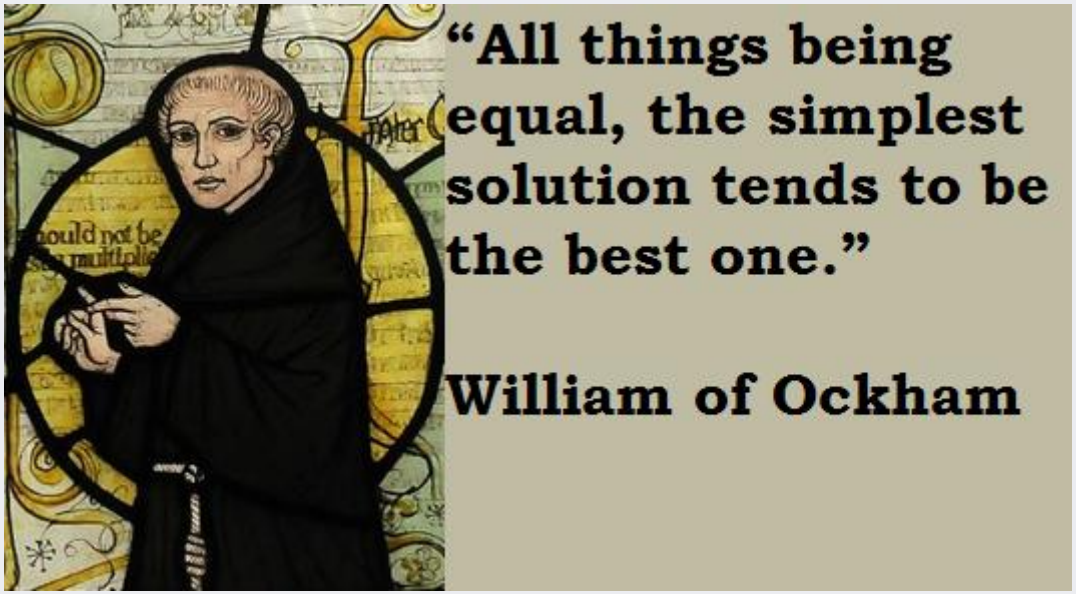


Lecture 1: Dimensionality Reduction

Pilsung Kang
School of Industrial Management Engineering
Korea University

AGENDA

01 Dimensionality Reduction: Overview

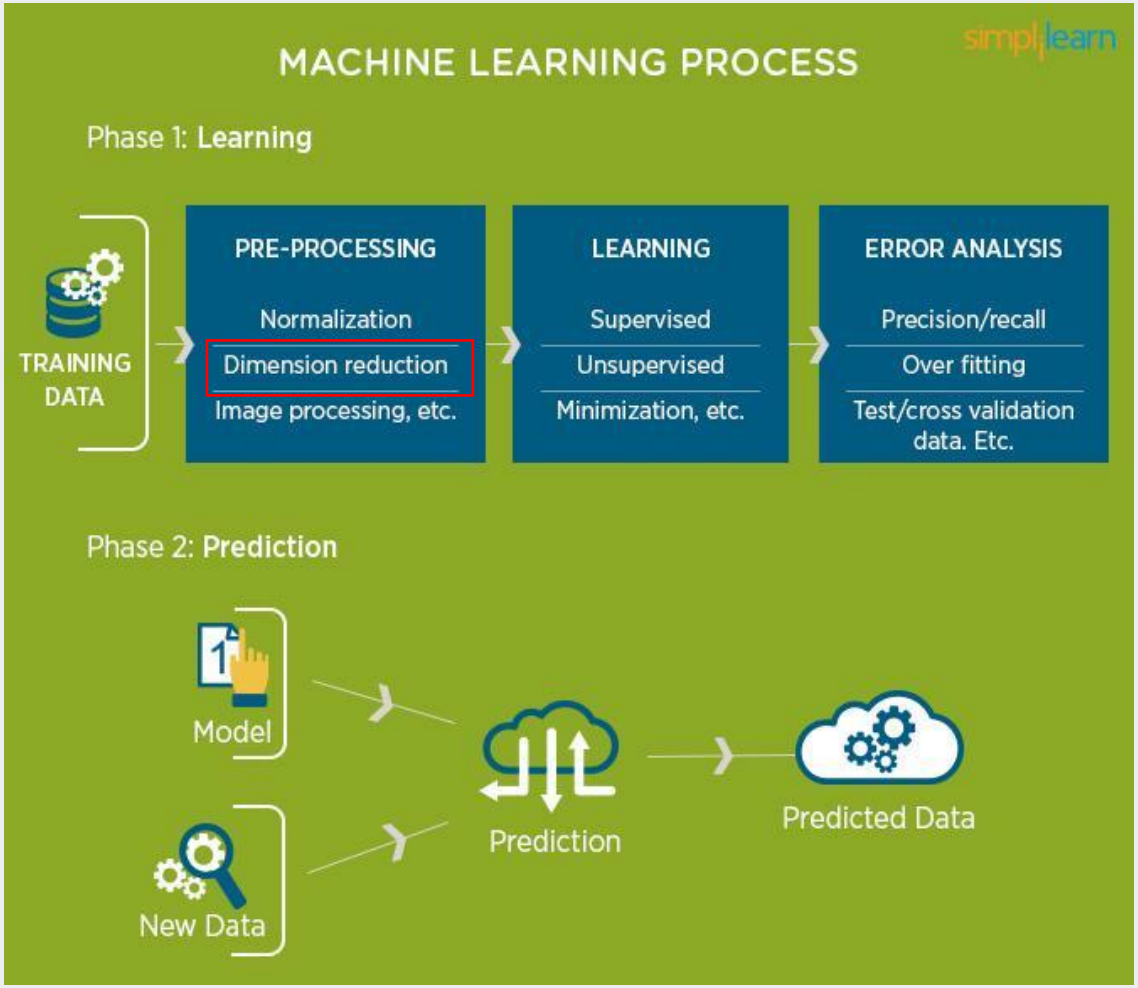
02 Supervised Methods

03 Unsupervised Methods: Linear Mapping

04 Unsupervised Methods: Non-linear Mapping

Data Analytics Process

- Process of Business Analytics with Machine Learning



https://www.simplilearn.com/what-is-machine-learning-and-why-it-matters-article

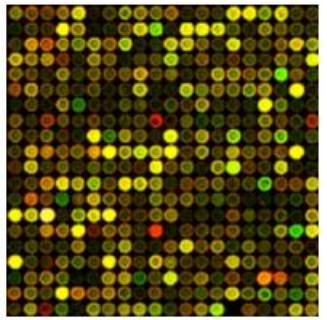
High-dimensional Data

- Examples of high dimensional data

Document classification:
Billions of documents x Thousands/
Millions of words/bigrams matrix

Recommendation systems:
480,189 users x 17,770 movies matrix

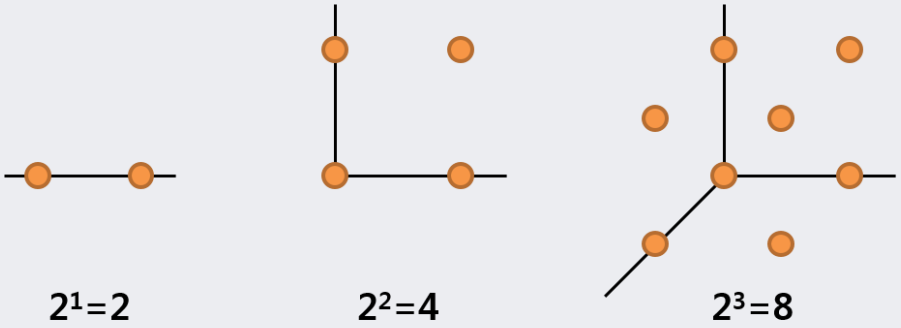
Clustering gene expression profiles:
10,000 genes x 1,000 conditions

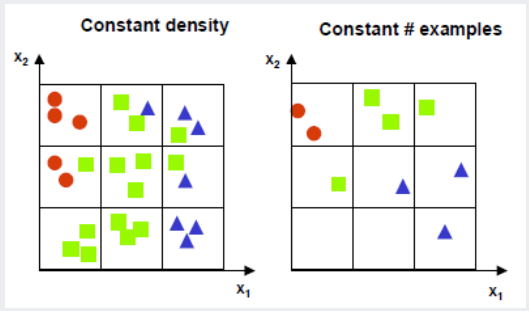


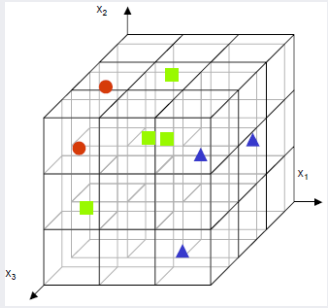
Dimensionality Reduction: Overview

- Curse of dimensionality
  - ✓ The number of instances increases exponentially to achieve the same explanation ability when the number of variables increases

"If there are various logical ways to explain a certain phenomenon, the simplest is the best" - Occam's Razor

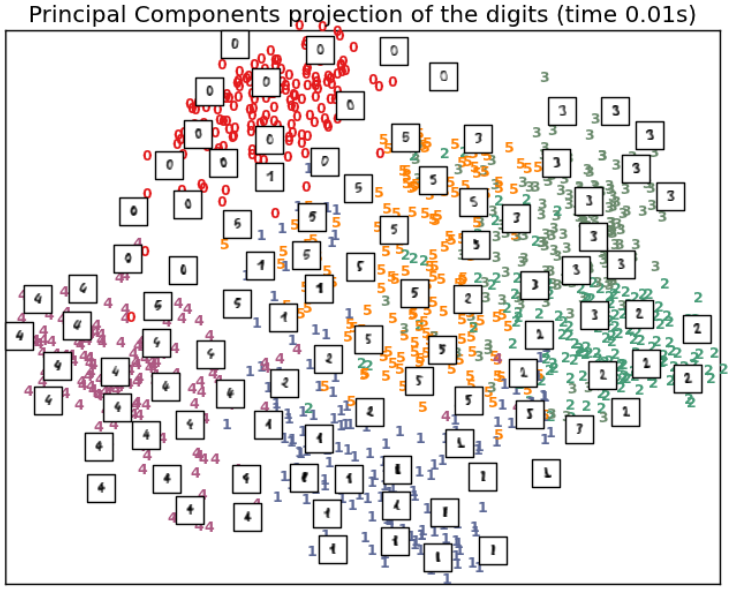


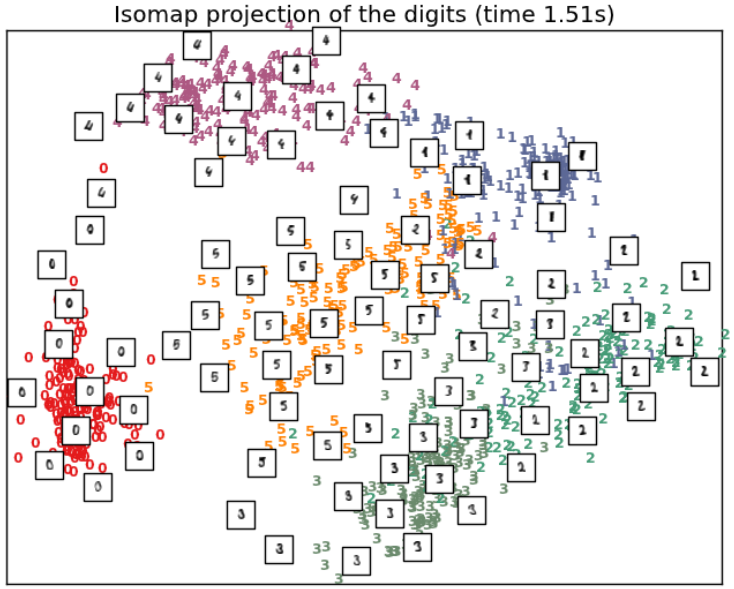




Dimensionality Reduction: Overview

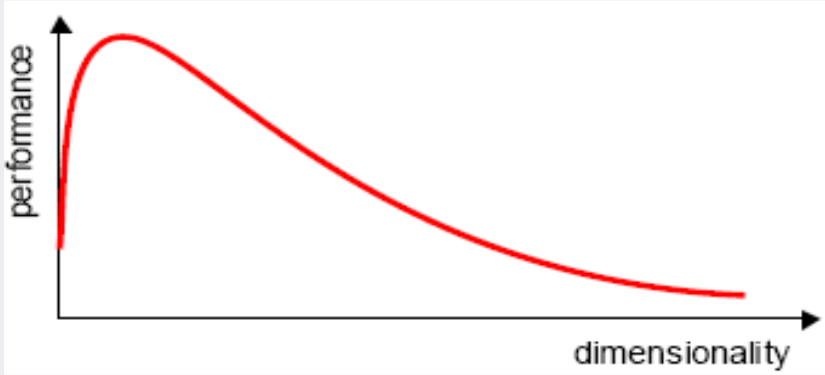
- Curse of dimensionality
  - ✓ Sometimes, an intrinsic dimension is relatively low compared to the original dimension.
    - Ex: handwritten digits in a 16 by 16 pixel (256 dimensions)
    - Reduced to two dimensions by PCA and ISOMAP





Dimensionality Reduction: Overview

- Curse of dimensionality
  - ✓ Problems caused by high-dimensionality
    - Increase the probability of having noise in data → degenerate the prediction performance
    - Increase computational burden for training/applying prediction models
    - Require more number of examples to secure generalization ability of prediction model
  - ✓ To resolve the curse of dimensionality
    - Utilize domain knowledge
    - Use a regularization term in objective function
    - Employ a quantitative reduction technique

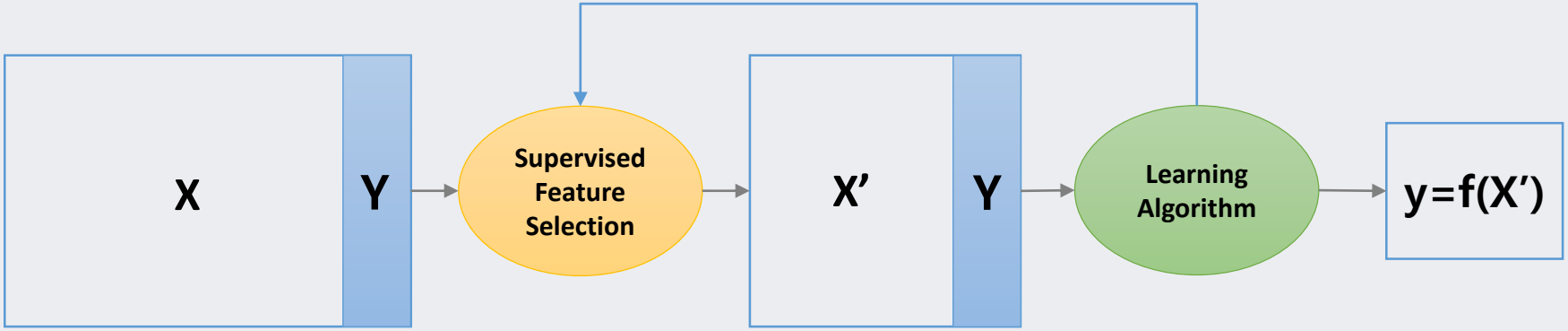


Dimensionality Reduction: Overview

- Backgrounds
  - ✓ Theoretically, model performance improves when the number of variables increases (Under variable independence condition)
  - ✓ In reality, model performance degenerates due to variable dependence, existence of noise, etc.
- Purpose
  - ✓ Identify a subset of variables that best fit the model
- Effect
  - ✓ Remove correlations between variables
  - ✓ Simplified post-processing
  - ✓ Remove redundant or unnecessary variables while keeping relevant information
  - ✓ Visualization can be possible

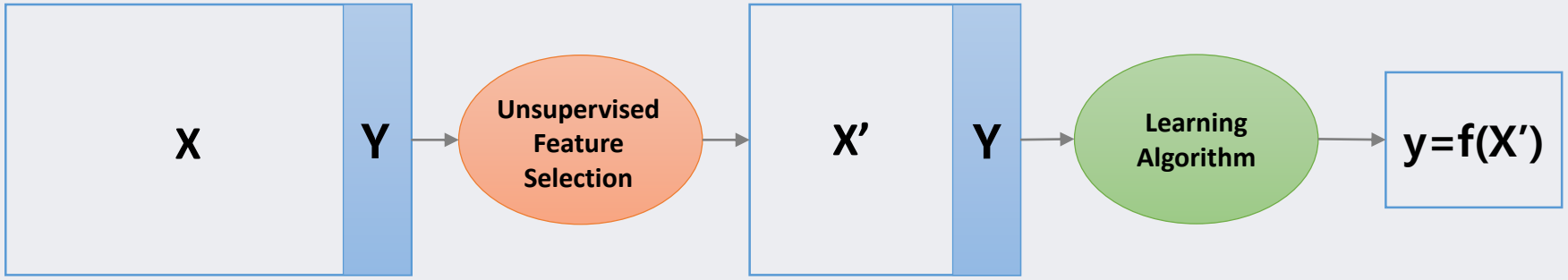
Dimensionality Reduction: Overview

- Supervised vs. Unsupervised Dimensionality Reduction
  - ✓ Supervised dimensionality reduction
    - Use data mining models to verify the reduced dimensions
    - Dimensionality reduction results can be different according to the data mining algorithms employed



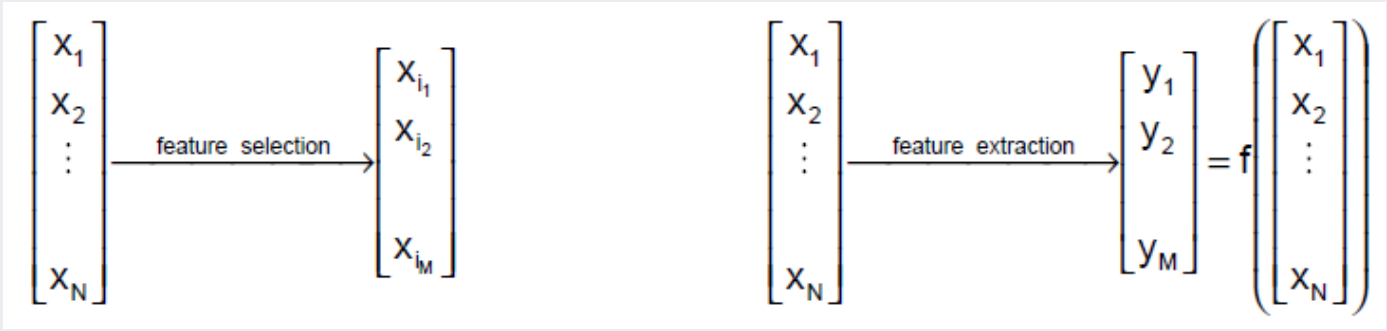
Dimensionality Reduction: Overview

- Supervised vs. Unsupervised Dimensionality Reduction
  - ✓ Unsupervised dimensionality reduction
    - Find a set of coordinate systems in a lower dimension that preserve the information (e.g., variance, distance, etc.) in the original input space as much as possible
    - Do not use data mining models during the process
    - Dimensionality reduction results are identical if the data and method is same



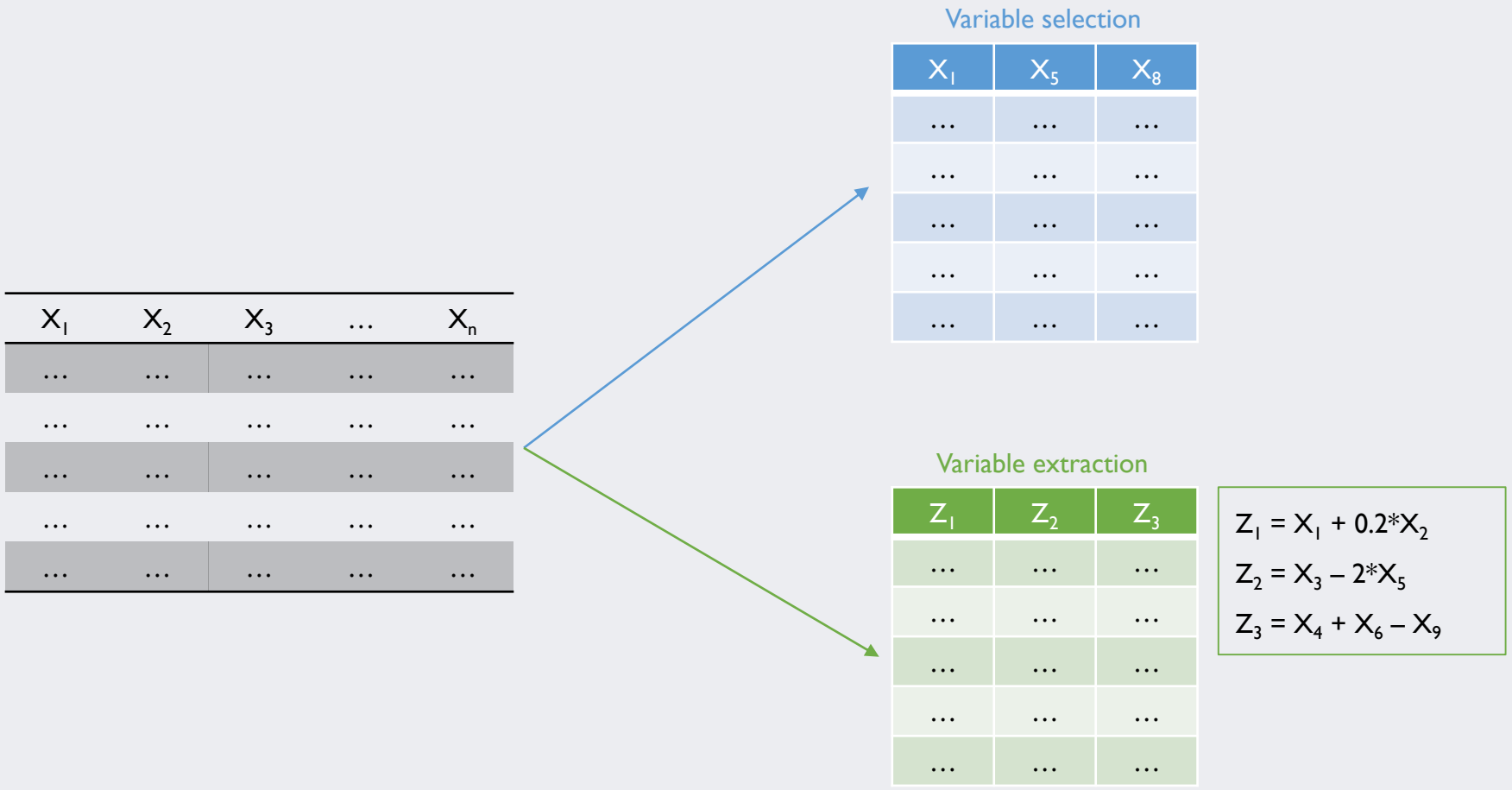
Dimensionality Reduction: Overview

- Dimensionality reduction techniques
  - ✓ Variable/feature selection
    - Select a subset of variables from the original variable set
    - Filter – Variable selection and model training are independent
    - Wrapper – Variable selection is done to optimizes the result of the considered data mining model
  - ✓ Variable/feature extraction
    - Extract a new smaller set of variables that preserve the characteristics of the original data
    - Performance metric that is independent from data mining models is used



Dimensionality Reduction: Overview

- Selection vs. Extraction
  - ✓ Conceptual difference between variable selection and variable extraction



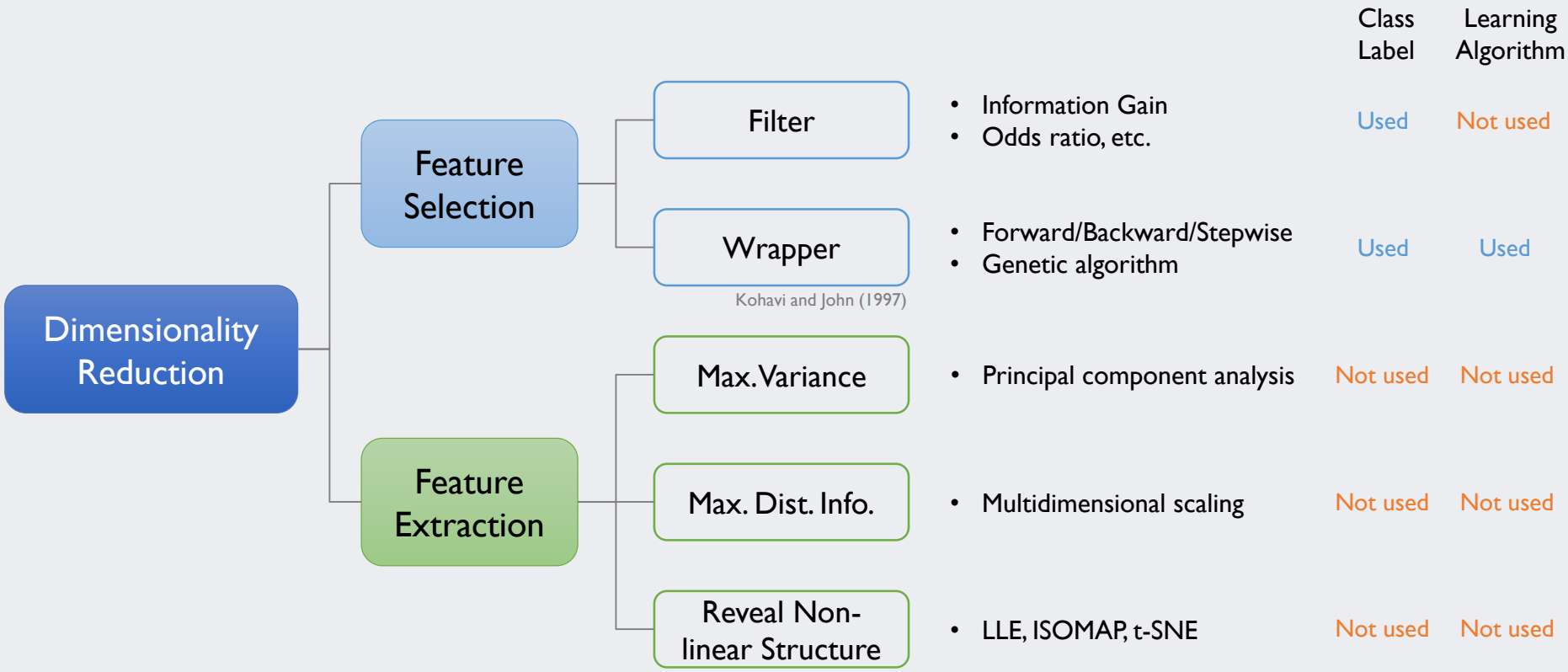
X_1	X_2	X_3	…	X_n
…	…	…	…	…
…	…	…	…	…
…	…	…	…	…
…	…	…	…	…
…	…	…	…	…

X_1	X_5	X_8
…	…	…
…	…	…
…	…	…
…	…	…
…	…	…

Z_1	Z_2	Z_3
…	…	…
…	…	…
…	…	…
…	…	…
…	…	…

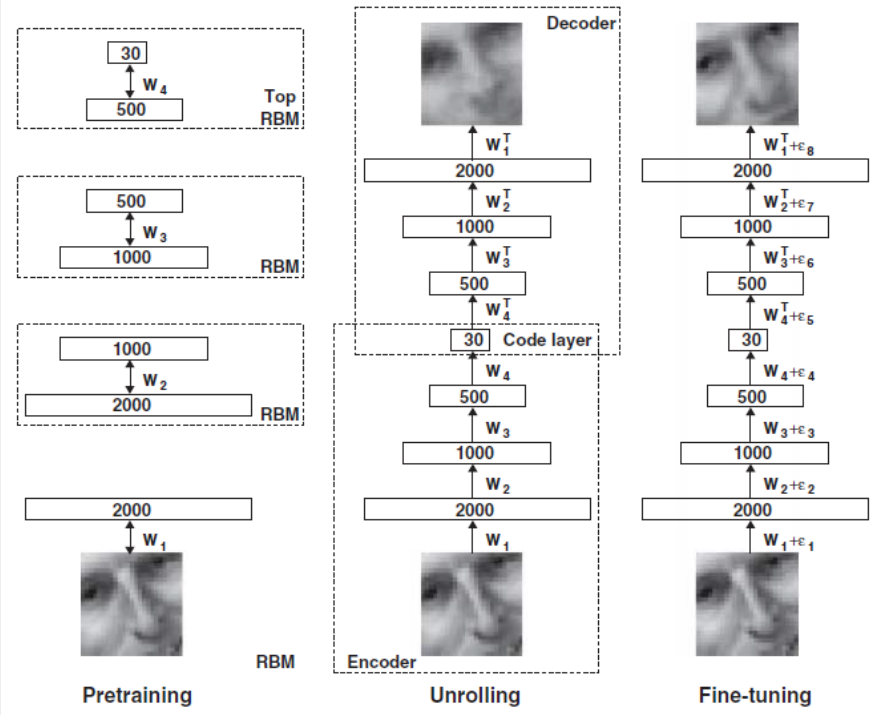
Dimensionality Reduction: Overview

- A simplified taxonomy of dimensionality reduction techniques



Dimensionality Reduction: Recent Trends

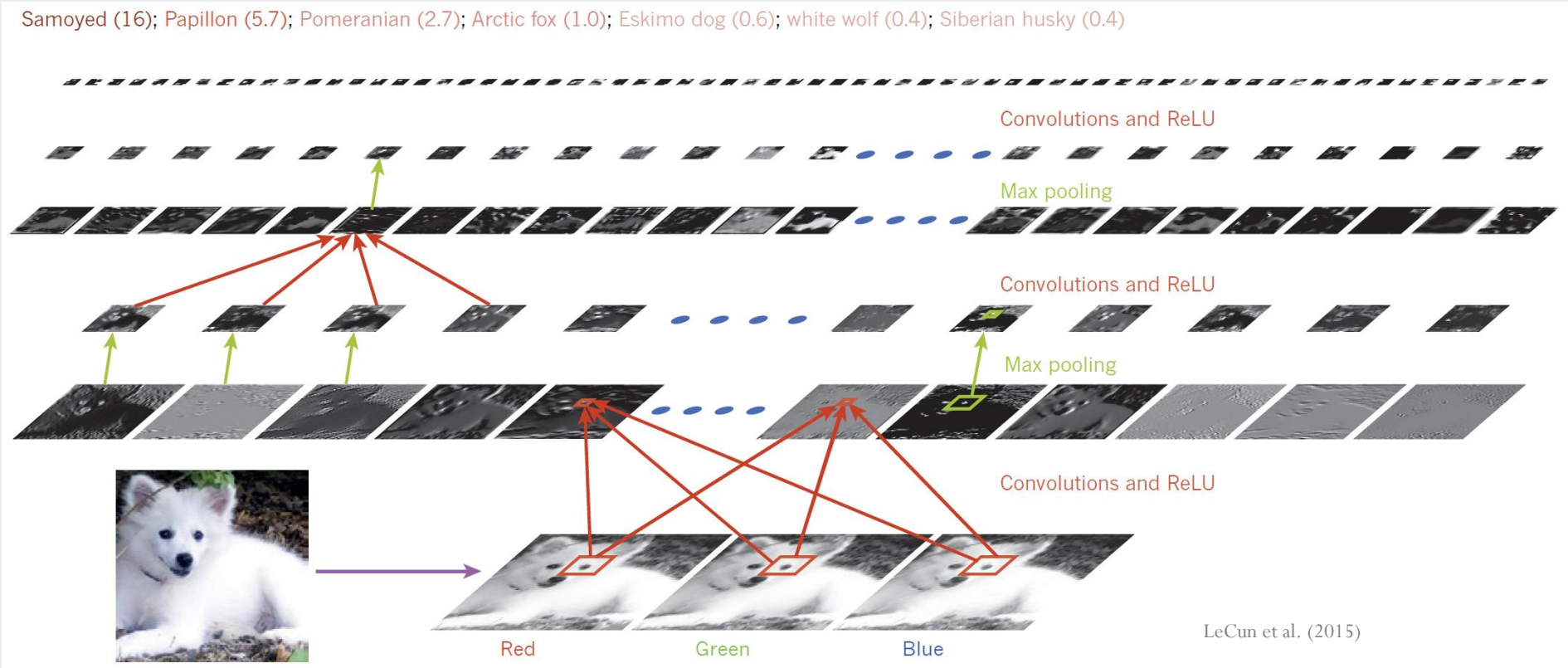
- Representation learning: Deep auto-encoder
  - ✓ Try to extract (learn) features from very low-level components (e.g., image pixels, text words, etc.)
    - Article recommendation
      - Bengio, Y., Courville, A., & Vincent, P. (2013). Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, *35*(8), 1798-1828.



