老师github：https://github.com/FayeL37/python\_notebook

**Why Python？**

Engineers use Python a lot. Prove that you are “technical”.

python是完整体系，图灵完备的，能完成SQL无法完成的；方便和工程师交流。

Python is highly transferable skill.

Python can be connected with R and SQL.

在Python中可以用R，SQL，ML统计建模库更完备，更新及时。Google Tensor Flow。

**Python at interview**

面试中不要求live coding写python。不用LeetCode刷题。不用记具体函数。

至少know the basics.

Common questions: 面DS很可能问，BS一般不会问

考验你有没有在python中处理过类似的问题

- How to remove duplicates in a table? 有可能是重复的行，也可能是重复的列

- How can you join two tables? 和SQL的join是一样的

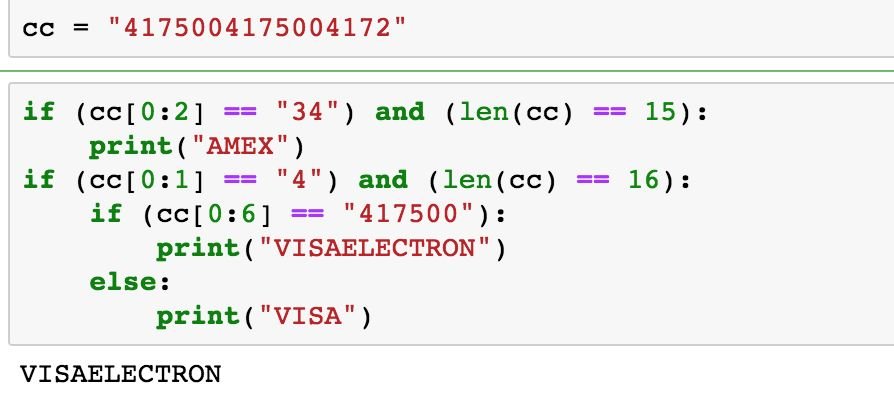
Less common, but are sometimes asked 小公司可能会让你Live coding，界面是能实时跑出结果的

- Coding (loop, if/else)

**Live Coding Question Example:** 用来刷掉没有真正自己写过Python的

“Issuer” of a credit card are companies like Visa, Mastercard, American Express. Now we know the

following rules:



CC number starts with: '4'

CC len : 16

Issuer: VISA

CC number starts with: '34', '37'

CC len : 15

Issuer : AMEX

CC number starts with: '36'

CC len : 14

Issuer : Diners Club

Credit card number starts with

'4026', '417500', '4405', '4508', '4844',

'4913', '4917'

CC len : 16

Issuer : VISAELECTRON

Now write Python scripts to identify the issuer of the following credit card numbers:

'4175004175004172'

'4917491749174917'

'346416800707698'

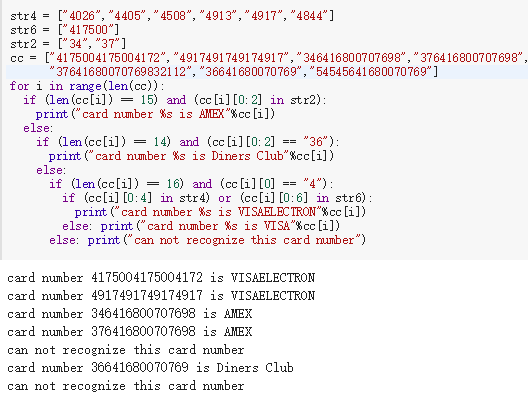
'376416800707698'

'37641680070769832112'

'36641680070769'

'54545641680070769’

一个等于号是赋值，两个等于号是做判断，比较。Python大小写敏感，SQL不分大小写



**Python Notebook**

Coding in terminal is terrible experience.

We need something better (like the RStudio): Python Notebook.

Better UI, cleaner format, allows navigation through codes, and pre-installed with many libraries.

**Github**

Github is a place to store files.

Great place to upload, manage, and share your codes and files.

All you need to do:

● Code in Notebook

● Save your Notebook as .ipynb

● Upload .ipynb onto Github

● Commit

● Create a shareable link, so people love you.

Python materials

If you are serious, you can also learn Python systematically on the internet.

