Representation

Hendrik Santoso Sugiarto

IBDA2032 – *Artificial Intelligence*

Capaian Pembelajaran

- Konsep Representasi
- Reduksi Dimensi Linear
- Sparse Representation
- Reduksi Dimensi Nonlinear
- Autoencoders
- Embedding

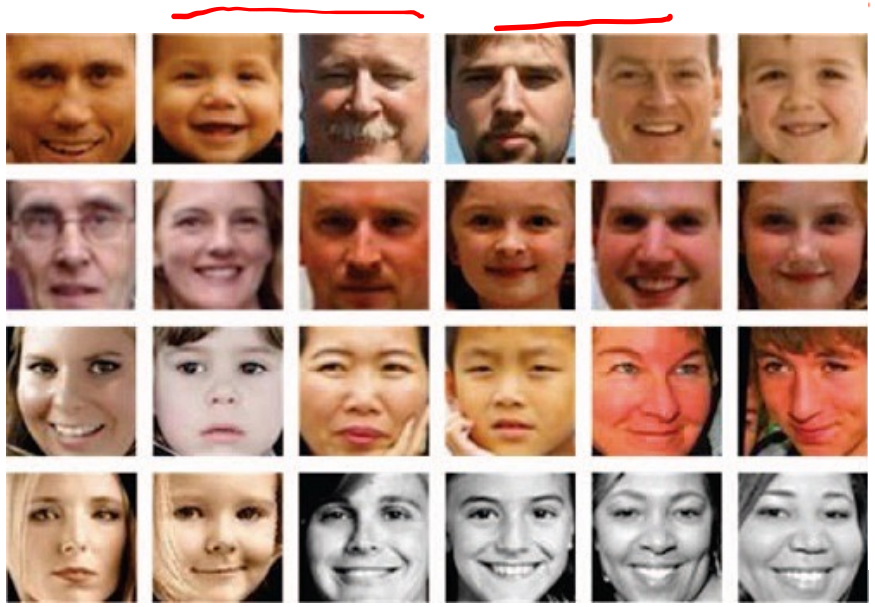
Representation

Unsupervised Learning

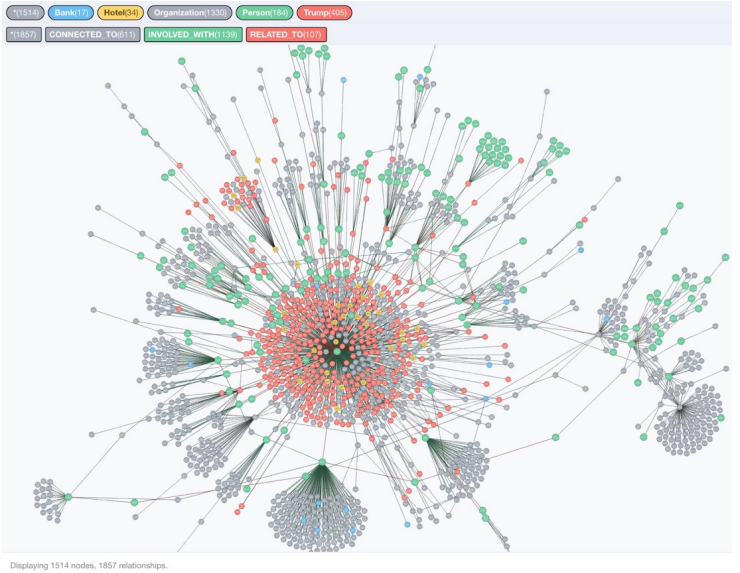
- Bentuk formal:
  - Input: $\boldsymbol{x} \in \mathcal{X} \in \mathbb{R}^n$
  - Output: $y \in \mathcal{Y} \in \begin{cases} \{1,2,\dots,K\} \rightarrow clustering\ K\ grup \\ \mathbb{R}^K \rightarrow embedding\ K\ dimension \end{cases}$
  - Target function: $f: \mathcal{X} \rightarrow \mathcal{Y}\ (unknown)$
  - Training data: $D = \{(\boldsymbol{x}^{(1)}), (\boldsymbol{x}^{(2)}), \dots, (\boldsymbol{x}^{(m)})\}$
  - Hypothesis: $h: \mathcal{X} \rightarrow \mathcal{Y}$
  - Hypothesis space: $h \in \mathcal{H}$

Representasi

- Teknik transformasi data input menjadi representasi yang dapat digunakan untuk machine learning task secara efektif
- Teknik: reduksi dimensi, sparse coding/representation, embedding, autoencoder
- Data pada umumnya bersifat high dimensional: image, text, gene expression, graph



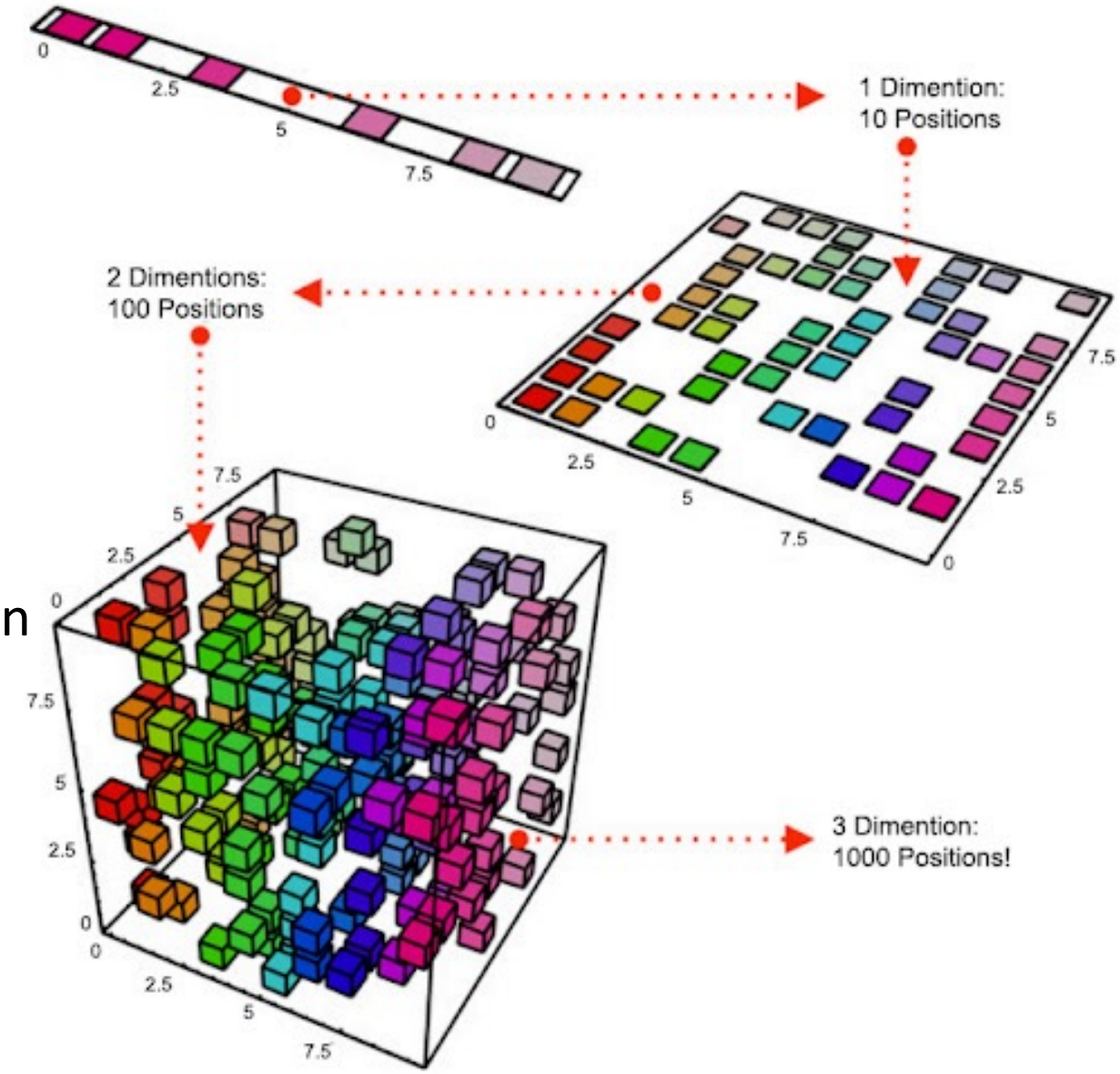
	title	label
44	Which Zodiac Should You Date Based On Your Fav...	clickbait
14	Guidant announces more defibrillator problems	not-clickbait
18	Many nations offer material aid to hurricane v...	not-clickbait
15	Woman Files Complaint Against Bernie Williams	not-clickbait
35	21 Reasons Why No One Should Have New York Values	clickbait
26	Washington Train Crash Prompts Safety Warning	not-clickbait
42	St Paul's cathedral to shut down following 'Oc...	not-clickbait
24	French tourists killed in California bus rollo...	not-clickbait
32	17 Things You Only See At A Historically Black...	clickbait
33	Which One Direction Music Video Is Your Favorite	clickbait



CALVIN INSTITUTE OF TECHNOLOGY

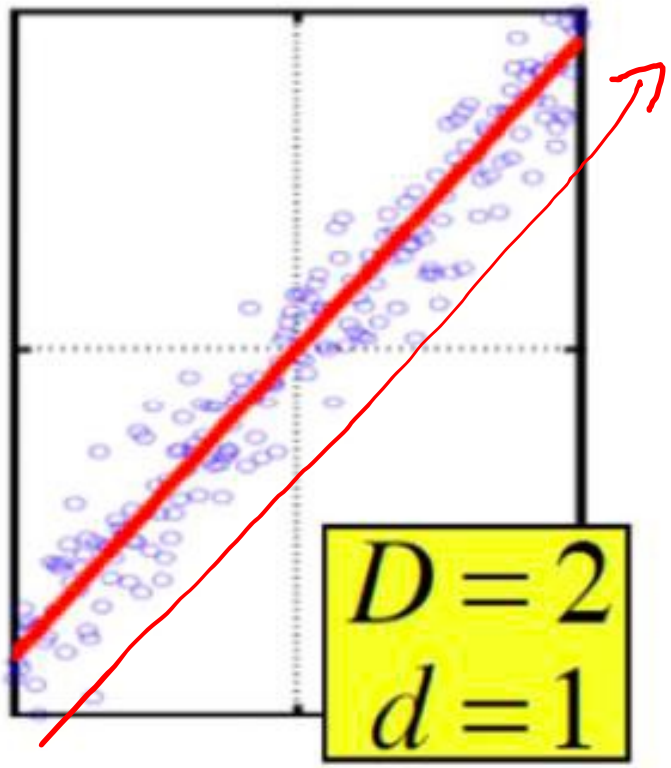
Curse of Dimensionality

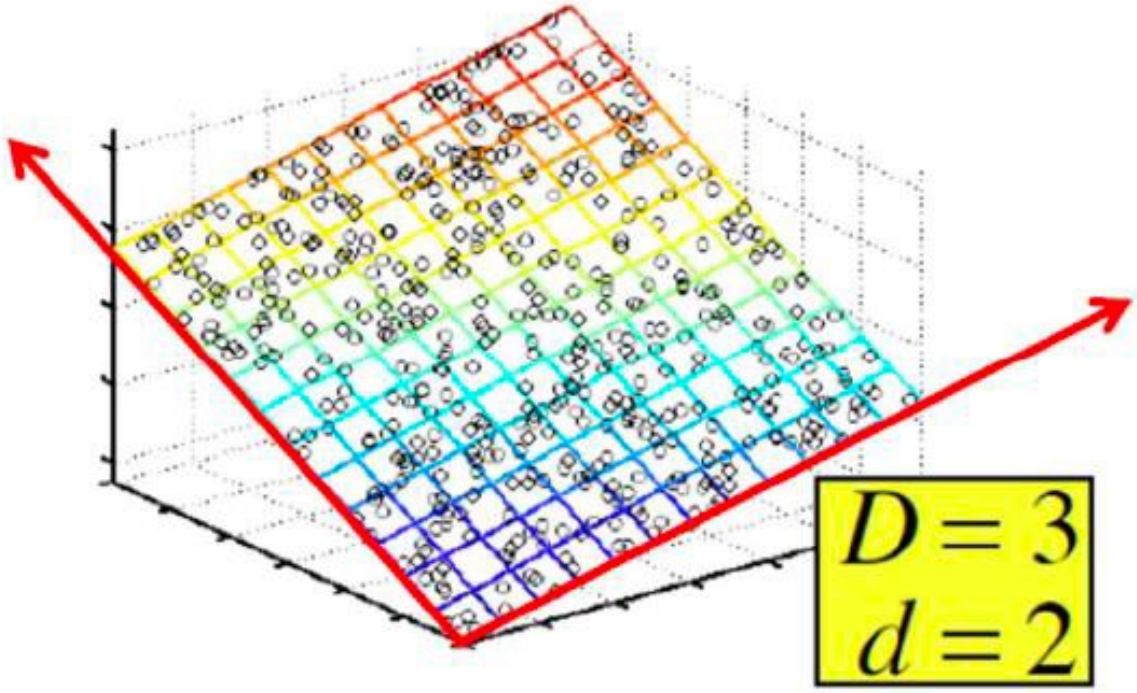
- Banyak dimensi di dalam ruang fitur → ukuran dari dimensi tersebut sangat besar
- Titik yang kita miliki di dalam dimensi tersebut (baris) seringkali merepresentasikan sampel kecil saja
- Hal ini dapat secara drastik mengurangi performa dari algoritma pembelajaran mesin → curse of dimensionality



Reduksi Dimensi

- Asumsi: data sebenarnya terletak di dimensi yang lebih kecil
- Contoh:





CALVIN INSTITUTE OF TECHNOLOGY

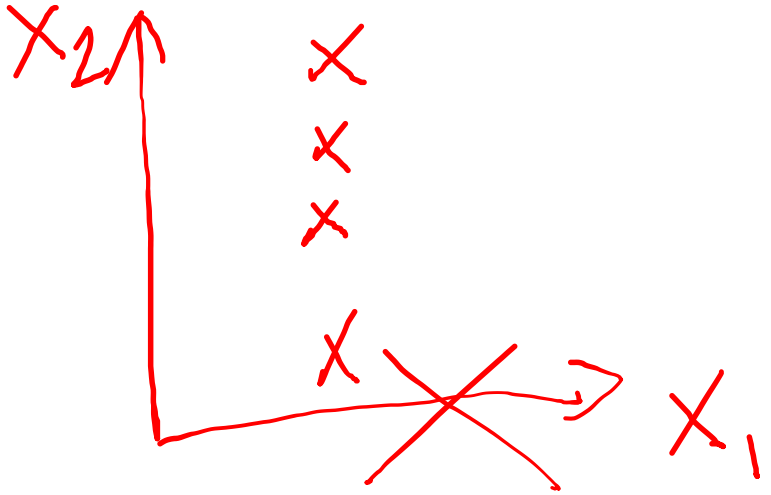
Dimension Reduction

- Beban komputasi: compress data→efisien (baik latensi maupun memori), lebih mudah belajar
- Visualisasi: strukturnya mudah divisualisasi
- Sederhana: setiap informasi berguna, mengurangi noise
- Anomaly detection: mudah mendeteksi data yang ”berbeda/outlier”
- Contoh: PCA, MDS, Non-linear Reduction

Reduksi Dimensi Linear

Metode Seleksi Fitur

- Metode seleksi fitur merupakan teknik reduksi dimensi yang paling umum.
- Biasanya dilakukan dengan menggunakan skor atau nilai statistik untuk menentukan fitur mana yang dipertahankan dan yang dibuang.
- Ada 3 metode umum:
  - wrapper methods: RFE (Recursive Features Elimination).
  - filter methods: Pearson's correlation dan Chi-Squared test.
  - intrinsic methods: Lasso.



Faktorisasi

- Teknik dari aljabar linear dapat digunakan untuk mengurangi dimensi
- Contohnya adalah penggunaan eigendecomposition dan singular value decomposition
- Setiap bagian dari matriks dapat diranking, dan subset dari bagian tersebut yang paling banyak memiliki subset akan dipilih
- Ada beberapa contoh, yang paling banyak digunakan adalah PCA (principal component analysis)
- Ilustrasi: https://setosa.io/ev/principal-component-analysis/

Faktorisasi

- Terdapat n data dengan d dimensi: $x^{(1)}, \dots, x^{(m)} \in \mathbb{R}^d$

$$X = \begin{pmatrix} | & & | \\ x^{(1)} & \dots & x^{(m)} \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times m}$$

- Kita ingin mereduksi dimensi dari d menjadi k
- Pilih k vector $u_1, \dots, u_k$ dimana:

$$U = \begin{pmatrix} | & & | \\ u_1 & \dots & u_k \\ | & & | \end{pmatrix} \in \mathbb{R}^{d \times k}$$

- Untuk setiap u_j hitung similarity $z_j = u_j^T x$
- Proyeksikan x menuju $z = (z_1, \dots, z_k)^T = U^T x$
- Bagaimana cara memilih U?

