



Python 培训第四次

Pandas 初步

By 李亚军

CONTENT

- Intro and Installation
- Object Creation
- Viewing Data
- Selection
- Missing Data
- File IO
- Visualization

Intro and Installation

- Pandas 是一个 Python 数据分析方面的开源库，含有使数据分析工作变得更快更简单的高级数据结构和操作工具，基于 Numpy 构建。
- 和其他第三方包安装方式一样，可以采用 conda 或者 pip 方式安装。

```
$ conda install pandas
```

```
$ pip install pandas
```

Object Creation

- Pandas 中最主要的两个数据结构就是 *Series* 和 *DataFrame*, 前者是类似于一维数组的结构, 而后者是表格类似结构。
- *Series* 和 *DataFrame* 的创建可以有多种方法, 其中 *Series* 可以用列表和字典来创建, 而 *DataFrame* 最多的而是用等长列表或者 Numpy 数组组成的字典。

Object Creation - Series

```
In [4]: ser = Series([4, 7, -5, 3])
```

```
In [5]: ser
```

```
Out[5]:
```

```
0 4
```

```
1 7
```

```
2 -5
```

```
3 3
```

```
dtype: int64
```

```
In [9]: ser2 = Series([4, 7, -5, 3], index=['a', 'b', 'c', 'd'])
```

```
In [10]: ser2
```

```
Out[10]:
```

```
a 4
```

```
b 7
```

```
c -5
```

```
d 3
```

```
dtype: int64
```

```
In [13]: ser3 = Series({'a': 4, 'b': 7, 'c': -5, 'd': 3})
```

Object Creation - DataFrame

使用字典构建 DataFrame

```
In [22]: df = DataFrame({'col1': np.arange(5),  
                        'col2': np.arange(5, 10),  
                        'col3': np.arange(10, 15),  
                        'col4': np.arange(15, 20)})
```

```
In [23]: df
```

```
Out[23]:
```

	col1	col2	col3	col4
0	0	5	10	15
1	1	6	11	16
2	2	7	12	17
3	3	8	13	18
4	4	9	14	19

Object Creation - DataFrame

为 DataFrame 指定行名和列名

```
In [25]: df1 = DataFrame(np.arange(20).reshape((5, 4)),  
                        index=['row1', 'row2', 'row3', 'row4', 'row5'],  
                        columns=['col1', 'col2', 'col3', 'col4'])
```

```
In [26]: df1
```

```
Out[26]:
```

	col1	col2	col3	col4
row1	0	1	2	3
row2	4	5	6	7
row3	8	9	10	11
row4	12	13	14	15
row5	16	17	18	19

Viewing Data

- 我们可以使用一些基本函数来查看一个 DataFrame 的数据。

函数	说明
<code>df.head()</code>	查看 df 的前 n 行，默认前 5 行
<code>df.tail()</code>	查看 df 的后 n 行，默认后 5 行
<code>df.index</code>	查看 df 的索引
<code>df.columns</code>	查看 df 的列名
<code>df.values</code>	查看 df 的值

Selection

- Series 的值选取方法类似于 Numpy 数组的选取，只不过 Series 可以根据 index 来索引，而 index 不一定是整数。

```
In [29]: ser = Series(np.arange(4), index=list('abcd'))
```

```
In [30]: ser
```

```
Out[30]:
```

```
a 0
```

```
b 1
```

```
c 2
```

```
d 3
```

```
dtype: int32
```

```
In [31]: ser['c'] # ser[2]
```

```
Out[31]: 2
```

```
In [32]: ser['b':'d']
```

```
Out[32]:
```

```
b 1
```

```
c 2
```

```
d 3
```

```
dtype: int32
```

Selection

- DataFrame 的选择可以通过 `.loc`, `.iloc` 和 `.ix` 函数, 也可以使用类似字典选取的方式。

函数

说明

`df[val]`

选取某一列或者多列

`df.loc[]`

依据行名和列名来选取值

`df.iloc[]`

依据位置来选取值

`df.ix[]`

既可依据行名列名, 也可依据位置, 但行名列名优先

Selection

```
In [41]: df['col1']
```

```
Out[41]:
```

```
row1 0
```

```
row2 4
```

```
row3 8
```

```
row4 12
```

```
row5 16
```

```
Name: col1, dtype: int32
```

```
In [47]: df.loc['row2', 'col3']
```

```
Out[47]: 6
```

```
In [49]: df.iloc[1, 2]
```

```
Out[49]: 6
```

```
In [50]: df.ix[1, 2] # df.ix['row2', 'col3']
```

```
Out[50]: 6
```

