Module 02

Pandas
Data Visualization

Data Science Developer

Purwadhika
Startup and Coding School

Pandas Data Visualization

In this lecture we will learn about pandas built-in capabilities for data visualization! It's built-off of matplotlib, but it baked into pandas for easier usage!

Imports

```
import numpy as np
import pandas as pd
%matplotlib inline
```

Data

```
df1 = pd.read_csv('df1',index_col=0)
df1.head()
```

	A	B	C	D
2000-01-01	1.339091	-0.163643	-0.646443	1.041233
2000-01-02	-0.774984	0.137034	-0.882716	-2.253382
2000-01-03	-0.921037	-0.482943	-0.417100	0.478638
2000-01-04	-1.738808	-0.072973	0.056517	0.015085
2000-01-05	-0.905980	1.778576	0.381918	0.291436

```
df2 = pd.read_csv('df2')
df2.head()
```

	a	b	c	d
0	0.039762	0.218517	0.103423	0.957904
1	0.937288	0.041567	0.899125	0.977680
2	0.780504	0.008948	0.557808	0.797510
3	0.672717	0.247870	0.264071	0.444358
4	0.053829	0.520124	0.552264	0.190008

Purwadhika
Startup and Coding School

Style Sheets

Matplotlib has style sheets you can use to make your plots look a little nicer. These style sheets include plot_bmh, plot_fivethirtyeight, plot_ggplot and more.

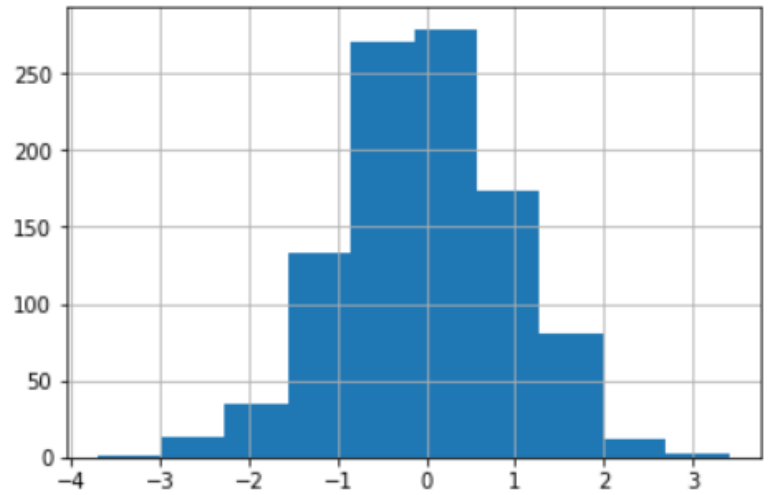
They basically create a set of style rules that your plots follow. They make all your plots have the same look and feel more professional.

Here is how to use them.

Before plt.style.use() your plots look like this:

```
df1['A'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db2270b8>
```



Purwadhika
Startup and Coding School

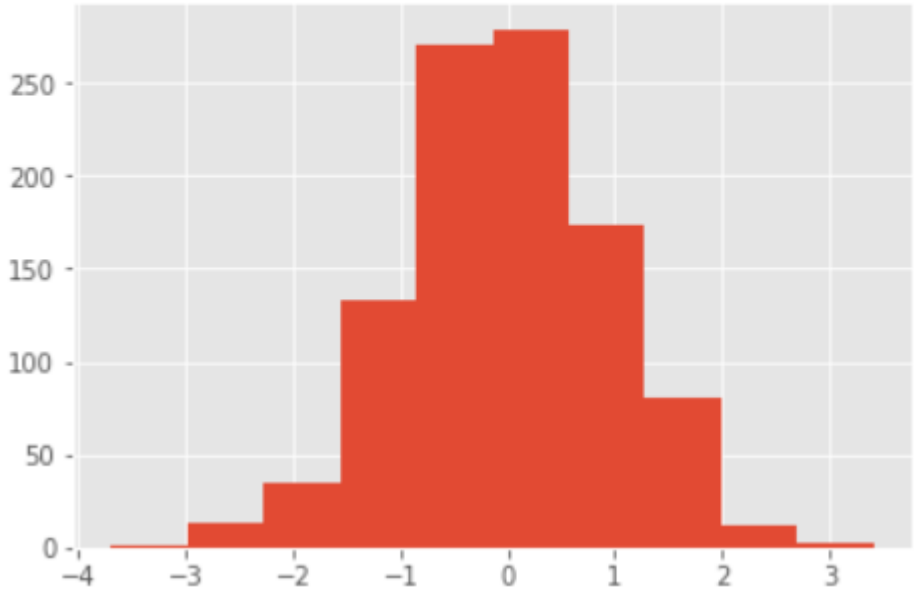
Call the style:

```
import matplotlib.pyplot as plt
plt.style.use('ggplot')
```

Now your plots look like this:

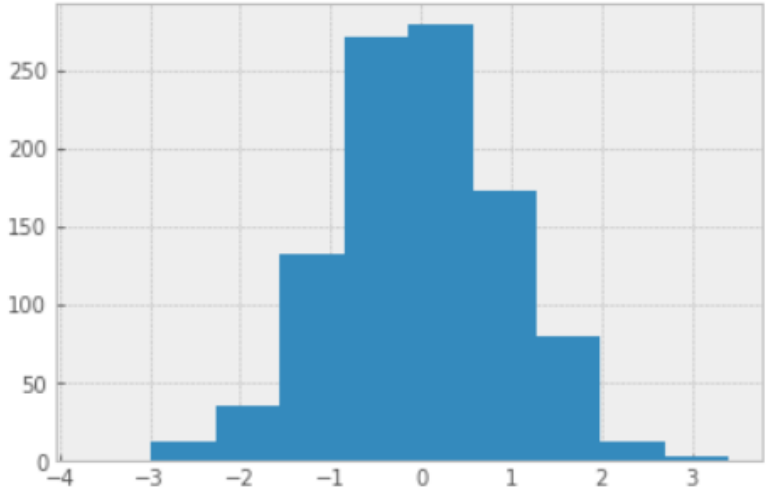
```
df1['A'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db60ea58>
```



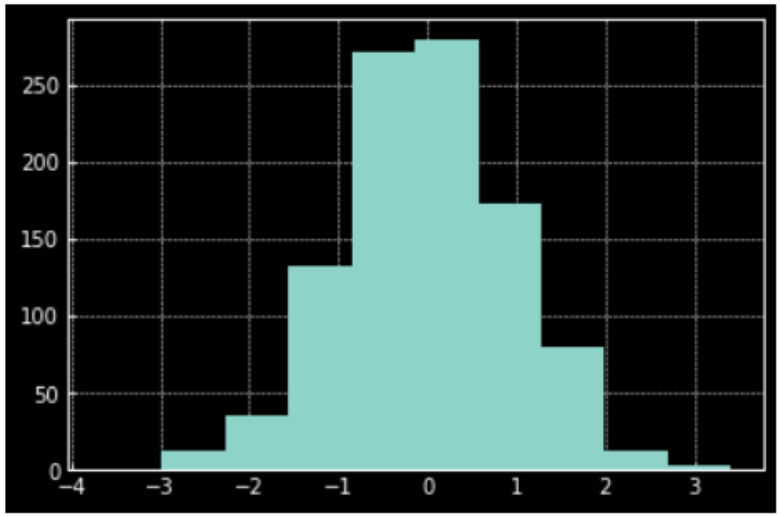
```
plt.style.use('bmh')
df1['A'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db6aafd0>
```




```
plt.style.use('dark_background')
df1['A'].hist()
```

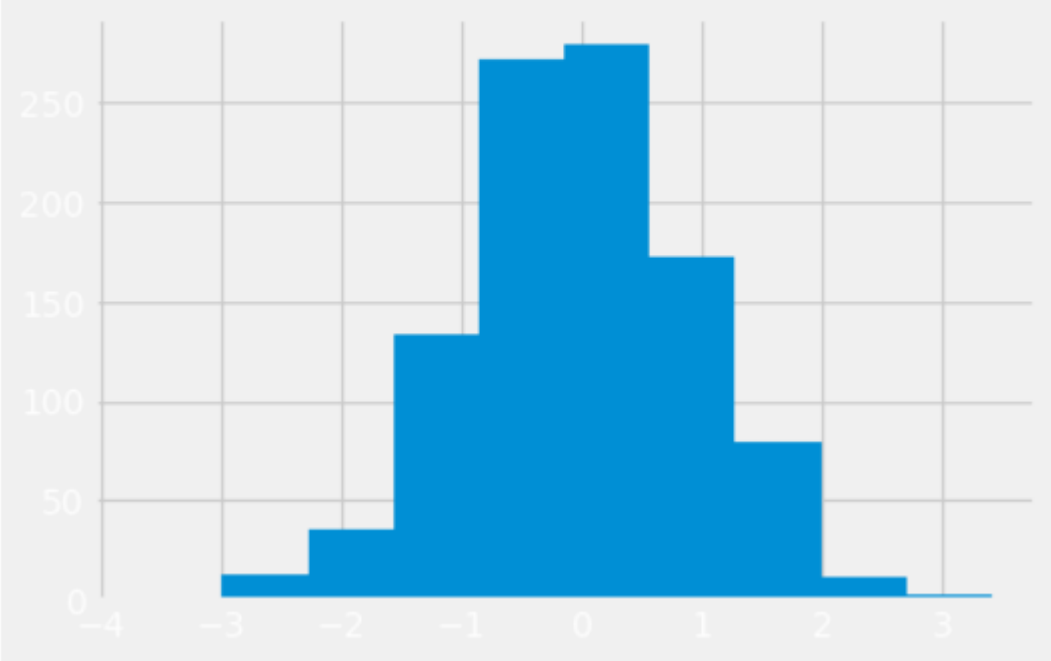
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db6fa550>
```



Purwadhika
Startup and Coding School

