Tutorial 05 - Handle Missing Data fillna, dropna, interpolate

In [1]:
```
import pandas as pd
df = pd.read_csv('sample_data_tutorial_05.csv')
df
```

Out[1]:

	day	temperature	windspeed	event
0	1/1/2017	32.0	6.0	Rain
1	1/4/2017	NaN	9.0	Sunny
2	1/5/2017	28.0	NaN	Snow
3	1/6/2017	NaN	7.0	NaN
4	1/7/2017	32.0	NaN	Rain
5	1/8/2017	NaN	NaN	Sunny
6	1/9/2017	NaN	NaN	NaN
7	1/10/2017	34.0	8.0	Cloudy
8	1/11/2017	40.0	12.0	Sunny

In [2]:
```
type(df.day[0])
```

Out[2]:
```
str
```

In [3]:
```
# Observe que no primeiro df a primeira coluna é uma "string" e com o comando abaix
o um "Timestamp"
df = pd.read_csv('sample_data_tutorial_05.csv', parse_dates=["day"])
type(df.day[0])
```

Out[3]:
```
pandas._libs.tslibs.timestamps.Timestamp
```

In [4]:
```
# Inserindo a coluna 'day' como índice:
df.set_index('day', inplace=True)
df
```

Out[4]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	NaN	9.0	Sunny
2017-01-05	28.0	NaN	Snow
2017-01-06	NaN	7.0	NaN
2017-01-07	32.0	NaN	Rain
2017-01-08	NaN	NaN	Sunny
2017-01-09	NaN	NaN	NaN
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [5]:
```
# Trocando os "NaN" por zero
newdf = df.fillna(0)
newdf
```

Out[5]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	0.0	9.0	Sunny
2017-01-05	28.0	0.0	Snow
2017-01-06	0.0	7.0	0
2017-01-07	32.0	0.0	Rain
2017-01-08	0.0	0.0	Sunny
2017-01-09	0.0	0.0	0
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [6]:
```
# Observe que para 'event' o número 0 não faz muito sentido portanto iremos fazer o
 'fillna' com um dicionário:
newdf = df.fillna({
    'temperature': 0,
    'windspeed': 0,
    'event': 'no event'
})
newdf
```

Out[6]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	0.0	9.0	Sunny
2017-01-05	28.0	0.0	Snow
2017-01-06	0.0	7.0	no event
2017-01-07	32.0	0.0	Rain
2017-01-08	0.0	0.0	Sunny
2017-01-09	0.0	0.0	no event
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [7]:
```
# Substitur por zero pode alterar cálculos estastísticos portanto iremos fazer outr
as alterações:
# ffill = forward fill, faz o prenchimento para frente (pega valor da célula de trá
s)
# bfill = backward fill, faz o prenchimento para trás (pega valor da célula da fren
te)
# Google "pandas fillna"
newdf = df.fillna(method='bfill')
newdf
```

Out[7]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	28.0	9.0	Sunny
2017-01-05	28.0	7.0	Snow
2017-01-06	32.0	7.0	Rain
2017-01-07	32.0	8.0	Rain
2017-01-08	34.0	8.0	Sunny
2017-01-09	34.0	8.0	Cloudy
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [8]:
```
# Se usarmos a opção axis="columns" ele irá copiar da coluna
newdf1 = df.fillna(method='bfill', axis="columns")
newdf1
```

Out[8]:

	temperature	windspeed	event
day			
2017-01-01	32	6	Rain
2017-01-04	9	9	Sunny
2017-01-05	28	Snow	Snow
2017-01-06	7	7	NaN
2017-01-07	32	Rain	Rain
2017-01-08	Sunny	Sunny	Sunny
2017-01-09	NaN	NaN	NaN
2017-01-10	34	8	Cloudy
2017-01-11	40	12	Sunny

In [9]:
```
# Se usarmos a opção limit=1 a cópia será limitada ao número imposto
newdf2 = df.fillna(method='bfill', limit=1)
newdf2
```

Out[9]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	28.0	9.0	Sunny
2017-01-05	28.0	7.0	Snow
2017-01-06	32.0	7.0	Rain
2017-01-07	32.0	NaN	Rain
2017-01-08	NaN	NaN	Sunny
2017-01-09	34.0	8.0	Cloudy
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [10]:
```
# O comando interpolate sem mais argumentos considera linear e espaçamentos iguais
newdf = df.interpolate()
newdf
```

Out[10]:

	temperature	windspeed	event
day			
2017-01-01	32.000000	6.00	Rain
2017-01-04	30.000000	9.00	Sunny
2017-01-05	28.000000	8.00	Snow
2017-01-06	30.000000	7.00	NaN
2017-01-07	32.000000	7.25	Rain
2017-01-08	32.666667	7.50	Sunny
2017-01-09	33.333333	7.75	NaN
2017-01-10	34.000000	8.00	Cloudy
2017-01-11	40.000000	12.00	Sunny

In [11]:
```
# A opção 'time' levará em consideração o tempo que está no índice. Observe que do
 dia 1 pula-se ao 4.
newdf = df.interpolate(method='time')
newdf
```

Out[11]:

	temperature	windspeed	event
day			
2017-01-01	32.000000	6.00	Rain
2017-01-04	29.000000	9.00	Sunny
2017-01-05	28.000000	8.00	Snow
2017-01-06	30.000000	7.00	NaN
2017-01-07	32.000000	7.25	Rain
2017-01-08	32.666667	7.50	Sunny
2017-01-09	33.333333	7.75	NaN
2017-01-10	34.000000	8.00	Cloudy
2017-01-11	40.000000	12.00	Sunny

In [12]:
```
# dropna irá remover a células que contém NaN
newdf = df.dropna()
newdf
```

Out[12]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [13]:
```
# Caso se queira remover somente quando todas as colunas contém NaN
newdf = df.dropna(how='all')
newdf
```

Out[13]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	NaN	9.0	Sunny
2017-01-05	28.0	NaN	Snow
2017-01-06	NaN	7.0	NaN
2017-01-07	32.0	NaN	Rain
2017-01-08	NaN	NaN	Sunny
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [14]:
```
# O comando thresh (threshold) define o número de valores válidos para não ser remo
vido
newdf = df.dropna(thresh=2)
newdf # Observe que irá remover os dias 06, 08 e 09 que contém somente 1 valor váli
do ou zero (nenhum).
```

Out[14]:

	temperature	windspeed	event
day			
2017-01-01	32.0	6.0	Rain
2017-01-04	NaN	9.0	Sunny
2017-01-05	28.0	NaN	Snow
2017-01-07	32.0	NaN	Rain
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [15]:
```
# Veja que faltam os dias 02 e 03
dt = pd.date_range('01-01-2017','01-11-2017')
idx = pd.DatetimeIndex(dt)
df = df.reindex(idx)
df
```

Out[15]:

	temperature	windspeed	event
2017-01-01	32.0	6.0	Rain
2017-01-02	NaN	NaN	NaN
2017-01-03	NaN	NaN	NaN
2017-01-04	NaN	9.0	Sunny
2017-01-05	28.0	NaN	Snow
2017-01-06	NaN	7.0	NaN
2017-01-07	32.0	NaN	Rain
2017-01-08	NaN	NaN	Sunny
2017-01-09	NaN	NaN	NaN
2017-01-10	34.0	8.0	Cloudy
2017-01-11	40.0	12.0	Sunny

In [16]:
```
newdf = df.interpolate(method='time')
newdf
```

Out[16]:

	temperature	windspeed	event
2017-01-01	32.000000	6.00	Rain
2017-01-02	31.000000	7.00	NaN
2017-01-03	30.000000	8.00	NaN
2017-01-04	29.000000	9.00	Sunny
2017-01-05	28.000000	8.00	Snow
2017-01-06	30.000000	7.00	NaN
2017-01-07	32.000000	7.25	Rain
2017-01-08	32.666667	7.50	Sunny
2017-01-09	33.333333	7.75	NaN
2017-01-10	34.000000	8.00	Cloudy
2017-01-11	40.000000	12.00	Sunny