Dealing with NaN hands on codes

. . . by swarnadeep

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
# Creating a list of Python dictionaries
items2 = [{'bikes': 20, 'pants': 30, 'watches': 35, 'shirts': 15, 'shoes':8, 'suits':45},
{'watches': 10, 'glasses': 50, 'bikes': 15, 'pants':5, 'shirts': 2, 'shoes':5, 'suits':7},
{'bikes': 20, 'pants': 30, 'watches': 35, 'glasses': 4, 'shoes':10}]
```

In [3]:
```
# Creating a DataFrame  and providing the row index
store_items = pd.DataFrame(items2, index = ['store 1', 'store 2', 'store 3'])
store_items
```

Out[3]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	20	30	35	15.0	8	45.0	NaN
store 2	15	5	10	2.0	5	7.0	50.0
store 3	20	30	35	NaN	10	NaN	4.0

Counting `NaN` and `non-NaN` values in a DataFrame

`.isnull()` : returns a Boolean DataFrame of the same size as `store_items`

In [4]:
```
store_items.isnull()
```

Out[4]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	False	False	False	False	False	False	True
store 2	False	False	False	False	False	False	False
store 3	False	False	False	True	False	True	False

`.isnull().sum()` : counts NaN values from each columns

In [5]:
```
store_items.isnull().sum()
```

Out[5]:
```
bikes      0
pants      0
watches    0
shirts     1
shoes      0
suits      1
glasses    1
dtype: int64
```

`.isnull().sum()` : counts total NaN values of the DataFrame

In [6]:
```
store_items.isnull().sum().sum()
```

Out[6]:
```
3
```

`.count()` : counts non-NaN values from each columns

In [7]:
```
store_items.count()
```

Out[7]:
```
bikes      3
pants      3
watches    3
shirts     2
shoes      3
suits      2
glasses    2
dtype: int64
```

Deleting any Row / Column with `NaN` Values

`.dropna(axis = 0 or 1)` : Drops any Rows or Columns with NaN value

In [8]:
```
# Drop any Rows with NaN value
store_items.dropna(axis = 0)
```

Out[8]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 2	15	5	10	2.0	5	7.0	50.0

In [9]:
```
# Drop any Columns with NaN value
store_items.dropna(axis = 1)
```

Out[9]:

	bikes	pants	watches	shoes
store 1	20	30	35	8
store 2	15	5	10	5
store 3	20	30	35	10

We must use `inplace = True` in `.dropna()` to update in DataFrame permanently

Replacing NaN values

`.fillna(value)` : Replace all NaN values with the value given

In [10]:
```
store_items.fillna(0)
```

Out[10]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	20	30	35	15.0	8	45.0	0.0
store 2	15	5	10	2.0	5	7.0	50.0
store 3	20	30	35	0.0	10	0.0	4.0

`.fillna(method = 'ffill', axis)` : use forward filling (ffill) method to replace NaN values on given axis

In [11]:
```
# Replacing NaN values with the previous value in the column
store_items.fillna(method = 'ffill', axis = 0)
```

Out[11]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	20	30	35	15.0	8	45.0	NaN
store 2	15	5	10	2.0	5	7.0	50.0
store 3	20	30	35	2.0	10	7.0	4.0

`.fillna(method = 'backfill', axis)` : use backward filling (backfill) method to replace NaN values on given axis

In [12]:
```
# Replacing NaN values with the next value in the column
store_items.fillna(method = 'backfill', axis = 0)
```

Out[12]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	20	30	35	15.0	8	45.0	50.0
store 2	15	5	10	2.0	5	7.0	50.0
store 3	20	30	35	NaN	10	NaN	4.0

`.interpolate(method = 'linear', axis)` : Use `linear` interpolation to replace NaN values using the values along the given axis

In [13]:
```
store_items.interpolate(method = 'linear', axis = 0)
```

Out[13]:

	bikes	pants	watches	shirts	shoes	suits	glasses
store 1	20	30	35	15.0	8	45.0	NaN
store 2	15	5	10	2.0	5	7.0	50.0
store 3	20	30	35	2.0	10	7.0	4.0

In [ ]: