Pandas

May 13, 2019

1 Pandas

1.0.1 Pandas is used for data analysis and visualization

```
In [13]: import pandas as pd
         df = pd.DataFrame({'Marks':[44,55,36,46,34,22,18,55],'Name':['def','pqr','gpi','xyz',
         df.head()
```

```
Out[13]:    Marks Name
         0     44  def
         1     55  pqr
         2     36  gpi
         3     46  xyz
         4     34  abc
```

```
In [14]: df['Marks'].value_counts()
```

```
Out[14]: 55    2
         46    1
         44    1
         22    1
         18    1
         36    1
         34    1
         Name: Marks, dtype: int64
```

```
In [17]: ans_arr=df['Name'].unique()
```

```
In [18]: print(ans_arr)
         print(type(ans_arr))
```

```
['def' 'pqr' 'gpi' 'xyz' 'abc' 'ccv' 'otp' 'por']
<class 'numpy.ndarray'>
```

```
In [19]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8 entries, 0 to 7
Data columns (total 2 columns):
Marks    8 non-null int64
Name     8 non-null object
dtypes: int64(1), object(1)
memory usage: 208.0+ bytes
```

```
In [20]: df.describe()
```

```
Out[20]:            Marks
         count   8.00000
         mean   38.75000
         std    13.88473
         min    18.00000
         25%    31.00000
         50%    40.00000
         75%    48.25000
         max    55.00000
```

```
In [21]: # selection of data from dataframe
         df_2=df[df['Marks']>50]
         print(df_2)
```

```
   Marks Name
1     55  pqr
7     55  por
```

```
In [22]: def pract_marks(data1):
             return data1+5
         df_3=df['Marks'].apply(pract_marks)
```

```
In [23]: df_3.head()
```

```
Out[23]: 0    49
         1    60
         2    41
         3    51
         4    39
         Name: Marks, dtype: int64
```

```
In [25]: df.sort_values(by='Marks')
```

```
Out[25]:    Marks Name
         6     18  otp
         5     22  ccv
         4     34  abc
```

```
2     36  gpi
0     44  def
3     46  xyz
1     55  pqr
7     55  por
```

```
In [26]: from numpy.random import randn
         df = pd.DataFrame(randn(4,4),index='e1 e2 e3 e4'.split(),columns='f1 f2 f3 f4'.split()
         df
```

Out[26]:

	f1	f2	f3	f4
e1	0.378693	-0.704198	0.705654	0.304902
e2	-0.068748	-0.590469	-1.910214	0.297668
e3	-0.142854	0.492157	1.284218	-1.114191
e4	-1.281741	-0.505274	0.272717	-1.471717

```
In [27]: df[['f3','f4']]
```

Out[27]:

	f3	f4
e1	0.705654	0.304902
e2	-1.910214	0.297668
e3	1.284218	-1.114191
e4	0.272717	-1.471717

```
In [28]: df['f5'] = df['f3'] + df['f2']
         df
```

Out[28]:

	f1	f2	f3	f4	f5
e1	0.378693	-0.704198	0.705654	0.304902	0.001457
e2	-0.068748	-0.590469	-1.910214	0.297668	-2.500683
e3	-0.142854	0.492157	1.284218	-1.114191	1.776375
e4	-1.281741	-0.505274	0.272717	-1.471717	-0.232557

```
In [29]: df.loc['e1']  # Retrieval of data from rows
```

```
Out[29]: f1    0.378693
         f2   -0.704198
         f3    0.705654
         f4    0.304902
         f5    0.001457
         Name: e1, dtype: float64
```

```
In [30]: df>0
```

Out[30]:

	f1	f2	f3	f4	f5
e1	True	False	True	True	True
e2	False	False	False	True	False
e3	False	True	True	False	True
e4	False	False	True	False	False

