

# CS5228: Knowledge Discovery and Data Mining

## Lecture 1 — Introduction & Overview

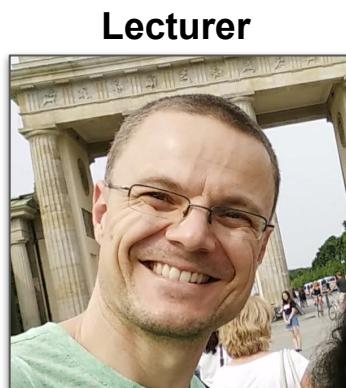
# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

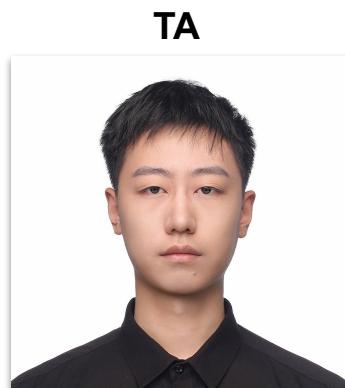
# Course Logistics

- **Lectures & Tutorials**
  - Friday, LT17: 6.30-8.30 pm / 8.30-9.30 pm
  - Physical classes (all recorded)
  - Announcements & materials on Canvas
- **Where to ask questions**
  - Canvas discussion (you are also strongly encouraged to answer questions!)
  - Email to teaching team (for private concerns or sensitive question, e.g., about an assignment)

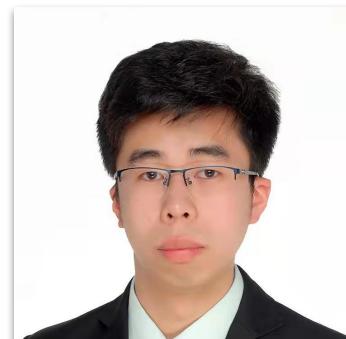
# Teaching Team



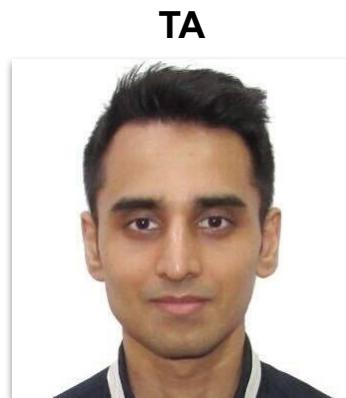
Christian von der Weth  
chris@comp.nus.edu.sg



Liu Chenyan  
chenyan@u.nus.edu



Gao Ruize  
e1023891@u.nus.edu



Rajdeep Singh Hundal  
rajdeep@nus.edu.sg



Luo Yang  
yang\_luo@u.nus.edu



Ayush Goyal  
e1124577@u.nus.edu

# Assessments

- **4 assignments (10% each)**
  - Programming tasks + theoretical questions (Python)
  - Discussions allowed, but code and answers must be submitted individually
- **Quiz in the last lecture (10% each)**
  - MCQ/MRQ Canvas quiz
  - Open-book but no Internet (screen recording will be required)
- **Midterm (20%)**
  - MCQ/MRQ + essay questions using Examplify
  - Open-book but blocked Internet
- **Project (30%)**
  - Group project (~4 students per group, more details after enrollment is complete)
  - Kaggle InClass Competition

# Lesson Plan (tentative deadlines; check Canvas!)

Release dates for assignments (Fri);  
submission deadline 2 weeks later (Thu)

Week	Date	Topics	Important Dates
1	Aug 16	Introduction	
2	Aug 23	Clustering I	
3	Aug 30	Clustering II	A1
4	Sep 06	Association Rules	
5	Sep 13	Regression & Classification I	A2
6	Sep 20	Regression & Classification II	
Recess	Sep 27	No class	
7	Oct 04	<b>Midterm Exam</b>	Midterm (Weeks 1-6)
8	Oct 11	Regression & Classification III	A3
9	Oct 18	Recommender Systems	
10	Oct 25	Graph Mining	A4
11	Nov 01	Dimensionality Reduction (recording, WBD)	
12	Nov 08	Data Stream Mining	
13	Nov 15	Review & Outlook + <b>Quiz</b>	Quiz 2 (Weeks 7-12)

# Course Policies

- Zero-Tolerance for Plagiarism
  - Students will be reported to University for disciplinary action for plagiarism/cheating offence
  - Offenders will receive F grade for the module (for any assessment with 10%+ weight!!!)
  - Assignments: discussion allowed but each student must submit their individual solutions
- Resources
  - <https://www.comp.nus.edu.sg/cug/plagiarism/>

# Course Policies

- AI use in class
  - Generally allowed for ideation, brainstorming, self-learning, improve writing
  - Take-home assignments: AI tools permitted but need to be acknowledged
  - Exams (midterm, quiz): AI tools not permitted incl. locally installed tools (e.g., open LLMs)
- Resources
  - <https://libguides.nus.edu.sg/new2nus/acadintegrity>  
(see the "Guidelines on the Use of AI Tools For Academic Work" tab)
  - <https://myportal.nus.edu.sg/studentportal/student-discipline/all/docs/NUS-Plagiarism-Policy.pdf>

# Course Policies

- Right Infringements on NUS Course Materials

*All course participants (including permitted guest students) who have access to the course materials on ~~LumiNUS~~ or any approved platforms by NUS for delivery of NUS modules are not allowed to re-distribute the contents in any forms to third parties without the explicit consent from the module instructors or authorized NUS officials.*

*Canvas*

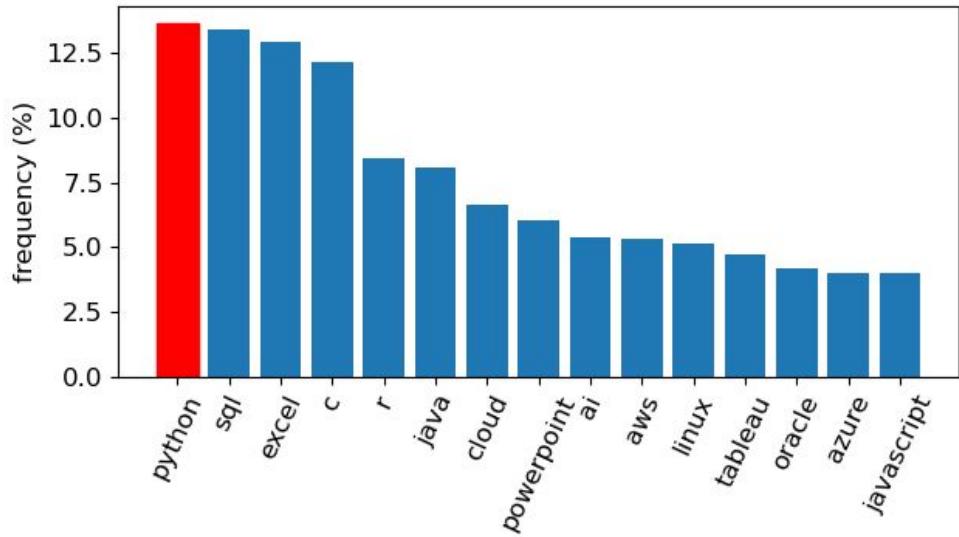
# What You Need

- Programming environment: Python + Jupyter
  - All implementation tasks will be in Python
  - Assignments will include Jupyter notebooks
  - Supplementary [Jupyter notebooks](#) for hands-on practice
- Common packages for data science
  - NumPy
  - pandas
  - NetworkX
  - scikit-learn



# Why Python?

- **Analysis of job descriptions**
  - 15k+ job offers from JobStreet  
(data analyst, data engineer, data scientist)
  - Quick-&-dirty keyword extraction
  - ...but check for yourself! :)



# Learning Outcomes

- Fundamental knowledge about concepts & algorithms in data mining
  - Nature of data: data representations, data and attribute types
  - Common data mining tasks and important algorithms (with their strengths and weaknesses)
  - Problems, risks & ethical issues of "unrestrained" data mining )
- Perform data mining tasks for new applications in practice
  - Given a dataset and task, select appropriate techniques to solve the task
  - Justify design and implementation decisions
  - Interpret results and assess limitations

assignment  
midterm  
quiz

project

# References

- **Textbooks** (useful but not required)
  - J. Leskovec, A. Rajaraman, J. Ullman: *Mining Massive Datasets*  
(online version available at: <http://www.mmmds.org/>)
  - P. Tan, M. Steinbach, A. Karpatne, V. Kumar: *Introduction to Data Mining*
  - More in Canvas Readings
- ...the Web

# Outline

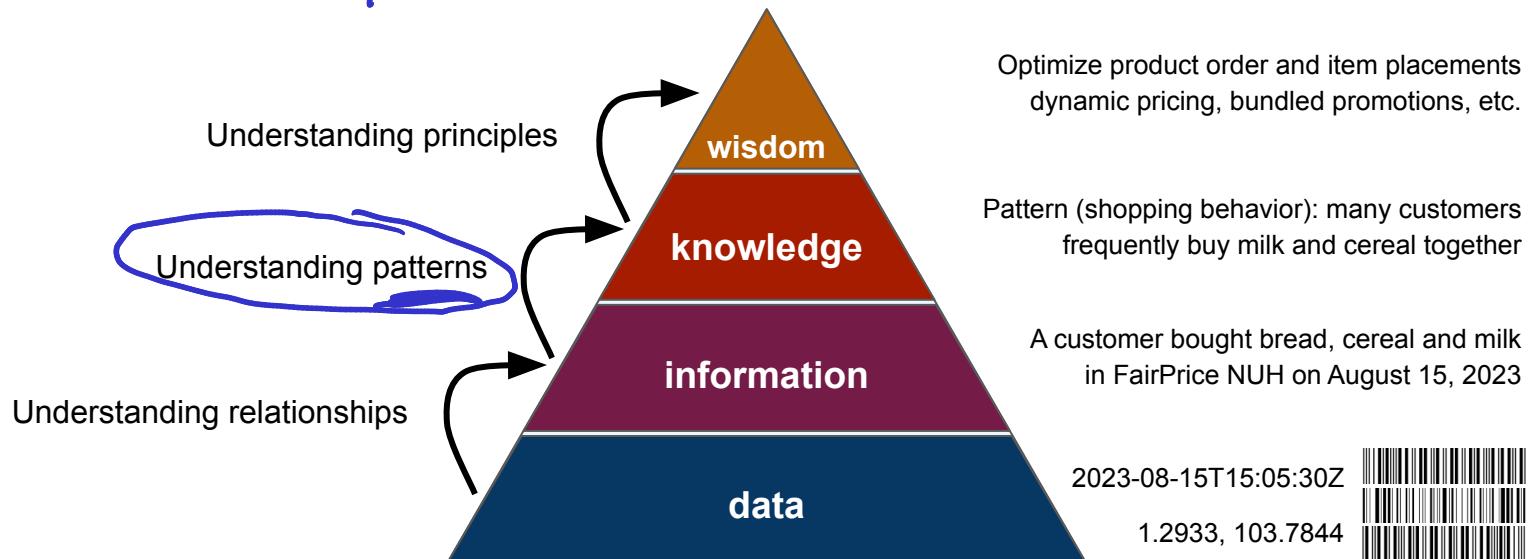
- Course Logistics
- **Overview**
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

# What is Knowledge Discovery & Data Mining

*"The **non-trivial** extraction of **implicit**, previously **unknown**, and potentially **useful** information from data."*

(Frawley, Piatetsky-Shapiro, Matheus; 1991)

DICW    pyramide



# From Data to Knowledge

## Data Selection

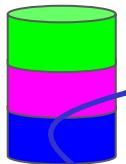
- Identify relevant data to solve a given task

## Data Transformation

- Convert data into suitable representation

## Postprocessing

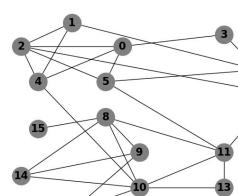
- Visualization
- Interpretation
- Understanding
- ...



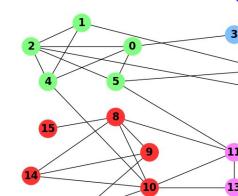
Age	Education	Marital Status	Income Level	Credit Approval
23	Masters	Single	Mid	No
23	College	Married	Low	No
35	College	Single	Mid	No
23	Masters	Single	Mid	No
35	College	Married	High	Yes
26	Masters	Single	High	No
41	PhD	Single	Mid	Yes
18	Poly	Single	Low	No
55	Poly	Married	High	Yes
...	...	...	...	...

Target Data

Preprocessed Data



Transformed Data



Patterns



## Data Preprocessing

- Handling missing data
- Duplicate elimination
- Feature selection
- Normalization
- ...

## Data Mining

- Clustering
- Classification
- Regression
- Associations
- Correlations
- ...

# What is NOT Knowledge Discovery & Data Mining?

- Trivial extraction of information/patterns from data
  - Looking up a phone number in phone directory
  - Dividing students based on their degree course
  - Calculating the total sales of a company
- Data analysis not yielding patterns (i.e., new information)
  - Monitoring a patient's heart rate for abnormalities
  - Querying a Web search engine

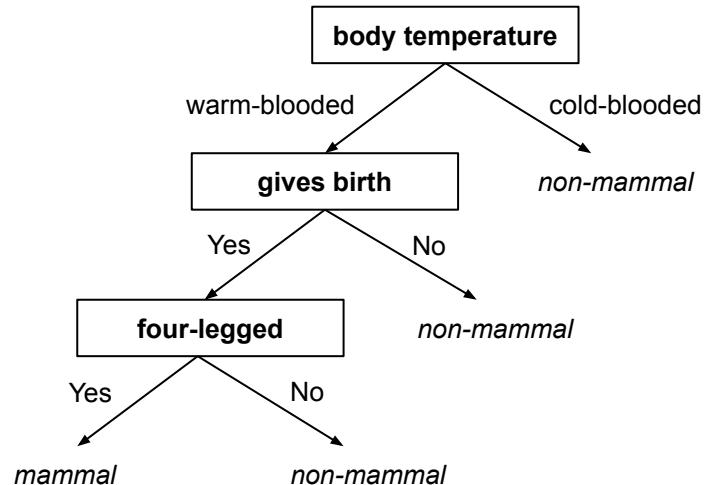
# What Makes a Pattern Useful or Meaningful?

*"If you torture the data long enough, it will confess to anything"*

(Ronald Coase; 1981 — slightly paraphrased)

- Main goal: **Generalizability**
  - Patterns should remain accurate over unseen data
  - Common causes: small and/or biased data

*But what about humans and platypuses, etc.?*

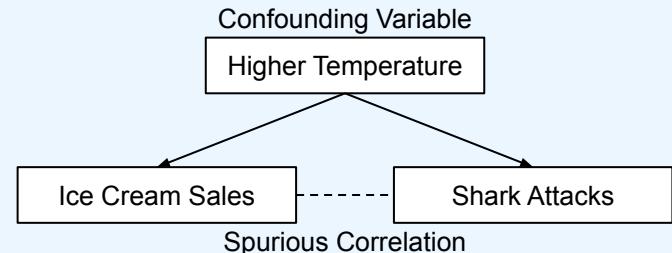


# There is Always Some Pattern in Your Data (even in random data)

- Bizarre and Surprising Insights

- "Female-named hurricanes kill more people than male hurricanes."
- "Users of Chrome and Firefox browsers make better employees."
- "Shark attacks increase when ice cream sales increase"
- "Music taste predicts political affiliation."
- "A job promotion can lead to quitting."
- "Vegetarians miss fewer flights."
- "Smart people like curly fries."
- "Higher status, less polite."

**Important:** Patterns indicate correlations, but correlation does not imply causation!



# Spotting "Shady" Patterns — Reality Check

- What is the (perceived) difference between the 2 statements below?
  - In the context of identifying and/or assessing patterns

*"The higher the sales of ice cream,  
the higher the number of shark attacks."*

vs.

*"The higher the concentration of  
anti-mullerian hormone, the lower the  
concentration of follicle-stimulating hormone."*

Note: "This doesn't make sense!" is rarely a good argument.

# Data Mining Gone Wrong

*"Your scientists were so preoccupied with whether they could,  
they didn't stop to think if they should."*

(Ian Malcolm; Jurassic Park, 1991)

Malaysian Bar troubled over judges using AI for sentencing

Applicant pre-selection by algorithm:  
How to convince an AI of yourself

Algorithms are deciding who gets the  
first vaccines. Should we trust them?

How Target Figured Out A Teen Girl  
Was Pregnant Before Her Father Did

Millions of black people affected by  
racial bias in health-care algorithms

The computers rejecting your job application

# Quick Quiz

What is (arguably) **NOT**  
a "proper" Data Mining task?  
(given a dataset of supermarket transactions)

**A**

Finding the largest sets of products  
most frequently bought together

**B**

Finding groups of similar users  
based on the buying behavior

**C**

Finding all purchases of a bundled  
promotion (i.e., multiple items)

**D** (✓)

Finding the products most frequently  
bought on weekends after 6pm

# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - **Common Data Mining tasks**
  - Types of data & data representations
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

# Methods — Association Rules

- Input: transactional data
  - Transaction: data record with set of items
  - Set of items are from a fixed collection
- Pattern: Association rules
  - Rules predicting the occurrence of items based on the occurrence of other items

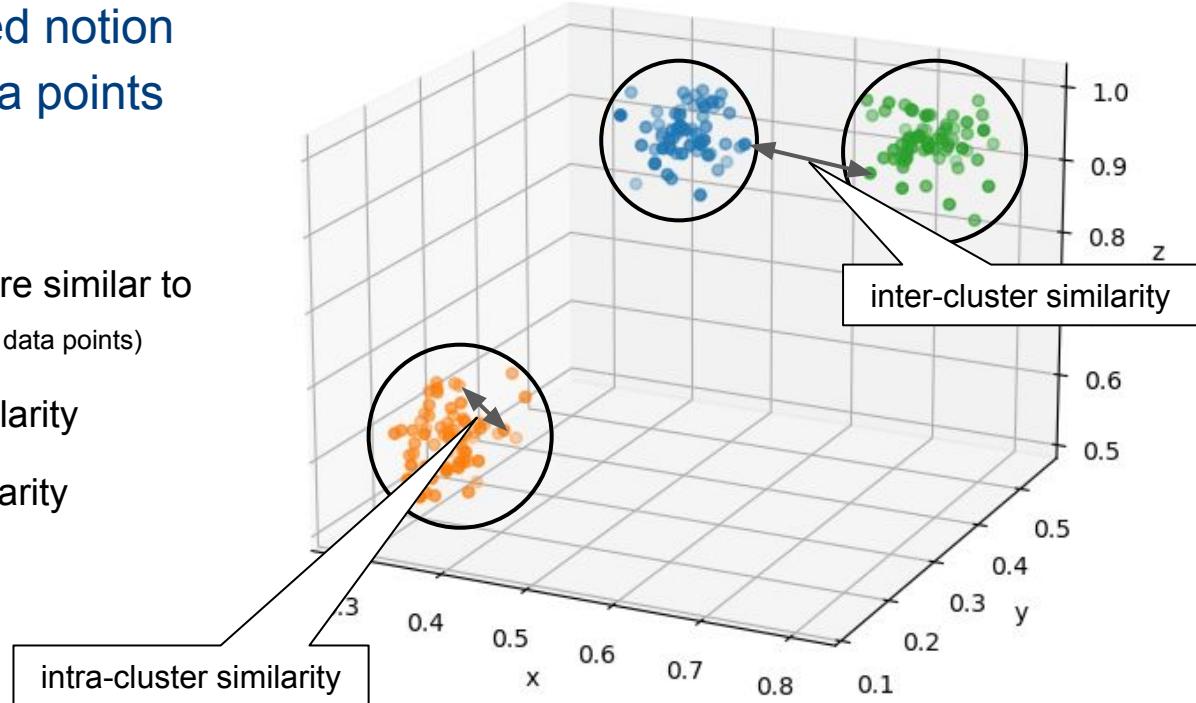
ID	Items
1	covid-19, anosmia, cough, fatigue
2	flu, anosmia, headache
3	covid-19, anosmia, headache, fatigue, fever
4	covid-19, flu, anosmia, fatigue
5	flu, depression, fatigue, fever, headache
...	



{anosmia, fatigue} → {covid-19}

# Methods — Clustering

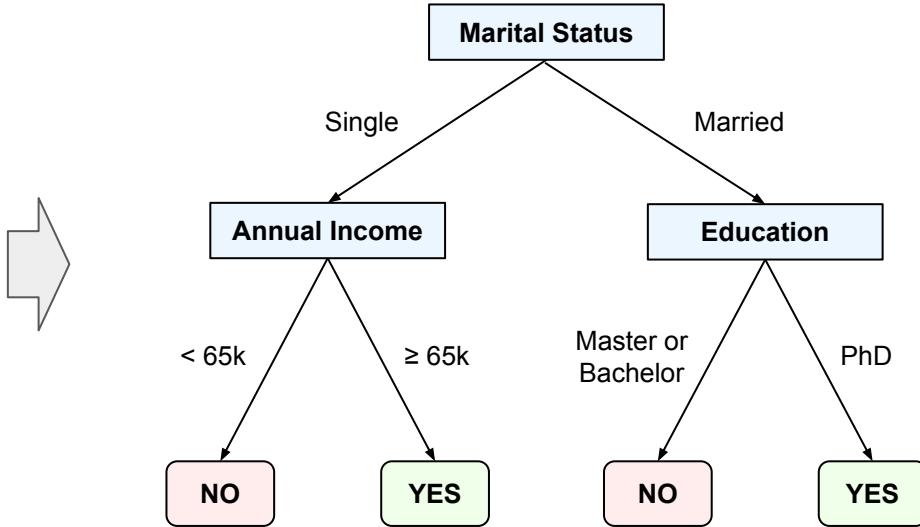
- Input: Data & well-defined notion of similarity between data points
- Pattern: Clusters
  - Groups of data points that are similar to each other (compared to the other data points)
  - Maximize **intra-cluster** similarity
  - Minimize **inter-cluster** similarity



# Methods — Classification

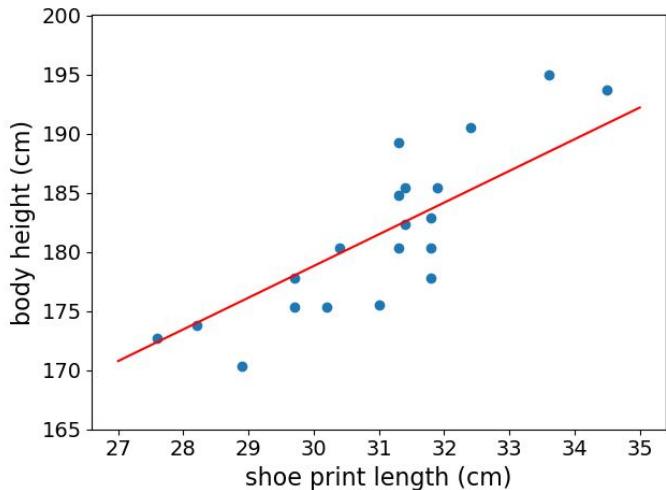
- Input: Dataset with multiple attributes
- Pattern: **Categorical** value of an attribute as function of other attribute values
  - K-Nearest Neighbor, Decision Trees, Linear Classification, etc.

Age	Edu- cation	Marital Status	Annual Income	Credit Default
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes



# Methods — Regression

- Input: Dataset with multiple attributes
- Pattern: **Numerical value of an attribute as function of other attribute values**
  - K-Nearest Neighbor, Regression Trees, Linear Regression, etc.



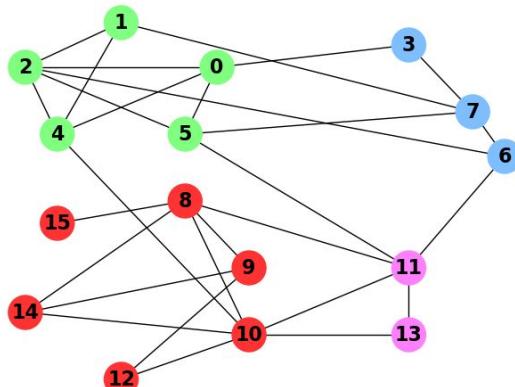
Question: "*What is the expected height of a person that leaves a shoe print of size 32.2cm?*"

Answer: ?

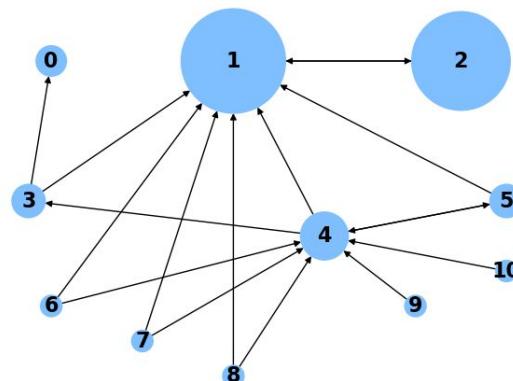
# Methods — Graph Mining

- Input:  $G = (V, E)$ 
  - Set of vertices (or nodes)  $V$  (data points)
  - Set of edges  $E$  (relationship between data points)
- Patterns based on graph structure, e.g.:

Finding communities of nodes



Finding "important" nodes



# Methods — Recommender Systems

- **Input: User-rated items**

(e.g., movies rated by viewers)

- How would Bob rate the movie "Heat"?
- Should "Heat" be recommended to Bob?

	Clueless	Heat	Jarhead	Big	Rocky
Alice	2	4	5	0	1
Bob	1	???	4	0	2
Claire	1	0	4	3	0
Dave	5	1	2	0	5
Erin	1	5	3	0	3

- **Patterns based on similarities to predict missing values**

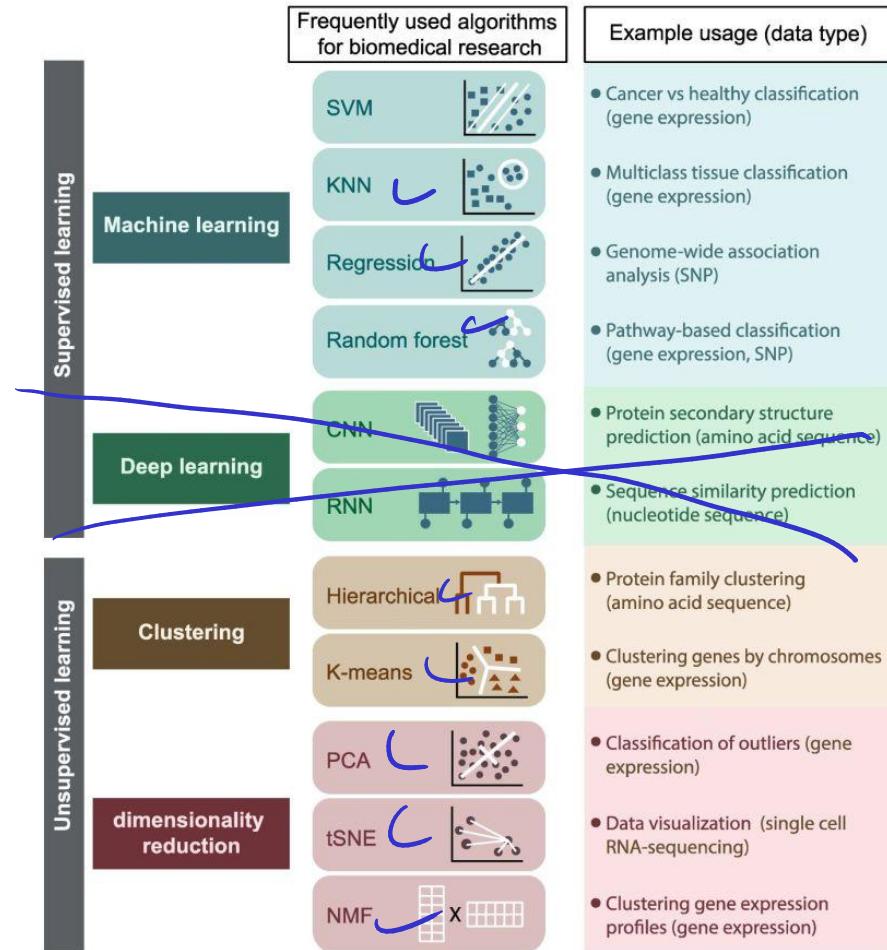
- Exploiting features of items
- Exploiting similarities between users or items

# Data Mining in Practice

- Example: Biomedical Research
  - Set of important data mining algorithms
  - Relevant for many other fields
  - Many covered here in CS5228  
(main exception: no deep learning)

Humanity

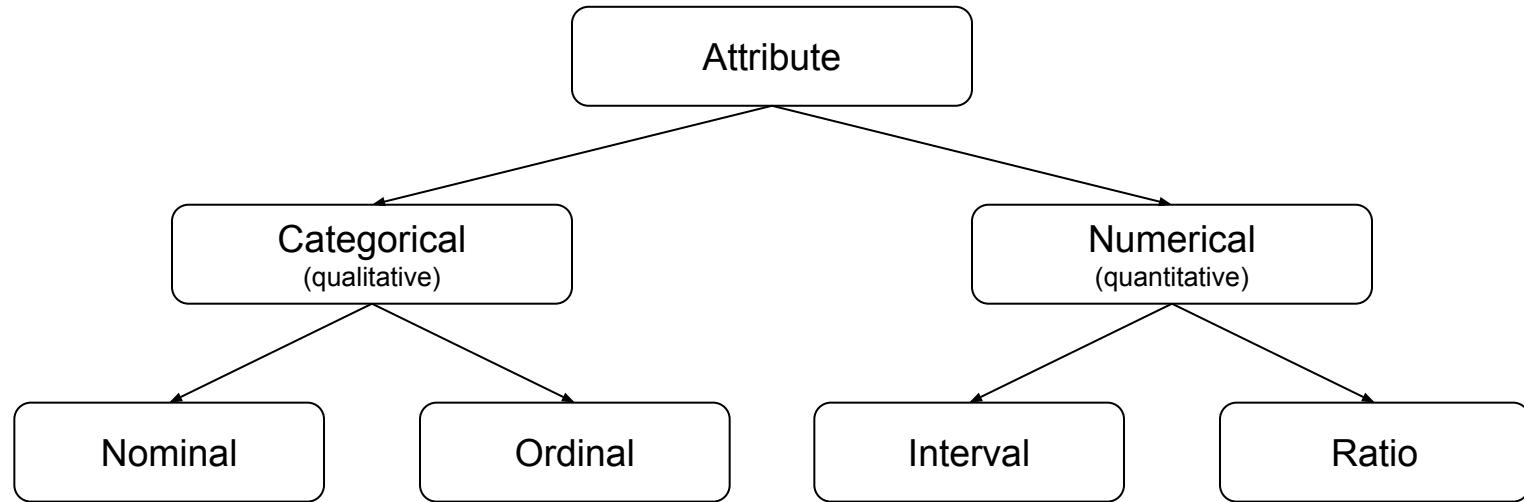
Physics



# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - **Types of data & data representations**
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

# Types of Attributes / Features



- Values are only labels
- Operations:  
 $=, \neq$
- Examples: sex (m/f), eye color, zip code

- Values are labels with a meaningful order
- Operations:  
 $=, \neq, <, >$
- Examples: street numbers, education level

- Values are measurements with a meaningful distance
- Operations:  
 $=, \neq, <, >, +, -$
- Examples: body temperature in  $^{\circ}\text{C}$ , calendar dates

- Values are measurements with a meaningful ratio
- Operations:  
 $=, \neq, <, >, +, -, *, /$
- Examples: age, weight, income, blood pressure

temp in K

# Types of Data

## (Well-)Structured Data

- Highly organized: adheres to predefined data model
- Each object has the same fixed set of attributes
- Easy to search, aggregate, manipulate, analyze data
- Examples: Relational databases, spreadsheets

Age	Education	Marital Status	Income Level	Credit Approval
23	Masters	Single	Mid	No
35	College	Married	High	Yes
26	Masters	Single	High	No
41	PhD	Single	Mid	Yes
18	Poly	Single	Low	No
55	Poly	Married	High	Yes
....	...	...	...	...

## Semi-Structured Data

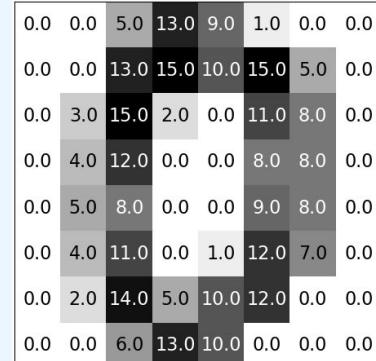
- No rigid data model: mix of structured & unstructured data
- Data exchange formats: XML, JSON, CSV
- Tagged unstructured data (e.g., photo + date/time, location, exposure, resolution, flash, etc)



**Name:** nature.jpg  
**Width:** 640 pixels  
**Height:** 637 pixels  
**Type:** JPEG image  
**Bytes:** 80.4 kB  
**Folder:** tmp

## Unstructured Data

- No fixed data model
- Requires more advanced data analysis techniques
- Examples: images, videos, audio, text, social media



# Types of Data Representations — Record Data

**Data matrix:** collection records; each record consisting of a fixed set of attributes

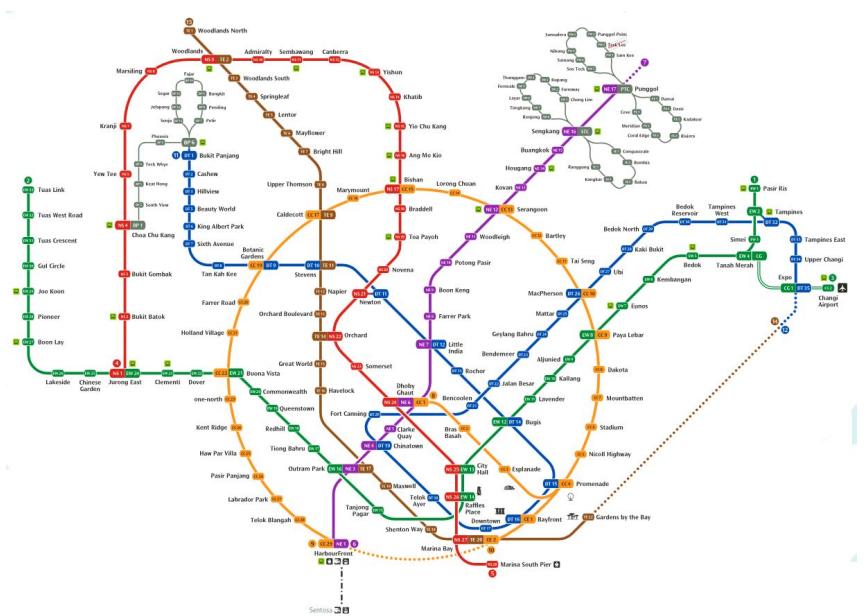
Age	Edu- cation	Marital Status	Annual Income	Credit Approval
23	Masters	Single	75k	Yes
35	Bachelor	Married	50k	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	85k	No
30	Bachelor	Single	60k	No
35	PhD	Married	60k	Yes
28	PhD	Married	65k	Yes

**Transaction data:** collection records; each record involves a set of items

ID	Items
1	covid-19, anosmia, cough, fatigue
2	flu, anosmia, headache
3	covid-19, anosmia, headache, fatigue, fever
4	covid-19, flu, anosmia, fatigue
5	flu, depression, fatigue, fever, headache
...	

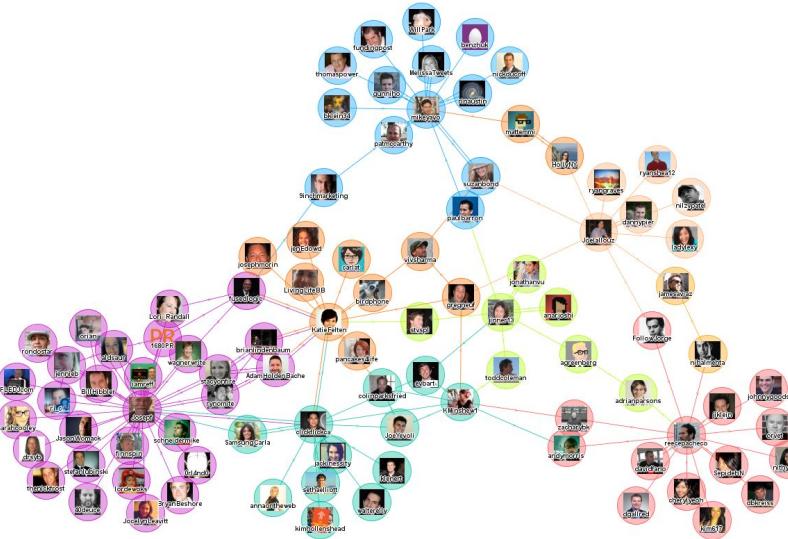
# Types of Data Representations — Graph Data

Example: traffic data



Source: <https://www.lta.gov.sg/>

Example: social network data



Source: <http://touchgraph.com/>

# Types of Data Representations — Ordered Data

Example: stock prices (sequence of data points)



# Quick Quiz

What type of attribute is  
**Annual Income?**

ID	Age	Edu- cation	Marital Status	Annual Income	Credit Approval
101	23	Masters	Single	75k	Yes
102	35	Bachelor	Married	50k	No
103	26	Masters	Single	70k	Yes
104	41	PhD	Single	95k	Yes
105	18	Bachelor	Single	40k	No
...	...	...	...	...	...

A

Nominal

B

Ordinal

C

Interval

D 

Ratio

# Quick Quiz

What type of attribute is  
**Education?**

ID	Age	Edu- cation	Marital Status	Annual Income	Credit Approval
101	23	Masters	Single	75k	Yes
102	35	Bachelor	Married	50k	No
103	26	Masters	Single	70k	Yes
104	41	PhD	Single	95k	Yes
105	18	Bachelor	Single	40k	No
...	...	...	...	...	...

A (✓) Nominal

B (✓) Ordinal

C Interval

D Ratio

# Quick Quiz

What type of attribute is  
**ID?**

ID	Age	Edu- cation	Marital Status	Annual Income	Credit Approval
101	23	Masters	Single	75k	Yes
102	35	Bachelor	Married	50k	No
103	26	Masters	Single	70k	Yes
104	41	PhD	Single	95k	Yes
105	18	Bachelor	Single	40k	No
...	...	...	...	...	...

A ✓ Nominal

B Ordinal

C Interval

D Ratio

# Outline

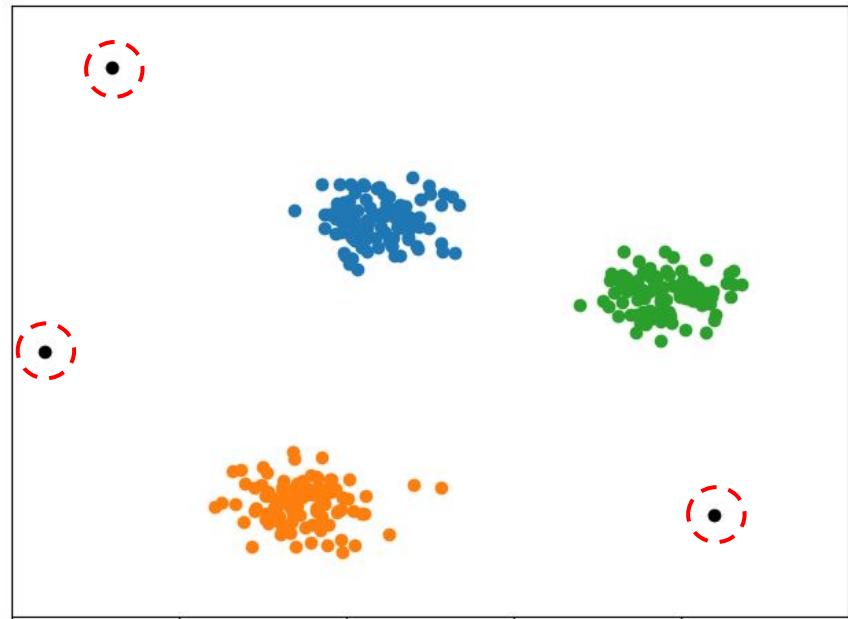
- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - **Data quality**
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

# Data Quality — Noise

- Data = true signal + noise
  - Sensor readings from faulty devices  
(also intrinsic noise or external influences)
  - Errors during data entry  
(by humans or machines)
  - Errors during data transmission
  - Inconsistencies in data formats  
(e.g., iso time vs unix time, DD/MM/YYYY vs. MM/DD/YYYY)
  - Inconsistencies in conventions  
(e.g., meters vs. miles, meters vs. centimeters)

# Data Quality — Outliers

- Outlier: Data point with attribute values considerably different from other points
- Case 1: Outliers are noise
  - Negatively interfere with data analysis
  - (Try to) remove outliers and/or use methods less prone to outliers
- Case 2: Outliers are targets
  - (the goal is to find rare/strange/odd data points)
  - Credit card fraud
  - Intrusion detection



# Data Quality — Missing Values

- Common causes

- Attribute values not collected  
(e.g., broken sensor, person refused to report age)
- Attributes not applicable in all cases  
(e.g., no income information for children)

- Handling missing values

- Remove data points with missing values
- Remove attributes with missing values  
(not all attributes are always equally important)
- (Try to) fill in missing values  
(e.g., average temperature readings of nearby sensors)

Age	Edu- cation	Marital Status	Annual Income	Credit Default
23	Masters	Single	75k	Yes
N/A	Bachelor	Married	N/A	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	N/A	No
30	Bachelor	Single	N/A	No
35	PhD	Married	60k	Yes
N/A	PhD	Married	65k	Yes

# Data Quality — Duplicates

- **Duplicates:** Data points referring to the same object/entity

(e.g., two records in a database refer to the same real-world person)

- Exact duplicates: data points have the same attribute values
- Near duplicates: data points (slightly) differ in their attribute values  
(e.g., same person with the same phone number but in different formats)

- **Task: Duplicate Elimination**

- Relatively easy for exact duplicates
- Generally very difficult for near duplicates

 **Note:** Duplicates are a major issue when merging data from multiple heterogeneous sources. Due to its complexity, duplicate elimination is beyond the scope of this lecture

# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - Data quality
  - **Exploratory Data Analysis (EDA)**
  - Data preprocessing
- Summary

# Exploratory Data Analysis (EDA)

- EDA — getting to know your data (through basic transformation and visualization)

- Assess data quality
- Basic sanity checks
- Get first insights into data
- Formulate new questions

}

No formal process with strict rules!



	<b>id</b>	<b>age</b>	<b>gender</b>	<b>height</b>	<b>weight</b>	<b>ap_hi</b>	<b>ap_lo</b>	<b>cholesterol</b>	<b>gluc</b>	<b>smoke</b>	<b>alco</b>	<b>active</b>	<b>cardio</b>
0	0	18393	2	168	62.0	110	80	1	1	0	0	1	0
1	1	20228	1	156	85.0	140	90	3	1	0	0	1	1
2	2	18857	1	165	64.0	130	70	3	1	0	0	0	1
3	3	17623	2	169	82.0	150	100	1	1	0	0	1	1
4	4	17474	1	156	56.0	100	60	1	1	0	0	0	0

## Running example:

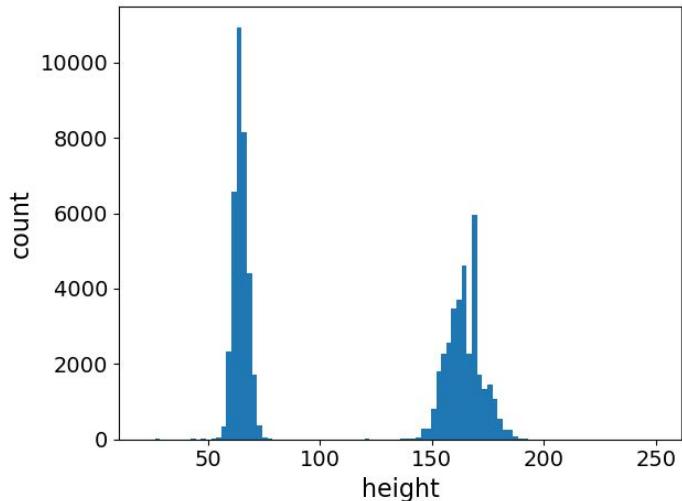
Cardiovascular Disease Dataset  
(modified to make some points)

# EDA — Identifying Noise

- Using histograms to inspect distribution of data values

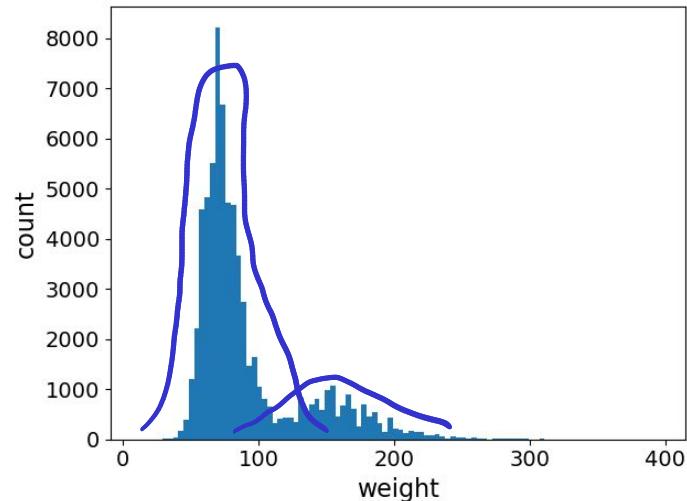
Noise in the height values

- 50% measured in inches
- 50% measured in centimeters



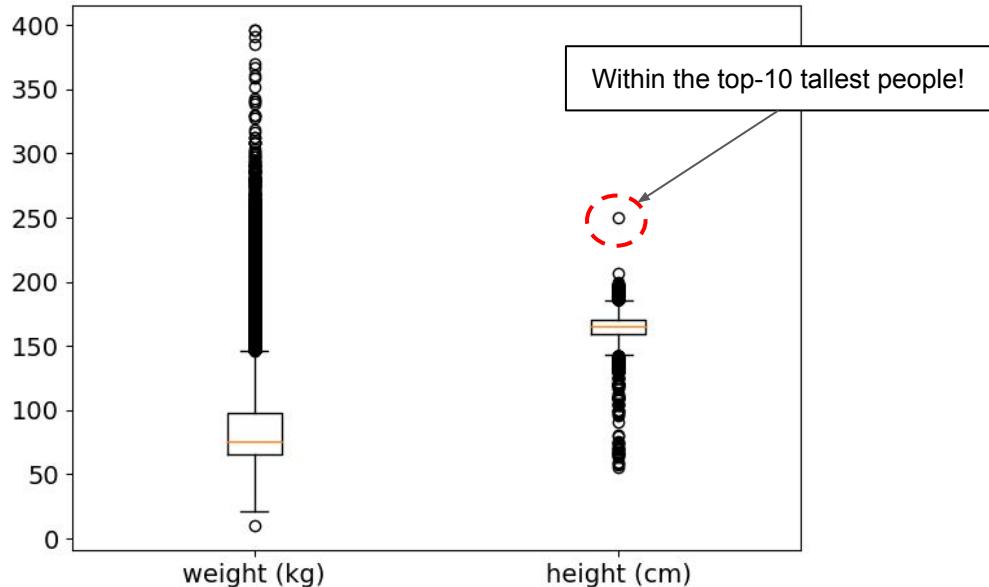
Noise in the weight values

- 80% measured in kilograms
- 20% measured in pounds



# EDA — Identifying Noise / Outliers

- Box plots to inspect distribution of attribute values
  - Make outliers explicit

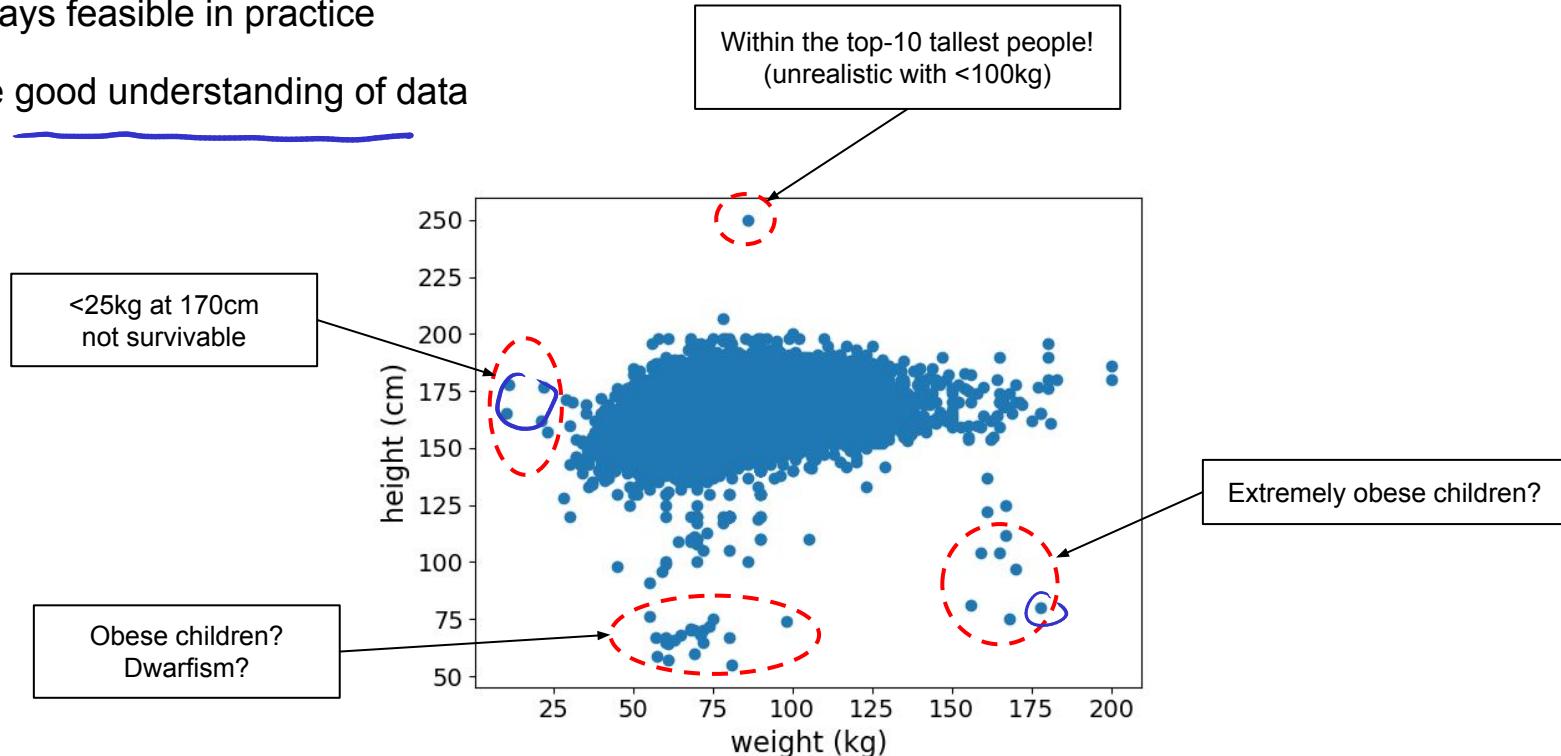


**Note:** Not all outliers are "bad" or considered noise. For example, a CEO's salary is typically much higher than the one of the average employee. Whether it should be removed depends on the goal of the analysis

# EDA — Identifying Noise / Outliers

- Using scatter plot to inspect correlations

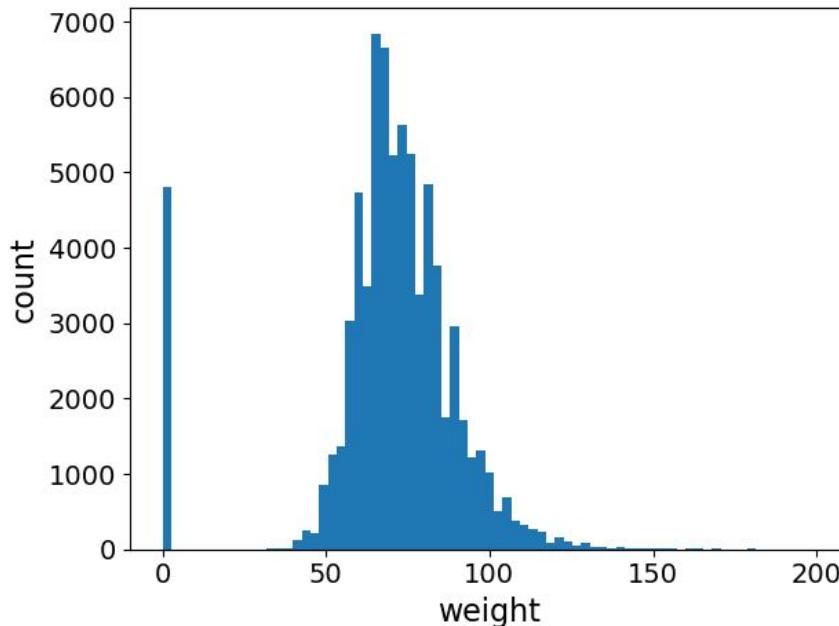
- Not always feasible in practice
- Require good understanding of data



# EDA — Missing Values

- Example: Default value (0) if people did not disclose weight
  - Can already negatively affect simple analysis such as calculating means/averages

NULL  
NIL  
NaN



# EDA — Attribute Types

		id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
0	0	18393	2	168	62.0	110	80		1	1	0	0	1	0
1	1	20228	1	156	85.0	140	90		3	1	0	0	1	1
2	2	18857	1	165	64.0	130	70		3	1	0	0	0	1
3	3	17623	2	169	82.0	150	100		1	1	0	0	1	1
4	4	17474	1	156	56.0	100	60		1	1	0	0	0	0

↑

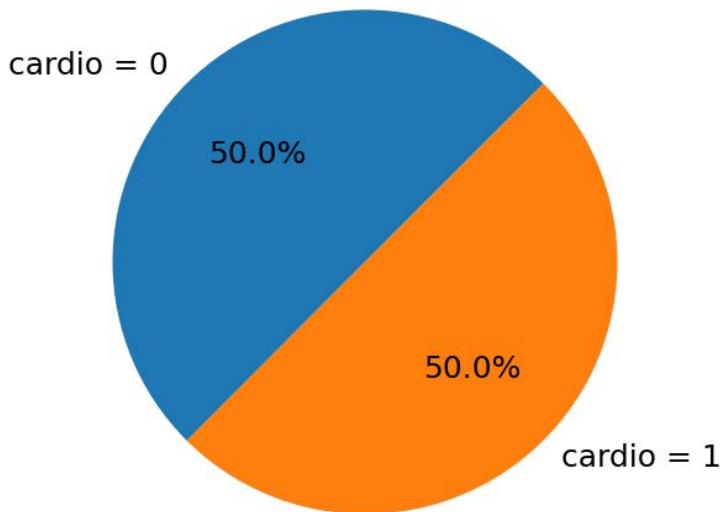
days

- Looks numerical but is categorical (ordinal)  
(1: normal, 2: above normal, 3: well above normal)
- Usually part of the documentation of dataset
- Interpretation requires good understanding of the data  
→ Generally impossible for automated methods



# EDA — Distribution of Class Labels

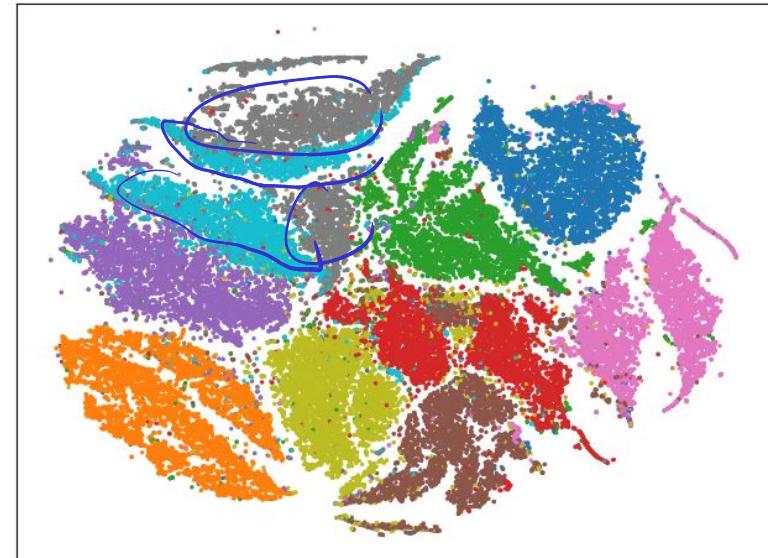
- Classification tasks generally benefit from balanced datasets
  - Balanced = all classes are (almost) equally represented
  - Distribution of classes also affects evaluation of found patterns



# EDA — Visualizing High-Dimensional Data

- Visualization using dimensionality reduction techniques (here: t-SNE)
- MNIST Dataset
  - 60k handwritten digits 0, 1, 2, ..., 9  
(~6k samples for class)
  - 28×28 pixels → 784 features  
(integer grayscale values 0..255)

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



# EDA — Unstructured Data (just some intuitions)

- Plain text
  - Language, (size of) vocabulary
  - Formal vs. informal text (e.g., social media content with slang, emoticons, emojis)
- Images/videos
  - Dimensions and resolutions
  - Color spaces
- Audio
  - Sampling rate and frequency range
  - Types of recording (e.g., voice vs. music)

# Quick Quiz

Which of the statements  
on the right is **True**?

**A**

Outliers are always noise and need  
to be removed before an analysis

**B**

As long as my class labels are  
balanced, I will get good results

**C**

Boxplots are often insufficient to  
identify all outliers in a dataset

**D**

If attribute values show a weird  
distribution, I know something is off

↙

can

# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - **Data preprocessing**
- Summary

# Data Preprocessing

- Main purposes
  - Improve data quality ("*Garbage in, garbage out!*")
  - Generate valid input for data mining algorithms
  - Remove complexity from data to ease analysis
- Core preprocessing task
  - Data cleaning
  - Data reduction
  - Data transformation
  - Data discretization

# Data Cleaning

- Improve data quality

- Remove or fill missing values
- Identify and remove outliers  
(if outliers are not the goal of the analysis)
- Identify and remove/merge duplicates  
*exact*
- Correct errors and inconsistencies  
(e.g., convert inches to centimeters)



Non-trivial tasks and typically  
very application-specific

# Data Reduction

- Reducing the number of data points
  - Sampling — select subset of data points (typically random or stratified sampling)
  - Commonly used for preliminary analysis or when the data size is extremely large
- Reducing the number of attributes
  - Removing irrelevant attributes (e.g., ids or ethically questionable attributes such as religion, sexual orientation, etc.)
  - Dimensionality reduction — mapping the data into a lower-dimensional space (PCA, LDA, t-SNE, etc.)
- Reducing the number of attribute values (form of noise removal)
  - Aggregation or generalization
  - Binning with smoothing

# Reducing the Number of Attribute Values — Examples

## • Aggregation

- Moving up concept hierarchy of numerical attributes (e.g., from days to years)
- Generalization for categorical attributes

	id	age	gender	height
0	0	18393	2	168
1	1	20228	1	156
2	2	18857	1	165
3	3	17623	2	169
4	4	17474	1	156



	id	age	gender	height
0	0	50.0	2	168
1	1	55.0	1	156
2	2	51.0	1	165
3	3	48.0	2	169
4	4	47.0	1	156

## • Binning and smoothing

- Sort by attribute value (e.g., height)
- Split data into bins of equal sizes
- Replace each value with bin mean  
(the means are also rounded in this example)

55 57 59 60 64 65 65 66 67 67 67 68 68 70 70 70 ...

55 57 59 60 64 | 65 65 66 67 67 | 67 68 68 70 70 | 70 ...

59 59 59 59 59 | 66 66 66 66 66 | 69 69 69 69 69 | 72 ...

# Data Transformation

- Some data reduction techniques also transform the data
  - Dimensionality reduction, aggregation/generalization, binning, etc.
- Attribute construction
  - Add or replace attribute inferred from existing attributes
  - Example: weight, volume → density
- Normalization
  - Scaling attribute values to value into a specified range (e.g., [0,1])
  - Standardization: scaling by using mean and standard deviation

# Normalization — Examples

$[0, 1]$

Min-max normalization

$$x_i^{weight} = \frac{x_i^{weight} - \min(x^{weight})}{\max(x^{weight}) - \min(x^{weight})}$$

weight

62.0

85.0

64.0

82.0

56.0

weight

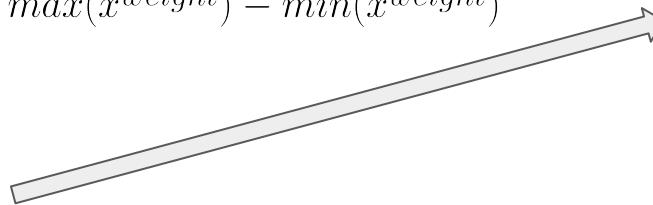
0.273684

0.394737

0.284211

0.378947

0.242105



weight

-0.847867

0.749826

-0.708937

0.541431

-1.264657

$$x_i^{weight} = \frac{x_i^{weight} - \mu^{weight}}{\sigma^{weight}}$$

Standardization

(z-score normalization)

# Data Discretization

- Converting continuous attributes into ordinal attributes
  - Some algorithms accept only categorical attributes
  - Convert a regression task to a classification task
- Example: Convert weight to a weight category
  - Many existing discretization methods
  - Here: discretization using 3 user-defined bins

	<b>id</b>	<b>age</b>	<b>gender</b>	<b>weight</b>
<b>0</b>	66283	14461	1	86.0
<b>1</b>	4780	14740	1	57.0
<b>2</b>	34457	21090	1	88.0
<b>3</b>	83116	15869	2	60.0
<b>4</b>	70356	20687	1	103.0



	<b>id</b>	<b>age</b>	<b>gender</b>	<b>weight</b>	<b>weight_bin</b>	<b>weight_class</b>
<b>0</b>	66283	14461	1	86.0	(70, 90]	average
<b>1</b>	4780	14740	1	57.0	(0, 70]	light
<b>2</b>	34457	21090	1	88.0	(70, 90]	average
<b>3</b>	83116	15869	2	60.0	(0, 70]	light
<b>4</b>	70356	20687	1	103.0	(90, 150]	heavy

# One-Hot Encoding

- Converting categorical attributes into numerical attributes
  - Converting categorical attributes into a series of binary attributes 0/1
  - Allows the application of any methods for numerical features on categorical attributes
- Example

	id	weight_class		id	weight_class	light	weight_class_average	weight_class_heavy
0	66598	light		0	66598	1	0	0
1	88878	average		1	88878	0	1	0
2	80363	heavy		2	80363	0	0	1
3	52546	light		3	52546	1	0	0
4	17715	average		4	17715	0	1	0

# Quick Quiz

Which attributes are generally(!)  
not relevant for the analysis and  
**SHOULD be removed?**

ID	Age	Education	Marital Status	Annual Income	Email	Credit Default
101	23	Masters	Single	75k	alice@...	Yes
102	35	Bachelor	Married	50k	bob@...	No
103	26	Masters	Single	70k	claire@...	Yes
104	41	PhD	Single	95k	dave@...	Yes
105	18	Bachelor	Single	40k	erin@...	No
106	24	Masters	Single	65k	fred@...	Yes

A ✓ ID + Email

B Age + Email

C ID + Education

D ID + Marital Status

# Quick Quiz — Side Note



ELSEVIER

Available online at [www.sciencedirect.com](http://www.sciencedirect.com)



ScienceDirect

Journal of Research in Personality 42 (2008) 1116–1122

---

JOURNAL OF  
RESEARCH IN  
PERSONALITY

---

[www.elsevier.com/locate/jrp](http://www.elsevier.com/locate/jrp)

Brief Report

## How extraverted is honey.bunny77@hotmail.de? Inferring personality from e-mail addresses

Mitja D. Back \*, Stefan C. Schmukle, Boris Egloff

*Department of Psychology, University of Leipzig, Seeburgstr. 14-20, 04103 Leipzig, Germany*

Available online 29 February 2008

# Quick Quiz

Which attributes are arguably not relevant or "problematic" and **SHOULD be removed?**

Age	Religion	Education	Has Account	Annual Income	Zodiac Sign	Credit Approval
23	Buddhist	Masters	Yes	75k	Leo	Yes
35	Buddhist	Bachelor	Yes	50k	Gemini	No
26	Muslim	Masters	Yes	70k	Libra	Yes
41	Christian	PhD	Yes	95k	Leo	Yes
18	Buddhist	Bachelor	Yes	40k	Virgo	No
24	Muslim	Masters	Yes	65k	Aries	Yes

A

Religion + Education + Zodiac Sign

B ✓

Religion + Zodiac Sign + Has Account

C

Religion + Zodiac Sign

D

Has Account + Zodiac Sign

# Quick Quiz — Side Note

CNA Insider

## Does a job seeker's horoscope matter? For some companies, the answer is yes

There are companies that turn to unconventional methods like astrology, tarot reading and numerology to help guide hiring decisions. What beliefs are these practices grounded in, and how legitimate are they?



