Intro to Data Science - Data Cleaning - 01

March 26, 2020

[ ]:
```python
# This Python 3 environment comes with many helpful analytics libraries␣
↪installed
# It is defined by the kaggle/python docker image: https://github.com/kaggle/
↪docker-python
# For example, here's several helpful packages to load in

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the "../input/" directory.
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
↪all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# Any results you write to the current directory are saved as output.
```

0.1 ### Data Cleaning

Most of the dataset are not cleaned and usually have messy data so we must clean the data before use that.

Data cleaning involves dealing with

- Missing data
- Duplicated data
- Outliers in the data
- Extra data that might not be needed
- Inconsistent data
- Converting data into a standard format so that it is easy to work on

0.2 ### Missing Value

During data collection and entry it is possible that some values are missed or data was not available. Pandas writes `NaN(Not a number)` when it found a missing value. It does not include NaN in any calculation like sum, mean etc.

we can detect missing value by using `isnull` function. whenever it found a missing value, return `True` otherwisde `False`.

```
[ ]: df = pd.read_csv('/kaggle/input/california-housing-prices/housing.csv')

     print('missing value in every column :', df.isnull().sum())
     print('total missing value of all the columns : ', df.isnull().sum().sum())

     #show all the rows of missing value
     display(df[df['total_bedrooms'].isnull()])
```

0.3 ### Filtering missing values

when encounter missing value we can either remove those values or fill with suitable value

```
[ ]: print('shape of original dataframe : ', df.shape)

     # remove missing rows
     new_df = df.dropna()
     print('shape of filtered dataframe :', new_df.shape)

     # remove the whole column (bad idea)
     new_df = df.dropna(axis=1)
     print('shape of filtered dataframe : ', new_df.shape)
```

0.4 ### Filling numerical missing values

Removing missing missing value is bad idea, the best way to deal with missing value is filling those missing value with suitable value. The strategy behind filling missing values can vary from problem to problem. In some cases, filling in with the **mean** or **median** value works very well, while in some cases **interpolation** works better. Interpolation is a mathematical technique of constructing new points from a range of points.

```
[ ]: print("Total missing values : ",df.isnull().sum().sum())

     #fill value with median value
     value = df['total_bedrooms'].median()
     new_df = df['total_bedrooms'].fillna(value)

     #fill missing value with interpolation
     new_df = df.interpolate('linear')

     print("Total missing values : ",new_df.isnull().sum().sum())

     #print(new_df.loc[290, 'total_bedrooms'])
```

For non-numerical missing values, we can simply write **missing** because we cannot use mathematical methods like mean, median, or interpolation to estimate these.

0.5 ### Duplicates Data

we may have some duplicate data that we need remove. we can use `dataframe.duplicated(subset=list of column names)` to find duplicate data and `drop_duplicates(subset=list of columns names)` to drop duplicate data from dataframe.