https://bit.ly/2I3JKMY

KYOTO UNIVERSITY

Statistical Learning Theory
- Introduction -

Hisashi Kashima / Makoto Yamada

DEPARTMENT OF INTELLIGENCE SCIENCE AND TECHNOLOGY

Statistical learning theory: Foundations of recent data analysis technologies

- This course will cover:
  - Basic ideas, problem, solutions, and applications of statistical machine learning
    - Supervised & unsupervised learning
    - Models & algorithms: linear regression, SVM, perceptron, …
  - Statistical learning theory
    - Probably approximately correct (PAC) learning
- Advanced topics:
  - Online learning, structured prediction, sparse modeling, …

Evaluations:
Report based on data analysis & final exam

- Evaluations will be based on:

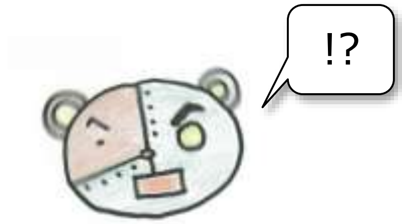
1. Report submission

2. Final exam

Introduction:
Basic ideas of machine learning and applications

1. What is machine learning?
2. Machine learning applications
3. Some machine learning topics
    1. Recommender systems
    2. Anomaly detection

What is machine learning?



"The third A.I. boom": Machine learning is a core technology

- Many successes of "Artificial Intelligence":
  - Q.A. machine beating quiz champions
  - Go program surpassing top players
  - Machine vision is better at recognizing objects than humans
- Current A.I. boom owes machine learning
  - Especially, deep learning

What is machine learning? : A branch of artificial intelligence

- Originally started as a branch of artificial intelligence
  - has its more-than-50-years history
  - Computer programs that “learns” from experience
  - Based on logical inference

What is machine learning? ：A data analytics technology

- Recently considered as a data analysis technology
- Rise of “statistical” machine learning
  – Successes in bioinformatics, natural language processing, and other business areas
  – Victory of IBM’s Watson QA system, Google’s Alpha Go
- “Big data” and “Data scientist”
  – Data scientist is “the sexiest job in the 21st century”
- Success of deep learning
  – The 3rd AI boom

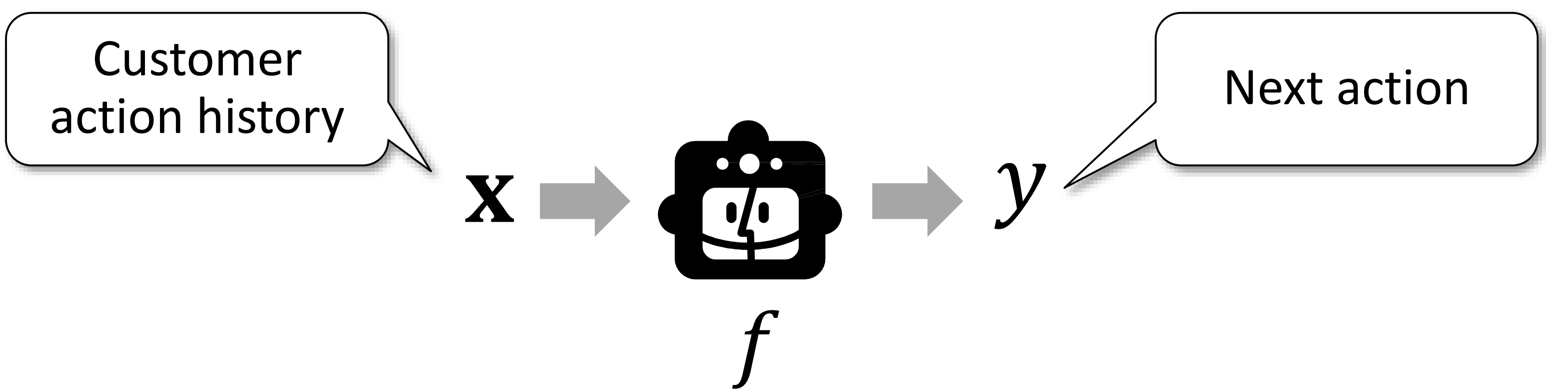
What can machine learning do?: Prediction and discovery

- Two categories of the use of machine learning:

1. Prediction (supervised learning)
   - "What will happen in future data?"
   - Given past data, predict about future data
2. Discovery (unsupervised learning)
   - "What is happening in data in hand?"
   - Given past data, find insights in them

Prediction machine:
A function from a vector to a scalar

- We model the intelligent machine as a mathematical function
- Relationship of input and output $f: \mathbf{x} \rightarrow y$
  - Input $\mathbf{x} = (x_1, x_2, \dots, x_D)^\top \in \mathbb{R}^D$ is a D-dimensional vector
  - Output y is one dimensional
    - Regression: real-valued output $y \in \mathbb{R}$
    - Classification: discrete output $y \in \{C_1, C_2, \dots, C_M\}$

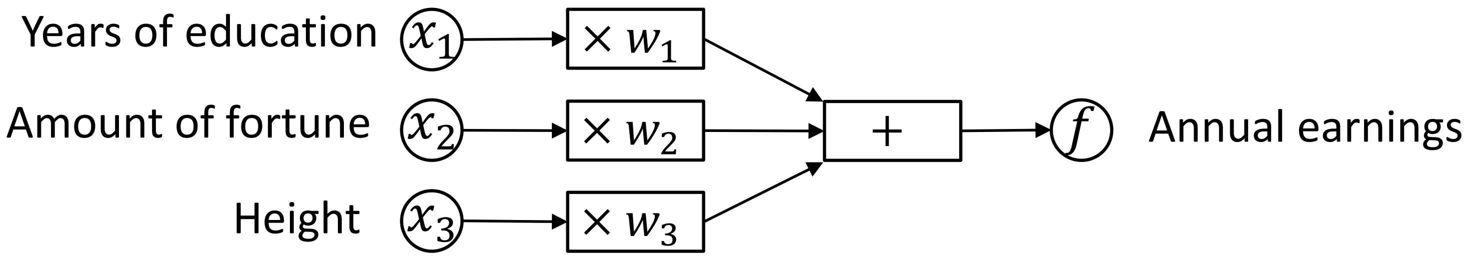


A model for regression:
Linear regression model

- Model f takes an input $\mathbf{x} = (x_1, x_2, \ldots, x_D)^\top$ and outputs a real value

$$\boxed{f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_D x_D}$$

  – Model parameter $\mathbf{w} = (w_1, w_2, \ldots, w_D)^\top \in \mathbb{R}^D$



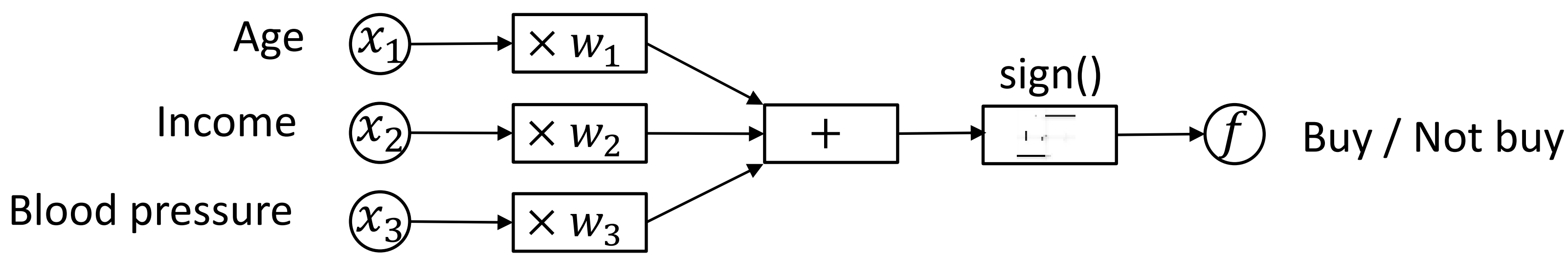
A model for classification:
Linear classification model

- Model f takes an input $\mathbf{x} = (x_1, x_2, \dots, x_D)^\top$ and outputs a value from $\{+1, -1\}$

$$f(\mathbf{x}) = \mathrm{sign}(w_1 x_1 + w_2 x_2 + \cdots + w_D x_D)$$

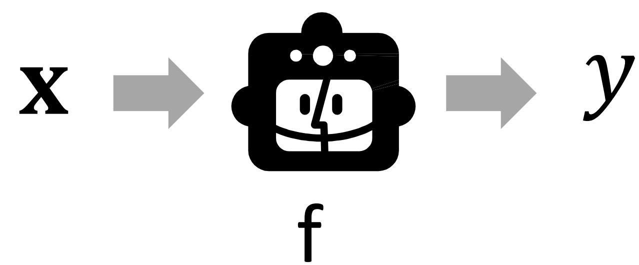
  – Model parameter $\mathbf{w} = (w_1, w_2, \dots, w_D)^\top \in \mathbb{R}^D$:
    - w_d : contribution of x_d to the output

