《Python与数据挖掘》

第1章 数据挖掘概述

讲师：武永亮

目录

数据挖掘简介

- 广义的数据挖掘是指针对收集的大规模数据，应用整套科学工具和挖掘技术（如数据、计算、可视化、分析、统计、实验、问题定义、建模与验证等），从数据之中发现隐含的、对决策有参考意义的信息、价值和趋势。
- 随着计算机技术的全面发展，企业生产、收集、存储和处理数据的能力大大提高，数据量与日俱增。数据的积累实质上是企业的经验和业务的沉淀。越来越多的企业引入“数据思维”——不只是依赖于数据的统计分析，更强调对数据进行挖掘，期待从这一“未来世界的石油”中发现潜在的价值。这一迫切的“开采”需求在世界范围内酝酿了一次“大数据”变革。

数据挖掘简介

- 数据挖掘确是21世界最具话题性的技术之一，包含数据预处理、算法应用、模型评价、结果检验等多个部分，并依靠其丰富的内涵向外延伸出数据分析、数据ETL、机器学习等多个领域。
- 数据挖掘的整体过程如下图：

目录

工具简介

- 数据挖掘软件的历史并不长，甚至连“数据挖掘”这个术语也只是在19世界90年代中期才正式提出。如今，商用数据挖掘软件和开源工具都已经非常成熟，不仅提供易用的可视化界面，还集成了数据处理、建模、评估等一整套功能。
- 部分开源的数据挖掘软件，采用可视化编程的设计思路。之所以这么做，是因为它能足够灵活和易用，更适合缺乏计算机科学知识的用户。如WEKA和RapidMiner。
- 当用户拥有较多特定的分析需求，或正在自行实现一个改进的机器学习算法时，脚本型语言如Python和R将更符合需要。同时，脚本型语言兼具运行效率和开发效率，支持敏捷型的迭代更新。

1、WEKA

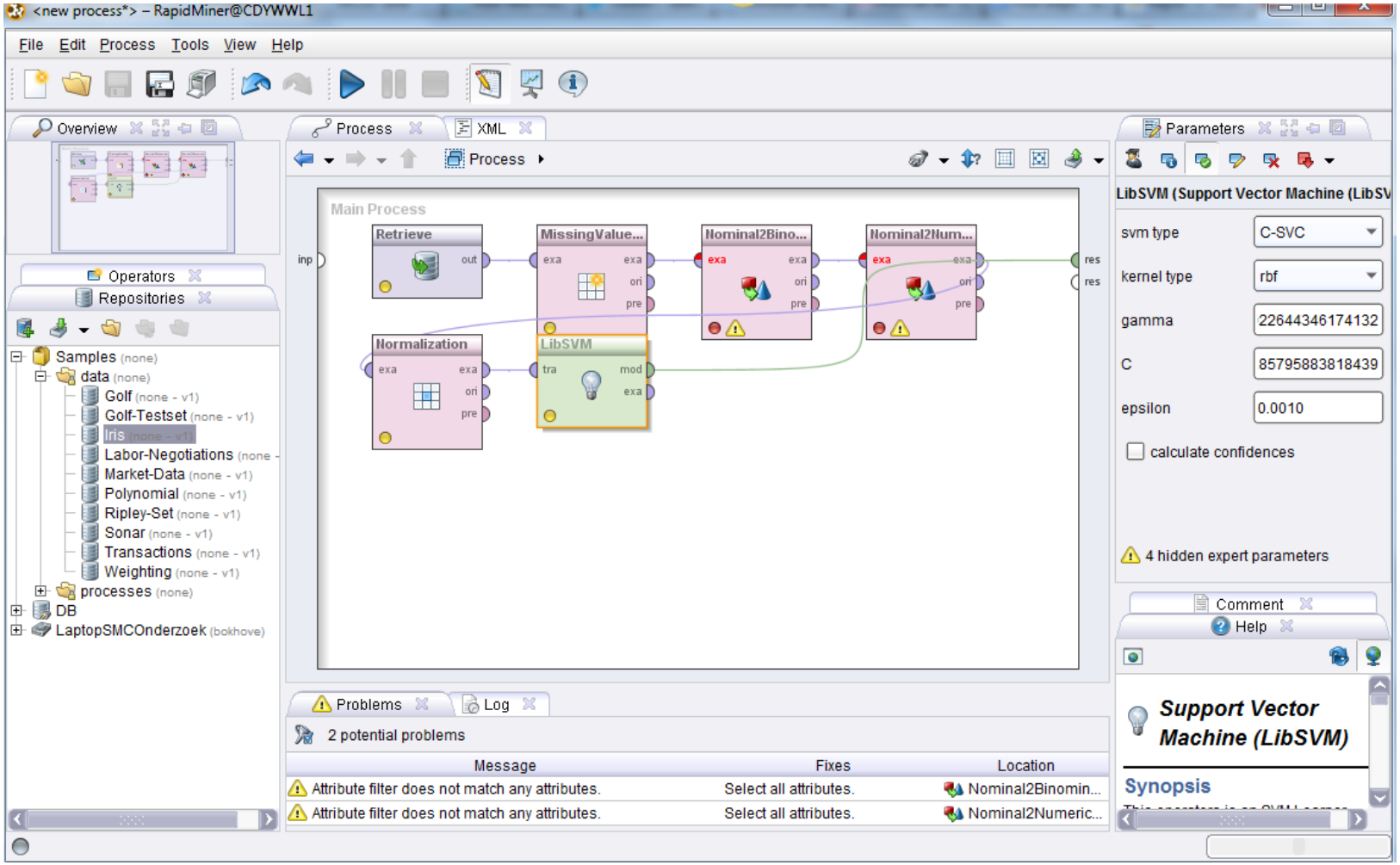
- 用Java编写的WEKA是一款知名的数据挖掘工作平台，它为解决数据挖掘任务的实际需求而生，集成了大量能处理数据挖掘任务的机器学习算法，这些算法能被用户直接应用于数据集之上。
- WEKA 支持多种标准数据挖掘任务，包括数据预处理，分类、回归分析、聚类、关联规则等算法的应用，以及特征工程和可视化。
- WEKA欢迎界面如下图：

2、RapidMiner Studio

- RapidMiner的目标是：“成为一个能将数据变成宝贵的战略资产的现代平台”，已被广泛使用于商业应用、学术研究、教育、敏捷开发等领域。
- RapidMiner是一个支持数据挖掘、文本挖掘、机器学习、商业分析等任务的集成环境。
- RapidMiner是基于WEKA二次开发的应用，这意味着它可以调用WEKA中的各种分析组件。

