Algorithms at Scale
(Week 13)

Dimensionality Reduction

Giovanni Pagliarini

Corentin Dumery

Summary

- Motivation
- Approaches
- Johnson-Lindenstrauss Transforms
- Fast JL Transform (+ Proof)
- Experiments' Design
- Results

Dimensionality Reduction - Why?

d

n

0.3	0.9	...	0.343
0.9	0.2	...	0.43
...	...	...	...
0.6	0.0	...	1.0

- Reduces cpu workload and storage needed;
- Alleviates curse of dimensionality;
- Allows noise/redundancies removal.

potential performance improvement

Source: https://blog.knoldus.com/machinex-when-data-is-a-curse-to-learning/amp/

Common approaches

- Drop useless variables
  - Variables with **low variance** may not tell anything useful;
  - When variables are **highly correlated**, one of them is unnecessary;

- Derive a new set of variables
  - Factor Analysis;
  - Principal Component Analysis;
  - Linear Discriminant Analysis;
  - ...
  - ***<u>Randomized Projections</u>***

$$n\left\{\overbrace{\begin{array}{|c|c|c|c|}\hline 0.3 & 0.9 & ... & 0.343 \\ \hline 0.9 & 0.2 & ... & 0.43 \\ \hline ... & ... & ... & ... \\ \hline 0.6 & 0.0 & ... & 1.0 \\ \hline \end{array}}^{d}\right.$$

What do we want?

A function that **reduces dimensionality**:

$$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$$

What do we want?

A function that **reduces dimensionality**:

$$f : \mathbb{R}^d \rightarrow \mathbb{R}^k, k \ll d$$

Without **losing information**:

$$\forall u, v \in X \subset \mathbb{R}^d, (1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

What do we want?

A function that **reduces dimensionality**:

$$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$$

Without **losing information**:

$$\forall u, v \in X \subset \mathbb{R}^d, (1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

Johnson-Lindenstrauss Lemma:

Such a function exists.

[1] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Conference in modern analysis and probability, New Haven, CI, 1982, Amer. Math. Soc., Providence, RI, 1984, pp. 189-206

Can we build a JL Transform?

- We call **Johnson-Lindenstrauss transform** any function that satisfies these properties.
- Recall that in the lemma f has to satisfy:

$$\forall u, v \in X \subset \mathbb{R}^d, (1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

Can we build a JL Transform?

- We call **Johnson-Lindenstrauss transform** any function that satisfies these properties.
- Recall that in the lemma f has to satisfy:

$$\forall u, v \in X \subset \mathbb{R}^d, (1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

f depends on X !

□ **Our data is too big, we can't go through the whole input data to define *f*...**

Let’s tweak our definition to make it easier.

What do we want?

A function that **reduces dimensionality**:

$$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$$

Without **losing information** **with probability at least ⅔**:

$$\forall u, v \in X \subset \mathbb{R}^d, (1-\epsilon)\|u-v\|^2 \leq \|f(u)-f(v)\|^2 \leq (1+\epsilon)\|u-v\|^2$$

Johnson-Lindenstrauss Lemma:

Such a function exists.

[1] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Conference in modern analysis and probability, New Haven, CI, 1982, Amer. Math. Soc., Providence, RI, 1984, pp. 189-206

What’s the simplest way of making new dimensions you can think of?

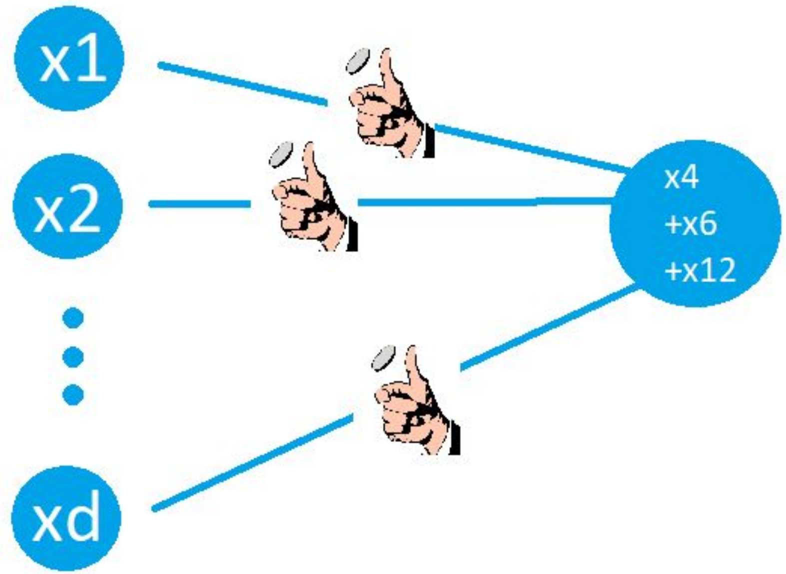
Binary coins

- Just take random linear combinations of the original dimensions.

- Very simple to implement:

```
buildTransform():
    P is a d × n matrix
    pick P[i][j] in {-1,0,1}
    return P

transform(vector):
    return P*vector
```



[2] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. J. Comput. Syst. Sci., 66(4):671-687, 2003.

Matrix multiplications can be slow.
How can we make this faster ?

The Sparsity principle

- If the matrix is sparse, the multiplication is faster to compute.

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 22 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 17 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 41 & 0 & 0 \end{bmatrix}$$

The Sparsity principle

- If the matrix is sparse, the multiplication is faster to compute.

$$
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 22 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 17 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 41 & 0 & 0
\end{bmatrix}
$$

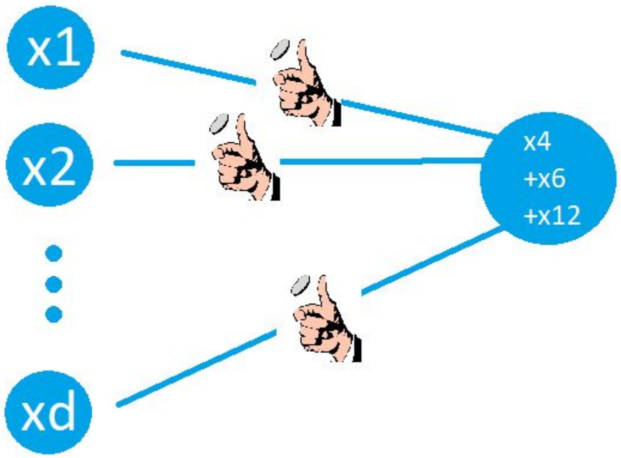
□Can simply be stored as :

- 22 in (1,1)
- 17 in (2,4)
- 41 in (4,3)

□If the fraction of non-zero elements is *1/k*, then the multiplication is (approximately) *k* times faster !

