K-Nearest Neighbors

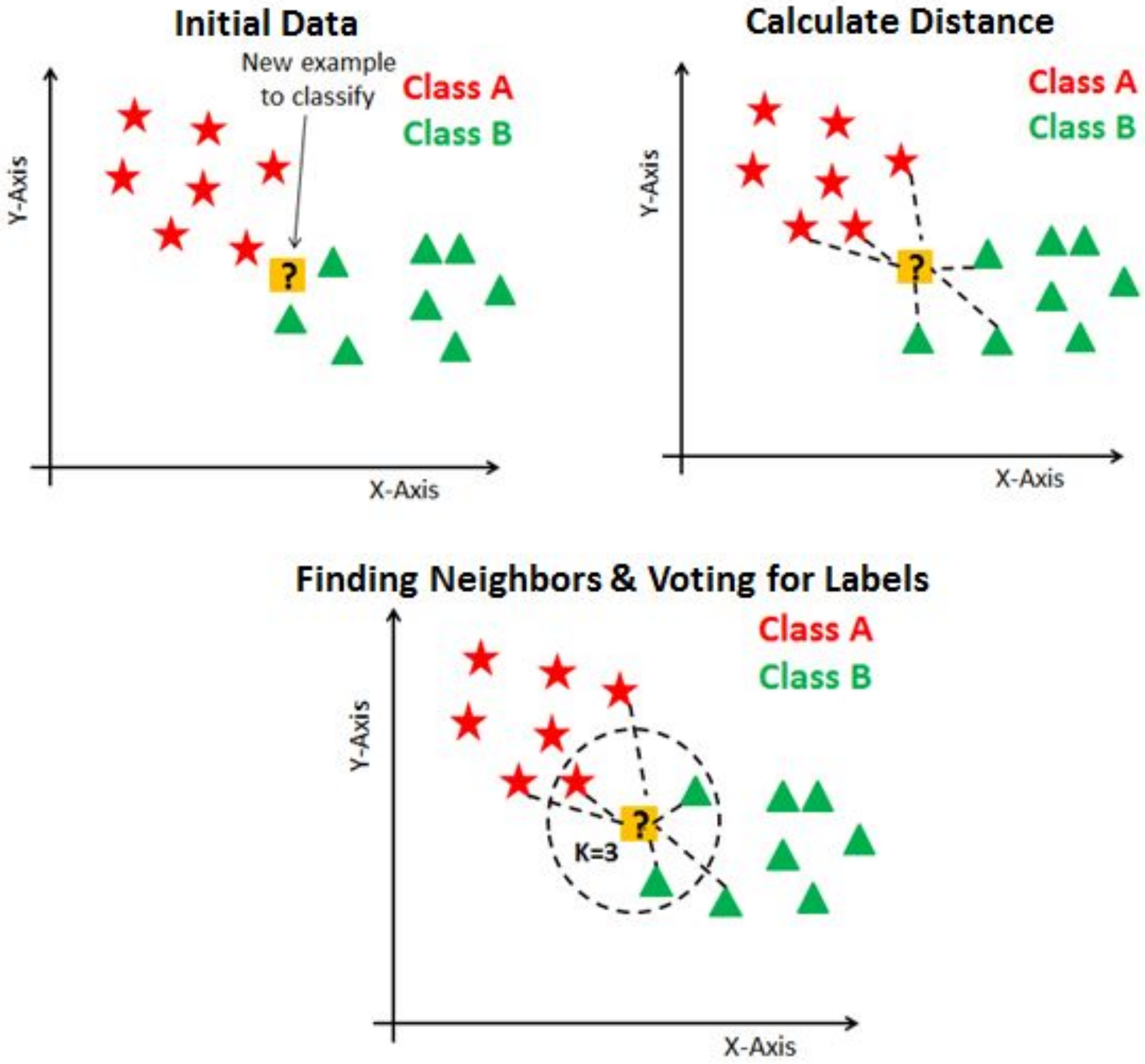
Original Slides by:
Courtney Anne Ngo
Daniel Stanley Tan, PhD
Arren Antioquia

Updated (AY 2023 – 2024 T3) by:
Thomas James Tiam-Lee, PhD

K-Nearest Neighbors (KNN)

- The most “naïve” kind of **supervised** machine learning model.

- It makes a prediction based on similarity to its closest neighbors.



Initial Data

Calculate Distance

Finding Neighbors & Voting for Labels

Sample Data 1

X		y
temperature	**humidity**	**weather**
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	23	rainy
21	25	rainy
17	14	rainy
34	29	rainy

KNN: Training

- Not much “training” to be done.
- KNN simply **memorizes the entire dataset** and uses that as the model!