Hinton and Salakhutdinov (2006)

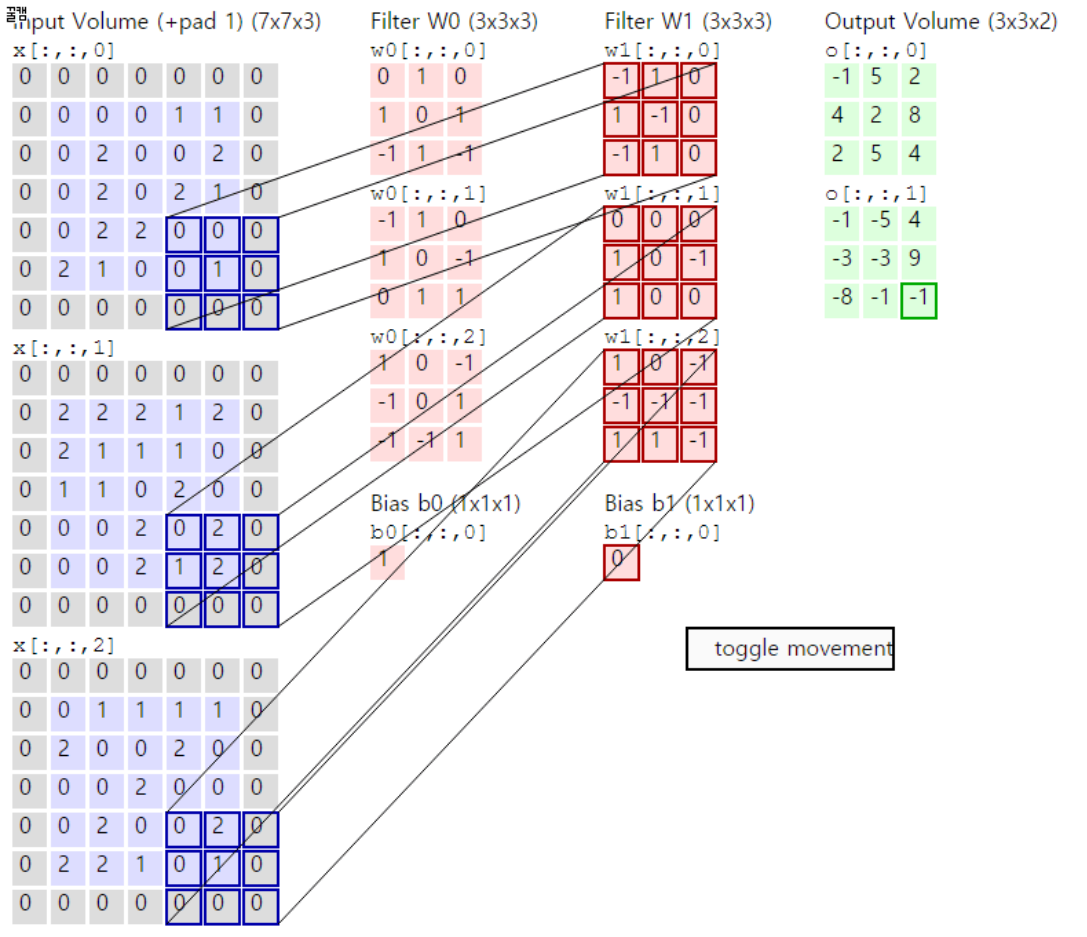
Dimensionality Reduction: Recent Trends

- Representation learning: Convolutional neural network
  - ✓ Try to extract (learn) features from very low-level components (e.g., image pixels, text words, etc.)



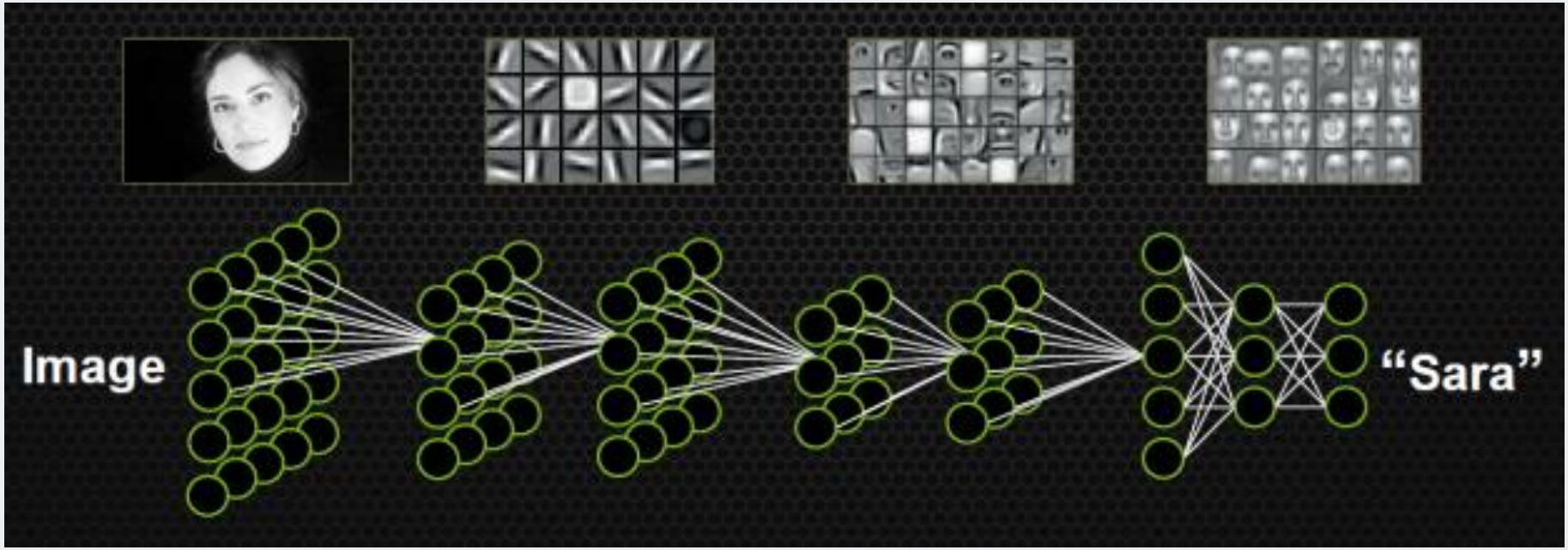
Dimensionality Reduction: Recent Trends

- Representation learning: Convolutional neural network
  - ✓ Try to extract (learn) features from very low-level components (e.g., image pixels, text words, etc.)



Dimensionality Reduction: Recent Trends

- Representation learning: Convolutional neural network
  - ✓ Try to extract (learn) features from very low-level components (e.g., image pixels, text words, etc.)

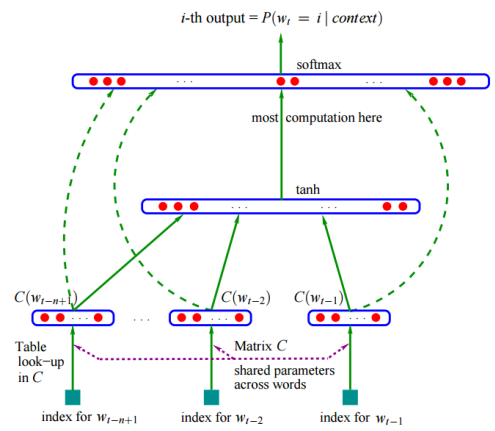


http://devblogs.nvidia.com/parallelforall/accelerate-machine-learning-cudnn-deep-neural-network-library/

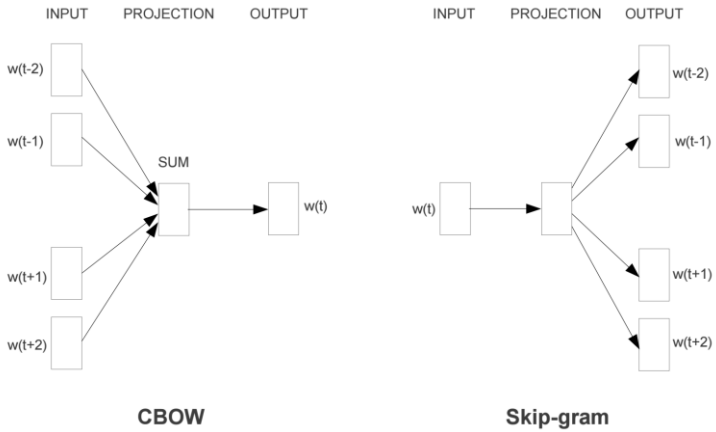
Dimensionality Reduction: Recent Trends

- Representation learning: Word/Document Embedding

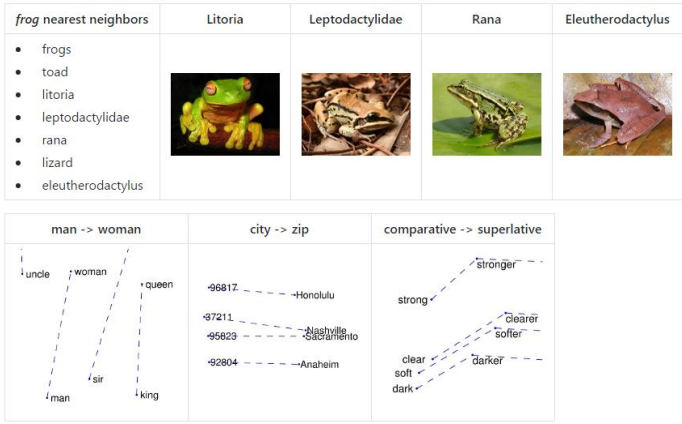
NNLM

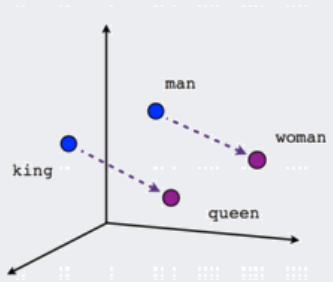


Word2Vec

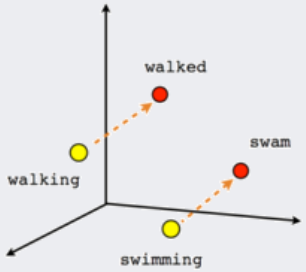


GloVe





Male-Female



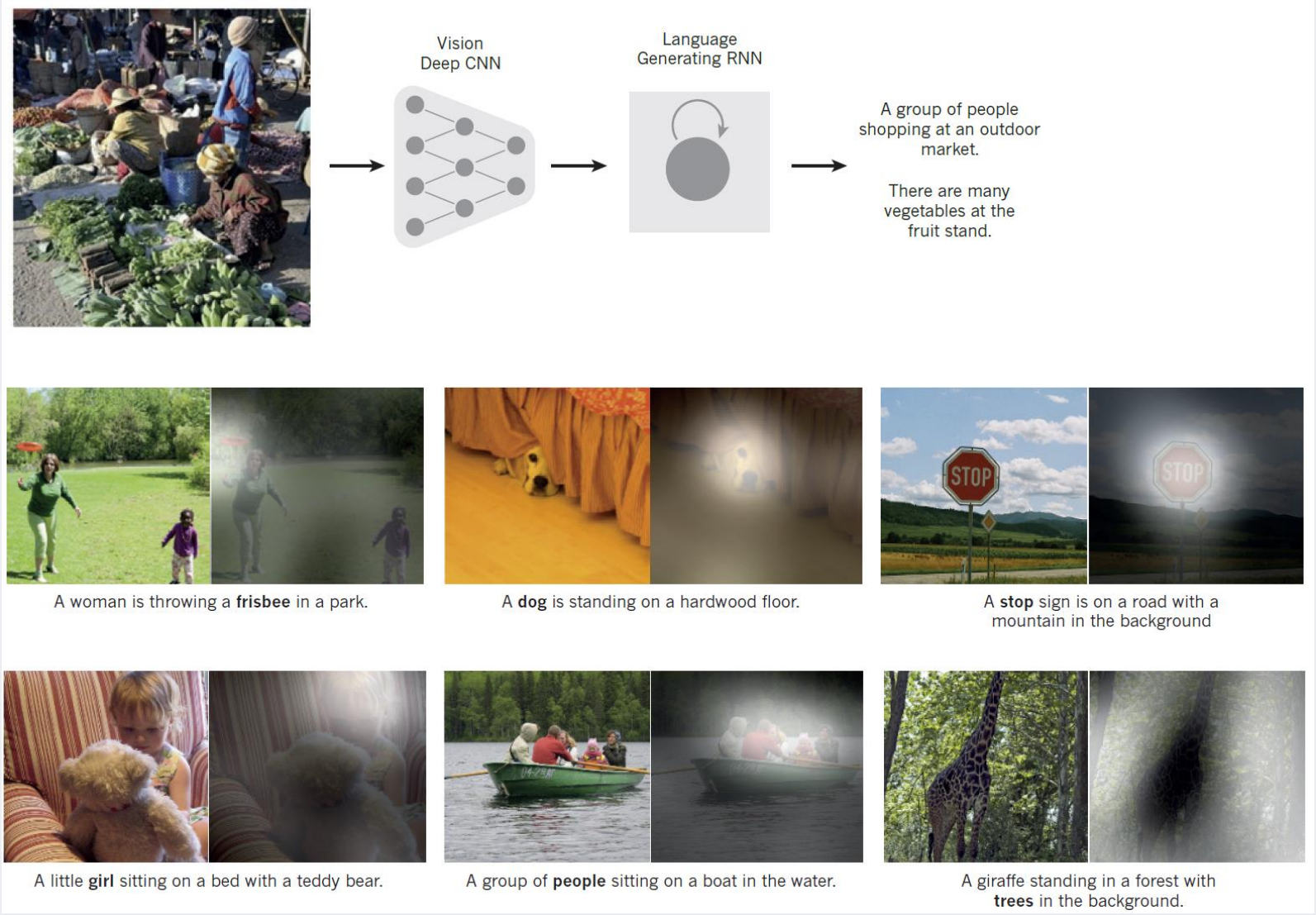
Verb tense

Country-Capital

https://www.tensorflow.org/versions/r0.7/tutorials/word2vec/index.html

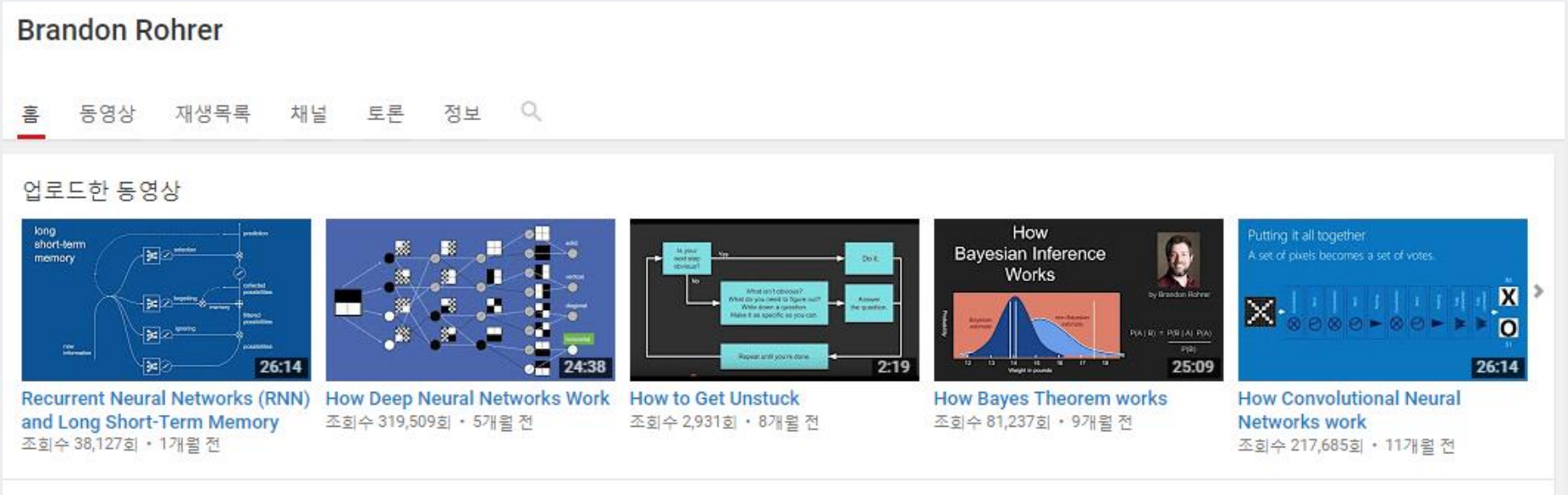
Dimensionality Reduction: Recent Trends

- Representation learning: Image+Text



Dimensionality Reduction: Recent Trends

- Video recommendation
  - ✓ Brandon Rohrer
    - Deep Learning: https://www.youtube.com/watch?v=ILsA4nyG7I0
    - CNN: https://www.youtube.com/watch?v=FmpDIaiMIeA
    - RNN: https://www.youtube.com/watch?v=WCUNPb-5EYI



AGENDA

01 Dimensionality Reduction: Overview

02 Supervised Methods

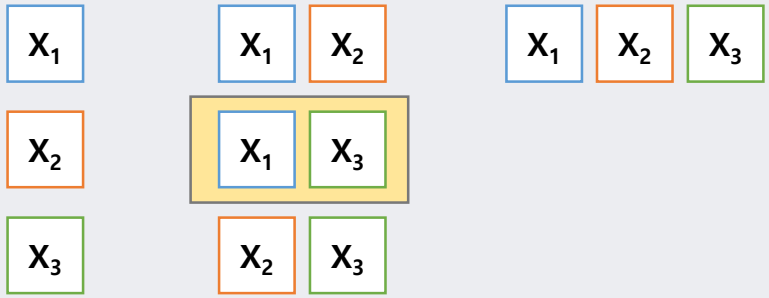
03 Unsupervised Methods: Linear Mapping

04 Unsupervised Methods: Non-linear Mapping

Exhaustive Search

- Exhaustive search
  - ✓ Search all possible combinations
    - Ex) 3 variables 

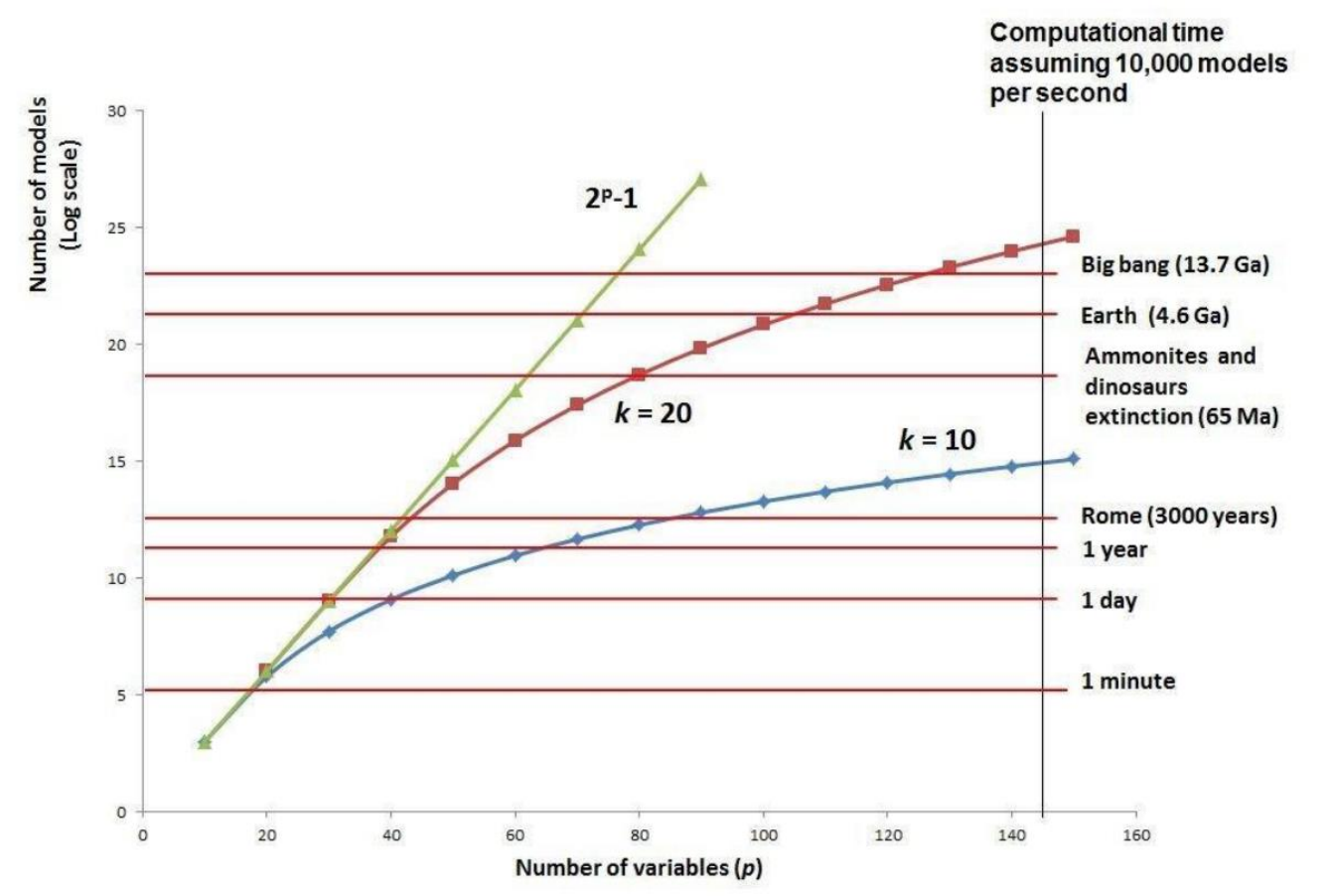
    - A total of 7 possible subsets are tested



  - ✓ Performance criteria for variable selection
    - Akaike Information Criteria (AIC), Bayesian Information Criteria (BIC), Adjusted R^2, Mallow's C_p, etc.

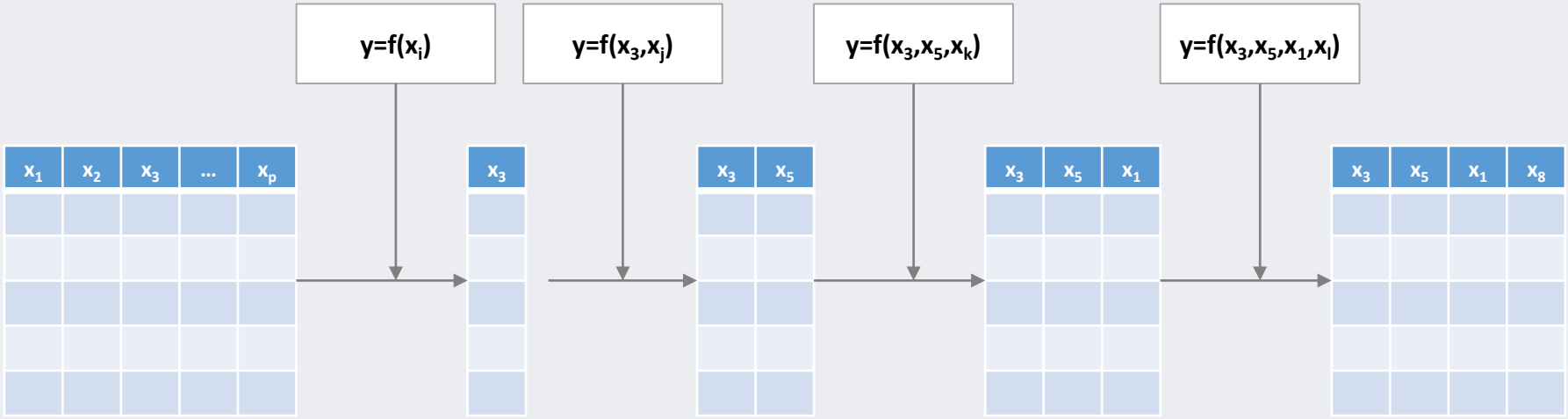
Exhaustive Search

- Exhaustive search
  - ✓ Assume that we have a computer that can evaluate 10,000 models/second



Forward Selection

- Forward selection
  - ✓ From the model with no variable, significant variables are sequentially added
  - ✓ Once the variable is selected, it will never be removed (The number of variables gradually increases)



Forward Selection

- Forward Selection
  - ✓ Forward Selection in the multiple linear regression

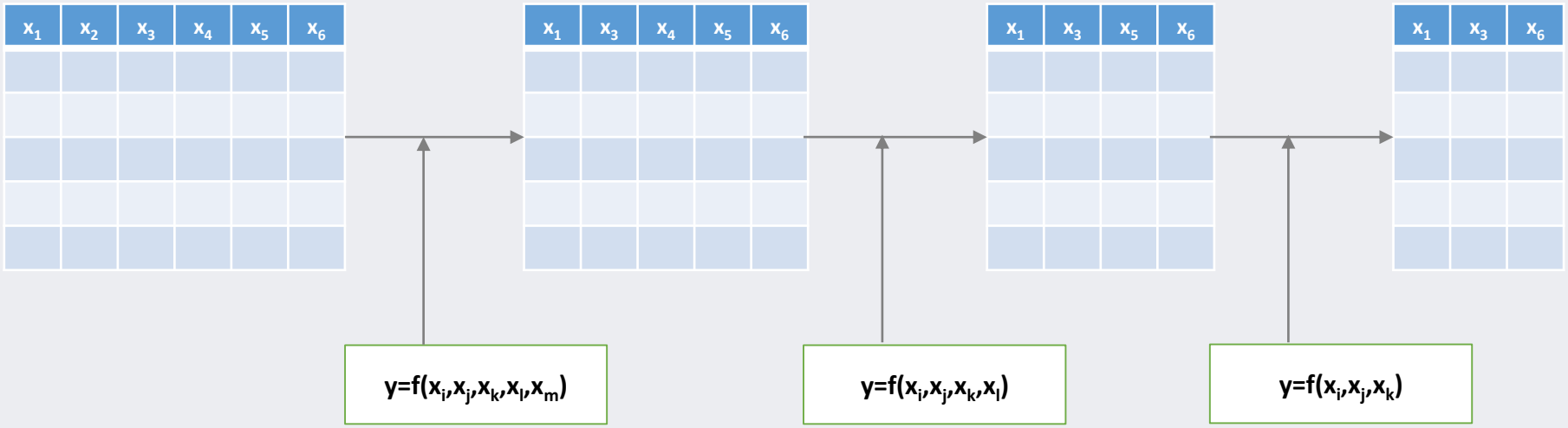
Step 5 Var CHEST Entered R-sq=0.5379 C(p)= 4.195

