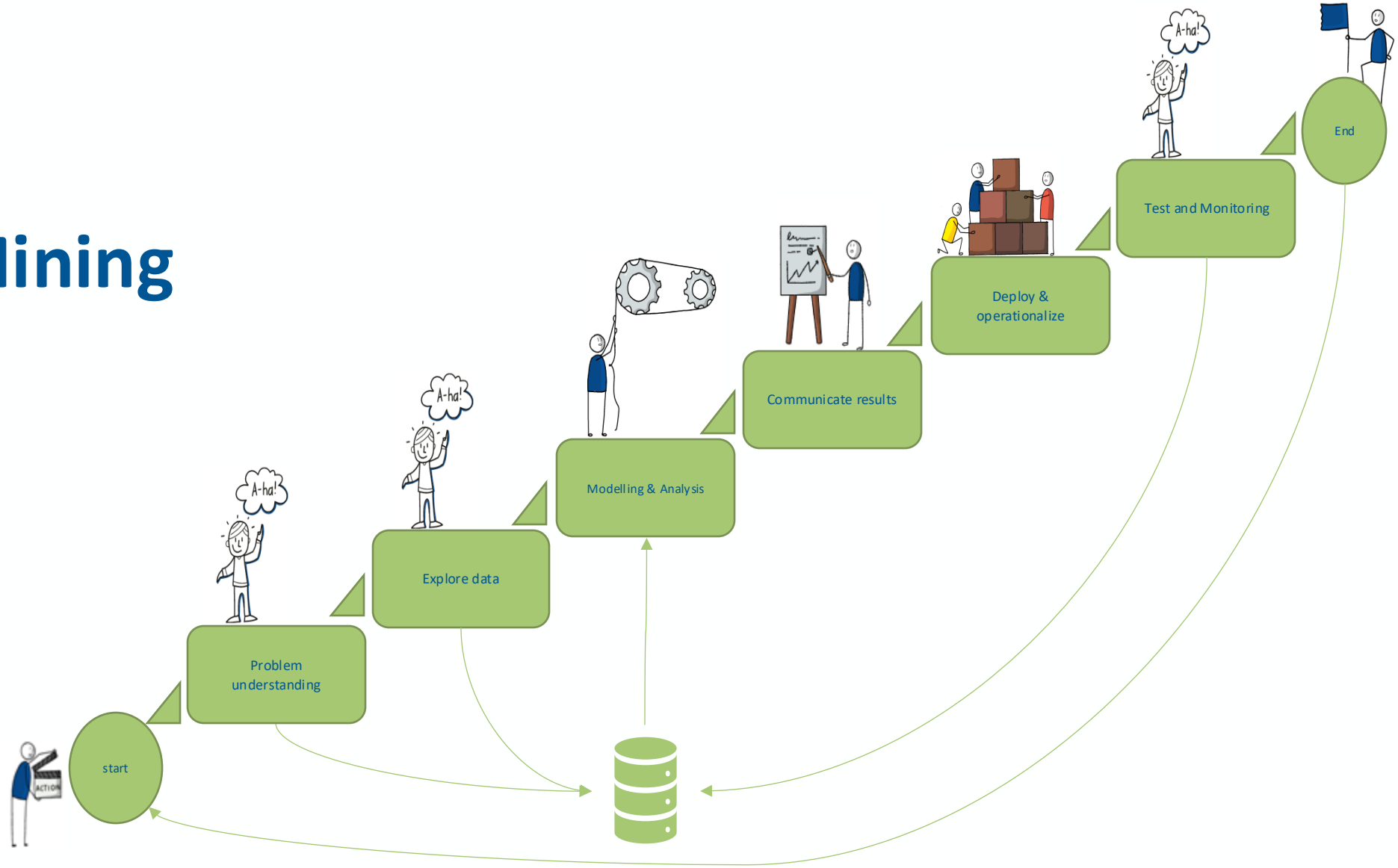
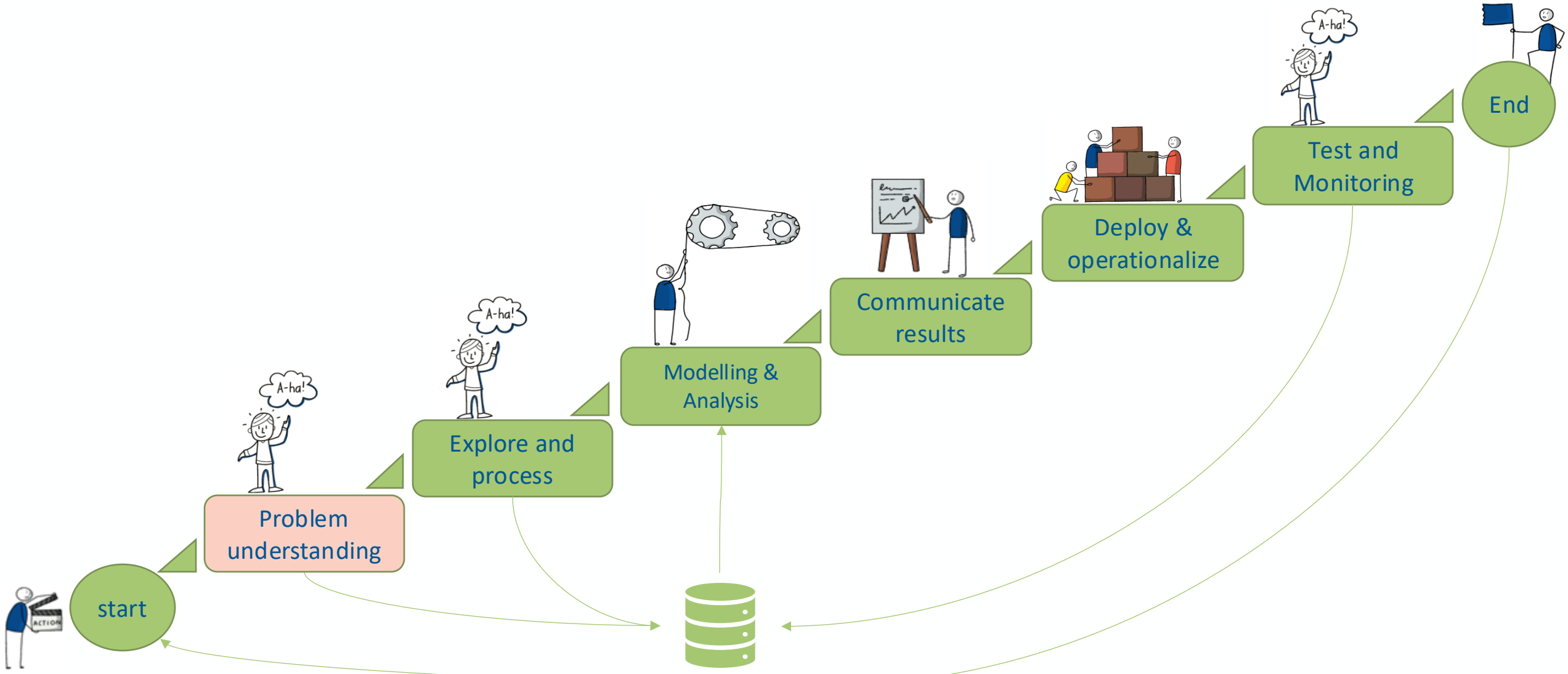
Basic Data Mining Practice-1

