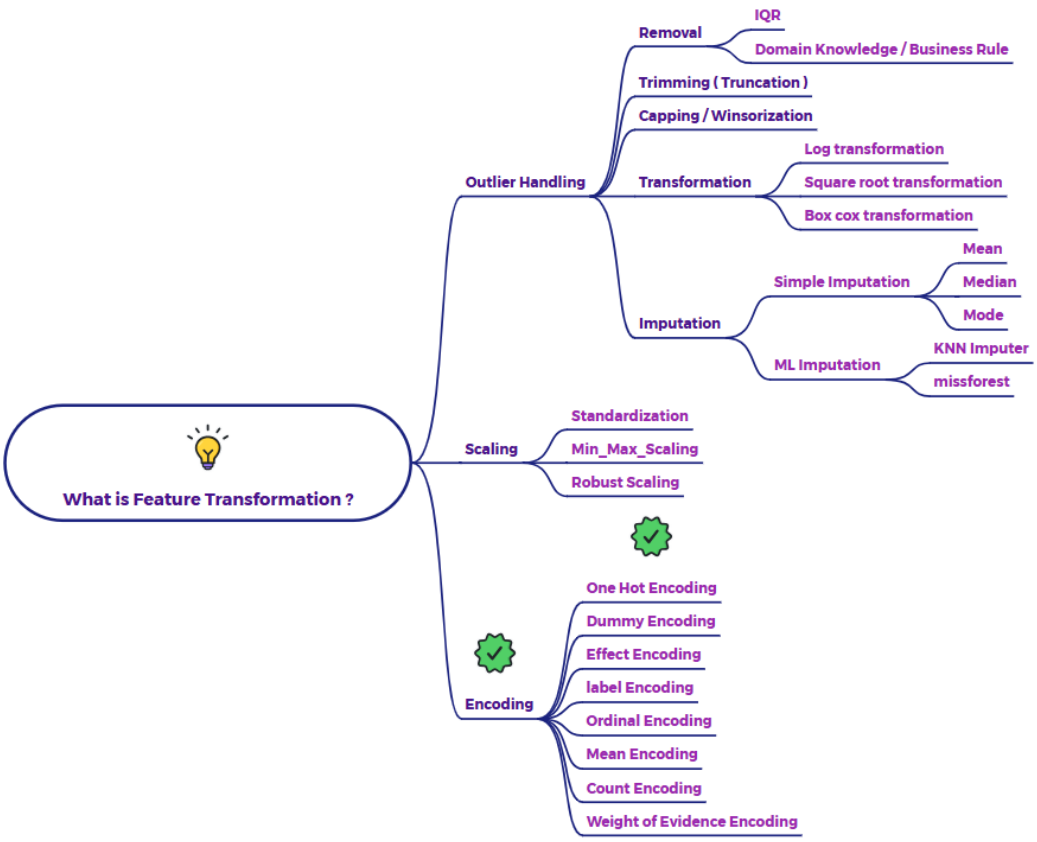
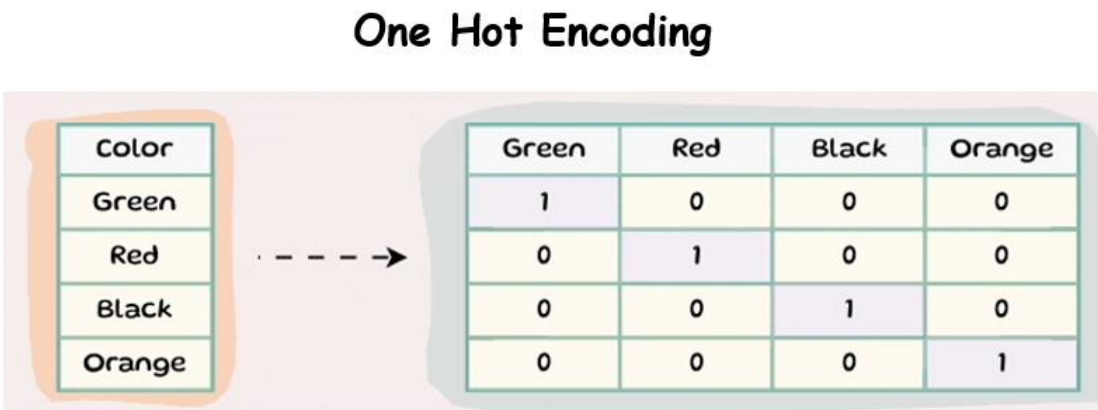
Explain One hot encoding with an example



1. **Explanation of One-Hot Encoding**

One-hot encoding is a technique used to convert categorical data into a numerical format that machine learning models can understand. It creates a new binary column for each unique category in the original categorical variable. For each row, the binary column corresponding to the row's category will have a value of 1, while all other binary columns will have a value of 0.


One Hot Encoding

Color
Green
Red
Black
Orange

Green	Red	Black	Orange
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

2. How to Calculate One-Hot Encoding

Here's a step-by-step explanation with an example:

Example:

Suppose we have a dataset with a "Color" column:

Color
Red
Blue
Green
Red
Blue

1. Identify Unique Categories: The unique categories in the "Color" column are "Red", "Blue", and "Green".

2. Create Binary Columns: For each unique category, create a new column. In this case, we'll create columns named "Color_Red", "Color_Blue", and "Color_Green".

3. Populate Binary Columns: For each row, assign a value of 1 to the column corresponding to the row's color, and 0 to the other columns.

The resulting one-hot encoded data looks like this:

Color	Color_Red	Color_Blue	Color_Green
Red	1	0	0
Blue	0	1	0
Green	0	0	1
Red	1	0	0
Blue	0	1	0

3. When to Use One-Hot Encoding

- When dealing with categorical variables that do not have an inherent order (nominal variables). Examples include:
  - Colors
  - Product types
  - City names
- When you want to avoid giving the model any ordinal bias that might be introduced by label encoding (where categories are assigned numerical labels in an arbitrary order).

- When your machine learning model expects numerical input and cannot directly handle categorical data.

4. **Strengths and Weaknesses of One-Hot Encoding**

    - **Strengths:**
        - Provides a clear representation of categorical data.
        - Does not introduce any ordinal bias.
        - Suitable for a wide range of machine learning algorithms.
    - **Weaknesses:**
        - Can significantly increase the dimensionality of the data, especially when dealing with categorical variables with many unique categories (high cardinality). This can lead to the "curse of dimensionality".
        - May lead to multicollinearity issues if not handled carefully (though many machine learning libraries can handle this).