法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

- 课程详情请咨询
  - 微信公众号：大数据分析挖掘
  - 新浪微博：ChinaHadoop

第二讲

数据采集与操作

--梁斌

目录

- 常用格式的本地数据读写
- Python的数据库基本操作
- 数据库多表连接
- 爬虫简介
- BeautifulSoup解析网页
- 爬虫框架Scrapy基础
- Logistic 回归
- 实战案例：获取国内城市空气质量指数数据

目录

常用格式的本地数据读写

常用的数据分析文件格式

- txt
- csv
- json
- xml
- xls, xlsx
- HDF
- 其他可以转换成以上格式的数据文件
  - 如GIS中的.dbf可以导出成.csv文件

常用格式的本地数据读写

txt

示例代码： 01_txt_file_process.ipynb

- 由字符串行组成，每行由EOL (End Of Line) 字符隔开， ‘\n’
- 打开文件 注意编码
  - file_obj = open(filename, access_mode)
  - access_mode: ‘r’ , ‘w’
- 读操作
  - file_obj.read() 读取整个文件内容
  - file_obj.readline() 逐行读取
  - file_obj.readlines() 返回列表，列表中的每个元素是行内容
- 写操作
  - file_obj.write() 将内容写入文件
  - file_obj.writelines() 将字符串列表内容逐行写入文件

常用格式的本地数据读写

txt (续)

示例代码： 01_txt_file_process.ipynb

- 关闭文件
  - `file_obj.close()`

`with` 语句

- 包括了异常处理，自动调用文件关闭操作，推荐使用
- 适用于对资源进行访问的场合，确保无论适用过程中是否发生异常都会执行“清理”操作，如文件关闭、线程的自动获取与释放等
-
```
with open(filename) as f_obj:
    # 执行相关操作
```

常用格式的本地数据读写

CSV (Comma-Separated Values)

- 以纯文本形式存储的表格数据（以逗号作为分隔符），通常第一行为列名
- 文件操作
  - numpy 的 np.loadtxt(), 较复杂
  - 利用pandas处理，快捷方便
- 读操作
  - df_obj = pd.read_csv()，返回DataFrame类型的数据
- 写操作
  - df_obj.to_csv()

示例代码： 02_csv_file_process.ipynb

常用格式的本地数据读写

Pandas

- 基于NumPy构建
- 索引在左，数值在右。索引是pandas自动创建的。
- 数据结构
  - Series，类似于一维数组的对象。
  - DataFrame，表格型数据结构，每列可以是不同的数据类型，可表示二维或更高维的数据

示例代码： 02_csv_file_process.ipynb

常用格式的本地数据读写

JSON (JavaScript Object Notation)

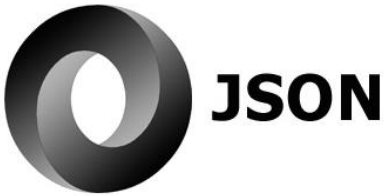
- 轻量级的数据交换格式
- 语法规则
  - 数据是键值对
  - 由逗号分隔
  - {}保存对象，如{key1:val1, key2,:val2}
  - []保存数组，如[val1, val2, …, valn]

示例代码： 03_json_file_process.ipynb

常用格式的本地数据读写

JSON (JavaScript Object Notation) (续)

- 读操作
  - `json.load(file_obj)`
  - 返回值是dict类型
- 类型转换 `json -> csv`
- 编码操作
  - `json.dumps()`
  - 编码注意
    - `ensure_ascii=False`



示例代码： `03_json_file_process.ipynb`

常用格式的本地数据读写

XLS/XLSX (Excel文件)

- 常用的电子表格数据
- 文件操作
  - 利用pandas处理，快捷方便
- 读操作
  - `df_obj = pd.read_excel()`，返回DataFrame类型的数据
- 写操作
  - `df_obj.to_excel()`

- 具体操作参考pandas如何处理CSV文件

目录

Python的数据库基本操作

SQLite

- 关系型数据库管理系统
- 嵌入式数据库，适用于嵌入式设备
- SQLite不是C/S的数据库引擎
- 集成在用户程序中
- 实现了大多数SQL标准

示例代码： 04_sqlite_basic.ipynb

Python的数据库基本操作

SQLite

- 连接数据库
  - conn = sqlite3.connect(db_name)
    - 如果db_name存在，读取数据库
    - 如果db_name不存在，新建数据库
- 获取游标
  - conn.cursor()
  - 一段私有的SQL工作区，用于暂时存放受SQL语句影响的数据

示例代码： 04_sqlite_basic.ipynb

Python的数据库基本操作

SQLite (续)

- CRUD操作
  - cursor.execute(sql_str)
  - cursor.executemany(sql_str) 批量操作
- fetchone()
- fetchall()
- conn.commit()，提交操作
- 关闭连接
  - conn.close()

示例代码： 04_sqlite_basic.ipynb

Python的数据库基本操作

其他常用数据库的连接

- Mysql
  - 主要面对互联网用户，比如建站等。
  - https://dev.mysql.com/doc/connector-python/en/
- PostgreSQL
  - Django推荐与PostgreSQL配合使用
  - Psycopg
  - http://initd.org/psycopg/docs/
- MongoDB
  - 分布式数据库
  - https://docs.mongodb.com/getting-started/python/client/

