The probability of anomalous data has increased in today's data due to its large size and its origin for heterogenous sources. Considering the fact that high-quality data leads to better models and predictions, data preprocessing has become vital, and the fundamental step in the data science/machine learning/AI pipeline. In this article, learn about the need to process data and discuss different approaches to each step in the process.

While gathering data, you might come across three main factors that would contribute to the quality of data:

1. **Accuracy**: Erroneous values that deviate from the expected. The causes for inaccurate data can vary, but include:
    - Human/computer errors during data entry and transmission
    - Users deliberately submitting incorrect values (called disguised missing data)
    - Incorrect formats for input fields
    - Duplication of training examples

2. **Completeness**: Lacking attribute/feature values or values of interest. The data set might be incomplete due to:
    - Unavailability of data
    - Deletion of inconsistent data
    - Deletion of data deemed irrelevant initially

3. **Consistency**: Aggregation of data is inconsistent.

Some other features that also affect the data quality include timeliness (the data is incomplete until all relevant information is submitted after certain time periods), believability (how much the data is trusted by the user) and interpretability (how easily the data is understood by all stakeholders).

To ensure high-quality data, it's crucial to preprocess it. To make the process easier, data preprocessing is divided into four stages: data cleaning, data integration, data reduction, and data transformation.

Steps in Data Preprocessing

Step 1: Import the necessary libraries

```
In [*]: # importing libraries
        import pandas as pd
        import scipy
        import numpy as np
        from sklearn.preprocessing import MinMaxScaler
        import seaborn as sns
        import matplotlib.pyplot as plt
```

```
In [ ]:
```

Step 2: Load the dataset

```
In [2]: # Load the dataset
        df = pd.read_csv('D:\country_vaccinations.csv')
        print(df.head())
```

```
In [15]: print(df.isnull().sum())
country                              0
iso_code                             0
date                                 0
total_vaccinations                   0
people_vaccinated                    0
people_fully_vaccinated              0
daily_vaccinations_raw               0
daily_vaccinations                   0
total_vaccinations_per_hundred       0
people_vaccinated_per_hundred        0
people_fully_vaccinated_per_hundred  0
daily_vaccinations_per_million       0
vaccines                             0
source_name                          0
source_website                       0
dtype: int64
```

	country	iso_code	date	total_vaccinations	people_vaccinated
0	Afghanistan	AFG	22-02-2021	0	0
1	Afghanistan	AFG	23-02-2021	0	0
2	Afghanistan	AFG	24-02-2021	0	0
3	Afghanistan	AFG	25-02-2021	0	0
4	Afghanistan	AFG	26-02-2021	0	0

	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations
0	0	0	0
1	0	0	1367
2	0	0	1367
3	0	0	1367
4	0	0	1367

	total_vaccinations_per_hundred	people_vaccinated_per_hundred
0	0.0	0.0
1	0.0	0.0
2	0.0	0.0
3	0.0	0.0
4	0.0	0.0

	people_fully_vaccinated_per_hundred	daily_vaccinations_per_million
0	0.0	0
1	0.0	34
2	0.0	34
3	0.0	34
4	0.0	34

	vaccines
0	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
1	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
2	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
3	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...
4	Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi...

	source_name	source_website
0	World Health Organization	https://covid19.who.int/
1	World Health Organization	https://covid19.who.int/
2	World Health Organization	https://covid19.who.int/
3	World Health Organization	https://covid19.who.int/
4	World Health Organization	https://covid19.who.int/

```
In [3]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 86512 entries, 0 to 86511
Data columns (total 15 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   country                              86512 non-null  object
 1   iso_code                             86512 non-null  object
 2   date                                 86512 non-null  object
 3   total_vaccinations                   86512 non-null  int64
 4   people_vaccinated                    86512 non-null  int64
 5   people_fully_vaccinated              86512 non-null  int64
 6   daily_vaccinations_raw               86512 non-null  int64
 7   daily_vaccinations                   86512 non-null  int64
 8   total_vaccinations_per_hundred       86512 non-null  float64
 9   people_vaccinated_per_hundred        86512 non-null  float64
 10  people_fully_vaccinated_per_hundred  86512 non-null  float64
 11  daily_vaccinations_per_million       86512 non-null  int64
 12  vaccines                             86512 non-null  object
 13  source_name                          86512 non-null  object
 14  source_website                       86512 non-null  object
dtypes: float64(3), int64(6), object(6)
memory usage: 7.9+ MB
```