# Outline

- Course Logistics
- Overview
  - What is Knowledge Discovery / Data Mining?
  - Common Data Mining tasks
  - Types of data & data representations
- Data preparation
  - Data quality
  - Exploratory Data Analysis (EDA)
  - Data preprocessing
- Summary

# Summary

- Course Logistics

- Core Concepts

- What is (not) Data Mining?
- Knowledge discovery process
- Overview to common tasks

- Data preparation

- Types of data and data quality
- Exploratory data analysis
- Data preprocessing

finding  
patterns

**Data → Knowledge**

**Know your data & clean your data!**

# Solutions to Quick Quizzes

- Slide 22: D
- Slide 37: D
- Slide 38: B (A also OK)
- Slide 39: A (in general)
- Slide 55: C
- Slide 65: A
- Slide 67: B (C also OK)

# CS5228: Knowledge Discovery and Data Mining

## Lecture 2 — Clustering I

# Course Logistics — Update

- Project — Team Formation

- Default team size: 4 (exception maybe later if needed)
- Final allocation at the teaching team's discretion
- Canvas self-signup group

- Project — Deadline

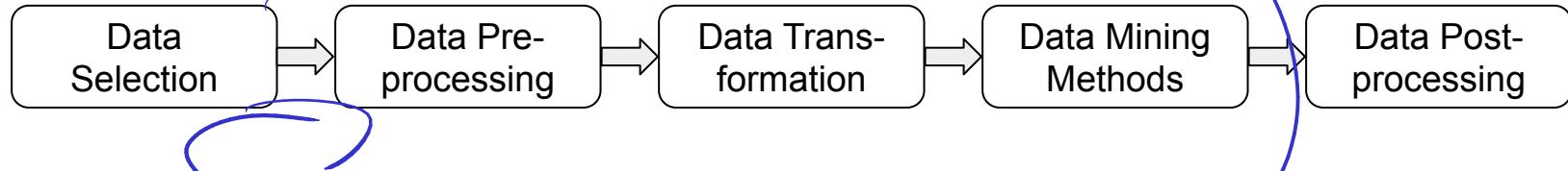
- Dataset release: Week ~3
- Progress Report: Week ~7
- Final Report: Week 11 (Thu, Oct 30)



Earlier submission are always welcome! :)

# Quick Recap

- Data Mining — from data to knowledge



- Nature of data

- Types of attributes: categorical (nominal / ordinal) vs. numerical (interval / ratio)
- Types of data and data representations  
(e.g., data matrix, transactions, graph, ordered data)
- Data quality

# Quick Recap — Data Preparation

- **Exploratory Data Analysis (EDA)**

- Assess data quality
- Get to know your data

- **Data Preprocessing**

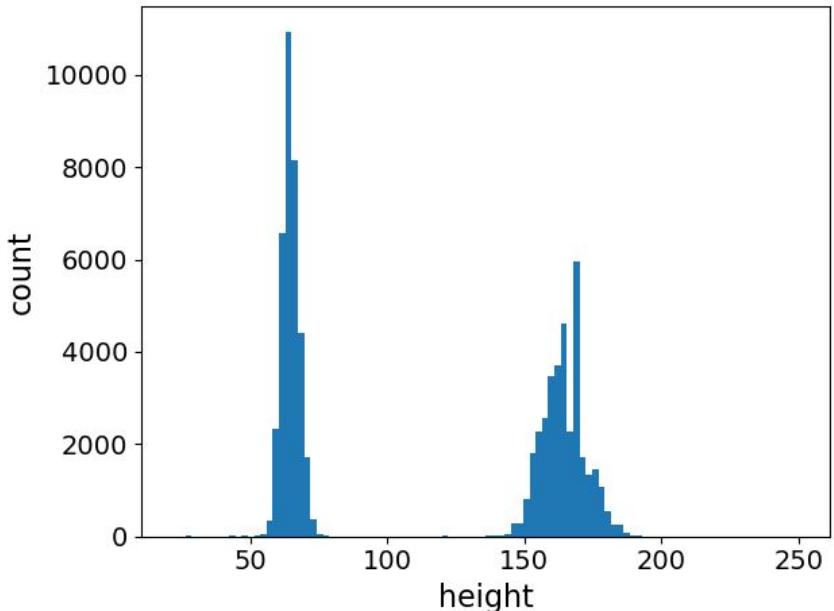
- Improve data quality ("*Garbage in, garbage out!*")
- Generate valid input for data mining algorithms
- Remove complexity from data to ease analysis

Example of noise: missing values

Age	Edu- cation	Marital Status	Annual Income	Credit Default
23	Masters	Single	75k	Yes
N/A	Bachelor	Married	N/A	No
26	Masters	Single	70k	Yes
41	PhD	Single	95k	Yes
18	Bachelor	Single	40k	No
55	Master	Married	N/A	No
30	Bachelor	Single	N/A	No
35	PhD	Married	60k	Yes
N/A	PhD	Married	65k	Yes

# Quick Recap — Clarifications

- Suspicious EDA results  $\Rightarrow$  errors in the data



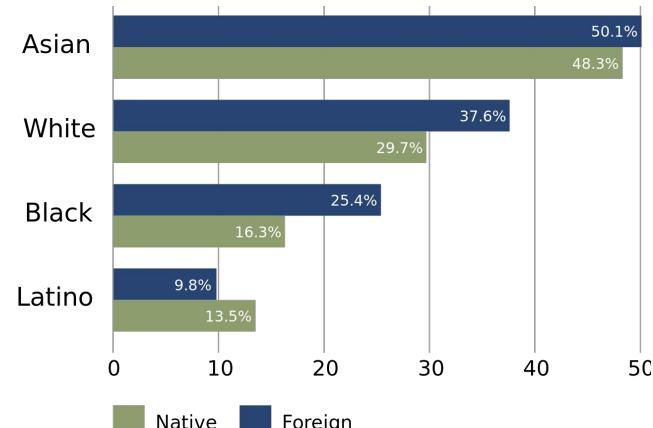
Data may be absolutely correct

Example: infant care dataset (infants + parents)

# Quick Recap — Clarifications

- Removing "questionable" attributes  $\Rightarrow$  better results
  - e.g., removing zodiac sign from credit default prediction might lower accuracy
- Removing "questionable" attributes  $\Rightarrow$  no biases (or perfect privacy)
  - e.g., removing ethnicity not foolproof if it correlates with other attributes

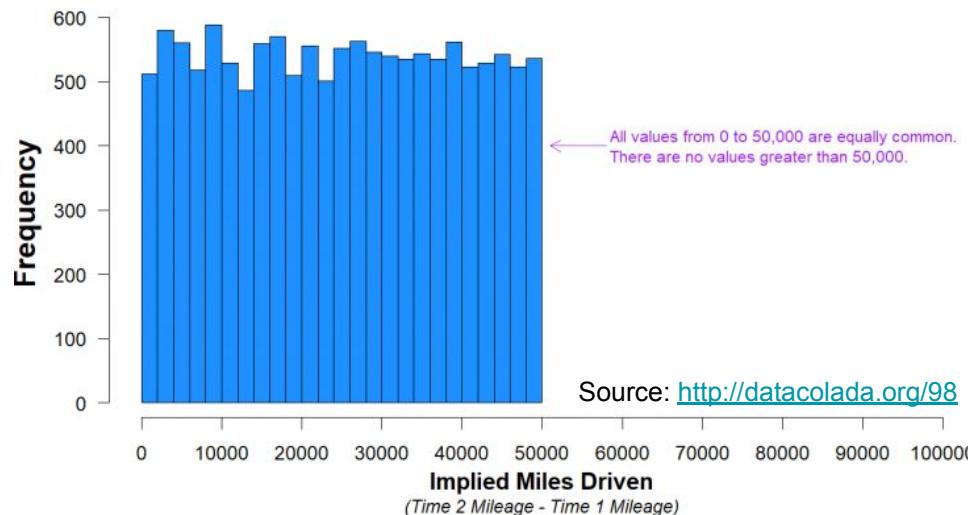
Age	Ethnicity	Education	Annual Income	Credit Approval
23	White	Masters	75k	Yes
35	Buddhist	Bachelor	50k	No
26	Asian	Masters	70k	Yes
41	Asian	PhD	95k	Yes
18	Black	Bachelor	40k	No
...	...	...	...	...



# Quick Recap — EDA: Additional Insights

- Manipulated / fudged data
  - Sometimes data just does not "look right"
  - Example: unexpected/unintuitive data distributions

Figure 1. Histogram of Miles Driven - Car #1 (N=13,488)



# Outline

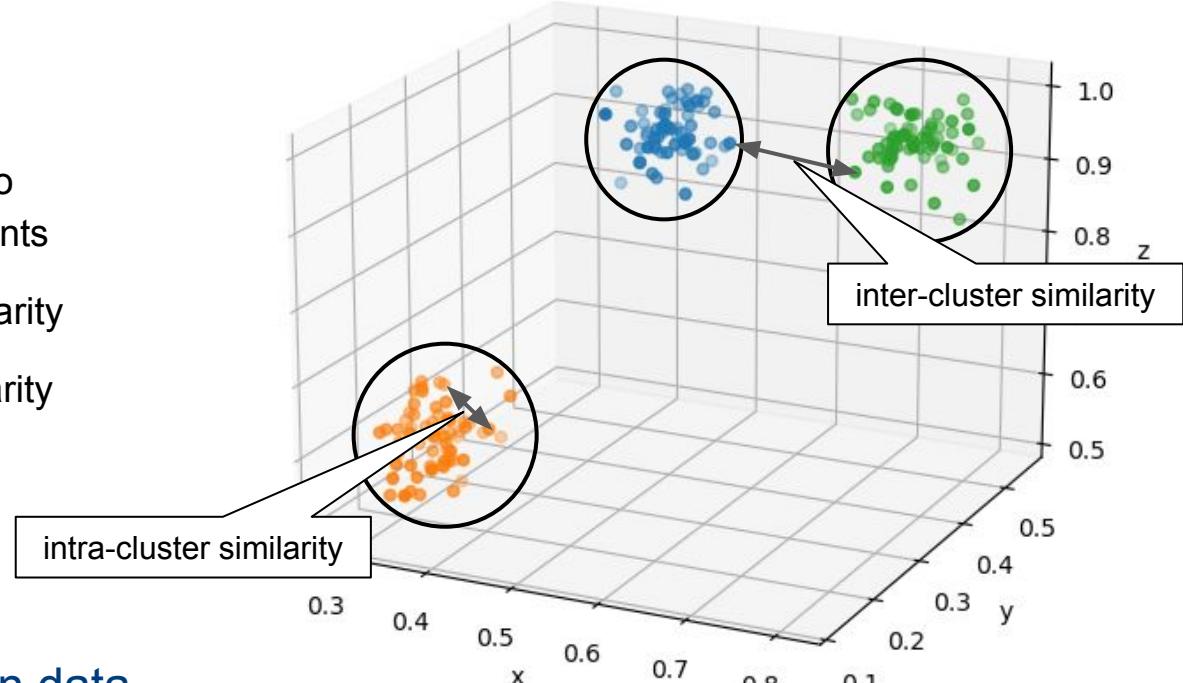
- **Clustering**
  - Overview
  - Applications
  - Concepts
- Clustering algorithms
  - K-Means
  - DBSCAN
  - Hierarchical Clustering
- Cluster Evaluation

L2

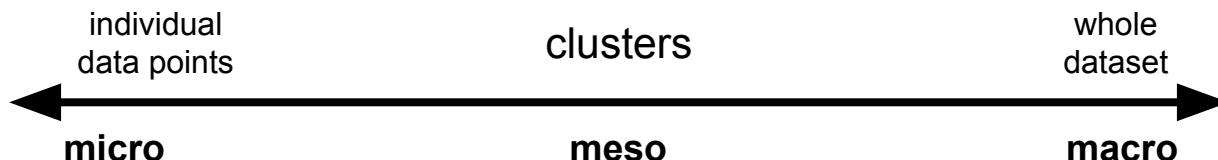
L3

# What is Clustering?

- Goal of Clustering
  - Separate **unlabeled** data into groups of **similar** objects/points
  - Maximize **intra-cluster** similarity
  - Minimize **inter-cluster** similarity



- Meso-level perspective on data



# Applications

- Market segmentation
  - Group customers based on behavior and/or preferences
  - Push tailored promotions to all customers in cluster
- Recommender systems
  - Group items (e.g., movies) based on their attributes (e.g., genre, length, budget)
  - Recommend movies from a cluster with movies a user liked
- Web Search Diversification
  - Group Web pages (e.g., news articles) based on content, source (type), etc.
  - Return search results from different clusters to ensure diversity
- ...and many more applications
- EDA

# Ingredients for Clustering

- Representation of objects, e.g.:
  - (Multidimensional) point coordinates  $x, y$
  - Sets  $A, B$  (e.g., items in a transaction)
  - Vectors  $u, v$  (e.g., TF-IDF)
- Similarity Measure, e.g.:
  - Euclidean Distance
  - Jaccard Similarity
  - Cosine Similarity
- Clustering Algorithm
  - Process that determines if an object belongs to a cluster

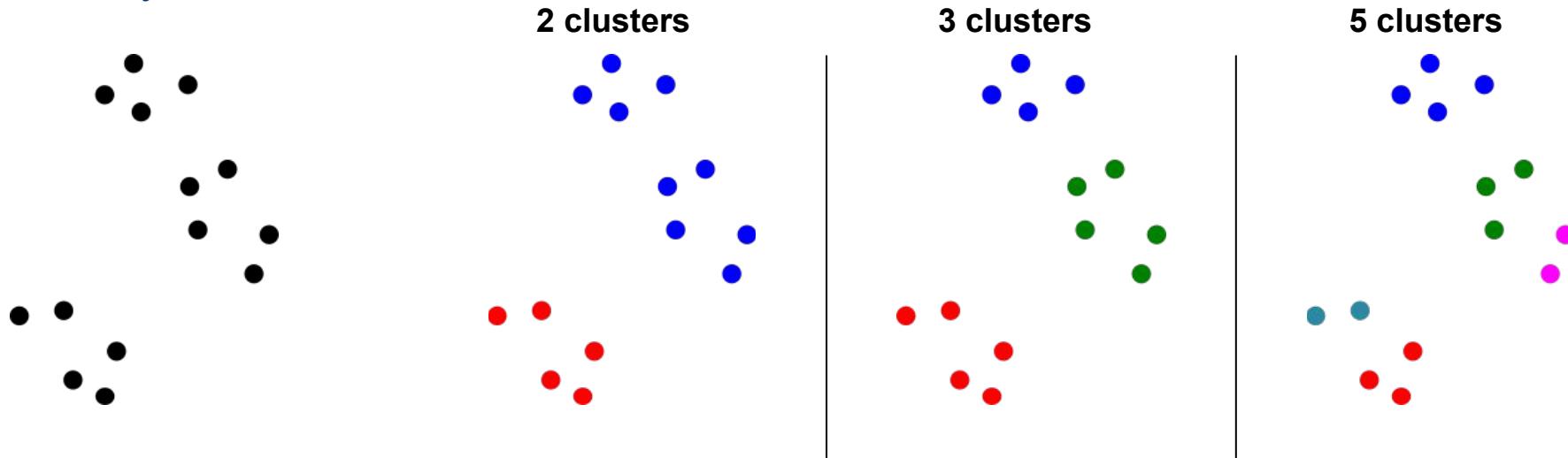
$$dist_{euclidean}(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

$$sim_{jaccard}(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

$$sim_{cosine}(u, v) = \frac{u \cdot v}{\|u\| \|v\|}$$

# What makes a clustering "good"?

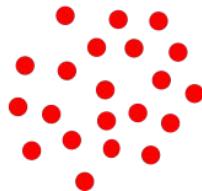
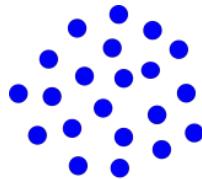
How many clusters?



→ Deciding on a good / meaningful / useful set of clusters not obvious

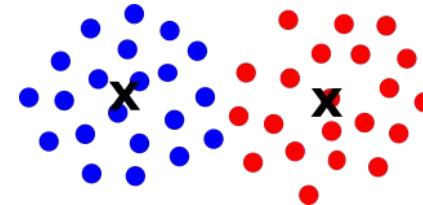
# Types of Clusters

- Well-separated



- Any object in a cluster is closer to every other object in the cluster than to any point outside the cluster

- Center-based

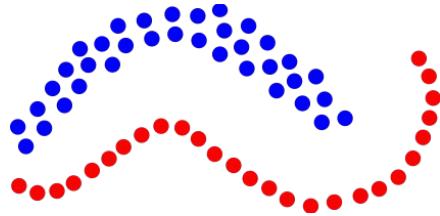


- Any object in a cluster is closer to the "center" of a cluster than to the center of any other cluster
- Example: mean of all data points (in Euclidean space)
- Cluster center commonly called **centroid**

K-Means

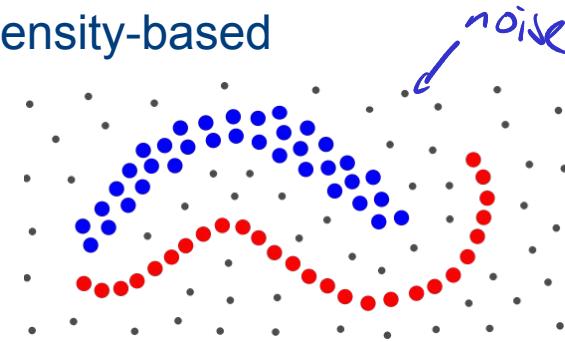
# Types of Clusters

- Contiguity-based



- 2 objects are in a cluster if they are more similar than a specified threshold
- Each object is more similar to some object in that cluster than to any point in a different cluster

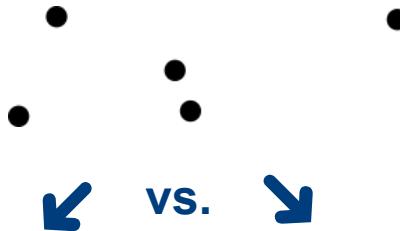
- Density-based



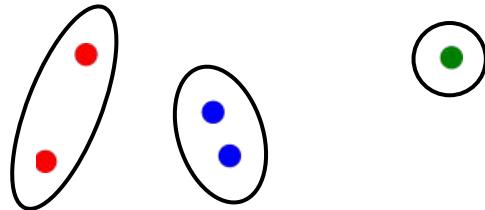
- Cluster = dense( $r$ ) region of objects surrounded by region of low(er) density
- Can typically address noise better than contiguity-based clusters

DBSCAN

# Types of Clusterings (i.e., sets of clusters)

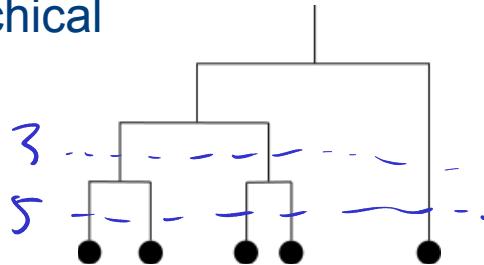


- Partitional



- Division of the set of data objects into non-overlapping subsets (i.e., clusters)
- Each object is in exactly 1 cluster (or in no cluster at all)

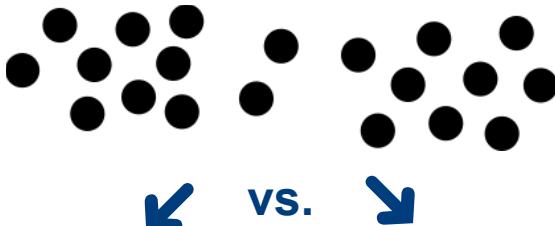
- Hierarchical



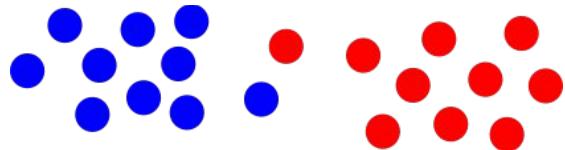
- Clusters can be nested
- A point can belong to different clusters depending on the hierarchy level

AleNes

# Types of Clusterings (i.e., sets of clusters)

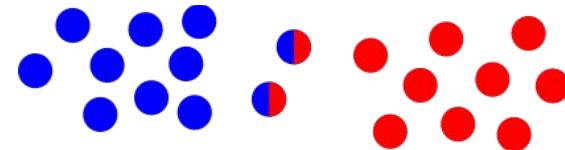


- Exclusive



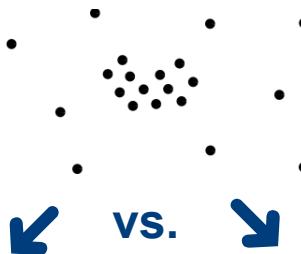
- Each object belongs to 1 cluster

- Non-exclusive / overlapping

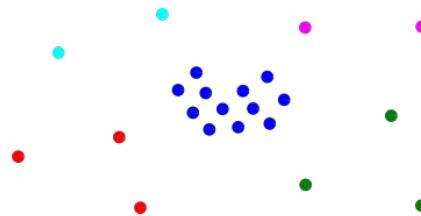


- An object can belong to more than 1 cluster at a time
- Fuzzy clustering: each object belongs to all clusters with a certain probability

# Types of Clusterings (i.e., sets of clusters)



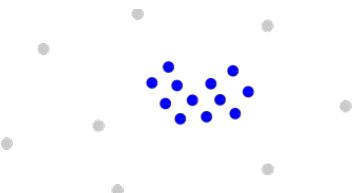
- Complete



- Every object is assigned to (at least) 1 cluster

*k-Means, AC UES*

- Partial



- An object might not belong to a cluster
- Examples: noise, outliers

*DBSCAN*

# Quick Quiz

In what situation can I **NOT** apply clustering on a dataset?

**A**

All attributes of my dataset are nominal attributes

**B**

My dataset is 1-dimensional, i.e., there is only one attribute

**C**

The values of the attributes are not normally distributed

**D**



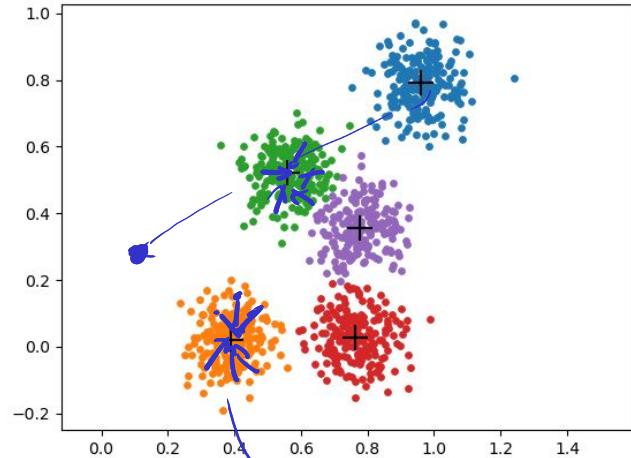
There is no similarity or distance between the data points defined

# Outline

- Clustering
  - Overview
  - Applications
  - Concepts
- Clustering Algorithms
  - K-Means
  - DBSCAN
  - Hierarchical Clustering
- Cluster Evaluation

# K-Means

- Basic characteristics
  - Clusters: centroid-based
  - Clustering: partitional, exclusive, complete
- Inputs (for d-dimensional Euclidean space)
  - $(x_1, x_2, \dots, x_N)$ ,  $x_i \in R^d$
  - Number of clusters K  $\rightarrow c_1, c_2, \dots, c_K$  cluster centers
- Optimization objective
  - Minimize Sum of Squared Error
  - Finding optimal solution is NP-hard  
 $O(N^{Kd+1})$
- Greedy solutions



$$SSE = \sum_{i=1}^K \sum_{x \in C_i} \overbrace{\|x - c_i\|^2}^{\text{squared distance between data point and center}}$$

set of points in i-th cluster

# K-Means — How to Define the Centroid of a Cluster?

- Simple case in Euclidean space  $SSE = \sum_{i=1}^K \sum_{x \in C_i} \|x - c_i\|^2$

$$\frac{\delta}{\delta c_k} SSE = \frac{\delta}{\delta c_k} \cancel{\sum_{i=1}^K \sum_{x \in C_i}} (x - c_i)^2$$

$$= \cancel{\sum_{i=1}^K \sum_{x \in C_i}} \frac{\delta}{\delta c_k} (x - c_k)^2$$

$$\Rightarrow \sum_{x \in C_k} 2 \cdot (x - c_k) \stackrel{!}{=} 0$$

# K-Means — How to Define the Centroid of a Cluster?

- Simple case in Euclidean space  $SSE = \sum_{i=1}^K \sum_{x \in C_i} \|x - c_i\|^2$

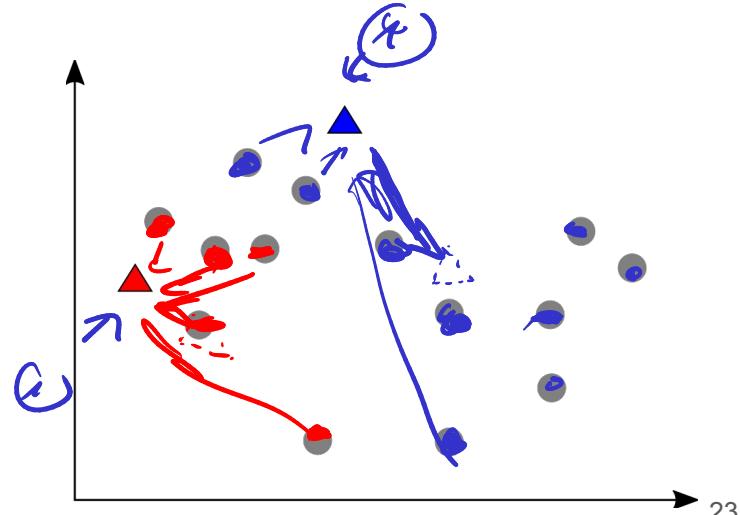
$$\begin{aligned}\frac{\delta}{\delta c_k} SSE &= \frac{\delta}{\delta c_k} \sum_{i=1}^K \sum_{x \in C_i} (x - c_i)^2 \\ &= \sum_{i=1}^K \sum_{x \in C_i} \frac{\delta}{\delta c_k} (x - c_i)^2 \\ \Rightarrow \sum_{x \in C_k} 2 \cdot (x - c_k) \stackrel{!}{=} 0 &\quad \sum_{x \in C_k} 2 \cdot (x - c_k) \stackrel{!}{=} 0 \\ &\Rightarrow \sum_{x \in C_k} x - \sum_{x \in C_k} c_k = 0 \\ &\Rightarrow m_k c_k = \sum_{x \in C_k} x \\ &\Rightarrow c_k = \underbrace{\frac{1}{m_k} \sum_{x \in C_k} x}_{\text{Centroid of cluster}}\end{aligned}$$

→ Centroid of cluster = Mean of all points in that cluster

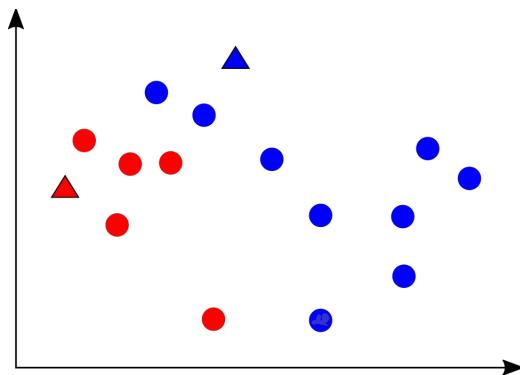
# K-Means — Basic Algorithm (Lloyd's Algorithm)

1. Initialization: Select K points as initial centroids  $c_1, c_2, \dots, c_K$
  2. Repeat
    - 2a) Assignment: assign each point to nearest cluster (i.e., centroid)
    - 2b) Update: move each centroid to the average of its assigned points
- Until no change in assignments

Example: K=2, after initialization

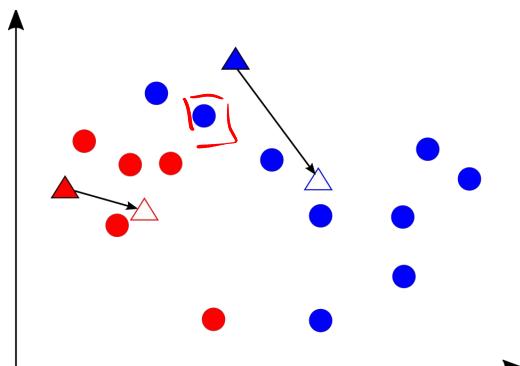


# K-Means — Repeated Steps



- Assignment

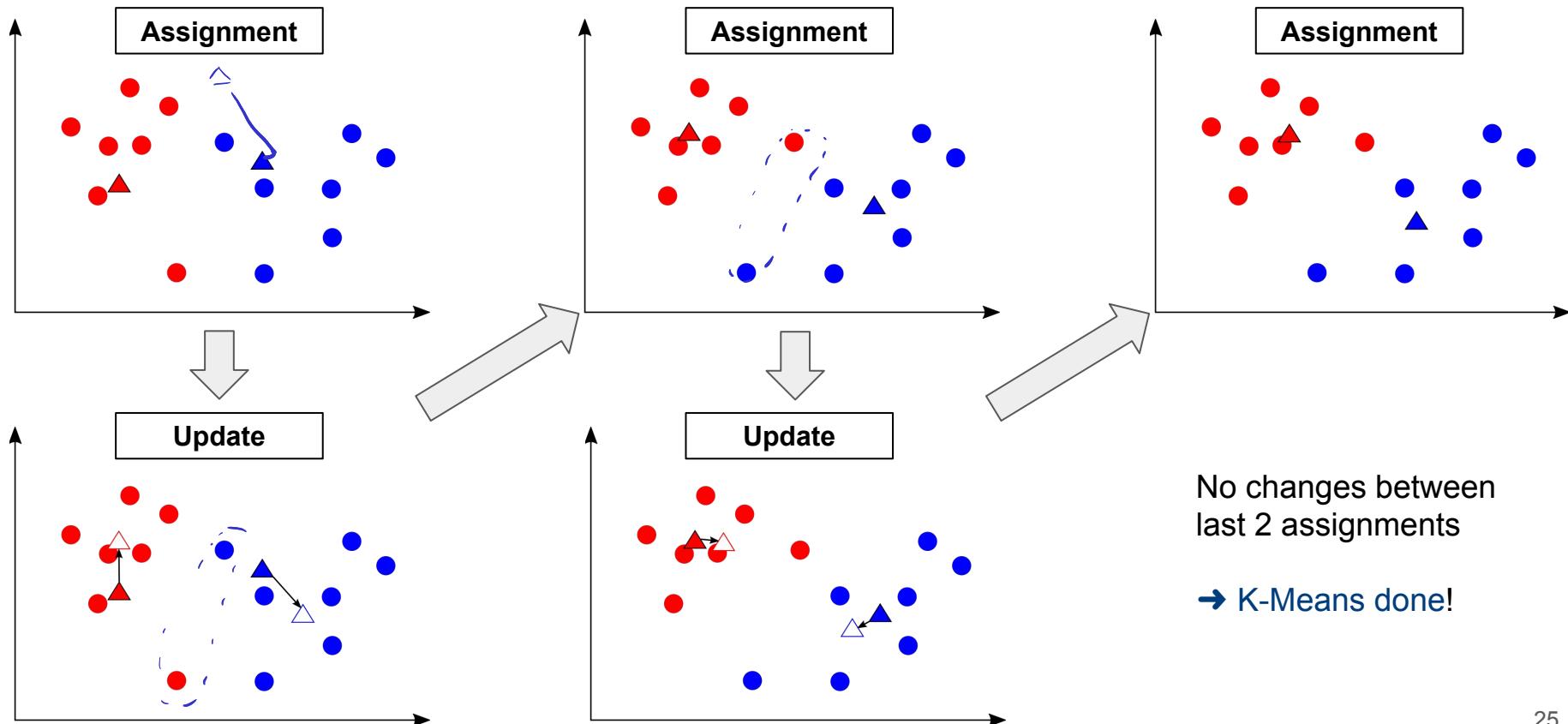
- For each data point  $x$ , find nearest centroid  $c_i$
- Assign  $x$  to cluster  $i$



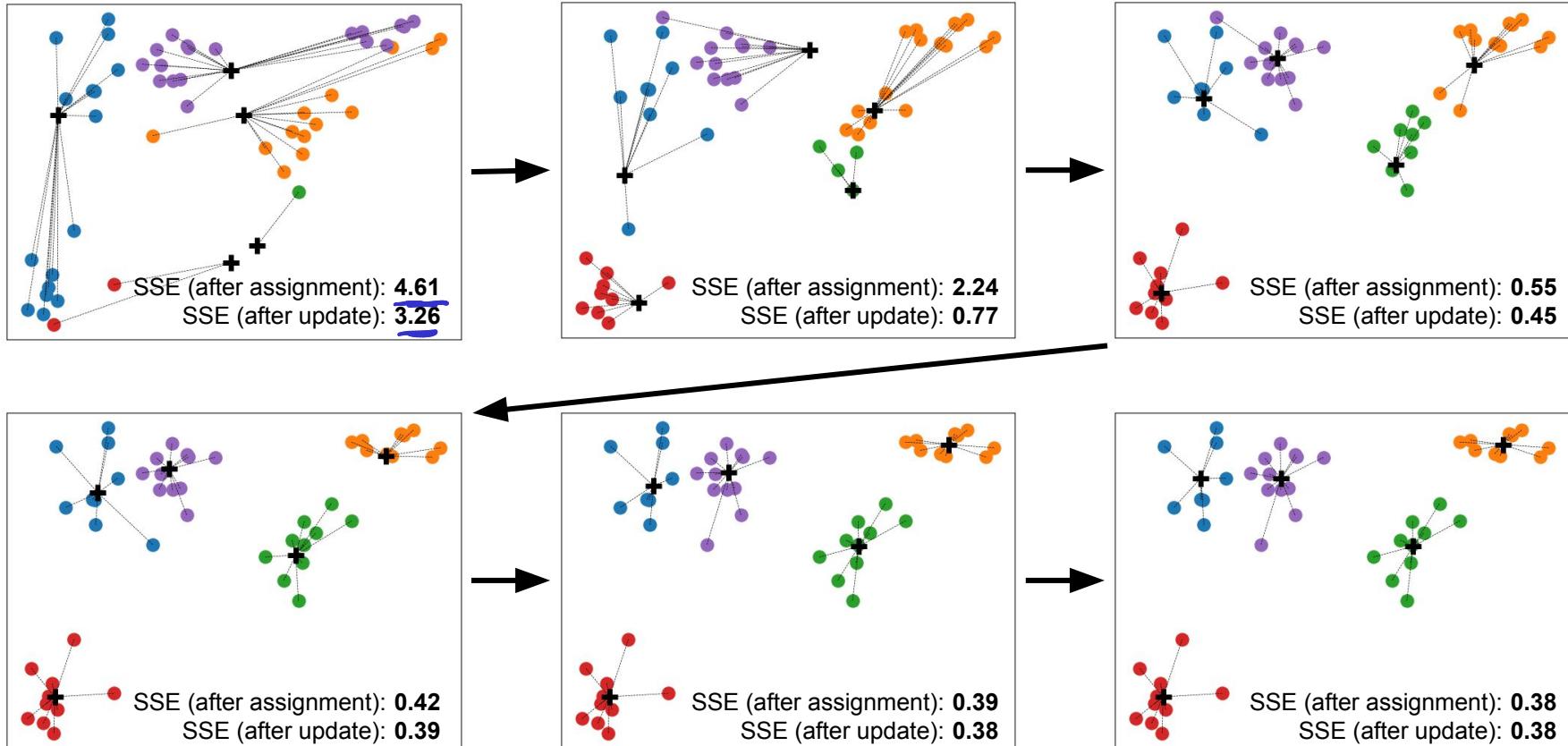
- Update

- Calculate mean of all data points of cluster  $i$
- Set centroid  $c_i$  to mean of cluster  $i$

# K-Means — Iteration until Convergence

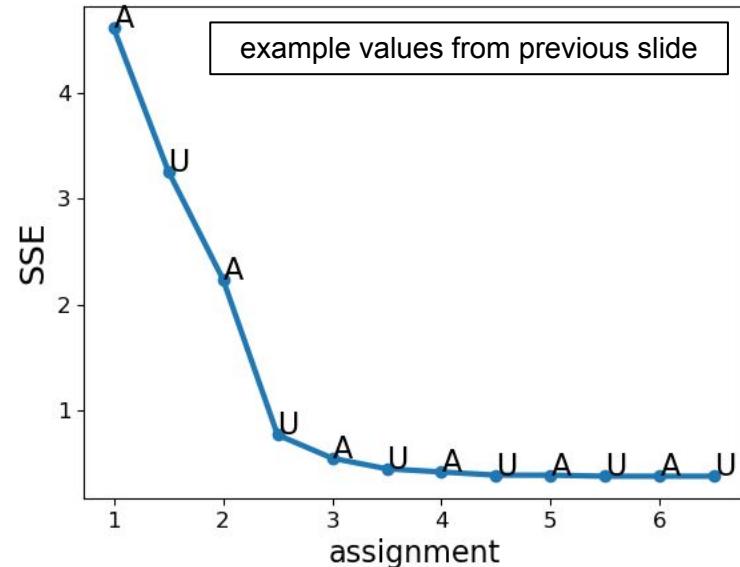


# K-Means — Convergence



# K-Means — Convergence

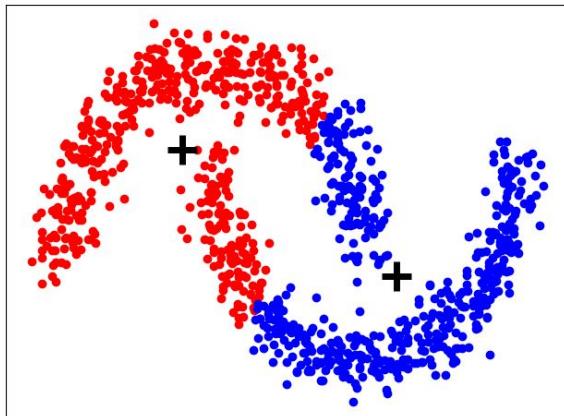
- Good news: K-Means always converges!
  - Both assignment (A) and update (U) reduce SSE (or no changes)
  - Most improvement during the first iterations
- Bad news
  - Lloyd's algorithm returns a **local optimum**, not necessarily a global optimum
  - Important: initialization of centroids  
(discussed in more detail later)



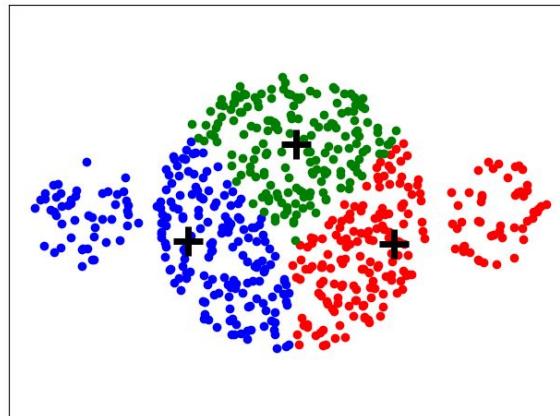
# K-Means — Limitations (Data Distribution)

- K-Means is susceptible to "natural" clusters

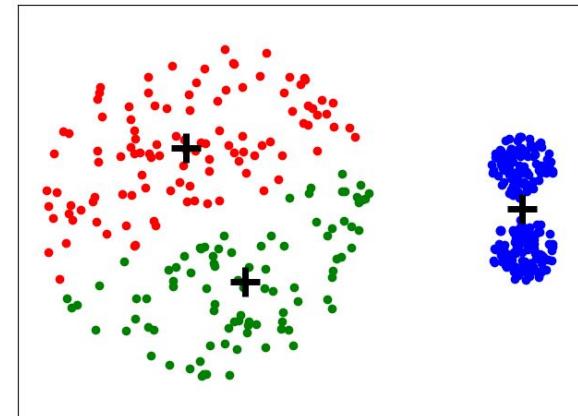
Non-Spherical Clusters



Clusters of different sizes

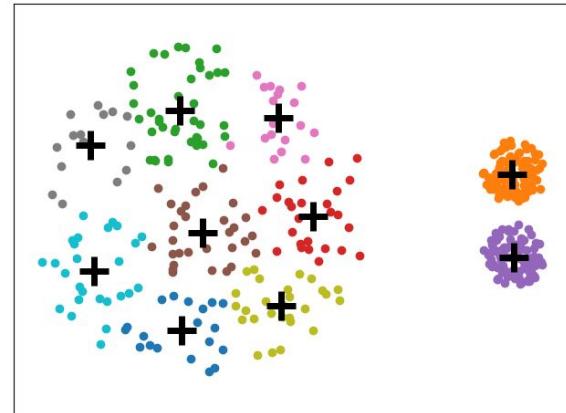
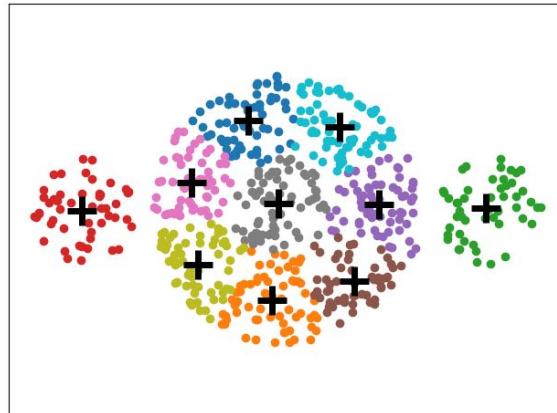
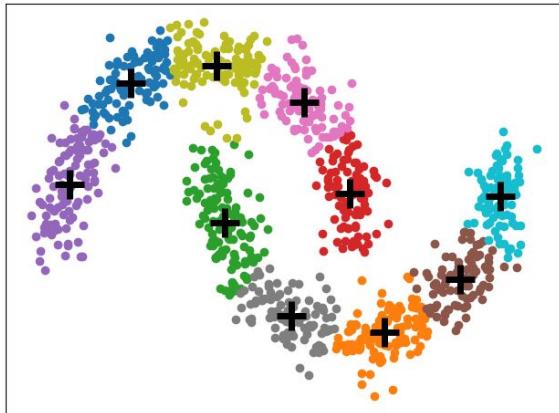


Clusters of different densities



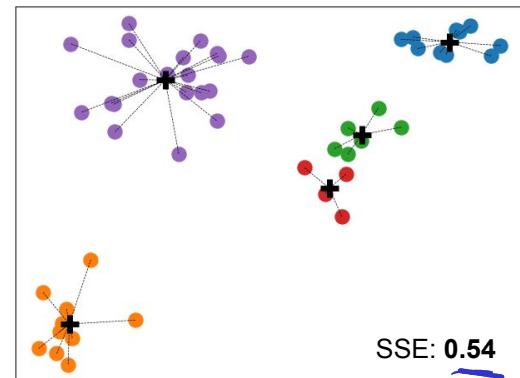
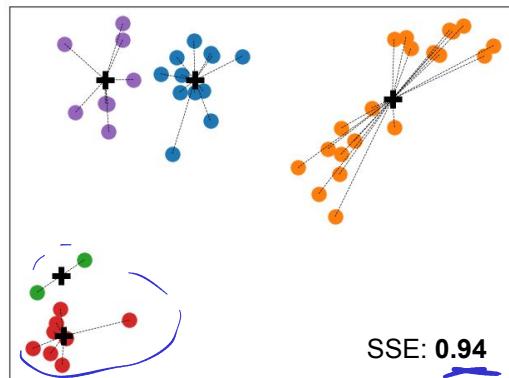
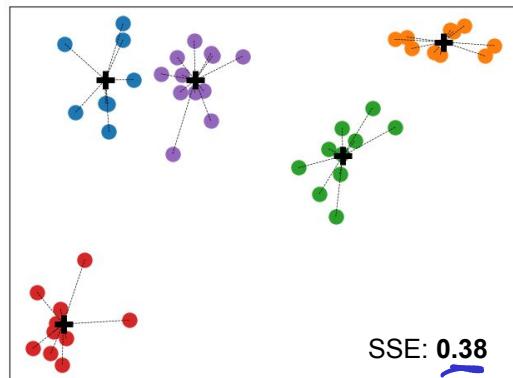
# K-Means — Limitations (Data Distribution)

- Potential workaround: Choose large(r) value for K
  - Intuition: split natural clusters into multiple "well-behaved" (blob-like) subclusters
  - Apply suitable postprocessing steps to merge subclusters



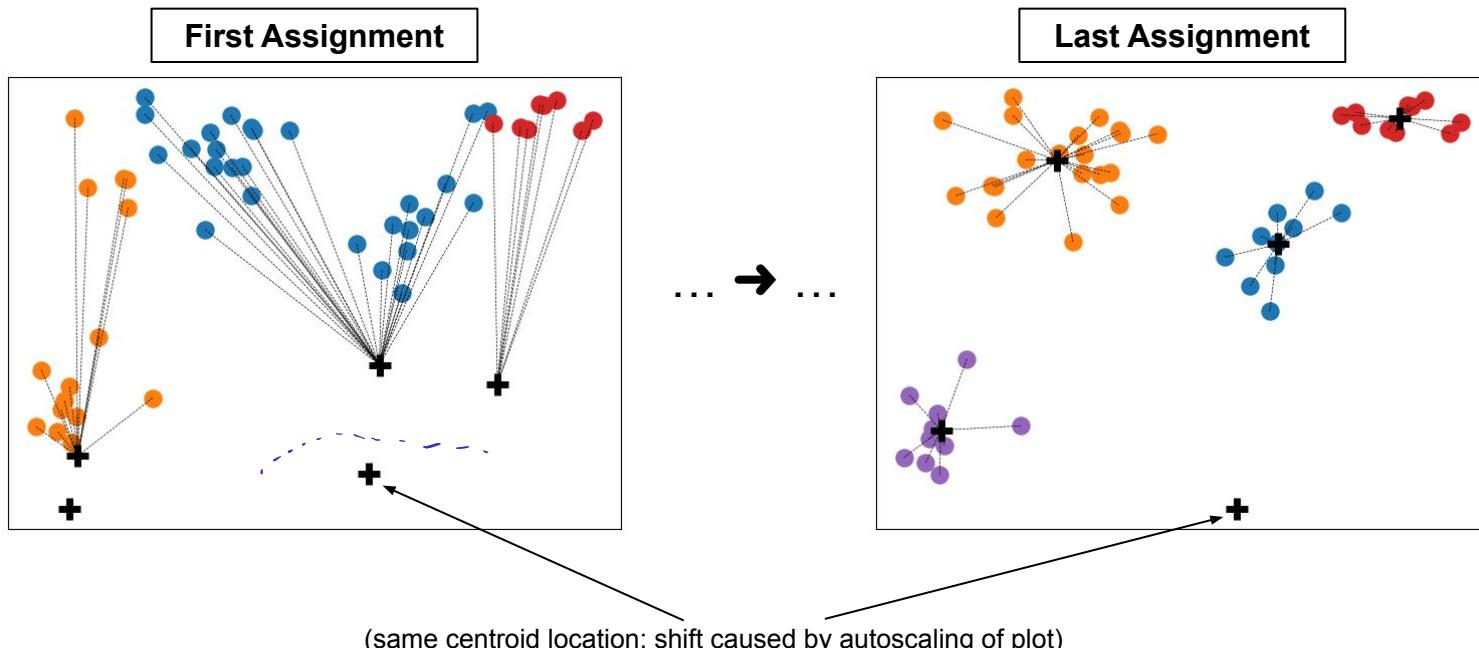
# K-Means — Limitations (Initial Centroids Issue)

- Different initializations of centroids may yield different clusterings
  - Different clusterings typically have different SSEs → global minimum!



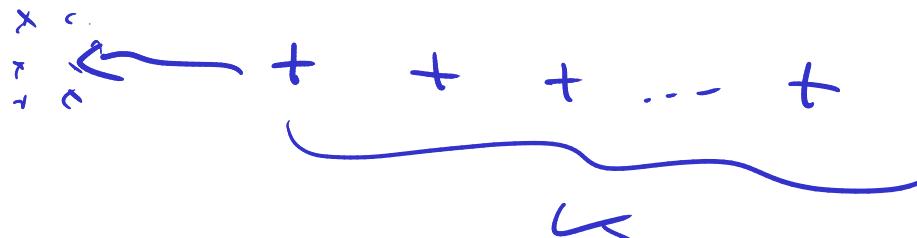
# K-Means — Limitations (Initial Centroids Issue)

- Some initialization of centroids can yield empty clusters
  - Occurs when a centroid is "blocked off" data points by other centroids



# Quick Quiz

What is the **maximum number of empty clusters** with K-Means with a really bad initialization of the centroids and  $K \geq 2$ ?



A

0

B

$K - 1$

C

$K$

D

$K + 1$

# K-Means — Handling Empty Clusters

- Artificially fill empty clusters after assignment step (and continue iterations)
  - Replace empty cluster with point that contributes most to SSE
  - Replace empty cluster with a point from the cluster with the highest SSE
- Post processing
  - Split "loose" clusters = clusters with very high SSE
- Modification of Lloyd's algorithm → K-Means variants
  - Typically aim to address the initial centroids issue

# K-Means Variants — K-Means++

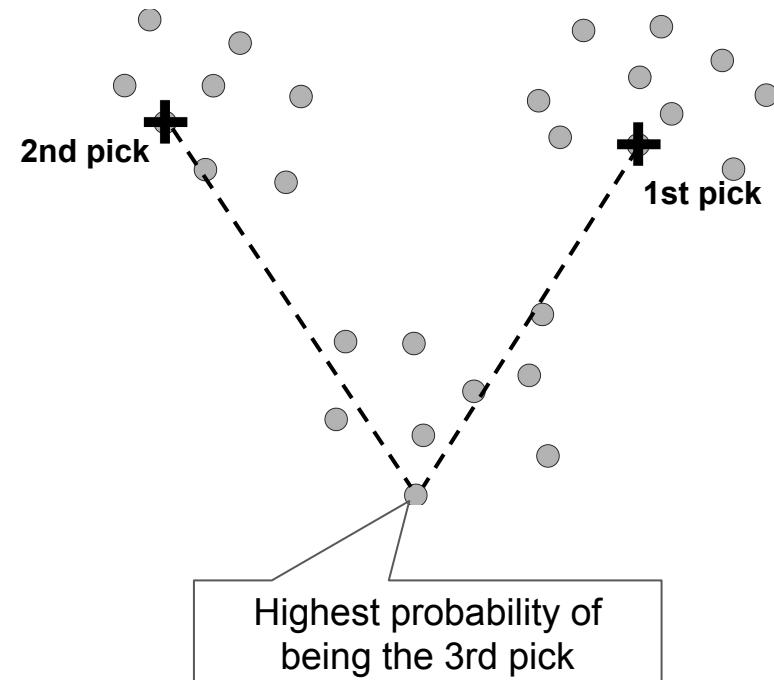
- Only changes initialization of centroids

(assignment/update steps remain the same)

- Goal: spread out centroids
- Better performance in practice
- Theoretical guarantees

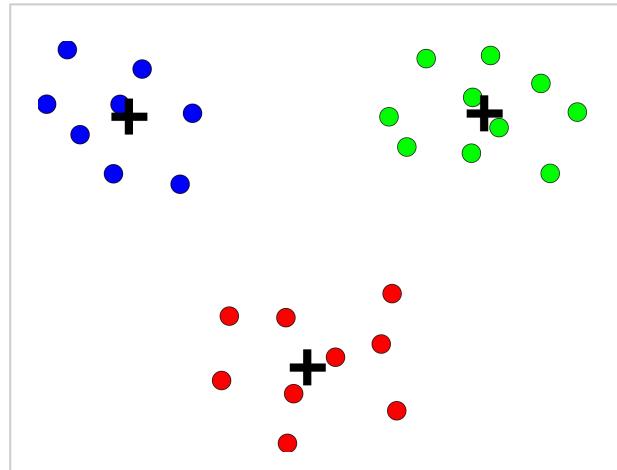
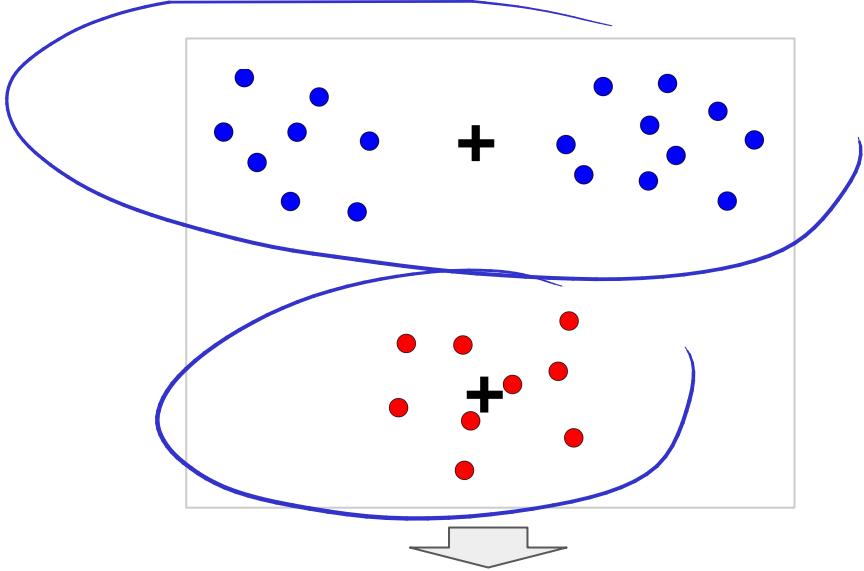
## Initialization process

- Pick random point as first centroid  $c_1$
  - Repeat
    - For each point  $x$ , calculate distance  $d_x$  to nearest existing centroid
    - Pick random point for next centroid with probability proportional to  $d_x^2$
- Until K centroids have been picked



# K-Means Variants — X-Means

- Automatic method to choose K
  - Run K-Means with K=2
  - Iteratively, run K-Means with K=2 over each subcluster
  - Split subcluster only if meaningful w.r.t. a scoring function
- Example scoring functions
  - Bayesian Information criterion (BIC)
  - Akaike information criterion (AIC)
  - Minimum Description Length (MDL)



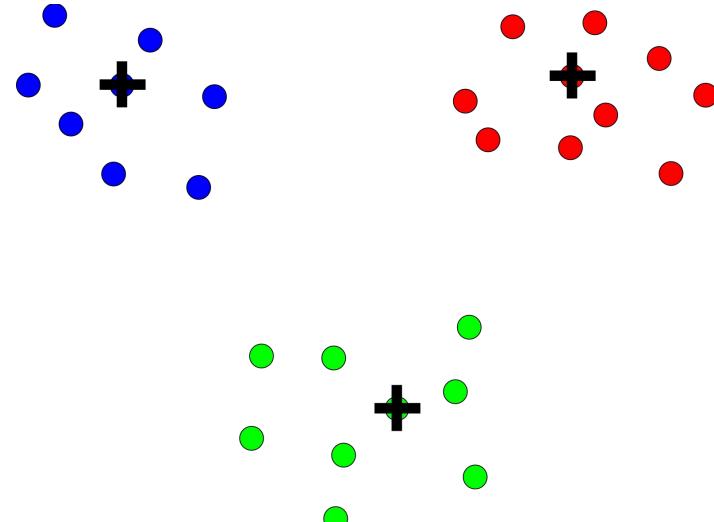
# K-Means Variants — K-Medoids

- **Restriction:** centroids are chosen from the data points

- Does not require the calculation of the averages  
(no average of sets or more complex objects)
- Only notion of distance or similarity still needed  
(e.g., Jaccard similarity for sets or custom metrics)
- More robust to noise and outliers

- **Main issue:** performance

- More expensive Update step
- Swap medoid with each point in cluster and calculate change in cost (e.g., SSE)
- Choose the point as new medoid that minimizes the cost after swapping



# Quick Quiz

Can a theoretically optimal initialization of centroids **guarantee** no empty clusters in any case of running K-Means?

$$n = \text{# points}$$

$$k > n$$

A  No

B  Yes

C  Impossible to say

D  Unlikely

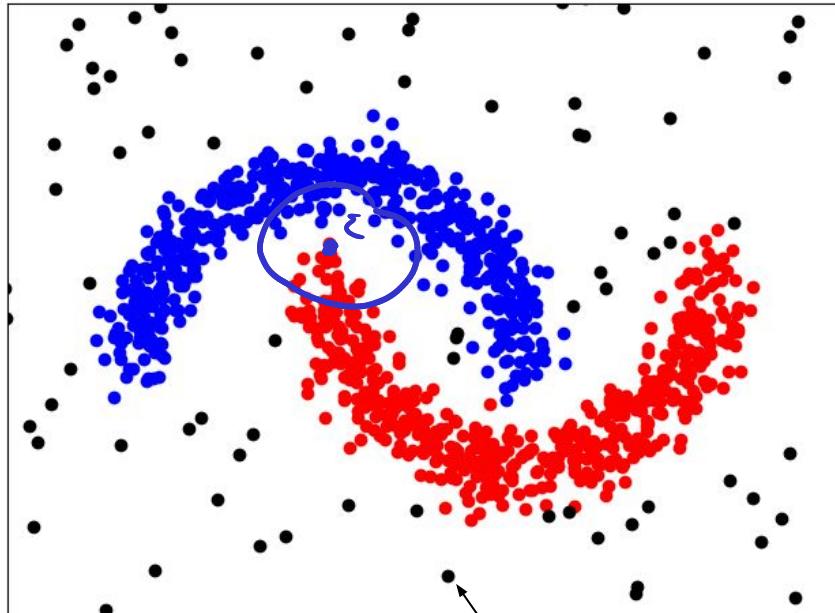
# Outline

- Clustering
  - Overview
  - Applications
  - Concepts
- Clustering Algorithms
  - K-Means
  - **DBSCAN**
  - Hierarchical Clustering
- Cluster Evaluation

# DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

- Basic characteristics
  - Clusters: density-based
  - Clustering: partitional, exclusive, partial
- Inputs (for d-dimensional Euclidean space)
  - $(x_1, x_2, \dots, x_N)$ ,  $x_i \in R^d$
  - $\varepsilon$  — radius defining a points neighborhood
  - $MinPts$  — minimum number of points

$$density = \frac{mass}{volume} = \frac{MinPts}{\varepsilon}$$



noise: not part  
of any cluster

# DBSCAN — Types of Points

inclus. the point } ideal

- **Core points** ●

- All points with at least  $MinPts$  neighbors with radius  $\mathcal{E}$  (this includes the point itself!)
- Form the **interior** of a cluster

- **Border Points** ○

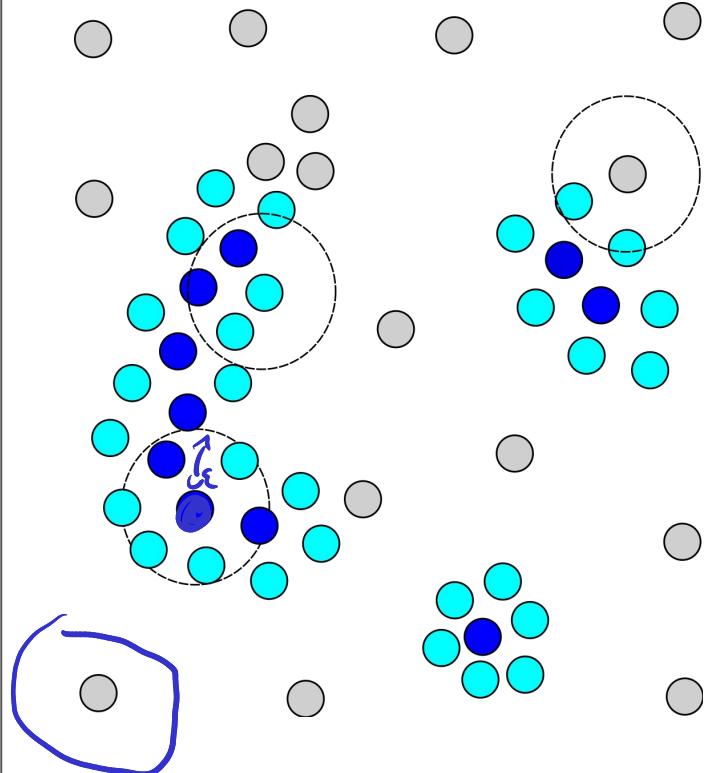
- Non-core points with at least one core point in their neighborhood
- Form the **border** of a cluster

- **Outliers / noise** ○

- All other points
- Default node type

Example:  $\mathcal{E}$  represented by circles

$MinPts = 5$



# DBSCAN — Algorithm (2 Iterative Phases)

## 1. Find new cluster seed (core point)

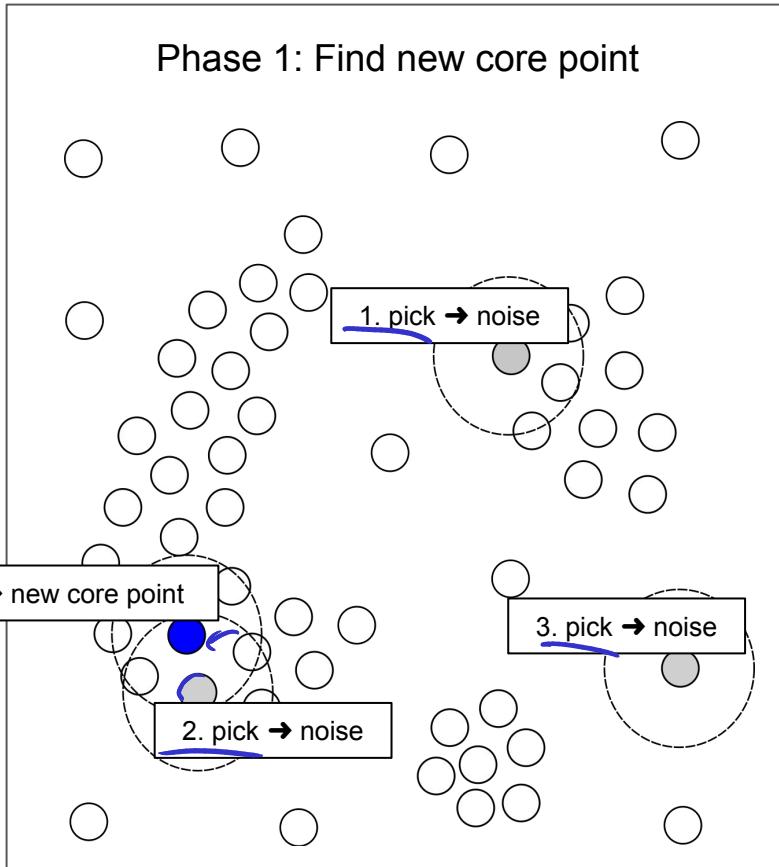
Repeat

- 1a) Pick random unexplored point  $x$
- 1b) If  $x$  has less than  $MinPts$  neighbors within  $\varepsilon$ ,  
label  $x$  as noise (might change later)

Until  $x$  is a new core point

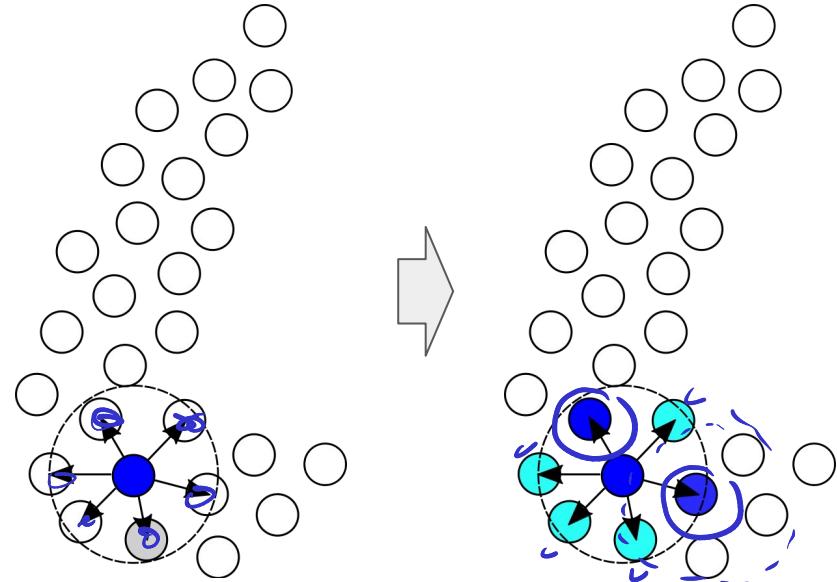


## 2. Explore new cluster



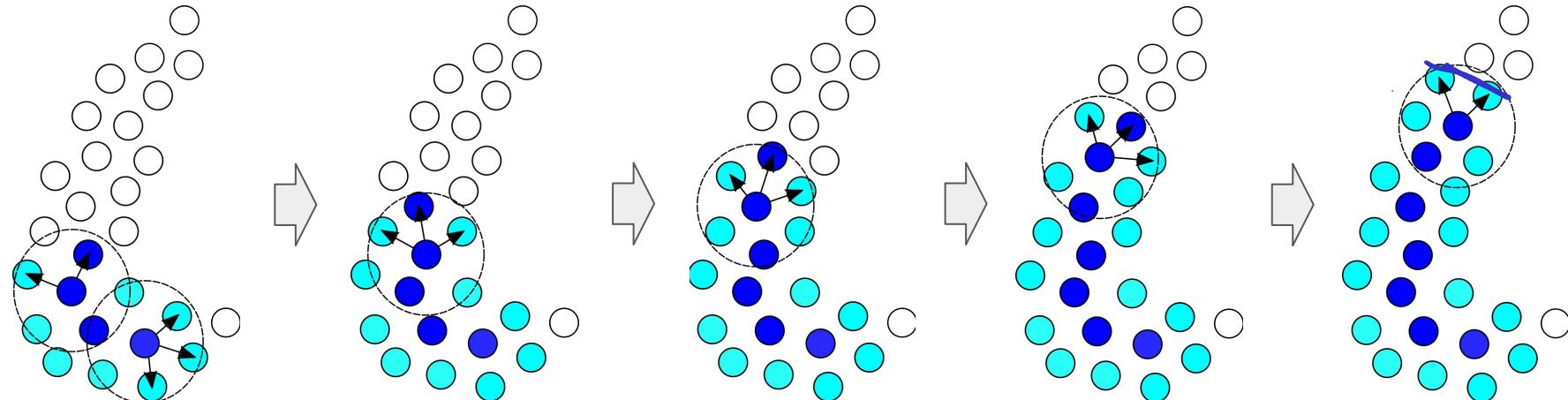
# DBSCAN — Algorithm (Cluster Exploration)

