



UNIVERSITAS  
INDONESIA

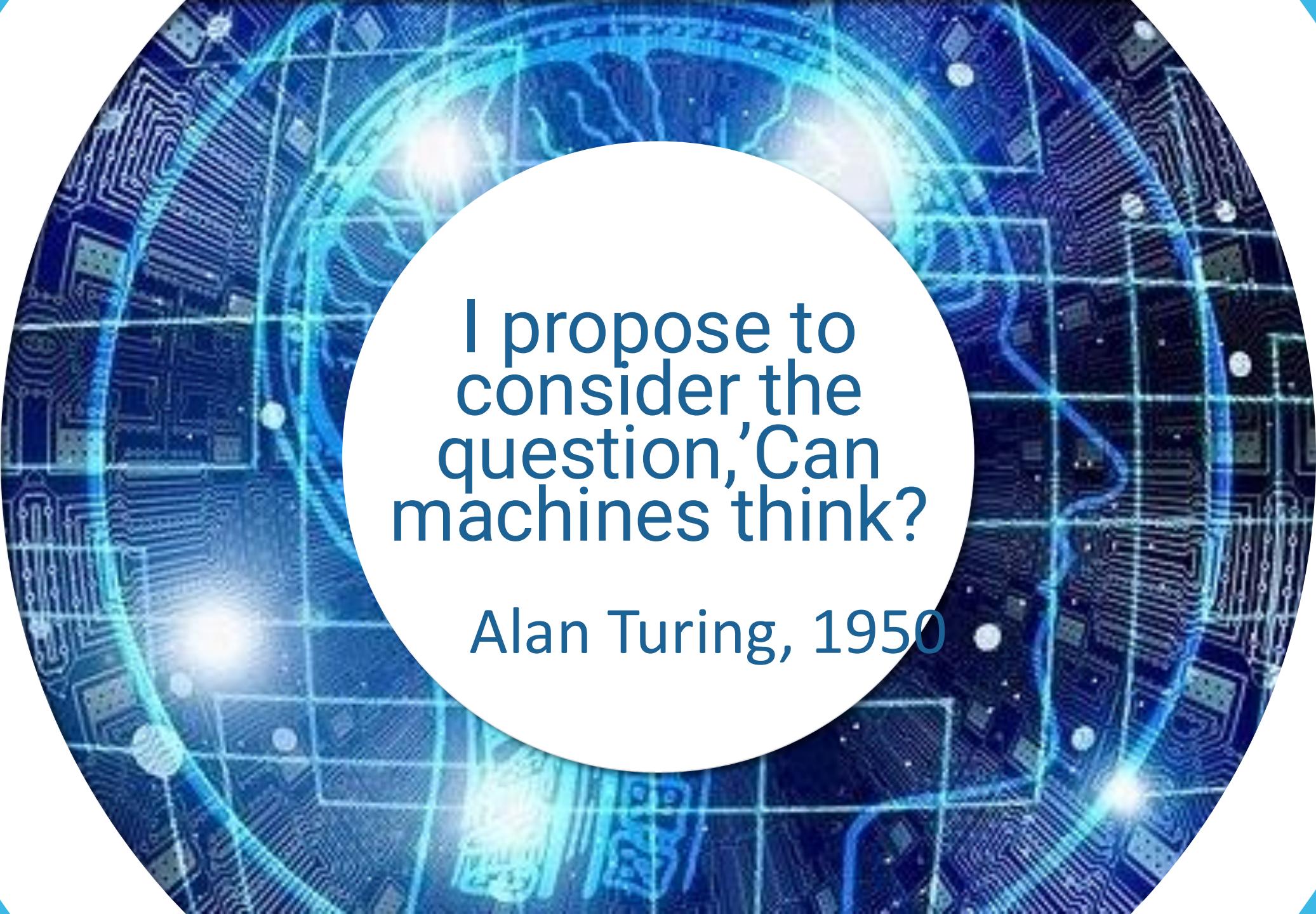
*Werdha Prakarsa Suciwa*

# KECERDASAN BUATAN

## 1 | PENDAHULUAN

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI





I propose to consider the question, 'Can machines think?

Alan Turing, 1950



UNIVERSITAS  
INDONESIA

Virtus, Prudentia, Justitia

---

Apakah yang dimaksud  
dengan Artificial  
Intelligence (AI)?





UNIVERSITAS  
INDONESIA

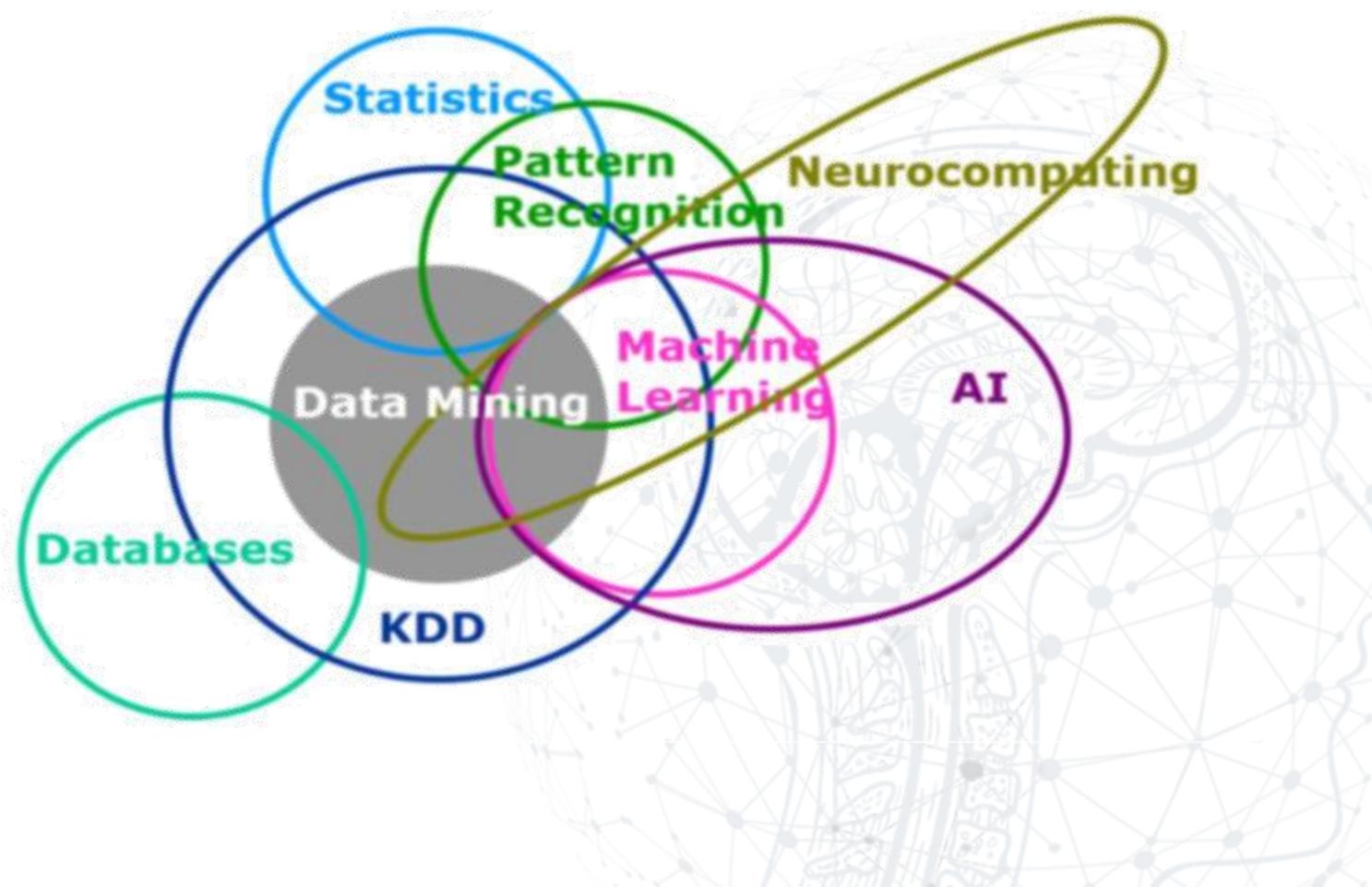
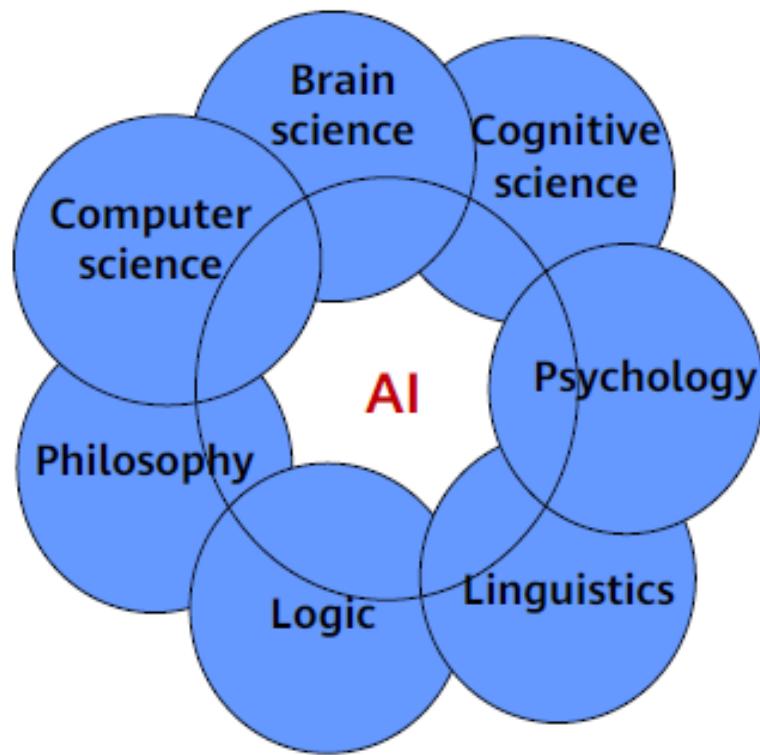
*Virtus, Prudentia, Justitia*



**Science and  
engineering of  
making intelligent  
machines, especially  
intelligent computer  
program**

*John McCarthy, father of AI, 1956*

# Bidang yang berkaitan

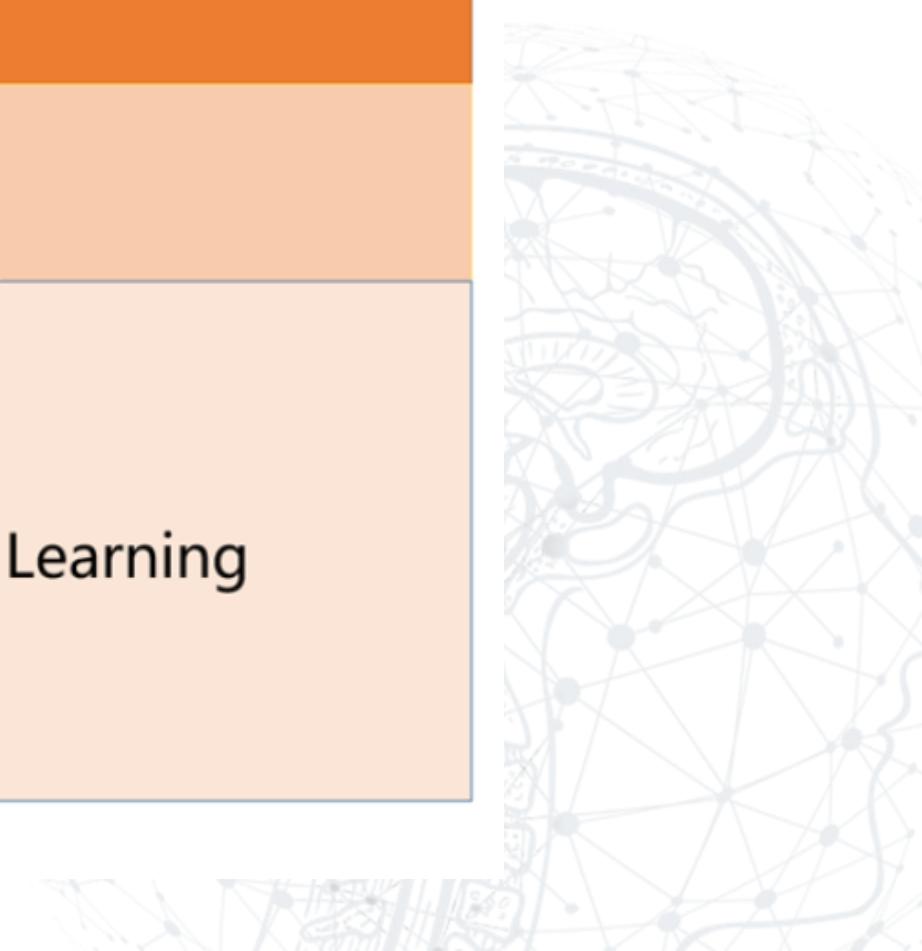
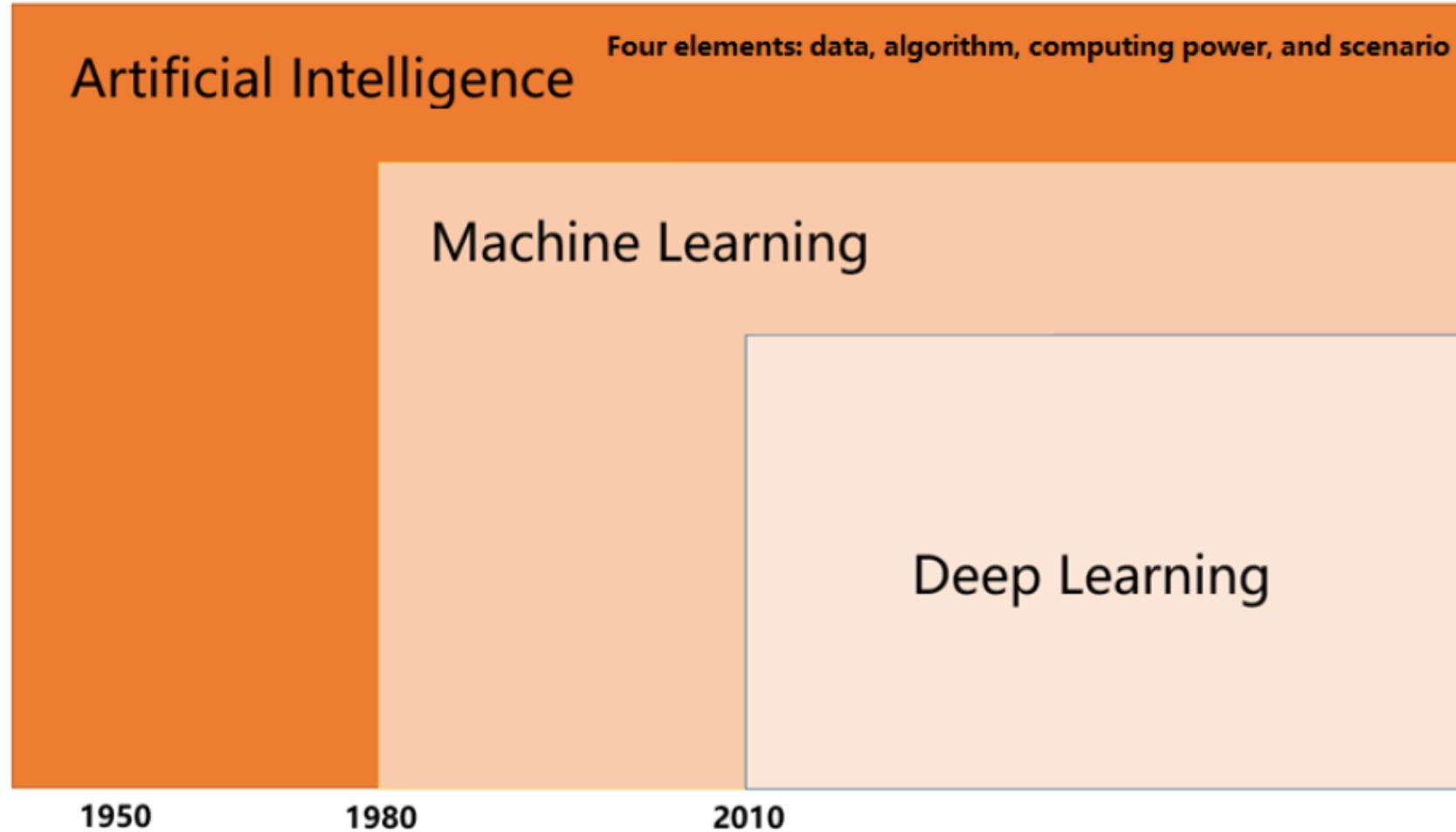


---

Apakah  
**AI = machine learning**



# Machine learning adalah salah satu bagian dari AI yang sedang sangat dikembangkan



# AI – MACHINE LEARNING – DEEP LEARNING



**AI:** Ilmu yang berfokus pada penelitian dan pengembangan teori, metode, teknik, dan sistem aplikasi untuk merangsang dan memperluas kecerdasan manusia.

**Machine learning:** Bidang penelitian inti Al. Berfokus pada studi tentang bagaimana komputer dapat memperoleh pengetahuan atau keterampilan baru dengan menirukan atau melakukan pembelajaran perilaku manusia, dan mengatur ulang arsitektur pengetahuan yang ada untuk meningkatkan kinerjanya.

**Deep learning:** Bidang baru pembelajaran mesin. Konsep ini berasal dari penelitian lain pada *artificial neural networks* (ANN). Tujuannya adalah untuk mensimulasikan otak manusia untuk menafsirkan data seperti gambar, suara, dan teks.

# Data Mining vs Machine Learning



data mining  
focuses on  
**analyzing input  
variables to predict  
a new output**

machine learning  
extends to  
**analyzing both  
input and output  
variables**

A large, stylized graphic of a human brain is positioned on the left side of the slide. The brain is composed of numerous glowing blue lines representing circuit boards or neural pathways, set against a dark blue background. To the left of the brain, there is a grid of binary code (0s and 1s) in a lighter shade of blue.

# Komponen utama AI

1

Data

2

Algoritma

3

Computing power

4

Scenario

# CONTOH IMPLEMENTASI AI



## Public sector

- Safe City
- Intelligent transport
- Disaster prediction



## Education

- Personalization
- Attention improvement
- Robot teacher



## Healthcare

- Early prevention
- Diagnosis assistance
- Precision cure



## Media

- Real-time translation
- Abstraction
- Inspection



## Pharmacy

- Fast R&D
- Precise trial
- Targeted medicine



## Logistics

- Routing planning
- Monitoring
- Auto sorting



## Finance

- Doc process
- Real-time fraud prevention
- Up-sell



## Insurance

- Auto detection
- Fraud prevention
- Innovative service



## Retail

- Staff-less shops
- Real-time inventory
- Precise recommendations



## Manufacturing

- Defect detection
- Industrial internet
- Predictive maintenance



## Telecom

- Customer service
- Auto O&M
- Auto optimization



## Oil and gas

- Localization
- Remote maintenance
- Operation optimization



## Agriculture

- Fertilization improvement
- Remote operation
- Seeds development



# AI Disputes

# ALGORITHMIC BIAS

- Algorithmic biases are mainly caused by data biases.

If we search with a name sounds like an African American, an advertisement for **a tool used to search criminal records** may be displayed. The advertisement, however, is not likely displayed in other cases.

Google's image software once mistakenly labeled an image of black people as "**gorilla**".

Online advertisers tend to display advertisements of lower-priced goods to **female users**.

# PRIVACY ISSUES

In principle, technology companies can record each click, each page scrolling, time of viewing any content, and browsing history when users access the Internet.

Technology companies can know our privacy including where we are, where we go, what we have done, education background, consumption capabilities, and personal preferences based on our ride-hailing records and consumption records.

# SEEING = BELIEVING?

- <https://www.youtube.com/watch?v=T76bK2t2r8g>
  - Deepfakes
- 
- <https://arstechnica.com/information-technology/2024/02/deepfake-scammer-walks-off-with-25-million-in-first-of-its-kind-ai-heist/>

# UNEMPLOYMENT?

- Jenis pekerjaan apa yang kira-kira bakal tergantikan oleh AI?
- <https://www.weforum.org/publications/the-future-of-jobs-report-2023/infographics-2128e451e0/>



# DEVELOPMENT TRENDS OF AI TECHNOLOGIES



- **Framework:** easier-to-use development framework
- **Algorithm:** algorithm models with better performance and smaller size
- **Computing power:** comprehensive development of device-edge-cloud computing
- **Data:** more comprehensive basic data service industry and more secure data sharing
- **Scenario:** continuous breakthroughs in industry applications



Apa yang akan  
dipelajari di kuliah  
ini?

# DESKRIPSI MATA KULIAH

Mata kuliah ini adalah mata kuliah wajib di Kurikulum 2020 pada Program Studi Teknik Komputer. Namun demikian, mata kuliah ini juga dapat diambil oleh mahasiswa dari prodi lain yang memiliki minat untuk belajar mengenai machine learning dan artificial intelligence. Pada mata kuliah ini mahasiswa akan **mempelajari dasar-dasar metode pembelajaran mesin untuk mengolah data sehingga dapat menghasilkan informasi secara otomatis.**

# CAPAIAN PEMBELAJARAN MATA KULIAH (CPMK)



1. Mampu merancang penyelesaian permasalahan pengolahan data dengan pendekatan pembelajaran mesin (C6)
2. Mampu bekerjasama dalam tim untuk menyelesaikan proyek machine learning (A3)

# SUB-CPMK

- 1.1 Mampu membedakan jenis-jenis pembelajaran mesin (C2)
- 1.2 Mampu melakukan tahapan persiapan data untuk digunakan dalam machine learning (C3)
- 1.3 Mampu mengimplementasikan metode regresi (C3)
- 1.4 Mampu mengimplementasikan metode pengelompokan (clustering) (C3)
- 1.5 Mampu mengevaluasi teknik klasifikasi non-ANN untuk penyelesaian permasalahan machine learning (C5)
- 1.6 Mampu mengevaluasi teknik klasifikasi ANN untuk penyelesaian permasalahan machine learning (C5)

- 1.7 Mampu mengevaluasi konsep dasar deep learning (C5)
- 1.8 Mampu mengevaluasi best practice dalam mengembangkan aplikasi deep learning (C5)
- 1.9 Mampu merancang penggunaan convolutional neural network dalam berbagai aplikasi sederhana (C6)
- 1.10 Mampu merancang penggunaan sequence models dalam berbagai aplikasi sederhana (C6)
  
- 2.1 Mampu menunjukkan tanggung jawab dalam kerja sama tim untuk menyelesaikan proyek (A3)
- 2.2 Mampu menunjukkan proses berpikir kritis dan kreatif dalam menyelesaikan permasalahan kelompok (A3)

# REFERENSI



1. Aurélien Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, O'Reilly, 2023
2. Ian Goodfellow, Deep Learning, MIT Press, 2016
3. Oliver Theobald, Machine Learning for Absolute Beginners: A Plain English Introduction, Independently published, 2018
4. Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer, 2011
5. Joel Gruss, Data Science from Scratch, O'Reilly, 2015

## Bahan bacaan lain:

Andrew Ng, Deep Learning Specialization, Online Course at Coursera

# Skills: Cognitiveclass.ai



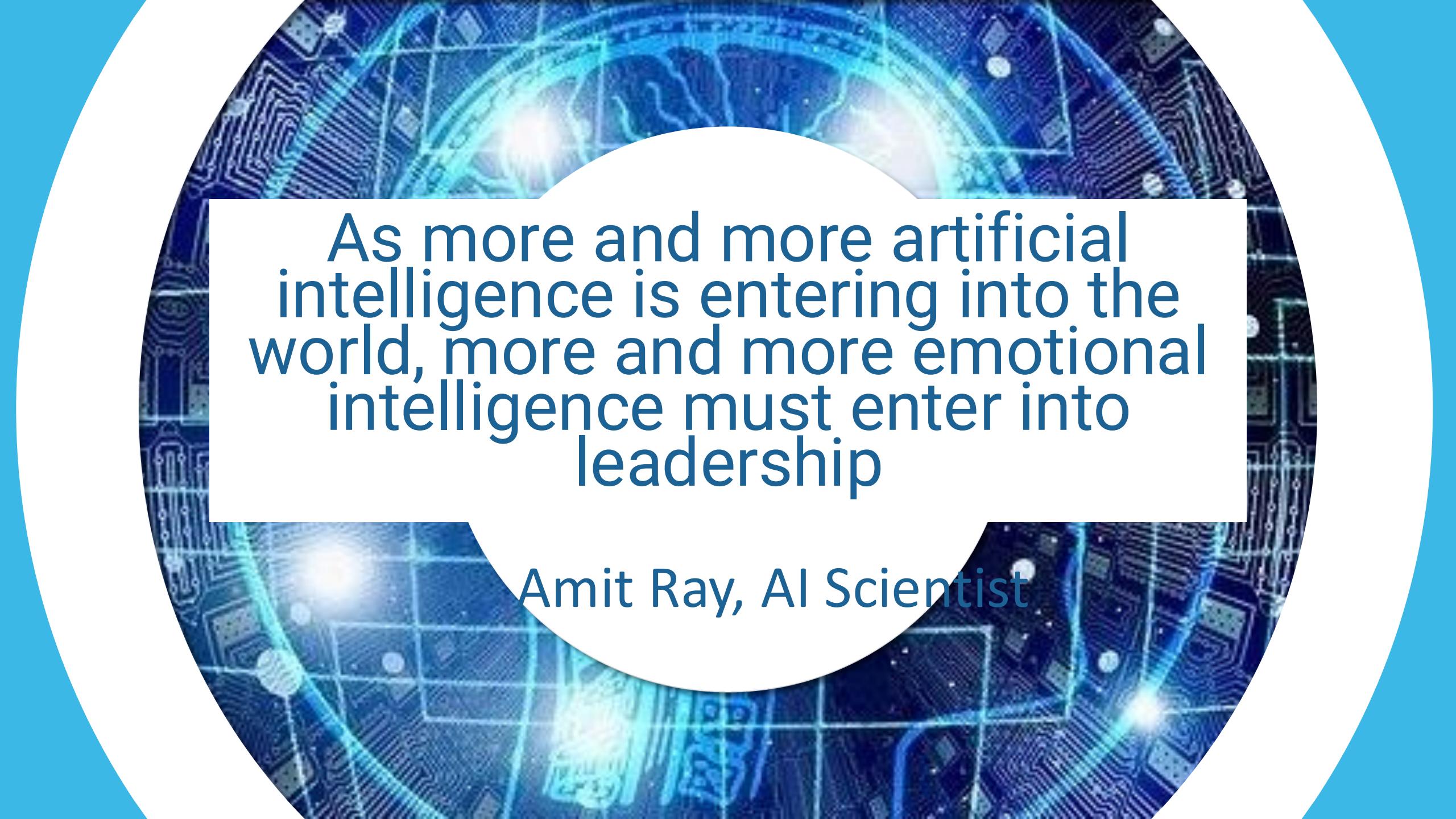
- Basic: If you are new to Python
  - Python for Data Science
- Compulsory:
  - **Machine Learning with Python** → as Quiz 1 Score

Enroll kelas tersebut dari sekarang.

Pelajari secara mandiri (self-paced) materi dan Latihan (hands-on) yang ada.

Kerjakan semua Review Question dan Final Exam.

Kumpulkan sertifikat dan nilai sebagai nilai Quiz.



As more and more artificial intelligence is entering into the world, more and more emotional intelligence must enter into leadership

Amit Ray, AI Scientist



# **EMAS2**

<https://emas2.ui.ac.id/course/view.php?id=87766>

key: **cerdas**

# PENDAHULUAN

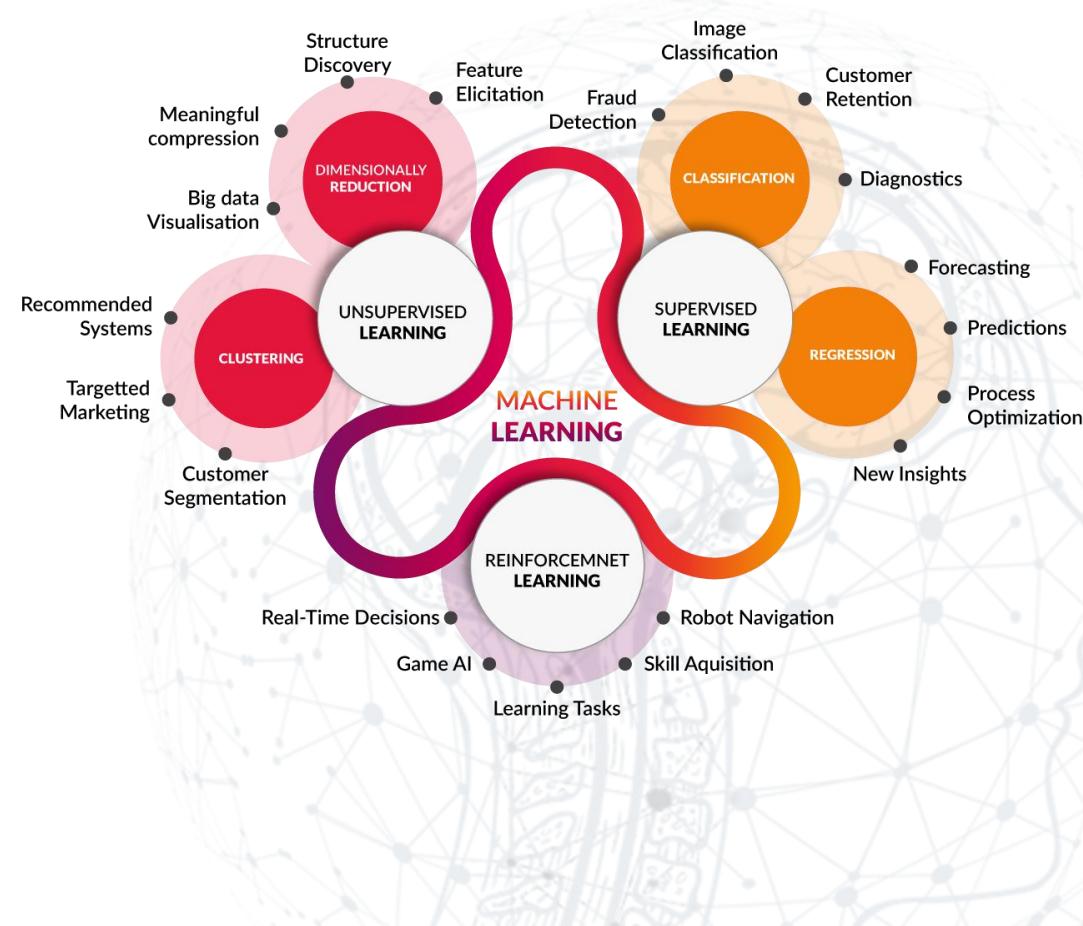
**Sub CPMK 1.1** Mampu membedakan jenis-jenis pembelajaran mesin (C2)

**Materi** Pengenalan machine learning dan artificial intelligence

**Referensi** Theobald, ch. 1

**Indikator:**

- Mampu membedakan kasus regresi
- Mampu membedakan kasus clustering
- Mampu membedakan kasus klasifikasi



# PERSIAPAN DATA

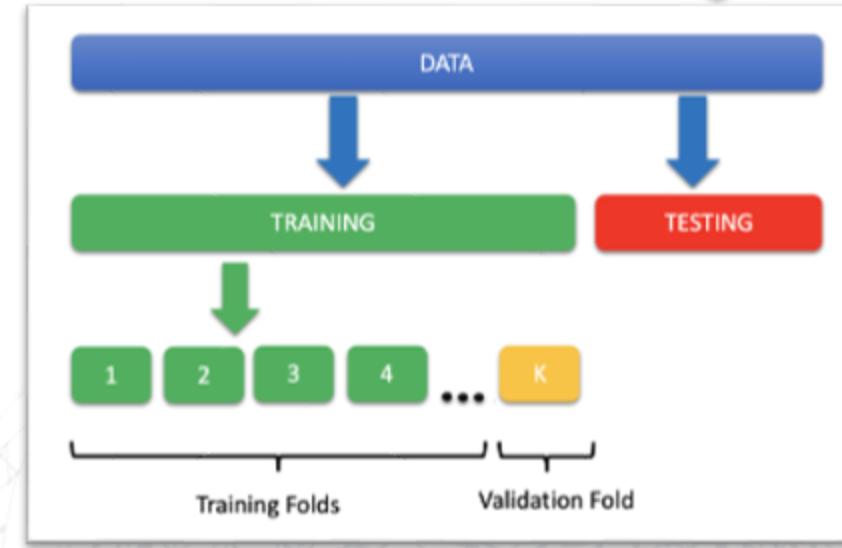
**Sub CPMK 1.2** Mampu menjelaskan tahapan dalam machine learning (C3)

**Materi** Persiapan data, modelling, correctness, bias-variance, feature extraction & selection, training-testing

**Referensi** Theobald, ch. 2, 3, 4 & Gruss, ch 11

**Indikator:**

- Mampu menjelaskan tahap persiapan data
- Mampu menjelaskan tahap memodelkan machine learning
- Mampu menjelaskan bias & variance
- Mampu menjelaskan correctness
- Mampu menjelaskan feature extraction & selection
- Mampu menjelaskan strategi training, testing dan validation



	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

# REGRESI

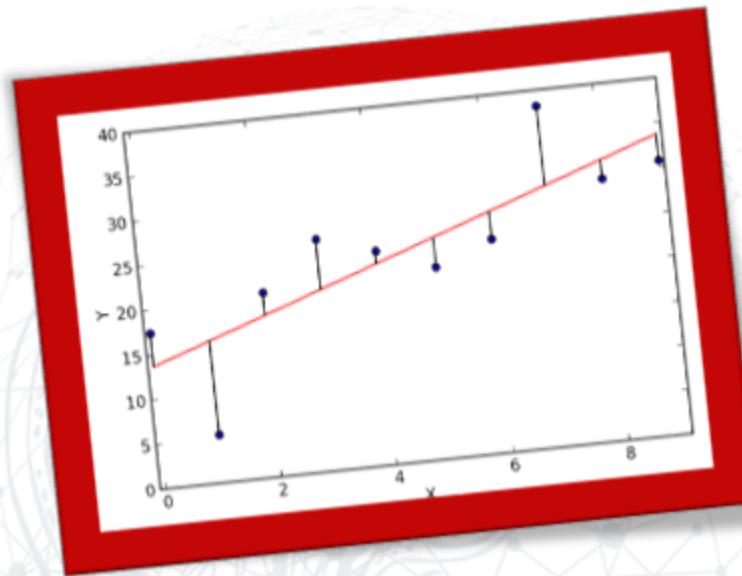
**Sub CPMK 1.3** Mampu mengimplementasikan metode regresi (C3)

**Materi** Simple linear regression, Gradient descent, Multiple regression

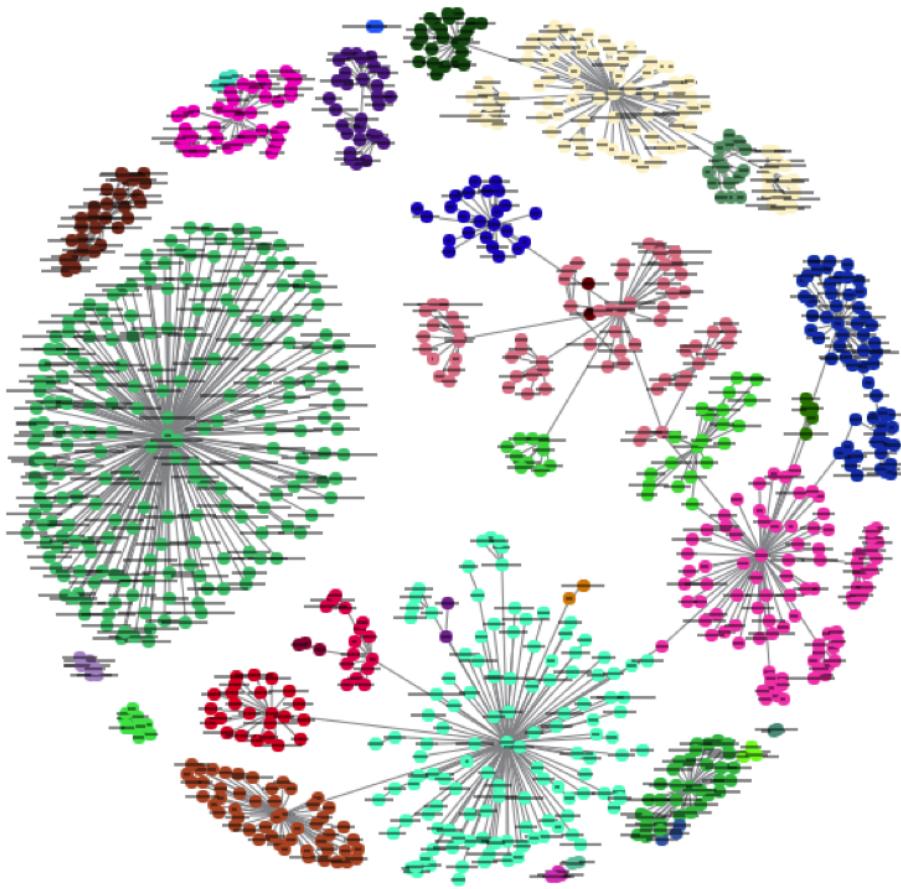
**Referensi** Gruss 14,15

**Indikator:**

- Mampu mengimplementasikan metode simple linear regression dengan metode statistik
- Mampu mengimplementasikan metode simple linear regression dengan metode gradient descent
- Mampu menjelaskan multiple regression



# CLUSTERING



**Sub CPMK 1.4** Mampu mengimplementasikan metode pengelompokan (clustering) (C3)

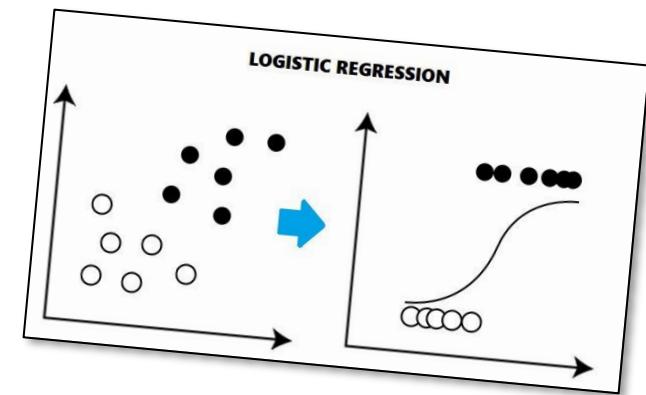
**Materi** Clustering, K-Means, Self Organizing Map (SOM)

**Referensi** Gruss chapter 19, Theobald chapter 7, Kohonen

**Indikator:**

- Mampu mengimplementasikan metode K-Means
- Mampu menjelaskan metode SOM

# CLASSIFIER NON-ANN



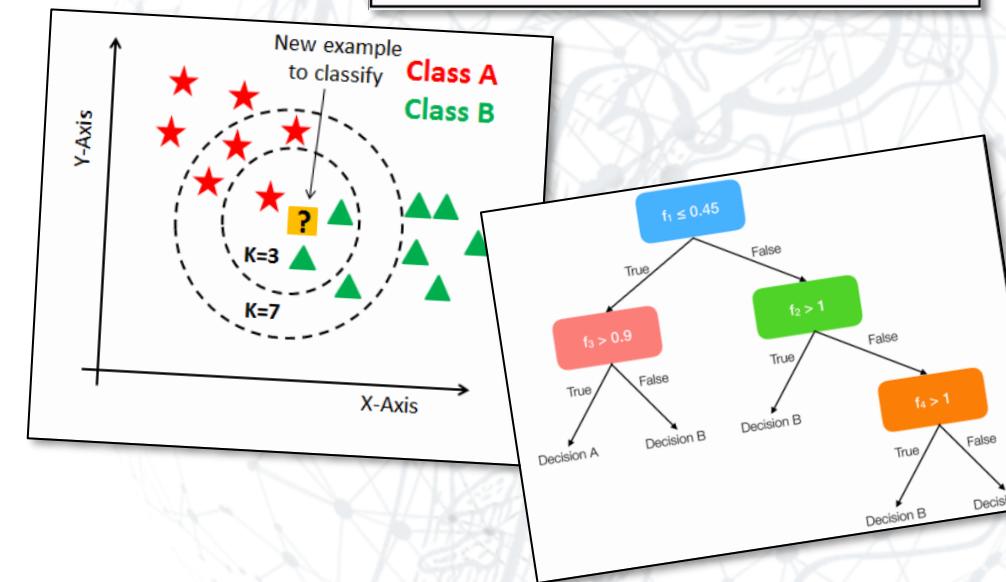
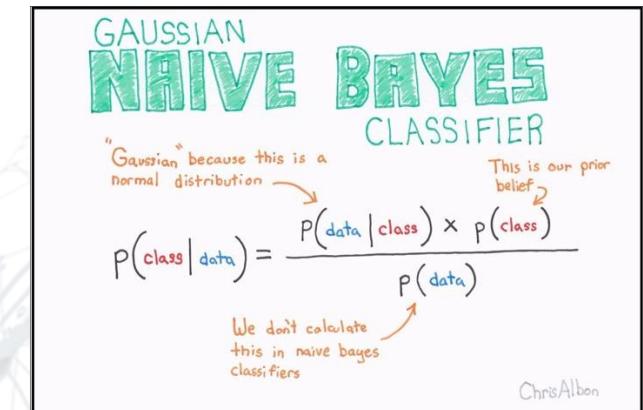
**Sub CPMK 1.5** Mampu mengevaluasi teknik klasifikasi untuk penyelesaian permasalahan machine learning (C5)

**Materi** KNN, Naïve Bayes, Logistic Regression, Decision Tree

**Referensi** Gruss, ch 12, 13, 16, 17

## Indikator:

- Mampu mengevaluasi KNN classifier
- Mampu mengevaluasi Naïve Bayes classifier
- Mampu mengevaluasi Logistic Regression classifier
- Mampu mengevaluasi Decision Tree classifier



# CLASSIFIER ANN

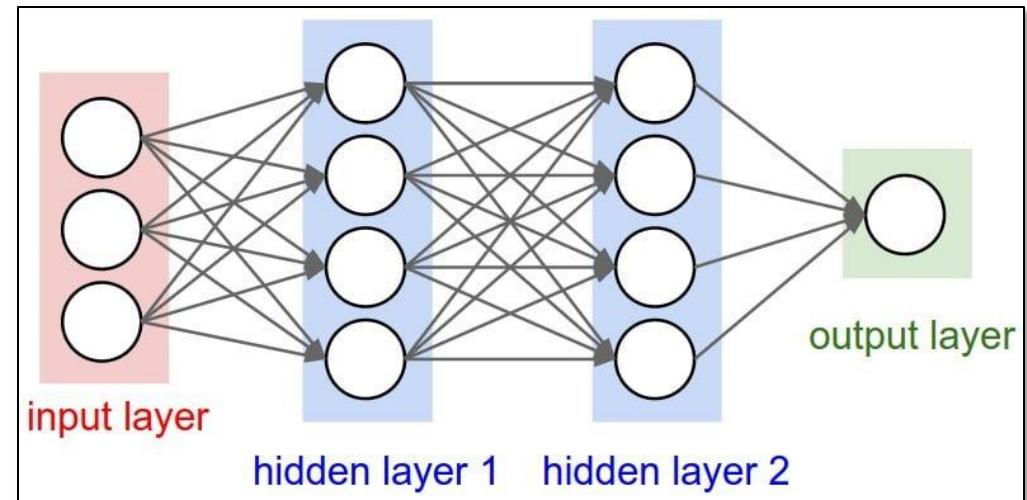
**Sub CPMK 1.6** Mampu mengevaluasi teknik klasifikasi ANN untuk penyelesaian permasalahan machine learning (C5)

**Materi** Artificial Neural Networks, Back Propagation Neural Networks

**Referensi** Gruss chapter 19, Theobald chapter 9, Bishop chapter 5

**Indikator:**

- Mampu menjabarkan konsep BPNN untuk klasifikasi
- Mampu mengimplementasikan BPNN ke dalam program



# PRINCIPAL COMPONENT ANALYSIS

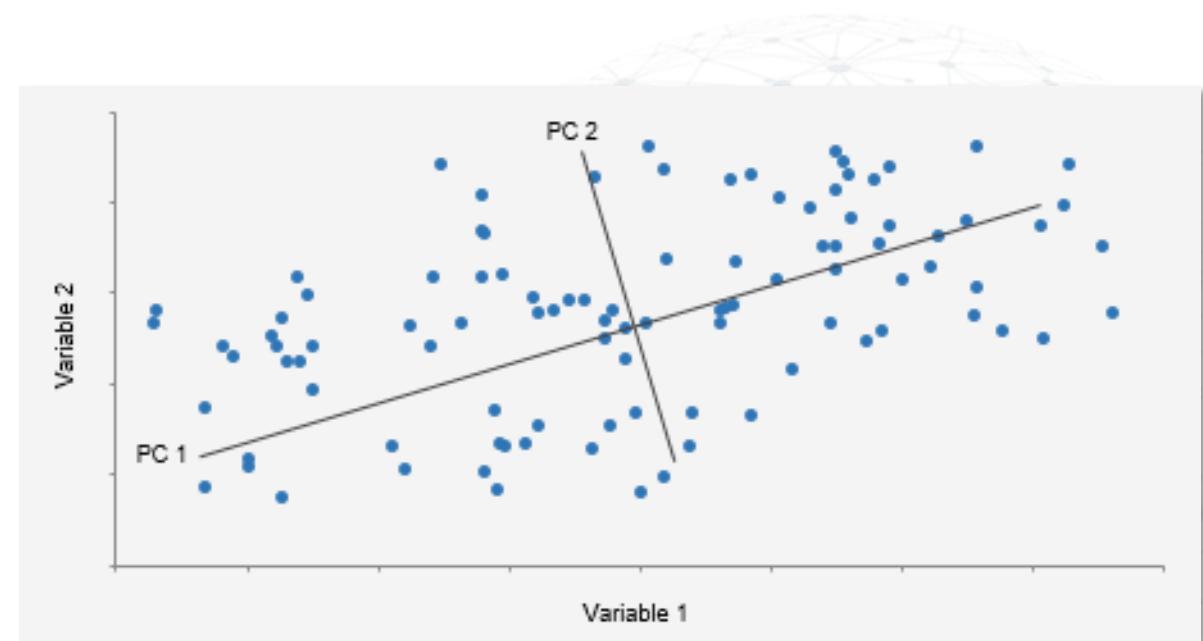
**Sub CPMK 1.7** Mampu merancang sistem pengenalan lengkap berbasis ANN (C6)

**Materi** Principal Component Analysis (PCA)

**Referensi** Lecture notes

**Indikator:**

Mampu mengimplementasikan reduksi dimensi fitur dengan PCA



# DEEP LEARNING

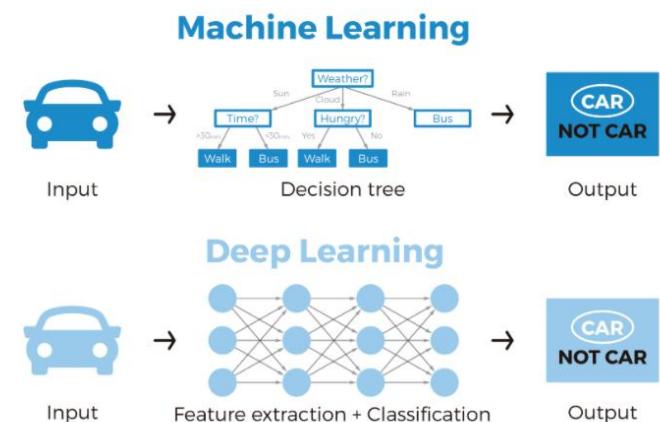
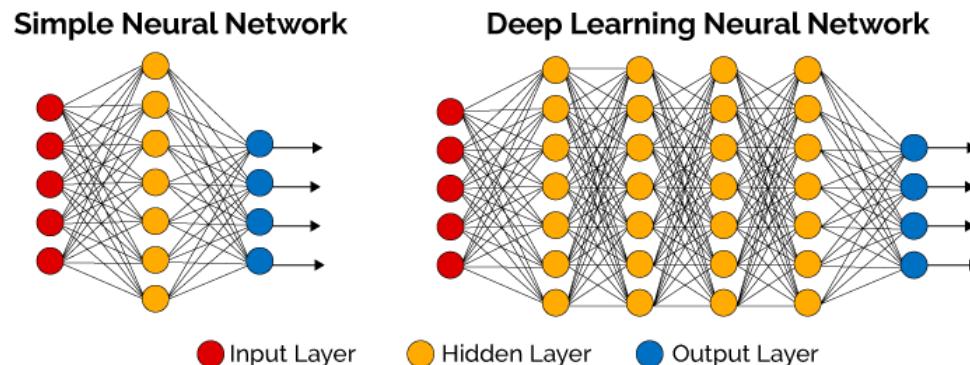
**Sub CPMK 1.7** Mampu merancang sistem pengenalan lengkap berbasis ANN (C6)

**Materi** Deep Learning: CNN, RNN, LSTM

**Referensi** Bahan bacaan dari Internet (Tensorflow, Keras)

**Indikator:**

Mampu mengimplementasikan konsep Deep learning untuk klasifikasi



TensorFlow



Keras



# PROYEK 1

## Sub CPMK

- 1.5 Mampu mengevaluasi teknik klasifikasi non-ANN untuk penyelesaian permasalahan machine learning (C5)
- 2.1 Mampu menunjukkan proses berpikir kritis dan kreatif dalam menyelesaikan permasalahan kelompok (A3)
- 2.2 Mampu menunjukkan tanggung jawab dalam kerja sama tim untuk menyelesaikan proyek (A3)

**Materi** KNN, Naïve Bayes, Logistic Regression, Decision Tree

**Referensi** Gruss, ch 12, 13, 16, 17

## Indikator:

- Mampu mengevaluasi Non-ANN classifier
- Mampu menunjukkan proses berpikir kritis dan kreatif dalam menyelesaikan permasalahan kelompok
- Mampu menunjukkan tanggung jawab dalam kerja sama tim untuk menyelesaikan proyek



# PROYEK 2

## Sub CPMK

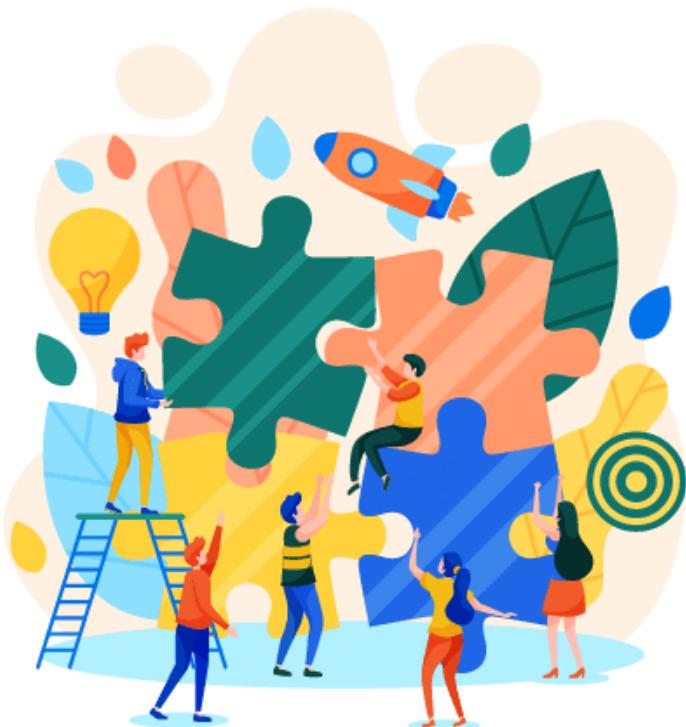
- 1.7 Mampu merancang sistem pengenalan lengkap berbasis ANN (C6)
- 2.1 Mampu menunjukkan proses berpikir kritis dan kreatif dalam menyelesaikan permasalahan kelompok (A3)
- 2.2 Mampu menunjukkan tanggung jawab dalam kerja sama tim untuk menyelesaikan proyek (A3)

**Materi** Pemrosesan Data, ANN, PCA, DNN, CNN, RNN

**Referensi** Lecture Notes

### Indikator:

- Mampu mengevaluasi ANN classifier
- Mampu menunjukkan proses berpikir kritis dan kreatif dalam menyelesaikan permasalahan kelompok
- Mampu menunjukkan tanggung jawab dalam kerja sama tim untuk menyelesaikan proyek



# MACHINE LEARNING

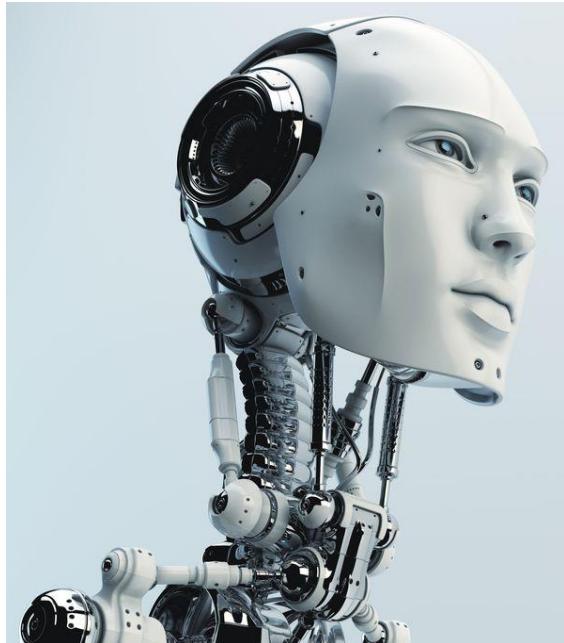
# WHAT IS MACHINE LEARNING?

Learn from experience



Human

data  
Learn from experience



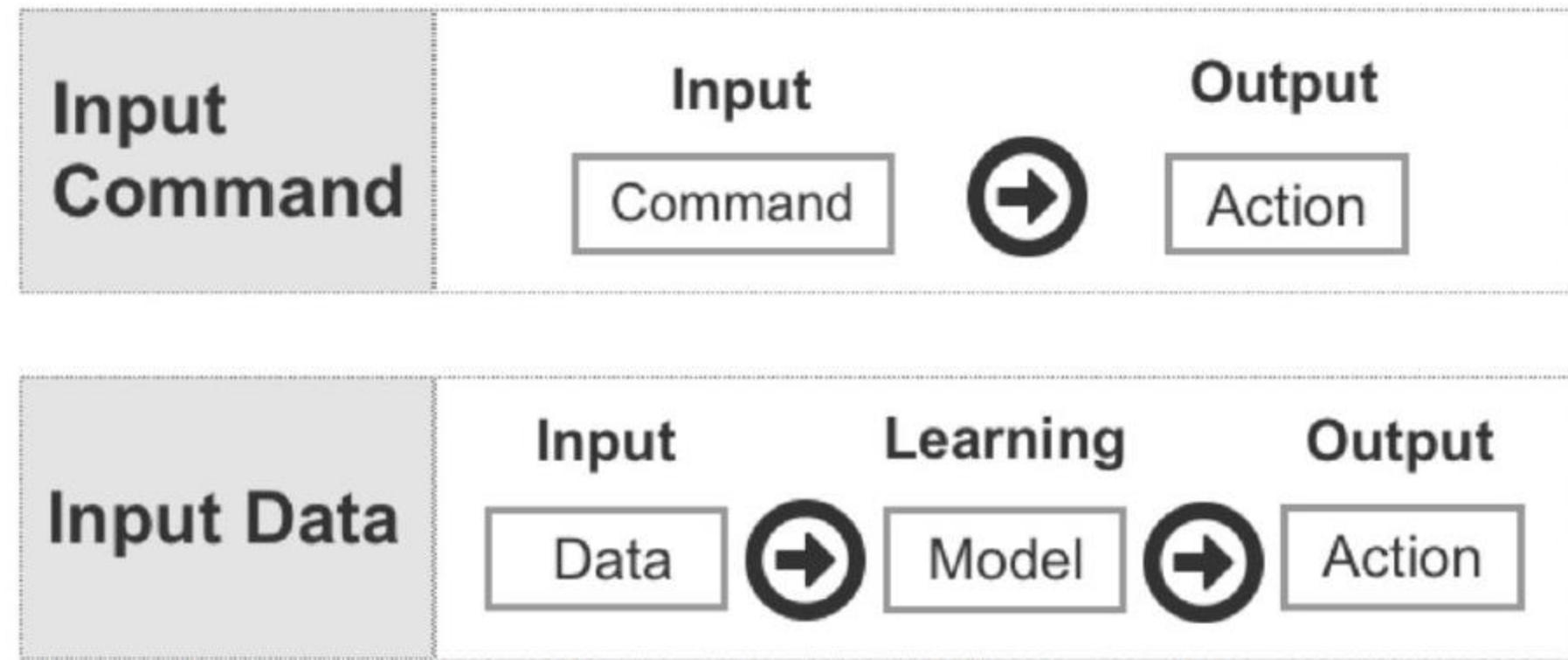
Machine Learning

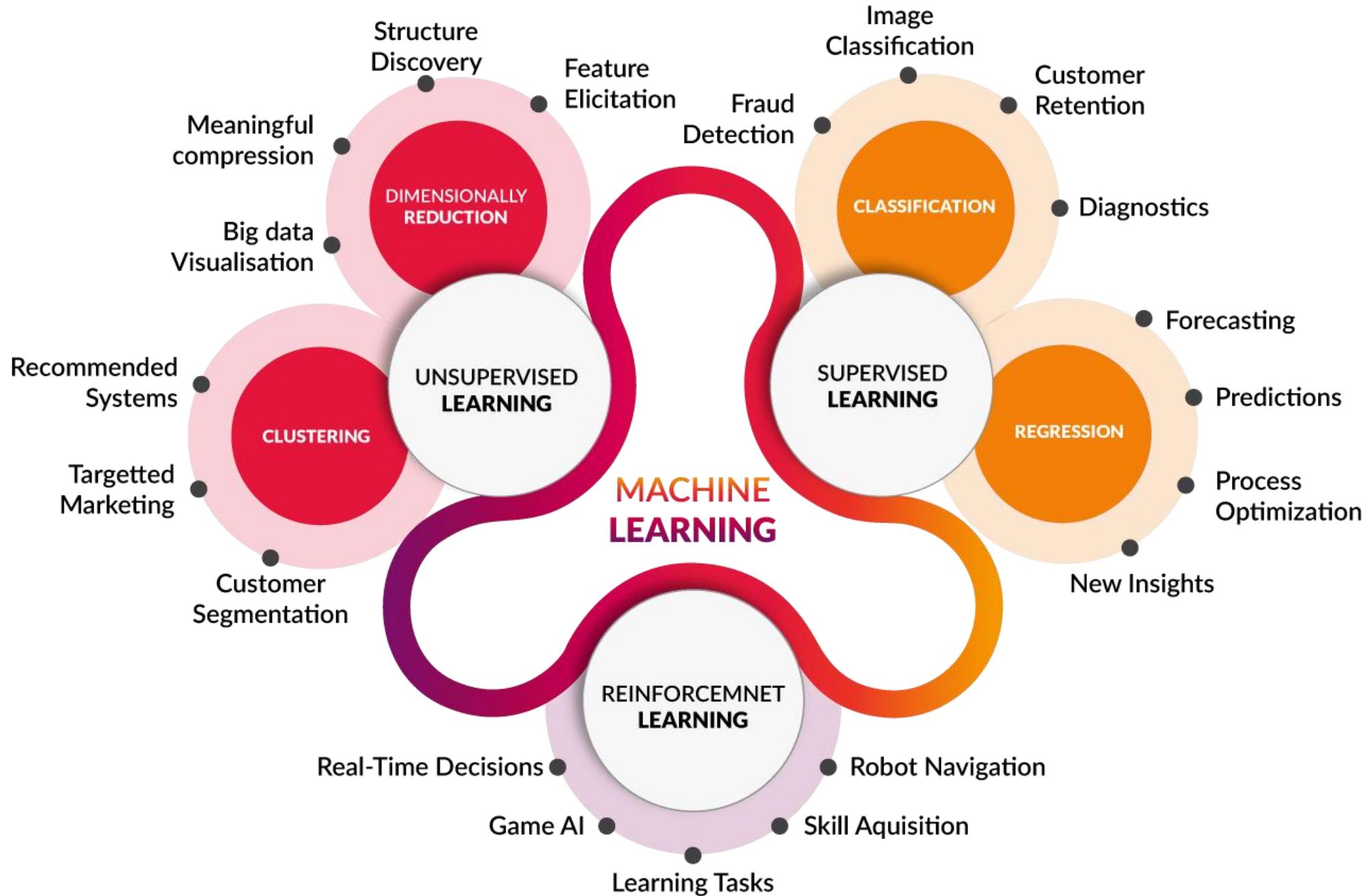
Follow instructions



Machine

# INPUT COMMAND VS INPUT DATA







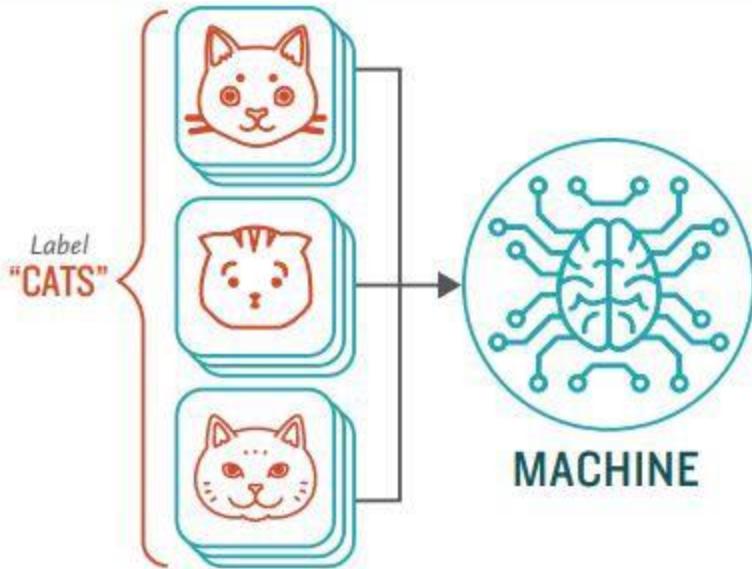
# SUPERVISED LEARNING

learning patterns from labeled datasets and decoding the relationship between input variables (independent variables) and their known output (dependent variable)

# How **Supervised** Machine Learning Works

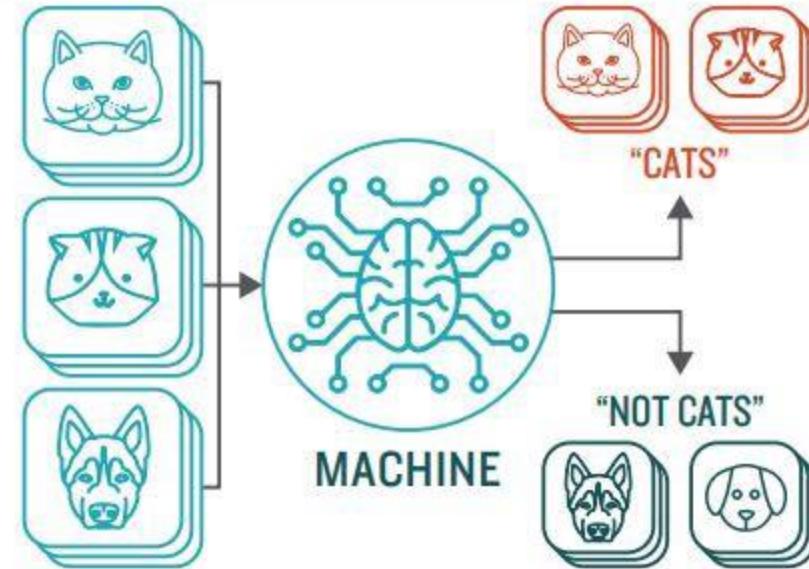
## STEP 1

Provide the machine learning algorithm categorized or "labeled" input and output data from to learn

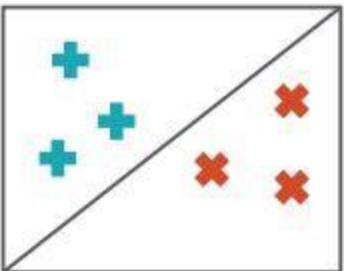


## STEP 2

Feed the machine new, unlabeled information to see if it tags new data appropriately. If not, continue refining the algorithm

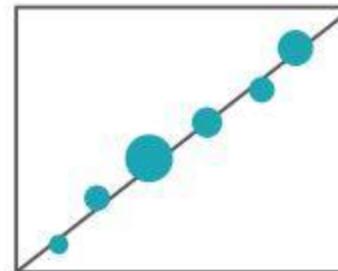


## TYPES OF PROBLEMS TO WHICH IT'S SUITED



### CLASSIFICATION

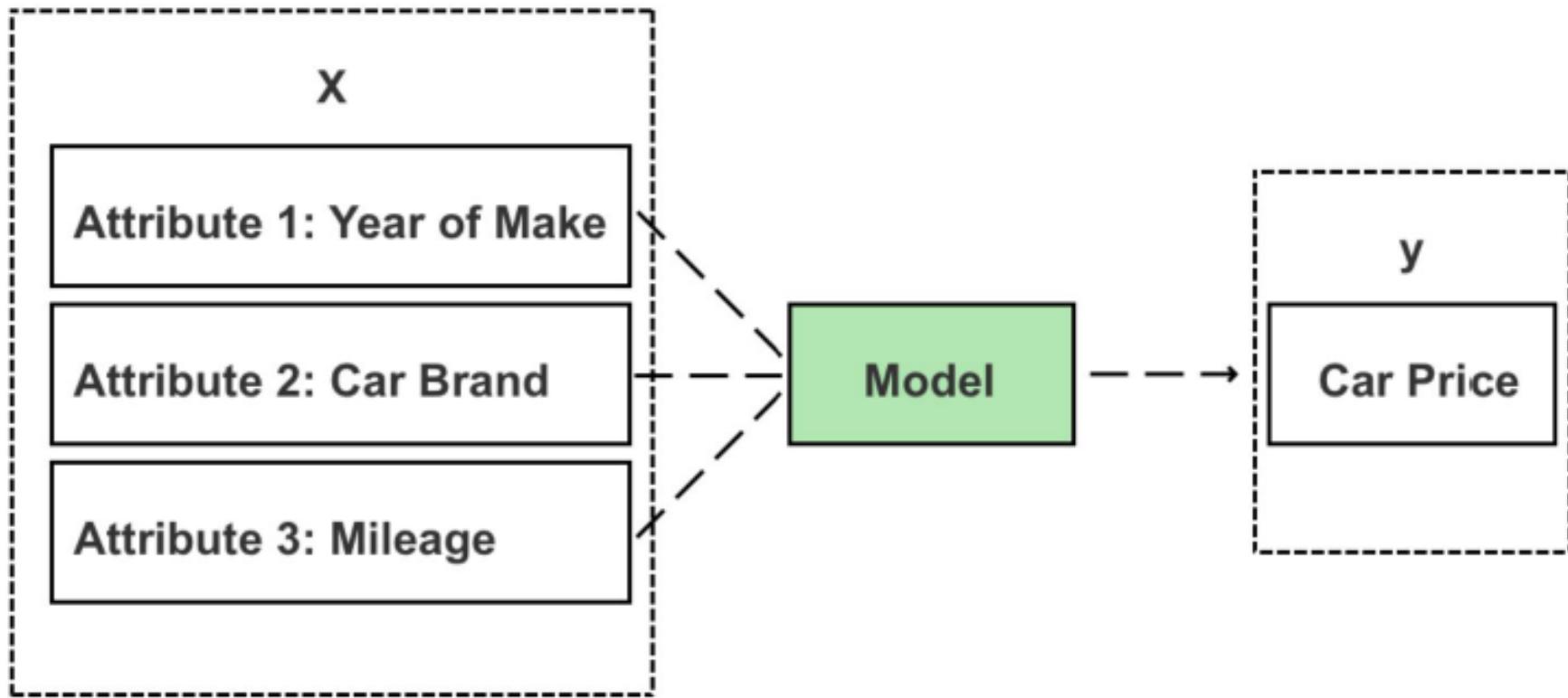
Sorting items into categories



### REGRESSION

Identifying real values (dollars, weight, etc.)

# REGRESSION





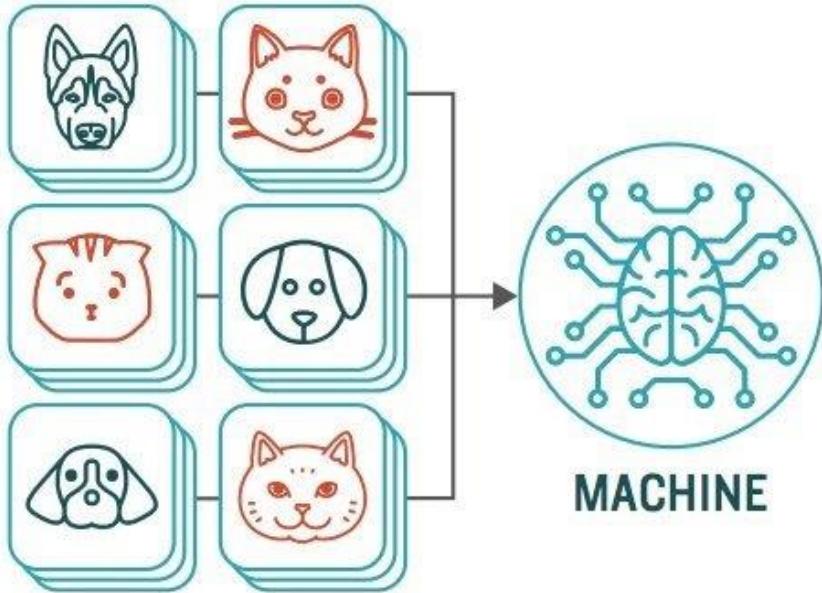
# UNSUPERVISED LEARNING

focuses on analyzing relationships between input variables and uncovering hidden patterns that can be extracted to create new labels regarding possible outputs.

# How **Unsupervised** Machine Learning Works

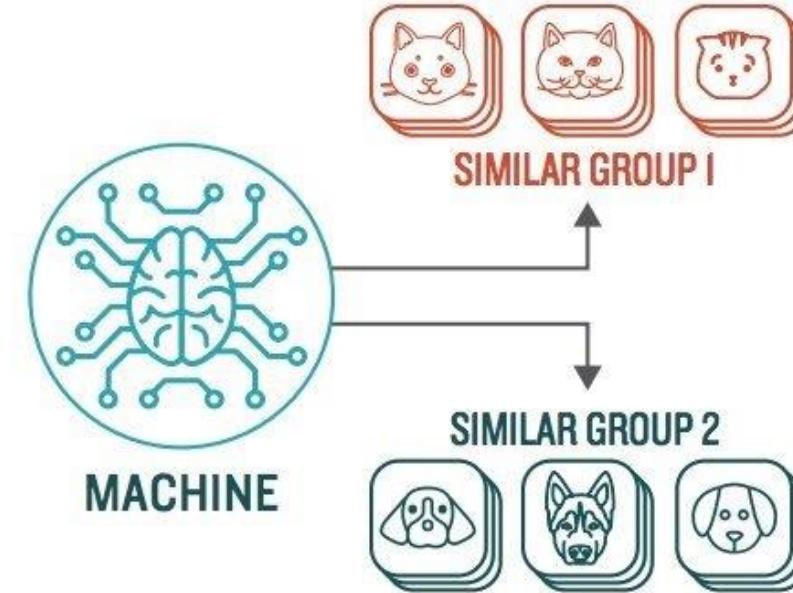
## STEP 1

Provide the machine learning algorithm uncategorized, unlabeled input data to see what patterns it finds

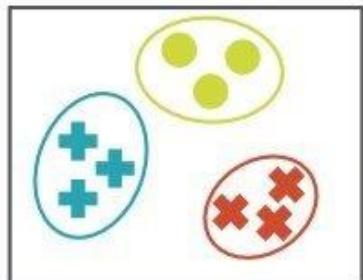


## STEP 2

Observe and learn from the patterns the machine identifies



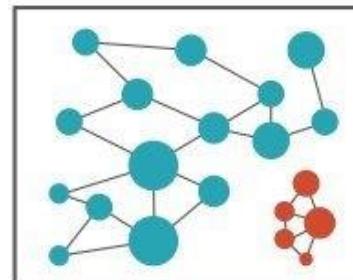
## TYPES OF PROBLEMS TO WHICH IT'S SUITED



### CLUSTERING

Identifying similarities in groups

For Example: Are there patterns in the data to indicate certain patients will respond better to this treatment than others?



### ANOMALY DETECTION

Identifying abnormalities in data

For Example: Is a hacker intruding in our network?



## REINFORCEMENT LEARNING

achieve a specific goal (output) by randomly trialing a vast number of possible input combinations and grading their performance

# REINFORCEMENT LEARNING



**Isilah daftar kelompok yang ada di EMAS**



# TK-01

1. Jelaskan mengenai regresi, klasifikasi, dan *clustering* dalam machine learning
2. Jelaskan mengenai “generative AI” dan apa perbedaannya dengan 3 jenis algoritma tadi (regresi, klasifikasi, *clustering*)
3. Buatlah contoh permasalahan untuk mengkategorikan contoh kasus dari ketiga algoritma machine learning:

- Classification
- Regression
- Clustering

Masing-masing 1 kasus

Ambil dari referensi buku atau Internet (tulis sumbernya)

# Contoh:

Masalah:

Misalkan Anda diminta untuk membuat sistem pengolahan data cuaca di Indonesia. Diinginkan sistem tersebut dapat memetakan daerah mana saja yang memiliki profil cuaca yang sama/mirip.

Jenis:

**CLUSTERING**

Success is no accident. It is hard work,  
perseverance, learning, studying, sacrifice  
and most of all, love of what you are doing  
or learning to do.

*– Pele*



# KECERDASAN BUATAN

2| DATA DALAM MACHINE LEARNING

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI

# 2 | DATA DALAM MACHINE LEARNING

**Sub CPMK 1.2** Mampu menjelaskan tahapan dalam machine learning (C3)

**Materi** Persiapan data, modelling, correctness, bias-variance, feature extraction & selection, training-testing

**Referensi** Theobald, ch. 2, 3, 4 & Gruss, ch 11

## Indikator:

Mampu menjelaskan tahap persiapan data

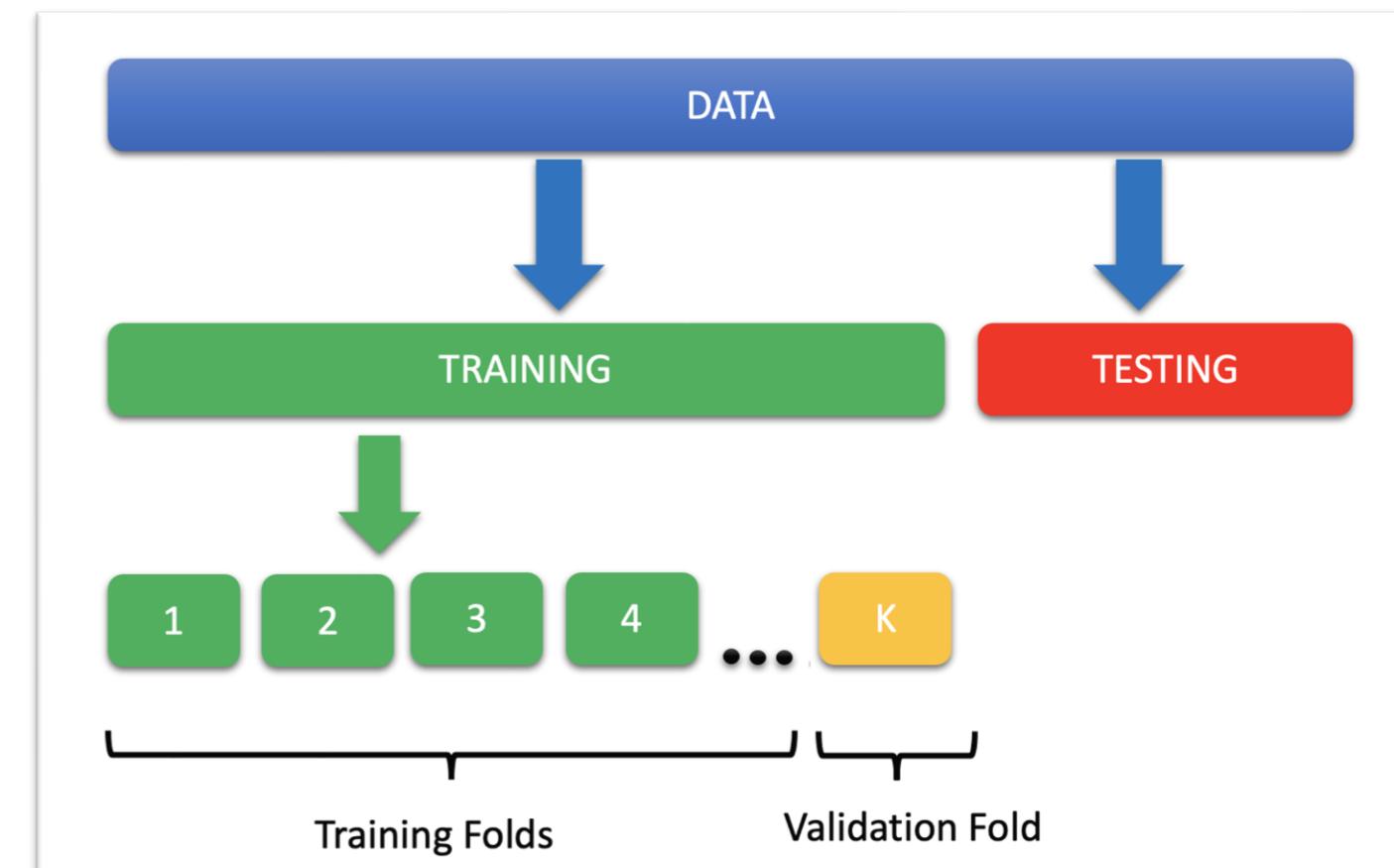
Mampu menjelaskan tahap memodelkan machine learning

Mampu menjelaskan bias & variance

Mampu menjelaskan correctness

Mampu menjelaskan feature extraction & selection

Mampu menjelaskan strategi training, testing dan validation



	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

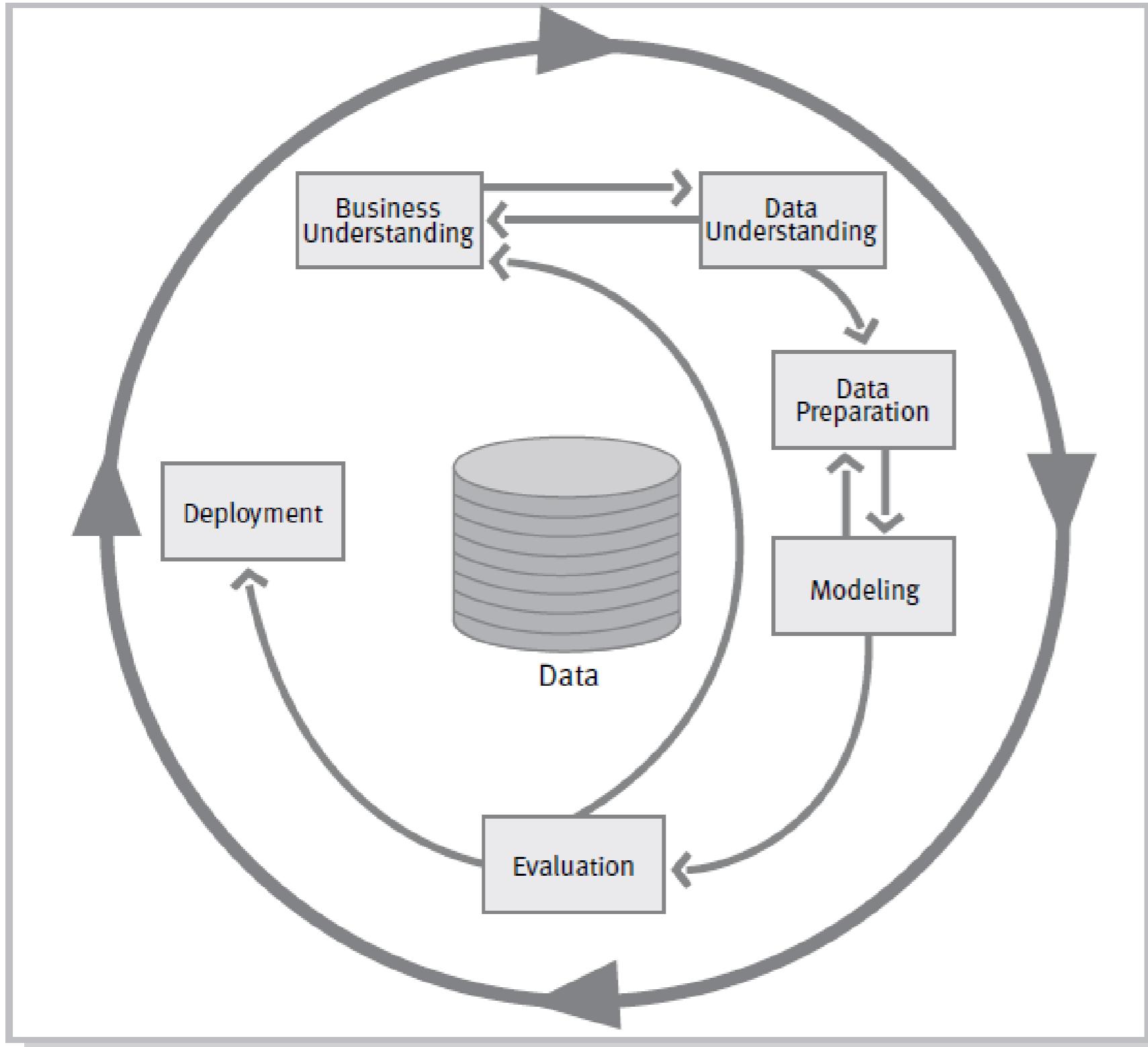


Contoh kasus, tabel dan gambar yang digunakan dalam materi ini diambil dari buku teks yang dijadikan referensi di mata kuliah ini, yaitu:

Oliver Theobald, **Machine Learning for Absolute Beginners: A Plain English Introduction**, Independently published, 2018, Chapter 4, 5, 6.

Joel Gruss, **Data Science from Scratch**, O'Reilly, 2015, Chapter 10

# CRISP-DM Model

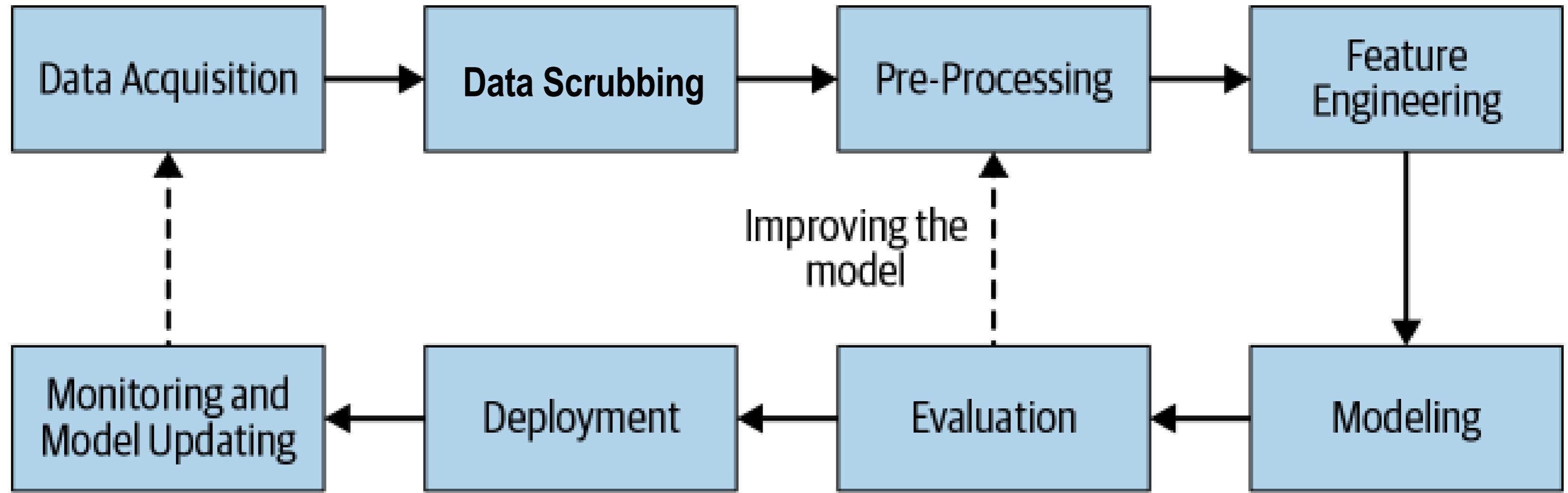


Source: Pete Chapman (NCR), Julian Clinton (SPSS),  
Randy Kerber (NCR),  
Thomas Khabaza (SPSS), Thomas Reinartz  
(DaimlerChrysler),  
Colin Shearer (SPSS) and Rüdiger Wirth  
(DaimlerChrysler), **CRISP-DM 1.0, Step-by-step data  
mining guide**



Business Understanding	Data Understanding	Data Preparation	Modeling	Evaluation	Deployment
<b>Determine Business Objectives</b> <i>Background Business Objectives</i> <i>Business Success Criteria</i>	<b>Collect Initial Data</b> <i>Initial Data Collection Report</i>	<b>Select Data</b> <i>Rationale for Inclusion/Exclusion</i>	<b>Select Modeling Techniques</b> <i>Modeling Technique</i> <i>Modeling Assumptions</i>	<b>Evaluate Results</b> <i>Assessment of Data Mining Results w.r.t. Business Success Criteria</i> <i>Approved Models</i>	<b>Plan Deployment</b> <i>Deployment Plan</i>
<b>Assess Situation</b> <i>Inventory of Resources Requirements, Assumptions, and Constraints</i> <i>Risks and Contingencies</i> <i>Terminology</i> <i>Costs and Benefits</i>	<b>Describe Data</b> <i>Data Description Report</i>	<b>Clean Data</b> <i>Data Cleaning Report</i>	<b>Generate Test Design</b> <i>Test Design</i>	<b>Review Process</b> <i>Review of Process</i>	<b>Plan Monitoring and Maintenance</b> <i>Monitoring and Maintenance Plan</i>
<b>Determine Data Mining Goals</b> <i>Data Mining Goals</i> <i>Data Mining Success Criteria</i>	<b>Explore Data</b> <i>Data Exploration Report</i>	<b>Construct Data</b> <i>Derived Attributes</i> <i>Generated Records</i>	<b>Build Model</b> <i>Parameter Settings</i> <i>Models</i> <i>Model Descriptions</i>	<b>Determine Next Steps</b> <i>List of Possible Actions</i> <i>Decision</i>	<b>Produce Final Report</b> <i>Final Report</i> <i>Final Presentation</i>
<b>Produce Project Plan</b> <i>Project Plan</i> <i>Initial Assessment of Tools and Techniques</i>	<b>Verify Data Quality</b> <i>Data Quality Report</i>	<b>Integrate Data</b> <i>Merged Data</i>	<b>Format Data</b> <i>Reformatted Data</i>	<b>Assess Model</b> <i>Model Assessment</i> <i>Revised Parameter Settings</i>	<b>Review Project</b> <i>Experience Documentation</i>

# Pipeline





UNIVERSITAS  
INDONESIA

*Veritas, Probitas, Justitia*

# Data scrubbing

# DATA SCRUBBING = DATA CLEANING

- Like most varieties of fruit, datasets need upfront cleaning and human manipulation before they are ready for consumption.
- The technical process of refining your dataset to make it more workable.
- Modifying and removing incomplete, incorrectly formatted, irrelevant or duplicated data, converting text-based data to numeric values and the redesigning of features.
- For data practitioners, data scrubbing typically demands the greatest application of time and effort. → 79 – 90%

# FEATURE SELECTION

- Identify which variables are most relevant to your hypothesis or objective.
- Selective in choosing the variables you include in your model.
  - Rather than creating a four-dimensional scatterplot with four features in your model, an opportunity may present to select two highly relevant features and build a two-dimensional plot that is easier to interpret and visualize.
  - Preserving features that don't correlate strongly with the output value can manipulate and derail the model's accuracy.

# Example: identify variables that contribute to a language becoming endangered

Name in English	Name in Spanish	Countries	Country Code	Num. of Speakers
South Italian	Napolitano -calabres	Italy	ITA	7500000
Sicilian	Siciliano	Italy	ITA	5000000
Low Saxon	Bajo Sajón	Germany, Denmark, Netherlands, Poland, Russian Federation	DEU, DNK, NLD, POL, RUS	4800000
Belarusian	Bielorruso	Belarus, Latvia, Lithuania, Poland, Russian Federation, Ukraine	BRB, LVA, LTU, POL, RUS, UKR	4000000
Lombard	Lombardo	Italy, Switzerland	ITA, CHE	3500000
Romani	Romaní	Albania, Germany, Austria, Belarus, Bosnia and Herzegovina, Bulgaria, Croatia, Estonia, Finland, France, Greece, Hungary, Italy, Latvia, Lithuania, The former Yugoslav Republic of Macedonia, Netherlands, Poland, Romania, United Kingdom of Great Britain and Northern Ireland, Russian Federation, Slovakia, Slovenia, Switzerland, Czech Republic, Turkey, Ukraine, Serbia, Montenegro	ALB, DEU, AUT, BRB, BIH, BGR, HRV, EST, FIN, FRA, GRC, HUN, ITA, LVA, LTU, MKD, NLD, POL, ROU, GBR, RUS, SVK, SVN, CHE, CZE, TUR, UKR, SRB, MNE	3500000
Yiddish	Yiddish	Israel	ISR	3000000
Gondi	Gondi	India	IND	2713790

Table 3: Endangered languages, database: <https://www.kaggle.com/the-guardian/extinct-languages>

# Scrubbing the data

- The language's "Name in Spanish" may not contribute to any insight → delete this vector (column) from the dataset.
- The dataset contains duplicated information in the form of separate vectors for "Countries" and "Country Code." Analyzing both of these vectors doesn't provide any additional insight; hence, we can choose to delete one and retain the other.



Name in English	Name in Spanish	Countries	Country Code	Num. of Speakers
South Italian	Napolitano - Siciliano	Italy	ITA	7500000
Sicilian	Siciliano	Italy	ITA	5000000
Low Saxon	Bajo Sajón	Germany, Denmark, Netherlands, Poland, Russian Federation	DEU, DNK, NLD, POL, RUS	4800000
Belarusian	Bielorruso	Belarus, Latvia, Lithuania, Poland, Russian Federation, Ukraine	BRB, LVA, LTU, POL, RUS, UKR	4000000
Lombard	Lombardo	Italy, Switzerland	ITA, CHE	3500000
Romani	Romaní	Albania, Germany, Austria, Belarus, Bosnia and Herzegovina, Bulgaria, Croatia, Estonia, Finland, France, Greece, Hungary, Italy, Latvia, Lithuania, The former Yugoslav Republic of Macedonia, Netherlands, Poland, Romania, United Kingdom of Great Britain and Northern Ireland, Russian Federation, Slovakia, Slovenia, Switzerland, Czech Republic, Turkey, Ukraine, Serbia, Montenegro	ALB, DEU, AUT, BRB, BIH, BGR, HRV, EST, FIN, FRA, GRC, HUN, ITA, LVA, LTU, MKD, NLD, POL, ROU, GBR, RUS, SVK, SVN, CHE, CZE, TUR, UKR, SRB, MNE	3500000
Yiddish	Yiddish	Israel	ISR	3000000
Gondi	Gondi	India	IND	2713790

# FEATURE REDUCTION (DIMENSION REDUCTION)

- Another method: roll multiple features into one

	Protein Shake	Nike Sneakers	Adidas Boots	Fitbit	Powerade	Protein Bar	Fitness Watch	Vitamins
Buyer 1	1	1	0	1	0	5	1	0
Buyer 2	0	0	0	0	0	0	0	1
Buyer 3	3	0	1	0	5	0	0	0
Buyer 4	1	1	0	0	10	1	0	0

Table 4: Sample product inventory

# MERGING SIMILAR FEATURES

- Reduce the number of columns by merging similar features into fewer columns.
- Remove individual product names and replace the eight product items with a lower number of categories or subtypes.
- As all product items fall under the category of “fitness,” we can sort by product subtype and compress the columns from eight to three: “Health Food,” “Apparel,” and “Digital.”



	Protein Shake	Nike Sneakers	Adidas Boots	Fitbit	Powerade Bar	Protein Watch	Fitness	Vitamins
Buyer 1	1	1	0	1	0	5	1	0
Buyer 2	0	0	0	0	0	0	0	1
Buyer 3	3	0	1	0	5	0	0	0
Buyer 4	1	1	0	0	10	1	0	0



Buyers will be recommended health food when they buy other health food or when they buy apparel (depending on the degree of correlation)

	Health Food	Apparel	Digital
Buyer 1	6	1	2
Buyer 2	1	0	0
Buyer 3	8	1	0
Buyer 4	12	1	0

Synthesized product inventory

- **Pros:** transform the dataset in a way that preserves and captures information using fewer variables.
- **Cons:** less information about the relationships between specific products.
- Trade-off between convenience and the overall precision of the model.

# Row Compression

- Reduce the number of rows and thereby compress the total number of data points.
- This may involve merging two or more rows into one,

Before				
Animal	Meat Eater	Legs	Tail	Race Time
Tiger	Yes	4	Yes	2:01 mins
Lion	Yes	4	Yes	2:05 mins
Tortoise	No	4	No	55:02 mins

After				
Animal	Meat Eater	Legs	Tail	Race Time
Carnivore	Yes	4	Yes	2:03 mins
Tortoise	No	4	No	55:02 mins

it's possible to merge the two rows because they possess the same categorical values for all features except Race Time—which can be easily aggregated.

The race time of the Tiger and the Lion can be added and divided by two.

Table 6: Example of row merge

- Numeric values are normally easy to aggregate given they are not categorical.
  - Beware: it would be impossible to aggregate an animal with four legs and an animal with two legs! We obviously can't merge these two animals and set "three" as the aggregate number of legs.
- Row compression can also be challenging to implement in cases where numeric values aren't available.
- Row compression is usually less attainable than feature compression and especially for datasets with a high number of features.

