Encoding of categorical variables

GROKKERS
AI FOR EVERYONE

Pre-reqs

- Python
- NumPy and PANDAS, SciPy, Visualizations
- Elementary stats and maths
- Some preprocessing steps – may need ML as well, for advanced topics

Background

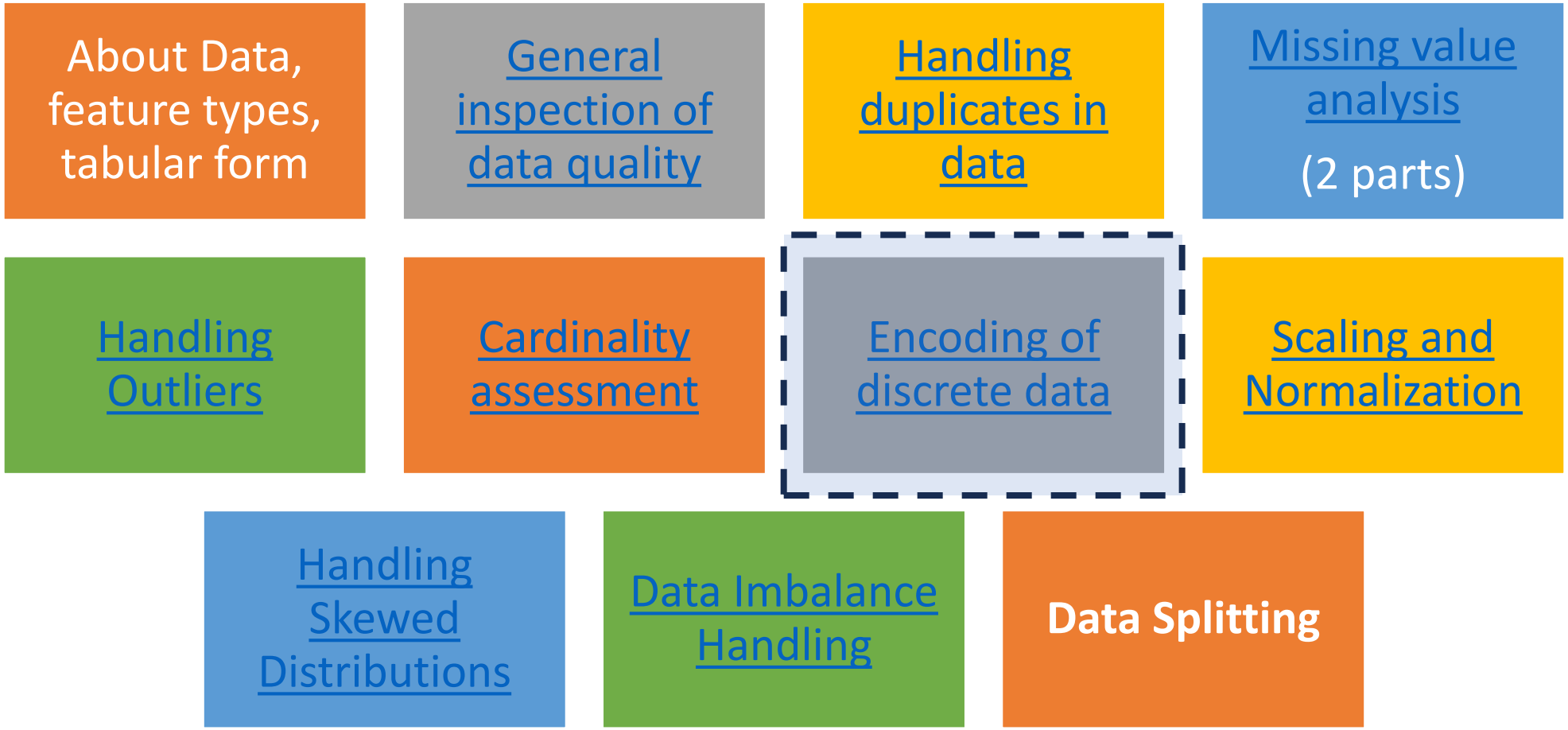
Numeric Data: Preprocessing involves handling missing values, scaling to a similar range, and possibly normalizing the distribution.

Text Data**:** Common preprocessing steps include text cleaning (removing stop words, punctuation, etc.), tokenization, and vectorization (converting text into numerical form, such as TF-IDF or word embeddings).

Image Data: Techniques like resizing, normalization of pixel values, and data augmentation (creating variations of existing images) are often used.

Time Series Data: Dealing with temporal aspects, handling missing values over time, and creating lag features are important steps in preprocessing time series data.

Topics



Encoding categorical variables

Binary, nominal, ordinal, label

Hands on python code

Define

01

Encoding categorical variables is a way to convert string/object data to numeric format

02

providing a way to include categorical data in machine learning models that require numerical input.

