COMP90049 Introduction to Machine Learning

K-Nearest Neighbours Classifier

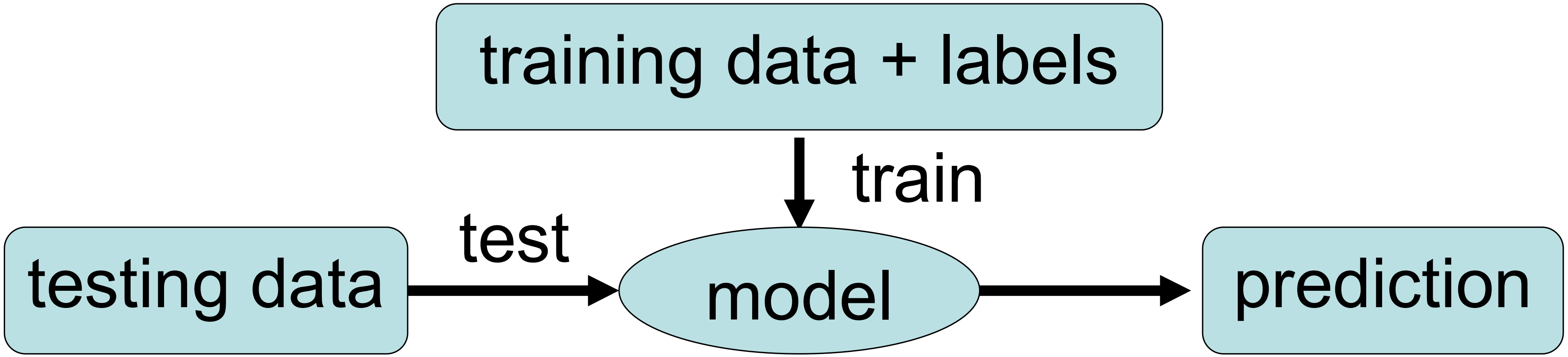
Semester 2, 2020

Hadi Khorshidi, CIS

The Slides prepared by Qiuhong Ke, CIS

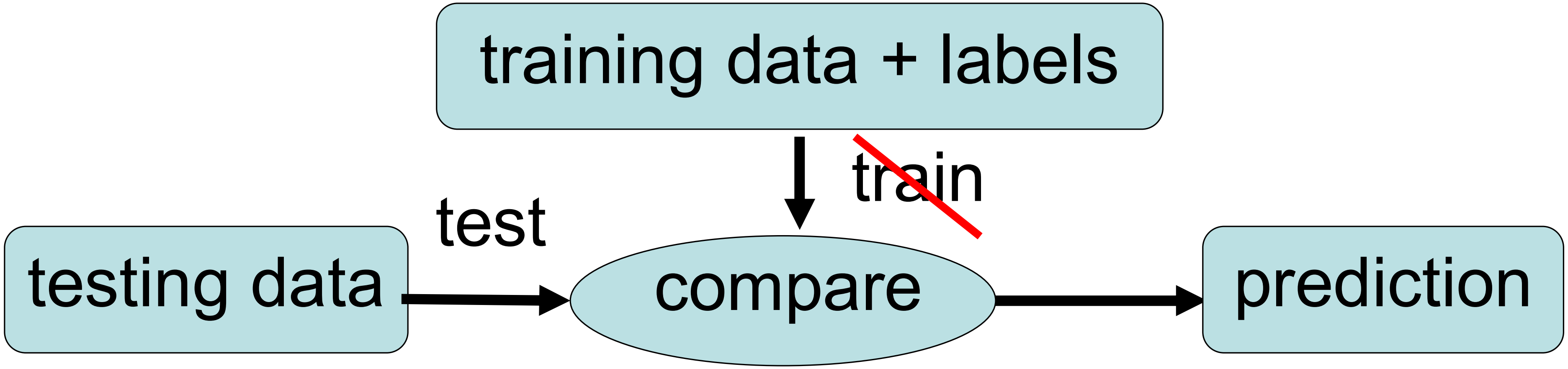
Eager Learning vs Lazy Learning

- Eager learning:
  - train a generalization model using training data
  - use the model to predict a new test sample.



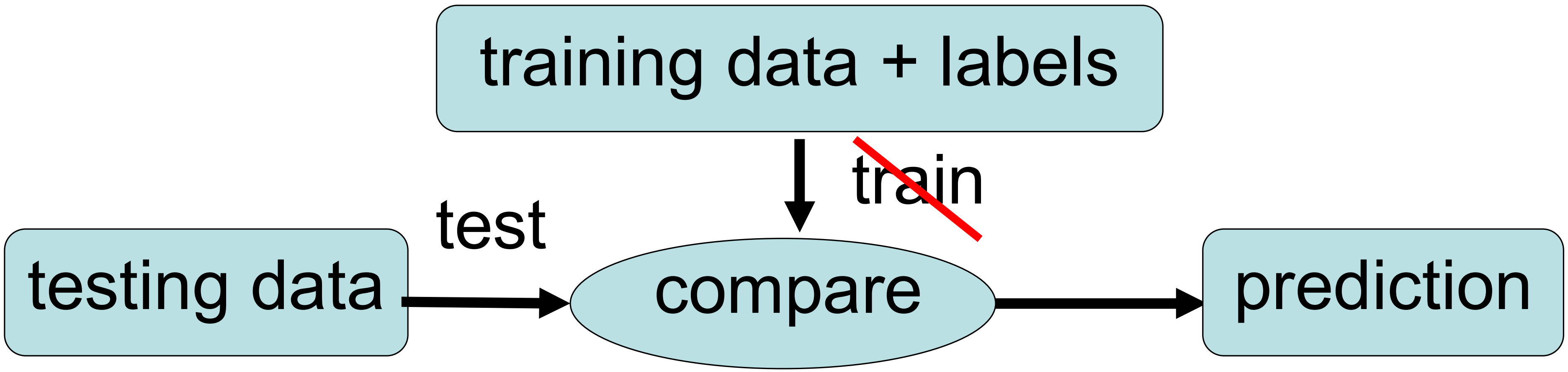
Eager Learning vs Lazy Learning

- Lazy Learning (Instance-based Learning):
  - no training of the model, store training data
  - prediction: compares new testing instances with stored instances



Eager Learning vs Lazy Learning

- K-NN is a lazy learner or Instance-based Learning method.
  - how to compare the training and testing data?
  - how to predict the class for the testing data?