      ($x_d > 0$ contributes to $+1$, $x_d < 0$ contributes to -1)

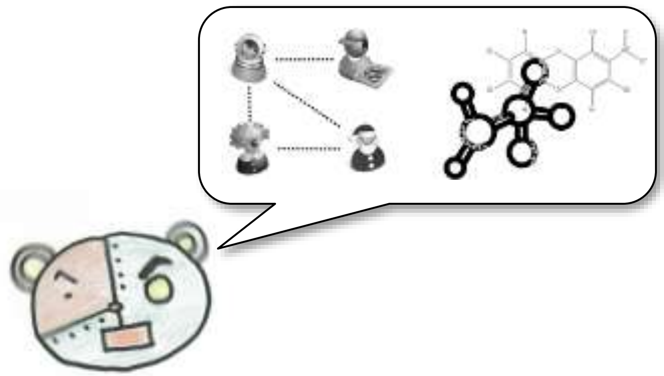


Formulations of machine learning problems: Supervised learning and unsupervised learning

- What we want is the function f
  - We estimate f from data
- Two learning problem settings: supervised and unsupervised
  - Supervised learning: input-output pairs are given
    - $\left\{\left(\mathbf{x}^{(1)}, y^{(1)}\right), \left(\mathbf{x}^{(2)}, y^{(2)}\right), \ldots, \left(\mathbf{x}^{(N)}, y^{(N)}\right)\right\}$: N pairs
  - Unsupervised learning: only inputs are given
    - $\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\right\}$: N inputs



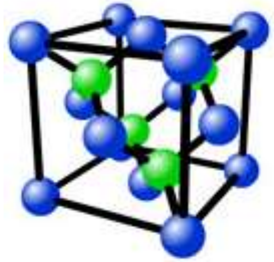
Machine learning applications



Growing ML applications: Emerging applications from IT areas to non-IT areas

- Recent advances in ML:
  - Methodologies to handle uncertain and enormous data
  - Black-box tools
- Not limited to IT areas, ML is wide-spreading over non-IT areas
  - Healthcare, airline, automobile, material science, education, …




Various applications of machine learning:
From on-line shopping to system monitoring

- Marketing
  - Recommendation
  - Sentiment analysis
  - Web ads optimization

- Finance
  - Credit risk estimation
  - Fraud detection

- Science
  - Biology
  - Material science

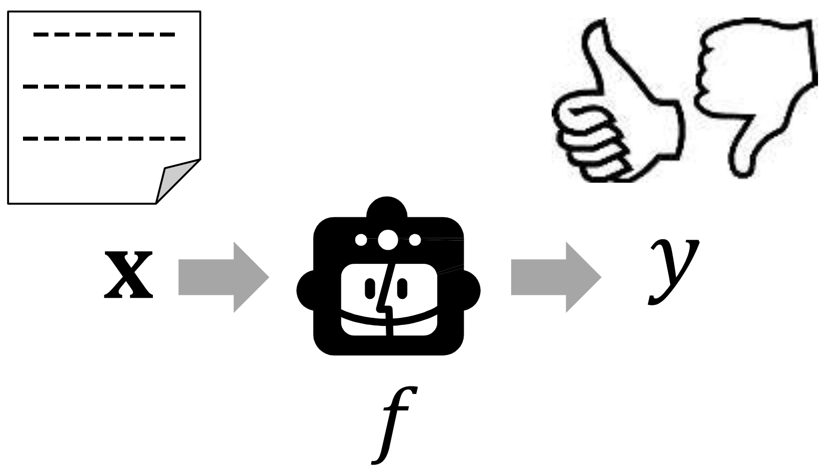
- Web
  - Search
  - Spam filtering
  - Social media

- Healthcare
  - Medical diagnosis

- Multimedia
  - Image/voice understanding
- System monitoring
  - Fault detection

An application of supervised classification learning: Sentiment analysis

- Judge if a document ($\mathbf{x}$) is positive or not ($y \in \{+1, -1\}$) toward a particular product or service
- For example, we want to know reputation of our newly launched service S
- Collect tweets by searching the word "S", and analyze them

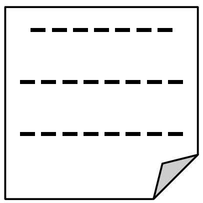


An application of supervised learning:
Some hand labeling followed by supervised learning

- First, give labels to some of the collected documents
  - 10,000 tweets hit the word "S"
  - Manually read 300 of them and give labels
    - "I used S, and found it not bad." → 
    - "I gave up S. The power was not on." → 
    - "I like S." → 
- Use the collected 300 labels to train a predictor.
  Then apply the predictor to the rest 9,700 documents

How to represent a document as a vector: bag-of-words representation

- Represent a document $\mathbf{x}$ using words appearing in it



$$\boldsymbol{x} := \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix}$$

Number of “good”

Number of “not”

⋮

Number of “like”

bag-of-words representation

- Note: design of the feature vector is left to users

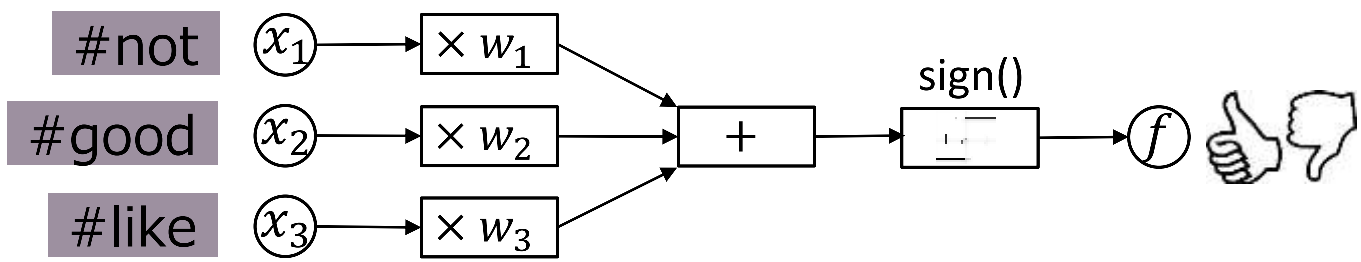
A model for classification:
Linear classification model

- Model f takes an input $\mathbf{x} = (x_1, x_2, \dots, x_D)^\top$ and outputs a value from $\{+1, -1\}$

$$\boxed{f(\mathbf{x}) = \mathrm{sign}(w_1 x_1 + w_2 x_2 + \cdots + w_D x_D)}$$

- Model parameter $\mathbf{w} = (w_1, w_2, \dots, w_D)^\top \in \mathbb{R}^D$:
    - w_d : contribution of x_d to the output

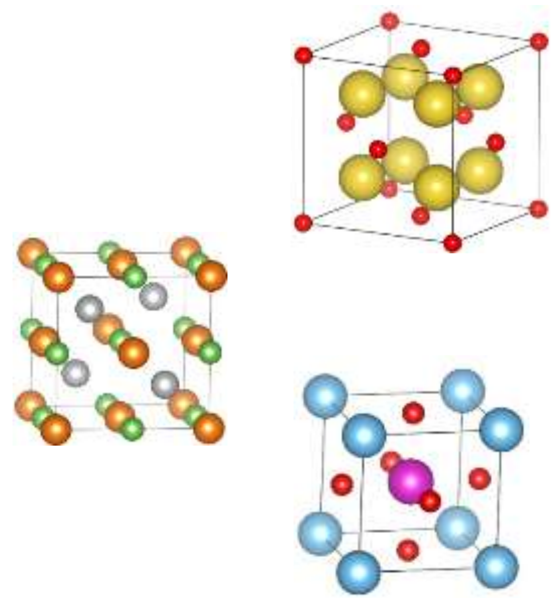
      ($x_d > 0$ contributes to $+1$, $x_d < 0$ contributes to -1)



An application of supervised regression learning: Discovering new materials

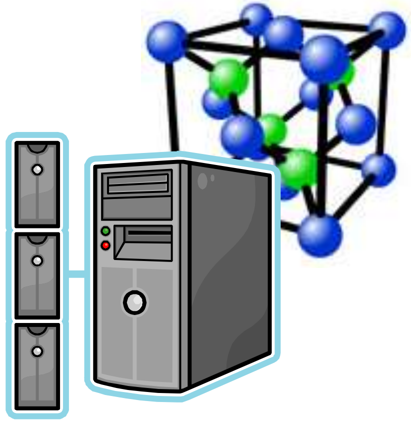
- Material science aims at discovering and designing new materials with desired properties
  - Volume, density, elastic coefficient, thermal conductivity, …
- Traditional approach:

1. Determine chemical structure
2. Synthesize the chemical compounds
3. Measure their physical properties



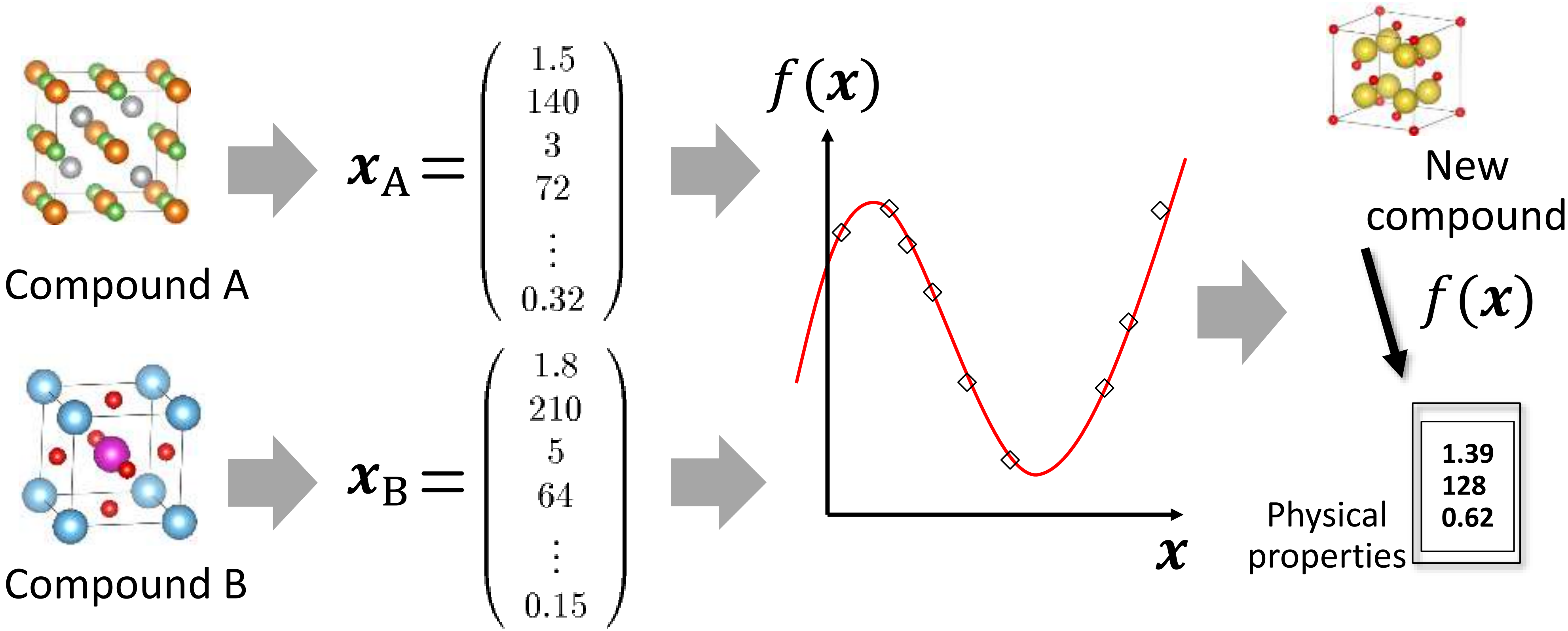
Computational approach to material discovery:
Still needs high computational costs

- Computational approach: First-order principle calculations based on quantum physics to run simulation to estimate physical properties
- First-order calculation still requires high computational costs
  - Proportional to the cubic number of atoms
  - Sometimes more than a month…



Data driven approach to material discovery: Regression to predict physical properties

- Predict the result of first-order principle calculation from data



Feature vector representation of chemical compounds

Estimate regression models of physical properties from data

Predict physical properties of new compounds

Recommendation systems

Recommender systems: Personalized information filter

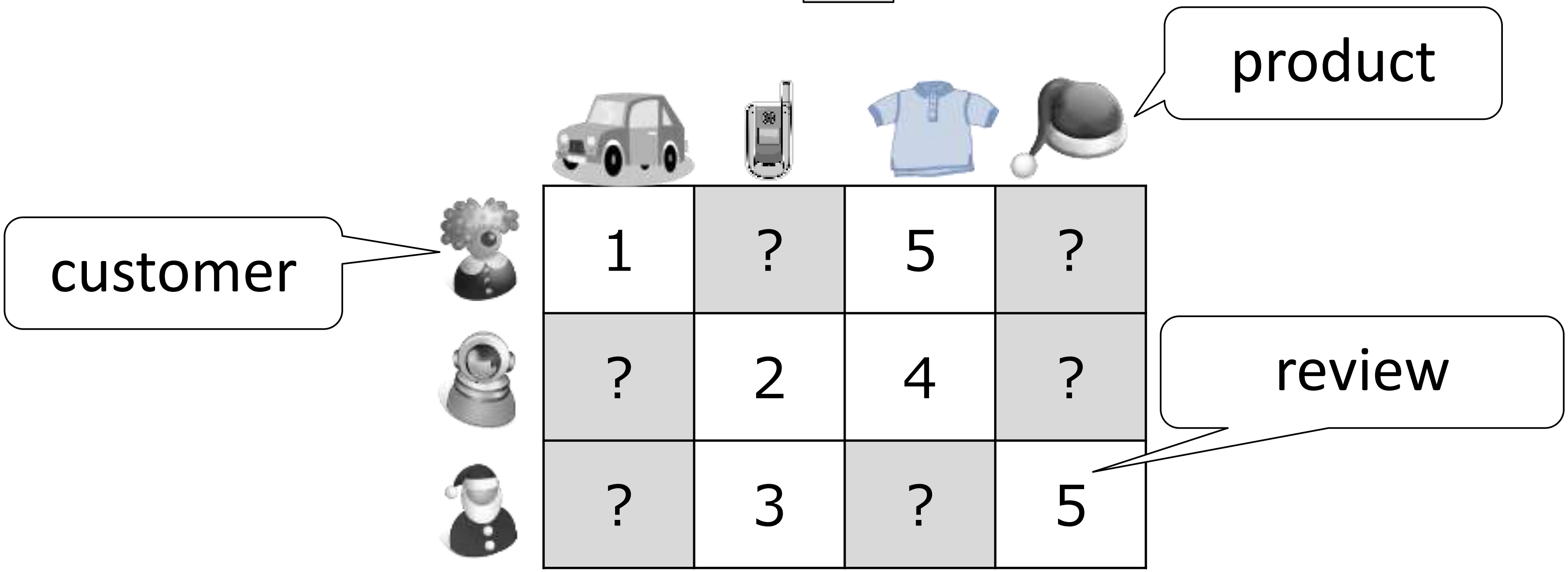
- Amazon offers a list of products I am likely to buy (based on my purchase history)

Ubiquitous recommender systems:
Recommender systems are present everywhere

- A major battlefield of machine learning algorithms
  - Netflix challenge (with $100 million prize)
- Recommender systems are present everywhere:
  - Product recommendation
    in online shopping stores
  - Friend recommendation on SNSs
  - Information recommendation
    (news, music, …)
  - …

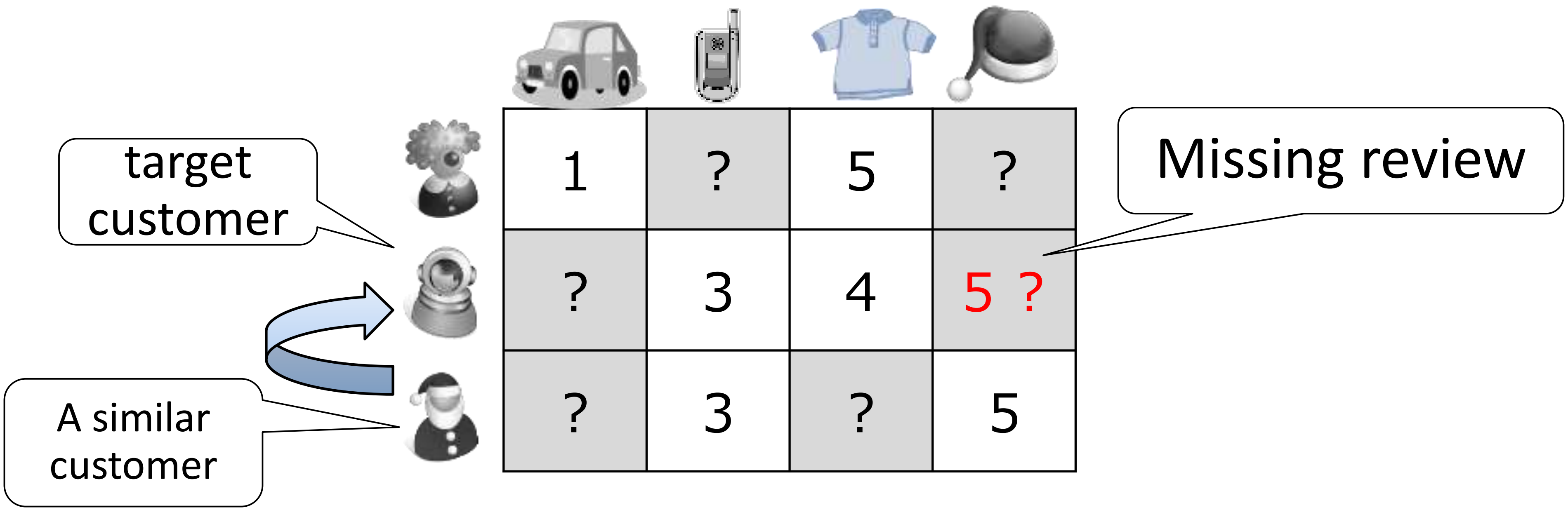
A formulation of recommendation problem: Matrix completion

- A matrix with rows (customers) and columns (products)
  - Each element = review score
- Given observed parts of the matrix, predict the unknown parts (?)