	DF	Sum Sq	Mean Sq	F	Prob>F
Regression	5	108.3272	21.6654	24.91	0.0001
Error	107	93.0527	0.86965099		
Total	112	201.37982301			

	Par Est	Std Error	Type II Sum Sq	F	Prob>F
Variable					
INTERCEP	-0.7680	0.6102	1.3776	1.58	0.2109
CULTURE	0.0432	0.0098	16.7198	19.23	0.0001
STAY	0.2339	0.0574	14.4381	16.60	0.0001
NRATIO	0.6724	0.2993	4.3888	5.05	0.0267
CHEST	0.0092	0.0054	2.5062	2.88	0.0925
FACIL	0.0184	0.0063	7.4571	8.57	0.0042

Backward Elimination

- Backward Elimination
  - ✓ From the model with all variables, irrelevant variables are sequentially removed
  - ✓ Once a variable is removed, it will never be selected (The number of variables gradually decreases)

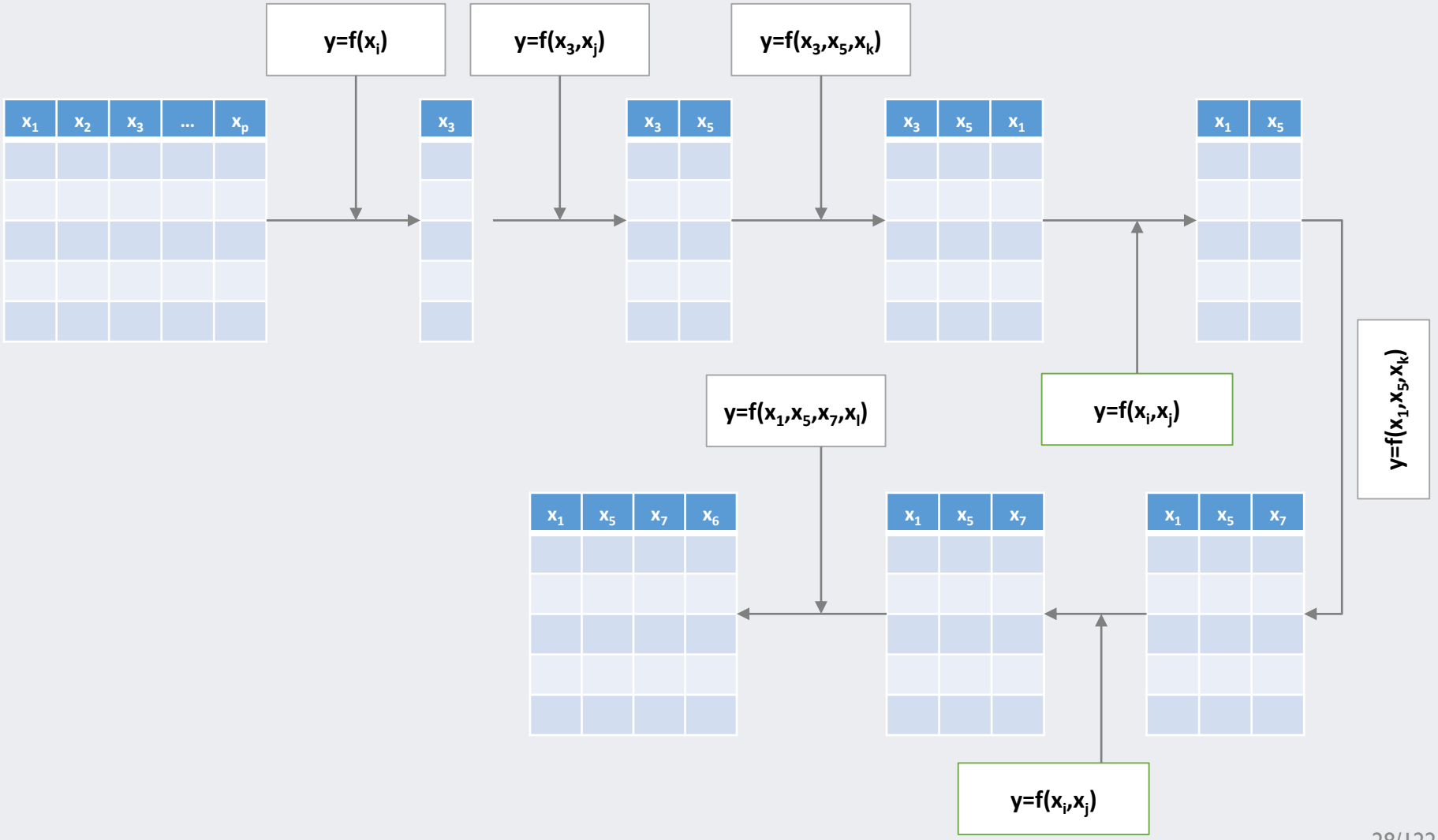


Stepwise Selection

- Stepwise Selection
  - ✓ From the model with no variable, conduct the forward selection and backward elimination alternately
  - ✓ Takes longer time than forward selection/backward elimination, but has more chances to find the optimal set of variables
  - ✓ Variables that is either selected/removed can be reconsidered for selection/removal
  - ✓ The number of variables increases in the early period, but it can either increase or decrease

Stepwise Selection

- Stepwise selection example



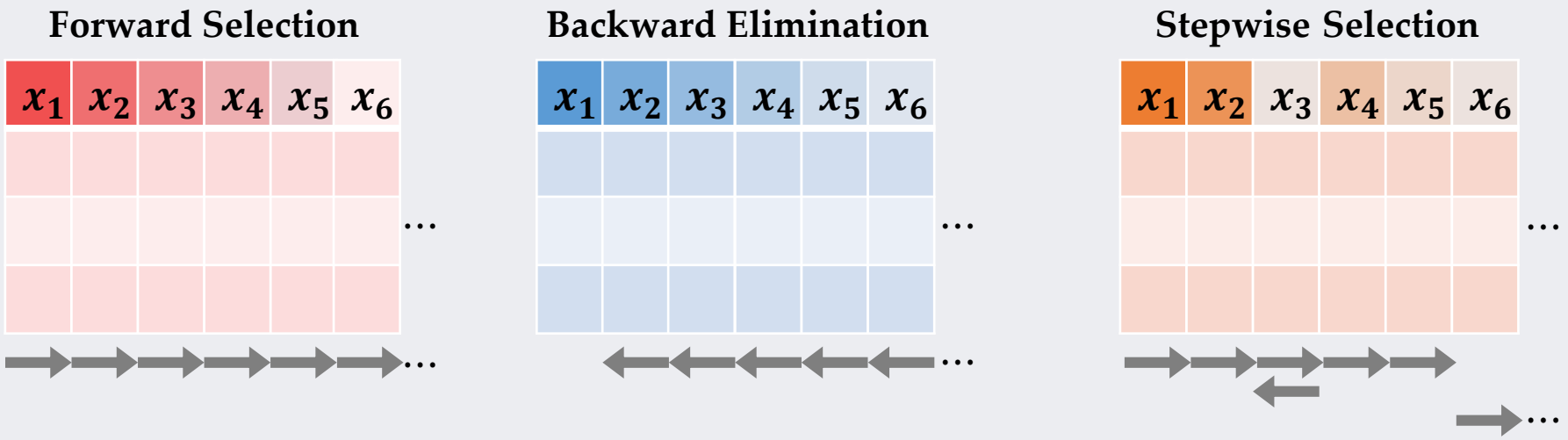
Stepwise Selection

- Stepwise Selection
  - ✓ Stepwise selection process

- Start with model with no predictors.
- Add variable with largest F-statistic (provided P less than some cut-off).
- Refit with this variable added. Recompute all F statistics for adding one of the remaining variables and add variable with largest F statistic.
- At each step after adding a variable try to eliminate any variable not significant at some level (that is, do BACKWARD elimination till that stops).
- After doing the backwards steps take another FORWARD step.
- Continue until every remaining variable is significant at cut-off level and every excluded variable is insignificant OR until variable to be added is same as last deleted variable.

Comparison among FS/BE/SS

- Illustrative Example



Performance Metrics

- Akaike Information Criteria (AIC)
  - ✓ Sum of squared error (SSE) with the number of variables as a penalty term

$$AIC = n \cdot ln\left(\frac{SSE}{n}\right) + 2k$$

- Bayesian Information Criteria (BIC)
  - ✓ SSE, number of variables, standard deviation obtained by the model with all variables

$$BIC = n \cdot ln\left(\frac{SSE}{n}\right) + \frac{2(k+2)n\sigma^2}{SSE} - \frac{2n^2\sigma^4}{SSE^2}$$

Performance Metrics

- Adjusted R^2

  ✓ Simple R^2 increases when the number of variable increases

Model 1 : $y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \epsilon$

Model 2 : $y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \ldots + \beta_{k+m} x_{k+m} \epsilon$

$$R^2(M2) \geq R^2(M1)$$

  ✓ Use the adjusted R^2 that account for the number of variables (k)

$$\text{Adjusted } R^2 = 1 - \left(\frac{n-1}{n-k-1}\right)(1-R^2) = 1 - \frac{n-1}{n-k-1}\frac{SSE}{SST}$$

Genetic Algorithm

Siedlecki, W. Sklansky (1989)

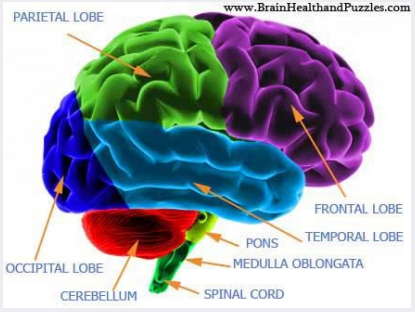
- Limitations of the previous variable selection methods
  - ✓ Exhaustive search: guarantee the optimal subset, but takes too long time (practically impossible for many tasks)
  - ✓ Local search (forward/backward/stepwise): efficient search but the search space is very limited, which leads to a low probability of finding the optimal solution

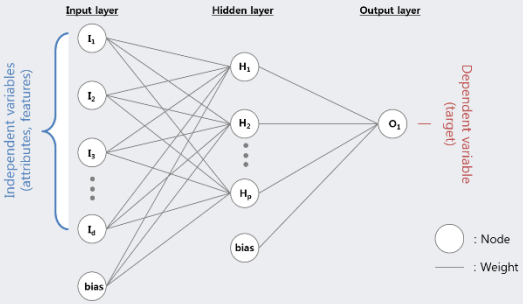
- Idea
  - ✓ Improve the performance of local searches with a little additional computational time!

Genetic Algorithm

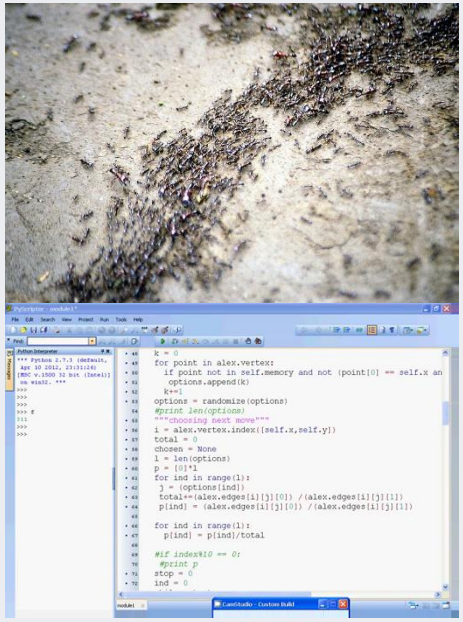
- Meta-Heuristic Approach
  - ✓ Solve a complex problem by doing trials and errors efficiently
  - ✓ Among the optimization algorithms, many of them mimic the way of a natural system works

Artificial Neural Networks

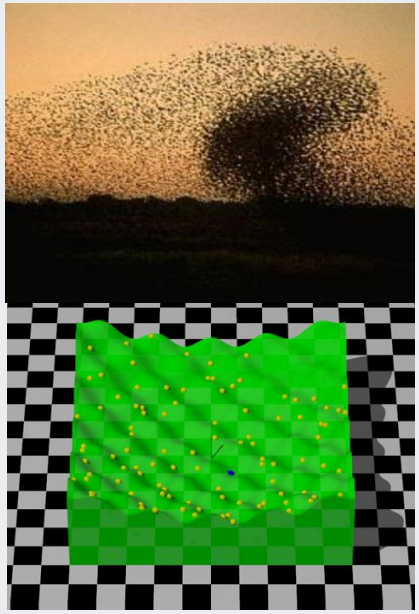




Ant Colony Algorithm

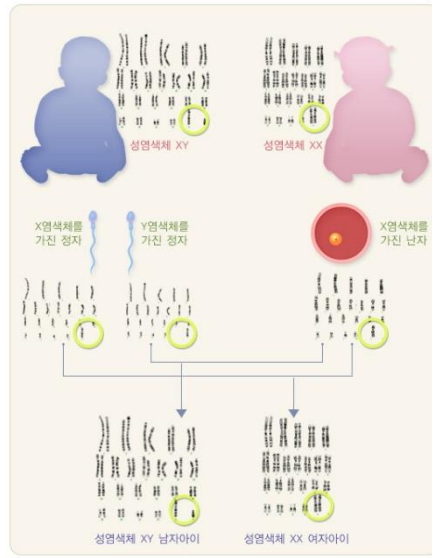


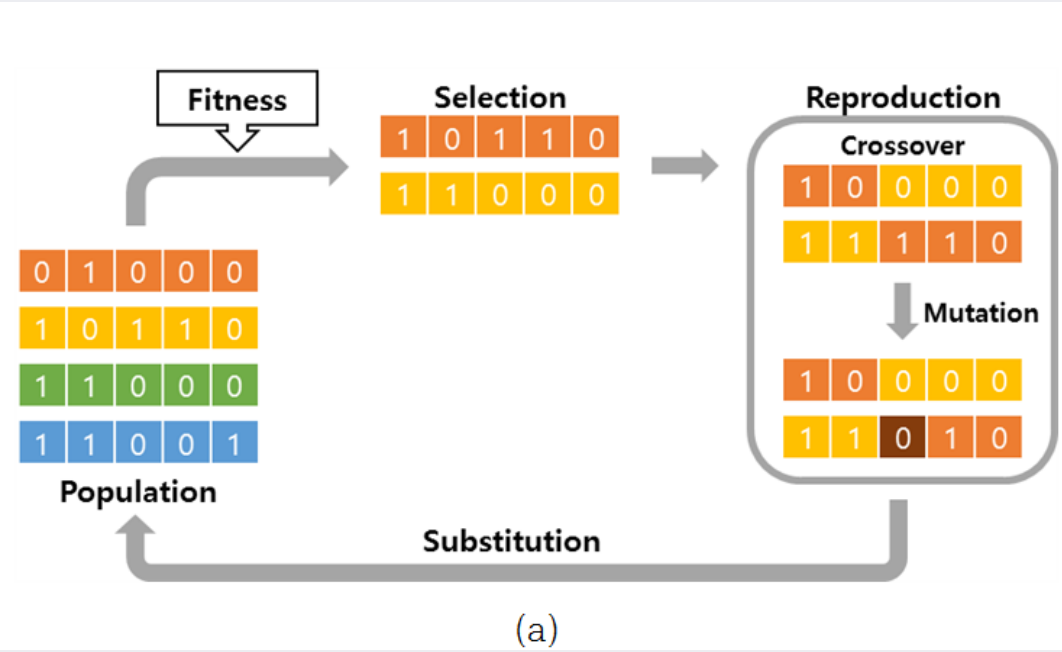
Particle Swarm Optimization

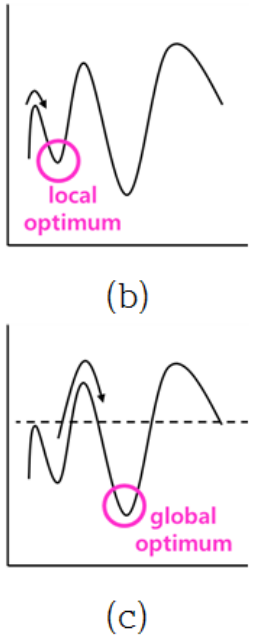


Genetic Algorithm

- An Evolutionary Algorithm that mimics the Reproduction of Creatures
  - ✓ Find a superior solutions and preserve by repeating the reproduction process
    - Selection: Select a superior solution to improve the quality
    - Crossover: Search various alternatives based on the current solutions
    - Mutation: Give a chance to escape the local optima

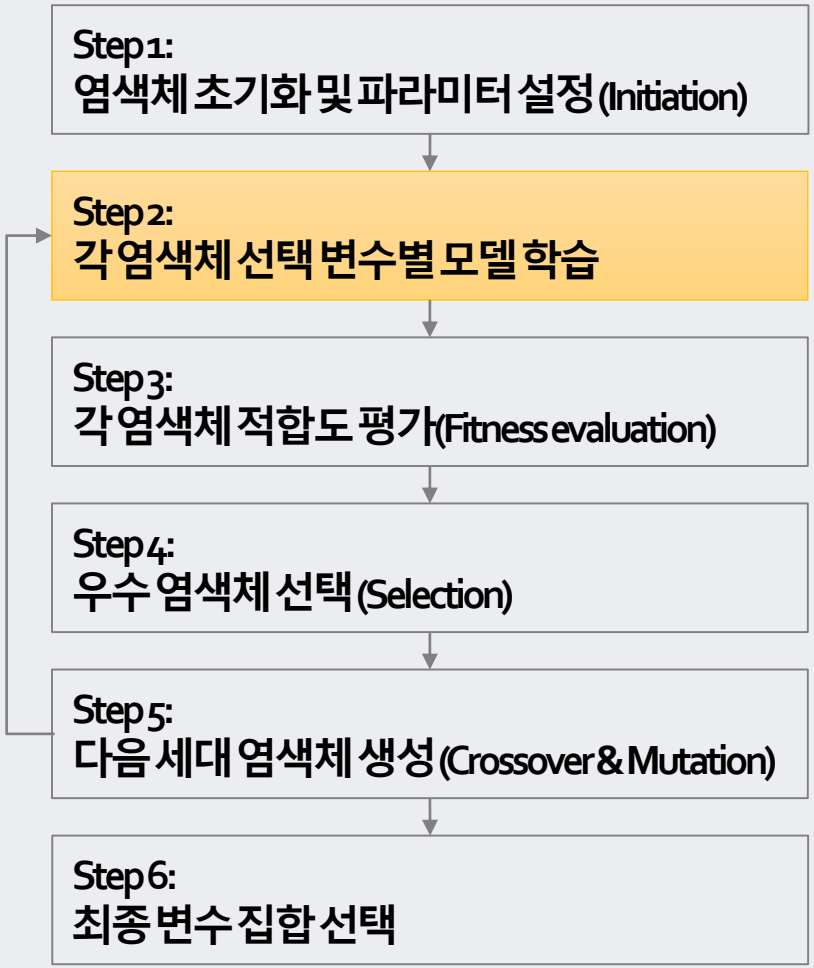


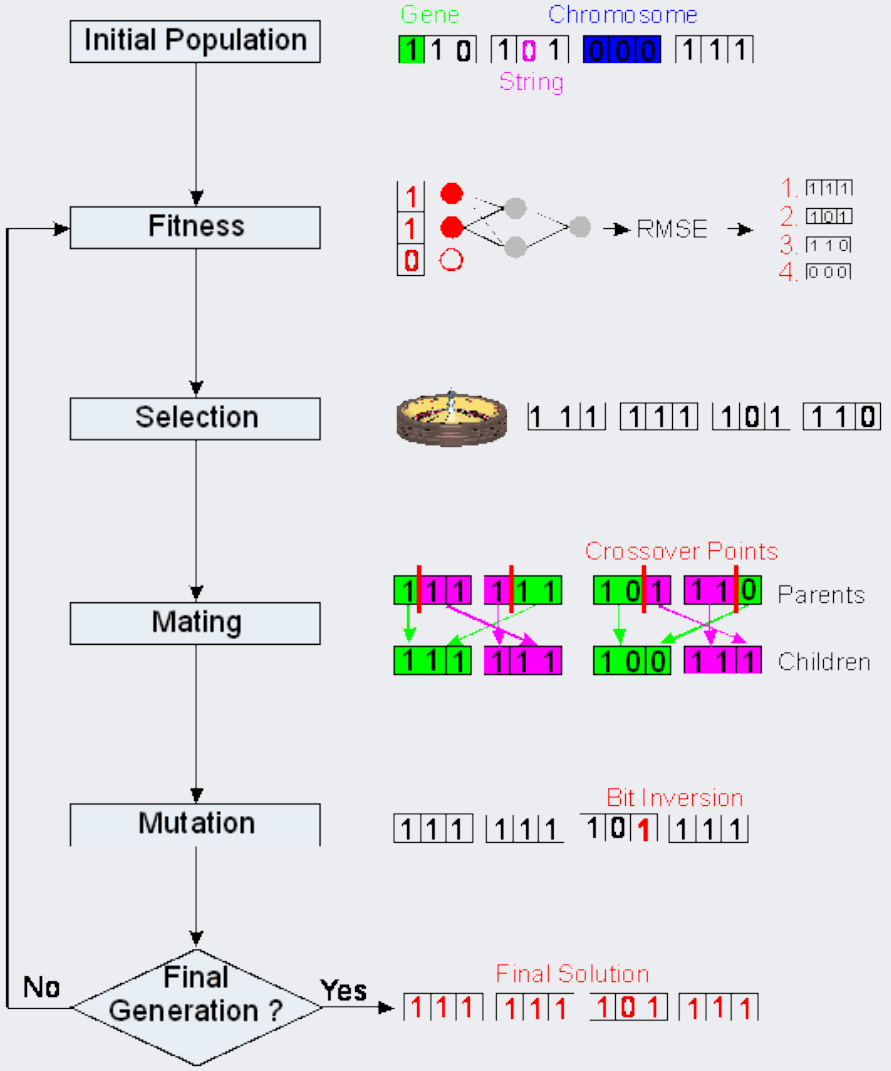




Genetic Algorithm

- Genetic Algorithm for Feature Selection





GA Step 1: Initialization

- Encoding Chromosomes
  - ✓ Genetic algorithm can be used not only for variable selection, but for a wide range of optimization problems
  - ✓ Encoding scheme can be different for different tasks
  - ✓ Binary encoding is commonly used for variable selection

Chromosome

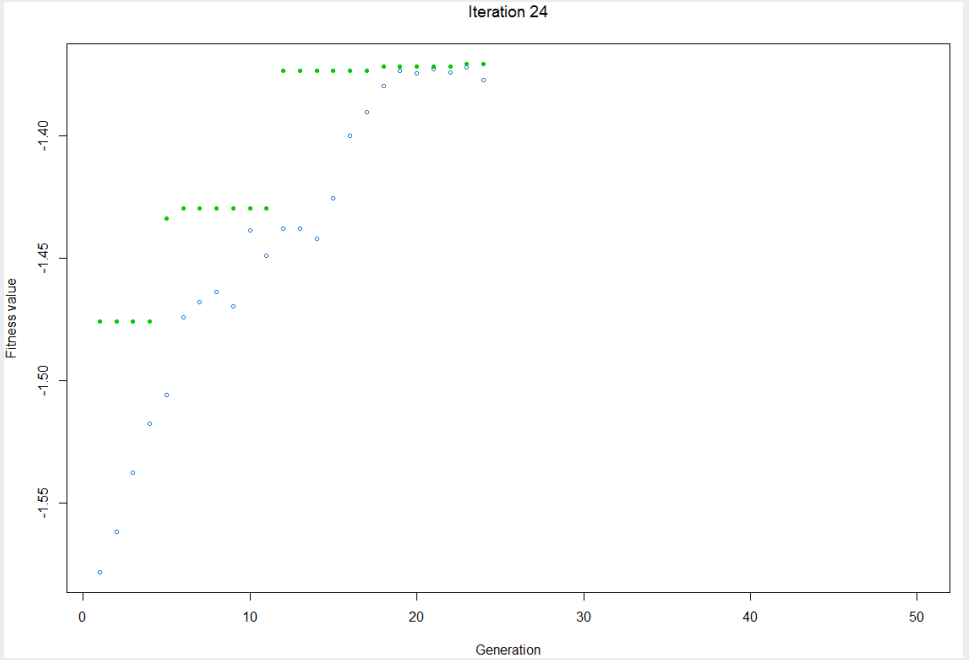
Gene

x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	...	x_d
1	0	0	1	0	1	1	0	...	1

1: Use the corresponding variable in the modeling
0: Do not use the variable

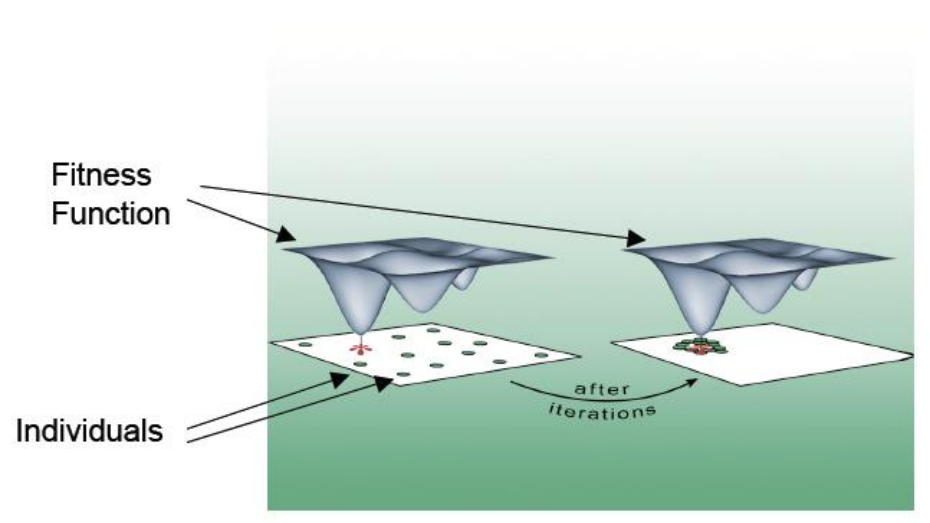
GA Step 1: Initialization

- Parameter Initialization
  - ✓ The number of chromosome (population)
  - ✓ Fitness function
  - ✓ Crossover mechanism
  - ✓ The rate of mutation
  - ✓ Stopping criteria
    - minimum fitness improvement
    - maximum iterations, etc.



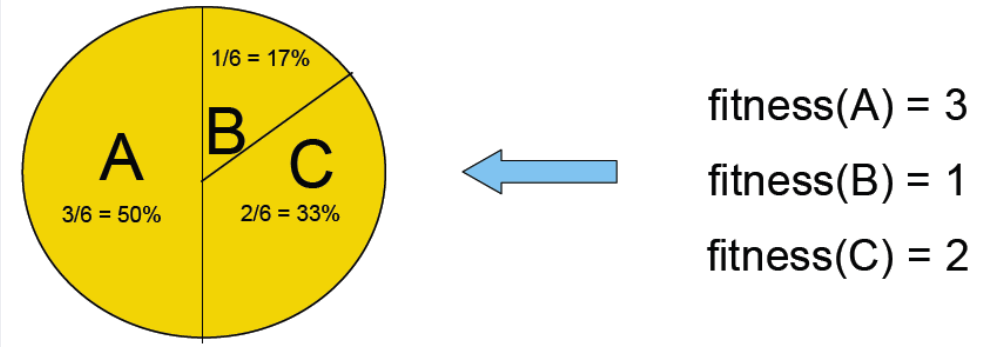
GA Step 3: Fitness Evaluation

- Fitness Function
  - ✓ A criterion that determines which chromosomes are better than others
  - ✓ In general, the higher the fitness value, the better the chromosomes
  - ✓ Common criteria that are embedded in the fitness function
    - If two chromosomes have the same fitness value, the one with fewer variables is preferred
    - If two chromosomes use the same number of variables, the one with higher predictive performance is preferred
  - ✓ In case of multiple linear regression
    - Adjusted R2
    - Akaike information criterion (AIC)
    - Bayesian information criterion (BIC)



GA Step 4: Selection

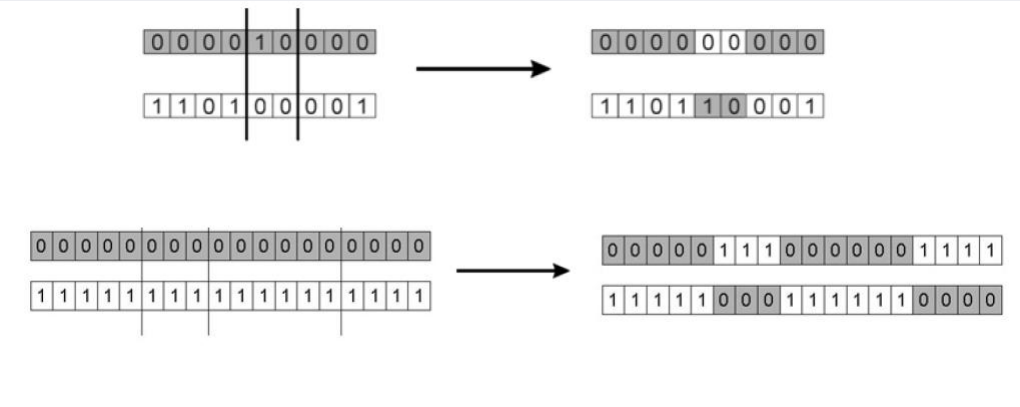
- Selection
  - ✓ Select superior chromosomes in the current population to reproduce the population of the next generation
  - ✓ Deterministic selection
    - Select only top N% of chromosomes
    - Bottom (100-N)% chromosomes are never selected
  - ✓ Probabilistic selection
    - Use the fitness value of each chromosome as the selection weight
    - All chromosomes can be selected with different probabilities



GA Step 5: Crossover & Mutation

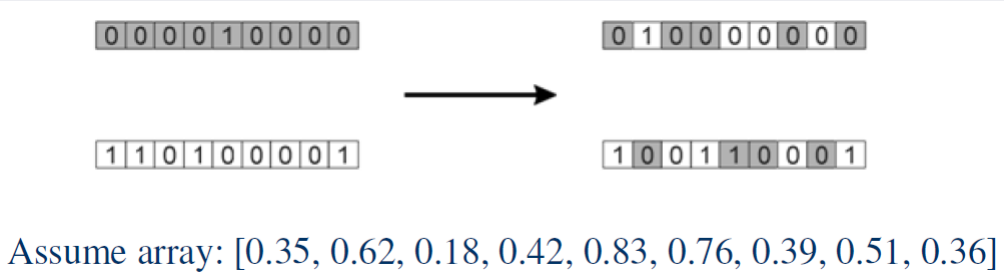
- Crossover (Reproduction)
  - ✓ Two child chromosomes are produced from two parent chromosomes
  - ✓ The number of crossover points can vary from 1 to n (total number of genes)

Crossover point = 2



Crossover points = 3

Crossover points = N



Assume array: [0.35, 0.62, 0.18, 0.42, 0.83, 0.76, 0.39, 0.51, 0.36]

GA Step 5: Crossover & Mutation

- Mutation
  - ✓ Genetic operator used to maintain diversity from one generation of a population of chromosomes to the next
  - ✓ Alters one or more gene values in a chromosome from its initial state, which result in entirely new gene values being added to the gene pool
  - ✓ By mutation, the current solution can have a chance to escape from the local optima
  - ✓ A too mutation rate can increase the time to converge (0.01 can be a good choice)

Consider the two original off-springs selected for mutation.

