Machine Learning and Data Mining

Last update: 01 November 2023

Academic Year 2023 – 2024

Alma Mater Studiorum · University of Bologna

Contents

Acronyms

BI Business Intelligence

CDC Change Data Capture

CRISP-DM Cross Industry Standard Process for Data Mining

DFM Dimensional Fact Model

DM Data Mart

DSS Decision Support System

DWH Data Warehouse

EIS Executive Information System

ERP Enterprise Resource Planning

ETL Extraction, Transformation, Loading

MIS Management Information System

OLAP Online Analysical Processing

OLTP Online Transaction Processing

1 Introduction

1.1 Data

Data Collection of raw values. Data

Information Organized data (e.g. relationships, context, ...). Information

Knowledge Understanding information. Knowledge

1.1.1 Data sources

Transaction Business event that generates or modifies data in an information system (e.g. database). Transaction

Signal Measure produced by a sensor. Signal

External subjects

1.1.2 Software

Online Transaction Processing (OLTP) Class of programs to support transaction oriented applications and data storage. Suitable for real-time applications. Online Transaction Processing

Enterprise Resource Planning (ERP) Integrated system to manage all the processes of a business. Uses a shared database for all applications. Suitable for real-time applications. Enterprise Resource Planning

1.1.3 Insight

Decision can be classified as:

Structured Established and well understood situations. What is needed is known. Structured decision

Unstructured Unplanned and unclear situations. What is needed for the decision is unknown. Unstructured decision

Different levels of insight can be extracted by:

Management Information System (MIS) Standardized reporting system built on existing OLTP. Used for structured decisions. Management Information System

Decision Support System (DSS) Analytical system to provide support for unstructured decisions. Decision Support System

Executive Information System (EIS) Formulate high level decisions that impact the organization. Executive Information System

Online Analysical Processing (OLAP) Grouped analysis of multidimensional data. Involves large amount of data. Online Analysical Processing

Business Intelligence (BI) Applications, infrastructure, tools and best practices to analyze information. Business Intelligence

Big data Large and/or complex and/or fast changing collection of data that traditional DBMSs are unable to process. Big data

Structured e.g. relational tables.

Unstructured e.g. videos.

Semi-structured e.g. JSON.

Anaylitics Structured decision driven by data. Anaylitics

Data mining Discovery process for unstructured decisions. Data mining

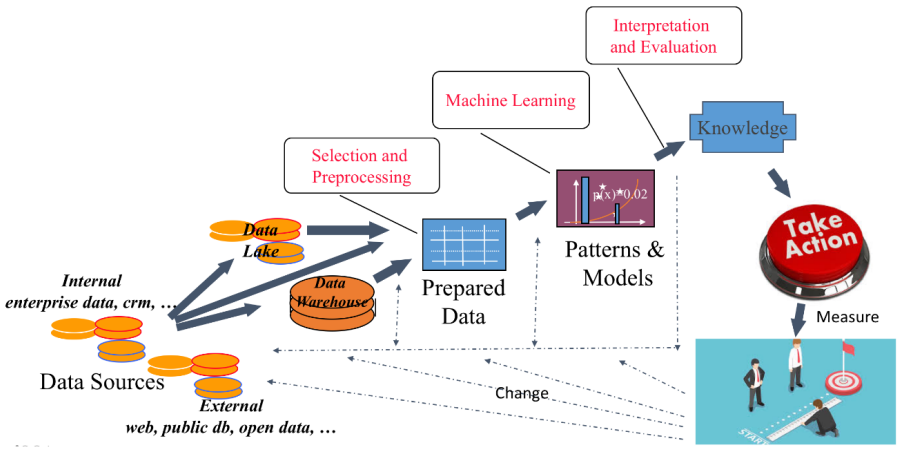


Figure 1.1: Data mining process

Machine learning Learning models and algorithms that allow to extract patterns from data. Machine learning

2 Data warehouse

Business Intelligence Transform raw data into information. Deliver the right information to the right people at the right time through the right channel.

Business Intelligence

Data Warehouse (DWH) Optimized repository that stores information for decision making processes. DWHs are a specific type of DSS.

Data Warehouse

Features:

- Subject-oriented: focused on enterprise specific concepts.
- Integrates data from different sources and provides an unified view.
- Non-volatile storage with change tracking.

Data Mart (DM) Subset of the primary DWH with information relevant to a specific business area.

Data Mart

2.1 Online Analysical Processing (OLAP)

OLAP analyses Able to interactively navigate the information in a data warehouse. Allows to visualize different levels of aggregation.

Online Analysical Processing (OLAP)

OLAP session Navigation path created by the operations that a user applied.

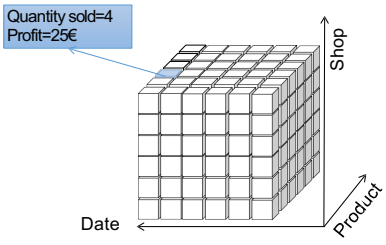
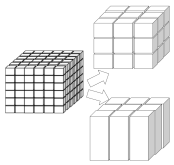


Figure 2.1: OLAP data cube

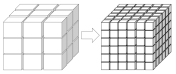
2.1.1 Operators

Roll-up Increases the level of aggregation (i.e. `GROUP BY` in SQL). Some details are collapsed together.