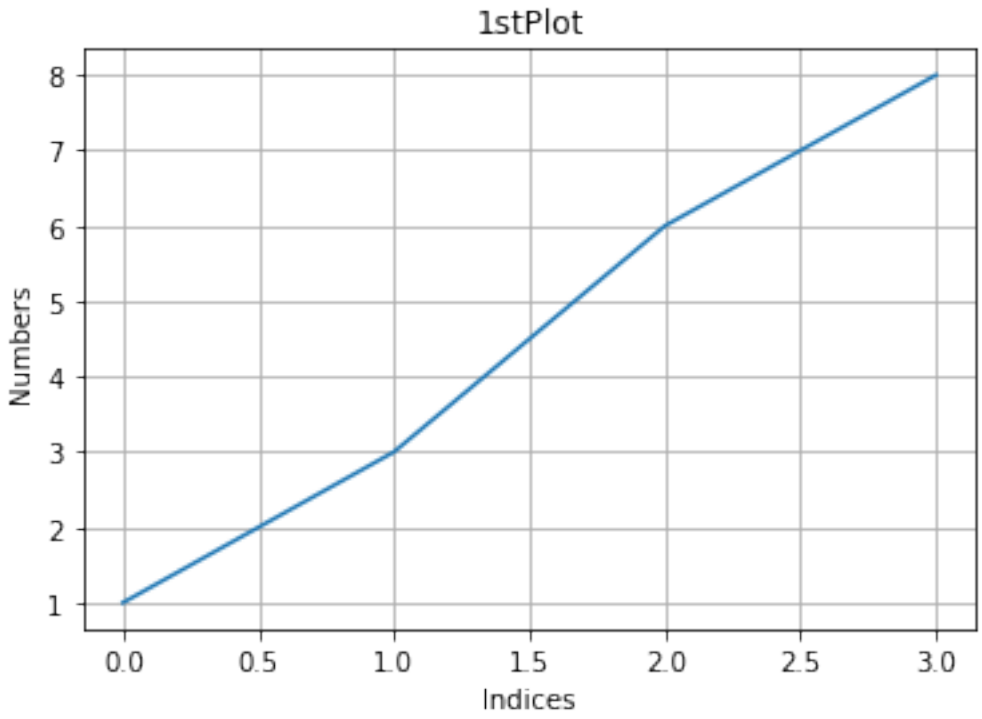
2 matplotlib - Plotting functions

3 Pyplot tutorial

matplotlib.pyplot is a collection of command style functions that make matplotlib work like matlab

```
In [1]: import matplotlib.pyplot as plt
```

```
In [5]: plt.plot([1,3,6,8])
        plt.ylabel("Numbers")
        plt.xlabel("Indices")
        plt.title("1stPlot")
        plt.grid()
        plt.show()
```