Original offspring 1	1	1	0	1	1	1	1	0	0	0	0	1	1	1	1	0
Original offspring 2	1	1	0	1	1	0	0	1	0	0	1	1	0	1	1	0

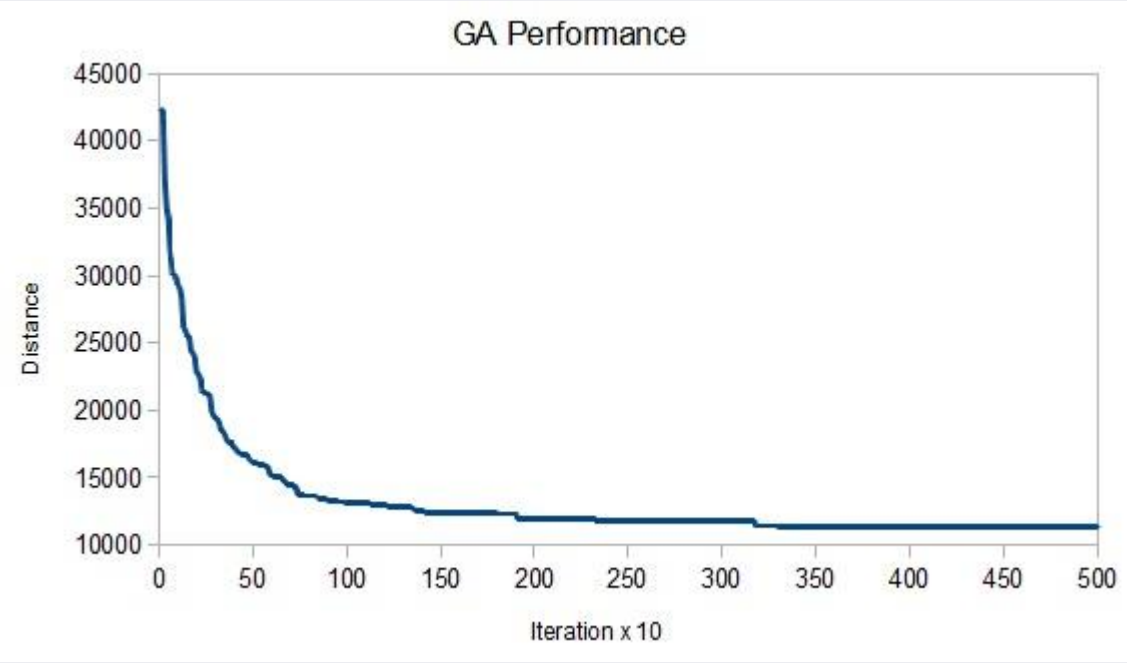
Invert the value of the chosen gene as **0** to **1** and **1** to **0**

The Mutated Off-spring produced are :

Mutated offspring 1	1	1	0	**0**	1	1	1	0	0	0	0	1	1	1	1	0
Mutated offspring 2	1	1	0	1	1	0	**1**	1	0	0	1	1	0	1	**0**	0

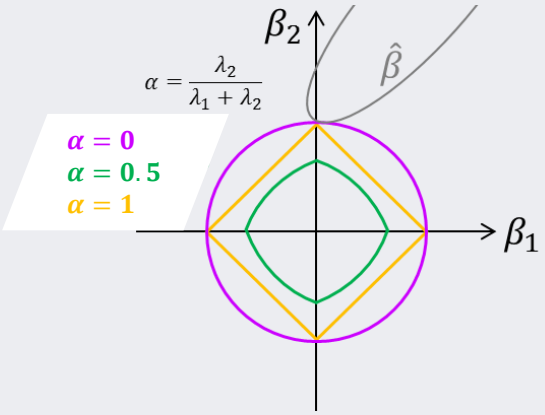
GA Step 5: Find the Best Solution

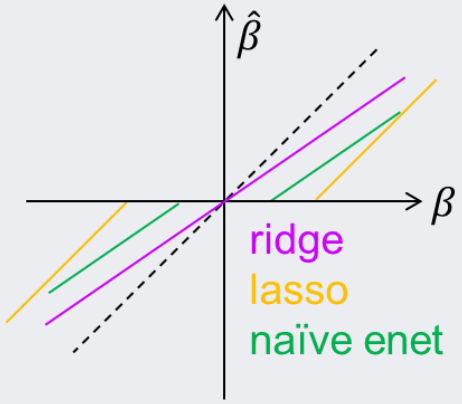
- Find the best variable subset
  - ✓ Select the chromosome with the highest fitness value after the stopping criteria are satisfied.
  - ✓ Generally, significant fitness improvement occurs in the early stages, which becomes marginal after some generations



Empirical Study

- Compare four variable selection and three shrinkage methods in linear regression
  - ✓ Variable selection: forward selection, backward elimination, stepwise selection, genetic algorithms
  - ✓ Shrinkage methods: ridge regression, lasso regression, elastic net





Ridge	$\widehat{\boldsymbol{\beta}} = \min_{\boldsymbol{\beta}} \lvert\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\rvert^2 + \lambda_1 \lvert\boldsymbol{\beta}\rvert^2$	**shrinkage**
Lasso	$\widehat{\boldsymbol{\beta}} = \min_{\boldsymbol{\beta}} \lvert\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\rvert^2 + \lambda_2 \lvert\boldsymbol{\beta}\rvert^1$	**shrinkage, variable selection**
Elastic net	$\widehat{\boldsymbol{\beta}} = \min_{\boldsymbol{\beta}} \lvert\mathbf{Y} - \mathbf{X}\boldsymbol{\beta}\rvert^2 + \lambda_2 \lvert\boldsymbol{\beta}\rvert^1 + \lambda_1 \lvert\boldsymbol{\beta}\rvert^2$	**shrinkage, variable selection, grouping effect**

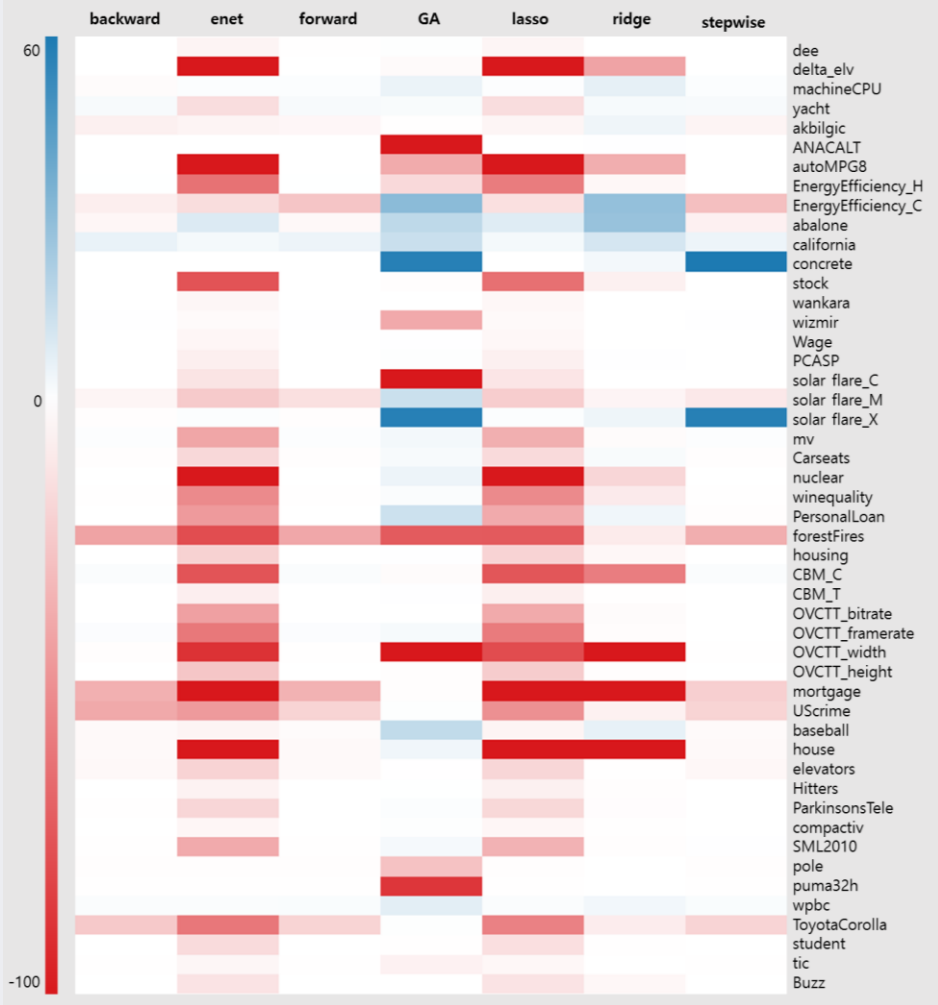
Empirical Study

- Data sets: 49 benchmark datasets

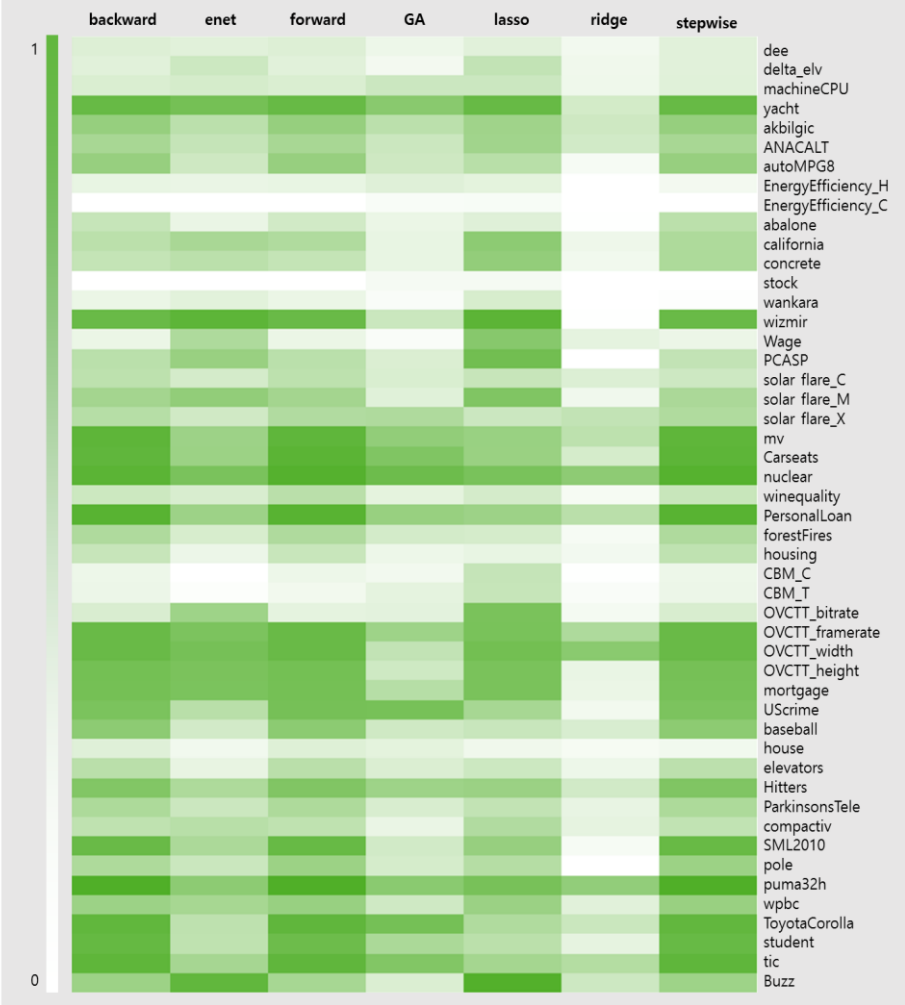
Dataset	source	records	variables	Dataset	source	records	variables
abalone	KEEL	4,177	9	OVCTT_bitrate	UCI	68,784	16
akbilgic	UCI	536	8	OVCTT_framerate	UCI	68,784	16
ANACALT	KEEL	4,052	8	OVCTT_height	UCI	68,784	16
autoMPG8	KEEL	392	8	OVCTT_width	UCI	68,784	16
baseball	KEEL	336	17	ParkinsonsTele	UCI	5,875	21
Buzz	UCI	28,179	95	PersonalLoan	etc.	2,500	13
california	KEEL	20,640	9	PCASP	UCI	45,730	10
Carseats	R	400	11	pole	KEEL	14,998	27
CBM_C	UCI	11,934	15	puma32h	KEEL	4,124	33
CBM_T	UCI	11,934	15	SML2010	UCI	4,137	24
compactiv	KEEL	8,192	22	solar flare_C	UCI	323	11
concrete	KEEL	1,030	9	solar flare_M	UCI	323	11
dee	KEEL	365	7	solar flare_X	UCI	323	11
delta_elv	KEEL	9,517	7	stock	KEEL	950	10
elevators	KEEL	16,599	19	student	UCI	382	51
EnergyEfficiency_C	UCI	768	9	tic	KEEL	9,822	86
EnergyEfficiency_H	UCI	768	9	ToyotaCorolla	etc.	1,436	34
forestFires	KEEL	517	13	UScrime	R	47	16
Hitters	R	263	20	Wage	R	3,000	10
house	KEEL	22,784	17	wankara	KEEL	1,609	10
housing	UCI	506	14	winequality	UCI	6,497	12
machineCPU	KEEL	209	7	wizmir	KEEL	1,461	10
mortgage	KEEL	1,049	16	wpbc	UCI	194	34
mv	KEEL	40,768	11	yacht	UCI	308	7
nuclear	R	32	11				

Empirical Study

Error Rate Improvement



Variable Reduction Ratio



Empirical Study

- Rankings in terms of
  - ✓ (1) Error rate improvement
  - ✓ (2) Variable reduction rate
  - ✓ (3) Computational efficiency

Variable selection technique	Error rate improvement	Variable reduction rate	Computational efficiency
Forward	5	4	1
Backward	4	3	2
Stepwise	3	2	6
GA	1	6	7
Ridge	2	7	5
LASSO	7	1	3
Enet	6	5	4

AGENDA

01 Dimensionality Reduction: Overview

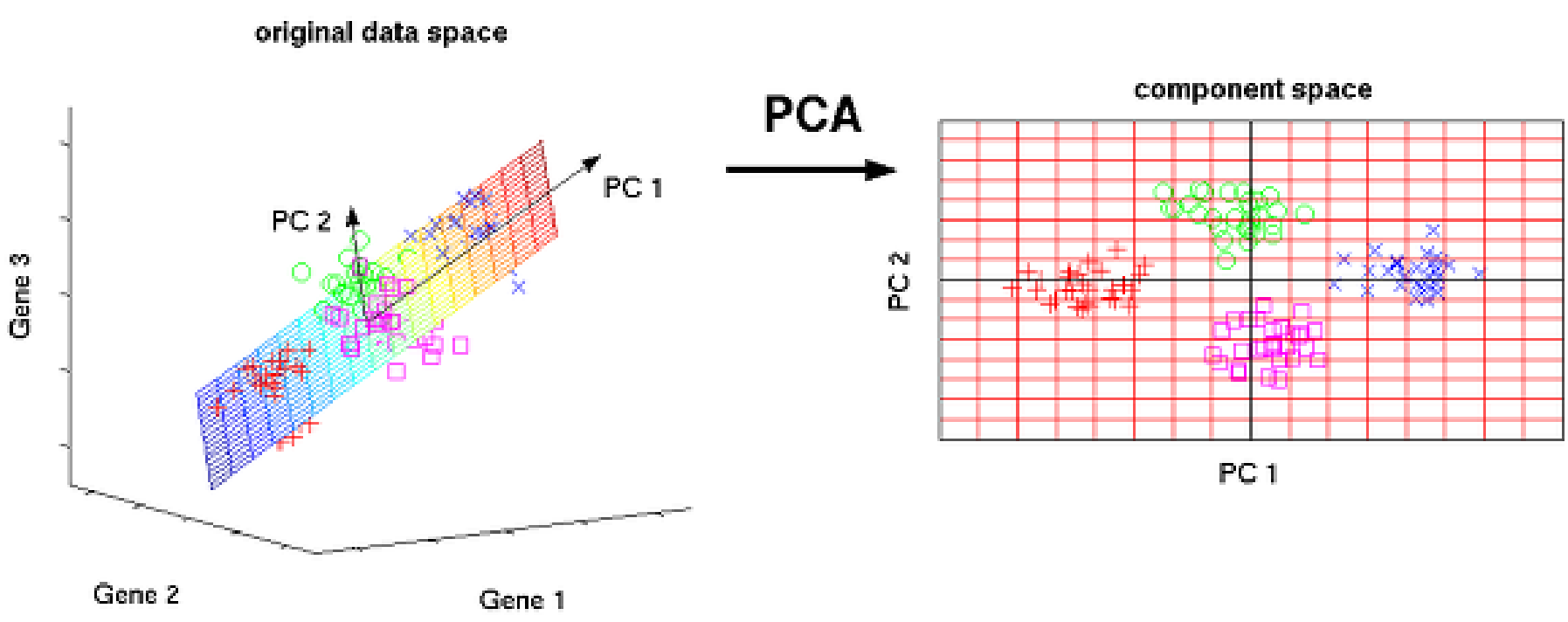
02 Supervised Methods

03 Unsupervised Methods: Linear Mapping

04 Unsupervised Methods: Non-linear Mapping

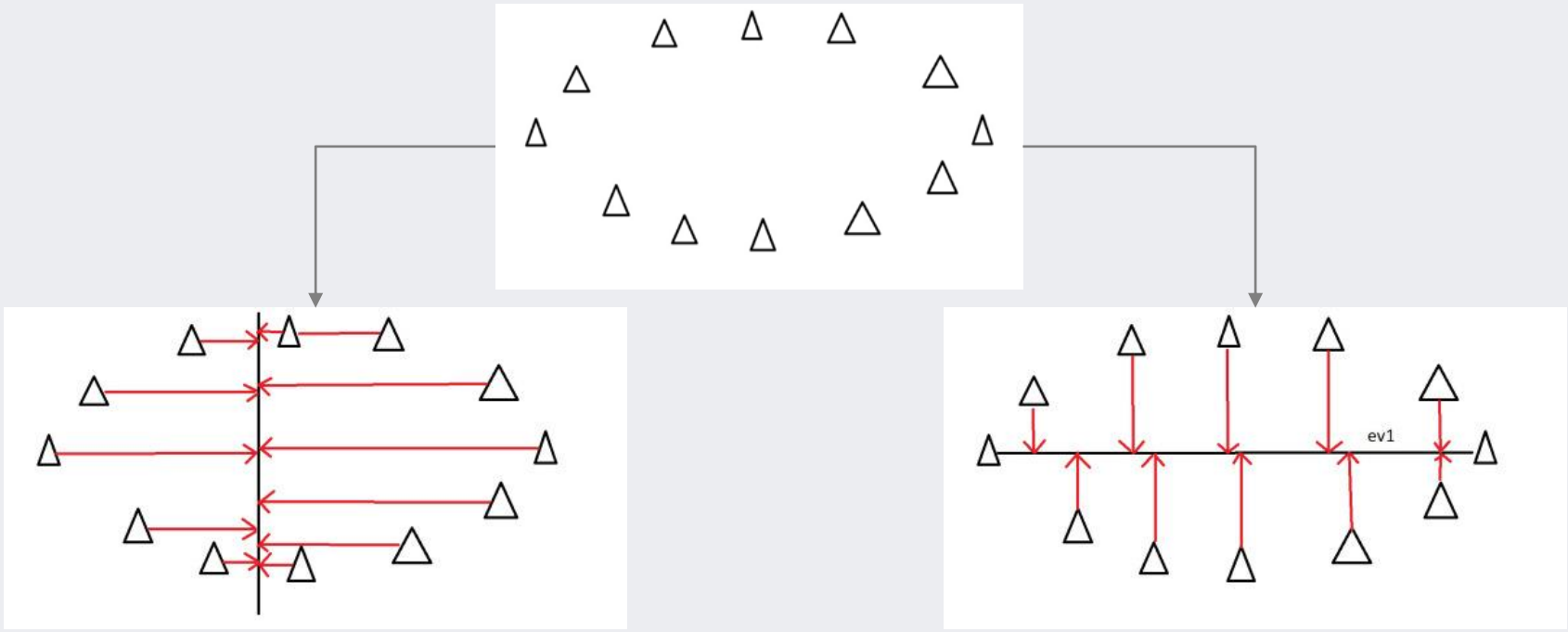
Principal Component Analysis: PCA

- Principal Component Analysis: PCA
  - ✓ To find a set orthogonal bases to preserve the variance of the original data



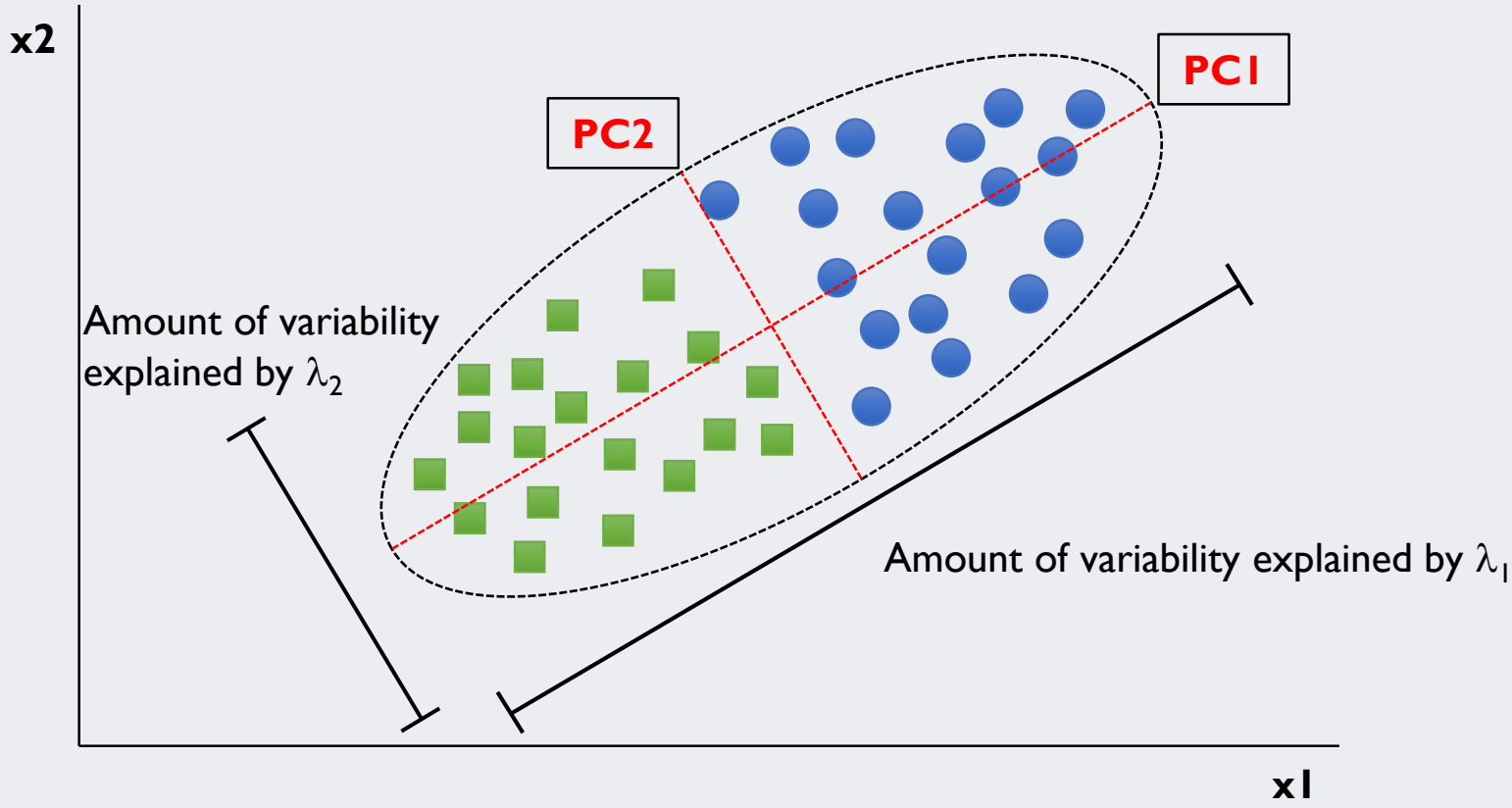
Principal Component Analysis: PCA

- Which bases are preferred?



Principal Component Analysis: PCA

- How much variance can be preserved?