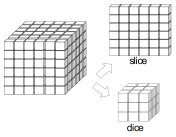
Roll-up

Drill-down Reduces the level of aggregation. Some details are reintroduced.



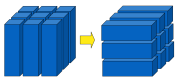
Drill-down

Slide-and-dice The slice operator reduces the number of dimensions (i.e. drops columns).
The dice operator reduces the number of data being analyzed (i.e. `LIMIT` in SQL).



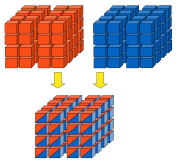
Slide-and-dice

Pivot Changes the layout of the data, to analyze it from a different viewpoint.



Pivot

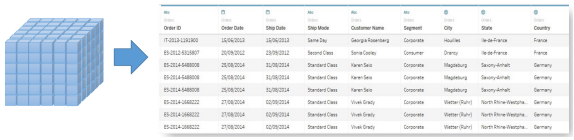
Drill-across Links concepts from different data sources (i.e. `JOIN` in SQL).



Drill-across

Drill-through Switches from multidimensional aggregated data to operational data (e.g. a spreadsheet).

Drill-through



2.2 Extraction, Transformation, Loading (ETL)

The ETL process extracts, integrates and cleans operational data that will be loaded into a data warehouse.

Extraction, Transformation, Loading (ETL)

2.2.1 Extraction

Extracted operational data can be:

Structured with a predefined data model (e.g. relational DB, CSV)

Strucured data

Untructured without a predefined data model (e.g. social media content)

Unstrucured data

Extraction can be of two types:

Static The entirety of the operational data are extracted to populate the data warehouse for the first time.

Static extraction

Incremental Only changes applied since the last extraction are considered. Can be based on a timestamp or a trigger.

Incremental extraction

2.2.2 Cleaning

Operational data may contain:

Duplicate data

Missing data

Improper use of fields (e.g. saving the phone number in the `notes` field)

Wrong values (e.g. 30th of February)

Inconsistency (e.g. use of different abbreviations)

Typos

Methods to clean and increase the quality of the data are:

Dictionary-based techniques Lookup tables to substitute abbreviations, synonyms or typos. Applicable if the domain is known and limited. — Dictionary-based cleaning

Approximate merging Merging data that do not have a common key. — Approximate merging

- **Approximate join** Use non-key attributes to join two tables (e.g. using the name and surname instead of an unique identifier).
- **Similarity approach** Use similarity functions (e.g. edit distance) to merge multiple instances of the same information (e.g. typo in customer surname).

Ad-hoc algorithms — Ad-hoc algorithms

2.2.3 Transformation

Data are transformed to respect the format of the data warehouse:

Conversion Modifications of types and formats (e.g. date format) — Conversion

Enrichment Creating new information by using existing attributes (e.g. compute profit from receipts and expenses) — Enrichment

Separation and concatenation Denormalization of the data: introduces redundances (i.e. breaks normal form[1]) to speed up operations. — Separation and concatenation

2.2.4 Loading

Adding data into a data warehouse:

Refresh The entire DWH is rewritten. — Refresh loading

Update Only the changes are added to the DWH. Old data are not modified. — Update loading

2.3 Data warehouse architectures

The architecture of a data warehouse should meet the following requirements:

Separation Separate the analytical and transactional workflows.

Scalability Hardware and software should be easily upgradable.

Extensibility Capability to host new applications and technologies without the need to redesign the system.

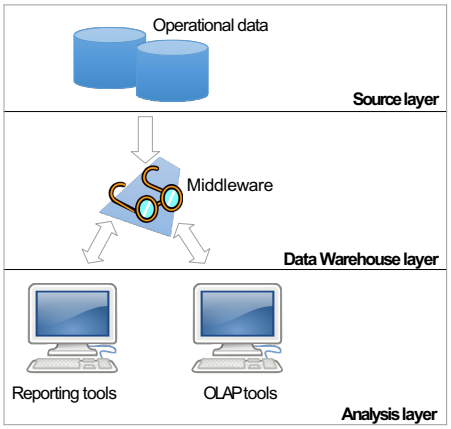
Security Access control.

Administrability Easily manageable.

[1] `https://en.wikipedia.org/wiki/Database_normalization`

2.3.1 Single-layer architecture

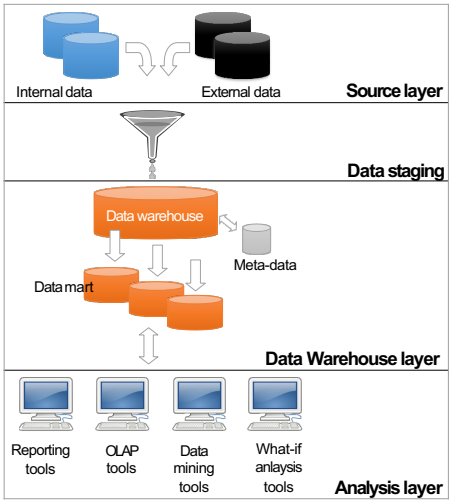
- Minimizes the amount of data stored (i.e. no redundances).
- The source layer is the only physical layer (i.e. no separation).
- A middleware provides the DWH features.