Prof. Dr. Matteo Marouf

15.04. 2025



Data Mining project life cycle may involve all or some of these stages



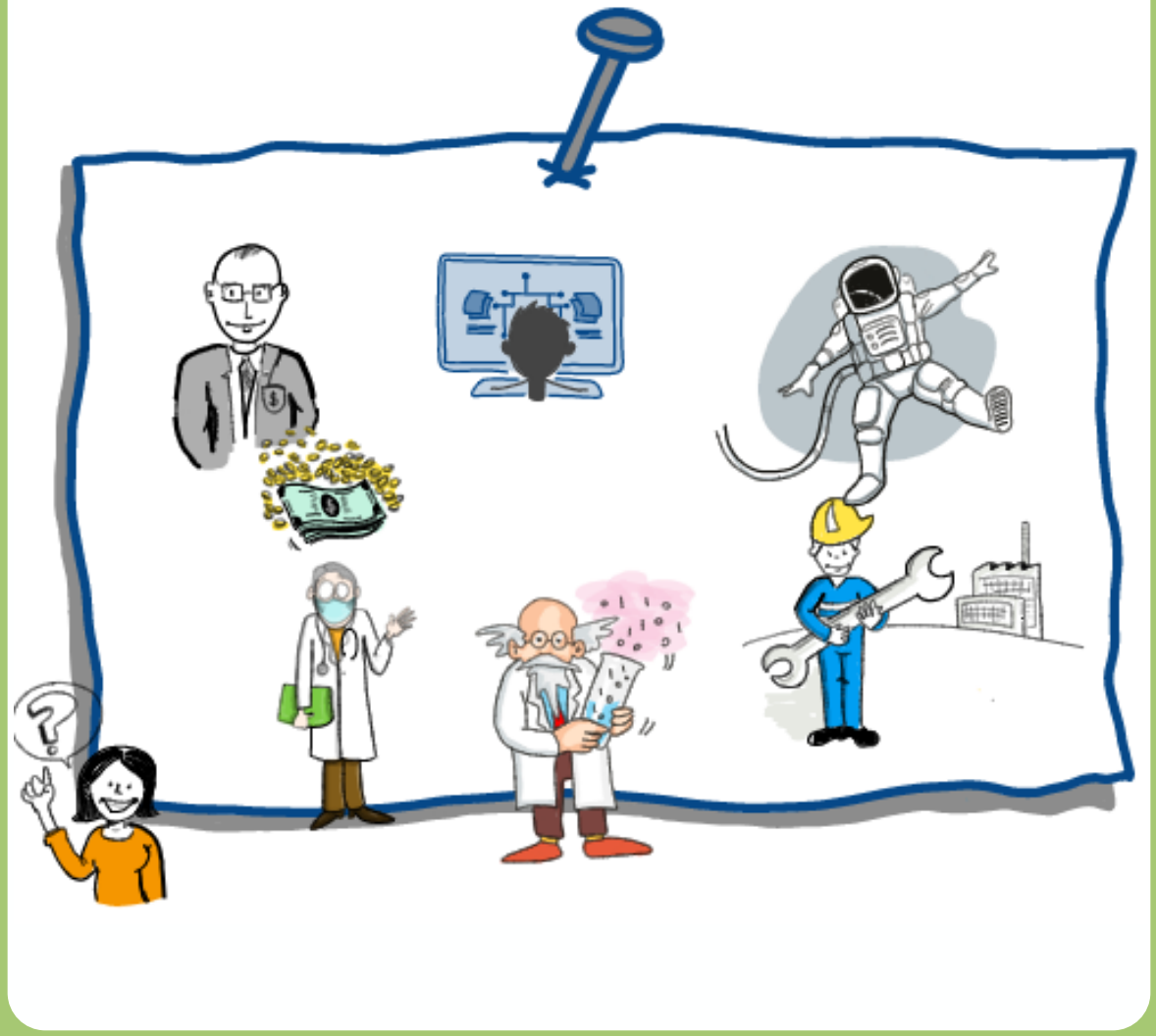
Domain knowledge plays a crucial role in successful data mining project

Domain Knowledge:

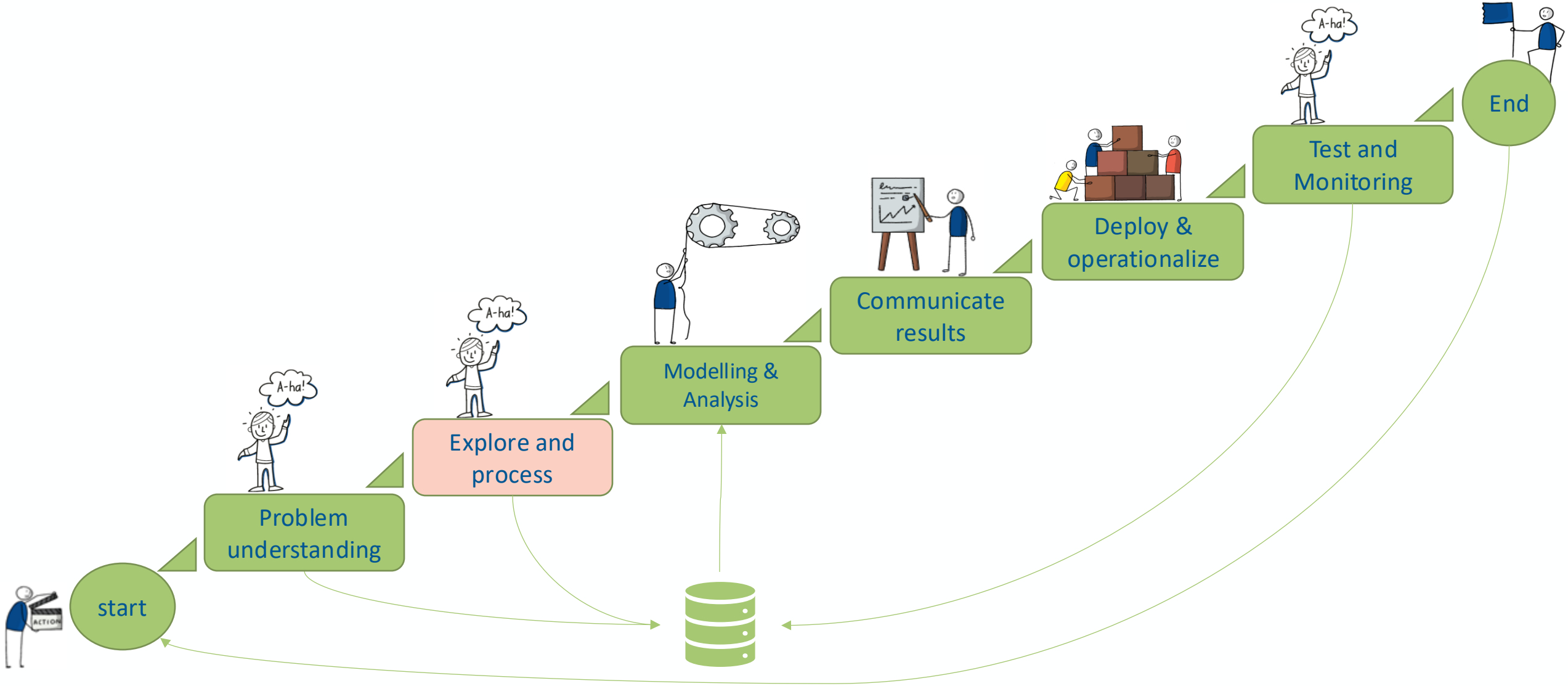
- guides the successful preprocessing
- Influnces the modeling
- Provides the the context to explain the results

The key takeaway

Always dive into the domain behind the data — it's the key to engineering meaningful features, focusing your analysis, and telling a story your audience understands.

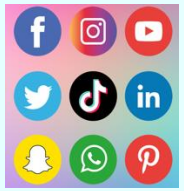


Data Mining project life cycle may involve all or some of these stages



Informative Data representation is a fundamental step in Data Mining

Data exist in different formats



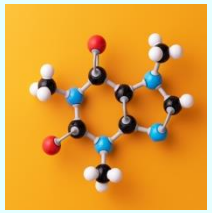
Social media

Imaging

EHR

Genomics

Knowledge graphs



Bioactive molecules

Pharmaceutical packaging machines mainly focus on large-volume products: Quantities of more than 100,000 units are standard. They are not working well for medicines that are produced and packaged only in small quantities, so-called microbatches. This is where conventional packaging machin are mostly inefficient due to the long setup and changeover times. A problem that production experts from Boehringer Ingelheim have been working on in recent years.

Textual data

Vital signals

Data Representation

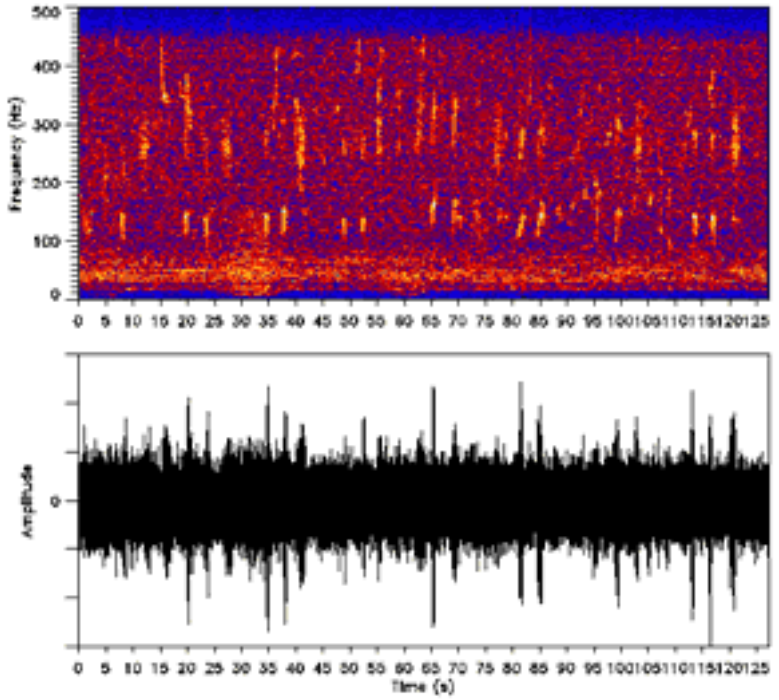
To train a model, we must create an **informative and numerical representation of the data** that can be manipulated by electronic devices.

Examples

- **Textual data:** Word Embedding is a term used for the representation of words for text analysis, typically in real valued vectors.
- **Images**: Pixel values are used.
- **Molecular structure:** Could be represented as mathematical graph where each atom is a node, and each bond is an edge.