Outline

- Introduction of K-NN
- Distance Measure
  - Features
  - Distance Metrics
- K-NN for Classification
- Summary

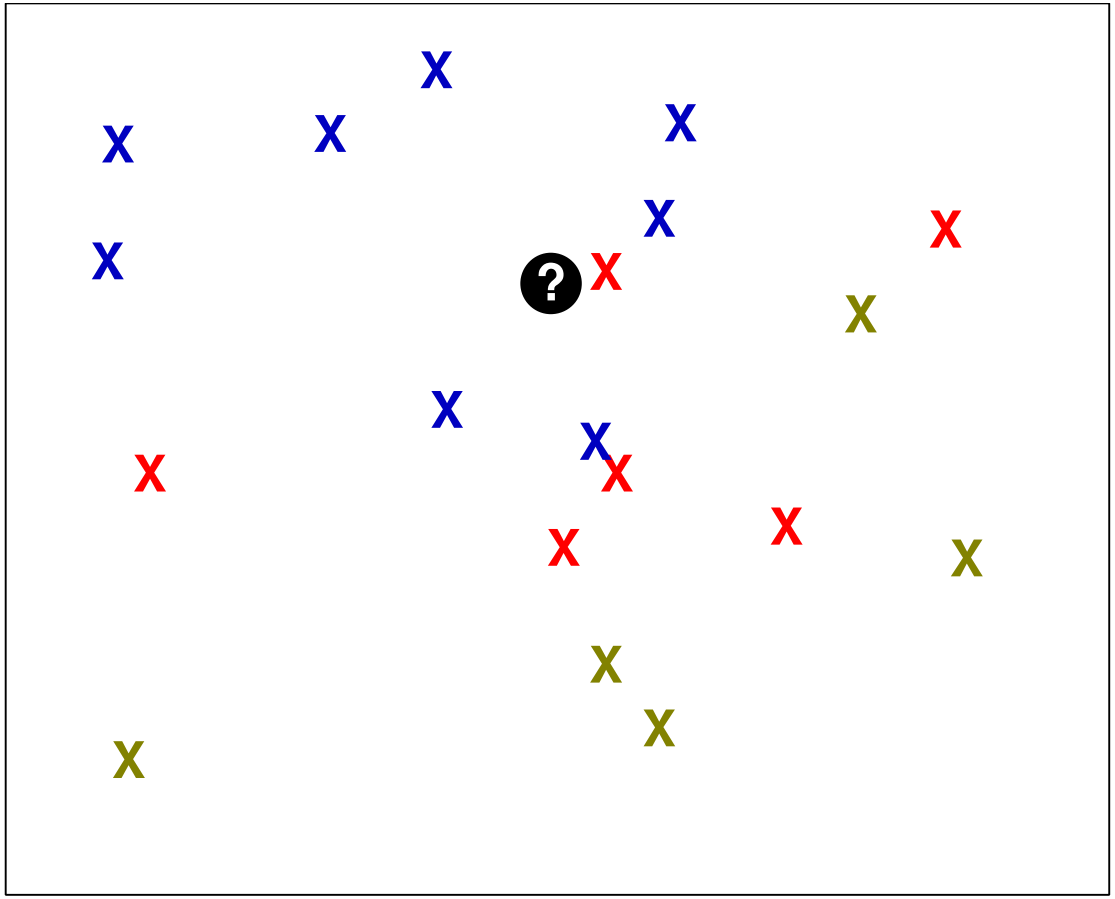
Outline

- Introduction of K-NN
- Distance Measure
  - Features
  - Distance Metrics
- K-NN for Classification
- Summary

K-NN Classifier

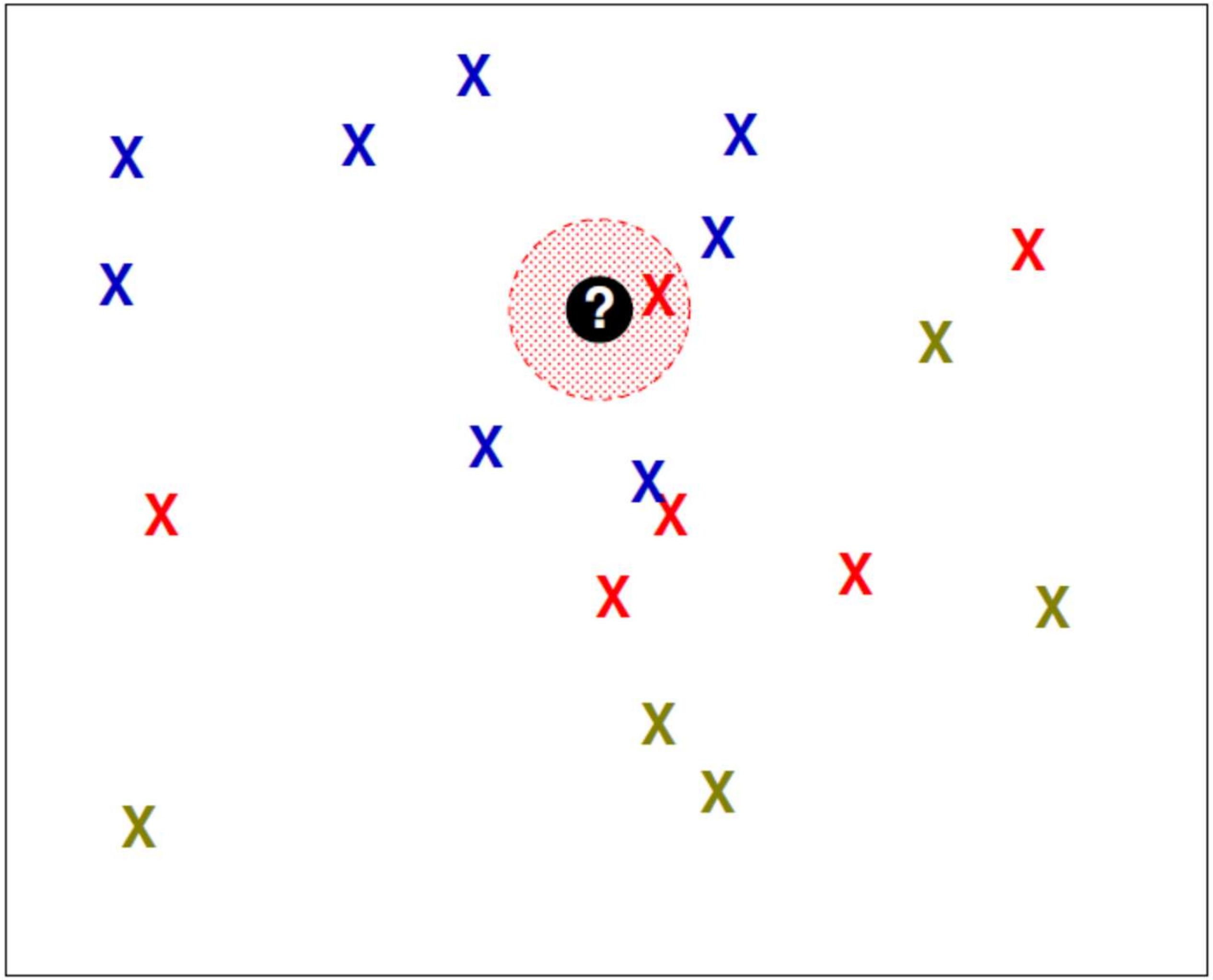
- Store training instances.
- Testing:
  - Measure the similarity (or distance) between the testing and training data
  - Find K-Nearest neighbours
  - Return the class of the testing data using the corresponding labels of the K-Nearest neighbours.