- Explore all neighbors of core point
  - Put all neighbors into the same cluster
  - A neighbor is either a core point or a border point (never noise)
  - New type of neighbor might overwrite previous noise label
- Recursively repeat for each newly found core point



# DBSCAN — Algorithm (Cluster Exploration)

- Example: complete exploration of a cluster
  - Further exploration stops at border points
  - Points beyond border points will become noise or part of a new cluster (after Phase 1)



# DBSCAN — Algorithm (Cluster Exploration)

- **Input:** newly found core  $x_c$  point signifying a new cluster

```
 $S \leftarrow get\_neighbors(x_c, \mathcal{E})$ 
```

```
WHILE  $|S| \neq 0$ 
```

```
     $s \leftarrow S.pop()$ 
```

```
    IF  $s.label = "noise"$  THEN  
         $s.label \leftarrow x_c.cluster\_id$ 
```

```
    IF  $s.label \neq "unknown"$  THEN  
        CONTINUE
```

```
     $s.label \leftarrow x_c.cluster\_id$ 
```

```
     $neighbors \leftarrow get\_neighbors(s, \mathcal{E})$ 
```

```
    IF  $|neighbors| \geq MinPts$  THEN  
         $S \leftarrow S \cup neighbors$ 
```

Initialize  $S$  with the the neighbors of  $x_c$

Pick next point  $s$  from  $S$  until  $S$  is empty

If point  $s$  is (currently) noise,  
add point to current cluster

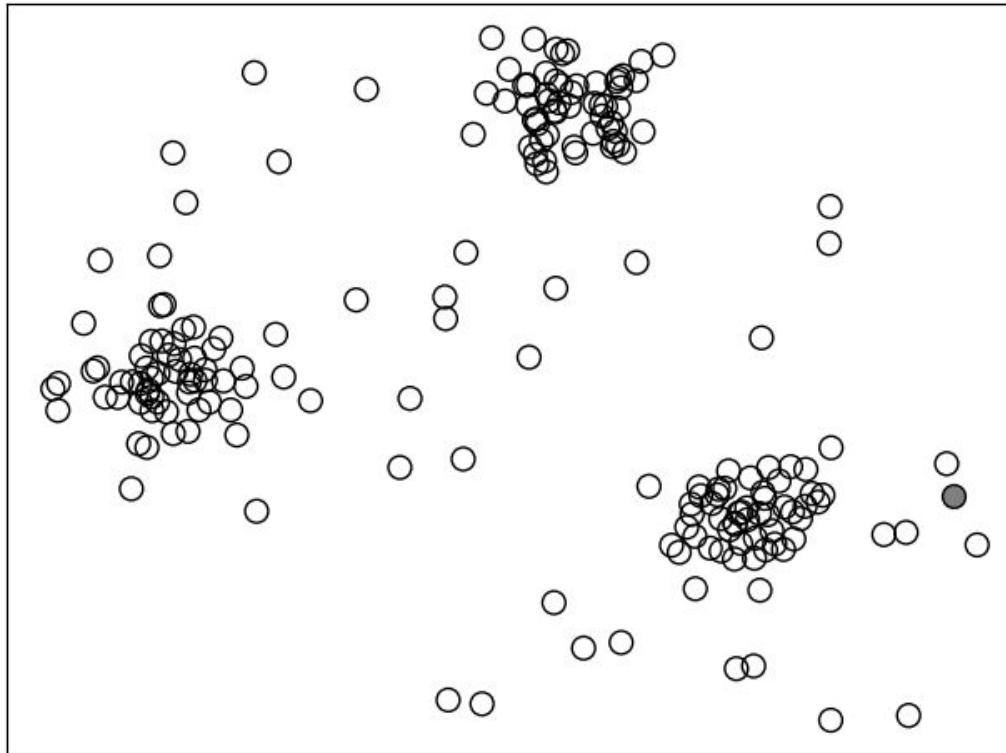
If  $s$  has already been explored,  
continue with next point

Add  $s$  current cluster ( $s$  so far unexplored)

Get all neighbors of  $s$

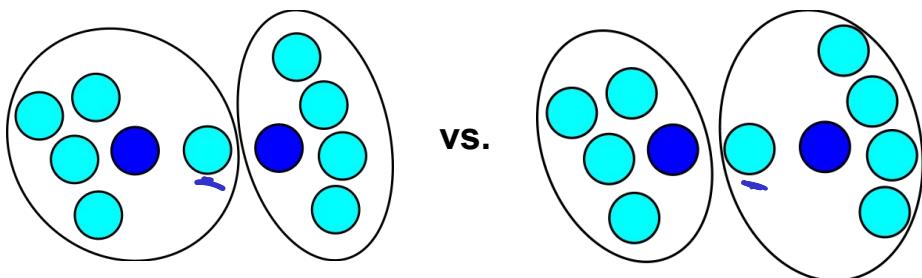
If  $s$  has more than  $MinPts$  neighbors,  $s$  is also a core point  
→ add neighbors to  $S$  (so they will be explored as well)

# DBSCAN — Example with Visualization



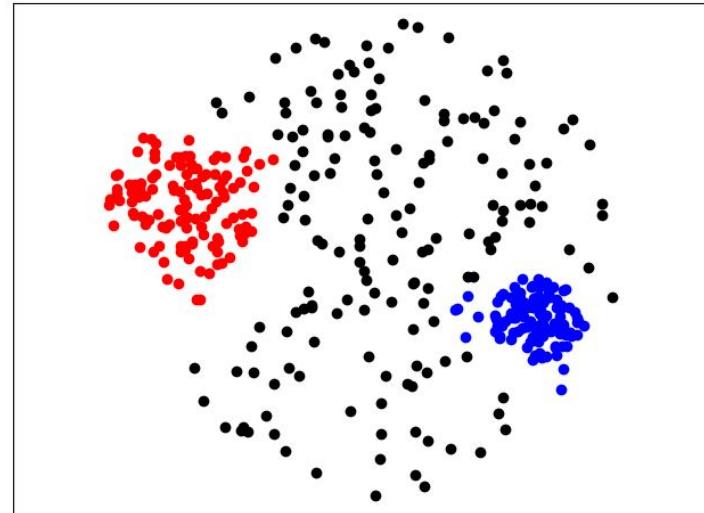
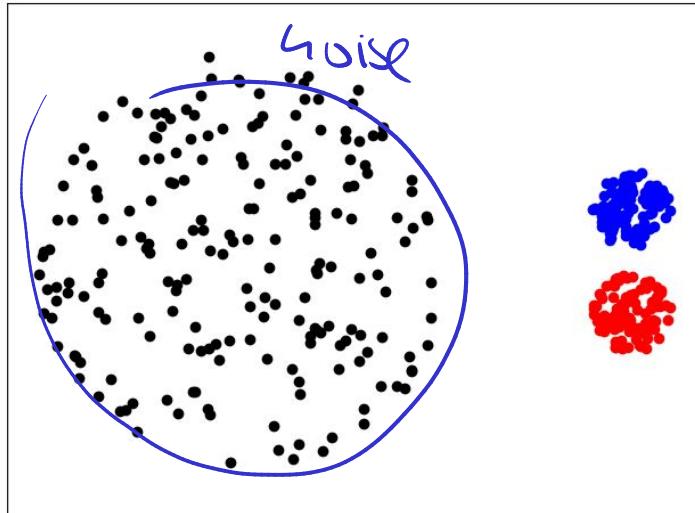
# DBSCAN — Characteristics

- DBSCAN always converges
  - Each data point gets explored (either in Phase 1 or 2)
  - A data point does not change its type (only exception: noise → border)
- DBSCAN is not completely deterministic
  - Phase 1 introduces randomness
  - Border points may be reachable from core points of different clusters
  - Noise and core points deterministic



# DBSCAN — Limitations

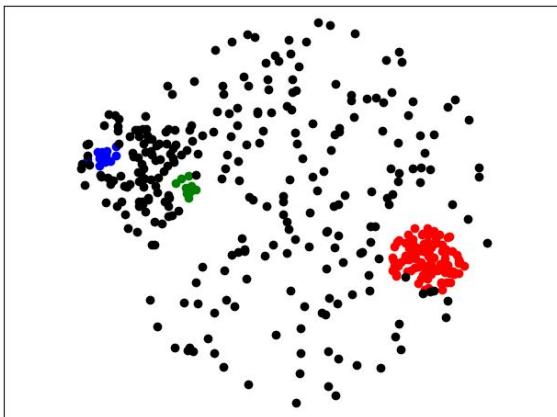
- DBSCAN cannot handle different densities



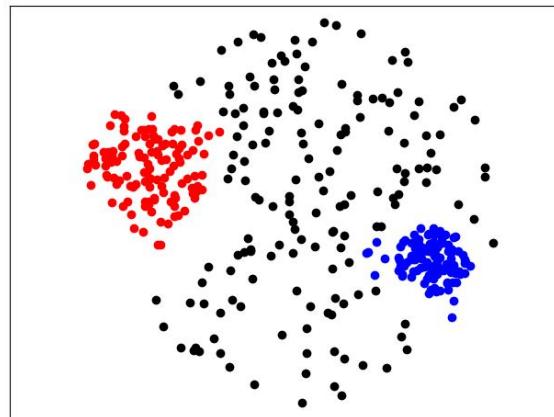
# DBSCAN — Limitations

- DBSCAN is generally very sensitive to parameters
  - Choosing  $\varepsilon$  and  $MinPts$  requires good understanding of data and context

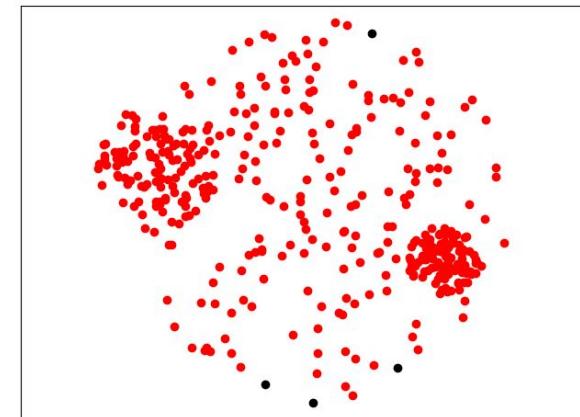
$$\varepsilon = 0.05$$



$$\varepsilon = 0.1$$



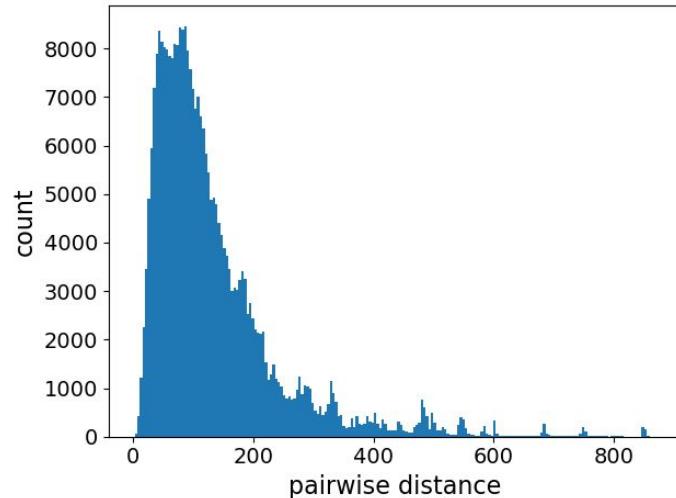
$$\varepsilon = 0.2$$



( $MinPts = 10$  for all three examples)

# DBSCAN — How to Choose Parameter Values?

- Informed by results of EDA, e.g.:



Distribution of all pairwise distances

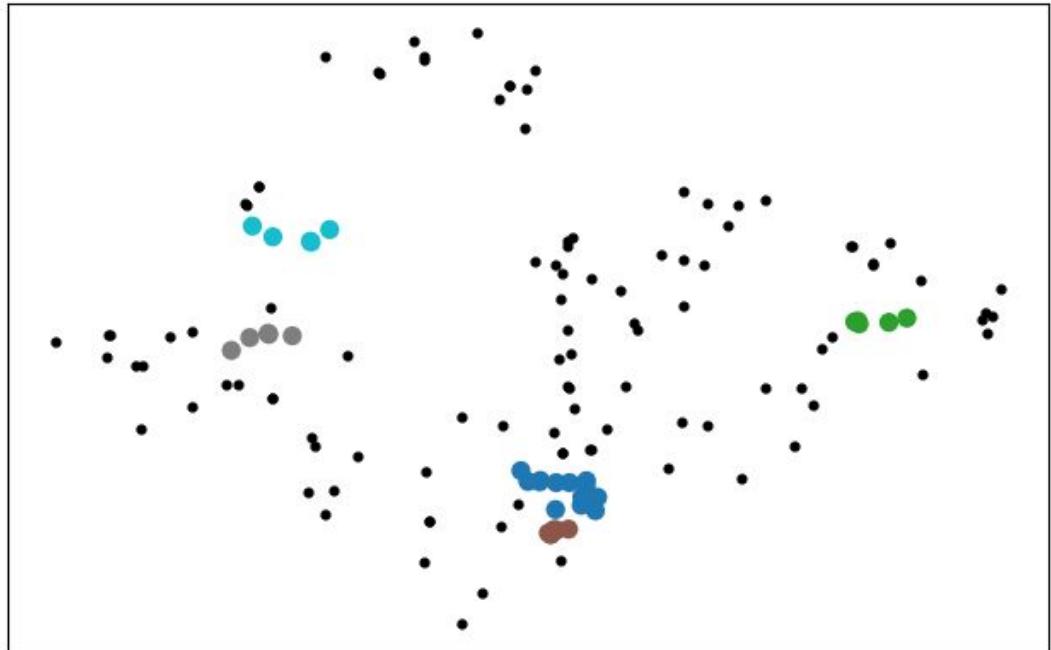
First insights into suitable values for  $\varepsilon$

- Density of data points has intuitive semantic meaning, e.g.:
  - Geographic distance between bars in a city
  - Task: Find areas (clusters) with more than 10 bars within 500m

# DBSCAN — How to Choose Parameter Values?

- Intuitive interpretations of meaningful parameter values
- Example
  - 141 McDonald's restaurants across Singapore
  - Find areas with more than 5 restaurants within a 500m radius

*min pts*  
~~~~~  
*ε*



# Quick Quiz

What is the **smallest cluster size** when using DBSCAN and a given value for  $MinPts$ ?

A

1

B

$MinPts - 1$

C

$MinPts$

D

$MinPts + 1$

# Quick Quiz

Given  $N=1,000$  data points and using DBSCAN with  $MinPts=10$ , what is the **minimum** possible number of clusters?

3

4

5

A 0

B 1

C MinPts

D MinPts + 1

# Quick Quiz — Side Note

- Slight inconsistencies across different sources
  - Relevant step in algorithm

$\text{neighbors} \leftarrow \text{get\_neighbors}(s, \mathcal{E})$

Does  $\text{neighbors}$  contain  $s$  itself?

- Original paper: Yes,  $s$  is part of neighborhood

- Smallest cluster size:  $\text{MinPts}$

- Maximum number of clusters:  $\left\lfloor \frac{N}{\text{MinPts}} \right\rfloor$      $N$  = number of data points

# Clustering Algorithms

- K-Means
- DBSCAN
- Hierarchical Clustering (next lecture)

# Outline

- Clustering
  - Overview
  - Applications
  - Concepts
- Clustering Algorithms
  - K-Means
  - DBSCAN
  - Hierarchical Clustering (next lecture)
- Cluster Evaluation

# Summary — Clustering I

- Clustering as fundamental data mining algorithm
  - Cluster provide a "meso-view" on data
  - Required: well-defined notion of similarity between data points
  - No single definition what a good cluster / clustering is

→ Wide range of different clustering algorithms

- In this lecture: K-Means & DBSCAN
  - K-Means: split all data points into k clusters based in their **relative similarities**
  - DBSCAN: find clusters based on **absolute similarities** between data points

# Solutions to Quick Quizzes

- Slide 18: D
- Slide 32: B
- Slide 37: A
- Slide 51: C
- Slide 52: A

# **CS5228: Knowledge Discovery and Data Mining**

## Lecture 3 — Clustering II

# Course Logistics — Update

- Assignment 1

- Topics: EDA, Data Preparation & K-Means
- Submission deadline: Thu, Sep 12 (11.59 pm)

- Reminder: Honor Code

- Cheat and plagiarism is serious academic offence
- Do not read, copy, steal, etc. others code
- Lecture slides & videos should easily suffice

- Project

- Team formation about to be finalized
- Kaggle Competition will launch soon.

New submission deadline:  
Thu, Nov 14 (Week 13)

# Quick Recap — Tutorial

| city | state | parent |
|------|-------|--------|
| p-b  | PA    | Bach   |
| p c  | OK    | S      |
| p-b  | FL    | ma     |
| p a  | CT    |        |
| p c  | WV    |        |
| p b  | IN    | assoc  |
| p b  | CO    |        |
| p-b  | MP    |        |
| p-d  | WY    |        |
| p-b  | MS    |        |

- Alternative encoding of nominal attributes: "*proxy encoding*"

- Replace nominal values with 1 or more numerical values
- Numerical values should reflect underlying assumption of the impact of attribute
- Example: "*What makes 'state' a potentially useful attribute?*"

| Interpretation            | Encoding through replacement          |
|---------------------------|---------------------------------------|
| Average Political leaning | → Percentage of democrats/republicans |
| State education budget    | → <u>Dollar-per-student value</u>     |
| School system             | → Rate of homeschooling               |
| Urbanization              | → #universities per capita            |
| ...                       | → ...                                 |

# Quick Recap — Tutorial

| city | state | parent |
|------|-------|--------|
| p-b  | PA    | Bach   |
| p-c  | OK    | S      |
| p-b  | FL    | ma     |
| p-a  | CT    |        |
| p-c  | WV    |        |
| p-b  | IN    | assoc  |
| p-b  | CO    |        |
| p-b  | MP    |        |
| p-d  | WY    |        |
| p-b  | MS    |        |

- Important: "careless" encoding may imply questionable interpretation
  - Question: *"What is the interpretation of my encoding, and is it meaningful?"*
  - In practice, often very difficult to answer

| Encoding                  | Interpretation                         |
|---------------------------|----------------------------------------|
| Ordinal values            | → PA < OK < FL < CT < WV < ...         |
| <u>Latitude/Longitude</u> | → Geographic location of state matters |
| #KFC per capita           | → Proliferation of fast food matters   |
| ...                       | → ...                                  |

It **might** be correct, even if only incidentally!

# Quick Recap — Lecture

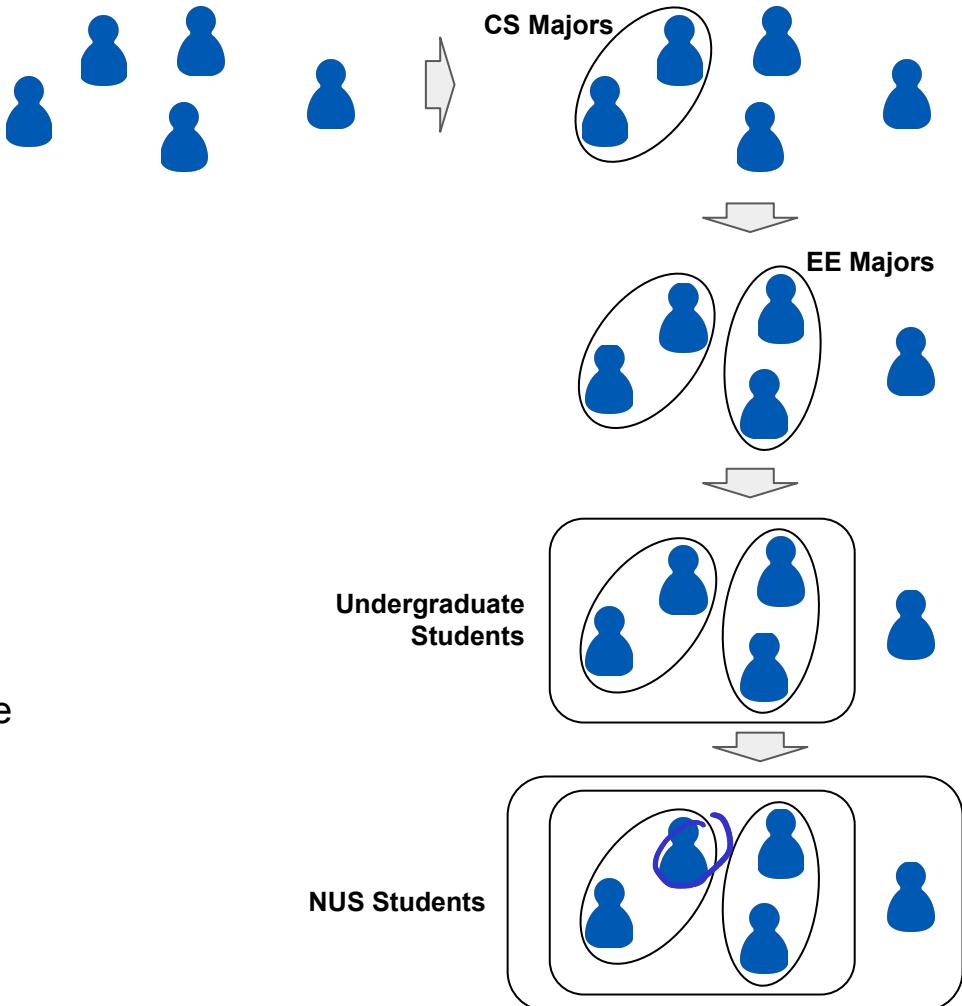
- **Clustering**
  - Grouping data points based on their similarities
  - No single definition for cluster or clustering → different meaningful intuitions
  - General-purpose data mining method ("only" distance/similarity measure required)
- **Algorithms discussed so far:**
  - K-Means (centroid-based, partitional, exclusive, complete)
  - DBSCAN (density-based, partitional, exclusive, partial)

# Outline

- **Clustering**
  - Overview
  - Concepts
  - Applications
- **Clustering algorithms**
  - K-Means
  - DBSCAN
  - Hierarchical Clustering
- Cluster Evaluation

# Hierarchical Clustering

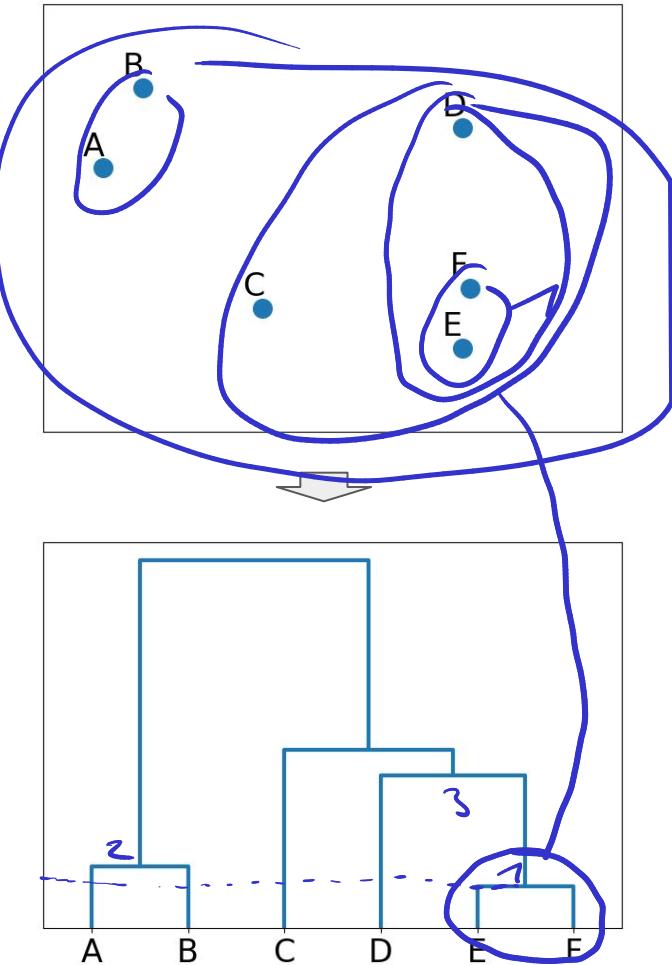
- Basic characteristics
  - Clusters: depends...
  - Clustering: hierarchical (duh!), complete, exclusive (at each level!)
- No parameterization (in principle)
  - In practice, typically number of clusters is specified (similar to K-means)
  - Different choices of measures to calculate distances between clusters



# Dendrograms

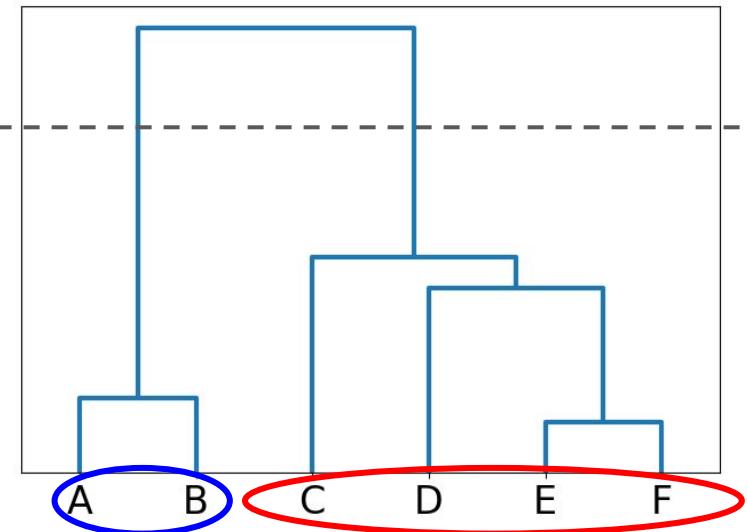
greek: tree

- Dendrogram: Visualization of hierarchical relationships
  - Binary tree showing how clusters are hierarchically merged/split
  - Each node is a cluster
  - Each leaf is a singleton cluster
  - Height reflects distance between clusters  
(e.g., large distance between A/B and C/D/E/F clusters)

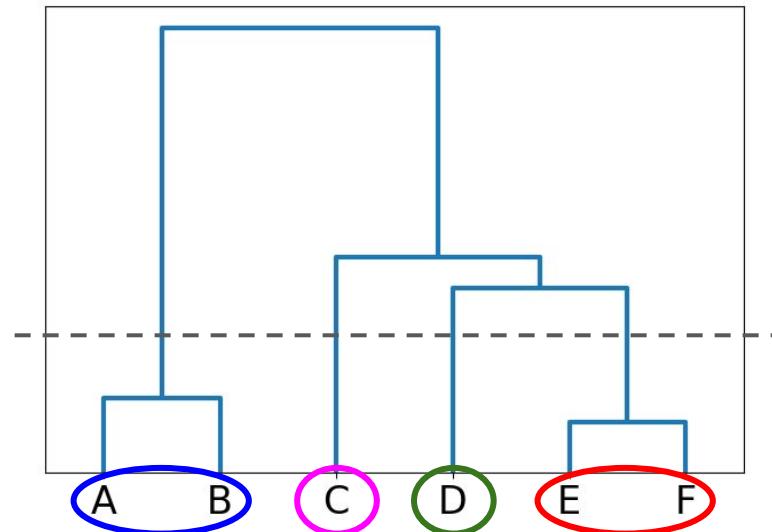


# Hierarchical Clustering — Dendograms

- A clustering can be obtained by cutting a dendrogram at the desired level



2 clusters

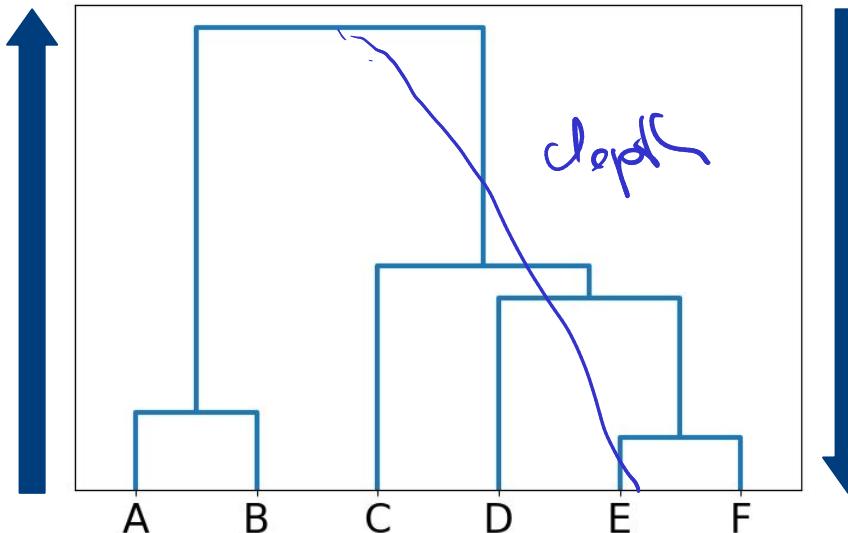


4 clusters

# Hierarchical Clustering — 2 Main Types

## Agglomerative (bottom-up)

- Start with each point being its own cluster
- At each step, merge closest pair of clusters
- Stop when only one cluster is left



**AGNES** (AGglomerative NESting)

## Divisive (top-down)

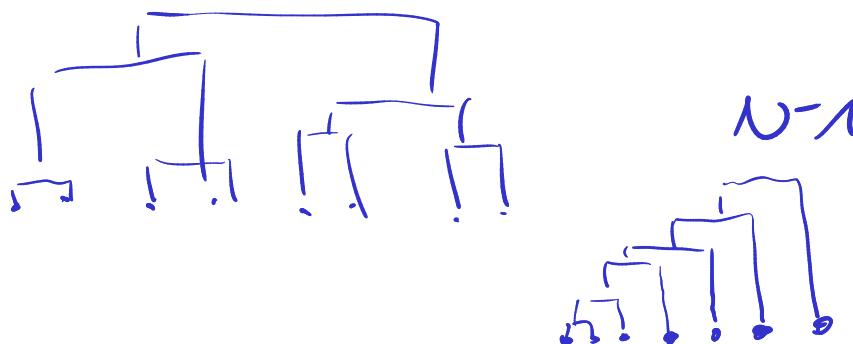
- Start with one cluster containing all points
- At each step, split a cluster
- Stop when each cluster contains a single point

**DIANA** (DIvise ANAlysis)

# Quick Quiz

What is the **minimum** and **maximum** possible **depth** of a dendrogram for a dataset with  $N$  data points?

(assume O-notation)



**A**



Min:  $\log_2 N$  Max:  $N$

**B**

Min:  $\sqrt{N}$  Max:  $N \log_2 N$

**C**

Min:  $\sqrt{N}$  Max:  $N$

**D**

Min:  $\log_2 N$  Max:  $N \log_2 N$

# AGNES — Basic Algorithm

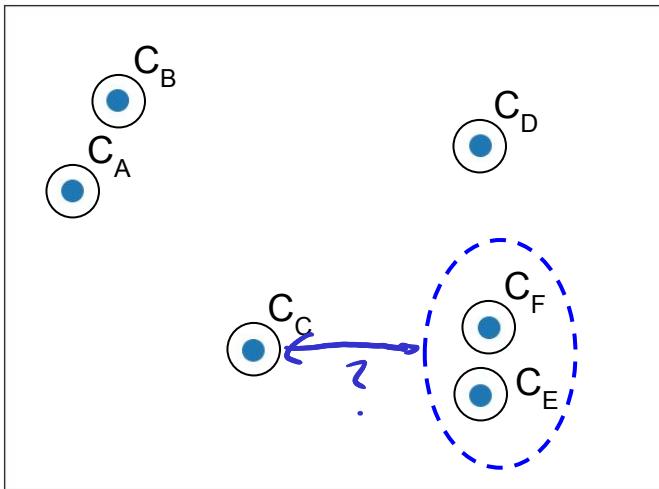
1. Initialization: Each point forms its own cluster
  2. Repeat
    - 2a) **Merge** the two closest clusters into one
- Until** only 1 cluster remains



# AGNES — Implementation

- Implementation using distance matrix

Initial clustering: each cluster, one point



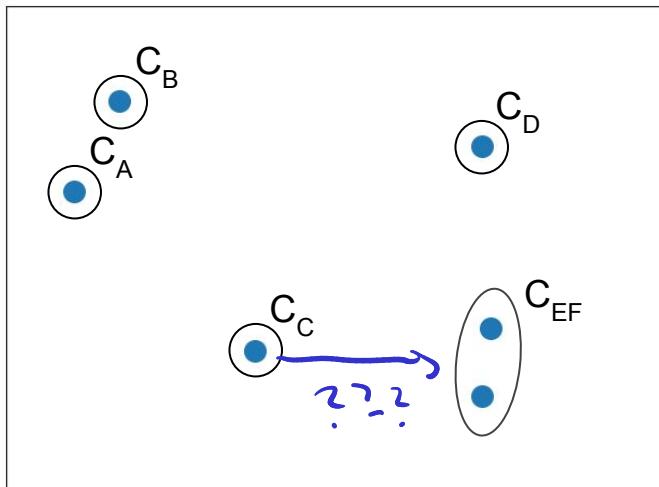
Distance between clusters = distance between points

|                | C <sub>A</sub> | C <sub>B</sub> | C <sub>C</sub> | C <sub>D</sub> | C <sub>E</sub> | C <sub>F</sub> |
|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| C <sub>A</sub> | $\infty$       | 2.25           | 5.32           | 9.06           | 9.79           | 9.49           |
| C <sub>B</sub> |                | $\infty$       | 6.08           | 7.85           | 9.86           | 9.21           |
| C <sub>C</sub> |                |                | $\infty$       | 6.73           | 4.81           | 5.02           |
| C <sub>D</sub> |                |                |                | $\infty$       | 5.51           | 4.00           |
| C <sub>E</sub> |                |                |                |                | $\infty$       | 1.53           |
| C <sub>F</sub> |                |                |                |                |                | $\infty$       |

# AGNES — Implementation

- What's the distance between clusters? (beyond containing single points)

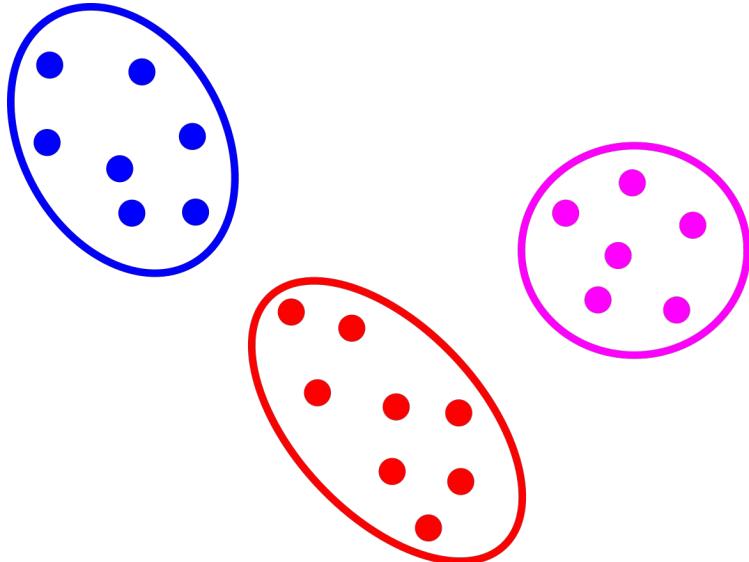
Clustering after merging  $C_E$  and  $C_F$  to  $C_{EF}$



|          | $C_A$    | $C_B$    | $C_C$    | $C_D$    | $C_{EF}$ |
|----------|----------|----------|----------|----------|----------|
| $C_A$    | $\infty$ | 2.25     | 5.32     | 9.06     | ???      |
| $C_B$    |          | $\infty$ | 6.08     | 7.85     | ???      |
| $C_C$    |          |          | $\infty$ | 6.73     | ???      |
| $C_D$    |          |          |          | $\infty$ | ???      |
| $C_{EF}$ |          |          |          |          | $\infty$ |

# Quick "Quiz"

Which 2 clusters should  
get merged next?



A ✓ Red & Blue

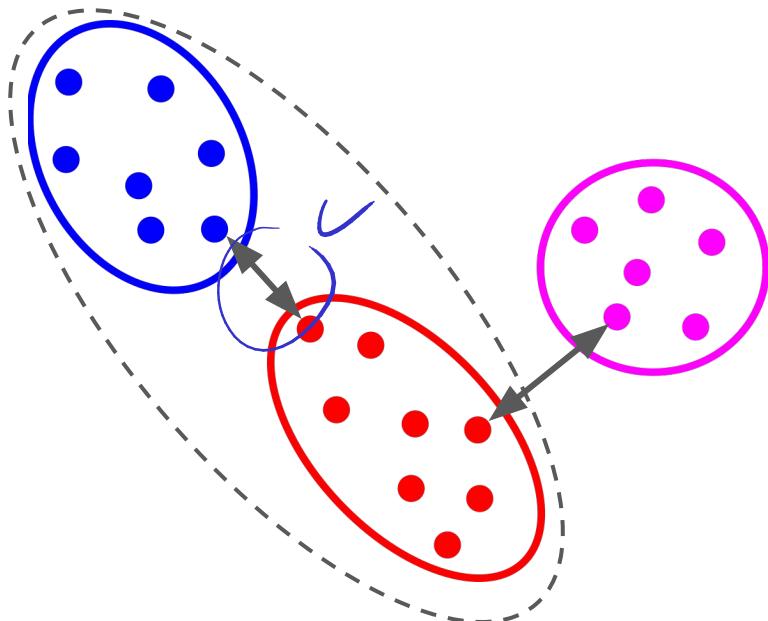
B ✓ Red & Magenta

C ✗ Blue & Magenta

D It's Friday,  
I'm out...

# AGNES — Single Linkage

- Single Linkage Clustering
  - Distance between clusters = **minimum distance** between two points from each cluster



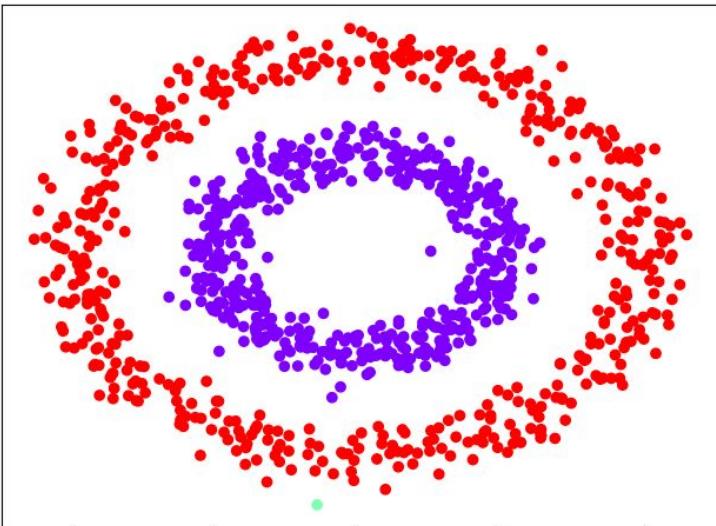
simple pointwise distance

$$d_{single}(C_i, C_j) = \min_{p \in C_i, q \in C_j} d(p, q)$$

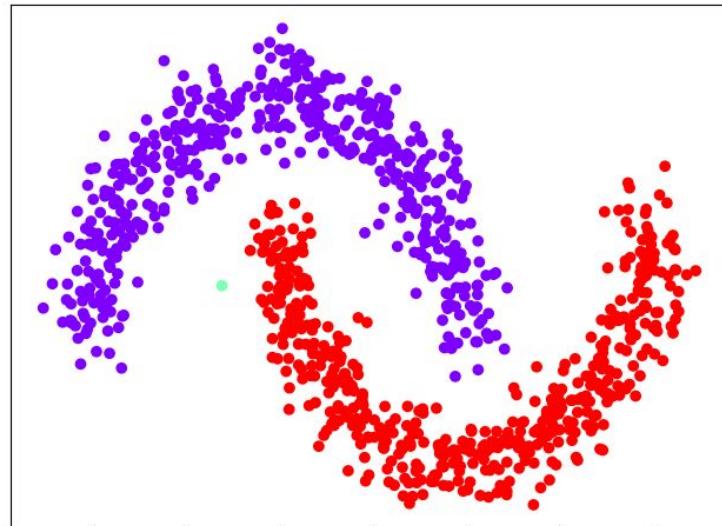
# AGNES — Single Linkage

- Strength: Can handle non-globular shapes

#cluster = 3

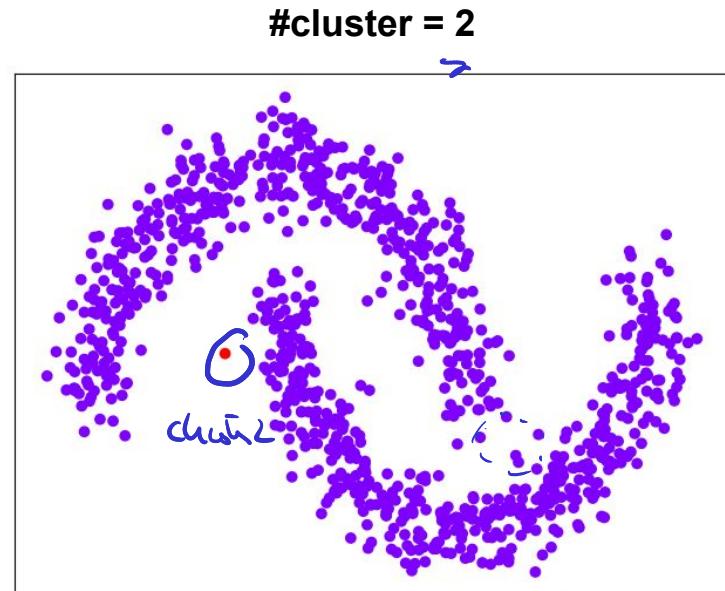
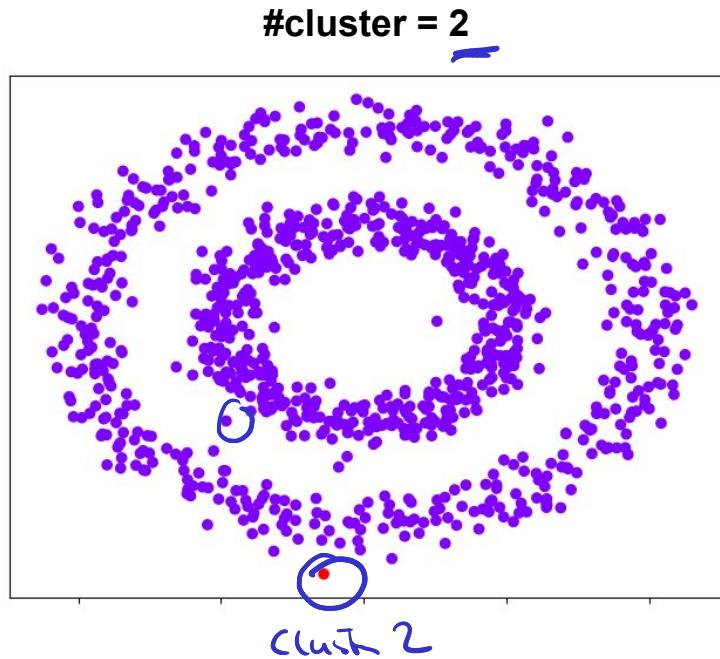


#cluster = 3



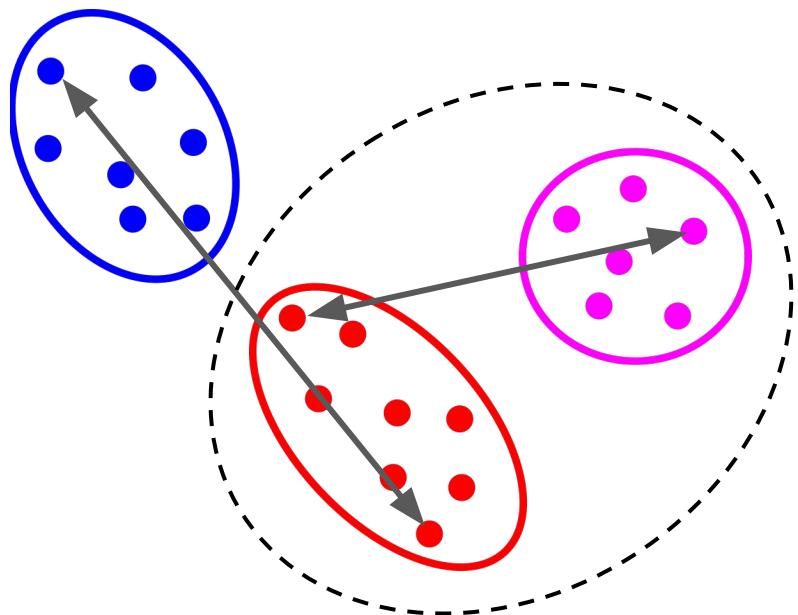
# AGNES — Single Linkage

- Weakness: Very susceptible to noise → "Chaining"
  - A single point may cause two clusters get merged



# AGNES — Complete Linkage

- Complete Linkage Clustering
  - Distance between clusters = **maximum distance** between two points from each cluster

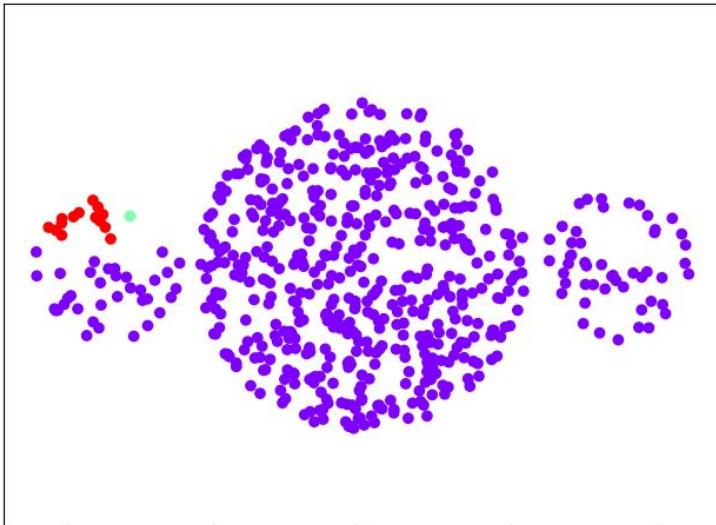


$$d_{complete}(C_i, C_j) = \max_{p \in C_i, q \in C_j} d(p, q)$$

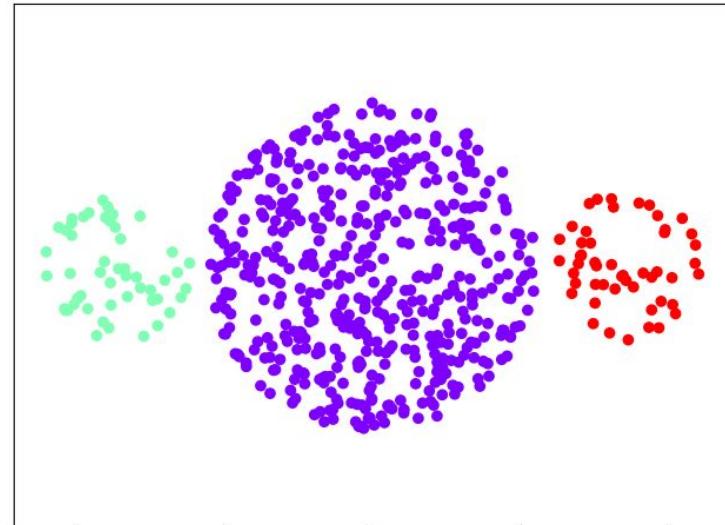
# AGNES — Complete Linkage

- Strength: Less susceptible to noise or outliers

Single Linkage, #cluster = 3



Complete Linkage, #cluster = 3

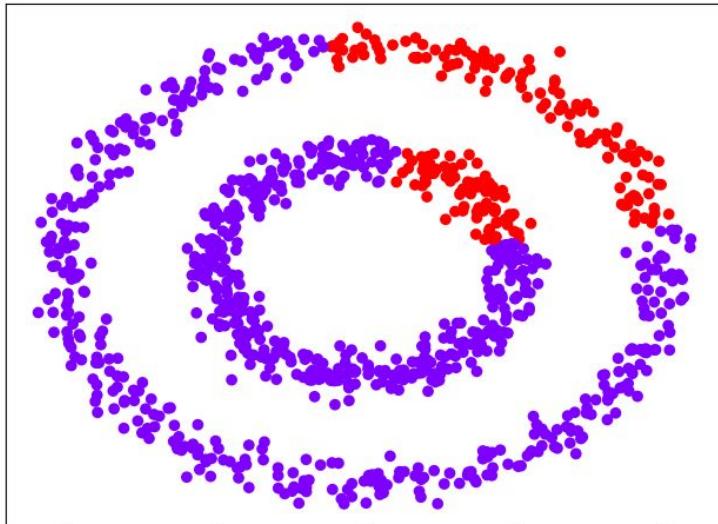


# AGNES — Complete Linkage

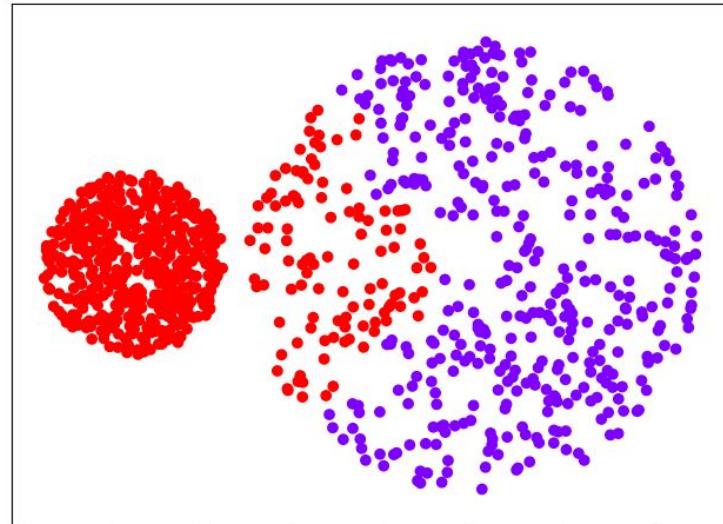
- Weaknesses

- Bias towards globular clusters
- Tends to break large clusters

#cluster = 2

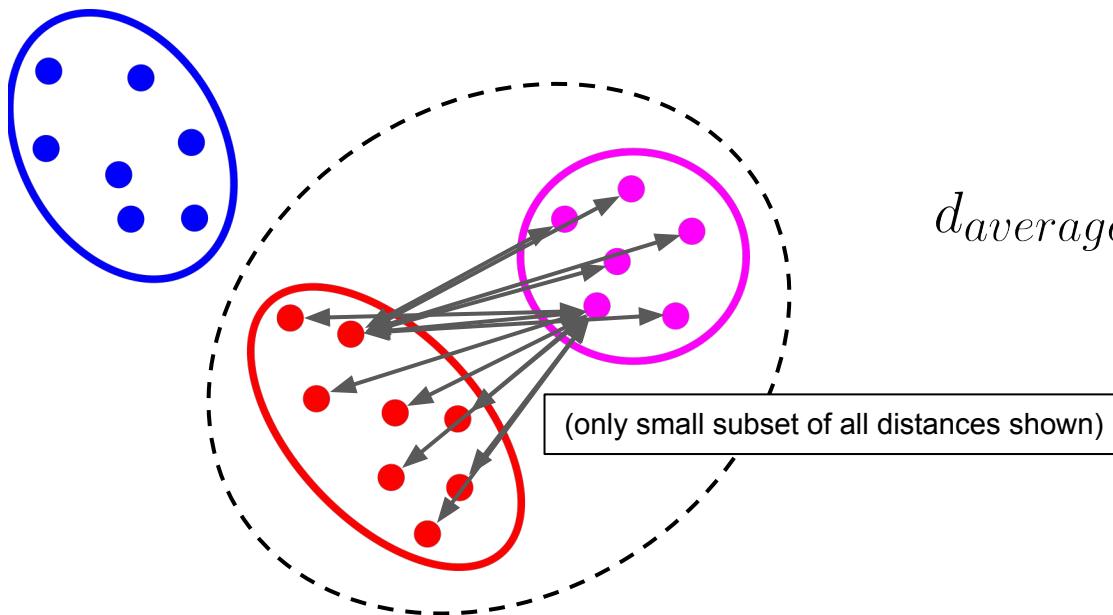


#cluster = 2



# AGNES — Average Linkage

- Complete Linkage Clustering (compromise between single and complete linkage)
  - Distance between clusters = **average distance** between two points from each cluster



$$d_{\text{average}}(C_i, C_j) = \underbrace{\text{avg}}_{p \in C_i, q \in C_j} d(p, q)$$

# AGNES — Linkage Alternatives

- **Centroid linkage**

- Distance between clusters = distance between the centroids of each cluster

$$d_{\text{centroid}}(C_i, C_j) = d(\underbrace{m_i, m_j}_{\text{centroid of cluster } i \text{ and } j (\text{m for mean})})$$

- **Ward linkage**

$$\begin{aligned} d_{\text{Ward}}(C_i, C_j) &= \overbrace{\sum_{k \in C_i \cup C_j} \|x_k - m_{ij}\|^2}^{\text{Variance of } C_{ij}} - \overbrace{\sum_{k \in C_i} \|x_k - m_i\|^2}^{\text{Variance of } C_i} - \overbrace{\sum_{k \in C_j} \|x_k - m_j\|^2}^{\text{Variance of } C_j} \\ &= \frac{n_i n_j}{n_i + n_j} \|m_i - m_j\|^2 \end{aligned}$$

$n_i$  = #points in cluster  $C_i$

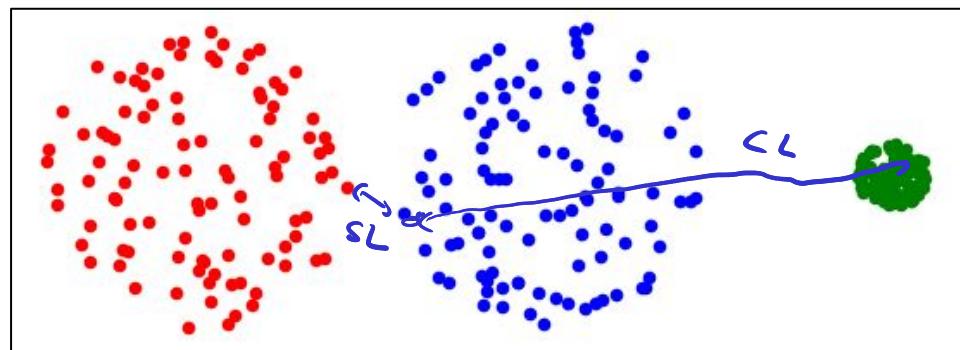
# Ward Linkage — Intuition

$$d_{Ward}(C_i, C_j) = \underbrace{\sum_{k \in C_i \cup C_j} ||x_k - m_{ij}||^2}_{\text{Variance of } C_{ij}} - \underbrace{\sum_{k \in C_i} ||x_k - m_i||^2}_{\text{Variance of } C_i} - \underbrace{\sum_{k \in C_j} ||x_k - m_j||^2}_{\text{Variance of } C_j}$$

- Example for Ward Linkage
  - Each blob: 100 data points

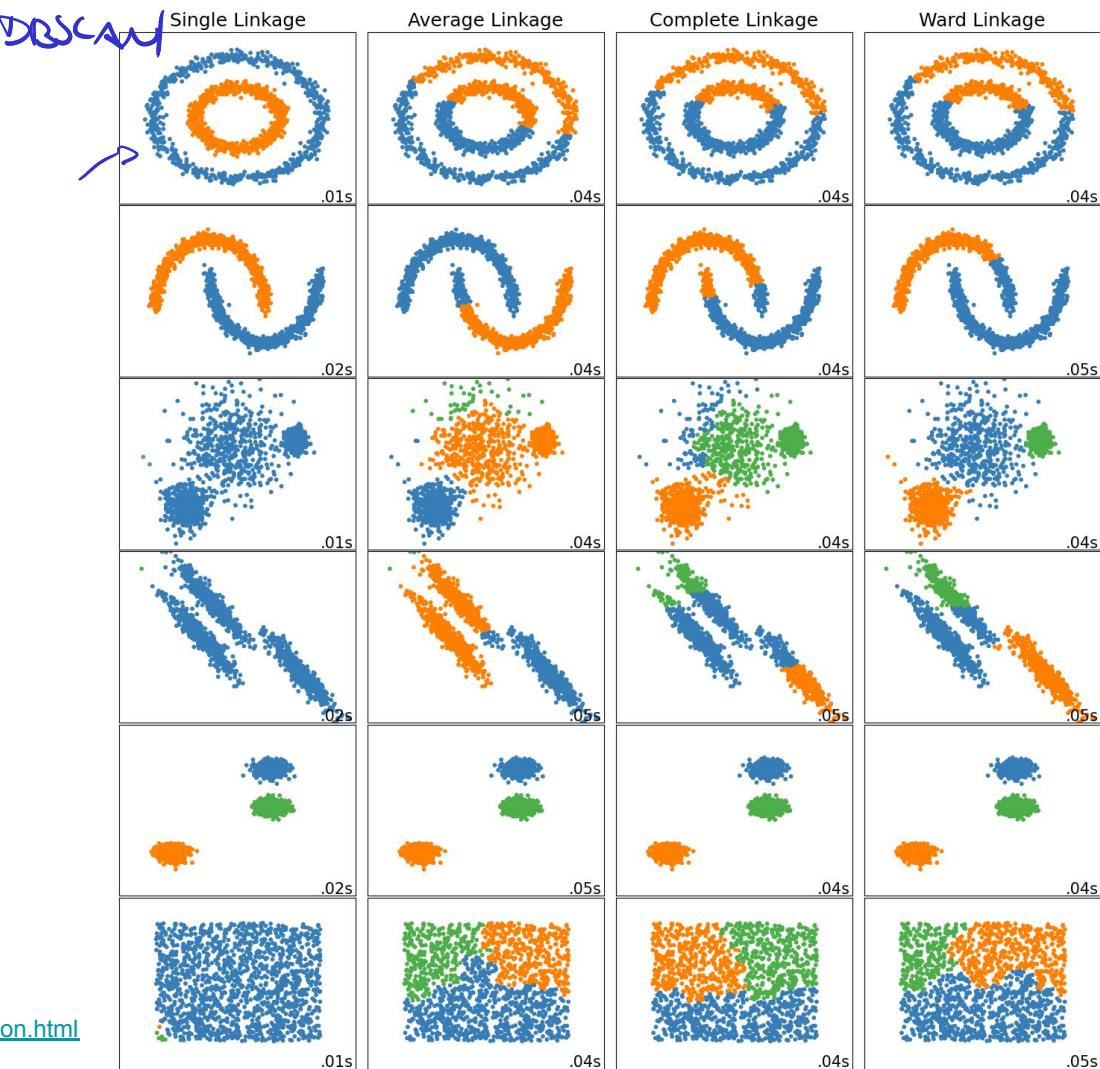
$$d_{Ward}(\text{Red, Blue}) = 1,635 - 195 - 200 = \underline{\underline{1,240}}$$

$$d_{Ward}(\text{Blue, Green}) = 1,450 - 200 - 10 = \underline{\underline{1,240}}$$



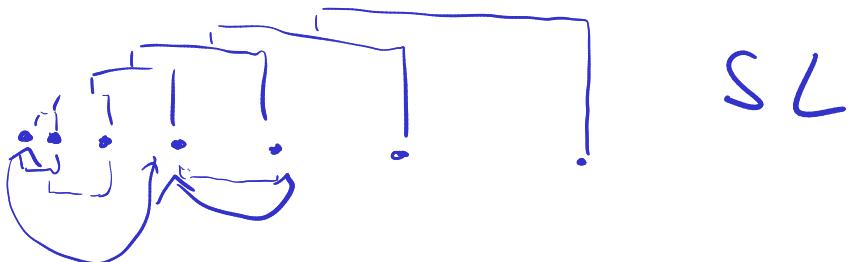
# AGNES

- Linkage comparison



# Quick Quiz

Which linkage method has intuitively the **highest chance** of returning a clustering where the dendrogram has a **depth of  $N-1$** ?



A ✓ Single

B Complete

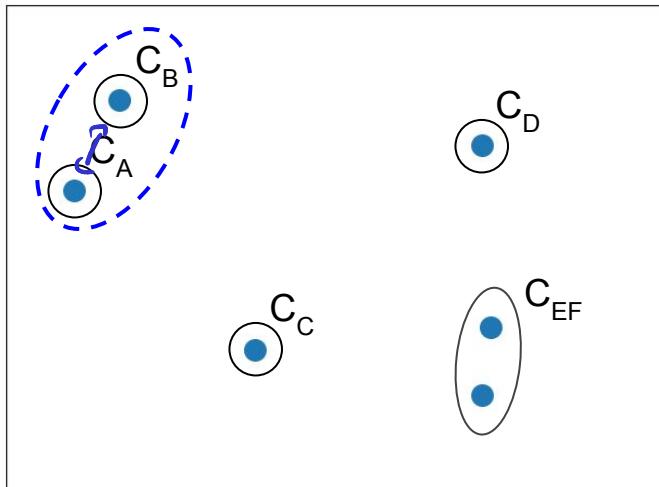
C Average

D No difference

# AGNES — Implementation

- Distance matrix after merging Cluster  $C_E$  and  $C_F$  + Average Linkage

Clustering after merging  $C_E$  and  $C_F$  to  $C_{EF}$

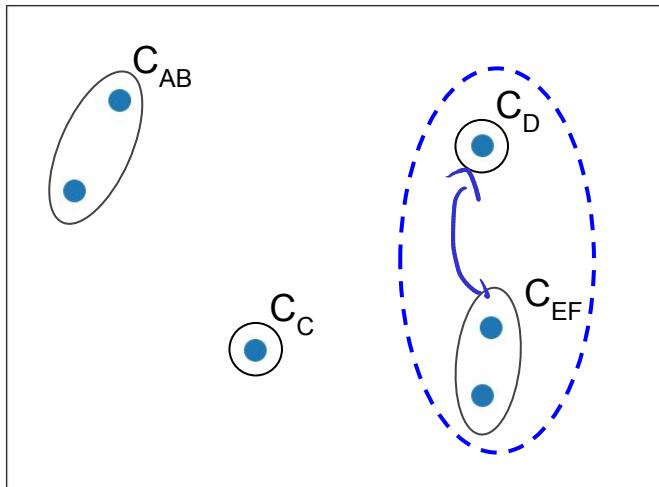


|          | $C_A$    | $C_B$       | $C_C$    | $C_D$    | $C_{EF}$ |
|----------|----------|-------------|----------|----------|----------|
| $C_A$    | $\infty$ | <b>2.25</b> | 5.32     | 9.06     | 9.64     |
| $C_B$    |          | $\infty$    | 6.08     | 7.85     | 9.54     |
| $C_C$    |          |             | $\infty$ | 6.73     | 4.92     |
| $C_D$    |          |             |          | $\infty$ | 4.76     |
| $C_{EF}$ |          |             |          |          | $\infty$ |

# AGNES — Implementation

- Distance matrix after merging Cluster  $C_A$  and  $C_B$  + Average Linkage

Clustering after merging  $C_A$  and  $C_B$  to  $C_{AB}$

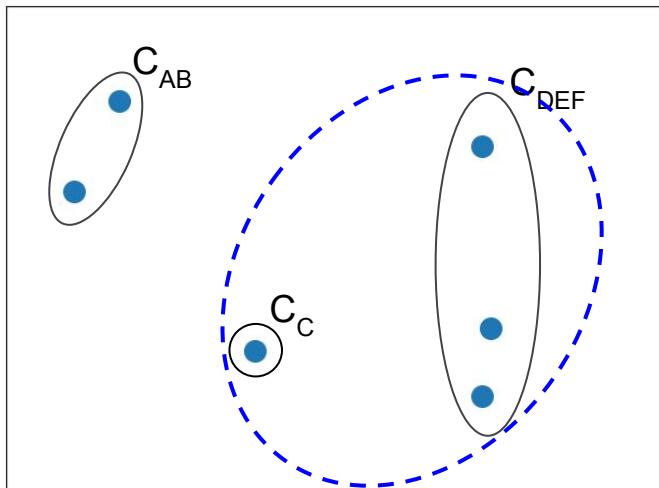


|          | $C_{AB}$ | $C_C$    | $C_D$    | $C_{EF}$    |
|----------|----------|----------|----------|-------------|
| $C_{AB}$ | $\infty$ | 5.70     | 8.45     | 9.59        |
| $C_C$    |          | $\infty$ | 6.73     | 4.92        |
| $C_D$    |          |          | $\infty$ | <b>4.76</b> |
| $C_{EF}$ |          |          |          | $\infty$    |

# AGNES — Implementation

- Distance matrix after merging Cluster  $C_C$  and  $C_{DEF}$  + Average Linkage

Clustering after merging  $C_D$  and  $C_EF$  to  $C_{DEF}$

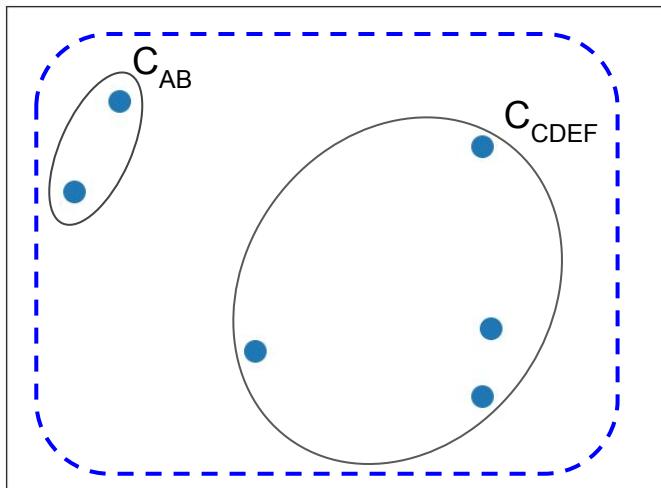


|           | $C_{AB}$ | $C_C$    | $C_{DEF}$   |
|-----------|----------|----------|-------------|
| $C_{AB}$  | $\infty$ | 5.70     | 9.21        |
| $C_C$     |          | $\infty$ | <b>5.51</b> |
| $C_{DEF}$ |          |          | $\infty$    |

# AGNES — Implementation

- Distance matrix after merging Cluster  $C_{AB}$  and  $C_{CDEF}$  + Average Linkage

Clustering after merging  $C_C$  and  $C_{DEF}$  to  $C_{CDEF}$

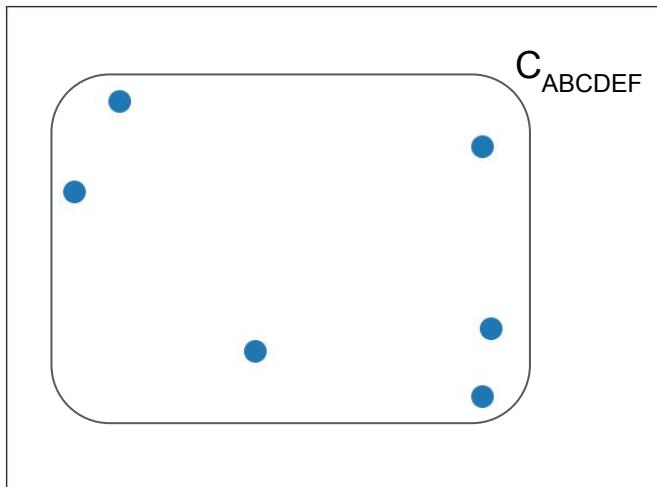


|            | $C_{AB}$ | $C_{CDEF}$  |
|------------|----------|-------------|
| $C_{CDEF}$ | $\infty$ | <b>8.33</b> |
|            |          |             |

# AGNES — Implementation

- Distance matrix after merging Cluster  $C_{AB}$  and  $C_{CDEF}$  + Average Linkage

Clustering after merging  $C_{AB}$  and  $C_{CDEF}$  to  $C_{ABCDEF}$



|              |              |
|--------------|--------------|
|              | $C_{ABCDEF}$ |
| $C_{ABCDEF}$ | $\infty$     |

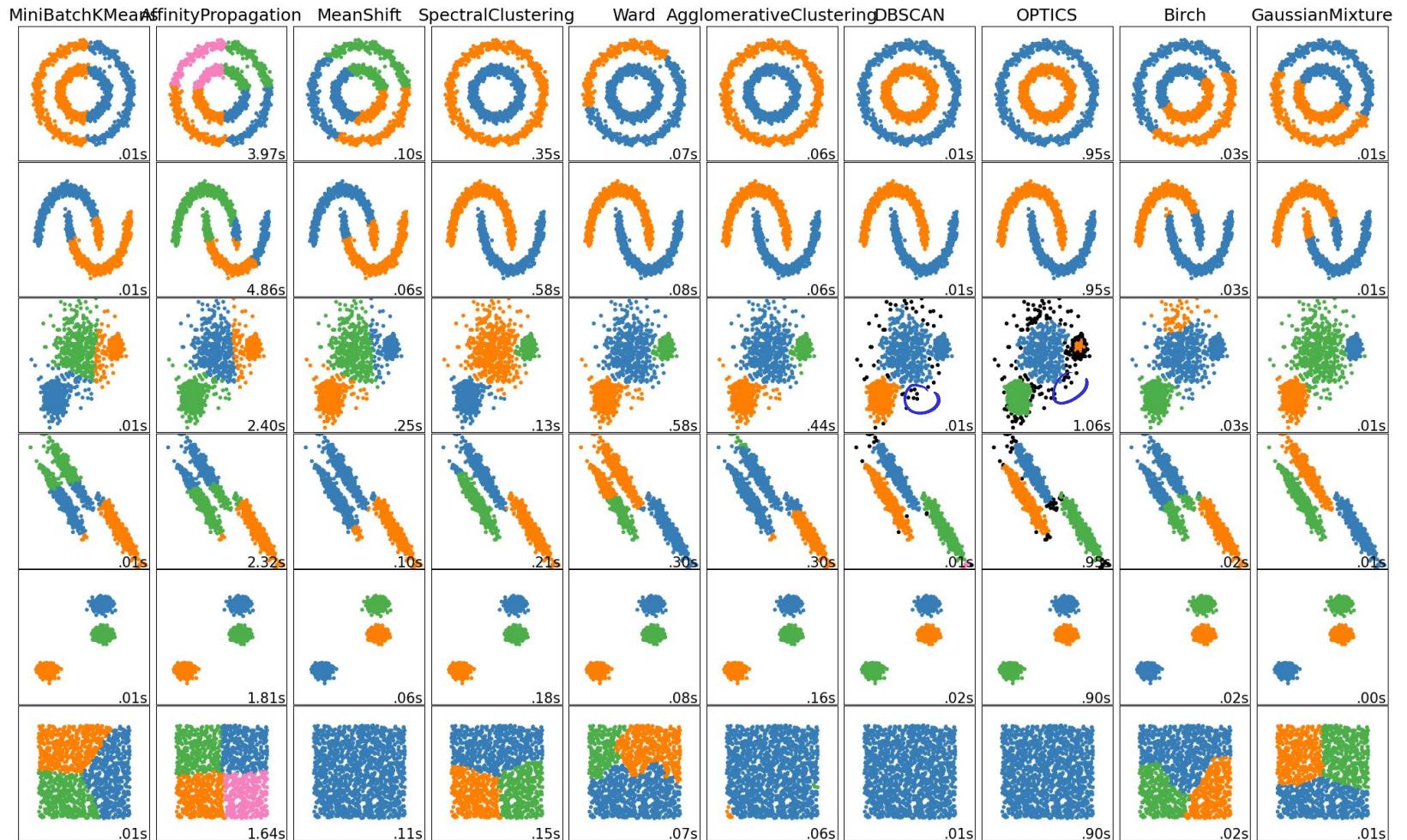
→ Done!

# AGNES — Complexity Analysis

- Space Complexity:  $O(N^2)$ 
  - Storing distance matrix
- Time Complexity
  - Baseline:  $O(N^3)$  — ( $N-1$ ) steps, each step  $O(N^2)$  to scan distance matrix
  - Using more sophisticated data structures, e.g, heap or priority queue:  $O(N^2 \log N)$
  - Special optimization for Single Linkage Clustering:  $O(N^2)$

# DIANA — DIvisive ANAlysis

- Top-Down Hierarchical Clustering
  - Start with all points forming one cluster
  - Recursively split one cluster until all clusters have size 1
- Challenge:  $2^n$  ways to split a cluster with  $n$  points
  - Heuristics needed to restrict search space
  - Generally slower and less common than AGNES
- Cases where DIANA can perform better
  - No complete clustering needed → early stopping
  - Splitting can utilize global knowledge (merging based on local knowledge only)



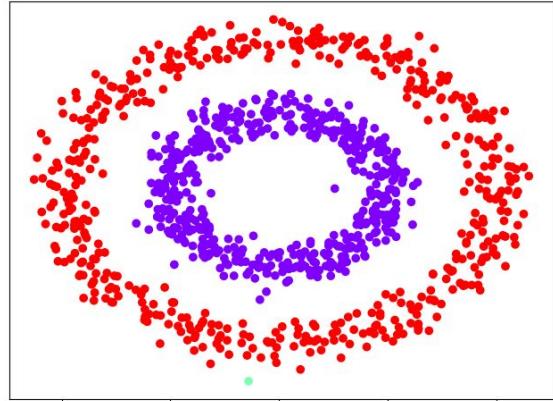
Source: <https://scikit-learn.org/stable/modules/clustering.html>

# Outline

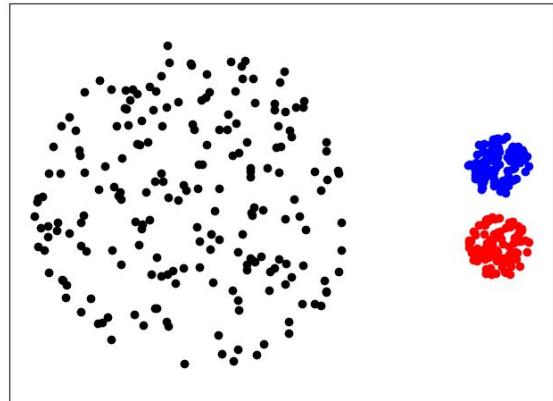
- **Clustering**
  - Overview
  - Concepts
  - Applications
- **Clustering algorithms**
  - K-Means
  - DBSCAN
  - Hierarchical Clustering
- **Cluster Evaluation**

# Cluster Evaluation

- Problem 1: Just eyeballing the clustering is rarely possible
  - High-dimensional data ( $\geq 3$  dimensions) difficult to impossible to visualize
  - Difficult to assess "nature" of clusters a-priori (e.g., variations in shape, size, density, etc)
  - Presence and distribution of noise or outliers



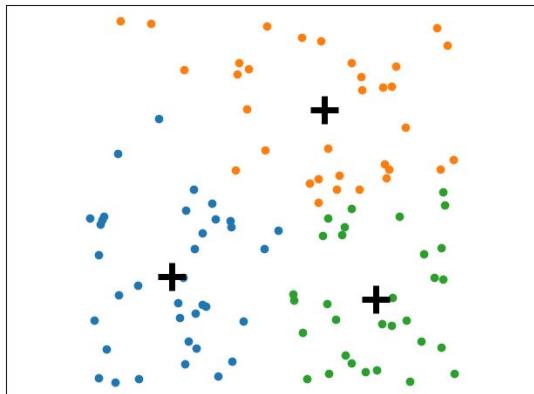
Your data usually does not look like this



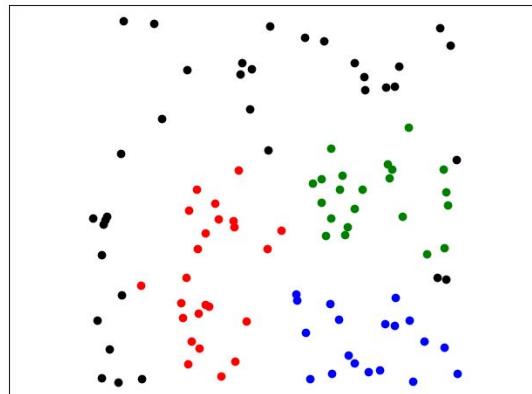
# Cluster Evaluation

- Problem 2: Clustering algorithms will always find some clusters
  - Example: K-Means, DBSCAN and AGNES applied to random data

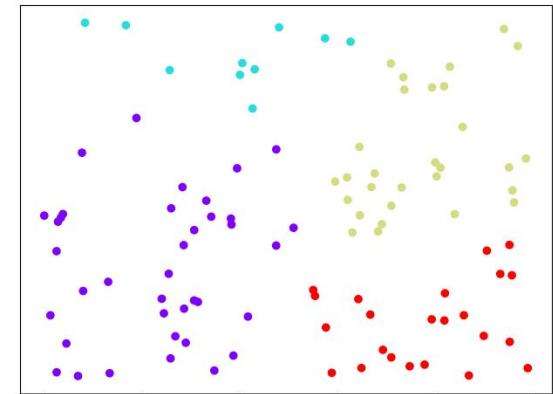
K-Means



DBSCAN



AGNES



# Cluster Evaluation

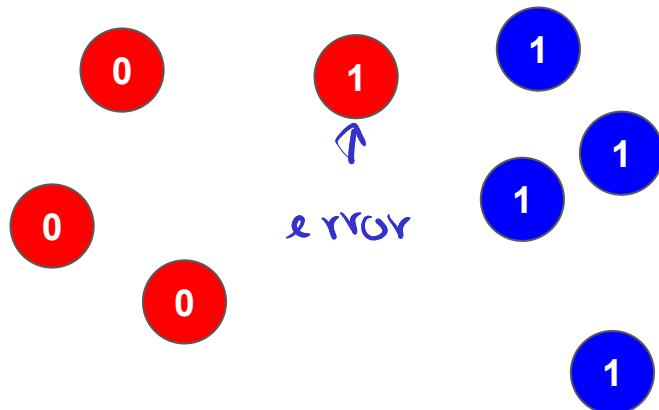
- Purpose of cluster evaluation
  - Comparing the results of different clustering algorithms
  - Comparing the results of a clustering algorithm with different parameters
  - Minimizing the effects of noise on the clustering

→ Getting a sense of the "goodness" of a clustering

- Two main approaches
  - External quality measures: evaluate a clustering against a ground truth (if available)
  - **Internal quality measures:** evaluate clustering from the data itself

# Cluster Evaluation — External Quality Measures

- Ground truth: Labeled data
  - Labels indicate that two points "belong together"
  - If cluster reflect this → good clustering



| Cluster | Label |
|---------|-------|
| Red     | 0     |
| Red     | 0     |
| Red     | 0     |
| Red     | 1     |
| Blue    | 1     |

# External Quality Measures — Cluster Purity

- Cluster purity  $P$

- $N$ : #points,  $C$ : set of cluster,  $L$ : set of labels

$$P = \frac{1}{N} \sum_{c \in C} \max_{l \in L} |c \cap l|$$

#points with most common  
label  $l$  in cluster  $c$

Purity for example:

$$P = \frac{1}{8}(3 + 4) = 0.875$$

- Limitations

- Purity does not penalize having many clusters  
→  $P=1$  easy to achieve with all clusters containing single point

$$P = \frac{1}{8} (1+1+1 \dots +1)$$

$\underbrace{\phantom{00000000}}_{8x}$

| Cluster | Label |
|---------|-------|
| Red     | 0     |
| Red     | 0     |
| Red     | 0     |
| Red     | 1     |
| Blue    | 1     |

# External Quality Measures: Information Retrieval Metrics

- Established metrics from classification tasks

- **TP** — true positives

same cluster, same label

(A/B, A/C, B/C, E/F, ..., G/H)

- **TN** — true negatives

different clusters, different labels

(A/E, A/F, A/G, A/H, B/E, ..., C/H)

- **FP** — false positives

same cluster, different labels

(A/D, B/D, C/D)

- **FN** — false negatives

different cluster, same label

(D/E, D/F, D/G, D/H)

For the example:

- **TP** = 9
- **TN** = 12
- **FP** = 3
- **FN** = 4

$$\binom{8}{2} = \overbrace{28}$$

| ID | Custer | Label |
|----|--------|-------|
| A  | Red    | 0     |
| B  | Red    | 0     |
| C  | Red    | 0     |
| D  | Red    | 1     |
| E  | Blue   | 1     |
| F  | Blue   | 1     |
| G  | Blue   | 1     |
| H  | Blue   | 1     |

# External Quality Measures — Information Retrieval Metrics

- Rand Index RI
  - Reflects accuracy

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

9 12  
} 28

$$RI_{example} = 0.75$$

- Precision P, Recall R, F1-Score

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

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

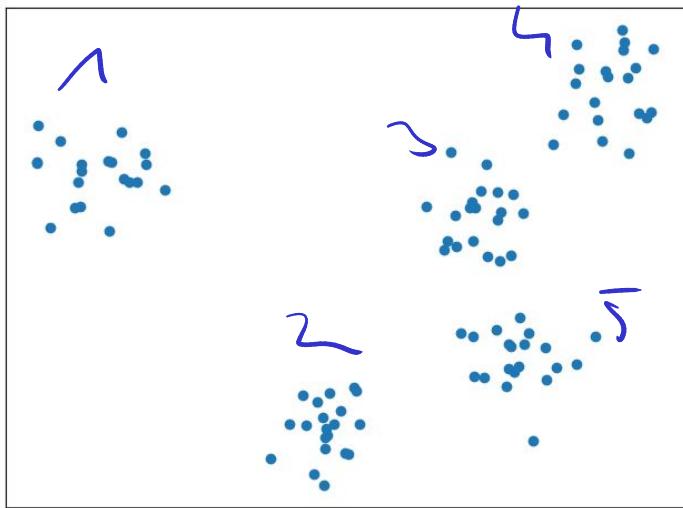
$$F1 = \frac{2 \cdot P \cdot R}{P + R}$$

$$P_{example} = 0.75 \quad R_{example} = 0.69 \quad F1_{example} = 0.72$$

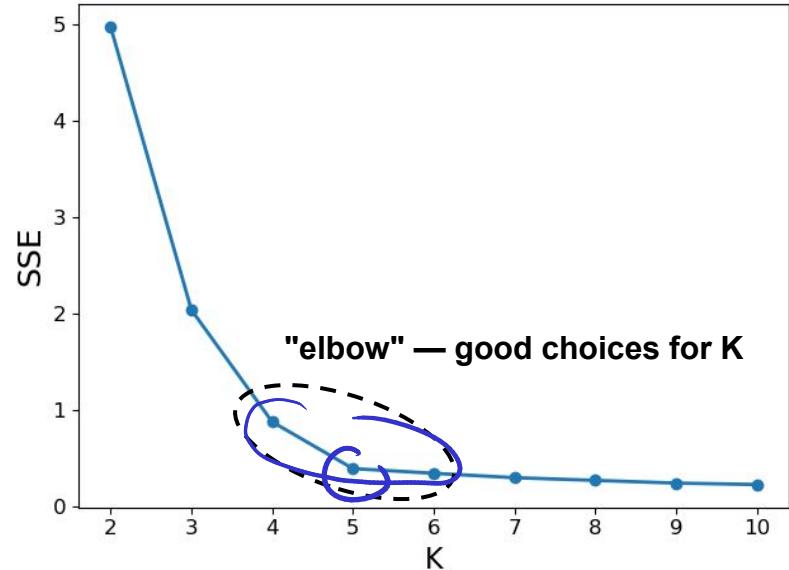
- ...and others using TP, TN, FP, FN

# Internal Quality Measures — SSE

- Use SSE to select number of clusters



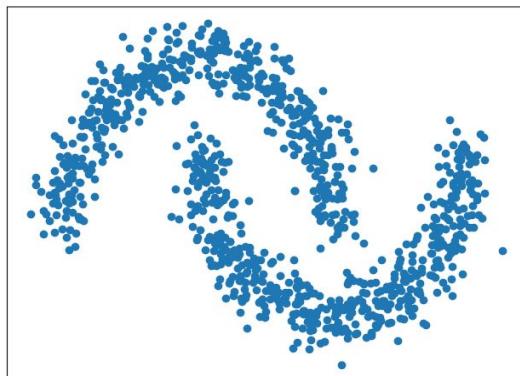
input data



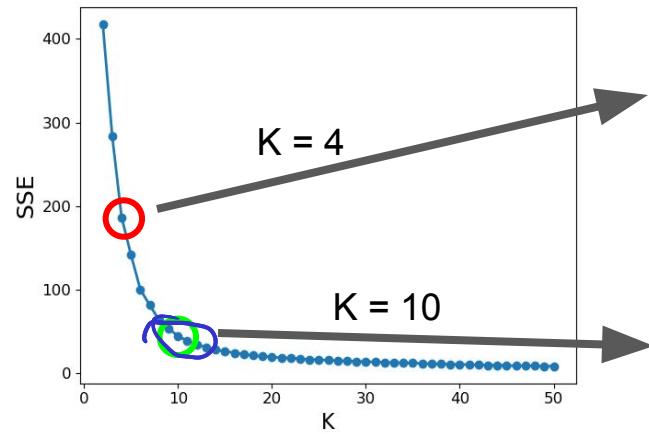
SSE for different K

# Internal Quality Measures — SSE

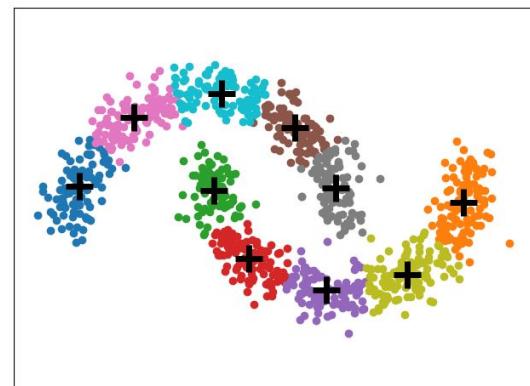
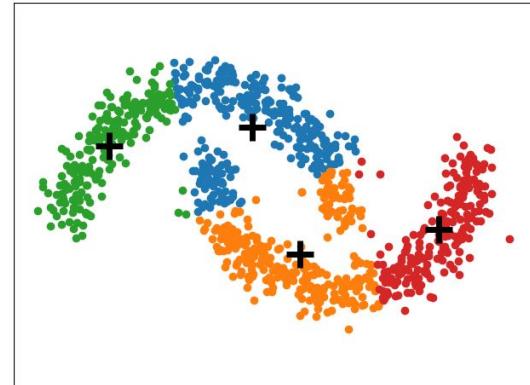
- Also applicable to more complicated data
  - But inherently "favors" globular clusters



input data

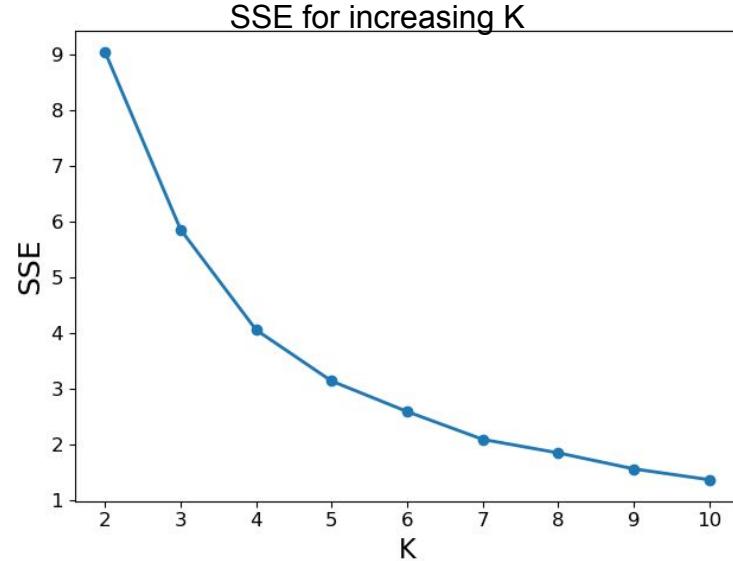
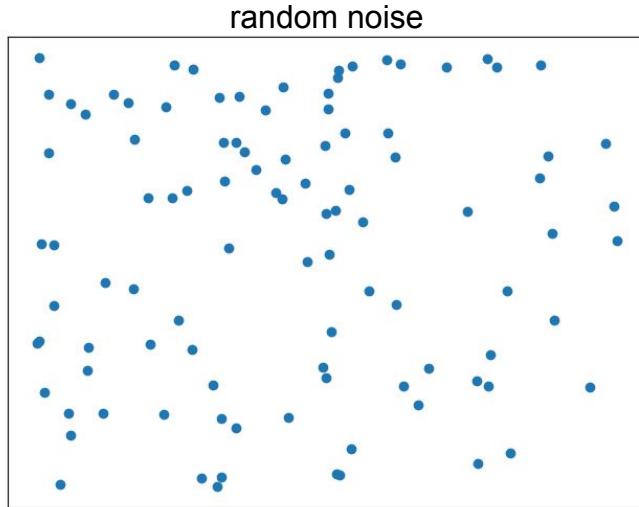


SSE for different K



# Internal Quality Measures — SSE

- Limitation of SSE as quality measure
  - SSE does not penalize large number of clusters
  - SSE decreases for increasing cluster counts
  - Applicable beyond K-Means, but less intuitive interpretation (in case of non-globular clusters)



# Quick Quiz

10      1,000

If  $K \ll N$ , can SSE=0?

**Why** or **why not?**

$K >$  unique points

A

Yes

B

No

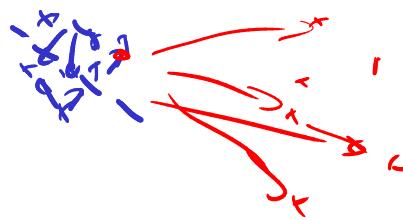
# Internal Quality Measures — Silhouette Coefficient

- Intuition: A good clustering has
  - High inter-cluster distances
  - Low intra-cluster distances

- For each data point  $x$ , define

- Cohesion**  $a(x)$ : average distance to points in the same cluster
- Separation**  $b(x)$ : minimum average distance to points in a different cluster
- Silhouette:**

$$s(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}}, \text{ if } |C_x| > 1$$



$$x \in C_X$$

$$a(x) = \frac{1}{|C_X - 1|} \sum_{p \in C_X, p \neq x} d(x, p)$$

the smaller, the better

$$b(x) = \min_{X \neq K} \frac{1}{|C_K|} \sum_{p \in C_K} d(x, p)$$

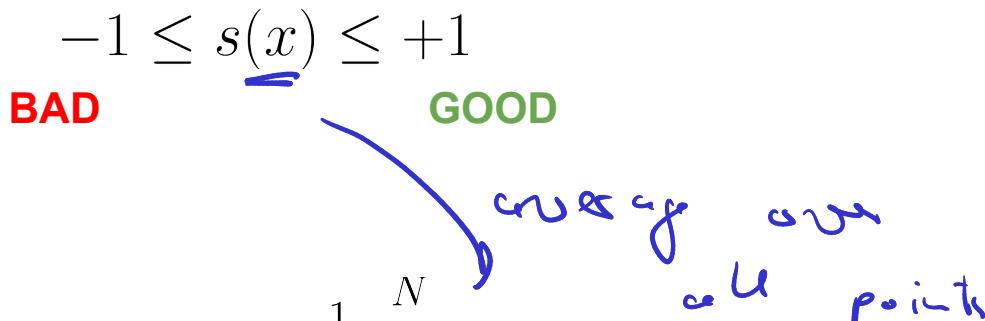
the larger, the better

$$s(x) = 0, \text{ if } |C_x| = 1$$

# Internal Quality Measures — Silhouette Coefficient

$$s(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}}$$

- Interpretation

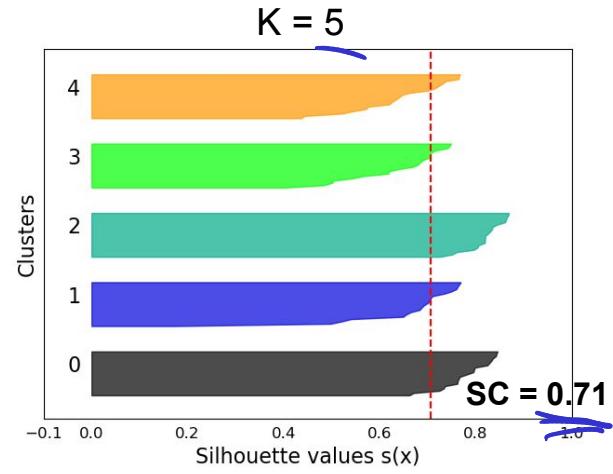
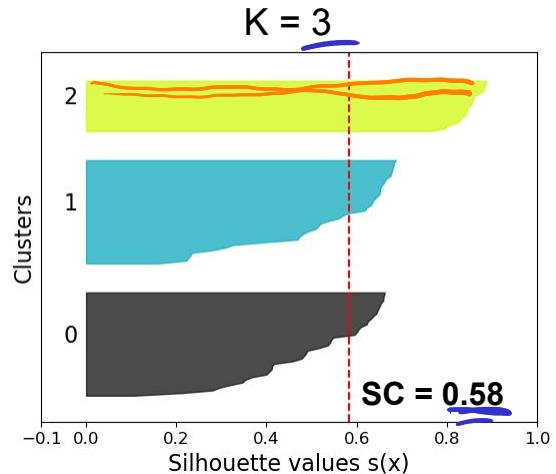
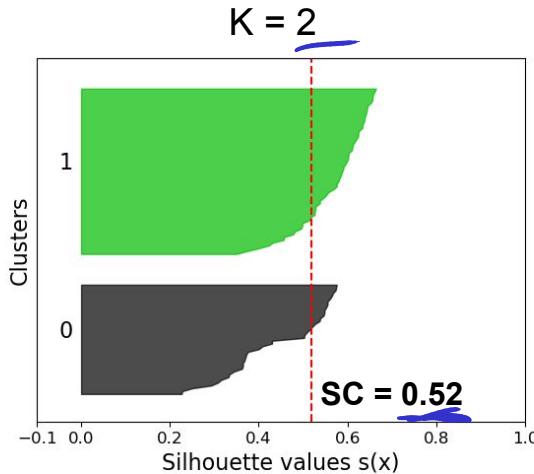
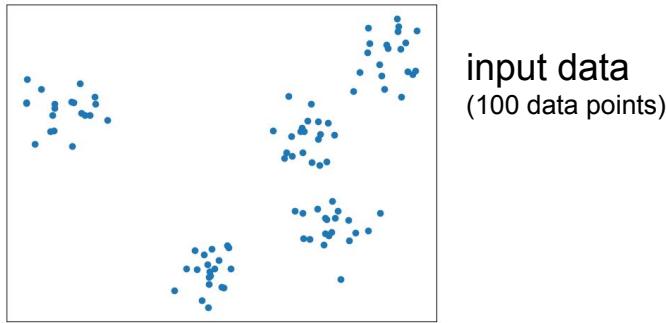


- Silhouette Coefficient SC:

$$SC = \frac{1}{N} \sum_{i=1}^N s(x_i)$$

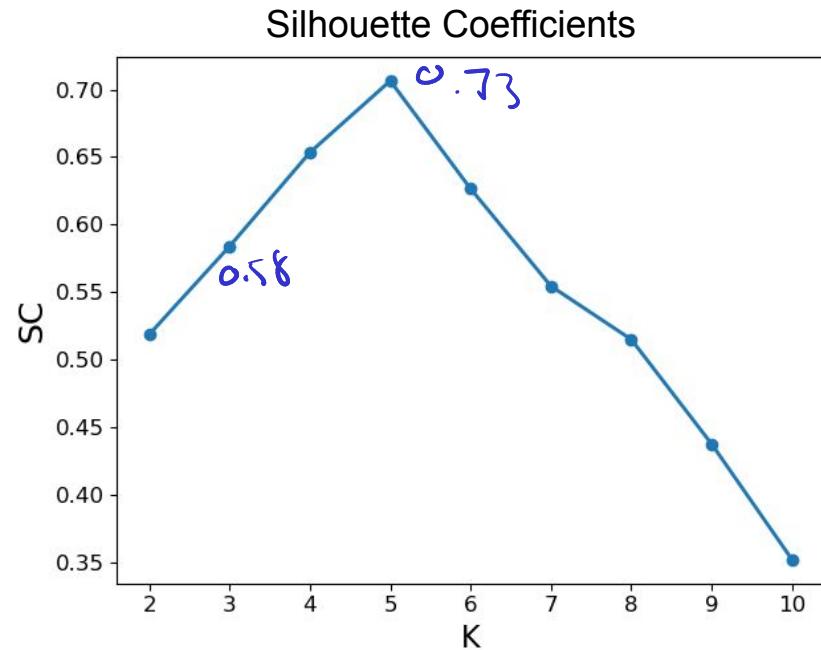
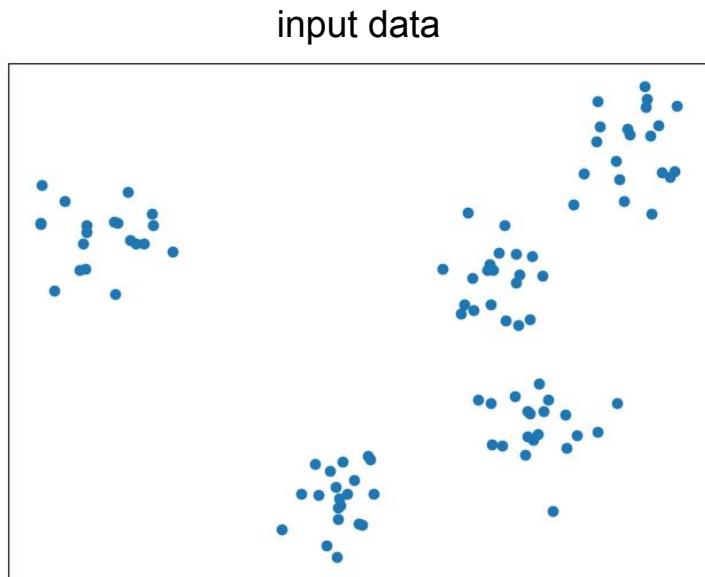
# Internal Quality Measures — Silhouette Coefficient

- Example: K-Means



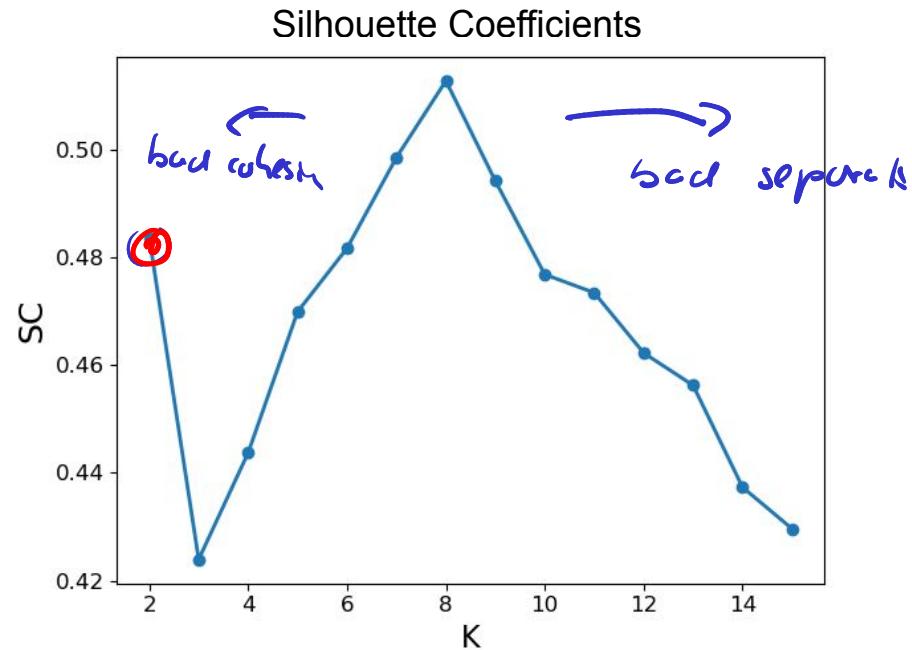
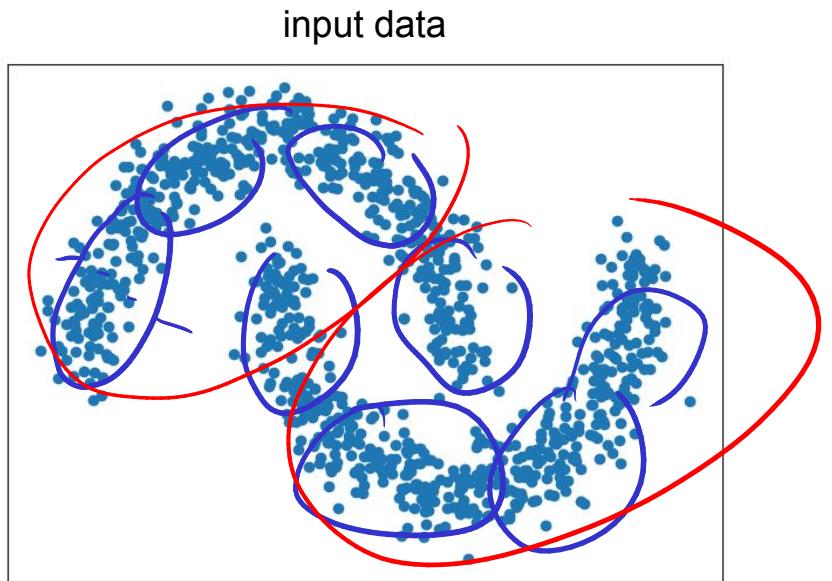
# Internal Quality Measures — Silhouette Coefficient

- Example: K-Means



# Internal Quality Measures — Silhouette Coefficient

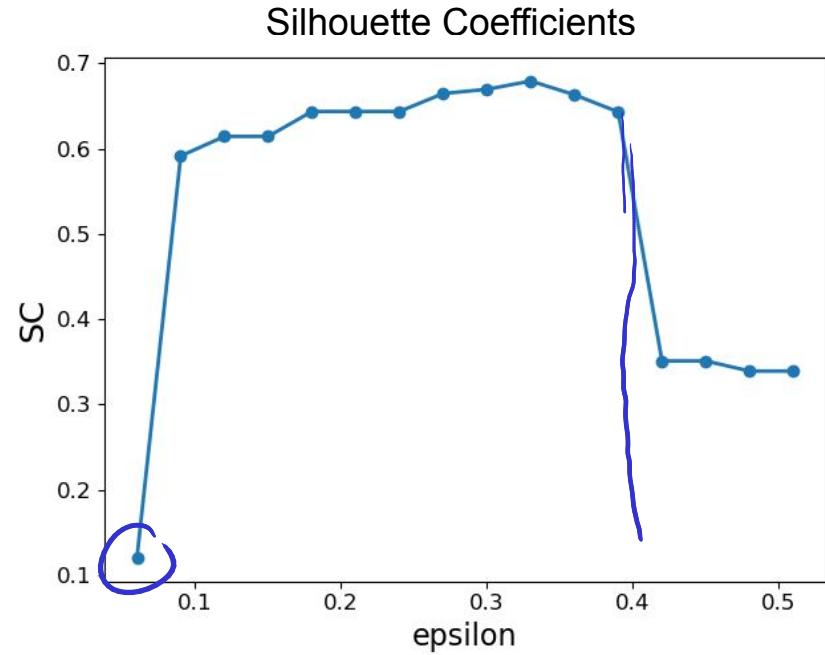
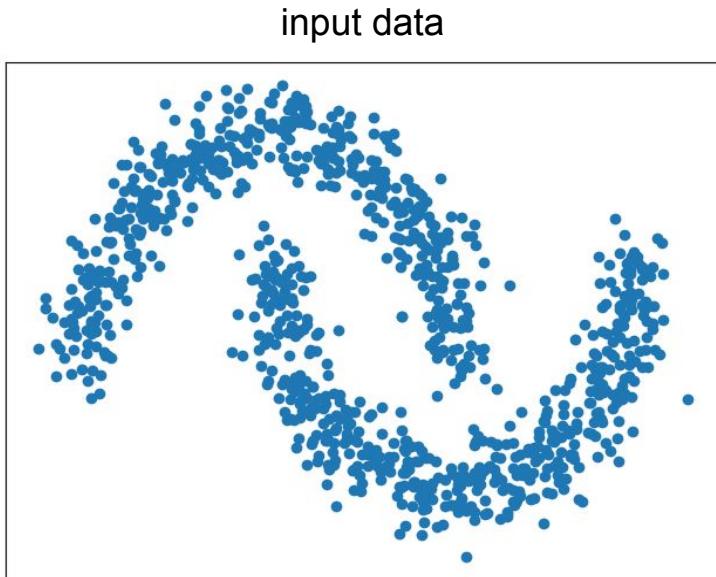
- Example: K-Means



# Internal Quality Measures — Silhouette Coefficient

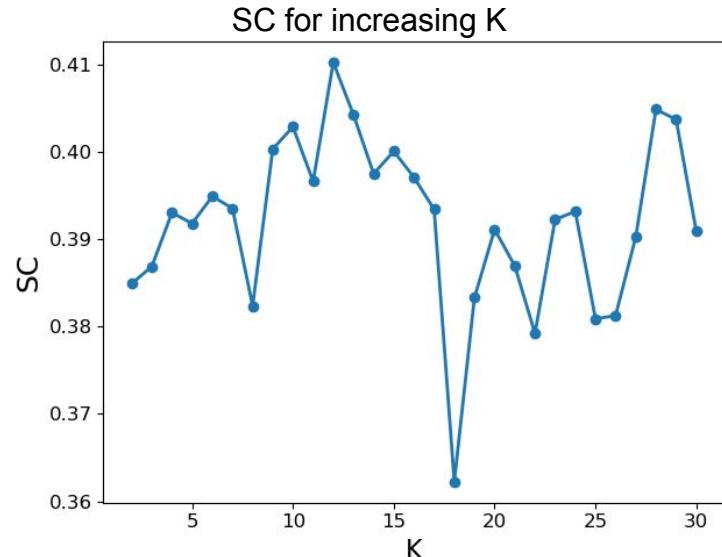
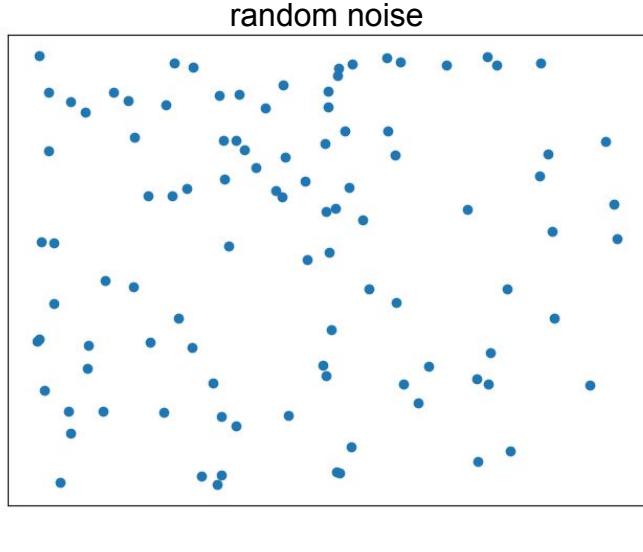
- Example: DBSCAN

$$\mu_{\text{in}} \text{ PIS} = 10$$



# Internal Quality Measures — Silhouette Coefficient

- SC for random data (K-Means)



**Note:** DBSCAN on random data quickly results in 0 or 1 cluster, for which SC is not defined

# Cluster Evaluation — Comments

- In practice, choice of "best" clustering often more pragmatic:
  - Fixed number of clusters (problematic for DBSCAN)
  - Parameters defined by tasks  
(e.g., "areas with more than 5 McDonalds within a radius of 500m")
  - Maximum, minimum, or average size of clusters
  - Focus in individual clusters instead of whole clustering  
(e.g., biggest/smallest cluster, cluster that contains certain points)
  - Set  $K$  "too high" and merge later if needed
  - ...

# Outline

- Clustering
  - Overview
  - Concepts
  - Applications
- Clustering algorithms
  - K-Means
  - DBSCAN
  - Hierarchical Clustering
- Cluster Evaluation

# Summary — Clustering

- **Clustering:** Finding patterns (here: cluster/groups) in unlabeled data
  - Very important concept in data mining
  - Wide range of clustering algorithms with varying characteristics (pros & cons)  
→ No "one-size-fits-all" algorithm
- **Discussed algorithms:** K-Means, DBSCAN, AGNES
  - Focus on the — arguably intuitive — conceptual inner workings
  - Emphasis on algorithms' strength and weaknesses
  - Many tweaks and optimizations to improve performance
- **Major challenge:** cluster evaluation
  - No fool-proof method to find the best algorithm or parameters (at least for unlabeled data)

# Solutions to Quick Quizzes

- Slide 11: A
- Slide 15: A and/or B
- Slide 26: A
- Slide 46: A (in case of duplicates and  $K < \#\text{unique points}$ )

# **CS5228: Knowledge Discovery and Data Mining**

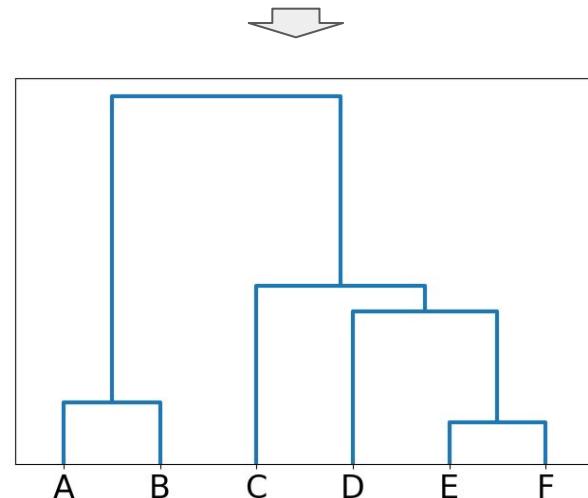
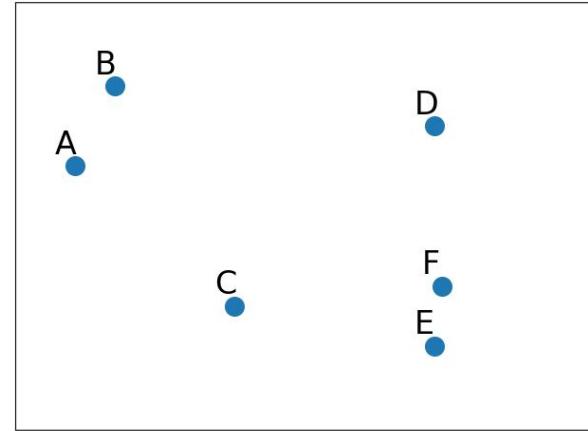
## Lecture 4 — Association Rule Mining

# Course Logistics — Update

- Assignment 1
  - Submission deadline: Thu, Sep 12 (11.59 pm)
  - Honor code: don't cheat, don't copy, don't steal, don't plagiarize, etc.
  - Don't forget to check Discussion and Errata page Canvas
- Project
  - Teams finalized (please get all in contact)
  - Kaggle competition launched

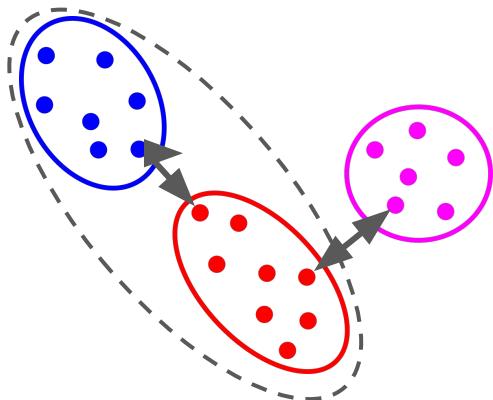
# Recap — Hierarchical Clustering

- AGNES (AGglomerative NESting)
  - Start with  $N$  clusters, one for each data point
  - Iteratively merge nearest clusters into one
  - Stop if all data points are in one cluster
- Core questions: How to calculate distances between clusters?

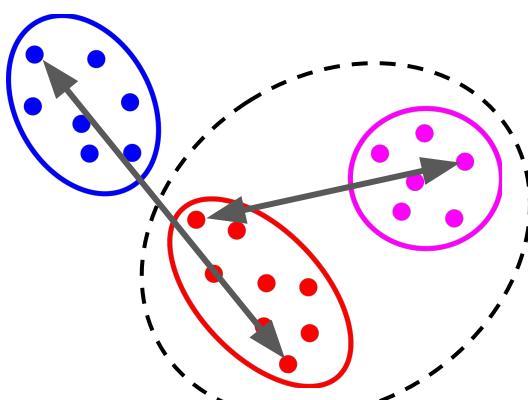


# Recap — Linkage Methods

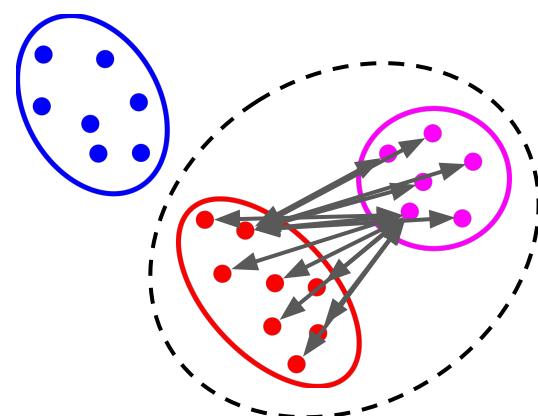
Single Linkage



Complete Linkage



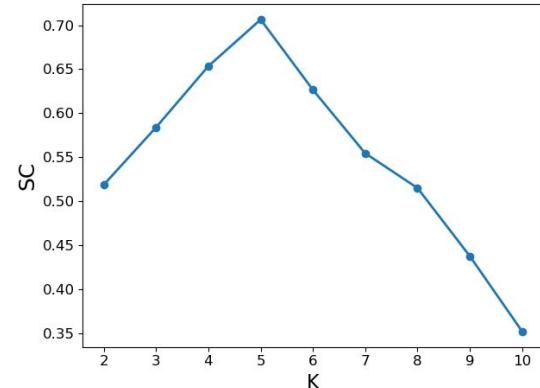
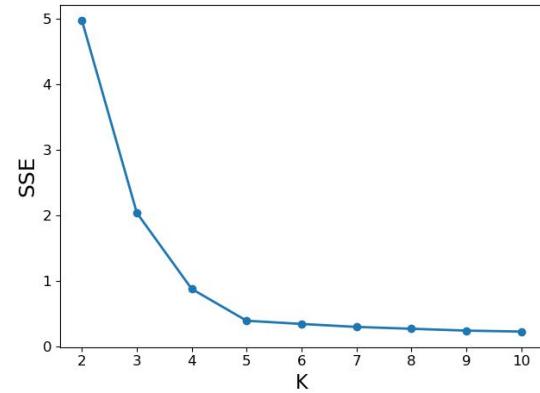
Average Linkage



# Recap — Cluster Evaluation

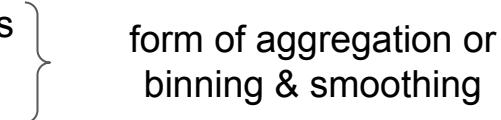
- If ground truth available: external quality measures
  - Cluster purity
  - TP/TN/FP/FN-based metrics (e.g., Rand index)
- Unlabelled data: internal quality measures
  - Elbow method using SSE
  - Silhouette Coefficient (SC)

} favor blob-like clusters
- Cluster evaluation in practice (unlabeled data)
  - No fool-proof method to find "best" clustering
  - Decision on clustering often rather pragmatic



# Recap — Clustering as a Means to an End

- Clustering as part of EDA
  - SSE plot, SC plot, dendrogram, etc. can provide useful insights into the data
  - Little requirements — "only" similarity/distance between data points needed
  - In the gray area between (simple) EDA and proper data analysis

- Clustering for data preprocessing — example:
    - Cluster persons according to their height into K=10 groups
    - Assign each person new height = centroid of cluster
- 
- form of aggregation or  
binning & smoothing

# Outline

- **Association Rule Mining**
  - Overview
  - Applications
- Definitions
- Algorithms
  - Brute-Force
  - Apriori
- Discussion

# Association Rules — Basic Setup

- Input database:
  - Set of **transactions**
  - Transaction = set of **items**
- Output: **Association Rules**
  - Rules predicting the occurrence of some items based on occurrence of other items

**antecedent → consequent**

$$\{\text{item}_2, \text{item}_3\} \rightarrow \{\text{item}_5\}$$

$$\{\text{item}_1\} \rightarrow \{\text{item}_3\}$$

| TID | Items                                                                                             |
|-----|---------------------------------------------------------------------------------------------------|
| 1   | item <sub>1</sub> , item <sub>2</sub> , item <sub>3</sub> , item <sub>4</sub> , item <sub>5</sub> |
| 2   | item <sub>2</sub> , item <sub>3</sub> , item <sub>5</sub>                                         |
| 3   | item <sub>1</sub> , item <sub>4</sub> , item <sub>5</sub>                                         |
| 4   | item <sub>2</sub> , item <sub>3</sub> , item <sub>5</sub> , item <sub>6</sub> , item <sub>7</sub> |
| 5   | item <sub>1</sub> , item <sub>3</sub> , item <sub>5</sub> , item <sub>7</sub>                     |
| ... |                                                                                                   |

# Applications — Market Basket Analysis

- Understanding customers shopping behavior
  - **Items:** products in supermarket/store
  - **Transaction:** baskets at check-out
- Interesting rules:
  - Customers who buy {a, b} also tend to buy {x, y}
  - Example: {cereal} → {milk}
- Purpose
  - Shelf management / item placement
  - Promotions (product bundles)
  - Recommendations
  - Pricing strategies

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |
| ... |                              |

# Applications — Medical Data Analysis

- Diagnosis Support Systems

- Items: symptoms, diseases
- Transaction: patient's medical history

| ID  | Items                                       |
|-----|---------------------------------------------|
| 1   | covid-19, anosmia, cough, fatigue           |
| 2   | flu, anosmia, headache                      |
| 3   | covid-19, anosmia, headache, fatigue, fever |
| 4   | covid-19, flu, anosmia, fatigue             |
| 5   | flu, depression, fatigue, fever, headache   |
| ... |                                             |



{anosmia, fatigue} → {covid-19}

- ADR discovery (adverse drug reaction)

- Items: drugs, reactions/symptoms
- Transaction: patient's medical history

| ID  | Items                                      |
|-----|--------------------------------------------|
| 1   | $d_1, d_2, d_3, \text{rash, vomit}$        |
| 2   | $d_1, d_3, \text{headache, nausea, rash,}$ |
| 3   | $d_2, d_3, \text{nausea, vomit}$           |
| 4   | $d_1, \text{nausea, rash, vomit}$          |
| 5   | $d_3, d_4, \text{headache, depression}$    |
| ... |                                            |



{ $d_1$ } → {rash}

# Applications — Census Data Analysis

- Getting insights into a population
  - Items: demographic data
  - Transaction: census record
- Interesting rules:
  - Correlations among groups of people based on shared demographics
  - Example: {uni-grad,  $\geq 30$ }  $\rightarrow$  {high-income}
- Purpose
  - Policy & decision making
  - Resource allocation
  - Urban planning

| TID | Items                                                     |
|-----|-----------------------------------------------------------|
| 1   | female, $\geq 25$ , uni-grad, hdb, single, high-income    |
| 2   | male, $\geq 25$ , uni-grad, hdb, single, mid-income       |
| 3   | male, $\geq 25$ , uni-grad, hdb, condo, high-income       |
| 4   | male, $\geq 30$ , uni-grad, condo, married, high-income   |
| 5   | female, $\geq 30$ , uni-grad, condo, married, high-income |
| ... |                                                           |

# Applications — Behavior Data Analysis

- User preferences & linkings
  - Items: movies, songs, books, etc.
  - Transaction: viewing/listening/reading history
- Interesting rules (movies):
  - Viewer who watched movies {a, b} also watched movies {x, y}
  - Example: {Jaws}→{It}
- Purpose
  - Recommendation systems

| TID | Items                          |
|-----|--------------------------------|
| 1   | Jaws, Halloween, Scream, It    |
| 2   | Alien, Jaws, Scream, It        |
| 3   | Tenet, Inception, Interstellar |
| 4   | Jaws, Halloween, It            |
| 5   | Alien, Tenet, Jaws, It         |
| ... |                                |

# Association Rules — Problem Statement

- Association rules are not "hard" rules
  - e.g., {cereal}→{milk} does not mean that customers always buy milk when buying cereal
  - each possible combination (e.g., {yogurt, bread}→{milk}) is potential association rule
- Given  $d$  unique items →  $\underline{3^d - 2^{d+1} + 1}$  rules
  - $d = 6 \rightarrow 602$  possible rules!
- Association Rule Mining
  - Finding **interesting/significant** association rules
  - Finding such rules **efficiently**

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |
| ... |                              |

*milk → eggs  
eggs, bread → yogurt*

# Outline

- **Association Rule Mining**
  - Overview
  - Applications
- **Definitions**
- **Algorithms**
  - Brute-Force
  - A-Priori
- **Discussion & Summary**

# Definitions — Itemset, K-itemset

- **Itemset**

- A subset of items

{bread}, {yogurt}, {bread, yogurt}, {milk}, {cereal},  
{eggs}, {bread, milk}, {bread, milk, cereal}, ...

- **K-itemset**

- An itemset containing k items, e.g., k=3:

{bread, milk, cereal}, {bread, yogurt, cheese},  
{yogurt, milk, cereal}, {yogurt, cereal, cheese},  
{milk, cereal, cheese}, {bread, milk, eggs}, ...

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

# Definitions — Support Count, Support (for itemsets)

- **Support count SC**
  - Number of transactions containing an itemset
  - e.g.,  $SC(\{\text{bread, yogurt, milk}\}) = 2$
- **Support S**
  - Fraction of transactions containing an itemset
  - e.g.,  $S(\{\text{bread, yogurt, milk}\}) = 2/5$

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

# Definitions — Frequent Itemset

- **Frequent itemset**

- Itemset with a support greater or equal than a minimum threshold  $minsup$
- e.g., all frequent itemsets if

$$minsup = 2/5$$

{yogurt}  
{milk}  
{cheese}  
{cereal}  
{bread}  
{bread, milk}  
{yogurt, milk}  
{bread, cereal}  
{cereal, milk}  
{bread, yogurt}  
{cereal, yogurt}  
{cereal, yogurt, milk}  
{bread, cereal, milk}  
{bread, yogurt, milk}

$$minsup = 3/5$$

{yogurt}  
{milk}  
{cereal}  
{bread}  
{bread, milk}  
{yogurt, milk}  
{cereal, milk}  
{bread, yogurt}

}

3-itemset

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

# Definitions — Association Rule

- **Association Rule**

- Implication expression  $X \rightarrow Y$ ,  
where X and Y are itemsets
- e.g.,  $\{\text{yogurt, milk}\} \rightarrow \{\text{bread}\}$

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

# Definitions — Support (for association rules)

- **Support of an association rule**

- Fraction of transactions containing all items of an association rule  $X \rightarrow Y$

$$S(X \rightarrow Y) = \frac{SC(\underline{X \cup Y})}{N} = S(X \cup Y)$$

#transactions

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |
| ... |                              |

$$S(\{yogurt, milk\} \rightarrow \{bread\}) = \frac{SC(\{yogurt, milk, bread\})}{N} = 2/5$$

≠

=

$$S(\{yogurt, bread\} \rightarrow \{milk\}) = \frac{SC(\{yogurt, milk, bread\})}{N} = 2/5$$

# Definitions — Confidence

- **Confidence** of an association rule  $X \rightarrow Y$

- Probability of  $Y$  given  $X$

$$C(X \rightarrow Y) = \frac{S(X \rightarrow Y)}{S(X)} = \frac{S(X \cup Y)}{S(X)}$$

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

$$C(\{yogurt, milk\} \rightarrow \{bread\}) = \frac{S(\{yogurt, milk, bread\})}{S(\{yogurt, milk\})} = 2/3$$

$$\neq C(\{m, b\} \rightarrow \{y\})$$

# High Support, High Confidence → Interesting Rules

| $X \rightarrow Y$ | Low Support                                                                                                                                                                                                    | High Support                                                                                                                                                                                       |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Low Confidence    | <ul style="list-style-type: none"><li>The items in <math>(X \cup Y)</math> do not frequently appear together</li><li>Even if the items in X appear together, they do so often without the items in Y</li></ul> | <ul style="list-style-type: none"><li>The items in <math>(X \cup Y)</math> frequently appear together</li><li>If the items in X appear together, they often do so without the items in Y</li></ul> |
| High Confidence   | <ul style="list-style-type: none"><li>The items in <math>(X \cup Y)</math> do not frequently appear together</li><li>If the items in X appear together, they often do so with the items in Y</li></ul>         | <ul style="list-style-type: none"><li>The items in <math>(X \cup Y)</math> frequently appear together</li><li>If the items in X appear together, they do so often with the items in Y</li></ul>    |

# Quick Quiz

Given an association rule R, **which inequality** regarding the support S(R) and confidence C(R) **holds**?

$$R \hat{=} X \rightarrow Y$$

$$S(R) = S(X \cup Y)$$

$$C(R) = \frac{S(X \cup Y)}{S(X)}$$

**A**

$$S(R) > C(R)$$

**B**

$$S(R) \geq C(R)$$

**C**

✓  $S(R) \leq C(R)$

**D**

$$S(R) < C(R)$$

# Outline

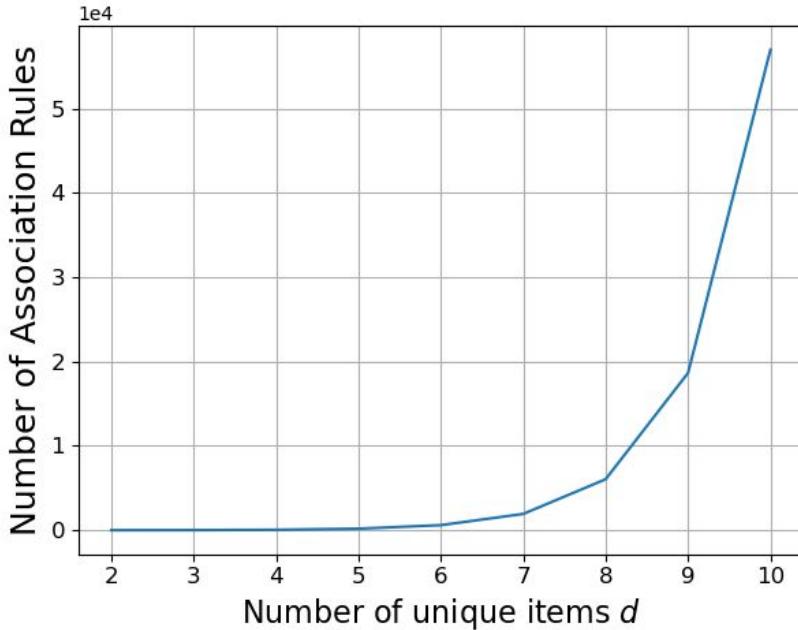
- Association Rule Mining
  - Overview
  - Applications
- Definitions
- Algorithms
  - Brute-Force
  - A-Priori
- Discussion & Summary

# Brute Force Approach — Algorithm

- Given a set of transactions,  
find all association rules  $X \rightarrow Y$  with
  - Support  $S(X \rightarrow Y) \geq \text{minsup}$
  - Confidence  $C(X \rightarrow Y) \geq \text{minconf}$
- Brute force algorithm
  - List all possible association rules  $X \rightarrow Y$
  - Calculate support  $S(X \rightarrow Y)$  and confidence  $C(X \rightarrow Y)$  for each rule
  - Drop rules with  $S(X \rightarrow Y) < \text{minsup}$  and  $C(X \rightarrow Y) < \text{minconf}$

# Brute Force Approach — Computation Complexity

- Given  $d$  unique items  $\rightarrow 3^d - 2^{d+1} + 1 \in O(3^d)$  rules
  - $d = 6 \rightarrow 602$  (theoretically) possible rules!



Average number items carried in a supermarket in 2019

Source: FMI

**28,112**

<https://www.fmi.org/our-research/supermarket-facts>

# Brute Force Approach — Computation Complexity

- Let  $w$  be the maximum number of items in a transaction within the database
  - $N = 5, w = 4 \rightarrow \leq 250$  "available" rules!



The difference between 250 and 602 seems negligible, but this is only because in this toy example,  $d = 6$  and  $w = 4$  are of the same magnitude.

The number 250 also ignores duplicate rules.

$$\rightarrow O(N \cdot (3^w - 2^{w+1} + 1)) \text{ rules}$$

(typically  $w \ll d$ )

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

$N$

$w$

True number of different rules: 154



# Decoupling Support and Confidence

- Recall  $S(X \rightarrow Y) = \frac{SC(X \cup Y)}{N} = S(X \cup Y)$

$$\left. \begin{array}{l} S(\{yogurt, milk\} \rightarrow \{bread\}) \\ S(\{yogurt, bread\} \rightarrow \{milk\}) \\ S(\{milk, bread\} \rightarrow \{yogurt\}) \end{array} \right\} = \frac{SC(\{yogurt, milk, bread\})}{N} = S(\{yogurt, milk, bread\})$$

## • Observation 1

- A rule  $X \rightarrow Y$  has only sufficient support if  $X \cup Y$  is a frequent itemset
- No need to calculate confidence of rules where  $X \cup Y$  is not a frequent item set

$$S(X \rightarrow Y) \geq \text{minsup} \iff S(X \cup Y) \geq \text{minsup}$$

# Two-Part Algorithm for Mining Association Rules

- Part 1 — Frequent Itemset Generation
  - Generate itemsets with support  $\geq \text{minsup}$
  - "Only"  $2^d - 1$  possible itemsets to check
- Part 2: — Association Rule Generation
  - Generate rules from frequent itemsets
  - Return rules with confidence  $\geq \text{minconf}$

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |



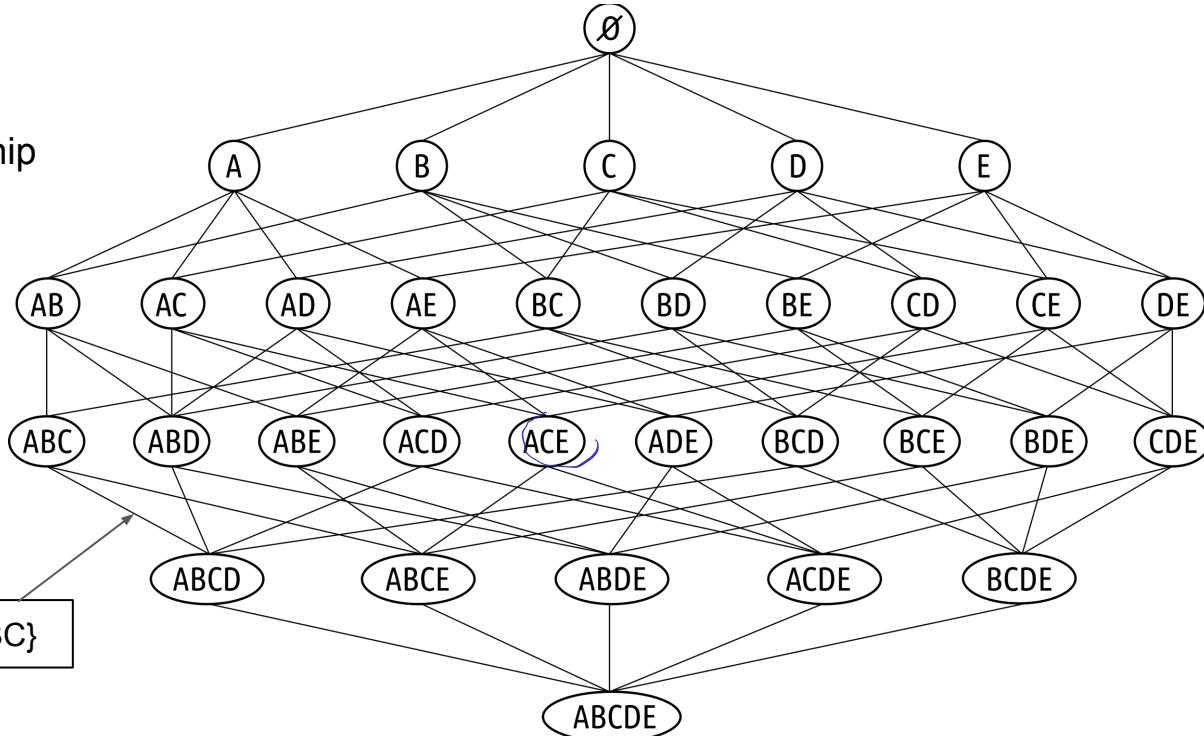
✓ **Frequent itemsets:**  
 $\{\text{milk}\}$ ,  $\{\text{cereal, milk}\}$ ,  $\{\text{bread, milk}\}$ , ...



✓ **Association rules:**  
 $\{\text{cereal}\} \rightarrow \{\text{milk}\}$

# Frequent Itemset Generation

- Itemset lattice
  - Node: itemset
  - Edge: containment relationship
- $2^d - 1$  nodes/itemsets
  - $d=5 \rightarrow 31$  itemsets



# Frequent Itemset Generation — Brute Force Algorithm

