Python Ctypes

- 

- C++可以重载，C语言不能重载
- 

- 库在系统目录(windows 下的system32)或者在当前执行目录
- 与python库的查找路径无关sys.path

Linux版本

- so 动态链接库 64位
- 代码字符集utf-8
- g++ -fPIC -shared -o $@ $< -finput-charset='gbk'
- so库在/usr/lib 或者环境变量路径
- 如果要在当前路径调用库，export LD_LIBRARY_PATH ./

```
jasper@LAPTOP-49F4RJGD: ~/py_lesson/src
testctypes:testctypes.cpp
        g++ -fPIC -shared -o $@ $<
        cp $@ /usr/lib
~
~
~
jasper@LAPTOP-49F4RJGD: /py_lesson/src$ make
g++ -fPIC -shared -o testctypes testctypes.cpp
cp testctypes /usr/lib
```

Ctypes类型对应

ctypes 类型对应

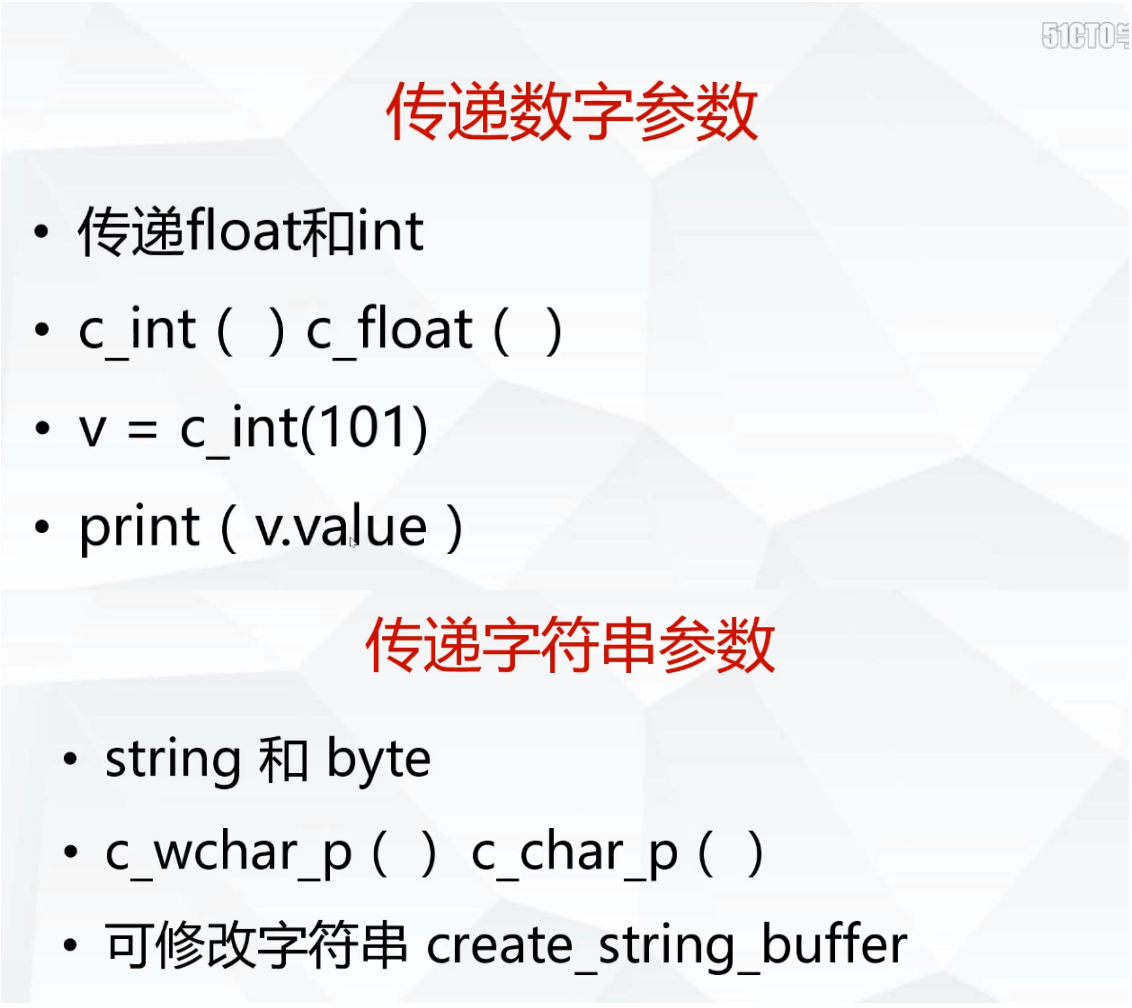
c_size_t	size_t	int
c_ssize_t	ssize_t or Py_ssize_t	int
c_float	float	float
c_double	double	float
c_longdouble	long double	float
c_char_p	char * (NUL terminated)	bytes object or None
c_wchar_p	wchar_t * (NUL terminated)	string or None
c_void_p	void *	int or None