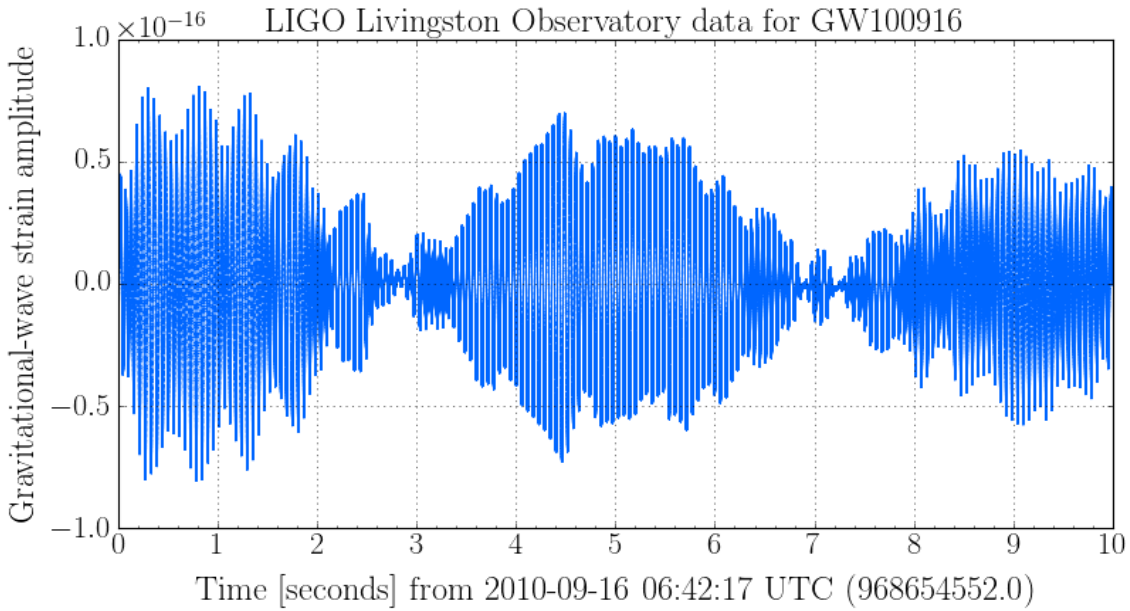
2、RapidMiner Studio

- RapidMiner Studio工作界面如下图：



3、Python

- Python是一门编程语言。随着NumPy、SciPy、Matplotlib和Pandas等众多程序库的开发，Python在科学计算和数据分析领域占据着越来越重要的地位。在大多数数据任务上，Python的运行效率已经可以媲美C/C++语言。
- 利用公开引力波数据绘制波形图如下图：

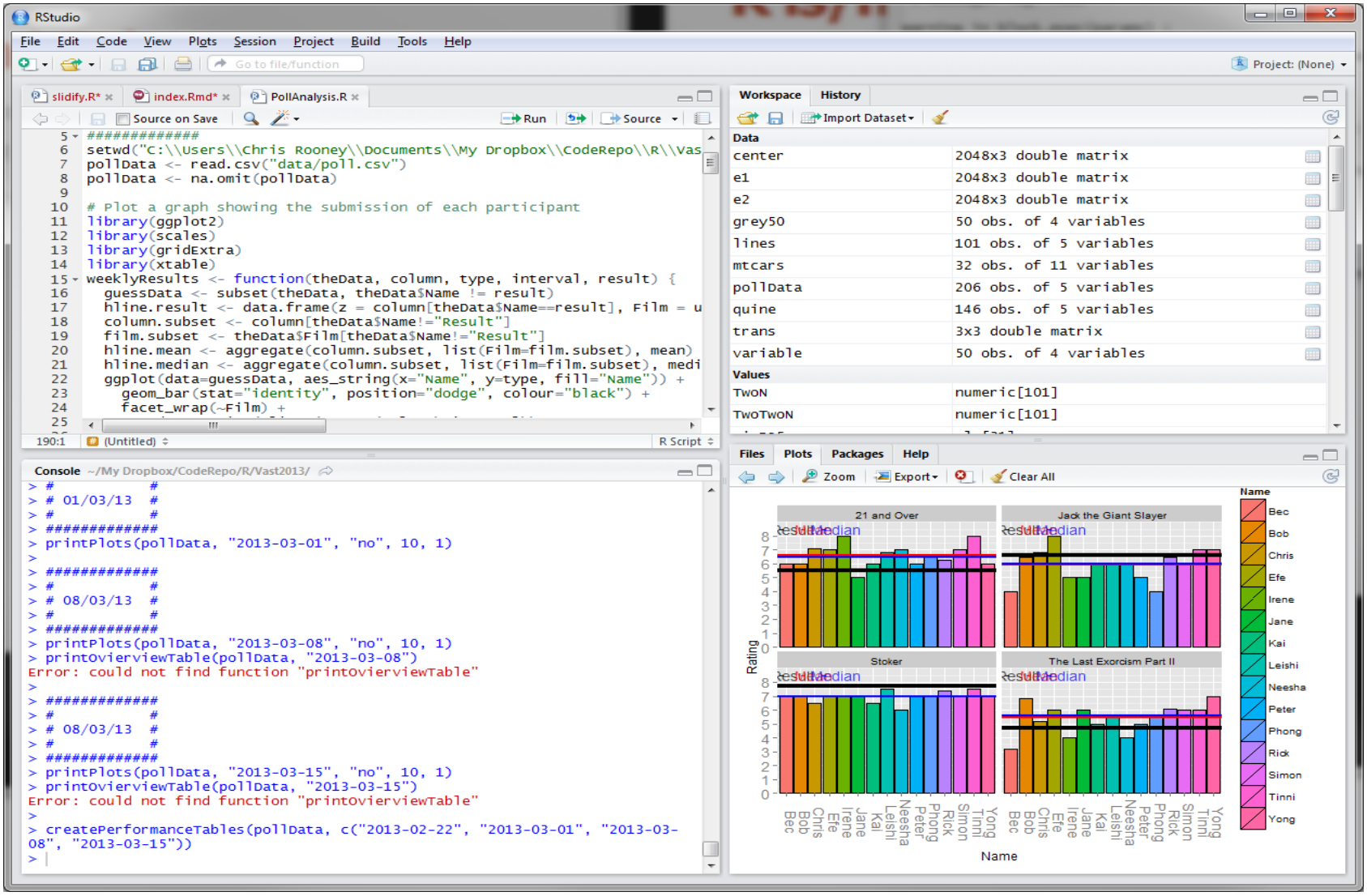


4、R语言

- R语言是一种为统计计算和图形显示而设计的语言环境，是贝尔实验室的Rick Becker, John Chambers和Allan Wilks开发的S语言的一种实现，包含一系列统计与图形显示工具。
- R语言至少拥有以下优势：
  - 方便地从各种类型的数据源中获取数据；
  - 高可拓展性；
  - 出色的统计计算功能；
  - 顶尖水准的制图功能；
  - 不断贡献强大功能的开源社区。
- 它与Python同属数据挖掘主流编程语言，而从功能与代码风格的角度来评价，R与MATLAB是最像的