● Udacity 免费的课，很有用，很好懂

● W3 School Python从零教起

● Google developers google edu 稍微technical

**Basic data types** 基本数据类型

● Numbers 包括小数和整数

● Strings 要用单引号或双引号括起来

**Strings**

At interview: Important.

In reality, you spend more than you think on “manipulating” strings.

● Length: len() 求字符串长度

● Concatenation: + 把两个字符串合在一起

● Coercion: str() 把括号中的内容强制转换成字符串格式

● Visit a character: []. 0-index 中括号中指定要访问的字符的位置是第几位 0表示第一位

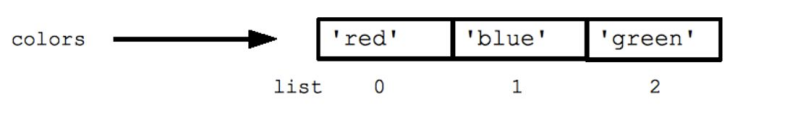
-1表示最后一位

● Manipulation. Don’t have to memorize, but remember that they are here. 常见操作，面试不太考

○ s.lower(), s.upper() 全部换成小写/大写

○ s.find('other') 在一个字符串中寻找这个子字符串，如果有，返回开始的索引值

○ s.replace('old', 'new') 把字符串中的 old（旧字符串）替换成 new(新字符串)



**Lists** 列表中有很多元素，按顺序排列 用[]圈起来

The most commonly used data structure.最重要、基本的数据结构

Don’t we also have list in R?

● 0-index. R is 1-index.

● Common manipulation:

○ list.append(elem) 在列表的结尾加一个元素 不用知道list中间的信息 更高效

○ list.insert(index, elem) 指定 要把新的元素插在第几个位置上 低效一些 还要从前往后数，插到相应位置，把后面的元素依次后移，比较昂贵

○ list.pop(index) 把某个位置的元素崩出去，不传参数的话会把最后一个元素崩出去

○ list.index(elem) 在list中找给定元素，如果有，返回它的第一次出现的位置

○ list.sort() 把list中的元素按某种顺序重新排列，参数决定升序/降序，默认（按字母顺序）升序

求list中有多少个元素：len()

**Control flow**

● If/else 如果条件成立，做。。。，如果条件不成立，做。。。 见第一道练习题

● Loop 循环。 SQL中默认把表格的每一行依次过一遍，隐含了循环。因为要提取数据的话默认要把每一行都要扫一遍。

**Notice**

● Like R, Python is case sensitive so "a" and "A" are different variables.

● Like R, the end of a line marks the end of a statement. Python does not require a semicolon（分号）at the end of each statement.

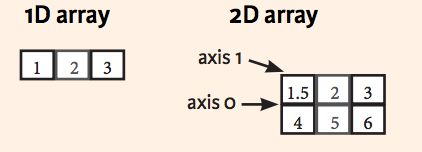
● Like R, comments begin with a '#' and extend to the end of the line.

● Unlike R, indentation（缩进）is very important for Python 一般不同（相邻）逻辑层之间要空两格（按一下tab键）

**数据科学三大库**

1. **Numpy Arrays** 处理数组的库：新的数据结构：array

有很多现成的function来处理数组，是list所不具有的

At interview: Important.

● Exactly like R arrays

● 1D array is like a list, but you can do more.有自己的一套操作

● 2D array is like a table. Use Pandas instead.表格类用pandas

● 3D array you will never use.几乎用不到

np.array()函数的作用就是构造一个新的numpy array

my\_array = np.array([1, 2, 3, 4])

打印出来为了和list区分开，会显示不同于list的格式：array([1,2,3,4])

len(my\_array)

np.shape(my\_array)

my\_array.sum()

my\_array.max()

my\_array.mean()

# deduplicate**去掉重复元素**（在list中只能循环一个一个找）

my\_array = np.array([1, 2, 2, 4, 5])

**np.unique(my\_array)**

# other stuff: # subset

my\_array[-1] 取数组最后一位

my\_array[my\_array>2] 取出数组中满足条件的元素

# add/remove element

np.append(my\_array, 6) 在数组最后贴上该元素

np.insert(my\_array, 9,3) #(array,index,element) 在9号位置插入元素3，后面的元素依次后移

np.delete(my\_array, 1) 删除某个位置的元素 ，1可以带[]也可以不带

list和array和tuple的区别：

list是python基本数据类型，它的元素类型可以不同，如：[1, 2, 'a', 3,4]

array是numpy的一种数据类型，它所包含的元素必须相同，如: [1, 2, 3, 4]

