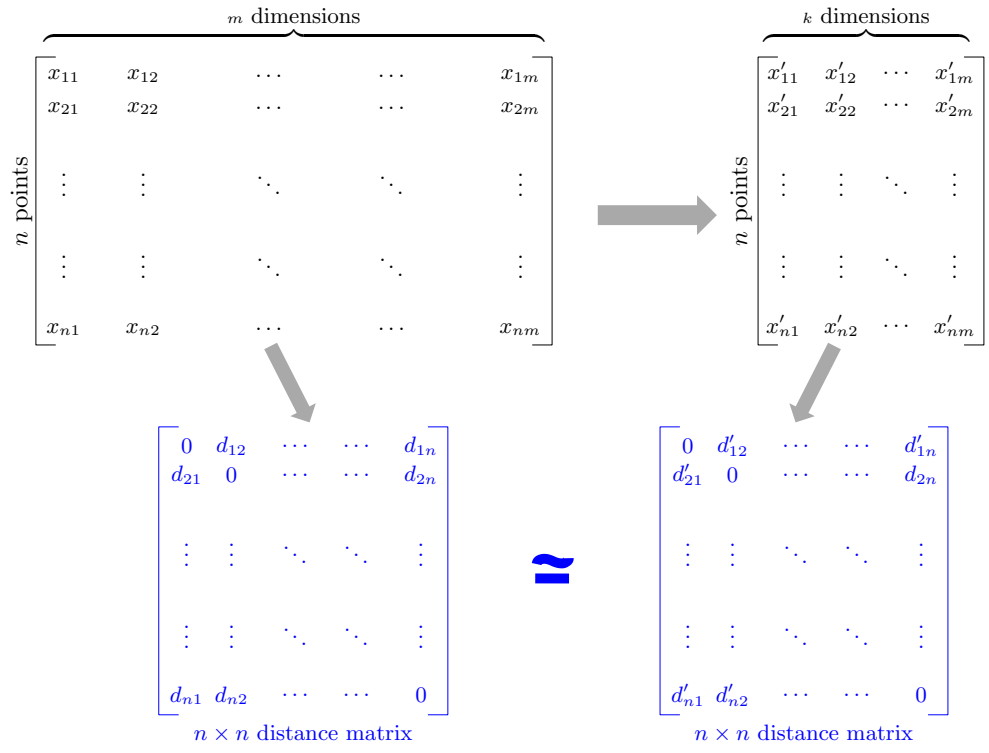


- A special case of low distortion embedding
  - distance measure d is the same in both domain and co-domain
- Different than data compression
  - do not require $x \simeq f(x)$, but only $d\big(f(x_i), f(x_j)\big) \simeq d(x_i, x_j)$

Dimensionality Reduction

- Given a point set $\mathcal{X} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$
- a dimensionality reduction function $f : \mathbb{R}^m \mapsto \mathbb{R}^k$, $k \ll m$ such that

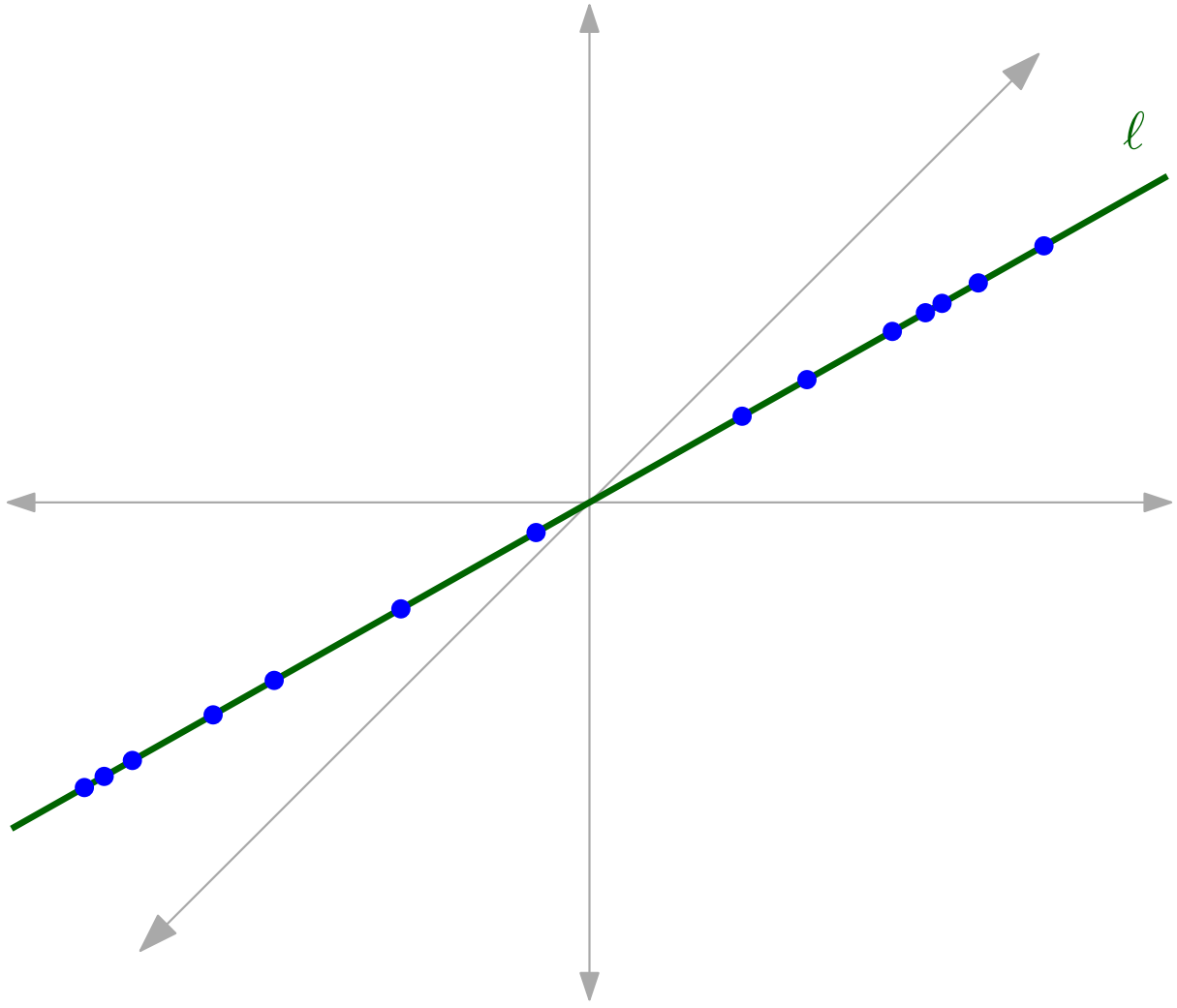
$$\forall\, x_i, x_j \in \mathcal{X} \quad (1-\epsilon)d(x_i, x_j) \;\leq\; d(f(x_i), f(x_j)) \;\leq\; (1+\epsilon)d(x, y)$$



