

# AI解密——体育中的数据科学与人工智能

Lecture 02

Application of data science

虞思逸

# Previous on AI course....

- 数据科学
  - 关于数据
  - 什么是数据科学
  - 什么是数据思维
- 人工智能
  - 什么是人工智能？

数据科学

# 定义

- 利用**数据学习**知识的学科
- 通过从数据中提取出有价值的部分来生产数据产品
  - 应用数学
  - 统计
  - 模式识别
  - 机器学习
  - 数据可视化
- 运用各种相关的数据来帮助非专业人士理解问题、进行研究、给予解决方案支持

人工智能

# 定义

- **人工智能（Artificial Intelligence）** 是研究、开发用于模拟、延伸和扩展人的智能的理论、方法、技术及应用系统的一门新的技术科学。
- AI的核心问题包括建构能够跟人类类似甚至超卓的**推理、知识、规划、学习、交流、感知、移物、使用工具和操控机械**的能力等
  - 机器人
  - 语言识别
  - 图像识别
  - 自然语言处理
  - 专家系统等

# Outline

- 人工智能的应用方向
- 棒球
- 棒球中的数据科学与人工智能
- 棒球数据分析实践
  - Python环境
  - Python入门

# 应用方向

- 感知能力（**Perception**）
- 认知能力（**Cognition**）
- 创造力（**Creativity**）
- 智能（**Wisdom**）



# 感知能力

- “看”：
  - 电脑视觉、图像识别、人脸识别、对象侦测。
- “听”：
  - 语音识别。
- “读”：
  - 自然语言处理、语音转换文本。
- “写”：
  - 机器翻译。
- “说”：
  - 语音生成、文本转换语音。

# 认知能力

- 分析识别能力

- 医学图像分析、产品推荐、垃圾邮件识别、法律案件分析、犯罪侦测、信用风险分析、消费行为分析等。

- 预测能力

- 例如AI运行的预防性维修（Predictive Maintenance）、智能自然灾害预测与防治。

- 判断能力

- AI下围棋、自动驾驶车、健保诈欺判断、癌症判断等。

- 学习能力

- 机器学习、深度学习、增强式学习等等各种学习方法。

# 创造力

- 指的是人类产生新思想，新发现，新方法，新理论，新设计，创造新事物的能力
- 结合知识、智力、能力、个性及潜意识等各种因素优化而成
- 主要领域包括：AI作曲、AI作诗、AI小说、AI绘画、AI设计等。
- AI作诗：<https://www.227g.com:8080/shi/>

# 智能

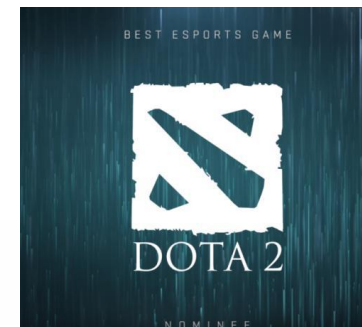
- 指的是人类深刻了解人、事、物的真相，能探求真实真理、明辨是非，指导人类可以过着有意义生活的一种能力，这个领域牵涉人类自我意识、自我认知与价值观，
- 是目前AI尚未触及的一部分，也是人类最难以模仿的一个领域。

# 具体应用领域：知识工程

- 以知识本身为处理对象，研究如何运用人工智能和软件技术，设计、构造和维护知识系统
  - 专家系统
  - 智能搜索引擎
  - 计算机视觉和图像处理
  - 机器翻译和自然语言理解
  - 数据挖掘和知识发现

# Discussion

- 你热爱的体育比赛项目是什么？
- 如果你是教练/选手/粉丝，你最想知道的数据是什么？



1861 1865 1920 1925 1955 1960

### 书籍 / 远见

比尔·詹姆斯出版了他的第一本《棒球摘要》。这本书里收录了很多文章，内容着重于解答棒球数据统计方面的诸多问题。不久，基于美国棒球研究协会（1971 年成立）制定的缩写规则，詹姆斯规范定义了棒球统计学的内容。詹姆斯被认为是棒球数据统计界的「一哥」，并且在世界体育行业内，他也是一位卓有影响力的思想家。

### 创新

BBC 发布了 Ceefax，一种能在模拟信号电视上传输文字的技术；它能够在电视屏幕上发布新闻和体育赛事信息。随后，Ceefax 成为英国「前互联网」时代人们获取体育赛事比分和新闻的首要手段。

### 创新

墨西哥城奥运会上，全自动计时系统首次亮相。这套系统由发令枪触发，精度达到了百分之一秒。这套系统后来成为了径项测速的标准配置。

1980

1975

1970

1965

### 创新

澳大利亚工程师詹姆斯·霍利率先开发出了赛车中的数据遥测技术，对赛车比赛过程中的车辆状况监测技术产生了革命性的影响。

### 创新

Cricinfo 上线运营。在 1996 年，它是第一个实现逐球报道的网站，并于 2007 年被 ESPN 收购。它于 2000 年开设了 statsguru 分支，该分支为所有网站用户提供全面的搜索工具，用户可以免费获取与球员和球队相关的所有类别数据。

1985

1990

1995

### 创新

克莱夫·伍德沃德在特维克纳姆体育场安装了 Prozone 系统。四年后，英格兰赢得了橄榄球世界杯。

### 公司

Prozone 公司成立。德比郡成为第一家采用 Prozone 的球员跟踪技术的俱乐部。曼联在 1999 年也与 Prozone 签约合作。

### 公司

体育数据公司 OPTA 成立，不久便获得了 SkySports 的注意。后者利用 OPTA Index 来衡量球员在比赛中的贡献（包括传球、射门、铲断等）。

### 球员任用

比利·比恩成为奥克兰运动家队的主教练。他利用数据去发掘被低估的球员，这在球员招募领域掀起了一场革命。随后，奥克兰运动家连续四年进入季后赛。

### 创新

PGA 巡回赛开始采用 ShortLink 系统，用于跟踪高尔夫球赛中每次击打的位置与结果信息。

### 书籍

迈克尔·刘易斯所著《点球

### 球员任用

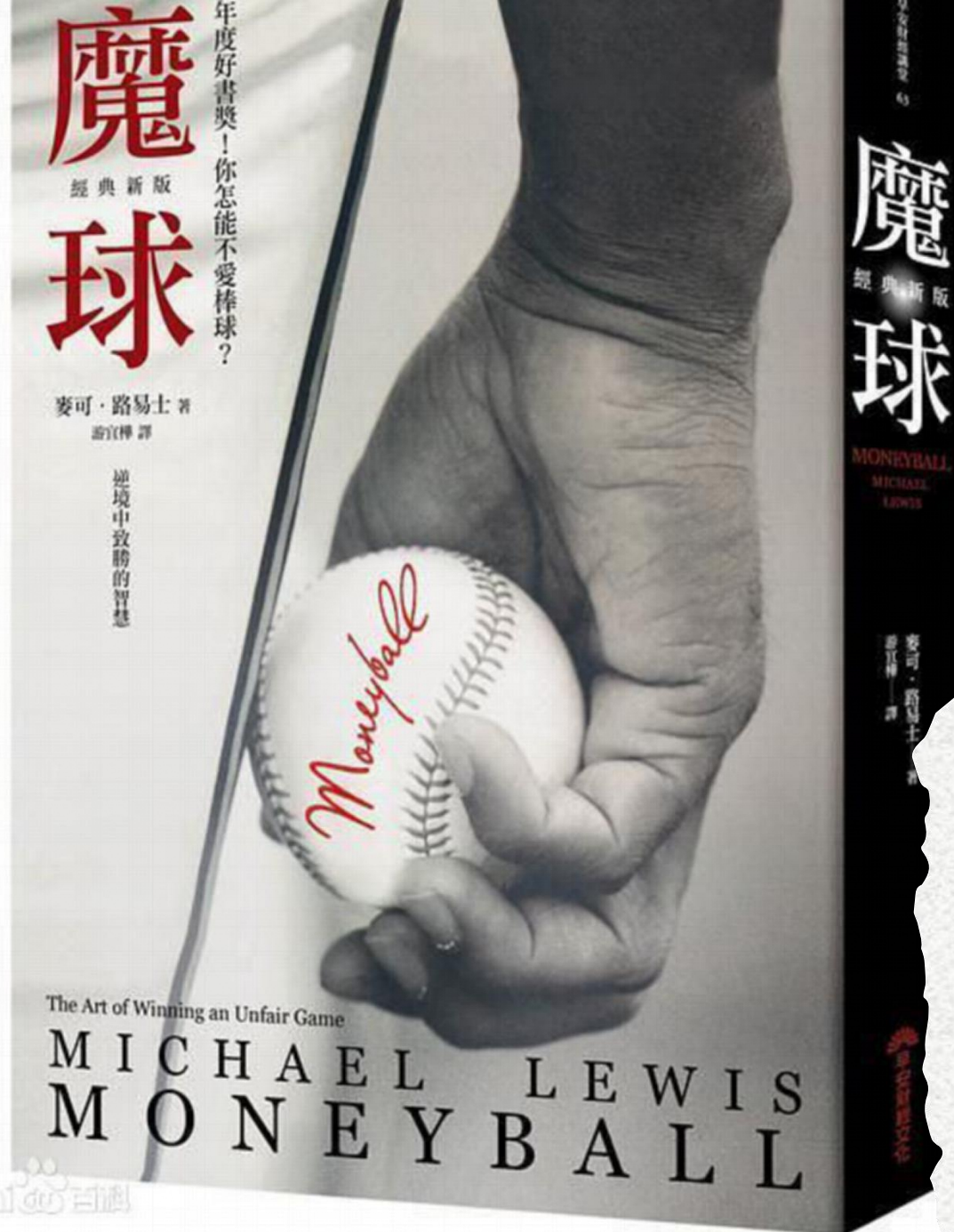
比尔·贝利奇克成为新英格兰爱国者队主教练。作为利用分析工具来评估球员招募和比赛策略的狂热拥护者，新英格兰爱国者成为了 20 世纪 NFL 最成功的球队，在贝利奇克治下四夺超级碗。

2005

2000

# 棒球的故事





## WHO:

- 球队经理比利·宾恩
- 耶鲁大学经济学硕士彼得

## HOW:

- 数据分析
- 数学建模

## 魔球定律:

- 遵循低买高卖的原则
- 选球员只看客观的数据，尤其是长期积累、经过精确计算的数据，得以低价挑选的球员。
- 善于挖掘别人看不上眼的明日之星，从一些不知名的球员中挖掘出明星。
- 善于球星的再生，一些受伤或者处于低潮期的明星，在低价加盟运动家队后都获得了再生。

# 棒球中的可能应用？

有哪些人工智能和数据科学的可能应用？

- Henry Chadwick
- 1976

| BOSTON.  |    |     |     |    |    |   | ATHLETIC.                   |    |     |     |    |    |   |
|--|----|-----|-----|----|----|---|-----------------------------|----|-----|-----|----|----|---|
| T.   | R. | 1B. | PO. | A. | E. |   | T.                          | R. | 1B. | PO. | A. | E. |   |
| G. Wright, s. s  | 6  | 4   | 4   | 1  | 5  | 2 | Force, s. s...              | 5  | 1   | 2   | 1  | 3  | 2 |
| Leonard, 2 b.  | 6  | 3   | 3   | 4  | 4  | 3 | Eggler, c. f..              | 5  | 3   | 3   | 0  | 0  | 0 |
| O'Rourke, 1b   | 6  | 2   | 3   | 9  | 0  | 1 | Fisler, r. f...             | 5  | 0   | 1   | 2  | 0  | 0 |
| Murnan, l. f.  | 6  | 1   | 0   | 3  | 1  | 0 | Meyerle, 3db                | 5  | 1   | 2   | 2  | 3  | 3 |
| Schafer, 2d b  | 6  | 3   | 3   | 3  | 1  | 2 | Sutton, 1st b.              | 5  | 1   | 2   | 10 | 0  | 0 |
| McGinley, c. f   | 6  | 0   | 0   | 0  | 0  | 1 | Coons, c....                | 5  | 1   | 0   | 1  | 1  | 3 |
| Manning, r. f.   | 6  | 0   | 2   | 2  | 0  | 0 | Hall, l. f....              | 5  | 1   | 3   | 5  | 0  | 0 |
| Morrill, c....   | 6  | 2   | 2   | 4  | 1  | 2 | Fowser, 2d b.               | 6  | 1   | 2   | 6  | 7  | 5 |
| Josephs, p..   | 5  | 4   | 4   | 1  | 1  | 2 | Knight, p...                | 5  | 2   | 2   | 0  | 1  | 2 |
| Totals....53 19 21 27 13 13                                |    |     |     |    |    |   | Totals....46 11 17 27 15 15 |    |     |     |    |    |   |
| Boston.....0 1 3 3 4 1                                     |    |     |     |    |    |   | Boston.....0 2 5—19         |    |     |     |    |    |   |
| Athletic.....1 0 0 0 3 3                                   |    |     |     |    |    |   | Athletic.....2 2 0—11       |    |     |     |    |    |   |
| Runs earned—Boston, 4; Athletic, 5. Home-run—Hall, 1.      |    |     |     |    |    |   |                             |    |     |     |    |    |   |
| Total bases on hits—Boson, 22; Athletic, 20. First base by |    |     |     |    |    |   |                             |    |     |     |    |    |   |
| errors—Boston, 8; Athletic, 5. Umpire, George White o      |    |     |     |    |    |   |                             |    |     |     |    |    |   |
| Lowell, Mass. Time 2h. 47m.                                |    |     |     |    |    |   |                             |    |     |     |    |    |   |