```
plt.style.use('fivethirtyeight')
df1['A'].hist()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db77d550>
```




```
plt.style.use('ggplot')
```

Let's stick with the ggplot style and actually show you how to utilize pandas built-in plotting capabilities!

Plot Types

There are several plot types built-in to pandas, most of them statistical plots by nature:

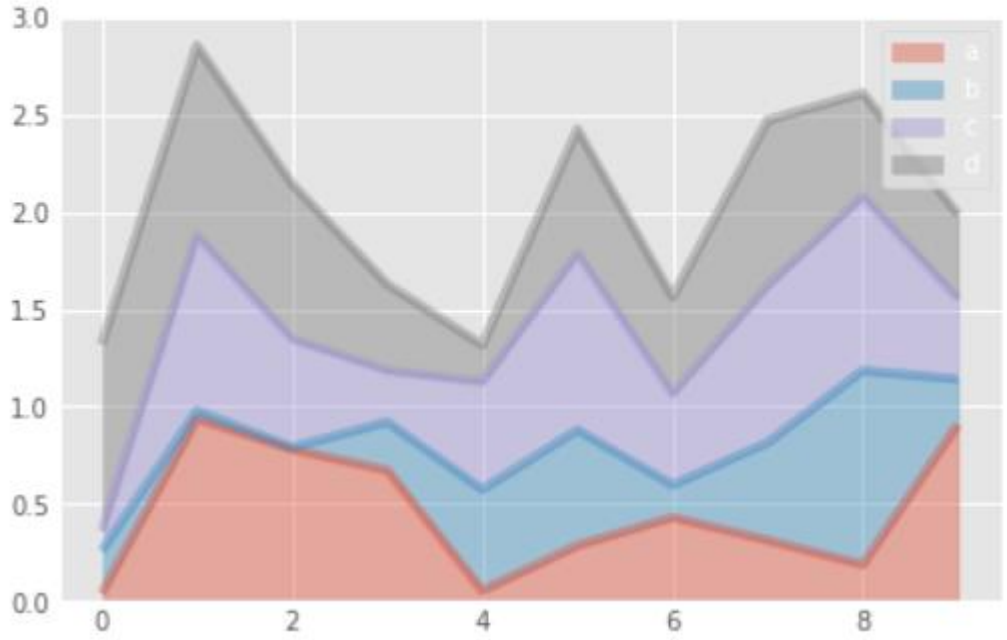
- df.plot.area
- df.plot.barh
- df.plot.density
- df.plot.hist
- df.plot.line
- df.plot.scatter
- df.plot.bar
- df.plot.box
- df.plot.hexbin
- df.plot.kde
- df.plot.pie

You can also just call df.plot(kind='hist') or replace that kind argument with any of the key terms shown in the list above (e.g. 'box','barh', etc..)

Area

```
df2.plot.area(alpha=0.4)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db7e27b8>
```

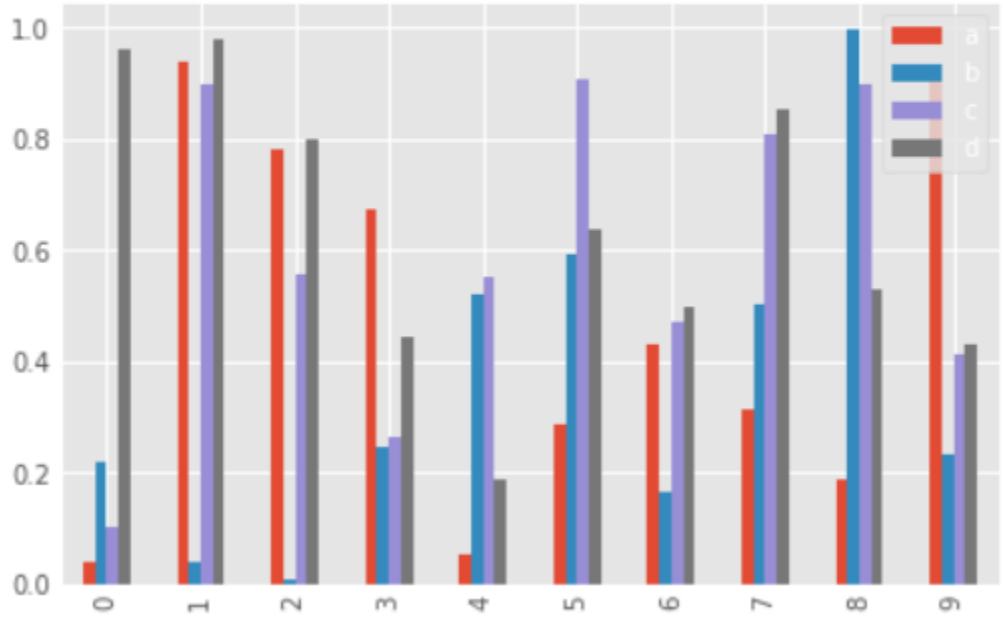


Purwadhika
Startup and Coding School

Barplots