Missing Data

- Pandas 中有一些方便的处理缺失值的方法。

方法

说明

`df.dropna()`

删除存在缺失值的行或列

`df.fillna()`

用指定值或差值方法填充缺失数据

`df.isnull()`

判断是否是缺失值

`df.notnull()`

判断是否不是缺失值

File IO

- Pandas 提供了很多高级函数来读取和写入数据文件。

方法

说明

`read_csv()`

读取 CSV 文件

`read_excel()`

读取 Excel 文件

`read_table()`

读取带分隔符的文件，默认分隔符为制表符

`read_hdf()`

判断是否不是缺失值

`read_pickle()`

读取 HDF5 数据文件

`read_clipboard()`

读取剪贴板的数据

Visualization

- Pandas 有许多能够利用 DataFrame 对象的数据组织特点来创建图标的高级绘图方法。

方法	说明
<code>plot()</code>	默认绘制折线图
<code>bar()</code>	条形图
<code>scatter()</code>	散点图
<code>hist()</code>	直方图

Visualization

```
In [2]: ts = pd.Series(np.random.randn(1000),  
                        index=pd.date_range('1/1/2000',  
                                           periods=1000))
```

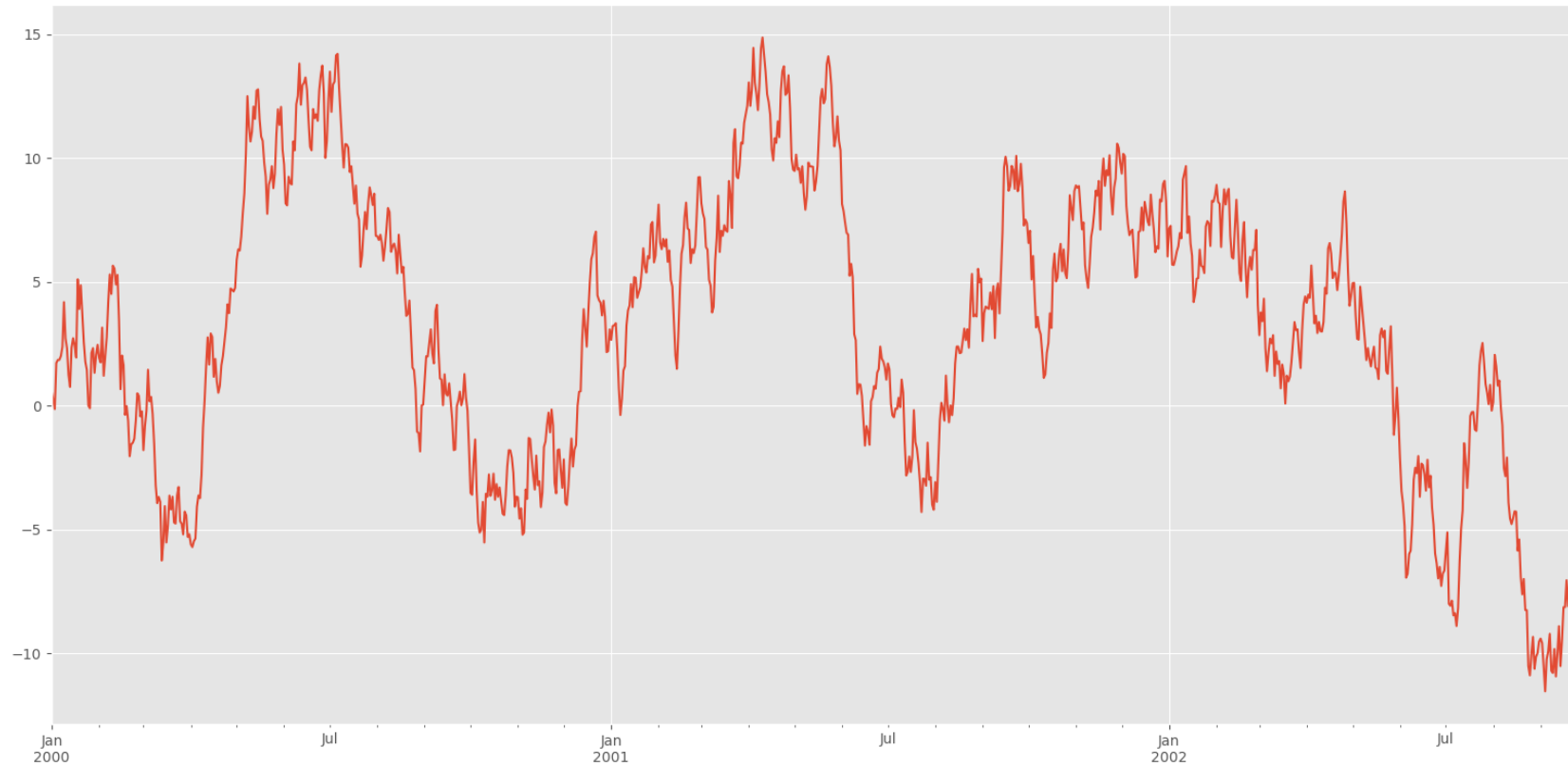
```
In [3]: ts = ts.cumsum()
```

```
In [4]: ts.plot()
```

