04a_exploratory_pandas

June 24, 2015

1 Getting Started with Exploratory Data Analysis

3 important Python packages 1. NumPy for efficient computation on arrays 2. Pandas for data analysis 3. Matplotlib for plotting in the notebook

```
In [2]: import os
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
```

1.1 Pandas

Python module for manipulating tabular data

1.2 `pandas`

- Provides python a `DataFrame`
- Structured manipulation tools
- Built on top of `numpy`
- Huge growth from 2011-2012
- Very **efficient**
- Great for *medium* data

Resources

- pandas.pydata.org
- Python for Data Analysis by Wes McKinney
- Data Wrangling Kung Fu with Pandas by Wes McKinney
- Cheat sheet by Quandl

1.2.1 Why `pandas`?

80% of the effort in data analysis is spent cleaning data. Hadley Wickham

Efficency

- Different views of data
- Tidy data by Hadley Wickham

Raw data is often in the wrong format

- How often to you download an array ready for array-oriented computing?
- e.g. `scikit-learn` interface

Storage may be best in a different format

- Sparse representations
- Upload to database

1.3 Simple example using the Walrus data

1.3.1 Reading a CSV file as text

```
In [3]: filename = os.path.join('Walrus_Data','Walruses.csv')
        f = open(filename, 'r')
        lines = f.readlines()
        lines[:10]
```

```
Out[3]: ['Walrus,DateTimeUTC,Xcoord,Ycoord,Behav,Longitude,Latitude\r\n',
         '271,5/31/2008 19:25,"95,616.95","-528,324.60",1.009,-167.9560949,65.24871506\r\n',
         '271,6/1/2008 3:24,"84,741.71","-511,653.75",1.0005,-168.1779869,65.40121711\r\n',
         '271,6/1/2008 11:24,"71,834.45","-491,176.95",1.00625,-168.4443605,65.58796906\r\n',
         '271,6/1/2008 19:24,"65,275.80","-478,935.62",1.02025,-168.5802843,65.6991429\r\n',
         '271,6/2/2008 3:24,"69,343.24","-473,948.91",1.00775,-168.4892154,65.74298362\r\n',
         '271,6/2/2008 11:24,"72,634.53","-457,308.67",1,-168.4082441,65.89142326\r\n',
         '271,6/2/2008 19:24,"73,253.86","-425,586.14",1,-168.3764825,66.17566325\r\n',
         '271,6/3/2008 3:24,"79,223.97","-401,784.87",1.00975,-168.2291622,66.38754225\r\n',
         '271,6/3/2008 11:24,"77,052.23","-382,920.49",1.20225,-168.2658694,66.5571914\r\n']
```

1.3.2 Creating a `DataFrame`

```
df = pd.read_csv(filename)
print df
```

Why store it this way?

- Different type
- Different metric

1.3.3 Converting the DateTimeUTC Column

NumPy datetime64 dtype

1.3.4 Converting the `Xcoord` and `Ycoord`

thousands=‘,’

```
In [6]: filename = os.path.join('Walrus_Data','Walruses.csv')
        df = pd.read_csv(filename, parse_dates=[1],
                         thousands=',')
        df.head(5)
```

```
Out[6]:    Walrus         DateTimeUTC    Xcoord     Ycoord    Behav   Longitude  \
        0     271 2008-05-31 19:25:00  95616.95 -528324.60  1.00900 -167.956095
        1     271 2008-06-01 03:24:00  84741.71 -511653.75  1.00050 -168.177987
        2     271 2008-06-01 11:24:00  71834.45 -491176.95  1.00625 -168.444360
        3     271 2008-06-01 19:24:00  65275.80 -478935.62  1.02025 -168.580284
        4     271 2008-06-02 03:24:00  69343.24 -473948.91  1.00775 -168.489215

            Latitude
        0  65.248715
```

```
1  65.401217
2  65.587969
3  65.699143
4  65.742984
```

```
In [10]: df.describe()
```

```
Out[10]:            Walrus         Xcoord         Ycoord       Behav   Longitude  \
         count  454.000000     454.000000     454.000000  454.000000  454.000000
         mean   281.759912  192488.612731   26570.073264    1.432807 -153.571969
         std     19.599907  192785.195478  225551.527159    0.391243   59.339289
         min    271.000000 -473786.460000 -528324.600000    1.000000 -179.603905
         25%    271.000000   73500.247500  -74775.707500    1.056688 -167.684144
         50%    271.000000  269021.360000  102471.775000    1.296250 -162.206341
         75%    281.000000  291732.090000  212473.185000    1.866563 -161.616675
         max    322.000000  504837.070000  246443.380000    1.996250  179.723232

                  Latitude
         count  454.000000
         mean    70.085316
         std      1.933906
         min     65.248715
         25%     69.287408
         50%     70.793950
         75%     71.576913
         max     72.041307
```

