Member-only story

Data Preprocessing with Python Pandas — Part 4 Standardization

Angelica Lo Duca · Follow
Published in Towards Data Science
3 min read · Dec 15, 2020

Listen Share More

Image by xresch from Pixabay

This tutorial explains how to preprocess data using the Pandas library. Preprocessing is the process of doing a pre-analysis of data, in order to transform

them into a standard and normalised format. Preprocessing involves the following aspects:

- missing values
- data formatting
- data normalisation
- data standardisation
- data binning

In this tutorial we deal only with standardization. In my previous tutorials I dealt with missing values, data formatting and data normalization.

You can download the source code of this tutorial as a Jupyter notebook from my Github Data Science Repository.

Standardization is often confused with normalization, however they refer to different things. Normalization involves adjusting values measured on different scales to a common scale, while standardization transforms data to have a mean of zero and a standard deviation of 1. Standardization is also done through a z-score transformation, where the new value is calculated as the difference between the current value and the average value, divided by the standard deviation.

Z-score is a statistical measure that specifies how far is a single data point from the rest of the dataset. As highlighted by Mahbubul Alam in his article, z-score can be used to detect outliers in a dataset.

Z-score can be calculated manually as described in my previous post. However, in this tutorial I will show you how to calculate z-score using some functions from the `scipy.stats` library.

In this tutorial we consider two types of standardizations:

- z-score
- z-map

Data Import

As example dataset, in this tutorial we consider the dataset provided by the Italian Protezione Civile, related to the number of COVID-19 cases registered since the beginning of the COVID-19 pandemic. The dataset is updated daily and can be downloaded from this link.

First of all, we need to import the Python `pandas` library and read the dataset through the `read_csv()` function. Then we can drop all the columns with `NaN` values. This is done through `dropna()` function.

```
import pandas as pd
df = pd.read_csv('https://raw.githubusercontent.com/pcm-dpc/COVID-19/master/dati-regioni/dpc-covid19-ita-regioni.csv')
df.dropna(axis=1,inplace=True)
df.tail(10)
```

	data	stato	codice_regione	denominazione_regione	lat	long	ricoverati_con_sintomi	terapia_intensiva	totale_ospedalizzati	isolamento
6206	2020-12-15T17:00:00	ITA	21	P.A. Bolzano	46.499335	11.356624	208	24	232	
6207	2020-12-15T17:00:00	ITA	22	P.A. Trento	46.068935	11.121231	412	53	465	
6208	2020-12-15T17:00:00	ITA	1	Piemonte	45.073274	7.680687	3761	266	4027	
6209	2020-12-15T17:00:00	ITA	16	Puglia	41.125596	16.867367	1530	187	1717	
6210	2020-12-15T17:00:00	ITA	20	Sardegna	39.215312	9.110616	570	58	628	
6211	2020-12-15T17:00:00	ITA	19	Sicilia	38.115697	13.362357	1225	185	1410	
6212	2020-12-15T17:00:00	ITA	9	Toscana	43.769231	11.255889	1156	214	1370	
6213	2020-12-15T17:00:00	ITA	10	Umbria	43.106758	12.388247	288	46	334	
6214	2020-12-15T17:00:00	ITA	2	Valle d'Aosta	45.737503	7.320149	74	6	80	
6215	2020-12-15T17:00:00	ITA	5	Veneto	45.434905	12.338452	2694	346	3040	

z-score

The new value is calculated as the difference between the current value and the average value, divided by the standard deviation. For example, we can calculate the z-score of the column `deceduti`. We can use the `zscore()` function of the `scipy.stats` library.

```
from scipy.stats import zscore
df['zscore-deceduti'] = zscore(df['deceduti'])
```

z-map

The new value is calculated as the difference between the current value and the average value of a comparison array, divided by the standard deviation of a comparison array. For example, we can calculate the z-map of the column `deceduti`, using the column `terapia_intensiva` as comparison array. We can use the `zmap()` function of the `scipy.stats` library.

```
from scipy.stats import zmap
zmap(df['deceduti'], df['terapia_intensiva'])
```

which gives the following output:

```
array([-0.42939174, -0.42939174, -0.42939174, ...,  3.47300249,
        2.16734162, 35.98322884])
```

Detect outliers

Standardization can be used to detect and delete outliers. For example, a threshold can be defined to specify which values can be considered as outliers. In this example, we set `threshold = 2`. We can add a new column to the dataframe, called `outliers` which is set to `True` if the value is less than `-2` or greater than `2`. We use the `numpy` function `where()` to perform comparisons.

```
threshold = 2
df['outliers'] = np.where((df['zscore-deceduti'] - threshold > 0),
True, np.where(df['zscore-deceduti'] + threshold < 0, True, False))
```

