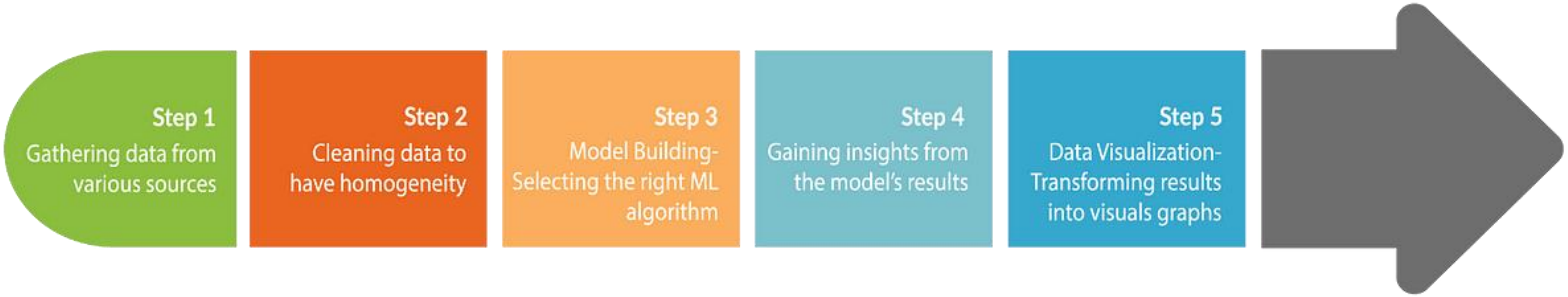
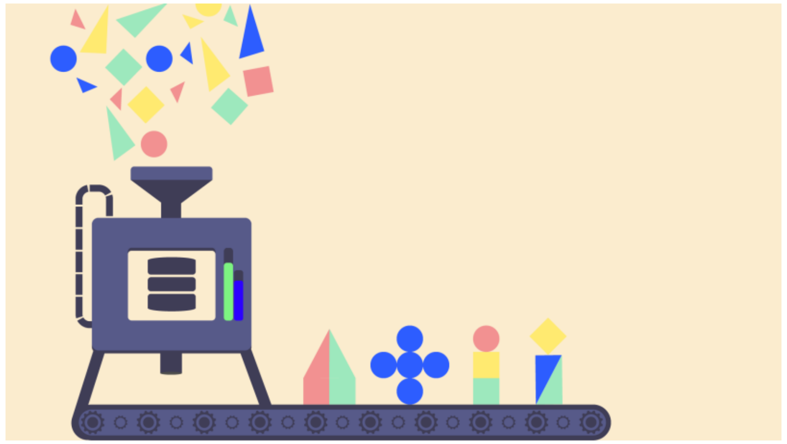
FeatureEgineering

The Machine Learning Process


Step 1
Gathering data from various sources
Step 2
Cleaning data to have homogeneity
Step 3
Model Building- Selecting the right ML algorithm
Step 4
Gaining insights from the model's results
Step 5
Data Visualization- Transforming results into visuals graphs

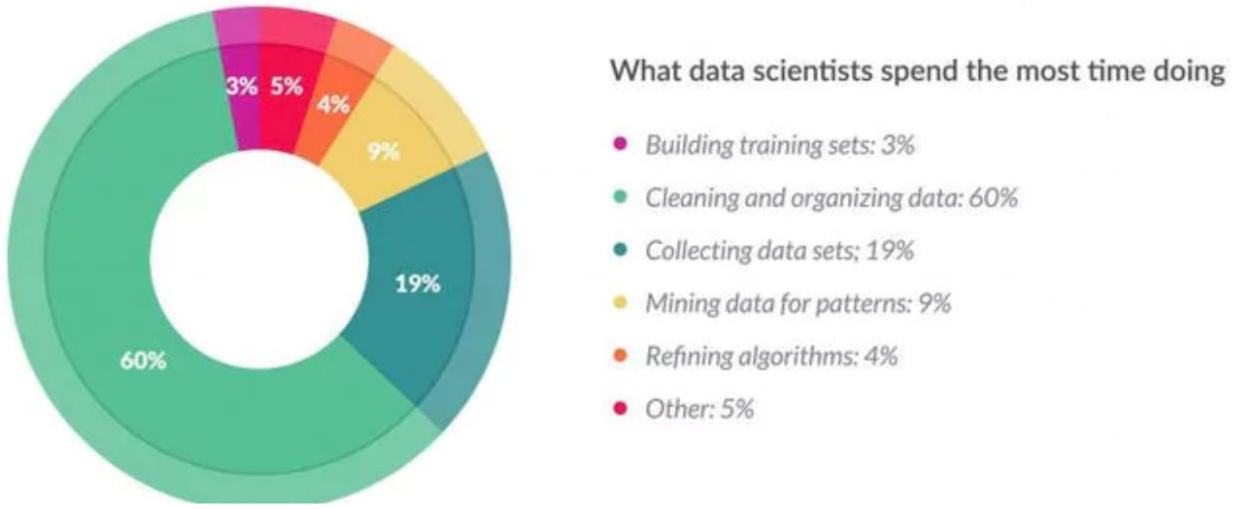
Overview

- ❑ • Data Preprocessing & Feature Engineering Basics
- ❑ • Feature Scaling & Transformation
- ❑ • Feature Encoding (Categorical Data)
- ❑ • Feature Cleaning & Imputation
- ❑ • Splitting Data for Model Training & Testing



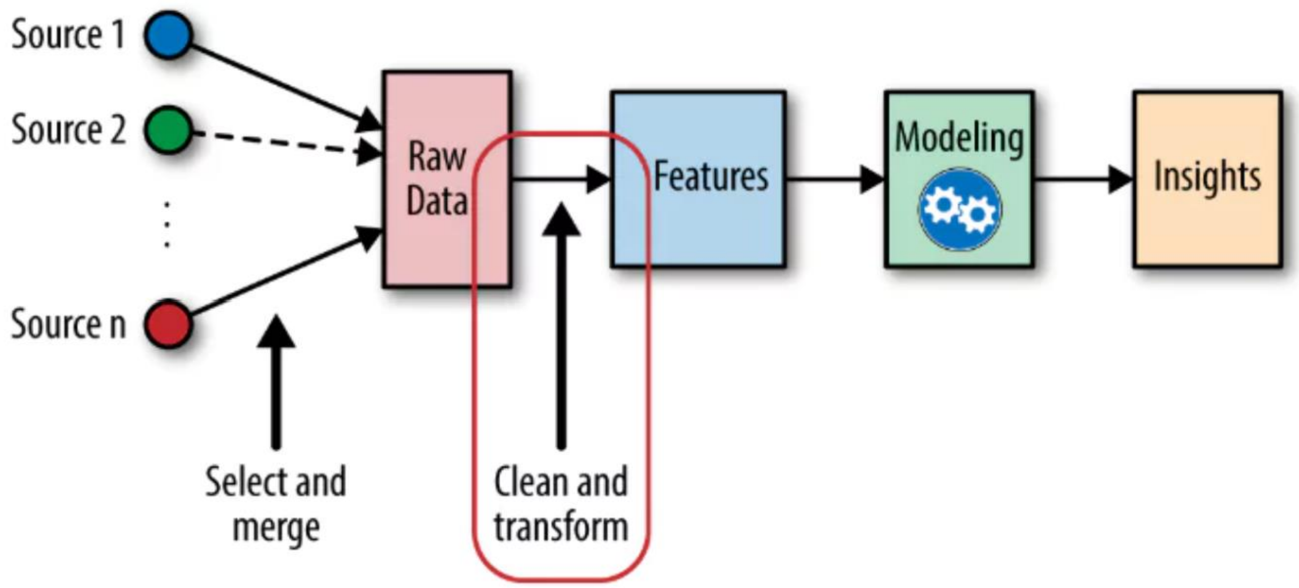
Why is Feature Engineering so important?

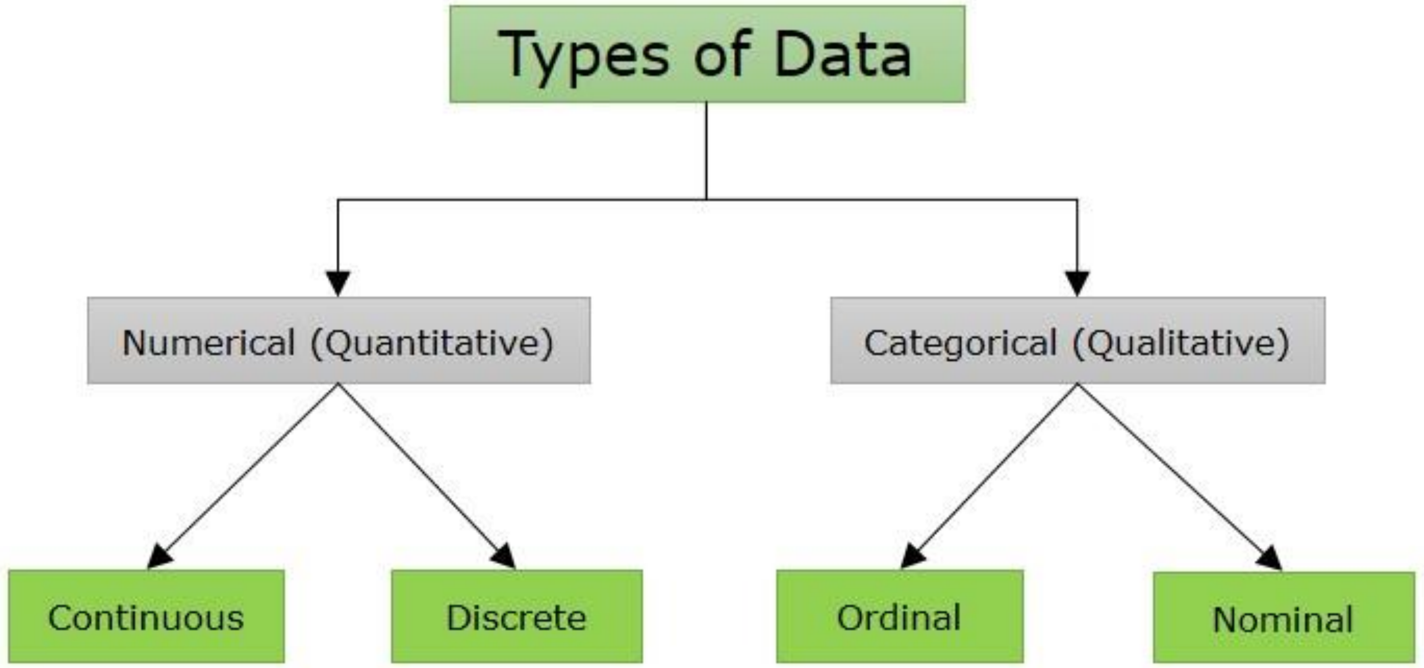
Do you know what takes the maximum amount of time and effort in a Machine Learning workflow?