4、R语言

- R-Studio工作界面如下图：



Python的优点

Python是彻底的面向对象语言，能被方便地集成到其他需要脚本语言的程度之内。

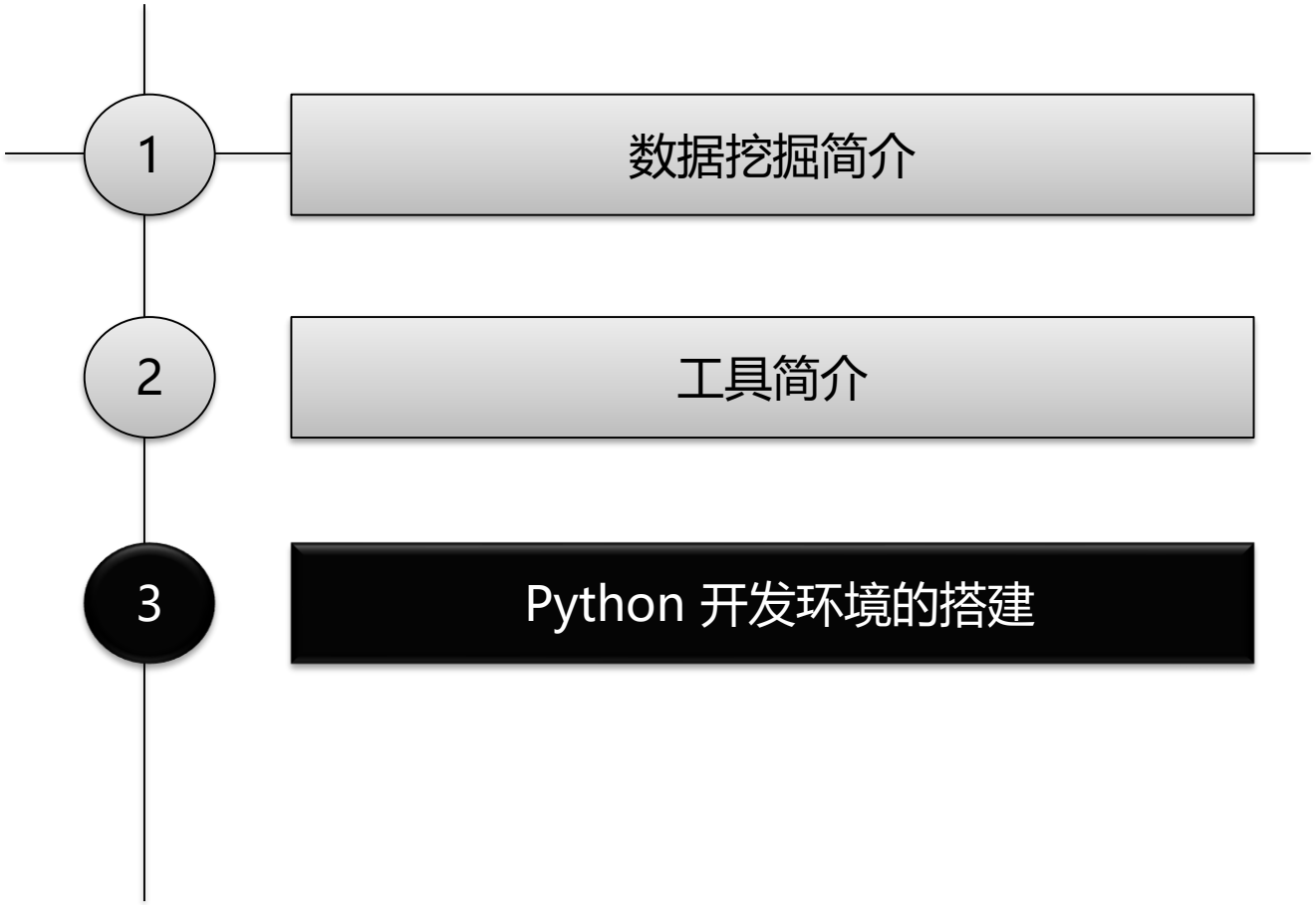
Python语法简洁，代码可读性强，开发效率高，能很好地连接“想法”和“实现”两个部分。

Python是一门多功能的语言，能够胜任脚本，博客网站搭建，嵌入使用，客户端等的开发工作。

Python的缺点

- Python唯一的缺点是，作为脚本语言与C/C++这类编译语言相比，Python的运行效率还不够快。
- 目前Python的标准实现方式是将源代码的语句编译（或者说是转换）为字节码的形式，再将字节码解释出来。由于字节码是一种与平台无关的形式，字节码具有可移植性。然而，因为Python没有将代码编译成底层的二进制代码，所以一些Python的程序将会比像C这样的完全编译语言慢一些。
- 在性能要求非常高的情况下，我们可以分离一部分需要优化速度的应用，将其转换为编译好的扩展形式，并在整个系统中使用Python脚本将这部分应用连接起来，仍然可以兼顾开发效率和运行效率。

目录



Python安装

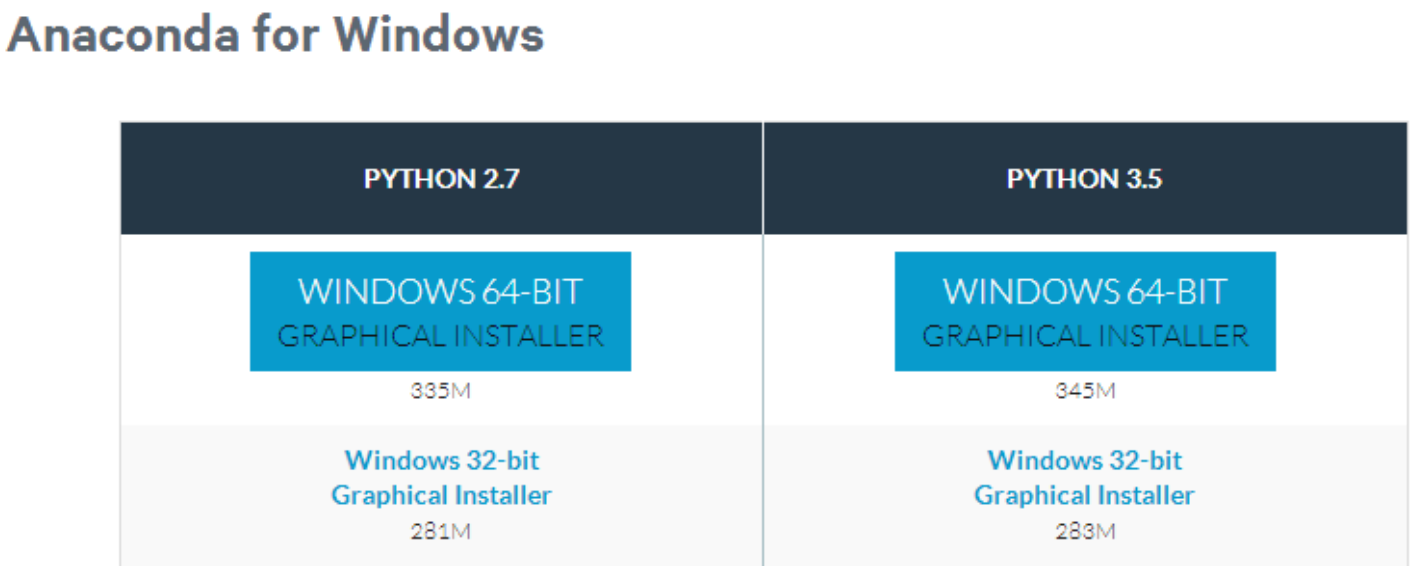
- 所谓编程语言，意指“与计算机交流时使用的语言”。它是一种被标准化的交流技巧，用于连接程序员的思维和计算机的操作。学习编程语言的第一关，就是安装和环境配置。我们必须与计算机约定如何理解代码、指令和语法，才能够顺利地与计算机交流，赋予它复杂的功能。Python便是其中的一种“方言”。
- 对于新手，Python及其第三方模块在安装环节有许多已知的难题。比如源码编译的安装方式、环境变量的配置、不同模块之间的版本依赖问题。如果陷入其中的某一个泥潭之中，将浪费大量初学者的时间，消磨热情；当然，如果能独立克服，就能熟悉相关的重要概念，大有裨益。

Python安装

- 为了能顺利进行后续内容的学习，以及避免不必要的麻烦，我们将采用更加简单的安装方式。本书使用的是Python的科学计算发行版——Anaconda. 除Python本身之外，Anaconda囊括了科学计算和数据分析所需的主流模块，独立的包管理工具Conda, 以及两款不同风格的编辑器Jupyter和Spyder. 它具有开源精神且支持学术用途的免费额外性能提升。官方软件下载地址为：https://www.continuum.io/downloads