[illegible]

# 棒球统计数据

- 进攻数据
  - 打席 (PA)
  - 安打 (H)
  - 全垒打 (HR)
  - 长打率 (SLG)
  - ...
- 投球数据
  - 胜投
  - 完封
  - ...
- 防守数据
  - 封杀



$$IsoP = \frac{(1B) + (2 \times 2B) + (3 \times 3B) + (4 \times HR)}{AB} - \frac{H}{AB}$$

- *Slugging Percentage (SLG) = Total Bases ÷ At Bats*
- *Total Bases = Singles + (2 x Doubles) + (3 x Triples) + (4 x Home Runs)*
- *Total Bases (alternate method) = Hits + Doubles + (2 x Triples) + (3 x Home Runs)*
- **IsoP = (H + 2B + (2 x 3B) + (3 x HR))/AB - H/AB**

## 棒球数据分析

- IsoP: **纯长打率**（英文：Isolated Power）
  - 纯长打率可用来评断一位打者的长打能力。
- 纯长打率 = 长打率 - 打击率。
  - 打击率是棒球运动中，评量打者（击球员）成绩的重要指标。
- “每一次打击可以贡献几个垒包”。

# MLB统计的数据

- **BATTING**（打击）
- **2B**（二垒安打）、**3B**（三垒安打）、**HR**（本垒打），一般**1B**（一垒安打）就不写了。后面跟着打者名字，括号里记上本赛季第几只，从哪个投手手里打出的，本垒打还会详细到局面（比如第几局，谁投球，垒上几人，几出局）
- **TB**：垒打数，打者名字后面跟数字，一个就不写数字了
- **RBI**：得分打数（**Runs Batted In**），又叫打点，打者名字后面跟数字，一个就不写数字了，括号里是本赛季总得分打数
- **2-out RBI**：两出局时得分打数，打者名字后面跟数字。这个统计数字的意义应该可以在一定程度上衡量打者的心理素质 and 打关键球的能力
- **Runners left in scoring position, 2 out**：两出局时跑垒员站上得点圈（站上二垒或三垒）但变成残垒数，打者名字后面跟数字。这个统计数字可以在一定程度上反向衡量队伍获取关键分的能力
- **GIDP**：地滚球双杀打数（**Ground Into Double Play**），打出地滚球造成了多人出局
- **Team RISP**：全队得点圈有跑者数（**Runners in scoring position**），**n-for-m**的意思是  $m$  次有跑者在得点圈时打出了  $n$  次安打（但不一定导致下分）。这个统计数字可以在一定程度上衡量队伍的得分能力，以及衡量进攻队员打关键球的能力
- **Team LOB**：全队残垒数（**Left On Base**）

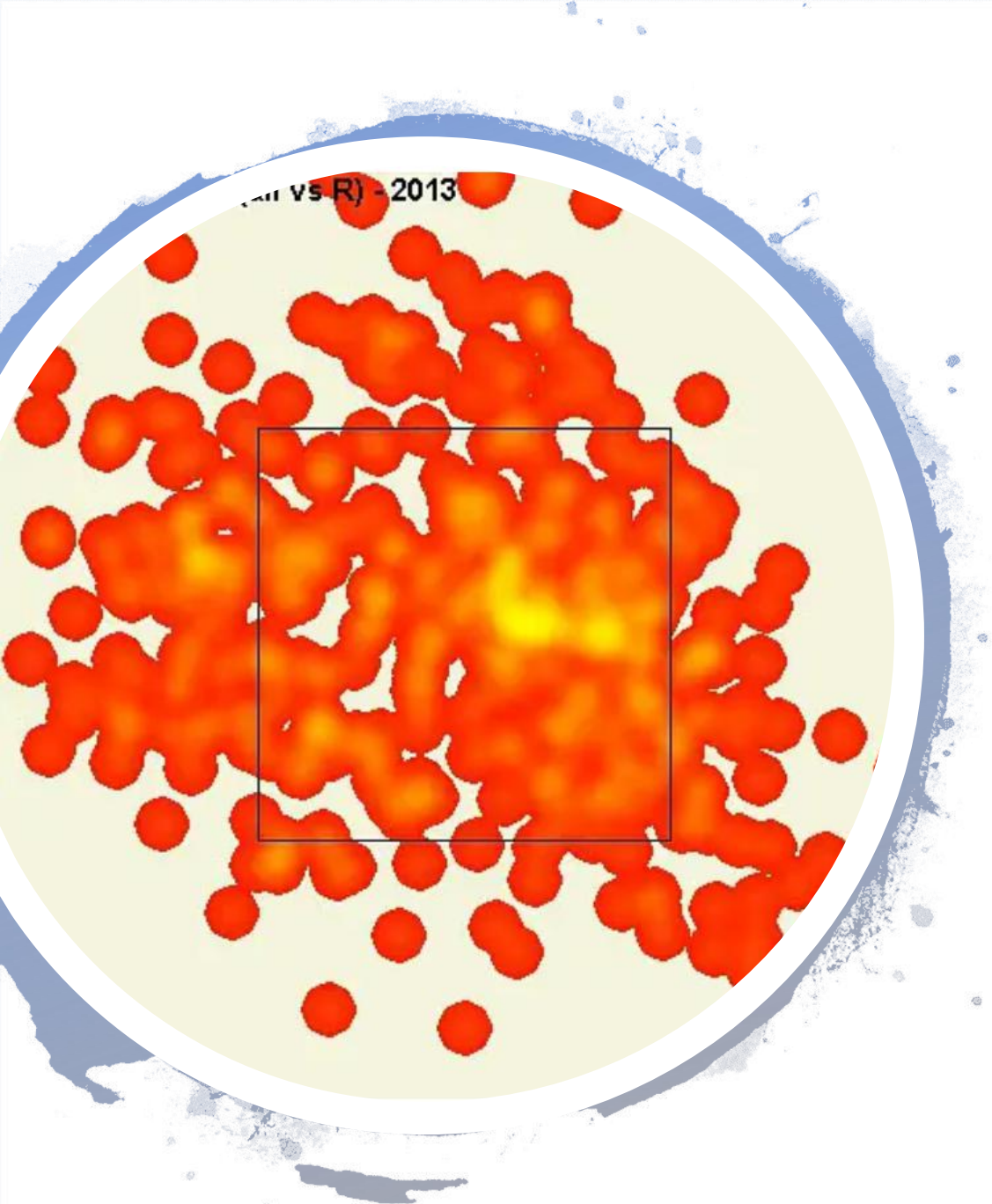
- BASERUNNING（跑垒）
  - SB: 盗垒数（Stolen Base）
- FIELDING（防守）
  - PB: 捕手漏接，写捕手名字，括号里是本赛季总漏接数
  - DP: 双杀，以传球顺序写防守队员名字
  - E: 失误，写失误队员名字，括号里是本赛季失误数，和本场比赛的失误类型



- 统计每位投手的：
- **IP** 投球局数、
- **H** 被安打数、
- **R** 被取分数、
- **ER** 自责分 或 责任失分、
- **BB** 保送上垒数、
- **SO** 三振出局数、
- **HR** 被打本垒打数、
- **ERA** 自责分率。

- **WP:** 暴投，写投手名字
- **IBB:** 故意投出四坏球，写被保送的球员名字，括号里是被哪个投手保送
- **HBP:** 投球中身，写被保送的球员名字，括号里是被哪个投手保送
- **Pitches-strikes:** 写 投手名字 投球数-好球数
- **Groundouts-flyouts:** 写 投手名字 滚地球封杀出局数-高飞球接杀出局数
- **Batters faced:** 写 投手名字 面对过的击球员数
- **Umpires:** 裁判，**HP** 本垒主审裁判名字；**1B** 一垒审裁判名字；**2B** 二垒审裁判名字；**3B** 三垒审裁判名字
- **Weather:** 比赛时天气情况
- **Wind:** 比赛时风速、风向
- **First pitch:** 开球时间
- **T:** 比赛用时，需要减去因故延误的时间
- **Att:** 观众数量
- **Venue:** 比赛场地
- 最后是比赛日期