Step 3: Statistical Analysis

```
In [4]: df.describe()
```

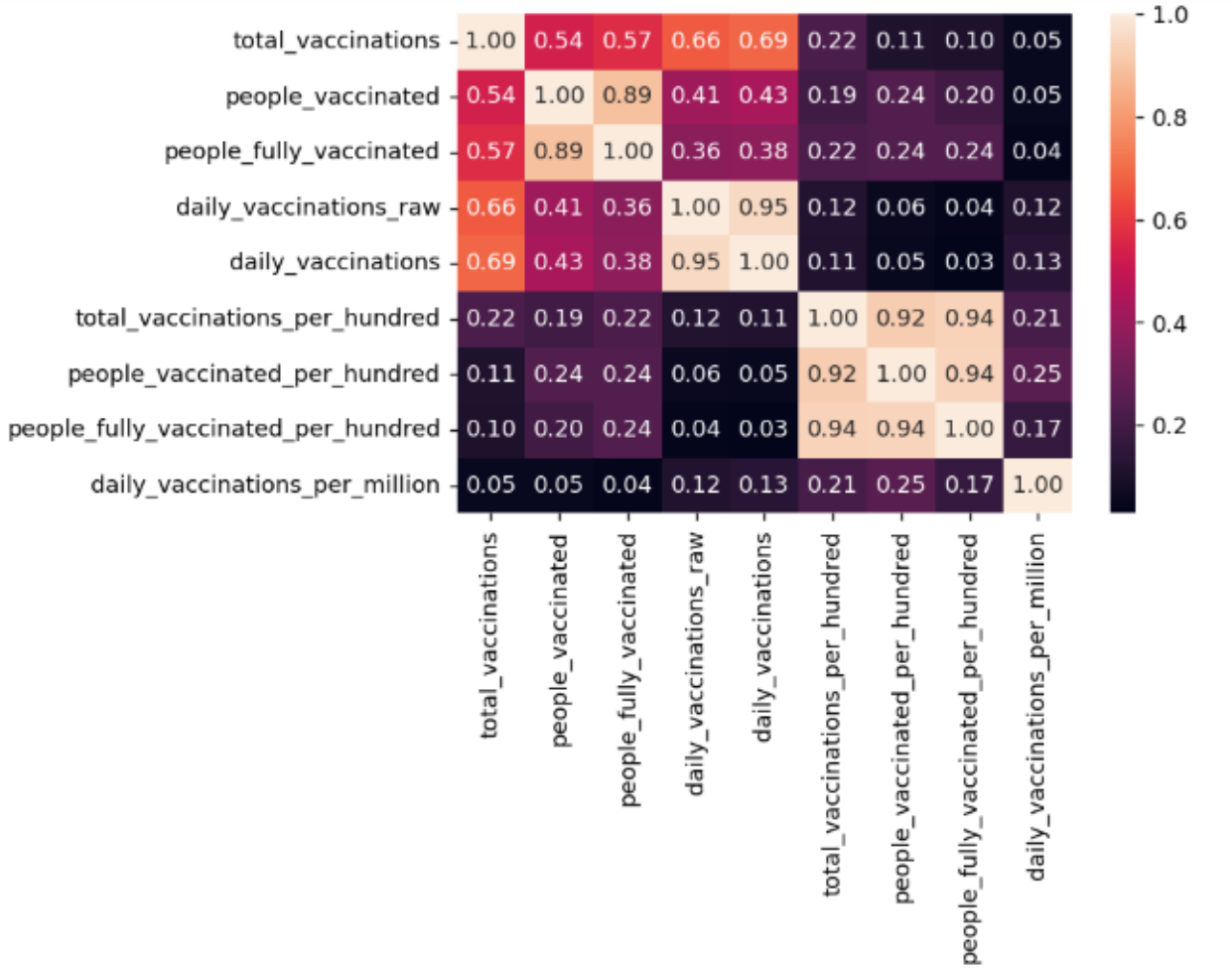
Out[4]:

	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations	total_vaccinations_per_hundred	people_vaccinat
count	8.651200e+04	8.651200e+04	8.651200e+04	8.651200e+04	8.651200e+04	86512.000000	
mean	2.315117e+07	8.451007e+06	6.341251e+06	1.106083e+05	1.308517e+05	40.419616	
std	1.611037e+08	4.969867e+07	3.890729e+07	7.864756e+05	7.669487e+05	62.707869	
min	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000	
25%	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	8.770000e+02	0.000000	
50%	1.008000e+03	0.000000e+00	0.000000e+00	0.000000e+00	7.245000e+03	0.010000	
75%	3.697554e+06	1.843103e+06	1.137869e+06	1.280625e+04	4.370450e+04	68.750000	
max	3.263129e+09	1.275541e+09	1.240777e+09	2.474100e+07	2.242429e+07	345.370000	

Step 4: Correlations

In [6]:
```
#correlation
corr = df.corr()

plt.figure(dpi=130)
sns.heatmap(df.corr(), annot=True, fmt= '.2f')
plt.show()
```



In [9]:
```
corr['total_vaccinations'].sort_values(ascending = False)
```

Out[9]:
```
total_vaccinations                     1.000000
daily_vaccinations                     0.688296
daily_vaccinations_raw                 0.662729
people_fully_vaccinated                0.571087
people_vaccinated                      0.535036
total_vaccinations_per_hundred         0.222264
people_vaccinated_per_hundred          0.106979
people_fully_vaccinated_per_hundred    0.104074
daily_vaccinations_per_million         0.050911
Name: total_vaccinations, dtype: float64
```