Dimensionality Reduction can be Data Dependent and Data Oblivious

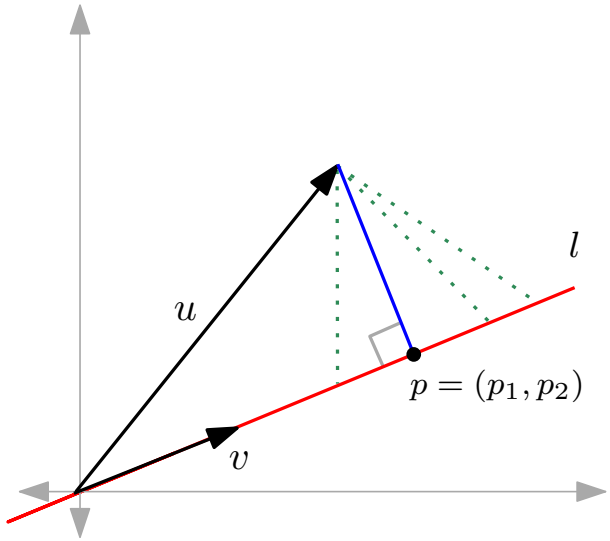
Dimensionality Reduction:

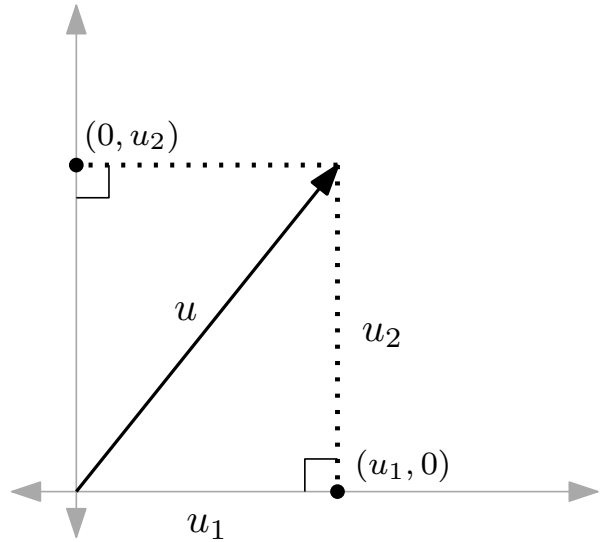
As a warm-up exercise, suppose the m-d data lies on a line



Projection

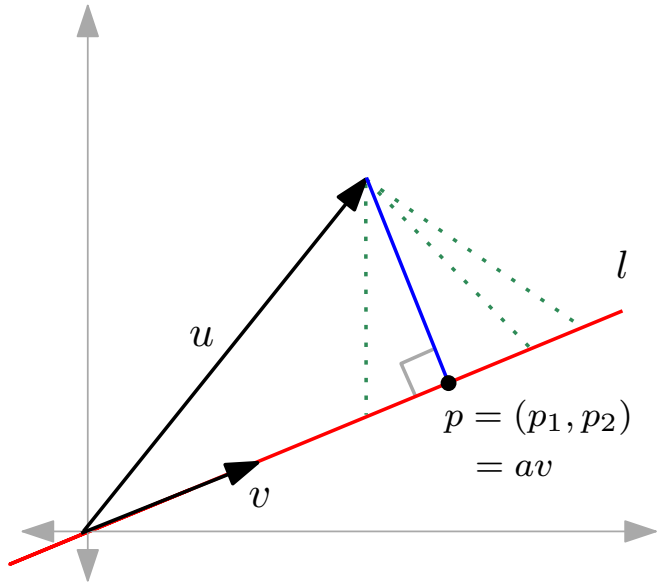
- Let **v** be a unit vector, let ℓ be a line in the direction of **v**
- Find the point p on ℓ that is closest to a vector (point) **u**
- The line connecting **u** to p is perpendicular to **v**
- Otherwise p will not be the closest point ▷ Pythagoras theorem
- The point (vector) p is called the projection of **u** on **v**
- Finding projection of **u** on the standard basis vectors is easy
- For general vectors we derive it from dot product