K-Nearest Neighbours



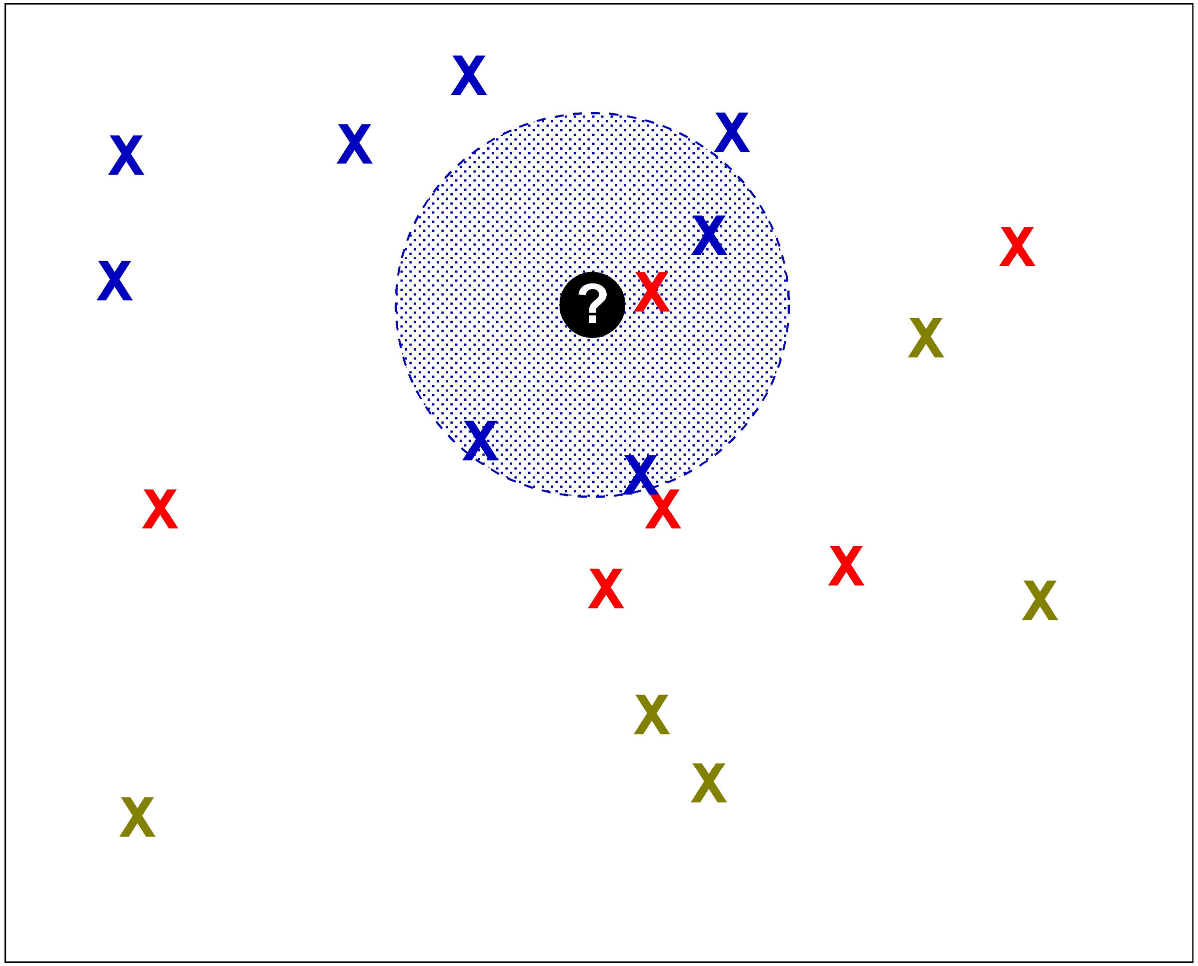
K-nearest neighbours: k closest data points

1-NN



k=1: the closest data point of the record

4-NN



k=4: 4 closest data points of the record

Why K-NN Classifier

- Intuitive and simple
- No assumptions
- No training: new data-> evolve and adapt immediately
- used for classification and regression

Outline

- Introduction of K-NN
- Distance Measure
  - Features
  - Distance Metrics
- K-NN for Classification
- Summary

Distance vs Similarity

- Distance (Dissimilarity)
  - Numerical measure of how different two data objects are
  - Lower when objects are more alike
  - No negative distances. $d(x, y) \geq 0$

- Similarity
  - Numerical measure of how alike two data objects are.
  - Is higher when objects are more alike.

Feature Vectors

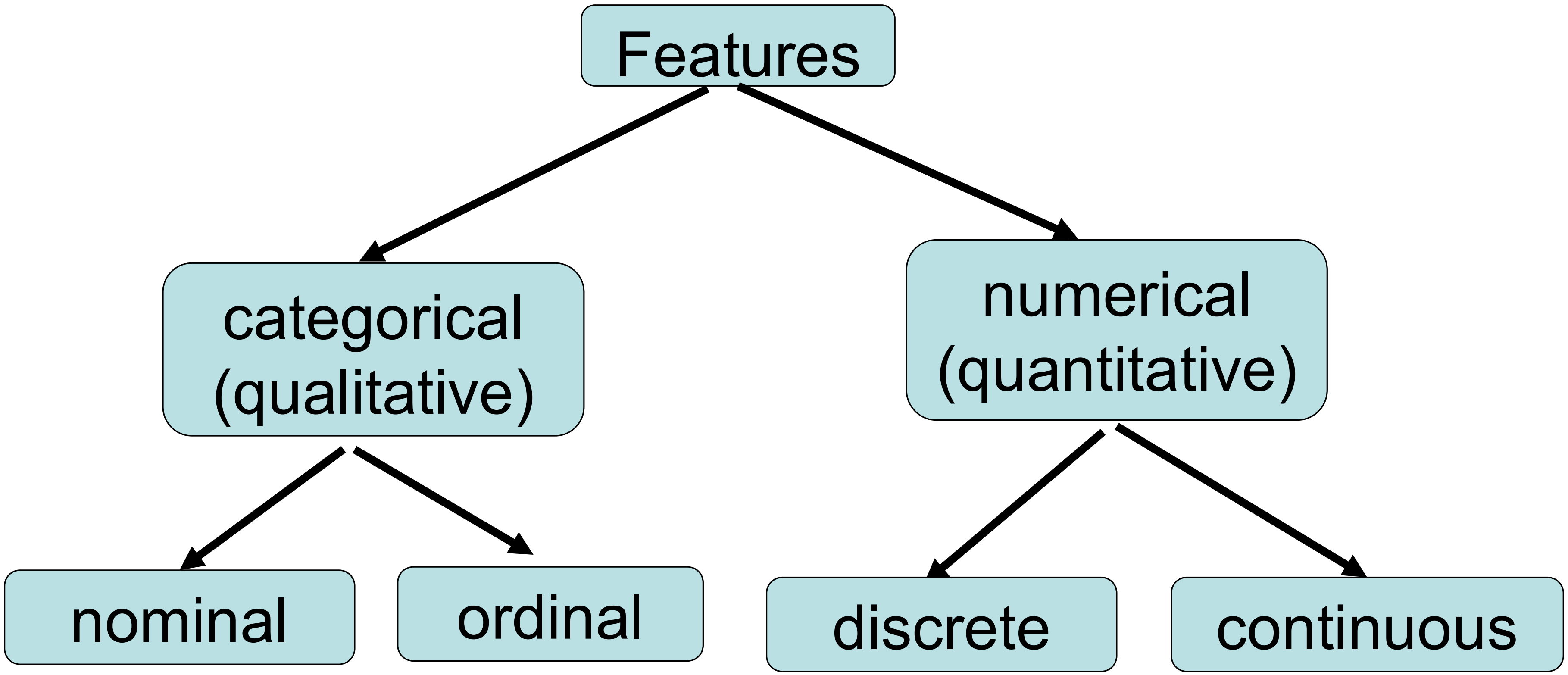
- an n-dimensional vector of features

	Outlook	Temperature	Humidity	Windy	Play
a	sunny	hot	high	FALSE	no
b	sunny	hot	high	TRUE	no
c	overcast	hot	high	FALSE	yes
d	rainy	mild	high	FALSE	yes
e	rainy	cool	normal	FALSE	yes
f	rainy	cool	normal	TRUE	no