Examples on finding a proper Data representation

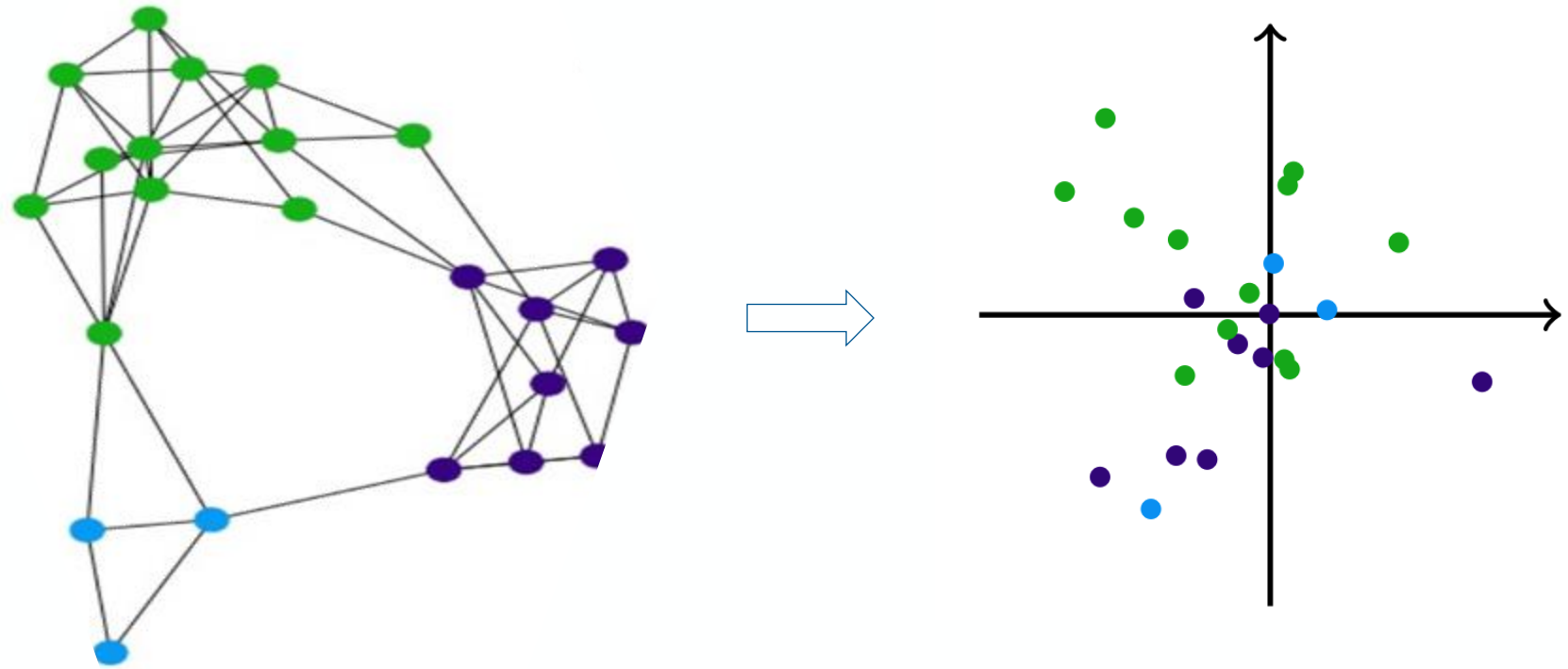
Spectrograms: An audio signal can be transformed into a spectrogram, representing the signal's frequency spectrum over time. This visual representation is useful for speech and music analysis.



https://commons.wikimedia.org/w/index.php?curid=34672291

Examples on finding a proper Data representation

- **Node Embeddings (e.g., Node2Vec) for graph data :** Learn continuous feature representations for nodes in a graph, capturing the network's structural information.



Examples on finding a proper Data representation

- **Term Frequency-Inverse Document Frequency (TF-IDF)** converts textual information into numerical vectors, reflecting the importance of words in documents.