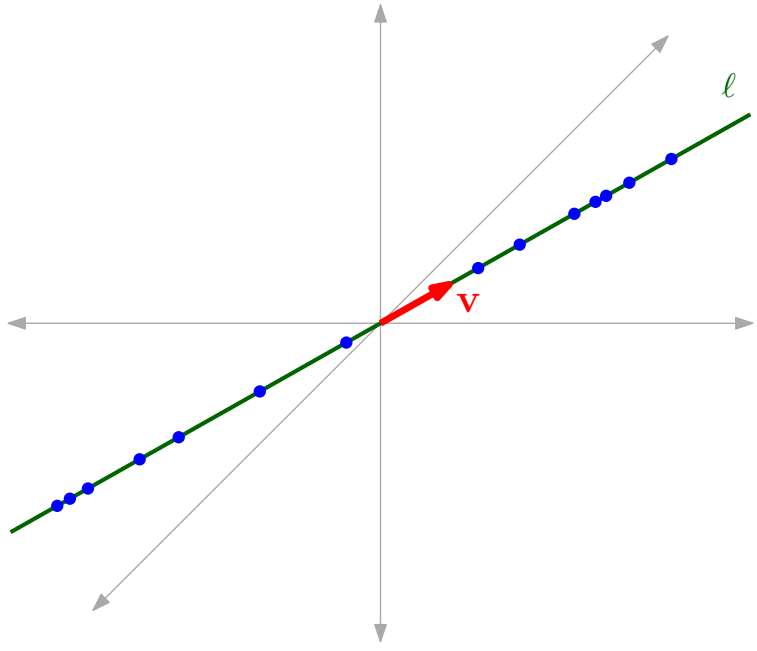
Dot product, Cosines and Projection

- p is just scaled vector $\mathbf{v}$, $p = a\,\mathbf{v}$, find that scalar a
- $\mathbf{u} - p = \mathbf{u} - a\mathbf{v}$ is perpendicular on $\mathbf{v}$, or $\mathbf{v} \cdot \mathbf{v} - a\,\mathbf{v} = 0$
- Hence $\mathbf{v} \cdot \mathbf{u} - \mathbf{v} \cdot a\,\mathbf{v} = \mathbf{v} \cdot \mathbf{u} - a\,\mathbf{v} \cdot \mathbf{v} = 0$
- Which means $a\,\mathbf{v} \cdot \mathbf{v} = \mathbf{v} \cdot \mathbf{u}$
- $a = \dfrac{\mathbf{v} \cdot \mathbf{u}}{\mathbf{v} \cdot \mathbf{v}} = \dfrac{\mathbf{v} \cdot \mathbf{u}}{\|\mathbf{v}\|}$



Dimensionality Reduction:

As a warm-up exercise, suppose the m-d data lies on a line

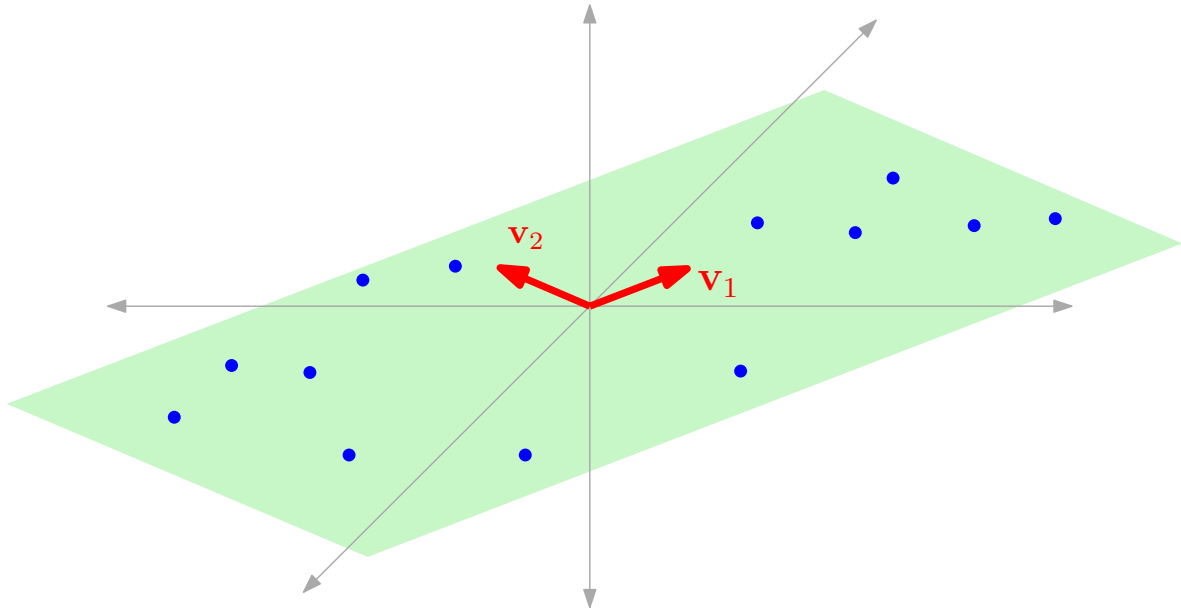


- Let $\mathbf{v}$ be the unit vector in direction of ℓ
- For $\mathbf{x_i} \in X$, let $f(\mathbf{x_i}) := \mathbf{v} \cdot \mathbf{x}$
- In this case, since $\mathbf{v} \cdot \mathbf{x}_i = \mathbf{x}_i$ (as $\mathbf{x}_i$ lies on ℓ), we get

$$\forall i, j \quad \|f(x_i) - f(x_j)\| = \|\mathbf{v} \cdot \mathbf{x}_i - \mathbf{v} \cdot \mathbf{x_j}\| = \|\mathbf{x}_i - \mathbf{x}_j\|$$

Dimensionality Reduction:

If the m-d data lies on a plane with orthonormal basis $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$