$$\text{feature vector} = \begin{bmatrix} \text{Outlook} \\ \text{Temperature} \\ \text{Humidity} \\ \text{Windy} \end{bmatrix}$$

Features

description (characteristic, quality) of an object.



Feature Types

- Nominal: unordered categories, mutually exclusive,
  - colours= {black, brown, red, grey, white}
  - marital status, occupation
- Ordinal: ordered categories, mutually exclusive
  - temperature: cool < mild < hot
- Discrete: certain values, can be counted.
  - e.g., number of visits
- Continuous: any value within a range,
  - e.g., height.

Compare Nominal Feature Vectors

- Simple Matching: k: # of matches (same features), m: total # of features

$$d = \frac{m-k}{m}$$

- Example:

feature	apple	banana
shape	round	curved
colour	red	yellow
taste	sweet	sweet

$$d = \frac{3-1}{3} = \frac{2}{3}$$

Compare Nominal Feature Vectors

- Use many binary variables: one for each of the M nominal states (values)

feature	apple	banana
shape	round	curved
colour	red	yellow
taste	sweet	sweet

state	apple	banana
round	1	0
curved	0	1
red	1	0
yellow	0	1
sweet	1	1

Compare Nominal Feature Vectors

- Simple Matching: K: # of matches (same states), M: total # of nominal states

$$d = \frac{M - K}{M}$$

state	apple	banana
round	1	0
curved	0	1
red	1	0
yellow	0	1
sweet	1	1

$$dis = \frac{5-1}{5} = \frac{4}{5}$$

Compare Ordinal Feature Vectors

Order is important.

- sort the value, return a rank

  $$r \in \{1, \ldots, M\}$$

- rank ->numerical values:

  $$z = \frac{r-1}{M-1}$$

- Compute the distance (using methods for comparing numerical variables)

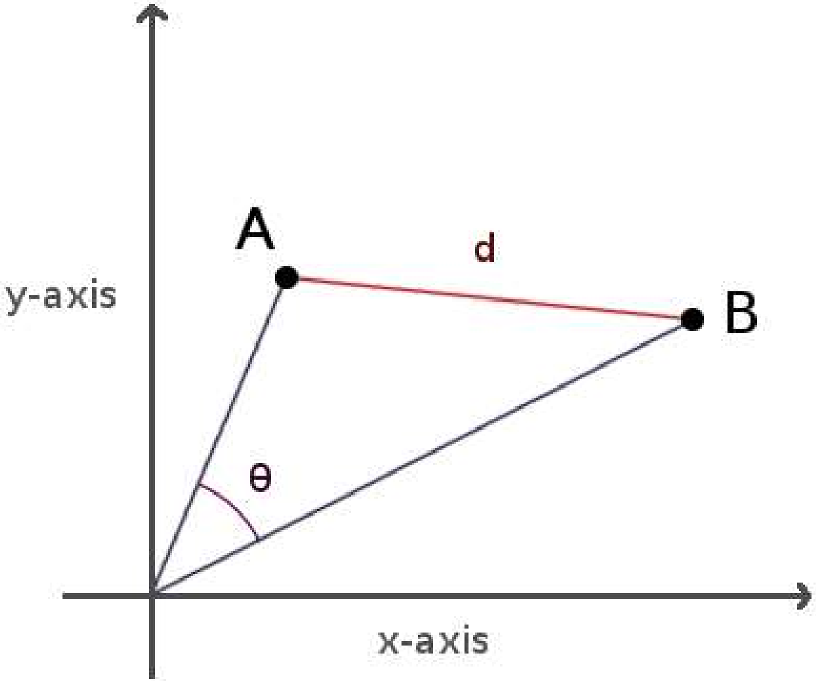
feature	A	B
safety	0	2
comfortable	-2	1
convenient	-1	2

feature	A	B
safety	2/4	4/4
comfortable	0	3/4
convenient	1/4	4/4

rating:
-2: very dissatisfied. -1: dissatisfied
0: indifference. 1: satisfied.
2: very satisfied

Compare Numerical Feature Vectors : Euclidean Distance

Given two items A and B, and their feature vectors a and b, respectively, we can calculate their distance d in Euclidean space:



In n-dimensional space:

$$d(A, B) = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2}$$

Compare Numerical Feature Vectors : Euclidean Distance

$$d(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

feature	A	B
safety	2/4	4/4
comfortable	0	3/4
convenient	1/4	4/4

$$d(A,B) = \sqrt{(\frac{2}{4} - 1)^2 + (0 - \frac{3}{4})^2 + (\frac{1}{4} - 1)^2} = 1.17$$

Example

$$z = \frac{r-1}{M-1}$$

$$d(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

temperature: cold<mild<hot
humidity: low<normal<high

cold, mild, hot: 1,2,3
low, normal, high: 1, 2 ,3

feature	A	B	C
temperature	hot	mild	cold
humidity	low	high	normal

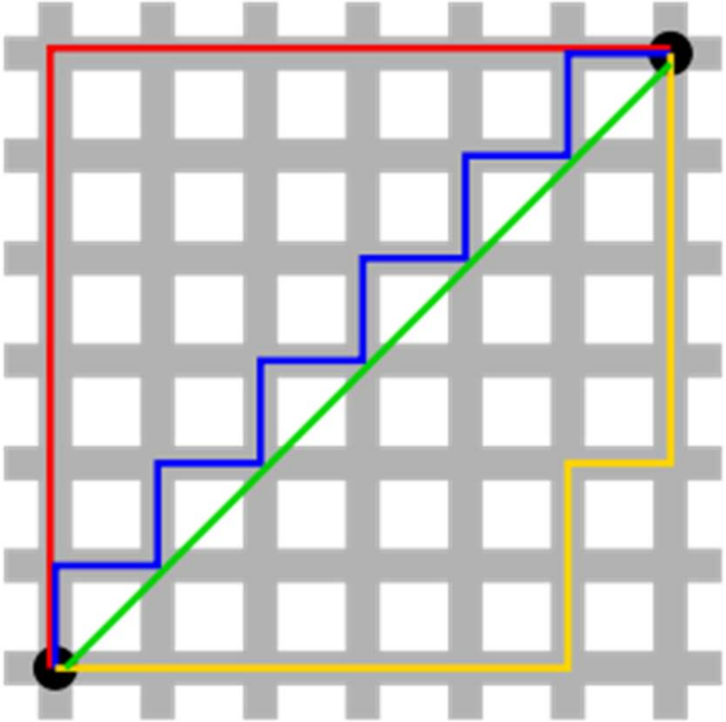
feature	A	B	C
temperature	1	1/2	0
humidity	0	1	1/2

