## 8. Standardize/Normalize Data

### Why It's Important

Data with varying scales can negatively impact machine learning models, especially distance-based models like k-NN and SVM.

### Steps to Standardize/Normalize Data:

1. **Choose the Scaling Method**:
    - **Standardization**:
        - Rescales data to have a mean of 0 and a standard deviation of 1.
        - Suitable for algorithms that assume normally distributed data (e.g., linear regression, SVM).
    - **Normalization**:
        - Rescales data to a fixed range, usually [0, 1].
        - Suitable for algorithms that do not assume normal distribution (e.g., neural networks).
2. **Identify Features to Scale**:
    - Numerical features with different ranges (e.g., income in thousands vs. age in years).
    - Exclude categorical and already scaled features.
3. **Apply the Scaling**:
    - Fit the scaler to the training set and transform both training and testing sets.
    - Avoid using test data during fitting to prevent data leakage.
4. **Verify the Results**:
    - Check the mean and standard deviation (for standardization) or the range (for normalization) of the scaled features.

### Best Practices:

- Use standardization for algorithms sensitive to feature variance.
- Apply scaling only after data splitting to avoid information leakage.

---

## 9. Feature Engineering

### Why It's Important

Feature engineering enhances the predictive power of models by creating or transforming data to better represent the problem.

**Steps for Feature Engineering:**

1. **Create New Features**:
    - **Combine Existing Features**:
        - Example: Create a "total_price" feature from "price" and "quantity."
    - **Extract Information**:
        - From date-time: Extract "day of the week," "month," or "hour."
        - From text: Count words, extract sentiment, etc.
2. **Transform Features**:
    - Log or square-root transformations can reduce skewness in data.
    - Polynomial transformations can add interaction terms.
3. **Encode Categorical Data**:
    - Use one-hot encoding for unordered categories or ordinal encoding for ordered ones.
4. **Remove Irrelevant Features**:
    - Drop features that have little or no correlation with the target variable.
    - Example: IDs or highly correlated features.

**Best Practices:**

- Use domain knowledge to create meaningful features.
- Avoid creating overly complex or redundant features.

---

## 10. Split the Data

**Why It's Important**

Splitting ensures models are trained on a subset of data (training set) and tested on unseen data (testing set) to evaluate generalizability.

**Steps to Split Data:**

1. **Determine Split Ratios**:
    - Common split: 70% training, 30% testing.
    - For large datasets, consider 80%-20% or even 90%-10%.

2. **Stratify the Data (if necessary)**:
    - For classification tasks, stratify to maintain class distribution across splits.
3. **Create Validation Set (Optional)**:
    - Split training data into further subsets (e.g., 80%-20%) for hyperparameter tuning.
4. **Perform the Split**:
    - Use a random state for reproducibility.
    - Ensure the test set remains unseen during training and validation.

**Best Practices:**

- Use cross-validation for small datasets to maximize data utility.
- Avoid using test data for any model tuning or feature engineering.

---

## 11. Check for Multicollinearity

**Why It's Important**

Multicollinearity occurs when features are highly correlated, leading to redundancy and unstable coefficients in some models (e.g., linear regression).

**Steps to Check and Handle Multicollinearity:**

1. **Analyze Feature Correlations**:
    - Use a correlation matrix to measure relationships between features.
    - Focus on absolute correlation values > 0.85 or a chosen threshold.
2. **Visualize Relationships**:
    - Heatmaps provide an intuitive representation of feature correlations.
3. **Handle Highly Correlated Features**:
    - **Remove One of the Features**:
        - Keep the feature most strongly correlated with the target variable.
    - **Combine Features**:
        - Example: Combine "height" and "weight" into "body_mass_index (BMI)."
    - **Dimensionality Reduction**:
        - Use PCA (Principal Component Analysis) to transform correlated features into independent components.

4. **Verify After Adjustment**:
    - Recheck the correlation matrix to confirm multicollinearity is addressed.

**Best Practices:**

- Keep domain knowledge in mind when choosing which features to drop.
- Avoid multicollinearity even in unsupervised learning to ensure meaningful results.