**1. Types of Missing Data**

There are three main types of missing data:

**Missing Completely at Random (MCAR)**: The missingness is completely independent of both observed and unobserved data. For example, survey responses lost due to technical issues.

**Missing at Random (MAR)**: The missingness depends on observed data but not on the missing values themselves. For instance, younger people being less likely to report their income, but this pattern is observable in the age variable.

**Missing Not at Random (MNAR)**: The missingness depends on the unobserved values themselves. For example, people with very high incomes refusing to disclose their salary specifically because it's high.

2. We handle the Categorical Variables by

| **Method** | **When Used** | **Example (Titanic)** |
| --- | --- | --- |
| **Label Encoding** | Ordinal categories (with order) | e.g., small/medium/large bins |
| **One-Hot Encoding** | Nominal categories (no order) | Embarked: C, Q, S |
| **Target Encoding** | Large cardinality with leakage control | e.g., encoding cities by survival rate |

**3. Normalization vs Standardization**

**Normalization** (Min-Max scaling) transforms data to a fixed range, typically 0-1, using the formula: (x - min) / (max - min). It preserves the original distribution shape but is sensitive to outliers.

**Standardization** (Z-score normalization) transforms data to have mean 0 and standard deviation 1 using: (x - mean) / standard deviation. It's less sensitive to outliers and works well when data follows a normal distribution. Standardization is often preferred for algorithms that assume normally distributed data.

**4. Detecting Outliers**

Multiple methods exist for outlier detection:

**Statistical methods** include the IQR method (values beyond Q1 - 1.5×IQR or Q3 + 1.5×IQR) and Z-score (typically |z| > 3).

**Visualization techniques** like box plots, scatter plots, and histograms help identify outliers visually.

**Machine learning approaches** include Isolation Forest, Local Outlier Factor, and DBSCAN clustering. The choice depends on data distribution, dimensionality, and domain knowledge.

| **Method** | **Description** | **Example** |
| --- | --- | --- |

|  |  |  |
| --- | --- | --- |
| **Method** | **Description** | **Example** |
| **Boxplot** | Visual method using IQR | Titanic: Fare outliers > 3 std dev |
| **IQR Rule** | Q1 - 1.5×IQR or Q3 + 1.5×IQR | Fare > upper bound = outlier |
| **Z-Score** | Values > | 2 |

**5. Importance of Preprocessing in ML**

Preprocessing is crucial because most ML algorithms make assumptions about data format and distribution. Raw data often contains noise, missing values, inconsistent formats, and varying scales that can severely impact model performance. Proper preprocessing improves model accuracy, reduces training time, ensures algorithm stability, and helps meet statistical assumptions. It's often said that 80% of data science work involves data cleaning and preprocessing.

**6. One-Hot Encoding vs Label Encoding**

**Label encoding** assigns a unique integer to each category (e.g., Red=0, Blue=1, Green=2). It's suitable for ordinal data but can introduce artificial ordering for nominal data, which some algorithms might interpret incorrectly.

**One-hot encoding** creates binary columns for each category, with 1 indicating presence and 0 indicating absence. It's ideal for nominal data as it doesn't impose ordering, but increases dimensionality significantly with high-cardinality variables. Tree-based algorithms can handle label encoding well, while linear models typically prefer one-hot encoding.

|  |  |  |
| --- | --- | --- |
| **Encoding Type** | **Description** | **Titanic Example** |
| **Label Encoding** | Converts to 0, 1, 2,… (ordinal risk) | Sex: male=0, female=1 |
| **One-Hot** | Creates separate binary columns | Embarked\_C, Embarked\_Q… |
| Use When | Label: Ordinal data | One-Hot: Nominal data with <10 categories |

**7. Handling Data Imbalance**

Several strategies address imbalanced datasets:

**Sampling techniques** include oversampling the minority class (SMOTE, ADASYN), undersampling the majority class, or combining both.

**Algorithm-level approaches** involve using class weights, cost-sensitive learning, or ensemble methods like balanced random forests.

**Evaluation metric changes** focus on precision, recall, F1-score, or AUC-ROC instead of accuracy.

**Data generation** through synthetic sample creation can also help balance classes.

**8. Preprocessing Effects on Model Accuracy**

Preprocessing significantly impacts model accuracy. Poor preprocessing can lead to data leakage, reduced model performance, and unreliable predictions. Proper scaling improves convergence in gradient-based algorithms, appropriate encoding prevents misinterpretation of categorical variables, and outlier handling prevents model bias. However, over-preprocessing can remove valuable information, so the key is finding the right balance based on your specific dataset and problem domain.

Each preprocessing step should be guided by understanding your data, the algorithms you plan to use, and the business problem you're solving.