

Week4

August 5, 2020

1 Pandas Visualization

```
In [1]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

```
%matplotlib notebook
```

```
In [3]: # see the pre-defined styles provided.
plt.style.available
```

```
Out[3]: ['grayscale',
'seaborn-colorblind',
'seaborn-bright',
'seaborn-deep',
'seaborn-pastel',
'classic',
'seaborn-muted',
'seaborn-darkgrid',
'seaborn-white',
'seaborn-paper',
'seaborn-dark',
'seaborn-whitegrid',
'dark_background',
'seaborn-poster',
'ggplot',
'seaborn-ticks',
'seaborn-dark-palette',
'seaborn-notebook',
'bmh',
'seaborn-talk',
'fivethirtyeight',
'seaborn']
```

```
In [4]: # use the 'seaborn-colorblind' style
plt.style.use('seaborn-colorblind')
```

1.0.1 DataFrame.plot

```
In [7]: np.random.seed(123)
```

```
df = pd.DataFrame({'A': np.random.randn(365).cumsum(0),  
                  'B': np.random.randn(365).cumsum(0) + 20,  
                  'C': np.random.randn(365).cumsum(0) - 20},  
                  index=pd.date_range('1/1/2017', periods=365))  
  
df.head()
```

```
Out [7]:
```

	A	B	C
2017-01-01	-1.085631	20.059291	-20.230904
2017-01-02	-0.088285	21.803332	-16.659325
2017-01-03	0.194693	20.835588	-17.055481
2017-01-04	-1.311601	21.255156	-17.093802
2017-01-05	-1.890202	21.462083	-19.518638

```
In [8]: df.plot(); # add a semi-colon to the end of the plotting call to suppress u
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

We can select which plot we want to use by passing it into the 'kind' parameter.

```
In [9]: df.plot('A', 'B', kind = 'scatter');
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

You can also choose the plot kind by using the `DataFrame.plot.kind` methods instead of providing the `kind` keyword argument.

kind: - 'line' : line plot (default) - 'bar' : vertical bar plot - 'barh' : horizontal bar plot
- 'hist' : histogram - 'box' : boxplot - 'kde' : Kernel Density Estimation plot - 'density'
: same as 'kde' - 'area' : area plot - 'pie' : pie plot - 'scatter' : scatter plot - 'hexbin' :
hexbin plot

```
In [11]: # create a scatter plot of columns 'A' and 'C', with changing color (c) and  
         df.plot.scatter('A', 'C', c='B', s=df['B'], colormap='viridis')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
/opt/conda/lib/python3.6/site-packages/matplotlib/axes/_axes.py:545: UserWarning: No labelled objects found. "
```

```
In [12]: ax = df.plot.scatter('A', 'C', c='B', s=df['B'], colormap='viridis')
        ax.set_aspect('equal')
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [13]: df.plot.box();
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [14]: df.plot.hist(alpha=0.7);
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

[Kernel density estimation plots](#) are useful for deriving a smooth continuous function from a given sample.

```
In [15]: df.plot.kde();
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

1.0.2 pandas.tools.plotting

[Iris flower data set](#)

```
In [16]: iris = pd.read_csv('iris.csv')
        iris.head()
```

```
Out[16]:
```

	SepalLength	SepalWidth	PetalLength	PetalWidth	Name
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
In [17]: pd.tools.plotting.scatter_matrix(iris);
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [18]: plt.figure()  
         pd.tools.plotting.parallel_coordinates(iris, 'Name');
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

2 Seaborn

```
In [20]: import numpy as np  
         import pandas as pd  
         import matplotlib.pyplot as plt  
         import seaborn as sns
```

```
%matplotlib notebook
```

```
In [21]: np.random.seed(1234)
```

```
v1 = pd.Series(np.random.normal(0,10,1000), name='v1')  
v2 = pd.Series(2*v1 + np.random.normal(60,15,1000), name='v2')
```

```
In [24]: plt.figure()  
         plt.hist(v1, alpha=0.7, bins=np.arange(-50,150,5), label='v1');  
         plt.hist(v2, alpha=0.7, bins=np.arange(-50,150,5), label='v2');  
         plt.legend();
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [27]: # plot a kernel density estimation over a stacked barchart  
         plt.figure()  
         plt.hist([v1, v2], histtype='barstacked', normed=True);  
         v3 = np.concatenate((v1,v2))  
         sns.kdeplot(v3);
```

```
<IPython.core.display.Javascript object>
```

<IPython.core.display.HTML object>

```
In [28]: plt.figure()
         # we can pass keyword arguments for each individual component of the plot
         sns.distplot(v3, hist_kws={'color': 'Teal'}, kde_kws={'color': 'Navy'});
```

<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>

```
In [29]: sns.jointplot(v1, v2, alpha=0.4);
```

<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>

```
In [30]: grid = sns.jointplot(v1, v2, alpha=0.4);
         grid.ax_joint.set_aspect('equal')
```

<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>

```
In [ ]: sns.jointplot(v1, v2, kind='hex');
```

```
In [31]: # set the seaborn style for all the following plots
         sns.set_style('white')
```

```
         sns.jointplot(v1, v2, kind='kde', space=0);
```

<IPython.core.display.Javascript object>

<IPython.core.display.HTML object>

```
In [32]: iris = pd.read_csv('iris.csv')
         iris.head()
```

```
Out[32]:
```

	SepalLength	SepalWidth	PetalLength	PetalWidth	Name
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

```
In [36]: sns.pairplot(iris, hue='Name', diag_kind='kde', size=2);
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [37]: plt.figure(figsize=(8,6))
         plt.subplot(121)
         sns.swarmplot('Name', 'PetalLength', data=iris);
         plt.subplot(122)
         sns.violinplot('Name', 'PetalLength', data=iris);
```

```
<IPython.core.display.Javascript object>
```

```
<IPython.core.display.HTML object>
```

```
In [ ]:
```

```
In [ ]:
```