TF

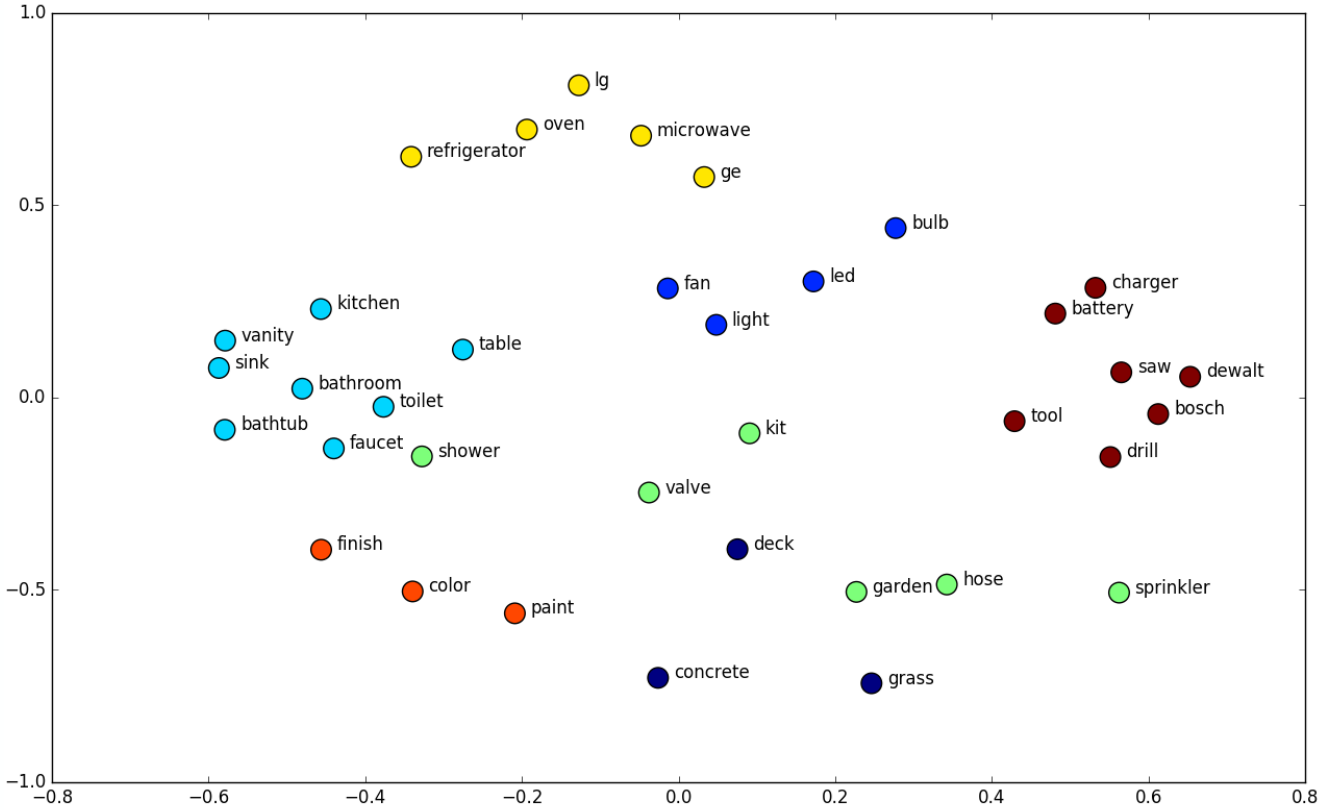
Frequency of a word withing a document

IDF

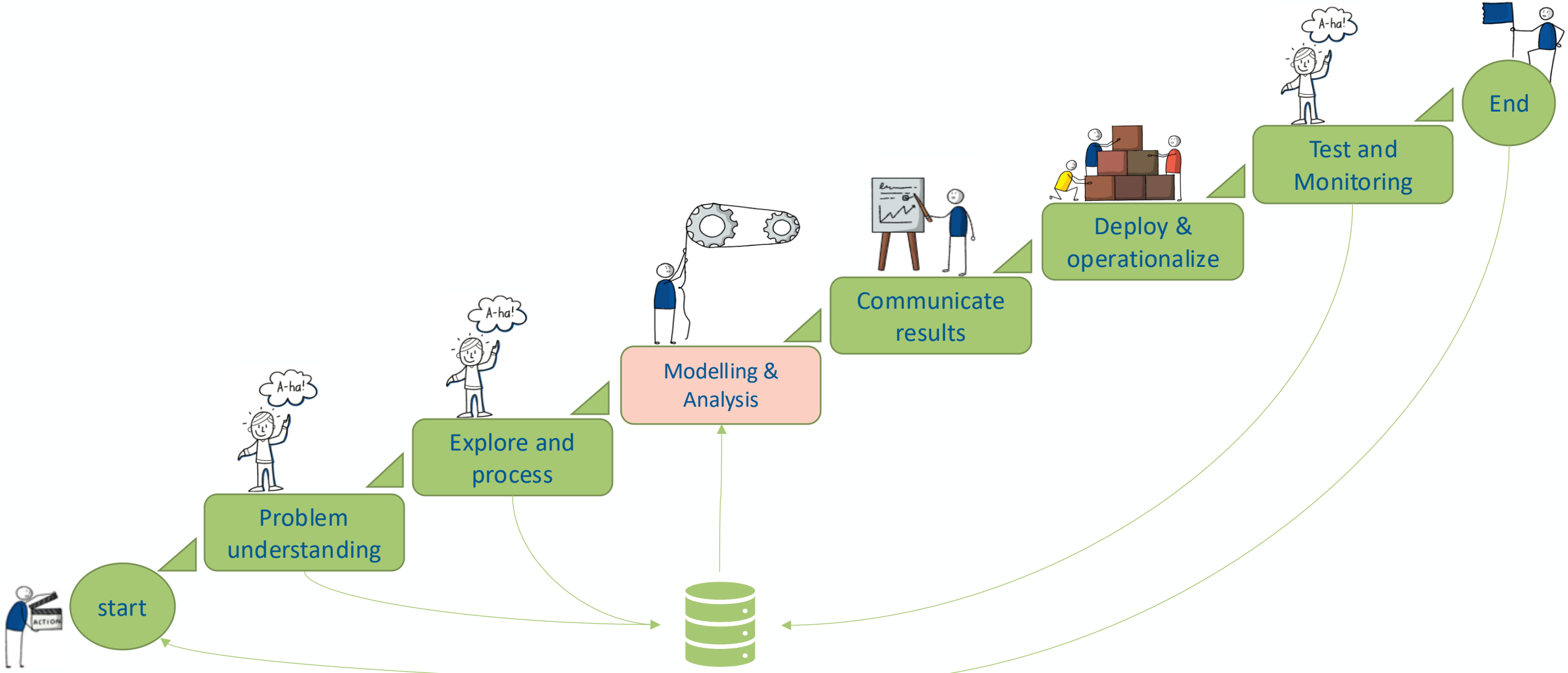
Frequency of a word across the corpus

Examples on finding a proper Data representation

- **Word Embeddings (e.g., Word2Vec, GloVe):** Word embeddings map words into continuous vector spaces where semantically similar words are positioned closely. These vectors capture contextual relationships and can be pre-trained on large corpora.



Data Science project life cycle

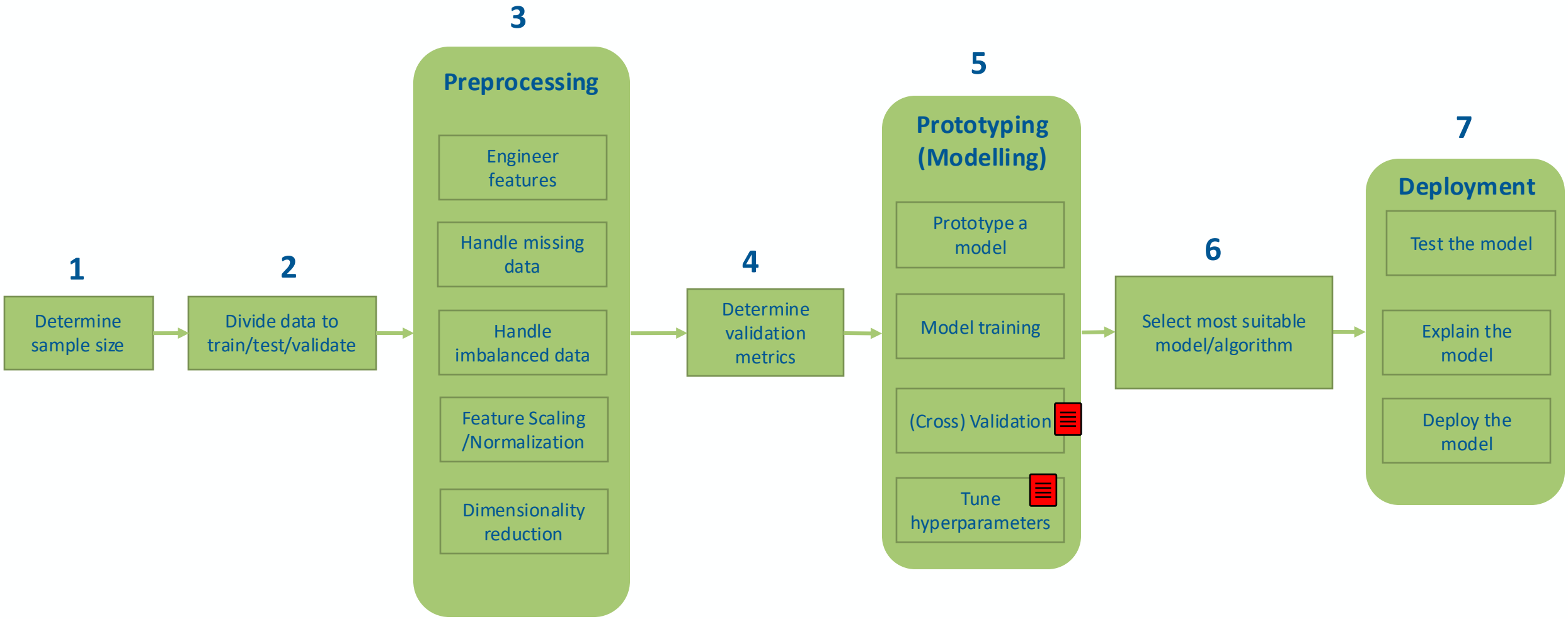


Main topics you may stumble across while working in this field

Classification:	Sorting data into predefined categories (e.g., spam detection).
Association Analysis:	identifying relationships between items in large datasets
Regression:	Predicting continuous values (e.g., stock prices, recovery time for hospitalized patients).
Clustering:	Grouping data based on similarity without predefined labels (e.g., customer segmentation).
Optimization:	AI often finds the best solutions to problems within constraints (e.g., route planning, resource allocation).
Recommendation:	Suggesting items or content based on user preferences (e.g., Netflix or Amazon recommendation engines).
Anomaly Detection:	Identifying unusual patterns or outliers in data (e.g., fraud detection, fault prediction).
Personalization:	Tailoring user experiences or interactions (e.g., personalized marketing, adaptive learning systems).
Decision Support:	Inform decision-making by analyzing large datasets and providing insights (e.g., medical diagnostics, business intelligence).
Simulation:	model complex systems or environments to predict behavior and outcomes (e.g., weather modeling, financial market).
Text mining	Understanding, mining, and generating human language (e.g., sentiment analysis, language translation).

Which steps are usually involved a Data Mining Analysis?

Example: The anatomy of classification using ML most steps are the shared with other analysis types



Sample size discussion

Statistical Power & Accuracy	A larger sample size increases the **statistical power** of models, reducing the risk of **Type II errors** (failing to detect a real effect).
	It enhances **estimation accuracy**, ensuring that patterns and relationships found in the sample reflect the true population.
Generalizability	A small sample may not adequately represent the **population**, leading to biased results.
	A sufficiently large and **diverse** sample helps models generalize well to new, unseen data.
Reducing Variability & Overfitting	Small samples often lead to **high variance**, where minor changes in data can lead to drastically different model outcomes.
	Larger samples help models learn **underlying patterns** rather than memorizing noise, reducing the risk of **overfitting**.
Choice of Algorithms	Some algorithms, like **deep learning**, require **large datasets** to perform well, while others (e.g., decision trees, k-NN) can work reasonably with smaller samples.
	Small samples may force the use of **simpler models** that might not capture complex relationships.
Handling Class Imbalance	In classification problems (e.g., fraud detection, medical diagnosis), a small sample may result in a **class imbalance**, making the model biased toward the majority class.
	A larger dataset allows better handling of rare events and minority class distributions.
Computational Cost vs. Performance Trade-Off	While larger samples improve model reliability, they also increase **computational costs** in terms of processing time and storage.
	In some cases, **sampling techniques** (e.g., stratified sampling, bootstrapping) are used to balance efficiency and accuracy.

Essential Questions to Guide Your Data Mining Analysis

Ensuring accuracy, generalizability, and computational feasibility from the start.

Determine sample size →

Do you have too much data?

- Consider sampling technique to select a representative sample.
- Consider using big data frameworks to facilitate data analysis.
- Consider using learning curves to find the sufficient sample size.

.....

Do you have too little data?

- Consider collecting more data.
- Consider using data augmentation methods.

....

Have you not collected data yet?

- Consider collecting data and evaluate whether it is enough.
- Resort to domain expert when collecting data.

....