```
df2.plot.bar()
```

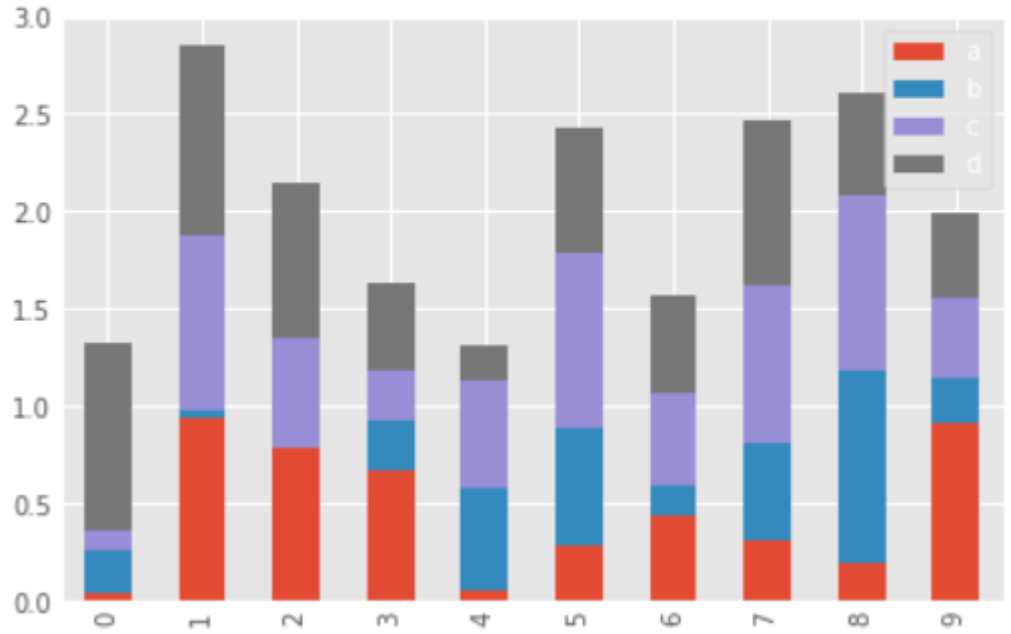
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db9256a0>
```



Purwadhika
Startup and Coding School

```
df2.plot.bar(stacked=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dca8fd68>
```

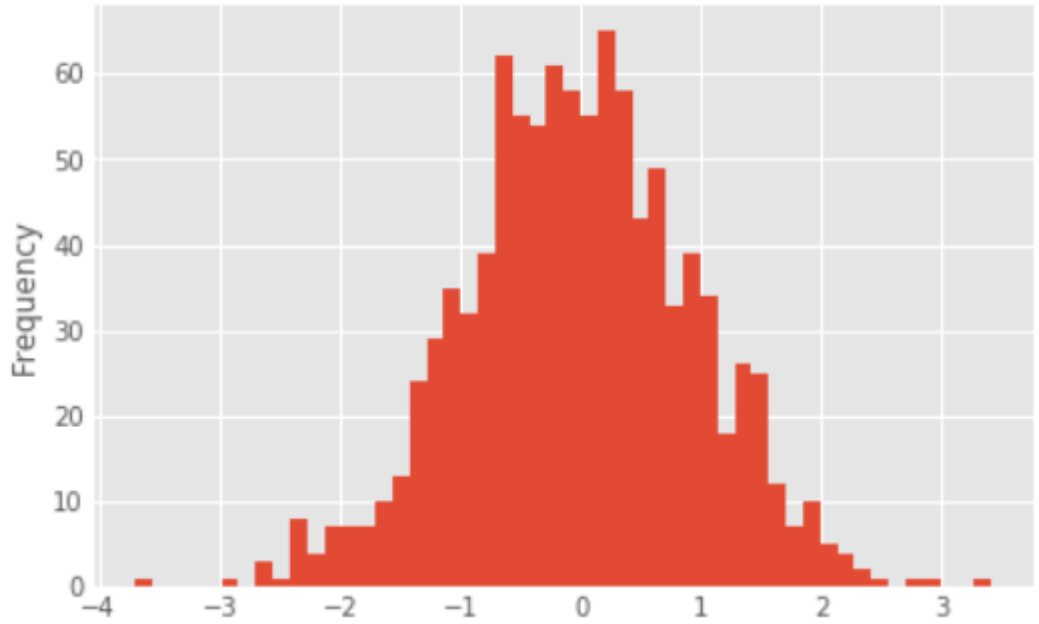


Purwadhika
Startup and Coding School

Histograms

```
df1['A'].plot.hist(bins=50)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0db60e320>
```

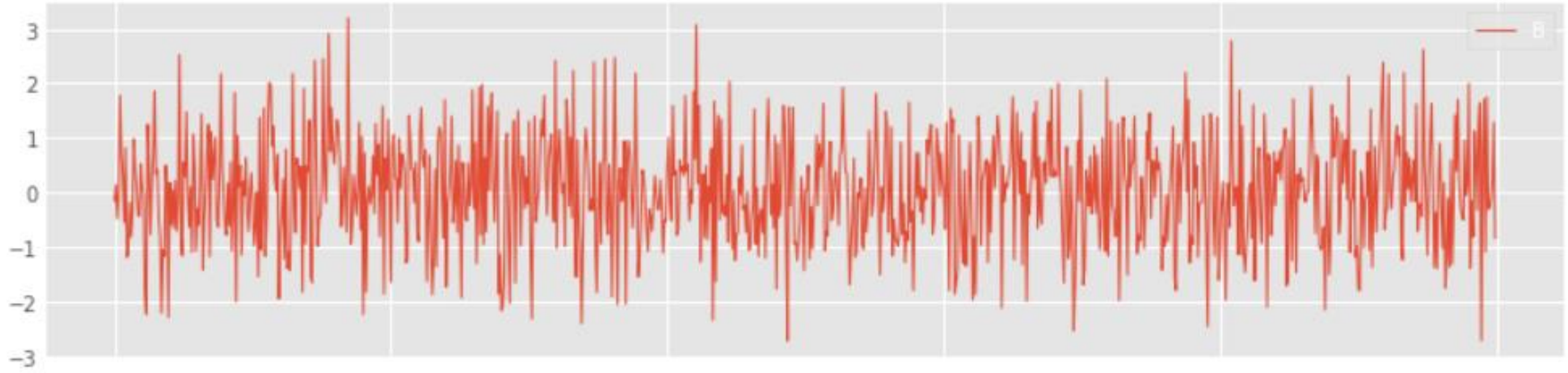


Purwadhika
Startup and Coding School

Line Plots

```
df1.plot.line(y='B',figsize=(12,3),lw=1)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dcb865c0>
```

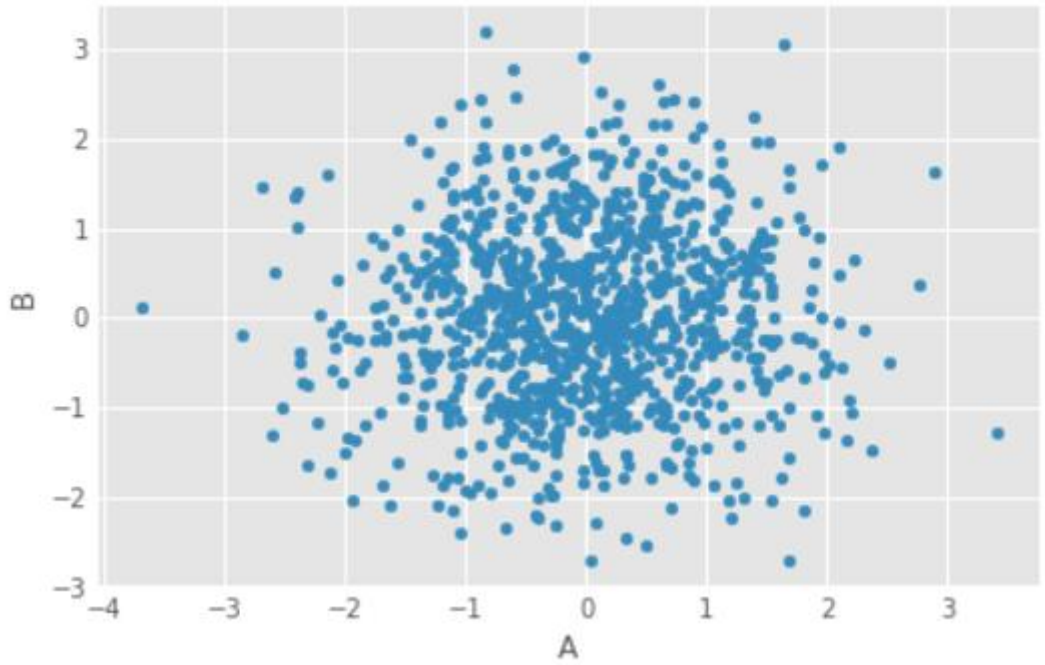


Purwadhika
Startup and Coding School

Scatter Plots

```
df1.plot.scatter(x='A',y='B')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dccdb390>
```

