

AI 神經網絡深層學習與交易策略 應用實作班

Deep Learning on Trading Strategy Workshop

昀騰金融科技
技術長
董夢雲 博士

目 錄

Part I Python 介紹與古典交易策略說明

一、Python 的安裝與基本使用

二、Python 使用與數值運算套件 Numpy

三、資料庫套件 Pandas 與繪圖套件 MatPlot

四、價格資料來源與套件 TuShare

五、MySQL 的安裝與使用

六、古典程式交易策略與回溯測試

Part II 時間數列與機器學習策略介紹

七、時間數列預測基礎

八、主成分分析與時間數列預測

九、ARIMA 與 GARCH 的合併預測

十、機器學習模型(一)：羅吉斯迴歸與鑑別分析

十一、機器學習模型(二)：支持向量機

十二、機器學習模型(三)：決策樹與隨機森林

十三、機器學習模型(四)：配對交易的選對

Part III 神經網絡的學習理論與交易策略的應用

十四、單層與多層神經元的學習與預測應用

十五、隨機梯度下降(SGD)與反向傳播法

十六、卷積神經網路(CNN)的預測應用

十七、遞迴神經層(RNN)

十八、長短期記憶模型(LSTM)的預測應用

十九、系統布署、GPU 使用與未來發展

昀騰金融科技股份有限公司

技術長
金融博士、證券分析師

董夢雲 Andy Dong



ID:50917111

Line/WeChat:andydong3137

E:andydong1209@gmail.com

<https://github/andydong1209>

M:(T)0988-065-751 (C)1508-919-2872

10647 台北市大安區辛亥路一段 50 號 4 樓

現職：國立台灣大學財務金融研究所兼任教授級專家
國立台灣科技大學財務金融研究所兼任助理教授
台灣金融研訓院 2020 年菁英講座

經歷：中國信託商業銀行交易室研發科主管
凱基證券風險管理部主管兼亞洲區風險管理主管
中華開發金控、工業銀行風險管理處處長
永豐金控、商業銀行風險管理處處長
永豐商業銀行結構商品開發部副總經理

學歷：國立台灣大學電機工程學系學士
國立中央大學財務管理學研究所博士
專業：證券暨投資分析人員合格(1996)

專長：風險管理理論與實務，資本配置與額度規劃、資產負債管理實務
外匯與利率結構商品評價實務，股權與債權及衍生商品評價實務
GPU 平行運算與結構商品系統開發，CUDA、OpenCL
CPU 平行運算與 ALM 系統開發，C#/C++/C、.Net Framework、SQL
人工智慧(Deep Learning)交易策略開發，Python、Keras、TensorFlow

Part I Python 介紹與古典交易策略說明

主題一、Python 的安裝與基本使用

一、Python介紹

二、Python的安裝

三、Python的基本使用

四、開發工具

一、Python 介紹

◆ 西方大型投資銀行策略性的使用 Python 語言，搭配其他已使用的技術，來建立、提升、與維護他們一些核心的 IT 系統。

- 2014 年 Bank of American Merrill Lynch 推出了 Quartz 專案，
- 2014 年 JP Morgan Chase 則推出了 Athena 專案。

(一)Python 簡史

- ◆ Python 的創始人為吉多・范・羅蘇姆（Guido van Rossum）。
- 1989 年的聖誕節期間，吉多・范・羅蘇姆為了在阿姆斯特丹打發時間，決心開發一個新的腳本 (Script) 程式，作為 ABC 語言的一種繼承語言。
 - ✓ 之所以選中 Python 作為程式的名字，是因為他是 BBC 電視劇—蒙提・派森的飛行馬戲團（Monty Python's Flying Circus）的愛好者。
- ABC 是由吉多參加設計的一種教學語言。
 - ✓ 吉多認為 ABC 這種語言非常優美和強大，是專門為非專業程式設計師設計的。
 - ✓ ABC 語言並沒有成功，吉多認為是非開放造成的。
- 吉多決心在 Python 中避免這一錯誤，並取得了非常好的效果，完美結合了 C 和其他一些語言。

◆ Python 在 Mac 機上完成第一個實現。

- Python 是從 ABC 發展起來，主要受到 Modula-3 的影響。
- 結合了 Unix shell 和 C 的習慣。

◆ 目前吉多仍然是 Python 的主要開發者，決定整個 Python 語言的發展方向。

- Python 社群經常稱呼他是仁慈的獨裁者。

(二)Python特性

- ◆ Python 2.0 於 2000 年 10 月 16 日發布，
 - 增加了實現完整的垃圾回收，並且支援 Unicode。
- ◆ Python 3.0 於 2008 年 12 月 3 日發布，此版不完全相容之前的 Python 原始碼。
 - 很多新特性後來也被移植到舊的 Python 2.6/2.7 版本。
- ◆ Python 是完全物件導向的語言。
 - 函式、模組、數字、字串都是物件。
 - ✓ 完全支援繼承、重載、衍生、多重繼承，有益於增強原始碼的複用性。
 - Python 支援重載運算符，
 - ✓ Python 也支援泛型設計。
 - Python 對函數式設計只提供了有限的支援。
 - ✓ 有兩個標準庫（functools, itertools）提供了與 Haskell 和 Standard ML 中類似的函數式程式設計工具。

◆ Python 被粗略地歸類為「腳本語言」(Script Language) ,

- 實際上一些大規模軟體開發計畫，例如 Zope 、 Mnet 及 BitTorrent 都廣泛地使用它。

◆ Python 的支持者較喜歡稱它為一種高階動態程式語言，

- 「腳本語言」泛指僅作簡單程式設計任務的語言，如 shell script 、 VBScript 等，
 - ✓ 只能處理簡單任務的程式語言，並不能與 Python 相提並論。

◆ Python 本身被設計為可擴充的，並非所有的特性和功能都整合到語言核心。

- Python 提供了豐富的 API 和工具，能夠輕鬆地使用 C 、 C++ 、 Cython 來編寫擴充模組。
 - ✓ Python 編譯器本身也可以被整合到其它需要腳本語言的程式內。
- 很多人把 Python 作為一種「膠水語言」(Glue Language) 使用。
 - ✓ 使用 Python 將其他語言編寫的程式進行整合和封裝。

- ◆ 在 Google 內部的很多專案，例如 Google App Engine，使用 C++編寫效能要求極高的部分，然後用 Python 或 Java/Go 呼叫相應的模組。
 - 《Python 技術手冊》的作者馬特利（Alex Martelli）說：「2004 年，Python 已在 Google 內部使用。Google 召募許多 Python 高手，但在這之前就已決定使用 Python。他們的目的是儘量使用 Python，在不得已時改用 C++；在操控硬體的場合使用 C++，在快速開發時候使用 Python。」

(三)Python的禪

- ◆ Python 的設計哲學是：「優雅」、「明確」、「簡單」。
 - Python 開發者的哲學是：「用一種方法，最好是只有一種方法來做一件事」。
- ◆ 在設計 Python 語言時，如果面臨多種選擇，Python 開發者一般會拒絕花俏的語法，而選擇明確沒有或者很少有歧義的語法。
 - 這些準則被稱為「**Python 的禪**」。
 - 在 Python 解釋器內運行 `import this` 可以獲得完整的列表

```
>>> import this
```

The Zen of Python by Tim Peters

Beautiful is better than ugly.

Explicit is better than implicit.

Simple is better than complex.

Complex is better than complicated.

Flat is better than nested.

Sparse is better than dense.

Readability counts.

Special cases aren't special enough to break the rules.

Although practicality beats purity.

Errors should never pass silently.

Unless explicitly silenced.

In the face of ambiguity, refuse the temptation to guess.

There should be one-- and preferably only one --obvious way to do it.

Although that way may not be obvious at first unless you're Dutch.

Now is better than never.

Although never is often better than *right* now.

If the implementation is hard to explain, it's a bad idea.

If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great idea -- let's do more of those!

--From: Python.org

◆ Python 開發人員盡量避開不成熟或者不重要的優化。

- 一些針對非重要部位的加快運行速度的補件(Patches)，通常不會被合併到 Python 內。

◆ Python 屬於動態型別語言，在執行期間檢查資料的類型，需要保持描述變數值的實際類型標記。

- 靜態型別語言在聲明變數時，已經指定了資料類型和表示方法，
 - ✓ 根據這一事實，導致 Python 相對於 C、Visual Basic 等靜態型別語言來說，執行速度較慢。
- 根據二八定律，大多數程式對速度要求不高。
- 在某些對運行速度要求很高的情況，Python 設計師傾向於使用 JIT 技術，或者用使用 C/C++語言改寫這部分程式。
 - ✓ 目前可用的 JIT 技術是 PyPy。

(四)Python應用範圍

甲、Web程式

◆ Python 經常被用於 Web 開發。

- 通過 mod_wsgi 模組，Apache 可以運行用 Python 編寫的 Web 程式。
- 使用 Python 語言編寫的 Gunicorn 作為 Web 伺服器，也能夠執行 Python 語言編寫的 Web 程式。
- Python 定義了 WSGI 標準應用介面，來協調 Http 伺服器與基於 Python 的 Web 程式之間的溝通。
- 一些 Web 框架，如 Django、Pyramid、TurboGears、Tornado、web2py、Zope、Flask 等，可以讓程式設計師輕鬆地開發和管理複雜的 Web 程式。

◆ Python 對於各種網路協定的支援很完善，因此經常被用於編寫伺服器軟體、網路爬蟲。

- 第三方函式庫 Twisted 支援非同步線上編寫程式和多數標準的網路協定(包含用戶端和伺服器)，並且提供了多種工具，被廣泛用於編寫高效能的伺服器軟體。
- 另有 gevent 這個流行的第三方函式庫，同樣能夠支援高效能高並行的網路開發。

乙、GUI開發

- ◆ Python 本身包含的 Tkinter 庫能夠支援簡單的 GUI 開發。
 - 越來越多的 Python 程式設計師選擇 wxPython 或者 PyQt 等 GUI 套件來開發跨平台的桌面軟體。
 - 使用它們開發的桌面軟體執行速度快，且與使用者的桌面環境相契合。
 - 通過 PyInstaller 還能將程式釋出為獨立的安裝程式包。

丙、作業系統

◆ 在很多作業系統裡，Python 是標準的系統元件。

- 大多數 Linux 發行版以及 NetBSD、OpenBSD 和 Mac OS X 都整合了 Python，可以在終端機下直接執行 Python。
 - ✓ 一些 Linux 發行版的安裝器使用 Python 語言編寫，比如 Ubuntu 的 Ubiquity 安裝器、Red Hat Linux 和 Fedora 的 Anaconda 安裝器。
 - ✓ Gentoo Linux 使用 Python 來編寫它的 Portage 軟體包管理系統。
- Python 標準庫包含了多個呼叫作業系統功能的函式庫。
- 通過 pywin32 這個第三方軟體包，Python 能夠存取 Windows 的 COM 服務及其它 Windows API。
- 使用 IronPython、pythonnet，Python 程式能夠直接調用 .Net Framework。

丁、AI平台應用

◆ 數量分析

- Numpy、SciPy

◆ 統計計量

- Statmodels、Arch

◆ 資料庫

- Pandas

◆ 繪圖

- Matplotlib

◆ 機器學習

- Scikit-Learn

◆ 類神經網路

- 前端：Keras(Keras-Team)
- TensorFlow(Google)、Caffe(BVLC)、CNTK(Microsoft)、
- MXNet(Amazon)、PyTorch(Facebook)、Torch7(Facebook)、
- Theano(Montreal Univ.)、Caffe2(Facebook) 、H2O(MIT)。

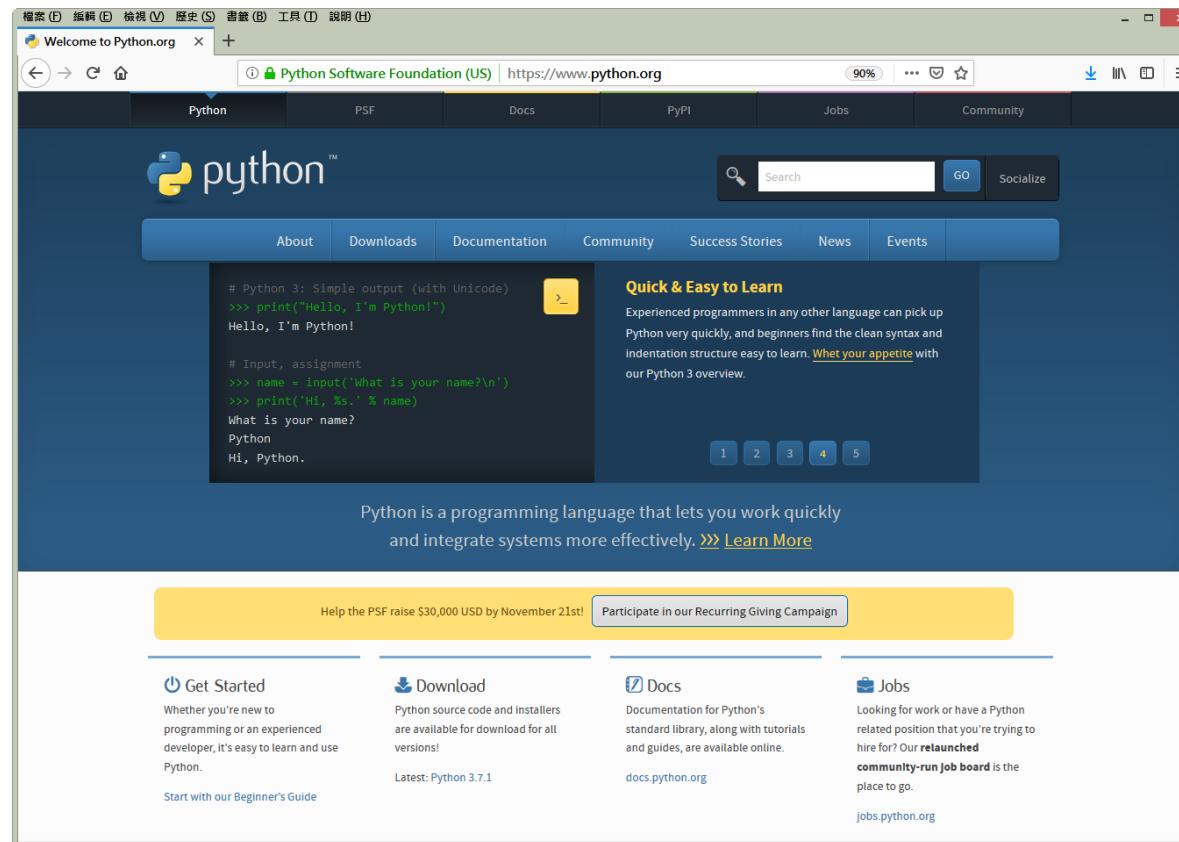
戊、其他

- ◆ NumPy、SciPy、Matplotlib 可以讓 Python 程式設計師編寫科學計算程式。
 - 有些公司會使用 Scons 代替 make 構建 C++程式。
- ◆ 很多遊戲使用 C++編寫圖形顯示等高效能模組，而使用 Python 或者 Lua 編寫遊戲的邏輯、伺服器。
 - 相較於 Python，Lua 的功能更簡單、體積更小；Python 則支援更多的特性和資料類型。
- ◆ YouTube、Google、Yahoo!、NASA 都在內部大量地使用 Python。

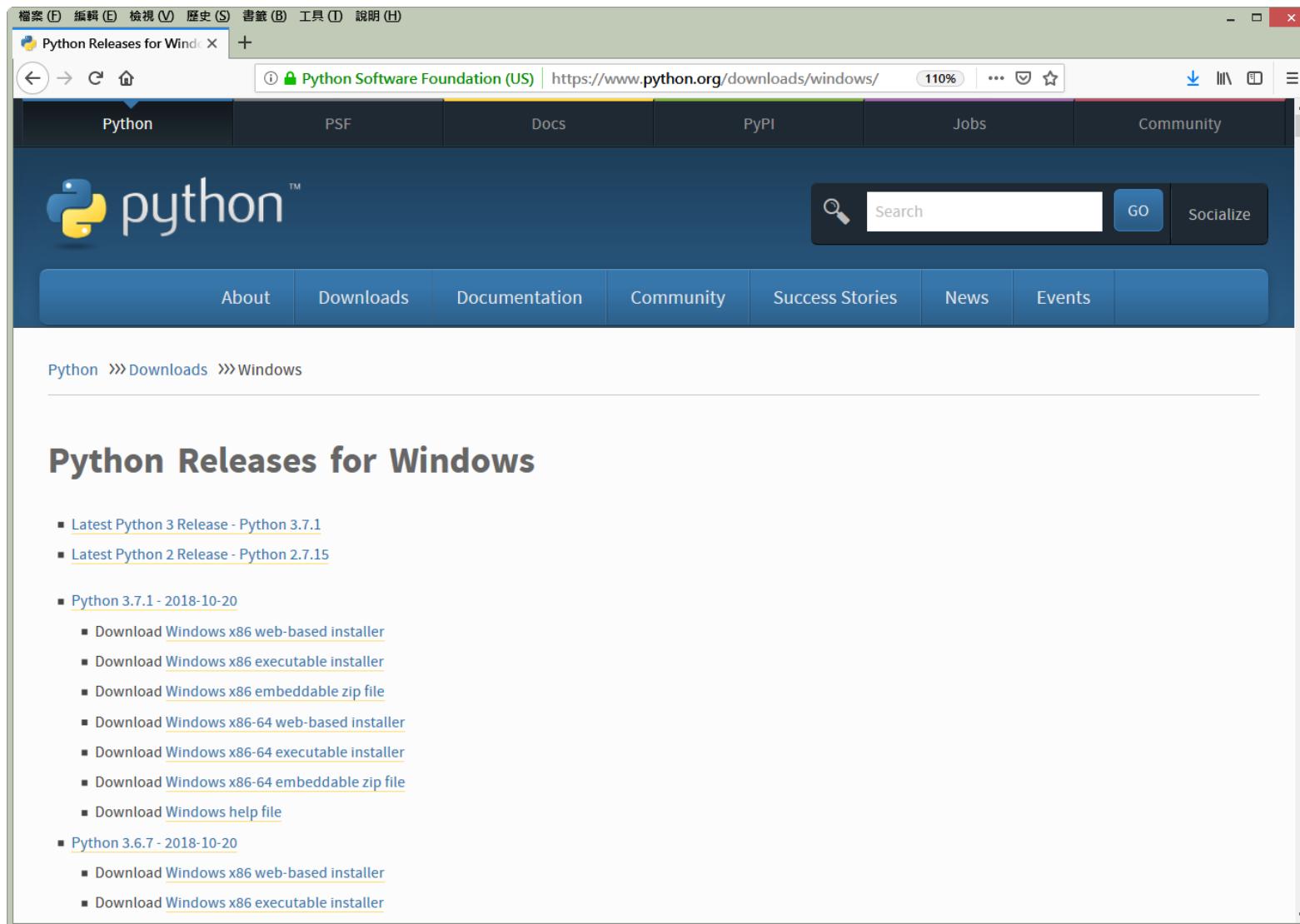
二、Python 安裝

(一)Python官網

◆ 到 Python 的官方網站，<https://www.python.org/>，下載要安裝的 Python 程式。



◆ 選擇《Downloads/Windows》

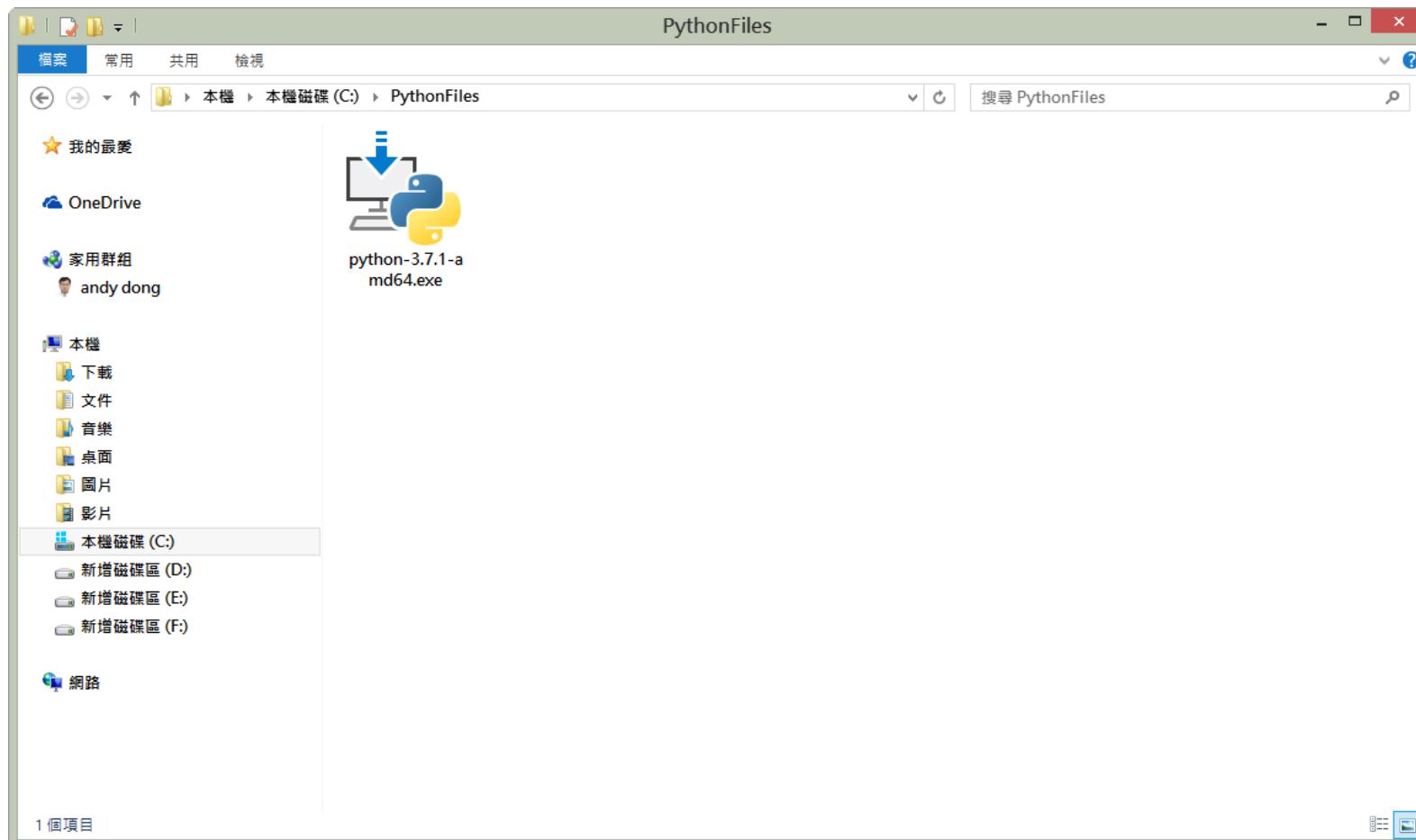


◆ Python 的版本選擇

- Python 3 是 Python 2 的進化版本，其目的在於語法結構的調整與效能優化，但他並不完全向下相容於 Python 2。
- 以作者的經驗是，現有的 Python 套件絕大多數都支援 Python 2，然而 Python 3 已經是未來的主流，因此應考慮 Python 3。
- 選擇 ■ [Download Windows x86-64 executable installer](#) 。

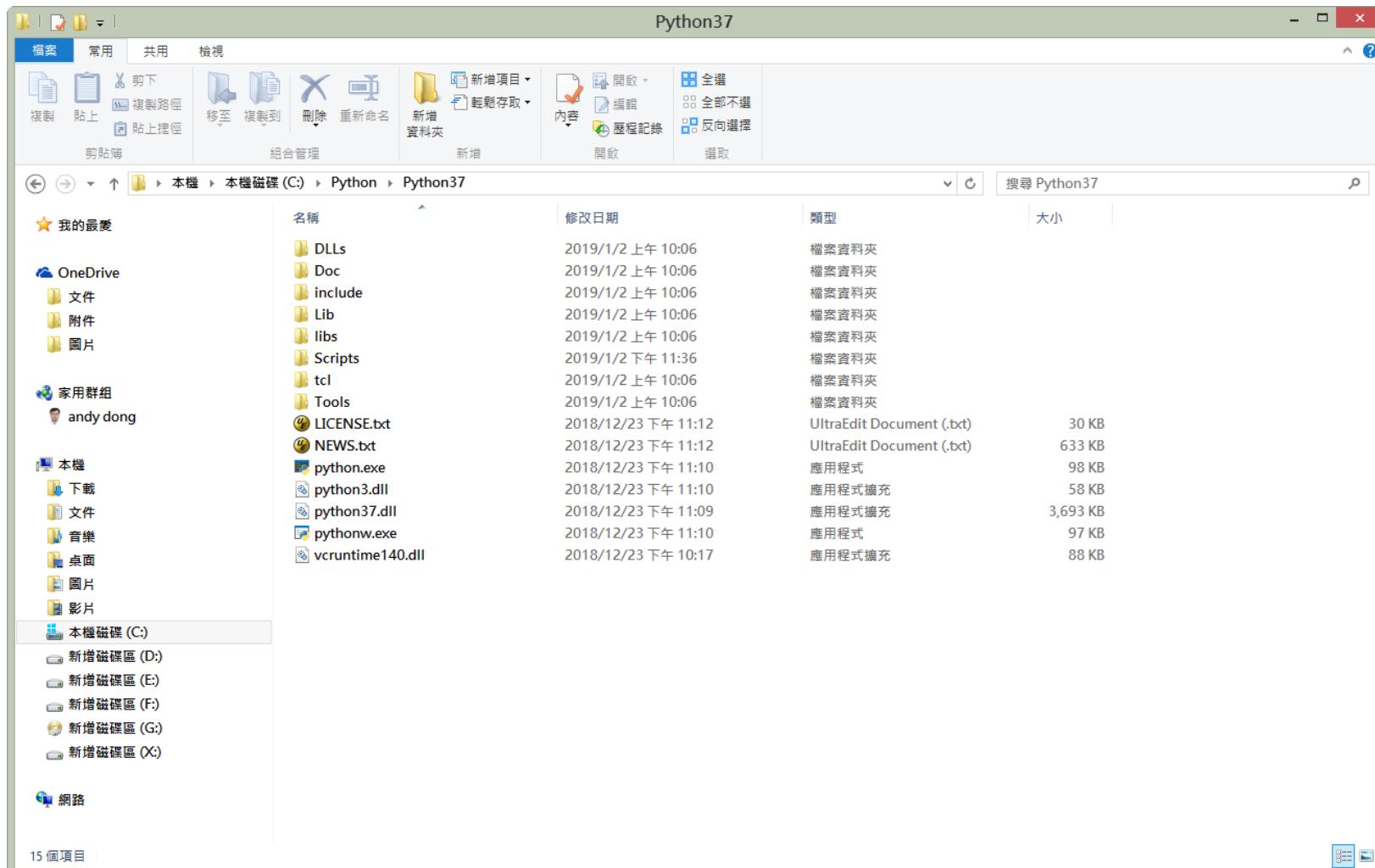


◆ 將下載檔案放置於 C:\PythonFiles 目錄下，如下圖。



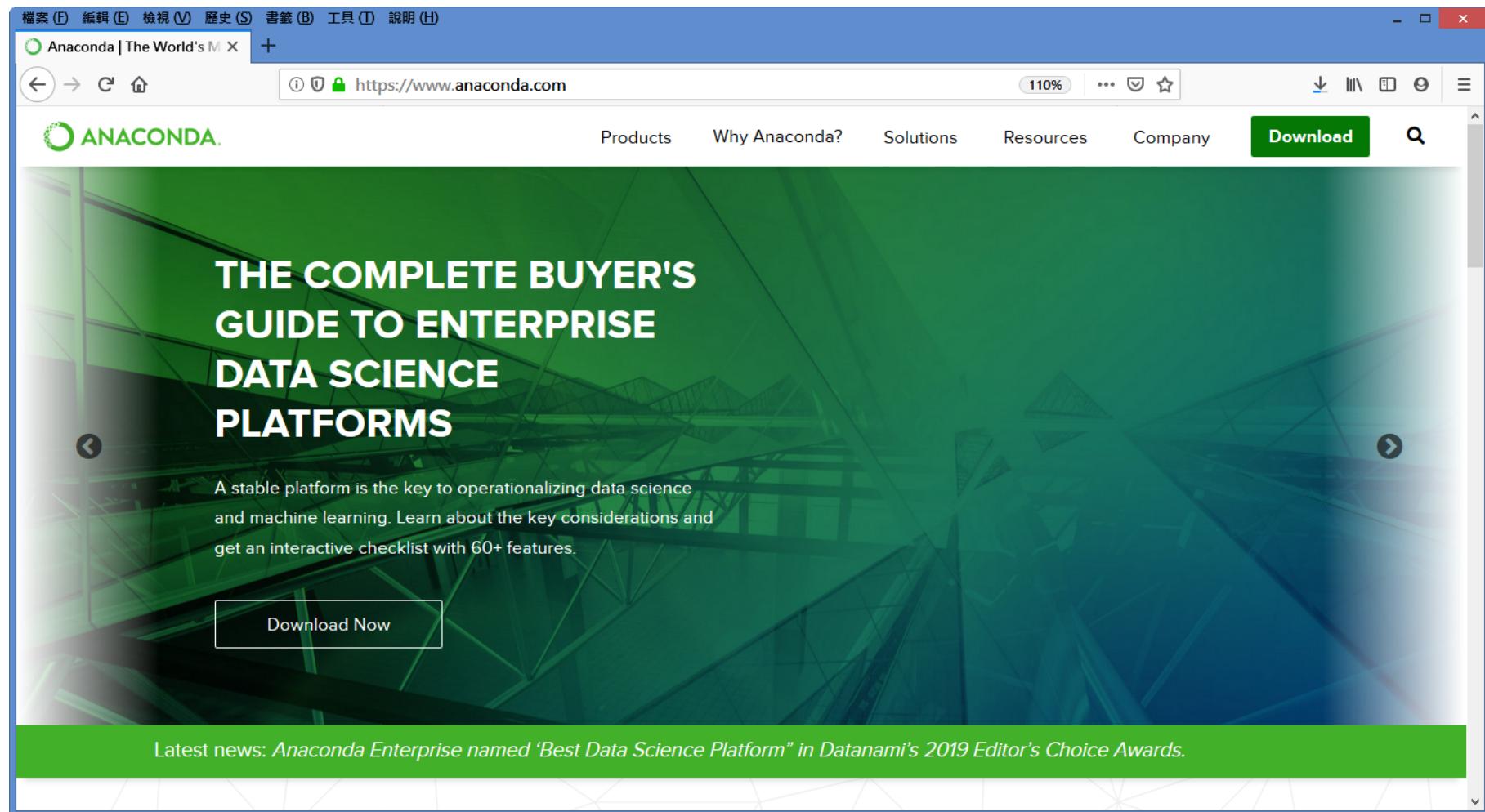
◆ 點選執行之，以預設安裝即可，安裝的位置在 C:\Python37 下。

➤ 安裝完成後的內容，類似下圖。

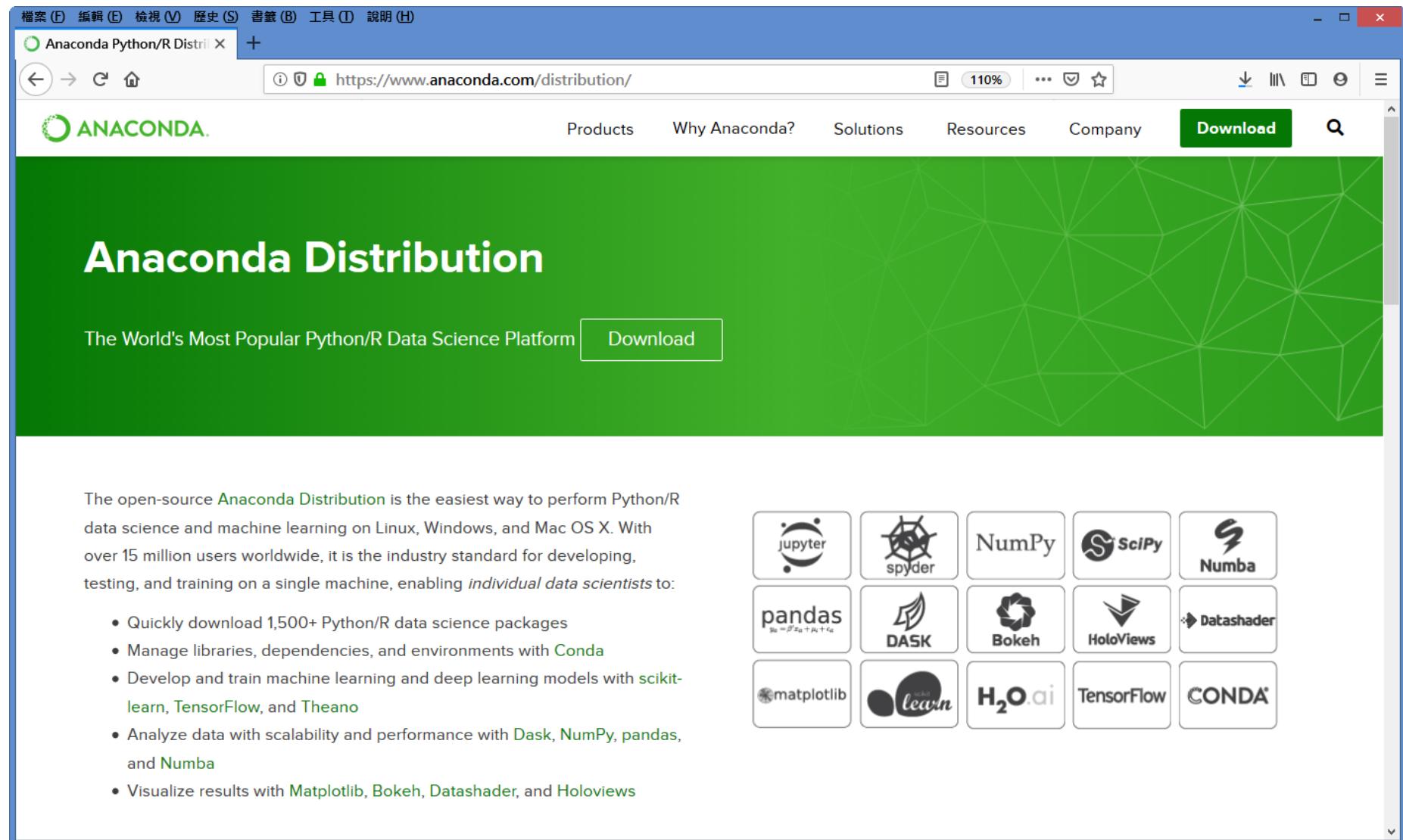


(二)安裝Anaconda

- ◆ Anaconda 官網，Anaconda 是一個發行套件，已選配常用套件與一些便利管理的工具。



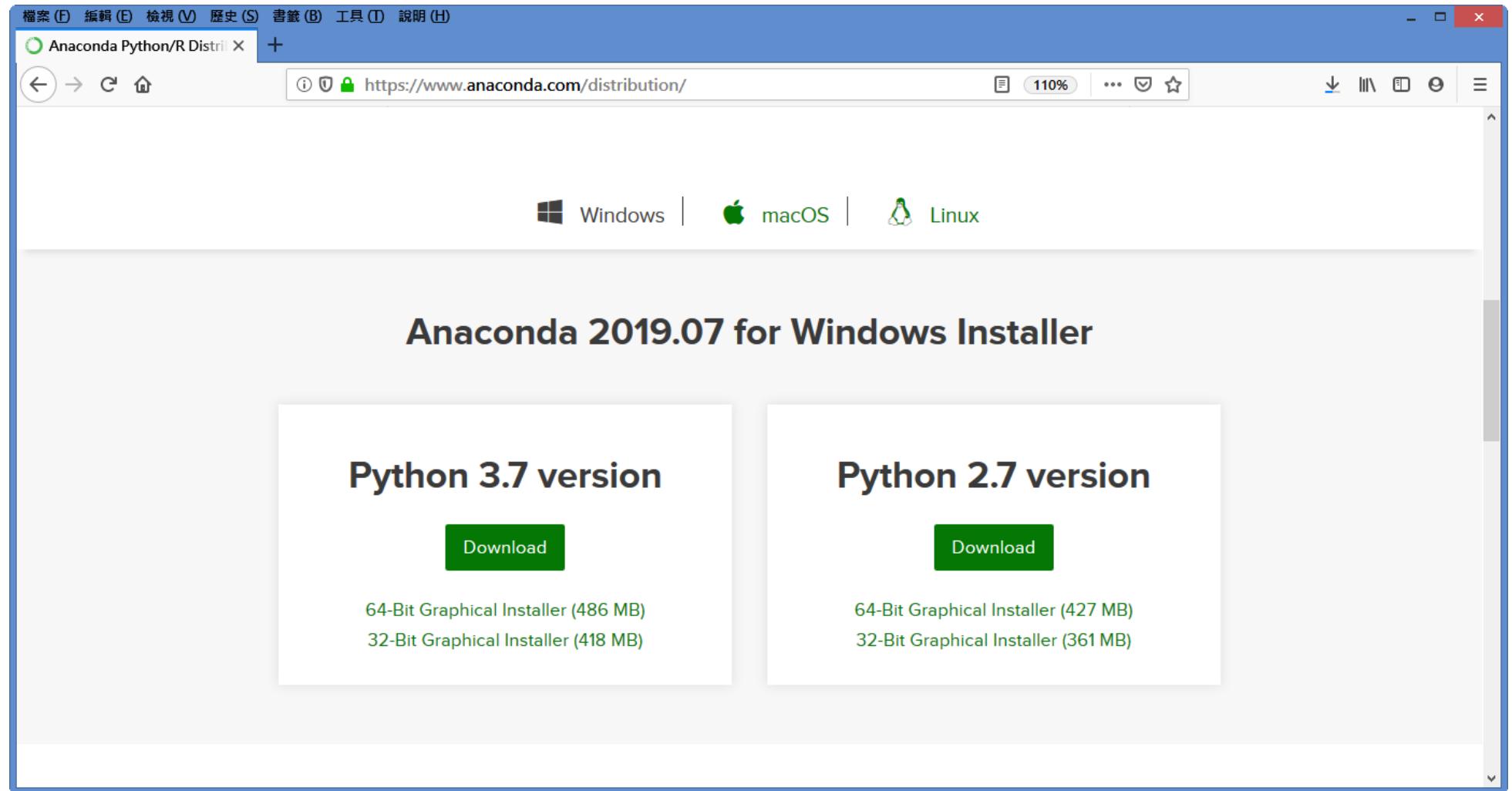
◆ 下載 Python3.7，64bits。



The screenshot shows a Microsoft Edge browser window displaying the Anaconda Distribution website at <https://www.anaconda.com/distribution/>. The page has a blue header with tabs for '檔案 (F)', '編輯 (E)', '檢視 (V)', '歷史 (S)', '書籤 (B)', '工具 (I)', and '說明 (H)'. A green tab for 'Anaconda Python/R Distri' is selected. The main content area features a large green banner with the text 'Anaconda Distribution' and 'The World's Most Popular Python/R Data Science Platform'. Below the banner is a 'Download' button. To the right of the banner is a geometric background graphic. The main text on the page describes the Anaconda Distribution as the easiest way to perform Python/R data science and machine learning on various operating systems, with over 15 million users worldwide. It lists several key features and supported libraries. On the right side, there is a grid of logos for various Python data science libraries, including Jupyter, Spyder, NumPy, SciPy, Numba, pandas, DASK, Bokeh, HoloViews, Datashader, matplotlib, scikit-learn, H2O.ai, TensorFlow, and CONDA.

The open-source [Anaconda Distribution](#) is the easiest way to perform Python/R data science and machine learning on Linux, Windows, and Mac OS X. With over 15 million users worldwide, it is the industry standard for developing, testing, and training on a single machine, enabling *individual data scientists* to:

- Quickly download 1,500+ Python/R data science packages
- Manage libraries, dependencies, and environments with [Conda](#)
- Develop and train machine learning and deep learning models with [scikit-learn](#), [TensorFlow](#), and [Theano](#)
- Analyze data with scalability and performance with [Dask](#), [NumPy](#), [pandas](#), and [Numba](#)
- Visualize results with [Matplotlib](#), [Bokeh](#), [Datashader](#), and [Holoviews](#)



◆ Old version, archive

The screenshot shows a web browser window with the URL <https://docs.anaconda.com/anaconda/>. The left sidebar has a dark grey header "Anaconda Distribution" and a list of links:

- Home
- Anaconda Enterprise 5
- Anaconda Enterprise 4
- Anaconda Distribution** (highlighted)
- Installation
- User guide
- Reference
- End User License Agreement
- Anaconda Cloud
- Archive

The main content area has two columns. The left column contains bullet points about pip and conda packages. The right column contains a section titled "Previous versions" with text about previous versions being available in the [archive](#), which is circled in red.

Previous versions of Anaconda are available in the [archive](#). For a list of packages included in each previous version, see [Old package lists](#).

Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it does not matter which one you download, because you can create new environments that include any version of Python packaged with conda. See [Managing Python with conda](#).

« Using Anaconda Distribution with AE4 Installation »

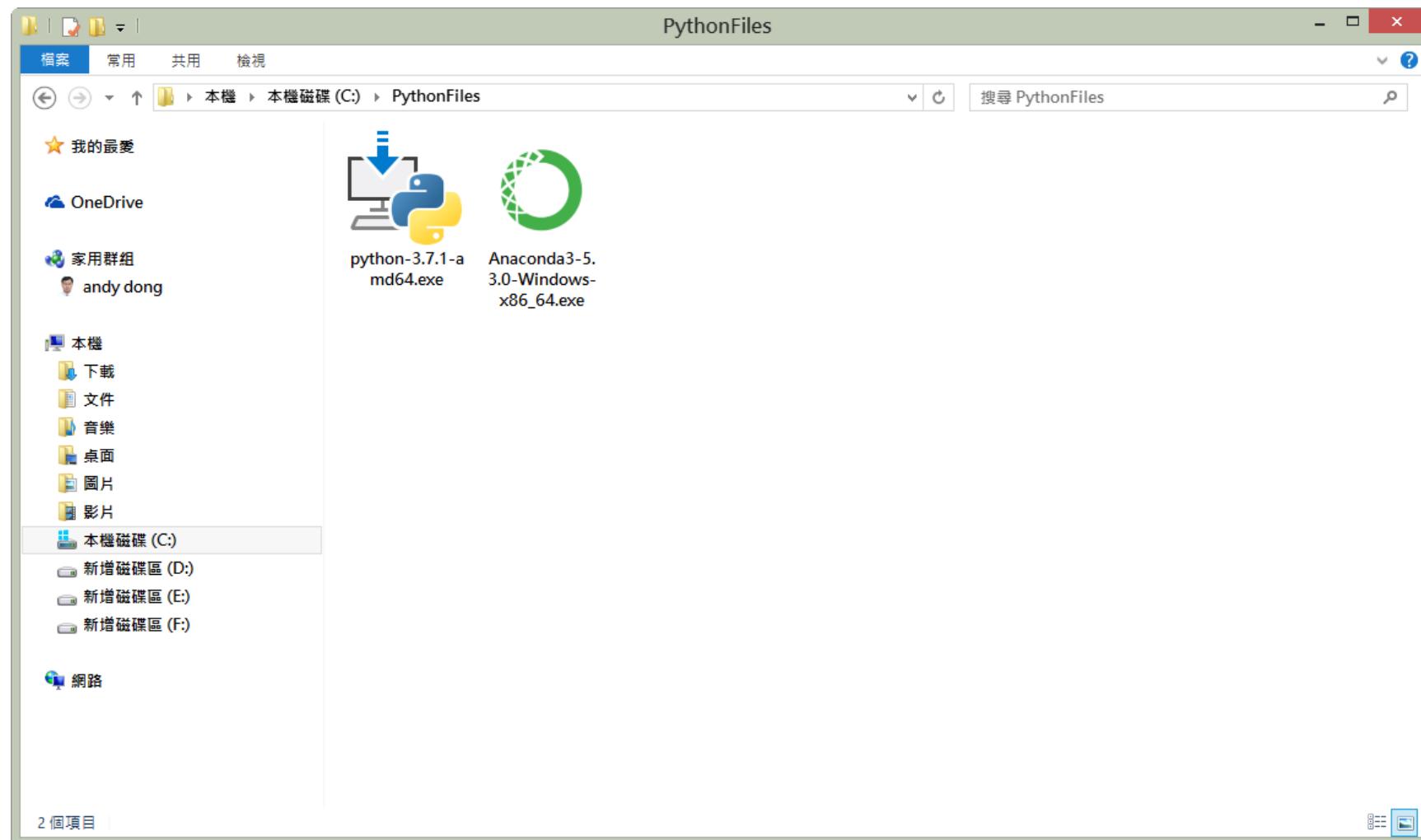
Docs Home
Anaconda Home
More Help & Support
v: latest ▾

© 2019 Anaconda, Inc. All Rights Reserved. [Privacy Policy](#)

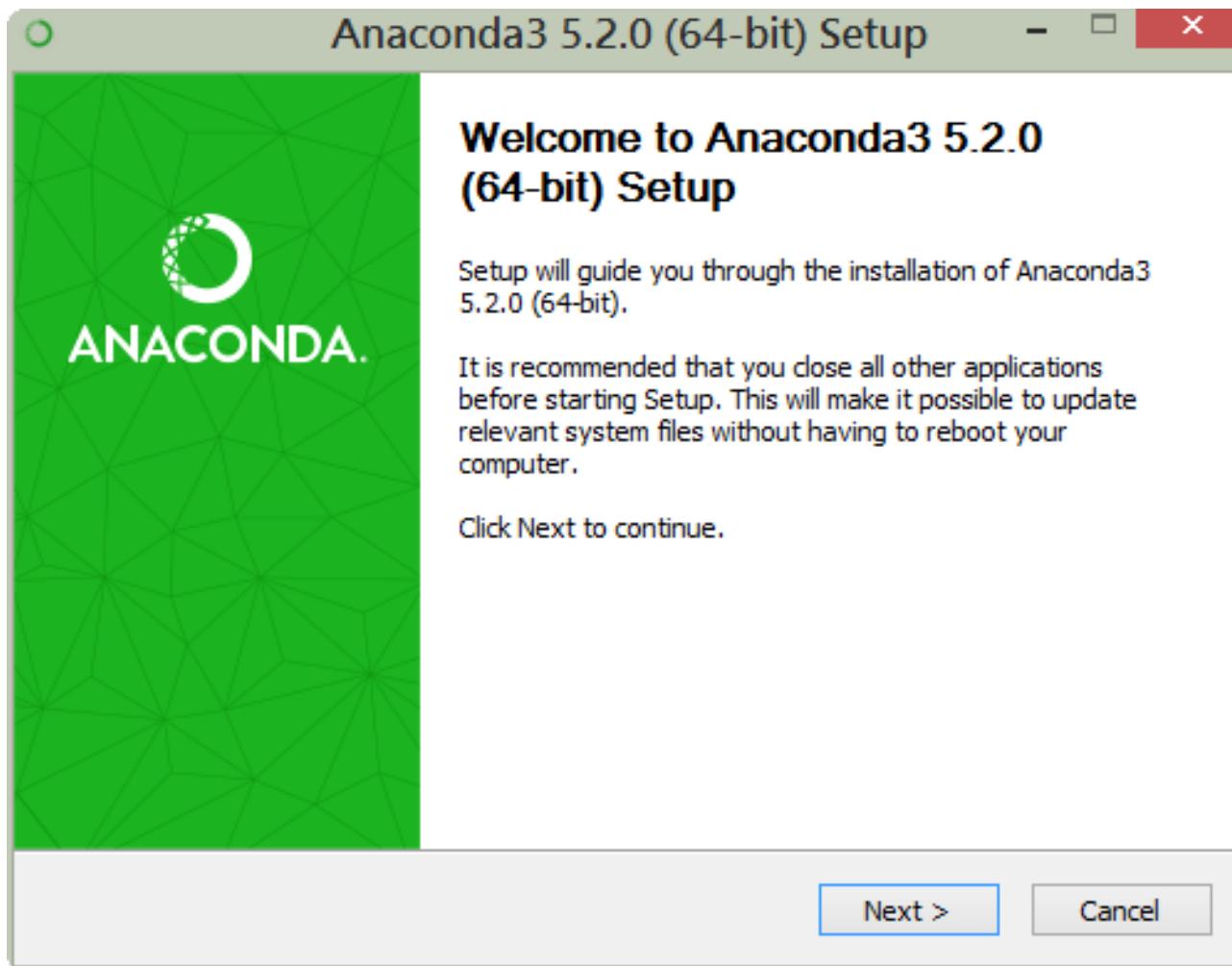
◆ Here

Anaconda installer archive					
https://repo.anaconda.com/archive/					
Anaconda2-5.3.1-Windows-x86_64.exe	580.1M	2018-11-19 13:37:47	ff29ffcd1f767cde91bab71110c00294		
Anaconda3-5.3.1-Linux-x86.sh	527.3M	2018-11-19 13:38:49	6878b6393add83e5fe77d7ala27ee789		
Anaconda3-5.3.1-Linux-x86_64.sh	637.0M	2018-11-19 13:38:46	334b43d5e8468507f123dbfe7437078f		
Anaconda3-5.3.1-MacOSX-x86_64.pkg	634.0M	2018-11-19 13:38:54	6a5cbe559a5b83e2508b39a3b72e90c8		
Anaconda3-5.3.1-MacOSX-x86_64.sh	543.7M	2018-11-19 13:38:57	3c9d849a305653f67edfefdbacdce4d		
Anaconda3-5.3.1-Windows-x86.exe	509.5M	2018-11-19 13:39:54	52d9041d33c0134dd3824e6c15b458c4		
Anaconda3-5.3.1-Windows-x86_64.exe	632.5M	2018-11-19 13:38:59	3e4d013223d8a71d0fa4d58fe5b31023		
Anaconda2-5.3.0-Linux-ppc64le.sh	285.7M	2018-09-27 16:00:22	20a0fad5ef7c3f3df10d350b8ec41bd2		
Anaconda2-5.3.0-Linux-x86.sh	507.5M	2018-09-27 16:00:27	a476ae6c3fe66711ec9e99f1d46f68e0		
Anaconda2-5.3.0-Linux-x86_64.sh	617.6M	2018-09-27 16:00:25	ae1da610739f953ea12e3c7d24bdef63		
Anaconda2-5.3.0-MacOSX-x86_64.pkg	628.3M	2018-09-27 15:59:12	8e02050e148d48a31b99994d906900fb		
Anaconda2-5.3.0-MacOSX-x86_64.sh	538.9M	2018-09-27 16:00:31	de3314d20376ff56a7c0a62087962c86		
Anaconda2-5.3.0-Windows-x86.exe	457.2M	2018-09-27 15:59:15	45a5880d1a56aa8e444b43edcc5e6aa2		
Anaconda2-5.3.0-Windows-x86_64.exe	579.0M	2018-09-27 15:59:14	19fb5f9eedf834b4329dcdea9824516		
Anaconda3-5.3.0-Linux-ppc64le.sh	305.1M	2018-09-27 16:01:33	ee13966b6528f0398a8216f394539255		
Anaconda3-5.3.0-Linux-x86.sh	527.2M	2018-09-27 16:01:37	34fe38d086f069656c2f3cbf13b87460		
Anaconda3-5.3.0-Linux-x86_64.sh	636.9M	2018-09-27 16:01:35	4321e9389b648b5a02824d4473cfdb5f		
Anaconda3-5.3.0-MacOSX-x86_64.pkg	633.9M	2018-09-27 15:59:18	d3075bb9e3d560af3908d5f092e1c07		
Anaconda3-5.3.0-MacOSX-x86_64.sh	543.6M	2018-09-27 16:01:41	e03e91c0aec76d4188b7656e1cec1b74		
Anaconda3-5.3.0-Windows-x86.exe	508.7M	2018-09-27 16:00:05	72e4f7bf75eb46c60f496d326631fddd		
Anaconda3-5.3.0-Windows-x86_64.exe	631.4M	2018-09-27 15:59:20	1807a3c595ed2dab9fc7662f2cdf79fd		
Anaconda2-5.2.0-Linux-ppc64le.sh	269.6M	2018-05-30 13:04:31	479633a95906ea6d41056ebe84a4c47b		
Anaconda2-5.2.0-Linux-x86.sh	488.7M	2018-05-30 13:05:30	758e172a824f467ea6b55d3d076c132f		
Anaconda2-5.2.0-Linux-x86_64.sh	603.4M	2018-05-30 13:04:33	5c034a4ab36ec9b6ae01fa13d8a04462		
Anaconda2-5.2.0-MacOSX-x86_64.pkg	616.8M	2018-05-30 13:05:32	2836c839d29be8d9569a715f4c631a3b		
Anaconda2-5.2.0-MacOSX-x86_64.sh	527.1M	2018-05-30 13:05:34	b1f3fcf58955830b65613a4a8d75c3cf		
Anaconda2-5.2.0-Windows-x86.exe	443.4M	2018-05-30 13:04:17	4a3729b14c2d3fccd3a050821679c702		
Anaconda2-5.2.0-Windows-x86_64.exe	564.0M	2018-05-30 13:04:16	595e427e4b625b6eab92623a28dc4e21		
Anaconda3-5.2.0-Linux-ppc64le.sh	288.3M	2018-05-30 13:05:40	cbd1d5435ead2b0b97dba5b3cf45d694		
Anaconda3-5.2.0-Linux-x86.sh	507.3M	2018-05-30 13:05:46	81d5a1648e3aca4843f88ca3769c0830		
Anaconda3-5.2.0-Linux-x86_64.sh	621.6M	2018-05-30 13:05:43	3e58f494ab9fbe12db4460dc152377b5		

◆ 放置檔案



◆ 安裝流程：舊版參考



◆ 安裝選項

Anaconda3 5.2.0 (64-bit) Setup

License Agreement
Please review the license terms before installing Anaconda3 5.2.0 (64-bit).

Press Page Down to see the rest of the agreement.

=====

Anaconda End User License Agreement

=====

Copyright 2015, Anaconda, Inc.

All rights reserved under the 3-clause BSD License:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Anaconda3 5.2.0 (64-bit).

Anaconda, Inc.

< Back I Agree Cancel

Anaconda3 5.2.0 (64-bit) Setup

Select Installation Type
Please select the type of installation you would like to perform for Anaconda3 5.2.0 (64-bit).

Install for:

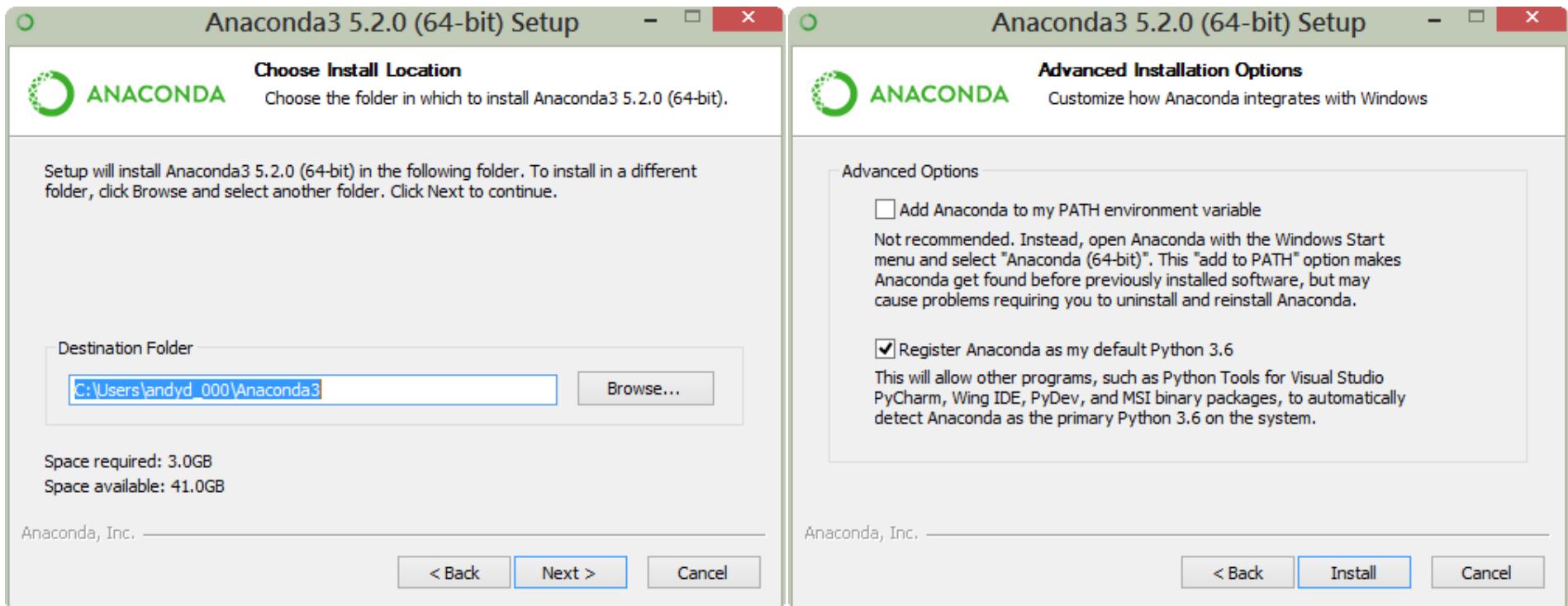
Just Me (recommended)

All Users (requires admin privileges)

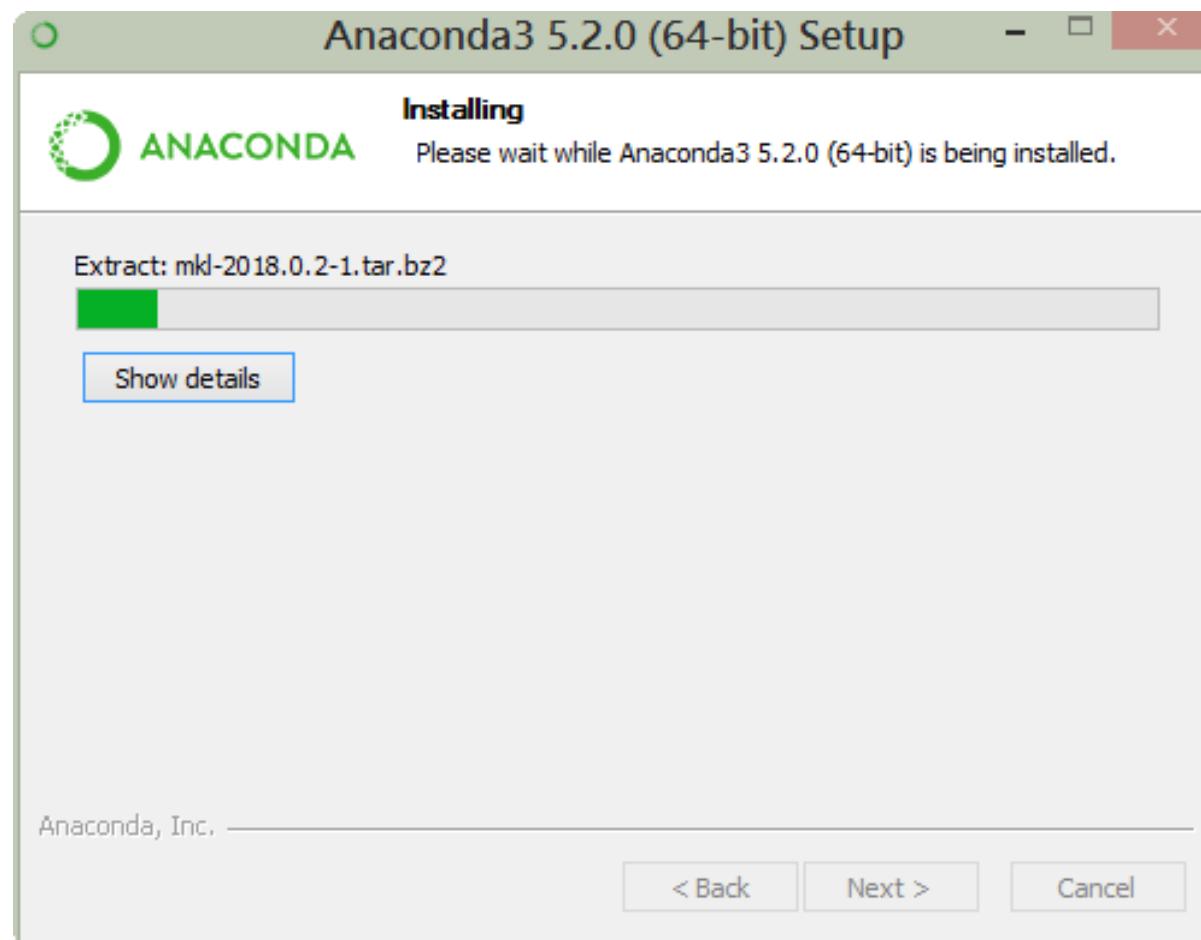
Anaconda, Inc.

< Back Next > Cancel

◆ 安裝選項



◆ 安裝進行



三、Python 的基本使用

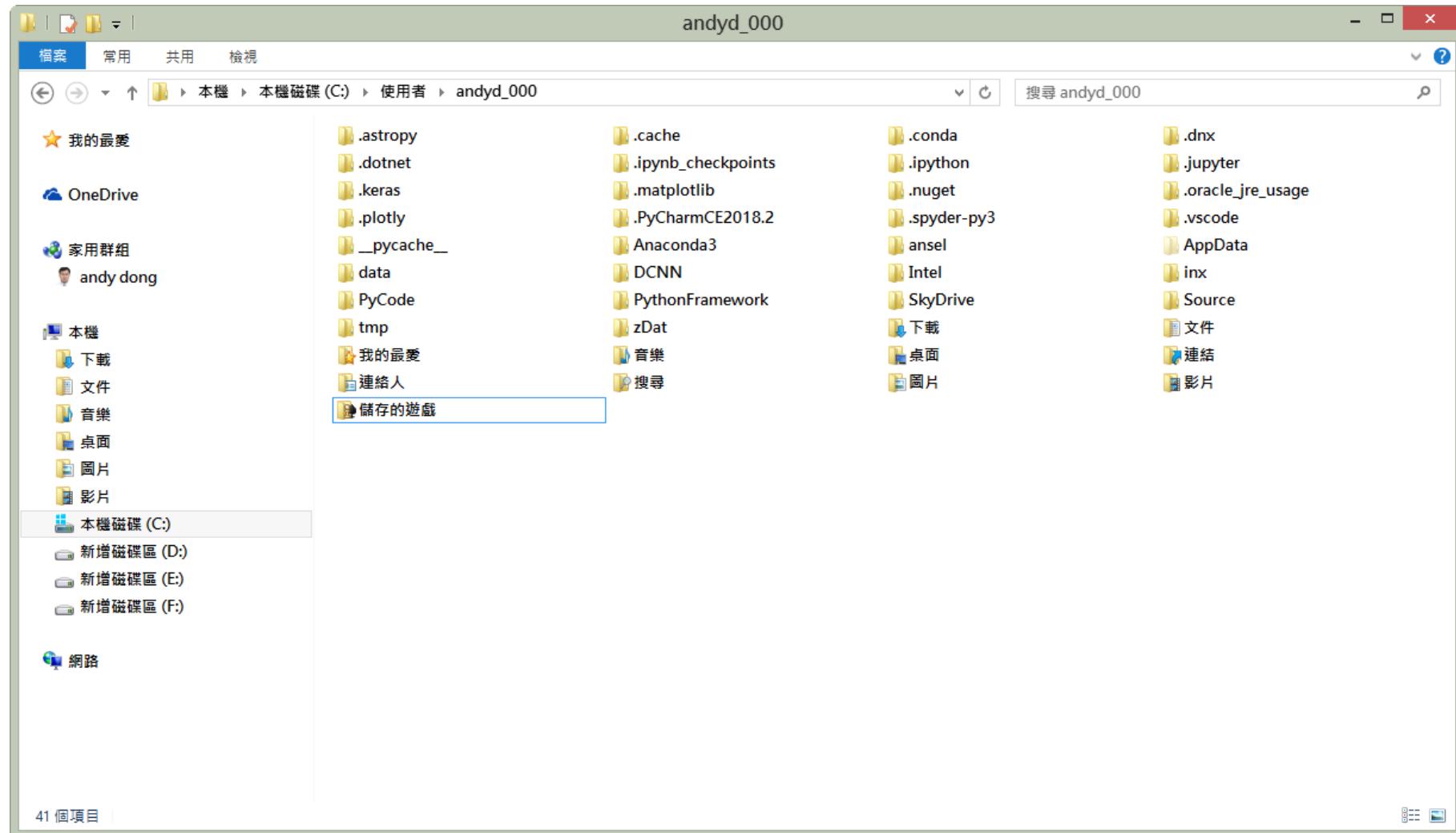
(一) 安裝路徑與內容

- ◆ 下面說明假設讀者已安裝好了 Anaconda3，預設路徑為：

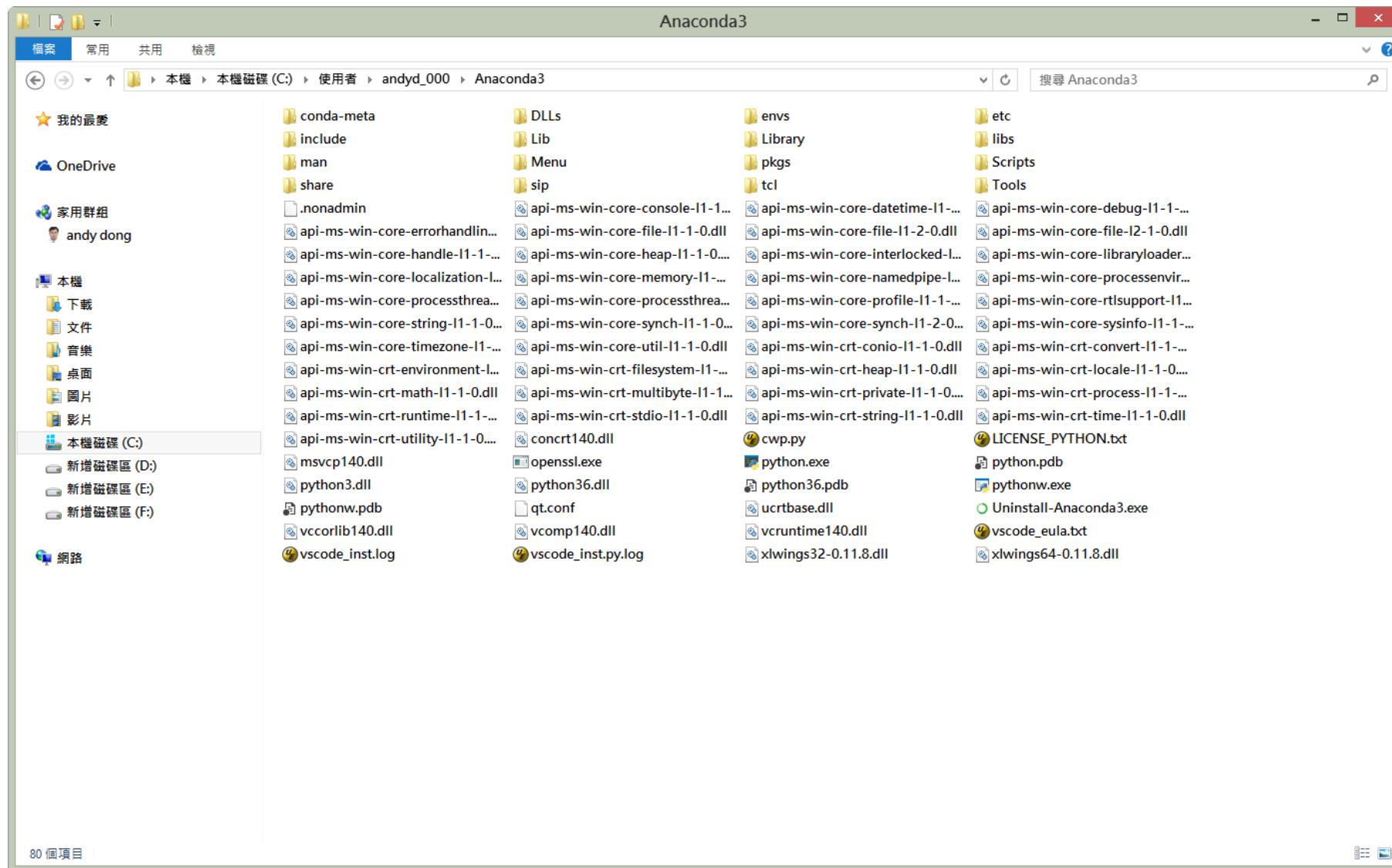
C:\Users\andyd_000\Anaconda3。

➤ 下圖是目錄下預設安裝程式。

◆ 根目錄，



◆ 安裝內容，



◆ 安裝於桌面的 Icon。



(二)主控台Python的使用

- ◆ 首先，打開主控台視窗，然後輸入下面命令。

C:\Users\andy_000>Python

- 便可直接進入 Python 的主控台交談視窗如下。

Python

```
(base) C:\Users\andyd_000>Python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

◆ 提示符號為 , >>> , 在其後輸入下面命令

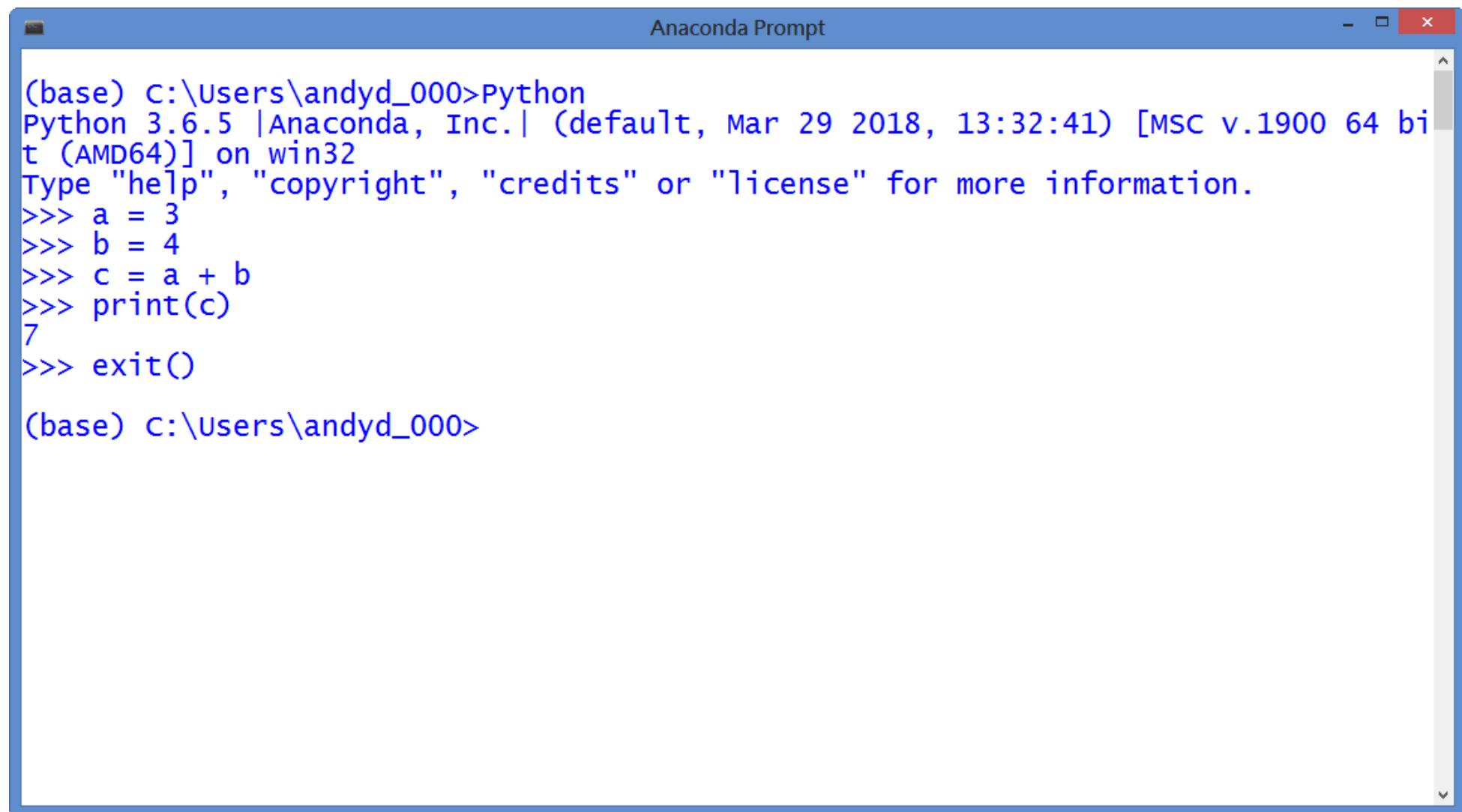
```
>>> a = 3  
>>> b = 4  
>>> c = a + b  
>>> print(c)  
7  
>>>
```

- 令 a 變數的值為 3, b 變數的值為 4, c 為 a 與 b 的和，並列印出 c 的值。
- Python 通過交談視窗，直接計算結果。
 - ✓ 這種直譯式的計算方式，讓我們可以得知每一步驟的結果。
 - ✓ 這對於程式的除錯與開發，非常的方便。

Python

```
(base) C:\Users\andyd_000>Python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 3
>>> b = 4
>>> c = a + b
>>> print(c)
7
>>>
```

◆ 輸入 exit() 命令，按下 Enter 便可結束交談程式，回到主控台的命令列。



The screenshot shows a Windows-style window titled "Anaconda Prompt". Inside, Python version 3.6.5 is running in a command-line interface. The user has entered several commands:

```
(base) c:\Users\andyd_000>Python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> a = 3
>>> b = 4
>>> c = a + b
>>> print(c)
7
>>> exit()

(base) c:\Users\andyd_000>
```

(三)常用Python命令

甲、內建函數

◆ 執行 Python 程式檔，execfile(filename)

```
>>> exec(open("D:\\Python\\AITrade\\Ch01\\test.py").read()) # Python 3.X  
>>> execfile('D:\\Python\\AITrade\\Ch01\\test.py') # Python 2.X
```

➤ 或直接在 Console 下，

```
C:\\Users\\andy_000>Python D:\\Python\\AITrade\\Ch01\\test.py
```

```
Anaconda Prompt - Python

(base) c:\users\andyd_000>Python D:\Python\AITrade\ch01\test.py
7
7
8
9

(base) c:\users\andyd_000>Python
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exec(open("D:\\Python\\AITrade\\ch01\\test.py").read())
7
7
8
9
>>>
```

◆ 取得物件類別

```
>>> A = 10  
>>> type(A)  
<class 'int'>
```

◆ 取得物件屬性

```
>>> dir(A)
```

```
Anaconda Prompt - Python
>>> A = 10
>>> type(A)
<class 'int'>
>>> dir(A)
['__abs__', '__add__', '__and__', '__bool__', '__ceil__', '__class__', '__delattr__',
 '__dir__', '__divmod__', '__doc__', '__eq__', '__float__', '__floor__', '__floordiv__',
 '__format__', '__ge__', '__getattribute__', '__getnewargs__', '__gt__',
 '__hash__', '__index__', '__init__', '__init_subclass__', '__int__', '__invert__',
 '__le__', '__lshift__', '__lt__', '__mod__', '__mul__', '__ne__', '__neg__',
 '__new__', '__or__', '__pos__', '__pow__', '__radd__', '__rand__', '__rdivmod__',
 '__reduce__', '__reduce_ex__', '__repr__', '__rfloordiv__', '__rlshift__',
 '__rmod__', '__rmul__', '__ror__', '__round__', '__rpow__', '__rrshift__',
 '__rshift__', '__rsub__', '__rtruediv__', '__rxor__', '__setattr__', '__sizeof__',
 '__str__', '__sub__', '__subclasshook__', '__truediv__', '__trunc__', '__xor__',
 'bit_length', 'conjugate', 'denominator', 'from_bytes', 'imag', 'numerator',
 'real', 'to_bytes']
>>>
```

乙、套件os

◆ 取得目前工作目錄，os.getcwd()

```
>>> import os  
>>> CurrentDir = os.getcwd()  
>>> Print(CurrentDir) # C:\Users\andy_000
```

◆ 更改工作目錄，os.chdir(path)

```
>>> os.chdir("D:\\\\CUDA")  
>>> CurrentDir = os.getcwd()  
>>> Print(CurrentDir) # D:\\CUDA
```

◆ 取得系統變數，os.getenv(varname[, value])

➤ Return the value of the environment variable *varname* if it exists, or *value* if it doesn't. *value* defaults to **None**.

```
>>> PathStr = os.getenv('path')  
PathStr = "C:\\\\Users\\\\andy_000;C:\\\\Python3\\\\Python37;C:\\\\Python3\\\\Python37\\\\DLLs;"
```

```
Anaconda Prompt - Python

>>> import os
>>> CurrentDir = os.getcwd()
>>> print(CurrentDir)
C:\Users\andyd_000
>>>
>>> os.chdir("D:\\CUDA")
>>> CurrentDir = os.getcwd()
>>> print(CurrentDir)
D:\\CUDA
>>>
>>> Pathstr = os.getenv("path")
>>> print(Pathstr)
C:\Users\andyd_000\Anaconda3;C:\Users\andyd_000\Anaconda3\Library\mingw-w64\bin;
C:\Users\andyd_000\Anaconda3\Library\usr\bin;C:\Users\andyd_000\Anaconda3\Library\bin;C:\Users\andyd_000\Anaconda3\Scripts;C:\Users\andyd_000\Anaconda3\bin;C:\Users\andyd_000\Anaconda3\condabin;C:\Users\andyd_000\Anaconda3\condabin\Library\mingw-w64\bin;C:\Users\andyd_000\Anaconda3\condabin\Library\usr\bin;C:\Users\andyd_000\Anaconda3\condabin\Library\bin;C:\Users\andyd_000\Anaconda3\condabin\Scripts;C:\Users\andyd_000\Anaconda3\condabin\bin;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v5.5\bin;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v5.5\libnvvp;C:\Program Files (x86)\Common Files\Oracle\Java\javapath;C:\ProgramData\oracle\Java\javapath;C:\Program Files\Microsoft HPC Pack 2012\Bin\;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\bin;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v8.0\libnvvp;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.2\bin;C:\Program Files\NVIDIA GPU Computing Toolkit\CUDA\v9.2\libnvvp;C:\
```

丙、套件sys

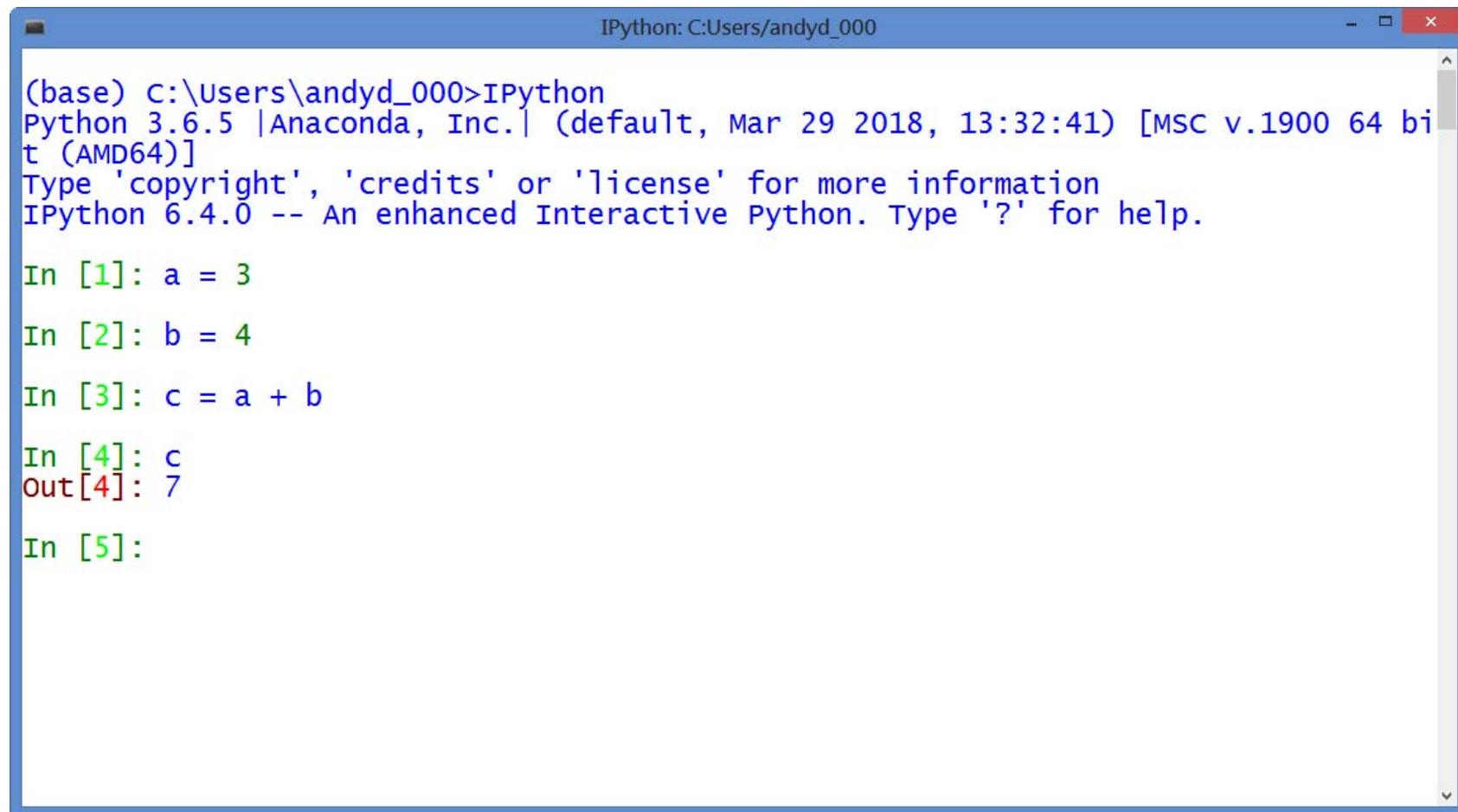
- ◆ 取得命令列參數，sys.argv[1]

```
C:\>Python sma.py 20
```

```
import sys  
N = int(sys.argv[1]) #N=20
```

四、開發工具

(一)交談模式IPython的使用



The screenshot shows a Windows-style terminal window titled "IPython: C:\Users\andyd_000". The window displays the following IPython session:

```
(base) C:\Users\andyd_000>IPython
Python 3.6.5 |Anaconda, Inc.| (default, Mar 29 2018, 13:32:41) [MSC v.1900 64 bit (AMD64)]
Type 'copyright', 'credits' or 'license' for more information
IPython 6.4.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: a = 3
In [2]: b = 4
In [3]: c = a + b
In [4]: c
Out[4]: 7
In [5]:
```

- ◆ 執行程式檔，run filename。

In [1]: run test.py

- ◆ 一般執行。



The screenshot shows an IPython notebook window titled "IPython: C:\Users\andyd_000". The window displays two code cells. The first cell, In [6], contains the command "run D:\Python\AITrade\ch01\test.py" followed by the output "7", "7", "8", and "9", which are the numbers from the file. The second cell, In [7], is currently empty.

```
In [6]: run D:\Python\AITrade\ch01\test.py
7
7
8
9

In [7]:
```

(二) 使用 Python Notebook

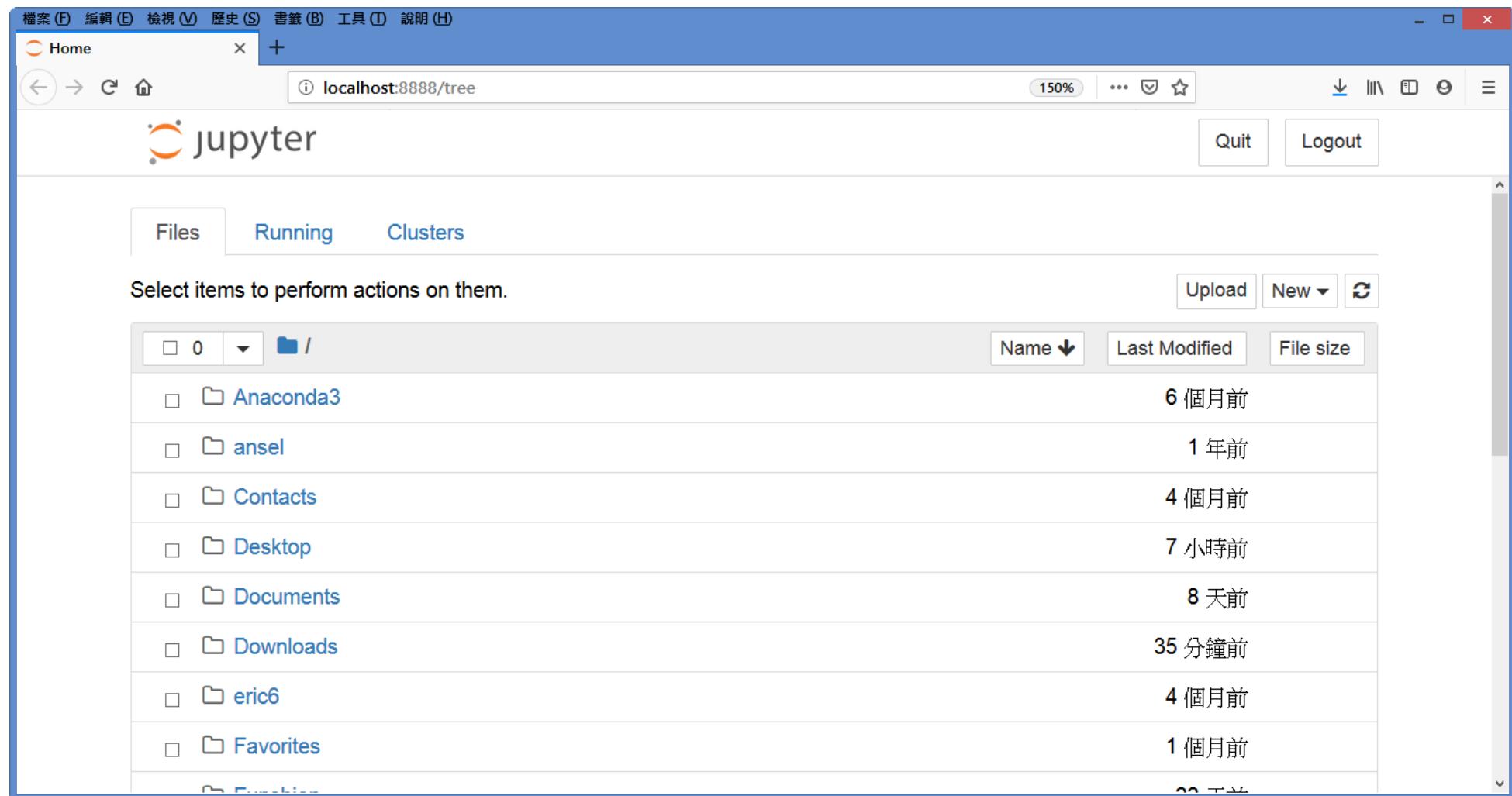
- ◆ 指令，C:\>jupyter notebook。

```
Anaconda Prompt - jupyter notebook

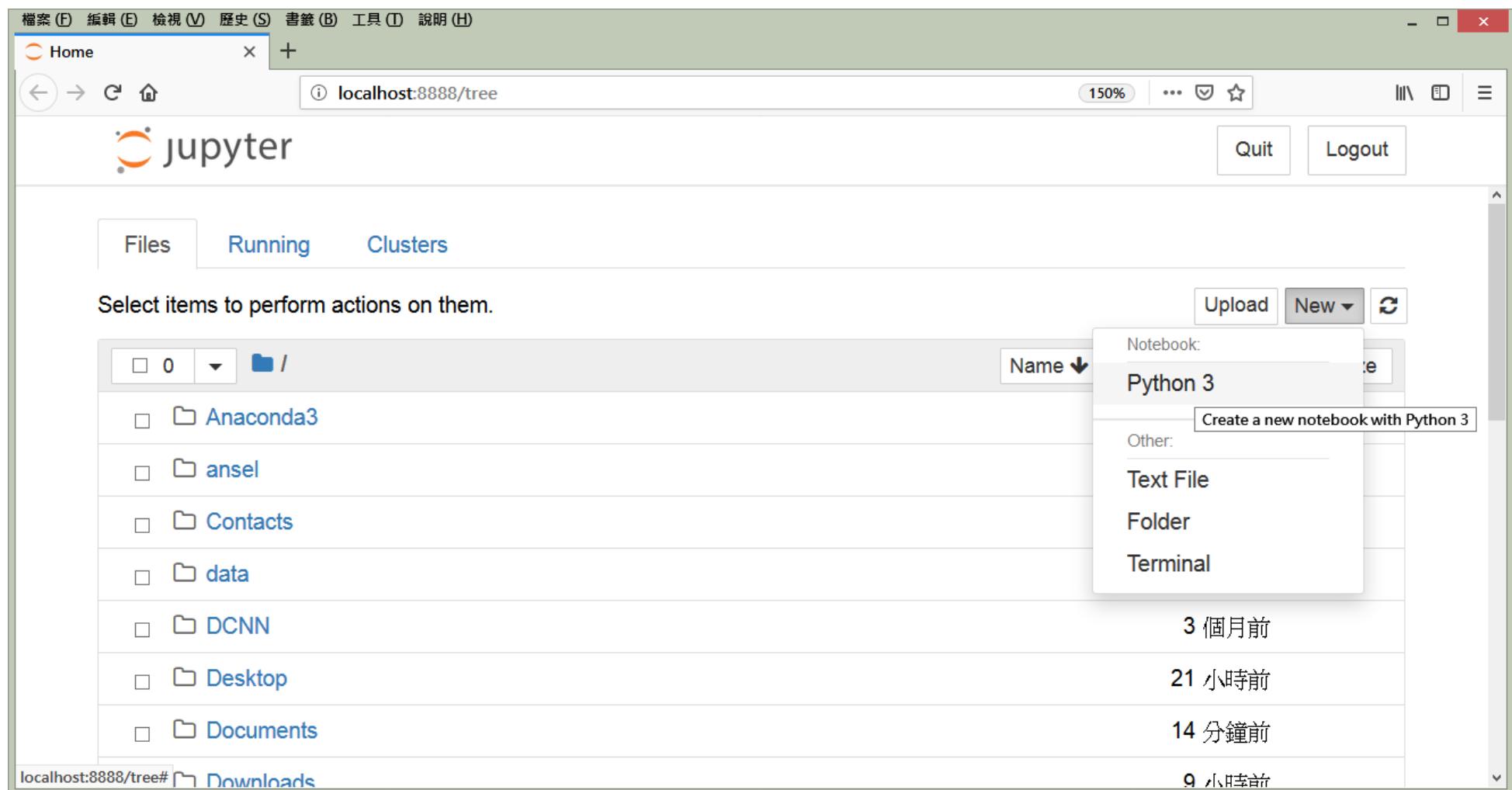
(base) c:\Users\andyd_000>jupyter notebook
[I 11:07:50.408 NotebookApp] JupyterLab beta preview extension loaded from c:\users\andyd_000\Anaconda3\lib\site-packages\jupyterlab
[I 11:07:50.409 NotebookApp] JupyterLab application directory is c:\users\andyd_000\Anaconda3\share\jupyter\lab
[I 11:07:51.227 NotebookApp] Serving notebooks from local directory: c:\users\andyd_000
[I 11:07:51.227 NotebookApp] 0 active kernels
[I 11:07:51.228 NotebookApp] The Jupyter Notebook is running at:
[I 11:07:51.228 NotebookApp] http://localhost:8888/?token=f6e68b6ec09ae218d5860dcce4289148f42f59cf75ce17cea
[I 11:07:51.229 NotebookApp] Use Control-C to stop this server and shut down all
kernels (twice to skip confirmation).
[C 11:07:51.233 NotebookApp]

Copy/paste this URL into your browser when you connect for the first time,
to login with a token:
http://localhost:8888/?token=f6e68b6ec09ae218d5860dcce4289148f42f59cf75ce17cea&token=f6e68b6ec09ae218d5860dcce4289148f42f59cf75ce17cea
[I 11:07:51.709 NotebookApp] Accepting one-time-token-authenticated connection f
rom ::1
```

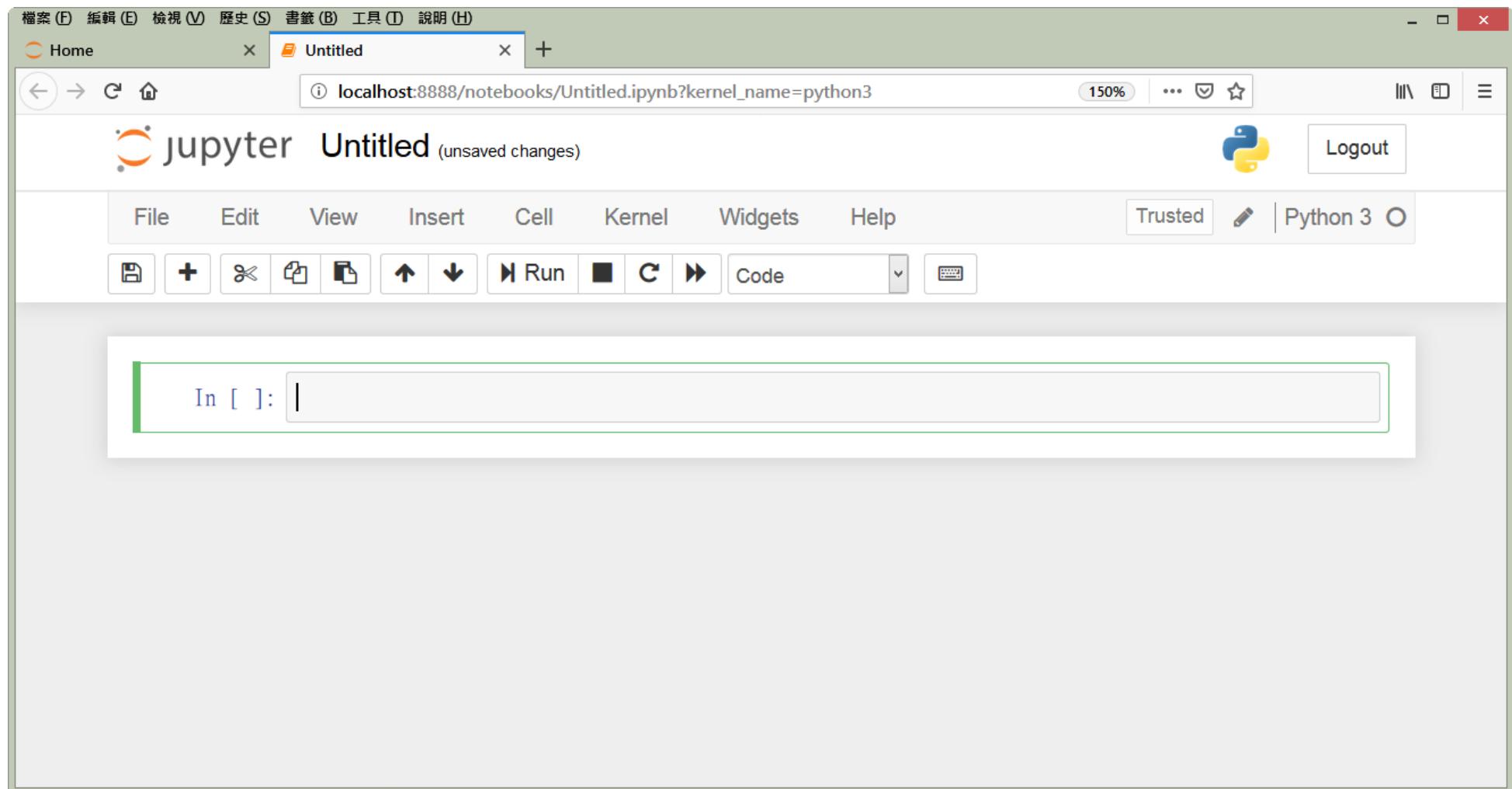
◆ Browser 產生工作頁面。



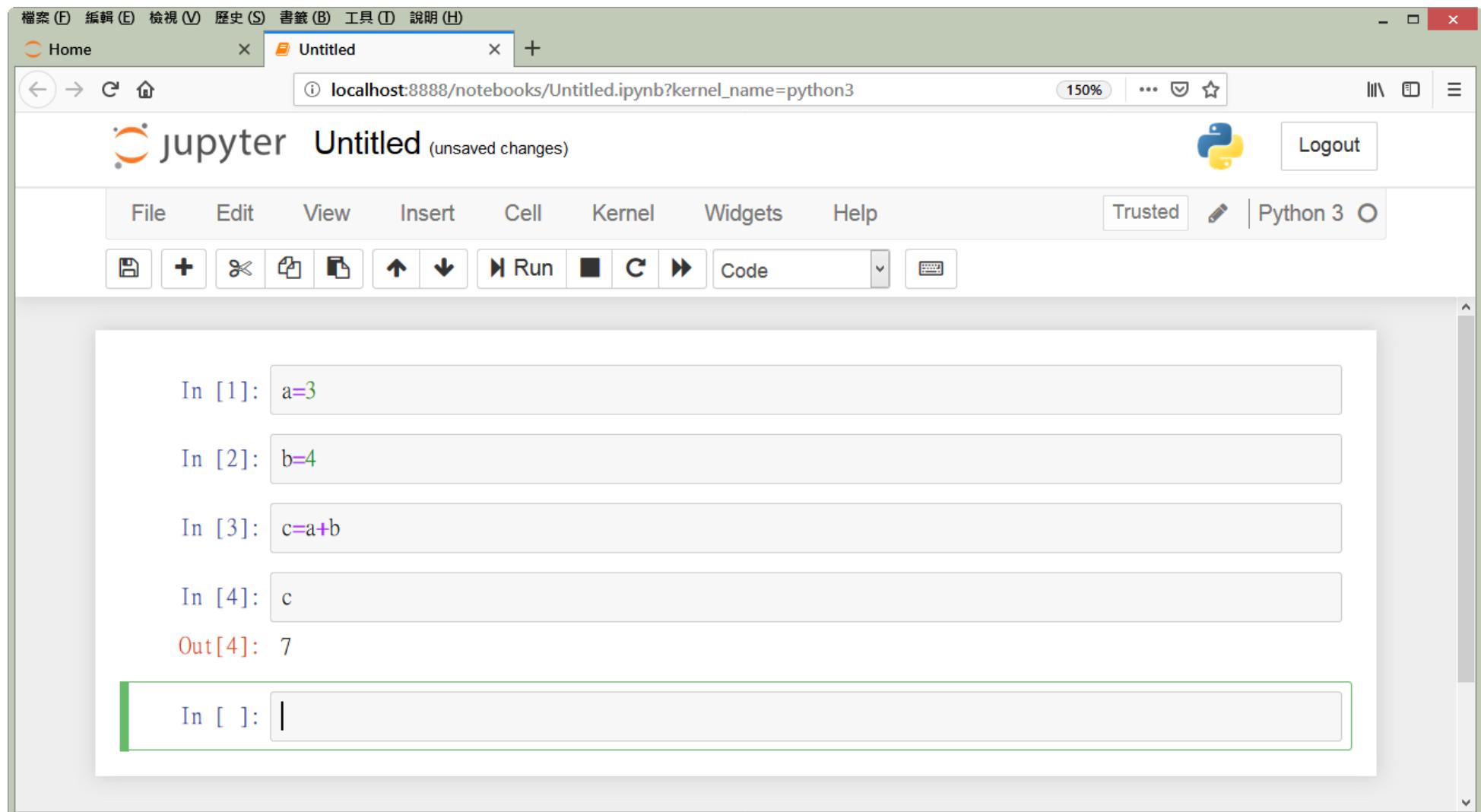
◆ New 一個新的 Notebook。



◆ 打入運算，執行 *Shift + Enter*。繪圖要先下，*%matplotlib inline*。

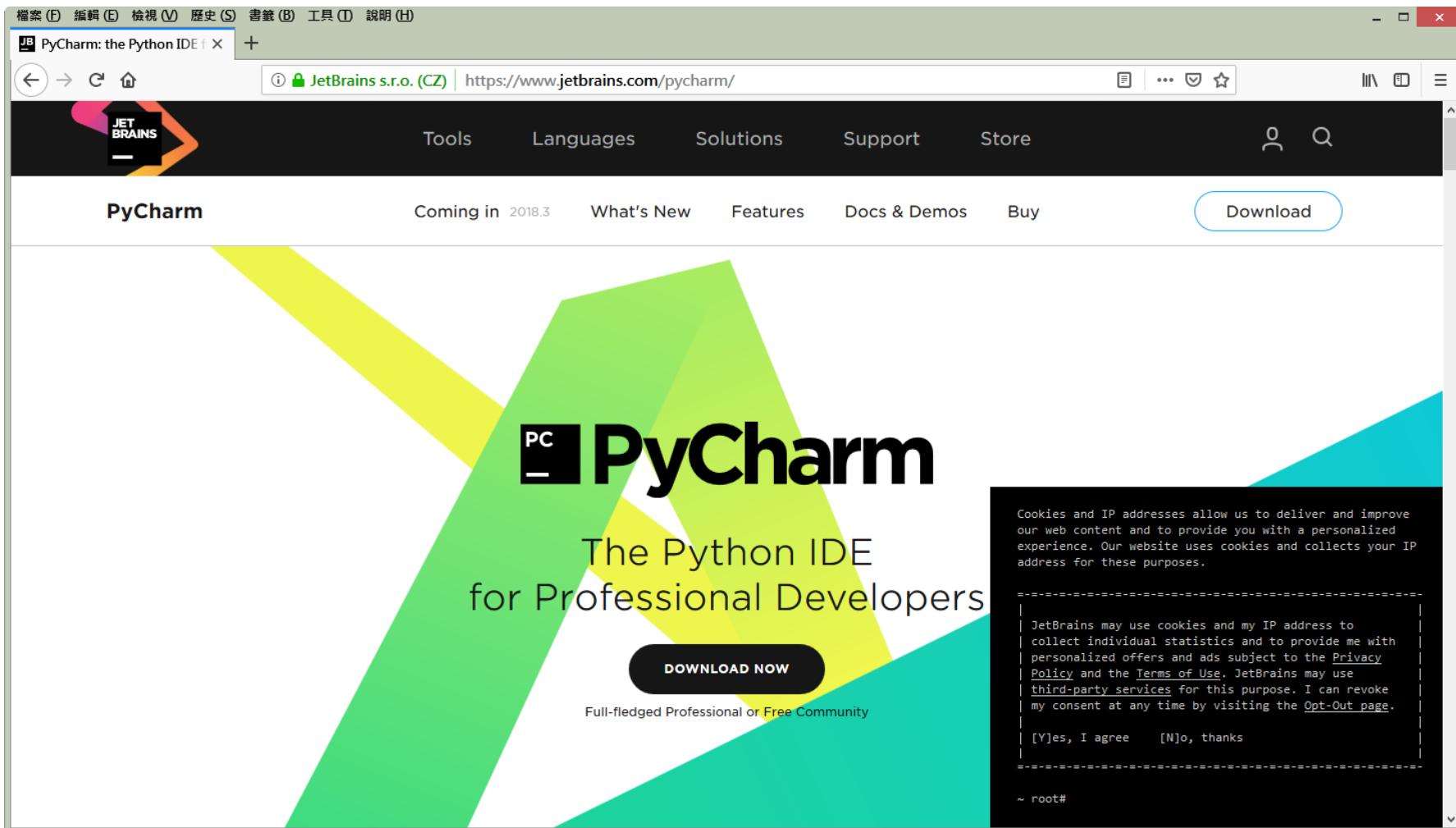


◆ 測試



(三)整合環境Python的使用

◆ PyCharm 整合工具



◆ Community

The screenshot shows the PyCharm download page on the JetBrains website. The page has a dark header with navigation links like Tools, Languages, Solutions, Support, and Store. A search bar and a user profile icon are also present. The main content area features a large 'PyCharm' logo and a 'Coming in 2018.3' message. Below this, there are two main sections: 'Professional' and 'Community'. The 'Professional' section is described as a 'Full-featured IDE for Python & Web development' and includes a 'DOWNLOAD' button and a 'Free trial' link. The 'Community' section is described as a 'Lightweight IDE for Python & Scientific development' and includes a 'DOWNLOAD' button and a 'Free, open-source' link. The URL in the browser address bar is https://www.jetbrains.com/pycharm/download/#section=windows.

JB Download PyCharm: Python X +

JetBrains s.r.o. (CZ) | https://www.jetbrains.com/pycharm/download/#section=windows

Tools Languages Solutions Support Store

PyCharm Coming in 2018.3 What's New Features Docs & Demos Buy Download

Version: 2018.2.4
Build: 182.4505.26
Released: September 20, 2018

System requirements
Installation Instructions
Previous versions

Windows macOS Linux

Download PyCharm

Professional

Full-featured IDE for Python & Web development

DOWNLOAD

Free trial

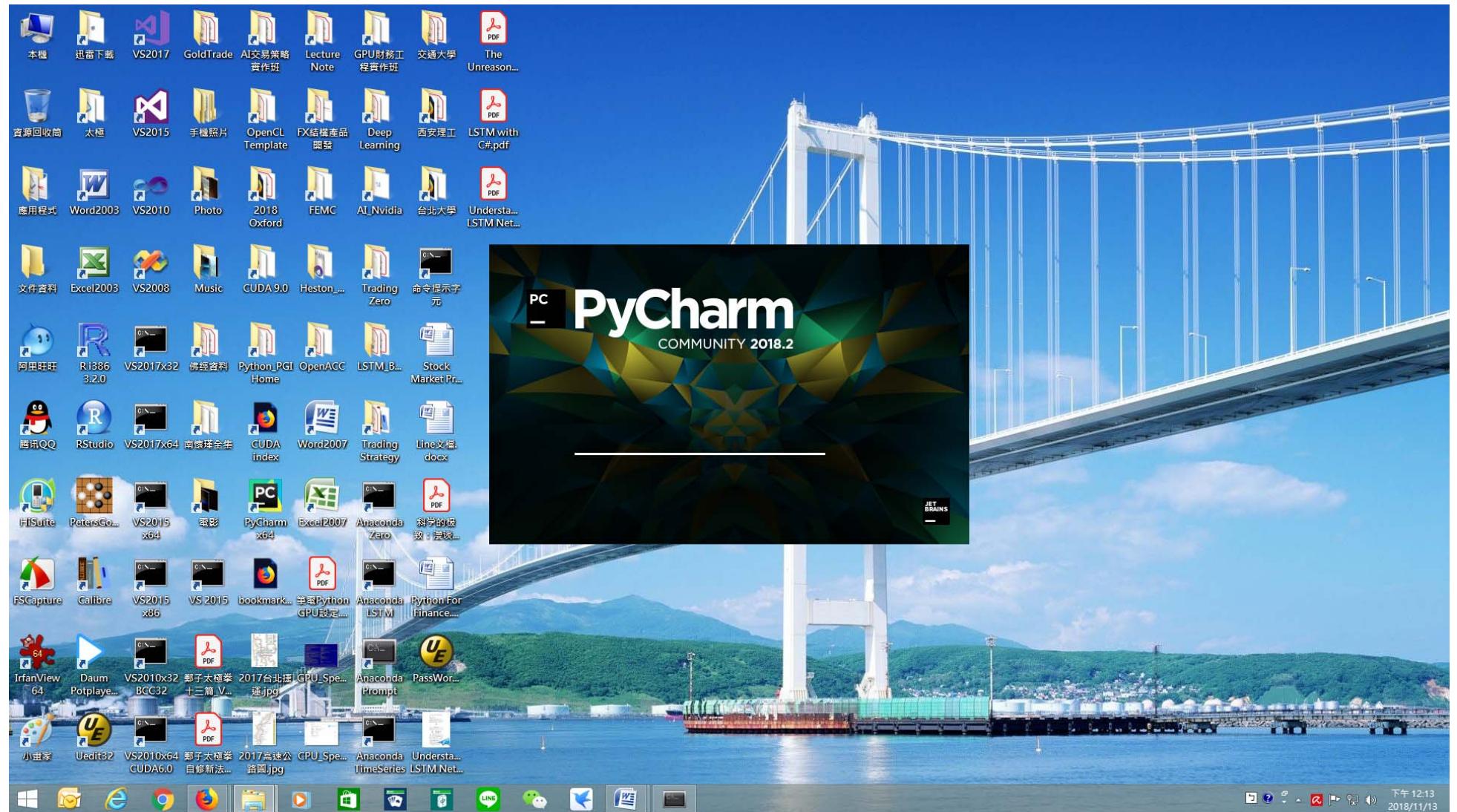
Community

Lightweight IDE for Python & Scientific development

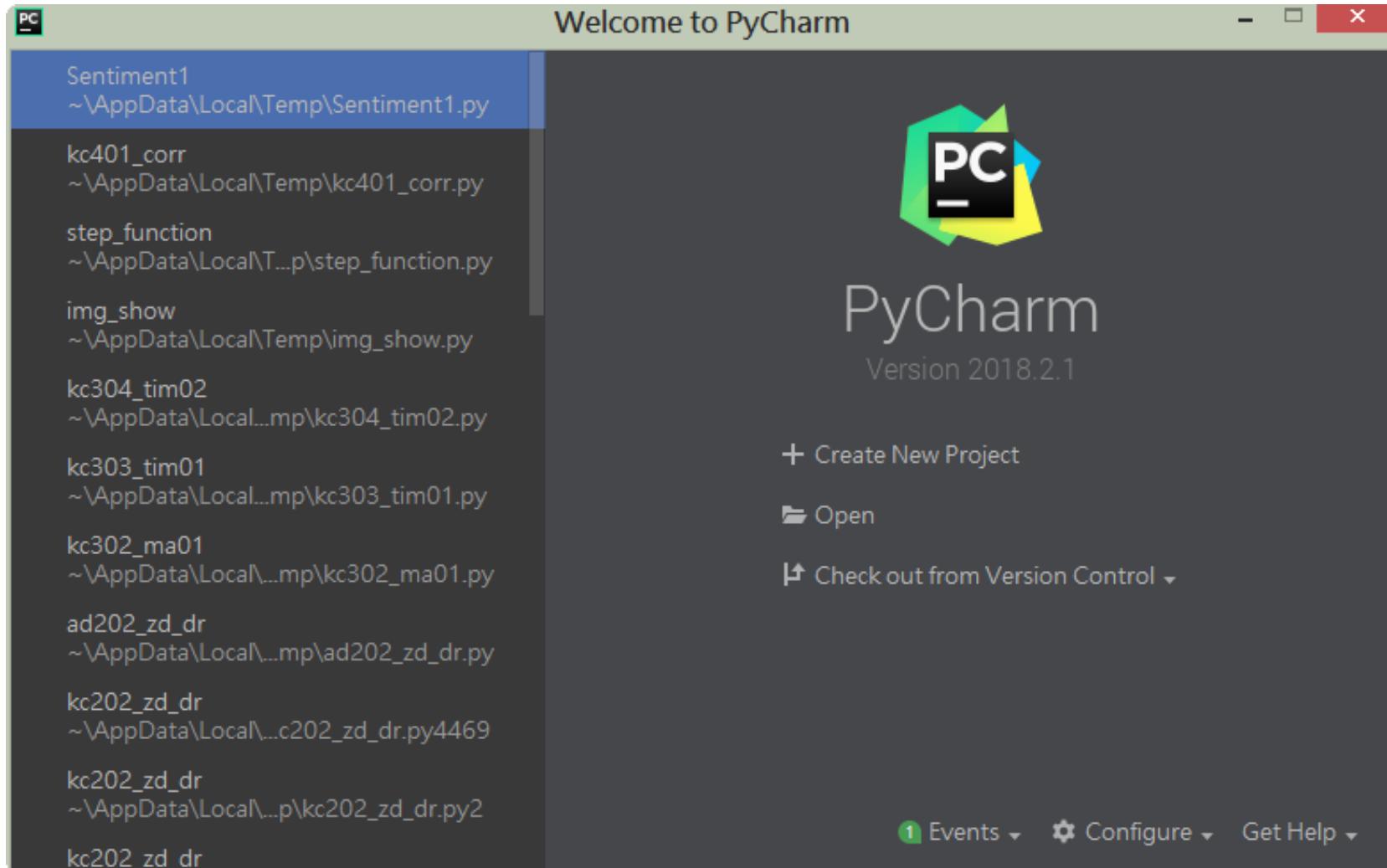
DOWNLOAD

Free, open-source

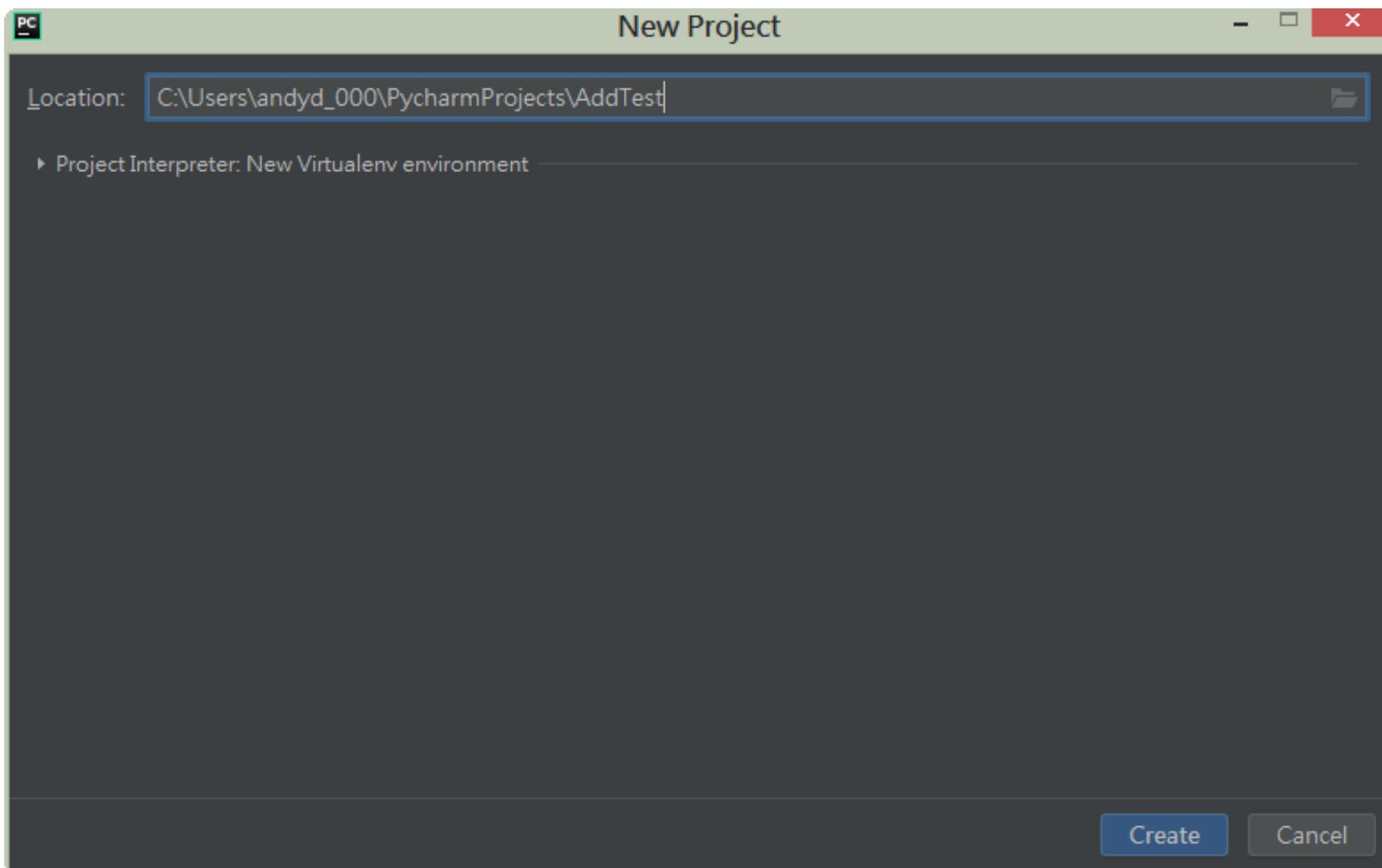
◆ 執行



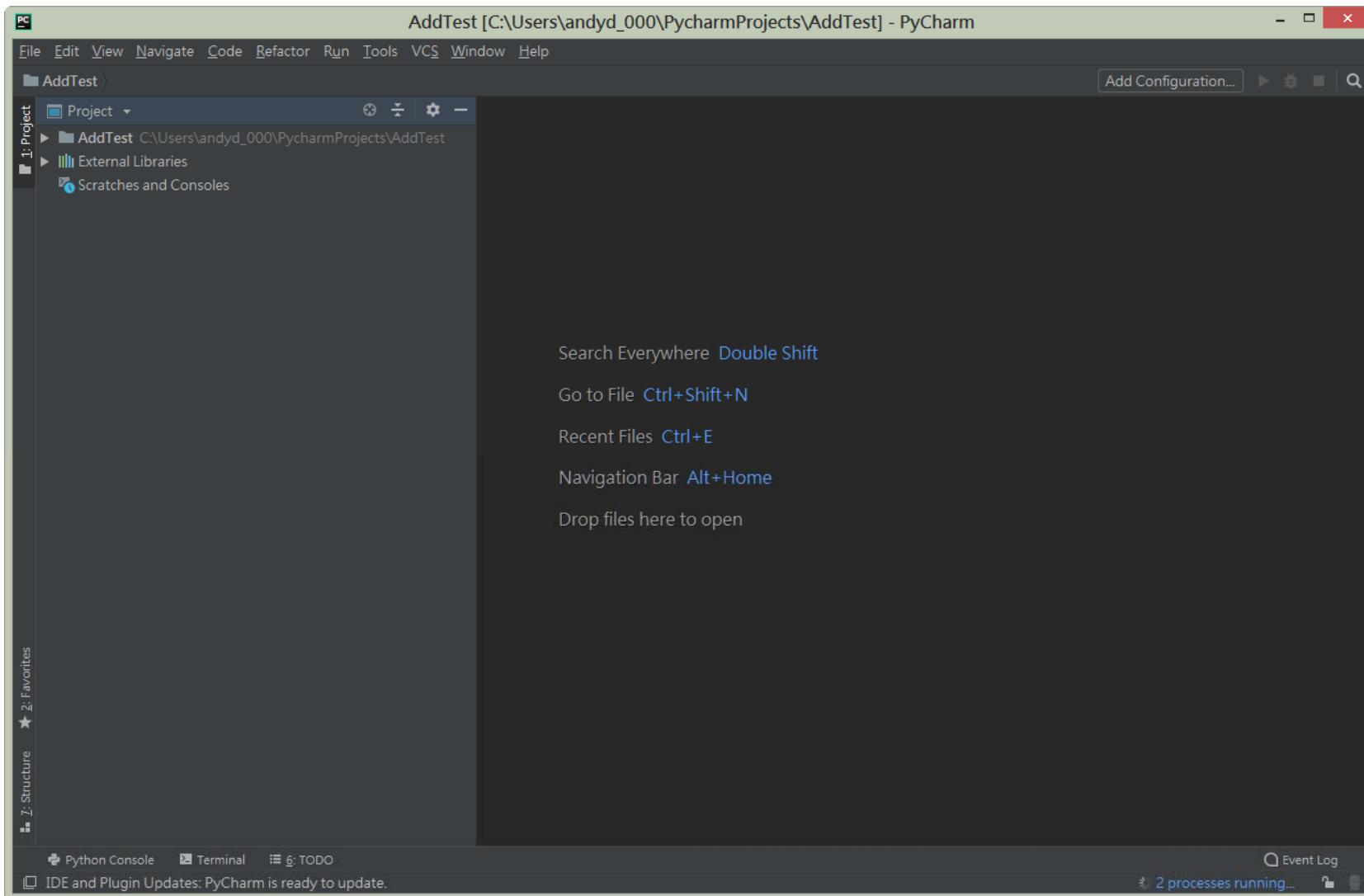
◆ Pycharm



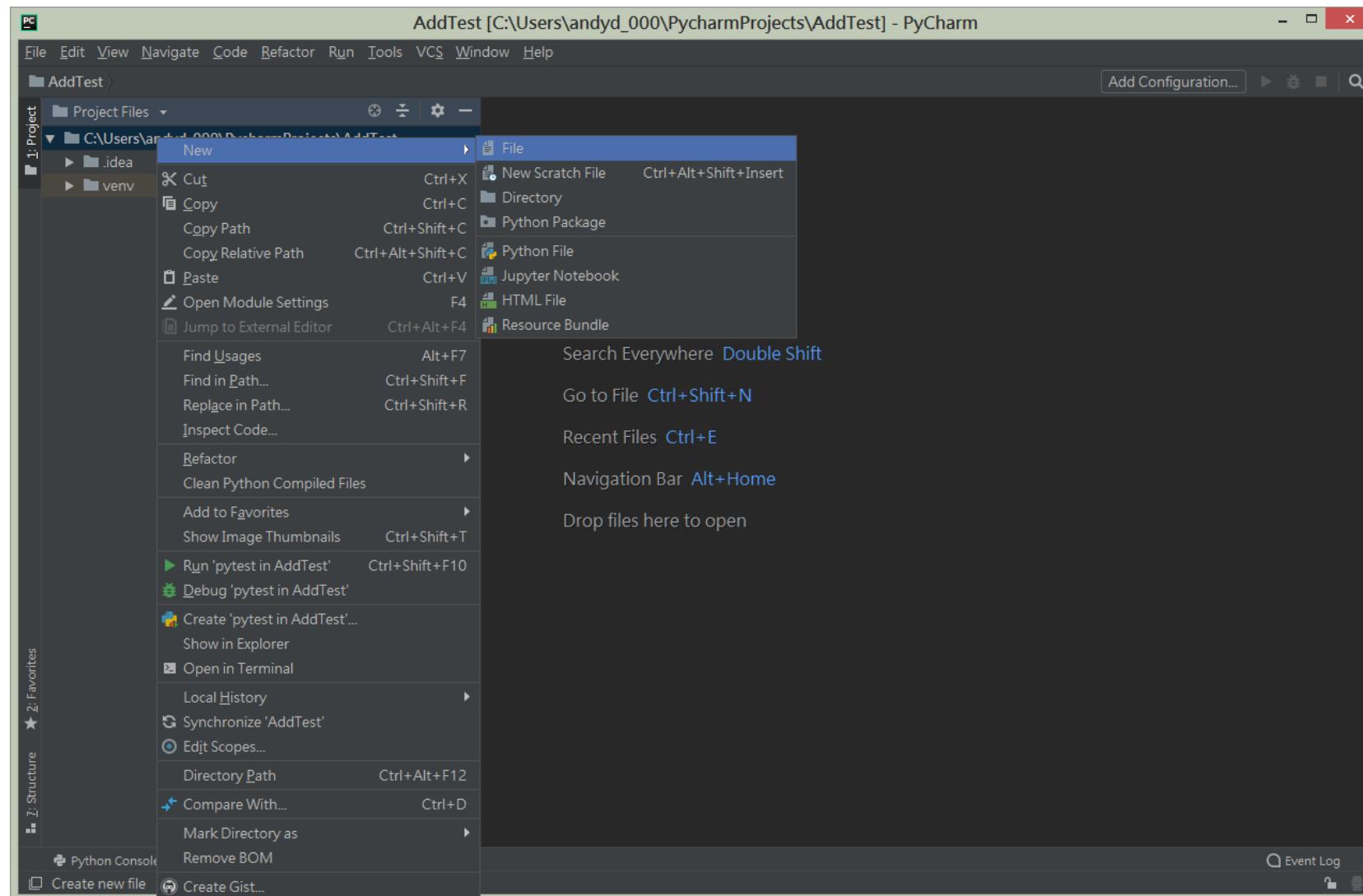
◆ 新增專案



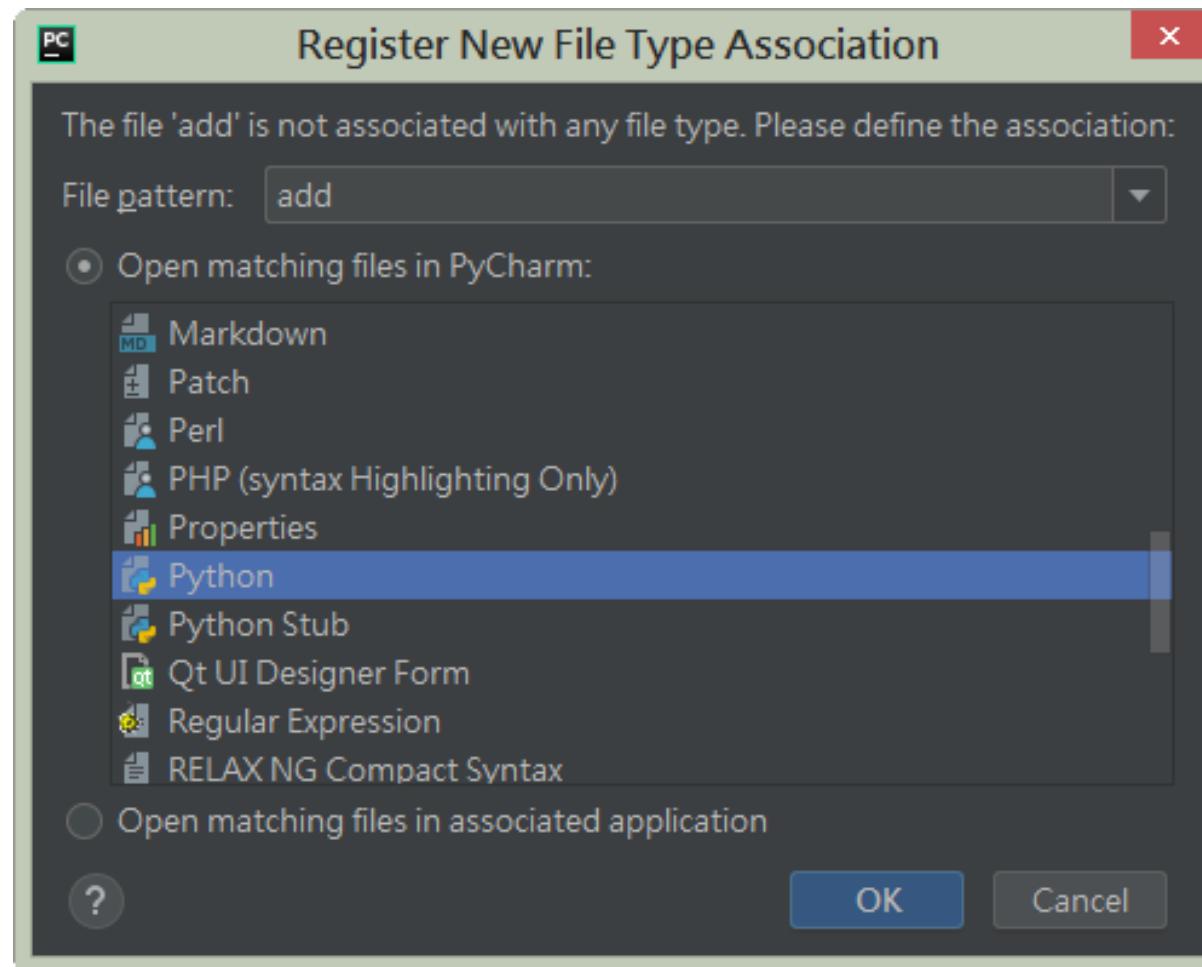
◆ 產生專案



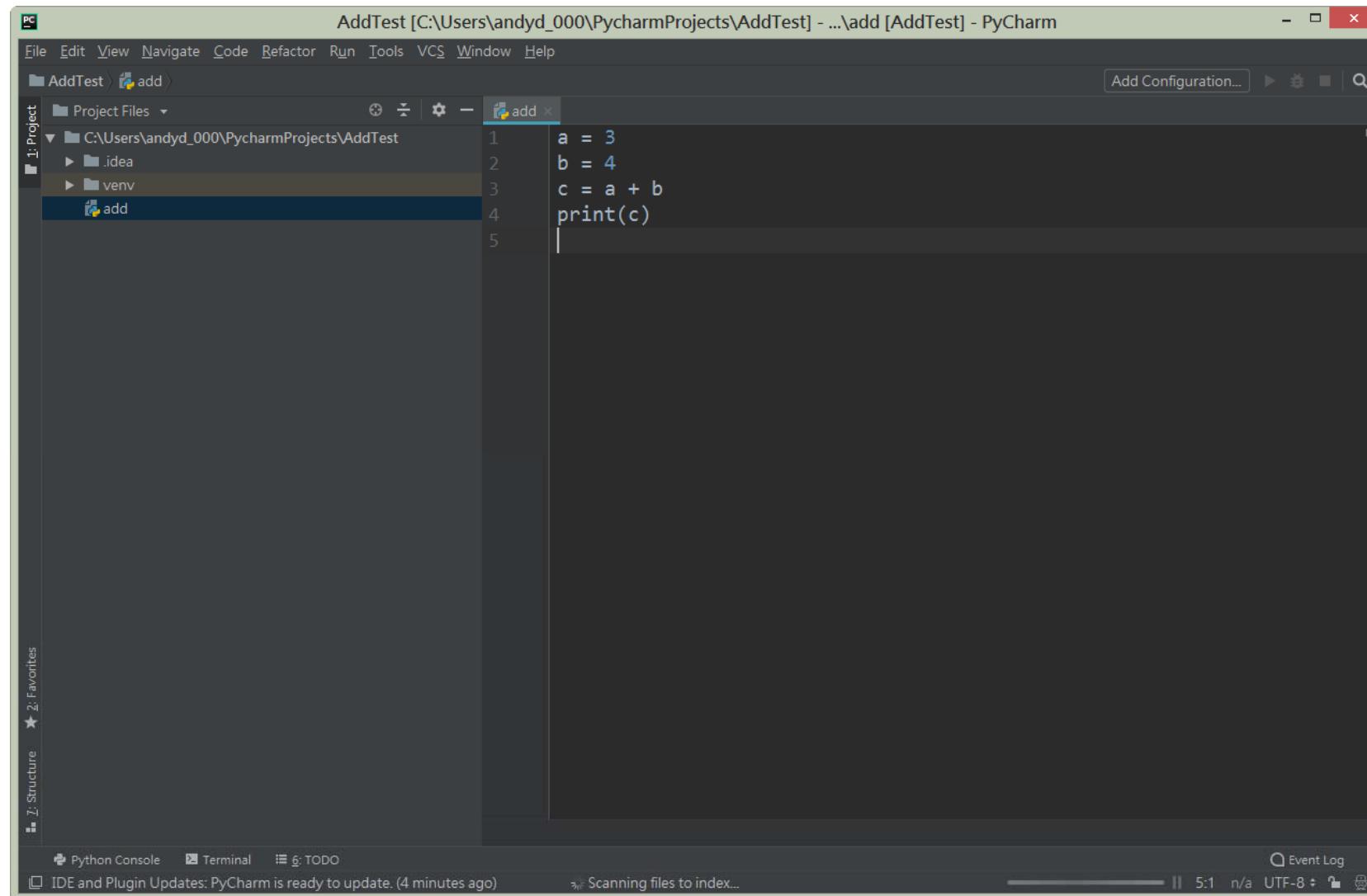
◆ 加入檔案



◆ Python File



◆ 輸入程式



The screenshot shows the PyCharm IDE interface with the following details:

- Title Bar:** AddTest [C:\Users\andyd_000\PycharmProjects\AddTest] - ...\\add [AddTest] - PyCharm
- Menu Bar:** File Edit View Navigate Code Refactor Run Tools VCS Window Help
- Toolbars:** Add Configuration... (top right), and standard PyCharm toolbars.
- Project View:** Shows the project structure with folders .idea, venv, and a file named 'add' which is currently selected.
- Code Editor:** Displays the following Python code:

```
1 a = 3
2 b = 4
3 c = a + b
4 print(c)
5
```
- Status Bar:** Python Console, Terminal, TODO, Event Log, and a message about PyCharm updates.

◆ 執行結果

The screenshot shows the PyCharm IDE interface. The top bar displays the project name "AddTest [C:\Users\andyd_000\PycharmProjects\AddTest] - ...\\add [AddTest] - PyCharm". The menu bar includes File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The left sidebar shows the "Project" view with a tree structure: 1: Project, C:\Users\andyd_000\PycharmProjects\AddTest, idea, venv, and add (which is selected). The main code editor window contains the following Python code:

```
a = 3
b = 4
c = a + b
print(c)
```

The bottom right terminal window shows the execution results:

```
C:\Users\andyd_000\PycharmProjects\AddTest\venv\Scripts\python.exe
C:/Users/andyd_000/PycharmProjects/AddTest/add
7
Process finished with exit code 0
```

The bottom status bar indicates the current time is 5:1, the encoding is UTF-8, and there is an update notification: "IDE and Plugin Updates: PyCharm is ready to update. (4 minutes ago)".