Single-layer architecture

2.3.2 Two-layer architecture

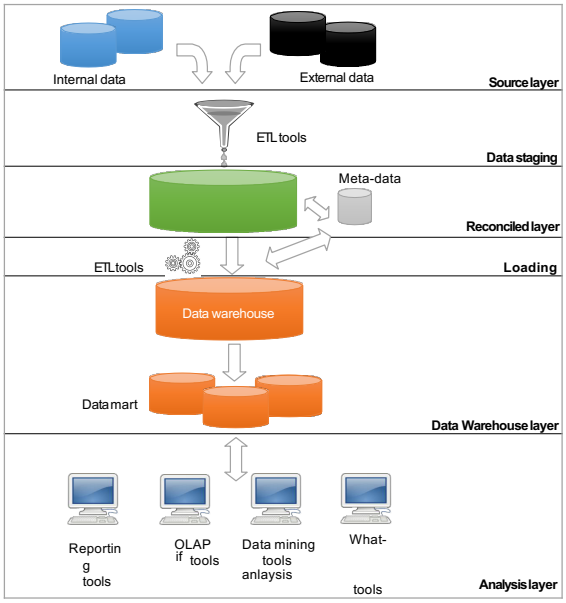
- Source data (source layer) are physically separated from the DWH (data warehouse layer).
- A staging layer applies ETL procedures before populating the DWH.
- The DWH is a centralized repository from which data marts can be created. Metadata repositories store information on sources, staging and data marts schematics.



Two-layer architecture

2.3.3 Three-layer architecture

- A reconciled layer enhances the cleaned data coming from the staging step by adding enterprise-level details (i.e. adds more redundancy before populating the DWH).



Three-layer architecture

2.4 Conceptual modeling

Dimensional Fact Model (DFM)

Dimensional Fact Model (DFM) Conceptual model to support the design of data marts. The main concepts are:

Fact Concept relevant to decision-making processes (e.g. sales).

Measure Numerical property to describe a fact (e.g. profit).

Dimension Property of a fact with a finite domain (e.g. date).

Dimensional attribute Property of a dimension (e.g. month).

Hierarchy A tree where the root is a dimension and nodes are dimensional attributes (e.g. date → month).

Primary event Occurrence of a fact. It is described by a tuple with a value for each dimension and each measure.

Secondary event Aggregation of primary events. Measures of primary events are aggregated if they have the same (preselected) dimensional attributes.

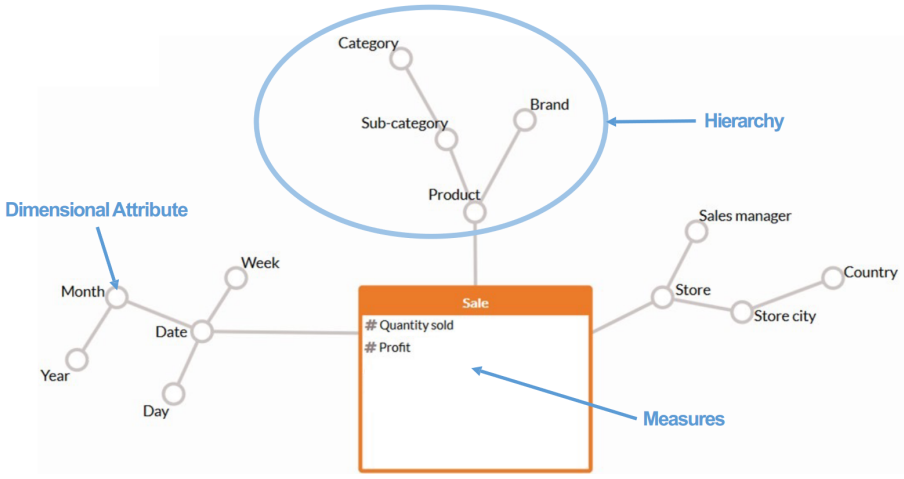


Figure 2.2: Example of DFM

Primary events

Date	Store	Product	Qty sold	Profit
01/03/15	Central store	Milk	20	60
01/03/15	Central store	Coke	25	50
02/03/15	Central store	Bread	40	70
10/03/15	Central store	Wine	15	150

SUM SUM

Secondary event

Month	Store	Category	Qty sold	Profit
March 2015	Central store	Food and Beverages	100	330

Figure 2.3: Example of primary and secondary events

2.4.1 Aggregation operators

Measures can be classified as:

Flow measures Evaluated cumulatively with respect to a time interval (e.g. quantity sold). — Flow measures

Level measures Evaluated at a particular time (e.g. number of products in inventory). — Level measures

Unit measures Evaluated at a particular time but expressed in relative terms (e.g. unit price). — Unit measures

Aggregation operators can be classified as:

Distributive Able to calculate aggregates from partial aggregates (e.g. `SUM`, `MIN`, `MAX`). — Distributive operators

Algebraic Requires a finite number of support measures to compute the result (e.g. `AVG`). — Algebraic operators

Holistic Requires an infinite number of support measures to compute the result (e.g. `RANK`). — Holistic operators

Additivity A measure is additive along a dimension if an aggregation operator can be applied. — Additive measure