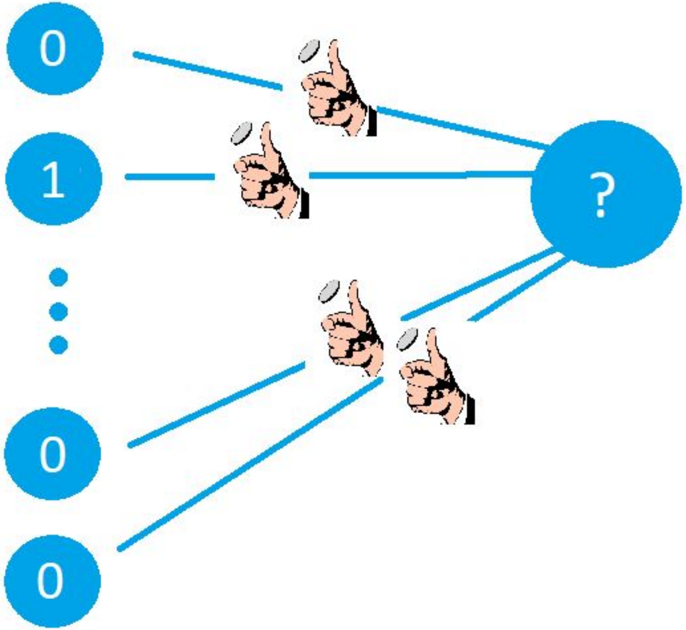
The Sparsity principle

- If the matrix is sparse, the multiplication is faster to compute.



- If the coin-flip is very likely to fail, the transform matrix will be sparse and the operation will be fast.

A problem



What if we miss the 1 in all of our new dimensions ?

Fast JL Transform (FJLT)

- We want to pre-process the data to avoid the previous problem (with high probability).

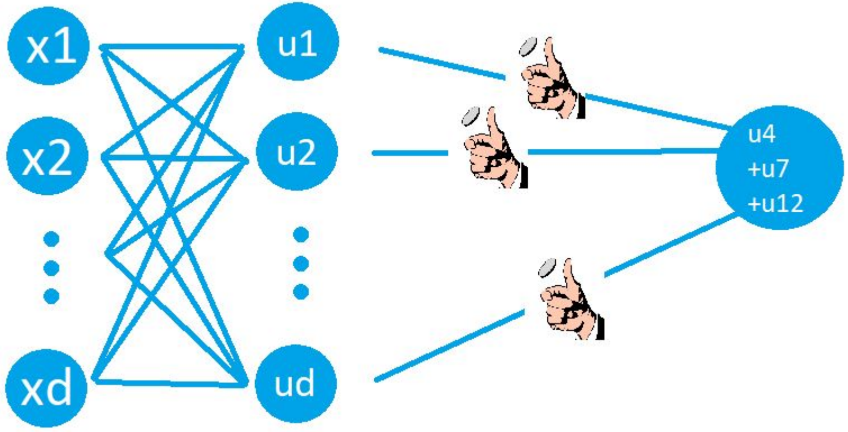
New transform:

$$f(x) = P\underbrace{HDx}$$

P is sparse

u = H D x is the pre-processed vector

How do we design H and D?



[3] Ailon, N., Chazelle, B.: The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. SIAM J. Comput. 39(1), 302-322 (2009)

FJLT: $f(x) = PHDx$

- P is the same kind of coin-flip matrix but with a very biased coin
-
-

FJLT: $f(x) = PHDx$

- P : coin-flip matrix

(H is a Walsh-Hadamard matrix. More formally, H is a square matrix of size d and: $H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$)

- $H = d^{-1/2} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$

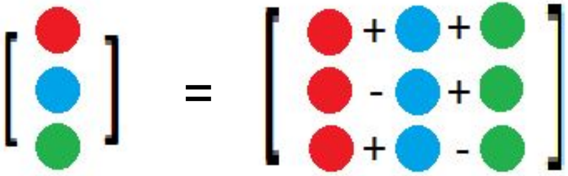
-

FJLT: $f(x) = PHDx$

(H is a Walsh-Hadamard matrix. More formally, H is a square matrix of size d and: $H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$)

- P : coin-flip matrix
- $H = d^{-1/2} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$

□H spreads out the vector:

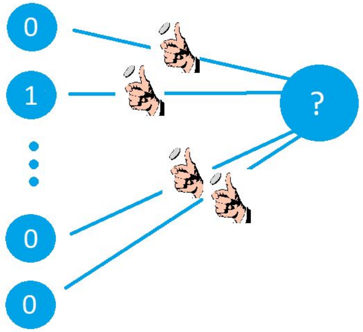
- $\begin{bmatrix} +1 & +1 & +1 \\ +1 & -1 & +1 \\ +1 & +1 & -1 \end{bmatrix}$ 

FJLT: $f(x) = PHDx$

- P : coin-flip matrix
- $H = d^{-1/2} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$
-

(H is a Walsh-Hadamard matrix. More formally, H is a square matrix of size d and: $H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$)

□This will solve our problems



FJLT: $f(x) = PHDx$

- P : coin-flip matrix
- $H = d^{-1/2} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$
- $D = \begin{bmatrix} \pm 1 & & \\ & \ddots & \\ & & \pm 1 \end{bmatrix}$

□Randomly switching signs of variables: we hope the values will cancel out in the aggregation

FJLT: $f(x) = PHDx$

- P : coin-flip matrix

$\frac{\varepsilon^{-1}\log n}{d}$ -sparse

- $H = d^{-1/2} \begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$

Walsh-Hadamard is well-known : using the FFT the multiplication takes *O(d log(d))* time

- $D = \begin{bmatrix} \pm 1 & & \\ & \ddots & \\ & & \pm 1 \end{bmatrix}$

O(d) instead of *O(d²)*

□*O(d log(d) + d* $\varepsilon^{-1}\log n$

FJLT properties and high-level proof

Unbiased property: $(1-\epsilon)\alpha\|x\|_1 \leq \mathrm{E}[\|y\|_1] \leq (1+\epsilon)\alpha\|x\|_1$

where α is a constant factor. $(\alpha = k\sqrt{2\pi^{-1}})$

FJLT properties and high-level proof

Unbiased property: $(1-\epsilon)\alpha\|x\|_1 \leq \mathrm{E}[\|y\|_1] \leq (1+\epsilon)\alpha\|x\|_1$

where α is a constant factor. $(\alpha = k\sqrt{2\pi^{-1}})$

Sharp concentration around the mean property:

$$\Pr[(1-\epsilon)\mathrm{E}[\|y\|_1] \leq \|y\|_1 \leq (1+\epsilon)\mathrm{E}[\|y\|_1]] \geq 1 - 1/20$$

FJLT properties and high-level proof

Unbiased property: $(1-\epsilon)\alpha\|x\|_1 \leq \mathrm{E}[\|y\|_1] \leq (1+\epsilon)\alpha\|x\|_1$

where α is a constant factor. $(\alpha = k\sqrt{2\pi^{-1}})$

Sharp concentration around the mean property:

$$\Pr[(1-\epsilon)\mathrm{E}[\|y\|_1] \leq \|y\|_1 \leq (1+\epsilon)\mathrm{E}[\|y\|_1]] \geq 1 - 1/20$$

