



案例实践一：编程与数据分类

主讲人：屠恩美

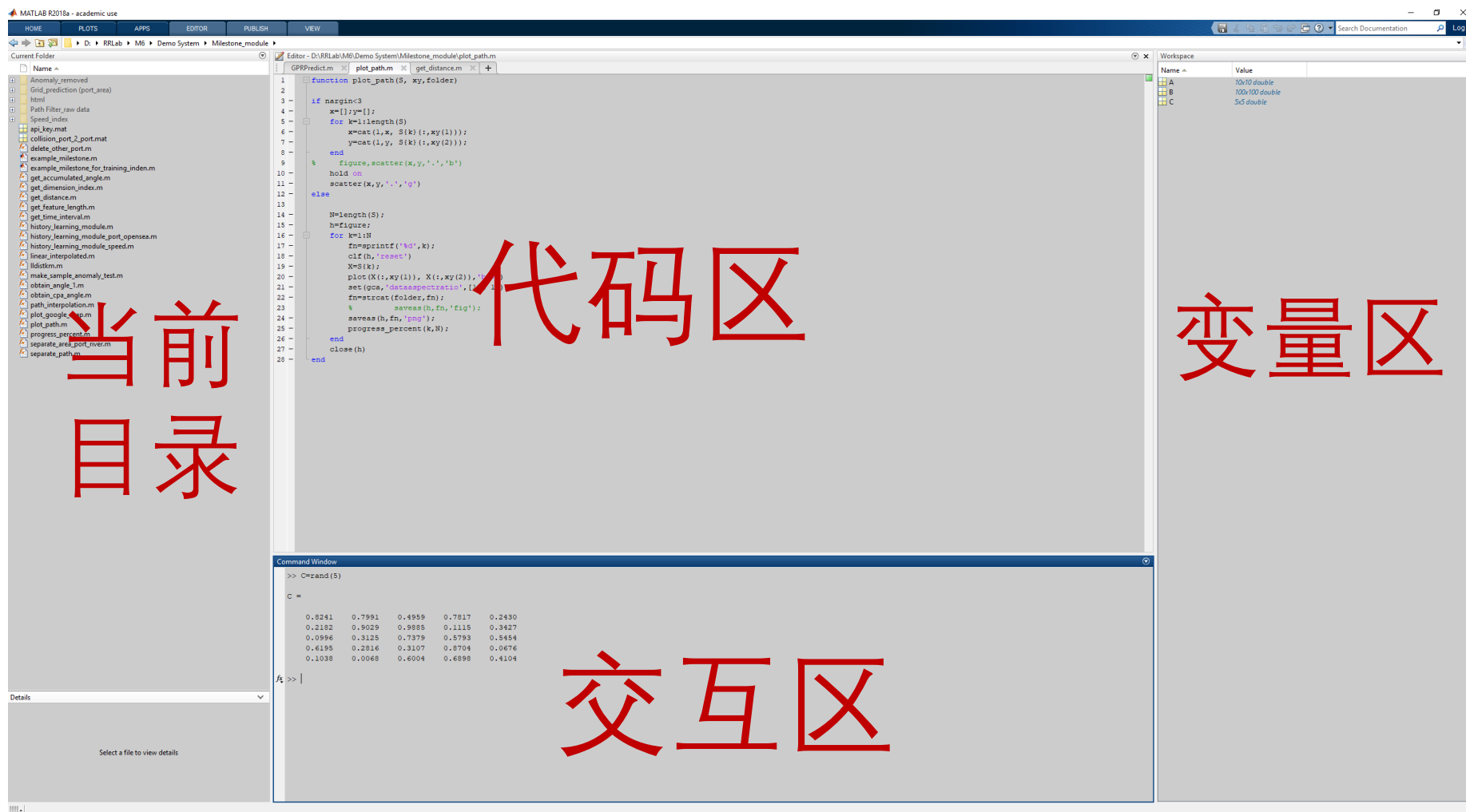
2018.10



上海交通大学

SHANGHAI JIAO TONG UNIVERSITY

MATLAB

当前目录

代码区

变量区

交互区

```
function plot_path(S, xy, folder)
1
2
3
4   x=[];y=[];
5   for k=1:length(S)
6       x=cat(1,x, S(k)(: ,xy(1)));
7       y=cat(1,y, S(k)(: ,xy(2)));
8   end
9   % figure,scatter(X,y,',' , 'b')
10  hold on
11  scatter(X,y,',' , 'g')
12  else
13
14      N=length(S);
15      h=figure;
16      for k=1:N
17          fprintf('%d',k);
18          if(h,'reset')
19              X=S(k);
20              plot(X(: ,xy(1)), X(: ,xy(2)), 'b');
21              set(gca, 'dataaspectratio', [1 1]);
22              fprintf(folder, '%d',k);
23              % saveas(h,fn,'fig');