03

E.g. creating binary columns for each category and using 0s and 1s to indicate the presence or absence of each category.

Nominal Categorical Variables

Color Categories: Red, Blue, Green, etc.

Gender Categories: Male, Female, Non-binary, etc.

City Categories: New York, London, Tokyo, etc.

Animal Type : Dog, Cat, Bird, etc.

Ordinal Categorical Variables

Education Level:

- Categories: High School, Bachelor's, Master's, PhD, etc.
- Example Data: ['Bachelor's', 'Master's', 'High School', 'PhD', 'Bachelor's']

Customer Satisfaction Rating:

- Categories: Poor, Fair, Good, Excellent, etc.
- Example Data: ['Good', 'Excellent', 'Fair', 'Poor', 'Good']

Temperature Level:

- Categories: Low, Medium, High
- Example Data: ['Medium', 'High', 'Low', 'High', 'Medium']

Income Bracket:

- Categories: Low Income, Middle Income, High Income
- Example Data: ['Middle Income', 'High Income', 'Low Income', 'Middle Income', 'High Income']

Binary Categorical Variables

Approval Status:

- Categories: Approved, Not Approved
- Example Data: ['Approved', 'Not Approved', 'Approved', 'Approved', 'Not Approved']

Subscription Status:

- Categories: Subscribed, Not Subscribed
- Example Data: ['Subscribed', 'Not Subscribed', 'Subscribed', 'Not Subscribed', 'Subscribed']

Default Status:

- Categories: Defaulted, Not Defaulted
- Example Data: ['Not Defaulted', 'Defaulted', 'Not Defaulted', 'Not Defaulted', 'Defaulted']

Example 1: One-Hot Encoding (Dummy Variables)

Consider a dataset with a 'Color' column containing categorical data: 3 categories

ID	Color
1	Red
2	Blue
3	Green

One-hot encoding would create binary columns for each color

ID	Color_Red	Color_Blue	Color_Green
1	1	0	0
2	0	1	0
3	0	0	1

Example 2: Label Encoding

Consider a 'Size' column with categorical data

ID	Size
1	Small
2	Medium
3	Large

Label encoding assigns numerical labels to categories

D	Size
1	0
2	1
3	2

Example 3: Ordinal Encoding

Consider an 'Education Level' column:

ID	Education Level
1	High School
2	Bachelor's
3	Master's

Ordinal encoding assigns numerical values based on the order:

ID	Education Level
1	1
2	2
3	3

Example: Frequency Encoding

Consider a dataset with a 'City' column:

ID	City
1	New York
2	Tokyo
3	London
4	Tokyo
5	New York

Frequency encoding assigns values based on the frequency of each category in the dataset:

ID	City_Frequency
1	0.4
2	0.4
3	0.2
4	0.4
5	0.4

'New York' and 'Tokyo' both appear twice in the dataset, resulting in a frequency of 0.4 for each. 'London' appears once, resulting in a frequency of 0.2.

Interpretation

Higher Value:

A higher frequency-encoded value indicates that the category appears more frequently in the dataset.

Lower Value:

A lower value still indicates that the category is less common.

Target Encoding (Mean Encoding)

- Target encoding assigns values based on the mean of the target variable for each category

City	Target
New York	1
Tokyo	0
London	1
Tokyo	1
New York	0
London	0
Tokyo	1
New York	1
London	0
Tokyo	0

- For 'New York,' the mean of the target variable for rows with 'New York' is (1 + 0 + 1) / 3 = 0.67.
- For 'Tokyo,' the mean is (0 + 1 + 0 + 1) / 4 = 0.33.
- For 'London,' the mean is (1 + 0 + 0) / 3 = 0.5.

City	Target	City_Target_Encoded
New York	1	0.67
Tokyo	0	0.33
London	1	0.5
Tokyo	1	0.33
New York	0	0.67
London	0	0.5
Tokyo	1	0.33
New York	1	0.67
London	0	0.5
Tokyo	0	0.33

Pros of Target Encoding

Handles High Cardinality

Effective for dealing with high-cardinality categorical variables, where the number of unique categories is large.

Reduces Dimensionality

Unlike one-hot encoding, target encoding reduces dimensionality by representing each category with a single numerical value.

May Improve Model Performance

In situations where there is a strong correlation between categorical variables and the target, target encoding can enhance the predictive performance of some machine learning models.

Cons of Target Encoding

Risk of Overfitting:

- Target encoding may lead to overfitting, especially when applied to small datasets.

Impact of Target Imbalance:

- Target encoding can be influenced by imbalances in the target variable.

Demo using python/sklearn

(Encoding examples)

Thanks !!