Not-too-big lemma:

$$\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$$

FJLT properties and high-level proof

output

Unbiased property: $(1-\epsilon)\alpha\|x\|_1 \leq \mathrm{E}[\|y\|_1] \leq (1+\epsilon)\alpha\|x\|_1$

where α is a constant factor. $(\alpha = k\sqrt{2\pi^{-1}})$

Sharp concentration around the mean property:

output

$$\Pr[(1-\epsilon)\mathrm{E}[\|y\|_1] \leq \|y\|_1 \leq (1+\epsilon)\mathrm{E}[\|y\|_1]] \geq 1 - 1/20$$

Not-too-big lemma:

pre-processed data

$$\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$$

FJLT: $f(x) = PHDx$

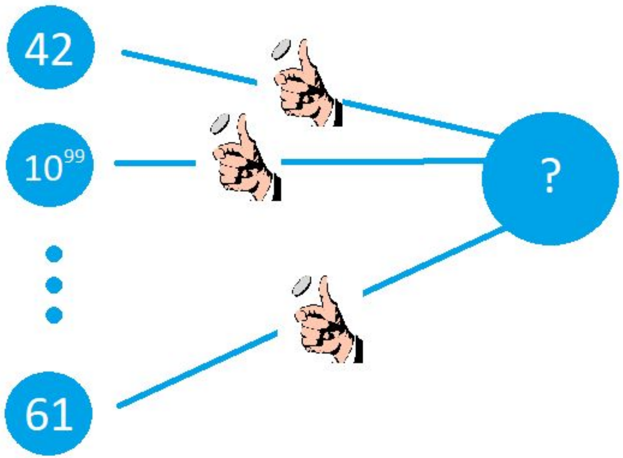
- P : coin-flip matrix

- $$H = d^{-1/2}\begin{bmatrix} +1 & +1 & +1 & +1 & +1 & +1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 \\ +1 & +1 & -1 & -1 & +1 & +1 & -1 & -1 \\ +1 & -1 & -1 & +1 & -1 & -1 & +1 & +1 \\ +1 & +1 & +1 & +1 & -1 & -1 & -1 & -1 \\ +1 & -1 & +1 & -1 & -1 & +1 & -1 & +1 \\ +1 & +1 & -1 & -1 & -1 & -1 & +1 & +1 \\ +1 & -1 & -1 & +1 & -1 & +1 & +1 & -1 \end{bmatrix}$$

□ Problem : H may actually produce much larger values

- $$D = \begin{bmatrix} \pm 1 & & \\ & \ddots & \\ & & \pm 1 \end{bmatrix}$$

Another problem



What happens if we aggregate a big value with small ones ?

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

Claim: $\max\limits_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

By construction: $u_1 = \sum_{i=1}^{d} \pm d^{-1/2} x_i$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

By construction: $u_1 = \sum_{i=1}^{d} \pm d^{-1/2} x_i$

$\mathrm{E}[e^{sdu_1}]$ ← Moment generating function

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

By construction: $u_1 = \sum_{i=1}^{d} \pm d^{-1/2} x_i$

$$\mathrm{E}[e^{sdu_1}] = \prod_{i=1}^{d} \mathrm{E}[e^{sd(\pm d^{-1/2})x_i}]$$

$$= \prod_{i=1}^{d} (\tfrac{1}{2}e^{s\sqrt{d}x_i} + \tfrac{1}{2}e^{-s\sqrt{d}x_i})$$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

By construction: $u_1 = \sum_{i=1}^{d} \pm d^{-1/2} x_i$

$$\begin{aligned}
\mathrm{E}[e^{sdu_1}] &= \prod_{i=1}^{d} \mathrm{E}[e^{sd(\pm d^{-1/2})x_i}] \\
&= \prod_{i=1}^{d} cosh(s\sqrt{d}x_i) \\
&\leq \prod_{i=1}^{d} e^{s^2 d x_i^2/2} \qquad \text{(using } cosh(x) \leq e^{x^2/2}\text{)} \\
&\leq e^{s^2 d \|x\|_2^2/2}
\end{aligned}$$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

$\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d \|x\|_2^2/2}$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

$$\Pr[|u_1| \geq s) = \Pr[u_1 \geq s] + \Pr[|u_1 \leq s]$$

By symmetry: $\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

$\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d \|x\|_2^2 / 2}$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

$$\Pr[|u_1| \geq s) = \Pr[u_1 \geq s] + \Pr[|u_1 \leq s]$$

By symmetry: $\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$

$$\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$$
$$\leq 2\mathrm{E}[e^{sdu_1}]/e^{s^2 d}$$

(Markov's inequality)

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

$\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d\|x\|_2^2/2}$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

$$\Pr[|u_1| \geq s) = \Pr[u_1 \geq s] + \Pr[|u_1 \leq s]$$

By symmetry: $\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$

$$\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$$

$$\leq 2\mathrm{E}[e^{sdu_1}]/e^{s^2 d}$$

$$\text{(Markov's inequality)}$$

$$\leq 2e^{s^2 d(\|x\|_2^2/2-1)}$$

Must-remember box

$$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$$

$$u := HDx$$

$$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1\rangle}$$

$$D_{i,i} = \pm 1$$

$$\boxed{\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d\|x\|_2^2/2}}$$

Claim: $\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$

$$\Pr[|u_1| \geq s) = \Pr[u_1 \geq s] + \Pr[|u_1 \leq s]$$

By symmetry: $\Pr[|u_1| \geq s) = 2\Pr[e^{sdu_1} \geq e^{s^2 d}]$

$$\begin{aligned}\Pr[|u_1| \geq s) &= 2\Pr[e^{sdu_1} \geq e^{s^2 d}] \\ &\leq 2\mathrm{E}[e^{sdu_1}]/e^{s^2 d} \\ &\qquad\qquad \text{(Markov's inequality)} \\ &\leq 2e^{s^2 d(\|x\|_2^2/2-1)} \\ &\leq 2e^{-s^2 d/2} \quad \text{(assuming } \|x\|_2 = 1)\end{aligned}$$

$$\Pr[|u_1| \geq d^{-1/2}\sqrt{\log(40n)}] \leq 2e^{-\log(40n)/2}$$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1\rangle}$

$D_{i,i} = \pm 1$

$\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d\|x\|_2^2/2}$

Claim: $$\max_{\substack{x \in X \\ \|x\|_2 = 1}} \|HDx\|_\infty = O(d^{-1/2}\sqrt{\log(n)})$$

$$\begin{aligned}
\Pr[|u_1| \geq d^{-1/2}\sqrt{\log(40n)}] &\leq 2e^{-\log(40n)/2} \\
&\leq 1/(20nd) \\
\Pr[\|u\|_\infty \geq d^{-1/2}\sqrt{\log(40n)}] &\leq 1/(20n) \\
\Pr[\exists x \in X, \|HDx\|_\infty \geq d^{-1/2}\sqrt{\log(40n)}] &\leq 1/20
\end{aligned}$$