24              saveas(h,fn,'png');
25              progress_percent(k,N);
26      end
27      close(h)
28  end
```

Name	Value
A	10x10 double
B	100x100 double
C	5x5 double

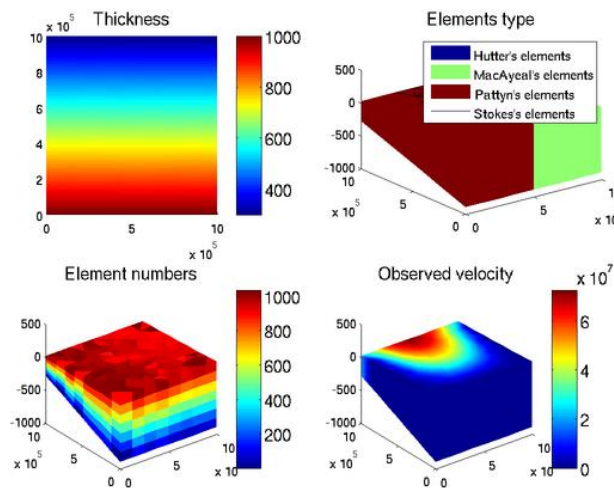
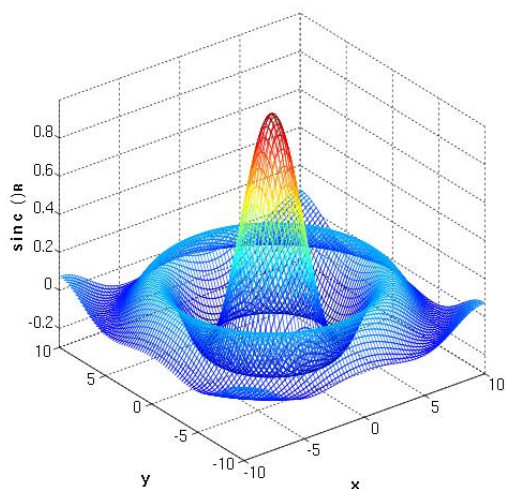
```
>> C=rand(5)
C =
    0.8241    0.7991    0.4959    0.7817    0.2430
    0.2182    0.9029    0.9885    0.1115    0.3427
    0.0996    0.3125    0.7379    0.5793    0.5454
    0.6195    0.2816    0.3107    0.8704    0.0676
    0.1038    0.0068    0.6004    0.6898    0.4104

f5 >> |
```

MATLAB



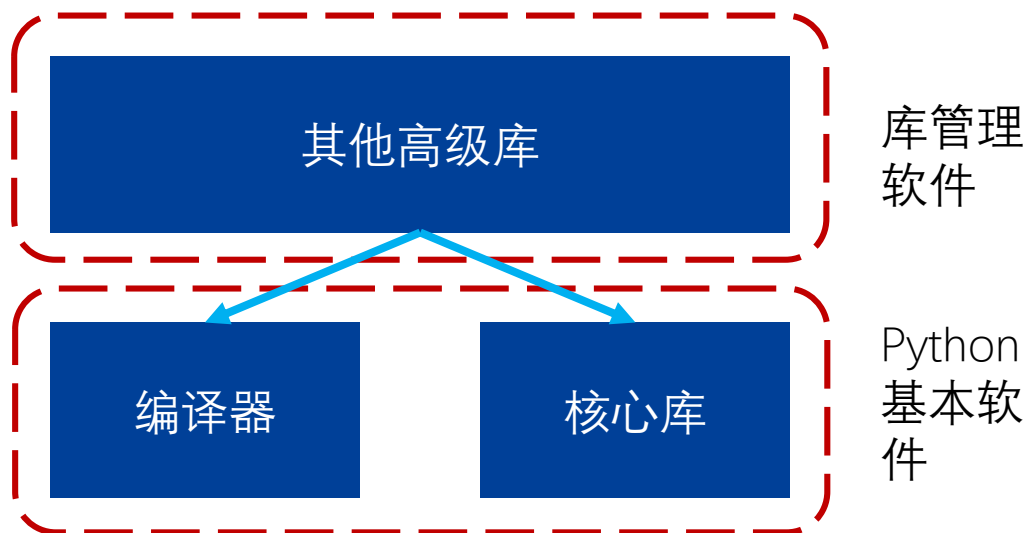
- 编程语言简洁，尤其适合科学计算
- 庞大的计算工具箱集成在同一个环境下，使用便捷
- 丰富的数据可视化(绘图)功能
- 专业的维护，算法实现可靠，更新速度快
- 集成帮助文档全面、细致、专业，教科书级别