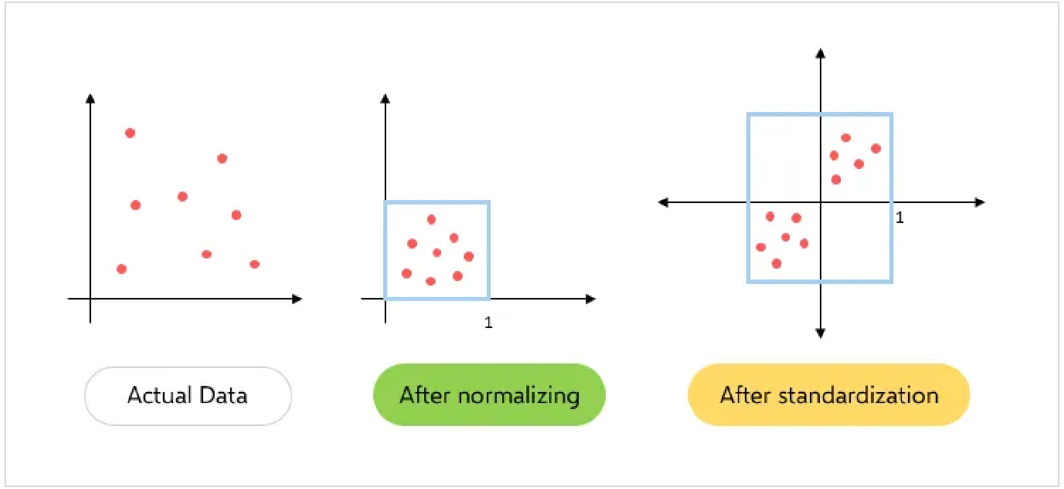
Feature Engineering

Is the process of using the Domain Knowledge to choose well prepared variables




Types of Data
Numerical (Quantitative)
Categorical (Qualitative)
Continuous
Discrete
Ordinal
Nominal

Feature Scaling & Transformation



- **Normalization**
- Normalization is the process of scaling the data values in such a way that that the value of all the features lies between 0 and 1

- $$x_{\text{norm}} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

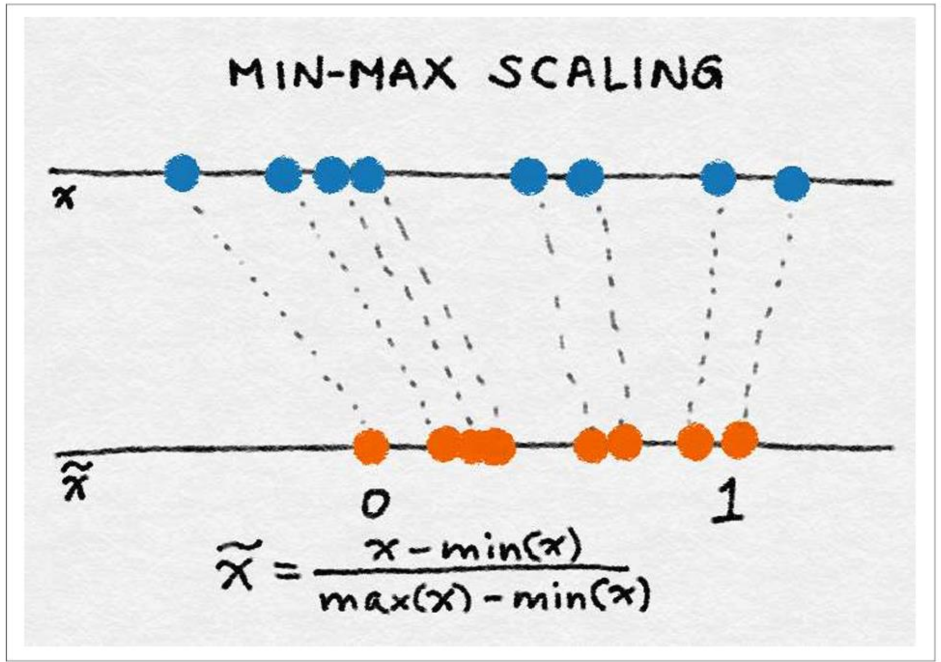
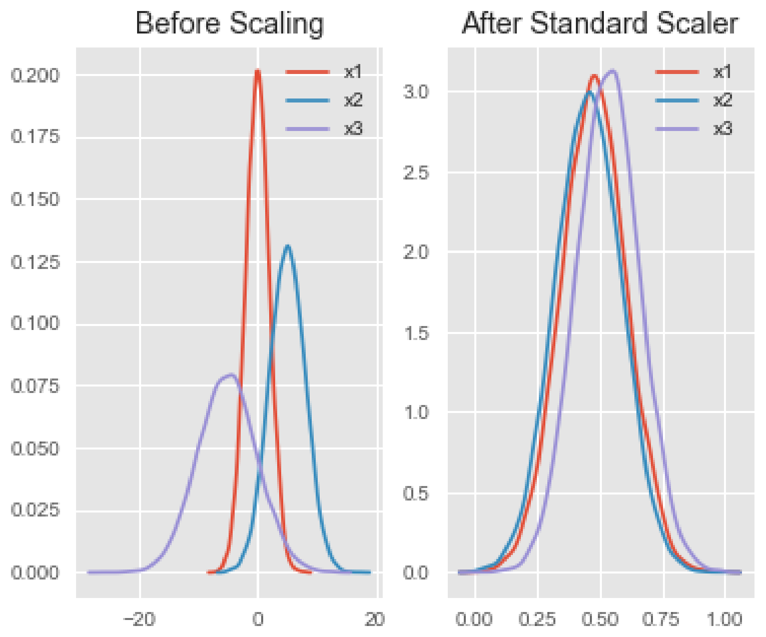


Figure 2-15. Illustration of min-max scaling

Standardization

Standardization is the process of scaling the data values in such a way that that they gain the properties of standard normal distribution. This means that the data is rescaled in such a way that the mean becomes zero and the data has unit

$$x_{\text{stand}} = \frac{x - \text{mean}(x)}{\text{standard deviation } (x)}$$



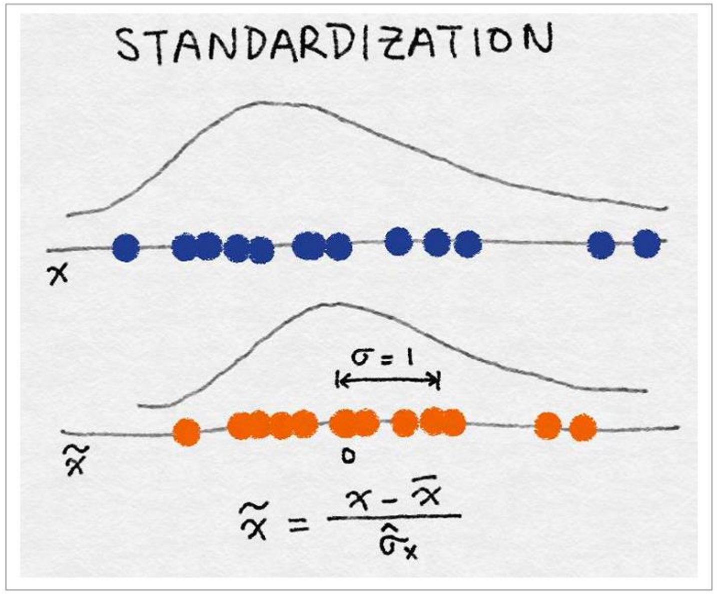


Figure 2-16. Illustration of feature standardization

in practice, more often benefit from standardization than from normalization;