Conclusion

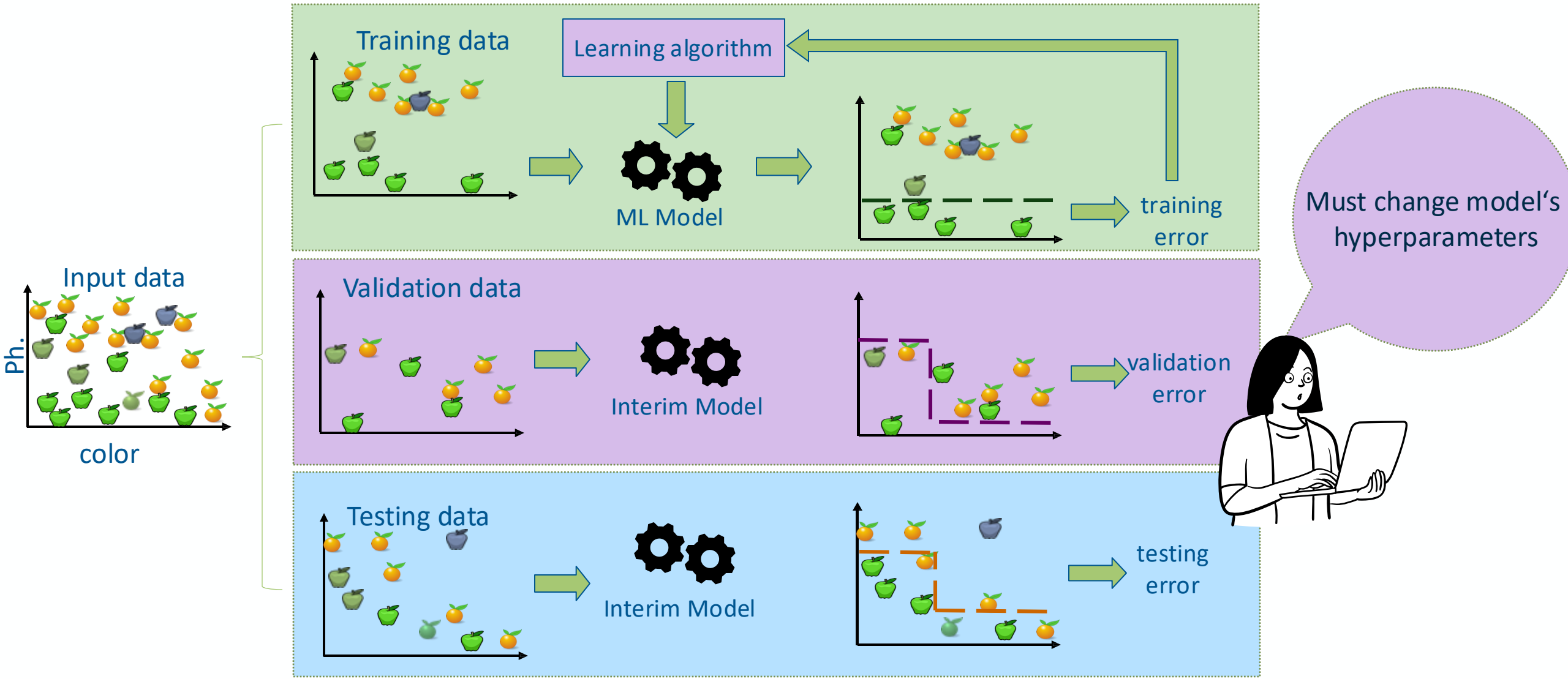
While larger sample sizes generally improve model performance, the optimal size depends on the complexity of the problem, the algorithm being used, the level of noise in the data, and the computational resources available.

Split to train, test and validate

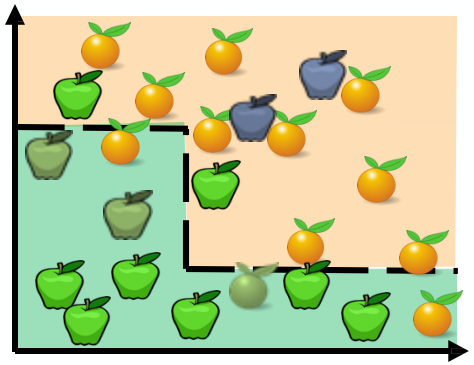
Determine sample size → Divide data set to train/test/validate →

Why?

Example on splitting to Training, Validating, Testing datasets



Anatomy of Machine Learning: inference

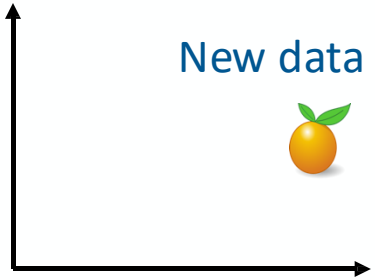


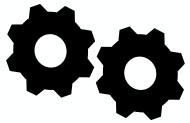
The model is now working sufficiently good for training, validating, and testing data

Final Model

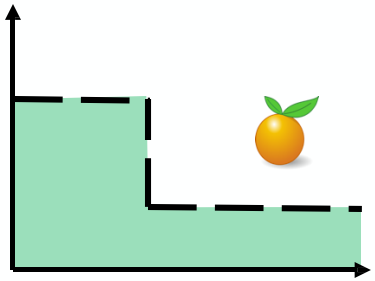
INFERENCE

New data point





Final Model



Split to train, test and validate

Determine sample size → Divide data set to train/test/validate →

Maximize generalization

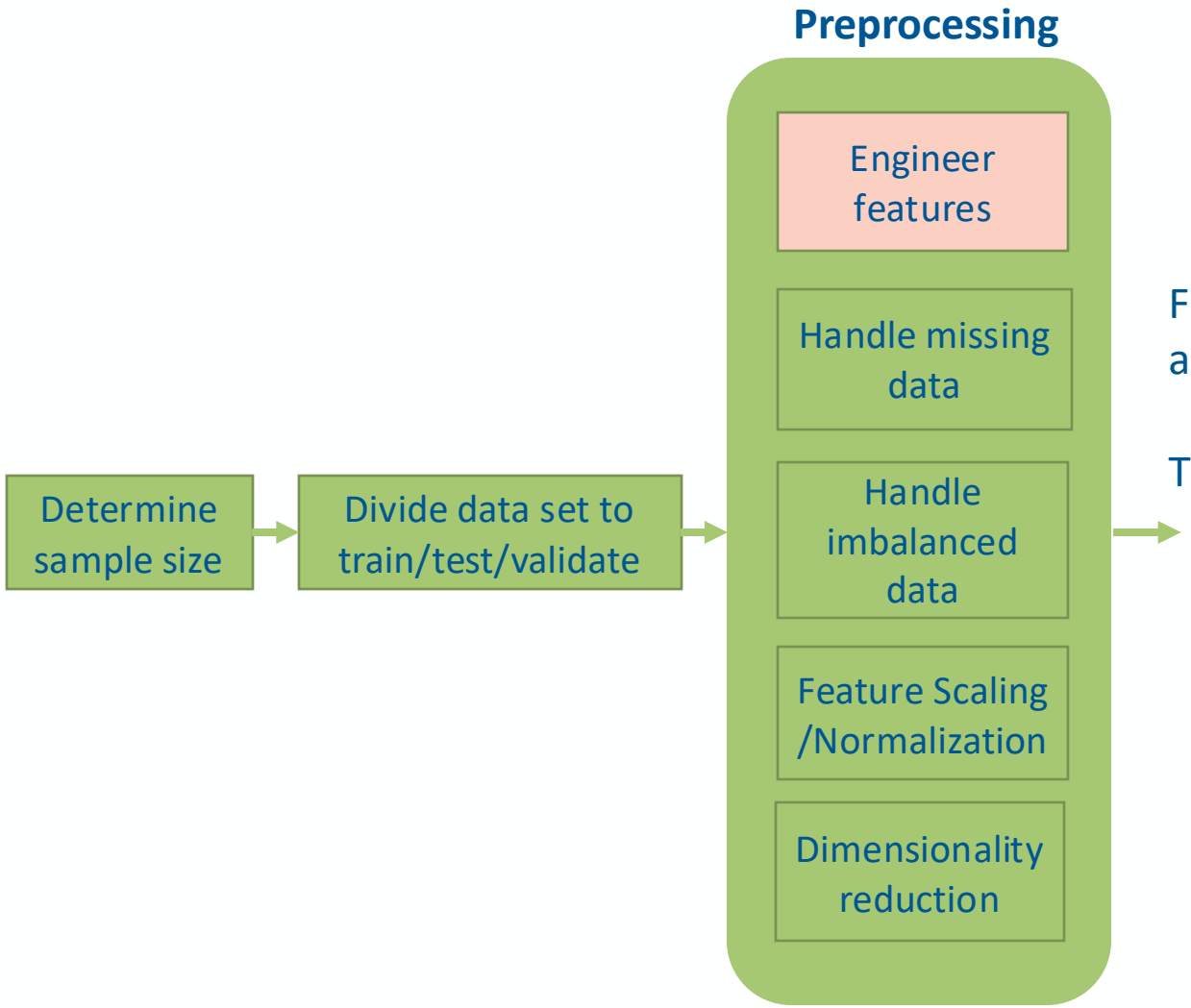
Avoid underfitting

Avoid overfitting

What we build on train data **is good for the train data** but also **good for other unseen data.**

Time for Analogy

Preprocessing: Features engineering



Feature engineering is **critical** for improving modelling accuracy and generating useful insights.

The **choice of technique depends on**:

- The **type of data** (numerical, categorical, text, image).
- The **problem** (classification, regression, clustering).
- The **model** being used (linear models, tree-based models, neural networks).