Principal Component Analysis: PCA

- Principal Component Analysis: PCA
  - ✓ Covariance
    - $\mathbf{X}$: a data set (d by n, m: # of variables, n: # of records)

$$Cov(\mathbf{X}) = \frac{1}{n}(\mathbf{X} - \bar{\mathbf{X}})(\mathbf{X} - \bar{\mathbf{X}})^T$$

$$\mathbf{C}_{\mathbf{X}} = Var[\mathbf{x}] = \begin{bmatrix} Var[x_1] & Cov[x_1,x_2] & \cdots & Cov[x_1,x_N] \\ Cov[x_2,x_1] & Var[x_2] & \cdots & Cov[x_2,x_N] \\ \vdots & \vdots & \vdots & \cdots \\ Cov[x_N,x_1] & Cov[x_N,x_2] & \cdots & Var[x_N] \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_{11} & \sigma_{12} & \cdots & \sigma_{1N} \\ \sigma_{21} & \sigma_{22} & \cdots & \sigma_{2N} \\ \vdots & \vdots & \vdots & \cdots \\ \sigma_{N1} & \sigma_{N2} & \cdots & \sigma_{NN} \end{bmatrix} = \begin{bmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2N} \\ \vdots & \vdots & \vdots & \cdots \\ \sigma_{N1} & \sigma_{N2} & \cdots & \sigma_N^2 \end{bmatrix}$$

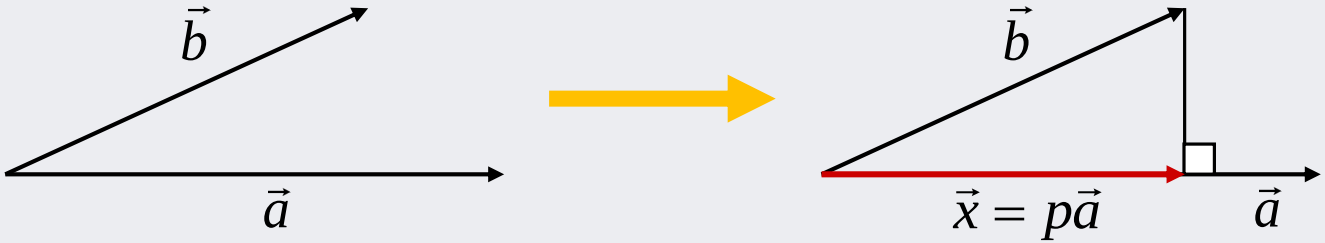
    - $\mathrm{Cov}(\mathbf{X})_{ij} = \mathrm{Cov}(\mathbf{X})_{ji}$
    - Total variance of the data set

    $= \mathrm{tr}[\mathrm{Cov}(\mathbf{X})]$

    $= \mathrm{Cov}(\mathbf{X})_{11} + \mathrm{Cov}(\mathbf{X})_{22} + \mathrm{Cov}(\mathbf{X})_{33} + \ldots + \mathrm{Cov}(\mathbf{X})_{dd}$

Principal Component Analysis: PCA

- Principal Component Analysis: PCA
  - ✓ Projection onto a basis



$$(\vec{b} - p\vec{a})^{\mathrm{T}}\vec{a} = 0 \quad \Rightarrow \quad \vec{b}^{\mathrm{T}}\vec{a} - p\vec{a}^{\mathrm{T}}\vec{a} = 0 \quad \Rightarrow \quad p = \frac{\vec{b}^{\mathrm{T}}\vec{a}}{\vec{a}^{\mathrm{T}}\vec{a}}$$

$$\vec{x} = p\vec{a} = \frac{\vec{b}^{\mathrm{T}}\vec{a}}{\vec{a}^{\mathrm{T}}\vec{a}}\vec{a}$$

If $\vec{a}$ is unit vector

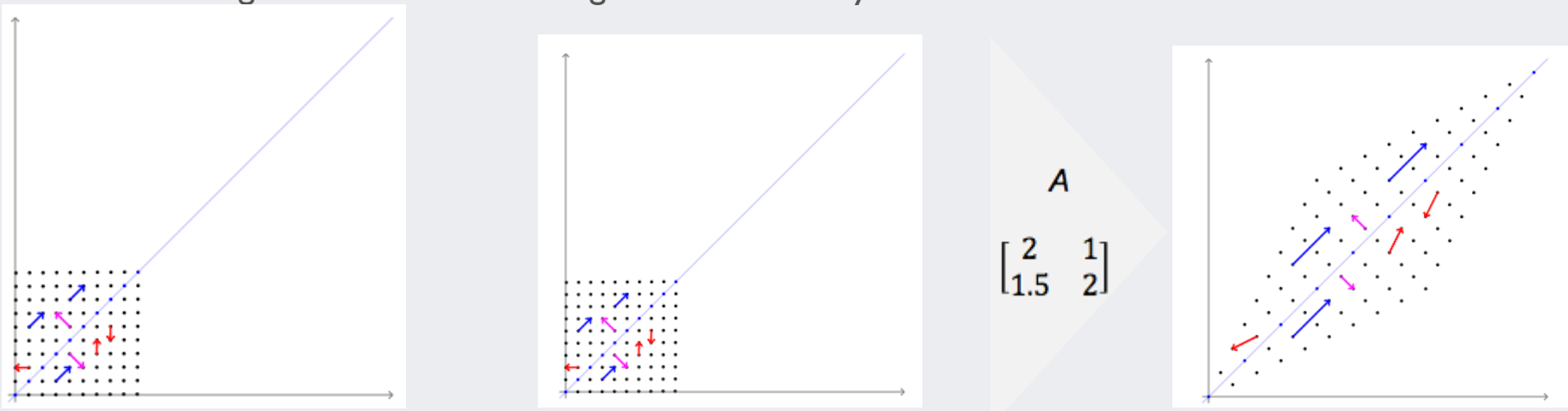
$$p = \vec{b}^{\mathrm{T}}\vec{a} \quad \Rightarrow \quad \vec{x} = p\vec{a} = (\vec{b}^{\mathrm{T}}\vec{a})\vec{a}$$

Principal Component Analysis: PCA

- Principal Component Analysis: PCA
  - ✓ Eigenvalue and eigenvector
    - When matrix A is given, scalar value λ and vector $\mathbf{x}$ that satisfy the following equation are called eigenvalue and eigenvector, respectively.

$$\mathbf{Ax} = \lambda\mathbf{x} \quad \rightarrow \quad (\mathbf{A} - \lambda\mathbf{I})\mathbf{x} = 0$$

  - ✓ Multiplying a matrix to a vector means a linear transformation is conducted.
    - Eigenvectors do not change the direction by the transformation



Principal Component Analysis: PCA

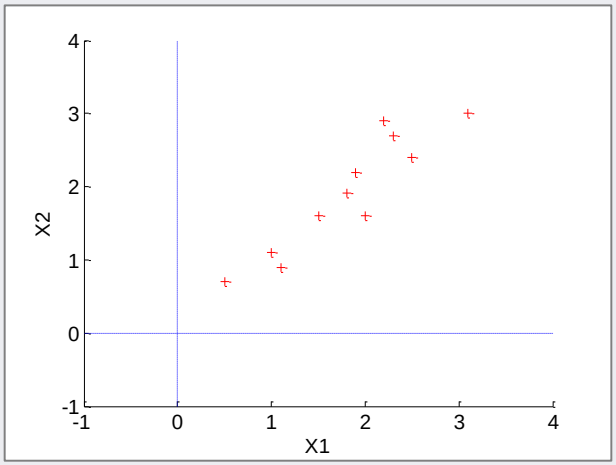
- Principal Component Analysis: PCA
  - ✓ Eigenvalue and eigenvector
  - ✓ If a matrix A is non-singular d by d matrix,
    - There exist d eigenvalue-eigenvector pairs
    - Eigenvectors are orthogonal to each other
    - $\mathrm{tr}(\mathbf{A}) = \lambda_1 + \lambda_2 + \lambda_3 + \ldots + \lambda_d$

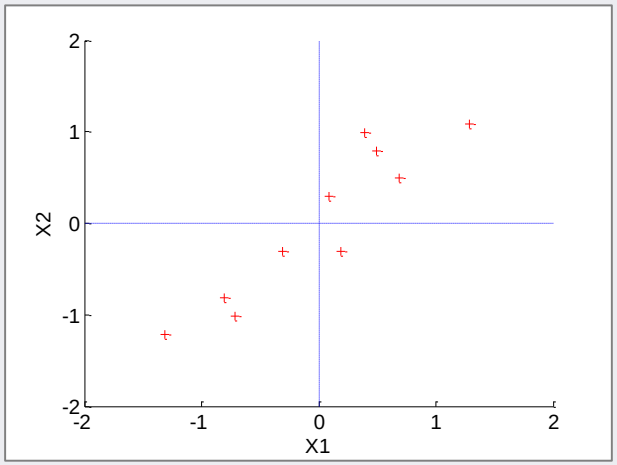
PCA Procedure

- Step 1: Data Centering
  - ✓ Make the mean of the variables equal to 0

x_1	2.5	0.5	2.2	1.9	3.1	2.3	2	1	1.5	1.1
x_2	2.4	0.7	2.9	2.2	3	2.7	1.6	1.1	1.6	0.9

x_1	0.69	-1.31	0.39	0.09	1.29	0.49	0.19	-0.81	-0.31	-0.71
x_2	0.49	-1.21	0.99	0.29	1.09	0.79	-0.31	-0.81	-0.31	-1.01





PCA Procedure

- Step 2: Formulate the optimization problem
  - ✓ If a vector x is projected onto a basis w, then the variance after the projection becomes

$$v = (\mathbf{w}^T\mathbf{X})(\mathbf{w}^T\mathbf{X})^T = \mathbf{w}^T\mathbf{X}\mathbf{X}^T\mathbf{w} = n\mathbf{w}^T\mathbf{S}\mathbf{w}$$

    - S is the sample covariance matrix where **x** is normalized.

  - ✓ The purpose of PCA is to maximize the variance V after projection

$$\max \quad \mathbf{w}^T\mathbf{S}\mathbf{w}$$
$$s.t. \quad \mathbf{w}^T\mathbf{w} = 1$$

$$\mathbf{S} = \begin{bmatrix} 0.6166 & 0.6154 \\ 0.6154 & 0.7166 \end{bmatrix}$$

PCA Procedure

- Step 3: Obtain the solution
  - ✓ By employing Lagrangian multiplier,

$$\begin{aligned} \max \quad & \mathbf{w}^T \mathbf{S} \mathbf{w} \\ s.t. \quad & \mathbf{w}^T \mathbf{w} = 1 \end{aligned}$$

$$L = \mathbf{w}^T \mathbf{S} \mathbf{w} - \lambda(\mathbf{w}^T \mathbf{w} - 1)$$

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{S}\mathbf{w} - \lambda \mathbf{w} = 0 \;\Rightarrow (\mathbf{S} - \lambda \mathbf{I})\mathbf{w} = 0$$

$$Eigenvectors = \begin{bmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{bmatrix} \qquad Eigenvalues = (1.2840 \quad 0.0491)$$

PCA Procedure

- Step 4: Find the base set of bases
  - ✓ Sort the eigenvectors in a descending order of eigenvalues

$$FeatureVector = (eig_1, eig_2, \cdots eig_n)$$

$$FeatureVector = \begin{pmatrix} 0.6779 & -0.7352 \\ 0.7352 & 0.6779 \end{pmatrix}$$

    - One basis can preserve 96% of the original variance in this example (1.2840/(0.0491+1.2840))
  - ✓ Let $\mathbf{w}_1$ be one of the eigenvectors and λ_1 be the corresponding eigenvalue.
  - ✓ The variation of the samples projected onto $\mathbf{w}_1$ is

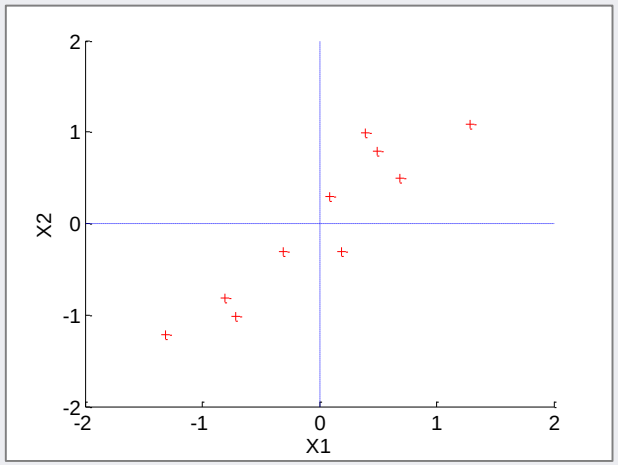
$$\mathbf{v} = (\mathbf{w}_1^T \mathbf{X})(\mathbf{w}_1^T \mathbf{X})^T = \mathbf{w}_1^T \mathbf{X}\mathbf{X}^T \mathbf{w}_1 = \mathbf{w}_1^T \mathbf{S}\mathbf{w}_1$$

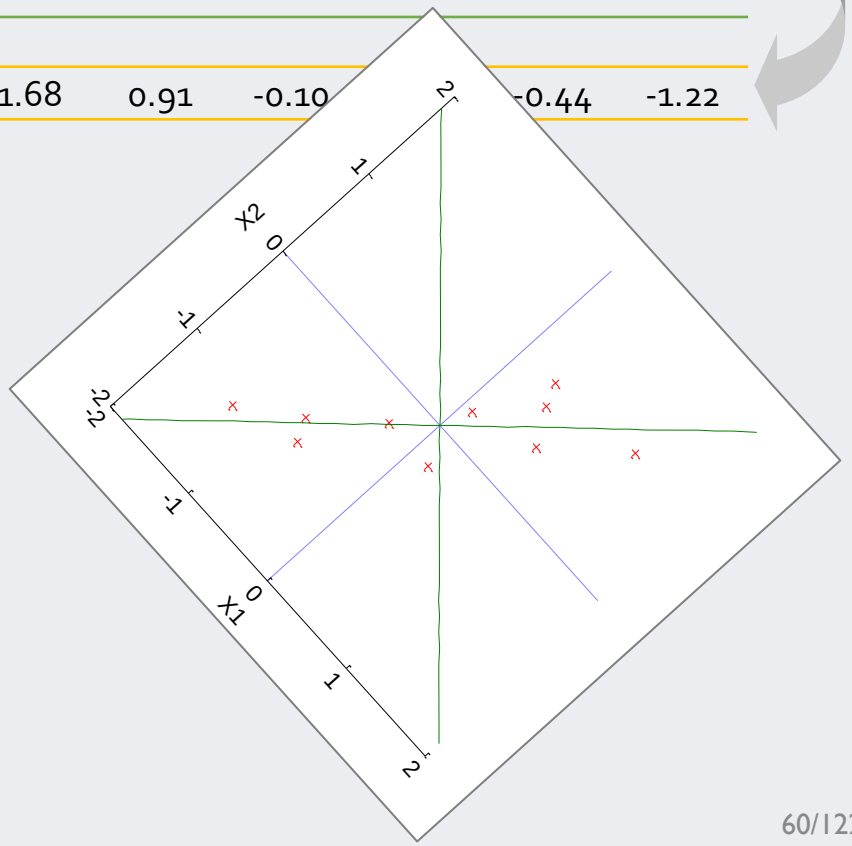
$$\text{Since } \mathbf{S}\mathbf{w}_1 = \lambda_1 \mathbf{w}_1, \quad \mathbf{w}_1^T \mathbf{S}\mathbf{w}_1 = \mathbf{w}_1^T \lambda_1 \mathbf{w}_1 = \lambda_1 \mathbf{w}_1^T \mathbf{w}_1 = \lambda_1$$

PCA Procedure

- Step 5: Extract new features
  - ✓ Project the original data onto the selected bases

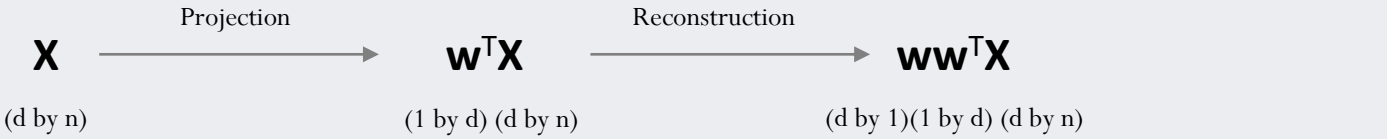
x_1	0.69	-1.31	0.39	0.09	1.29	0.49	0.19	-0.81	-0.31	-0.71
x_2	0.49	-1.21	0.99	0.29	1.09	0.79	-0.31	-0.81	-0.31	-1.01
z_1	0.83	-1.78	0.99	0.27	1.68	0.91	-0.10	-0.99	-0.44	-1.22





PCA Procedure

- Step 6: Reconstruct the original data
  - ✓ Reconstruct the data from the projected space into the original space



x_1	0.69	-1.31	0.39	0.09	1.29	0.49	0.19	-0.81	-0.31	-0.71
x_2	0.49	-1.21	0.99	0.29	1.09	0.79	-0.31	-0.81	-0.31	-1.01
z_1	0.83	-1.78	0.99	0.27	1.68	0.91	-0.10	-1.14	-0.44	-1.22
x'_1	0.56	-1.21	0.67	0.19	1.14	0.62	-0.07	-0.78	-0.30	-0.83
x'_2	0.61	-1.31	0.73	0.20	1.23	0.67	-0.07	-0.84	-0.32	-0.90

PCA Procedure

- PCA Example
  - ✓ Original data

시리얼 이름	제조업체명	유형	칼로리	단백질	지방	나트륨	식이섬유	복합탄수화물	설탕	칼륨	비타민
100% Bran	N	C	70	4	1	130	10	5	6	280	25
100% Natural Bran	Q	C	120	3	5	15	2	8	8	135	0
All-Bran	K	C	70	4	1	260	9	7	5	320	25
All-Bran with Extra Fiber	K	C	50	4	0	140	14	8	0	330	25
Almond Delight	R	C	110	2	2	200	1	14	8		25
Apple Cinnamon Cheerios	G	C	110	2	2	180	1.5	10.5	10	70	25
Apple Jacks	K	C	110	2	0	125	1	11	14	30	25
Basic 4	G	C	130	3	2	210	2	18	8	100	25
Bran Chex	R	C	90	2	1	200	4	15	6	125	25
Bran Flakes	P	C	90	3	0	210	5	13	5	190	25
Cap'n'Crunch	Q	C	120	1	2	220	0	12	12	35	25
Cheerios	G	C	110	6	2	290	2	17	1	105	25
Cinnamon Toast Crunch	G	C	120	1	3	210	0	13	9	45	25
Clusters	G	C	110	3	2	140	2	13	7	105	25
Cocoa Puffs	G	C	110	1	1	180	0	12	13	55	25
Corn Chex	R	C	110	2	0	280	0	22	3	25	25
Corn Flakes	K	C	100	2	0	290	1	21	2	35	25
Corn Pops	K	C	110	1	0	90	1	13	12	20	25
Count Chocula	G	C	110	1	1	180	0	12	13	65	25
Cracklin' Oat Bran	K	C	110	3	3	140	4	10	7	160	25

PCA Procedure

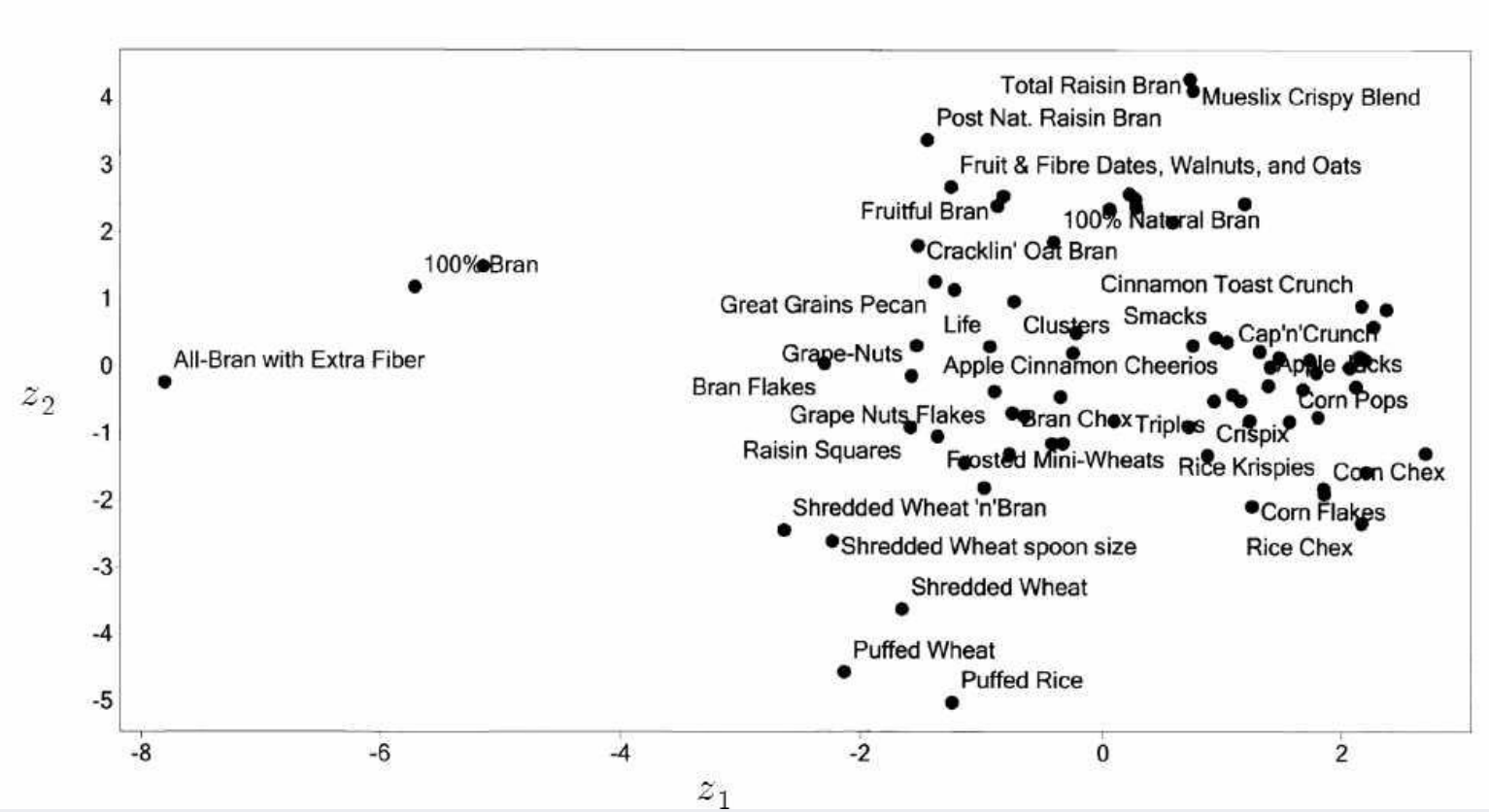
- PCA Example
  - ✓ Eigenvectors eigenvalues for each principal component

변수이름	1	2	3	4	5	6	7
calories	0.2995424	0.39314792	0.11485746	0.20435865	0.20389892	-0.25590625	-0.02559552
protein	-0.30735639	0.16532333	0.27728197	0.30074316	0.319749	0.120752	0.28270504
fat	0.03991544	0.34572428	-0.20489009	0.18683317	0.58689332	0.34796733	-0.05115468
sodium	0.18339655	0.13722059	0.38943109	0.12033724	-0.33836424	0.66437215	-0.28370309
fiber	-0.45349041	0.17981192	0.06976604	0.03917367	-0.255119	0.0642436	0.11232537
carbo	0.19244903	-0.14944831	0.56245244	0.0878355	0.18274252	-0.32639283	-0.26046798
sugars	0.22806853	0.35143444	-0.35540518	-0.02270711	-0.31487244	-0.15208226	0.22798519
potass	-0.40196434	0.30054429	0.06762024	0.09087842	-0.14836049	0.02515389	0.14880823
vitamins	0.11598022	0.1729092	0.38785872	-0.6041106	-0.04928682	0.12948574	0.29427618
shelf	-0.17126338	0.26505029	-0.00153102	-0.63887852	0.32910112	-0.05204415	-0.17483434
weight	0.05029929	0.45030847	0.24713831	0.15342878	-0.22128329	-0.39877367	0.01392053
cups	0.29463556	-0.21224795	0.13999969	0.04748911	0.12081645	0.09946091	0.74856687
rating	-0.43837839	-0.25153893	0.1818424	0.0383162	0.05758421	-0.18614525	0.06344455

분산	3.63360572	3.1480546	1.90934956	1.01947618	0.98935974	0.72206175	0.67151642
분산비(%)	27.95081329	24.21580505	14.6873045	7.84212446	7.61045933	5.55432129	5.16551113
누적분산비(%)	27.95081329	52.16661835	66.85391998	74.69604492	82.3065033	87.86082458	93.02633667

PCA Procedure

- PCA Example
  - ✓ In a 2-dimensional space



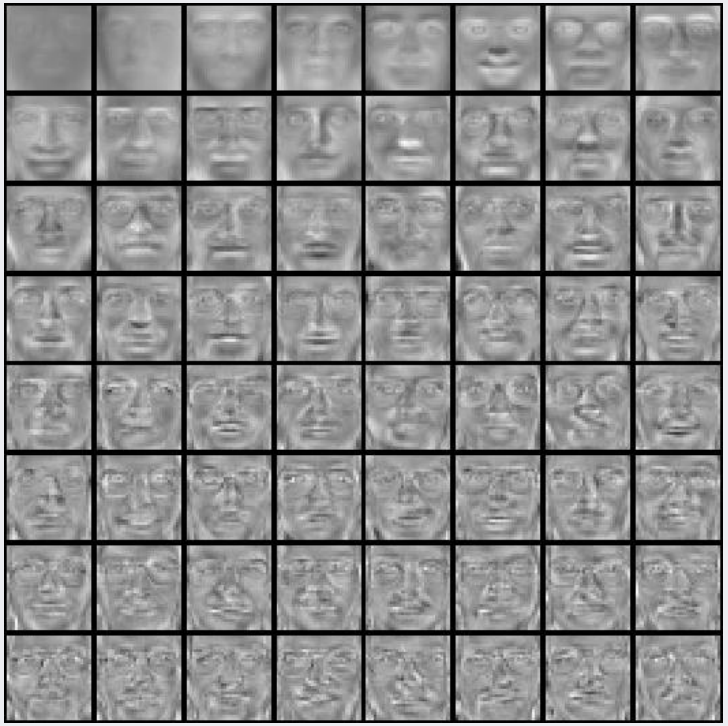
PCA Procedure

- PCA Example
  - ✓ Face image data

Average

Original Image

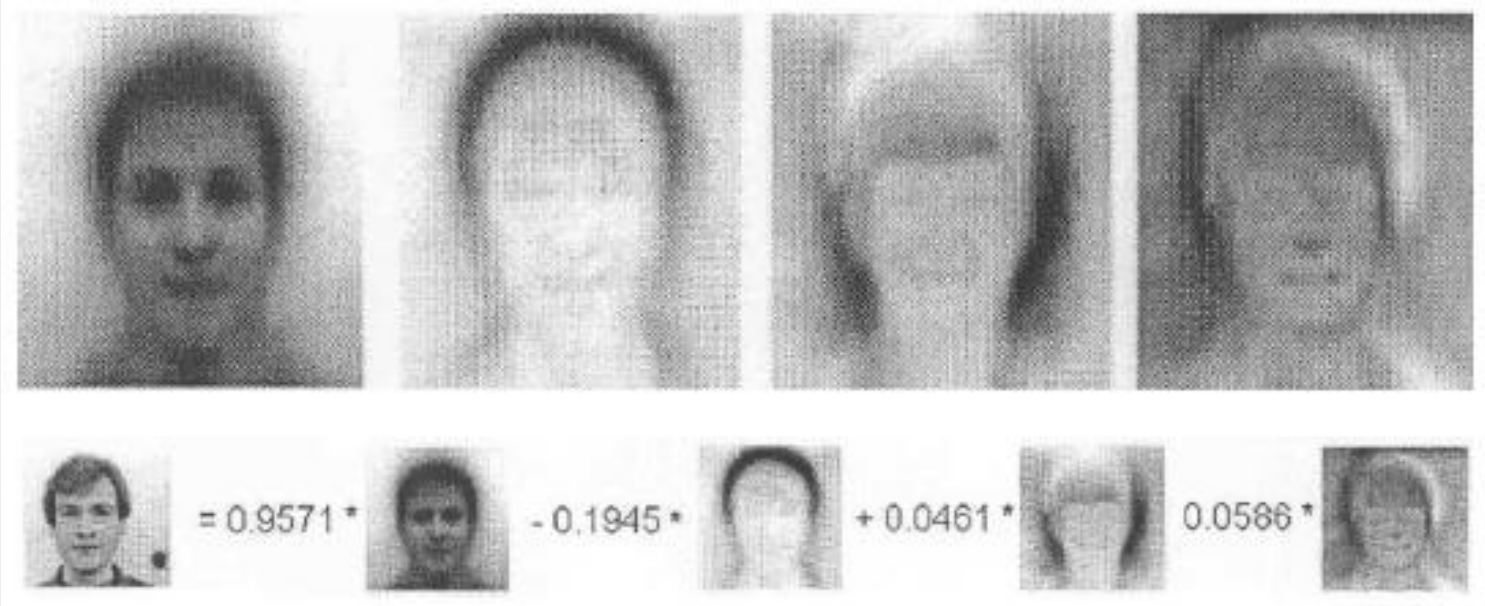
Eigenvectors



PCA Procedure

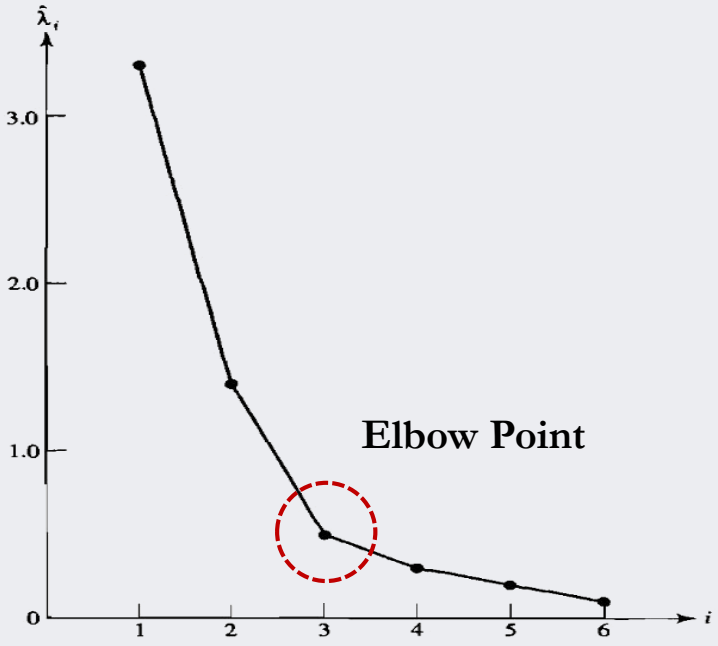
- PCA Example
  - ✓ Face image data

Image reconstruction

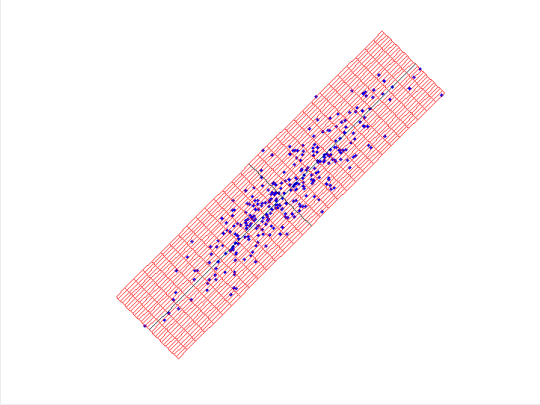


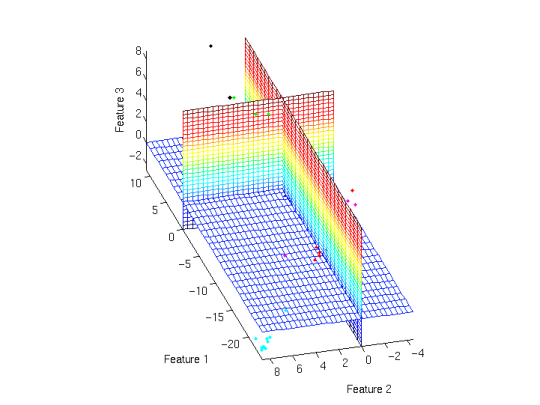
PCA Procedure

- How to determine the optimal number of features
  - ✓ Use the scree plot
  - ✓ In general, select the principal component around the saddle point