# ONE HOT ENCODING

- Aside from set text-based values such as True/False (that automatically convert to “1” and “0” respectively), most algorithms are not compatible with non- numeric data.
- Transforms values into binary form,
  - “1” or “0”—“True” or “False.”
  - “0,” representing False, means that the value does not belong to this particular feature,
  - “1”—True or “hot”— confirms that the value does belong to this feature.



Name in English	Speakers	Degree of Endangerment
<b>South Italian</b>	7500000	Vulnerable
<b>Sicilian</b>	5000000	Vulnerable
<b>Low Saxon</b>	4800000	Vulnerable
<b>Belarusian</b>	4000000	Vulnerable
<b>Lombard</b>	3500000	Definitely endangered
<b>Romani</b>	3500000	Definitely endangered
<b>Yiddish</b>	3000000	Definitely endangered
<b>Gondi</b>	2713790	Vulnerable
<b>Picard</b>	700000	Severely endangered

Table 7: Endangered languages

Name in English	Speakers	Vulnerable	Definitely Endangered	Severely Endangered
<b>South Italian</b>	7500000	1	0	0
<b>Sicilian</b>	5000000	1	0	0
<b>Low Saxon</b>	4800000	1	0	0
<b>Belarusian</b>	4000000	1	0	0
<b>Lombard</b>	3500000	0	1	0
<b>Romani</b>	3500000	0	1	0
<b>Yiddish</b>	3000000	0	1	0
<b>Gondi</b>	2713790	1	0	0
<b>Picard</b>	700000	0	0	1

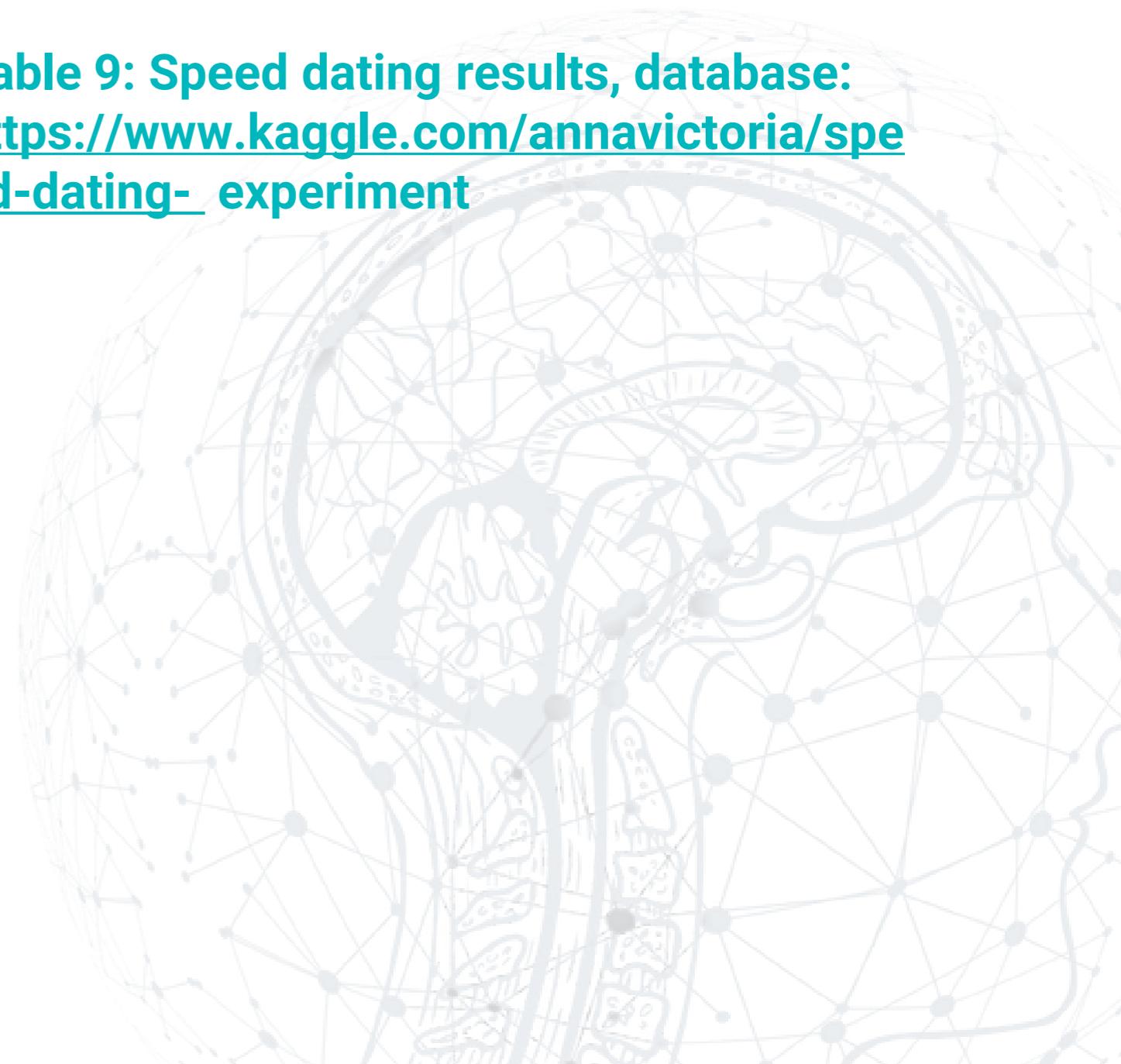
Table 8: Example of one-hot encoding

- **Pros:** Many of machine learning algorithms “likes” this kind of input.
- **Cons:** more dataset features, which may slightly extend processing time
- One hack to minimize the total number of features is to restrict binary cases to a single column.
  - A speed dating dataset on kaggle.com lists “Gender” in a single column using one-hot encoding. Rather than create discrete columns for both “Male” and “Female,” they merged these two features into one. According to the dataset’s key, females are denoted as “0” and males as “1.” The creator of the dataset also used this technique for “Same Race” and “Match.”

Subject Number	Gender	Same Race	Age	Match
1	0	0	27	0
1	0	0	22	0
1	0	1	22	1
1	0	0	23	1
1	0	0	24	1
1	0	0	25	0
1	0	0	30	0

<b>Gender:</b>	<b>Same Race:</b>	<b>Match:</b>
Female = 0	No = 0	No = 0
Male = 1	Yes = 1	Yes = 1

**Table 9: Speed dating results, database:  
<https://www.kaggle.com/annavictoria/speed-dating-experiment>**

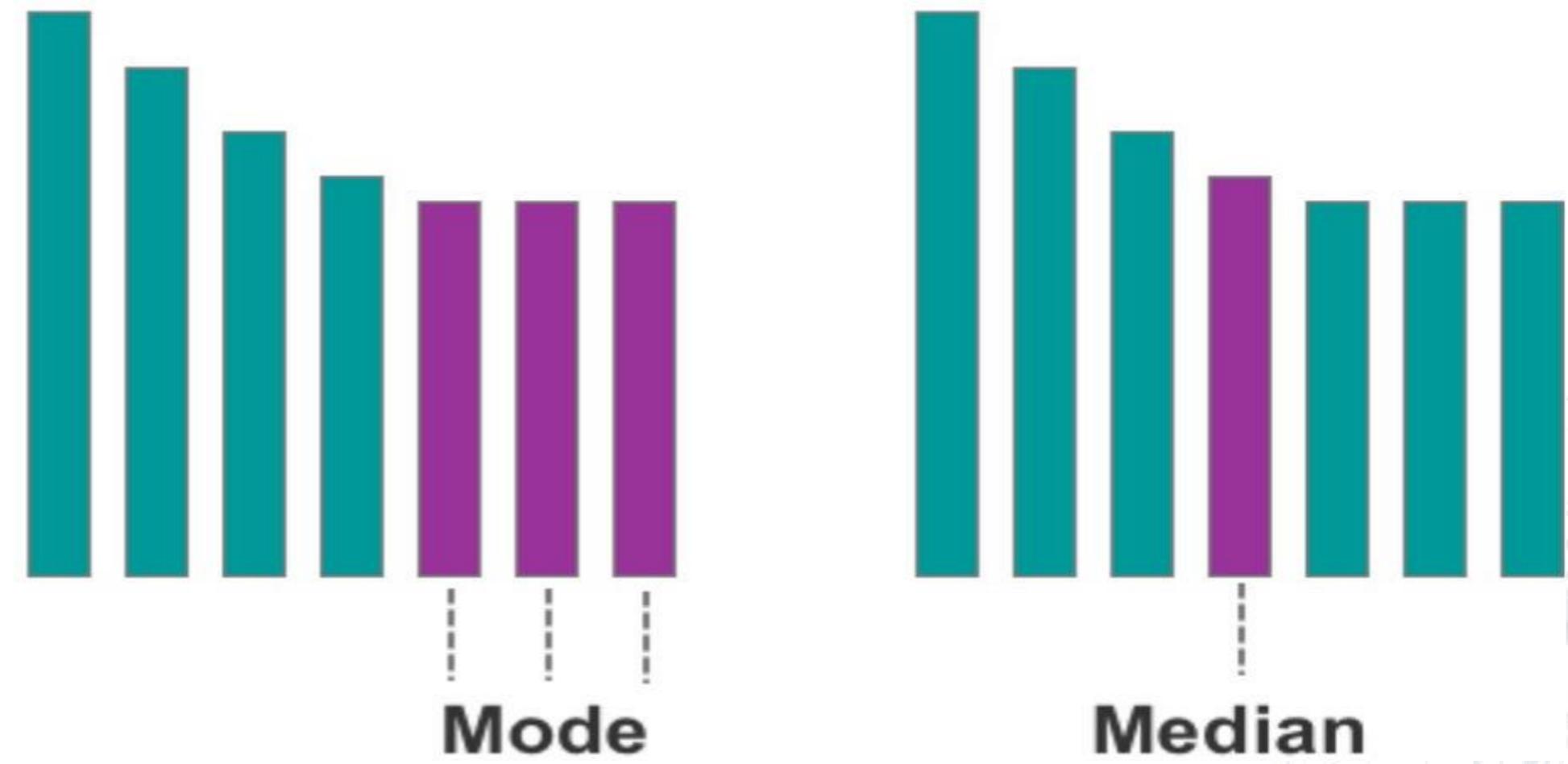


# BINNING

- Binning is another method of feature engineering but is used to convert numeric values into a category.
- Whoa, hold on! Aren't numeric values a good thing? Yes, in most cases numeric values are preferred as they are compatible with a broader selection of algorithms. Where numeric values are not ideal, is in situations where they list variations irrelevant to the goals of your analysis.
- Let's take house price evaluation as an example. The exact measurements of a tennis court might not matter greatly when evaluating house prices; the relevant information is whether the house has a tennis court. This logic probably also applies to the garage and the swimming pool, where the existence or non-existence of the variable is generally more influential than their specific measurements.
- The solution here is to replace the numeric measurements of the tennis court with a True/False feature or a categorical value such as "small," "medium," and "large." Another alternative would be to apply one-hot encoding with "0" for homes that do not have a tennis court and "1" for homes that do have a tennis court.

# Missing Data

- Missing values in your dataset can be equally frustrating and interfere with your analysis and the model's predictions.
- Approximate the missing value by mode value or median value
- **Mode value:** represents the single most common variable value available in the dataset.
  - This works best with categorical and binary variable types, such as one to five-star rating systems and positive/negative drug tests respectively.
- **Median value:** adopts the value(s) located in the middle of the dataset.
  - This works best with continuous variables, which have an infinite number of possible values, such as house prices.
- As a last resort, rows with missing values can be removed altogether.



**Figure 10: A visual example of the mode and median respectively**



UNIVERSITAS  
INDONESIA

*Veritas, Probitas, Iustitia*

# Setting up data

# Split validation

	Variable 1	Variable 2	Variable 3
Training Data	Row 1		
Test Data	Row 8		
Row 2			
Row 3			
Row 4			
Row 5			
Row 6			
Row 7			
Row 9			
Row 10			

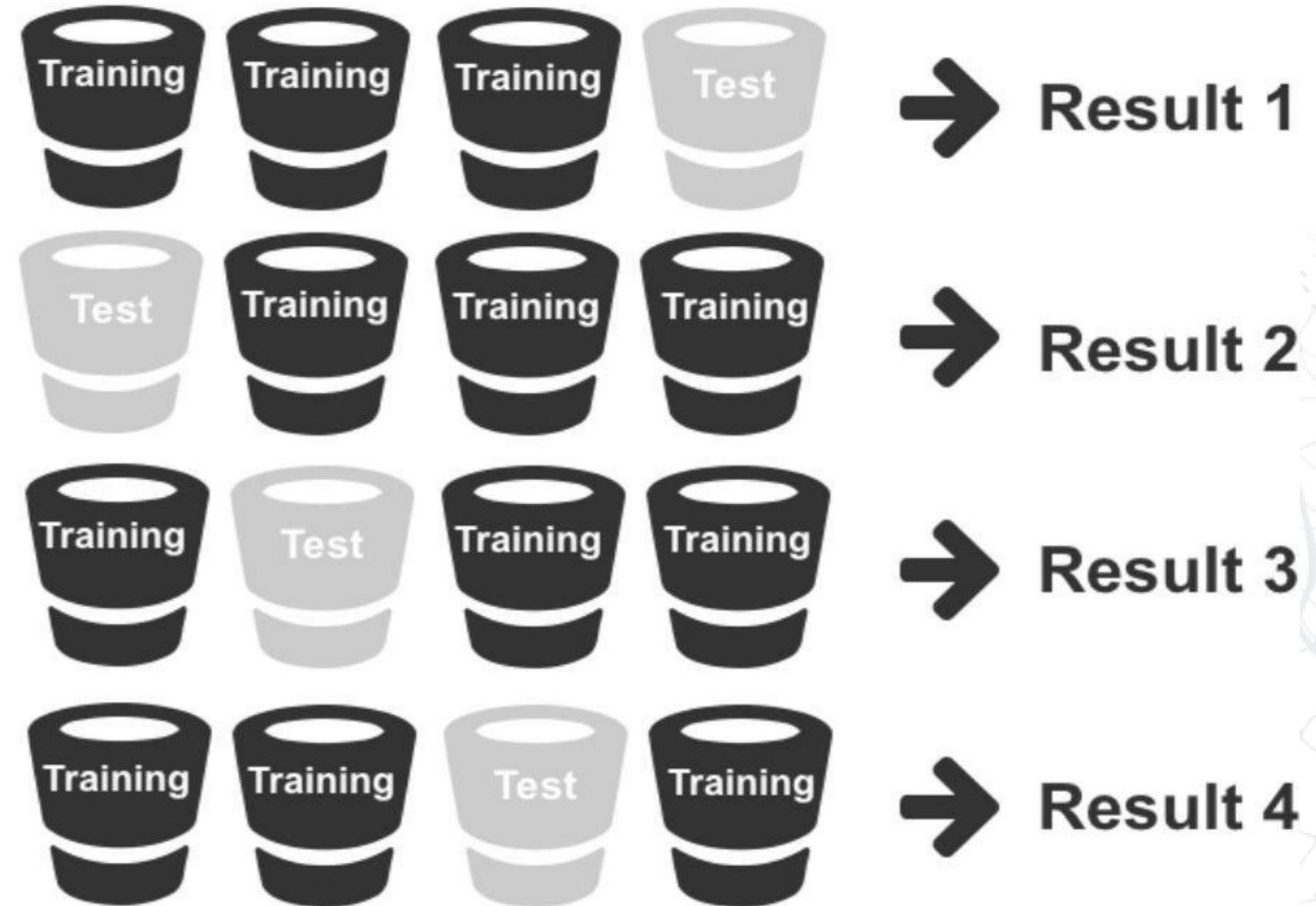
- The ratio of the two splits should be approximately 70/30 or 80/20.
- First: randomize the row order
  - This helps to avoid bias in your model, as your original dataset might be arranged alphabetically or sequentially according to when the data was collected.

Figure 11: 70/30 partitioning of training and test data

# Cross Validation

- Exhaustive cross validation
  - Finding and testing all possible combinations to divide the original sample into a training set and a test set.
- K-fold validation.
  - Splitting data into  $k$  assigned buckets and reserving one of those buckets for testing the training model at each round.
  - Data are randomly assigned .
  - One bucket is reserved as the test bucket and is used to measure and evaluate the performance of the remaining ( $k-1$ ) buckets.

## Buckets



**Figure 12: k-fold validation**

# How Much Data Do I Need?

- At minimum  $\sum \text{samples} \geq 10 \times \text{features}$ 
  - E.g. for a small dataset with 5 features, the training data should ideally have at least 50 rows.
- Generally, the more relevant data you have available as training data, the more combinations you can incorporate into your prediction model
- In general, machine learning works best when your training dataset includes a full range of feature combinations.

# What does a full range of feature combinations look like?

- Imagine you have a dataset about data scientists. The features :
  - University degree (X)
  - 5+ years of professional experience (X)
  - Children (X)
  - Salary (y)
- We need to know the salary for data scientists with a university degree and 5+ years of professional experience who don't have children, as well as data scientists with a university degree and 5+ years of professional experience who do have children.

# Matching data to an algorithm

- For datasets with less than 10,000 samples, clustering and dimensionality reduction algorithms can be highly effective
- Regression analysis and classification algorithms are more suitable for datasets with less than 100,000 samples.
- Neural networks require even more samples to run effectively and are more cost-effective and time-efficient for working with massive quantities of data.



UNIVERSITAS  
INDONESIA

*Veritas, Probitas, Justitia*

# Post analysis data

# What is a good model?

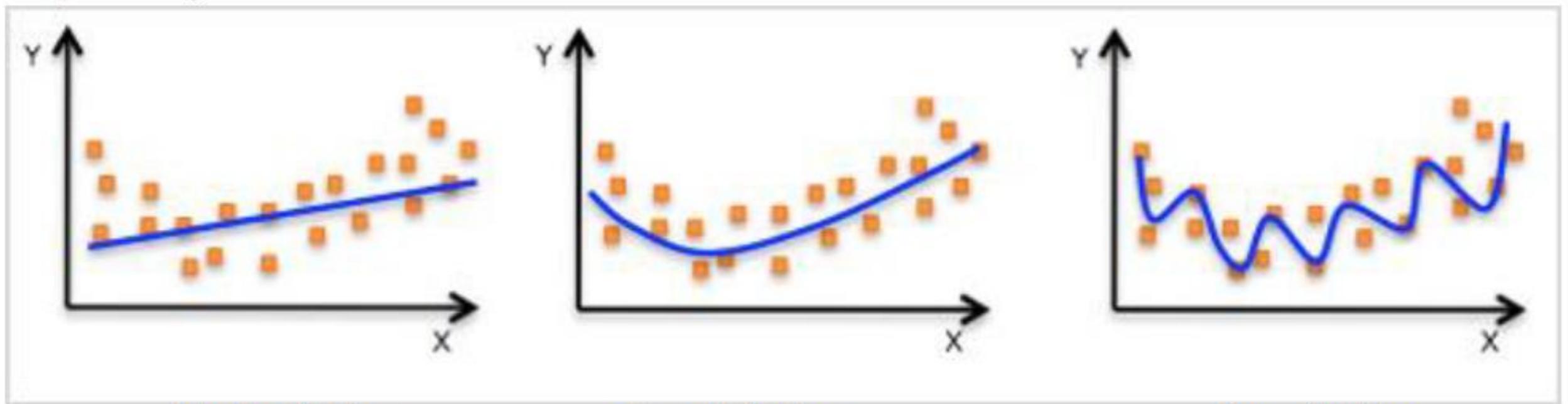
- Generalization capability
  - Can it accurately predict the actual service data?
- Interpretability
  - Is the prediction result easy to interpret?
- Prediction speed
  - How long does it take to predict each piece of data?
- Practicability
  - Is the prediction rate still acceptable when the service volume increases with a huge data volume?

# Error

- Difference between the sample result predicted by the model obtained after learning and the actual sample result.
- **Training error:** error that you get when you run the model on the training data.
- **Generalization error (testing error):** error that you get when you run the model on new samples. Obviously, we prefer a model with a smaller generalization error.

# Underfitting vs Overfitting

- Underfitting: occurs when the model or the algorithm does not fit the data well enough.
- Overfitting: occurs when the training error is small but the generalization error is large (poor generalization capability).



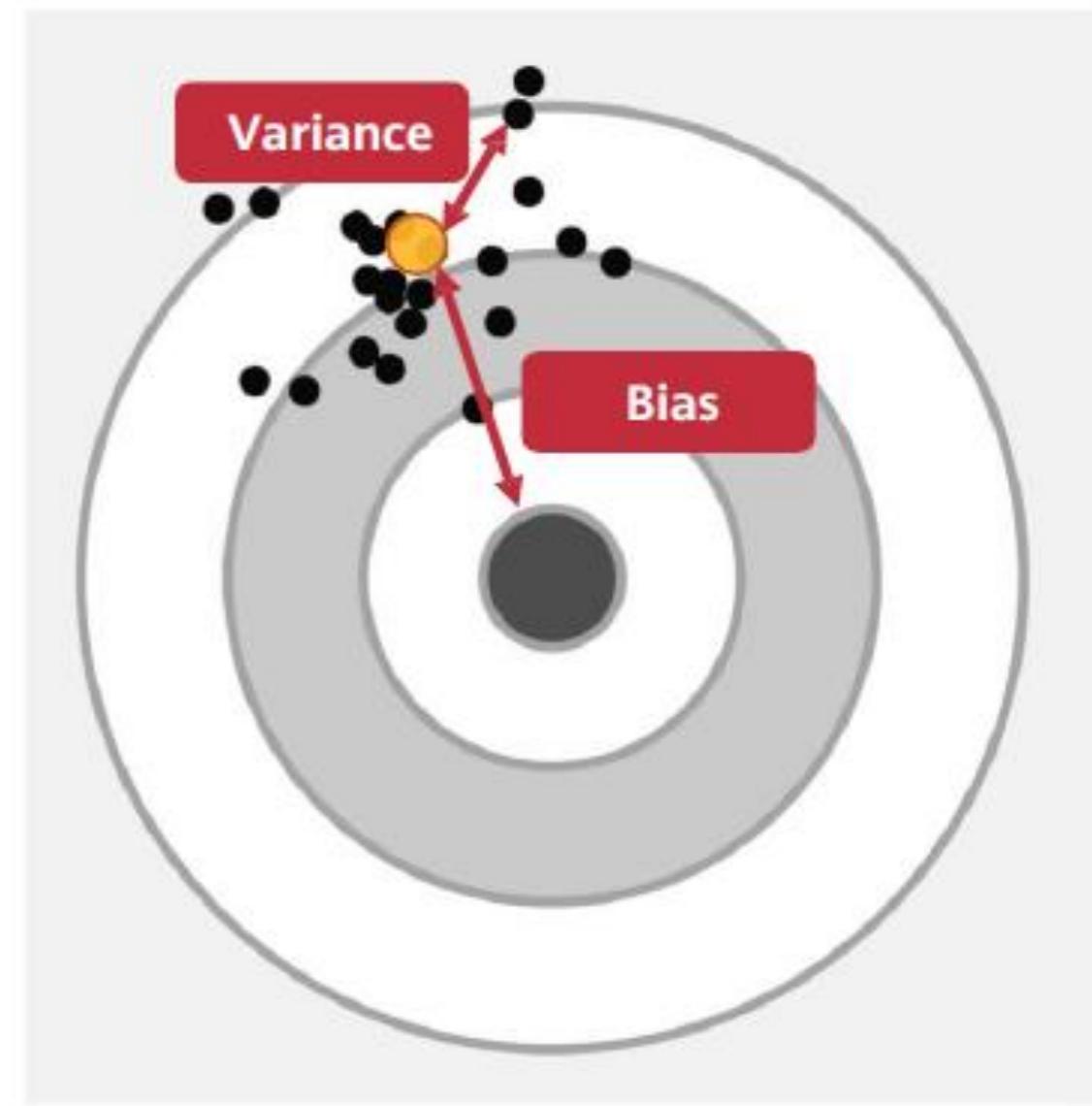
Underfitting  
Not all features are learned.

Good fitting

Overfitting  
Noises are learned.

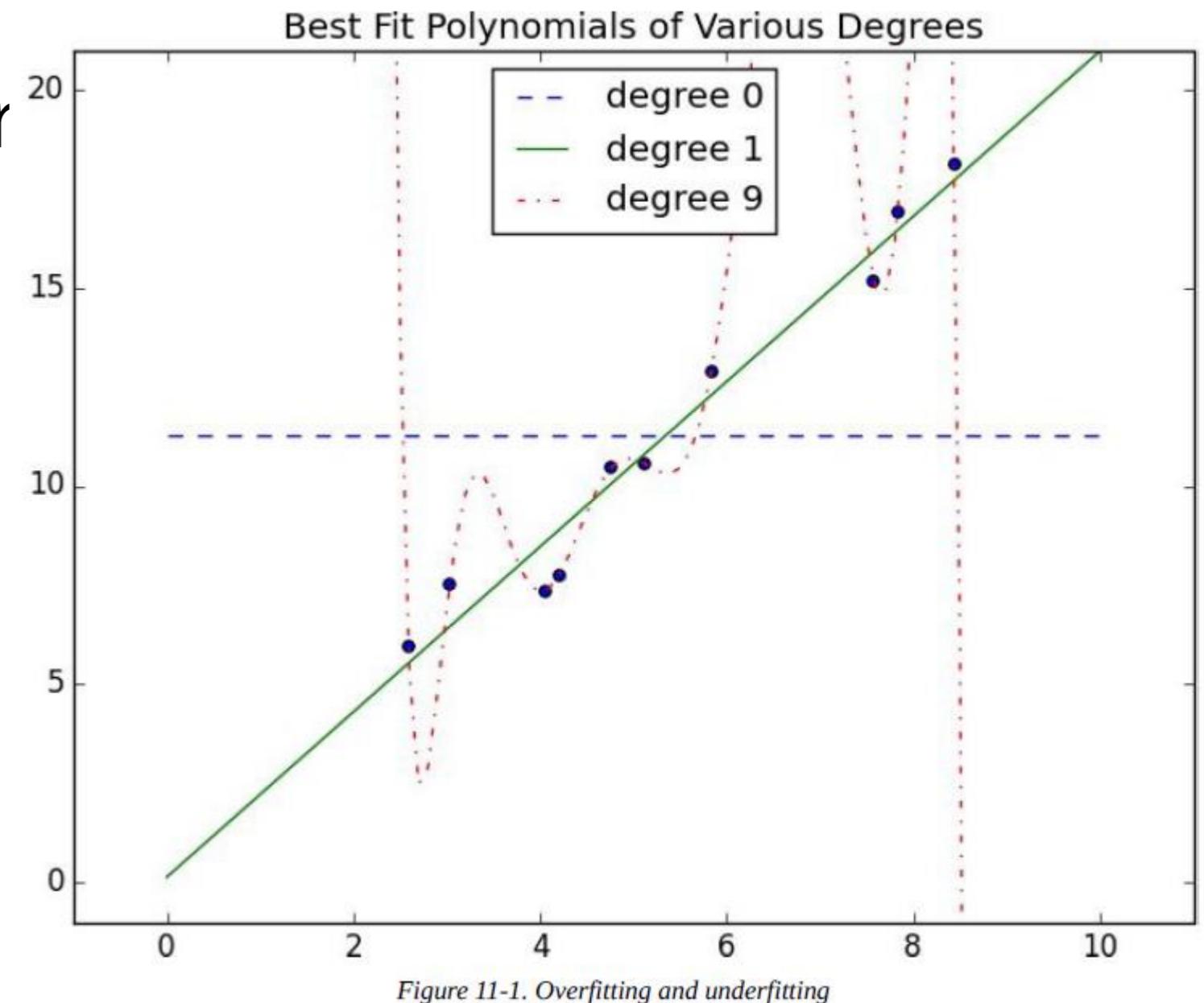
# Overfitting Cause – Error

- Generally, the prediction error can be divided into two types:
  - Error caused by "bias"
  - Error caused by "variance"
- Variance:
  - Offset of the prediction result from the average value
  - Error caused by the model's sensitivity to small fluctuations in the training set
- Bias:
  - Difference between the expected (or average) prediction value and the correct value we are trying to predict.



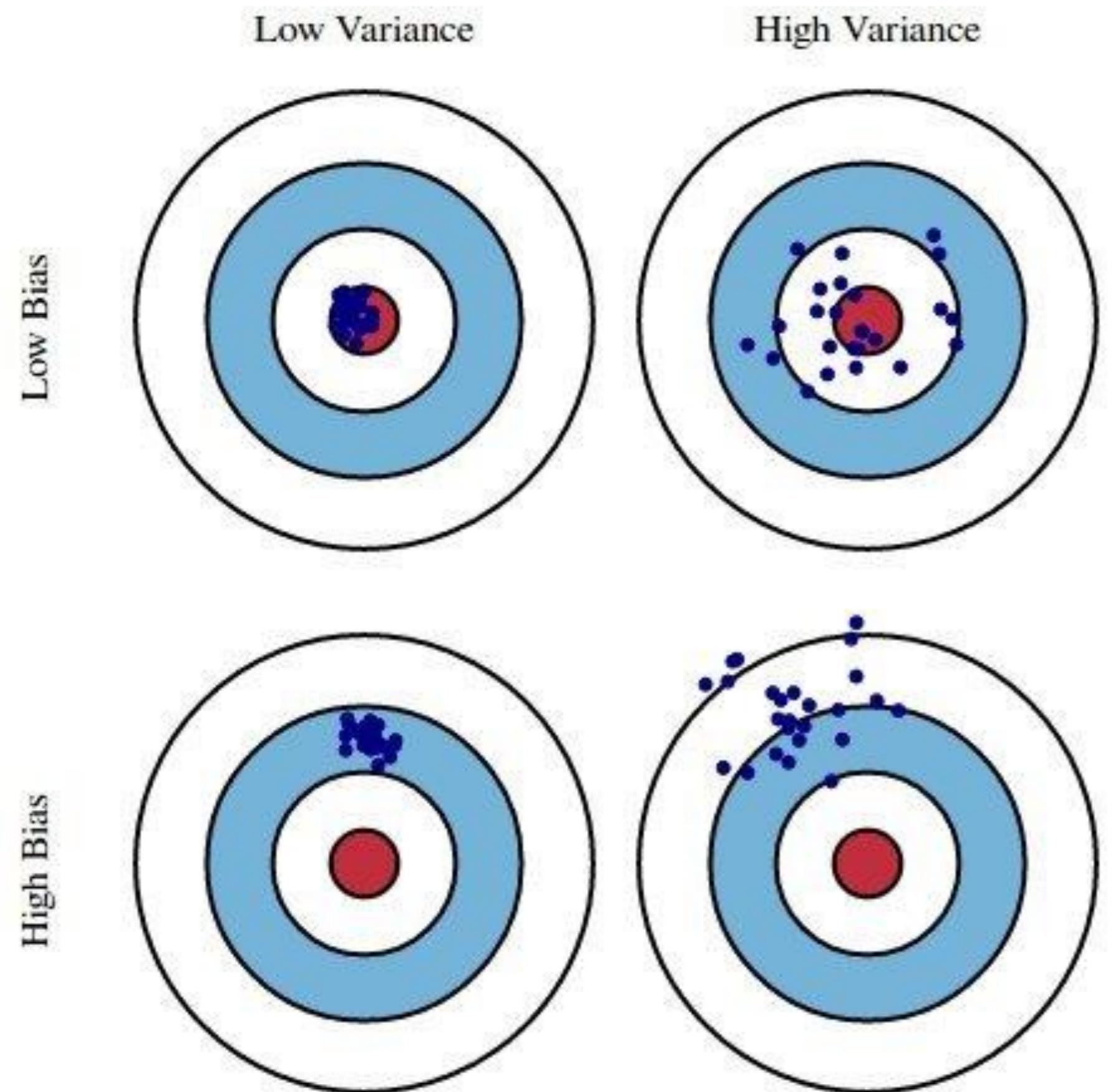
# Bias – Variance

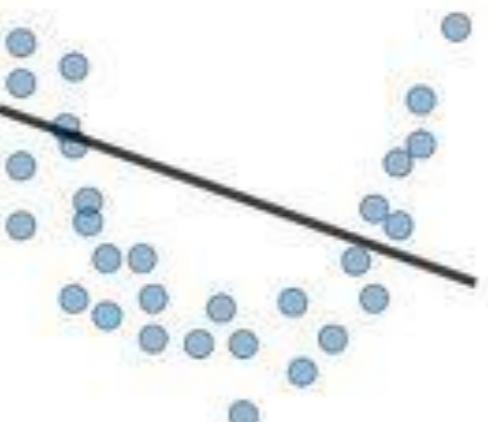
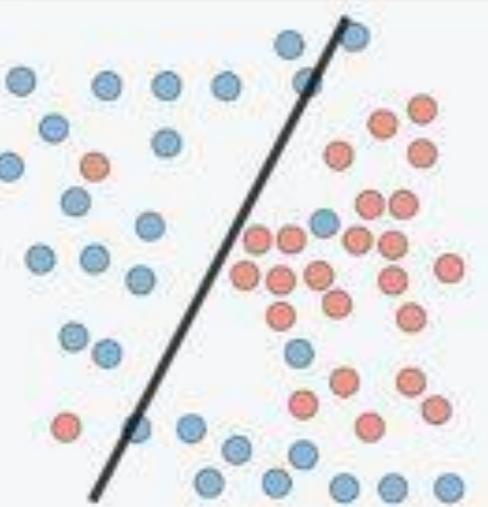
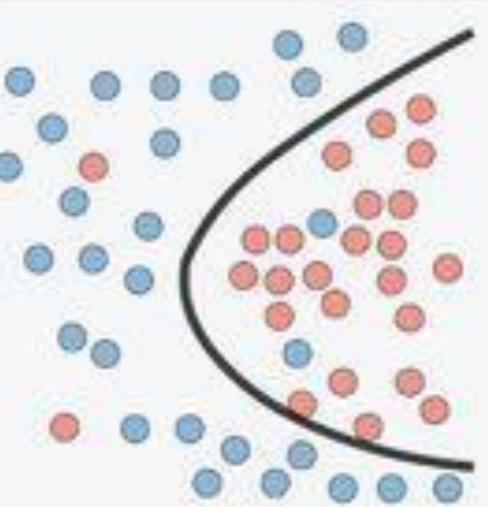
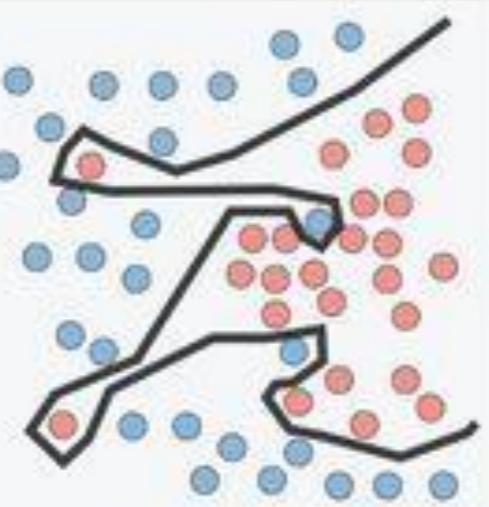
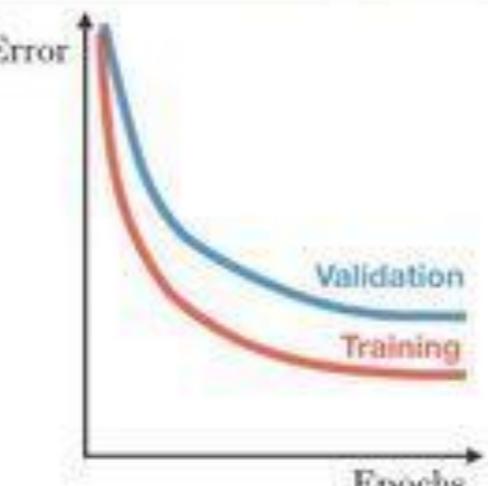
- If your model has high bias (which means it performs poorly even on your training data) then one thing to try is adding more features. Going from the degree 0 model in to the degree 1 model was a big improvement.
- If your model has high variance, then you can similarly remove features. But another solution is to obtain more data (if you can).

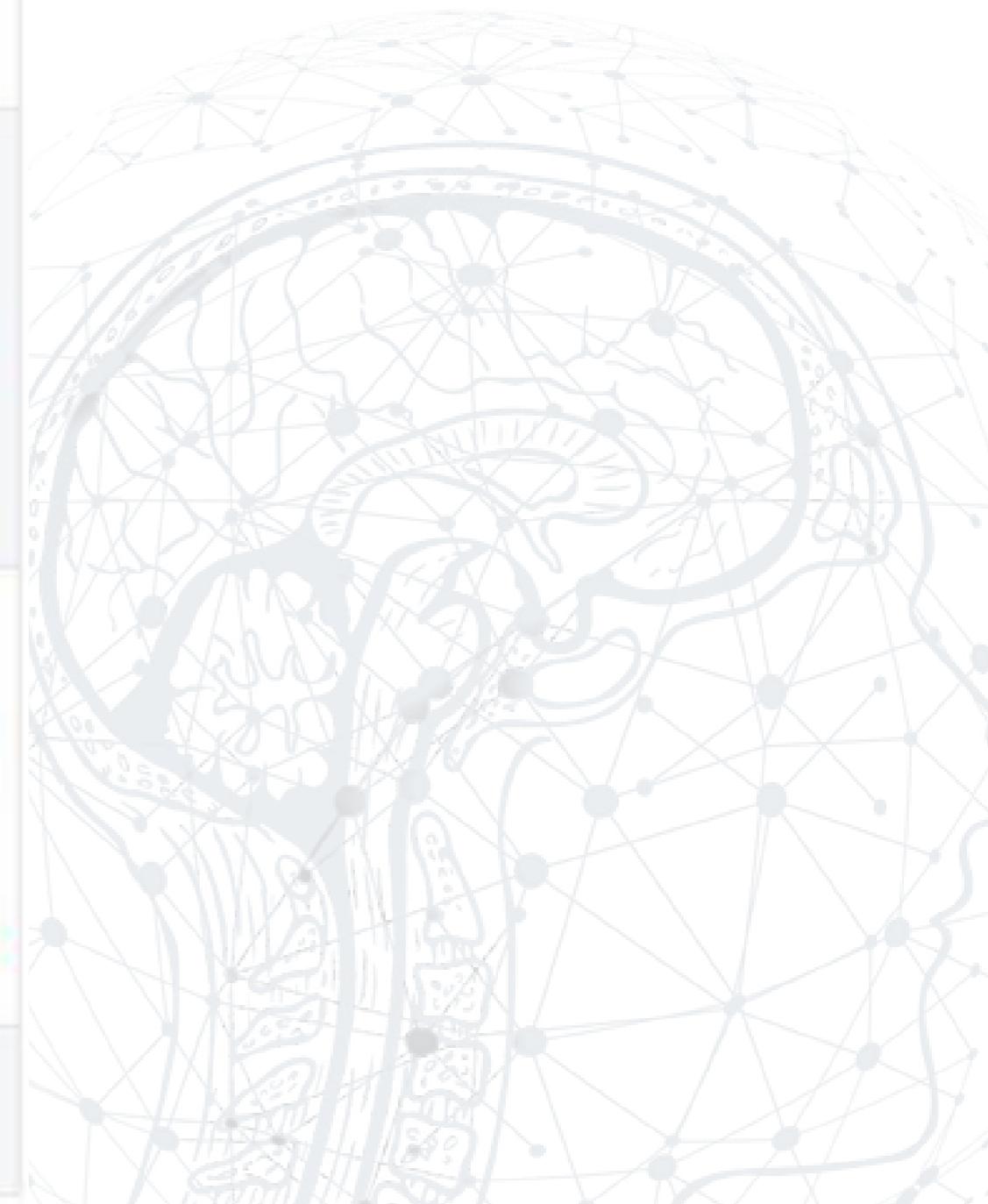




UNIVERSITAS  
INDONESIA  
*Veritas, Probitas, Iustitia*



	Underfitting	Just right	Overfitting
Symptoms	<ul style="list-style-type: none"> <li>• High training error</li> <li>• Training error close to test error</li> <li>• High bias</li> </ul>	<ul style="list-style-type: none"> <li>• Training error slightly lower than test error</li> </ul>	<ul style="list-style-type: none"> <li>• Very low training error</li> <li>• Training error much lower than test error</li> <li>• High variance</li> </ul>
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	<ul style="list-style-type: none"> <li>• Complexify model</li> <li>• Add more features</li> <li>• Train longer</li> </ul>		<ul style="list-style-type: none"> <li>• Perform regularization</li> <li>• Get more data</li> </ul>





UNIVERSITAS  
INDONESIA

*Veritas, Probitas, Justitia*

# Correctness

# Correctness

- True positive:
  - “This message is spam, and we correctly predicted spam.”
- False positive (Type 1 Error):
  - “This message is not spam, but we predicted spam.”
- False negative (Type 2 Error):
  - “This message is spam, but we predicted not spam.” True negative: “This message is not spam, and we correctly predicted not spam.”

	Spam	not Spam
predict “Spam”	True Positive	False Positive
predict “Not Spam”	False Negative	True Negative

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Confusion Matrix

# Classifier metric (for binary classification)

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

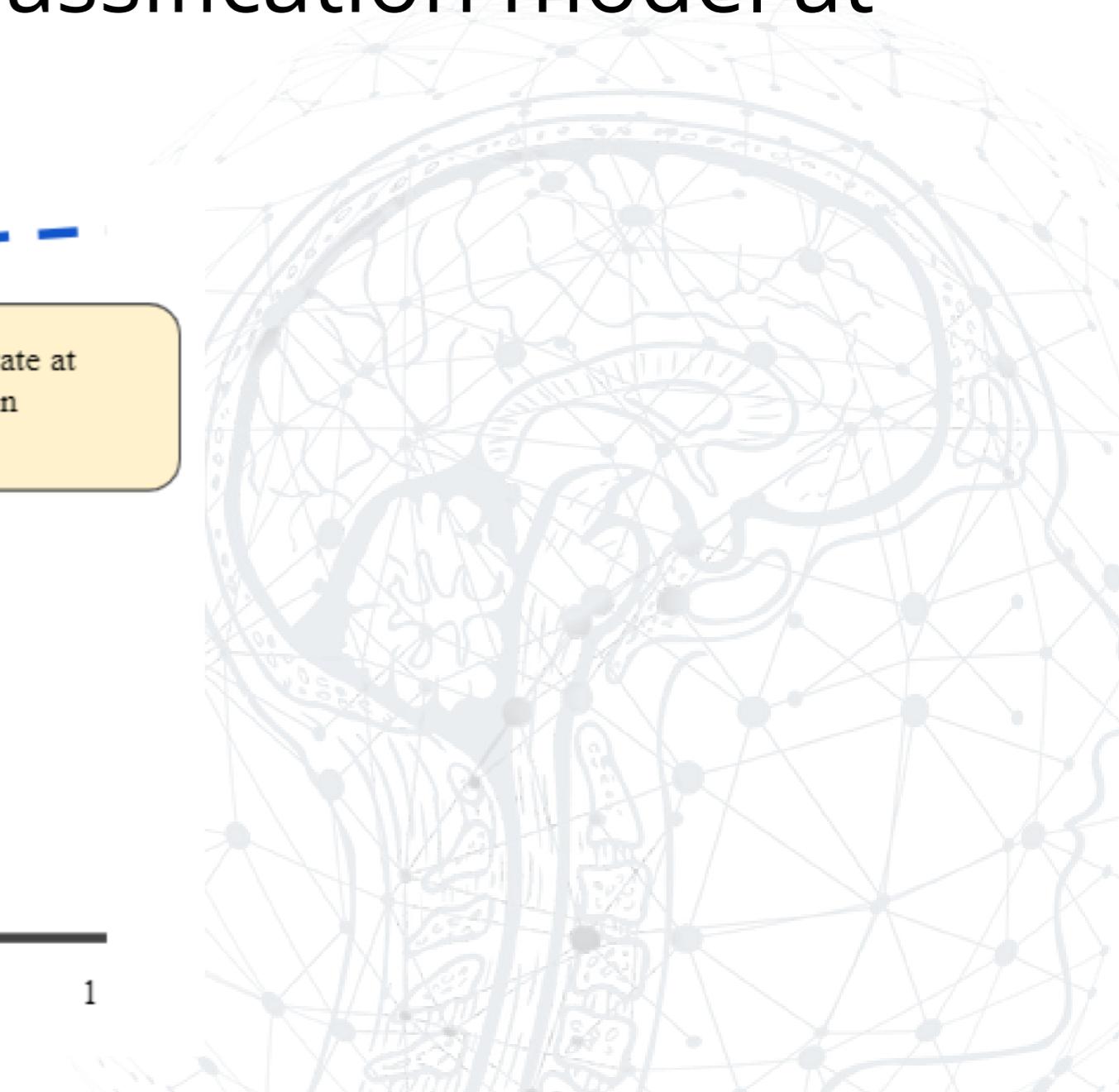
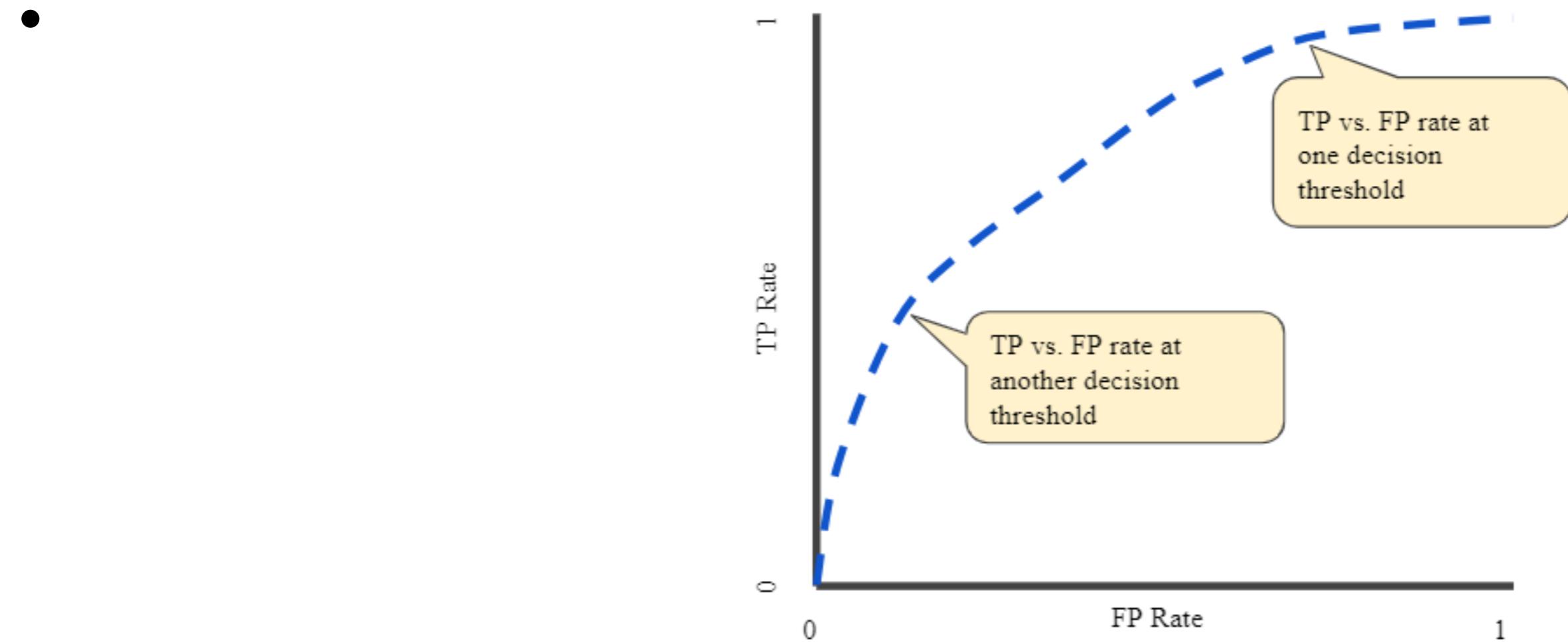
$$accuracy = \frac{TP + TN}{TP + FN + TN + FP}$$

$$specificity = \frac{TN}{TN + FP}$$



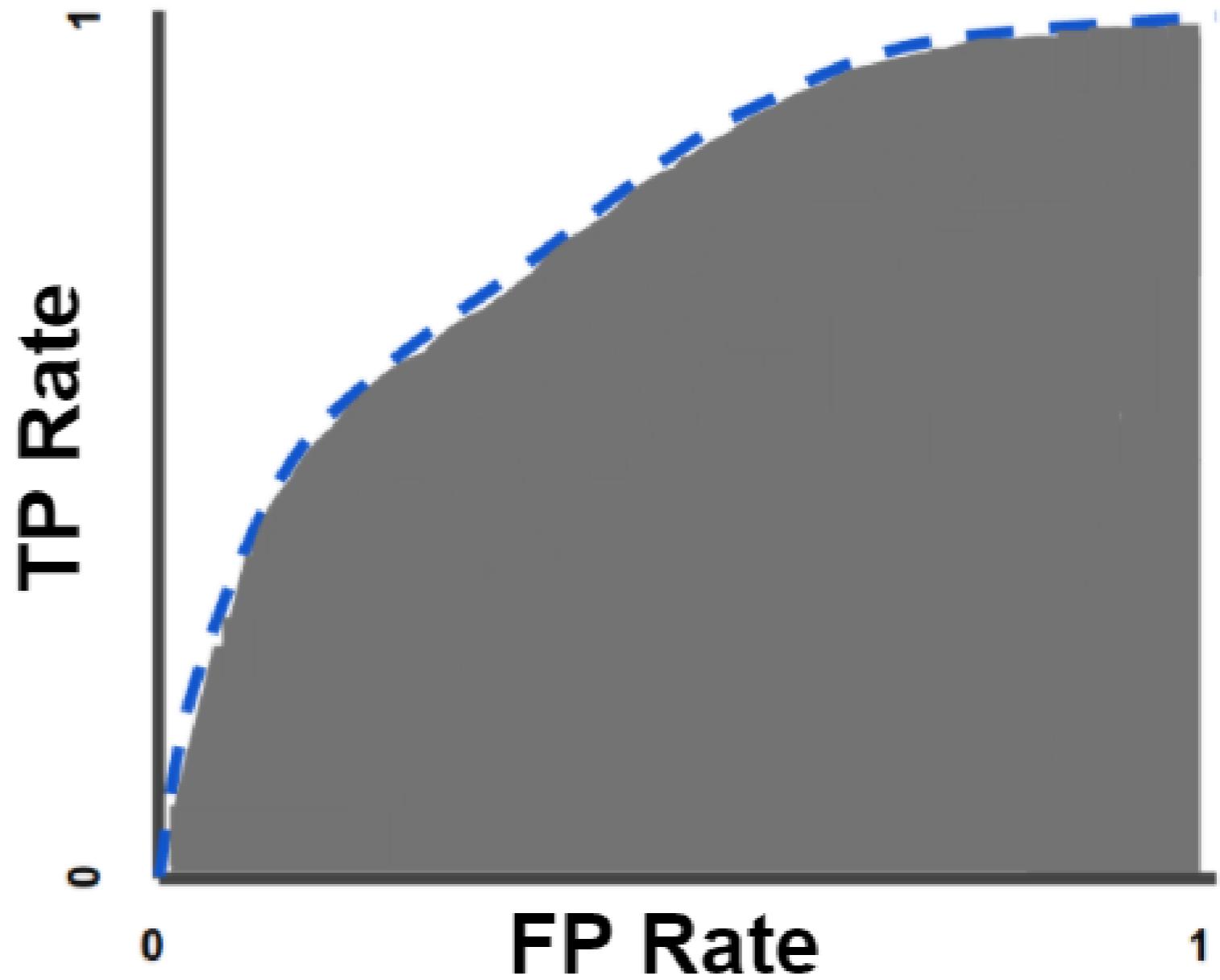
# ROC curve

- An **ROC curve (receiver operating characteristic curve)** is a graph showing the performance of a classification model at all classification thresholds.



# AUC: Area Under the ROC Curve

AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example





**Today's hands on**

**<https://www.kaggle.com/learn/data-cleaning>**



UNIVERSITAS  
INDONESIA

*Veritas, Prudentia, Justitia*

# KECERDASAN BUATAN

3 | REGRESI

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI





UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# Pendahuluan

# Apa itu Regresi?

- Analisis/uji regresi merupakan suatu kajian dari hubungan antara satu variabel, yaitu variabel yang diterangkan (*the explained variable*) dengan satu atau lebih variabel, yaitu variabel yang menerangkan (*the explanatory*).
- Apabila variabel bebasnya **hanya satu**, maka analisis regresinya disebut dengan regresi linear sederhana (**simple linear regression**).
- Apabila variabel bebasnya **lebih dari satu**, maka analisis regresinya dikenal dengan regresi linear berganda (**multiple regression**).
  - Dikatakan berganda karena terdapat beberapa variabel bebas yang mempengaruhi variabel tak bebas
- Hasil dari analisis/uji regresi berupa suatu persamaan regresi.
  - Persamaan regresi ini merupakan suatu fungsi prediksi variable yang mempengaruhi variabel lain

# Simple Linear Regression

- Regresi Linear Sederhana adalah Metode Statistik yang berfungsi untuk menguji sejauh mana hubungan sebab akibat antara Variabel Faktor Penyebab (X) terhadap Variabel Akibatnya (Y).
- X = predictor
- Y = response
- Regresi Linear Sederhana atau sering disingkat dengan SLR (Simple Linear Regression) juga merupakan salah satu Metode Statistik yang dipergunakan dalam produksi untuk melakukan peramalan ataupun prediksi tentang karakteristik kualitas maupun kuantitas.



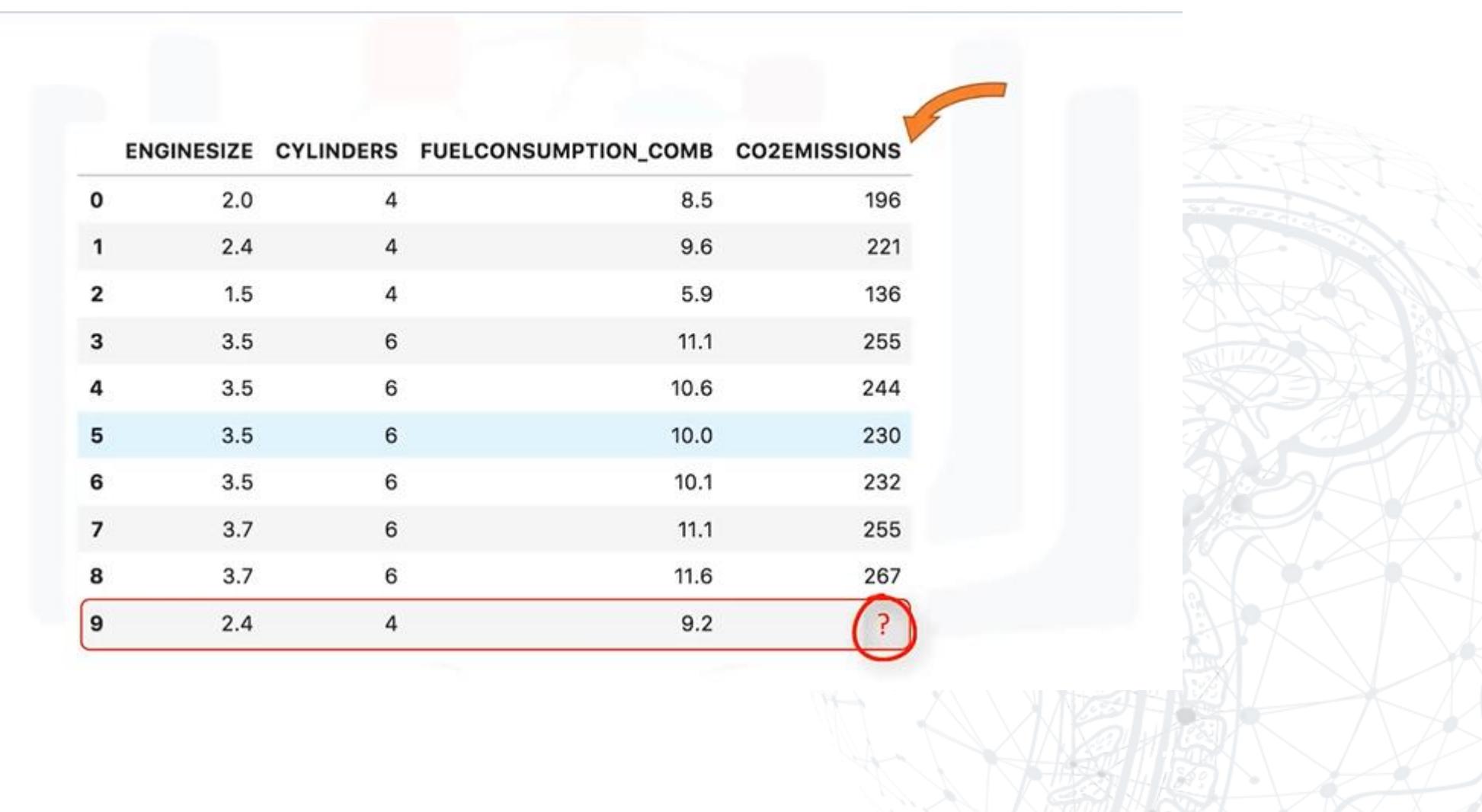
# Contoh Penggunaan

- Contoh penggunaan analisis Regresi Linear Sederhana dalam kegiatan produksi, antara lain:
  - Hubungan antara lamanya kerusakan mesin dengan kualitas produk yang dihasilkan
  - Hubungan jumlah pekerja dengan output yang diproduksi
  - Hubungan antara suhu ruangan dengan cacat produksi yang dihasilkan.

# Metode Regresi Linear Sederhana

diambil dari Machine Learning with Python cognitiveclass.ai

# Regresi Untuk Melakukan Prediksi Pada Data yang Kontinyu



	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

# Misal: 1 independent variable X untuk memprediksi dependent variable Y

X: Independent variable

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

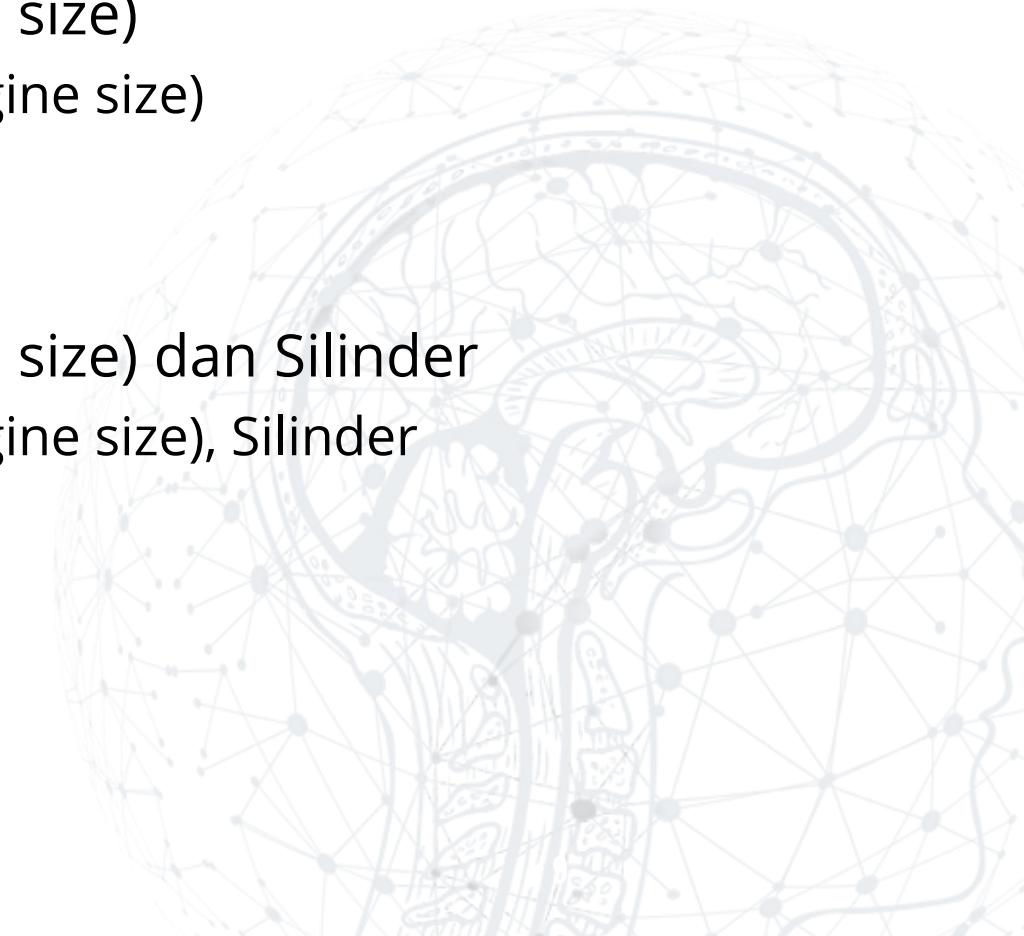
Y: Dependent variable

Continuous Values



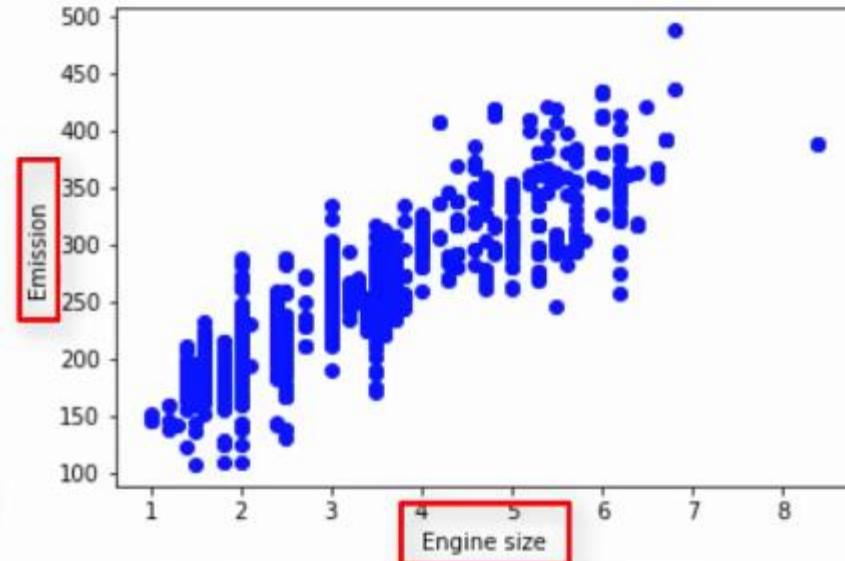
# Topologi Regresi Linear

- Regresi Linear Sederhana :
  - Prediksi emisi Co2 VS Ukuran mesin (engine size)
    - Variabel Independen (X): Ukuran mesin (engine size)
    - Variabel Dependen (Y): Emisi Co2
- Regresi Linear Berganda:
  - Prediksi emisi Co2 VS Ukuran mesin (engine size) dan Silinder
    - Variabel Independen (X): Ukuran mesin (engine size), Silinder
    - Variabel Dependen (Y): Emisi Co2



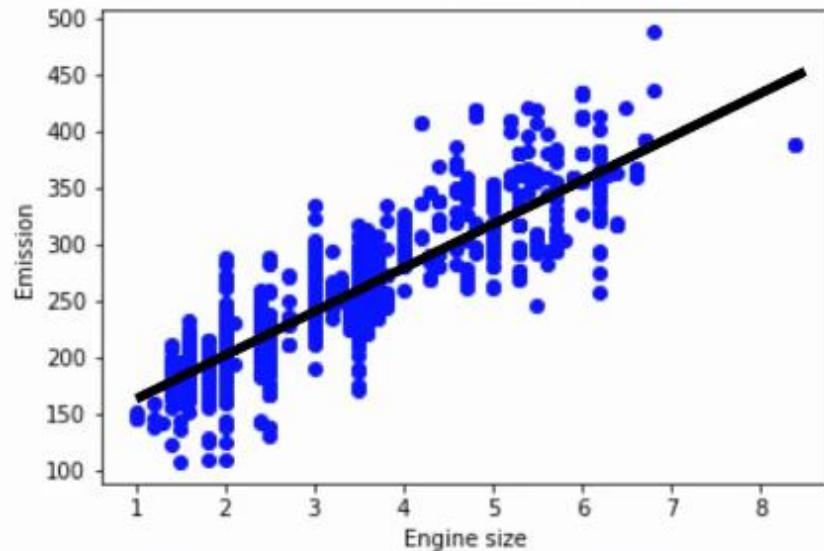
# Bagaimana Regresi Linear Bekerja?

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?



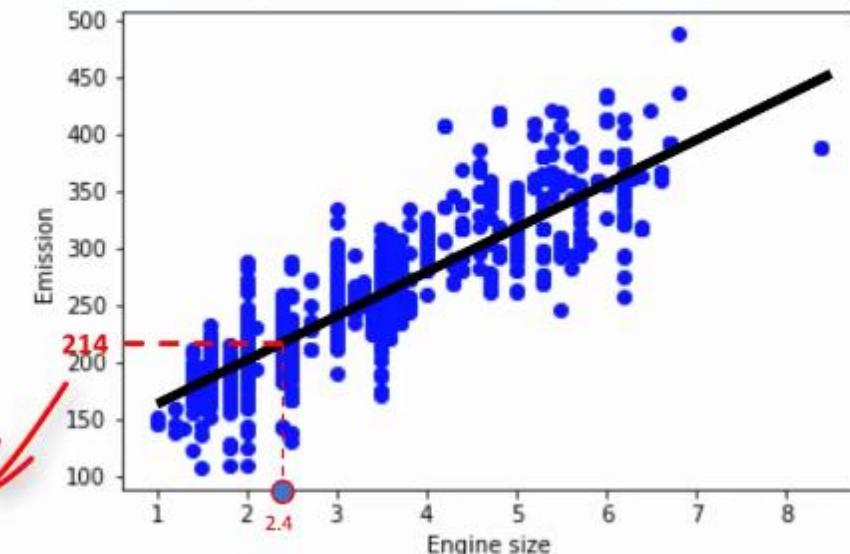
# Bagaimana Regresi Linear Bekerja?

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

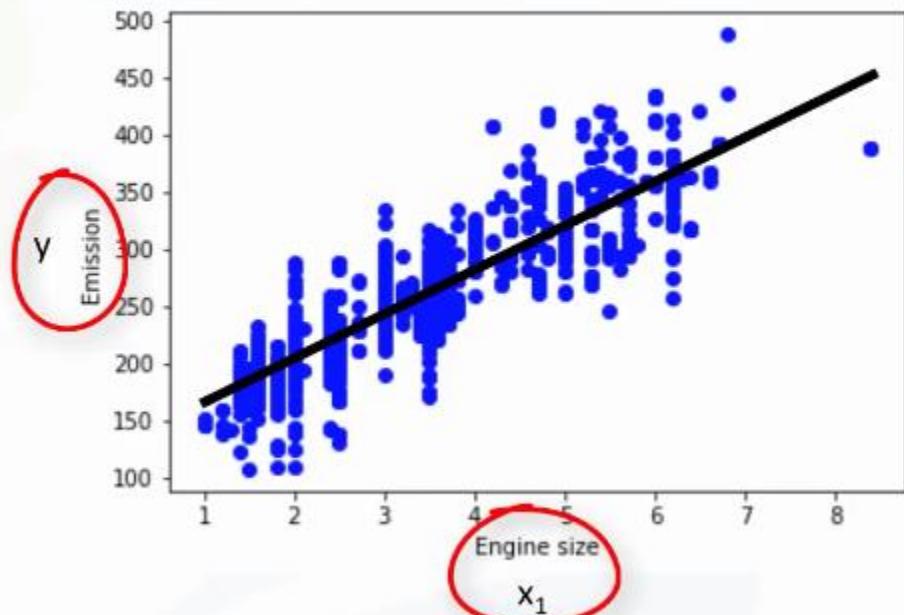


# Bagaimana Regresi Linear Bekerja?

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

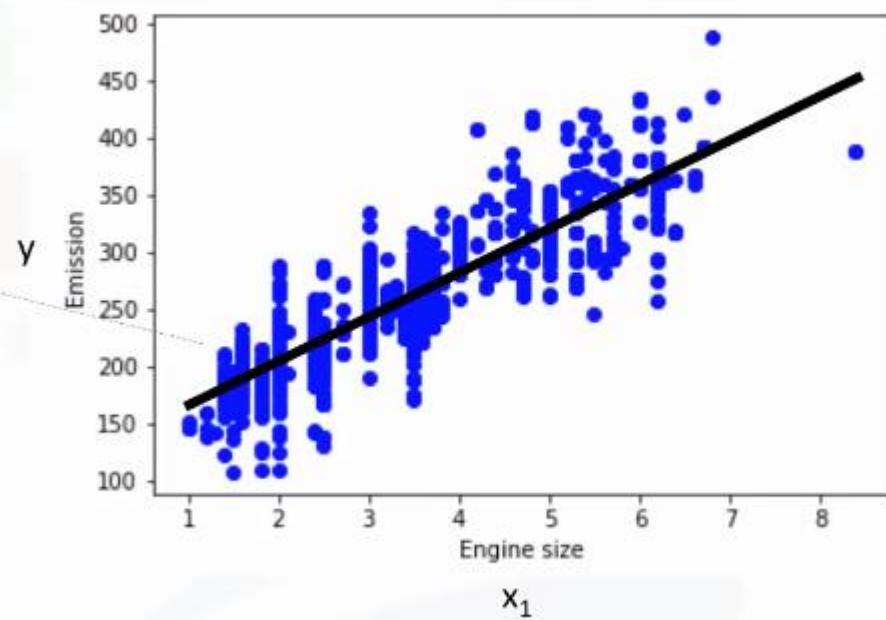


# Representasi Model Regresi Linear

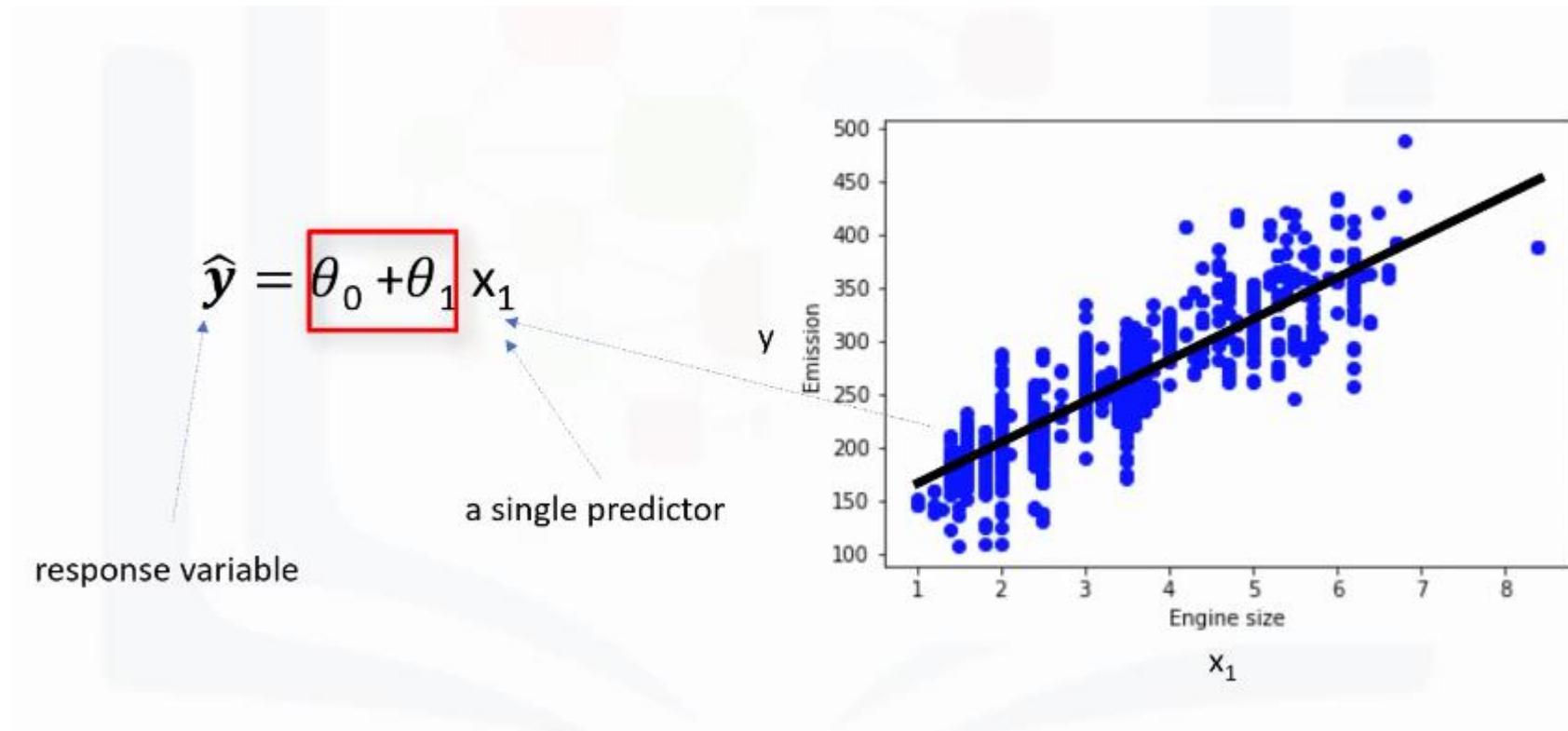


# Representasi Model Regresi Linear

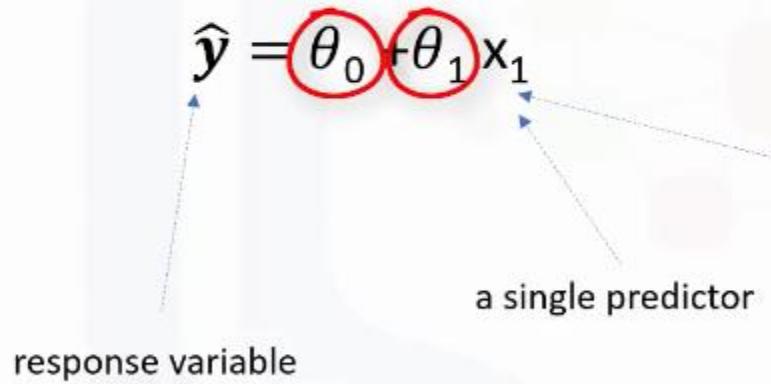
$$\hat{y} = \theta_0 + \theta_1 x_1$$



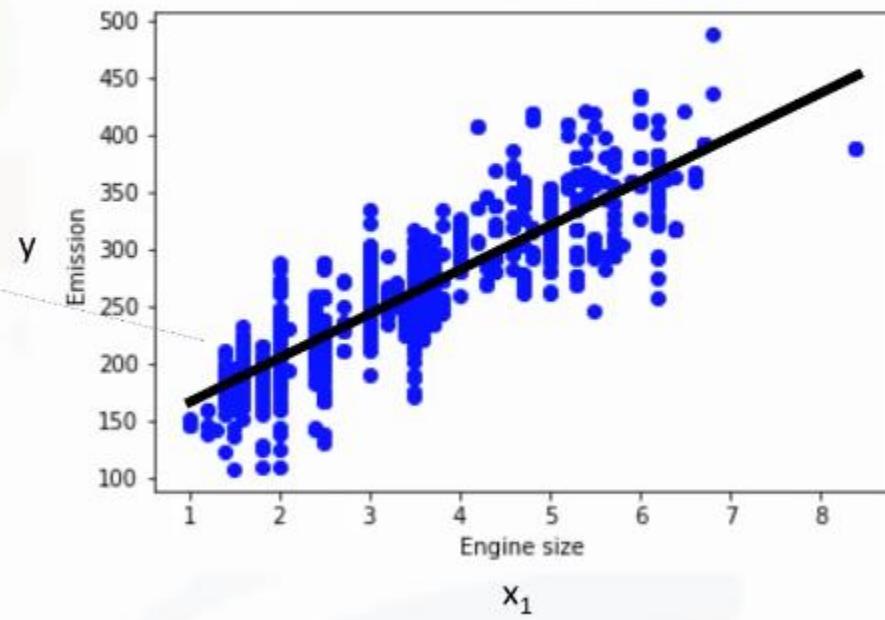
# Representasi Model Regresi Linear



# Representasi Model Regresi Linear


$$\hat{y} = \theta_0 + \theta_1 x_1$$

A diagram illustrating the linear regression equation  $\hat{y} = \theta_0 + \theta_1 x_1$ . A vertical dashed line represents the response variable  $y$ , and a horizontal dashed line represents the predictor variable  $x_1$ . The equation is shown with terms  $\theta_0$  and  $\theta_1 x_1$  circled in red. A blue arrow points from the term  $\theta_1 x_1$  to the label "a single predictor".

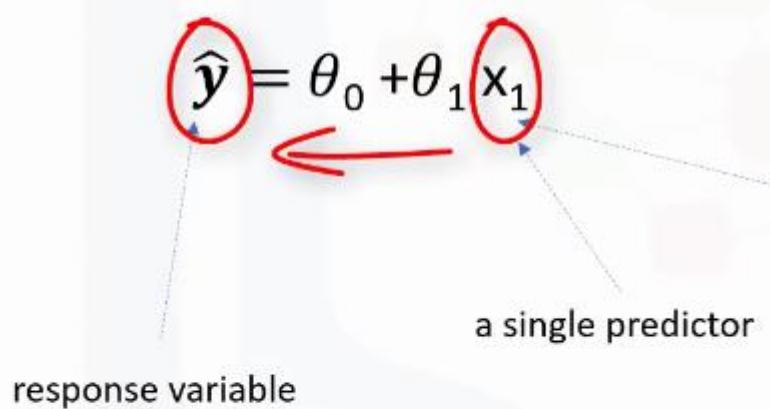


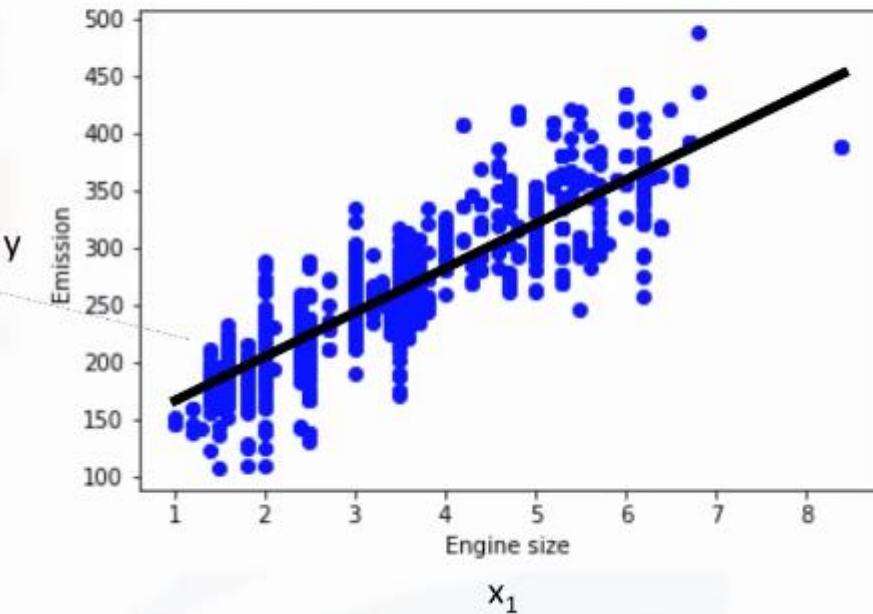
# Representasi Model Regresi Linear

$$\hat{y} = \theta_0 + \theta_1 x_1$$

response variable

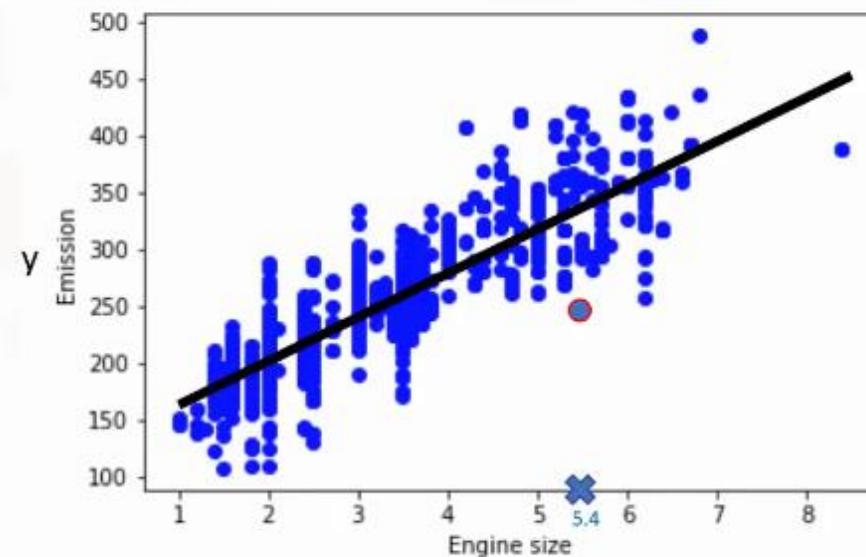
a single predictor







# Bagaimana Mencari Model yang Fit



# Bagaimana Mencari Model yang Fit

$x_1 = 2.4$  independent variable

$y = 250$  actual Co2 emission of  $x_1$

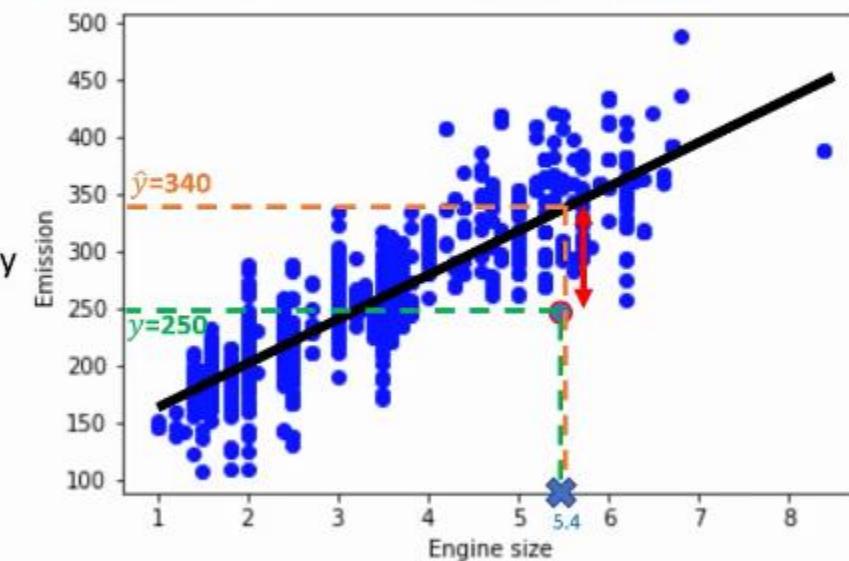
$$\hat{y} = \theta_0 + \theta_1 x_1$$

$\hat{y} = 340$  the predicted emission of  $x_1$

Error =  $y - \hat{y}$

$$= 250 - 340$$

$$= -90$$



# Bagaimana Mencari Model yang Fit

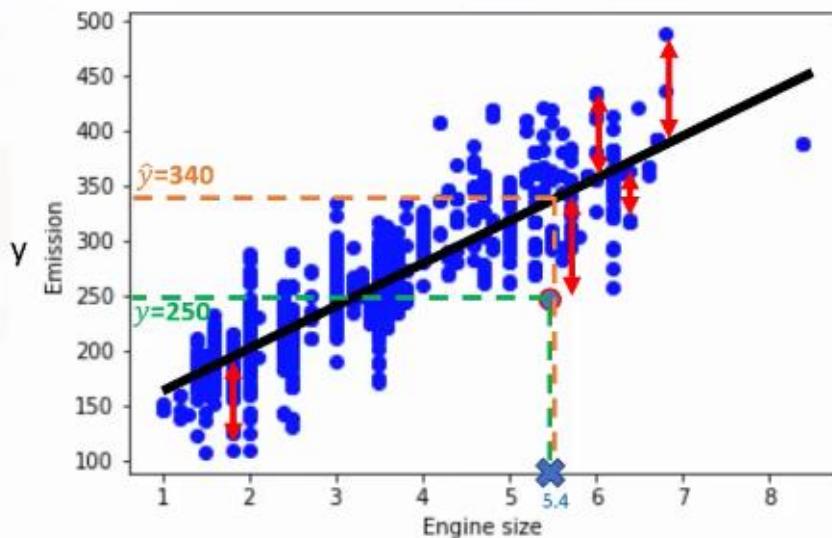
$x_1 = 2.4$  independent variable  
 $y = 250$  actual Co2 emission of  $x_1$

$\hat{y} = \theta_0 + \theta_1 x_1$   
 $\hat{y} = 340$  the predicted emission of  $x_1$

$$\begin{aligned}\text{Error} &= y - \hat{y} \\ &= 250 - 340 \\ &= -90\end{aligned}$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

Mean Squared Error



# Bagaimana Mencari Model yang Fit

$x_1 = 2.4$  independent variable

$y = 250$  actual Co2 emission of  $x_1$

$$\hat{y} = \theta_0 + \theta_1 x_1$$

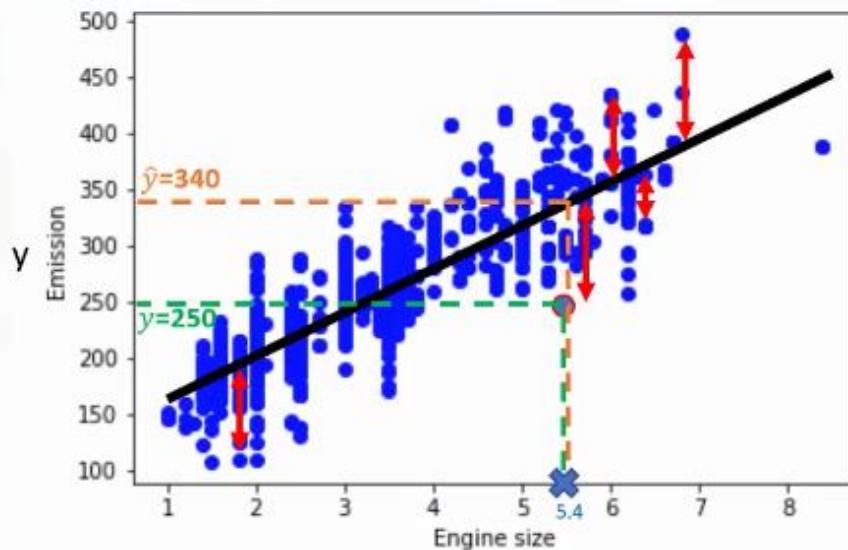
$\hat{y} = 340$  the predicted emission of  $x_1$

$$\text{Error} = y - \hat{y}$$

$$= 250 - 340$$

$$= -90$$

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$



# Estimasi Parameter-Parameter

$$\hat{y} = \theta_0 + \theta_1 x_1$$

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	X <sub>1</sub>	3.5	10.6	y
5	3.5	6	10.0	244
6	3.5	6	10.1	230
7	3.7	6	11.1	232
8	3.7	6	11.6	255
				267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4 $X_1$	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	X <sub>1</sub>	3.5	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_1 = \frac{(2.0 - 3.34)(196 - 256) + (2.4 - 3.34)(221 - 256) + \dots}{(2.0 - 3.34)^2 + (2.4 - 3.34)^2 + \dots}$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	X <sub>1</sub>	3.5	6	10.6
5	3.5	6	10.0	244
6	3.5	6	10.1	230
7	3.7	6	11.1	232
8	3.7	6	11.6	255
		y		267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_1 = \frac{(2.0 - 3.34)(196 - 256) + (2.4 - 3.34)(221 - 256) + \dots}{(2.0 - 3.34)^2 + (2.4 - 3.34)^2 + \dots}$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4 $X_1$	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_1 = \frac{(2.0 - 3.34)(196 - 256) + (2.4 - 3.34)(221 - 256) + \dots}{(2.0 - 3.34)^2 + (2.4 - 3.34)^2 + \dots}$$

$$\theta_1 = 39$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	X <sub>1</sub>	3.5	10.6	y
5	3.5	6	10.0	244
6	3.5	6	10.1	230
7	3.7	6	11.1	232
8	3.7	6	11.6	255
				267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_1 = \frac{(2.0 - 3.34)(196 - 256) + (2.4 - 3.34)(221 - 256) + \dots}{(2.0 - 3.34)^2 + (2.4 - 3.34)^2 + \dots}$$

$$\theta_1 = 39$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

$$\theta_0 = 256 - 39 * 3.34$$

$$\boxed{\theta_0 = 125.74}$$

# Estimasi Parameter-Parameter

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	X <sub>1</sub>	6	10.6	y
5	3.5	6	10.0	244
6	3.5	6	10.1	230
7	3.7	6	11.1	232
8	3.7	6	11.6	255
				267

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$\theta_1 = \frac{\sum_{i=1}^s (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^s (x_i - \bar{x})^2}$$

$$\bar{x} = (2.0 + 2.4 + 1.5 + \dots) / 9 = 3.34$$

$$\bar{y} = (196 + 221 + 136 + \dots) / 9 = 256$$

$$\theta_1 = \frac{(2.0 - 3.34)(196 - 256) + (2.4 - 3.34)(221 - 256) + \dots}{(2.0 - 3.34)^2 + (2.4 - 3.34)^2 + \dots}$$

$$\theta_1 = 39$$

$$\theta_0 = \bar{y} - \theta_1 \bar{x}$$

$$\theta_0 = 256 - 39 * 3.34$$

$$\theta_0 = 125.74$$

$$\boxed{\hat{y} = 125.74 + 39x_1}$$

# Prediksi dengan Model Garis (*line model*)

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

# Prediksi dengan Model Garis (*line model*)

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$Co2Emission = \theta_0 + \theta_1 EngineSize$$

$$Co2Emission = 125 + 39 EngineSize$$

# Prediksi dengan Model Garis (*line model*)

	ENGINESIZE	CYLINDERS	FUELCONSUMPTION_COMB	CO2EMISSIONS
0	2.0	4	8.5	196
1	2.4	4	9.6	221
2	1.5	4	5.9	136
3	3.5	6	11.1	255
4	3.5	6	10.6	244
5	3.5	6	10.0	230
6	3.5	6	10.1	232
7	3.7	6	11.1	255
8	3.7	6	11.6	267
9	2.4	4	9.2	?

$$\hat{y} = \theta_0 + \theta_1 x_1$$

$$Co2Emission = \theta_0 + \theta_1 EngineSize$$

$$Co2Emission = 125 + 39 \times EngineSize$$

$$Co2Emission = 125 + 39 \times 2.4$$

$$Co2Emission = 218.6$$





UNIVERSITAS  
INDONESIA

*Veritas, Prolisia, Nostria*

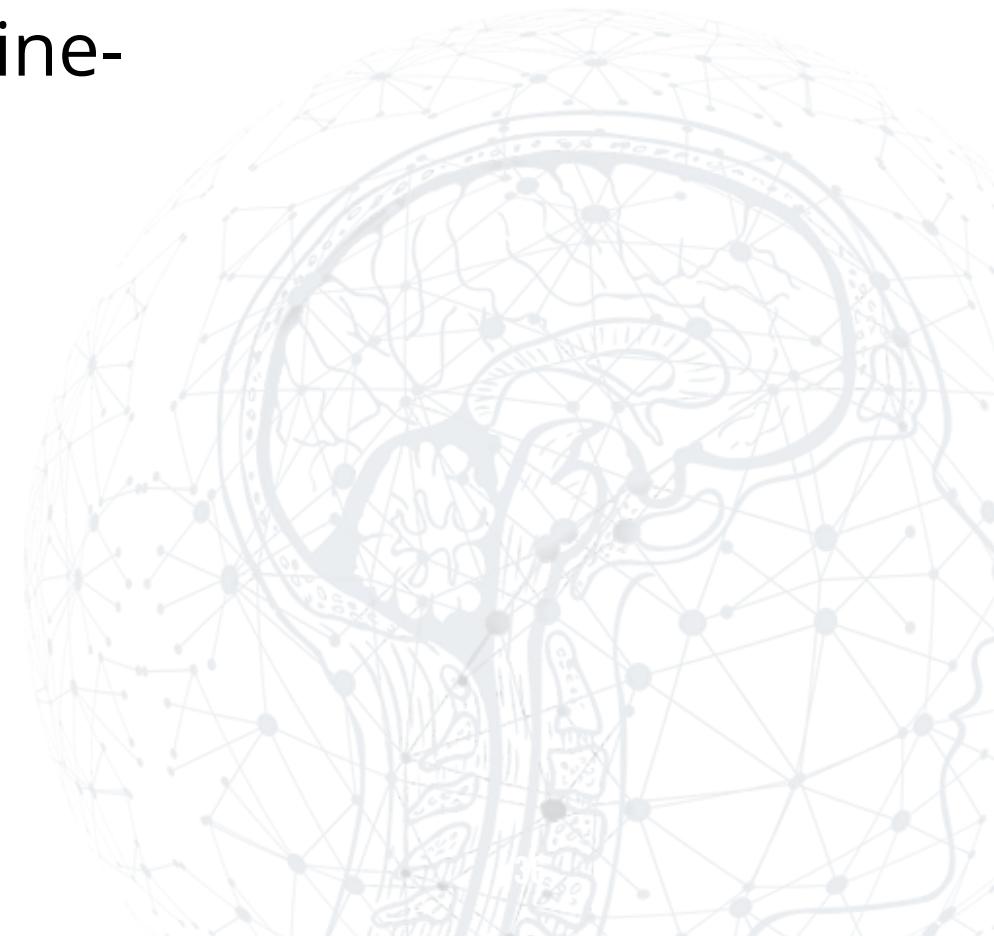
---

Metode tadi adalah metode statistika.

Bagaimana metode machine learning bekerja?



- The following slides are taken from:
- <https://www.coursera.org/learn/machine-learning/home/welcome>
- All credit to Prof. Andrew Ng





UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

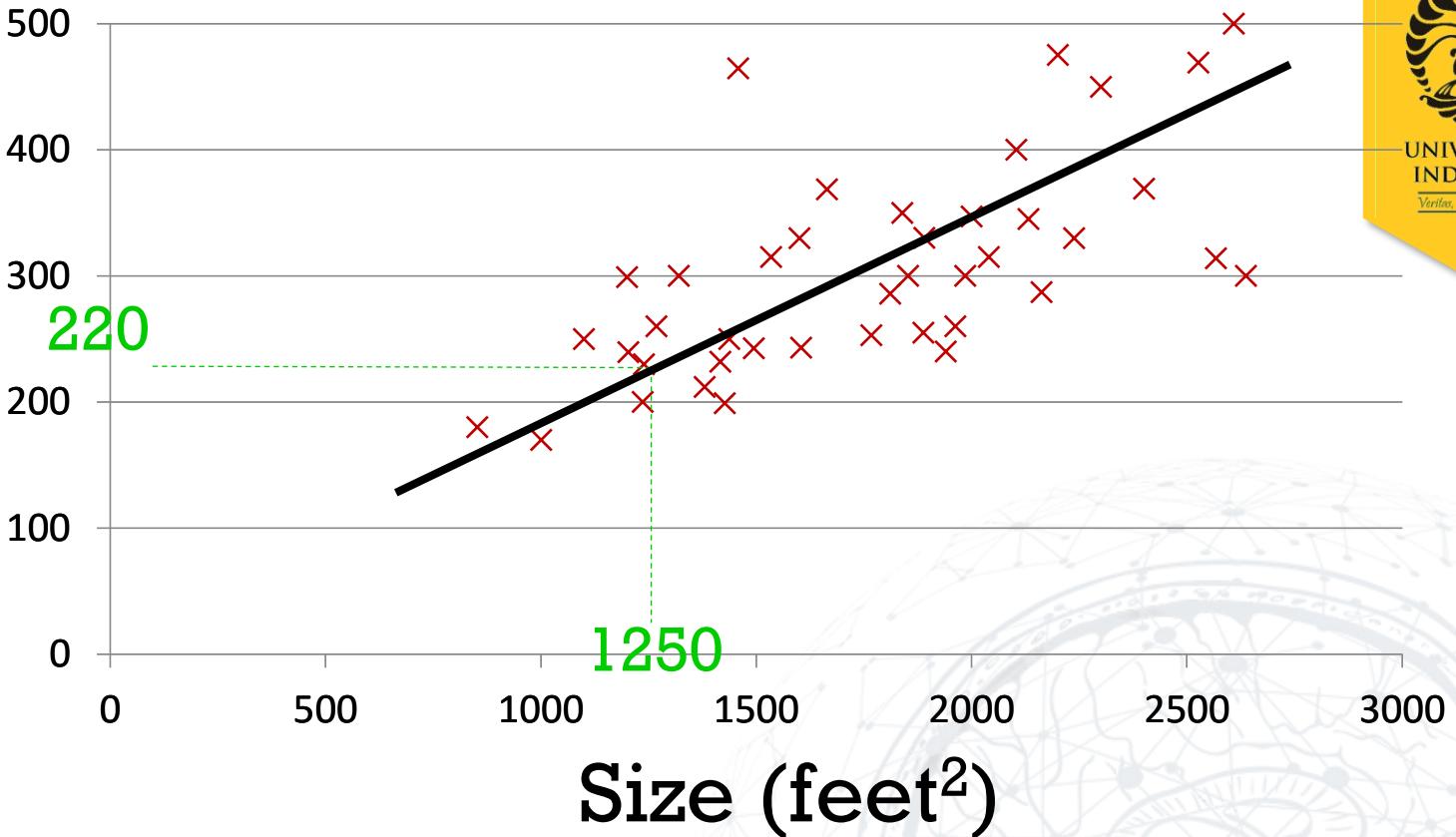
# Regression Model

# Housing Prices (Portland, OR)



UNIVERSITAS  
INDONESIA  
*Veritas, Prudentia, Justitia*

Price  
(in 1000s  
of dollars)



## Supervised Learning

Given the “right answer”  
for each example in the  
data.

## Regression Problem

Predict real-valued output  
**Classification : Discrete-valued  
output**

# Training set of housing prices (Portland, OR)

	Size in feet <sup>2</sup> (x)	Price (\$) in 1000's (y)
	2104	460
	1416	232
	1534	315
	852	178
	...	...

Notation:

**m** = Number of training examples

**x**'s = “input” variable / features

**y**'s = “output” variable / “target” variable

(**x**, **y**) – one training example

( $\mathbf{x}^{(i)}$ ,  $\mathbf{y}^{(i)}$ ) – ith trainingg example

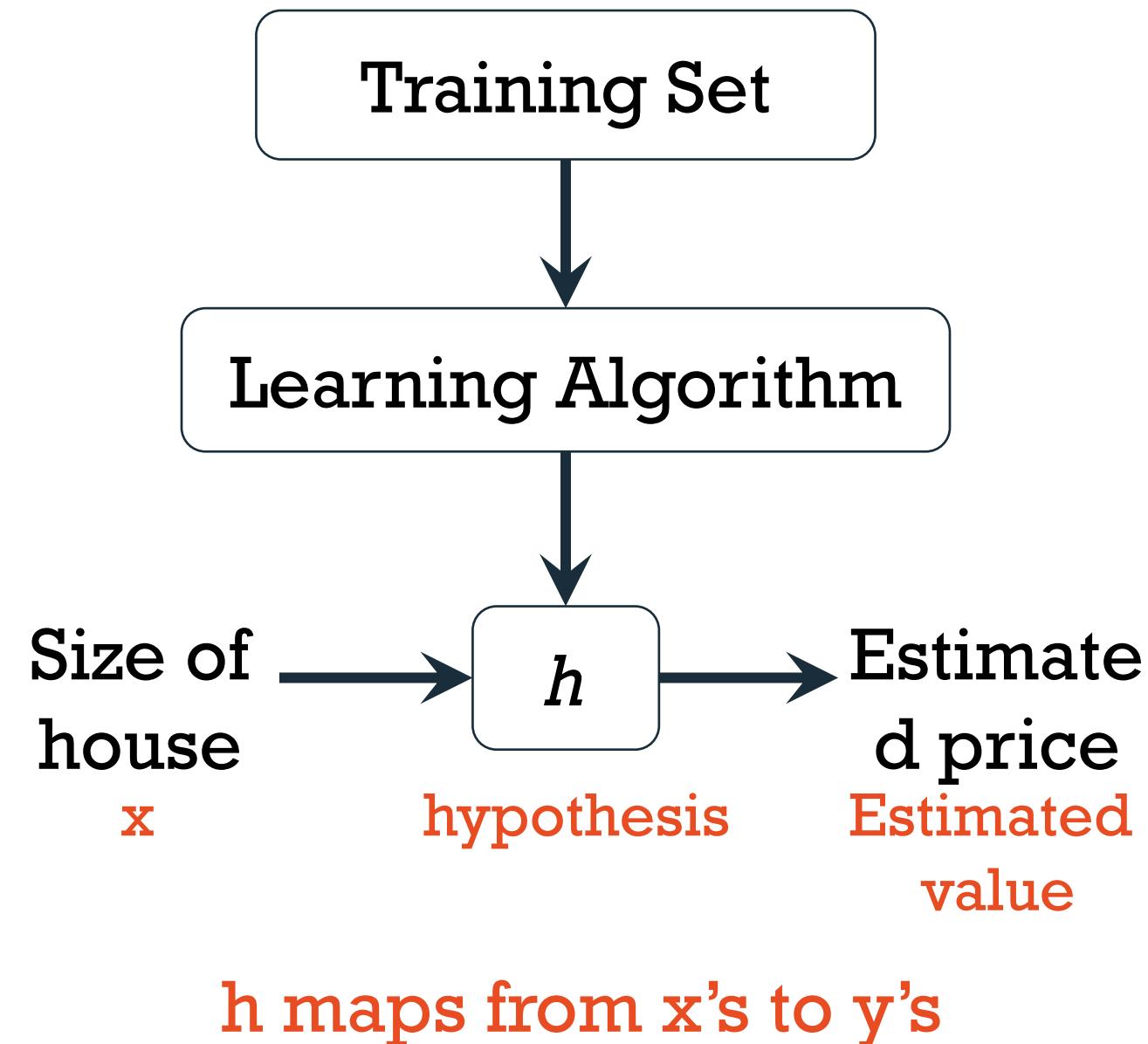
$$\mathbf{x}^{(1)} = 2104$$

$$\mathbf{x}^{(2)} = 1416$$

$$\mathbf{y}^{(1)} = 460$$

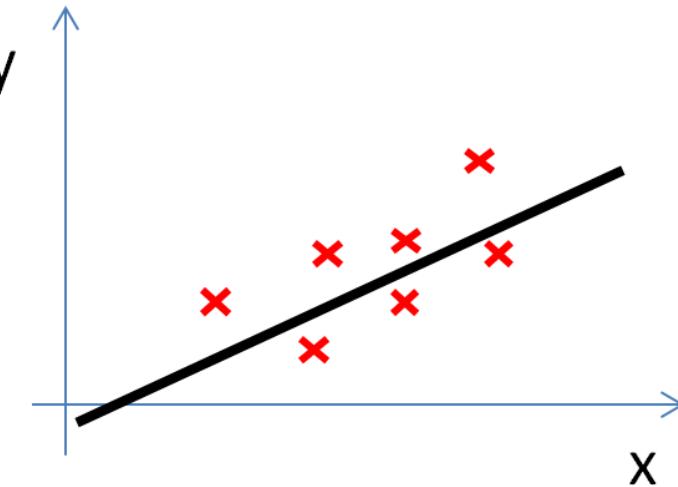


Veritas, Prudentia, Justitia



## How do we represent $h$ ?

$$h_{\Theta}(x) = \Theta_0 + \Theta_1 x$$



Linear regression with one variable  
Univariate linear regression.

One variable



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# COST FUNCTION

# Training Set

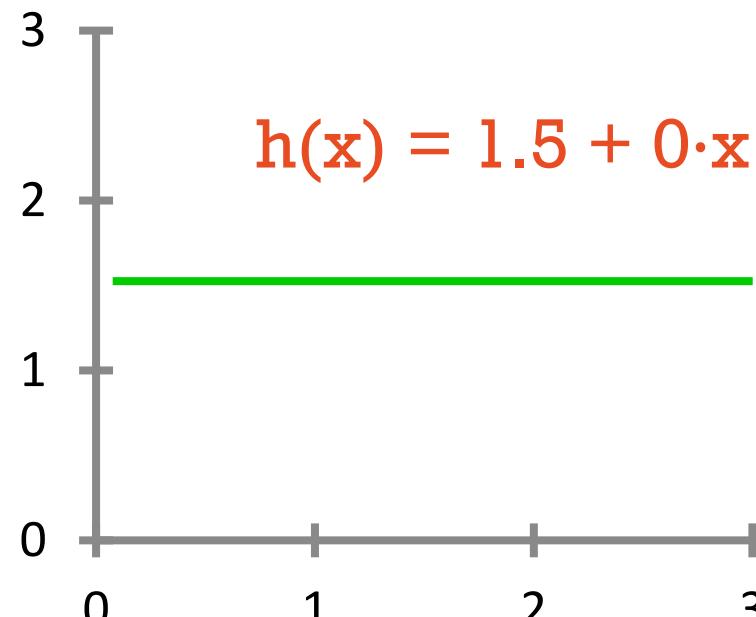
Size in feet <sup>2</sup> (x)	Price (\$) in 1000's (y)
2104	460
1416	232
1534	315
852	178
...	...

Hypothesis:  $h_{\theta}(x) = \theta_0 + \theta_1 x$

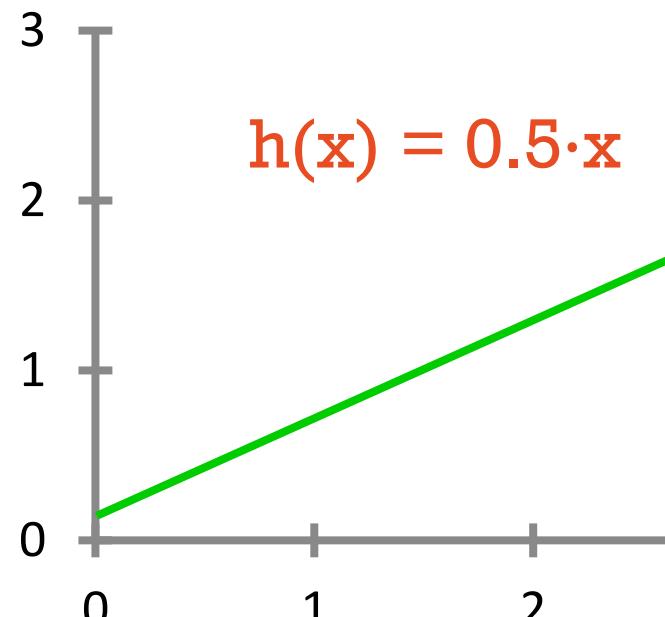
$\theta_i$ 's: Parameters

How to choose  $\theta_i$ 's ?

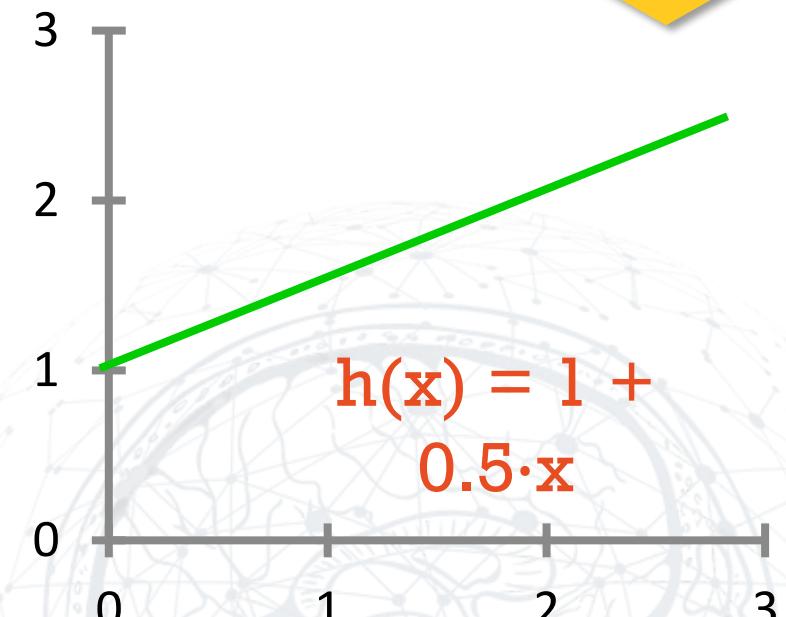
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



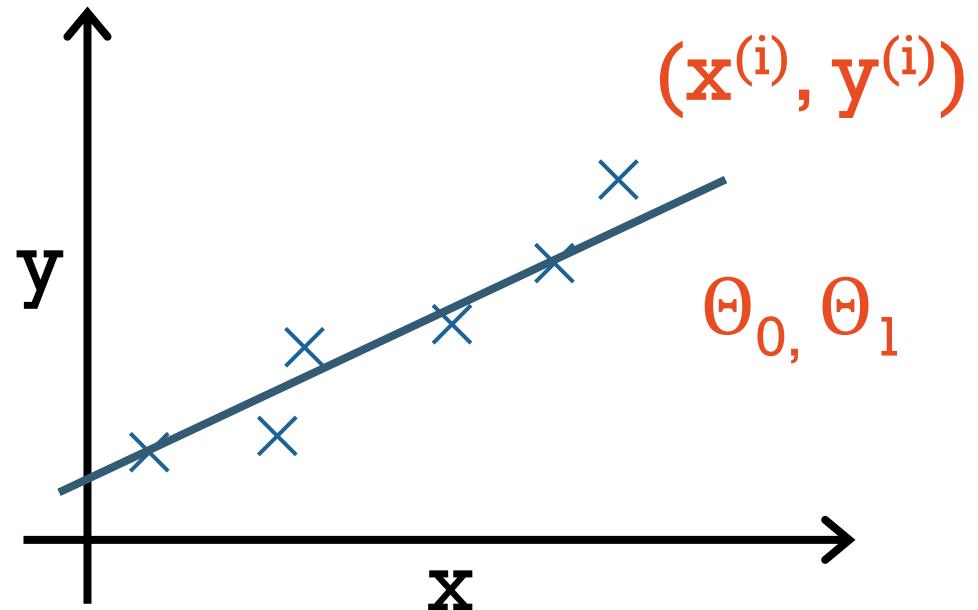
$$\theta_0 = 1.5$$
$$\theta_1 = 0$$



$$\theta_0 = 0$$
$$\theta_1 = 0.5$$



$$\theta_0 = 1$$
$$\theta_1 = 0.5$$



Idea: Choose  $\theta_0, \theta_1$  so that  $h_\theta(x)$  is close to  $y$  for our training examples  $(x, y)$

$$\underset{\theta_0 \theta_1}{\text{minimize}} \frac{1}{2m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2$$

$$h(x) = \theta_0 + \theta_1 x^{(i)}$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})^2$$

Minimize  $J(\theta_0, \theta_1)$  : Cost Function

Squared error function

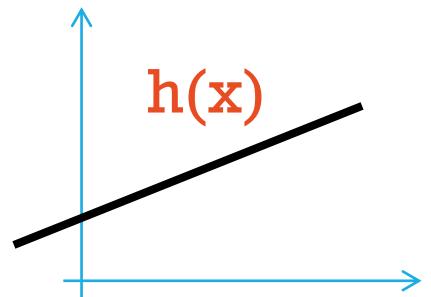
## Simplified

**Hypothesis:**

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

**Parameters:**

$$\theta_0, \theta_1$$



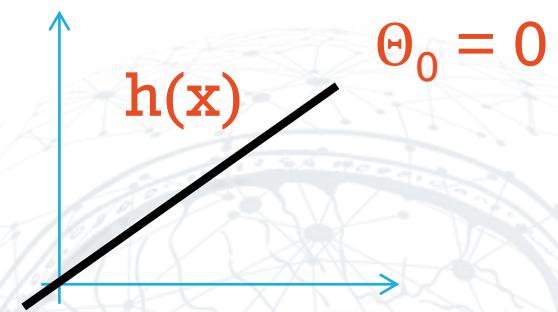
**Cost Function:**

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

**Goal:** minimize  $J(\theta_0, \theta_1)$   
 $\theta_0, \theta_1$

$$h_{\theta}(x) = \theta_1 x$$

$$\theta_1$$

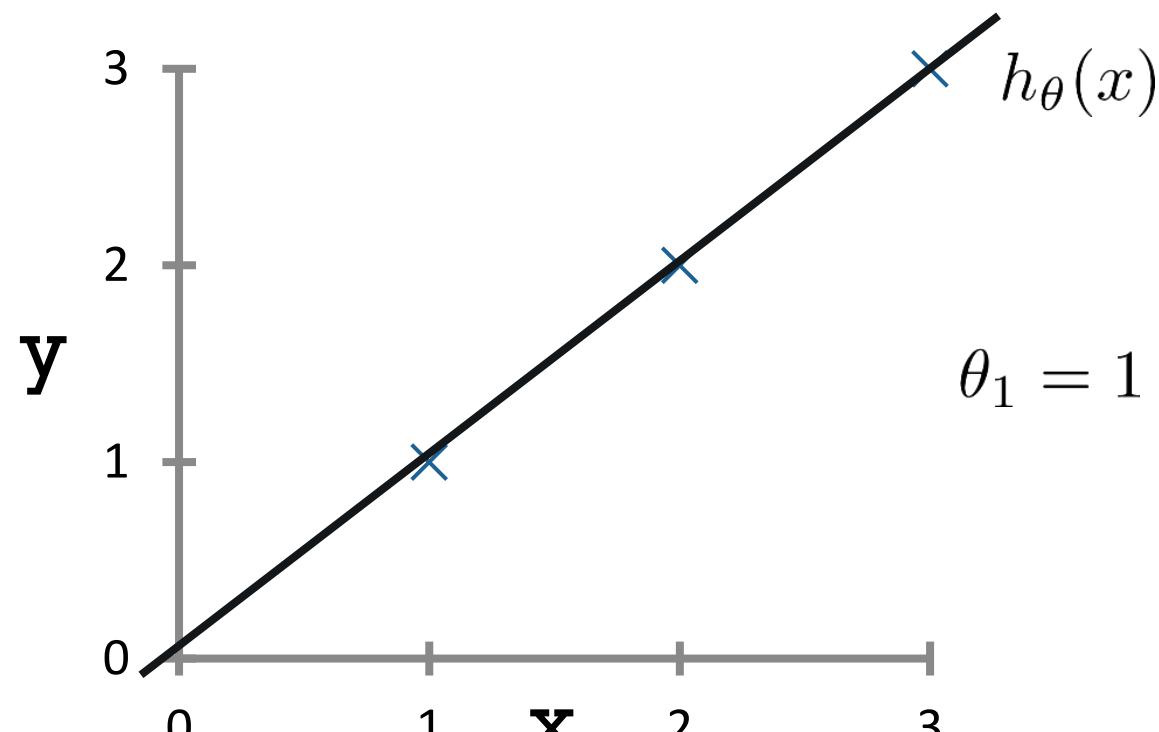


$$J(\theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

minimize  $J(\theta_1)$   
 $\theta_1$

$$h_{\theta}(x) \quad h_{\theta}(x) = \theta_1 x$$

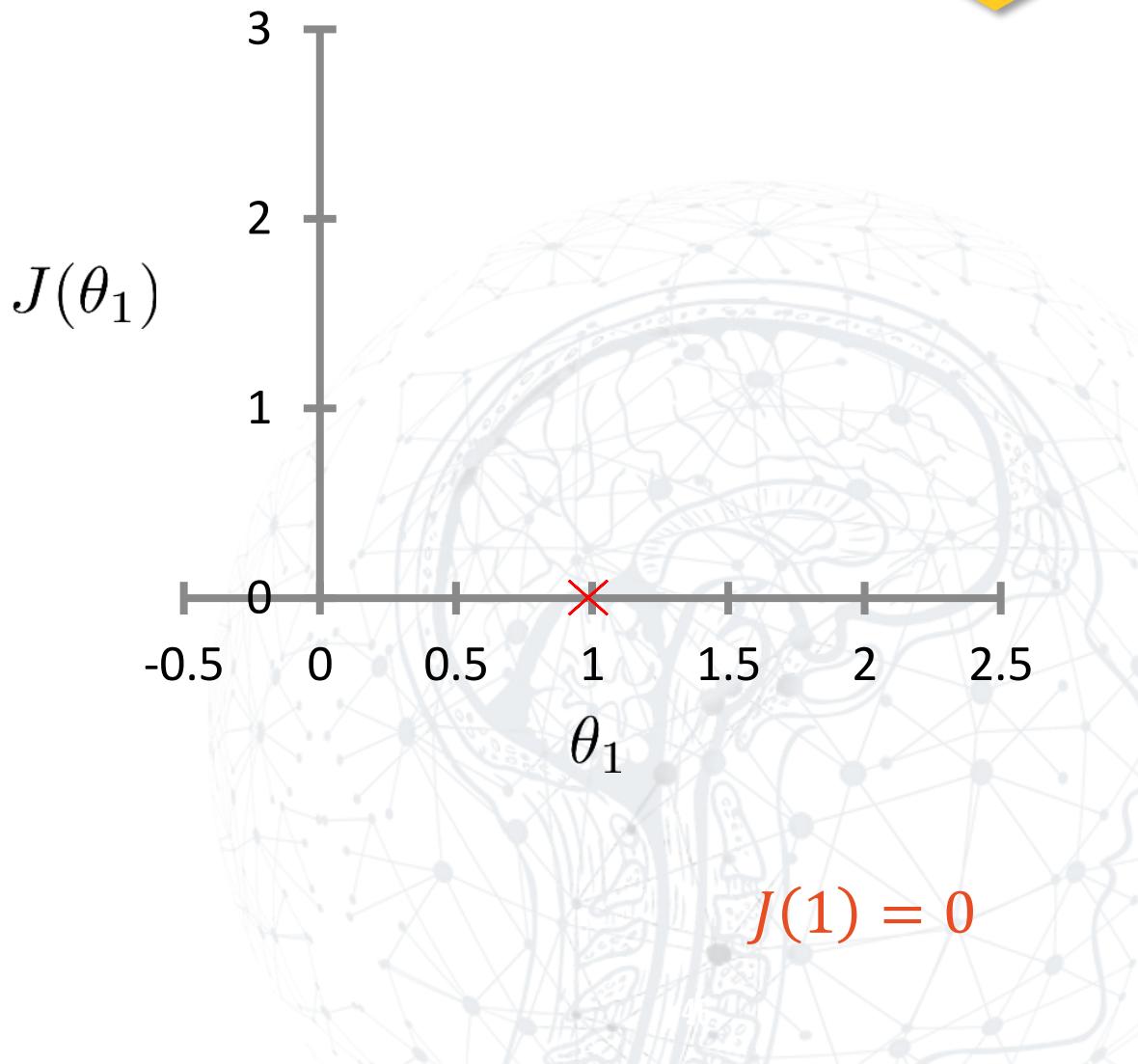
(for fixed  $\theta_1$ , this is a function of  $x$ )



$$\begin{aligned} J(\theta_1) &= \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2 \\ &= \frac{1}{2m} \sum_{i=1}^m (\theta_1 x^{(i)} - y^{(i)})^2 \\ &= \frac{1}{2m} (0^2 + 0^2 + 0^2) \end{aligned}$$

$$J(\theta_1)$$

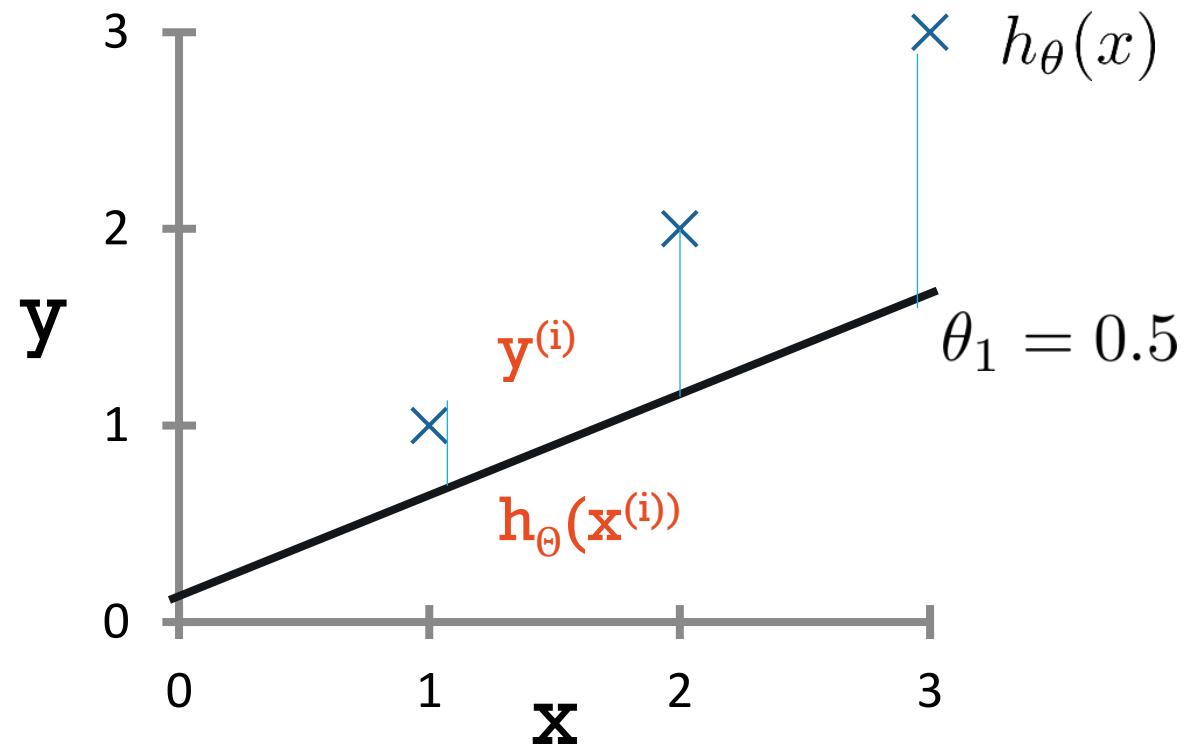
(function of the parameter  $\theta_1$ )



$$J(1) = 0$$

$$h_{\theta}(x)$$

(for fixed  $\theta_1$ , this is a function of x)

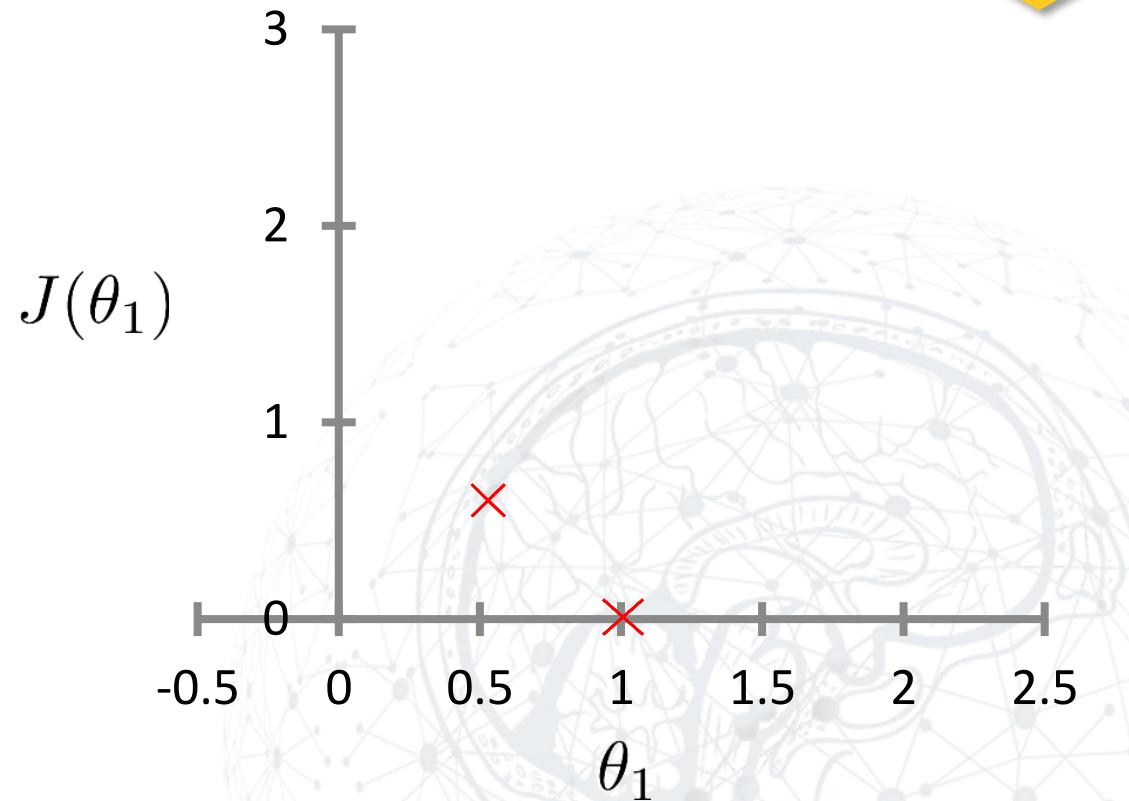


$$J(0.5) = \frac{1}{2 \cdot 3} \sum_{i=1}^3 [(0.5 - 1)^2 + (1 - 2)^2 + (1.5 - 3)^2]$$

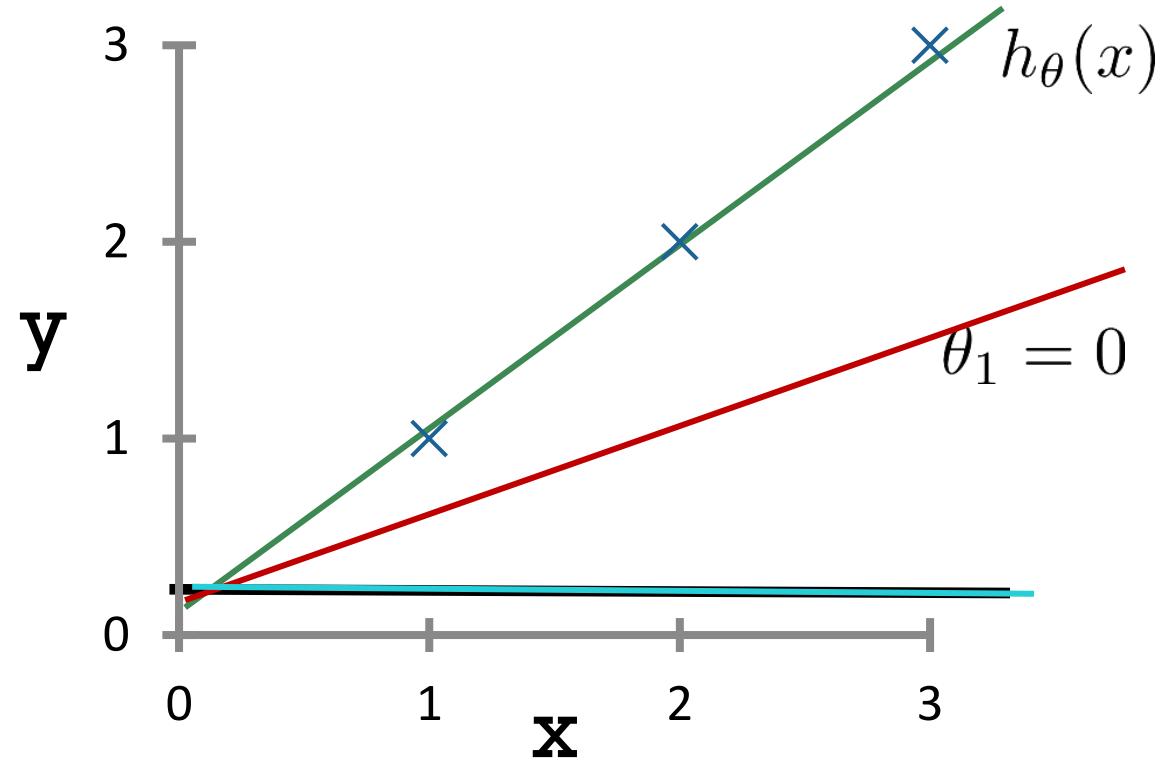
$$= \frac{1}{6} \cdot (3.5) = 0.58$$

$$J(\theta_1)$$

(function of the parameter  $\theta_1$ )

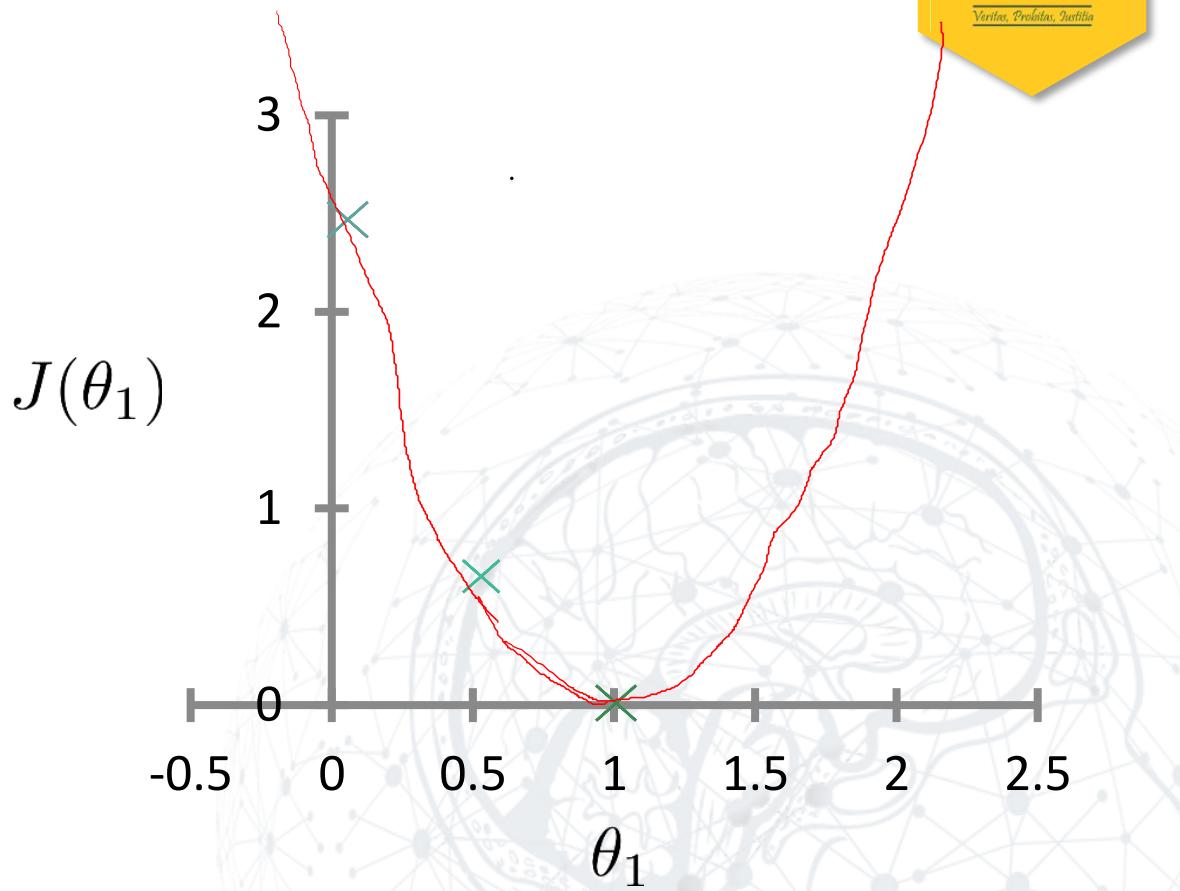


$h_{\theta}(x)$   
(for fixed  $\theta_1$ , this is a function of  $x$ )



$$\begin{aligned} J(0) &= \frac{1}{2 \cdot 3} \sum_{i=1}^3 [1^2 + 2^2 + 3^2] \\ &= \frac{1}{6} \cdot 14 = 2.3 \end{aligned}$$

$J(\theta_1)$   
(function of the parameter  $\theta_1$ )



**Hypothesis:**  $h_{\theta}(x) = \theta_0 + \theta_1 x$

**Parameters:**  $\theta_0, \theta_1$

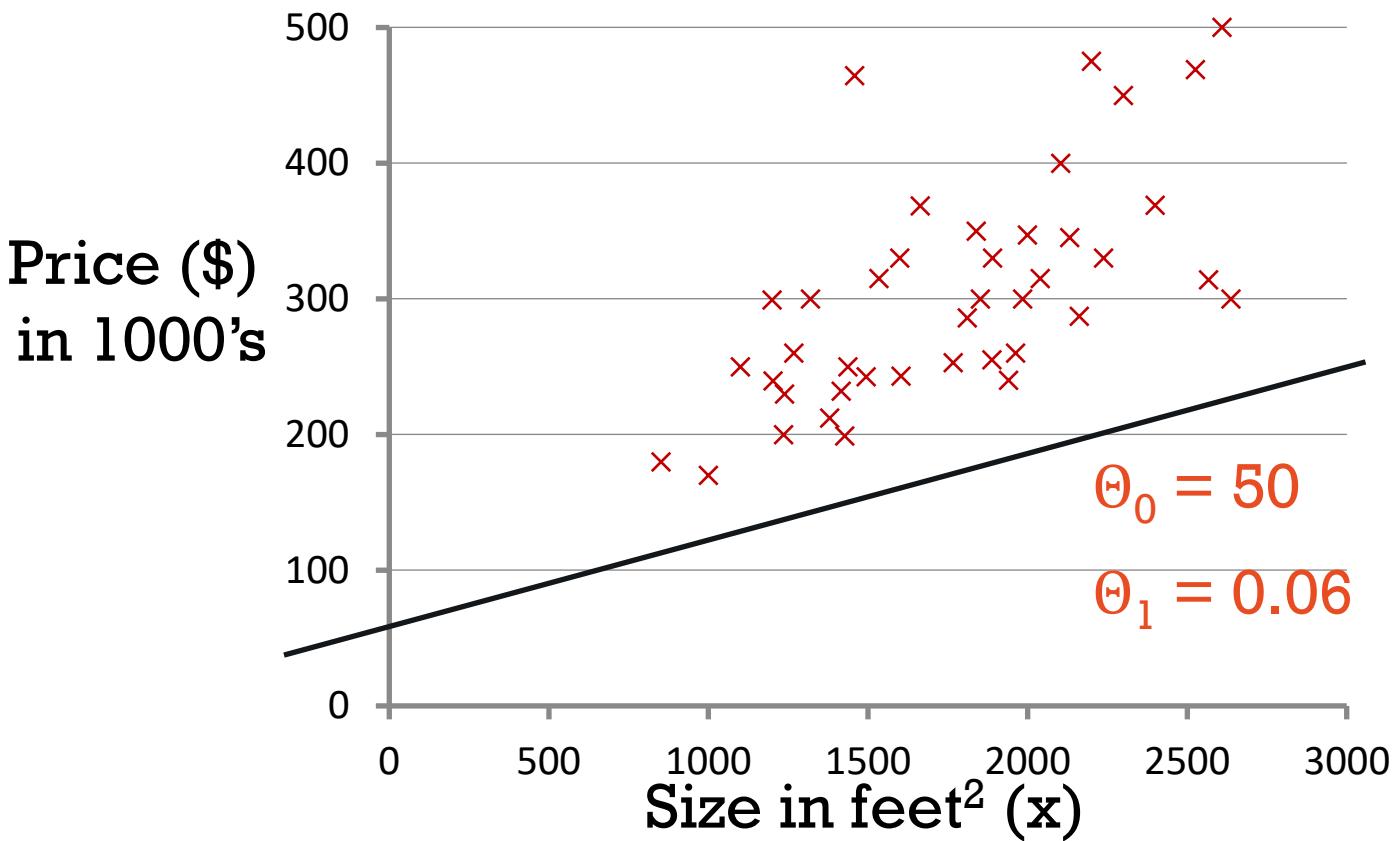
**Cost Function:**  $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$

**Goal:**  $\underset{\theta_0, \theta_1}{\text{minimize}} J(\theta_0, \theta_1)$



$$h_{\theta}(x)$$

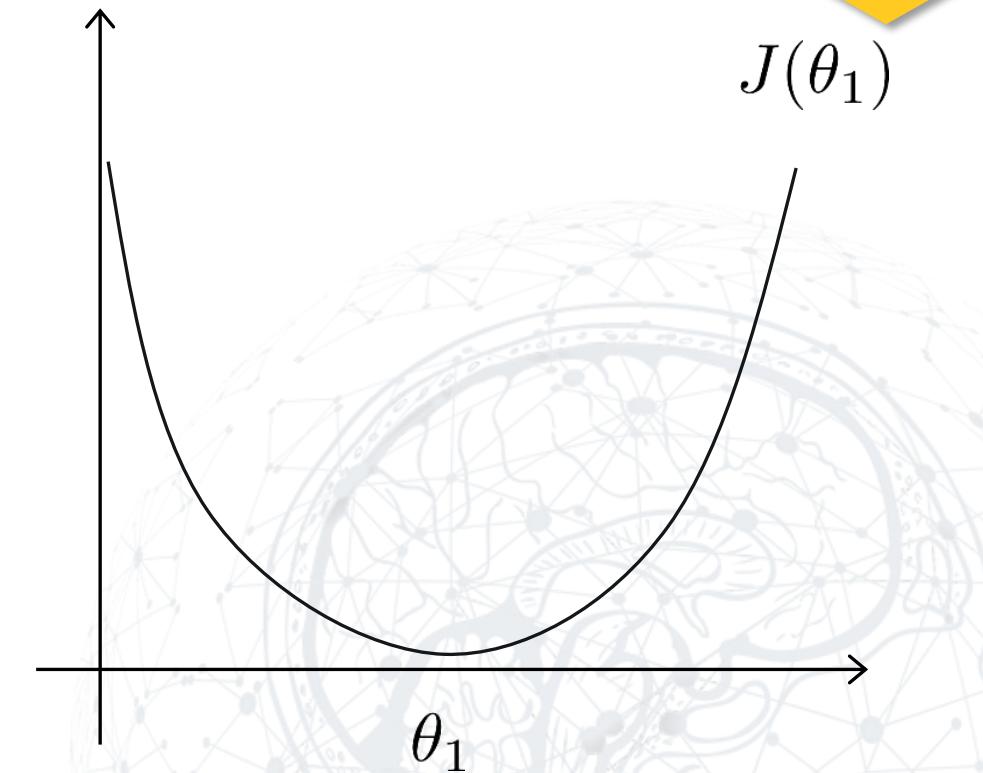
(for fixed  $\theta_0, \theta_1$ , this is a function of x)



$$h_{\theta}(x) = 50 + 0.06x$$

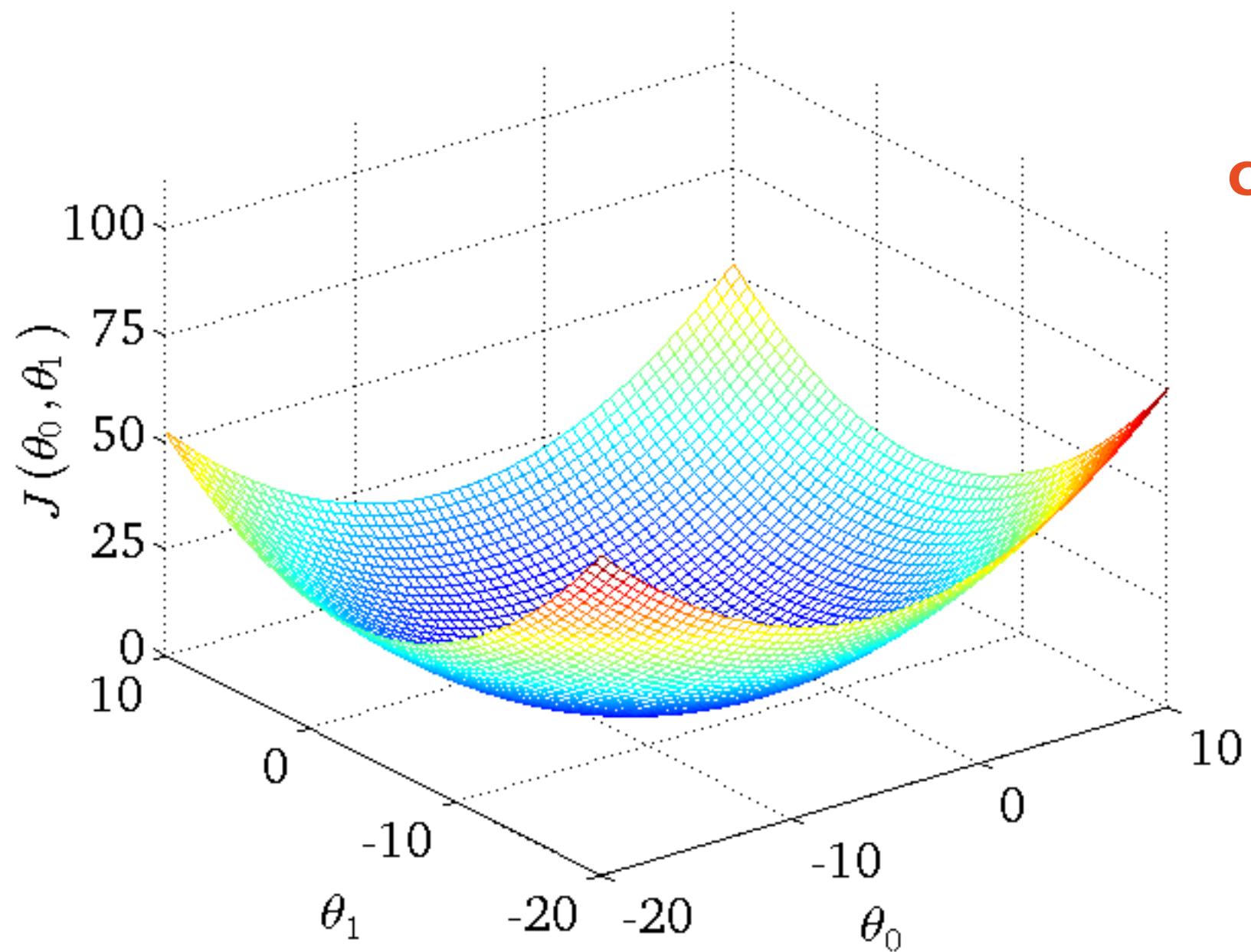
$$J(\theta_0, \theta_1)$$

(function of the parameters  $\theta_0, \theta_1$ )



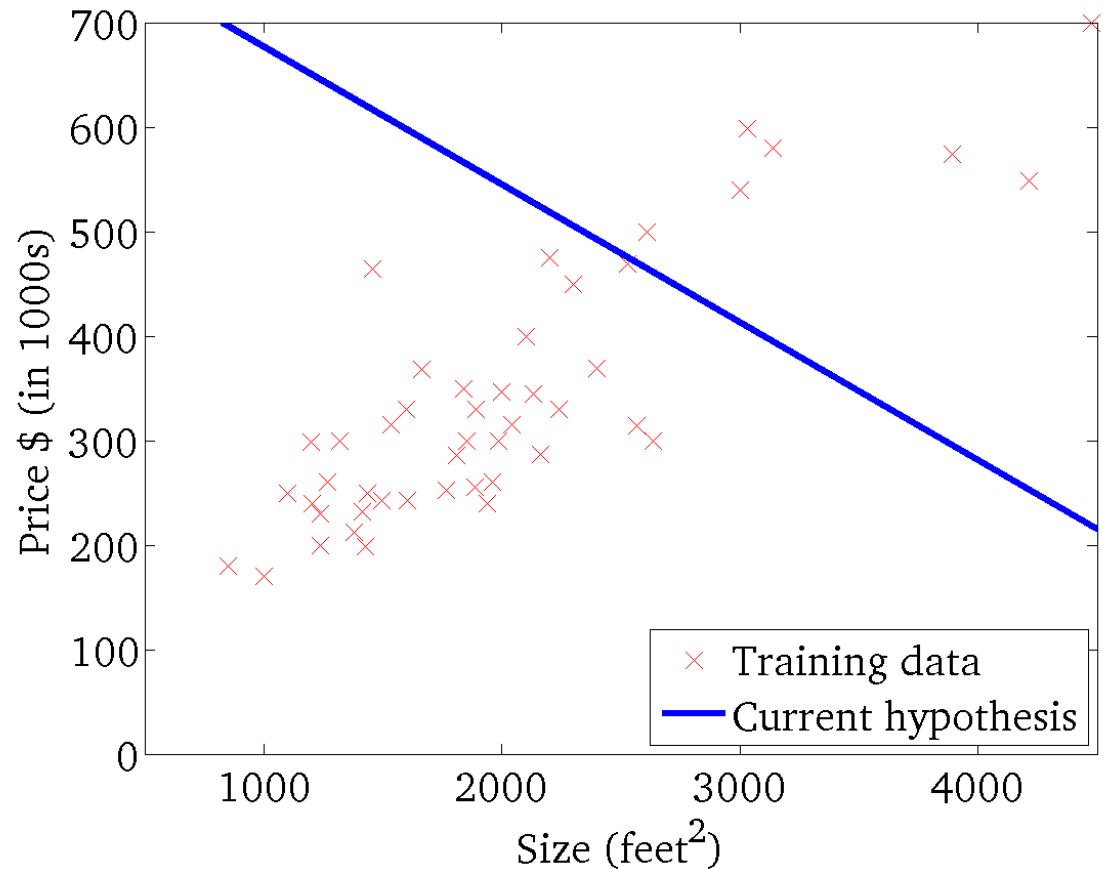


## Contour plots



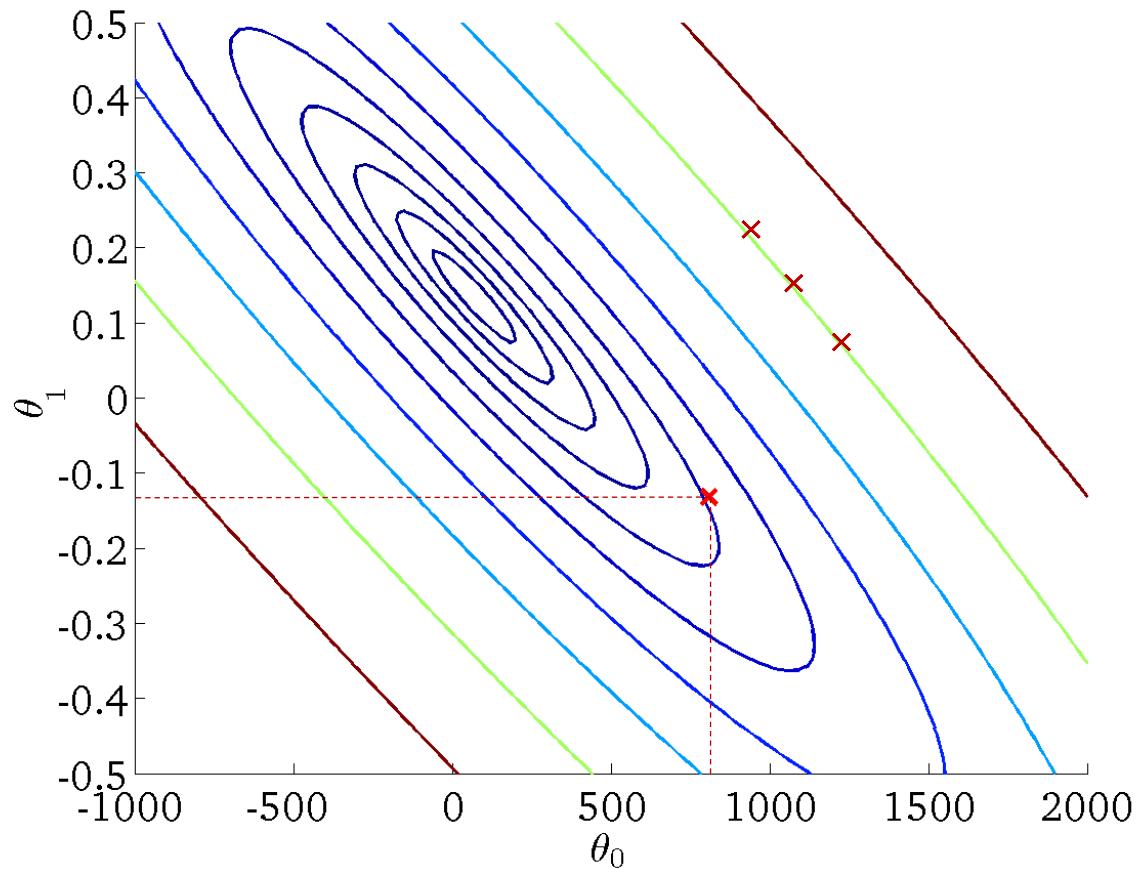
$$h_{\theta}(x)$$

(for fixed  $\theta_0, \theta_1$ , this is a function of  $x$ )



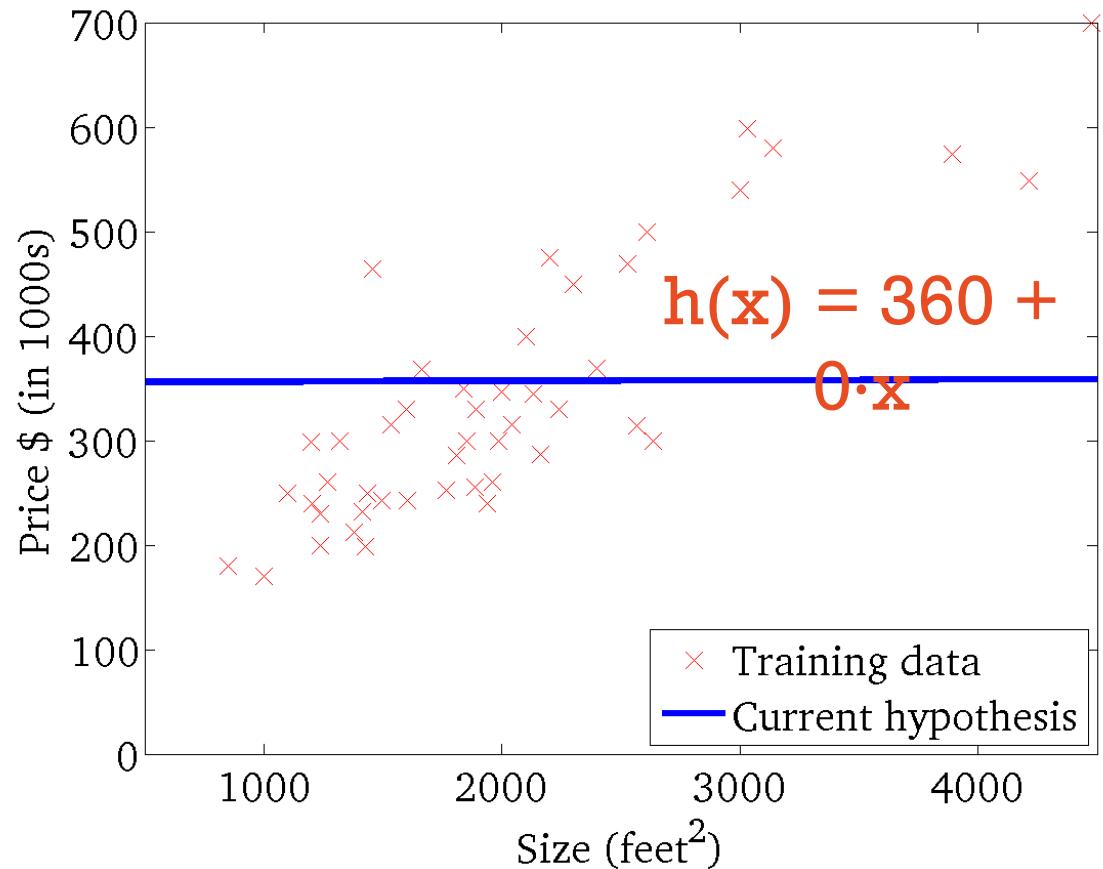
$$J(\theta_0, \theta_1)$$

(function of the parameters  $\theta_0, \theta_1$ )



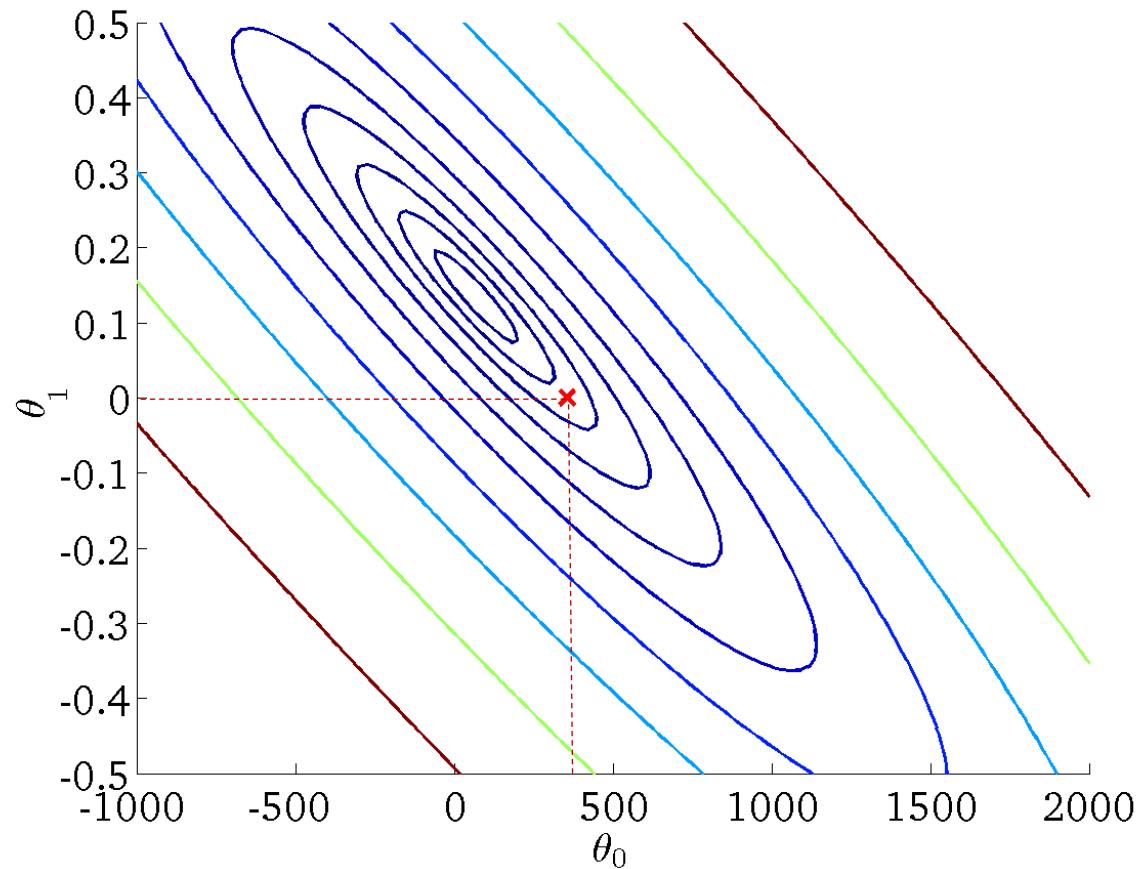
$$h_{\theta}(x)$$

(for fixed  $\theta_0, \theta_1$ , this is a function of  $x$ )



$$J(\theta_0, \theta_1)$$

(function of the parameters  $\theta_0, \theta_1$ )

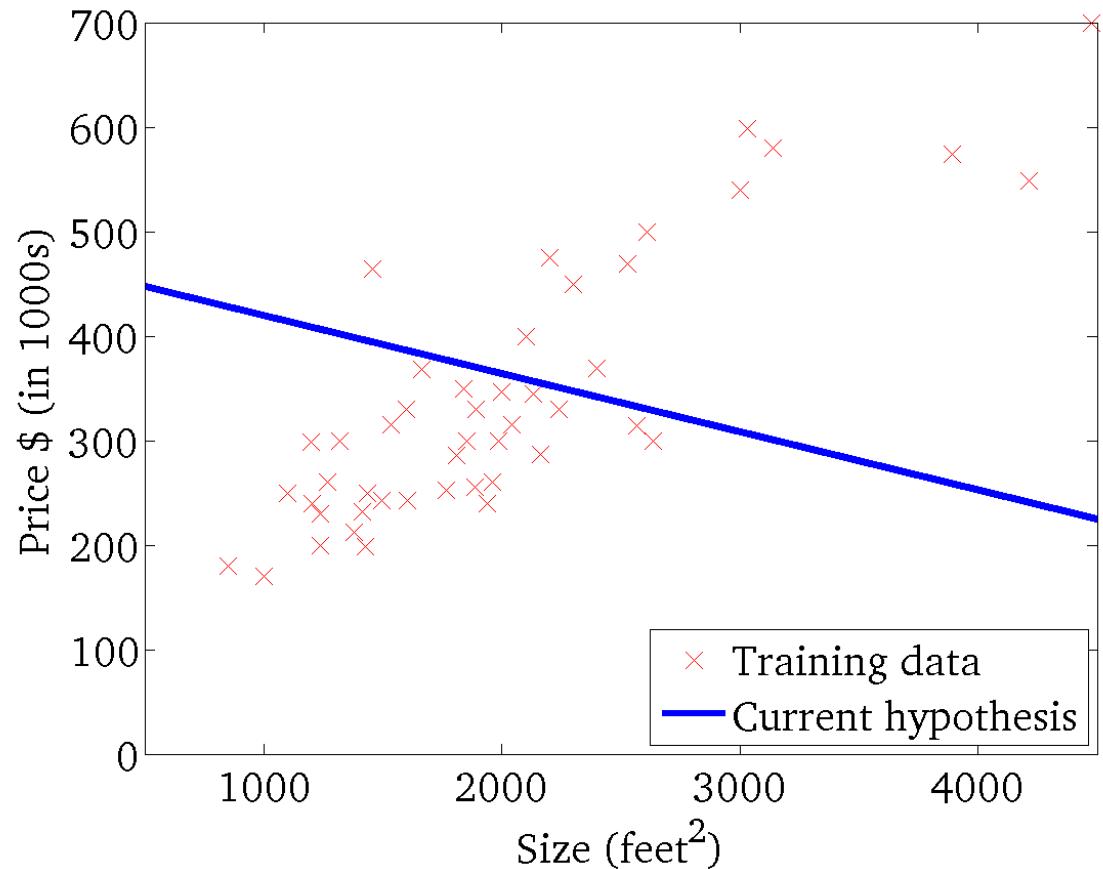


$$\theta_0 = 360$$

$$\theta_1 = 0$$

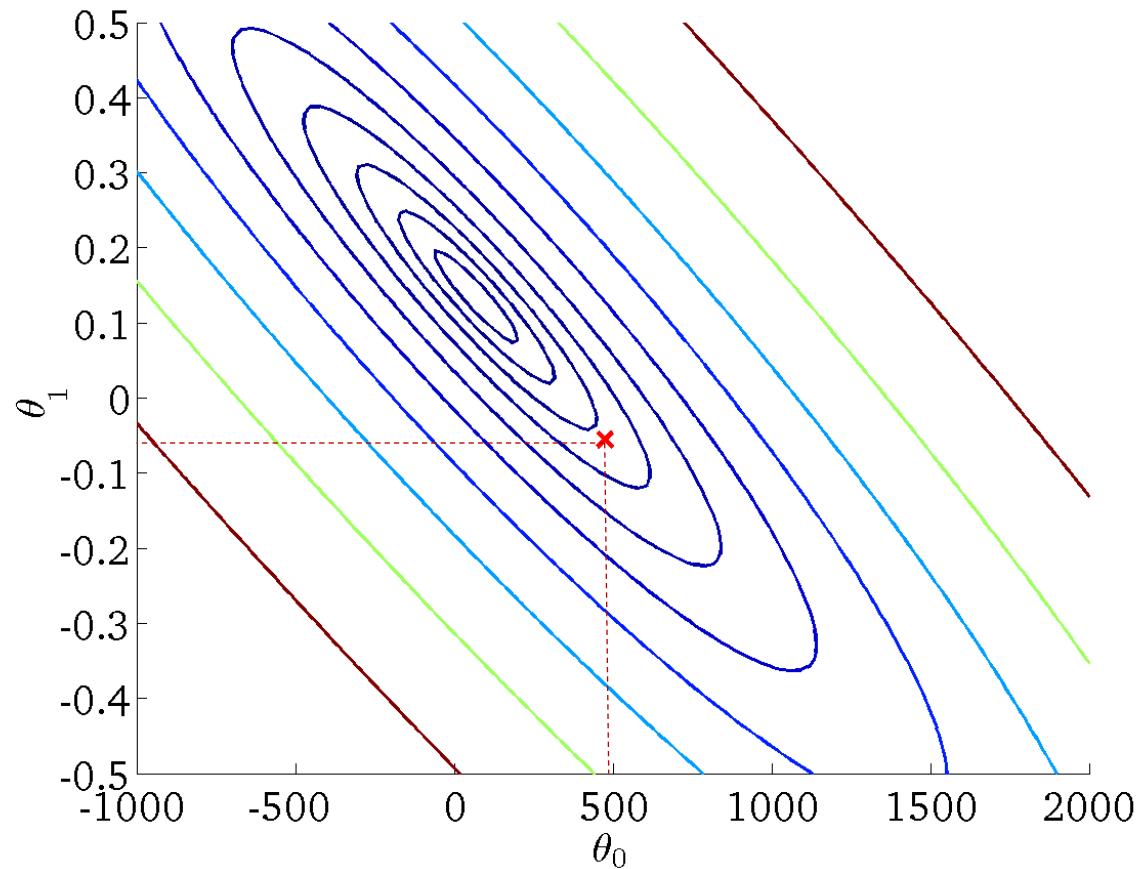
$$h_{\theta}(x)$$

(for fixed  $\theta_0, \theta_1$ , this is a function of  $x$ )



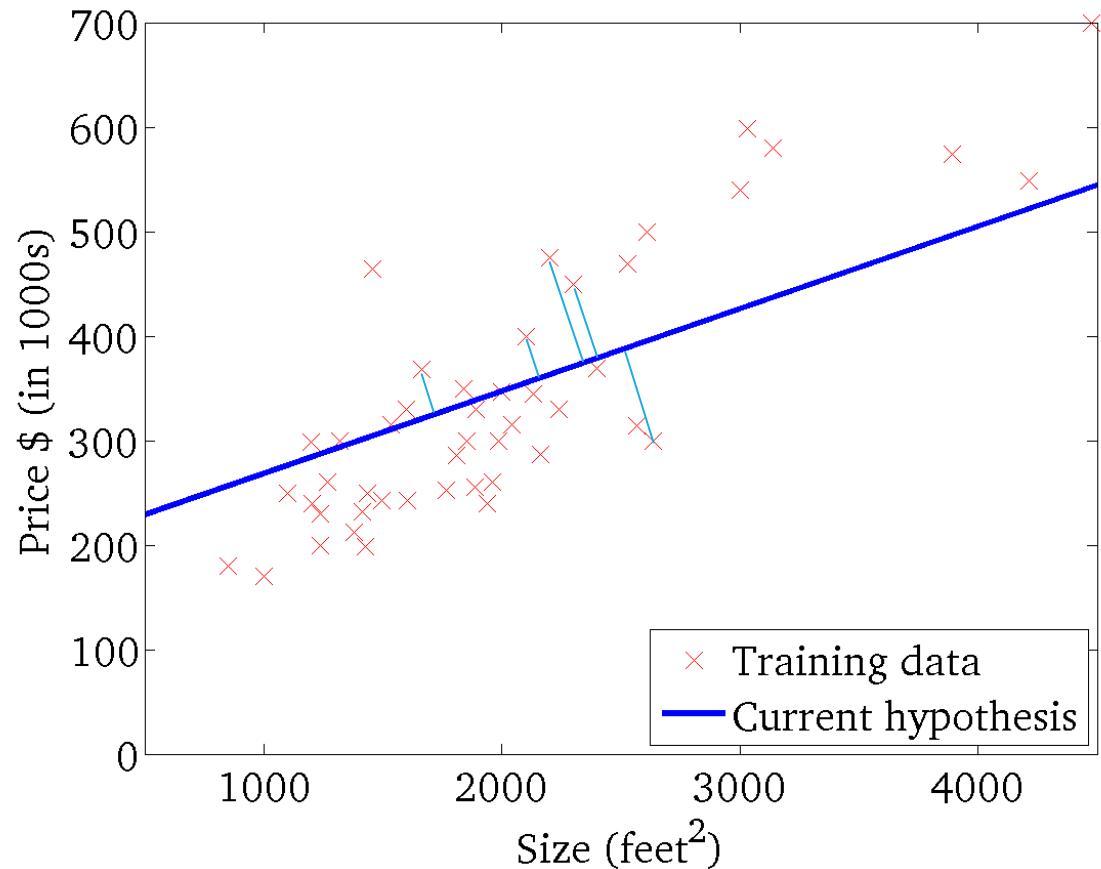
$$J(\theta_0, \theta_1)$$

(function of the parameters  $\theta_0, \theta_1$ )



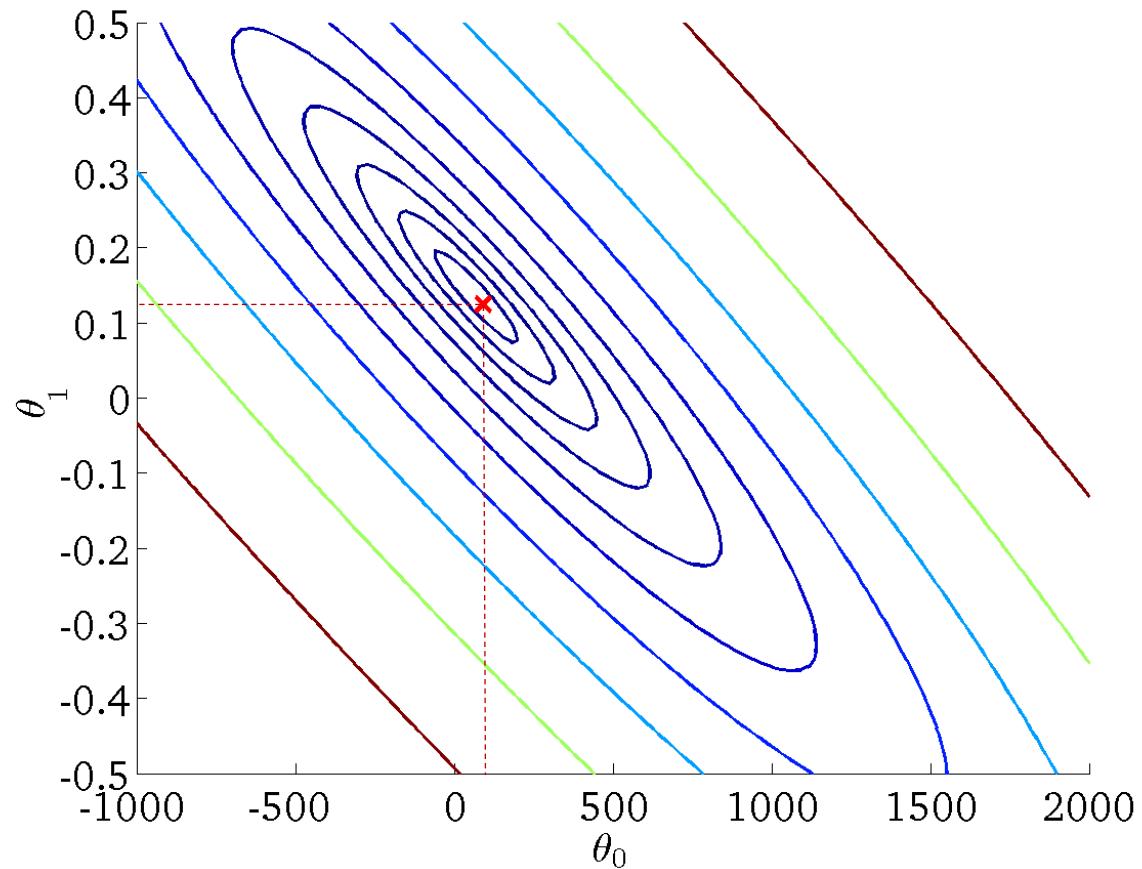
$$h_{\theta}(x)$$

(for fixed  $\theta_0, \theta_1$ , this is a function of  $x$ )



$$J(\theta_0, \theta_1)$$

(function of the parameters  $\theta_0, \theta_1$ )





UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

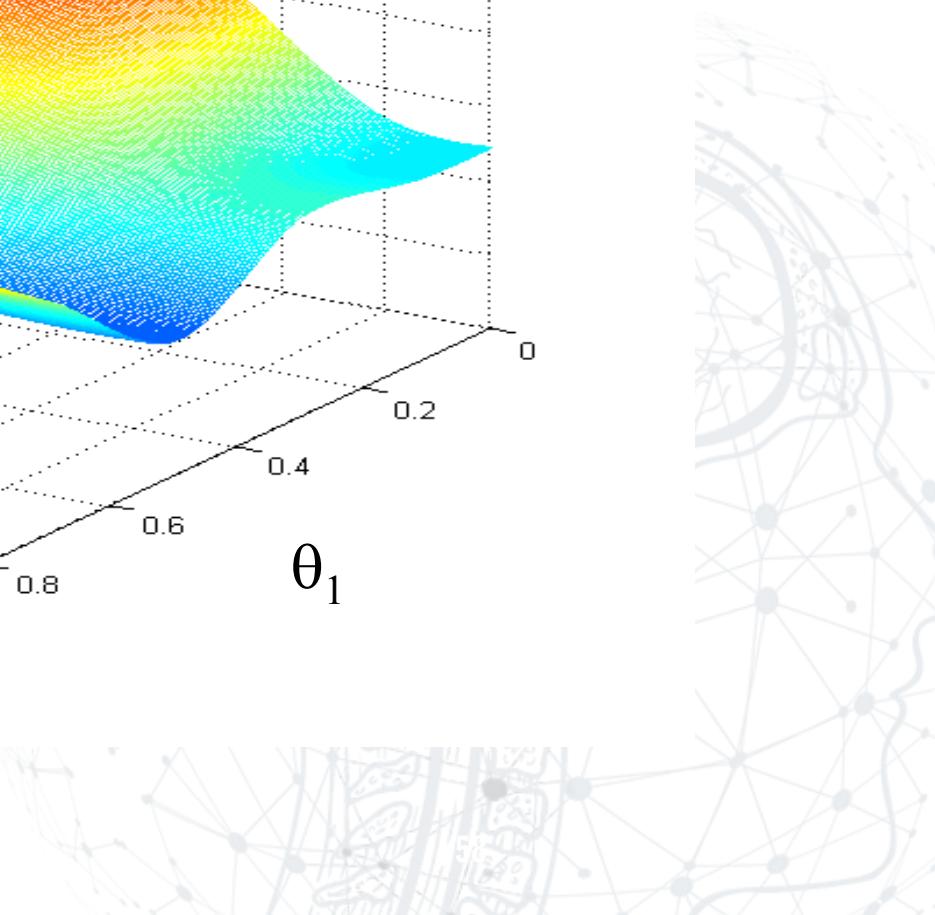
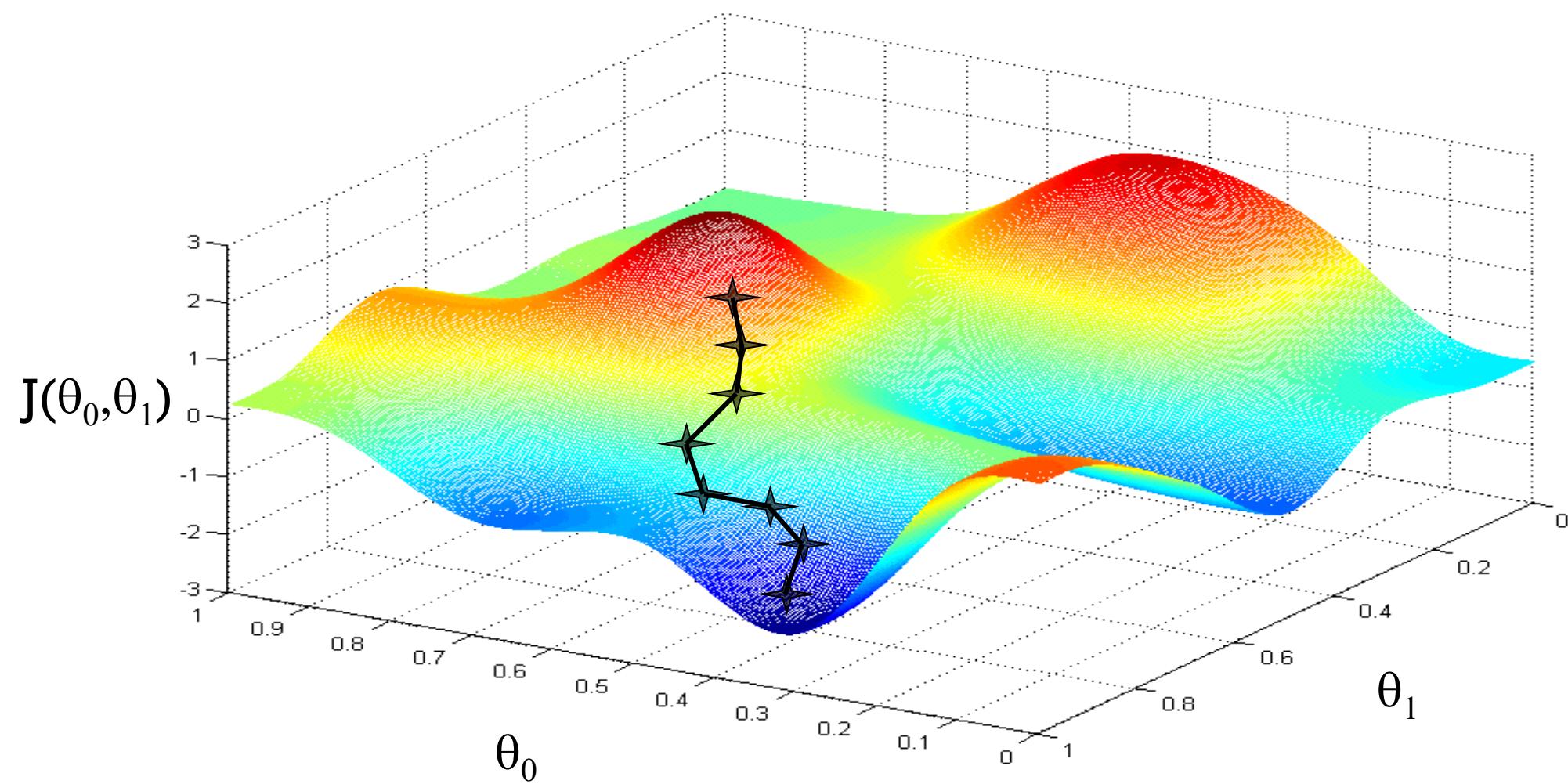
# Gradient Descent

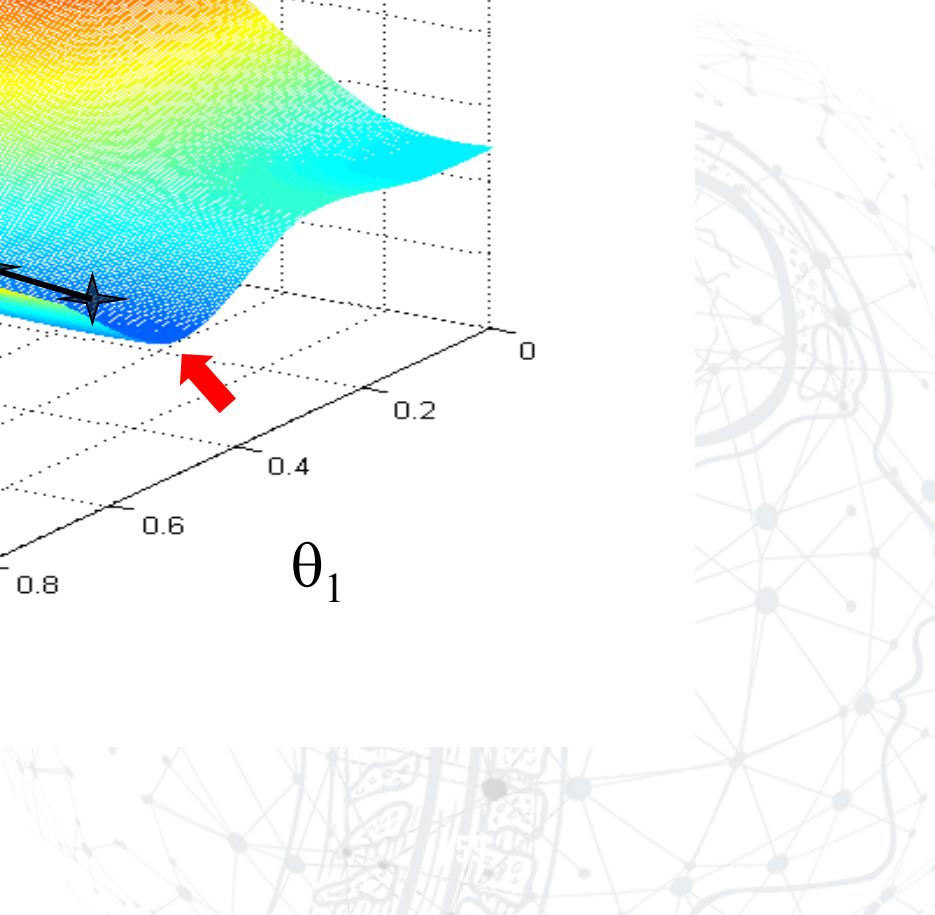
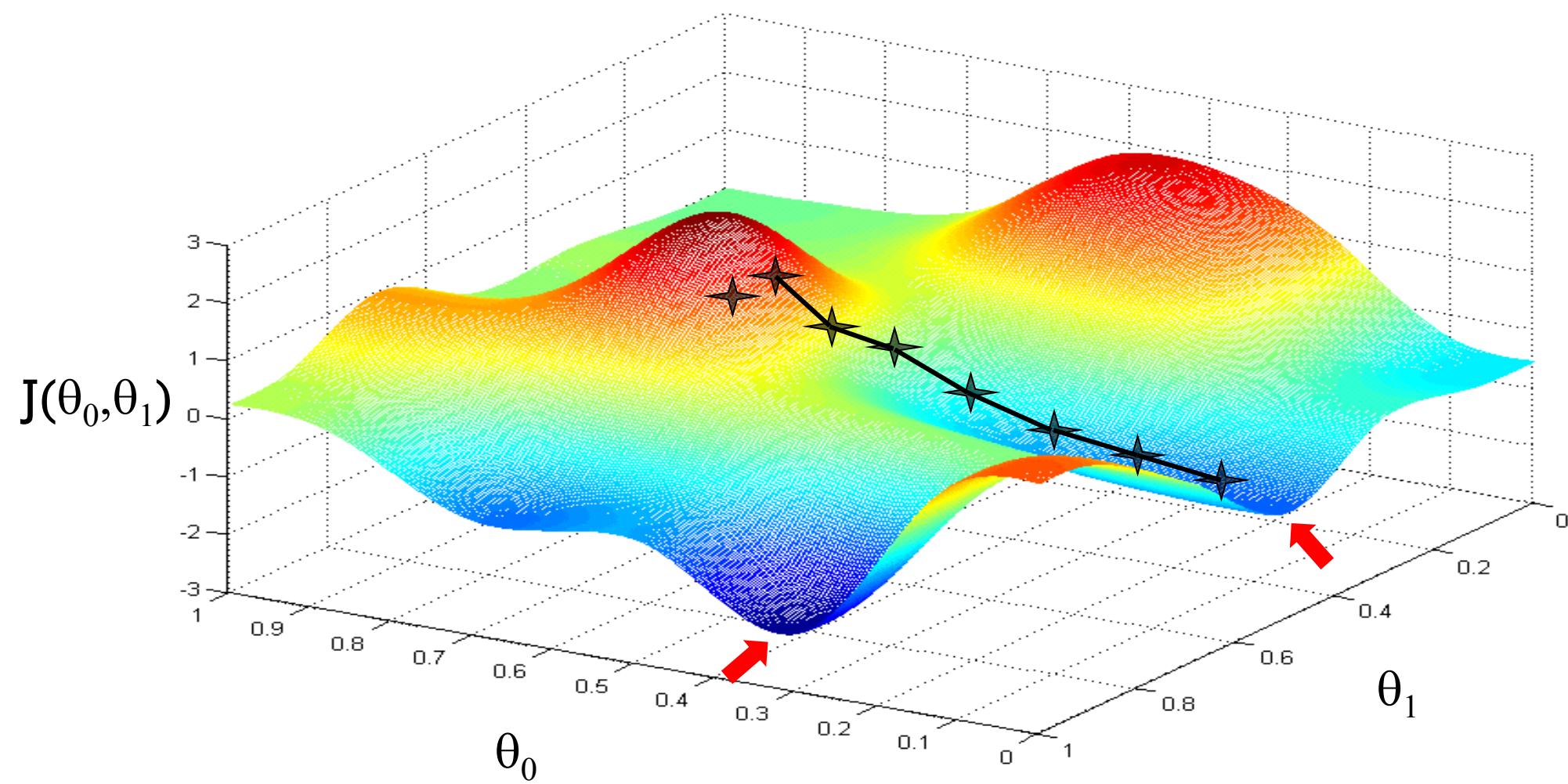
Have some function  $J(\theta_0, \theta_1)$

Want  $\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)$

## Outline:

- Start with some  $\theta_0, \theta_1$
  - Keep changing  $\theta_0, \theta_1$  to reduce  $J(\theta_0, \theta_1)$
- until we hopefully end up at a minimum





# Gradient descent algorithm

repeat until convergence {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) \quad (\text{for } j = 0 \text{ and } j = 1)$$

}

**Learning rate**

assignment

$a := b$

Simultaneously  
update  $\theta_0$  &  $\theta_1$

## Correct: Simultaneous update

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\theta_1 := \text{temp1}$$

## Incorrect:

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_1 := \text{temp1}$$

# Gradient descent algorithm

repeat until convergence {

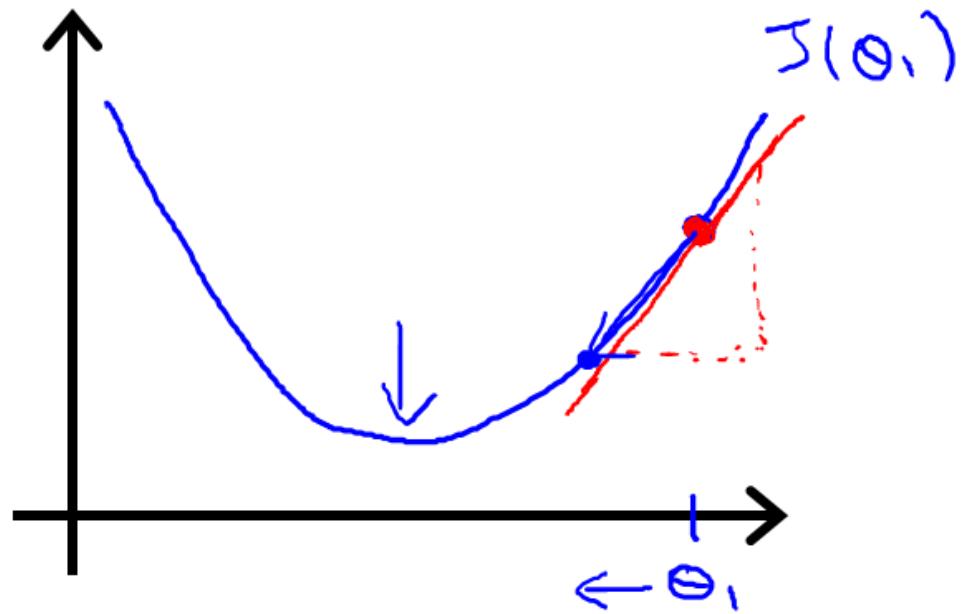
$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

}

**Learning rate**

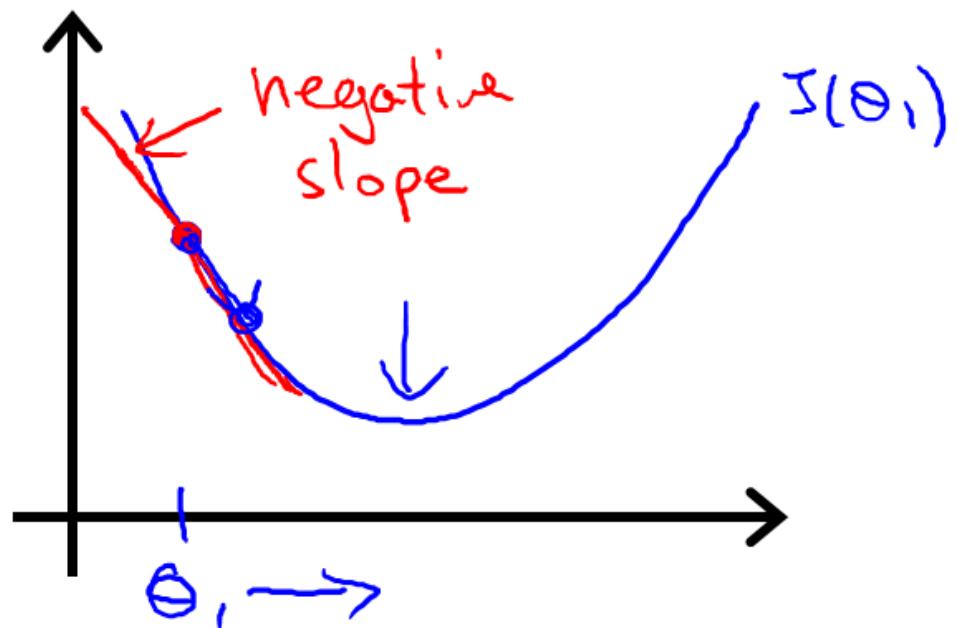
derivative

(simultaneously update  
 $j = 0$  and  $j = 1$ )



$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1) \geq 0$$

$\theta_1 := \theta_1 - \alpha \cdot J(\text{positive number})$



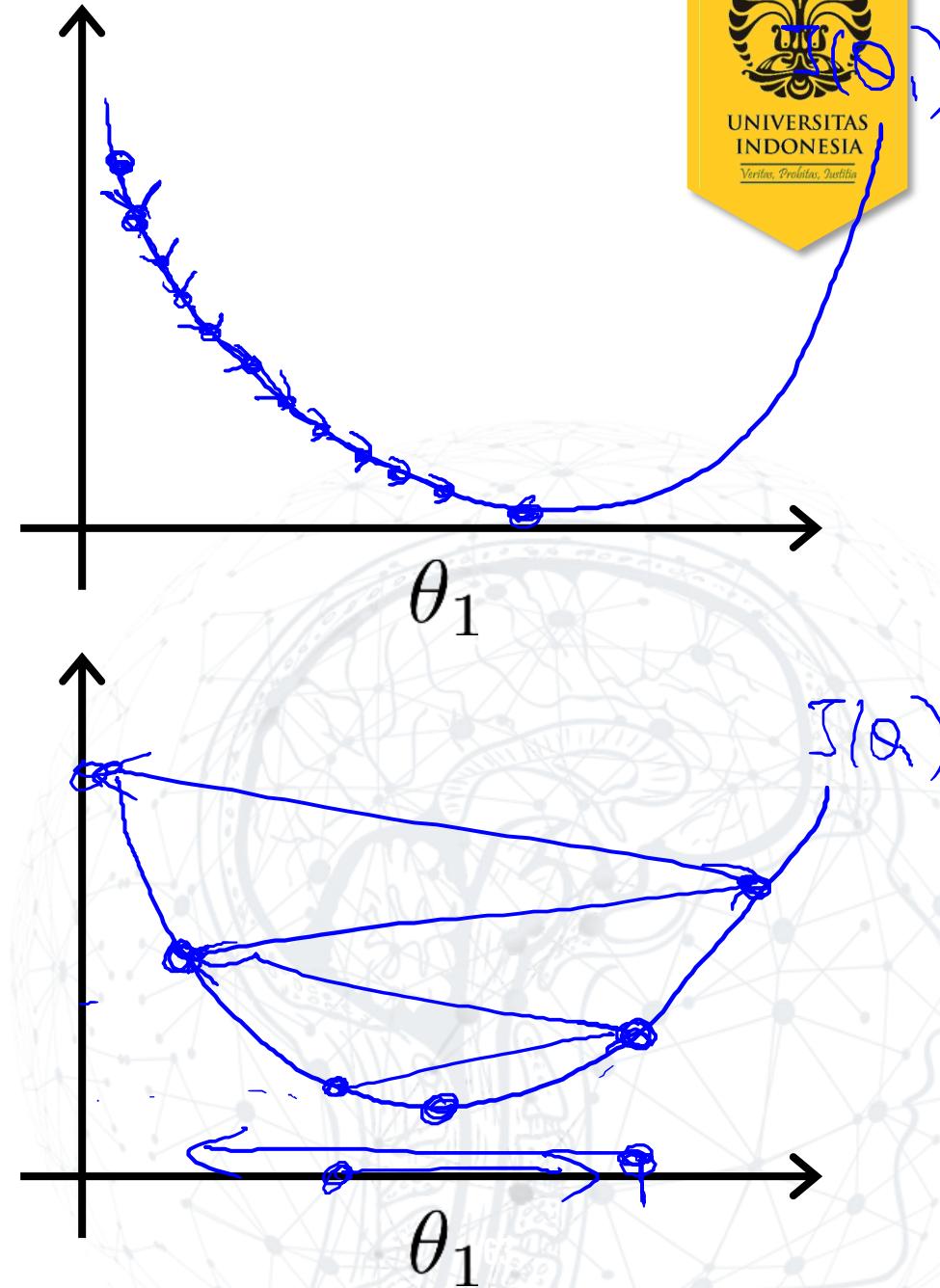
$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1) \leq 0$$

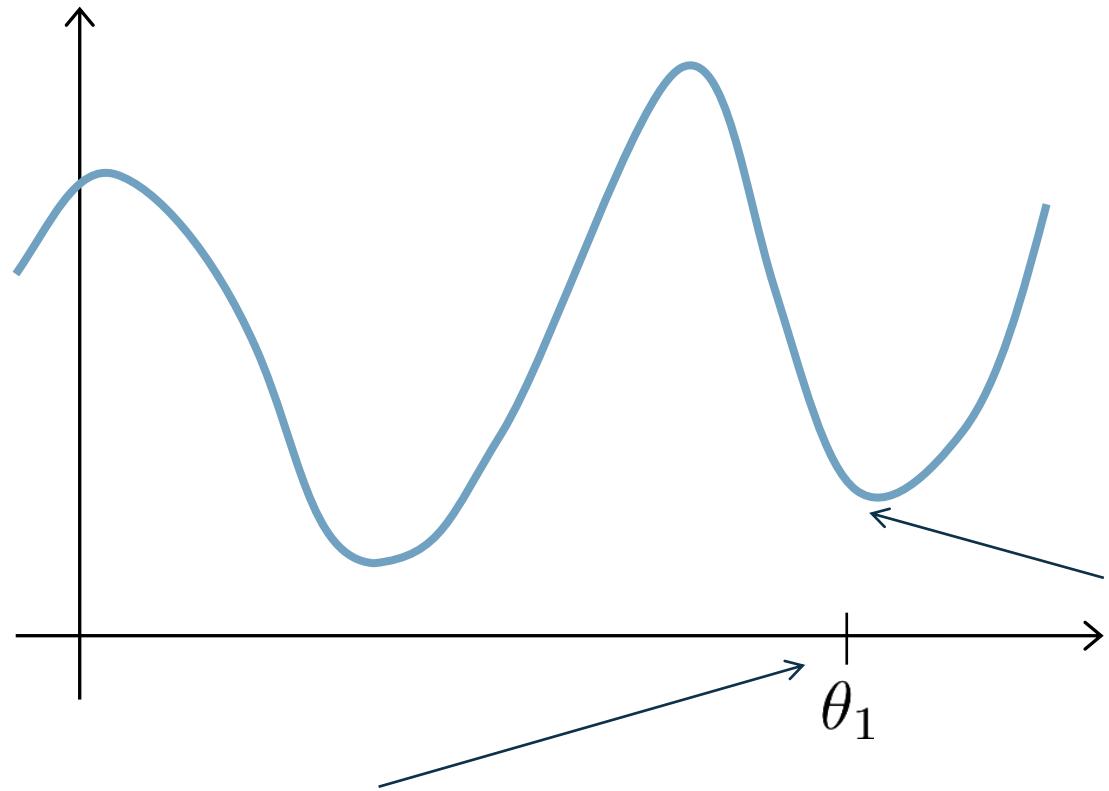
$\theta_1 := \theta_1 - \alpha \cdot J(\text{negative number})$

$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If  $\alpha$  is too small, gradient descent can be slow.