Python简介



- Python是一种高级编程语言，1991年发布
- 简单易用，更接近自然语言书写
- 下载网站: <https://www.python.org/downloads/release>
- 版本: 3.6.x



Guido van Rossum

集成安装 - Anaconda




- Anaconda是一个集成的Python软件包/环境管理软件
- 下载地址: <https://www.anaconda.com/download/>
- 下载版本: 3.6 (Windows, Linux, Mac都可以)
- 安装: 全部默认, 直接点“Next”
- 安装后(可能需要重启电脑), 启动Anaconda Navigator

集成安装 - Anaconda

Anaconda Navigator

File Help

 ANACONDA NAVIGATOR[Sign in to Anaconda Cloud](#) Home Environments Learning Community[Documentation](#)[Developer Blog](#)[Feedback](#)

Applications on

base (root)

Channels

Refresh



jupyterlab

0.32.1

An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Launch



notebook

5.5.0

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Launch



qtconsole

4.3.1

PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Launch



spyder

3.2.8

Scientific PYTHON Development Environment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

Launch



glueviz

0.13.3

Multidimensional data visualization across files. Explore relationships within and among related datasets.

Install



orange3

3.13.0

Component based data mining framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Install



rstudio

1.1.423

A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Install



vscode

1.26.1

Streamlined code editor with support for development operations like debugging, task running and version control.

Install

集成安装 - Anaconda

Anaconda Navigator

File Help

ANACONDA NAVIGATOR

Sign in to Anaconda Cloud

The screenshot shows the Anaconda Navigator application window. On the left sidebar, the 'Environments' tab is selected and highlighted with a red dashed box. The main panel displays the 'base (root)' environment. A red dashed box highlights the top navigation bar, which includes a search bar, a dropdown menu for 'Installed' (showing options like 'Installed', 'Not installed', 'Updatable', 'Selected', and 'All'), and buttons for 'Channels', 'Update index...', and 'Search Packages'. Below this, a table lists installed packages with their descriptions and versions.

	Description	Version
Configurable, python 2+3 compatible sphinx theme		0.1.0
		0.7.10
		5.2.0
anaconda-client	Anaconda.org command line client library	1.6.14
anaconda-project	Reproducible, executable project directories	0.8.2
asn1crypto	Asn.1 parser and serializer	0.24.0
astroid	Abstract syntax tree for python with inference support	1.6.3
astropy	Community-developed python library for astronomy	3.0.2
attrs	Implement attribute-related object protocols without boilerplate	18.1.0
babel	Utilities to internationalize and localize python applications	2.5.3
backcall		0.1.0
backports		1.0
backports.shutil_g...		1.0.0
beautifulsoup4	Python library designed for screen-scraping	4.6.0
bitarray	Efficient representation of arrays of booleans -- c extension	0.8.1
bkcharts	Optional high level charts api built on top of bokeh	0.2


