

PERTEMUAN 1

Konsep Pembelajaran Mesin

Pendahuluan

Mesin learning memungkinkan sebuah sistem perangkat lunak mempelajari data-data yang ada pada saat ini, untuk kemudian digunakan dalam memprediksi masa depan. Dengan serangkaian algoritma yang digunakan untuk meningkatkan proses dan penemuan pola dalam sebuah data.

Pada praktiknya machine learning tidak dapat berjalan sendiri system ini perlu sebuah kolaborasi antara beberapa aspek dalam penyelesaian permasalahan seperti data science, data engineer, dan bisnis analis

History of Machine Learning

- Tahun 1920-an Thomas Bayes dan Andrey Markov mengemukakan dasar-dasar machine learning dan konsepnya.
- Contoh machine learning di era 90an *Deep Blue* yang dibuat oleh IBM pada tahun 1996.

Apa Itu *Machine Learning..?*

- *Machine Learning (ML)* merupakan sebuah cabang ilmu dari AI (*artificial intelligent*) adalah mesin yang dikembangkan untuk bisa belajar dengan sendirinya tanpa arahan dari penggunanya. Pembelajaran mesin dikembangkan berdasarkan disiplin ilmu lainnya seperti statistika, matematika dan *data mining* sehingga mesin dapat belajar dengan menganalisa data tanpa perlu di program ulang atau diperintah.
- *Machine Learning* memiliki kemampuan memperoleh data dengan perintah ia sendiri (ML) dapat mempelajari data yang ia peroleh sehingga bisa melakukan tugas tertentu

Apa Itu *Machine Learning..?* (Lanjutan)

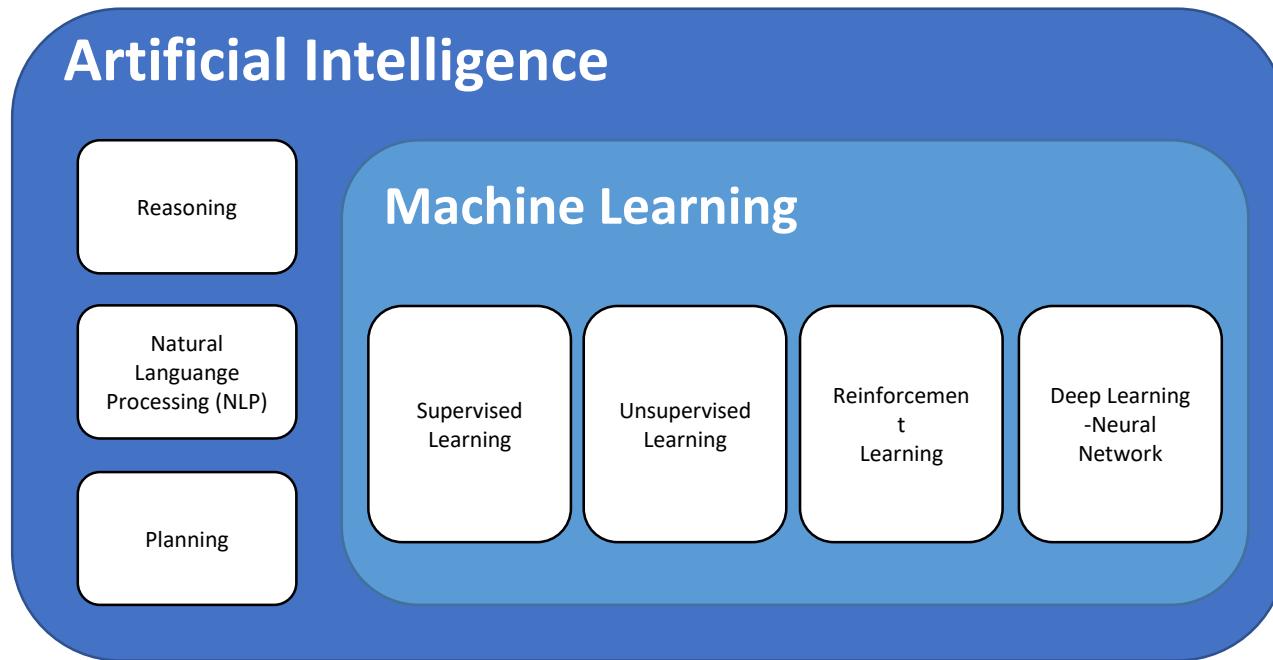
Meniru kemampuan belajar manusia

Contoh Implementasi *Machine Learning*

1. Self Driving Car
2. Recommended System
3. Chat Bot
4. Search engine
5. Face Recognition
6. Voice Recognition
7. Dll.

Bagian dari *Artificial intelligence (AI)*

Artificial intelligence sebuah kategori yang didalam nya terdapat *Machine Learning (ML)* dan *Natural Languange Processing (NLP)*



Subset AI Lain

Reasoning (Penalaran)

Reasoning memungkinkan sistem membuat kesimpulan berdasarkan data.

Contoh

Jika suatu system **telah** memiliki data tentang penetasan telur

Ditanya “Pada suhu berapa yg digunakan untuk penetasan telur..?”

Maka system akan menjawab “38 derajat celcius”

Contoh lain screening Covid-19 di aplikasi Mobile JKN

Natural Language Processing

Kemampuan sebuah mesin untuk memahami teks tertulis dan ucapan manusia.

Contoh : Applikasi google Assisten, siri Dll.

Subset AI Lain (lanjutan)

Planning (Penalaran)

Perencanaan otomatis adalah kemampuan sistem cerdas untuk bertindak secara mandiri dan fleksibel untuk membangun urutan tindakan untuk mencapai tujuan akhir.

Contoh Pencarian Rute pada Maps google

Pendekatan *Machine Learning*

Teknik pembelajaran mesin diperlukan untuk meningkatkan akurasi dalam sebuah model prediktif, terdapat beberapa pendekatan yang berbeda tergantung dari karakteristik jenis data dan volume

Supervised learning

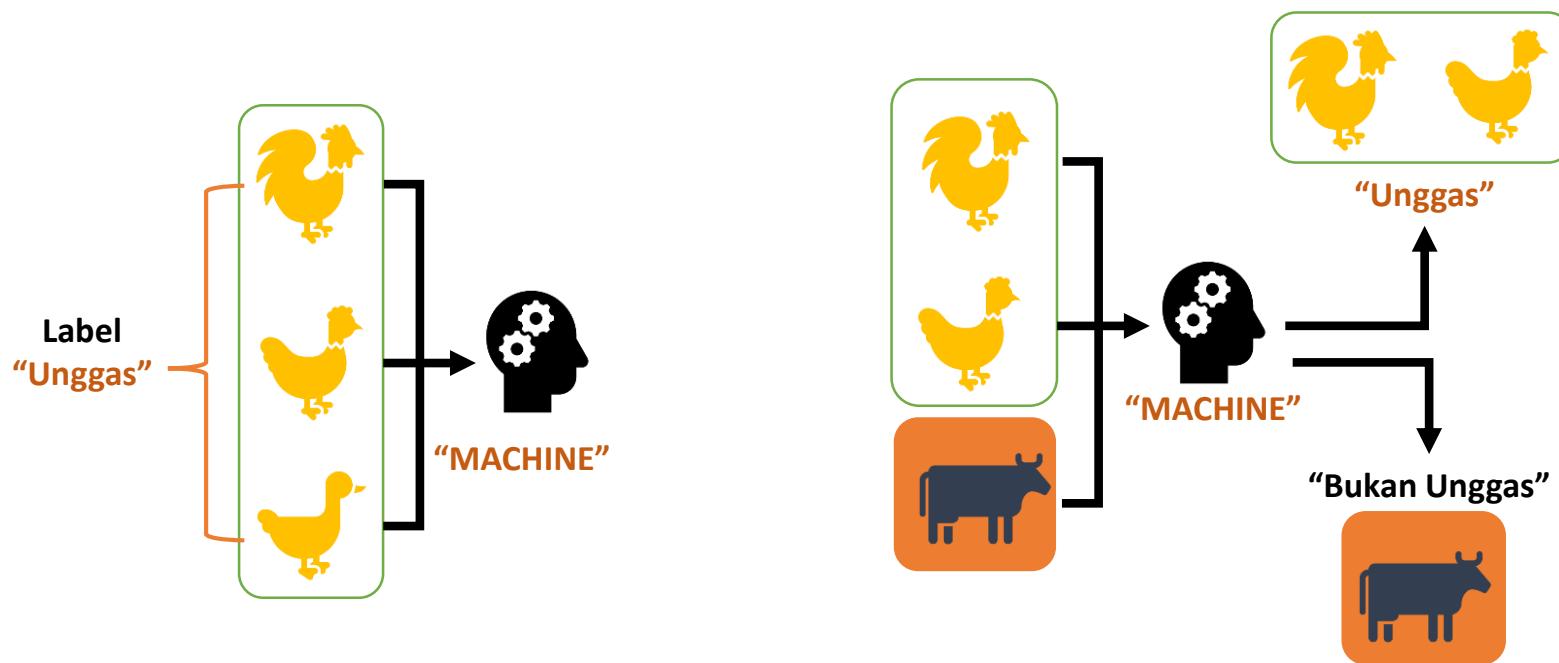
Unsupervised learning

Reinforcement learning

Deep learning

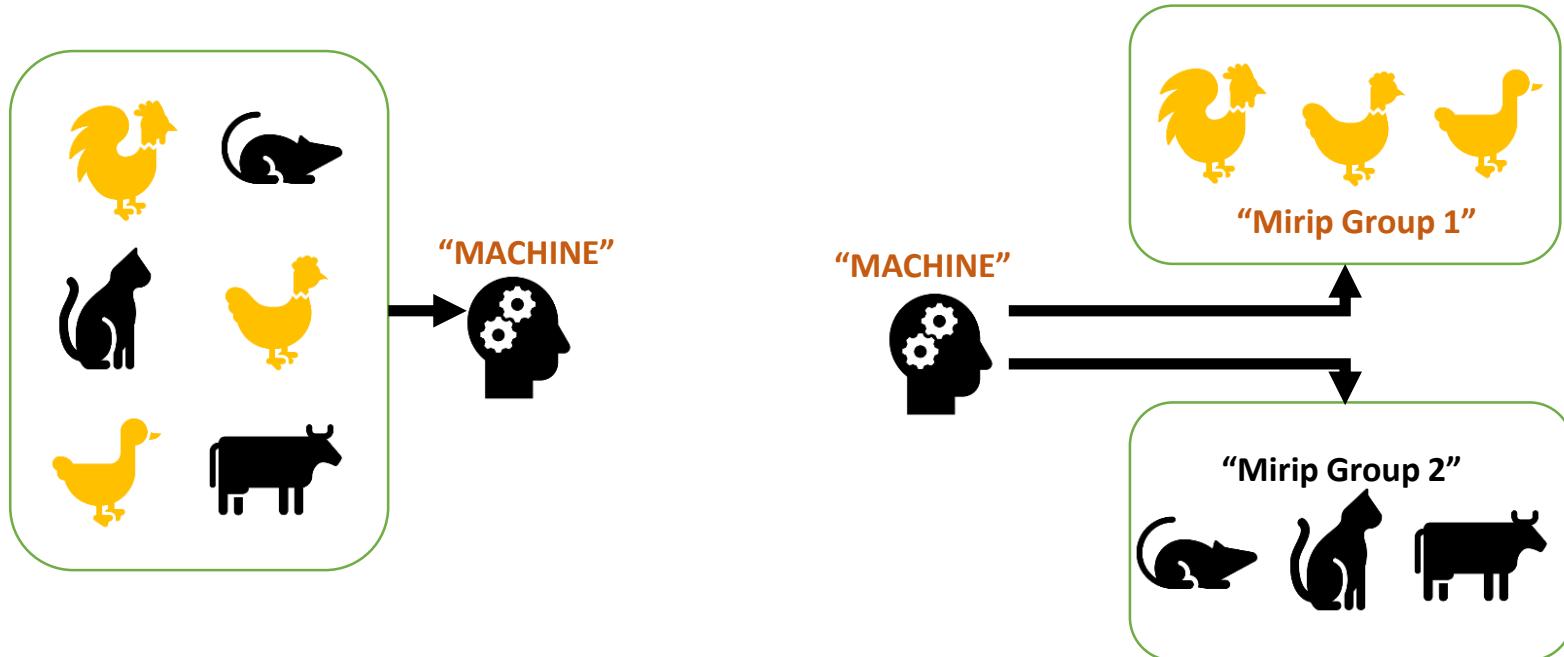
Supervised learning

Contohnya gambar unggas di tag “Unggas” di tiap masing masing image unggas dan gambar mamalia di tag “mamalia” di tiap masing gambar mamalia. Machine learning kategori dapat berupa clasification (“unggas”, “mamalia”, “invertebrata”, dsb)



Unsupervised learning

Unsupervised learning tidak menggunakan label dalam memprediksi target features / variable. Melainkan menggunakan ke samaan dari attribut attribut yang dimiliki. Jika attribut dan sifat sifat dari data data feature yang diekstrak memiliki kemiripan miripan, maka akan dikelompok kelompok (clustering)

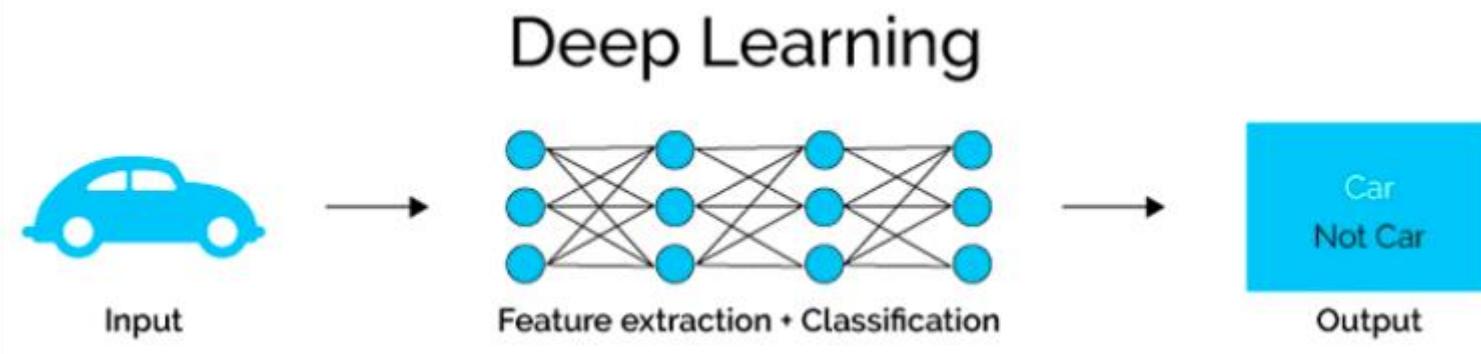


Reinforcement learning

Algoritma ini dimaksudkan untuk membuat komputer dapat belajar sendiri dari lingkungan (*environment*) melalui sebuah *agent*. Jadi komputer akan melakukan pencarian sendiri (*self discovery*) dengan cara berinteraksi dengan *environment*

Deep learning

metode pembelajaran yang dilakukan oleh mesin dengan cara meniru bagaimana sistem dasar otak manusia bekerja. Sistem dasar otak manusia bekerja ini disebut *neural networks*. deep learning disebut menggunakan *artificial neural networks* yang dengan kata lain menggunakan '*neural networks* buatan'.



Pertemuan 2

Taksonomi Pembelajaran Mesin

Dalam mempelajari teori machine learning maka diperlukan pemahaman tentang

1. Vektor dan Matrix
2. Operasi pada Vektor dan Matrix
3. Teori Probabilitas
4. Bayes Rule
5. Maksimum Likelihood

Vektor

Vektor adalah matriks yang hanya memiliki satu kolom saja atau hanya memiliki satu baris saja. Vektor dinotasikan dengan huruf kecil dan dicetak tebal. Penulisan vektor adalah sebagai berikut.

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Vektor tersebut dapat ditulis juga dalam bentuk transpose vektor yaitu

$$\mathbf{x}' = (x_1 \ x_2 \ \dots \ x_n)$$

atau bisa juga ditulis dalam bentuk

$$\mathbf{x}' = (x_1, x_2, \dots, x_n)$$

Berikut ini adalah beberapa buah contoh vektor.

$$\mathbf{a} = \begin{pmatrix} 5 \\ 6 \\ 6 \\ 9 \end{pmatrix} \quad \mathbf{b} = \begin{pmatrix} 8 \\ 1 \\ 9 \end{pmatrix} \quad \mathbf{c} = \begin{pmatrix} 4 \\ 0 \\ 5 \\ 4 \\ 5 \\ 9 \end{pmatrix}$$

MATRIKS

Matriks adalah susunan angka-angka dalam bentuk bujur sangkar atau persegi panjang yang diapit oleh tanda kurung. Semua bilangan yang ada dalam matriks adalah bilangan real. Matriks dinotasikan dengan huruf kapital dan dicetak tebal.

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1p} \\ a_{21} & a_{22} & \dots & a_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{np} \end{pmatrix}$$

Matriks **A** di atas memiliki n baris dan p kolom. Matriks **A** biasa disebut juga dengan matriks yang berukuran $n \times p$. Notasi a_{ij} dalam matriks menunjukkan elemen matriks baris ke- i dan kolom ke- j .

Di bawah ini adalah beberapa buah contoh matriks.

$$\mathbf{B} = \begin{pmatrix} 6 & 5 & 4 & 9 \\ 3 & 7 & 3 & 5 \\ 4 & 4 & 6 & 1 \end{pmatrix} \quad \mathbf{C} = \begin{pmatrix} 3 & 4 & 8 \\ 7 & 6 & 5 \\ 4 & 6 & 4 \end{pmatrix} \quad \mathbf{D} = \begin{pmatrix} 4 & 6 & 4 \\ 8 & 4 & 6 \\ 5 & 3 & 5 \\ 8 & 5 & 1 \end{pmatrix}$$

Matriks **B** merupakan matriks berukuran 3×4 , matriks **C** berukuran 3×3 dan matriks **D** berukuran 4×3 .

Taksonomi Pembelajaran Mesin

Taksonomi diambil dari Bahasa Yunani, *tassein* yang berarti mengelompokkan dan *nomos* yang berarti aturan.

Sehingga taksonomi dapat diartikan sebagai pengelompokan suatu hal berdasarkan aturan tertentu.

Dalam konteks pembelajaran mesin, pembelajaran mesin dapat dikelompokan menjadi 2 kelompok berdasarkan cara atau Teknik pembelajarannya, yaitu:

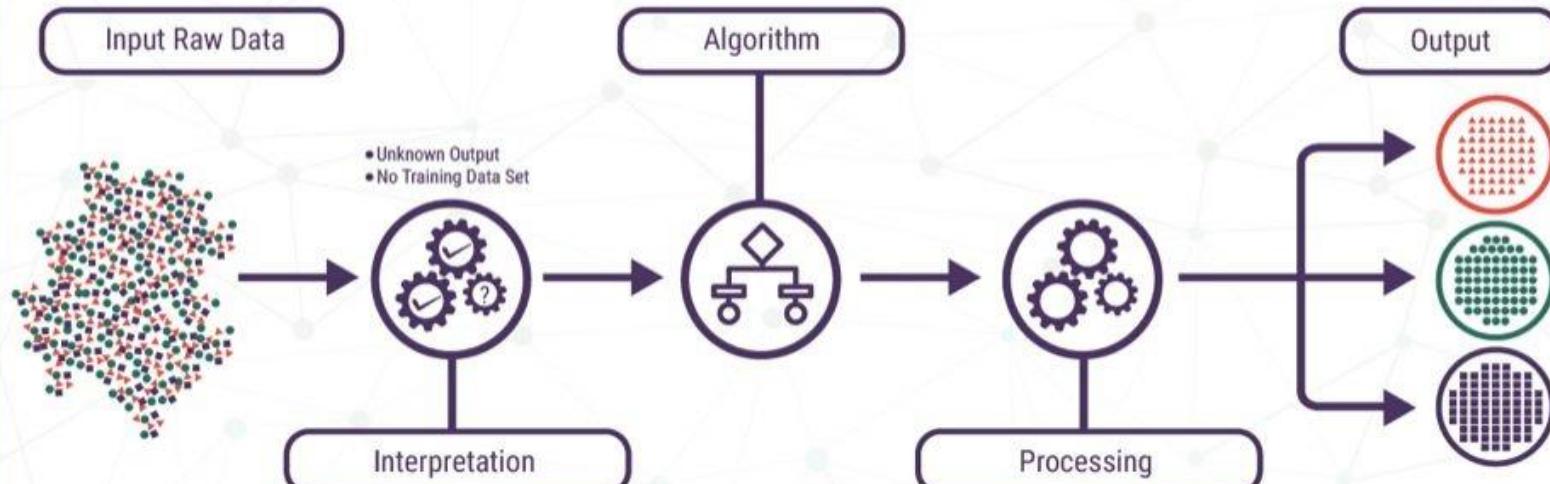
1. Pembelajaran tanpa supervisi (*unsupervised learning*)
2. Pembelajaran dengan supervisi (*supervised learning*)

Pembelajaran tanpa Supervisi *(unsupervised learning)*

- Sebuah pendekatan dimana tidak terdapat data latih, dan tidak memiliki atribut target
- Tujuan dari pendekatan ini adalah mengelompokkan suatu data kedalam beberapa kelompok baru
- Algoritma digunakan untuk mencari pola dari semua atribut
- Kumpulan data atau dataset yang digunakan tidak memiliki atribut target
- Clustering merupakan contoh dari *unsupervised learning*

Pembelajaran tanpa Supervisi *(unsupervised learning)*

UNSUPERVISED LEARNING



Dataset tanpa target

Attribute/Feature

	Sepal Length (cm)	Sepal Width (cm)	Petal Length (cm)	Petal Width (cm)
1	5.1	3.5	1.4	0.2
2	4.9	3.0	1.4	0.2
3	4.7	3.2	1.3	0.2
4	4.6	3.1	1.5	0.2
5	5.0	3.6	1.4	0.2
...				
51	7.0	3.2	4.7	1.4
52	6.4	3.2	4.5	1.5
53	6.9	3.1	4.9	1.5
54	5.5	2.3	4.0	1.3
55	6.5	2.8	4.6	1.5
...				
101	6.3	3.3	6.0	2.5
102	5.8	2.7	5.1	1.9
103	7.1	3.0	5.9	2.1

Algoritma *unsupervised learning*

Salah satu Teknik *unsupervised learning* adalah *clustering*. *Clustering* dapat dilakukan menggunakan beberapa algoritma diantaranya:

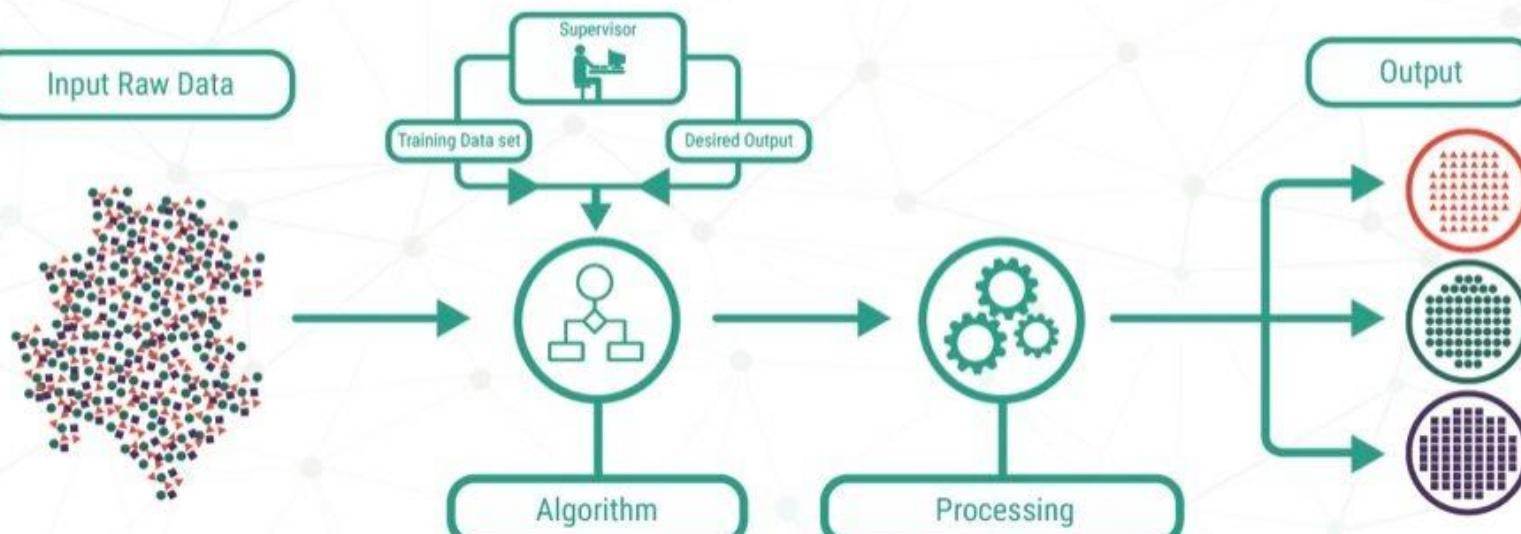
1. k-means
2. K-medoids
3. Self-organizing map
4. Fuzzy c means

Pembelajaran dengan Supervisi (*supervised learning*)

- Sebuah pendekatan dimana sudah terdapat data yang dilatih, dan terdapat atribut target (nilai yang akan diprediksi)
- Tujuan dari pendekatan ini adalah mengelompokkan suatu data kedalam kelompok data yang sudah ada
- Kumpulan data atau dataset yang digunakan sudah memiliki atribut target
- Klasifikasi dan Regresi (Prediksi dan Estimasi) merupakan contoh dari *supervised learning*

Pembelajaran dengan Supervisi (*supervised learning*)

SUPERVISED LEARNING



Dataset dengan target

Attribute/Feature

Class/Label/Target

	Sepal Length (cm)	Sepal Width (cm)	Petal Length (cm)	Petal Width (cm)	Type
1	5.1	3.5	1.4	0.2	Iris setosa
2	4.9	3.0	1.4	0.2	Iris setosa
3	4.7	3.2	1.3	0.2	Iris setosa
4	4.6	3.1	1.5	0.2	Iris setosa
5	5.0	3.6	1.4	0.2	Iris setosa
...					
51	7.0	3.2	4.7	1.4	Iris versicolor
52	6.4	3.2	4.5	1.5	Iris versicolor
53	6.9	3.1	4.9	1.5	Iris versicolor
54	5.5	2.3	4.0	1.3	Iris versicolor
55	6.5	2.8	4.6	1.5	Iris versicolor
...					
101	6.3	3.3	6.0	2.5	Iris virginica
102	5.8	2.7	5.1	1.9	Iris virginica
103	7.1	3.0	5.9	2.1	Iris virginica

Nominal

Numerik

Algoritma *supervised learning*

Salah satu Teknik *supervised learning* adalah *estimasi*, *prediksi* dan *klasifikasi*. Dapat dilakukan menggunakan beberapa algoritma diantaranya:

1. Estimasi

Linear Regression, Neural Network dan Support Vector Machine

2. Prediksi

Linear Regression, Neural Network dan Support Vector Machine

3. Klasifikasi

Naïve Bayes, k-nearest neighbours, decision tree, neural network dan support vector machine

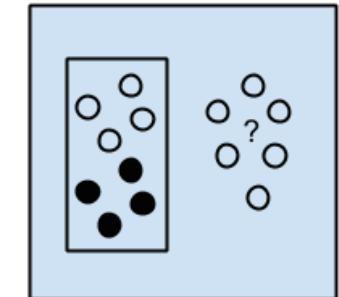
Pertemuan 3

Unsupervised Learning Clustering

Supervised vs Unsupervised

Supervised

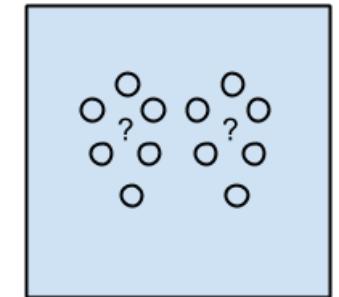
- Memiliki target
- Temukan fungsi yang dapat memetakan data pada targetnya
- Menemukan pola yang menghubungkan atribut dengan targetnya



Supervised Learning
Algorithms

Unsupervised

- Tidak memiliki target
- Menemukan struktur data yang mendasarinya
- Tidak memprediksi secara spesifik, hanya mengelompokan saja



Unsupervised Learning
Algorithms

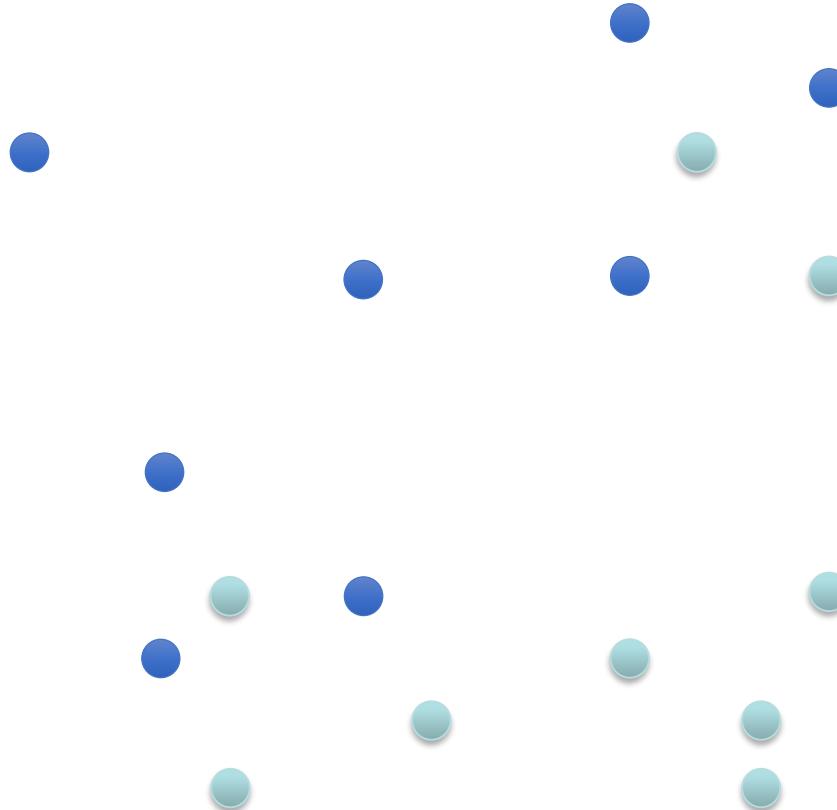
Clustering

- Clustering merupakan proses mengelompokan kumpulan objek menjadi beberapa kelas tertentu berdasarkan kemiripan antara objek-objek tersebut
 - Data pada sebuah kelas (*cluster*) harus berhubungan/mirip
 - Data antar kelas yang berbeda harus tidak saling berhubungan
- Cluster: Kelompok yang berisi data yang mirip
- Analisa Cluster: menemukan kemiripan antara data berdasarkan karakteristik lalu mengelompokkannya kedalam sebuah kelas

Masalah Clustering

- Berapa banyak kelas/*cluster* yang akan dihasilkan?
- Berapa jumlah data pada masing-masing cluster?
- Apakah data pada sebuah cluster memiliki kemiripan?
- Apakah data pada sebuah cluster tidak memiliki kemiripan dengan data pada cluster lain?

Masalah Clustering



Dapat menjadi berapa cluster?

Apakah cluster yang dihasilkan berkualitas?

Bagaimana cara mengelompokan data tersebut?

