

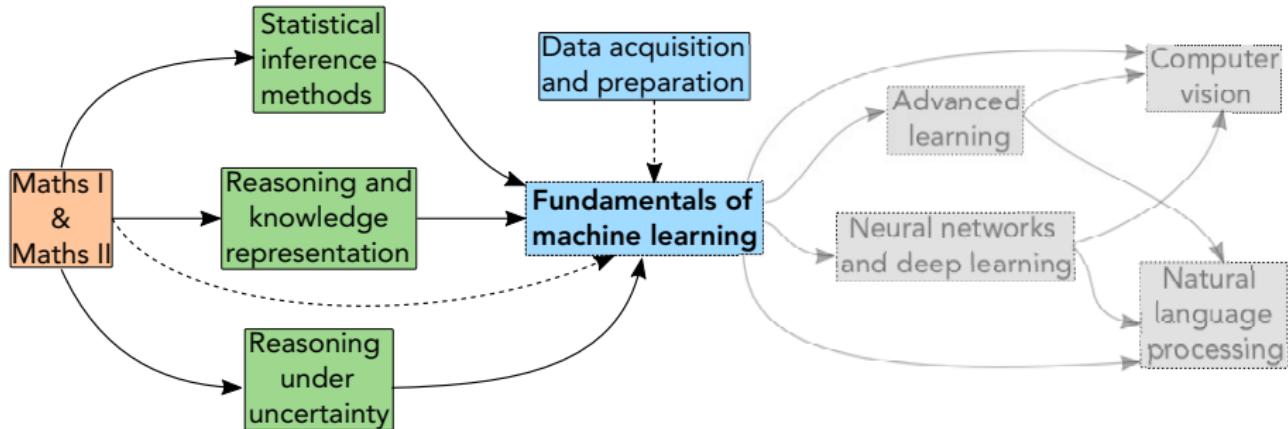
Fundamentos del Aprendizaje Automático

Grado en Ingeniería en Inteligencia Artificial

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante

Curso 2025/2026

Context



Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Objectives

- ① Understand the foundations of machine learning.
- ② Formalize machine learning problems from a mathematical and statistical perspective.
- ③ Select, train, and evaluate supervised and unsupervised learning models.
- ④ Apply data preprocessing techniques and choose appropriate evaluation metrics.
- ⑤ Use and interpret model evaluation and validation methods.
- ⑥ Critically analyse the results obtained with different algorithms and justify model design and selection decisions.

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, underfitting, overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression metrics, methods, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

KNN, kernel methods, density estimation

5 Linear methods and perceptron (~2-3 s.)

Perceptron, linear regression, logistic regression

6 Unsupervised learning (~1-2 s.)

Clustering, PCA, t-SNE

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, p-values, AIC, BIC, cross-validation, information criteria

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression metrics, including cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear models

6 Unsupervised learning (~1-2 s.)

Clustering, PCA

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, p-values, AIC, BIC, cross-validation, ROC curves, F1 score

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression scenarios, metrics, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear models, perceptrons, activation, gradient, backpropagation

6 Unsupervised learning (~1-2 s.)

Clustering, PCA

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, p-values, AIC, BIC, cross-validation, ROC curves

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression scenarios, metrics, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear regression, logistic regression, perceptron, gradient descent

6 Unsupervised learning (~1-2 s.)

Clustering, metrics

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, p-values, confidence intervals, AIC, BIC, cross-validation

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression scenarios, metrics, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear models, perceptron, activation, gradient, backpropagation

6 Unsupervised learning (~1-2 s.)

Clustering, methods

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, statistical significance, pairwise multiple comparisons, post-hoc analysis

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression scenarios, metrics, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear models, perceptron, activation, gradient, backpropagation

6 Unsupervised learning (~1-2 s.)

Clustering, metrics

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, statistical significance, pairwise multiple comparisons, post-hoc analysis

Syllabus

1 Introduction to Machine Learning (~1-2 s.)

Concepts, taxonomy, historical evolution, areas and applications

2 Computational learning (~3 s.)

Decision theory, error probability, empirical risk minimization, model likelihood, under/overfitting, dimensionality

3 Model evaluation (~2-3 s.)

Classification and regression scenarios, metrics, cross-validation and model selection

4 Nonparametric and distance-based learning (~2-3 s.)

Distance-based classifiers, metrics, k-Nearest neighbor, Voronoi diagram

5 Linear methods and perceptron (~2-3 s.)

Linear models, perceptron, activation, gradient, backpropagation

6 Unsupervised learning (~1-2 s.)

Clustering, metrics

7 Statistical methods for model comparison (~2 s.)

Hypothesis testing, statistical significance, pair-wise/multiple comparison, post-hoc analysis

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?

The practical sessions will be organized in three main parts:

- Individual work
- Group work
- Self-study

- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?

→ Different exercises in Python (basic scripts/programs; notebooks)
→ Individual work

- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?
 - Different exercises in Python (basic scripts/programs; notebooks)
 - Report (2-3 long)
- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?
 - Different exercises in Python (basic scripts/programs; notebooks)
 - Report (2-3 long)
- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?
 - Different exercises in Python (basic scripts/programs; notebooks)
 - Report (2-3 long)
- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?
 - Different exercises in Python (basic scripts/programs; notebooks)
 - Report (2-3 long)
- Individual development and evaluation of the work
- No exam of the practical sessions

Practical sessions

- Practical contents to support the understanding of the syllabus
- 3 (+1) labs
- What to do?
 - Different exercises in Python (basic scripts/programs; notebooks)
 - Report (2-3 long)
- Individual development and evaluation of the work
- No exam of the practical sessions

Calendar

Date	Lesson	Practical content
09/09	T0: Module description T1: Introduction to Machine Learning	-
16/09	T1: Introduction to Machine Learning T2: Computational learning	P0
23/09	T2: Computational learning	
30/09	T2: Computational learning	P1
07/10	T3: Model evaluation	(delivery: 28/10)
14/10	T3: Model evaluation	
21/10	T4: Nonparametric and distance-based learning	
28/10	T4: Nonparametric and distance-based learning	
04/11	T4: Nonparametric and distance-based learning T5: Linear methods and perceptron	P2 (delivery: 25/11)
11/11	T5: Linear methods and perceptron	
18/11	T5: Linear methods and perceptron	
25/11	T6: Unsupervised learning	
02/12	T6: Unsupervised learning	P3
09/12	T7: Statistical methods for model comparison	(delivery: 23/12)
16/12	T7: Statistical methods for model comparison	

Evaluation

- Ordinary examination period. Two components:
 - Final theoretical exam (50%): written exam
 - Assignment submission (50%): practical assignments based on the contents covered in the lab sessions

Average mark must be ≥ 5 , with a minimum of 4 in each part;
otherwise, failed course

- Extraordinary examination period.

Only if there is a significant drop in average mark

Evaluation

- Ordinary examination period. Two components:
 - **Final theoretical exam** (50%): written exam
 - **Assignment submission** (50%): practical assignments based on the contents covered in the lab sessions

Average mark must be ≥ 5 , with a **minimum of 4 in each part**; otherwise, **failed course**

- Extraordinary examination period.

Only if there is a justified absence from the ordinary examination

Evaluation

- Ordinary examination period. Two components:
 - **Final theoretical exam** (50%): written exam
 - **Assignment submission** (50%): practical assignments based on the contents covered in the lab sessions

Average mark must be ≥ 5 , with a minimum of 4 in each part;
otherwise, failed course

- Extraordinary examination period.
 - Single theoretical-practical exam (100%)

Evaluation

- Ordinary examination period. Two components:
 - **Final theoretical exam** (50%): written exam
 - **Assignment submission** (50%): practical assignments based on the contents covered in the lab sessions

Average mark must be ≥ 5 , with a **minimum of 4 in each part**; otherwise, **failed course**

- Extraordinary examination period.
 - Single theoretical-practical exam (100%)

Evaluation

- Ordinary examination period. Two components:
 - **Final theoretical exam** (50%): written exam
 - **Assignment submission** (50%): practical assignments based on the contents covered in the lab sessions

Average mark must be ≥ 5 , with a **minimum of 4 in each part**; otherwise, **failed course**

- Extraordinary examination period.
 - **Single** theoretical-practical **exam** (100%).

Lecturers



José Javier Valero Mas



Wilson Anthony Mamani Machaca

Bibliography

- Hart, P. E., Stork, D. G., & Duda, R. O. (2001). *Pattern classification*. Hoboken: Wiley.
- Bishop, C. M. (2006). *Pattern recognition and machine learning* (Vol. 4, No. 4, p. 738). New York: springer.
- Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Fundamentos del Aprendizaje Automático

Grado en Ingeniería en Inteligencia Artificial

Departamento de Lenguajes y Sistemas Informáticos
Universidad de Alicante

Curso 2025/2026

T1: Introduction to Machine Learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Definition

What is Machine Learning?

② Structure

Conceptual stages

From (conceptual) stages to (practical) scheme

③ Taxonomies

Introduction

Categorizations

④ Areas and application

Outline

① Definition

What is Machine Learning?

② Structure

Conceptual stages

From (conceptual) stages to (practical) scheme

③ Taxonomies

Introduction

Categorizations

④ Areas and application

What is Machine Learning?

- Trending term nowadays

What is Machine Learning?

- Trending term nowadays
- Nevertheless, the field dates back to the 80s

What is Machine Learning?

- Trending term nowadays
- Nevertheless, the field dates back to the 80s
 - Grounded on statistical reasoning

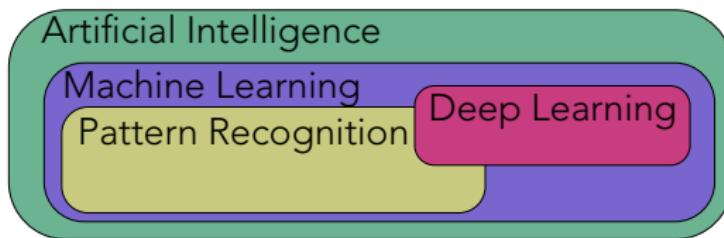
What is Machine Learning?

- Trending term nowadays
- Nevertheless, the field dates back to the 80s
 - Grounded on statistical reasoning
- **Premise:** Infer knowledge from data by computational means

What is Machine Learning?

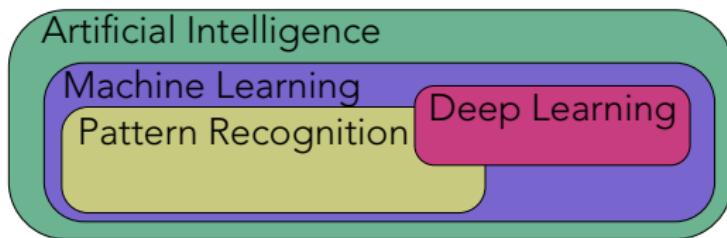
- Trending term nowadays
- Nevertheless, the field dates back to the 80s
 - Grounded on statistical reasoning
- Premise: Infer knowledge from data by computational means
- How does it relate to other terms such as Artificial Intelligence or Pattern Recognition?

What is Machine Learning?



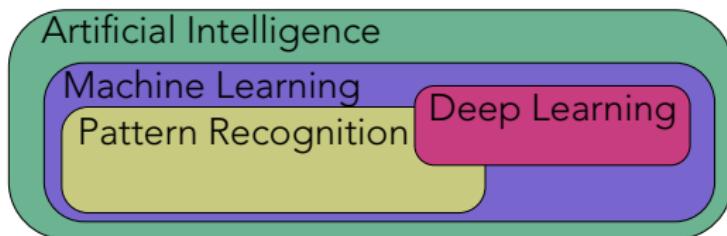
- **Artificial Intelligence:** Mimic human intelligence/intelligent behaviour
 - Could be hand-crafted rules by a programmer
- **Machine Learning:** AI subfield focused on the design of algorithms capable of inferring knowledge from data
- **Pattern Recognition:** ML subfield that mainly focuses on classification tasks with hand-crafted features
- **Deep Learning:** ML subfield that focuses on deep neural models for both feature extraction and knowledge inference

What is Machine Learning?



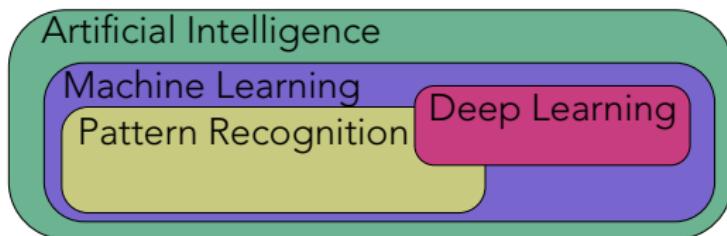
- **Artificial Intelligence:** Mimic [human intelligence](#)/intelligent behaviour
 - Could be hand-crafted rules by a programmer
- **Machine Learning:** AI subfield focused on the design of algorithms capable of inferring knowledge from data
- **Pattern Recognition:** ML subfield that mainly focuses on classification tasks with hand-crafted features
- **Deep Learning:** ML subfield that focuses on deep neural models for both feature extraction and knowledge inference

What is Machine Learning?



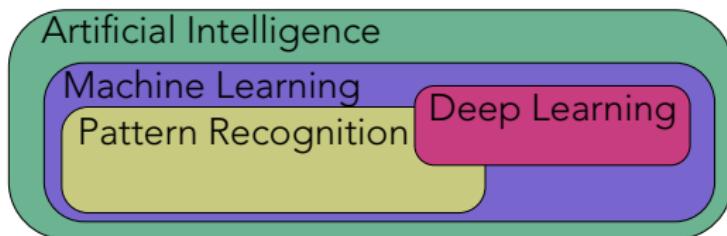
- **Artificial Intelligence:** Mimic **human intelligence**/intelligent behaviour
 - Could be hand-crafted rules by a programmer
- **Machine Learning:** AI subfield focused on the design of algorithms capable of **inferring knowledge from data**
- **Pattern Recognition:** ML subfield that mainly focuses on classification tasks with hand-crafted features
- **Deep Learning:** ML subfield that focuses on deep neural models for both feature extraction and knowledge inference

What is Machine Learning?



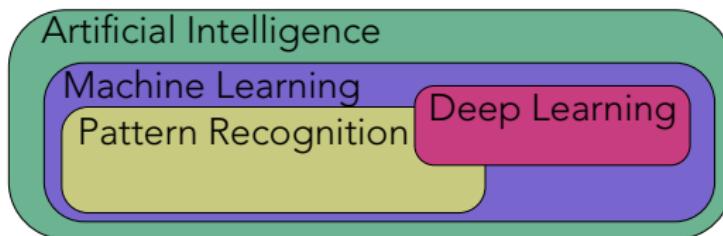
- **Artificial Intelligence:** Mimic **human intelligence**/intelligent behaviour
 - Could be hand-crafted rules by a programmer
- **Machine Learning:** AI subfield focused on the design of algorithms capable of **inferring knowledge from data**
- **Pattern Recognition:** ML subfield that mainly focuses on **classification tasks** with **hand-crafted features**
- **Deep Learning:** ML subfield that focuses on deep neural models for both feature extraction and knowledge inference

What is Machine Learning?



- **Artificial Intelligence:** Mimic **human intelligence**/intelligent behaviour
 - Could be hand-crafted rules by a programmer
- **Machine Learning:** AI subfield focused on the design of algorithms capable of **inferring knowledge from data**
- **Pattern Recognition:** ML subfield that mainly focuses on **classification tasks** with **hand-crafted features**
- **Deep Learning:** ML subfield that focuses on deep neural models for both **feature extraction** and **knowledge inference**

What is Machine Learning?



Field	Goal	Representative tasks	Models
Pattern Recognition	Automaticall detect and categorize patterns	Classification	k -Nearest Neighbor Logistic Regression Gaussian Mixture Models
Machine Learning	Infer knowledge	Classification, regression, clustering, sequence labelling	Same as ML + Decision Trees, Neural models, Support Vector Machine
Deep Learning	Infer knowledge with deep models	Classification, regression, clustering, sequence labelling	Deep neural networks
Artificial Intelligence	Mimic (human) intelligence	All previous + automated planning + multi-agent systems + ...	All previous

Outline

① Definition

What is Machine Learning?

② Structure

Conceptual stages

From (conceptual) stages to (practical) scheme

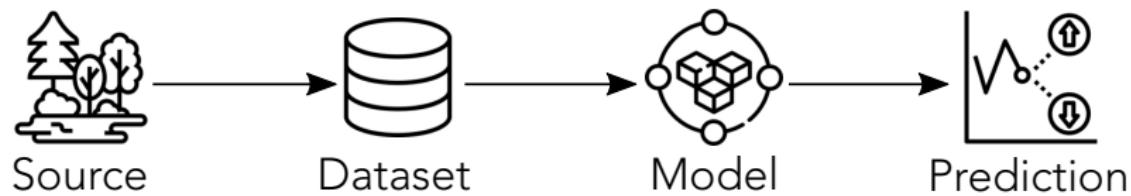
③ Taxonomies

Introduction

Categorizations

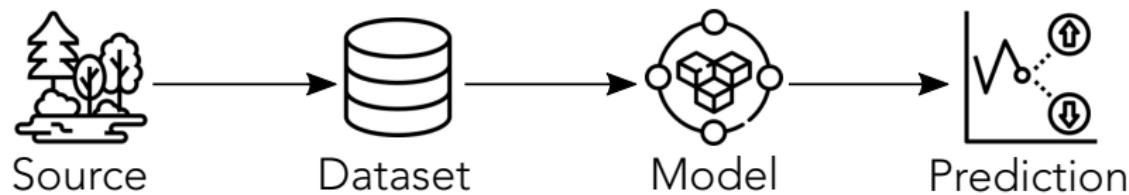
④ Areas and application

Conceptual stages



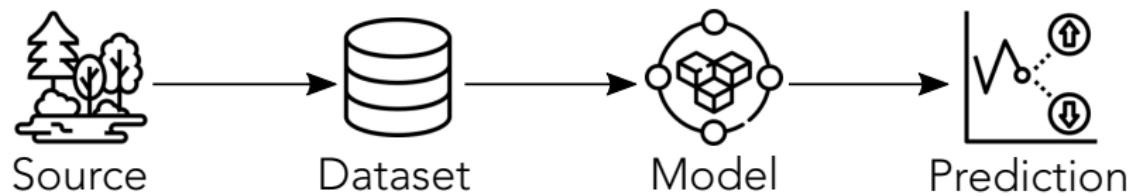
- **Source:** Data captured by sensors (photographs, recordings, scanned documents...)
 - Feature extraction: Obtaining adequate descriptors for the task at hand
- **Dataset:** Collection of elements from *Source* represented as descriptors
- **Model:** Knowledge extracted from the *Dataset* using an ML algorithm
- **Prediction:** Estimation on novel data unseen in the previous stages

Conceptual stages



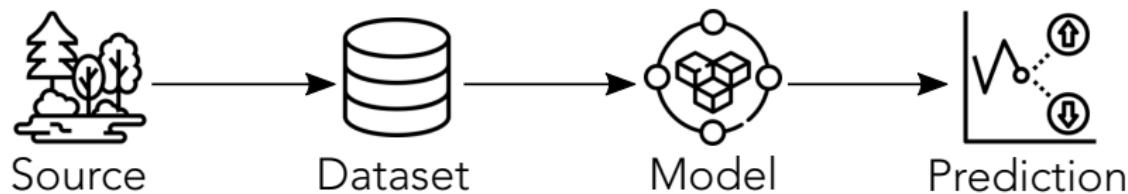
- **Source:** Data captured by *sensors* (photographs, recordings, scanned documents...)
 - **Feature extraction:** Obtaining adequate descriptors for the task at hand
- **Dataset:** Collection of elements from *Source* represented as descriptors
- **Model:** Knowledge extracted from the *Dataset* using an ML algorithm
- **Prediction:** Estimation on novel data unseen in the previous stages

Conceptual stages



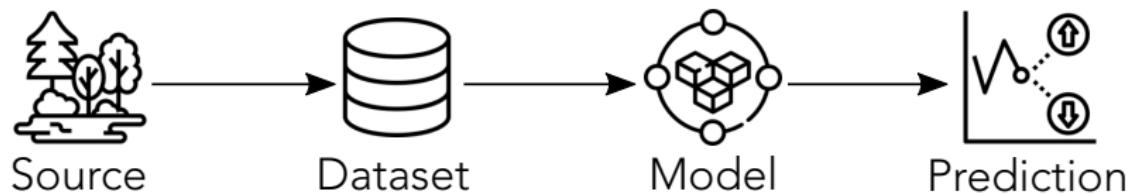
- **Source:** Data captured by **sensors** (photographs, recordings, scanned documents...)
 - **Feature extraction:** Obtaining **adequate descriptors** for the **task at hand**
- **Dataset:** Collection of elements from *Source* represented as **descriptors**
- **Model:** Knowledge extracted from the *Dataset* using an **ML algorithm**
- **Prediction:** Estimation on novel data unseen in the previous stages

Conceptual stages



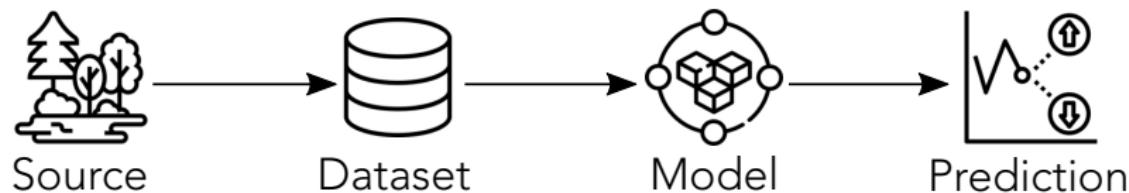
- **Source:** Data captured by **sensors** (photographs, recordings, scanned documents...)
 - **Feature extraction:** Obtaining **adequate descriptors** for the **task at hand**
- **Dataset:** **Collection of elements** from *Source* represented as **descriptors**
- **Model:** Knowledge extracted from the *Dataset* using an **ML algorithm**
- **Prediction:** Estimation on novel data unseen in the previous stages

Conceptual stages



- **Source:** Data captured by **sensors** (photographs, recordings, scanned documents...)
 - **Feature extraction:** Obtaining **adequate descriptors** for the **task at hand**
- **Dataset:** **Collection of elements** from *Source* represented as **descriptors**
- **Model:** **Knowledge extracted** from the *Dataset* using an **ML algorithm**
- **Prediction:** Estimation on novel data unseen in the previous stages

Conceptual stages



- **Source:** Data captured by **sensors** (photographs, recordings, scanned documents...)
 - **Feature extraction:** Obtaining **adequate descriptors** for the **task at hand**
- **Dataset:** **Collection of elements** from *Source* represented as **descriptors**
- **Model:** **Knowledge extracted** from the *Dataset* using an **ML algorithm**
- **Prediction:** **Estimation on novel data** unseen in the previous stages

From (conceptual) stages to (practical) scheme

ML comprises **two main processes**:

- ① **Train process:** The knowledge is extracted from **known** data
- ② **Test process:** The knowledge is exploited for **new** data

From (conceptual) stages to (practical) scheme

ML comprises **two main processes**:

- ① **Train** process: The knowledge is extracted from **known** data
- ② **Test** process: The knowledge is exploited for **new** data

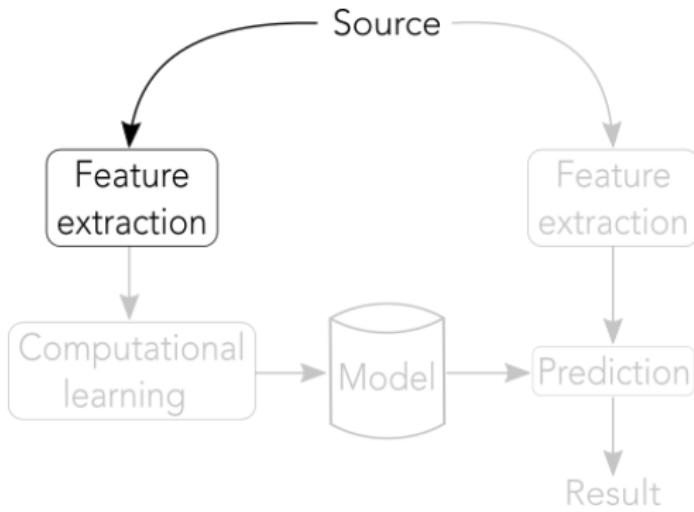
From (conceptual) stages to (practical) scheme

ML comprises **two main processes**:

- ① **Train** process: The knowledge is extracted from **known** data
- ② **Test** process: The knowledge is exploited for **new** data

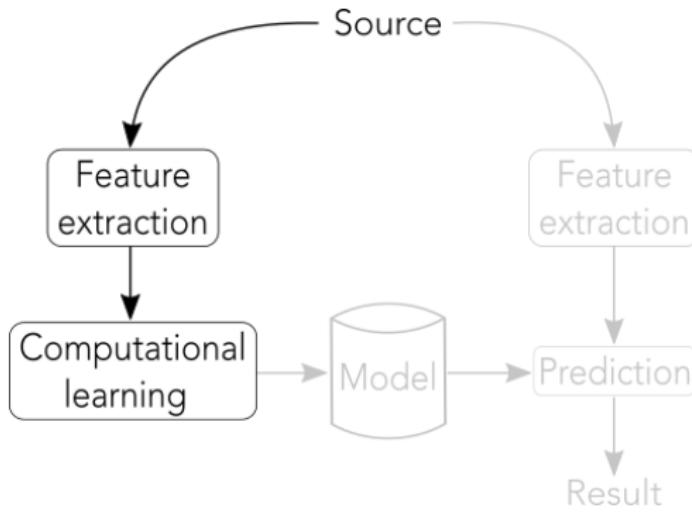
From (conceptual) stages to (practical) scheme

Train
process



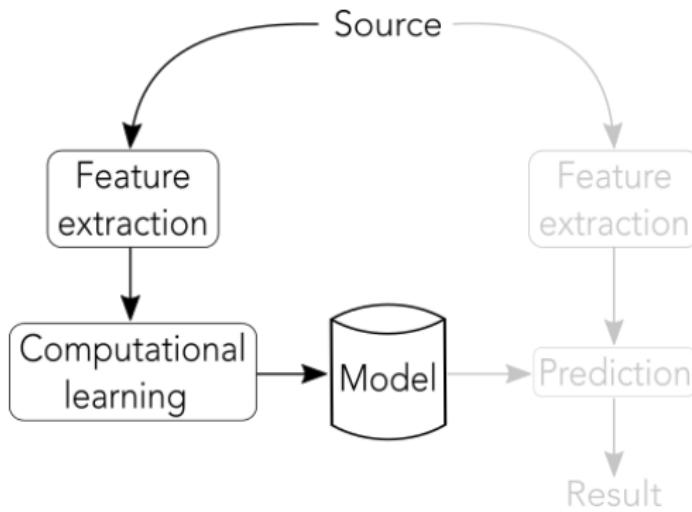
From (conceptual) stages to (practical) scheme

Train
process

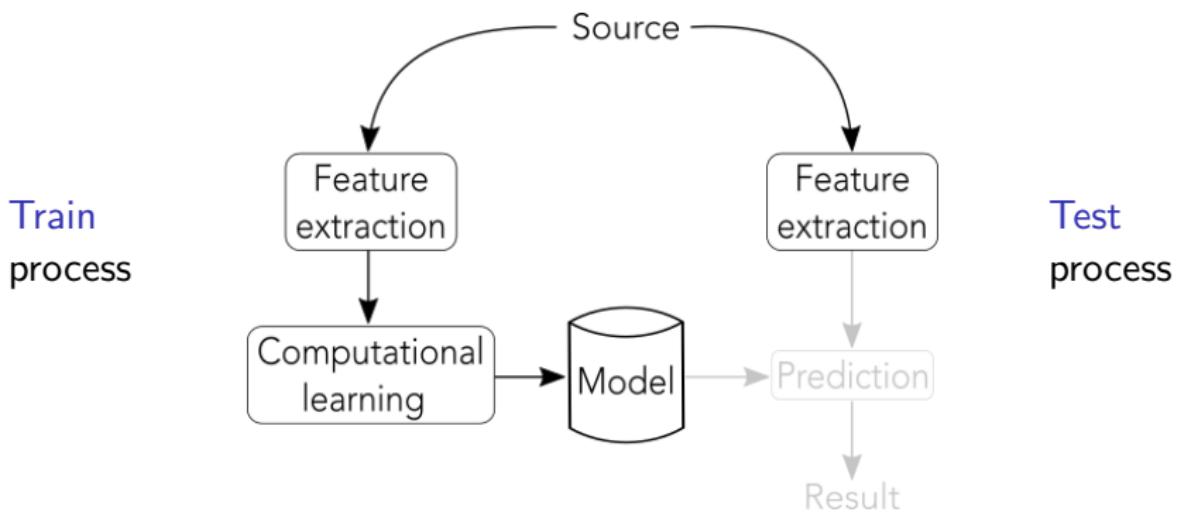


From (conceptual) stages to (practical) scheme

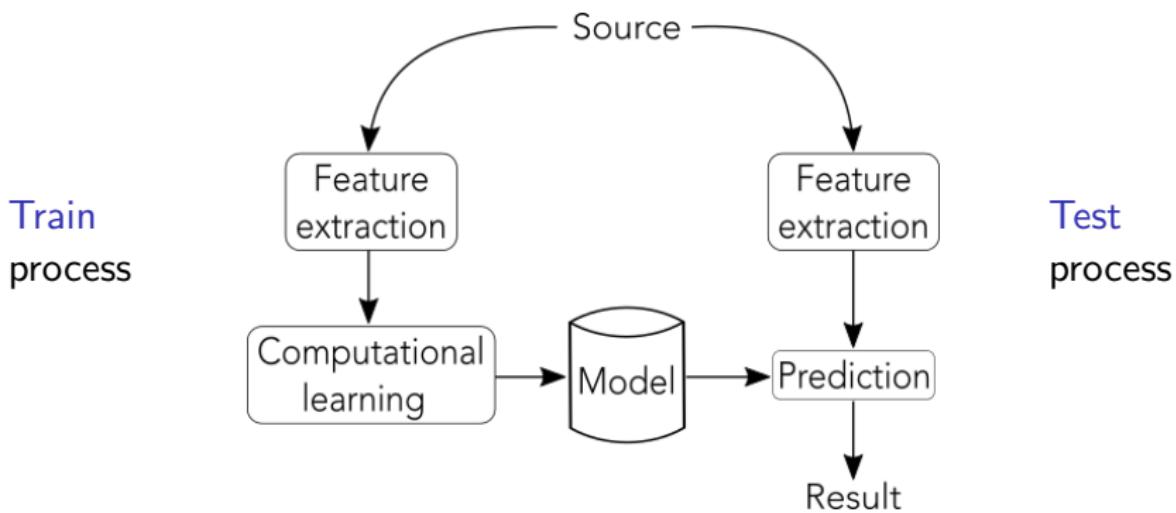
Train
process



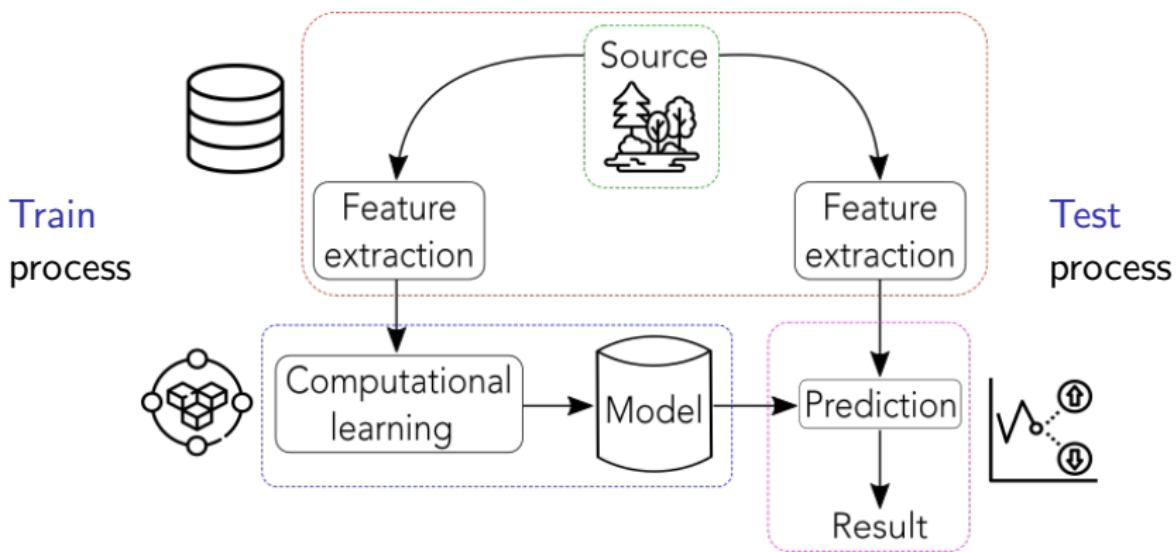
From (conceptual) stages to (practical) scheme



From (conceptual) stages to (practical) scheme



From (conceptual) stages to (practical) scheme



Feature extraction

- Encode the elements from *Source* as a set of descriptors
- The set must be adequate for the task being modeled

Feature extraction

- Encode the elements from *Source* as a set of descriptors
- The set must be adequate for the task being modeled

Feature extraction

- Encode the elements from *Source* as a set of descriptors
- The set must be adequate for the task being modeled

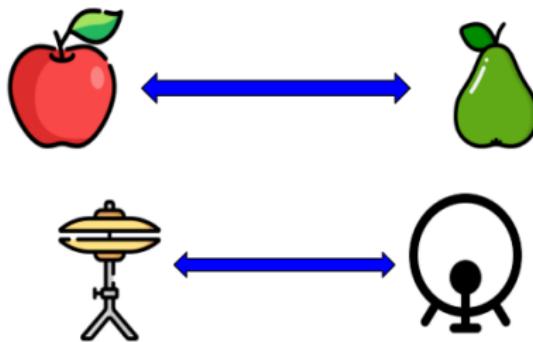
Example: How could you differentiate these two elements?



Feature extraction

- Encode the elements from *Source* as a set of descriptors
- The set must be adequate for the task being modeled

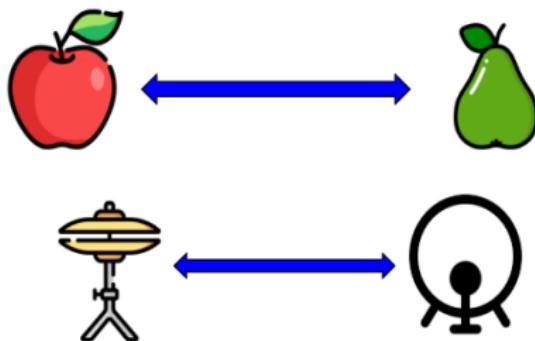
Example: How could you differentiate these two elements?



Feature extraction

- Encode the elements from *Source* as a set of descriptors
- The set must be adequate for the task being modeled

Example: How could you differentiate these two elements?



Must be derived by a computer!

Feature extraction

Representation strategies:

Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation

Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)

Feature extraction

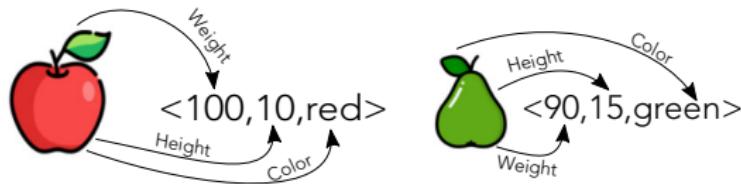
Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features

Feature extraction

Representation strategies:

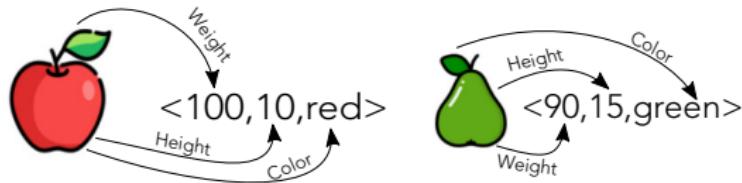
- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features



Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features

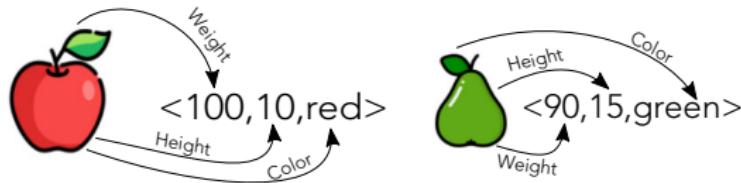


- **Structural** representation

Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features



- **Structural** representation
 - Encoding the structure (shape) of the object

Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features

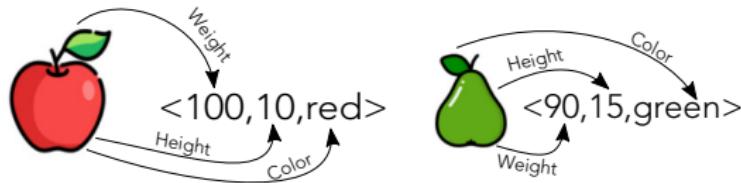


- **Structural** representation
 - Encoding the structure (shape) of the object
 - The size of the structure depends on the object

Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features

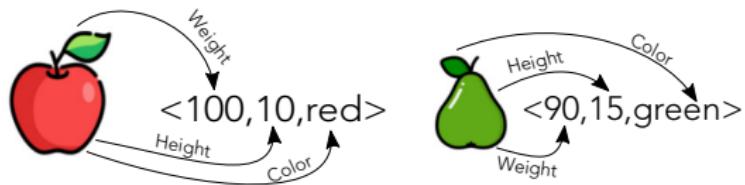


- **Structural** representation
 - Encoding the structure (shape) of the object
 - The size of the structure depends on the object
 - Strings, trees, graphs

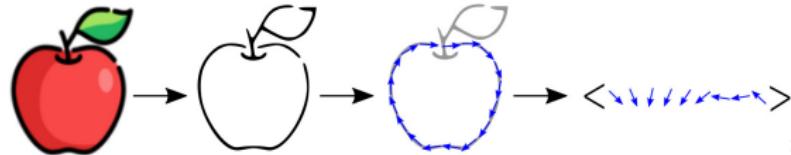
Feature extraction

Representation strategies:

- **Statistical (feature-based)** representation
 - Vector of characteristics (features)
 - All elements have the same number of features



- **Structural** representation
 - Encoding the structure (shape) of the object
 - The size of the structure depends on the object
 - Strings, trees, graphs



Feature extraction

Representation strategies:

Strategy	Representation capabilities	Flexibility
Statistical (feature-based)	Limited	Addressable by (almost) all existing algorithms
Structural	Usually achieve superior performance rates	Limited number of algorithms to process them

Computational learning

How does the model **infer knowledge** from the (extracted) representations?

Computational learning

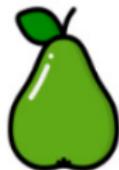
How does the model **infer knowledge** from the (extracted) representations?

Collection of **train** data

<100,10,red>
<120,12,red>
<110,9,red>
<95,11,red>



<90,15,green>
<80,17,green>
<85,16,green>
<87,15,green>



Computational learning

How does the model infer knowledge from the (extracted) representations?

Collection of train data

<100,10,red>
<120,12,red>
<110,9,red>
<95,11,red>



<90,15,green>
<80,17,green>
<85,16,green>
<87,15,green>



Rule-based approach

If "color" = "red" → 
else → 

Computational learning

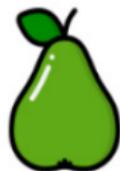
How does the model **infer knowledge** from the (extracted) representations?

Collection of train data

<100,10,red>
<120,12,red>
<110,9,red>
<95,11,red>



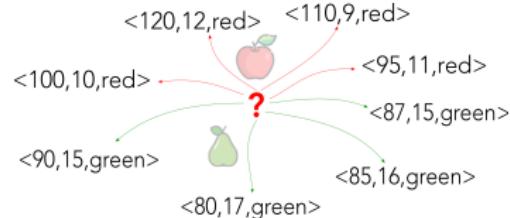
<90,15,green>
<80,17,green>
<85,16,green>
<87,15,green>



Rule-based approach

If "color" = "red" →
else →

Distance-based approach



Computational learning

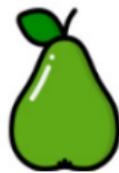
How does the model **infer knowledge** from the (extracted) representations?

Collection of train data

<100,10,red>
 <120,12,red>
 <110,9,red>
 <95,11,red>



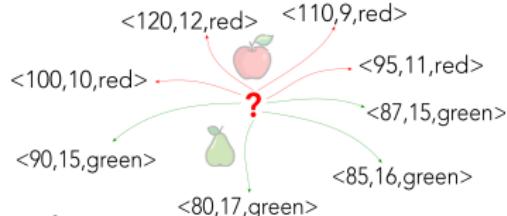
<90,15,green>
 <80,17,green>
 <85,16,green>
 <87,15,green>



Rule-based approach

If "color" = "red" →
 else →

Distance-based approach



Probabilistic approach

$P(?, \text{red}) > P(?, \text{green}) \rightarrow \text{apple}$
 $P(?, \text{red}) < P(?, \text{green}) \rightarrow \text{pear}$

Model

What is **the model** in each case?

Model

What is **the model** in each case?

- Distance-based approach: Collection of samples stored in a database

Model

What is **the model** in each case?

- Distance-based approach: Collection of samples stored in a database
- Rule-based approach: Set of rules

Model

What is **the model** in each case?

- Distance-based approach: Collection of samples stored in a database
- Rule-based approach: Set of rules
- Probabilistic approach: Parameters of a probabilistic distribution

T2: Computational learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Introduction

Where are we?

Computational learning VS Decision theory

② Bayesian decision theory

Two-class problem

General form

Risk

Discriminant functions

③ Statistical likelihood

Maximum likelihood estimation

④ Issues in computational learning

Bias-variance issues

Curse of dimensionality

Outline

① Introduction

Where are we?