- MLB 数据集
- <http://www.seanlahman.com/baseball-archive/statistics/>

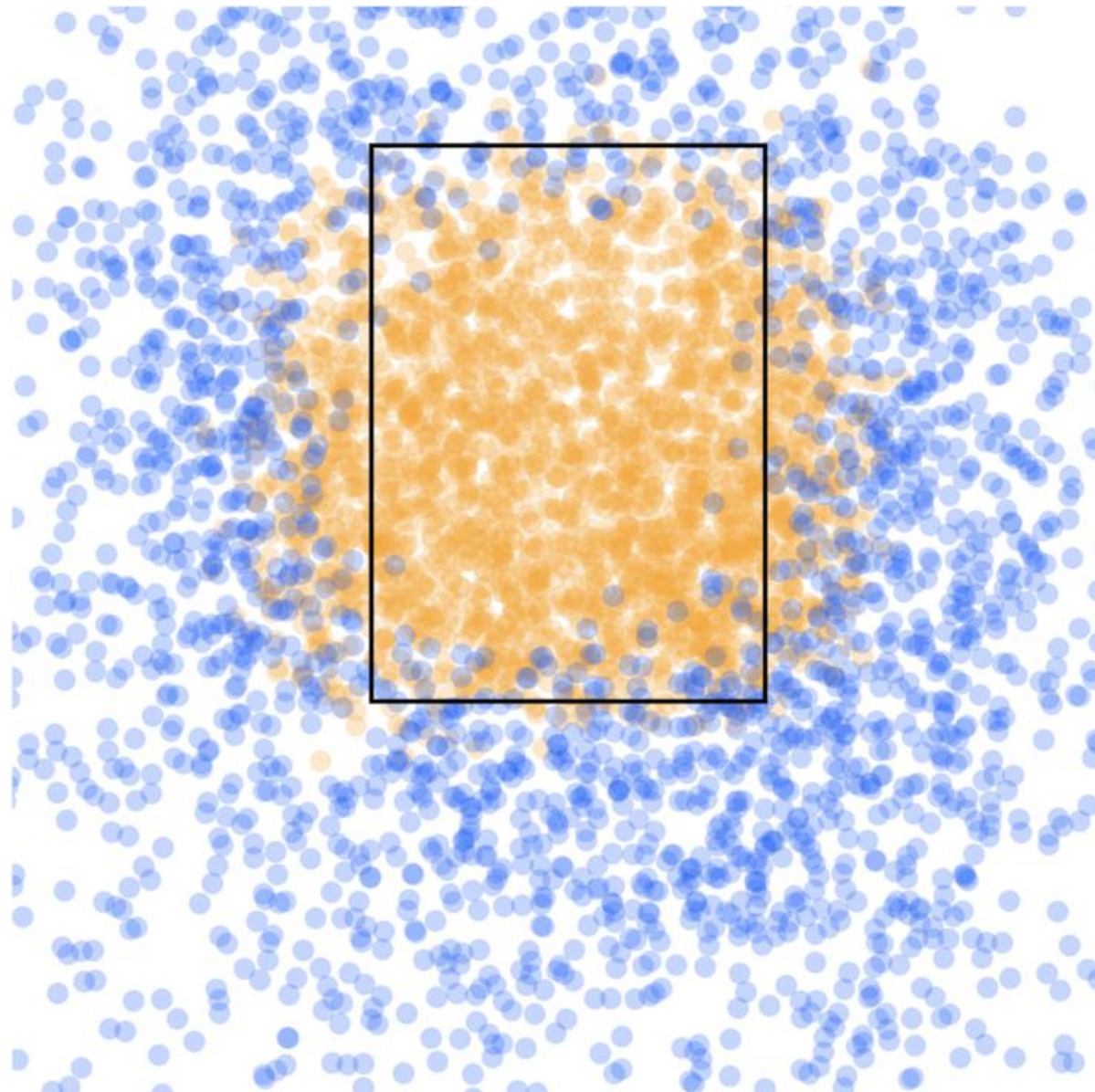


# 数据分析

- 球员个人情况分析
- 技战术分析
- 时间序列分析
- 空间分析

影片：点球成金

好球？ 坏球？



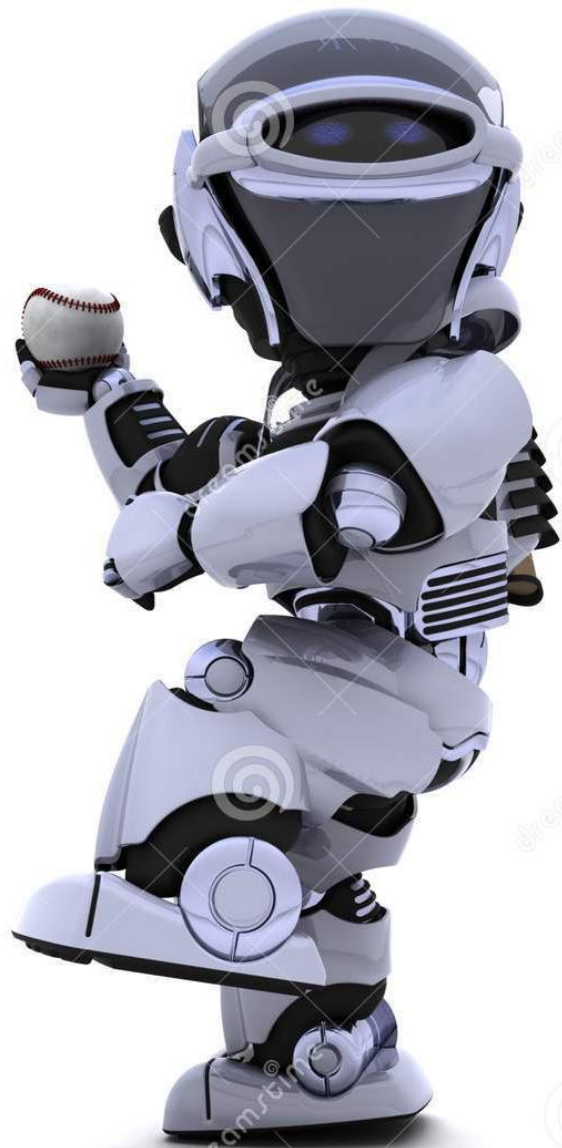
- **Input:**

- 协调球越过本垒的位置（'px'和'pz'）
- 击球手站在球场的哪一侧
- 击球区（击球手的躯干）的高度，以英尺为单位。
- 击球区底部的高度（击球手的膝盖）以英尺为单位

- **Output:**

- 该次击球是好球还是坏球（由裁判员判定的）





# 棒球机器人？



全球種マシン **主力品**

**HNB-620K** 硬式用  
**HNB-620N** 軟式用

本体価格 ¥540,000 + 税

■仕様

- 電源 / AC100V
- モーター / DC165W×2
- 重量 / 90kg
- 寸法 / 横 70× 縦 85× 高さ 130cm
- ローター寸法 / 320φ
- 最高速度 / 硬式…140km/h  
軟式…120km/h
- ボール投球高さ / ストレート…105cm  
カーブ…120cm
- 付属品 / マシンカバー

# 数据分析实践



- 程序编辑环境
  - Python
  - Ipython
  - Ipython notebook
  - jupyter notebook
- 语言
  - **Python**
  - R
- 编辑软件
  - Notepad++
  - Python IDLE
  - Spyder
  - **VS code**
  - **Pycharm**

# 计算机能听懂的语言之一—Python

- 语法简洁
- 类库强大
  - 运维自动化、数据分析、机器学习首选编程语言。
- 开发效率高
  - Python简单10行代码实现的功能，用其他语言可能需要100行才能完成。
- 行业应用领域广泛
  - 云计算、机器学习、科学运算、自动化运维
  - 爬虫、数据分析、GUI图形化、Web开发

# Python环境搭建

- Python下载:
  - <https://www.python.org/downloads/>
  - 2.x
  - **3.x**
  - 勾选 Add Python to PATH
- VS code下载:
  - <https://code.visualstudio.com/>

- Anaconda

- Anaconda2

- Anaconda3

- Miniconda

- Pycharm

- .edu 获取license

Jupyter Notebook

## 1.Jupyter Notebook基本情况

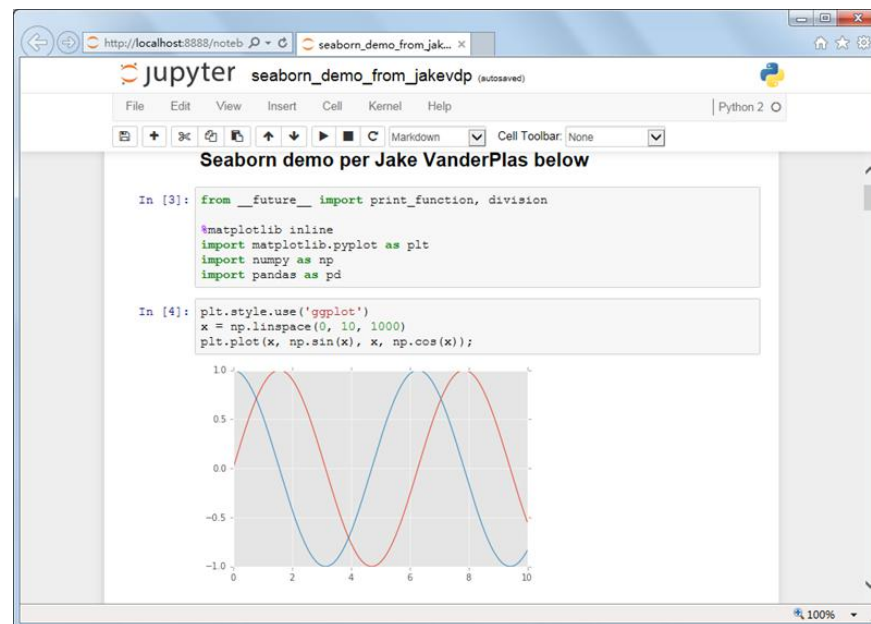
2.IPython介绍

3.使用ipywidgets添加控件

4.使用Markdown编写文档


# 1.1Jupyter Notebook是什么？

- Jupyter Notebook是一个基于网页的文档编辑与运行环境，可以编辑和运行代码，并能以多种形式显示运行结果，特别适合软件开发过程中的交流与协同。



- Jupyter Notebook是由Jupyter项目团队开发的，是一个开源软件，以前是叫IPython Notebook。
- Anaconda等Python发行版软件集成了Jupyter Notebook，在安装Anaconda时会同时安装Jupyter Notebook。



 Jupyter Notebook

stable

Search docs

USER DOCUMENTATION

[-] The Jupyter Notebook

[-] Introduction

[-] Starting the notebook server

- Notebook user interface

[-] Structure of a notebook document


[-] Basic workflow

- Plotting
- Installing kernels
- Trusting Notebooks
- Browser Compatibility

User interface components


Notebook Examples

What to do when things go wrong

 Read the Docs

v: stable ▾

Docs » The Jupyter Notebook

 Edit on GitHub


# The Jupyter Notebook

## Introduction

The notebook extends the console-based approach to interactive computing in a qualitatively new direction, providing a web-based application suitable for capturing the whole computation process: developing, documenting, and executing code, as well as communicating the results. The Jupyter notebook combines two components:

**A web application:** a browser-based tool for interactive authoring of documents which combine explanatory text, mathematics, computations and their rich media output.

**Notebook documents:** a representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects.

 See also

See the [installation guide](#) on how to install the notebook and its dependencies.

Jupyter Notebook使用文档

<https://jupyter-notebook.readthedocs.io/en/stable/notebook.html>

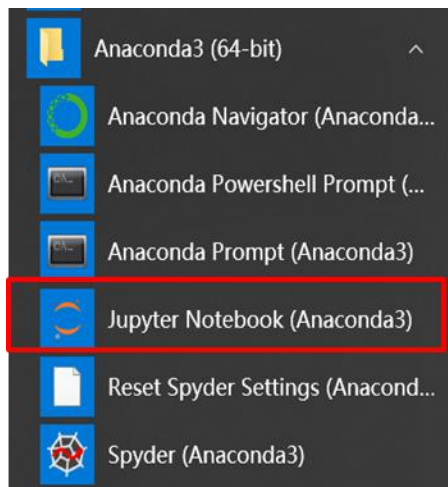
- Jupyter Notebook的特点：
  - 支持多种编程语言，Jupyter这个名字是其中最主要的三种语言（Julia、Python和R）的缩写。
  - 一个Notebook可以包含多个单元（cell），每个单元可以单独运行，也可以关联运行，即可以把多个应用程序放在同个Notebook中。
  - 代码运行结果能以多种形式显示，包括HTML、LaTeX、PNG、SVG等。

- 可以利用ipywidgets包在输出结果中添加交互操作控件，实现对输出结果的交互操作。
- 可以插入用HTML、Markdown、LaTeX等标记语言编写的文本。

## 1.2 Jupyter Notebook 程序运行

- Jupyter Notebook 是 B/S 架构的软件，程序的运行是由后台的服务和前台的浏览器页面操作两部分组成，前台的操作请求发送到后台 Jupyter Notebook 服务器进行处理，服务器响应请求，然后把处理结果返回到浏览器进行显示。

- 运行Jupyter Notebook程序，首先将启动后台的Notebook服务器。



```
Jupyter Notebook (Anaconda3)
[I 09:20:37.111 NotebookApp] [jupyter_nbextensions_configurator] enabled 0.4.1
[I 09:20:37.150 NotebookApp] JupyterLab extension loaded from C:\Anaconda3\lib\site-packages\jupyterlab
[I 09:20:37.151 NotebookApp] JupyterLab application directory is C:\Anaconda3\share\jupyter\lab
[I 09:20:37.153 NotebookApp] Serving notebooks from local directory: c:\notebook
[I 09:20:37.153 NotebookApp] The Jupyter Notebook is running at:
[I 09:20:37.153 NotebookApp] http://localhost:8888/?token=af433fc9ab07884a7953f69842fe3566bf66b65805c5d414
[I 09:20:37.153 NotebookApp] or http://127.0.0.1:8888/?token=af433fc9ab07884a7953f69842fe3566bf66b65805c5d414
[I 09:20:37.154 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 09:20:37.196 NotebookApp]

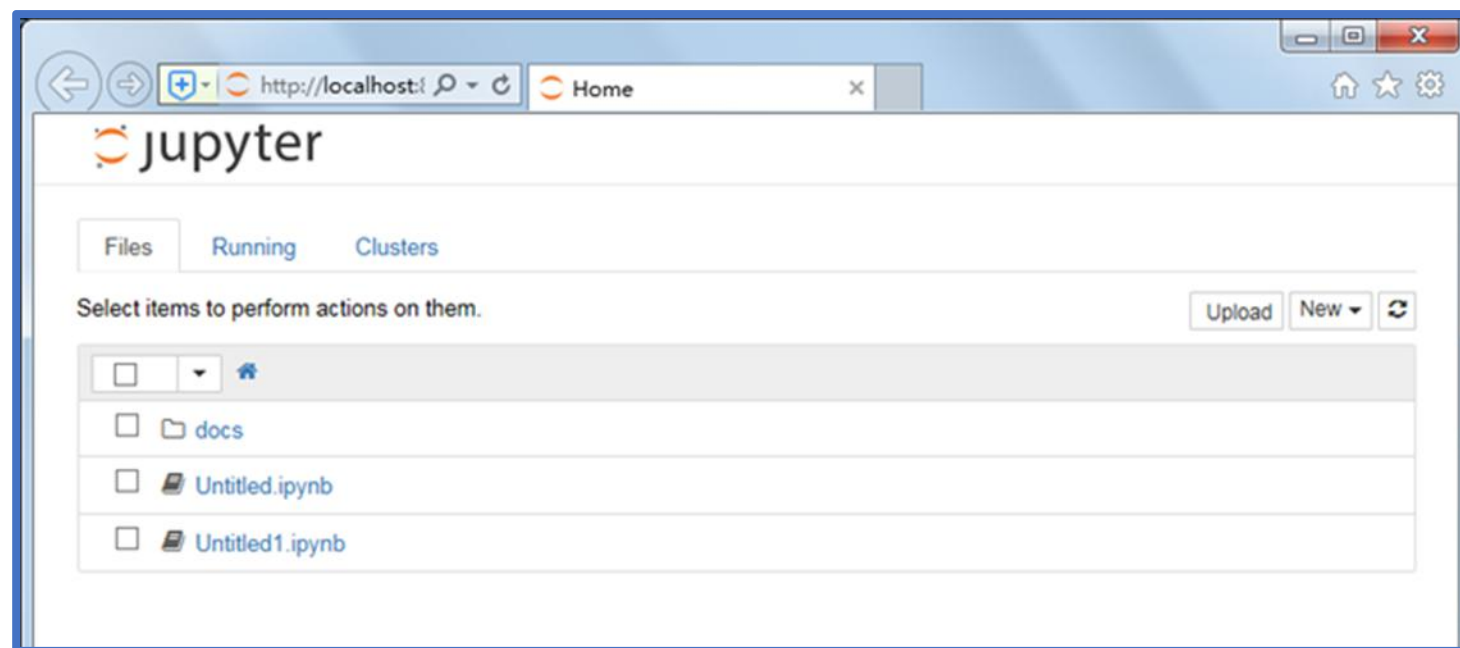
To access the notebook, open this file in a browser:
file:///C:/Users/GIS/AppData/Roaming/jupyter/runtime/nbserver-51480-open.html
Or copy and paste one of these URLs:
http://localhost:8888/?token=af433fc9ab07884a7953f69842fe3566bf66b65805c5d414
or http://127.0.0.1:8888/?token=af433fc9ab07884a7953f69842fe3566bf66b65805c5d414
```

- Notebook服务器启动后，会在默认的Web浏览器中打开Notebook的主页面，主页面的地址是：  
`http://localhost:8888/tree`，这里的8888是Jupyter Notebook默认的端口号，如端口号已被其它Notebook占用，则端口号的数字自动加1，如“8889”、“8890”.....。

- 默认的Web浏览器可在Windows系统设置的应用/默认应用中改变。

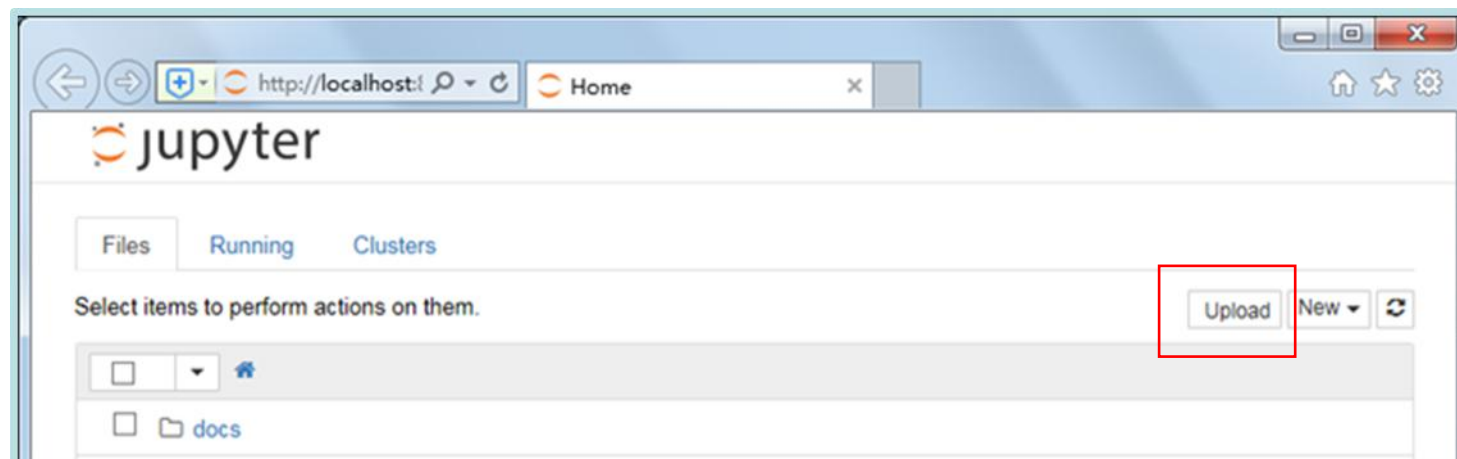


- 主页面类似资源管理器，显示工作目录（home directory）下的文件夹、Notebook文件以及其它支持的文件。

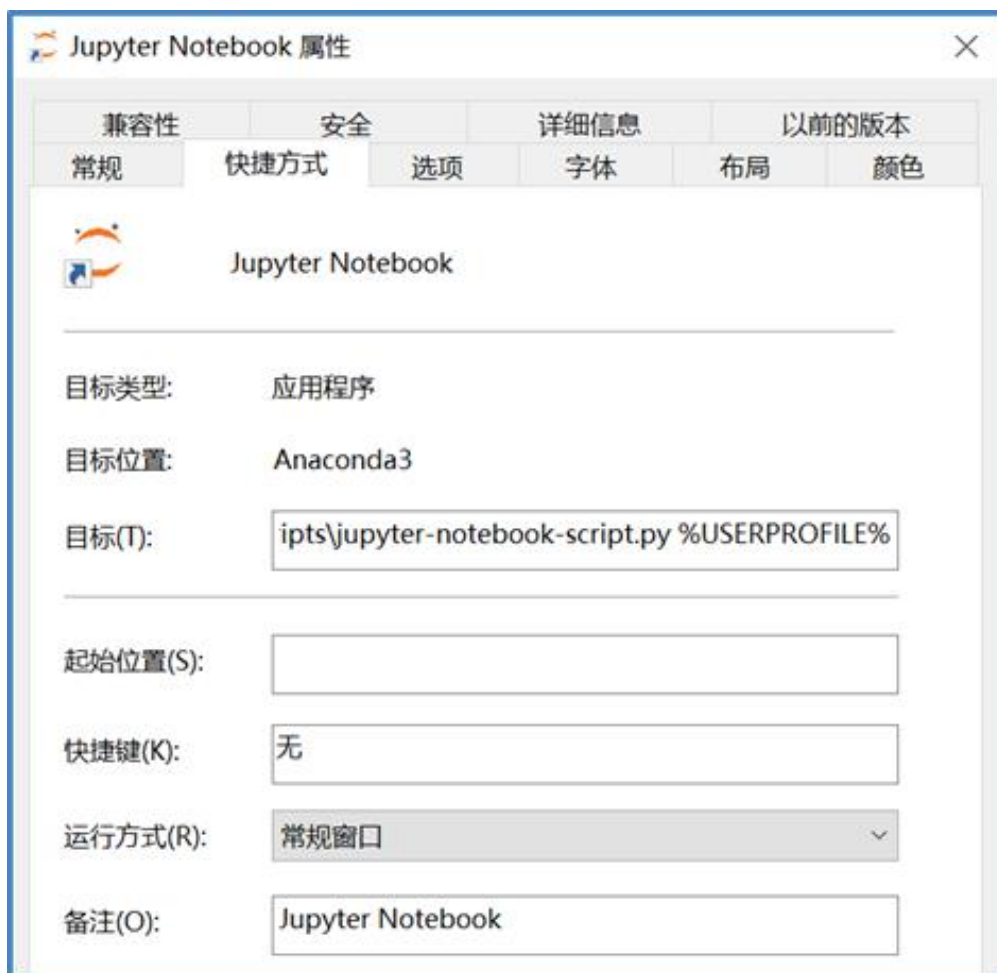




- Jupyter Notebook只能操作工作目录（包括子目录）下的文件，如要操作其它目录下的文件，需要点击Upload按钮，把文件加载到工作目录（或子目录）下。



- 默认的工作目录是c:\user\user\_name。如要改变工作目录，一种方式是打开Jupyter Notebook快捷文件的属性对话框，改变“目标”中定义的工作目录，保存后重新启动Jupyter Notebook，将进入新的工作目录。

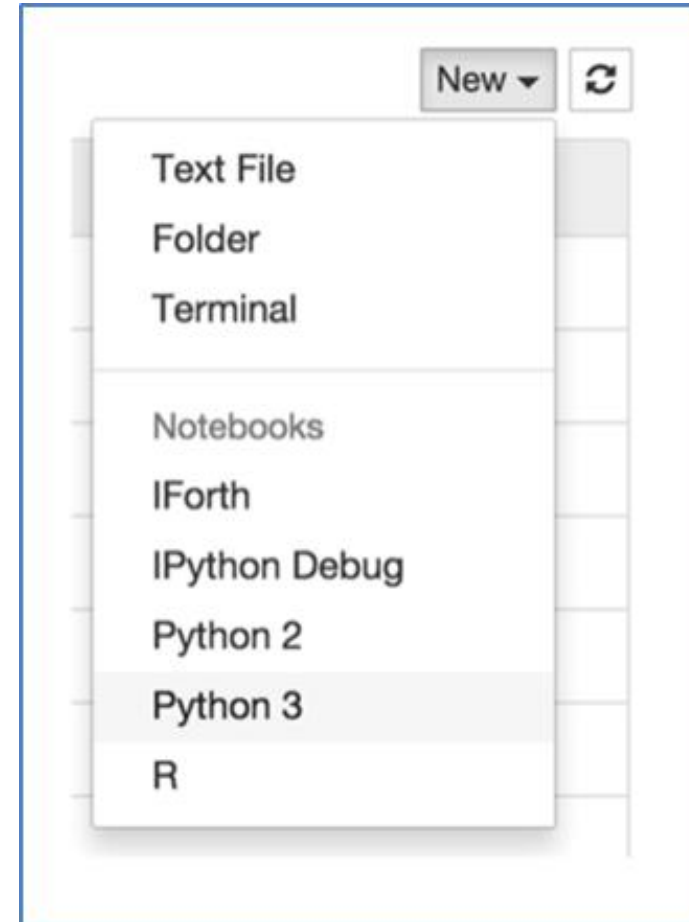


如新的工作目录为c:\gis，  
则把目录中的  
%USERPROFILE%改成  
c:\gis。

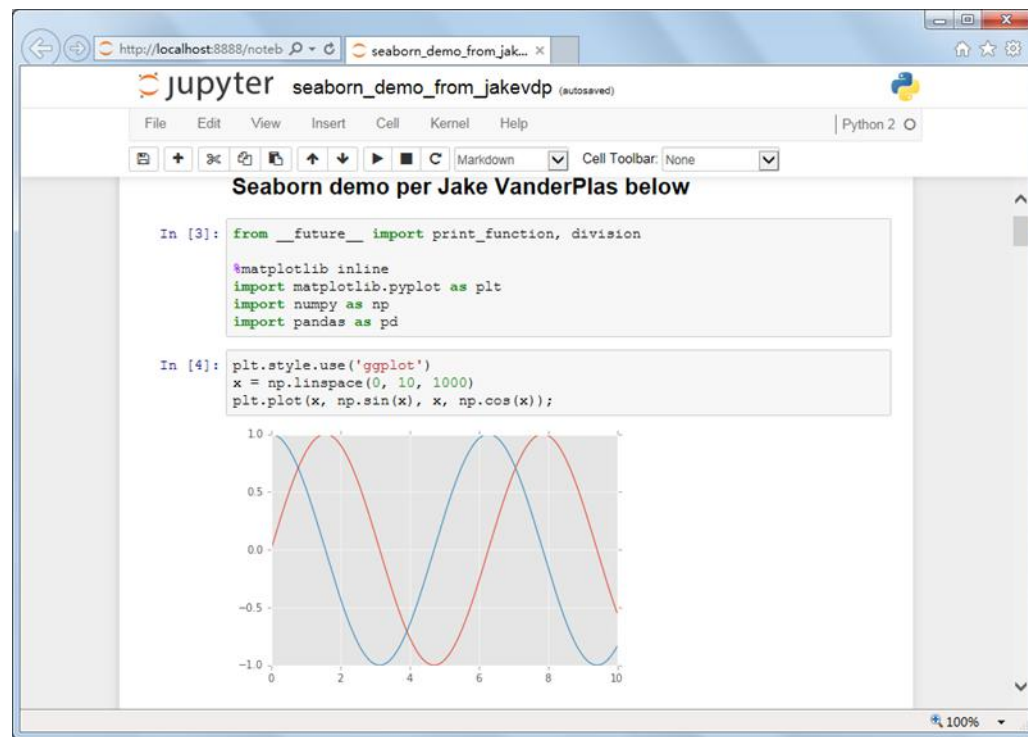
Jupyter Notebook快捷文件的属性对话框

# 1.3 Notebook文档的创建与编辑

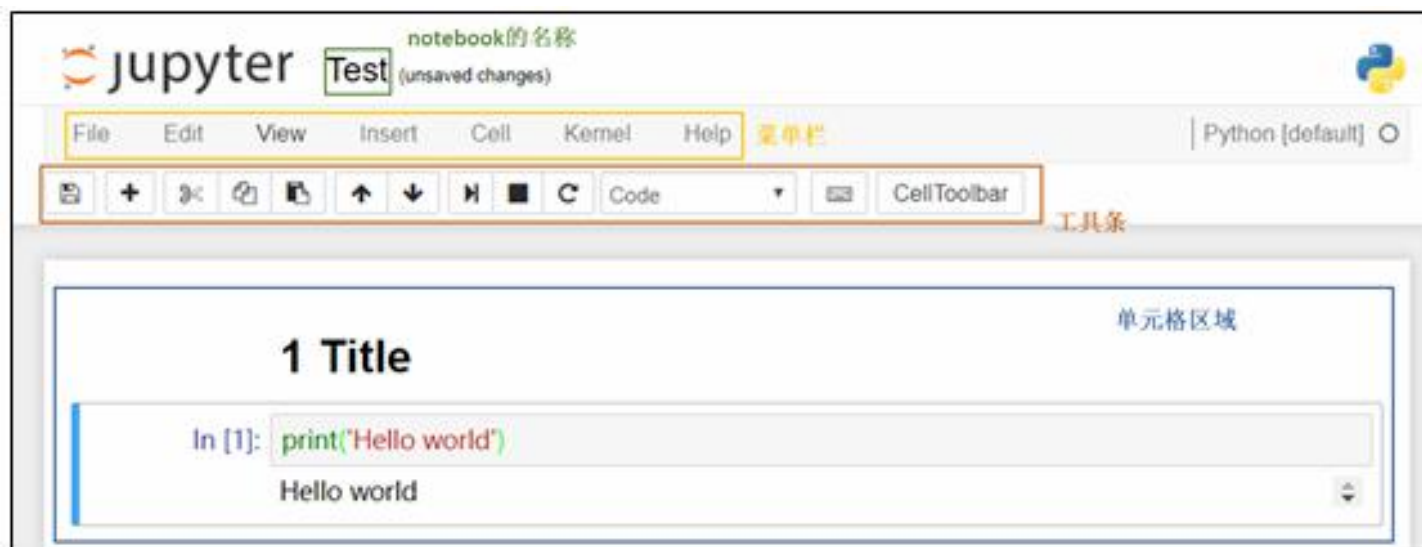
- 点击主页面右侧的“New”按钮，可以新建Notebook文档。如新建文档涉及程序代码，需要选择相应的编程语言解释器（Kernel），可选的Kernel依赖于服务器安装了哪些Kernel。



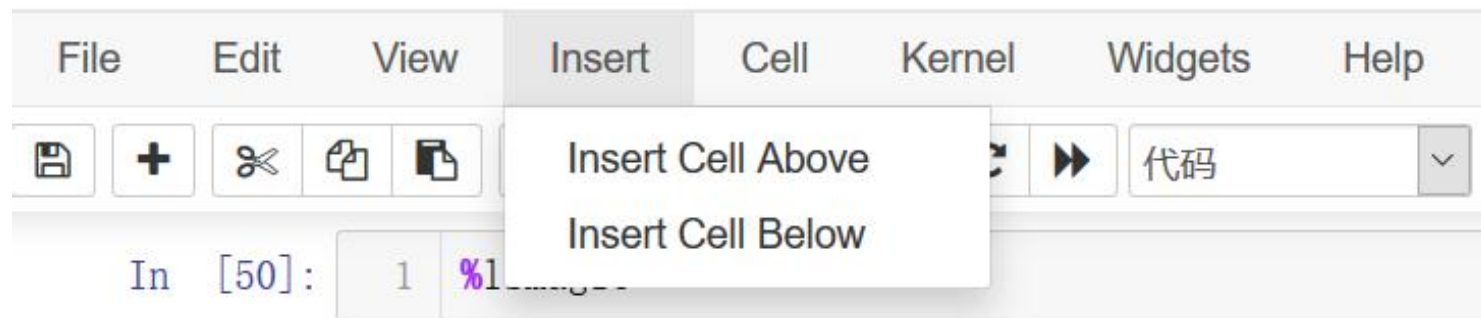
- 新建或打开一个Notebook后，将切换到Notebook编辑界面。



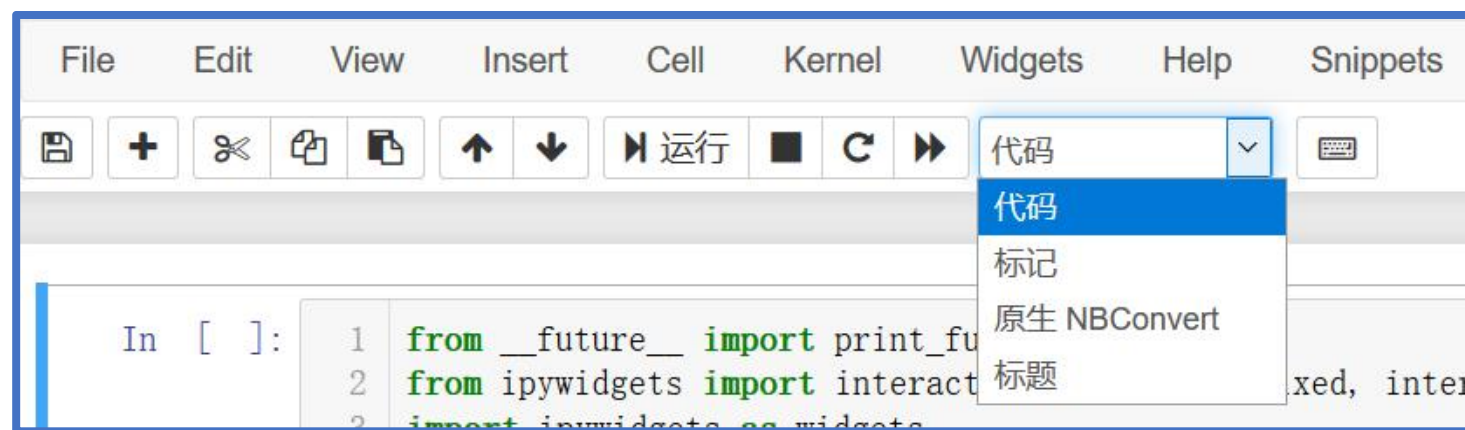
- Notebook的编辑界面由四部分组成： Notebook名称、菜单栏、工具条以及单元（Cell）区域。



- 一个Notebook可以有很多单元，点击Insert/Insert Cell Above或Insert Cell Below可以插入新的单元。

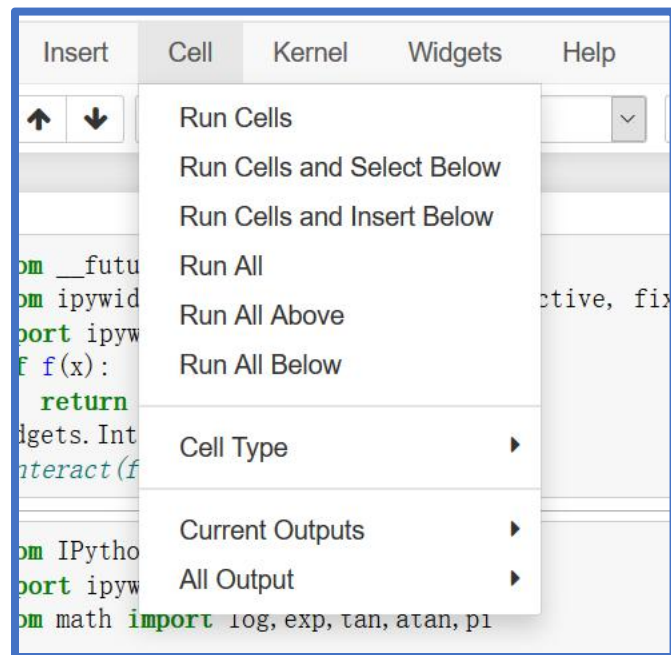


- 单元的类型主要有三种：代码、标记和原生 NBConvert，分别用于输入代码、标记以及不被渲染的文档。点击工具条中的下拉列表框可以设置单元类型。

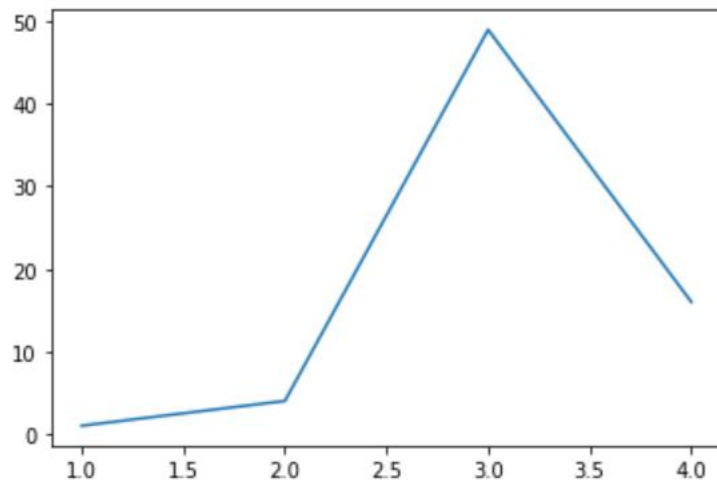




- 点击“运行”按钮将运行当前Cell；点击Cell菜单，可以选择部分单元或所有单元进行运行。



```
In [1]: 1 %matplotlib inline
        2 import matplotlib.pyplot as plt
        3 x = [1, 2, 3, 4]
        4 y = [1, 4, 49, 16]
        5 plt.plot(x, y)
        6 plt.show()
```



代码单元运行后，标准输出在单元下显示。

```
1 ### 以下是引用:
2 > This is the first level of quoting.
3 >
4 >> This is nested blockquote.
5 >
6 > Back to the first level.
7 >>>这是引用
```

以下是引用:

This is the first level of quoting.

This is nested blockquote.

Back to the first level.

这是引用

标记单元运行后，输入的文本将按标签进行显示。

- 点击Cell菜单下的Current Output\Clear或All Output\Clear将清除当前代码单元或所有代码单元的输出结果。
- 双击标记单元的运行结果，将恢复显示原始的文档。

# 1.4文档的保存与输出

- 缺省情况下，Notebook保存的文件是JSON格式文件，扩展名为ipynb，它包含所有单元的内容及运行结果。



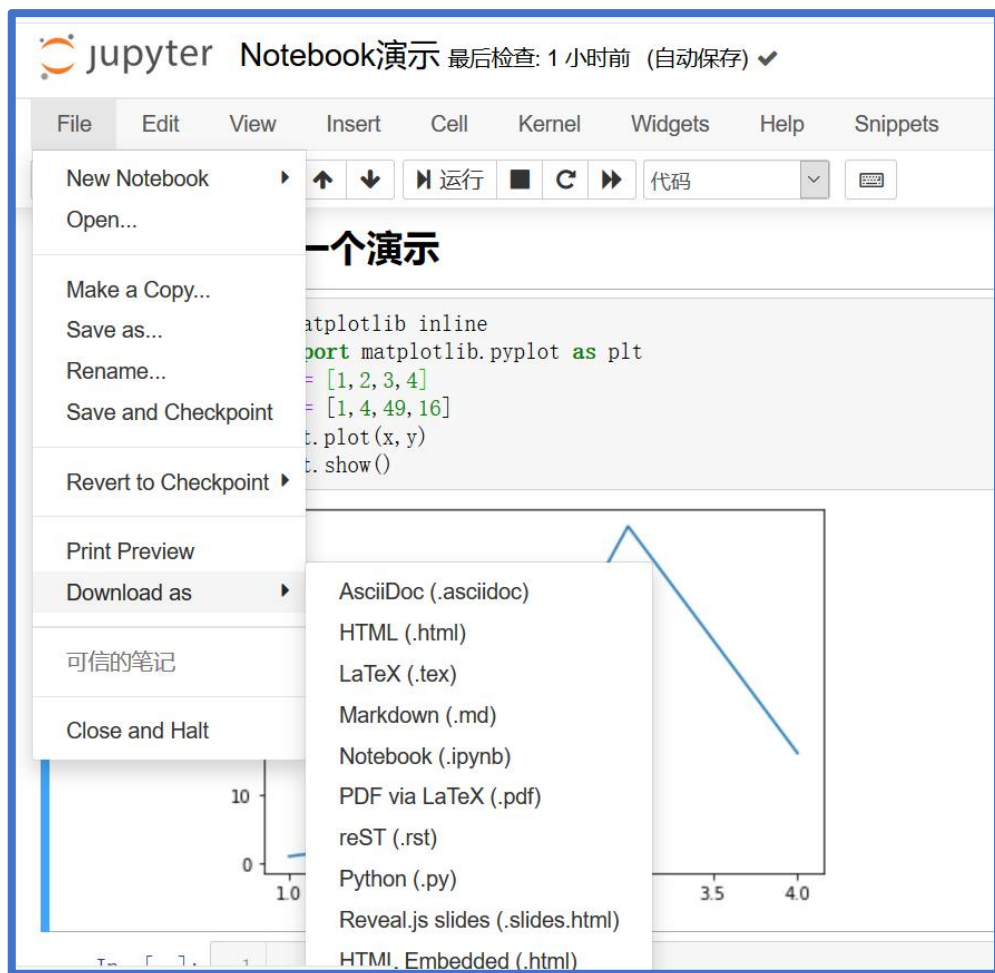
```

{
  "cells": [
    {
      "cell_type": "markdown",
      "metadata": {},
      "source": [
        "## 这是一个演示"
      ]
    },
    {
      "cell_type": "code",
      "execution_count": 93,
      "metadata": {},
      "outputs": [

```

第 1 行, 第 1 列    100%    Unix (LF)    UTF-8

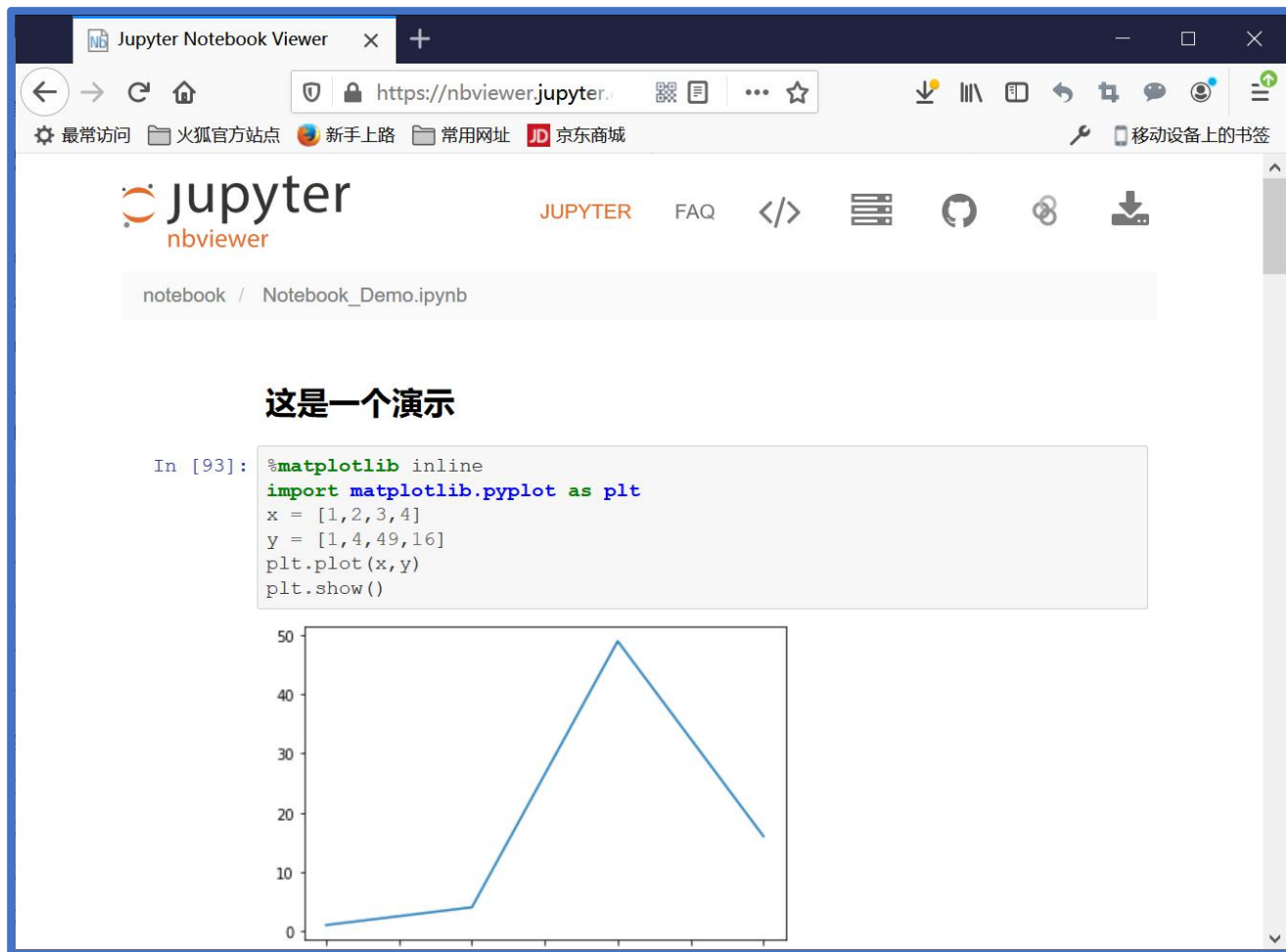
- Notebook文档也可以下载输出到其它格式文件，包括py文件、HTML文件、Markdown文件等。



- Jupyter项目提供了一个Notebook文档在线浏览工具（nbviewer），该工具会把在线的Notebook文档转成HTML格式并显示。



<https://nbviewer.jupyter.org/>



扫一扫，直接在手机上打开

在输入地址栏中输入：

`http://119.3.40.193:8080/gis_development/Notebook_Demo.ipynb`

1.Jupyter Notebook基本情况

2.IPython介绍

3.使用ipywidgets添加控件

4.使用Markdown编写文档



- IPython是由IPython项目团队开发的，是为Python提供更多交互操作功能。
  - 提供了一个功能更强的交互操作环境（`shell`），支持变量自动补全、自动缩进、快速显示对象的帮助信息等。
  - 提供了更多交互操作方面的API。
  - 提供了高性能并行计算工具。

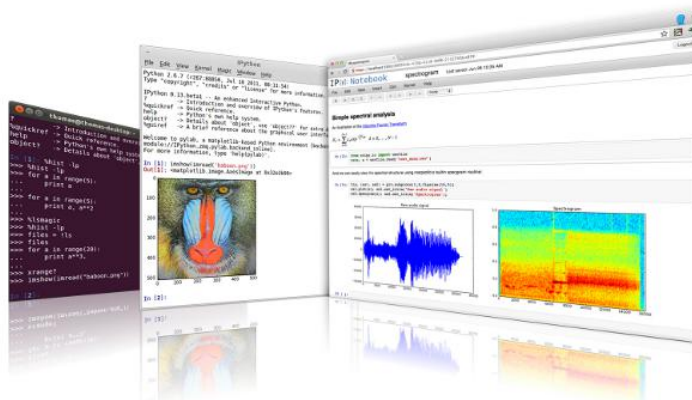
# IP[y]: IPython

Interactive Computing

[Install](#) · [Documentation](#) · [Project](#) · [Jupyter](#) · [News](#) · [Cite](#) · [Donate](#) · [Books](#)

IPython provides a rich architecture for interactive computing with:

- A powerful interactive shell.
- A kernel for [Jupyter](#).
- Support for interactive data visualization and use of [GUI toolkits](#).
- Flexible, [embeddable](#) interpreters to load into your own projects.
- Easy to use, high performance tools for [parallel computing](#).



To get started with IPython in the Jupyter Notebook, see our [official example collection](#). Our [notebook gallery](#) is an excellent way to see the many things you can do with IPython while learning about a

IPython官方网站 (<https://ipython.org/>)

- Jupyter Notebook上的Python使用的是IPython Kernel，因此在Jupyter Notebook中可以直接使用IPython功能，如直接利用对象名输出对象值（默认情况下，如代码中最后一行是对象，则直接输出该对象值）。

```
In [4]: 1 a = 1
        2 b = 2
        3 c = a + b
        4 c

Out[4]: 3
```

- IPython提供了很多用于交互操作的magic函数，magic函数有两种类型：line magic函数和cell magic函数。
  - line magic函数用一个%符号表示，作用于当前行。
  - cell magic函数是用两个%符号表示，作用于当前单元（cell）。

- 使用%lsmagic函数将显示所有可用的magic函数。

```
1 %lsmagic
```

Available line magics:

```
%alias %alias_magic %autoawait %autocall %automagic %autosave %bookmark %cd %clear %cls %colors %conda %config %connect_info %copy %ddir %debug %dhist %dirs %doctest_mode %echo %ed %edit %env %gui %hist %history %killbgscripts %ldir %less %load %load_ext %loadpy %logoff %logon %logstart %logstate %logstop %ls %lsmagic %macro %magic %matplotlib %mkdir %more %notebook %page %pastebin %pdb %pdef %pdoc %pfile %pinfo %pinfo2 %pip %popd %pprint %precision %prun %psearch %psource %pushd %pwd %pycat %pylab %qtconsole %quickref %recall %rehashx %reload_ext %ren %rep %rerun %reset %reset_selective %rmdir %run %save %sc %set_env %store %sx %system %tb %time %timeit %unalias %unload_ext %who %who_ls %whos %xdel %xmode
```

Available cell magics:

```
%%! %%HTML %%SVG %%bash %%capture %%cmd %%debug %%file %%html %%javascript %%js %%latex %%markdown %%perl %%prun %%pypy %%python %%python2 %%python3 %%ruby %%script %%sh %%svg %%sx %%system %%time %%timeit %%writefile
```

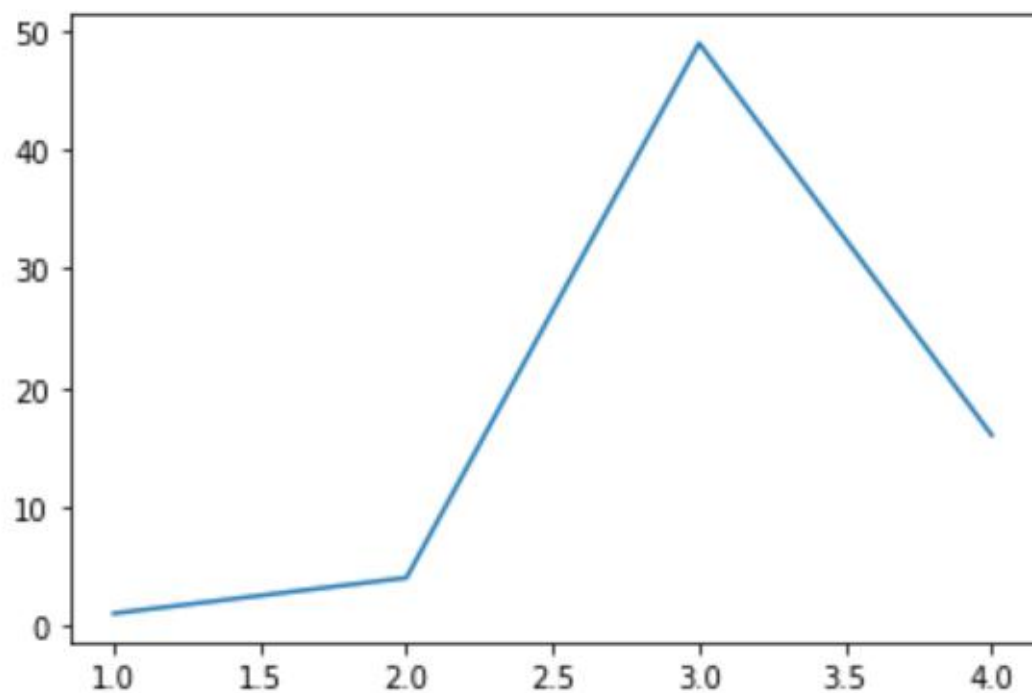
Automagic is ON, % prefix IS NOT needed for line magics.

- 常用的magic函数：
  - %matplotlib
  - %%time和%%timeit
  - %%writefile
  - %load
  - %run

## 2.1%matplotlib函数

- matplotlib是一个用于绘制图表的Python包，  
%matplotlib函数用于定义图表输出的GUI（图形用户界面），如%matplotlib inline表示在Notebook中输出；%matplotlib qt表示在qt开发的GUI中输出（产生一个新的窗口）。

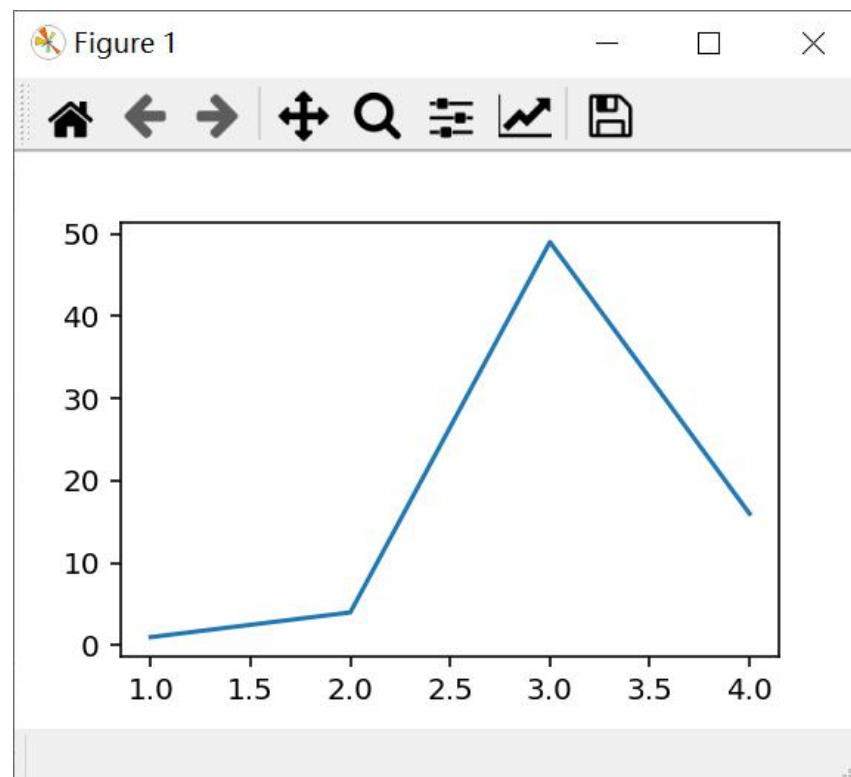
```
1 %matplotlib inline
2 import matplotlib.pyplot as plt
3 x = [1, 2, 3, 4]
4 y = [1, 4, 49, 16]
5 plt.plot(x, y)
6 plt.show()
```