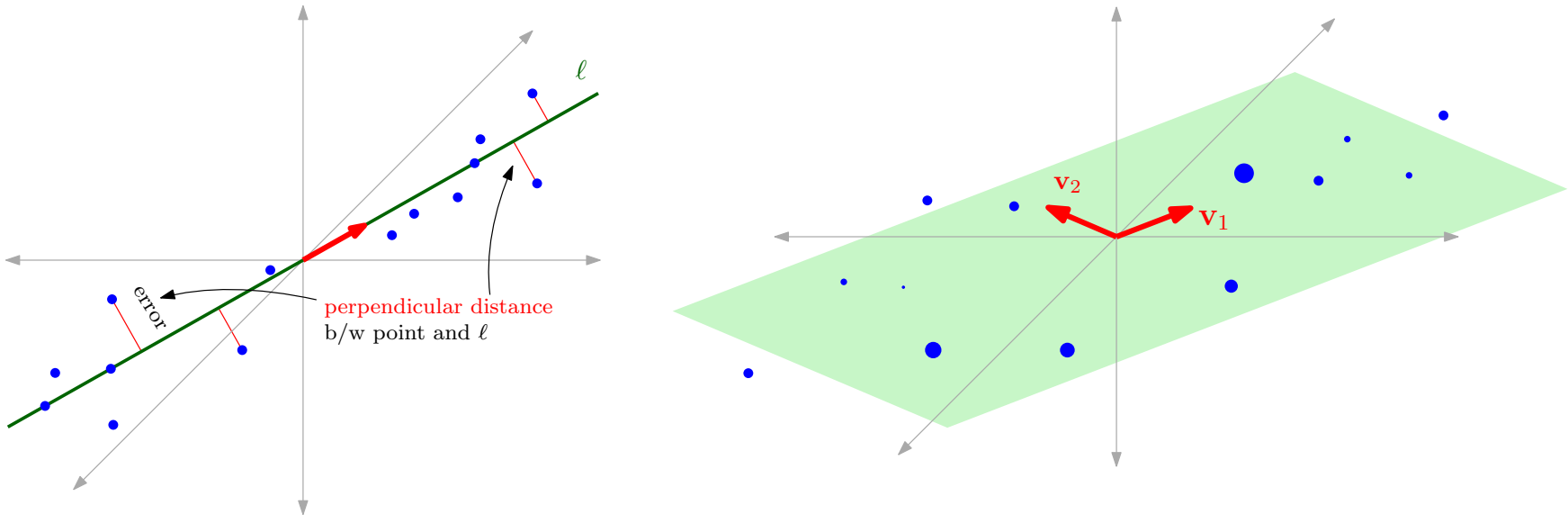
- Let $\mathbf{V}$ be the matrix with $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ as columns
- For $\mathbf{x_i} \in X$, let $f(\mathbf{x_i}) := \mathbf{xV}$, we get

$$\forall i, j \quad \|f(x_i) - f(x_j)\| = \|\mathbf{x}_i\mathbf{V} - \mathbf{x_j}\mathbf{V}\| = \|\mathbf{x}_i - \mathbf{x}_j\|$$

In above cases, we get 0 error (no-distortion) dimensionality reduction

Do not know $\mathbf{V}$

Dimensionality Reduction: Sidenote



We can find the low dimensional space to which the data is close by

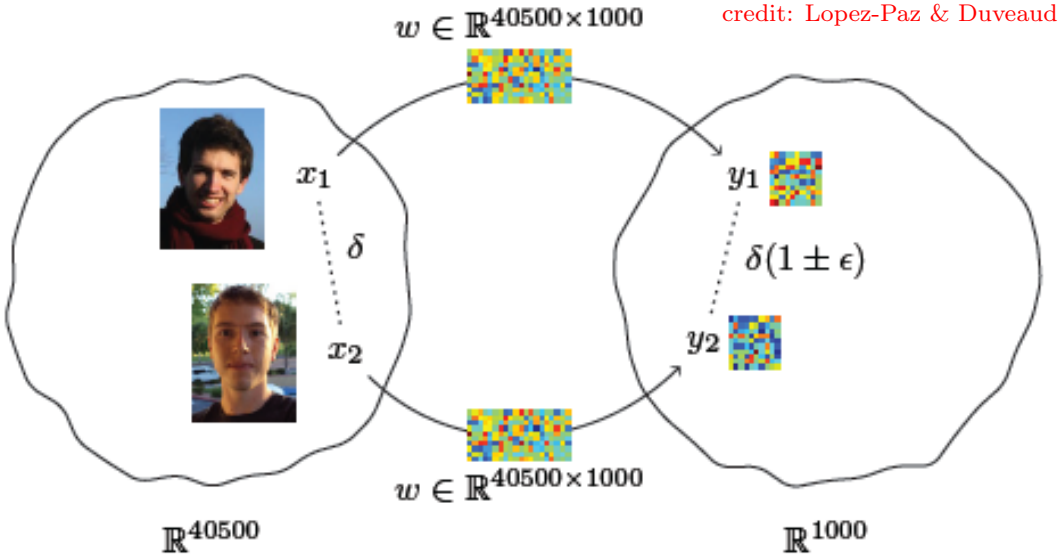
- Similar to (multiple) linear regression, but
  1. Error here is perpendicular distance not vertical distance
  2. Goal there is to minimize SSE, here it is to minimize pairwise distances
- With modified goals can take this approach but it is data dependent dimensionality reduction (principal component analysis PCA)

Linear Dimensionality Reduction

- Given a point set $\mathcal{X} \subset \mathbb{R}^m$, $\{x_1, \ldots, x_n\}$
- a linear function $f : \mathbb{R}^m \mapsto \mathbb{R}^k$, $k \ll n$ such that

$$\forall\, x_i, x_j \in \mathcal{X} \quad (1-\epsilon)d(x_i, x_j) \;\leq\; d(f(x_i), f(x_j)) \;\leq\; (1+\epsilon)d(x, y)$$

- f can be represented by a linear transformation A, i.e. $f(\mathcal{X}) = A\mathcal{X}$, $\mathcal{X}$: the $n \times m$ data matrix with each $x_i \in \mathcal{X}$ as a row
- Feature selection/extraction are also linear dimensionality reduction



$$(1-\epsilon)\|x_1 - x_2\|^2 \leq \|y_1 - y_2\|^2 \leq (1+\epsilon)\|x_1 - x_2\|^2$$

Johnson-Lindenstrauss Lemma

Theorem

Given $\mathcal{X} = \{x_1, \ldots, x_n\} \subset \mathbb{R}^m$. *For* $\epsilon \in (0, 1/2)$, *there exists a linear map* $f : \mathbb{R}^m \to \mathbb{R}^k$, $k = c \log n/\epsilon^2$ *such that for any* $x_i, x_j \in \mathcal{X}$