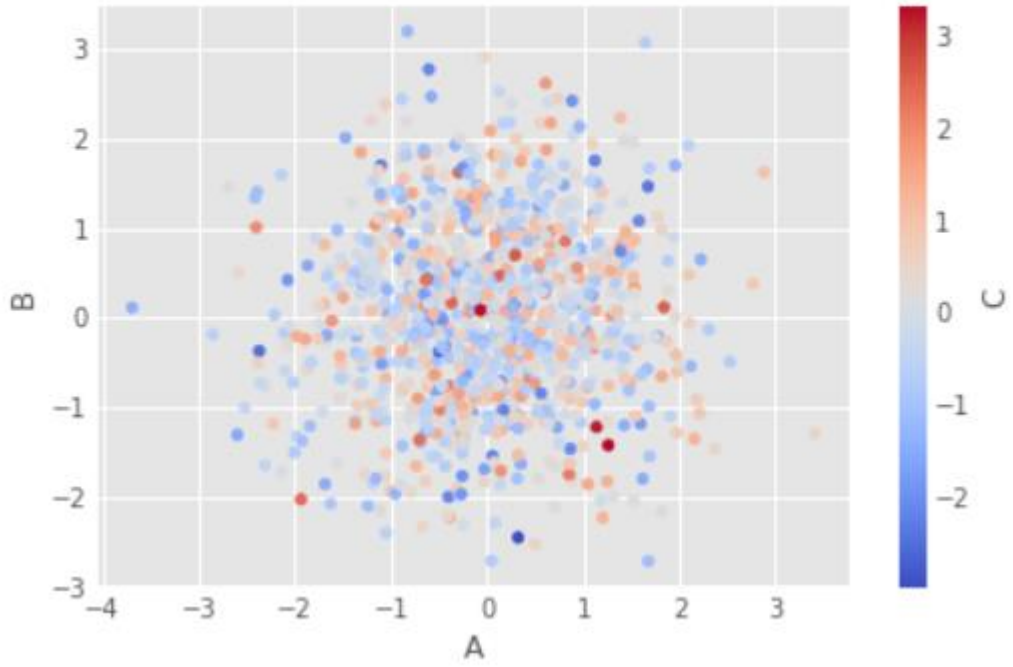


Purwadhika
Startup and Coding School

You can use c to color based off another column value Use cmap to indicate colormap to use. For all the colormaps, check out: http://matplotlib.org/users/colormaps.html

```
df1.plot.scatter(x='A',y='B',c='C',cmap='coolwarm')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dcd4c198>
```



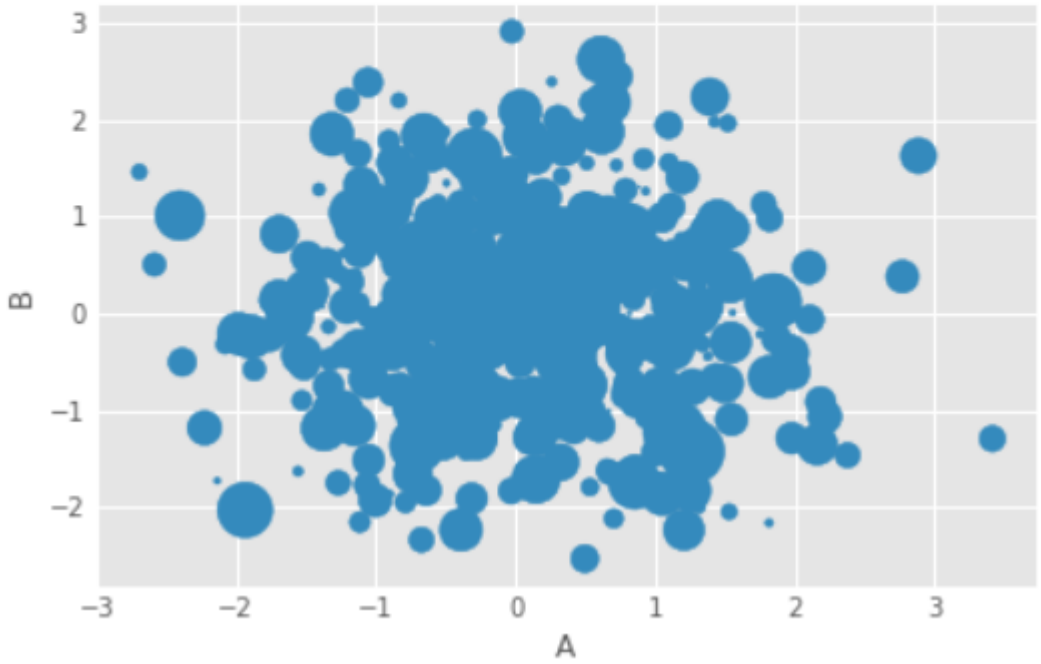
Purwadhika
Startup and Coding School

Or use s to indicate size based off another column. s parameter needs to be an array, not just the name of a column:

```
df1.plot.scatter(x='A',y='B',s=df1['C']*200)
```

```
C:\Users\harto\Anaconda3\lib\site-packages\matplotlib\col
  scale = np.sqrt(self._sizes) * dpi / 72.0 * self._factc
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dcee1940>
```

