Here:

- **Feature Matrix (X)**:

$$\mathbf{X} = \begin{bmatrix} 16 & 5.4 & 78 \\ 18 & 5.9 & 88 \\ 17 & 5.6 & 65 \end{bmatrix}$$

- **Label Vector (y)**:

$$\mathbf{y} = \begin{bmatrix} \text{Yes} \\ \text{Yes} \\ \text{No} \end{bmatrix}$$

Data Preprocessing in Machine Learning

Data preprocessing is a critical step in machine learning that involves transforming raw data into a clean, structured format that can be effectively used by algorithms. Proper preprocessing improves the model's performance, accuracy, and training speed, as well as helping to eliminate noise and irrelevant data.

Steps in Data Preprocessing

1. **Data Cleaning**
    - **Handling Missing Values**: Missing data can be managed by:
        - Removing rows or columns with missing values.
        - Imputing missing values with statistical measures like mean, median, or mode.
        - Using algorithms that can handle missing values directly.
    - **Outlier Detection and Removal**: Outliers, or extreme values, can skew results. These can be detected and either removed or adjusted to avoid negatively impacting the model.
    - **Handling Inconsistent Data**: Fixing errors in data entry, such as wrong formats, duplicate entries, and incorrect data types, is essential for maintaining data quality.
2. **Data Transformation**
    - **Normalization**: Scales the data to a range of [0, 1] or [-1, 1] to bring features to a common scale, useful for distance-based algorithms like K-Nearest Neighbors (KNN).
    - **Standardization**: Transforms data to have a mean of 0 and a standard deviation of 1, helpful for algorithms that assume normally distributed data, such as linear regression and logistic regression.
    - **Encoding Categorical Data**: Machine learning algorithms require numerical input, so categorical data must be converted:

        - **Label Encoding**: Assigns unique integers to each category.
        - **One-Hot Encoding**: Creates binary columns for each category, with a 1 indicating the presence of a category and 0 indicating absence.
3. **Feature Selection and Extraction**
    - **Feature Selection**: Selects the most relevant features, improving model accuracy and efficiency. Techniques include statistical tests, correlation analysis, and model-based selection.
    - **Feature Extraction**: Creates new features from existing ones. Techniques include **Principal Component Analysis (PCA)** for dimensionality reduction and **polynomial feature generation** for creating more complex relationships between features.
4. **Data Splitting**
    - Splits data into training and testing sets to evaluate model performance. Common ratios are 70-30 or 80-20 for training and testing, respectively.
    - Sometimes a **validation set** is also used to fine-tune model parameters, especially in more complex models.

Example of Data Preprocessing Steps

Imagine a customer dataset with the following steps applied for preprocessing:

1. **Data Cleaning**: Fill missing income values with the mean income and remove outliers in the age column.
2. **Normalization**: Scale age and income to a 0-1 range.
3. **Encoding Categorical Data**: Convert gender from categorical values (e.g., male, female) into one-hot encoding.
4. **Feature Selection**: Choose relevant features like income and age for predicting customer purchases.
5. **Splitting**: Divide the dataset into 80% training and 20% testing sets.

Importance of Data Preprocessing

- **Improves Model Accuracy**: Removing irrelevant information and reducing noise helps models learn more effectively.
- **Increases Model Efficiency**: Scaling and transforming data enable faster computation and better convergence.
- **Reduces Overfitting**: Helps eliminate unnecessary features and noise that can cause the model to perform poorly on new data.

Data preprocessing ensures data quality, which is essential for building accurate and reliable machine learning models.

Feature Normalization

Feature normalization is a preprocessing technique in machine learning that transforms features to a similar scale, often within a specific range. This process is essential for algorithms sensitive to feature scales, like distance-based methods (e.g., K-Nearest Neighbors and Support Vector Machines) and optimization-based methods (e.g., gradient descent in neural networks).

Why Feature Normalization is Important

1. **Improves Model Performance**: Different scales across features can lead to slower training, incorrect weights, and inaccurate results.
2. **Speeds Up Convergence**: Algorithms converge faster when data is normalized, as features contribute proportionally to the objective function.
3. **Enhances Interpretability**: Normalization makes it easier to interpret the influence of each feature on the model output.

Common Methods of Feature Normalization

1. **Min-Max Normalization**
    - Scales features to a specific range, usually between 0 and 1, or -1 and 1.
    - **Formula**:

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

    - **Use Case**: Suitable for data where features have known minimum and maximum values, or when maintaining relative feature relationships is essential.
2. **Z-Score Normalization (Standardization)**
    - Transforms features to have a mean of 0 and a standard deviation of 1.
    - **Formula**:

$$x_{\text{norm}} = \frac{x - \mu}{\sigma}$$

    - **Use Case**: Preferred when features have different units or scales, and in algorithms assuming normally distributed data (e.g., linear regression, logistic regression).
3. **Robust Normalization**