列表会使用更多的存储空间相对于array。

Python中的列表(list)类似于C#中的可变数组（ArrayList），用于顺序存储结构。它可以方便、高效的的添加删除元素，并且列表中的元素可以是多种类型。

列表很多操作都跟元组一样，它们的不同在于元组是只读的，那更新列表的操作，比如切片操作来更新一部分元素的操作，就不能用于元组。可以用list的 [],:操作符提取元素。就是不能直接修改元素。

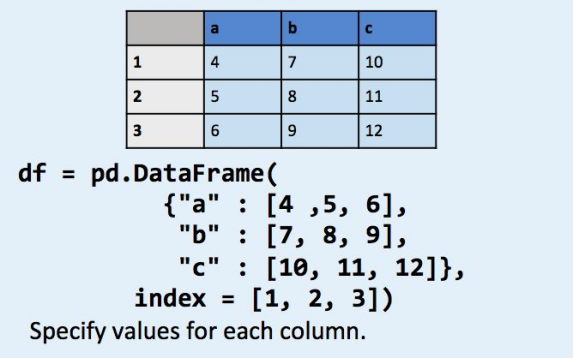
1. **Pandas Dataframe** 专门处理表格的工具

Pandas 库中基本的数据结构：Dataframe，有行有列，就是表格。可以做几乎所有SQL能做的，SQL不能做的也能做。

At interview: Very important.

● Basically a SQL table.

● Very similar to R.

**Basic operations** of Pandas Dataframe: Column Axis = 1

● Shape

● Subsetting Index(Row #)

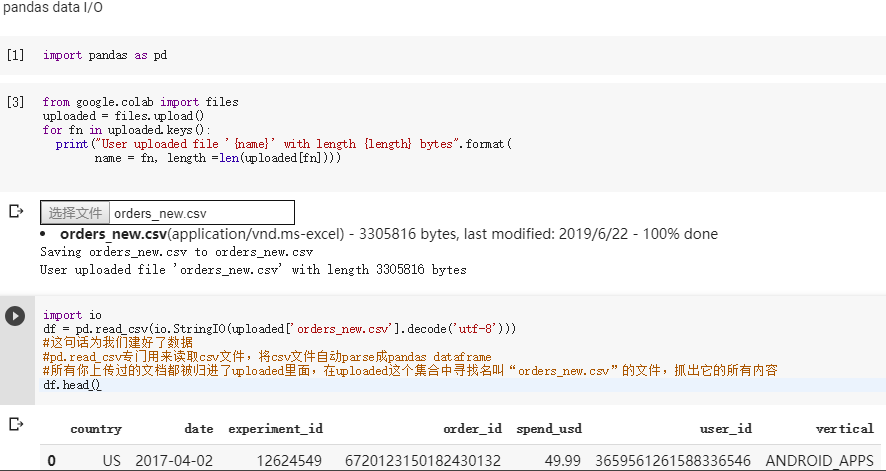
○ Columns 相当于SQL中primary key

○ Rows Axis = 0

● Create/Remove columns

Index即Unique identifier,每行要有自己独有的标号





**SQL-like operations**

○ Lots of SQL-like operations

○ can actually run sql queries in Python, too

● SQL-like operations:

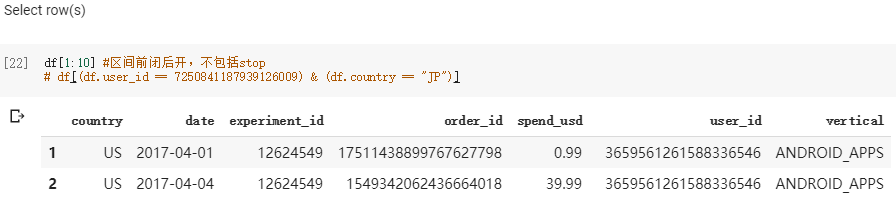
○ Primary key: index()