Python的数据库基本操作

其他常用数据库的连接

- Oracle
  - 适用于各类大、中、小、微机环境。它是一种高效率、可靠性好的 适应高吞吐量的数据库解决方案
  - http://www.oracle.com/technetwork/articles/dsl/python-091105.html

目录

数据库多表连接用法详解

多表连接

- 查询记录时将多个表中的记录连接(join)并返回结果
- join方式
  - 交叉连接 (cross join)
  - 内连接 (inner join)
  - 外连接 (outer join)

- cross join
  - 生成两张表的笛卡尔积
  - 返回的记录数为两张表的记录数的乘积

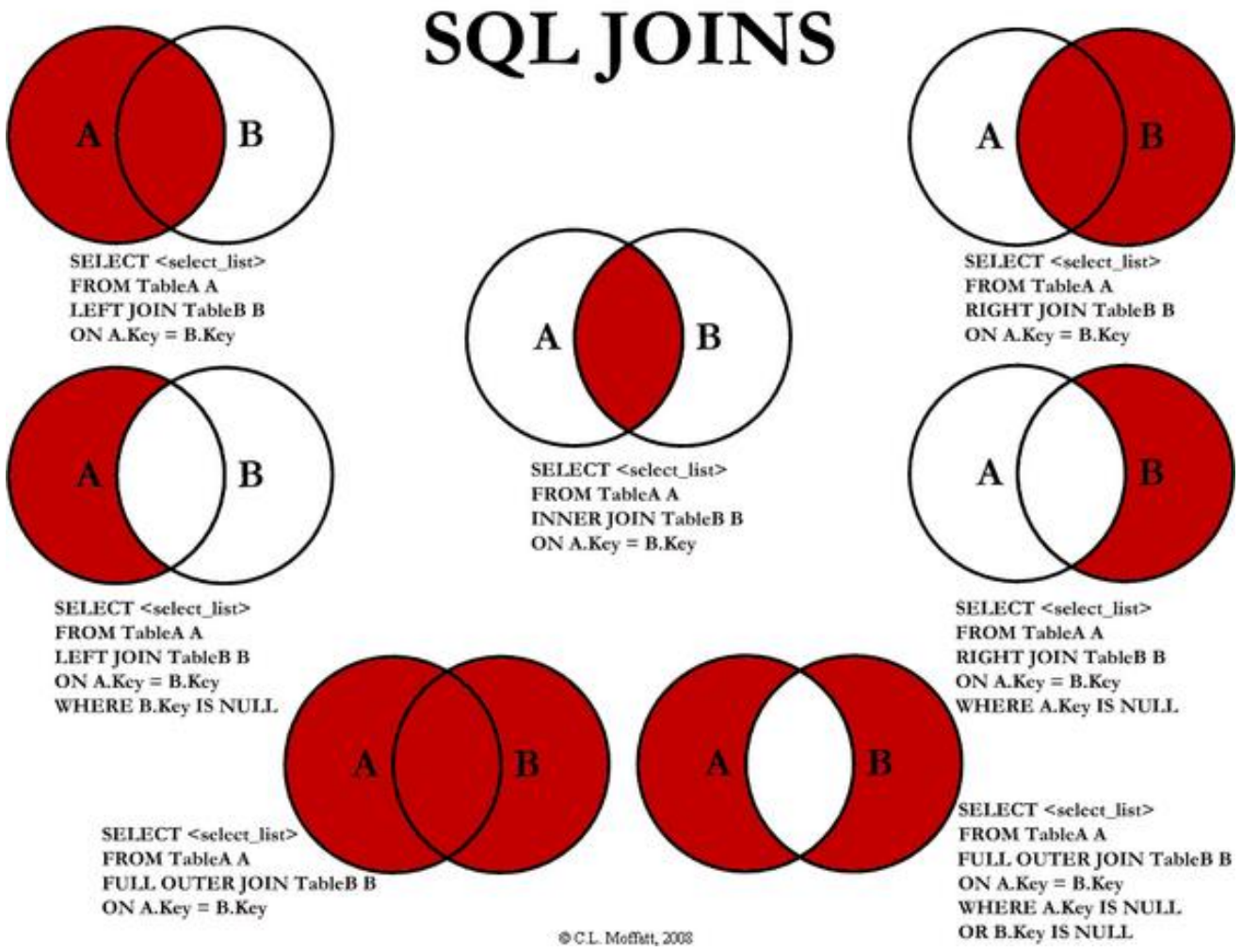
示例代码： 05_sqlite_join.ipynb

数据库多表连接用法详解

多表连接 (续)

- inner join
  - 生成两张表的交集
  - 返回的记录数为两张表的交集的记录数
- outer join
  - left join (A,B)，返回表A的所有记录，另外表B中匹配的记录有值，没有匹配的记录返回null
  - right join (A,B)，返回表B的所有记录，另外表A中匹配的记录有值，没有匹配的记录返回null
    - [注]目前在sqlite3中不支持，可考虑交换A、B表操作