243 packages available

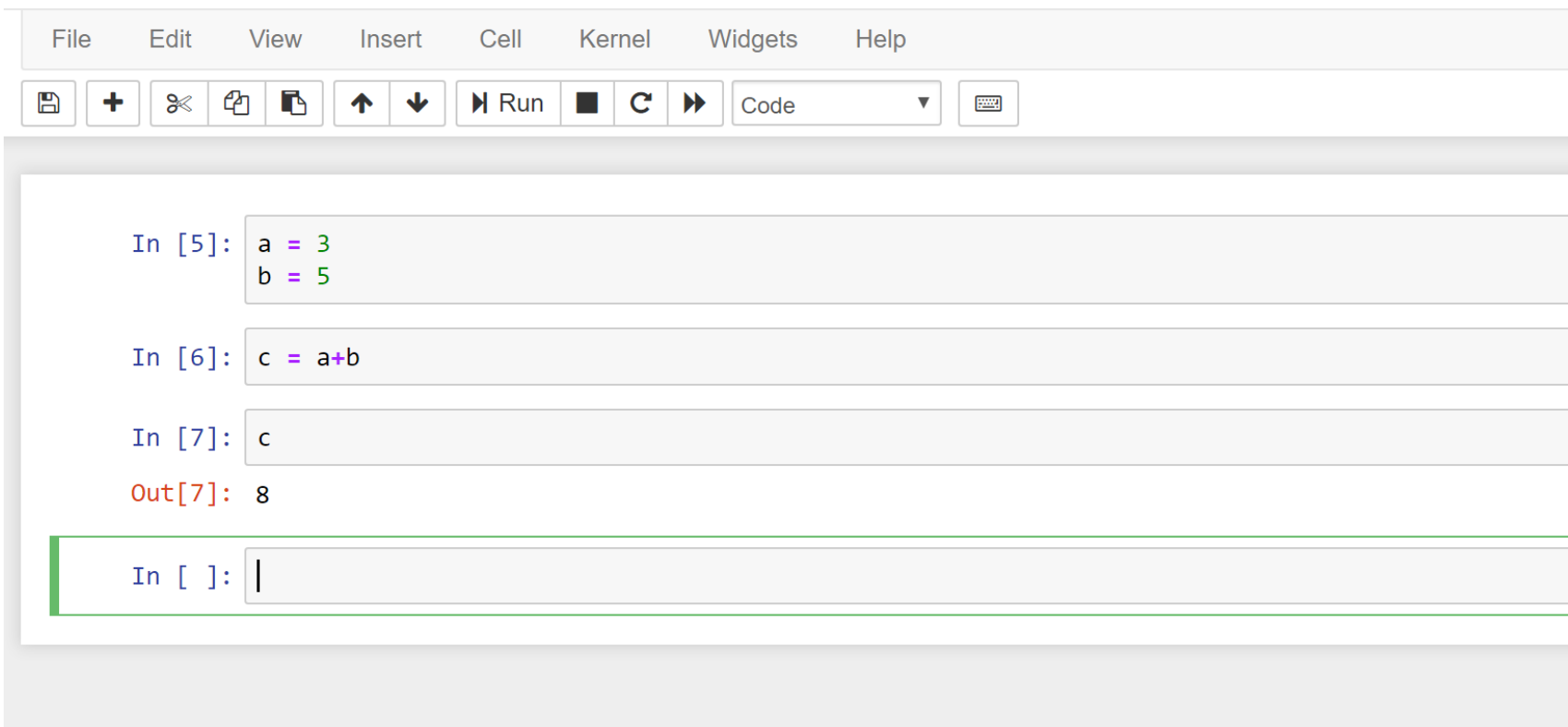
集成安装 - Jupyter



集成安装 - Jupyter



 jupyter Untitled1 Last Checkpoint: 3 minutes ago (unsaved changes)



The screenshot displays the Jupyter Notebook interface. At the top is a menu bar with options: File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. Below the menu bar is a toolbar containing icons for saving, creating a new file, opening a recent file, closing a file, navigating between cells, running a cell, and a dropdown menu currently set to 'Code'. The main workspace contains three code cells. The first cell, labeled 'In [5]:', contains the code `a = 3` and `b = 5`. The second cell, labeled 'In [6]:', contains the code `c = a+b`. The third cell, labeled 'In [7]:', contains the code `c`. Below the third cell, the output is displayed as 'Out[7]: 8'. A fourth code cell, labeled 'In []:', is currently active and empty, with a cursor at the end of the line.