- python的类型与c语言类型的转换

- 

ctypes type	C type	Python type
c_bool	_Bool	bool (1)
c_char	char	1-character bytes object
c_wchar	wchar_t	1-character string
c_byte	char	int
c_ubyte	unsigned char	int
c_short	short	int
c_ushort	unsigned short	int
c_int	int	int
c_uint	unsigned int	int
c_long	long	int
c_ulong	unsigned long	int
c_longlong	__int64 or long long	int
c_ulonglong	unsigned __int64 or unsigned long long	int

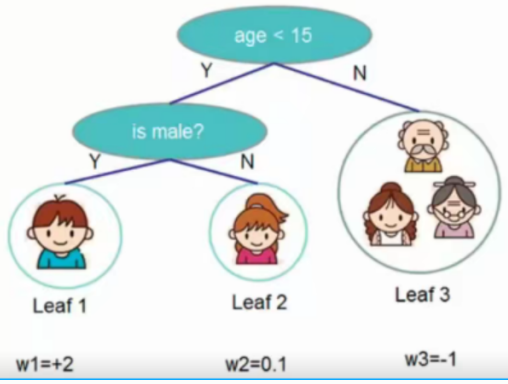
c_size_t	size_t	int
c_ssize_t	ssize_t or Py_ssize_t	int
c_float	float	float
c_double	double	float
c_longdouble	long double	float
c_char_p	char * (NUL terminated)	bytes object or None
c_wchar_p	wchar_t * (NUL terminated)	string or None
c_void_p	void *	int or None

- c中的字符都是直接转为python的int
- 32位4个字节，32位
- 64位8个字节
- 

- xgboost需要每加一个树都进行提升

- $\Omega(f_t) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$

  叶子的个数　　w的L2模平方



$\Omega = \gamma 3 + \frac{1}{2}\lambda(4 + 0.01 + 1)$

- 现在还剩下一个问题，我们如何选择每一轮加入什么f呢？答案是非常直接的，选取一个f来使得我们的目标函数尽量最大地降低

$$Obj^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t)}) + \sum_{i=1}^{t} \Omega(f_i)$$
$$= \sum_{i=1}^{n} l\left(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) + constant$$

目标：找到 f_t 来优化这一目标

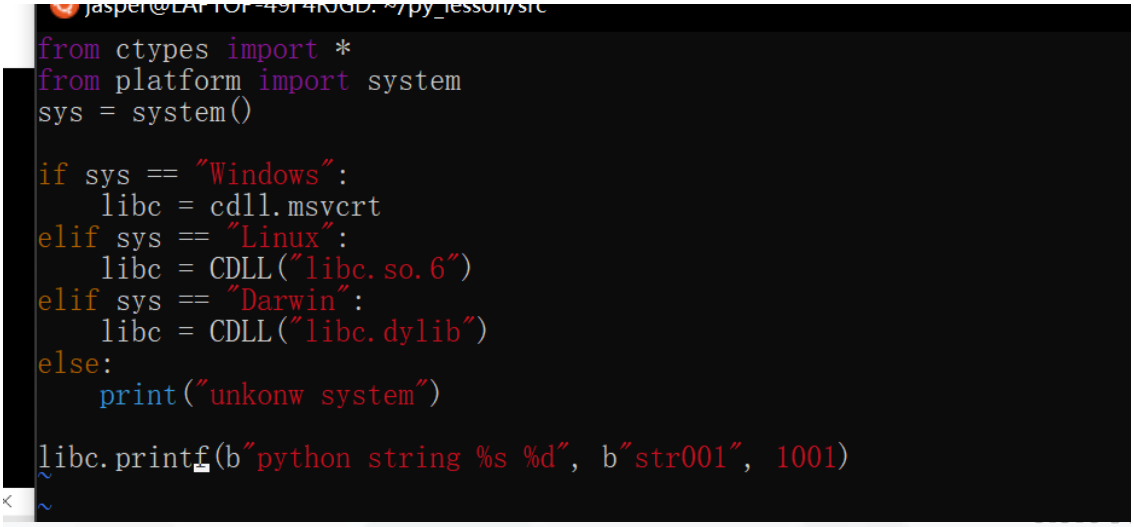
- # 跨平台调用c语言库函数

- libc = cdll.msvcrt
- libc = CDLL ('libc.so.6')
- libc = CDLL(“libc.dylib”)
- libc.printf (b “hello world”)