PCA Example

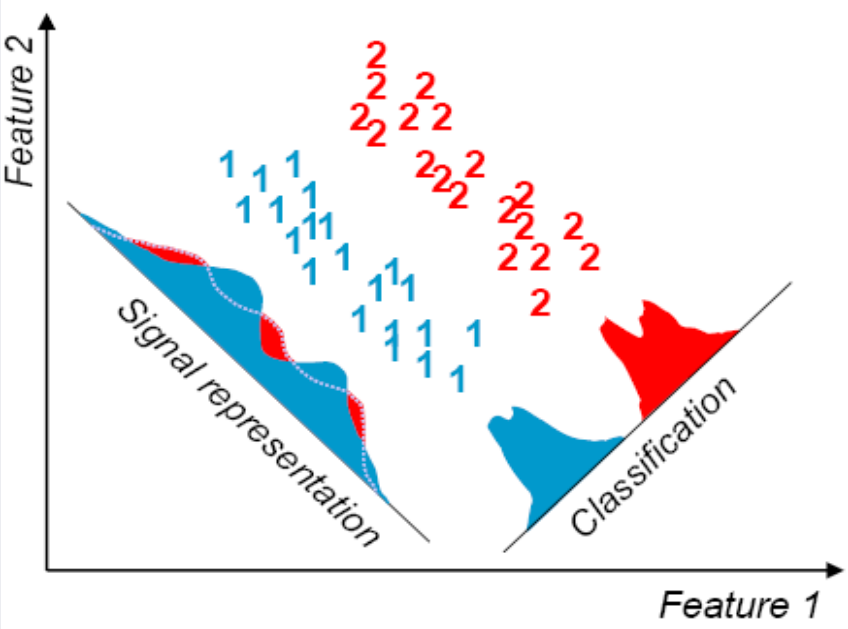




PCA: Limitations

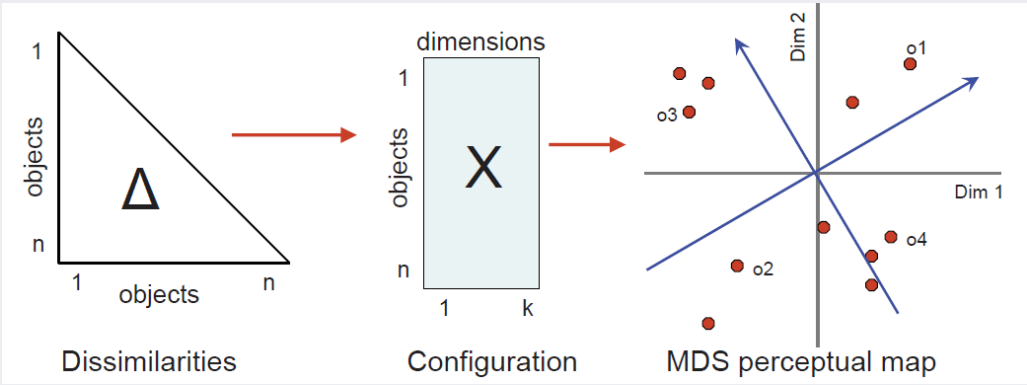
- Cannot work well with non-Gaussian or multimodal Gaussian distributions

- Not designed for classification



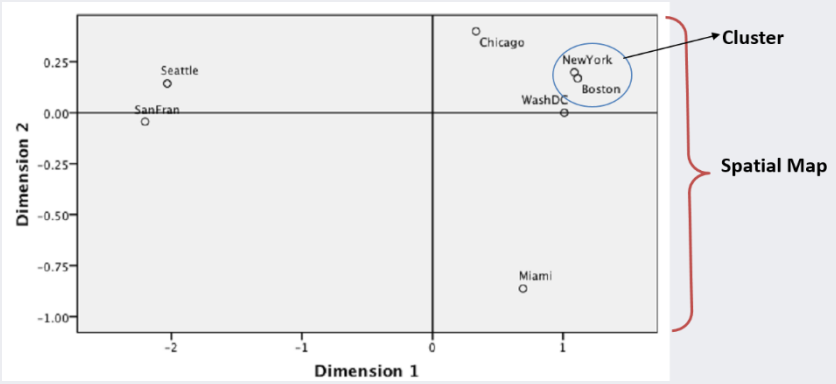
Multidimensional Scaling: MDS

- Multidimensional Scaling
  - ✓ Aims to place each object in D-dimensional space such that the between-object distances are preserved as well as possible



Distances Matrix: Symmetric

		1	2	3	4	5	6	7	8	9
		BOST	NY	DC	MIAM	CHIC	SEAT	SF	LA	DENV
1	BOSTON	0	206	429	1504	963	2976	3095	2979	1949
2	NY	206	0	233	1308	802	2815	2934	2786	1771
3	DC	429	233	0	1075	671	2684	2799	2631	1616
4	MIAMI	1504	1308	1075	0	1329	3273	3053	2687	2037
5	CHICAGO	963	802	671	1329	0	2013	2142	2054	996
6	SEATTLE	2976	2815	2684	3273	2013	0	808	1131	1307
7	SF	3095	2934	2799	3053	2142	808	0	379	1235
8	LA	2979	2786	2631	2687	2054	1131	379	0	1059
9	DENVER	1949	1771	1616	2037	996	1307	1235	1059	0



Multidimensional Scaling: MDS

- PCA vs. MDS

	Principal Component Analysis (PCA)	Multidimensional Scaling (MDS)
Data	n objects in a d-dimensional space ($\mathbf{X}$ in R^d)	Proximity matrix between n objects (n by n matrix $\mathbf{D}$)
Purpose	Find a set of bases to preserve the original variance	Find a set of coordinates that preserve the distance information between objects
Output	1. n bases (eigenvectors, PCs) 2. n eigenvalues	Coordinate of each object in d-dimension ($\mathbf{X}$ in R^d)

MDS Procedure

- Step 1: Construct Proximity/Distance Matrix
  - ✓ If there exist the coordinates for the objects, compute the similarity/distance between them

$$distance-like \text{ if } (1)\ d_{ij} \geq 0, \quad (2)\ d_{ii} = 0, \quad (3)\ d_{ij} = d_{ji}$$

$$metric \text{ if in addition to } (1),\ (2),\ (3),\ \text{it satisfies } d_{ij} \leq d_{ik} + d_{jk}$$

  - ✓ Distance: Euclidean, Manhattan, etc.
  - ✓ Similarity: Correlation, Jaccard, etc.

X (d by n)

	x_1	x_2	x_3	x_4	…	x_n
v_1						
v_2						
v_3						
…						
v_d						

D (n by n)

	x_1	x_2	x_3	x_4	…	x_n
x_1						
x_2						
x_3						
x_4						
…						
x_n						

$$d_{rs}^2 = (\mathbf{x}_r - \mathbf{x}_s)^T(\mathbf{x}_r - \mathbf{x}_s)$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information
  - ✓ Each element of the distance matrix **D** can be expressed as

$$d_{rs}^2 = (\mathbf{x}_r - \mathbf{x}_s)^T(\mathbf{x}_r - \mathbf{x}_s)$$

  - ✓ Inner product matrix **B** can be obtained from the distance matrix **D**

$$[\mathbf{B}]_{rs} = b_{rs} = \mathbf{x}_r^T \mathbf{x}_s$$

    - Assume that the means of all p variables are 0

$$\sum_{r=1}^{n} x_{ri} = 0 \ , \ (i = 1, 2, ..., p) \qquad d_{rs}^2 = \mathbf{x}_r^T \mathbf{x}_r + \mathbf{x}_s^T \mathbf{x}_s - 2\mathbf{x}_r^T \mathbf{x}_s$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information

$$d_{rs}^2 = (\mathbf{x}_r - \mathbf{x}_s)^T(\mathbf{x}_r - \mathbf{x}_s)$$

$$\frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 = \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r + \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_s^T\mathbf{x}_s - \frac{2}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_s = \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r + \mathbf{x}_s^T\mathbf{x}_s$$

$$\mathbf{x}_s^T\mathbf{x}_s = \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r$$

$$\frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 = \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_r^T\mathbf{x}_r + \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T\mathbf{x}_s - \frac{2}{n}\sum_{s=1}^{n} \mathbf{x}_r^T\mathbf{x}_s = \mathbf{x}_r^T\mathbf{x}_r + \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T\mathbf{x}_s$$

$$\mathbf{x}_r^T\mathbf{x}_r = \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T\mathbf{x}_s$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information

$$d_{rs}^2 = (\mathbf{x}_r - \mathbf{x}_s)^T(\mathbf{x}_r - \mathbf{x}_s)$$

$$\frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2 = \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \mathbf{x}_r^T\mathbf{x}_r + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \mathbf{x}_s^T\mathbf{x}_s - \frac{2}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} \mathbf{x}_r^T\mathbf{x}_s$$

$$= \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r + \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T\mathbf{x}_s = \frac{2}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r$$

$$\frac{2}{n}\sum_{r=1}^{n} \mathbf{x}_r^T\mathbf{x}_r = \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information

$$b_{rs} = \mathbf{x}_r^T \mathbf{x}_s \qquad (d_{rs}^2 = \mathbf{x}_r^T \mathbf{x}_r + \mathbf{x}_s^T \mathbf{x}_s - 2\mathbf{x}_r^T \mathbf{x}_s)$$

$$= -\frac{1}{2}(d_{rs}^2 - \mathbf{x}_r^T \mathbf{x}_r - \mathbf{x}_s^T \mathbf{x}_s)$$

$$\mathbf{x}_s^T \mathbf{x}_s = \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T \mathbf{x}_r \quad \mathbf{x}_r^T \mathbf{x}_r = \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T \mathbf{x}_s$$

$$= -\frac{1}{2}\left(d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 + \frac{1}{n}\sum_{s=1}^{n} \mathbf{x}_s^T \mathbf{x}_s - \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 + \frac{1}{n}\sum_{r=1}^{n} \mathbf{x}_r^T \mathbf{x}_r\right)$$

$$= -\frac{1}{2}\left(d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2\right)$$

$$\frac{2}{n}\sum_{r=1}^{n} \mathbf{x}_r^T \mathbf{x}_r = \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2$$

$$= a_{rs} - a_{r\cdot} - a_{\cdot s} + a_{\cdot\cdot}$$

$$\text{where } a_{rs} = -\frac{1}{2}d_{rs}^2,\ a_{r\cdot} = \frac{1}{n}\sum_{s} a_{rs},\ a_{\cdot s} = \frac{1}{n}\sum_{r} a_{rs},\ a_{\cdot\cdot} = \frac{1}{n^2}\sum_{r}\sum_{s} a_{rs}$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information

$$b_{rs} = \mathbf{x}_r^T \mathbf{x}_s = -\frac{1}{2}\left(d_{rs}^2 - \frac{1}{n}\sum_{s=1}^{n} d_{rs}^2 - \frac{1}{n}\sum_{r=1}^{n} d_{rs}^2 + \frac{1}{n^2}\sum_{r=1}^{n}\sum_{s=1}^{n} d_{rs}^2\right)$$

$$= a_{rs} - a_{r\cdot} - a_{\cdot s} + a_{\cdot\cdot}$$

$$\left(\text{where } a_{rs} = -\frac{1}{2}d_{rs}^2,\ a_{r\cdot} = \frac{1}{n}\sum_{r} a_{rs},\ a_{\cdot s} = \frac{1}{n}\sum_{s} a_{rs},\ a_{\cdot\cdot} = \frac{1}{n^2}\sum_{r}\sum_{s} a_{rs}\right)$$

$$[\mathbf{A}]_{rs} = a_{rs} \qquad \mathbf{B} = \mathbf{HAH} \qquad \mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^T$$

MDS Procedure

- Step 2: Extract the coordinates that preserve the distance information
  - ✓ Obtain coordinates of **X** from **B** (**X**: n by p, p < n)

$$\mathbf{B} = \mathbf{X}\mathbf{X}^T \quad rank(\mathbf{B}) = rank(\mathbf{X}\mathbf{X}^T) = rank(\mathbf{X}) = p$$

  - ✓ **B** is symmetric, positive semi-definite and of rank p, so it has p non-negative eigenvalues and (n-p) zero eigenvalues (by Eigen-decomposition)

$$\mathbf{B} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T, \quad \mathbf{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_n), \mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_n]$$

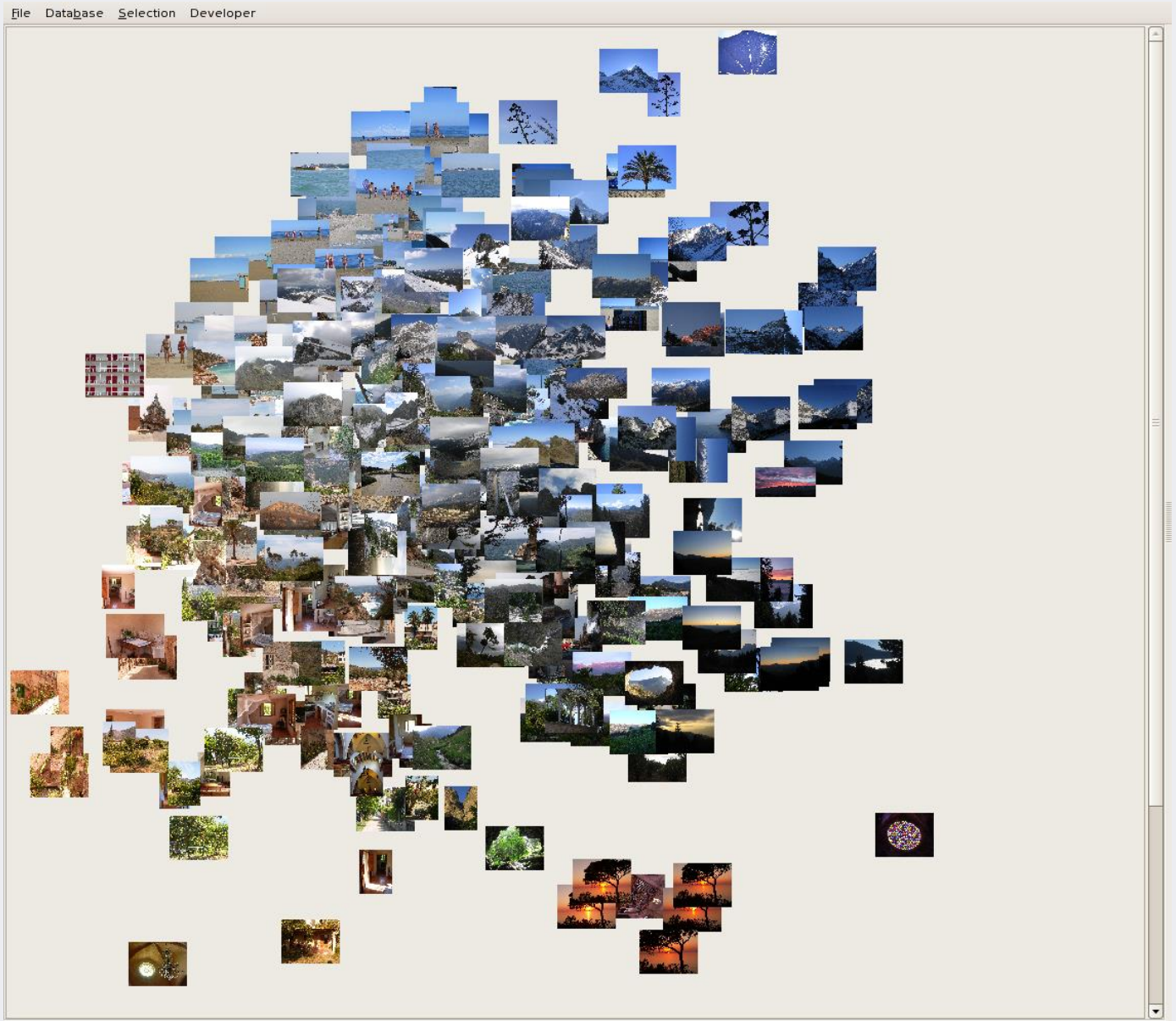
  - ✓ Because of (n-p) zero eigenvalues, **B** can be rewritten as

$$\mathbf{B}_1 = \mathbf{V}_1\mathbf{\Lambda}_1\mathbf{V}_1^T, \quad \mathbf{\Lambda}_1 = diag(\lambda_1, \lambda_2, ..., \lambda_p), \mathbf{V}_1 = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_p]$$

  - ✓ The coordinate matrix **X** is given by

$$\mathbf{X} = \mathbf{V}_1\mathbf{\Lambda}_1^{\frac{1}{2}}$$

MDS Example



AGENDA

01 Dimensionality Reduction: Overview

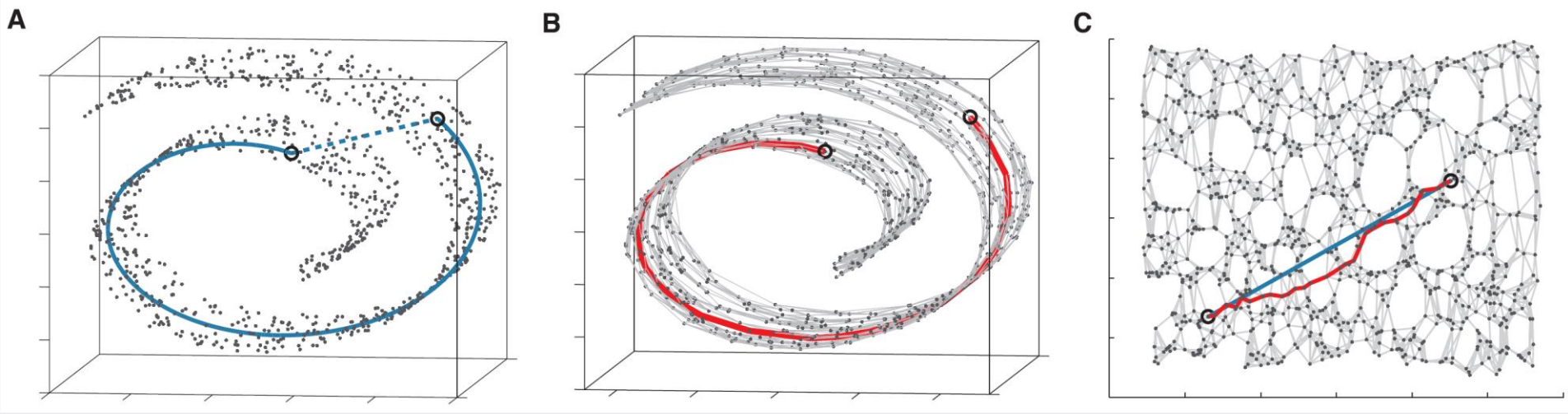
02 Supervised Methods

03 Unsupervised Methods: Linear Mapping

04 Unsupervised Methods: Non-linear Mapping

Isometric Feature Mapping (Isomap) Tenenbaum et al. (2000)

- ISOMAP
  - ✓ Combines the major algorithmic features of PCA and MDS
    - Computational efficiency, global optimality, and asymptotic convergence guarantees
  - ✓ Builds on classical MDS but seeks to preserve the intrinsic geometry of the data, as captured in the geodesic manifold distances between all pairs of data points



Isometric Feature Mapping (Isomap)

- Isomap procedure
  - ✓ Step 1: Construct neighborhood graph
    - ε-Isomap: connect two points if they are closer than ε
    - k-Isomap: connect the point i to the point j if the i is one of the k-nearest neighbor of j
  - ✓ Step 2: Compute the shortest paths
    - Initialize $d_G(i, j) = d_X(i, j)$ if i and j are linked by an edge, $d_G(i, j)$ = inf otherwise
    - For each value of k = 1,2, …, N in turn, replace all entries $d_G(i, j)$ by

    min $\{d_G(i, j), d_G(i, k)+d_G(k, i)$
  - ✓ Step 3: Construct d-dimensional embedding by traditional MDS

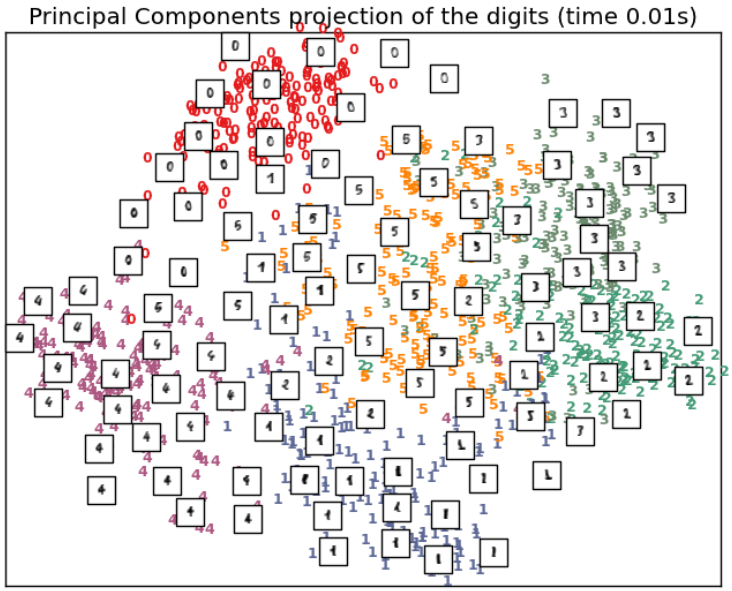
Isometric Feature Mapping (Isomap)

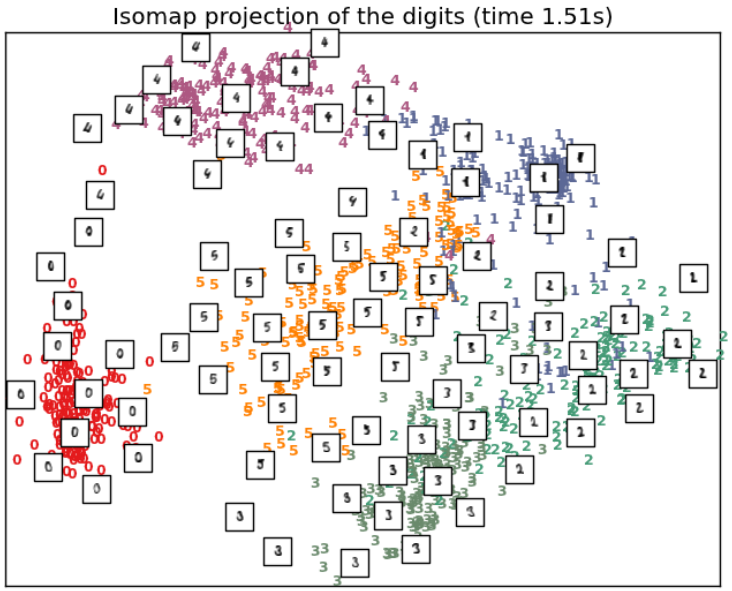
- Isomap example: Hand digit recognition

A selection from the 64-dimensional digits dataset

Isometric Feature Mapping (Isomap)

- Isomap example: Hand digit recognition

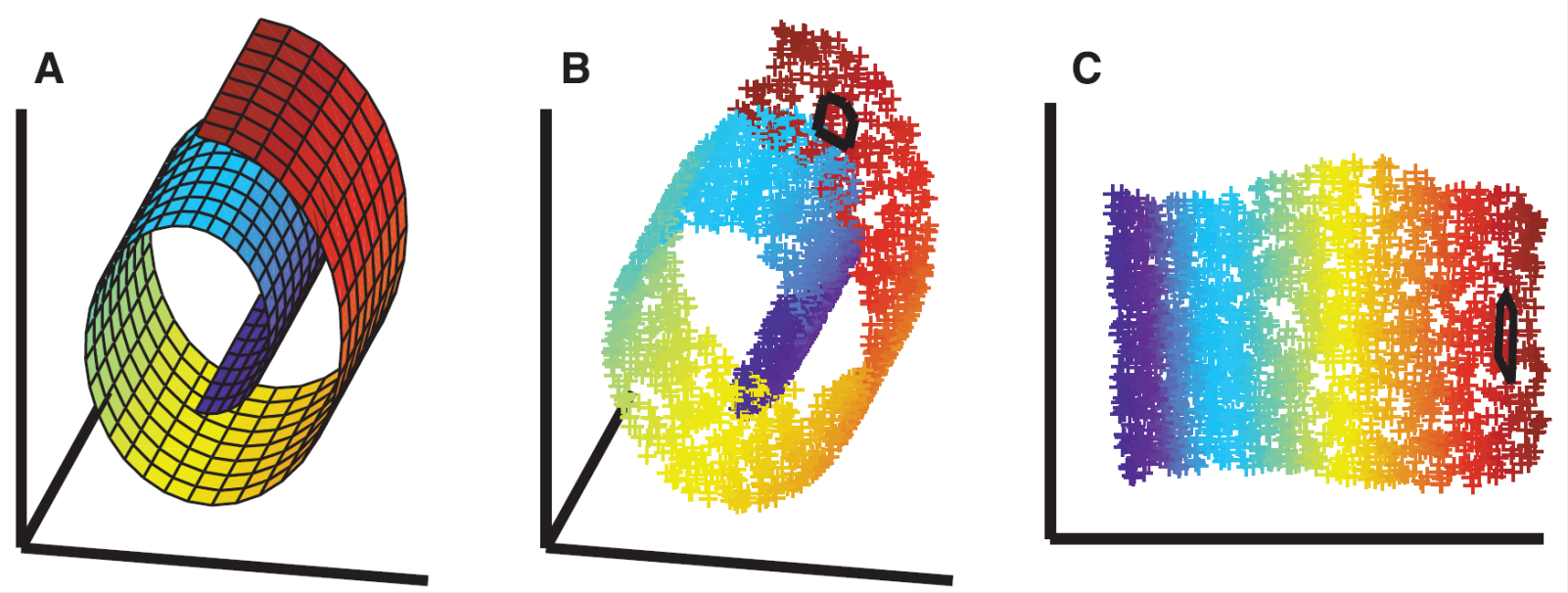




Locally Linear Embedding (LLE)

Roweis and Saul (2000)

- An eigenvector method for nonlinear dimensionality reduction
  - ✓ Simple to implement
  - ✓ Optimizations do not involve local minina
  - ✓ Capable of generating highly nonlinear embeddings
  - ✓ Map high dimensional data into a single global coordinate system of lower dimension



Locally Linear Embedding (LLE)

- LLE Procedure
  - ✓ Step 1: Compute the neighbors of each data point
  - ✓ Step 2: Compute the weight W_{ij} that best reconstruct each data point from its neighbors, minimizing the cost function by constrained linear fits

$$E(\mathbf{W}) = \sum_i \left| \mathbf{X}_i - \sum_j \mathbf{W}_{ij} \mathbf{X}_j \right|^2$$

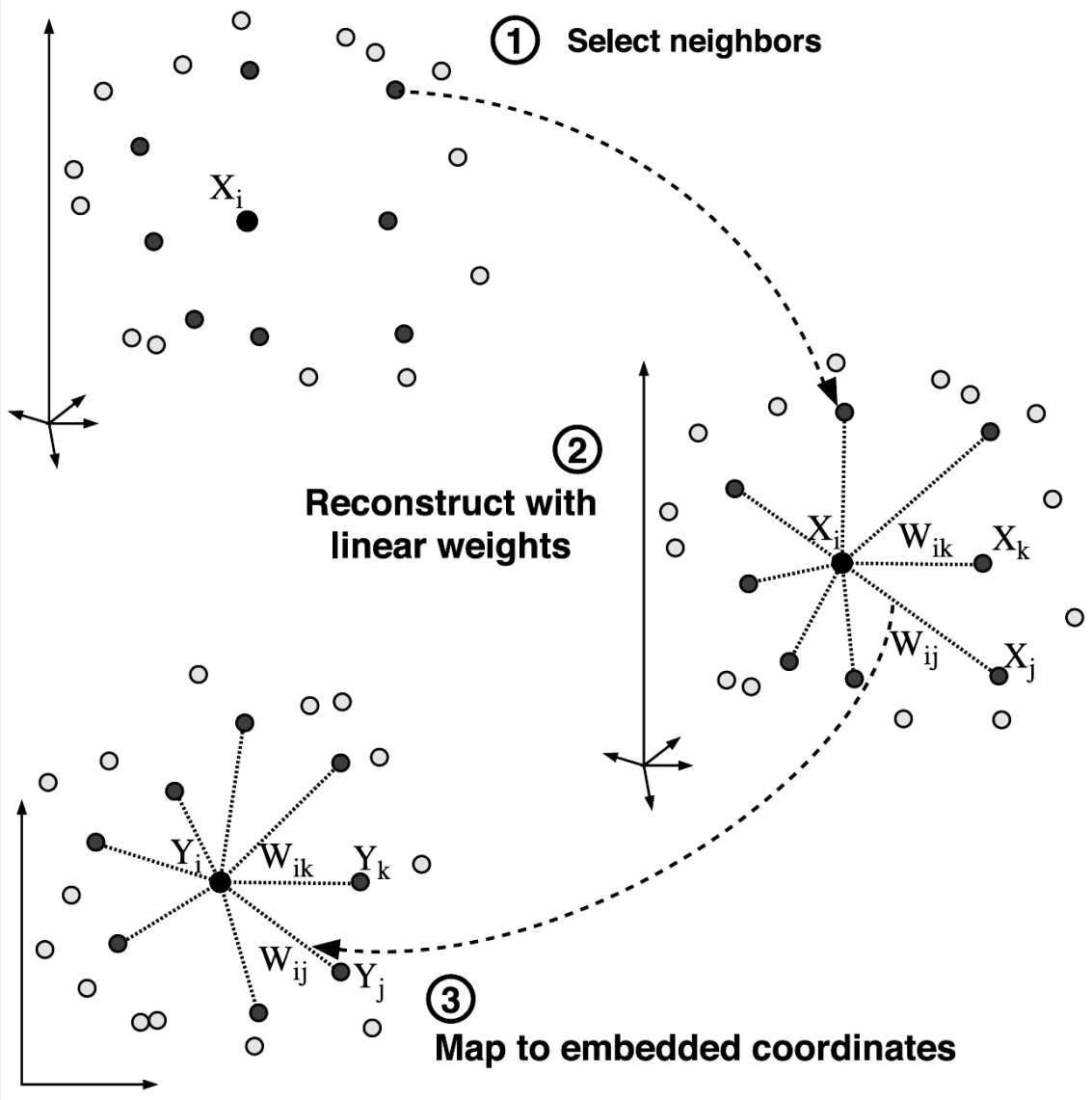
$$s.t.\ \mathbf{W}_{ij} = 0 \text{ if } \mathbf{X}_j \text{does not belong to the neighbor of } \mathbf{X}_i$$

$$\sum_j \mathbf{W}_{ij} = 1 \text{ for all } i$$

  - ✓ Step 3: Compute the vectors best reconstructed by the weights W_{ij}, minimizing the quadratic form by its bottom nonzero eigenvectors

$$\Phi(\mathbf{W}) = \sum_i \left| \mathbf{Y}_i - \sum_j \mathbf{W}_{ij} \mathbf{Y}_j \right|^2$$

Locally Linear Embedding (LLE)



Weights

Invariant to rotation, re-scalings, translations of the data points due to the constraint.

