**大家好，我是renrenyi。我想和大家分享一下我学习python以及machine learning的过程，希望能给大家一点借鉴，所以决定做这个视频。**

**推荐两个网站，大家可以看一下。**

**Python基础教程|菜鸟教程** [**http://www.runoob.com/python/python-intro.html**](http://www.runoob.com/python/python-intro.html)

**莫烦python** [**https://morvanzhou.github.io/tutorials/python-basic/basic/**](https://morvanzhou.github.io/tutorials/python-basic/basic/)

整个操作都是在VM虚拟机上完成的，Win 10，64位，刚装好的全新系统

**大家好，我叫renrenyi。我想和大家分享一下我学习python以及machine learning的过程，希望能给大家一点借鉴，所以决定做这个视频。本人非计算机专业的，所以可能会有一些错误等，希望可以多交流。**

#### Python的安装与测试- 基于Windows 10 / 64位系统

1. **测试是否安装了python**

**打开命令行command/cmd**

测试，输入python，如果显示出版本信息，说明安装了python；如果报错，表明系统没有安装python。

1. **Python下载**

百度/Google一下python

Python官网：<https://www.python.org/>

下载python3.5.4版本，for Windows 10，64位

1. **Window 平台安装 Python**

选择exe文件，安装python

1. **环境变量配置**

（1）右键点击"计算机"，然后点击"属性"

（2）然后点击"高级系统设置"

（3）选择"系统变量"窗口下面的"Path"

（4）然后在"Path"行，添加python安装路径即可(我的C:\Python35)，所以在后面，添加该路径即可。

C:\Python35 和 C:\Python35\Scripts

1. **测试是否安装成功**

cmd输入python，显示出版本等信息

1. **运行python**
2. cmd运行

里面输入print(“hello, LEO”)

print(1+2)

（2）打开IDLE，输入print(“hello, LEO”)，按回车键

新建一个.py格式的文件，

（3）集成开发环境，安装anaconda或者用PyCharm --以后再聊

#### 二. Python 基础语法- 注释/print/input/行和缩进

1. **注释**

# 注释单行

''' ''' 注释一段

# python单行注释是用井号

# python多行是符号用三引号，可以使三队单引号，也可以是三对双引号

'''

一段注释

一段注释

'''

"""

注释

注释

"""

1. **print**

print后加括号，以前版本的python是不用加括号的，3.x版本需要加括号，格式print()

输出数字 print(123) print(1+2)

输出字符串 print(“Hello”) print(‘LEO’)

换行输出

不换行输出 变量末尾加上逗号 ,

1. **等待用户输入input**

name = input("Please intput your name:")

print('hello, '+name)

1. **行和缩进**

学习 Python 与其他语言最大的区别就是，Python 的代码块不使用大括号 {} 来控制类，函数以及其他逻辑判断。python 最具特色的就是用缩进来写模块。

缩进的空白数量是可变的，但是所有**代码块语句必须包含相同的缩进空白数量**，这个必须严格执行。如下所示：

if True:

print("True")

else:

print("False")

if True:

print("Answer")

print("True")

else:

print("Answer")

# 没有严格缩进，在执行时会报错

print("False")

#### 三. Python基本数据类型- 数字/字符串/列表/元组/字典

1. **变量赋值**

等号（=）用来给变量赋值。

等号（=）运算符左边是一个变量名,等号（=）运算符右边是存储在变量中的值。例如：

name = "LEO"

print(name)

多个变量赋值

a = b = c = 1

a, b, c = 1, 2, "leo"

1. **标准数据类型**

单引号 ' 或双引号 " 括起来，同时使用反斜杠 \ 转义特殊字符。

1. **Python数字**

int（整型，就是整数）

float（浮点型，带有小数）

bool（布尔型，true/false）

complex（复数）

a=100

b=100.0

c=True

d=100+10j

print(type(a))

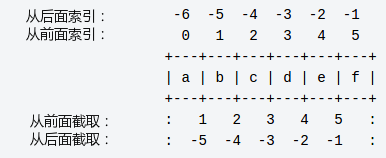
print(type(b))

print(type(c))

print(type(d))

1. **Python字符串 加双引号/单引号**

**字符串用单引号 ' 或双引号 " 括起来，同时使用反斜杠 \ 转义特殊字符。**



str = "Hello LEO!"