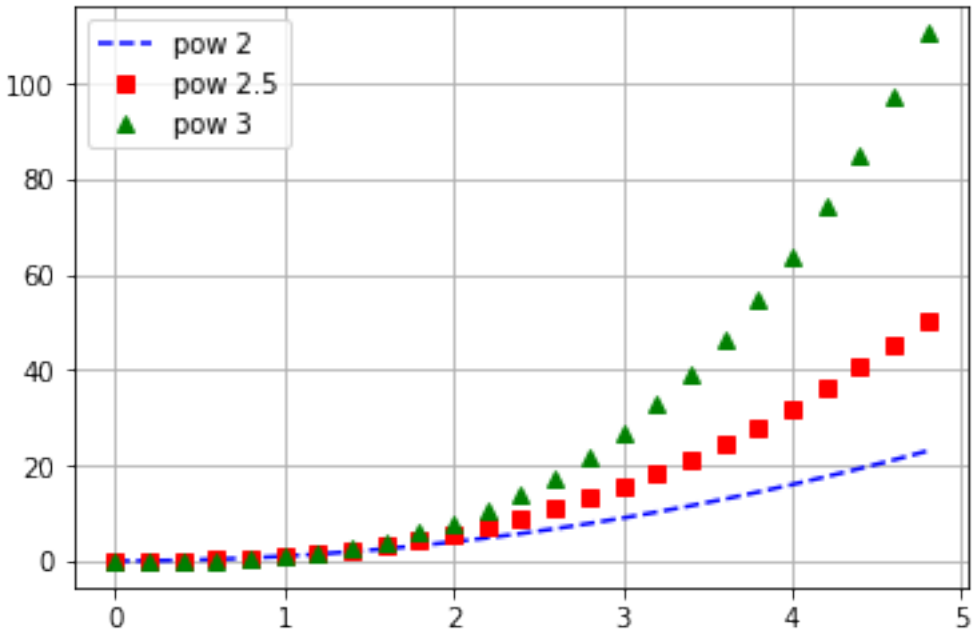


4 Task- plot square numbers where x= numbers & y= squares.

```
In [19]:  # Ans?????
```

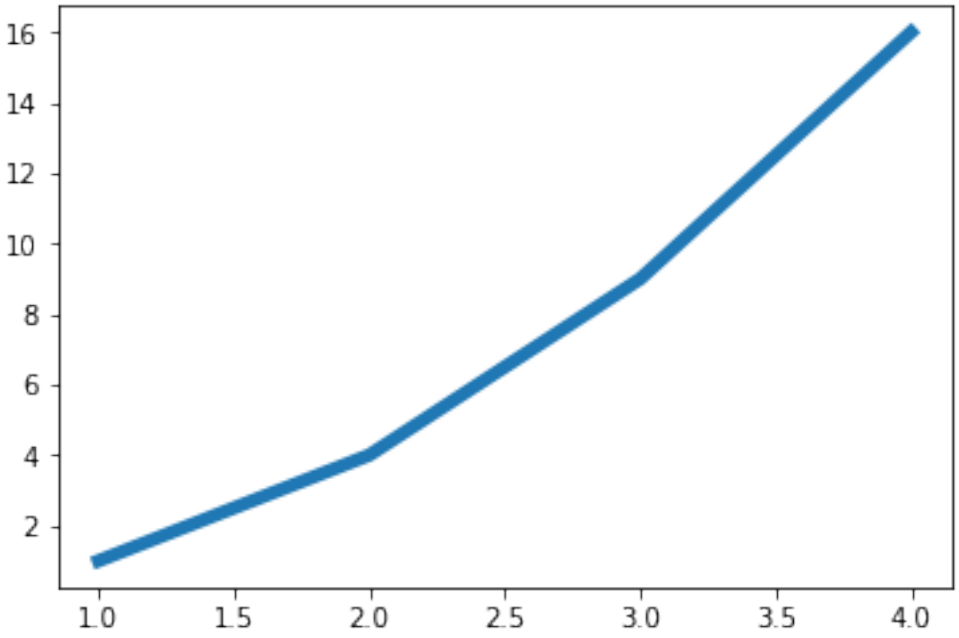
```
In [ ]: import numpy as np
        a = np.arange(0.,5.,0.2)
```

```
In [12]: plt.plot(a, a**2, 'b--', label= 'pow 2' )
         plt.plot(a, a**2.5, 'rs', label= 'pow 2.5' )
         plt.plot(a, a**3, 'g^', label= 'pow 3' )

         plt.grid()
         plt.legend()
         plt.show()
```

