day01_SurojitSengupta_Q1US Birthrate Data

August 6, 2018

0.1 Day 1

0.2 Birthrate data of United States, provided by the Centers for Disease Control (CDC)

```
In [ ]: #Summary: The given dataset,
        # presents various datapoints related to the number of births of 2 Genders on,
        # various days/months over a span of 40 years.
        # The analysis continued below presents
        # the number of years,surveyed, the years and days with the maximum births,
        # the total number of births by each year after dropping data noise.
```

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

```
In [2]: df=pd.read_excel('Birthrate.xlsx')
```

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

```
Out[3]:    year  month  day gender  births
        0  1969      1  1.0      F    4046
        1  1969      1  1.0      M    4440
        2  1969      1  2.0      F    4454
        3  1969      1  2.0      M    4548
        4  1969      1  3.0      F    4548
```

```
In [14]: #Find the list of years surveyed
         df['year'].unique()
         #OR
         df['year'].unique().tolist()
```

```
Out[14]: [1969,
          1970,
          1971,
          1972,
          1973,
          1974,
          1975,
          1976,
          1977,
```

```
          1978,
          1979,
          1980,
          1981,
          1982,
          1983,
          1984,
          1985,
          1986,
          1987,
          1988,
          1989,
          1990,
          1991,
          1992,
          1993,
          1994,
          1995,
          1996,
          1997,
          1998,
          1999,
          2000,
          2001,
          2002,
          2003,
          2004,
          2005,
          2006,
          2007,
          2008]
```

```
In [21]: #Find the number of years surveyed. Please note, the unique() function returns a numpy
         import numpy as np
         ar=df['year'].unique()
         print(ar.size)
         #OR
         print(len(df['year'].unique().tolist()))
```

```
40
40
```

```
In [22]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15547 entries, 0 to 15546
Data columns (total 5 columns):
year      15547 non-null int64
```

```
month     15547 non-null int64
day       15067 non-null float64
gender    15547 non-null object
births    15547 non-null int64
dtypes: float64(1), int64(3), object(1)
memory usage: 607.4+ KB
```

```
In [40]: df.loc[df['day']=='null']
```

```
/opt/usr/anaconda3/lib/python3.6/site-packages/pandas/core/ops.py:816: FutureWarning: elementwis
  result = getattr(x, name)(y)
```

```
---------------------------------------------------------------------------

TypeError                                 Traceback (most recent call last)

<ipython-input-40-a35162b61d6f> in <module>()
----> 1 df.loc[df['day']=='null']


/opt/usr/anaconda3/lib/python3.6/site-packages/pandas/core/ops.py in wrapper(self, other
877
878             with np.errstate(all='ignore'):
--> 879                 res = na_op(values, other)
880             if is_scalar(res):
881                 raise TypeError('Could not compare {typ} type with Series'


/opt/usr/anaconda3/lib/python3.6/site-packages/pandas/core/ops.py in na_op(x, y)
816                     result = getattr(x, name)(y)
817                 if result is NotImplemented:
--> 818                     raise TypeError("invalid type comparison")
819             except AttributeError:
820                 result = op(x, y)


TypeError: invalid type comparison
```

```
In [29]: df.tail()
```

Out[29]:

	year	month	day	gender	births
15542	2008	10	NaN	M	183219
15543	2008	11	NaN	F	158939
15544	2008	11	NaN	M	165468
15545	2008	12	NaN	F	173215
15546	2008	12	NaN	M	181235

```
In [35]: df[df['day'].isnull()]
```

Out[35]:

	year	month	day	gender	births
15067	1989	1	NaN	F	156749
15068	1989	1	NaN	M	164052
15069	1989	2	NaN	F	146710
15070	1989	2	NaN	M	154047
15071	1989	3	NaN	F	165889
15072	1989	3	NaN	M	174433
15073	1989	4	NaN	F	155689
15074	1989	4	NaN	M	163432
15075	1989	5	NaN	F	163800
15076	1989	5	NaN	M	172892
15077	1989	6	NaN	F	165525
15078	1989	6	NaN	M	173823
15079	1989	7	NaN	F	174054
15080	1989	7	NaN	M	183063
15081	1989	8	NaN	F	178986
15082	1989	8	NaN	M	188074
15083	1989	9	NaN	F	174808
15084	1989	9	NaN	M	182962
15085	1989	10	NaN	F	168303
15086	1989	10	NaN	M	176258
15087	1989	11	NaN	F	159013
15088	1989	11	NaN	M	166923
15089	1989	12	NaN	F	164186
15090	1989	12	NaN	M	172022
15091	1990	1	NaN	F	163576
15092	1990	1	NaN	M	172073
15093	1990	2	NaN	F	153015
15094	1990	2	NaN	M	159915
15095	1990	3	NaN	F	171463
15096	1990	3	NaN	M	179499
...	...	...	...	...	...
15517	2007	10	NaN	F	180912
15518	2007	10	NaN	M	189157
15519	2007	11	NaN	F	173513
15520	2007	11	NaN	M	180814
15521	2007	12	NaN	F	173787
15522	2007	12	NaN	M	181426
15523	2008	1	NaN	F	174255
15524	2008	1	NaN	M	182789
15525	2008	2	NaN	F	165669
15526	2008	2	NaN	M	173434
15527	2008	3	NaN	F	172053
15528	2008	3	NaN	M	179129
15529	2008	4	NaN	F	169585
15530	2008	4	NaN	M	177399

```
15531  2008      5  NaN      F  173141
15532  2008      5  NaN      M  182294
15533  2008      6  NaN      F  169958
15534  2008      6  NaN      M  179267
15535  2008      7  NaN      F  183391
15536  2008      7  NaN      M  192714
15537  2008      8  NaN      F  182713
15538  2008      8  NaN      M  191315
15539  2008      9  NaN      F  179696
15540  2008      9  NaN      M  188964
15541  2008     10  NaN      F  175314
15542  2008     10  NaN      M  183219
15543  2008     11  NaN      F  158939
15544  2008     11  NaN      M  165468
15545  2008     12  NaN      F  173215
15546  2008     12  NaN      M  181235

[480 rows x 5 columns]
```

```
In [44]: #Record with max birth count
         df[df['births']==df['births'].max()]
```

```
Out[44]:        year  month  day gender  births
         15514  2007      8  NaN      M  199622
```

```
In [45]: #Total births by months
```

```
In [46]: df_new=df.dropna()
```

```
In [47]: df_new.tail()
```

```
Out[47]:        year  month   day gender  births
         15062  1988     12  29.0      M    5944
         15063  1988     12  30.0      F    5742
         15064  1988     12  30.0      M    6095
         15065  1988     12  31.0      F    4435
         15066  1988     12  31.0      M    4698
```

```
In [48]: df.set_index('month')
```

```
Out[48]:        year   day gender  births
         month
         1      1969   1.0      F    4046
         1      1969   1.0      M    4440
         1      1969   2.0      F    4454
         1      1969   2.0      M    4548
         1      1969   3.0      F    4548
         1      1969   3.0      M    4994
         1      1969   4.0      F    4440
```

1	1969	4.0	M	4520
1	1969	5.0	F	4192
1	1969	5.0	M	4198
1	1969	6.0	F	4710
1	1969	6.0	M	4850
1	1969	7.0	F	4646
1	1969	7.0	M	5092
1	1969	8.0	F	4800
1	1969	8.0	M	4934
1	1969	9.0	F	4592
1	1969	9.0	M	4842
1	1969	10.0	F	4852
1	1969	10.0	M	5190
1	1969	11.0	F	4580
1	1969	11.0	M	4598
1	1969	12.0	F	4126
1	1969	12.0	M	4324
1	1969	13.0	F	4758
1	1969	13.0	M	5076
1	1969	14.0	F	5070
1	1969	14.0	M	5296
1	1969	15.0	F	4798
1	1969	15.0	M	5096
...	...	...	...	...
10	2007	NaN	F	180912
10	2007	NaN	M	189157
11	2007	NaN	F	173513
11	2007	NaN	M	180814
12	2007	NaN	F	173787
12	2007	NaN	M	181426
1	2008	NaN	F	174255
1	2008	NaN	M	182789
2	2008	NaN	F	165669
2	2008	NaN	M	173434
3	2008	NaN	F	172053
3	2008	NaN	M	179129
4	2008	NaN	F	169585
4	2008	NaN	M	177399
5	2008	NaN	F	173141
5	2008	NaN	M	182294
6	2008	NaN	F	169958
6	2008	NaN	M	179267
7	2008	NaN	F	183391
7	2008	NaN	M	192714
8	2008	NaN	F	182713
8	2008	NaN	M	191315
9	2008	NaN	F	179696
9	2008	NaN	M	188964

```
10      2008    NaN      F  175314
10      2008    NaN      M  183219
11      2008    NaN      F  158939
11      2008    NaN      M  165468
12      2008    NaN      F  173215
12      2008    NaN      M  181235

[15547 rows x 4 columns]
```

```
In [55]: #top 3 months with the highest births
         df.groupby('month')['births'].sum().sort_values(ascending=False).head(3)
```

```
Out[55]: month
         8    13528007
         7    13367556
         9    13252831
         Name: births, dtype: int64
```

```
In [56]: df.head()
```

```
Out[56]:    year  month  day gender  births
         0  1969      1  1.0      F    4046
         1  1969      1  1.0      M    4440
         2  1969      1  2.0      F    4454
         3  1969      1  2.0      M    4548
         4  1969      1  3.0      F    4548
```

```
In [58]: df_new.tail()
```

```
Out[58]:        year  month   day gender  births
         15062  1988     12  29.0      M    5944
         15063  1988     12  30.0      F    5742
         15064  1988     12  30.0      M    6095
         15065  1988     12  31.0      F    4435
         15066  1988     12  31.0      M    4698
```

```
In [59]: df_new.set_index('day')
```

```
Out[59]:       year  month gender  births
         day
         1.0   1969      1      F    4046
         1.0   1969      1      M    4440
         2.0   1969      1      F    4454
         2.0   1969      1      M    4548
         3.0   1969      1      F    4548
         3.0   1969      1      M    4994
         4.0   1969      1      F    4440
         4.0   1969      1      M    4520
         5.0   1969      1      F    4192
```

5.0	1969	1	M	4198
6.0	1969	1	F	4710
6.0	1969	1	M	4850
7.0	1969	1	F	4646
7.0	1969	1	M	5092
8.0	1969	1	F	4800
8.0	1969	1	M	4934
9.0	1969	1	F	4592
9.0	1969	1	M	4842
10.0	1969	1	F	4852
10.0	1969	1	M	5190
11.0	1969	1	F	4580
11.0	1969	1	M	4598
12.0	1969	1	F	4126
12.0	1969	1	M	4324
13.0	1969	1	F	4758
13.0	1969	1	M	5076
14.0	1969	1	F	5070
14.0	1969	1	M	5296
15.0	1969	1	F	4798
15.0	1969	1	M	5096
...	...	...	...	...
17.0	1988	12	F	4270
17.0	1988	12	M	4486
18.0	1988	12	F	4211
18.0	1988	12	M	4220
19.0	1988	12	F	5651
19.0	1988	12	M	6065
20.0	1988	12	F	6092
20.0	1988	12	M	6343
21.0	1988	12	F	5462
21.0	1988	12	M	5861
22.0	1988	12	F	5219
22.0	1988	12	M	5510
23.0	1988	12	F	4887
23.0	1988	12	M	5110
24.0	1988	12	F	4024
24.0	1988	12	M	4269
25.0	1988	12	F	3874
25.0	1988	12	M	3961
26.0	1988	12	F	4274
26.0	1988	12	M	4409
27.0	1988	12	F	5633
27.0	1988	12	M	5895
28.0	1988	12	F	5858
28.0	1988	12	M	5989
29.0	1988	12	F	5760
29.0	1988	12	M	5944

```
30.0  1988     12      F    5742
30.0  1988     12      M    6095
31.0  1988     12      F    4435
31.0  1988     12      M    4698

[15067 rows x 4 columns]
```

In [60]:
```
#top 3 days with the highest number of births
df_new.groupby('day')['births'].sum().sort_values(ascending=False).head(3)
```

Out[60]:
```
day
20.0    2337631
14.0    2335551
17.0    2335458
Name: births, dtype: int64
```