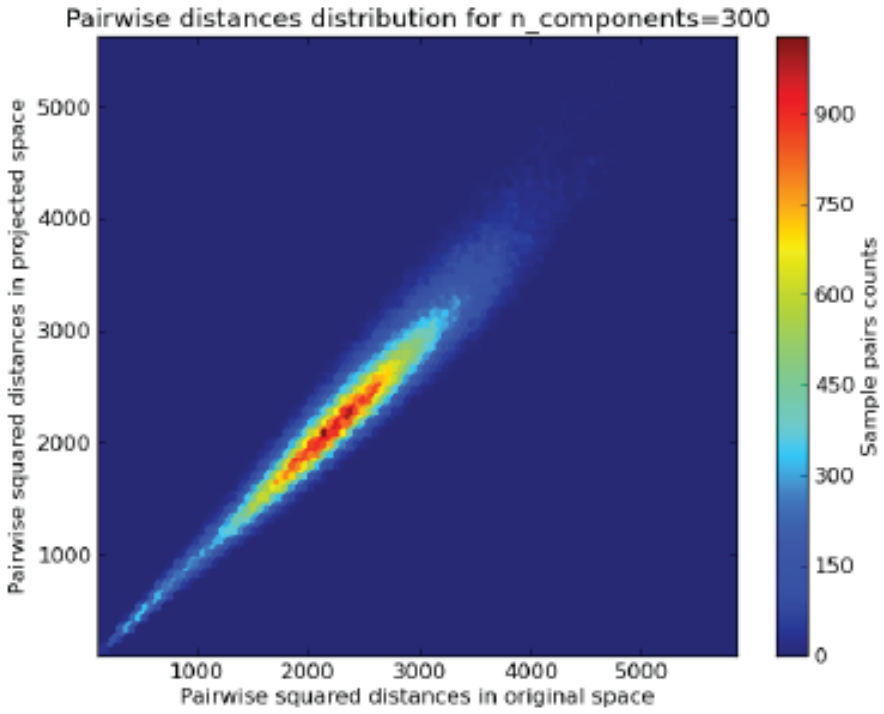
$$(1 - \epsilon)\|x_i - x_j\|_2 \leq \|f(x_i) - f(x_j)\|_2 \leq (1 + \epsilon)\|x_i - x_j\|_2$$

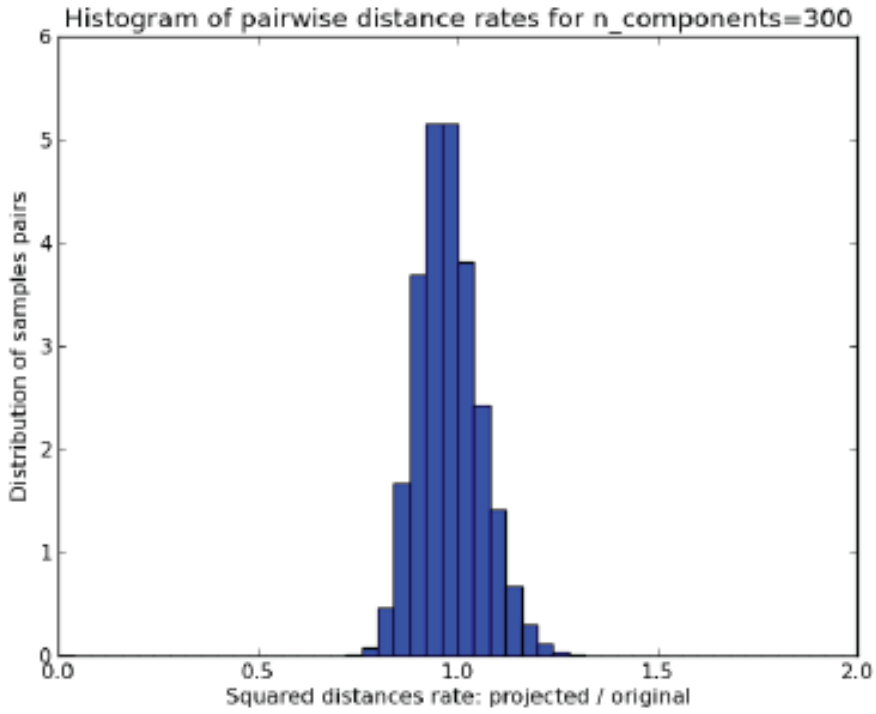
- Distance matrix computation now takes $O(n^2 \frac{\log n}{\epsilon^2})$ instead of $O(n^2 m)$
- Nearest neighbor computation now takes $O(n \frac{\log n}{\epsilon^2})$ instead of $O(nm)$

Note the lemma works only for ℓ_2 distance

Johnson-Lindenstrauss Lemma

Data: 20-newsgroups, from 100.000 features to 300 (0.3%)