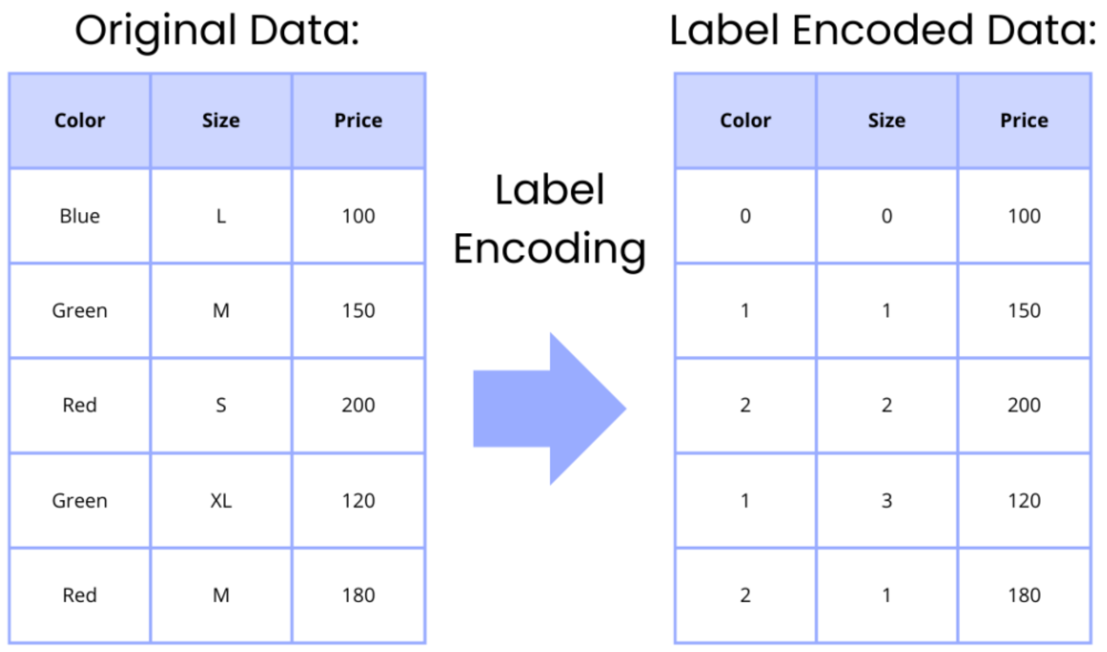
- Standardization is also preferred for a feature if the values this feature takes are distributed close to a normal distribution (so-called bell curve)
- standardization is preferred for a feature if it can sometimes have extremely high or low values (outliers); this is because normalization will “squeeze” the normal values into a very small range
- In all other cases, normalization is preferable.
- Feature rescaling is usually beneficial to most learning algorithms. However, modern implementations of the learning algorithms, which you can find in popular libraries, are robust to features lying in different ranges.

Feature Encoding (Catogrical data)

Label Encoding

a data preprocessing technique used in machine learning to convert categorical data into a numerical format

It’s often used for ordinal variables where there’s a clear order to the categories, such as education levels (e.g., primary, secondary, tertiary) or product ratings (e.g., 1 star, 2 stars, 3 stars).


Label
Encoding

Original Data:

Color	Size	Price
Blue	L	100
Green	M	150
Red	S	200
Green	XL	120
Red	M	180

Label Encoded Data:

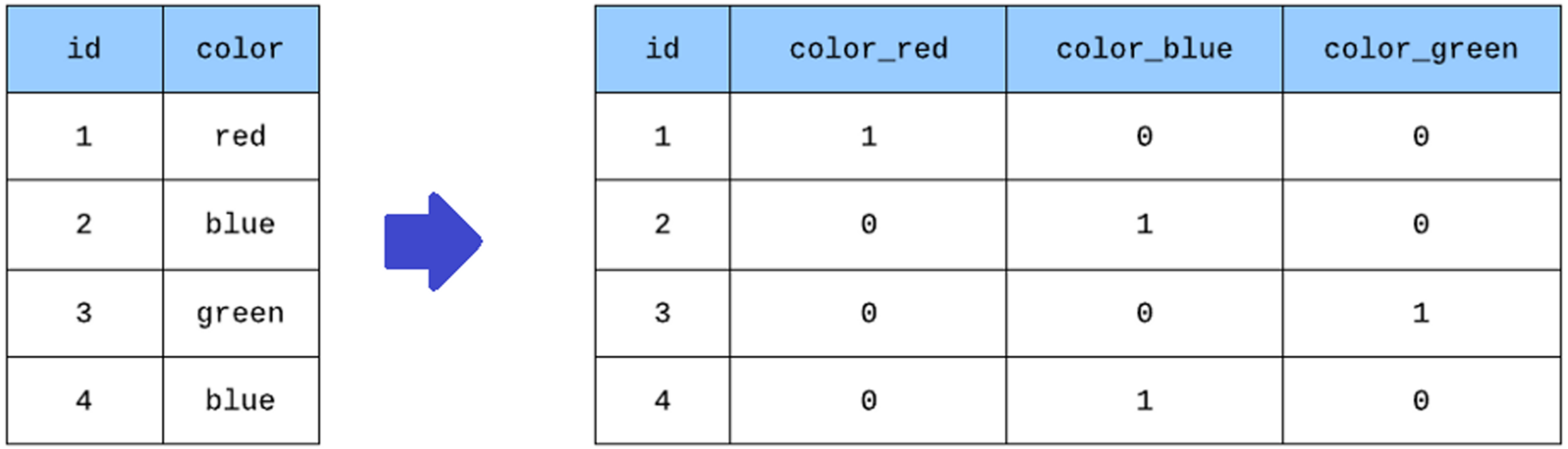
Color	Size	Price
0	0	100
1	1	150
2	2	200
1	3	120
2	1	180

Import the necessary libraries:

```python
from sklearn.preprocessing import LabelEncoder

# Sample data
categories = ['red', 'blue', 'green', 'red', 'blue']

# Create a LabelEncoder object
label_encoder = LabelEncoder()

# Fit the encoder and transform the data
encoded_categories = label_encoder.fit_transform(categories)

print(encoded_categories)  # Output:
```

One Hot Encoding

One-hot encoding is a technique used in machine learning to represent categorical data as numerical data. It converts each category into a binary vector, where only one element is "on" (1) and the rest are "off" (0). This allows machine learning models, which typically require numerical input, to work with categorical feature



id	color
1	red
2	blue
3	green
4	blue

id	color_red	color_blue	color_green
1	1	0	0
2	0	1	0
3	0	0	1
4	0	1	0

Missing Values

Missing values are essentially gaps in your data, where information for a specific variable is absent for certain observations

Why are missing values a problem?