	Temporal hierarchies	Non-temporal hierarchies
Flow measures	`SUM`, `AVG`, `MIN`, `MAX`	`SUM`, `AVG`, `MIN`, `MAX`
Level measures	`AVG`, `MIN`, `MAX`	`SUM`, `AVG`, `MIN`, `MAX`
Unit measures	`AVG`, `MIN`, `MAX`	`AVG`, `MIN`, `MAX`

Table 2.1: Allowed operators for each measure type

2.4.2 Logical design

Defining the data structures (e.g. tables and relationships) according to a conceptual model. There are mainly two strategies: — Logical design

Star schema A fact table that contains all the measures and linked to dimensional tables. — Star schema

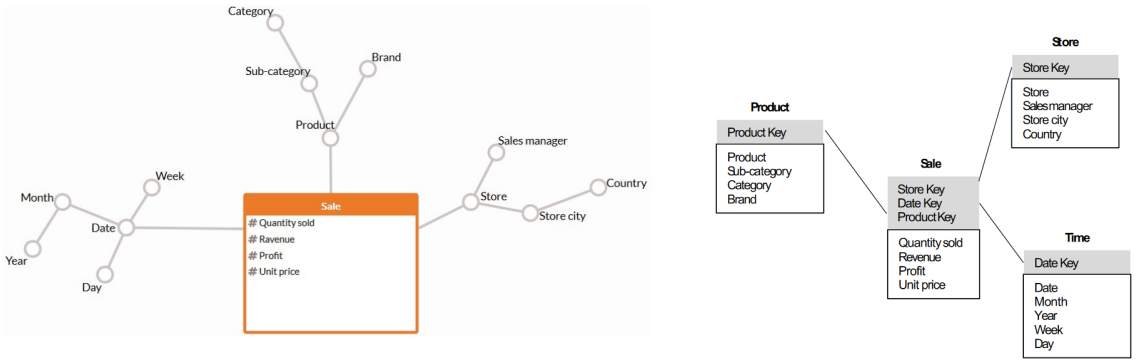


Figure 2.4: Example of star schema

Snowflake schema A star schema variant with partially normalized dimension tables. — Snowflake schema

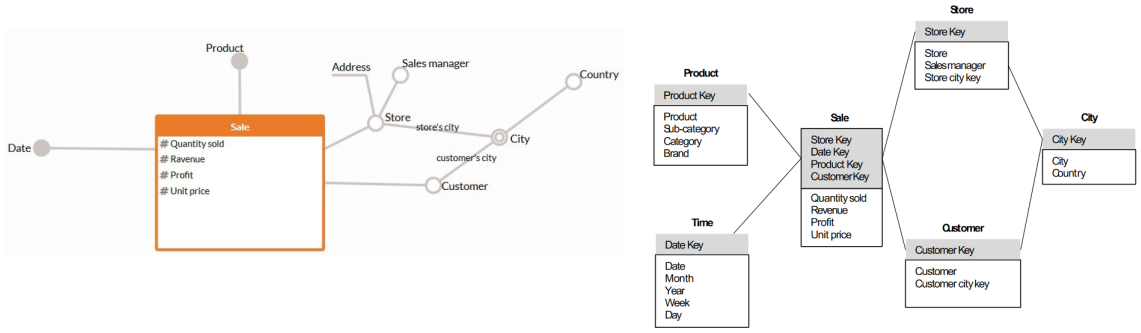


Figure 2.5: Example of snowflake schema

3 Data lake

Dark data Acquired and stored data that are never used for decision-making processes. Dark data

Data lake Repository to store raw (unstructured) data. It has the following features: Data lake

- Does not enforce a schema on write.
- Allows flexible access and applies schemas on read.
- Single source of truth.
- Low cost and scalable.

Storage Stored data can be classified as:

Hot A low volume of highly requested data that require low latency. More expensive HW/SW. Hot storage

Cold A large amount of data that does not have latency requirements. Less expensive. Cold storage

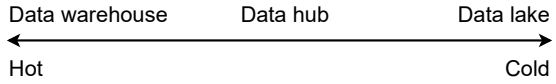


Figure 3.1: Data storage technologies

3.1 Traditional vs insight-driven data systems

	Traditional (data warehouse)	Insight-driven (data lake)
Sources	Structured data	Structured, semi-structured and unstructured data
Storage	Limited ingestion and storage capability	Virtually unlimited ingestion and storage capability
Schema	Schema designed upfront	Schema not fixed
Transformations	ETL upfront	Transformations on query
Analytics	SQL, BI tools, full-text search	Traditional methods, self-service BI, big data, machine learning, ...
Price Performance	High storage cost Fast queries	Low storage cost Scalability/speed/cost tradeoffs
Quality	High data quality	Depends on the use case

3.2 Data architecture evolution

Traditional data warehouse (i.e. in-house data warehouse) Traditional data warehouse

- Structured data with predefined schemas.
- High setup and maintenance cost. Not scalable.

- Relational high-quality data.
- Slow data ingestion.

Modern cloud data warehouse

Modern cloud data warehouse

- Structured and semi-structured data.
- Low setup and maintenance cost. Scalable and easier disaster recovery.
- Relational high-quality data and mixed data.
- Fast data ingestion if supported.

On-premise big data (i.e. in-house data lake)

On-premise big data

- Any type of data with schemas on read.
- High setup and maintenance cost.
- Fast data ingestion.