If  $\alpha$  is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.





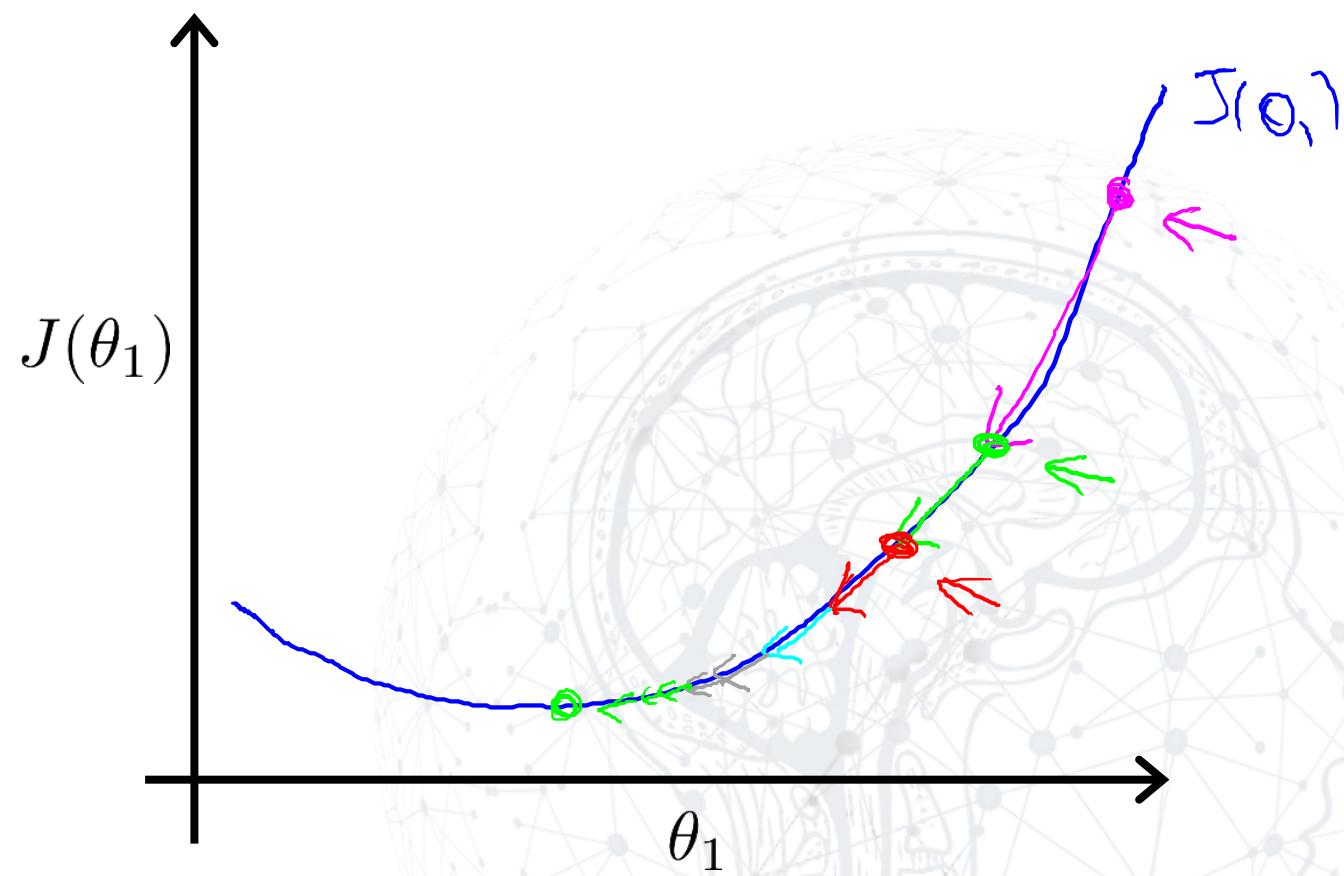
Current value of  $\theta_1$

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

Gradient descent can converge to a local minimum, even with the learning rate  $\alpha$  fixed.

$$\theta_1 := \theta_1 - \alpha \frac{d}{d\theta_1} J(\theta_1)$$

As we approach a local minimum, gradient descent will automatically take smaller steps. So, no need to decrease  $\alpha$  over time.



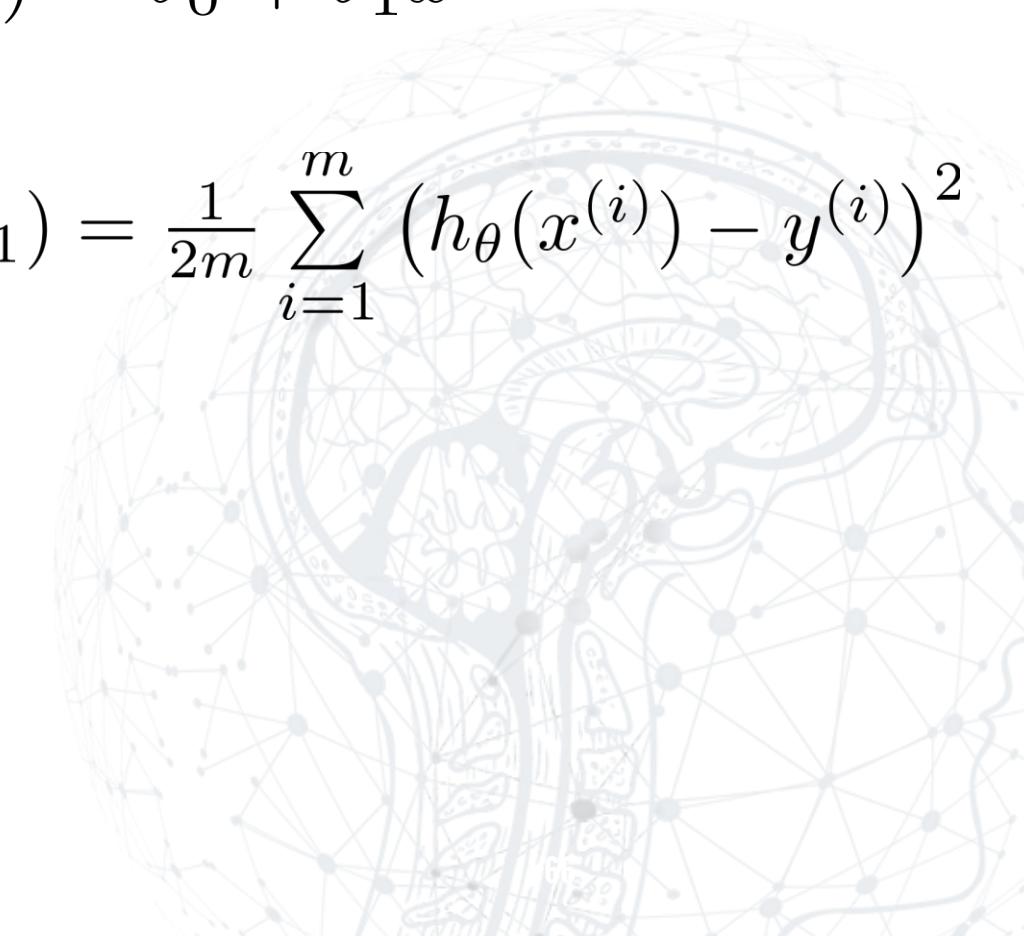
## Gradient descent algorithm

```
repeat until convergence {  
     $\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$   
    (for  $j = 1$  and  $j = 0$ )  
}
```

## Linear Regression Model

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$



$$\begin{aligned}\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1) &= \frac{d}{d\theta_j} \cdot \frac{1}{2m} \sum_{i=1}^m (\mathbf{h}_\Theta(\mathbf{x}^{(i)}) - \mathbf{y}^{(i)})^2 \\ &= \frac{d}{d\theta_j} \cdot \frac{1}{2m} \sum_{i=1}^m (\theta_0 + \theta_1 \cdot \mathbf{x}^{(i)} - \mathbf{y}^{(i)})^2\end{aligned}$$

$$j = 0 : \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (\mathbf{h}_\Theta(\mathbf{x}^{(i)}) - \mathbf{y}^{(i)})$$

$$j = 1 : \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1) = \frac{1}{m} \sum_{i=1}^m (\mathbf{h}_\Theta(\mathbf{x}^{(i)}) - \mathbf{y}^{(i)}) \cdot x^{(i)}$$

# Gradient descent algorithm

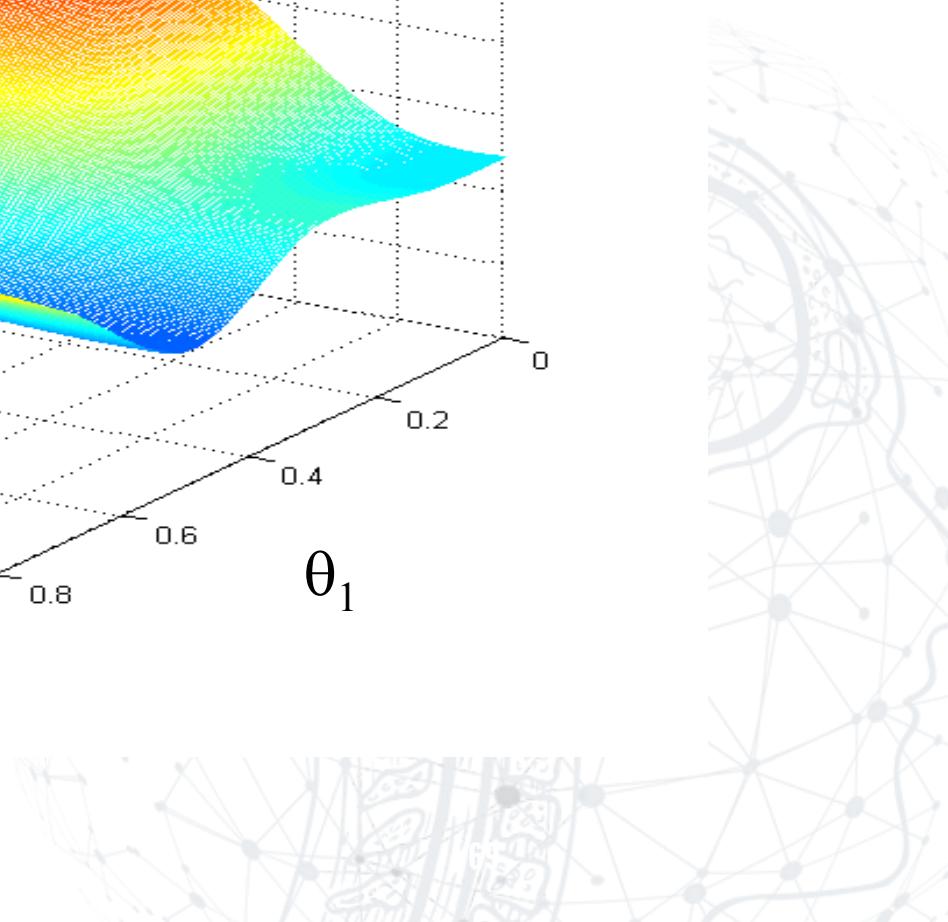
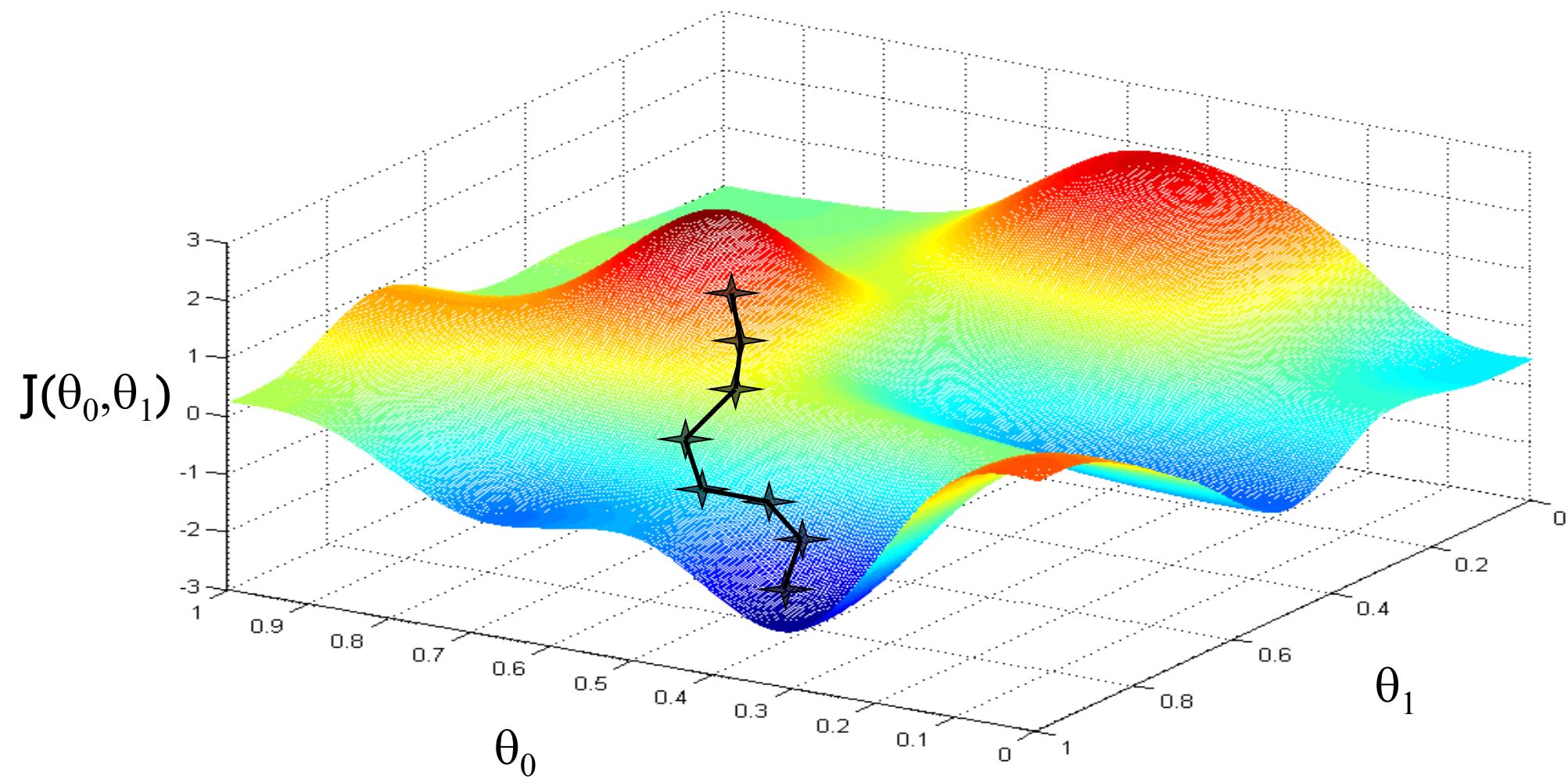
repeat until convergence {

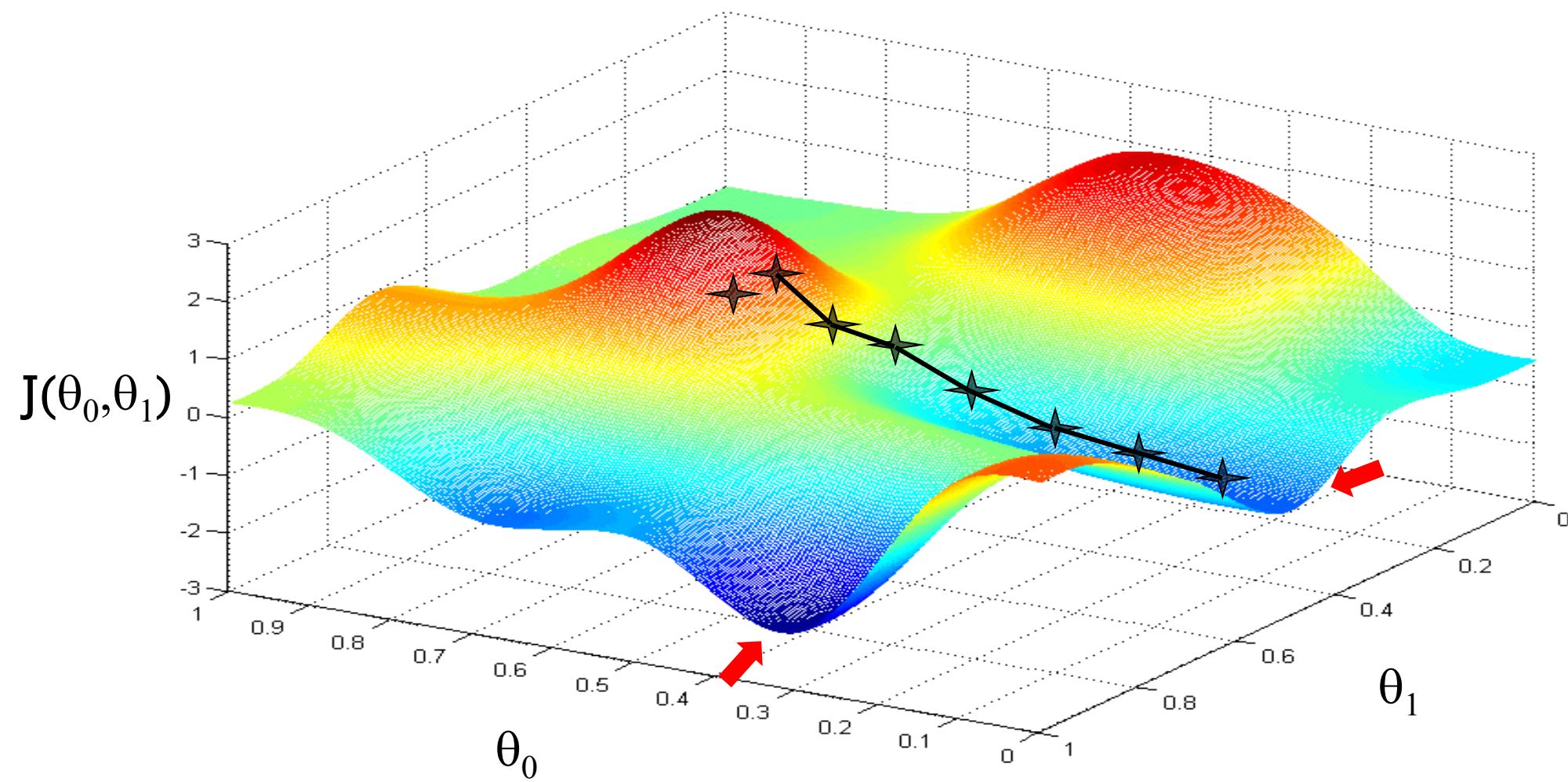
$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) \cdot \frac{d}{d\theta_0} \cdot J(\theta_0, \theta_1)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) \cdot x^{(i)} \cdot \frac{d}{d\theta_1} \cdot J(\theta_0, \theta_1)$$

}

**update  
 $\theta_0$  and  $\theta_1$   
simultaneously**







UNIVERSITAS  
INDONESIA

*Veritas, Probono, Justitia*

# STOCHASTIC VS BATCH Gradient Descent



UNIVERSITAS  
INDONESIA  
*Veritas, Prudentia, Justitia*

# Stochastic Gradient Descent

- Repeatedly run through the training set, and each time we encounter a training example, we update the parameters according to the gradient of the error with respect to that single training example only.
- Also called incremental gradient descent



# BATCH Gradient Descent

- This method looks at every example in the entire training set on every step, to update the  $\theta$
- Modifikasi: mini batch  $\rightarrow$  sekelompok data digunakan dalam 1x iterasi, misal batch\_size = 16 artinya 16 buah data digunakan dalam 1x iterasi

# STOCHASTIC

Loop {

for i=1 to m, {

$$\theta_j := \theta_j + \alpha (y^{(i)} - h_{\theta}(x^{(i)})) x_j^{(i)} \quad (\text{for every } j).$$

}

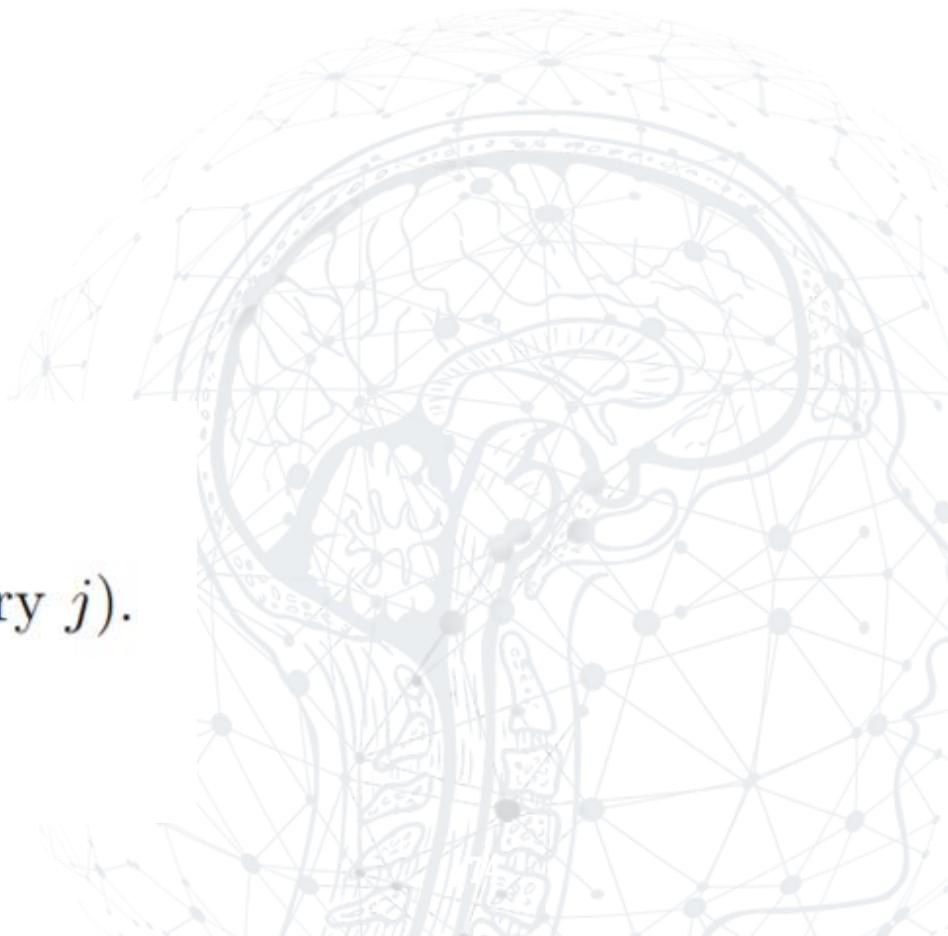
}

# BATCH

Repeat until convergence {

$$\theta_j := \theta_j + \alpha \sum_{i=1}^m (y^{(i)} - h_{\theta}(x^{(i)})) x_j^{(i)} \quad (\text{for every } j).$$

}



- Whereas batch gradient descent has to scan through the entire training set before taking a single step—a costly operation if  $m$  is large—stochastic gradient descent can start making progress right away, and continues to make progress with each example it looks at.
- Often, stochastic gradient descent gets  $\theta$  “close” to the minimum much faster than batch gradient descent.
- Note however that it may never “converge” to the minimum, and the parameters  $\theta$  will keep oscillating around the minimum of  $J(\theta)$ ; but in practice most of the values near the minimum will be reasonably good approximations to the true minimum.
- For these reasons, particularly when the training set is large, stochastic gradient descent is often preferred over batch gradient descent



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# MULTIPLE FEATURES/ Multiple variable regression



# Multiple features (variables).

Size (feet <sup>2</sup> )	Price (\$1000)
$x$	$y$
2104	460
1416	232
1534	315
852	178
...	...

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



# Multiple features (variables).

Size (feet <sup>2</sup> )	Number of bedrooms	Number of floors	Age of home (years)	Price (\$1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178
...	...	...	...	...

Notation:

$n$  = number of features

$x^{(i)}$  = input (features) of  $i^{th}$  training example.

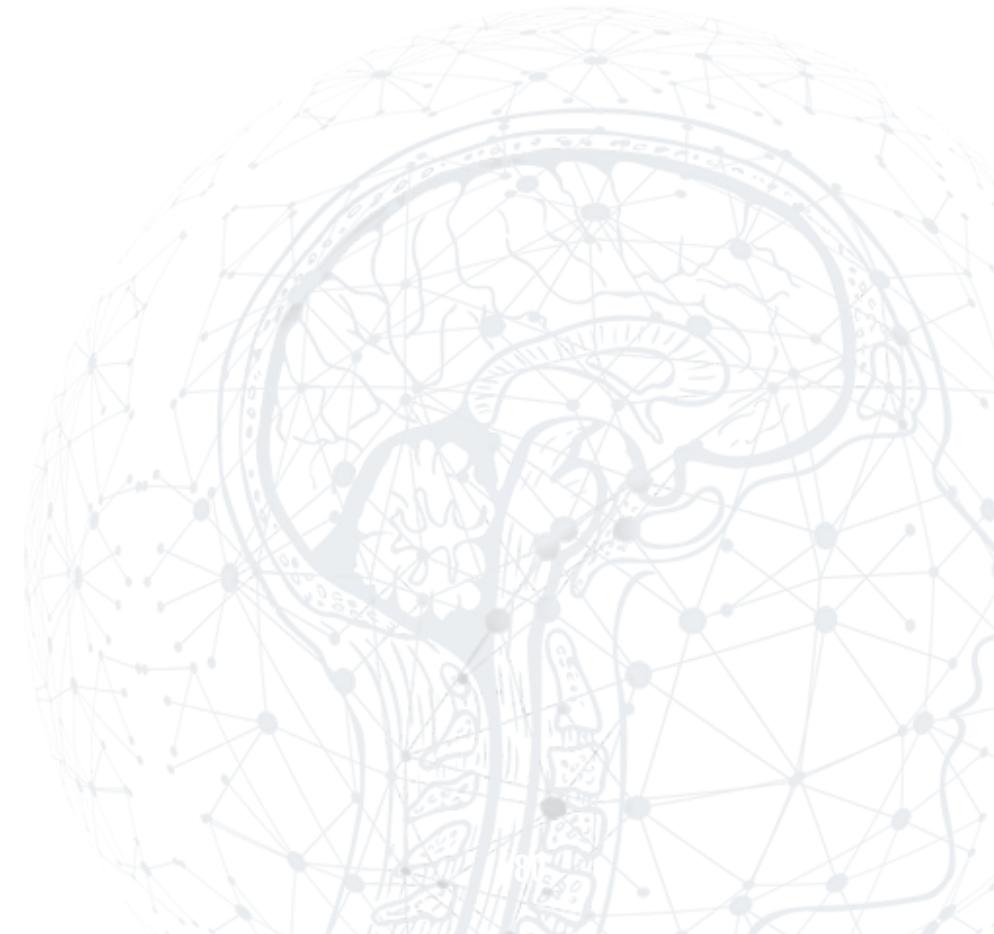
$x_j^{(i)}$  = value of feature  $j$  in  $i^{th}$  training example.



UNIVERSITAS  
INDONESIA  
*Veritas, Prudentia, Justitia*

**Hypothesis:**

**Previously:**  $h_{\theta}(x) = \theta_0 + \theta_1 x$



$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

For convenience of notation, define  $x_0 = 1$  .

Multivariate linear regression.

# Gradient descent for multiple variables

**Hypothesis:**  $h_{\theta}(x) = \theta^T x = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$

**Parameters:**  $\theta_0, \theta_1, \dots, \theta_n$

**Cost function:**

$$J(\theta_0, \theta_1, \dots, \theta_n) = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})^2$$

**Gradient descent:**

Repeat {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \dots, \theta_n)$$

}

(simultaneously update for every  $j = 0, \dots, n$  )

# Gradient Descent

Previously (n=1):

Repeat {

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial \theta_0} J(\theta)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x^{(i)}$$

(simultaneously update  $\theta_0, \theta_1$ )

}

New algorithm( $n \geq 1$ ) :

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update  $\theta_j$  for  
 $j = 0, \dots, n$  )

}

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^m (h_\theta(x^{(i)}) - y^{(i)}) x_2^{(i)}$$

...

Now, how to program?  
See the notebook given,  
and do the task in TK02



# Regresi

PENGANTAR KECERDASAN BUATAN

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI



Halo!

Selamat datang  
kembali di kuliah  
pengantar kecerdasan  
buatan!

**Dr. Prima Dewi Purnamasari**



Sebelumnya:  
Pembelajaran Mesin

**Dr. Prima Dewi Purnamasari**



Sekarang:  
Regresi

**Dr. Prima Dewi Purnamasari**

## Capaian pembelajaran

**Mampu menjelaskan metode regresi  
dalam pembelajaran mesin**

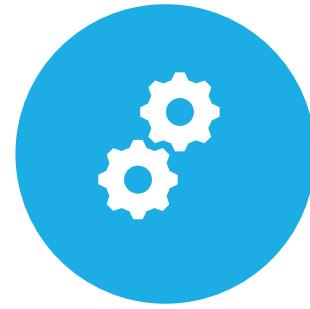


# Tipe pembelajaran machine learning



## Supervised Learning

Data yang dipelajari memiliki label input dan output yang jelas (berupa nilai kontinyu, kategori atau kelas)



## Unsupervised Learning

Data yang dipelajari hanya memiliki label input tanpa output yang jelas



## Reinforcement Learning

Data didapatkan berdasarkan reward dari lingkungan mesin



# Supervised Learning

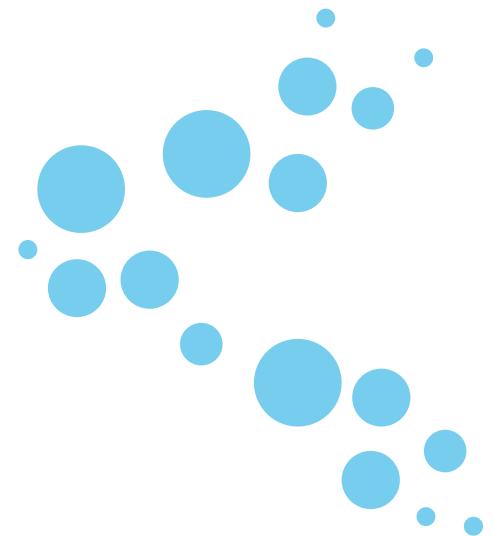
Data yang dipelajari memiliki label input dan output yang jelas (berupa nilai kontinyu, kategori atau kelas)

Label berupa kategori/kelas

Label berupa nilai kontinyu

## Klasifikasi

## Regresi



APEL



PISANG



ANGGUR



Rp.300juta



Rp.600juta



Rp.700juta



# Regresi dengan metode statistika



# Diketahui harga rumah



300 jt



600 jt



700 jt



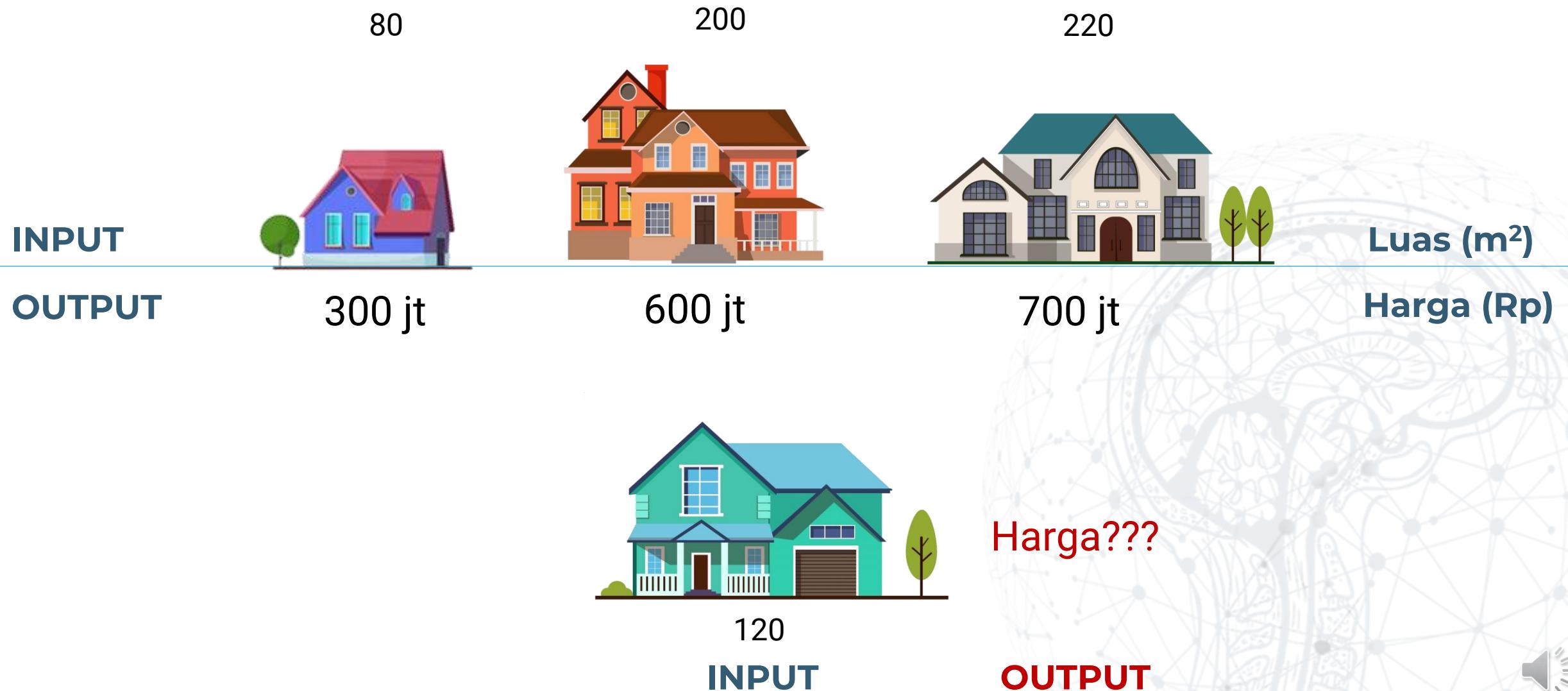
INPUT

Harga???

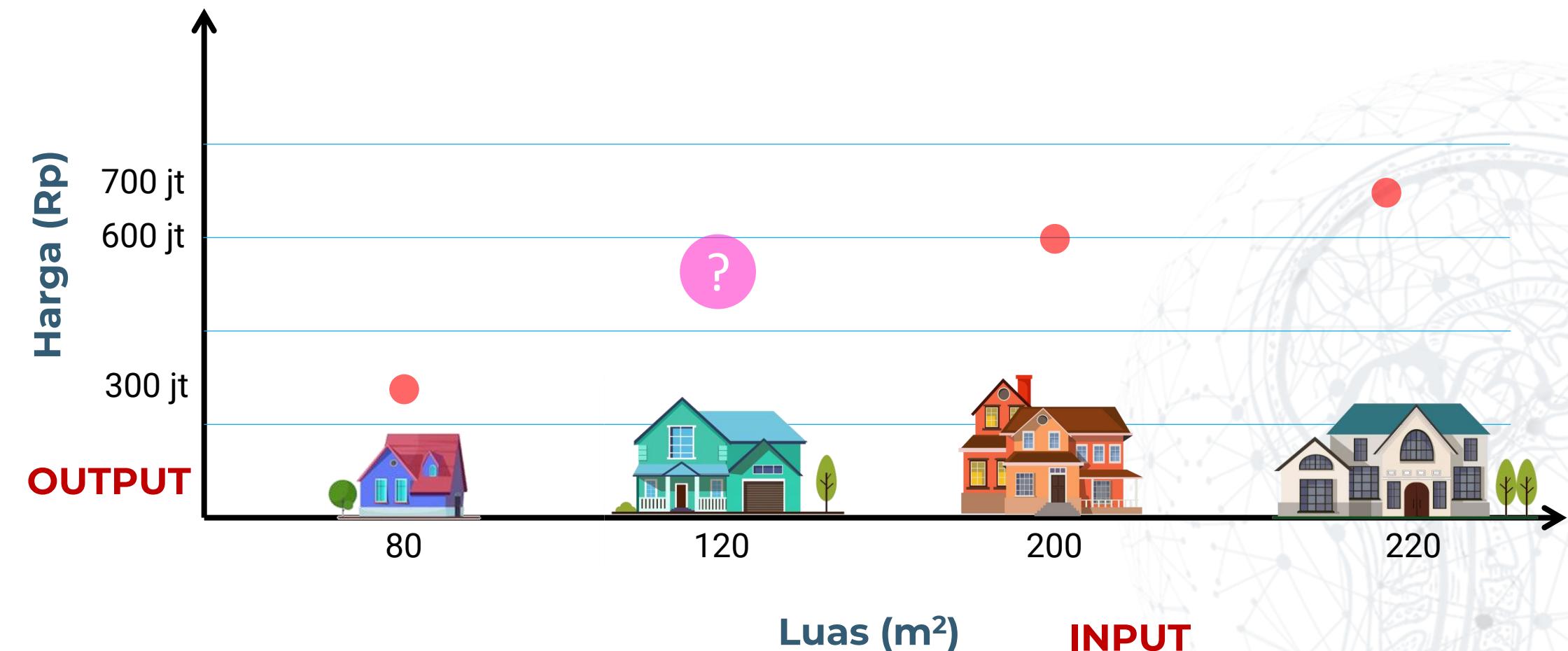
OUTPUT



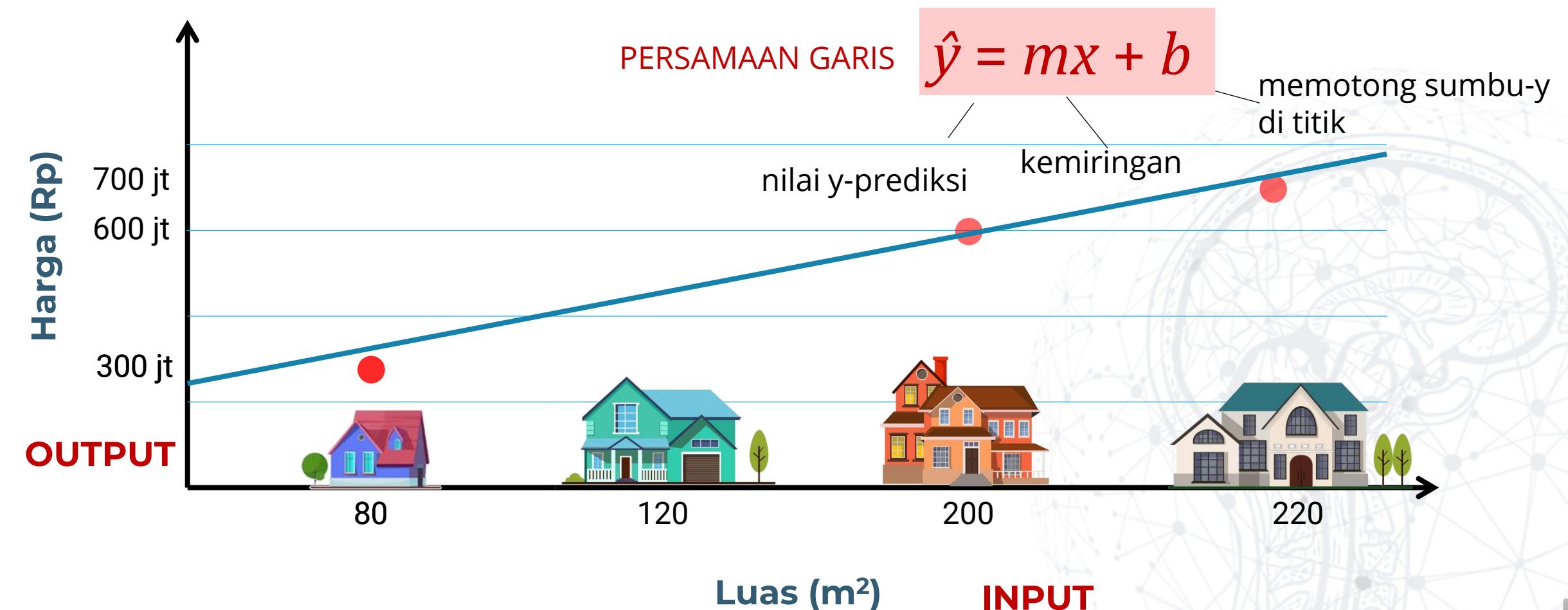
# Misalnya hanya diperhatikan luas rumah saja



## Cari hubungan antara input (*independent variable*) dan output (*dependent/response variable*)



# Hubungan yang paling mudah untuk dibuat: GARIS



# Metode regresi statistika (*simple linear regression*)

$$\hat{y} = mx + b$$

$$m = \frac{n \sum xy - (\sum x)(\sum y)}{n \sum x^2 - (\sum x)^2}$$

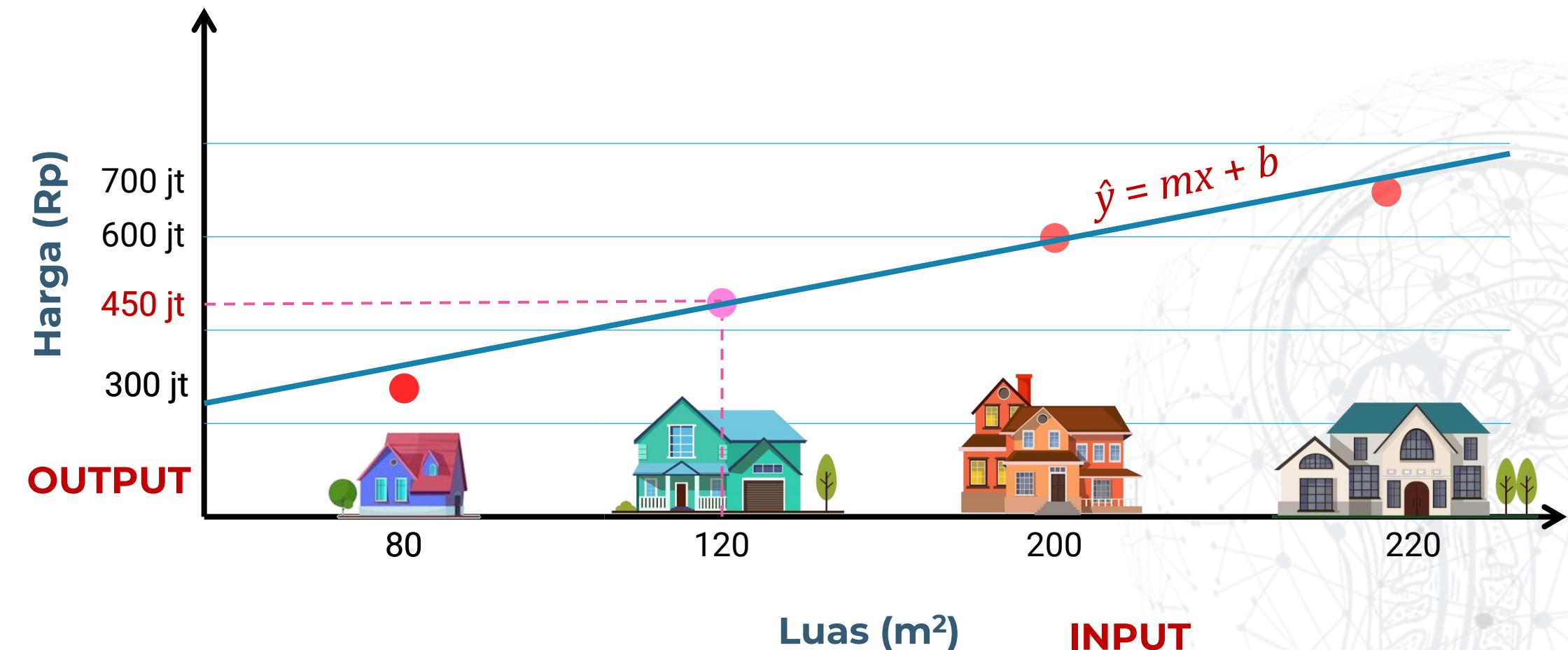
$$b = \bar{y} - m\bar{x} = \frac{\sum y}{n} - m \frac{\sum x}{n}$$

Dengan:

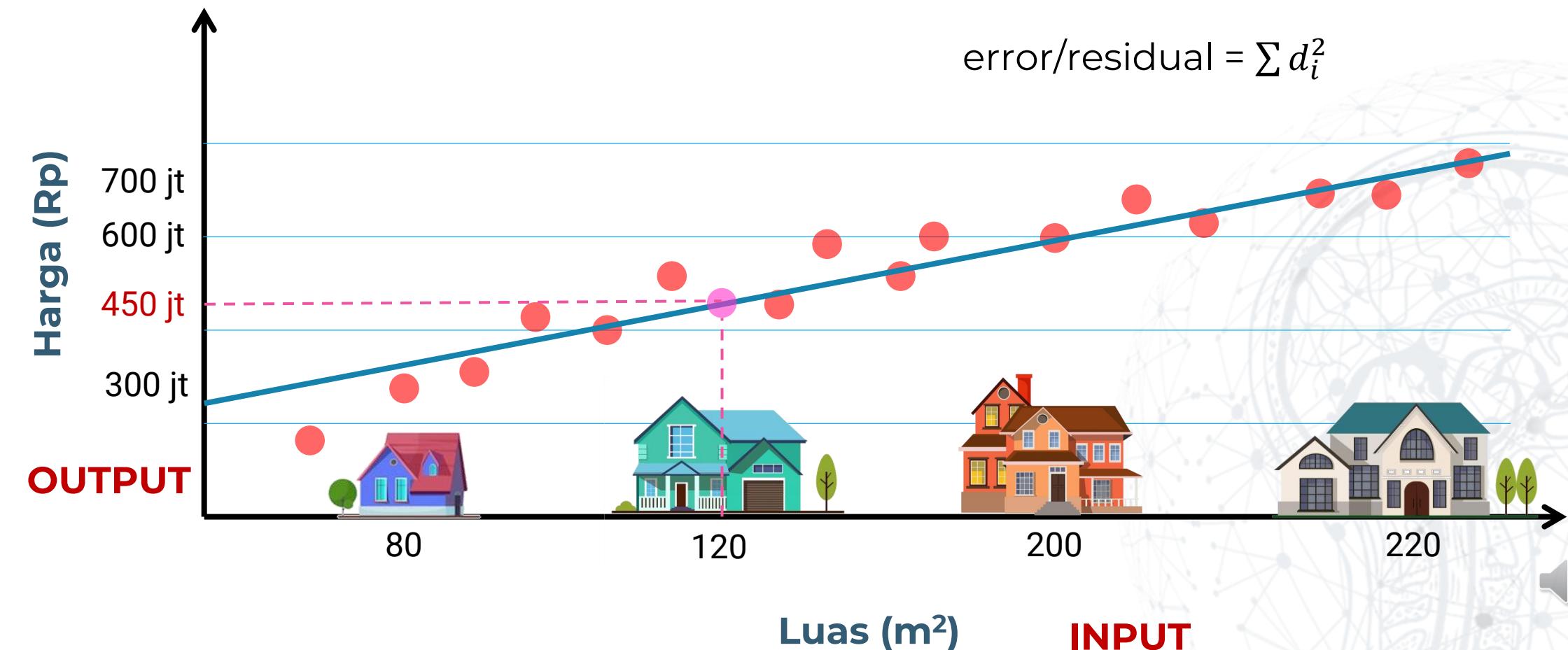
- data adalah pasangan x & y
- x = independent variable
- y = dependent variable
- $\bar{x}$  = rata-rata x
- $\bar{y}$  = rata-rata y
- n = jumlah data
- $\hat{y}$  = y hasil prediksi
- m = gradien garis
- b = intercept pada sumbu y



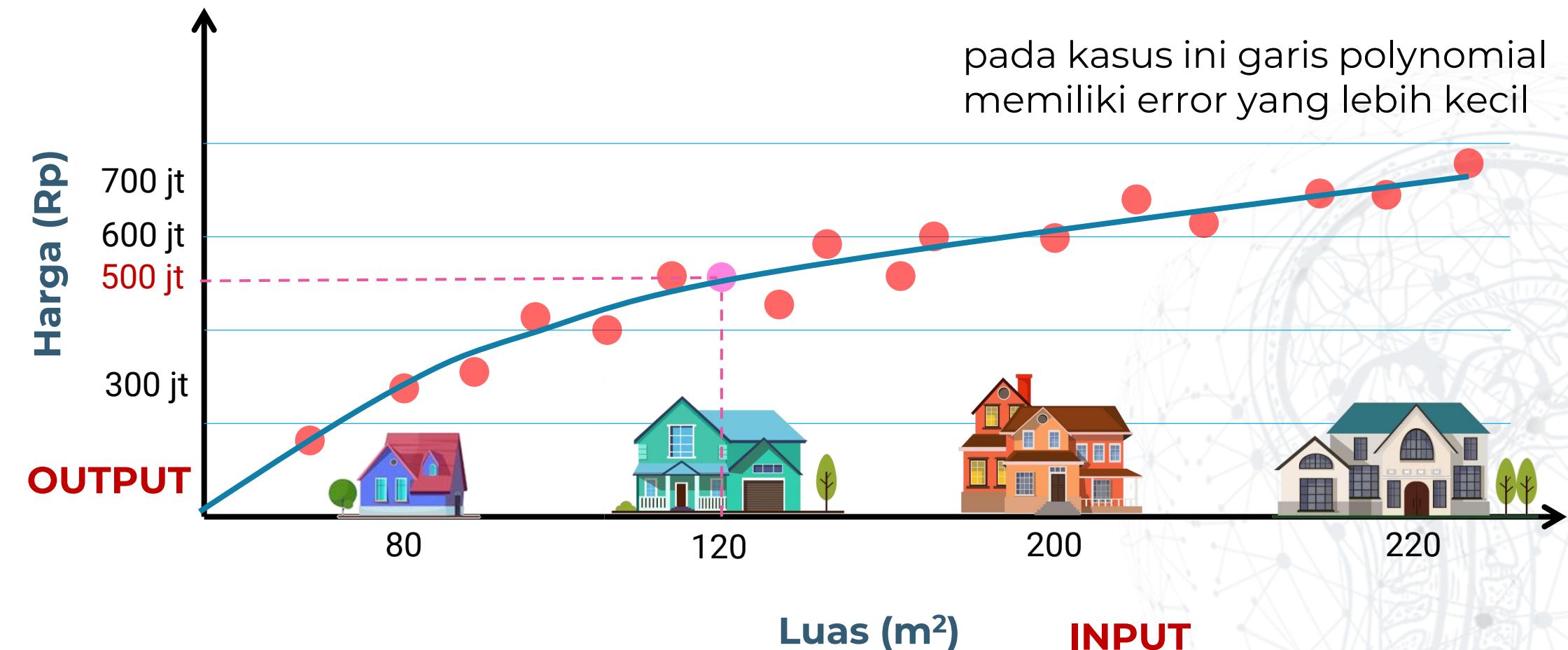
Dari persamaan garis ( $m$  dan  $b$ ) dapat dicari nilai  $\hat{y}$  jika diketahui  $x$



**Garis terbaik = garis yang memiliki  
error/residual terkecil**



## Ada kalanya garis lurus **bukan** pilihan terbaik



# Bagaimana jika dependent variable ada banyak?



**Bagaimana jika dependent variable ada banyak?**

**Matriks semakin besar**  
→ **Biaya komputasi semakin mahal**



## REGRESI dengan MACHINE LEARNING

**Semakin kompleks permasalahan yang ingin dipecahkan, maka secara komputasi akan semakin “mahal”.**

**Oleh karena itu pendekatan aproksimasi atau optimasi dengan machine learning jadi lebih “murah” dibanding pendekatan statistika**

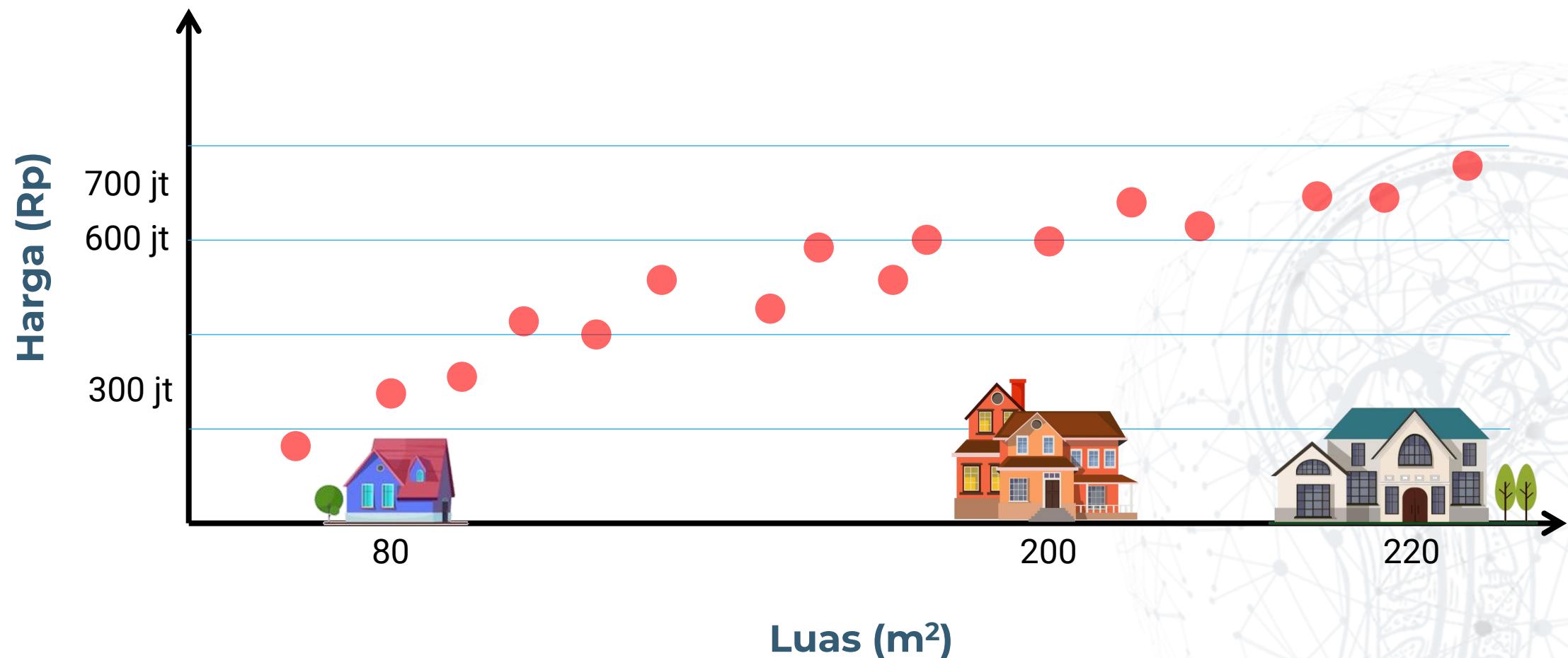


**Salah satu teknik dasar dalam melakukan optimasi dalam machine learning adalah  
GRADIENT DESCENT**



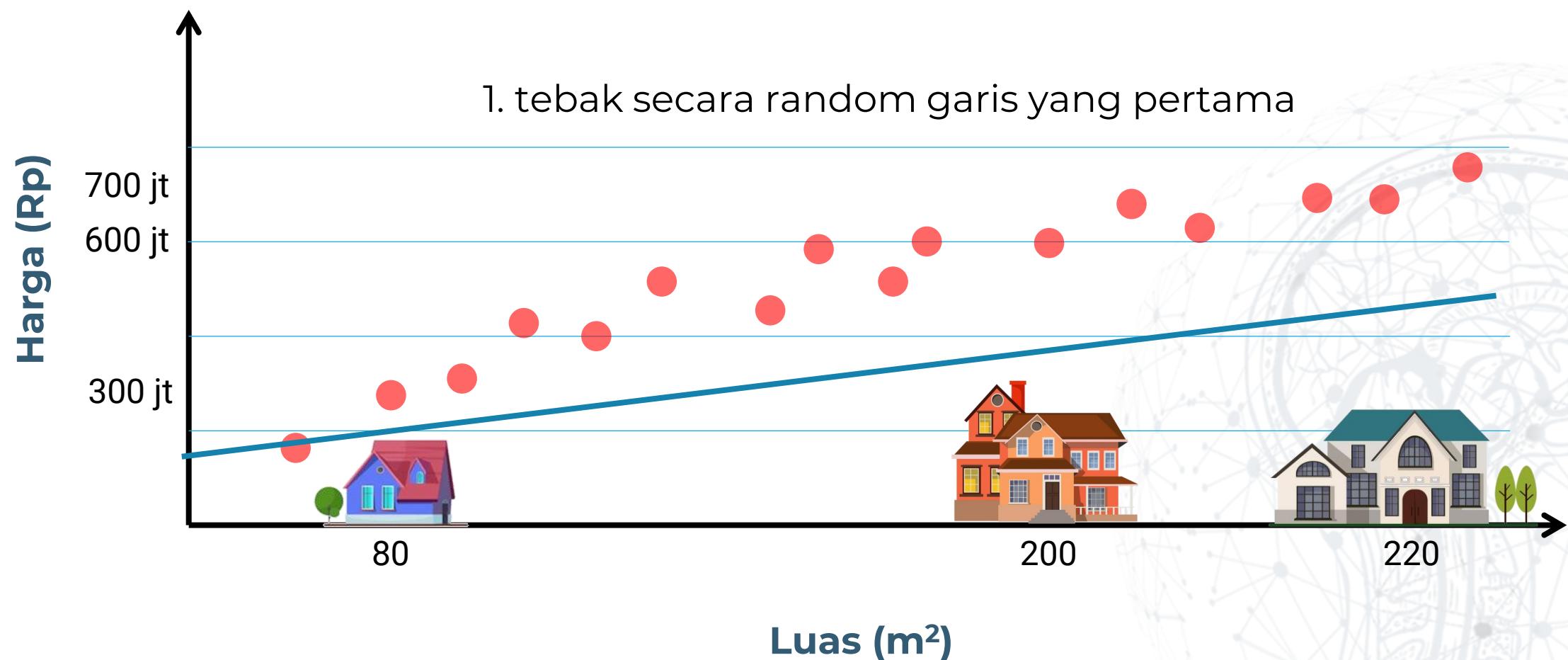
# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil



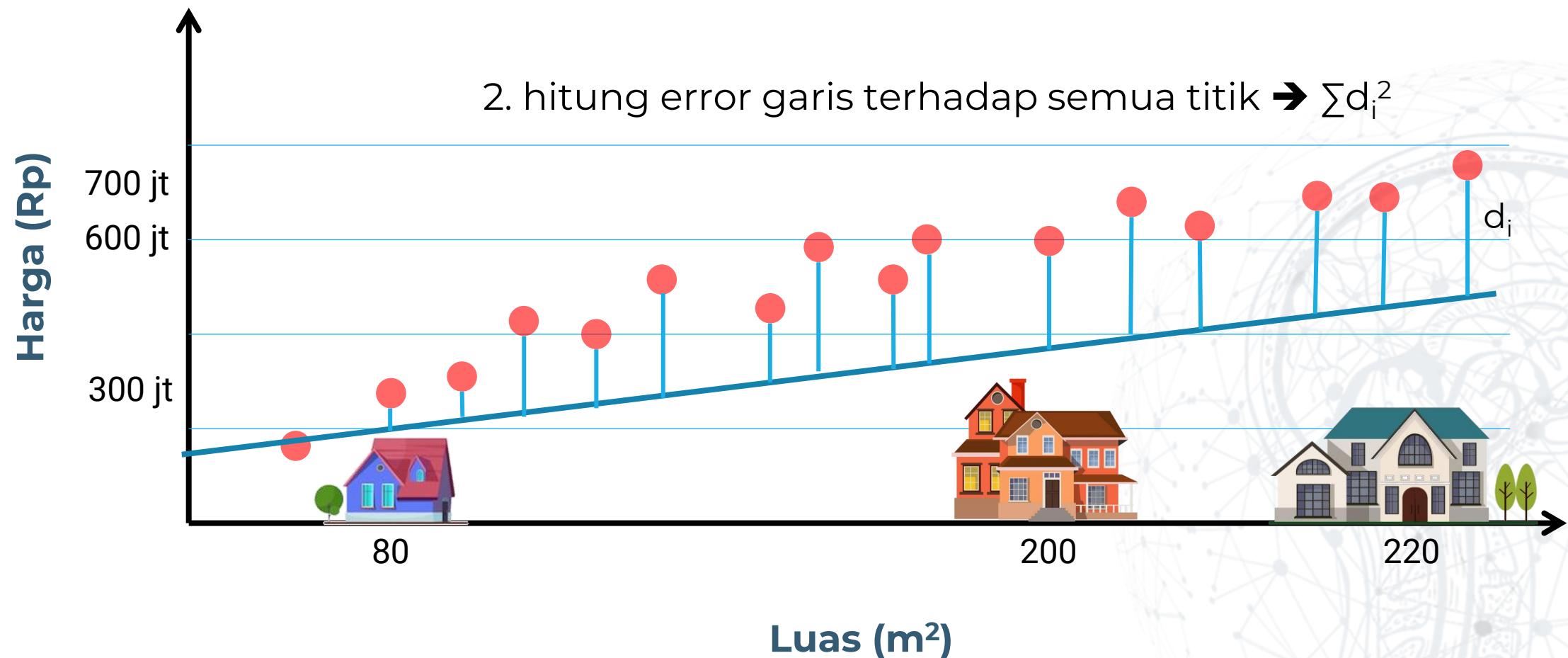
# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil



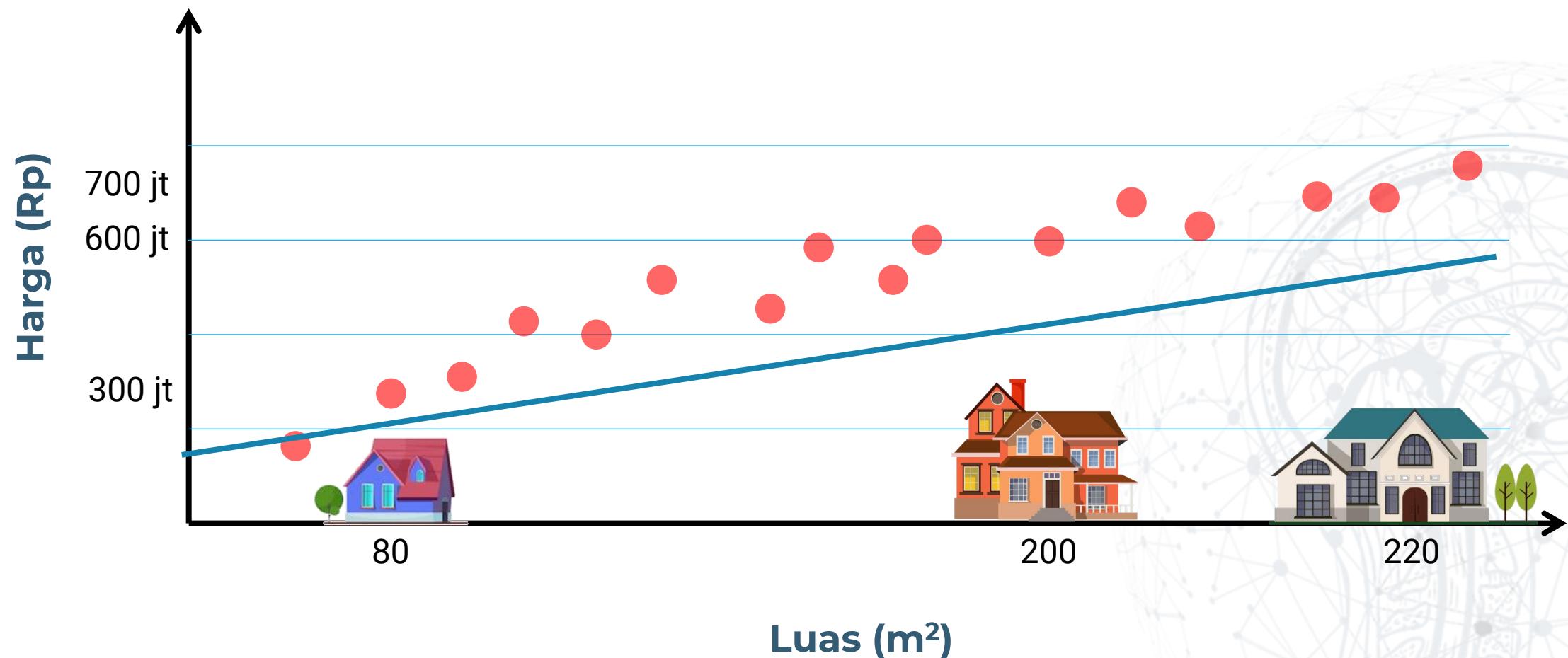
# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil



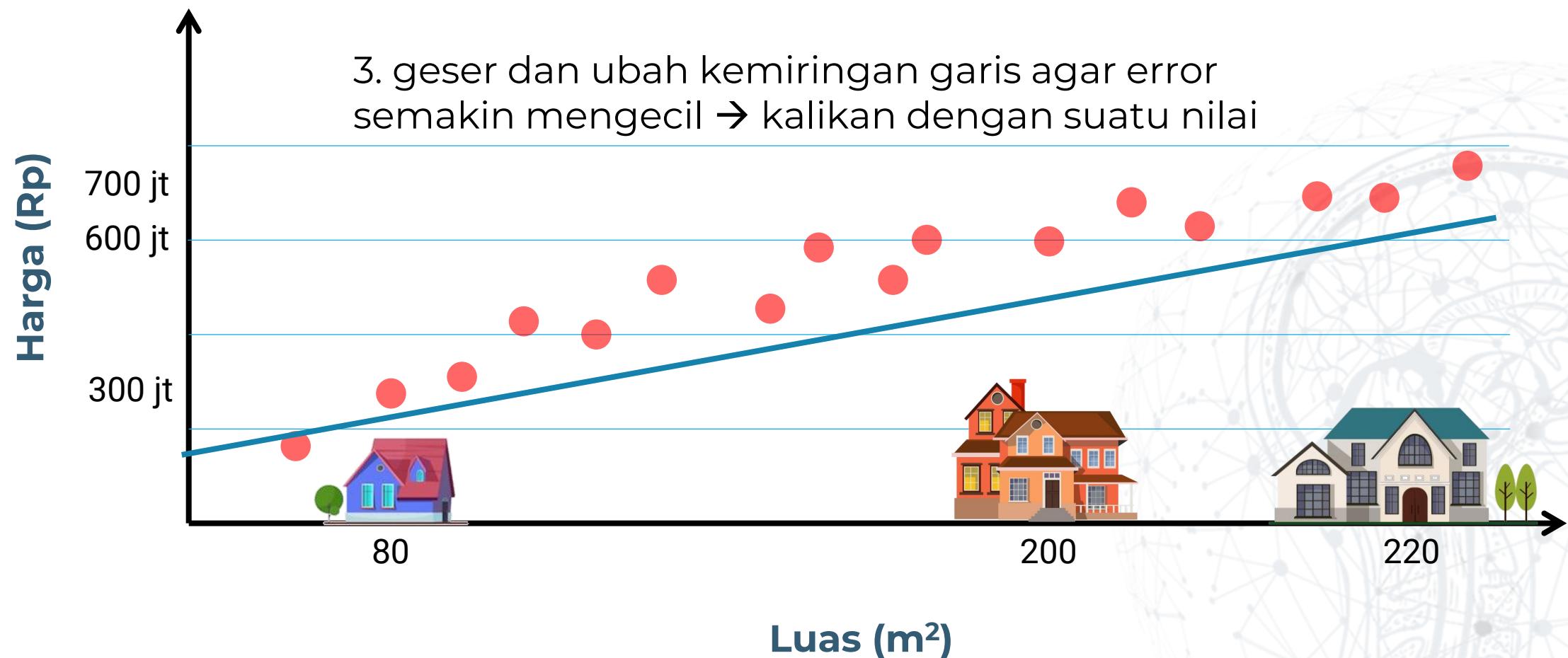
# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil



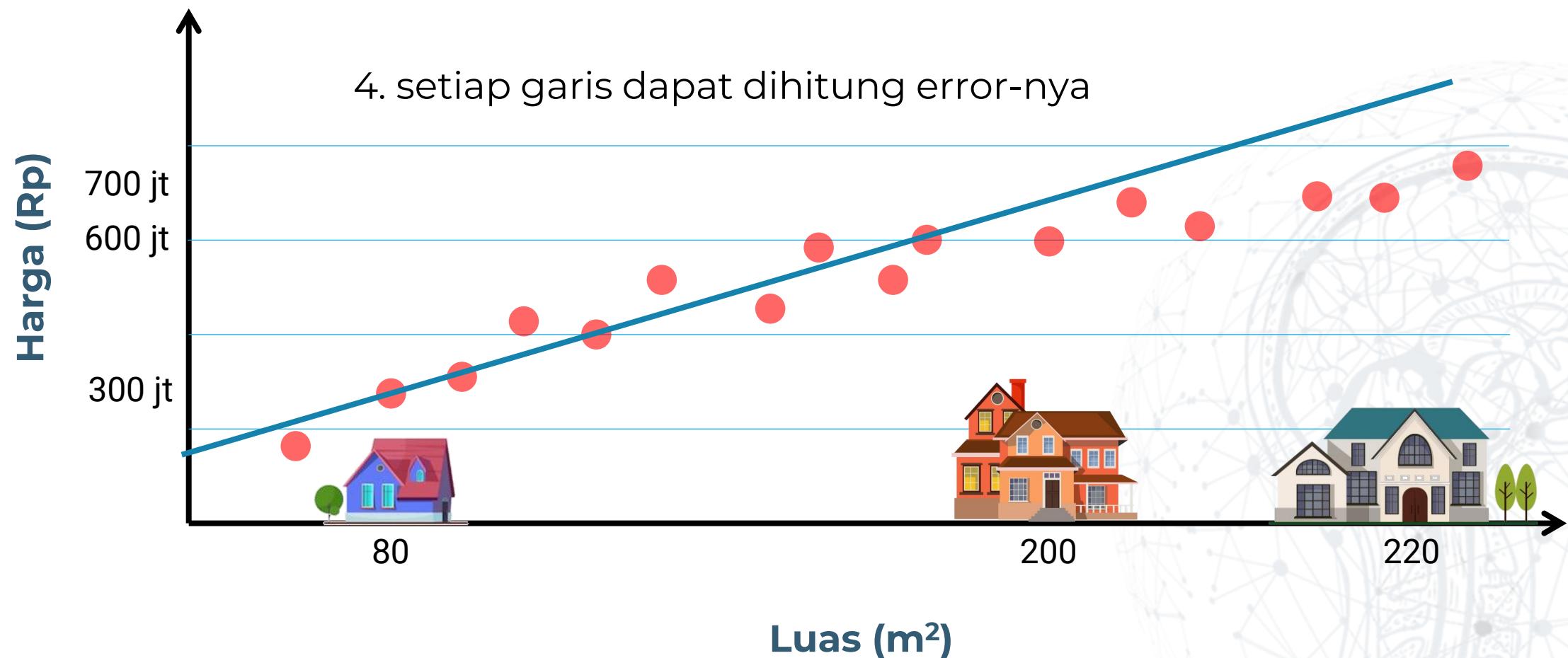
# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil



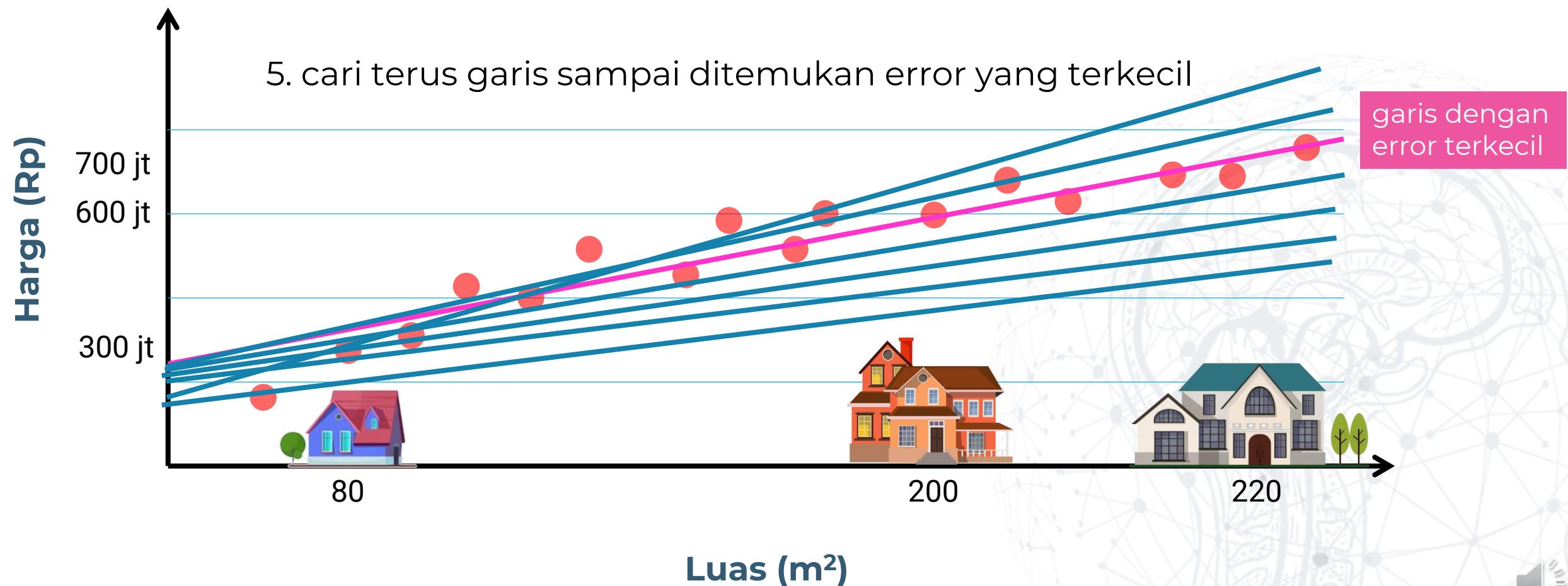
# GRADIENT DESCENT:

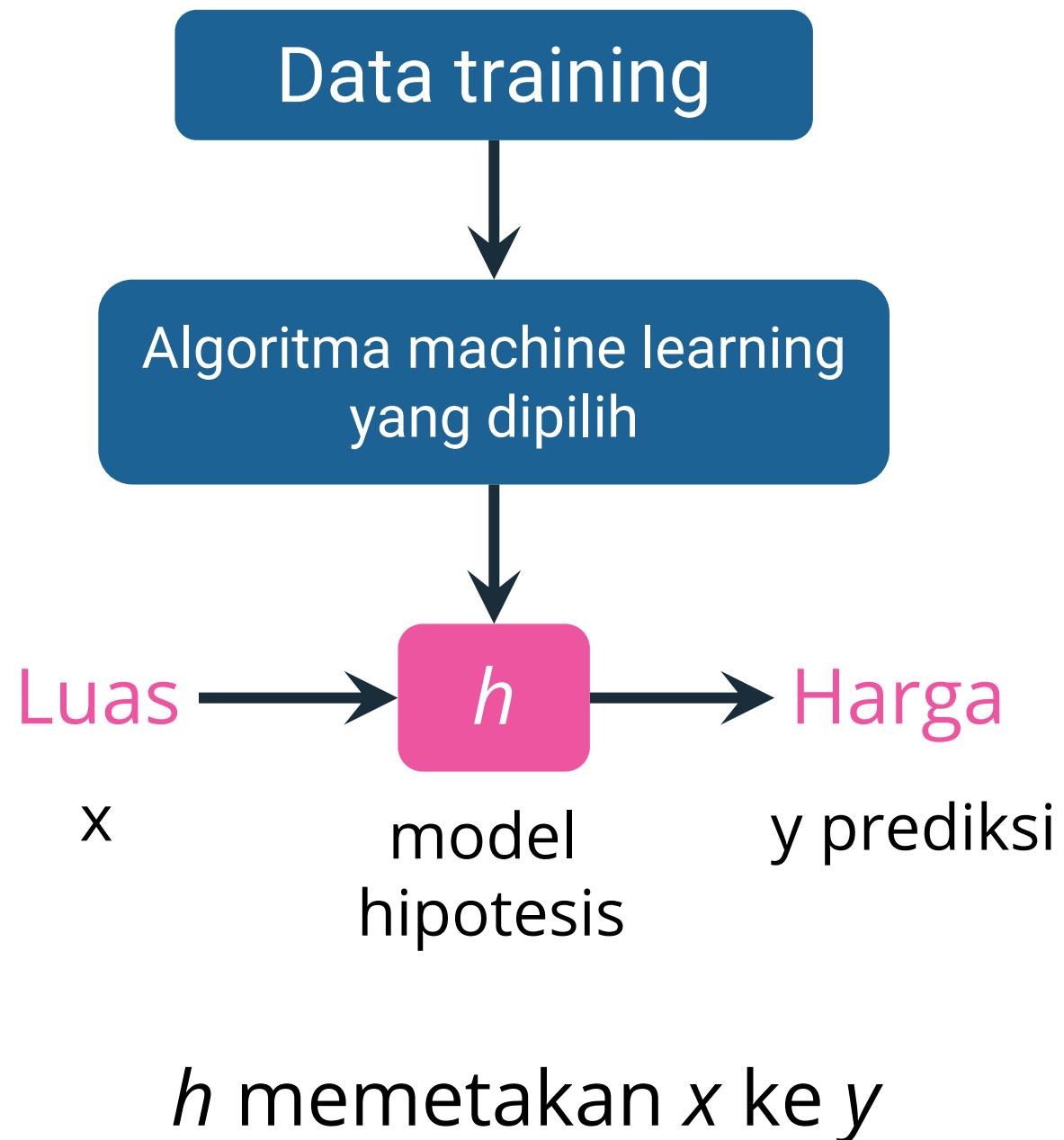
## mencari gradien berulang kali agar diperoleh error yang terkecil



# GRADIENT DESCENT:

## mencari gradien berulang kali agar diperoleh error yang terkecil





# Bagaimana menyatakan $h$ ?

tergantung dari keinginan kita  
ingin memodelkan data ke bentuk apa

## Regresi linier dengan 1 variabel

persamaan garis biasa

supaya dapat dikembangkan ke lebih  
dari 1 variabel independen

dapat juga ditulis sebagai fungsi  
hipotesis

$$\hat{y} = mx + b$$

$$\hat{y} = \theta_0 + \theta_1 x$$

$$h_{\theta}(x) = \theta_0 + \theta_1 x$$



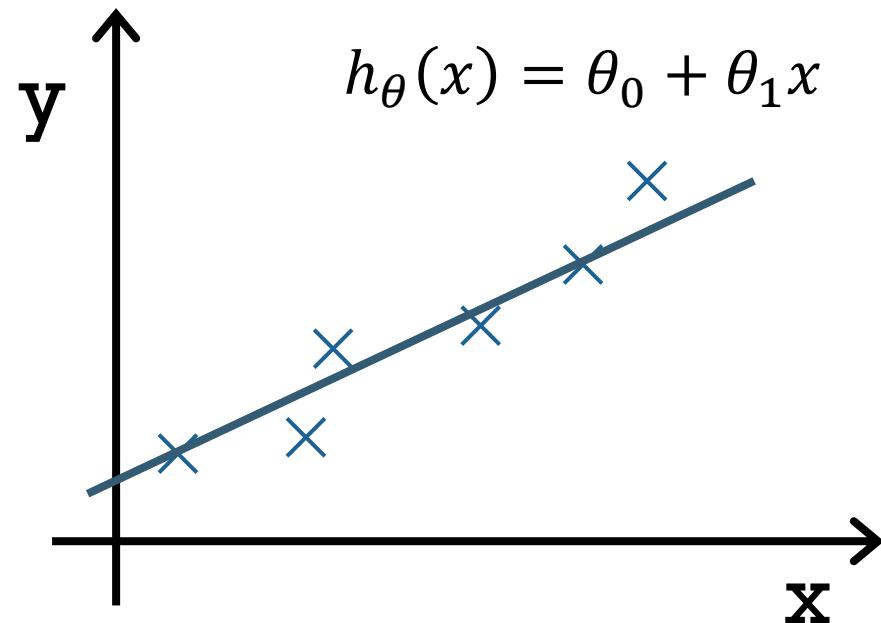
$$h_{\theta}(x) = \theta_0 + \theta_1 x$$

**Bagaimana cara mencari  $\theta_i$   
tanpa metode SLR?**



## IDE:

Cari  $\theta_0$  dan  $\theta_1$  sehingga  $h_\theta(x)$  mendekati  $y$  dari data training yang dimiliki  $(x,y)$



## cost function

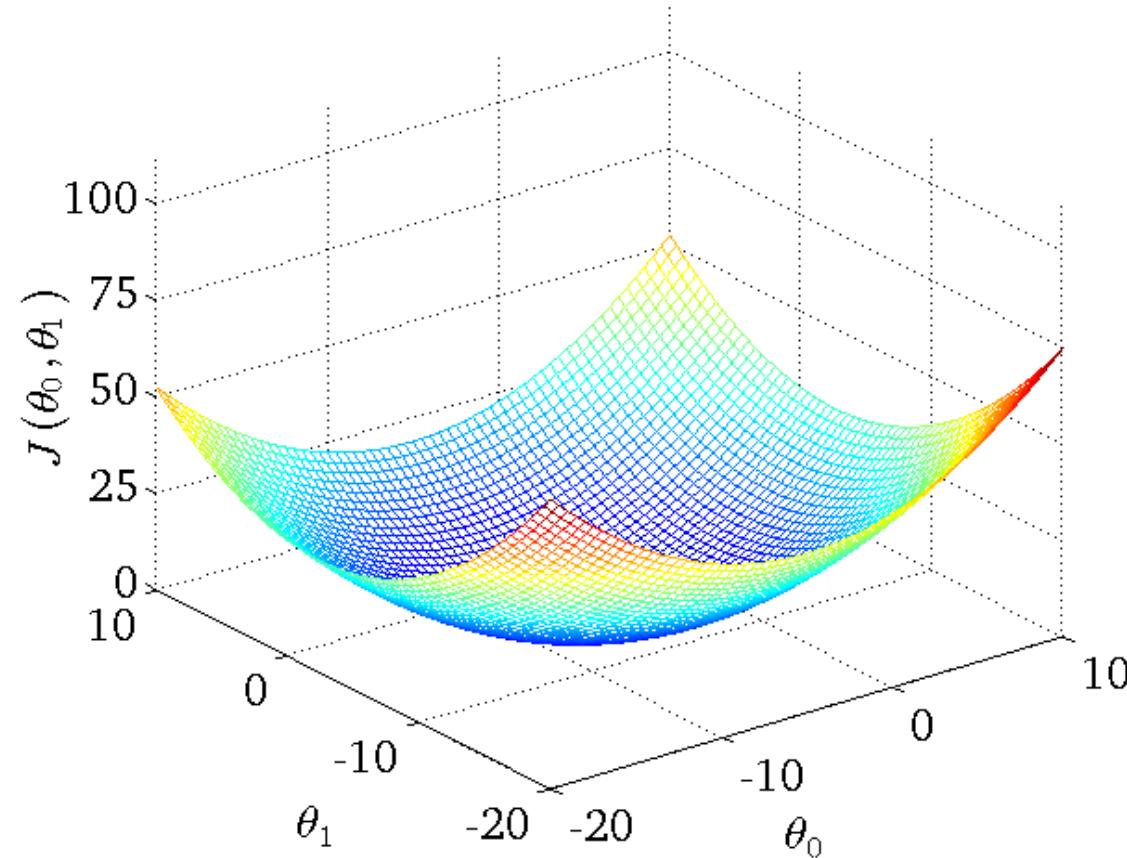
$$J(\theta_0, \theta_1) = \frac{1}{m} \sum (h_\theta(x) - y)^2$$

**Mean squared error (MSE)**



$$\min_{\theta_0 \theta_1} J(\theta_0, \theta_1)$$

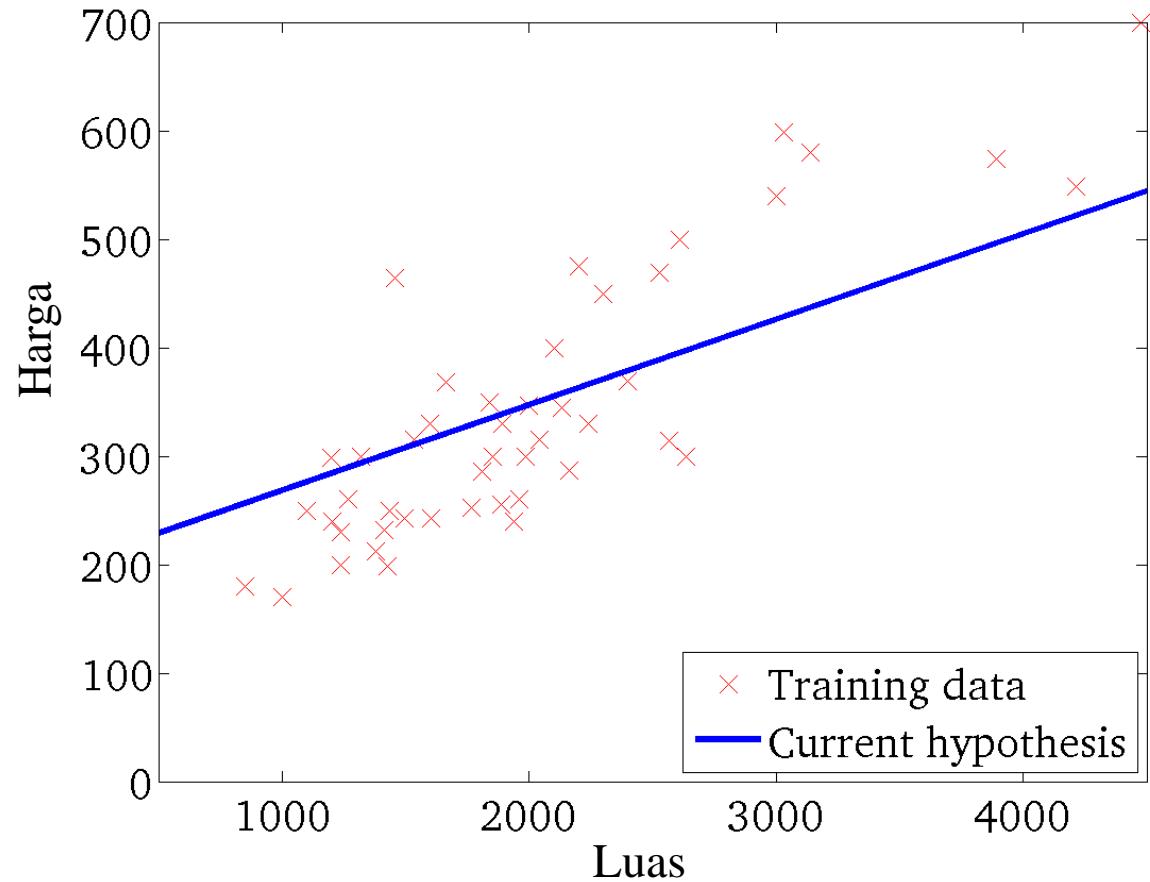




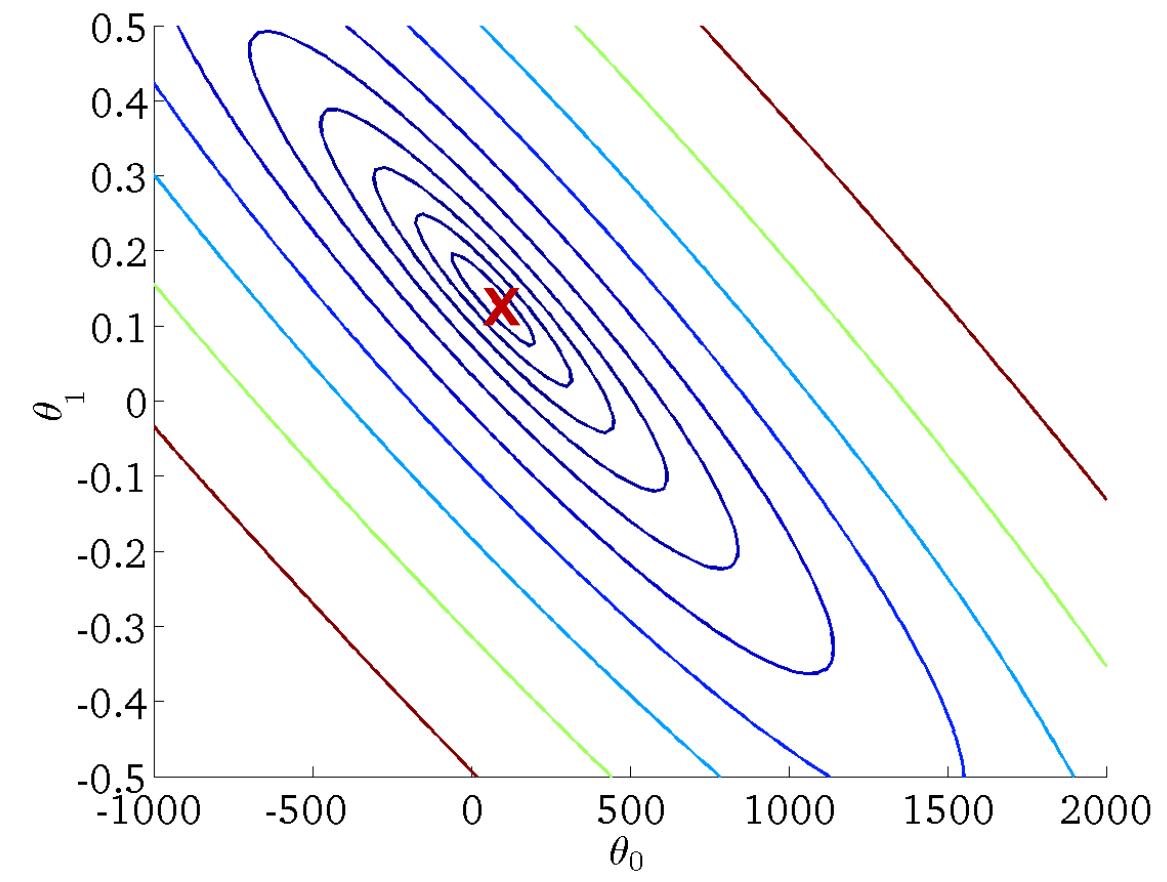
**Bagaimana cara  
mencari titik  
 $J(\theta_0, \theta_1)$   
yang paling  
minimal  
(kalau bisa 0)**



Line plot Luas vs Harga



3D mesh plot cost function  $J(\theta_0, \theta_1)$



# Algoritma dasar

- Mulai dengan nilai inisial  $\theta_0, \theta_1$
- Hitung cost:  $J(\theta_0, \theta_1) = \frac{1}{m} \sum (h_{\theta}(x) - y)^2$
- Ubah  $\theta_0, \theta_1$
- Lakukan sampai didapatkan  $\min_{\theta_0 \theta_1} J(\theta_0, \theta_1)$



# ALGORITMA GRADIENT DESCENT

ulangi sampai konvergen: {

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

Learning rate

untuk  $j = 0$  dan  $j = 1$

assignment  
a:=b  
↑

ulangi sampai konvergen: {

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum (h_\theta(x) - y)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum (h_\theta(x) - y) \cdot x$$

}

$$\frac{d}{d\theta_0} \cdot J(\theta_0, \theta_1)$$

}

$\theta_0$  dan  $\theta_1$  harus diupdate secara simultan

$$\frac{d}{d\theta_1} \cdot J(\theta_0, \theta_1)$$



**Jika independent variabel banyak**

**implementasikan algoritma ke dalam  
program dengan bentuk matriks**



# Bagaimana cara menentukan learning rate $\alpha$ ?



**Tidak ada panduan baku untuk memilih  $\alpha$**

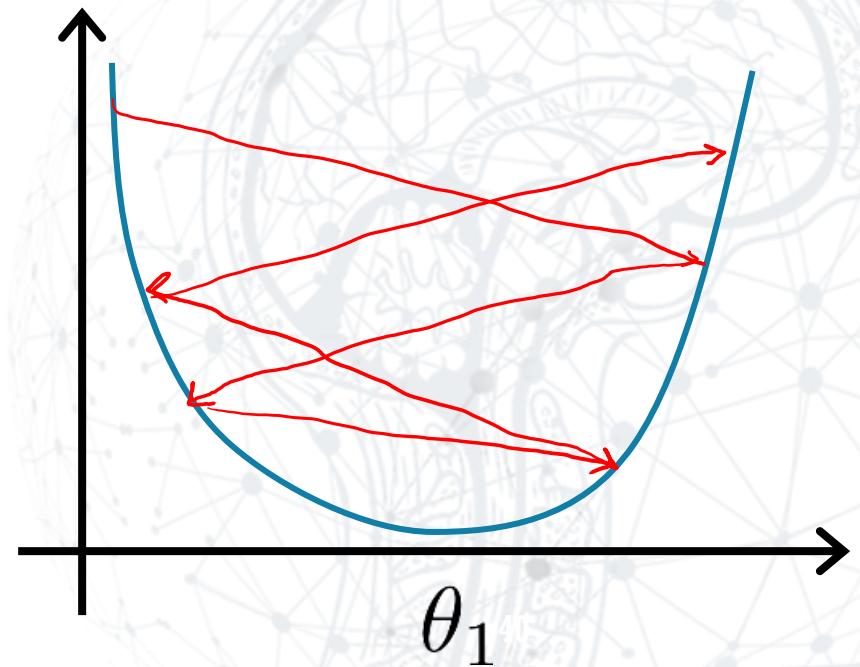
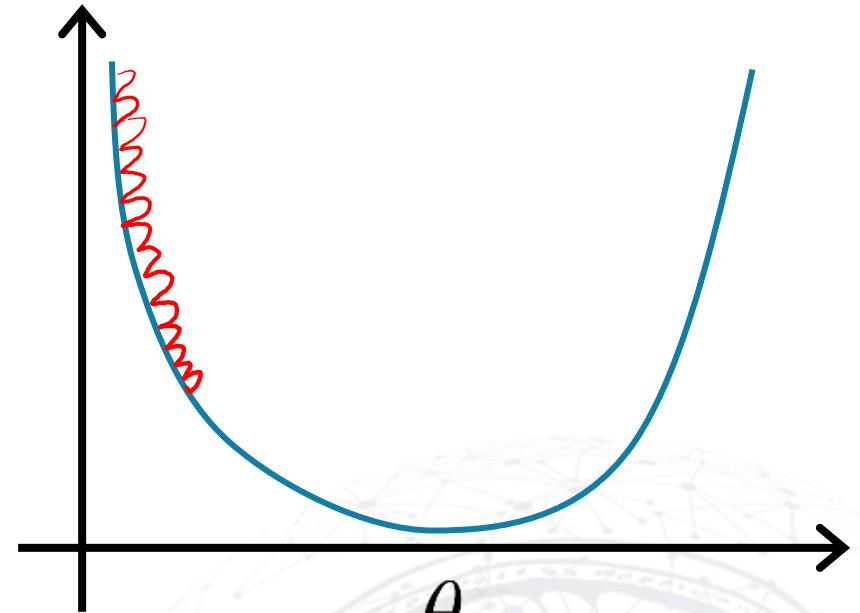
$\alpha \rightarrow$  hyperparameter

**ditebak dan dicari secara heuristik (coba-coba) atau menggunakan metode optimasi**

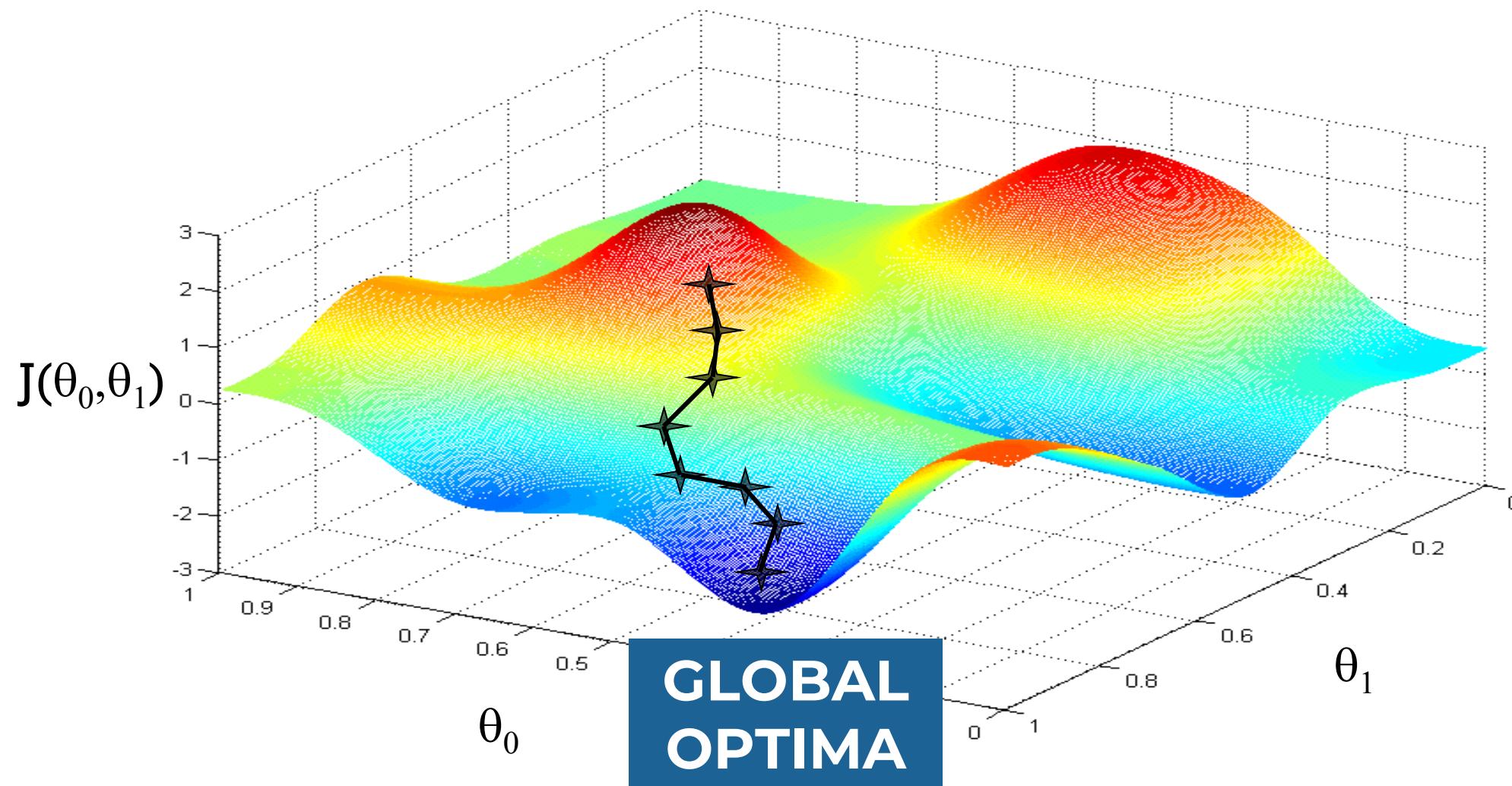


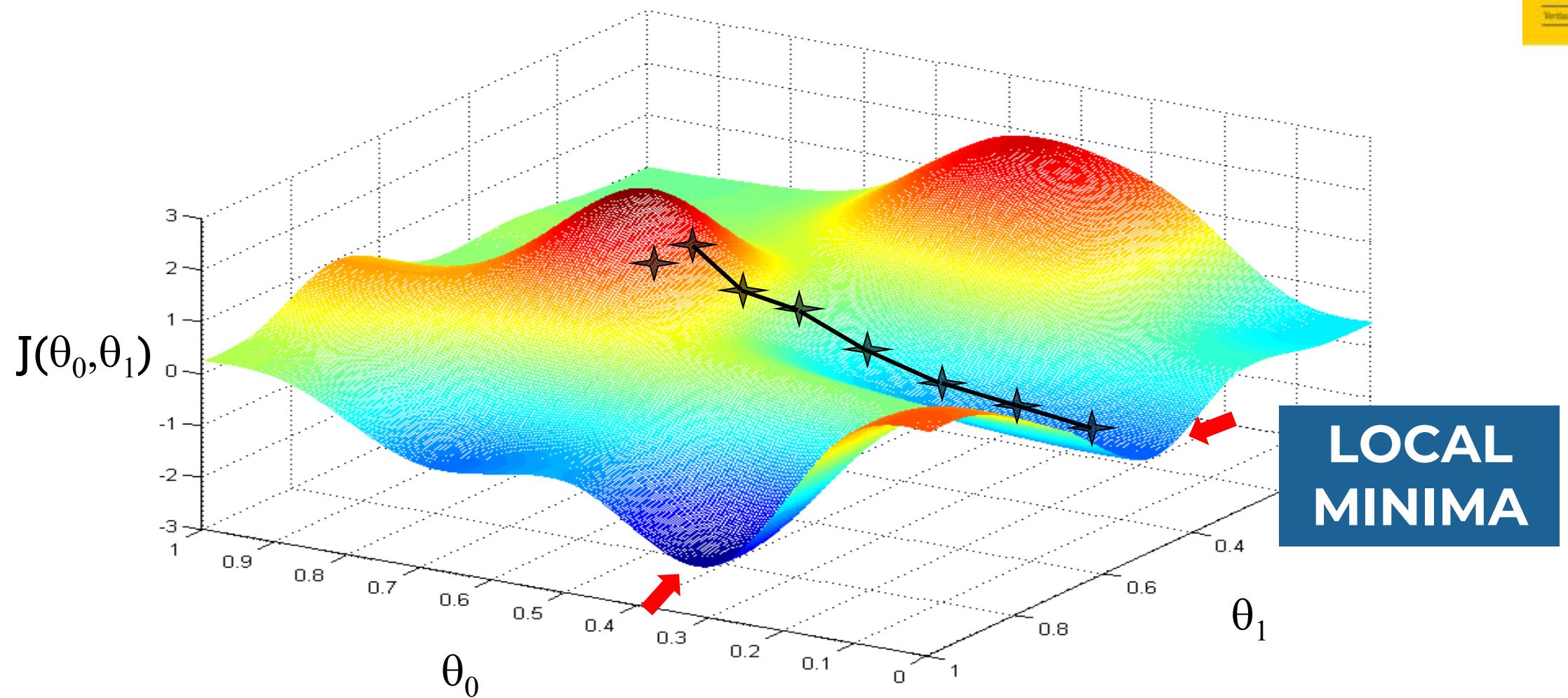
Jika  **$\alpha$  terlalu kecil**, proses gradient descent akan terlalu lama → error hanya sedikit turunnya di setiap epoch

Jika  **$\alpha$  terlalu besar** gradient descent bisa jadi overshoot local minima. Bisa jadi gagal mencapai konvergen (atau bahkan menjadi divergen).



