# 资料：

Python 菜鸟教程

官方文档

Python 学习手册 o’reilly

# Anaconda

<http://www.jianshu.com/p/2f3be7781451>

## Conda 环境

有一个默认的根环境，可以新建环境

如果在默认环境下运行，要用spyder(及其它安装在默认环境下的东西)，则最好在根环境下安装文件。

## Conda 包管理

Conda install tensorflow 会安装许多配套的东西，而且可能不是我要的版本

Conda install tensorflow某个版本的文件，则只会装这个文件

Conda search package 查找包的信息

# Jupyter Notebook

## 适合教学，运行的结果可以保存在文件中，以后方便回顾

## 启用

在命令行输入以下代码：

activate Anaconda3

d:

cd D:\in ZJU\计算机学习\CS231n-Convolutional Neural Networks for Visual Recognition\spring1617\_assignment2\assignment2

jupyter notebook

## Ipynb转换为py

#### Spyder中右击文件

#### jupyter notebook中转换

# Spyder下使用

D:\in ZJU\计算机学习\计算机基础\python基础\zlf\_python.py

# 三种程序运行方式

#### 交互式运行

#### 文件-命令行

在控制台运行python相关命令，需要在annaconda promot下，不能在系统控制台（cmd）下

在spyder中可以快捷地在当前目录下打开控制台:右击文件

#### 、点击图标、IDLE

#### IDE

# 模块

## 作用

import有点类似于c++中的include,方便程序组织，模块中定义了很多函数，类等等，然后又一个py文件中导入这些模块，可以定义主函数，然后去运行程序，或者直接写脚本

## 模块的导入和重载

## 查看变量信息

#### 交互式编程中直接输入变量或属性名

#### 程序中print

#### Spyder变量浏览器

# 数据类型

## 基本：Numbers，Boolean，String，

算术运算：\*\*幂运算，有+=，无++

逻辑运算：and，or,not

字符串

hw12 **=** '%s %s %d' **%** (hello, world, 12) *# sprintf style string formatting*

## 容器：List，Dictionary，Set，Tuple

列表：表示[1,2,3]，append(),pop

字典d **=** {'cat': 'cute', 'dog': 'furry'},

集合animals **=** {'cat', 'dog'} 无序

元组（5，6）不可修改

#### slicing容器的索引

用[a:b]，表示的其实是[a,b]，从0开始，： ，-1表示最后一个

#### 循环

nums **=** list(range(5)) *# range is a built-in function that creates a list of integers*

squares **=** []

**for** x **in** nums:

squares**.**append(x **\*\*** 2)

**print**(squares) *# Prints [0, 1, 4, 9, 16]*

#### 递推式

nums **=** [0, 1, 2, 3, 4]

squares **=** [x **\*\*** 2 **for** x **in** nums] #集合，字典用{}

**print**(squares) *# Prints [0, 1, 4, 9, 16]*

## 语句

条件，循环，判读：有相应关键字的行末尾加：

## 函数def a():

## 类

class a:

\_init\_(self,param)

#类的属性：与一般变量定义一样，要初始化，但要加上self，比如

self.varName=None

#使用时也需要加上self

# Numpy：科学计算的核心库之一

## Array

import numpy as np

a **=** np**.**array([1, 2, 3]) *# Create a rank 1 array*

**print**(type(a)) *# Prints "<class 'numpy.ndarray'>"*

**print**(a**.**shape) *# Prints "(3,)"*

b **=** np**.**array([[1,2,3],[4,5,6]]) *# Create a rank 2 array*

**print**(b**.**shape) *# Prints "(2, 3)"*

**print**(b[0, 0], b[0, 1], b[1, 0]) *# Prints "1 2 4"*

### 索引

Slicing，整数，或者混合（多维），逻辑

#### 每行选一个arange(区别与list的range)

*# Create a new array from which we will select elements*

a **=** np**.**array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])

**print**(a) *# prints "array([[ 1, 2, 3],*

*# [ 4, 5, 6],*

*# [ 7, 8, 9],*

*# [10, 11, 12]])"*

*# Create an array of indices*

b **=** np**.**array([0, 2, 0, 1])

*# Select one element from each row of a using the indices in b*

**print**(a[np**.**arange(4), b]) *# Prints "[ 1 6 7 11]"*

### 运算

#### elementwise ：+，-，×，/

#### 矩阵运算：

a.dot(b),np.dot(a,b)

小结：

严格按照矩阵运算法则

#行或列向量阶数为1：（n，）与矩阵阶数为2：（n,1）

有一个运算对象为一阶:You can treat one-dimensional arrays as

either row or column vectors. dot(A,v) treats v as a column vector,

while dot(v,A) treats v as a row vector.

'''

### 轴

#%% :axis=0是行，1是列，函数使用时的轴表示的是结果的方向

import numpy as np

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

print(np.sum(x)) # Compute sum of all elements; prints "10"

print(np.sum(x, axis=0)) # Compute sum of each column; prints "[4 6]"

print(np.sum(x, axis=1)) # Compute sum of each row; prints "[3 7]"

### 广播

对不同shape的array进行算术运算