在Notebook中输出



```
%matplotlib qt  
import matplotlib.pyplot as plt  
x = [1,2,3,4]  
y = [1,4,49,16]  
plt.plot(x,y)  
plt.show()
```



在qt开发的GUI中输出

## 2.2%%time和%%timeit函数

- %%time和%%timeit函数用于计算代码执行时间，其中%%time是计算代码一次执行的时间；%%timeit是计算代码多次执行的平均时间，因为每次执行同一个代码的时间是不一样的，通过多次执行代码求出的平均时间更能反映实际情况。

```
%%time  
x = range(100000000)  
min_x = min(x)  
max_x = max(x)
```

Wall time: 501 ms

```
%%timeit  
x = range(100000000)  
min_x = min(x)  
max_x = max(x)
```

474 ms  $\pm$  18.1 ms per loop (mean  $\pm$   
std. dev. of 7 runs, 1 loop each)

计算代码的执行时间

## 2.3%%writefile函数

- %%writefile函数用于把单元中的代码写到文件中，如：

```
%%writefile c:/data/write_file.py  
x = range(100000000)  
print(min(x))  
print(max(x))
```

## 2.4%load函数

- %load函数用于加载文件中的代码，文件可以是本地文件（如%load c:/data/write\_file.py），也可以是网络文件（如%load http://119.3.40.193:8080/gis\_development/write\_file.py）。

```
In [ ]: 1 # %load c:/data/write_file.py
        2 x = range(10000000)
        3 print(min(x))
        4 print(max(x))
        5
        6
```

加载后的结果

## 2.5%run函数

- %run函数用于运行文件中的代码。

```
In [18]: 1 %run c:/data/write_file.py  
0  
9999999
```

1.Jupyter Notebook基本情况

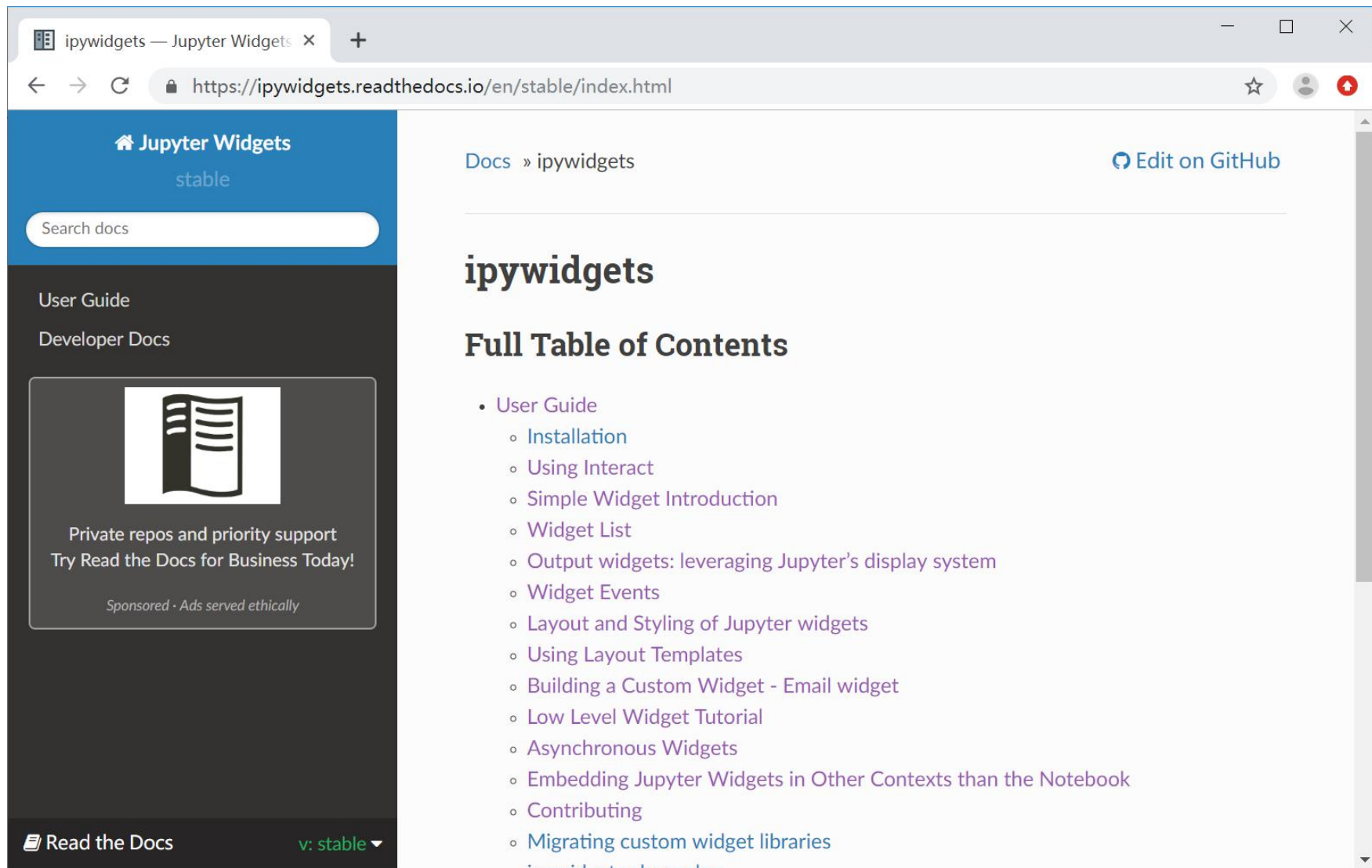
2.IPython介绍

3.使用ipywidgets添加控件

4.使用Markdown编写文档

- `ipywidgets`以前是IPython包中的一个部分，现在是单独的一个扩展包，用于在Notebook中构建各种控件（Widgets），如slider、textbox等。





ipywidgets文档

<https://ipywidgets.readthedocs.io/en/stable/index.html>

- **ipywidgets**模块中的控件包括滑动条控件、文本框控件、选择控件、标注控件、按钮控件、输出控件、容器控件等。

## 3.1滑动条控件

- 滑动条控件包括IntSlider、FloatSlider等控件，用户可以通过滑动条上的指针改变滑动条的值。在创建Slider控件可以设置最小值、最大值、步长、缺省值等属性。

```
widgets.IntSlider(  
    value=7,  
    min=0,  
    max=10,  
    step=1,  
    description='value:'  
)
```



创建IntSlider控件，控件的值为整数。

## 3.2 文本框控件

- 文本框控件包括Text、Textarea、IntText、FloatText、BoundedIntText、BoundedFloatText等控件，其中：
  - Text和Textarea控件的值是单行字符串和多行字符串。
  - IntText和FloatText控件的值是整数和浮点数。
  - BoundedIntText和BoundedFloatText控件的值也分别是整数和浮点数，但可以定义最小和最大值。

```
widgets.Text(  
    value='Hello World!\nThis is a test',  
    description='输入字符串:'  
)
```

输入字符串:

Hello World!This is a test

```
widgets.Textarea(  
    value='Hello World!\nThis is a test',  
    description='输入字符串:'  
)
```

输入字符串:

Hello World!  
This is a test

Text控件和Textarea控件

```
widgets.BoundedFloatText(  
    value=7.5,  
    min=0,  
    max=10.0,  
    step=0.1,  
    description='输入数字:'  
)
```

输入数字:

BoundedFloatText控件

## 3.3选择控件

- 选择控件包括Dropdown（下拉选择框）、RadioButtons（单选按钮）、SelectMultiple（多选框）等，选择控件的属性包括options、value、description等，options属性值是一个列表，表示可选元素；value属性值表示选中的元素；description用于描述控件。

```
widgets.DropDown(  
    options=['北京', '上海', '广州'],  
    value='北京',  
    description='选择城市:'  
)
```



下拉选择框



```
widgets.RadioButton(  
    options=['北京', '上海', '广州'],  
    value='北京',  
    description='选择城市:'  
)
```

选择城市: ☒ 北京  
☐ 上海  
☐ 广州

单选按钮

```
widgets.SelectMultiple(  
    options=['北京', '上海', '广州'],  
    value=['北京', '上海'],  
    description='选择城市:'  
)
```



多选框

## 3.4标注控件

- 标注控件包括Label、HTML（HTML形式的标注）等控件，用于显示描述信息。

```
widgets.Label(value="这是一个标注")
```

这是一个标注

## 3.5按钮控件

- 按钮控件即Button控件，用于处理鼠标点击事件。Button控件的on\_click方法可关联一个函数，当点击Button时，执行关联的函数。关联函数的参数只有一个，即Button控件对象。

```
from IPython.display import display
import ipywidgets as widgets
def on_button_clicked(b):
    print(dropdown.value)
dropdown = widgets.Dropdown(
    options=['北京', '上海', '广州'],
    value='北京',
    description='选择城市:'
)
button = widgets.Button(description="显示选择的城市")
button.on_click(on_button_clicked)
display(dropdown,button)
```

选择城市:  ▼

上海

点击按钮打印输出dropdown控件的值

## 3.6输出控件

- 输出控件即**Output**控件，用于显示输出结果，在创建**Output**控件后，可以利用上下文管理器把输出结果显示在**Output**控件上。
- 不同于标准输出，**Output**控件中的输出结果可以利用**clear\_output()**方法清除。

```
from IPython.display import display
import ipywidgets as widgets
def on_button_clicked(b):
    with out:
        out.clear_output()
        print(dropdown.value)
dropdown = widgets.Dropdown(
    options=['北京', '上海', '广州'],
    value='北京',
    description='选择城市:'
)
button = widgets.Button(description="显示选择的城市")
button.on_click(on_button_clicked)
out = widgets.Output()
display(dropdown,button,out)
```

利用Output控件输出，每次输出先清除原先的内容。

## 3.7 容器控件

- 容器控件用于把多个控件组合在一个容器中，包括Box、HBox、VBox等，其中Box和HBox是水平放置控件，Vbox是垂直放置控件。

```
from IPython.display import display
button1 = widgets.Button(description="button1")
button2 = widgets.Button(description="button2")
button3 = widgets.Button(description="button3")
hBox = widgets.HBox([button1,button2,button3])
display(hBox)
```

button1

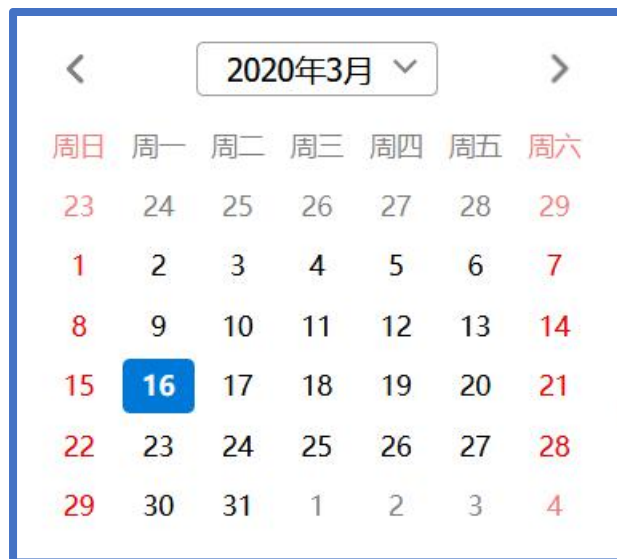
button2

button3



## 3.8 日期控件

- 日期控件即**DatePicker**控件，用于选择日期，点击该控件，会弹出一个显示日期的窗口用于选择日期。



```

from IPython.display import display
def on_button_clicked(b):
    with out:
        out.clear_output()
        print(date.value)
date = widgets.DatePicker(
    description='选择一个日期',
)
button = widgets.Button(description="显示日期")
button.on_click(on_button_clicked)
out = widgets.Output()
display(date,button,out)

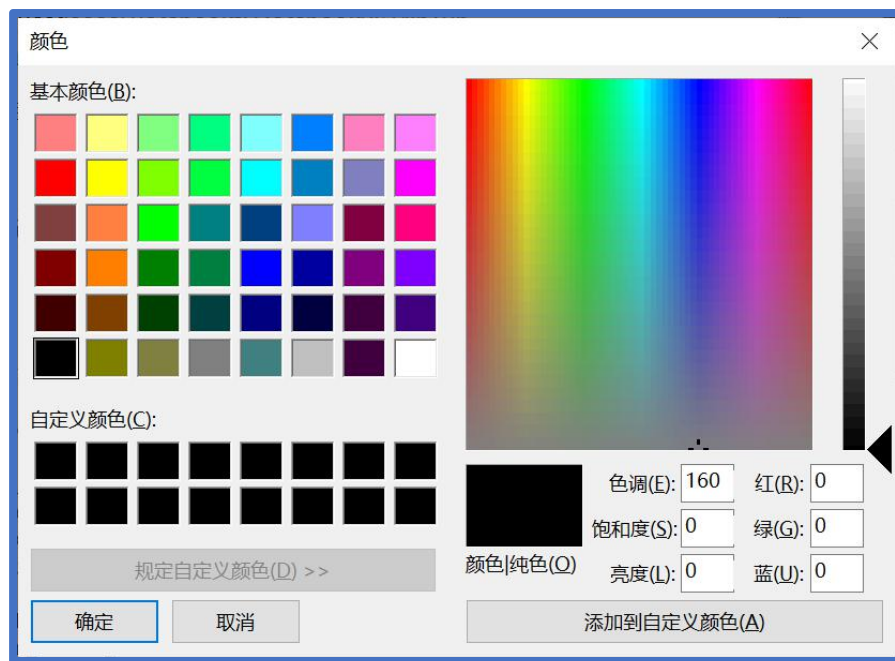
```