Features Transformation: Engineering new features

Feature Creation creating new features from existing ones to capture additional patterns in data, e.g. extract speed using time and distance to study rate of movement .

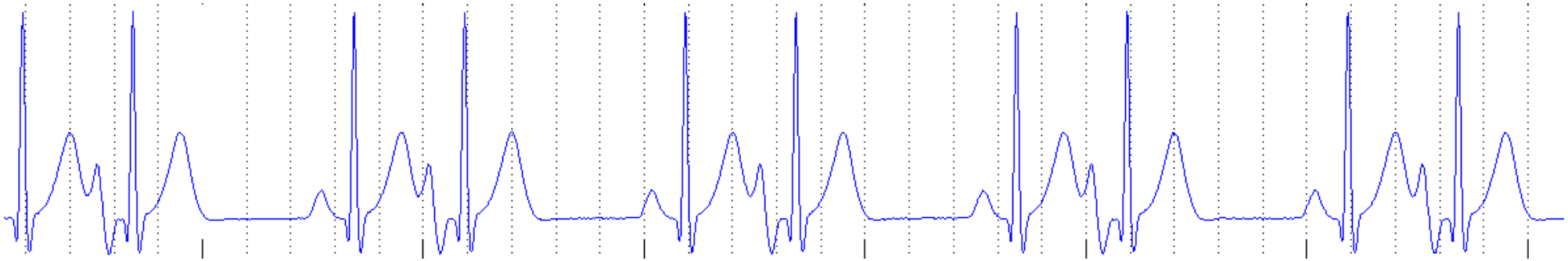
Combining– use an interaction term (e.g., x2,xyx^2, xyx2,xy) e.g. when studying income number of working hours plus income could be converted into income per hour.

Aggregations – Computing statistics (mean, sum, count) over groups.

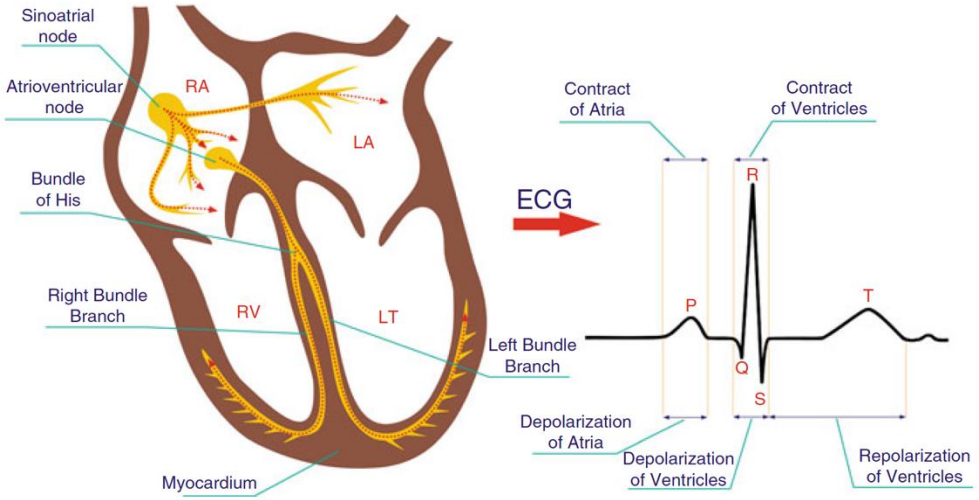
Convert Datetime Features – Extracting hour, day, month, or season from timestamps.

Extract Text Features – Extracting word counts, sentiment, or embeddings from text.

Example: Feature engineering in ECG signals



Successful feature engineering leverages domain knowledge to transform meaning features from raw data.



Example: in ECG signals, features (covariates) are derived by processing the signal to find the different peaks and valleys and compute the main intervals that reflect heart activity.

Feature Encoding

Most algorithms cannot work directly with categorical variables, so we need to convert them into a numerical format

ID	Production Year	Speed	Horsepower	Car Brand	price
0	2018	220	75	Toyota	….
1	2015	240	90	Ford	…
2	2019	190	95	Tesla	…

Example: car band in a data set we want to use to model and predict used-car price

Feature Encoding

Most algorithms cannot work directly with categorical variables, so we need to convert them into a numerical format.

1

Toyota	Ford	Tesla
1	0	0
0	1	0
0	0	1

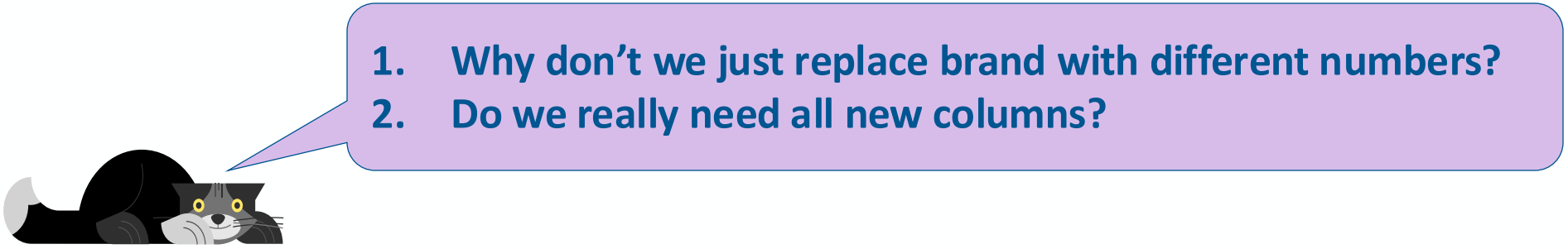
We first create a code for each category

2

ID	Production Year	Speed	Horsepower	Toyota	Ford	Tesla	price
0	2018	220	75	1	0	0	….
1	2015	240	90	0	1	0	…
2	2019	190	95	0	0	1	…

Replace the categories so that models can process the categorical information numerically.

1. **Why don't we just replace brand with different numbers?**
2. **Do we really need all new columns?**



Answers

1. **Label encoding introduces an artificial ordinal relationship.**
   - The model thinks that Blue (2) > Green (1) > Red (0)
   - It assumes **order** and **distance**, like "Green is closer to Blue than Red"

   - This can **mislead** models that use math on input values (e.g. linear regression, logistic regression, k-NN)

2. **There is a redundant column:** Keeping the original column can lead to multicollinearity, where redundant information can affect the model's performance (it can assume some weird non-existing relation).

Feature Encoding

dropping one column after encoding prior to modelling

1

Toyota	Ford	Tesla
1	0	0
0	1	0
0	0	1

We first create a code for each category

2

ID	Production Year	Speed	Horsepower	Toyota	Ford	Tesla	price
0	2018	220	75	1	0	0	….
1	2015	240	90	0	1	0	…
2	2019	190	95	0	0	1	…

Replace the categories so that models can process the categorical information numerically.

3

ID	Production Year	Speed	Horsepower	Toyota	Ford	price
0	2018	220	75	1	0	….
1	2015	240	90	0	1	…
2	2019	190	95	0	0	…

Drop the redundant column: Keeping the original column can lead to multicollinearity, where redundant information can affect the model's performance (it can assume some weird non-existing relation).