Now, we can remove outliers, using the `drop()` function.

```
df.drop(df[df['outliers'] == True].index,inplace=True)
```

Summary

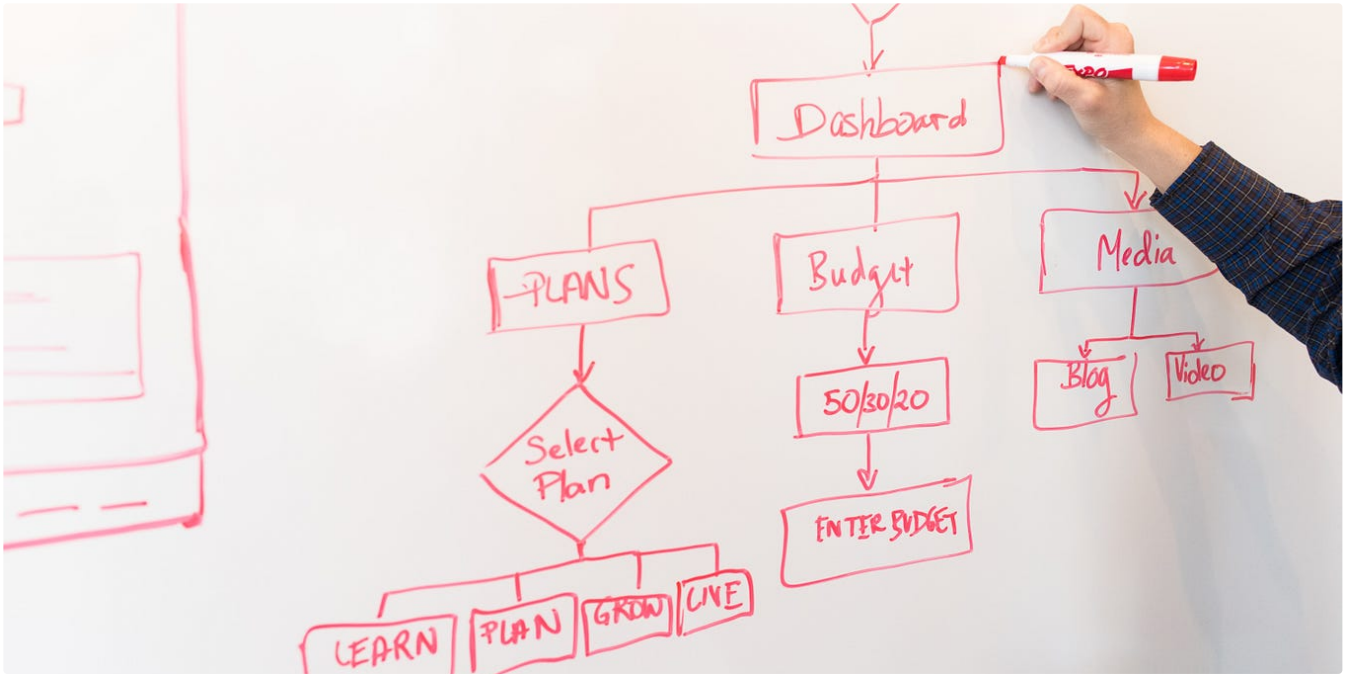
In this tutorial I have illustrated the difference between normalization and standardization. Normalization in some way includes standardization.

Two types of standardization exist: z-score and z-map.

Standardization can be used to detect and remove outliers from a dataset. In addition, it can be used to perform comparisons among different datasets.

If you wanted to learn about the last aspect of data preprocessing (i.e. data binning), stay tuned...

If you wanted to be updated on my research and other activities, you can follow me on Twitter, Youtube and and Github.