选择日期

## 3.9颜色控件

- 颜色控件即ColorPicker控件，用于选择颜色，点击该控件，会弹出一个显示颜色的窗口用于选择颜色。



```
from IPython.display import display
def on_button_clicked(b):
    with out:
        out.clear_output()
        print(color.value)
color = widgets.ColorPicker(
    description='选择一个颜色',
)
button = widgets.Button(description="显示颜色值")
button.on_click(on_button_clicked)
out = widgets.Output()
display(color,button,out)
```



选择颜色

## 3.10 自动产生控件的函数

- `ipywidgets`还提供了几个函数用于自动产生控件，最常用的是`interact`函数。
- `interact`函数有两个参数，第一个参数是自定义函数，用于响应控件的操作；第二个参数是`x`关键字参数，`interact`函数会根据参数值产生不同的控件。

| Keyword argument   | Widget      |
|--|-------------|
| <code>True</code> or <code>False</code>  | Checkbox    |
| <code>'Hi there'</code>  | Text        |
| <code>value</code> or <code>(min,max)</code> or <code>(min,max,step)</code> if integers are passed | IntSlider   |
| <code>value</code> or <code>(min,max)</code> or <code>(min,max,step)</code> if floats are passed   | FloatSlider |
| <code>['orange','apple']</code> or <code>`(['one', 1), ('two', 2)]</code>                          | Dropdown    |

不同的关键字参数值会产生不同的控件

```
from IPython.display import display
from ipywidgets import interact
def f(x):
    return x
intSlide = interact(f, x=(1,10))
display(intSlide)
```



自动产生一个IntSlide控件（控件的值实时传输到自定义函数）

```
from IPython.display import display
from ipywidgets import interact
def f(x):
    return x
dropdown = interact(f, x=['北京','上海','广州'])
display(dropdown)
```



x 北京

'北京'

自动产生一个Dropdown控件（控件的值实时传输到自定义函数）



1.Jupyter Notebook基本情况

2.IPython介绍

3.使用ipywidgets添加控件

4.使用Markdown编写文档

- **Markdown**是一种文本格式的标记语言，能够使文本按照一定的格式进行显示。类似于HTML文档，但标签比HTML简单。
- 目前有很多网站及软件平台支持Markdown文档的显示，如Jupyter Notebook、github、简书等。

**RUNOOB.COM**

搜索.....

首页HTMLCSSJAVASCRIPTJQUERYBOOTSTRAPPYTHON3PYTHON2JAVAC C++C#SQL

Markdown 教程

Markdown 教程

Markdown 标题

Markdown 段落格式

Markdown 列表

Markdown 区块

Markdown 代码

Markdown 链接

Markdown 图片

Markdown 表格

Markdown 高级技巧

Markdown 标题 →

## Markdown 教程



Markdown 是一种轻量级标记语言，它允许人们使用易读易写的纯文本格式编写文档。

Markdown 语言在 2004 由约翰·格鲁伯（英语：John Gruber）创建。

Markdown 编写的文档可以导出 HTML 、 Word、图像、PDF、Epub 等多种格式的文档。

Markdown 编写的文档后缀为 `.md` , `.markdown` 。

## Markdown 应用

Markdown 能被使用来撰写电子书，如：Gitbook。

当前许多网站都广泛使用 Markdown 来撰写帮助文档或是用于论坛上发表消息。例如：GitHub、简书、reddit、Diaspora、Stack Exchange、OpenStreetMap 、SourceForge等。

## Markdown菜鸟教程

(<https://www.runoob.com/markdown/md-tutorial.html>)

- Markdown定义的标签只有10几种，包括标题标签、字体标签、目录标签、引用标签、代码标签、图像标签、链接标签等。

## 4.1标题标签

- Markdown标签用#符号的数量表示不同级别的标题，一个#符号表示一级标题，两个#符号表示二级标题，最小标题是六级。注意：#符号和后面的字符串之间要加一个空格。

# Header 1

## Header 2

### Header 3

#### Header 4

##### Header 5

##### Header 6

**Header 1**

**Header 2**

**Header 3**

**Header 4**

***Header 5***

***Header 6***

不同级别标题的表示与显示

## 4.2 字体标签

- Markdown标签定义的字体包括粗体、斜体以及加删除线的字体：
  - 字符串两边各加两个\*符号，表示粗体。
  - 字符串两边各加一个\*符号或\_符号，表示斜体。
  - 字符串两边各加一个或两个~符号，表示加删除线。

**\*\*粗体\*\***

**粗体**

*\*斜体\**

*斜体*

~~~删除线~~~

~~删除线~~

## 4.3 目录标签

- **Markdown**标签用-符号或\*符号表示目录（注意：符号后面需要加一个空格），显示时会在字符串前自动加一个目录符号。
- 通过缩进方式可以对目录进行分层，下个层次的目录比上个层次目录缩进两个空格，不同层次的目录显示不同符号。



缩进两  
个空格

- 1
- 1.1
- 1.2
- 1.3
- 2
- 2.1
- 2.1.1
- 2.1.2
- 2.2

- 1
  - 1.1
  - 1.2
  - 1.3
- 2
  - 2.1
    - 2.1.1
    - 2.1.2
  - 2.2

目录标签的表示与显示

## 4.4 引用标签

- 在字符串前加>符号，表示引用，显示时，字符串前会有一条竖线。可以用多个>符号表示多级引用。

### 以下是引用：

> This is the first level of quoting.

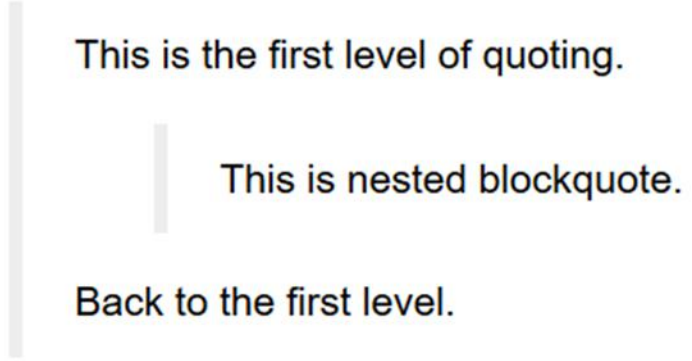
>

> > This is nested blockquote.

>

> Back to the first level.

**以下是引用：**



This is the first level of quoting.

This is nested blockquote.

Back to the first level.

## 4.5代码标签

- Markdown文档中嵌入的代码作为演示，而不是执行，代码的上面和下面添加``符号。

### 以下是程序代码：

``

```
%matplotlib inline
import matplotlib.pyplot as plt
x = [1,2,3,4]
y = [1,4,49,16]
plt.plot(x,y)
plt.show()
```

``

### 代码的执行结果是输出一个图表。

**以下是程序代码：**

```
%matplotlib inline
import matplotlib.pyplot as plt
x = [1, 2, 3, 4]
y = [1, 4, 49, 16]
plt.plot(x, y)
plt.show()
```

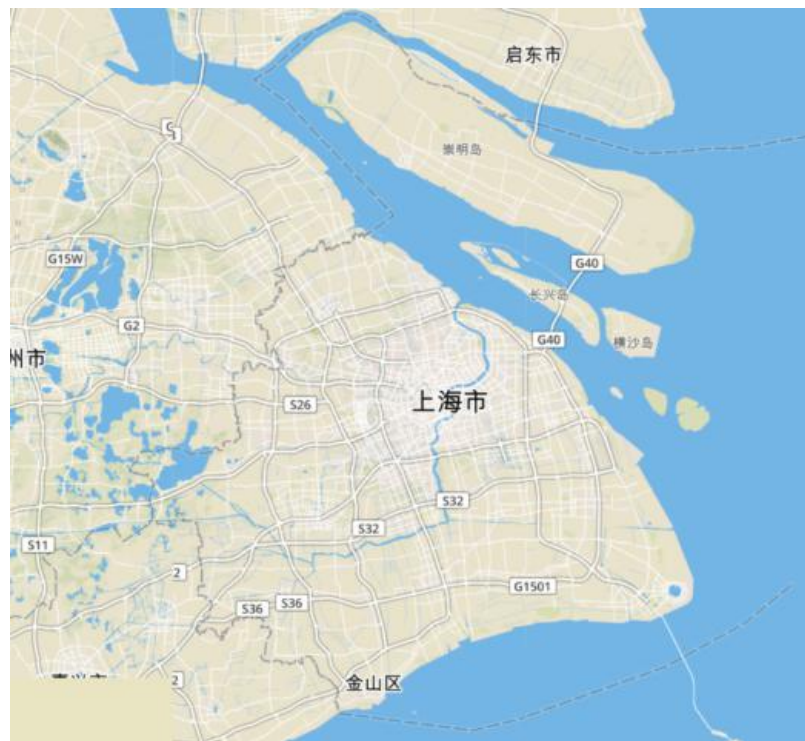
**代码的执行结果是输出一个图表。**

## 4.6 图像标签

- Markdown文档不存储实际图像，只是存储图像的url。
- 图像标签的形式为： `![alt](url)`，(url)是图像的url，[alt]是图像不能显示的情况下显示的文本，可以是空值。注意：[alt]和(url)之间不能有其它符号。

- 图像的url有如下几种形式：
  - 图像文件和notebook在同一目录下，写法为：img.jpg。
  - 图像文件在notebook同一目录下的images文件夹中，写法为：images/img.jpg。
  - 图像文件所在的文件夹和notebook文件所在的文件夹在同个目录下，写法为：../images/img.jpg。
  - 如果图像来自网上，写法为：  
src="http://www.baidu.com/pic/img.jpg"。

!["上海地图"]([http://119.3.40.193:8080/gis\\_development/photos/shanghai\\_map.png](http://119.3.40.193:8080/gis_development/photos/shanghai_map.png))



图像标签的表示与显示

## 4.7 链接标签

- 链接标签的形式为：[链接文本](url)，显示时，链接文本以特殊符号显示，点击链接文本将打开链接url的网页。注意：两者之间不能有其它符号。

[华东师范大学](http://www.ecnu.edu.cn)位于上海市，创建于1951年。

[华东师范大学](http://www.ecnu.edu.cn)位于上海市，创建于1951年。

## 4.8段落标签

- Markdown用两个或两个以上空格加回车表示新的段落，单一回车符号在Markdown中将被忽略。



## 4.9分隔线标签

- 在一行中用三个或三个以上的星号、减号、下划线将显示一个分隔线，注意：行内不能有其它符号。

## 4.10表格标签

- Markdown中的表格使用“|”标签来分隔不同的单元格，使用“-”标签来分隔表头和其它行。
- 如要设置单元格中文本的对齐方式，则在“-”标签前后加“:”标签，“-:”表示右对齐，“:-”表示左对齐，“:-:”表示居中对齐。

```
|城市|人口（万人）|  
|:-:|:-:|  
|上海|2418|  
|北京|2170|  
|广州|1449|
```

| 城市 | 人口（万人） |
|----|--------|
| 上海 | 2418   |
| 北京 | 2170   |
| 广州 | 1449   |

表格标签的表示与显示

- <https://www.datacamp.com/community/tutorials/scikit-learn-tutorial-baseball-1>

- 1.安装python环境
- 2.尝试运行一个.py文件
- 3.了解棒球数据分析思维

\*Python

# Reading assignment

- 《魔球》