$$d(A,B) = \sqrt{(1-\frac{1}{2})^2 + (0-1)^2} = 1.12$$

$$d(B,C) = \sqrt{(\frac{1}{2}-0)^2 + (1-\frac{1}{2})^2} = 0.71$$

Compare Numerical Feature Vectors : Manhattan Distance

[“City block” distance or “Taxicab geometry” or “L1 distance”] Given two items A and B, and their corresponding feature vectors a and b, respectively, we can calculate their distance d based on their absolute differences of each feature (cartesian coordinates):



$$d(A,B) = \sum_{i=1}^{n} |a_i - b_i|$$

Example

$$z = \frac{r-1}{M-1}$$

$$d(A,B) = \sum_{i=1}^{n} |a_i - b_i|$$

temperature: cold<mild<hot
humidity: low<normal<high

cold, mild, hot: 1,2,3
low, normal, high: 1, 2 ,3

feature	A	B	C
temperature	hot	mild	cold
humidity	low	high	normal

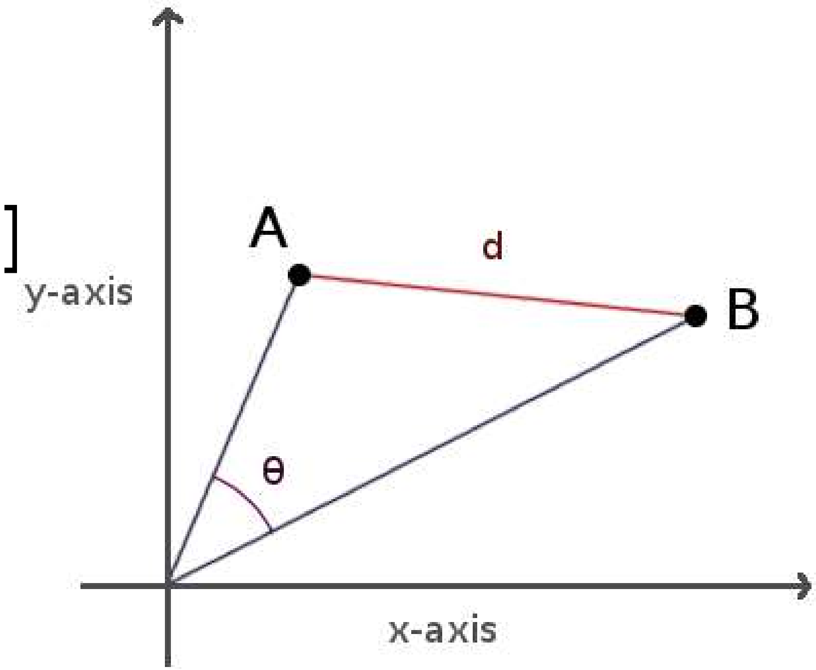
feature	A	B	C
temperature	1	1/2	0
humidity	0	1	1/2

$$d(A,B) = \left|1 - \frac{1}{2}\right| + |0-1| = 1.5$$

$$d(B,C) = \left|\frac{1}{2} - 0\right| + \left|1 - \frac{1}{2}\right| = 1$$

Compare Numerical Feature Vectors : Cosine Similarity

- the magnitudes are ignored
- similarity range [-1,1]:
  angle 0: similarity 1
  angle π : similarity -1
- distance:1-similarity: range [0,2]



$$cos(A,B) = \frac{a \cdot b}{|a||b|} = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2}\sqrt{\sum_i b_i^2}}$$

Compare Numerical Feature Vectors : Cosine Similarity

used for data that the magnitude of the vectors is not important

e.g., text data represented by word counts:

- larger document-> larger word count
- topic is more important, not the size
- Using cosine similarity to ignore the magnitude.

Example

$$cos(A,B) = \frac{a \cdot b}{|a||b|} = \frac{\sum_i a_i b_i}{\sqrt{\sum_i a_i^2}\sqrt{\sum_i b_i^2}}$$

feature	A	B	B_s
Word 1	200	300	50
Word 2	300	200	40
Word 3	200	100	25

$$cos(A,B) = \frac{200 \times 300 + 300 \times 200 + 200 \times 100}{\sqrt{200^2 + 300^2 + 200^2} \times \sqrt{300^2 + 200^2 + 100^2}} = 0.91$$

$$cos(B,B\_s) = \frac{300 \times 50 + 200 \times 40 + 100 \times 25}{\sqrt{300^2 + 200^2 + 100^2} \times \sqrt{50^2 + 40^2 + 25^2}} = 0.99$$