Features Binning of numerical subranges

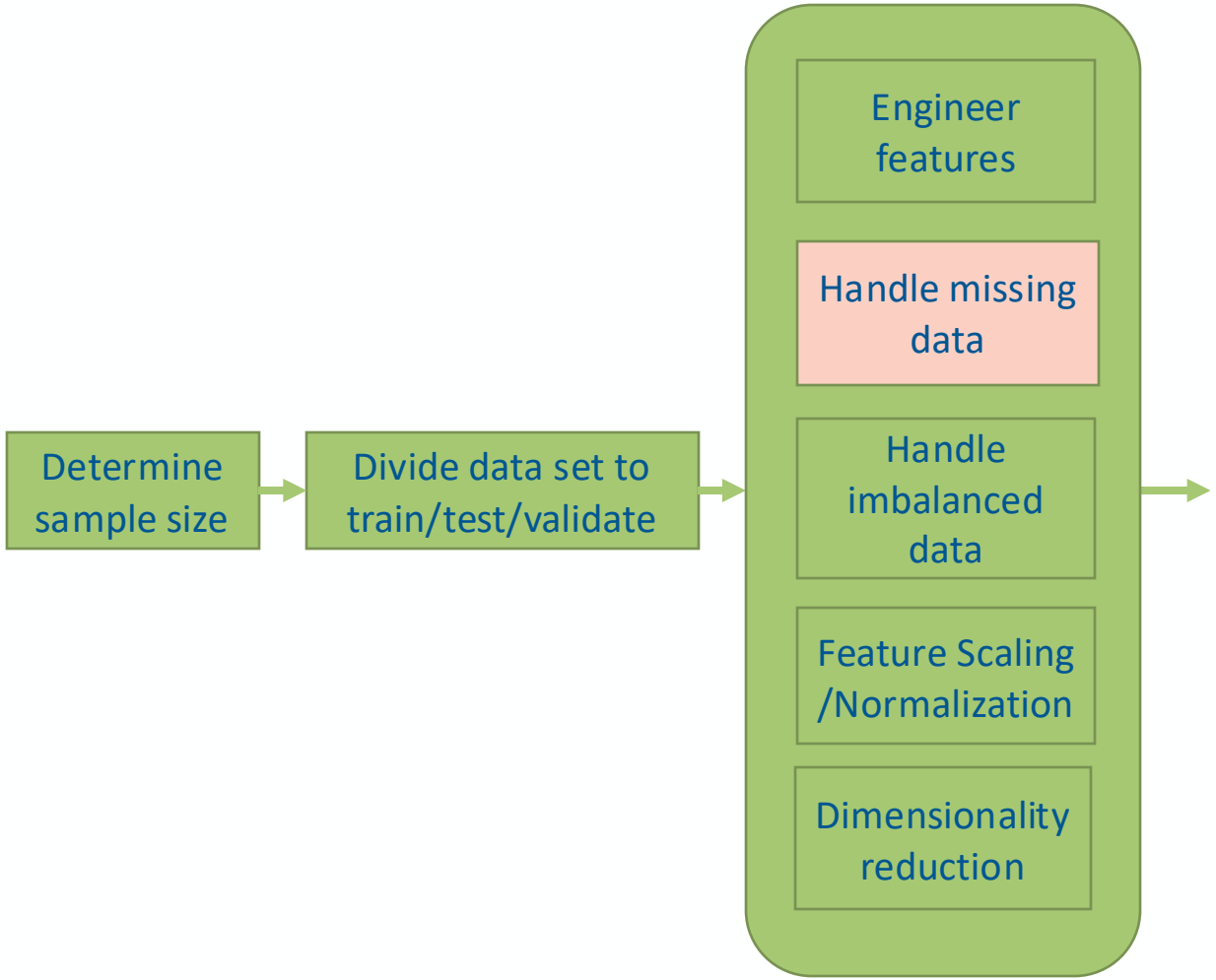
Binning (also called bucketing) is a feature engineering technique that groups different numerical subranges into bins or buckets. In many cases, binning turns numerical data into categorical data.

Example: consider a feature named Age with lowest value is 0 and highest value is 130. Using binning, you could represent it with the following five bins:

- Bin 1:infant 0 to 1
- Bin 2: toddler 1 to 3
- Bin 3: pre-schooler. 3 to 6
- Bin 4: Grade-schooler 5 to 12
- Bin 5: Teen 12 to 18
- Bin 6: Young Adult 18-21
- …

A model trained on these bins will react differently to X values of 1 and 1.5 since both values are in two different Bins and similarly to 6 and 10 since both are in Bin 4.

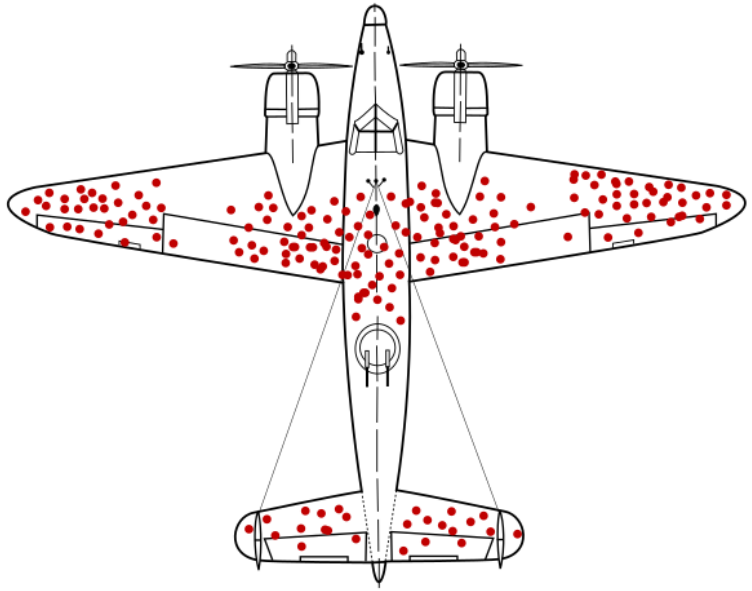
Preprocessing: strategies to handle missing data



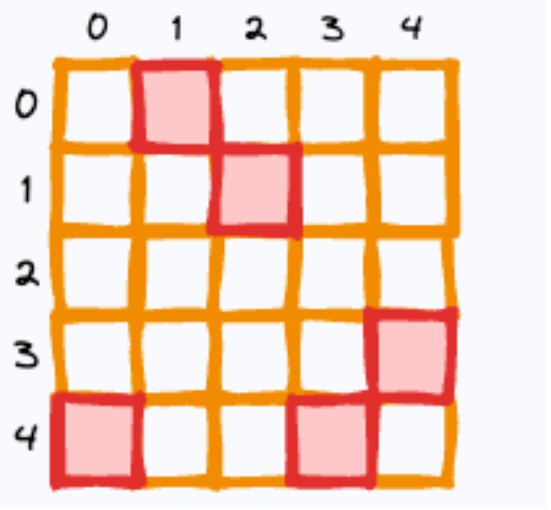
- Define the pattern of missing data.
- Process the missing data (dropping, replacement, imputation…etc.)
- Consider using algorithm which is robust to missing data.

...

Preprocessing: **Handle missing data**
missing categories or missing values



survivorship bias
missing data points or categories



Missing values

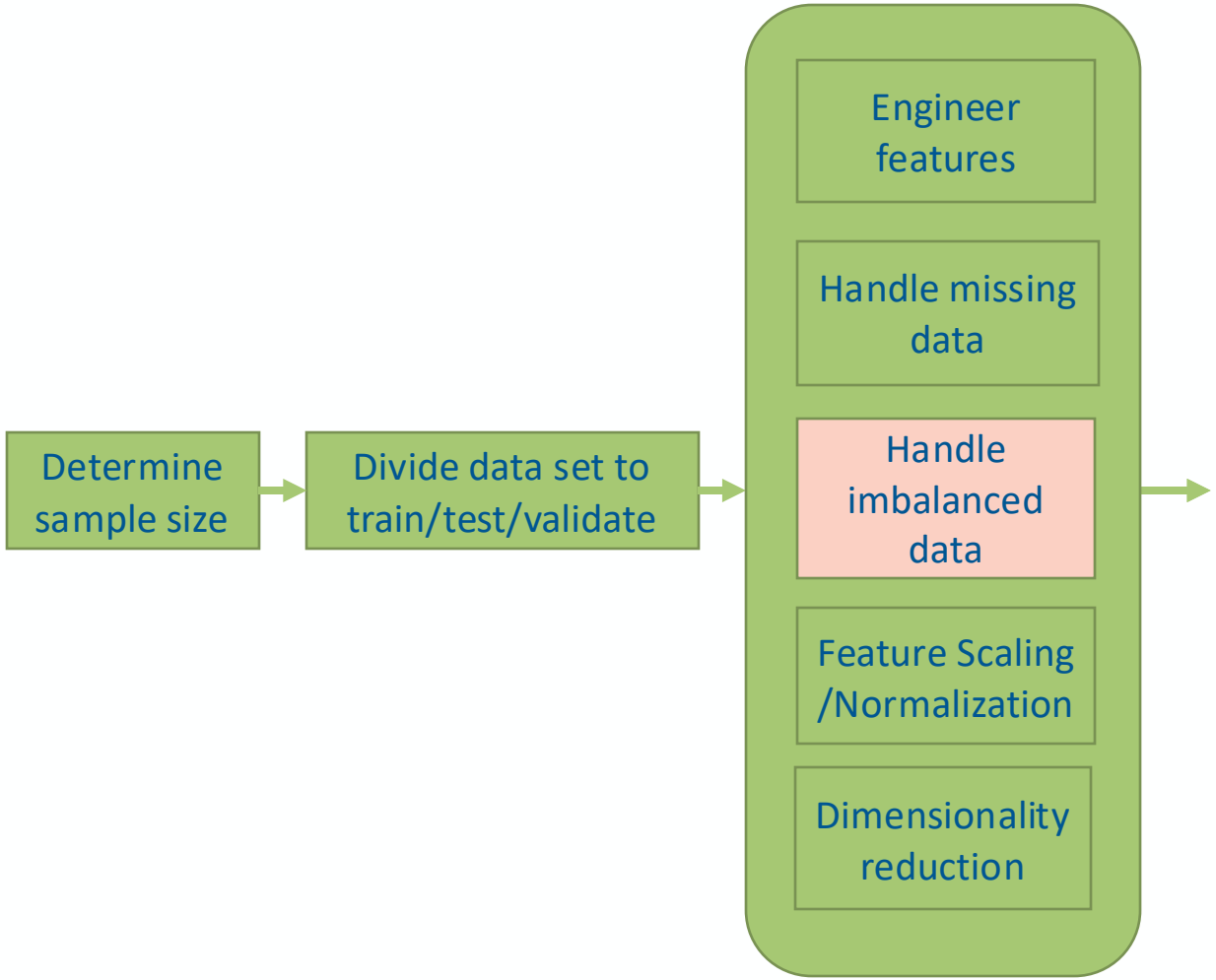
Strategies for handling missing values

- Ignore records with missing data
- Include NA indicator variable that become a part of the overall modelling.
- Do nothing and use method which can use data record with missing data
- **Impute (fill in) missing values (= guess!)**

Strategies for Imputing missing values

- **Manual Imputation**
  Expert or domain knowledge is used to fill missing values.
- **Global Constant**
  Replace with a constant like "unknown" or -999.
- **Statistical Imputation**
  - **Mean/Median of Variable**
    Replace missing values with the **mean or median** of that feature.
  - **Class-wise Mean/Median**
    If labels are available, use the **mean/median within each class**.
- **Model-based Imputation**
  Use other variables to predict missing values **using regression model**, **decision trees**, or other models.