```
support_counts ← dict({})  
for each transaction  $t$  in database:  
    for  $k$  in  $1..(t.length)$ :  
         $k\_itemsets$  ← generate_itemsets( $t, k$ )  
        for each itemset in  $k\_itemsets$ :  
            support_counts[itemset] += 1
```

Global counter for all found itemsets

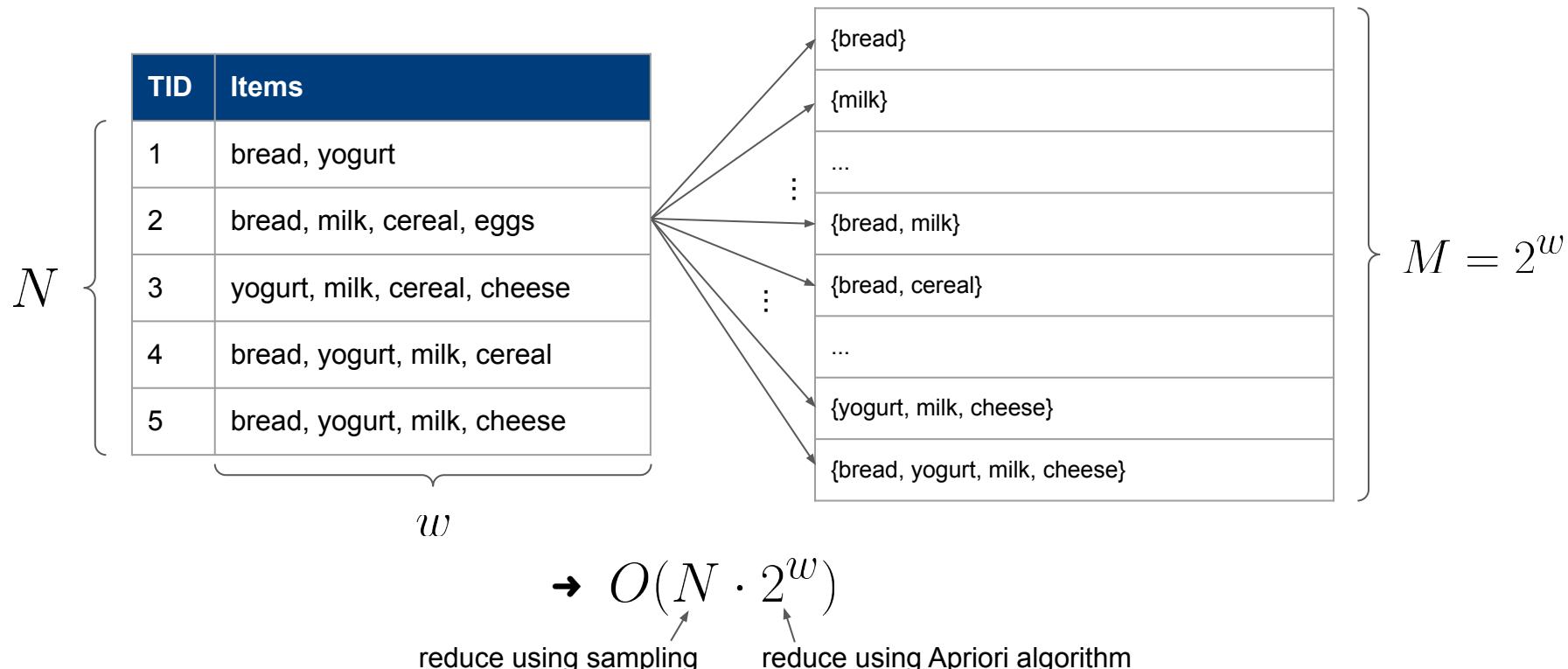
For each transaction, generate k-itemsets, with  
 $k = 1, 2, 3, \dots$  (up to #items in transaction)

For k-itemset, increase its global counter by 1

**Question:** Why do we need to count 1-itemsets  
if an association rule requires at least 2 items?

# Frequent Itemset Generation — Brute Force Algorithm

- Complexity Analysis



# Outline

- Association Rule Mining
  - Overview
  - Applications
- Definitions
- Algorithms
  - Brute-Force
  - A-Priori
- Discussion & Summary

# Apriori Principle (Anti-Monotonicity Principle)

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |
| ... |                              |

| TID | Items                                       |
|-----|---------------------------------------------|
| 1   | bread, yogurt                               |
| 2   | bread, <u>milk</u> , cereal, eggs           |
| 3   | yogurt, milk, cereal, cheese                |
| 4   | bread, <u>yogurt</u> , <u>milk</u> , cereal |
| 5   | bread, <u>yogurt</u> , <u>milk</u> , cheese |
| ... |                                             |

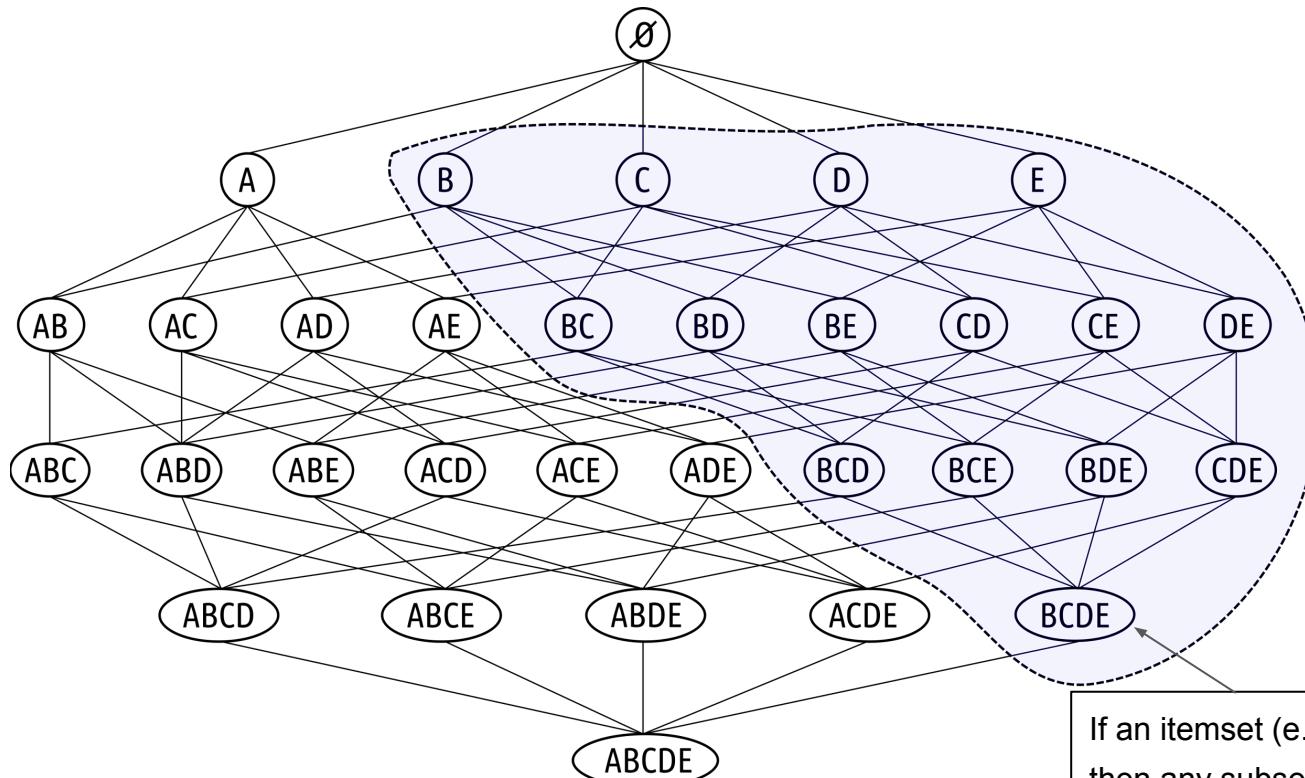
- **Observation 2:** If X and Y are itemsets and  $X \subseteq Y$ , then

- $S(X) \geq S(Y)$
- If Y is frequent, then X is frequent
- If X is not frequent, then Y is not frequent

$$S(\{\text{bread, yogurt}\}) = 3/5$$

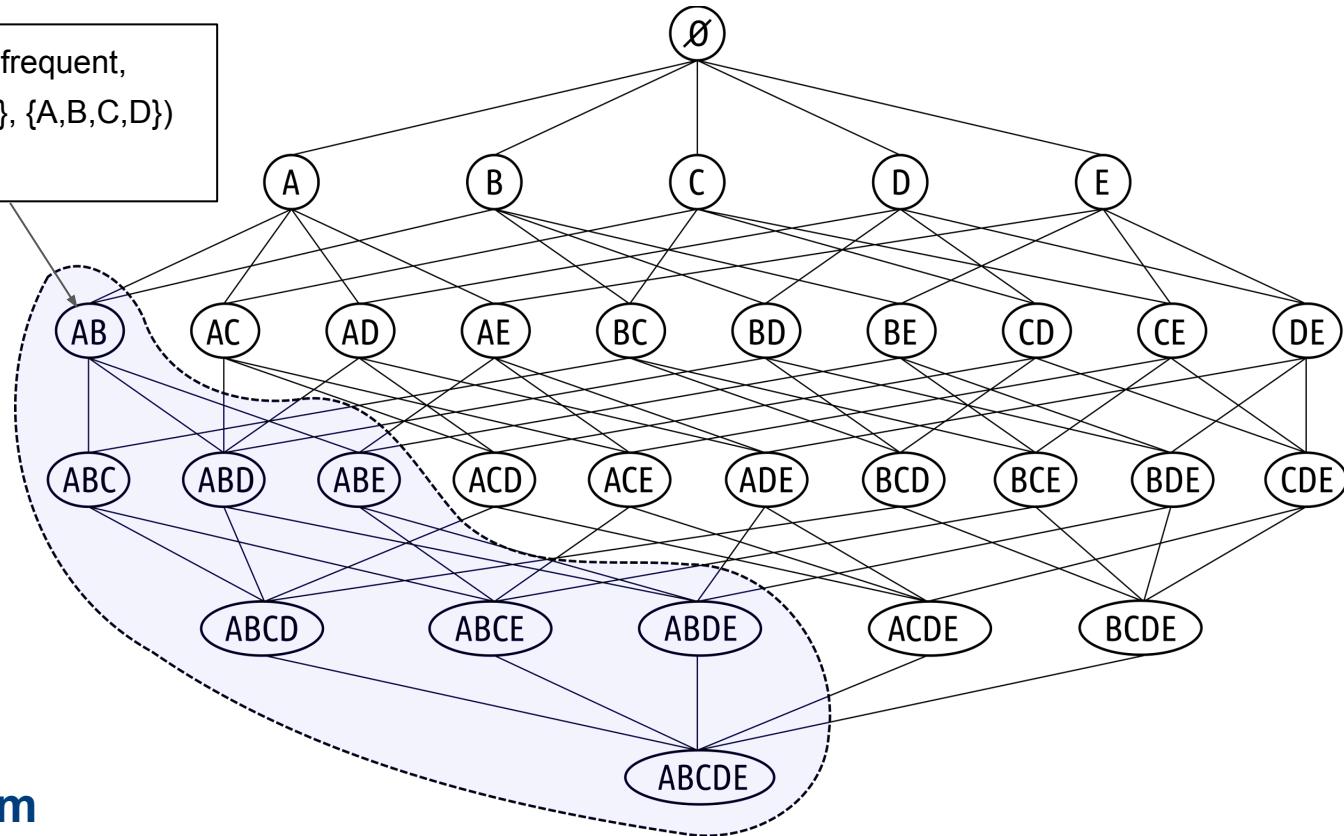
$$S(\{\text{bread, yogurt, milk}\}) = 2/5$$

# Apriori Principle (Anti-Monotonicity Principle)



# Apriori Principle (Anti-Monotonicity Principle)

If an itemset (e.g., {A,B}) is not frequent, then any superset (e.g., {A,B,D}, {A,B,C,D}) is also not frequent



→ Apriori Algorithm

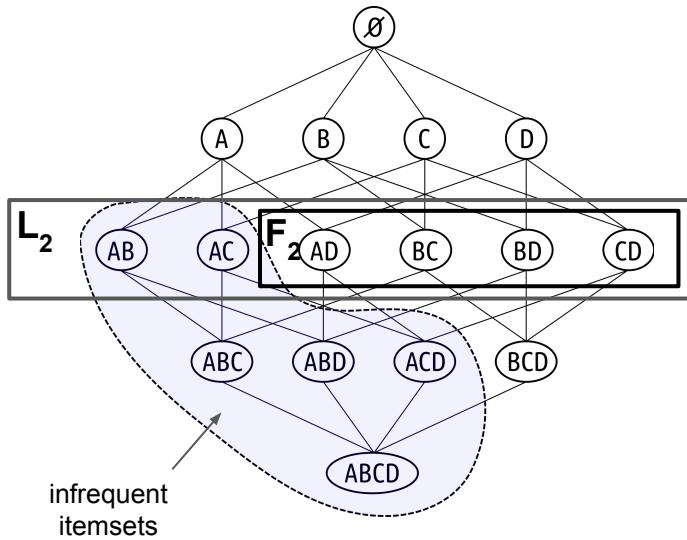
# Apriori Algorithm

- Notations

- $L_k$  — candidate k-itemsets
- $F_k$  — frequent k-itemsets ( $F_k \subseteq L_k$ )

For  $k$  in  $1..w$ :

- **Generate**  $L_k$  from  $F_{k-1}$
- **Prune** k-itemsets from  $L_k$  using  $F_{k-1}$
- **Calculate** SC for remaining  $L_k$  itemsets
- **Filter**  $L_k$  itemsets with insufficient SC  $\rightarrow F_k$
- If  $|F_k| = 0$ , stop



# Quick Quiz

Which of the four steps is generally the **most expensive** one?

For  $k$  in  $1..w$ :

- **Generate**  $L_k$  from  $F_{k-1}$
- **Prune**  $k$ -itemsets from  $L_k$  using  $F_{k-1}$
- **Calculate** SC for remaining  $L_k$  itemsets
- **Filter**  $L_k$  itemsets with insufficient SC →  $F_k$
- If  $|F_k| = 0$ , stop

A

Generate

B

Prune

C ✓

Calculate

D

Filter

# Apriori Algorithm

$\text{minsup} = 0.4 \rightarrow \text{minimum support count: } 2$

$$d = 6$$



Generating

| Itemset  |
|----------|
| {bread}  |
| {cereal} |
| {cheese} |
| {eggs}   |
| {milk}   |
| {yogurt} |

Calculating

| Itemset  | SC |
|----------|----|
| {bread}  | 4  |
| {cereal} | 3  |
| {cheese} | 2  |
| {eggs}   | 1  |
| {milk}   | 4  |
| {yogurt} | 4  |

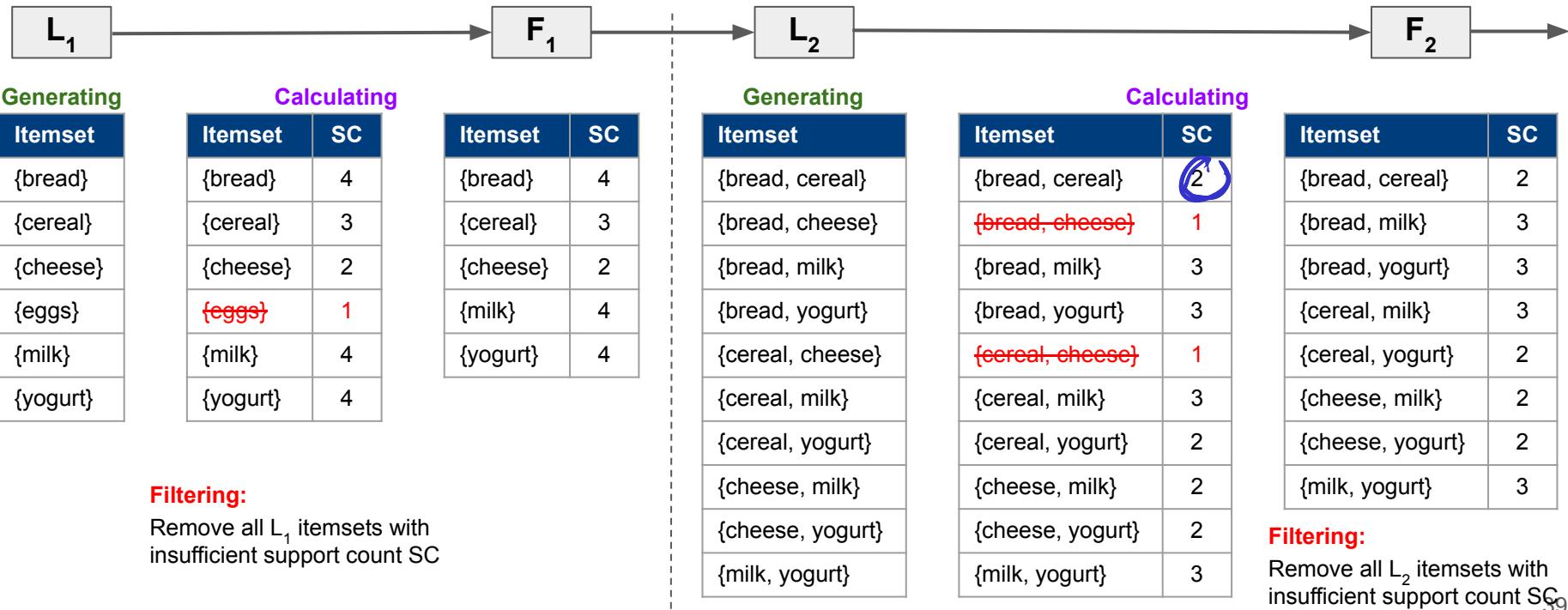
| Itemset  | SC |
|----------|----|
| {bread}  | 4  |
| {cereal} | 3  |
| {cheese} | 2  |
| {eggs}   | 1  |
| {milk}   | 4  |
| {yogurt} | 4  |

Filtering:

Remove all  $L_1$  itemsets with insufficient support count SC

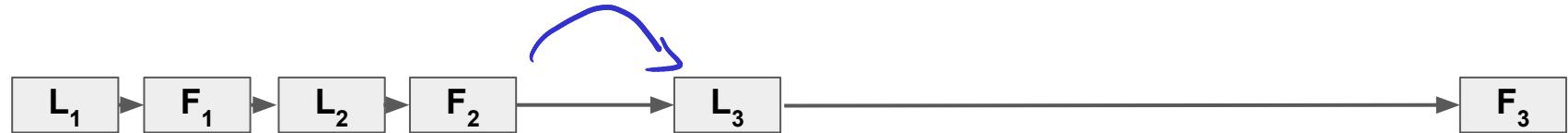
# Apriori Algorithm

$\text{minsup} = 0.4 \rightarrow \text{minimum support count: } 2$



# Apriori Algorithm

$\text{minsup} = 0.4 \rightarrow \text{minimum support count: } 2$



| k-1 itemsets    |   |
|-----------------|---|
| {bread, cheese} | ✗ |
| {bread, milk}   | ✓ |
| {cheese, milk}  | ✓ |

| Itemset                         |
|---------------------------------|
| {bread, cereal, cheese}         |
| {bread, cereal, milk}           |
| {bread, cereal, yogurt}         |
| <u>{bread, cheese, milk}</u>    |
| <u>{bread, cheese, yogurt}</u>  |
| {bread, milk, yogurt}           |
| <u>{cereal, cheese, milk}</u>   |
| <u>{cereal, cheese, yogurt}</u> |
| {cereal, milk, yogurt}          |
| {cheese, milk yogurt}           |

## Pruning:

$\{\text{bread, cheese}\} \notin F_2$ ,

$\rightarrow \{\text{bread, cheese, milk}\} \notin F_3$

$\rightarrow \text{SC}(\{\text{bread, cheese, milk}\}) \text{ not needed!}$

| Itemset                        | SC |
|--------------------------------|----|
| {bread, cereal, milk}          | 2  |
| <u>{bread, cereal, yogurt}</u> | 1  |
| {bread, milk, yogurt}          | 2  |
| {cereal, milk, yogurt}         | 2  |
| {cheese, milk yogurt}          | 2  |

## Filtering:

Remove all  $L_3$  itemsets with insufficient support count SC

| Itemset                | SC |
|------------------------|----|
| {bread, cereal, milk}  | 2  |
| {bread, milk, yogurt}  | 2  |
| {cereal, milk, yogurt} | 2  |
| {cheese, milk yogurt}  | 2  |

# Apriori Algorithm

minsup = 0.4 → minimum support count: 2



Generating

| k-1 itemsets            | Itemset                        | Itemset | sc |
|-------------------------|--------------------------------|---------|----|
| {bread, cheese, milk}   | {bread, cereal, cheese, milk}  |         |    |
| {bread, cheese, yogurt} | {bread, cereal, milk, yogurt}  |         |    |
| {bread, milk, yogurt}   | {bread, cheese, milk, yogurt}  |         |    |
| {cheese, milk, yogurt}  | {cereal, cheese, milk, yogurt} |         |    |

F4 is empty → done!

## Pruning:

Only {bread, milk, yogurt} is in  $F_3$

→ {bread, cheese, milk, yogurt}  $\notin F_4$

→ SC({bread, cheese, milk, yogurt}) not needed!

# Apriori Algorithm

- Output: All frequent itemsets  $F_i$  with
  - $i \geq 2$  — cannot create rules from a single item
  - $|F_i| > 0$  — set of itemsets is not empty

$F_1$

- Implementation details
  - Generating/Pruning — How to get from  $F_{k-1}$  to  $L_k$ ?
  - Calculating — How to calculate SC for  $L_k$  itemsets efficiently?  
(not covered here as this is done on the implementation level)

| Itemset                | sc |
|------------------------|----|
| {bread, cereal}        | 2  |
| {bread, milk}          | 3  |
| {bread, yogurt}        | 3  |
| {cereal, milk}         | 3  |
| {cereal, yogurt}       | 2  |
| {cheese, milk}         | 2  |
| {cheese, yogurt}       | 2  |
| {milk, yogurt}         | 3  |
| {bread, cereal, milk}  | 2  |
| {bread, milk, yogurt}  | 2  |
| {cereal, milk, yogurt} | 2  |
| {cheese, milk yogurt}  | 2  |

$F_2$

$F_3$

# Generating/Pruning: $F_{k-1} \times F_1$ Method

$F_2$ : frequent 2-itemsets

| Itemset          |
|------------------|
| {bread, cereal}  |
| {bread, milk}    |
| {bread, yogurt}  |
| {cereal, milk}   |
| {cereal, yogurt} |
| {cheese, milk}   |
| {cheese, yogurt} |
| {milk, yogurt}   |

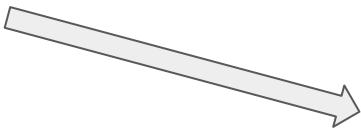
$F_1$ : frequent 1-itemsets

| Itemset  |
|----------|
| {bread}  |
| {cereal} |
| {cheese} |
| {milk}   |
| {yogurt} |

$$\{bread, cereal\} \cup \{bread\} = \{bread, cereal\}$$

## Generating:

Merge frequent  $(k-1)$ -itemsets and frequent 1-itemsets to get all possible  $k$ -itemsets



## L<sub>3</sub>: 3-itemsets

| Itemset                  |
|--------------------------|
| {bread, cereal, cheese}  |
| {bread, cereal, milk}    |
| {bread, cereal, yogurt}  |
| {bread, cheese, milk}    |
| {bread, cheese, yogurt}  |
| {bread, milk, yogurt}    |
| {cereal, cheese, milk}   |
| {cereal, cheese, yogurt} |
| {cereal, milk, yogurt}   |
| {cheese, yogurt, milk}   |

## Pruning:

Delete all  $k$ -itemsets with at least one containing  $(k-1)$ -itemset not in  $F_{k-1}$



## L<sub>3</sub>: 3-itemsets (pruned)

| Itemset                 |
|-------------------------|
| {bread, cereal, milk}   |
| {bread, cereal, yogurt} |
| {bread, milk, yogurt}   |
| {cereal, milk, yogurt}  |
| {cheese, milk, yogurt}  |

||

Calculate SC

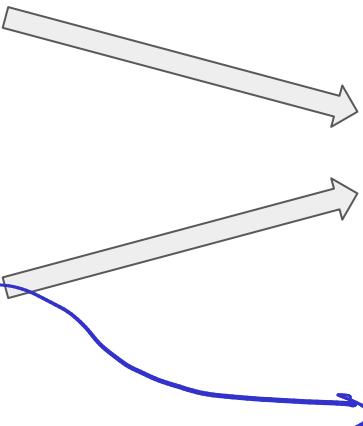
# Generating/Pruning: $F_{k-1} \times F_{k-1}$ Method

$F_2$ : frequent 2-itemsets

| Itemset          |
|------------------|
| {bread, cereal}  |
| {bread, milk}    |
| {bread, yogurt}  |
| {cereal, milk}   |
| {cereal, yogurt} |
| {cheese, milk}   |
| {cheese, yogurt} |
| {milk, yogurt}   |

Generating:

Merge frequent  $(k-1)$ -itemsets that overlap in  $(k-2)$  items to get all possible  $k$  itemsets



$L_3$ : 3-itemsets

| Itemset                  |
|--------------------------|
| {bread, cereal, milk}    |
| {bread, cheese, milk}    |
| {bread, cereal, yogurt}  |
| {bread, cheese, yogurt}  |
| {bread, milk, yogurt}    |
| {cereal, cheese, milk}   |
| {cereal, cheese, yogurt} |
| {cereal, milk, yogurt}   |
| {cheese, milk, yogurt}   |

Pruning:

Delete all  $k$ -itemsets with at least one containing  $(k-1)$ -itemset not in  $F_{k-1}$



$L_3$ : 3-itemsets (pruned)

| Itemset                 |
|-------------------------|
| {bread, cereal, milk}   |
| {bread, cereal, yogurt} |
| {bread, milk, yogurt}   |
| {cereal, milk, yogurt}  |
| {cheese, milk, yogurt}  |

Candidate SC

# Calculating Support Counts

- Calculating SC for each candidate itemset in  $L_k$ 
  - Requires **full scan** of database
  - For **each** transactions T, check for **each** itemset s if  $s \in T$
  - If  $s \in T$ , **update** counter of s

→ This is the step we want to minimize!

$L_3$ : 3-itemsets

| Itemset                 |
|-------------------------|
| {bread, cereal, milk}   |
| {bread, cereal, yogurt} |
| {bread, milk, yogurt}   |
| {cereal, milk, yogurt}  |
| {cheese, milk, yogurt}  |

Calculating

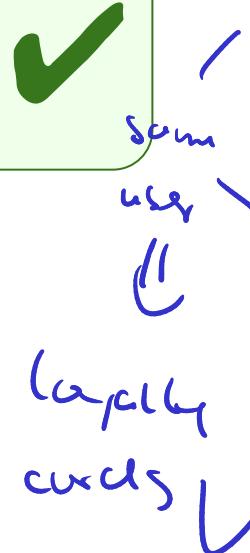
$L_3$ : 3-itemsets with SC values

| Itemset                 | SC |
|-------------------------|----|
| {bread, cereal, milk}   | 2  |
| {bread, cereal, yogurt} | 1  |
| {bread, milk, yogurt}   | 2  |
| {cereal, milk, yogurt}  | 2  |
| {cheese, milk, yogurt}  | 2  |

# Two-Part Algorithm for Mining Association Rules

- Part 1 — Frequent Itemset Generation

- General itemsets with support  $\geq \text{minsup}$
- Apriori algorithm



- Part 2: — Association Rule Generation

- Generate rules from frequent itemsets through binary partitioning of itemsets
- Return rules with confidence  $\geq \text{minconf}$

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |



Frequent itemsets:  
 $\{\text{milk}\}$ ,  $\{\text{cereal, milk}\}$ ,  $\{\text{bread, milk}\}$ , ...



Association rules:  
 $\{\text{cereal}\} \rightarrow \{\text{milk}\}$

# Rule Generation

- For each frequent itemset  $S$ , derive candidate rules  $X \rightarrow Y$ 
  - A rule is a binary split of  $s$ , i.e.,  $Y = S - X$
- For each rule  $X \rightarrow Y$ 
  - Calculate confidence  $C(X \rightarrow Y)$
  - If confidence  $\geq minconf$ , add rule to final result set

$\{\underline{A, B, C}\} \in F_3$

$\left. \begin{array}{l} \\ \\ \end{array} \right\} S$

$2^{|S|-2}$  possible rules for each frequent itemset

$\begin{array}{l} A, B \rightarrow C \\ A, C \rightarrow B \\ B \rightarrow AC \\ \vdots \end{array}$

$$C(X \rightarrow Y) = \frac{SC(X \cup Y)}{SC(X)}$$

Both values have been calculated during Frequent Itemset Generation!

- No need to access database
- Fast

# Apriori Principle (Anti-Monotonicity Principle)

$$2^3 - 2 = 6$$

$$S = \{A, B, C\}$$

$$R_1$$

$$R_2$$

- Given itemset  $S$  and two derived rules  $X_1 \rightarrow Y_1, X_2 \rightarrow Y_2$  with  $\overbrace{X_1 \cup Y_1 = X_2 \cup Y_2 = S}$

$$C(X_1 \rightarrow Y_1) = \frac{S(X_1 \cup Y_1)}{S(X_1)}$$

$$C(X_2 \rightarrow Y_2) = \frac{S(X_2 \cup Y_2)}{S(X_2)}$$

$$\underline{X_1 \subseteq X_2} \Rightarrow S(X_1) \geq S(X_2)$$

$$\Rightarrow C(X_1 \rightarrow Y_1) \leq C(X_2 \rightarrow Y_2)$$

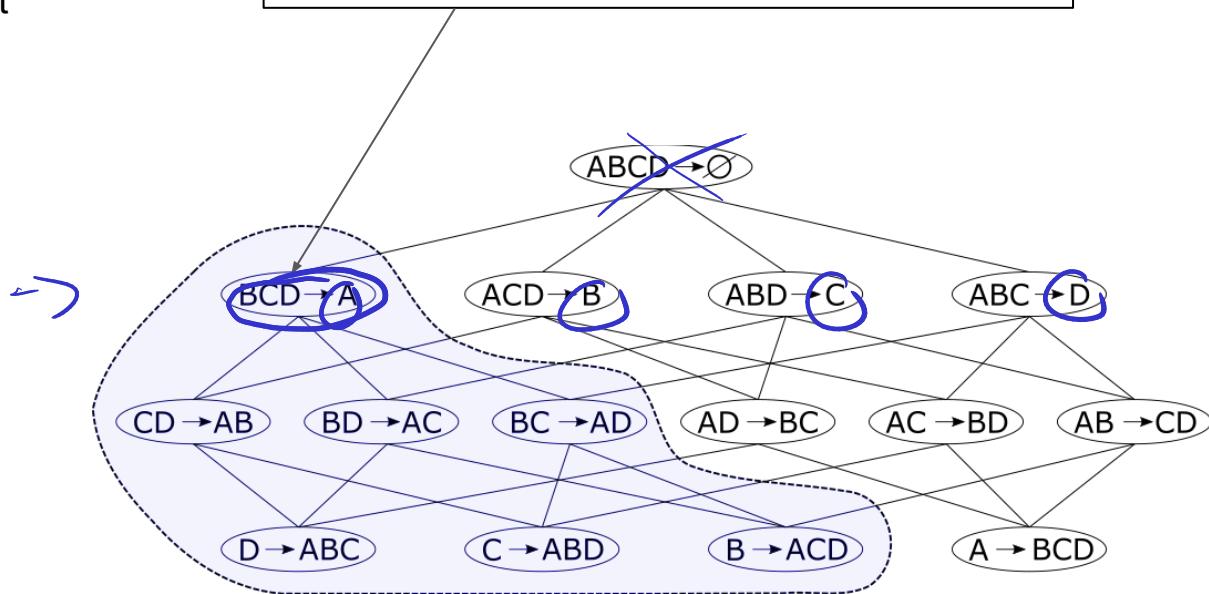
- Example: If  $\{A, B, C\} \rightarrow \{D\}$  has low confidence, so have:

- $\{A\} \rightarrow \{B, C, D\}, \{B\} \rightarrow \{A, C, D\}, \{C\} \rightarrow \{A, B, D\}, \{A, B\} \rightarrow \{C, D\}, \{A, C\} \rightarrow \{B, D\}, \{B, C\} \rightarrow \{A, D\}$

# Apriori Principle (Anti-Monotonicity Principle)

- Rule lattice
  - Node: association rule
  - Edge: containment relationship w.r.t. antecedent/consequent
- $2^{|S|}-2$  rules
  - $|S|=4 \rightarrow 14$  rules

If rule  $BCD \rightarrow A$  has a insufficient confidence,  
so do all rules containing A in the consequent



# Rule Generation Algorithm (for a single itemset S)

$$YS = \{ \{s\} \mid s \in S \}$$

repeat:

$YS\_valid = \text{evaluate}(S, YS, minconf)$

$YS = \text{generate}(YS\_valid)$

until  $|YS| = 0$

$$YS = \{ \{A\}, \{B\}, \{C\}, \{D\} \}$$

/

all right-hand side,

**evaluate**( $S, YS, minconf$ ) :

$YS\_obsolete \leftarrow \{\}$

for each  $Y$  in  $YS$ :

LHS       $X = S - Y + \text{rhs}$

if  $C(X \rightarrow Y) \geq minconf$ :

    output  $(X \rightarrow Y)$  as a valid rule

else:

$YS\_obsolete \leftarrow YS\_obsolete \cup Y$

return  $YS - YS\_obsolete$

|                             |                             |                             |                             |
|-----------------------------|-----------------------------|-----------------------------|-----------------------------|
| $\cancel{Y = \{A\}}$        | $\check{Y = \{B\}}$         | $\check{Y = \{C\}}$         | $\check{Y = \{D\}}$         |
| $\{BCD\} \rightarrow \{A\}$ | $\{ACD\} \rightarrow \{B\}$ | $\{ABD\} \rightarrow \{C\}$ | $\{ABC\} \rightarrow \{D\}$ |

$$\{ \{B\}, \{C\}, \{D\} \}$$

# Rule Generation Algorithm (for a single itemset S)

$$YS = \{ \{s\} \mid s \in S \}$$

repeat:

$YS\_valid = \text{evaluate}(S, YS, minconf)$

$YS = \text{generate}(YS\_valid)$

until  $|YS| = 0$

**evaluate**( $S, YS, minconf$ ) :

$YS\_obsolete \leftarrow \{\}$

for each  $Y$  in  $YS$ :

$X = S - Y$

if  $C(X \rightarrow Y) \geq minconf$ :

    output  $(X \rightarrow Y)$  as a valid rule

else:

$YS\_obsolete \leftarrow YS\_obsolete \cup Y$

return  $YS - YS\_obsolete$

$\{\dots\} \rightarrow \{\dots, A\}$

|                                                                               |
|-------------------------------------------------------------------------------|
| $YS\_valid = \{ \{B\}, \{C\}, \{D\} \}$                                       |
| $YS = \{ \underbrace{\{B,C\}}, \underbrace{\{B,D\}}, \underbrace{\{C,D\}} \}$ |

|                             |                             |                             |
|-----------------------------|-----------------------------|-----------------------------|
| $Y = \{B, C\}$              | $Y = \{B, D\}$              | $Y = \{C, D\}$              |
| $\{AD\} \rightarrow \{BC\}$ | $\{AC\} \rightarrow \{BD\}$ | $\{AB\} \rightarrow \{CD\}$ |

$\times \quad \checkmark \quad \times \quad \checkmark \quad \times \quad \checkmark$

|                                   |
|-----------------------------------|
| $\{ \{B,C\}, \{B,D\}, \{C,D\} \}$ |
|-----------------------------------|

# Rule Generation Algorithm (for a single itemset S)

$YS = \{ \{s\} \mid s \in S \}$

**repeat**:

$YS\_valid = \text{evaluate}(S, YS, minconf)$

$YS = \text{generate}(YS\_valid)$

**until**  $|YS| = 0$

**evaluate**( $S, YS, minconf$ ) :

$YS\_obsolete \leftarrow \{\}$

**for each**  $Y$  **in**  $YS$  :

$X = S - Y$

**if**  $C(X \rightarrow Y) \geq minconf$ :

    output  $(X \rightarrow Y)$  as a valid rule

**else**:

$YS\_obsolete \leftarrow YS\_obsolete \cup Y$

**return**  $YS - YS\_obsolete$

$YS\_valid = \{ \{B, C\}, \{B, D\}, \{C, D\} \}$

$YS = \{ \{B, C, D\} \}$

$Y = \{B, C, D\}$

$\{A\} \rightarrow \{B, C, D\}$

X      Y

$\{ \{B, C, D\} \}$

# Rule Generation Algorithm (for a single itemset S)

$$YS = \{ \{s\} \mid s \in S \}$$

repeat:

$YS\_valid = \text{evaluate}(S, YS, minconf)$

$YS = \text{generate}(YS\_valid)$

until  $|YS| = 0$

**evaluate**( $S, YS, minconf$ ) :

$YS\_obsolete \leftarrow \{\}$

    for each  $Y$  in  $YS$ :

$X = S - Y$

        if  $C(X \rightarrow Y) \geq minconf$ :

            output  $(X \rightarrow Y)$  as a valid rule

        else:

$YS\_obsolete \leftarrow YS\_obsolete \cup Y$

    return  $YS - YS\_obsolete$

|                                 |
|---------------------------------|
| $YS\_valid = \{ \{B, C, D\} \}$ |
| $YS = \{\}$                     |



**Done!**

# Two-Part Algorithm for Mining Association Rules

- Part 1 — Frequent Itemset Generation
  - General itemsets with support  $\geq \text{minsup}$
  - Apriori algorithm



- Part 2: — Association Rule Generation
  - Generate rules from frequent itemsets through binary partitioning of itemsets
  - Return rules with confidence  $\geq \text{minconf}$



| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |



Frequent itemsets:  
 $\{\text{milk}\}$ ,  $\{\text{cereal, milk}\}$ ,  $\{\text{bread, milk}\}$ , ...



Association rules.  
 $\{\text{cereal}\} \rightarrow \{\text{milk}\}$

# Definitions — Lift

- **Lift of an association rule  $X \rightarrow Y$**

- Probability of  $Y$  given  $X$  while controlling for support of  $Y$  (i.e., popularity of  $Y$ )

$$L(X \rightarrow Y) = \frac{S(X \rightarrow Y)}{S(X)S(Y)} = \frac{S(X \cup Y)}{S(X)S(Y)}$$

*Cereal*

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, cereal, milk, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, cereal, yogurt, milk  |
| 5   | bread, yogurt, milk, cheese  |

$$L(\{\text{cereal}\} \rightarrow \{\text{bread}\}) = \frac{S(\{\text{cereal}, \text{bread}\})}{S(\{\text{cereal}\})S(\{\text{bread}\})} = \frac{0.4}{0.6 \cdot 0.8} = 0.833$$

# Lift — Interpretation

$$L(\{cereal\} \rightarrow \{bread\}) = \frac{S(\{cereal, bread\})}{S(\{cereal\})S(\{bread\})} = \frac{0.4}{0.6 \cdot 0.8} = \underline{\underline{0.833}}$$

- Probability of {bread}  
 $S(\{bread\}) = 0.8$
- Probability of {bread} given {cereal}  
 $C(\{cereal\} \rightarrow \{bread\}) = \underline{\underline{0.66}}$

Presence of cereal **reduces** probability of bread!

$$\Rightarrow L(\{cereal\} \rightarrow \{bread\}) \leq 1.0$$

- **Usage of lift** (and other metrics for association rules)
  - Further filtering and ranking of association rules
  - Finding "substitution" items

**Note:** Lift is not part of Apriori algorithm since anti-monotonicity principle does not hold here

# Quick "Quiz"

Which association rule would you expect to have the **smallest lift**?

A

$\{cereal\} \rightarrow \{milk\}$

B 

$\{coke\} \rightarrow \{pepsi\}$

C

$\{milo\} \rightarrow \{nutella\}$

D

$\{banana\} \rightarrow \{grapes\}$

# Outline

- Association Rule Mining
  - Overview
  - Applications
- Definitions
- Algorithms
  - Brute-Force
  - A-Priori
- Discussion & Summary

# Discussion

- Alternative metric to decide whether a rule is interesting (beyond confidence and lift)
  - Conviction, all-confidence, collective strength, leverage
- Additional useful information to consider, for example:
  - Attributes of items (e.g., quantity and price of products)
  - Sequence of items (e.g., order when products have been added to the cart)
  - Categories of items (e.g., "milk" and "yogurt" are both "dairy" products)
  - User information (e.g., associating multiple transactions to the same user)
- Reminder: Rules indicate correlations / co-occurrences, NOT causality!

# Summary

- **Pattern of interest: Association Rule  $X \rightarrow Y$** 
  - Predicting the occurrence of some items  $Y$  based on occurrence of other items  $X$
  - Applicable to a wider range of task for transactional data
  - Various metrics that define whether a rule is useful (e.g., support, confidence, lift)
- **Practical algorithm to handle complexity**
  - Decoupling calculations of support and confidence
  - Apriori algorithm for Frequent Itemset Generation and Association Rule Generation

# Solutions to Quick Quizzes

- Slide 22: C
- Slide 37: C
- Slide 57: B

# **CS5228: Knowledge Discovery and Data Mining**

## Lecture 5 — Classification & Regression I

# Course Logistics — Update

- Assignment 2
  - Release: Fri, Sep 13
  - Submission deadline: Thu, Oct 03 (11.59 pm)
  - Reminder: Please adhere to naming and uploading guidelines
- Midterm exam preparation
  - Install and check out Examplify (see [student guide](#))
  - Try practice exam (Non-Secure Block Internet)

☒ Survey for lower laptops

# Recap — Association Rules

- **Association Rule Mining**
  - Input: database of transactions (i.e., sets of items)
  - Output: rules predicting the occurrence of some items based on occurrence of other items
- **Example: Market Basket Analysis**
  - Item = product in supermarket
  - Transaction = products in basket at check-out
  - Interesting rules = customers who buy X also tend to Y

| TID | Items                        |
|-----|------------------------------|
| 1   | bread, yogurt                |
| 2   | bread, milk, cereal, eggs    |
| 3   | yogurt, milk, cereal, cheese |
| 4   | bread, yogurt, milk, cereal  |
| 5   | bread, yogurt, milk, cheese  |

# Recap — Association Rules

- **Interesting association rules**
  - Different definitions that quantify usefulness of a rule:support, confidence, lift, conviction, leverage, etc.
- **Important observation**
  - Relationship between support and confidence

→ Decoupling Support and Confidence — 2 steps:  
1) Frequent Itemset Generation  
2) Association Rule Generation

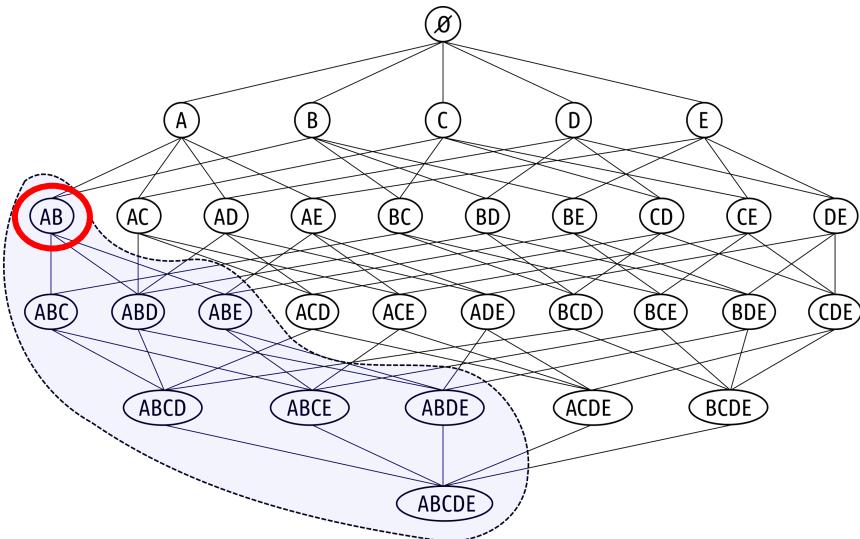
$$C(X \rightarrow Y) = \frac{S(X \rightarrow Y)}{S(X)} = \frac{S(X \cup Y)}{S(X)}$$

- **Anti-monotone** property of support and confidence

→ Apriori Algorithms for Frequent Itemset and Association Rule Generation

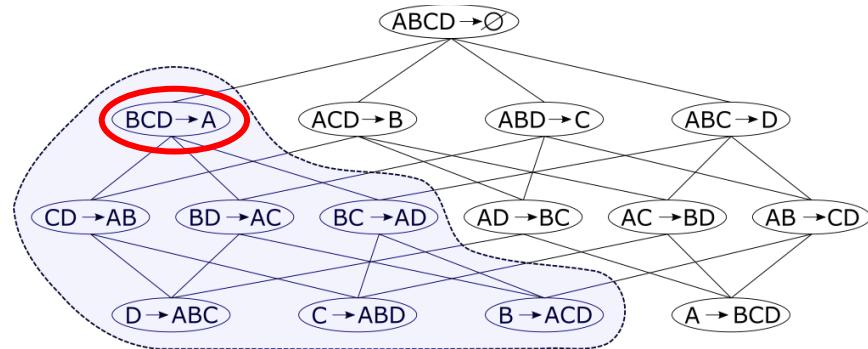
# Recap — Association Rules

Itemset Lattice



If an itemset (e.g., {A,B}) is not frequent, then any superset (e.g., {A,B,D}, {A,B,C,D}) is also not frequent

Rule Lattice



If rule  $BCD \rightarrow A$  has a insufficient confidence, so do all rules containing A in the consequent

# Outline

- **Classification & Regression**
  - Overview & Examples
  - Basic Setup
  - Evaluation
- **Nearest Neighbor Methods**
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# Classification & Regression: Overview

- **Classification AND Regression**
  - Core tasks of supervised machine learning
  - Training: Find patterns in dataset between the values of **dependent variable(s)** given the values **independent variable(s) / features**
  - Prediction: Use patterns to assign values to dependent variables new/unseen data
- **Classification VS. Regression**
  - Classification: Dependent variable is categorical
  - Regression: Dependent variable is continuous

Independent variables (features)

| Age | Edu-<br>cation | Marital<br>Status | Income<br>Level | Credit<br>Approval | Credit<br>Limit |
|-----|----------------|-------------------|-----------------|--------------------|-----------------|
| 23  | Masters        | Single            | Mid             | No                 | —               |
| 35  | College        | Married           | High            | Yes                | \$S7,000        |
| 26  | Masters        | Single            | High            | No                 | —               |
| 41  | PhD            | Single            | Mid             | Yes                | \$S5,000        |
| 18  | Poly           | Single            | Low             | No                 | —               |
| 55  | Poly           | Married           | High            | Yes                | \$S10,000       |
| 30  | College        | Single            | High            | Yes                | \$S8,000        |
| 35  | PhD            | Married           | High            | Yes                | \$S10,000       |
| 28  | Masters        | Married           | Mid             | Yes                | \$S5,000        |
| 45  | Masters        | Married           | Mid             | ???                | ???             |

Dependent, categorical variable

Dependent, numerical variable

# Application Examples

| month   | flat_type | block | street_name           | floor_area_sqm | flat_model     | lease_commence_date | latitude | longitude  | subzone       | planning_area | resale_price |
|---------|-----------|-------|-----------------------|----------------|----------------|---------------------|----------|------------|---------------|---------------|--------------|
| 2013-02 | 4 room    | 4B    | boon tiona road       | 91.0           | model a        | 2005                | 1.286589 | 103.831950 | tiong bahru   | bukit merah   | 663888.0     |
| 1997-08 | 3 room    | 11    | holland drive         | 65.0           | improved       | 1975                | 1.309054 | 103.794063 | holland drive | queenstown    | 228000.0     |
| 2007-08 | executive | 558   | jurong west street 42 | 157.0          | maisonette     | 1985                | 1.354198 | 103.717344 | hong kah      | jurong west   | 305000.0     |
| 1996-02 | 4 room    | 354   | hougang avenue 7      | 105.0          | new generation | 1986                | 1.372408 | 103.899433 | kangkar       | hougang       | 235000.0     |
| 2015-06 | 3 room    | 99    | old airport road      | 56.0           | standard       | 1969                | 1.308590 | 103.888245 | aljunied      | geylang       | 320000.0     |

- Prediction of flat prices (regression task)
  - Help sellers and buyers to make better decisions
  - Understand the importance of attributes on flat prices

# Application Examples

Has heart disease (1)  
or not (0)

| ID | age  | sex | chest | rest_bp | chol   | rest_ecg | max_hr | oldpeak | slope | num_vessels | thal | class |
|----|------|-----|-------|---------|--------|----------|--------|---------|-------|-------------|------|-------|
| 0  | 49.2 | 0   | 4.00  | 163.00  | 181.11 | 0        | 148.23 | 0.94    | 2     | 0           | 3    | 1     |
| 1  | 53.6 | 1   | 1.74  | 130.23  | 276.47 | 2        | 152.92 | 0.12    | 2     | 0           | 3    | 0     |
| 2  | 49.6 | 1   | 4.00  | 147.00  | 223.30 | 2        | 102.35 | 1.62    | 2     | 2           | 7    | 1     |
| 3  | 59.0 | 1   | 4.00  | 112.37  | 187.25 | 0        | 158.16 | 0.00    | 1     | 1           | 7    | 1     |
| 4  | 51.1 | 1   | 1.95  | 138.03  | 238.48 | 0        | 172.54 | 1.15    | 1     | 1           | 3    | 0     |

- Health analytics: prediction of heart disease
  - Diagnosis and therapy support systems
  - Identify most important risk factors

# Application Examples

- Sentiment Analysis / Opinion Mining (over text)

Regression



94%



90%

TOMATOMETER

AUDIENCE SCORE



What you will be getting when you walk into an inevitably overstuffed movie theater is something singular that reflects our age in a way that none of the MCU films that preceded it have—indeed, very few Hollywood spectacles ever have.

May 1, 2019 | Rating: 3.5/4 | [Full Review...](#)

Oliver Jones  
Observer  
★ Top Critic

What's missing from "Endgame" is the free play of imagination, the liberation of speculation, the meandering paths and loose ends that start in logic and lead to wonder.

April 28, 2019 | [Full Review...](#)

Richard Brody  
New Yorker  
★ Top Critic

Endgame consists almost entirely of the downtime scenes that were always secretly everyone's favorite parts of these movies anyway.

April 26, 2019 | [Full Review...](#)

Dana Stevens  
Slate  
★ Top Critic

Classification



**Fresh**



**Rotten**



**Fresh**



**Rotten**

Tomatometer

Audience

# Application Examples

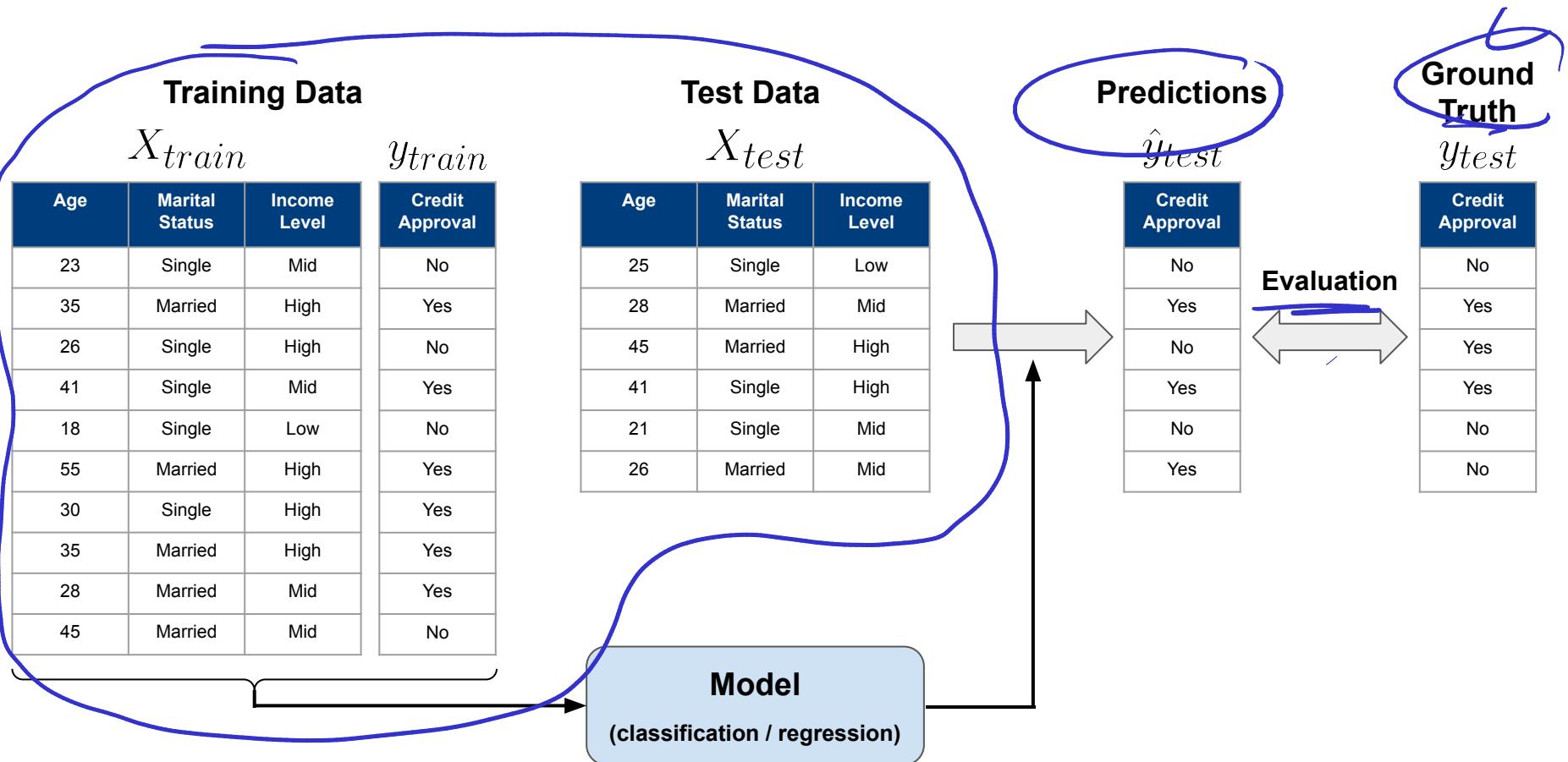
- Self-driving vehicles
  - Input: image & sensor data
- Regression tasks, e.g.:
  - Acceleration
  - Steering angle
  - Event prediction (%)
- Classification tasks, e.g.:
  - Obstacle detection
  - Street sign identification



# Outline

- **Classification & Regression**
  - Overview & Examples
  - **Basic Setup**
  - Evaluation
- **Nearest Neighbor Methods**
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# Supervised Training/Learning — Basic Setup



# Supervised Training/Learning — Training & Test Data

- Given: Dataset D of pairs  $\{(x, y)\}$ 
  - x — features (independent variables)
  - y — label (dependent variable)
- Split dataset D into:
  - $D_{train}$  — data for training the model
  - $D_{test}$  — data for evaluating the model
  - Important:  $D_{train} \cap D_{test} = \emptyset$   
(the test data is not allowed to be used during training)