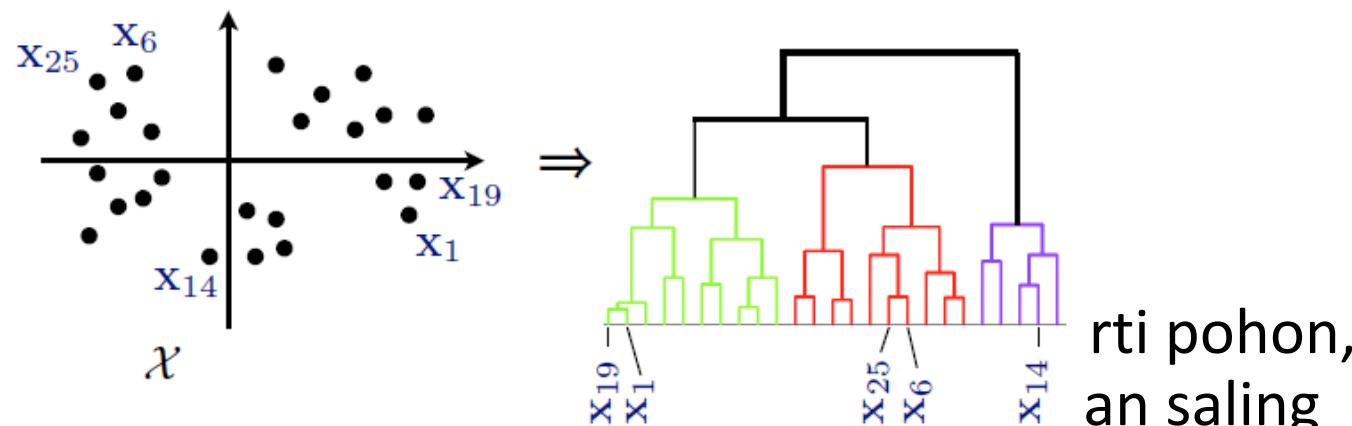
Jenis Clustering

- Hirarkikal (Hierarchical)
 - Objek menjadi lebih terkait dengan objek di dekatnya daripada objek yang lebih jauh
- Partisional (Partitional)
 - Setiap cluster diwakili oleh centroid
 - Ditentukan oleh pengukuran kedekatan objek dengan centroid pada cluster tertentu

Hierarchical Clustering

Objek menjadi lebih terkait dengan objek di dekatnya daripada objek yang lebih jauh

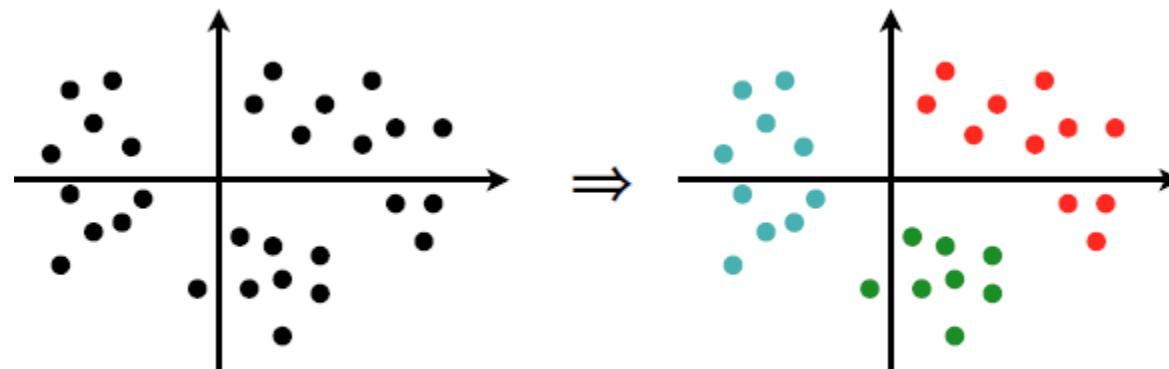
Pada pe
dimana
berdekatan satu sama lain.



rti pohon,
an saling

Partitional Clustering

Setiap cluster diwakili oleh centroid dan diukur berdasarkan pengukuran jarak



Biasanya \mathcal{X} ak
ditentukan sebelumnya/

Algoritma Clustering

1. K-Means
2. Fuzzy C Means
3. Agglomerative
4. K-D Trees
5. EM Clustering
6. Quality Threshold

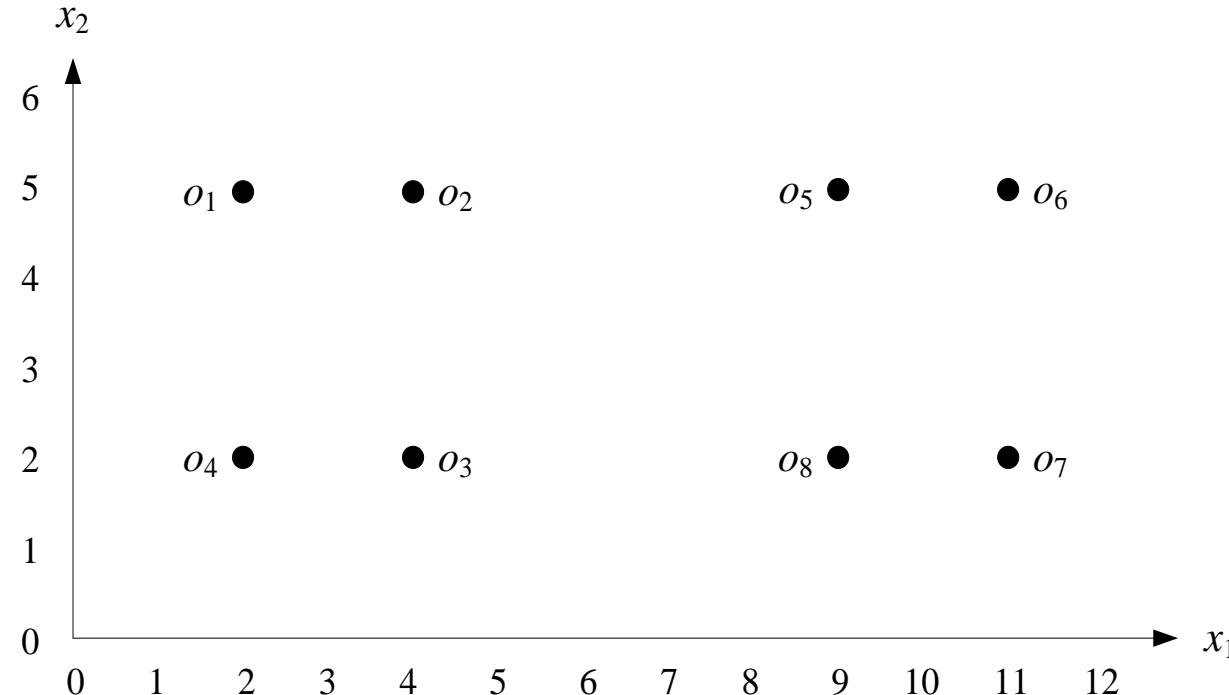
Kegunaan Clustering

1. Menemukan kelas pada dataset yang tidak memiliki target
2. Dimensionality reduction
3. Color-based image segmentation
4. Analisa jejaring media social
5. Segmentasi pasar

Algoritma K-Means

- Algoritma paling sederhana dan paling sering digunakan untuk kasus clustering
- Data dipartisi/dikelompokan menjadi k cluster (k merupakan jumlah cluster yang diinginkan)
- Setiap data pada sebuah cluster mirip dengan centroidnya

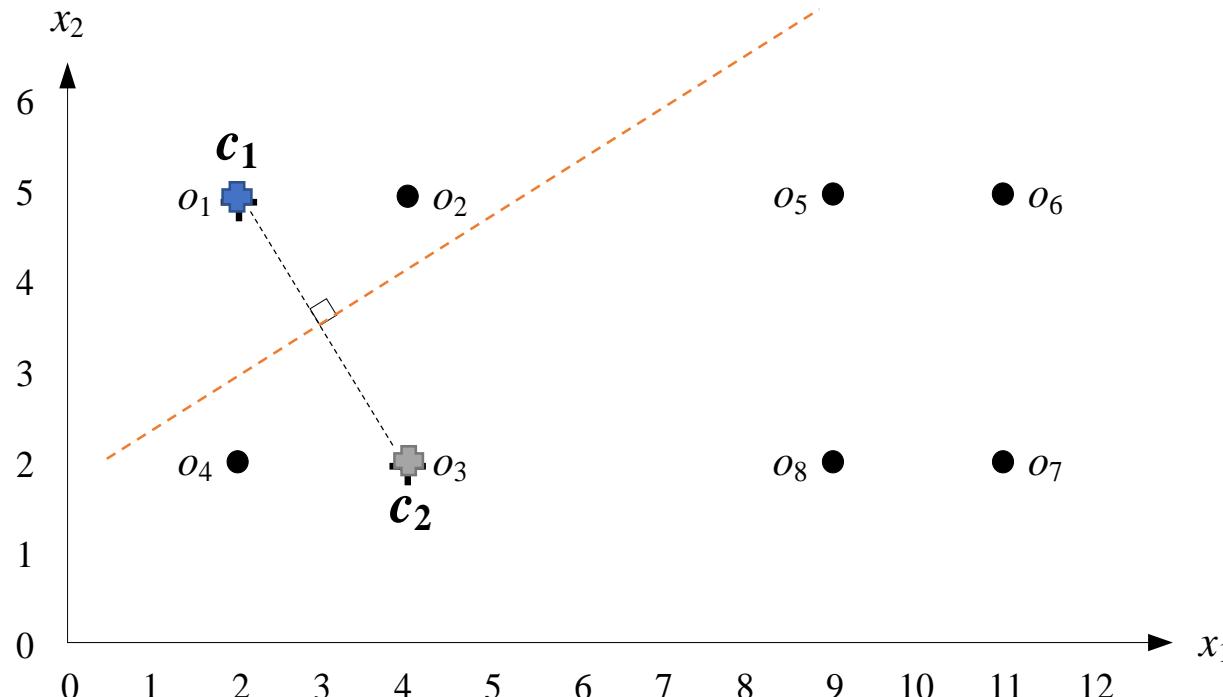
Contoh Kasus K-Means



Jumlah Cluster (k) = 2

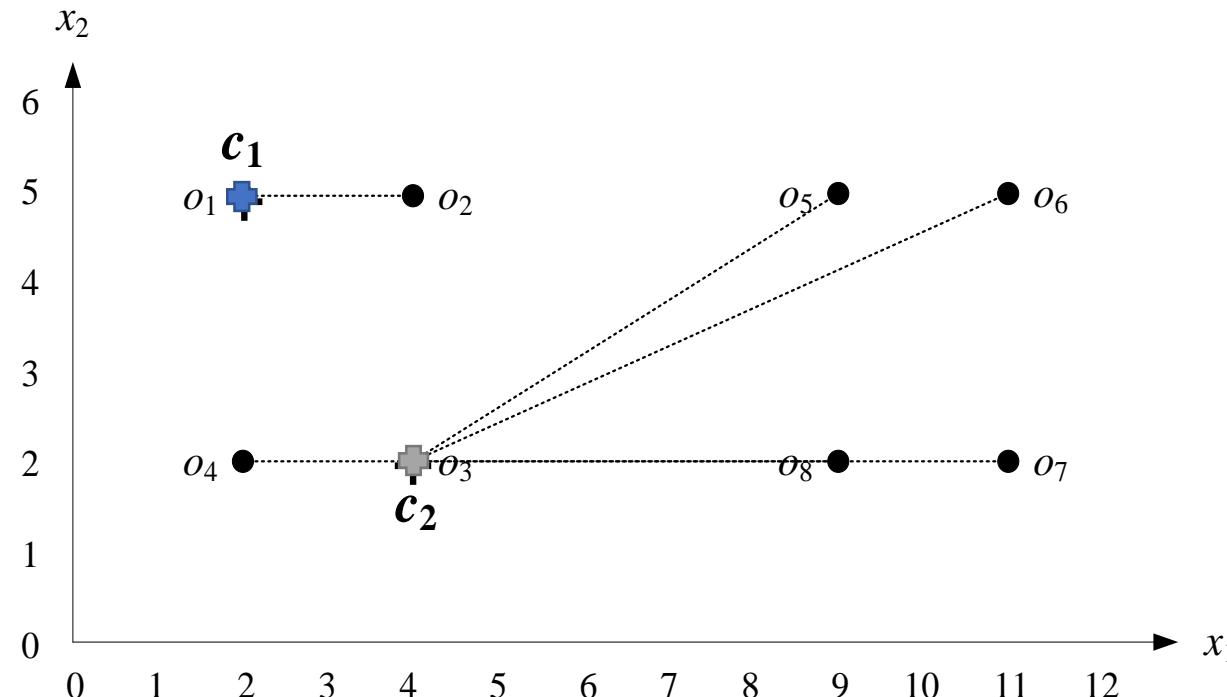
Inisialisasi 2 buah centroid secara acak

Contoh Kasus K-Means



Setiap data point o , temukan centroid c yang paling dekat

Contoh Kasus K-Means

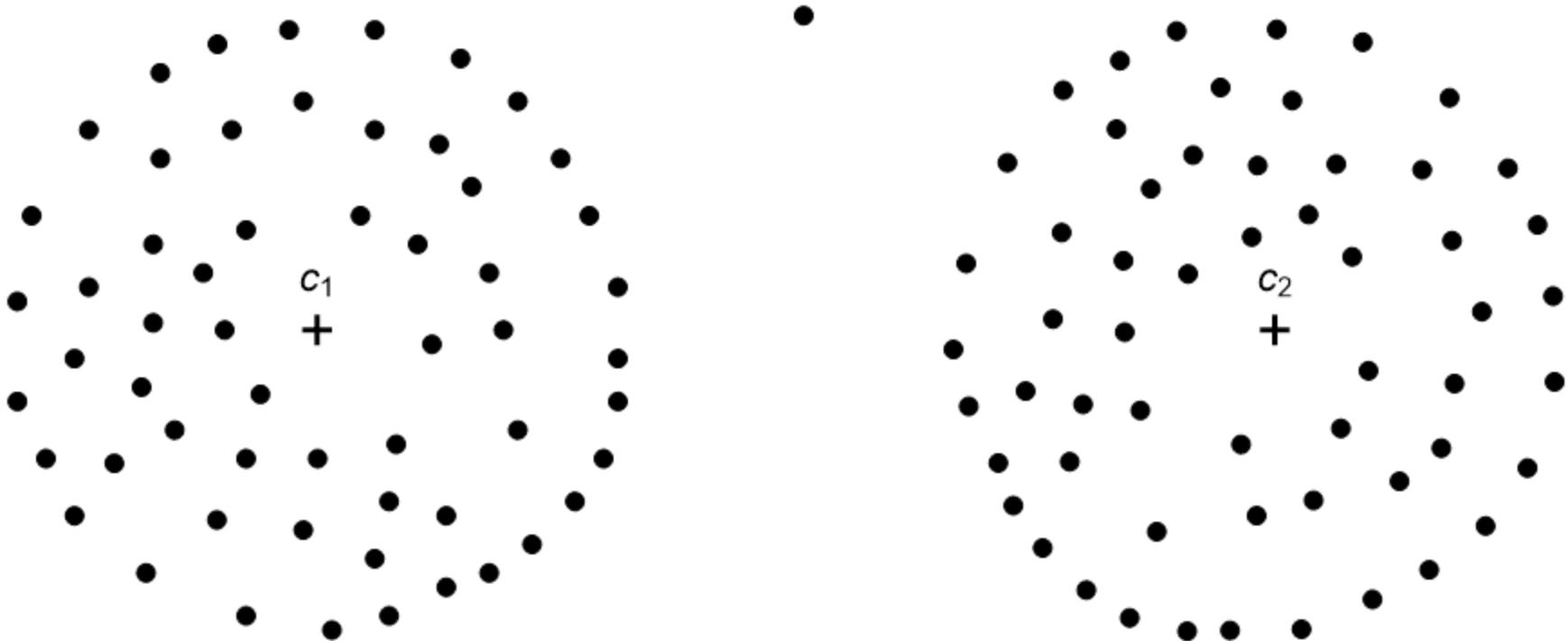


Kalkulasikan rata-rata data dari setiap cluster

Lalu update centroid nya

Masalah k-Means

Data point ini termasuk pada cluster yang mana?

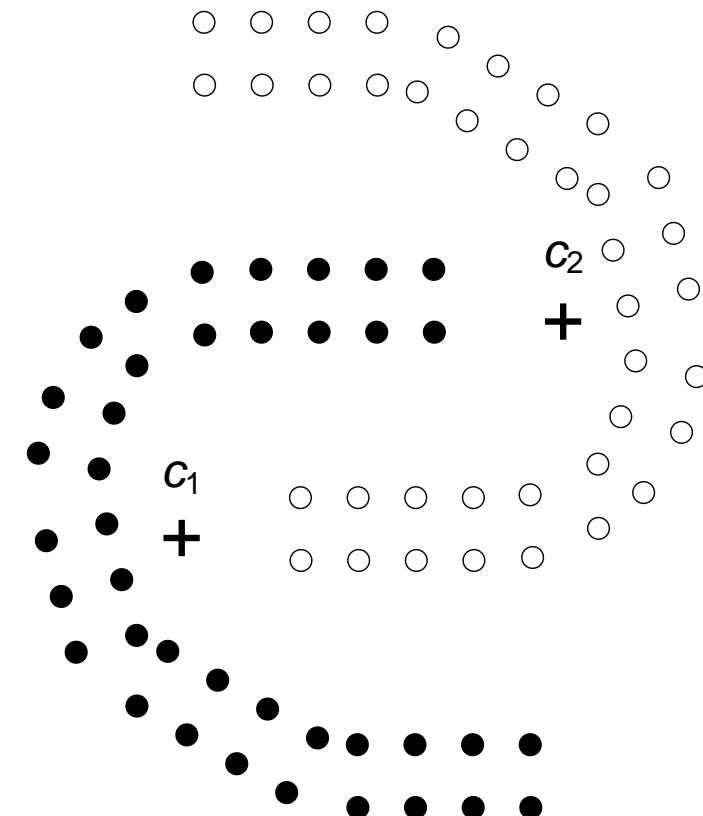


Masalah k-Means

Apakah hasil clustering ini berkualitas?

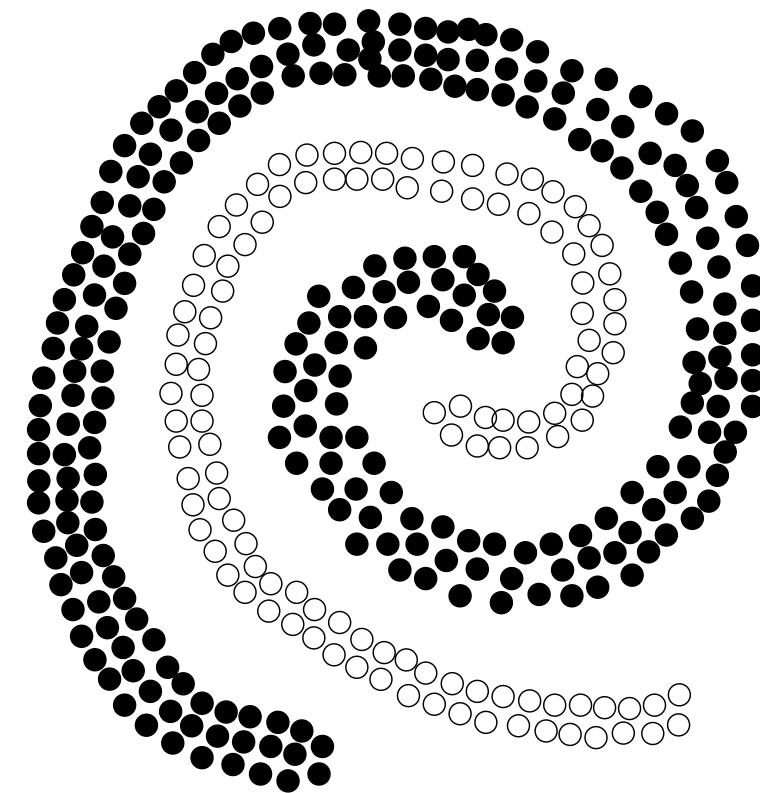
Apakah centroid ini tepat?

Apakah akurat?



Masalah k-Means

Dimana harus meletakan centroidnya?



Kelebihan dan Kekurangan K-Means

- Kelebihan:
 - Relatif sederhana untuk diimplementasikan
 - Lebih efisien dari segi waktu dan biaya komputasi
 - Cocok untuk data yang rapih dan terstruktur
- Kekurangan:
 - Sulit menentukan k
 - Sensitif terhadap penentuan centroid awal
 - Tidak cocok untuk data yang sebarannya terlalu bervariasi

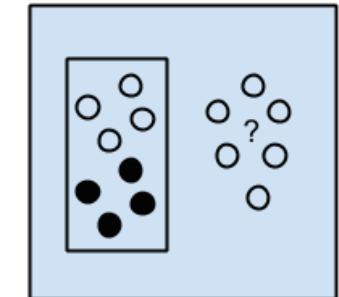
Pertemuan 4

Supervised Learning Klasifikasi dan Regresi

Supervised vs Unsupervised

Supervised

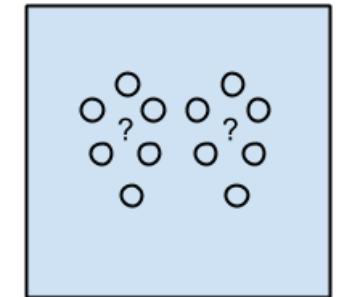
- Memiliki target
- Temukan fungsi yang dapat memetakan data pada targetnya
- Menemukan pola yang menghubungkan atribut dengan targetnya



Supervised Learning
Algorithms

Unsupervised

- Tidak memiliki target
- Menemukan struktur data yang mendasarinya
- Tidak memprediksi secara spesifik, hanya mengelompokan saja



Unsupervised Learning
Algorithms

Klasifikasi

- Klasifikasi adalah menentukan sebuah record data baru ke salah satu dari beberapa kategori (atau klas) yang telah didefinisikan sebelumnya.
- Termasuk kedalam “*supervised learning*”.

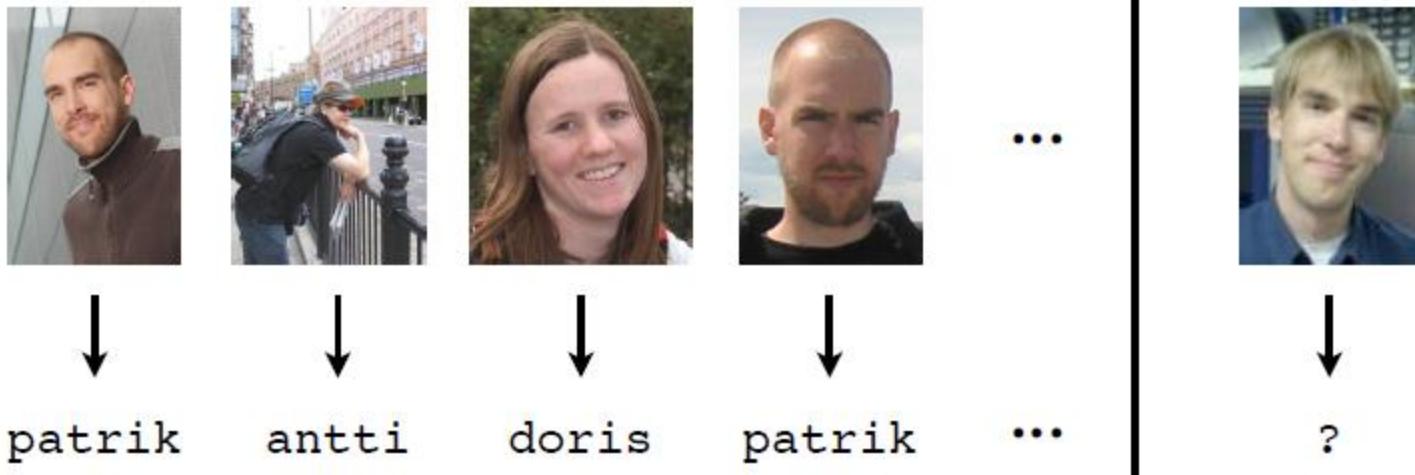
Contoh Klasifikasi Spam Filter

- X adalah kumpulan email
- Y adalah target/kelas apakah { spam atau non-spam }

From: medshop@spam.com Subject: viagra cheap meds...	spam
From: my.professor@helsinki.fi Subject: important information here's how to ace the test...	non-spam
:	:
From: mike@example.org Subject: you need to see this how to win \$1,000,000...	?

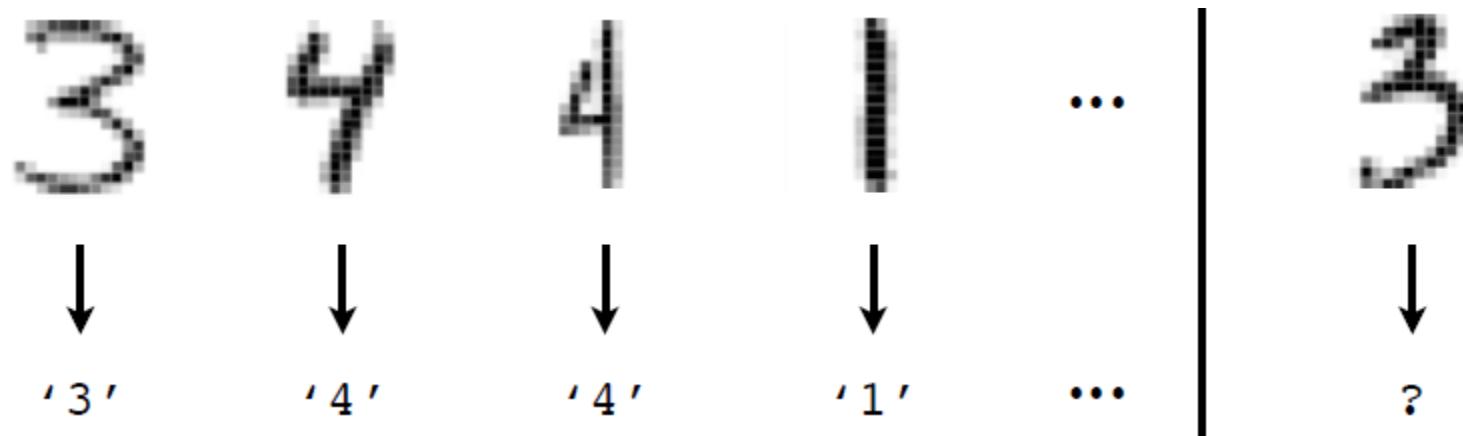
Contoh Klasifikasi Face Recognition

- X adalah kumpulan gambar foto wajah
- Y adalah target/kelas



Contoh Klasifikasi Pengenalan Tulisan Tangan

- X adalah kumpulan gambar tulisan tangan berisi angka
- Y adalah target/kelas berupa angka { 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 }



Contoh Klasifikasi

Klasifikasi Kelulusan Mahasiswa

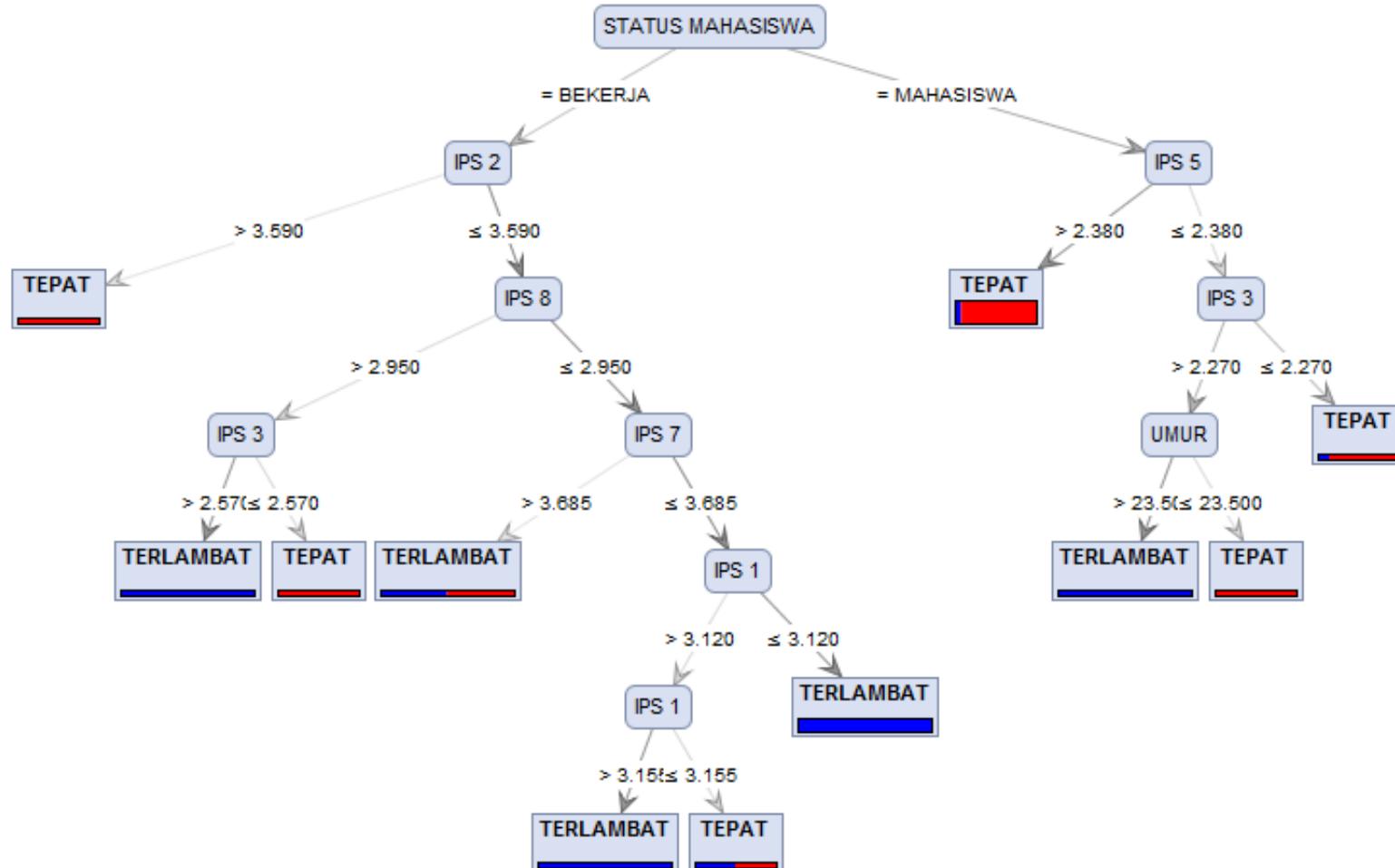
Label
↓

NIM	Gender	Nilai UN	Asal Sekolah	IPS1	IPS2	IPS3	IPS 4	...	Lulus Tepat Waktu
10001	L	28	SMAN 2	3.3	3.6	2.89	2.9		Ya
10002	P	27	SMA DK	4.0	3.2	3.8	3.7		Tidak
10003	P	24	SMAN 1	2.7	3.4	4.0	3.5		Tidak
10004	L	26.4	SMAN 3	3.2	2.7	3.6	3.4		Ya
...									
...									
11000	L	23.4	SMAN 5	3.3	2.8	3.1	3.2		Ya

■ Pembelajaran dengan
■ Metode Klasifikasi (C4.5)

Klasifikasi Kelulusan Mahasiswa

Hasil Model Klasifikasi Menggunakan Decision Tree



Regresi

- Memprediksi nilai dari suatu variabel kontinyu yang diberikan berdasarkan nilai dari variabel yang lain, dengan mengasumsikan sebuah model ketergantungan linier atau nonlinier
- Termasuk kedalam "supervised learning"

Contoh Regresi

- Memprediksi jumlah penjualan produk baru berdasarkan pada belanja promosi/iklan
- Memprediksi kecepatan angin sebagai suatu fungsi suhu, kelembaban, tekanan udara, dsb.
- *Time series prediction dari indeks stock market*

Simulasi Regresi (Estimasi)

Customer	Jumlah Pesanan (P)	Jumlah Traffic Light (T)	Jarak (J)	Waktu Tempuh (T)
1	3	3	3	16
2	1	7	4	20
3	2	4	6	18
4	4	6	8	36
...				
1000	2	4	2	12

Pembelajaran dengan
Metode Estimasi (*Regresi Linier*)

$$\text{Waktu Tempuh (T)} = 0.48P + 0.23T + 0.5J$$

Pengetahuan

Simulasi Regresi (Prediksi)

Label



Row No.	Close	Date	Open	High	Low	Volume
1	1286.570	Apr 11, 2006	1296.600	1300.710	1282.960	2232880000
2	1288.120	Apr 12, 2006	1286.570	1290.930	1286.450	1938100000
3	1289.120	Apr 13, 2006	1288.120	1292.090	1283.370	1891940000
4	1285.330	Apr 17, 2006	1289.120	1292.450	1280.740	1794650000
5	1307.280	Apr 18, 2006	1285.330	1309.020	1285.330	2595440000
6	1309.930	Apr 19, 2006	1307.650	1310.390	1302.790	2447310000
7	1311.460	Apr 20, 2006	1309.930	1318.160	1306.380	2512920000
8	1311.280	Apr 21, 2006	1311.460	1317.670	1306.590	2392630000
9	1308.110	Apr 24, 2006	1311.280	1311.280	1303.790	2117330000
10	1301.740	Apr 25, 2006	1308.110	1310.790	1299.170	2366380000
11	1305.410	Apr 26, 2006	1301.740	1310.970	1301.740	2502690000
12	1309.720	Apr 27, 2006	1305.410	1315	1295.570	2772010000
13	1310.610	Apr 28, 2006	1309.720	1316.040	1306.160	2419920000

Dataset harga saham dalam bentuk **time series** (rentet waktu)

Pembelajaran dengan
Metode Prediksi (*Neural Network*)



Pertemuan 5

Evaluasi dan Validasi

Evaluasi

- Evaluasi merupakan kegiatan yang dilakukan untuk menentukan nilai dari suatu hal, dalam hal ini yang dinilai adalah kinerja metode pembelajaran mesin
- Tujuan dilakukan evaluasi adalah untuk menganalisa hasil kinerja metode pembelajaran mesin

Kriteria Evaluasi

- Secara umum pengukuran model data mining mengacu kepada tiga kriteria: Akurasi (Accuracy), Kehandalan(Reliability) dan Kegunaan (Usefulness)
- Keseimbangan diantaranya ketiganya diperlukan karena belum tentu model yang akurat adalah handal, dan yang handal atau akurat belum tentu berguna

Kriteria Evaluasi

1. **Akurasi** adalah ukuran dari seberapa baik model mengkorelasikan antara hasil dengan atribut dalam data yang telah disediakan. Terdapat berbagai model akurasi, tetapi semua model akurasi tergantung pada data yang digunakan
2. **Kehandalan** adalah ukuran di mana model data mining diterapkan pada dataset yang berbeda akan menghasilkan sebuah model data mining dapat diandalkan jika menghasilkan pola umum sama terlepas dari data testing yang disediakan
3. **Kegunaan** mencakup berbagai metrik yang mengukur apakah model tersebut memberikan informasi yang berguna.

Evaluasi Berdasarkan Metode Pembelajaran Mesin

1. Estimation:
 - **Error:** Root Mean Square Error (RMSE), MSE, MAPE, etc
2. Prediction/Forecasting (Prediksi/Peramalan):
 - **Error:** Root Mean Square Error (RMSE) , MSE, MAPE, etc
3. Classification:
 - **Confusion Matrix:** Accuracy
 - **ROC Curve:** Area Under Curve (AUC)
4. Clustering:
 - **Internal Evaluation:** Davies–Bouldin index, Dunn index,
 - **External Evaluation:** Rand measure, F-measure, Jaccard index, Fowlkes–Mallows index, Confusion matrix

Root Mean Square Error

- Root Mean Square Error adalah metode alternatif untuk mengevaluasi Teknik peramalan yang digunakan untuk mengukur tingkat akurasi hasil perkiraan suatu model.
- RMSE merupakan nilai rata-rata dari jumlah kuadrat kesalahan.
- Semakin rendah RMSE semakin baik kinerja metode pembelajaran mesin

Confussion Matrix

- Confussion Matrix merupakan salah satu metode yang dapat digunakan untuk mengukur kinerja suatu metode klasifikasi.
- Pada dasarnya confusion matrix mengandung informasi yang membandingkan hasil klasifikasi yang dilakukan oleh sistem dengan hasil klasifikasi yang seharusnya.

Confusion Matrix (2)

		Actual Class		
		+	-	
Predicted Class	+	105 TP	12 FP	Predicted as positive = 117
	-	3 FN	53 TN	Predicted as negative = 56
		Positive data= 108	Negative data= 65	Total = 173

- Akurasi = $\frac{TP+TN}{TP+TN+FP+FN}$
- Rasio kesalahan = 1 - akurasi

Confusion Matrix (3)

- True Positive Rate = $\frac{TP}{TP+FN}$
 - Recall
 - Sensitivity
- True Negative Rate = $\frac{TN}{TN+FP}$
 - Specificity

Evaluasi Clustering

- Evaluasi Internal

- Ketika hasil pengelompokan dievaluasi berdasarkan data yang mengelompok itu sendiri
- Metode-metode ini biasanya menetapkan skor terbaik untuk algoritma yang menghasilkan cluster dengan kesamaan tinggi dalam sebuah cluster dan kesamaan rendah antara kluster
- Evaluasi Internal dapat menggunakan Davies Bouldin Index atau Dunn Index

Evaluasi Clustering (2)

- Evaluasi Eksternal

- hasil pengelompokan dievaluasi berdasarkan data yang tidak digunakan untuk pengelompokan, seperti label kelas yang dikenal dan tolok ukur eksternal.
- Tolak ukur tersebut terdiri dari seperangkat item pradiklasifikasikan, dan set ini sering dibuat oleh (ahli) manusia.
- Evaluasi eksternal dapat menggunakan confusion matrix

Validasi

- Validasi dalam pembelajaran mesin adalah validasi data untuk memastikan bahwa program beroperasi pada data yang benar.