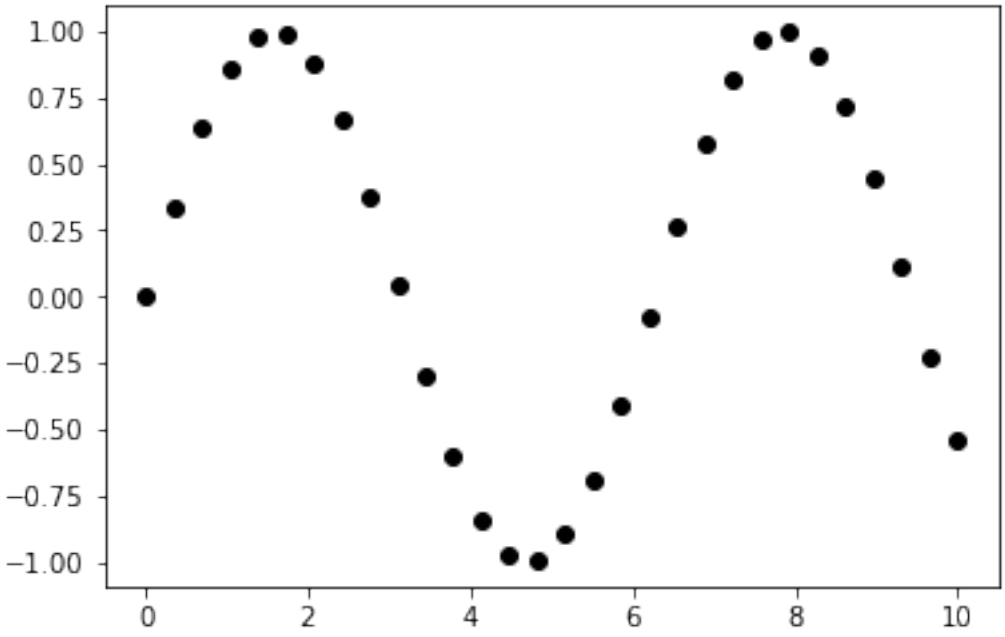


5 line propoerties

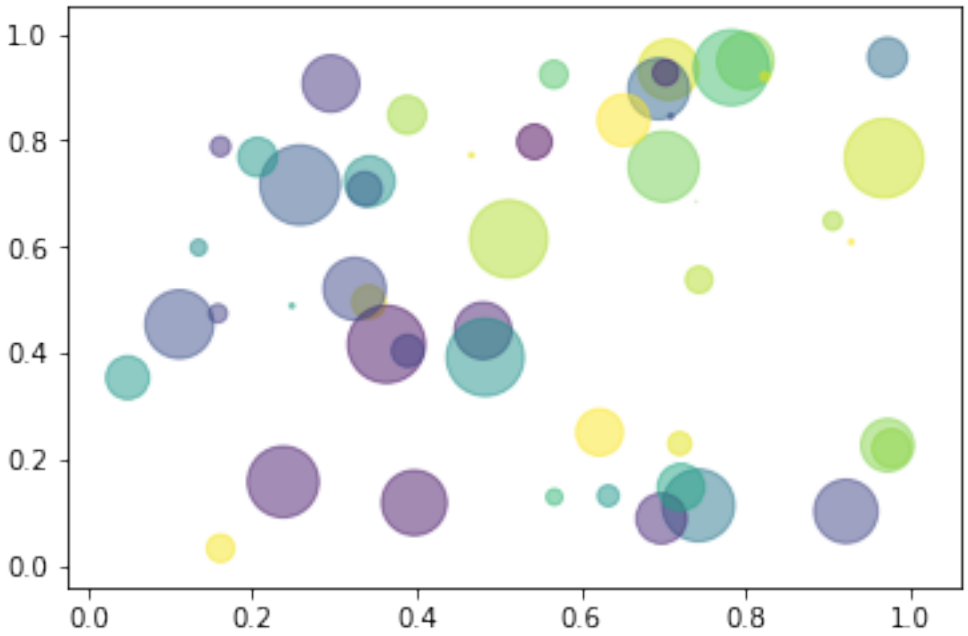
```
In [13]: x =[1,2,3,4]

         y=[1,4,9,16]

         plt.plot(x,y,linewidth=5.0)

         plt.show()
```



In [15]:
```
x = np.linspace(0, 10, 30)
y = np.sin(x)

plt.plot(x, y, 'o', color='black')
plt.show()
```



In [17]:
```
import numpy as np
import matplotlib.pyplot as plt

# Fixing random state for reproducibility
np.random.seed(19680801)


N = 50
x = np.random.rand(N)
y = np.random.rand(N)
colors = np.random.rand(N)
area = (30 * np.random.rand(N))**2  # 0 to 15 point radii

plt.scatter(x, y, s=area, c=colors, alpha=0.5)
plt.show()
```



In [ ]: