Chapter 2 - Data Preparation Basics

Segment 2 - Treating missing values

```
import numpy as np
import pandas as pd

from pandas import Series, DataFrame
```

Figuring out what data is missing

```
missing = np.nan

series_obj = Series(['row 1', 'row 2', missing, 'row 4', 'row 5', 'row 6', missing, 'row 8']) # Series in pandas are like numpy arrays
series_obj
```

```
0    row 1
1    row 2
2      NaN
3    row 4
4    row 5
5    row 6
6      NaN
7    row 8
dtype: object
```

```
series_obj.isna() #isna() or isnull(), both can be used
```

```
0    False
1    False
2     True
3    False
4    False
5    False
6     True
7    False
dtype: bool
```

Filling in for missing values

```
np.random.seed(25)
DF_obj = DataFrame(np.random.rand(36).reshape(6,6))
DF_obj
#a=np.random.rand(6)
#print(a)
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	0.113041
2	0.447031	0.585445	0.161985	0.520719	0.326051	0.699186
3	0.366395	0.836375	0.481343	0.516502	0.383048	0.997541
4	0.514244	0.559053	0.034450	0.719930	0.421004	0.436935
5	0.281701	0.900274	0.669612	0.456069	0.289804	0.525819

```
DF_obj.loc[3:5, 0] = missing
#DF_obj[3:5] = missing
DF_obj
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	NaN
1	0.684969	0.437611	0.556229	0.367080	0.402366	NaN
2	0.447031	0.585445	0.161985	0.520719	0.326051	NaN
3	NaN	NaN	NaN	NaN	NaN	NaN
4	NaN	NaN	NaN	NaN	NaN	NaN
5	NaN	0.900274	0.669612	0.456069	0.289804	NaN

```
filled_DF = DF_obj.fillna(0)
filled_DF
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.0
1	0.684969	0.437611	0.556229	0.367080	0.402366	0.0
2	0.447031	0.585445	0.161985	0.520719	0.326051	0.0
3	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
4	0.000000	0.000000	0.000000	0.000000	0.000000	0.0
5	0.000000	0.900274	0.669612	0.456069	0.289804	0.0

```
filled_DF = DF_obj.fillna({0: 0.1, 5:1.25}) #under {} braces, the null values are filled at specified columns with provided values { 0
filled_DF
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	1.250000
2	0.447031	0.585445	0.161985	0.520719	0.326051	1.250000
3	0.100000	0.836375	0.481343	0.516502	0.383048	1.250000
4	0.100000	0.559053	0.034450	0.719930	0.421004	1.250000
5	0.100000	0.900274	0.669612	0.456069	0.289804	0.525819

```
fill_DF = DF_obj.fillna(method='ffill') #fill the null with values which are above them
fill_DF
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	NaN

```
fiilingdf= DF_obj.fillna(DF_obj.mean())
fiilingdf
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	NaN
1	0.684969	0.437611	0.556229	0.367080	0.402366	NaN
2	0.447031	0.585445	0.161985	0.520719	0.326051	NaN
3	0.667375	0.626402	0.416666	0.382445	0.357330	NaN
4	0.667375	0.626402	0.416666	0.382445	0.357330	NaN
5	0.667375	0.900274	0.669612	0.456069	0.289804	NaN

Counting missing values

```
np.random.seed(25)
DF_obj = DataFrame(np.random.rand(36).reshape(6,6))
DF_obj.loc[3:5, 0] = missing
DF_obj.loc[1:4, 5] = missing
DF_obj
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.411100	0.117376
1	0.684969	0.437611	0.556229	0.367080	0.402366	NaN
2	0.447031	0.585445	0.161985	0.520719	0.326051	NaN
3	NaN	0.836375	0.481343	0.516502	0.383048	NaN
4	NaN	0.559053	0.034450	0.719930	0.421004	NaN
5	NaN	0.900274	0.669612	0.456069	0.289804	0.525819

```
DF_obj.isnull().sum()
```

```
0    3
1    0
2    0
3    0
4    0
5    4
dtype: int64
```

Filtering out missing values

```
DF_no_NaN = DF_obj.dropna() #drops rows, axis=0 default
DF_no_NaN
```

	0	1	2	3	4	5
0	0.870124	0.582277	0.278839	0.185911	0.4111	0.117376

```
DF_no_NaN = DF_obj.dropna(axis=1) #drops col by specifying axis=1, na at axis=1 dropped
DF_no_NaN
```

	1	2	3	4
0	0.582277	0.278839	0.185911	0.411100
1	0.437611	0.556229	0.367080	0.402366
2	0.585445	0.161985	0.520719	0.326051
3	0.836375	0.481343	0.516502	0.383048
4	0.559053	0.034450	0.719930	0.421004
5	0.900274	0.669612	0.456069	0.289804

Colab paid products - Cancel contracts here