Basic idea of recommendation algorithms: "Find people like you"

- GroupLens: an earliest algorithm (for news recommendation)
  - Inherited by MovieLens (for Movie recommendation)
- Find people similar to the target customer, and predict missing reviews with theirs



1	?	5	?
?	3	4	5 ?
?	3	?	5

GroupLens: Weighted prediction using correlations among customers

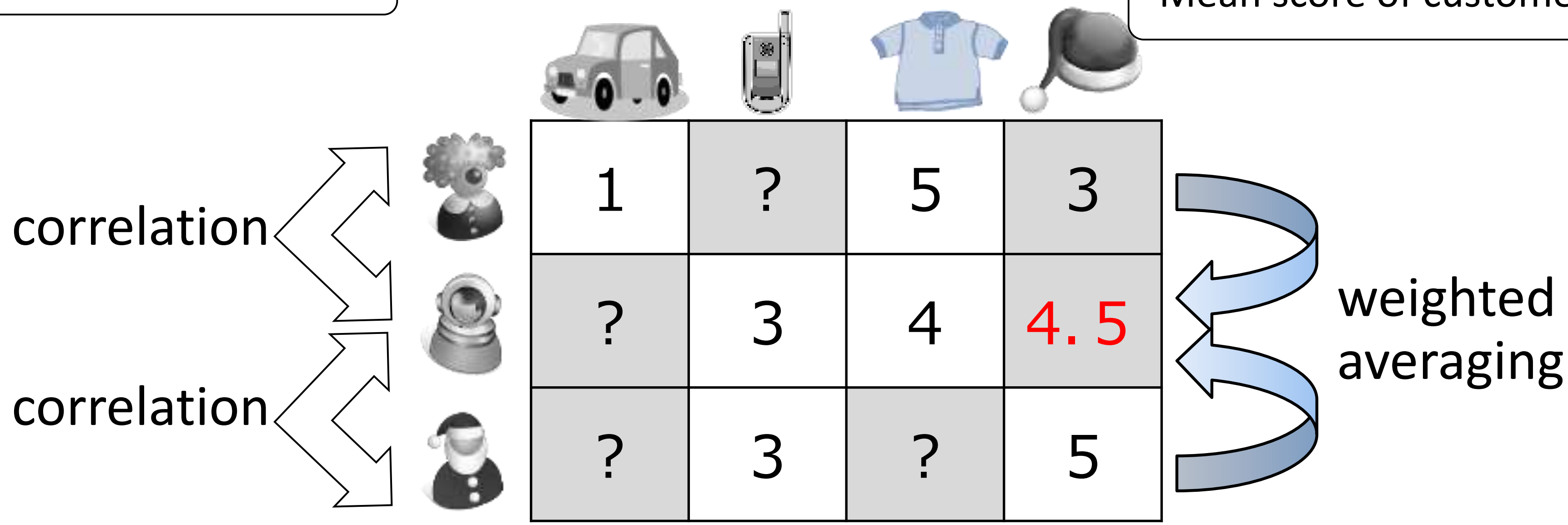
- Define customer similarity by correlation（of observed parts）
- Prediction by weighted averaging with correlations：

$$\hat{y}_{i,j} = \bar{y}_i + \sum_{k\neq i} r_{i,k} \left(y_{k,j} - \bar{y}_k \right) / \sum_{k\neq i} |r_{ij}|$$

Mean score of user i

Pearson correlation between users i and k

Mean score of customer k



1	?	5	3
?	3	4	4. 5
?	3	?	5

Low-rank assumption for matrix completion: GroupLens implicitly assumes low-rank matrices

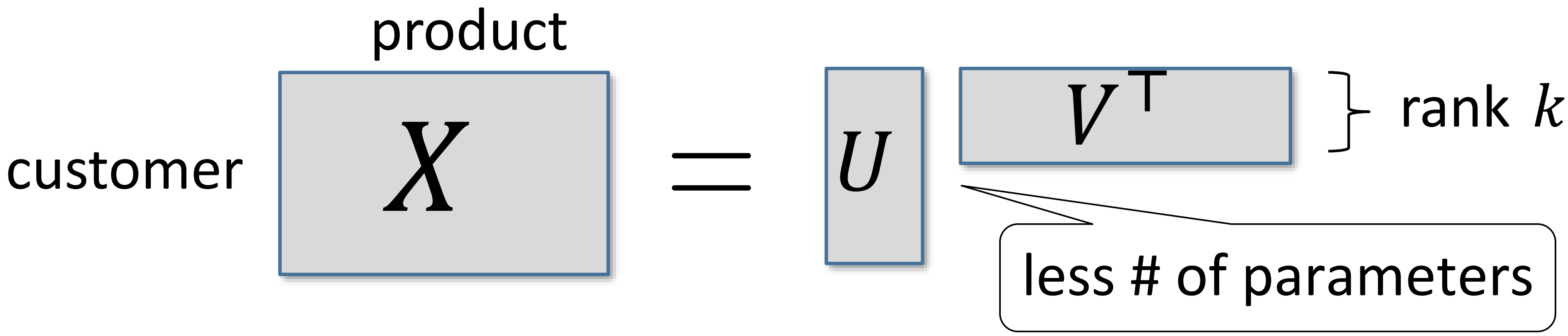
- Assumption of GroupLens algorithm:
  Each row is represented by a linear combination of the other rows (i.e. linearly dependent)

  ⇒ The matrix is not full-rank　（≒ low-rank）

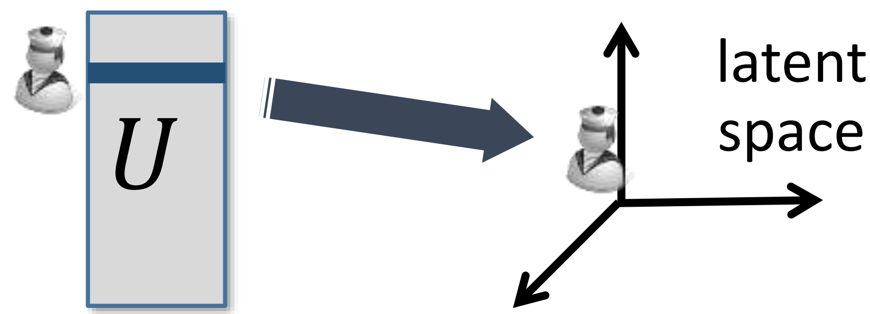
- Low-rank assumption helps matrix completion

Low-rank matrix factorization: Projection onto low-dimensional latent space

- Low-rank matrix: product of two (thin) matrices



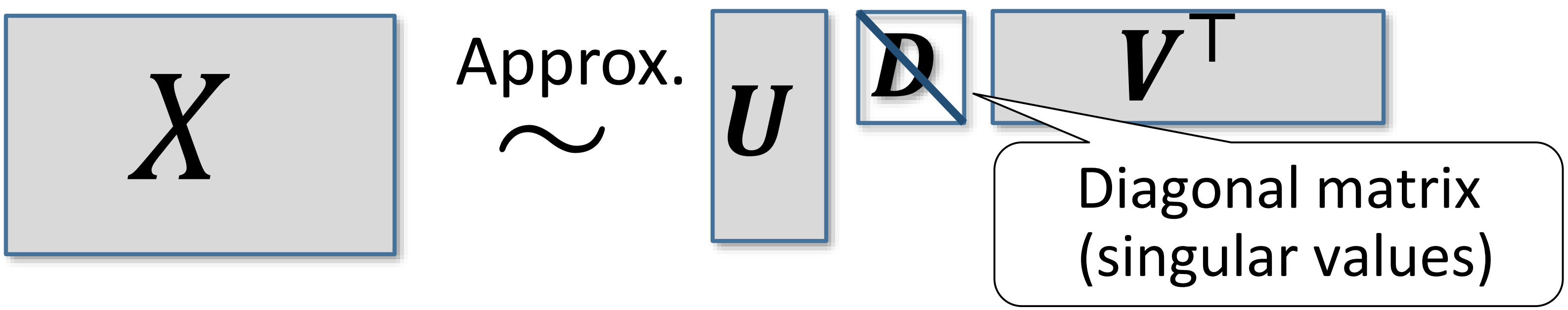
- Each row of $\boldsymbol{U}$ and $\boldsymbol{V}$ is an embedding of each customer (or product) onto low-dimensional latent space