○ Subset on columns

○ Subset on rows

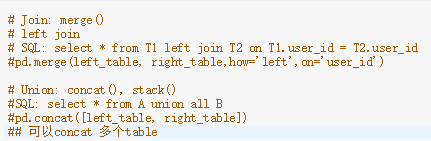




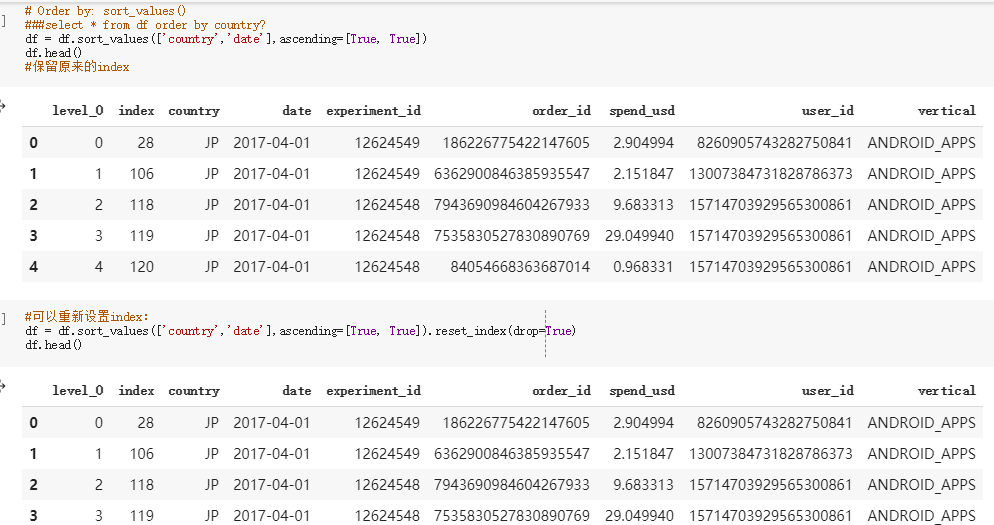


○ Join: merge()

○ Union: concat()



○ Order by: sort\_values()



○ Groupby: groupby()



1. **Matplotlib:Data Visualization**

● We first build a quick and dirty chart.

● Then we look at some tips to make it look cool.

* What chart to use?

“I chose this chart for my project, because I was essentially dealing with a comparison/composition/distribution/relationship

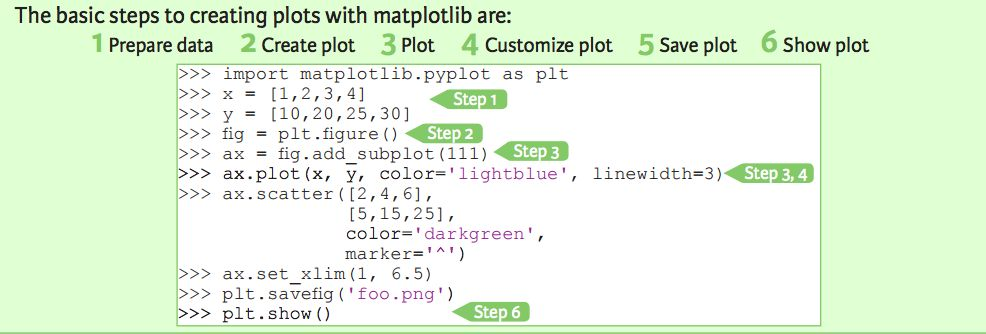
visualization problem”. -- a cool interviewee

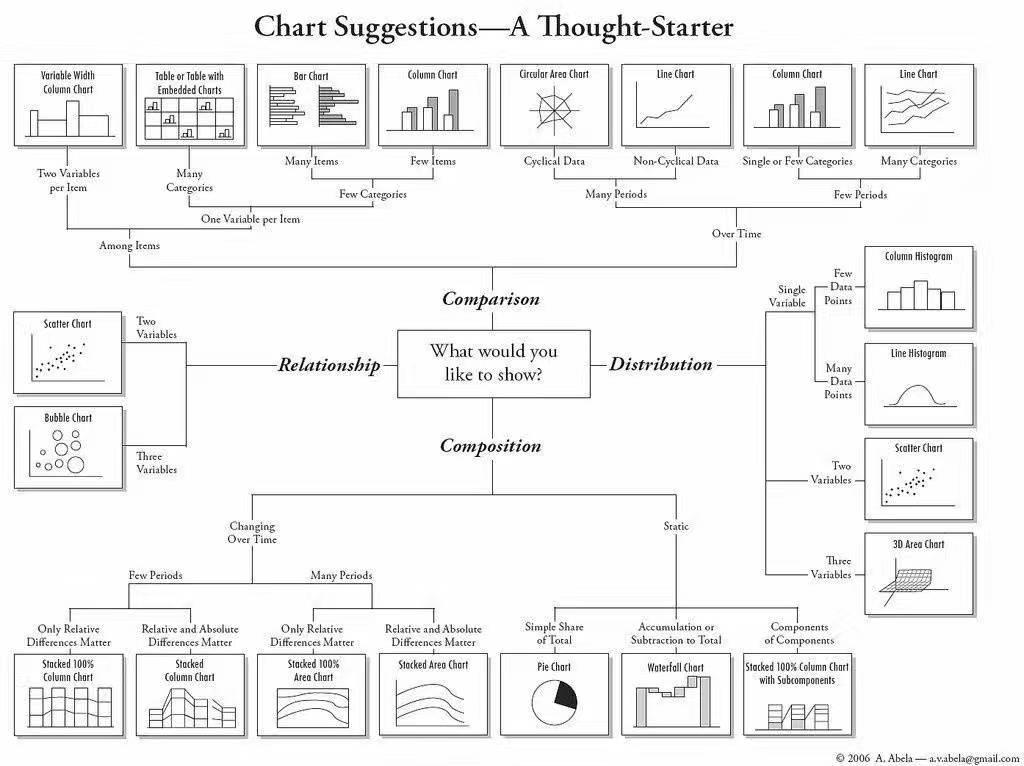
Try them out in these tools:

- Google spreadsheet (Excel) - ggplot (R) - Matplotlib (Python) - D3 gallery (Javascript)

* Matplotlib: basic steps to create a plot

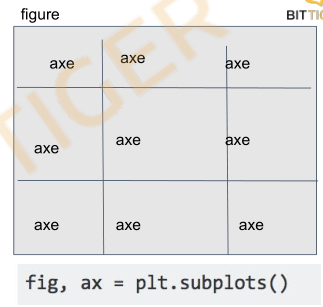
1. Prepare data 2. Create figure 3. Create plot 4. Draw plot





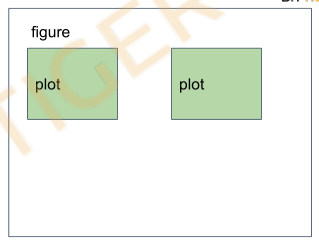
Waterfall chart:把每一变化步骤，增增减减，体现在图上。描述中间变化过程的好方法。

1. prepare data 将数据准备成pandas data-frame格式
2. Create figure 创建figure，可理解为白板，将在它上面画东西
3. Create plot （Paint plot）相当于在白板上切块（先画几个框-ax），定义好要画几个图，上面几个下面几个



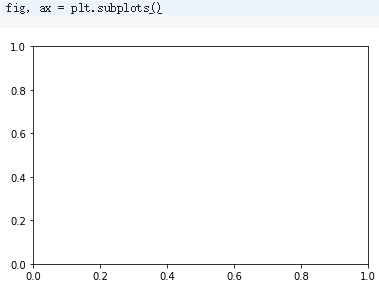
每个ax里要花一个图

生成一个画布figure，默认只生成一个ax（subplot）



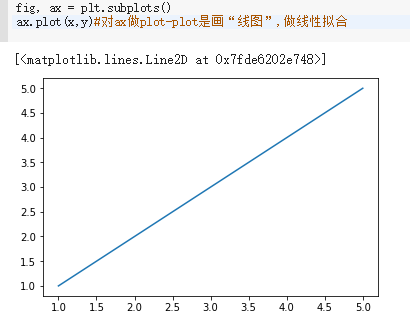
1. customize plot 在前面定义好的框（plot）里（依次）画内容

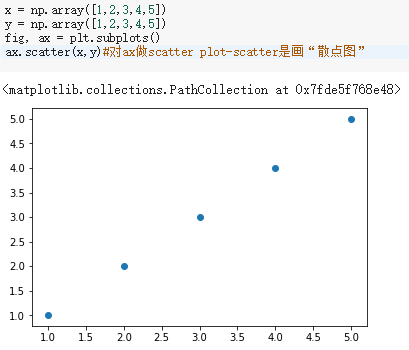
Matplotlib提供接口来个性化调整角标颜色等



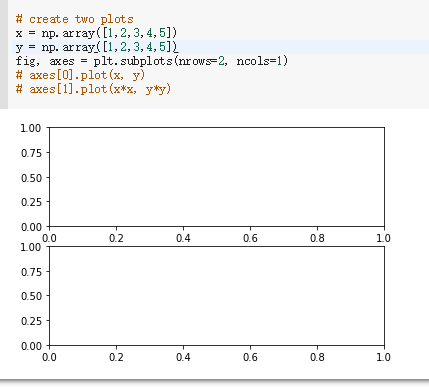
x = np.array([1,2,3,4,5])

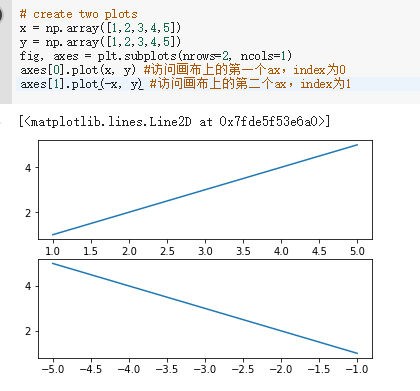
y = np.array([1,2,3,4,5])