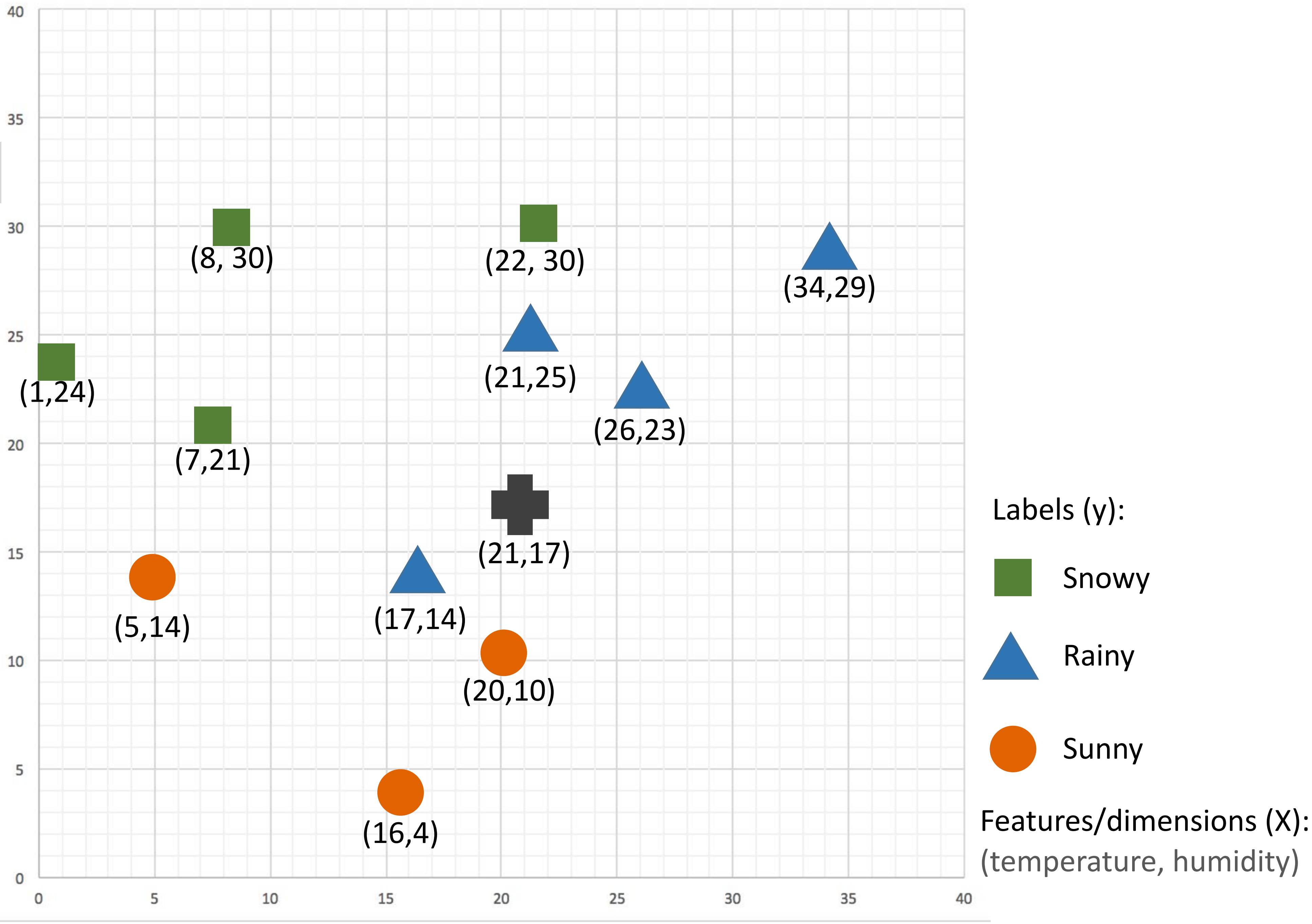
temperature	humidity	weather
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	23	rainy
21	25	rainy
17	14	rainy
34	29	rainy

Sample Data 1

x		y
temperature	**humidity**	**weather**
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	23	rainy
21	25	rainy
17	14	rainy
34	29	rainy

temp	humidity	weather
21	17	?

temp	humidity	weather
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	223	rainy
21	25	rainy
17	14	rainy
34	29	rainy



KNN: Prediction

- To make a prediction on an unknown instance, **find the most similar object and copy its class label!**

temp	humidity	weather
21	17	?

temperature	humidity	weather
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	23	rainy
21	25	rainy
17	14	rainy
34	29	rainy

Measures of Similarity

- Similarity of an instance z to the i-th training instance $X^{(i)}$
- **Euclidean Distance** (**L2-Distance**):

$$dist\left(z, X^{(i)}\right) = \sqrt{\sum_{j=1}^{d} \left(z_j - X_j^{(i)}\right)^2}$$

- **Manhattan Distance** (**L1-Distance**):

$$dist\left(z, X^{(i)}\right) = \sum_{j=1}^{d} \left|z_j - X_j^{(i)}\right|$$

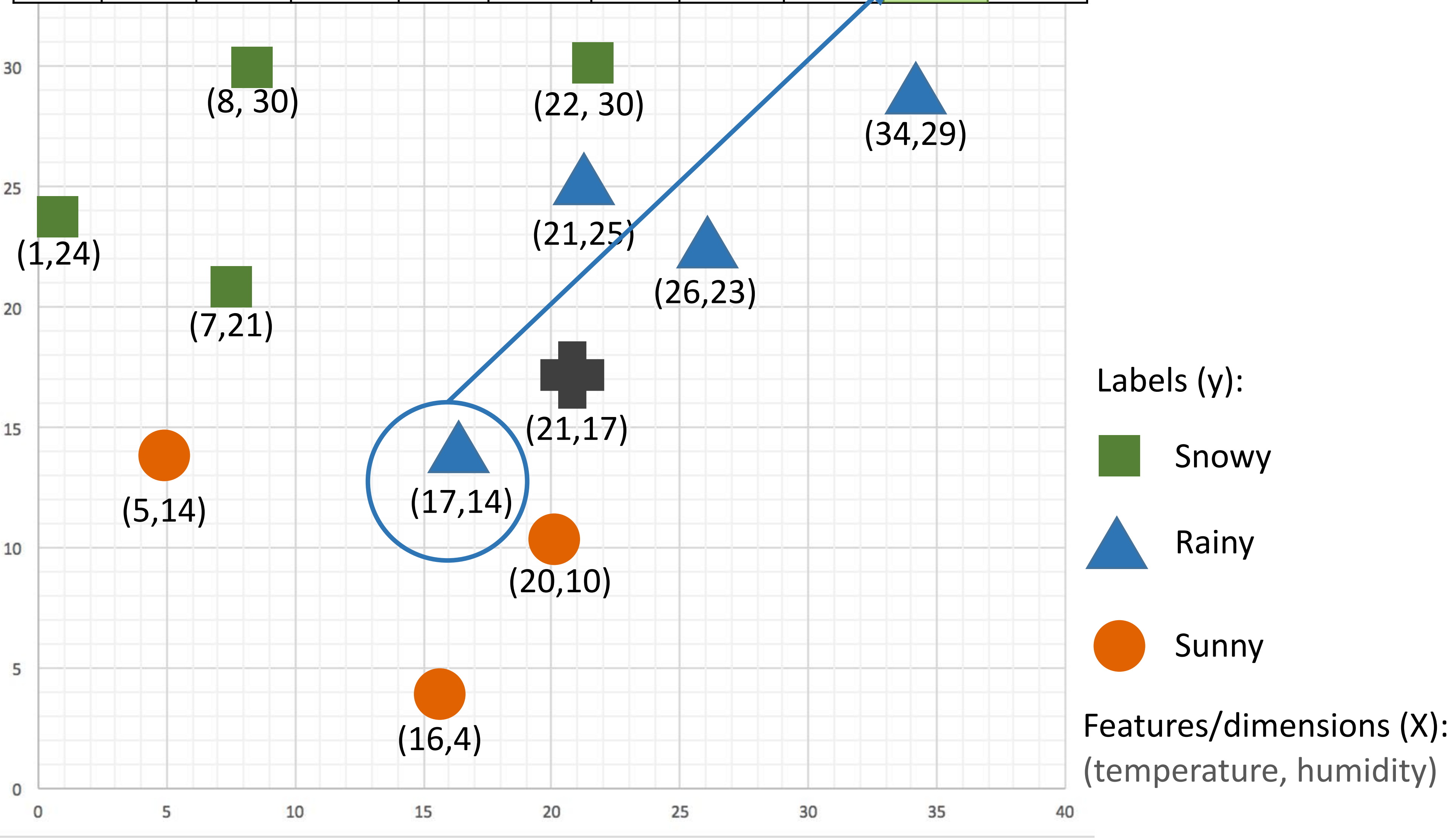
temp	humidity	weather
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	223	rainy
21	25	rainy
17	14	rainy
34	29	rainy

Distances of each training data to ✚ (21,17)

$k = 1$

(1,24)	(8, 30)	(7,21)	(22, 30)	(5,14)	(20,10)	(16,4)	(26,23)	(21,25)	(17,14)	(34,29)
21.19	18.38	14.56	13.03	16.28	7.07	13.93	7.81	8.00	5.00	17.69

▲



The Hyperparameter k

- A **hyperparameter** is an option that you **manually decide on** when training an ML model.

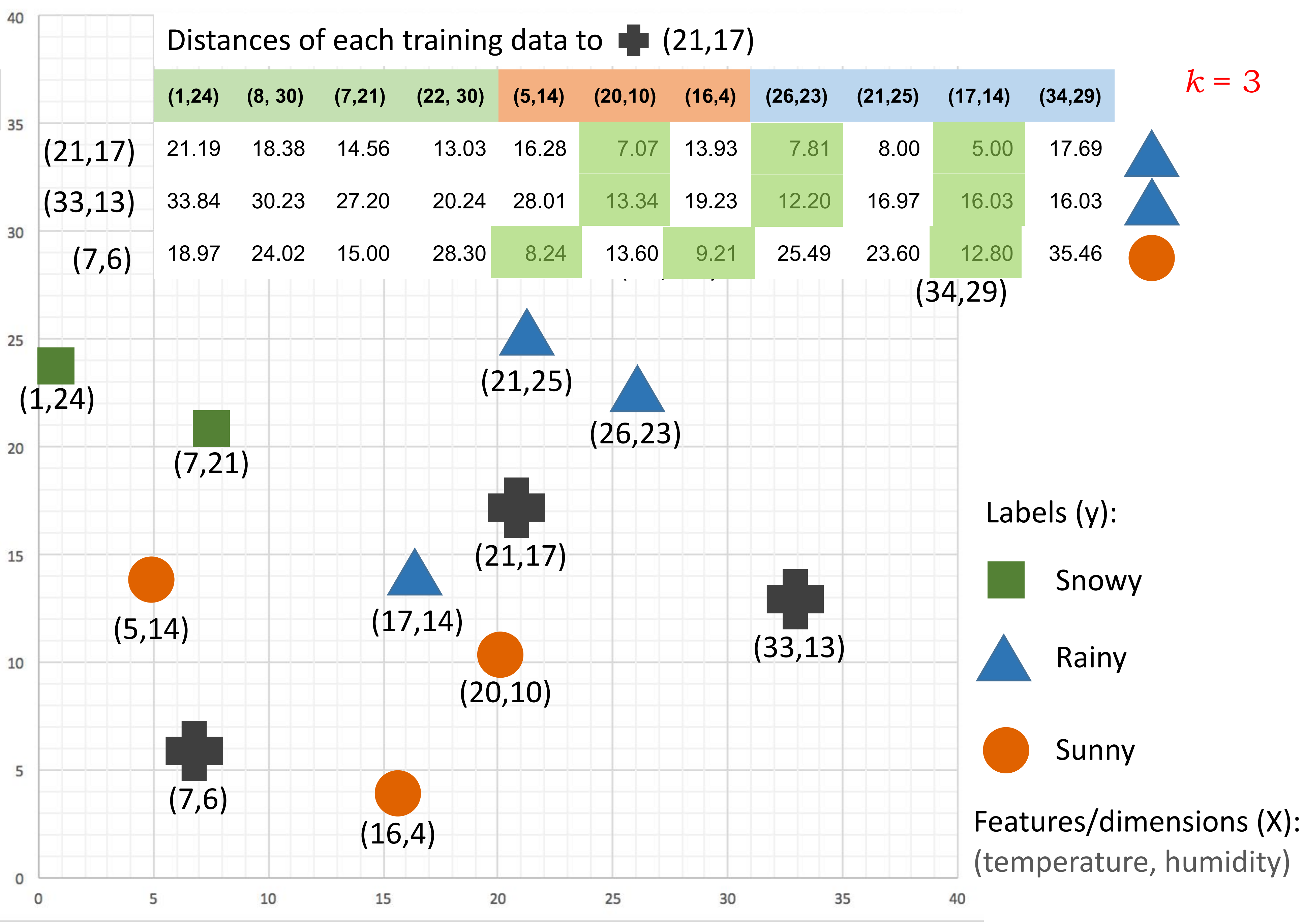
- $\boldsymbol{k}$ is the **number of nearest neighbors** to consider before making a prediction.

temp	humidity	weather
1	24	snowy
8	30	snowy
7	21	snowy
22	30	snowy
5	14	sunny
20	10	sunny
16	4	sunny
26	223	rainy
21	25	rainy
17	14	rainy
34	29	rainy

Distances of each training data to ✚ (21,17)

$k = 3$

	(1,24)	(8, 30)	(7,21)	(22, 30)	(5,14)	(20,10)	(16,4)	(26,23)	(21,25)	(17,14)	(34,29)	
(21,17)	21.19	18.38	14.56	13.03	16.28	7.07	13.93	7.81	8.00	5.00	17.69	▲
(33,13)	33.84	30.23	27.20	20.24	28.01	13.34	19.23	12.20	16.97	16.03	16.03	▲
(7,6)	18.97	24.02	15.00	28.30	8.24	13.60	9.21	25.49	23.60	12.80	35.46	●



Labels (y):

🟩 Snowy

▲ Rainy

● Sunny

Features/dimensions (X):
(temperature, humidity)

Sample Data 2

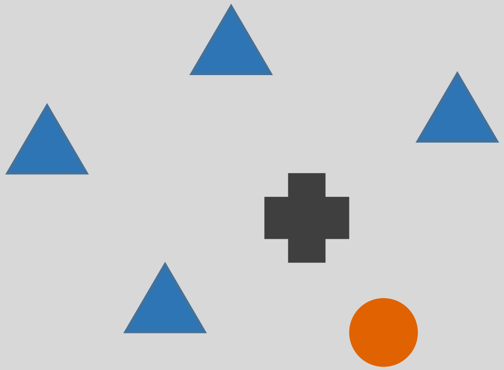
x		y
temperature	**humidity**	**bacterial growth**
1	24	4
8	30	10
7	21	6
22	30	10
5	14	2
20	10	8
16	4	4
26	23	8
21	25	10
17	14	10
34	29	2

temp	**humidity**	**bacterial growth**
21	17	?