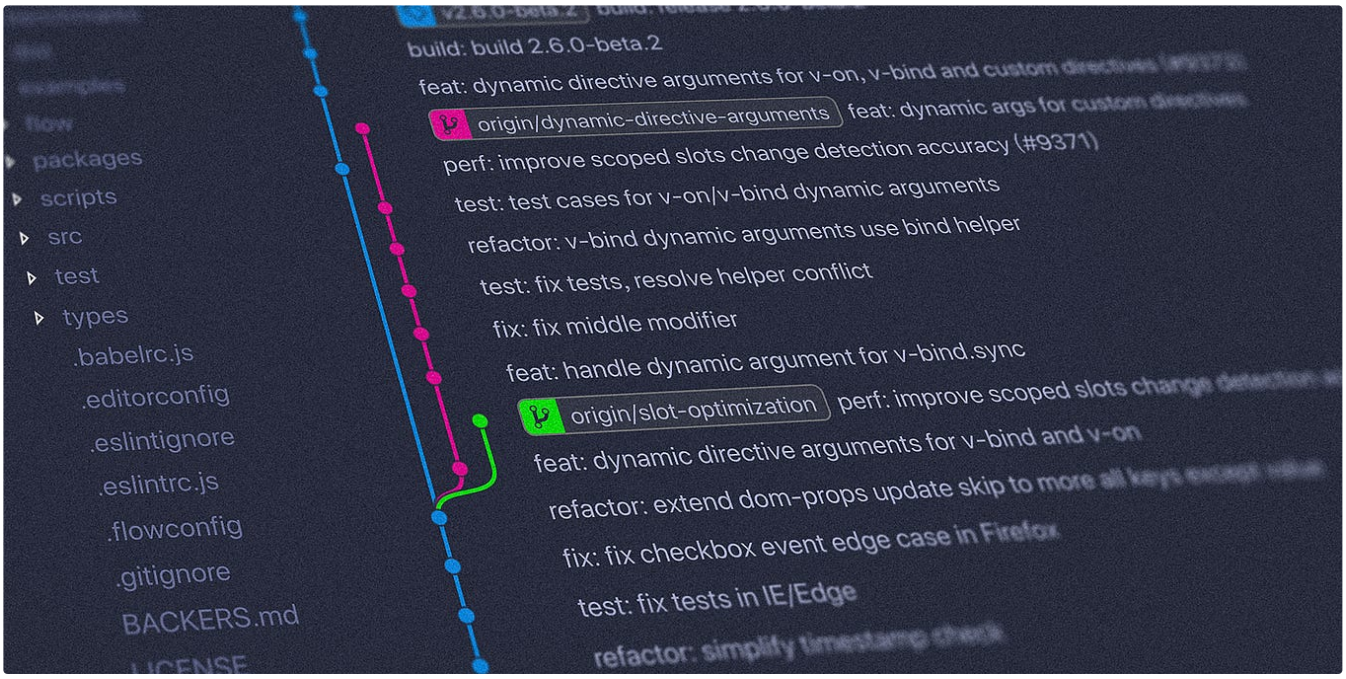
Angelica Lo Duca in Towards Data Science

How to Use ChatGPT to Generate Diagrams

A quick tutorial on how to write proper prompts to make ChatGPT generate diagrams