Sampai kapan  
pembelajaran  
harus dilakukan?





Oleh karena itu biasanya dalam machine learning yang diawali dengan **inisialisasi random** maka harus **diulang beberapa kali** dengan harapan akan diperoleh local minima yang mendekati global optima

# EPOCH

**1 epoch =  
1 kali pembelajaran untuk satu set data  
training**

100 epoch = 100 kali belajar

**Proses pembelajaran dapat dihentikan jika sudah mencapai epoch tertentu atau sampai diperoleh loss/error di bawah standar yang ditetapkan**

## STOPPING CRITERIA

# Stochastic Gradient Descent

Nilai  $\theta_j$  di-update untuk setiap pasangan data training x dan y.  
Juga disebut incremental gradient descent

# Batch Gradient Descent

Nilai  $\theta_j$  di-update ketika sudah belajar 1 set data (1 epoch)

**Mini batch** sekelompok data digunakan dalam 1x iterasi, misal batch\_size = 16 artinya 16 buah data digunakan untuk update  $\theta_j$



Bagaimana cara  
implementasinya  
ke dalam program  
Python  
?

## Kesimpulan

- Pada regresi, mesin akan memprediksi nilai kontinyu
- Metode gradient descent pada regresi machine learning dilakukan dengan mencari parameter garis yang memiliki error minimum
- 1 set pembelajaran disebut epoch
- Metode gradient descent: stochastic, batch, mini batch



Selanjutnya:  
Klasifikasi



# KECERDASAN BUATAN

## 5 | CLASSIFICATION

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI

# Machine Learning - Classification

In Machine Learning, classification is a **supervised** learning approach, which can be thought of as a means of **categorizing** or "classifying" some unknown items into a discrete set of "classes."

Classification attempts to learn the relationship between a set of **feature** variables and a **target** variable of interest.

The target attribute in classification is a **categorical variable with discrete values**.



# How does classification work?

**Classification** determines the class label for an unlabeled test case.

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
41	3	17	12	176	9.3	11.359	5.009	1
27	1	10	6	31	17.3	1.362	4.001	0
40	1	15	14	55	5.5	0.856	2.169	0
41	1	15	14	120	2.9	2.659	0.821	0
24	2	2	0	28	17.3	1.787	3.057	1
41	2	5	5	25	10.2	0.393	2.157	0
39	1	20	9	67	30.6	3.834	16.668	0
43	1	12	11	38	3.6	0.129	1.239	0
24	1	3	4	19	24.4	1.358	3.278	1
36	1	0	13	25	19.7	2.778	2.147	0

Categorical Variable

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
37	2	16	10	130	9.3	10.23	3.21	

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
41	3	17	12	176	9.3	11.359	5.009	1
27	1	10	6	31	17.3	1.362	4.001	0
40	1	15	14	55	5.5	0.856	2.169	0
41	1	15	14	120	2.9	7.659	0.821	0
24	2	2	0	28	17.3	1.787	3.057	1
41	2	5	5	25	10.2	0.393	2.157	0
39	1	20	9	67	30.6	3.834	16.668	0
43	1	12	11	38	3.6	0.129	1.239	0
24	1	3	4	19	24.4	1.358	3.278	1
35					---	2.778	2.147	1

# Machine Learning - Classification

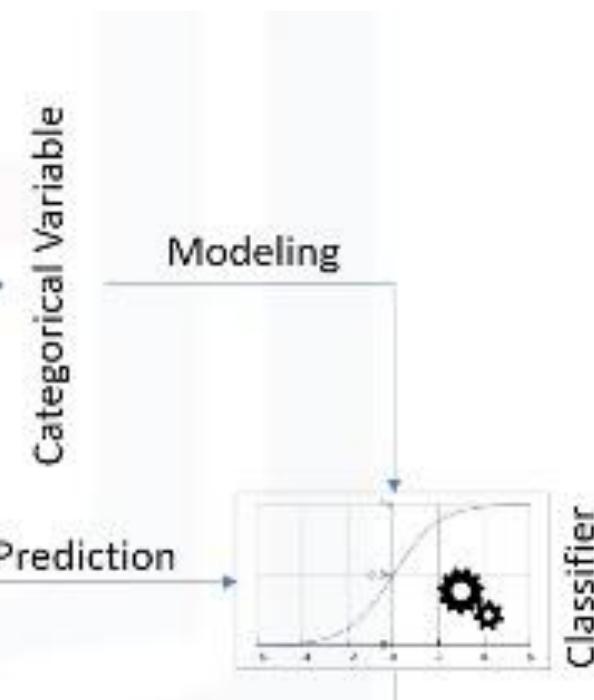
- Let's explain this with an example. A good sample of classification is the loan default prediction. Suppose a bank is concerned about the potential for loans not to be repaid.
- If previous loan default data can be used to predict which customers are likely to have problems repaying loans, these "bad risk" customers can either have their loan application declined or offered alternative products.

# Machine Learning - Classification

The goal of a loan default predictor is to use existing loan default data, which is information about the customers (such as age, income, education, etc.).

To build a classifier, pass a new customer or potential future defaulter to the model, and then label it (i.e. the data points) as "Defaulter" or "Not Defaulter", or for example, 0 or 1.

age	ed	employ	address	income	debtinc	creddebt	othdebt	default
41	3	17	12	176	9.3	11.359	5.009	1
27	1	10	6	31	17.3	1.362	4.001	0
40	1	15	14	55	5.5	0.856	2.169	0
41	1	15	14	120	2.9	2.659	0.821	0
24	2	2	0	28	17.3	1.787	3.057	1
41	2	5	5	25	10.2	0.393	2.157	0
39	1	20	9	67	30.6	3.834	16.668	0
43	1	12	11	38	3.6	0.129	1.239	0
24	1	3	4	19	24.4	1.358	3.278	1
36	1	0	13	25	19.7	2.778	2.147	0



age	ed	employ	address	income	debtinc	creddebt	othdebt	default
37	2	16	10	130	9.3	10.23	3.21	0

# Machine Learning - Classification

This is how a classifier predicts an unlabeled test case.  
Please notice that this specific example was about a binary classifier with two values.

We can also build classifier models for both binary classification and multi-class classification.

For example, imagine that you collected data about a set of patients, all of whom suffered from the same illness.

Age	Sex	BP	Cholesterol	Na	K	Drug
23	F	HIGH	HIGH	0.793	0.031	drugY
47	M	LOW	HIGH	0.739	0.056	drugC
47	M	LOW	HIGH	0.697	0.069	drugC
28	F	NORMAL	HIGH	0.564	0.072	drugX
61	F	LOW	HIGH	0.559	0.031	drugY
22	F	NORMAL	HIGH	0.677	0.079	drugX
49	F	NORMAL	HIGH	0.79	0.049	drugY
41	M	LOW	HIGH	0.767	0.069	drugC
60	M	NORMAL	HIGH	0.777	0.051	drugY
43	M	LOW	NORMAL	0.526	0.027	drugY

# Machine Learning - Classification

During their course of treatment, each patient responded to one of three medications.

You can use this labeled dataset, with a classification algorithm, to build a classification model.

Then you can use it to find out which drug might be appropriate for a future patient with the same illness.

Age	Sex	BP	Cholesterol	Na	K	Drug
23	F	HIGH	HIGH	0.793	0.031	drugY
47	M	LOW	HIGH	0.739	0.056	drugC
47	M	LOW	HIGH	0.697	0.069	drugC
28	F	NORMAL	HIGH	0.564	0.072	drugX
61	F	LOW	HIGH	0.559	0.031	drugY
22	F	NORMAL	HIGH	0.677	0.079	drugX
49	F	NORMAL	HIGH	0.79	0.049	drugY
41	M	LOW	HIGH	0.767	0.069	drugC
60	M	NORMAL	HIGH	0.777	0.051	drugY
43	M	LOW	NORMAL	0.526	0.027	drugY

Categorical Variable

Modeling

Age	Sex	BP	Cholesterol	Na	K	Drug
36	F	LOW	HIGH	0.697	0.069	Drug DrugX

Prediction



Classifier

Predicted Labels

# Classification Use Case

As you can see, it is a sample of multi-class classification.

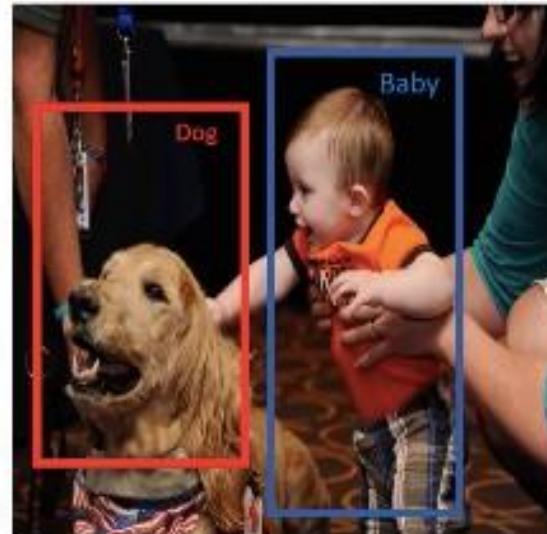
Classification has different business use cases as well, e.g.:

- To predict the category to which a customer belongs;
- For Churn detection, where we predict whether a customer switches to another provider or brand;
- Or to predict whether or not a customer responds to a particular advertising campaign.

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	Yes
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	Yes
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	No
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	No
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	?

# Classification Use Case

- Data classification has several applications in a wide variety of industries.
- Essentially, many problems can be expressed as associations between feature and target variables, especially when labeled data is available.
- This provides a broad range of applicability for classification.
- For example, classification can be used for email filtering, speech recognition, handwriting recognition, bio- metric identification, document classification, and much more.



# Popular Classification Algorithms

Decision Trees

Naïve Bayes

K-nearest  
neighbor

Logistic  
regression

Support Vector  
Machines

Neural  
Networks





UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# K-Nearest Neighbor (KNN)

# K-Nearest Neighbors

Imagine that a telecommunications provider has segmented its customer base by service usage patterns, categorizing the customers into four groups.

	X: Independent variable											Y: Dependent variable	
	region	age	marital	address	income	ed	employ	retire	gender	reside	custcat		
0	2	44	1	9	64	4	5	0	0	2	1		
1	3	33	1	7	135	5	5	0	0	8	4		
2	3	52	1	24	118	1	29	0	1	2	3		
3	2	33	0	12	33	2	0	0	1	1	1		
4	2	30	1	8	30	1	2	0	0	4	3		
5	2	39	0	17	79	2	16	0	1	1	3		
6	3	22	1	2	19	2	4	0	1	5	2		
7	2	35	0	5	76	2	10	0	0	3	4		
8	3	50	1	7	168	4	31	0	0	5	?		

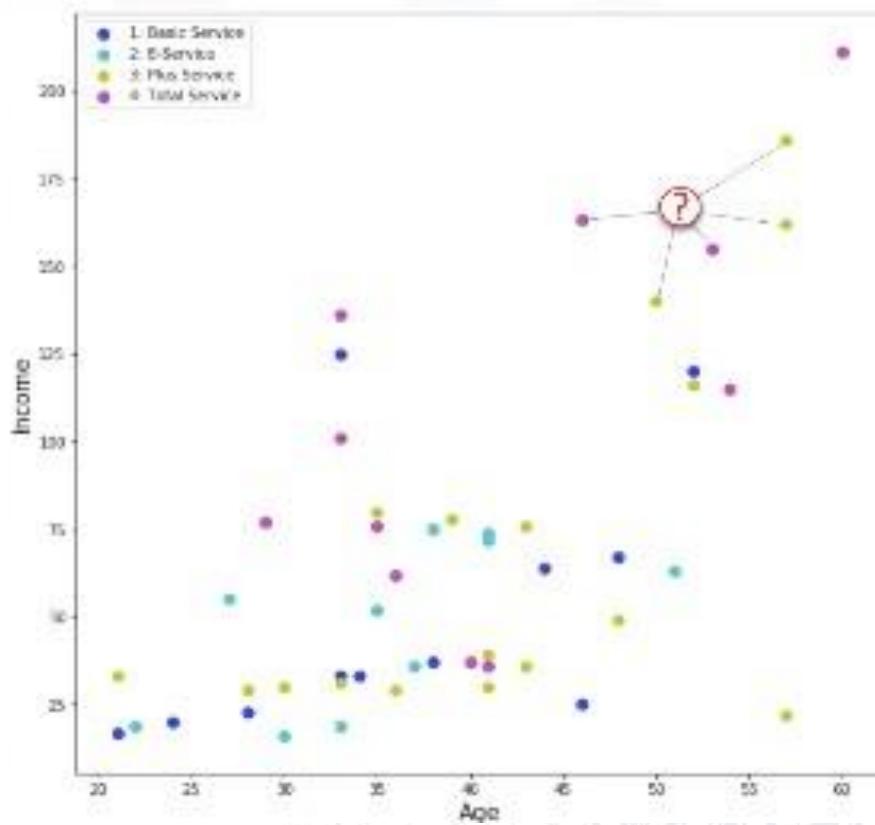
Value      Label

- 1      Basic Service
- 2      E-Service
- 3      Plus Service
- 4      Total Service

Our objective is to build a classifier, for example using the rows 0 to 7, to predict the class of row 8. We will use a specific type of classification called K-nearest neighbor.

# K-Nearest Neighbors

- A method for **classifying** cases based on their similarity to other cases
- Cases that are near each other are said to be “**neighbors**”
- Based on similar cases with same class labels are near each other



Let's use only two fields as predictors - specifically, Age and Income, and then plot the customers based on their group membership.

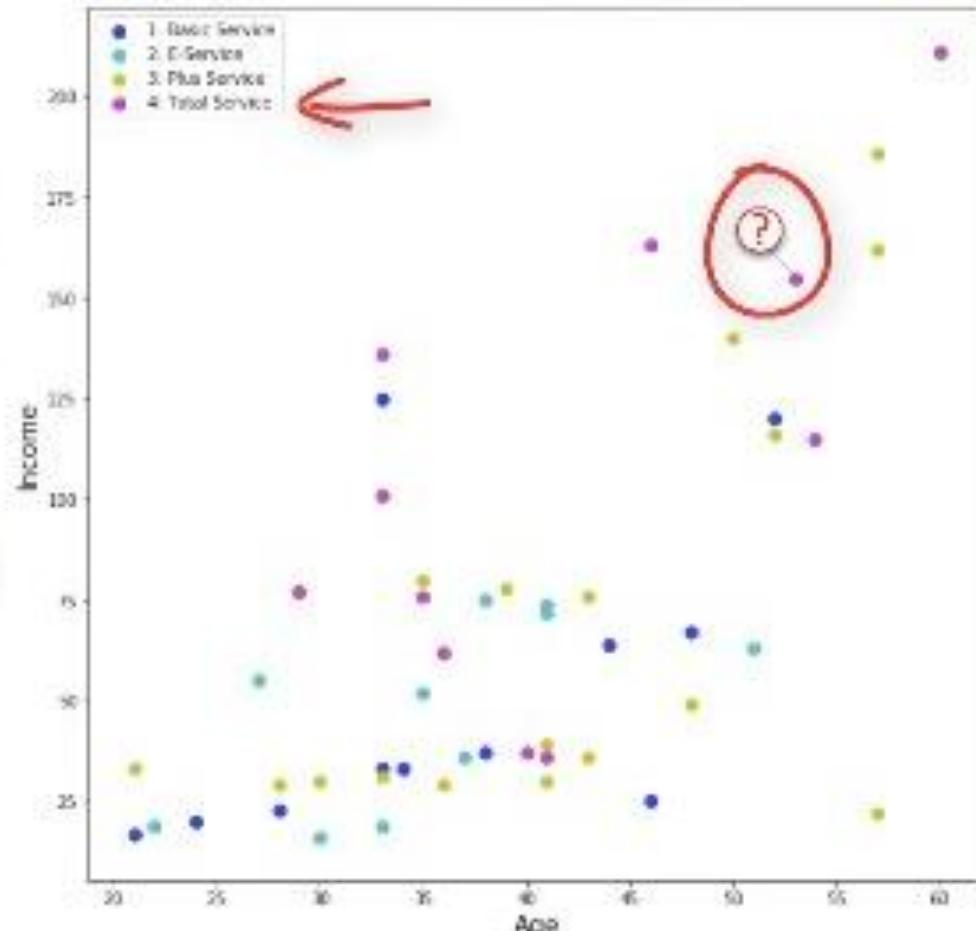
	region	age	marital	address	income	ed	employ	retire	gender	reside	custcat
0	2	44	1	8	64	4	5	0	0	2	1
1	3	33	1	7	136	5	5	0	0	6	4
2	3	52	1	24	118	1	29	0	1	2	3
3	2	32	0	12	33	2	0	0	1	1	1
4	2	30	1	9	30	1	2	0	0	4	3
5	2	38	0	17	78	2	16	0	1	1	3
6	3	22	1	2	19	2	4	0	1	6	2
7	2	35	0	5	78	2	10	0	0	3	4
8	3	50	1	2	168	4	31	0	0	5	?

Now, let's say that we have a new customer, for example, record number 8 with a known age and income. How can we find the class of this customer?

Can we find one of the closest cases and assign the same class label to our new customer?

Can we also say that the class of our new customer is most probably group 4 (i.e. total service) because its nearest neighbor is also of class 4?

	region	age	marital	address	income	ed	employ	retire	gender	reside	custcat
0	2	44	1	8	64	4	5	0	0	2	1
1	3	32	1	7	136	5	5	0	0	6	4
2	3	52	1	24	118	1	29	0	1	2	3
3	2	33	0	12	33	2	0	0	1	1	1
4	2	30	1	8	30	1	2	0	0	4	3
5	2	38	0	17	78	2	18	0	1	1	3
6	3	22	1	2	19	2	4	0	1	5	2
7	2	35	0	5	28	2	10	0	0	3	4
8	3	50	1	7	168	4	31	0	0	5	?

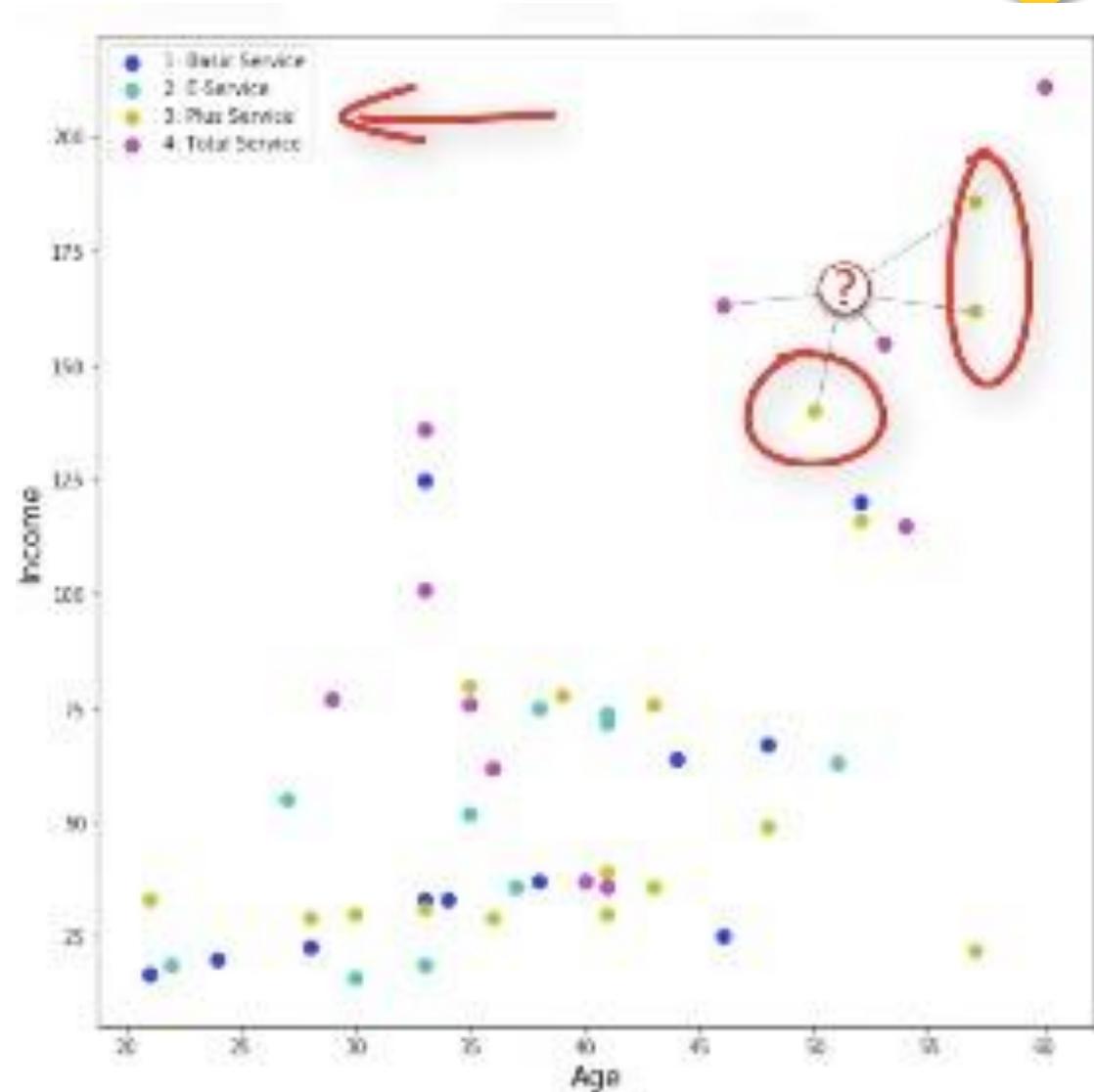


Yes, we can. In fact, it is the first-nearest neighbor.

It might be a poor judgment, especially if the first nearest neighbor is a very specific case, or an outlier.

What if we chose the five nearest neighbors, and did a majority vote among them to define the class of our new customer?

- 3 yellow and 2 magenta
- yellow wins → class = 3



# KNN Algorithm

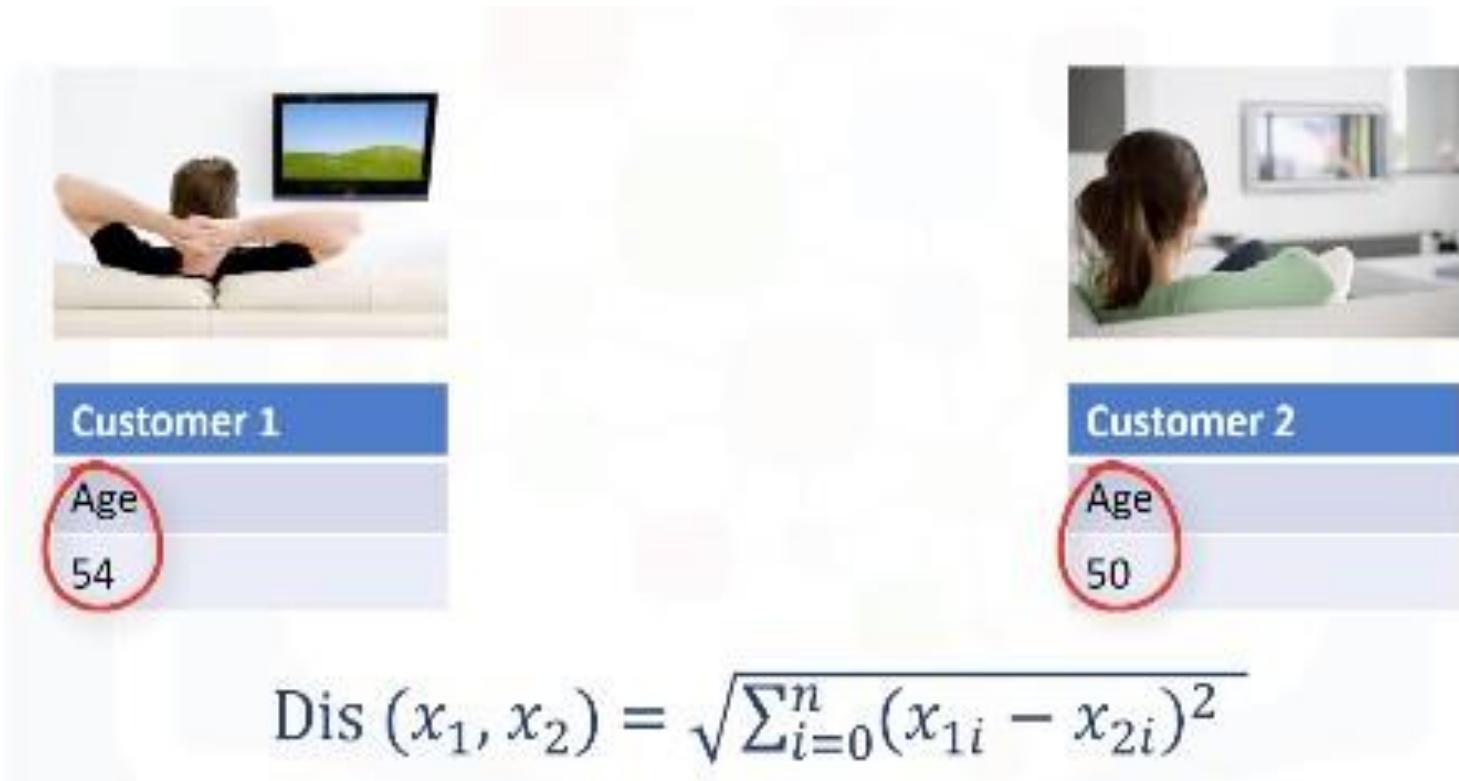


In a classification problem, the k-nearest neighbors algorithm works as follows:

1. Pick a value for K.
2. Calculate the distance from the new case (holdout from each of the cases in the dataset).
3. Search for the K observations in the training data that are 'nearest' to the measurements of the unknown data point.
4. predict the response of the unknown data point using the most popular response value from the K nearest neighbors.

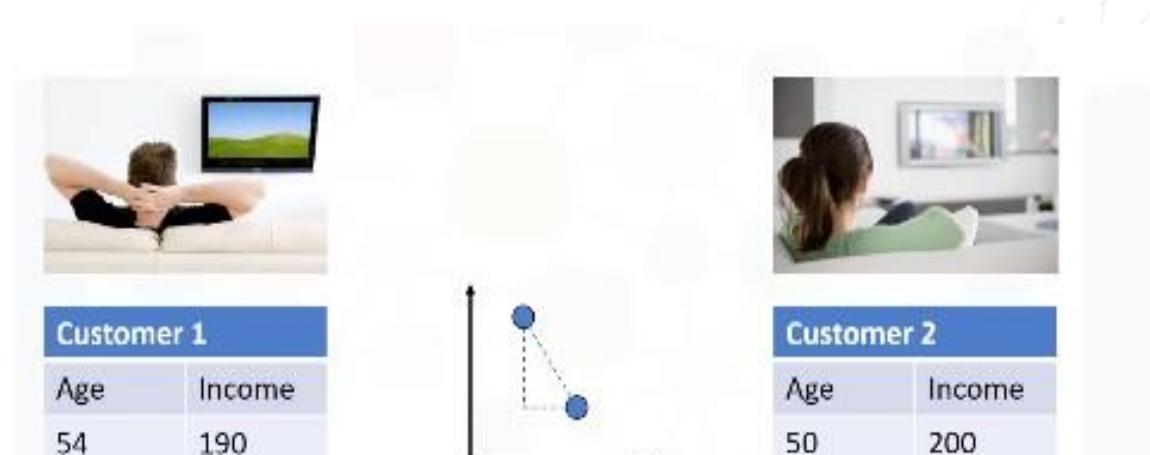
# Key questions

- How to select the correct K
- How to compute the similarity between cases

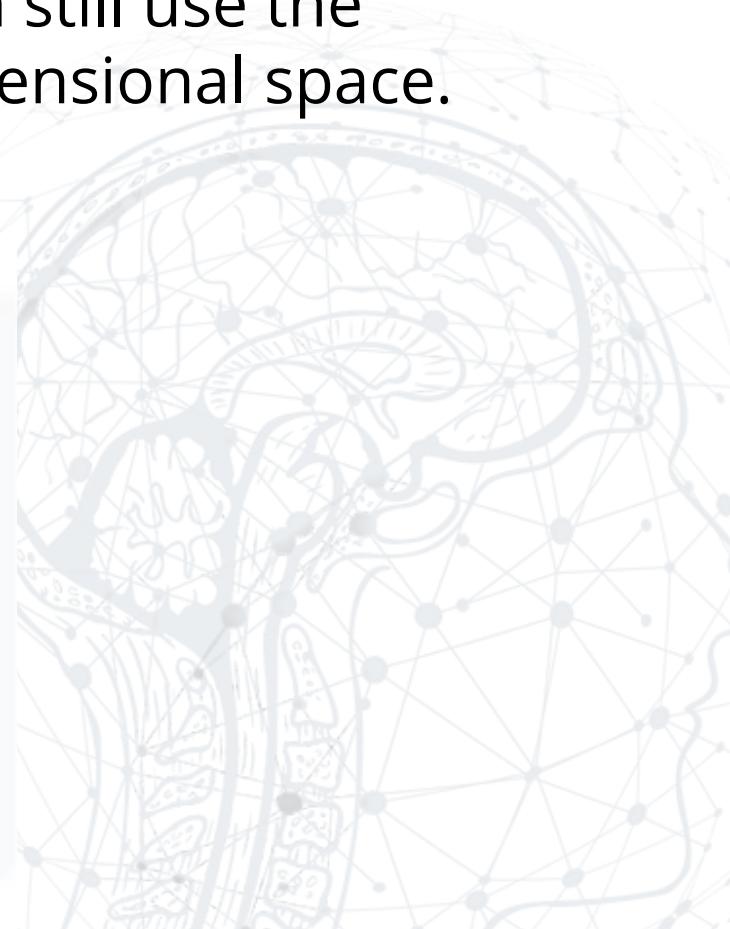


# Calculating distance → e.g. Euclidean

- What about if we have more than one feature, for example Age and Income?
- If we have income and age for each customer, we can still use the same formula, but this time, we're using it in a 2-dimensional space.



$$\begin{aligned}
 \text{Dis } (x_1, x_2) &= \sqrt{\sum_{i=0}^n (x_{1i} - x_{2i})^2} \\
 &= \sqrt{(54 - 50)^2 + (190 - 200)^2} = 10.77
 \end{aligned}$$



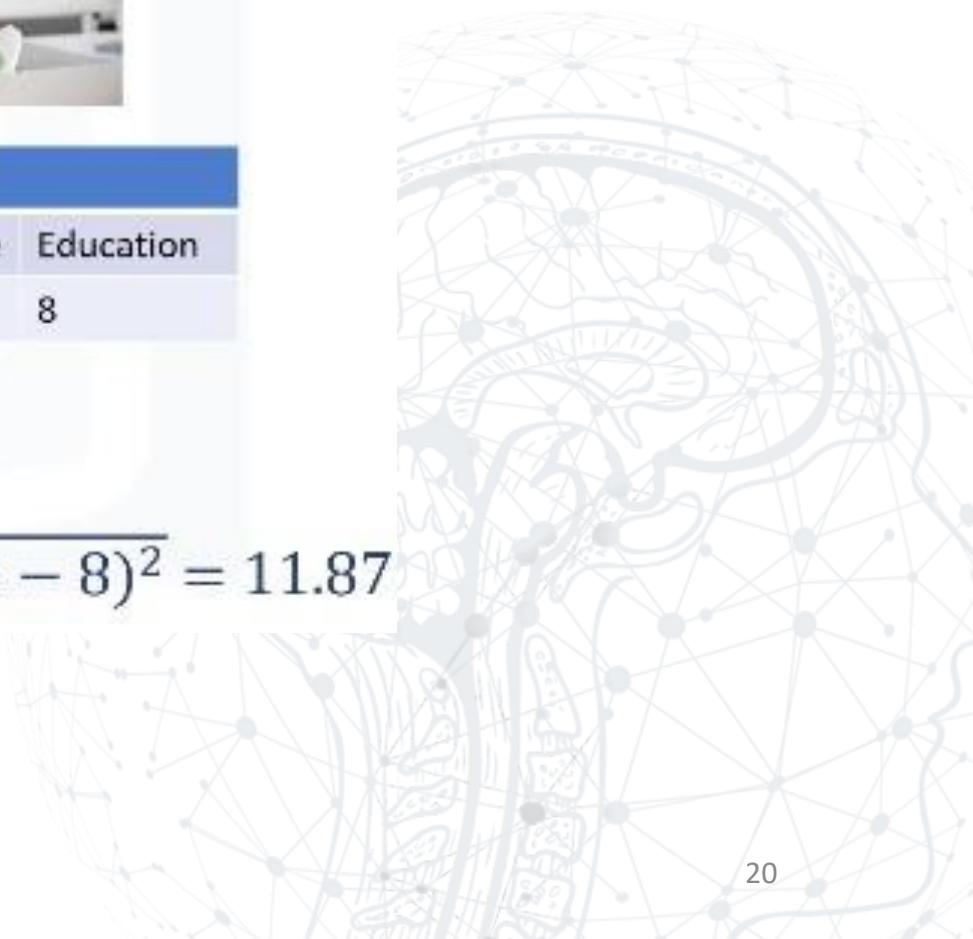
## 3 dimensional



Customer 1		
Age	Income	Education
54	190	3

Customer 2		
Age	Income	Education
50	200	8

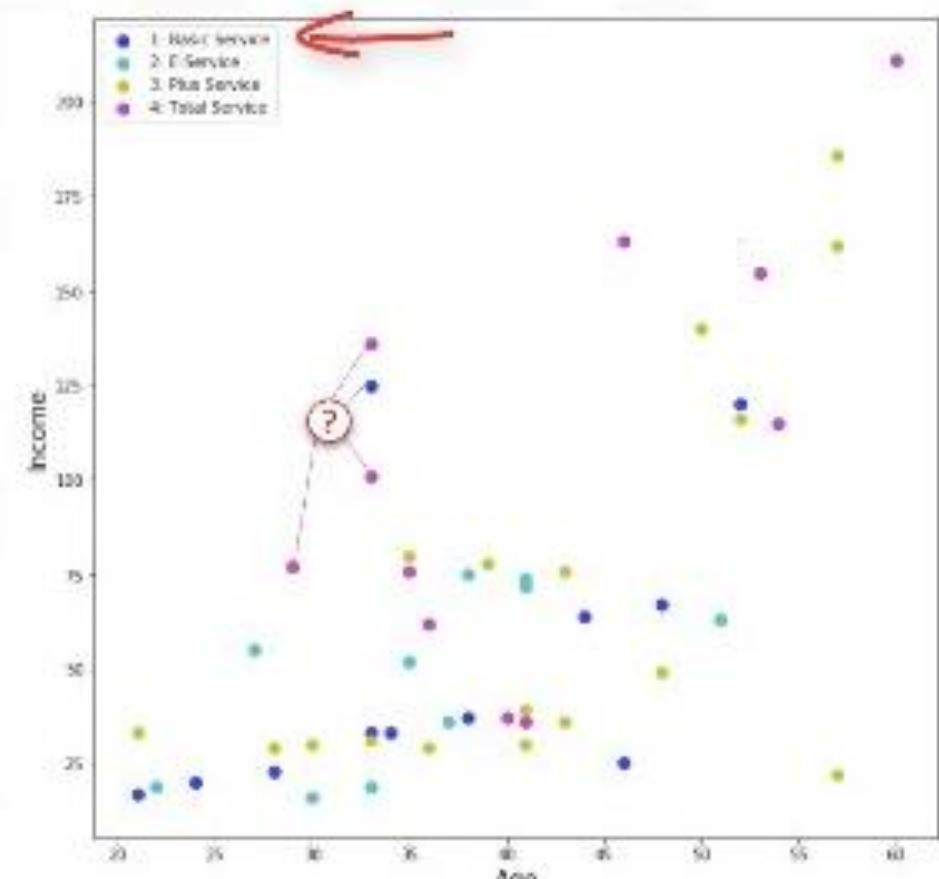
$$\begin{aligned}\text{Dis } (x_1, x_2) &= \sqrt{\sum_{i=0}^n (x_{1i} - x_{2i})^2} \\ &= \sqrt{(54 - 50)^2 + (190 - 200)^2 + (3 - 8)^2} = 11.87\end{aligned}$$



# Choose the right K

What happens if we choose a very low value of K, let's say, k=1? The first nearest point would be Blue, which is class 1.

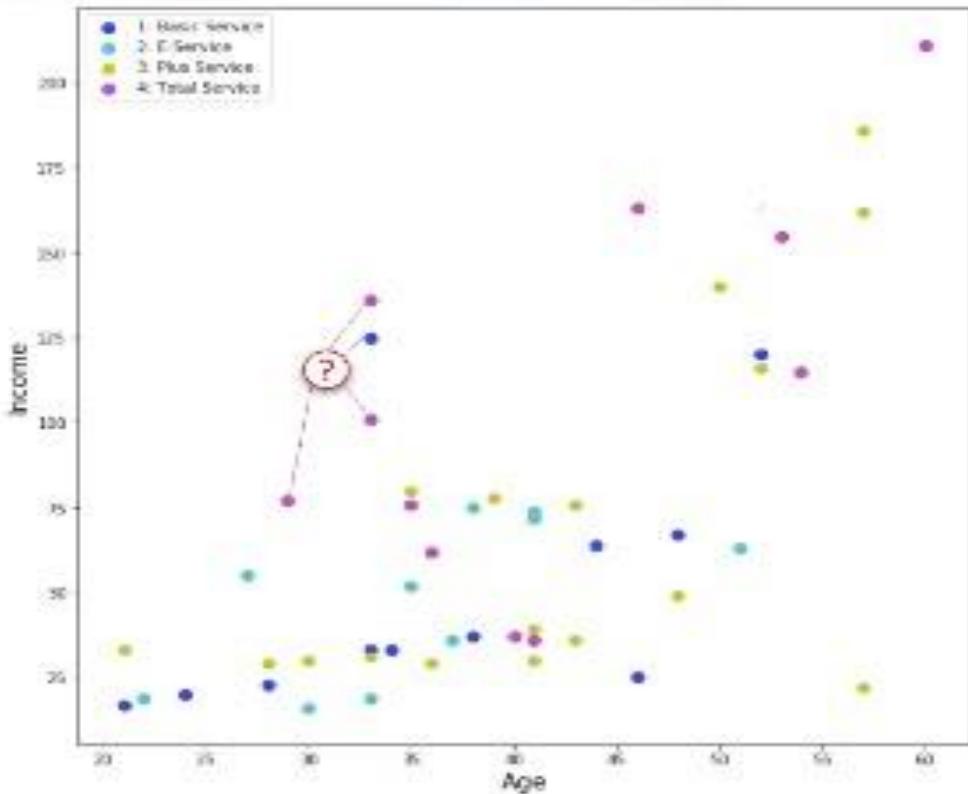
- This would be a bad prediction, since more of the points around it are Magenta, or class 4.
- It means the prediction process is **not generalized enough** to be used for out-of-sample cases.
- Out-of-sample data is data that is outside of the dataset used to train the model.
- In other words, it cannot be trusted to be used for prediction of unknown samples.
- It's important to remember that **over-fitting** is bad, as we want a general model that works for any data, not just the data used for training.



Now, on the opposite side of the spectrum, if we choose a very high value of K, such as K=20, then the model becomes **overly generalized → underfitting**

So, how we can find the best value for K?

- K =1      class 1
- K =20      ?



# Solution



- The general solution is to reserve a part of your data for testing the accuracy of the model.
- Once you've done so, choose  $k = 1$ , and then use the training part for modeling, and calculate the accuracy of prediction using all samples in your test set.
- Repeat this process, increasing the  $k$ , and see which  $k$  is best for your model.



UNIVERSITAS  
INDONESIA

*Veritas, Prudentia, Justitia*

# Model Evaluation

We have trained the model, and now we want to calculate its accuracy using the test set. We pass the test set to our model, and we find the predicted labels.

Now the question is, “How accurate is this model?”



# Evaluation Metrics



- Evaluation metrics explain the performance of a model.
- Basically, we compare the actual values in the test set with the values predicted by the model, to calculate the accuracy of the model.
- Evaluation metrics provide a key role in the development of a model, as they provide insight to areas that might require improvement.
- There are different model evaluation metrics but we just talk about three of them here, specifically: Jaccard index, F1-score, and Log Loss.

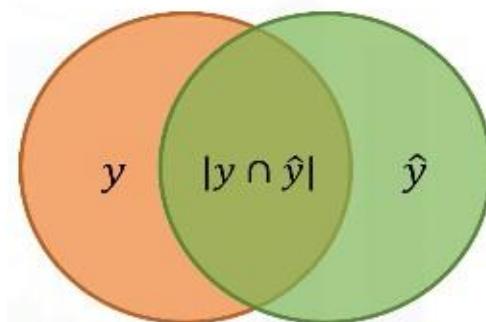
# Jaccard Index

- Let's first look at one of the simplest accuracy measurements, the Jaccard index – also known as the Jaccard similarity coefficient.
- Let's say  $y$  shows the true labels of the churn dataset. And  $\hat{y}$  shows the predicted values by our classifier.

## Jaccard index

$y$ : Actual labels

$\hat{y}$ : Predicted labels



# Jaccard Index

- Jaccard = size of the intersection divided by the size of the union of two label sets.
- For example, for a test set of size 10, with 8 correct predictions, or 8 intersections, the accuracy by the Jaccard index would be 0.66.
- If the entire set of predicted labels for a sample strictly matches with the true set of labels, then the subset accuracy is 1.0; otherwise it is 0.0.

$y$ : Actual labels

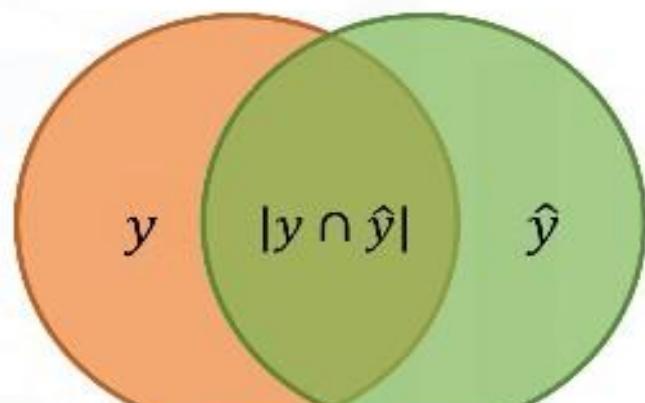
$\hat{y}$ : Predicted labels

$$J(y, \hat{y}) = \frac{|y \cap \hat{y}|}{|y \cup \hat{y}|} = \frac{|y \cap \hat{y}|}{|y| + |\hat{y}| - |y \cap \hat{y}|}$$

$y$ : [0, 0, 0, 0, 0, 1, 1, 1, 1, 1]

$\hat{y}$ : [1, 1, 0, 0, 0, 1, 1, 1, 1, 1]

$$J(y, \hat{y}) = \frac{8}{10+10-8} = 0.66$$



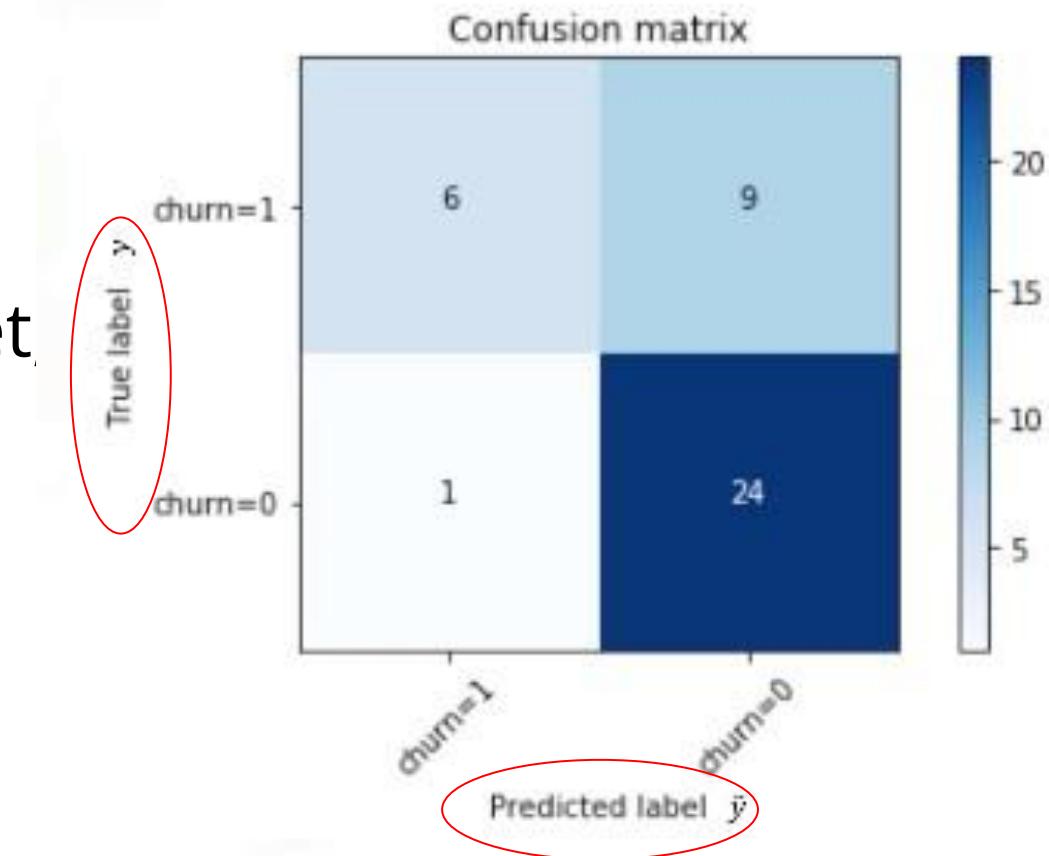
$$J(y, \hat{y}) = 0.0$$



$$J(y, \hat{y}) = 1.0$$

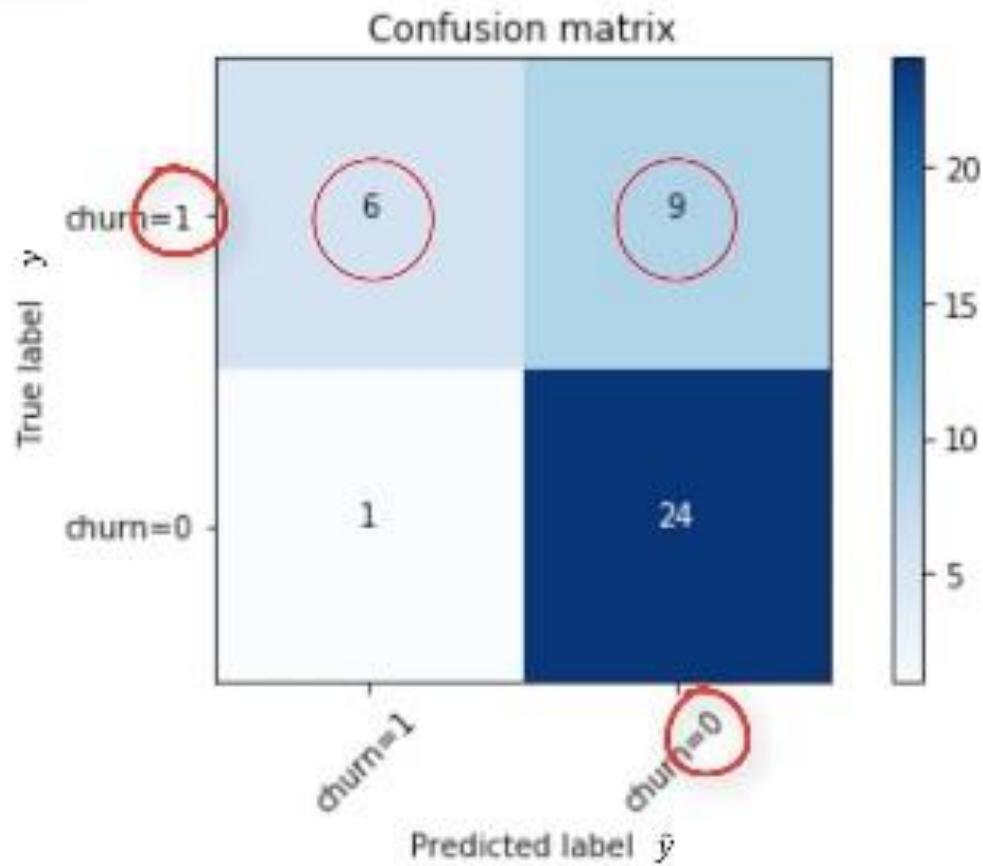
# Confusion Matrix

- This matrix shows the corrected and wrong predictions, in comparison with the actual labels.
- Each confusion matrix row shows the Actual/True labels in the test set and the columns show the predicted labels by classifier.
- For example, let's assume that our test set has only 40 rows.

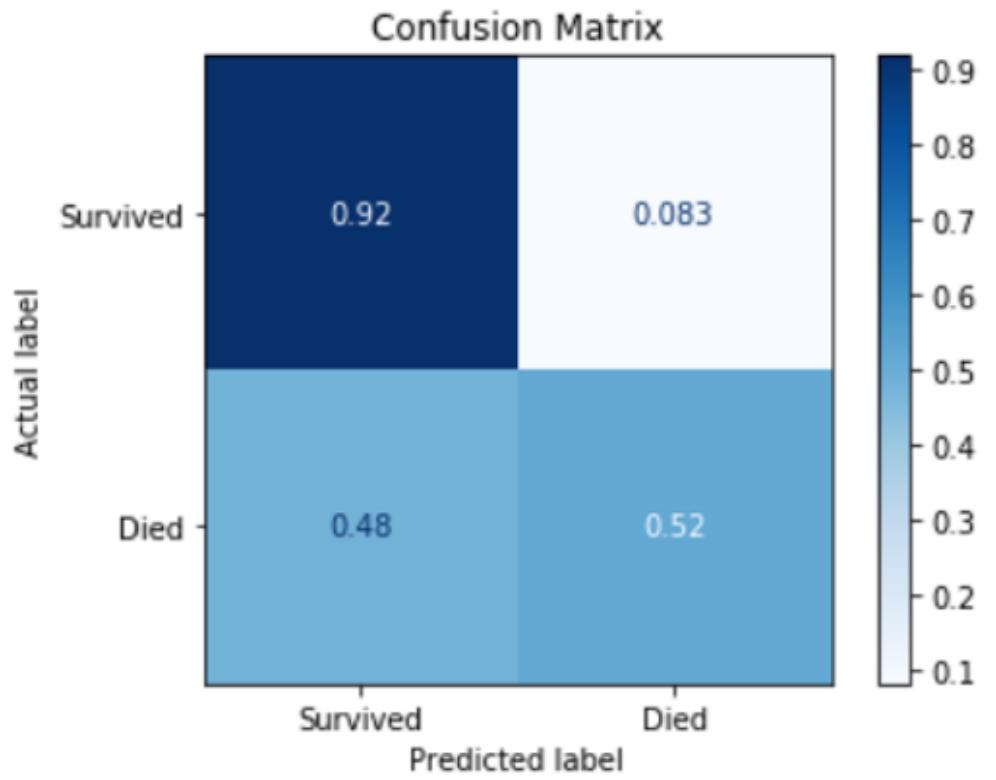
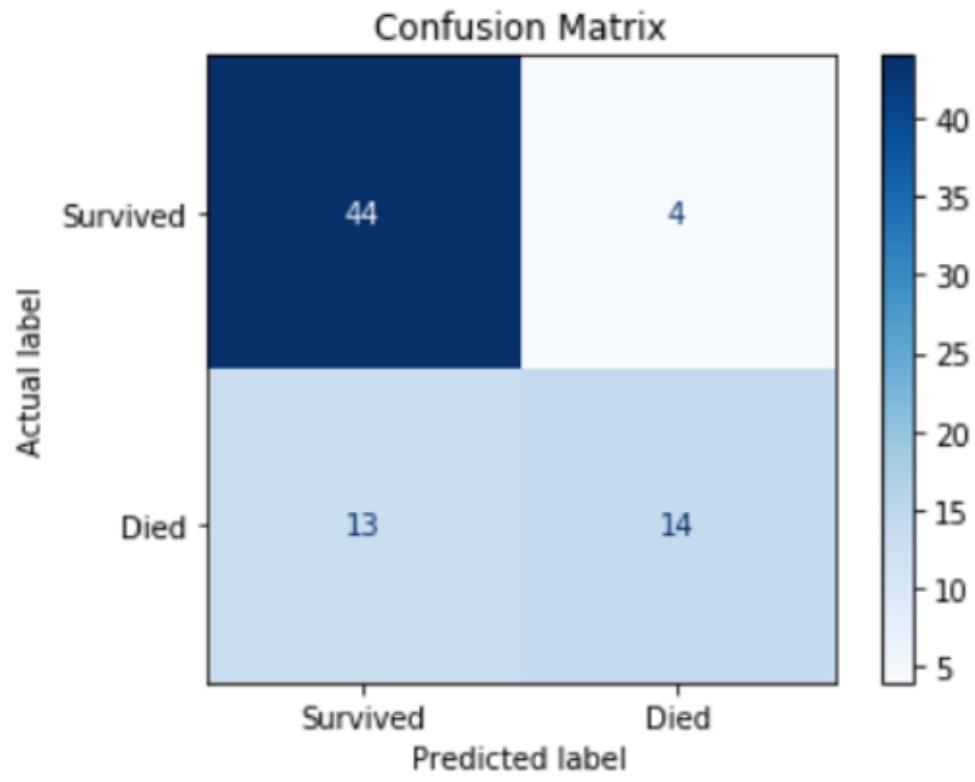


# Confusion Matrix

- The first row: actual churn value = 1 → 15 people
  - correctly predicted 6 of them as 1,
  - and 9 of them as 0 (wrong).
- Second row: actual churn value = 0 → 24 people
  - correctly predicted 24 of them as 0,
  - and 1 of them as 1 (wrong) .
- So, it has done a good job in predicting the customers with a churn value of 0, but not very good in predicting value of 1



# Confusion Matrix: Not Normalized/Normalized



- A good thing about the confusion matrix is that it shows the model's ability to correctly predict or separate the classes.
- In the specific case of a binary classifier, such as this example, we can interpret these numbers as the count of **true positives**, **false positives**, **true negatives**, and **false negatives**.
- Based on the count of each section, we can calculate the precision and recall of each label.

# TP, TN, FP, FN

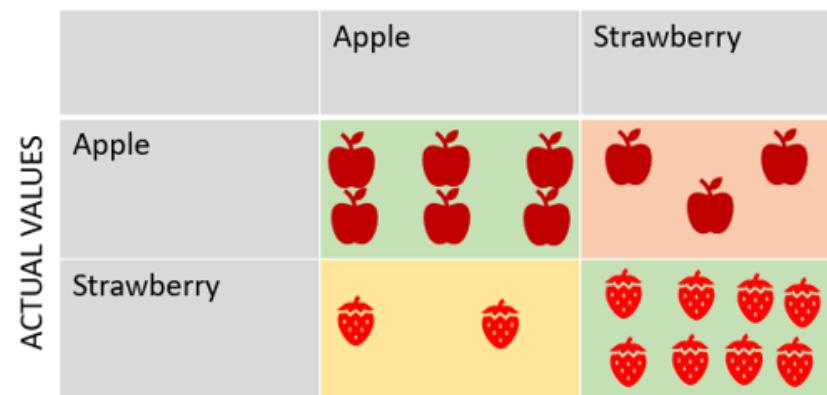
PREDICTED VALUES

ACTUAL VALUES	Positive	Negative
	True Positive (TP)	False Negative (FN)
Positive		
Negative	False Positive (FP)	True Negative (TN)

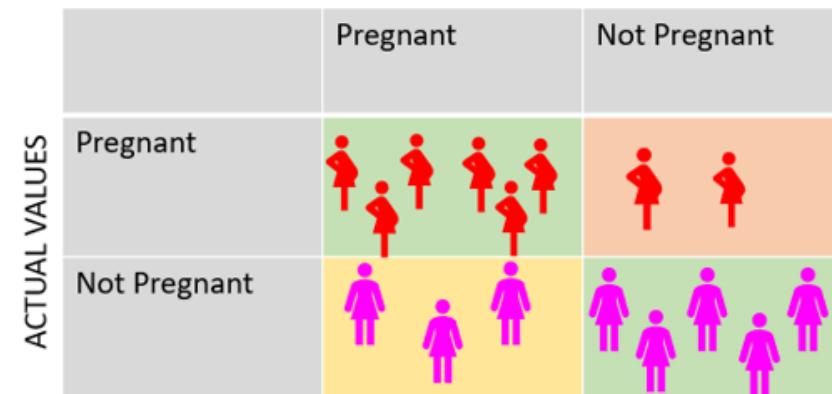
ACTUAL VALUES

PREDICTED VALUES	Positive	Negative
	True Positive (TP)	False Positive (FP)
Positive		
Negative	False Negative (FN)	True Negative (TN)

PREDICTED VALUES



PREDICTED VALUES



# Precision – Recall – F1 Score

- **Precision** is a measure of the accuracy, provided that a class label has been predicted.

$$\text{Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$$

- **Recall** is the true positive rate.

$$\text{Recall} = \text{True Positive} / (\text{True Positive} + \text{False Negative}).$$

- **F1 score** is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (which represents perfect precision and recall) and its worst at 0.

$$\bullet \text{F1-score} = 2x (\text{prc} \times \text{rec}) / (\text{prc} + \text{rec})$$





# Precision – Recall – F1 Score

		ACTUAL VALUES	
		Positive	Negative
PREDICTED VALUES	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)
		Recall/Sensitivity= $TP/TP+FN$	Specificity= $TN/TN+FP$
			ACCURACY= $\frac{TP+TN}{TP+TN+FN+FP}$

		PREDICTED VALUES	
		Positive	Negative
ACTUAL VALUES	Positive	True Positive (TP)	False Negative (FN)
	Negative	False Positive (FP)	True Negative (TN)
		Recall/Sensitivity= $TP/TP+FN$	Specificity= $TN/TN+FP$
		Precision= $TP/TP+FP$	NPV= $TN/TN+FN$
			ACCURACY= $\frac{TP+TN}{TP+TN+FN+FP}$

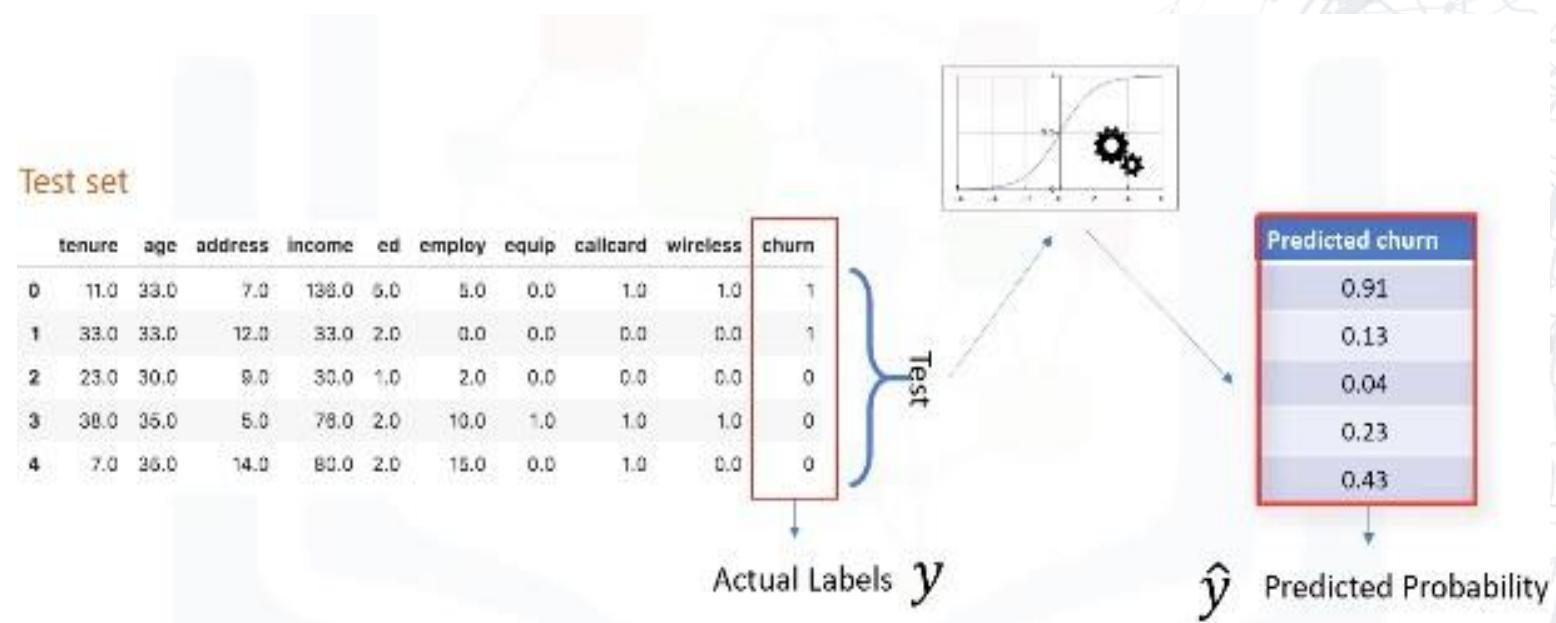
# Accuracy

- It is a good way to show that a classifier has a good value for both recall and precision. It is defined using the F1-score equation.
- For example, the F1-score for class 0 (i.e. churn=0), is 0.83, and the F1-score for class 1 (i.e. churn=1), is 0.55.
- And finally, we can tell the average accuracy for this classifier is the average of the F1-score for both labels, which is 0.72 in our case.
- Please notice that both Jaccard and F1-score can be used for multi-class classifiers as well.

	precision	recall	f1-score
Churn = 0	0.73	0.96	0.83
Churn = 1	0.86	0.40	0.55
Avg Accuracy = 0.72			

# Log-loss

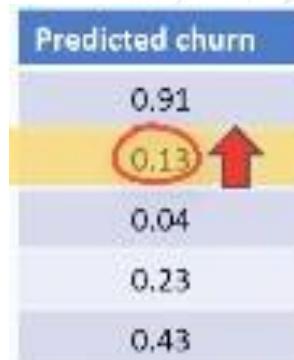
- Sometimes, the output of a classifier is **the probability of a class label**, instead of the label.
- For example, in logistic regression, the output can be the probability of customer churn, i.e., yes (or equals to 1). This probability is a value between 0 and 1.



# Evaluation Matrix

- Logarithmic loss (also known as Log loss) measures the performance of a classifier where the predicted output is a probability value between 0 and 1.
- So, for example, predicting a probability of 0.13 when the actual label is 1, would be bad and would result in a high log loss.

Test set											
	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn	
0	11.0	33.0	7.0	138.0	6.0	5.0	0.0	1.0	1.0	1	
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1	
2	23.0	30.0	9.0	33.0	1.0	2.0	0.0	0.0	0.0	0	
3	36.0	35.0	6.0	78.0	2.0	10.0	1.0	1.0	1.0	0	
4	7.0	36.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0	



# Evaluation Matrix

- We can calculate the log loss for each row using the log loss equation, which measures how far each prediction is, from the actual label.

$$(y \times \log(\hat{y}) + (1 - y) \times \log(1 - \hat{y}))$$

- Then, we calculate the average log loss across all rows of the test set.
- It is obvious that more ideal classifiers have progressively smaller values of log loss. So, the classifier with lower log loss has better accuracy.

# Log loss

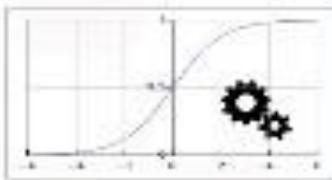
Performance of a classifier where the predicted output is a probability value between 0 and 1.

Test set

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	6.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	8.0	33.0	1.0	2.0	0.0	0.0	0.0	0
3	38.0	35.0	6.0	78.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	36.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

Test

Actual Labels  $y$



Predicted churn	LogLoss
0.91	0.11
0.13	2.04
0.04	0.04
0.23	0.26
0.43	0.56

$$\text{LogLoss} = 0.60$$

$\hat{y}$  Predicted Probability

LogLoss: 0.00 ... 0.35 ... 0.60 ... 1.00

Higher Accuracy

$$\text{LogLoss} = -\frac{1}{n} \sum (y \times \log(\hat{y}) + (1 - y) \times \log(1 - \hat{y}))$$



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# Decision Tree

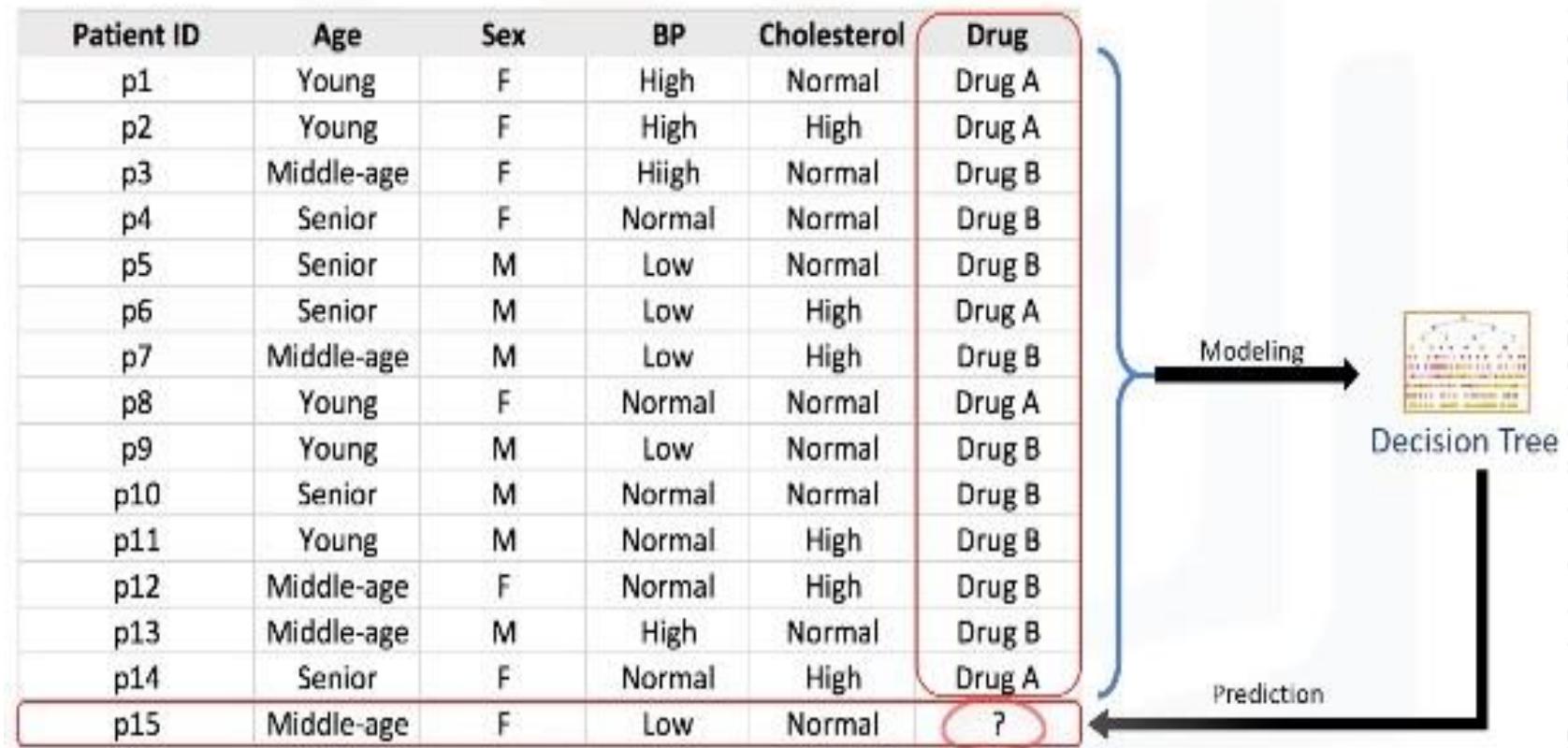
# Introduction to Decision Trees

- Imagine that you're a medical researcher compiling data for a study.
- You've already collected data about a set of patients, all of whom suffered from the same illness.

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	Hiigh	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A
p15	Middle-age	F	Low	Normal	?

# Decision Trees

- The feature sets of this dataset are Age, Gender, Blood Pressure, and Cholesterol of our group of patients, and the target is the drug that each patient responded to.
- It is a sample of binary classifiers, and you can use the training part of the dataset to build a decision tree, and then, use it to predict the class of an unknown patient ... in essence, to come up with a decision on which drug to prescribe to a new patient.



# How to Build Decision Trees

Decision trees are built by splitting the training set into distinct nodes, where one node contains all of, or most of, one category of the data.

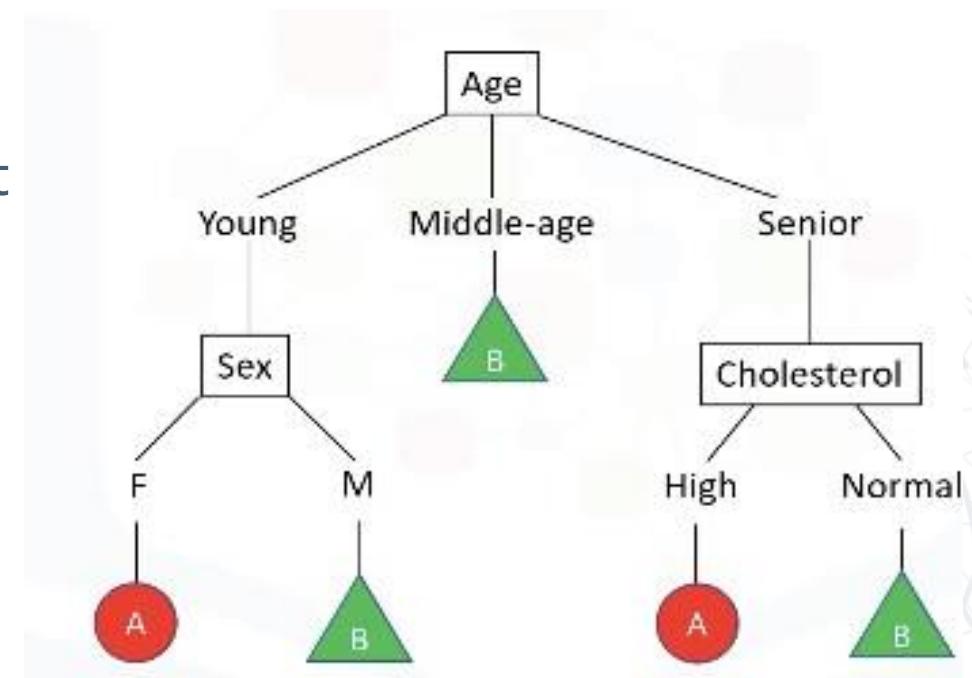
If we look at the diagram here, we can see that it's a patient classifier.

So, as mentioned, we want to prescribe a drug to a new patient, but the decision to choose drug A or B, will be influenced by the patient's situation.

We start with the Age, which can be Young, Middle-aged, or Senior.

If the patient is Middle-aged, then we'll definitely go for Drug B.

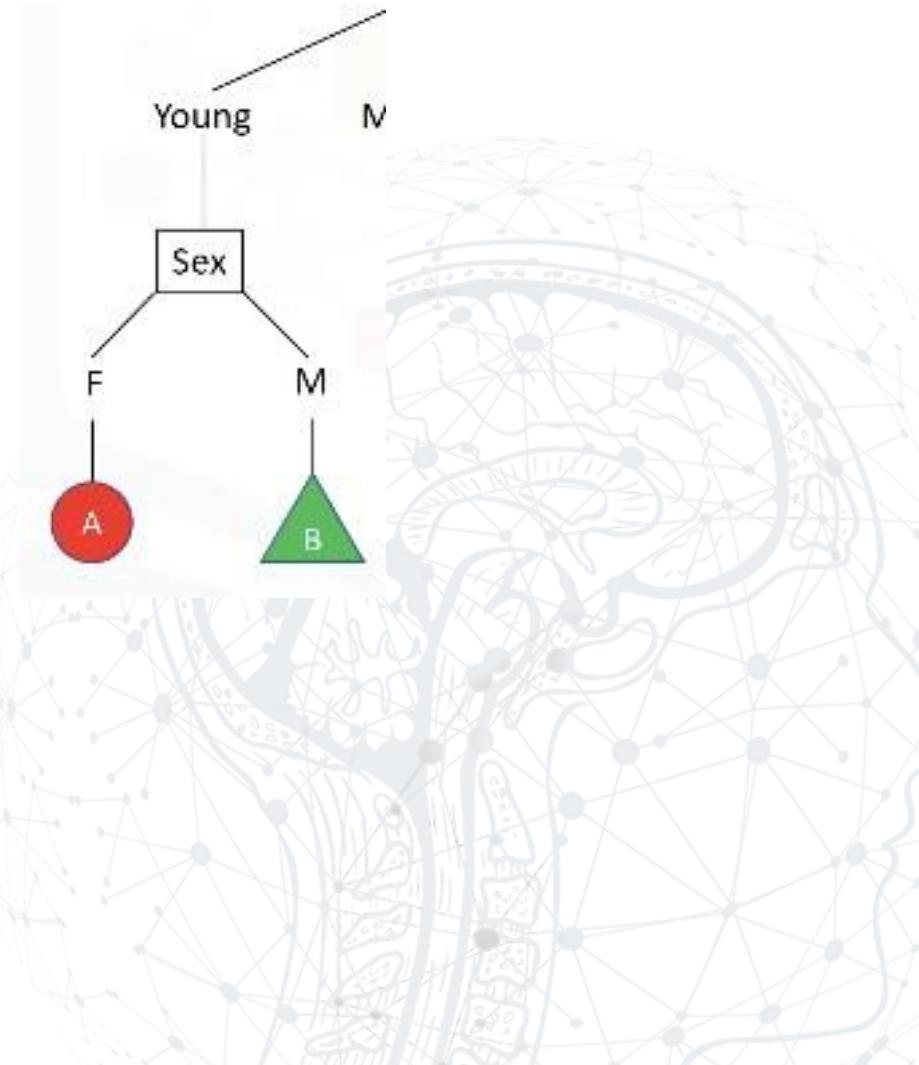
On the other hand, if he is a Young or a Senior patient, we'll need more details to help us determine which drug to prescribe.



# How to Build Decision Trees

The additional decision variables can be things such as Cholesterol levels, Gender or Blood Pressure.

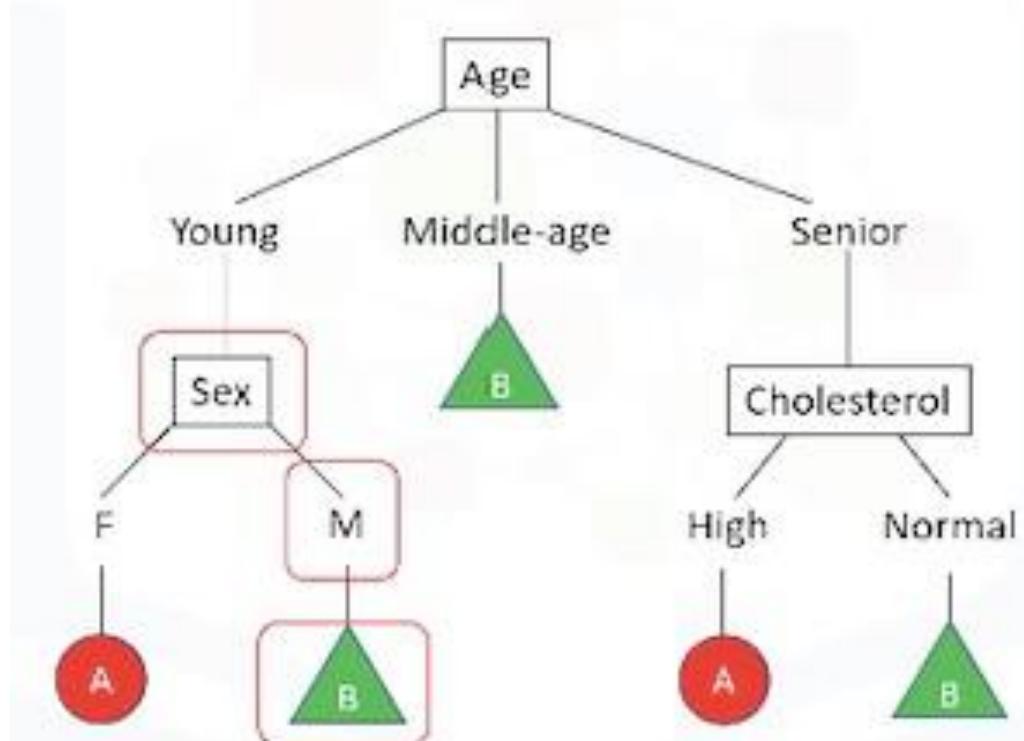
For example, if the patient is Female, then we will recommend Drug A, but if the patient is Male, then we'll go for Drug B. As you can see, decision trees are about testing an attribute and branching the cases, based on the result of the test.



# How to Build Decision Trees

Each internal node corresponds to a test. And each branch corresponds to a result of the test. And each leaf node assigns a patient to a class.

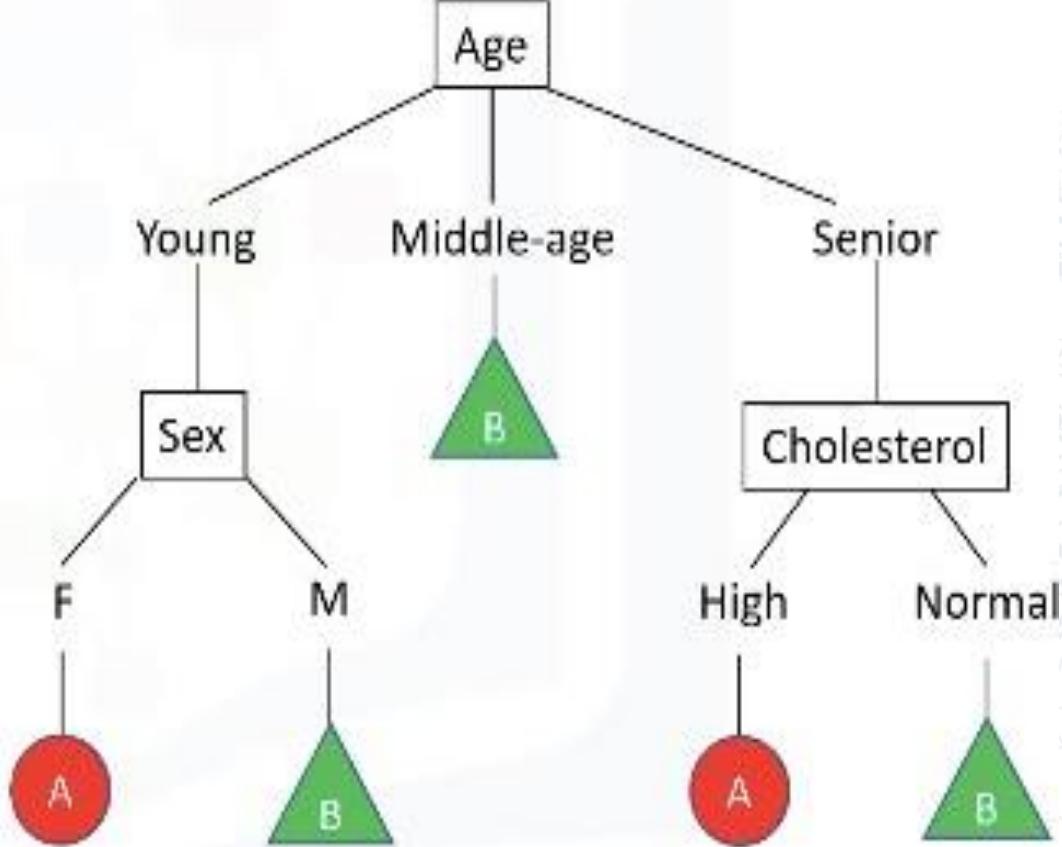
Now the question is how can we build such a decision tree?



- Each **internal node** corresponds to a test
- Each **branch** corresponds to a result of the test
- Each **leaf node** assigns a classification

# Decision Tree Algorithm

1. Choose an attribute from your dataset.
2. Calculate the significance of attribute in splitting of data.
3. Split data based on the value of the best attribute.
4. Go to step 1.



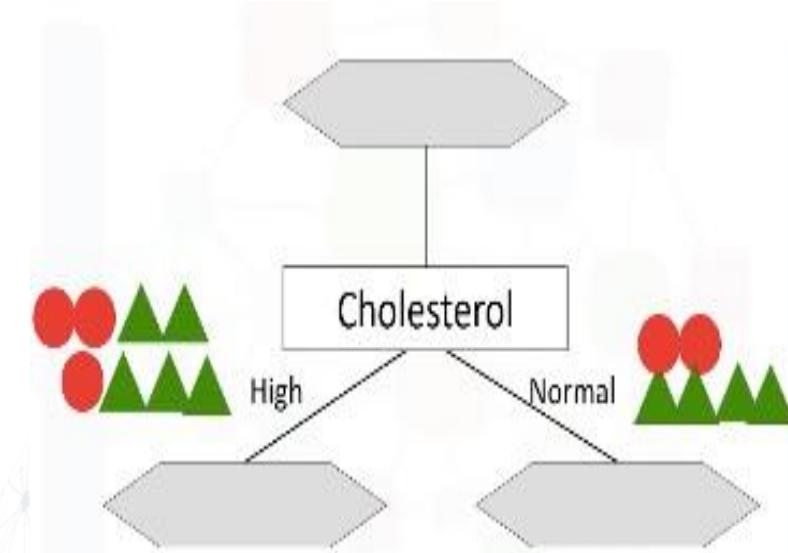
# Building Decision Trees

- Consider the drug dataset again.
- The question is, “How do we build the decision tree based on that dataset?”