Computational learning VS Decision theory

② Bayesian decision theory

Two-class problem

General form

Risk

Discriminant functions

③ Statistical likelihood

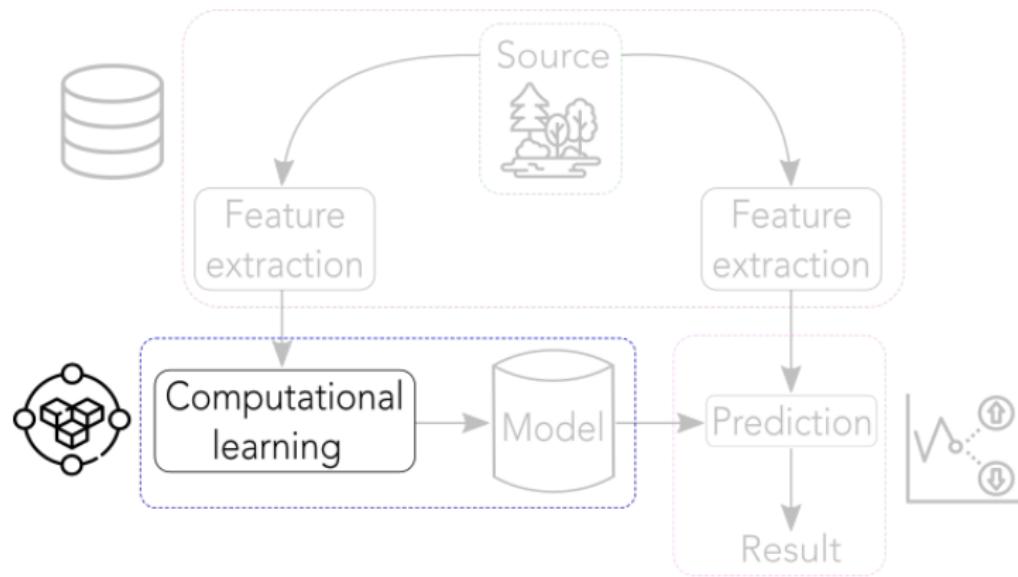
Maximum likelihood estimation

④ Issues in computational learning

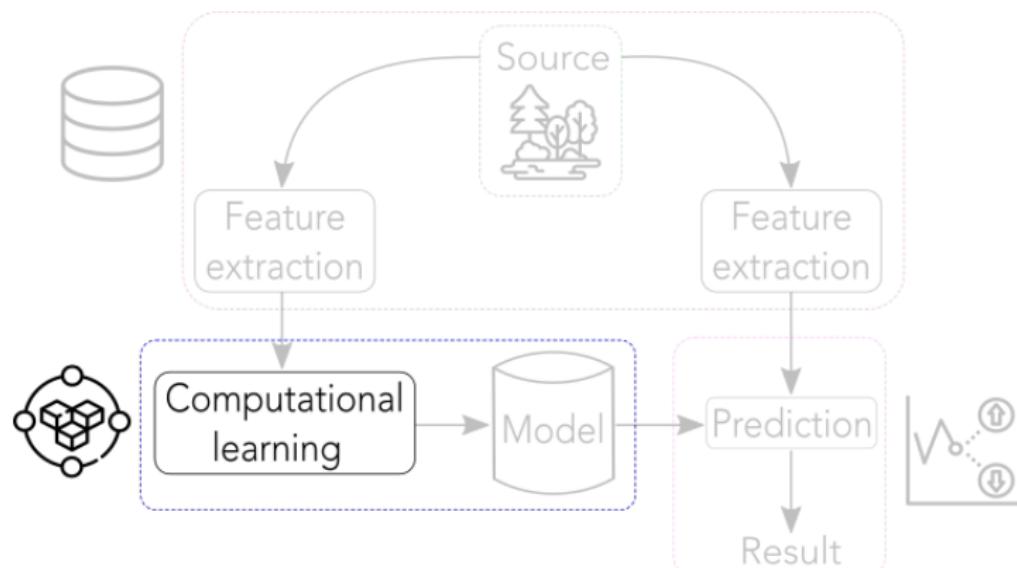
Bias-variance issues

Curse of dimensionality

Where are we?

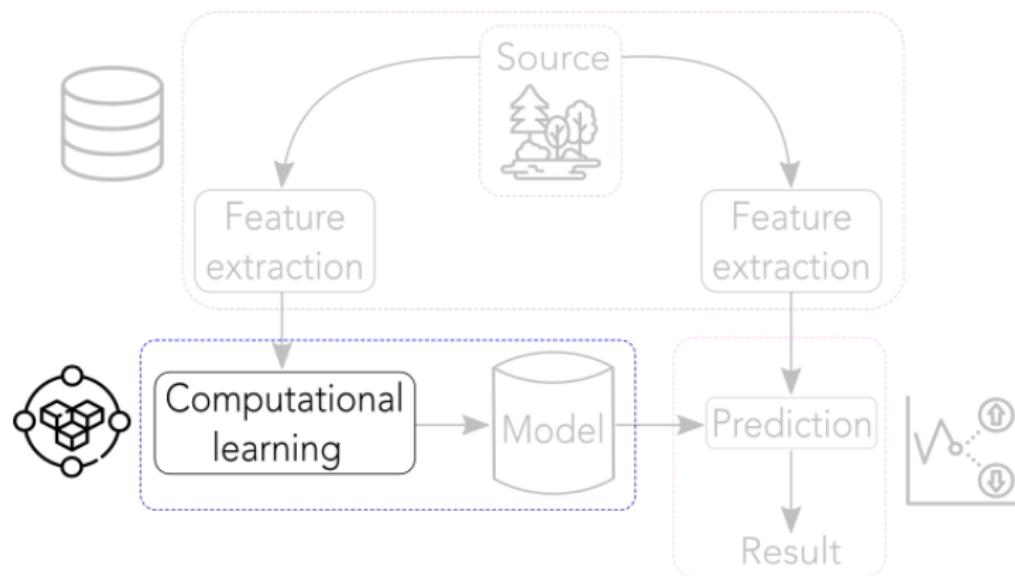


Where are we?



Computational learning:

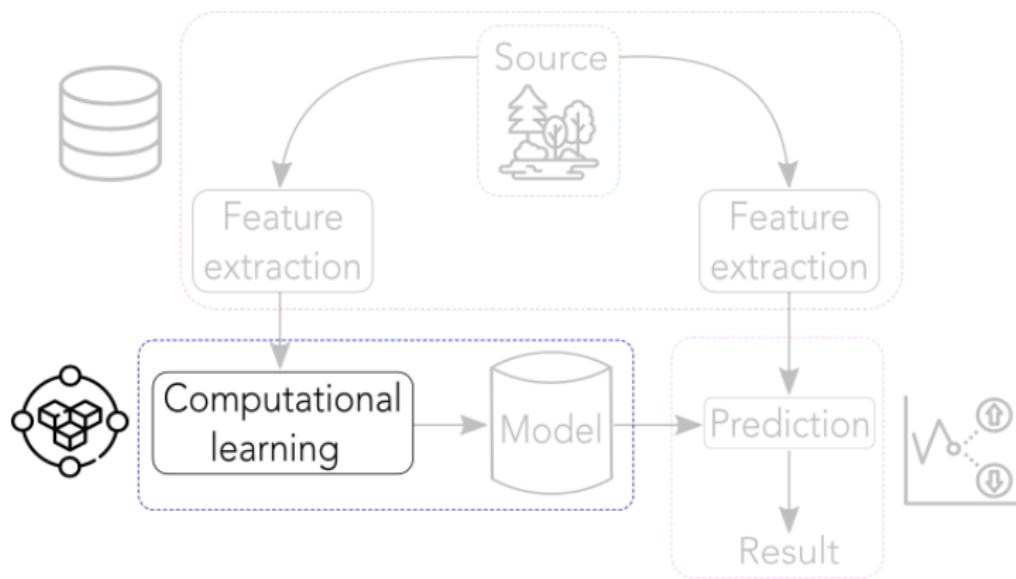
Where are we?



Computational learning:

- Infer knowledge from data

Where are we?



Computational learning:

- Infer knowledge from data
- Derive a model out of that knowledge

Computational learning VS Decision theory

Computational learning VS Decision theory

Two related, but different, concepts:

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data
- **Decision theory:** Make optimal decisions under uncertainty

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data
 - What can be learned and how efficiently
 - Also statistical learning theory
- **Decision theory:** Make optimal decisions under uncertainty
 - Foundational mathematical framework for Machine Learning
 - Pattern classification is the most important subfield

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data
 - What can be learned and how efficiently
 - Also statistical learning theory
- **Decision theory:** Make optimal decisions under uncertainty
 - Foundational mathematical framework for Machine Learning
 - Pattern classification is the most important subfield

Bayesian decision theory (for classification):

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data
 - What can be learned and how efficiently
 - Also statistical learning theory
- **Decision theory:** Make optimal decisions under uncertainty
 - Foundational mathematical framework for Machine Learning
 - Pattern classification is the most important subfield

Bayesian decision theory (for classification):

- Fundamental statistical approach for the pattern classification problem

Computational learning VS Decision theory

Two related, but different, concepts:

- **Computational learning:** Studies the limits of learning from data
 - What can be learned and how efficiently
 - Also statistical learning theory
- **Decision theory:** Make optimal decisions under uncertainty
 - Foundational mathematical framework for Machine Learning
 - Pattern classification is the most important subfield

Bayesian decision theory (for classification):

- Fundamental statistical approach for the pattern classification problem
- Quantifying tradeoffs between various classification decisions using probability and the cost of such decisions

Outline

① Introduction

Where are we?

Computational learning VS Decision theory

② Bayesian decision theory

Two-class problem

General form

Risk

Discriminant functions

③ Statistical likelihood

Maximum likelihood estimation

④ Issues in computational learning

Bias-variance issues

Curse of dimensionality

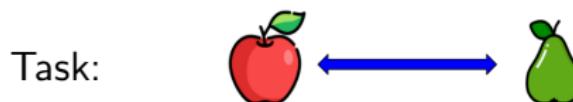
Introduction

Introduction

Task:



Introduction



- **State of nature (w):** future event **not under the control** of the decision maker

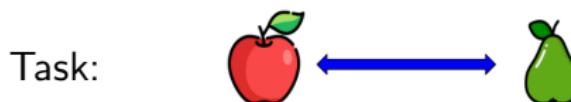
Introduction

Task:



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)

Introduction



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)
- We assume the existence of some *a priori probability/prior*:

Introduction

Task:



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)
- We assume the existence of some *a priori probability/prior*:
 - $P(\omega_1)$ that the element is an apple (ω_1) and $P(\omega_2)$ that the element is a pear (ω_2)

Introduction

Task:



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)
- We assume the existence of some *a priori probability/prior*:
 - $P(\omega_1)$ that the element is an apple (ω_1) and $P(\omega_2)$ that the element is a pear (ω_2)
 - Based on geographical region, season...

Introduction

Task:



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)
- We assume the existence of some *a priori probability/prior*:
 - $P(\omega_1)$ that the element is an apple (ω_1) and $P(\omega_2)$ that the element is a pear (ω_2)
 - Based on geographical region, season...
- **Decision rule:**

$$w = \begin{cases} w_1 & \text{if } P(\omega_1) > P(\omega_2) \\ w_2 & \text{otherwise} \end{cases}$$

Introduction

Task:



- **State of nature (w):** future event **not under the control** of the decision maker
 - In our case two classes: apple (ω_1) and pear (ω_2)
- We assume the existence of some *a priori probability/prior*:
 - $P(\omega_1)$ that the element is an apple (ω_1) and $P(\omega_2)$ that the element is a pear (ω_2)
 - Based on geographical region, season...
- **Decision rule:**
$$w = \begin{cases} \omega_1 & \text{if } P(\omega_1) > P(\omega_2) \\ \omega_2 & \text{otherwise} \end{cases}$$
- **Additional information** may be incorporated!

Class-conditional probability density

- Different fruits will yield different color readings

Class-conditional probability density

- Different fruits will yield different color readings
 - Single descriptor/feature to encode color $\Rightarrow x \in \mathbb{R}$

Class-conditional probability density

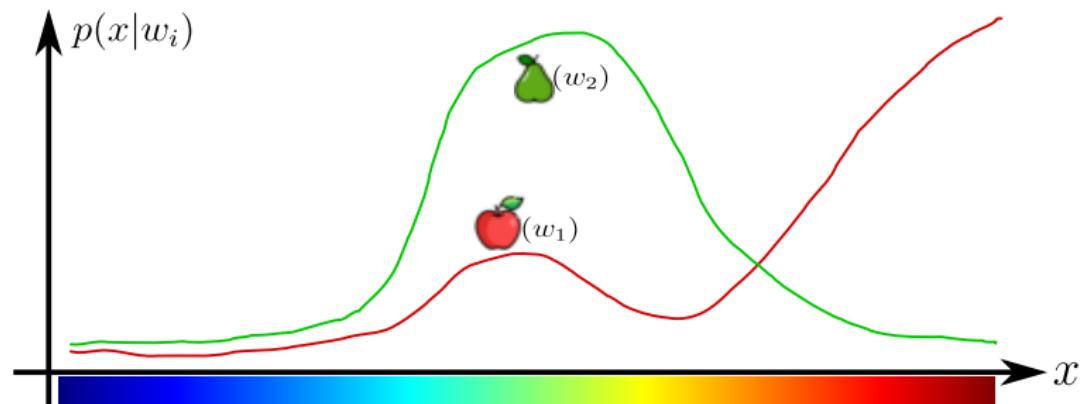
- Different fruits will yield different color readings
 - Single descriptor/feature to encode color $\Rightarrow x \in \mathbb{R}$
- Class-conditional probability density: $p(x|\omega)$

Class-conditional probability density

- Different fruits will yield different color readings
 - Single descriptor/feature to encode color $\Rightarrow x \in \mathbb{R}$
- Class-conditional probability density: $p(x|\omega)$
 - How likely is it to observe color x on fruit ω ?

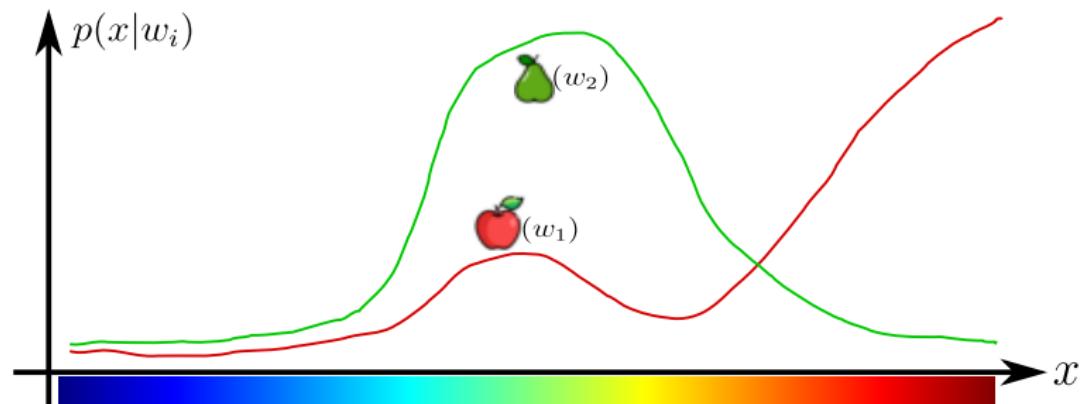
Class-conditional probability density

- Different fruits will yield different color readings
 - Single descriptor/feature to encode color $\Rightarrow x \in \mathbb{R}$
- Class-conditional probability density: $p(x|\omega)$
 - How likely is it to observe color x on fruit ω ?



Class-conditional probability density

- Different fruits will yield different color readings
 - Single descriptor/feature to encode color $\Rightarrow x \in \mathbb{R}$
- Class-conditional probability density: $p(x|\omega)$
 - How likely is it to observe color x on fruit ω ?



$$\int p(x|\omega_i) dx = 1 \quad \forall i \in \{1, 2\}$$

Joint probability density

- Likelihood for simultaneous occurrence of different variables

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- ✗ If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$
- ✓ If statistically dependent:

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- ✗ If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$
- ✓ If statistically dependent:

$$p(\omega_j, x)$$

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- ✗ If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$
- ✓ If statistically dependent:

$$p(\omega_j, x) = P(\omega_j|x) \cdot p(x)$$

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- ✗ If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$
- ✓ If statistically dependent:

$$p(\omega_j, x) = P(\omega_j|x) \cdot p(x) = p(x|\omega_j) \cdot P(\omega_j)$$

Joint probability density

- Likelihood for simultaneous occurrence of different variables
 - In our case $\Rightarrow P(\omega_j, x)$
- ✗ If statistically independent: $p(\omega_j, x) = P(\omega_j) \cdot p(x)$
- ✓ If statistically dependent:

$$p(\omega_j, x) = P(\omega_j|x) \cdot p(x) = p(x|\omega_j) \cdot P(\omega_j)$$



$$p(\omega_j, x) = P(\omega_j|x) \cdot p(x) = p(x|\omega_j) \cdot P(\omega_j)$$

Bayes theorem

Bayes theorem

$$P(\omega_j|x) = \frac{p(x|\omega_j) \cdot P(\omega_j)}{p(x)}$$

Bayes theorem

Bayes theorem

$$\underbrace{P(\omega_j|x)}_{posterior} = \frac{\overbrace{p(x|\omega_j) \cdot P(\omega_j)}^{likelihood \cdot prior}}{\underbrace{p(x)}_{evidence}}$$

Bayes theorem

Bayes theorem

$$\underbrace{P(\omega_j|x)}_{posterior} = \frac{\overbrace{p(x|\omega_j) \cdot P(\omega_j)}^{likelihood \cdot prior}}{\underbrace{p(x)}_{evidence}}$$

- **Posterior:** Observing state of nature w_j given feature value x

Bayes theorem

Bayes theorem

$$\underbrace{P(\omega_j|x)}_{posterior} = \frac{\overbrace{p(x|\omega_j) \cdot P(\omega_j)}^{likelihood \cdot prior}}{\underbrace{p(x)}_{evidence}}$$

- **Posterior:** Observing state of nature w_j given feature value x
- **Likelihood:** Observing feature value x assuming that state of nature w_j is correct

Bayes theorem

Bayes theorem

$$\underbrace{P(\omega_j|x)}_{\text{posterior}} = \frac{\overbrace{p(x|\omega_j) \cdot P(\omega_j)}^{\text{likelihood prior}}}{\underbrace{p(x)}_{\text{evidence}}}$$

- **Posterior:** Observing state of nature ω_j given feature value x
- **Likelihood:** Observing feature value x assuming that state of nature ω_j is correct
- **Prior:** Initial probability of observing state of nature ω_j without further evidence (existing knowledge, historical data...)

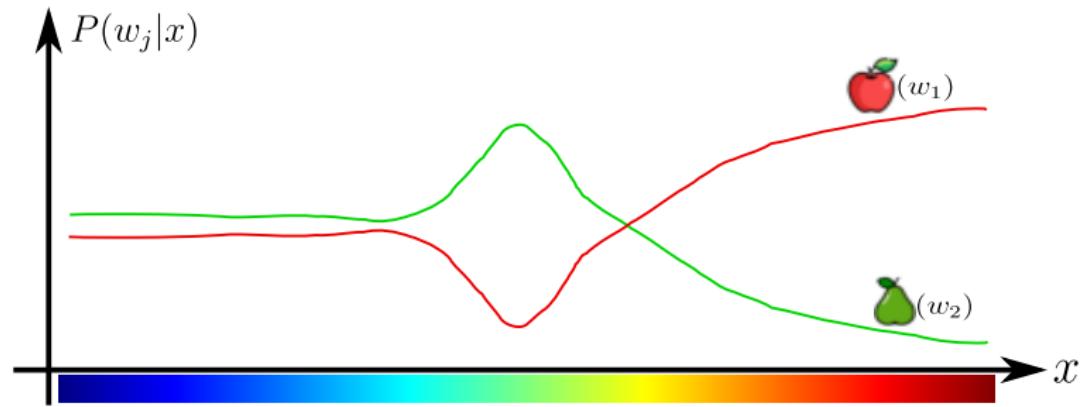
Bayes theorem

Bayes theorem

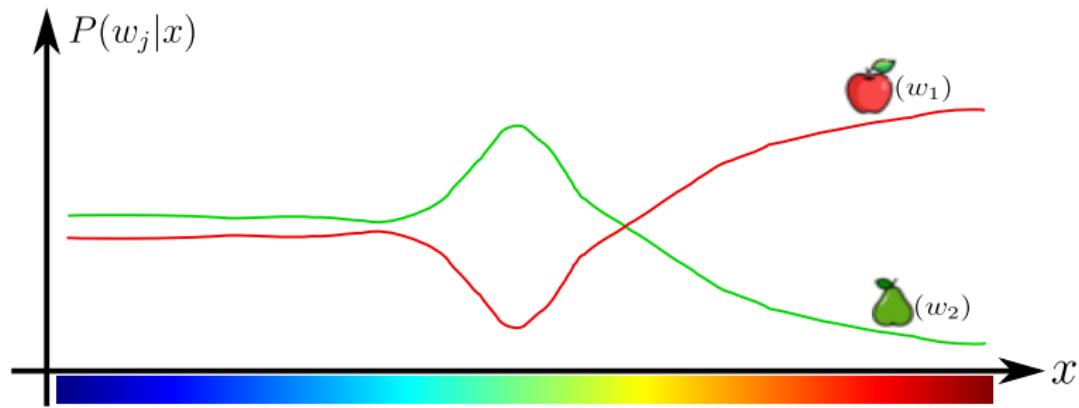
$$\underbrace{P(\omega_j|x)}_{\text{posterior}} = \frac{\overbrace{p(x|\omega_j) \cdot P(\omega_j)}^{\text{likelihood prior}}}{\underbrace{p(x)}_{\text{evidence}}}$$

- **Posterior:** Observing state of nature w_j given feature value x
- **Likelihood:** Observing feature value x assuming that state of nature w_j is correct
- **Prior:** Initial probability of observing state of nature w_j without further evidence (existing knowledge, historical data...)
- **Evidence:** Scale factor $\Rightarrow p(x) = \sum_{\forall j} p(x|\omega_j) \cdot P(\omega_j)$

Posterior probability



Posterior probability



For any color $x = x_0 \longrightarrow P(\omega_1|x_0) + P(\omega_2|x_0) = 1$

Practical exercise

Estimate the probability of obtaining a particular fruit (state of nature) given *priors* and its color

1. Historically, the harvest of apples represents the 70% of the fruit, whereas that of pears is the 30%
2. The distribution of color for each fruit is:

Color ([0, 100])	Blue (0)	Green (40)	Yellow (50)	Orange (70)	Red (100)
$p(x w_1)$	0.05	0.25	0.15	0.1	0.45
$p(x w_2)$	0.05	0.6	0.25	0.05	0.05

- Q1. How likely is it to obtain a yellow apple? And a same-colored pear?
- Q2. Obtain all posterior probabilities for all possible color and fruit combinations

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
→ w_1 if $P(w_1|x_0) > P(w_2|x_0)$

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
 - w_1 if $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
 - w_1 if $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise
- Is this **correct**?

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
 - w_1 if $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise
- Is this **correct**? **Probability of error** when performing this action:

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
 - w_1 if $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise
- Is this **correct**? **Probability of error** when performing this action:

$$P(\text{error}|x_0) = \begin{cases} P(w_1|x_0) & \text{if we decide } w_2 \\ P(w_2|x_0) & \text{if we decide } w_1 \end{cases}$$

Probability of error

- Given an observation $x = x_0$, one would **select** as **state of nature**:
 - w_1 if $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise
- Is this **correct**? **Probability of error** when performing this action:

$$P(\text{error}|x_0) = \begin{cases} P(w_1|x_0) & \text{if we decide } w_2 \\ P(w_2|x_0) & \text{if we decide } w_1 \end{cases}$$

- We can **minimize** this expression by selecting:
 - w_1 when $P(w_1|x_0) > P(w_2|x_0)$
 - w_2 otherwise

Probability of error

- Will this rule **minimize** the average **probability of error**?

$$P(\text{error}) = \int_{-\infty}^{\infty} P(\text{error}, x) dx = \int_{-\infty}^{\infty} P(\text{error}|x) \cdot p(x) dx$$

Probability of error

- Will this rule **minimize** the average **probability of error**?

$$P(\text{error}) = \int_{-\infty}^{\infty} P(\text{error}, x) dx = \int_{-\infty}^{\infty} P(\text{error}|x) \cdot p(x) dx$$

- **Minimize** $P(\text{error}) \Rightarrow$ ensure $P(\text{error}|x)$ is as small as possible $\forall x$

Probability of error

- Will this rule **minimize** the average **probability of error**?

$$P(\text{error}) = \int_{-\infty}^{\infty} P(\text{error}, x) dx = \int_{-\infty}^{\infty} P(\text{error}|x) \cdot p(x) dx$$

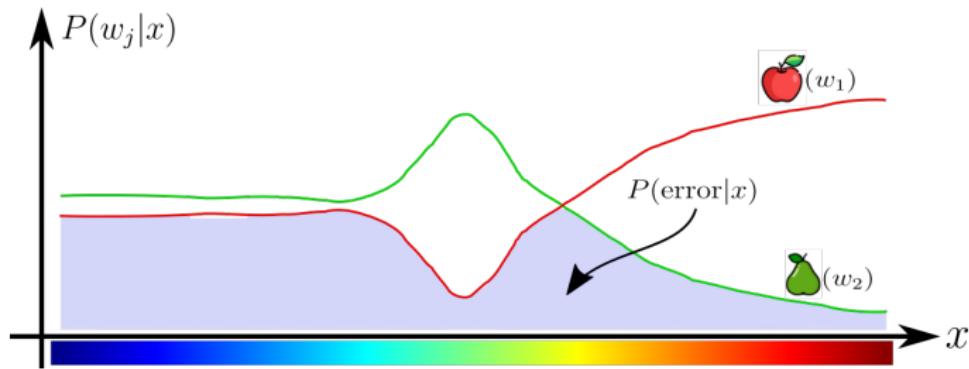
- **Minimize** $P(\text{error}) \Rightarrow$ ensure $P(\text{error}|x)$ is as small as possible $\forall x$
 $\rightarrow P(\text{error}|x) = \min[P(w_1|x), P(w_2|x)]$

Probability of error

- Will this rule **minimize** the average **probability of error**?

$$P(\text{error}) = \int_{-\infty}^{\infty} P(\text{error}, x) dx = \int_{-\infty}^{\infty} P(\text{error}|x) \cdot p(x) dx$$

- **Minimize** $P(\text{error}) \Rightarrow$ ensure $P(\text{error}|x)$ is as small as possible $\forall x$
 $\rightarrow P(\text{error}|x) = \min[P(w_1|x), P(w_2|x)]$



Bayes decision rule

Bayes decision rule

- Optimal decision rule \Rightarrow minimizes probability of error

Decide w_1 if $P(w_1|x) > P(w_2|x)$; otherwise decide w_2

Bayes decision rule

- Optimal decision rule \Rightarrow minimizes probability of error

Decide w_1 if $P(w_1|x) > P(w_2|x)$; otherwise decide w_2

- Evidence is unimportant as far as making a decision is concerned

Bayes decision rule

- Optimal decision rule \Rightarrow minimizes probability of error

Decide w_1 if $P(w_1|x) > P(w_2|x)$; otherwise decide w_2

- Evidence is unimportant as far as making a decision is concerned
 \rightarrow It may be obviated

Bayes decision rule

- Optimal decision rule \Rightarrow minimizes probability of error

Decide w_1 if $P(w_1|x) > P(w_2|x)$; otherwise decide w_2

- Evidence is unimportant as far as making a decision is concerned
→ It may be obviated

Bayes decision rule for two-class classification

Decide w_1 if $p(x|w_1) \cdot P(w_1) > p(x|w_2) \cdot P(w_2)$;
otherwise decide w_2

Bayes decision rule

- Optimal decision rule \Rightarrow minimizes probability of error

Decide w_1 if $P(w_1|x) > P(w_2|x)$; otherwise decide w_2

- Evidence is unimportant as far as making a decision is concerned
→ It may be obviated

Bayes decision rule for two-class classification

Decide w_1 if $p(x|w_1) \cdot P(w_1) > p(x|w_2) \cdot P(w_2)$;
otherwise decide w_2

- The rule may be also written as:

$$w = \arg \max_{w \in \{w_1, w_2\}} P(w|x) = \arg \max_{w \in \{w_1, w_2\}} p(x|w) \cdot P(w)$$

Practical exercise

Considering the same exercise as before where

1. Historically, the harvest of apples represents the 70% of the fruit, whereas that of pears is the 30%
2. The distribution of color for each fruit is:

Color ([0, 100])	Blue (0)	Green (40)	Yellow (50)	Orange (70)	Red (100)
$p(x w_1)$	0.05	0.25	0.15	0.1	0.45
$p(x w_2)$	0.05	0.6	0.25	0.05	0.05

- Q1.** Which is the minimum error in each case considering the Bayes decision rule?
- Q2.** What would occur if priors were equiprobable?

$$P(w_1) = P(w_2) = 50\%$$

Definitions

We shall **generalize** these ideas if four different aspects:

Definitions

We shall **generalize** these ideas if four different aspects:

- 1) More than one **feature**

Definitions

We shall **generalize** these ideas if four different aspects:

- 1) More than one **feature**
- 2) More than two **states of nature**

Definitions

We shall **generalize** these ideas if four different aspects:

- 1) More than one **feature**
- 2) More than two **states of nature**
- 3) **Actions** beyond selecting labels

Definitions

We shall **generalize** these ideas in four different aspects:

- 1) More than one **feature**
- 2) More than two **states of nature**
- 3) **Actions** beyond selecting labels
- 4) **Loss function** as a generalization of the probability of error

Definitions

Definitions

- Feature space: use more than a single descriptor

Definitions

- Feature space: use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$
- **Actions:** possibilitate other beyond classification

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$
- **Actions:** possibilities other beyond classification
 - Finite set of actions: $\mathcal{A} = \{\alpha_1, \dots, \alpha_{|\mathcal{A}|}\}$ s.t. $\alpha_i : \mathbb{R}^d \rightarrow \mathcal{W}$

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$
- **Actions:** possibilities other beyond classification
 - Finite set of actions: $\mathcal{A} = \{\alpha_1, \dots, \alpha_{|\mathcal{A}|}\}$ s.t. $\alpha_i : \mathbb{R}^d \rightarrow \mathcal{W}$
- **Loss function:** Generalization of probability of error \Rightarrow errors are **not equally important**

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$
- **Actions:** possibilities other beyond classification
 - Finite set of actions: $\mathcal{A} = \{\alpha_1, \dots, \alpha_{|\mathcal{A}|}\}$ s.t. $\alpha_i : \mathbb{R}^d \rightarrow \mathcal{W}$
- **Loss function:** Generalization of probability of error \Rightarrow errors are **not equally important**
 - Cost of performing α_i when true state is $w_j \Rightarrow \lambda(\alpha_i|w_j)$

Definitions

- **Feature space:** use more than a single descriptor
 - Fixed d -dimensional space: $\mathbf{x} \in \mathbb{R}^d$
- **States of nature:** allow more than two states
 - Finite set of states: $\mathcal{W} = \{w_1, \dots, w_{|\mathcal{W}|}\}$
- **Actions:** possibilities other beyond classification
 - Finite set of actions: $\mathcal{A} = \{\alpha_1, \dots, \alpha_{|\mathcal{A}|}\}$ s.t. $\alpha_i : \mathbb{R}^d \rightarrow \mathcal{W}$
- **Loss function:** Generalization of probability of error \Rightarrow errors are **not equally important**
 - Cost of performing α_i when true state is $w_j \Rightarrow \lambda(\alpha_i|w_j)$

Bayes theorem (general form)

$$P(w_j|\mathbf{x}) = \frac{p(\mathbf{x}|w_j) \cdot P(w_j)}{p(\mathbf{x})} \quad \text{with} \quad p(\mathbf{x}) = \sum_{j=1}^{|\mathcal{W}|} p(\mathbf{x}|w_j) \cdot P(w_j)$$

Risk

Risk

Conditional risk

Expected loss associated to each action $\alpha_i \in \mathcal{A}$ given \mathbf{x} :

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{|\mathcal{W}|} \lambda(\alpha_i|w_j) \cdot P(w_j|\mathbf{x})$$

*If I see feature vector \mathbf{x} , what is my expected loss if I decide α_i ?
How risky is this α_i at this particular \mathbf{x}*

Risk

Conditional risk

Expected loss associated to each action $\alpha_i \in \mathcal{A}$ given \mathbf{x} :

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{|\mathcal{W}|} \lambda(\alpha_i|w_j) \cdot P(w_j|\mathbf{x})$$

If I see feature vector \mathbf{x} , what is my expected loss if I decide α_i ?
How risky is this α_i at this particular \mathbf{x}

Risk (or loss)

Average over all possible observations of conditional risk

$$R = \int_{\mathbb{R}^d} R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

where decision rule $\gamma : \mathbb{R}^d \rightarrow \mathcal{A}$ selects the appropriate action to take

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

- If $\gamma(\mathbf{x})$ is chosen so that $R(\gamma(\mathbf{x})|\mathbf{x})$ is as small as possible for every \mathbf{x}

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

- If $\gamma(\mathbf{x})$ is chosen so that $R(\gamma(\mathbf{x})|\mathbf{x})$ is as small as possible for every \mathbf{x}
→ Overall risk will be minimized

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

- If $\gamma(\mathbf{x})$ is chosen so that $R(\gamma(\mathbf{x})|\mathbf{x})$ is as small as possible for every \mathbf{x}
→ Overall risk will be minimized
- Bayes decision rule: Select the action $\alpha_i \in \mathcal{A}$ that minimizes $R(\alpha_i|\mathbf{x})$

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

- If $\gamma(\mathbf{x})$ is chosen so that $R(\gamma(\mathbf{x})|\mathbf{x})$ is as small as possible for every \mathbf{x}
→ Overall risk will be minimized
- Bayes decision rule: Select the action $\alpha_i \in \mathcal{A}$ that minimizes $R(\alpha_i|\mathbf{x})$

Minimum-risk Bayes rule

$$\gamma(\mathbf{x}) = \arg \min_{\alpha_i \in \mathcal{A}} R(\alpha_i|\mathbf{x})$$

Minimum-risk Bayes rule

$$R = \int R(\gamma(\mathbf{x})|\mathbf{x}) \cdot p(\mathbf{x}) d\mathbf{x}$$

- If $\gamma(\mathbf{x})$ is chosen so that $R(\gamma(\mathbf{x})|\mathbf{x})$ is as small as possible for every \mathbf{x}
→ Overall risk will be minimized
- Bayes decision rule: Select the action $\alpha_i \in \mathcal{A}$ that minimizes $R(\alpha_i|\mathbf{x})$

Minimum-risk Bayes rule

$$\gamma(\mathbf{x}) = \arg \min_{\alpha_i \in \mathcal{A}} R(\alpha_i|\mathbf{x})$$

- Bayes risk: Best performance that can be achieved
→ Usually denoted as R^*

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow R(\alpha_1|\mathbf{x}) < R(\alpha_2|\mathbf{x})$$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow \overbrace{\lambda_{11} \cdot P(\omega_1|\mathbf{x}) + \lambda_{12} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_1|\mathbf{x})} < \overbrace{\lambda_{21} \cdot P(\omega_1|\mathbf{x}) + \lambda_{22} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_2|\mathbf{x})}$$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow \overbrace{\lambda_{11} \cdot P(\omega_1|\mathbf{x}) + \lambda_{12} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_1|\mathbf{x})} < \overbrace{\lambda_{21} \cdot P(\omega_1|\mathbf{x}) + \lambda_{22} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_2|\mathbf{x})}$$
$$\Downarrow P(\omega|\mathbf{x}) \propto p(\mathbf{x}|\omega) \cdot P(\omega)$$
$$(\lambda_{21} - \lambda_{11}) \cdot p(\mathbf{x}|\omega_1) \cdot P(\omega_1) > (\lambda_{12} - \lambda_{22}) \cdot p(\mathbf{x}|\omega_2) \cdot P(\omega_2)$$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow \overbrace{\lambda_{11} \cdot P(\omega_1|\mathbf{x}) + \lambda_{12} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_1|\mathbf{x})} < \overbrace{\lambda_{21} \cdot P(\omega_1|\mathbf{x}) + \lambda_{22} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_2|\mathbf{x})}$$

$$\Downarrow P(\omega|\mathbf{x}) \propto p(\mathbf{x}|\omega) \cdot P(\omega)$$

$$(\lambda_{21} - \lambda_{11}) \cdot p(\mathbf{x}|\omega_1) \cdot P(\omega_1) > (\lambda_{12} - \lambda_{22}) \cdot p(\mathbf{x}|\omega_2) \cdot P(\omega_2)$$

$$\Downarrow$$

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow \overbrace{\lambda_{11} \cdot P(\omega_1|\mathbf{x}) + \lambda_{12} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_1|\mathbf{x})} < \overbrace{\lambda_{21} \cdot P(\omega_1|\mathbf{x}) + \lambda_{22} \cdot P(\omega_2|\mathbf{x})}^{R(\alpha_2|\mathbf{x})}$$

$$\Downarrow P(\omega|\mathbf{x}) \propto p(\mathbf{x}|\omega) \cdot P(\omega)$$

$$(\lambda_{21} - \lambda_{11}) \cdot p(\mathbf{x}|\omega_1) \cdot P(\omega_1) > (\lambda_{12} - \lambda_{22}) \cdot p(\mathbf{x}|\omega_2) \cdot P(\omega_2)$$

$$\Downarrow$$

Likelihood ratio $\Rightarrow \frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)} = \theta$

Classification: the two-category case

- $\mathcal{A} = \{\alpha_1, \alpha_2\}$
- $\mathcal{W} = \{\omega_1, \omega_2\}$
- $\lambda(\alpha_i|\omega_j) = \lambda_{ij}$ with $i, j \in \{1, 2\}$

$$\omega_1 \Rightarrow \overbrace{\lambda_{11} \cdot P(\omega_1|x) + \lambda_{12} \cdot P(\omega_2|x)}^{R(\alpha_1|x)} < \overbrace{\lambda_{21} \cdot P(\omega_1|x) + \lambda_{22} \cdot P(\omega_2|x)}^{R(\alpha_2|x)}$$

$$\Downarrow P(\omega|x) \propto p(x|\omega) \cdot P(\omega)$$

$$(\lambda_{21} - \lambda_{11}) \cdot p(x|\omega_1) \cdot P(\omega_1) > (\lambda_{12} - \lambda_{22}) \cdot p(x|\omega_2) \cdot P(\omega_2)$$

$$\Downarrow$$

Likelihood ratio $\Rightarrow \frac{p(x|\omega_1)}{p(x|\omega_2)} = \underbrace{\frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}}_{\text{Independent from } x} = \theta$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Consider a medical scenario where $\omega_1 \Rightarrow$ healthy and $\omega_2 \Rightarrow$ sick

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Consider a medical scenario where $\omega_1 \Rightarrow$ healthy and $\omega_2 \Rightarrow$ sick

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases}$$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Consider a medical scenario where $\omega_1 \Rightarrow$ healthy and $\omega_2 \Rightarrow$ sick

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad \lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i = 2 \wedge j = 1 \\ 2 & \text{if } i = 1 \wedge j = 2 \end{cases}$$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Consider a medical scenario where $\omega_1 \Rightarrow \text{healthy}$ and $\omega_2 \Rightarrow \text{sick}$

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad \lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i = 2 \wedge j = 1 \\ 2 & \text{if } i = 1 \wedge j = 2 \end{cases}$$

$$\text{LR}_A = \frac{1 - 0}{1 - 0} \cdot \frac{P(\omega_2)}{P(\omega_1)} = \theta$$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

Consider a medical scenario where $\omega_1 \Rightarrow \text{healthy}$ and $\omega_2 \Rightarrow \text{sick}$

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad \lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i = 2 \wedge j = 1 \\ 2 & \text{if } i = 1 \wedge j = 2 \end{cases}$$

$$\text{LR}_A = \frac{1 - 0}{1 - 0} \cdot \frac{P(\omega_2)}{P(\omega_1)} = \theta \quad \text{LR}_B = \frac{2 - 0}{1 - 0} \cdot \frac{P(\omega_2)}{P(\omega_1)} = 2\theta$$

Classification: likelihood ratio

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} = \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \cdot \frac{P(\omega_2)}{P(\omega_1)}$$

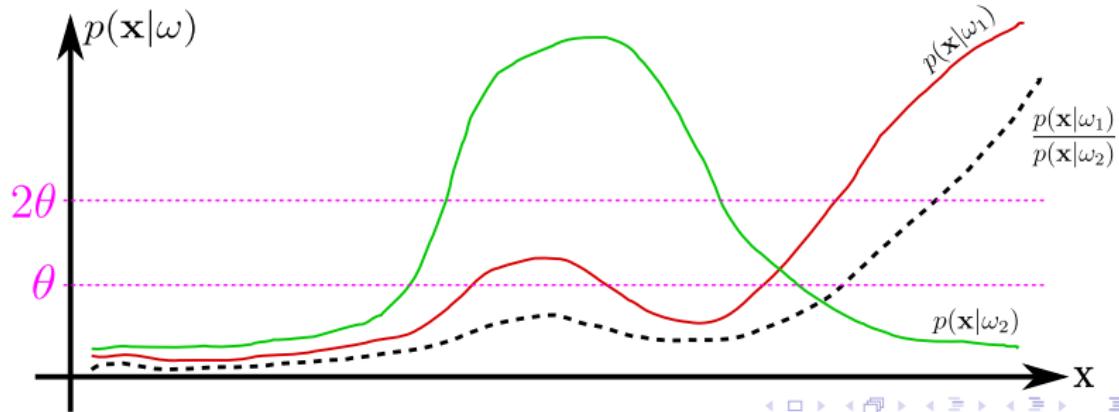
Consider a medical scenario where $\omega_1 \Rightarrow$ healthy and $\omega_2 \Rightarrow$ sick

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases}$$

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i = 2 \wedge j = 1 \\ 2 & \text{if } i = 1 \wedge j = 2 \end{cases}$$

$$\text{LR}_A = \frac{1 - 0}{1 - 0} \cdot \frac{P(\omega_2)}{P(\omega_1)} = \theta$$

$$\text{LR}_B = \frac{2 - 0}{1 - 0} \cdot \frac{P(\omega_2)}{P(\omega_1)} = 2\theta$$



Classification: Bayes error

General classification case:

- $\mathcal{A} = \{\alpha_1, \dots, \alpha_c\}$
- $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$

Classification: Bayes error

General classification case:

- $\mathcal{A} = \{\alpha_1, \dots, \alpha_c\}$
- $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$

Symmetrical/zero-one loss

- No loss to correct decision
- Unit loss to any error

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad i, j = 1, \dots, c$$

Classification: Bayes error

General classification case:

- $\mathcal{A} = \{\alpha_1, \dots, \alpha_c\}$
- $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$

Symmetrical/zero-one loss

- No loss to correct decision
- Unit loss to any error

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad i, j = 1, \dots, c$$

Conditional risk:

$$R(\alpha_i | \mathbf{x}) = \sum_{j=1}^c \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = \sum_{j \neq i}^c \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = 1 - P(\omega_i | \mathbf{x})$$

Classification: Bayes error

General classification case:

- $\mathcal{A} = \{\alpha_1, \dots, \alpha_c\}$
- $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$

Symmetrical/zero-one loss

- No loss to correct decision
- Unit loss to any error

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad i, j = 1, \dots, c$$

Conditional risk:

$$R(\alpha_i | \mathbf{x}) = \sum_{j=1}^c \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = \sum_{j \neq i} \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = 1 - P(\omega_i | \mathbf{x})$$

→ Bayes rule ⇒ Minimize conditional risk ⇒ Maximize posterior probability

Classification: Bayes error

General classification case:

- $\mathcal{A} = \{\alpha_1, \dots, \alpha_c\}$
- $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$

Symmetrical/zero-one loss

- No loss to correct decision
- Unit loss to any error

$$\lambda_{ij} = \begin{cases} 0 & \text{if } i = j \\ 1 & \text{if } i \neq j \end{cases} \quad i, j = 1, \dots, c$$

Conditional risk:

$$R(\alpha_i | \mathbf{x}) = \sum_{j=1}^c \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = \sum_{j \neq i} \lambda(\alpha_i | \omega_j) \cdot P(\omega_j | \mathbf{x}) = 1 - P(\omega_i | \mathbf{x})$$

→ Bayes rule ⇒ Minimize conditional risk ⇒ Maximize posterior probability

→ Bayes error:

⇒ Bayes risk for classification problems with zero-one loss

⇒ Irreducible misclassification probability

Summary

- **Loss (λ)**: cost of one action (α_i) VS a state of nature (ω_j)
- **Conditional risk ($R(\alpha_i|x)$)**: expected loss for observation x and action (α_i)
- **Risk (R)**: also *expected loss*; overall expected loss for a decision rule (α_i) across the whole distribution of data
- **Bayes risk**: the minimum possible risk (the theoretical lower bound)
- **Bayes rule**: Action selection function (γ) that selects the optimal action that minimizes the risk (Bayes risk)
- **Bayes error**: Bayes risk for classification problems with zero-one loss

Discriminant functions

- General manner for representing classifiers

Discriminant functions

- General manner for representing classifiers
- Notation $\Rightarrow g : \mathbb{R}^d \rightarrow \mathcal{R}$, where \mathcal{R} denotes the *acceptance region*

Discriminant functions

- General manner for representing classifiers
- Notation $\Rightarrow g : \mathbb{R}^d \rightarrow \mathcal{R}$, where \mathcal{R} denotes the *acceptance region*
- For $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$ states of nature:

Discriminant functions

- General manner for representing classifiers
- Notation $\Rightarrow g : \mathbb{R}^d \rightarrow \mathcal{R}$, where \mathcal{R} denotes the acceptance region
- For $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$ states of nature:
 - $\rightarrow g_1(\mathbf{x}), \dots, g_c(\mathbf{x})$
 - $\rightarrow \mathcal{R}_1, \dots, \mathcal{R}_c$

Discriminant functions

- General manner for representing classifiers
- Notation $\Rightarrow g : \mathbb{R}^d \rightarrow \mathcal{R}$, where \mathcal{R} denotes the acceptance region
- For $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$ states of nature:
 - $\rightarrow g_1(\mathbf{x}), \dots, g_c(\mathbf{x})$
 - $\rightarrow \mathcal{R}_1, \dots, \mathcal{R}_c$
- The classifier is described as:

$$\alpha_a(\mathbf{x}) : a = \arg \max_{i=1, \dots, c} g_i(\mathbf{x})$$

Discriminant functions

- General manner for representing classifiers
- Notation $\Rightarrow g : \mathbb{R}^d \rightarrow \mathcal{R}$, where \mathcal{R} denotes the acceptance region
- For $\mathcal{W} = \{\omega_1, \dots, \omega_c\}$ states of nature:
 - $\rightarrow g_1(\mathbf{x}), \dots, g_c(\mathbf{x})$
 - $\rightarrow \mathcal{R}_1, \dots, \mathcal{R}_c$
- The classifier is described as:

$$\alpha_a(\mathbf{x}) : a = \arg \max_{i=1, \dots, c} g_i(\mathbf{x})$$

\rightarrow For the Bayes classifier: $g_i(\mathbf{x}) = -R(\alpha_i | \mathbf{x}) = P(\omega_i | \mathbf{x})$

Discriminant functions

- Acceptance region:

$$\mathcal{R}_i = \{\mathbf{x} : g_i(\mathbf{x}) > g_j(\mathbf{x}) \forall i, j \in \{1, \dots, c\}, j \neq i\}$$

Discriminant functions

- Acceptance region:

$$\mathcal{R}_i = \{\mathbf{x} : g_i(\mathbf{x}) > g_j(\mathbf{x}) \forall i, j \in \{1, \dots, c\}, j \neq i\}$$

- Decision boundaries:

$$g_i(\mathbf{x}) = g_j(\mathbf{x})$$

Discriminant functions

- Acceptance region:

$$\mathcal{R}_i = \{\mathbf{x} : g_i(\mathbf{x}) > g_j(\mathbf{x}) \forall i, j \in \{1, \dots, c\}, j \neq i\}$$

- Decision boundaries:

$$g_i(\mathbf{x}) = g_j(\mathbf{x})$$

- When $c = 2$ (binary classification), **only one function** is considered:

$$g(\mathbf{x}) = g_1(\mathbf{x}) - g_2(\mathbf{x})$$

Outline

① Introduction

Where are we?