- python判断操作系统

- # Python判断操作系统

- import platform
- platform.system()
- Darwin Linux Windows

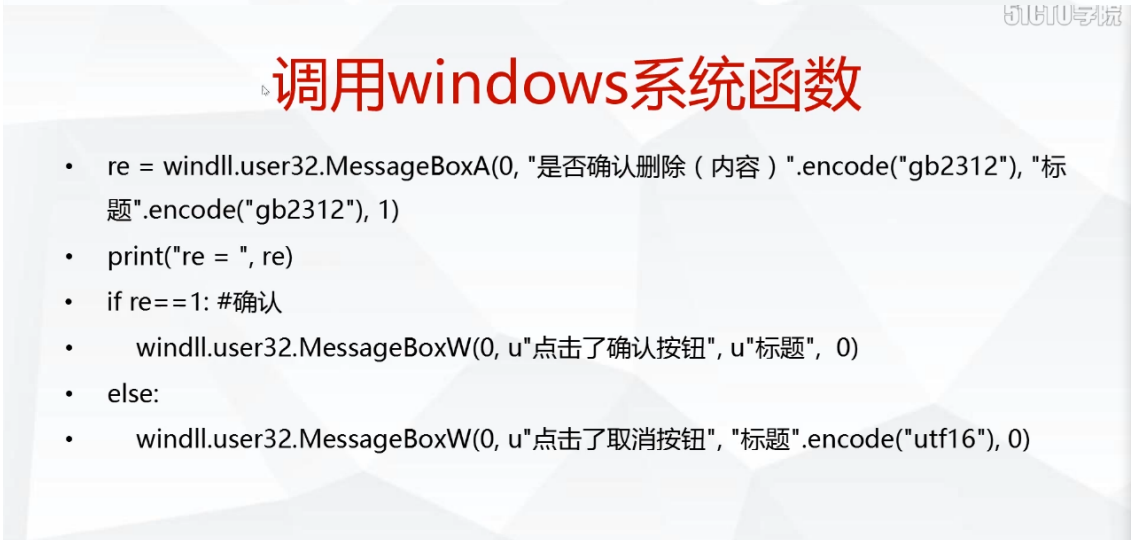
- 

```
from ctypes import *
from platform import system
sys = system()

if sys == "Windows":
    libc = cdll.msvcrt
elif sys == "Linux":
    libc = CDLL("libc.so.6")
elif sys == "Darwin":
    libc = CDLL("libc.dylib")
else:
    print("unkonw system")

libc.printf(b"python string %s %d", b"str001", 1001)
```

- 

WinDLL 调用_stdcall

- _stdcall 参数传递顺序和由谁清理 WINAPI Pascal
- __cdecl C语言默认的函数
- __declspec(dllexport) void _stdcall TestStdCall(int a)
- wdll = WinDLL("testctype")

- 

调用windows系统函数

- re = windll.user32.MessageBoxA(0, "是否确认删除（内容）".encode("gb2312"), "标题".encode("gb2312"), 1)
- print("re = ", re)
- if re==1: #确认
- windll.user32.MessageBoxW(0, u"点击了确认按钮", u"标题", 0)
- else:
- windll.user32.MessageBoxW(0, u"点击了取消按钮", "标题".encode("utf16"), 0)

- 

```
print("Test Win API")
from ctypes import *
# 0是普通窗口，1是需要确认
re = windll.user32.MessageBoxA(0, "窗口内容".encode("gbk"), "请选择".encode("gbk"), 1)
if re == 1: # 确认
    windll.user32.MessageBoxW(0, "点击了确认按钮".encode("gbk"), "已选择".encode("gbk"), 0)
else:
    windll.user32.MessageBoxW(0, "点击了取消按钮", "已选择", 0)
```

- Ctypes获取返回值
- 默认返回int

```python
from ctypes import *

lib = CDLL("./testctypes.dll")
# 设定返回值类型
# int 是默认参数
print("TestReturnInt = ", lib.TestReturnInt())

# return 返回的直接是byte
lib.TestReturnChar.restype = c_char_p
re = lib.TestReturnChar()
print(type(re))
print("TestReturnChar: ", lib.TestReturnChar())

# return直接返回的是string
lib.TestReturnWChar.restype = c_wchar_p
rew = lib.TestReturnWChar()
print(type(rew))
print("TestReturnWChar: ", lib.TestReturnWChar())

input()
```

- 只有括号不一定表示元组，需要加逗号

Ctypes传递和返回指针

- lib.Function.argtypes = (POINTER(c_float),)
- lib.Function.restype= (POINTER(c_void_p),)

- 指针一定指向的是一块空间，必须知道谁申请，谁释放，谁调用

- pointer 返回实例
- POINTER 返回类型

- byref(x [, offset])

Ctypes传递数组

- 通过指针的方式来传递

- (c_int*10) (1,2,3,4,5,6,7,8,9,10)

- a=[1,2,3,4,5,6,7,8,9,10]
- (c_int*10) (*a)

- TenArr = c_int*10
- a = TenArr(*a)

Ctypes传递和返回结构体

- class Pos(Structure):
  fields = [("x", c_int),("y", c_int)]

Ctypes传递回调函数

Ctypes传递回调函数

- CFUNCTYPE（返回值类型，参数类型。。。。）
- CMPFUNC = CFUNCTYPE(c_int, c_int, POINTER(c_int))
- def py_cmp_func(a, b):
- print("py_cmp_func", a, b)
  return 0 ;
- cmp_func = CMPFUNC(py_cmp_func)

qsort 快速排序

- void qsort(void *base, size_t nitems, size_t size, int (*compar)(const void *, const void*))

python调用鬼火（irrlicht）三维引擎

- 环境准备
  - python3.7.0
  - vs2015
  - irrlicht 1.8
  - 三维场景和动画人物
- python 与 c++交互的方式
  - ctypes
  - 扩展库