| Age | Edu-<br>cation | Marital<br>Status | Income<br>Level | Credit<br>Approval |
|-----|----------------|-------------------|-----------------|--------------------|
| 23  | Masters        | Single            | Mid             | No                 |
| 35  | College        | Married           | High            | Yes                |
| 26  | Masters        | Single            | High            | No                 |
| 41  | PhD            | Single            | Mid             | Yes                |
| 18  | Poly           | Single            | Low             | No                 |
| 55  | Poly           | Married           | High            | Yes                |
| 30  | College        | Single            | High            | Yes                |
| 35  | PhD            | Married           | High            | Yes                |
| 28  | Masters        | Married           | Mid             | Yes                |
| 45  | Masters        | Married           | Mid             | No                 |

$D_{train}$

$D_{test}$

features      labels

# Supervised Training/Learning

- Model — Parameterized function  $h(x, \Theta) = y$ 
  - Maps input space (features) to the output space (labels)
  - $\Theta$  — trainable/learnable parameters of the model  
(note: not all models are parameterized)
- Model selection — Selection of a "family" of functions  $h(x, \Theta) = y$ 
  - K-Nearest Neighbor, Decision Trees, Linear Models, Neural Networks, ...
- Training / Learning
  - Process of systematically finding the best values for  $\Theta$
  - $\Theta_{best} \Leftrightarrow$  best mapping between features and labels

# Outline

- **Classification & Regression**
  - Overview & Examples
  - Basic Setup
  - **Evaluation**
- **Nearest Neighbor Methods**
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# Classification & Regression — Evaluation

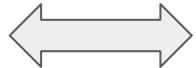
## Regression

$\hat{y}_{test}$

| Credit Limit |
|--------------|
| \$S9,000     |
| \$S4,000     |
| \$S5,800     |
| \$S10,000    |
| \$S11,000    |
| \$S3,500     |

$y_{test}$

| Credit Limit |
|--------------|
| \$S10,000    |
| \$S4,000     |
| \$S6,000     |
| \$S12,000    |
| \$S10,000    |
| \$S3,000     |



Direct comparison of numerical values

Common: Root Mean Squared Error (RMSE):

$$\underline{RMSE} = \sqrt{\frac{\sum_{i=1}^N (\hat{y}_i - y_i)^2}{N}}$$

*prediction / estimate*

## Classification

$\hat{y}_{test}$

| Credit Approval |
|-----------------|
| No              |
| Yes             |
| No              |
| Yes             |
| No              |
| Yes             |

$y_{test}$

| Credit Approval |
|-----------------|
| No              |
| Yes             |
| Yes             |
| Yes             |
| No              |
| No              |

*ground truth*



Question: How to compare classification results?

# Classification: Evaluation

- Base case: Binary classification (2 labels: 0 or 1)
- Example: COVID-19 test

| test result      | infected   |
|------------------|------------|
| $\hat{y}_{test}$ | $y_{test}$ |
| 0                | 0          |
| 0                | 0          |
| 0                | 1          |
| 1                | 1          |
| 0                | 0          |
| 1                | 0          |



Confusion Matrix

|                           |   | ground truth label $y$ |   |
|---------------------------|---|------------------------|---|
|                           |   | 1                      | 0 |
| predicted label $\hat{y}$ | 1 | 1                      | 1 |
|                           | 0 | 1                      | 3 |

# Classification: Evaluation — Confusion Matrix

|                           |   | ground truth label $y$ |                      |
|---------------------------|---|------------------------|----------------------|
|                           |   | 1                      | 0                    |
| predicted label $\hat{y}$ | 1 | True Positives (TP)    | False Positives (FP) |
|                           | 0 | False Negatives (FN)   | True Negatives (TN)  |

- True Positives (TP):** Number of positive classes that have been correctly predicted as positive
- True Negatives (TN):** Number of negative classes that have been correctly predicted as negative
- False Positives (FP):** Number of negative classes that have been incorrectly predicted as positive
- False Negatives (FN):** Number of positive classes that have been incorrectly predicted as negative

# Classification: Evaluation — Popular Metrics

- Accuracy

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

|           |   | $y$ |    |
|-----------|---|-----|----|
|           |   | 1   | 0  |
| $\hat{y}$ | 1 | TP  | FP |
|           | 0 | FN  | TN |

- Sensitivity, Specificity

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

|           |   | $y$ |    |
|-----------|---|-----|----|
|           |   | 1   | 0  |
| $\hat{y}$ | 1 | TP  | FP |
|           | 0 | FN  | TN |

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

|           |   | $y$ |    |
|-----------|---|-----|----|
|           |   | 1   | 0  |
| $\hat{y}$ | 1 | TP  | FP |
|           | 0 | FN  | TN |

# Classification: Evaluation — Popular Metrics

- Precision, Recall, F1 Score

$$F_1 = \frac{\frac{P}{R} + \frac{R}{P}}{2} = 0.5$$

Harmonic Mean of  
Precision and Recall

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{\frac{1}{Precision} + \frac{1}{Recall}}{\frac{1}{Precision} + \frac{1}{Recall}} = 0$$

|   |    | 1  | 0 |
|---|----|----|---|
| 1 | TP | FP |   |
| 0 | FN | TN |   |

|   |    | 1  | 0 |
|---|----|----|---|
| 1 | TP | FP |   |
| 0 | FN | TN |   |

healthy  
patient

|   |    | 1  | 0 |
|---|----|----|---|
| 1 | TP | FP |   |
| 0 | FN | TN |   |

# Classification: Evaluation — Why so Many Measures?

- Problem: (Highly) imbalanced datasets
- Example use case: COVID-19 test
  - Most people in a population are not infected
  - Assume a test that always(!) returns "negative"

|                           |   | ground truth label $y$ |        |
|---------------------------|---|------------------------|--------|
|                           |   | 1                      | 0      |
| predicted label $\hat{y}$ | 1 | 0                      | 0      |
|                           | 0 | 200                    | 10,000 |

*(Note: The table shows a 10,000x10,000 population where 99.8% are healthy (0) and 0.2% are infected (1). A test that always returns negative (0) will correctly identify all healthy individuals but incorrectly identify 200 healthy individuals as infected.)*

$$\text{Precision} = \frac{C_1 C_1}{C_1 C_1 + C_0 C_0} = ?$$

$$\text{Recall} = \frac{C_1 C_1}{C_1 C_1 + C_0 C_0} = 0$$

$$\text{Accuracy} = \frac{0 + 10000}{0 + 0 + 10000 + 200} = 98\%$$

$$\text{Specificity} = \frac{10000}{10000 + 0} = 100\%$$

→ Very high values despite "useless" test

# Classification: Evaluation — Why so Many Measures?

- Observation: FP and FN not always equally problematic
- Example: Heart disease prediction
  - BAD: misclassifying a high-risk person
  - OK-ish: misclassifying a healthy person
- Example: News article classification  
(e.g., for search engines such as Google News)
  - BAD: showing article of wrong category
  - OK: missing a relevant article in result

$$Precision = \frac{TP}{TP + FP} \qquad Recall = \frac{TP}{TP + FN}$$



**Recall > Precision**



**Recall < Precision**

# Classification: Evaluation — Numerical Class Scores

- Assumption so far:  
 $\hat{y}$  are **class labels**
- Output of many models:  
 $\hat{y}$  are **class scores**
  - e.g., probability of a class

## → Thresholding

- Use threshold to convert  $\hat{y}$  to a binary variable 0/1

*model  
output*

|      |
|------|
| 0.45 |
| 0.30 |
| 0.55 |
| 0.25 |
| 0.35 |
| 0.55 |
| 0.30 |
| 0.60 |
| 0.40 |
| 0.45 |

class scores

$\hat{y}_{0.5}$

|          |
|----------|
| 0.45 → 0 |
| 0.30 → 0 |
| 0.55 → 1 |
| 0.25 → 0 |
| 0.35 → 0 |
| 0.55 → 1 |
| 0.30 → 0 |
| 0.60 → 1 |
| 0.40 → 0 |
| 0.45 → 0 |

threshold=0.5

$\hat{y}_{0.43}$

|          |
|----------|
| 0.45 → 1 |
| 0.30 → 0 |
| 0.55 → 1 |
| 0.25 → 0 |
| 0.35 → 0 |
| 0.55 → 1 |
| 0.30 → 0 |
| 0.60 → 1 |
| 0.40 → 0 |
| 0.45 → 1 |

threshold=0.43

$y$

|   |
|---|
| 1 |
| 0 |
| 0 |
| 0 |
| 0 |
| 1 |
| 1 |
| 0 |
| 1 |
| 0 |
| 1 |
| 0 |
| 1 |
| 0 |
| 1 |

ground truth

# Classification: Evaluation — Numerical Class Scores

- Different threshold yield different results

|                 |   | $y$ |               |
|-----------------|---|-----|---------------|
|                 |   | 1   | 0             |
| $\hat{y}_{0.5}$ | 1 | 2   | 1             |
|                 | 0 | 3   | 4             |
|                 |   |     | threshold=0.5 |

$$F1 = 0.66$$

$\hat{y}_{0.43}$

|                  |   | $y$ |                |
|------------------|---|-----|----------------|
|                  |   | 1   | 0              |
| $\hat{y}_{0.43}$ | 1 | 4   | 1              |
|                  | 0 | 1   | 4              |
|                  |   |     | threshold=0.43 |

$$F1 = \underline{0.80}$$

→ Question: What threshold to use? — Answer: Try them all!

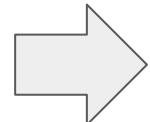
# Receiver Operating Characteristic (ROC)

- Plot True Positive Rate (Sensitivity) against False Positive Rate (1-Specificity)
  - Plot values for each meaningful threshold  $t$  (meaningful = has effect on result)

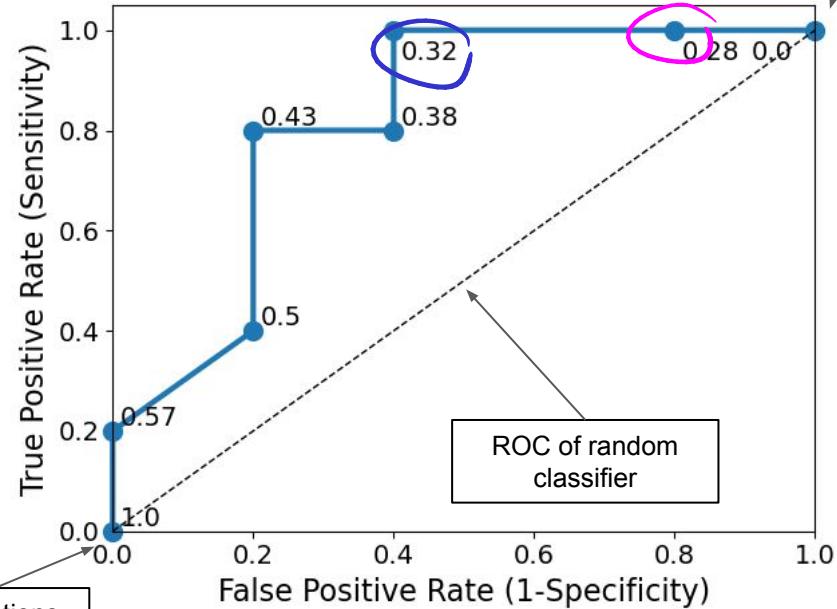
9

| $t = 0.0 >$  | 0.25 | 0 |
|--------------|------|---|
| $t = 0.28 >$ | 0.30 | 0 |
| $t = 0.32 >$ | 0.30 | 0 |
| $t = 0.38 >$ | 0.35 | 1 |
| $t = 0.43 >$ | 0.40 | 0 |
| $t = 0.45 >$ | 0.45 | 1 |
| $t = 0.45 >$ | 0.45 | 1 |
| $t = 0.50 >$ | 0.55 | 0 |
| $t = 0.55 >$ | 0.55 | 1 |
| $t = 0.57 >$ | 0.60 | 1 |
| $t = 1.0 >$  |      |   |

class scores      ground truth  
both sorted w.r.t. scores

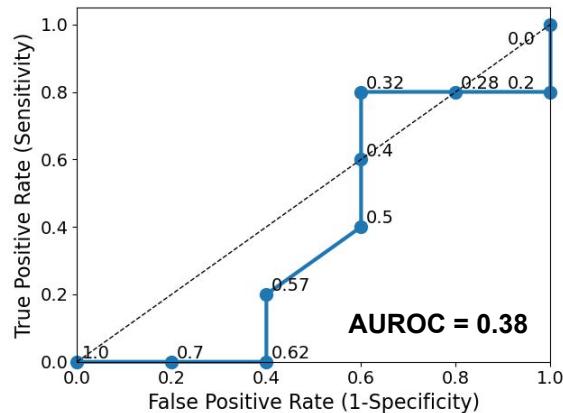


predictions  
are all 0

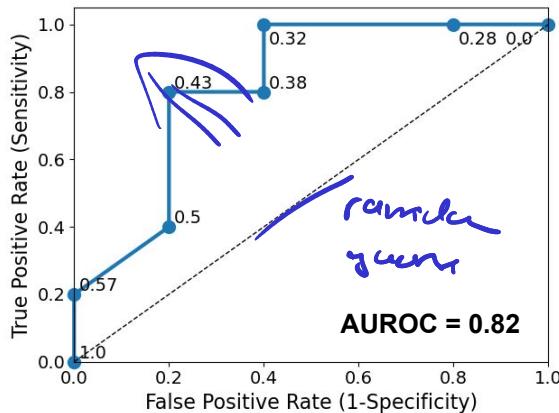


# Receiver Operating Characteristic (ROC) — Comparison

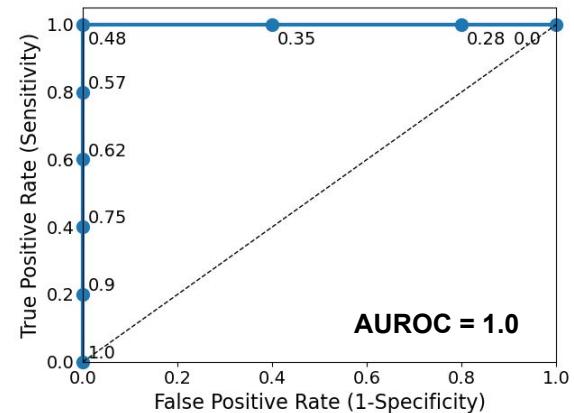
Class scores from 3 different classifiers



Poor classifier



"Decent" classifier

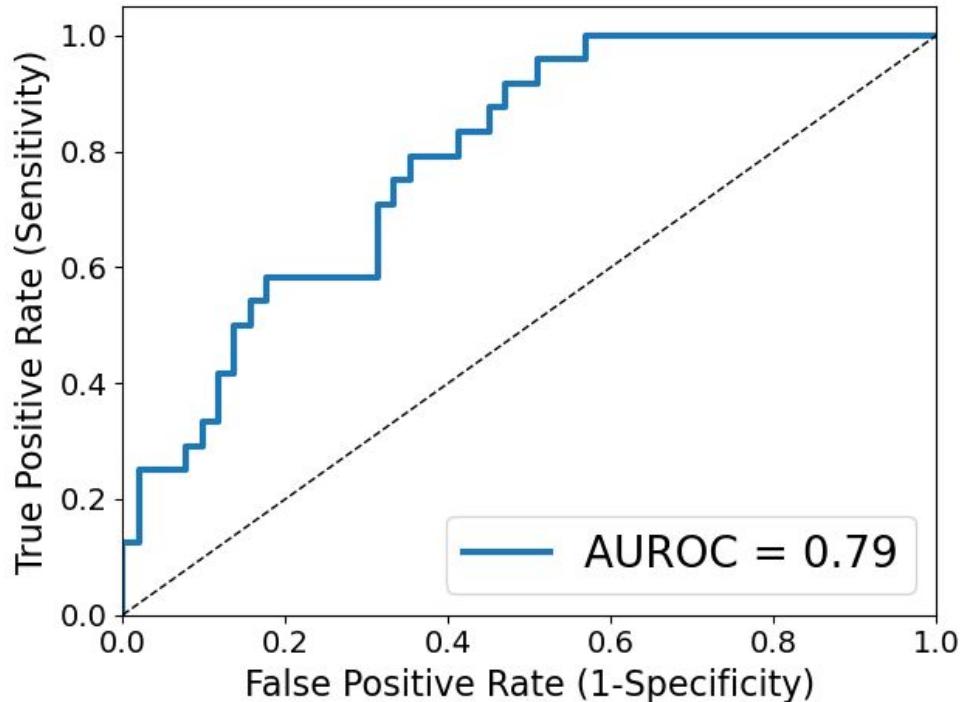


Perfect classifier

- Quantify quality of classifiers using **AUROC** (or AUC)
  - Area Under Receiver Operating Characteristic
  - AUROC of random classifier: 0.5

# ROC / AUROC — Practical Example

- IRIS dataset
  - 3 classes of Iris plants
  - 50 samples per class
  - 4 continuous features (sepal/petal length/width)
- ROC / AUROC
  - More samples, more thresholds
  - Smoother ROC
  - Thresholds typically omitted



# Quick Quiz

In what situation is **accuracy** a good measure?

**A**

The test data contains a sufficient amount of data samples

**B**

There are more than 2 class labels

**C**

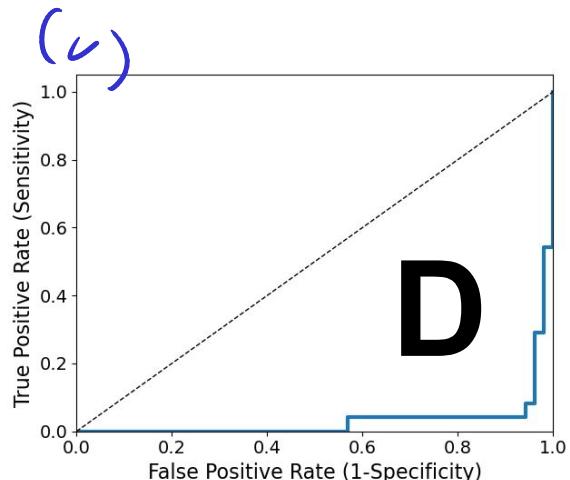
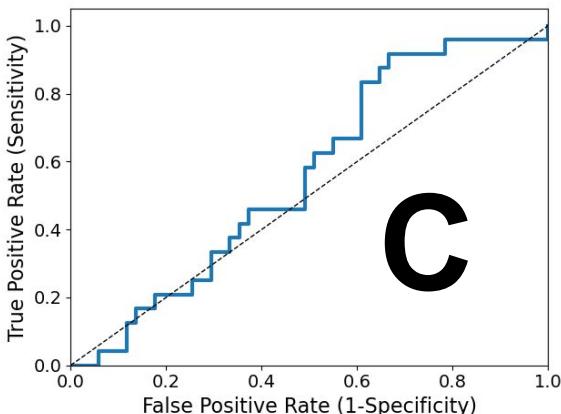
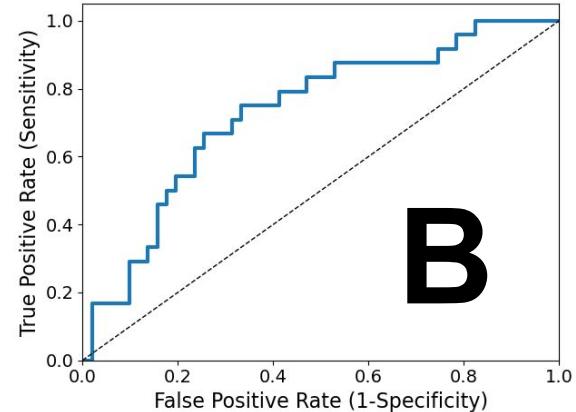
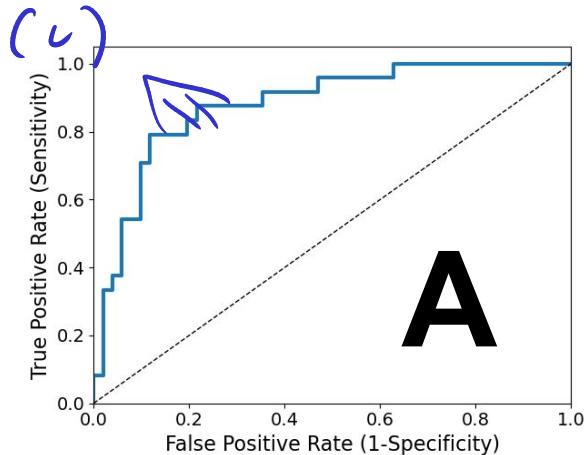
Accuracy is never a good measure

**D** 

The dataset is reasonably balanced (similar number of class labels)

# Quick Quiz

For a binary classification task, which of the 4 classifiers would you choose?



# Classification: Evaluation — Beyond 2 Classes

- Example: 3 classes, 50 samples

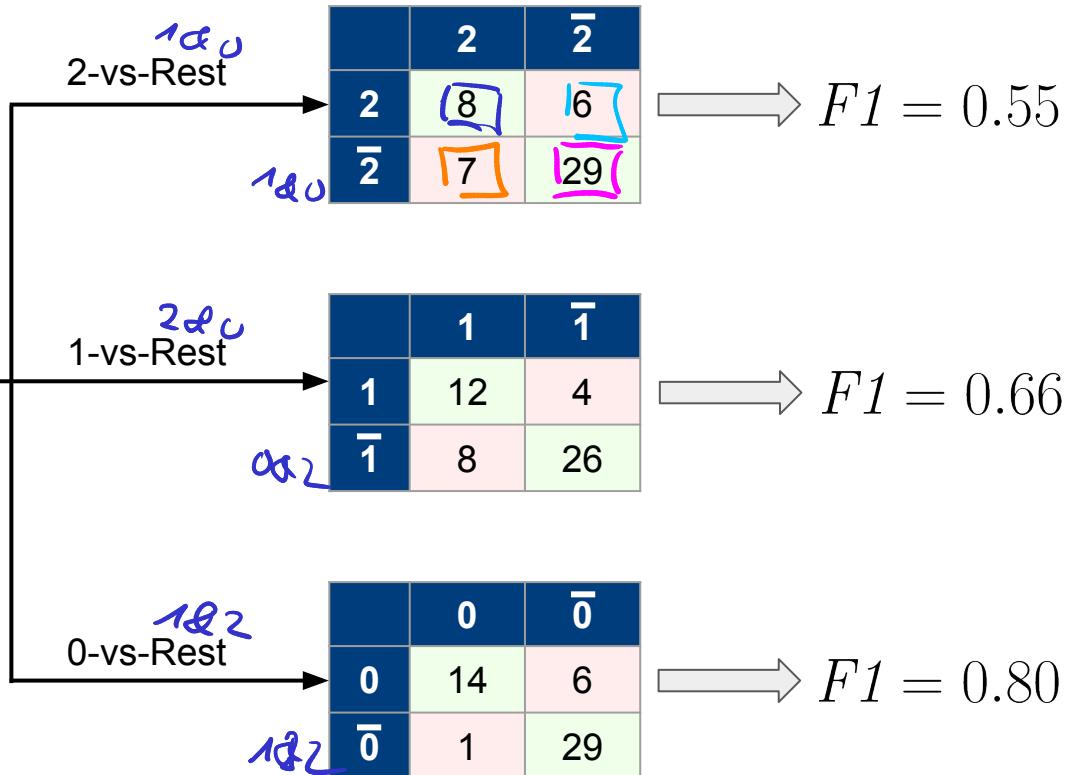
|                           |   | ground truth label $y$ |    |    |
|---------------------------|---|------------------------|----|----|
|                           |   | 2                      | 1  | 0  |
| predicted label $\hat{y}$ | 2 | 8                      | 6  | 0  |
|                           | 1 | 3                      | 12 | 1  |
|                           | 0 | 4                      | 2  | 14 |

$$\text{Accuracy} = \frac{8 + 12 + 14}{8 + 12 + 14 + 6 + 3 + 1 + 4 + 2} = 0.68$$

# Multiclass Evaluation — One-vs-Rest Confusion Matrices

- Example:

|                           |   | ground truth label $y$ |    |   |
|---------------------------|---|------------------------|----|---|
|                           |   | 2                      | 1  | 0 |
| predicted label $\hat{y}$ | 2 | 8                      | 6  | 0 |
|                           | 1 | 3                      | 12 | 1 |
| 0                         | 4 | 2                      | 14 |   |



# One-vs-Rest — Micro Averaging

|           |   |           |
|-----------|---|-----------|
|           | 2 | $\bar{2}$ |
| 2         | 8 | 6         |
| $\bar{2}$ | 7 | 29        |

|           |    |           |
|-----------|----|-----------|
|           | 1  | $\bar{1}$ |
| 1         | 12 | 4         |
| $\bar{1}$ | 8  | 26        |

|           |    |           |
|-----------|----|-----------|
|           | 0  | $\bar{0}$ |
| 0         | 14 | 6         |
| $\bar{0}$ | 1  | 29        |

Average over all  
TP, FP, FN, TN

|           |       |           |
|-----------|-------|-----------|
|           | c     | $\bar{c}$ |
| c         | 11.33 | 5.33      |
| $\bar{c}$ | 5.33  | 28        |

$$\frac{8+12+14}{3}$$

$$F1 = 0.68$$

# One-vs-Rest — Macro Averaging

|           |   |           |
|-----------|---|-----------|
|           | 2 | $\bar{2}$ |
| 2         | 8 | 6         |
| $\bar{2}$ | 7 | 29        |

$$\longrightarrow F1 = 0.55$$

|           |    |           |
|-----------|----|-----------|
|           | 1  | $\bar{1}$ |
| 1         | 12 | 4         |
| $\bar{1}$ | 8  | 26        |

$$\longrightarrow F1 = 0.66$$

|           |    |           |
|-----------|----|-----------|
|           | 0  | $\bar{0}$ |
| 0         | 14 | 1         |
| $\bar{0}$ | 6  | 29        |

$$\longrightarrow F1 = 0.80$$

Average over  
all metrics

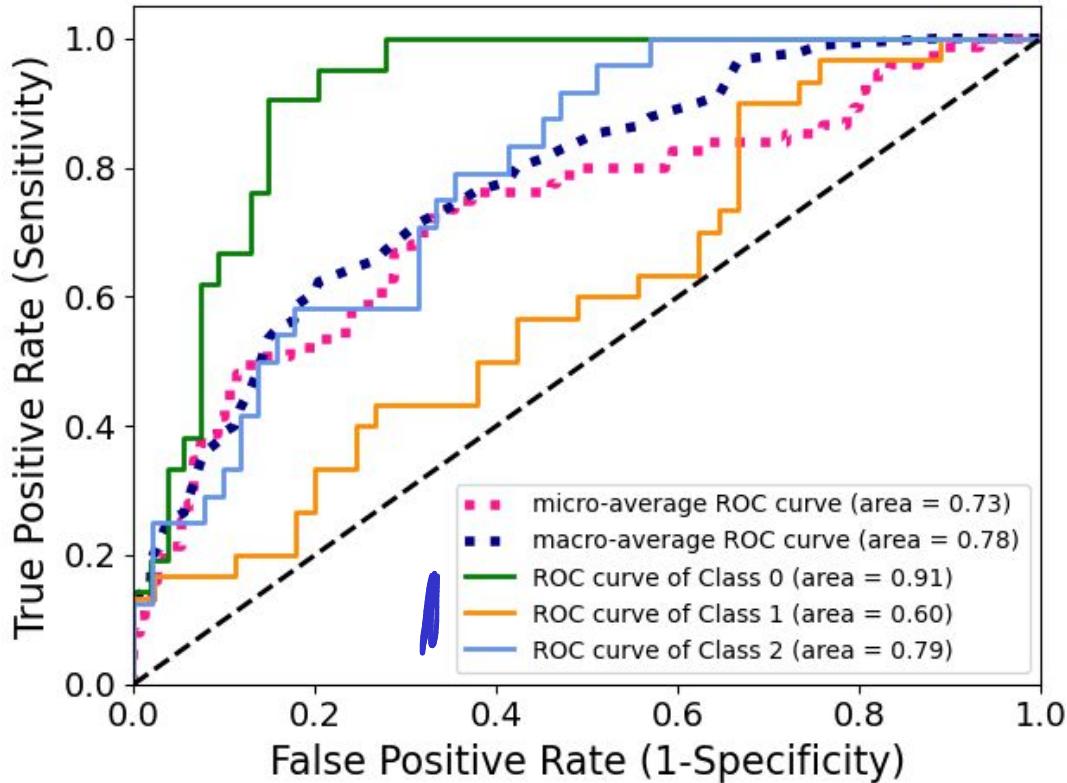
$$\longrightarrow F1 = 0.67$$

# One-vs-Rest — Macro vs. Micro Averaging

- Both methods use One-vs-Rest confusion matrices
  - All introduced metrics applicable (incl. ROC/AUROC)
- Micro-averaging
  - Averaging over TP, FP, FN, TN values of all One-vs-Rest confusion matrices
  - Favors bigger classes (since average over counts)
- Macro-averaging
  - Averaging over metrics derived from each One-vs-Rest confusion matrix
  - Treats all class equally (since metrics are normalized)

# ROC / AUROC — Practical Example (Multiclass)

- IRIS dataset
  - 3 classes of Iris plants
  - 50 samples per class
  - 4 continuous features (sepal/petal length/width)



# Quick Quiz

A **2-class** classifier and a **10-class** classifier have a f1-score of 0.6:  
Which classifier does a **better** job?

**A**

The 2-class classifier

**B** ✓

The 10-class classifier

**C**

Both are equally good

**D**

Not comparable

# Outline

- Classification & Regression
  - Overview & Examples
  - Basic Setup
  - Evaluation
- Nearest Neighbor Methods
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# K-Nearest Neighbor Algorithm (KNN)

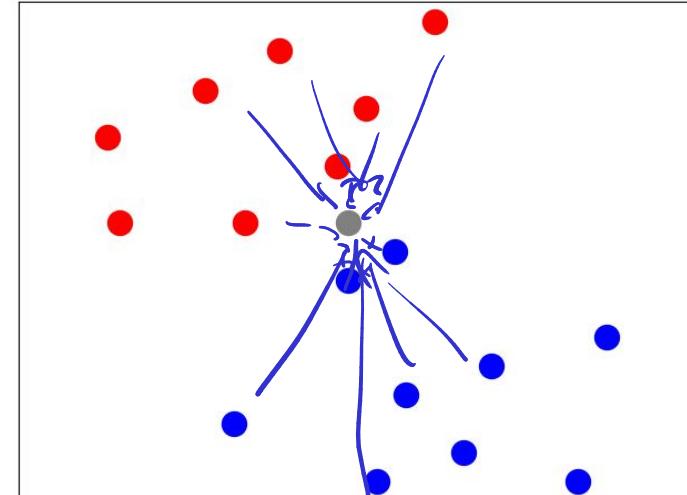
- Intuition behind KNN:

- Label of an unseen data point  $x$  derives from the labels of the  $k$ -nearest neighbors of  $x$
- Similar data points → similar labels

} Required: notion of similarity/distance

- Example

- What should be the label/color of the unseen (gray) data point?



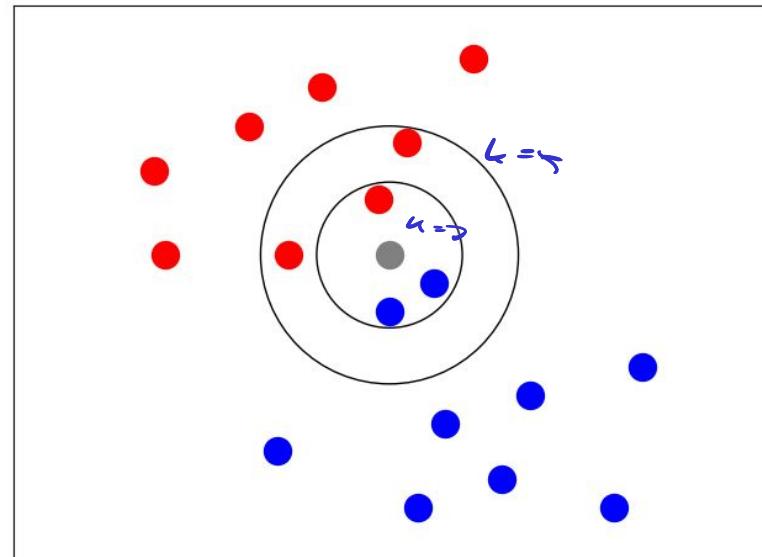
# KNN for Classification

- "Training"
  - Remember training data
- Prediction for unseen point  $x_i$ 
  - Calculate distances to all training data points  $x_j$ , e.g.:
$$d(p, q) = \sqrt{\sum_i^d (q_i - p_i)^2}$$
Euclidean distance,  $d = \# \text{features}$
  - Get the  $k$ -nearest neighbors
  - Label of  $x_i$  = most frequent label among all  $k$ -nearest neighbors



$k=3 \Rightarrow \text{Blue}$

$k=5 \Rightarrow \text{Red}$



$k$  is typically odd to minimize the chance of ties

# Quick "Quiz"

Should  $k$  not better be a **prime**  
and not just an odd number?

(e.g., to limit the chance of 3/3/3 ties for  $k=9$ )



**A**

Never set  $k$  to a prime

**B**



No need but also no harm

**C**

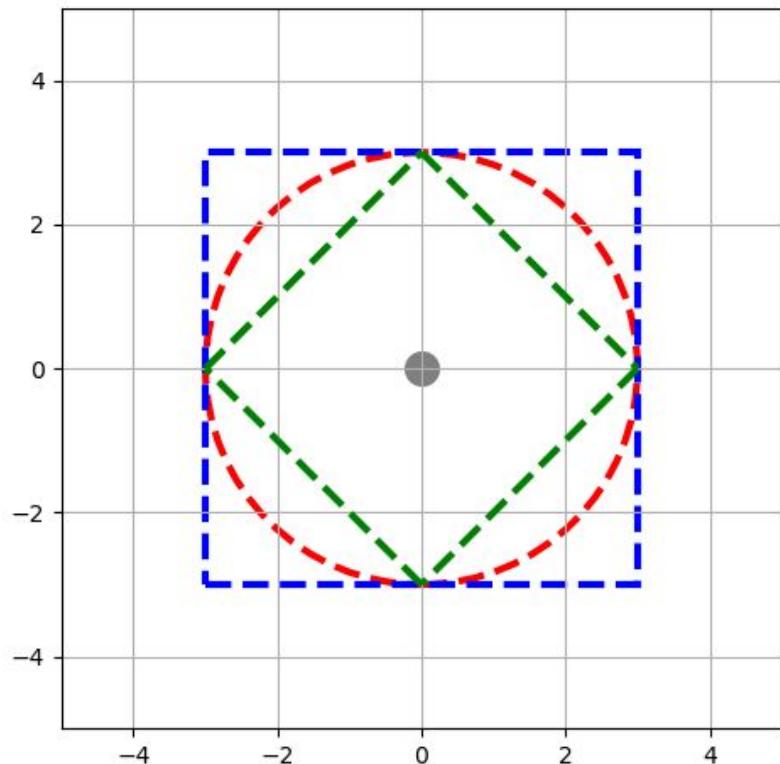
Possible but there are risks

**D**

Setting  $k$  to a prime  
is always preferable

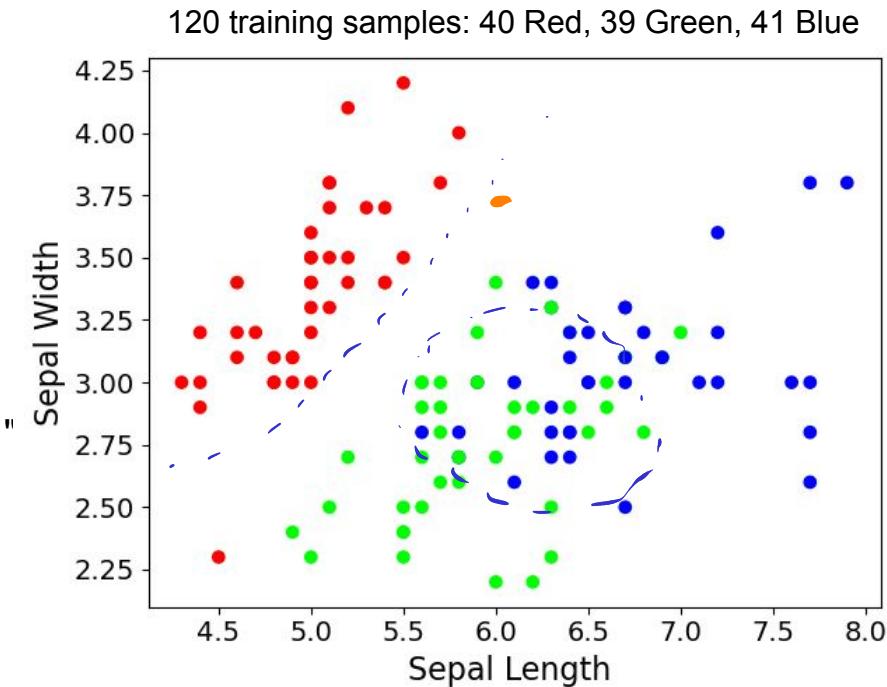
# KNN for Classification — Distance Metrics

- Example for different distance metrics
  - Euclidean distance
  - Manhattan distance
  - Chebyshev distance
- Other metrics, e.g.:
  - Cosine similarity
  - Jaccard similarity
  - ...
  - User-defined metrics



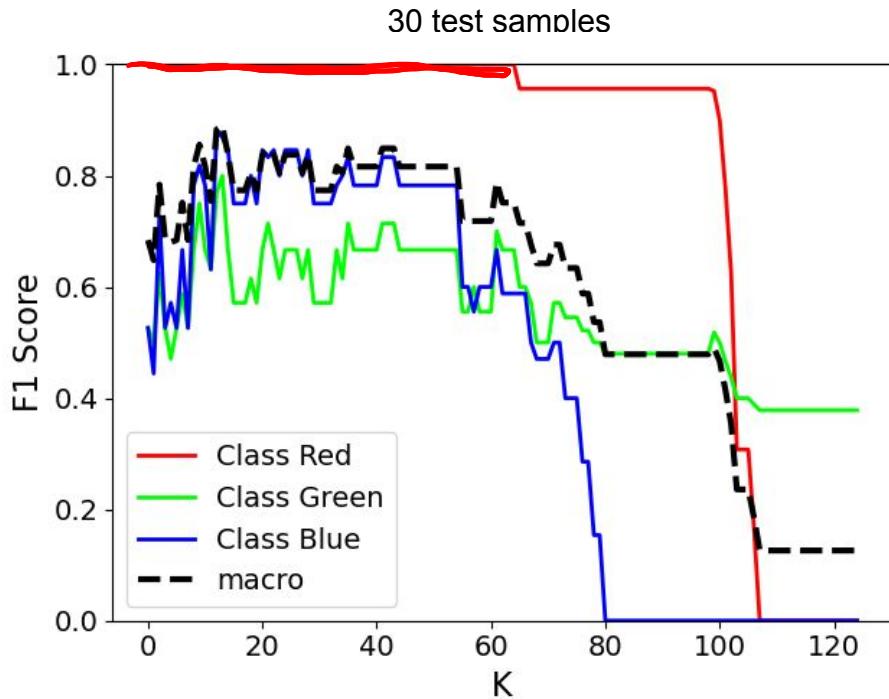
# KNN for Classification — Example

- IRIS dataset
  - 3 classes, 50 samples per class
  - 4 continuous features, but only 2 used: (sepal length & width)
- Basic EDA
  - Class "Red" well separated
  - A lot of overlap between Classes "Green" and "



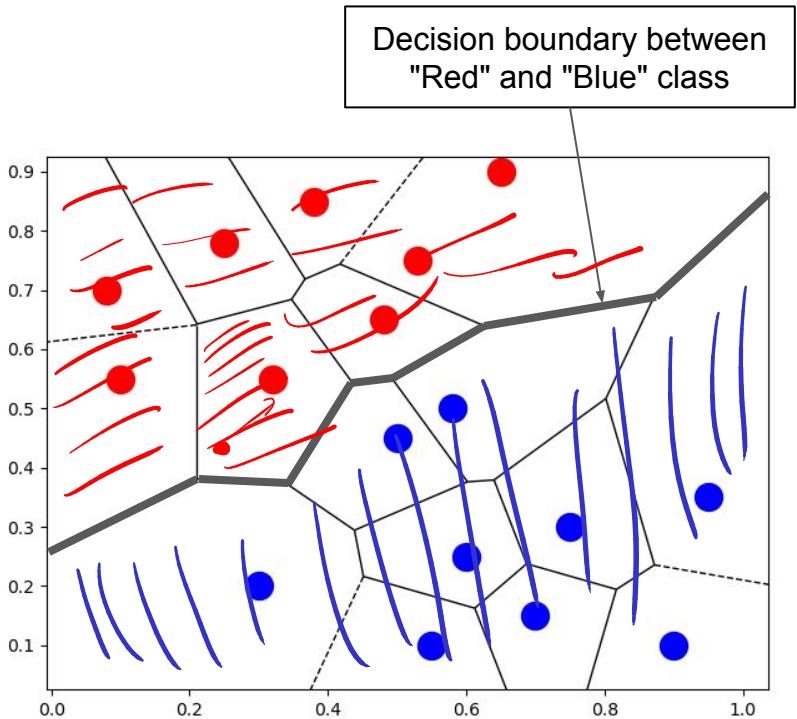
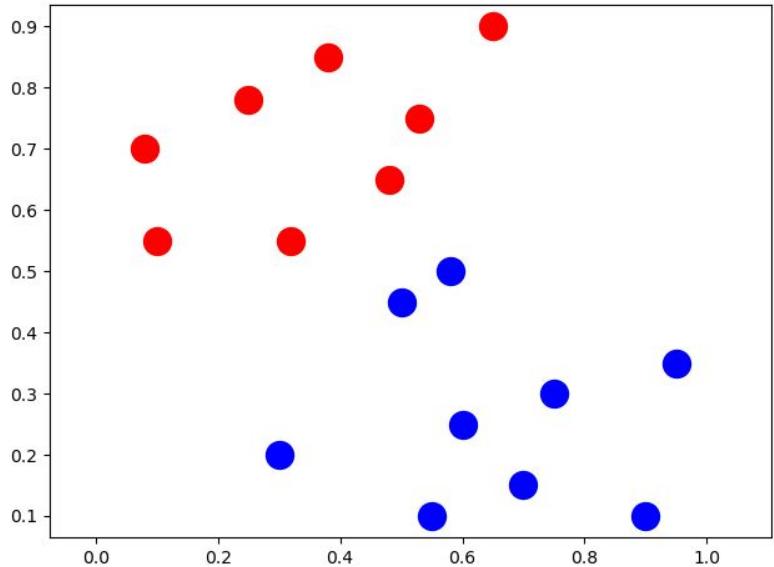
# KNN for Classification — Example

- Common outcomes
  - Different  $k$  yield different results
  - (Very) small  $k$  — results very sensitive to noise and outliers
  - Large  $k$  — insufficient capacity to properly sep
  - Very large  $k$  — most frequent class in training data starts to dominate

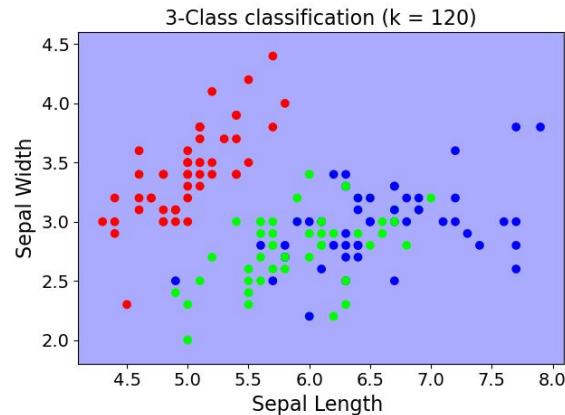
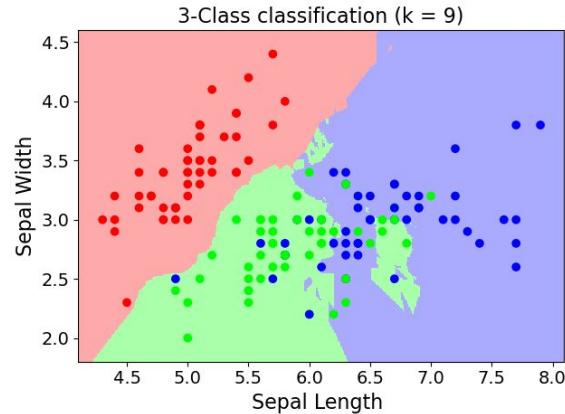
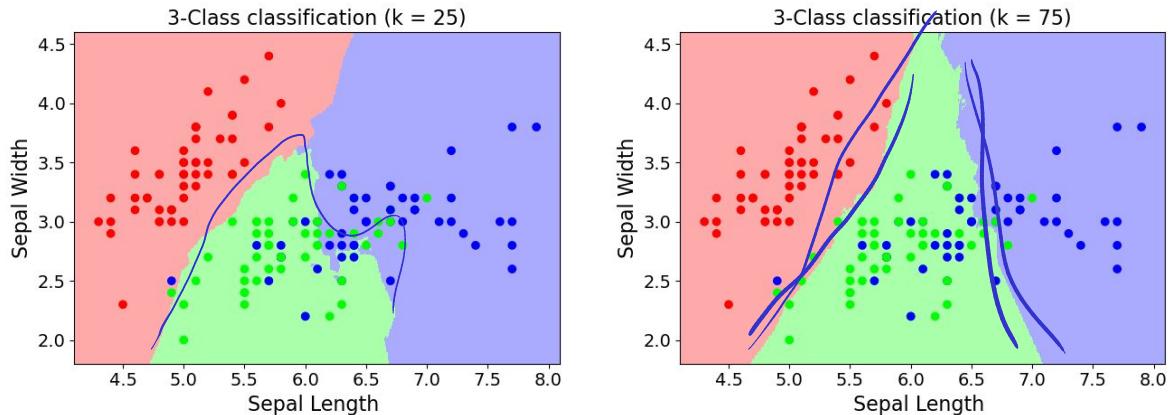
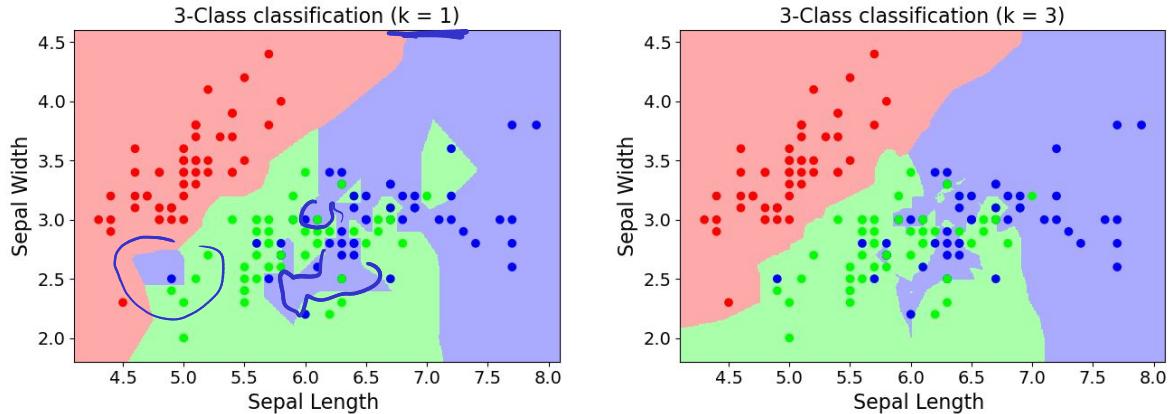
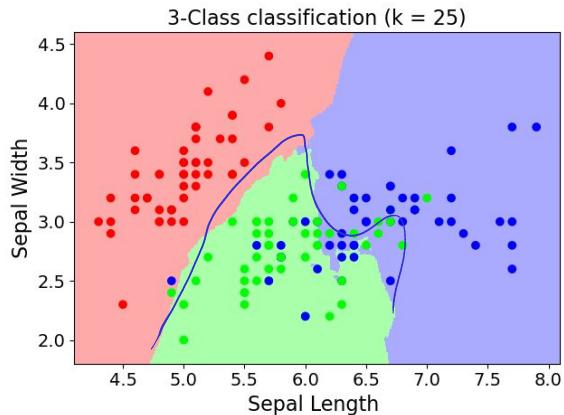
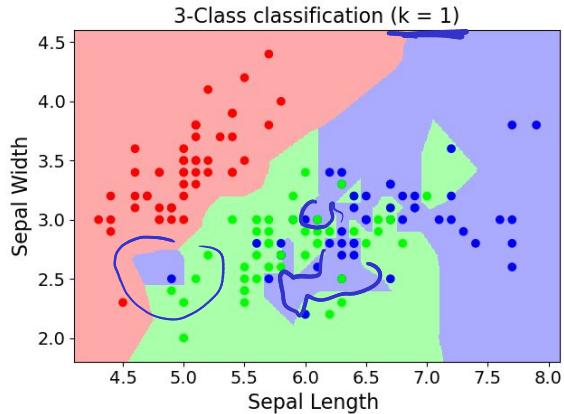


# 1-Nearest Neighbor: Voronoi Tessellation

- Decision boundaries for 1-NN =derived from Voronoi Tessellation
  - Separation of space into cells
  - Cell: set of points nearest to a single data point

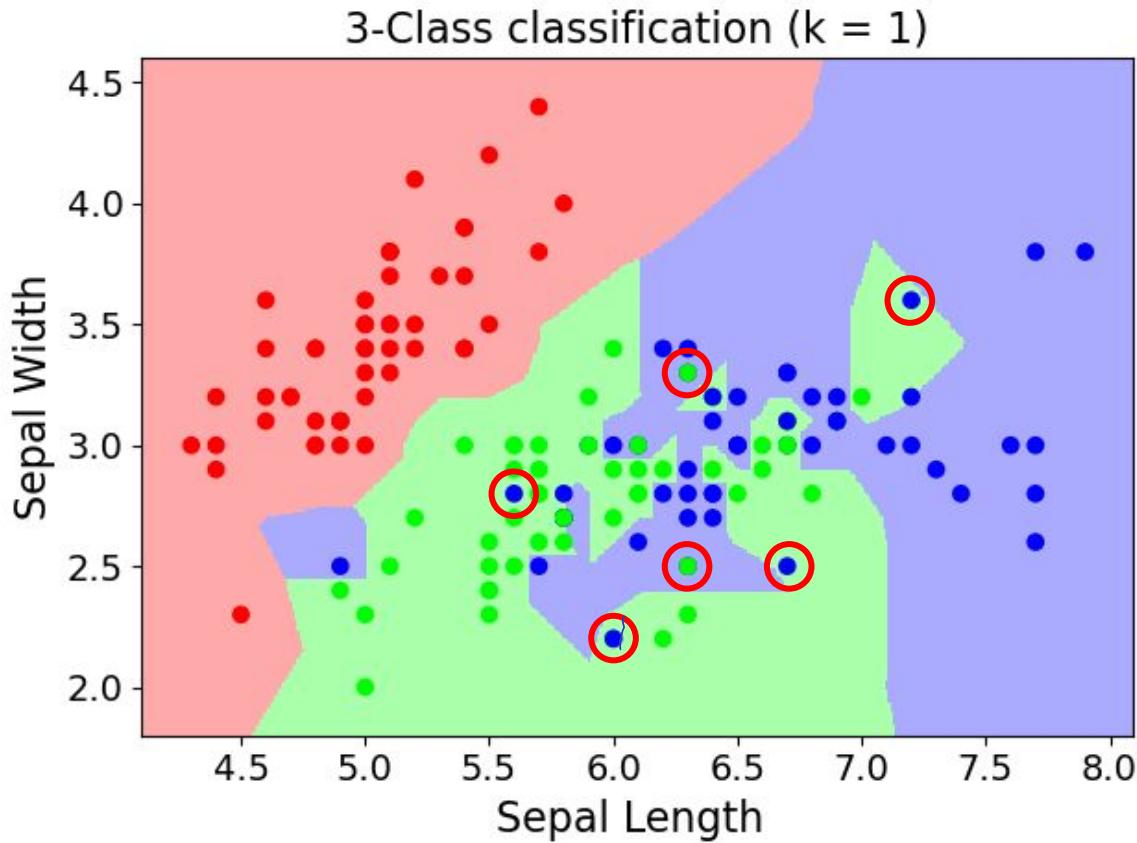


# KNN for Classification — Example



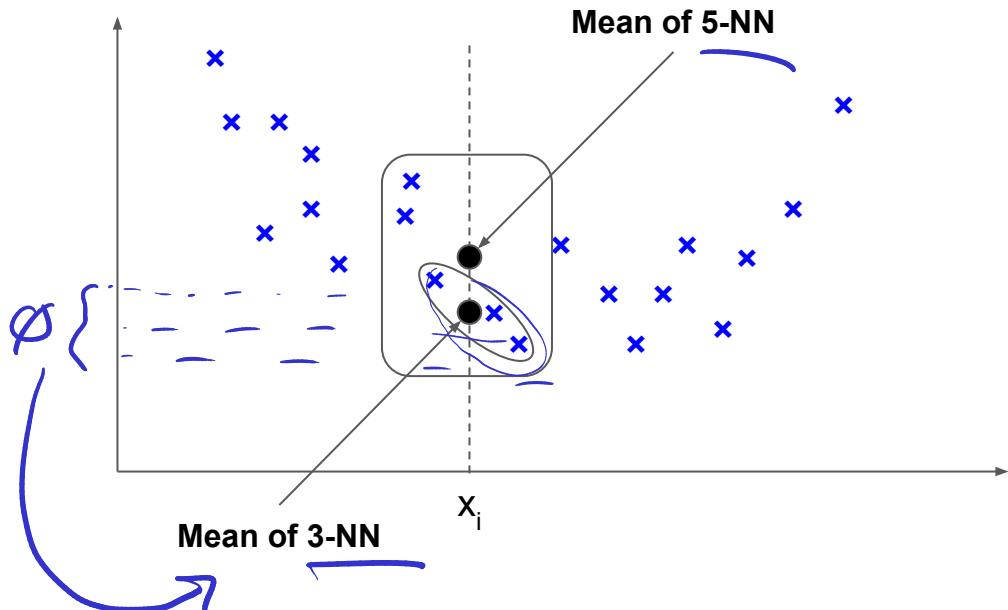
# Quick Quiz

All data points come  
from the **training data**:  
What's going on there?



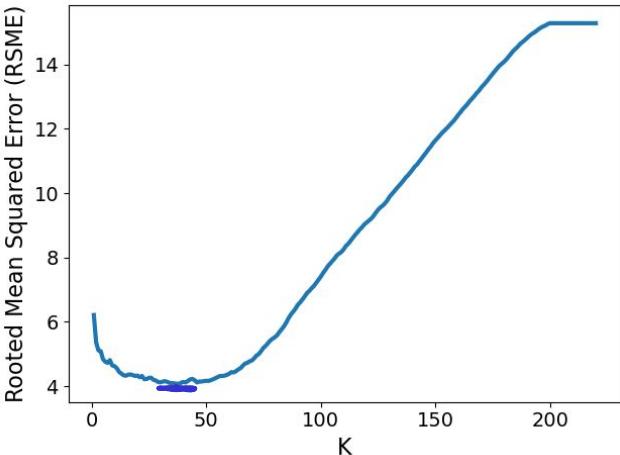
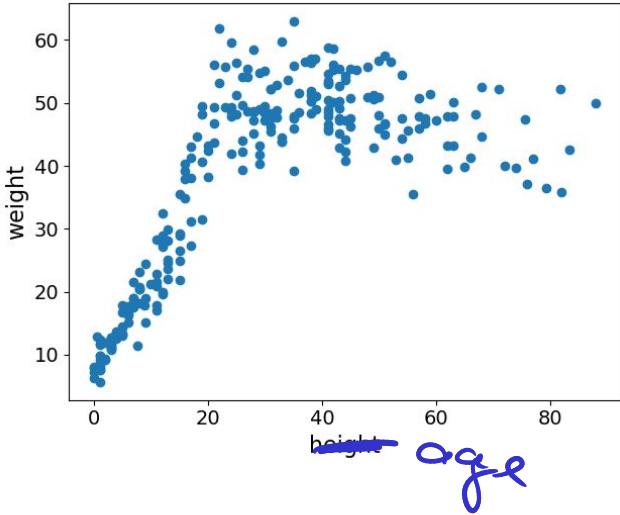
# KNN for Regression

- "Training"
  - Remember training data
- Prediction for unseen point  $x_i$ 
  - Calculate distances to all training data points  $x_j$ , e.g.:
$$d(p, q) = \sqrt{\sum_i^d (q_i - p_i)^2}$$
Euclidean distance,  $d = \# \text{features}$
  - Get the  $k$ -nearest neighbors
  - Label of  $x_i$  = mean of scores of all  $k$ -nearest neighbors

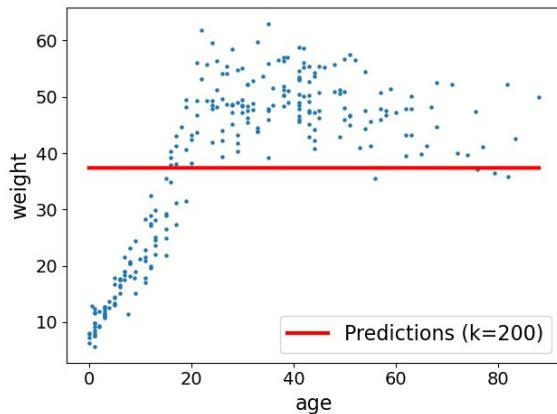
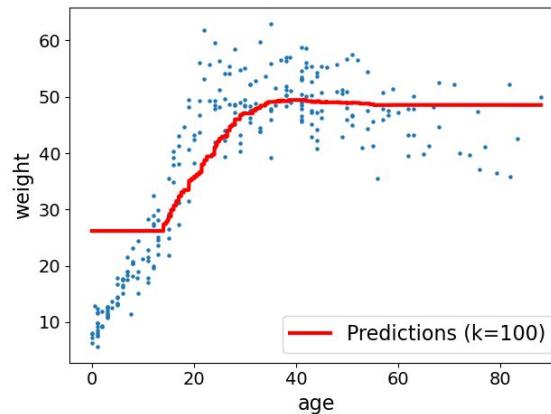
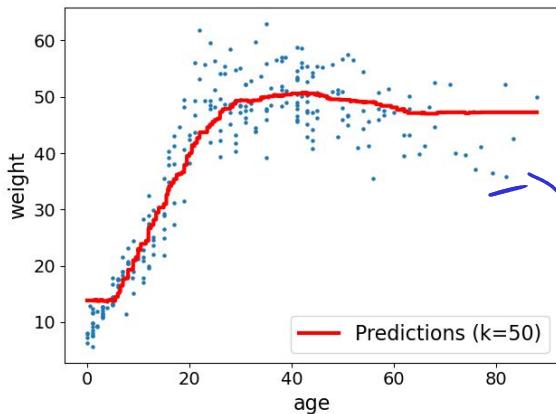
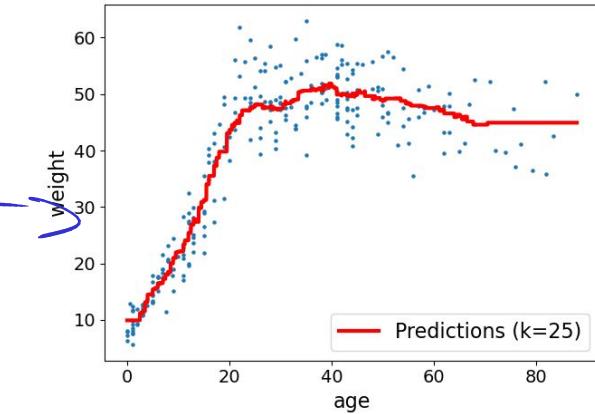
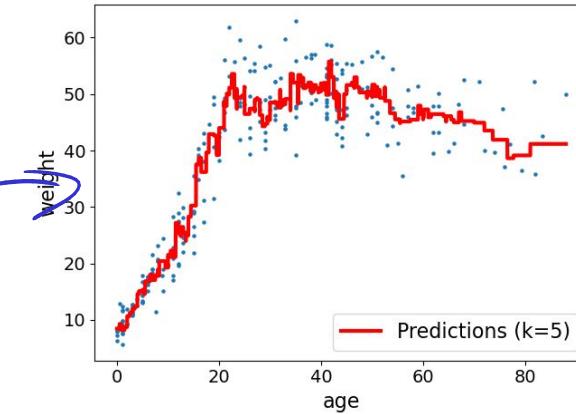
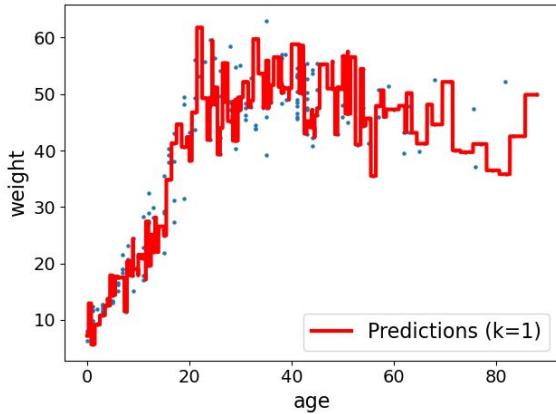


# KNN for Regression — Example

- "age vs. weight" dataset
  - age and weight of 257 males
  - 200 training samples, 57 test samples
- Common outcomes (again)
  - Different  $k$  yield different scores
  - (Very) small  $k$  — predicted scores very sensitive to noise and outliers
  - Large  $k$  — mean over too many neighbors
  - Very large  $k$  — predicted scores converge to the mean over



# KNN for Regression — Example



# Choice of $k$ — Summary

- $k$  too small
    - Predictions sensitive to noise/outliers
    - Very uneven decision boundaries  
(or regression lines)
- }
- Risk of **overfitting**
- 
- $k$  too large
    - Unable to capture local patterns
    - Very smooth decision boundaries  
(or regression lines)
- }
- Risk of **underfitting**

# Outline

- Classification & Regression
  - Overview & Examples
  - Basic Setup
  - Evaluation
- Nearest Neighbor Methods
  - KNN — K-Nearest Neighbor Algorithm
  - Pros, Cons & Caveats

# Pros & Cons...and Caveats

- Pros

- Very simple and intuitive algorithm — but often surprisingly good performance!
- Generic algorithm — applicable as long as distance between points can be "meaningfully" measured
- Can produce arbitrarily shaped decision boundaries
- No training time

} Clustering

- Cons

- Finding neighbors at test time may be slow  
(in practice: data structures and auxiliary algorithms to speed up k-NN search)
- All training data need to be stored

- Caveats

- "...applicable as long as distance between points can be "meaningfully" measured"

# Caveat 1: Feature Values (Range, Magnitude)

- Euclidean distance sensitive to range and magnitude of attribute values

| id | weight (kg) | height (cm) | sex    |
|----|-------------|-------------|--------|
| A  | 80          | 165         | male   |
| B  | 55          | 180         | female |
| C  | 70          | 180         | ???    |

| id | weight (kg) | height (m) | sex    |
|----|-------------|------------|--------|
| A  | 80          | 1.65       | male   |
| B  | 55          | 1.80       | female |
| C  | 70          | 1.80       | ???    |

$$d(A, C) = \sqrt{(70 - 80)^2 + (180 - 165)^2} = 18.0$$

$$d(B, C) = \sqrt{(70 - 55)^2 + (180 - 180)^2} = 15.0$$

→ C classified as female

$$d(A, C) = \sqrt{(70 - 80)^2 + (1.80 - 1.65)^2} = 10.0$$

$$d(B, C) = \sqrt{(70 - 55)^2 + (1.80 - 1.80)^2} = 15.0$$

→ C classified as male

# Data Normalization

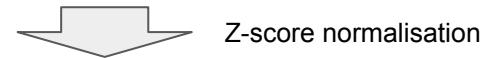
- Z-score normalisation, e.g.:

$$x_i^{\text{weight}} = \frac{x_i^{\text{weight}} - \mu^{\text{weight}}}{\sigma^{\text{weight}}}$$

- Min-max normalization, e.g.:

$$x_i^{\text{weight}} = \frac{x_i^{\text{weight}} - \min(x^{\text{weight}})}{\max(x^{\text{weight}}) - \min(x^{\text{weight}})}$$

| id | weight (kg) | height (m) | sex    |
|----|-------------|------------|--------|
| A  | 80          | 1.65       | male   |
| B  | 55          | 1.80       | female |
| C  | 70          | 1.80       | ???    |



Z-score normalisation

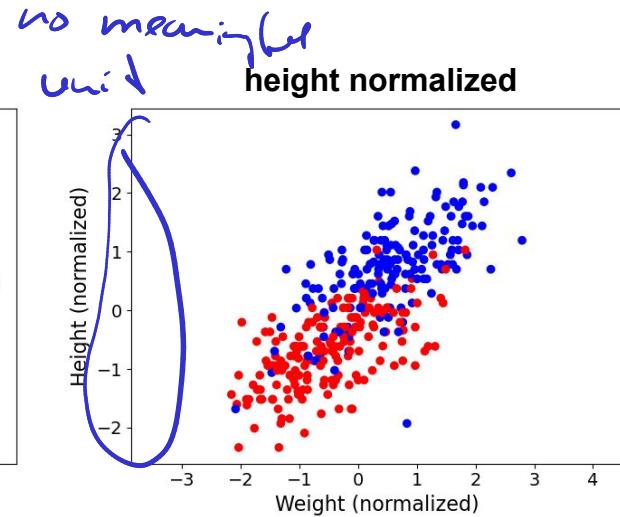
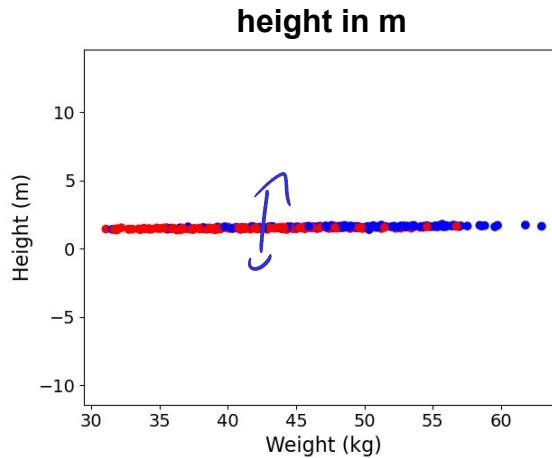
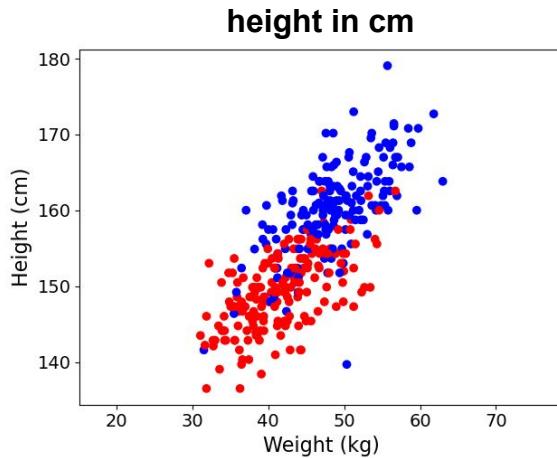
| id | weight | height | sex    |
|----|--------|--------|--------|
| A  | 1.14   | -1.41  | male   |
| B  | -1.30  | 0.71   | female |
| C  | 0.16   | 0.71   | ???    |

$$d(A, C) = 2.33, \quad d(B, C) = 1.46$$

→ C classified as female

# Data Normalization — Visualized

- (weight, height) data points for 187 females (red) and 165 males (blue)

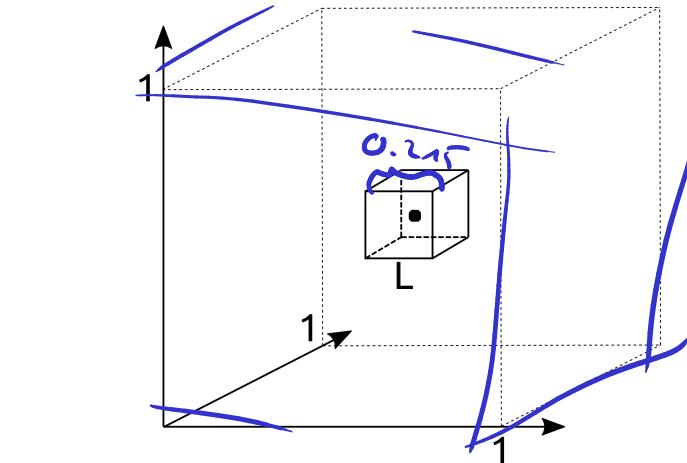


**Note:** Normalization assumes that all features are equally important. This may not always be true!

# Caveat 2: Curse of Dimensionality

- Effect of high dimensions (i.e., many features)
  - Data points tend to never be close together
  - Average distance between points converges
- Intuition
  - Assume  $N$  data points uniformly distributed within a unit cube with  $d$  dimensions
  - Let  $L$  be the length of the smallest cube containing the  $k$ -NN of a data point

$$L^d \approx \frac{k}{N} \Rightarrow L \approx \left(\frac{k}{N}\right)^{\frac{1}{d}}$$



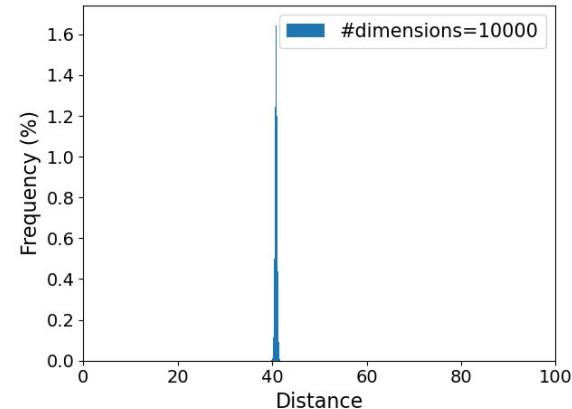
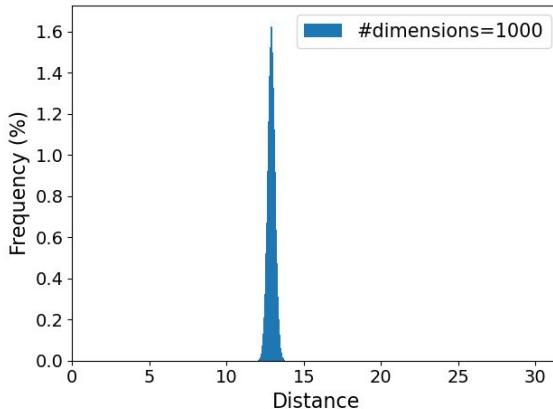
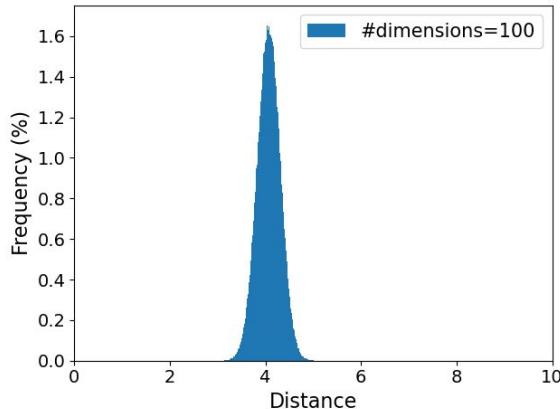
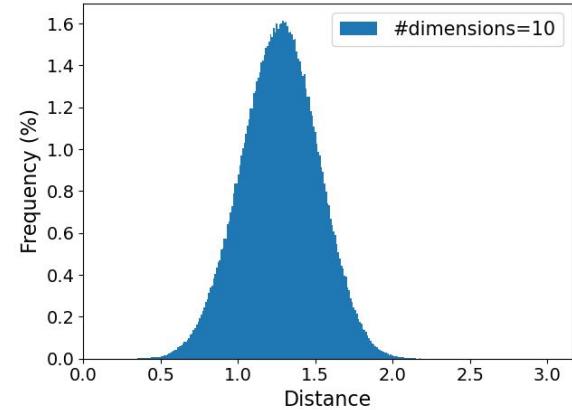
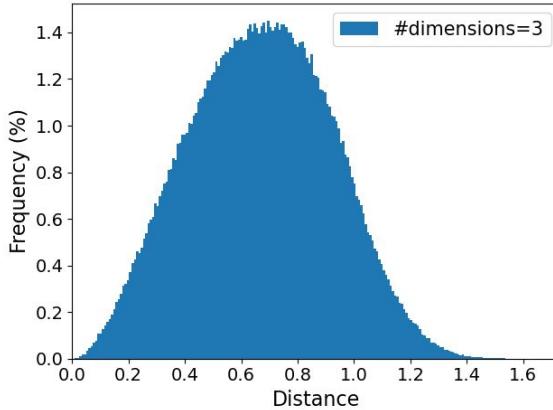
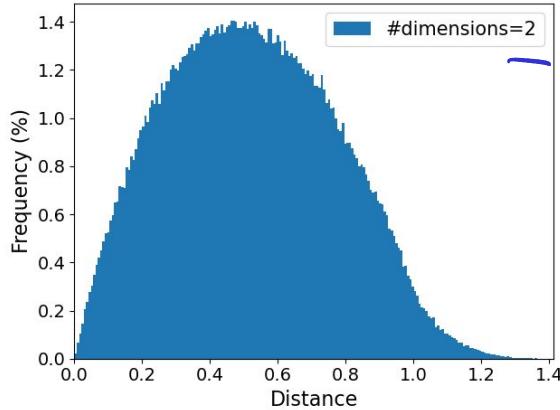
N=1,000, k=10

| $d$    | $L$   |
|--------|-------|
| 2      | 0.100 |
| 3      | 0.215 |
| 10     | 0.631 |
| 100    | 0.955 |
| 1,000  | 0.995 |
| 10,000 | 0.999 |

The cube with the  $k$ -NN is almost the whole unit cube!

# Curse of Dimensionality

Distribution of pairwise distances between 1,000 random data points and different number of dimensions



# Caveat 3: Non-Numerical Data

- In practice: Features often categorical
  - Ordinal — education level, grades, etc.
  - Nominal — sex, marital status, etc.
- Basic approach: Convert categorical features into numerical features
  - Allows direct use of common distance metrics
  - Does not automatically yield good results

| Age | Sex    | Edu-<br>cation | Marital<br>Status |
|-----|--------|----------------|-------------------|
| 23  | male   | Masters        | Single            |
| 35  | male   | College        | Married           |
| 26  | female | Masters        | Single            |
| 41  | female | PhD            | Single            |
| 18  | female | Poly           | Single            |
| 55  | male   | Poly           | Divorced          |
| 30  | female | College        | Single            |
| 35  | make   | PhD            | Married           |
| 28  | male   | Masters        | Married           |
| 45  | female | Masters        | Married           |

# Caveat 3: Non-Numerical Data

- **Binary features**

- Special case of categorical data
- Convert into binary variable 0/1 to indicate absence/presence



| Sex    |
|--------|
| male   |
| male   |
| female |
| female |
| female |

| Sex |
|-----|
| 0   |
| 0   |
| 1   |
| 1   |
| 1   |

- **Ordinal features**

- Utilize natural order of feature values
- Convert into numerical values while preserving their original order



| Education |
|-----------|
| Masters   |
| College   |
| Masters   |
| PhD       |
| Poly      |

| Education |
|-----------|
| 4         |
| 2         |
| 4         |
| 5         |
| 1         |

# Caveat 3: Non-Numerical Data

- Nominal features (with  $n$  different values)
  - One-hot encoding: convert nominal feature into  $N$  binary features
  - Convert each new binary feature into binary variable 0/1 to indicate absence/presence
  - Increases dimensionality!  
(particularly if  $N$  is very large)

| Marital Status |
|----------------|
| Single         |
| Married        |
| Single         |
| Single         |
| Single         |
| Divorced       |



| Single | Married | Divorced |
|--------|---------|----------|
| 1      | 0       | 0        |
| 0      | 1       | 0        |
| 1      | 0       | 0        |
| 1      | 0       | 0        |
| 1      | 0       | 0        |
| 0      | 0       | 1        |

# Caveat 3: Non-Numerical Data

- Discussion / Limitations

- One-hot encoding increases dimensionality of data
- Distance between ordinal values often not intuitive

$$5 \text{ (PhD)} - 4 \text{ (Masters)} = 2 \text{ (College)} - \text{Poly (1)}$$

- Typically requires data normalization  
(particularly when combined with numerical features)

| Education |
|-----------|
| Masters   |
| College   |
| Masters   |
| PhD       |
| Poly      |



| Education |
|-----------|
| 4         |
| 2         |
| 4         |
| 5         |
| 1         |

- Alternative approach: Custom distance metric

- Design metric that appropriately quantifies distance between categorical values
- Typically very specific to given dataset and application context.

# Caveat 4: Semantic vs. Low-Level Similarity

- Unstructured data  
(text, image, video, audio)
  - Object is often more than just the sum of its parts
- Example (dog food ad)
  - Pixelwise similarity large; easy to compute
  - Semantic similarity small

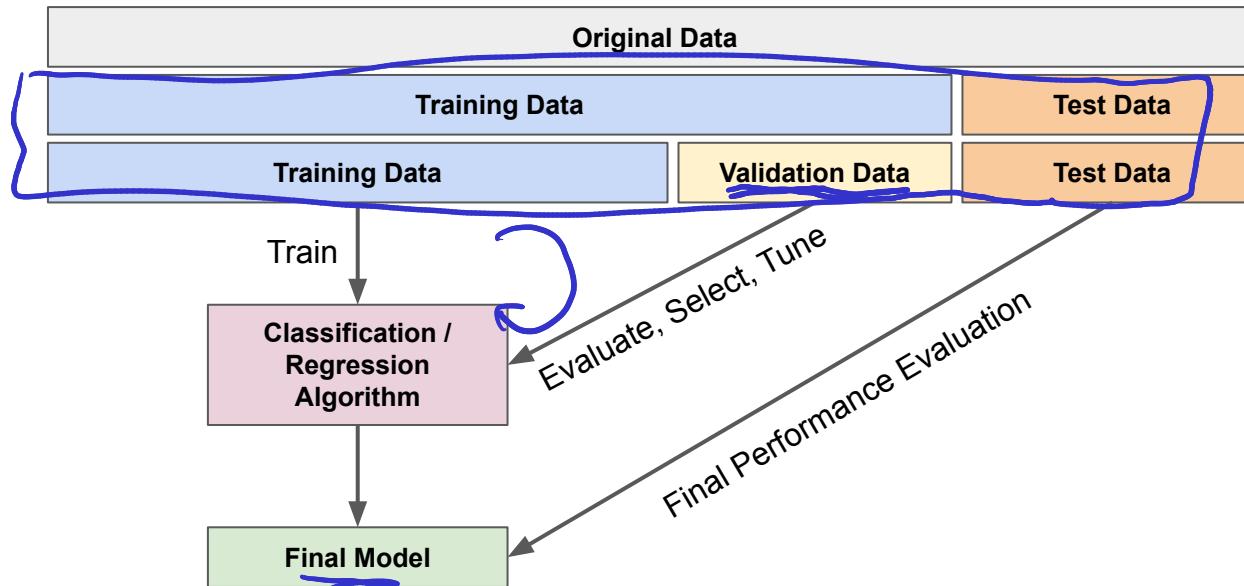


# Supervised Training/Learning — Extended Setup

- Recall basic setup: training data + test data
  - Building a good classifier or regressor
    - Find the best data preprocessing steps
    - Find the best model (model selection)
    - Find the best hyperparameter values
  - **Important:** Not allowed to use test data for this!
    - Test data is supposed to contain truly unseen data
    - Test data no longer unseen when used to optimize/tune model  
(for example, results might be different for a different training-test split)
- 
- The diagram illustrates the iterative nature of building a good classifier or regressor. It shows a brace grouping the three sub-points under the heading 'Building a good classifier or regressor'. An arrow points from this brace to the text 'Iterative evaluation required', indicating that each step in this process must be evaluated iteratively.

# Training & Evaluation Process Using Validation Data

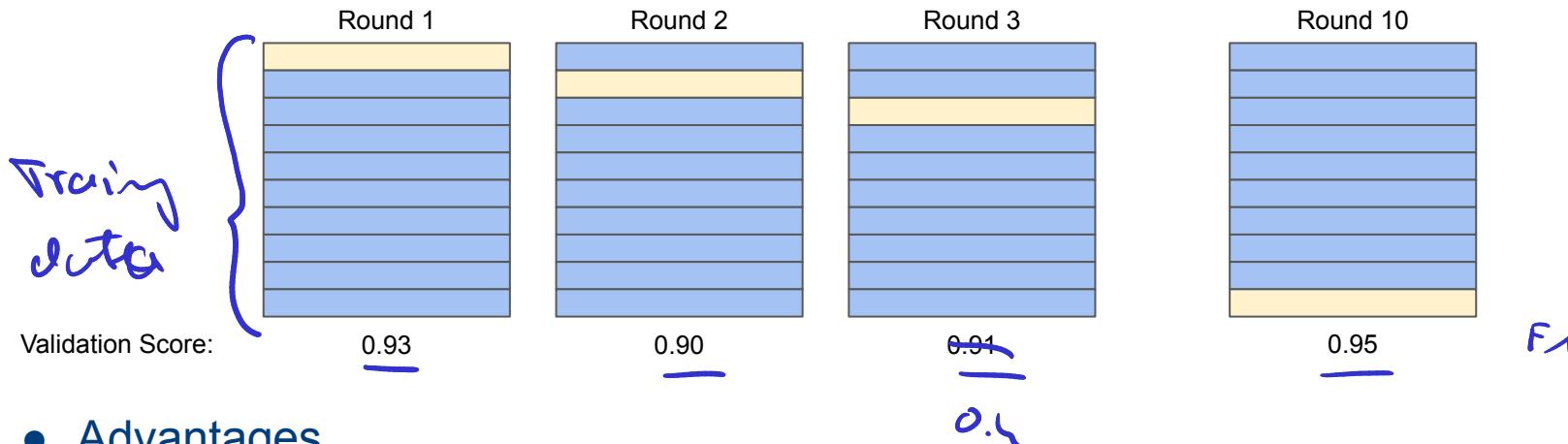
- Common data split to ensure generalizability of results
  - Use test data only at the very end to evaluate performance of final model
  - Use validation data for model selection and hyperparameter tuning



# K-Fold Cross Validation (just one of many validation techniques)

- Core idea of  $k$ -fold cross validation

- Split training data into  $k$  blocks of equal size (e.g.,  $k=10$ )
- Use  $(k-1)$  blocks for training and remaining block for evaluation
- Repeat for  $k$  rounds with different permutations



- Advantages

- Averaged results more reliable
- Variance between results useful indicator

# Information Leakage through Normalization

- Important guidelines
  - Do not normalize before splitting dataset into training and test data!
  - Normalize training and test data but only based on training data!
- Example (pseudo) code using scikit-learn

Why?

