

# \*\*\* Applied Machine Learning Fundamentals \*\*\*

## $k$ -Nearest Neighbors

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

Winter term 2020/2021



Find all slides on [GitHub](#) (DaWe1992/Applied\_ML\_Fundamentals)

# Lecture Overview

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Probability Density Estimation
Unit V	Regression
<b>Unit VI</b>	<b>Classification I</b>
Unit VII	Evaluation
Unit VIII	Classification II
Unit IX	Clustering
Unit X	Dimensionality Reduction

# Agenda for this Unit

## 1 Introduction

- Overview of the Algorithm
- Derivation of the Algorithm

## 2 Distance Metrics

- Properties of Distance Metrics
- Minkowski, Manhattan, Euclidean
- Cosine Similarity

## 3 $k$ -nearest Neighbors Algorithm

- General Procedure
- Calculation of Distances

Prediction of the Class Label

## 4 Choice of $k$

- Danger of Overfitting
- Selection Strategies

## 5 Wrap-Up

- Summary
- Self-Test Questions
- Lecture Outlook
- Recommended Literature and further Reading
- Meme of the Day

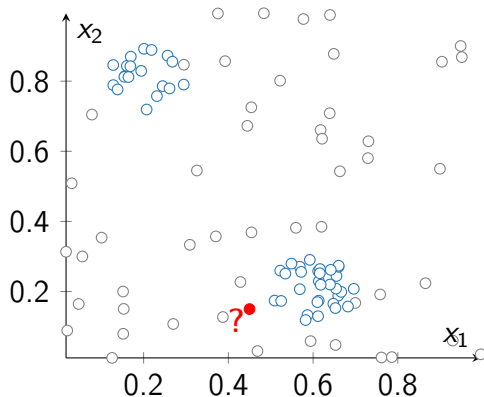
Section:  
**Introduction**



# Introduction

- **Basic idea:** Predict the class label based on nearby known examples
- Instance-based learning, a. k. a. **lazy learning**

We do not learn any model,  
the data speaks for itself!





# Derivation of the Algorithm

- Unconditional density:

$$p(\mathbf{x}) = \frac{k}{n \cdot v}$$

- Class priors:

$$p(\mathcal{C}_j) = \frac{n_j}{n}$$

Remember non-parametric  
density estimation?

Combine using Bayes' theorem:

$$p(\mathcal{C}_j|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_j) \cdot p(\mathcal{C}_j)}{p(\mathbf{x})} = \frac{\frac{k_j}{n_j \cdot v} \cdot \frac{n_j}{n}}{\frac{k}{n \cdot v}} = \frac{k_j}{k} \quad (1)$$

Section:  
**Distance Metrics**



# Distance Metrics

- How to measure the distance between two data points  $i$  and  $j$ ?  
 $\Rightarrow$  **distance metrics**
- Let  $d$  be a function  $d : (u, v) \mapsto \mathbb{R}^+$  (including 0)
- Function  $d$  has the following properties:

- ①  $d(u, v) = d(v, u)$  (**commutativity**)
- ②  $d(u, v) = 0 \Rightarrow u = v$
- ③  $d(u, k) \leq d(u, v) + d(v, k)$  (**triangle inequality**)



## Distance Metrics (Ctd.)

Minkowski distance:

$$d_p(u, v) = \left( \sum_{j=1}^m |x_j^{(u)} - x_j^{(v)}| \right)^{1/p} \quad (2)$$

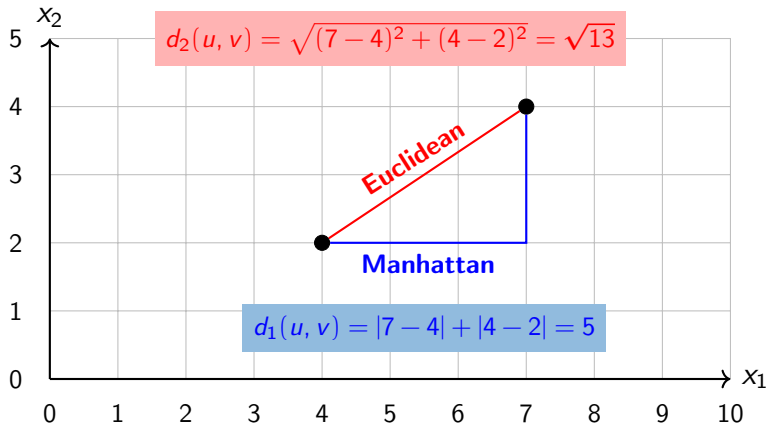
Manhattan distance: ( $p = 1$ )

$$d_1(u, v) = \sum_{j=1}^m |x_j^{(u)} - x_j^{(v)}|$$

Euclidean distance: ( $p = 2$ )

$$d_2(u, v) = \sqrt{\sum_{j=1}^m |x_j^{(u)} - x_j^{(v)}|^2}$$

## Distance Metrics (Ctd.)



# Cosine Similarity

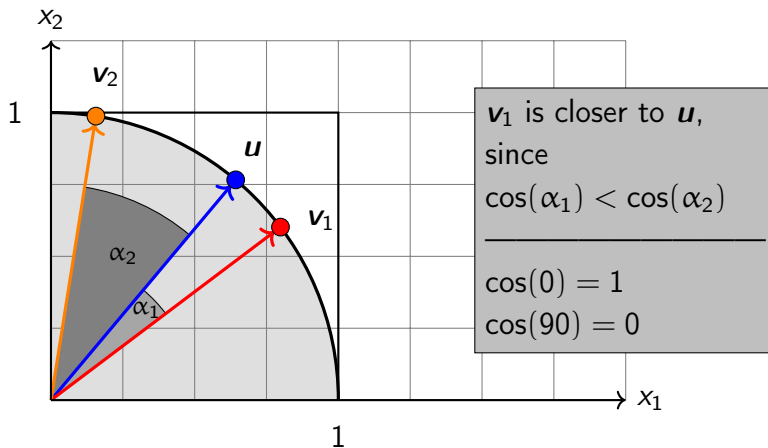
- **Similarity metrics** are an alternative to distance metrics
- **Example: Cosine similarity**
- The cosine similarity of two vectors  $\mathbf{a}$  and  $\mathbf{b}$  is the cosine of the angle:

$$\cos \angle(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \cdot \|\mathbf{b}\|} = \frac{\sum_{j=1}^m a_j \cdot b_j}{\sqrt{\sum_{j=1}^m (a_j)^2} \cdot \sqrt{\sum_{j=1}^m (b_j)^2}} \quad (3)$$

- The dot product is defined as:

$$\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \cdot \|\mathbf{b}\| \cdot \cos \angle(\mathbf{a}, \mathbf{b}) \quad (4)$$

## Cosine Similarity (Ctd.)



Section:  
*k*-nearest Neighbors Algorithm



## Prediction with $k$ -Nearest Neighbors (Ctd.)

### $k$ -nearest neighbors algorithm:

- 1 Calculate the distances between the new data point and **all data points in the data set**
- 2 Sort the data points by distances **in ascending order**  
*(if similarity metrics are used, sort in descending order)*
- 3 Look at the first  $k$  examples and **count how often each class occurs**
- 4 Predict the class with **the maximum score**

# ① Calculation of Distances

$v$	$x_1$	$x_2$	$\mathcal{C}$	$d_2(u, v)$
1	0.66	0.24	1	0.23
2	0.25	0.79	1	0.67
3	0.16	0.81	1	0.73
4	0.57	0.21	1	0.13
5	0.21	0.72	1	0.62
6	0.66	0.27	1	0.24
7	0.27	0.11	0	0.19
8	0.39	0.13	0	0.07
9	0.39	0.86	0	0.71
10	0.44	0.67	0	0.52
11	0.31	0.33	0	0.23
12	0.03	0.51	0	0.55
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

- $\mathbf{x}^{(u)} = (0.45, 0.15)$
- Calculate the **Euclidean distance** between  $\mathbf{x}^{(u)}$  and all other data points  $\mathbf{x}^{(v)}$

Prediction is expensive!

## ②/③/④ Prediction of the Class Label

- Let  $k$  be set to 10
- Step ②: Sort data set by distances  
(*cf. table right*)
- Step ③: Count the classes
  - Class 0: 3
  - Class 1: 7
- Step ④: Predict class 1!

$x_1$	$x_2$	$\mathcal{C}$	$d_2(u, v)$
0.51	0.17	1	0.06
0.39	0.13	0	0.07
0.52	0.17	1	0.08
0.43	0.23	0	0.08
0.47	0.03	0	0.12
0.52	0.26	1	0.13
0.57	0.21	1	0.13
0.53	0.25	1	0.13
0.58	0.12	1	0.14
0.59	0.13	1	0.14
$\vdots$	$\vdots$	$\vdots$	$\vdots$

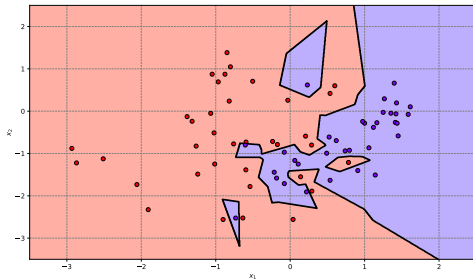


Section:  
Choice of  $k$

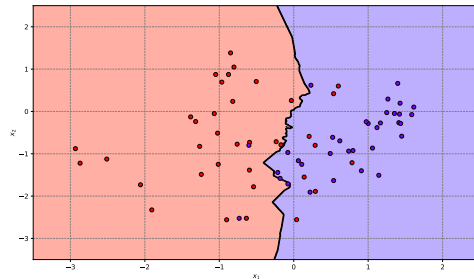


# How to choose $k$ ?

The choice of  $k$  is important:



$k = 1$  (💀 overfitting 💀)



$k = 30$  (about right)

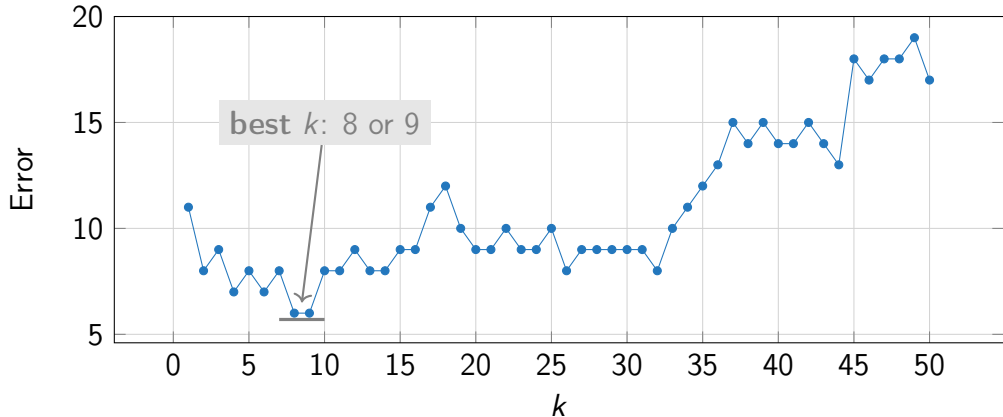
## How to choose $k$ ? (Ctd.)

- First of all, it is recommended to use **odd values** for  $k$   
(*no tie-breaking necessary*)
- Compute  $k$  depending on the size of the data set  $\mathcal{D}$ :

$$k = \sqrt{\frac{n}{2}} \quad \text{or} \quad k = \sqrt{n} \quad (5)$$

- **Other strategy:** Evaluate different  $k$  on a dev set and choose the best one

## How to choose $k$ ? (Ctd.)



Section:  
**Wrap-Up**



# Summary

- The basic idea is to classify unknown instances **based on nearby examples**
- The algorithm is an example for **instance-based learning**
- **Distance metrics** allow to calculate the distance between data points:
  - Manhattan distance
  - Euclidean distance
  - Cosine similarity
- Choose the value of *k* wisely:
  - Too small: **Overfitting**
  - Too large: **Underfitting**



# Self-Test Questions

- ① Outline the *k*-nearest neighbors algorithm.
- ② What is instance-based learning (in contrast to model-based learning)?
- ③ How can you compute distances? What properties do distance metrics have?
- ④ What is the intuition behind the triangle inequality?
- ⑤ How can you choose *k*?
- ⑥ Suppose you have a data set comprising  $n = 50$  examples.  
If you set  $k = n$ , what class does the algorithm predict?
- ⑦ What are advantages and disadvantages of the algorithm?

# What's next...?

Unit I	Machine Learning Introduction
Unit II	Mathematical Foundations
Unit III	Bayesian Decision Theory
Unit IV	Probability Density Estimation
Unit V	Regression
<b>Unit VI</b>	<b>Classification I</b>
Unit VII	Evaluation
Unit VIII	Classification II
Unit IX	Clustering
Unit X	Dimensionality Reduction



# Recommended Literature and further Reading I



## [1] Machine Learning

*Tom Mitchell. McGraw-Hill Science. 1997.*

→ [Link](#), cf. chapter 8.2

## Meme of the Day



Thank you very much for the attention!

**Topic:** \*\*\* Applied Machine Learning Fundamentals \*\*\* *k*-Nearest Neighbors

**Term:** Winter term 2020/2021

**Contact:**

Daniel Wehner, M.Sc.

SAP SE / DHBW Mannheim

[daniel.wehner@sap.com](mailto:daniel.wehner@sap.com)

Do you have any questions?