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A
p15	Middle-age	F	Low	Normal	?

# Building Decision Trees

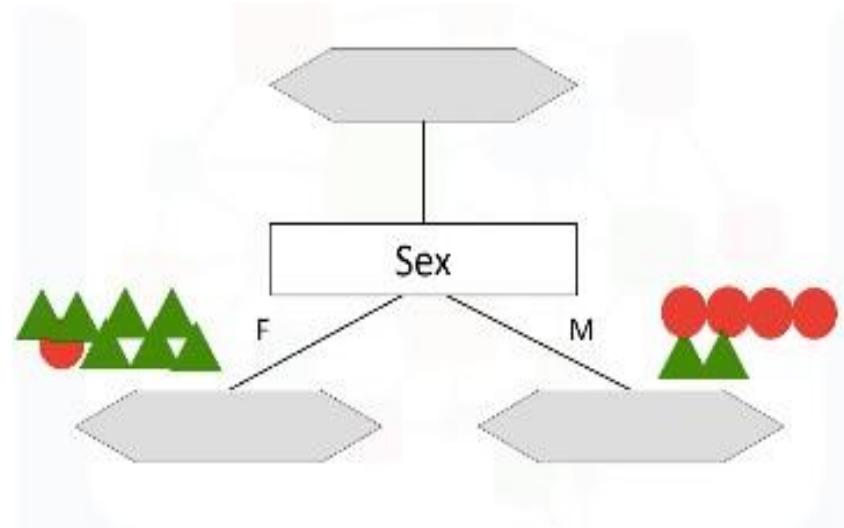
- Decision trees are built using recursive partitioning to classify the data.
- The algorithm chooses the most predictive feature to split the data on.
  - “which attribute is the best, or more predictive, to split data based on the feature.”
- Let’s say we pick “Cholesterol” as the first attribute to split data. It will split our data into 2 branches.
- As you can see, if the patient has high “Cholesterol,” we cannot say with high confidence that Drug B might be suitable for him.
- Also, if the Patient’s “Cholesterol” is normal, we still don’t have sufficient evidence or information to determine if either Drug A or Drug B is, in fact, suitable.



Thus, Cholesterol is not a good attribute now

# Building Decision Trees

- So, let's try another attribute. This time, we pick the "sex" attribute of patients.
- It will split our data into 2 branches, Male and Female.
- As you can see, if the patient is Female, we can say Drug B might be suitable for her with high certainty.

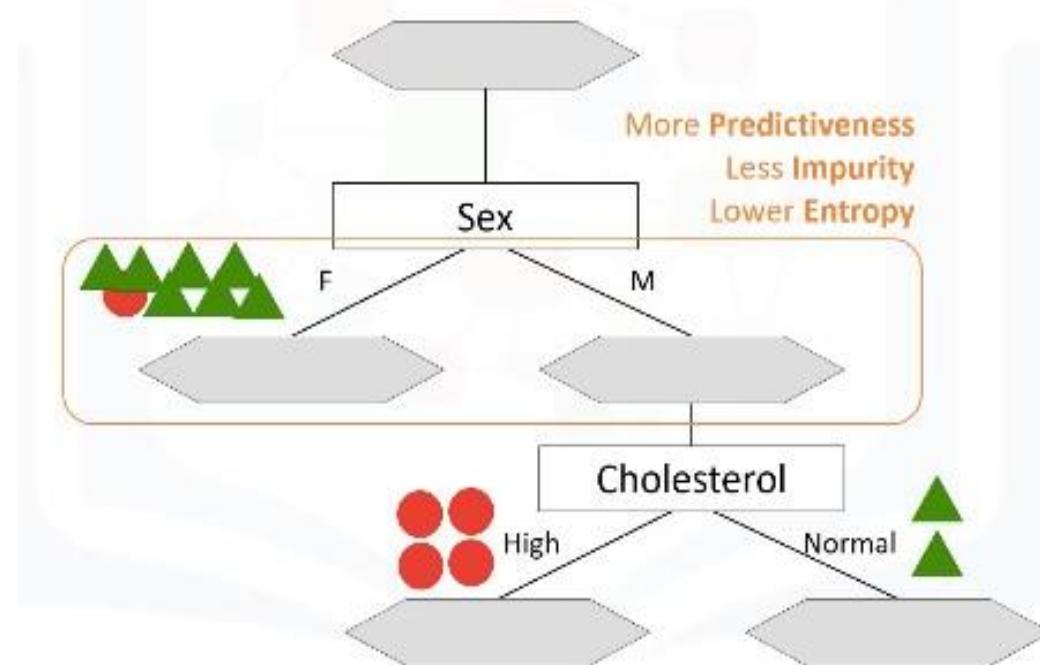


# Building Decision Trees

- But, if the patient is Male, we don't have sufficient evidence or information to determine if Drug A or Drug B is suitable.
- However, it is still a better choice in comparison with the "Cholesterol" attribute, because the result in the nodes are more pure.
- It means, nodes that are either mostly Drug A or Drug B.
- So, we can say the "Sex" attribute is more significant than "Cholesterol," or in other words, it's more predictive than the other attributes.
- Indeed, "predictiveness" is based on decrease in "impurity" of nodes.
- We're looking for the best feature to decrease the "impurity" of patients in the leaves, after splitting them up based on that feature.
- So, the "Sex" feature is a good candidate in the following case, because it almost found the pure patients. Let's go one step further.
- For the Male patient branch, we again test other attributes to split the subtree.

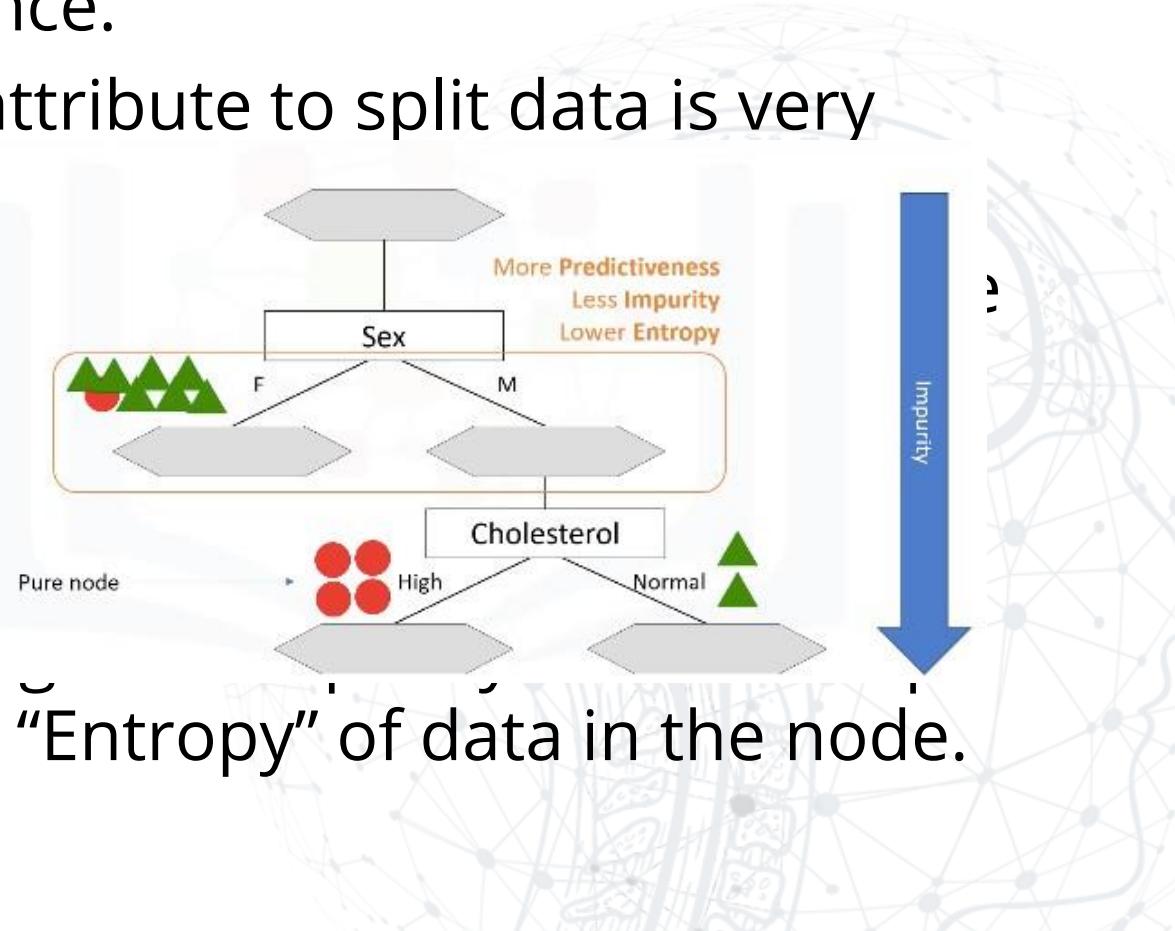
# Building Decision Trees

- We test “Cholesterol” again here.
- As you can see, it results in even more pure leaves. So, we can easily make a decision here.



# Building Decision Trees

- For example, if a patient is “Male”, and his “Cholesterol” is “High”, we can certainly prescribe Drug A, but if it is “Normal”, we can prescribe Drug B with high confidence.
- As you might notice, the choice of attribute to split data is very important, and it is all about
- A node in the tree is considered a pure node if all the data points fall into a specific category. 
- In fact, the method uses recursive splitting to divide the records into segments by minimizing impurity. The “Impurity” of nodes is calculated by “Entropy” of data in the node.



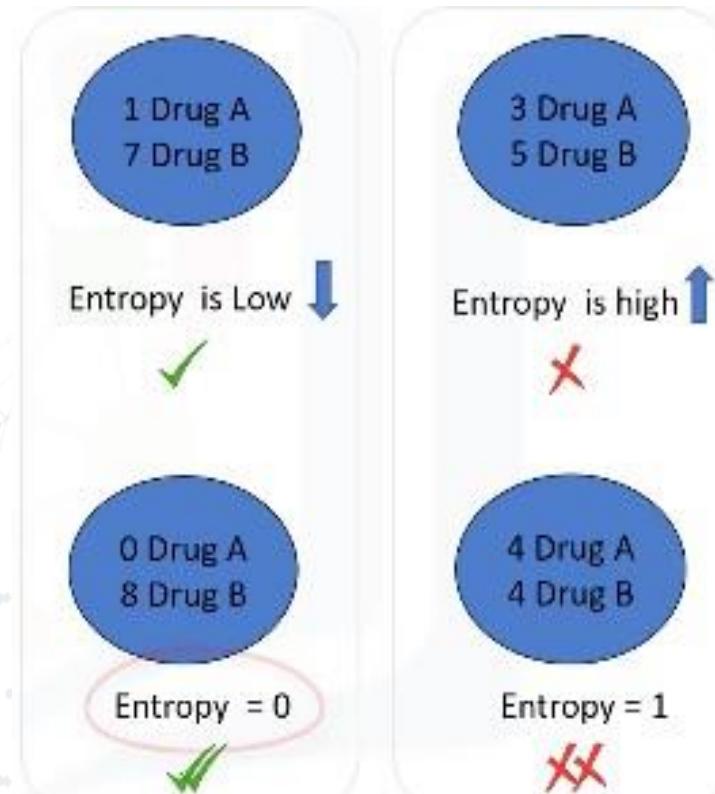
# Decision Trees - Entropy

- So, what is “Entropy”?
- Entropy is the amount of information disorder, or the amount of randomness in the data.
- The entropy in the node depends on how much random data is in that node and is calculated for each node. In decision trees, we're looking for trees that have the smallest entropy in their nodes.



# Decision Trees - Entropy

- The entropy is used to calculate the homogeneity of the samples in that node.
- If the samples are completely homogeneous the entropy is zero and if the samples are equally divided, it has an entropy of one.
- This means, if all the data in a node are either Drug A or Drug B, then the entropy is zero, but if half of the data are Drug A and other half are B, then the entropy is one.



# Decision Trees - Entropy

- You can easily calculate the entropy of a node using the frequency table of the attribute through the Entropy formula, where P is for the proportion or ratio of a category, such as Drug A or B.
- Please remember, though, that you don't have to calculate these, as it's easily calculated by the libraries or packages that you use.

$$\text{Entropy} = - p(A)\log(p(A)) - p(B)\log(p(B))$$

- As an example, let's calculate the entropy of the dataset before splitting it. We have 9 occurrences of Drug B and 5 of Drug A.
- You can embed these numbers into the Entropy formula to calculate the impurity of the target attribute before splitting it. In this case, it is 0.94.

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A

S: [9 B, 5 A]

$$E = - p(B) \log(p(B)) - p(A) \log(p(A))$$

$$E = - (9/14) \log(9/14) - (5/14) \log(5/14)$$

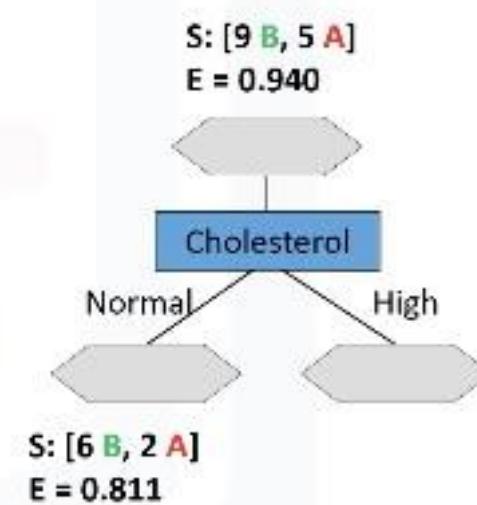
$$E = 0.940$$



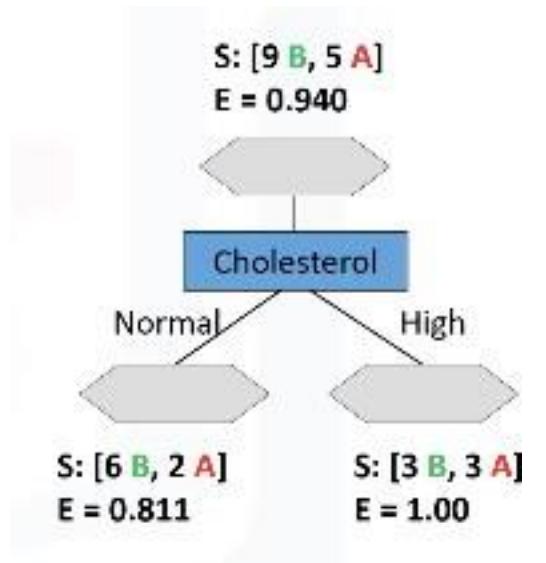
# Decision Trees - Entropy

- So, what is entropy after splitting?
- Now we can test different attributes to find the one with the most “predictiveness,” which results in two more pure branches.
- Let’s first select the “Cholesterol” of the patient and see how the data gets split, based on its values.
- For example, when it is “normal,” we have 6 for Drug B, and 2 for Drug A.
- We can calculate the Entropy of this node based on the distribution of drug A and B, which is 0.8 in this case.

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A



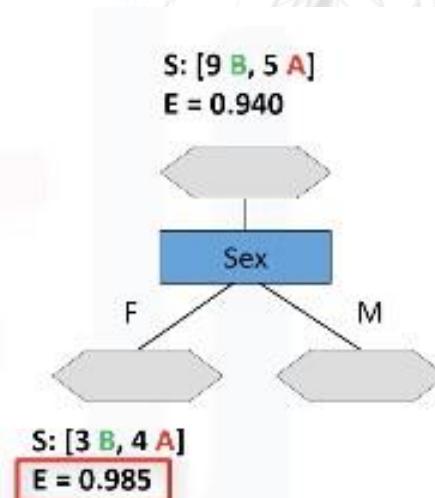
But, when Cholesterol is “High,” the data is split into 3 for drug B and 3 for drug A. Calculating its entropy, we can see it would be 1.0.



# Decision Trees - Entropy

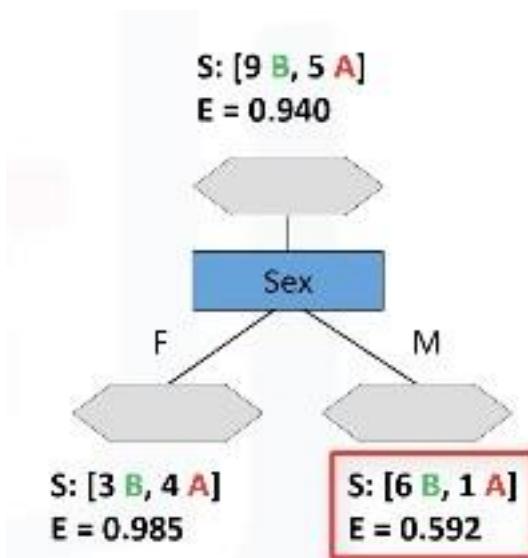
- We should go through all the attributes and calculate the “Entropy” after the split, and then chose the best attribute.
- Let’s try another field.
- Let’s choose the Sex attribute for the next check.
- As you can see, when we use the Sex attribute to split the data, when its value is “Female,” we have 3 patients that responded to Drug B, and 4 patients that responded to Drug A.
- The entropy for this node is 0.98 which is not very promising

Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A



# Decision Trees - Entropy

- However, on other side of the branch, when the value of the Sex attribute is Male, the result is more pure with 6 for Drug B and only 1 for Drug A.
- The entropy for this group is 0.59.

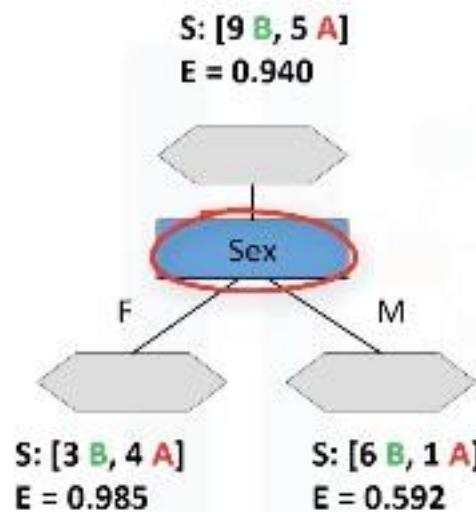


Now, the question is, between the Cholesterol and Sex attributes, which one is a better choice?

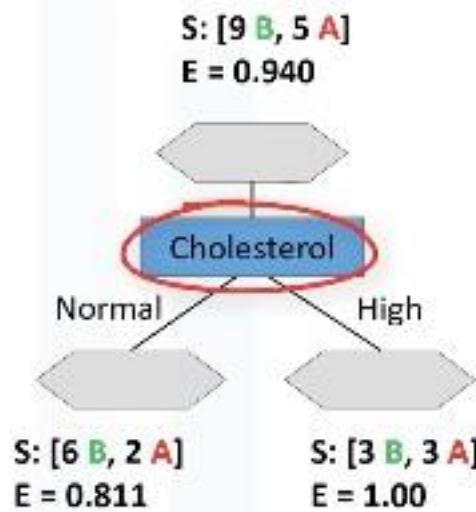
Which one is better as the first attribute to divide the dataset into 2 branches?

Or, in other words, which attribute results in more pure nodes for our drugs?

Or, in which tree, do we have less entropy after splitting rather than before splitting?



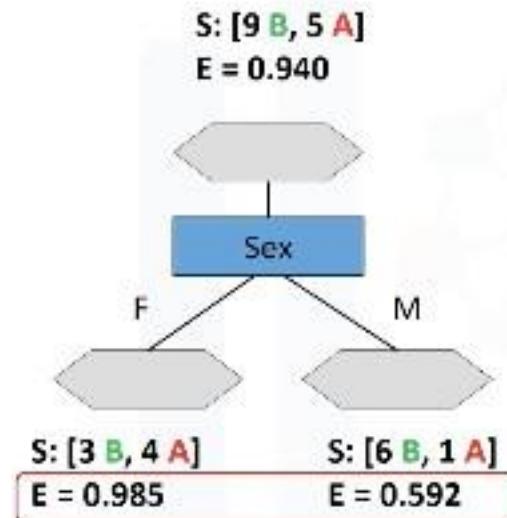
Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A



?

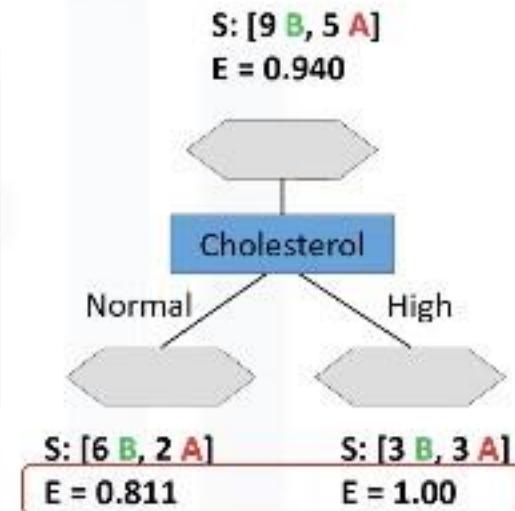
# Decision Trees - Entropy

- The “Sex” attribute with entropy of 0.98 and 0.59, or the “Cholesterol” attribute with entropy of 0.81 and 1.0 in its branches?
- The answer is, “The tree with the higher information gain after splitting.”



Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle-age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle-age	F	Normal	High	Drug B
p13	Middle-age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A

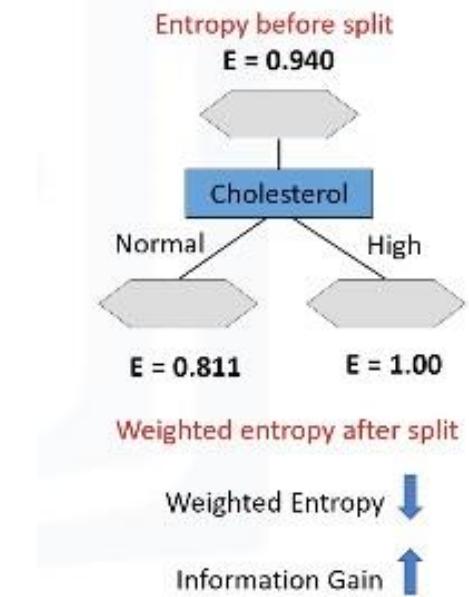
?



# Information gain

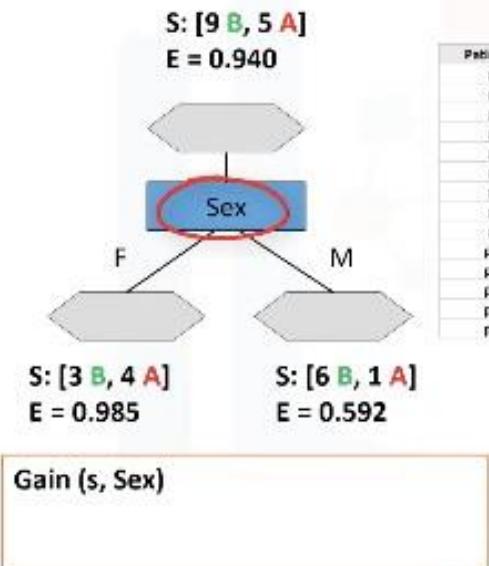
- Information gain is the information that can increase the level of certainty after splitting.
- It is the entropy of a tree before the split minus the weighted entropy after the split by an attribute.

Information Gain = (Entropy before split) – (weighted entropy after split)

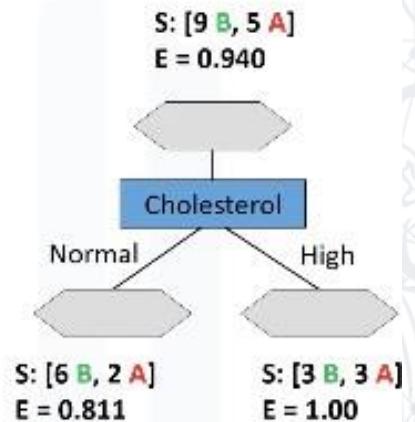


# Decision Trees

- We can think of information gain and entropy as opposites.
- As entropy, or the amount of randomness, decreases, the information gain, or amount of certainty, increases, and vice-versa.
- So, constructing a decision tree is all about finding attributes that return the highest information gain.
- Let's see how "information gain" is calculated for the Sex attribute.
- As mentioned, the information gain is the entropy of the tree before the split, minus the weighted

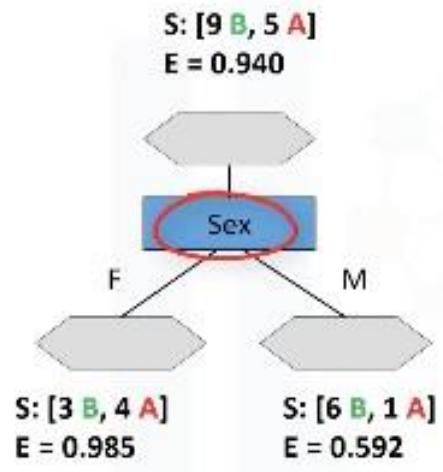


Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle-age	M	Low	High	Drug B
p8	Young	F	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Senior	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle age	M	Normal	High	Drug B
p13	Middle age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A



# Decision Trees

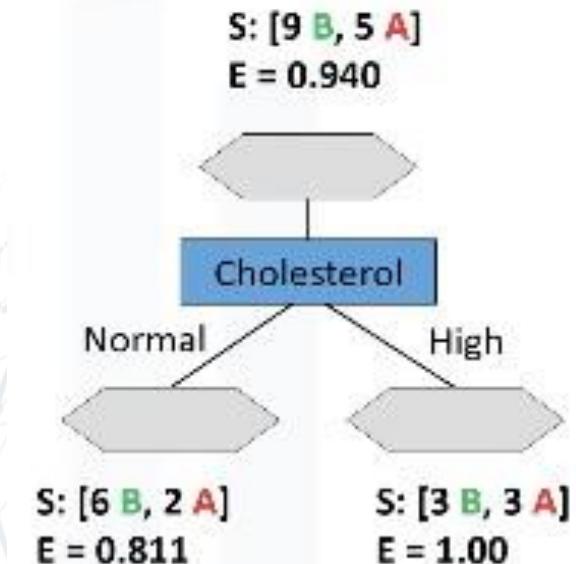
- The portion of Female patients is 7 out of 14, and its entropy is 0.985.
- Also, the portion of men is 7 out of 14, and the entropy of the Male node is 0.592.
- The result is the weighted entropy after the split. So, the information gain of the tree if we use the “Sex” attribute to split the dataset is 0.151.



$$\begin{aligned}
 \text{Gain } (s, \text{Sex}) &= 0.940 - [(7/14)0.985 + (7/14)0.592] \\
 &= 0.151
 \end{aligned}$$

# Decision Trees

- As you can see, we will consider the entropy over the distribution of samples falling under each leaf node, and we'll take a weighted average of that entropy – weighted by the proportion of samples falling under that leaf.
- We can calculate the information gain of the tree if we use “Cholesterol” as well. It is 0.48.

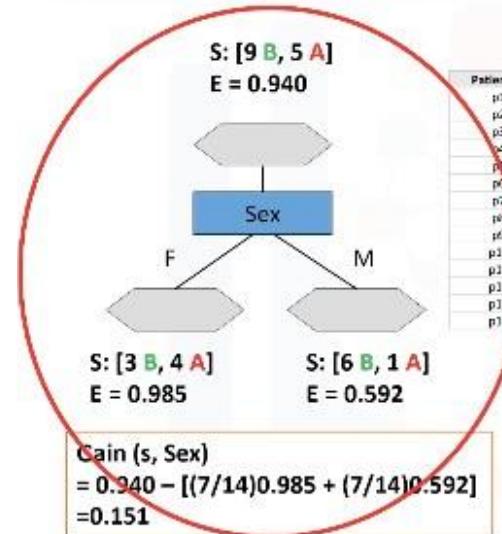


$$\begin{aligned}
 \text{Gain}(s, \text{Cholesterol}) &= 0.940 - [(8/14) \cdot 0.811 + (6/14) \cdot 1.0] \\
 &= 0.048
 \end{aligned}$$

# Decision Trees

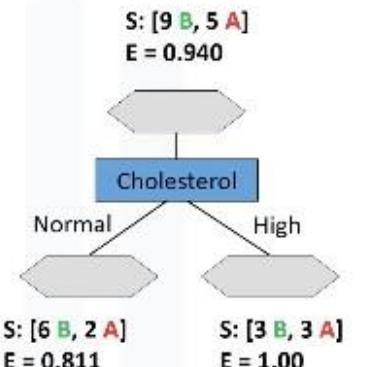
- Now, the question is, “Which attribute is more suitable?”
- Well, as mentioned, the tree with the higher information gain after splitting. This means the “Sex” attribute.

Which attribute is the best?



Patient ID	Age	Sex	BP	Cholesterol	Drug
p1	Young	F	High	Normal	Drug A
p2	Young	F	High	High	Drug A
p3	Middle age	F	High	Normal	Drug B
p4	Senior	F	Normal	Normal	Drug B
p5	Senior	M	Low	Normal	Drug B
p6	Senior	M	Low	High	Drug A
p7	Middle age	M	Low	High	Drug A
p8	Young	M	Normal	Normal	Drug A
p9	Young	M	Low	Normal	Drug B
p10	Young	M	Normal	Normal	Drug B
p11	Young	M	Normal	High	Drug B
p12	Middle age	F	Normal	High	Drug B
p13	Middle age	M	High	Normal	Drug B
p14	Senior	F	Normal	High	Drug A

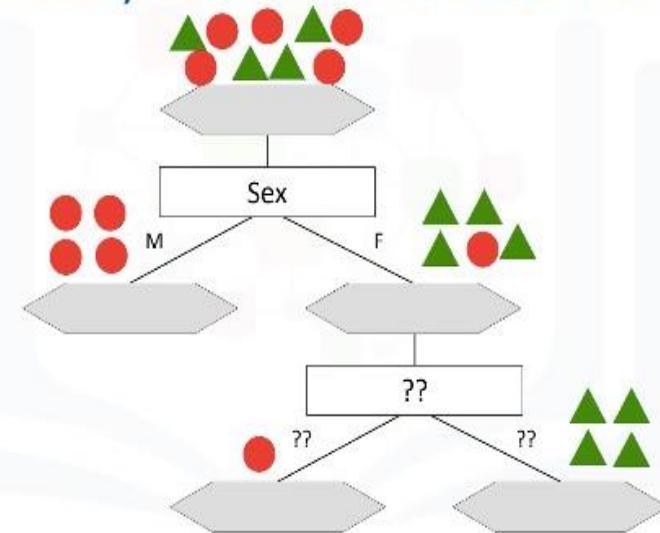
?



# Decision Trees

- So, we select the “Sex” attribute as the first splitter.
- Now, what is the next attribute after branching by the “Sex” attribute?
- Well, as you can guess, we should repeat the process for each branch, and test each of the other attributes to continue to reach the most pure leaves.
- This is the way that you build a decision tree!

Correct way to build a decision tree



# Other impurity index: Gini impurity index

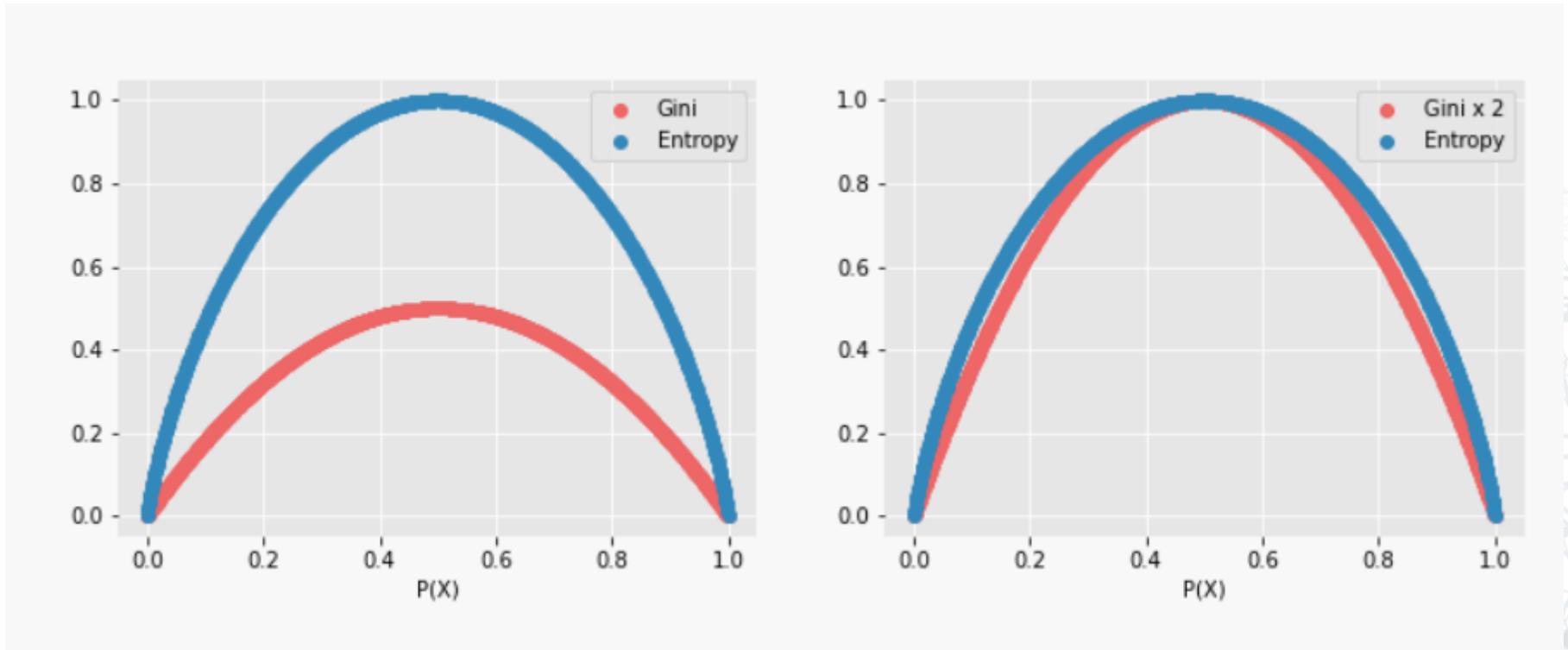
$$GiniIndex = 1 - \sum_j p_j^2$$

- The gini impurity measures the frequency at which any element of the dataset will be mislabelled when it is randomly labeled.
- The minimum value of the Gini Index is 0. This happens when the node is pure, this means that all the contained elements in the node are of one unique class. Therefore, this node will not be split again. Thus, the optimum split is chosen by the features with less Gini Index. Moreover, it gets the maximum value when the probability of the two classes are the same.

$$Gini_{min} = 1 - (1^2) = 0$$

$$Gini_{max} = 1 - (0.5^2 + 0.5^2) = 0.5$$

# Gini vs Entropy



- The gini criterion is much faster because it is less computationally expensive.
- On the other hand, the obtained results using the entropy criterion are slightly better



UNIVERSITAS  
INDONESIA

*Veritas, Prudentia, Justitia*

# Logistic Regression

# Logistic Regression

- Logistic regression is analogous to linear regression but tries to predict a categorical or discrete target field instead of a numeric one.
- In linear regression, we might try to predict a continuous value of variables, such as the price of a house, blood pressure of patient, or fuel consumption of a car.
- But, in logistic regression, we predict a variable which is **binary**, such as, Yes/No, TRUE/FALSE, successful or Not successful, pregnant/Not pregnant, and so on, all of which can all be coded as 0 or 1.
- In logistic regression, **dependent variables should be continuous**; if categorical, they should be dummy or indicator-coded.
- This means we have to transform them to some continuous value.
- Please note that logistic regression can be used for both binary classification and multiclass classification, but for simplicity, we'll focus on binary classification.

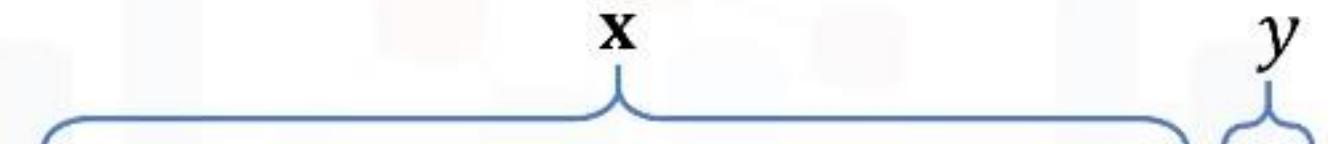
# When to use Logistic Regression

- Here are four situations in which Logistic regression is a good candidate:
- First, when the target field in your data is categorical, or specifically, is binary, such as 0/1, yes/no, churn or no churn, positive/negative, and so on.
- Second, when you need the probability of your prediction, for example, if you want to know what the probability is, of a customer buying a product.
  - **Logistic regression returns a probability score between 0 and 1 for a given sample of data.** In fact, logistic regressing predicts the probability of that sample, and we map the cases to a discrete class based on that probability.

- Third, if your data is linearly separable. The decision boundary of logistic regression is a line or a plane or a hyper-plane.
  - A classifier will classify all the points on one side of the decision boundary as belonging to one class and all those on the other side as belonging to the other class.
  - For example, if we have just two features (and are not applying any polynomial processing), we can obtain an inequality like
$$\theta_0 + \theta_1 x_1 + \theta_2 x_2 > 0$$
which is a half-plane, easily plottable.
  - Please note that in using logistic regression, we can also achieve a complex decision boundary using polynomial processing as well. You'll get more insight from decision boundaries when you understand how logistic regression works.

- Fourth, you need to understand the impact of a feature.
  - You can select the best features based on the statistical significance of the logistic regression model coefficients or parameters.
  - That is, after finding the optimum parameters, a feature  $x$  with the weight  $\theta_1$  close to 0, has a smaller effect on the prediction, than features with large absolute values of  $\theta_1$ .
  - Indeed, it allows us to understand the impact an independent variable has on the dependent variable while controlling other independent variables.

**X**



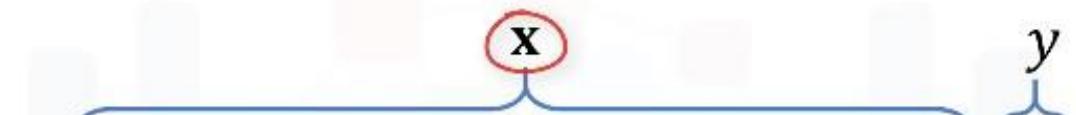
**y**

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	Yes
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	Yes
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	No
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	No

# Example

- We define the independent variables as X, and dependent variable as Y.
- Notice that, for the sake of simplicity, we can code the target or dependent values to 0 or 1.
- The goal of logistic regression is to build a model to predict the class of each sample (which in this case is a customer) as well as the probability of each sample belonging to a class.
- Given that, let's start to formalize the problem.
- X is our dataset, in the space of real numbers of m by n, that is, of m dimensions or features and n records.

**X**



	tenure	age	address	income	ed	employ	equip	ccallcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1.0
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1.0
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0.0
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0.0

$X \in \mathbb{R}^{m \times n}$   
 $y \in \{0,1\}$

And  $y$  is the class that we want to predict, which can be either zero or one.

Ideally, a logistic regression model, so called  $\hat{y}$  ( $y$ -hat), can predict that the class of a customer is 1, given its features  $x$ .

$$\hat{y} = P(y=1|x)$$

It can also be shown quite easily, that the probability of a customer being in class 0 can be calculated as 1 minus the probability that the class of the customer is 1.

$$\hat{y} = P(y=1|x)$$

$$P(y=0|x) = 1 - P(y=1|x)$$

# Logistic Regression Vs Linear Regression

- We go over linear regression and see why it cannot be used properly for some binary classification problems. We will also look at the Sigmoid function, which is the main part of logistic regression.
- Let's look at the telecommunication dataset again.

	x										y
	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn	
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	Yes	
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	Yes	
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	No	
3	38.0	35.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	No	
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	No	

$$\hat{y} = P(y=1|x)$$

# Logistic Regression Vs Linear Regression



- The goal of logistic regression is to build a model to predict the class of each customer, and also the probability of each sample belonging to a class.
- Ideally, we want to build a model,  $\hat{y}$ , that can estimate that the class of a customer is 1, given its features,  $x$ .
- I want to emphasize that  $y$  is the “labels vector,” also called “actual values” that we would like to predict, and  $\hat{y}$  is the vector of the predicted values by our model.
- Mapping the class labels to integer numbers, can we use linear regression to solve this problem?

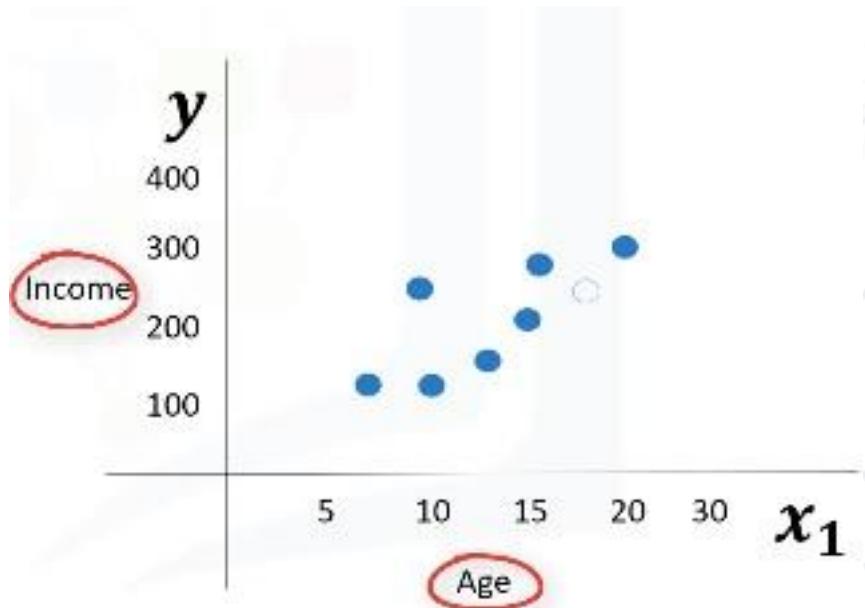
# Logistic Regression Vs Linear Regression

- First, let's recall how linear regression works to better understand logistic regression.
- Forget about the churn prediction for a minute, and assume our goal is to predict the income of customers in the dataset.
- This means that instead of predicting churn, which is a categorical value, let's predict income, which is a continuous value.
- So, how can we do this?
- Let's select an independent variable, such as customer age, and predict a dependent variable, such as income.

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	39.0	36.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

# Logistic Regression Vs Linear Regression

- Of course we can have more features, but for the sake of simplicity, let's just take one feature here.
- We can plot it, and show age as an independent variable, and income as the target value we would like to predict.

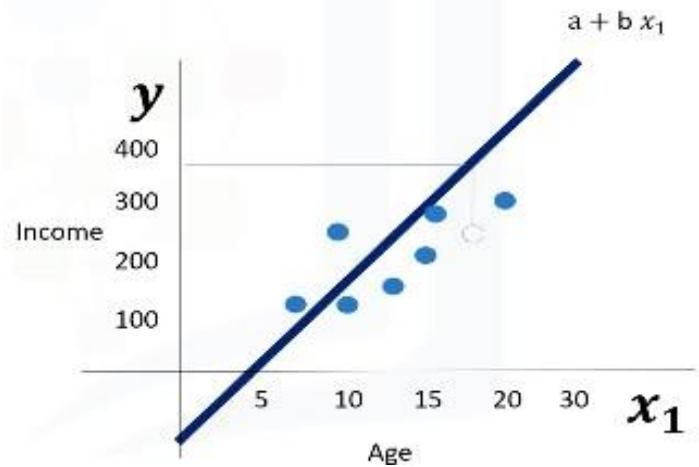




# Logistic Regression Vs Linear Regression

With linear regression, you can fit a line or polynomial through the data.

We can find this line through the training of our model, or calculating it mathematically based on the sample sets.



We'll say this is a straight line through the sample set. This line has an equation shown as  $a+b_x1$ .

Now, use this line to predict the continuous value  $y$ , that is, use this line to predict the income of an unknown customer based on his or her age. And it is done.

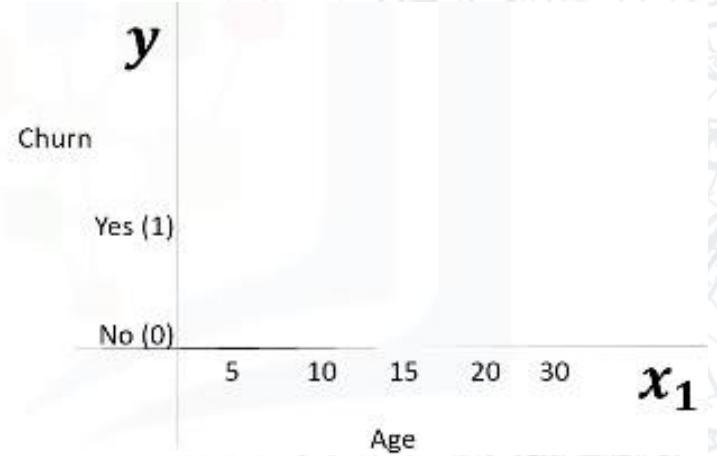
# Logistic Regression Vs Linear Regression

What if we want to predict churn? Can we use the same technique to predict a categorical field such as churn? OK, let's see.

Say we're given data on customer churn and our goal this time is to predict the churn of customers based on their age.

We have a feature, age denoted as  $x_1$ , and a categorical feature, churn, with two classes: churn is yes and churn is no.

	tenure	age	address	income	ed	employ	equip	callcard	wireless	churn
0	11.0	33.0	7.0	136.0	5.0	5.0	0.0	1.0	1.0	1
1	33.0	33.0	12.0	33.0	2.0	0.0	0.0	0.0	0.0	1
2	23.0	30.0	9.0	30.0	1.0	2.0	0.0	0.0	0.0	0
3	39.0	36.0	5.0	76.0	2.0	10.0	1.0	1.0	1.0	0
4	7.0	35.0	14.0	80.0	2.0	15.0	0.0	1.0	0.0	0

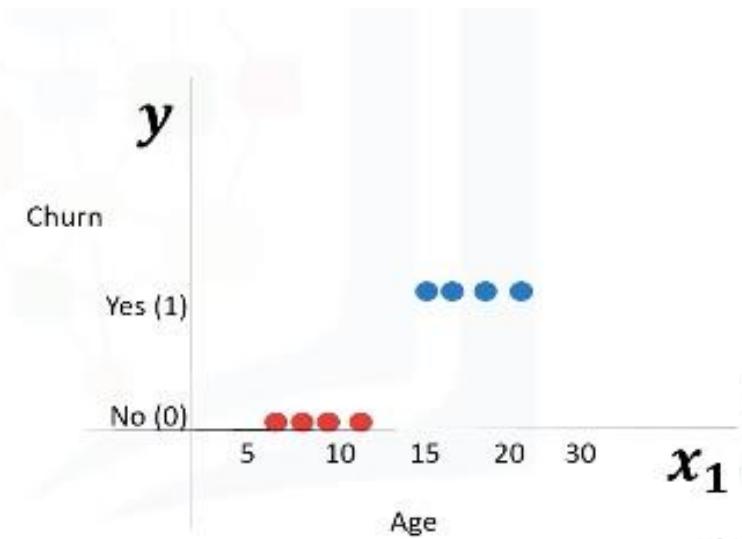




# Logistic Regression Vs Linear Regression

As mentioned, we can map yes and no to integer values, 0 and 1.

How can we model it now? Well, graphically, we could represent our data with a scatter plot. But this time we have only 2 values for the y axis.



In this plot, class zero is denoted in red and class one is denoted in blue.

Our goal here is to make a model based on existing data, to predict if a new customer is red or blue.

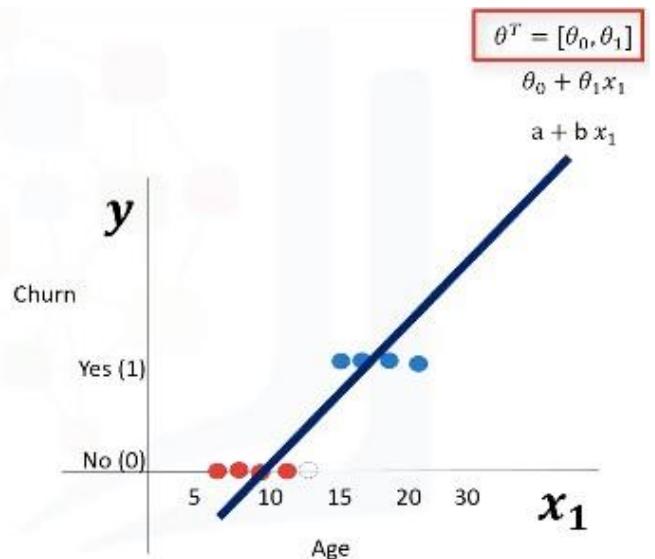
Let's do the same technique that we used for linear regression here to see if we can solve the problem for a categorical attribute, such as churn.



# Logistic Regression Vs Linear Regression

With linear regression, you again can fit a polynomial through the data, which is shown traditionally as  $a + b x$ .

This polynomial can also be shown traditionally as  $\theta_0 + \theta_1 x_1$ . This line has 2 parameters, which are shown with vector  $\theta$ , where the values of the vector are  $\theta_0$  and  $\theta_1$ .



# Logistic Regression Vs Linear Regression

We can also show the equation of this line formally as  $\theta^T X$ .

And generally, we can show the equation for a multi-dimensional space as  $\theta^T X$ , where  $\theta$  is the parameters of the line in 2-dimensional space, or parameters of a plane in 3-dimensional space, and so on.

As  $\theta$  is a vector of parameters, and is supposed to be multiplied by  $X$ , it is shown conventionally as  $T^{\theta}$ .  $\theta$  is also called the “weight vector” or “confidences of the equation” – with both of these terms used interchangeably.

$$\theta^T X = \theta_0 + \theta_1 x_1$$

$$\theta^T X = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots$$

$$\theta^T = [\theta_0, \theta_1, \theta_2, \dots]$$

# Logistic Regression Vs Linear Regression

And X is the feature set, which represents a customer.

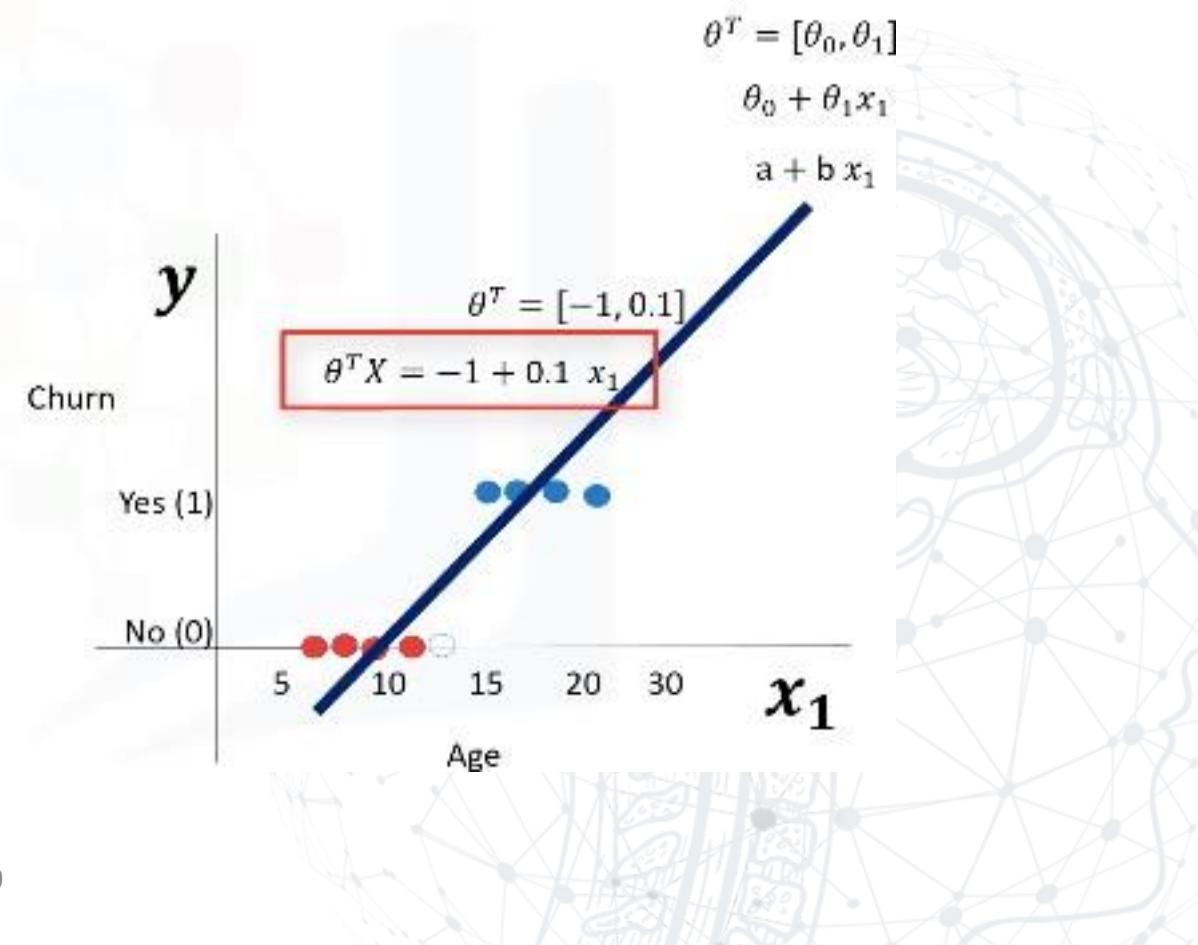
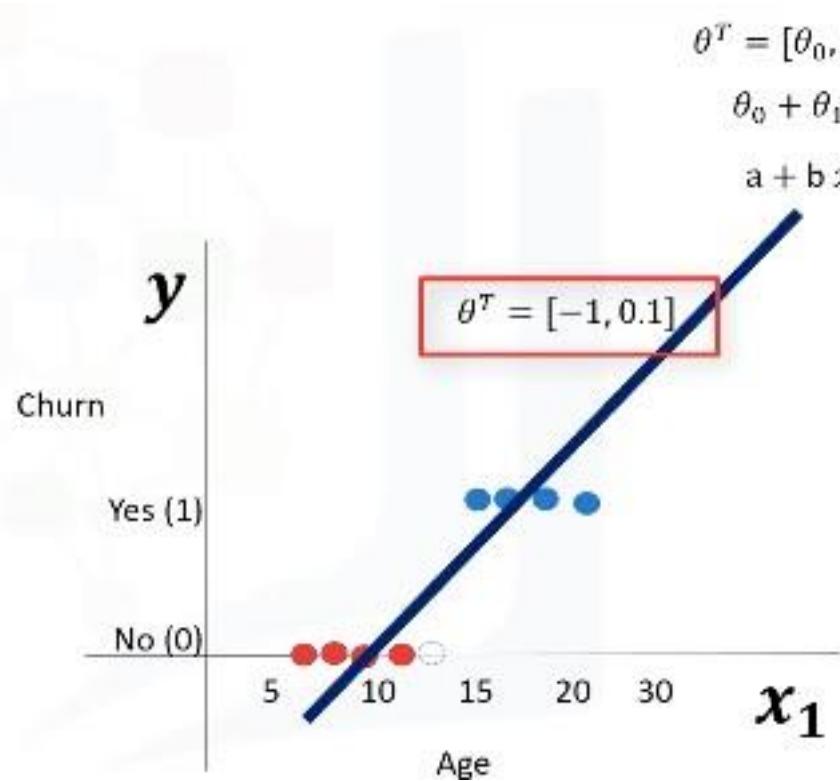
Anyway, given a dataset, all the feature sets X,  $\theta$  parameters, can be calculated through an optimization algorithm or mathematically, which results in the equation of the fitting line.

$$X = \begin{bmatrix} 1 \\ x_1 \\ x_2 \\ \dots \end{bmatrix}$$



## Logistic Regression Vs Linear Regression

For example, the parameters of this line are -1 and 0.1 and the equation for the line is  $-1 + 0.1 \times x_1$ .



# Logistic Regression Vs Linear Regression

- Now, we can use this regression line to predict the churn of the new customer.
- For example, for our customer, or let's say a data point with x value of age=13, we can plug the value into the line formula, and the y value is calculated and returns a number

$$\theta^T X = \theta_0 + \theta_1 x_1$$

$$p_1 = [13] \rightarrow \theta^T X = -1 + 0.1 \cdot x_1 \\ = -1 + 0.1 \times 13 \\ = 0.3$$

## Logistic Regression Vs Linear Regression

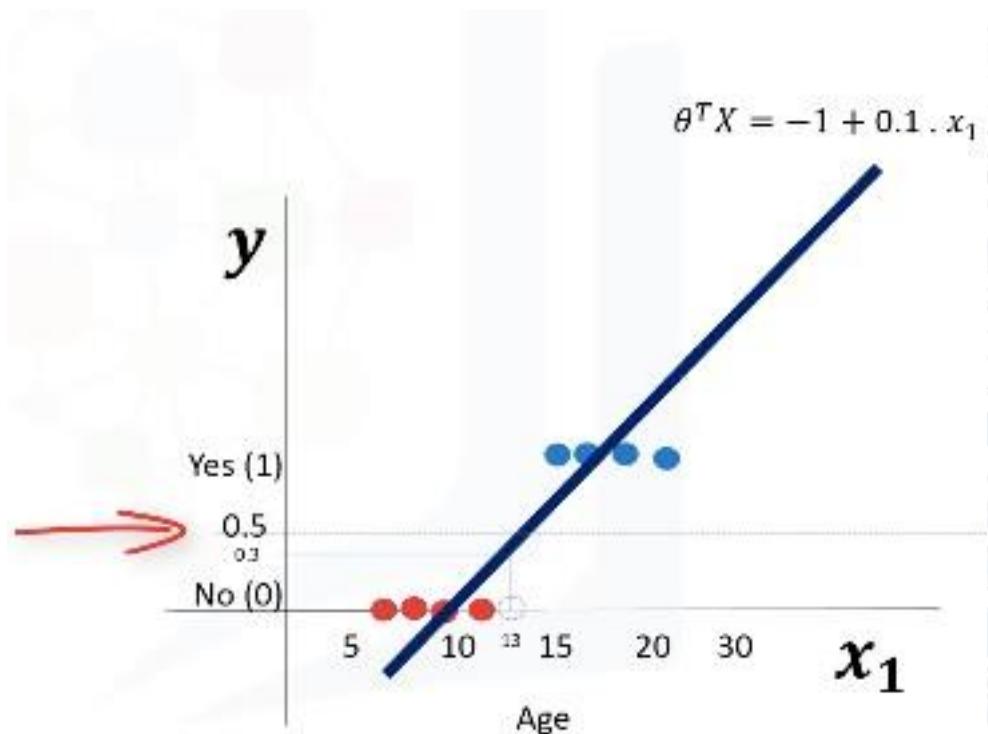
For instance, for p1 point we have:  $\theta^T X = -1 + 0.1 \times x_1 = -1 + 0.1 \times 13 = 0.3$  We can show it on our graph.

Now we can define a threshold here, for example at 0.5, to define the class.

$$\theta^T X = \theta_0 + \theta_1 x_1$$

$$p_1 = [13] \quad \rightarrow$$

$$\begin{aligned} \theta^T X &= -1 + 0.1 \cdot x_1 \\ &= -1 + 0.1 \times 13 \\ &= 0.3 \end{aligned}$$



# Logistic Regression Vs Linear Regression

So, we write a rule here for our model,  $\hat{y}$ , which allows us to separate class 0 from class 1.

If the value of  $\theta^T X$  is less than 0.5, then the class is 0, otherwise, if the value of  $\theta^T X$  is more than 0.5, then the class is 1.

And because our customer's  $y$  value is less than the threshold, we can say it belongs to class 0, based on our model.

$$\theta^T X = \theta_0 + \theta_1 x_1$$

$$p_1 = [13] \rightarrow \theta^T X = -1 + 0.1 \cdot x_1 \\ = -1 + 0.1 \times 13 \\ = 0.3$$

$$\hat{y} = \begin{cases} 0 & \text{if } \theta^T X < 0.5 \\ 1 & \text{if } \theta^T X \geq 0.5 \end{cases}$$

$\theta^T X = 0.3$   
 $\theta^T X < 0.5 \rightarrow \text{Class 0}$

# Logistic Regression Vs Linear Regression



- But there is one problem here; what is the probability that this customer belongs to class 0? As you can see, it's not the best model to solve this problem.
- Also, there are some other issues, which verify that linear regression is not the proper method for classification problems.



# Logistic Regression Vs Linear Regression

So, as mentioned, if we use the regression line to calculate the class of a point, it always returns a number, such as 3, or -2, and so on.

$$\theta^T X = \theta_0 + \theta_1 x_1 + \dots$$

Then, we should use a threshold, for example, 0.5, to assign that point to either class of 0 or 1.

This threshold works as a step function that outputs 0 or 1, regardless of how big or small, positive or negative, the input is.

# Logistic Regression Vs Linear Regression

So, using the threshold, we can find the class of a record.

Notice that in the step function, no matter how big the value is, as long as it's greater than 0.5, it simply equals 1.

And vice versa, regardless of how small the value  $y$  is, the output would be zero if it is less than 0.5. In other words, there is no difference between a customer who has a value of one or 1000; the outcome would be 1.

$$\hat{y} = \begin{cases} 0 & \text{if } \theta^T X < 0.5 \\ 1 & \text{if } \theta^T X \geq 0.5 \end{cases}$$

# Logistic Regression Vs Linear Regression

Instead of having this step function, wouldn't it be nice if we had a smoother line – one that would project these values between zero and one?

Indeed, the existing method does not really give us the probability of a customer belonging to a class, which is very desirable.

We need a method that can give us the probability of falling in a class as well.

So, what is the scientific solution here?

Well, if instead of using  $\theta^T X$

$$\theta^T X = \theta_0 + \theta_1 x_1 + \dots$$

we use a specific function called sigmoid, then, sigmoid of  $\theta^T X$  gives us the probability of a point belonging to a class, instead of the value of  $y$  directly.

$$\sigma(\theta^T X) = \sigma(\theta_0 + \theta_1 x_1 + \dots)$$

# Logistic Regression Vs Linear Regression

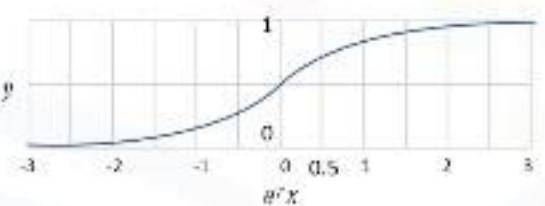
Sigmoid will be explained in a second, but for now, please accept that it will do the trick.

Instead of calculating the value of  $\theta^T X$  directly, it returns the probability that a  $\theta^T X$  is very big or very small.

It always returns a value between 0 and 1 depending on how large the  $\theta^T X$  actually is.

Now, our model is  $\sigma(\theta^T X)$ , which represents the probability that the output is 1, given x.

$$\sigma(\theta^T X) = \sigma(\theta_0 + \theta_1 x_1 + \dots)$$



$$\hat{y} = \sigma(\theta^T X)$$

$$P(y=1|x)$$



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# Sigmoid function

# Now the question is, “What is the sigmoid function?”

- Let's look in detail what sigmoid really is.
- The sigmoid function, also called the logistic function, resembles the step function and is used by the following expression in the logistic regression.
- The sigmoid function looks a bit complicated at first, but don't worry about remembering this equation. It'll make sense to you after working with it.

$$\sigma(\theta^T X) = \frac{1}{1 + e^{-\theta^T X}}$$

# Sigmoid

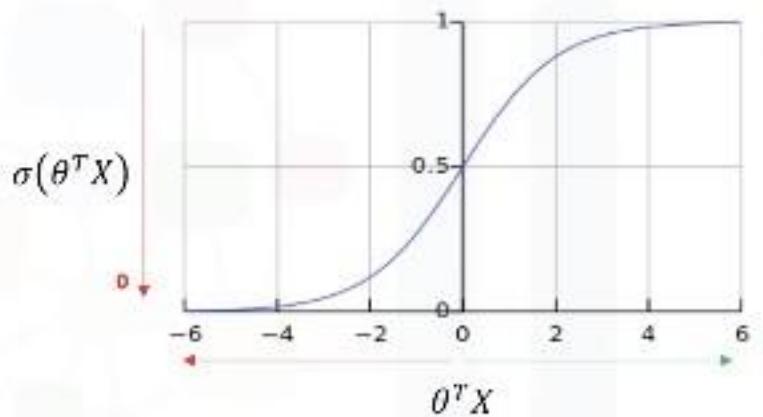
Notice that in the sigmoid equation, when  $\theta^T X$  gets very big, the  $e^{-\theta^T X}$  in the denominator of the fraction becomes almost zero, and the value of the sigmoid function gets closer to 1.

If  $\theta^T X$  is very small, the sigmoid function gets closer to zero.

$$\sigma(\theta^T X) = \frac{1}{1 + e^{-\theta^T X}}$$

$$\sigma(\theta^T X) = 1$$

$$\sigma(\theta^T X) = 0$$



# Sigmoid

Depicting on the sigmoid plot, when  $\theta^T X$ , gets bigger, the value of the sigmoid function gets closer to 1, and also,

if the  $\theta^T X$  is very small, the sigmoid function gets closer to zero.

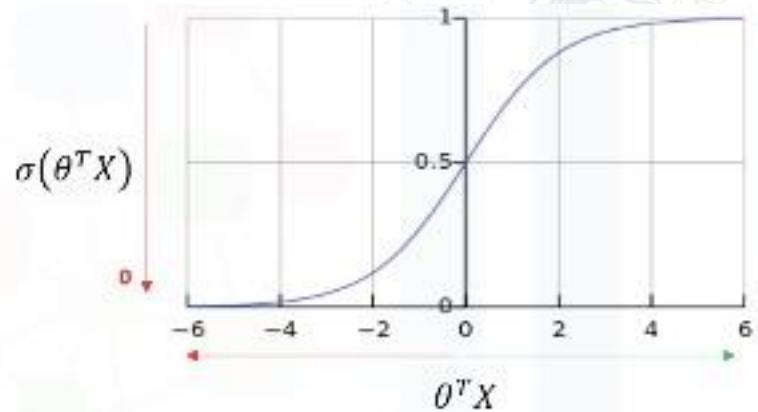
So, the sigmoid function's output is always between 0 and 1, which makes it proper to interpret the results as probabilities.

It is obvious that when the outcome of the sigmoid function gets closer to 1, the probability of  $y=1$ , given  $x$ , goes up, and in contrast, when the sigmoid value is closer to zero, the probability of  $y=1$ , given  $x$ , is very small.

$$\sigma(\theta^T X) = \frac{1}{1 + e^{-\theta^T X}}$$

$\sigma(\theta^T X) = 1$   
 $\sigma(\theta^T X) = 0$

[0, 1]



# Sigmoid

So what is the output of our model when we use the sigmoid function?

In logistic regression, we model the probability that an input (X) belongs to the default class (Y=1), and we can write this formally as,  $P(Y=1|X)$ .

We can also write  $P(y=0|X) = 1 - P(y=1|x)$ .

What is the output of our model?

- $P(Y=1|X)$
- $P(y=0|x) = 1 - P(y=1|x)$

For example, the probability of a customer staying with the company can be shown as probability of churn equals 1 given a customer's income and age, which can be, for instance, 0.8.

And the probability of churn is 0, for the same customer, given a customer's income and age can be calculated as  $1-0.8=0.2$ .

- $P(\text{Churn}=1|\text{income},\text{age}) = 0.8$
- $P(\text{Churn}=0|\text{income},\text{age}) = 1 - 0.8 = 0.2$

# Sigmoid

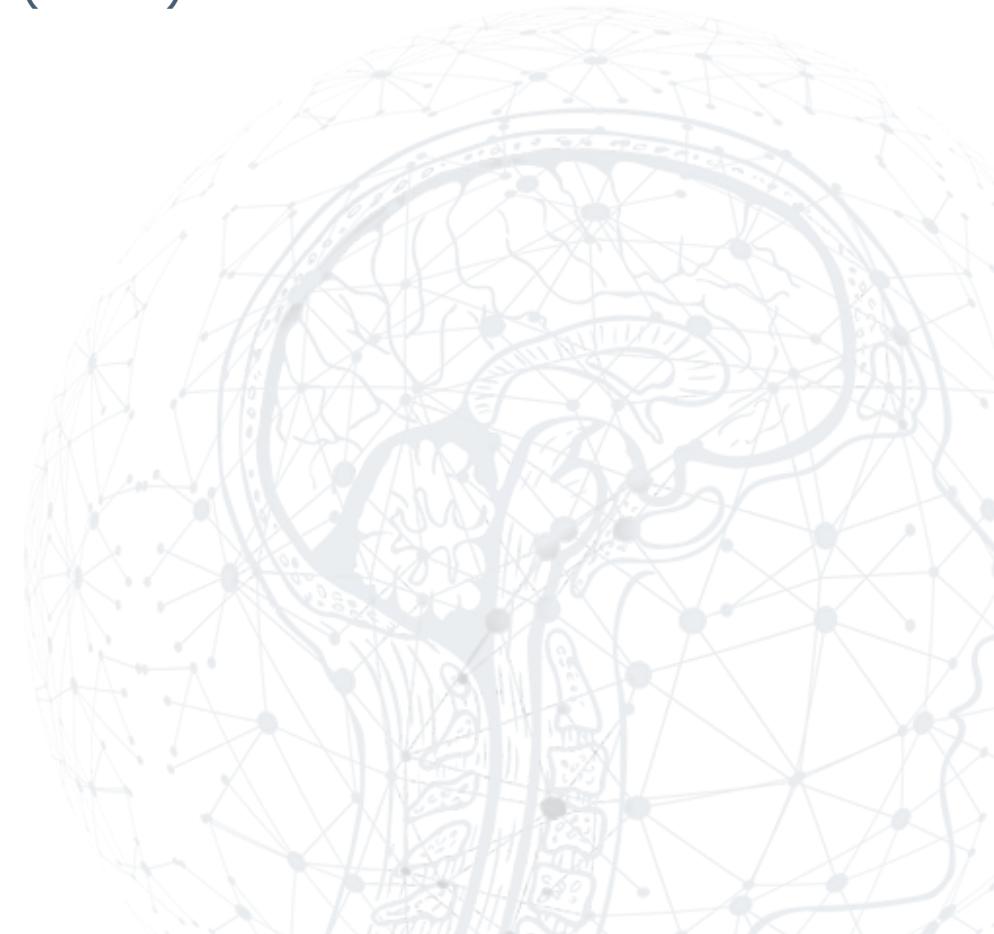
So, now, our job is to train the model to set its parameter values in such a way that our model is a good estimate of  $P(y=1|x)$ .

In fact, this is what a good classifier model built by logistic regression is supposed to do for us.

Also, it should be a good estimate of  $P(y=0|x)$  that can be shown as  $1 - \sigma(\theta^T X)$ .

$$\sigma(\theta^T X) \longrightarrow P(y=1|x)$$

$$1 - \sigma(\theta^T X) \longrightarrow P(y=0|x)$$





# Sigmoid

Now, the question is: "How we can achieve this?" We can find  $\theta$  through the training process, so let's see what the training process is.

Step 1. Initialize  $\theta$  vector with random values, as with most machine learning algorithms, for example -1 or 2.

$$\sigma(\theta^T X) \rightarrow P(y=1|x)$$

$$\theta = [-1, 2]$$

Step 2. Calculate the model output, which is  $\sigma(\theta^T X)$ , for a sample customer in your training set.

**X in  $\theta^T X$  is the feature vector values – for example, the age and income of the customer, for instance [2,5].**

**And  $\theta$  is the confidence or weight that you've set in the previous step. The output of this equation is the prediction value ... in other words, the probability that the customer belongs to class 1.**

Calculate  $\hat{y} = \sigma(\theta^T X)$  for a customer.

$$\hat{y} = \sigma([-1, 2] \times [2, 5]) = 0.7$$

# Sigmoid

Step 3. Compare the output of our model,  $\hat{y}$ , which could be a value of, let's say, 0.7, with the actual label of the customer, which is for example, 1 for churn.

**Then, record the difference as our model's error for this customer, which would be  $1-0.7$ , which of course equals 0.3. This is the error for only one customer out of all the customers in the training set.**

Compare the output of  $\hat{y}$  with actual output of customer,  $y$ , and record it as error.

$$\text{Error} = 1 - 0.7 = 0.3$$

Step. 4. Calculate the error for all customers as we did in the previous steps, and add up these errors.

**The total error is the cost of your model, and is calculated by the model's cost function.**

**The cost function, by the way, basically represents how to calculate the error of the model, which is the difference between the actual and the model's predicted values.**

**So, the cost shows how poorly the model is estimating the customer's labels. Therefore, the lower the cost, the better the model is at estimating the customer's labels correctly. And so, what we want to do is to try to minimize this cost.**

$$\text{Cost} = J(\theta)$$

# Sigmoid

Step 5. But, because the initial values for  $\theta$  were chosen randomly, it's very likely that the cost function is **very high**.

**So, we change the  $\theta$  in such a way to hopefully reduce the total cost.**

Step 6. After changing the values of  $\theta$ , we go back to step 2.

**Then we start another iteration, and calculate the cost of the model again.**

## The training process

1. Initialize  $\theta$ .
2. Calculate  $\hat{y} = \sigma(\theta^T X)$  for a customer.
3. Compare the output of  $\hat{y}$  with actual output of customer,  $y$ , and record it as error.
4. Calculate the error for all customers.
5. Change the  $\theta$  to reduce the cost.
6. Go back to step 2.

$$\sigma(\theta^T X) \rightarrow P(y=1|x)$$

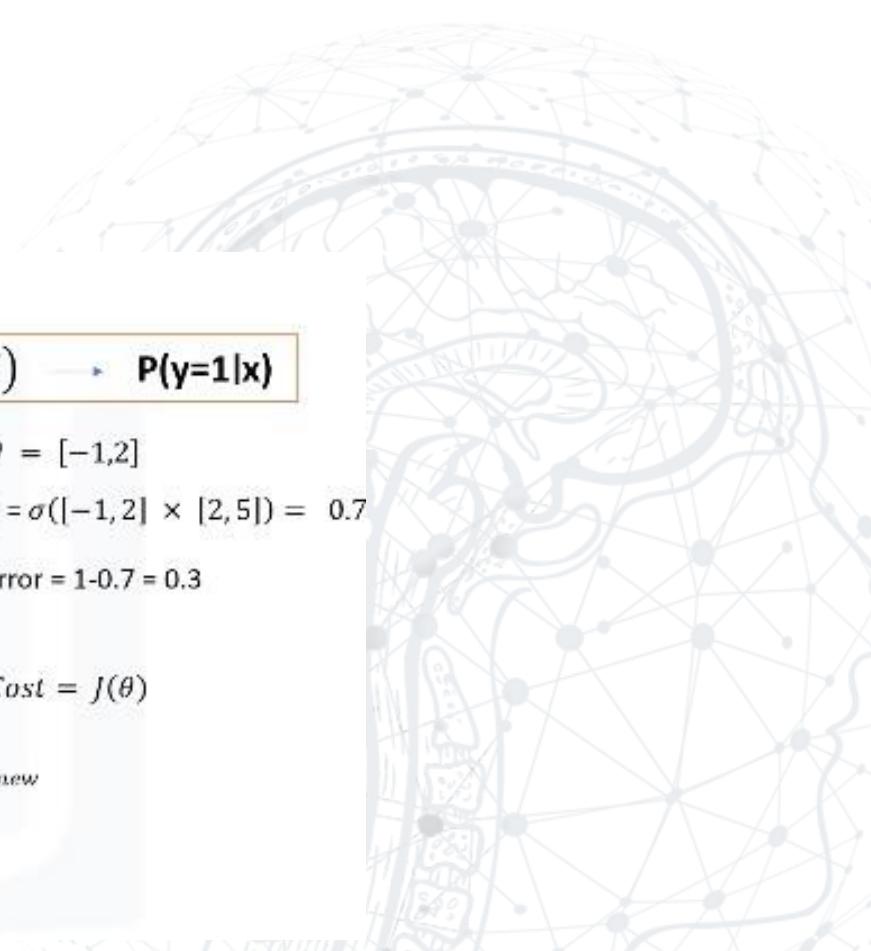
$$\theta = [-1, 2]$$

$$\hat{y} = \sigma([-1, 2] \times [2, 5]) = 0.7$$

$$\text{Error} = 1 - 0.7 = 0.3$$

$$Cost = J(\theta)$$

$$\theta_{new}$$



# Sigmoid

- And we keep doing those steps over and over, changing the values of  $\theta$  each time, until the cost is low enough. So, this brings up two questions:
- first, "How can we change the values of  $\theta$  so that the cost is reduced across iterations?" second, "When should we stop the iterations?"
- There are different ways to change the values of  $\theta$ , but one of the most popular ways is gradient descent.
- Also, there are various ways to stop iterations, but essentially you stop training by calculating the accuracy of your model and stop it when it's satisfactory.



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# Support Vector Machine (SVM)

# Support Vector Machines



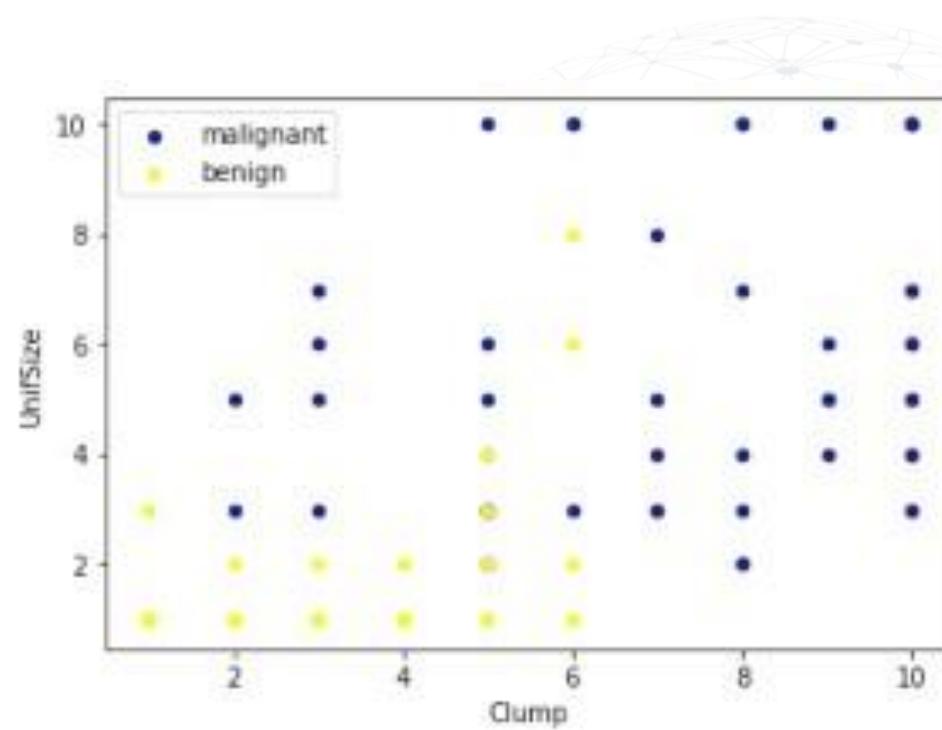
- A Support Vector Machine is a supervised algorithm that can classify cases **by finding a separator**.
- SVM works by:
  1. First, **mapping data to a high-dimensional feature space** so that data points can be categorized, even when the data are not otherwise linearly separable.
  2. Then, a **separator is estimated** for the data.
- The data should be transformed in such a way that a separator could be drawn as a hyperplane.

Imagine that you've obtained a dataset containing characteristics of thousands of human cell samples extracted from patients who were believed to be at risk of developing cancer.

Analysis of the original data showed that many of the characteristics differed significantly between **benign** and **malignant** samples.

For example, consider the following figure, which shows the distribution of a small set of cells, only based on their Unit Size and Clump thickness.

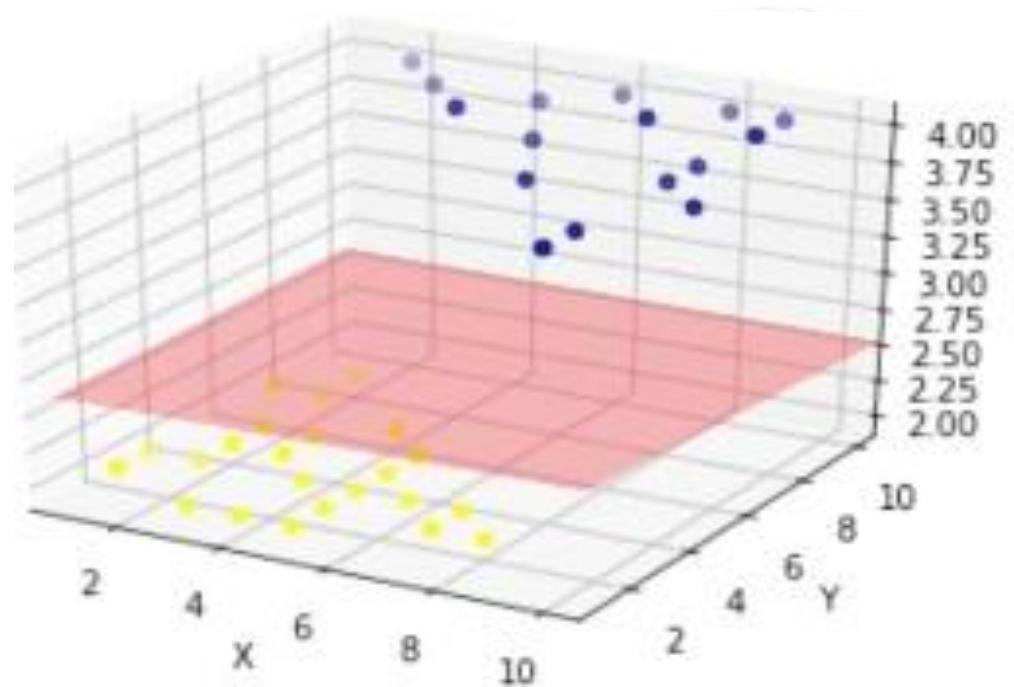
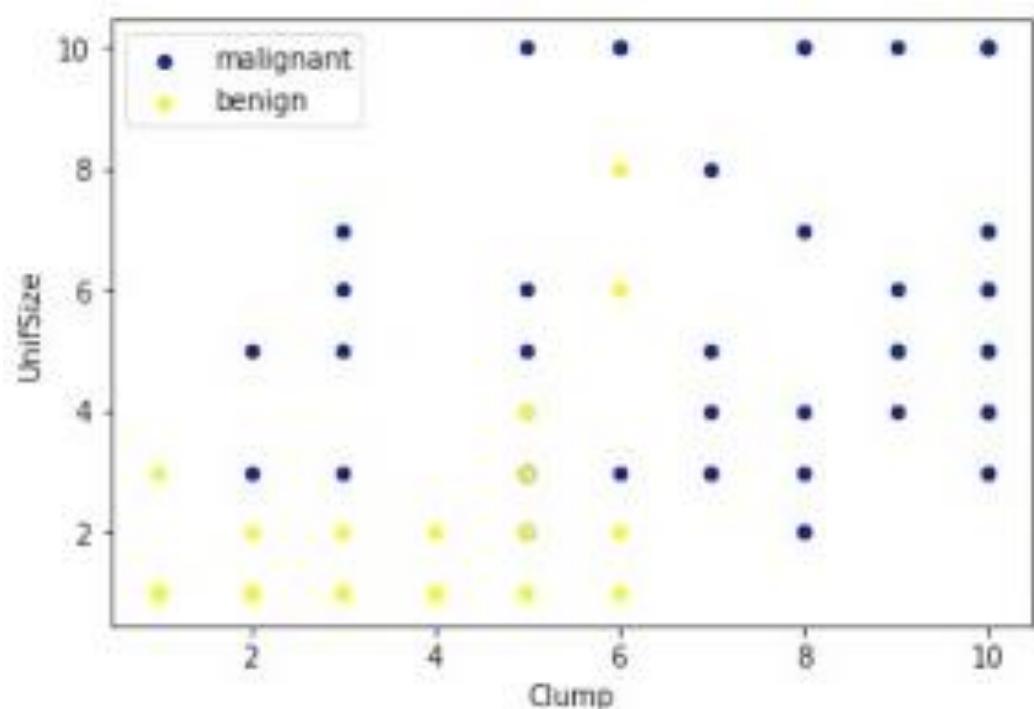
Clump	UnifSize	UnifShape	MargAdh	SingEpiSize	BareNuc	BlandChrom	NormNucl	Mit	Class
5	1	1	1	2	1	3	1	1	benign
5	4	4	5	7	10	3	2	1	benign
3	1	1	1	2	2	3	1	1	malignant
6	8	8	1	3	4	3	7	1	benign
4	1	1	3	2	1	3	1	1	benign
8	10	10	8	7	10		7	1	malignant
1	1	1	1	2	10	3	1	1	benign
2	1	2	H	2	1	3	1	1	benign
2	1	1	1	2	1	1	1	5	benign
4	2	1	1	2	1	2	1	1	benign



As you can see, the data points fall into two different categories. It represents a **linearly, non-separable**, dataset.

The two categories can be separated with a curve, but not a line.

We can **transfer this data to a higher dimensional space** ... for example, mapping it to a 3-dimensional space.



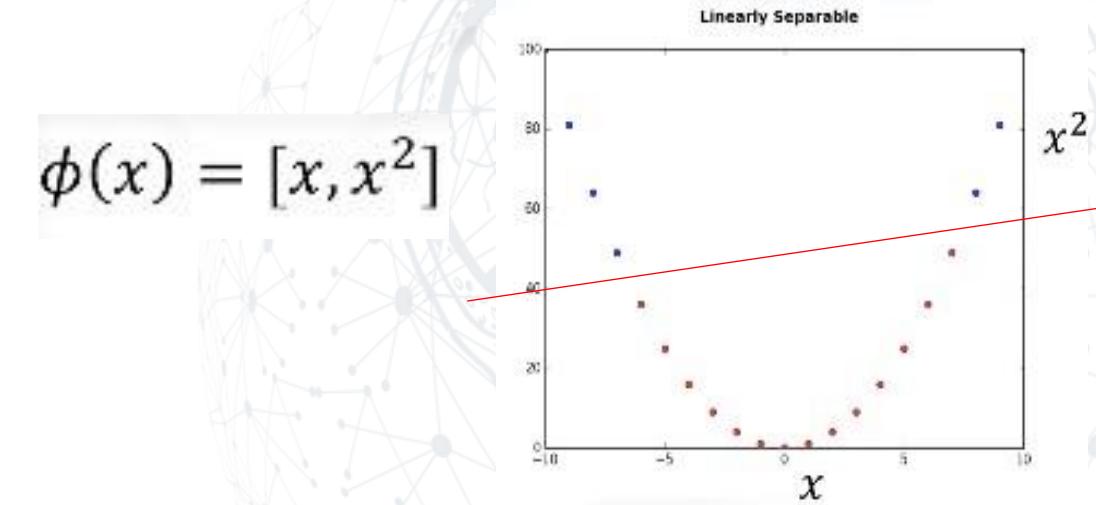
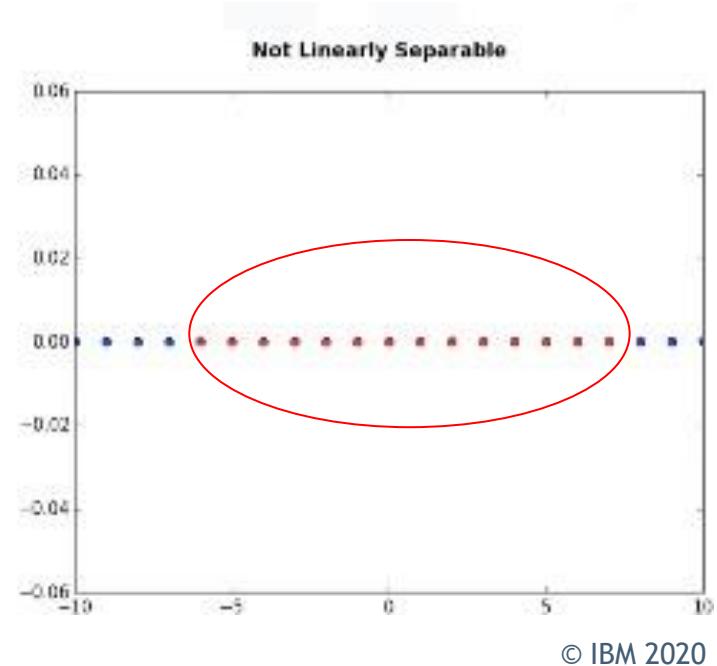
# Challenging questions



- How do we transfer data in such a way that a separator could be drawn as a hyperplane?
- How can we find the best/optimized hyperplane separator after transformation?

# Transforming data

- Imagine that our dataset is 1-dimensional data, this means, we have only one feature  $x$ .
- Transfer it into a 2-dimensional space, e.g. mapping  $x$  into a new space using a function, with outputs  $x$  and  $x$ -squared

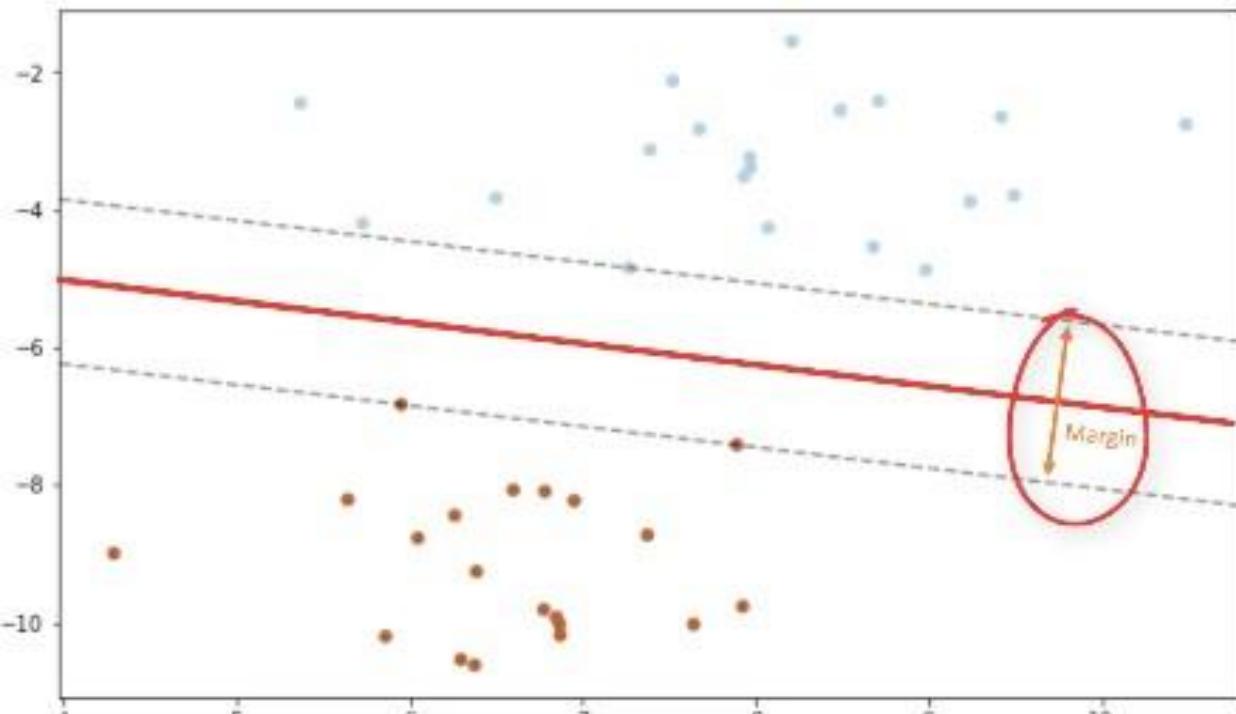


# SVM Kernel

- Basically, mapping data into a higher dimensional space is called kernelling.
- The mathematical function used for the transformation is known as the kernel function, and can be of different types, such as:
  - Linear, Polynomial, Radial basis function (or RBF), and Sigmoid.
- Each of these functions has its own characteristics, its pros and cons, and its equation, but the good news is that you don't need to know them, as most of them are already implemented in libraries of data science programming languages.
- Also, as there's no easy way of knowing which function performs best with any given dataset, we usually choose different functions in turn and compare the results.