运行某个Cell: “Run” 按钮或者 “Shift”+“ Enter”

集成安装 - Spyder



代码区

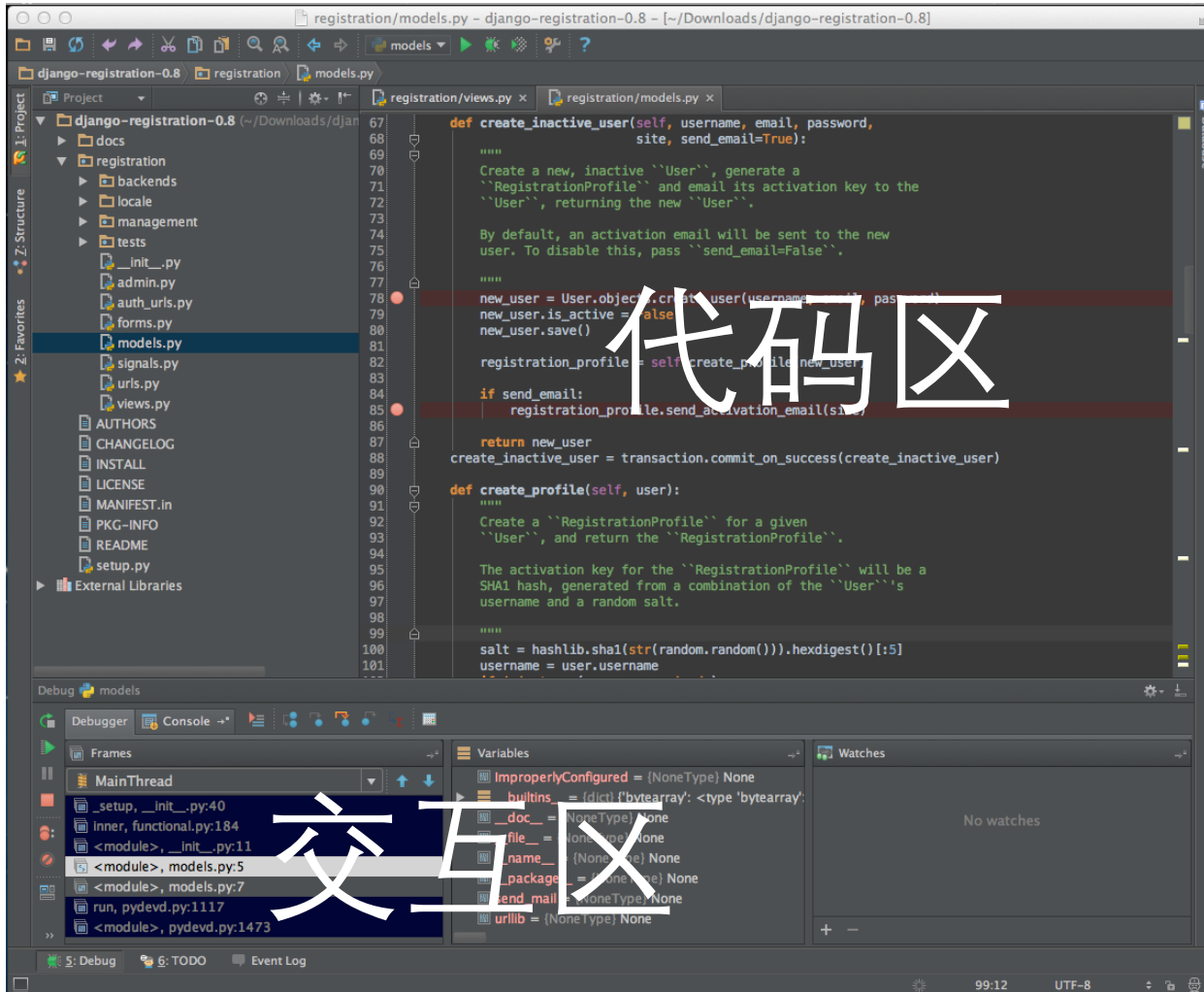
交互区

按需安装



- 下载并安装python 3.65:
<https://www.python.org/downloads/release/python-365/>
- 安装所需的软件包: pip install packages (numpy, scipy, matplotlib, etc)
- 安装Jupyter:
 - `python3 -m pip install --upgrade pip`
 - `python3 -m pip install jupyter`
- 或者安装Pycharm
<https://www.jetbrains.com/pycharm-edu/download/#section=windows>

PyCharm



环境比较



Jupyter 还是 PyCharm 还是 Spyder?

- Jupyter 使用方便，交互性好，容易上手，适合初学和小项目
- PyCharm使用稍微繁琐，但功能强大：调试方便，代码提示，版本分离
- Spyder介于二者之间，针对科学计算而开发，对标Matlab部分功能
- 另外PyCharm和Spyder界面可自定义(字体，颜色，主题等)，按个人习惯设置



Python编程



```
x = 34 - 23 # A comment.  
y = "Hello" # Another one.  
z = 3.45  
if z == 3.45 or y == "Hello":  
    x = x + 1  
    y = y + "World" # String concat.  
print(x)  
print(y)
```

12

HelloWorld

Python编程 – if, for, while



- if, for, while等语句块以缩进控制，冒号： 起始

```
a = 5
if a < 3:
    a += 1

for i in range(a):
    print(i)
```

0
1
2
3
4

Python编程 – 函数



- 函数使用def定义，缩进控制函数范围
- 接收两种参数：位置参数和关键字参数

```
def add(x, y=1):  
    z = x+y  
    return z  
  
a = 1  
b = 2  
c = add(a)  
d = add(a, b)  
print(c,d)
```

2 3

Python编程 – 类



- 类使用class定义，缩进控制类成员

```
class AutoCounter(object):  
    def __init__(self, base):  
        if base > 0:  
            self.base = base  
        else:  
            base = 10  
        print(self.base)  
    def increase(self, a=1):  
        self.base += a  
        print(self.base)
```

```
c1 = AutoCounter(5)  
c1.increase()
```

5

6

Python编程 – 基本数据类型



- 四种基本数据类型：int, float, complex, string

Numbers: Integers and floats work as you would expect from other languages:

```
x = 3
print(type(x)) # Prints "<class 'int'>"
print(x)       # Prints "3"
print(x + 1)    # Addition; prints "4"
print(x - 1)    # Subtraction; prints "2"
print(x * 2)    # Multiplication; prints "6"
print(x ** 2)   # Exponentiation; prints "9"
x += 1
print(x)        # Prints "4"
x *= 2
print(x)        # Prints "8"
y = 2.5
print(type(y))  # Prints "<class 'float'>"
print(y, y + 1, y * 2, y ** 2) # Prints "2.5 3.5 5.0 6.25"
```


Python编程 – 基本数据类型



Strings: Python has great support for strings:

```
hello = 'hello'      # String literals can use single quotes
world = "world"      # or double quotes; it does not matter.
print(hello)         # Prints "hello"
print(len(hello))    # String length; prints "5"
hw = hello + ' ' + world # String concatenation
print(hw)            # prints "hello world"
hw12 = '%s %s %d' % (hello, world, 12) # sprintf style string formatting
print(hw12)          # prints "hello world 12"
```

Python编程 – 高级数据类型



- 四种符合数据类型 [List](#), Dictionary, Tuple, Set

```
xs = [3, 1, 2]      # Create a list
print(xs, xs[2])    # Prints "[3, 1, 2] 2"
print(xs[-1])       # Negative indices count from the end of the list; prints "2"
xs[2] = 'foo'       # Lists can contain elements of different types
print(xs)           # Prints "[3, 1, 'foo']"
xs.append('bar')     # Add a new element to the end of the list
print(xs)           # Prints "[3, 1, 'foo', 'bar']"
x = xs.pop()        # Remove and return the last element of the list
print(x, xs)        # Prints "bar [3, 1, 'foo']"
```

```
nums = list(range(5))    # range is a built-in function that creates a list of integers
print(nums)              # Prints "[0, 1, 2, 3, 4]"
print(nums[2:4])         # Get a slice from index 2 to 4 (exclusive); prints "[2, 3]"
print(nums[2:])           # Get a slice from index 2 to the end; prints "[2, 3, 4]"
print(nums[:2])           # Get a slice from the start to index 2 (exclusive); prints "[0, 1]"
print(nums[:])            # Get a slice of the whole list; prints "[0, 1, 2, 3, 4]"
print(nums[:-1])          # Slice indices can be negative; prints "[0, 1, 2, 3]"
nums[2:4] = [8, 9]       # Assign a new sublist to a slice
print(nums)              # Prints "[0, 1, 8, 9, 4]"
```