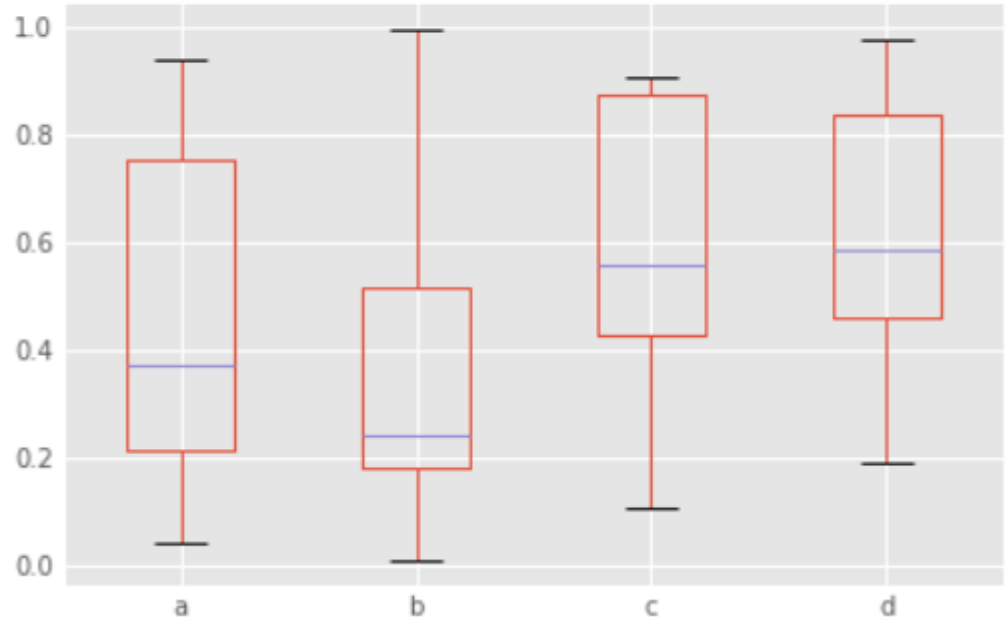


Purwadhika
Startup and Coding School

BoxPlots

```
df2.plot.box() # Can also pass a by= argument for groupby
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dcf3ed68>
```



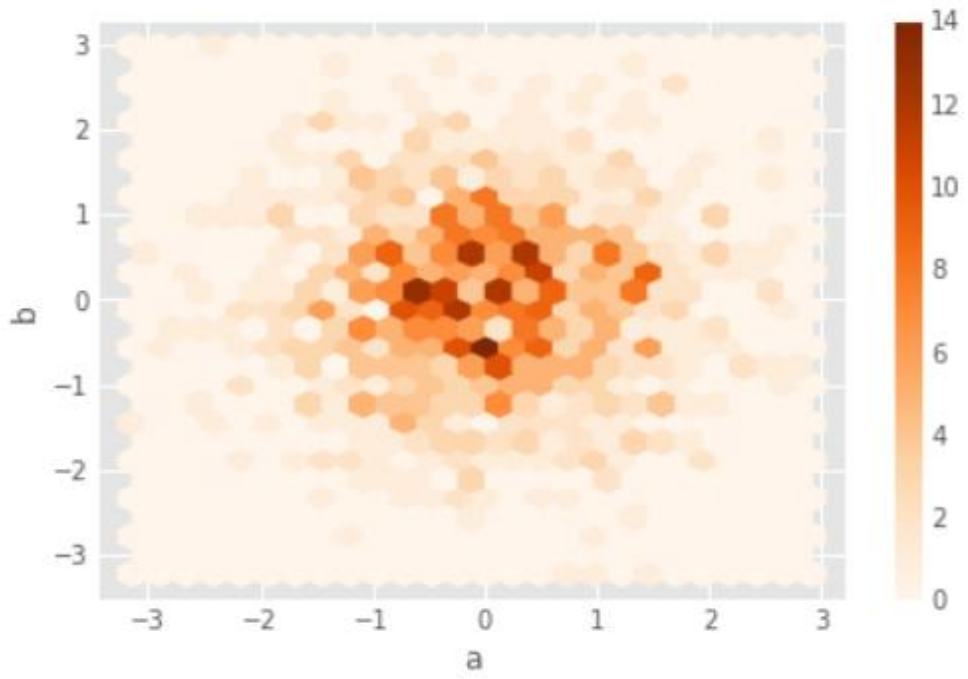
Purwadhika
Startup and Coding School

Hexagonal Bin Plot

Useful for Bivariate Data, alternative to scatterplot:

```
df = pd.DataFrame(np.random.randn(1000, 2), columns=['a', 'b'])
df.plot.hexbin(x='a',y='b',gridsize=25,cmap='Oranges')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dd07e128>
```