```
# Split input data into training and test set
X_train, X_test = split(X, 0.2)

# Fit StandardScaler (i.e., calculate mean and variance)
scaler = preprocessing.StandardScaler().fit(X_train) # CORRECT!
#scaler = preprocessing.StandardScaler().fit(X) # WRONG!!!

# Fit both training and test data
X_train_transformed = scaler.transform(X_train)
X_test_transformed = scaler.transform(X_test)
```

# Summary

- **Evaluation of classifiers** (straightforward for regressors)
  - Different metric with different interpretations
  - Applicability of metrics depending on data and task
- **K-Nearest Neighbor (KNN) classifier/regressor**
  - Very intuitive supervised learning model
  - Limited applicability for (very) large datasets  
(heavy lifting done during prediction time not the training time)
  - Choice of hyperparameter  $k$  important to address risk of overfitting and underfitting

# Solutions to Quick Quizzes

- Slide 29: D
- Slide 30: Technically D, but A is the more practical result
- Slide 37: B
- Slide 41: B
- Slide 47: Duplicate data points
- Slide 67: Otherwise, risk of data leakage
  - Mean and standard deviation will be affected by test data

# **CS5228: Knowledge Discovery and Data Mining**

## Lecture 6 — Classification & Regression II

# Course Logistics — Update

- Assignment 2
  - Submission deadline: Oct 03, 11.59 pm
  - Don't forget to check the Discussion and Errata page on Canvas
- Midterm
  - Check new Canvas page for midterm exam
  - Report any issues with the practice exam early enough
  - If needed, 2nd practice exam during Recess Week
  - Coming up: survey regarding request for loaner laptop
- Project
  - Submission deadline for progress report: Oct 10, 11.59 pm

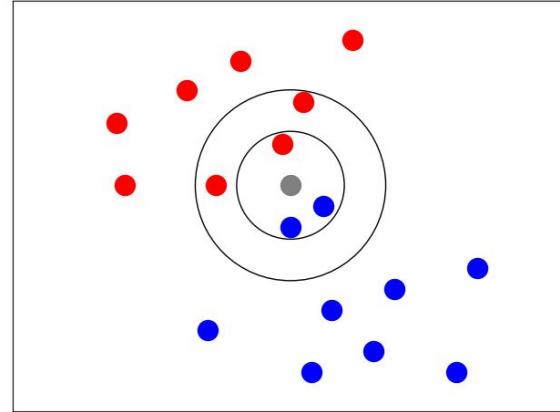
# Quick Recap — Classification & Regression

- Pattern of interest
  - Matching or function between input features and output
  - Goal: use matching to predict outputs for unseed samples
  - Categorical output → classification
  - Numerical output → regression
- Important: Evaluation of predictions
  - Straightforward for regression
  - Series of metrics for classification  
(accuracy, recall, precision, f1, AUC-ROC)

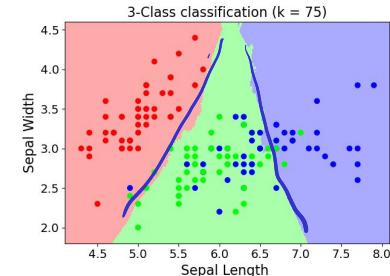
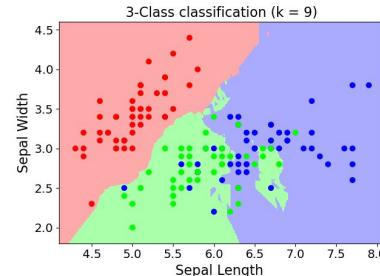
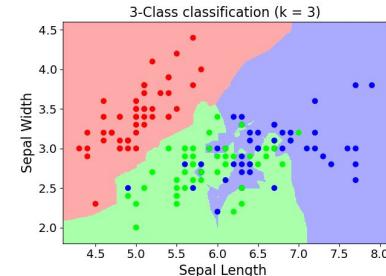
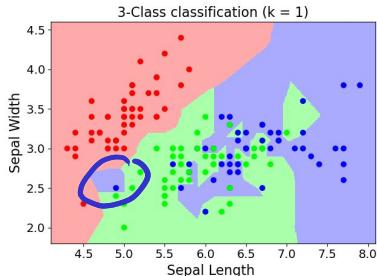
| Age | Edu-<br>cation | Marital<br>Status | Income<br>Level | Credit<br>Approval | Credit<br>Limit |
|-----|----------------|-------------------|-----------------|--------------------|-----------------|
| 23  | Masters        | Single            | Mid             | No                 | \$S5,000        |
| 35  | College        | Married           | High            | Yes                | \$S7,000        |
| 26  | Masters        | Single            | High            | No                 | \$S9,000        |
| 41  | PhD            | Single            | Mid             | Yes                | \$S5,000        |
| 18  | Poly           | Single            | Low             | No                 | \$S6,000        |
| 55  | Poly           | Married           | High            | Yes                | \$S10,000       |
| 30  | College        | Single            | High            | Yes                | \$S8,000        |
| 35  | PhD            | Married           | High            | Yes                | \$S10,000       |
| 28  | Masters        | Married           | Mid             | Yes                | \$S5,000        |
| 45  | Masters        | Married           | Mid             | ???                | ???             |

# Quick Recap — KNN Algorithm

- Intuition behind KNN:
  - Label of an unseen data point  $x$  derives from the labels of the  $k$ -nearest neighbors of  $x$
  - Similar data points → similar labels
  - Caveats due to reliance of similarity metric



- Effects of hyperparameter  $k$ 
  - Tradeoff between (risks of) underfitting and overfitting



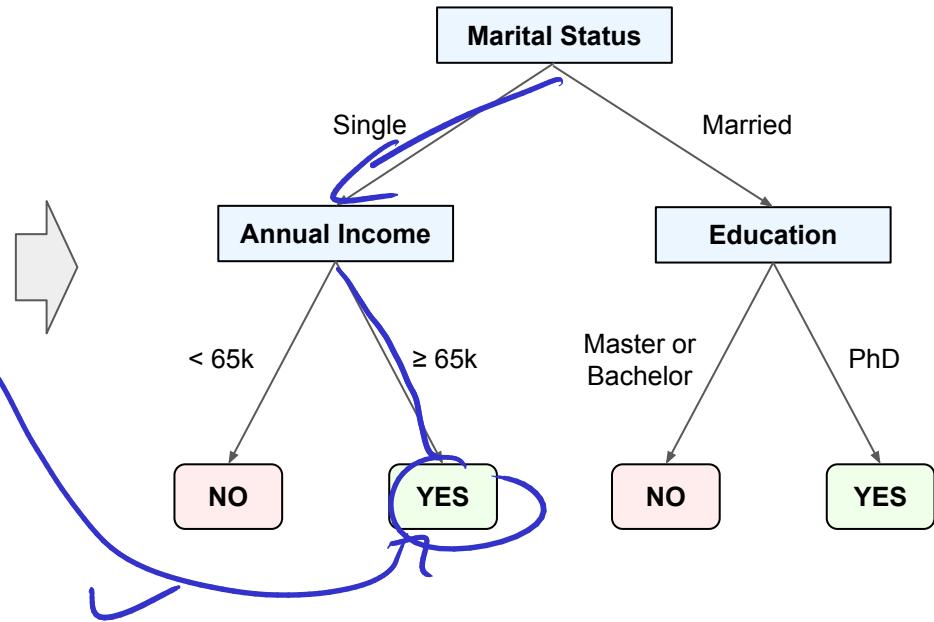
# Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - Random Forest
  - Boosting

# Decision Tree

- Example: Decision Tree for classification

| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 23  | Masters   | Single         | 75k           | Yes             |
| 35  | Bachelor  | Married        | 50k           | No              |
| 26  | Masters   | Single         | 70k           | Yes             |
| 41  | PhD       | Single         | 85k           | Yes             |
| 18  | Bachelor  | Single         | 40k           | No              |
| 55  | Masters   | Married        | 85k           | No              |
| 30  | Bachelor  | Single         | 60k           | No              |
| 35  | PhD       | Married        | 60k           | Yes             |
| 28  | PhD       | Married        | 65k           | Yes             |



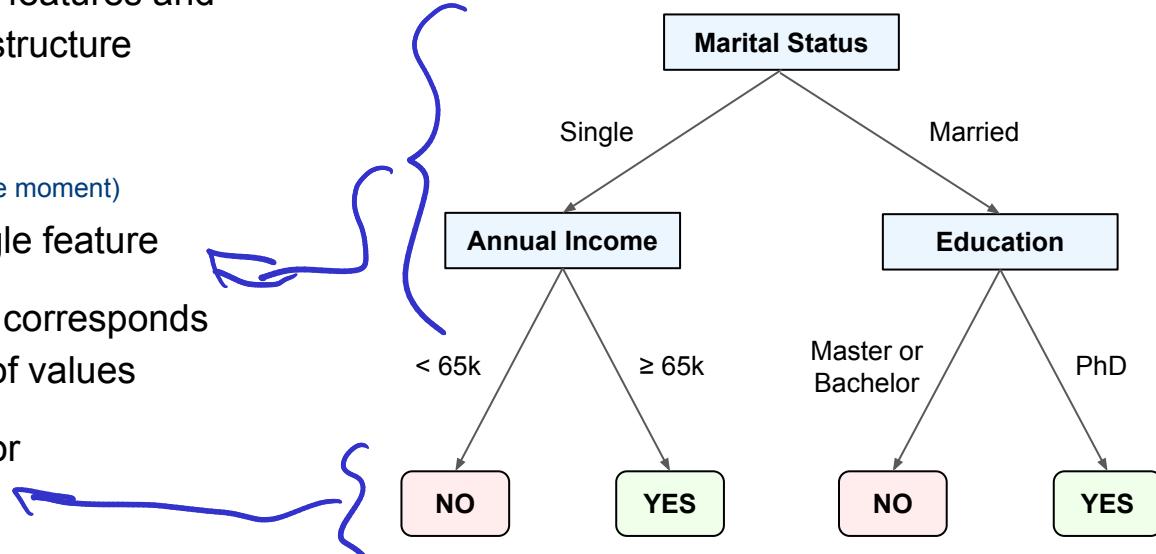
# Decision Tree

- Decision Tree — idea

- Represent mapping between features and label/value as flowchart-like structure

- Components (a bit simplified at the moment)

- (Inner) node — test on a single feature
- Branch — outcome of a test; corresponds to a feature values or range of values
- Leaf — label (classification) or real value (regression)



# Decision Tree — Application to Unseen Data

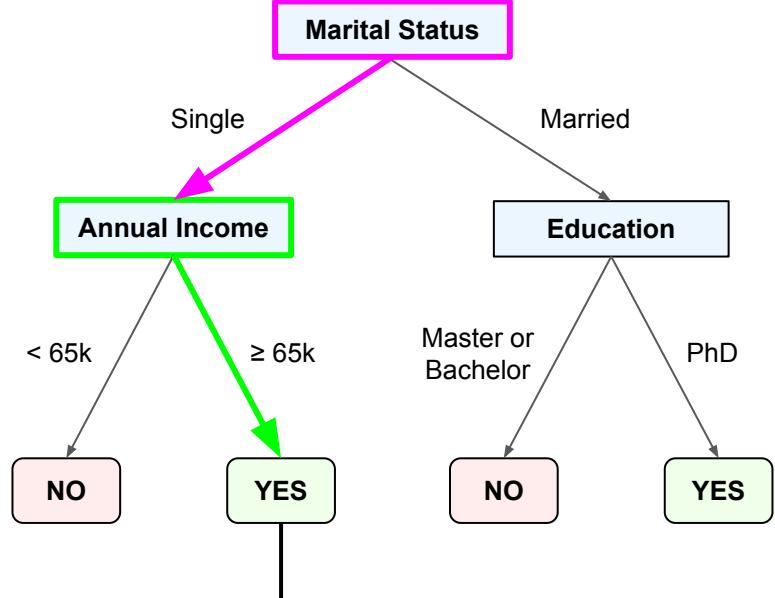
| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 50  | PhD       | Single         | 70k           | ???             |



| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 50  | PhD       | Single         | 70k           | ???             |



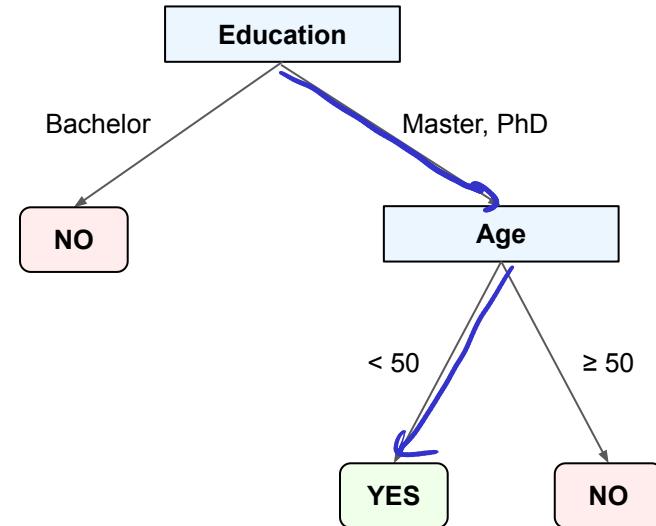
| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 50  | PhD       | Single         | 70k           | YES             |



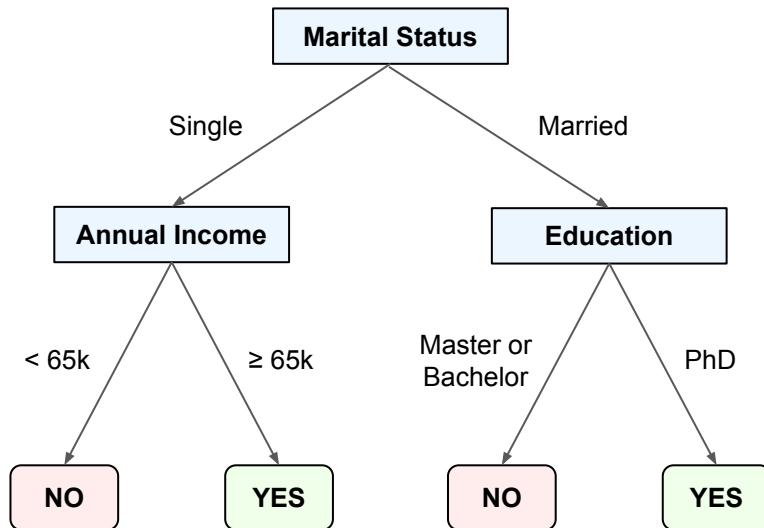
# Decision Tree

- Same dataset, different Decision Tree
  - In general, there are multiple trees that match a dataset

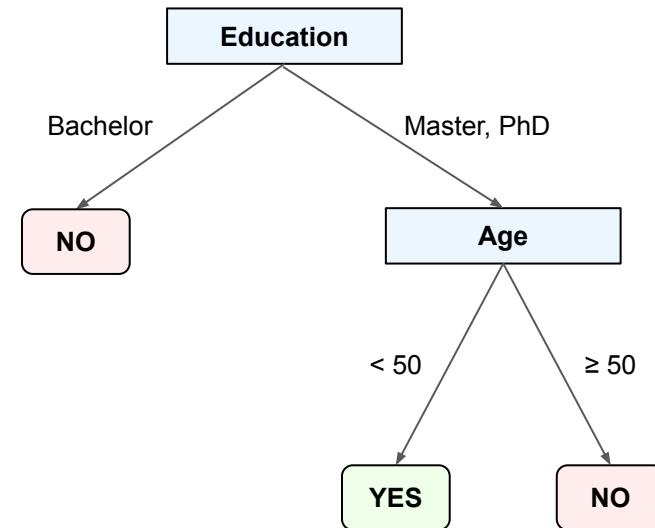
| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 2  | 35  | Bachelor  | Married        | 50k           | No              |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 5  | 18  | Bachelor  | Single         | 40k           | No              |
| 6  | 55  | Masters   | Married        | 85k           | No              |
| 7  | 30  | Bachelor  | Single         | 60k           | No              |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



# Which Decision Tree is Better?



| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 50  | PhD       | Single         | 70k           | Yes             |



| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 50  | PhD       | Single         | 70k           | No              |

# Quick Quiz

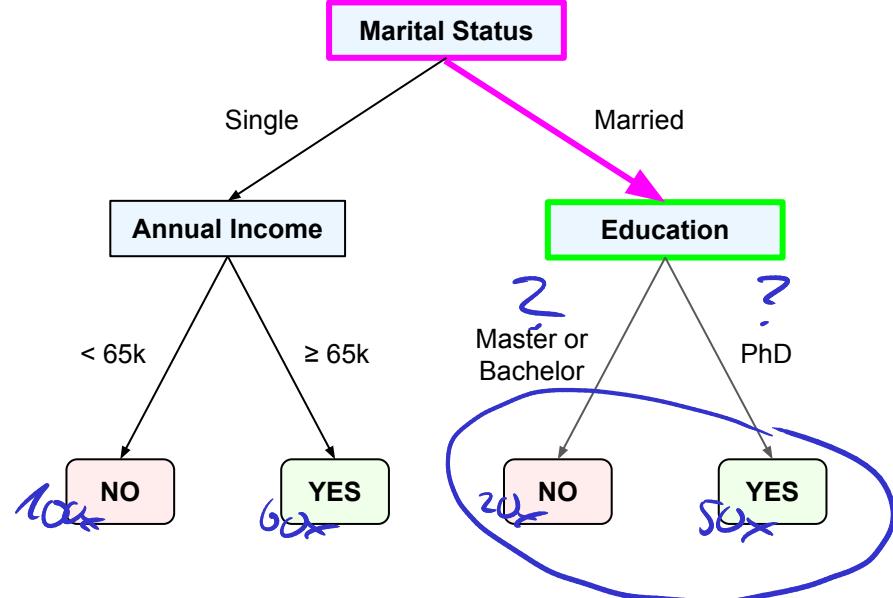
| Age | Edu-<br>cation | Marital<br>Status | Annual<br>Income | Credit<br>Approval |
|-----|----------------|-------------------|------------------|--------------------|
| 50  | PhD            | Married           | 70k              | ???                |



| Age | Edu-<br>cation | Marital<br>Status | Annual<br>Income | Credit<br>Approval |
|-----|----------------|-------------------|------------------|--------------------|
| 50  | Poly           | Single            | 70k              | ???                |

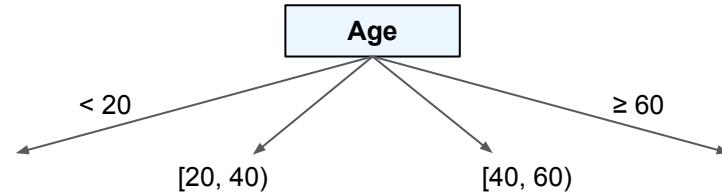
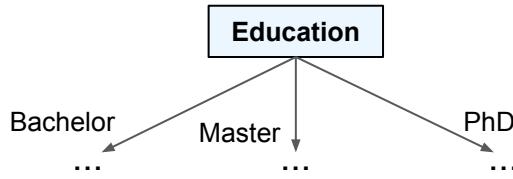


What to do in case  
of **unknown values**  
for a feature?



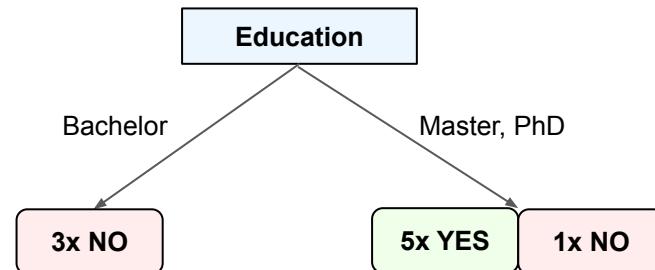
# Decision Tree — Diversity (Sneak Preview)

- Different branching factors



- Different depth

- Leaves can have more than one label or real value 
- Required based on dataset or based on choice (→ Pruning)
- Final output: majority label (classification) or mean of values (regression)



# Outline

- **Decision Trees**
  - Overview
  - **Training Decision Trees**
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - Random Forest
  - Boosting

# Building a Decision Tree

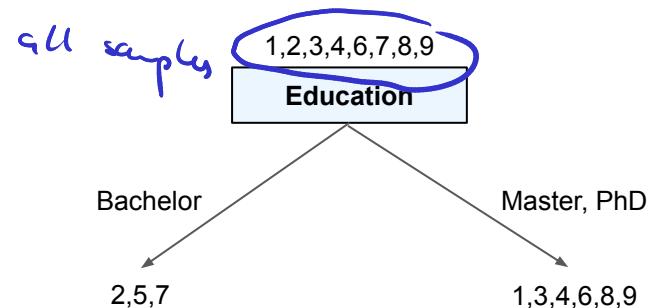
- Notations

- $D_t$  — set of records that reach node  $t$
- $D_0$  — set of all records at root node

- General procedure

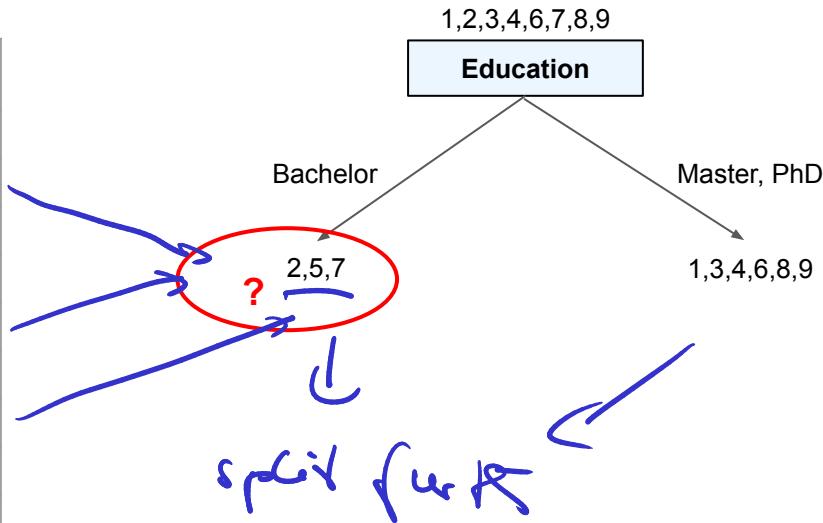
- If  $|D_t| = 1$  or all records in  $D_t$  have the same class or value  $\rightarrow t$  is leaf node
- Otherwise, choose test (feature + conditions) to split  $D_t$  into smaller subsets (i.e., subtrees)
- Recursively apply procedure to each subtree

| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 2  | 35  | Bachelor  | Married        | 50k           | No              |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 5  | 18  | Bachelor  | Single         | 40k           | No              |
| 6  | 55  | Masters   | Married        | 85k           | No              |
| 7  | 30  | Bachelor  | Single         | 60k           | No              |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



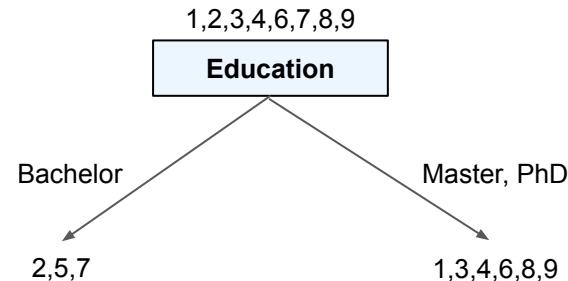
# Building a Decision Tree — Step-by-Step Example

| ID | Age | Edu-cation | Marital Status | Annual Income | Credit Approval |
|----|-----|------------|----------------|---------------|-----------------|
| 1  | 23  | Masters    | Single         | 75k           | Yes             |
| 2  | 35  | Bachelor   | Married        | 50k           | No              |
| 3  | 26  | Masters    | Single         | 70k           | Yes             |
| 4  | 41  | PhD        | Single         | 95k           | Yes             |
| 5  | 18  | Bachelor   | Single         | 40k           | No              |
| 6  | 55  | Masters    | Married        | 85k           | No              |
| 7  | 30  | Bachelor   | Single         | 60k           | No              |
| 8  | 35  | PhD        | Married        | 60k           | Yes             |
| 9  | 28  | PhD        | Married        | 65k           | Yes             |



# Building a Decision Tree — Step-by-Step Example

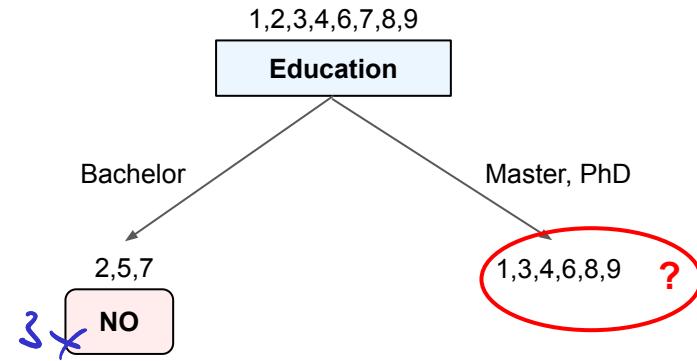
| ID | Age | Edu-cation | Marital Status | Annual Income | Credit Approval |
|----|-----|------------|----------------|---------------|-----------------|
| 2  | 35  | Bachelor   | Married        | 50k           | No              |
| 5  | 18  | Bachelor   | Single         | 40k           | No              |
| 7  | 30  | Bachelor   | Single         | 60k           | No              |
|    |     |            |                |               |                 |
|    |     |            |                |               |                 |
|    |     |            |                |               |                 |
|    |     |            |                |               |                 |
|    |     |            |                |               |                 |



All the same labels → leaf node

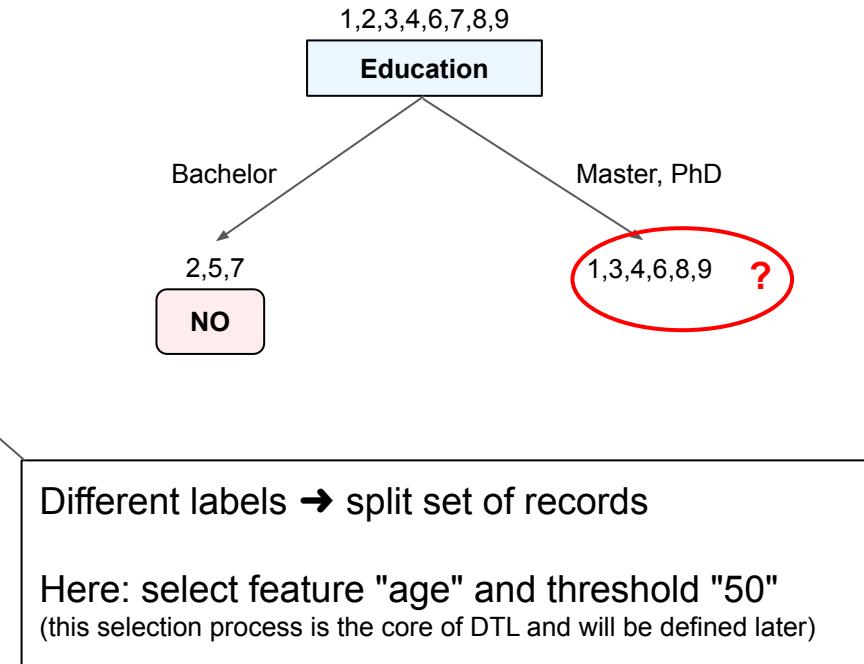
# Building a Decision Tree — Step-by-Step Example

| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 2  | 35  | Bachelor  | Married        | 50k           | No              |
| 5  | 18  | Bachelor  | Single         | 40k           | No              |
| 7  | 30  | Bachelor  | Single         | 60k           | No              |



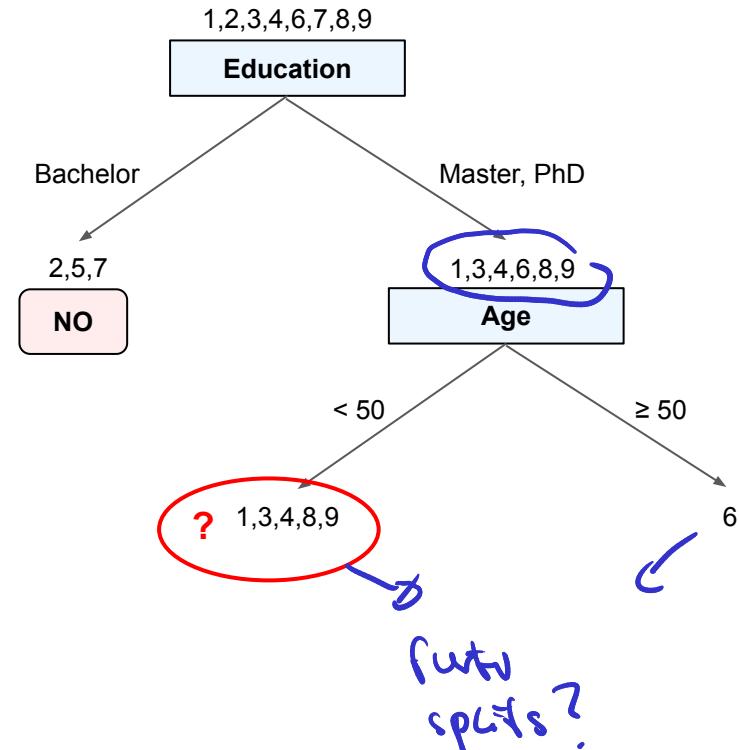
# Building a Decision Tree — Step-by-Step Example

| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 6  | 55  | Masters   | Married        | 85k           | No              |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



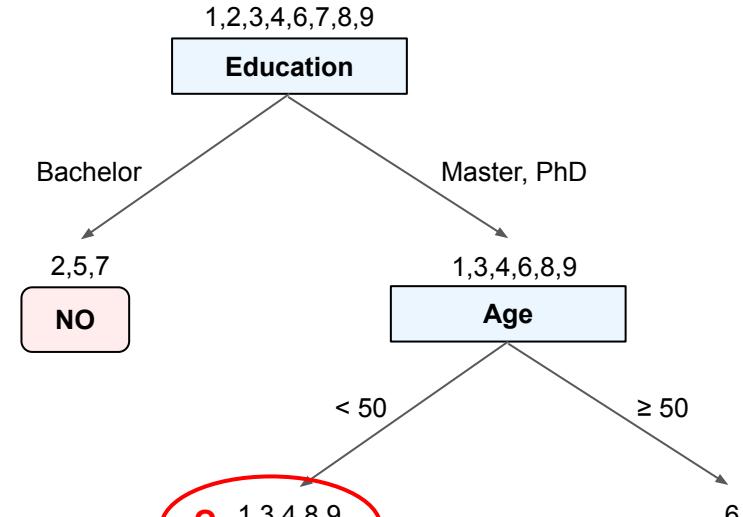
# Building a Decision Tree — Step-by-Step Example

| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 2  | 30  | Masters   | Single         | 60k           | No              |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 5  | 32  | PhD       | Single         | 80k           | No              |
| 6  | 55  | Masters   | Married        | 85k           | No              |
| 7  | 45  | PhD       | Married        | 70k           | Yes             |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



# Building a Decision Tree — Step-by-Step Example

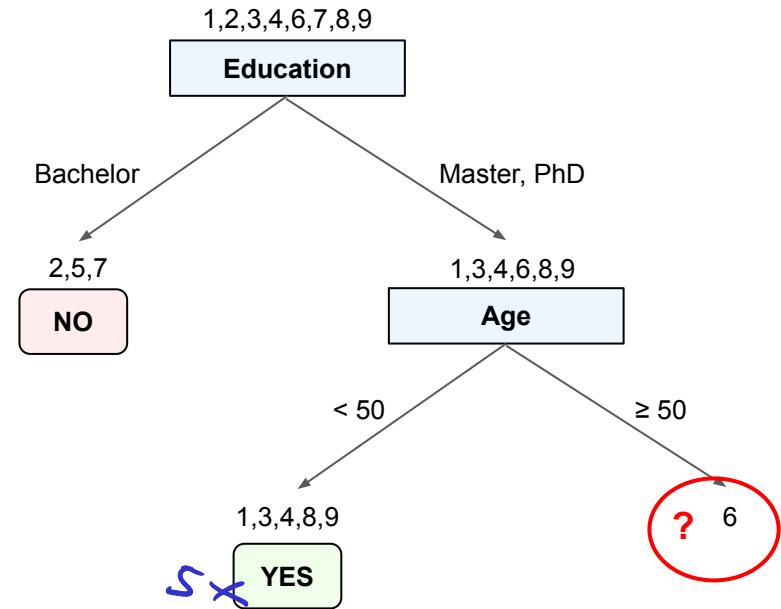
| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



All the same labels → leaf node

# Building a Decision Tree — Step-by-Step Example

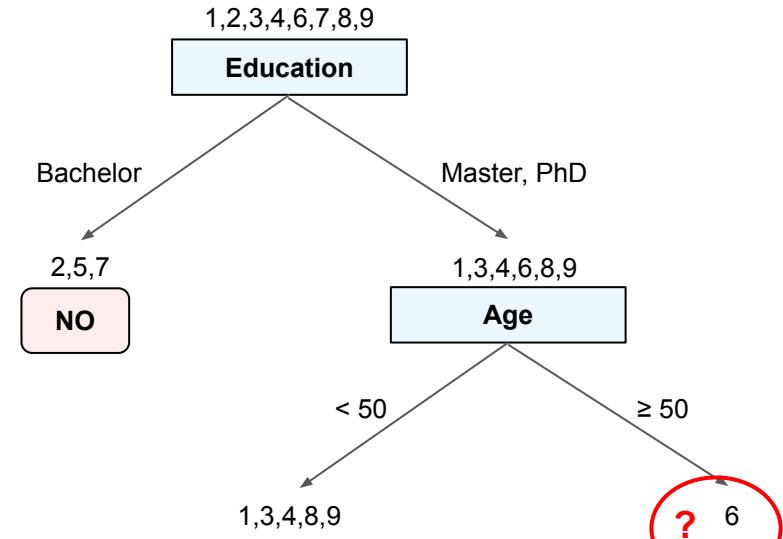
| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
| 1  | 23  | Masters   | Single         | 75k           | Yes             |
| 2  | 25  | Masters   | Single         | 70k           | No              |
| 3  | 26  | Masters   | Single         | 70k           | Yes             |
| 4  | 41  | PhD       | Single         | 95k           | Yes             |
| 5  | 30  | PhD       | Married        | 60k           | No              |
| 6  | 35  | PhD       | Married        | 60k           | Yes             |
| 7  | 28  | PhD       | Married        | 65k           | Yes             |
| 8  | 35  | PhD       | Married        | 60k           | Yes             |
| 9  | 28  | PhD       | Married        | 65k           | Yes             |



# Building a Decision Tree — Step-by-Step Example

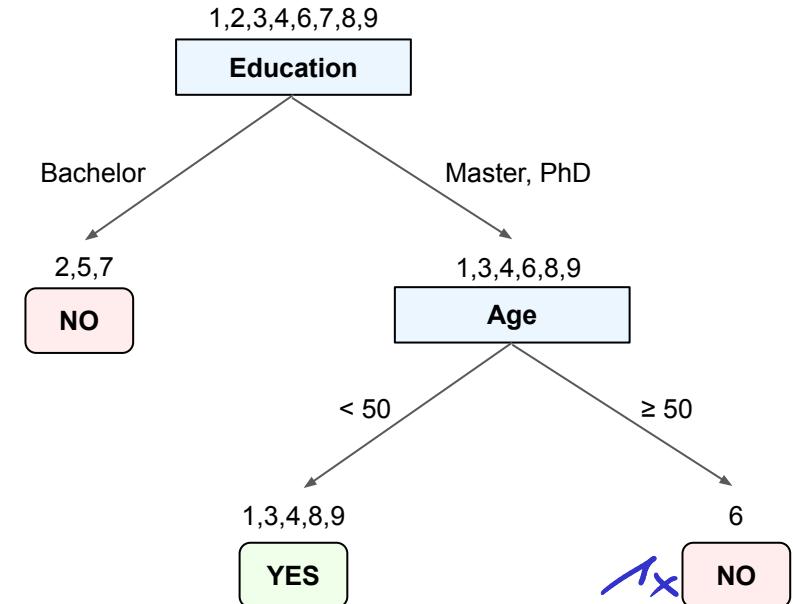
| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
| 6  | 55  | Masters   | Married        | 85k           | No              |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |

Only one record → leaf node



# Building a Decision Tree — Step-by-Step Example

| ID | Age | Education | Marital Status | Annual Income | Credit Approval |
|----|-----|-----------|----------------|---------------|-----------------|
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |
|    |     |           |                |               |                 |

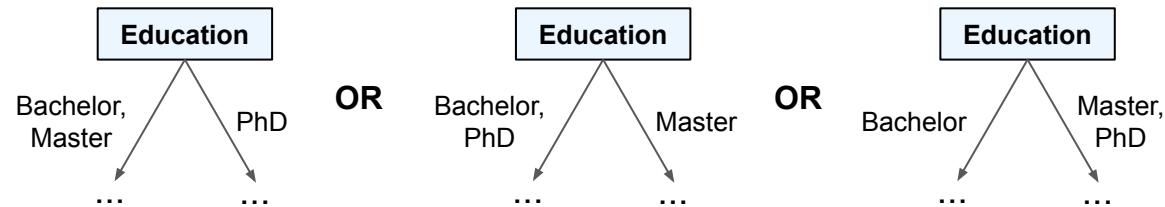


All records covered → Done!

# How to Split? — Nominal Attributes

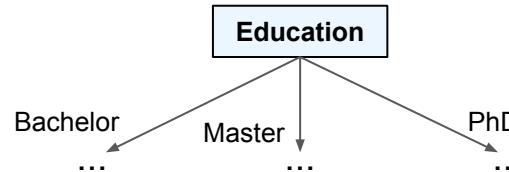
- **Binary split**

- Partition all  $d$  values into two subsets
- $\frac{2^d - 2}{2}$  possible splits



- **Multiway split**

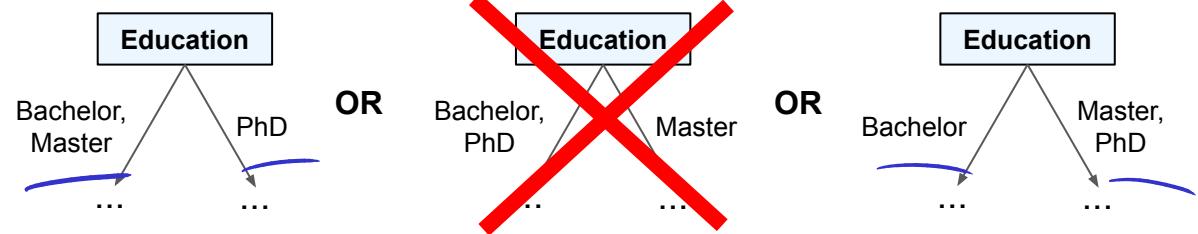
- Each value yields one subtree
- In principle, arbitrary splits into  $2 \leq s \leq d$  subtrees possible, but number of possible splits explodes



# How to Split? — Ordinal Attributes

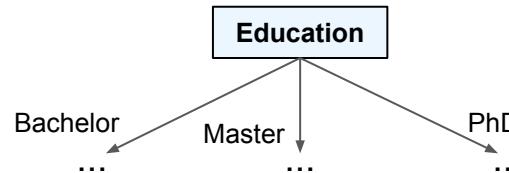
- **Binary split**

- Partition all  $d$  values into two subsets
- Partitions must preserve natural order of values
- $d - 1$  possible splits



- **Multiway split**

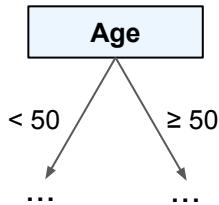
- Each value yields one subtree



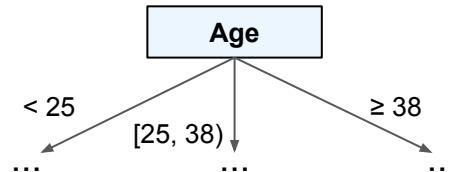
**Note:** Whether "Education" is treated as nominal or ordinal feature is up to interpretation and a design choice of the user

# How to Split? — Numerical Values

- Binary split

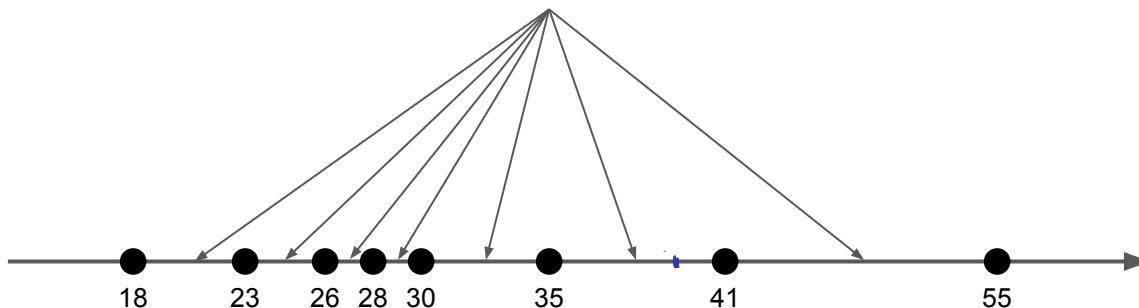


- Multiway split



| Age |
|-----|
| 23  |
| 35  |
| 26  |
| 41  |
| 18  |
| 55  |
| 30  |
| 35  |
| 28  |

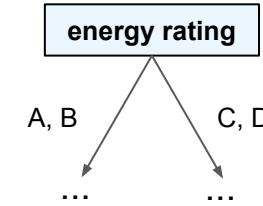
Possible thresholds



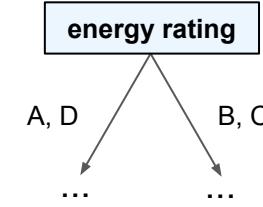
# Quick Quiz

For the **energy rating** of a building,  
what is generally **not** a suitable split?

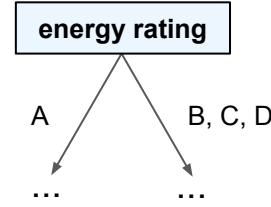
A



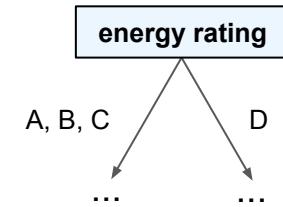
B ✓



C

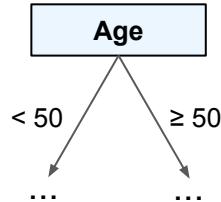


D

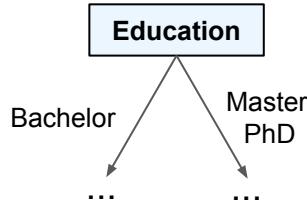


# Finding the Best Splits?

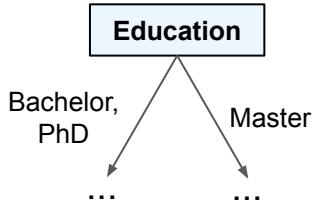
- Which feature to use for splitting the training records



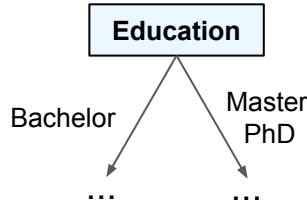
vs.



- How to split the w.r.t. a selected feature?



vs.

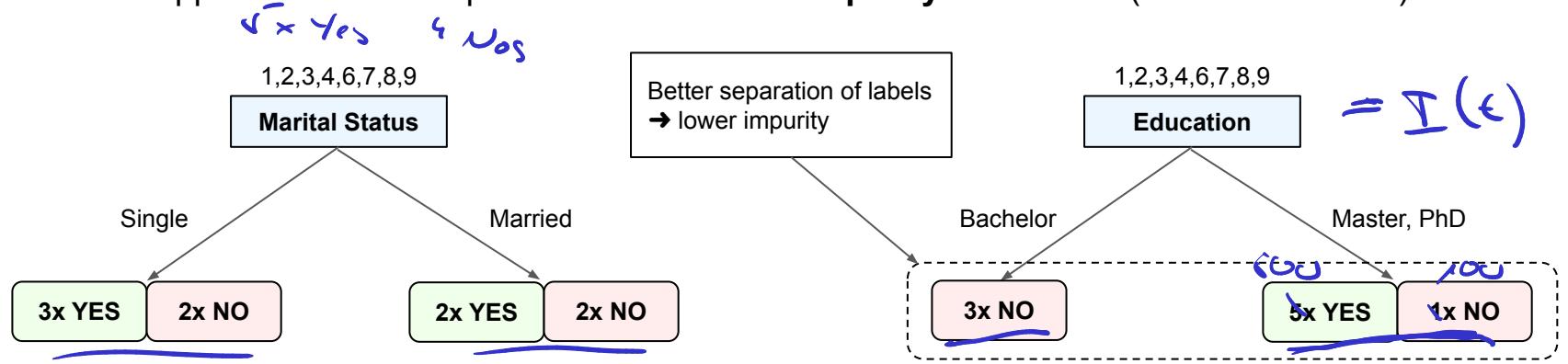


Why choose **Age** over  
**Education**, or vice versa?

What makes **{B,M}/P}** a  
better split than **{B}/{MP}**,  
or vice versa — or any  
other alternative way?

# Finding the Best Splits

- Global optimum
  - Best splits = splits that result in a Decision Tree with the highest accuracy
  - Problem: Finding the optimal tree is NP-complete → not practical for large datasets
- Greedy approach
  - Fast(er) heuristics but no guarantees for optimal results
  - Basic approach: Pick the split that minimizes the **impurity** of subtrees (w.r.t. class labels)



# Finding the Best Splits

- General procedure
  - Calculate impurity  $I(t)$  of node  $t$  before splitting
  - For each possible / considered split, calculate impurity of split  $I_{split}$   
(weighted average of impurities of resulting child nodes)
  - Select split with lowest impurity  $I_{split}$
  - Perform split if  $I_{split} < I(t)$  — (not necessarily always the case)

# Impurity of a Node (Classification)

## Gini Index

$$Gini(t) = 1 - \sum_{c \in C} P(c|t)^2$$

perfectly pure

|        |       |
|--------|-------|
| 0x YES | 6x NO |
|--------|-------|

$$1 - (1.0^2 + 0^2) = 0$$

## Entropy

$$Entropy(t) = - \sum_{c \in C} P(c|t) \log P(c|t)$$

underline

$$-(0 \log_2 0 + 1 \log_2 1) = 0$$

underline

|        |       |
|--------|-------|
| 4x YES | 2x NO |
|--------|-------|

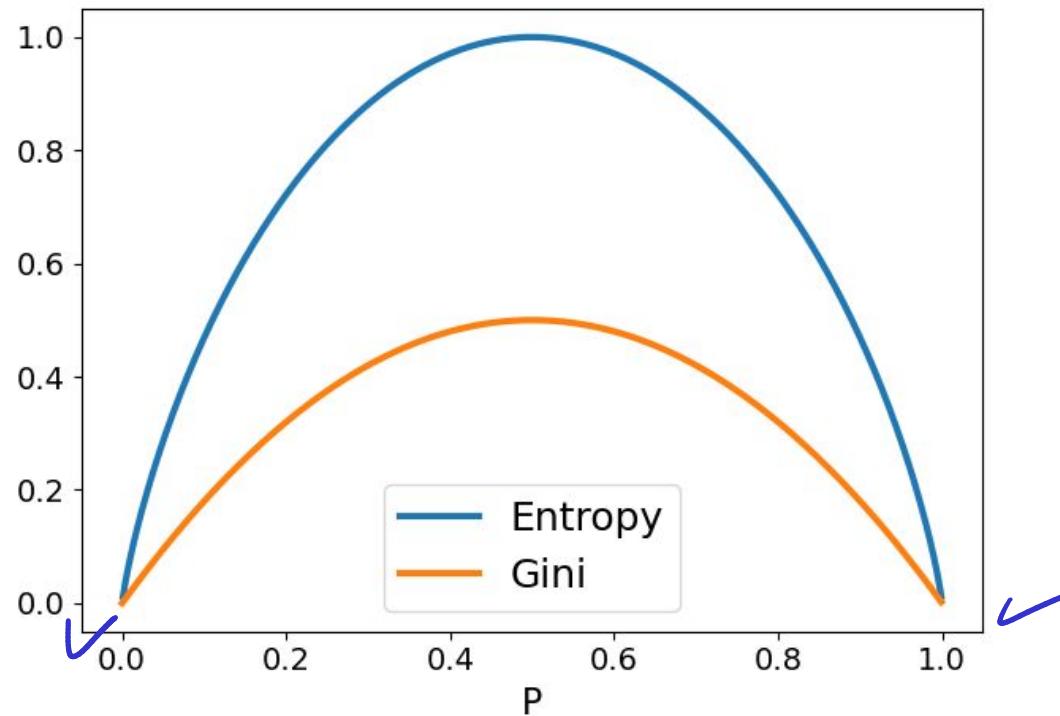
$$1 - ((4/6)^2 + (2/6)^2) = 0.44$$

$$-(4/6 \log_2 4/6 + 2/6 \log_2 2/6) = 0.92$$

$P(c|t)$  = relative frequency of class  $c$  in node  $t$

# Impurity of a Node (Classification)

- Gini Index vs. Entropy for 2-class problem



# Impurity of a Split (Classification)

- Assume node  $t$  is split into  $k$  children

- $n_i$  — number of records at  $i$ -th child
- $n$  — number of records at node  $t$
- $I(i)$  — impurity of node (e.g., Gini, Entropy)

Sum of impurity values of all children, weighted by the number of records at each child.

$$I_{split} = \sum_i^k \frac{n_i}{n} I(i)$$

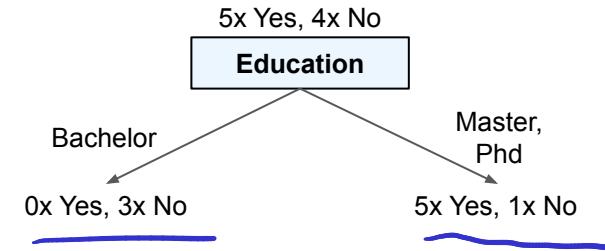
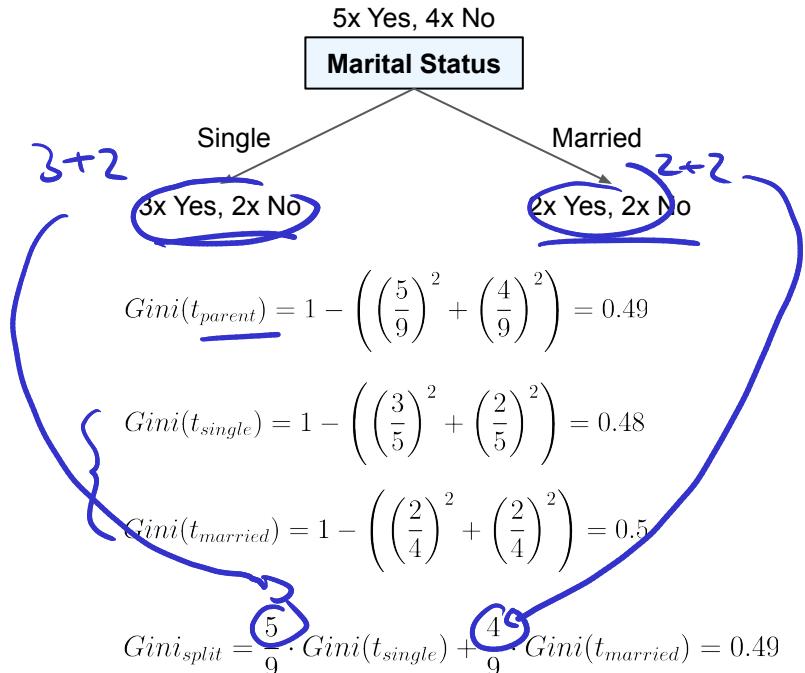
- Information Gain IG

- Difference between  $I(t)$  and  $I_{split}$
- Choose split that minimizes  $I_{split}$  =  
split that maximizes  $IG$
- Required condition:  $IG > 0$

$$IG = I(t) - I_{split}$$

---

# Impurity of Split (Classification) — Example



$$Gini(t_{parent}) = 1 - \left( \left(\frac{5}{9}\right)^2 + \left(\frac{4}{9}\right)^2 \right) = 0.49$$

$$Gini(t_B) = 1 - \left( \left(\frac{0}{3}\right)^2 + \left(\frac{3}{3}\right)^2 \right) = 0$$

$$Gini(t_{M/P}) = 1 - \left( \left(\frac{5}{6}\right)^2 + \left(\frac{1}{6}\right)^2 \right) = 0.28$$

$$Gini_{split} = \frac{3}{9} \cdot Gini(t_B) + \frac{6}{9} \cdot Gini(t_{M/P}) = 0.19$$

$$IG = Gini(t_{parent}) - Gini_{split} = 0.3$$

$$IG = Gini(t_{parent}) - Gini_{split} = 0.3$$

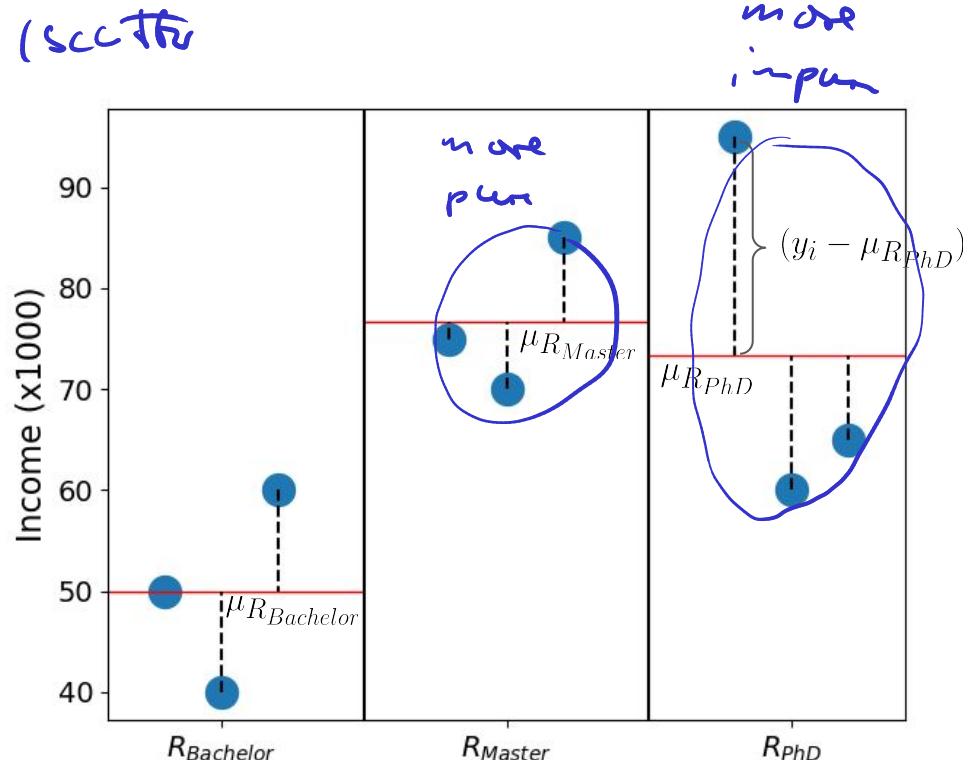
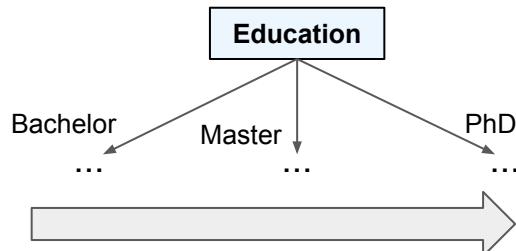
# Impurity of a Split (Regression)

## Residual Sum of Squares (RSS)

$$RSS_{split} = \sum_{k=1}^K \sum_{i \in R_k} (y_i - \mu_{R_k})^2$$

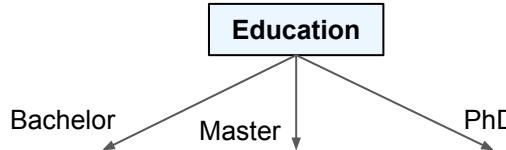
variance

| Education | Annual Income |
|-----------|---------------|
| Masters   | 75k           |
| Bachelor  | 50k           |
| Masters   | 70k           |
| PhD       | 95k           |
| Bachelor  | 40k           |
| Master    | 85k           |
| Bachelor  | 60k           |
| PhD       | 60k           |
| PhD       | 65k           |



# Impurity of a Split (Regression)

$$RSS_{split} = \sum_{k=1}^K \sum_{i \in R_k} (y_i - \mu_{R_k})^2$$



|                            |
|----------------------------|
| $\mu_{R_{Bachelor}} = 50$  |
| $\mu_{R_{Master}} = 76.67$ |
| $\mu_{R_{PhD}} = 73.33$    |

$$= \sum_{i \in R_{Bachelor}} (y_i - \mu_{R_{Bachelor}})^2 + \sum_{i \in R_{Master}} (y_i - \mu_{R_{Master}})^2 + \sum_{i \in R_{PhD}} (y_i - \mu_{R_{PhD}})^2$$

$$\begin{aligned}
 &= (50 - 50)^2 + (40 - 50)^2 + (60 - 50)^2 \\
 &\quad + (75 - 76.67)^2 + (70 - 76.67)^2 + (85 - 76.67)^2 \\
 &\quad + (95 - 73.33)^2 + (60 - 73.33)^2 + (65 - 73.33)^2 \\
 &= 1033.34
 \end{aligned}$$

| Edu-<br>cation | Annual<br>Income |
|----------------|------------------|
| Masters        | 75k              |
| Bachelor       | 50k              |
| Masters        | 70k              |
| PhD            | 95k              |
| Bachelor       | 40k              |
| Master         | 85k              |
| Bachelor       | 60k              |
| PhD            | 60k              |
| PhD            | 65k              |

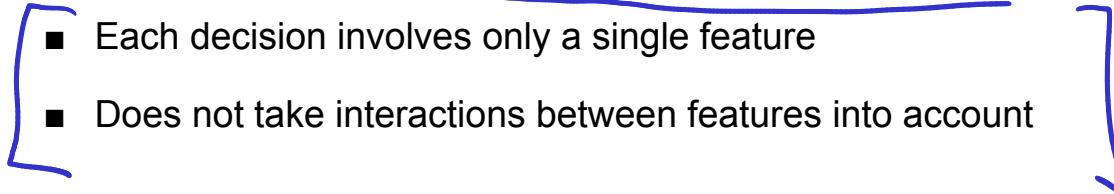
# Decision Trees — Pros & Cons

- Pros

- Inexpensive to train and test
- Easy to interpret (if tree is not too large)
- Can handle categorical and numerical data

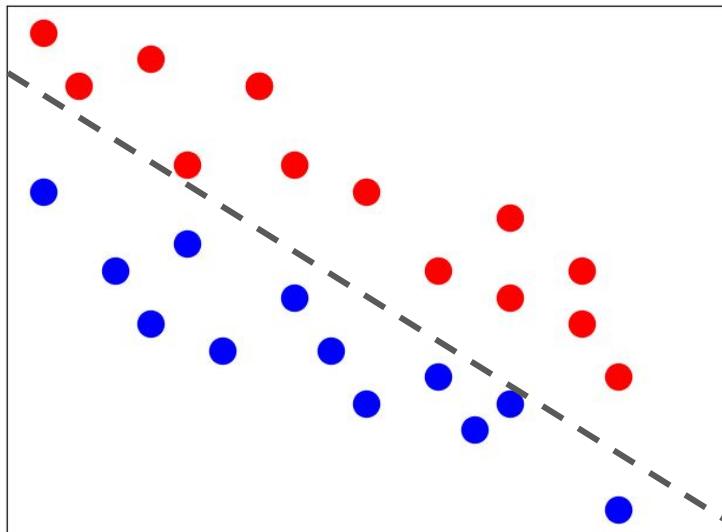
- Cons

- Sensitive to small changes in the training data  
(Hierarchical structure: errors early on propagate down)
- Greedy approach does not guarantee optimal tree
  - Each decision involves only a single feature
  - Does not take interactions between features into account

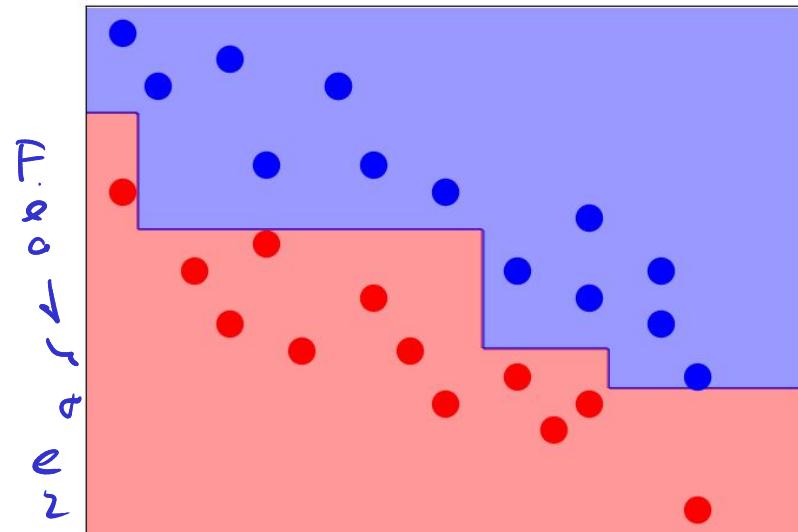


# Decision Trees — Interaction between Features

Optimal decision boundary



Decision Tree boundaries



Feature 1

# Outline

- **Decision Trees**

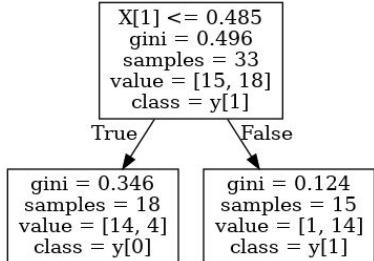
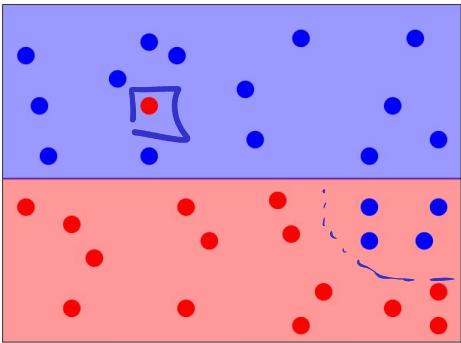
- Overview
- Training Decision Trees
- **Overfitting**

- **Tree Ensembles**

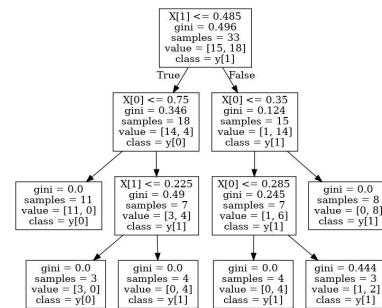
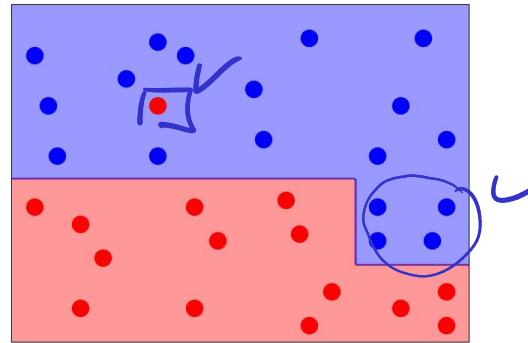
- Bagging
- Random Forest
- Boosting

# Decision Trees — Underfitting & Overfitting

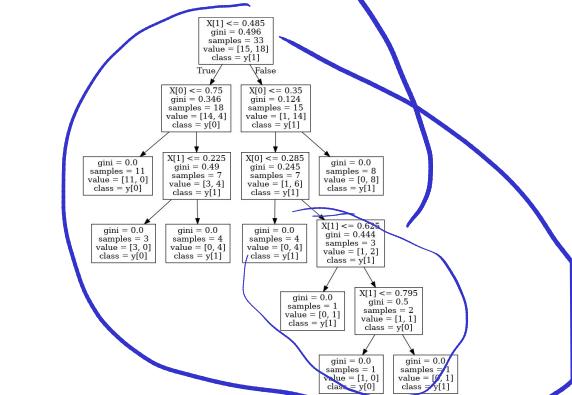
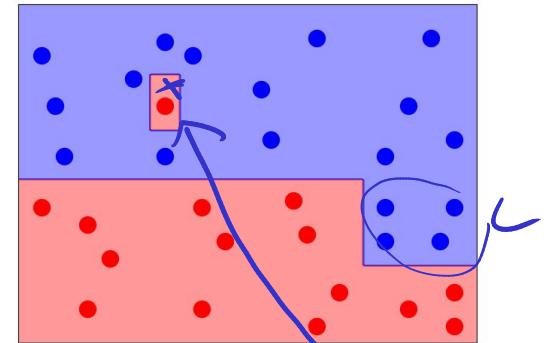
## Underfitting



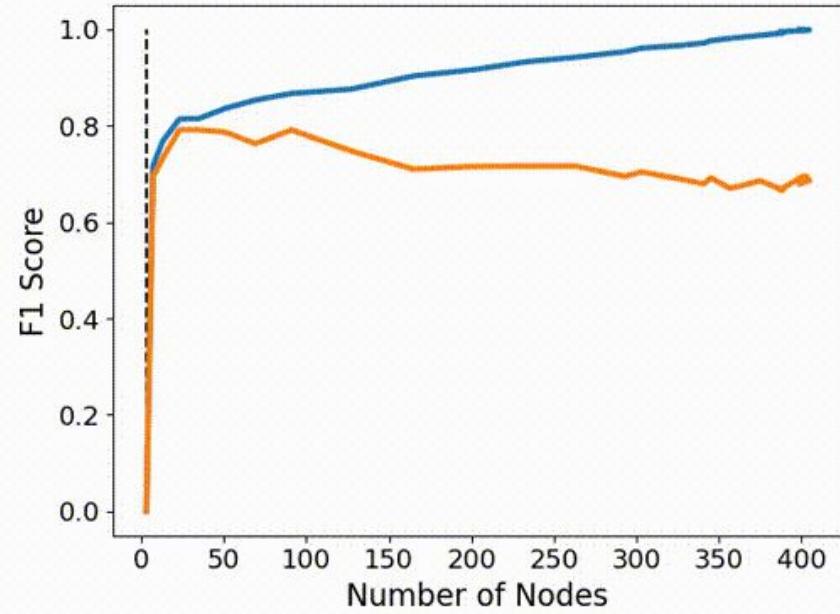
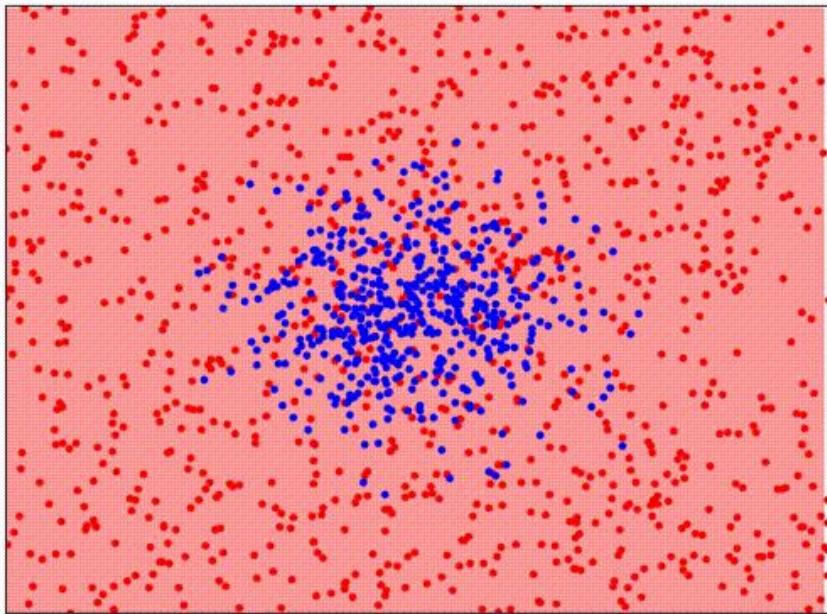
## Good fit



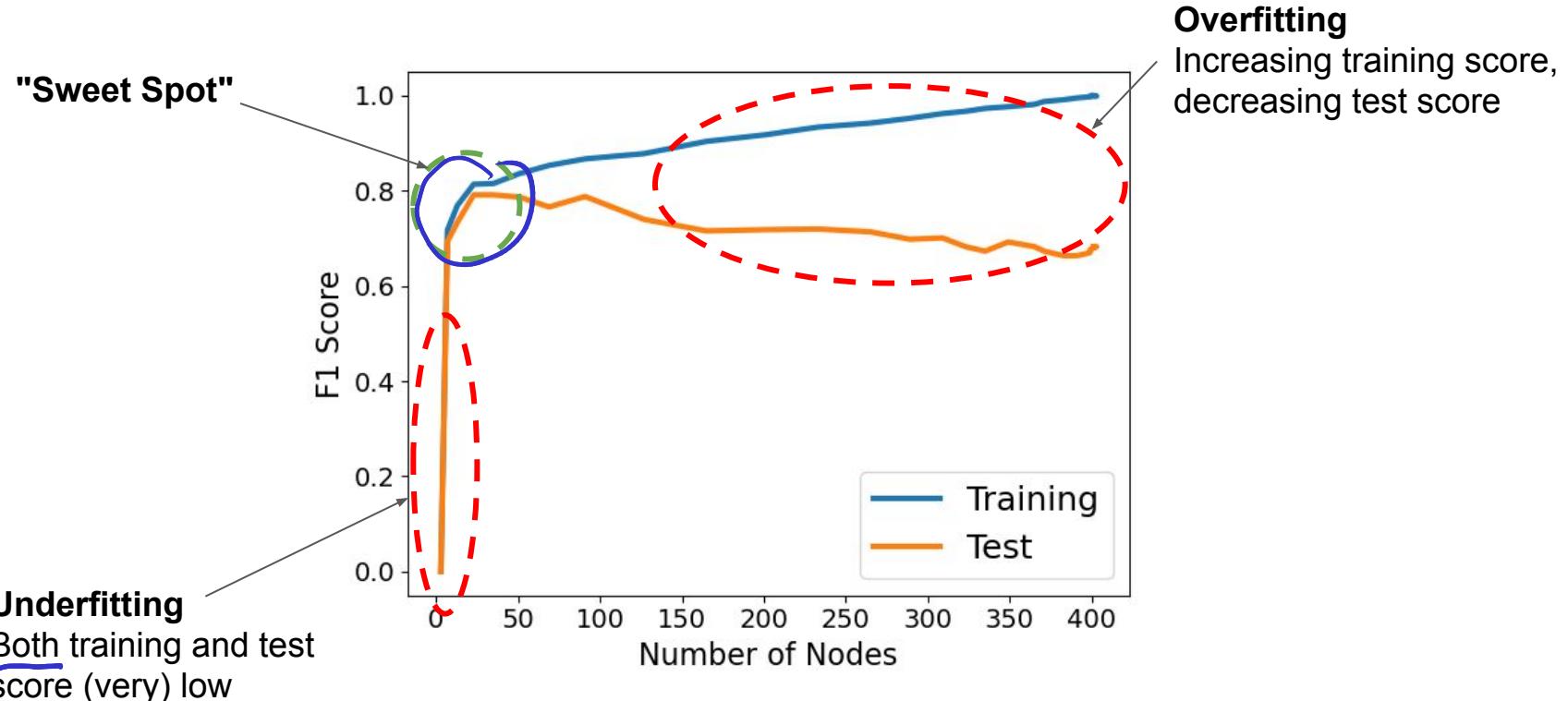
## Overfitting



# Decision Trees — Underfitting & Overfitting

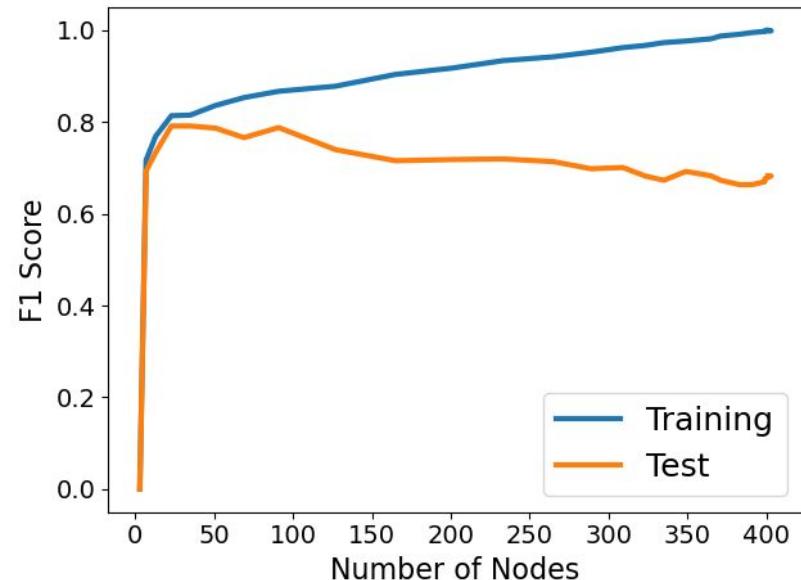


# Decision Trees — Underfitting & Overfitting



# Decision Trees — Overfitting

- Decision Tree algorithm can always split the training data perfectly\*
  - Keep splitting (i.e., increase height of tree) until each leaf contains only one data sample
  - One data sample → 100% pure

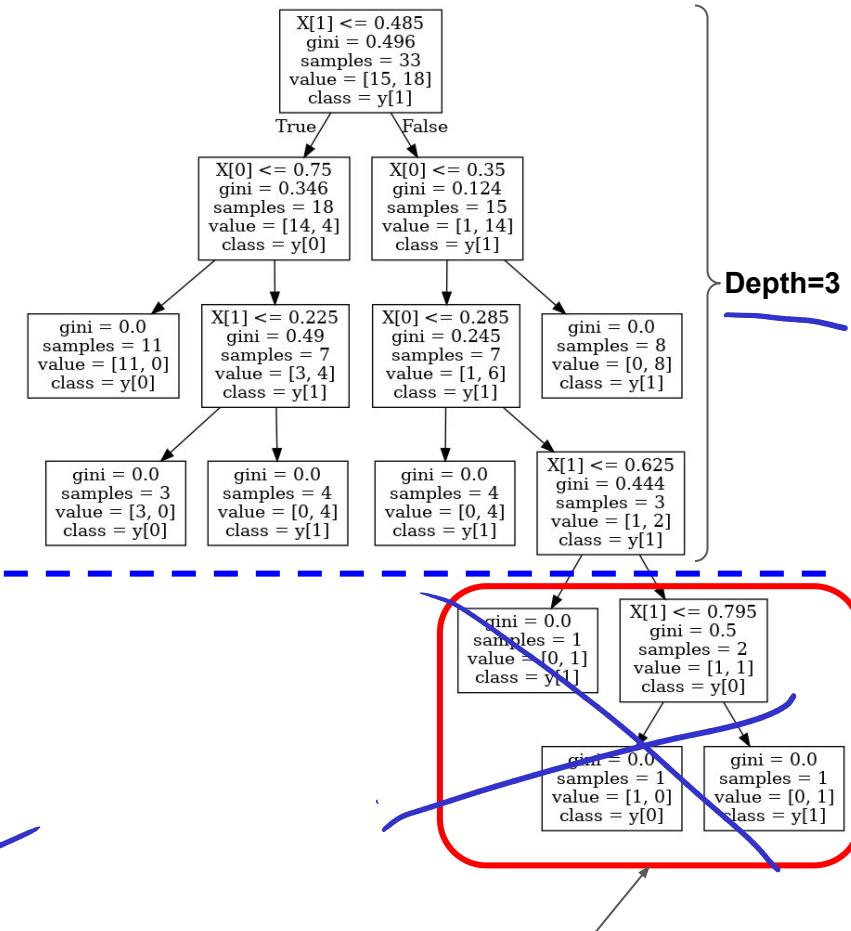
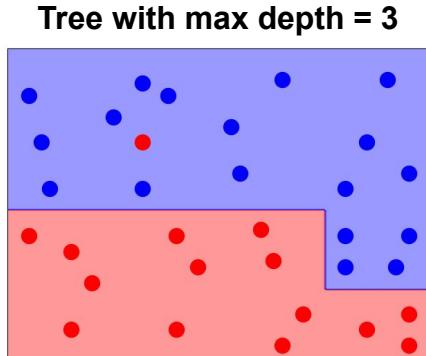
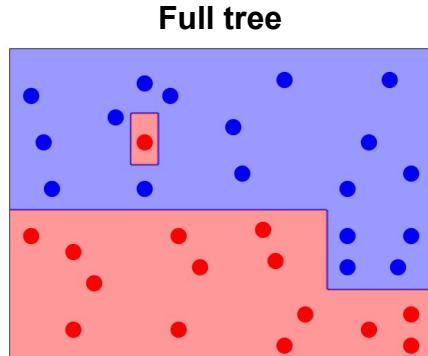


- Solution: Limit size/height of Decision Tree → Pruning
  - Pre-pruning: Stop splitting nodes ahead of time
  - Post-pruning: Build full tree, but then remove leaves/splits if beneficial
  - ... combination of multiple approaches

\*assuming not duplicate data points with different target labels/values

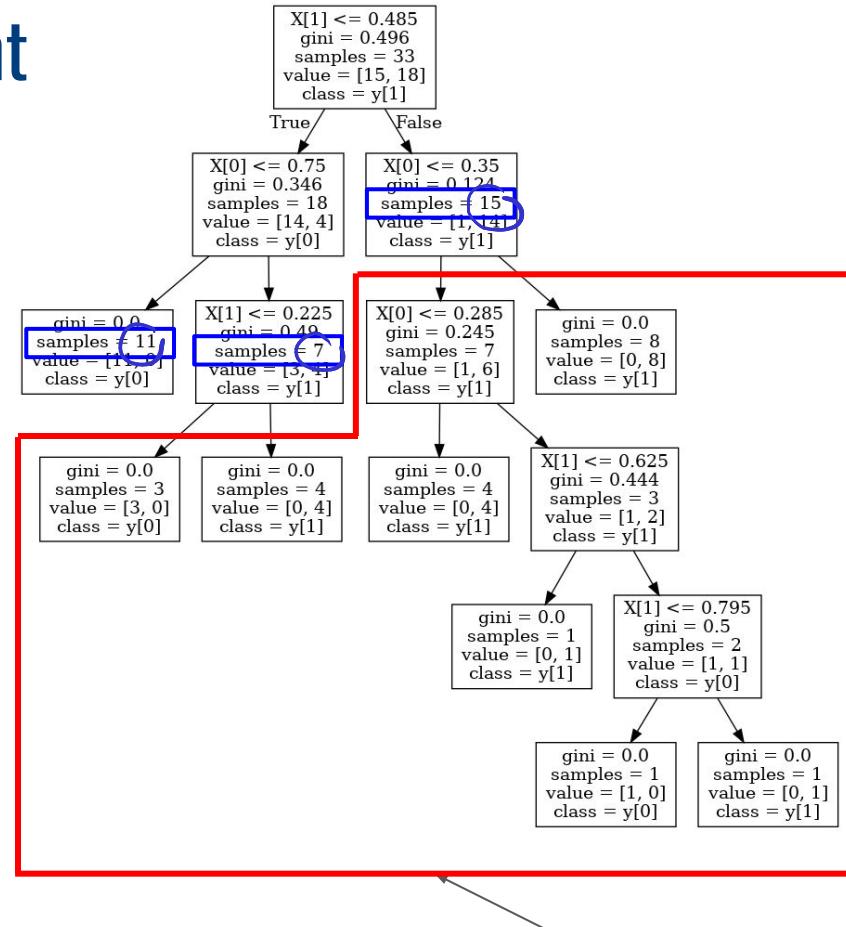
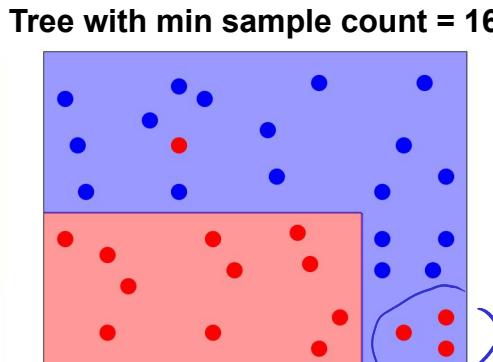
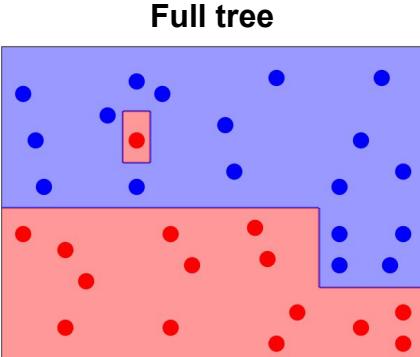
# Pre-Pruning: Maximum Depth

- Define maximum depth/height of tree
  - Stop splitting if maximum depth is reached
- Example: maximum depth = 3

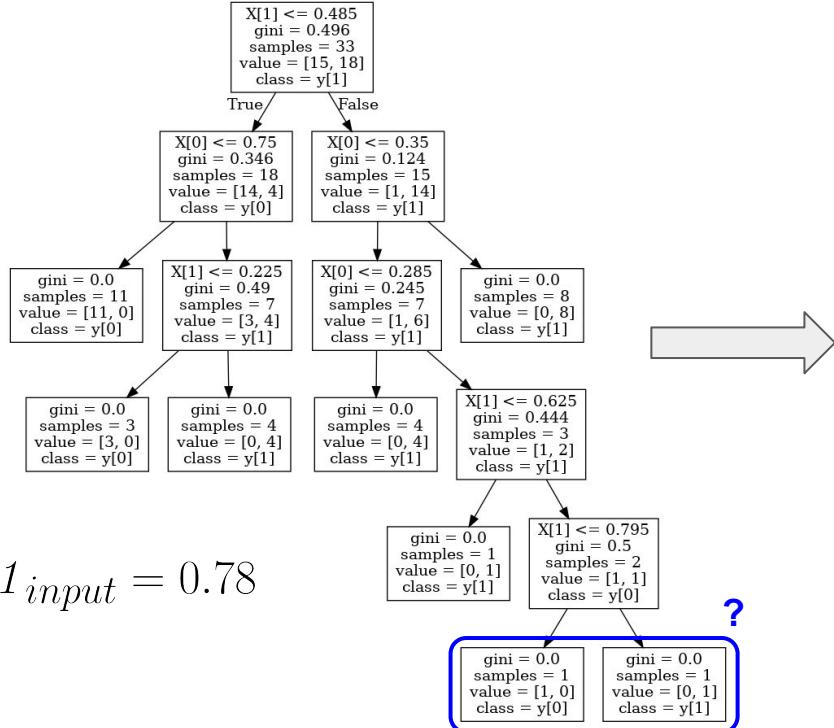
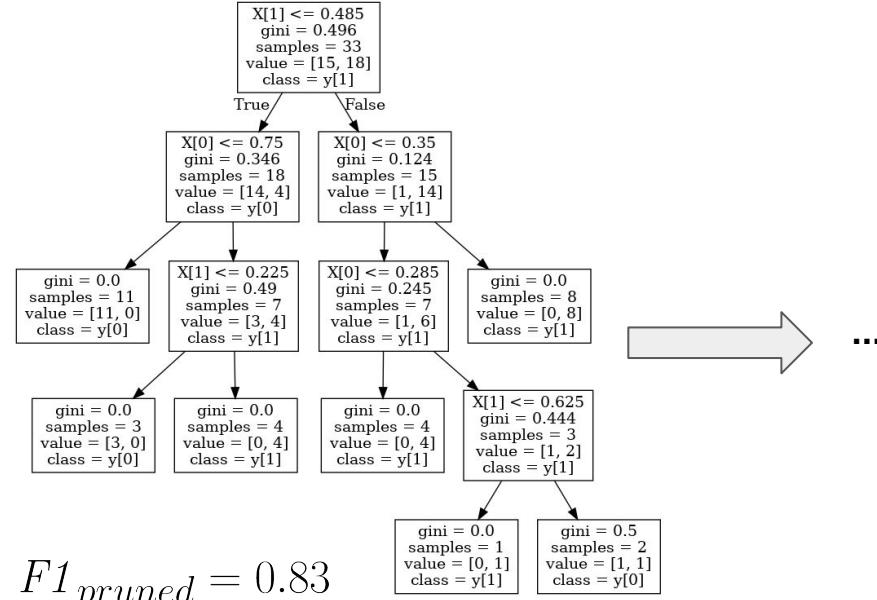


# Pre-Pruning: Minimum Sample Count

- Define minimum number of samples each node must have