Computational learning VS Decision theory

② Bayesian decision theory

Two-class problem

General form

Risk

Discriminant functions

③ Statistical likelihood

Maximum likelihood estimation

④ Issues in computational learning

Bias-variance issues

Curse of dimensionality

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available
- Most typical scenario:

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available
- Most typical scenario:
 - Some vague and general knowledge about the situation
 - A number of *design samples* or *training data*

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available
 - Most typical scenario:
 - Some vague and general knowledge about the situation
 - A number of *design samples* or *training data*
- ⇒ Task: Estimate distributions using this piece of information

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available
 - Most typical scenario:
 - Some vague and general knowledge about the situation
 - A number of *design samples* or *training data*
- ⇒ Task: Estimate distributions using this piece of information
- **Parametric Pattern Recognition:** Particular case in which $p(\mathbf{x}|\omega_i)$ can be characterized by few parameters

Introduction

- Optimal classifier: requires $P(\omega_i)$ and $p(\mathbf{x}|\omega_i)$
 - Unfortunately, this complete knowledge is rarely available
 - Most typical scenario:
 - Some vague and general knowledge about the situation
 - A number of *design samples* or *training data*
- ⇒ Task: Estimate distributions using this piece of information
- Parametric Pattern Recognition: Particular case in which $p(\mathbf{x}|\omega_i)$ can be characterized by few parameters
 - Simplifies the problem of estimating an unknown function $p(\mathbf{x}|\omega_i)$ to one of estimating the parameters of a distribution

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$
 - $\mathcal{D}_i \sim p(\mathbf{x}|\omega_i)$ with $i = 1, \dots, c$ (i.i.d.)
 - $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{D}|}\}$

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$
 - $\mathcal{D}_i \sim p(\mathbf{x}|\omega_i)$ with $i = 1, \dots, c$ (i.i.d.)
 - $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{D}|}\}$
- Assume $p(\mathbf{x}|\omega_i)$ has a known parametric form ⇒ determined by θ_i

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$
 - $\mathcal{D}_i \sim p(\mathbf{x}|\omega_i)$ with $i = 1, \dots, c$ (i.i.d.)
 - $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{D}|}\}$
- Assume $p(\mathbf{x}|\omega_i)$ has a known parametric form \Rightarrow determined by θ_i
 - Distribution is written as $p(\mathbf{x}|\omega_i, \theta_i)$
 - Samples in \mathcal{D}_i give no information about θ_j if $j \neq i$

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$
 - $\mathcal{D}_i \sim p(\mathbf{x}|\omega_i)$ with $i = 1, \dots, c$ (i.i.d.)
 - $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{D}|}\}$
- Assume $p(\mathbf{x}|\omega_i)$ has a known parametric form ⇒ determined by θ_i
 - Distribution is written as $p(\mathbf{x}|\omega_i, \theta_i)$
 - Samples in \mathcal{D}_i give no information about θ_j if $j \neq i$
- We have **c separate problems**: use a set \mathcal{D} of **training samples** drawn independently from the probability density $p(\mathbf{x}|\theta)$ to estimate the unknown parameter vector θ

Parameter estimation

- Suppose the existence of a dataset $\mathcal{D} = \{\mathcal{D}_1, \dots, \mathcal{D}_c\}$
 - $\mathcal{D}_i \sim p(\mathbf{x}|\omega_i)$ with $i = 1, \dots, c$ (i.i.d.)
 - $\mathcal{D} = \{\mathbf{x}_1, \dots, \mathbf{x}_{|\mathcal{D}|}\}$
- Assume $p(\mathbf{x}|\omega_i)$ has a known parametric form ⇒ determined by θ_i
 - Distribution is written as $p(\mathbf{x}|\omega_i, \theta_i)$
 - Samples in \mathcal{D}_i give no information about θ_j if $j \neq i$
- We have **c separate problems**: use a set \mathcal{D} of **training samples** drawn independently from the probability density $p(\mathbf{x}|\theta)$ to estimate the unknown parameter vector θ

⇒ What about **prior** $P(\omega)$?

Parameter estimation

Likelihood

The function that measures how likely is to obtain precisely this dataset, given the parameters is the so-called **likelihood** of θ with regard to \mathcal{D} .

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{|\mathcal{D}|} p(\mathbf{x}_n|\theta)$$

Recall: Samples are drawn i.i.d. from the original data distribution

Parameter estimation

Likelihood

The function that measures how likely is to obtain precisely this dataset, given the parameters is the so-called **likelihood** of θ with regard to \mathcal{D} .

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{|\mathcal{D}|} p(\mathbf{x}_n|\theta)$$

Recall: Samples are drawn i.i.d. from the original data distribution

- **Task:** Find $\hat{\theta}$ that maximizes $p(\mathcal{D}|\theta)$

Parameter estimation

Likelihood

The function that measures how likely is to obtain precisely this dataset, given the parameters is the so-called **likelihood** of θ with regard to \mathcal{D} .

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{|\mathcal{D}|} p(\mathbf{x}_n|\theta)$$

Recall: Samples are drawn i.i.d. from the original data distribution

- **Task:** Find $\hat{\theta}$ that maximizes $p(\mathcal{D}|\theta)$
→ Maximum Likelihood Estimation (MLE)

Parameter estimation

Likelihood

The function that measures how likely is to obtain precisely this dataset, given the parameters is the so-called **likelihood** of θ with regard to \mathcal{D} .

$$p(\mathcal{D}|\theta) = \prod_{n=1}^{|\mathcal{D}|} p(\mathbf{x}_n|\theta)$$

Recall: Samples are drawn i.i.d. from the original data distribution

- **Task:** Find $\hat{\theta}$ that maximizes $p(\mathcal{D}|\theta)$
 - Maximum Likelihood Estimation (MLE)
 - In practise *log-likelihood* function: $I(\theta) \equiv \ln p(\mathcal{D}|\theta)$

Maximum Likelihood Estimation

- The MLE problem may be formally described as:

$$\hat{\theta} = \arg \max_{\theta} I(\theta) = \arg \max_{\theta} \ln p(\mathcal{D}|\theta)$$

Maximum Likelihood Estimation

- The MLE problem may be formally described as:

$$\hat{\theta} = \arg \max_{\theta} I(\theta) = \arg \max_{\theta} \ln p(\mathcal{D}|\theta)$$

- The set of necessary conditions for the MLE of θ can be obtained from the set of $|\theta|$ equations:

$$\nabla_{\theta} I(\theta) = 0 \text{ where } \nabla_{\theta} = \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_{|\theta|}} \end{bmatrix}$$

Gaussian case

Univariate case

Gaussian distribution with $\theta = (\mu, \sigma^2)$ where $\mathbf{x} \in \mathbb{R} \equiv x$:

$$\hat{\mu} = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} x_n \quad \hat{\sigma}^2 = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} (x_n - \hat{\mu})^2$$

Multivariate case

Gaussian distribution with $\theta = (\boldsymbol{\mu}, \boldsymbol{\Sigma})$ where $\mathbf{x} \in \mathbb{R}^d$:

$$\hat{\boldsymbol{\mu}} = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} \mathbf{x}_n \quad \hat{\boldsymbol{\Sigma}} = \frac{1}{|\mathcal{D}|} \sum_{n=1}^{|\mathcal{D}|} (\mathbf{x}_n - \hat{\boldsymbol{\mu}})(\mathbf{x}_n - \hat{\boldsymbol{\mu}})^t$$

Outline

① Introduction

Where are we?

Computational learning VS Decision theory

② Bayesian decision theory

Two-class problem

General form

Risk

Discriminant functions

③ Statistical likelihood

Maximum likelihood estimation

④ Issues in computational learning

Bias-variance issues

Curse of dimensionality

Generalization error

Generalization error

- How accurately an algorithm is able to **predict** outcomes for previously **unseen data**

Generalization error

- How accurately an algorithm is able to **predict** outcomes for previously **unseen data**
 - **Risk** or **out-of-sample** error

Generalization error

- How accurately an algorithm is able to **predict** outcomes for previously **unseen data**
 - **Risk** or **out-of-sample** error
- Main causes:
 - 1) **Model complexity** → Reproduce the underlying distribution
 - 2) **Data availability** → Representative of the distribution to model

Generalization error

- How accurately an algorithm is able to **predict** outcomes for previously **unseen data**
 - Risk or **out-of-sample** error
- Main causes:
 - 1) **Model complexity** → Reproduce the underlying distribution
 - 2) **Data availability** → Representative of the distribution to model
- Sources of error:
 - 1) **Bias** → **Systematic assumptions** by the model (*underfitting*)
 - 2) **Variance** → **Sensitivity** to training data (*overfitting*)

Bias-variance decomposition

Bias-variance decomposition

Regression task:

Bias-variance decomposition

Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$

Bias-variance decomposition

Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$
- **Premise:** approximate $\hat{f}(\mathbf{x})$ with set \mathcal{D} generated by $f(\mathbf{x})$

Bias-variance decomposition

Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$
- **Premise:** approximate $\hat{f}(\mathbf{x})$ with set \mathcal{D} generated by $f(\mathbf{x})$
 - Random variations selecting $\mathcal{D} \Rightarrow$ better or worse approximation

Bias-variance decomposition

Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$
- **Premise:** approximate $\hat{f}(\mathbf{x})$ with set \mathcal{D} generated by $f(\mathbf{x})$
 - Random variations selecting \mathcal{D} ⇒ better or worse approximation
- **Effectiveness** ⇒ Deviation from desired optimal averaged over all \mathcal{D} sets

Bias-variance decomposition

Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$
- **Premise:** approximate $\hat{f}(\mathbf{x})$ with set \mathcal{D} generated by $f(\mathbf{x})$
 - Random variations selecting \mathcal{D} ⇒ better or worse approximation
- **Effectiveness** ⇒ Deviation from desired optimal averaged over all \mathcal{D} sets

$$\text{Err}(\mathbf{x}) = \underbrace{\left(\mathbb{E}_{\mathcal{D}}[\hat{f}(\mathbf{x}; \mathcal{D})] - f(\mathbf{x}) \right)^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathcal{D}} \left[\hat{f}(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[\hat{f}(\mathbf{x}; \mathcal{D})] \right]}_{\text{Variance}} + \underbrace{\sigma_e^2}_{\text{Irreducible noise}}$$

Bias-variance decomposition

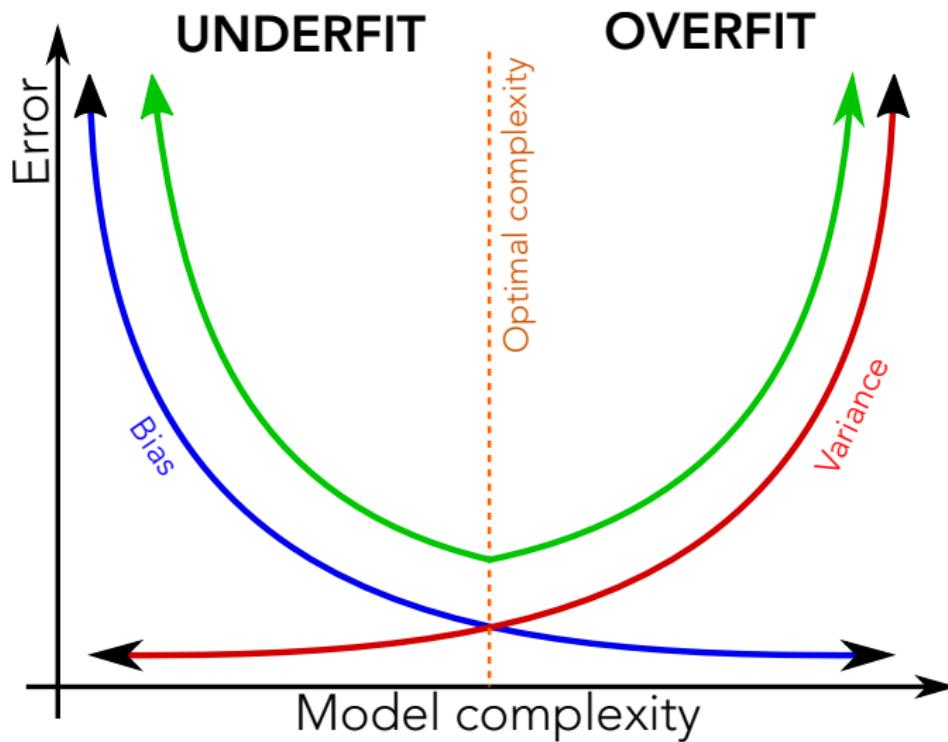
Regression task:

- There exists a true (but unknown) function $f(\mathbf{x})$
- **Premise:** approximate $\hat{f}(\mathbf{x})$ with set \mathcal{D} generated by $f(\mathbf{x})$
 → Random variations selecting \mathcal{D} ⇒ better or worse approximation
- **Effectiveness** ⇒ Deviation from desired optimal averaged over all \mathcal{D} sets

$$\text{Err}(\mathbf{x}) = \underbrace{\left(\mathbb{E}_{\mathcal{D}}[\hat{f}(\mathbf{x}; \mathcal{D})] - f(\mathbf{x}) \right)^2}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathcal{D}} \left[\hat{f}(\mathbf{x}; \mathcal{D}) - \mathbb{E}_{\mathcal{D}}[\hat{f}(\mathbf{x}; \mathcal{D})] \right]}_{\text{Variance}} + \underbrace{\sigma_e^2}_{\text{Irreducible noise}}$$

- **Bias:** Systematic deviation from $f(\mathbf{x})$
- **Variance:** Dependence on the data sampling in \mathcal{D}

Bias-variance tradeoff



Curse of dimensionality

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
→ In practise, may become an issue (high-dimensional spaces)

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse
 - Severe overfitting risk

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse
 - Severe overfitting risk
- Possible strategies to palliate it:

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse
 - Severe overfitting risk
- Possible strategies to palliate it:
 - Adequate design of the feature extraction stage

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse
 - Severe overfitting risk
- Possible strategies to palliate it:
 - Adequate design of the feature extraction stage
 - Feature selection: Select subset of features $d' < d$

Curse of dimensionality

- Amount of features d used $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
- Theoretically, estimator benefits from the use of larger d
 - In practise, may become an issue (high-dimensional spaces)
 - Data increasingly becomes sparse
 - Severe overfitting risk
- Possible strategies to palliate it:
 - Adequate design of the feature extraction stage
 - Feature selection: Select subset of features $d' < d$
 - Model regularization: Add constraints/penalties to the loss function

T2: Computational learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

T3: Model evaluation

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Introduction

Motivation

Relevance of the figure of merit

② General principles

Data partitioning

Cross-validation procedures

③ Classification

Binary case

Multiclass scenario

Other cases

④ Regression

Figures of merit

Outline

① Introduction

Motivation

Relevance of the figure of merit

② General principles

Data partitioning

Cross-validation procedures

③ Classification

Binary case

Multiclass scenario

Other cases

④ Regression

Figures of merit

Principles

- So far: **Infer knowledge** (*train* model) and **predict** (*test* model)

Principles

- So far: **Infer knowledge** (*train* model) and **predict** (*test* model)
→ How **well** is it **performing**? ⇒ Model evaluation

Principles

- So far: **Infer knowledge** (*train* model) and **predict** (*test* model)
→ How **well** is it **performing**? ⇒ Model evaluation
- Is model evaluation connected to **loss**, **risk**, and **error**?

Principles

- So far: **Infer knowledge** (*train* model) and **predict** (*test* model)
 - How **well** is it **performing**? ⇒ Model evaluation

- Is model evaluation connected to **loss**, **risk**, and **error**?
 - Related but **not the same**:

Concept	Phase	Goal	Module	Meaning
Loss	Train	Guide the optimization process	T2 (Computational learning)	Error on a single sample
Risk ¹				Expected loss value across the data distribution

Evaluation	Test	Quantify the performance of the model	T3 (Model evaluation)	How well the model performs on unseen data
------------	------	---------------------------------------	--------------------------	--

¹ Equals error considering a zero-one loss function.

True, expected, and empirical risk/error

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

$$R(\gamma) = \mathbb{E}_{(\mathbf{x}, \omega) \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}) | \omega)]$$

⇒ Purely theoretical (not computable)

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

$$R(\gamma) = \mathbb{E}_{(\mathbf{x}, \omega) \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}) | \omega)]$$

⇒ Purely theoretical (not computable)

- Expected risk/error: **expected true error** averaged over all possible training sets \mathcal{D} that could be drawn from $P(\mathbf{x}, \omega)$:

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

$$R(\gamma) = \mathbb{E}_{(\mathbf{x}, \omega) \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}) | \omega)]$$

⇒ Purely theoretical (not computable)

- Expected risk/error: **expected true error** averaged over all possible training sets \mathcal{D} that could be drawn from $P(\mathbf{x}, \omega)$:

$$R_{\mathcal{D}}(\gamma) = \mathbb{E}_{(\mathbf{x}_i, \omega_i) \in \mathcal{D} \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}_i; \mathcal{D}) | \omega_i)]$$

⇒ Purely theoretical (not computable)

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

$$R(\gamma) = \mathbb{E}_{(\mathbf{x}, \omega) \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}) | \omega)]$$

⇒ Purely theoretical (not computable)

- Expected risk/error: **expected true error** averaged over all possible training sets \mathcal{D} that could be drawn from $P(\mathbf{x}, \omega)$:

$$R_{\mathcal{D}}(\gamma) = \mathbb{E}_{(\mathbf{x}_i, \omega_i) \in \mathcal{D} \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}_i; \mathcal{D}) | \omega_i)]$$

⇒ Purely theoretical (not computable)

- Empirical risk/error: estimation of the **true/expected error** using a finite data collection $\mathcal{D} = \{(\mathbf{x}_1, \omega_1), \dots, (\mathbf{x}_{|\mathcal{D}|}, \omega_{|\mathcal{D}|})\}$:

True, expected, and empirical risk/error

- True risk/error: the **expected loss** under true distribution $P(\mathbf{x}, \omega)$:

$$R(\gamma) = \mathbb{E}_{(\mathbf{x}, \omega) \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}) | \omega)]$$

⇒ Purely theoretical (not computable)

- Expected risk/error: **expected true error** averaged over all possible training sets \mathcal{D} that could be drawn from $P(\mathbf{x}, \omega)$:

$$R_{\mathcal{D}}(\gamma) = \mathbb{E}_{(\mathbf{x}_i, \omega_i) \in \mathcal{D} \sim P(\mathbf{x}, \omega)} [\lambda(\gamma(\mathbf{x}_i; \mathcal{D}) | \omega_i)]$$

⇒ Purely theoretical (not computable)

- Empirical risk/error: estimation of the **true/expected error** using a finite data collection $\mathcal{D} = \{(\mathbf{x}_1, \omega_1), \dots, (\mathbf{x}_{|\mathcal{D}|}, \omega_{|\mathcal{D}|})\}$:

$$\hat{R}(\gamma) = \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x}_i, \omega_i) \in \mathcal{D}} \lambda(\gamma(\mathbf{x}_i) | \omega_i)$$

⇒ Practical and computable

(Training) Loss and (evaluation) metric/figure of merit

There exist **two levels** of relationship:

(Training) Loss and (evaluation) metric/figure of merit

There exist **two levels** of relationship:

1. **Aligned** case: Training loss and evaluation metric **match**
2. **Misaligned** case: Training loss and evaluation metric **differ**

(Training) Loss and (evaluation) metric/figure of merit

There exist **two levels** of relationship:

1. **Aligned** case: Training loss and evaluation metric **match**
2. **Misaligned** case: Training loss and evaluation metric **differ**
 - **Loss** is a *proxy* easier to optimize
 - **Metric** measures *real-world* performance

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

1. Feature extraction $\Rightarrow \mathbf{x} \in \mathbb{R}^d$

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

1. Feature extraction $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
2. Two states of nature $\Rightarrow \mathcal{W} = \{\text{sunny}, \text{rainy}\}$

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

1. Feature extraction $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
2. Two states of nature $\Rightarrow \mathcal{W} = \{\text{sunny}, \text{rainy}\}$
3. Labeled dataset $\Rightarrow \mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_r$ with $|\mathcal{D}| = 365$ days

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

1. Feature extraction $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
2. Two states of nature $\Rightarrow \mathcal{W} = \{\text{sunny}, \text{rainy}\}$
3. Labeled dataset $\Rightarrow \mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_r$ with $|\mathcal{D}| = 365$ days
 - $\mathcal{D}_s = \{(\mathbf{x}_1, \text{s}), \dots, (\mathbf{x}_{355}, \text{s})\}$
 - $\mathcal{D}_r = \{(\mathbf{x}_{356}, \text{r}), \dots, (\mathbf{x}_{365}, \text{r})\}$

Example: Weather prediction

A client reaches two AI engineers to design a **wheather prediction** system that works on a daily basis:

1. Feature extraction $\Rightarrow \mathbf{x} \in \mathbb{R}^d$
2. Two states of nature $\Rightarrow \mathcal{W} = \{\text{sunny}, \text{rainy}\}$
3. Labeled dataset $\Rightarrow \mathcal{D} = \mathcal{D}_s \cup \mathcal{D}_r$ with $|\mathcal{D}| = 365$ days
 - $\mathcal{D}_s = \{(\mathbf{x}_1, \text{s}), \dots, (\mathbf{x}_{355}, \text{s})\}$
 - $\mathcal{D}_r = \{(\mathbf{x}_{356}, \text{r}), \dots, (\mathbf{x}_{365}, \text{r})\}$
4. Procedure:
 - Use \mathcal{D} to adjust and evaluate
 - **Metric:** number of correct predictions

Example: Weather prediction

Engineer #1

Engineer #2

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

Engineer #2

Policy: Always predicts sunny

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

- Estimations for \mathcal{D}_s :

- ✓ 350 as sunny
- ✗ 5 as rainy

- Estimations for \mathcal{D}_r :

- ✗ 5 as sunny
- ✓ 5 as rainy

Engineer #2

Policy: Always predicts sunny

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

- Estimations for \mathcal{D}_s :

- ✓ 350 as sunny
- ✗ 5 as rainy

- Estimations for \mathcal{D}_r :

- ✗ 5 as sunny
- ✓ 5 as rainy

Engineer #2

Policy: Always predicts sunny

$$\text{Acc}_{\#1} = \frac{350 + 5}{365} \approx 97\%$$

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

- Estimations for \mathcal{D}_s :

- ✓ 350 as sunny
- ✗ 5 as rainy

- Estimations for \mathcal{D}_r :

- ✗ 5 as sunny
- ✓ 5 as rainy

$$\text{Acc}_{\#1} = \frac{350 + 5}{365} \approx 97\%$$

Engineer #2

Policy: Always predicts sunny

- Estimations for \mathcal{D}_s :

- ✓ 355 as sunny
- ✓ 0 as rainy

- Estimations for \mathcal{D}_r :

- ✗ 10 as sunny
- ✗ 0 as rainy

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

- Estimations for \mathcal{D}_s :

✓ 350 as sunny
✗ 5 as rainy

- Estimations for \mathcal{D}_r :

✗ 5 as sunny
✓ 5 as rainy

$$\text{Acc}_{\#1} = \frac{350 + 5}{365} \approx 97\%$$

Engineer #2

Policy: Always predicts sunny

- Estimations for \mathcal{D}_s :

✓ 355 as sunny
✓ 0 as rainy

- Estimations for \mathcal{D}_r :

✗ 10 as sunny
✗ 0 as rainy

$$\text{Acc}_{\#2} = \frac{355 + 0}{365} \approx 97\%$$

Example: Weather prediction

Engineer #1

Policy: Adequate study/design

- Estimations for \mathcal{D}_s :

✓ 350 as sunny
✗ 5 as rainy

- Estimations for \mathcal{D}_r :

✗ 5 as sunny
✓ 5 as rainy

$$\text{Acc}_{\#1} = \frac{350 + 5}{365} \approx 97\%$$

Engineer #2

Policy: Always predicts sunny

- Estimations for \mathcal{D}_s :

✓ 355 as sunny
✓ 0 as rainy

- Estimations for \mathcal{D}_r :

✗ 10 as sunny
✗ 0 as rainy

$$\text{Acc}_{\#2} = \frac{355 + 0}{365} \approx 97\%$$

Which is the **issue** here?

Outline

① Introduction

Motivation

Relevance of the figure of merit

② General principles

Data partitioning

Cross-validation procedures

③ Classification

Binary case

Multiclass scenario

Other cases

④ Regression

Figures of merit

Introduction

Introduction

- In **practical** scenarios \Rightarrow finite set of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$

Introduction

- In **practical** scenarios \Rightarrow **finite set** of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- Assortment \mathcal{D} is used for several **procedures** in the model creation:

Introduction

- In **practical** scenarios \Rightarrow **finite set** of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- Assortment \mathcal{D} is used for several **procedures** in the model creation:
 1. **Train:** Provide **data** for the model **to learn**
 2. **Tune:** Obtain the **most adequate** set of **(hyper)parameters**
 3. **Assess:** Evaluate the **goodness** of the model

Introduction

- In **practical** scenarios \Rightarrow **finite set** of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- Assortment \mathcal{D} is used for several **procedures** in the model creation:
 1. **Train:** Provide **data** for the model **to learn**
 2. **Tune:** Obtain the **most adequate** set of **(hyper)parameters**
 3. **Assess:** Evaluate the **goodness** of the model \Rightarrow *Entire dataset? Subsets?*

Introduction

- In **practical** scenarios \Rightarrow **finite set** of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- Assortment \mathcal{D} is used for several **procedures** in the model creation:
 1. **Train:** Provide **data** for the model **to learn**
 2. **Tune:** Obtain the **most adequate** set of **(hyper)parameters**
 3. **Assess:** Evaluate the **goodness** of the model \Rightarrow *Entire dataset? Subsets?*
- Typical strategies:

Introduction

- In **practical** scenarios \Rightarrow finite set of data $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- Assortment \mathcal{D} is used for several **procedures** in the model creation:
 1. **Train:** Provide **data** for the model **to learn**
 2. **Tune:** Obtain the **most adequate** set of **(hyper)parameters**
 3. **Assess:** Evaluate the **goodness** of the model

\Rightarrow *Entire dataset? Subsets?*
- Typical strategies:
 1. **Data partitioning:** Adequately **divide** assortment \mathcal{D}
 2. **Cross-validation procedures:** Exhaustively **explore** assortment \mathcal{D}

Formalization

Formalization

- **Premise:** The procedures must use different data partitions

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage \Rightarrow Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$
- **Considerations** (typical):

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$
- **Considerations** (typical):
 1. \mathcal{D}_{train} is the largest subset: $|\mathcal{D}_{train}| \gg |\mathcal{D}_{validation}|, |\mathcal{D}_{test}|$

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$
- **Considerations** (typical):
 1. \mathcal{D}_{train} is the largest subset: $|\mathcal{D}_{train}| \gg |\mathcal{D}_{validation}|, |\mathcal{D}_{test}|$
 2. $\mathcal{D}_{validation}$ is a subset of \mathcal{D}_{train} : $\mathcal{D}_{validation} \subset \mathcal{D}_{train}$

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$
- **Considerations** (typical):
 1. \mathcal{D}_{train} is the largest subset: $|\mathcal{D}_{train}| \gg |\mathcal{D}_{validation}|, |\mathcal{D}_{test}|$
 2. $\mathcal{D}_{validation}$ is a subset of \mathcal{D}_{train} : $\mathcal{D}_{validation} \subset \mathcal{D}_{train}$
 3. \mathcal{D}_{test} may be external

Formalization

- **Premise:** The procedures must use different data partitions
 - Otherwise, data leakage ⇒ Optimistic estimates
- **Subsets** of the assortment: $\mathcal{D} = \underbrace{\mathcal{D}_{train}}_{Train} \cup \underbrace{\mathcal{D}_{validation}}_{Tune} \cup \underbrace{\mathcal{D}_{test}}_{Assess}$
 - Disjoint subsets: $\mathcal{D}_{train} \cap \mathcal{D}_{validation} \cap \mathcal{D}_{test} = \emptyset$
- **Considerations** (typical):
 1. \mathcal{D}_{train} is the largest subset: $|\mathcal{D}_{train}| \gg |\mathcal{D}_{validation}|, |\mathcal{D}_{test}|$
 2. $\mathcal{D}_{validation}$ is a subset of \mathcal{D}_{train} : $\mathcal{D}_{validation} \subset \mathcal{D}_{train}$
 3. \mathcal{D}_{test} may be external
 4. In general, random division
 - Stratification to maintain label distribution

Summary

Partition	Purpose	Used for	Ratio (%)
D_{train}	Fit model parameters	Learning the model	60 – 80
$D_{validation}$	Tune hyperparameters, model selection	Model selection	10 – 20
D_{test}	Evaluate final performance on unseen data	Assessment	10 – 20

Introduction

- So far \Rightarrow static partitions over assortment \mathcal{D}

Introduction

- So far \Rightarrow static partitions over assortment \mathcal{D}
 - \rightarrow Possible biases due to skews in \mathcal{D}

Introduction

- So far \Rightarrow static partitions over assortment \mathcal{D}
 - Possible biases due to skews in \mathcal{D}
- **Possible solution:** Repeatedly perform data partitioning on \mathcal{D}

Introduction

- So far \Rightarrow static partitions over assortment \mathcal{D}
 - \rightarrow Possible biases due to skews in \mathcal{D}
- **Possible solution:** Repeatedly perform data *partitioning* on \mathcal{D}
 - \rightarrow Reduces variance of the performance estimate (by averaging)
 - \rightarrow Reduces estimation bias compared to a single partitioning

Introduction

- So far \Rightarrow static partitions over assortment \mathcal{D}
 - Possible biases due to skews in \mathcal{D}
- **Possible solution:** Repeatedly perform data *partitioning* on \mathcal{D}
 - Reduces variance of the performance estimate (by averaging)
 - Reduces estimation bias compared to a single partitioning
- Typical strategies:
 1. Hold-out validation
 2. k -fold Cross-validation
 3. Stratified k -fold Cross-validation
 4. Leave-one-out Cross-validation
 5. Leave-p-out Cross-validation

Strategies

1. Hold-out validation

- Simplest case: fixed \mathcal{D}_{train} , $\mathcal{D}_{validation}$, and \mathcal{D}_{test}
- Error is computed only once
- Features:
 - ✓ Fast
 - ✗ High variance (split-dependent)

Strategies

1. Hold-out validation

- Simplest case: fixed \mathcal{D}_{train} , $\mathcal{D}_{validation}$, and \mathcal{D}_{test}
- Error is computed only once
- Features:
 - ✓ Fast
 - ✗ High variance (split-dependent)

2. k -fold CV

- Split data into k roughly equal folds
- Train with $k - 1$ partitions; test with the remaining one
- Repeat the procedure k times \Rightarrow As many as folds created
- Features:
 - ✗ Bias: Slightly over-optimistic for small k (\mathcal{D}_{train} sets are smaller)
 - ✓ Variance: Decreases as k decreases

Strategies

3. Stratified k -fold CV

- Variant of *k-fold* where class proportions are preserved in each fold (label imbalance)

Strategies

3. Stratified k -fold CV

- Variant of k -fold where class proportions are preserved in each fold (label imbalance)

4. Leave-one-out CV

- Extreme case of k -fold CV with $k = |\mathcal{D}|$
- Train with $|\mathcal{D}| - 1$ points; test with the remaining one
- Features:
 - ✓ Bias: Very low (trained on nearly full dataset)
 - ✗ Variance: Very high (small changes in data affect each fold)

Strategies

3. Stratified k -fold CV

- Variant of k -fold where class proportions are preserved in each fold (label imbalance)

4. Leave-one-out CV

- Extreme case of k -fold CV with $k = |\mathcal{D}|$
- Train with $|\mathcal{D}| - 1$ points; test with the remaining one
- Features:
 - ✓ Bias: Very low (trained on nearly full dataset)
 - ✗ Variance: Very high (small changes in data affect each fold)

5. Leave- p -out CV

- Generalization of leave-one-out: leaves p samples out at a time
- Theoretical number of folds: $\binom{|\mathcal{D}|}{p}$
 - Infeasible in practical cases unless p is low
- Used mainly in theoretical studies of CV properties

Summary

Strategy	Bias	Variance	Cost
Hold-out	Medium	High	1 fit
(Stratified) k -fold	Low	Medium	k fits
Leave-one-out	Very low	High	$ \mathcal{D} $ fits
Leave- p -out	Very low	Very high	$\binom{ \mathcal{D} }{p}$ fits

Outline

① Introduction

Motivation

Relevance of the figure of merit

② General principles

Data partitioning

Cross-validation procedures

③ Classification

Binary case

Multiclass scenario

Other cases

④ Regression

Figures of merit

Confusion or error matrix

- Binary scenario $\mathcal{W} = \{\omega_1, \omega_2\} \equiv \{\omega, \bar{\omega}\}$

Confusion or error matrix

- Binary scenario $\mathcal{W} = \{\omega_1, \omega_2\} \equiv \{\omega, \bar{\omega}\}$
- Confusion / error matrix:

Expected	Prediction	
	ω	$\bar{\omega}$
ω	True Positive (TP)	False Negative (FN)
$\bar{\omega}$	False Positive (FP)	True Negative (TN)

Confusion or error matrix

- Binary scenario $\mathcal{W} = \{\omega_1, \omega_2\} \equiv \{\omega, \bar{\omega}\}$
- Confusion / error matrix:

Expected	Prediction	
	ω	$\bar{\omega}$
ω	True Positive (TP)	False Negative (FN)
$\bar{\omega}$	False Positive (FP)	True Negative (TN)

- Trained classification model $\hat{f} : \mathbb{R}^d \rightarrow \mathcal{W}$
- Test dataset $\Rightarrow \mathcal{D}_{test} = \mathcal{D}_\omega \cup \mathcal{D}_{\bar{\omega}}$

Confusion or error matrix

- Binary scenario $\mathcal{W} = \{\omega_1, \omega_2\} \equiv \{\omega, \bar{\omega}\}$
- Confusion / error matrix:

Expected	Prediction	
	ω	$\bar{\omega}$
ω	True Positive (TP)	False Negative (FN)
$\bar{\omega}$	False Positive (FP)	True Negative (TN)

- Trained classification model $\hat{f} : \mathbb{R}^d \rightarrow \mathcal{W}$
- Test dataset $\Rightarrow \mathcal{D}_{test} = \mathcal{D}_\omega \cup \mathcal{D}_{\bar{\omega}}$

$$\begin{array}{ll} \text{TP: } \sum_{\mathbf{x} \in \mathcal{D}_\omega} [\hat{f}(\mathbf{x}) = \omega] & \text{FN: } \sum_{\mathbf{x} \in \mathcal{D}_\omega} [\hat{f}(\mathbf{x}) = \bar{\omega}] \\ \text{TN: } \sum_{\mathbf{x} \in \mathcal{D}_{\bar{\omega}}} [\hat{f}(\mathbf{x}) = \bar{\omega}] & \text{FP: } \sum_{\mathbf{x} \in \mathcal{D}_{\bar{\omega}}} [\hat{f}(\mathbf{x}) = \omega] \end{array}$$

Confusion or error matrix

- Binary scenario $\mathcal{W} = \{\omega_1, \omega_2\} \equiv \{\omega, \bar{\omega}\}$
- Confusion / error matrix:

Expected	Prediction	
	ω	$\bar{\omega}$
ω	True Positive (TP)	False Negative (FN)
$\bar{\omega}$	False Positive (FP)	True Negative (TN)

- Trained classification model $\hat{f} : \mathbb{R}^d \rightarrow \mathcal{W}$
- Test dataset $\Rightarrow \mathcal{D}_{test} = \mathcal{D}_\omega \cup \mathcal{D}_{\bar{\omega}}$

$$\begin{array}{lll} \text{TP: } \sum_{\mathbf{x} \in \mathcal{D}_\omega} [\hat{f}(\mathbf{x}) = \omega] & \text{FN: } \sum_{\mathbf{x} \in \mathcal{D}_\omega} [\hat{f}(\mathbf{x}) = \bar{\omega}] & |\mathcal{D}_\omega| = \text{TP} + \text{FN} \\ \text{TN: } \sum_{\mathbf{x} \in \mathcal{D}_{\bar{\omega}}} [\hat{f}(\mathbf{x}) = \bar{\omega}] & \text{FP: } \sum_{\mathbf{x} \in \mathcal{D}_{\bar{\omega}}} [\hat{f}(\mathbf{x}) = \omega] & |\mathcal{D}_{\bar{\omega}}| = \text{TN} + \text{FP} \end{array}$$

Examples of confusion matrices

Engineer #1

Expected	Prediction	
	Sunny	Rainy
Sunny	350	5
Rainy	5	5

Engineer #2

Expected	Prediction	
	Sunny	Rainy
Sunny	355	0
Rainy	10	0

Metrics

Metrics

1. Accuracy (Acc)

- Ratio between **correct predictions** and **total number of guesses**:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} = \frac{\text{TP} + \text{TN}}{|\mathcal{D}_\omega| + |\mathcal{D}_{\bar{\omega}}|}$$