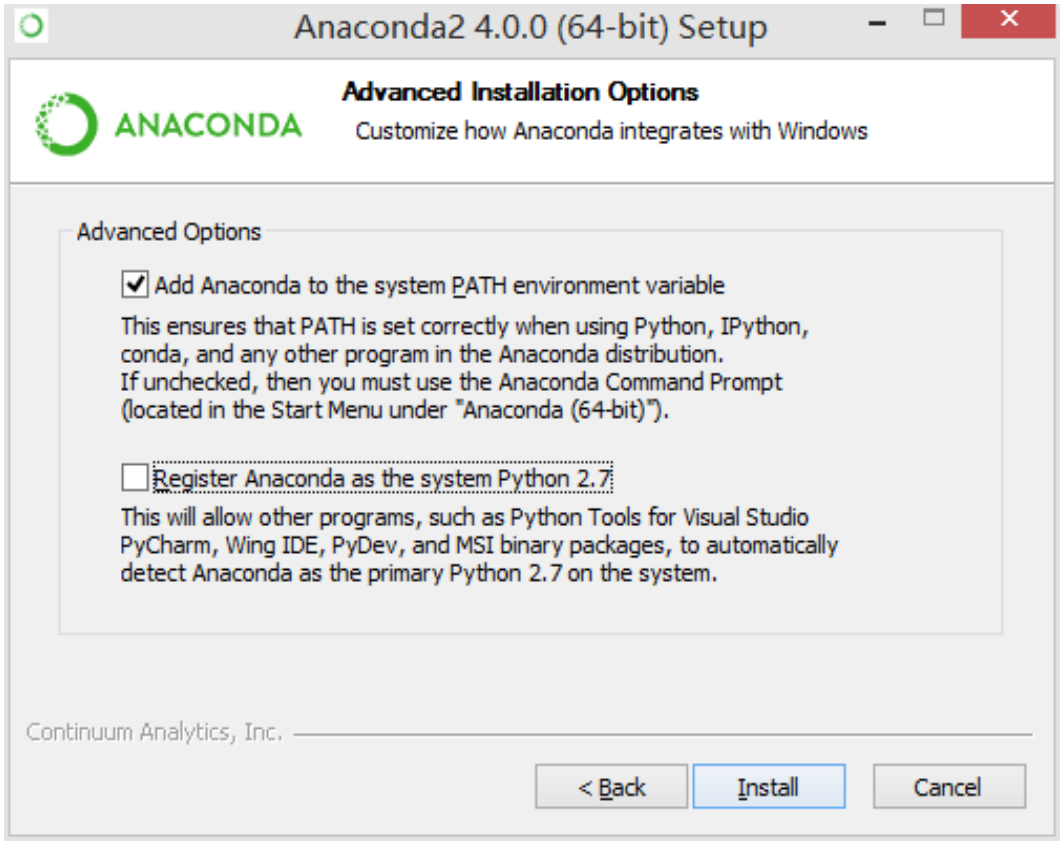
Windows下安装Python

- Anaconda的存在使得在Windows系统中安装Python得到极度简化，直接前往官方网站找到对应的下载内容如下图，并选择Python 2.7对应的安装包，注意区分32位和64位的版本。



Windows下安装Python

- 下载后运行Anaconda的安装程序，这里大部分的操作和一般软件的安装无异，需要注意的是：如下图 所示，Anaconda默认会自动改写环境变量配置参数，使得用户能在任何的路径下使用Python命令行模式。
- 如果自行安装原始的Python版本，极容易忽略这一步，从而走入思维的盲区，导致永远不能自行安装成功。这也是我们推荐使用科学计算发行版Anaconda的原因。



Linux下安装Python

- 大多数Linux发行版，如CentOS、Debian、Ubuntu等，都已经自带了Python 2.x的主程序。因此，额外安装Anaconda需要做好管理的工作，避免两个不同版本的Python冲突，导致不必要的错误。如果确定内置版Python能够兼容书中代码，亦可不额外安装Anaconda.
- 下面介绍如何安装Anaconda，并避免与内置版的Python冲突。本教程以Ubuntu 16.06为例。

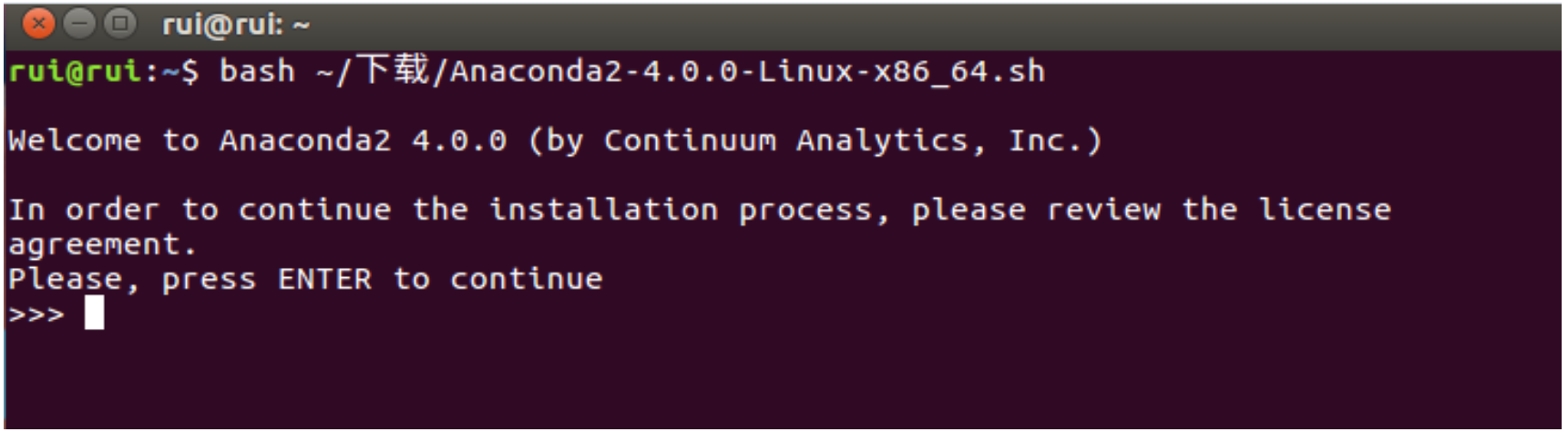
a) 前往官方网站下载对应版本的Anaconda，默认情况下，Linux会自动将下载所得文件归档在“下载”文件夹中。

b) 假设下载所得文件在“下载”这一文件夹中，如果不是，请替换路径，并输入下面的命令，以执行批处理指令，安装Anaconda.

```
$ bash ~/下载/Anaconda2-4.0.0-Linux-x86_64.sh
```