Low-rank matrix decomposition methods: Singular value decomposition (SVD)

- Find a best low-rank approximation of a given matrix

$$\underset{\boldsymbol{Y}}{\text{minimize}} \ \| \boldsymbol{X} - \boldsymbol{Y} \|_{\mathrm{F}}^{2} \ \text{ s.t. } \operatorname{rank}(\boldsymbol{Y}) \le k$$

- Singular value decomposition (SVD)
  - 

    w.r.t. the constraints: $\boldsymbol{U}^{\top}\boldsymbol{U} = \boldsymbol{I}, \ \boldsymbol{V}^{\top}\boldsymbol{V} = \boldsymbol{I}$
  - The k leading eigenvectors of $\boldsymbol{X}^{\top}\boldsymbol{X}$ best approximate

Strategies for matrices with missing values: EM algorithm, gradient descent, and trace norm

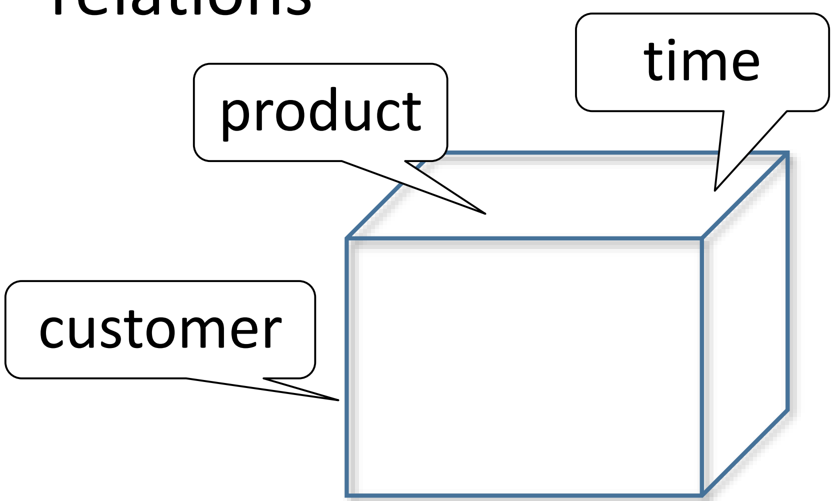
- SVD is not directly applicable to matrices with missing values
  - Our goal is to fill in missing values in a partially observed matrix
- For completion problem:
  - Direct application of SVD to a (somehow) filled matrix
  - Iterative applications: iterations of completion and decomposition
- For large scale data:
  Gradient descent using only observed parts
- Convex formulation: Trace norm constraint

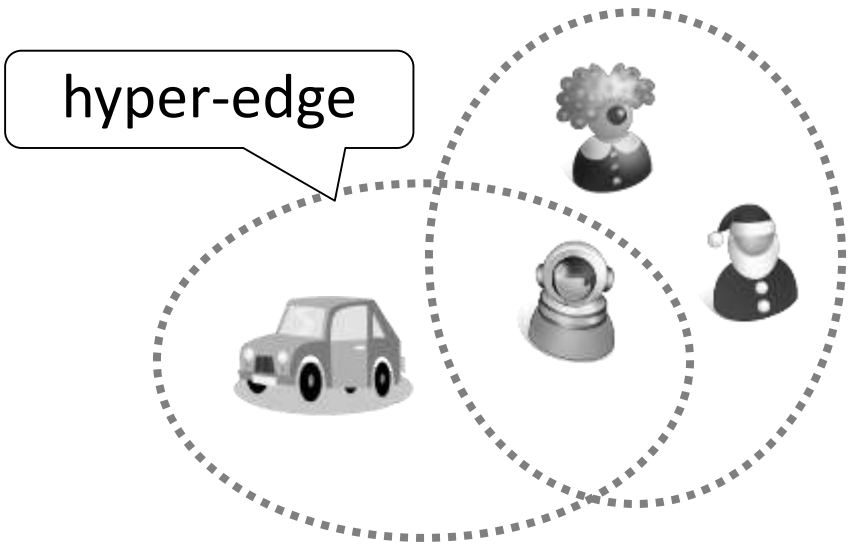
Predicting more complex relations: Multinomial relations

- Matrices can represent only one kind of relations
  - Various kinds of relations (actions):
    Review scores, purchases, browsing product information, …
  - Correlations among actions might help
- Multinomial relations:
  - (customer, product, action)-relation:
    (Alice, iPad, buy) represents “Alice bought an iPad.”
  - (customer, product, time)-relation:
    (John, iPad, July 12th) represents “John bought an iPad on July 12th.”

Multi-dimensional arrays: Representation of multinomial relations

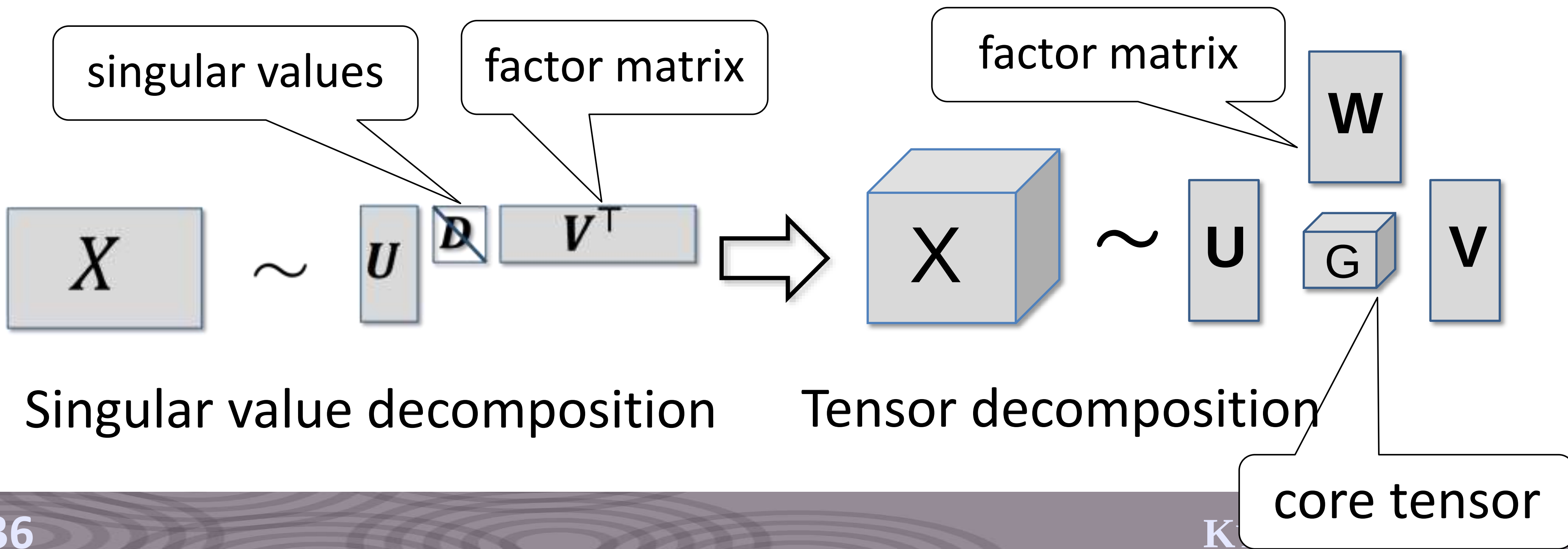
- Multidimensional array: Representation of complex relations among multiple objects
  - Types of relations (actions, time, conditions, …)
  - Relations among more than two objects
- Hypergraph: allows variable number of objects involved in relations





Tensor decomposition:
Generalization of low-rank matrix decomposition

- Generalization of matrix decomposition to multidimensional arrays
  - A small core tensor and multiple factor matrices
- Increasingly popular in machine learning/data mining