Validasi

- Pembagian dataset:
 - Dua subset: **data training** dan **data testing**
 - Tiga subset: **data training**, **data validation** dan **data testing**
- Data **training** untuk pembentukan model, dan data **testing** digunakan untuk pengujian model
- Data **validation** untuk memvalidasi model kita valid atau tidak

Cross Validation

- Metode cross-validation digunakan untuk **menghindari overlapping** pada data testing
- Tahapan cross-validation:
 1. Bagi data menjadi **k subset** yg berukuran sama
 2. Gunakan **setiap subset untuk data testing** dan sisanya untuk data training
- Disebut juga dengan **k-fold cross-validation**
- Seringkali subset dibuat stratified (bertingkat) sebelum cross-validation dilakukan, karena **stratifikasi akan mengurangi variansi** dari estimasi

Cross Validation

- Metode evaluasi standard: **stratified 10-fold cross-validation**
- Mengapa **10**? Hasil dari berbagai percobaan yang ekstensif dan pembuktian teoritis, menunjukkan bahwa **10-fold cross-validation adalah pilihan terbaik** untuk mendapatkan hasil validasi yang akurat
- 10-fold cross-validation akan **mengulang pengujian sebanyak 10 kali** dan **hasil pengukuran adalah nilai rata-rata dari 10 kali pengujian**

Cross Validation

Testing	Dataset									
1		Orange								
2			Orange							
3				Orange						
4					Orange					
5						Orange				
6							Orange			
7								Orange		
8									Orange	
9										Orange
10										Orange

Pertemuan 6

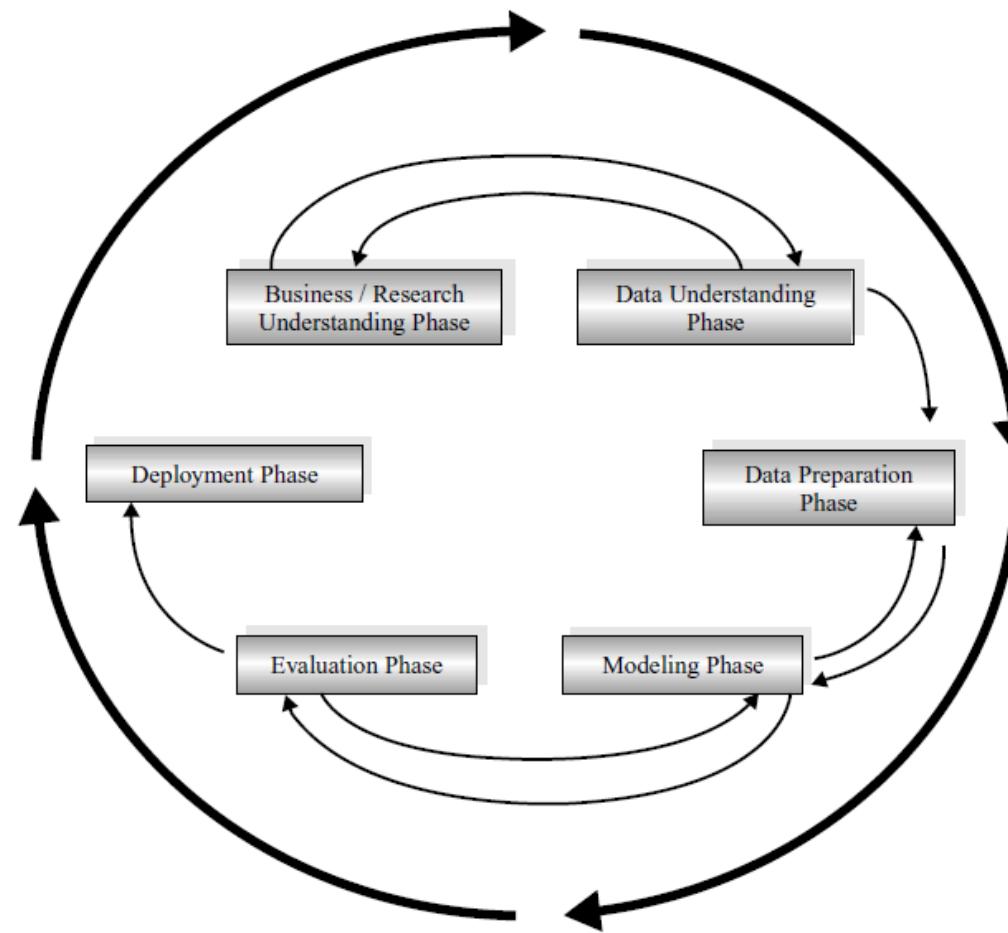
PENELITIAN

PEMBELAJARAN MESIN

Standard Proses Penelitian

- Standar lintas industri jelas diperlukan yaitu industri, alat, dan aplikasi
- Proses Standar Lintas Industri untuk Data Mining (CRISP-DM) dikembangkan pada tahun 1996
- CRISP-DM menyediakan proses standar yang tidak tersedia dan tersedia secara bebas untuk memasukkan data mining ke dalam strategi pemecahan masalah umum dari unit bisnis atau riset

Standard Proses Penelitian



Business Understanding Phase

- Mengumumkan tujuan dan persyaratan proyek dengan jelas dalam hal bisnis atau unit penelitian secara keseluruhan
- Menerjemahkan tujuan dan pembatasan ini ke dalam rumusan definisi masalah penambangan data
- Siapkan strategi awal untuk mencapai tujuan ini

Data Understanding Phase

- Kumpulkan datanya
- Gunakan analisis data eksplorasi untuk membiasakan diri dengan data dan menemukan wawasan awal
- Evaluasi kualitas data
- Jika diinginkan, pilih subset menarik yang mungkin berisi pola yang dapat ditindaklanjuti

Data Preparation Phase

- Siapkan dari data mentah awal set data akhir yang akan digunakan untuk semua fase selanjutnya. Fase ini sangat padat karya
- Pilih kasus dan variabel yang ingin Anda analisis dan yang sesuai untuk analisis Anda
- Lakukan transformasi pada variabel tertentu, jika diperlukan
- Bersihkan data mentah sehingga siap untuk alat pemodelan

Modeling Phase

- Pilih dan terapkan teknik pemodelan yang sesuai
- Kalibrasi pengaturan model untuk mengoptimalkan hasil
- Beberapa teknik yang berbeda dapat digunakan untuk masalah penambangan data yang sama
- Jika perlu, putar kembali ke fase persiapan data untuk membawa bentuk data ke sejalan dengan persyaratan khusus dari teknik penambangan data tertentu

Evaluation Phase

- Evaluasi satu atau lebih model yang disampaikan dalam fase pemodelan untuk kualitas dan efektivitas sebelum menerapkannya untuk digunakan di lapangan
- Tentukan apakah model tersebut benar-benar mencapai tujuan yang ditetapkan untuknya di fase pertama
- Menetapkan apakah beberapa aspek penting dari bisnis atau masalah penelitian belum cukup dipertanggungjawabkan
- Datangkan ke keputusan tentang penggunaan hasil penambangan data

Deployment Phase

- Memanfaatkan model yang dibuat: Penciptaan model tidak menandakan penyelesaian proyek
- Contoh penerapan sederhana: Buat laporan
- Contoh penerapan yang lebih kompleks: Menerapkan proses penambangan data paralel di departemen lain
- Untuk bisnis, pelanggan sering melakukan penerapan berdasarkan model Anda

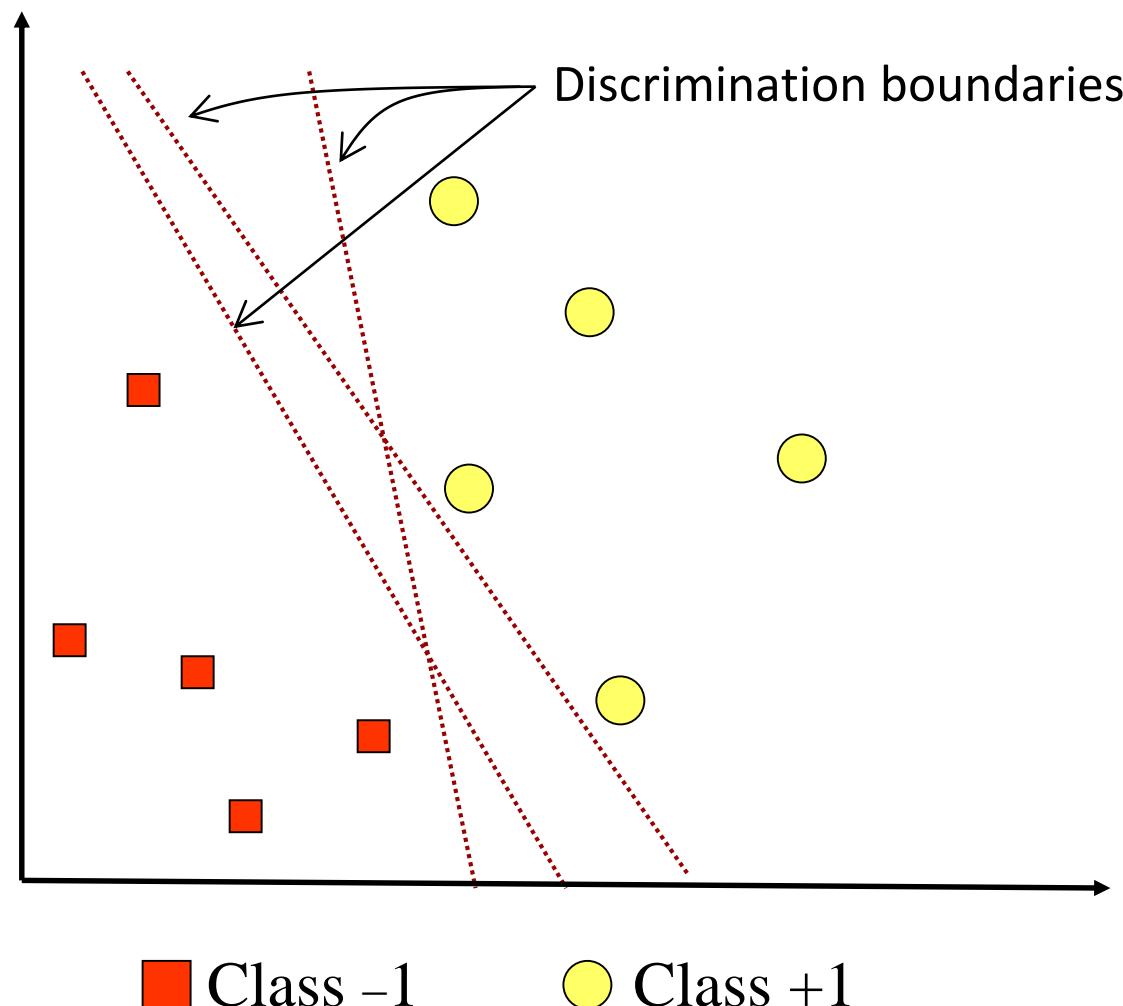
PERTEMUAN 9

Support Vector Machine

Support Vector Machine

- Diperkenalkan oleh Vapnik (1992)
- Support Vector Machine memenuhi 3 syarat utama sebuah metode PR
 - Robustness
 - Theoretically Analysis
 - Feasibility
- Pada prinsipnya bekerja sebagai binary classifier. Saat ini tengah dikembangkan untuk multiclass problem
- Structural-Risk Minimization

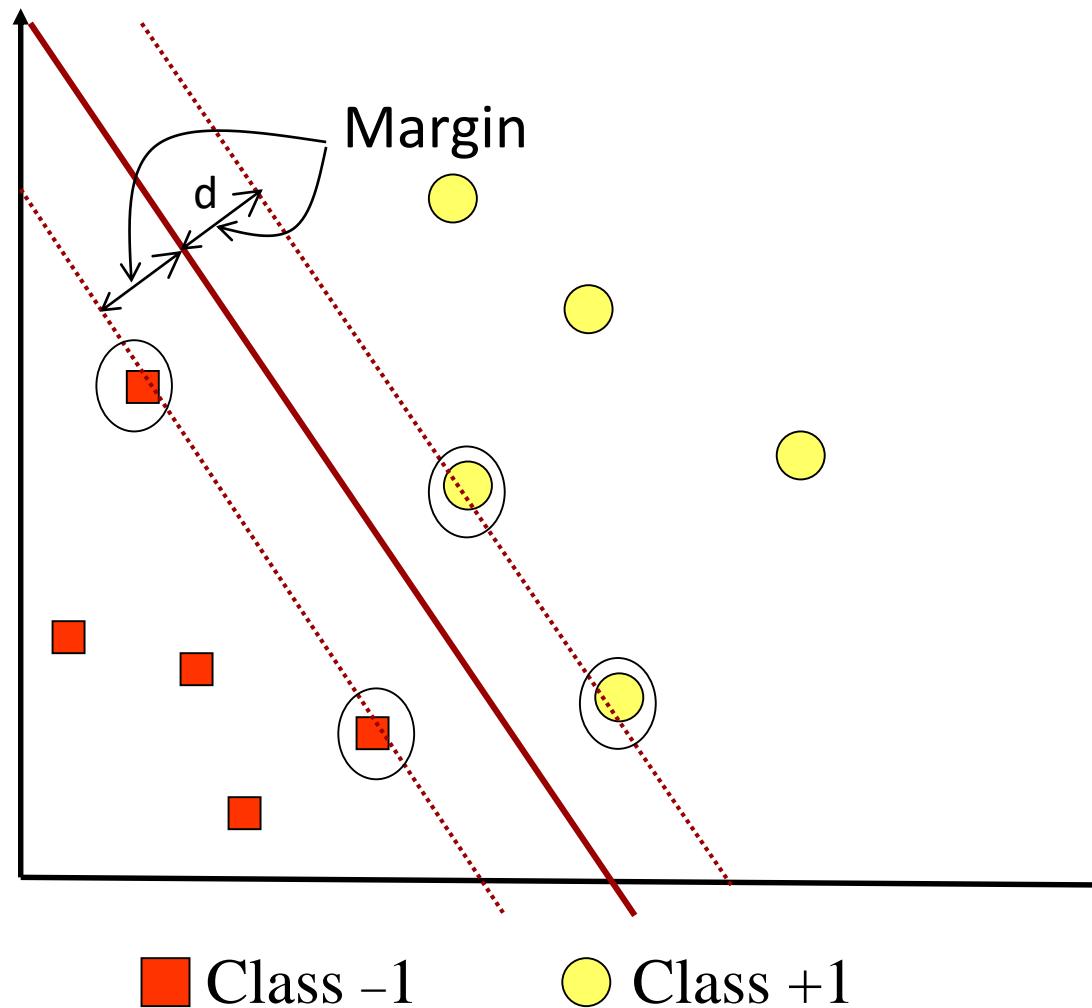
Binary Classification



Optimal Hyperplane by SVM

- Margin (d) = minimum distance antara hyperplane and training samples
- Hyperplane yang paling baik diperoleh dengan memaksimalkan nilai margin
- Hyperplane yang paling baik itu akan melewati pertengahan antara kedua class
- Sample yang paling dekat lokasinya terhadap hyperplane disebut support vector
- Proses learning dalam SVM : mencari support vector untuk memperoleh hyperplane yang terbaik

Optimal Hyperplane by SVM



Kernel & Non-Linear SVM

- Latar belakang
- Kelemahan Linear Learning-Machines
- Representasi data & Kernel
- Non linear SVM

Latar belakang

- Machine Learning
 - Supervised learning: berikan satu set input-output data, dan buatlah satu model yang mampu memprediksi dengan benar output terhadap data baru. Contoh : pattern classification, regression
 - Unsupervised learning: berikan satu set data (tanpa output yang bersesuaian), dan ekstraklah suatu informasi bermanfaat. Contoh : clustering, Principal Component Analysis
- Apabila banyaknya data yang diberikan “cukup banyak”, metode apapun yang dipakai akan menghasilkan model yang bagus
- Tetapi jika data yang diberikan sangat terbatas, untuk mendapatkan performa yang baik, mutlak perlu memakai informasi spesifik masalah yang dipecahkan (prior knowledge of the problem domain). Contoh : masalah yg dipecahkan apakah berupa character recognition, analisa sekuens DNA, voice dsb. Prior knowledge seperti “masalah yg dianalisa adalah DNA” ini tidak dapat dinyatakan dengan angka.

- Pemanfaatan prior knowledge :
 - Fungsi Kernel (kemiripan sepasang data)
 - Probabilistic model of data distribution (Gaussian, Markov model, HMM, dsb)
- Pemakaian Kernel :

user memanfaatkan pengetahuannya mengenai domain masalah yang dipecahkan dengan mendefinisikan fungsi kernel untuk mengukur kemiripan sepasang data

Linear Learning Machines

- Kelebihan :
 - Algoritma pembelajarannya simple dan mudah dianalisa secara matematis
- Kelemahan
 - Perceptron (salah satu contoh linear learning machine) hanya mampu memecahkan problem klasifikasi linear (Minsky & Papert)
 - Umumnya masalah dari *real-world domain* bersifat non-linear dan kompleks, sehingga linear learning machines tidak mampu dipakai memecahkan masalah riil.

Representasi Data & Kernel

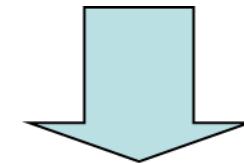
- Representasi data seringkali mampu menyederhanakan satu masalah
- Data yang dipetakan ke ruang vektor berdimensi lebih tinggi, memiliki potensi lebih besar untuk dapat dipisahkan secara linear (Cover theorem)
- Masalah : semakin tinggi dimensi suatu data, akan mengakibatkan tertimpa kutukan dimensi tinggi Curse of dimensionality.
 - turunnya generalisasi model
 - meningkatnya komputasi yang diperlukan
- Pemakaian konsep Kernel akan mengatasi masalah di atas

Representasi Data & Kernel

- Representasi data seringkali mampu menyederhanakan satu masalah
- Formula sebagaimana pada persamaan (20) tidak dapat dipecahkan dengan linear machines
- Representasi dengan menghasilkan (21) yang berupa persamaan linear, sehingga bisa dipecahkan dengan linear machines

Newton's law gravitation

$$f(m_1, m_2, r) = C \frac{m_1 m_2}{r^2} \quad (20)$$



$$\begin{aligned} g(x, y, z) &= \ln f(m_1, m_2, r) \\ &= \ln C + \ln m_1 + \ln m_2 - 2 \ln r \\ &= c + x + y - 2z \end{aligned} \quad (21)$$

Representasi Data & Kernel

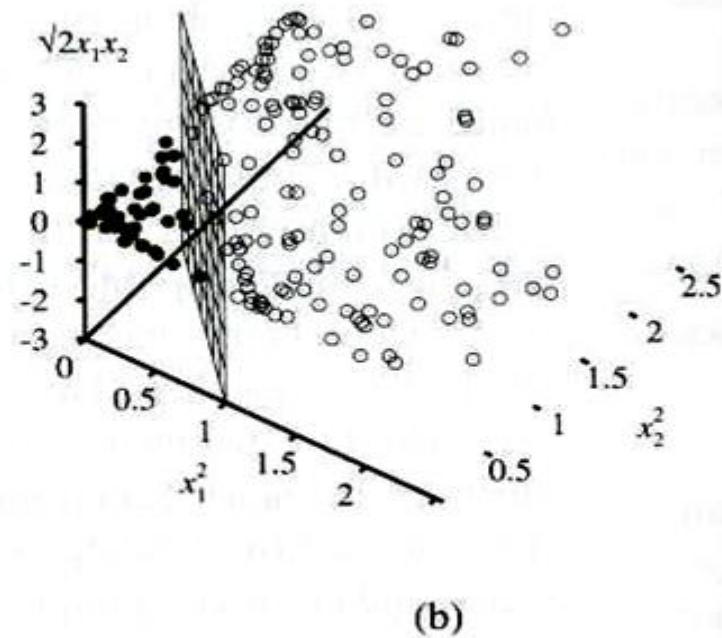
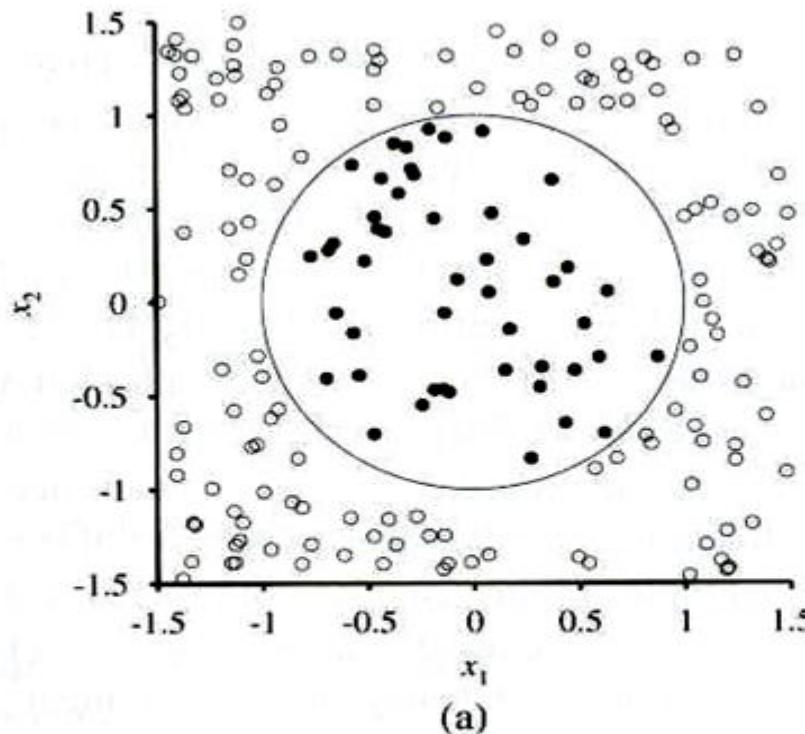
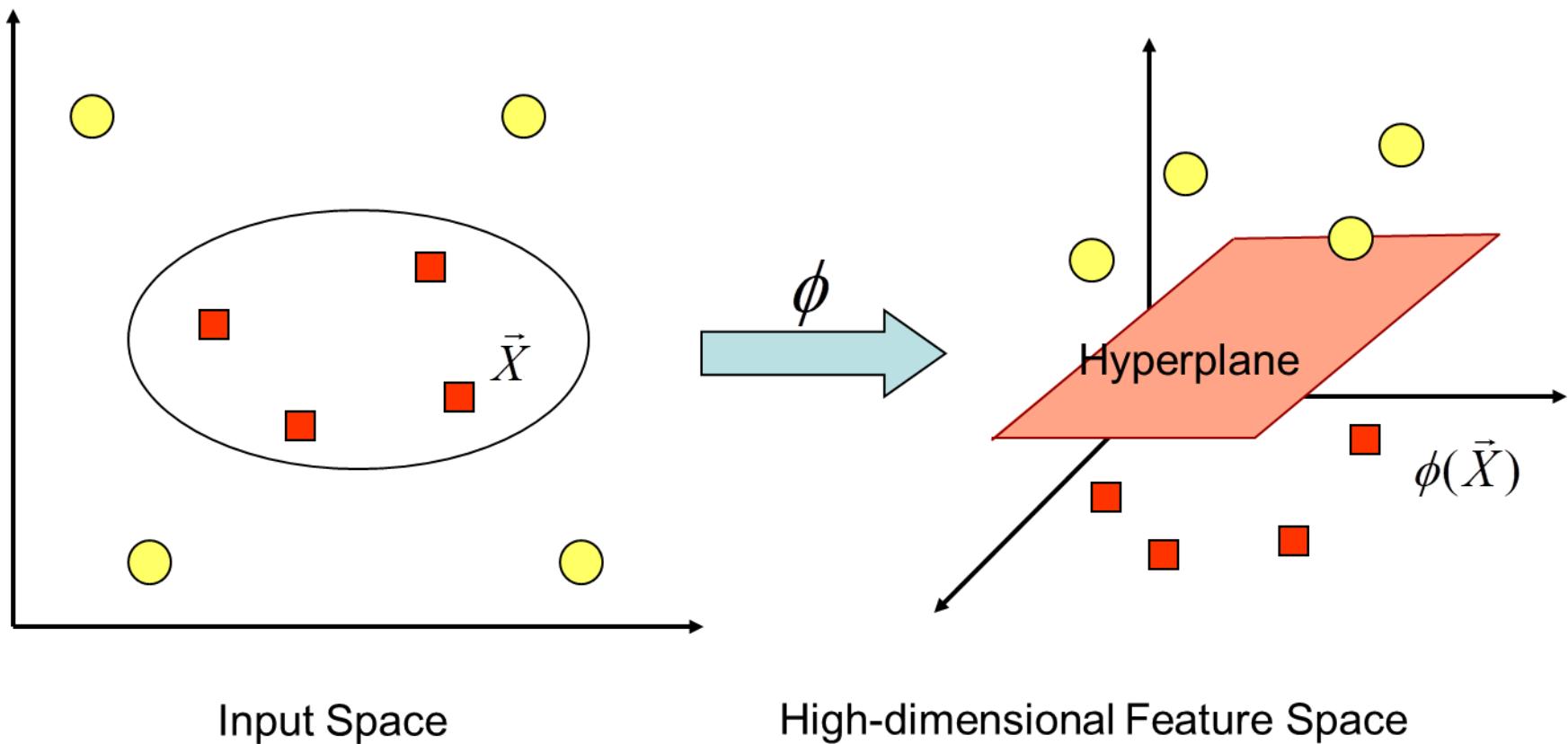


Figure 20.27 (a) A two-dimensional training with positive examples as black circles and negative examples as white circles. The true decision boundary, $x_1^2 + x_2^2 \leq 1$, is also shown. (b) The same data after mapping into a three-dimensional input space $(x_1^2, x_2^2, \sqrt{2}x_1x_2)$. The circular decision boundary in (a) becomes a linear decision boundary in three dimensions.

Non Linear Classification dalam SVM

Non Linear Classification dalam SVM



- Linear learning machines dapat ditulis dalam dua bentuk: primal form & dual form

$$\vec{w} = \sum_{i=1}^l \alpha_i y_i \vec{x}_i$$

primal $f(\vec{x}) = \langle \vec{w} \cdot \vec{\phi}(x) \rangle + b = \sum_{i=1}^{\dim} w_i \phi_i(x) + b$

dual $f(\vec{x}) = \sum_{i=1}^l \alpha_i y_i \langle \phi(\vec{x}_i) \cdot \phi(\vec{x}) \rangle + b$

- Hypotheses function dapat direpresentasikan sebagai kombinasi linear training points. Sehingga decision rule dapat dievaluasi berdasarkan inner product (dot product) antara test point & training points
- Keuntungan dual form : dimensi feature space tidak mempengaruhi perhitungan. Informasi yang dipakai hanya Gram matrix

Gram Matrix

$$G = (\langle \vec{x}_i \cdot \vec{x}_j \rangle)_{i,j=1}^l$$

$$\begin{pmatrix} \langle \vec{x}_1 \cdot \vec{x}_1 \rangle & \dots & \langle \vec{x}_l \cdot \vec{x}_1 \rangle \\ \vdots & \ddots & \vdots \\ \langle \vec{x}_1 \cdot \vec{x}_l \rangle & \dots & \langle \vec{x}_l \cdot \vec{x}_l \rangle \end{pmatrix}$$

Fungsi Kernel

Representasi dual form

$$f(\vec{x}) = \sum_{i=1}^l \alpha_i y_i \langle \phi(\vec{x}_i) \cdot \phi(\vec{x}) \rangle + b$$



$$\langle \phi(\vec{x}_i) \cdot \phi(\vec{x}) \rangle$$

Bisa dihitung secara IMPLISIT. Yaitu tidak perlu mengetahui wujud fungsi pemetaan melainkan langsung menghitungnya lewat fungsi KERNEL

$$\langle \phi(\vec{x}_i) \cdot \phi(\vec{x}) \rangle = K(\vec{x}_i, \vec{x})$$

Contoh-contoh Fungsi Kernel

Polynomial

$$K(\vec{x}, \vec{y}) = \langle \vec{x} \cdot \vec{y} \rangle^d$$

Gaussian

$$K(\vec{x}, \vec{y}) = \exp\left(-\frac{\|\vec{x} - \vec{y}\|^2}{2\sigma^2}\right)$$

where $\sigma > 0$

Sigmoid

$$K(\vec{x}, \vec{y}) = \tanh(\kappa \langle \vec{x} \cdot \vec{y} \rangle + \vartheta)$$

where and $\vartheta < 0$

Representasi Data & Kernel

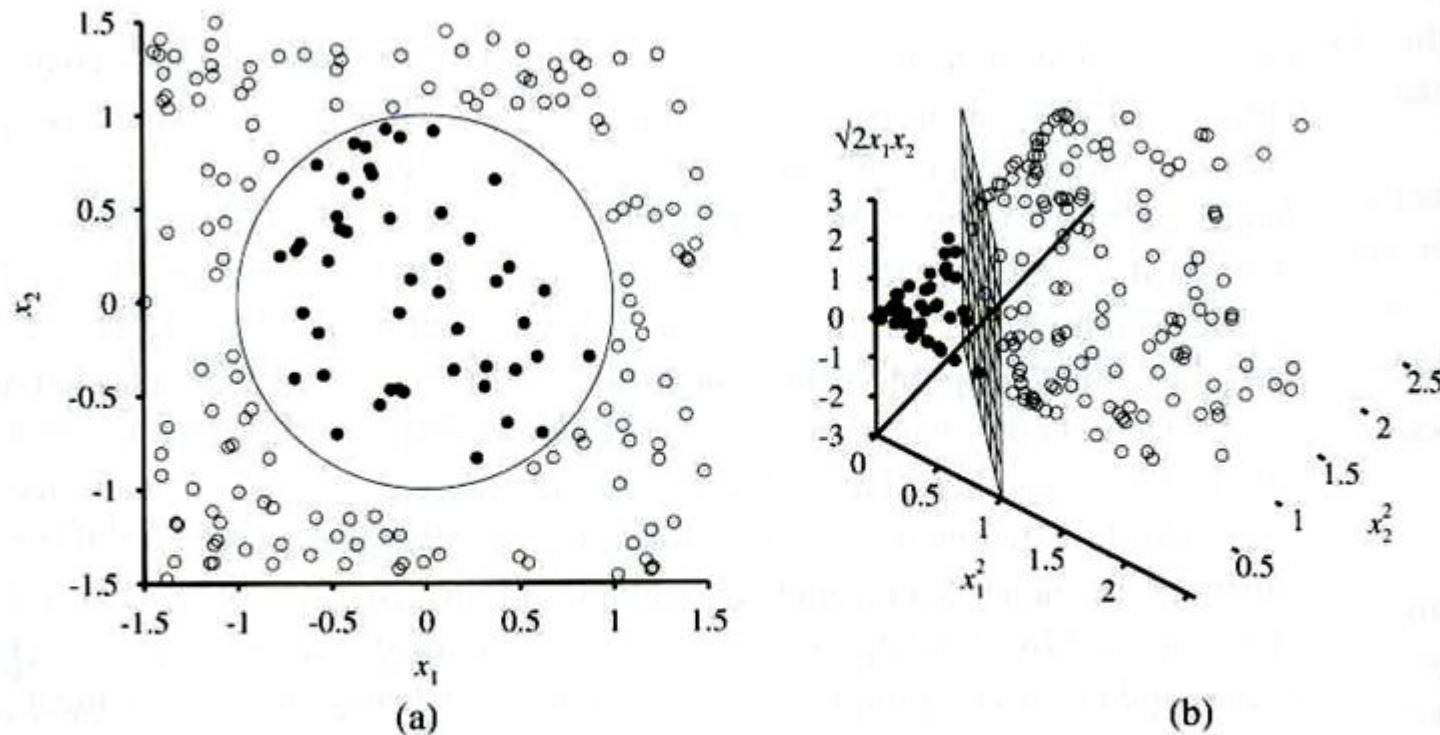


Figure 20.27 (a) A two-dimensional training with positive examples as black circles and negative examples as white circles. The true decision boundary, $x_1^2 + x_2^2 \leq 1$, is also shown. (b) The same data after mapping into a three-dimensional input space ($x_1^2, x_2^2, \sqrt{2}x_1x_2$). The circular decision boundary in (a) becomes a linear decision boundary in three dimensions.

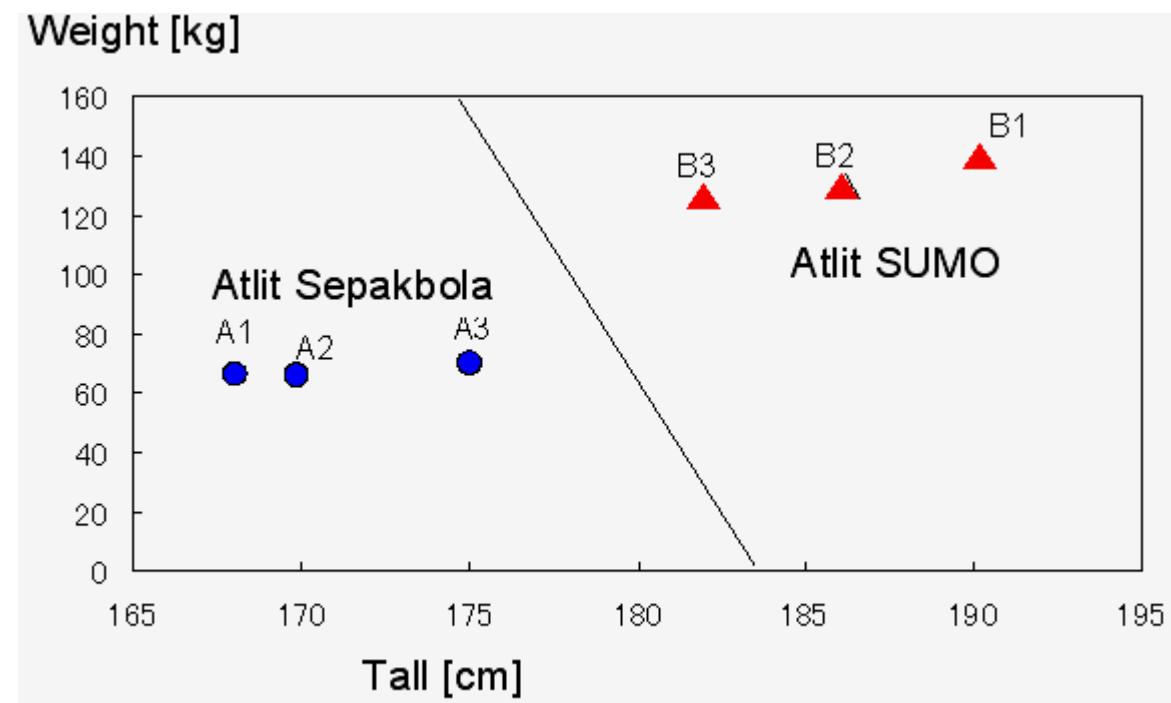
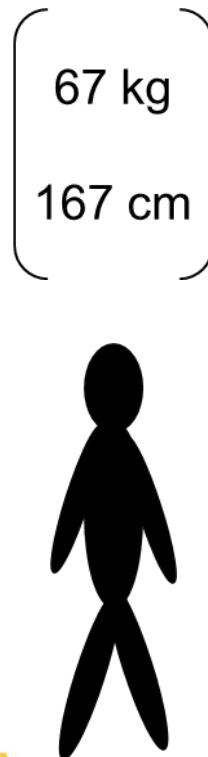
Representasi Data & Kernel

$$(x_1, x_2) \mapsto (x_1^2, x_2^2, \sqrt{2}x_1x_2)$$

$$\begin{aligned}<\phi(\vec{x}) \cdot \phi(\vec{y})> &= (x_1^2, x_2^2, \sqrt{2}x_1x_2)(y_1^2, y_2^2, \sqrt{2}y_1y_2)^T \\ &= ((x_1, x_2)(y_1, y_2)^T)^2 \\ &= (<\vec{x} \cdot \vec{y}>)^2 \\ &=: K(\vec{x}, \vec{y})\end{aligned}$$

Representasi Data & Kernel

- Umumnya data direpresentasikan secara individual. Misalnya, untuk membedakan atlit Sumo dan atlit sepakbola, bisa dengan mengukur berat badan dan tinggi mereka



Representasi Data & Kernel

- Metode Kernel : data tidak direpresentasikan secara individual, melainkan lewat perbandingan antara sepasang data

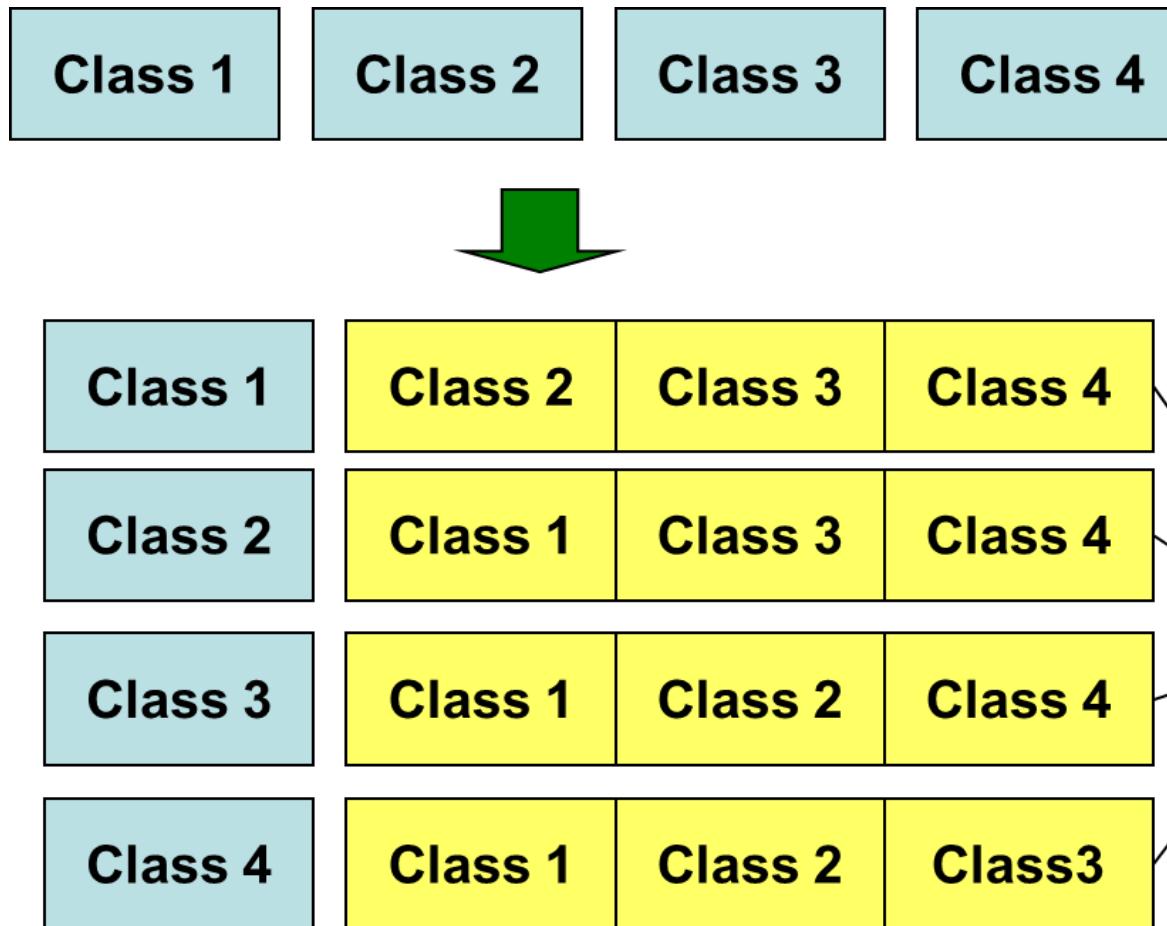
	A1	A2	A3	B1	B2	B3
A1	$K(A1,A1)$	$K(A1,A2)$	$K(A1,A3)$	$K(A1,B1)$	$K(A1,B2)$	$K(A1,B3)$
A2	$K(A2,A1)$	$K(A2,A2)$	$K(A2,A3)$	$K(A2,B1)$	$K(A2,B2)$	$K(A2,B3)$
A3	$K(A3,A1)$	$K(A3,A2)$	$K(A3,A3)$	$K(A3,B1)$	$K(A3,B2)$	$K(A3,B3)$
B1	$K(B1,A1)$	$K(B1,A2)$	$K(B1,A3)$	$K(B1,B1)$	$K(B1,B2)$	$K(B1,B3)$
B2	$K(B2,A1)$	$K(B2,A2)$	$K(B2,A3)$	$K(B2,B1)$	$K(B2,B2)$	$K(B2,B3)$
B3	$K(B3,A1)$	$K(B3,A2)$	$K(B3,A3)$	$K(B3,B1)$	$K(B3,B2)$	$K(B3,B3)$