print(str) # 输出完整字符串

print(str[0]) # 输出字符串中的第一个字符

print(str[2:5]) # 输出字符串中第三个至第五个之间的字符串

print(str[2:]) # 输出从第三个字符开始的字符串

print(str \* 2) # 输出字符串两次

print(str + "TEST") # 输出连接的字符串

1. **Python列表**

列表用 [ ] 标识，是 python 最通用的复合数据类型。

list1 = [ 'Love', 520 , 5.21, 'Leo', 666 ]

list2 = [ 123, 'Happy']

print(list1) # 输出完整列表

print(list1[0]) # 输出列表的第一个元素

print(list1[1:3]) # 输出第二个至第三个元素

print(list1[2:]) # 输出从第三个开始至列表末尾的所有元素

# 加号 + 是列表连接运算符，星号 \* 是重复操作。

print(list1 + list2) # 打印组合的列表

print(list2 \* 2) # 输出列表两次

1. **Python元组**

元组用"()"标识。内部元素用逗号隔开。但是**元组不能二次赋值，相当于只读列表。**

tuple1 = ( 'Love', 520 , 5.21, 'Leo', 666 )

tuple2 = (123, 'Happy')

print(tuple1) # 输出完整元组

print(tuple1[0]) # 输出元组的第一个元素

print(tuple1[1:3]) # 输出第二个至第三个的元素

print(tuple1[2:]) # 输出从第三个开始至列表末尾的所有元素

print(tuple2 \* 2) # 输出元组两次

print(tuple1 + tuple2) # 打印组合的元组

list1[2] = 1000 # 列表中是合法应用

print(list1)

tuple1[2] = 1000 # 元组中是非法应用

1. **Python 字典**

字典用"{ }"标识。字典由索引(key)和它对应的值value组成。

字典当中的元素是通过**键**来存取的，而不是通过偏移存取。

dict1 = {}

dict1['LEO'] = "Hello, LEO"

dict1[2] = "This is 2"

dict2 = {'name': 'Jack','code':521, 'dept': 'sales'}

print(dict1['LEO']) # 输出键为'one' 的值

print(dict1[2]) # 输出键为 2 的值

print(dict2) # 输出完整的字典

print(dict2.keys()) # 输出所有键

print(dict2.values()) # 输出所有值

dict1['LEO']="Cute LEO"

print(dict1['LEO']) #修改后的值

#### 四. Python 运算符

□算术运算符 □比较（关系）运算符 □逻辑运算符 □Python成员运算符

1. **Python算术运算符**

a=10，b=20：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| + | 加 - 两个对象相加 | a + b 输出结果 30 |
| - | 减 - 得到负数或是一个数减去另一个数 | a - b 输出结果 -10 |
| \* | 乘 - 两个数相乘或是返回一个被重复若干次的字符串 | a \* b 输出结果 200 |
| / | 除 - x除以y | b / a 输出结果 2 |
| % | 取模 - 返回除法的余数 | b % a 输出结果 0 |
| \*\* | 幂 - 返回x的y次幂 | 10\*\*4 为10的4次方， 输出结果 10000 |
| // | 取整除 - 返回商的整数部分（**向下取整**） | 9//2 输出结果 4 , 9.0//2.0 输出结果 4.0 |

a = 10

b = 20

c = 0

c = a + b

print("1 - c 的值为：", c)

c = a - b

print("2 - c 的值为：", c)

c = a \* b

print("3 - c 的值为：", c)

c = a / b

print("4 - c 的值为：", c)

c = a % b

print("5 - c 的值为：", c)

1. **Python比较运算符**

a=10，b=20：

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| == | 等于 - 比较对象是否相等 | (a == b) 返回 False。 |
| != | 不等于 - 比较两个对象是否不相等 | (a != b) 返回 true. |
| > | 大于 - 返回x是否大于y | (a > b) 返回 False。 |
| < | 小于 - 返回x是否小于y。所有比较运算符返回1表示真，返回0表示假。这分别与特殊的变量True和False等价。 | (a < b) 返回 true。 |
| >= | 大于等于 - 返回x是否大于等于y。 | (a >= b) 返回 False。 |
| <= | 小于等于 - 返回x是否小于等于y。 | (a <= b) 返回 true。 |

a = 20

b = 10

c = 0

if ( a == b ):

print ("1 - a 等于 b")

else:

print ("1 - a 不等于 b")

if ( a != b ):

print ("2 - a 不等于 b")

else:

print ("2 - a 等于 b")

if ( a < b ):

print ("3 - a 小于 b")

else:

print ("3 - a 大于等于 b")

if ( a > b ):

print ("4 - a 大于 b")

else:

print ("4 - a 小于等于 b")

1. **Python逻辑运算符**

|  |  |  |  |
| --- | --- | --- | --- |
| **运算符** | **逻辑表达式** | **描述** | **实例** |
| and | x and y | 布尔"与" - 如果 x 为 False，x and y 返回 False，否则它返回 y 的计算值。 | (a and b) 返回 20。 |
| or | x or y | 布尔"或" - 如果 x 是非 0，它返回 x 的值，否则它返回 y 的计算值。 | (a or b) 返回 10。 |
| not | not x | 布尔"非" - 如果 x 为 True，返回 False 。如果 x 为 False，它返回 True。 | not(a and b) 返回 False |

a=True

b=False

print(a and b)

print(a or b)

print(not a)

1. **Python成员运算符**

|  |  |  |
| --- | --- | --- |
| **运算符** | **描述** | **实例** |
| in | 如果在指定的序列中找到值返回 True，否则返回 False。 | x 在 y 序列中 , 如果 x 在 y 序列中返回 True。 |
| not in | 如果在指定的序列中没有找到值返回 True，否则返回 False。 | x 不在 y 序列中 , 如果 x 不在 y 序列中返回 True。 |

a = 3

b = 6

list1 = [1, 2, 3, 4, 5 ];

if ( a in list1 ):

print("1 - 变量 a 在 list1 中")

else:

print("1 - 变量 a 不在 list1 中")

if ( b not in list1 ):

print("2 - 变量 b 不在 list1 中")

else:

print("2 - 变量 b 在 list1 中")

#### 五. Python 条件语句

Python 编程中 if 语句用于控制程序的执行，基本形式为：

if 判断条件**：**

执行语句1……

else**：**

执行语句2……

name = 'LEO'

if name == 'python': # 判断变量否为'python'

print('Big Welcome') # 输出欢迎信息

else:

print("Welcome, ", name) # 条件不成立时输出变量名称

if 语句的判断条件可以用>（大于）、<(小于)、==（等于）、>=（大于等于）、<=（小于等于）来表示其关系。

当判断条件为多个值时，可以使用以下形式：

if 判断条件1:

执行语句1……

**elif** 判断条件2: #elseif

执行语句2……

elif 判断条件3:

执行语句3……

else:

执行语句4……

#### 六. Python 循环语句1 for/while/嵌套循环

Python提供了for循环和while循环。

|  |  |
| --- | --- |
| **循环类型** | **描述** |
| [while 循环](http://www.runoob.com/python/python-while-loop.html) | 在给定的判断条件为 true 时执行循环体，否则退出循环体。 |
| [for 循环](http://www.runoob.com/python/python-for-loop.html) | 重复执行语句 |
| [嵌套循环](http://www.runoob.com/python/python-nested-loops.html) | 你可以在while循环体中嵌套for循环 |

|  |  |
| --- | --- |
| **控制语句** | **描述** |
| [break 语句](http://www.runoob.com/python/python-break-statement.html) | 在语句块执行过程中终止循环，并且跳出整个循环 |
| [continue 语句](http://www.runoob.com/python/python-continue-statement.html) | 在语句块执行过程中终止当前循环，跳出该次循环，执行下一次循环。 |
| [pass 语句](http://www.runoob.com/python/python-pass-statement.html) | pass是空语句，是为了保持程序结构的完整性。 |

1. **While**

i= 0

while (i <= 3):

print('i:', i)

i = i + 1

1. **For**

Python for循环可以遍历任何序列的项目，如一个列表或者一个字符串。

for循环的语法格式如下：

for iterating\_var in sequence:

statements(s)

# 一颗栗子，字符串

for letter in 'LEO':

print('字母 :', letter)

# 两颗栗子，列表

friends = ['Jack', 'Tom','David']

for friend in friends:

print('我的朋友 :', friend)

通过序列索引迭代:

# 两颗栗子-2，列表

friends = ['Jack', 'Tom','David']

for index in range(len(friends)):

print('我的朋友 :', friends[index])

print(len(friends)) #frinds的长度

print(range(len(friends))) # [0,1,2]

1. **Else**

在 python 中，while … else / for… else 在循环条件为 **false** 时执行 else 语句块：

**While循环：**

i= 0

while (i <= 3):

print('i:', i, '小于等于3')

i = i + 1

else:

print('i:', i, '大于3')

**For循环：**

for letter in 'LEO':

print('字母 :', letter)

else:

print("for循环判断为false")

1. **Python 循环嵌套**

For/while 循环嵌套

打印三遍LEO

i= 0

while (i <= 3):

print('i:', i, '小于等于3') #执行三遍

i = i + 1

i= 0

while (i <= 3):

# while循环开始

for letter in 'LEO':

print('字母 :', letter) #打印LEO的循环，为for循环

i = i + 1

# while循环结束

#### 七. Python 循环语句2 break/continue/pass

#### 八. Python 数字/字符串/日期和时间

#### 九. Python 列表/元组/字典

#### 十. Python 函数