1.3.5 Indexing

```
In [45]: df[2:10]
```

```
Out[45]:    Walrus         DateTimeUTC    Xcoord     Ycoord    Behav   Longitude  \
         2     271 2008-06-01 11:24:00  71834.45 -491176.95  1.00625 -168.444360
         3     271 2008-06-01 19:24:00  65275.80 -478935.62  1.02025 -168.580284
         4     271 2008-06-02 03:24:00  69343.24 -473948.91  1.00775 -168.489215
         5     271 2008-06-02 11:24:00  72634.53 -457308.67  1.00000 -168.408244
         6     271 2008-06-02 19:24:00  73253.86 -425586.14  1.00000 -168.376483
         7     271 2008-06-03 03:24:00  79223.97 -401784.87  1.00975 -168.229162
         8     271 2008-06-03 11:24:00  77052.23 -382920.49  1.20225 -168.265869
         9     271 2008-06-03 19:24:00  73380.11 -379615.25  1.24225 -168.346486

             Latitude
         2  65.587969
         3  65.699143
         4  65.742984
         5  65.891423
         6  66.175663
         7  66.387542
         8  66.557191
         9  66.587727
```

```
In [7]: print(len(df))
        df[-5:]
```

```
454
```

Out[7]:

	Walrus	DateTimeUTC	Xcoord	Ycoord	Behav	Longitude	Latitude
449	322	2009-07-05 12:11:00	-465471.95	169270.91	1.32675	177.051563	71.071287
450	322	2009-07-05 20:11:00	-473786.46	174848.10	1.52075	176.794183	71.104176
451	322	2009-07-06 04:11:00	-462401.15	175580.80	1.47450	177.097679	71.132315
452	322	2009-07-06 12:11:00	-449812.23	178045.34	1.47200	177.424821	71.177357
453	322	2009-07-06 20:11:00	-443963.94	180789.20	1.46775	177.568265	71.212108

1.3.6 Hierarchical columns

In [8]:
```
wd = df.pivot(index='DateTimeUTC', columns='Walrus') #row, column, values (optional)
wd[:7]
```

Out[8]:

	Xcoord		Ycoord		Behav		Longitude		Latitude	
Walrus	271	281	322	271	281	322	271	281	322	271	281	322
DateTimeUTC												
2008-05-31 19:25:00	95616.95	NaN	NaN	-528324.60	NaN	NaN	1.00900	NaN	NaN	-167.956095	NaN	NaN	65.248715	NaN	NaN
2008-06-01 03:24:00	84741.71	NaN	NaN	-511653.75	NaN	NaN	1.00050	NaN	NaN	-168.177987	NaN	NaN	65.401217	NaN	NaN
2008-06-01 11:24:00	71834.45	NaN	NaN	-491176.95	NaN	NaN	1.00625	NaN	NaN	-168.444360	NaN	NaN	65.587969	NaN	NaN
2008-06-01 19:24:00	65275.80	NaN	NaN	-478935.62	NaN	NaN	1.02025	NaN	NaN	-168.580284	NaN	NaN	65.699143	NaN	NaN
2008-06-02 02:25:00	NaN	65600.95	NaN	NaN	-417464.74	NaN	NaN	1.544	NaN	NaN	-168.541792	NaN	NaN	66.250190	NaN
2008-06-02 03:24:00	69343.24	NaN	NaN	-473948.91	NaN	NaN	1.00775	NaN	NaN	-168.489215	NaN	NaN	65.742984	NaN	NaN
2008-06-02 10:24:00	NaN	61574.27	NaN	NaN	-421676.30	NaN	NaN	1.567	NaN	NaN	-168.633333	NaN	NaN	66.213262	NaN

In [48]:
```
wd['Behav'][:5]
```

Out[48]:

Walrus	271	281	322
DateTimeUTC			

```
2008-05-31 19:25:00  1.00900    NaN  NaN
2008-06-01 03:24:00  1.00050    NaN  NaN
2008-06-01 11:24:00  1.00625    NaN  NaN
2008-06-01 19:24:00  1.02025    NaN  NaN
2008-06-02 02:25:00      NaN  1.544  NaN
```

```
In [11]: longLat = df[['Walrus', 'Longitude', 'Latitude']]
         longLat[2:10:2]
```

```
Out[11]:    Walrus   Longitude   Latitude
         2     271 -168.444360  65.587969
         4     271 -168.489215  65.742984
         6     271 -168.376483  66.175663
         8     271 -168.265869  66.557191
```

```
In [15]: wd[['Longitude', 'Latitude']][::50]
```

```
Out[15]:                       Longitude                              Latitude             \
         Walrus                      271         281         322         271         281
         DateTimeUTC
         2008-05-31 19:25:00 -167.956095         NaN         NaN  65.248715         NaN
         2008-06-09 18:24:00         NaN -169.854440         NaN         NaN  66.761239
         2008-06-18 02:24:00         NaN -170.786003         NaN         NaN  66.934435
         2008-07-02 11:24:00 -161.826253         NaN         NaN  70.555427         NaN
         2008-07-19 03:24:00 -158.004690         NaN         NaN  71.153361         NaN
         2008-08-04 19:24:00 -161.169659         NaN         NaN  71.449231         NaN
         2008-08-21 11:24:00 -162.127242         NaN         NaN  71.795703         NaN
         2008-09-07 03:24:00 -161.886692         NaN         NaN  71.839187         NaN
         2009-06-19 04:11:00         NaN         NaN -163.064091         NaN         NaN
         2009-07-05 20:11:00         NaN         NaN  176.794183         NaN         NaN


         Walrus                     322
         DateTimeUTC
         2008-05-31 19:25:00        NaN
         2008-06-09 18:24:00        NaN
         2008-06-18 02:24:00        NaN
         2008-07-02 11:24:00        NaN
         2008-07-19 03:24:00        NaN
         2008-08-04 19:24:00        NaN
         2008-08-21 11:24:00        NaN
         2008-09-07 03:24:00        NaN
         2009-06-19 04:11:00  71.241689
         2009-07-05 20:11:00  71.104176
```

```
In [9]: wd['Behav'][281].values
```

```
Out[9]: array([     nan,      nan,      nan,      nan,  1.544  ,      nan,
                1.567  ,      nan,  1.62975,      nan,  1.70475,      nan,
                1.82125,      nan,  1.8405 ,      nan,  1.80675,      nan,
                1.73125,      nan,  1.7165 ,      nan,  1.69875,      nan,
                1.62175,      nan,  1.654  ,      nan,  1.70825,      nan,
                1.24925,      nan,  1.0765 ,      nan,  1.0415 ,      nan,
                1.048  ,      nan,  1.04925,      nan,  1.0425 ,      nan,
                1.05   ,      nan,  1.16075,      nan,  1.3745 ,      nan,
```

```
1.38275,      nan,  1.3365 ,      nan,  1.30775,      nan,
1.29125,      nan,  1.30925,      nan,  1.24925,      nan,
1.2215 ,      nan,  1.255  ,      nan,  1.36875,      nan,
1.4405 ,      nan,  1.4865 ,      nan,  1.50425,      nan,
1.52275,      nan,  1.5245 ,      nan,  1.53975,      nan,
1.53825,      nan,  1.53775,      nan,  1.52325,      nan,
1.49275,      nan,  1.47225,      nan,  1.42675,      nan,
1.3735 ,      nan,  1.29825,      nan,  1.22075,      nan,
1.14375,      nan,  1.0705 ,      nan,  1.05375,      nan,
1.165  ,      nan,  1.32725,      nan,  1.4355 ,      nan,
1.4635 ,      nan,  1.45175,      nan,  1.429  ,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
    nan,      nan,      nan,      nan,      nan,      nan,
```

```
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan,      nan,      nan,
                    nan,      nan,      nan,      nan])
```

1.3.7 Extracting with a Condition

- Extracting Walrus 271 from the table

```
In [18]: df[df.Walrus == 271][:5]
```

```
Out[18]:    Walrus         DateTimeUTC    Xcoord     Ycoord    Behav   Longitude  \
         0     271 2008-05-31 19:25:00  95616.95 -528324.60  1.00900 -167.956095
         1     271 2008-06-01 03:24:00  84741.71 -511653.75  1.00050 -168.177987
         2     271 2008-06-01 11:24:00  71834.45 -491176.95  1.00625 -168.444360
         3     271 2008-06-01 19:24:00  65275.80 -478935.62  1.02025 -168.580284
         4     271 2008-06-02 03:24:00  69343.24 -473948.91  1.00775 -168.489215

             Latitude
         0  65.248715
         1  65.401217
         2  65.587969
         3  65.699143
         4  65.742984
```

```
In [21]: wd.columns
```

```
Out[21]: MultiIndex(levels=[[u'Xcoord', u'Ycoord', u'Behav', u'Longitude', u'Latitude'], [271, 281, 322]
                    labels=[[0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4], [0, 1, 2, 0, 1, 2, 0, 1, 2, 0
                    names=[None, u'Walrus'])
```

```
In [32]: wd[[('Latitude', 271), ('Longitude', 271)]][:5]
```

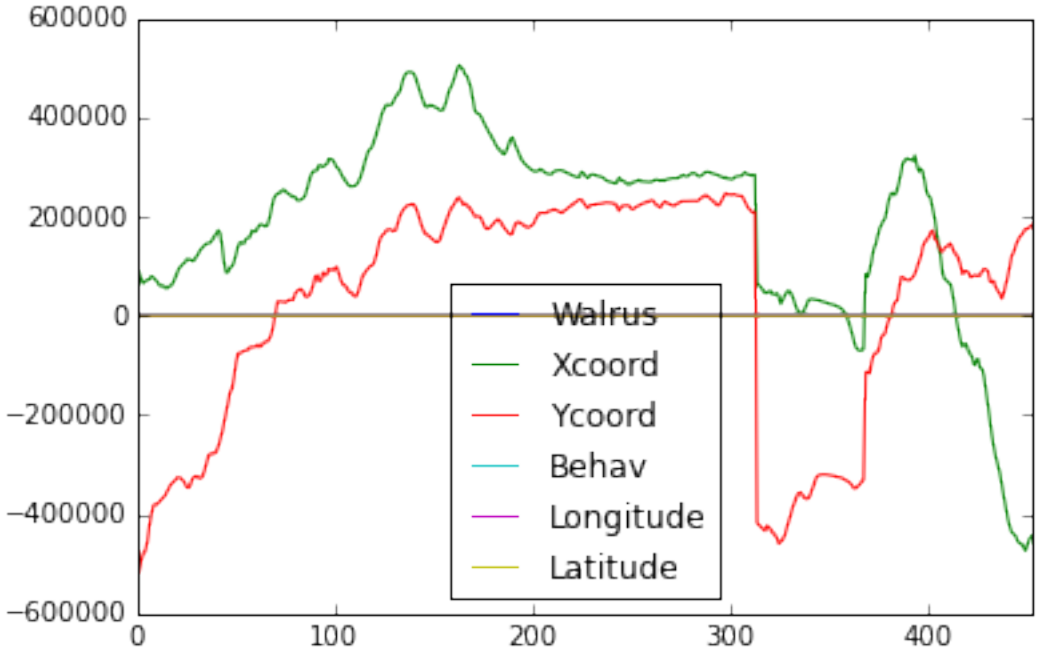
```
Out[32]:                       Latitude   Longitude
         Walrus                     271         271
         DateTimeUTC
         2008-05-31 19:25:00  65.248715 -167.956095
         2008-06-01 03:24:00  65.401217 -168.177987
         2008-06-01 11:24:00  65.587969 -168.444360
         2008-06-01 19:24:00  65.699143 -168.580284
         2008-06-02 02:25:00        NaN         NaN
```