Principal Component Analysis (PCA)

- U dapat dipakai untuk 2 tujuan:
  - Encode: $z = U^T x$
  - Decode: $\tilde{x} = Uz$
- Kita ingin selisih rekonstruksi $||x - \tilde{x}||$ sekecil mungkin
- Cost function dari PCA: meminimalkan total squared reconstruction error

$$\min_{U \in \mathbb{R}^{d \times k}} \sum_{i=1}^{m} \left|\left| x^{(i)} - UU^T x^{(i)} \right|\right|^2$$

- Solusi untuk U dapat diperoleh dengan menyelesaikan dekomposisi eigen dari matriks kovarian

Algoritma PCA

- $X \in \mathbb{R}^{m \times d}$ adalah matriks data
- Recenter: kurangkan dengan rata-rata untuk setiap kolom

$$\tilde{X} = X - \bar{X}$$

- Hitung matriks kovarian Σ dari $\tilde{X}$

$$\Sigma = \frac{1}{n} \tilde{X} \tilde{X}^T$$

- Temukan k eigenvalue tertinggi beserta eigen vectornya

$$\Sigma v = \lambda v$$

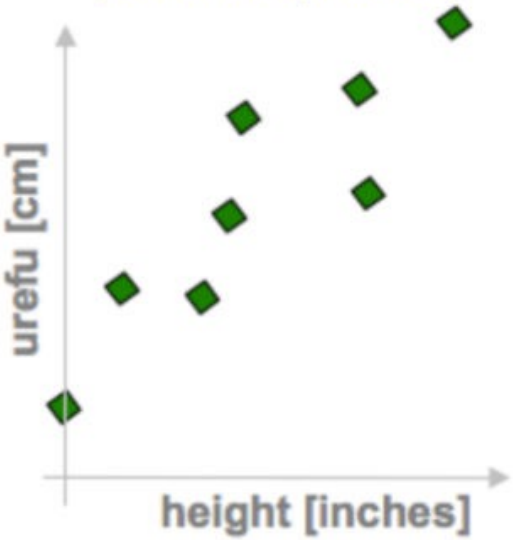
- Bentuk U dengan memilih k eigenvectors
- Gunakan U untuk memproyeksikan data ke dimensi yang lebih kecil:

$$z = U^T \tilde{x}$$

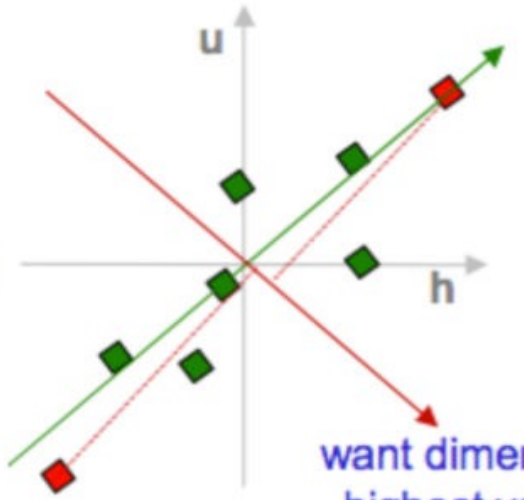
PCA in a nutshell

1. correlated hi-d data

("urefu" means "height" in Swahili)



2. center the points



want dimension of highest variance

3. compute covariance matrix

	h	u
h	2.0	0.8
u	0.8	0.6

$$\text{cov}(h,u) = \frac{1}{n}\sum_{i=1}^{n} h_i u_i$$

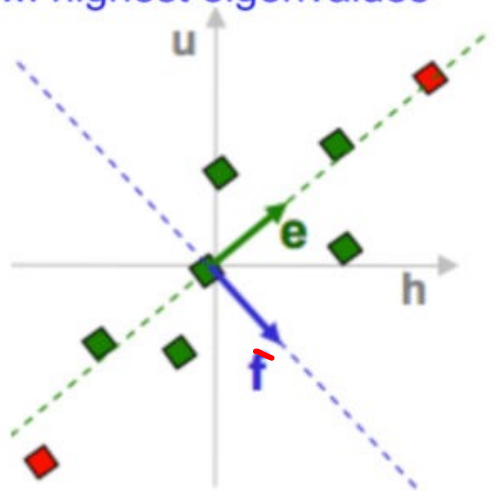
4. eigenvectors + eigenvalues

$$\begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix} \begin{pmatrix} e_h \\ e_u \end{pmatrix} = \lambda_e \begin{pmatrix} e_h \\ e_u \end{pmatrix}$$

$$\begin{pmatrix} 2.0 & 0.8 \\ 0.8 & 0.6 \end{pmatrix} \begin{pmatrix} f_h \\ f_u \end{pmatrix} = \lambda_f \begin{pmatrix} f_h \\ f_u \end{pmatrix}$$

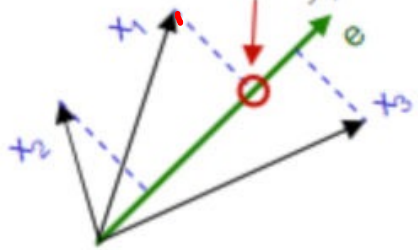
```
eig(cov(data))
```

5. pick m<d eigenvectors w. highest eigenvalues

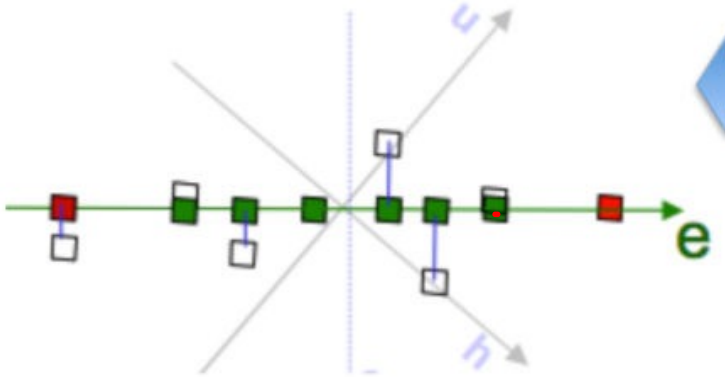


6. project data points to those eigenvectors

$$x_e' = x^T e = \sum_{j=1}^{d} x_{ij} e_j$$

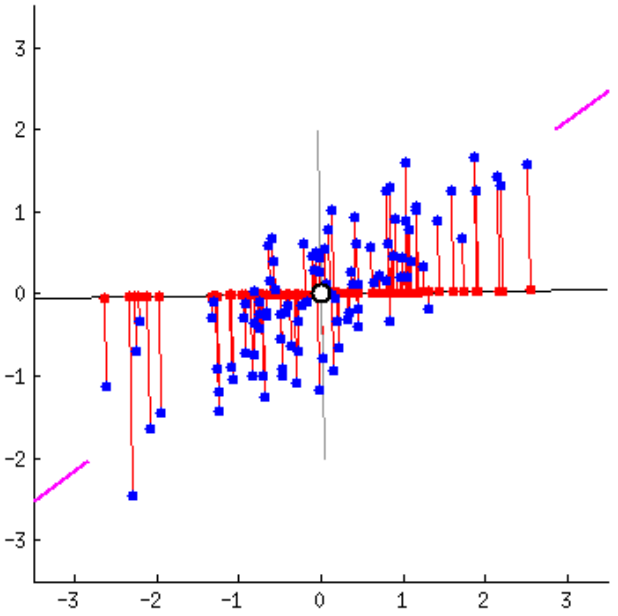


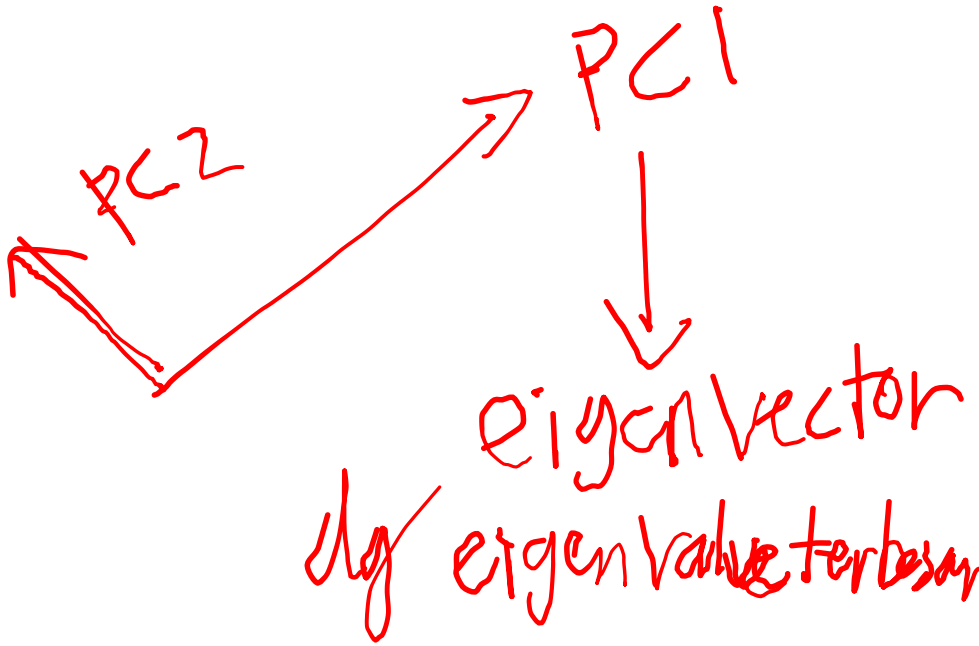
7. uncorrelated low-d data

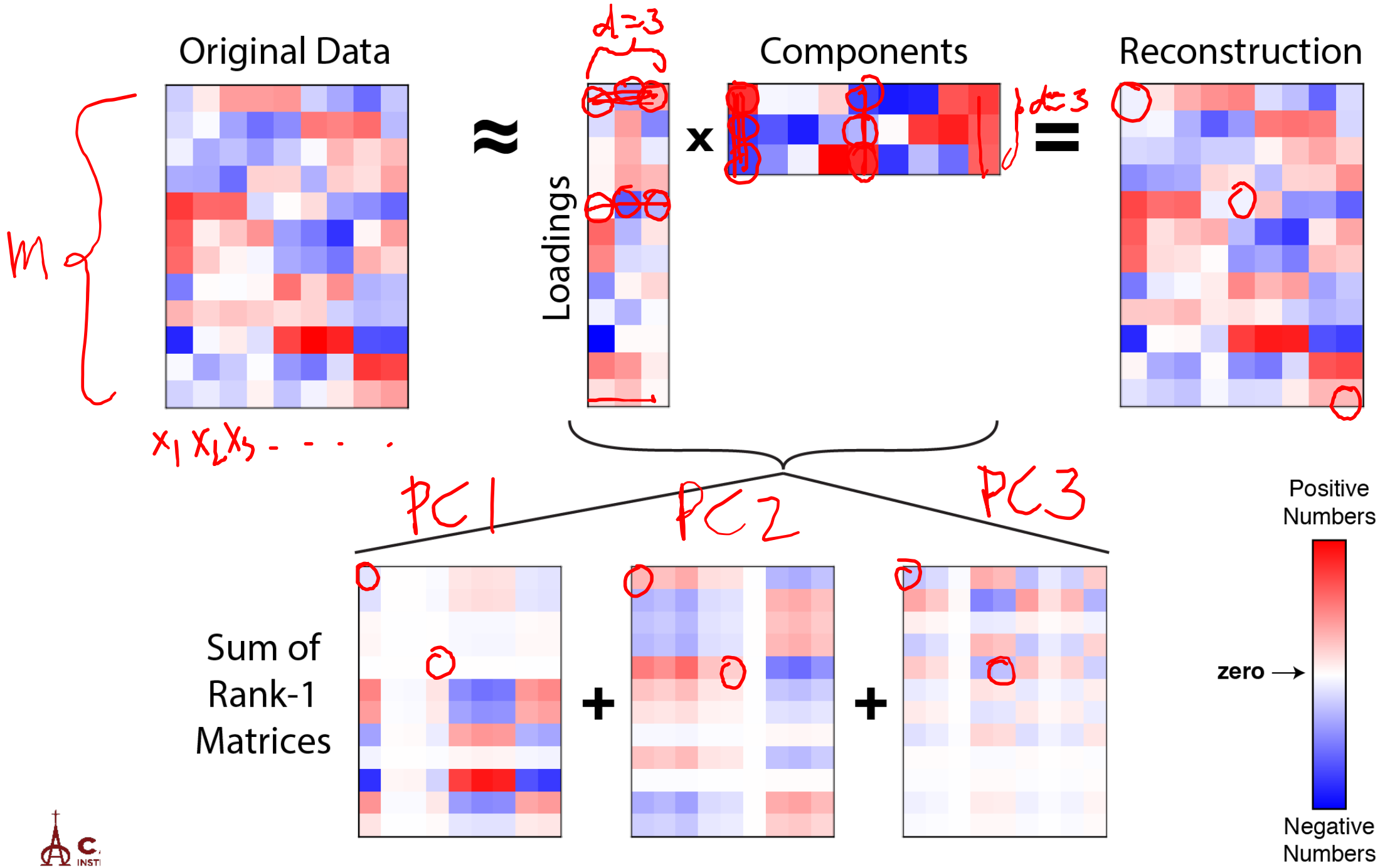


Principal Component Analysis

- PCA memproyeksikan data dengan n-fitur menuju subspace dengan ruang lebih kecil namun masih menjaga esensi data asal melalui variasi data terbesar

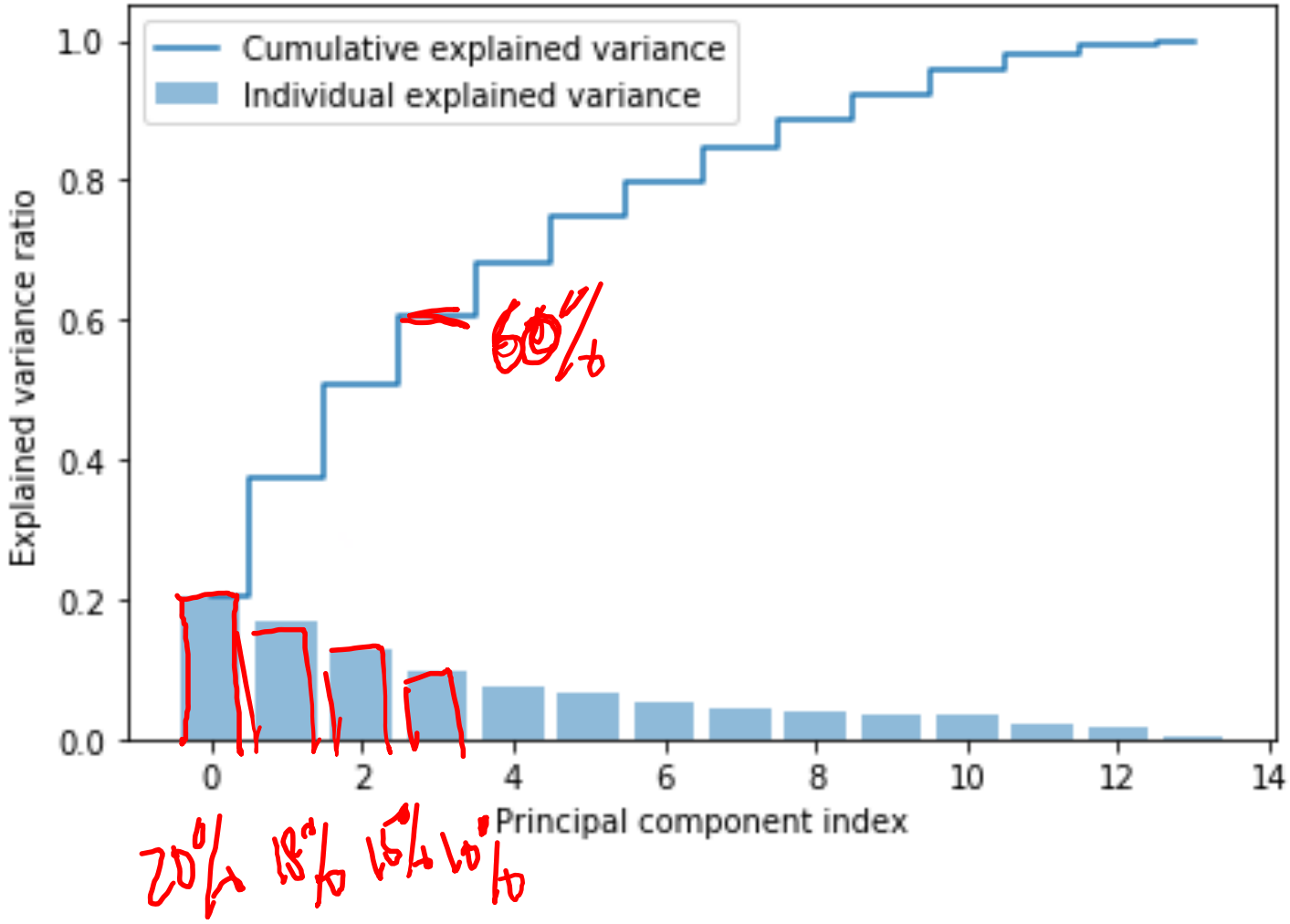


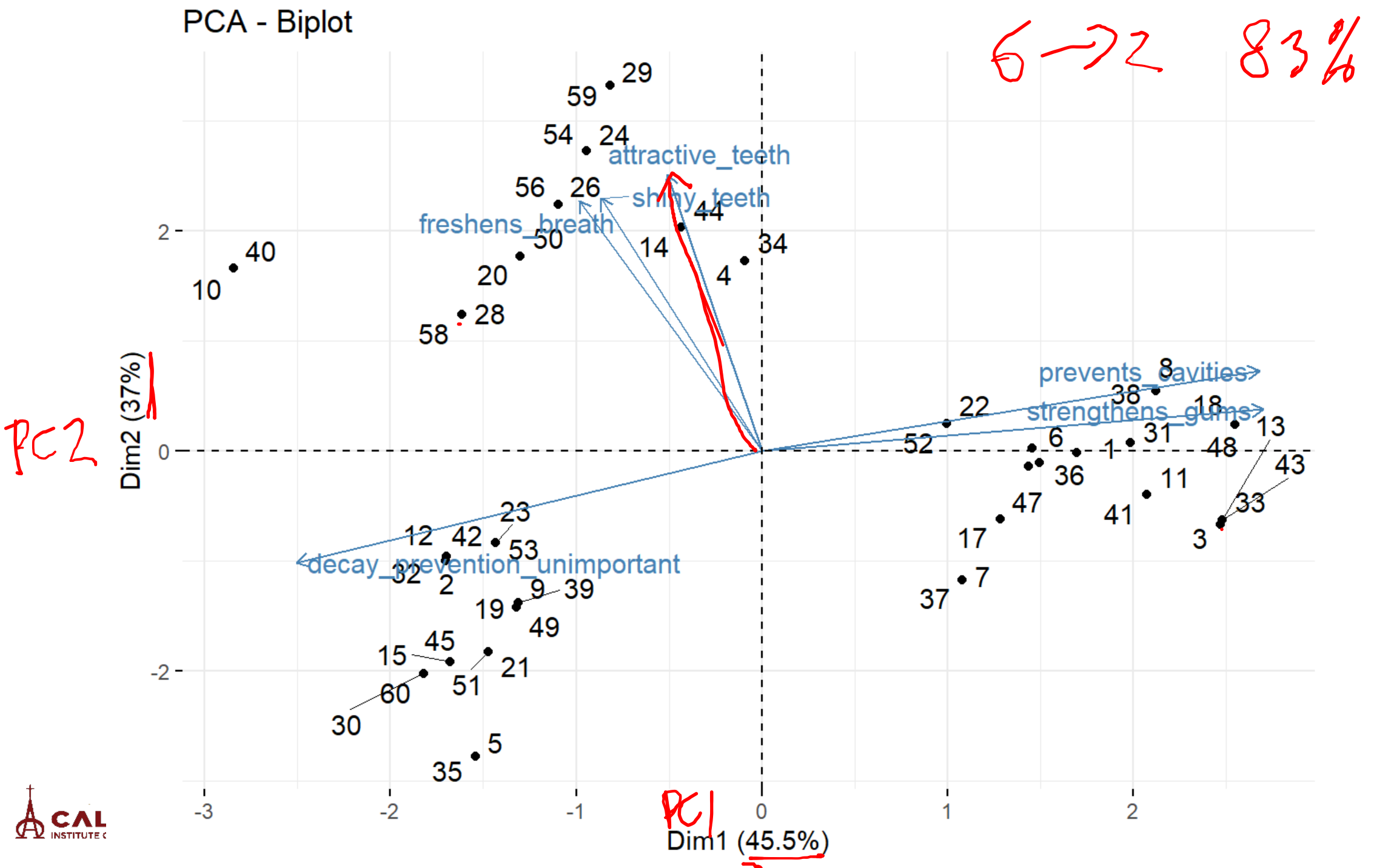



Original Data
≈
Loadings
x
Components
=
Reconstruction
PC1
PC2
PC3
Sum of Rank-1 Matrices
+
+
Positive Numbers
zero
Negative Numbers

Variance Explained

- Principal components: Sebagian besar informasi berada pada komponen pertama, diikuti komponen kedua, dst
- Maka kita dapat mereduksi data ke beberapa principal components utama tanpa kehilangan banyak informasi



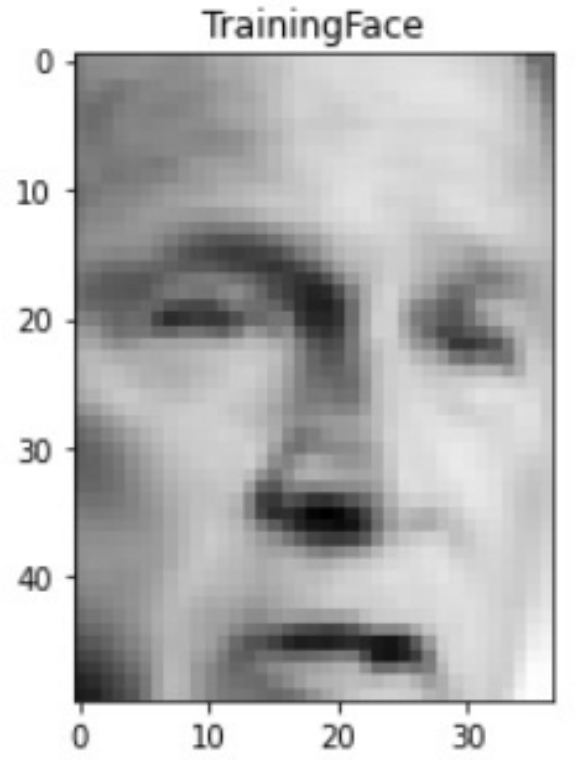

PCA - Biplot
attractive_teeth
shiny_teeth
freshens_breath
prevents_cavities
strengthens_gums
decay_prevention_unimportant
Dim2 (37%)
Dim1 (45.5%)
29
59
54
24
56
26
44
14
34
4
50
20
10
40
58
28
8
38
18
22
13
31
6
52
1
48
36
43
11
47
41
33
17
3
7
37
23
12
42
53
32
2
9
39
19
49
15
45
21
51
60
30
5
35
-3
-2
-1
0
1
2
2
0
-2

Aplikasi PCA

- Kompresi:
  - Mengurangi memori yang diperlukan untuk menyimpan data
  - Mempercepat algoritma pembelajaran
- Visualisasi:
  - Memproyeksikan data dalam 2D atau 3D
- Kegunaan lain:
  - Denoising
  - Menghindari overfitting (tapi tidak ideal)

CALVIN INSTITUTE OF TECHNOLOGY

EigenFaces



36

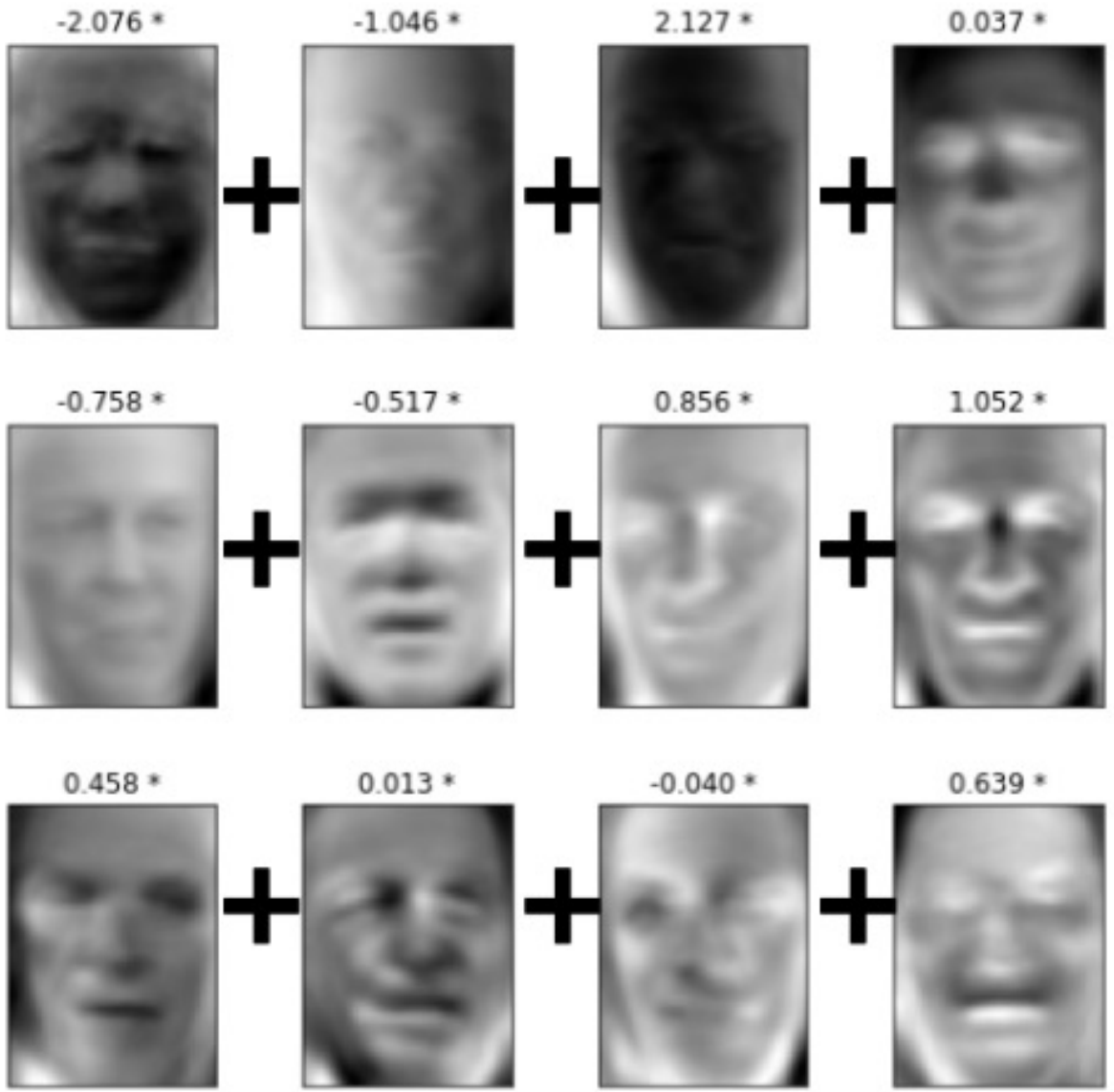
52

$2^{36 \times 52}$

=

PC1 PC2 PC3 PC4

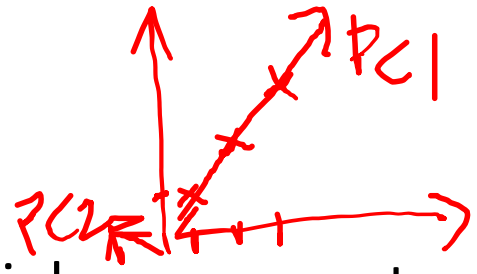
PC5



Kelemahan PCA

- Matriks kovarian dapat menjadi sangat besar:
  - Σ adalah matriks $d \times d$
  - Data dengan $d > 10000$ fitur cukup umum ditemukan
  - Perhitungan eigenvector dapat sangat lambat
  - Complexity $O(n \times d^2 + d^3)$
- Gunakan singular value decomposition (SVD)

Uji Pemahaman



- Berapa variance explained dari principle component pertama dari data berikut:
- $X = \begin{pmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \end{pmatrix}$ } 3 data

2 dimensi

$\bar{X} = (2, 4)$

$\tilde{X} = X - \bar{X} = \begin{pmatrix} -1 & -2 \\ 0 & 0 \\ 1 & 2 \end{pmatrix}$

$\Sigma = \frac{1}{2}\begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \end{pmatrix}\begin{pmatrix} -1 & -2 \\ 0 & 0 \\ 1 & 2 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 2 & 4 \\ 4 & 8 \end{pmatrix}$

$\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix} V = \lambda V \rightarrow \left[\begin{pmatrix} 1 & 2 \\ 2 & 4 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix}\right] V = 0$

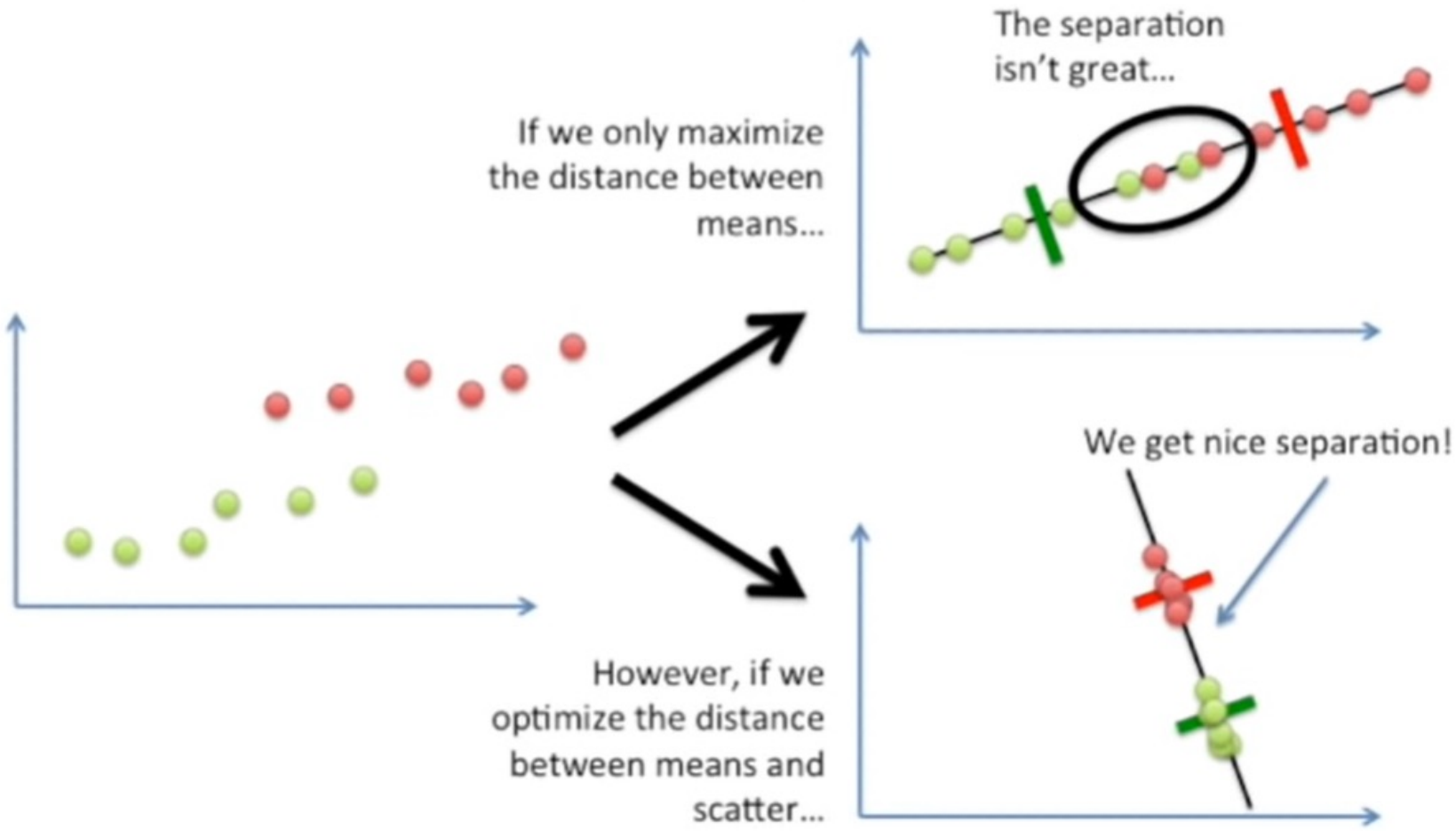
$\begin{vmatrix} 1-\lambda & 2 \\ 2 & 4-\lambda \end{vmatrix} = 0 \rightarrow \lambda = 5, \lambda = 0$

CALVIN INSTITUTE OF TECHNOLOGY

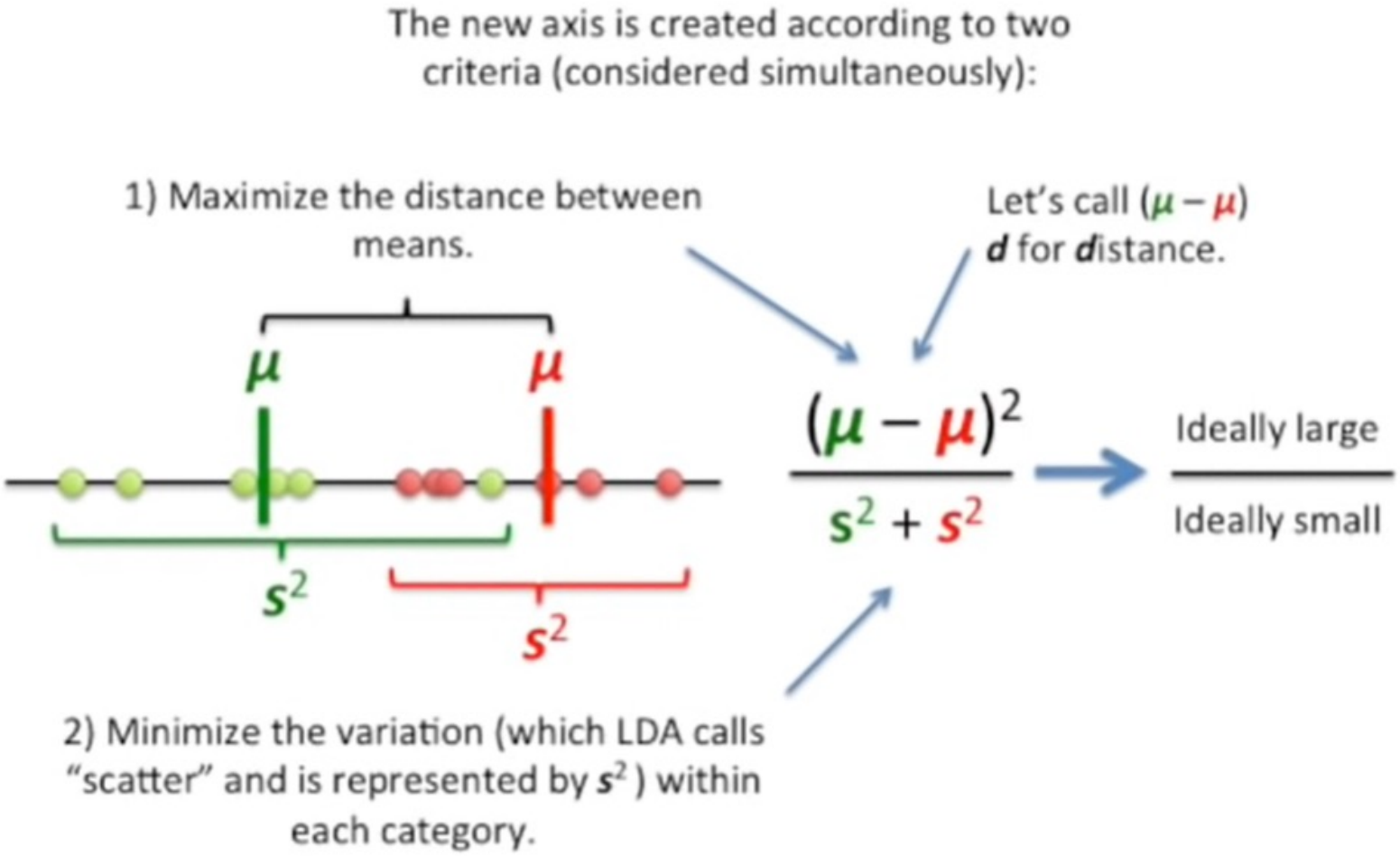
Linear Discriminant Analysis

- *Linear Discriminant Analysis* atau LDA merupakan algoritma pembelajaran mesin untuk klasifikasi multikelas.
- Jangan sampai tertukar dengan *Latent Dirichlet Allocation* (LDA) yang merupakan teknik reduksi dimensi untuk dokumen teks
- LDA berusaha untuk mencari cara terbaik memisahkan sampel di dalam data latih berdasarkan kelas mereka
- Secara spesifik, model ini akan mencari kombinasi linear dari variabel input untuk mendapatkan pemisahan maksimum untuk sampel antar kelas dan pemisahan minimum antar sampel dalam kelas yang sama
- Untuk melihat penjelasan lebih detail lihat: (dijelaskan pada slide berikutnya) https://www.youtube.com/watch?v=azXCzI57Yfc

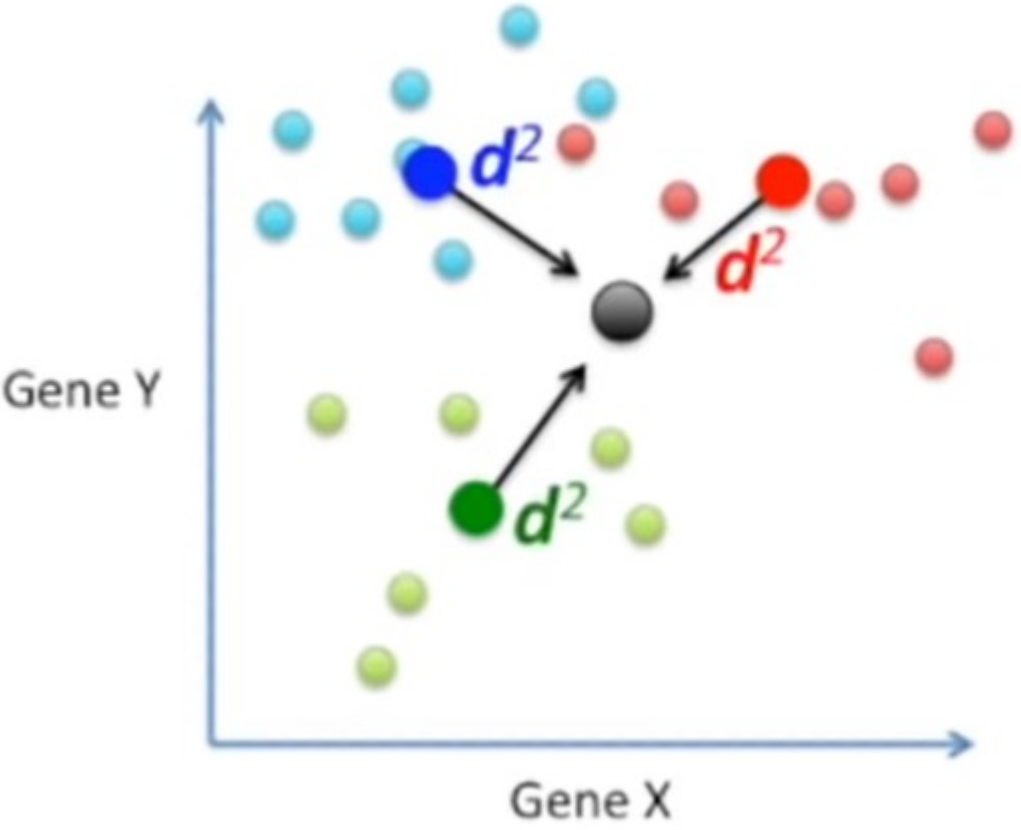
LDA



LDA 2 kategori



LDA 3 kategori



$$\frac{d^2 + d^2 + d^2}{s^2 + s^2 + s^2}$$

This is the same equation as before, but now there are terms for the **blue** category.

Now maximize the distance between each category and the central point while minimizing the scatter for each category.

Sparse Representation

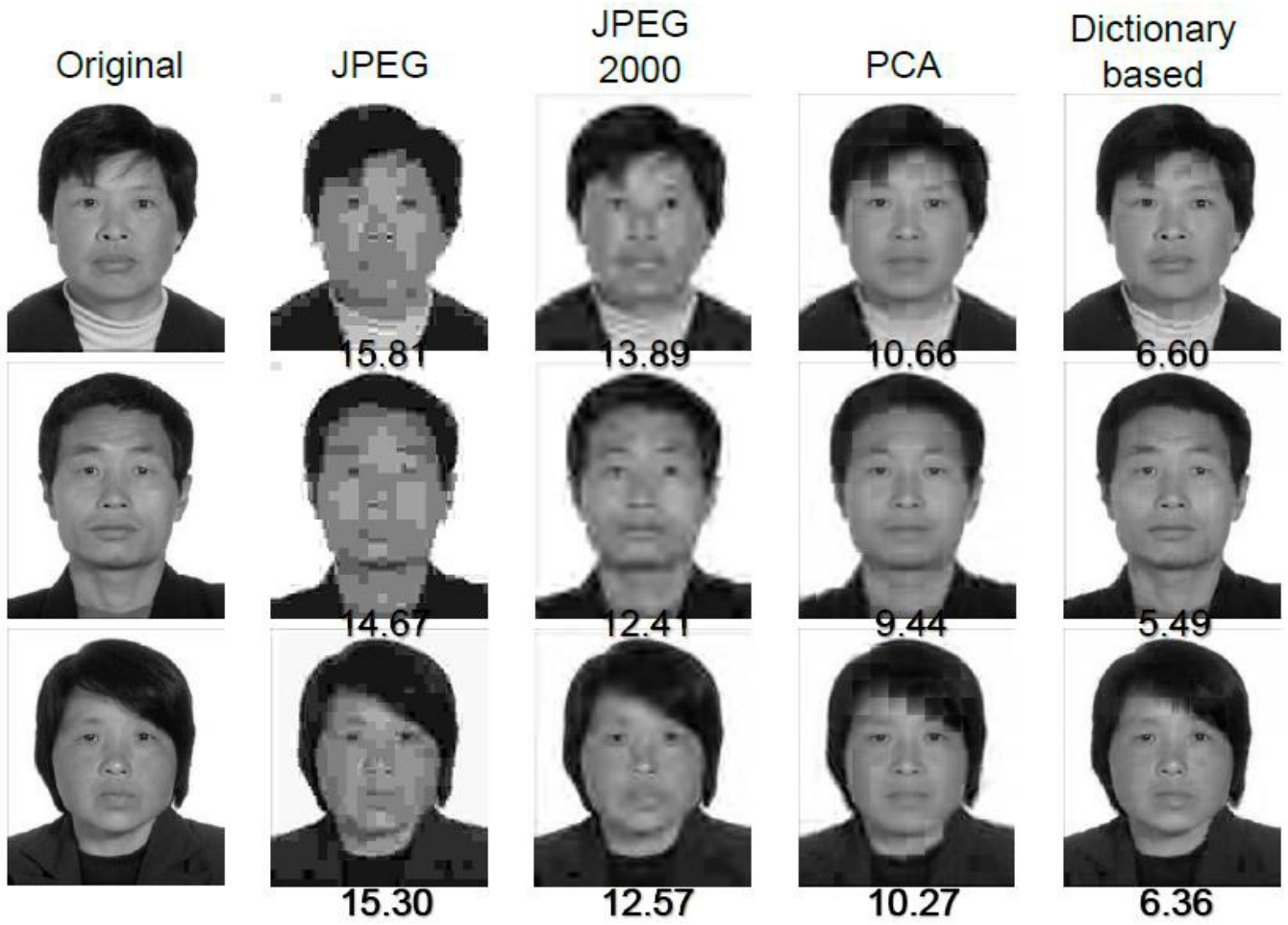
Sparsity

- Sparsity artinya banyak nilai kosong pada vektor atau matriks
- Representasi sparse

$$\begin{pmatrix} 1.0 & 0 & 5.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 3.0 & 0 & 0 & 0 & 0 & 11.0 & 0 \\ 0 & 0 & 0 & 0 & 9.0 & 0 & 0 & 0 \\ 0 & 0 & 6.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 7.0 & 0 & 0 & 0 & 0 \\ 2.0 & 0 & 0 & 0 & 0 & 10.0 & 0 & 0 \\ 0 & 0 & 0 & 8.0 & 0 & 0 & 0 & 0 \\ 0 & 4.0 & 0 & 0 & 0 & 0 & 0 & 12.0 \end{pmatrix}$$

CALVIN INSTITUTE OF TECHNOLOGY

Aplikasi: kompresi

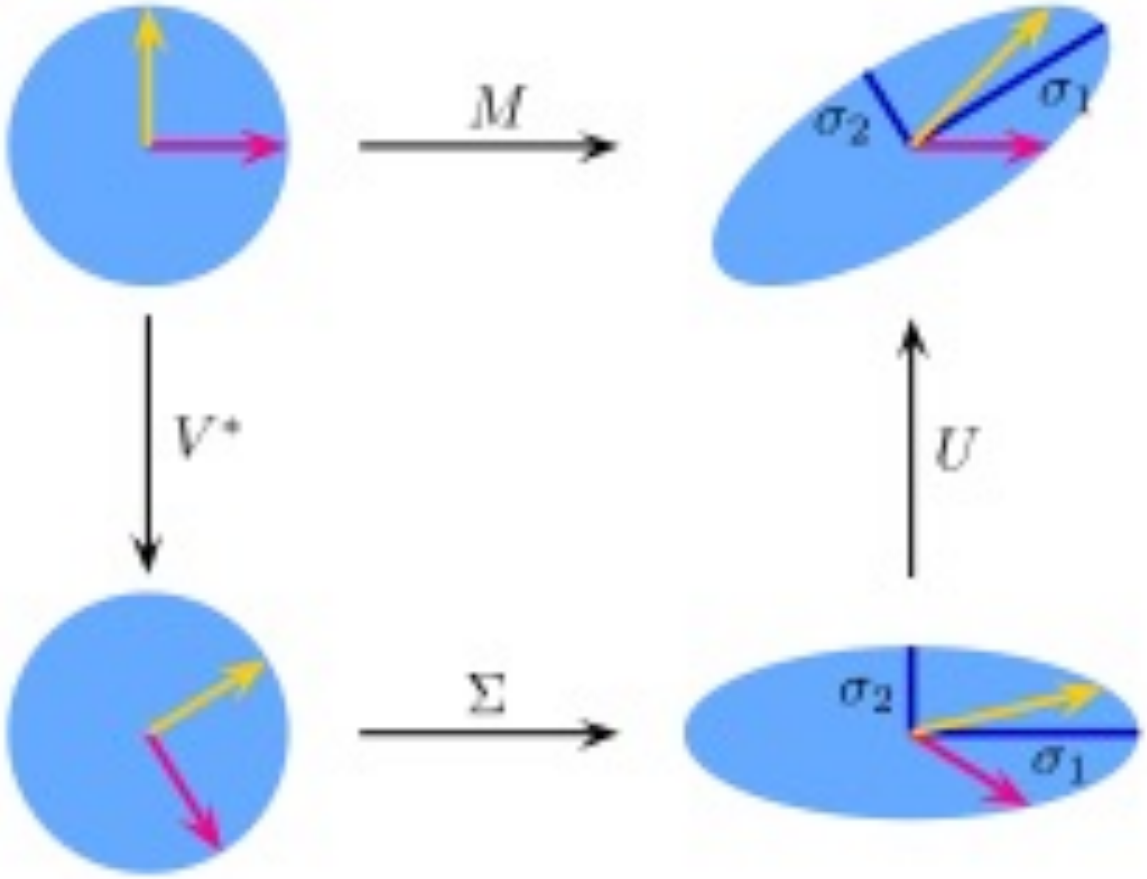


[O. Bryta, M. Elad, 2008]

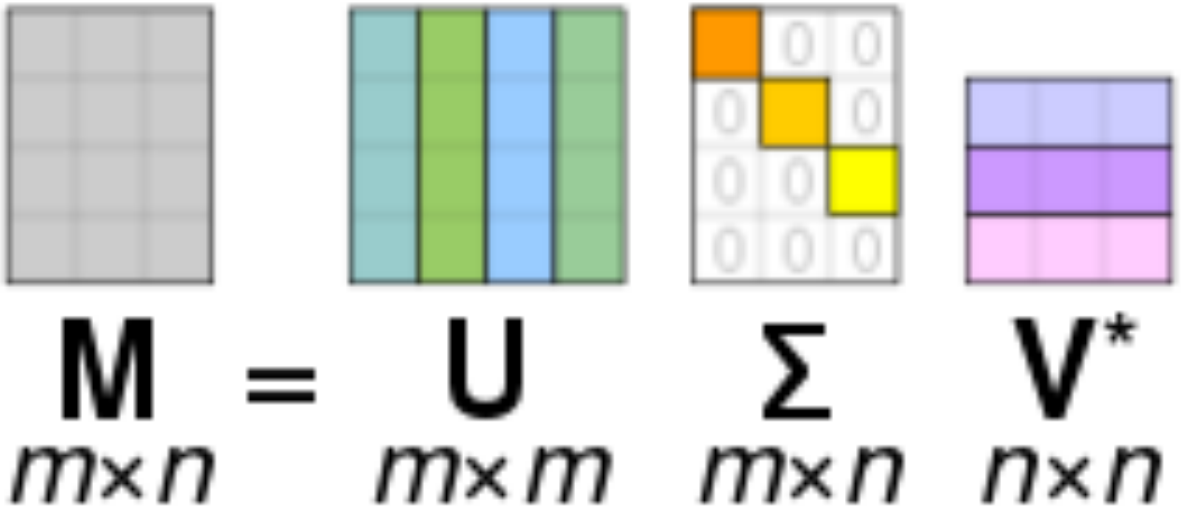
Singular Value Decomposition

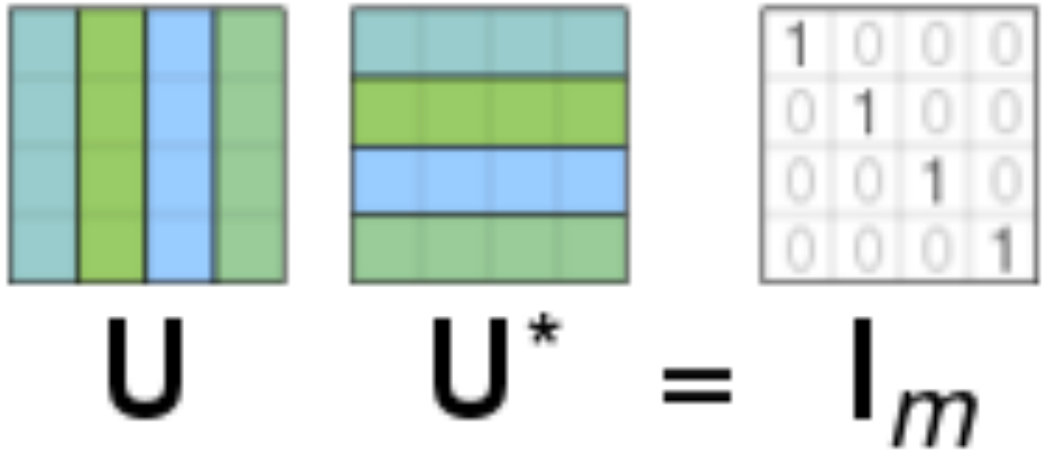
- Temukan $X = U\Sigma V^T$
- Dimana $U^T U = I_{d\times d}, V^T V = I_{n\times n}$, Σ adalah sebuah matriks diagonal
- Complexity: menghitung top k singular vectors membutuhkan $O(ndk)$
- X: matriks data
- Σ: matriks singular value
- U: matriks singular vector
- V: matriks bobot

Ilustrasi SVD



$$M = U \cdot \Sigma \cdot V^*$$





Contoh

- Relasi SVD:

$$M = U\Sigma V^*$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & \sqrt{5} & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & -\sqrt{0.8} \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{pmatrix}$$

- Relasi identitas komponen SVD

$$UU^* = \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$VV^* = \begin{pmatrix} 0 & 0 & -1 & 0 & 0 \\ \sqrt{0.2} & 0 & 0 & 0 & -\sqrt{0.8} \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ \sqrt{0.8} & 0 & 0 & 0 & \sqrt{0.2} \end{pmatrix} \begin{pmatrix} 0 & \sqrt{0.2} & 0 & 0 & \sqrt{0.8} \\ 0 & 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & -\sqrt{0.8} & 0 & 0 & \sqrt{0.2} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

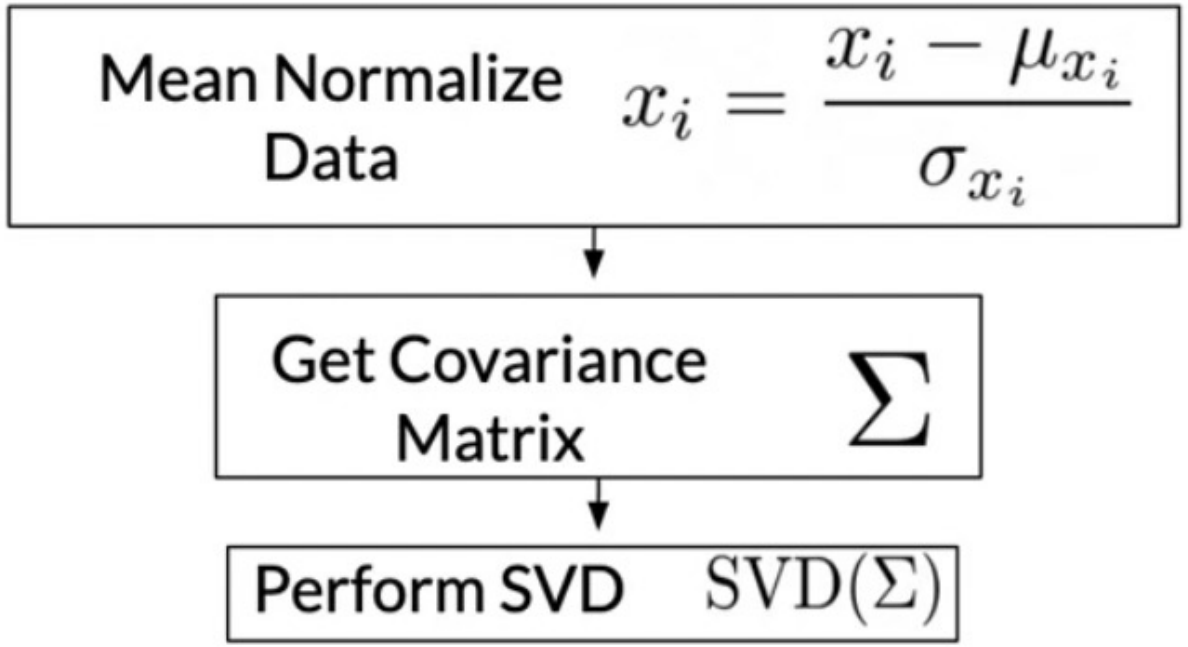
CALVIN INSTITUTE OF TECHNOLOGY

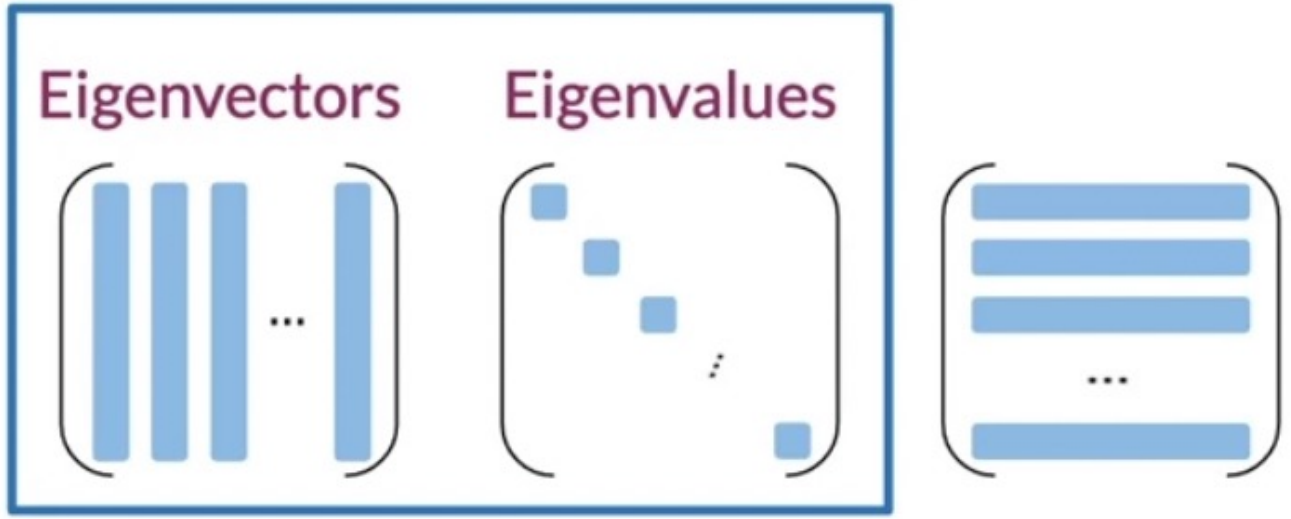
Kegunaan SVD

- SVD biasanya digunakan pada sparse data
- Sparse data adalah jenis data yang memiliki banyak nilai kosong
- Contoh sparse data adalah:
  - Recommender systems
  - Customer-product purchases
  - User-song listen counts
  - User-movie ratings
  - Text classification
  - One-hot encoding
  - Bag-of-words counts
- SVD memproyeksikan data dengan n-fitur menuju subspace dengan ruang lebih kecil namun masih menjaga esensi data asal

SVD untuk PCA

- $\tilde{X} = U\Lambda V^T$
- $U \rightarrow$ *eigenvectors*
- $\Lambda \rightarrow$ eigenvalues
- $\tilde{X}V = U\Lambda$
- $\Sigma = \frac{\tilde{X}\tilde{X}^T}{(m-1)} = \frac{U\Lambda V^T V^T \Lambda U}{(m-1)} = U\frac{\Lambda^2}{m-1}U^T$





Matrix Factorization

- Matrix factorization adalah teknik untuk mereduksi sparse matrix menjadi 2 matriks yang lebih kecil (vs SVD yang menggunakan 3 matriks)
- Salah satu metode collaborative filtering yang paling popular dan dapat digunakan sebagai alternatif embedding
- https://developers.google.com/machine-learning/recommendation/collaborative/matrix
- Matriks X dapat diestimasi dengan menggunakan 2 matriks WH

$$X = WH$$

Optimisasi

- Cost function:

$$J(H,W) = \frac{1}{2}\left|\left|X - WH\right|\right|^2$$

- Optimisasi:

$$\min_{H,W} \frac{1}{2}\left|\left|X - WH\right|\right|^2$$

- Iterasi:

$$W := W - \frac{\partial J}{\partial W}$$

$$H := H - \frac{\partial J}{\partial H}$$

CALVIN INSTITUTE OF TECHNOLOGY

✔		✔	✔	
	✔			✔
✔	✔	✔		
			✔	✔

≈

1	.1
-1	0
.2	-1
.1	1

.9	-1	1	1	-.9
-.2	-.8	-1	.9	1

×

.88	-1.08	0.9	1.09	-0.8
-0.9	1.0	-1.0	-1.0	0.9
0.38	0.6	1.2	-0.7	-1.18
-0.11	-0.9	-0.9	1.0	0.91

■	.9	-.8	1	1	-.9
▲	-.2	-.8	-1	.9	1

Harry Potter

The Triplets of Belleville

Shrek

The Dark Knight Rises

Memento

●	◆					
1	.1		✓		✓	✓	
-1	0			✓			✓
.2	-1		✓	✓	✓		
.1	1				?	✓	✓

■ arthouse <-> blockbuster

● preference for arthouse <-> blockbuster

▲ children's <-> adult's

◆ preference for children's <-> adult's

CALVIN INSTITUTE OF TECHNOL

Shrek

+1

Blockbuster

Harry Potter

The Dark Knight Rises

-1

Children

+1

Adult

The Triplets of Belleville

-1

Arthouse

Memento

Non-Negative Matrix Factorization

- NMF adalah salah satu pendekatan matrix factorization yang paling umum digunakan
- Memaksa semua nilai matriks positif
- Nilai positif akan memastikan nilai kecil berefek kecil, nilai besar berefek besar (tidak terjadi interaksi negatif)
- https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.NMF.html

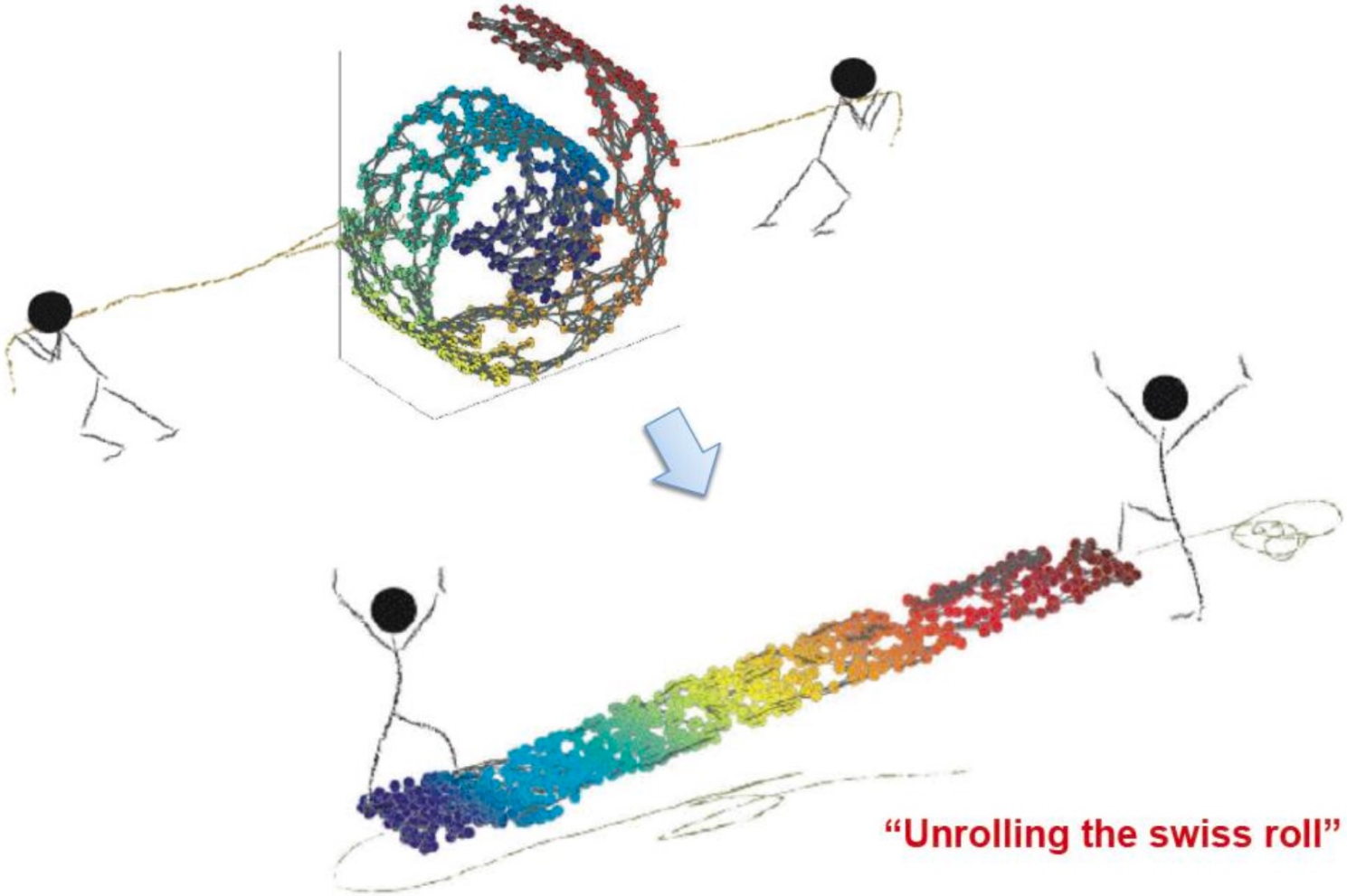
Uji Pemahaman

- User A memiliki vector MF [1, 0.1] dan buku X memiliki vector MF [0.9, -0.2], berapakah kemungkinan user A tertarik pada buku X?

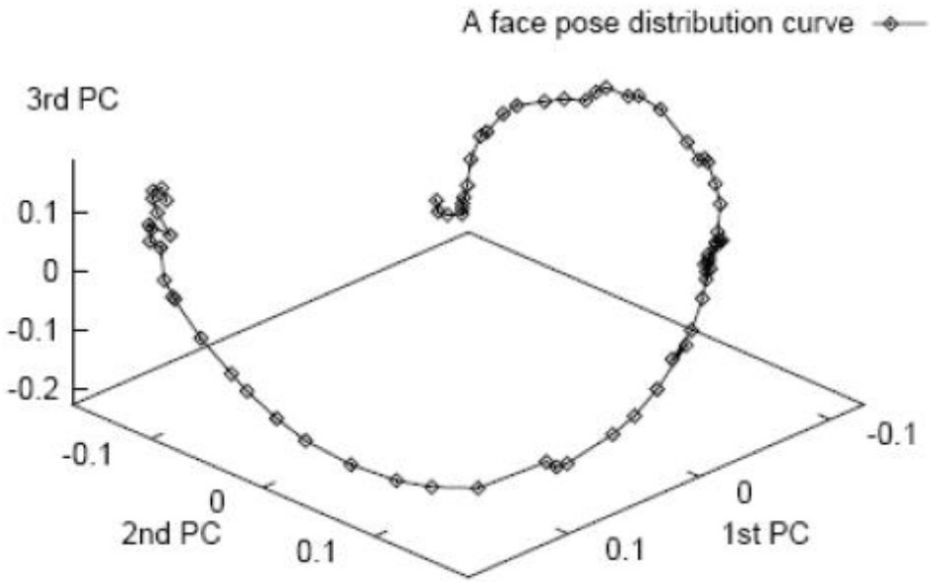
Reduksi Dimensi Non-Linear

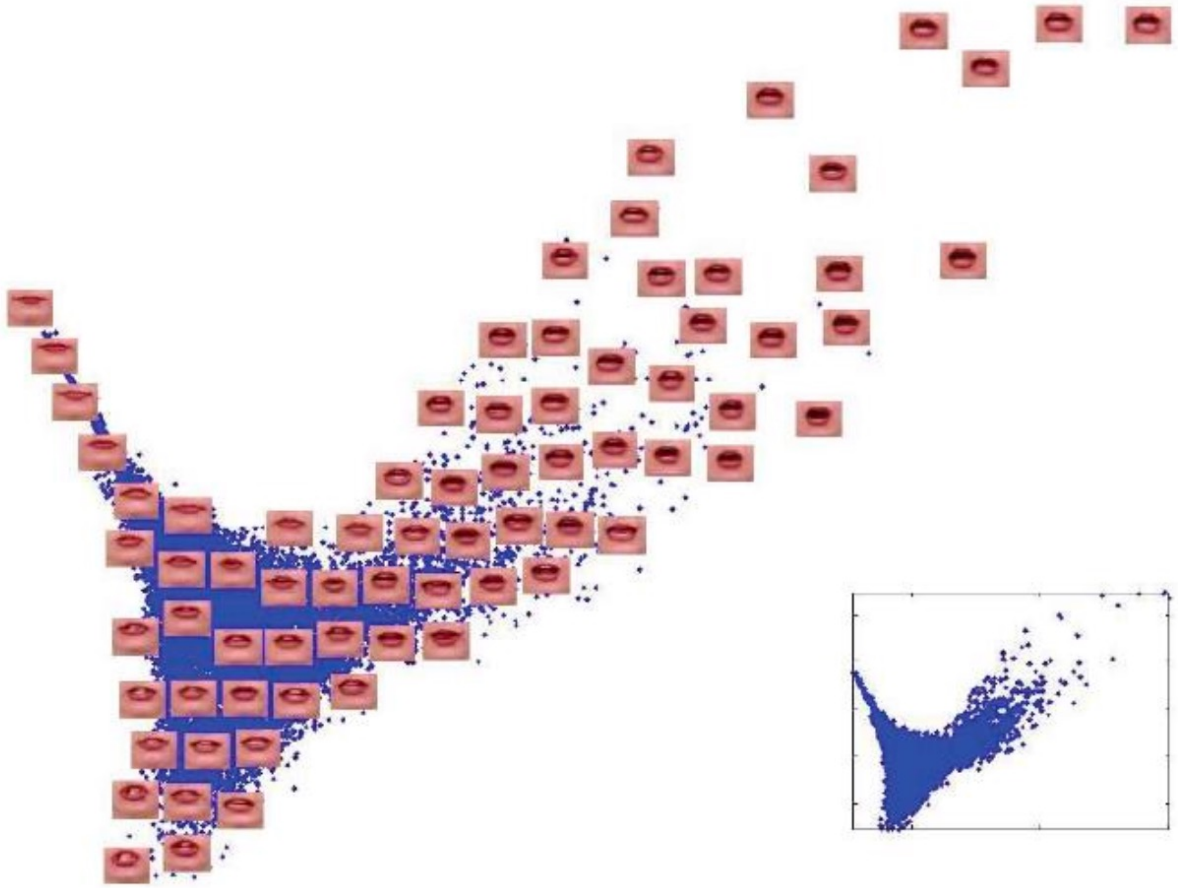
Nonlinear Dimension Reduction

- Data pada umumnya tersembunyi dekat nonlinear manifold



Beberapa contoh nonlinear manifold

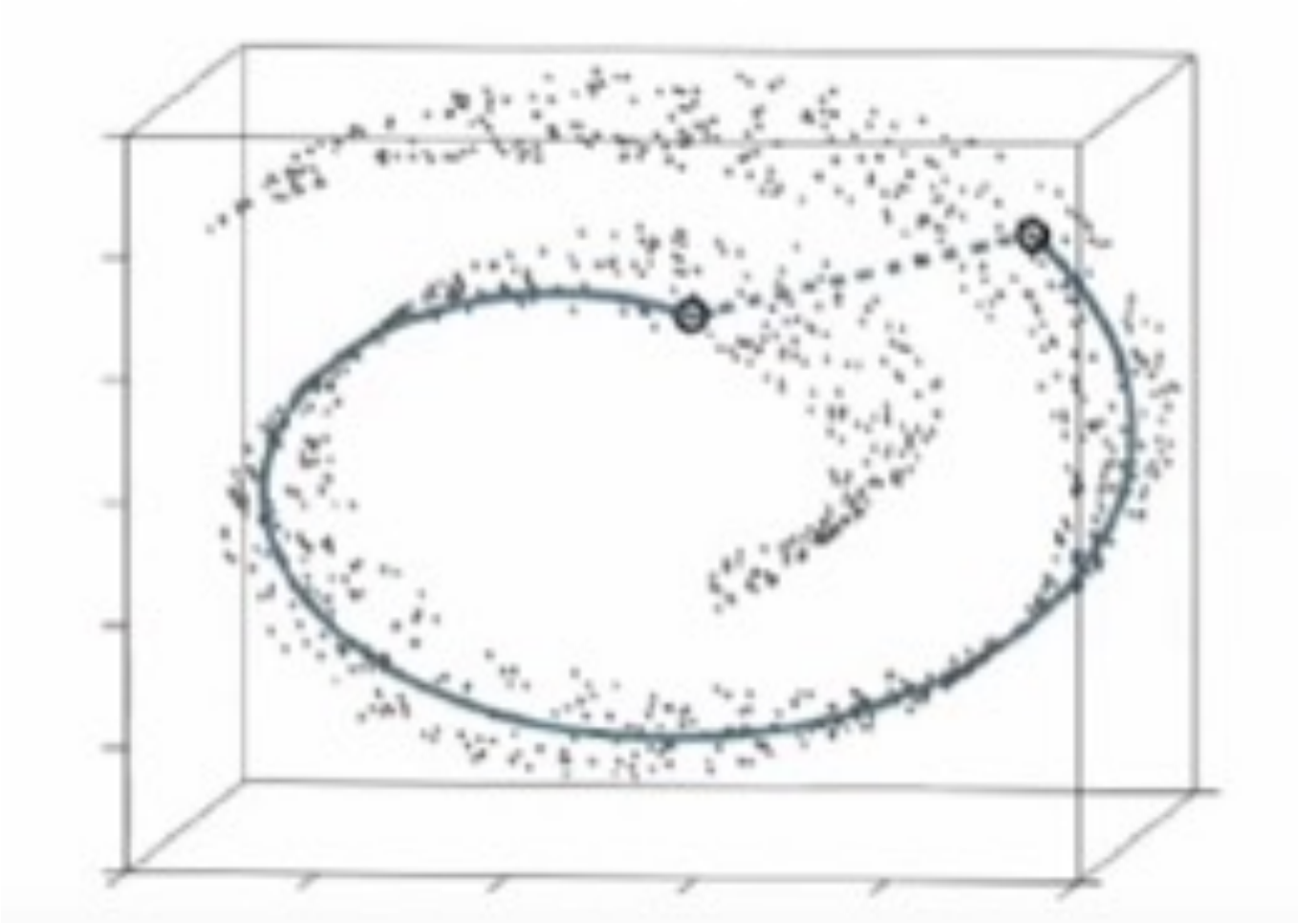




Manifold Learning

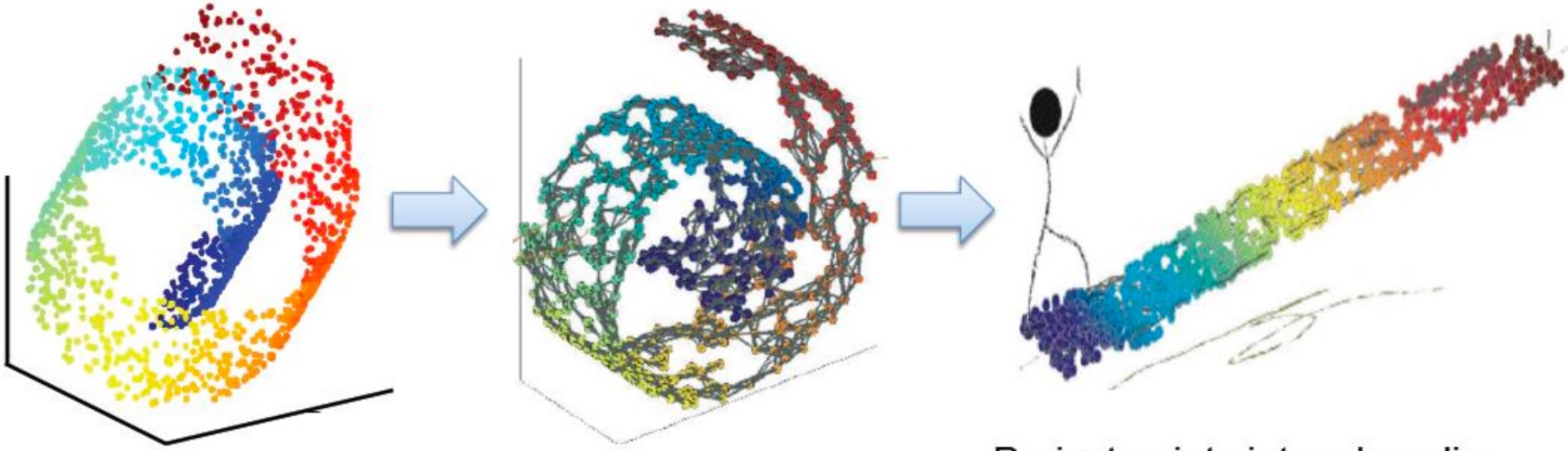
- Teknik ini digunakan untuk membuat proyeksi *low-dimensional* dari data *high-dimensional*
- Biasanya dilakukan untuk keperluan visualisasi
- Proyeksi didesain untuk membuat representasi data *low-dimensional* dengan dengan mempertahankan struktur data asli
- Ada beberapa metode *manifold learning*: Kohonen Self-Organizing Map (SOM), Sammons Mapping, Multidimensional Scaling (MDS), dan t-distributed Stochastic Neighbor Embedding (t-SNE)
- Penjelasan lebih detail mengenai t-SNE: https://towardsdatascience.com/an-introduction-to-t-sne-with-python-example-5a3a293108d1
- Video penjelasan intuisi t-SNE: https://www.youtube.com/watch?v=NEaUSP4YerM

Manifold Learning



Manifold Learning

- Laplacian eigenmap: Local information preservation
  - Konstruksi Adjacency Matrix
  - Pilih bobot
  - Hitung eigenmaps
- Kernel PCA
- Isomap
- Embedding

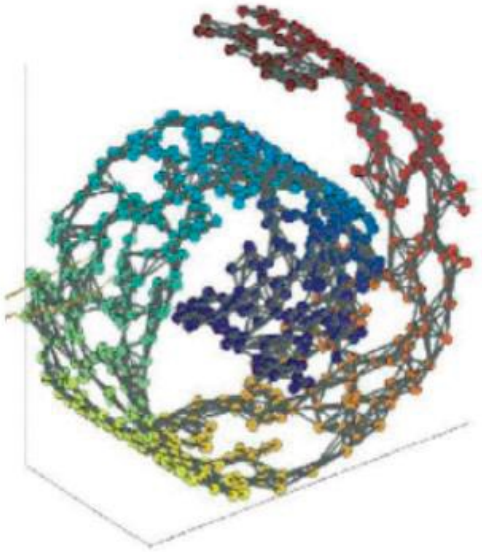


Construct graph from data points (capture local information)

Project points into a low-dim space using “eigenvectors of the graph”

CALVIN INSTITUTE OF TECHNOLOGY

Laplacian EigenMap



- Step 1: konstruksi matriks adjacency
  - Terdapat link antara i dan j jika $\left|\left|x^{(i)} - x^{(j)}\right|\right| \leq \varepsilon$
- Step 2: pilih bobot
  - Tanpa bobot, $W_{ij} = 1$
  - Gaussian kernel function $W_{ij} = e^{-\frac{\left|\left|x^{(i)} - x^{(j)}\right|\right|^2}{2\sigma^2}}$
- Step 3: hitung eigenmaps
  - Temukan f dimana jika $x^{(i)}$ dekat dengan $x^{(j)}$ pada adjacency graph, maka proyeksinya dekat di embedding space
  - Optimisasi yang dibutuhkan adalah:

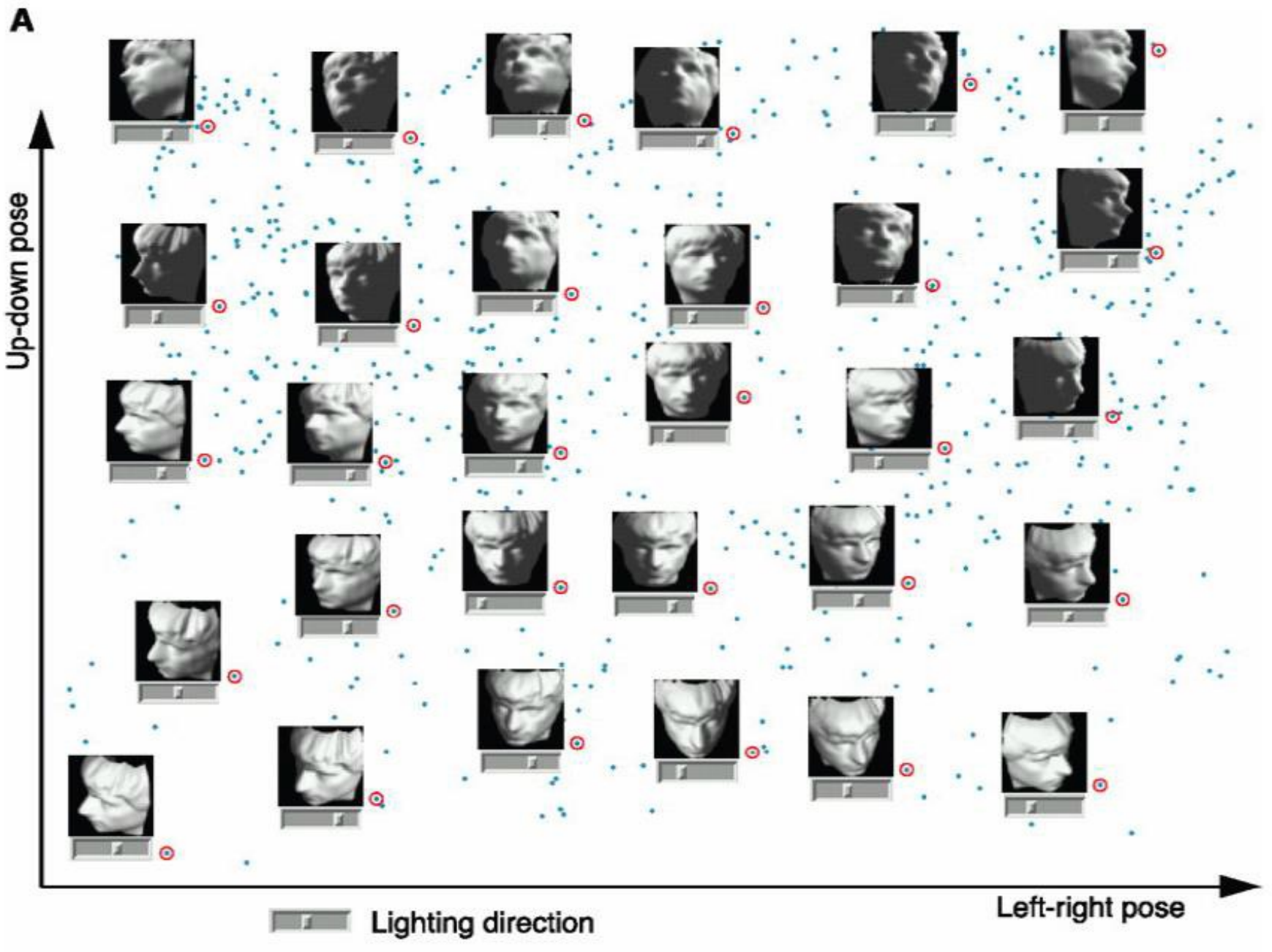
$$\min \sum_{ij} W_{ij}\left(f^{(i)} - f^{(j)}\right)^2$$

  - Solusi ini ekuivalen dengan menemukan eigenvectors dari graph Laplacian $L = D - W$

$$Lf = \lambda f$$

CALVIN INSTITUTE OF TECHNOLOGY

Contoh



PCA vs Laplacian Eigenmaps

PCA	Laplacian Eigenmaps
linear	nonlinear
Menggunakan eigenvector terbesar dari matriks kovarian/korelasi	Menggunakan eigenvector terkecil dari matriks Laplacian $n \times n$ antar titik data
Eigenvectors menghasilkan ruang fitur laten: Embedding suatu data diperoleh dengan memproyeksikan fitur menuju latent subspace $Z = U^T x$	Eigenvectors dapat menghasilkan embedding dari titik data secara langsung: $x^{(i)} \rightarrow (f_1(i), \ldots, f_d(i))$

Kernel PCA

- Rekonstruksi PCA sehingga bisa memproses data nonlinear
- Matriks kovarian pada PCA

$$\Sigma = \frac{1}{n}\tilde{X}\tilde{X}^T$$

- Matriks kovarian pada Kernel PCA

$$\Sigma = \frac{1}{n}\phi(\tilde{X})\phi(\tilde{X})^T$$

- Dimana $\phi(x)$ adalah sebuah kernel atau fungsi pemetaan nonlinear
- Konsep kernel akan dibahas lebih lanjut di bab SVM
- Prosedur selanjutnya mirip denga PCA biasa

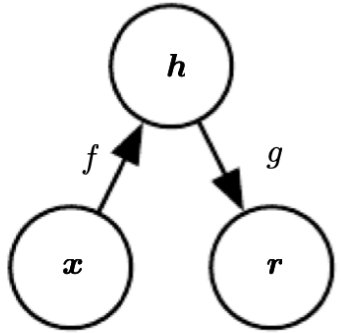
Autoencoder

Representasi yang lebih dalam

- Deep representation learning
  - Terinspirasi dari arsitektur hierarki dari neural network
  - Atsitekur ini belajar melalui menumpuk beberapa lapisan blok pembelajaran
- Contoh:
  - Autoencoder
  - RBM

AutoEncoder

- Model yang dilatih untuk menyalin input menuju output
- Tujuan: mengungkap fitur utama penyebab variasi dalam dataset
- Cara: mempelajari fitur yang dapat digunakan untuk rekonstruksi gambar
- Terdiri dari 2 bagian:
  - Encoder: $\boldsymbol{h} = f(\boldsymbol{x})$
  - Decoder: $\boldsymbol{r} = g(\boldsymbol{h})$

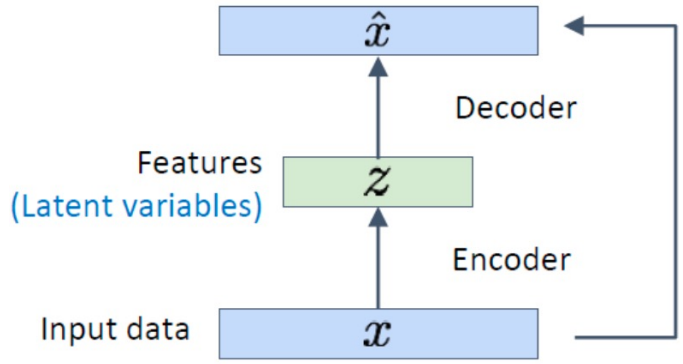


Deep learning, p503

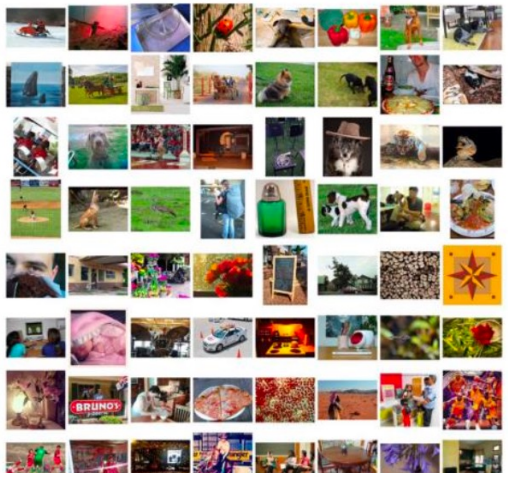
Figure 14.1: The general structure of an autoencoder, mapping an input $\boldsymbol{x}$ to an output (called reconstruction) $\boldsymbol{r}$ through an internal representation or code $\boldsymbol{h}$. The autoencoder has two components: the encoder f (mapping $\boldsymbol{x}$ to $\boldsymbol{h}$) and the decoder g (mapping $\boldsymbol{h}$ to $\boldsymbol{r}$).

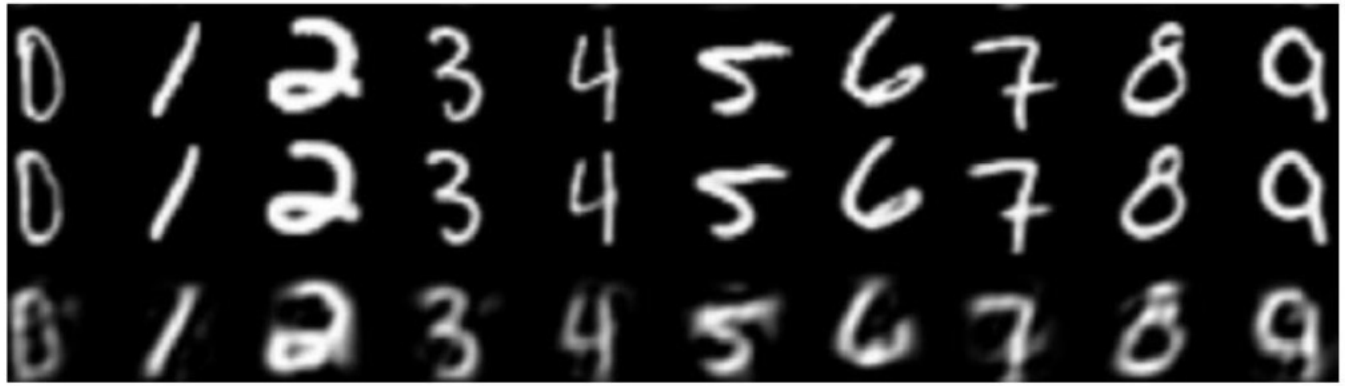
AutoEncoder

- Transformasi linear menghasilkan hasil yang dekat dengan PCA
- Deep Neural networks menghasilkan rekonstruksi non-linear yang lebih baik