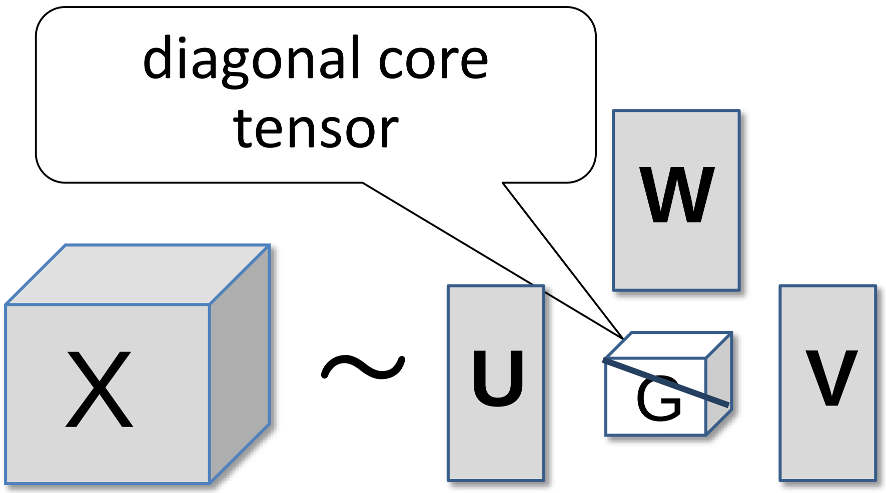
Singular value decomposition

Tensor decomposition

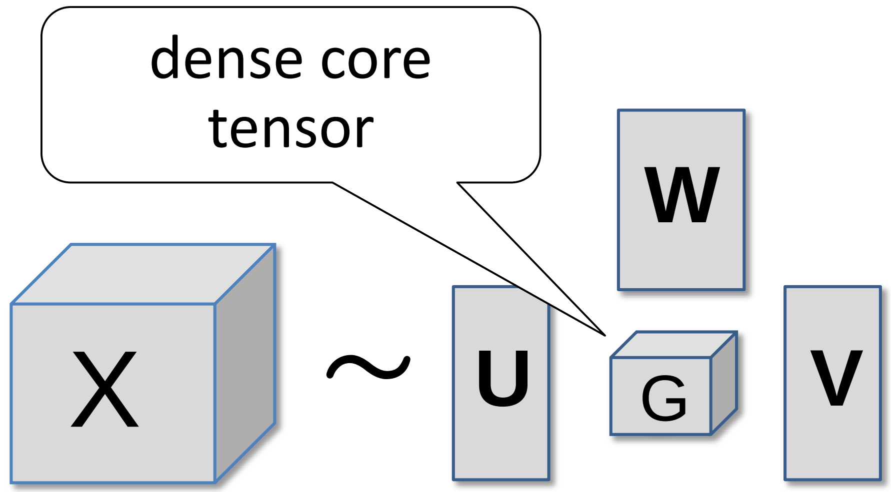
Tensor decompositions:
CP decomposition and Tucker decomposition

- CP decomposition: A natural extension of SVD
  (with a diagonal core)

- Tucker decomposition: A more compact model
  (with a dense core)



CP decomposition



Tucker decomposition

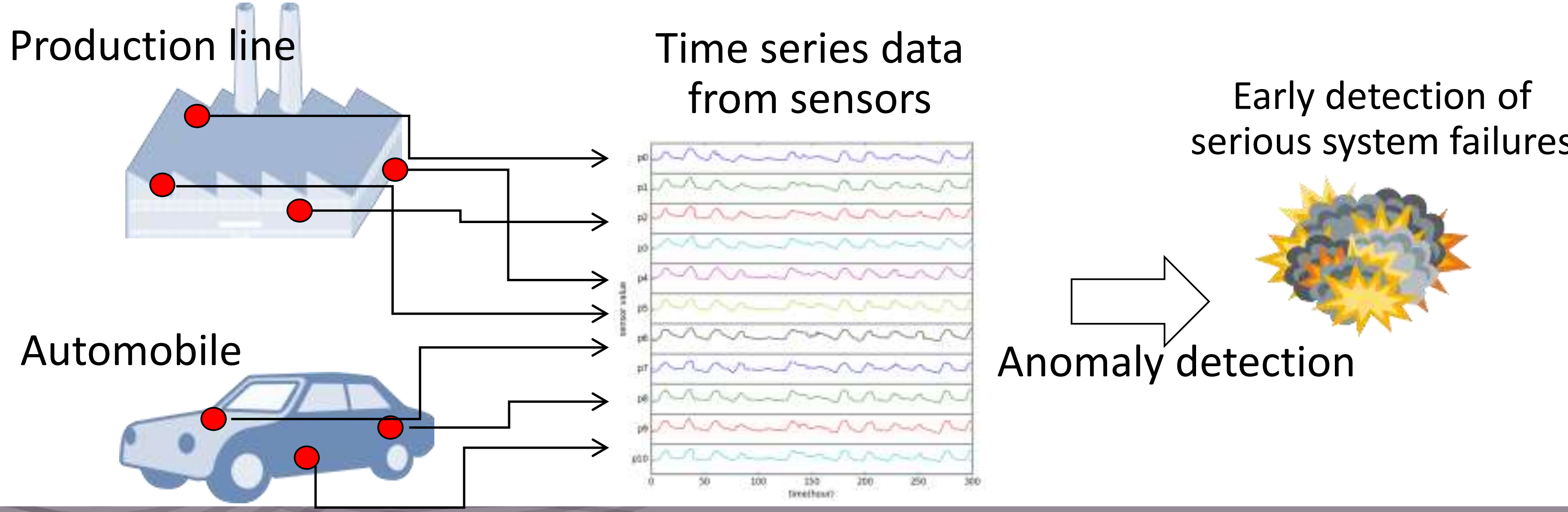
Applications of tensor decomposition: Tag recommendation, social network analysis, …

- Personalized tag recommendation (user×webpage×tag)
  - predicts tags a user gives a webpage
- Social network analysis (user×user×time)
  - analyzes time-variant relationships
- Web link analysis (webpage×webpage×anchor text)
- Image analysis (image×person×angle×light×…)

Anomaly detection

Anomaly detection:
Early warning for system failures reduces costs

- A failure of a large system can cause a huge loss
  - Breakdown of production lines in a factory, infection of computer virus/intrusion to computer systems, credit card fraud, terrorism, …
- Modern systems have many sensors to collect data
- Early detection of failures from data collected from sensors



Anomaly detection techniques: Find "abnormal" behaviors in data

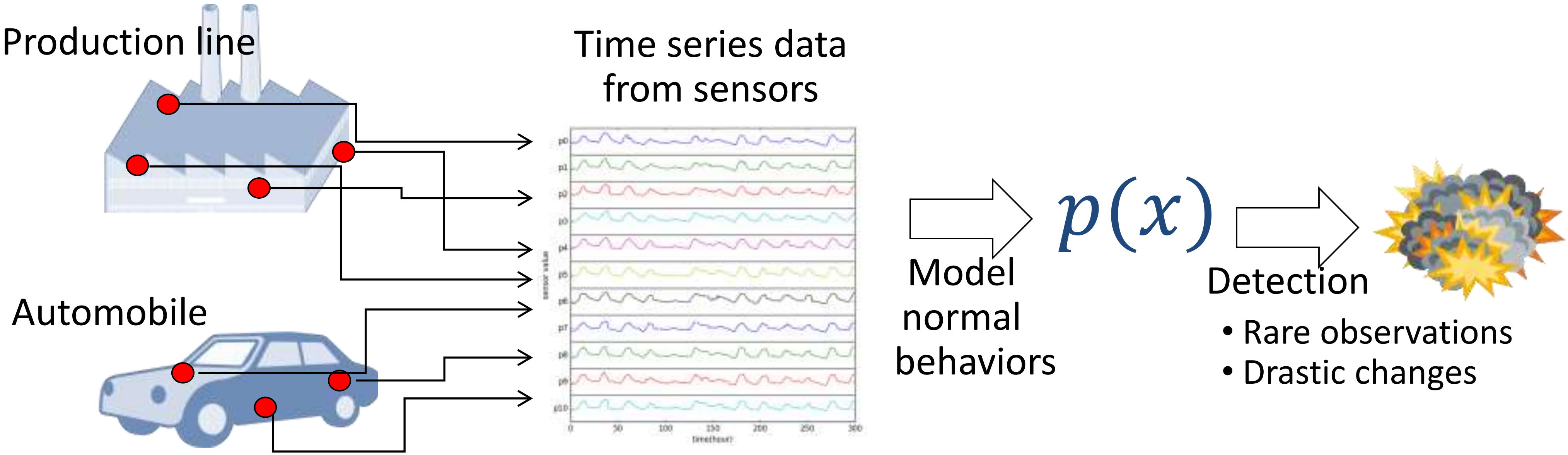
- We want to find precursors of failures in data
  - Assumption: Precursors of failures are hiding in data
- Anomaly: An "abnormal" patterns appearing in data
  - In a broad sense, state changes are also included: appearance of news topics, configuration changes, …
- Anomaly detection techniques find such patterns from data and report them to system administrators