Pemakaian SVM pada Multiclass Problem

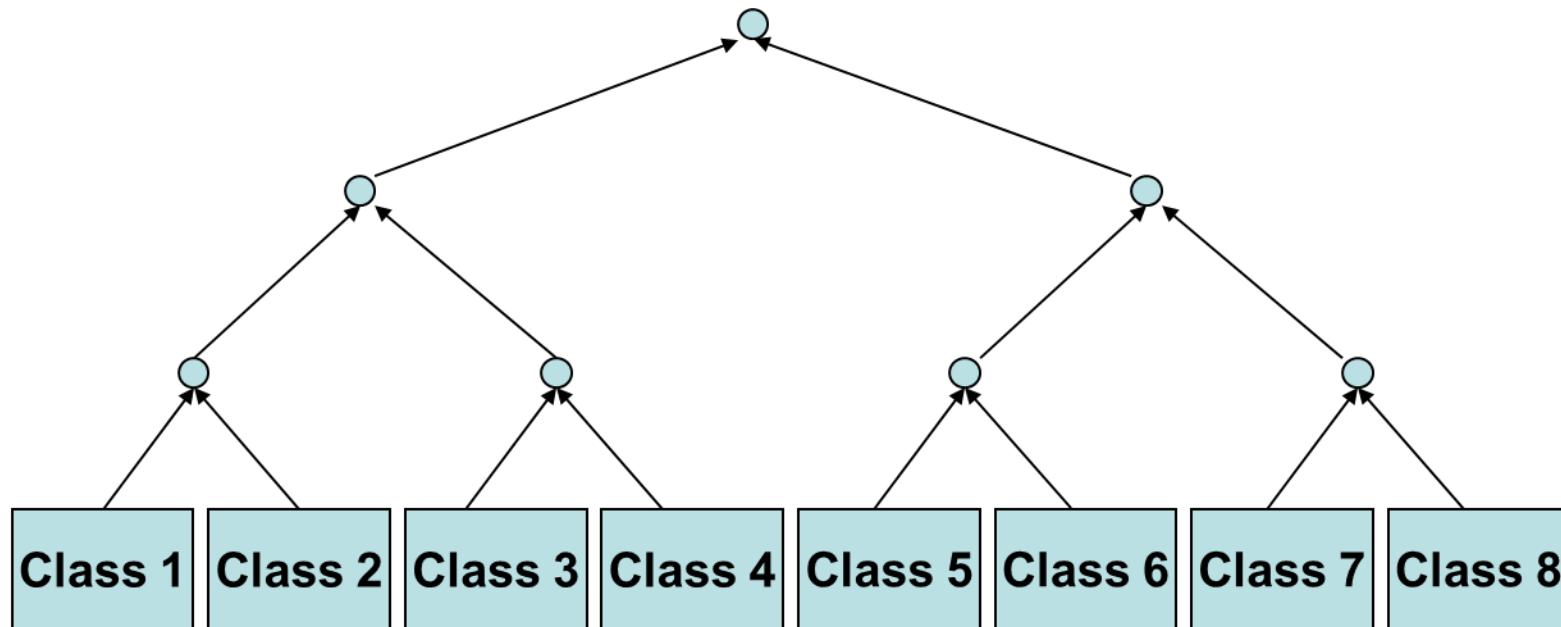
Multiclass Problems

- Pada prinsipnya SVM adalah binary classifier
- Expansion to multiclass classifier:
 1. One vs Others Approach
 2. One vs One : tree structured approach
 1. Bottom-up tree (Pairwise)
 2. Top-down tree (Decision Directed Acyclic Graph)
- Dari sisi training effort : One to Others lebih baik daripada One vs One
- Runtime : keduanya memerlukan evaluasi q SVMs ($q = \text{num. of classes}$)

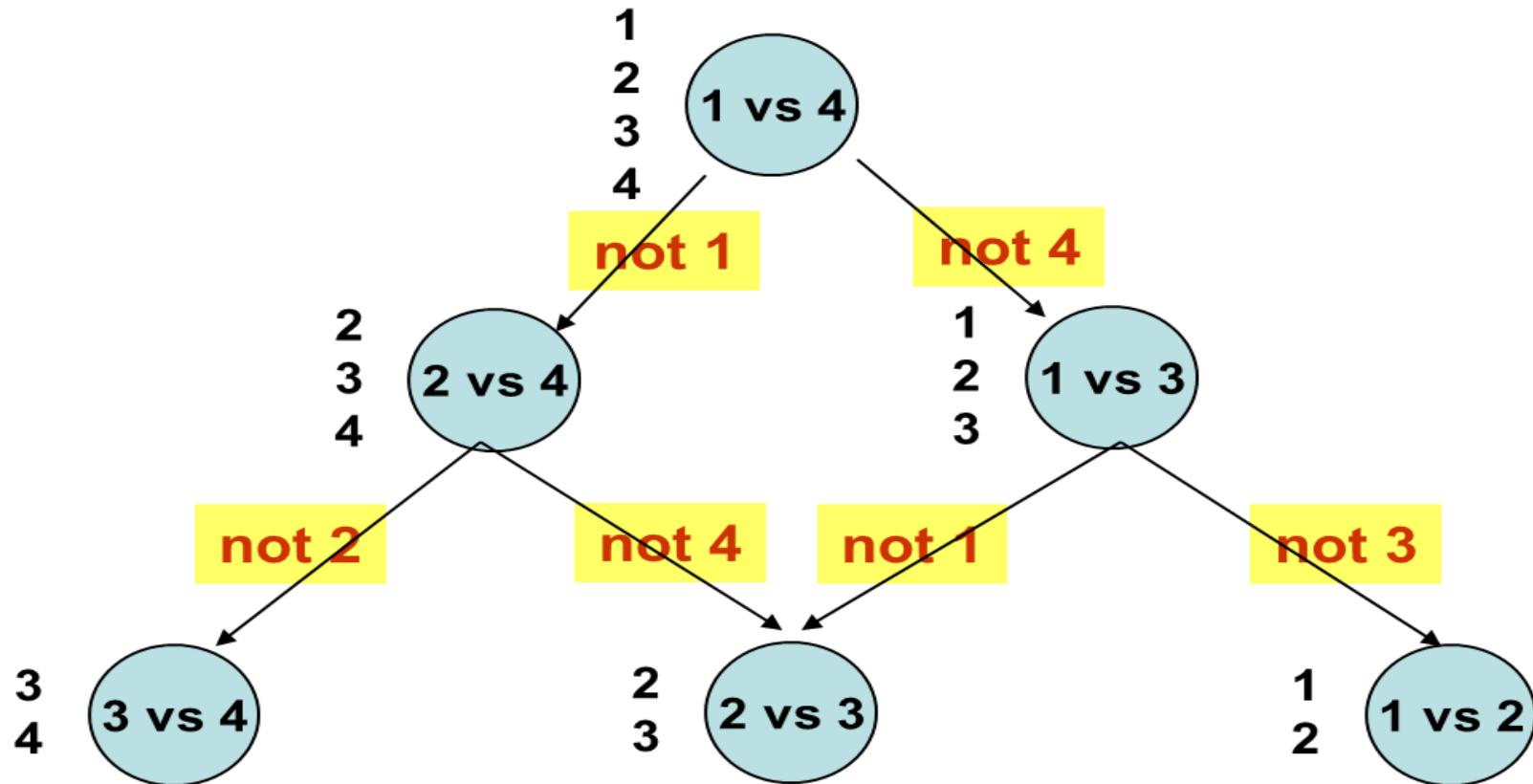
One vs Others



Bottom-Up Tree



Top-Down Tree (DDAG)



Experiment : Digit Recognition

- Num. of class : 10
- Num. of samples
 - Training Set : 100 samples/class
 - Test Set : 100 samples/class
- Num. of attributes (Dimension) : 64
- Feature Extraction : Mesh 8x8
- Database source : SANYO Handwriting Numeral Database (we used only printed-font characters)

Part of patterns in training set

0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

Part of patterns in test set

0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9
0 1 2 3 4 5 6 7 8 9

SVM Experimental Results

- SVM Parameters :
 - $\gamma=0.01$
 - $\lambda:3.0$
 - C:1.0
 - Vijayakumar Algorithm max iteration : 100
 - Gaussian Kernel with $\sigma=0.5$
- Recognition rate :
 - Training Set : 100%
 - Test set : 100%

KARAKTERISTIK SVM

Karakteristik SVM sebagaimana telah dijelaskan sebelumnya, dirangkumkan sebagai berikut:

1. SVM adalah suatu teknik untuk melakukan prediksi, baik dalam kasus klasifikasi maupun regresi. SVM berada dalam satu kelas dengan Artificial Neural Network (ANN) dalam hal fungsi dan kondisi permasalahan yang bisa diselesaikan. Keduanya masuk dalam kelas *supervised learning*.
2. Secara prinsip SVM adalah linear classifier

3. Pattern recognition dilakukan dengan mentransformasikan data pada input space ke ruang yang berdimensi lebih tinggi, dan optimisasi dilakukan pada ruang vector yang baru tersebut. Hal ini membedakan SVM dari solusi pattern recognition pada umumnya, yang melakukan optimisasi parameter pada ruang hasil transformasi yang berdimensi lebih rendah daripada dimensi input space.
4. Menerapkan strategi *Structural Risk Minimization (SRM)*
5. Prinsip kerja SVM pada dasarnya hanya mampu menangani klasifikasi dua class.

KELEBIHAN DAN KEKURANGAN SVM

Kelebihan SVM antara lain sbb:

1. Generalisasi

kemampuan suatu metode (SVM, neural network, dsb.) untuk mengklasifikasikan suatu pattern, yang tidak termasuk data yang dipakai dalam fase pembelajaran metode itu.

Vapnik menjelaskan bahwa generalization error dipengaruhi oleh dua faktor: error terhadap training set, dan error dipengaruhi oleh dimensi VC (Vapnik-Chervokinensis).

Strategi pembelajaran pada neural network dan umumnya metode learning machine difokuskan pada usaha untuk meminimalkan error pada training-set. Strategi ini disebut *Empirical Risk Minimization (ERM)*.

Adapun SVM selain meminimalkan error pada training-set, juga meminimalkan faktor kedua. Strategi ini disebut *Structural Risk Minimization (SRM)*, dan dalam SVM diwujudkan dengan memilih hyperplane dengan margin terbesar. Pendekatan SRM pada SVM memberikan error generalisasi yang lebih kecil daripada yang diperoleh dari strategi ERM pada neural network maupun metode yang lain.

2. Curse of dimensionality

Curse of dimensionality didefinisikan sebagai masalah yang dihadapi suatu metode pattern recognition dalam mengestimasikan parameter (misalnya jumlah hidden neuron pada neural network, stopping criteria dalam proses pembelajaran dsb.) dikarenakan jumlah sampel data yang relatif sedikit dibandingkan dimensional ruang vektor data tersebut.

Semakin tinggi dimensi dari ruang vektor informasi yang diolah, membawa konsekuensi dibutuhkannya jumlah data dalam proses pembelajaran. Tingkat generalisasi yang diperoleh oleh SVM tidak dipengaruhi oleh dimensi dari input vector. SVM merupakan salah satu metode yang tepat dipakai untuk memecahkan masalah berdimensi tinggi, dalam keterbatasan sampel data yang ada.

3. Feasibility

SVM dapat diimplementasikan relatif mudah, karena proses penentuan support vector dapat dirumuskan dalam QP problem. Dengan demikian jika memiliki *library* untuk menyelesaikan QP problem, dengan sendirinya SVM dapat diimplementasikan dengan mudah. Selain itu dapat diselesaikan dengan metode sekuensial sebagaimana penjelasan sebelumnya.

SVM memiliki kelemahan atau keterbatasan, antara lain:

1. Sulit dipakai dalam problem berskala besar. Skala besar dalam hal ini dimaksudkan dengan jumlah sample yang diolah.
2. SVM secara teoritik dikembangkan untuk problem klasifikasi dengan dua class. Dewasa ini SVM telah dimodifikasi agar dapat menyelesaikan masalah dengan class lebih dari dua, antara lain strategi One versus rest dan strategi Tree Structure.

PERTEMUAN 10

Naive Bayes dan K-Nearest-Neighbor

Algoritma pembelajaran mesin dapat dibagi menjadi beberapa kategori. Dari sudut pandang apakah algoritma memiliki parameter yang harus dioptimasi, dapat dibagi menjadi:

1. **Parametrik.** Pada kelompok ini, kita mereduksi permasalahan sebagai optimisasi parameter. Kita mengasumsikan permasalahan dapat dilambangkan oleh fungsi dengan bentuk tertentu (e.g., linear, polinomial, dsb). Contoh kelompok ini adalah model linear.
2. **Non parametrik.** Pada kelompok ini, kita tidak mengasumsikan permasalahan dapat dilambangkan oleh fungsi dengan bentuk tertentu. Contoh kelompok ini adalah Naive Bayes, decision tree (ID3) dan K-Nearest Neighbors.

Dari sudut pandang lainnya, jenis algoritma dapat dibagi menjadi:

1. **Model linear**, contoh regresi linear, regresi logistik, support vector machine.
2. **Model probabilistik**, contoh Naive Bayes, hidden markov model.
3. **Model non-linear**, yaitu (typically) articial neural network.

Naive Bayes

Naive Bayes adalah algoritma supervised learning yang sangat sederhana. Secara formal, persamaan Naive Bayes untuk klasifikasi diberikan pada persamaan:

$$\text{likelihood}(c_i) = P(c_i) \prod_{f=1}^F P(t_f | c_i)$$

Dimana: c_i adalah suatu nilai kelas, C adalah kelas (himpunan), t adalah fitur (satu fitur, bukan feature vector) dan F adalah banyaknya fitur.

Kita memprediksi kelas berdasarkan probabilitas kemunculan nilai fitur pada kelas tersebut. Pertama, kita hitung likelihood suatu feature vector diklasifikasikan ke kelas tertentu berdasarkan bagaimana probabilitas korespondensi fitur-fiturnya terhadap kelas tersebut.

Kemudian, kita normalisasi likelihood semua kelas untuk mendapatkan probabilitas class-assignment (softmax).

Akhirnya, kita pilih kelas dengan probabilitas tertinggi

$$\text{likelihood}(c_i) = P(c_i) \prod_{f=1}^F P(t_f | c_i)$$

$$P_{\text{assignment}}(c_i) = \frac{\text{likelihood}(c_i)}{\sum_{c_j \in C} \text{likelihood}(c_j)}$$

$$\hat{c}_i = \arg \max_{c_i \in C} P_{\text{assignment}}(c_i)$$

Agar mendapatkan gambaran praktis, mari bangun model Naive Bayes untuk Tabel 4.1. Tabel ini disebut sebagai dataset, yaitu memuat entry data (tiap baris disebut sebagai instans/instance). Kita anggap Tabel 4.1 sebagai training data. Untuk menghitung probabilitas, pertama-tama hitung terlebih dahulu frekuensi nilai atribut seperti pada Tabel 4.2, setelah itu bangun model probabilitasnya seperti pada Tabel 4.3.

id	outlook	temperature	humidity	windy	play (class)
1	sunny	hot	high	false	no
2	sunny	hot	high	true	no
3	overcast	hot	high	false	yes
4	rainy	mild	high	false	yes
5	rainy	cool	normal	false	yes
6	rainy	cool	normal	true	no
7	overcast	cool	normal	true	yes
8	sunny	mild	high	false	no
9	sunny	cool	normal	false	yes
10	rainy	mild	normal	false	yes
11	sunny	mild	normal	true	yes
12	overcast	mild	high	true	yes
13	overcast	hot	normal	false	yes
14	rainy	mild	high	true	no

Tabel 4.1. Contoh dataset *play tennis* (UCI machine learning repository)

	outlook		temperature		humidity		windy	play (class)		
	yes	no	yes	no	yes	no	yes	no	yes	no
sunny	2	3	hot	2	3	high	3	4	false	6
overcast	4	0	mild	4	2	normal	6	1	true	3
rainy	3	2	cool	3	1					3

Tabel 4.2. Frekuensi setiap nilai atribut

	outlook		temperature		humidity		windy	play (class)		
	yes	no	yes	no	yes	no	yes	no	yes	no
sunny	2/9	3/5	hot	2/9	3/5	high	3/9	4/5	false	6/9
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9
rainy	3/9	2/5	cool	3/9	1/5					3/5

Tabel 4.3. Probabilitas setiap nilai atribut

Untuk menguji kebenaran model yang telah dibangun, kita menggunakan testing data, diberikan pada Tabel 4.4. testing data berisi unseen example yaitu contoh yang tidak ada pada training data.

id	outlook	temperature	humidity	windy	play (class)
1	sunny	cool	high	true	no

Tabel 4.4. Contoh testing data *play tennis* [7]

$$\begin{aligned} \text{likelihood}(play = yes) &= P(yes)P(sunny|yes)P(cool|yes)P(high|yes)P(true|yes) \\ &= \frac{9}{14} * \frac{2}{9} * \frac{3}{9} * \frac{3}{9} * \frac{3}{9} \\ &= 0.0053 \end{aligned}$$

$$\begin{aligned} \text{likelihood}(play = no) &= P(no)P(sunny|no)P(cool|no)P(high|no)P(true|no) \\ &= \frac{5}{14} * \frac{3}{5} * \frac{1}{5} * \frac{4}{5} * \frac{3}{5} \\ &= 0.0206 \end{aligned}$$

$$\begin{aligned} P_{\text{assignment}}(play = yes) &= \frac{\text{likelihood}(play = yes)}{\text{likelihood}(play = yes) + \text{likelihood}(play = no)} \\ &= \frac{0.0053}{0.0053 + 0.0206} \\ &= 0.205 \end{aligned}$$

$$\begin{aligned} P_{\text{assignment}}(play = no) &= \frac{\text{likelihood}(play = no)}{\text{likelihood}(play = yes) + \text{likelihood}(play = no)} \\ &= \frac{0.0206}{0.0053 + 0.0206} \\ &= 0.795 \end{aligned}$$

Karena $P_{\text{assignment}}(\text{play} = \text{no}) > P_{\text{assignment}}(\text{play} = \text{yes})$
maka diputuskan bahwa kelas untuk unseen example adalah
 $\text{play} = \text{no}$.

Proses klasifikasi untuk data baru sama seperti proses klasifikasi untuk testing data, yaitu kita ingin menebak kelas data. Karena model berhasil menebak kelas pada training data dengan tepat, akurasi model adalah 100% (kebetulan contohnya hanya ada satu).

K-means

Pada *supervised learning* diketahui kelas data untuk setiap *feature vector*, sedangkan untuk *unsupervised learning* tidak diketahui.

Tujuan *unsupervised learning* salah satunya adalah melakukan ***clustering***. Yaitu mengelompokkan data-data dengan karakter mirip.

Untuk melakukan pembelajaran menggunakan **K-means** harus mengikuti langkah-langkah sebagai berikut:

1. Tentukan jumlah kelompok yang diinginkan.
2. Inisiasi **centroid** untuk setiap kelompok (secara acak).
Centroid adalah data yang merepresentasikan suatu kelompok (ibaratnya ketua kelompok)
3. Hitung kedekatan suatu data terhadap *centroid*, kemudian masukkan data tersebut ke kelompok yang **centroid**-nya memiliki sifat terdekat dengan dirinya.
4. Pilih kembali **centroid** untuk masing-masing kelompok, yaitu dari anggota kelompok tersebut (semacam memilih ketua yang baru).
5. Ulangi langkah-langkah sebelumnya sampai tidak ada perubahan anggota untuk semua kelompok.

id	rich	intelligent	good looking
1	yes	yes	yes
2	yes	no	no
3	yes	yes	no
4	no	no	no
5	no	yes	no
6	no	no	yes

Tabel 4.5. Contoh dataset orang kaya

Perhatikan Tabel 4.5, Kelompokkan data pada tabel tersebut menjadi dua clusters (dua kelompok) yaitu k_1 , k_2 menggunakan algoritma K-means. Pertama-tama inisiasi centroid secara acak, id_1 untuk k_1 dan id_6 untuk k_2 .

Hitung kedekatan data lainnya terhadap centroid. Untuk mempermudah contoh, hitung perbedaan data satu dan lainnya dengan menghitung perbedaan nilai atribut (nilai atributnya sama atau tidak).

Apabila perbedaan suatu data terhadap kedua centroid bernilai sama, masukkan ke kelas dengan nomor urut lebih kecil.

Setelah langkah ini, kelompok satu beranggotakan $\{id_1, id_2, id_3, id_5\}$ sementara kelompok dua beranggotakan $\{id_4, id_6\}$

Pilih kembali centroid untuk masing-masing kelompok yang mana berasal dari anggota kelompok itu sendiri. Misal pilih secara acak, centroid untuk kelompok pertama adalah id_2 sementara untuk kelompok kedua adalah id_4 . Hitung kembali assignment anggota kelompok yang ilustrasinya dapat dilihat pada Tabel 4.7.

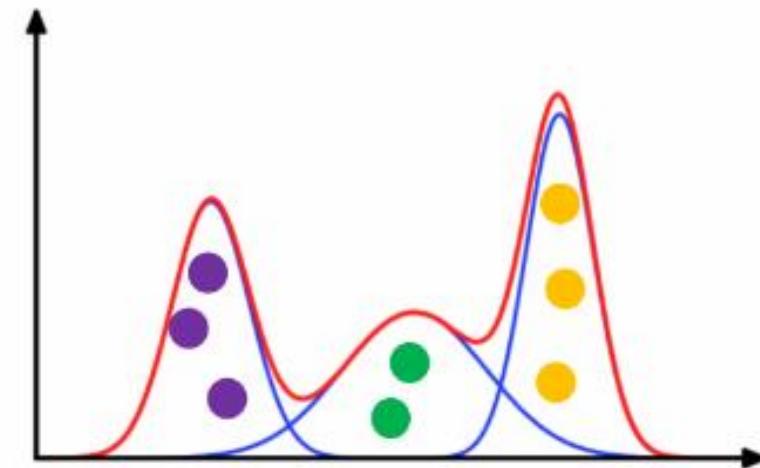
<i>id</i>	perbedaan dengan <i>centroid₁</i>	perbedaan dengan <i>centroid₂</i>	assignment
1	2	3	k_1
3	1	3	k_1
5	3	1	k_2
6	2	2	k_1

Tabel 4.7. Assignment K-Means langkah 2

Hasil langkah ke-2 adalah perubahan anggota kelompok, $k_1 = \{id_1, id_2, id_3, id_5\}$ dan $k_2 = \{id_4, id_6\}$ Anggap pada langkah ke-3 memilih kembali id_2 dan id_4 sebagai centroid masing-masing kelompok sehingga tidak ada perubahan keanggotaan.

Clustering itu memiliki hubungan erat dengan Gaussian Mixture Model. Secara sederhana, satu cluster (atau satu kelas) sebenarnya seolah-olah dapat dipisahkan dengan kelas lainnya oleh distribusi gaussian.

Perhatikan Gambar 4.1! Suatu cluster atau kelas, seolah-olah “dibungkus” oleh suatu distribusi gaussian. Distribusi seluruh dataset dapat diaproksimasi dengan Gaussian Mixture Model (GMM).



Gambar 4.1. Ilustrasi hubungan Clustering, kelas, dan Gaussian

Data memiliki suatu pola (dalam statistik disebut distribusi), GMM dipercaya dapat mengaproksimasi fungsi apapun.

Dengan demikian, machine learning yang mempunyai salah satu tujuan untuk menemukan pola dataset, memiliki hubungan yang sangat erat dengan distribusi gaussian karena pola tersebut dapat diaproksimasi dengan distribusi gaussian.

K-Nearest-Neighbor

- Ide **K-Nearest-Neighbor** (KNN) adalah mengelompokkan data ke kelompok yang memiliki sifat termirip dengannya.
- Hal ini sangat mirip dengan **K-means**. Bila K-means digunakan untuk *clustering*, KNN digunakan untuk klasifikasi.

Algoritma klasifikasi ini disebut juga algoritma malas karena tidak mempelajari cara mengkategorikan data, melainkan hanya mengingat data yang sudah ada yang kita telah mengelompokkan data orang kaya menjadi dua kelompok.

id	rich	intelligent	good looking
1	yes	yes	yes
2	yes	no	no
3	yes	yes	no
4	no	no	no
5	no	yes	no
6	no	no	yes

Tabel 4.5. Contoh dataset orang kaya

KNN mencari K *feature vector* dengan sifat termirip, kemudian mengelompokkan data baru ke kelompok *feature vector* tersebut. Sebagai ilustrasi mudah, lakukan klasifikasi algoritma KNN dengan K = 3 untuk data baru {*rich* = no; *intelligent* = yes; *good looking* = yes}.

Pada Tabel 4.8. *feature vector* termirip dimiliki oleh data dengan id₁, id₅, id₆.

id	perbedaan
1	1
2	3
3	3
4	2
5	1
6	1

Tabel 4.8. Perbedaan data baru vs data orang kaya

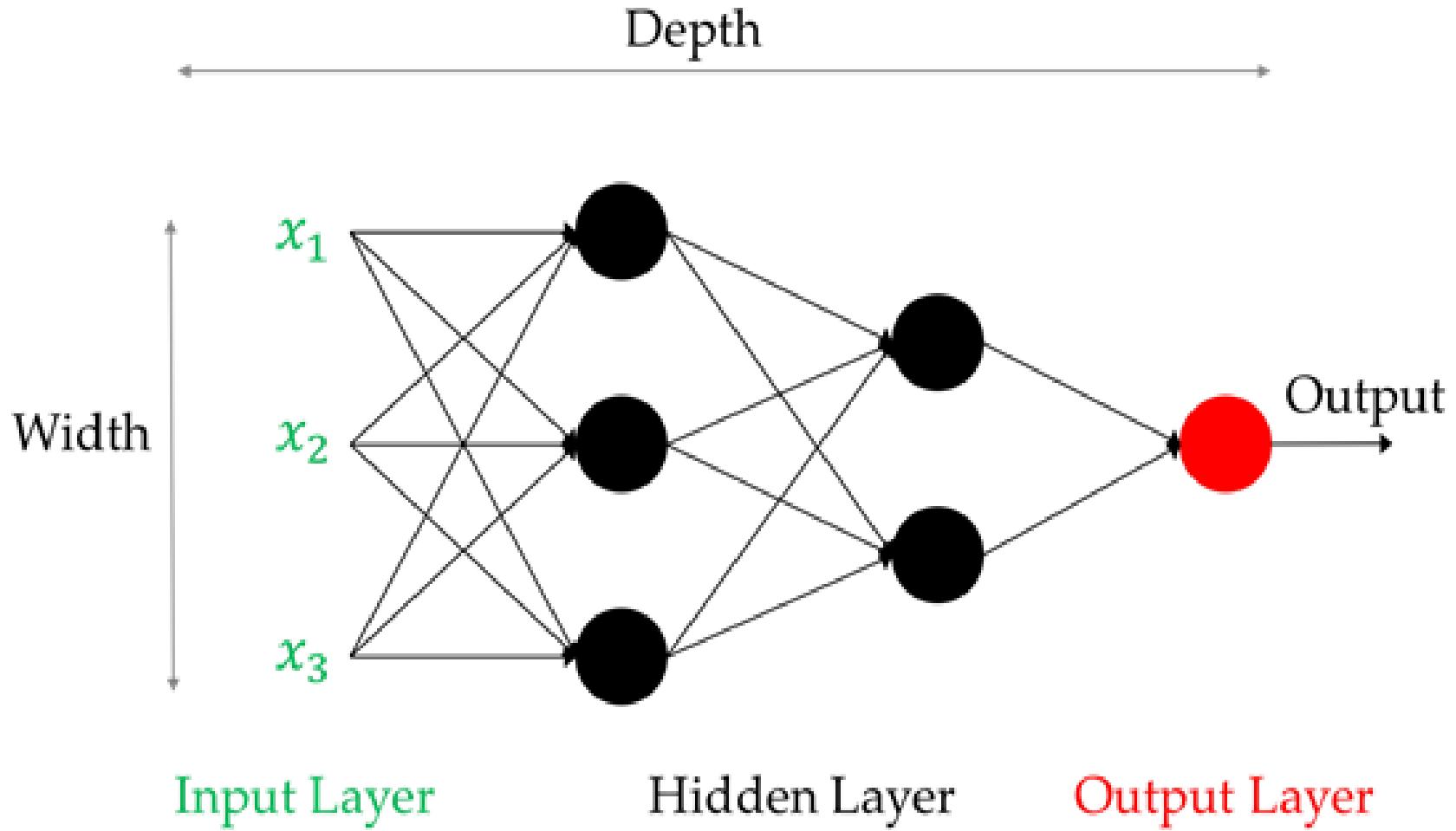
PERTEMUAN 11

Neural Network
&
Self Organizing Map (SOM)

Neural Network adalah salah satu algoritma *supervised learning* yang populer dan bisa juga digunakan untuk *semi-supervised* atau *unsupervised learning*

Artificial Neural Network (ANN), menghasilkan model yang sulit dibaca dan dimengerti oleh manusia karena memiliki banyak layer (kecuali *single perceptron*) dan sifat **non-linear** (merujuk pada fungsi aktivasi).

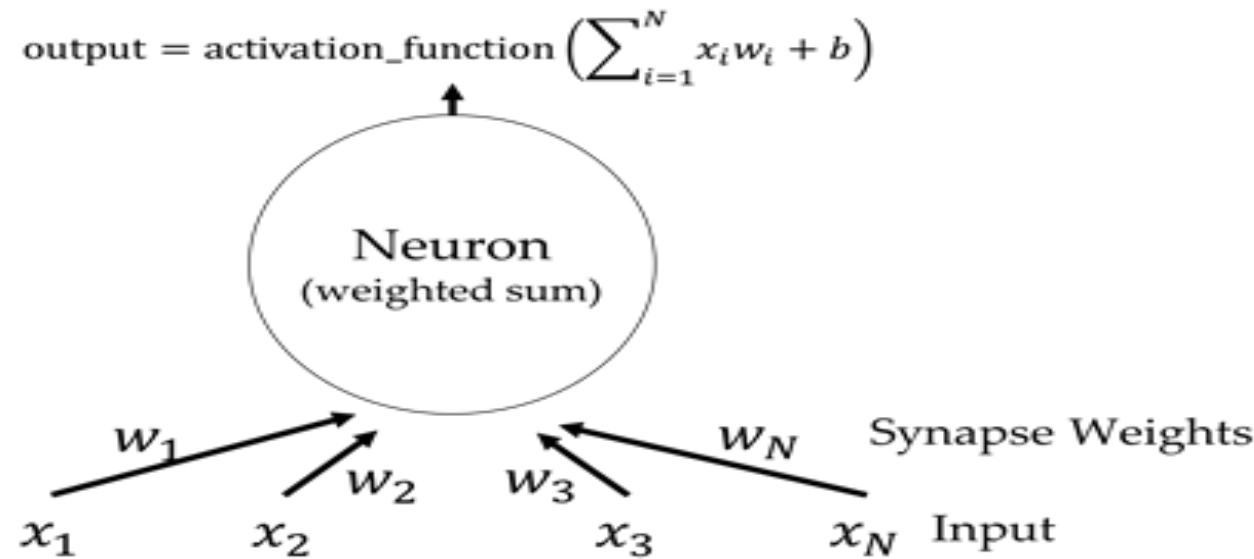
Secara matematis, ANN ibarat sebuah graf, ANN memiliki neuron/*node* (*vertex*), dan sinapsis (*edge*). Topologi ANN akan dibahas lebih detil subbab berikutnya. Karena memiliki struktur seperti graf operasi pada ANN mudah dijelaskan dalam notasi aljabar linear



Single Perceptron

Bentuk terkecil (minimal) sebuah ANN adalah *single perceptron* yang hanya terdiri dari sebuah neuron

Sebuah neuron diilustrasikan pada gambar dibawah ini :



Secara matematis, terdapat *feature vector* x yang menjadi *input* bagi neuron tersebut. Ingat kembali, *feature vector* merepresentasikan suatu *data point*, *event* atau instans. Neuron akan memproses *input* x melalui perhitungan

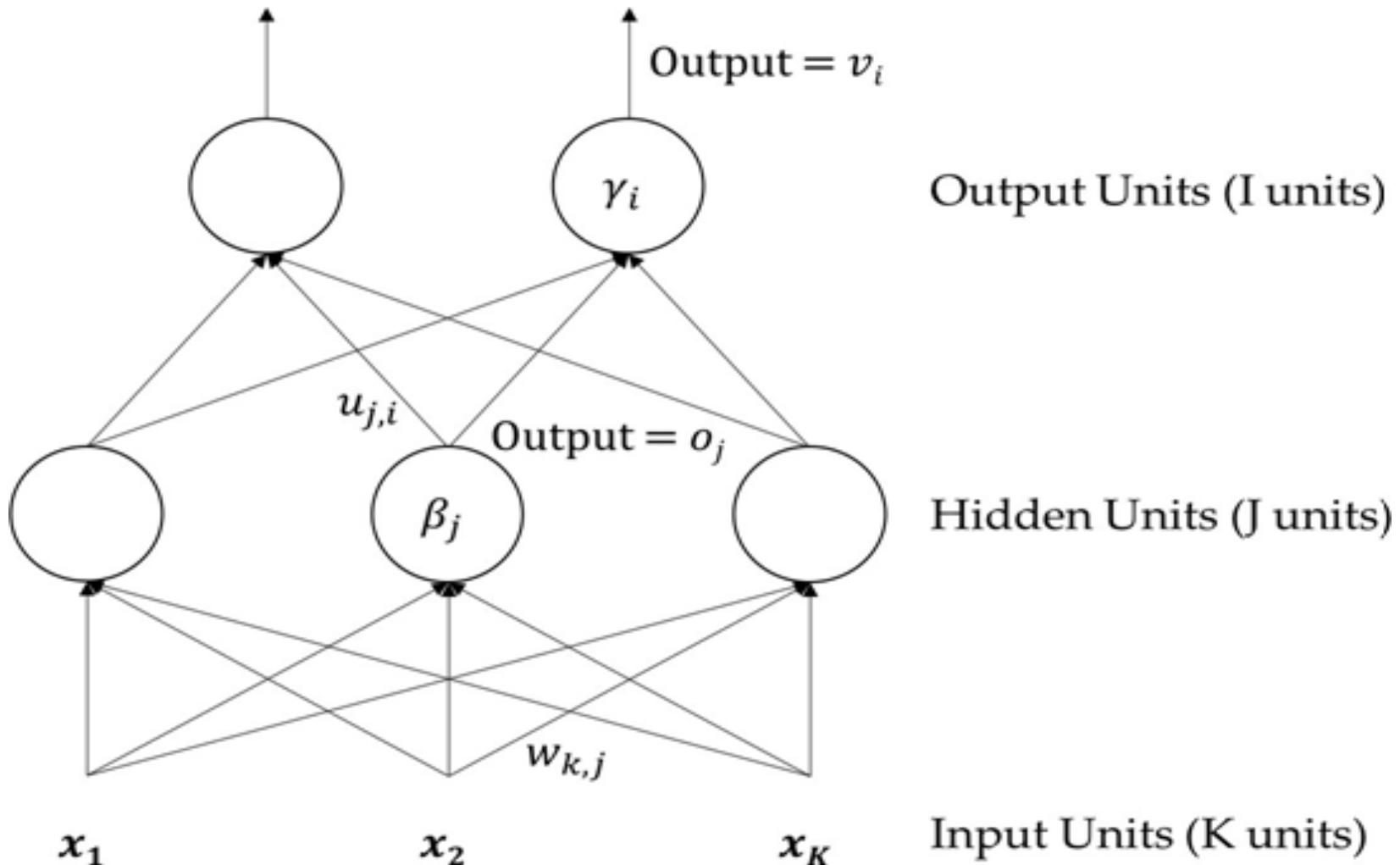
Untuk melakukan pembelajaran *single perceptron*, *training* dilakukan menggunakan ***perceptron training rule***. Prosesnya sebagai berikut

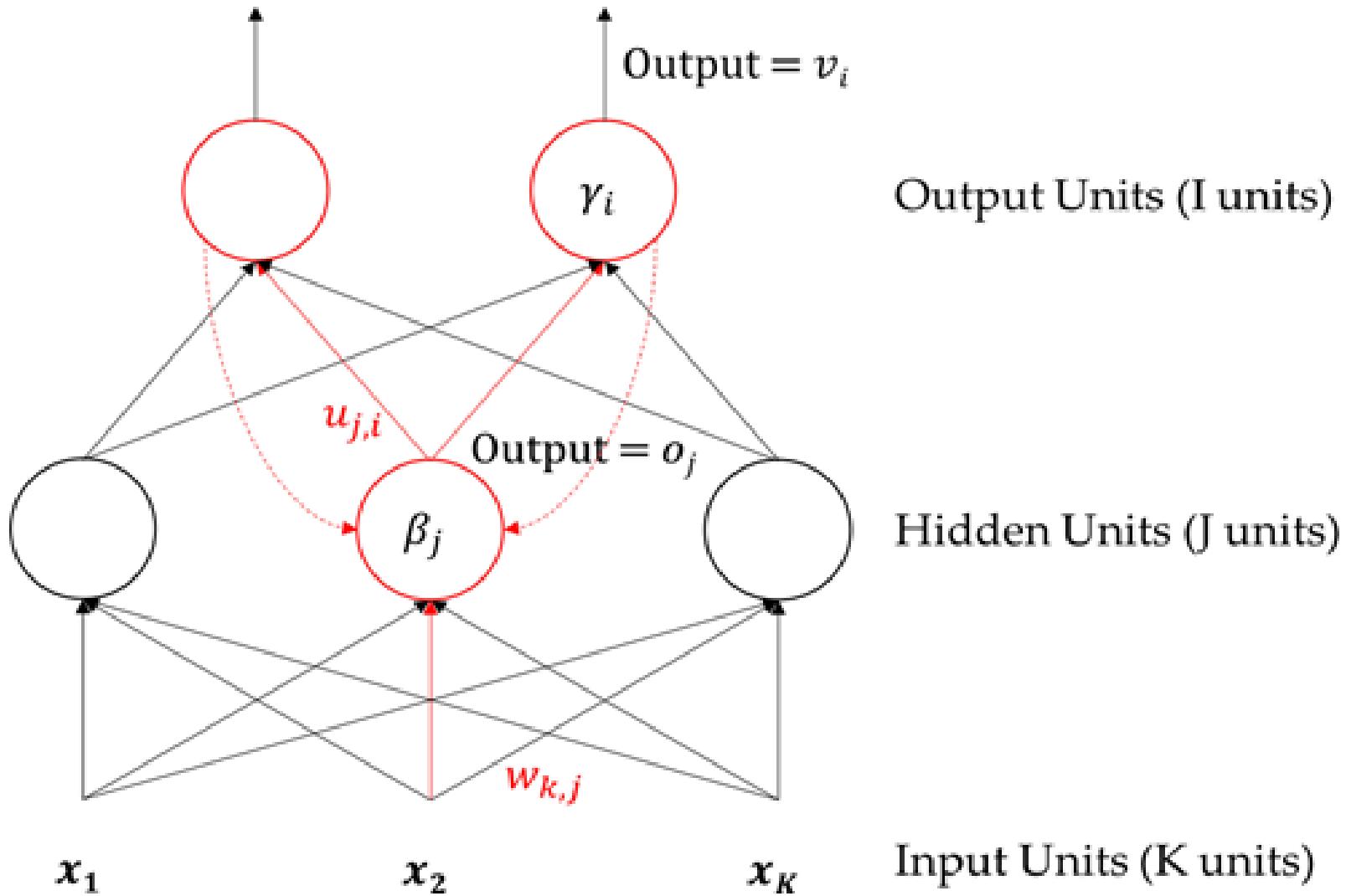
1. Inisiasi nilai *synapse weights*, bisa *random* ataupun dengan aturan ter- tentu. Untuk heuristik aturan inisiasi, ada baiknya membaca buku referensi [.](#)
2. Lewatkan input pada neuron, kemudian kita akan mendapatkan nilai *out- put*. Kegiatan ini disebut ***feedforward***

3. Nilai output (actual output) tersebut dibandingkan dengan desired output.
4. Apabila nilai output sesuai dengan desired output, tidak perlu mengubah apa-apa.
5. Apabila nilai output tidak sesuai dengan desired output, hitung nilai error (loss) kemudian lakukan perubahan terhadap learning parameter (synapse weight).
6. Ulangi langkah-langkah ini sampai tidak ada perubahan nilai error, nilai error kurang dari sama dengan suatu threshold (biasanya mendekati 0), atau sudah mengulangi proses latihan sebanyak T kali (threshold).

Multilayer Perceptron

multilayer perceptron secara literal memiliki beberapa *layers*. Pada *lecture note* ini, secara umum ada tiga *layers*: *input*, *hidden*, dan *output layer*. *Input layer* menerima *input* (tanpa melakukan operasi apapun), kemudian nilai *input* (tanpa dilewatkan ke fungsi aktivasi) diberikan ke *hidden units*. Pada *hidden units*, *input* diproses dan dilakukan perhitungan hasil fungsi aktivasi untuk tiap-tiap neuron, lalu hasilnya diberikan ke *layer* berikutnya. *Output* dari *input layer* akan diterima sebagai input bagi *hidden layer*. Begitupula seterusnya *hidden layer* akan mengirimkan hasilnya untuk *output layer*. Kegiatan ini dinamakan ***feed forward***.





Self Organizing Map (SOM)

Pengertian

SOM merupakan salah satu teknik dalam Neural Network yang bertujuan untuk melakukan visualisasi data dengan cara mengurangi dimensi data melalui penggunaan self-organizing neural networks sehingga manusia dapat mengerti high-dimensional data yang dipetakan dalam bentuk low-dimensional data

Self-Organizing Map (SOM) atau sering disebut topology-preserving map pertama kali diperkenalkan oleh Teuvo Kohonen pada tahun 1996

Metode pembelajaran yang digunakan SOM adalah tanpa bimbingan dari suatu data input-target atau unsupervised learning yang mengasumsikan sebuah topologi yang terstruktur menjadi unit-unit kelas/cluster (Kohonen, 1989 dan Fausett, 1993).

Pada algoritma SOM, vektor bobot untuk setiap unit cluster berfungsi sebagai contoh dari input pola yang terkait dengan cluster itu. Selama proses self-organizing, cluster satuan yang bobotnya sesuai dengan pola vektor input yang paling dekat (biasanya, kuadrat dari jarak Euclidean minimum) dipilih sebagai pemenang

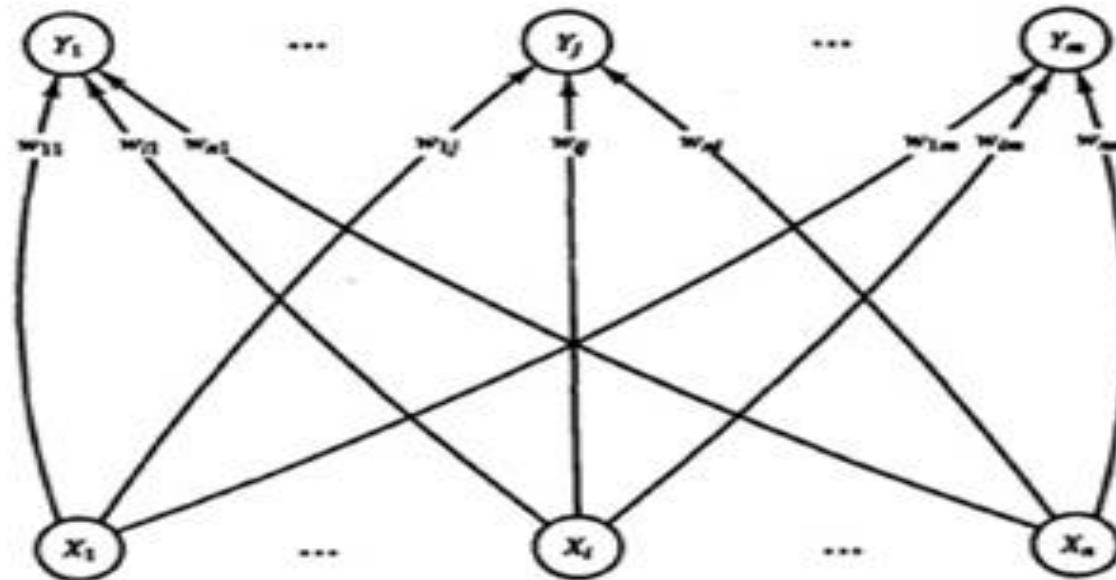
Unit pemenang dan unit tetangganya (dalam pengertian topologi dari unit cluster) terus memperbarui bobot mereka (Fausett, 1993). Setiap output akan bereaksi terhadap pola input tertentu sehingga hasil Kohonen SOM akan menunjukkan adanya kesamaan ciri antar anggota dalam cluster yang sama.

Dalam jaringan SOM, neuron target tidak diletakkan dalam sebuah baris seperti layaknya model JST yang lain. Neuron target diletakkan dalam dua dimensi yang bentuk/topologinya dapat diatur. Topologi yang berbeda akan menghasilkan neuron sekitar neuron pemenang yang berbeda sehingga bobot yang dihasilkan juga akan berbeda

Pada SOM, perubahan bobot tidak hanya dilakukan pada bobot garis yang terhubung ke neuron pemenang saja, tetapi juga pada bobot garis ke neuron-neuron di sekitarnya. neuron di sekitar neuron pemenang ditentukan berdasarkan jaraknya dari neuron pemenang.

Arsitektur Topologi SOM

Arsitektur SOM merupakan jaringan yang terdiri dari dua lapisan (layer), yaitu lapisan input dan lapisan output. Setiap neuron dalam lapisan input terhubung dengan setiap neuron pada lapisan output. Setiap neuron dalam lapisan output merepresentasikan kelas (cluster) dari input yang diberikan.



Topologi SOM

topologi, SOM memiliki 3 jenis topologi hubungan ketetanggaan (neighborhood) yaitu linear array, rectangular dan heksagonal grid

Topologi Linier

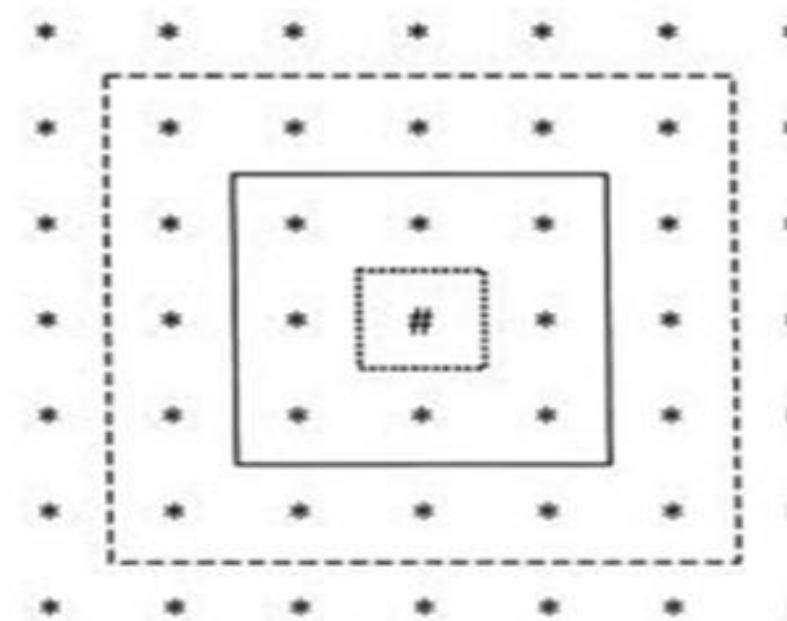
Topologi linear aray menunjukkan cluster unit yang tersusun secara linear. Cluster unit yang menjadi pemenang [#] memiliki dua unit tetangga (neighbour) yang berjarak 1 ($R = 1$), dan mempunyai dua unit tetangga yang berjarak 2 ($R = 2$).

* * * { * (* [#] *) * } *

Keterangan: []: $R=0$; () : $R=1$; {} : $R=2$

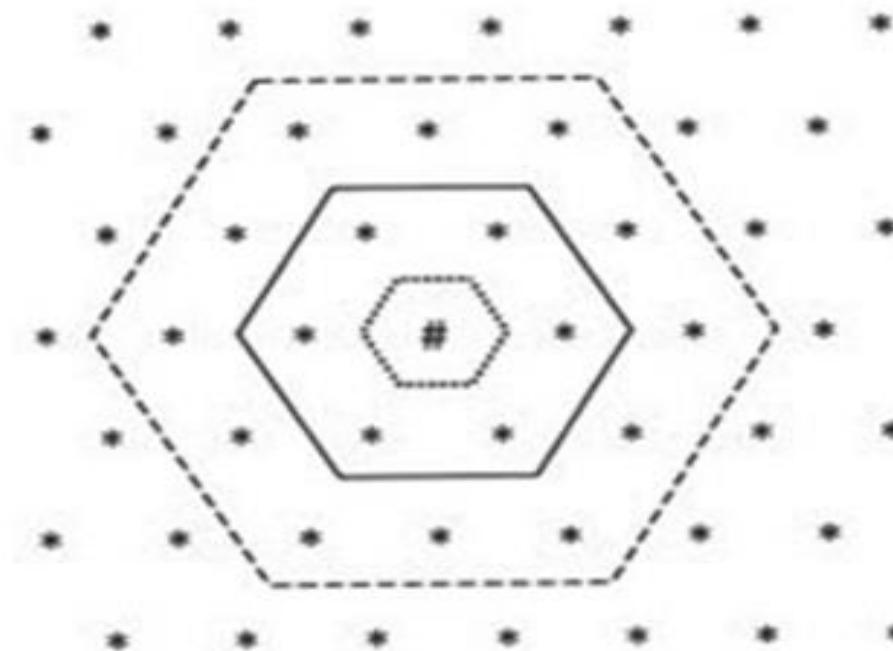
Rectangular grid

Rectangular grid adalah topologi dari cluster unit dua dimensi. Unit tetangga (neighbour) dari unit pemenang membentuk bujur sangkar. Unit pemenang [#] memiliki 8 neighbour berjarak 1 ($R=1$) dan 16 neighbour berjarak 2 ($R=2$).



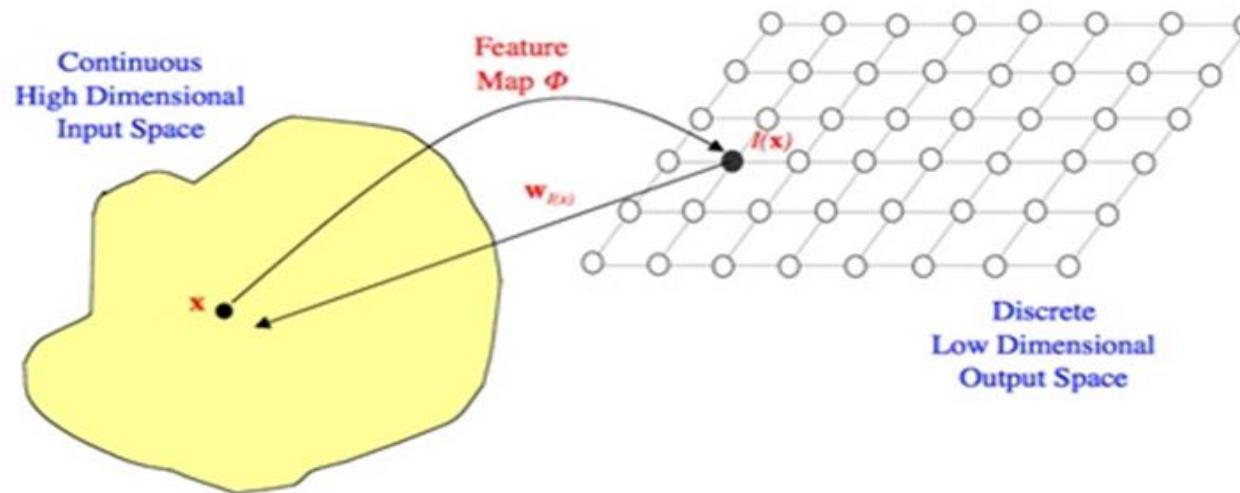
topologi heksagonal

Dalam topologi heksagonal grid, unit tetangga (neighbour) yang berjarak 1 ($R=1$) dari unit pemenang adalah 6 dan yang berjarak 2 ($R=2$) adalah 12.



cara kerja SOM

Secara umum, cara kerja SOM ditunjukkan oleh Gambar 5 dibawah ini:



Terdapat titik (x) pada ruang input untuk dipetakan ke titik $l(x)$ pada ruang output. Setiap titik (l) dalam ruang output akan memetakan ke titik yang sesuai dalam ruang input melalui bobot $w_l(x)$.

Menurut Haykin (1999) terdapat tiga komponen penting dalam SOM yaitu:

1. Competition: Untuk setiap pola input, neuron menghitung nilai masing-masing fungsi diskriminan yang memberi dasar untuk kompetisi. Neuron tertentu dengan nilai terkecil dari fungsi diskriminan dinyatakan sebagai pemenang.
2. Cooperation: Neuron pemenang menentukan lokasi spasial dari lingkungan topologi excited neuron untuk memberi dasar kerjasama dalam suatu lingkungan neuron.
3. Synaptic Adaption: Excited neuron menurunkan nilai fungsi diskriminan yang berkaitan dengan pola input melalui penyesuaian bobot terkait sehingga respon dari neuron pemenang keaplikasi berikutnya dengan pola input yang sama akan meningkat.

Algoritma SOM

Pengelompokan data menggunakan algoritma SOM yang terdiri dari 4 tahapan yaitu:

1. Kompetisi: Setiap simpul output j , dihitung nilai $D(x, w_i)$ yang merupakan fungsi jarak Euclidian antara x dan w_i . Fungsi ini didefinisikan sebagai berikut:

$$D(x, w_m) = \sqrt{\sum_{i=1}^n (x_i - w_{mi})^2} \quad (1)$$

dimana x adalah vektor dari *node input*

sedangkan w_m adalah vektor bobot dari node neuron ke- m .

2. Update Bobot: setelah mendapat nilai jarak dari tiap-tiap vektor *input* kevektor bobot, pilih nilai jarak yang minimum sebagai neuron pemenang. Setiap neuron pemenang beserta tetangganya dilakukan proses adaptasi yaitu memperbaharui nilai bobot dimana $h(t)$ adalah fungsi *node tetangga* (neighborhood function) dan t adalah banyaknya iterasi. Fungsi *node tetangga* yang digunakan adalah fungsi Gauss (Kohonen et al, 2001) dengan formula:

$$h(t) = \alpha(t) \times e^{(-\frac{\|r_i - r_c\|^2}{2\delta^2(t)})} \quad (3)$$

Dimana $\alpha(t)$ adalah nilai laju pembelajaran atau biasa disebut nilai alpha. Laju pembelajaran adalah fungsi penurunan tingkat pembelajaran seiring perubahan waktu (Fausett 1993).

$\|r_i - r_c\|^2$ adalah jarak kuadrat antara *neuron* ke- i dengan *neuron* pemenang dalam grid dan $\delta(t)$ adalah lebar tetangga. Nilai laju pembelajaran diperoleh dari:



$$\alpha(t) = \alpha_i \left(1 - \frac{t}{t_{max}}\right) \quad (4)$$

Dimana α_i adalah nilai awal laju pembelajaran dan t_{max} adalah iterasi maksimum. Perubahan lebar tetangga didapat dari perhitungan berikut ini:

$$\delta(t) = \delta_i \left(\frac{\delta_f}{\delta_i}\right)^{\frac{t}{t_{max}}} \quad (7)$$

PERTEMUAN 12

Konsep Deep Learning

DL memberikan hasil terobosan dalam pengenalan suara dan klasifikasi gambar

From this Hinton et al 2012 paper:

<http://static.googleusercontent.com/media/research.google.com/en//pubs/archive/38131.pdf>

modeling technique	#params [10 ⁶]	WER	
		Hub5'00-SWE	RT03S-FSH
GMM, 40 mix DT 309h SI	29.4	23.6	27.4
NN 1 hidden-layer × 4634 units	43.6	26.0	29.4
+ 2 × 5 neighboring frames	45.1	22.4	25.7
DBN-DNN 7 hidden layers × 2048 units	45.1	17.1	19.6
+ updated state alignment	45.1	16.4	18.6
+ sparsification	15.2 nz	16.1	18.5
GMM 72 mix DT 2000h SA	102.4	17.1	18.6

task	hours of training data	DNN-HMM	GMM-HMM with same data	GMM-HMM with more data
Switchboard (test set 1)	309	18.5	27.4	18.6 (2000 hrs)
Switchboard (test set 2)	309	16.1	23.6	17.1 (2000 hrs)
English Broadcast News	50	17.5	18.8	
Bing Voice Search (Sentence error rates)	24	30.4	36.2	
Google Voice Input	5,870	12.3		16.0 (>>5,870hrs)
Youtube	1,400	47.6	52.3	

go here: <http://yann.lecun.com/exdb/mnist/>

From here:

<http://people.idsia.ch/~juergen/cvpr2012.pdf>

Dataset	Best result of others [%]	MCDNN [%]	Relative improv. [%]
MNIST	0.39	0.23	41
NIST SD 19	see Table 4	see Table 4	30-80
HWDB1.0 on.	7.61	5.61	26
HWDB1.0 off.	10.01	6.5	35
CIFAR10	18.50	11.21	39
traffic signs	1.69	0.54	72
NORB	5.00	2.70	46

Apa sebenarnya ***deep Learning?***

mengapa umumnya lebih baik daripada metode lain pada gambar (*IMAGE*), ucapan (*Speech*) dan jenis data tertentu lainnya?

The short answers

1. ‘Deep Learning’ berarti menggunakan jaringan syaraf (neural network) dengan beberapa lapisan node antara input dan output

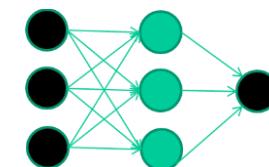
2. Serangkaian lapisan antara input & output lakukan identifikasi dan pemrosesan fitur dalam serangkaian tahapan, seperti otak kita tampaknya.

jaringan saraf multilayer telah ada selama 25 tahun.

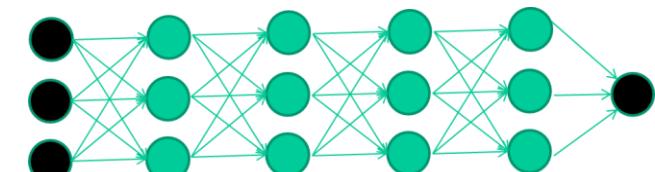
Apa yang sebenarnya baru?

Terdapat selalu memiliki algoritma yang baik untuk mempelajari bobot dalam jaringan dengan 1 lapisan tersembunyi

tetapi algoritma ini tidak pandai mempelajari bobot untuk jaringan dengan lapisan yang lebih tersembunyi

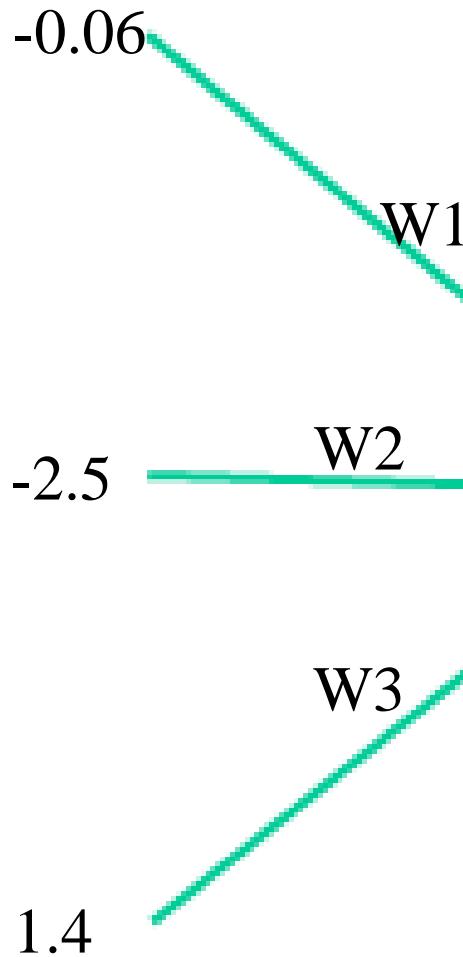


Apa yang terbaru: algorithms for training many-layer networks

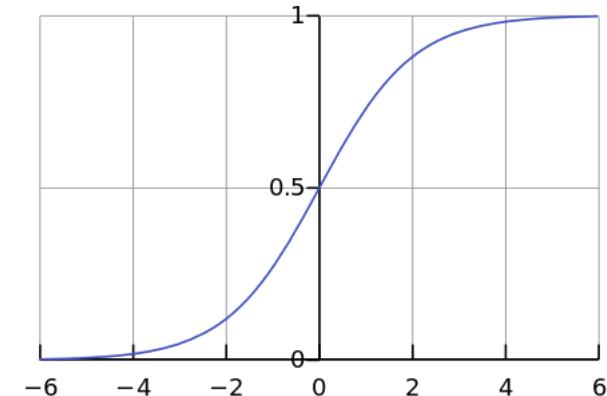


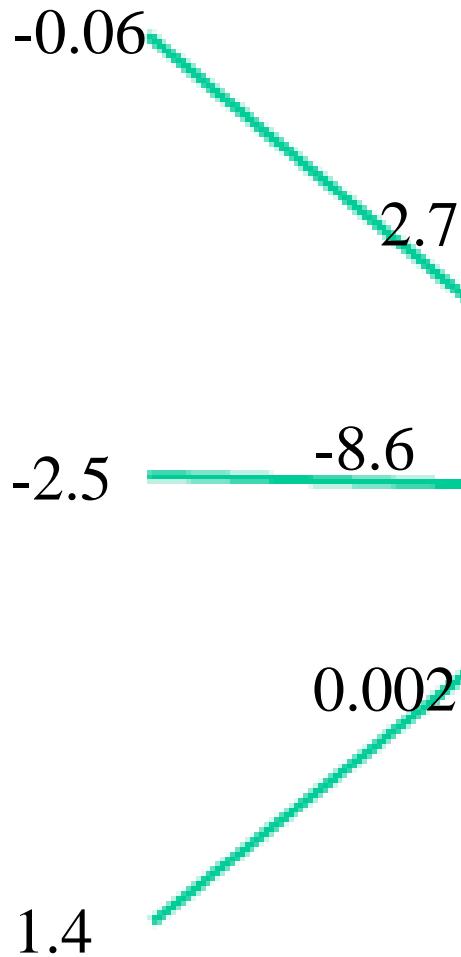
Jawaban yang lebih panjang

1. pengingat / penjelasan singkat tentang bagaimana bobot jaringan saraf dipelajari;
2. gagasan pembelajaran fitur tanpa pengawasan (*unsupervised feature learning*) (mengapa 'fitur menengah' penting untuk tugas klasifikasi yang sulit, dan bagaimana NN tampaknya mempelajarinya secara alami)
3. 'Terobosan' - trik sederhana untuk melatih jaringan saraf yang dalam

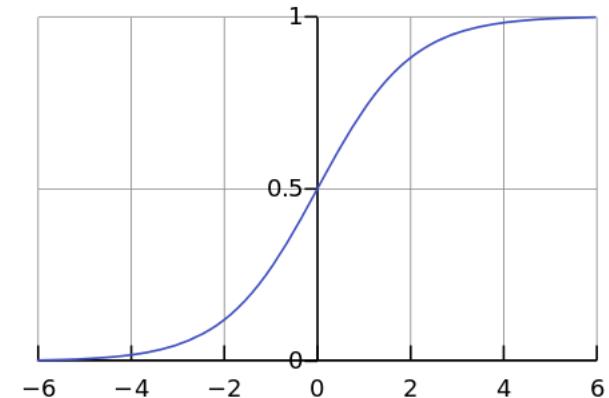


$$f(x) = \frac{1}{1 + e^{-x}}$$





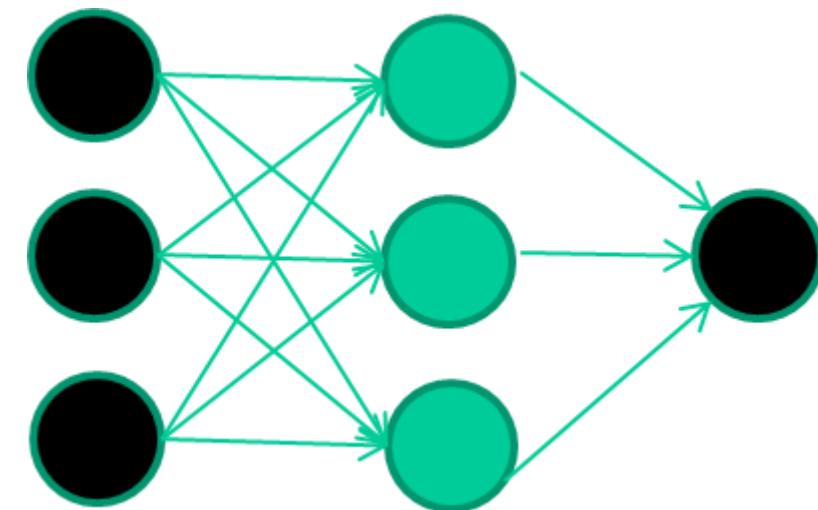
$$f(x) = \frac{1}{1 + e^{-x}}$$



$$x = -0.06 \times 2.7 + 2.5 \times 8.6 + 1.4 \times 0.002 = 21.34$$

A dataset

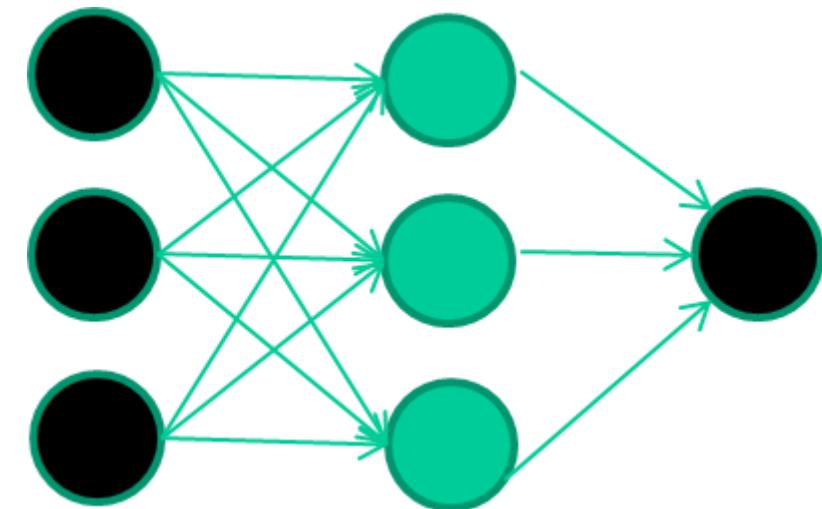
<i>Fields</i>	<i>class</i>
1.4 2.7 1.9	0
3.8 3.4 3.2	0
6.4 2.8 1.7	1
4.1 0.1 0.2	0
etc ...	



Training the neural network

Fields **class**

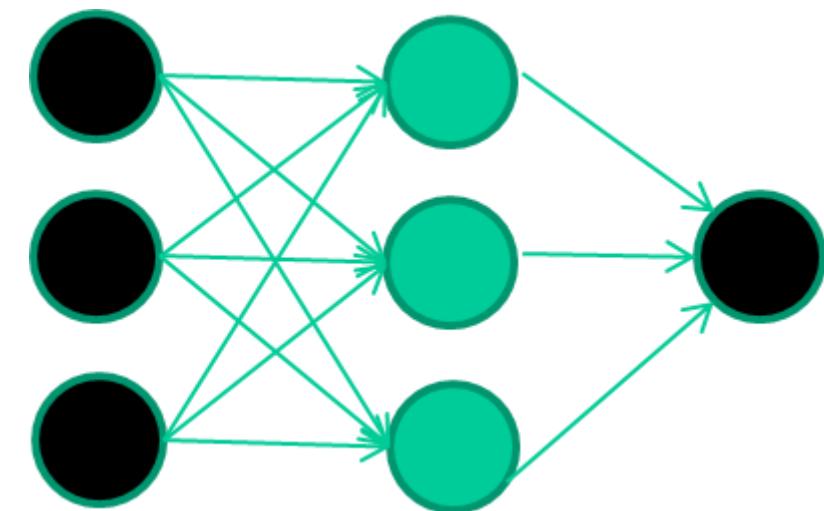
1.4	2.7	1.9	0
3.8	3.4	3.2	0
6.4	2.8	1.7	1
4.1	0.1	0.2	0
etc ...			



Training data

<i>Fields</i>	<i>class</i>		
1.4	2.7	1.9	0
3.8	3.4	3.2	0
6.4	2.8	1.7	1
4.1	0.1	0.2	0
etc ...			

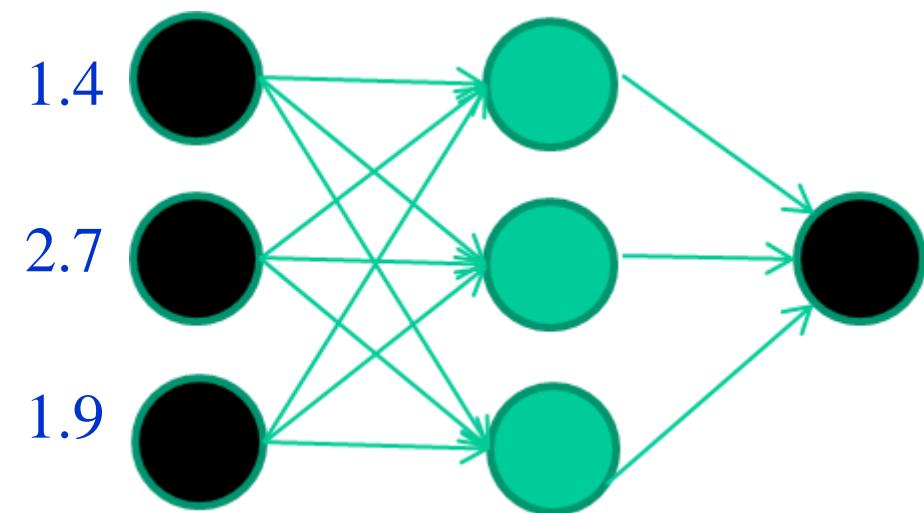
Initialise with random weights



Training data

<i>Fields</i>	<i>class</i>
1.4 2.7 1.9	0
3.8 3.4 3.2	0
6.4 2.8 1.7	1
4.1 0.1 0.2	0
etc ...	

Present a training pattern



Training data

Fields class

1.4	2.7	1.9	0
-----	-----	-----	---

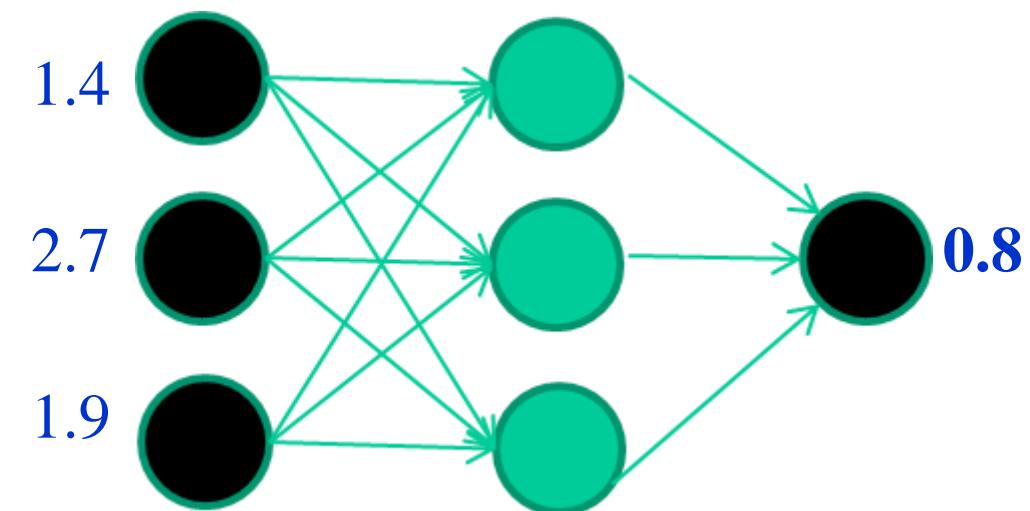
3.8	3.4	3.2	0
-----	-----	-----	---

6.4	2.8	1.7	1
-----	-----	-----	---

4.1	0.1	0.2	0
-----	-----	-----	---

etc ...

Feed it through to get output



Training data

Fields class

Fields	class
1.4 2.7 1.9	0

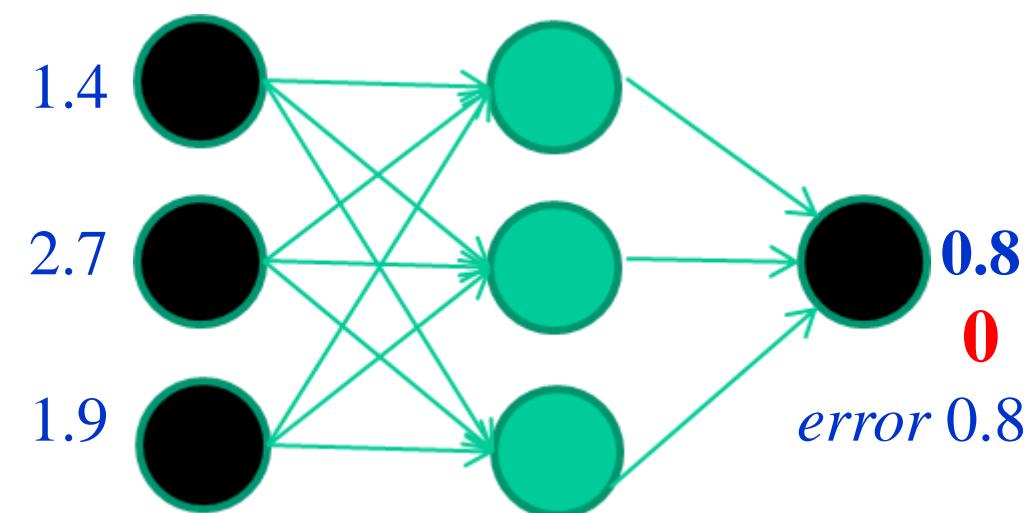
3.8 3.4 3.2	0
-------------	---

6.4 2.8 1.7	1
-------------	---

4.1 0.1 0.2	0
-------------	---

etc ...

Compare with target output



Training data

Fields class

1.4	2.7	1.9	0
-----	-----	-----	---

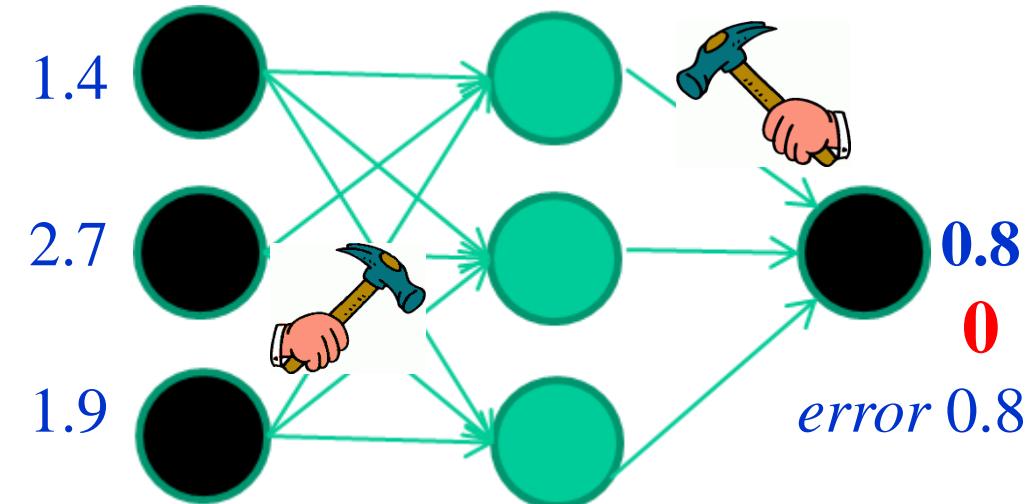
3.8 3.4 3.2 0

6.4 2.8 1.7 1

4.1 0.1 0.2 0

etc ...

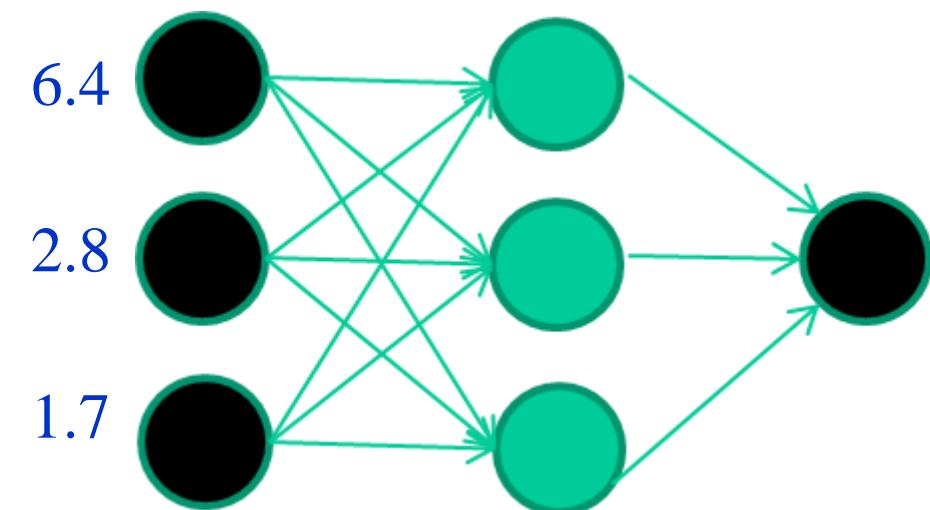
Adjust weights based on error



Training data

<i>Fields</i>	<i>class</i>
1.4 2.7 1.9	0
3.8 3.4 3.2	0
6.4 2.8 1.7	1
4.1 0.1 0.2	0
etc ...	

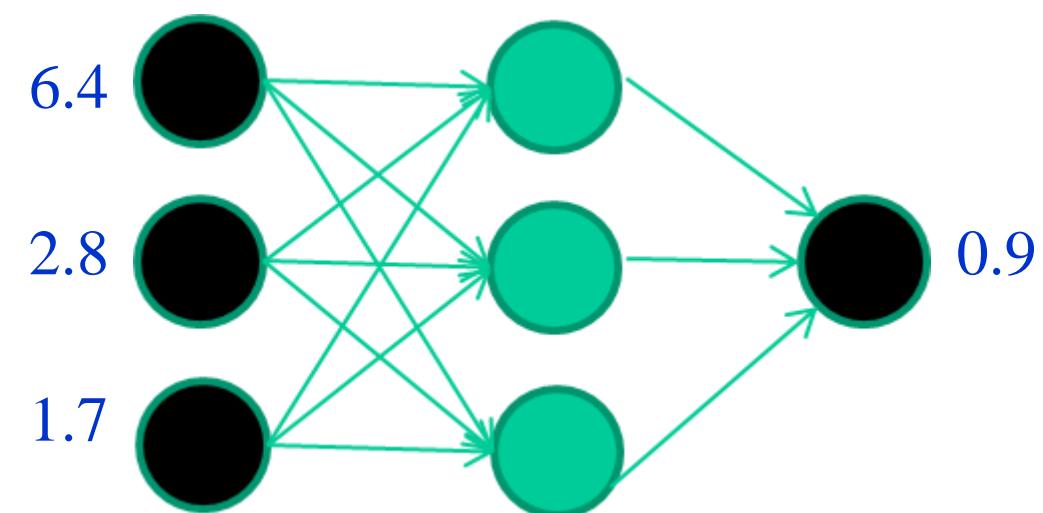
Present a training pattern



Training data

<i>Fields</i>	<i>class</i>
1.4 2.7 1.9	0
3.8 3.4 3.2	0
6.4 2.8 1.7	1
4.1 0.1 0.2	0
etc ...	

Feed it through to get output



Training data

Fields class

1.4 2.7 1.9 0

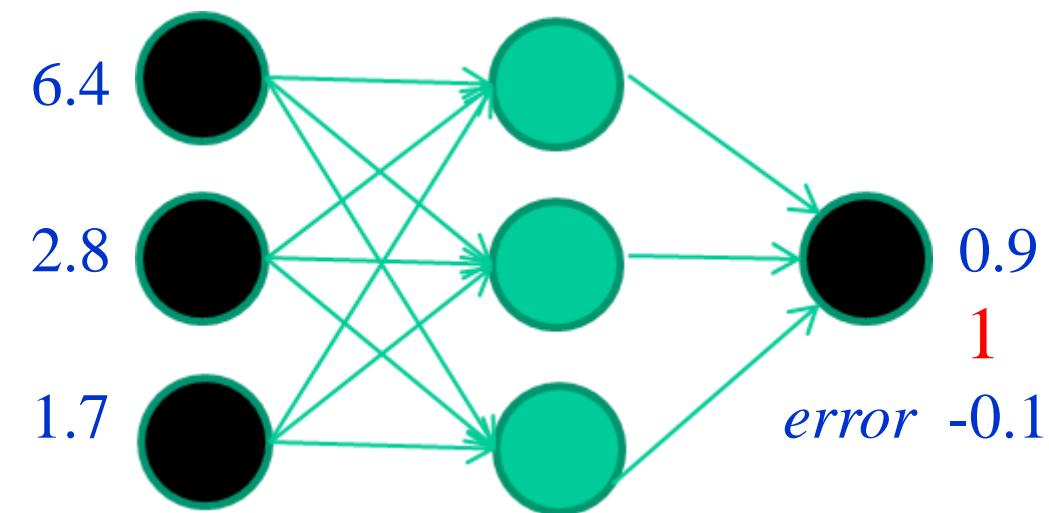
3.8 3.4 3.2 0

6.4 2.8 1.7 1

4.1 0.1 0.2 0

etc ...

Compare with target output



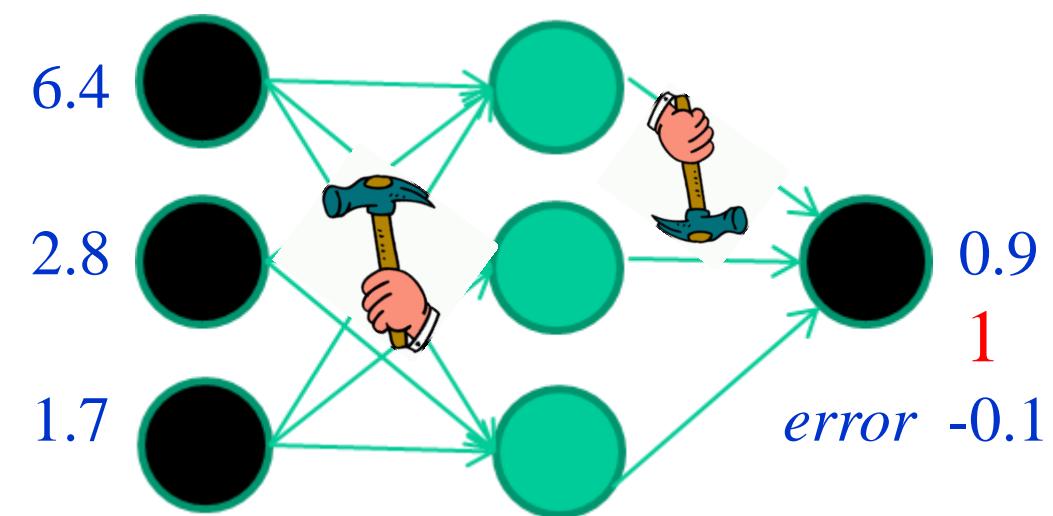
Training data

Fields *class*

1.4	2.7	1.9	0
3.8	3.4	3.2	0
6.4	2.8	1.7	1
4.1	0.1	0.2	0

etc ...

Adjust weights based on error

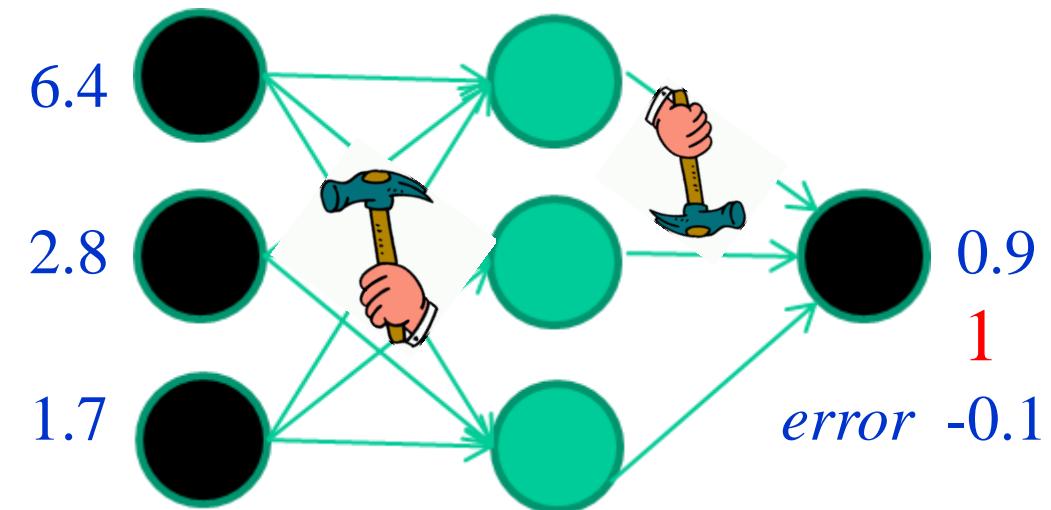


Training data

Fields *class*

1.4	2.7	1.9	0
3.8	3.4	3.2	0
6.4	2.8	1.7	1
4.1	0.1	0.2	0
etc ...			

And so on

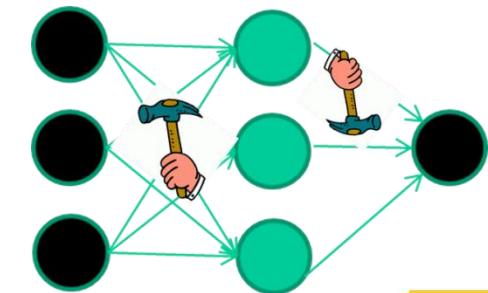


Ulangi ini ribuan, mungkin jutaan kali - setiap kali mengambil contoh pelatihan acak, dan membuat sedikit penyesuaian berat badan

Algoritma untuk penyesuaian berat dirancang untuk membuatnya perubahan yang akan mengurangi kesalahan

Inti yang akan dilakukan

- weight-learning algorithms (algoritma pembelajaran berat) untuk NNs are dumb
- mereka bekerja dengan membuat ribuan penyesuaian kecil, masing-masing membuat jaringan melakukan lebih baik pada pola terbaru, tetapi mungkin sedikit lebih buruk pada banyak lainnya
- tetapi, karena keberuntungan, akhirnya ini cenderung cukup baik pelajari pengklasifikasi yang efektif untuk banyak aplikasi nyata



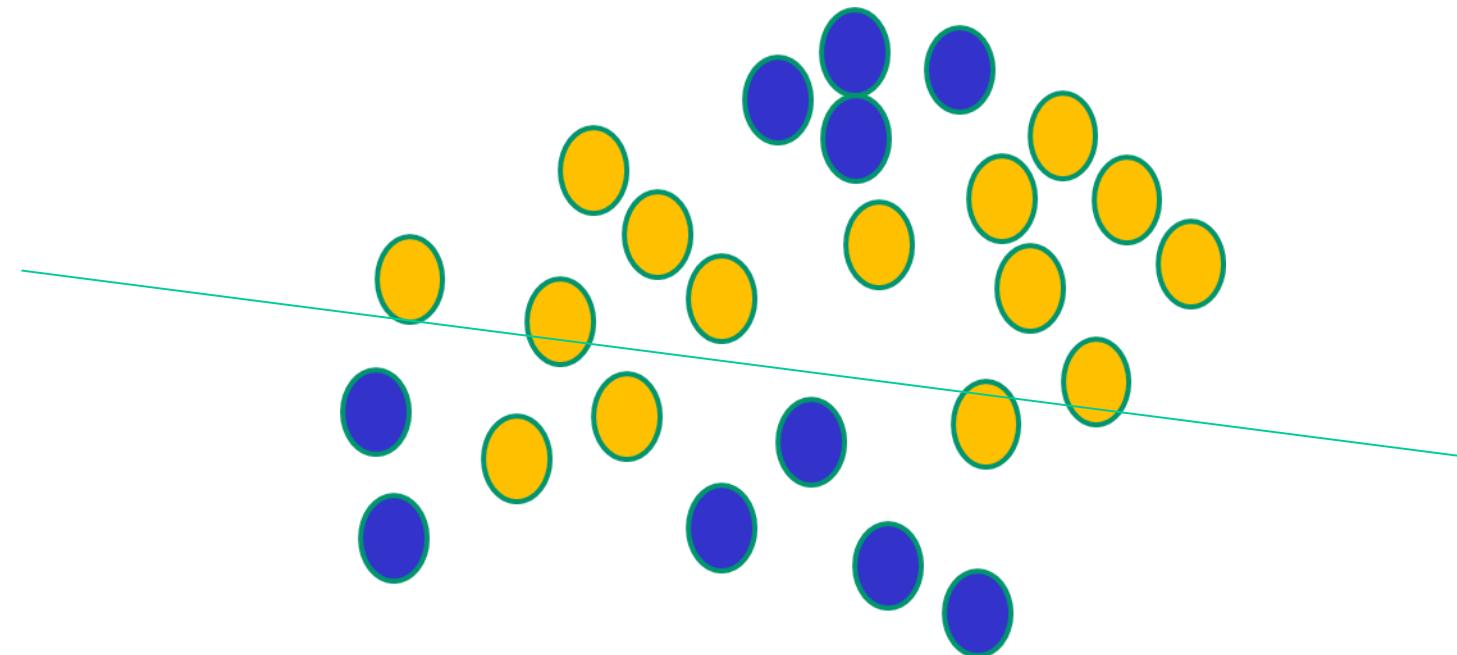
Beberapa inti lainnya

Detail dari standar algoritma pembelajaran bobot NN
- nanti

Jika $f(x)$ non-linear, jaringan dengan 1 lapisan tersembunyi, secara teori, dapat mempelajari dengan sempurna masalah klasifikasi. Ada set bobot yang dapat menghasilkan target dari input. Masalahnya adalah menemukannya.

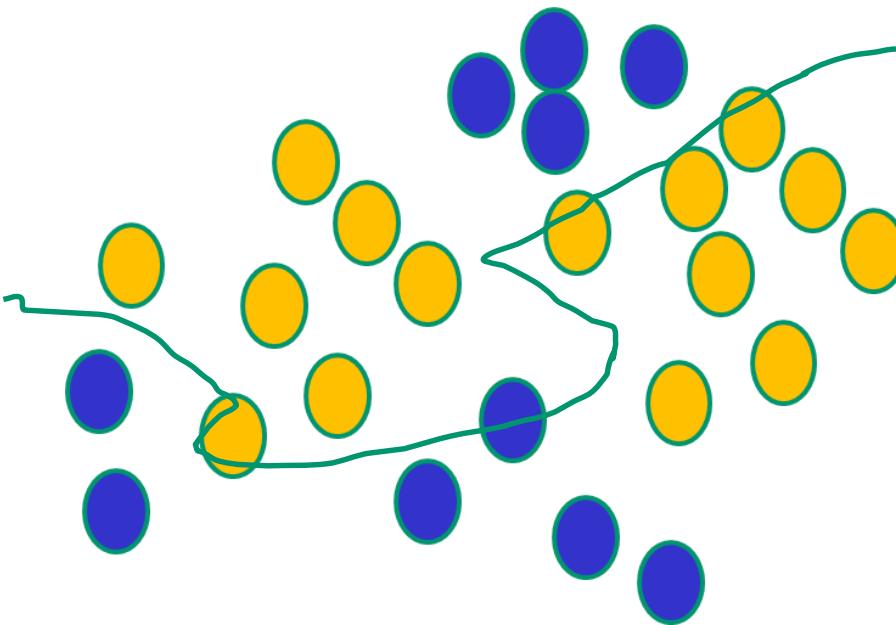
Beberapa poin lain nya

Jika $f(x)$ linier, NN hanya dapat menggambar batas keputusan langsung (bahkan jika ada banyak lapisan unit)



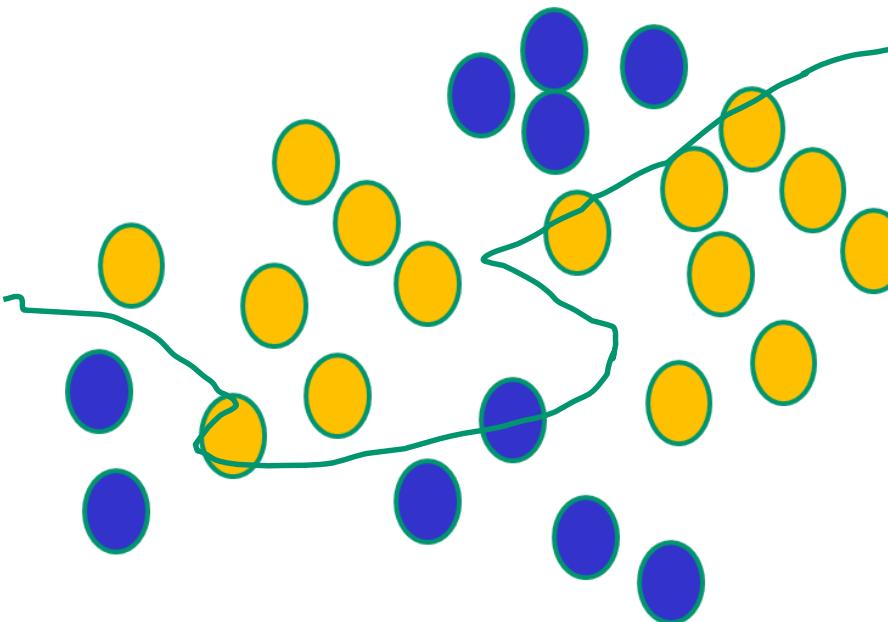
Beberapa poin lain nya

NN menggunakan nonlinear $f(x)$ sehingga mereka dapat menggambar batas yang kompleks, tetapi menjaga data tidak berubah

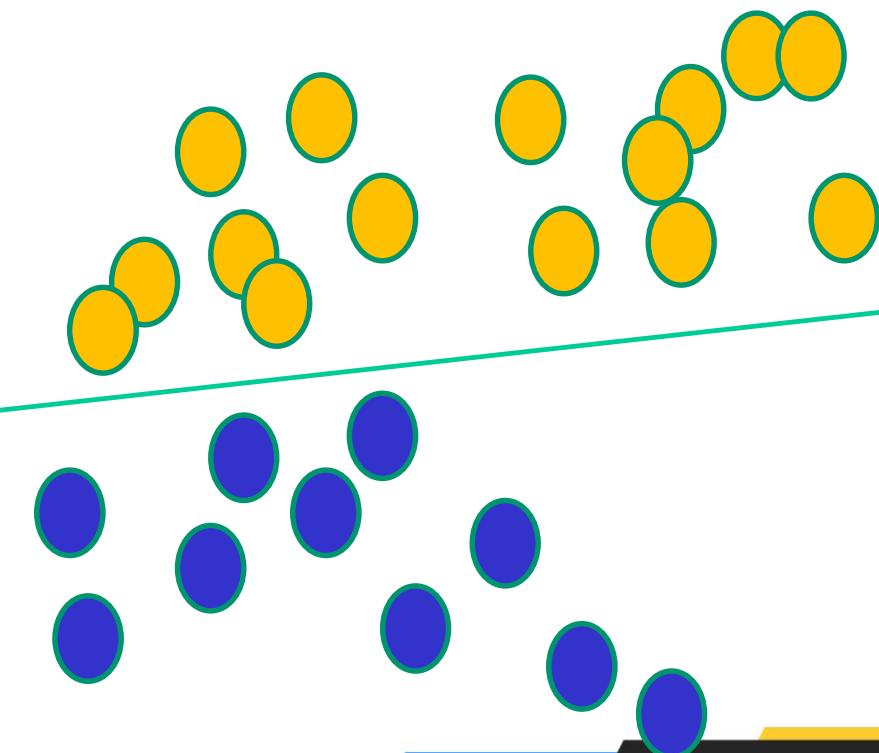


Beberapa poin lain nya

NN menggunakan nonlinear $f(x)$ sehingga mereka dapat menggambar batas yang kompleks, tetapi menjaga data tidak berubah



SVM hanya menggambar garis lurus tetapi mereka mengubah data terlebih dahulu dengan cara yang membuat itu OK



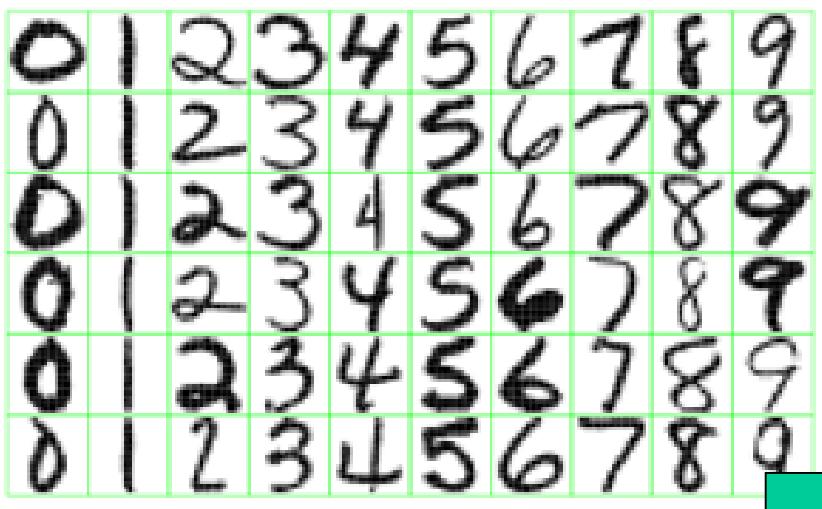
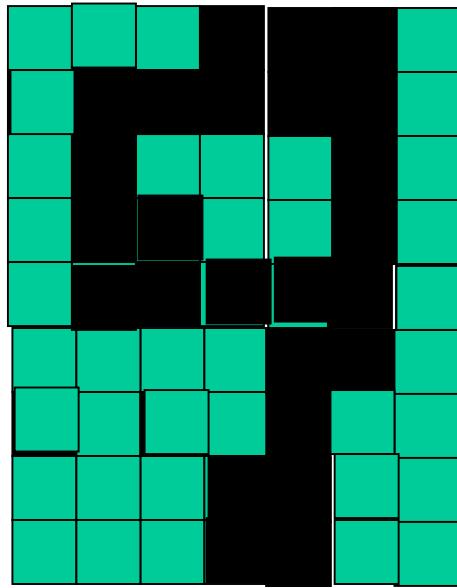
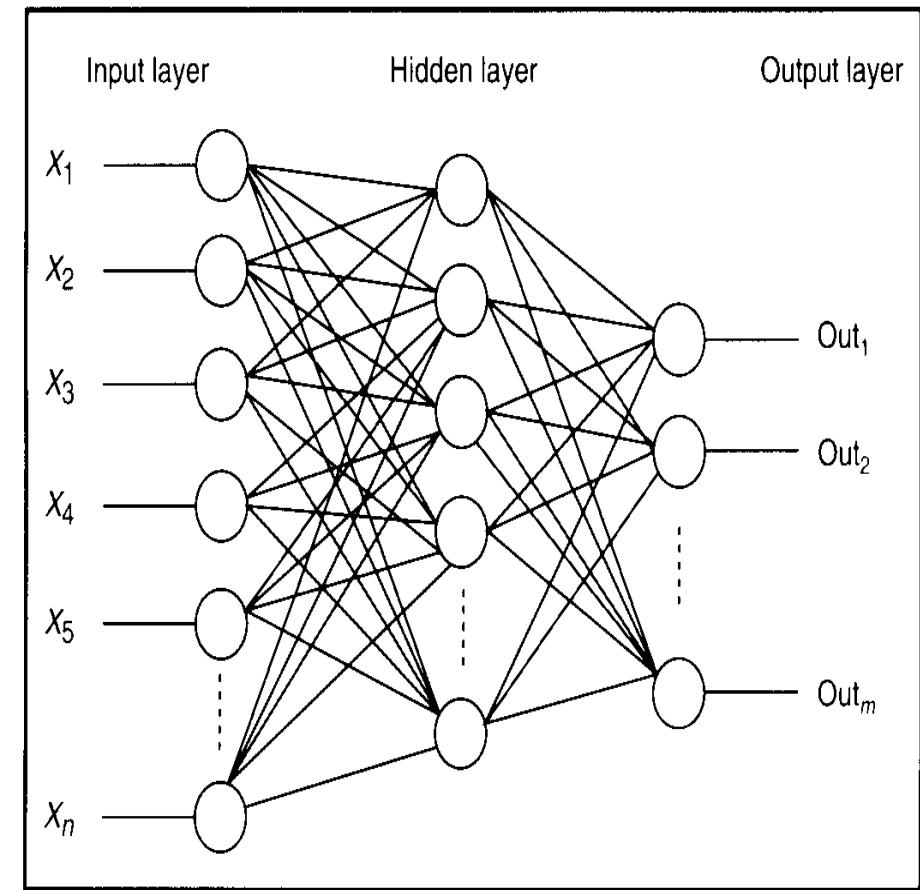


Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.



Feature detectors



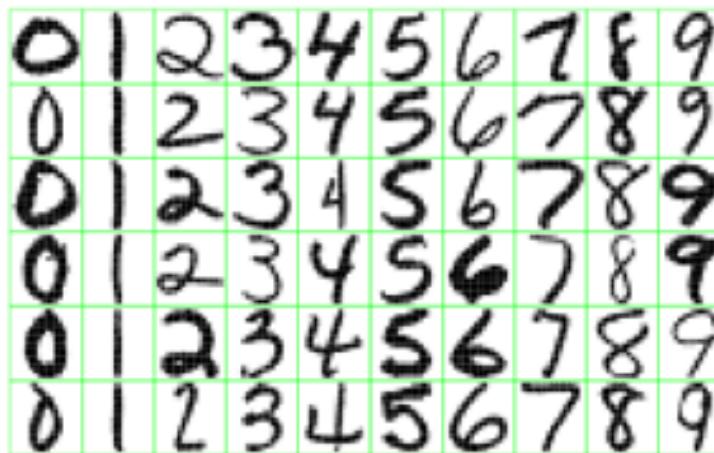
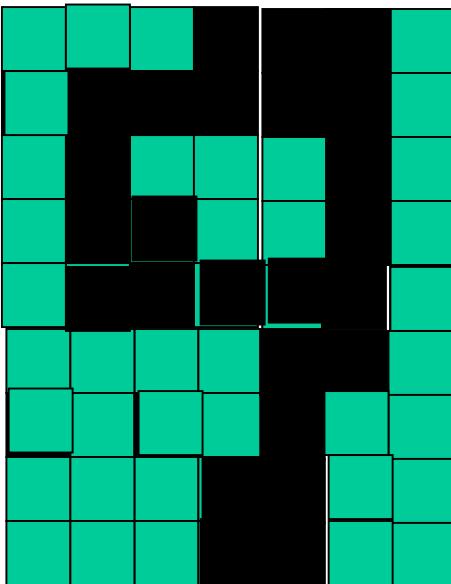
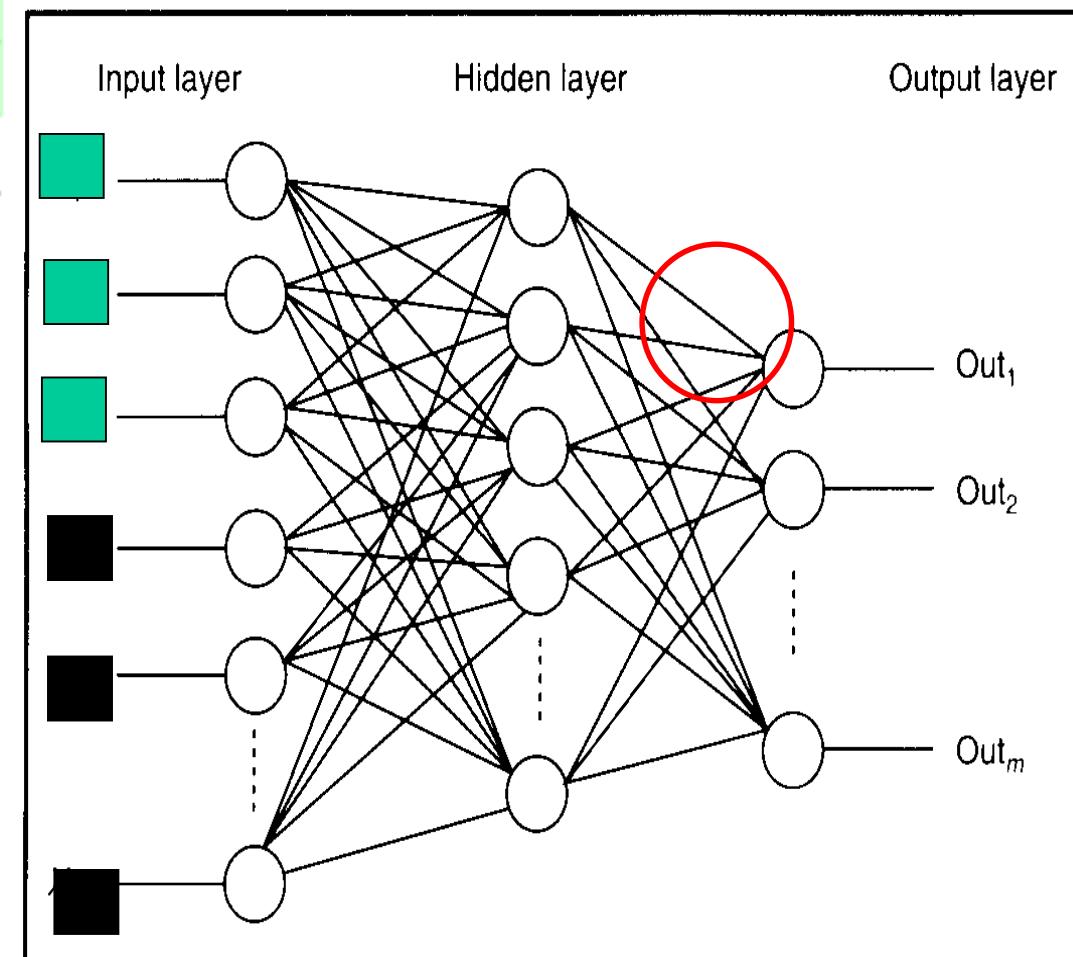


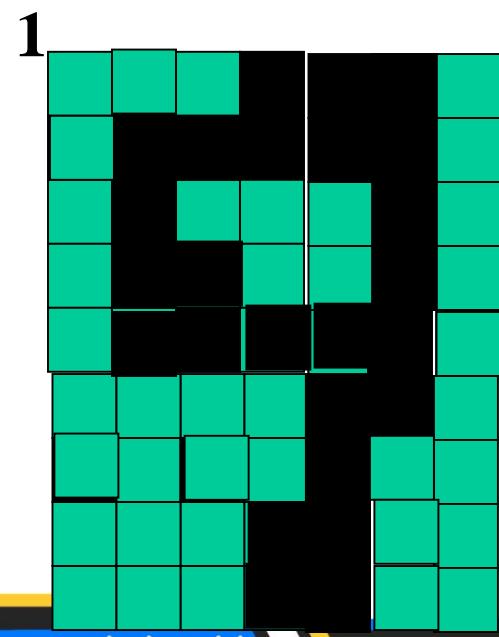
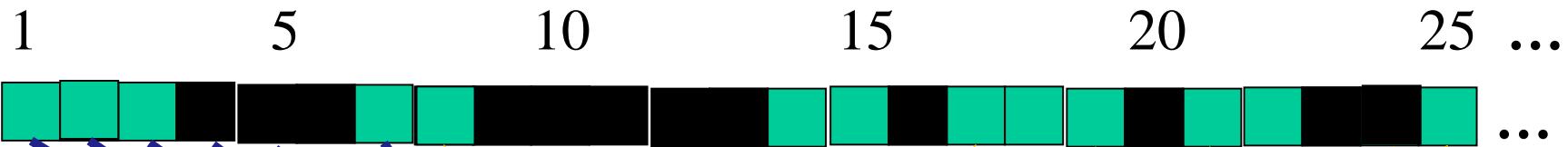
Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.