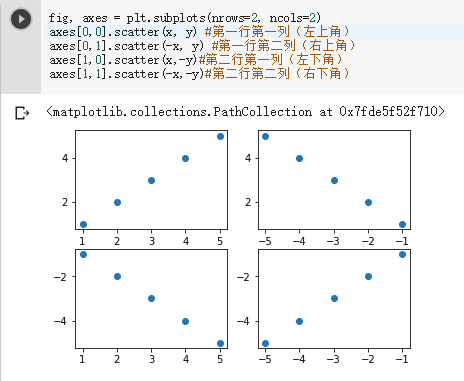




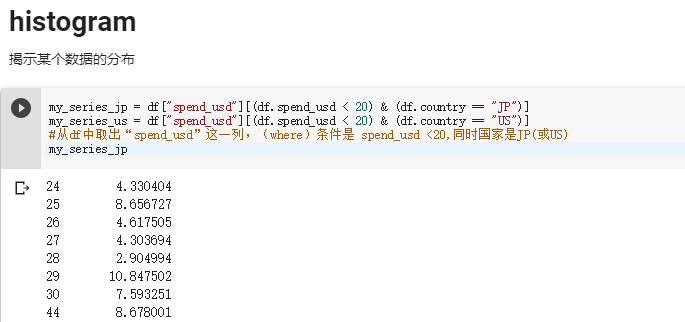
画两个图，上下排列（两行一列）

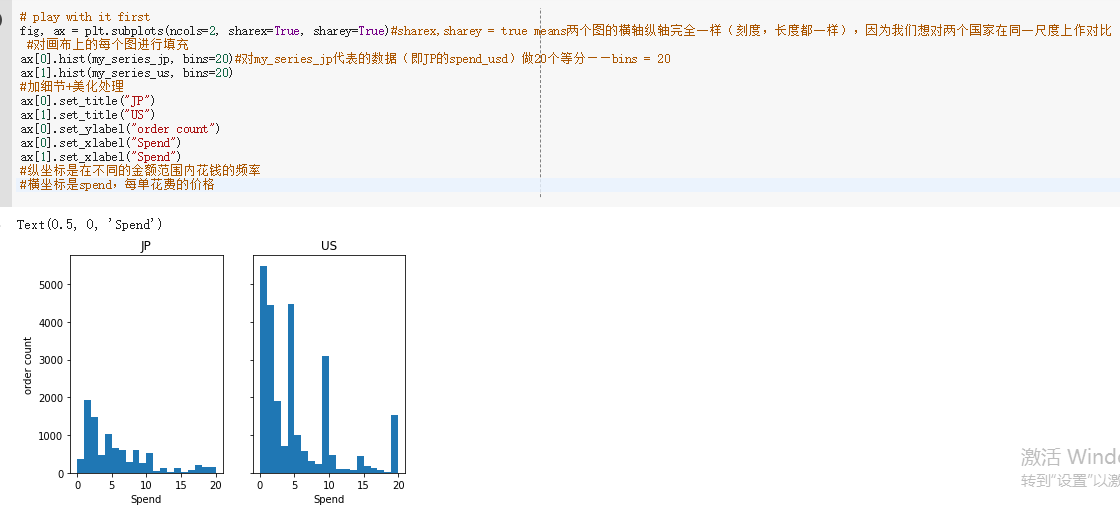


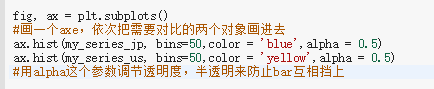


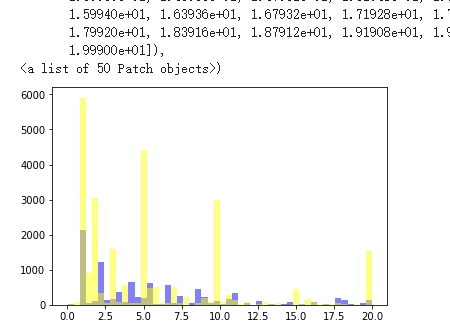


真实数据：画histogram









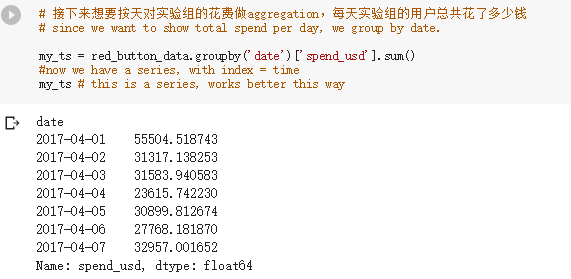


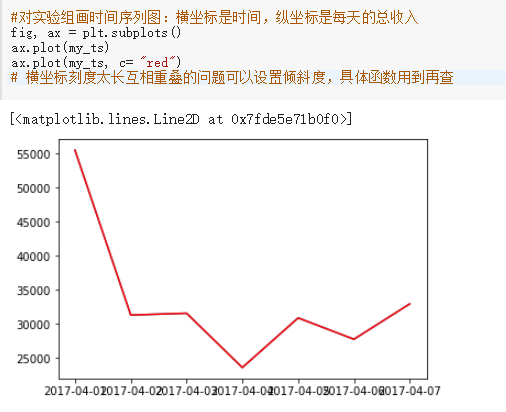
如果不人为设置图的横纵坐标最大值，机器会自动检测最大值最小值，留一些空余，自动选取一个标尺

如果想人为设定，在plt.hist中加range：plt.hist(...,...,range = [下限，上限],...,)

bins描述了颗粒的数量，不代表横轴最大值，最大值由数据最大值决定或人为定义range，bins表示整个hist上有多少个块，颗粒数量越多数据描述越细致，颗粒数量越少每个颗粒越粗大，损失的数据细节越多。







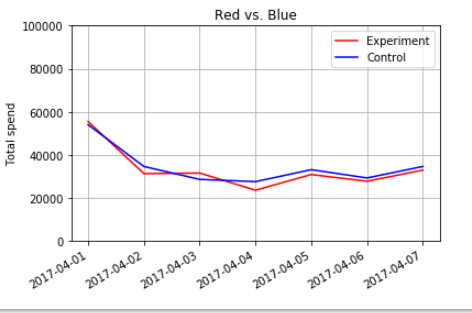


给老大看时不想给她看太剧烈的变化，看全局而不太微观，就不用默认的范围（数据最小值，数据最大值）

人为自动设置范围

标网格：ax.grid(True,which = “both”)





Python功能强大，可能替代R，但还是无法替代SQL，SQL只专注做一件事，就是数据库query。在提取数据这方面还是SQL能做。比如提取google search data：一个用户翻页面花多少秒，搜索了什么，点击之后又点击了什么，记录这一系列会产生大量数据，python做不了，还是要用SQL提取数据。

库：scikit-learn——machine learning in python

R可以做的它都可以做到，比如Classification, Regression, Clustering, Dimensionality reduction, Model Selection, Preprocessing

Python做测试也很方便，网站，服务器，都可以用这个写

另一个库： plotly 画图神器 interactive chart,提供了很多很多图表的类型。可以直接在里面做slides

Dashboards&reports

实时更新，数据可动

* Python Functions

Goal of a function is to reduce redundancy. So far we have done written quite a few codes -let’s try wrap them into a function.

