

Data preprocessing, correlation and probability overview

Part 1: Missing values and data type transformations

By: Nouredin Sadawi, PhD
University of London

Missing values

- It is important to be familiar with the concept of missing values.
- This is a common problem in data science and machine learning.
- It refers to the problem of **having blank entries in a dataset**.
- This can happen for multiple reasons.
- It is important to deal with it **appropriately**.

Missing values

Pregnancies	Glucose	BloodPressure	SkinThickness
8	125.0	96	0
7	105.0	0	0
2	84.0	0	0
1	NaN	48	20
2	74.0	0	0
0	102.0	75	23
1	NaN	74	20
1	NaN	68	35
5	NaN	80	32
0	118.0	64	23
0	94.0	0	0
3	80.0	0	0
6	NaN	68	41
6	114.0	0	0
5	136.0	82	0
10	115.0	0	0

Why is it a problem?

- It adds ambiguity to the analysis.
- Imagine you would like to compute the average glucose level (see previous slide). It would be impossible to work it out accurately.
- Hence, an observation that has one or more missing values can be problematic.
- Any assumptions about the actual values of missing data are almost always unsound.
- It can happen in any data type (i.e. regardless of the data being numeric, categorical and so on).

How to deal with it

- The way to deal with, or handle, missing values depends on several factors (such as the problem domain, the percentage of missing data and others).
- There is no magic solution to this problem.
- One common method is to drop columns or rows that contain missing values (or sometimes if the column has a large fraction of its data missing).
- Instead of dropping rows or columns, sometimes we replace missing data with some value(s).
- This is known as missing value **imputation**.

Missing value imputation

- An important field in statistics.
- One method is to replace the missing values with a fixed value.
 - For example, sometimes people use 0 or any other number if the column is numeric.
 - It is common to replace missing values in a numeric column by the mean or median of existing values in that column.
 - If the column is categorical, sometimes people use the mode of existing entries in that column.
- Another method is to make educated guesses about the values of missing data using machine/statistical learning models.
- Or, sometimes, people use analysis methods that are designed specifically for analysing data with missing values.

Data type transformation

- It is common in data science and machine learning to transform data from one type to another.
- For example:
 - categorical to numeric.
 - numeric to categorical.
- A typical situation is when we would like to use a modelling technique that can only deal with numeric data to model or analyse categorical data (e.g. artificial neural networks).

Categorical to numeric

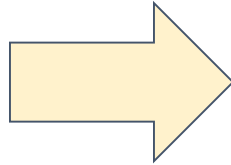
- This is known as **encoding**.
- A common technique is **binary encoding** (also known as **one-hot encoding**). Here is how it is done:
 1. If the categorical variable has n categories, then we create n new binary columns in the data.
 2. For each row, we set all values in those columns to 0 except the column that represents the categorical value in that row (we set it to 1).
 3. It is possible (and perhaps better) to create $n-1$ instead of n new binary columns (because one categorical value can be assumed when all values in the $n-1$ columns for the row are 0).

One-hot encoding example

City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3

One-hot encoding example

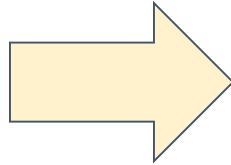
City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3



City	city_numeric	population
Tripoli	1	2
London	2	8
Tripoli	1	3
London	2	10
Sydney	3	3

One-hot encoding example

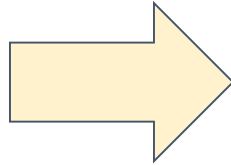
City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3



City	city_numeric	population
Tripoli	1	2
London	2	8
Tripoli	1	3
London	2	10
Sydney	3	3

One-hot encoding example

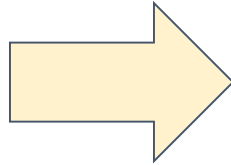
City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3



London	Sydney	Tripoli
0	0	1
1	0	0
0	0	1
1	0	0
0	1	0

One-hot encoding example

City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3

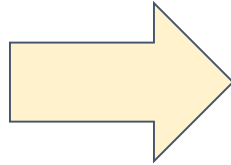


London	Sydney
0	0
1	0
0	0
1	0
0	1

Dummy encoding

One-hot encoding example

City	Population
Tripoli	2
London	8
Tripoli	3
London	10
Sydney	3



population	London	Sydney
2	0	0
8	1	0
3	0	0
10	1	0
3	0	1

Numeric to categorical

- This is known as **binning** or **discretisation**.
- It transforms numeric data into bins (or intervals).
- There are multiple ways of data binning:
 - a. Supervised: by involving the class (or target) column. An example is the entropy-based binning.
 - b. Unsupervised: by not involving the class (or target) column. Two examples are equal width binning and equal frequency binning.

Equal width binning

- Remember histogram?
- Here the numeric data is sorted and divided into n intervals (or bins). The width of each interval is:

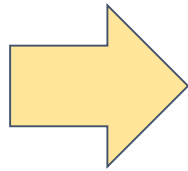
$$w = \frac{\max - \min}{n}$$

- In other words, the original variable values are represented in equal range bins regardless of the number of cases in each bin.

Example

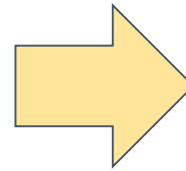
137.0
78.0
197.0
189.0
166.0
118.0
103.0
115.0
126.0
143.0
125.0
97.0
145.0
158.0
88.0
103.0
111.0
180.0
171.0

Extract
Bins



(77.881, 107.75]
(107.75, 137.5]
(137.5, 167.25]
(167.25, 197.0]

Transform
Data



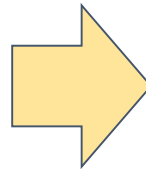
(107.75, 137.5]
(77.881, 107.75]
(167.25, 197.0]
(167.25, 197.0]
(137.75, 167.25]
(107.75, 137.5]
(77.881, 107.75]
(107.75, 137.5]
(107.75, 137.5]
(107.75, 137.5]
(137.75, 167.25]
(107.75, 137.5]
(77.881, 107.5]
(137.75, 167.25]
(137.75, 167.25]
(77.881, 107.75]
(77.881, 107.75]
(107.75, 137.5]
(167.25, 197.0]
(167.25, 197.0]

The parenthesis means the endpoint is not included and the square bracket means the endpoint is included.

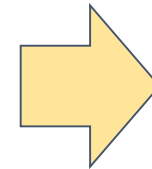
Equal frequency binning

- As the name suggests, this method works by dividing the data into n bins in such a way that all bins have approximately the same size (i.e. the same number of cases).

137.0
78.0
197.0
189.0
166.0
118.0
103.0
115.0
126.0
143.0
125.0
97.0
145.0
158.0
88.0
103.0
111.0
180.0
171.0



(77.999, 107.0]
(107.0, 126.0]
(126.0, 162.0]
(162.0, 197.0]



(126.0, 162.0]
(77.999, 107.0]
(162.0, 197.0]
(162.0, 197.0]
(162.0, 197.0]
(107.0, 126.0]
(77.999, 107.0]
(107.0, 126.0]
(107.0, 126.0]
(126.0, 162.0]
(107.0, 126.0]
(77.999, 107.0]
(126.0, 162.0]
(126.0, 162.0]
(77.999, 107.0]
(77.999, 107.0]
(107.0, 126.0]
(162.0, 197.0]
(162.0, 197.0]