Cloud data lake

Cloud data lake

- Any type of data with schemas on read.
- Low setup and maintenance cost. Scalable and easier disaster recovery.
- Fast data ingestion.

3.3 Components

3.3.1 Data ingestion

Data ingestion

Workload migration Inserting all the data from an existing source.

Incremental ingestion Inserting changes since the last ingestion.

Streaming ingestion Continuously inserting data.

Change Data Capture (CDC) Mechanism to detect changes and insert the new data into the data lake (possibly in real-time).

Change Data Capture (CDC)

3.3.2 Storage

Raw Immutable data useful for disaster recovery.

Raw storage

Optimized Optimized raw data for faster query.

Optimized storage

Analytics Ready to use data.

Analytics storage

Columnar storage

- Homogenous data are stores contiguously.
- Speeds up methods that process entire columns (i.e. all the values of a feature).
- Insertion becomes slower.

Data catalog Methods to add descriptive metadata to a data lake. This is useful to prevent an unorganized data lake (data swamp).

3.3.3 Processing and analytics

Processing and analytics

Interactive analytics Interactive queries to large volumes of data. The results are stored back in the data lake.

Big data analytics Data aggregations and transformations.

Real-time analytics Streaming analysis.

3.4 Architectures

3.4.1 Lambda lake

Lambda lake

Batch layer Receives and stores the data. Prepares the batch views for the serving layer.

Serving layer Indexes batch views for faster queries.

Speed layer Receives the data and prepares real-time views. The views are also stored in the serving layer.

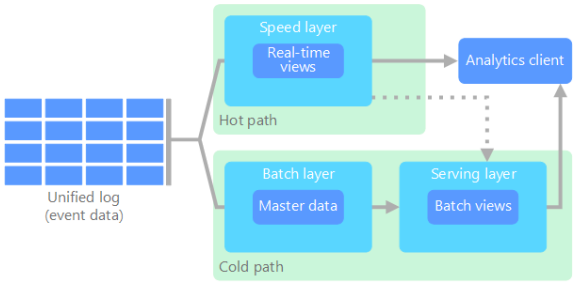


Figure 3.2: Lambda lake architecture

3.4.2 Kappa lake

The data are stored in a long-term store. Computations only happen in the speed layer (avoids lambda lake redundancy between batch layer and speed layer).

Kappa lake

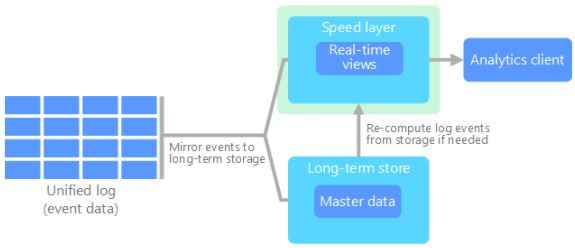


Figure 3.3: Kappa lake architecture

3.4.3 Delta lake

Framework that adds features on top of an existing data lake. Delta lake

- ACID transactions
- Scalable metadata handling
- Data versioning
- Unified batch and streaming
- Schema enforcement

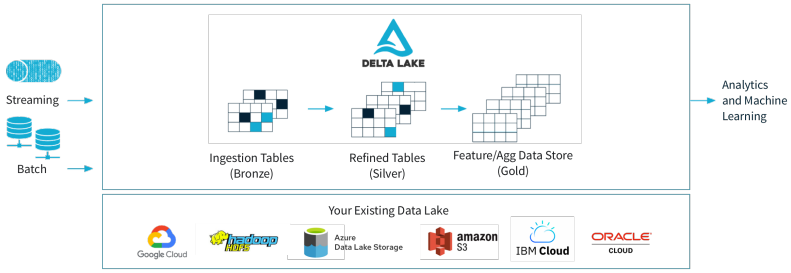


Figure 3.4: Delta lake architecture

3.5 Metadata

Metadata are used to organize a data lake. Useful metadata are: Metadata

Source Origin of the data.

Schema Structure of the data.

Format File format or encoding.

Quality metrics (e.g. percentage of missing values).

Lifecycle Retention policies and archiving rules.

Ownership

Lineage History of applied transformations or dependencies.

Access control

Classification Sensitivity level of the data.

Usage information Record of who accessed the data and how it is used.

4 CRISP-DM

Cross Industry Standard Process for Data Mining Standardized process for data mining. CRISP-DM

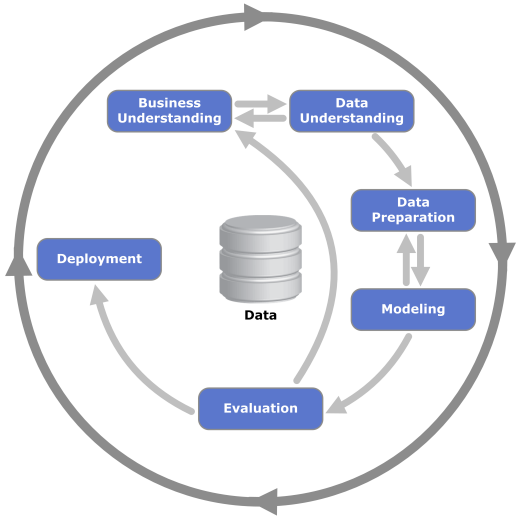


Figure 4.1: CRISP-DM workflow

4.1 Business understanding

- Determine the objective and the success criteria. Business understanding
- Feasibility study.
- Produce a plan.

