


DATA PRE-PROCESSING

Converting or mapping raw data to another format, in order to prepare it for further analysis.

Also called DATA CLEANING or DATA WRANGLING

Missing Values

Data Formatting

Data Normalization (centering / scaling)

Data Binning (Numerical to Categorical)
Categorical to Numeric Variables

To make statistical modeling easier

For comparing groups of data

DEALING WITH MISSING VALUES

1. Try to find out the corresponding value, going to the source of the data

2. Remove the data where the missing value is found. → REDUCING THE DATA

3. Replacing data → INACCURACY RISKS

- Average
- Most common (categorical)
- Guessing based on knowledge from the gatherer

PYTHON LIBS FOR REMOVING → MISSING VALUES

`dropna` `axis=0` → remove rows

`axis=1` → remove columns

`inplace=True` → affects df

→ REPLACING VALUES

`mean = df["column"].mean()`

`df["column"].replace(np.nan, mean)`

DATA FORMATTING

Put in a standard of expression that will allow users make meaningful comparisons

- more clear
- easy to aggregate
- easy to compare

ex.

`df["column"] = ZSS(df["column"])`

`df.rename(columns={"column1": "new_name"},
 inplace=True)`

INCORRECT DATA TYPES

`df["column"] = df["column"].astype("int")`

DATA NORMALIZATION IN PYTHON

- Uniform the features value with different range.
- Can make some statistical analyses easier.
- Enables a fairer comparison.
- Avoids biasing the linear regression model weighing the larger feature more heavily.

* Simple feature scaling $(0, 1)$
 $\frac{\text{old}}{\text{old}/\text{max}}$

* Min - Max $\frac{\text{old} - \text{min}}{\text{max} - \text{min}}$ $(0, 1)$

* Z-score $\frac{\text{old} - \mu}{\sigma}$ $(-3, +3)$

ex.

$$\text{df}["\text{length}"] = \text{df}["\text{length}"] / \text{df}["\text{length}"].\text{max}()$$

DATA BINNING

- Grouping of values into "bins"
- Converts numerical into categorical variables
- Groups a set of numerical values into a set of "bins"
- Sometimes can improve accuracy of the predictive models

0, 1, ..., 100, ..., 1000
Low Med High

Example

```
bins = np.linspace(min(df["price"]),
                    max(df["price"]),
                    4)
group_names = {"Low", "Medium", "High"}  
df[["price_binned"]] = pd.cut(df["price"], bins  
, labels=group_names  
, include_lowest=True)
```

We can use histograms to visualize the distribution after they've been divided into bins.

CATEGORICAL → NUMERIC

- Most statistical models cannot take in the objects/strings as input

```
pd.get_dummies(df[["fuel"]])
```