Example

$$d(A,B) = \sqrt{\sum_{i=1}^{n}(a_i - b_i)^2}$$

feature	A	B	B_s
Word 1	200	300	50
Word 2	300	200	40
Word 3	200	100	25

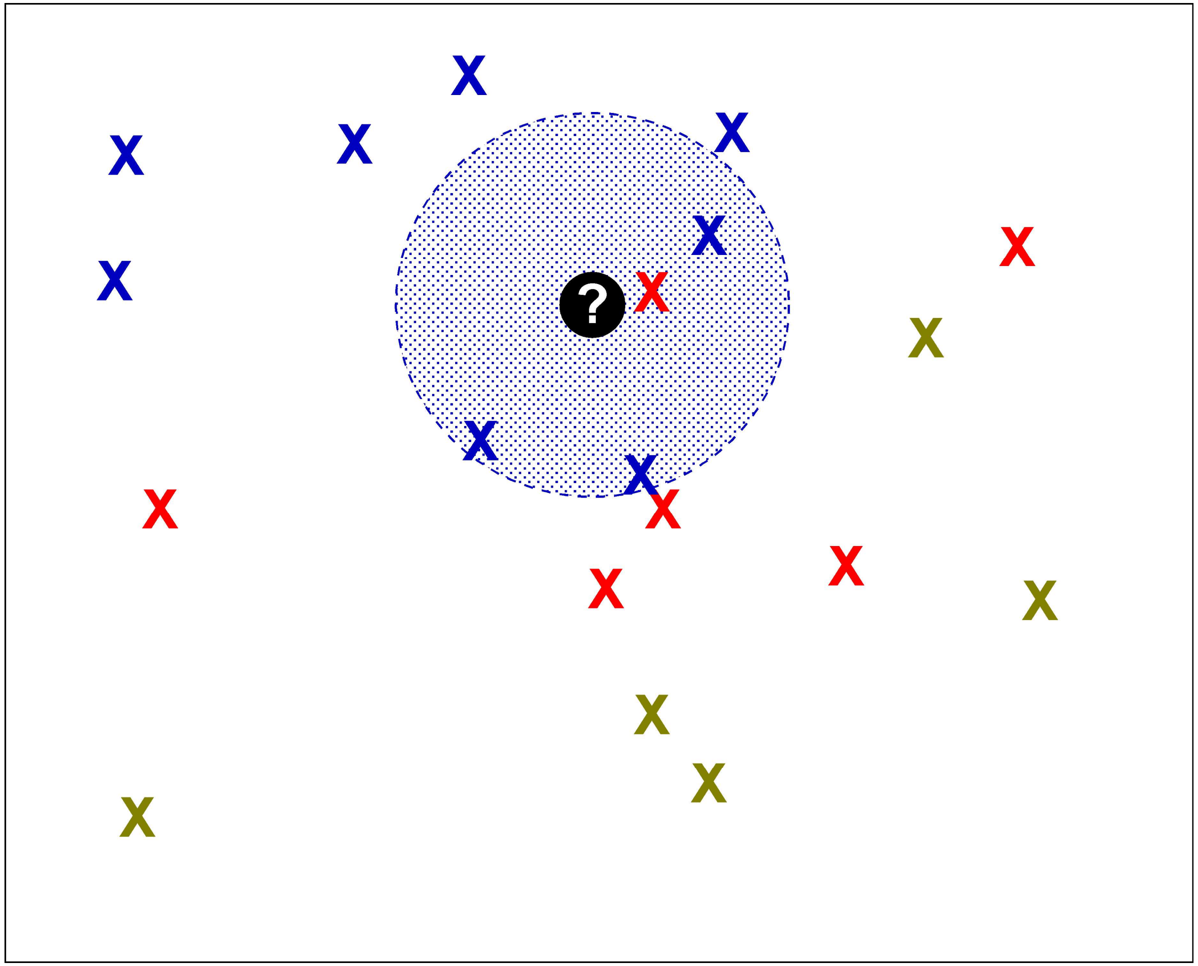
$$d(A,B)$$

$$= \sqrt{(200-300)^2+(300-200)^2+(200-100)^2} = 173.2$$

$$d(B,B\_s)$$

$$= \sqrt{(300-50)^2+(200-40)^2+(100-25)^2} = 306.2$$

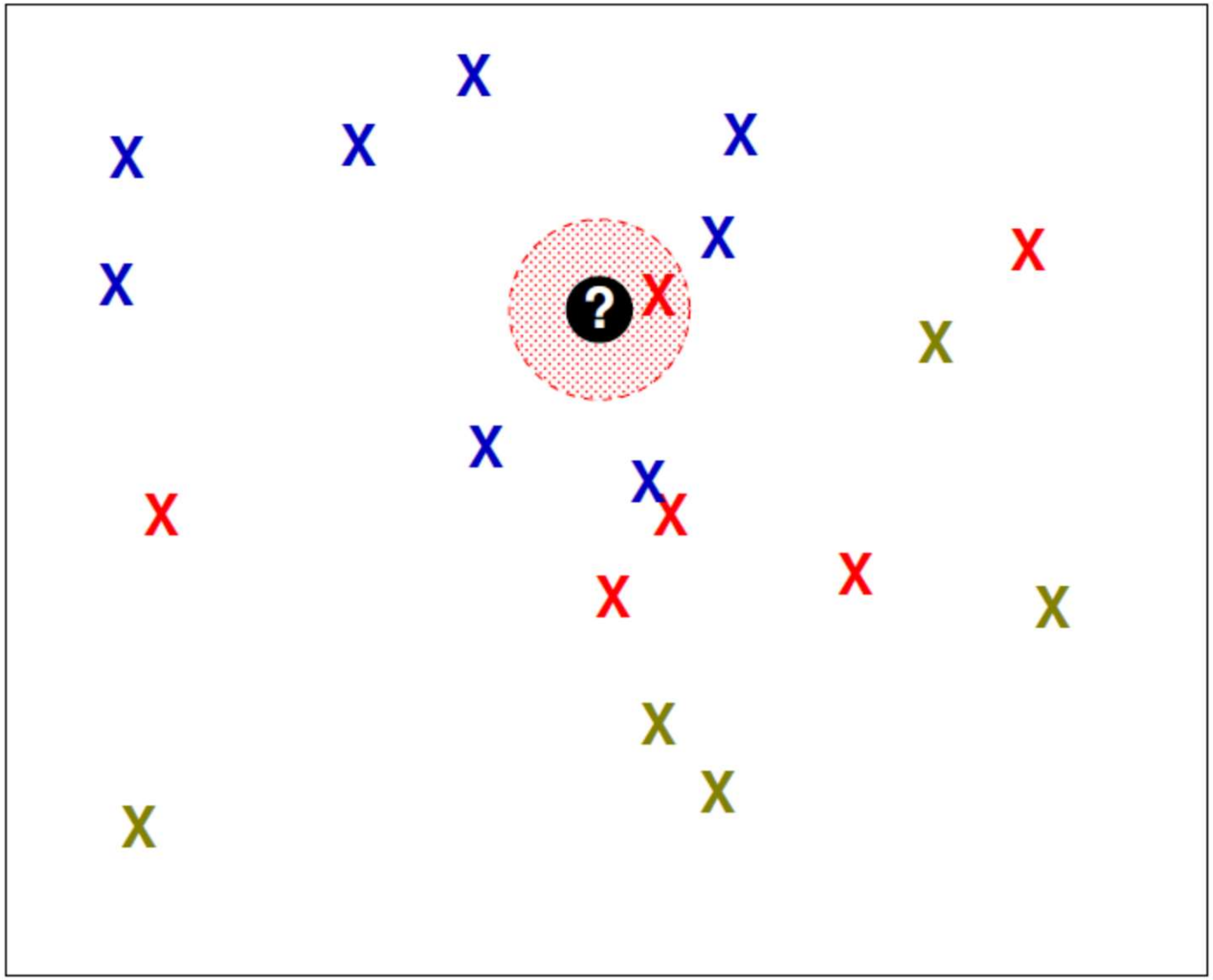
K-NN



Outline