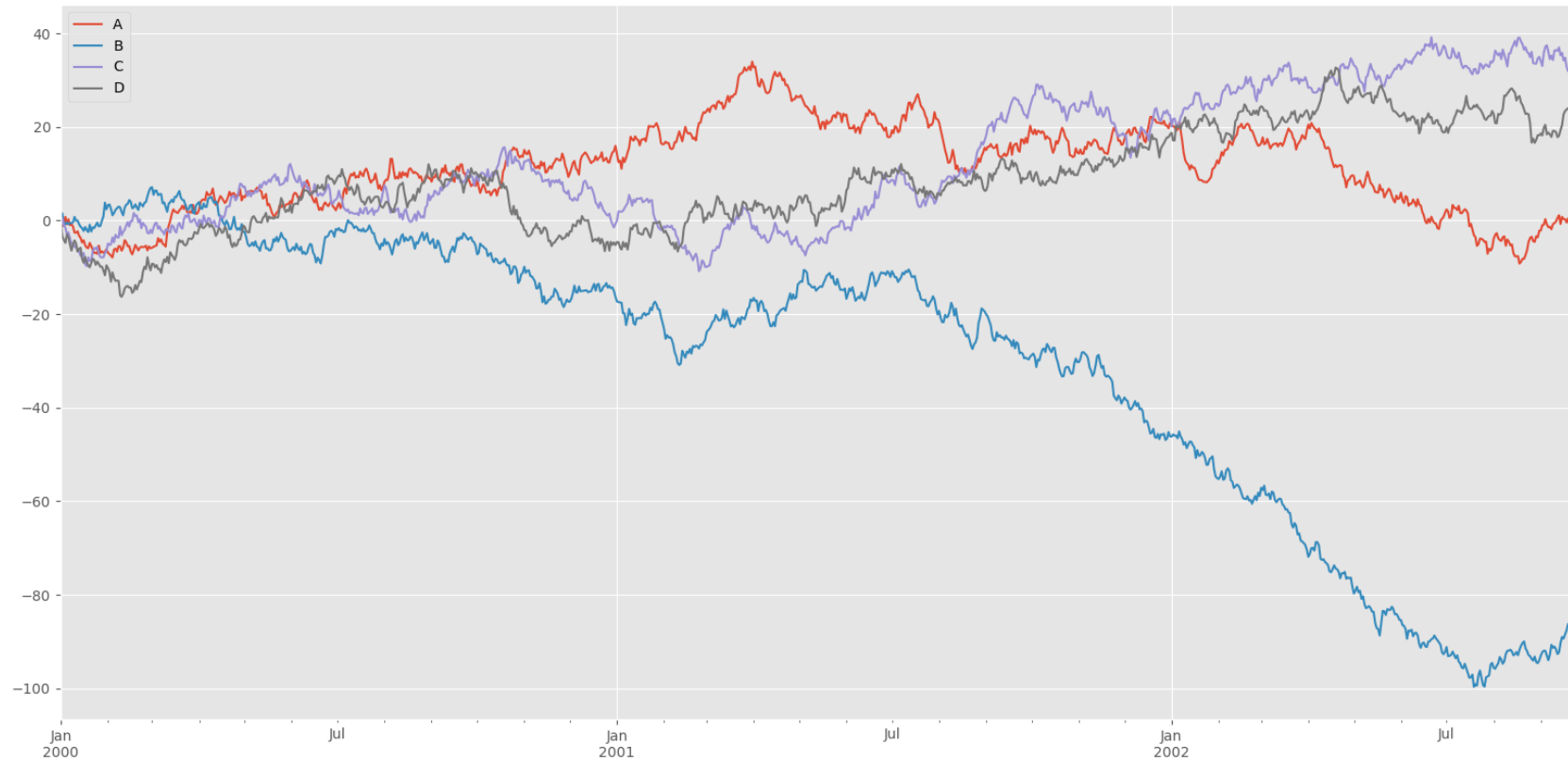
```
Out[4]: <matplotlib.axes._subplots.AxesSubplot at  
0x17cdc8255f8>
```

```
In [5]: plt.show()
```