Python编程 – 高级数据类型



- 四种符合数据类型 List, [Dictionary](#), Tuple, Set

```
d = {'cat': 'cute', 'dog': 'furry'} # Create a new dictionary with some data
print(d['cat']) # Get an entry from a dictionary; prints "cute"
print('cat' in d) # Check if a dictionary has a given key; prints "True"
d['fish'] = 'wet' # Set an entry in a dictionary
print(d['fish']) # Prints "wet"
# print(d['monkey']) # KeyError: 'monkey' not a key of d
print(d.get('monkey', 'N/A')) # Get an element with a default; prints "N/A"
print(d.get('fish', 'N/A')) # Get an element with a default; prints "wet"
del d['fish'] # Remove an element from a dictionary
print(d.get('fish', 'N/A')) # "fish" is no longer a key; prints "N/A"
```

```
d = {'person': 2, 'cat': 4, 'spider': 8}
for animal in d:
    legs = d[animal]
    print('A %s has %d legs' % (animal, legs))
# Prints "A person has 2 legs", "A cat has 4 legs", "A spider has 8 legs"
```

Python编程 – 高级数据类型



- 四种符合数据类型 List, Dictionary, [Tuple](#), Set

```
d = {(x, x + 1): x for x in range(10)} # Create a dictionary with tuple keys
t = (5, 6) # Create a tuple
print(type(t)) # Prints "<class 'tuple'>"
print(d[t]) # Prints "5"
print(d[(1, 2)]) # Prints "1"
```

- 与list的区别：
 - Tuple中的数值一旦创建后，不能被修改
 $t[0] = 2$ ❌
 - Tuple可以做dictionary的索引值

Python编程 – 高级数据类型



- 四种符合数据类型 List, Dictionary, Tuple, [Set](#)

```
animals = {'cat', 'dog'}  
print('cat' in animals)    # Check if an element is in a set; prints "True"  
print('fish' in animals)   # prints "False"  
animals.add('fish')        # Add an element to a set  
print('fish' in animals)   # Prints "True"  
print(len(animals))        # Number of elements in a set; prints "3"  
animals.add('cat')         # Adding an element that is already in the set does nothing  
print(len(animals))        # Prints "3"  
animals.remove('cat')      # Remove an element from a set  
print(len(animals))        # Prints "2"
```


Numpy



- Numpy is the core library for scientific computing in Python
- 基本数据结构是矩阵(array), 包括向量, 矩阵和张量(多维矩阵)

```
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,)"
print(a[0], a[1], a[2])   # Prints "1 2 3"
a[0] = 5                  # Change an element of the array
print(a)                  # Prints "[5, 2, 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"
```