# Hyperplane

- Basically, SVMs are based on the idea of finding a hyperplane that best divides a dataset into two classes, as shown here.
- One reasonable choice as the best hyperplane is **the one that represents the largest separation, or margin, between the two classes.**

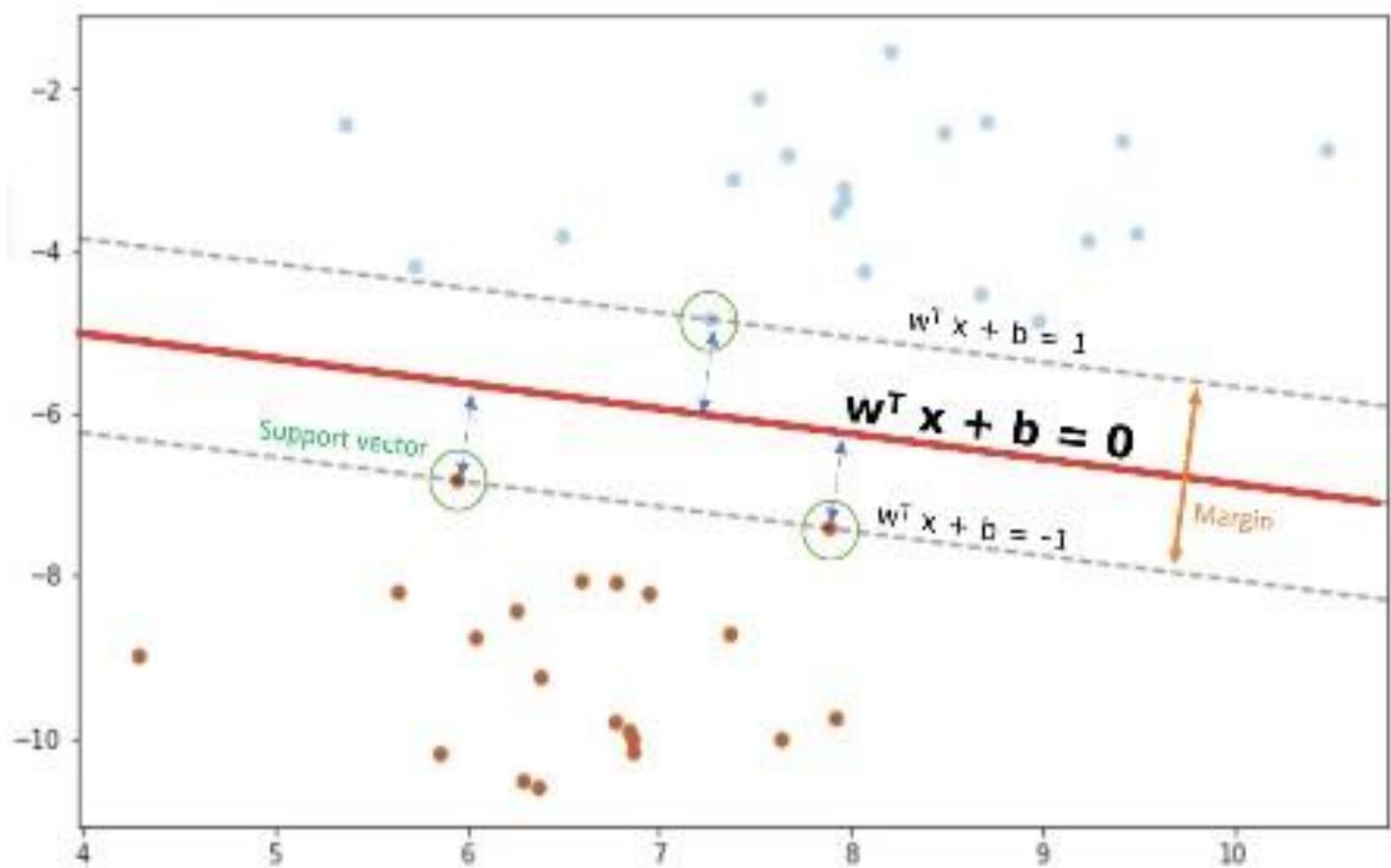


So, finding the optimized hyperplane can be formalized using an equation which involves quite a bit more math,

Find  $w$  and  $b$  such that

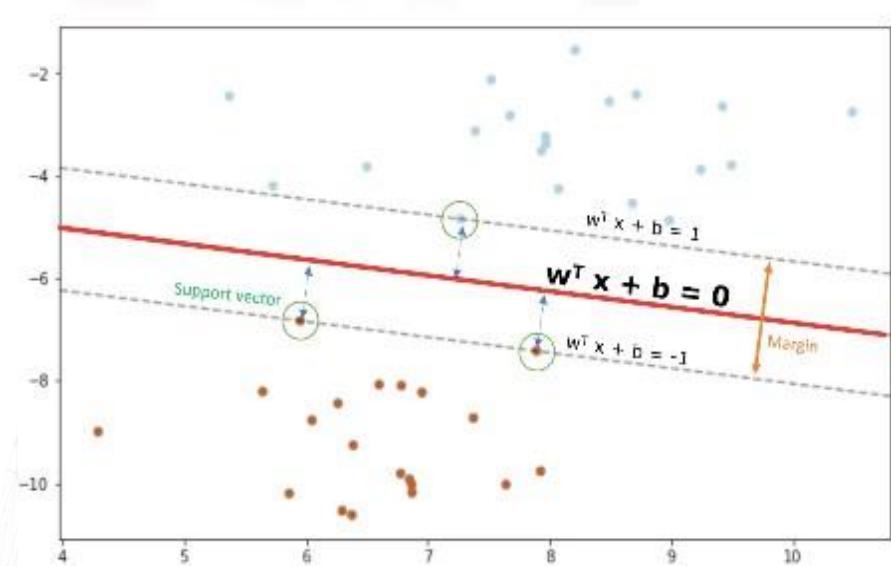
$\Phi(w) = \frac{1}{2} w^T w$  is minimized;

and for all  $\{(x_i, y_i)\}$ :  $y_i (w^T x_i + b) \geq 1$



# Support Vector Machines

- That said, the hyperplane is learned from training data using an optimization procedure that maximizes the margin;
- and like many other problems, this optimization problem can also be solved by gradient descent.
- Therefore, the output of the algorithm is the values 'w' and 'b' for the line. You can make classifications using this estimated line.
- It is enough to plug in input values into the line equation, then, you can calculate whether an unknown point is above or below the line.
- If the equation returns a value greater than 0, then the point belongs to the first class, which is above the line, and vice versa.



# Support Vector Machines

- The two main advantages of support vector machines are that they're accurate in high dimensional spaces; and, they use a subset of training points in the decision function (called support vectors), so it's also memory efficient.
- The disadvantages of support vector machines include the fact that the algorithm is prone for over-fitting, if the number of features is much greater than the number of samples.
- Also, SVMs do not directly provide probability estimates, which are desirable in most classification problems.
- And finally, SVMs are not very efficient computationally, if your dataset is very big, such as when you have more than one thousand rows.

- **Advantages:**

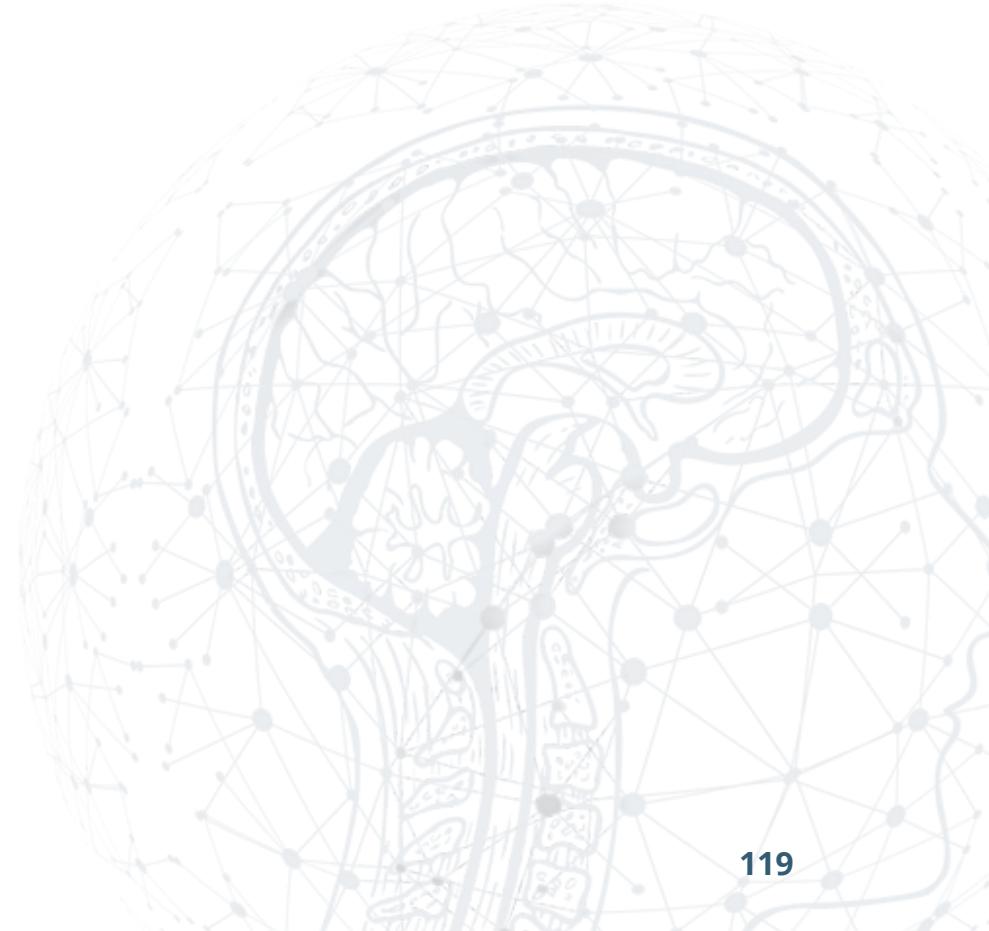
- Accurate in high-dimensional spaces
- Memory efficient

- **Disadvantages:**

- Prone to over-fitting
- No probability estimation
- Small datasets

# Reference

- IBM – Cognitiveclass



# Klasifikasi

PENGANTAR KECERDASAN BUATAN



UNIVERSITAS  
INDONESIA

Copyright © Universitas Indonesia

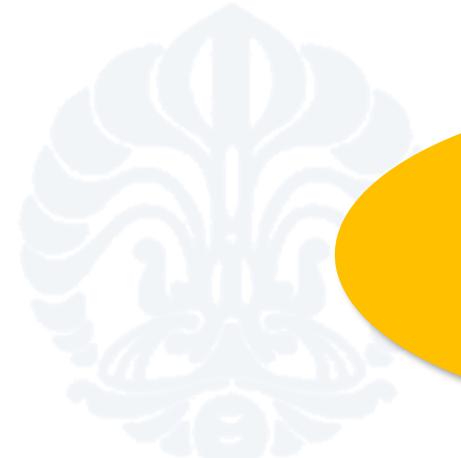
Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI



UNIVERSITAS  
INDONESIA

Universitas Indonesia

**Dr. Prima Dewi Purnamasari**

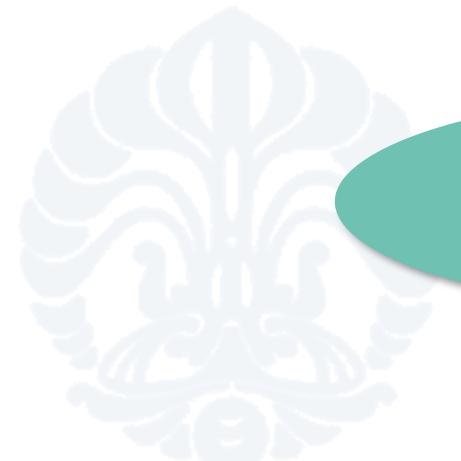


UNIVERSITAS  
INDONESIA

Universitas Indonesia

Sebelumnya:  
Regresi &  
gradient descent

**Dr. Prima Dewi Purnamasari**



UNIVERSITAS  
INDONESIA

Universitas Indonesia

Sekarang:  
Kelasifikasi

**Dr. Prima Dewi Purnamasari**

# Capaian pembelajaran

**Mampu menjelaskan metode  
kelasifikasi dalam pembelajaran mesin**

UNIVERSITAS  
INDONESIA

Copyright © Universitas Indonesia

# Klasifikasi

**Mempelajari hubungan antara sekumpulan variabel independent (disebut fitur) dan variabel dependen (disebut kelas/target)**

**Atribut target dalam klasifikasi adalah variabel kategorikal atau nilai diskrit.**

Copyright © Universitas Indonesia

# Bagaimana cara klasifikasi bekerja ?

Copyright © Universitas Indonesia

# Bunga Iris

## R.A. Fisher's tahun 1936



**Iris Versicolor**



**Iris Setosa**



**Iris Virginica**

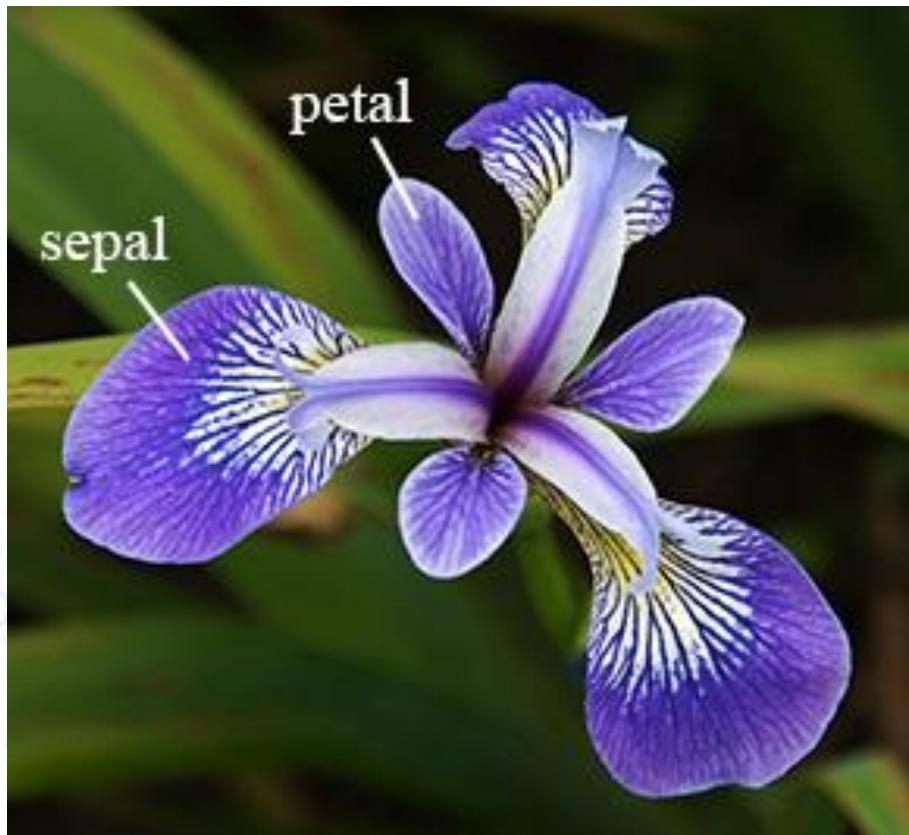
**sekilas mirip baik bentuk maupun warnanya**

<b>Id</b>	<b>SepalLengthCm</b>	<b>SepalWidthCm</b>	<b>PetalLengthCm</b>	<b>PetalWidthCm</b>	<b>Species</b>
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3	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	3.6	1.4	0.2	Iris-setosa
51	7	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	1.3	Iris-versicolor
55	6.5	2.8	4.6	1.5	Iris-versicolor
101	6.3	3.3	6	2.5	Iris-virginica
102	5.8	2.7	5.1	1.9	Iris-virginica
103	7.1	3	5.9	2.1	Iris-virginica
104	6.3	2.9	5.6	1.8	Iris-virginica
105	6.5	3	5.8	2.2	Iris-virginica

Bunga IRIS dapat dibedakan berdasarkan ciri:

- Sepal length
- Sepal width
- Petal length
- Petal width

Contoh data dari  
Fisher's Iris



<b>Id</b>	<b>SepalLengthCm</b>	<b>SepalWidthCm</b>	<b>PetalLengthCm</b>	<b>PetalWidthCm</b>	<b>Species</b>
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3	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	3.6	1.4	0.2	Iris-setosa
51	7	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	1.3	Iris-versicolor
55	6.5	2.8	4.6	1.5	Iris-versicolor
101	6.3	3.3	6	2.5	Iris-virginica
102	5.8	2.7	5.1	1.9	Iris-virginica
103	7.1	3	5.9	2.1	Iris-virginica
104	6.3	2.9	5.6	1.8	Iris-virginica
105	6.5	3	5.8	2.2	Iris-virginica



**FITUR (feature)**

4 fitur

**KELAS**

3 kelas

Bunga IRIS dapat dibedakan berdasarkan ciri:

- Sepal length
- Sepal width
- Petal length
- Petal width

# Tujuan Algoritma Klasifikasi

Jika diberikan informasi

- Sepal length
- Sepal width
- Petal length
- Petal width

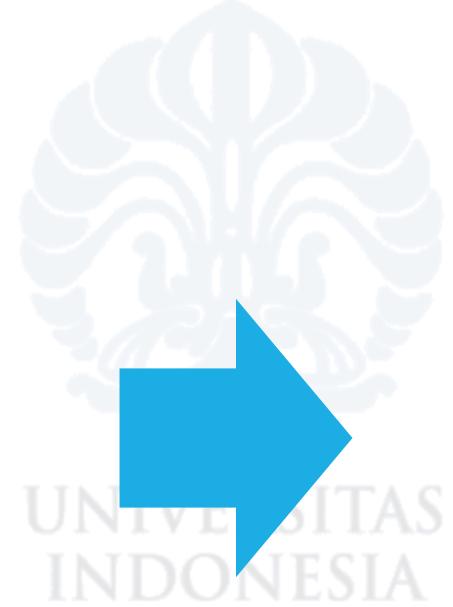


Dapat diketahui jenis Iris:

- Setosa
- Versicolor
- Virginica

# Tujuan Algoritma Klasifikasi

Jika diberikan  
informasi  
**FITUR**



Dapat diketahui  
**KELAS**

Copyright © Universitas Indonesia

# Algoritma Klasifikasi Populer

Decision Trees

Naïve Bayes

K-nearest  
neighbor

Logistic  
regression

Support Vector  
Machines

Neural  
Networks



# K-Nearest Neighbor (KNN)

Copyright © Universitas Indonesia



# KNN

**Metode untuk mengklasifikasi sebuah sampel data (*data testing*) berdasarkan kemiripannya dengan sampel data lain yang sudah memiliki label (*data training*)**

**Sampel yang mirip disebut “neighbor”**

# Data Fisher's Iris



UNIVERSITAS  
INDONESIA

**4 fitur  
3 kelas**

**masing-masing kelas: 50 sampel  
total 150 sampel**

Copyright © Universitas Indonesia

# MEMODELKAN MESIN

Membagi data yang dimiliki menjadi:

**data training  
dan  
data testing**

40 sampel → data training

10 sampel → data testing

40 sampel → data training

10 sampel → data testing

40 sampel → data training

10 sampel → data testing

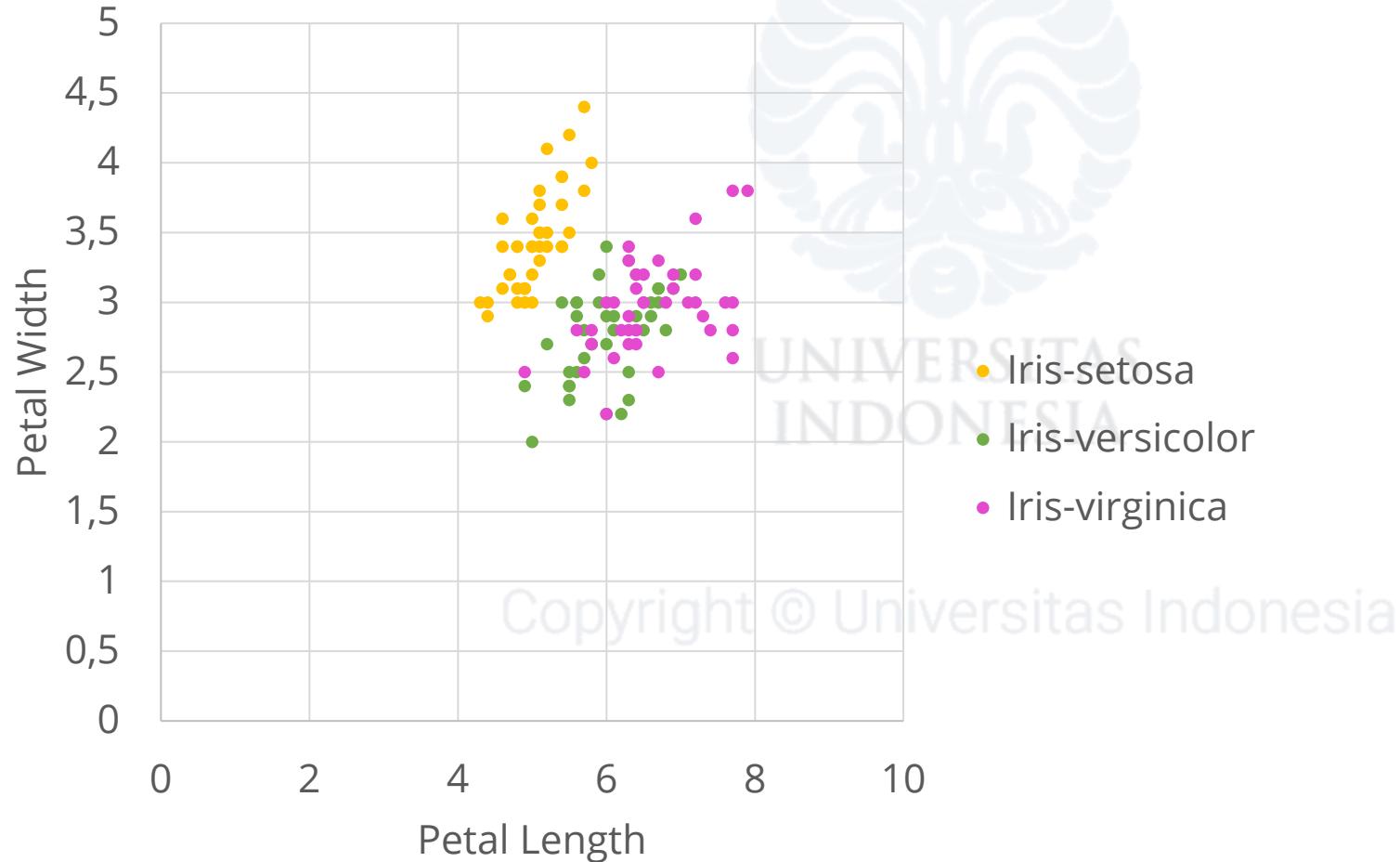
**SETOSA**

**VERSICOLOR**

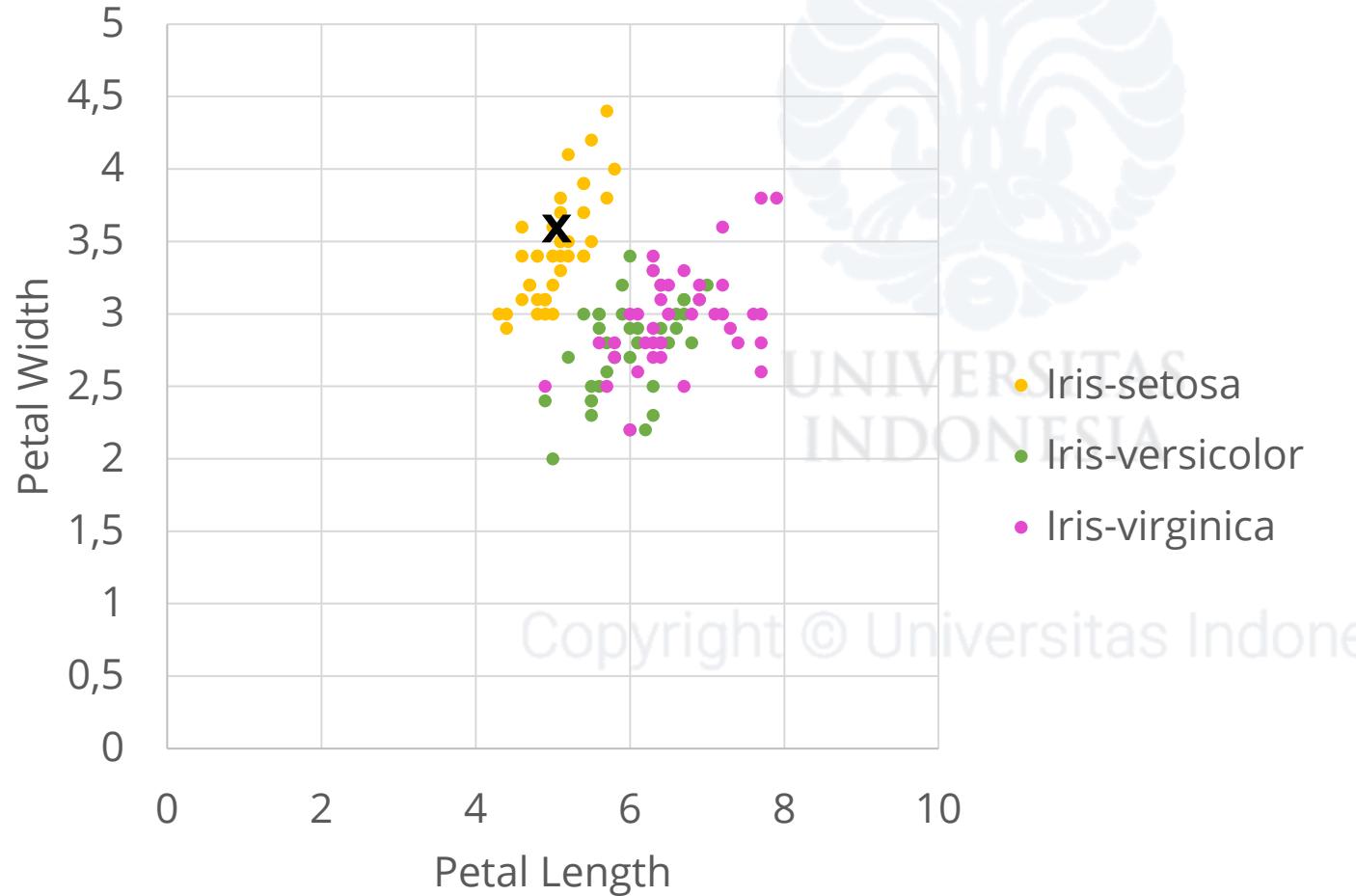
**VIRGINICA**

Copyright © Universitas Indonesia

**Untuk memudahkan visualisasi, ambil 2 fitur pertama: sepal length & sepal width**



# Untuk memudahkan visualisasi, ambil 2 fitur pertama: sepal length & sepal width



**Titik x** akan diklasifikasikan sebagai Setosa karena jika dihitung jaraknya paling dekat ke *neighbor* sampel data training Setosa

# Menghitung jarak 2 titik

**Euclidean distance**

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

**Manhattan distance**

$$d(x, y) = \sum_{i=1}^n |x_i - y_i|$$

Copyright © Universitas Indonesia

dengan:

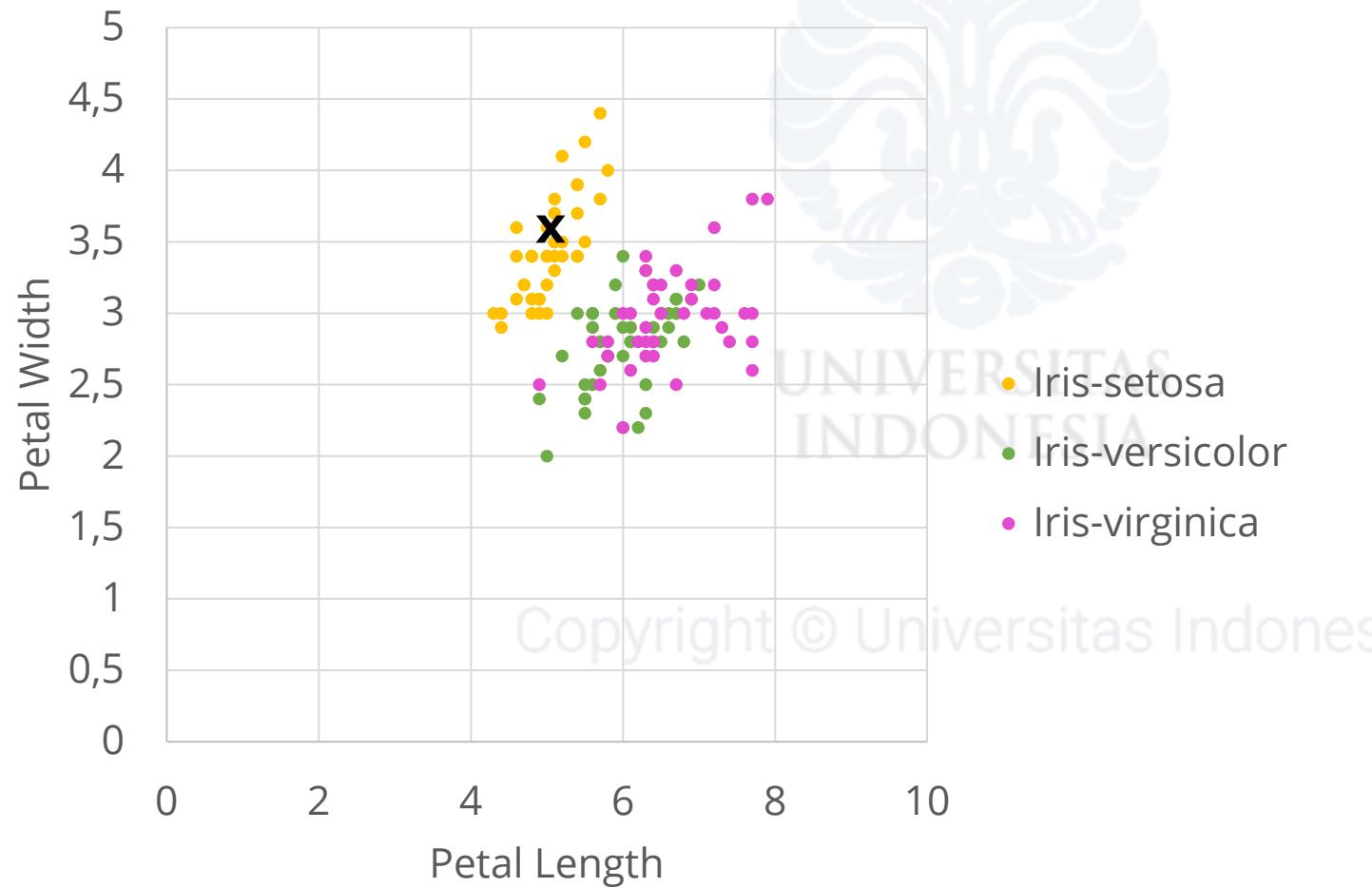
x = titik pertama

y = titik kedua

## Algoritma KNN

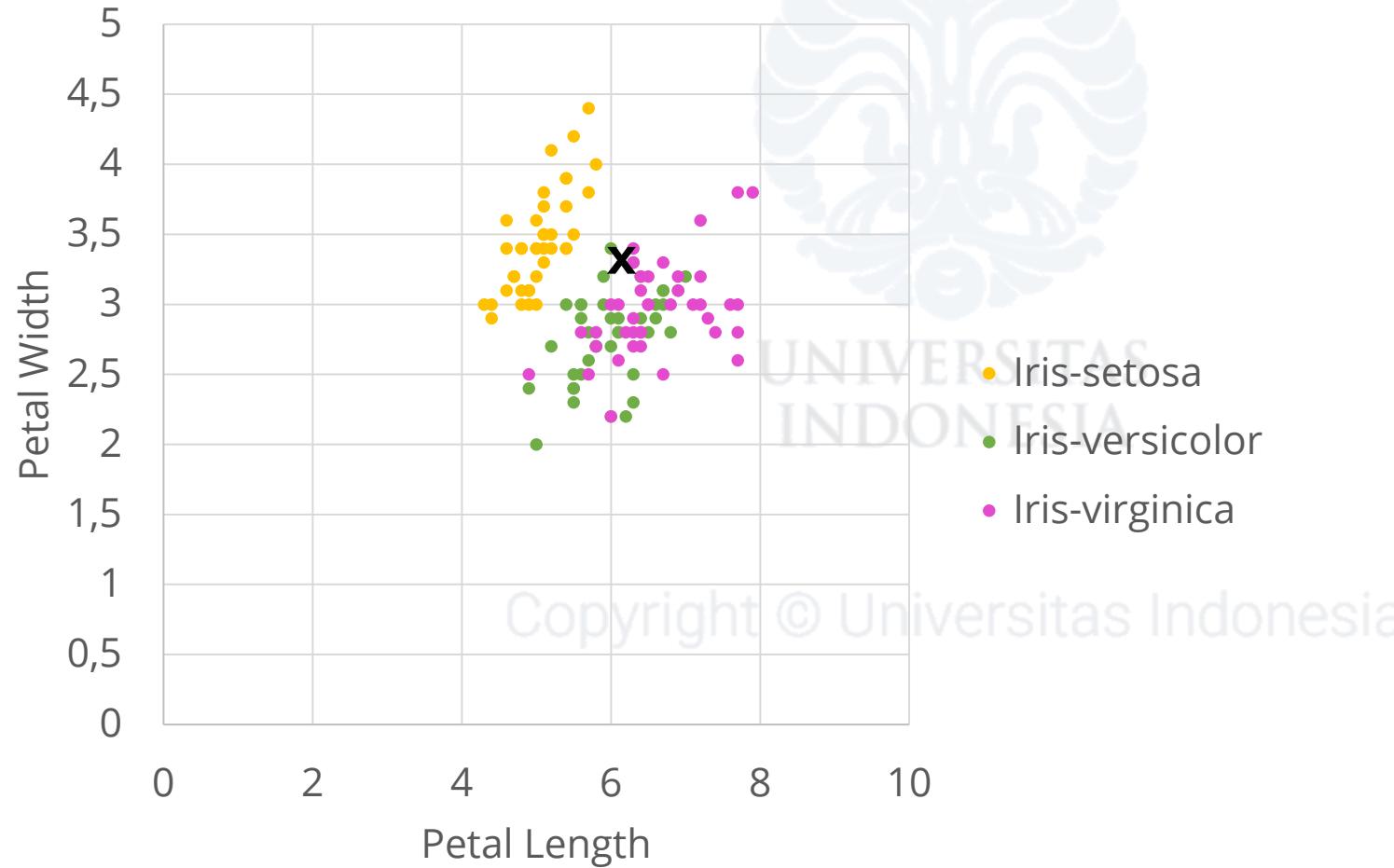
1. Pilih jumlah *neighbor* yang diinginkan → K
2. Ambil 1 titik baru yang ingin diketahui kelasnya
3. Hitung jarak dari semua data training ke titik baru tersebut
4. Urutkan titik data training dari jarak terdekat ke terjauh terhadap titik baru
5. Ambil sejumlah k titik di urutan teratas
6. Mayoritas kelas dari titik terdekat akan menjadi kelas untuk titik baru ini

## Lihat contoh sebelumnya



Misal  $k = 4$   
Dapat dihitung 4  
titik terdekat  
semua berwarna  
kuning → kelas  
Setosa

## Lihat contoh sebelumnya



Misal  $k = 4$

Titik x

- 3 neighbor pink
- 1 neighbor hijau

kelas = pink (Virginica)

# Bagaimana cara memilih nilai K yang tepat?

- Menghitung akurasi untuk setiap K yang dipilih
- Akurasi (paling sederhana) → *Recognition rate*

$$\text{Recognition rate (\%)} = \frac{\sum \text{data yang diklasifikasi dengan benar}}{\sum \text{data}}$$

Mulai dari  $k = 1$ , lalu tingkatkan  $k$ , dan lihat  $k$  mana yang terbaik untuk model yang dibuat

# Bagaimana cara kerja algoritma klasifikasi yang lain?

Copyright © Universitas Indonesia





UNIVERSITAS  
INDONESIA

Universitas Indonesia

Selanjutnya:  
Pengelompokan



# KECERDASAN BUATAN

## 4 | CLUSTERING

Dr. Prima Dewi Purnamasari  
Program Studi Teknik Komputer FTUI



UNIVERSITAS  
INDONESIA

*Veritas, Prudentia, Justitia*

# Pendahuluan

Materi berikut diambil dari Cognitivelass.ai

# Clustering

- Imagine that you have a customer dataset, and you need to apply customer segmentation on this historical data.
- Customer segmentation is the practice of partitioning a customer base into groups of individuals that have similar characteristics.

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	5	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA005	15.5	1

# Clustering

Knowing this information allows a business to devote more time and attention to retaining these customers.

Another group might include customers from non-profit organizations, and so on.

A general segmentation process is not usually feasible for large volumes of varied data.

Therefore, you need an analytical approach to deriving segments and groups from large data sets.



It is a significant strategy as it allows a business to target specific groups of customers so as to more effectively allocate marketing resources.



For example, one group might contain customers who are high-profit and low-risk, that is, more likely to purchase products, or subscribe for a service.

# Clustering

One of the most adopted approaches that can be used for customer segmentation is clustering.

Clustering can group data only “unsupervised,” based on the similarity of customers to each other.

It will partition your customers into mutually exclusive groups, for example, into 3 clusters.

The customers in each cluster are similar to each other demographically.



Customers can be grouped based on several factors: including age, gender, interests, spending habits, and so on.



Let's learn how to divide a set of customers into categories, based on characteristics they share.

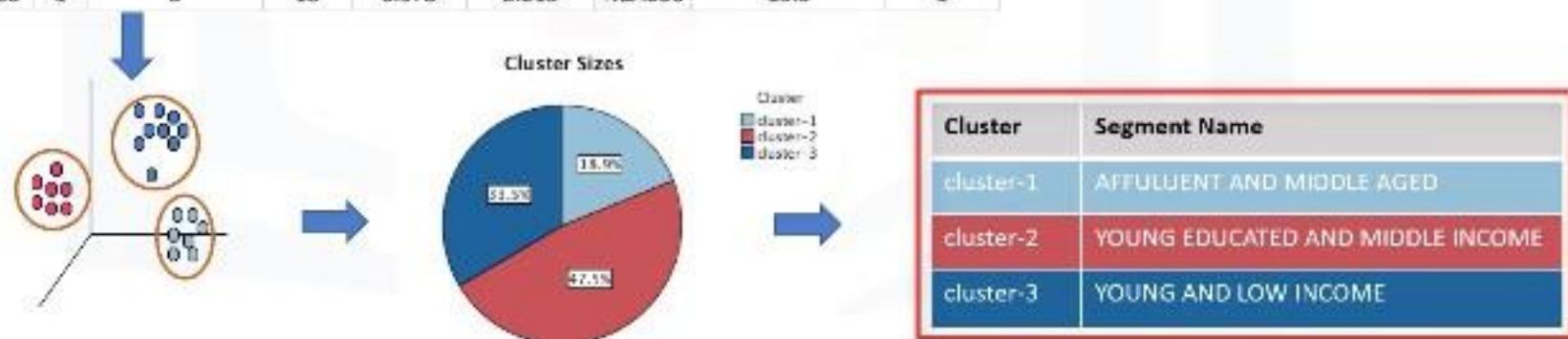


The important requirement is to use the available data to understand and identify how customers are similar to each other.

# Clustering

Now we can create a profile for each group, considering the common characteristics of each cluster.

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1



# Clustering

- For example, the first group is made up of AFFLUENT AND MIDDLE AGED customers. The second is made up of YOUNG EDUCATED AND MIDDLE INCOME customers. And the third group includes YOUNG AND LOW INCOME customers.
- Finally, we can assign each individual in our dataset to one of these groups or segments of customers.

Customer ID	Segment
1	YOUNG AND LOW INCOME
2	AFFLUENT AND MIDDLE AGED
3	AFFLUENT AND MIDDLE AGED
4	YOUNG AND LOW INCOME
5	AFFLUENT AND MIDDLE AGED
6	AFFLUENT AND MIDDLE AGED
7	YOUNG AND LOW INCOME
8	YOUNG AND LOW INCOME
9	AFFLUENT AND MIDDLE AGED

# Clustering

Customer segmentation is one of the popular usages of clustering.

Cluster analysis also has many other applications in different domains.

So let's first define clustering, and then we'll look at other applications.

Clustering means finding clusters in a dataset, unsupervised.



Now imagine that you cross-join this segmented dataset, with the dataset of the product or services that customers purchase from your company.



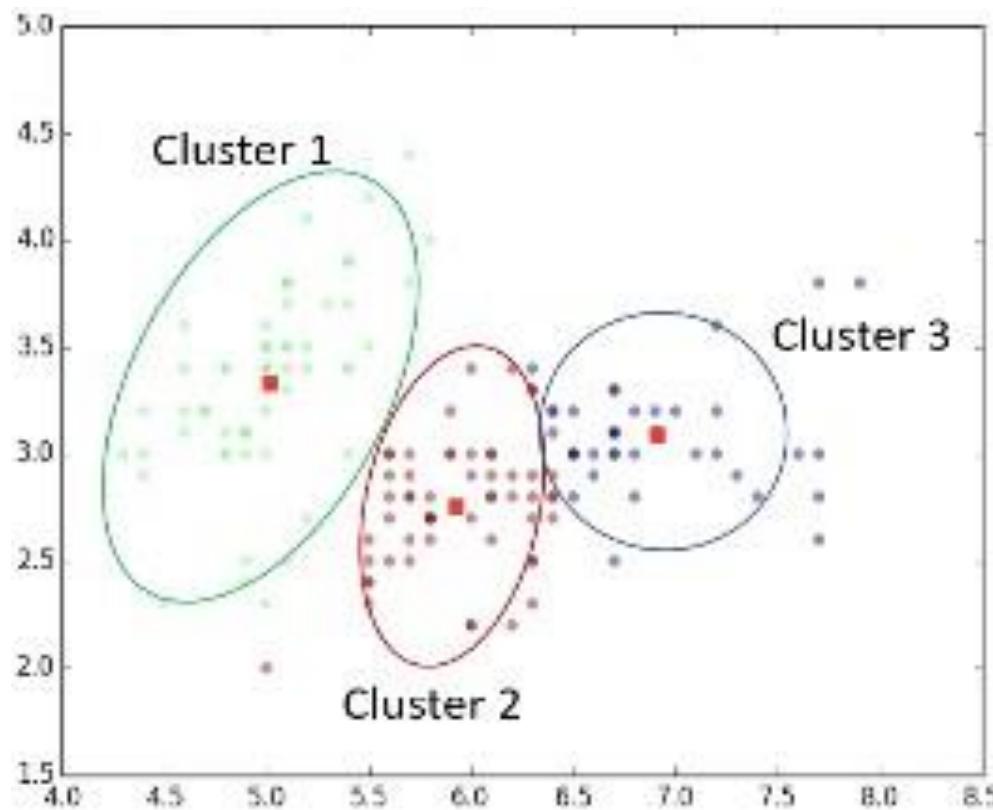
This information would really help to understand and predict the differences in individual customers' preferences and their buying behaviors across various products.



Indeed, having this information would allow your company to develop highly personalized experiences for each segment.

# Clustering

- Clustering means **finding clusters** in a dataset, **unsupervised**.
- So, what is a cluster? A cluster is group of data points or objects in a dataset that are **similar to other objects in the group**, and dissimilar to data points in other clusters.



# Clustering VS Classification

- Now, the question is, “What is different between clustering and classification?”
- Let’s look at our customer dataset again. Classification algorithms predict categorical class labels.

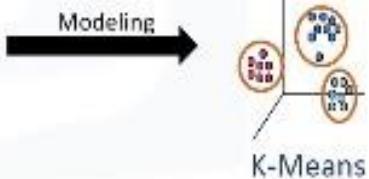
Labeled dataset

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	5	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.581	0.515	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1
10	47	3	23	115	0.653	3.947	NBA011	4	



Unlabeled dataset

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	5	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.581	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	64	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1



# Clustering VS Classification

- This means, assigning instances to pre-defined classes such as “Defaulted” or “Non-Defaulted.”
- For example, if an analyst wants to analyze customer data in order to know which customers might default on their payments, she uses a labeled dataset as training data, and uses classification approaches such as a decision tree, Support Vector Machines (or SVM), or, logistic regression to predict the default value for a new, or unknown customer.
- Generally speaking, classification is a supervised learning where each training data instance belongs to a particular class.
- In clustering, however, the data is unlabelled and the process is unsupervised.
- For example, we can use a clustering algorithm such as k-Means, to group similar customers as mentioned, and assign them to a cluster, based on whether they share similar attributes, such as age, education, and so on.

# Clustering Applications

- In the Retail industry, clustering is used to find associations among customers based on their demographic characteristics and use that information to identify buying patterns of various customer groups.
  - Also, it can be used in recommendation systems to find a group of similar items or similar users, and use it for collaborative filtering, to recommend things like books or movies to customers.
- In Banking, analysts find clusters of normal transactions to find the patterns of fraudulent credit card usage.
  - Also, they use clustering to identify clusters of customers, for instance, to find loyal customers, versus churn customers.
- In the Insurance industry, clustering is used for fraud detection in claims analysis, or to evaluate the insurance risk of certain customers based on their segments.
- In Publication Media, clustering is used to auto-categorize news based on its content, or to tag news, then cluster it, so as to recommend similar news articles to readers.
- In Medicine: it can be used to characterize patient behavior, based on their similar characteristics, so as to identify successful medical therapies for different illnesses.
- In Biology: clustering is used to group genes with similar expression patterns, or to cluster genetic markers to identify family ties.

# Why Clustering?

- If you look around, you can find many other applications of clustering, but generally, clustering can be used for one of the following purposes:
  - exploratory data analysis,
  - summary generation or reducing the scale,
  - outlier detection, especially to be used for fraud detection, or noise removal, finding duplicates in datasets,
  - pre-processing step for either prediction, other data mining tasks, or, as part of a complex system.
- Let's briefly look at different clustering algorithms and their characteristics.
- Partitioned-based clustering is a group of clustering algorithms that produces sphere-like clusters, such as k-Means, k-Median, or Fuzzy c-Means.
- These algorithms are relatively efficient and are used for Medium and Large sized databases.
- Hierarchical clustering algorithms produce trees of clusters, such as Agglomerative and Divisive algorithms.

# Why Clustering?



**This group of algorithms are very intuitive and are generally good for use with small size datasets.**



Density based clustering algorithms produce arbitrary shaped clusters.



They are especially good when dealing with spatial clusters or when there is noise in your dataset, for example, the DBSCAN algorithm.

Hierarchical clustering algorithms produce trees of clusters, such as Agglomerative and Divisive algorithms.



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# K-Means

# K-Means Clustering

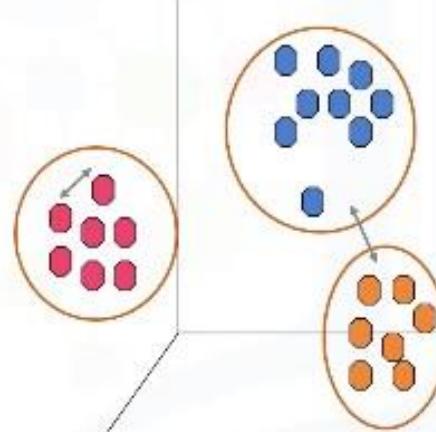
- Customer segmentation is the practice of partitioning a customer base into groups of individuals that have similar characteristics.
- One of the algorithms that can be used for customer segmentation is k-Means clustering.
- k-Means can group data only “unsupervised,” based on the similarity of customers to each other. Let’s define this technique more formally.
- Imagine that you have a customer dataset, and you need to apply customer segmentation on this historical data.

Customer Id	Age	Edu	Years Employed	Income	Card Debt	Other Debt	Address	DebtIncomeRatio	Defaulted
1	41	2	6	19	0.124	1.073	NBA001	6.3	0
2	47	1	26	100	4.582	8.218	NBA021	12.8	0
3	33	2	10	57	6.111	5.802	NBA013	20.9	1
4	29	2	4	19	0.681	0.516	NBA009	6.3	0
5	47	1	31	253	9.308	8.908	NBA008	7.2	0
6	40	1	23	81	0.998	7.831	NBA016	10.9	1
7	38	2	4	56	0.442	0.454	NBA013	1.6	0
8	42	3	0	54	0.279	3.945	NBA009	6.6	0
9	26	1	5	18	0.575	2.215	NBA006	15.5	1

# K-Means Clustering

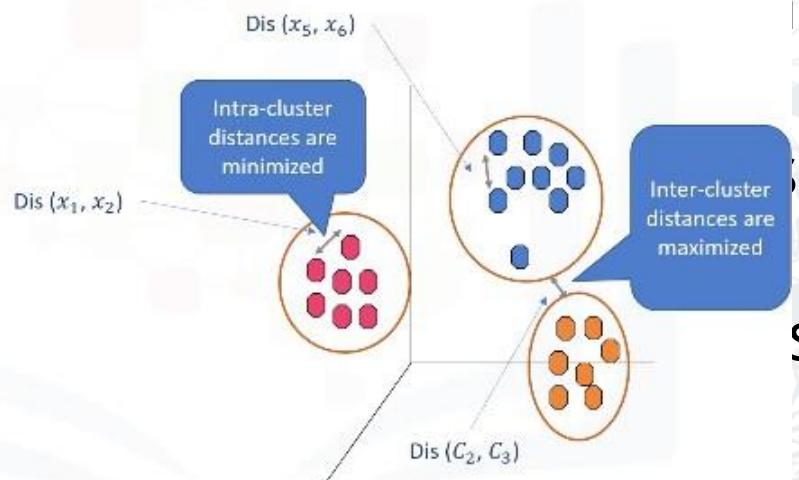
- There are various types of clustering algorithms, such as partitioning, hierarchical, or density-based clustering.
- k-Means is a type of partitioning clustering, that is, it divides the data into  $k$  non-overlapping subsets (or clusters) without any cluster-internal structure, or labels.
- This means, it's an unsupervised **k-Means algorithms**
- Objects within a cluster are very similar, while objects across different clusters are very different.  
for using k-Means, we have to define the number of clusters ( $k$ ).  
(e.g., 3 clusters of similar customers).

- Partitioning Clustering
- K-means divides the data into non-overlapping subsets (clusters) without any cluster-internal structure
  - Examples within a cluster are very similar
  - Examples across different clusters are very different



# K-Means Alogarithms

- Now we face a couple of key questions.
- First, “How can we find the similarity of samples in clustering?”
- And then, “How do we measure how similar two customers are with regard to their demographics?”
- Though the objective of k-Means is to form clusters in such a way that similar samples go into a cluster, and dissimilar samples fall into different clusters, it can be metric, we can use dissimilarity
- In other words, conventionally each other is used to shape the cluster
- So, we can say, k-Means tries to minimize the “intra-cluster distances and maximize the “inter-cluster distances”



# K-Means Alogarithms

Now, the question is, "How we can calculate the dissimilarity or distance of two cases, such as two customers?" Assume that we have two customers, we'll call them customer 1 and 2.



Let's also assume that we have only one feature for each of these two customers, and that feature is Age. We can easily use a specific type of Minkowski distance to calculate the distance of these two customers.

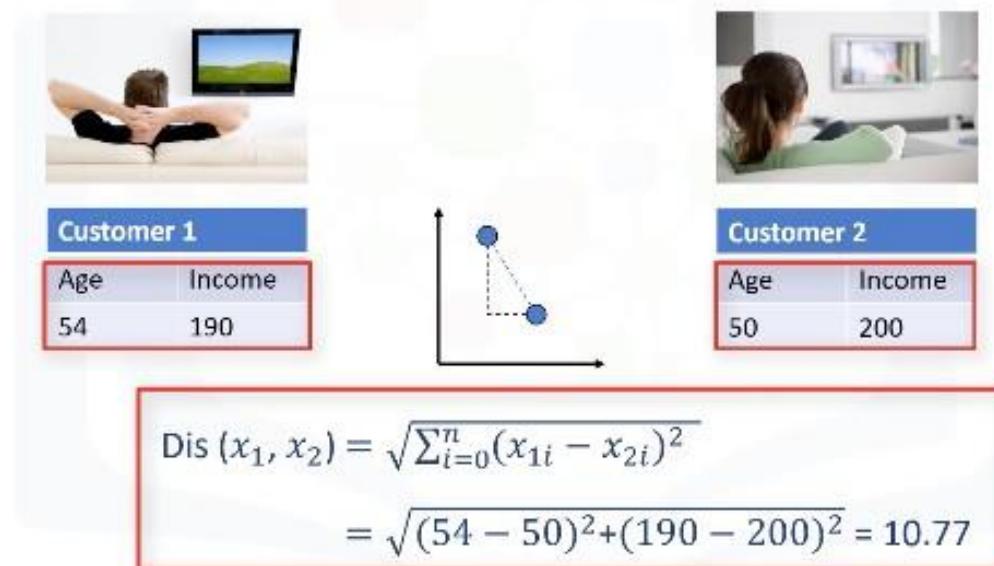
$$\text{Dis } (x_1, x_2) = \sqrt{\sum_{i=0}^n (x_{1i} - x_{2i})^2}$$

# K-Means Alogarithms

- Indeed, it is the Euclidian distance. Distance of  $x_1$  from  $x_2$ :

$$\text{Dis}(x_1, x_2) = \sqrt{(34 - 30)^2} = 4$$

- How about 2 feature? For example, if we have income and age for each customer, we can still use the same formula, but this time in a 2-dimensional space.



# K-Means Alogarithms

- Also, we can use the same distance matrix for multi-dimensional vectors.
- Of course, we have to normalize our feature set to get the accurate dissimilarity measure.

Customer 1			Customer 2		
Age	Income	education	Age	Income	education
54	190	3	50	200	8

$$\begin{aligned}
 \text{Dis} (x_1, x_2) &= \sqrt{\sum_{i=0}^n (x_{1i} - x_{2i})^2} \\
 &= \sqrt{(54 - 50)^2 + (190 - 200)^2 + (3 - 8)^2} = 11.87
 \end{aligned}$$

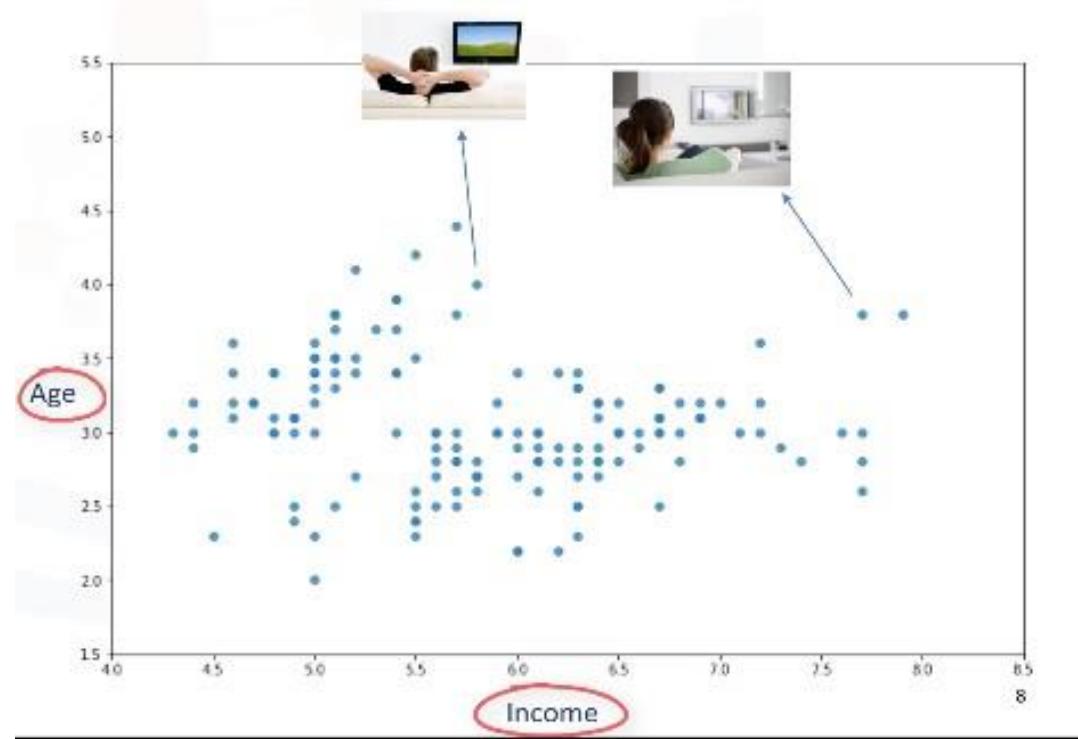
# K-Means Alogarithms

- There are other dissimilarity measures as well that can be used for this purpose, but it is highly dependent on data type and also the domain that clustering is done for it.
- For example, you may use Euclidean distance, cosine similarity, average distance, and so on.
- Indeed, the similarity measure highly controls how the clusters are formed, so it is recommended to understand the domain knowledge of your dataset, and data type of features, and then choose the meaningful distance measurement.
- Now, let's see how k-Means clustering works. For the sake of simplicity, let's assume that our dataset has only two features, the age and income of customers.

Customer ID	Age	Income
1	3	4
2	2	6
3	3.5	2
...	...	..

# K-Means Alogarithms

- This means, it's a 2-dimentional space. We can show the distribution of customers using a scatterplot. The y- axes indicates Age and the x-axes shows Income of customers.



# Pseudocode of KMeans

1. Initialize K=3 centroids randomly
2. Distance Calculation
3. Assign each point to closest centroid
4. Compute the new centroids for each user
5. Repeat until there are no more changes

# Initialize K=3 Centroids Randomly



We try to cluster the customer dataset into distinct groups (or clusters) based on these two dimensions.



In the first step, we should determine the number of clusters.



The key concept of the k-Means algorithm is that it randomly picks a center point for each cluster.



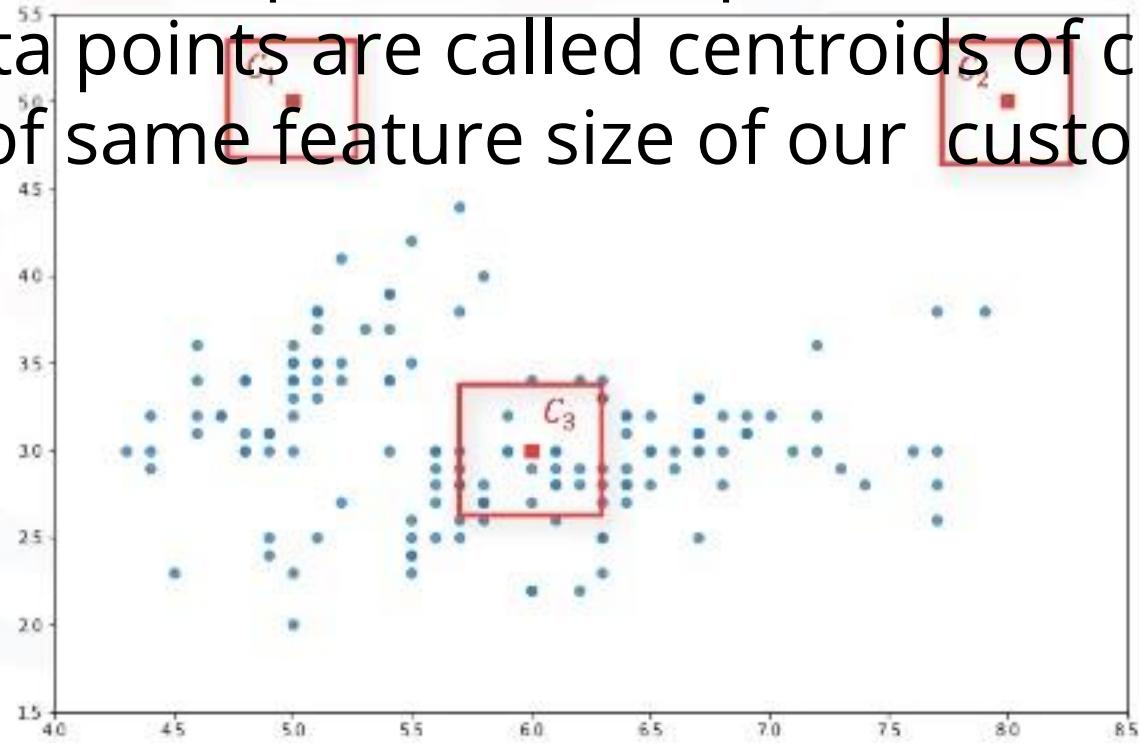
It means, we must initialize  $k$ , which represents "number of clusters."



Essentially, determining the number of clusters in a data set, or  $k$ , is a hard problem in k-Means that we will discuss later.

# Initialize K=3 Centroids Randomly

- For now, let's put k equals 3 here, for our sample dataset. It is like we have 3 representative points for our clusters. These 3 data points are called centroids of clusters, and should be of same feature size of our customer feature set.



# Initialize K=3 Centroids Randomly

There are two approaches to choose these centroids:

- 1) We can randomly choose 3 observations out of the dataset and use these observations as the initial means. Or,
- 2) We can create 3 random points as centroids of the clusters, which is our choice that is shown in this plot with red color.

$$\begin{aligned}C_1 &= [8., 5.] \\C_2 &= [5., 5.] \\C_3 &= [6., 3.] \end{aligned}$$

After the initialization step, which was defining the centroid of each cluster, we have to assign each customer to the closest center.

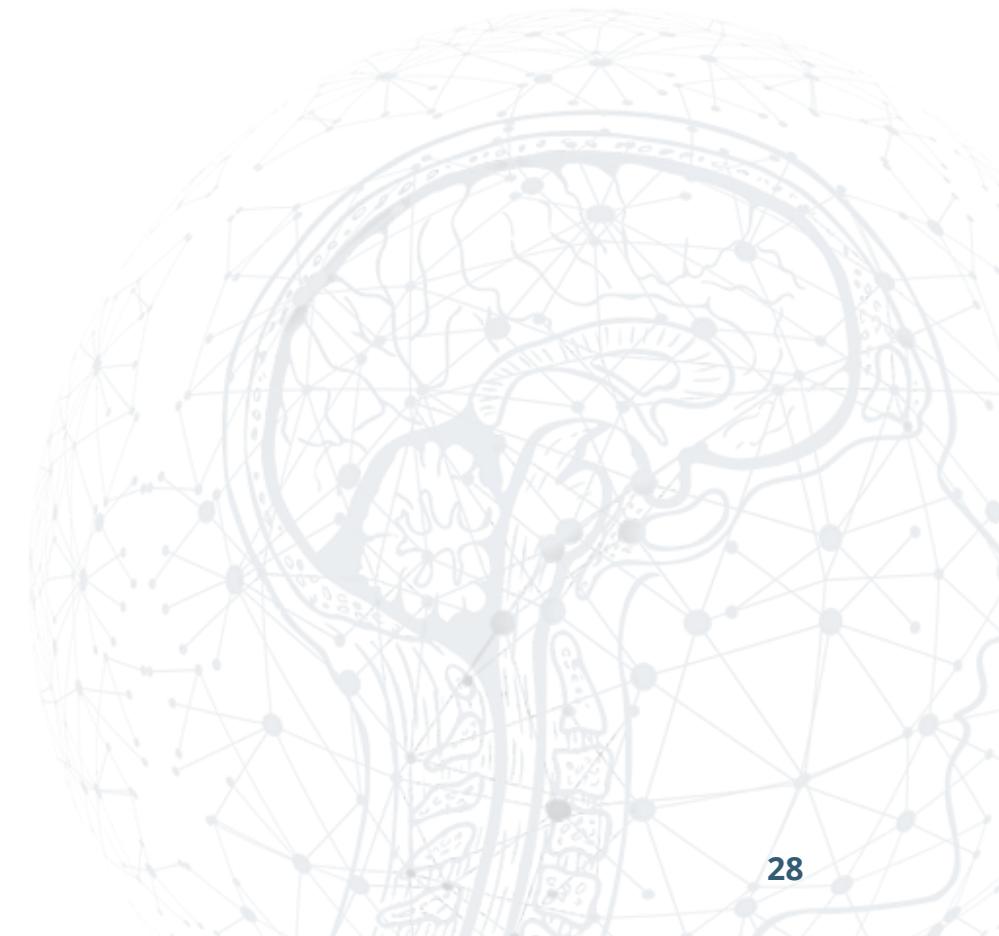
For this purpose, we have to calculate the distance of each data point (or in our case, each customer) from the centroid points.

As mentioned before, depending on the nature of the data and the purpose for which clustering is being used, different measures of distance may be used to place items into clusters.

## Distance Calculation

Therefore, you will form a matrix where each row represents the distance of a customer from each centroid. It is called the "distance-matrix."

$C_1$	$C_2$	$C_3$
$d(p_1, c_1)$	$d(p_1, c_2)$	$d(p_1, c_3)$
$d(p_2, c_1)$	$d(p_2, c_2)$	$d(p_2, c_3)$
$d(p_3, c_1)$	$d(p_3, c_2)$	$d(p_3, c_3)$
$d(p_4, c_1)$	$d(p_4, c_2)$	$d(p_4, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p_n, c_1)$	$d(p_n, c_2)$	$d(p_n, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p_n, c_1)$	$d(p_n, c_2)$	$d(p_n, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p_n, c_1)$	$d(p_n, c_2)$	$d(p_n, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p_n, c_1)$	$d(p_n, c_2)$	$d(p_n, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p \dots, c_1)$	$d(p \dots, c_2)$	$d(p \dots, c_3)$
$d(p_n, c_1)$	$d(p_n, c_2)$	$d(p_n, c_3)$



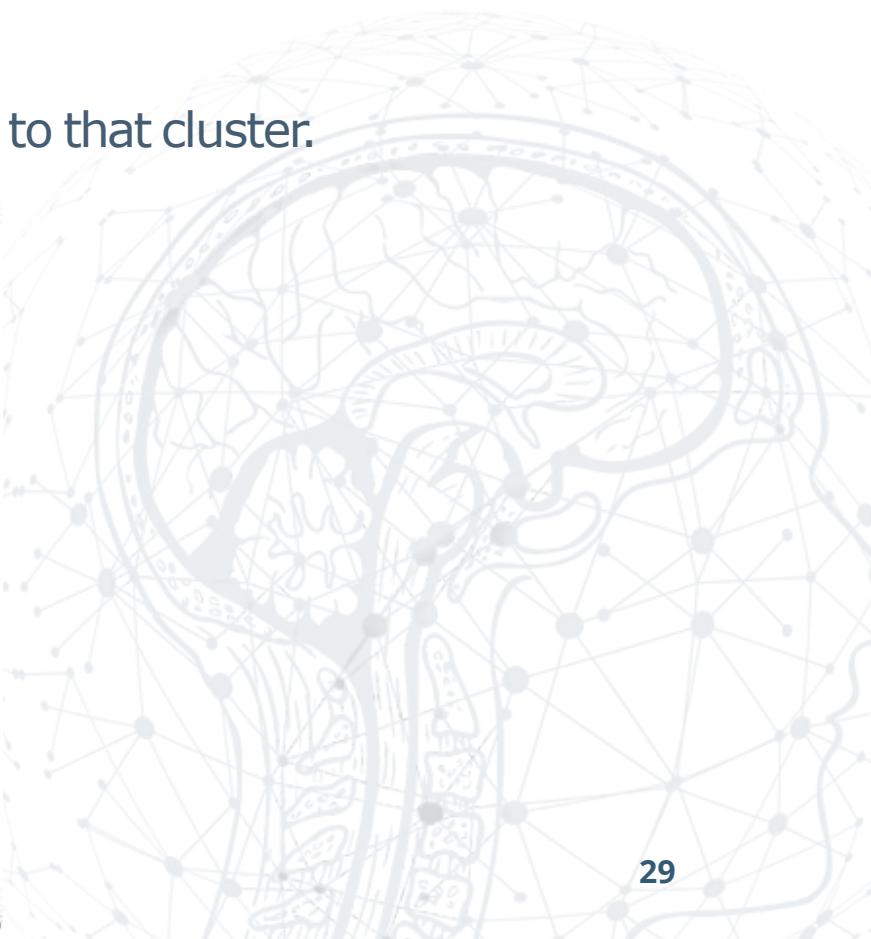
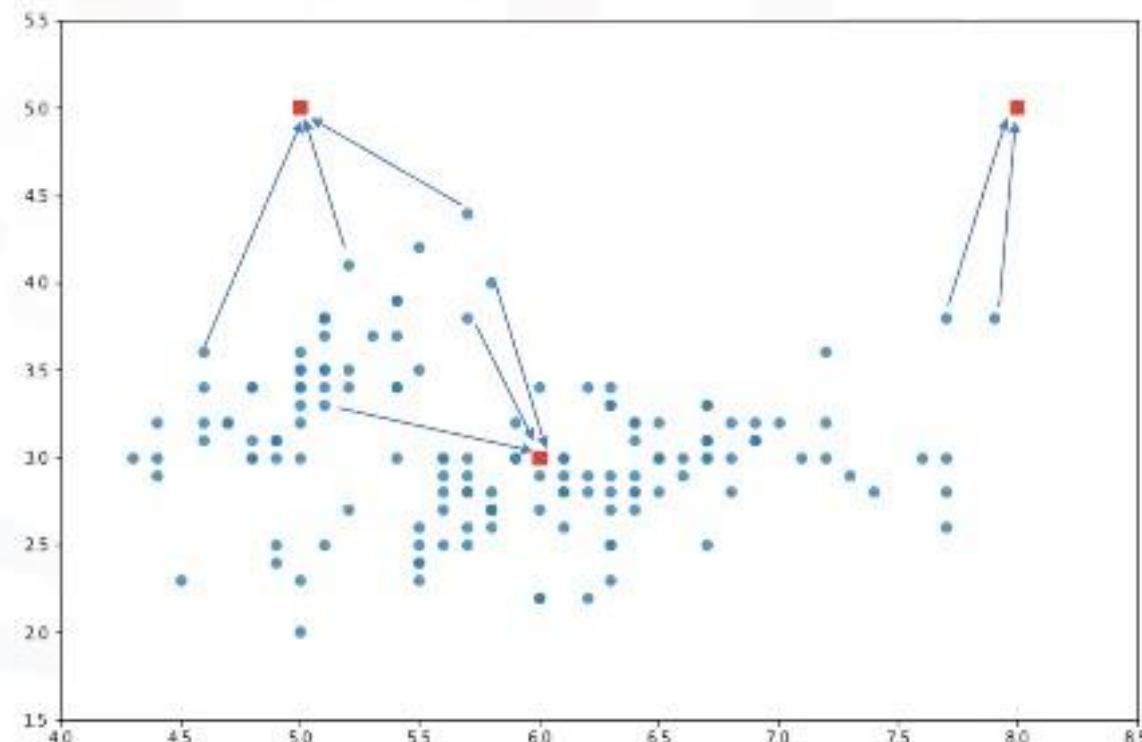
# Assign each point to the closest centroid

The main objective of k-Means clustering is to minimize the distance of data points from the centroid of its cluster and maximize the distance from other cluster centroids.

So, in this step we have to find the closest centroid to each data point.

We can use the distance-matrix to find the nearest centroid to data points.

Finding the closest centroids for each data point, we assign each data point to that cluster.



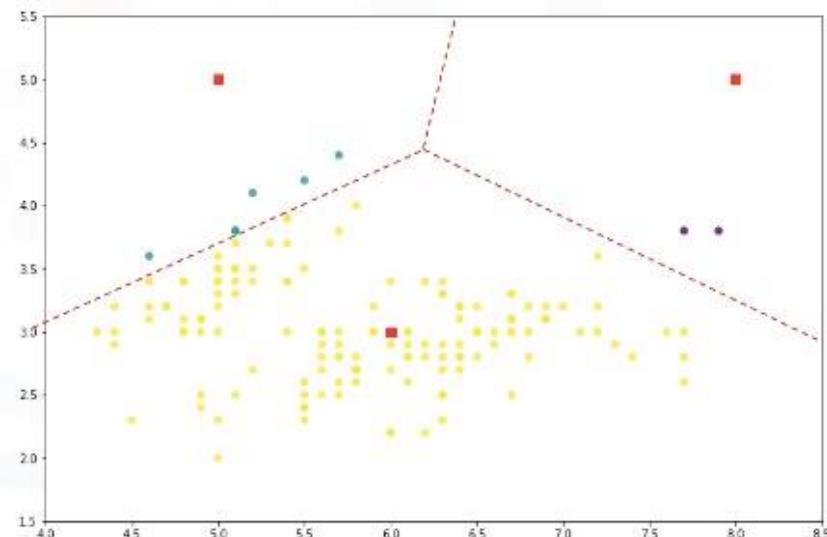
# Compute the new centroid for each cluster

In other words, all the customers will fall to a cluster, based on their distance from centroids.

We can easily say that it does not result in good clusters, because the centroids were chosen randomly from the first.

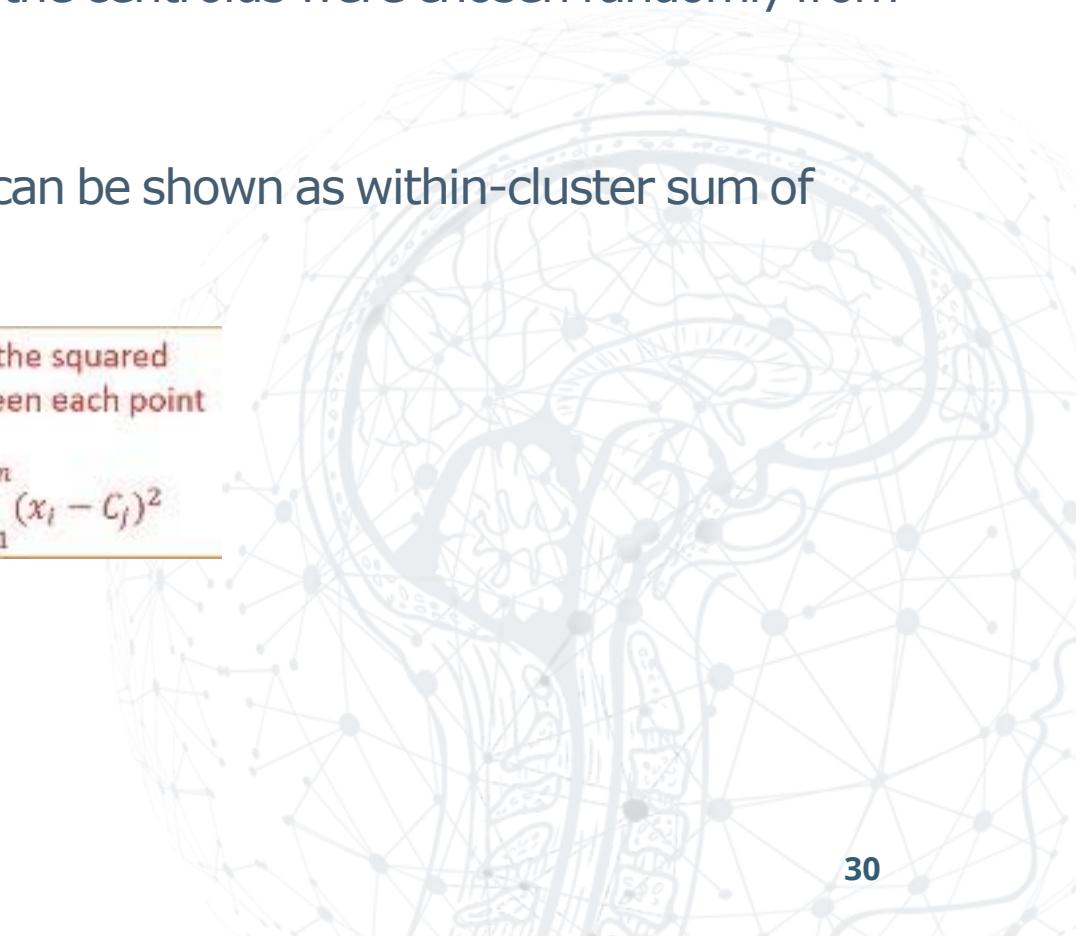
Indeed, the model would have a high error.

Here, error is the total distance of each point from its centroid. It can be shown as within-cluster sum of squares error. Intuitively, we try to reduce this error.



SSE = the sum of the squared differences between each point and its centroid.

$$SSE = \sum_1^n (x_i - c_j)^2$$



# Compute the new centroid for each cluster

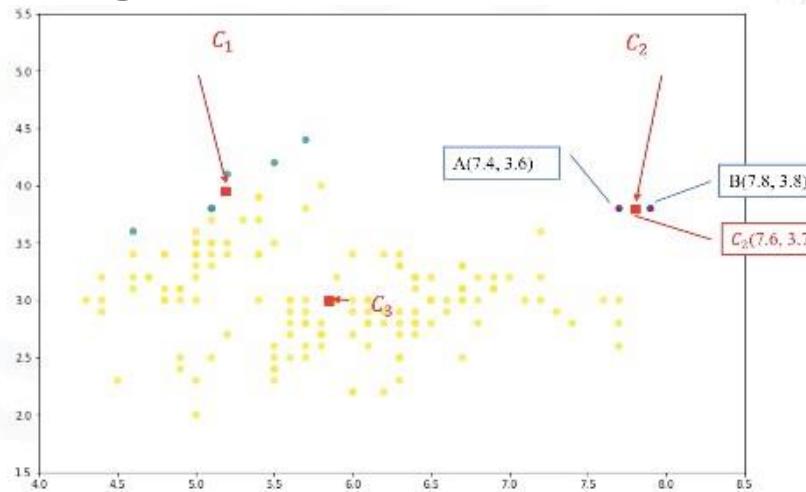
It means we should shape clusters in such a way that the total distance of all members of a cluster from its centroid be minimized.

Now, the question is, "How we can turn it into better clusters, with less error?"

Okay, we move centroids. In the next step, each cluster center will be updated to be the mean for data points in its cluster.

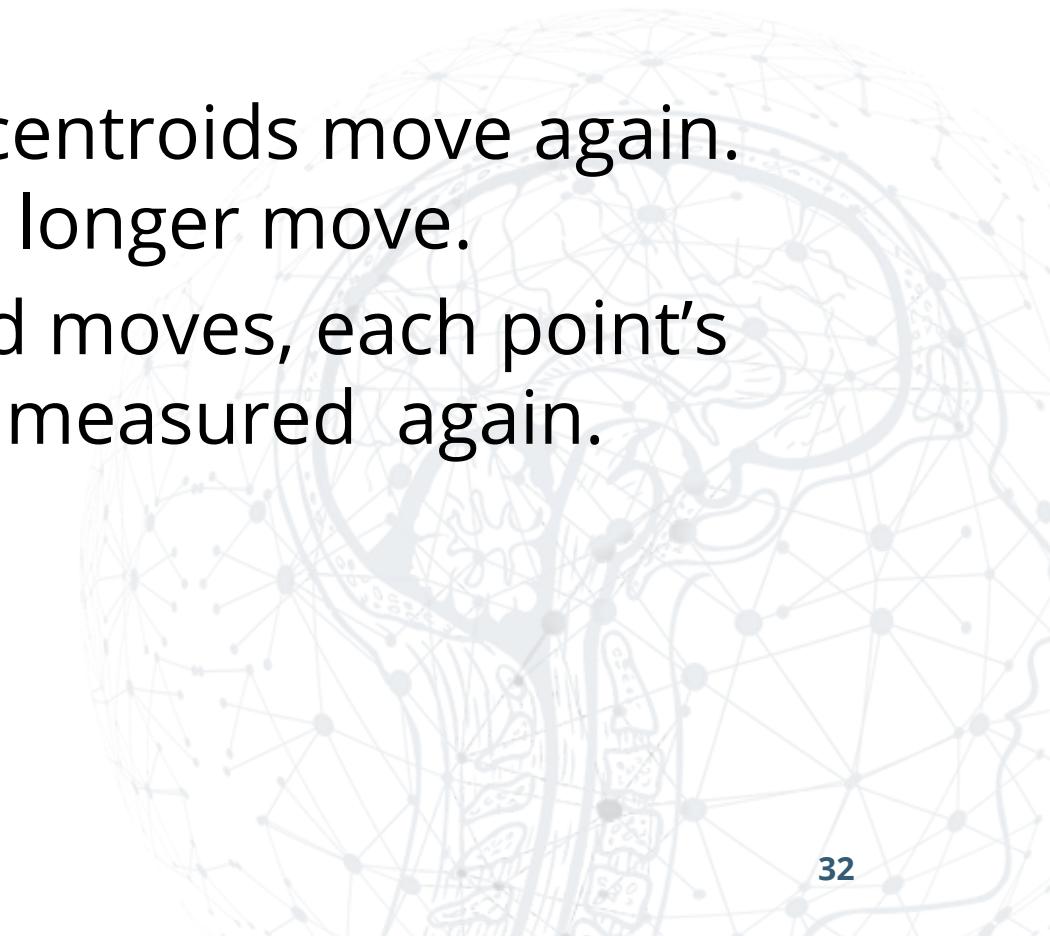
Indeed, each centroid moves according to their cluster members. In other words, the centroid of each of the 3 clusters becomes the new mean.

For example, if Point A coordination is 7.4 and 3.6, and Point B features are 7.8 and 3.8, the new centroid of this cluster with 2 points, would be the average of them, which is 7.6 and 3.7.



# Compute the new centroid for each cluster

- Now we have new centroids. As you can guess, once again, we will have to calculate the distance of all points from the new centroids.
- The points are re-clustered and the centroids move again. This continues until the centroids no longer move.
- Please note that whenever a centroid moves, each point's distance to the centroid needs to be measured again.



# Repeat until there are no more changes



Yes, k-Means is an iterative algorithm, and we have to repeat steps 2 to 4 until the algorithm converges.



In each iteration, it will move the centroids, calculate the distances from new centroids, and assign the data points to the nearest centroid.



It results in the clusters with minimum error, or the most dense clusters.



However, as it is a heuristic algorithm, there is no guarantee that it will converge to the global optimum, and the result may depend on the initial clusters.

It means this algorithm is guaranteed to converge to a result, but the result may be a local optimum (i.e. not necessarily the best possible outcome).

To solve this problem, it is common to run the whole process, multiple times, with different starting conditions.

This means, with randomized starting centroids, it may give a better outcome.

And as the algorithm is usually very fast, it wouldn't be any problem to run it multiple times.

# More on K-Means

---

Please note, however, that you can also use different types of distance measurements, not just Euclidean distance.

Euclidean distance is used because it's the most popular.

Then, assign each data point (or object) to its closest centroid, creating a group.

Next, once each data point has been classified to a group, recalculate the position of the k centroids. The new centroid position is determined by the mean of all points in the group.

Finally, this continues until the centroids no longer move.

---

Let's define the algorithm more concretely before we talk about its accuracy.

---

A k-Means algorithm works by randomly placing k centroids, one for each cluster.

---

The farther apart the clusters are placed, the better.

---

The next step is to calculate the distance of each data point (or object) from the centroids.

---

Euclidean distance is used to measure the distance from the object to the centroid.

# Pseudocode

## k-Means clustering algorithm

1. Randomly placing  $k$  centroids, one for each cluster.
2. Calculate the distance of each point from each centroid.
3. Assign each data point (object) to its closest centroid, creating a cluster.
4. Recalculate the position of the  $k$  centroids.
5. Repeat the steps 2-4, until the centroids no longer move.

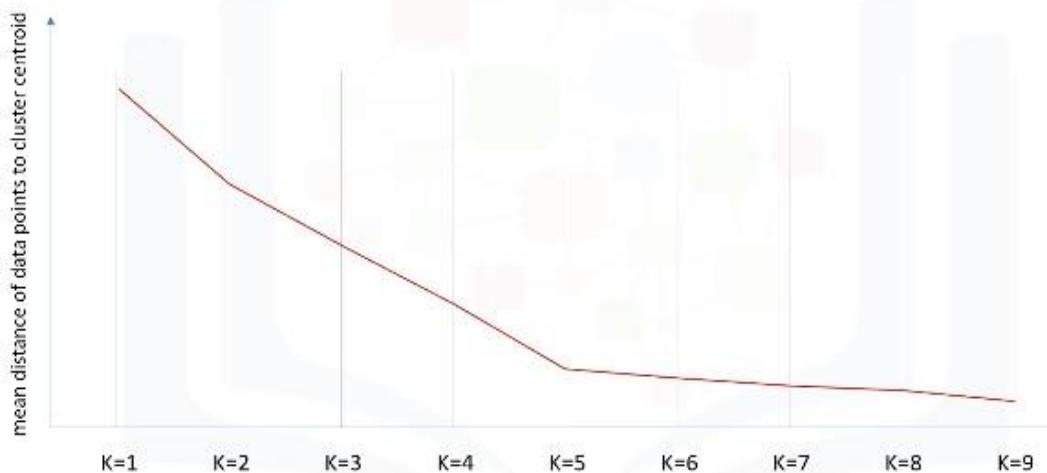
# Accuracy of K-Means

- "How can we evaluate the 'goodness' of the clusters formed by k-Means?"
- "How do we calculate the accuracy of k-Means clustering?"
  1. Compare the clusters with the ground truth, if it's available.
  2. However, because k-Means is a unsupervised algorithm, we usually don't have ground truth in real world problems to be used. But, there is still a way to say how bad each cluster is using:
    1. **average distance between data points within a cluster**
    2. **average of the distances of data points from their cluster centroids**

# K-Means Accuracy

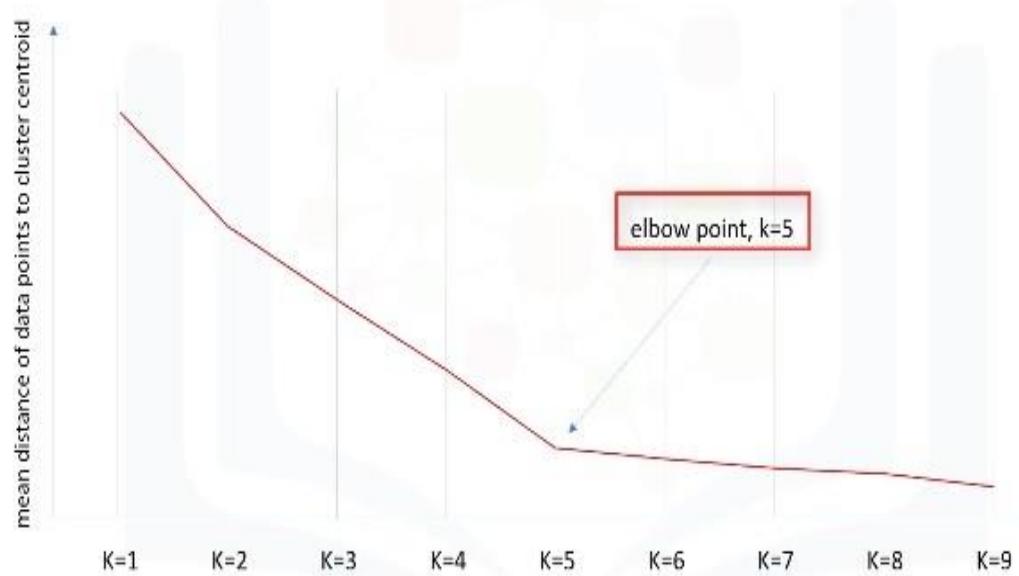
- Run the clustering across the different values of K, and looking at a metric of accuracy for clustering.
- Metric → **mean distance between data points and their cluster centroid**, which indicate how dense our clusters are, or to what extend we minimized the error of clustering.
- Find the best value for k.

Choosing k



# Elbow Method

- Problem: with increasing the number of clusters, the distance of centroids to data points will always reduce.
- This means, increasing K will always decrease the “error.”
- So, the value of the metric as a function of K is plotted and the **“elbow point”** is determined, where the rate of decrease sharply shifts.



# K-Means

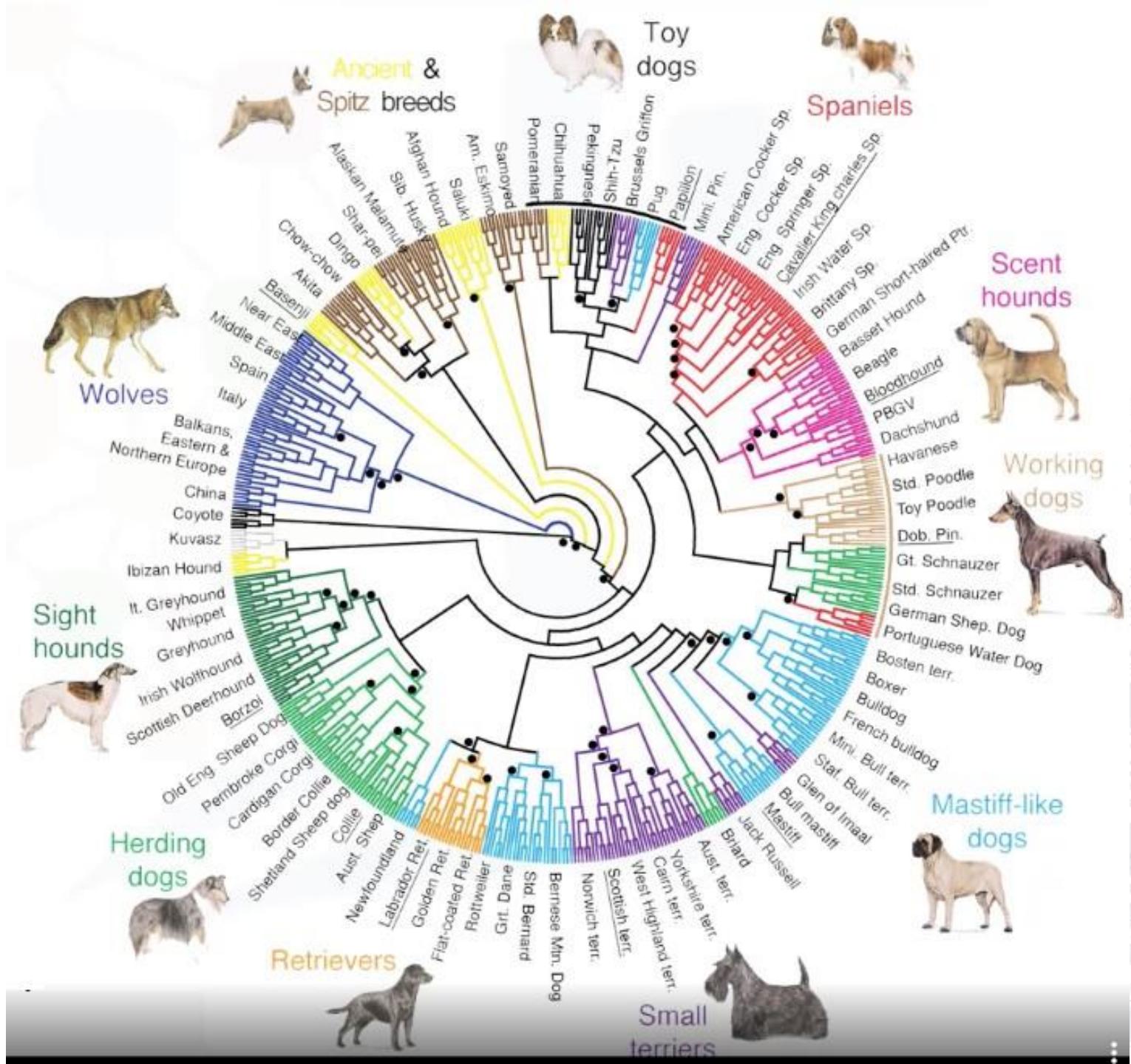
- So, let's recap k-Means clustering: k-Means is a partitioned-based clustering, which is:
  - Relatively efficient on medium and large sized datasets;
  - Produces sphere-like clusters, because the clusters are shaped around the centroids;
  - Its drawback is that we should pre-specify the number of clusters, and this is not an easy task.
- Let's look at this chart. An international team of scientists, led by UCLA biologists, used this dendrogram to report genetic data from more than 900 dogs from 85 breeds -- and more than 200 wild gray wolves worldwide, including populations from North America, Europe, the Middle East, and East Asia.
- They used molecular genetic techniques to analyze more than 48,000 genetic markers.



UNIVERSITAS  
INDONESIA

*Veritas, Probatus, Justitia*

# Hierarchical Clustering



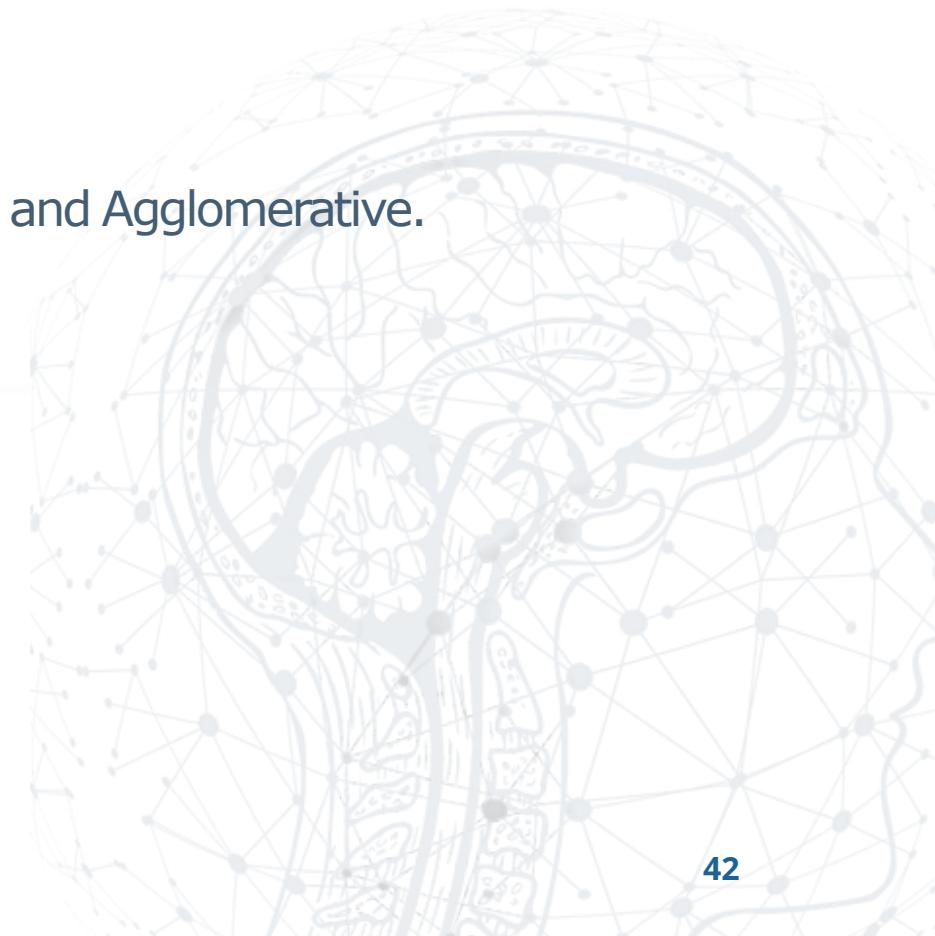
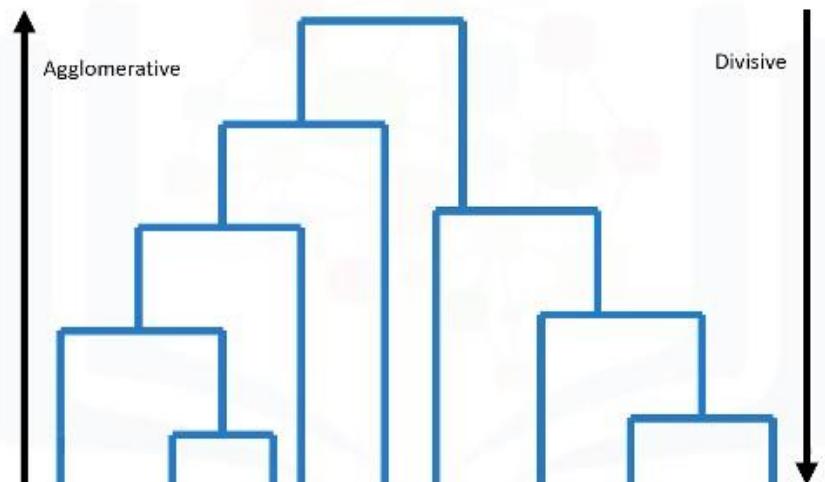
# Hierarchical Clustering

This diagram shows hierarchical clustering of these animals based on the similarity in their genetic data.

Hierarchical clustering algorithms build a hierarchy of clusters where each node is a cluster consists of the clusters of its daughter nodes.

Strategies for hierarchical clustering generally fall into two types: Divisive and Agglomerative.

Hierarchical clustering

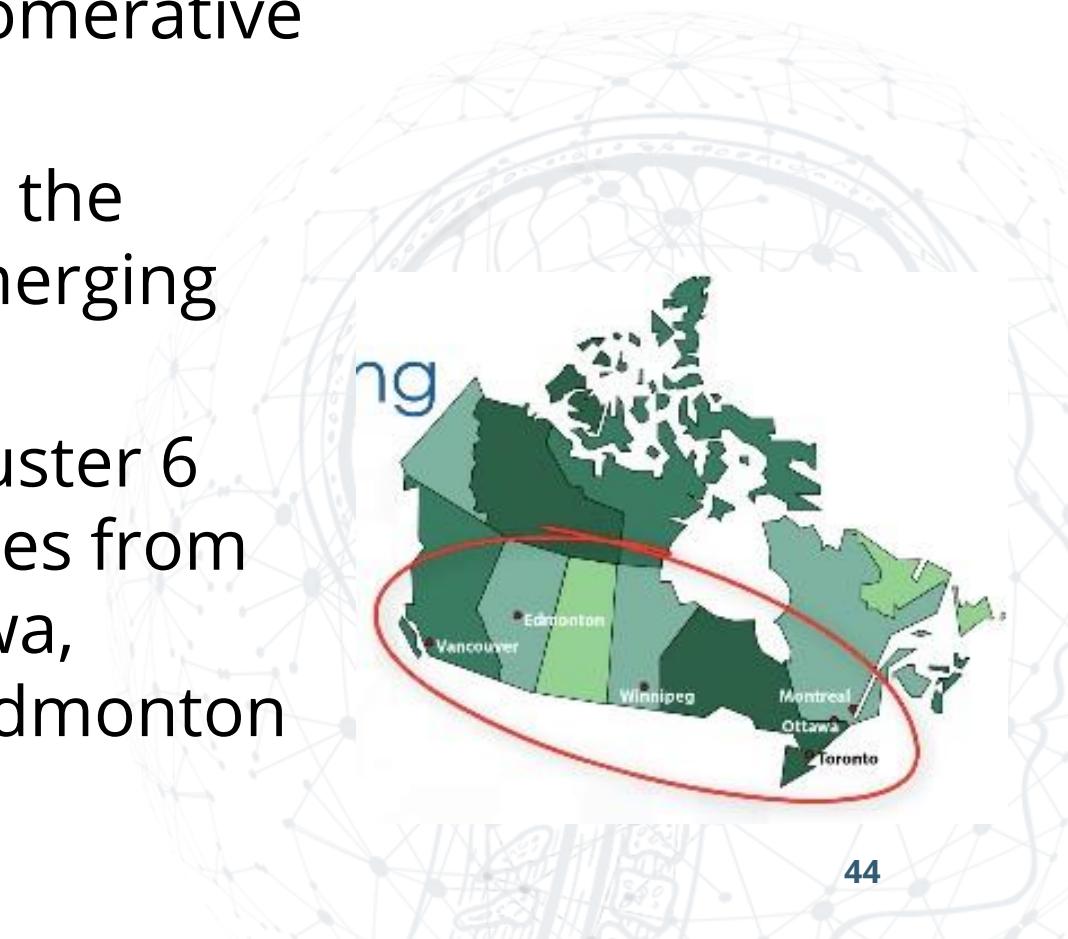


# Type

- **Divisive** is top-down, so you start with all observations in a large cluster and break it down into smaller pieces. Think about divisive as "dividing" the cluster.
- **Agglomerative** is the opposite of divisive, so it is bottom-up, where each observation starts in its own cluster and pairs of clusters are merged together as they move up the hierarchy.
  - Agglomeration means to amass or collect things, which is exactly what this does with the cluster.

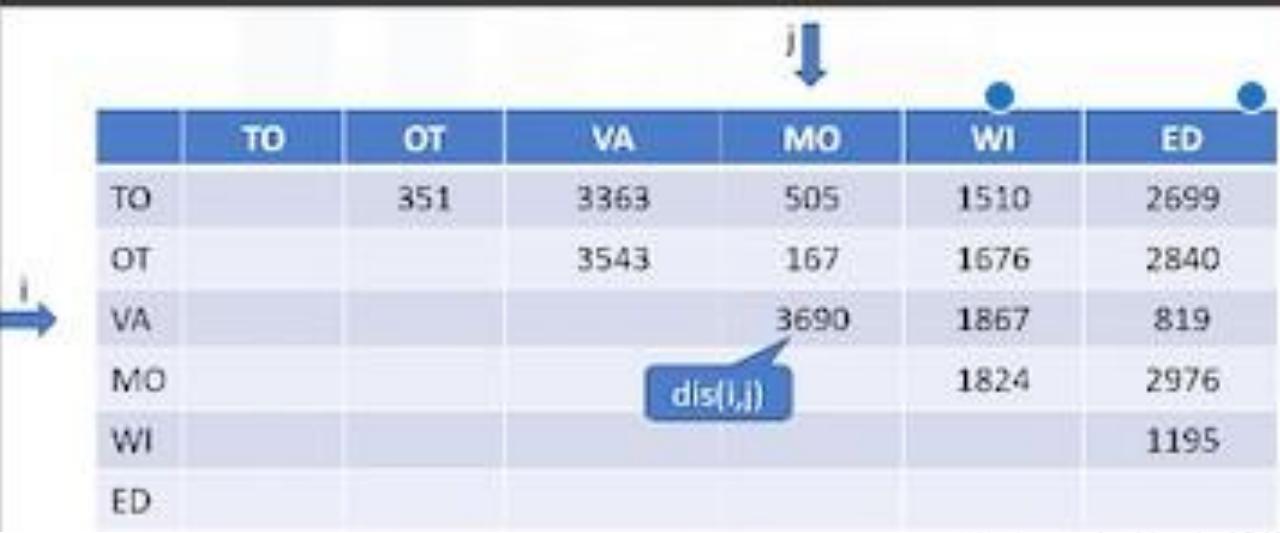
# Hierarchical Clustering

- The Agglomerative approach is more popular among data scientists and so it is the main subject. Let's look at a sample of Agglomerative clustering.
- This method builds the hierarchy from the individual elements by progressively merging clusters.
- In our example, let's say we want to cluster 6 cities in Canada based on their distances from one another. They are: Toronto, Ottawa, Vancouver, Montreal, Winnipeg, and Edmonton



# Hierarchical Clustering

- We construct a distance matrix at this stage, where the numbers in the row i column j is the distance between the i and j cities.
- In fact, this table shows the distances between each pair of cities.



	TO	OT	VA	MO	WI	ED
TO		351	3363	505	1510	2699
OT			3543	167	1576	2840
VA				3690	1867	819
MO					1824	2976
WI						1195
ED						

# Hierarchical Clustering

The algorithm is started by assigning each city to its own cluster. So, if we have 6 cities, we have 6 clusters, each containing just one city.

Let's note each city by showing the first two characters of its name.

TO	OT	MO	VA	ED	WI	
TO	TO	OT	VA	MO	WI	ED
TO		351	3363	505	1510	2699
OT			3543	167	1676	2840
VA				3690	1867	819
MO					1824	2976
WI						1195
ED						

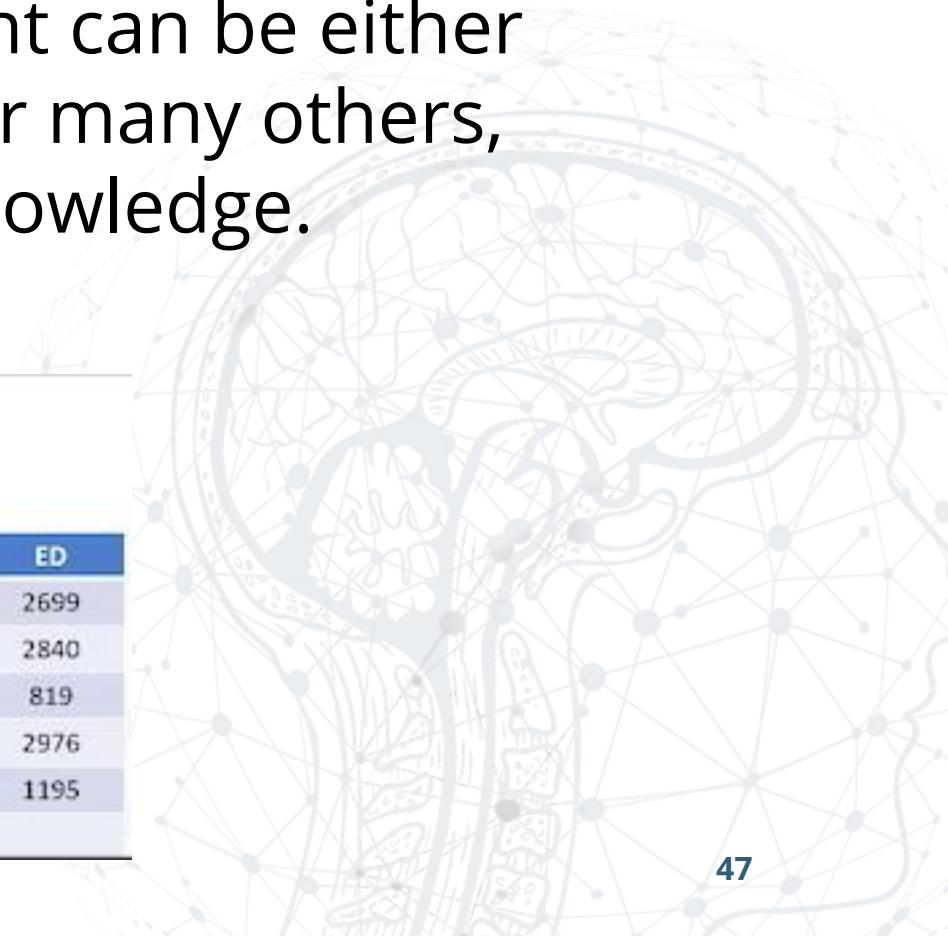
The first step is to determine which cities -- let's call them clusters from now on -- to merge into a cluster.

Usually, we want to take the two closest clusters according to the chosen distance. Looking at the distance matrix, Montreal and Ottawa are the closest clusters.

So, we make a cluster out of them.

# Hierarchical Clustering

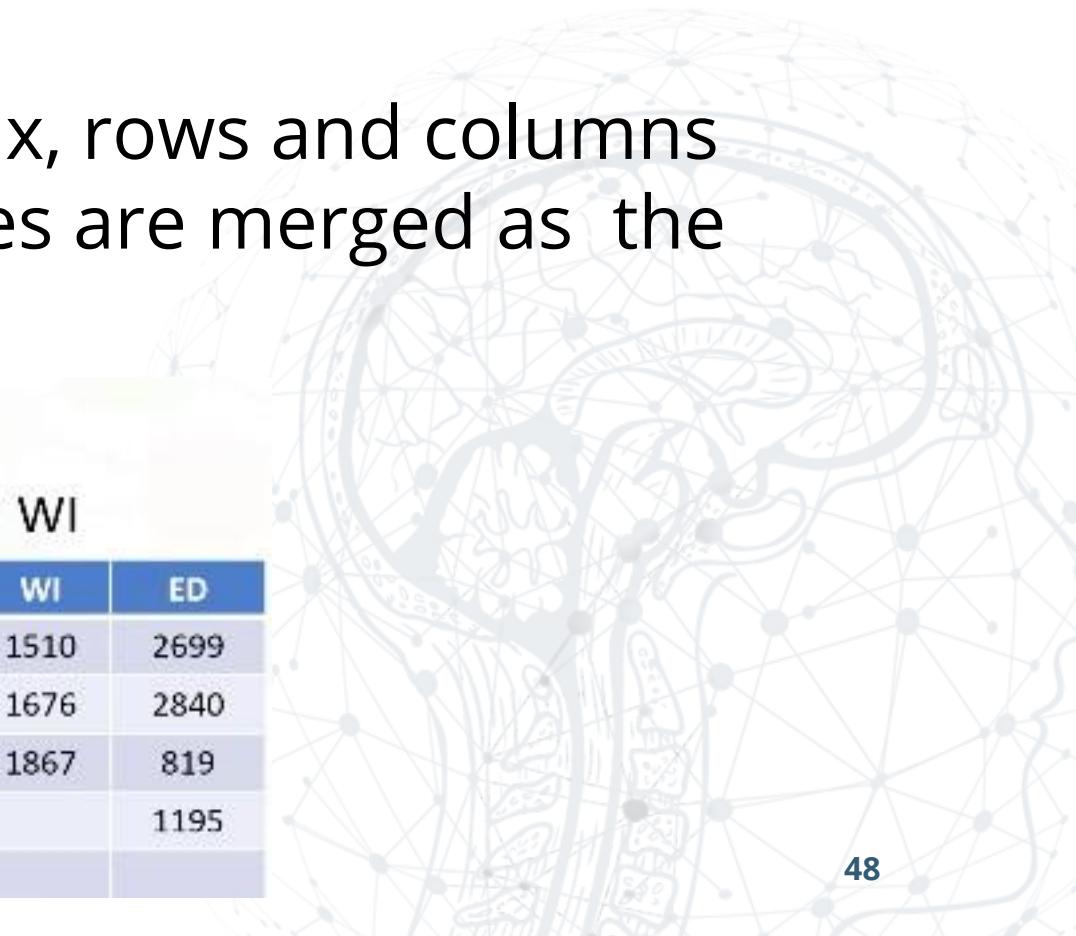
- Please notice that we just use a simple 1-dimentional distance feature here, but our object can be multi-dimensional, and distance measurement can be either Euclidean, Pearson, average distance, or many others, depending on data type and domain knowledge.



	TO	OT	MO	VA	ED	WI
TO		351	3363	505	1510	2699
OT			3543	167	1676	2840
VA				3690	1867	819
MO					1824	2976
WI						1195
ED						

# Hierarchical Clustering

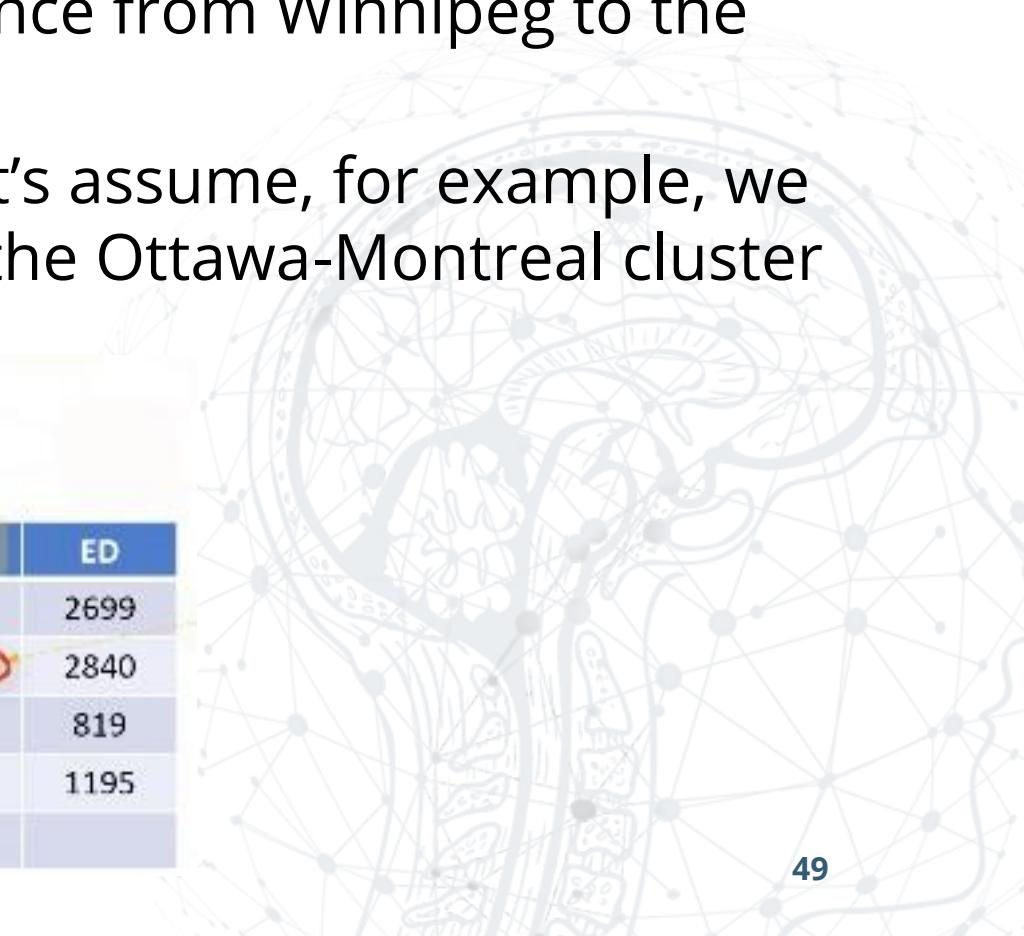
- Anyhow, we have to merge these two closest cities in the distance matrix as well. So, rows and columns are merged as the cluster is constructed.
- As you can see in the distance matrix, rows and columns related to Montreal and Ottawa cities are merged as the cluster is constructed.



	TO	OT/MO	VA	WI	ED
TO		351	3363	1510	2699
OT/MO			3543	1676	2840
VA				1867	819
WI					1195
ED					

# Hierarchical Clustering

- Then, the distances from all cities to this new merged cluster get updated. But how?
- For example, how do we calculate the distance from Winnipeg to the Ottawa-Montreal cluster?
- Well, there are different approaches, but let's assume, for example, we just select the distance from the centre of the Ottawa-Montreal cluster to Winnipeg.

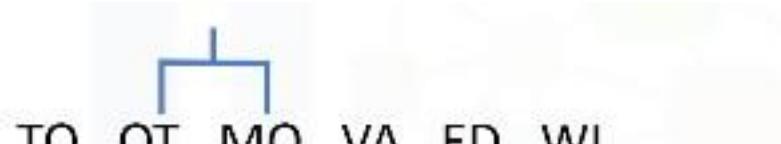


		TO	OT	MO	VA	ED	WI
TO							
OT/MO							
VA						1867	819
WI							1195
ED							

A hierarchical clustering dendrogram is shown above the table, with branches merging TO and OT/MO. The distance between the merged cluster (OT/MO) and VA is highlighted with a red oval around the value 1676.

# Hierarchical Clustering

- Updating the distance matrix, we now have one less cluster.

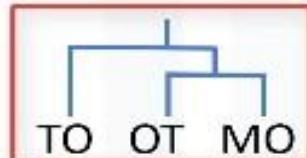


	TO	OT	MO	VA	WI	ED
TO				3363	1510	2699
OT/MO				3543	1676	2840
VA					1867	819
WI						1195
ED						

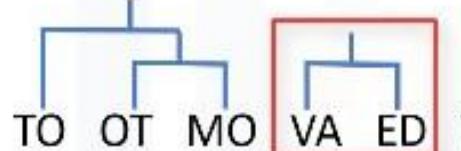


# Hierarchical Clustering

- Next, we look for the closest clusters once again.
- In this case, Ottawa-Montreal and Toronto are the closest ones, which creates another cluster.
- In the next step, the closest distance is between the Vancouver cluster and the Edmonton cluster. Forming a new cluster, their data in the matrix table gets updated.



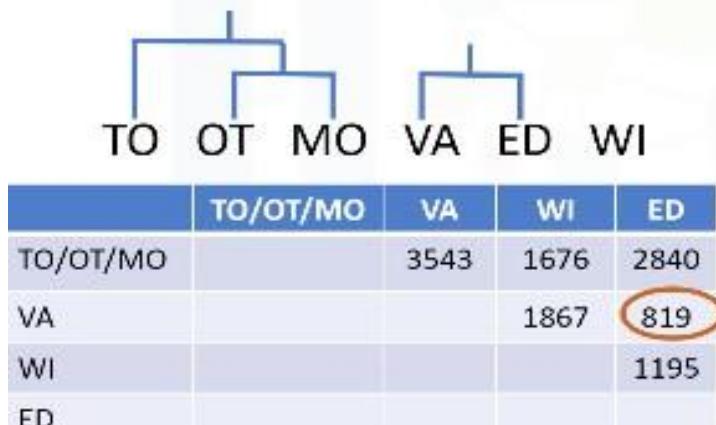
	TO	OT/MO	VA	WI	ED
TO		351	3363	1510	2699
OT/MO			3543	1676	2840
VA				1867	819
WI					1195
ED					



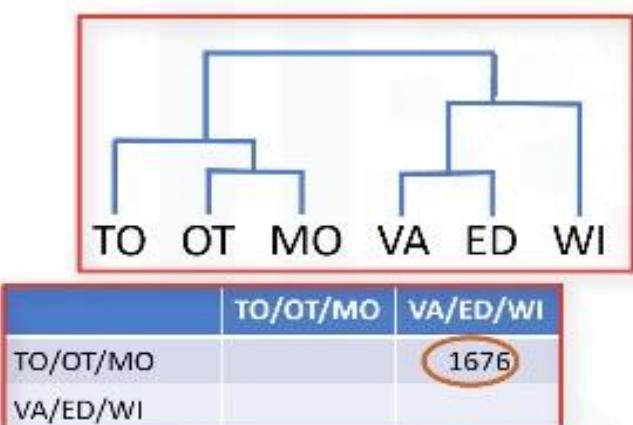
	TO/OT/MO	VA	WI	ED
TO/OT/MO		3543	1676	2840
VA			1867	819
WI				1195
ED				

# Hierarchical Clustering

- Essentially, the rows and columns are merged as the clusters are merged and the distance updated.
- This is a common way to implement this type of clustering, and has the benefit of caching distances between clusters.

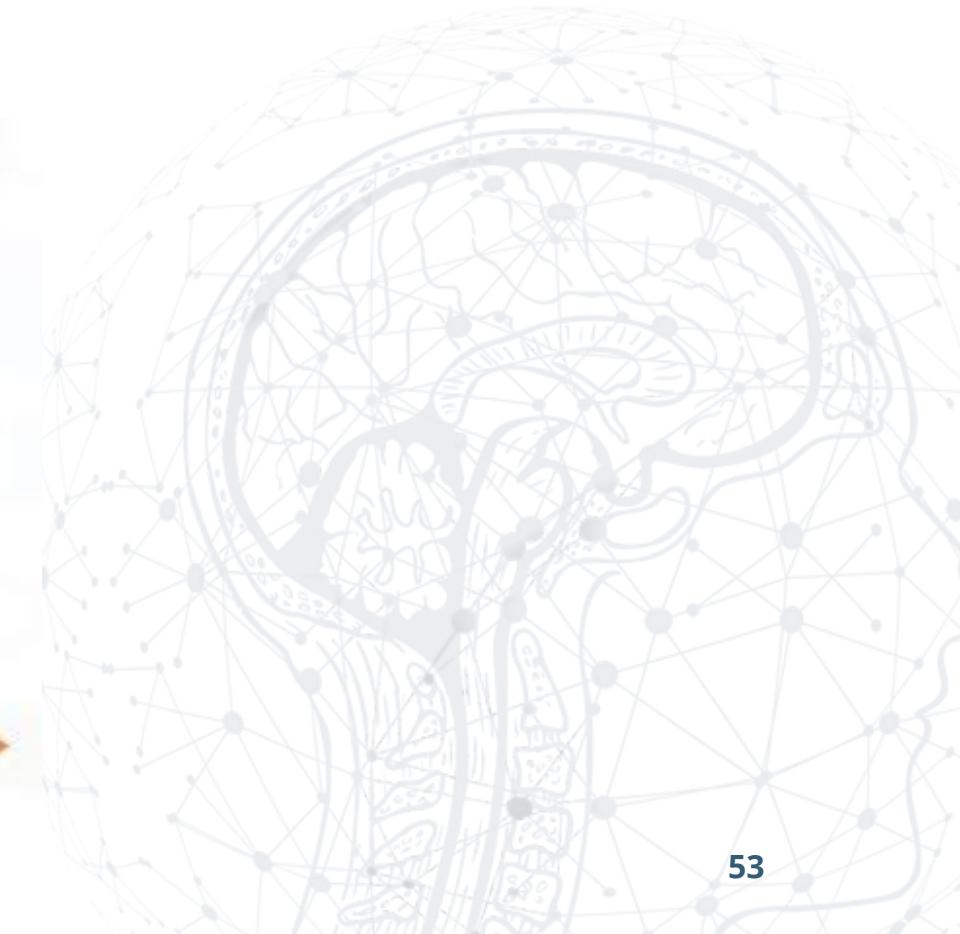
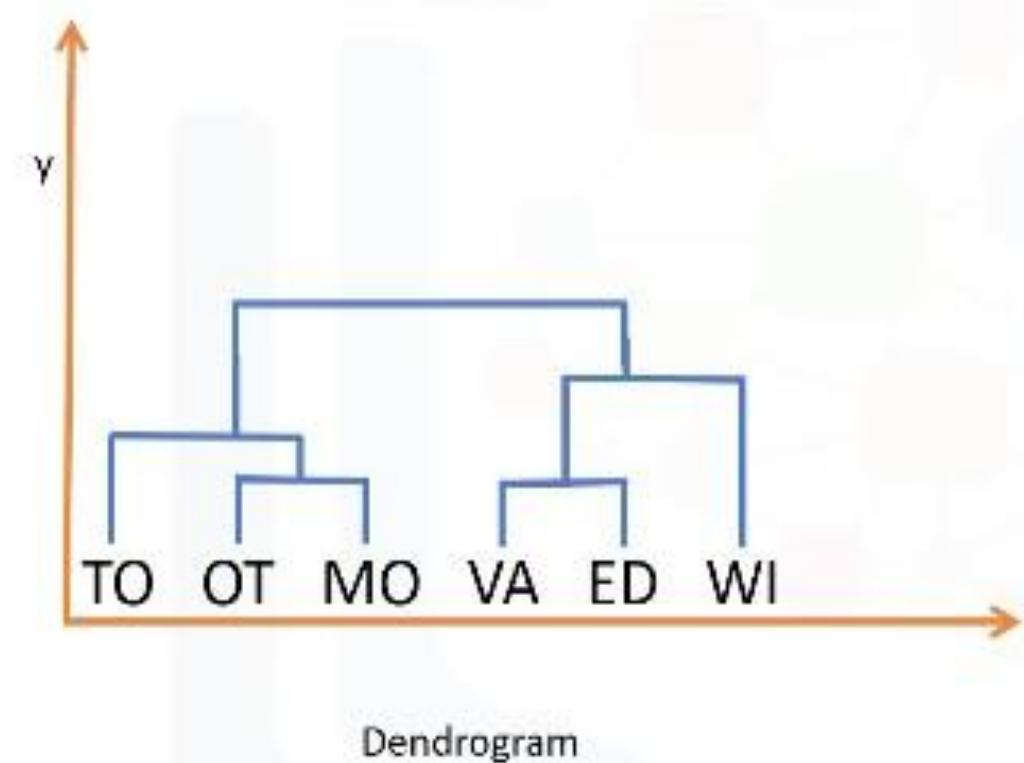


- In the same way, agglomerative algorithm proceeds by merging clusters.
- And we repeat it until all clusters are merged and the tree becomes completed. It means, until all cities are clustered into a single cluster of size 6.



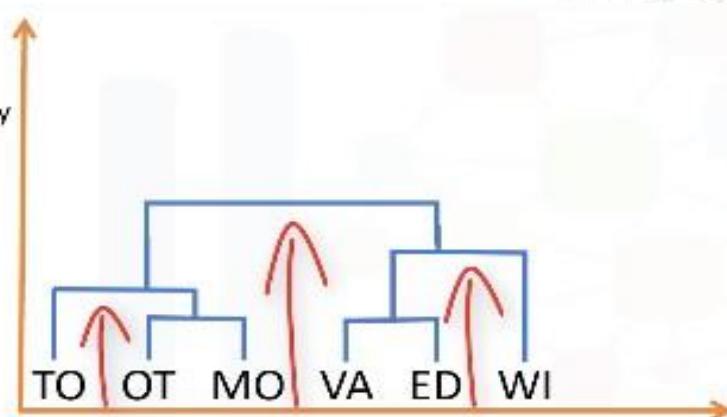
# Hierarchical Clustering

- Hierarchical clustering is typically visualized as a dendrogram as shown on this slide.



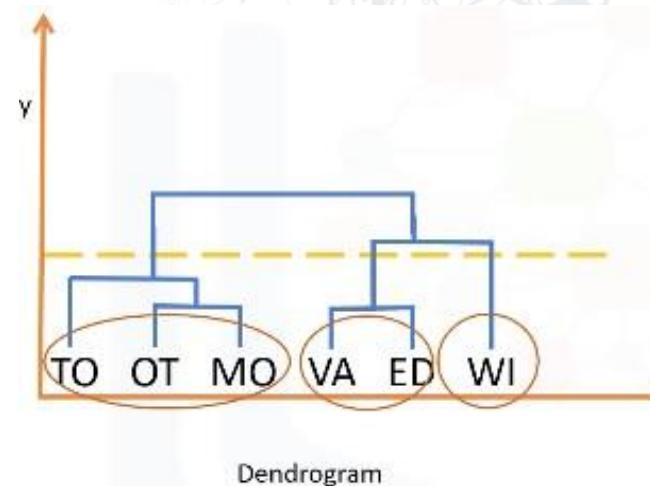
# Hierarchical Clustering

- Each merge is represented by a horizontal line.
- The y-coordinate of the horizontal line is the similarity of the two clusters that were merged, where cities are viewed as singleton clusters.
- By moving up from the bottom layer to the top node, a dendrogram allows us to reconstruct the history of merges that resulted in the depicted clustering.



# Hierarchical Clustering

- Essentially, Hierarchical clustering does not require a pre-specified number of clusters. However, in some applications we want a partition of disjoint clusters just as in flat clustering. In those cases, the hierarchy needs to be cut at some point.
- For example here, cutting in a specific level of similarity, we create 3 clusters of similar cities.

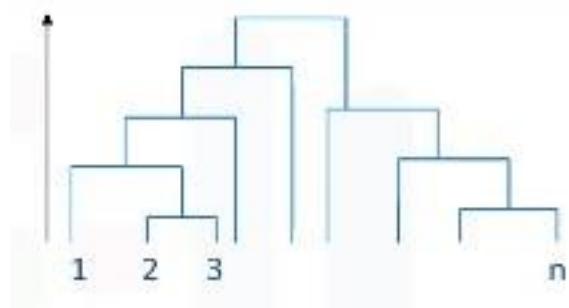


# More on Hierarchical Clustering

Let's look at Agglomerative algorithm for Hierarchical Clustering.

Remember that Agglomerative clustering is a bottom-up approach.

Let's say our dataset has  $n$  data points.



First, we want to create  $n$  clusters, one for each data point. Then each point is assigned as a cluster.

Next, we want to compute the distance/proximity matrix, which will be an  $n$  by  $n$  table.

$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

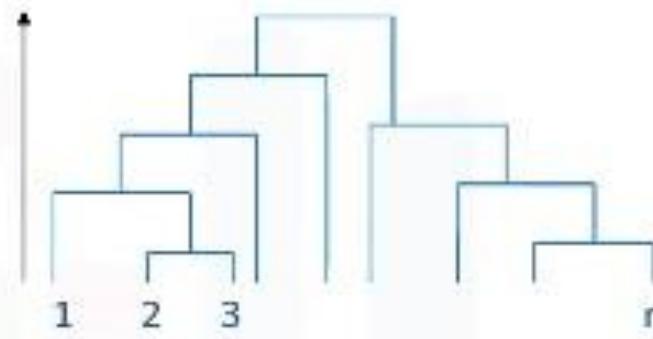
# More on Hierarchical Clustering

- After that, we want to iteratively run the following steps until the specified cluster number is reached, or until there is only one cluster left.
- First, MERGE the two nearest clusters. (Distances are computed already in the proximity matrix.) Second, UPDATE the proximity matrix with the new values.
- We stop after we've reached the specified number of clusters, or there is only one cluster remaining, with the result stored in a dendrogram.
- So, in the proximity matrix, we have to measure the distances between clusters, and also merge the clusters that are “nearest.”

# Pseudocode

## Agglomerative algorithm

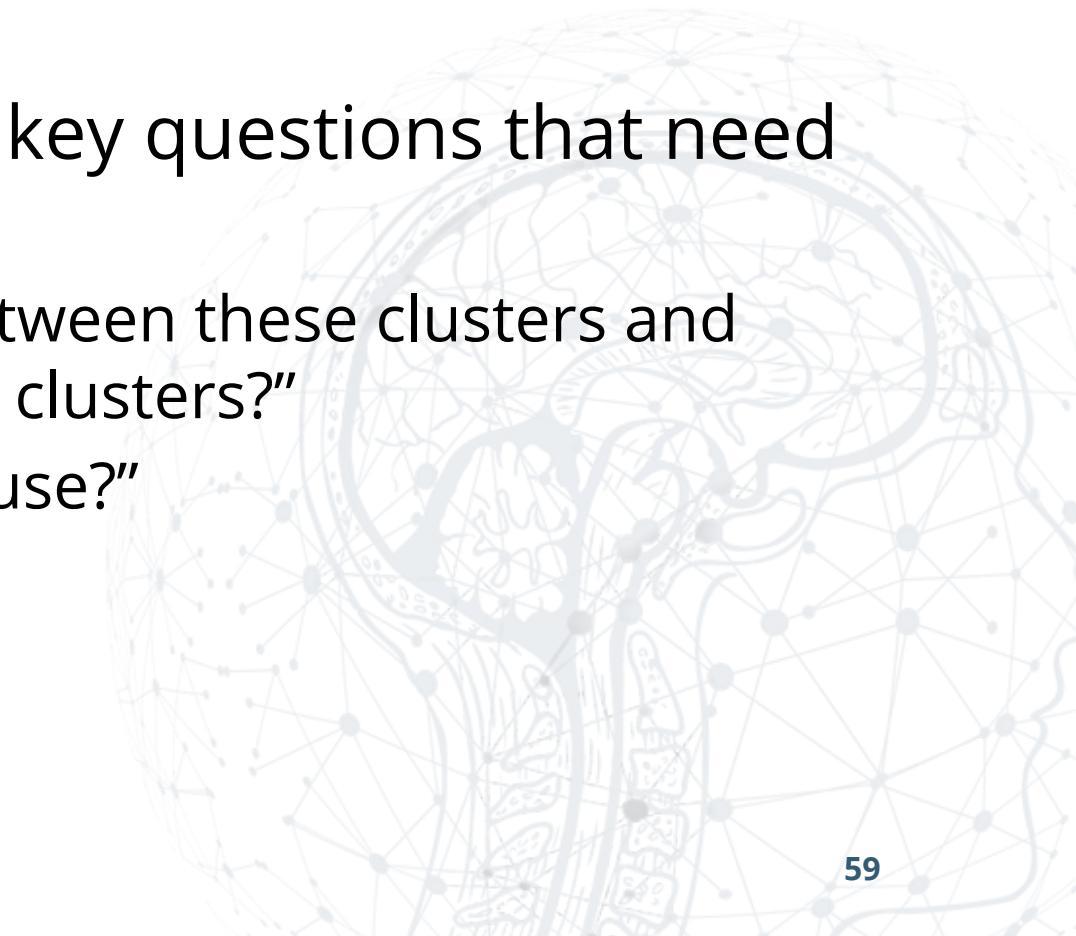
1. Create  $n$  clusters, one for each data point
2. Compute the Proximity Matrix
3. Repeat
  - i. Merge the two closest clusters
  - ii. Update the proximity matrix
4. Until only a single cluster remains



$$\begin{bmatrix} 0 & & & & \\ d(2,1) & 0 & & & \\ d(3,1) & d(3,2) & 0 & & \\ \vdots & \vdots & \vdots & \ddots & \\ d(n,1) & d(n,2) & \dots & \dots & 0 \end{bmatrix}$$

# More on Hierarchical Clustering

- So, the key operation is the computation of the proximity between the clusters with one point, and also clusters with multiple data points.
- At this point, there are a number of key questions that need to be answered. For instance,
  - “How do we measure the distances between these clusters and How do we define the ‘nearest’ among clusters?”
  - We also can ask, “Which points do we use?”



# More on Hierarchical Clustering

First, let's see how to calculate the distance between 2 clusters with 1 point each.

Let's assume that we have a dataset of patients, and we want to cluster them using hierarchy clustering.

So, our data points are patients, with a feature set of 3 dimensions.



Patient 1		
Age	BMI	BP
54	190	120



Patient 2		
Age	BMI	BP
50	200	125



# More on Hierarchical Clustering

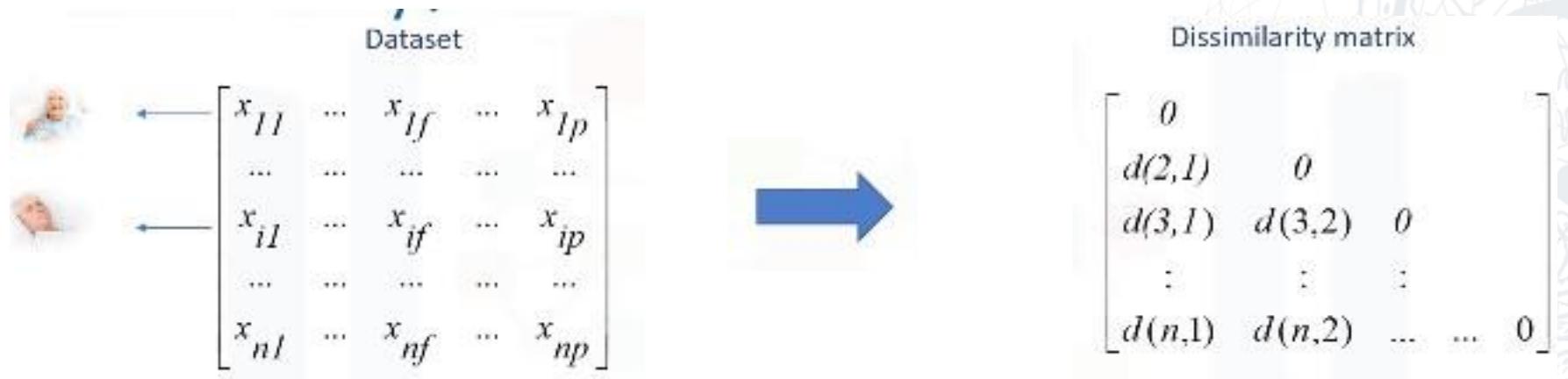
For example, Age, Body Mass Index (or BMI), and Blood Pressure.

We can use different distance measurements to calculate the proximity matrix. For instance, Euclidean distance.

$$\begin{aligned}
 \text{Dis } (p_1, p_2) &= \sqrt{\sum_{i=0}^n (x_i - y_i)^2} \\
 &= \sqrt{(54 - 50)^2 + (190 - 200)^2 + (120 - 125)^2} \\
 &= 11.87
 \end{aligned}$$

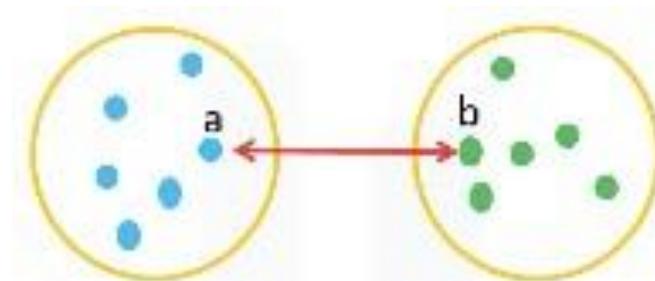
So, if we have a dataset of  $n$  patients, we can build an  $n$  by  $n$  dissimilarity-distance matrix.

It will give us the distance of clusters with 1 data point. However, as mentioned, we merge clusters in



# More on Hierarchical Clustering

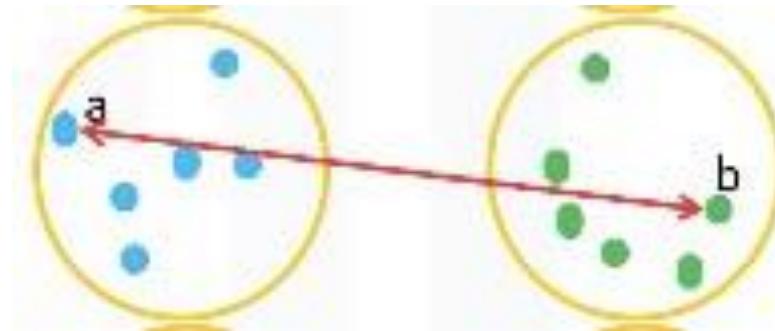
- Agglomerative clustering. Now, the question is, “How can we calculate the distance between clusters when there are multiple patients in each cluster?”
- We can use different criteria to find the closest clusters, and merge them.
- In general, it completely depends on the data type, dimensionality of data, and most importantly, the domain knowledge of the dataset.
- In fact, different approaches to defining the distance between clusters, distinguish the different algorithms.
- As you might imagine, there are multiple ways we can do this. The first one is called Single-Linkage Clustering.
  - Single linkage is defined as the shortest distance between 2 points in each cluster, such as point “a” and “b”.



# More on Hierarchical Clustering

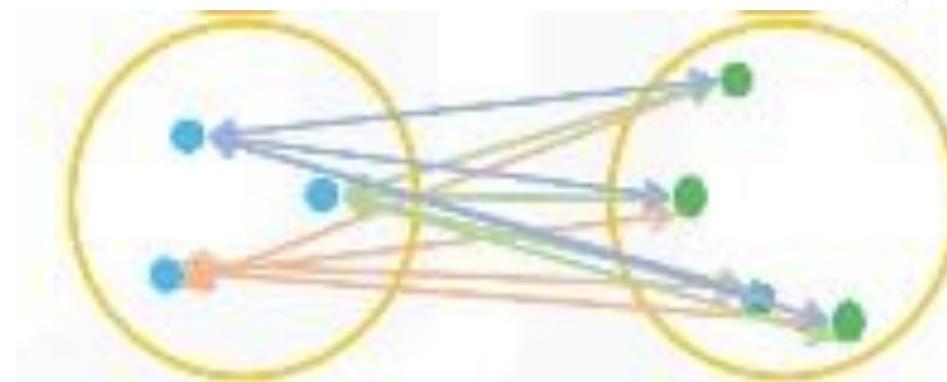
Next up is Complete-Linkage Clustering.

This time, we are finding the longest distance between points in each cluster, such as the distance between point "a" and "b".



The third type of linkage is Average Linkage Clustering, or the mean distance.

This means we're looking at the average distance of each point from one cluster to every point in another cluster.

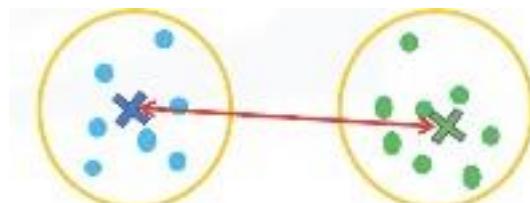


# More on Hierarchical Clustering

The final linkage type to be reviewed is Centroid Linkage Clustering.

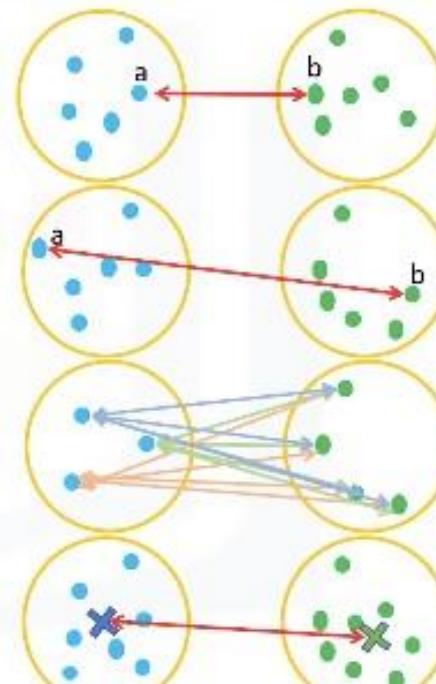
**Centroid is the average of the feature sets of points in a cluster.**

**This linkage takes into account the centroid of each cluster when determining the minimum distance.**



## Distance between clusters

- Single-Linkage Clustering
  - Minimum distance between clusters
- Complete-Linkage Clustering
  - Maximum distance between clusters
- Average Linkage Clustering
  - Average distance between clusters
- Centroid Linkage Clustering
  - Distance between cluster centroids

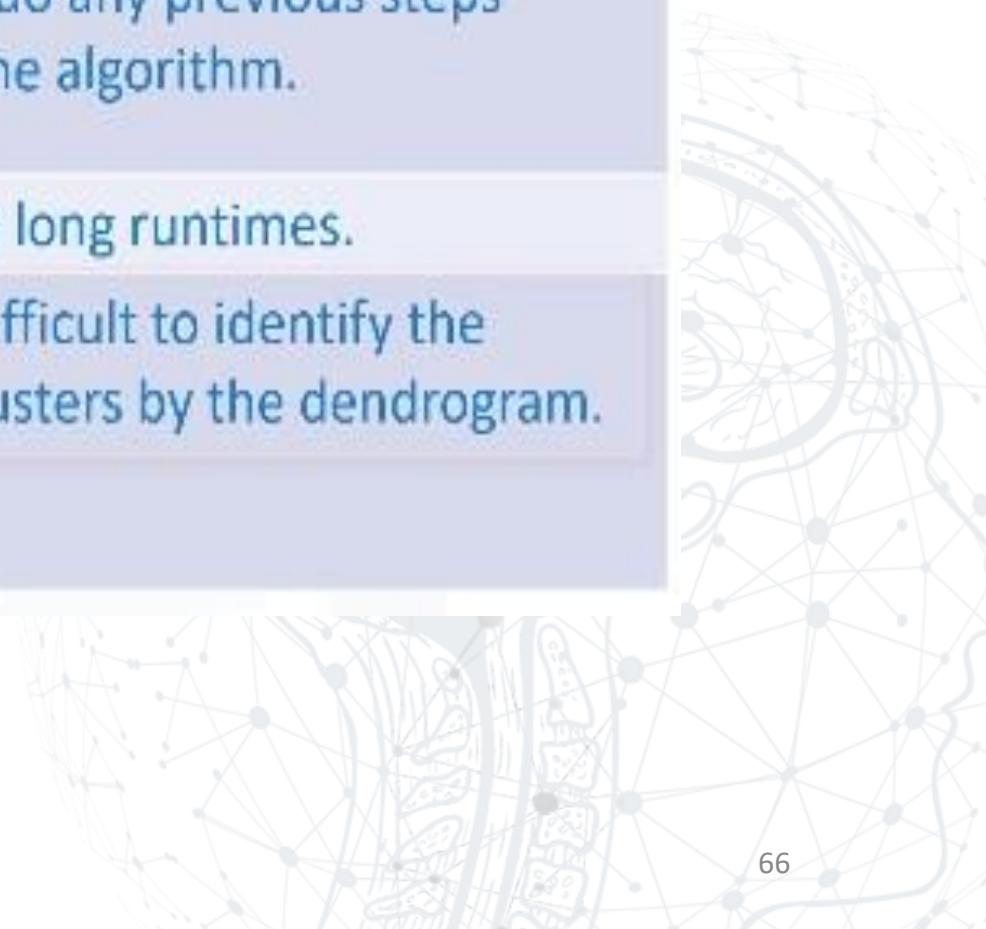


# Advantages vs Disadvantages

- There are 3 main advantages to using hierarchical clustering.
  - First, we do not need to specify the number of clusters required for the algorithm.
  - Second, hierarchical clustering is easy to implement.
  - And third, the dendrogram produced is very useful in understanding the data.
- There are some disadvantages as well.
  - First, the algorithm can never undo any previous steps. So for example, the algorithm clusters 2 points, and later on we see that the connection was not a good one, the program cannot undo that step.
  - Second, the time complexity for the clustering can result in very long computation times, in comparison with efficient algorithms, such k-Means.
  - Finally, if we have a large dataset, it can become difficult to determine the correct number of clusters by the dendrogram.

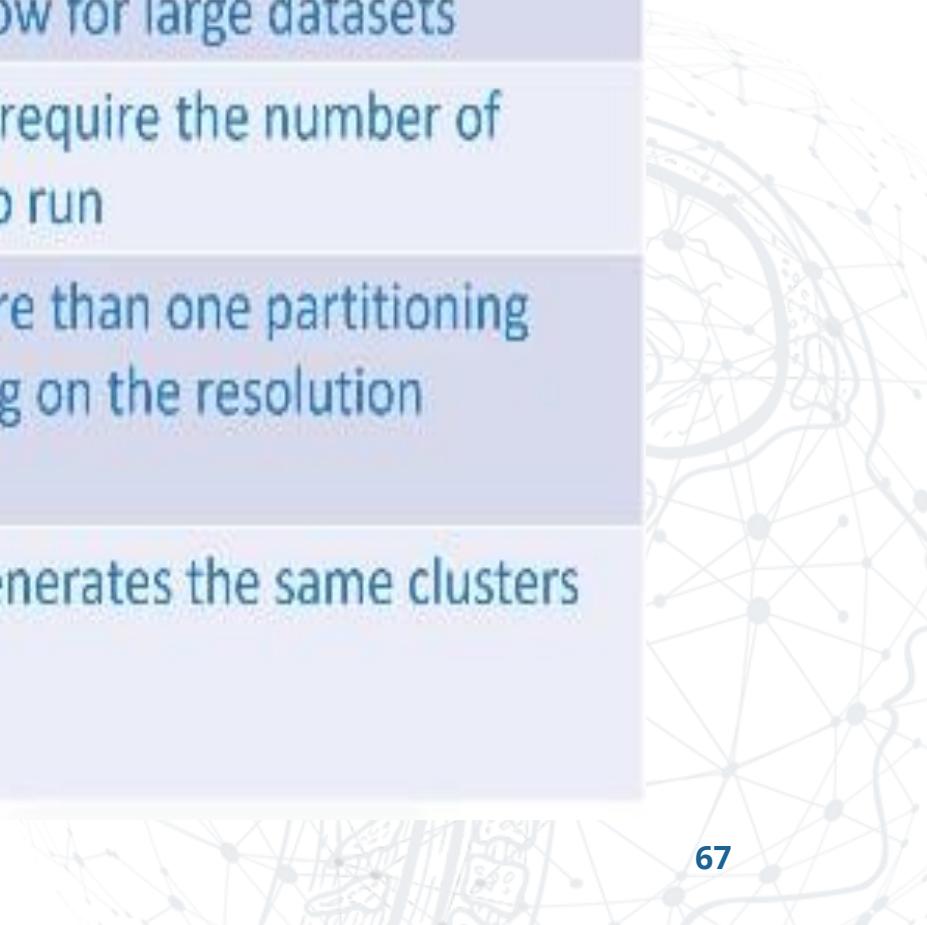
# Advantages vs. disadvantages

Advantages	Disadvantages
Doesn't required number of clusters to be specified.	Can never undo any previous steps throughout the algorithm.
Easy to implement.	Generally has long runtimes.
Produces a dendrogram, which helps with understanding the data.	Sometimes difficult to identify the number of clusters by the dendrogram.



# Hierarchical Clustering VS K- Means

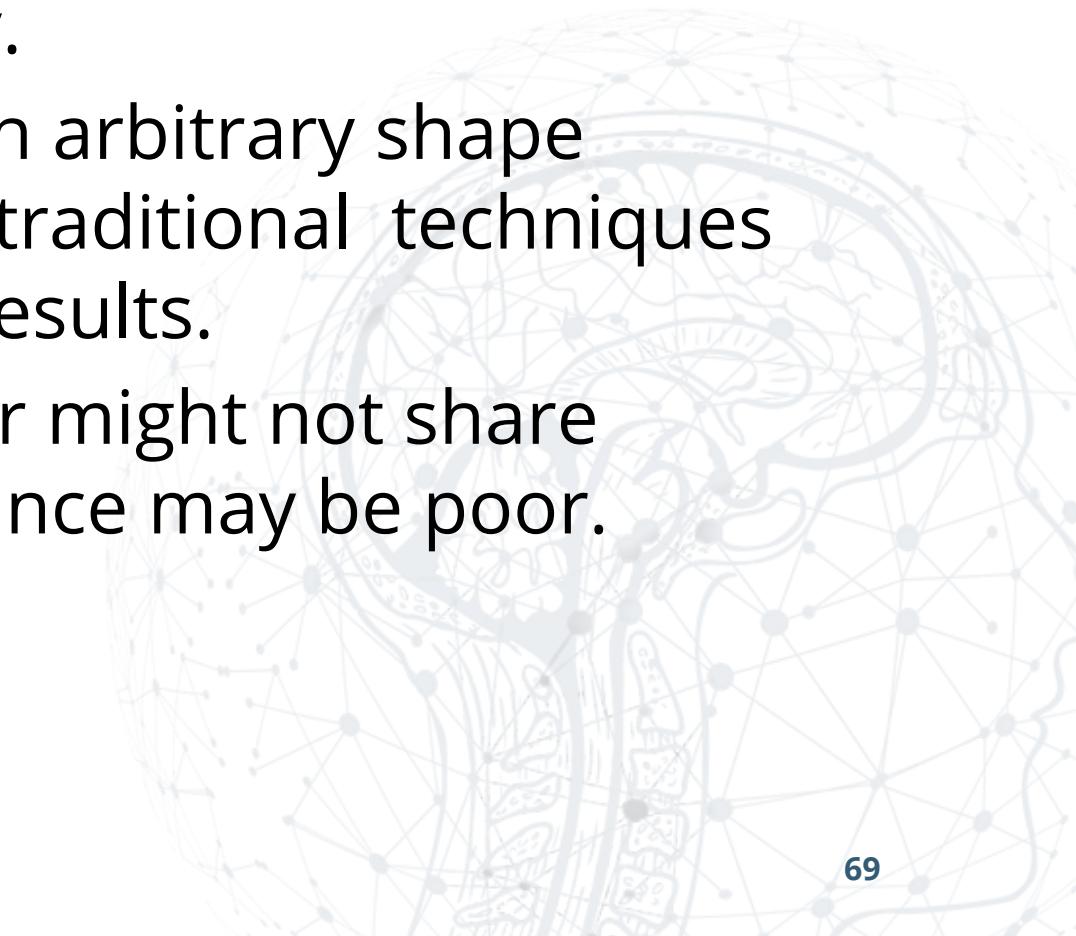
K-means	Hierarchical Clustering
1. Much more efficient	1. Can be slow for large datasets
2. Requires the number of clusters to be specified	2. Does not require the number of clusters to run
3. Gives only one partitioning of the data based on the predefined number of clusters	3. Gives more than one partitioning depending on the resolution
4. Potentially returns different clusters each time it is run due to random initialization of centroids	4. Always generates the same clusters



# DBSCAN

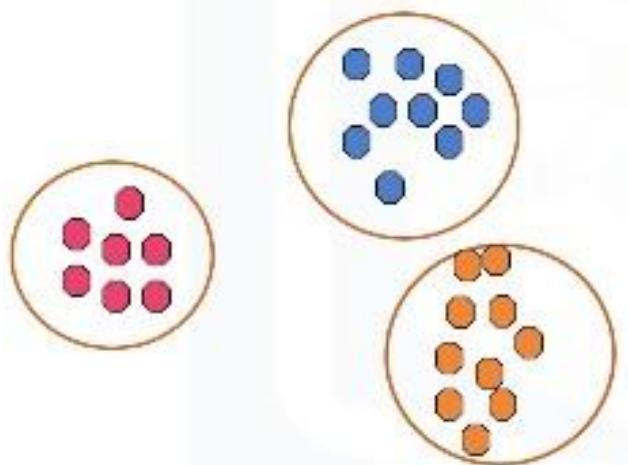
# DBSCAN Clustering

- Most of the traditional clustering techniques, such as k-means, hierarchical, and fuzzy clustering, can be used to group data in an un-supervised way.
- However, when applied to tasks with arbitrary shape clusters, or clusters within clusters, traditional techniques might not be able to achieve good results.
- That is, elements in the same cluster might not share enough similarity -- or the performance may be poor.

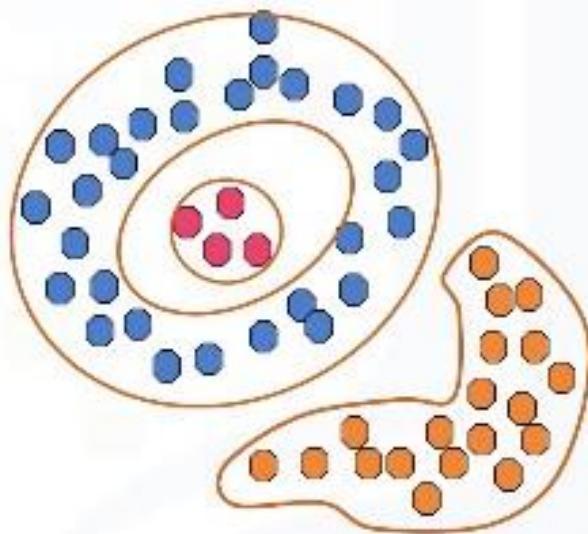


# Density-based clustering

- Spherical-shape clusters

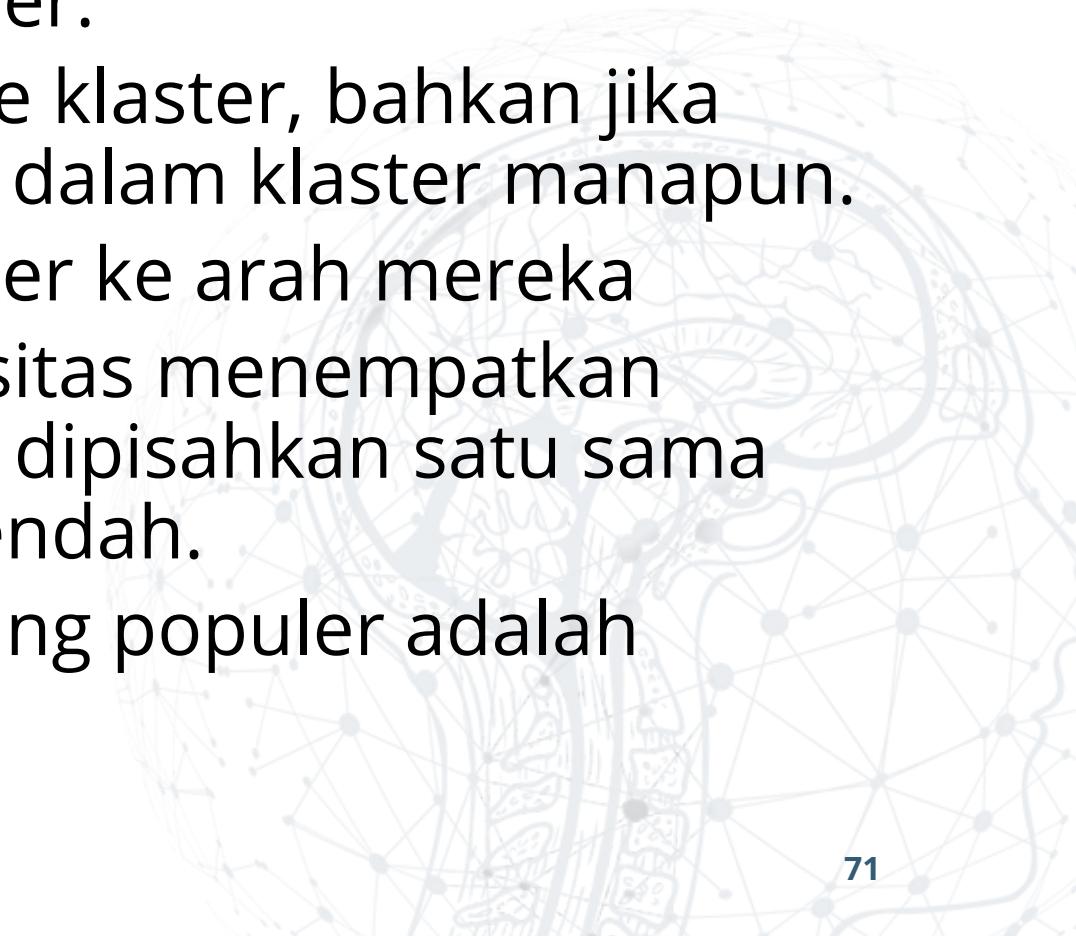


- Arbitrary-shape clusters



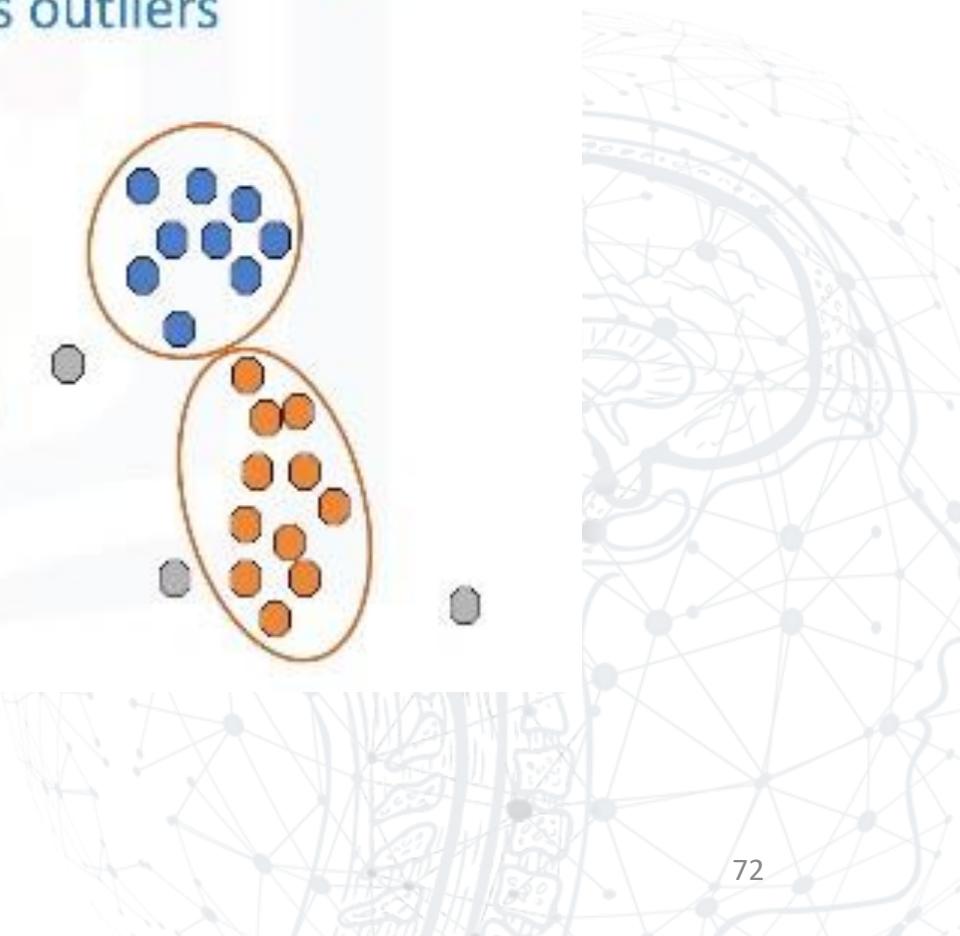
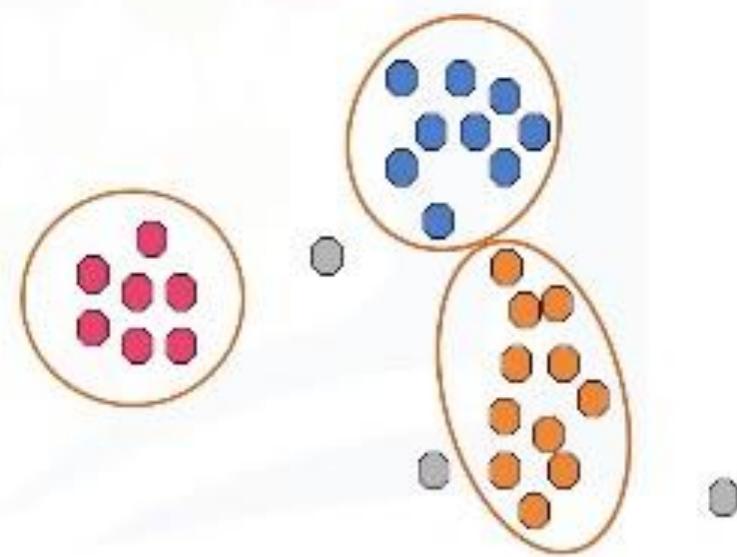
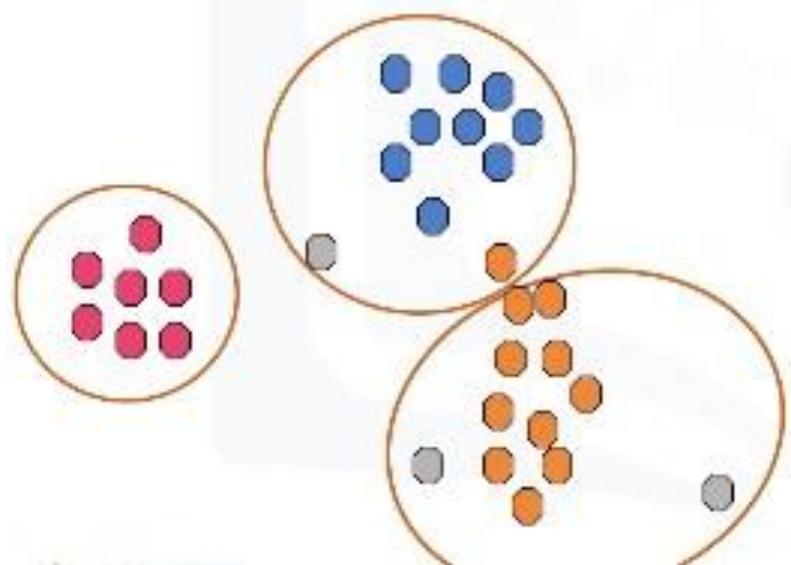
# DBSCAN Clustering

- Algoritma clustering seperti K-Means, mungkin mudah dipahami dan diterapkan dalam praktiknya, tetapi algoritma tsb tidak dapat mengidentifikasi outlier.
- Semua sampel harus dimasukkan ke klaster, bahkan jika mereka sebenarnya tidak termasuk dalam klaster manapun.
- Titik anomali menarik sentroid cluster ke arah mereka
- Sebaliknya, clustering berbasis densitas menempatkan daerah dengan densitas tinggi yang dipisahkan satu sama lain oleh daerah dengan densitas rendah.
- Jenis clustering berbasis densitas yang populer adalah DBSCAN.



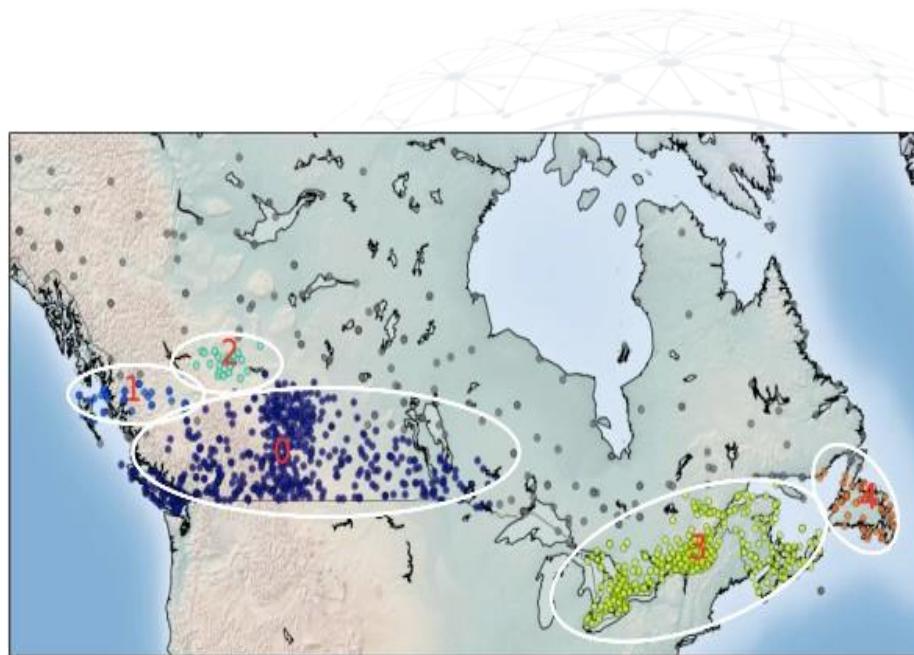
# k-Means Vs. density-based clustering

- k-Means assigns all points to a cluster even if they do not belong in any
- Density-based Clustering locates regions of **high density**, and separates outliers



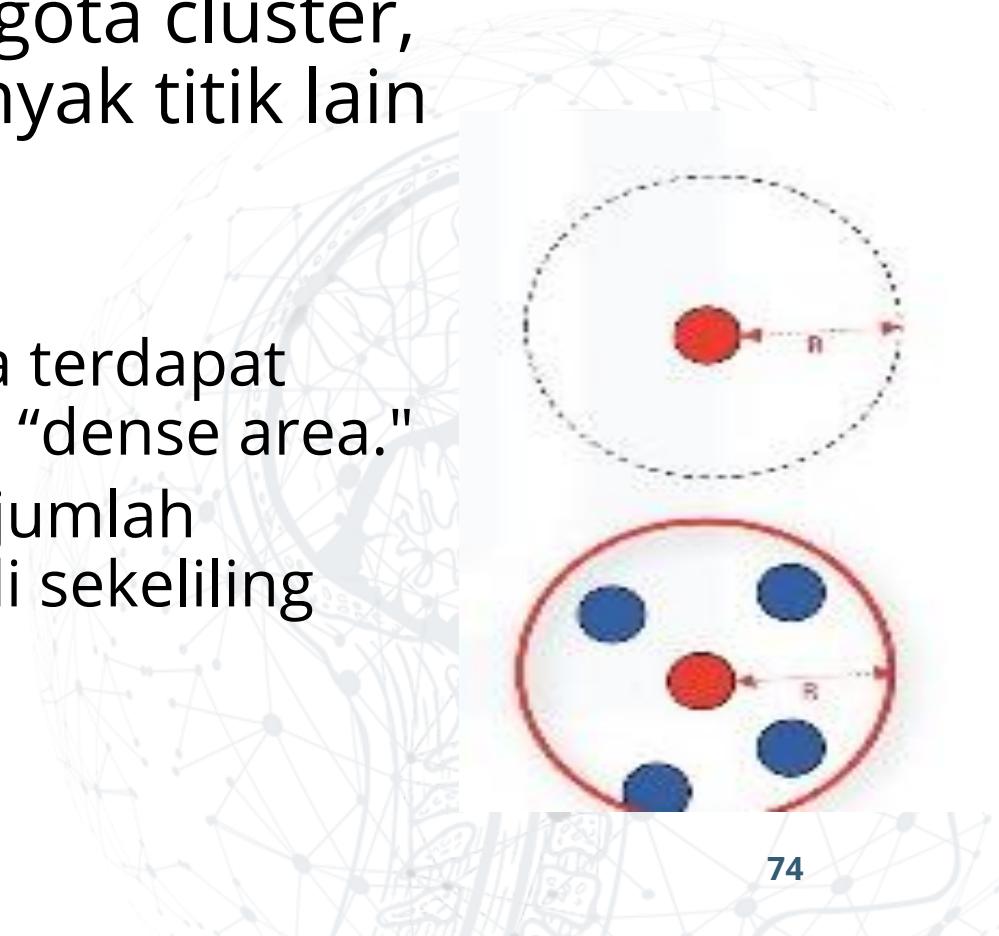
# DBSCAN Clustering

- DBSCAN is particularly effective for tasks like class identification on a spatial context.
- The wonderful attribute of the DBSCAN algorithm is that it can find out any arbitrary shape cluster without getting affected by noise.
- For example, this map shows the location of weather stations in Canada.
- DBSCAN can be used here to find the group of stations, which show the same weather conditions.
- As you can see, it not only finds different arbitrary shaped clusters, it can find the denser part of data-centered samples by ignoring less-dense areas or noises.



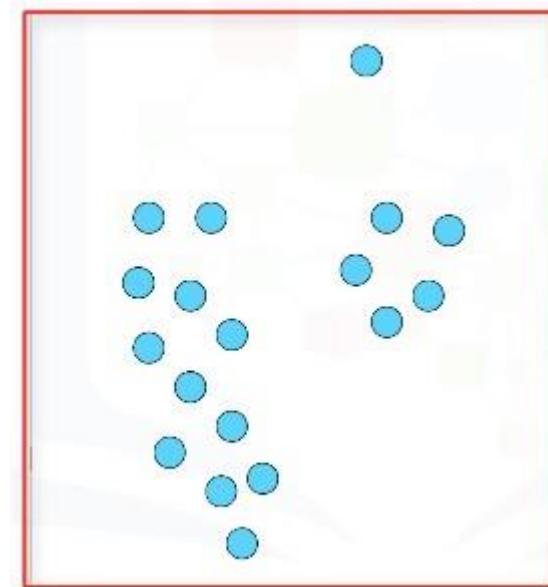
# DBSCAN Clustering

- DBSCAN → Density-Based Spatial Clustering of Applications with Noise.
- Ide dasar: jika sebuah titik adalah anggota cluster, maka titik tsb harus dekat dengan banyak titik lain dalam cluster itu.
- Parameter utama:
  1. Radius ( $R$ ) = radius yang ditentukan, jika terdapat "cukup" titik di dalamnya, maka disebut "dense area."
  2. Minimum Neighbor ( $M$ ) = menentukan jumlah minimum titik data yang kita inginkan di sekeliling untuk menentukan klaster.



# How DBSCAN Works

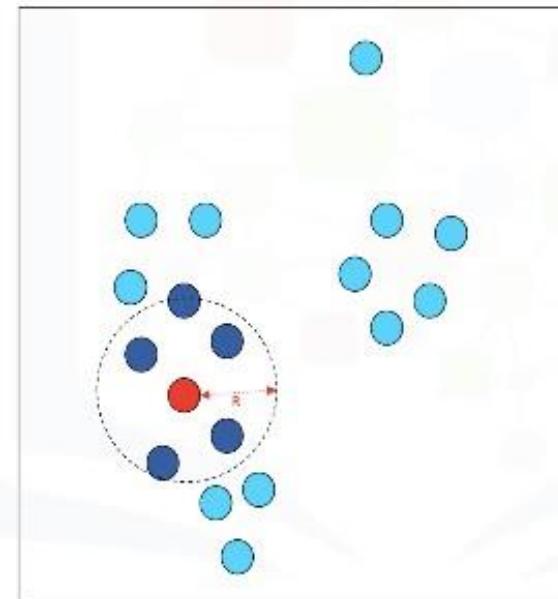
- Contoh:
  - $R = 2$  units. Misal 2 cm dari point of interest.
  - $M = 6$  points (termasuk point of interest)
- Jenis point:
  - Core, border, atau outlier point.
- Algoritma:
  - Kunjungi setiap point
  - Temukan jenisnya
  - Kelompokkan point sebagai cluster berdasarkan jenisnya.



$R = 2$  unit,  $M = 6$

# How DBSCAN Works

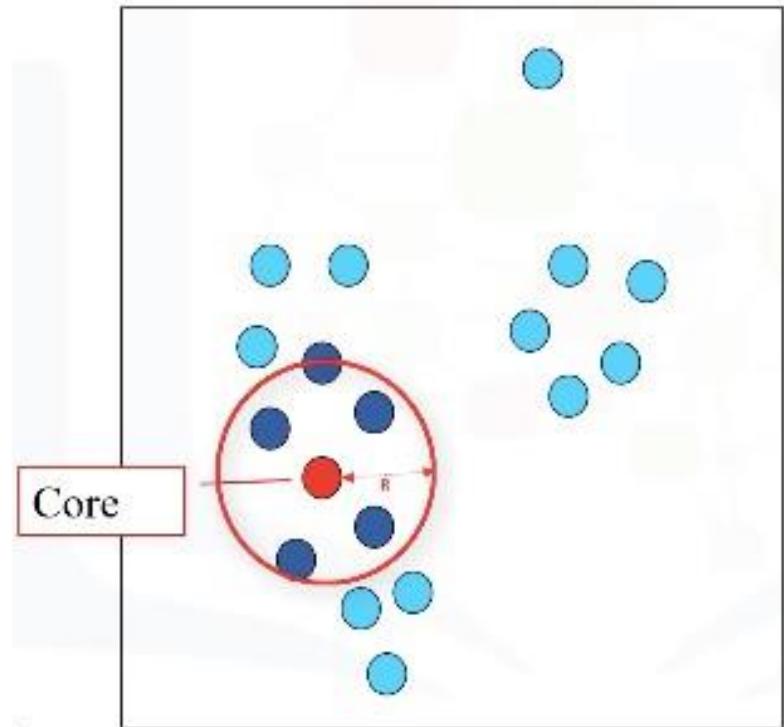
- Mari kita pilih titik secara acak. Pertama kita periksa untuk melihat apakah titik tsb merupakan core point.
- Sebuah titik merupakan **core point** jika dalam radius  $R$ , setidaknya ada sejumlah  $M$  titik.



$R = 2\text{unit}$ ,  $M = 6$

# How DBSCAN Works

- Misalnya, karena ada 6 titik di tetangga pada radius 2 cm dari titik merah, maka titik merah ini adalah core



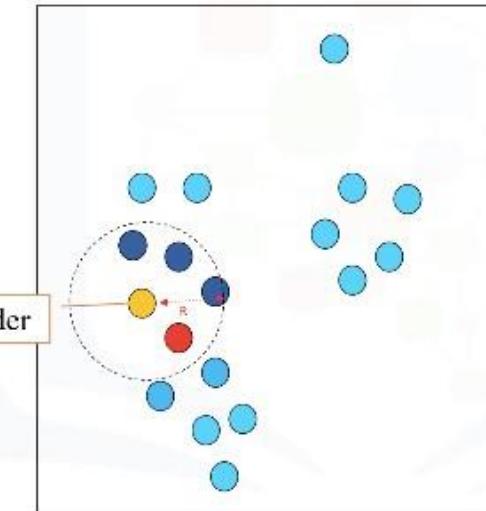
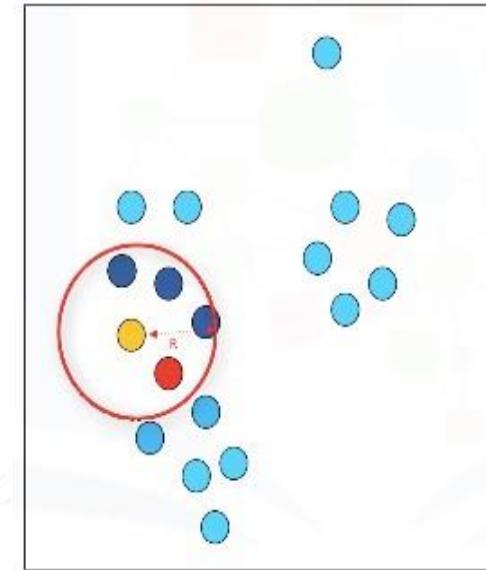
$R = 2\text{unit}$ ,  $M = 6$





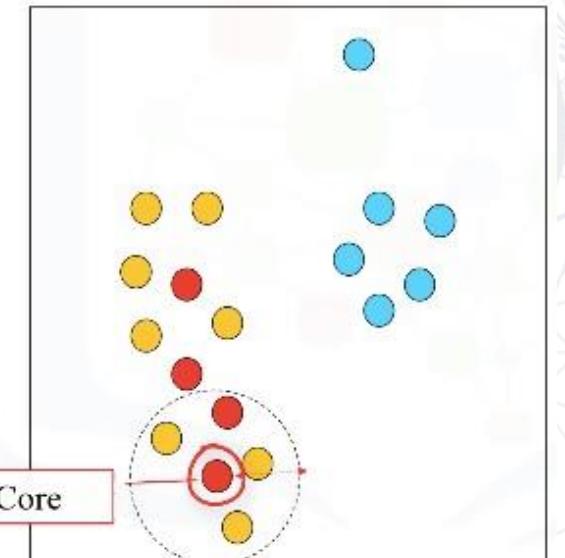
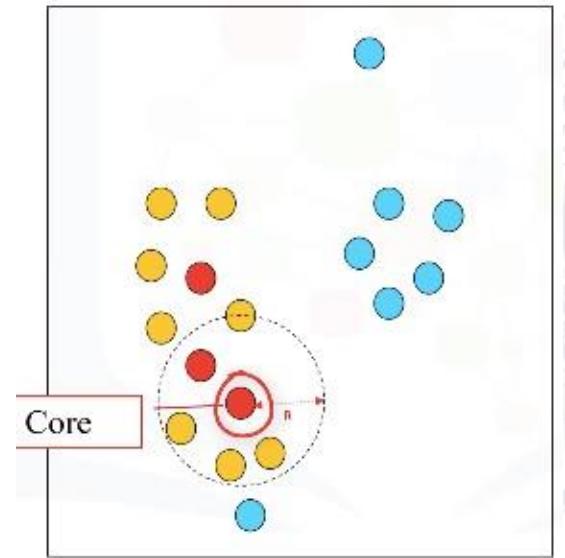
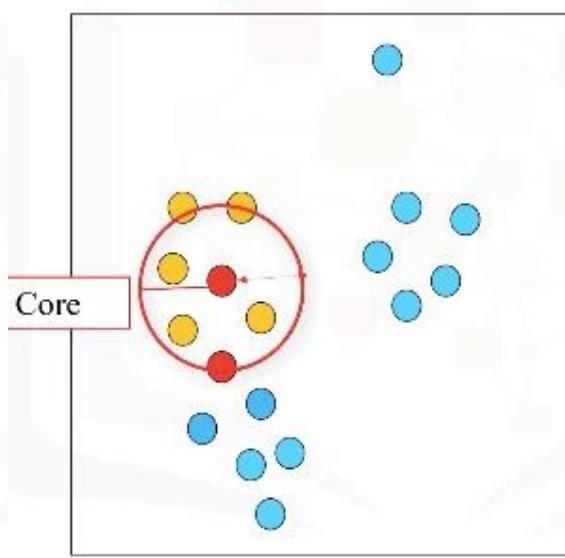
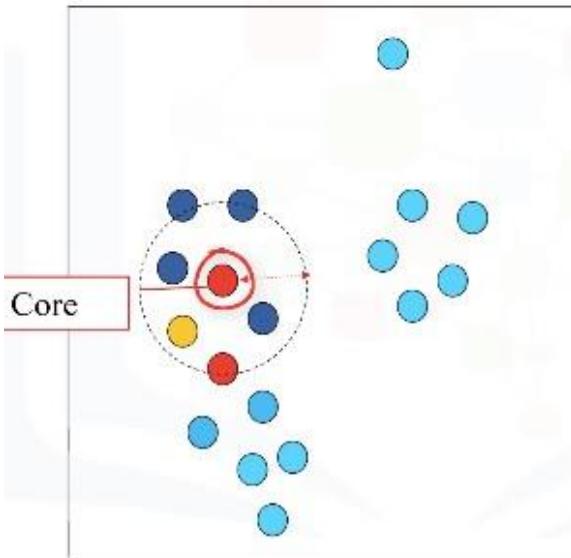
# How DBSCAN Works

- Perhatikan titik kuning
- Hanya ada 5 titik di lingkungan ini, termasuk titik kuning.
- Titik kuning merupakan **border point**
- Border point: Its neighborhood contains less than M data points, or It is reachable within R-distance from some core point.
- It means that even though the yellow point is within the 2-centimeter neighborhood of the red point, it is not by itself a core point, because it does not have at least 6 points in its neighborhood.



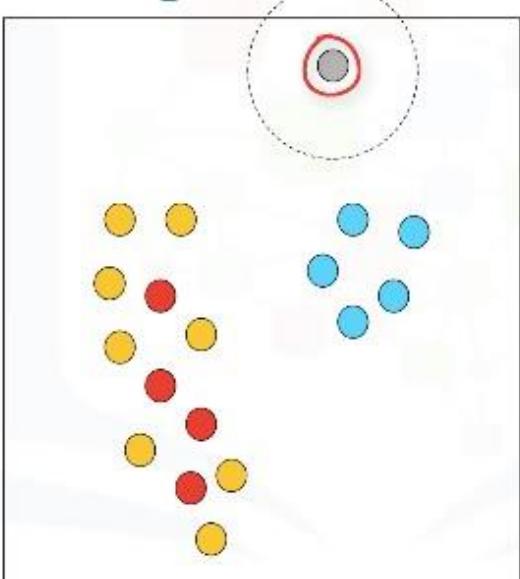
# How DBSCAN Works

- We continue with the next point.
- As you can see it is also a core point.
- And all points around it, which are not core points, are border points. Next core point. And next core point.



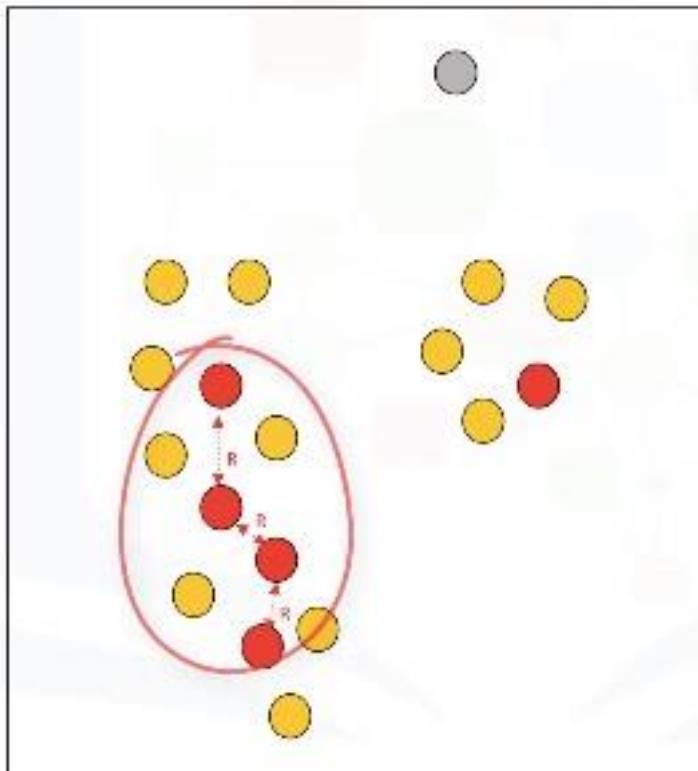
# How DBSCAN Works

- Let's take this point. You can see it is not a core point, nor is it a border point. So, we'd label it as an outlier.
- What is an outlier? An outlier is a point that: Is not a core point, and also, is not close enough to be reachable from a core point.



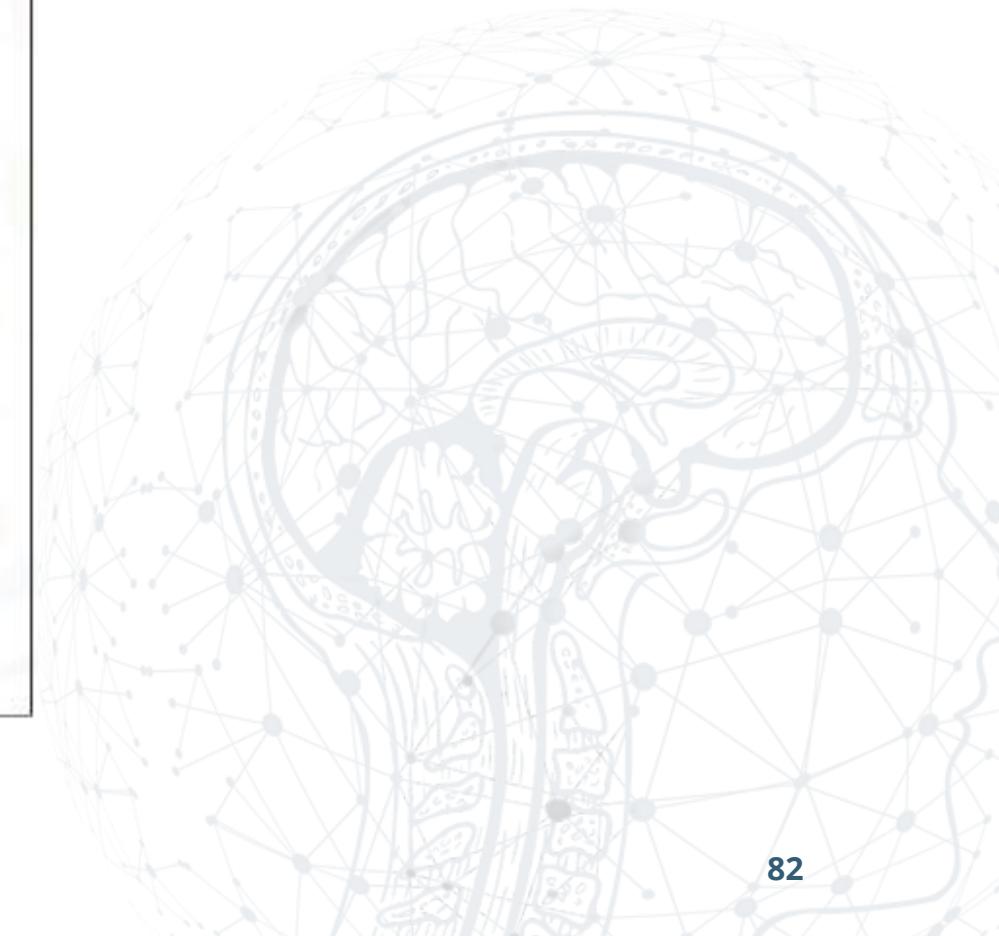
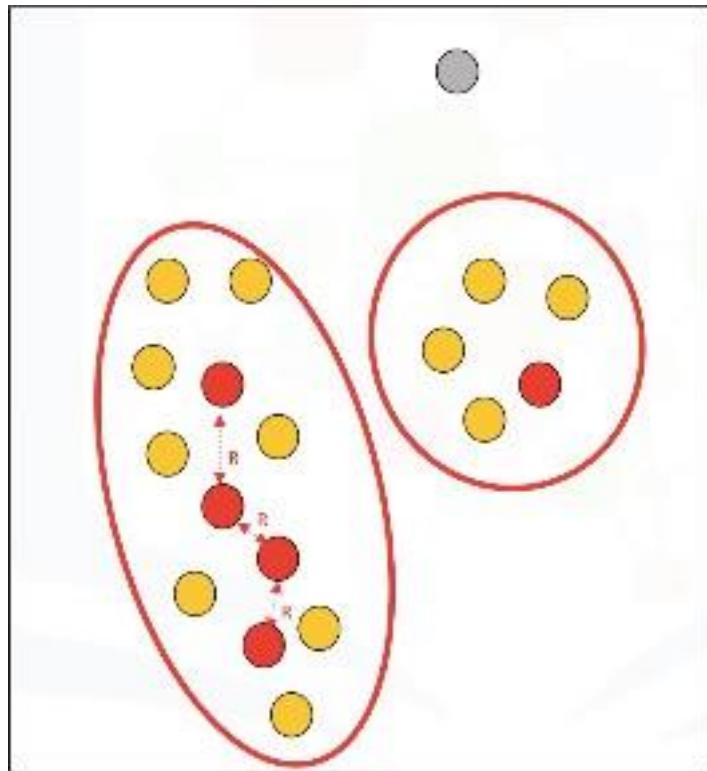
# How DBSCAN Works

We continue and visit all the points in the dataset and label them as either Core, Border, or Outlier.  
The next step is to connect core points that are neighbors, and put them in the same cluster.



# How DBSCAN Works

So, a cluster is formed as at least one core point, plus all reachable core points, plus all their borders. It simply shapes all the clusters and finds outliers as well.



# DBSCAN Advantages

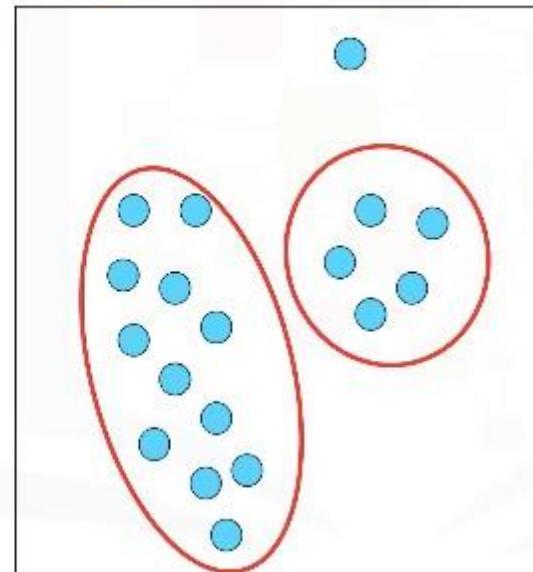
Let's review this one more time to see why DBSCAN is cool.

DBSCAN can find arbitrarily shaped clusters. It can even find a cluster completely surrounded by a different cluster.

DBSCAN has a notion of noise, and is robust to outliers.

On top of that, DBSCAN makes it very practical for use in many really world problems because it does not require one to specify the number of clusters, such as K in k-Means.

## Advantages of DBSCAN



1. Arbitrarily shaped clusters
2. Robust to outliers
3. Does not require specification of the number of clusters

# LAB

- Lihat Lab untuk Kmeans, Hierarchical Clustering dan DBSCAN di Cognitiveclass/EMAS