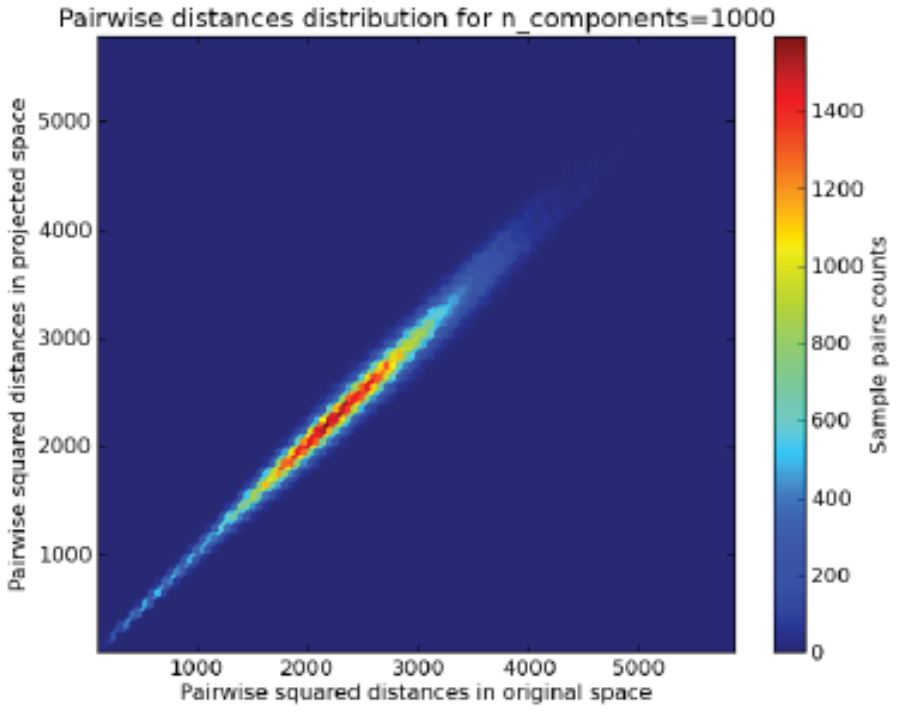


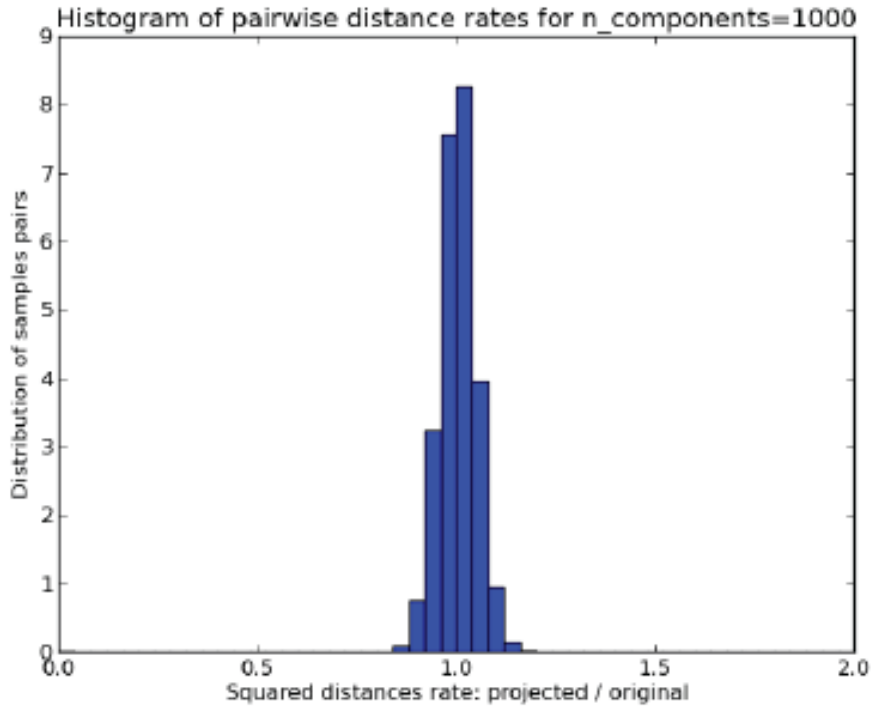


source: van de Meent @ Northeastern Uni.

Johnson-Lindenstrauss Lemma

Data: 20-newsgroups, from 100.000 features to 1.000 (1%)