4.2 Data understanding

- Determine the available (raw) data. Data understanding
- Determine the cost of the data.
- Collect, describe, explore and verify data.

4.3 Data preparation

- Data cleaning. Data preparation
- Data transformations.

4.4 Modelling

Modelling

- Select modelling technique.
- Build/train the model.

4.5 Evaluation

Evaluation

- Evaluate results.
- Review process.

4.6 Deployment

Deployment

- Plan deployment.
- Plan monitoring and maintenance.
- Final report and review.

5 Machine learning

Machine learning Application of methods and algorithms to extract patterns from data. Machine learning

5.1 Tasks

Classification Estimation of a finite number of classes.

Regression Estimation of a numeric value.

Similarity matching Identify similar individuals.

Clustering Grouping individuals based on their similarities.

Co-occurrence groupping Identify associations between entities based on the transactions in which they appear together.

Profiling Behavior description.

Link analysis Analysis of connections (e.g. in a graph).

Data reduction Reduce the dimensionality of data with minimal information loss.

Casual modeling Understand the connections between events and actions.

5.2 Categories

Supervised learning Problem where the target(s) is defined. Supervised learning

Unsupervised learning Problem where no specific target is known. Unsupervised learning

Reinforcement learning Learn a policy to generate a sequence of actions. Reinforcement learning

5.3 Data

Dataset Set of N individuals, each described by D features. Dataset

5.3.1 Data types

Categorical Values with a discrete domain.

- **Nominal** The values are a set of non-ordered labels. Categorical nominal data
  - **Operators.** $=, \neq$
  - **Example.** Name, surname, zip code.
- **Ordinal** The values are a set of totally ordered labels. Categorical ordinal data
  - **Operators.** $=, \neq, <, >, \leq, \geq$
  - **Example.** Non-numerical quality evaluations (excellent, good, fair, poor, bad).

Numerical Values with a continuous domain.

Interval Numerical values without an univocal definition of 0 (i.e. 0 is not used as reference). It is not reasonable to compare the magnitude of this type of data. Numerical interval data

Operators. $=, \neq, <, >, \leq, \geq, +, -$

Example. Celsius and Fahrenheit temperature scales, CGPA, time.

For instance, there is a 6.25% increase from 16°C to 17°C, but converted in Fahrenheit, the increase is of 2.96% (from 60.8°F to 62.6°F).

Ratio Values with an absolute 0 point. Numerical ratio data

Operators. $=, \neq, <, >, \leq, \geq, +, -$

Example. Kelvin temperature scale, age, income, length.

For instance, there is a 10% increase from 100$ to 110$. Converted in euro (1€= 1.06$), the increase is still of 10% (from 94.34€ to 103.77€).

5.3.2 Transformations

Data type		Transformation
Categorical	Nominal	One-to-one transformations
	Ordinal	Order preserving transformations (i.e. monotonic functions)
Numerical	Interval	Linear transformations
	Ratio	Any mathematical function, standardization, variation in percentage

5.3.3 Dataset format

Relational table The attributes of each record are the same. Relational table

Data matrix Matrix with N rows (entries) and D columns (attributes). Data matrix

Sparse matrix Data matrix with lots of zeros. Sparse matrix

Example (Bag-of-words)**.** Each row represents a document, each column represents a term. The i, j-th cell contains the frequency of the j-th term in the i-th document.

Transactional data Each record contains a set of objects (not necessarily a relational table). Transactional data

Graph data Set of nodes and edges. Graph data

Ordered data e.g. temporal data. Ordered data

5.3.4 Data quality

Noise Alteration of the original values. Noise

Outliers Data that considerably differ from the majority of the dataset. May be caused by noise or rare events. Outliers

Box plots can be used to visually detect outliers.

Missing values Data that have not been collected. Sometimes they are not easily recognizable (e.g. when special values are used, instead of `null`, to mark missing data). Missing values

Can be handled in different ways:

- Ignore the records with missing values.
- Estimate or default missing values.
- Ignore the fact that some values are missing (not always applicable).
- Insert all the possible values and weight them by their probability.

Duplicated data Data that may be merged. Duplicated data

6 Classification

Classification

(Supervised) classification Given a finite set of classes C and a dataset $\boldsymbol{X}$ of N individuals, each associated to a class $y(\mathbf{x}) \in C$, we want to learn a model $\mathcal{M}$ able to guess the value of $y(\bar{\mathbf{x}})$ for unseen individuals.

Classification can be:

Crisp classification

Crisp Each individual has one and only one label.

Probabilistic classification

Probabilistic Each individual is assigned to a label with a certain probability.