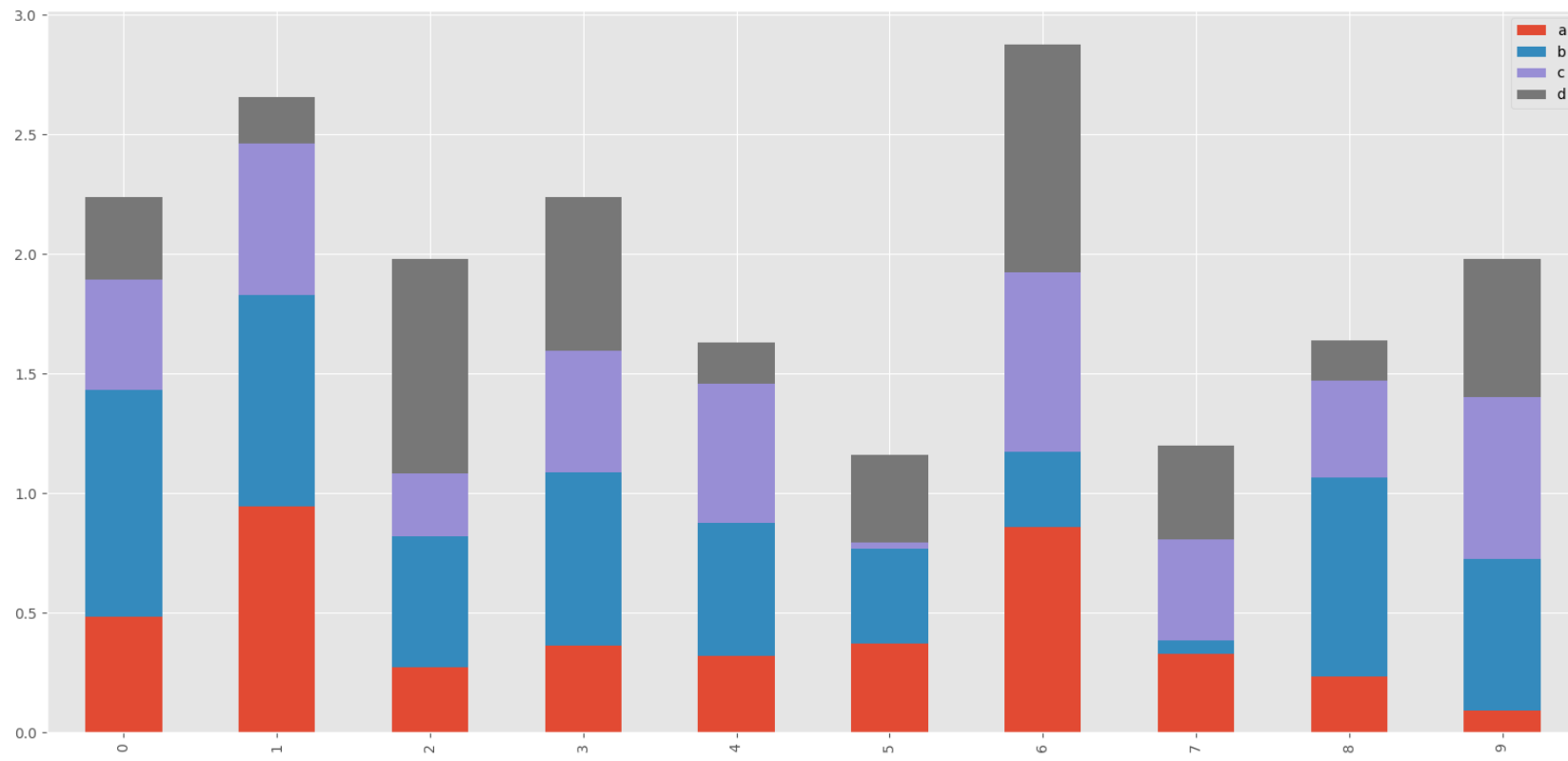
Visualization



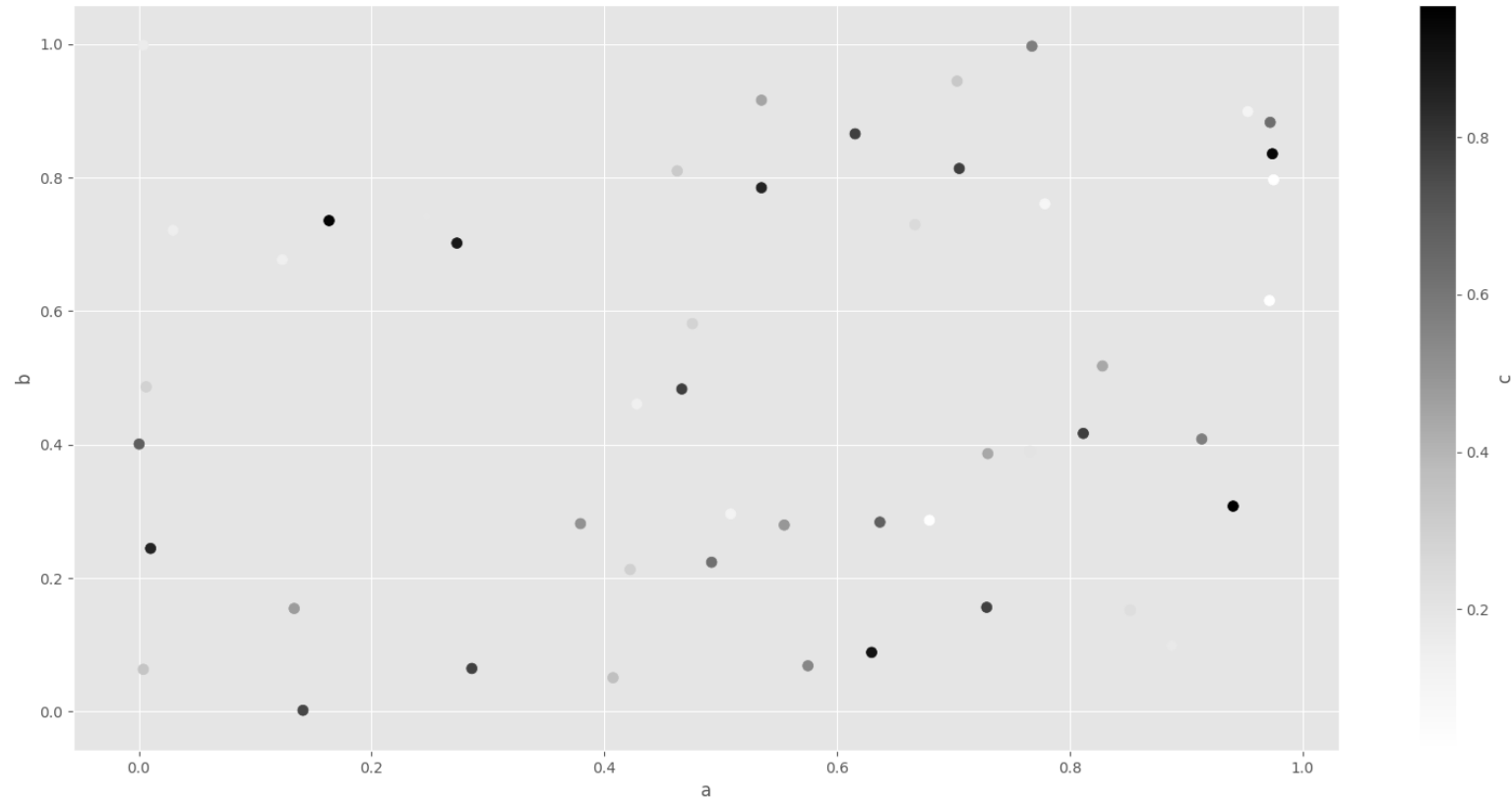
Visualization



Visualization



Visualization



THANK YOU
FOR LISTENING