- **Errors are equally weighted**
- Suitable for **balanced** scenarios

Metrics

1. Accuracy (Acc)

- Ratio between **correct predictions** and **total number of guesses**:

$$\text{Acc} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FN} + \text{TN} + \text{FP}} = \frac{\text{TP} + \text{TN}}{|\mathcal{D}_\omega| + |\mathcal{D}_{\bar{\omega}}|}$$

- **Errors are equally weighted**
- Suitable for **balanced scenarios**

2. Precision (P)

- Correct claims VS all positive predictions:

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

- Penalizes **false alarms**

3. Recall (R)

- Correct claims VS expected positive predictions:

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

- Penalizes **missed positives**

Metrics

4. F-measure (F_β)

- P and R **optimize** different aspects \Rightarrow need for a **single indicator**
- **Harmonic mean** between P and R:

$$F_\beta = (1 + \beta^2) \cdot \frac{P \cdot R}{\beta^2 \cdot P + R}$$

- Most commonly, $\beta = 1$:

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

Metrics

4. F-measure (F_β)

- P and R optimize different aspects \Rightarrow need for a single indicator
- Harmonic mean between P and R:

$$F_\beta = (1 + \beta^2) \cdot \frac{P \cdot R}{\beta^2 \cdot P + R}$$

- Most commonly, $\beta = 1$:

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

\Rightarrow Considerations:

- Acc: Global for all classes
- P, R, F_1 : Computed for each individual class

Exercise

Engineer #1

Expected	Prediction	
	Sunny	Rainy
Sunny	350	5
Rainy	5	5

Engineer #2

Expected	Prediction	
	Sunny	Rainy
Sunny	355	0
Rainy	10	0

Metrics (ii)

5. True Positive Rate (TPR)

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

- Also: *Recall, Sensitivity*

6. True Negative Rate (TNR)

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- Also: *Specificity*

Metrics (ii)

5. True Positive Rate (TPR)

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

- Also: *Recall, Sensitivity*

6. True Negative Rate (TNR)

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- Also: *Specificity*

7. False Positive Rate (FPR)

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

8. False Negative Rate (FNR)

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

Metrics (ii)

5. True Positive Rate (TPR)

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

- Also: *Recall, Sensitivity*

6. True Negative Rate (TNR)

$$\text{TNR} = \frac{\text{TN}}{\text{TN} + \text{FP}}$$

- Also: *Specificity*

7. False Positive Rate (FPR)

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

8. False Negative Rate (FNR)

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

$$\text{TPR} + \text{FNR} = 1$$

$$\text{TNR} + \text{FPR} = 1$$

Summary

Figure of merit	Formula	Computation
Accuracy (Acc)	$(TP+TN)/(TP+FN+TN+FP)$	Global
Precision (P)	$TP/(TP+FP)$	Class-wise
Recall (R)	$TP/(TP+FN)$	Class-wise
F-measure (F_1)	$2 \cdot TP / (2 \cdot TP + FP + FN)$	Class-wise
True Positive Rate (TPR)	$TP/(TP+FN)$	Class-wise
True Negative Rate (TNR)	$TN/(TN+FP)$	Class-wise
False Positive Rate (FPR)	$FP/(FP+TN)$	Class-wise
False Negative Rate (FNR)	$FN/(FN+TP)$	Class-wise

Error trade-offs

- Decision thresholds remarkably impacts the recognition performance

Error trade-offs

- Decision thresholds remarkably impacts the recognition performance
- Intuitively:
 - **Lower** threshold: Increase in Positive predictions ($\text{TPR} \uparrow, \text{FPR} \uparrow$)
 - **Higher** thresholds: Decrease in Positive predictions ($\text{TPR} \downarrow, \text{FPR} \downarrow$)

Error trade-offs

- Decision thresholds remarkably impacts the recognition performance
- Intuitively:
 - Lower threshold: Increase in Positive predictions ($\text{TPR} \uparrow, \text{FPR} \uparrow$)
 - Higher thresholds: Decrease in Positive predictions ($\text{TPR} \downarrow, \text{FPR} \downarrow$)
- Each threshold corresponds to a particular duple (TPR, FPR)

Receiver Operating Characteristic and Area Under Curve

ROC curve: **FPR** against **TPR** as the **decision threshold** varies

- Top-left corner ($\text{FPR}=0, \text{TPR}=1$): perfect model
- Diagonal line ($\text{FPR} = \text{TPR}$): random guessing
- Below diagonal: worse than random

Receiver Operating Characteristic and Area Under Curve

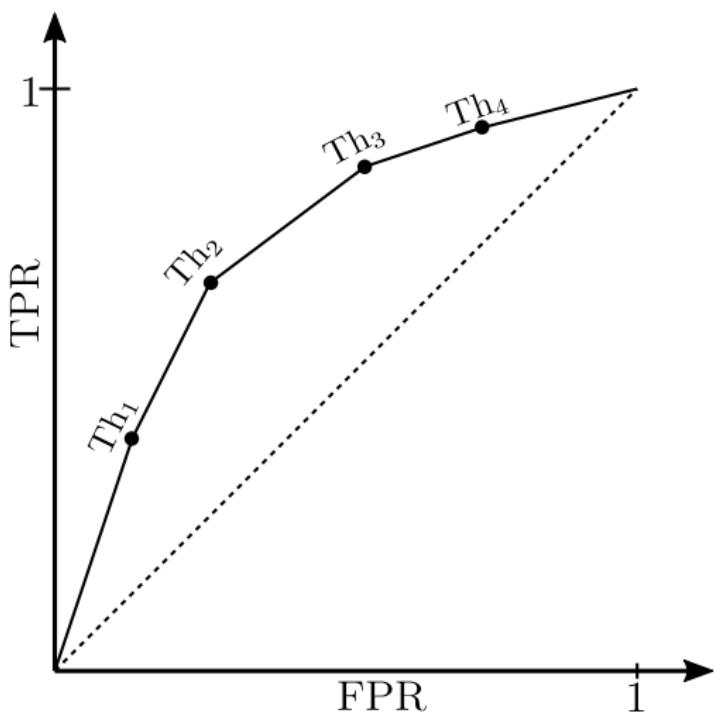
ROC curve: FPR against TPR as the decision threshold varies

- Top-left corner ($\text{FPR}=0, \text{TPR}=1$): perfect model
- Diagonal line ($\text{FPR} = \text{TPR}$): random guessing
- Below diagonal: worse than random

AUC: Area under the ROC curve

- Independent of class imbalance
- Threshold-free metric
- Ranges:
 - $0.5 < \text{AUC} \leq 1 \Rightarrow$ Performs adequately
 - $\text{AUC} = 0.5 \Rightarrow$ Random guessing
 - $0 \leq \text{AUC} < 0.5 \Rightarrow$ Underperforming

Receiver Operating Characteristic and Area Under Curve



Introduction

- In **binary** scenarios ($|\mathcal{W}| = 2$), one class is selected as **positive**
→ **Multiclass** ($|\mathcal{W}| > 2$): each class can be positive ⇒ **one-vs-all**
- Accuracy requires **no adaptation**

Introduction

- In **binary** scenarios ($|\mathcal{W}| = 2$), one class is selected as **positive**
→ **Multiclass** ($|\mathcal{W}| > 2$): each class can be positive ⇒ **one-vs-all**
- Accuracy requires **no adaptation**
- **Averaging** process to **summarize the performance** across all classes
→ Different averaging strategies

Adaptations

1. Micro-Average

- Aggregates at the error level:

$$\text{Micro-P} = \frac{\sum_i \text{FP}_i}{\sum_i (\text{TP}_i + \text{FP}_i)} \quad \text{Micro-R} = \frac{\sum_i \text{FP}_i}{\sum_i (\text{TP}_i + \text{FN}_i)} \quad \text{Micro-F}_1 = \frac{2 \cdot \sum_i \text{TP}_i}{\sum_i (2\text{TP}_i + \text{FP}_i + \text{FN}_i)}$$

- Treats individual predictions equally
- Favors majority classes

Adaptations

1. Micro-Average

- Aggregates at the error level:

$$\text{Micro-P} = \frac{\sum_i \text{FP}_i}{\sum_i (\text{TP}_i + \text{FP}_i)} \quad \text{Micro-R} = \frac{\sum_i \text{FP}_i}{\sum_i (\text{TP}_i + \text{FN}_i)} \quad \text{Micro-F}_1 = \frac{2 \cdot \sum_i \text{TP}_i}{\sum_i (2\text{TP}_i + \text{FP}_i + \text{FN}_i)}$$

- Treats individual predictions equally
- Favors majority classes

2. Macro-Average (weighted)

- Aggregates at the metric level:

$$\text{Macro-P} = \frac{1}{|\mathcal{W}|} \sum_{i=1}^{|\mathcal{W}|} \epsilon_i \cdot P_i \quad \text{Macro-R} = \frac{1}{|\mathcal{W}|} \sum_{i=1}^{|\mathcal{W}|} \epsilon_i \cdot R_i \quad \text{Macro-F}_1 = \frac{1}{|\mathcal{W}|} \sum_{i=1}^{|\mathcal{W}|} \epsilon_i \cdot F_{1i}$$

- General case: $\epsilon_i = |\mathcal{D}_i|/|\mathcal{D}| \Rightarrow$ Without weighting: $\epsilon_i = 1$
- Treats all classes equally ($\epsilon_i = 1$)
- Highlights performance on minority classes ($\epsilon_i = 1$)

Exercise

Expected	Prediction			
	Sunny	Rainy	Windy	Cloudy
Sunny	285	5	5	10
Rainy	2	10	2	1
Windy	0	5	15	10
Cloudy	0	3	2	10

Extensions to other scenarios

- **Ordinal classification:** Typically evaluated as a **regression** task

Extensions to other scenarios

- **Ordinal classification:** Typically evaluated as a **regression** task

- **Multilabel classification:**
 - **Adapted** metrics: Macro-averaging (average per label), micro-averaging (aggregate over labels)
 - **Ad-hoc** metrics: Hamming Loss, Jaccard Index

Outline

① Introduction

Motivation

Relevance of the figure of merit

② General principles

Data partitioning

Cross-validation procedures

③ Classification

Binary case

Multiclass scenario

Other cases

④ Regression

Figures of merit

Introduction

- Continuous target $\Rightarrow \mathcal{W} \subseteq \mathbb{R}$

Introduction

- Continuous target $\Rightarrow \mathcal{W} \subseteq \mathbb{R}$
 - How far the prediction deviates from the expected value

Introduction

- Continuous target $\Rightarrow \mathcal{W} \subseteq \mathbb{R}$
 - How far the prediction deviates from the expected value
- Trained model $\hat{f} : \mathbb{R}^d \rightarrow \mathcal{W}$
- Test dataset $\Rightarrow \mathcal{D}_{test} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}_{test}|}$
 - Feature vector $\mathbf{x} \in \mathbb{R}^d$

Main metrics

1. Mean Absolute Error (MAE)

- Aggregates at the error level:

$$\text{MAE} = \frac{1}{|\mathcal{D}_{test}|} \sum_{i=1}^{|\mathcal{D}_{test}|} |\hat{f}(\mathbf{x}_i) - \omega_i|$$

Main metrics

1. Mean Absolute Error (MAE)

- Aggregates at the **error level**:

$$\text{MAE} = \frac{1}{|\mathcal{D}_{test}|} \sum_{i=1}^{|\mathcal{D}_{test}|} |\hat{f}(\mathbf{x}_i) - \omega_i|$$

2. Mean Squared Error (MSE)

- Aggregates at the **metric level**:

$$\text{MSE} = \frac{1}{|\mathcal{D}_{test}|} \sum_{i=1}^{|\mathcal{D}_{test}|} (\hat{f}(\mathbf{x}_i) - \omega_i)^2$$

T3: Model evaluation

Fundamentos del Aprendizaje Automático

Curso 2025/2026

T4: Nonparametric and distance-based learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Introduction

Contextualization

② Density estimation

Histogram approach

Parzen windows

k_n -Nearest Neighbor estimator

Final remarks

③ The Nearest Neighbor rule

Formulation

Metrics

The k -Nearest-Neighbor rule

④ Other models

Decision tree

Support Vector Machine

Outline

① Introduction

Contextualization

② Density estimation

Histogram approach

Parzen windows

k_n -Nearest Neighbor estimator

Final remarks

③ The Nearest Neighbor rule

Formulation

Metrics

The k -Nearest-Neighbor rule

④ Other models

Decision tree

Support Vector Machine

Bayesian statistical learning

$$P(\omega|x) = \frac{p(x|\omega) \cdot P(\omega)}{p(x)}$$

Bayesian statistical learning

$$P(\omega|x) = \frac{p(x|\omega) \cdot P(\omega)}{p(x)}$$

- **Ideal** case: $p(x|\omega)$ and $P(\omega)$ are known and well defined

Bayesian statistical learning

$$P(\omega|x) = \frac{p(x|\omega) \cdot P(\omega)}{p(x)}$$

- **Ideal** case: $p(x|\omega)$ and $P(\omega)$ are known and well defined
- **Practical** case: $p(x|\omega)$ and $P(\omega)$ are estimated from assortment \mathcal{D}

Bayesian statistical learning

$$P(\omega|x) = \frac{p(x|\omega) \cdot P(\omega)}{p(x)}$$

- **Ideal** case: $p(x|\omega)$ and $P(\omega)$ are known and well defined
- **Practical** case: $p(x|\omega)$ and $P(\omega)$ are estimated from assortment \mathcal{D}

How is it estimated?

Approach	Module	Prior $P(\omega)$	Likelihood $p(x \omega)$	Estimation
			Assumption	
Parametric	T2	Frequentist	Multivariate distribution Statistical independence and univariate distributions	Maximum Likelihood Estimation
Nonparametric	T4	Frequentist	No distribution is assumed	Histogram, Parzen windows, Nearest Neighbor

Nonparametric learning

Two scenarios:

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$
2. Estimate *posteriors* $P(\omega|\mathbf{x})$

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$
 - **Key procedure:** (probability) density estimation
2. Estimate *posteriors* $P(\omega|\mathbf{x})$

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$
 - **Key procedure:** (probability) density estimation
 - Reconstructing the probability **density function** of a random variable
 - Independent of the true distribution
2. Estimate *posteriors* $P(\omega|\mathbf{x})$

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$
 - **Key procedure:** (probability) density estimation
 - Reconstructing the probability **density function** of a random variable
 - Independent of the true distribution
 - The function **approximates** the **true likelihood** $\Rightarrow \hat{p}(\mathbf{x}|\omega)$
2. Estimate *posteriors* $P(\omega|\mathbf{x})$

Nonparametric learning

Two scenarios:

1. Estimate *likelihoods* $p(\mathbf{x}|\omega)$ and *priors* $P(\omega)$
 - **Key procedure:** (probability) density estimation
 - Reconstructing the probability **density function** of a random variable
 - Independent of the true distribution
 - The function **approximates** the **true likelihood** $\Rightarrow \hat{p}(\mathbf{x}|\omega)$
2. Estimate *posteriors* $P(\omega|\mathbf{x})$
 - Avoids *likelihood* and *prior* estimation

Outline

① Introduction

Contextualization

② Density estimation

Histogram approach

Parzen windows

k_n -Nearest Neighbor estimator

Final remarks

③ The Nearest Neighbor rule

Formulation

Metrics

The k -Nearest-Neighbor rule

④ Other models

Decision tree

Support Vector Machine

Formulation

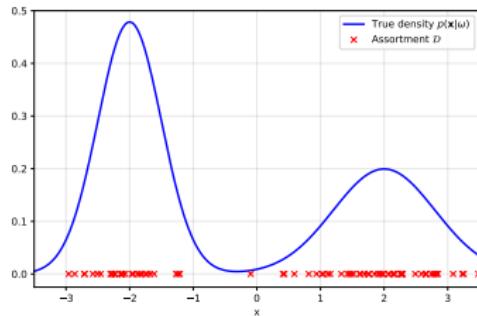
- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$

Formulation

- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$
→ **Goal:** estimate the unknown $p(\mathbf{x}|\omega)$ for each label

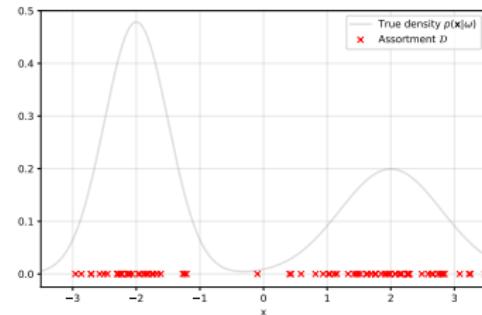
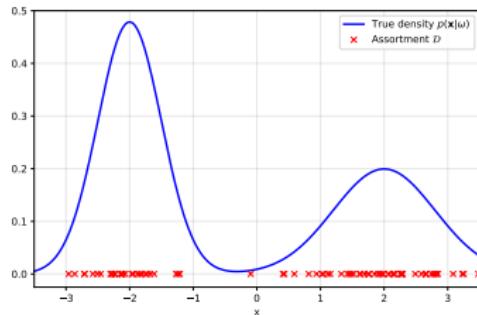
Formulation

- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$
→ Goal: estimate the unknown $p(\mathbf{x}|\omega)$ for each label



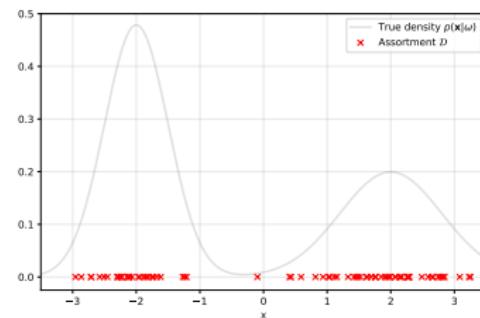
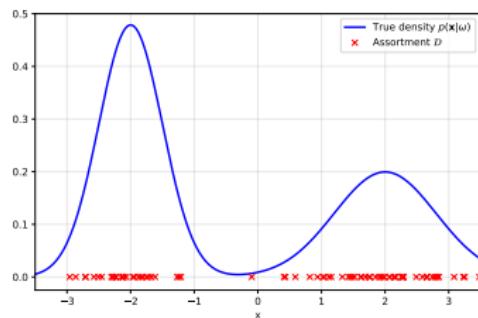
Formulation

- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$
→ Goal: estimate the unknown $p(\mathbf{x}|\omega)$ for each label



Formulation

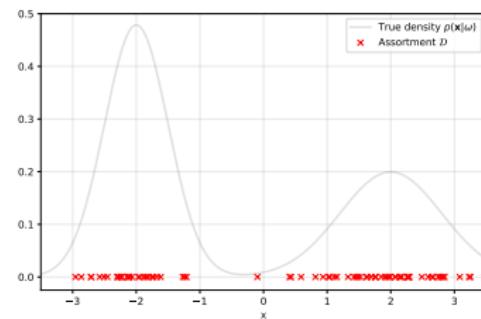
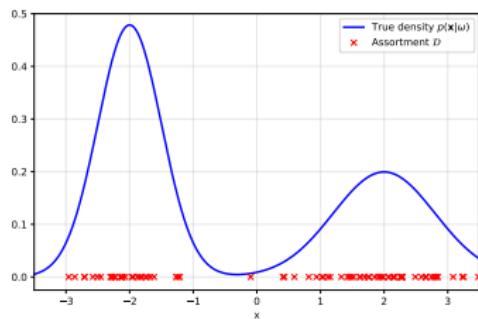
- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$
 - Goal: estimate the unknown $p(\mathbf{x}|\omega)$ for each label



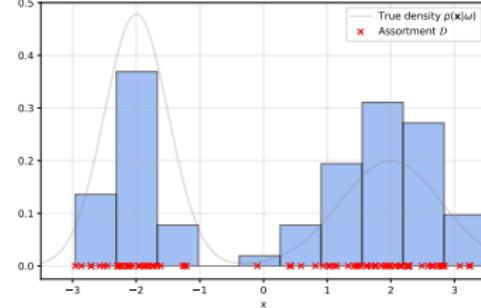
- Histogram estimator:
 - Model $p(\mathbf{x}|\omega)$ as a histogram
 - Process:
 1. Divide data space into regions
 2. Count samples per region
 3. Density as a ratio

Formulation

- Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|} \subset \mathbb{R}^d \times \mathcal{W}$
 - Goal: estimate the unknown $p(\mathbf{x}|\omega)$ for each label



- Histogram estimator:
 - Model $p(\mathbf{x}|\omega)$ as a histogram
 - Process:
 1. Divide data space into regions
 2. Count samples per region
 3. Density as a ratio



Formulation

- Assume a set of regions $\{\mathcal{R}_1, \dots, \mathcal{R}_N\}$ that divide the data space

Formulation

- Assume a set of regions $\{\mathcal{R}_1, \dots, \mathcal{R}_N\}$ that divide the data space
- The probability of $\mathbf{x} \in \mathcal{D}_\omega$ being in region \mathcal{R}_j :

$$P(\mathbf{x} \in \mathcal{R}_j) = \int_{\mathcal{R}_j} p(\mathbf{u}) d\mathbf{u} \approx p(\mathbf{x}|\omega) \cdot V \text{ where } V \in \mathbb{R}^d$$

Formulation

- Assume a set of regions $\{\mathcal{R}_1, \dots, \mathcal{R}_N\}$ that divide the data space
- The probability of $\mathbf{x} \in \mathcal{D}_\omega$ being in region \mathcal{R}_j :

$$P(\mathbf{x} \in \mathcal{R}_j) = \int_{\mathcal{R}_j} p(\mathbf{u}) d\mathbf{u} \approx p(\mathbf{x}|\omega) \cdot V \text{ where } V \in \mathbb{R}^d$$

- Assuming regular sample distribution in data space:

$$P(\mathbf{x} \in \mathcal{R}_j) = \frac{k_j}{|\mathcal{D}|} \text{ where } k_j = |\{i : \mathbf{x}_i \in \mathcal{R}_j\}|$$

Formulation

- Assume a set of regions $\{\mathcal{R}_1, \dots, \mathcal{R}_N\}$ that divide the data space
- The probability of $\mathbf{x} \in \mathcal{D}_\omega$ being in region \mathcal{R}_j :

$$P(\mathbf{x} \in \mathcal{R}_j) = \int_{\mathcal{R}_j} p(\mathbf{u}) d\mathbf{u} \approx p(\mathbf{x}|\omega) \cdot V \text{ where } V \in \mathbb{R}^d$$

- Assuming regular sample distribution in data space:

$$P(\mathbf{x} \in \mathcal{R}_j) = \frac{k_j}{|\mathcal{D}|} \text{ where } k_j = |\{i : \mathbf{x}_i \in \mathcal{R}_j\}|$$

- The density function is approximated as:

$$\hat{p}(\mathbf{x}|\omega) = \frac{k_j}{|\mathcal{D}| \cdot V}$$

Weight feature example

Weight feature example

- Unaccessible male **weight distribution**: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$

Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95] \text{ kg} \Rightarrow V = 5 \text{ kg}$

$$k_{[90,95]} = 23$$

Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95] \text{ kg} \Rightarrow V = 5 \text{ kg}$

$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

Weight feature example

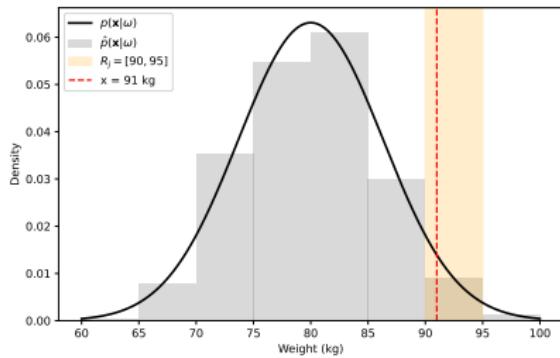
- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$



Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

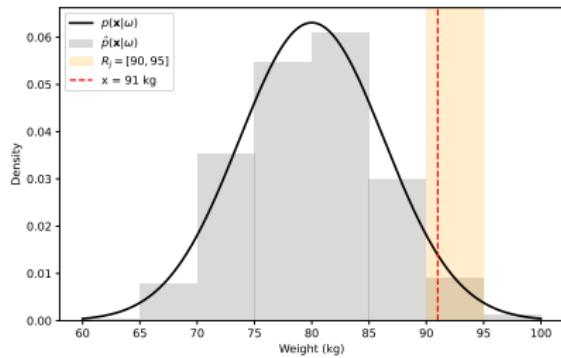
Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

Case #2



Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

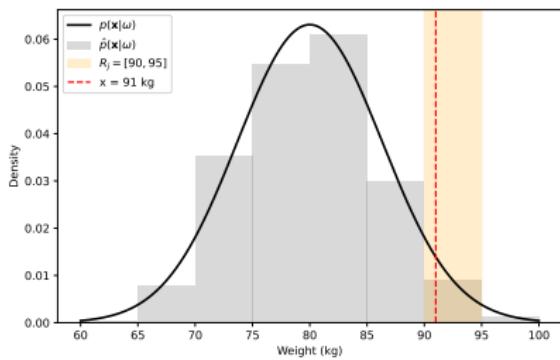
Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$
 $k_{[90,95]} = 23$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

Case #2

Volume range $[90, 92.5]\text{kg} \Rightarrow V = 2.5\text{kg}$



Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

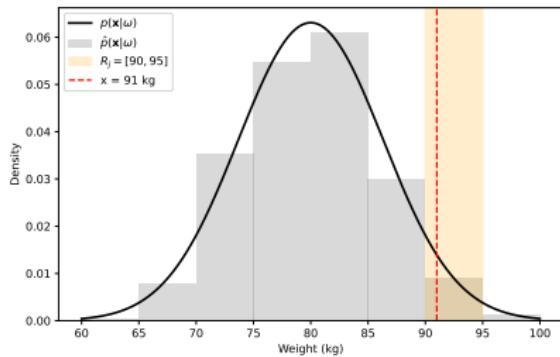
$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

Case #2

Volume range $[90, 92.5]\text{kg} \Rightarrow V = 2.5\text{kg}$

$$k_{[90,92.5]} = 14$$



Weight feature example

- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

$$k_{[90,95]} = 23$$

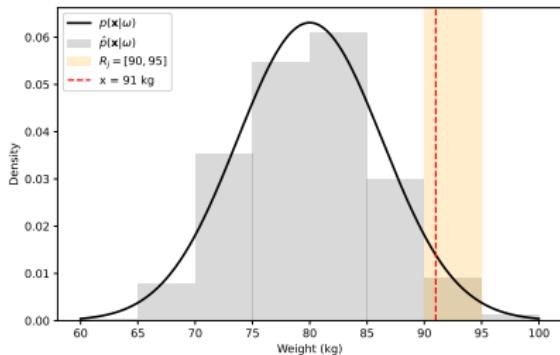
$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

Case #2

Volume range $[90, 92.5]\text{kg} \Rightarrow V = 2.5\text{kg}$

$$k_{[90,92.5]} = 14$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,92.5]}}{|\mathcal{D}| \cdot V} \approx 0.011$$



Weight feature example

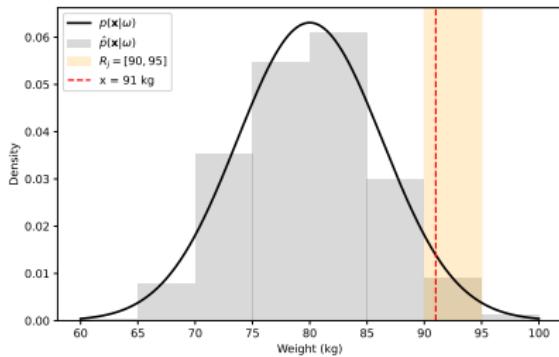
- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$

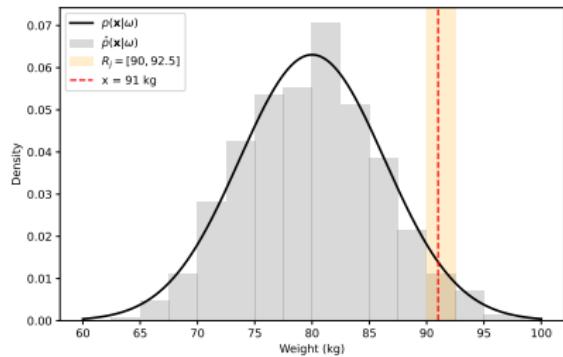


Case #2

Volume range $[90, 92.5]\text{kg} \Rightarrow V = 2.5\text{kg}$

$$k_{[90,92.5]} = 14$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,92.5]}}{|\mathcal{D}| \cdot V} \approx 0.011$$



Weight feature example

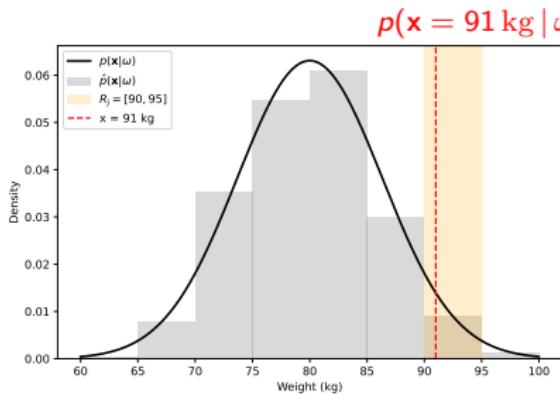
- Unaccessible male weight distribution: $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
- Our real scenario:
 - Assortment $\mathcal{D}_{\text{male}}$ with $|\mathcal{D}_{\text{male}}| = 500$ samples
 - Approximate $p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male})$

Case #1

Volume range $[90, 95]\text{kg} \Rightarrow V = 5\text{kg}$

$$k_{[90,95]} = 23$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,95]}}{|\mathcal{D}| \cdot V} \approx 0.009$$



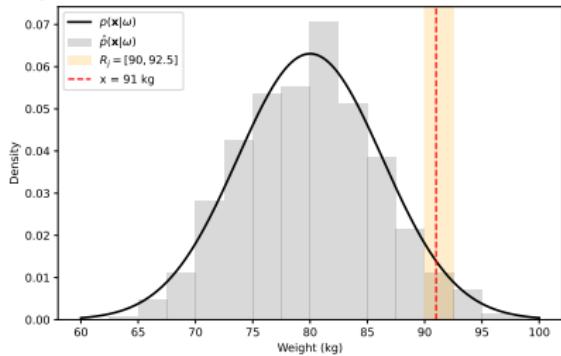
Case #2

Volume range $[90, 92.5]\text{kg} \Rightarrow V = 2.5\text{kg}$

$$k_{[90,92.5]} = 14$$

$$\hat{p}(\mathbf{x} = 91 | \omega) = \frac{k_{[90,92.5]}}{|\mathcal{D}| \cdot V} \approx 0.011$$

$$p(\mathbf{x} = 91 \text{ kg} | \omega = \text{male}) = 0.014$$



Classification example

- Classification task: *male* VS *female*:

Classification example

- Classification task: *male* VS *female*:

$$\rightarrow p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$$

$$\rightarrow p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$$

Classification example

- Classification task: *male* VS *female*:

$$\rightarrow p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$$

$$\rightarrow p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$$

- What we have:

$\rightarrow \mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples

$\rightarrow \mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples

Classification example

- Classification task: *male* VS *female*:
 - $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
 - $p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$
- What we have:
 - $\mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples
 - $\mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples
- What we need: Most likely gender when $\mathbf{x} = 72\text{kg}$

Classification example

- Classification task: *male* VS *female*:
 - $p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$
 - $p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$
- What we have:
 - $\mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples
 - $\mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples
- What we need: Most likely gender when $\mathbf{x} = 72\text{kg}$

Male

Volume $V = 5\text{kg}$

$$k_{[70,75]} = 76$$

$$\hat{p}(72\text{ kg}|\text{male}) = 0.1689$$

$$P(\text{male}) = 450/450+550 = 0.45$$

Classification example

- Classification task: *male* VS *female*:

$$\rightarrow p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$$

$$\rightarrow p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$$

- What we have:

$\rightarrow \mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples

$\rightarrow \mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples

- What we need: Most likely gender when $\mathbf{x} = 72\text{kg}$

Male

Volume $V = 5\text{kg}$

$k_{[70,75]} = 76$

$\hat{p}(72\text{ kg}|\text{male}) = 0.1689$

$P(\text{male}) = 450/450+550 = 0.45$

Female

Volume $V = 5\text{kg}$

$k_{[70,75]} = 3$

$\hat{p}(72\text{ kg}|\text{female}) = 0.001$

$P(\text{female}) = 550/450+550 = 0.55$

Classification example

- Classification task: *male* VS *female*:

$$\rightarrow p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$$

$$\rightarrow p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$$

- What we have:

$\rightarrow \mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples

$\rightarrow \mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples

- What we need: Most likely gender when $\mathbf{x} = 72\text{kg}$

Male

Volume $V = 5\text{kg}$

$k_{[70,75]} = 76$

$\hat{p}(72\text{ kg}|\text{male}) = 0.1689$

$P(\text{male}) = 450/450+550 = 0.45$

Female

Volume $V = 5\text{kg}$

$k_{[70,75]} = 3$

$\hat{p}(72\text{ kg}|\text{female}) = 0.001$

$P(\text{female}) = 550/450+550 = 0.55$

$$\hat{\omega} = \arg \max_{\omega \in \{\text{male, female}\}} \hat{p}(\mathbf{x} = 72|\omega) \cdot P(\omega)$$

Classification example

- Classification task: *male* VS *female*:

$$\rightarrow p(\mathbf{x}|\omega = \text{male}) = \mathcal{N}(80, \sqrt{40})$$

$$\rightarrow p(\mathbf{x}|\omega = \text{female}) = \mathcal{N}(60, \sqrt{20})$$

- What we have:

$\rightarrow \mathcal{D}_{\text{male}}$ assortment with $|\mathcal{D}_{\text{male}}| = 450$ samples

$\rightarrow \mathcal{D}_{\text{female}}$ assortment with $|\mathcal{D}_{\text{female}}| = 550$ samples

- What we need: Most likely gender when $\mathbf{x} = 72\text{kg}$

Male

Volume $V = 5\text{kg}$

$k_{[70,75]} = 76$

$\hat{p}(72\text{ kg}|\text{male}) = 0.1689$

$P(\text{male}) = 450/450+550 = 0.45$

Female

Volume $V = 5\text{kg}$

$k_{[70,75]} = 3$

$\hat{p}(72\text{ kg}|\text{female}) = 0.001$

$P(\text{female}) = 550/450+550 = 0.55$

$$\hat{\omega} = \arg \max_{\omega \in \{\text{male, female}\}} \hat{p}(\mathbf{x} = 72|\omega) \cdot P(\omega) \Rightarrow \text{male}$$

Limitations

- Does reducing the volume always improve the estimation?

Limitations

- Does reducing the volume always **improve** the estimation?
- Remarkably **sensitive** to resolution:
 - **Large** $V \Rightarrow$ oversmoothing / high bias
 - **Small** $V \Rightarrow$ noisy estimate / high variance

Limitations

- Does reducing the volume always **improve** the estimation?
- Remarkably **sensitive** to resolution:
 - **Large** $V \Rightarrow$ oversmoothing / high bias
 - **Small** $V \Rightarrow$ noisy estimate / high variance
- Also **discontinuous**, sensible to bin alignment, and coarse

Formulation

Parzen window estimator: Generalization of the histogram estimator

Formulation

Parzen window estimator: Generalization of the histogram estimator

- Query \mathbf{x} contributes locally to the density around its own position:

$$\hat{p}(\mathbf{x}|\omega) = \frac{k_j}{|\mathcal{D}| \cdot V} \Rightarrow \frac{k_j(\mathbf{x})}{|\mathcal{D}| \cdot V}$$

Formulation

Parzen window estimator: Generalization of the histogram estimator

- Query \mathbf{x} contributes locally to the density around its own position:

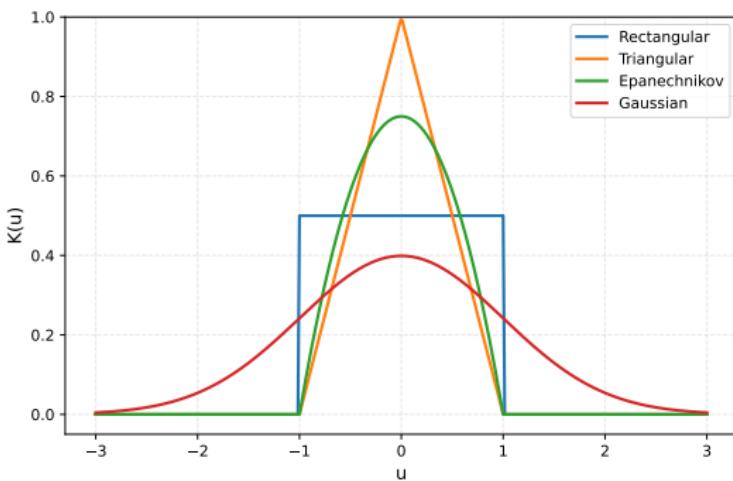
$$\hat{p}(\mathbf{x}|\omega) = \frac{k_j}{|\mathcal{D}| \cdot V} \Rightarrow \frac{k_j(\mathbf{x})}{|\mathcal{D}| \cdot V}$$

- Instead of rigid bins, **kernel function** around \mathbf{x} :

$$\hat{p}(\mathbf{x}|\omega) = \frac{1}{|\mathcal{D}| \cdot h^d} \sum_{i=1}^{|\mathcal{D}|} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right)$$

$h \rightarrow$ window width

Window shapes



$$K_{\text{rec}}(u) = \begin{cases} 1/2 & |u| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$K_{\text{epa}}(u) = \begin{cases} \frac{3}{4}(1-u^2) & |u| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$K_{\text{tri}}(u) = \begin{cases} 1 - |u| & |u| \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

$$K_{\text{gau}}(u) = \frac{1}{\sqrt{2\pi}} e^{-u^2/2}$$

Formulation

Issues

- **Main limitation:** **fixed** window width h regardless of x

Issues

- **Main limitation:** **fixed** window width h regardless of x
- Same neighborhood size regardless of (local) data density/sparsity:
 - High density \Rightarrow too many samples \Rightarrow over-smoothed estimation
 - Low density \Rightarrow too few samples \Rightarrow under-smoothed estimation

Formulation

k_n -Nearest Neighbor estimator: Addresses the limitations in Parzen

Formulation

k_n -Nearest Neighbor estimator: Addresses the limitations in Parzen

- ✗ Does not fix window width h
- ✓ Fixes number of samples k and lets the window volume vary

$$\hat{p}(\mathbf{x}|\omega) = \frac{k_j}{|\mathcal{D}| \cdot V} \Rightarrow \frac{k}{|\mathcal{D}| \cdot V(\mathbf{x})}$$

Formulation

Posterior probability

Posterior probability

- Assortment \mathcal{D} from an unknown distribution
→ Subset $\mathcal{D}_\omega = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = \omega\}_{i=1}^{|\mathcal{D}|}$ with class ω

Posterior probability

- Assortment \mathcal{D} from an unknown distribution
 - Subset $\mathcal{D}_\omega = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = \omega\}_{i=1}^{|\mathcal{D}|}$ with class ω
- Estimation using k neighbors
 - k_ω : number of neighbors with class ω

Posterior probability

- Assortment \mathcal{D} from an unknown distribution
 - Subset $\mathcal{D}_\omega = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = \omega\}_{i=1}^{|\mathcal{D}|}$ with class ω
- Estimation using k neighbors
 - k_ω : number of neighbors with class ω

$$\hat{P}(\omega | \mathbf{x}) = \frac{\hat{p}(\mathbf{x} | \omega) \cdot P(\omega)}{\hat{p}(\mathbf{x})}$$

Posterior probability

- Assortment \mathcal{D} from an unknown distribution
 - Subset $\mathcal{D}_\omega = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = \omega\}_{i=1}^{|\mathcal{D}|}$ with class ω
- Estimation using k neighbors
 - k_ω : number of neighbors with class ω

$$\hat{P}(\omega | \mathbf{x}) = \frac{\hat{p}(\mathbf{x} | \omega) \cdot P(\omega)}{\hat{p}(\mathbf{x})} \approx \frac{\frac{k_\omega / |\mathcal{D}_\omega|}{V(\mathbf{x})} \cdot \frac{|\mathcal{D}_\omega|}{|\mathcal{D}|}}{\frac{k / |\mathcal{D}|}{V(\mathbf{x})}}$$

Posterior probability

- Assortment \mathcal{D} from an unknown distribution
 - Subset $\mathcal{D}_\omega = \{(\mathbf{x}_i, y_i) \in \mathcal{D} : y_i = \omega\}_{i=1}^{|\mathcal{D}|}$ with class ω
- Estimation using k neighbors
 - k_ω : number of neighbors with class ω

$$\hat{P}(\omega | \mathbf{x}) = \frac{\hat{p}(\mathbf{x} | \omega) \cdot P(\omega)}{\hat{p}(\mathbf{x})} \approx \frac{\frac{k_\omega / |\mathcal{D}_\omega|}{V(\mathbf{x})} \cdot \frac{|\mathcal{D}_\omega|}{|\mathcal{D}|}}{\frac{k / |\mathcal{D}|}{V(\mathbf{x})}} = \frac{k_\omega}{k}$$

Issues

- Hyperparameter k remarkably influences the estimation:
 - $k \downarrow$: noisy estimate
 - $k \uparrow$: oversmoothed estimate
- Computational inefficiency
- Lack of smoothness / discontinuity
 - Piecewise estimate

Comparative summary of the density estimators

Estimator	Volume size	Number of points	Advantages	Disadvantages
Histogram	Fixed (bins)	Variable	Simple	Discontinuous
Parzen	Fixed (width h)	Variable	Smooth	Not adaptive
k_n -NN	Variable	Fixed	Adaptive	Not smooth, costly

Outline

① Introduction

Contextualization

② Density estimation

Histogram approach

Parzen windows

k_n -Nearest Neighbor estimator

Final remarks

③ The Nearest Neighbor rule

Formulation

Metrics

The k -Nearest-Neighbor rule

④ Other models

Decision tree

Support Vector Machine

Formulation

Formulation

Given a query $q = (\mathbf{x}_q, \omega_q)$, the **NN rule** assigns the **label** of the closest sample in the training set \mathcal{T} :

Formulation

Given a query $q = (\mathbf{x}_q, \omega_q)$, the **NN rule** assigns the **label** of the closest sample in the training set \mathcal{T} :

$$\hat{\omega}_q = \omega_i : \arg \min_{1 \leq i \leq |\mathcal{T}|} D(\mathbf{x}_q, \mathbf{x}_i)$$

where $D : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$

Formulation

Given a query $q = (\mathbf{x}_q, \omega_q)$, the **NN rule** assigns the **label** of the closest sample in the training set \mathcal{T} :

$$\hat{\omega}_q = \omega_i : \arg \min_{1 \leq i \leq |\mathcal{T}|} D(\mathbf{x}_q, \mathbf{x}_i)$$

where $D : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$

Main **features**:

- Only requires the definition of a **(dis)similarity measure**:
 - ✓ Useful for **feature-based** and **structural** representations

Formulation

Given a query $q = (\mathbf{x}_q, \omega_q)$, the **NN rule** assigns the **label** of the closest sample in the training set \mathcal{T} :

$$\hat{\omega}_q = \omega_i : \arg \min_{1 \leq i \leq |\mathcal{T}|} D(\mathbf{x}_q, \mathbf{x}_i)$$

where $D : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$

Main **features**:

- Only requires the definition of a **(dis)similarity measure**:
 - ✓ Useful for **feature-based** and **structural** representations
- Does **not derive a model** out of the \mathcal{T} :
 - ✓ Adaptive
 - ✗ Inefficient as $|\mathcal{T}| \uparrow\uparrow$

Voronoi tessellation

Division of the feature space into regions

- Any point in that falls in a region is closer to the sample that defines it than to any other point/region

Voronoi tessellation

Division of the feature space into regions

- Any point in that falls in a region is closer to the sample that defines it than to any other point/region

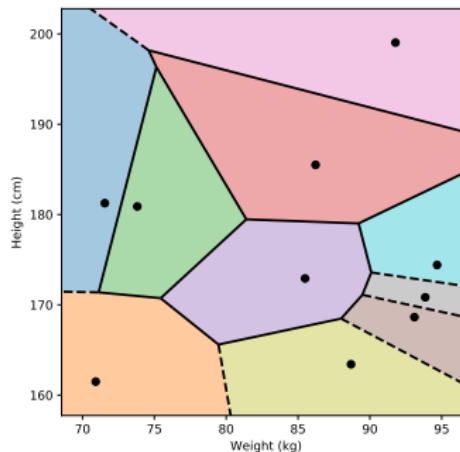
$$V_i = \left\{ \mathbf{x} \in \mathbb{R}^d : D(\mathbf{x}, \mathbf{x}_i) \leq D(\mathbf{x}, \mathbf{x}_j), \forall j \neq i \right\}$$

Voronoi tessellation

Division of the feature space into regions

- Any point in that falls in a region is closer to the sample that defines it than to any other point/region

$$V_i = \left\{ \mathbf{x} \in \mathbb{R}^d : D(\mathbf{x}, \mathbf{x}_i) \leq D(\mathbf{x}, \mathbf{x}_j), \forall j \neq i \right\}$$



Error rate and bounds

Assume a **binary classification case** $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Which is the **probability of error** at \mathbf{x} for **NN**?