Difficulty in anomaly detection: Failures are rare events

- If target failures are known ones, they are detected by using supervised learning:
  1. Construct a predictive model from past failure data
  2. Apply the model to system monitoring
- However, serious failures are usually rare, and often new ones → (Almost) no past data are available
- Supervised learning is not applicable

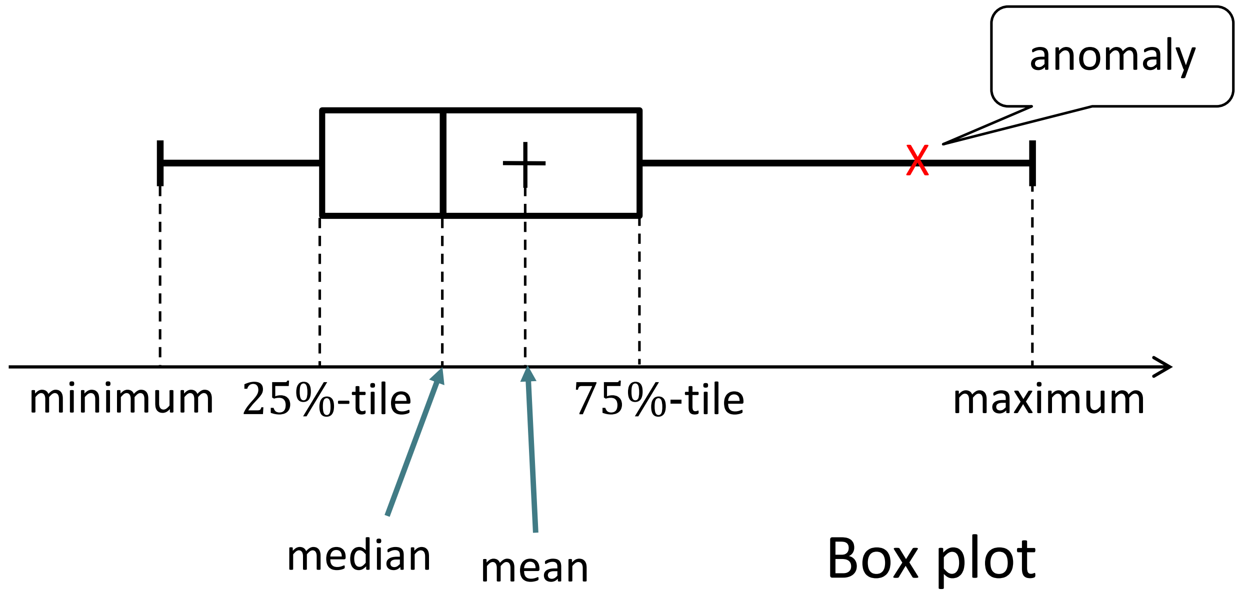
An alternative idea:
Model the normal times, detect deviations from them

- Difficult to model anomalies → Model normal times
  - Data at normal times are abundant
- Report “strange” data according to the normal time model
  - Observation of rare data is a precursor of failures



A simple unsupervised approach: Anomaly detection using thresholds

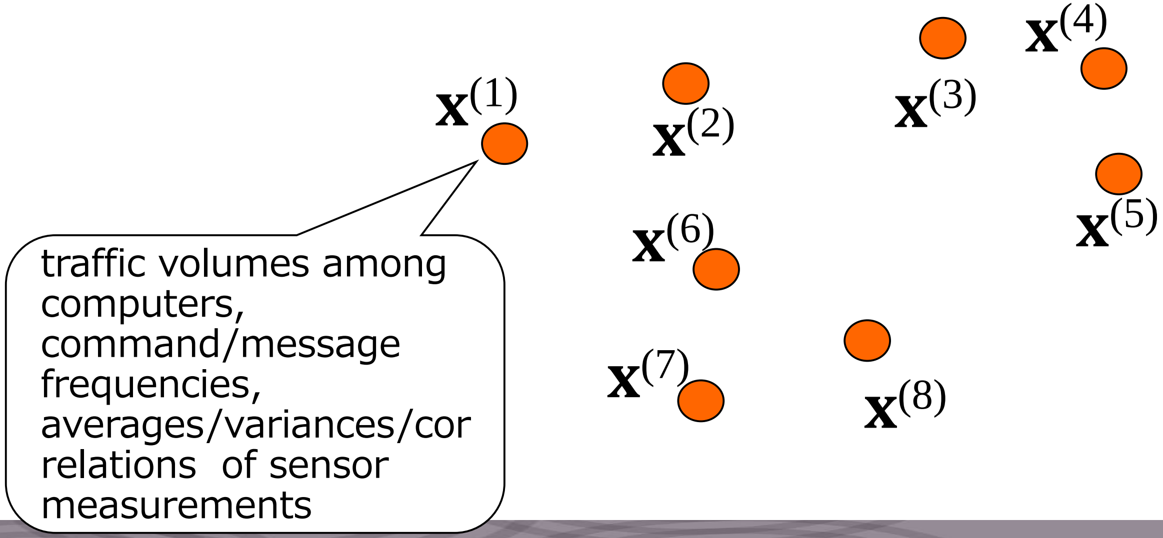
- Suppose a 1-dimensional case (e.g. temperature)
- Find the value range of the normal data (e.g. 20-50 ℃)
- Detect values deviates from the range, and report them as anomalies （e.g. 80℃ is not in the normal range)



Box plot

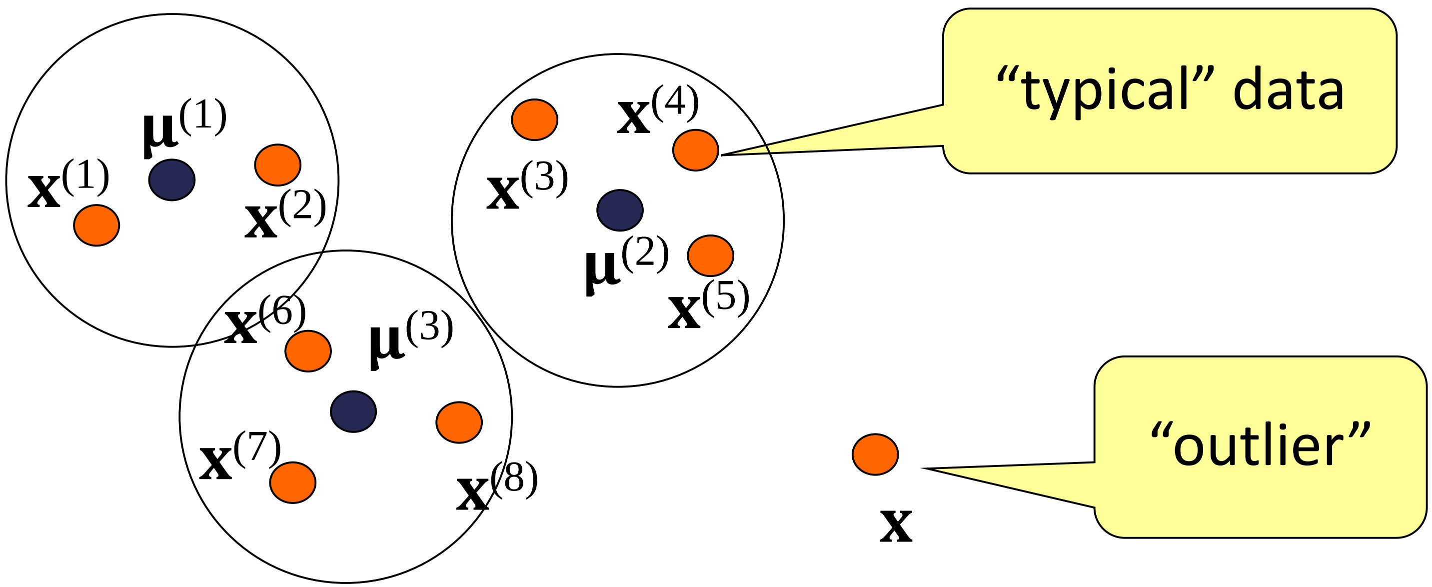
Clustering for high-dimensional anomaly detection: Model the normal times by grouping the data

- More complex cases:
  - Multi-dimensional data
  - Several operation modes in the systems
- Divide normal time data $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$ into K groups
  - Groups are represented by centers $\{\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(2)}, \dots, \boldsymbol{\mu}^{(N)}\}$



Clustering for high-dimensional anomaly detection: Find anomalies not belonging to the groups

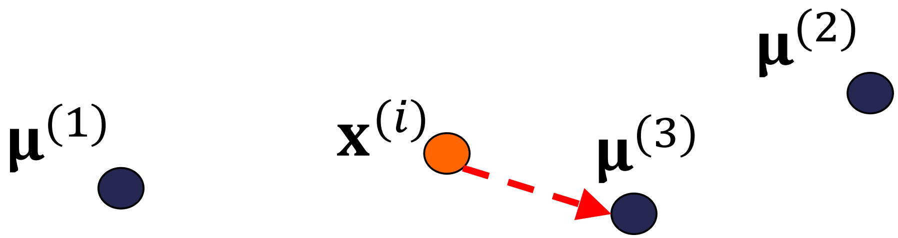
- Divide normal time data $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$ into K groups
  - Groups are represented by centers $\{\boldsymbol{\mu}^{(1)}, \boldsymbol{\mu}^{(2)}, \dots, \boldsymbol{\mu}^{(N)}\}$
- Data $\mathbf{x}$ is an "outlier" if it lies far from all of the centers
  = system failures, illegal operations, instrument faults



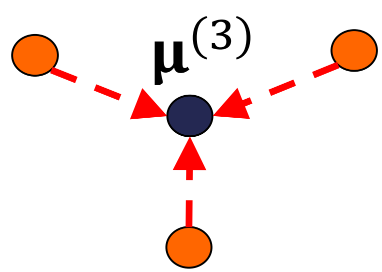
K-means algorithm:
Iterative refinement of groups

- Repeat until convergence:

1. Assign each data $\mathbf{x}^{(i)}$ to its nearest center $\boldsymbol{\mu}^{(k)}$



2. Update each center to the center of the assigned data



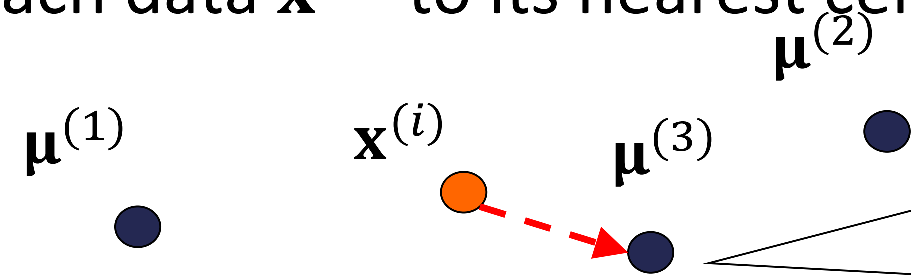
Anomaly detection in time series: On-line anomaly detection

- Most anomaly detection applications require real-time system monitoring
- Data instances arrive in a streaming manner:
  - $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(t)}, \ldots$: at each time t, new data $\mathbf{x}^{(t)}$ arrives
- Each time a new data arrives, evaluate its anomaly
- Also, models are updated in on-line manners:
  - In the one dimensional case, the threshold is sequentially updated
  - In clustering, groups (clusters) are sequentially updated

Sequential K-means: Simultaneous estimation of clusters and outliers

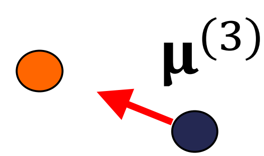
- Data arrives in a streaming manner, and apply clustering and anomaly detection at the same time

1. Assign each data $\mathbf{x}^{(i)}$ to its nearest center $\boldsymbol{\mu}^{(k)}$



If the distance is large, report the data as an anomaly

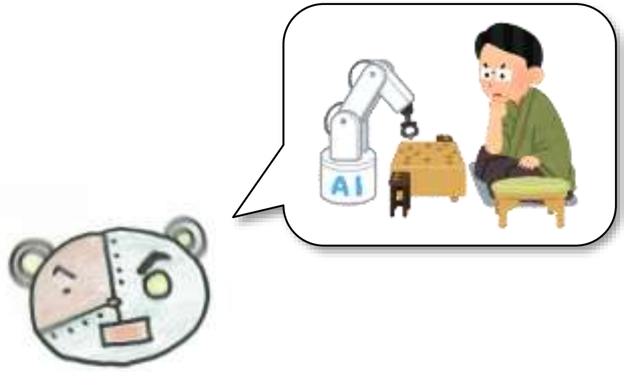
2. Slightly move the center to the data



Limitation of unsupervised anomaly detection: Details of failures are unknown

- In supervised anomaly detection, we know what the failures are
- In unsupervised anomaly detection,
  we can know something is happening in the data,
  but cannot know what it is
  - Failures are not defined in advance
- Based on the reports to system administrators,
  they have to investigate what is happening, what are the reasons, and what they should do

Recent topics

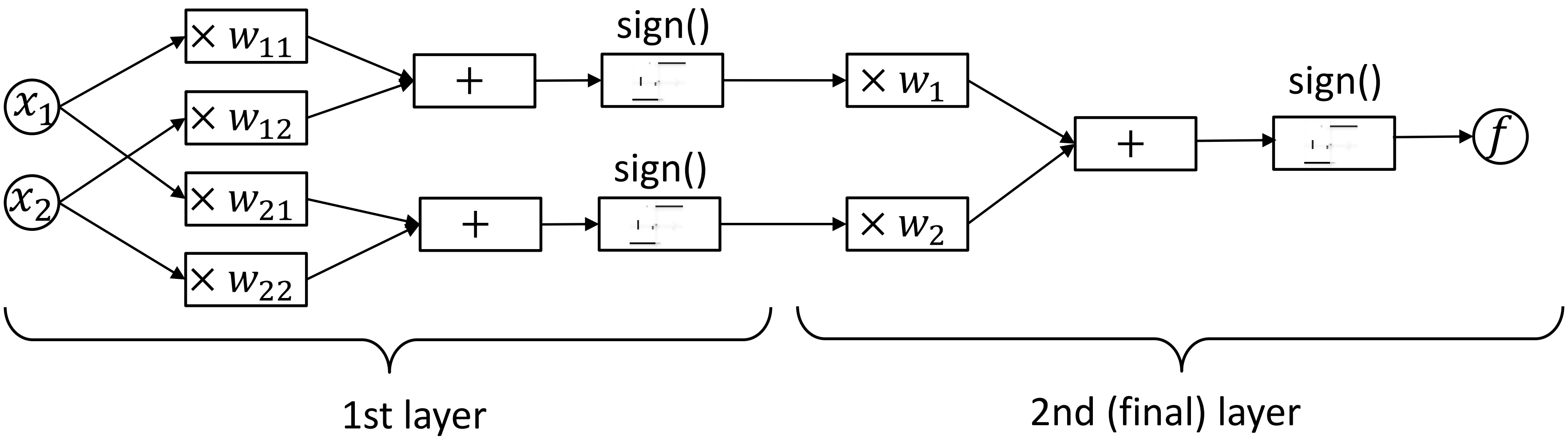


Emergence of deep learning: Significant improvement of prediction accuracy

- Artificial neural networks were hot in 1980s, but burnt low after that…
- In 2012, a deep NN system won in the ILSVRC image recognition competition with 10% improvement
- Major IT companies (such as Google and Facebook) invest much in deep learning technologies
- Big trend in machine learning research

Deep neural network:
Deeply stacked NN for high representational power

- Essentially, multi-layer neural networks
  - Regarded as stacked linear classification models
    - First to semi-final layers bear feature extraction
    - Final layer makes predictions
- Deep stacking introduces high non-linearity in the model and ensures high representational power



A model for classification:
Linear classification model

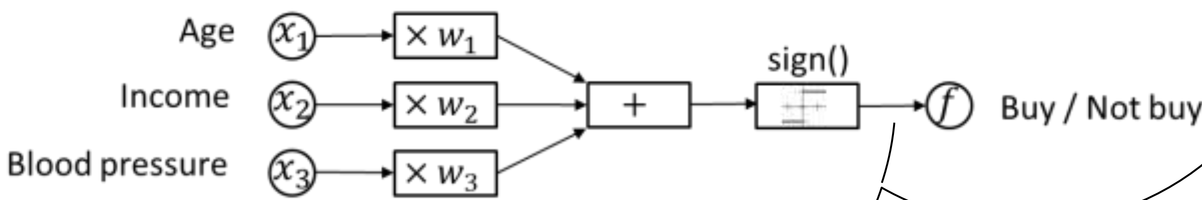
- Model f takes an input $\mathbf{x} = (x_1, x_2, \ldots, x_D)^\top$ and outputs a value from $\{+1, -1\}$

$$\boxed{f(\mathbf{x}) = \mathrm{sign}(w_1 x_1 + w_2 x_2 + \cdots + w_D x_D)}$$

– Model parameter $\mathbf{w} = (w_1, w_2, \ldots, w_D)^\top \in \mathbb{R}^D$:

- w_d : contribution of x_d to the output

  ($x_d > 0$ contributes to $+1$, $x_d < 0$ contributes to -1)



What is the difference from the past NN?: Deep structures and new techniques with modern flavors

- Differences from the ancient NNs:
  - Far more computational resources are available now
  - Deep network structure: from wide-and-shallow to narrow-and-deep
  - New techniques: Dropout, ReLU, Adversarial learning, …
- Unfortunately we will not cover DNNs in this lecture ….