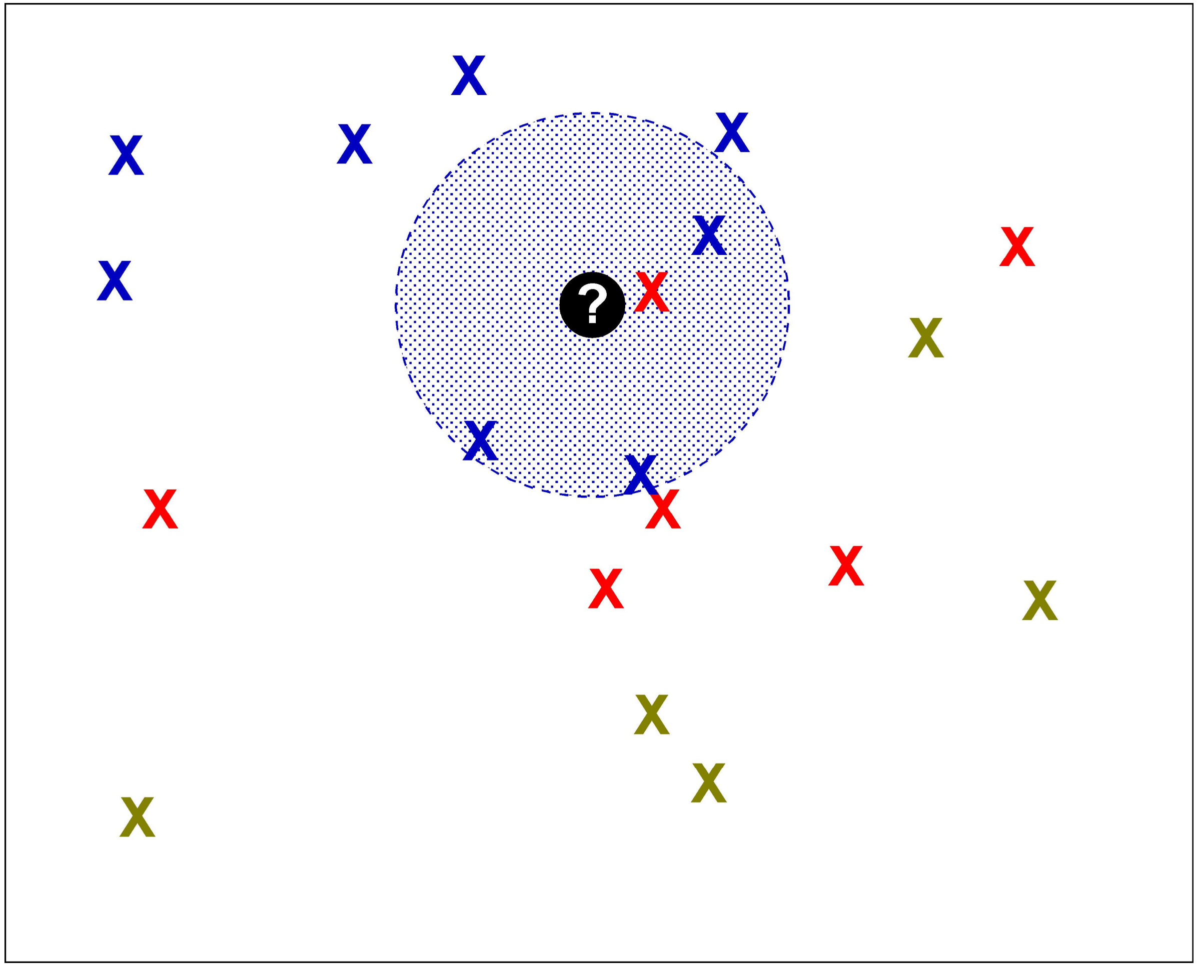
- Introduction of K-NN
- Distance Measure
  - Features
  - Distance Metrics
- K-NN for Classification
- Summary

1-NN



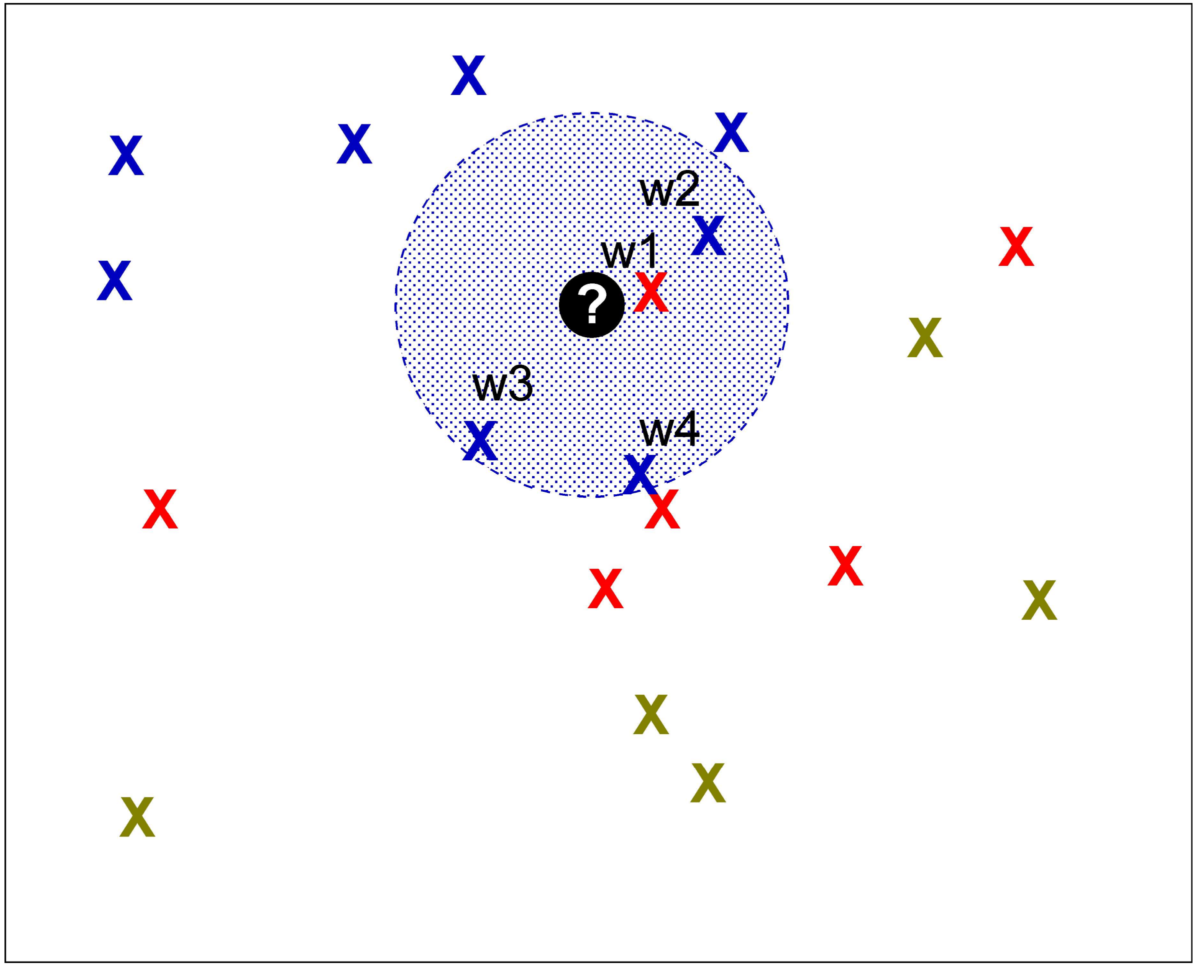
- 1-NN: Classify the test input according to the class of the closest training instance.

K-NN



- K-NN: Classify the test input according to the majority class of the K-Nearest training instances.

Weighted K-NN



- blue: w2+w3+w4
- red: w1

- weighted K-NN: Classify the test input according to the weighted accumulative class of the K-Nearest training instances

Weighting Strategies

There are a number of strategies for weighting:

- Give each neighbour equal weight (= classify according to the majority class of set of neighbours)
- Weight the vote of each instance by its **inverse linear distance (ILD)** from the test instance:

$$w_j = \frac{d_k - d_j}{d_k - d_1}$$

Where
d_1 is the minimum distance between the neighbours and the test instance;
d_k is the maximum distance between the neighbours and the test instance
d_j is the distance between the jth neighbour and the test instance

Weighting Strategies

- Weight the vote of each instance by its **inverse distance (ID)** from the test instance:

$$w_j = \frac{1}{d_j + \epsilon}$$

ϵ is a small constant (e.g., $1e - 10$ (10^{-10})) to make sure the denominator is not zero

- Weighted accumulative class of the K-Nearest training instances for each class:

$$c = \sum w_j$$

Voting Strategies in Action (k = 3)

Instance	Class	Distance
1	Yes	0
2	No	1
3	No	1.5

$$w_j = \frac{d_k - d_j}{d_k - d_1}$$

majority class voting: no = 2 vs. yes = 1

ILD-based voting:

yes = $\frac{1.5-0}{1.5-0}$=1

vs.

no = $\frac{1.5-1}{1.5-0}+\frac{1.5-1.5}{1.5-0}$ =0.33+0=0.33

Voting Strategies in Action (k = 3)

Instance	Class	Distance
1	Yes	0
2	No	1
3	No	1.5

$$w_j = \frac{1}{d_j + \epsilon}$$

ID-based voting (ϵ= 1e-10):

yes= $\frac{1}{0+\epsilon} = 1e10$

vs.

no = $\frac{1}{1+\epsilon} + \frac{1}{1.5+\epsilon} = 1.67$

Voting Strategies in Action (k = 5)

Instance	Class	Distance
1	Yes	0
2	No	1
3	No	1.5
4	Yes	1.75
5	No	2

ILD-based voting:

$$w_j = \frac{d_k - d_j}{d_k - d_1}$$

ID-based voting:

$$w_j = \frac{1}{d_j + \epsilon}$$

majority class voting: yes = 2 vs. no = 3

ILD-based voting:

yes = $\frac{2-0}{2-0} + \frac{2-1.75}{2-0}$ =1+0.125=1.125

vs.

no = $\frac{2-1}{2-0} + \frac{2-1.5}{2-0} + \frac{2-2}{2-0}$ =0.5+0.25+0=0.75

Voting Strategies in Action (k = 5)

Instance	Class	Distance
1	Yes	0
2	No	1
3	No	1.5
4	Yes	1.75
5	No	2

ILD-based voting:

$$w_j = \frac{d_k - d_j}{d_k - d_1}$$

ID-based voting:

$$w_j = \frac{1}{d_j + \epsilon}$$

ID-based voting (ϵ = 1e-10):

yes= $\frac{1}{0+\epsilon} + \frac{1}{1.75+\epsilon} = 1e10$

vs.

no = $\frac{1}{1+\epsilon} + \frac{1}{1.5+\epsilon} + \frac{1}{2+\epsilon} = 2.17$

Breaking Ties

In the case that we have an equal number of votes for different classes, we need some tie breaking mechanism:

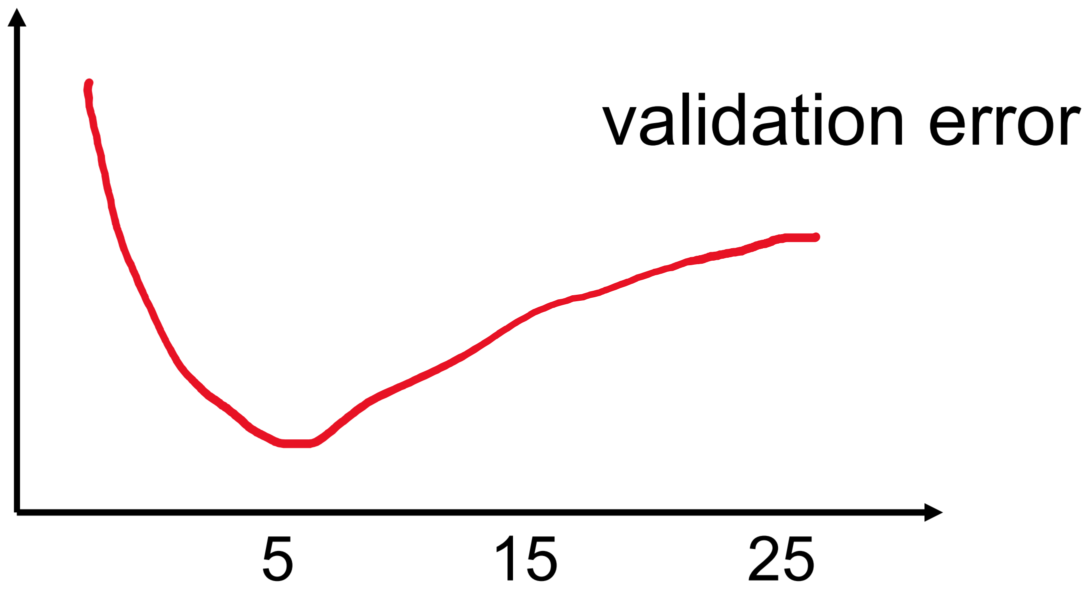
- random tie breaking
- take class with highest prior probability
- see if the addition of the k + 1th instance breaks the tie

Choosing the Value of k

- Smaller k:
  - jagged decision boundary
  - noise
  - lower classifier performance
- Larger k:
  - smooth decision boundary,
  - includes unrelated classes
  - lower classifier performance
  - ➢ K==N: Zero-R (simply predicts the majority class) performance

Choosing the Value of k

- Choose k by cross validation
- Note: k is generally set to an odd value



Weaknesses of NN methods

Weaknesses

- Distance function: what?
- Combining the labels of multiple neighbours: how?
- Arbitrary K value
- Expensive if data set is large.
  - Typical implementation: brute-force computation of distances between a test instance and every training instance.
  - efficient if data is small
  - infeasible if data grows

Quiz

K-NN algorithm spends more computation time on testing than train.

A) TRUE
B) FALSE

Summary

Today: K-Nearest Neighbour

- Distance metrics for different feature types
- Euclidean distance, Manhattan distance, Cosine similarity
- Majority voting, Inverse Linear Distance (ILD) and Inverse Distance (ID) for weighted K-Nearest Neighbour classification

Next lecture:

- *Coding demo*
- *Optimization Part II*

References

Some slides are from:

- *Altman, Naomi S. (1992). "An introduction to kernel and nearest-neighbor nonparametric regression" (PDF). The American Statistician. 46 (3): 175-185.*
- *Data Mining: Concepts and Techniques, 2nd ed., Jiawei Han and Micheline Kamber, Morgan Kaufmann, 2006. Chapter 2.*
- *Tan et al . Introduction to Data Mining.2006. Chapter 2. Tan et al . Introduction to Data Mining. 2006.* Chapter 5, Section 5.2
- Jeremy Nicholson & Tim Baldwin & Karin Verspoor: Machine Learning
- https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn
- https://en.wikipedia.org/wiki/Instance-based_learning
- https://towardsdatascience.com/a-simple-introduction-to-k-nearest-neighbors-algorithm-b3519ed98e
- https://www.c-sharpcorner.com/article/knn-k-nearest-neighbors/
- https://en.wikipedia.org/wiki/Cosine_similarity