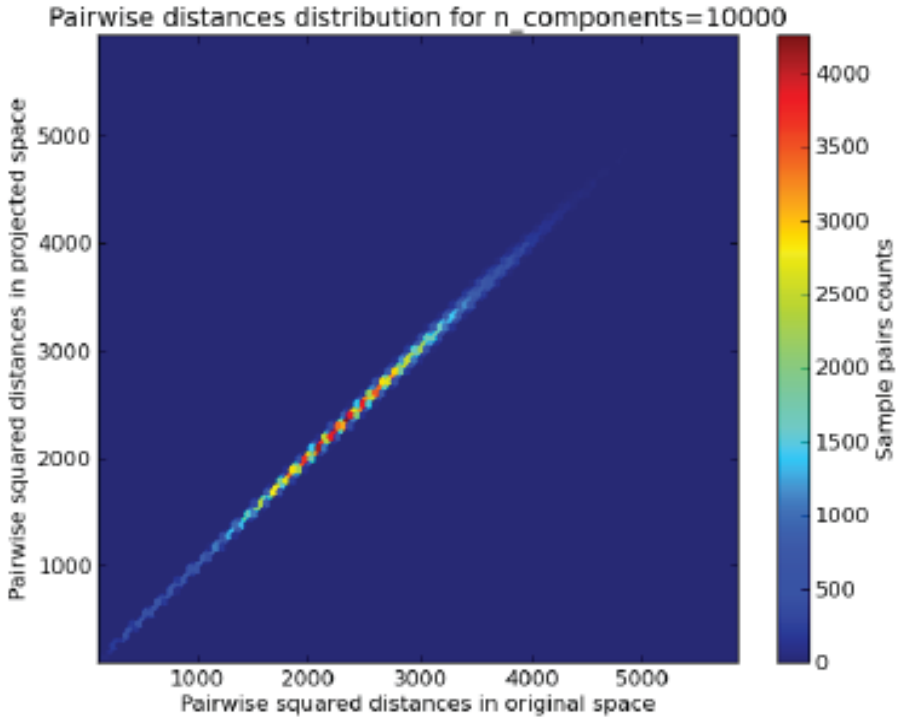


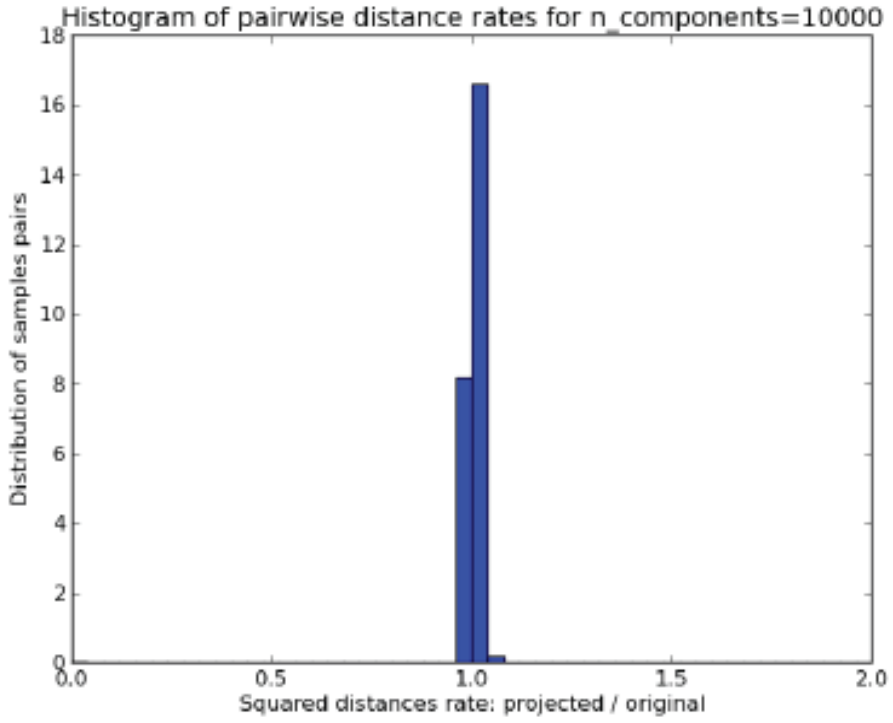


source: van de Meent @ Northeastern Uni.

Johnson-Lindenstrauss Lemma

Data: 20-newsgroups, from 100.000 features to 10.000 (10%)





source: van de Meent @ Northeastern Uni.

Johnson-Lindenstrauss Lemma: Proof

- A constructive proof of JL lemma:
  project $\mathcal{X}$ onto k random directions
- Choose k random unit vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k \in \mathbb{R}^m$
- Let $\mathcal{V}$ be the $m \times k$ matrix with $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ as columns
- Each row of $\mathcal{X}\mathcal{V}$ is the reduced dimensional version of x_i

$$\underset{n \times m}{\mathcal{X}} \quad \underset{m \times k}{\mathcal{V}} \quad \underset{n \times k}{\mathcal{Y}}$$

$$\begin{bmatrix} x_{11} & x_{12} & \cdots & \cdots & x_{1m} \\ x_{21} & x_{22} & \cdots & \cdots & x_{2m} \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & \cdots & x_{nm} \end{bmatrix} \begin{bmatrix} & & \\ & & \\ & & \\ & & \\ & & \\ & & \end{bmatrix} = \begin{bmatrix} & \\ & \\ & \\ & \end{bmatrix}$$

Johnson-Lindenstrauss Lemma: Proof

Recall our discussion on generating random unit vectors

$\mathbf{v} = (\underbrace{\mathcal{N}(0,1), \mathcal{N}(0,1), \ldots, \mathcal{N}(0,1)}_{m\text{-coordinates}})$, normalized by $\|v\|$ is a provably random unit vector (a point on the surface of the unit m-ball)

We also discussed that the more discrete version $\mathbf{v} \in [-1,1]^m$ is a good enough approximation of a random unit vector

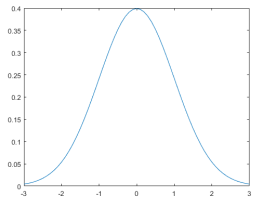
We give the sketch of the constructive proof of JL-Lemma by projecting on such random unit vectors

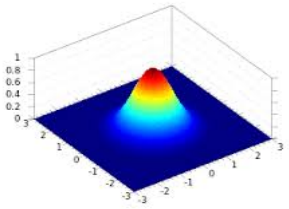
Approximate Random Direction

Generating a random direction in $\mathbb{R}^m$

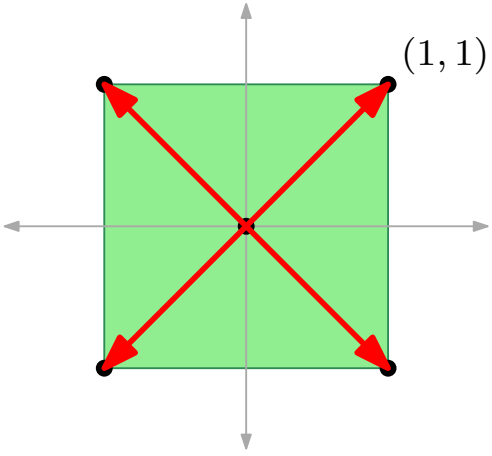
$$\mathbf{v} = \underbrace{(\mathcal{N}(0,1), \mathcal{N}(0,1), \ldots, \mathcal{N}(0,1))}_{m\text{-coordinates}}$$

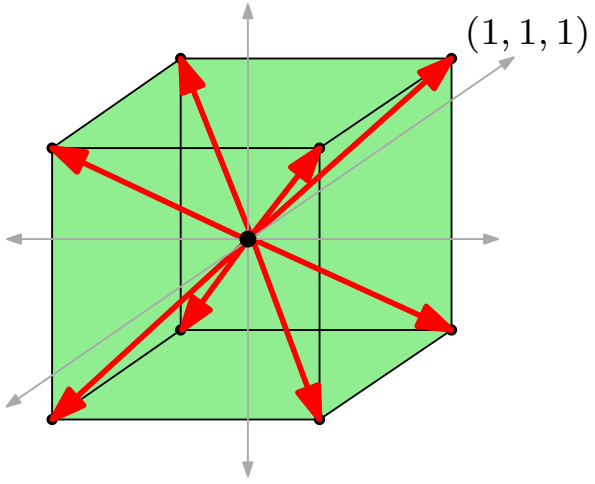
normalized by $\|\mathbf{v}\|$





- Approximately generate unit directions
  - generate directions towards corners of the m-cubes $[-1,1]^m$
- For $m \gg 1$, these 2^m directions approximately cover surface of m-ball
- Achlioptas (2003), Database-friendly random projections: ...