Deciding the Final Prediction

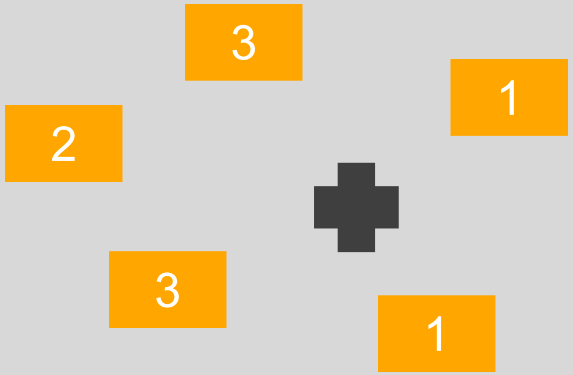
- If $k > 1$…

For **classification**, majority wins! (mode)



Final prediction: ▲

For **regression**, ?????????

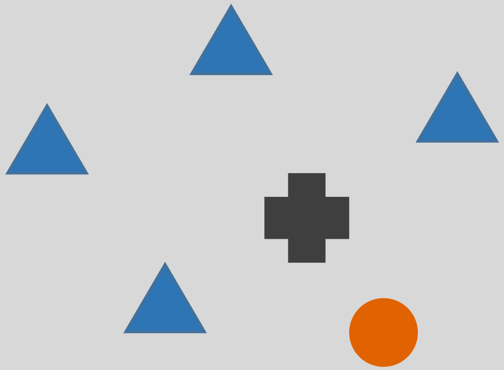


Final classification: ?

Deciding the Final Prediction

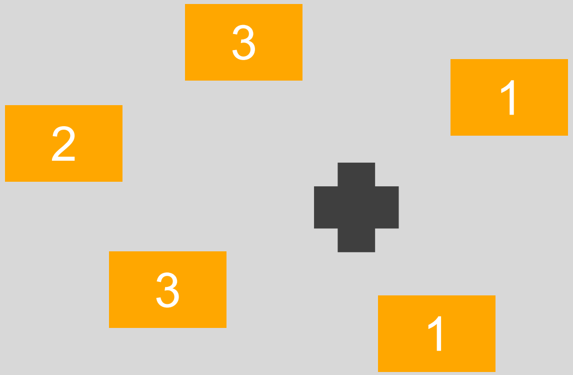
- If $k > 1$…

For **classification**, majority wins! (mode)



Final prediction: ▲

For **regression**, get the average!

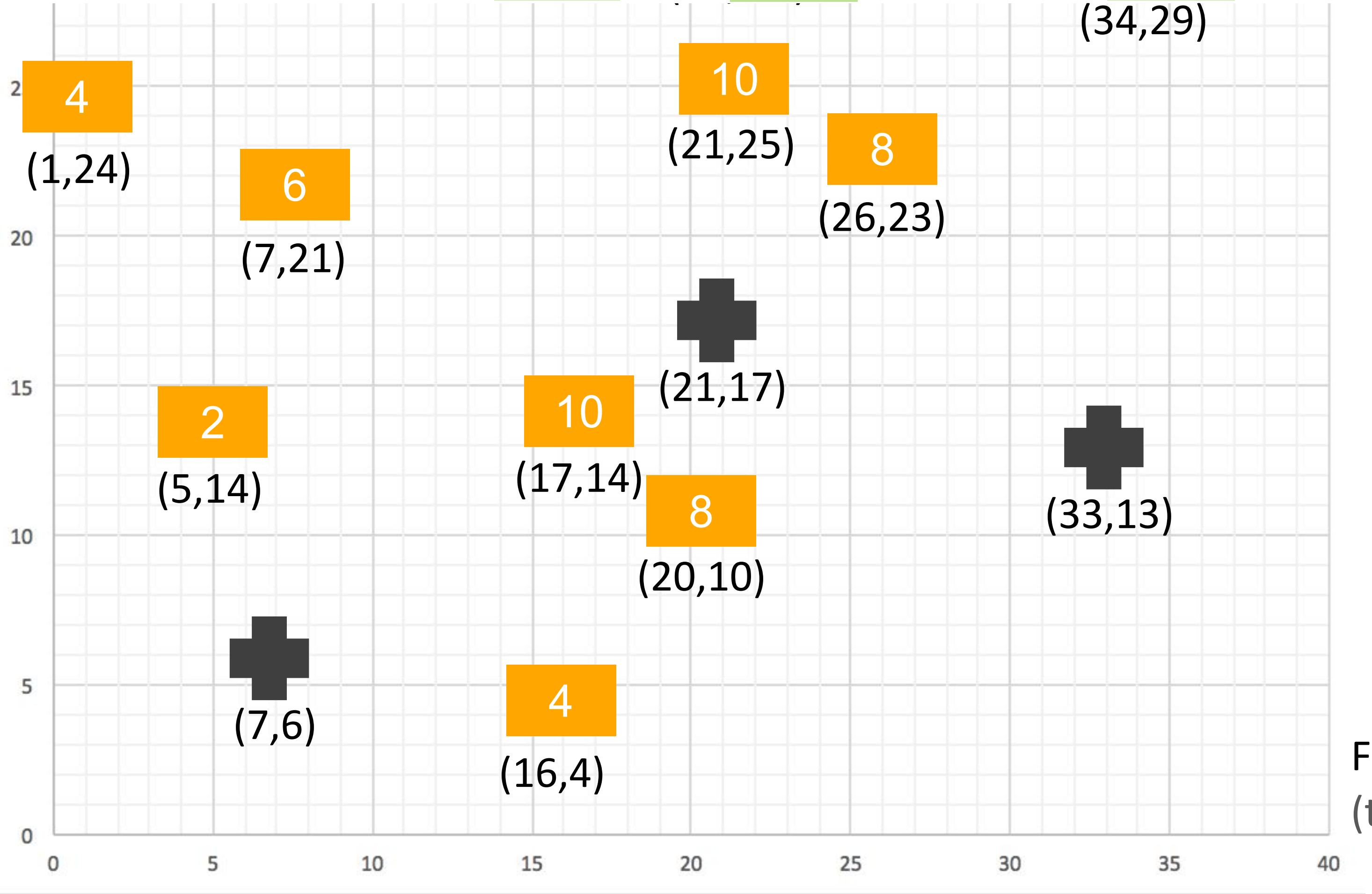


Final classification: 2

Distances of each training data to the test data

	(1,24)	(8, 30)	(7,21)	(22, 30)	(5,14)	(20,10)	(16,4)	(26,23)	(21,25)	(17,14)	(34,29)	$k = 3$
(21,17)	21.19	18.38	14.56	13.03	16.28	7.07	13.93	7.81	8.00	5.00	17.69	8.667
(33,13)	33.84	30.23	27.20	20.24	28.01	13.34	19.23	12.20	16.97	16.03	16.03	8.667
(7,6)	18.97	24.02	15.00	28.30	8.24	13.60	9.21	25.49	23.60	12.80	35.46	5.667

temperature	humidity	bacterial growth
1	24	4
8	30	10
7	21	6
22	30	10
5	14	2
20	10	8
16	4	4
26	23	8
21	25	10
17	14	10
34	29	2



Features/dimensions (X):
(temperature, humidity)

K-Nearest Neighbors (KNN)

- Assumes all dimensions correspond to points in a d-dimensional space $\mathbb{R}^d$.
  - temperature, humidity ($\mathbb{R}^2$)

- **Features** may be discrete or continuous.

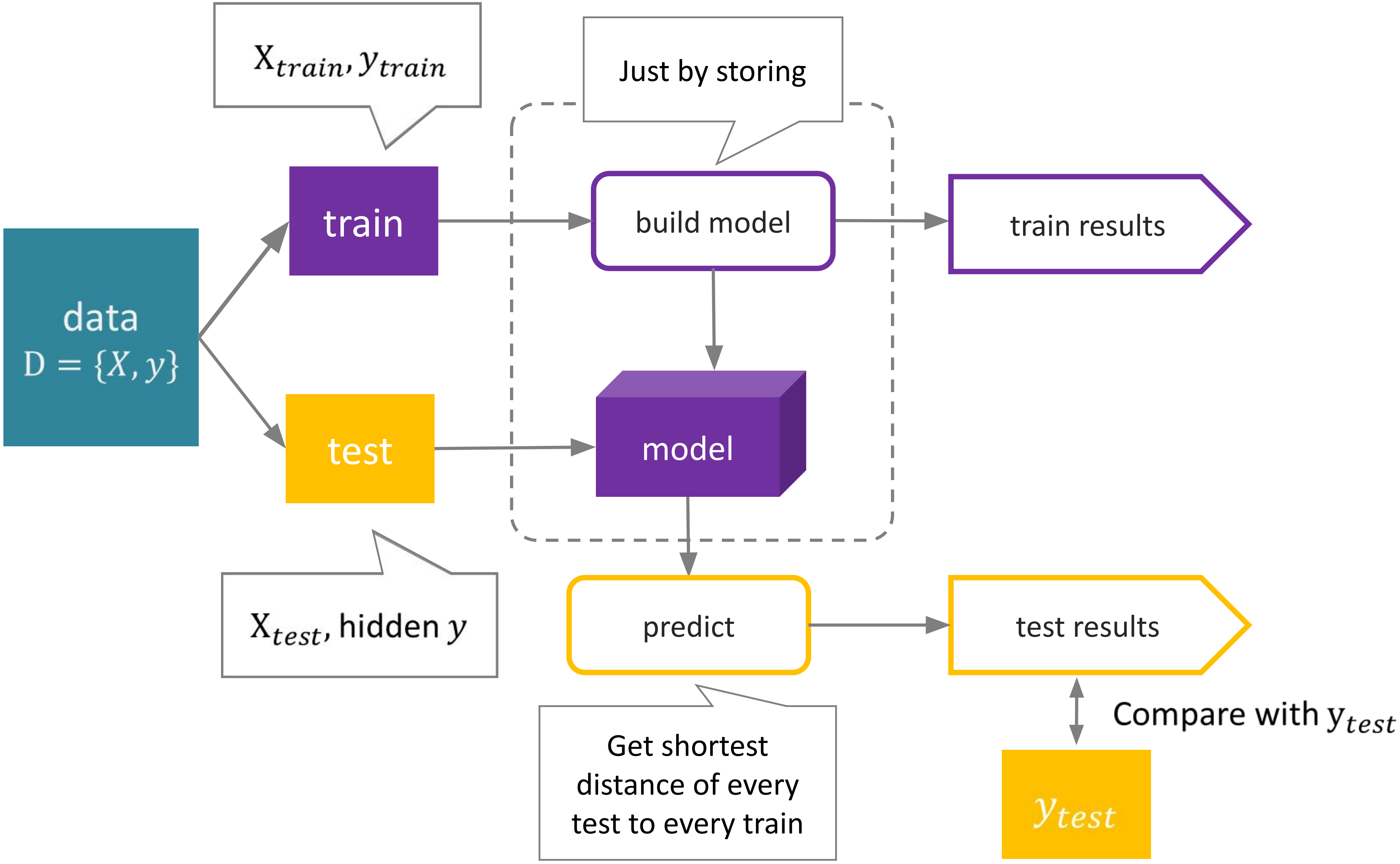
- **Labels** can be continuous (regression) or categorical (classification) as well.