Must-remember box

$f : \mathbb{R}^d \to \mathbb{R}^k, k \ll d$

$u := HDx$

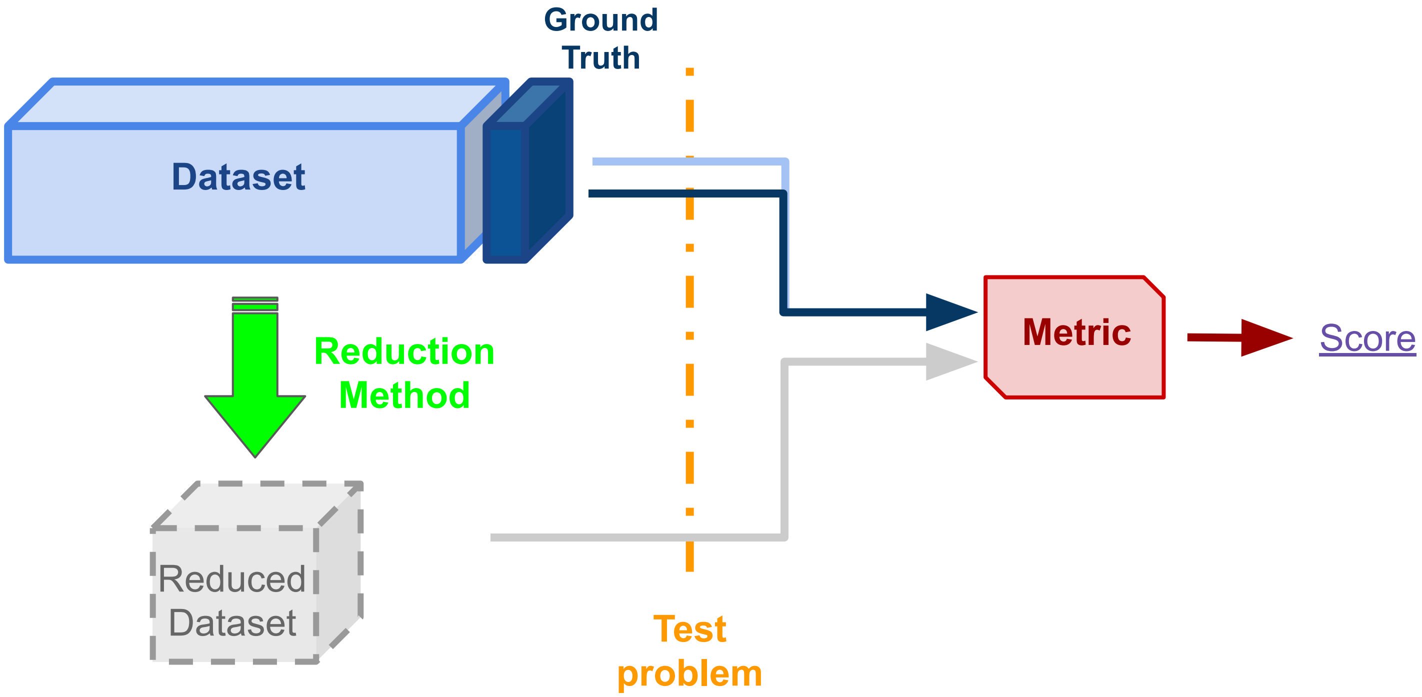
$H_{i,j} = d^{-1/2}(-1)^{\langle i-1, j-1 \rangle}$

$D_{i,i} = \pm 1$

$\mathrm{E}[e^{sdu_1}] \leq e^{s^2 d \|x\|_2^2/2}$

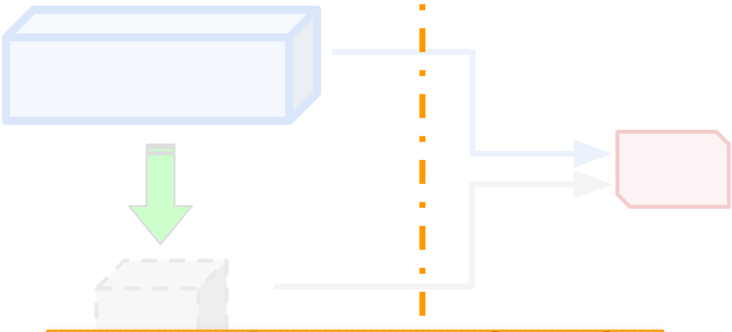
Experiments’ Design

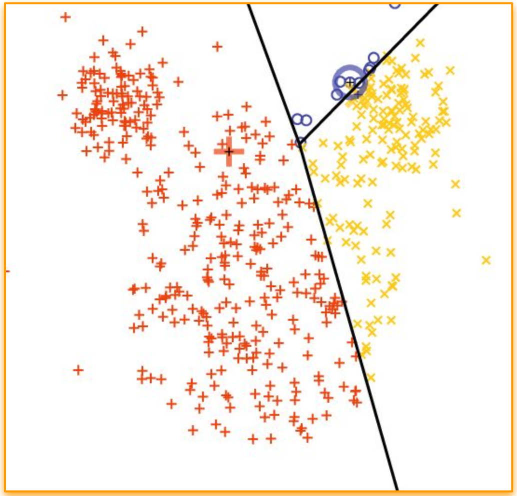
We assessed the goodness of some dim. reduction methods

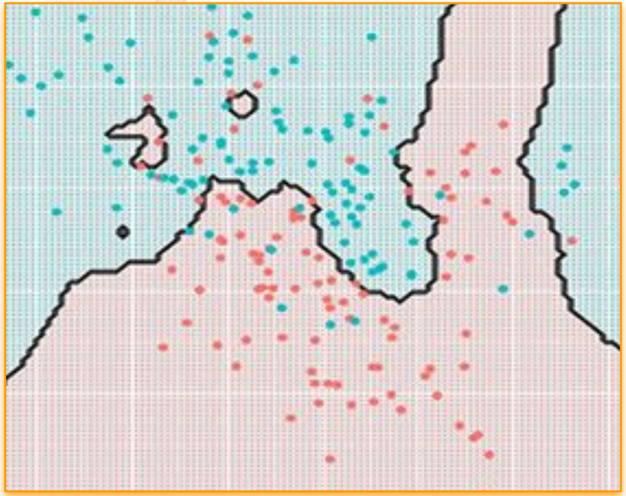


Test problems

- K-means Clustering
  - Elliptical clusters
- KNN Clustering
  - Complex shaped clusters