Johnson-Lindenstrauss Lemma: Proof

Generate a random direction $\mathbf{v} \in \{-1, 1\}^m$

For $\mathbf{x} \in \mathcal{X}$ let $f_v(\mathbf{x}) = \langle \mathbf{x}, \mathbf{v} \rangle = \mathbf{x} \cdot \mathbf{v} = \sum_{i=1}^{m} \mathbf{x}_i \mathbf{v}_i$

$$E\left[(f_v(\mathbf{x}) - f_v(\mathbf{y}))^2\right] = E\left[\sum_{i=1}^{m} \mathbf{v}_i^2 (\mathbf{x}_i - \mathbf{y}_i)^2\right] = \sum_{i=1}^{m} (\mathbf{x}_i - \mathbf{y}_i)^2 E[\mathbf{v}_i^2]$$

- Note that dimensionality of $\mathbf{x}$ and $\mathbf{y}$ is reduced to only 1
- $E[\mathbf{v}_i^2] = 1 \implies E\left[\|f_v(\mathbf{x}) - f_v(\mathbf{y})\|^2\right] = \|\mathbf{x} - \mathbf{y}\|^2$

Two Issues with this result

1. We want to preserve distances almost surely, not in expectation only
2. We want guarantee on distances not squared distances
   - $E[X^2] = \mu^2 \not\Longrightarrow E[X] = \mu$
   - $X = \begin{cases} 0 & \text{w. prob. } 1/2 \\ 1 & \text{else} \end{cases}$ $E[X] = 1$, while $E[X^2] = 2$, and $\sqrt{2} \neq 1$

Johnson-Lindenstrauss Lemma: Proof

Resolve issues with probability amplification - repeated independent trials

Generate k random directions $\mathbf{v}^1, \mathbf{v}^2, \ldots, \mathbf{v}^k \in \{-1, 1\}^m$, scale by $1/\sqrt{k}$

For $\mathbf{x} \in \mathcal{X}$, let $f(\mathbf{x}) = (f_{v^1}(x), f_{v^2}(x), \ldots, f_{v^k}(x))$ i.e. $f(\mathbf{x})[i] = \mathbf{x} \cdot \mathbf{v}^i$

$$E(\|f(\mathbf{x}) - f(\mathbf{y})\|^2) = E\left(\sum_{j=1}^{k} \left(f_{v^j}(\mathbf{x}) - f_{v^j}(\mathbf{y})\right)^2\right) = \sum_{j=1}^{k} E\left(\sum_{i=1}^{n} (\mathbf{v}_i^j)^2 (\mathbf{x}_i - \mathbf{y}_i)^2\right)$$

$$E\left(\sum_{i=1}^{n} (\mathbf{v}_i^j)^2 (\mathbf{x}_i - \mathbf{y}_i)^2\right) = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{y}_i)^2 E((\mathbf{v}_i^j)^2) = \frac{\|x - y\|^2}{k}$$

Thus $$E[\|f(x) - f(y)\|^2] = \|x - y\|^2$$

In expectation, mapping f preserves the squared ℓ_2 distance between a pair

Johnson-Lindenstrauss Lemma: Proof

The ℓ_2^2-distance in reduced dimensions is concentrated around its mean

Using Hoeffding's inequality (intervals for X_j's hidden in constants), we get

There exists constants c_1 and c_2, such that

- $Pr\left(\|f(x) - f(y)\|^2 \geq (1+\epsilon)\|x-y\|^2\right) \leq e^{-c_1\epsilon^2 k}$
- $Pr\left(\|f(x) - f(y)\|^2 \leq (1-\epsilon)\|x-y\|^2\right) \leq e^{-c_2\epsilon^2 k}$

Thus, there is some constant c, such that

$$Pr\left((1-\epsilon)\|x-y\|^2 \leq \|f(x) - f(y)\|^2 \leq (1+\epsilon)\|x-y\|^2\right) \geq 1 - e^{-c\epsilon^2 k}$$

- Choose k so $e^{-c\epsilon^2 k} < 1/n^3 \quad \Longrightarrow \quad k \geq 1/\epsilon^2(\log(n) + \log(1/c))$
- By union bound probability that some pair is 'bad' is at most $1/n$
- With prob. $\geq 1 - 1/n$ squared ℓ_2-distance is preserved for all pairs

Johnson-Lindenstrauss Lemma: Remarks

- Exact proof of JL-lemma uses vectors $\mathbf{v}^j$'s from $\mathcal{N}(0,1)^m$
  - Original proof was actually different, required $\mathbf{v}^j$'s to be orthonormal
- Dimensionality of resulting space, k is $O(1/\epsilon^2(\log(n) + \log(1/c)))$
- k is independent of m (original dimensions) and depends on n only
- $k \propto \epsilon$ (the error margin), require less error, k naturally would grow
- This is essentially the best for linear maps ▷ Larsen & Nelson (2016), *The Johnson Lindenstrauss lemma is optimal for linear dimenisonality reduction*
- Even other maps can't do much better ▷ Larsen & Nelson (2017), *Optimality of the Johnson-Lindenstrauss lemma*
- Can precompute the matrix $\mathcal{V}$ ▷ Data Oblivious
- No need to store this matrix - can generate it using a random number generator with fixed seeds or hash functions ▷ streaming algorithms

Johnson-Lindenstrauss Lemma: Remarks

- JL lemma works only for the ℓ_2 distance
- Meaning random projection may not work for other distance measures
- To preserve ℓ_1-distance within $(1 \pm \epsilon)$, the number of dimensions required k is $\geq n^{1/2 - O(\epsilon \log(1/\epsilon))}$
  - ▷ Brinkman & Charikar (2003), *On the impossibility of dimension reduction in* ℓ_1