Matplotlib

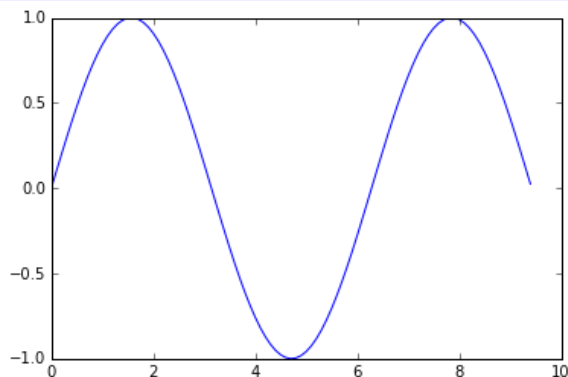


- Matplotlib is a plotting library to generate various types of figures

```
import numpy as np
import matplotlib.pyplot as plt

# Compute the x and y coordinates for points on a sine curve
x = np.arange(0, 3 * np.pi, 0.1)
y = np.sin(x)

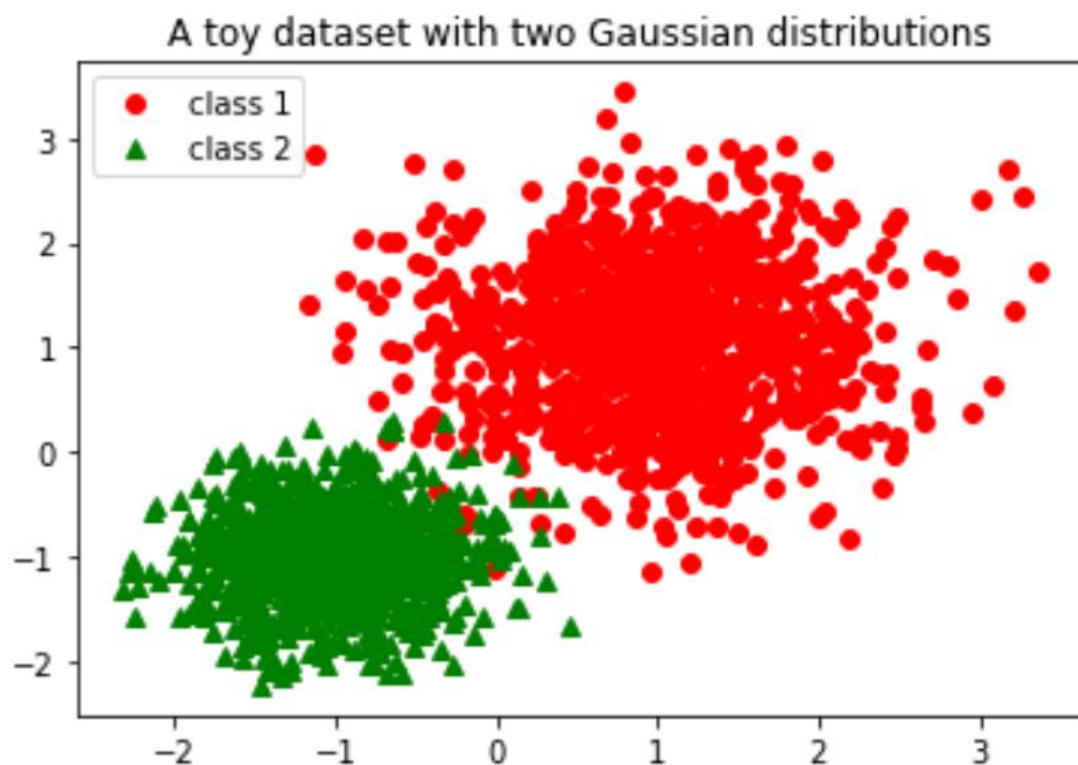
# Plot the points using matplotlib
plt.plot(x, y)
plt.show() # You must call plt.show() to make graphics appear.
```



更多例子:

<http://cs231n.github.io/python-numpy-tutorial/>

双高斯分布数据集



	feature 1	feature2	label
0	0.541078	0.084685	1.0
1	1.835211	0.414565	1.0
2	0.463693	0.600649	1.0
3	0.215385	0.838101	1.0
4	0.402105	1.605083	1.0
5	0.595065	0.079514	1.0
6	0.604953	0.947740	1.0
7	1.126134	-0.553929	1.0
8	0.860798	1.610948	1.0
9	1.414678	1.456316	1.0

作业



■ 理论题：25%

1) 写出最小二乘求解如下广义线性模型的 w, b 详细推到过程

$$y = e^{wx+b}$$

2) 假设有三家工厂A, B, C共同生产一种台灯，他们产品占比和次品率分别如下：

工厂名 (类别名)	产品占比(先验概率)	次品率(条件概率)
A	$0.35 = P(A)$	$0.015 = P(\text{次品} A)$
B	$0.35 = P(B)$	$0.010 = P(\text{次品} B)$
C	$0.30 = P(C)$	$0.020 = P(\text{次品} C)$

某次随机抽检一个样品，该样品是次品概率有多大？如果该样品是次品，则它来自工厂A, B, C 的概率分别有多大？

作业



- 实践题：75%=25% x 3
 - 1) 实现线性分类器并在西瓜3.0数据集上用前80%训练、后20%测试时的精度
 - 2) 实现Naïve Bayes分类器并在西瓜3.0数据集上测试k=5重交叉验证精度
 - 3) 比较SVM使用不同（至少4种）核函数时，西瓜3.0数据集上用前80%训练、后20%测试的精度（可使用任意svm算法实现软件包）
- 附加题：（可不做，做对20%额外分）

实现对数几率回归并在西瓜3.0上与线性分类器、NB和SVM做性能比较
- 注意：只允许实践题3)中SVM算法使用已有库中的实现，其他均需自己实现（**发现作弊或抄袭，本次作业0分处理**）。

作业提交



- 文档：包括理论题的回答和实践题的结果截图
- 代码：包括所有运行需要的代码和数据
- 把以上两部分压缩在一个压缩包，发送到邮箱wangzihao33@sjtu.edu.cn
- 邮件标题格式：课程ML作业1_学号_姓名
- DDL：第5周周日（10月16日）23:59之前（以邮件收到时间为准）