The Distance Function Matters!

- Other less-commonly used distance functions:

  - **Minkowski Distance**
    - generalization of Euclidean and Manhattan distance
  - **Cosine distance**
    - similarity between two vectors
  - **Hamming distance**
    - similarity between two strings

How to Choose k?



data

$D = \{X, y\}$

train

X_{train}, y_{train}

test

X_{test}, hidden y

Just by storing

build model

model

train results

predict

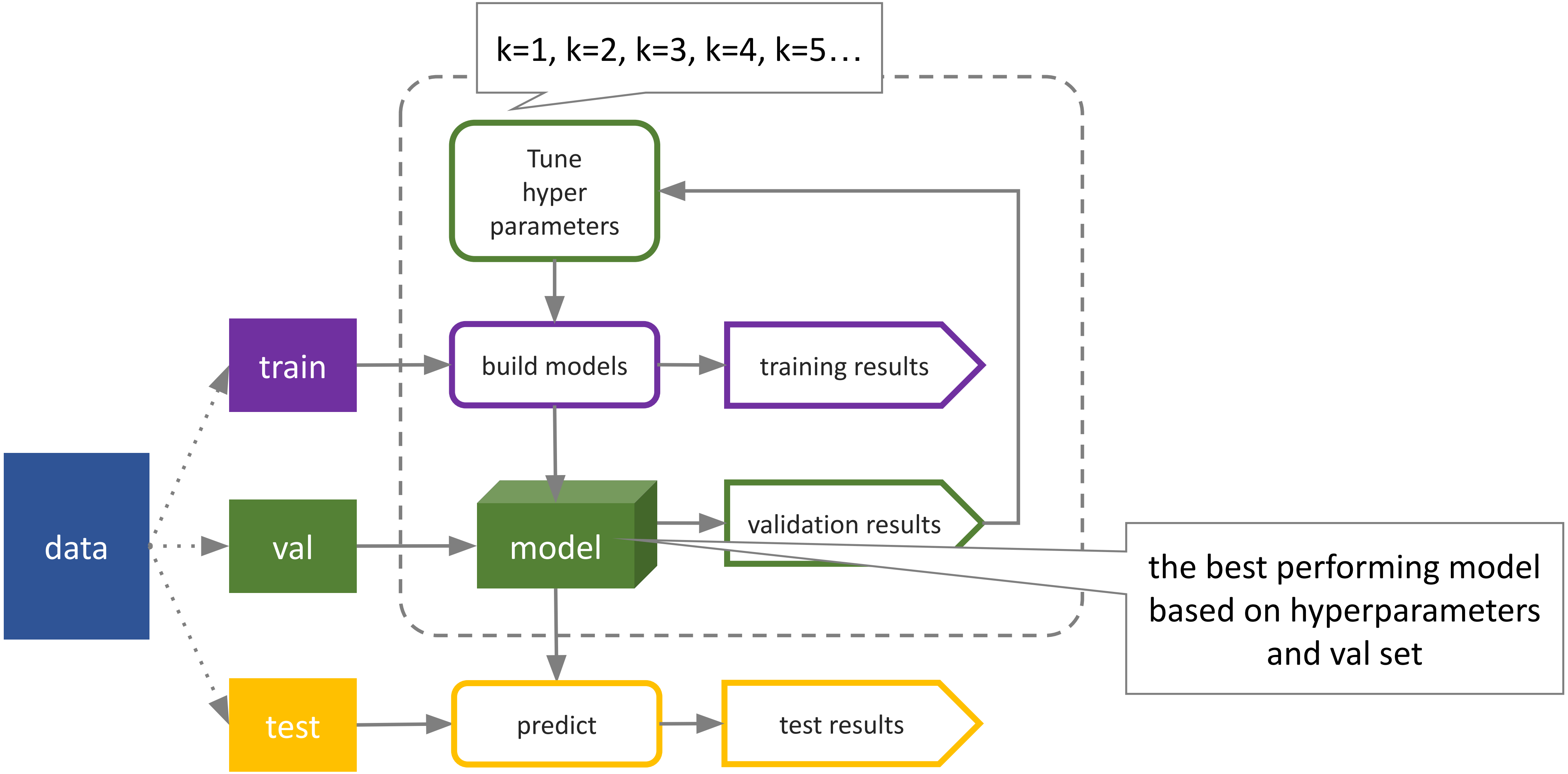
Get shortest distance of every test to every train