  - Stop splitting if node has less than the minimum number of samples
- Example: minimum sample count = 16



# Post-Pruning: Prune Leaves/Splits using Validation


 $F1_{input} = 0.78$ 

 $F1_{pruned} = 0.83$ 

$F1_{pruned} > F1_{input} \rightarrow$  Remove split from Decision Tree (and continue checking next split)

# Quick Quiz

What is the **maximum** possible **depth** of a Decision Tree given a dataset with  $N$  data points?

**A**

$O(N)$

**B**

$O(\log N)$

**C**

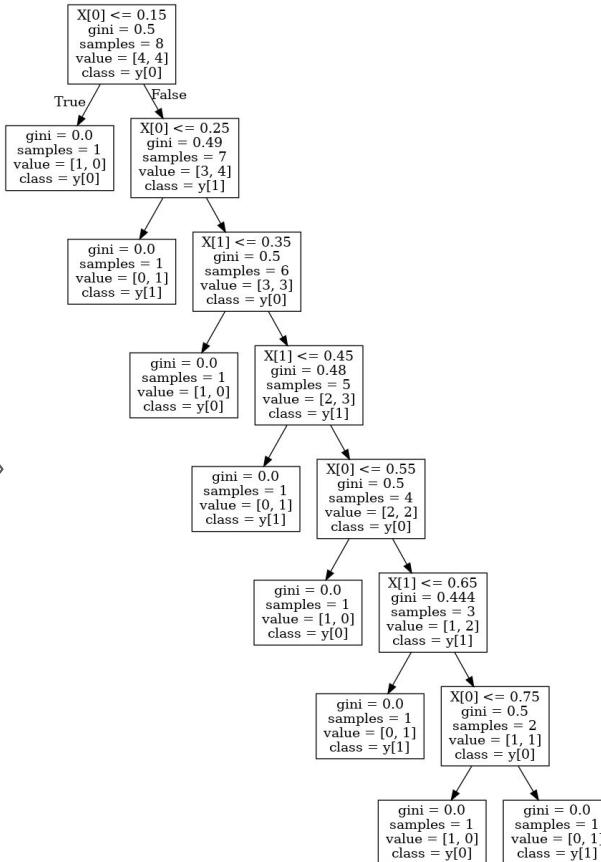
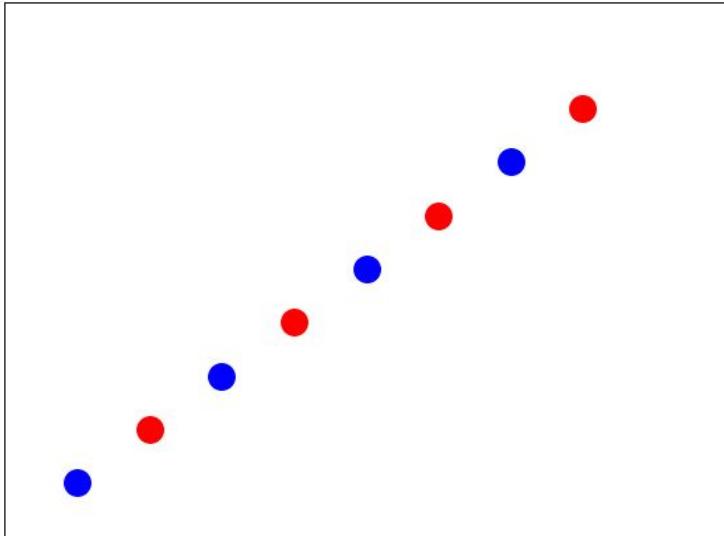
$O(N \log N)$

**D**

$O(\log \log N)$

# Quick Quiz

$N = 8$



# Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - Random Forest
  - Boosting

# Tree Ensembles

- Aim to address limitations of (single) Decision Trees
  - High variance — i.e., sensitivity to small changes in training data
  - Typically not the same accuracy as other approaches

- Countermeasure: Tree Ensembles

Construct many decision trees and combine their predictions

- Bagging
- Random Forests
- Boosting

**Basic trade-off of ensemble methods:**

Higher accuracy, lower variance  
vs.  
Lower interpretability, longer training time

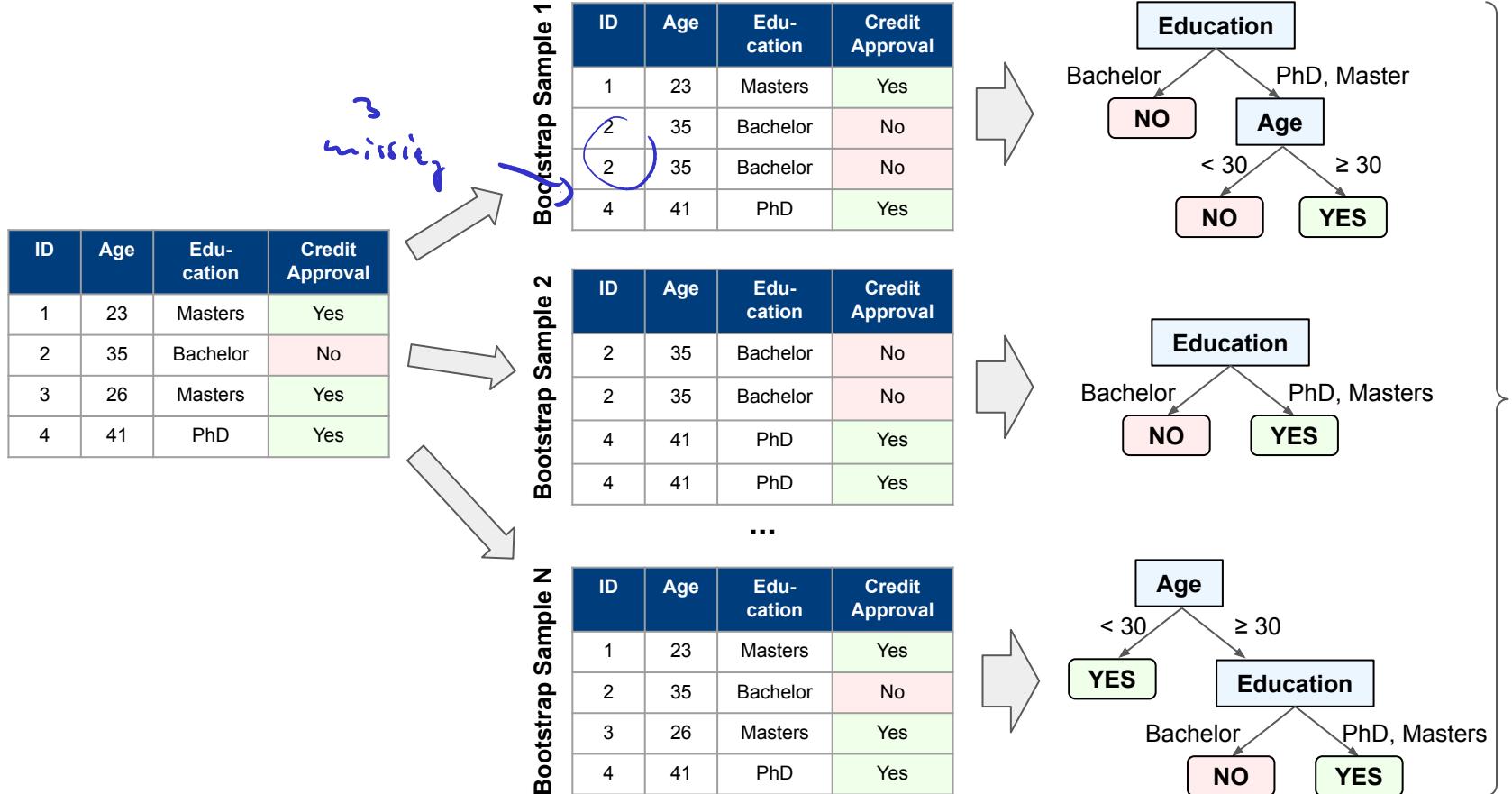
# Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - Random Forest
  - Boosting

# Bagging — Bootstrap Aggregation

- Bagging — basic idea (not limited to Decision Trees)
  - Train many models (classifiers/regressors) of different training data
  - Combine predictions of each models for final prediction
  - Increases accuracy and lowers variance
- Where to get more training data from? → Bootstrap Sampling
  - Take repeated samples from a single training dataset  $D$
  - Bootstrap sample  $D_i$  sampled from  $D$ , **uniformly** and **with replacement** ( $|D_i| = |D|$ )
  - Train a model over each bootstrap dataset  $D_i$

# Bagging — Bootstrap Aggregation



# Bagging — Bootstrap Aggregation

- Limitations

- Assume original dataset  $D$  has one or more strong predictors — features that yield splits with a (very) high information gain
- Bootstrap samples  $D_i$  are also likely to have those strong predictors

## → Consequences

- Most bagged trees will use strong predictors on top
- Most bagged trees will look very similar
- Predictions of bagged trees will be highly correlated



→ Only limited reduction in variance!

# Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - **Random Forest**
  - Boosting

# Random Forests

- Random Forest = bootstrap sampling (bagging) + feature sampling
  - Create bootstrap samples  $D_i$  like for bagging
  - Feature sampling: For each  $D_i$ , consider only a random subset of features of size  $m$

$d$  features

| Age | Edu-<br>cation | Marital<br>Status | Annual<br>Income | Credit<br>Approval |
|-----|----------------|-------------------|------------------|--------------------|
| 23  | Masters        | Single            | 75k              | Yes                |
| 35  | Bachelor       | Married           | 50k              | No                 |
| 26  | Masters        | Single            | 70k              | Yes                |
| 41  | PhD            | Single            | 95k              | Yes                |
| 18  | Bachelor       | Single            | 40k              | No                 |
| 55  | Master         | Married           | 85k              | No                 |
| 30  | Bachelor       | Single            | 60k              | No                 |
| 35  | PhD            | Married           | 60k              | Yes                |
| 28  | PhD            | Married           | 65k              | Yes                |

bootstrap sampling  
+ feature sampling



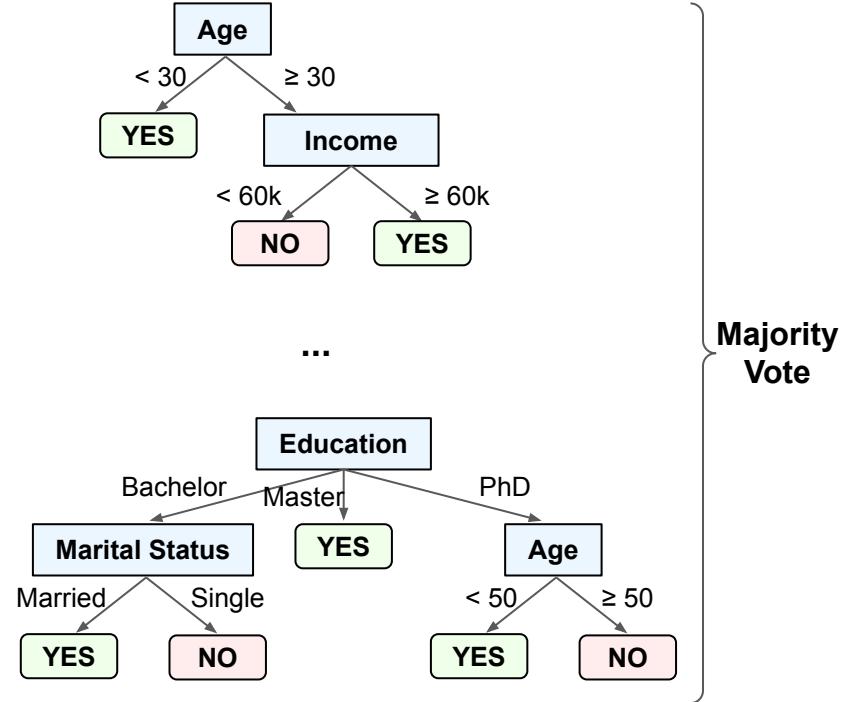
typically  
 $m \approx \sqrt{d}$

$m$  features

| Edu-<br>cation | Annual<br>Income | Credit<br>Approval |
|----------------|------------------|--------------------|
| Masters        | 75k              | Yes                |
| Bachelor       | 50k              | No                 |
| Masters        | 70k              | Yes                |
| PhD            | 95k              | Yes                |
| Bachelor       | 40k              | No                 |
| Masters        | 70k              | Yes                |
| Masters        | 75k              | Yes                |
| Bachelor       | 40k              | No                 |
| Bachelor       | 40k              | No                 |

# Random Forests

- Effects of feature sampling
  - Strong predictors in  $D$  are often absent in  $D_i$
  - Resulting trees often look very different
  - Predictions of trees much less correlated
- Higher reduction in variance + typically higher accuracy



# Random Forests — Pros & Cons (Compared to Decision Trees)

- Pros

- High accuracy — fairly close to state of the art
- Sampling and training independent across  $D_i \rightarrow$  parallelizable!
- Not much tuning required

- Cons

- Less Interpretable
- Slower training and prediction

# Outline

- **Decision Trees**
  - Overview
  - Training Decision Trees
  - Overfitting
- **Tree Ensembles**
  - Bagging
  - Random Forest
  - Boosting

# Boosting

- Like bagging, boosting combines multiple trees (in general, multiple models)
- So what are the key differences?

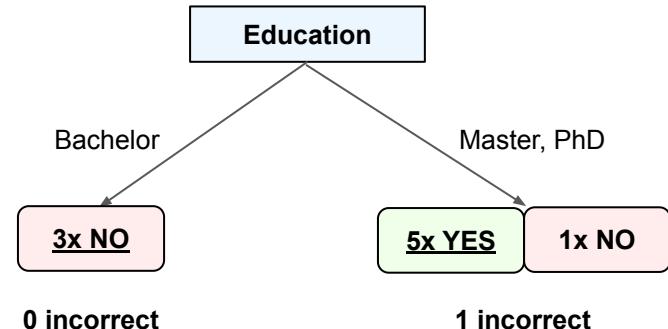
|            | Bagging                                                          | Boosting                                                                                                |
|------------|------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------|
| Training   | Trees are trained independently<br>(and can be done in parallel) | Trees are trained in sequence;<br>(the accuracy of the last tree affects the training of the next tree) |
| Prediction | All trees have the same amount of say in the final prediction    | Trees have different amount of say in the final prediction<br>(depending on their individual accuracy)  |

# Boosting & Weak Learners

- So far, all models discussed are **Strong Learners**
  - Goal: perform as best as possible on a given classification or regression task

- **Weak Learner**

- Goal: perform (slightly) better than guessing
- Very common weak learner: **Decision Stump**  
(e.g., decision tree of height 1, i.e., only one split)
- Very simple model → very fast training



- **Boosting:** Combine many weak classifiers into a single strong learner
  - Basic idea: subsequent models try to improve the errors of previous models

# AdaBoost — Adaptive Boosting (for Decision Trees)

- AdaBoost
  - Applicable to many classification/regression algorithms to improve performance
  - Very commonly combined with Decision Trees
- Basic training algorithm
  - Train a Weak Learner over  $D_i$  (e.g., Decision Stump)
  - Identify all misclassified samples
  - Calculate error rate of learner to quantify its amount of say
  - Sample  $D_{i+1}$  such that misclassified samples are more likely to be picked than correctly classified samples
  - Repeat...

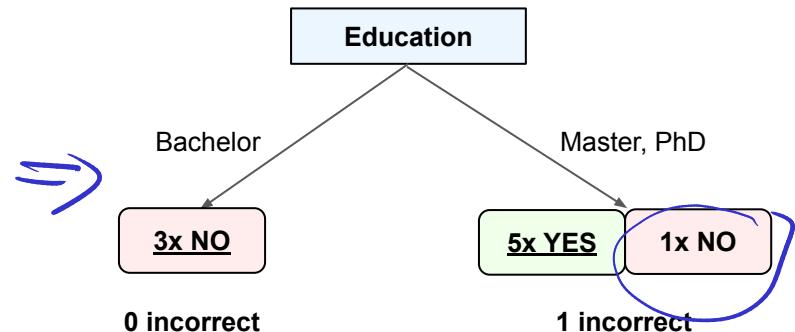
# AdaBoost Training — Step-by-Step Example (in the $m$ -th iteration)

- Step 1:

- Train Decision Stump  $h_m$  over sampled dataset  $D_m$   
(original dataset  $D$  in the beginning)
- Identify all misclassified training samples in  $D$

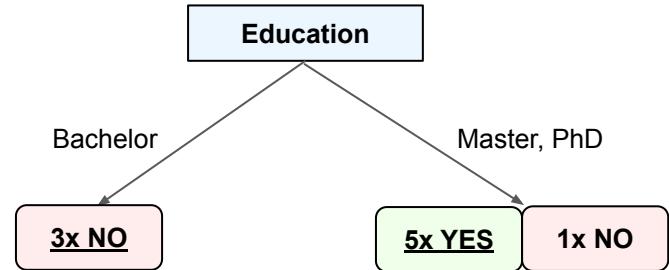
| Sample Weight $w$ |
|-------------------|
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |
| 1/9               |

| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 23  | Masters   | Single         | 75k           | Yes             |
| 35  | Bachelor  | Married        | 50k           | No              |
| 26  | Masters   | Single         | 70k           | Yes             |
| 41  | PhD       | Single         | 95k           | Yes             |
| 18  | Bachelor  | Single         | 40k           | No              |
| 55  | Master    | Married        | 85k           | No              |
| 30  | Bachelor  | Single         | 60k           | No              |
| 35  | PhD       | Married        | 60k           | Yes             |
| 28  | PhD       | Married        | 65k           | Yes             |



Initial step: assign each data sample in  $D$  with a weight  $w_i = 1/|D|$

# AdaBoost Training — Step-by-Step Example (in the $m$ -th iteration)



- **Step 2**

2a) Calculate total error  $\epsilon_m$

$$\epsilon_m = \sum_i^N w_i \cdot \underbrace{\delta(h_m(x_i) \neq y_i)}_{\begin{array}{l} 0, \text{ if } x_i \text{ is correctly classified} \\ 1, \text{ if } x_i \text{ is misclassified} \end{array}}$$

$$\epsilon_m = 1/9$$

2b) Calculate "amount of say"  $\alpha_m$  of  $h_m$

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \epsilon_m}{\epsilon_m}$$

$$\alpha_m = 1.04$$

# AdaBoost Training — Step-by-Step Example (in the $m$ -th iteration)

- Step 3

- 3a) Update sample weights

$$w_i = w_i \cdot \begin{cases} e^{\alpha_m}, & \text{if } x_i \text{ was misclassified} \\ e^{-\alpha_m}, & \text{if } x_i \text{ was correctly classified} \end{cases}$$

- 3b) Normalize sample weights

$$w_i = \frac{w_i}{\sum_i^N w_i}$$

Sum up to 1

| Age | Education | Marital Status | Annual Income | Credit Approval |
|-----|-----------|----------------|---------------|-----------------|
| 23  | Masters   | Single         | 75k           | Yes             |
| 35  | Bachelor  | Married        | 50k           | No              |
| 26  | Masters   | Single         | 70k           | Yes             |
| 41  | PhD       | Single         | 95k           | Yes             |
| 18  | Bachelor  | Single         | 40k           | No              |
| 55  | Master    | Married        | 85k           | No              |
| 30  | Bachelor  | Single         | 60k           | No              |
| 35  | PhD       | Married        | 60k           | Yes             |
| 28  | PhD       | Married        | 65k           | Yes             |



| Sample Weight w | 3a)  | 3b)    |
|-----------------|------|--------|
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.31 | 0.492  |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |
| 1/9             | 0.04 | 0.0635 |



| Sample Weight w |
|-----------------|
| 0.0635          |
| 0.0635          |
| 0.0635          |
| 0.0635          |
| 0.0635          |
| 0.0635          |
| 0.492           |
| 0.0635          |
| 0.0635          |
| 0.0635          |
| 0.0635          |

# AdaBoost Training — Step-by-Step Example (in the $m$ -th iteration)

- Step 4

4a) Generate new  $D_i$  based on sample weights

(misclassified samples are much more likely to be picked)

4b) With new  $D_i$ , go to Step 1 and continue

| Sample Weight w | Age | Edu-<br>cation | Marital Status | Annual Income | Credit Approval |
|-----------------|-----|----------------|----------------|---------------|-----------------|
| 0.0635          | 23  | Masters        | Single         | 75k           | Yes             |
| 0.0635          | 35  | Bachelor       | Married        | 50k           | No              |
| 0.0635          | 26  | Masters        | Single         | 70k           | Yes             |
| 0.0635          | 41  | PhD            | Single         | 95k           | Yes             |
| 0.0635          | 18  | Bachelor       | Single         | 40k           | No              |
| 0.492           | 55  | Master         | Married        | 85k           | No              |
| 0.0635          | 30  | Bachelor       | Single         | 60k           | No              |
| 0.0635          | 35  | PhD            | Married        | 60k           | Yes             |
| 0.0635          | 28  | PhD            | Married        | 65k           | Yes             |



New input for Step 1

| Age | Edu-<br>cation | Marital Status | Annual Income | Credit Approval |
|-----|----------------|----------------|---------------|-----------------|
| 23  | Masters        | Single         | 75k           | Yes             |
| 55  | Master         | Married        | 85k           | No              |
| 26  | Masters        | Single         | 70k           | Yes             |
| 41  | PhD            | Single         | 95k           | Yes             |
| 55  | Master         | Married        | 85k           | No              |
| 55  | Master         | Married        | 85k           | No              |
| 26  | Masters        | Single         | 70k           | Yes             |
| 35  | PhD            | Married        | 60k           | Yes             |
| 55  | Master         | Married        | 85k           | No              |

# AdaBoost Training — Basic Algorithm

**Initialization:** Dataset  $D$ ,  $|D|=N$ , with initial sample weights  $w_i = \frac{1}{N}$

**for**  $m = 1$  to  $M$  **do:**

    Generate  $D_m$  by sampling from  $D$  w.r.t. sampling weights  $w$

    Train Decision Stump  $h_m$  over  $\underline{D_m}$

    Apply  $h_m$  to all samples in  $D$  and identify misclassified samples

    Calculate total error

$$\epsilon_m = \sum_i^N w_i \cdot \delta(h_m(x_i) \neq y_i)$$

    Calculate amount of say

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \epsilon_m}{\epsilon_m}$$

    Update sample weights

$$w_i = w_i \cdot \begin{cases} e^{\alpha_m}, & \text{if } x_i \text{ was misclassified} \\ e^{-\alpha_m}, & \text{if } x_i \text{ was correctly classified} \end{cases} \quad \& \quad w_i = \frac{w_i}{\sum_i^N w_i}$$

**end for**

# AdaBoost Prediction

$\alpha_1, \dots, \alpha_r$

- Assume 8 boosted Decision Stumps  $h_1, \dots, h_8$

- Each tree has an "amount of say"  $\alpha_m$
- Let  $h_1, h_3, h_8$  say "Yes"; all other trees say "No"

$$\alpha_m = \frac{1}{2} \ln \frac{1 - \epsilon_m}{\epsilon_m}$$

|       |                   |
|-------|-------------------|
| $h_1$ | $\alpha_1 = 0.34$ |
| $h_3$ | $\alpha_3 = 1.20$ |
| $h_8$ | $\alpha_8 = 0.97$ |
| $h_2$ | $\alpha_2 = 0.14$ |
| $h_4$ | $\alpha_4 = 0.58$ |
| $h_5$ | $\alpha_5 = 0.09$ |
| $h_6$ | $\alpha_6 = 0.62$ |
| $h_7$ | $\alpha_7 = 0.45$ |

}

$$0.34 + 1.20 + 0.97 = \underline{\underline{2.51}}$$



Final prediction: "Yes"

$$0.14 + 0.58 + 0.09 + 0.62 + 0.45 = \underline{\underline{1.88}}$$

# Gradient Boosted Trees

- Gradient Boosting
  - Mainly applied to regression algorithms to improve performance
  - Very commonly combined with Decision Trees (for regression)
- Basic training algorithm
  - Start with a initial prediction (e.g., mean over all values)
  - Calculate residuals = error between true value and current prediction
  - Train Decision Stump to predict residuals
  - Update predictions based on predicted residuals
  - Repeat...

# GB Training — Step-by-Step Example (in the $m$ -th iteration)

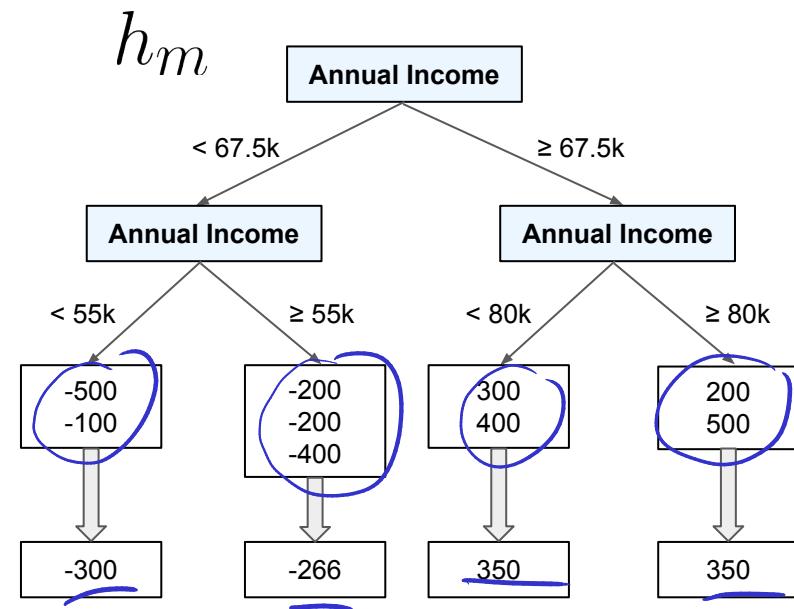
- Step 1:

1a) Calculate residuals  $r_{i,m} = y_i - f_{m-1}(x_i)$

1b) Fit Decision Stump  $h_m$  to residuals  $r_{i,m}$

| Age | Education | Marital Status | Annual Income | Credit Limit y | target       | initial prediction | $r_{i,m}(x)$ |
|-----|-----------|----------------|---------------|----------------|--------------|--------------------|--------------|
|     |           |                |               |                | $f_{m-1}(x)$ | $=$                |              |
| 23  | Master    | Single         | 75k           | 1,400          | 1,000        | =                  | 400          |
| 35  | Bachelor  | Married        | 50k           | 900            | 1,000        | =                  | -100         |
| 26  | Master    | Single         | 70k           | 1,300          | 1,000        | =                  | 300          |
| 41  | PhD       | Single         | 95k           | 1,500          | 1,000        | =                  | 500          |
| 18  | Bachelor  | Single         | 40k           | 500            | 1,000        | =                  | -500         |
| 55  | Master    | Married        | 85k           | 1,200          | 1,000        | =                  | 200          |
| 30  | Bachelor  | Single         | 60k           | 800            | 1,000        | =                  | -200         |
| 35  | PhD       | Married        | 60k           | 800            | 1,000        | =                  | -200         |
| 28  | PhD       | Married        | 65k           | 600            | 1,000        | =                  | -400         |

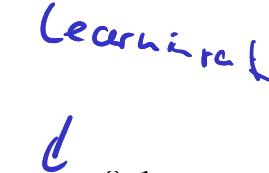
Assume  $m = 1 \quad f_0(x_i) = 1000$



# GB Training — Step-by-Step Example (in the $m$ -th iteration)

- Step 2:

- Calculate predicted residuals  $h_m(x_i)$  for all training samples
- Calculate new predictions  $f_m(x_i) = f_{m-1} + \eta \cdot h_m(x_i)$  (here:  $\eta = 0.1$ )
- Set  $m = m+1$ , go to Step 1



| Age | Edu-<br>cation | Marital<br>Status | Annual<br>Income | Credit<br>Limit $y$ | $f_{m-1}(x)$ | $r_m(x)$ | $h_m(x)$ | $f_m(x)$ |
|-----|----------------|-------------------|------------------|---------------------|--------------|----------|----------|----------|
| 23  | Master         | Single            | 75k              | 1,400               | 1,000        | 400      | 350      | 1,035    |
| 35  | Bachelor       | Married           | 50k              | 900                 | 1,000        | -100     | -300     | 970      |
| 26  | Master         | Single            | 70k              | 1,300               | 1,000        | 300      | 350      | 1,035    |
| 41  | PhD            | Single            | 95k              | 1,500               | 1,000        | 500      | 350      | 1,035    |
| 18  | Bachelor       | Single            | 40k              | 500                 | 1,000        | -500     | -300     | 970      |
| 55  | Master         | Married           | 85k              | 1,200               | 1,000        | 200      | 350      | 1,035    |
| 30  | Bachelor       | Single            | 60k              | 800                 | 1,000        | -200     | -266     | 973      |
| 35  | PhD            | Married           | 60k              | 800                 | 1,000        | -200     | -266     | 973      |
| 28  | PhD            | Married           | 65k              | 600                 | 1,000        | -400     | -266     | 973      |

Note: long-term trend

- The residuals  $r_m$  go towards 0
- The predicted values  $f_m$  are closer to the true values  $y$

# GB Training — Step-by-Step Example (in the $m$ -th iteration)

- Output for after Step 1 & 2 for  $m+1$

Includes building new Decision Stump  $h_{m+1}$

$1,400 - 1,035 = 365$

Step 1      Step 2

| Age | Education | Marital Status | Annual Income | Credit Limit y | $f_{m-1}(x)$ | $r_m(x)$ | $h_m(x)$ | $f_m(x)$ | $r_{m+1}(x)$ | $h_{m+1}(x)$ | $f_{m+1}(x)$ |
|-----|-----------|----------------|---------------|----------------|--------------|----------|----------|----------|--------------|--------------|--------------|
| 23  | Master    | Single         | 75k           | 1,400          | 1,000        | 400      | 350      | 1,035    | 365          | 315          | 1,067        |
| 35  | Bachelor  | Married        | 50k           | 900            | 1,000        | -100     | -300     | 970      | -70          | -270         | 943          |
| 26  | Master    | Single         | 70k           | 1,300          | 1,000        | 300      | 350      | 1,035    | 265          | 315          | 1,067        |
| 41  | PhD       | Single         | 95k           | 1,500          | 1,000        | 500      | 350      | 1,035    | 465          | 315          | 1,067        |
| 18  | Bachelor  | Single         | 40k           | 500            | 1,000        | -500     | -300     | 970      | -470         | -270         | 943          |
| 55  | Master    | Married        | 85k           | 1,200          | 1,000        | 200      | 350      | 1,035    | 165          | 315          | 1,067        |
| 30  | Bachelor  | Single         | 60k           | 800            | 1,000        | -200     | -266     | 973      | -173         | -240         | 949          |
| 35  | PhD       | Married        | 60k           | 800            | 1,000        | -200     | -266     | 973      | -173         | -240         | 949          |
| 28  | PhD       | Married        | 65k           | 600            | 1,000        | -400     | -266     | 973      | -373         | -240         | 949          |

# Gradient Boosting Training — Basic Algorithm

**Initialization:** Dataset  $D$ ,  $f_0(x_i) = \text{mean}(y)$ ,  $\eta = 0.1$

**for**  $m = 1$  to  $M$  **do:**

    Calculate residuals  $r_{i,m} = y_i - f_{m-1}(x_i)$

    Train Decision Stump  $h_m$  over  $D$  with  $r_{i,m}$  as targets

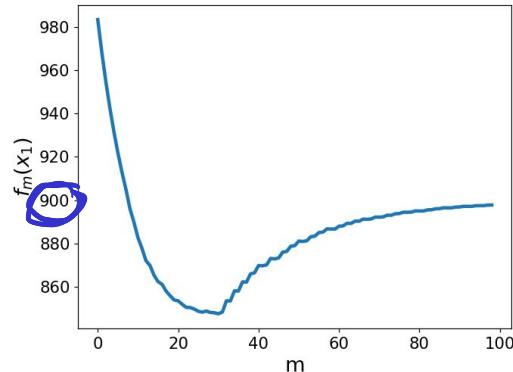
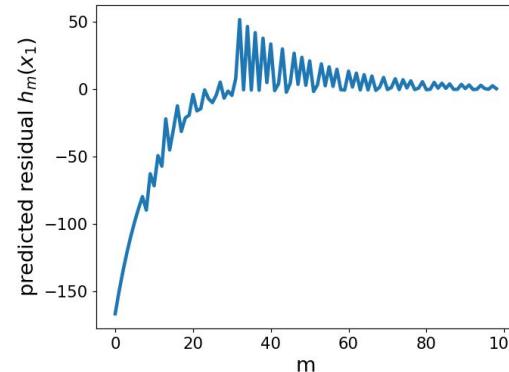
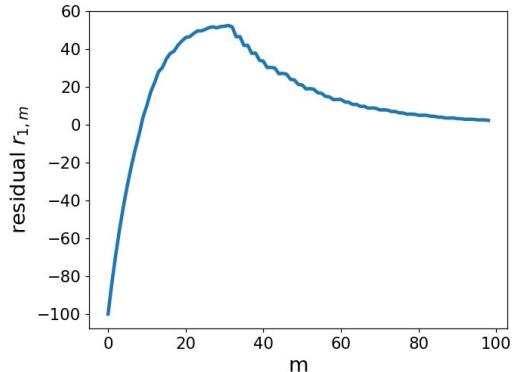
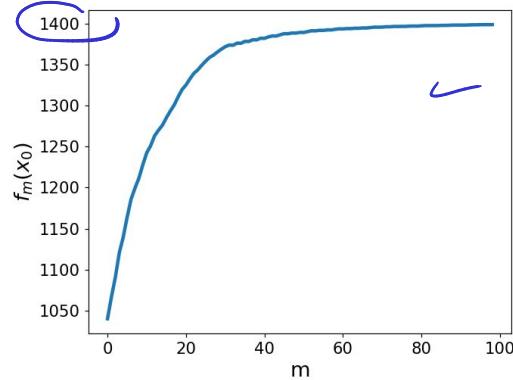
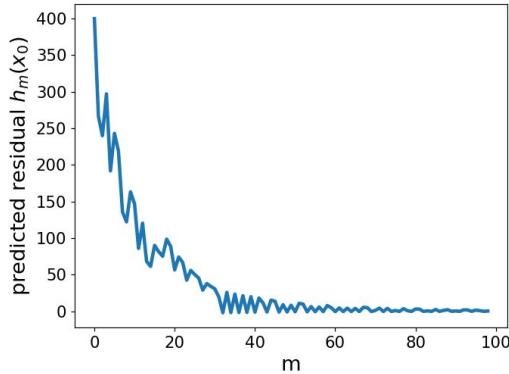
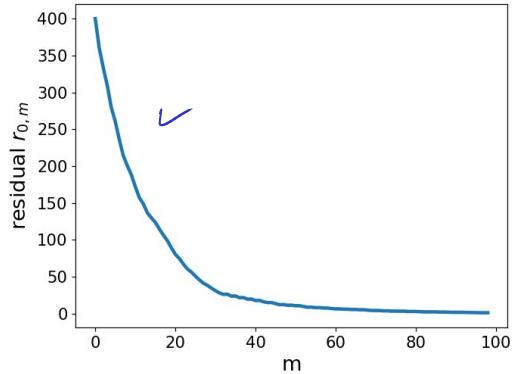
    Predicted residuals  $h_m(x_i)$  for all training samples

    Calculate new predictions  $f_m(x_i) = f_{m-1} + \eta \cdot h_m(x_i)$

**end for**

**Output:**  $M$  Decision Stumps  $h_1, h_2, \dots, h_M$

# Gradient Boosting Training — Convergence for $x_0$ and $x_1$



# Gradient Boosting Prediction

| Age | Edu-<br>cation | Marital<br>Status | Annual<br>Income | Credit<br>Limit |
|-----|----------------|-------------------|------------------|-----------------|
| 50  | PhD            | Single            | 70k              | ???             |

$$h(x) = h_0(x) + \eta h_1(x) + \eta h_2(x) + \dots + \eta h_M(x)$$

Initial prediction  $f_0(x)$   
(e.g.,  $f_0(x) = 1000$ )

# Boosting Methods — Pros & Cons (Compared to Decision Trees)

- Pros
  - High accuracy — often state of the art
- Cons
  - Less Interpretable (arguably even less compared to Random Forests)
  - Slower training and prediction → sequential processing → not parallelizable

# Summary

- **Decision Trees**
  - Intuitive model for classification and regression → interpretable!
  - Can handle categorical and numerical data (although tricky in practice)
  - Typically good but not great results
- **Tree Ensembles**
  - Aim to address limitations of single decision trees (particularly high variance)
  - Ensembles of independent models: Bagging, Random Forests
  - Ensembles of dependent models: AdaBoost, Gradient Boosted Trees
  - State of the art in many application contexts

# Solutions to Quick Quizzes

- Slide 11: Throw error (safe default), use majority label of subtree
- Slide 27: B ("natural" ranking of energy labels is broken)
- Slide 47: A