1.4 Simple Plotting

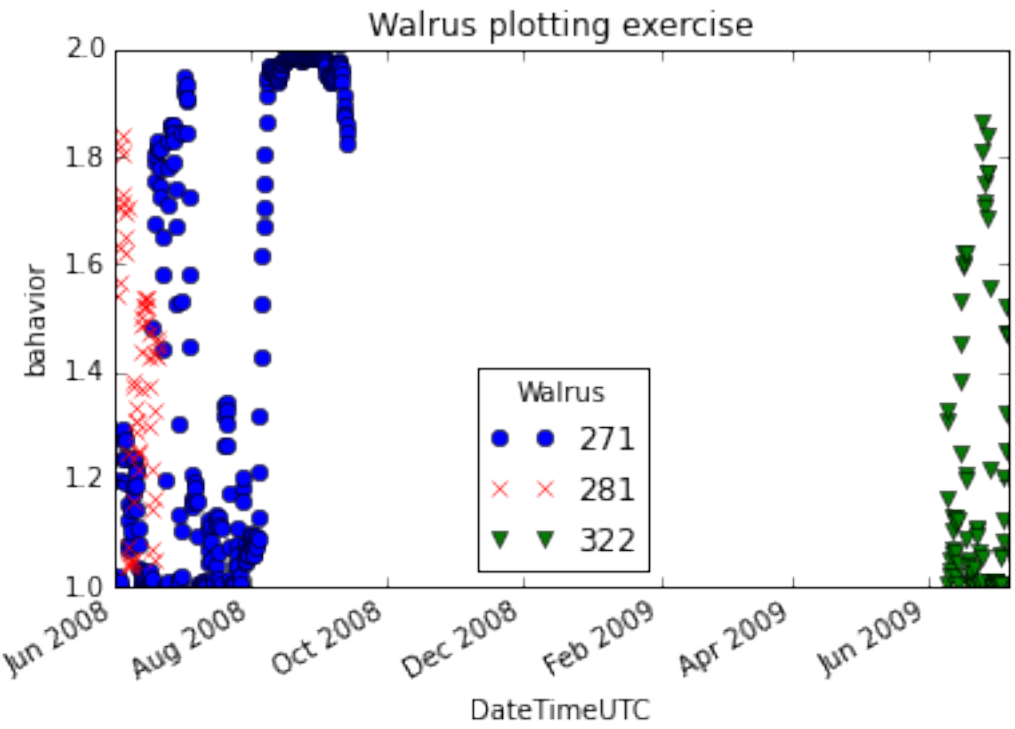
- Plot the behavior of each walrus over time

In [34]:
```
df.plot()
```

Out[34]: <matplotlib.axes._subplots.AxesSubplot at 0x1057dcd50>



In [44]:
```
wd['Behav'].plot(style=['bo','rx','gv'])
plt.ylabel('bahavior')
plt.title('Walrus plotting exercise')
plt.savefig('walrus_behav.pdf')
plt.show()
```



In [61]: !ls

```
01_exploratory_pandas.ipynb                 06_CSV_to_NetCDF_Solution.ipynb
01_introduction-IPython-notebook.ipynb        Walrus_Data
01_introduction-IPython-notebook.pdf         data_overview.png
01_introduction-IPython-notebook_files         ipython-notebook-keyboard.png
02_Data_Transfer.pdf                         ipython-notebook-sharing.png
03_HPC_File_Systems.pdf                        ipython-notebook.png
04_python_reading-plotting.ipynb          rc_logo.png
05_Data_Conversion_Cleaning.pdf                traditional_python.png
06_CSV_to_NetCDF_Exercise.ipynb                walrus_behav.png
```

Importing the image into the Markdown