Classification model

Classification model A classification model (classifier) makes a prediction by taking as input a data element $\mathbf{x}$ and a decision function $y_{\boldsymbol{\theta}}$ parametrized on $\boldsymbol{\theta}$:

$$\mathcal{M}(\mathbf{x}, \boldsymbol{\theta}) = y_{\boldsymbol{\theta}}(\mathbf{x})$$

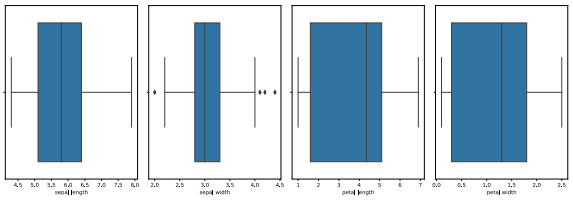
Vapnik-Chervonenkis dimension

Vapnik-Chervonenkis dimension A dataset with N elements defines 2^N learning problems. A model $\mathcal{M}$ has Vapnik-Chervonenkis (VC) dimension N if it is able to solve all the possible learning problems with N elements.

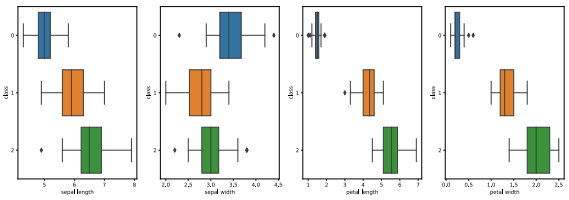
Example. A straight line has VC dimension 3.

Data exploration

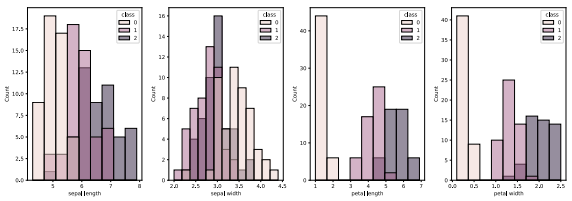
Data exploration



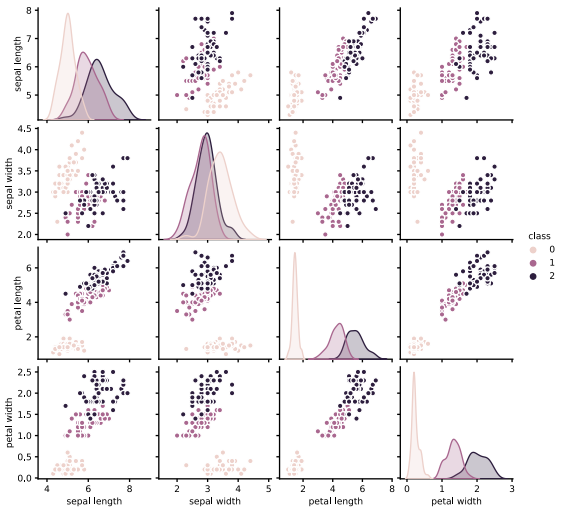
(a) Iris dataset general boxplot



(b) Iris dataset class boxplot



(c) Iris dataset histograms



(d) Iris dataset pairplots

Dataset split A supervised dataset can be randomly split into:

Train set

Train set Used to learn the model. Usually the largest split.

Test set

Test set Used to evaluate the trained model.

Validation set

Validation set Used to evaluate the model during training.

It is assumed that the splits have similar characteristics.

Overfitting

Overfitting Given a dataset $\boldsymbol{X}$, a model $\mathcal{M}$ is overfitting if there exists another model $\mathcal{M}'$ such that:

$$\texttt{error}_{\text{train}}(\mathcal{M}) < \texttt{error}_{\text{train}}(\mathcal{M}')$$
$$\texttt{error}_{\boldsymbol{X}}(\mathcal{M}) > \texttt{error}_{\boldsymbol{X}}(\mathcal{M}')$$

Possible causes of overfitting are:

- Noisy data.
- Lack of representative instances.

6.1 Decision trees

6.1.1 Information theory

Shannon theorem

Entropy

Shannon theorem Let $\boldsymbol{X} = \{\mathbf{v}_1, \dots, \mathbf{v}_V\}$ be a data source where each of the possible value has probability $p_i = \mathcal{P}(\mathbf{v}_i)$. The best encoding allows to transmit $\boldsymbol{X}$ with an average number of bits given by the **entropy** of X:

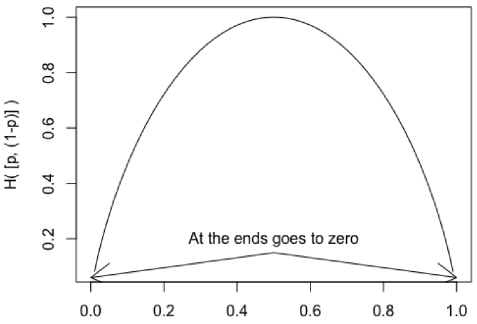
$$H(\boldsymbol{X}) = -\sum_j p_j \log_2(p_j)$$

$H(\boldsymbol{X})$ can be seen as a weighted sum of the surprise factor $-\log_2(p_j)$. If $p_j \sim 1$, then the surprise of observing $\mathbf{v}_j$ is low, vice versa, if $p_j \sim 0$, the surprise of observing $\mathbf{v}_j$ is high.

Therefore, when $H(\boldsymbol{X})$ is high, $\boldsymbol{X}$ is close to an uniform distribution. When $H(\boldsymbol{X})$ is low, $\boldsymbol{X}$ is close to a constant.