Missing values can lead to inaccurate or biased results in data analysis, especially if the missing data isn't random

- **Blank cells**
- **Null values**
- **NaN (Not a Number)**

The Reasons For Missing Values:

•**Missing Completely at Random (MCAR):**

•The missingness is unrelated to any other variables in the dataset. This is the easiest scenario to deal with.

•**Missing at Random (MAR):**

•The missingness is related to other variables in the dataset, but not to the missing value itself. This requires more sophisticated handling.

•**Missing Not at Random (MNAR):**

•The missingness is related to the missing value itself. This is the most difficult scenario and may require special techniques or careful consideration.

How to handle missing values?

- **Deletion:**

- **Imputation:**

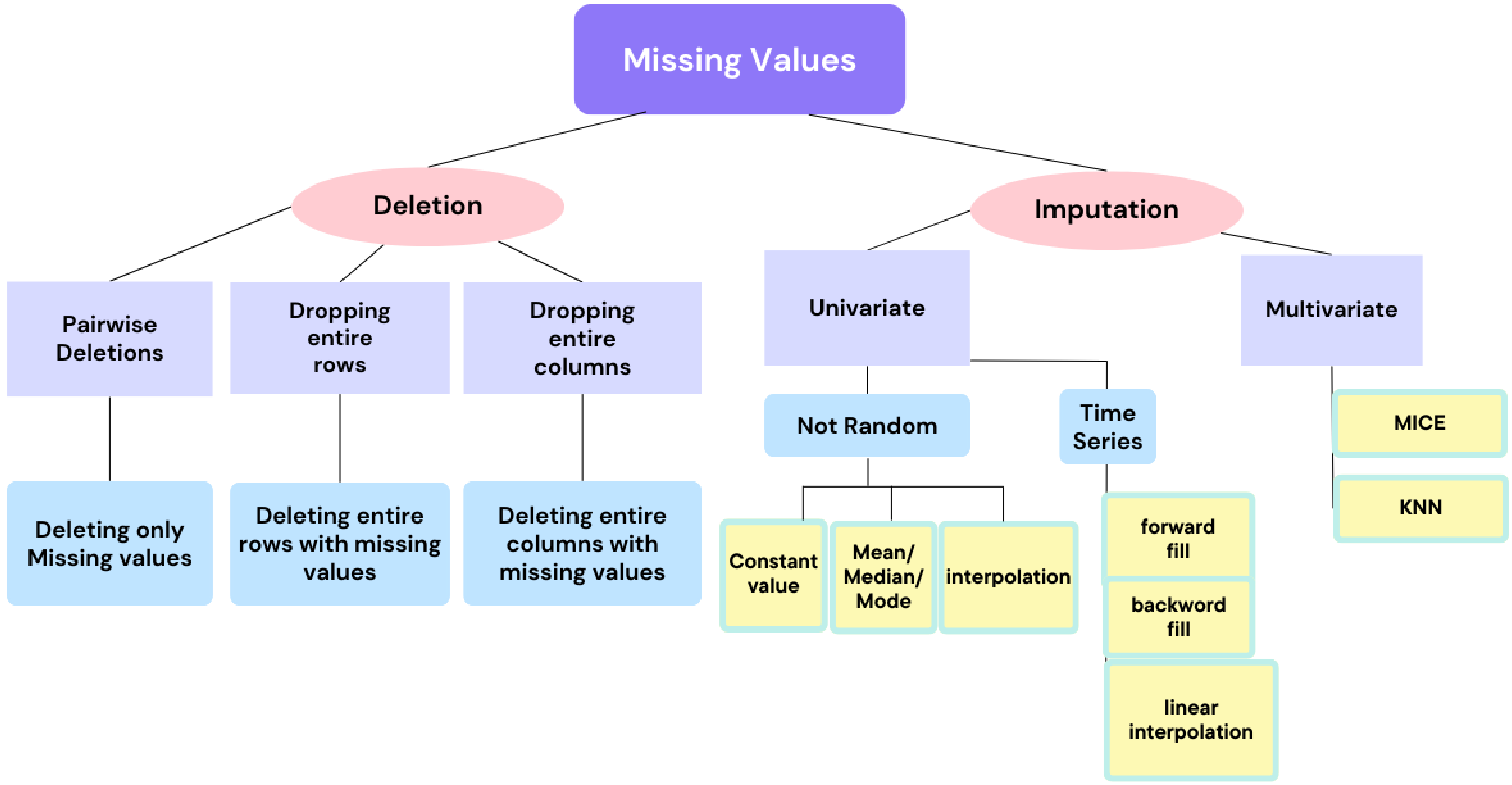
Mean/median imputation: Replacing missing values with the mean or median of the variable.

Mode imputation: Replacing missing values with the most frequent value (mode) for categorical variables.

K-Nearest Neighbors (KNN) imputation: Estimating missing values based on the values of similar observations.

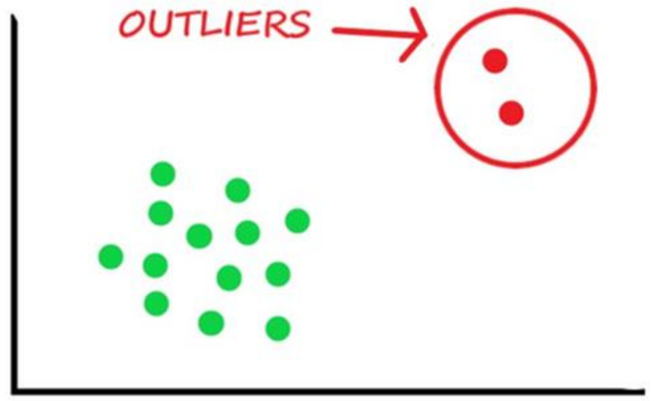
Model-based imputation: Using machine learning models to predict missing values based on other variable

How to handle missing values?



Detect & Handle Outliers

outliers :
are data points that are significantly different from the majority of data in a dataset.



What's outliers can do ?

Distort the mean and standard deviation:
They can make the average value much higher or lower than it should be.
Influence regression lines:
In linear regression, outliers can shift the regression line, making predictions inaccurate.

Identifying Outliers

1. Visualization:

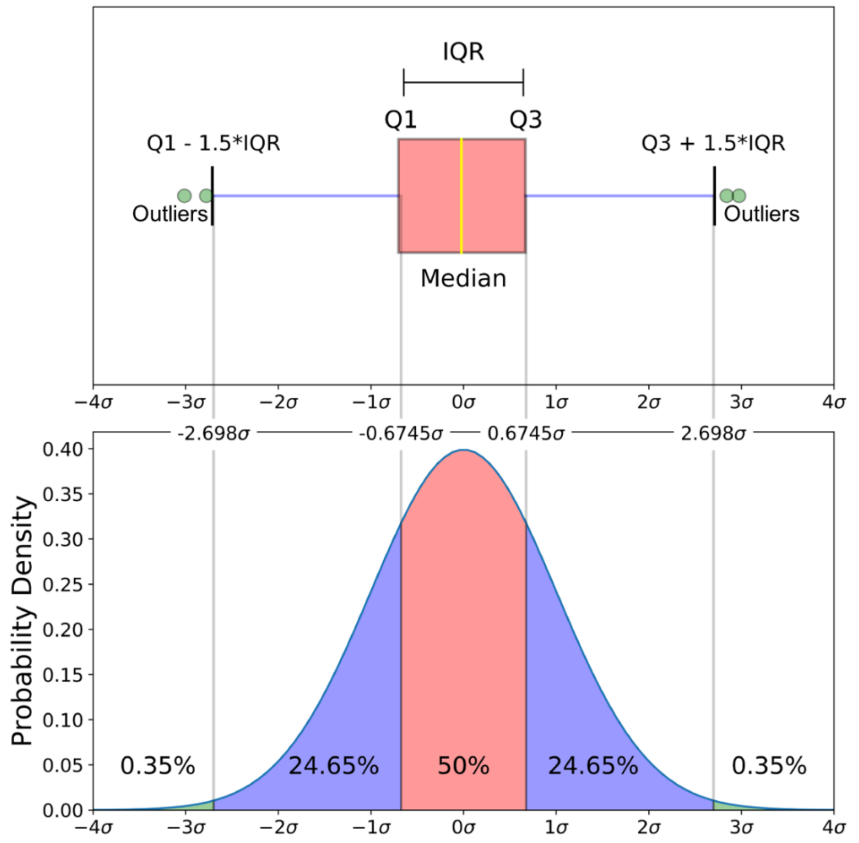
2. Statistical Methods:

Z-score:

Measures how far a data point is from the mean in terms of standard deviations. If a Z-score is higher than a certain threshold (typically ±3), it's considered an outlier.

Interquartile Range (IQR):

IQR is a measure of statistical dispersion and is calculated as the difference between the 75th percentile (Q3) and the 25th percentile (Q1). Data points that fall below Q1 - 1.5 * IQR or above Q3 + 1.5 * IQR are considered outliers.



Handling Outliers

1. **Remove the Outliers:**

2. Transform Data:

Apply transformations like **normalization** to reduce the impact of outliers. This approach is helpful in datasets with values that vary widely, such as income data.

3. Use Robust Algorithms:

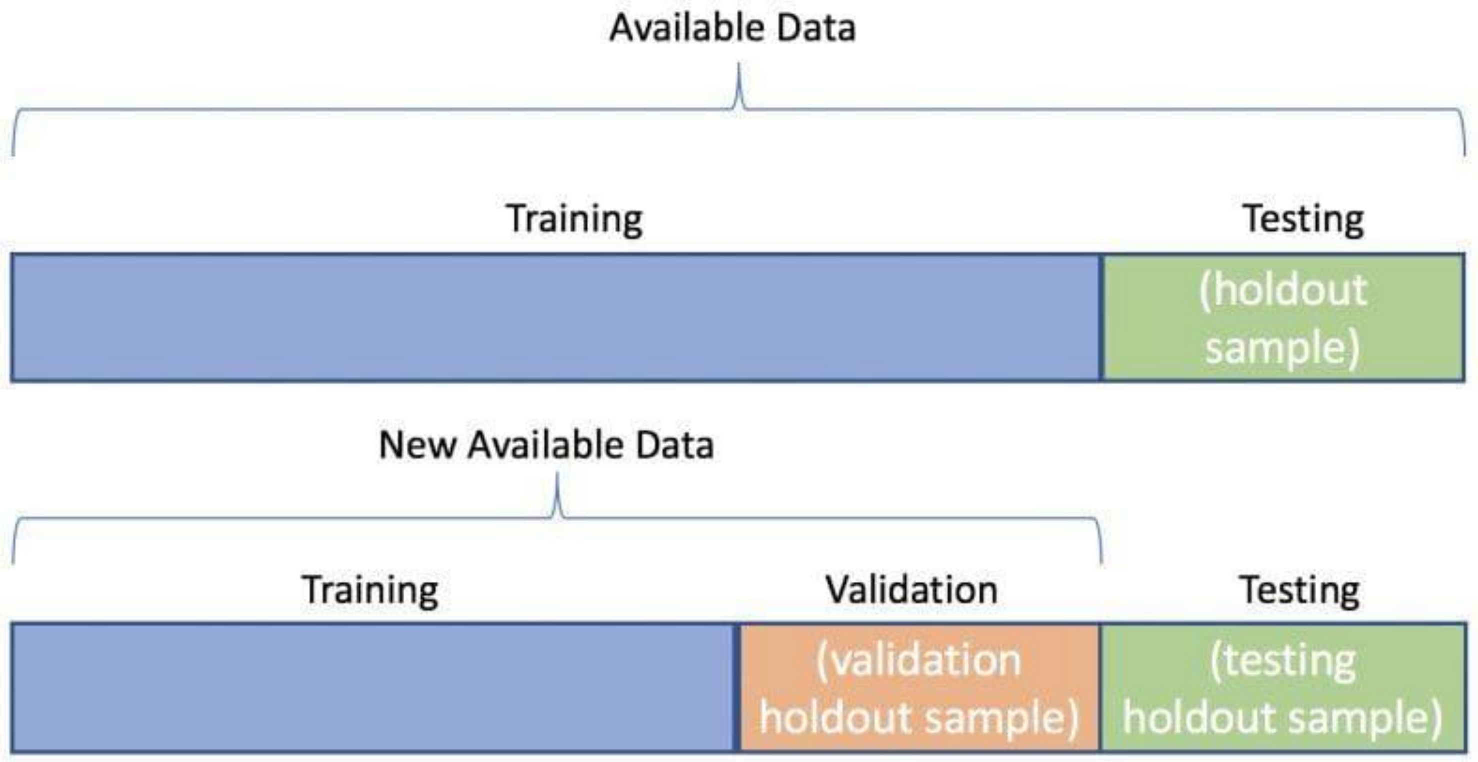
Certain algorithms like **decision trees** and **random forests** are less sensitive to outliers. When working with data that naturally includes outliers, such as financial data, these algorithms can help reduce the impact of outliers on the model.

4. Use Clipping:

Clipping involves setting an upper or lower limit on values. For example, you could clip ages at 100 if it's unlikely anyone in the dataset is older.

Splitting the Data Set for Training &Testing

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.



Why is data splitting important?

- •Preventing Overfitting:
- •Hyperparameter Tuning:
- •Model Evaluation:
- •Bias and Variance Assessment:

Common data splitting techniques

Train/Test Split:
The simplest method, dividing the data into a training set for model development and a testing set for performance evaluation

Train/Validation/Test Split:
A more robust approach that adds a validation set for hyperparameter tuning.

Cross-Validation:
Techniques like k-fold cross-validation divide the data into k subsets, using each in turn as a validation set while training on the remaining k-1 subsets.

Stratified Sampling:
Ensures that the distribution of classes in the dataset is preserved in each subset, especially useful for imbalanced datasets.

Time Series Splitting:
Preserves the temporal order of data points when dealing with time series data