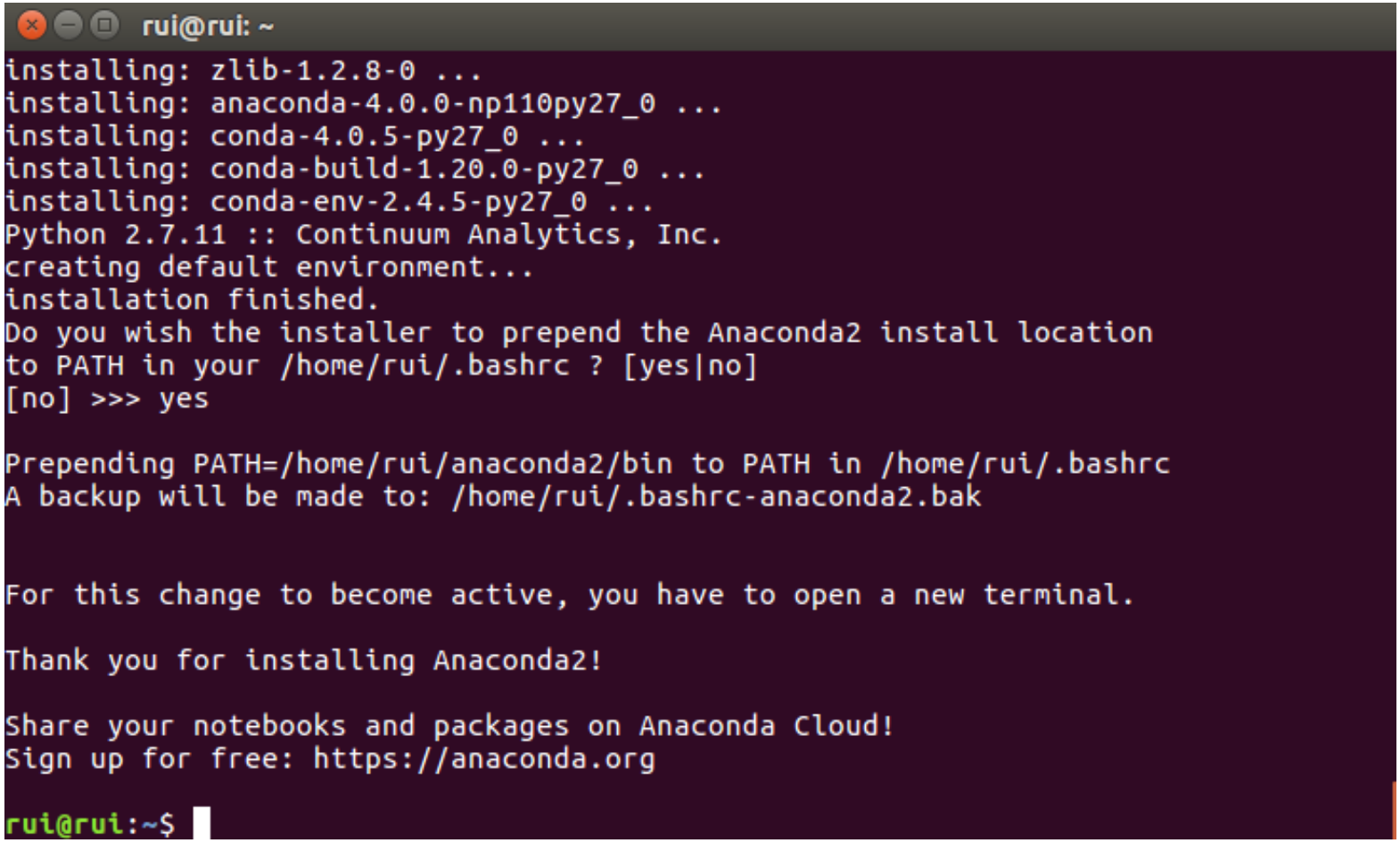
Linux下安装Python

- 安装过程中，将会在屏幕上打印出用户协议许可，你需要利用Enter继续阅读。阅读至文件末尾，输入yes并敲击Enter键来表示你同意以上内容并使用默认路径开始安装。如下图所示：



Linux下安装Python

c) 如下图，输入yes来确认允许Anaconda为你自动配置环境变PATH.



Linux下安装Python

- 当看到下图中的欢迎信息之后，代表已经成功安装Anaconda. 这里我们执行下面的命令，将Anaconda的位置加载至环境变量PATH的开头，使得当我们使用Python时，总是优先使用Anaconda版。
- $ export PATH=”$HOME/anaconda2/bin:$PATH
  之后，我们可以直接输入python，以检查我们能够正确使用Anaconda版的Python.

```
Thank you for installing Anaconda2!

Share your notebooks and packages on Anaconda Cloud!
Sign up for free: https://anaconda.org

rui@rui:~$ export PATH="$HOME/anaconda2/bin:$PATH"
rui@rui:~$ echo $PATH
/home/rui/anaconda2/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:
/bin:/usr/games:/usr/local/games:/snap/bin:/usr/lib/jvm/java-8-oracle/bin:/usr/l
ib/jvm/java-8-oracle/db/bin:/usr/lib/jvm/java-8-oracle/jre/bin
rui@rui:~$ python
Python 2.7.11 |Anaconda 4.0.0 (64-bit)| (default, Dec  6 2015, 18:08:32)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out: http://continuum.io/thanks and https://anaconda.org
>>>
```

Mac下安装Python

- 类似Windows下的安装，Mac OS X系统用户可以直接前往官方网站下载一个图形化安装程序。同时，因为OS X系统是基于UNIX内核开发的，所以我们也能够打开终端，通过命令行的方式来安装。这里主要叙述利用终端安装的方法。

1、下载OS X下对应版本的Anaconda, 注意：利用终端安装Anaconda实际上是在进行“源码编译”。后续步骤中需要的是二进制文件（Command-Line Installer），而非图形化的安装界面（Graphical Installer）。

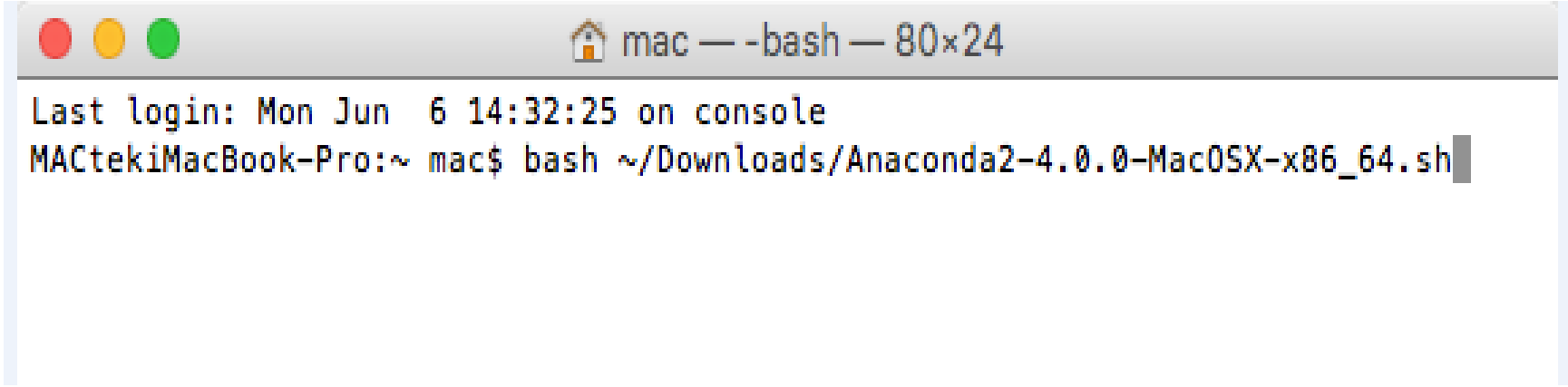
Mac下安装Python

2、按下Alt + Space，打开Search界面，输入terminal，点击搜索出来的“Terminal”（终端）图标。

3、输入下面的命令，执行批处理指令，安装Anaconda.

$ bash~/Downloads/Anaconda2-4.0.0-MacOSX-x86_64.sh

- 安装过程中，将会在屏幕上打印出用户协议许可，你需要利用Enter继续阅读。阅读至文件末尾，输入yes并敲击Enter键来表示你同意以上内容并使用默认路径开始安装。



Mac下安装Python

4、输入yes来确认允许Anaconda为你自动配置环境变量PATH.

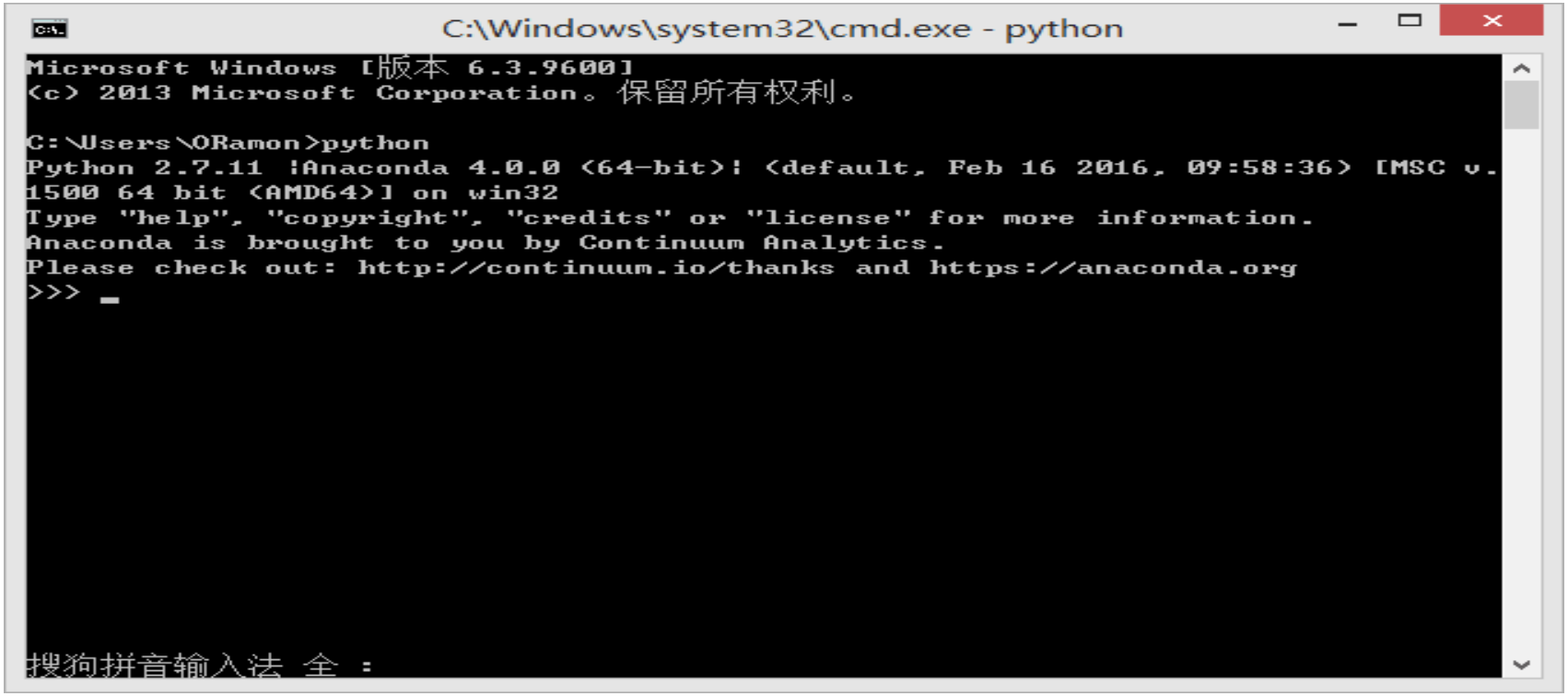
5、与Linux下安装类似，同样需要将Anaconda的位置加载至环境变量PATH的开头，使得当我们使用Python时，总是优先使用Anaconda版。

- $ export PATH="$HOME/anaconda2/bin:$PATH"

  之后，我们可以直接输入python，以检查我们能够正确使用Anaconda版的Python.

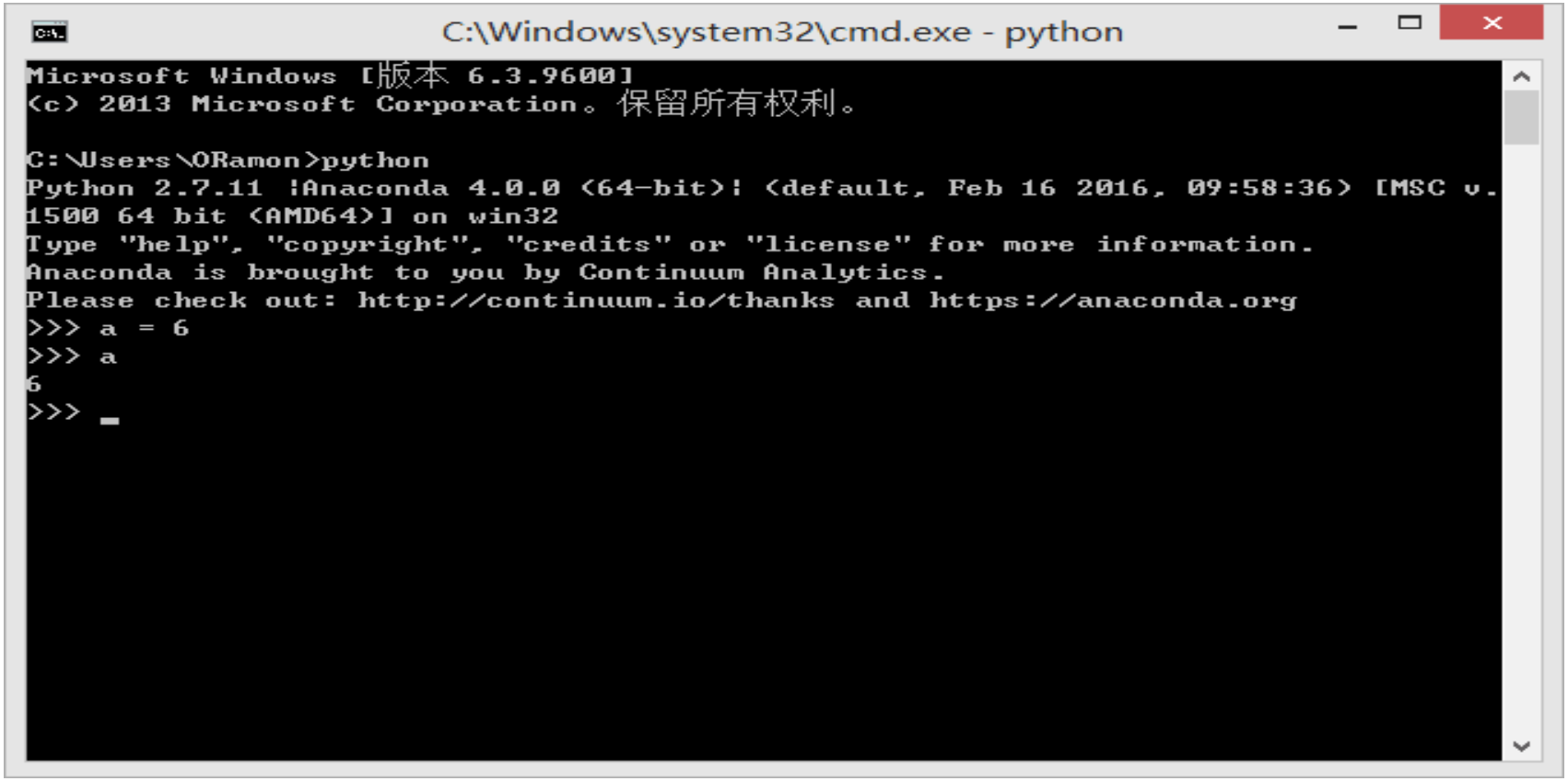
Python窗口

- 命令行版本的Python Shell-Python(Command)
- 以Windows系统为例，安装Python后，你可以在开始菜单中，找到对应的Command Line版本的Python Shell，或者同时按下Win + R键，输入cmd并按回车，打开命令窗口。在命令窗口中输入python即可使用进入Python的命令行模式。