Example (Binary source)**.**

The two values of a binary source $\boldsymbol{X}$ have respectively probability p and $(1-p)$.
When $p \sim 0$ or $p \sim 1$, $H(\boldsymbol{X}) \sim 0$.
When $p \sim 0.5$, $H(\boldsymbol{X}) \sim \log_2(2) = 1$



Entropy threshold split

Entropy threshold split Given a dataset $\boldsymbol{D}$, a real-valued attribute $d \in \boldsymbol{D}$, a threshold t in the domain of d and the class attribute c of $\boldsymbol{D}$. The entropy of the class c of the dataset $\boldsymbol{D}$ split with threshold t on d is a weighted sum:

$$H(c \,|\, d : t) = \mathcal{P}(d < t)H(c \,|\, d < t) + \mathcal{P}(d \geq t)H(c \,|\, d \geq t)$$

Information gain Information gain measures the reduction in entropy after applying a split. It is computed as: Information gain

$$IG(c \mid d : t) = H(c) - H(c \mid d : t)$$

When $H(c \mid d : t)$ is low, $IG(c \mid d : t)$ is high as splitting with threshold t result in purer groups. Vice versa, when $H(c \mid d : t)$ is high, $IG(c \mid d : t)$ is low as splitting with threshold t is not very useful.

The information gain of a class c split on a feature d is given by:

$$IG(c \mid d) = \max_t IG(c \mid d : t)$$

6.1.2 Tree construction

Decision tree (C4.5) Tree-shaped classifier where leaves are class predictions and inner nodes represent conditions that guide to a leaf. This type of classifier is non-linear (i.e. does not represent a linear separation). Decision tree

Each node of the tree contains:

- The applied splitting criteria (i.e. feature and threshold). Leaves do not have this value.
- The entropy of the current split.
- Dataset coverage of the current split.
- Classes distribution.

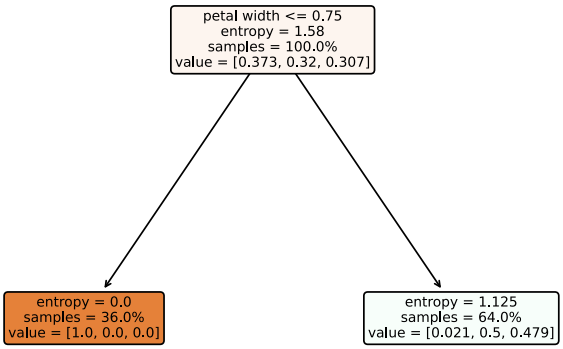


Figure 6.2: Example of decision tree

Note: the weighted sum of the entropies of the children is always smaller than the entropy of the parent.

Possible stopping conditions are:

- When most of the leaves are pure (i.e. nothing useful to split).
- When some leaves are impure but none of the possible splits have positive IG. Impure leaves are labeled with the majority class.

Purity Value to maximize when splitting a node of a decision tree. Purity

Nodes with uniformly distributed classes have a low purity. Nodes with a single class have the highest purity.

Possible impurity measures are:

Entropy/Information gain See Section 6.1.1.

Gini index Let $\boldsymbol{X}$ be a dataset with classes C. The Gini index measures how often an element of $\boldsymbol{X}$ would be misclassified if the labels were randomly assigned based on the frequencies of the classes in $\boldsymbol{X}$. Gini index

Given a class $i \in C$, p_i is the probability (i.e. frequency) of classifying an element with i and $(1 - p_i)$ is the probability of classifying it with a different label. The Gini index is given by:

$$GINI(\boldsymbol{X}) = \sum_{i}^{C} p_i(1 - p_i) = \sum_{i}^{C} p_i - \sum_{i}^{C} p_i^2$$
$$= 1 - \sum_{i}^{C} p_i^2$$

When $\boldsymbol{X}$ is uniformly distributed, $GINI(\boldsymbol{X}) \sim (1 - \frac{1}{|C|})$. When $\boldsymbol{X}$ is constant, $GINI(\boldsymbol{X}) \sim 0$.

Given a node p split in n children $p_1, \ldots, p_n$, the Gini gain of the split is given by:

$$GINI_{\text{gain}} = GINI(p) - \sum_{i=1}^{n} \frac{|p_i|}{|p|} GINI(p_i)$$

Misclassification error Skipped. Misclassification error

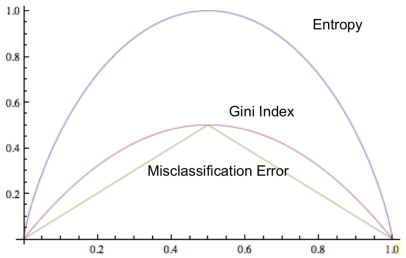


Figure 6.3: Comparison of impurity measures

Compared to Gini index, entropy is more robust to noise.

Algorithm 1 Decision tree construction using information gain as impurity measure

```
def buildTree(split):
    node = Node()
    if len(split.classes) == 1: # Pure split
        node.label = split.classes[0]
        node.isLeaf = True
    else:
        ig, attribute, threshold = getMaxInformationGain(split)
        if ig < 0:
            node.label = split.majorityClass()
            node.isLeaf = True
        else:
            node.left = buildTree(split[attribute < threshold])
            node.right = buildTree(split[attribute >= threshold])
    return node
```

Pruning Remove branches to reduce overfitting.

Pruning