Preprocessing: **strategies to handle imbalanced data**



- **Imbalanced data:** Data set in which some cases or risk categories occur much less frequently than the others.

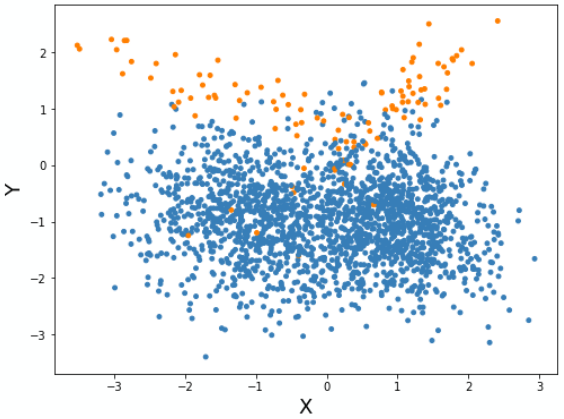
What can we do?

1. Use a model or an objective functions suited to handling imbalanced data
2. Use data resampling techniques
3. Use data augmentation to augment the rare classes
4. Use balanced to suited evaluation metrics

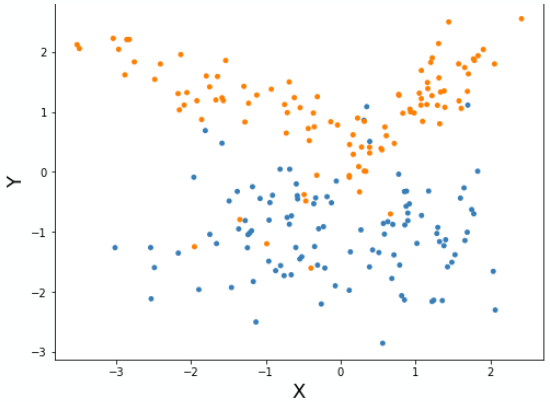
Preprocessing: **strategies to handle imbalanced data**

Determine sample size → Divide data set to train/test/validate → Engineer features / Handle missing data / Handle imbalanced data / Feature Scaling /Normalization / Dimensionality reduction

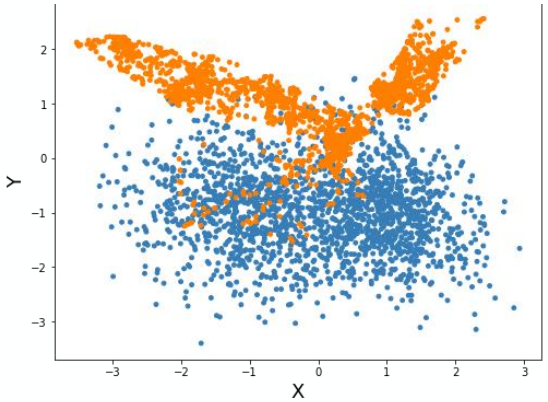
Imbalanced Dataset



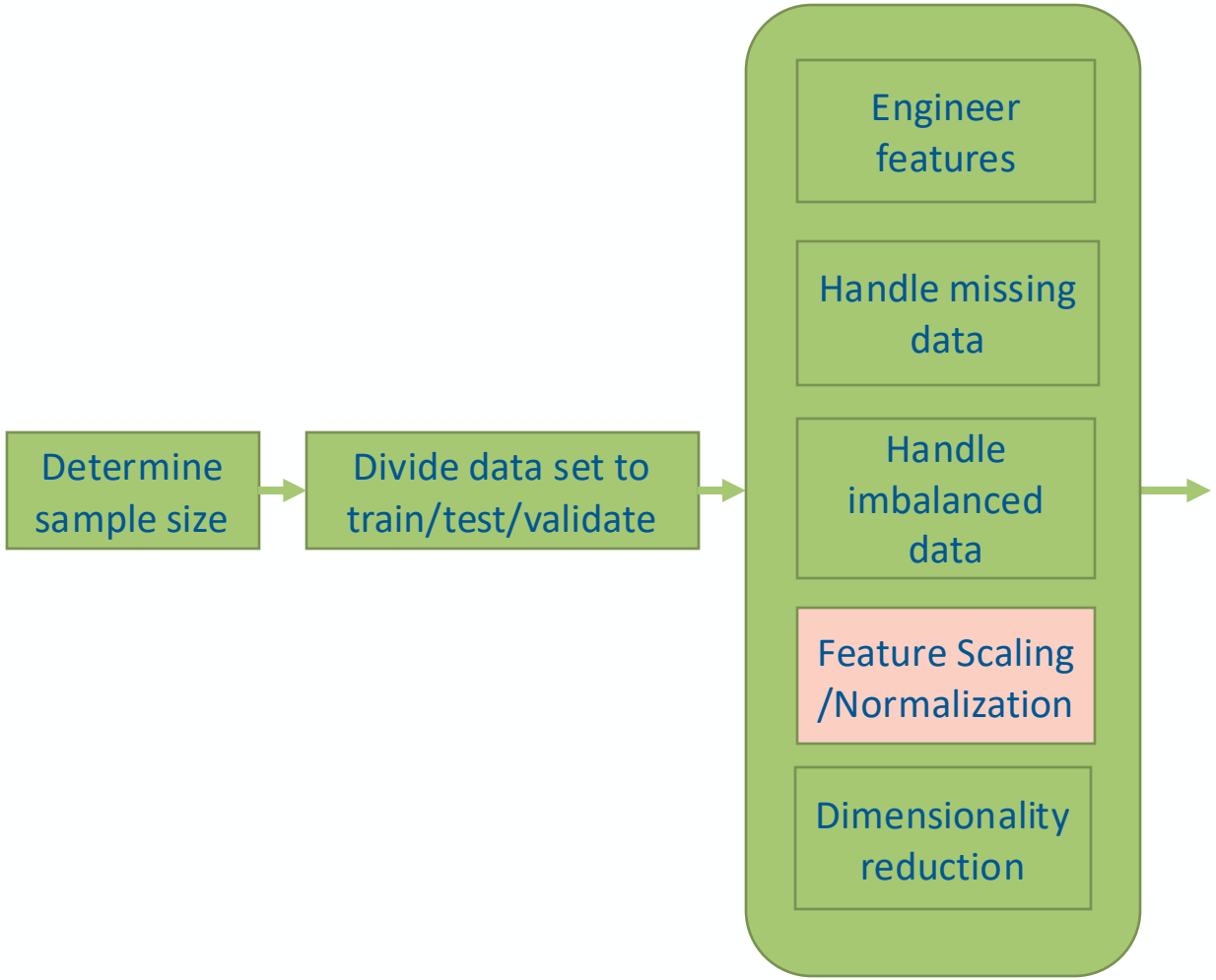
Downsample to balance



Upsample or augment



Preprocessing: Feature Scaling and Normalization



📌 **Why features scaling Matters**

🚀 **Improves model performance** by ensuring that features **contribute equally**.

⚖️ Prevents features with larger scales from **dominating others**.

📏 Required for **distance-based algorithms** (e.g., KNN, SVM, clustering).

🔄 **Speeds up convergence** in gradient-based algorithms (e.g., neural networks).

🔥 Helps avoid exploding/vanishing gradients in **deep learning models**.

Common Methods

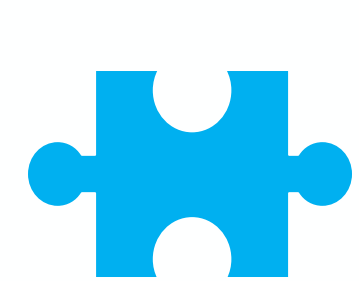
Method	Description	Typical Use Case
Min-Max Scaling	Rescales values to a [0, 1] range	Neural networks, image data
Z-score (Standardization)	Centers data to mean 0, std 1	Most ML algorithms, general purpose
Robust Scaling	Uses median and IQR, resistant to outliers	Datasets with heavy-tailed features

🧠 Important for Production

✅ Always **fit scalers on training data only**.

🔁 Apply **same transformation to test/production** data.

📦 **Normalization parameters (mean, std, min, max, etc.) are part of the model** and must be saved and reused in deployment.



Questions

Answers