test results

Compare with y_{test}

y_{test}

Hyperparameter Tuning



k=1, k=2, k=3, k=4, k=5…

Tune hyper parameters

train

build models

training results

data

val

model

validation results

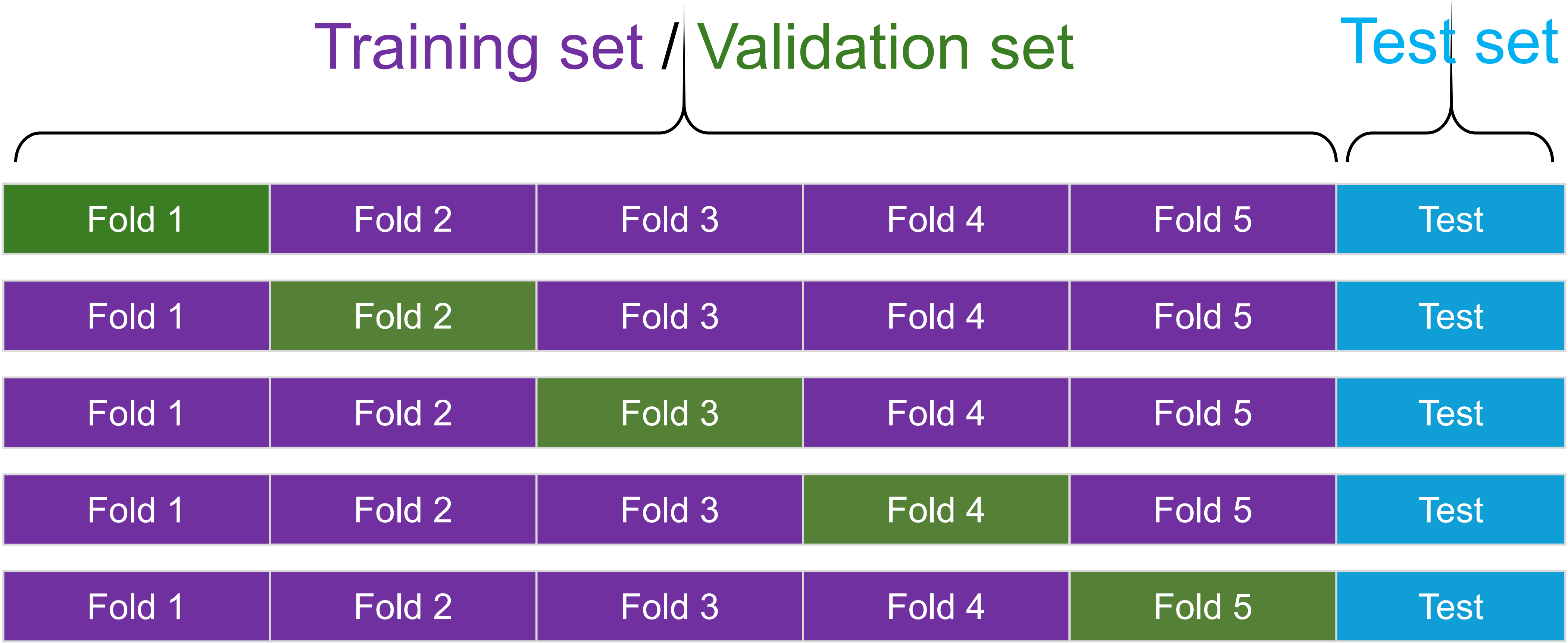
the best performing model based on hyperparameters and val set

test

predict

test results

Hyperparameter Tuning with Cross-fold Validation



Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test
Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Test

KNN Advantages and Disadvantages

- **Advantages**
  - Fast training time
  - Straightforward and easy to implement

- **Disadvantages**
  - Model is large
  - Prediction is slow if dataset is large
  - Considers all features equally, regardless of whether they are relevant or not