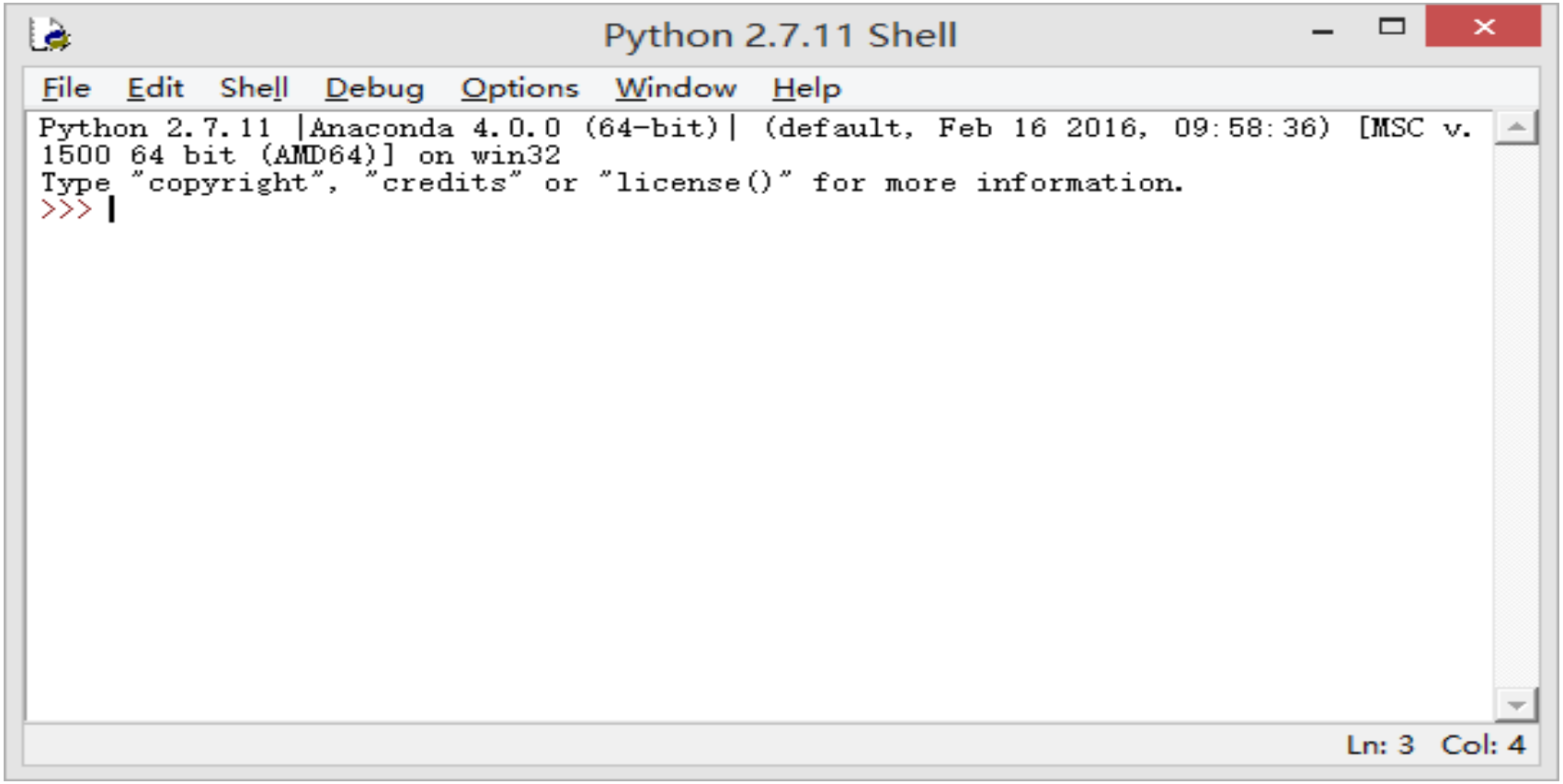
Python窗口

- 其中，可以看到对应的Python版本信息和系统信息。我们可以在标识符>>>后面输入代码，程序就会马上返回一个结果。
- Python Shell是交互式Shell，交互式是指当你输入代码到Python Shell中时就可以动态地看到相应的返回结果。



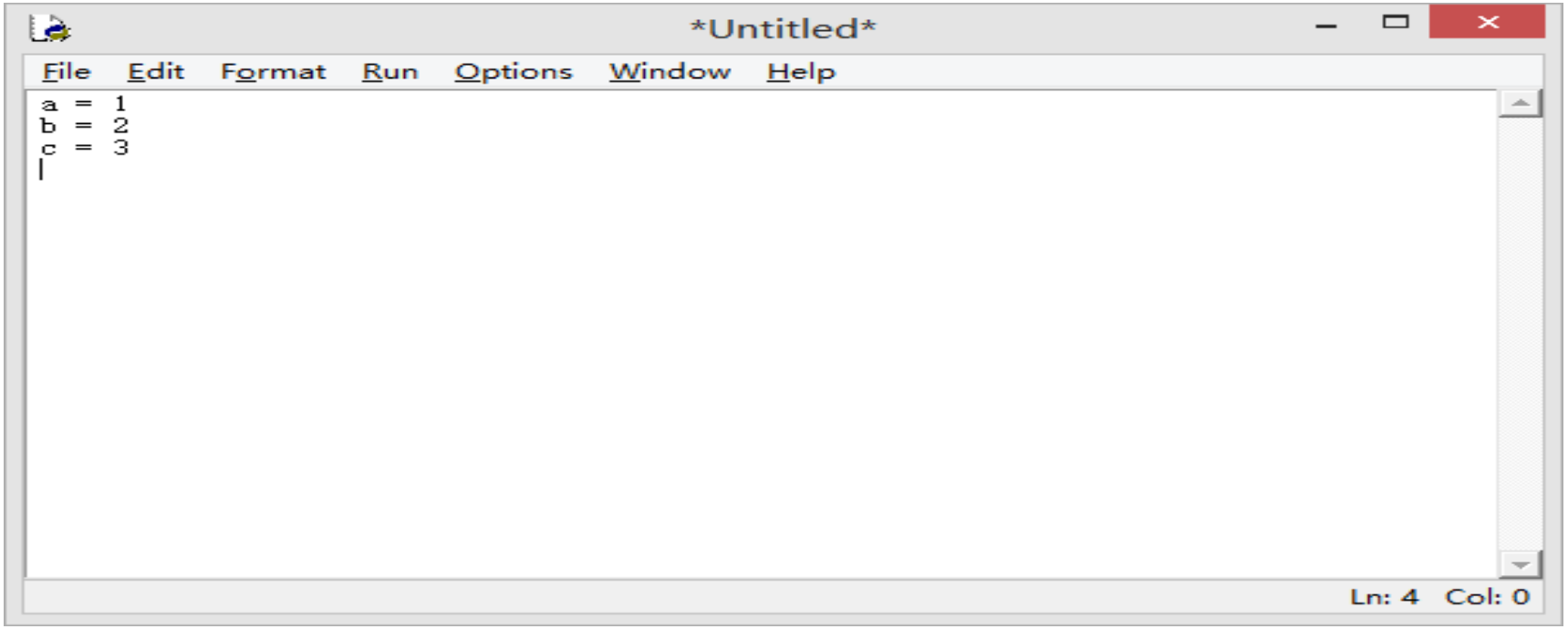
带图形界面的Python GUI

- 带图形界面的Python Shell-IDLE(Python GUI)
- 下面将要介绍的，是带图形界面的Python GUI. Windows下在所有程序上搜索IDLE，就可以直接打开Python Shell – IDLE。打开后界面如下：