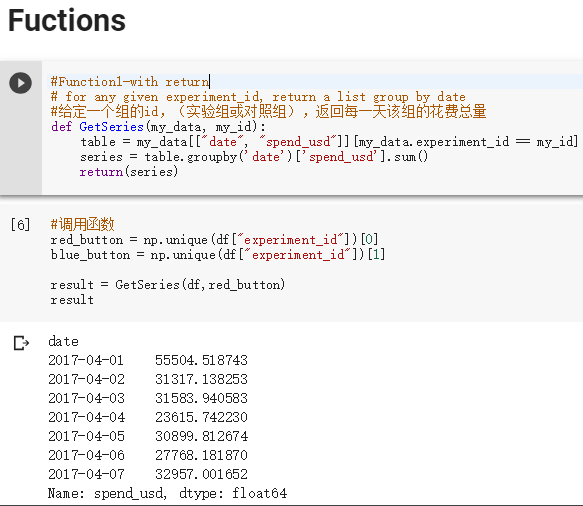
**Write function for everything**

-Return an output:

def function\_name\_1(x, y):

z = do\_something(x, y)

return(z)



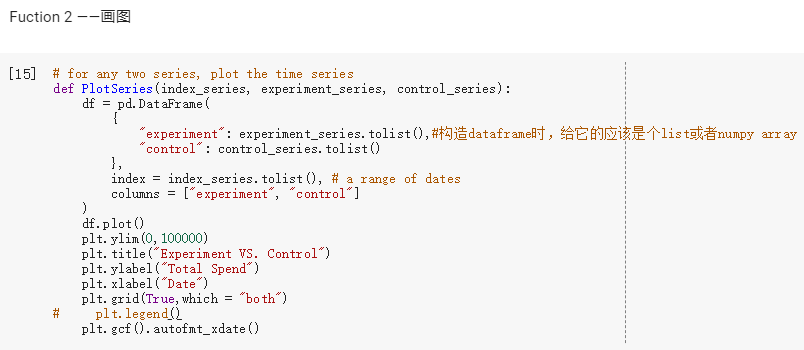
-No output:

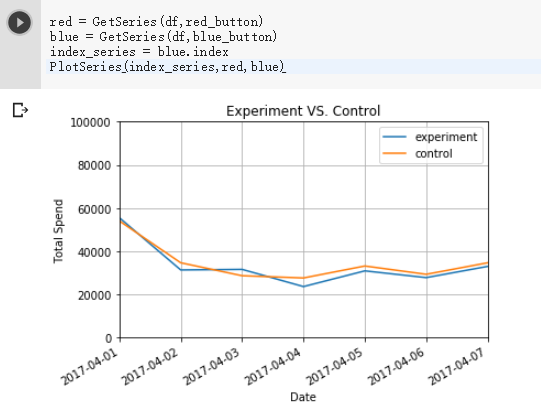
def function\_name\_2(x, y):

do\_something(x, y)

print()

plot()





-Calling a function:

result = function\_name\_1(5, “yes”)

function\_name\_2(5, “yes”)

**Write our first function:**

PrintDailyTimeSeries(3243248,2324249)

Input:

experiment\_id

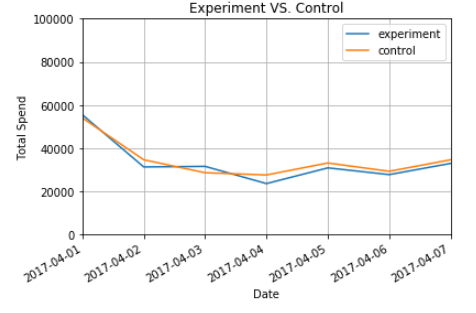
Output:

Time series chart

How to write?

The function should do two things: First, takes in parameters and prepares data. Second, plot the chart.



8

**Homework: PrintCumulativeTimeSeries()**

Input: Three parameters: experiment\_id, country, and

Vertical Output: - Time series chart showing the uplift - Overall % uplift on revenue, experiment vs. control: (red - blue) / blue %

Non-cumulative cumulative