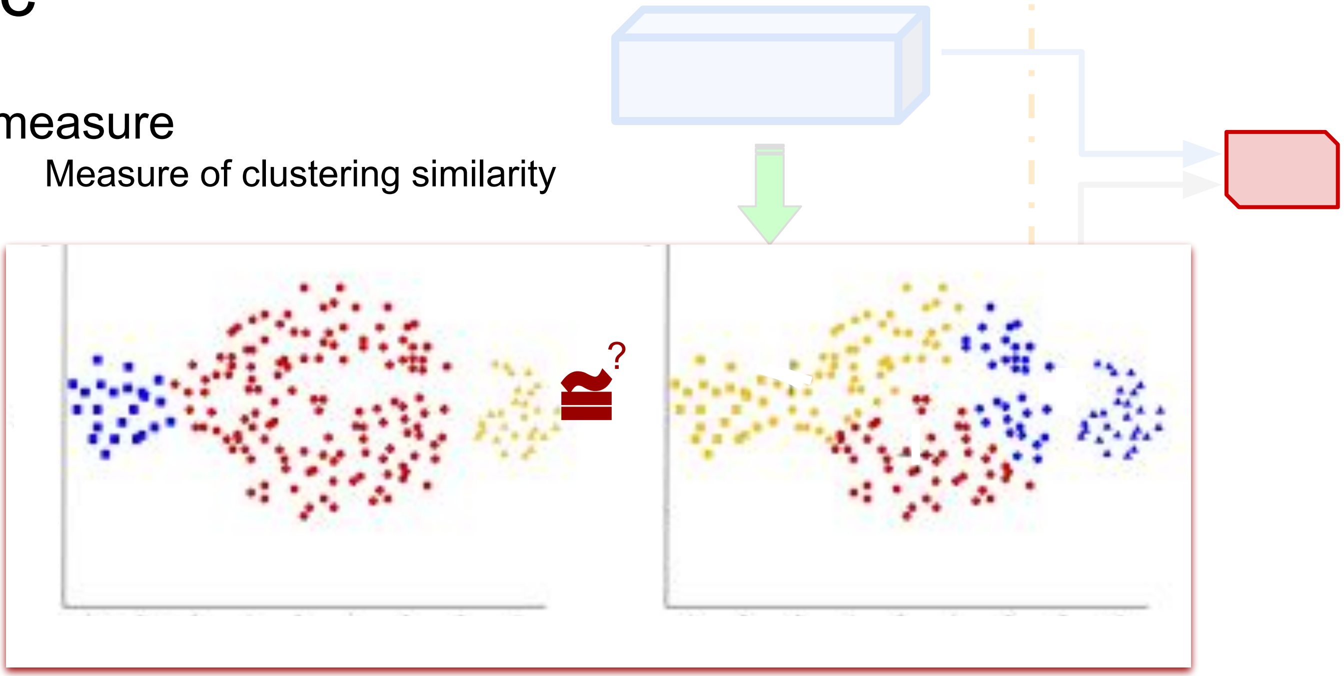






Metric

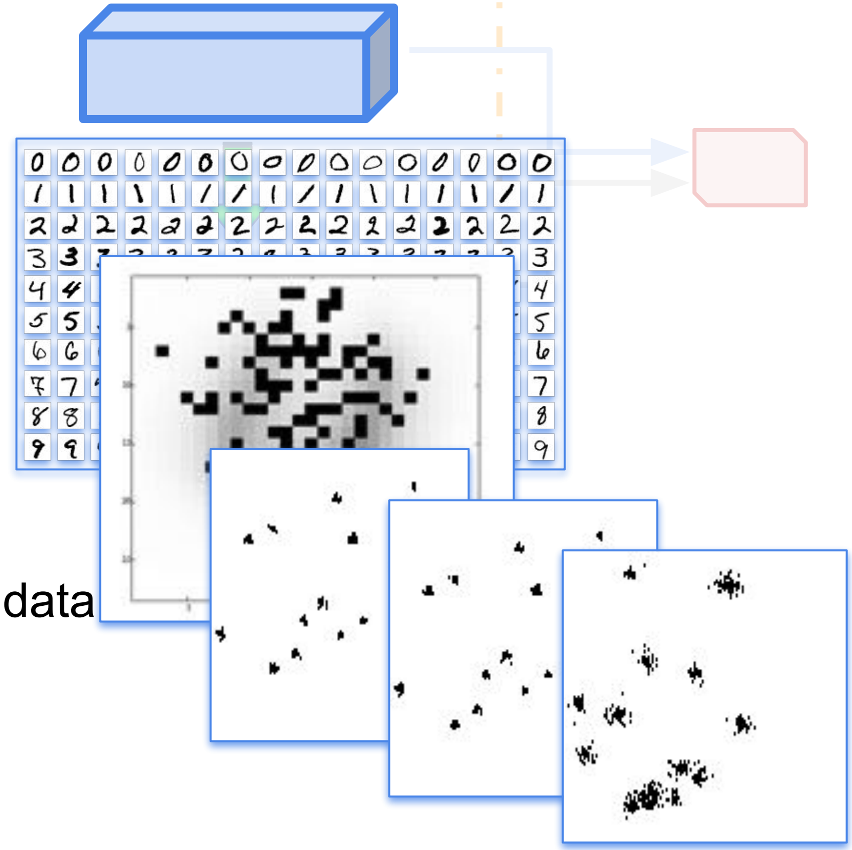
- v-measure
  - Measure of clustering similarity



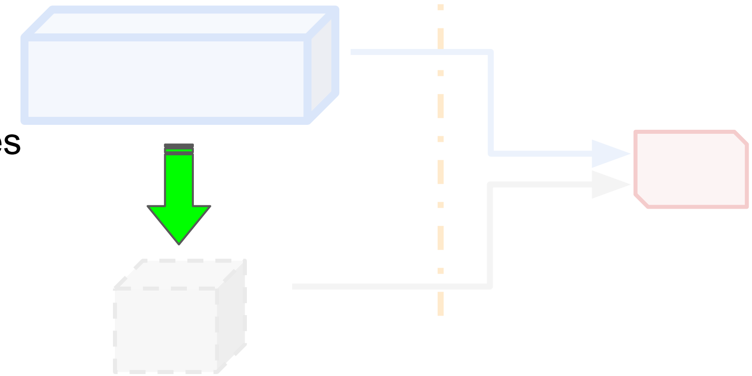
[4] Andrew Rosenberg, Julia Hirschberg, V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007, pp. 410–420

Datasets

- MNIST
  - *d = 28*28 = 784*
  - *N = 10’000*
  - *# clusters = 10*
- GISETTE
  - *d = 5’000*
  - *N = 7’000*
  - *# clusters = 2*
- Synthetic high-dimensional data
  - *d = …, 256, 512, 1024*
  - *N = 1024*
  - *# clusters = 16*

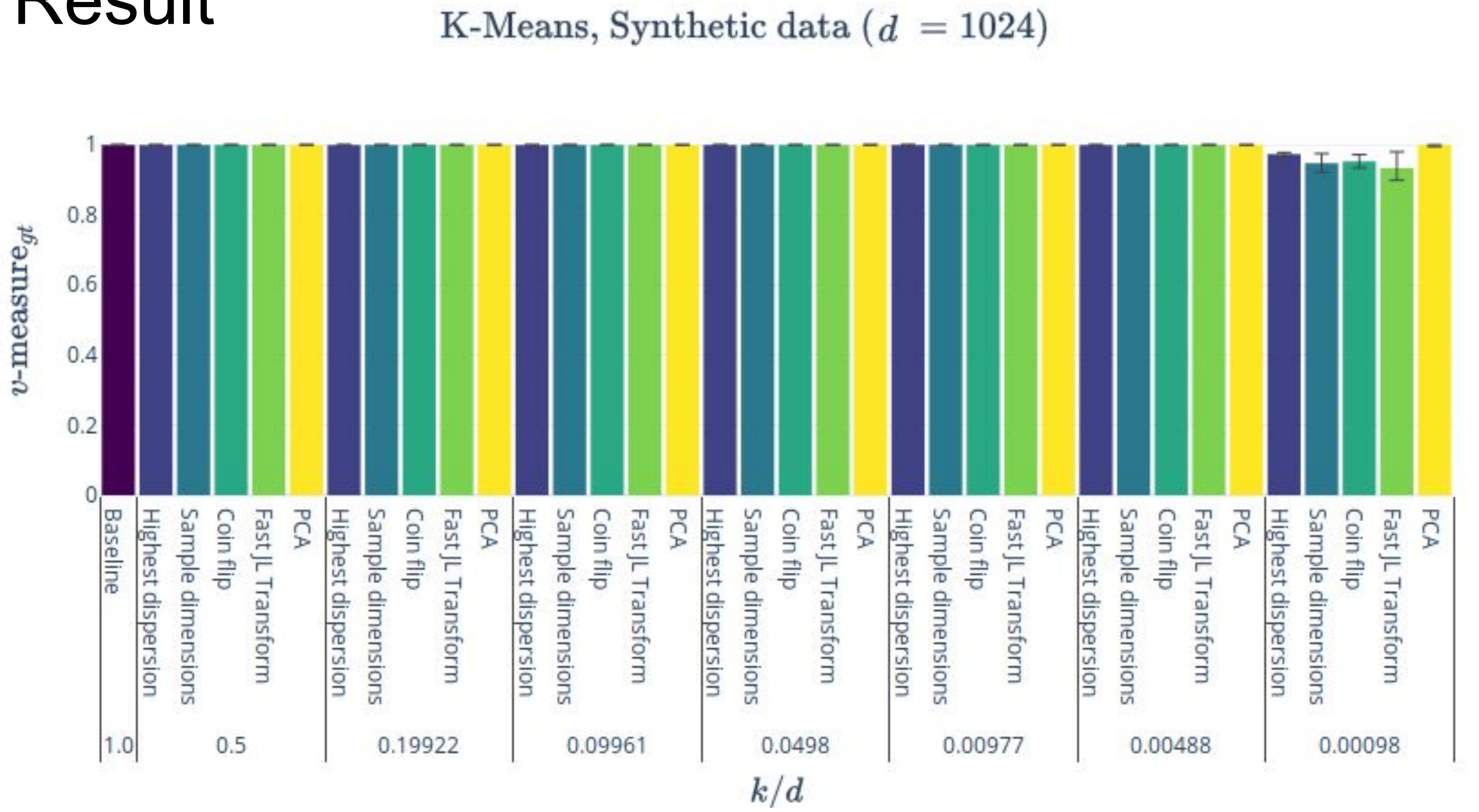


Methods

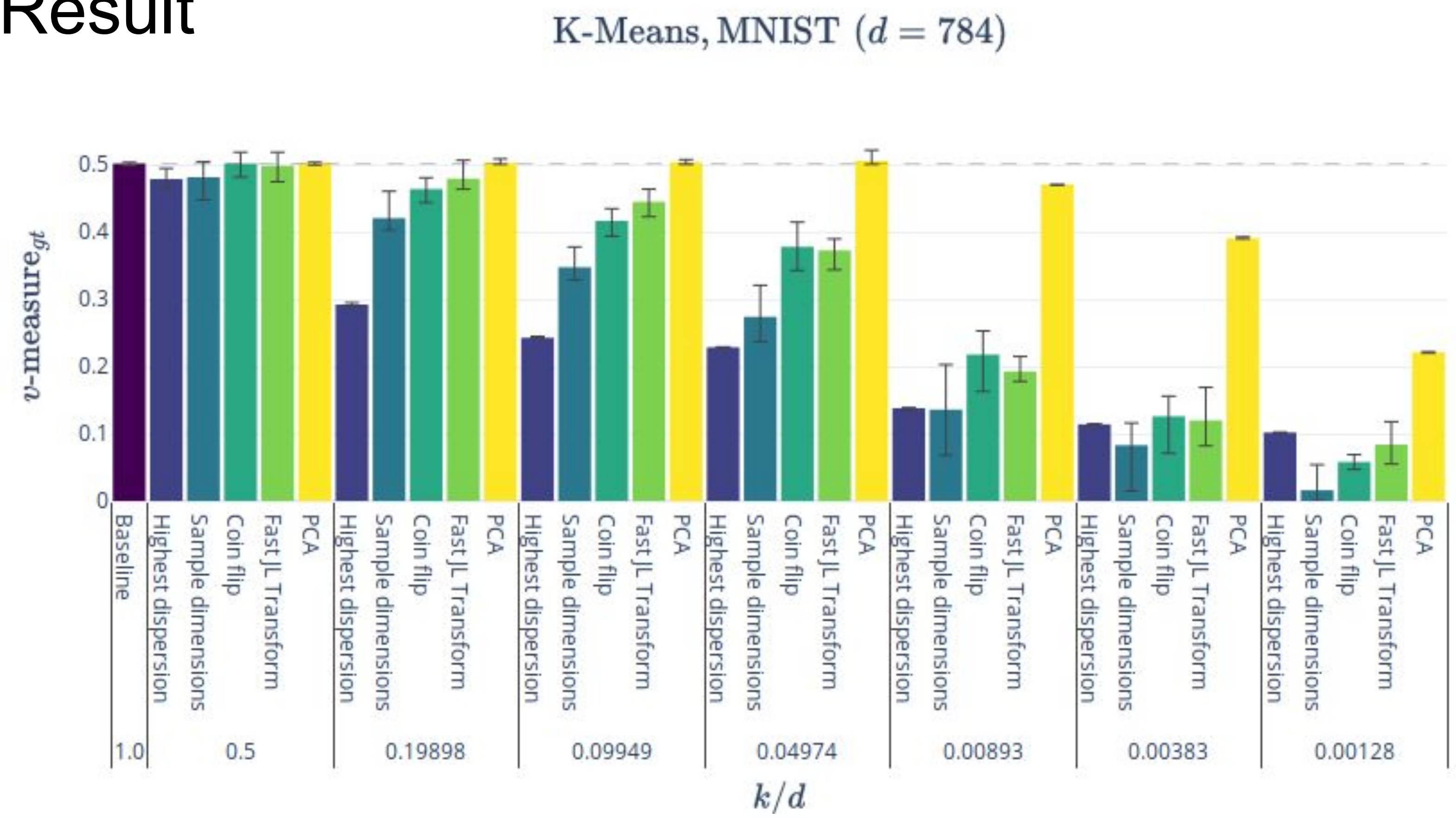


- Low variance filter
  - Select the *k* most variant variables
- Sampling variables
  - Random sample *k* variables
- Coin Flip method
  - Linear project on a random *k*-hyperplane
- Fast JL Transform
  - Random projection with improved bounds
- Principal Component Analysis
  - Presumably explained about 20 minutes ago

Result

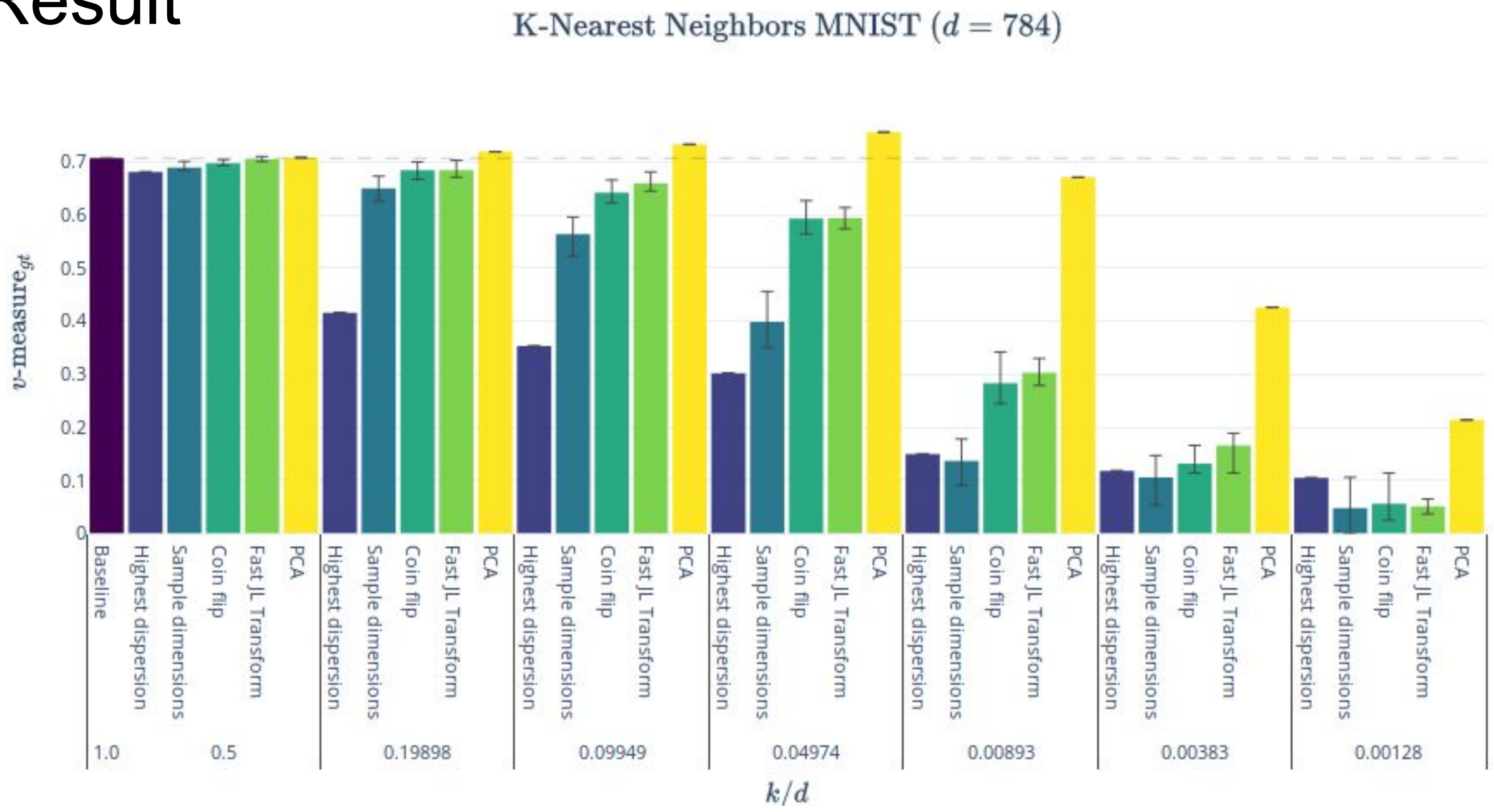

K-Means, Synthetic data ($d = 1024$)
v-measure$_{gt}$
1
0.8
0.6
0.4
0.2
0
Baseline
Highest dispersion
Sample dimensions
Coin flip
Fast JL Transform
PCA
1.0
0.5
0.19922
0.09961
0.0498
0.00977
0.00488
0.00098
k/d

Result



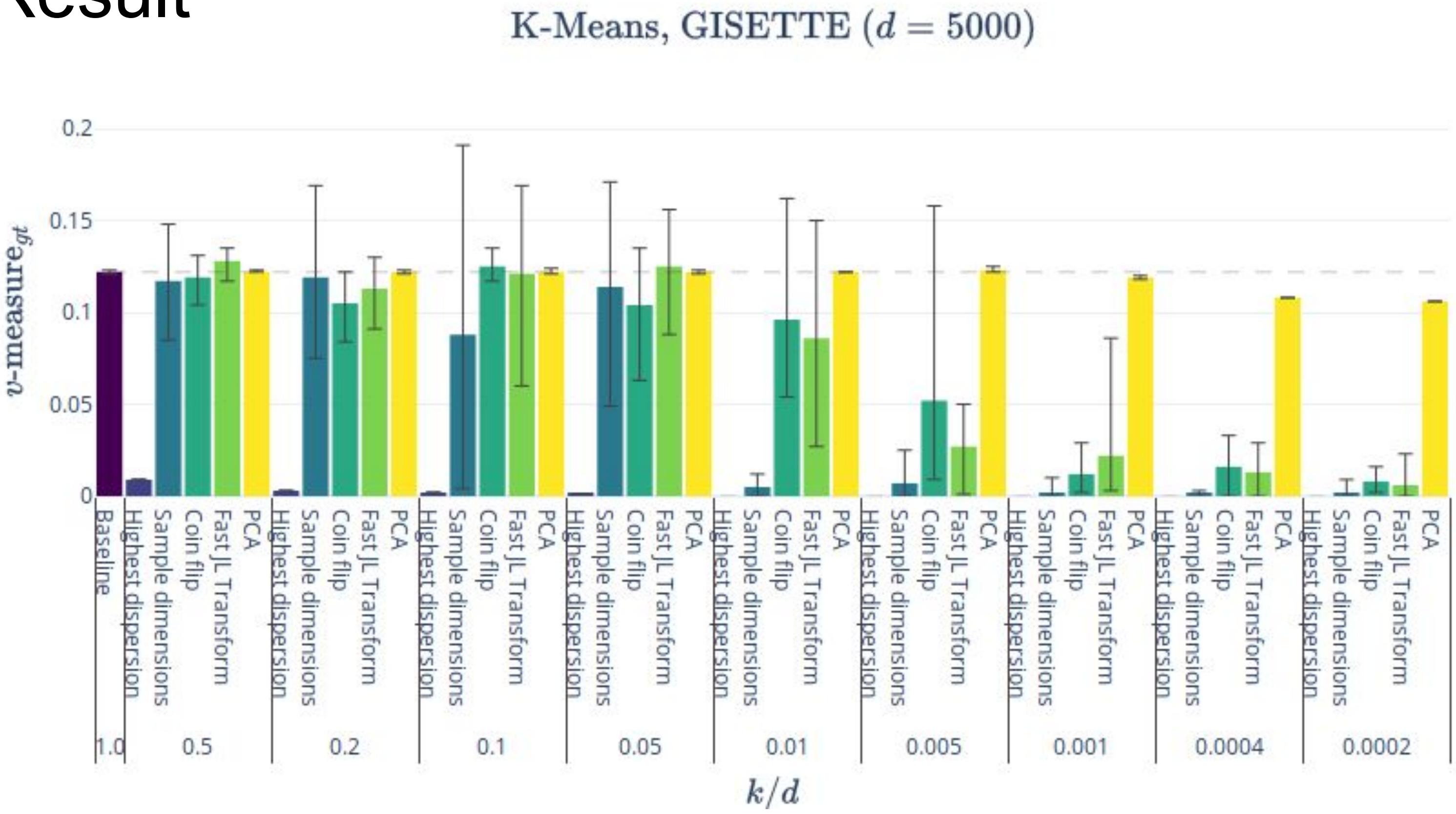
K-Means, MNIST $(d = 784)$
v-measure$_{gt}$
0.5
0.4
0.3
0.2
0.1
0
Baseline
Highest dispersion
Sample dimensions
Coin flip
Fast JL Transform
PCA
1.0
0.5
0.19898
0.09949
0.04974
0.00893
0.00383
0.00128
k/d