· 4 min read · May 29

192 3



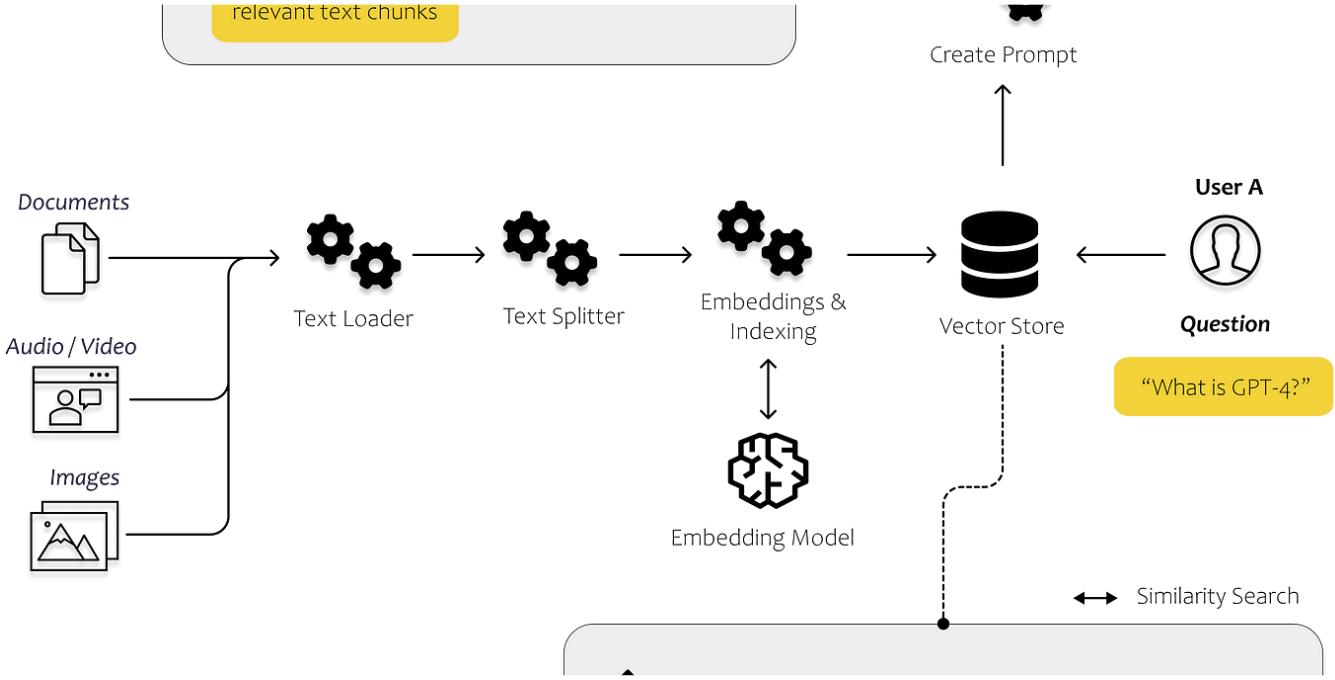
Miriam Santos in Towards Data Science

Pandas 2.0: A Game-Changer for Data Scientists?

The Top 5 Features for Efficient Data Manipulation

7 min read · Jun 27

1.8K 19



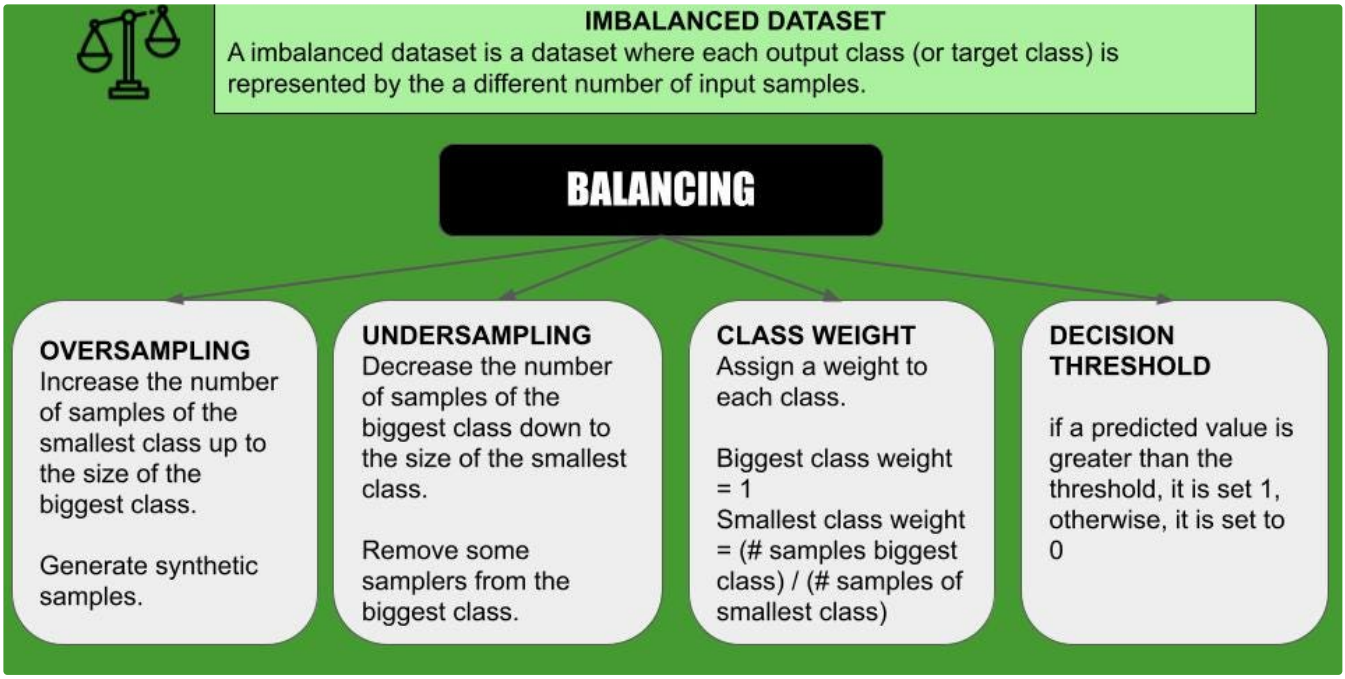
Dominik Polzer in Towards Data Science

All You Need to Know to Build Your First LLM App

A step-by-step tutorial to document loaders, embeddings, vector stores and prompt templates

· 26 min read · Jun 22

2.9K 26



Angelica Lo Duca in Towards Data Science

How to balance a dataset in Python

This tutorial belongs to the series How to improve the performance of a Machine Learning Algorithm. In this tutorial I deal with balancing...

· 9 min read · Mar 6, 2021