L2 Loss function:
$\|x - \hat{x}\|^2$

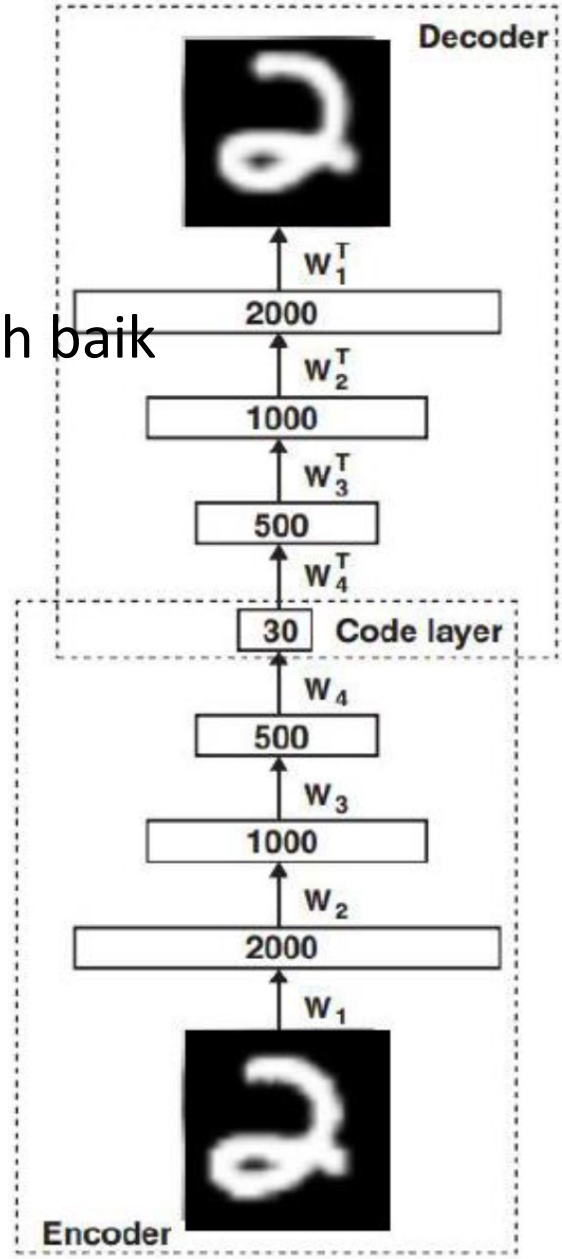




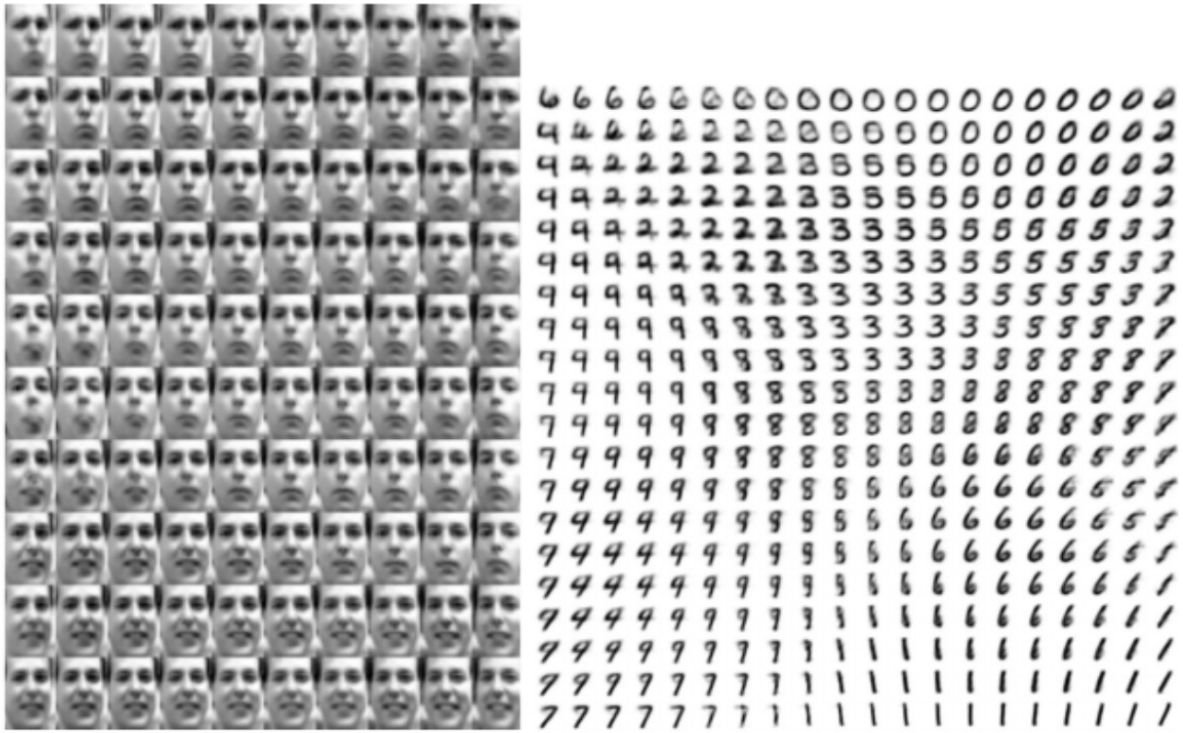
Original

Autoencoder

PCA



Visualisasi Autoencoder

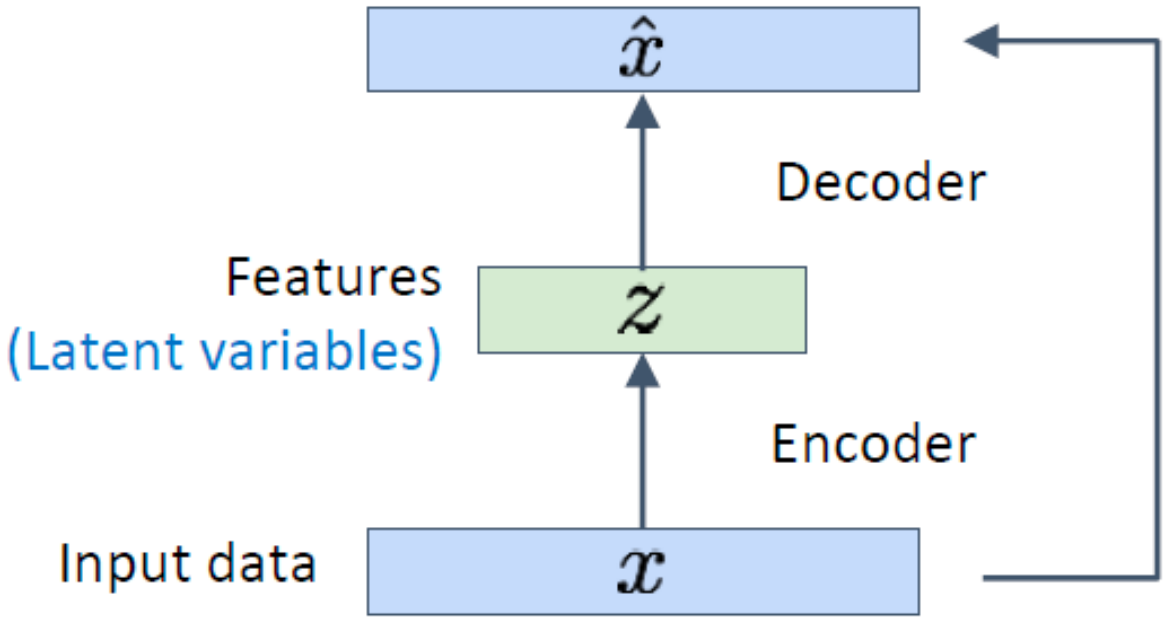


Deep Learning, p700

Figure 20.6: Examples of two-dimensional coordinate systems for high-dimensional manifolds, learned by a variational autoencoder (Kingma and Welling, 2014a). Two dimensions may be plotted directly on the page for visualization, so we can gain an understanding of how the model works by training a model with a 2-D latent code, even if we believe the intrinsic dimensionality of the data manifold is much higher. The images shown are not examples from the training set but images $\boldsymbol{x}$ actually generated by the model $p(\boldsymbol{x} \mid \boldsymbol{z})$, simply by changing the 2-D "code" $\boldsymbol{z}$ (each image corresponds to a different choice of "code" $\boldsymbol{z}$ on a 2-D uniform grid). *(Left)*The two-dimensional map of the Frey faces manifold. One dimension that has been discovered (horizontal) mostly corresponds to a rotation of the face, while the other (vertical) corresponds to the emotional expression. *(Right)*The two-dimensional map of the MNIST manifold.

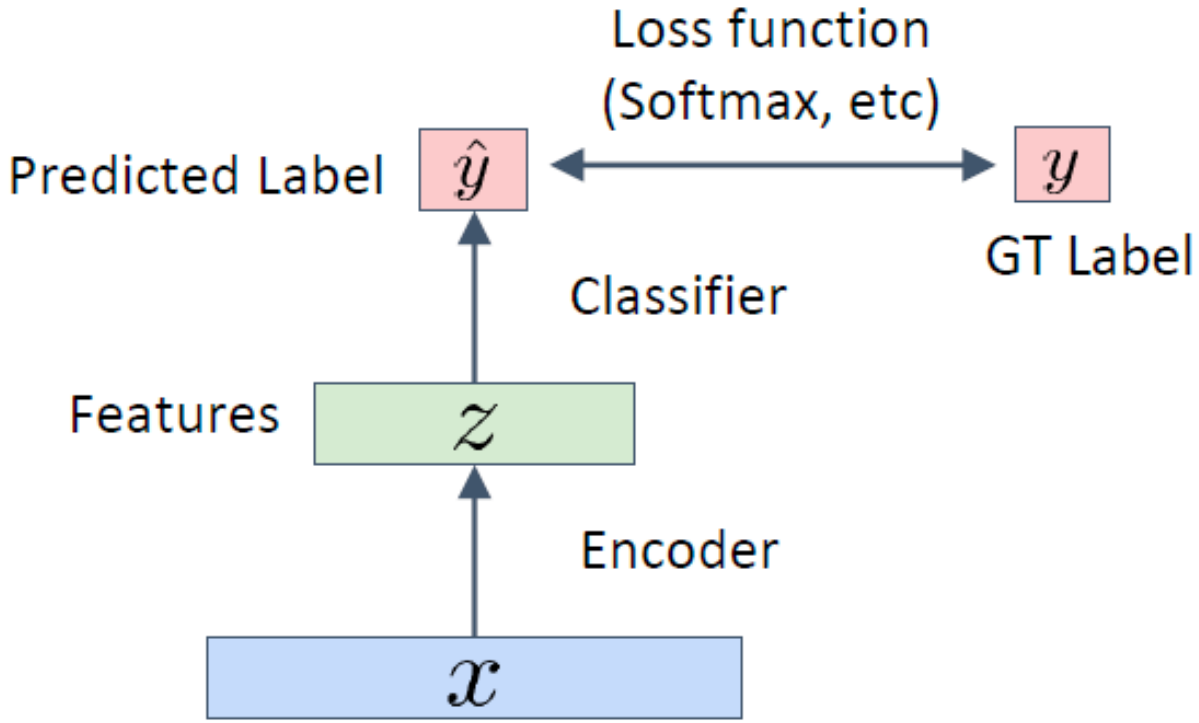
Aplikasi: Klasifikasi Semi-Supervised

- Banyak gambar, tapi sedikit label
- Dimulai secara unsupervised
- Latih autoencoder pada banyak gambar



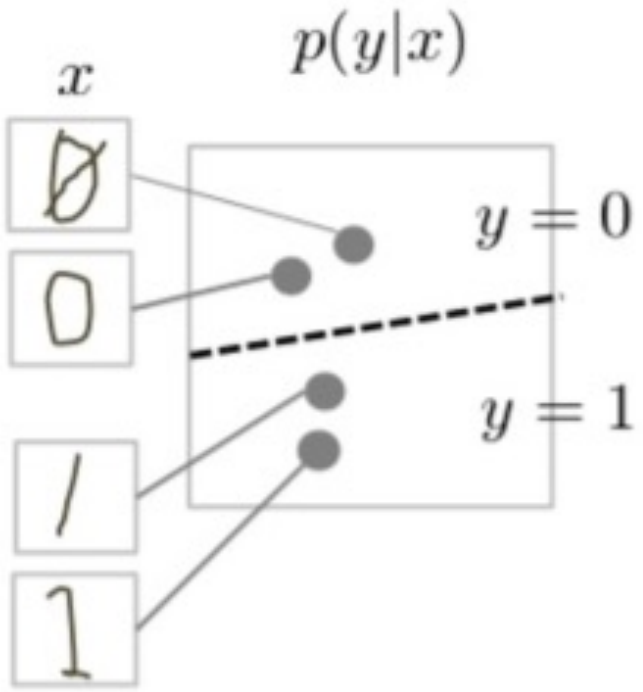
finetune secara supervised

latih klasifikasi pada data berlabel

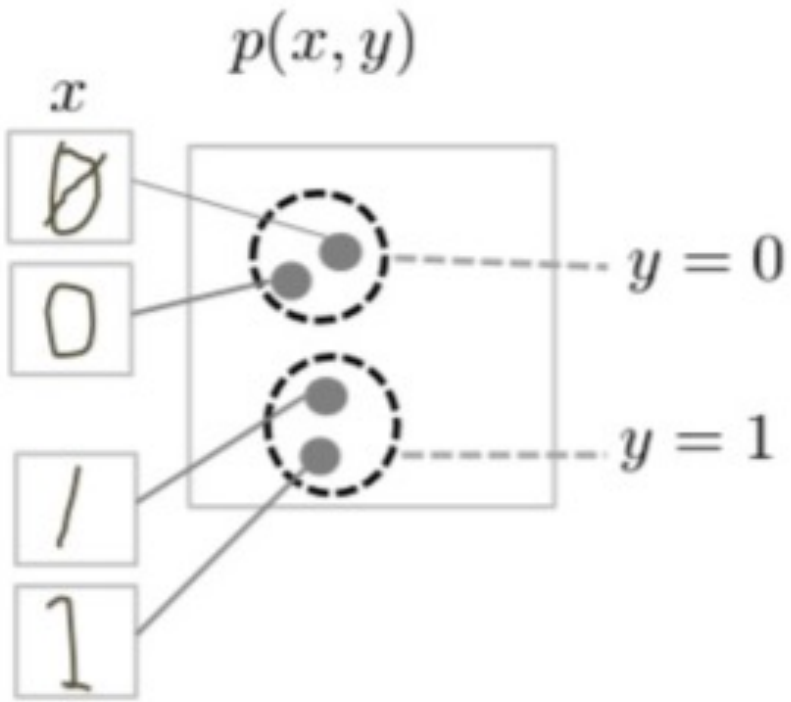


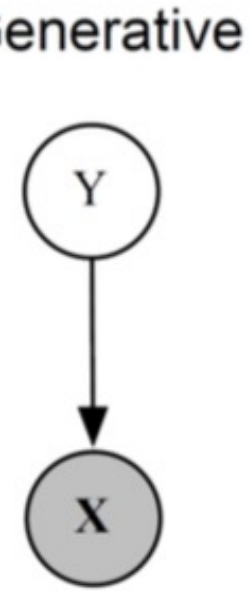
Model Generatif

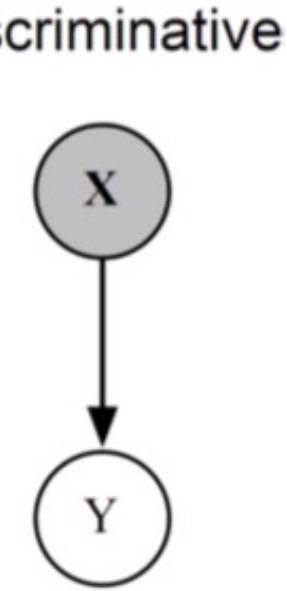
- Discriminative Model



- Generative Model

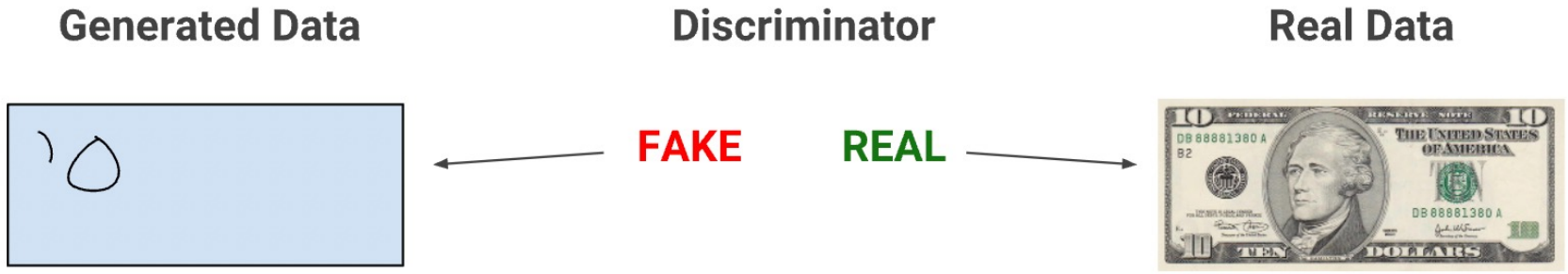






Proses Belajar

When training begins, the generator produces obviously fake data, and the discriminator quickly learns to tell that it's fake:



As training progresses, the generator gets closer to producing output that can fool the discriminator:

Finally, if generator training goes well, the discriminator gets worse at telling the difference between real and fake. It starts to classify fake data as real, and its accuracy decreases.