Note

Although the weights W_{ij} and the vectors Y_i are computed by methods in linear algebra, the constraint that the points are only reconstructed from neighbors can result in highly nonlinear embeddings.

Locally Linear Embedding (LLE)

- LLE Procedure (cont')
  - ✓ In Step 2, the reconstruction error of $\varepsilon(\mathbf{W})$ should be zero if $\mathbf{X}_i$ is located in the convex set of its neighbors.
  - ✓ How to find Y_is in Step 3?

$$\min_{\mathbf{Y}} \Phi(\mathbf{W}) = \sum_i \left| \mathbf{Y}_i - \sum_j \mathbf{W}_{ij} \mathbf{Y}_j \right|^2 \Rightarrow \Phi(\mathbf{W}) = \sum_{i,j} \mathbf{M}_{ij} (\mathbf{Y}_i \cdot \mathbf{Y}_j)$$

$$\text{where } \mathbf{M}_{ij} = \delta_{ij} - \mathbf{W}_{ij} - \mathbf{W}_{ji} + \sum_k \mathbf{W}_{ki} \mathbf{W}_{kj}, \ \delta_{ij} = 1 \text{ if } i = j, \ 0 \text{ otherwise}$$

$$s.t. \ \sum_i \mathbf{Y}_i = 0, \ \frac{1}{n} \sum_i \mathbf{Y} \mathbf{Y}^T = \mathbf{I}$$

Locally Linear Embedding (LLE)

- LLE Procedure (cont')
  - ✓ In Step 2, the reconstruction error of $\varepsilon(\mathbf{W})$ should be zero if $\mathbf{X}_i$ is located in the convex set of its neighbors.
  - ✓ How to find Y_is in Step 3?

$$\begin{aligned}
\min_{\mathbf{Y}} \Phi(\mathbf{W}) &= \sum_i \left| \mathbf{Y}_i - \sum_j \mathbf{W}_{ij} \mathbf{Y}_j \right|^2 \\
&= \Big[(\mathbf{I} - \mathbf{W}) \mathbf{Y} \Big]^T (\mathbf{I} - \mathbf{W}) \mathbf{Y} \\
&= \mathbf{Y}^T (\mathbf{I} - \mathbf{W})^T (\mathbf{I} - \mathbf{W}) \mathbf{Y} \\
&= \mathbf{Y}^T \mathbf{M} \mathbf{Y}
\end{aligned}$$

Locally Linear Embedding (LLE)

- LLE Procedure (cont')
  - ✓ The optimal embedding is found by computing the bottom d+1 eigenvectors of the matrix M (Rayleitz-Ritz theorem)
  - ✓ The bottom eigenvector is the unit vector with all equal components
  - ✓ Discarding this eigenvector enforces the constraint that the embeddings have zero mean

The **Rayleigh–Ritz method** allows for the computation of Ritz pairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i)$ which approximate the solutions to the eigenvalue problem [1]

$$A\mathbf{x} = \lambda\mathbf{x}$$

Where $A \in \mathbb{C}^{N\times N}$.

The procedure is as follows:[2]

1. Compute an orthonormal basis $V \in \mathbb{C}^{N\times m}$ approximating the eigenspace corresponding to m eigenvectors
2. Compute $R \leftarrow V^* A V$
3. Compute the eigenvalues of R solving $R\mathbf{v}_i = \tilde{\lambda}_i \mathbf{v}_i$
4. Form the ritz pairs $(\tilde{\lambda}_i, \tilde{\mathbf{x}}_i) = (\tilde{\lambda}_i, V\mathbf{v}_i)$

One can always compute the accuracy of such an approximation via $\|A\tilde{\mathbf{x}}_i - \tilde{\lambda}_i\tilde{\mathbf{x}}_i\|$

If a Krylov subspace is used and A is a general matrix, then this is the Arnoldi Algorithm.

Locally Linear Embedding (LLE)

- LLE Procedure (cont')

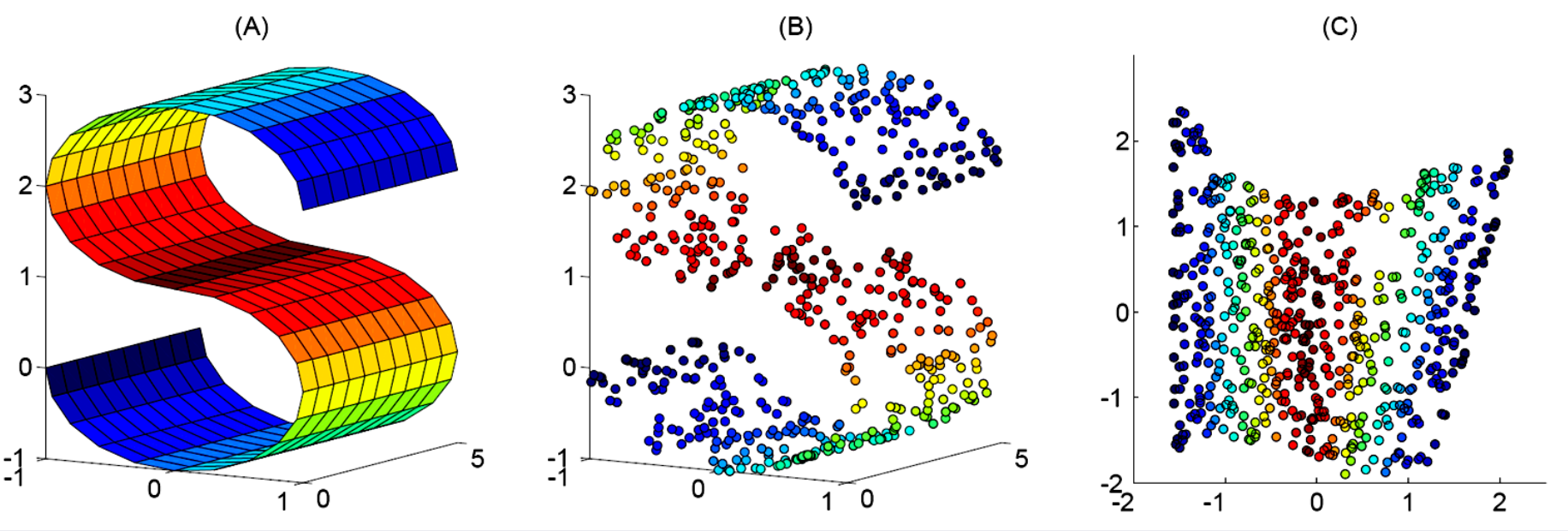
Compute embedding coordinates Y using weights W

- create sparse matrix M = (I-W)'*(I-W)
- find bottom d+1 eigenvectors of M (corresponding to the d+1 smallest eigenvalues)
- set the qth ROW of Y to be the q+1 smallest eigenvector (discard the bottom eigenvector [1,1,1,1...] with eigenvalue zero) …(from LLE Homepage)

Matrix transition during LLE process

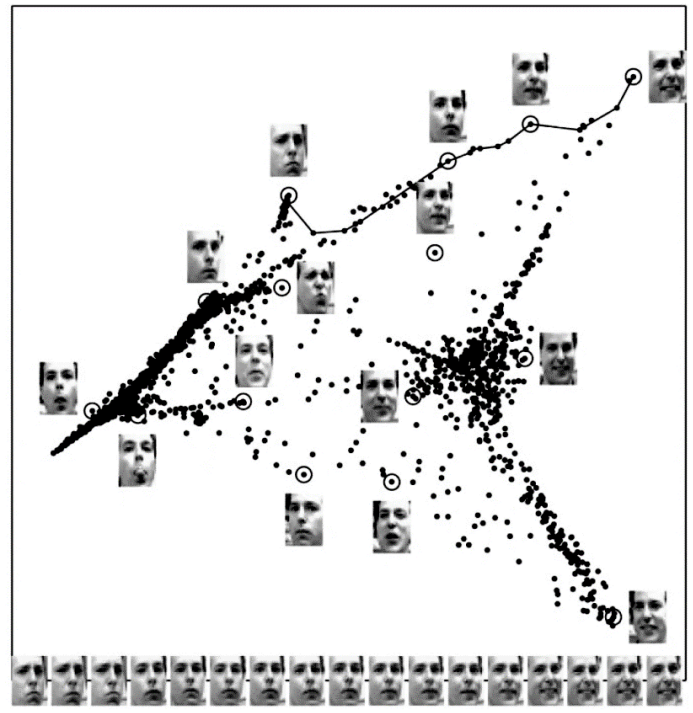
Locally Linear Embedding (LLE)

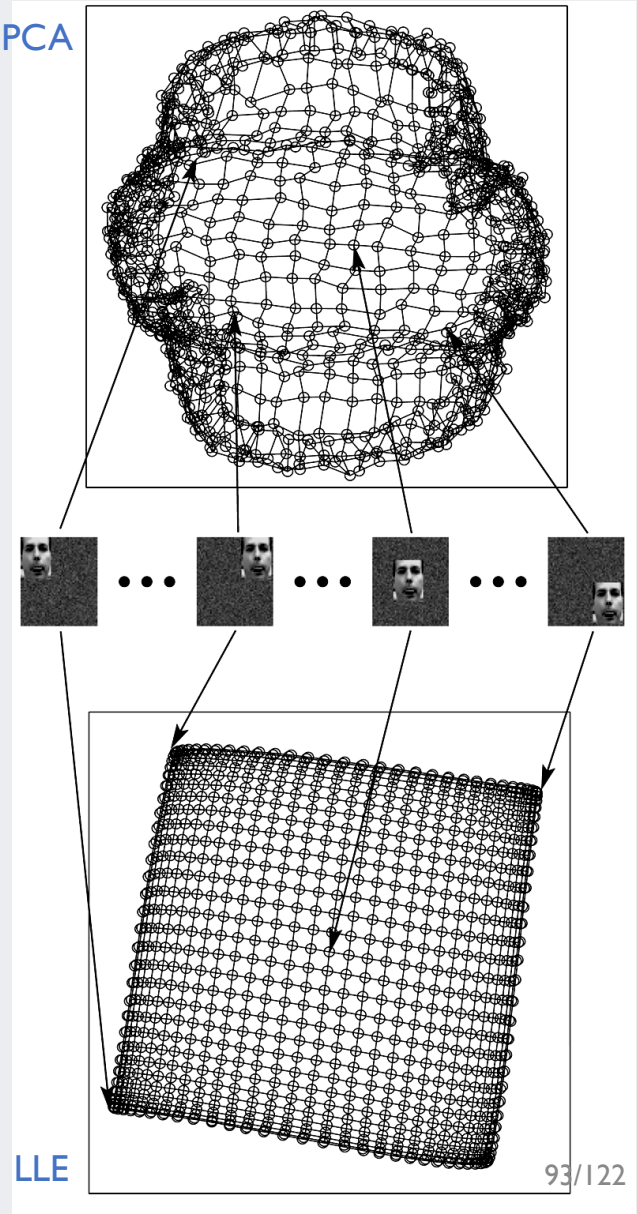
- LLE Example 1
  - ✓ N=600, k=12



Locally Linear Embedding (LLE)

- LLE Example 2
  - ✓ N = 961 grayscale images
  - ✓ Each image containing 28 X 20 face superimposed on 59 X 51 background of noise
  - ✓ D = 3009 K = 4

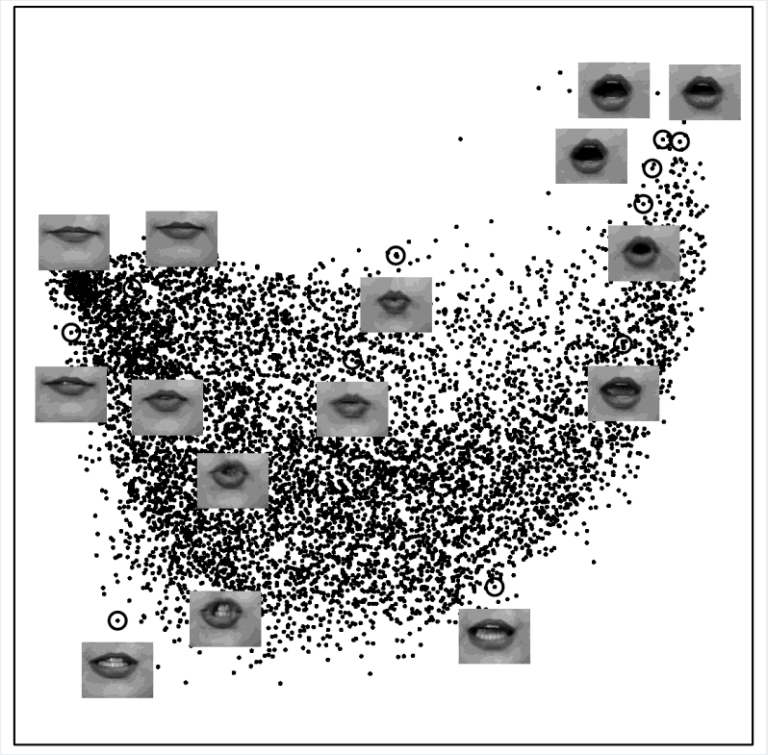




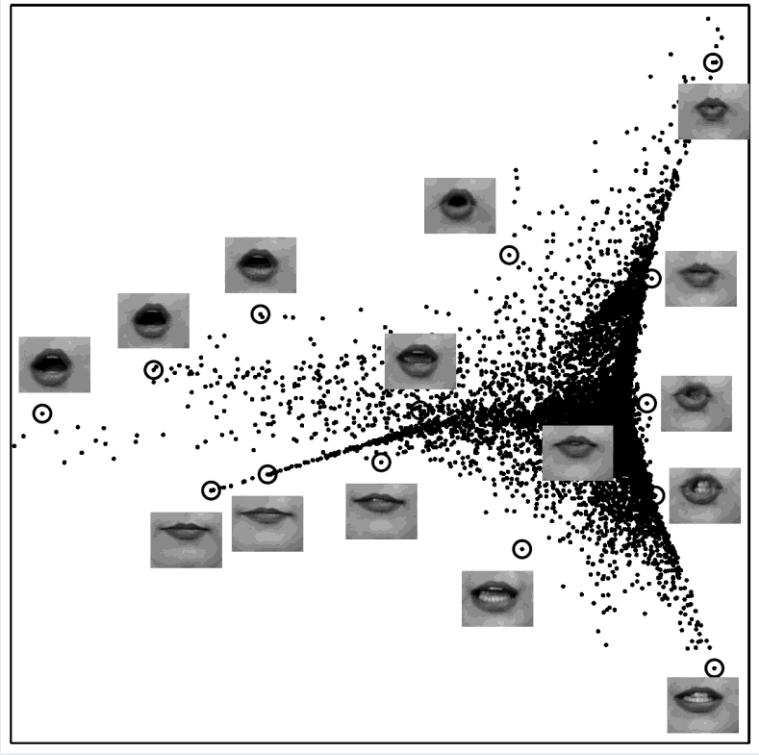
Locally Linear Embedding (LLE)

- LLE Example 3
  - ✓ Images of lips used in the animation talking head
  - ✓ N = 8588 RGB images of lips at 108 X 84 resolution
  - ✓ D = 27,216, K = 16

PCA

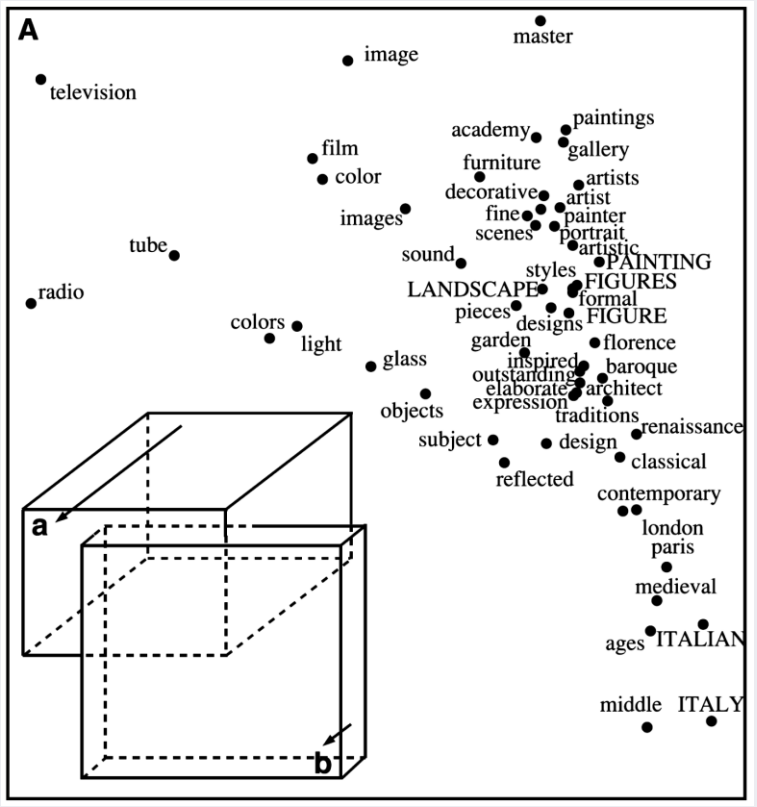


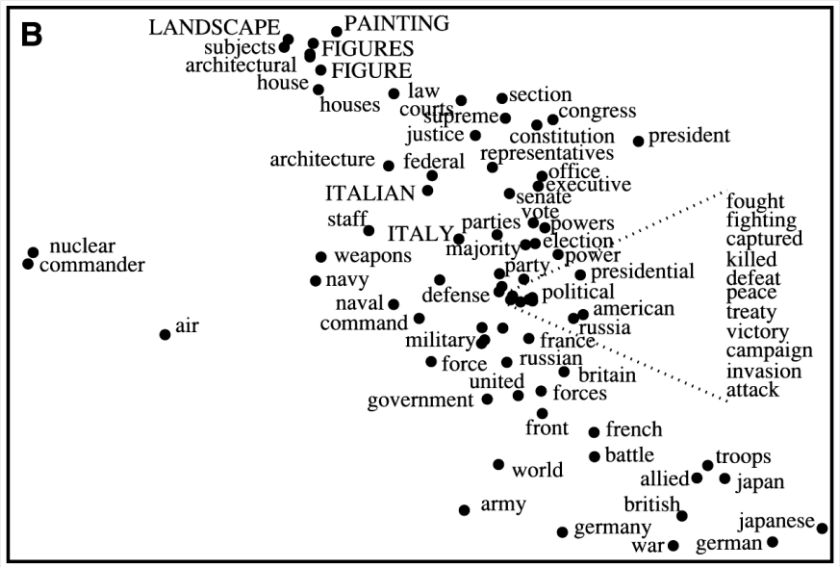
LLE



Locally Linear Embedding (LLE)

- LLE Example 3
  - ✓ Arranging words in a continuous semantic space
  - ✓ Each word was initially represented by a high-dimensional vector that counted the number of times it appeared in different encyclopedia articles





Stochastic Neighbor Embedding

Hinton and Roweis (2002)

- Stochastic Neighbor Embedding (SNE)
  - ✓ It is more important to get local distances right than non-local ones
  - ✓ SNE has a probabilistic way of deciding if a pairwise distance is local
  - ✓ Convert each high-dimensional similarity into the probability that one data point will pick the other data point as its neighbor
    - Probability of picking j given in high D
    - Probability of picking j given in low D

$$p_{j|i} = \frac{e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{-\frac{||\mathbf{x}_i - \mathbf{x}_k||^2}{2\sigma_i^2}}} \qquad q_{j|i} = \frac{e^{-||\mathbf{y}_i - \mathbf{y}_j||^2}}{\sum_{k \neq i} e^{-||\mathbf{y}_i - \mathbf{y}_k||^2}}$$

Stochastic Neighbor Embedding

- Picking the Radius of the Gaussian in p
  - ✓ We need to use different radii in different parts of the space so that we keep the effective number of neighbors about constant
  - ✓ A big radius leads to a high entropy for the distribution over neighbors of i, whereas a small radius leads to a low entropy
  - ✓ Decide what entropy you want and then find the radius that produces that entropy

$$Perplexity(P_i) = 2^{H(P_i)}$$

$$H(P_i) = \sum_j p_{j|i} log_2 p_{j|i}$$

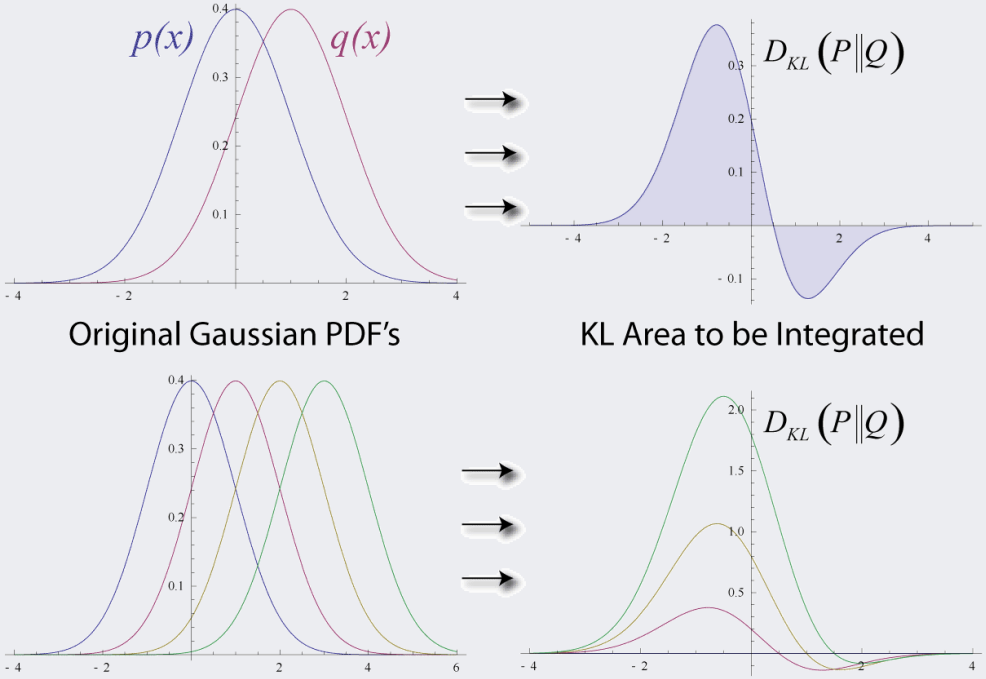
$$p_{j|i} = \frac{e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{-\frac{||\mathbf{x}_i - \mathbf{x}_k||^2}{2\sigma_i^2}}}$$

  - ✓ The performance of SNE is fairly robust to changes in the perplexity (5~50)

Stochastic Neighbor Embedding

- Cost Function for a Low-dimensional Representation
  - ✓ Kullback-Leibler divergence
    - A non-symmetric measure of the difference between two probability distribution P and Q

$$Cost = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$



Stochastic Neighbor Embedding

- Cost Function for a Low-dimensional Representation
  - ✓ Kullback-Leibler divergence
    - A non-symmetric measure of the difference between two probability distribution P and Q

$$Cost = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

  - ✓ Gradient
    - Differencing Cost is tedious because $\mathbf{y}_k$ affect q_{ij} via the normalized term in Eq. (3), but the result is simple Hinton and Roweis (2002)
    - The gradient has a surprisingly simple form Maaten and Hinton (2008)

$$\frac{\partial C}{\partial \mathbf{y}_i} = 2 \sum_j (\mathbf{y}_j - \mathbf{y}_i)(p_{j|i} - q_{j|i} + p_{i|j} - q_{i|j})$$

Stochastic Neighbor Embedding

- Cost Function for a Low-dimensional Representation
  - ✓ Gradient
    - Differencing Cost is tedious because $\mathbf{y}_k$ affect q_{ij} via the normalized term in Eq. (3), but the result is simple Hinton and Roweis (2002)
    - The gradient has a surprisingly simple form Maaten and Hinton (2008)

$$Cost = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

Stochastic Neighbor Embedding

- Gradient of the cost function

$$C = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{j|i} log \frac{p_{j|i}}{q_{j|i}}$$

$$C = \sum_i \sum_j p_{j|i} log p_{j|i} - \sum_i \sum_j p_{j|i} log q_{j|i}$$

$$C' = -\sum_i \sum_j p_{j|i} log q_{j|i} \qquad \left(\frac{\partial C}{\partial y_t} = \frac{\partial C'}{\partial y_t}\right)$$

$$C' = \underbrace{-\sum_i p_{i|t} log q_{i|t}}_{①} \underbrace{-\sum_j p_{t|j} log q_{t|j}}_{②} \underbrace{-\sum_{i \neq t} \sum_{j \neq t} p_{i|j} log q_{i|j}}_{③}$$

Stochastic Neighbor Embedding

- Gradient of the cost function

$$d_{ti} = exp(-||y_t - y_i||^2) = d_{it}$$

$$\frac{\partial d_{ti}}{\partial y_t} = d'_{ti} = -2(y_t - y_i)exp(-||y_t - y_i||^2) = d_{it} = 2(y_t - y_i)d_{ti}$$

$$q_{t|j} = \frac{exp(-||y_j - y_t||^2)}{\sum_{k \neq j} exp(-||y_j - y_k||^2)} = \frac{d_{jt}}{\sum_{k \neq j} d_{jk}}$$

$$q_{i|t} = \frac{exp(-||y_t - y_i||^2)}{\sum_{k \neq t} exp(-||y_t - y_k||^2)} = \frac{d_{ti}}{\sum_{k \neq t} d_{tk}}$$

$$q_{i|j} = \frac{exp(-||y_j - y_i||^2)}{\sum_{k \neq j} exp(-||y_j - y_i||^2)} = \frac{d_{ji}}{\sum_{k \neq j} d_{jk}}$$

Stochastic Neighbor Embedding

- Gradient of the cost function ①

$$\frac{\partial}{\partial y_t}\Big(-\sum_i p_{i|t} log q_{i|t}\Big) = -\sum_i p_{i|t} \cdot \frac{1}{q_{i|t}} \cdot \frac{\partial q_{i|t}}{\partial y_t}$$

$$= -\sum_i p_{i|t} \cdot \frac{1}{q_{i|t}} \cdot \frac{d'_{ti} \cdot (\sum_{k \neq t} d_{tk}) - d_{ti} \cdot (\sum_{k \neq t} d'_{tk})}{(\sum_{k \neq t} d_{tk})^2}$$

$$= -\sum_i p_{i|t} \cdot \frac{1}{q_{i|t}} \cdot \frac{-2(y_t - y_i) \cdot d_{ti} \cdot (\sum_{k \neq t} d_{tk}) - d_{ti} \cdot (\sum_{k \neq t} d'_{tk})}{(\sum_{k \neq t} d_{tk})^2}$$

$$= 2\sum_i p_{i|t} \cdot (y_t - y_i) + 2\sum_i p_{i|t} \cdot \frac{\sum_{k \neq t} d'_{tk}}{\sum_{k \neq t} d_{tk}} \qquad (d'_{tt} = 0, \sum_i p_{i|t} = 1)$$

$$= 2\sum_i p_{i|t} \cdot (y_t - y_i) + \sum_i \cdot \frac{d'_{ti}}{\sum_{k \neq t} d_{tk}}$$

Stochastic Neighbor Embedding

- Gradient of the cost function ①

$$= 2\sum_i p_{i|t} \cdot (y_t - y_i) + \sum_i \cdot \frac{d'_{ti}}{\sum_{k \neq t} d_{tk}}$$

$$= 2\sum_i p_{i|t} \cdot (y_t - y_i) + 2\sum_i (y_t - y_i) \cdot \frac{d_{ti}}{\sum_{k \neq t} d_{tk}}$$

$$= 2\sum_i p_{i|t} \cdot (y_t - y_i) + 2\sum_i (y_t - y_i) \cdot q_{i|t} = 2\sum_i (y_t - y_i)(p_{i|t} - q_{i|t})$$

Stochastic Neighbor Embedding

- Gradient of the cost function ②

$$\frac{\partial}{\partial y_t}\Big(-\sum_j p_{t|j} log q_{t|j}\Big) = -\sum_j p_{t|j} \cdot \frac{1}{q_{t|j}} \cdot \frac{\partial q_{t|j}}{\partial y_t}$$

$$= -\sum_j p_{t|j} \cdot \frac{1}{q_{t|j}} \cdot \frac{d'_{tj} \cdot (\sum_{k \neq j} d_{jk}) - d_{tj} \cdot d'_{jt}}{(\sum_{k \neq j} d_{jk})^2}$$

$$= -\sum_j p_{t|j} \cdot \frac{1}{q_{t|j}} \cdot \frac{-2(y_t - y_j) \cdot d_{jt} \cdot (\sum_{k \neq j} d_{jk}) + 2(y_t - y_j) \cdot d_{jt}^2}{(\sum_{k \neq j} d_{jk})^2}$$

$$= -\sum_j p_{t|j} \cdot \frac{1}{q_{t|j}} \cdot \Big(-2(y_t - y_j) \cdot q_{t|j} + 2(y_t - y_j) \cdot q_{t|j}^2\Big)$$

$$= \sum_j p_{t|j} \cdot 2(y_t - y_j)(1 - q_{t|j})$$

Stochastic Neighbor Embedding

- Gradient of the cost function ③

$$\frac{\partial}{\partial y_t}\Big(-\sum_{i\neq t}\sum_{j\neq t} p_{i|j} log q_{i|j}\Big) = -\sum_{i\neq t}\sum_{j\neq t} p_{i|j}\cdot\frac{1}{q_{i|j}}\cdot\frac{\partial q_{i|j}}{\partial y_t}$$

$$= -\sum_{i\neq t}\sum_{j\neq t} p_{i|j}\cdot\frac{1}{q_{i|j}}\cdot\frac{d'_{ji}\cdot\sum_{k\neq j} d_{jk} - d_{ij}\cdot d'_{jt}}{(\sum_{k\neq j} d_{jk})^2} \qquad (d'_{ji}=0)$$

$$= -\sum_{i\neq t}\sum_{j\neq t} p_{i|j}\cdot\frac{1}{q_{i|j}}\cdot\frac{2(y_t-y_j)\cdot d_{ij}\cdot d_{tj}}{(\sum_{k\neq j} d_{jk})^2}$$

$$= -\sum_{i\neq t}\sum_{j\neq t} p_{i|j}\cdot\frac{1}{q_{i|j}}\cdot 2(y_t-y_j)\cdot q_{i|j}\cdot q_{t|j}$$

$$= -\sum_{i\neq t}\sum_{j\neq t} 2(y_t-y_j)\cdot p_{i|j}\cdot q_{t|j}$$

Stochastic Neighbor Embedding

- Gradient of the cost function ② + ③

$$\sum_j p_{t|j} \cdot 2(y_t - y_j)(1 - q_{t|j}) - \sum_{i \neq t} \sum_{j \neq t} 2(y_t - y_j) \cdot p_{i|j} \cdot q_{t|j}$$

$$= 2 \sum_j (y_t - y_j) \cdot p_{t|j} - 2 \sum_j (y_t - y_j) \cdot p_{t|j} \cdot q_{t|j} - 2 \sum_{i \neq t} \sum_{j \neq t} (y_t - y_j) \cdot p_{i|j} \cdot q_{t|j}$$

$$= 2 \sum_j (y_t - y_j) \cdot p_{t|j} - 2 \sum_i \sum_j (y_t - y_j) \cdot p_{i|j} \cdot q_{t|j}$$

$$= 2 \sum_j (y_t - y_j) \cdot p_{t|j} - 2 \sum_j \sum_i p_{i|j} \cdot (y_t - y_j) \cdot q_{t|j} \qquad (\sum_i p_{i|j} = 1)$$

$$= 2 \sum_j (y_t - y_j) \cdot p_{t|j} - 2 \sum_j (y_t - y_j) \cdot q_{t|j} = 2 \sum_j (y_t - y_j)(p_{t|j} - q_{t|j})$$

Stochastic Neighbor Embedding

- Gradient of the cost function ① + ② + ③

$$2\sum_i (y_t - y_i)(p_{i|t} - q_{i|t}) + 2\sum_j (y_t - y_j)(p_{t|j} - q_{t|j})$$

$$= 2\sum_j (y_t - y_j)(p_{j|t} - q_{j|t} + p_{t|j} - q_{t|j})$$

- Update the coordinate in the lower dimension to minimize the cost function
  - ✓ Gradient update with a momentum term

$$\mathcal{Y}^{(t+1)} = \mathcal{Y}^{(t)} + \eta \frac{\partial C}{\partial \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t)} - \mathcal{Y}^{(t-1)}\right)$$

Symmetric SNE

- Turning conditional probabilities into pairwise probabilities

$$p_{ij} = \frac{e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2}}}{\sum_{k \neq l} e^{-\frac{||\mathbf{x}_k - \mathbf{x}_l||^2}{2\sigma_i^2}}} \quad \blacktriangleright \quad p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n} \qquad \sum_j p_{ij} > \frac{1}{2n}$$

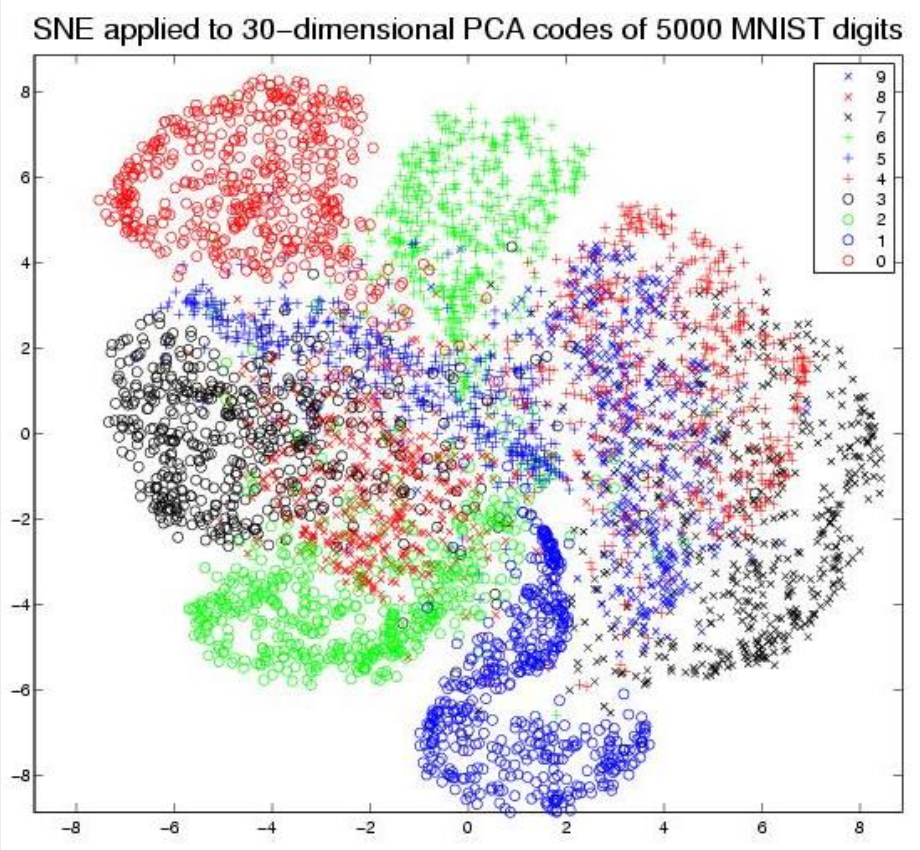
✓ Cost function and gradient

$$Cost = \sum_i KL(P_i||Q_i) = \sum_i \sum_j p_{ij} log \frac{p_{ij}}{q_{ij}}$$

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_j (\mathbf{y}_j - \mathbf{y}_i)(p_{ij} - q_{ij})$$

Symmetric SNE

- Crowding problem
  - ✓ The area accommodating moderately distant data points is not large enough compared with the area accommodating nearby data points



t-SNE

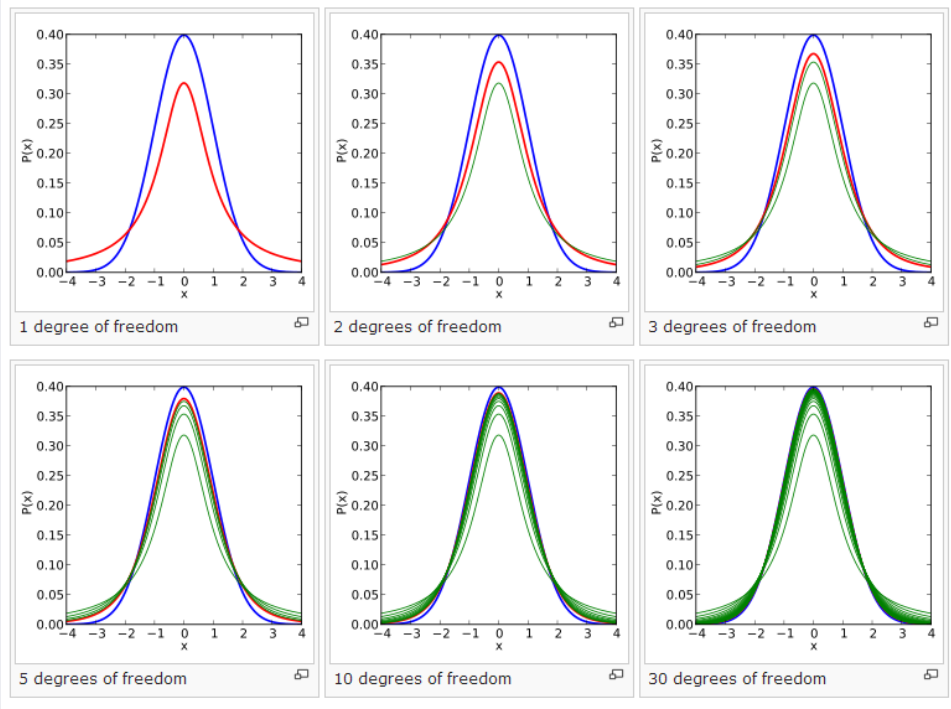
Maaten and Hinton (2008)

- Resolution to the Crowding Problem
  - ✓ Use a probability distribution that has much heavier tails than a Gaussian to convert distances into probabilities in the low-dimensional map
  - ✓ Student's t-distribution with one degree of freedom

$$f(t) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)}\left(1+\frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}$$

$$\Gamma(n) = (n-1)!$$

$$q_{j|i} = \frac{(1+||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k\neq l}(1+||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$



t-SNE

- Optimization of t-SNE

$$p_{j|i} = \frac{e^{-\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\sigma_i^2}}}{\sum_k e^{-\frac{||\mathbf{x}_i - \mathbf{x}_k||^2}{2\sigma_i^2}}} \qquad q_{j|i} = \frac{(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}}{\sum_{k \neq l}(1 + ||\mathbf{y}_k - \mathbf{y}_l||^2)^{-1}}$$

✓ Gradient:

$$\frac{\partial C}{\partial \mathbf{y}_i} = 4 \sum_j (\mathbf{y}_j - \mathbf{y}_i)(p_{ij} - q_{ij})(1 + ||\mathbf{y}_i - \mathbf{y}_j||^2)^{-1}$$

t-SNE

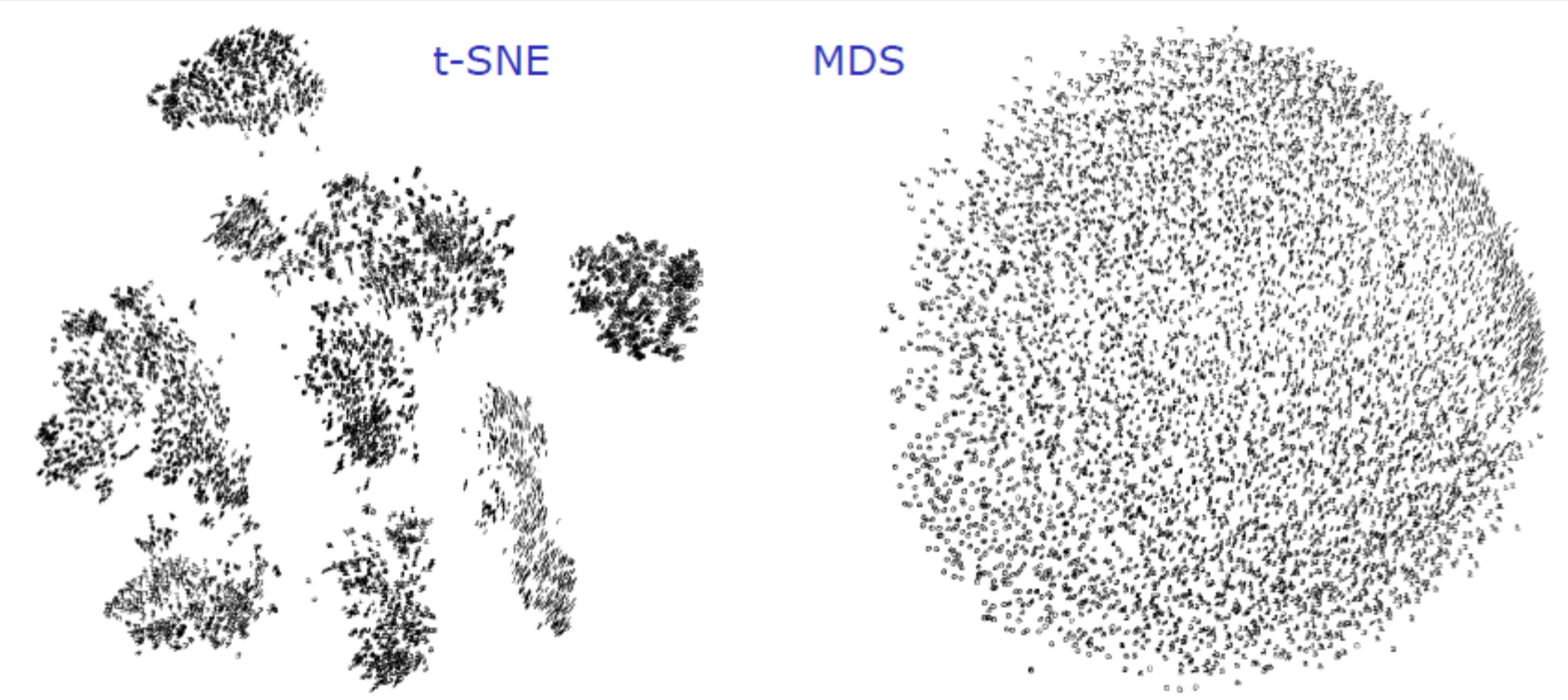
- t-SNE algorithm

Algorithm 1: Simple version of t-Distributed Stochastic Neighbor Embedding.

Data: data set $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$,
cost function parameters: perplexity *Perp*,
optimization parameters: number of iterations T, learning rate η, momentum $\alpha(t)$.
Result: low-dimensional data representation $\mathcal{Y}^{(T)} = \{y_1, y_2, \ldots, y_n\}$.
begin
  compute pairwise affinities $p_{j|i}$ with perplexity *Perp* (using Equation 1)
  set $p_{ij} = \frac{p_{j|i} + p_{i|j}}{2n}$
  sample initial solution $\mathcal{Y}^{(0)} = \{y_1, y_2, \ldots, y_n\}$ from $\mathcal{N}(0, 10^{-4}I)$
  for *t=1* **to** T **do**
    compute low-dimensional affinities q_{ij} (using Equation 4)
    compute gradient $\frac{\delta C}{\delta \mathcal{Y}}$ (using Equation 5)
    set $\mathcal{Y}^{(t)} = \mathcal{Y}^{(t-1)} + \eta \frac{\delta C}{\delta \mathcal{Y}} + \alpha(t)\left(\mathcal{Y}^{(t-1)} - \mathcal{Y}^{(t-2)}\right)$
  end
end

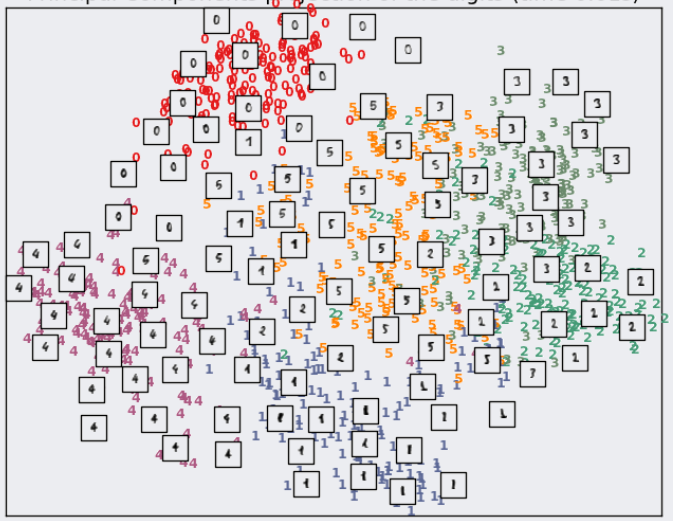
t-SNE vs. MDS

- MNIST dataset

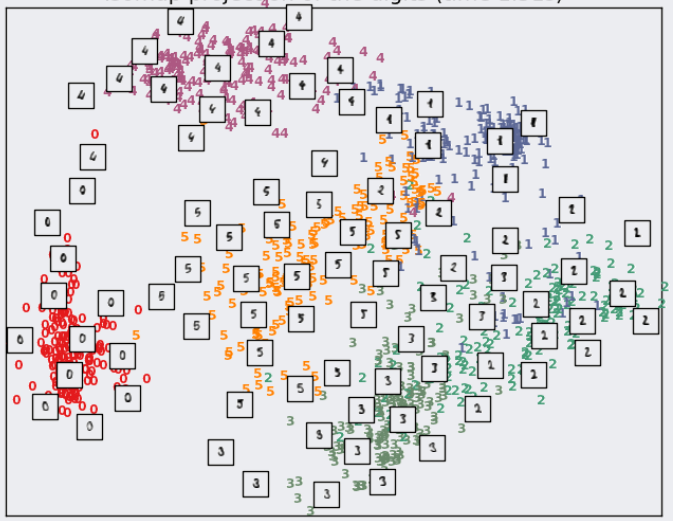


t-SNE Examples

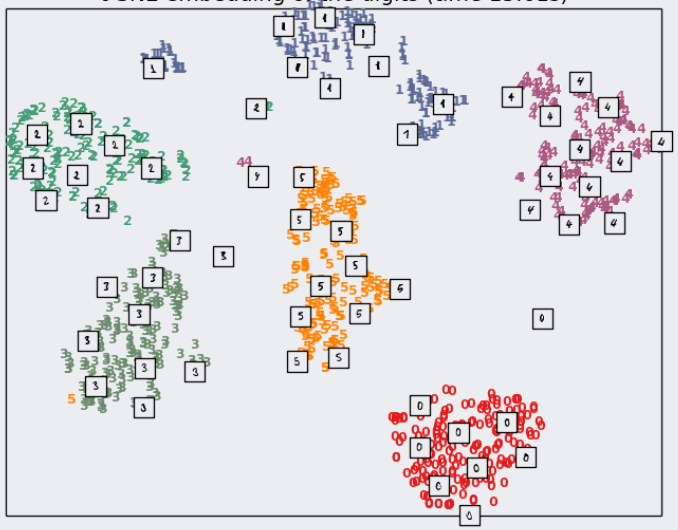
Principal Components projection of the digits (time 0.01s)



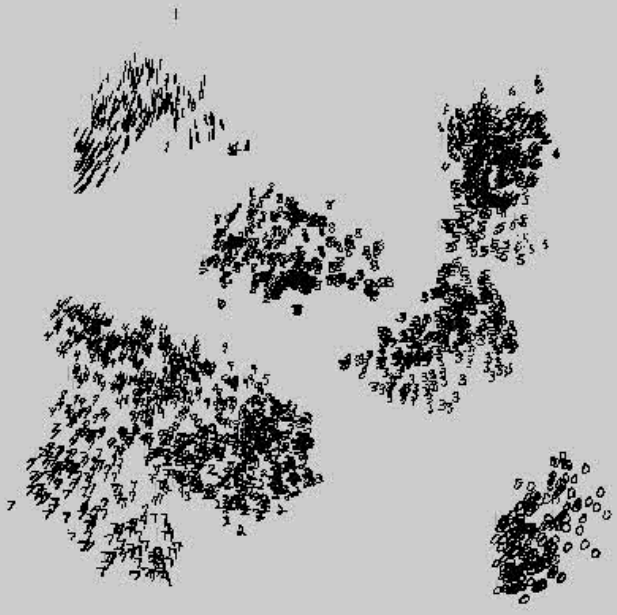
Isomap projection of the digits (time 1.51s)



t-SNE embedding of the digits (time 15.61s)

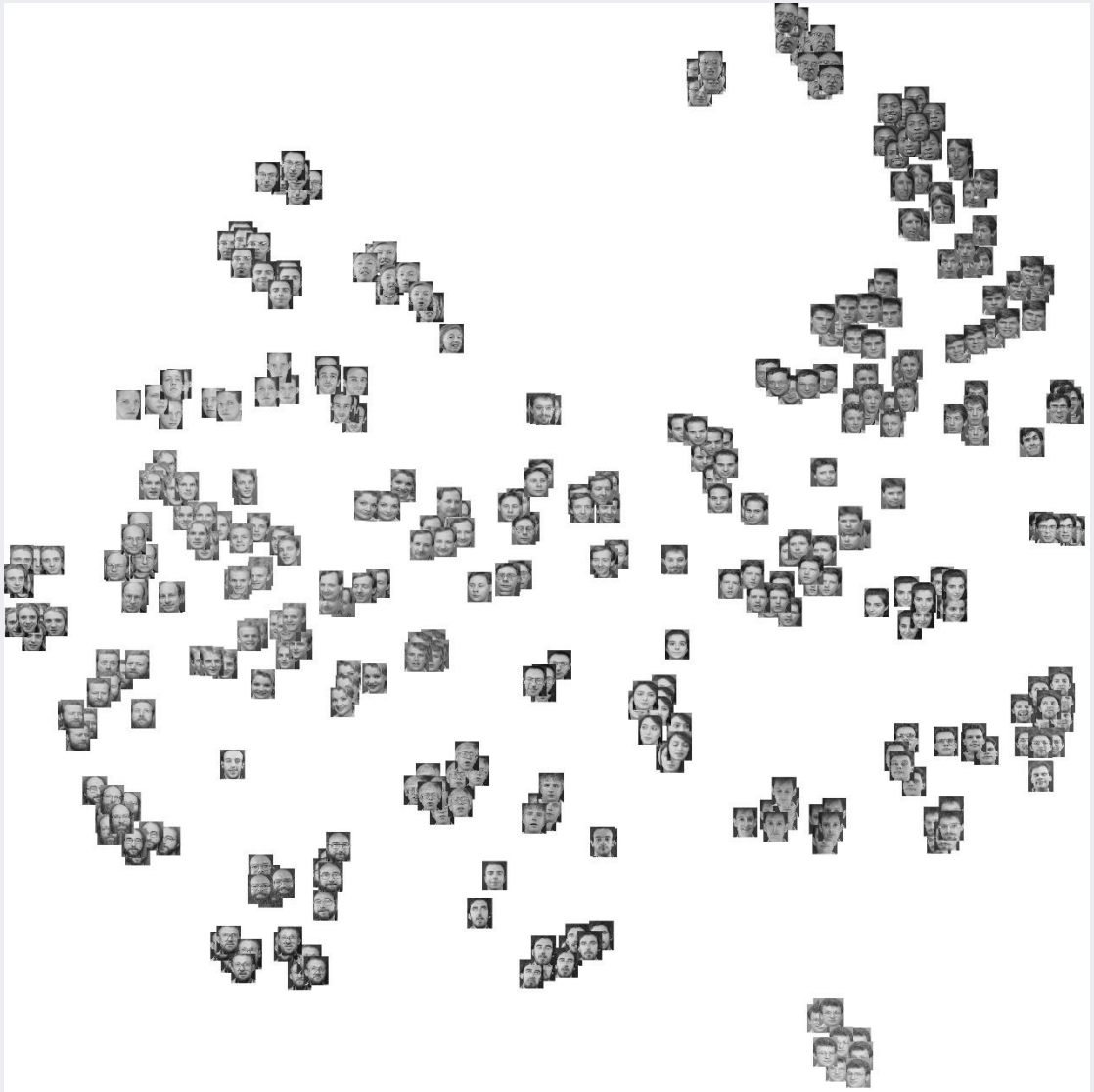


t-SNE Examples



t-SNE Examples

- Olivetti faces datasets



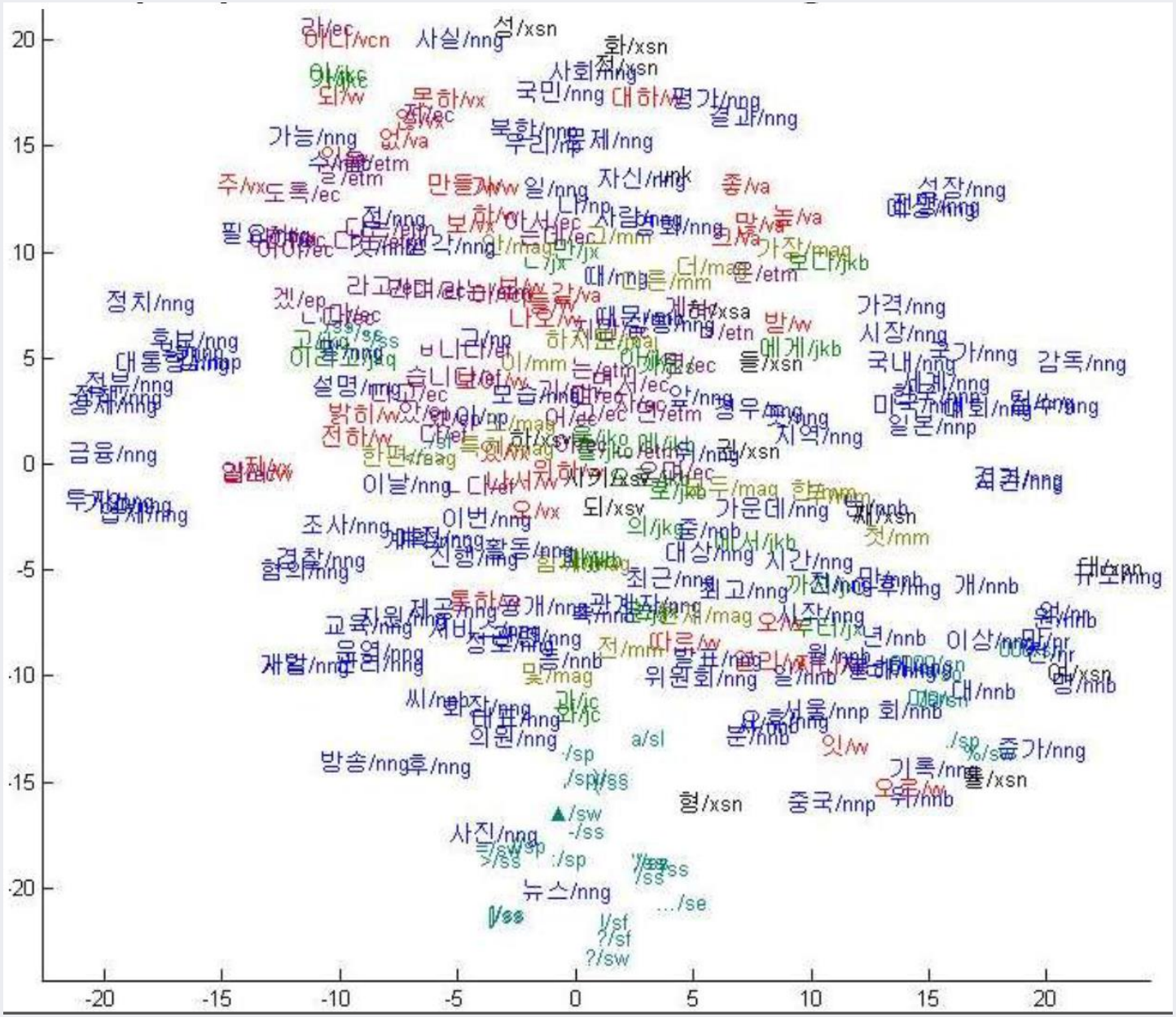
t-SNE Examples

- Netflix dataset

t-SNE Examples

- CalTech-101

t-SNE Examples




Q & A

References

Research Papers

- Bengio, Y., Courville, A., Vincent, P. (2013). Representation learning: A review and new perspectives, IEEE Transactions on Pattern Analysis and Machine Intelligence 35(8): 1798-1828.
- Hinton, G.E. (2007). Learning multiple layers of representation, TRENDS in Cognitive Sciences 11(10): 428-434
- Hinton, G., & Roweis, S. (2002, January). Stochastic neighbor embedding. In NIPS (Vol. 15, pp. 833-840).
- Hinton, G.E., Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks, SCIENCE 313:504-507.
- Kohavi, R. and John, G.H. (1997). Wrapper for feature subset selection. Artificial Intelligence 97(1-2): 273-324.
- LeCun, Y., Bengio, Y., and Hinton, G.E. (2015). Deep learning, NATURE 521: 436-444.
- Roweis, S. and Saul, L. (2000). Nonlinear dimensionality reduction by locally linear embedding. SCIENCE 290:2323-2326.
- Siedlecki, W. Sklansky, J. (1989). A note on genetic algorithms for large-scale feature selection. Pattern Recognition Letters 10(5): 335-347
- Tenenbaum, J.B., de Silva, V., and Langford, J.C. (2000). A global geometric framework for nonlinear dimensionality reduction. SCIENCE 290:2319-2323
- van der Maaten, L.J.P. and Hinton, G.E. (2008). Visualizing high-dimensional data using t-SNE. Journal of Machine Learning Research 9: 2579-2605.

Other materials

- Figure in the title page: https://wattsupwiththat.com/2015/12/12/is-climate-forecasting-immune-from-occams-razor/