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Which is the **probability of error** at \mathbf{x} for **NN**?

→ Assume \mathbf{x}_N is the **closest neighbor** to \mathbf{x}

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Which is the **probability of error** at \mathbf{x} for **NN**?

→ Assume \mathbf{x}_N is the **closest neighbor** to \mathbf{x}

$$R_{\text{NN}}(\mathbf{x}) = P(\omega \neq \hat{\omega}|\mathbf{x})$$

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Which is the **probability of error** at \mathbf{x} for **NN**?

→ Assume \mathbf{x}_N is the **closest neighbor** to \mathbf{x}

$$R_{\text{NN}}(\mathbf{x}) = P(\omega \neq \hat{\omega}|\mathbf{x}) = P(\omega = \omega_1, \hat{\omega} = \omega_2|\mathbf{x}) + P(\omega = \omega_2, \hat{\omega} = \omega_1|\mathbf{x})$$

Error rate and bounds

Assume a **binary classification** case $\Rightarrow \mathcal{W} = \{\omega_1, \omega_2\}$

- Bayes risk at \mathbf{x} : $R^*(\mathbf{x}) = \min \left[P(\omega_1|\mathbf{x}), \underbrace{P(\omega_2|\mathbf{x})}_{1-P(\omega_1|\mathbf{x})} \right]$
- Bayes risk: $\int_{-\infty}^{\infty} R^*(\mathbf{x}) p(\mathbf{x}) d\mathbf{x}$

Which is the **probability of error** at \mathbf{x} for **NN**?

→ Assume \mathbf{x}_N is the **closest neighbor** to \mathbf{x}

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega \neq \hat{\omega}|\mathbf{x}) = P(\omega = \omega_1, \hat{\omega} = \omega_2|\mathbf{x}) + P(\omega = \omega_2, \hat{\omega} = \omega_1|\mathbf{x}) = \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \end{aligned}$$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i | \mathbf{x}_N) \approx P(\omega_i | \mathbf{x})$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i|\mathbf{x}_N) \approx P(\omega_i|\mathbf{x})$

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \approx \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}) \end{aligned}$$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i|\mathbf{x}_N) \approx P(\omega_i|\mathbf{x})$

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \approx \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}) = \\ &= 2 \cdot P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) = 2 \cdot P(\omega_1|\mathbf{x}) [1 - P(\omega_1|\mathbf{x})] \end{aligned}$$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i|\mathbf{x}_N) \approx P(\omega_i|\mathbf{x})$

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \approx \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}) = \\ &= 2 \cdot P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) = 2 \cdot P(\omega_1|\mathbf{x}) [1 - P(\omega_1|\mathbf{x})] = \\ &= 2 \cdot R^*(\mathbf{x}) [1 - R^*(\mathbf{x})] \end{aligned}$$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i|\mathbf{x}_N) \approx P(\omega_i|\mathbf{x})$

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \approx \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}) = \\ &= 2 \cdot P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) = 2 \cdot P(\omega_1|\mathbf{x}) [1 - P(\omega_1|\mathbf{x})] = \\ &= 2 \cdot R^*(\mathbf{x}) [1 - R^*(\mathbf{x})] \end{aligned}$$

Error bound for NN (binary)

$$R^* \leq R_{\text{NN}} \leq 2R^*(1 - R^*)$$

Error rate and bounds

If $|\mathcal{T}| \rightarrow \infty \Rightarrow P(\omega_i|\mathbf{x}_N) \approx P(\omega_i|\mathbf{x})$

$$\begin{aligned} R_{\text{NN}}(\mathbf{x}) &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}_N) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}_N) \approx \\ &= P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) + P(\omega_2|\mathbf{x}) \cdot P(\omega_1|\mathbf{x}) = \\ &= 2 \cdot P(\omega_1|\mathbf{x}) \cdot P(\omega_2|\mathbf{x}) = 2 \cdot P(\omega_1|\mathbf{x}) [1 - P(\omega_1|\mathbf{x})] = \\ &= 2 \cdot R^*(\mathbf{x}) [1 - R^*(\mathbf{x})] \end{aligned}$$

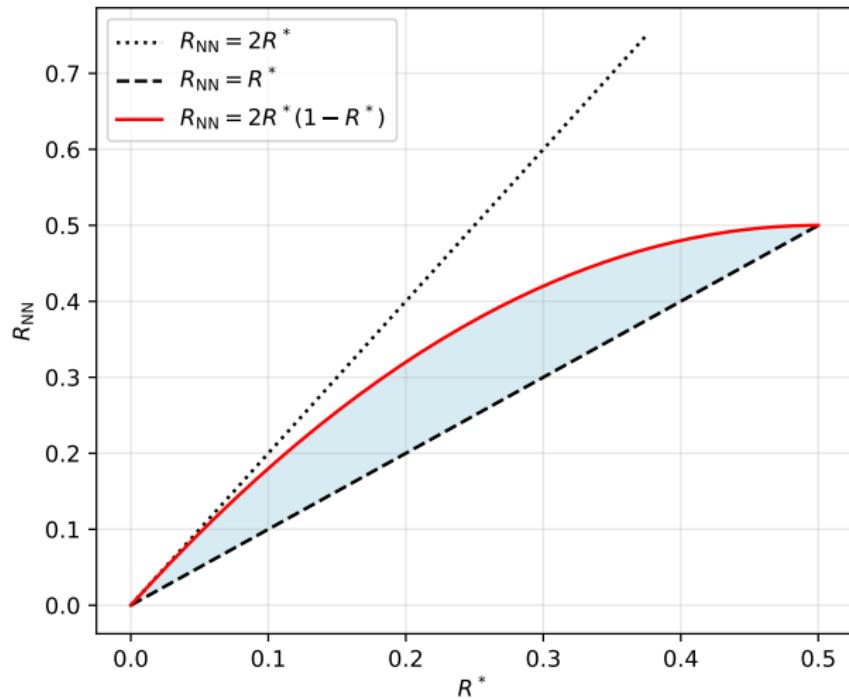
Error bound for NN (binary)

$$R^* \leq R_{\text{NN}} \leq 2R^*(1 - R^*)$$

Error bound for NN (multiclass)

$$R^* \leq R_{\text{NN}} \leq R^* \left(2 - \frac{c}{c-1} R^* \right)$$

Error rate and bounds



Concept of metric

Concept of metric

- NN relies on a **metric** or **distance function** $D(\cdot, \cdot)$
⇒ Gives a generalized scalar distance between two arguments

Concept of metric

- NN relies on a **metric** or **distance function** $D(\cdot, \cdot)$
 - ⇒ Gives a generalized scalar distance between two arguments
- A **metric** must follow **four properties**:
 - ① Nonnegativity: $D(\mathbf{a}, \mathbf{b}) \geq 0$
 - ② Reflexivity: $D(\mathbf{a}, \mathbf{b}) = 0$ iif $\mathbf{a} = \mathbf{b}$
 - ③ Symmetry: $D(\mathbf{a}, \mathbf{b}) = D(\mathbf{b}, \mathbf{a})$
 - ④ Triangle inequality: $D(\mathbf{a}, \mathbf{b}) + D(\mathbf{b}, \mathbf{c}) \geq D(\mathbf{a}, \mathbf{c})$

Minkowski distance

Generalized metric that **unifies** most well-known **distance measures**:

Minkowski distance

Generalized metric that **unifies** most well-known **distance measures**:

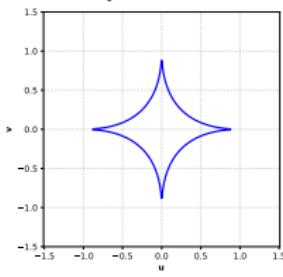
$$D(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{|a|} |a_i - b_i|^p \right)^{\frac{1}{p}} \text{ with } p \geq 1$$

Minkowski distance

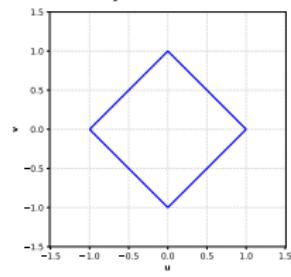
Generalized metric that **unifies** most well-known **distance measures**:

$$D(\mathbf{a}, \mathbf{b}) = \left(\sum_{i=1}^{|a|} |a_i - b_i|^p \right)^{\frac{1}{p}} \quad \text{with } p \geq 1$$

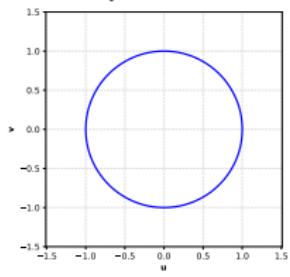
$p = 0.5$



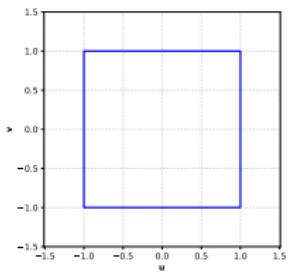
$p = 1$



$p = 2$



$p = \infty$



Formulation

Generalization of the NN rule \Rightarrow considers the k nearest elements

Formulation

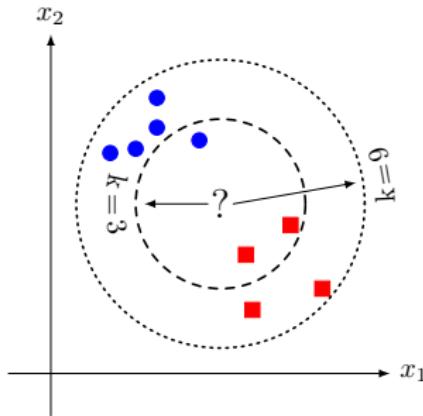
Generalization of the NN rule \Rightarrow considers the k nearest elements

- Estimated class $\hat{\omega} \Rightarrow$ The most frequent label among the k neighbors

Formulation

Generalization of the NN rule \Rightarrow considers the k nearest elements

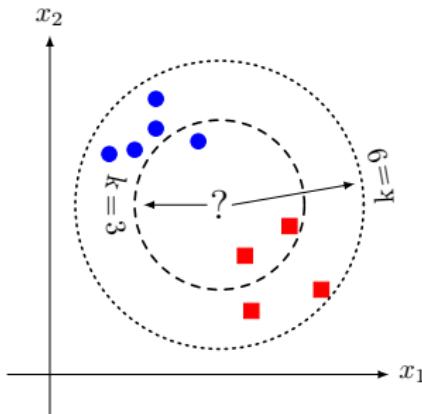
- Estimated class $\hat{\omega} \Rightarrow$ The most frequent label among the k neighbors



Formulation

Generalization of the NN rule \Rightarrow considers the k nearest elements

- Estimated class $\hat{\omega} \Rightarrow$ The most frequent label among the k neighbors



Hyperparameter k :

- Low k values: Sensitive to noise (high variance)
- High k values: Too smooth/general (high bias)

Error rates

k NN with $k > 1$ trades bias for lower variance

Error rates

k NN with $k > 1$ trades bias for lower variance

→ Estimated risk ($R_{k\text{NN}}$) tends to R^*

Error rates

k NN with $k > 1$ trades bias for lower variance

→ Estimated risk ($R_{k\text{NN}}$) tends to R^*

Error bound for kNN (multiclass)

$$R^* \leq R_{k\text{NN}} \leq R_{\text{NN}} \leq R^* \left(2 - \frac{c}{c-1} R^* \right)$$

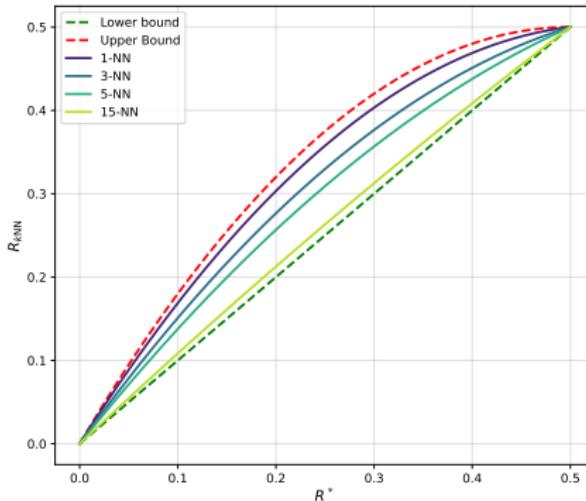
Error rates

k NN with $k > 1$ trades bias for lower variance

→ Estimated risk ($R_{k\text{NN}}$) tends to R^*

Error bound for kNN (multiclass)

$$R^* \leq R_{k\text{NN}} \leq R_{\text{NN}} \leq R^* \left(2 - \frac{c}{c-1} R^* \right)$$



Decision boundaries

Practicalities

Practicalities

- ✓ Interactive learning / fast adaptation
 - Model is updated by simply adding new elements

Practicalities

- ✓ Interactive learning / fast adaptation
 - Model is updated by simply adding new elements
- ✓ Good error properties
 - Bounded by twice the Bayes risk

Practicalities

- ✓ Interactive learning / fast adaptation
 - Model is updated by simply adding new elements
- ✓ Good error properties
 - Bounded by twice the Bayes risk
- ✓ Tasks beyond classification and regression
 - Search of elements in databases

Practicalities

- ✓ Interactive learning / fast adaptation
 - Model is updated by simply adding new elements
- ✓ Good error properties
 - Bounded by twice the Bayes risk
- ✓ Tasks beyond classification and regression
 - Search of elements in databases
- ✗ Features may depict different ranges
 - Dominant feature that eclipses the others ⇒ Normalization

Practicalities

- ✓ Interactive learning / fast adaptation
 - Model is updated by simply adding new elements
- ✓ Good error properties
 - Bounded by twice the Bayes risk
- ✓ Tasks beyond classification and regression
 - Search of elements in databases
- ✗ Features may depict different ranges
 - Dominant feature that eclipses the others ⇒ Normalization
- ✗ Inefficient classifier $\Rightarrow \mathcal{O}(n^2)$
 1. Fast similarity search
 2. Approximate search
 3. Data reduction

Outline

① Introduction

Contextualization

② Density estimation

Histogram approach

Parzen windows

k_n -Nearest Neighbor estimator

Final remarks

③ The Nearest Neighbor rule

Formulation

Metrics

The k -Nearest-Neighbor rule

④ Other models

Decision tree

Support Vector Machine

Contextualization

- **k -Nearest Neighbor:** relevant example of nonparametric learning
⇒ But there exist other types of models

Contextualization

- **k -Nearest Neighbor:** relevant example of nonparametric learning
⇒ But there exist other types of models
- Each model has its own idiosyncratic way of dividing the label space:
⇒ Different types of decision boundaries

Contextualization

- ***k*-Nearest Neighbor:** relevant example of nonparametric learning
 - ⇒ But there exist other types of models
- Each model has its own idiosyncratic way of dividing the label space:
 - ⇒ Different types of decision boundaries
- Two examples of models:
 1. Decision tree
 2. Support Vector Machine

Decision tree

Decision tree

- Derives a **tree structure**:
 - **Nodes** evaluate the **value** of the feature
 - **Leaves** denote the **labels**

Decision tree

- Derives a **tree structure**:
 - Nodes evaluate the **value** of the feature
 - Leaves denote the **labels**
- Typically constructed following **information gain** principles

Decision tree

- Derives a **tree structure**:
 - Nodes evaluate the **value** of the feature
 - Leaves denote the **labels**
- Typically constructed following **information gain** principles
- Tend to **overfit**:
 - ✗ Pruning strategies to improve generalization
 - ✓ Useful for **outlier detection**

Decision tree

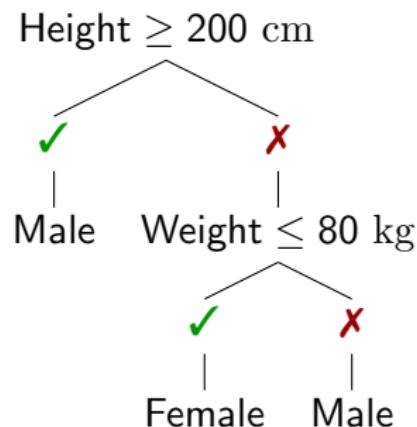
Example:

- Binary task: $\mathcal{W} = \{\text{male, female}\}$
- Feature space \mathbb{R}^2 : weight (kg), height (cm)

Decision tree

Example:

- Binary task: $\mathcal{W} = \{\text{male}, \text{female}\}$
- Feature space \mathbb{R}^2 : weight (kg), height (cm)



Support Vector Machine

Support Vector Machine

- Linear binary classifier that uses a **hyperplane** to separate the labels
→ Support vectors: elements that **define** the hyperplane

Support Vector Machine

- Linear binary classifier that uses a **hyperplane** to separate the labels
 - Support vectors: elements that **define** the hyperplane
- Kernel: maps the data into a **higher dimensionality** space

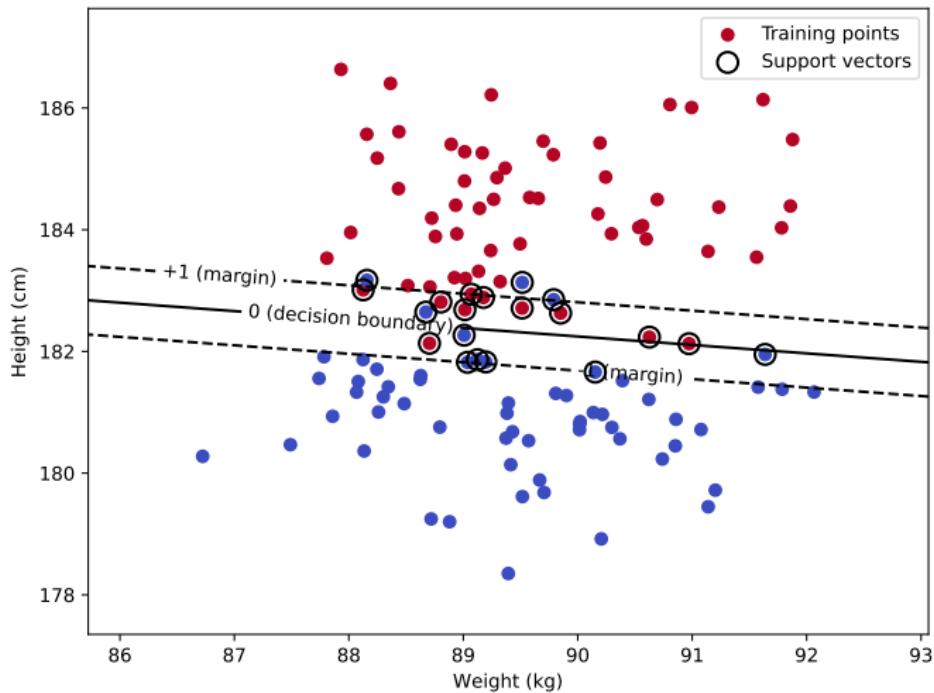
Support Vector Machine

- Linear binary classifier that uses a **hyperplane** to separate the labels
 - Support vectors: elements that **define** the hyperplane
- Kernel: maps the data into a **higher dimensionality** space
- Distinct classes must be in **different sides** of the hyperplane
 - Premise is relaxed via **penalties**

Support Vector Machine

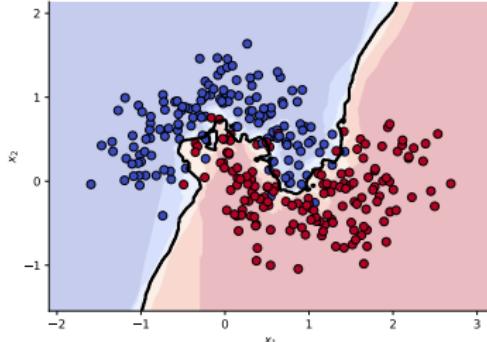
- Linear binary classifier that uses a **hyperplane** to separate the labels
 - Support vectors: elements that **define** the hyperplane
- Kernel: maps the data into a **higher dimensionality** space
- Distinct classes must be in **different sides** of the hyperplane
 - Premise is relaxed via **penalties**
- Adaptations for **multiclass** scenarios
 - One-vs-all configuration
 - One-vs-one configuration

Support Vector Machine

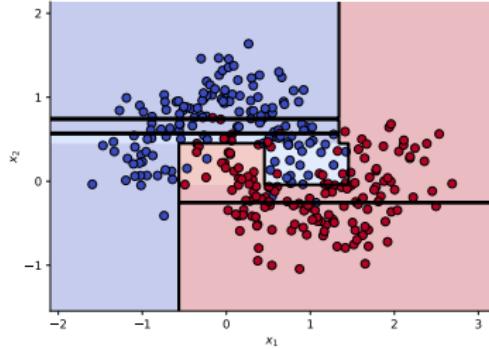


Decision boundaries (comparison)

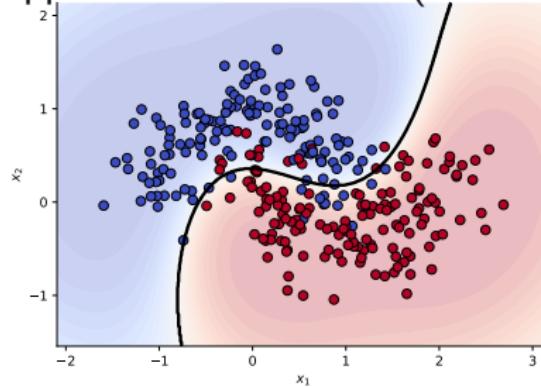
kNN with $k = 5$



Decision Tree with 5 levels



Support Vector Machine (RBF kernel)



T4: Nonparametric and distance-based learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

T5: Linear methods and Perceptron

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Linear models

Binary

Multiclass

Parameter estimation

② Perceptron

Introduction

Training

Limitations

Multiclass Perceptron

③ Multi-layer Perceptron

Introduction

Structure

Training

Training dynamics and regularization

Problem statement

- So far:

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \theta)$

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \theta)$

T4: Nonparametric density estimator for approximating *likelihood*

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \theta)$

T4: Nonparametric density estimator for approximating *likelihood*

T4: Distance-based classification for *posterior* $\hat{P}(\omega|\mathbf{x})$

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \theta)$

T4: Nonparametric density estimator for approximating *likelihood*

T4: Distance-based classification for *posterior* $\hat{P}(\omega|\mathbf{x})$

- **T5:** Parametric model for *discriminant functions* $\Rightarrow g(\mathbf{x}; \theta)$

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \boldsymbol{\theta})$
T4: Nonparametric density estimator for approximating *likelihood*
T4: Distance-based classification for *posterior* $\hat{P}(\omega|\mathbf{x})$

- **T5:** Parametric model for discriminant functions $\Rightarrow g(\mathbf{x}; \boldsymbol{\theta})$
→ $\boldsymbol{\theta}$ ⇒ weights of the model

Problem statement

- So far:

T2: Parametric models for modeling *likelihood*: $P(\mathbf{x}|\omega) \approx \hat{P}(\mathbf{x}|\omega, \boldsymbol{\theta})$
T4: Nonparametric density estimator for approximating *likelihood*
T4: Distance-based classification for *posterior* $\hat{P}(\omega|\mathbf{x})$

- **T5:** Parametric model for *discriminant functions* $\Rightarrow g(\mathbf{x}; \boldsymbol{\theta})$
 - $\boldsymbol{\theta}$ ⇒ *weights* of the model
 - $J(\boldsymbol{\theta})$ ⇒ *criterion function* to estimate $\boldsymbol{\theta}$

Outline

1 Linear models

Binary

Multiclass

Parameter estimation

2 Perceptron

Introduction

Training

Limitations

Multiclass Perceptron

3 Multi-layer Perceptron

Introduction

Structure

Training

Training dynamics and regularization

Linear discriminant function

Linear discriminant function

- Simplest type of ML model

Linear discriminant function

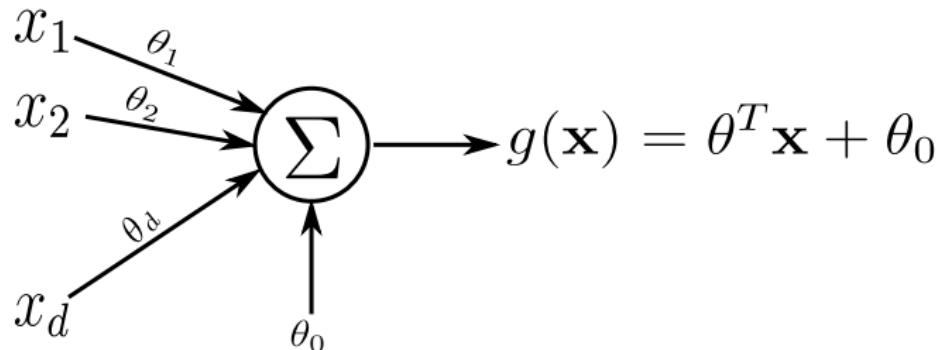
- Simplest type of ML model
- **Assumption:** output is a linear combination of the input features

Linear discriminant function

- Simplest type of ML model
- Assumption: output is a linear combination of the input features
- Two-category case ($\mathcal{W} = \{\omega_1, \omega_2\}$):

$$g(\mathbf{x}) = \boldsymbol{\theta}^T \mathbf{x} + \theta_0$$

$\mathbf{x} \in \mathbb{R}^d$ feature vector
 $\boldsymbol{\theta} \in \mathbb{R}^d$ weight vector
 $\theta_0 \in \mathbb{R}$ weight threshold



Decision surface

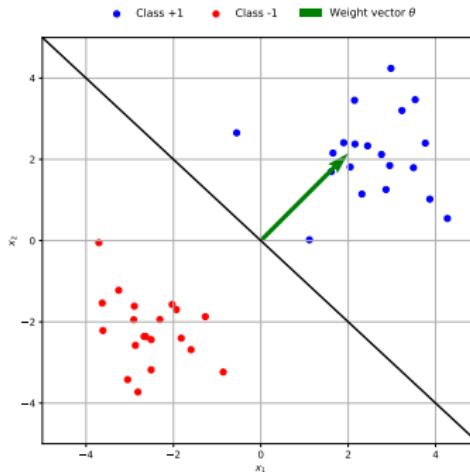
- Decision surface (\mathcal{H}): Given by equation $g(\mathbf{x}) = 0$

Decision surface

- Decision surface (\mathcal{H}): Given by equation $g(\mathbf{x}) = 0$
 - If $g(\mathbf{x}) > 0$: Region $\mathcal{R}_1 \Rightarrow \hat{\omega} = \omega_1$
 - If $g(\mathbf{x}) < 0$: Region $\mathcal{R}_2 \Rightarrow \hat{\omega} = \omega_2$

Decision surface

- Decision surface (\mathcal{H}): Given by equation $g(\mathbf{x}) = 0$
 - If $g(\mathbf{x}) > 0$: Region $\mathcal{R}_1 \Rightarrow \hat{\omega} = \omega_1$
 - If $g(\mathbf{x}) < 0$: Region $\mathcal{R}_2 \Rightarrow \hat{\omega} = \omega_2$
- Geometric interpretation:
 - Vector θ : normal to surface \mathcal{H} and defines its *orientation*
 - Bias θ_0 : Shifts surface \mathcal{H} along θ



Decision surface

Formalization

- Multiclass: Generalizes to **one** linear function **per label**:

$$g_k(\mathbf{x}) = \boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k}, \quad k = 1, 2, \dots, |\mathcal{W}|$$

Formalization

- Multiclass: Generalizes to one linear function per label:

$$g_k(\mathbf{x}) = \boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k}, \quad k = 1, 2, \dots, |\mathcal{W}|$$

- Decision rule:

$$\hat{\omega} = \arg \max_{k=1, \dots, |\mathcal{W}|} g_k(\mathbf{x}) = \arg \max_{k=1, \dots, |\mathcal{W}|} \left(\boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k} \right)$$

Formalization

- Multiclass: Generalizes to one linear function per label:

$$g_k(\mathbf{x}) = \boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k}, \quad k = 1, 2, \dots, |\mathcal{W}|$$

- Decision rule:

$$\hat{\omega} = \arg \max_{k=1, \dots, |\mathcal{W}|} g_k(\mathbf{x}) = \arg \max_{k=1, \dots, |\mathcal{W}|} \left(\boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k} \right)$$

- Decision surface(s):

→ Each tuple $(\boldsymbol{\theta}_k^T, \theta_{0k})$ defines a hyperplane

Formalization

- Multiclass: Generalizes to one linear function per label:

$$g_k(\mathbf{x}) = \boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k}, \quad k = 1, 2, \dots, |\mathcal{W}|$$

- Decision rule:

$$\hat{\omega} = \arg \max_{k=1, \dots, |\mathcal{W}|} g_k(\mathbf{x}) = \arg \max_{k=1, \dots, |\mathcal{W}|} (\boldsymbol{\theta}_k^T \mathbf{x} + \theta_{0k})$$

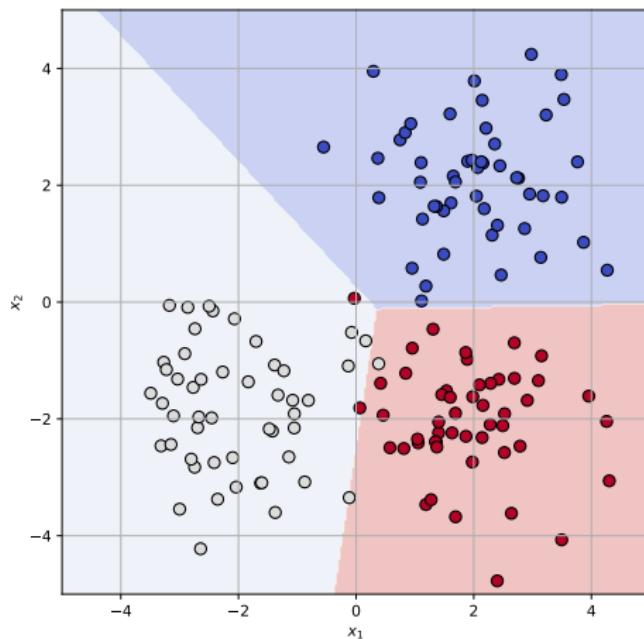
- Decision surface(s):

→ Each tuple $(\boldsymbol{\theta}_k^T, \theta_{0k})$ defines a hyperplane

- Decision boundaries: Pairs $\langle i, j \rangle \in \{1, \dots, |\mathcal{W}|\}$ have equal scores:

$$g_i(\mathbf{x}) = g_j(\mathbf{x}) \Rightarrow (\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)^T \mathbf{x} + (\theta_{0i} - \theta_{0j}) = 0$$

Decision frontiers



Introduction

- Consider a reference set of samples $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$

Introduction

- Consider a reference set of samples $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- **Goal:** Estimate weights $\boldsymbol{\theta}$ in discriminant function $g(\mathbf{x}; \boldsymbol{\theta}) = \boldsymbol{\theta}^T \mathbf{x}$

Introduction

- Consider a reference set of samples $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- **Goal:** Estimate weights θ in discriminant function $g(\mathbf{x}; \theta) = \theta^T \mathbf{x}$
→ Minimize discrepancy between predictions and ground truth (ω) on \mathcal{D} :

$$\min_{\theta} J(\theta) = \min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}(g(\mathbf{x}_i; \theta), \omega_i)$$

Introduction

- Consider a reference set of samples $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- **Goal:** Estimate weights θ in discriminant function $g(\mathbf{x}; \theta) = \theta^T \mathbf{x}$
 - Minimize discrepancy between predictions and ground truth (ω) on \mathcal{D} :

$$\min_{\theta} J(\theta) = \min_{\theta} \frac{1}{|\mathcal{D}|} \sum_{i=1}^{|\mathcal{D}|} \mathcal{L}(g(\mathbf{x}_i; \theta), \omega_i)$$

- This minimization process may be solved analytically in some cases
 - Practical scenarios: iterative optimization process

Gradient descent

Gradient descent algorithm

$\theta_0 = \text{arbitrary}$

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
→ Typical values: constant, $\eta(k) = \eta(0)/k$

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined
 2. Vector θ_0 is arbitrarily chosen

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined
 2. Vector θ_0 is arbitrarily chosen
 3. Gradient vector $\nabla J(\theta^{(0)})$ is computed

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined
 2. Vector θ_0 is arbitrarily chosen
 3. Gradient vector $\nabla J(\theta^{(0)})$ is computed
 4. Value $\theta^{(1)}$: moving $\propto \eta(k)$ from $\theta^{(0)}$ in the steepest descent

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined
 2. Vector θ_0 is arbitrarily chosen
 3. Gradient vector $\nabla J(\theta^{(0)})$ is computed
 4. Value $\theta^{(1)}$: moving $\propto \eta(k)$ from $\theta^{(0)}$ in the steepest descent
 5. Repeat from (3) until convergence criterion is achieved

Gradient descent

Gradient descent algorithm

θ_0 = arbitrary

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \nabla J(\theta^{(k)})$$

- Learning rate $\eta(k)$: controls the amount of change at each iteration
 - Typical values: constant, $\eta(k) = \eta(0)/k$
- Process description:
 1. Criterion function $J(\theta)$ is defined
 2. Vector θ_0 is arbitrarily chosen
 3. Gradient vector $\nabla J(\theta^{(0)})$ is computed
 4. Value $\theta^{(1)}$: moving $\propto \eta(k)$ from $\theta^{(0)}$ in the steepest descent
 5. Repeat from (3) until convergence criterion is achieved

→ Example: $|\eta(k) \nabla J(\theta^{(k)})| < \delta$

Gradient descent

Example:

- Function $J(\theta) = (\theta - 3)^2$
- Initial value $\theta_0 = -2$
- Learning rate $\eta(k) = 0.1$

Outline

① Linear models

Binary

Multiclass

Parameter estimation

② Perceptron

Introduction

Training

Limitations

Multiclass Perceptron

③ Multi-layer Perceptron

Introduction

Structure

Training

Training dynamics and regularization

Description

- Introduced by Rosenblatt in 1958

Description

- Introduced by Rosenblatt in 1958
- Practical linear **binary classifier**
 - Linear model is wrapped in a step function \Rightarrow activation

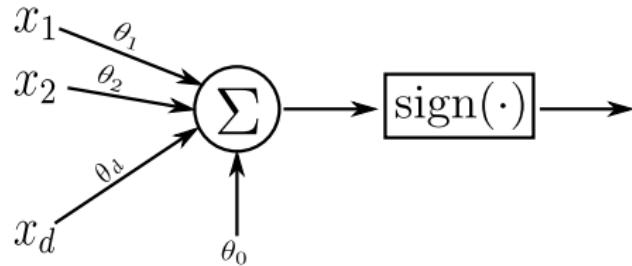
Description

- Introduced by Rosenblatt in 1958
- Practical linear **binary classifier**
 - Linear model is wrapped in a step function \Rightarrow activation
- Represents the **foundation of neural models**

Description

- Introduced by Rosenblatt in 1958
- Practical linear **binary classifier**
 - Linear model is wrapped in a step function ⇒ activation
- Represents the **foundation of neural models**

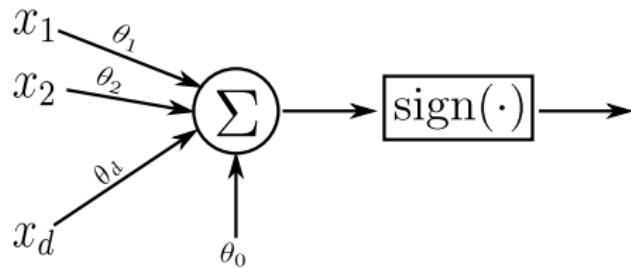
$$\hat{\omega} = \text{sign}(\theta^T \mathbf{x} + \theta_0)$$



Description

- Introduced by Rosenblatt in 1958
- Practical linear **binary classifier**
 - Linear model is wrapped in a step function \Rightarrow activation
- Represents the **foundation of neural models**

$$\hat{\omega} = \text{sign}(\theta^T \mathbf{x} + \theta_0) \quad \text{sign}(\cdot) = \begin{cases} +1 & \text{if } \theta^T \mathbf{x} + \theta_0 \geq 0 \\ -1 & \text{if } \theta^T \mathbf{x} + \theta_0 < 0 \end{cases}$$



Context

- Assume **binary scenario** $\mathcal{W} = \{\omega_1, \omega_2\}$

Context

- Assume **binary scenario** $\mathcal{W} = \{\omega_1, \omega_2\}$
- **Dataset** $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $\omega_i \in \mathcal{W}$ for $1 \leq i \leq |\mathcal{D}|$

Context

- Assume **binary scenario** $\mathcal{W} = \{\omega_1, \omega_2\}$
- **Dataset** $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $\omega_i \in \mathcal{W}$ for $1 \leq i \leq |\mathcal{D}|$
- Vector of **parameters** $\boldsymbol{\theta} = [\theta_0, \theta_1, \dots, \theta_d]$

Context

- Assume **binary scenario** $\mathcal{W} = \{\omega_1, \omega_2\}$
- **Dataset** $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $\omega_i \in \mathcal{W}$ for $1 \leq i \leq |\mathcal{D}|$
- Vector of **parameters** $\boldsymbol{\theta} = [\theta_0, \theta_1, \dots, \theta_d]$
- We may define vector $\mathbf{z} = [z_1, \dots, z_{|\mathcal{D}|}]$ such that:

$$z_i = \begin{cases} +1 & \text{if } \omega_i = \omega_1 \\ -1 & \text{if } \omega_i = \omega_2 \end{cases}$$

Context

- Assume **binary scenario** $\mathcal{W} = \{\omega_1, \omega_2\}$
- **Dataset** $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d$ and $\omega_i \in \mathcal{W}$ for $1 \leq i \leq |\mathcal{D}|$
- Vector of **parameters** $\boldsymbol{\theta} = [\theta_0, \theta_1, \dots, \theta_d]$
- We may define vector $\mathbf{z} = [z_1, \dots, z_{|\mathcal{D}|}]$ such that:

$$z_i = \begin{cases} +1 & \text{if } \omega_i = \omega_1 \\ -1 & \text{if } \omega_i = \omega_2 \end{cases}$$

- This way, **all samples** in \mathcal{D} will be **correctly classified** if and only if:

$$z_i \boldsymbol{\theta}^T \mathbf{x}_i > 0 \quad \forall i$$

The Perceptron Criterion

How can we estimate θ ?

The Perceptron Criterion

How can we estimate θ ? Define $J(\theta)$

The Perceptron Criterion

How can we estimate θ ? Define $J(\theta)$

- ✗ Number of misclassified samples \Rightarrow unsuitable for gradient search

The Perceptron Criterion

How can we estimate θ ? Define $J(\theta)$

- ✗ Number of misclassified samples \Rightarrow unsuitable for gradient search
- ✓ Sum of the distances to surface \mathcal{H} for misclassified samples:

$$J_p(\theta) = \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \theta^T \mathbf{x}_i)$$

The Perceptron Criterion

How can we estimate θ ? Define $J(\theta)$

- ✗ Number of misclassified samples \Rightarrow unsuitable for gradient search
- ✓ Sum of the distances to surface \mathcal{H} for misclassified samples:

$$J_p(\theta) = \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \theta^T \mathbf{x}_i)$$

→ The corresponding gradient is:

$$\nabla J_p(\theta) = \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \mathbf{x}_i)$$

The Perceptron Criterion

Batch Perceptron algorithm

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \cdot \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \mathbf{x}_i)$$

The Perceptron Criterion

Batch Perceptron algorithm

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \cdot \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \mathbf{x}_i)$$

- Instead of **batches**, correction may be done on a **singe-sample policy**:

The Perceptron Criterion

Batch Perceptron algorithm

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \cdot \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \mathbf{x}_i)$$

- Instead of **batches**, correction may be done on a **singe-sample policy**:

$$\rightarrow J_p(\theta, \mathbf{x}_i) = \begin{cases} -z_i \theta^T \mathbf{x}_i & \text{if } z_i \theta^T \mathbf{x}_i \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

The Perceptron Criterion

Batch Perceptron algorithm

$$\theta^{(k+1)} = \theta^{(k)} - \eta(k) \cdot \sum_{i: z_i \theta^T \mathbf{x}_i \leq 0} (-z_i \mathbf{x}_i)$$

- Instead of **batches**, correction may be done on a **singe-sample policy**:

$$\rightarrow J_p(\theta, \mathbf{x}_i) = \begin{cases} -z_i \theta^T \mathbf{x}_i & \text{if } z_i \theta^T \mathbf{x}_i \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

Single-sample Perceptron algorithm

$$\theta^{(k+1)} = \theta^{(k)} + \eta(k) z_i \mathbf{x}_i$$

The Perceptron Criterion

Approach	Update frequency	Pros	Cons
Single-sample	Every sample	Fast reaction to new data stochasticity	Noisy updates
Batch	Accumulation	Smoother Stable convergence	Slower

The Perceptron Criterion

The XOR problem

The *Exclusive OR* (XOR) operation \Rightarrow **binary classification** task:

- Feature space: $\mathbf{x} \in \{0, 1\}^2$
- Label space: $\mathcal{W} = \{0, 1\}$
- Function $f_{xor}(\mathbf{x}) = \begin{cases} \omega_1 & \text{if } x_1 = x_2 \\ \omega_2 & \text{if } x_1 \neq x_2 \end{cases}$

The XOR problem

- Showcases the main **limitation** of the **perceptron** mechanism
- **Unable** to address **non-linearly separable** labels
 - Need for **non-linearities** in the scheme
- Two related **mechanisms**:
 - Stacking perceptrons into **layers**
 - Using non-linear **activation functions**

Multiclass Perceptron

The Perceptron learner may address **multiclass scenarios** by:

Multiclass Perceptron

The Perceptron learner may address multiclass scenarios by:

1. **One-VS-Rest** scheme:

- Requires $|\mathcal{W}|$ Perceptrons
- Perceptron with the largest activation determines $\hat{\omega}$

Multiclass Perceptron

The Perceptron learner may address multiclass scenarios by:

1. **One-VS-Rest** scheme:

- Requires $|\mathcal{W}|$ Perceptrons
- Perceptron with the largest activation determines $\hat{\omega}$

2. **One-VS-One** scheme:

- Requires $|\mathcal{W}| \cdot (|\mathcal{W}| - 1)/2$ Perceptrons
- Class $\hat{\omega}$ is estimated by voting

Multiclass Perceptron

The Perceptron learner may address multiclass scenarios by:

1. One-VS-Rest scheme:

- Requires $|\mathcal{W}|$ Perceptrons
- Perceptron with the largest activation determines $\hat{\omega}$

2. One-VS-One scheme:

- Requires $|\mathcal{W}| \cdot (|\mathcal{W}| - 1)/2$ Perceptrons
- Class $\hat{\omega}$ is estimated by voting

3. Multiple weight vectors:

- Requires only one Perceptron
- Weight matrix with a weight vector per class: $\theta_1, \theta_2, \dots, \theta_{|\mathcal{W}|}$
- Only the weight vector associated to the class is updated while training
- Weight vector with the largest activation determines $\hat{\omega}$

Outline

① Linear models

- Binary
- Multiclass
- Parameter estimation

② Perceptron

- Introduction
- Training
- Limitations
- Multiclass Perceptron

③ Multi-layer Perceptron

- Introduction
- Structure
- Training
- Training dynamics and regularization

Introduction to MLP

- Natural evolution of the Perceptron learner

Introduction to MLP

- Natural **evolution** of the **Perceptron** learner
- Stack of **Perceptron** units \Rightarrow approximating **non-linear functions**

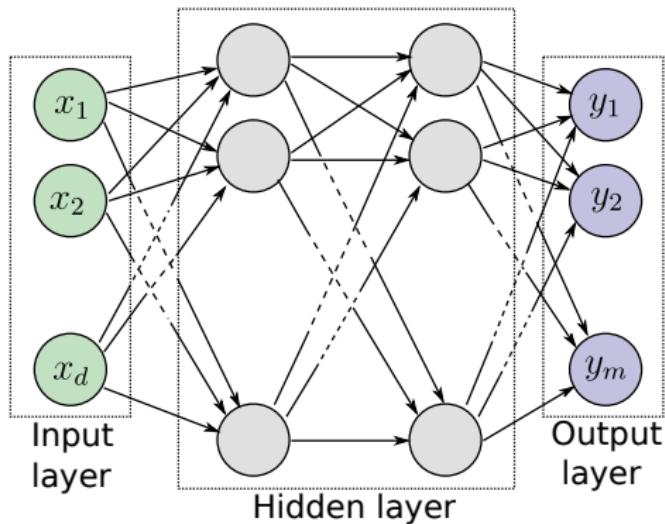
Introduction to MLP

- Natural **evolution** of the **Perceptron** learner
- **Stack of Perceptron** units \Rightarrow approximating **non-linear functions**
- Constitutes a particular case of **Feedforward Neural Networks**
 - \rightarrow Information flows in **one direction** \Rightarrow input to output

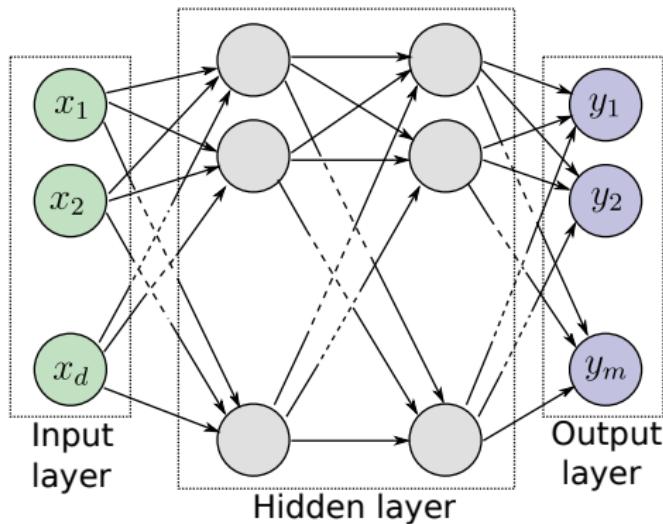
Introduction to MLP

- Natural **evolution** of the **Perceptron** learner
- Stack of **Perceptron** units \Rightarrow approximating **non-linear functions**
- Constitutes a particular case of **Feedforward Neural Networks**
 - \rightarrow Information flows in **one direction** \Rightarrow input to output
- **Universal Approximation Theorem:**
A feedforward neural network with at least one hidden layer, using a nonlinear activation, can approximate any continuous function to any desired degree of accuracy, given sufficient neurons in the hidden layer

Structure



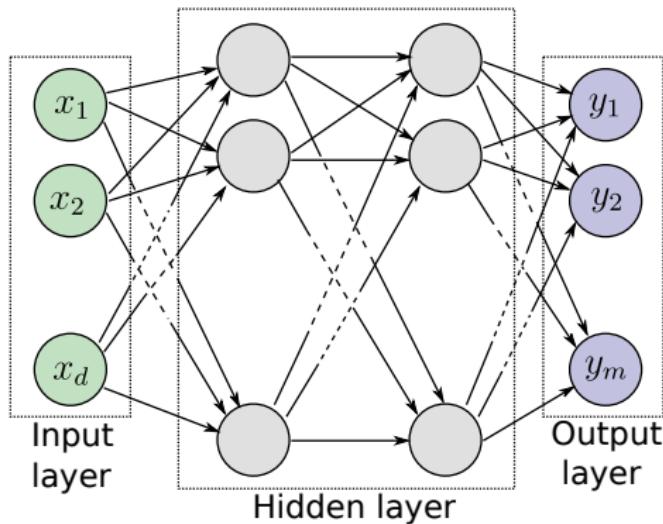
Structure



Input

- Representation space \mathbb{R}^d
⇒ d neurons

Structure



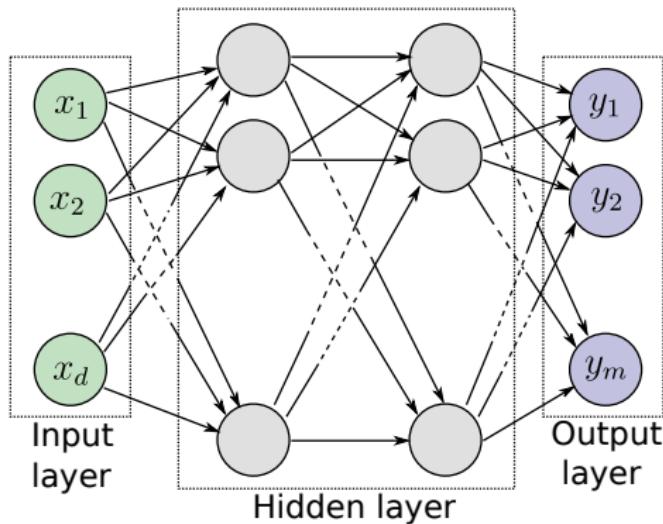
Input

- Representation space \mathbb{R}^d
 $\Rightarrow d$ neurons

Hidden

- Non-linear activations
- Fully connected
- Configuration depends on the complexity of the function

Structure



Input

- Representation space \mathbb{R}^d
 $\Rightarrow d$ neurons

Hidden

- Non-linear activations
- Fully connected
- Configuration depends on the complexity of the function

Output

- Configuration depends on the task

Input layer

- Entrance point to the model for datum $\mathbf{x} \in \mathbb{R}^d$
- As many neurons as the dimensionality of $\mathbf{x} \Rightarrow d$ neurons
- There may be additional neurons \Rightarrow additional biases

Hidden layer

- Sequence of (vertical) stacks of Perceptron units:
 - I stacks
 - J_i neurons at the i -th stack ($1 \leq i \leq I$)

Hidden layer

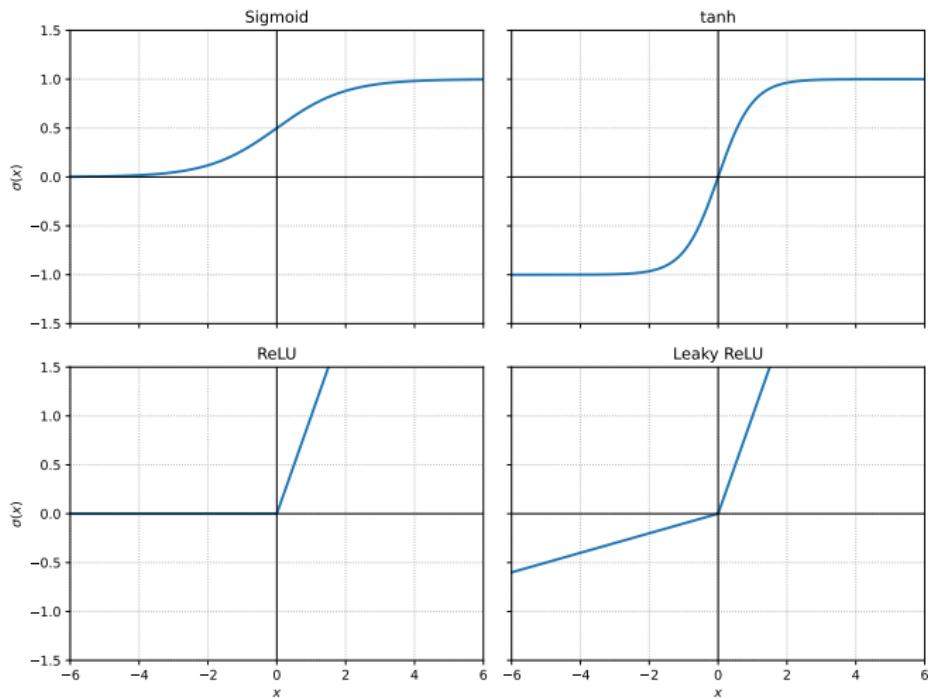
- Sequence of (vertical) stacks of Perceptron units:
 - I stacks
 - J_i neurons at the i -th stack ($1 \leq i \leq I$)
- Each stack may depict a different number of Perceptron units

Hidden layer

- Sequence of (vertical) stacks of Perceptron units:
 - I stacks
 - J_i neurons at the i -th stack ($1 \leq i \leq I$)
- Each stack may depict a different number of Perceptron units
- In general, non-linear activation functions:

$$y_{h_{i,j}} = \sigma \left(\mathbf{h}_{i,j}^T \cdot \mathbf{y}_{h_{i-1}} \right)$$

Activation functions



Activation functions

- Sigmoid:

$$\sigma(x) = \frac{e^x}{1 + e^x}$$

- Hyperbolic tangent (tanh):

$$\sigma(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

- Rectified Linear Unit (ReLU):

$$\sigma(x) = \max \{0, x\}$$

- Leaky Rectified Linear Unit (Leaky ReLU):

$$\sigma(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha \cdot x & \text{if } x < 0 \end{cases} \quad \text{with } \alpha \in [0.1, 0.3]$$

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$
- Regression tasks:
 - One **regressor per output** unit

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$
- **Regression** tasks:
 - One **regressor per output** unit
- **Classification** tasks ($|\mathcal{W}|$ classes):
 - One **unit per class** $\Rightarrow m = |\mathcal{C}|$

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$
- Regression tasks:
 - One **regressor per output** unit
- Classification tasks ($|\mathcal{W}|$ classes):
 - One **unit per class** $\Rightarrow m = |\mathcal{C}|$
 - **Highest score** is the **estimated class** $\hat{\omega} = \arg \max_{j \in \{1, \dots, m\}} y_j$

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$
- **Regression** tasks:
 - One **regressor per output** unit
- **Classification** tasks ($|\mathcal{W}|$ classes):
 - One **unit per class** $\Rightarrow m = |\mathcal{C}|$
 - **Highest score** is the **estimated class** $\hat{\omega} = \arg \max_{j \in \{1, \dots, m\}} y_j$
 - **Softmax activation as estimated probability**:

$$\hat{P}(\omega | \mathbf{x}) = \frac{e^{y_\omega}}{\sum_{j=1}^{|\mathcal{W}|} e^{y_j}} \in [0, 1]$$

Output layer

- Each output is a **real-valued** function: $y_j(\mathbf{x}; \boldsymbol{\theta}) \in \mathbb{R}$ with $i \leq j \leq m$

- **Regression** tasks:
 - One **regressor per output** unit

- **Classification** tasks ($|\mathcal{W}|$ classes):
 - One **unit per class** $\Rightarrow m = |\mathcal{C}|$
 - **Highest score** is the **estimated class** $\hat{\omega} = \arg \max_{j \in \{1, \dots, m\}} y_j$
 - **Softmax activation as estimated probability**:

$$\hat{P}(\omega | \mathbf{x}) = \frac{e^{y_\omega}}{\sum_{j=1}^{|\mathcal{W}|} e^{y_j}} \in [0, 1]$$

→ **Binary** case ($|\mathcal{C}| = 2$) may be modeled with a **single neuron**:

$$\hat{\omega} = \begin{cases} \omega_1 & \text{if } y \geq 0 \\ \omega_2 & \text{if } y < 0 \end{cases}$$

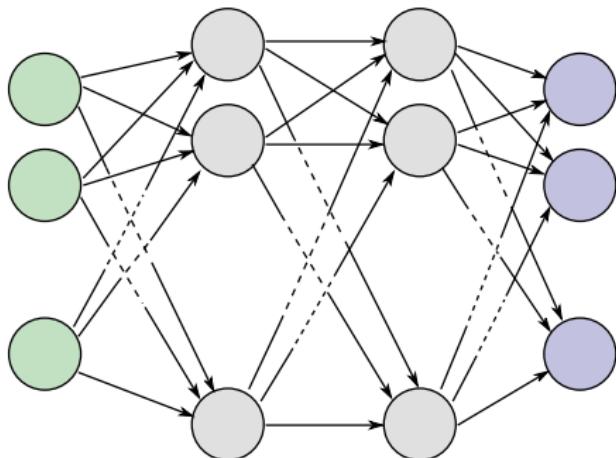
Contextualization

- Training is done following the *forward-backward* propagation

Contextualization

- Training is done following the *forward-backward* propagation

- Steps:

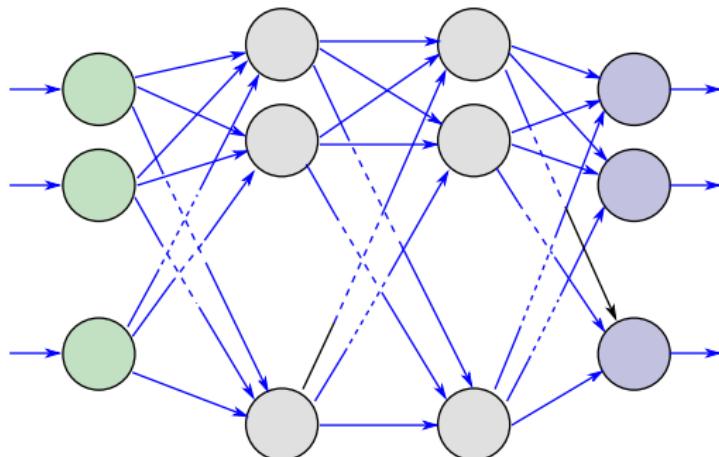


Contextualization

- Training is done following the *forward-backward* propagation

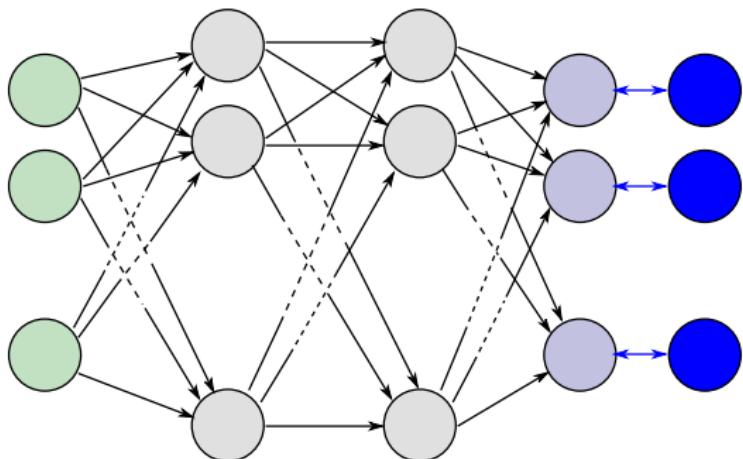
- Steps:

1. Forward pass



Contextualization

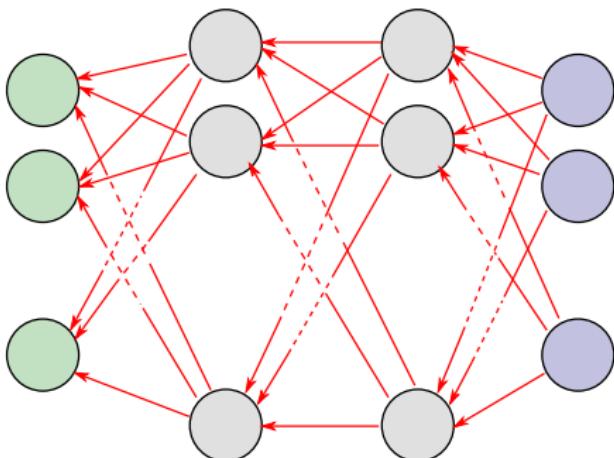
- Training is done following the *forward-backward* propagation
- Steps:
 1. Forward pass
 2. Loss computation



Contextualization

- Training is done following the *forward-backward* propagation

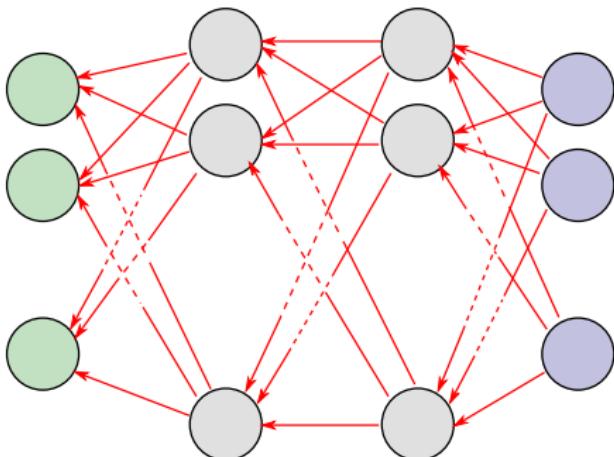
- Steps:
 1. Forward pass
 2. Loss computation
 3. Backward pass
 4. Weight update



Contextualization

- Training is done following the *forward-backward* propagation

- Steps:
 1. Forward pass
 2. Loss computation
 3. Backward pass
 4. Weight update

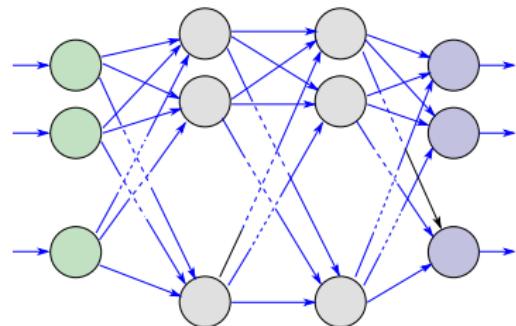


- The process is done until a convergence criterion is reached

Forward-backward training

1. Forward pass:

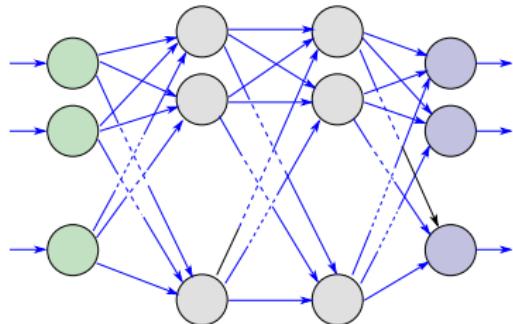
- Datum x goes from **input to output**
- Weights θ modify x across the network



Forward-backward training

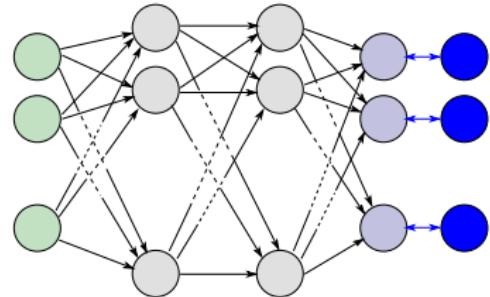
1. Forward pass:

- Datum x goes from **input to output**
- Weights θ modify x across the network



2. Loss computation:

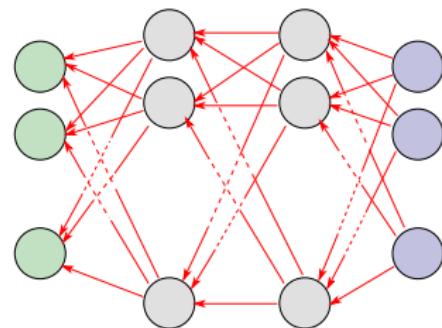
- Estimation \hat{y} is compared to reference y
- Depends on the task:
 - Regression: MAE, MSE
 - Classification: Categorical Cross Entropy



Forward-backward training

3. Backward pass (backpropagation) :

- Gradient of the loss function with respect to θ
- Chain rule for the derivatives of the (nested) equation
- Gradients are propagated from output to input



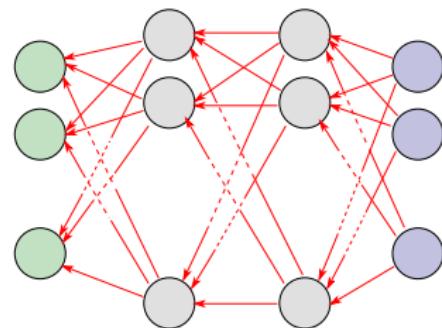
Forward-backward training

3. Backward pass (backpropagation) :

- Gradient of the loss function with respect to θ
- Chain rule for the derivatives of the (nested) equation
- Gradients are propagated from output to input

4. Weight update:

- Update parameters θ (weights and biases)
- Optimization methods:
 - Stochastic Gradient Descent
 - Momentum, RMSProp, Adam, ...



Classification scenarios

- Number of neurons must match labels $\Rightarrow m = |\mathcal{W}|$

Classification scenarios

- Number of **neurons** must match **labels** $\Rightarrow m = |\mathcal{W}|$
- Consider a case of $m = 3$:

Classification scenarios

- Number of **neurons** must match **labels** $\Rightarrow m = |\mathcal{W}|$
- Consider a case of $m = 3$:

Ideal case

Class A: [1, 0, 0]

Class B: [0, 1, 0]

Class C: [0, 0, 1]

Classification scenarios

- Number of **neurons** must match **labels** $\Rightarrow m = |\mathcal{W}|$
- Consider a case of $m = 3$:

Ideal case

Class A: [1, 0, 0]

Class B: [0, 1, 0]

Class C: [0, 0, 1]

Real case

Class A: [0.67, 0.10, 0.23]

Class B: [0.05, 0.80, 0.15]

Class C: [0.25, 0.25, 0.50]

Classification scenarios

- Number of **neurons** must match **labels** $\Rightarrow m = |\mathcal{W}|$
- Consider a case of $m = 3$:

Ideal case

Class A: [1, 0, 0]
Class B: [0, 1, 0]
Class C: [0, 0, 1]

Real case

Class A: [0.67, 0.10, 0.23]
Class B: [0.05, 0.80, 0.15]
Class C: [0.25, 0.25, 0.50]

- **Training** \Rightarrow reference data must be adequately \Rightarrow **one-hot encoding**
 \rightarrow **Mapping** function: $[\omega_1, \dots, \omega_{|\mathcal{W}|}] \rightarrow [0, 1]^{|\mathcal{W}|}$

Classification scenarios

- Number of **neurons** must match **labels** $\Rightarrow m = |\mathcal{W}|$
- Consider a case of $m = 3$:

Ideal case

Class A: [1, 0, 0]
 Class B: [0, 1, 0]
 Class C: [0, 0, 1]

Real case

Class A: [0.67, 0.10, 0.23]
 Class B: [0.05, 0.80, 0.15]
 Class C: [0.25, 0.25, 0.50]

- **Training** \Rightarrow reference data must be adequately \Rightarrow **one-hot encoding**
 \rightarrow **Mapping** function: $[\omega_1, \dots, \omega_{|\mathcal{W}|}] \rightarrow [0, 1]^{|\mathcal{W}|}$
- **Categorical Cross Entropy** \Rightarrow Typical **loss** function for classification

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = - \sum_{j=1}^{|\mathcal{W}|} y_j \cdot \log(\hat{y}_j)$$

Classification scheme

Consider the following [example](#):

Classification scheme

Consider the following [example](#):

- Feature space \mathbb{R}^d with $d = 2$ dimensions

Classification scheme

Consider the following example:

- Feature space \mathbb{R}^d with $d = 2$ dimensions
- Three possible categories: $\mathcal{W} = \{\omega_1, \omega_2, \omega_3\}$

Classification scheme

Consider the following example:

- Feature space \mathbb{R}^d with $d = 2$ dimensions
- Three possible categories: $\mathcal{W} = \{\omega_1, \omega_2, \omega_3\}$
- Test element $\mathbf{x} = [0.4, -2.2]$ with $\omega = \omega_2$

Classification scheme

Consider the following example:

- Feature space \mathbb{R}^d with $d = 2$ dimensions
- Three possible categories: $\mathcal{W} = \{\omega_1, \omega_2, \omega_3\}$
- Test element $\mathbf{x} = [0.4, -2.2]$ with $\omega = \omega_2 \Rightarrow \mathbf{y} = [0, 1, 0]$

Classification scheme

Consider the following example:

- Feature space \mathbb{R}^d with $d = 2$ dimensions
- Three possible categories: $\mathcal{W} = \{\omega_1, \omega_2, \omega_3\}$
- Test element $\mathbf{x} = [0.4, -2.2]$ with $\omega = \omega_2 \Rightarrow \mathbf{y} = [0, 1, 0]$
- Predicted vector $\hat{\mathbf{y}} = [0.1, 0.7, 0.2]$

Classification scheme

Consider the following example:

- Feature space \mathbb{R}^d with $d = 2$ dimensions
- Three possible categories: $\mathcal{W} = \{\omega_1, \omega_2, \omega_3\}$
- Test element $\mathbf{x} = [0.4, -2.2]$ with $\omega = \omega_2 \Rightarrow \mathbf{y} = [0, 1, 0]$
- Predicted vector $\hat{\mathbf{y}} = [0.1, 0.7, 0.2]$

The associated loss is:

$$\begin{aligned}\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) &= - \sum_{j=1}^3 y_j \cdot \log(\hat{y}_j) \\ &= -[0 \cdot \log(0.1) + 1 \cdot \log(0.7) + 0 \cdot \log(0.2)] = 0.357\end{aligned}$$

Practical considerations

MLPs (and other Neural models) depict remarkable limitations:

Practical considerations

MLPs (and other Neural models) depict remarkable limitations:

1. **Overfitting:** *Memorizing* instead of *learning*

Practical considerations

MLPs (and other Neural models) depict remarkable limitations:

1. **Overfitting:** *Memorizing* instead of *learning*
2. **Vanishing/exploding gradients:** Gradients become negligible/excessively large producing stagnation/oscillations

Practical considerations

To prevent overfitting \Rightarrow regularization strategies:

Practical considerations

To prevent **overfitting** \Rightarrow **regularization** strategies:

- **L1/L2 regularization:** Adding additional penalties during training

Practical considerations

To prevent **overfitting** \Rightarrow **regularization** strategies:

- **L1/L2 regularization:** Adding additional penalties during training
- **Dropout:** Randomly disconnecting neurons during training

Practical considerations

To prevent **overfitting** \Rightarrow **regularization** strategies:

- **L1/L2 regularization**: Adding additional penalties during training
- **Dropout**: Randomly disconnecting neurons during training
- **Early stopping**: Stop training as validation loss increases

Practical considerations

To prevent **overfitting** \Rightarrow **regularization** strategies:

- **L1/L2 regularization**: Adding additional penalties during training
- **Dropout**: Randomly disconnecting neurons during training
- **Early stopping**: Stop training as validation loss increases
- **Data augmentation**: Apply transformations on (train) data to artificially expand the size of the set

Practical considerations

To prevent vanishing/exploding gradients:

Practical considerations

To prevent vanishing/exploding gradients:

- **Non-linear activations:** (Leaky) ReLU palliate vanishing issue

Practical considerations

To prevent vanishing/exploding gradients:

- **Non-linear activations:** (Leaky) ReLU palliate vanishing issue
- **Initialization strategies:** Weight initialization policies

Practical considerations

To prevent **vanishing/exploding** gradients:

- **Non-linear activations:** (Leaky) ReLU palliate vanishing issue
- **Initialization strategies:** Weight initialization policies
- **Batch/Layer normalization:** Maintaining activations in stable ranges

Practical considerations

To prevent **vanishing/exploding** gradients:

- **Non-linear activations:** (Leaky) ReLU palliate vanishing issue
- **Initialization strategies:** Weight initialization policies
- **Batch/Layer normalization:** Maintaining activations in stable ranges
- **Gradient clipping:** Limiting gradient value above threshold

T5: Linear methods and Perceptron

Fundamentos del Aprendizaje Automático

Curso 2025/2026

T6: Unsupervised learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Introduction

② Clustering

Definition

Taxonomy

The k -means clustering method

Cluster determination techniques

③ Dimensionality reduction

Definition

Statistical approaches

Neural approaches

④ Other tasks

Outline

① Introduction

② Clustering

Definition

Taxonomy

The k -means clustering method

Cluster determination techniques

③ Dimensionality reduction

Definition

Statistical approaches

Neural approaches

④ Other tasks

Problem statement

- **Supervised learning:** target to *guide* the *learning* process $\Rightarrow P(\omega|x)$

Problem statement

- **Supervised learning:** target to *guide* the *learning* process $\Rightarrow P(\omega|\mathbf{x})$
 - Statistical distributions $P(\mathbf{x}|\omega)$, $P(\omega)$
 - Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$

Problem statement

- **Supervised learning:** target to *guide* the learning process $\Rightarrow P(\omega|\mathbf{x})$
 - Statistical distributions $P(\mathbf{x}|\omega)$, $P(\omega)$
 - Data assortment $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$
- **Unsupervised learning:** No specific target is provided
 - Data assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$
 - Related to Exploratory Data Analysis

Main tasks

1. Clustering:

- Create **groups** (**clusters**) of data with **similar characteristics**
- Identifying/characterizing **patterns** in the data

Main tasks

1. Clustering:

- Create **groups** (**clusters**) of data with **similar characteristics**
- Identifying/characterizing **patterns** in the data

2. Dimensionality reduction:

- Derive **compact representations** of the initial feature space
- Reduce **complexity** of the task
- **Visualization** purposes

Main tasks

1. Clustering:

- Create **groups** (**clusters**) of data with **similar characteristics**
- Identifying/characterizing **patterns** in the data

2. Dimensionality reduction:

- Derive **compact representations** of the initial feature space
- Reduce **complexity** of the task
- **Visualization** purposes

3. Outlier/anomaly detection:

- Identify elements that **differ** for the rest in the assortment
- Remove **noise** in assortments
- Track **unexpected behaviours**

Outline

① Introduction

② Clustering

Definition

Taxonomy

The k -means clustering method

Cluster determination techniques

③ Dimensionality reduction

Definition

Statistical approaches

Neural approaches

④ Other tasks

Definition

- Divide assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ into $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_{|\mathcal{C}|}\}$ clusters

Definition

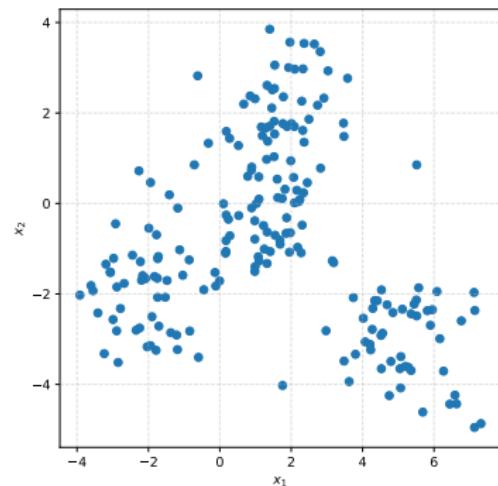
- Divide assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ into $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_{|\mathcal{C}|}\}$ clusters
 $\Rightarrow \bigcup_{j=1}^{|\mathcal{C}|} \mathcal{C}_j = \mathcal{D}$

Definition

- Divide assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ into $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_{|\mathcal{C}|}\}$ clusters
 $\Rightarrow \bigcup_{j=1}^{|\mathcal{C}|} \mathcal{C}_j = \mathcal{D}$
- Division policy depends on ground principle \Rightarrow Definition of *cluster*

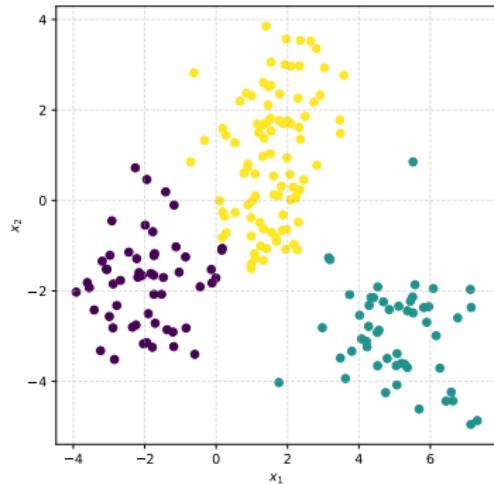
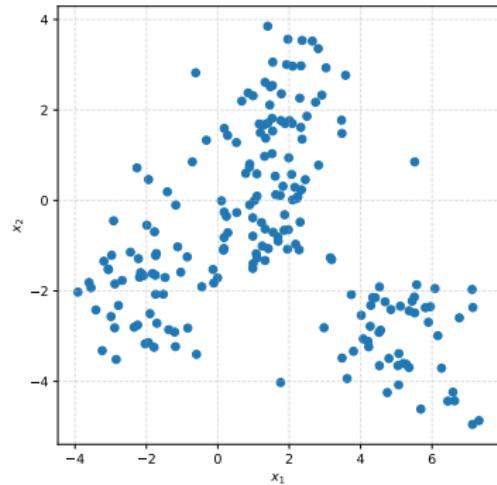
Definition

- Divide assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ into $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_{|\mathcal{C}|}\}$ clusters
 $\Rightarrow \bigcup_{j=1}^{|\mathcal{C}|} \mathcal{C}_j = \mathcal{D}$
- Division policy depends on ground principle \Rightarrow Definition of cluster



Definition

- Divide assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ into $\mathcal{C} = \{\mathcal{C}_1, \dots, \mathcal{C}_{|\mathcal{C}|}\}$ clusters
 $\Rightarrow \bigcup_{j=1}^{|\mathcal{C}|} \mathcal{C}_j = \mathcal{D}$
- Division **policy** depends on ground principle \Rightarrow Definition of *cluster*



Example

Taxonomy

Taxonomy

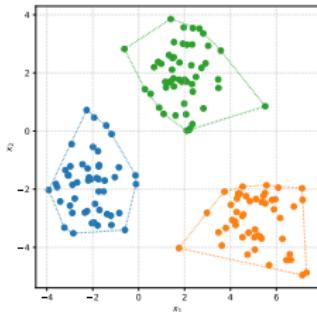
1. Partitional clustering
2. Hierarchical clustering

Taxonomy

1. Partitional clustering

- Data is organized in a plain structure without hierarchy
- Generates \mathcal{C} to recover the natural groupings inherent in \mathcal{D}

2. Hierarchical clustering



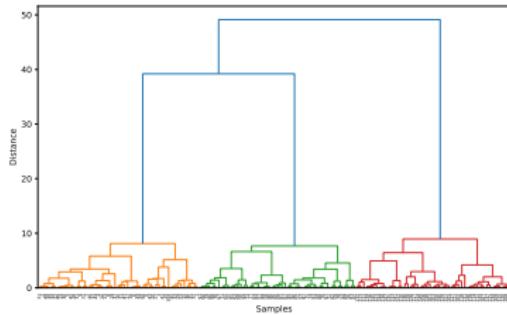
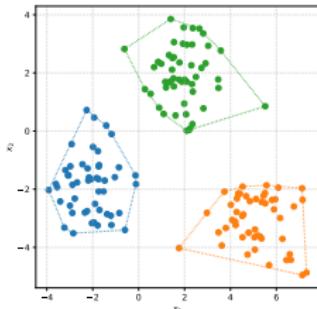
Taxonomy

1. Partitional clustering

- Data is organized in a plain structure without hierarchy
- Generates \mathcal{C} to recover the natural groupings inherent in \mathcal{D}

2. Hierarchical clustering

- \mathcal{D} is partitioned into levels in a hierarchical format \Rightarrow Dendrogram
 - Tree-like diagram that shows how items are grouped
- Two main approaches:
 - Top-down or divisive
 - Bottom-up or agglomerative



Partitional clustering

1.1 Hard clustering

1.2 Mixture resolving

1.3 Fuzzy clustering

Partitional clustering

1.1 Hard clustering

- Disjoint clusters: $\cap_{j=1}^{|C|} C_j = \emptyset$
- Common approaches: Graph-theoretic, density-based, centroid-based

1.2 Mixture resolving

1.3 Fuzzy clustering

Partitional clustering

1.1 Hard clustering

- Disjoint clusters: $\cap_{j=1}^{|C|} C_j = \emptyset$
- Common approaches: Graph-theoretic, density-based, centroid-based

1.2 Mixture resolving

- Assumes points come from mixtures of statistical distributions
- Goal is to estimate the parameters
- Example: Gaussian Mixture Model

1.3 Fuzzy clustering

Partitional clustering

1.1 Hard clustering

- Disjoint clusters: $\cap_{j=1}^{|C|} C_j = \emptyset$
- Common approaches: Graph-theoretic, density-based, centroid-based

1.2 Mixture resolving

- Assumes points come from mixtures of statistical distributions
- Goal is to estimate the parameters
- Example: Gaussian Mixture Model

1.3 Fuzzy clustering

- Clusters are defined as fuzzy sets
- Elements may belong to more than one cluster: $\cap_{j=1}^{|C|} C_j \neq \emptyset$

Definition

- Centroid/distance-based method \Rightarrow **Partitional** and **hard** clustering

Definition

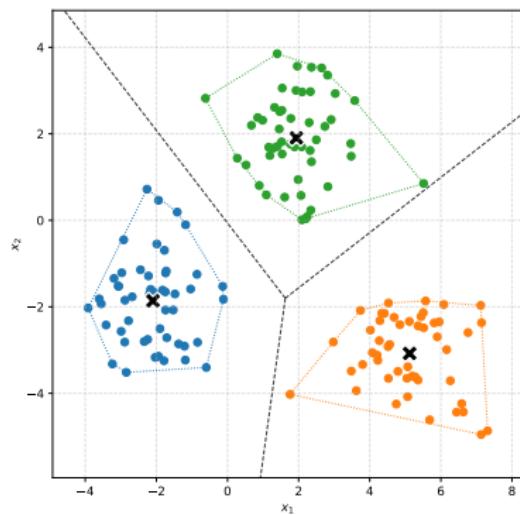
- Centroid/distance-based method \Rightarrow **Partitional** and **hard** clustering
 - \rightarrow Somehow **k -Nearest Neighbor** for unsupervised learning

Definition

- Centroid/distance-based method \Rightarrow **Partitional** and **hard** clustering
 - \rightarrow Somehow **k -Nearest Neighbor** for unsupervised learning
- Cluster: subset of elements whose distance to its centroid is minimal

Definition

- Centroid/distance-based method \Rightarrow **Partitional** and **hard** clustering
 \rightarrow Somehow **k -Nearest Neighbor** for unsupervised learning
- Cluster: subset of elements whose distance to its centroid is minimal



Formulation

We assume the following conditions:

Formulation

We assume the following conditions:

- Data assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$

Formulation

We assume the following conditions:

- Data **assortment** $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- Distance **metric** $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_0^+$

Formulation

We assume the following conditions:

- Data **assortment** $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- Distance **metric** $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_0^+$
- Total amount of $|\mathcal{C}|$ **clusters** $\Rightarrow \mathcal{C} = \{\mathcal{C}_j\}_{j=1}^{|\mathcal{C}|}$

Formulation

We assume the following conditions:

- Data **assortment** $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- Distance **metric** $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_0^+$
- Total amount of $|\mathcal{C}|$ **clusters** $\Rightarrow \mathcal{C} = \{\mathcal{C}_j\}_{j=1}^{|\mathcal{C}|}$
 - Collection of **centroids** $\{\boldsymbol{\mu}_j\}_{j=1}^{|\mathcal{C}|}$ where $\boldsymbol{\mu}_j \in \mathbb{R}^d$

Formulation

We assume the following conditions:

- Data **assortment** $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- Distance **metric** $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_0^+$
- Total amount of $|\mathcal{C}|$ **clusters** $\Rightarrow \mathcal{C} = \{\mathcal{C}_j\}_{j=1}^{|\mathcal{C}|}$
 - Collection of **centroids** $\{\boldsymbol{\mu}_j\}_{j=1}^{|\mathcal{C}|}$ where $\boldsymbol{\mu}_j \in \mathbb{R}^d$
 - Centroid: typically, **mean** of its elements $\Rightarrow \boldsymbol{\mu}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_j} \mathbf{x}$

Formulation

We assume the following conditions:

- Data **assortment** $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ where $\mathbf{x}_i \in \mathbb{R}^d$
- Distance **metric** $d : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}_0^+$
- Total amount of $|\mathcal{C}|$ **clusters** $\Rightarrow \mathcal{C} = \{\mathcal{C}_j\}_{j=1}^{|\mathcal{C}|}$
 - Collection of **centroids** $\{\boldsymbol{\mu}_j\}_{j=1}^{|\mathcal{C}|}$ where $\boldsymbol{\mu}_j \in \mathbb{R}^d$
 - Centroid: typically, **mean** of its elements $\Rightarrow \boldsymbol{\mu}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_j} \mathbf{x}$

Elements from \mathcal{D} associated to a given cluster \mathcal{C}_j :

$$\mathcal{C}_j = \left\{ \mathbf{x}_i \in \mathcal{D} : \arg \min_{1 \leq m \leq |\mathcal{C}|} d(\mathbf{x}_i, \boldsymbol{\mu}_m) \right\}$$

Formulation

Premise: Given a number of clusters $|\mathcal{C}|$ and a distance $d(\cdot, \cdot)$, obtain the data agrupation that minimizes the accumulated distance to the centroids

Formulation

Premise: Given a number of clusters $|\mathcal{C}|$ and a distance $d(\cdot, \cdot)$, obtain the data agrupation that minimizes the accumulated distance to the centroids

$$\arg \min_{\mathcal{C}} \underbrace{\sum_{j=1}^{|\mathcal{C}|} \sum_{x \in \mathcal{C}_j} d(x, \mu_j)}_{\text{Within-Cluster Sum of Squares}}$$

Formulation

Premise: Given a number of clusters $|\mathcal{C}|$ and a distance $d(\cdot, \cdot)$, obtain the data agrupation that minimizes the accumulated distance to the centroids

$$\arg \min_{\mathcal{C}} \underbrace{\sum_{j=1}^{|\mathcal{C}|} \sum_{x \in \mathcal{C}_j} d(x, \mu_j)}_{\text{Within-Cluster Sum of Squares}}$$

The exact solution to this equation is NP-hard

- The iterative Lloyd's algorithm provides a suboptimal solution

Lloyd's algorithm

Iterative algorithm that alternates between two steps:

Lloyd's algorithm

Iterative algorithm that alternates between two steps:

1. **Assignment** step:

2. **Update** step:

Lloyd's algorithm

Iterative algorithm that alternates between two steps:

1. Assignment step:

- Assign each datum $\mathbf{x}_i \in \mathcal{D}$ to the corresponding cluster $\mathcal{C}_j \in \mathcal{C}$:

$$\mathcal{C}_j = \left\{ \mathbf{x}_i \in \mathcal{D} : \arg \min_{1 \leq m \leq |\mathcal{C}|} d(\mathbf{x}_i, \boldsymbol{\mu}_m) \right\}$$

2. Update step:

Lloyd's algorithm

Iterative algorithm that alternates between two steps:

1. Assignment step:

- Assign each datum $\mathbf{x}_i \in \mathcal{D}$ to the corresponding cluster $\mathcal{C}_j \in \mathcal{C}$:

$$\mathcal{C}_j = \left\{ \mathbf{x}_i \in \mathcal{D} : \arg \min_{1 \leq m \leq |\mathcal{C}|} d(\mathbf{x}_i, \boldsymbol{\mu}_m) \right\}$$

2. Update step:

- Compute the centroid values $\boldsymbol{\mu}_j$ with $1 \leq j \leq |\mathcal{C}|$:

$$\boldsymbol{\mu}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_j} \mathbf{x}$$

Lloyd's algorithm

Iterative algorithm that alternates between two steps:

1. Assignment step:

- Assign each datum $\mathbf{x}_i \in \mathcal{D}$ to the corresponding cluster $\mathcal{C}_j \in \mathcal{C}$:

$$\mathcal{C}_j = \left\{ \mathbf{x}_i \in \mathcal{D} : \arg \min_{1 \leq m \leq |\mathcal{C}|} d(\mathbf{x}_i, \boldsymbol{\mu}_m) \right\}$$

2. Update step:

- Compute the centroid values $\boldsymbol{\mu}_j$ with $1 \leq j \leq |\mathcal{C}|$:

$$\boldsymbol{\mu}_j = \frac{1}{|\mathcal{C}_j|} \sum_{\mathbf{x} \in \mathcal{C}_j} \mathbf{x}$$

Process seeks to minimize *Within-Cluster Sum of Squares*:

- No changes in the cluster assignment between iterations

Lloyd's algorithm

Initialization

Initialization plays a key role in the convergence:

Initialization

Initialization plays a key role in the convergence:

- *Classic k-means*: selects $|\mathcal{C}|$ random points from \mathcal{D} as centroids

Initialization

Initialization plays a key role in the convergence:

- Classic k -means: selects $|\mathcal{C}|$ random points from \mathcal{D} as centroids
- k -means++: avoids issues related to random initialization

Initialization

Initialization plays a key role in the convergence:

- Classic k -means: selects $|\mathcal{C}|$ random points from \mathcal{D} as centroids
- k -means++: avoids issues related to random initialization

k -means++ initialization method

Input: Number of clusters $|\mathcal{C}|$

Result: Set of centroids $\{\mu_j\}_{j=1}^{|\mathcal{C}|}$

$K = 1, \mu_1 \in_R \mathcal{D};$

while $K < |\mathcal{C}|$ **do**

$$\mathcal{P} = \left\{ \min_{k=1}^K \|\mathbf{x}_i - \mu_k\|^2 \right\} \forall \mathbf{x}_i \in \mathcal{D};$$

$$\mu_{K+1} = \mathcal{D}_{\text{Categorical}}(\mathcal{P} / \sum \mathcal{P});$$

$$K = K + 1;$$

end

Example of k -means++

Additional remarks

- ✓ May address both **statistical** and **structural** representations
 - Only requires a **distance** metric

Additional remarks

- ✓ May address both **statistical** and **structural** representations
 - Only requires a **distance** metric

- ✗ Mean operator is **sensitive** to **outliers**

Additional remarks

- ✓ May address both **statistical** and **structural** representations
 - Only requires a **distance** metric
- ✗ Mean operator is **sensitive** to **outliers**
- ✓ Alternative approach: **k -Medoids**
 - **In-set mean** \Rightarrow Closest element in cluster \mathcal{C}_j to mean μ_j ($1 \leq j \leq |\mathcal{C}|$)
$$\mu_j^s = \arg \min_{\mathbf{x} \in \mathcal{C}_j} d(\mathbf{x}, \mu_j)$$

Contextualization

Key question ⇒ how many clusters do we have in our data?

Contextualization

Key question ⇒ how many clusters do we have in our data?

✗ It is not possible to know the *true number* of groups (**unsupervised**)

Contextualization

Key question ⇒ how many clusters do we have in our data?

- ✗ It is not possible to know the *true number* of groups (**unsupervised**)
- ✓ There exist strategies to estimate this parameter

Contextualization

Key question ⇒ how many clusters do we have in our data?

- ✗ It is not possible to know the *true number* of groups (**unsupervised**)
- ✓ There exist strategies to estimate this parameter

Two representative strategies to perform this estimation:

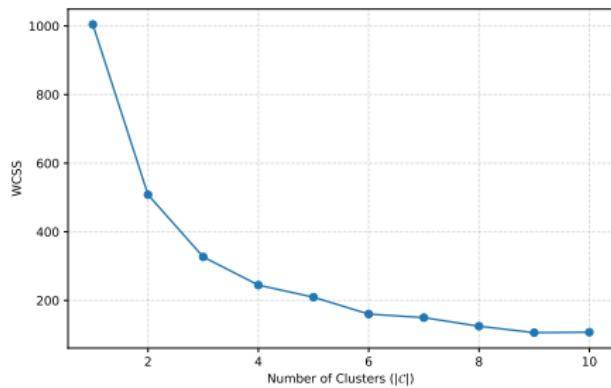
1. Elbow method
2. Silhouette method

Elbow method

- In general, higher $|\mathcal{C}|$ values imply lower WCSS scores

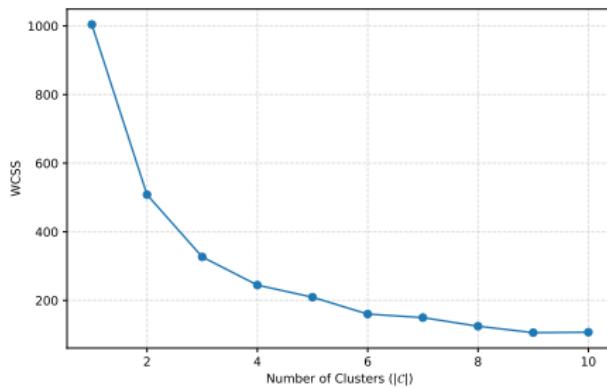
Elbow method

- In general, higher $|\mathcal{C}|$ values imply lower WCSS scores
 - Graphically, this relation resembles a negative exponential curve



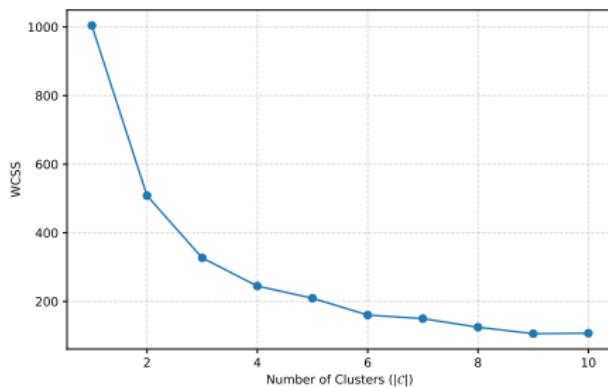
Elbow method

- In general, higher $|C|$ values imply lower WCSS scores
 - Graphically, this relation resembles a negative exponential curve
- After a certain point: more clusters \Leftrightarrow small WCSS improvements
 - Turning point ⇒ Elbow of the curve



Elbow method

- In general, higher $|C|$ values imply lower WCSS scores
 - Graphically, this relation resembles a negative exponential curve
- After a certain point: more clusters \Leftrightarrow small WCSS improvements
 - Turning point ⇒ Elbow of the curve
- The elbow is assumed to be the optimal number of clusters $|C|$



Elbow method

Silhouette method

Measures **how well** each **sample x_i** fits into its assigned cluster \mathcal{C}_j compared to other clusters

Silhouette method

Measures **how well** each **sample x_i** fits into its assigned cluster \mathcal{C}_j compared to other clusters

Considers **two aspects** for assessing the ***quality*** of the **clustering**:

1. **Compactness**: how **tight** a cluster is
2. **Separation**: how **far apart** clusters are

Silhouette method

Consider a point $\mathbf{x}_m \in \mathcal{D}$ currently assigned to cluster \mathcal{C}_j :

Silhouette method

Consider a point $\mathbf{x}_m \in \mathcal{D}$ currently assigned to cluster \mathcal{C}_j :

- Cohesion indicator $a(\mathbf{x}_m)$:

$$a(\mathbf{x}_m) = \frac{1}{|\mathcal{C}_j| - 1} \sum_{\mathbf{x}_n \in \mathcal{C}_j, \mathbf{x}_m \neq \mathbf{x}_n} d(\mathbf{x}_m, \mathbf{x}_n)$$

Silhouette method

Consider a point $\mathbf{x}_m \in \mathcal{D}$ currently assigned to cluster \mathcal{C}_j :

- Cohesion indicator $a(\mathbf{x}_m)$:

$$a(\mathbf{x}_m) = \frac{1}{|\mathcal{C}_j| - 1} \sum_{\mathbf{x}_n \in \mathcal{C}_j, \mathbf{x}_m \neq \mathbf{x}_n} d(\mathbf{x}_m, \mathbf{x}_n)$$

- Separation indicator $b(\mathbf{x}_m)$:

$$b(\mathbf{x}_m) = \min_{c \in \{1, \dots, |\mathcal{C}|, c \neq j\}} \frac{1}{|\mathcal{C}_c|} \sum_{\mathbf{x}_n \in \mathcal{C}_c} d(\mathbf{x}_m, \mathbf{x}_n)$$

Silhouette method

Consider a point $\mathbf{x}_m \in \mathcal{D}$ currently assigned to cluster \mathcal{C}_j :

- Cohesion indicator $a(\mathbf{x}_m)$:

$$a(\mathbf{x}_m) = \frac{1}{|\mathcal{C}_j| - 1} \sum_{\mathbf{x}_n \in \mathcal{C}_j, \mathbf{x}_m \neq \mathbf{x}_n} d(\mathbf{x}_m, \mathbf{x}_n)$$

- Separation indicator $b(\mathbf{x}_m)$:

$$b(\mathbf{x}_m) = \min_{c \in \{1, \dots, |\mathcal{C}|, c \neq j\}} \frac{1}{|\mathcal{C}_c|} \sum_{\mathbf{x}_n \in \mathcal{C}_c} d(\mathbf{x}_m, \mathbf{x}_n)$$

- Silhouette score $s(\mathbf{x}_m)$:

$$s(\mathbf{x}_m) = \frac{b(\mathbf{x}_m) - a(\mathbf{x}_m)}{\max \{a(\mathbf{x}_m), b(\mathbf{x}_m)\}} \in [-1, +1]$$

Silhouette method

The Silhouette indicator exhibits **three situations**:

$$s(\mathbf{x}_m) \begin{cases} > 0 & \text{element adequately clustered} \\ = 0 & \text{element on decision boundary} \\ < 0 & \text{element probably in incorrect cluster} \end{cases}$$

Silhouette method

The Silhouette indicator exhibits **three situations**:

$$s(\mathbf{x}_m) \begin{cases} > 0 & \text{element adequately clustered} \\ = 0 & \text{element on decision boundary} \\ < 0 & \text{element probably in incorrect cluster} \end{cases}$$

Since the indicator is individual, a **global Silhouette** may be derived as:

$$\hat{s} = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_m \in \mathcal{D}} s(\mathbf{x}_m)$$

Silhouette method

The Silhouette indicator exhibits **three situations**:

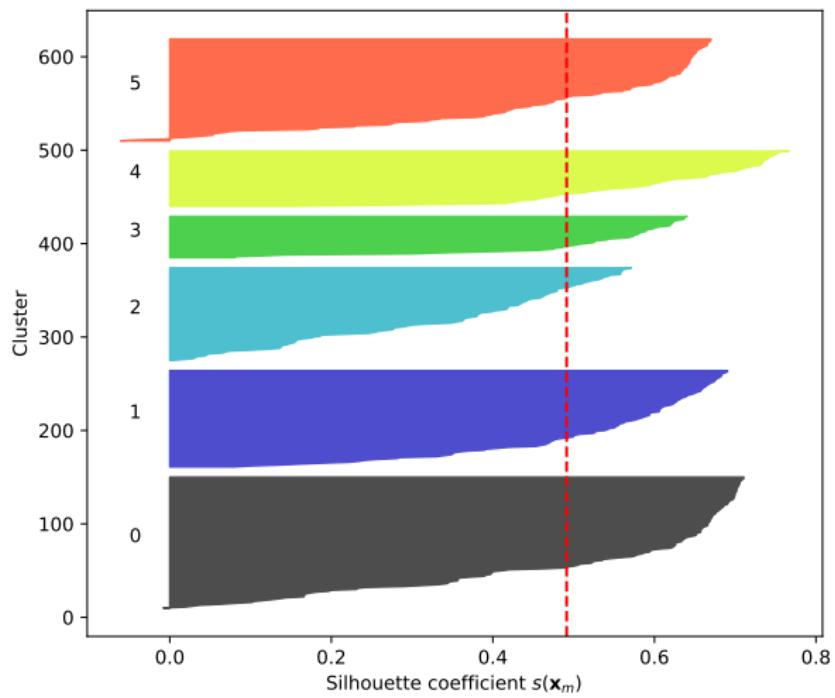
$$s(\mathbf{x}_m) \begin{cases} > 0 & \text{element adequately clustered} \\ = 0 & \text{element on decision boundary} \\ < 0 & \text{element probably in incorrect cluster} \end{cases}$$

Since the indicator is individual, a **global Silhouette** may be derived as:

$$\hat{s} = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_m \in \mathcal{D}} s(\mathbf{x}_m)$$

→ Can be used as **alternative metric** for the **Elbow method**

Silhouette method



Outline

① Introduction

② Clustering

Definition

Taxonomy

The k -means clustering method

Cluster determination techniques

③ Dimensionality reduction

Definition

Statistical approaches

Neural approaches

④ Other tasks

Definition

Curse of dimensionality: Large feature dimensionalities $\mathbb{R}^d \not\Rightarrow$ beneficial

Definition

Curse of dimensionality: Large feature dimensionalities $\mathbb{R}^d \nLeftrightarrow$ beneficial

- Data sparsity
- Correlations in the feature space

Definition

Curse of dimensionality: Large feature dimensionalities $\mathbb{R}^d \nLeftrightarrow$ beneficial

- Data sparsity
- Correlations in the feature space

Dimensionality reduction: Decrease the number d of features

Definition

Curse of dimensionality: Large feature dimensionalities $\mathbb{R}^d \nLeftrightarrow$ beneficial

- Data sparsity
- Correlations in the feature space

Dimensionality reduction: Decrease the number d of features

- $\mathcal{R} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'} \text{ with } d' \leq d$
- Does not reduce the number of elements in the \mathcal{D}
- **Unsupervised** strategies

Definition

Curse of dimensionality: Large feature dimensionalities $\mathbb{R}^d \nLeftrightarrow$ beneficial

- Data sparsity
- Correlations in the feature space

Dimensionality reduction: Decrease the number d of features

- $\mathcal{R} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$ with $d' \leq d$
- Does not reduce the number of elements in the \mathcal{D}
- **Unsupervised** strategies

Common paradigms:

- **Statistical** frameworks: PCA, t-SNE
- **Neural-based** approaches: Autoencoder

Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:

Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:

→ Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$

Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:

→ Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$

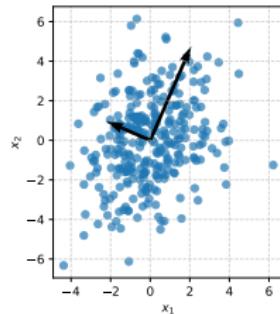
$$\Rightarrow \mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V} \text{ with } \mathcal{V} \in \mathbb{R}^{d \times d'}$$

Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$
 - ⇒ $\mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V}$ with $\mathcal{V} \in \mathbb{R}^{d \times d'}$
 - If $d' < d \Rightarrow$ Dimensionality reduction

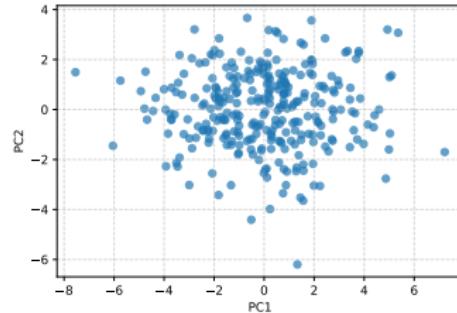
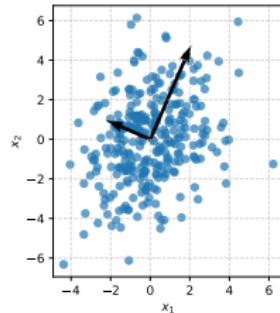
Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$
 - ⇒ $\mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V}$ with $\mathcal{V} \in \mathbb{R}^{d \times d'}$
 - If $d' < d \Rightarrow$ Dimensionality reduction
- Goal: Find *directions* that capture the most variance in the data



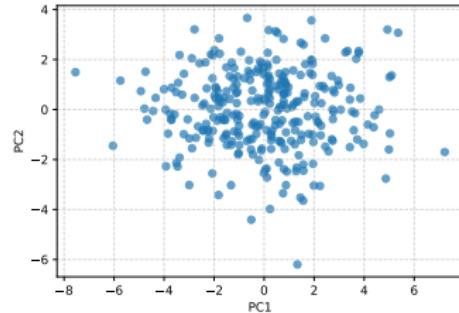
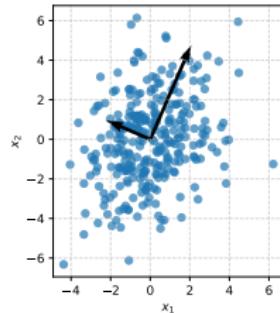
Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$
 - ⇒ $\mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V}$ with $\mathcal{V} \in \mathbb{R}^{d \times d'}$
 - If $d' < d \Rightarrow$ Dimensionality reduction
- Goal: Find *directions* that capture the most variance in the data
 - These *directions* constitute the coordinates of the new space



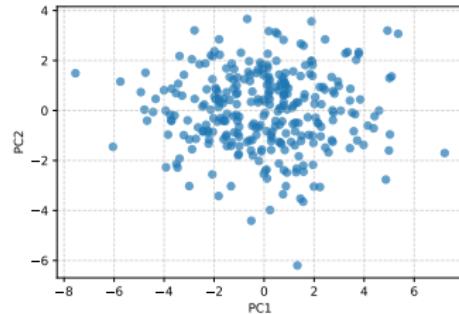
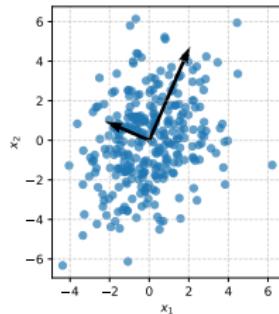
Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$
 - ⇒ $\mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V}$ with $\mathcal{V} \in \mathbb{R}^{d \times d'}$
 - If $d' < d \Rightarrow$ Dimensionality reduction
- Goal: Find *directions* that capture the most variance in the data
 - These *directions* constitute the coordinates of the new space
 - Correspond to the *eigenvectors* of the data



Principal Components Analysis

- Linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Datum $\mathbf{x} \in \mathbb{R}^d$ projected onto $\mathbb{R}^{d'}$ as: $(\mathbf{x}^T \mathbf{v}_1, \dots, \mathbf{x}^T \mathbf{v}_{d'})$ with $\mathbf{v} \in \mathbb{R}^d$
 - ⇒ $\mathbf{x}_{\text{proj}} = \mathbf{x}\mathcal{V}$ with $\mathcal{V} \in \mathbb{R}^{d \times d'}$
 - If $d' < d \Rightarrow$ Dimensionality reduction
- Goal: Find *directions* that capture the most variance in the data
 - These *directions* constitute the coordinates of the new space
 - Correspond to the *eigenvectors* of the data
 - Known as Principal Components



Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d$

Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d \Rightarrow \mathbf{D} \in \mathbb{R}^{|\mathcal{D}| \times d}$

Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d \Rightarrow \mathbf{D} \in \mathbb{R}^{|\mathcal{D}| \times d}$
→ Column-wise mean zero $\Rightarrow \sum_{i=1}^{|\mathcal{D}|} x_{ij} = 0 \quad \text{with } 1 \leq j \leq d$

Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d \Rightarrow \mathbf{D} \in \mathbb{R}^{|\mathcal{D}| \times d}$
→ Column-wise mean zero $\Rightarrow \sum_{i=1}^{|\mathcal{D}|} x_{ij} = 0 \quad \text{with } 1 \leq j \leq d$
- Covariance matrix (Σ): pairwise relationships between features

$$\Sigma = \frac{1}{|\mathcal{D}| - 1} \mathbf{D}^T \mathbf{D} \quad (\in \mathbb{R}^{d \times d})$$

Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d \Rightarrow \mathbf{D} \in \mathbb{R}^{|\mathcal{D}| \times d}$
→ Column-wise mean zero $\Rightarrow \sum_{i=1}^{|\mathcal{D}|} x_{ij} = 0 \quad \text{with } 1 \leq j \leq d$
- Covariance matrix (Σ): pairwise relationships between features

$$\Sigma = \frac{1}{|\mathcal{D}| - 1} \mathbf{D}^T \mathbf{D} \quad (\in \mathbb{R}^{d \times d})$$

- Variance on the projected space: $\text{Var}(\mathbf{D}\mathbf{v}) = \mathbf{v}^T \Sigma \mathbf{v}$

Mathematical formulation

- Initial assortment $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^{|\mathcal{D}|}$ with $\mathbf{x}_i \in \mathbb{R}^d \Rightarrow \mathbf{D} \in \mathbb{R}^{|\mathcal{D}| \times d}$
→ Column-wise mean zero $\Rightarrow \sum_{i=1}^{|\mathcal{D}|} x_{ij} = 0 \quad \text{with } 1 \leq j \leq d$
- Covariance matrix (Σ): pairwise relationships between features

$$\Sigma = \frac{1}{|\mathcal{D}| - 1} \mathbf{D}^T \mathbf{D} \quad (\in \mathbb{R}^{d \times d})$$

- Variance on the projected space: $\text{Var}(\mathbf{D}\mathbf{v}) = \mathbf{v}^T \Sigma \mathbf{v}$
→ Key condition: *eigenvector* must maximize $\text{Var}(\mathbf{D}\mathbf{v})$

Mathematical formulation

- First *eigenvector*: variance condition

Mathematical formulation

- First *eigenvector*: variance condition

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} \mathbf{v}^T \Sigma \mathbf{v}$$

Mathematical formulation

- First *eigenvector*: variance condition

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Second *eigenvector*: variance condition + orthogonal to \mathbf{v}_1

Mathematical formulation

- First *eigenvector*: variance condition

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Second *eigenvector*: variance condition + orthogonal to \mathbf{v}_1

$$\mathbf{v}_2 = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1} \mathbf{v}^T \Sigma \mathbf{v}$$

Mathematical formulation

- First *eigenvector*: variance condition

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Second *eigenvector*: variance condition + orthogonal to \mathbf{v}_1

$$\mathbf{v}_2 = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Third *eigenvector*: variance condition + orthogonal to $\mathbf{v}_1, \mathbf{v}_2$

$$\mathbf{v}_3 = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \{\mathbf{v}_1, \mathbf{v}_2\}} \mathbf{v}^T \Sigma \mathbf{v}$$

Mathematical formulation

- First *eigenvector*: variance condition

$$\mathbf{v}_1 = \arg \max_{\|\mathbf{v}\|=1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Second *eigenvector*: variance condition + orthogonal to \mathbf{v}_1

$$\mathbf{v}_2 = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \mathbf{v}_1} \mathbf{v}^T \Sigma \mathbf{v}$$

- Third *eigenvector*: variance condition + orthogonal to $\mathbf{v}_1, \mathbf{v}_2$

$$\mathbf{v}_3 = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \{\mathbf{v}_1, \mathbf{v}_2\}} \mathbf{v}^T \Sigma \mathbf{v}$$

- d' -th *eigenvector*: variance condition + orthogonal to $\{\mathbf{v}_i\}_{i=1}^{d'-1}$

$$\mathbf{v}_{d'} = \arg \max_{\|\mathbf{v}\|=1, \mathbf{v} \perp \{\mathbf{v}_i\}_{i=1}^{d'-1}} \mathbf{v}^T \Sigma \mathbf{v}$$

Mathematical formulation

- Vector $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{d'}$ \Rightarrow *eigenvectors* sorted by the **encoded variance**

Mathematical formulation

- Vector $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{d'}$ \Rightarrow *eigenvectors* sorted by the **encoded variance**
 \rightarrow Relates to the **compression degree** of the **Dimesionality Reduction**

Mathematical formulation

- Vector $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{d'}$ \Rightarrow *eigenvectors* sorted by the **encoded variance**
 \rightarrow Relates to the **compression degree** of the **Dimesionality Reduction**
- How much **information** are we **keeping**? \rightarrow Cumulative Variance

Mathematical formulation

- Vector $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^{d'} \Rightarrow$ *eigenvectors* sorted by the **encoded variance**
→ Relates to the **compression degree** of the **Dimesionality Reduction**
- How much **information** are we **keeping**? → **Cumulative Variance**

$$\text{CumVar}(d') = \frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{j=1}^d \lambda_j} \quad \text{where } \lambda_j = \mathbf{v}_j^T \Sigma \mathbf{v}_j$$

Example - Single component

Example - Two components

t-Stochastic Nearest Embedding

- Non-linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:

t-Stochastic Nearest Embedding

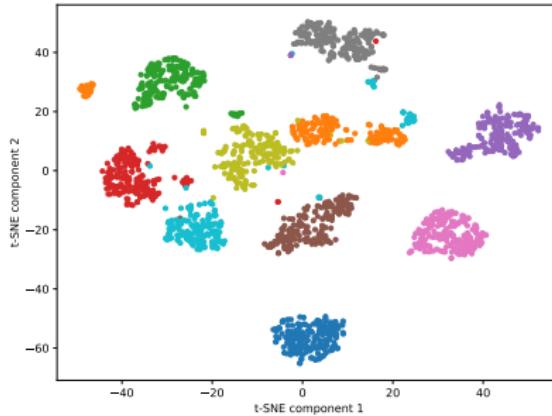
- Non-linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Typically, $d' = \{2, 3\}$
 - Most commonly used for visualization purposes

t-Stochastic Nearest Embedding

- Non-linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Typically, $d' = \{2, 3\}$
 - Most commonly used for visualization purposes
- **Key premise:** preserving local neighborhoods
 - Points close in \mathbb{R}^d remain close in $\mathbb{R}^{d'}$

t-Stochastic Nearest Embedding

- Non-linear projection from \mathbb{R}^d to $\mathbb{R}^{d'}$:
 - Typically, $d' = \{2, 3\}$
 - Most commonly used for visualization purposes
- **Key premise:** preserving local neighborhoods
 - Points close in \mathbb{R}^d remain close in $\mathbb{R}^{d'}$



Mathematical formulation

Models the problem from a **probabilistic perspective**:

Mathematical formulation

Models the problem from a probabilistic perspective:

1. Distribution $P \Rightarrow$ similarities in high-dimensional space
2. Distribution $Q \Rightarrow$ similarities in low-dimensional space

Mathematical formulation

Models the problem from a probabilistic perspective:

1. Distribution $P \Rightarrow$ similarities in high-dimensional space
2. Distribution $Q \Rightarrow$ similarities in low-dimensional space

Task: allocate elements in Q to resemble distribution in P

Mathematical formulation

Models the problem from a probabilistic perspective:

1. Distribution $P \Rightarrow$ similarities in high-dimensional space
2. Distribution $Q \Rightarrow$ similarities in low-dimensional space

Task: allocate elements in Q to resemble distribution in P

- Q is considerably less dimensional than P

Mathematical formulation

Models the problem from a probabilistic perspective:

1. Distribution $P \Rightarrow$ similarities in high-dimensional space
2. Distribution $Q \Rightarrow$ similarities in low-dimensional space

Task: allocate elements in Q to resemble distribution in P

- Q is considerably less dimensional than P
- Optimization goal:

$$\min \sum_{\mathbf{x} \in \mathcal{D}} P(\mathbf{x}) \log \frac{P(\mathbf{x})}{Q(\mathbf{x})}$$

Mathematical formulation

Models the problem from a probabilistic perspective:

1. Distribution $P \Rightarrow$ similarities in high-dimensional space
2. Distribution $Q \Rightarrow$ similarities in low-dimensional space

Task: allocate elements in Q to resemble distribution in P

- Q is considerably less dimensional than P
- Optimization goal:

$$\min \sum_{\mathbf{x} \in \mathcal{D}} P(\mathbf{x}) \log \frac{P(\mathbf{x})}{Q(\mathbf{x})} = \min D_{KL}(P||Q)$$

$D_{KL} \rightarrow$ Kullback-Leibler Divergence

Autoencoders

- Neural *disposition* devised for learning efficient representations

Autoencoders

- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding

Autoencoders

- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding
 - **Unsupervised** ⇒ Which is the target for **training the model?**

Autoencoders

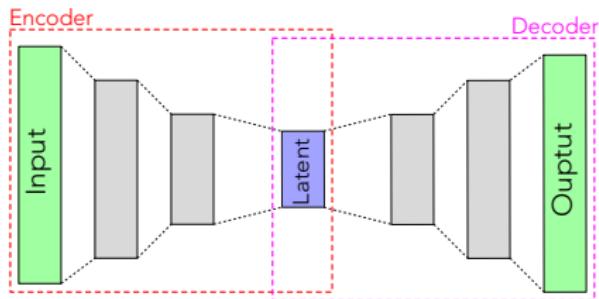
- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding
 - **Unsupervised** ⇒ Which is the **target** for **training the model?**
 - **Shallow** and **deep** models

Autoencoders

- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding
 - **Unsupervised** ⇒ Which is the **target** for **training the model?**
 - **Shallow** and **deep** models
- Exhibits **two parts**:

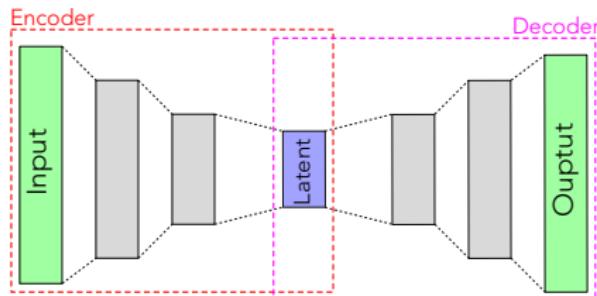
Autoencoders

- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding
 - **Unsupervised** ⇒ Which is the **target** for **training the model?**
 - **Shallow** and **deep** models
- Exhibits **two parts**:
 1. Encoder: **Reduces** the dimensionality of the data
 2. Decoder: **Recovers** the original dimensionality of the data



Autoencoders

- Neural *disposition* devised for learning efficient representations
 - Typically known as **latent representation** / embedding
 - **Unsupervised** ⇒ Which is the **target** for **training the model?**
 - **Shallow** and **deep** models
- Exhibits **two parts**:
 1. Encoder: **Reduces** the dimensionality of the data
 2. Decoder: **Recovers** the original dimensionality of the data



- Once trained, we are only **interested in the Encoder**
 - Decoder in this case is used for **training purposes**

Mathematical formulation

Goal: Learn two **mapping** functions:

Mathematical formulation

Goal: Learn two **mapping** functions:

- Encoder: $E_{\phi_E} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$

Mathematical formulation

Goal: Learn two **mapping** functions:

- **Encoder:** $E_{\phi_E} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
- **Decoder:** $D_{\phi_D} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$

Mathematical formulation

Goal: Learn two **mapping** functions:

- **Encoder:** $E_{\phi_E} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
- **Decoder:** $D_{\phi_D} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$

Given an **input** datum \mathbf{x} :

- **Embedded** representation: $\mathbf{x}^e = E_{\phi_E}(\mathbf{x})$
- **Reconstructed** datum: $\tilde{\mathbf{x}} = D_{\phi_D}(\mathbf{x}^e) = D_{\phi_D}(E_{\phi_E}(\mathbf{x}))$

Mathematical formulation

Goal: Learn two **mapping** functions:

- **Encoder:** $E_{\phi_E} : \mathbb{R}^d \rightarrow \mathbb{R}^{d'}$
- **Decoder:** $D_{\phi_D} : \mathbb{R}^{d'} \rightarrow \mathbb{R}^d$

Given an **input** datum \mathbf{x} :

- **Embedded** representation: $\mathbf{x}^e = E_{\phi_E}(\mathbf{x})$
- **Reconstructed** datum: $\tilde{\mathbf{x}} = D_{\phi_D}(\mathbf{x}^e) = D_{\phi_D}(E_{\phi_E}(\mathbf{x}))$

Optimization: minimize the difference between **input** and reconstruction

$$\arg \min_{\phi_E, \phi_D} \mathcal{L}(\mathbf{x}, \tilde{\mathbf{x}}) = \arg \min_{\phi_E, \phi_D} \mathcal{L}(\mathbf{x}, D_{\phi_D}(E_{\phi_E}(\mathbf{x})))$$

Final considerations

- Typically **symmetrical** but **not** a **necessary** condition
→ Decoder $D_{\phi_D}(\cdot)$ **undoes** the mapping by the encoder $E_{\phi_E}(\cdot)$

Final considerations

- Typically **symmetrical** but **not** a **necessary** condition
 - Decoder $D_{\phi_D}(\cdot)$ **undoes** the mapping by the encoder $E_{\phi_E}(\cdot)$
- There exist **other versions** of these structures:
 - **Variational AE**: Assumes data follows a particular **statistical distribution**
 - **Denoising AE**: Trained to **remove noise** from input data

Final considerations

- Typically **symmetrical** but **not** a **necessary** condition
 - Decoder $D_{\phi_D}(\cdot)$ **undoes** the mapping by the encoder $E_{\phi_E}(\cdot)$
- There exist **other versions** of these structures:
 - **Variational AE**: Assumes data follows a particular **statistical distribution**
 - **Denoising AE**: Trained to **remove noise** from input data
- Related to **PCA**:
 - **Single hidden layer with linear activation** ⇒ **Equivalent** approaches

Outline

① Introduction

② Clustering

Definition

Taxonomy

The k -means clustering method

Cluster determination techniques

③ Dimensionality reduction

Definition

Statistical approaches

Neural approaches

④ Other tasks

Other unsupervised learning tasks

1. Outlier detection

- Detect elements whose **features** remarkably **differ** from the rest
- **Strategies:** PCA error, Autoencoder, Isolation Forest

Other unsupervised learning tasks

1. Outlier detection

- Detect elements whose **features** remarkably **differ** from the rest
- **Strategies:** PCA error, Autoencoder, Isolation Forest

2. Generative modeling

- **Generate** new samples similar to the existing ones
- **Strategies:** Variational AE (decoder $D_{\phi_D}(\cdot)$)

Other unsupervised learning tasks

1. Outlier detection

- Detect elements whose **features** remarkably **differ** from the rest
- **Strategies:** PCA error, Autoencoder, Isolation Forest

2. Generative modeling

- **Generate** new samples similar to the existing ones
- **Strategies:** Variational AE (decoder $D_{\phi_D}(\cdot)$)

3. Motif discovery

- **Mining** patterns in time-series data collections

T6: Unsupervised learning

Fundamentos del Aprendizaje Automático

Curso 2025/2026

T7: Statistical model comparison

Fundamentos del Aprendizaje Automático

Curso 2025/2026

Structure

① Introduction

Contextualization

Statistical hypothesis test

② Pairwise classifier comparison

Paired t -test

Wilcoxon signed-rank test

③ Multiple classifier comparison

ANOVA

Friedman test

Post-hoc tests

Outline

① Introduction

Contextualization

Statistical hypothesis test

② Pairwise classifier comparison

Paired t -test

Wilcoxon signed-rank test

③ Multiple classifier comparison

ANOVA

Friedman test

Post-hoc tests

Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

Classifier A

Classifier B

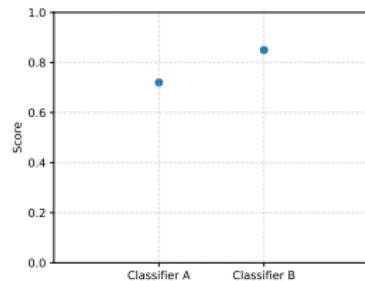
Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

1. Single partitioning: **train** \mathcal{T} and **test** \mathcal{S}

Classifier A

Case 1) $s_A \leftarrow \mathcal{S} \Leftrightarrow f_A(\mathcal{S}; \mathcal{T})$



(FAA)

Classifier B

Case 1) $s_B \leftarrow \mathcal{S} \Leftrightarrow f_B(\mathcal{S}; \mathcal{T})$

Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

1. Single partitioning: **train** \mathcal{T} and **test** \mathcal{S}
2. Cross-validation partitioning: **train** $[\mathcal{T}_1, \dots, \mathcal{T}_k]$ and **test** $[\mathcal{S}_1, \dots, \mathcal{S}_k]$

Classifier A

$$\text{Case 1)} s_A \leftarrow \mathcal{S} \Leftrightarrow f_A(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} s_{A1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_A(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Ak} \leftarrow \mathcal{S}_k \Leftrightarrow f_A(\mathcal{S}_k; \mathcal{T}_k)$$

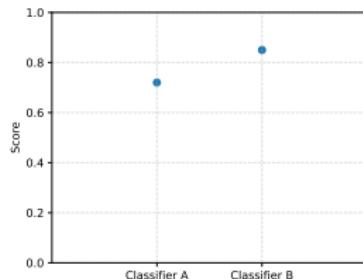
Classifier B

$$\text{Case 1)} s_B \leftarrow \mathcal{S} \Leftrightarrow f_B(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} s_{B1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_B(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Bk} \leftarrow \mathcal{S}_k \Leftrightarrow f_B(\mathcal{S}_k; \mathcal{T}_k)$$



Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

1. Single partitioning: **train** \mathcal{T} and **test** \mathcal{S}
2. Cross-validation partitioning: **train** $[\mathcal{T}_1, \dots, \mathcal{T}_k]$ and **test** $[\mathcal{S}_1, \dots, \mathcal{S}_k]$

Classifier A

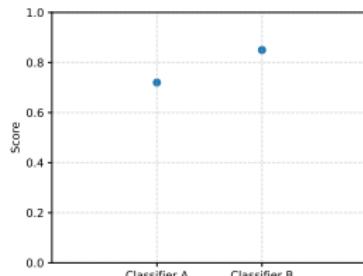
$$\text{Case 1)} \ s_A \leftarrow \mathcal{S} \Leftrightarrow f_A(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} \ s_{A1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_A(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Ak} \leftarrow \mathcal{S}_k \Leftrightarrow f_A(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_A \pm \sigma_A$$



(FAA)

Classifier B

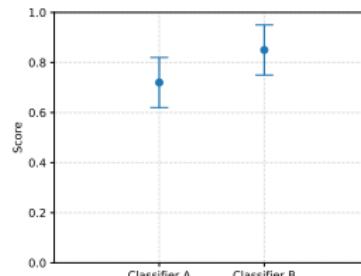
$$\text{Case 1)} \ s_B \leftarrow \mathcal{S} \Leftrightarrow f_B(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} \ s_{B1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_B(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Bk} \leftarrow \mathcal{S}_k \Leftrightarrow f_B(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_B \pm \sigma_B$$



T7: Statistical model comparison

Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

1. Single partitioning: **train** \mathcal{T} and **test** \mathcal{S}
2. Cross-validation partitioning: **train** $[\mathcal{T}_1, \dots, \mathcal{T}_k]$ and **test** $[\mathcal{S}_1, \dots, \mathcal{S}_k]$

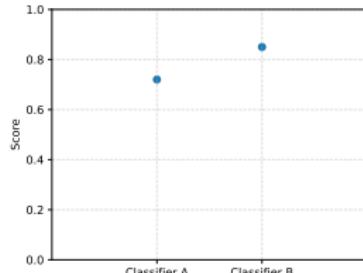
Classifier A

- Case 1) $s_A \leftarrow \mathcal{S} \Leftrightarrow f_A(\mathcal{S}; \mathcal{T})$
 Case 2) $s_{A1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_A(\mathcal{S}_1; \mathcal{T}_1)$

$$\dots$$

$$s_{Ak} \leftarrow \mathcal{S}_k \Leftrightarrow f_A(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_A \pm \sigma_A$$



(FAA)

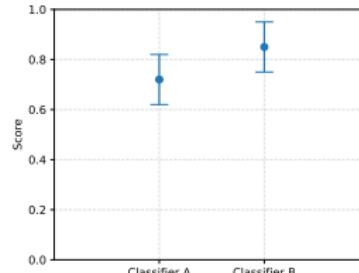
Classifier B

- Case 1) $s_B \leftarrow \mathcal{S} \Leftrightarrow f_B(\mathcal{S}; \mathcal{T})$
 Case 2) $s_{B1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_B(\mathcal{S}_1; \mathcal{T}_1)$

$$\dots$$

$$s_{Bk} \leftarrow \mathcal{S}_k \Leftrightarrow f_B(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_B \pm \sigma_B$$



T7: Statistical model comparison



Curso 2025/2026

4 / 44

Open question

Task: Given set $\mathcal{D} = \{(\mathbf{x}_i, \omega_i)\}_{i=1}^{|\mathcal{D}|}$, which is the **best classification** option?