Random Face Generator

- https://this-person-does-not-exist.com/en

Embedding

Latent Space

- Latent space (dikenal dengan istilah embedding) menyimpan proyeksi informasi tertentu dari data
- Vector space ini dapat digunakan untuk melakukan interpolasi dan aritmatika
- Karena bersifat continuous, maka operasi yang dilakukan pada latent space berdampak secara continuous pada data

Integer Representation

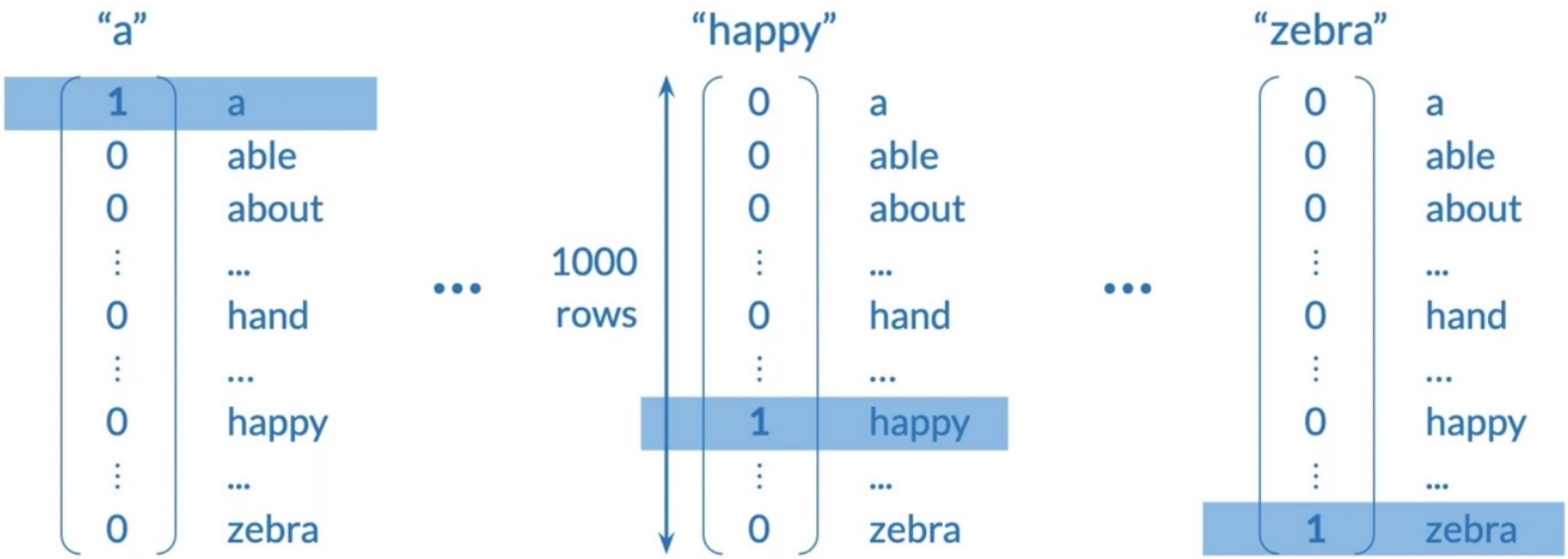
- Keuntungan & Kekurangan

\+ Simple

\- Ordering: little semantic sense

Word	Number
a	1
able	2
about	3
...	...
hand	615
...	...
happy	621
...	...
zebra	1000

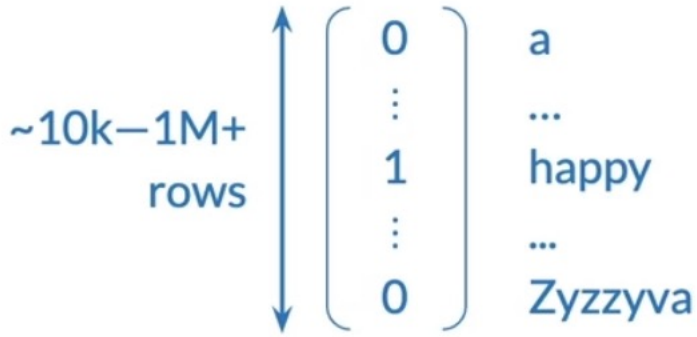
One-hot vectors

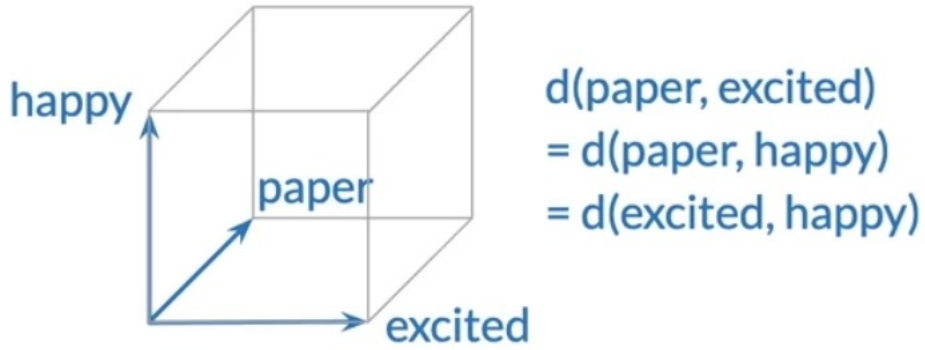


Keuntungan dan kekurangan one-hot vectors

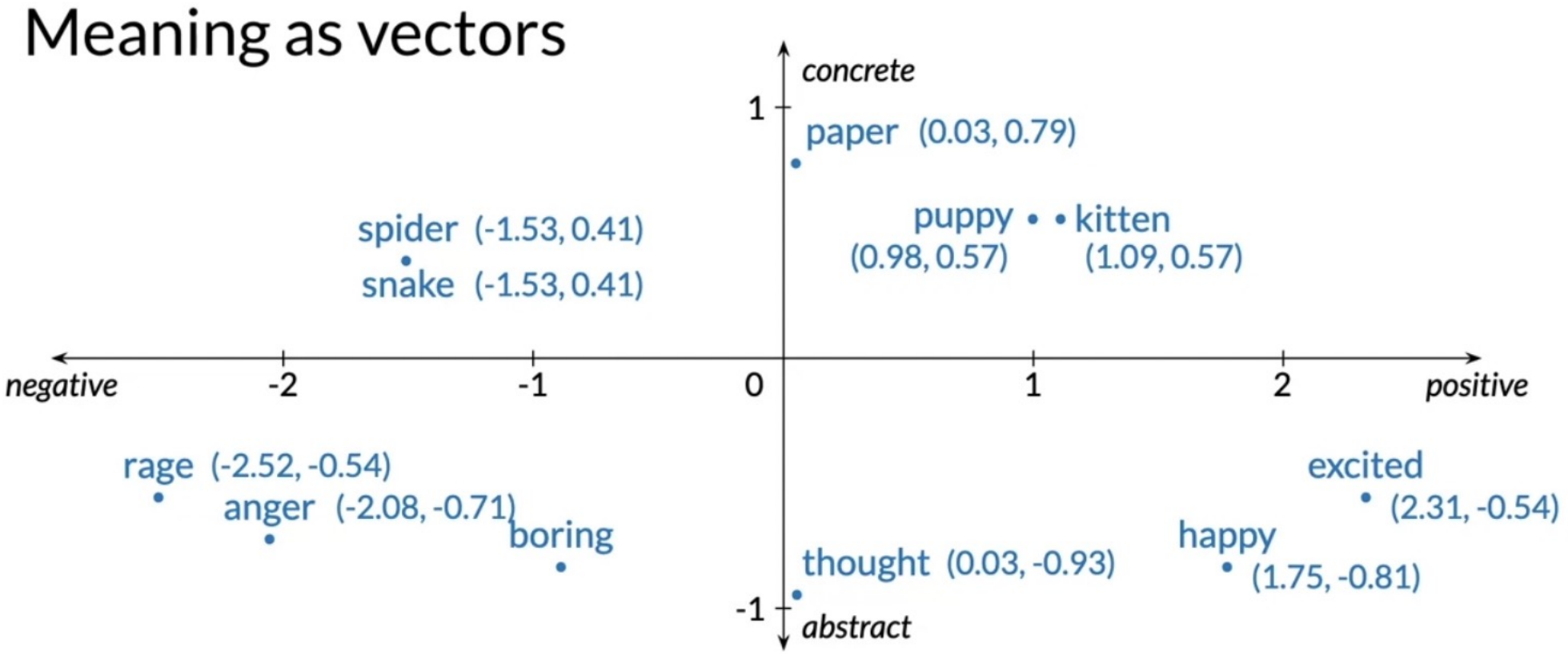
One-hot vectors

+ Simple
+ No implied ordering
- Huge vectors
- No embedded meaning





Representasi Vektor



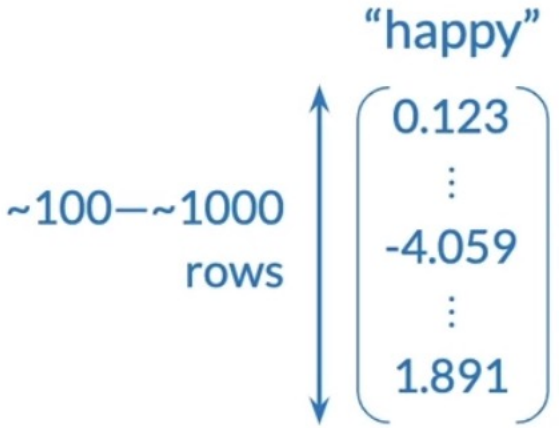
Keuntungan embedding vectors

Word embedding vectors

- \+ Low dimension
- \+ Embed meaning
  - e.g. semantic distance

    forest ≈ tree forest ≉ ticket
  - e.g. analogies

    Paris:France :: Rome:?



Uji Pemahaman

- Representasi manakah yang paling dekat dengan konsep word embedding?

- A

  car -> 2

  caravan -> 3

- B

  car -> (0.1 1)

  caravan -> (-0.1 0.9)

- C

  car -> (0 1 0 0)

  caravan -> (0 0 1 0)

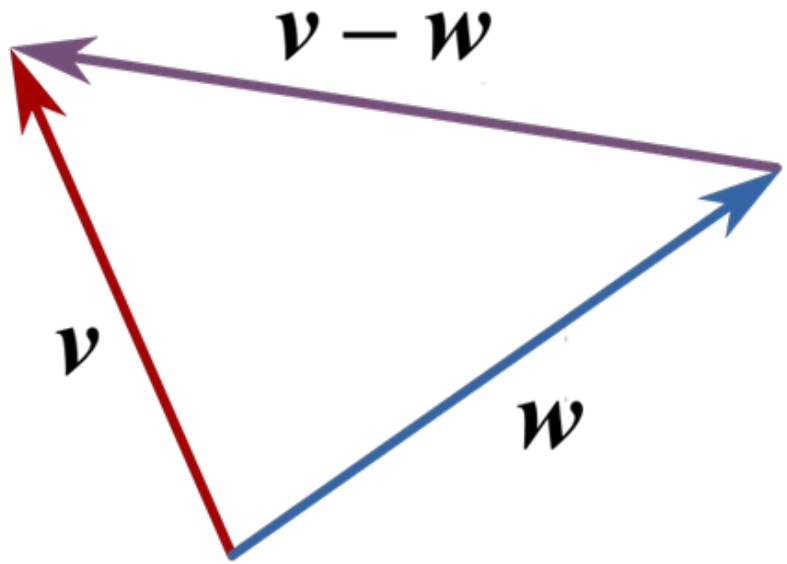
- D

  car -> (1 0.1)

  caravan -> (-1 -0.9)

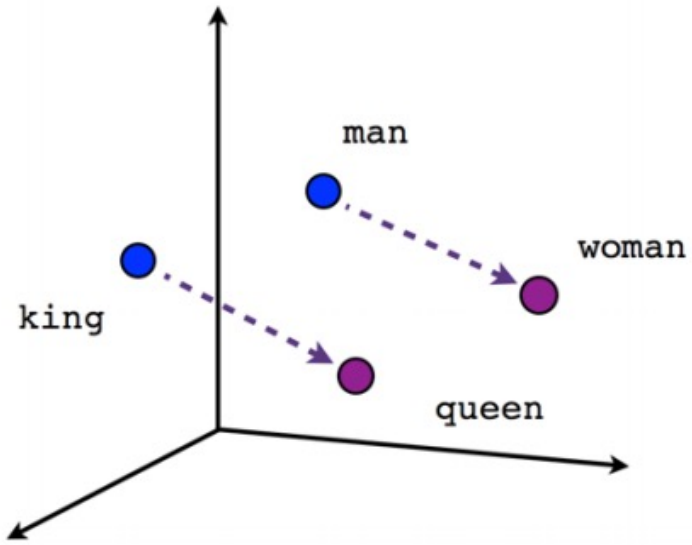
CALVIN INSTITUTE OF TECHNOLOGY

Operasi Aritmatika pada vektor

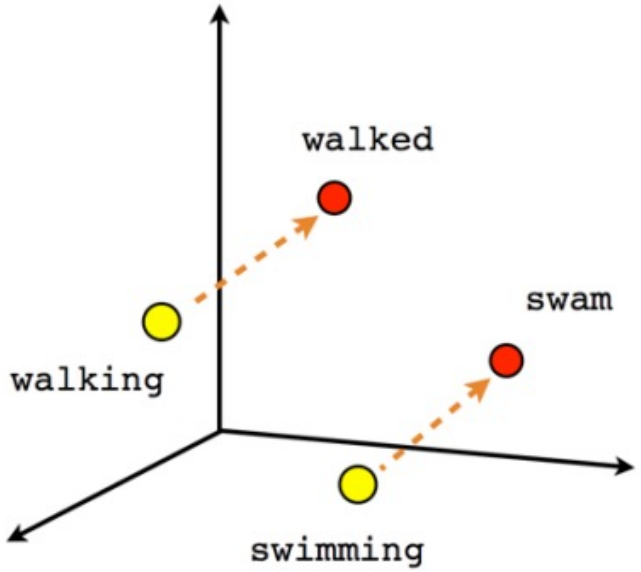


- $\vec{v} + \vec{w} = \vec{w} + \vec{v}$
- $\vec{v} + \vec{0} = \vec{v}$
- $a(\vec{v} + \vec{w}) = a\vec{v} + a\vec{w}$
- $\vec{u} + (\vec{v} + \vec{w}) = (\vec{u} + \vec{v}) + \vec{w}$
- $1\vec{v} = \vec{v}$
- $(a + b)\vec{v} = a\vec{v} + b\vec{v}$

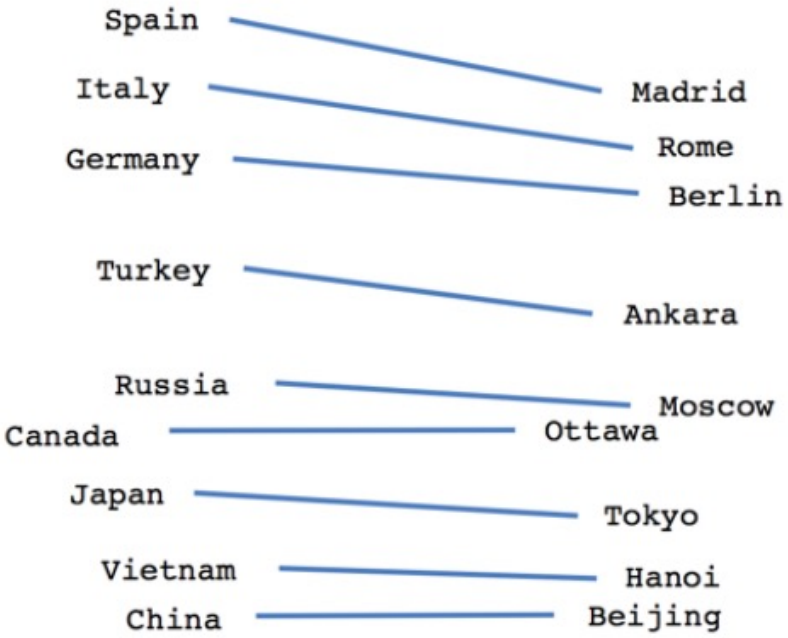
Contoh operasi aritmatik pada word embedding



Male-Female



Verb tense



Country-Capital

Contoh operasi aritmatik pada image embedding

- https://arxiv.org/pdf/1511.06434.pdf

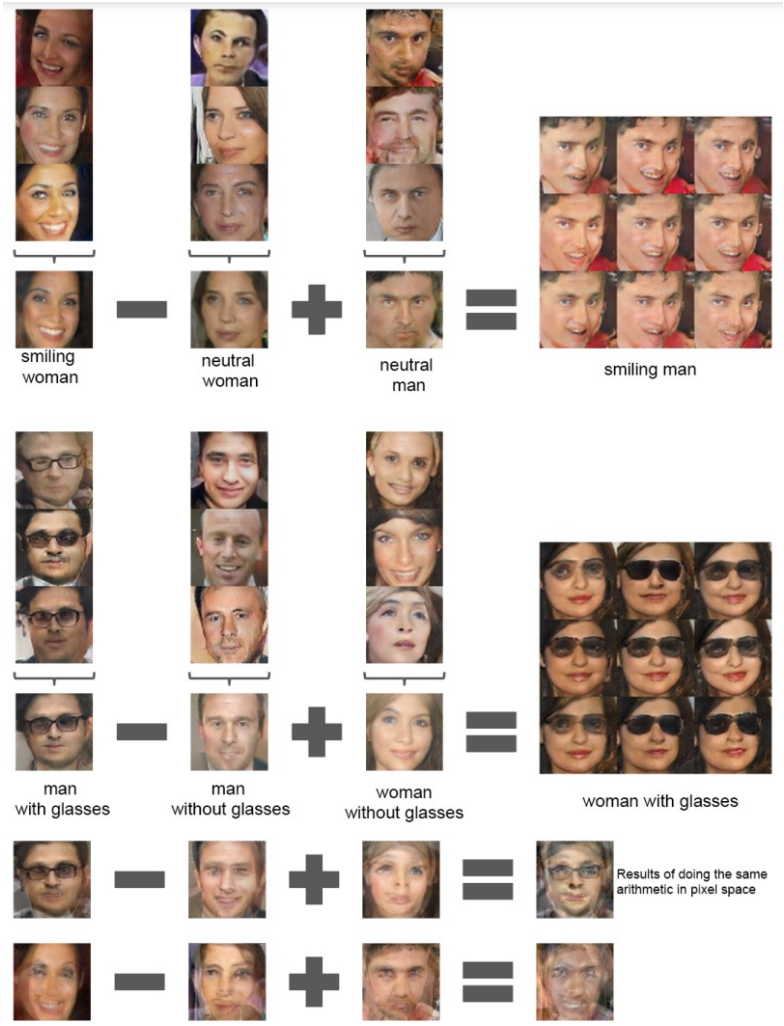


Figure 7: Vector arithmetic for visual concepts. For each column, the Z vectors of samples are averaged. Arithmetic was then performed on the mean vectors creating a new vector Y. The center sample on the right hand side is produce by feeding Y as input to the generator. To demonstrate the interpolation capabilities of the generator, uniform noise sampled with scale +-0.25 was added to Y to produce the 8 other samples. Applying arithmetic in the input space (bottom two examples) results in noisy overlap due to misalignment.

CALVIN INSTITUTE OF TECHNOLOGY

First Order Motion

Tuhan Memberkati