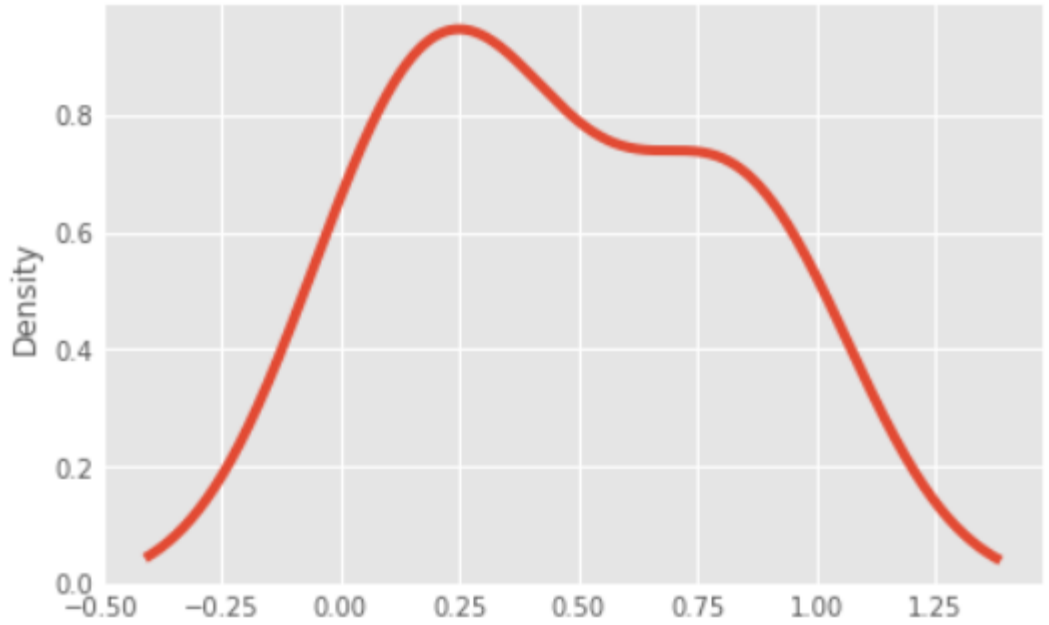


Purwadhika
Startup and Coding School

Kernel Density Estimation Plot (KDE)

```
df2['a'].plot.kde()
```

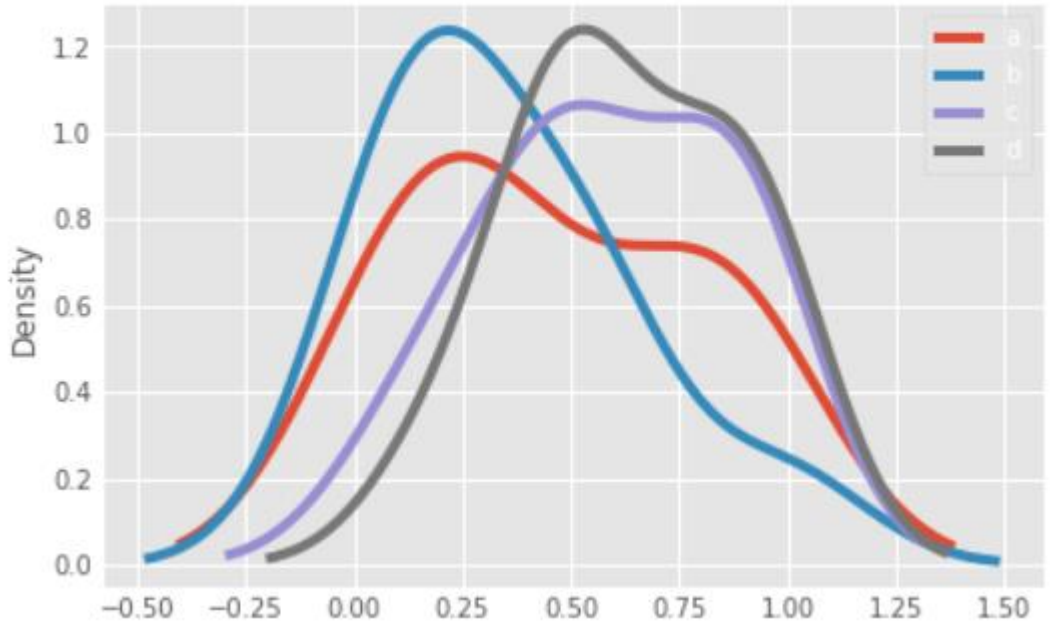
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dd116668>
```



Purwadhika
Startup and Coding School

```
df2.plot.density()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2b0dec4f470>
```



Purwadhika
Startup and Coding School