1. Single partitioning: **train** \mathcal{T} and **test** \mathcal{S}
2. Cross-validation partitioning: **train** $[\mathcal{T}_1, \dots, \mathcal{T}_k]$ and **test** $[\mathcal{S}_1, \dots, \mathcal{S}_k]$
3. Several datasets: $\mathcal{D}_1, \dots, \mathcal{D}_M$

Classifier A

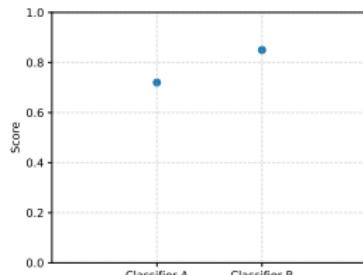
$$\text{Case 1)} s_A \leftarrow \mathcal{S} \Leftrightarrow f_A(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} s_{A1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_A(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Ak} \leftarrow \mathcal{S}_k \Leftrightarrow f_A(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_A \pm \sigma_A$$



(FAA)

Classifier B

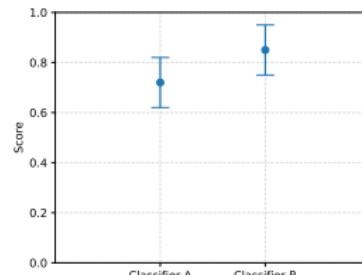
$$\text{Case 1)} s_B \leftarrow \mathcal{S} \Leftrightarrow f_B(\mathcal{S}; \mathcal{T})$$

$$\text{Case 2)} s_{B1} \leftarrow \mathcal{S}_1 \Leftrightarrow f_B(\mathcal{S}_1; \mathcal{T}_1)$$

...

$$s_{Bk} \leftarrow \mathcal{S}_k \Leftrightarrow f_B(\mathcal{S}_k; \mathcal{T}_k)$$

$$\bar{s}_B \pm \sigma_B$$



T7: Statistical model comparison



Curso 2025/2026

4 / 44

Open question

Model **performance** is **influenced** by several sources of **randomness**:

Open question

Model **performance** is **influenced** by several sources of **randomness**:

1. Model stochasticity
 - **Initialization** of the **weights**

Open question

Model **performance** is **influenced** by several sources of **randomness**:

1. Model stochasticity
 - **Initialization** of the **weights**
2. Data sampling
 - **Different** train/test **splits** often yields different **performance scores**

Open question

Model **performance** is **influenced** by several sources of **randomness**:

1. Model stochasticity
 - **Initialization** of the **weights**
2. Data sampling
 - **Different** train/test **splits** often yields different **performance scores**
3. Evaluation procedures
 - **Cross-validation** produces **correlated** estimates

Open question

Model **performance** is **influenced** by several sources of **randomness**:

1. Model stochasticity
 - **Initialization** of the **weights**
2. Data sampling
 - **Different** train/test **splits** often yields different **performance scores**
3. Evaluation procedures
 - **Cross-validation** produces **correlated** estimates

*Performance difference is **not necessarily** a real difference*

Statistical model comparison

Statistical methods for model comparison overcome this issue:

Statistical model comparison

Statistical methods for model comparison overcome this issue:

1. Determine whether observed differences are **statistically significant**
 - Observed difference is **random** or **real**
 - Essential for **rigorous** and **reproducible** evaluation in machine learning

Statistical model comparison

Statistical methods for model comparison overcome this issue:

1. Determine whether observed differences are **statistically significant**
 - Observed difference is **random** or **real**
 - Essential for **rigorous** and **reproducible** evaluation in machine learning
2. Quantify **uncertainty**
 - Typically in terms of *p-values*, *confidence intervals*, *test statistics*

Statistical model comparison

Statistical methods for model comparison overcome this issue:

1. Determine whether observed differences are **statistically significant**
 - Observed difference is **random** or **real**
 - Essential for **rigorous** and **reproducible** evaluation in machine learning
2. Quantify **uncertainty**
 - Typically in terms of *p-values*, *confidence intervals*, *test statistics*
3. Avoid **erroneous claims** based on **noisy** estimates

Statistical model comparison

Statistical methods for model comparison overcome this issue:

1. Determine whether observed differences are **statistically significant**
 - Observed difference is **random** or **real**
 - Essential for **rigorous** and **reproducible** evaluation in machine learning
2. Quantify **uncertainty**
 - Typically in terms of *p-values*, *confidence intervals*, *test statistics*
3. Avoid **erroneous claims** based on **noisy** estimates

Comparison	Parametric	Non-parametric
Pairwise	Paired <i>t</i> -test	Wilcoxon signed-rank
Multiple	ANOVA	Friedman + post-hoc

Key concepts

Key concepts

Always **two propositions** to explain the posed **scenario**:

Key concepts

Always **two propositions** to explain the posed **scenario**:

- Null hypothesis (H_0): **No relation** between sets
- Alternative hypothesis (H_1): There **exists a difference** between sets

Key concepts

Always **two propositions** to explain the posed **scenario**:

- Null hypothesis (H_0): **No relation** between sets
- Alternative hypothesis (H_1): There **exists a difference** between sets

The tests compute a **statistical significance** or *p*-value:

- **Probability** of obtaining **results** as **extreme as or more extreme** than the ones observed if H_0 were true
- The concept of **extremeness** depends on the test
- Smaller *p*-value \Rightarrow stronger evidence against H_0

Key concepts

Always **two propositions** to explain the posed **scenario**:

- Null hypothesis (H_0): **No relation** between sets
- Alternative hypothesis (H_1): There **exists a difference** between sets

The tests compute a **statistical significance** or *p*-value:

- **Probability** of obtaining **results** as **extreme as or more extreme** than the ones observed if H_0 were **true**
- The concept of **extremeness** depends on the test
- **Smaller** *p*-value \Rightarrow **stronger evidence** against H_0

The **significance level** or α acts as threshold to accept/reject H_0 :

- **Maximum probability** of **rejecting** H_0 when it is actually **true**
- The **lower**, the more **strict**

Procedure

Procedure

1. Select suitable statistical test

Procedure

1. Select suitable statistical test
2. Formulate hypotheses: H_0 and H_1

Procedure

1. Select suitable statistical test
2. Formulate hypotheses: H_0 and H_1
3. Compute the test statistic
4. Compute the *p*-value

Procedure

1. Select suitable statistical test
2. Formulate hypotheses: H_0 and H_1
3. Compute the *p-value*
4. Compute the *p-value*
5. Compare *p-value* to significance level α :
 - If $p < \alpha \Rightarrow$ **reject H_0**
 - If $p \geq \alpha \Rightarrow$ **do not reject H_0**

Consideration about data collections

Ideally, these **principles** should be followed:

Consideration about data collections

Ideally, these **principles** should be followed:

1. Providing **paired results**:

- Evaluation on the **same** data instances, folds, or datasets
- All models are **equally affected** by **random fluctuations** in data
- A **must** in some tests

Consideration about data collections

Ideally, these **principles** should be followed:

1. Providing paired results:

- Evaluation on the **same** data instances, folds, or datasets
- All models are **equally affected** by **random fluctuations** in data
- A **must** in some tests

2. Considering independent data assortments ($\mathcal{D}_1, \dots, \mathcal{D}_M$):

- Most statistical tests **assume independent** datasets
- Cross-validation strategies **violate** this independence **assumption**
 - Optimistic estimations
- Possible **solutions**:
 - a) Specific tests that compensate the bias
 - b) Averaging across **folds**

Outline

① Introduction

Contextualization

Statistical hypothesis test

② Pairwise classifier comparison

Paired t -test

Wilcoxon signed-rank test

③ Multiple classifier comparison

ANOVA

Friedman test

Post-hoc tests

Parametric case: the paired *t*-test

- Assumes **data** are approximately **normally distributed**
- **Types of tests:**

Type	Use case
One-sample t-test	Compare a sample mean to a known value
Independent two-sample t-test	Compare means of two independent groups
Paired t-test	Compare means of related or paired data

- We focus on the **paired *t*-test**:
 - Statistic t represents the ratio between **difference** and **variability**
 - Larger $|t|$ represents **more evidence** against H_0

Formulation

Consider the following conditions:

Formulation

Consider the following conditions:

- Consider two classifiers: f_A and f_B
- Consider M data assortments: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
- We have M performance scores for each classifier:
 - Classifier f_A : s_{A1}, \dots, s_{AM}
 - Classifier f_B : s_{B1}, \dots, s_{BM}

Formulation

Consider the following conditions:

- Consider two classifiers: f_A and f_B
- Consider M data assortments: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
- We have M performance scores for each classifier:
 - Classifier f_A : s_{A1}, \dots, s_{AM}
 - Classifier f_B : s_{B1}, \dots, s_{BM}

Procedure:

Formulation

Consider the following conditions:

- Consider two classifiers: f_A and f_B
- Consider M data assortments: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
- We have M performance scores for each classifier:
 - Classifier f_A : s_{A1}, \dots, s_{AM}
 - Classifier f_B : s_{B1}, \dots, s_{BM}

Procedure:

1. Dataset-wise performance difference: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$

Formulation

Consider the following conditions:

- Consider two classifiers: f_A and f_B
- Consider M data assortments: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
- We have M performance scores for each classifier:
 - Classifier f_A : s_{A1}, \dots, s_{AM}
 - Classifier f_B : s_{B1}, \dots, s_{BM}

Procedure:

1. Dataset-wise performance difference: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Compute mean (\bar{d}) and standard deviation (σ_d) as:

$$\bar{d} = \frac{1}{M} \sum_{i=1}^M d_i, \quad \sigma_d = \sqrt{\frac{1}{M-1} \sum_{i=1}^M (d_i - \bar{d})^2}$$

Formulation

Consider the following conditions:

- Consider two classifiers: f_A and f_B
- Consider M data assortments: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
- We have M performance scores for each classifier:
 - Classifier f_A : s_{A1}, \dots, s_{AM}
 - Classifier f_B : s_{B1}, \dots, s_{BM}

Procedure:

1. Dataset-wise performance difference: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Compute mean (\bar{d}) and standard deviation (σ_d) as:

$$\bar{d} = \frac{1}{M} \sum_{i=1}^M d_i, \quad \sigma_d = \sqrt{\frac{1}{M-1} \sum_{i=1}^M (d_i - \bar{d})^2}$$

3. Compute test statistic:

$$t = \frac{\bar{d}}{\sigma_d / \sqrt{M}}$$

Formulation

4. Obtain the value of the *t*-student distribution as t_{stu} :

- Single / both directions:
 - Single-tail when H_0 states $f_A > f_B$ or $f_A < f_B$
 - Two-tail when H_0 states $f_A = f_B$
- Degrees of freedom: $M - 1$
- Significance value α

Formulation

4. Obtain the value of the *t*-student distribution as t_{stu} :

- Single / both directions:
 - Single-tail when H_0 states $f_A > f_B$ or $f_A < f_B$
 - Two-tail when H_0 states $f_A = f_B$
- Degrees of freedom: $M - 1$
- Significance value α

5. Compare $|t|$ with t_{stu} :

- If $|t| > t_{\text{stu}}$: **reject** H_0
- If $|t| \leq t_{\text{stu}}$: **accept** H_0

Example

Dataset	Classifier A	Classifier B
\mathcal{D}_1	85	87
\mathcal{D}_2	70	68
\mathcal{D}_3	79	85
\mathcal{D}_4	78	75
\mathcal{D}_5	83	83

Is **Classifier A** equivalent to **Classifier B** for $\alpha = 0.05^1$?

¹ $t_{\text{stu}} \approx 2.776$

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

S2) Mean $\bar{d} = -0.6$

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

S2) Mean $\bar{d} = -0.6$

S2) Deviation $\sigma_d = 3.2$

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

S2) Mean $\bar{d} = -0.6$

S2) Deviation $\sigma_d = 3.2$

$$S3) t = \frac{-0.6}{3.2/\sqrt{5-1}} = -0.375$$

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

S2) Mean $\bar{d} = -0.6$

S2) Deviation $\sigma_d = 3.2$

$$S3) t = \frac{-0.6}{3.2/\sqrt{(5-1)}} = -0.375$$

S4) $t_{\text{stu}} = 2.776$

Example (solution)

Dataset	Classifier A	Classifier B	Difference
\mathcal{D}_1	85	87	-2
\mathcal{D}_2	70	68	+2
\mathcal{D}_3	79	85	-6
\mathcal{D}_4	78	75	+3
\mathcal{D}_5	83	83	0

S2) Mean $\bar{d} = -0.6$

S2) Deviation $\sigma_d = 3.2$

$$S3) \quad t = \frac{-0.6}{3.2/\sqrt{5-1}} = -0.375$$

S4) $t_{\text{stu}} = 2.776$

S5) $\underbrace{|-0.375|}_{|t|} \not> \underbrace{2.776}_{t_{\text{stu}}} \Rightarrow \text{Accept } H_0$

Non-parametric case: the Wilcoxon signed-rank test

- Wilcoxon tests \Rightarrow family of **non-parametric** alternatives to t -tests

Non-parametric case: the Wilcoxon signed-rank test

- Wilcoxon tests \Rightarrow family of **non-parametric** alternatives to t -tests

Wilcoxon test	Parametric equivalent
Signed-rank	Paired t -test
Rank-sum	Independent two-sample t -test

Non-parametric case: the Wilcoxon signed-rank test

- Wilcoxon tests \Rightarrow family of **non-parametric** alternatives to t -tests

Wilcoxon test	Parametric equivalent
Signed-rank	Paired t -test
Rank-sum	Independent two-sample t -test

- Focus on **Wilcoxon signed-rank test** for paired data
 - No normality assumption
 - Robust to outliers

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Remove **zeroes**

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Remove **zeroes**
3. **Rank the absolute differences** $|d_i|$ (smallest to largest)
 - Assign **sign** to the ranked differences

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Remove **zeroes**
3. **Rank the absolute differences** $|d_i|$ (smallest to largest)
 - Assign **sign** to the ranked differences
4. Sum **positive** (W^+) and **negative** (W^-) ranks
 - Test **statistic** $W = \min\{W^+, W^-\}$

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Remove **zeroes**
3. **Rank the absolute differences** $|d_i|$ (smallest to largest)
 - Assign **sign** to the ranked differences
4. Sum **positive** (W^+) and **negative** (W^-) ranks
 - Test **statistic** $W = \min\{W^+, W^-\}$
5. Obtain the **critical value** from the **critical values** table:
 - **Single / both** directions:
 - **Single-tailed** when H_0 states $f_A > f_B$ or $f_A < f_B$
 - **Two-tailed** when H_0 states $f_A = f_B$
 - Significance value α

Procedure

1. Dataset-wise **performance difference**: $d_i = s_{Ai} - s_{Bi}$, $i = 1, \dots, M$
2. Remove **zeroes**
3. **Rank the absolute differences** $|d_i|$ (smallest to largest)
 - Assign **sign** to the ranked differences
4. Sum **positive** (W^+) and **negative** (W^-) ranks
 - Test **statistic** $W = \min\{W^+, W^-\}$
5. Obtain the **critical value** from the **critical values** table:
 - **Single / both** directions:
 - **Single-tailed** when H_0 states $f_A > f_B$ or $f_A < f_B$
 - **Two-tailed** when H_0 states $f_A = f_B$
 - Significance value α
6. **Reject** H_0 if:
 - **Single-tailed** test: $W \leq W_\alpha$
 - **Two-tailed** test: $W \leq W_{\alpha/2}$

Procedure

n	Two-Tailed			One-Tailed		
	$\alpha = .10$	$\alpha = .05$	$\alpha = .01$	$\alpha = .10$	$\alpha = .05$	$\alpha = .01$
5	0	0	0	0	0	0
6	2	2	0	2	2	1
7	3	3	1	4	3	2
8	5	4	2	6	5	3
9	8	6	3	8	7	5
10	10	8	5	11	9	6
11	13	10	7	14	11	8
12	17	13	9	18	14	11
13	21	16	11	22	18	13
14	25	19	14	26	21	16
15	30	23	17	31	25	19
16	35	27	21	36	29	22
17	40	31	24	41	33	26
18	46	36	28	47	38	29
19	52	40	32	53	42	33
20	59	45	37	60	47	38
21	66	51	41	67	53	43
22	73	56	46	74	58	48
23	81	62	51	82	64	53
24	89	68	56	90	70	58
25	98	75	62	99	77	64

Example

Dataset	Classifier A	Classifier B
\mathcal{D}_1	87	85
\mathcal{D}_2	68	70
\mathcal{D}_3	85	79
\mathcal{D}_4	75	78
\mathcal{D}_5	83	83
\mathcal{D}_6	90	85

Is **Classifier A** equivalent to **Classifier B** for $\alpha = 0.1$?

Example (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	87	85	+2	1.5
\mathcal{D}_2	68	70	-2	1.5
\mathcal{D}_3	85	79	+6	5
\mathcal{D}_4	75	78	-3	3
\mathcal{D}_5	83	83	0	-
\mathcal{D}_6	90	85	+5	4

Example (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	87	85	+2	1.5
\mathcal{D}_2	68	70	-2	1.5
\mathcal{D}_3	85	79	+6	5
\mathcal{D}_4	75	78	-3	3
\mathcal{D}_5	83	83	0	-
\mathcal{D}_6	90	85	+5	4

$$\text{S4)} W^+ = 1.5 + 4 + 5 = 10.5$$

$$\text{S4)} W^- = 1.5 + 3 = 4.5$$

Example (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	87	85	+2	1.5
\mathcal{D}_2	68	70	-2	1.5
\mathcal{D}_3	85	79	+6	5
\mathcal{D}_4	75	78	-3	3
\mathcal{D}_5	83	83	0	-
\mathcal{D}_6	90	85	+5	4

$$\text{S4)} \quad W^+ = 1.5 + 4 + 5 = 10.5$$

$$\text{S4)} \quad W^- = 1.5 + 3 = 4.5$$

$$\text{S4)} \quad W = \min(W^+, W^-) = \min(10.5, 4.5) = 4.5$$

Example (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	87	85	+2	1.5
\mathcal{D}_2	68	70	-2	1.5
\mathcal{D}_3	85	79	+6	5
\mathcal{D}_4	75	78	-3	3
\mathcal{D}_5	83	83	0	-
\mathcal{D}_6	90	85	+5	4

$$\text{S4)} \quad W^+ = 1.5 + 4 + 5 = 10.5$$

$$\text{S4)} \quad W^- = 1.5 + 3 = 4.5$$

$$\text{S4)} \quad W = \min(W^+, W^-) = \min(10.5, 4.5) = 4.5$$

$$\text{S5)} \quad W_{\alpha/2} = 0$$

Example (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	87	85	+2	1.5
\mathcal{D}_2	68	70	-2	1.5
\mathcal{D}_3	85	79	+6	5
\mathcal{D}_4	75	78	-3	3
\mathcal{D}_5	83	83	0	-
\mathcal{D}_6	90	85	+5	4

S4) $W^+ = 1.5 + 4 + 5 = 10.5$

S4) $W^- = 1.5 + 3 = 4.5$

S4) $W = \min(W^+, W^-) = \min(10.5, 4.5) = 4.5$

S5) $W_{\alpha/2} = 0$

S6) $\underbrace{4.5}_{W} \not\leq \underbrace{0}_{W_{\alpha/2}} \Rightarrow \textbf{Accept } H_0$

Exercise

Dataset	Classifier A	Classifier B
\mathcal{D}_1	95	80
\mathcal{D}_2	88	70
\mathcal{D}_3	90	80
\mathcal{D}_4	85	60
\mathcal{D}_5	92	78
\mathcal{D}_6	87	80
\mathcal{D}_7	91	85
\mathcal{D}_8	95	70
\mathcal{D}_9	82	60
\mathcal{D}_{10}	96	80
\mathcal{D}_{11}	88	85
\mathcal{D}_{12}	90	78
\mathcal{D}_{13}	85	80
\mathcal{D}_{14}	92	78
\mathcal{D}_{15}	87	80

Exercise (solution)

Dataset	Classifier A	Classifier B	Difference	Rank
\mathcal{D}_1	95	80	+15	1
\mathcal{D}_2	88	70	+18	2
\mathcal{D}_3	90	80	+10	3
\mathcal{D}_4	85	60	+25	4.5
\mathcal{D}_5	92	78	+14	4.5
\mathcal{D}_6	87	80	+7	6
\mathcal{D}_7	91	85	+6	7
\mathcal{D}_8	95	70	+25	8.5
\mathcal{D}_9	82	60	+22	8.5
\mathcal{D}_{10}	96	80	+16	10
\mathcal{D}_{11}	88	85	+3	11
\mathcal{D}_{12}	90	78	+12	12
\mathcal{D}_{13}	85	80	+5	13
\mathcal{D}_{14}	92	78	+14	14.5
\mathcal{D}_{15}	87	80	+7	14.5

- $W^+ = 120, W^- = 0 \Rightarrow \min(W^+, W^-) = 0$
- $n = 15$
- If $\alpha = 0.1 \Rightarrow W_{\alpha/2} = 23 \Rightarrow W < W_{\alpha/2} \Rightarrow \text{Reject}$

Exercise (solution)

One-tailed test:

- Is **A** > **B**?
- $W \leq W_\alpha \Rightarrow 0 \leq 25 \Rightarrow \text{Reject!}$

Classifier **A** is significantly better than **B** (with $\alpha = 0.1$)

Outline

① Introduction

Contextualization

Statistical hypothesis test

② Pairwise classifier comparison

Paired t -test

Wilcoxon signed-rank test

③ Multiple classifier comparison

ANOVA

Friedman test

Post-hoc tests

Motivation

- **Premise:** comparing C different classifiers $\Rightarrow f_1, \dots, f_C$
→ Generalization of the pairwise comparison

Motivation

- **Premise:** comparing C different classifiers $\Rightarrow f_1, \dots, f_C$
→ Generalization of the pairwise comparison
- Pairwise tests are not directly applicable in this case
→ Scenario may be adapted

Motivation

- **Premise:** comparing C different classifiers $\Rightarrow f_1, \dots, f_C$
→ Generalization of the pairwise comparison
- Pairwise tests are not directly applicable in this case
→ Scenario may be adapted
- Possible approaches:
 1. One-VS-one comparison
 2. Specific multiple comparison tests

One-VS-one comparison

One-VS-one comparison

- Subdividing the multiple comparison into $\binom{C}{2}$ binary problems

One-VS-one comparison

- Subdividing the multiple comparison into $\binom{C}{2}$ binary problems
- Allows using pairwise comparison methods

One-VS-one comparison

- Subdividing the multiple comparison into $\binom{C}{2}$ binary problems
- Allows using pairwise comparison methods
- Difficult to state a global optimum for the task

One-VS-one comparison

- Subdividing the multiple comparison into $\binom{C}{2}$ binary problems
- Allows using pairwise comparison methods
- Difficult to state a global optimum for the task
- Consider the following scenario:
 - Four classifiers: **A, B, C, D**
 - Pairwise comparison: Wilcoxon signed-rank test

One-VS-one comparison

- Subdividing the multiple comparison into $\binom{C}{2}$ binary problems
- Allows using pairwise comparison methods
- Difficult to state a global optimum for the task

- Consider the following scenario:
 - Four classifiers: A, B, C, D
 - Pairwise comparison: Wilcoxon signed-rank test

Classifier	Classifier			
	A	B	C	D
A	—	=	>	>
B	=	—	=	<
C	<	=	—	=
D	<	>	=	—

Multiple comparison tests

- Compare **all populations** at the same time
 - Easy to state **global optima**

Multiple comparison tests

- Compare **all populations** at the same time
 - Easy to state **global optima**
- Typically, a **two-stage** analysis:
 1. Initial process to state **whether** populations differ among them
 2. Post-hoc analysis to state **which** populations differ

Multiple comparison tests

- Compare **all populations** at the same time
 - Easy to state **global optima**
- Typically, a **two-stage** analysis:
 1. Initial process to state **whether populations differ** among them
 2. Post-hoc analysis to state **which populations differ**
- Consider the **following conditions**:
 - Set of **C classifiers**: f_1, \dots, f_C
 - Collection **M data assortments**: $\mathcal{D}_1, \dots, \mathcal{D}_M$ with $\mathcal{D}_i = \mathcal{T}_i \cup \mathcal{S}_i$
 - Matrix of **$M \times C$ values**

The parametric case: ANOVA

- Acronym for *Analisis of Variance*

The parametric case: ANOVA

- Acronym for *Analisis of Variance*
- Analyzes whether three or more models (significantly) differ in their mean performance
 - Null hypothesis (H_0): All population means are equal
 - Relies on the F-test

The parametric case: ANOVA

- Acronym for *Analisis of Variance*
- Analyzes whether three or more models (significantly) differ in their mean performance
 - Null hypothesis (H_0): All population means are equal
 - Relies on the F-test
- Assumptions on the measurements to be compared:
 - Follow a normal distribution
 - Are independent among them

The non-parametric case: Friedman test

- Non-parametric alternative to the ANOVA test
→ No normality assumption

The non-parametric case: Friedman test

- Non-parametric alternative to the ANOVA test
 - No normality assumption
- Relies on ranking procedures
 - Requires paired measurements

The non-parametric case: Friedman test

- Non-parametric alternative to the ANOVA test
 - No normality assumption
- Relies on ranking procedures
 - Requires paired measurements
- States whether there exist differences among the measurements
 - Post-hoc analysis to state which are the different measurements

Friedman test - Procedure

Friedman test - Procedure

1. Rank models for each assortment $1 \leq i \leq M$:

- Sort $f_1, \dots, f_C \Rightarrow$ Best (pos. #1) to worst (pos. #C)

Friedman test - Procedure

1. Rank models for each assortment $1 \leq i \leq M$:
 - Sort $f_1, \dots, f_C \Rightarrow$ Best (pos. #1) to worst (pos. #C)
2. Average rank per model:

$$\bar{R}_j = \frac{1}{M} \sum_{i=1}^M R_{ij} \quad \text{with} \quad 1 \leq j \leq C$$

Friedman test - Procedure

1. Rank models for each assortment $1 \leq i \leq M$:
 - Sort $f_1, \dots, f_C \Rightarrow$ Best (pos. #1) to worst (pos. #C)
2. Average rank per model:

$$\bar{R}_j = \frac{1}{M} \sum_{i=1}^M R_{ij} \quad \text{with} \quad 1 \leq j \leq C$$

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot M}{C \cdot (C + 1)} \left[\sum_{j=1}^C \bar{R}_j^2 \right] - 3 \cdot M \cdot (C + 1)$$

Friedman test - Procedure

1. Rank models for each assortment $1 \leq i \leq M$:
 - Sort $f_1, \dots, f_C \Rightarrow$ Best (pos. #1) to worst (pos. #C)
2. Average rank per model:

$$\bar{R}_j = \frac{1}{M} \sum_{i=1}^M R_{ij} \quad \text{with} \quad 1 \leq j \leq C$$

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot M}{C \cdot (C + 1)} \left[\sum_{j=1}^C \bar{R}_j^2 \right] - 3 \cdot M \cdot (C + 1)$$

4. Obtain chi-square critical value: $\chi_{\alpha, C-1}^2$ ($\alpha \rightarrow$ Significance threshold)

Friedman test - Procedure

1. Rank models for each assortment $1 \leq i \leq M$:
 - Sort $f_1, \dots, f_C \Rightarrow$ Best (pos. #1) to worst (pos. #C)
2. Average rank per model:

$$\bar{R}_j = \frac{1}{M} \sum_{i=1}^M R_{ij} \quad \text{with} \quad 1 \leq j \leq C$$

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot M}{C \cdot (C + 1)} \left[\sum_{j=1}^C \bar{R}_j^2 \right] - 3 \cdot M \cdot (C + 1)$$

4. Obtain chi-square critical value: $\chi_{\alpha, C-1}^2$ ($\alpha \rightarrow$ Significance threshold)
5. Reject H_0 if $\chi_F^2 > \chi_{\alpha, C-1}^2$

Procedure - Chi-square critical value table

$C - 1$	$\alpha = 0.10$	$\alpha = 0.05$	$\alpha = 0.01$
1	2.706	3.841	6.635
2	4.605	5.991	9.210
3	6.251	7.815	11.345
4	7.779	9.488	13.277
5	9.236	11.070	15.086
6	10.645	12.592	16.812
7	12.017	14.067	18.475
8	13.362	15.507	20.090
9	14.684	16.919	21.666
10	15.987	18.307	23.209
11	17.275	19.675	24.725
12	18.549	21.026	26.217
13	19.812	22.362	27.688
14	21.064	23.685	29.141
15	22.307	24.996	30.578
16	23.542	26.296	32.000
17	24.769	27.587	33.409
18	25.989	28.869	34.805
19	27.204	30.144	36.191
20	28.412	31.410	37.566
21	29.615	32.671	38.932
22	30.813	33.924	40.289
23	32.007	35.172	41.638
24	33.196	36.415	42.980
25	34.382	37.652	44.314

Example

Dataset	Classifiers			
	1	2	3	4
\mathcal{D}_1	70	73	78	82
\mathcal{D}_2	68	76	75	80
\mathcal{D}_3	72	74	79	85
\mathcal{D}_4	69	72	78	81
\mathcal{D}_5	71	74	77	82
\mathcal{D}_6	67	70	73	79

Are there any **statistical differences** among the classifiers considering a **significance threshold** of $\alpha = 0.05$?

Example (solution)

1. Rank models for each assortment:

Dataset	Classifiers			
	1	2	3	4
\mathcal{D}_1	70 (4)	73 (3)	78 (2)	82 (1)
\mathcal{D}_2	68 (4)	76 (2)	75 (3)	80 (1)
\mathcal{D}_3	72 (4)	74 (3)	79 (2)	85 (1)
\mathcal{D}_4	69 (4)	72 (3)	78 (2)	81 (1)
\mathcal{D}_5	71 (4)	74 (3)	77 (2)	82 (1)
\mathcal{D}_6	67 (4)	70 (3)	73 (2)	79 (1)

Example (solution)

1. Rank models for each assortment:

Dataset	Classifiers			
	1	2	3	4
\mathcal{D}_1	70 (4)	73 (3)	78 (2)	82 (1)
\mathcal{D}_2	68 (4)	76 (2)	75 (3)	80 (1)
\mathcal{D}_3	72 (4)	74 (3)	79 (2)	85 (1)
\mathcal{D}_4	69 (4)	72 (3)	78 (2)	81 (1)
\mathcal{D}_5	71 (4)	74 (3)	77 (2)	82 (1)
\mathcal{D}_6	67 (4)	70 (3)	73 (2)	79 (1)

2. Average rank per model:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

Example (solution)

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot 6}{4 \cdot (4 + 1)} [4^2 + 2.83^2 + 2.17^2 + 1^2] - 3 \cdot 6 \cdot (4 + 1) = 17$$

Example (solution)

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot 6}{4 \cdot (4 + 1)} [4^2 + 2.83^2 + 2.17^2 + 1^2] - 3 \cdot 6 \cdot (4 + 1) = 17$$

4. Chi-square critical value $\Rightarrow \chi_{\alpha, C-1}^2 = \chi_{0.05, 4-1}^2 = 7.815$

Example (solution)

3. Friedman statistic:

$$\chi_F^2 = \frac{12 \cdot 6}{4 \cdot (4 + 1)} [4^2 + 2.83^2 + 2.17^2 + 1^2] - 3 \cdot 6 \cdot (4 + 1) = 17$$

4. Chi-square critical value $\Rightarrow \chi_{\alpha, C-1}^2 = \chi_{0.05, 4-1}^2 = 7.815$

5. Check possible H_0 rejection $\rightarrow \chi_F^2 > \chi_{\alpha, C-1}^2$:
 $\rightarrow 17 > 7.815 \rightarrow H_0$ rejected!

Post-hoc analysis

- Required to clarify the measurement/s that significantly differ
 - Previous analysis proved a statistical difference among them

Post-hoc analysis

- Required to clarify the measurement/s that significantly differ
 - Previous analysis proved a statistical difference among them
- Two methods that rely on the principle of **Critical Difference**:
 1. Nemenyi test:
 - Compares all measurements among them
 - Which specific pairs of models differ
 2. Bonferroni-Dunn test:
 - Compares all measurements against a reference
 - Comparison against a single control model

Nemenyi test - Procedure

1. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$):
 - Compute the **mean rank** across assortments for each classifier
 - Same as Friedman test

Nemenyi test - Procedure

1. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$):
 - Compute the **mean rank** across assortments for each classifier
 - Same as Friedman test
2. Compute the **Critical Difference**:

$$CD = q_\alpha(C) \cdot \sqrt{\frac{C \cdot (C + 1)}{6 \cdot M}}$$

→ q_α : Studentized Range critical values

Nemenyi test - Procedure

1. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$):
 - Compute the **mean rank** across assortments for each classifier
 - Same as Friedman test
2. Compute the **Critical Difference**:

$$CD = q_\alpha(C) \cdot \sqrt{\frac{C \cdot (C + 1)}{6 \cdot M}}$$

→ q_α : Studentized Range critical values

3. **Pairwise comparison** of the models ($1 \leq i, j \leq C$ with $i \neq j$):
 - **Hypotheses** posed:
 - $H_0: f_i = f_j$
 - $H_1: f_i \neq f_j$
 - **Reject** condition: $|\bar{R}_i - \bar{R}_j| > CD$

Nemenyi test - q_α

C	$\alpha = 0.10$	$\alpha = 0.05$	$\alpha = 0.01$
2	1.960	2.241	2.807
3	2.052	2.343	2.949
4	2.108	2.403	3.020
5	2.146	2.444	3.069
6	2.174	2.475	3.105
7	2.195	2.499	3.133
8	2.211	2.518	3.157
9	2.224	2.534	3.176
10	2.235	2.548	3.192
11	2.244	2.559	3.206
12	2.252	2.569	3.218
13	2.259	2.577	3.228
14	2.265	2.584	3.237
15	2.270	2.590	3.245
16	2.275	2.596	3.252
17	2.279	2.601	3.258
18	2.283	2.605	3.264
19	2.286	2.609	3.269
20	2.289	2.613	3.274

Example

Dataset	Classifiers			
	1	2	3	4
\mathcal{D}_1	70	73	78	82
\mathcal{D}_2	68	76	75	80
\mathcal{D}_3	72	74	79	85
\mathcal{D}_4	69	72	78	81
\mathcal{D}_5	71	74	77	82
\mathcal{D}_6	67	70	73	79

Example (solution)

1. Average rank per model:

$$\text{- } \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$\text{- } \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$\text{- } \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$\text{- } \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

Example (solution)

1. Average rank per model:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

2. Compute de Critical Difference:

$$CD = q_{\alpha=0.05}(C=4) \cdot \sqrt{\frac{4 \cdot (4+1)}{6 \cdot 6}} = 2.403 \cdot 0.745 = 1.79$$

Example (solution)

1. Average rank per model:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

2. Compute de Critical Difference:

$$CD = q_{\alpha=0.05}(C=4) \cdot \sqrt{\frac{4 \cdot (4+1)}{6 \cdot 6}} = 2.403 \cdot 0.745 = 1.79$$

3. Pairwise comparison:

Classifier	Classifiers			
	1	2	3	4
1	—	1.17 (✗)	1.83 (✓)	3.00 (✓)
2	1.17 (✗)	—	0.66 (✗)	1.83 (✓)
3	1.83 (✓)	0.66 (✗)	—	1.17 (✗)
4	3.00 (✓)	1.83 (✓)	1.17 (✗)	—

Bonferroni-Dunn - Procedure

1. Select the **reference** case $\Rightarrow f_{\text{ref}}$

Bonferroni-Dunn - Procedure

1. Select the **reference** case $\Rightarrow f_{\text{ref}}$
2. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$)

Bonferroni-Dunn - Procedure

1. Select the **reference** case $\Rightarrow f_{\text{ref}}$
2. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$)
3. Compute the **Critical Difference**:

$$CD = q_{\alpha/C-1} \cdot \sqrt{\frac{C \cdot (C + 1)}{6 \cdot M}}$$

Bonferroni-Dunn - Procedure

1. Select the **reference** case $\Rightarrow f_{\text{ref}}$
2. Obtain the **average ranks** (\bar{R}_i with $1 \leq i \leq C$)
3. Compute the **Critical Difference**:

$$CD = q_{\alpha/C-1} \cdot \sqrt{\frac{C \cdot (C+1)}{6 \cdot M}}$$

4. Compare with the **reference** case:
 - **Hypotheses** posed:
 - $H_0: f_i = f_{\text{ref}}$
 - $H_1: f_i \neq f_{\text{ref}}$
 - **Reject** condition: $|\bar{R}_i - \bar{R}_{\text{ref}}| > CD$

Bonferroni-Dunn test - $q_{\alpha/C-1}$

C	$\alpha = 0.10$	$\alpha = 0.05$	$\alpha = 0.01$
3	1.645	1.960	2.576
4	1.282	1.645	2.326
5	1.163	1.533	2.241
6	1.095	1.476	2.192
7	1.054	1.440	2.160
8	1.027	1.414	2.136
9	1.006	1.395	2.120
10	0.990	1.380	2.107
11	0.977	1.368	2.096
12	0.966	1.357	2.088
13	0.957	1.349	2.081
14	0.949	1.341	2.075
15	0.943	1.335	2.070
16	0.937	1.329	2.066
17	0.932	1.324	2.062
18	0.928	1.320	2.058
19	0.924	1.316	2.055
20	0.921	1.312	2.053

Example

Dataset	Classifiers			
	1	2	3	4
\mathcal{D}_1	70	73	78	82
\mathcal{D}_2	68	76	75	80
\mathcal{D}_3	72	74	79	85
\mathcal{D}_4	69	72	78	81
\mathcal{D}_5	71	74	77	82
\mathcal{D}_6	67	70	73	79

Which is the result of the **Bonferroni-Dunn** test with $\alpha = 0.05$ considering as reference **Classifier 4**?

Example (solution)

1. Reference case $\rightarrow f_4$

Example (solution)

1. Reference case $\rightarrow f_4$
2. Obtain the average ranks:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

Example (solution)

1. Reference case $\rightarrow f_4$

2. Obtain the average ranks:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

3. Compute the Critical Difference:

$$CD = q_{0.05/4-1} \cdot \sqrt{\frac{4 \cdot (4+1)}{6 \cdot 6}} = 1.960 \cdot 0.745 = 1.46$$

Example (solution)

1. Reference case $\rightarrow f_4$

2. Obtain the average ranks:

$$- \bar{R}_1 = \frac{4+4+4+4+4+4}{6} = 4$$

$$- \bar{R}_3 = \frac{2+3+2+2+2+2}{6} = 2.17$$

$$- \bar{R}_2 = \frac{3+2+3+3+3+3}{6} = 2.83$$

$$- \bar{R}_4 = \frac{1+1+1+1+1+1}{6} = 1$$

3. Compute the Critical Difference:

$$CD = q_{0.05/4-1} \cdot \sqrt{\frac{4 \cdot (4+1)}{6 \cdot 6}} = 1.960 \cdot 0.745 = 1.46$$

4. Compare with f_4 :

$$f_1) |\bar{R}_1 - \bar{R}_4| > CD \Rightarrow |4 - 1| > 1.46 \Rightarrow 3 > 1.46 \checkmark$$

$$f_2) |\bar{R}_2 - \bar{R}_4| > CD \Rightarrow |2.83 - 1| > 1.46 \Rightarrow 1.83 > 1.46 \checkmark$$

$$f_3) |\bar{R}_3 - \bar{R}_4| > CD \Rightarrow |2.17 - 1| > 1.46 \Rightarrow 1.17 > 1.46 \times$$

T7: Statistical model comparison

Fundamentos del Aprendizaje Automático

Curso 2025/2026