Result



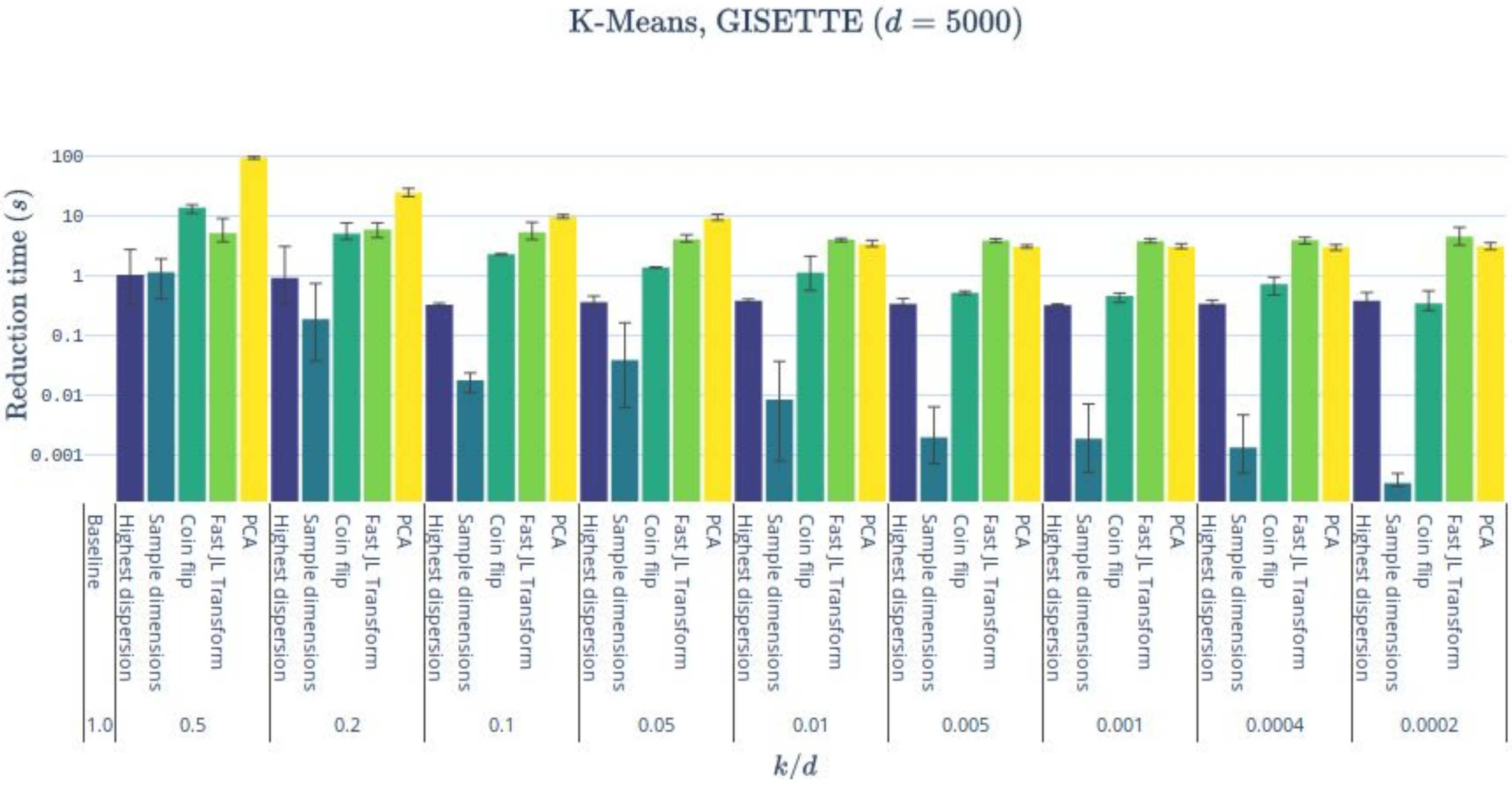
K-Nearest Neighbors MNIST ($d = 784$)
v-measure$_{gt}$
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
Baseline
Highest dispersion
Sample dimensions
Coin flip
Fast JL Transform
PCA
1.0
0.5
0.19898
0.09949
0.04974
0.00893
0.00383
0.00128
k/d

Result



K-Means, GISETTE ($d = 5000$)
v-measure$_{gt}$
0.2
0.15
0.1
0.05
0
Baseline
Highest dispersion
Sample dimensions
Coin flip
Fast JL Transform
PCA
1.0
0.5
0.2
0.1
0.05
0.01
0.005
0.001
0.0004
0.0002
k/d

Result


K-Means, GISETTE ($d = 5000$)
Reduction time (s)
100
10
1
0.1
0.01
0.001
Baseline
Highest dispersion
Sample dimensions
Coin flip
Fast JL Transform
PCA
1.0
0.5
0.2
0.1
0.05
0.01
0.005
0.001
0.0004
0.0002
k/d

Bibliography

[1] W.B. Johnson, J. Lindenstrauss, Extensions of Lipschitz mappings into a Hilbert space, Conference in modern analysis and probability, New Haven, CI, 1982, Amer. Math. Soc., Providence, RI, 1984, pp. 189-206

[2] D. Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. J. Comput. Syst. Sci., 66(4):671-687, 2003

[3] Ailon, N., Chazelle, B.: The fast Johnson-Lindenstrauss transform and approximate nearest neighbors. SIAM J. Comput. 39(1), 302-322 (2009)

[4] Andrew Rosenberg, Julia Hirschberg, V-Measure: A Conditional Entropy-Based External Cluster Evaluation Measure, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), 2007, pp. 410–420