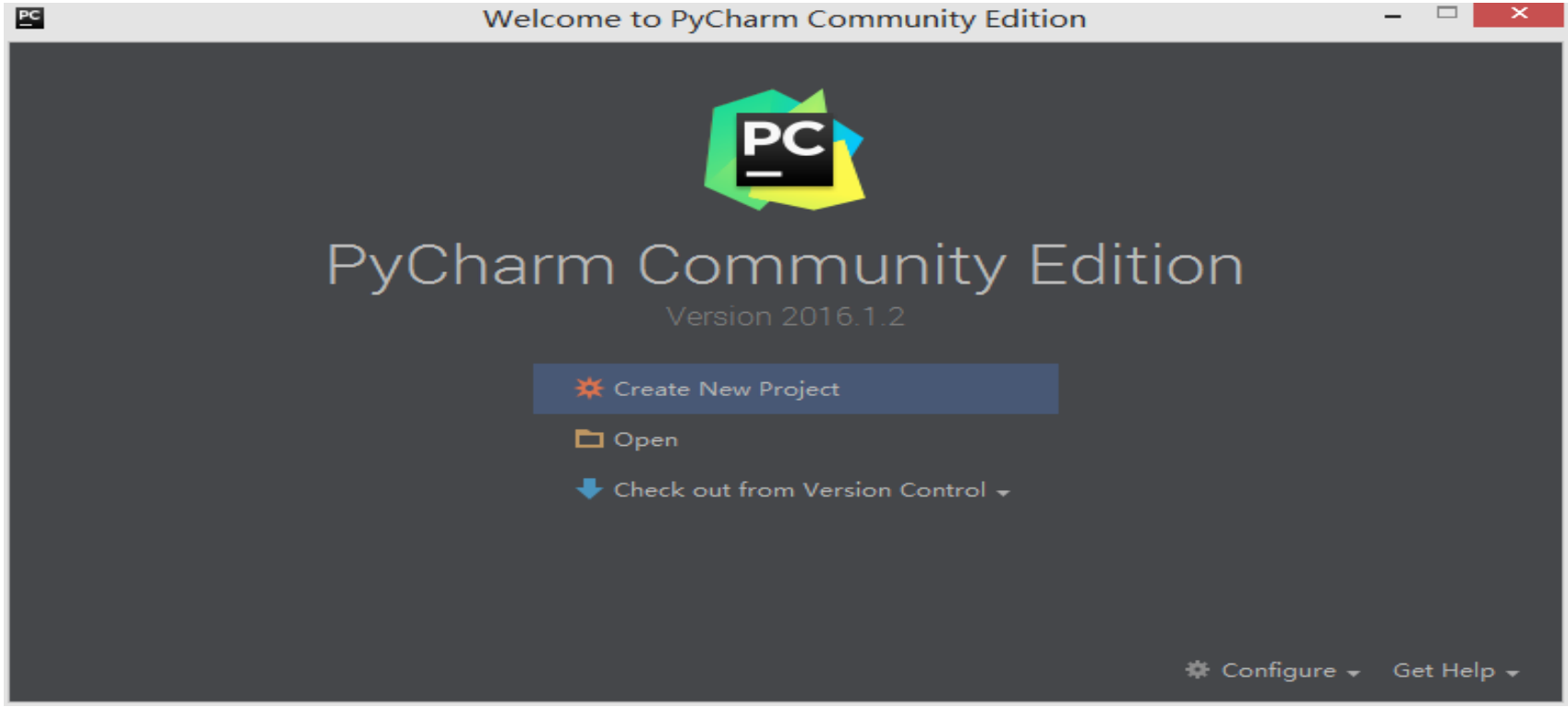
带图形界面的Python GUI

- 在这界面上可以通过菜单栏的File –> New File 创建Python脚本，能够在Python脚本上写多行代码，保存为.py文件后并能够运行该脚本，而在Command Line上运行多行代码只能一行接着一行输入并按回车输出，显得十分繁琐。运行Python脚本实际上也是按顺序运行每行的代码，运行脚本后将回到Python GUI界面，这时候Python已经存储脚本运行后的数据，可以在界面上继续输入代码。



第三方Python IDE

- IDE是集成开发环境（Integrated Development Environment）的英文简称。而第三方IDE通常聚合了更强大的功能，包括代码版本管理、项目代码管理、代码自动补全等。PyCharm就是这样一个跨平台的，多功能的集成开发环境，主要分为免费社区版和付费商业版。
- PyCharm社区版如下图：

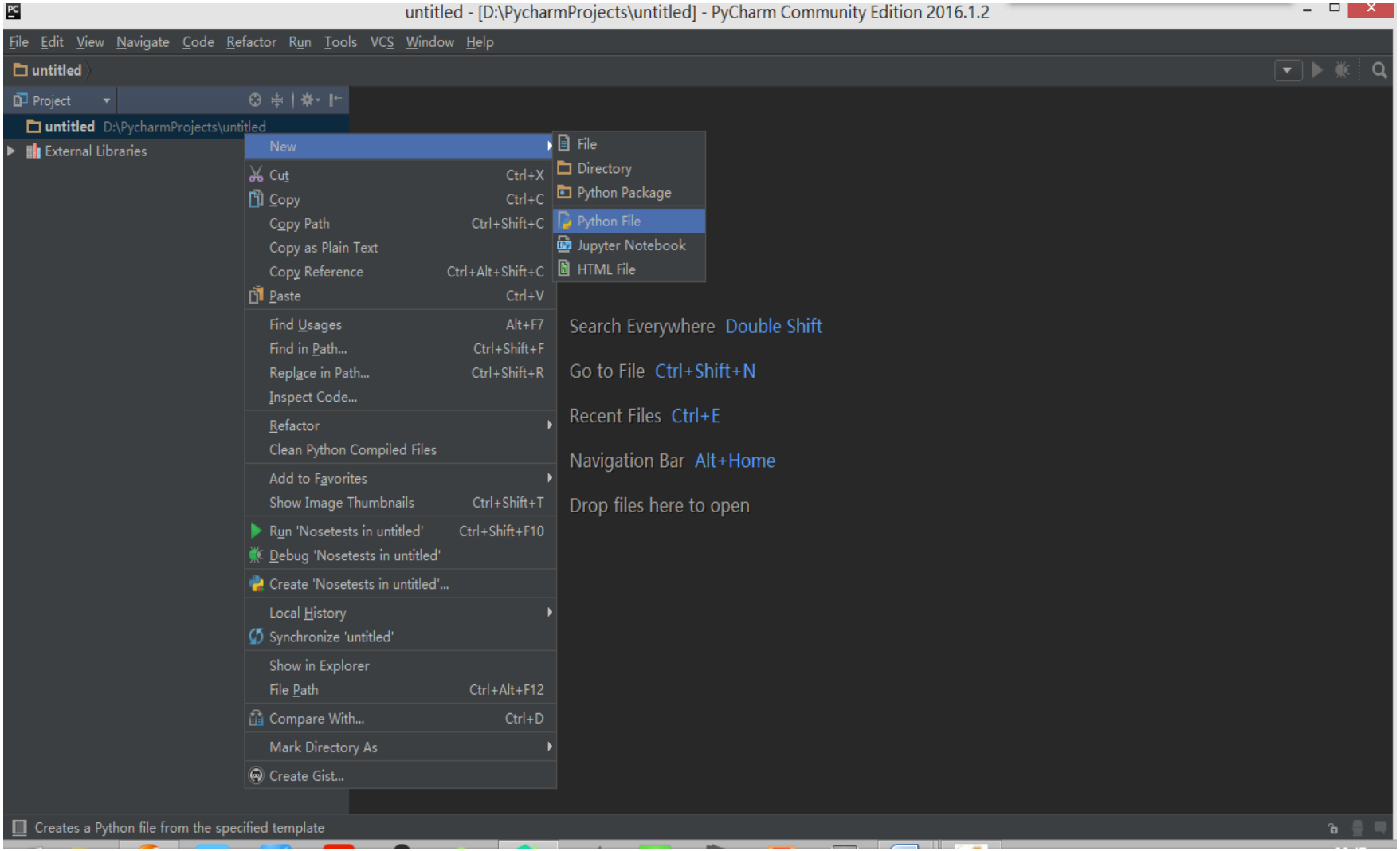


第三方Python IDE

- 在选择创建项目以及确定项目存储路径之后，我们能看到一个清晰，简洁的界面。
- 左侧栏是项目管理窗口，负责组织Python实现的项目中所涉及的全部代码和数据文件。
- 右边是正式的编辑区。在选择创建新的Python File之后，将能配合内置的自动补全，代码提示，调试运行功能进行代码的编辑、改正和优化。
- 同时，它还能自动结合Git进行代码版本控制。有兴趣的话可以自行查找资料。当我们需要做一个大型项目，代码量较多时，用带有项目管理功能的PyCharm会更加方便。

第三方Python IDE

- PyCharm新建Python File 如下图：



Thank You!