*what is this
unit doing?*



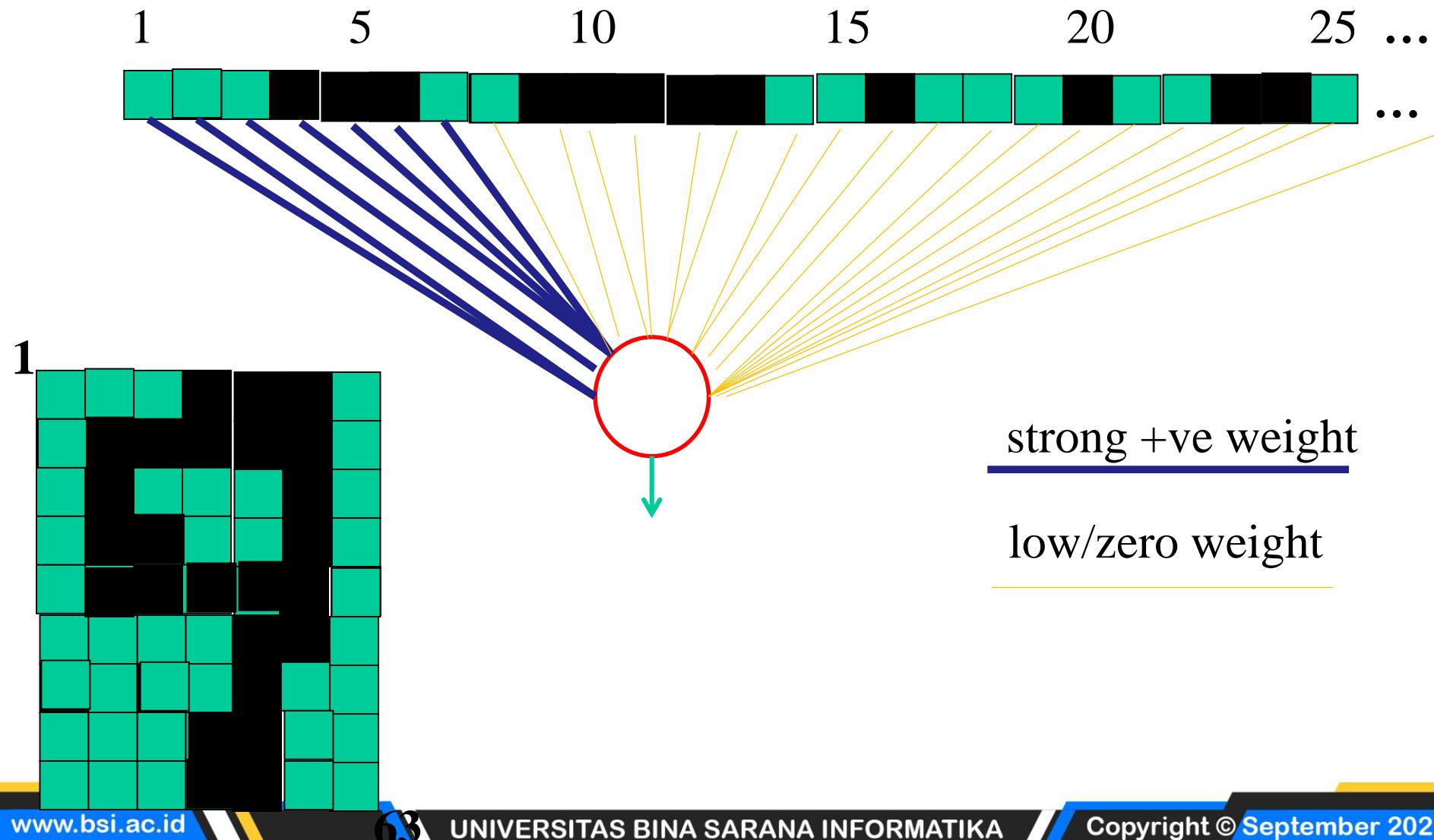
Unit layer tersembunyi menjadi pendeksi fitur yang diatur sendi



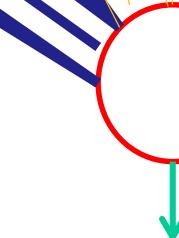
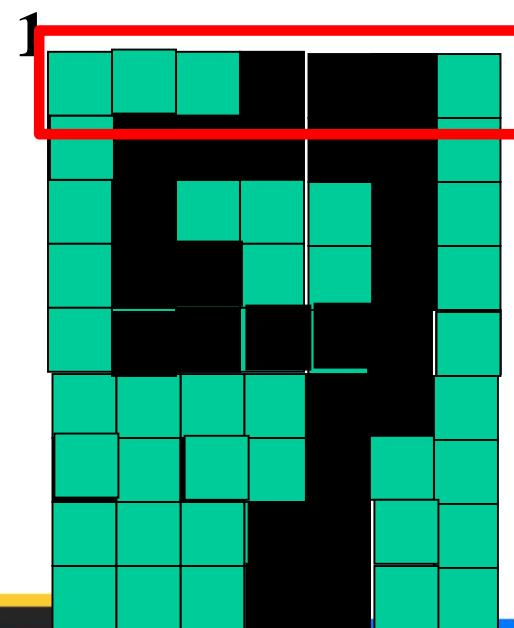
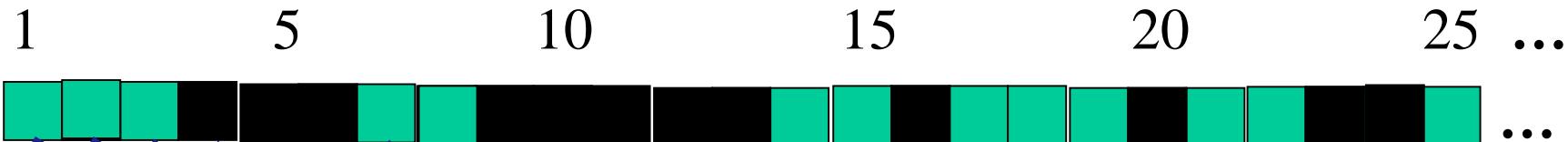
strong +ve weight

low/zero weight

Apa yang dideteksi unit ini?



Apa yang dideteksi unit ini?

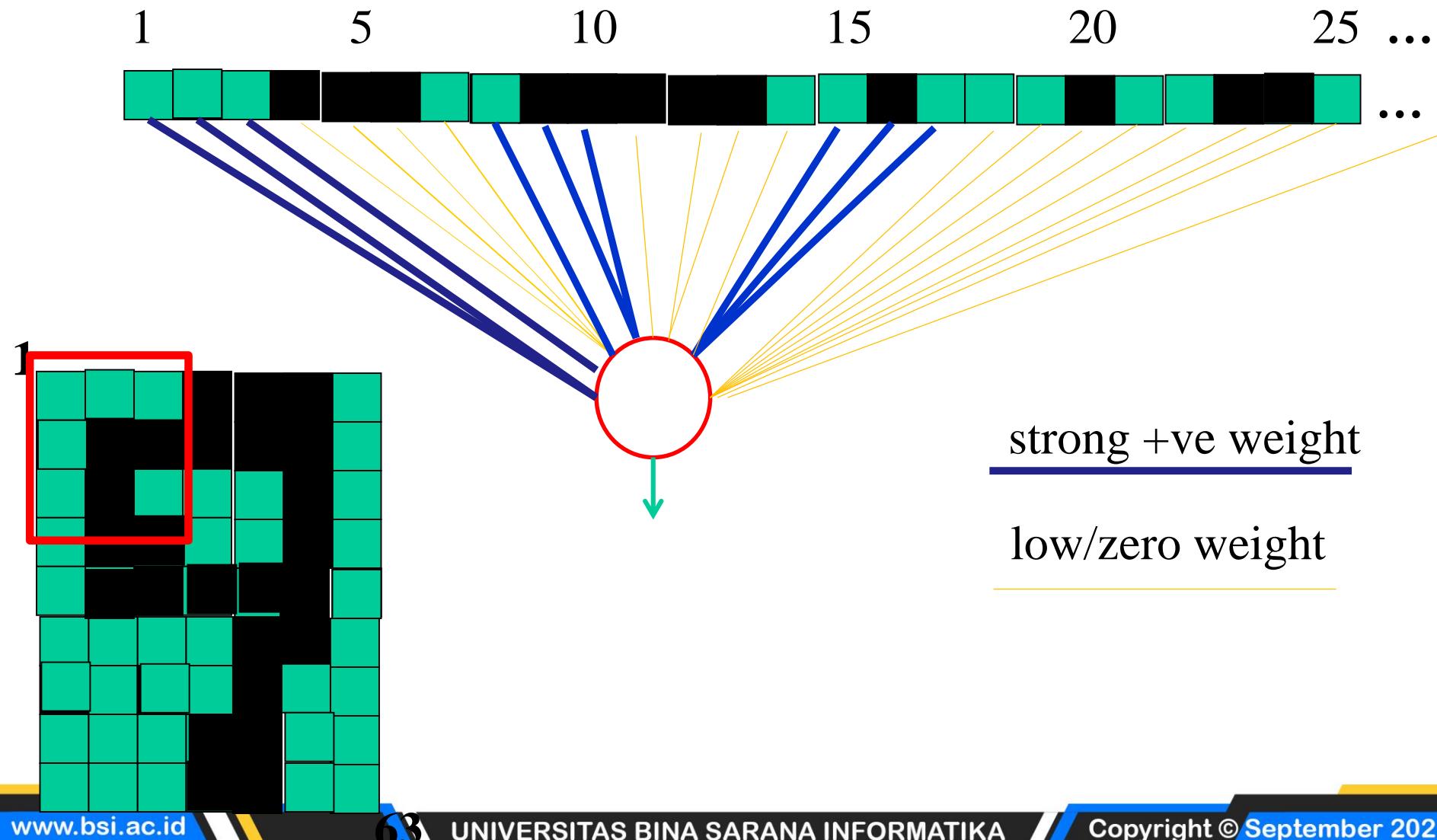


strong +ve weight

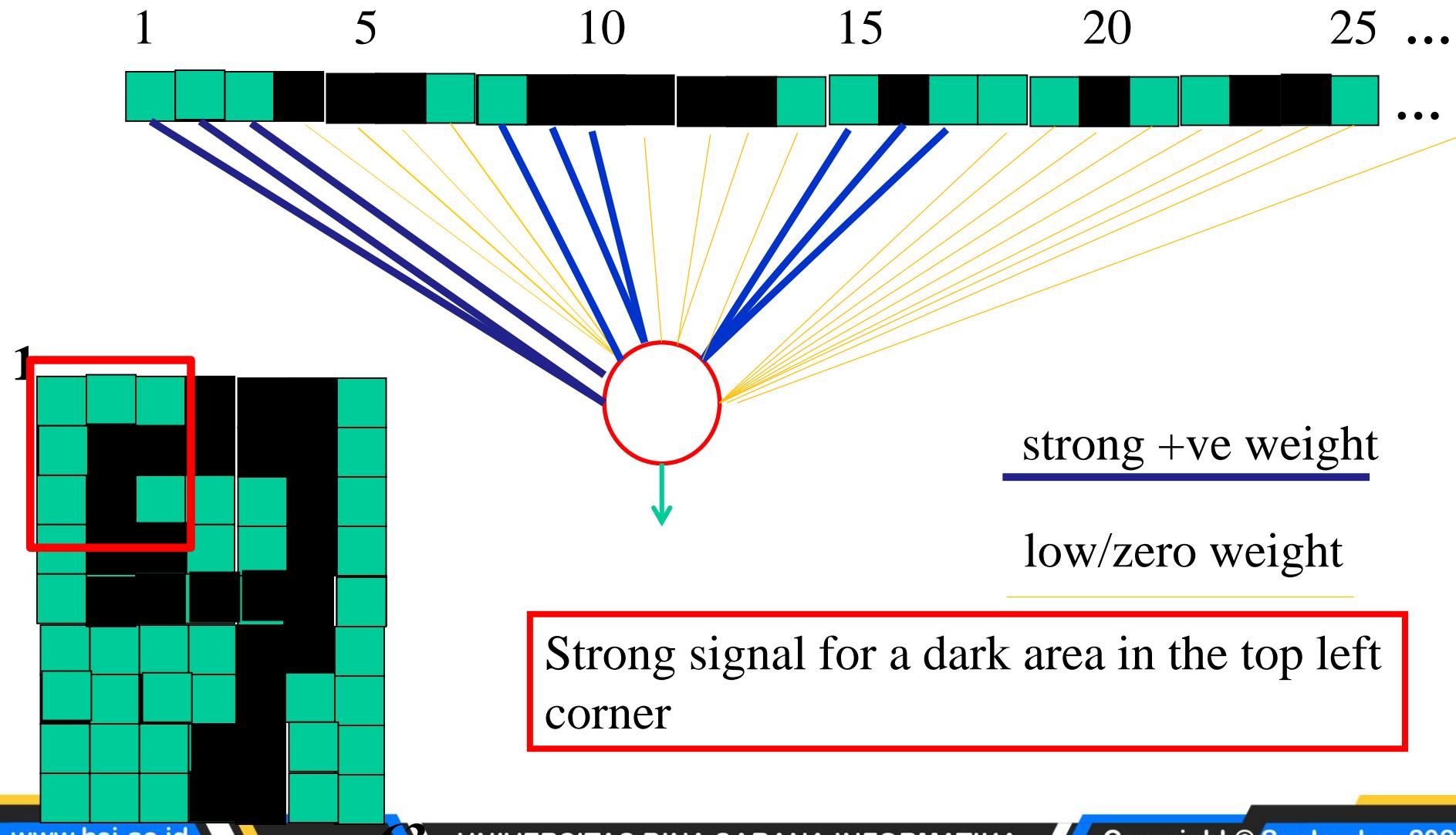
low/zero weight

it will send strong signal for a horizontal line in the top row, ignoring everywhere else

Apa yang dideteksi unit ini?



Apa yang dideteksi unit ini?



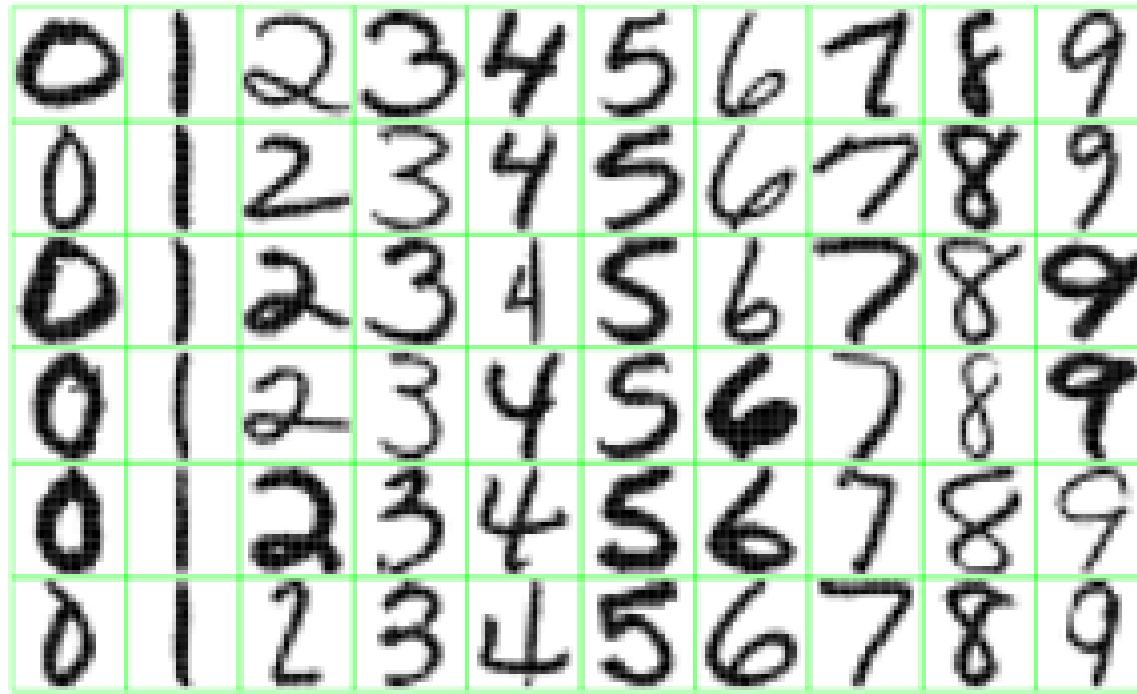


Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.

What features might you expect a good NN to learn, when trained with data like this?

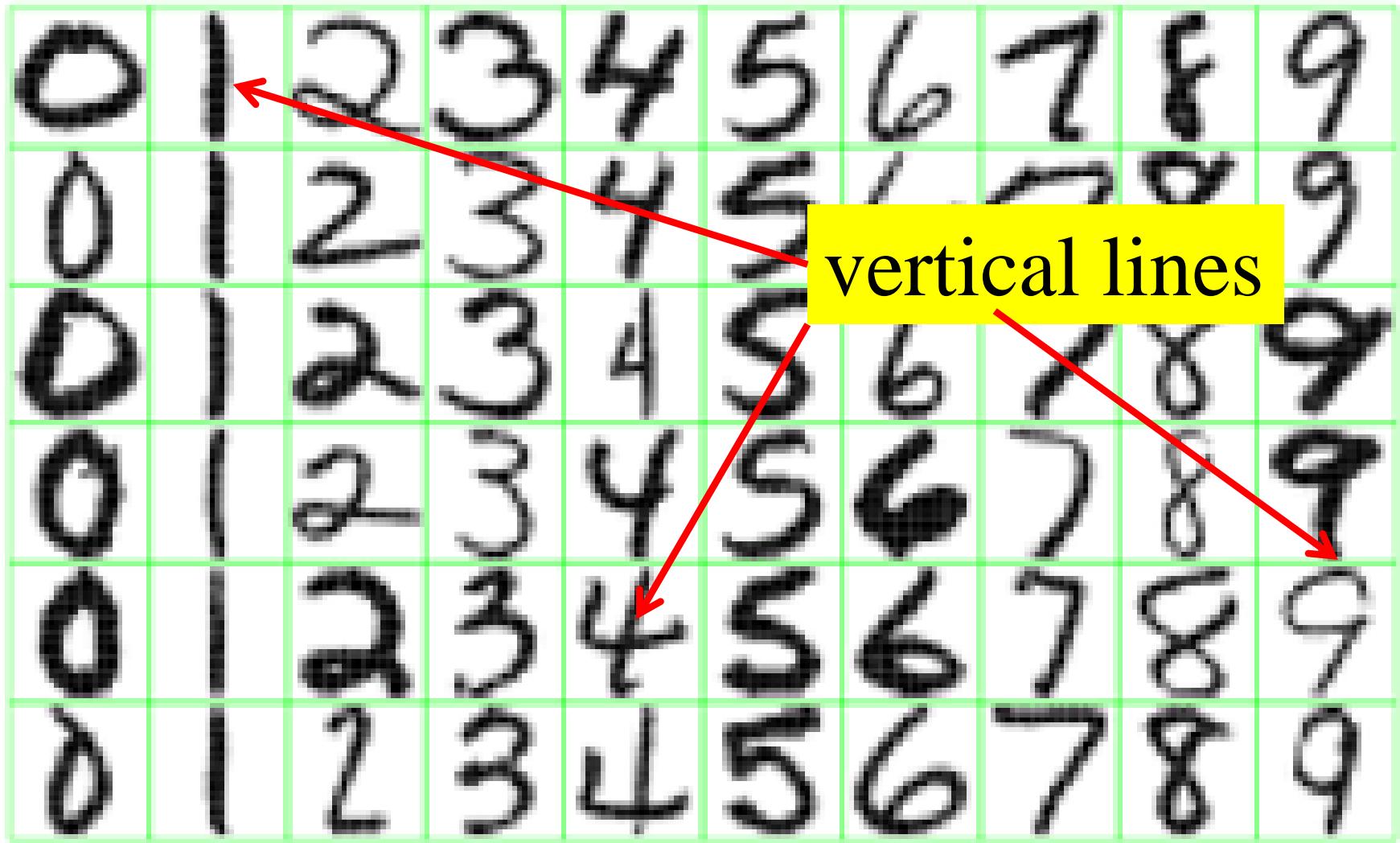


Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.

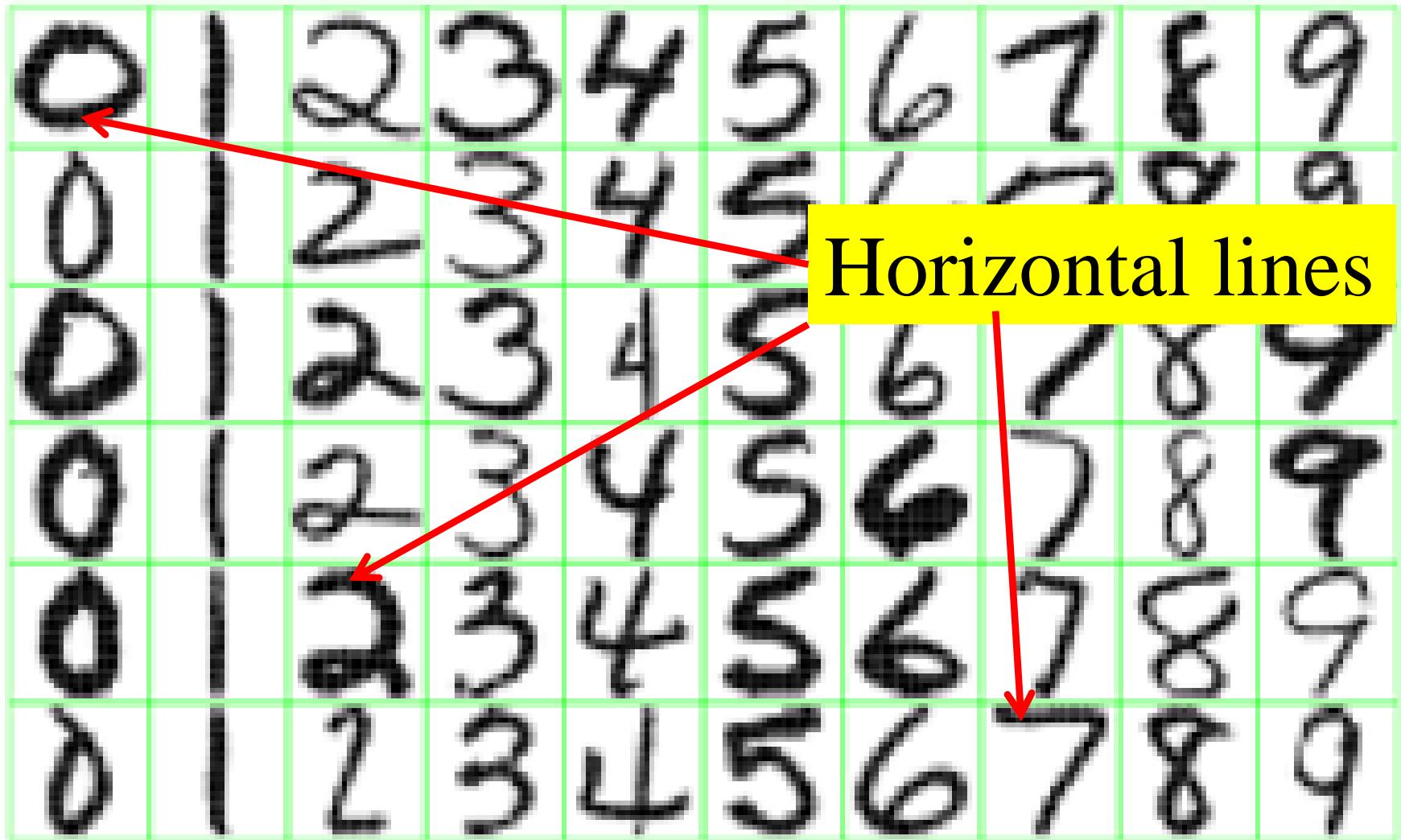


Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.

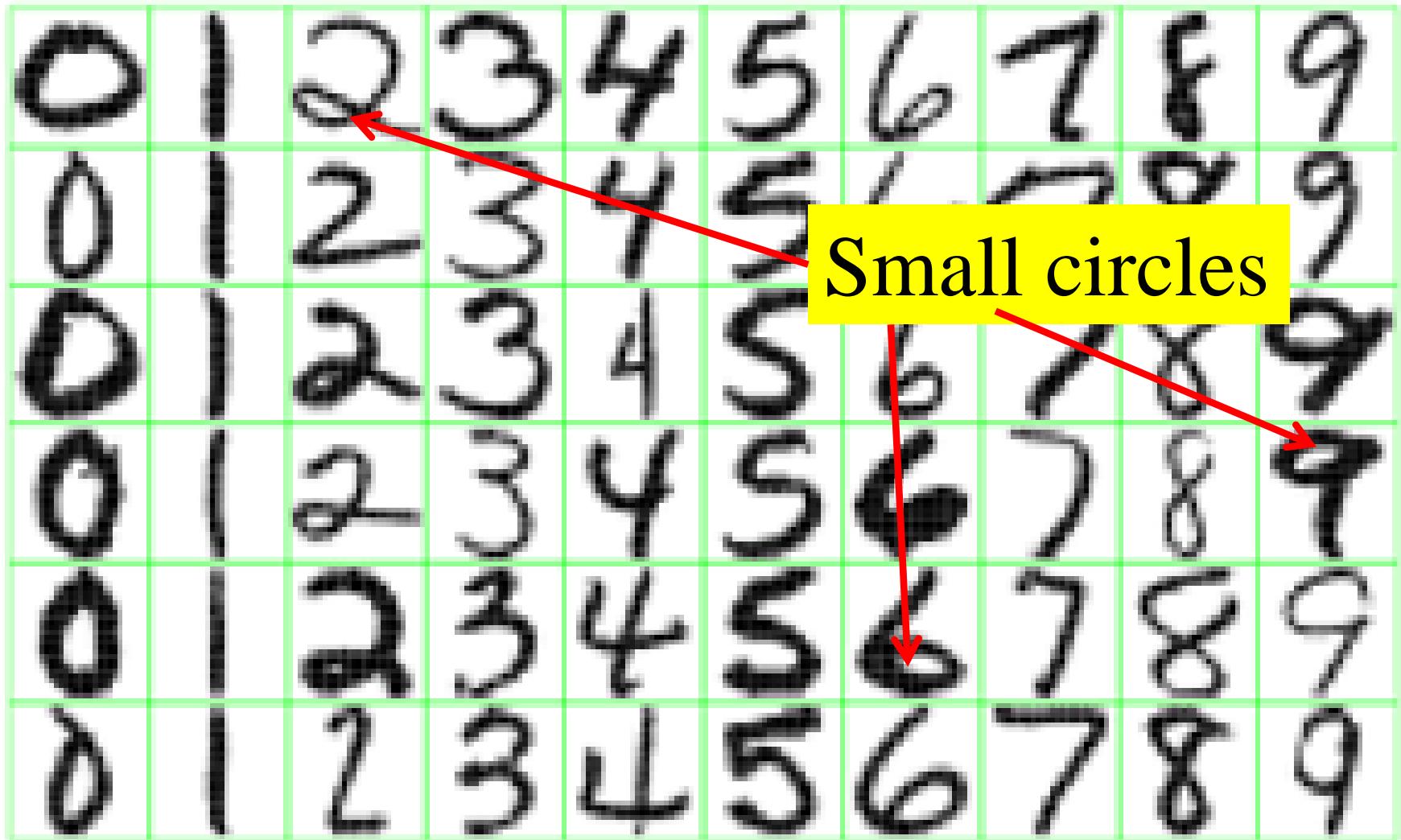
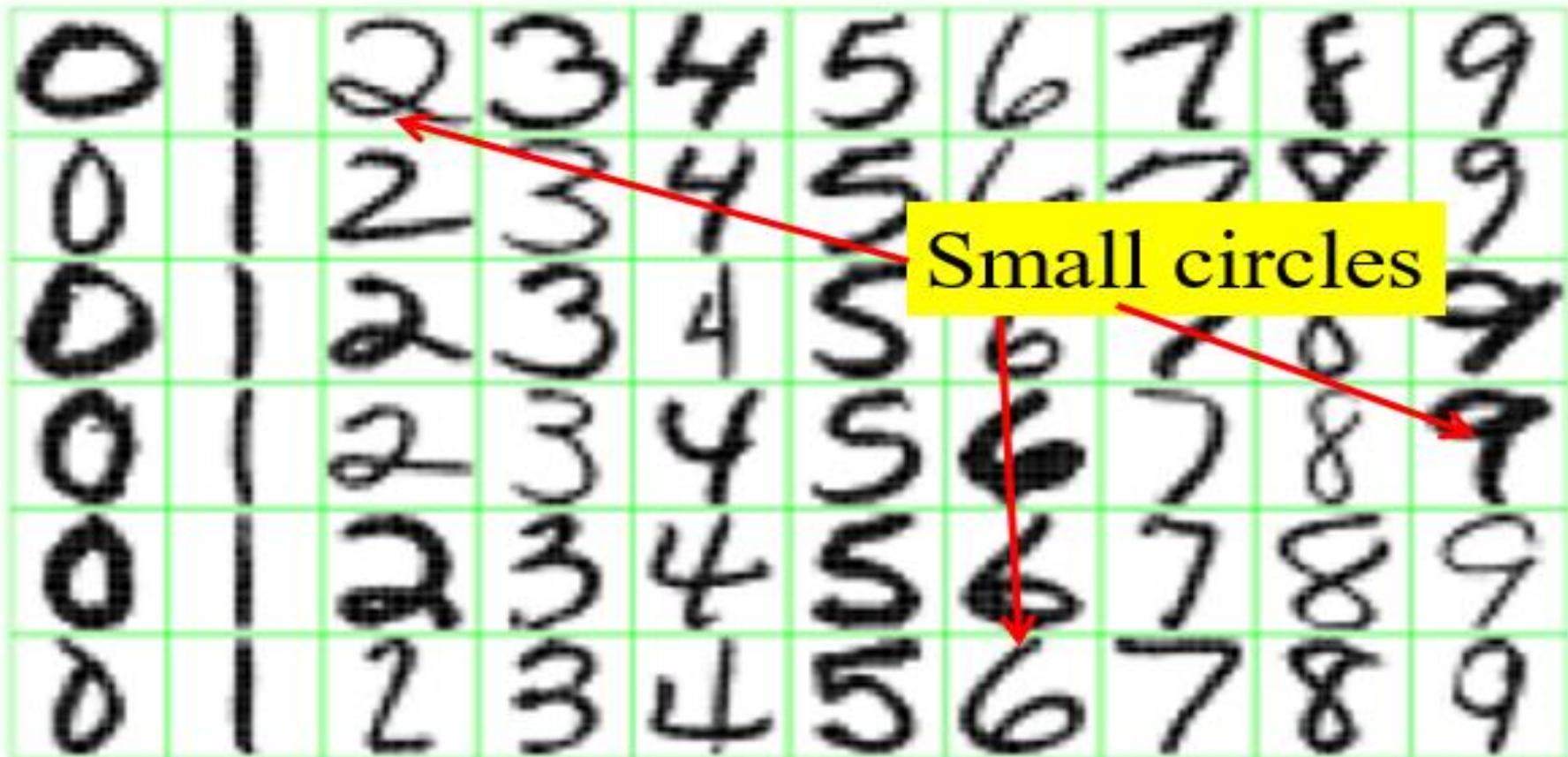


Figure 1.2: Examples of handwritten digits from U.S. postal envelopes.

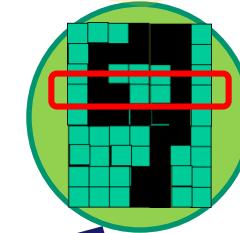
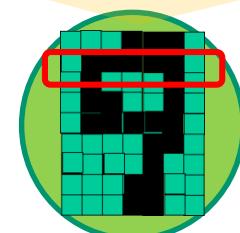
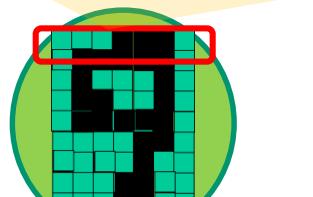


Tapi bagaimana dengan invariant posisi ???
detektor unit contoh diatas terikat bagian tertentu dari gambar

successive layers can learn higher-level features ...

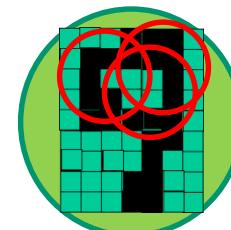
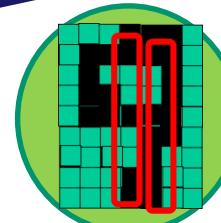
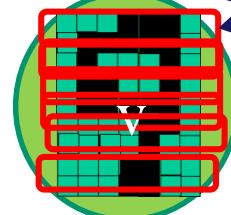


detect lines in
Specific positions



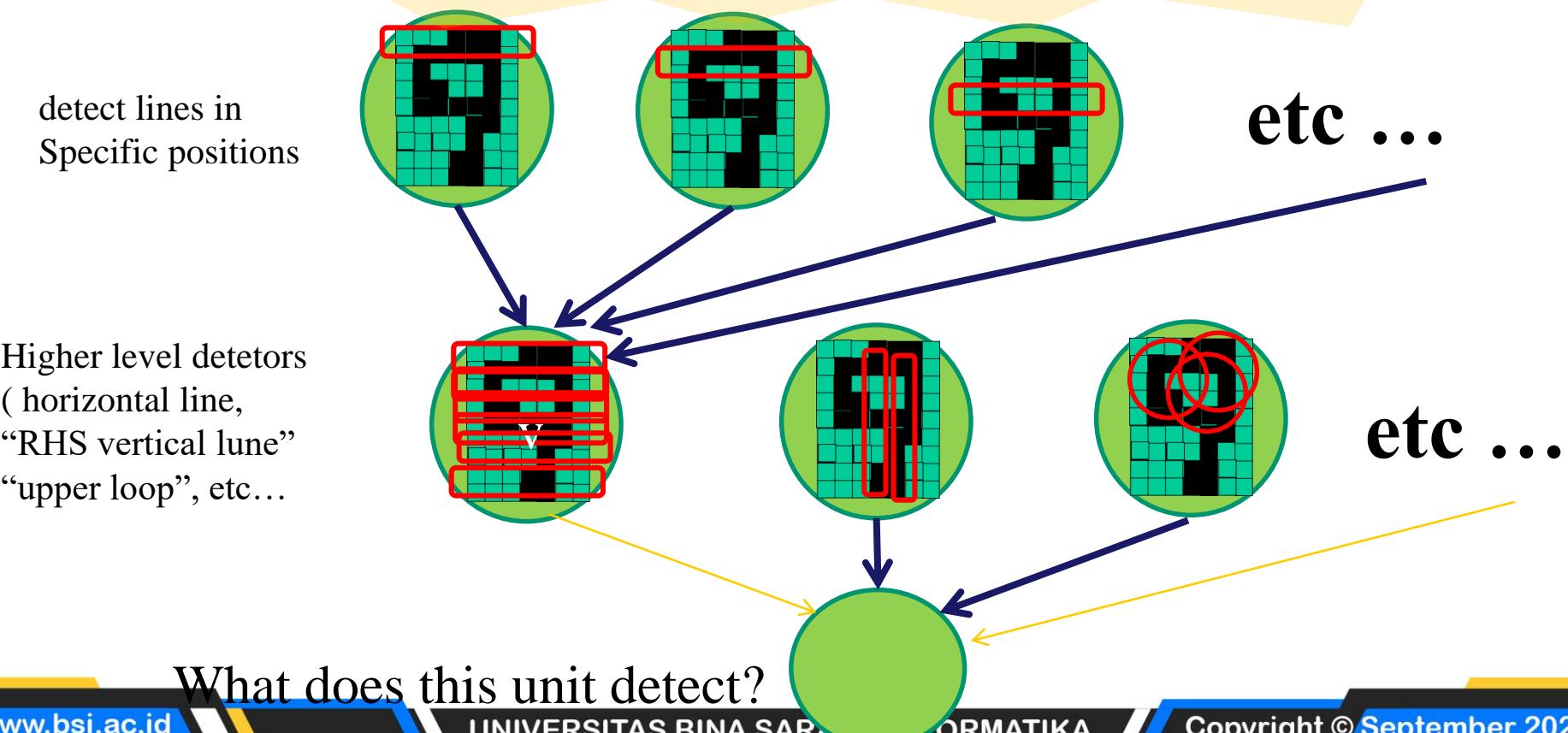
etc ...

Higher level detectors
(horizontal line,
"RHS vertical lune"
"upper loop", etc...)

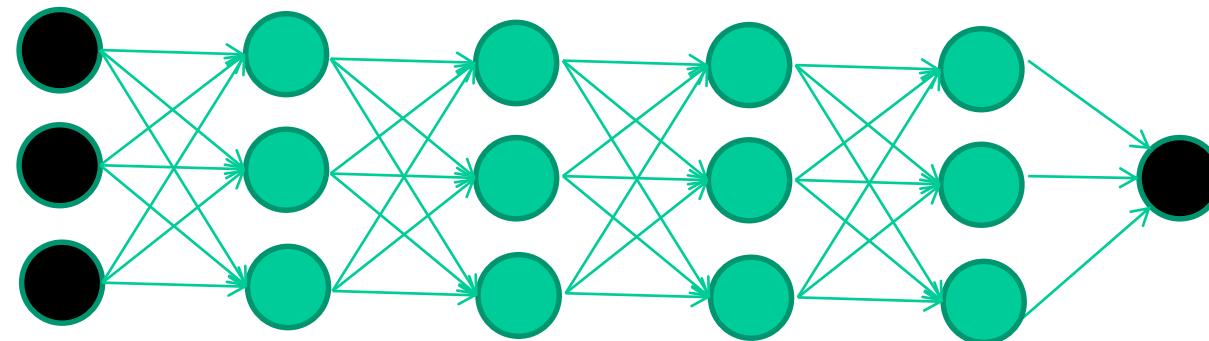


etc ...

successive layers can learn higher-level features ...

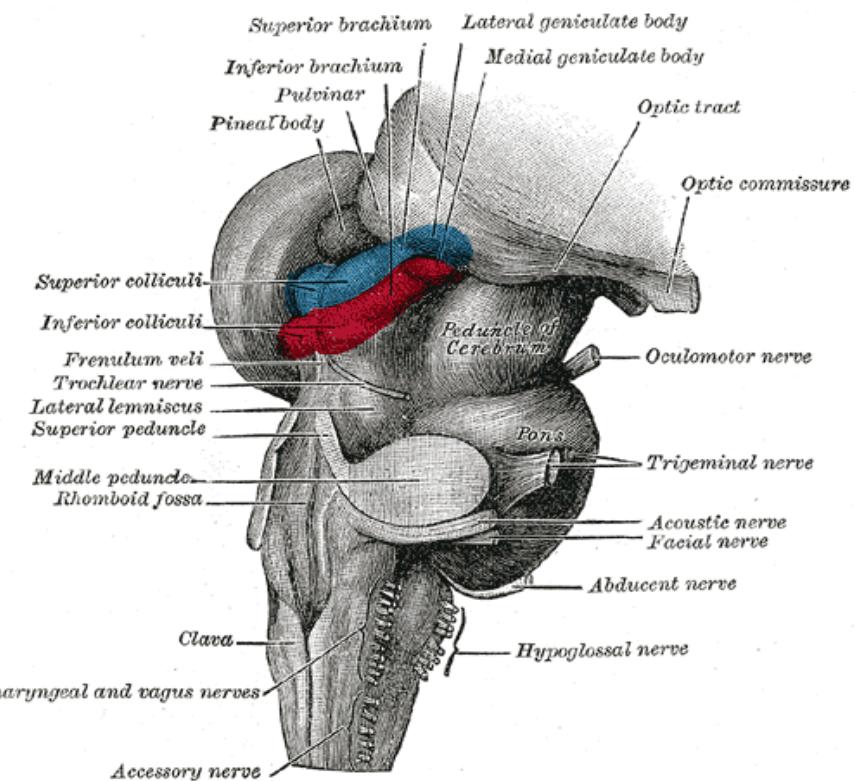
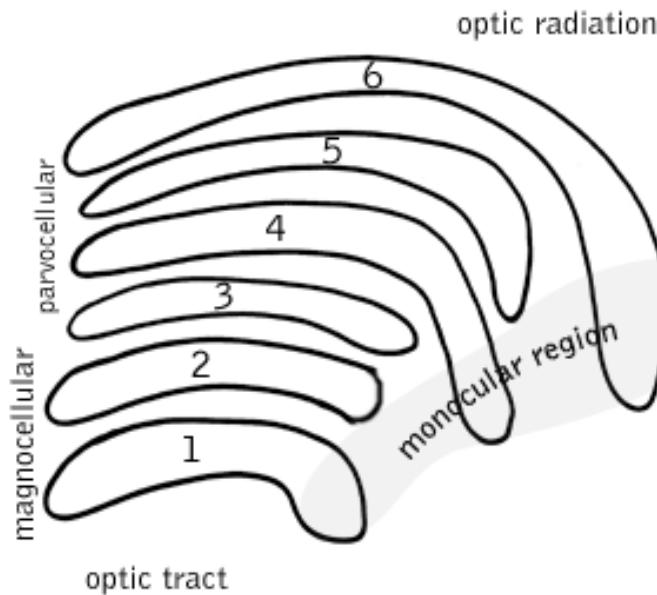


So: multiple layers make sense



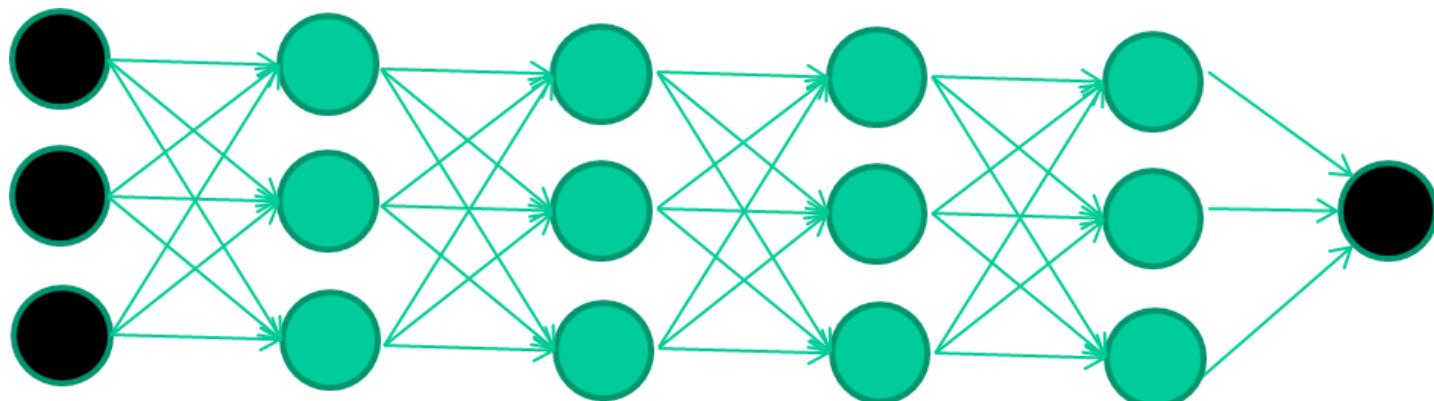
So: multiple layers make sense

Your brain works that way

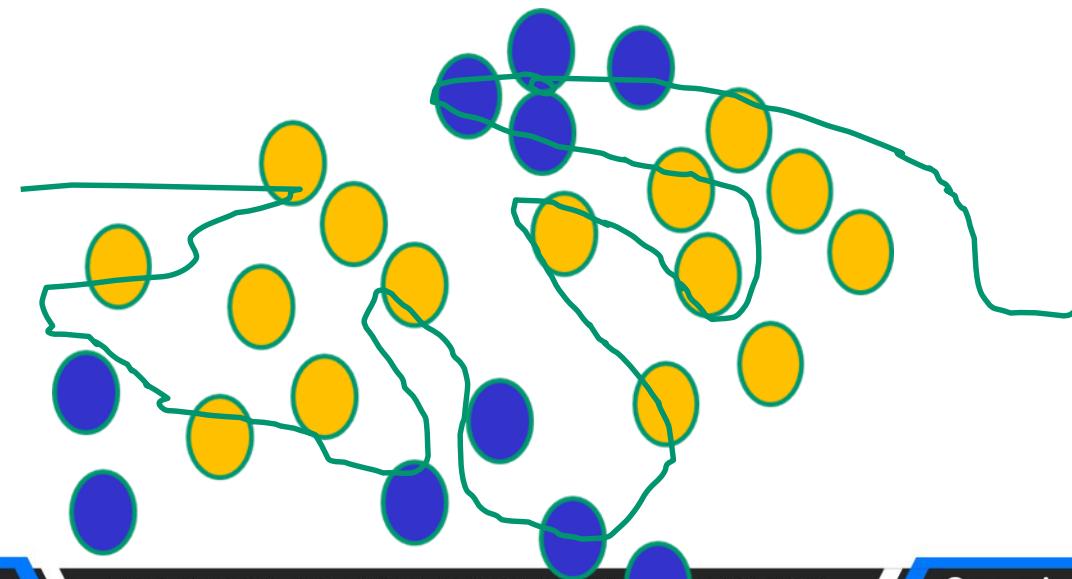
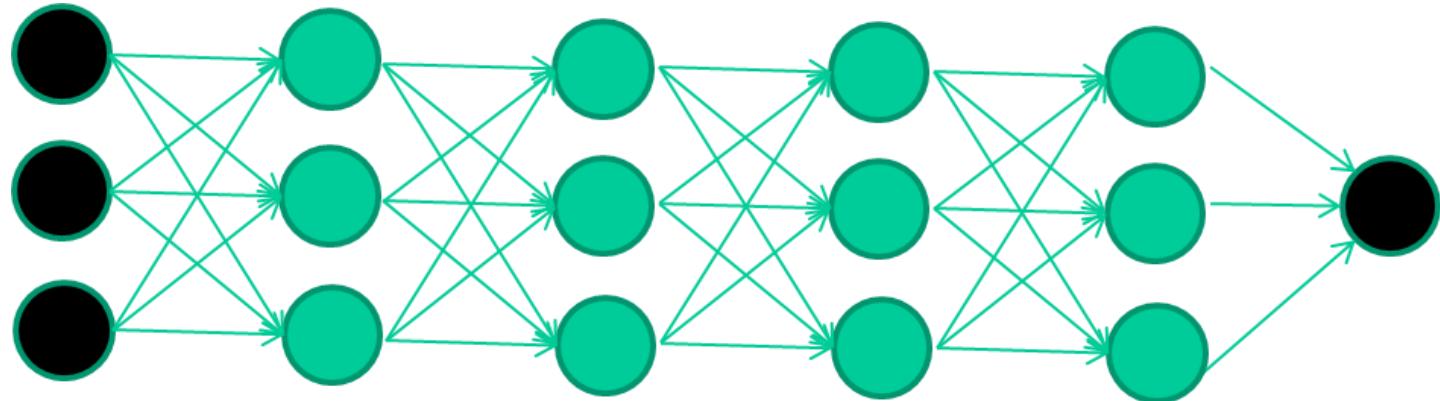


So: *multiple layers make sense*

Banyak-lapisan Arsitektur Neural Network (jaringan saraf) harus mampu mempelajari fitur-fitur mendasar yang sebenarnya dan 'logika fitur', dan karenanya menggeneralisasi dengan sangat baik

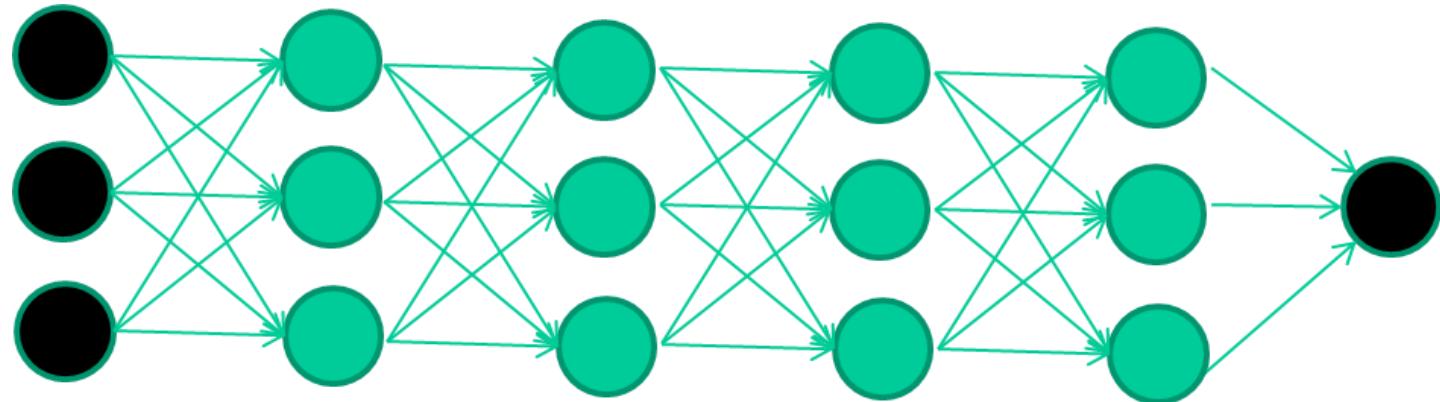


Tetapi, hingga saat ini, algoritma pembelajaran berat tidak berfungsi pada arsitektur multi-layer

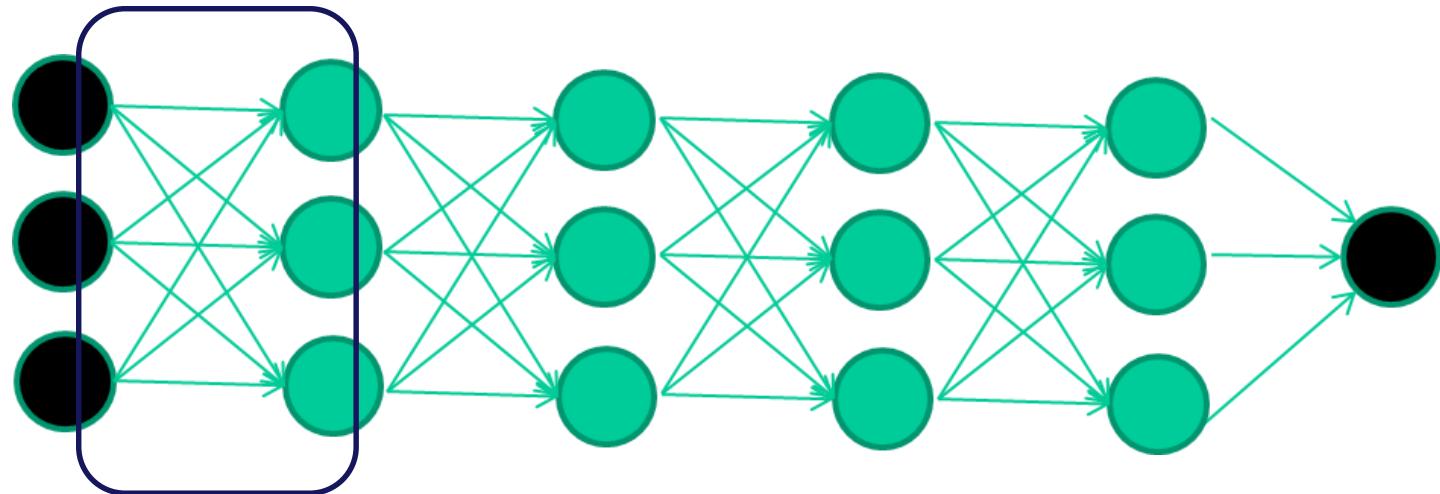


Kemudian datanglah “Deep Learning”

The new way to train multi-layer NNs...

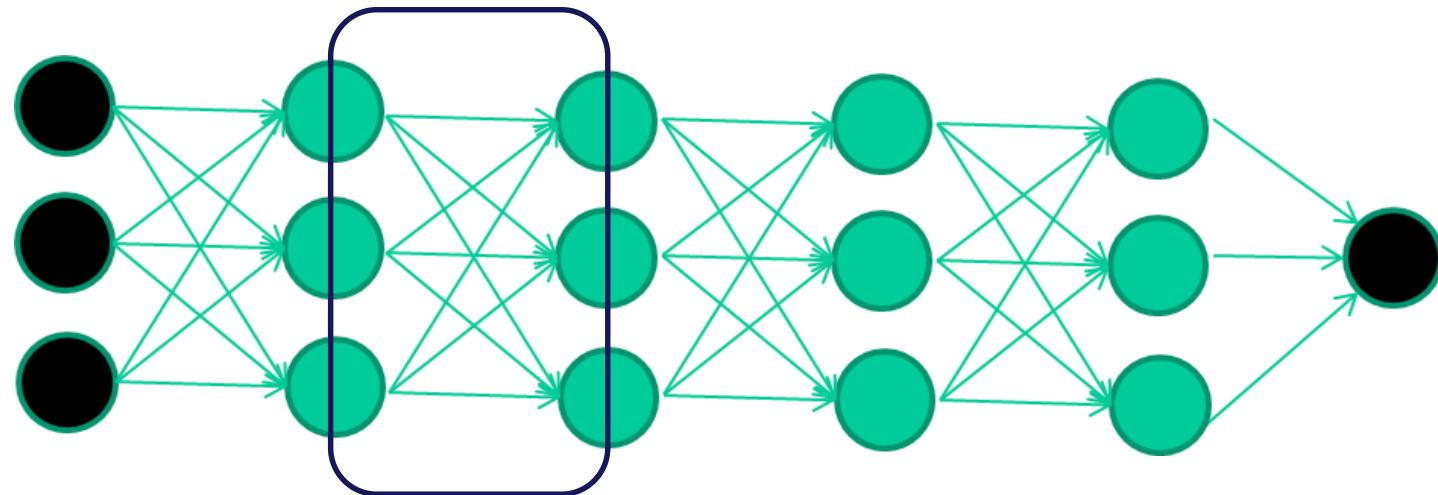


The new way to train multi-layer NNs...



Train **this** layer first

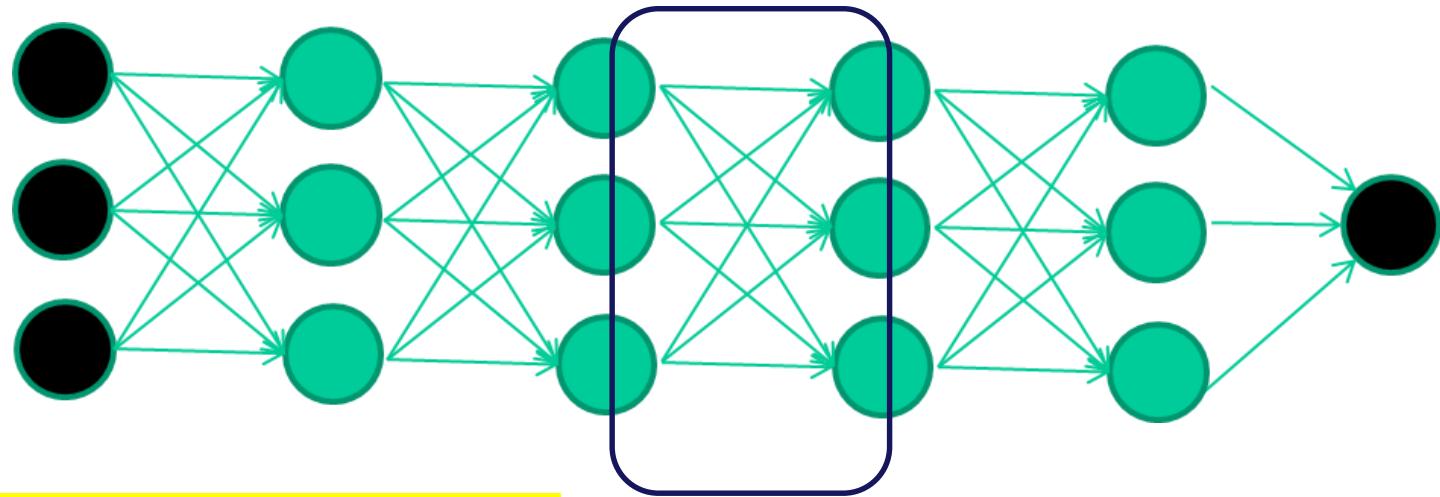
The new way to train multi-layer NNs...



Train **this** layer first

then **this** layer

The new way to train multi-layer NNs...

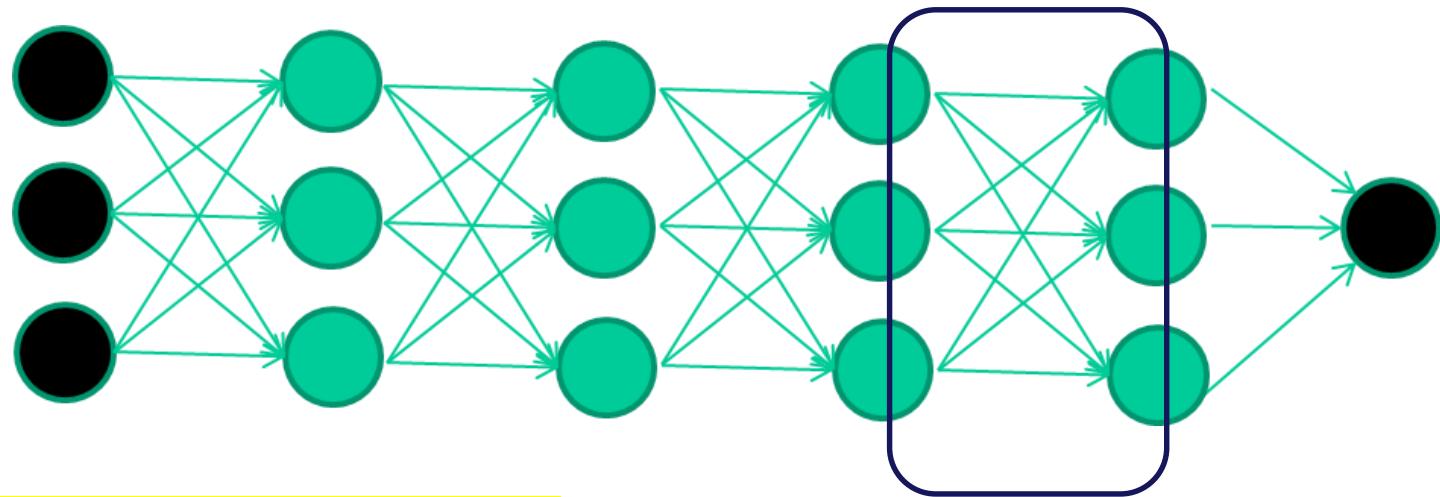


Train **this** layer first

then **this** layer

then **this** layer

The new way to train multi-layer NNs...



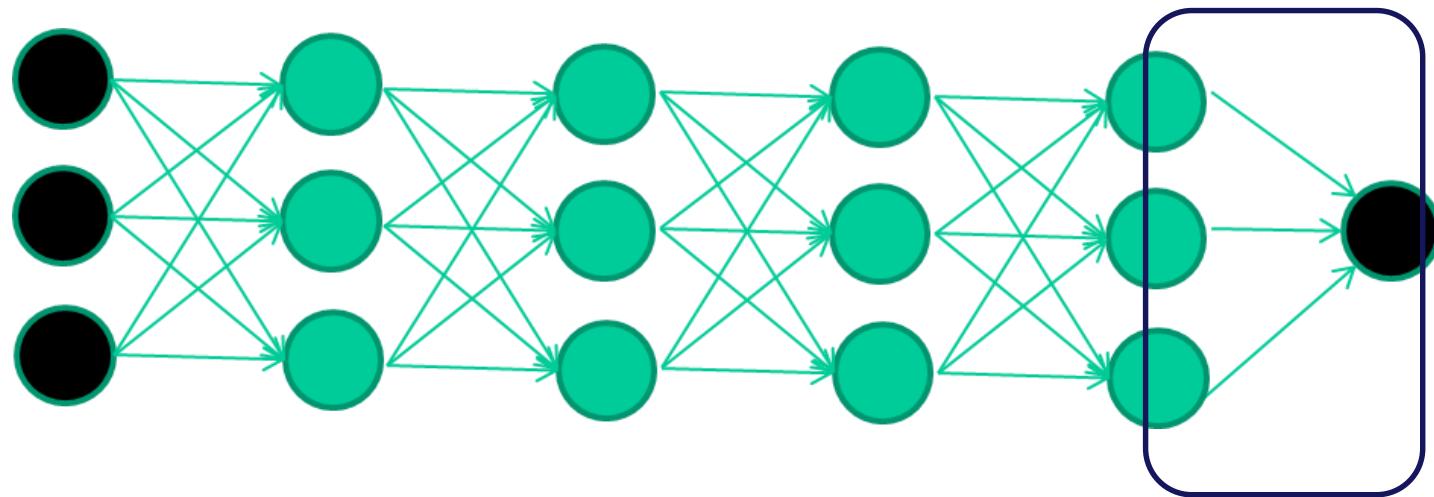
Train **this** layer first

then **this** layer

then **this** layer

then **this** layer

The new way to train multi-layer NNs...



Train **this** layer first

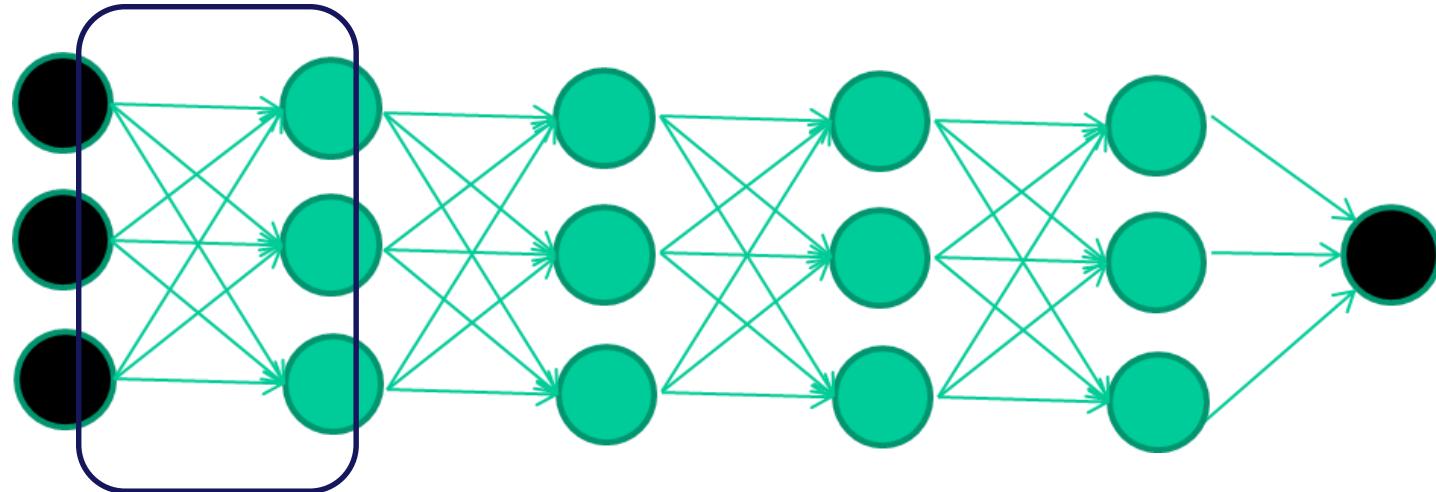
then **this** layer

then **this** layer

then **this** layer

finally **this** layer

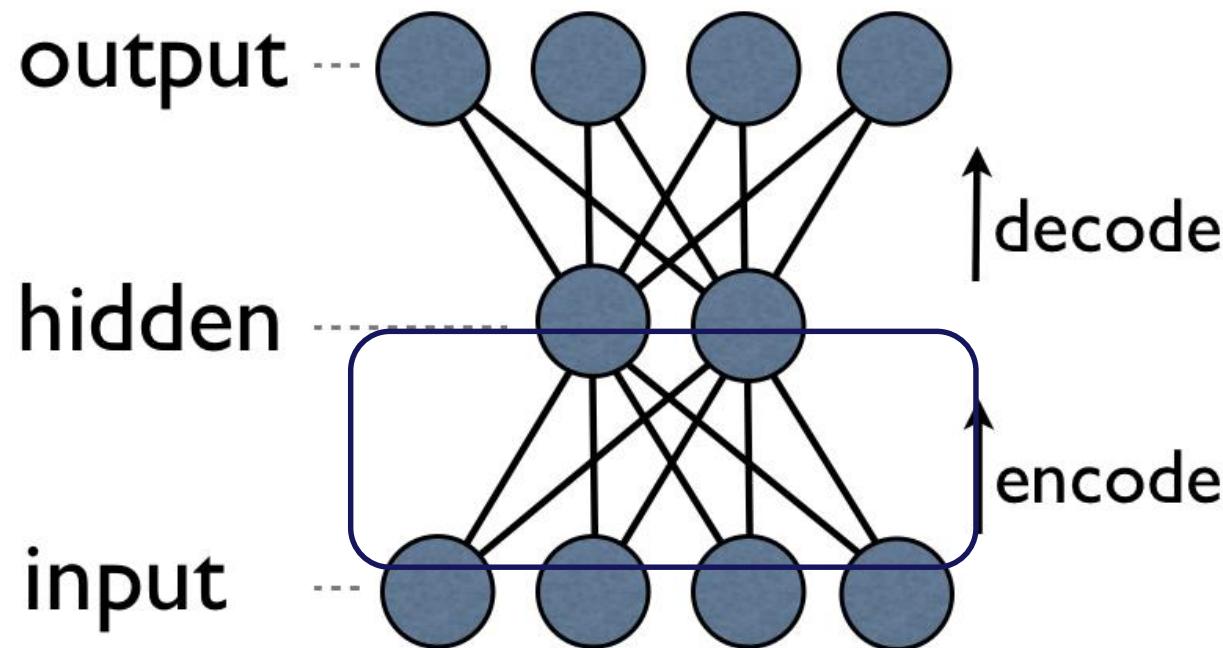
The new way to train multi-layer NNs...



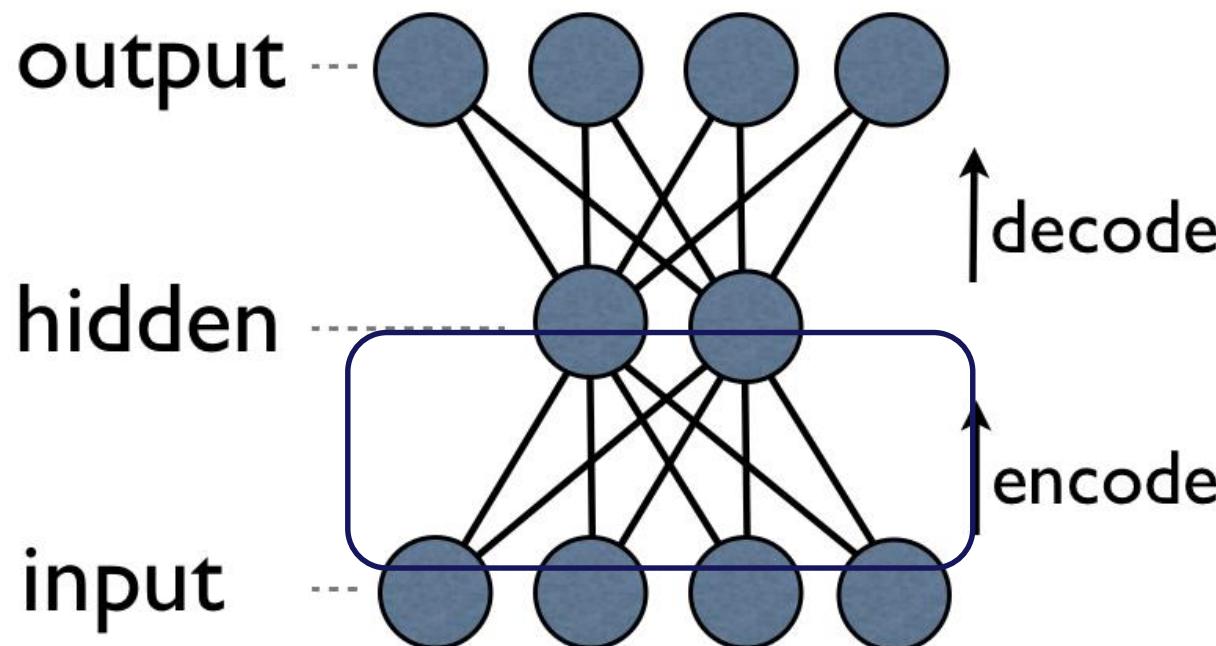
SETIAP lapisan (non-output) dilatih
untuk menjadi auto-encoder

Pada dasarnya, ia dipaksa untuk mempelajari fitur-fitur bagus yang menggambarkan apa yang berasal dari lapisan sebelumnya

sebuah auto-encoder dilatih, dengan algoritma penyesuaian berat yang benar-benar standar untuk mereproduksi input

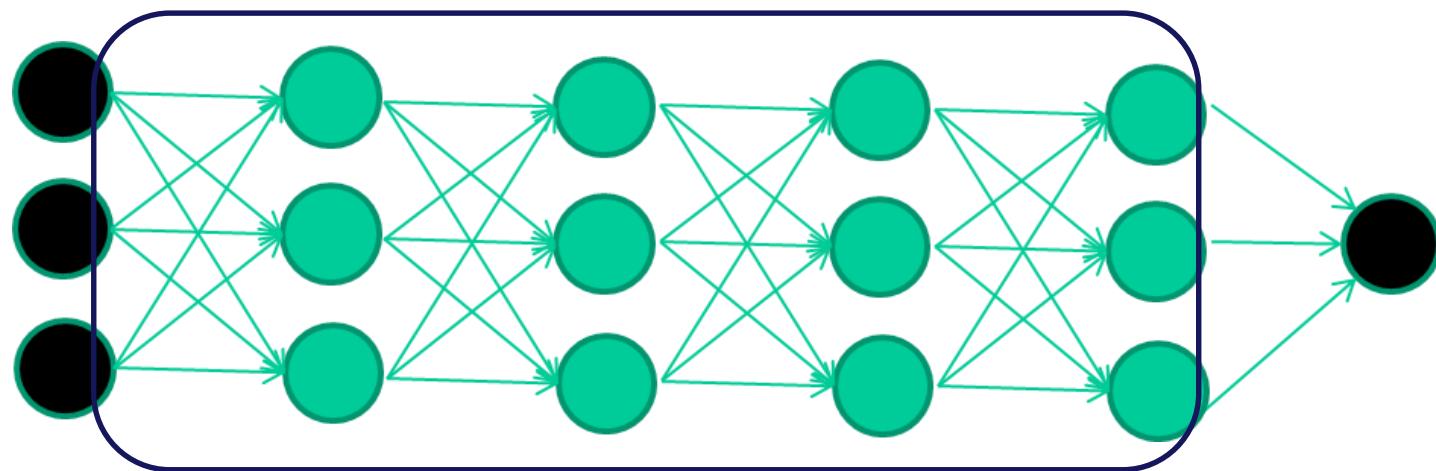


sebuah auto-encoder dilatih, dengan algoritma penyesuaian berat yang benar-benar standar untuk merekproduksi input

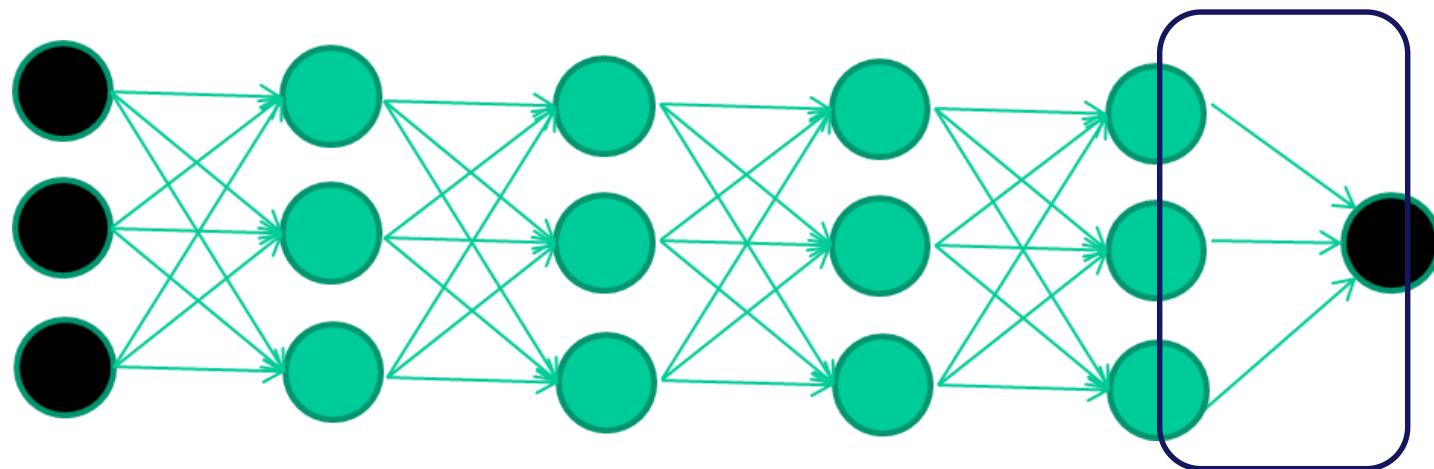


Dengan mewujudkan hal ini dengan (banyak) unit lebih sedikit daripada input, ini memaksa unit 'lapisan tersembunyi' untuk menjadi pendekripsi fitur yang baik

lapisan menengah masing-masing dilatih untuk menjadi penyandi otomatis (atau serupa)



Lapisan akhir dilatih untuk memprediksi kelas berdasarkan keluaran dari lapisan sebelumnya



Penutup

- Ada banyak banyak jenis pembelajaran mendalam (*Deep Learning*), berbagai jenis autoencoder, variasi arsitektur dan algoritma pelatihan, dll ...
- Area penelitian yang tumbuh sangat cepat