数据库多表连接用法详解

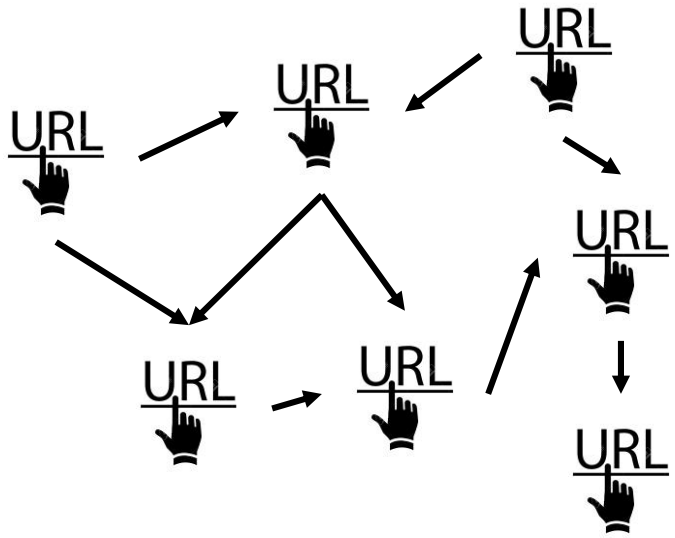


目录

爬虫简介

爬虫

- 自动抓取互联网信息的程序
- 利用互联网数据进行分析、开发产品



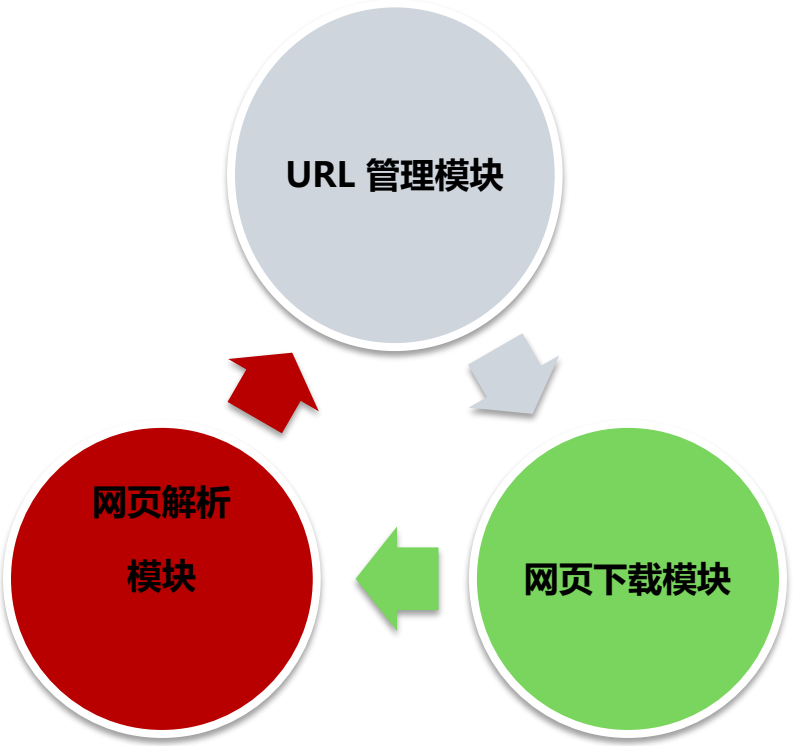
爬虫简介

爬虫基本架构

- URL 管理模块
  - 对计划爬取的或已经爬取的URL进行管理
- 网页下载模块
  - 将URL管理模块中指定的URL进行访问下载
- 网页解析模块
  - 解析网页下载模块中的URL，处理或保存数据
  - 如果解析到要继续爬取的URL，返回URL管理模块继续循环

爬虫简介

爬虫基本架构



爬虫简介

URL管理模块

- 防止重复爬取或循环指向
- 实现方式
  - Python的set数据结构，原因？
  - 数据库中的数据表，how?
  - 缓存数据库Redis，适用于大型互联网公司

爬虫简介

URL下载模块

- 将URL对应的网页下载到本地或读入内存(字符串)
- 实现方式
  - urllib，Python官方基础模块
  - requests或其他第三方的模块
- 通过URL直接下载

```
response = urllib.request.urlopen(url)
response.getcode()
response.read()
```

示例代码： 06_crawl_basic.ipynb

爬虫简介

URL下载模块 (续)

- 通过Request访问下载

```
request = urllib.request.Request(url)
request.add_head()
request.add_data()
response = urllib.urlopen(request)
```

示例代码： 06_crawl_basic.ipynb

爬虫简介

URL下载模块 (续)

- 通过Cookie访问下载
- 使用http.cookiejar模块
- cookie_jar = http.cookiejar.CookieJar()

  opener = urllib.request.build_opener()

  urllib.request.install_opener(opener)

  response = urllib.request.urlopen(url)

示例代码： 06_crawl_basic.ipynb

爬虫简介

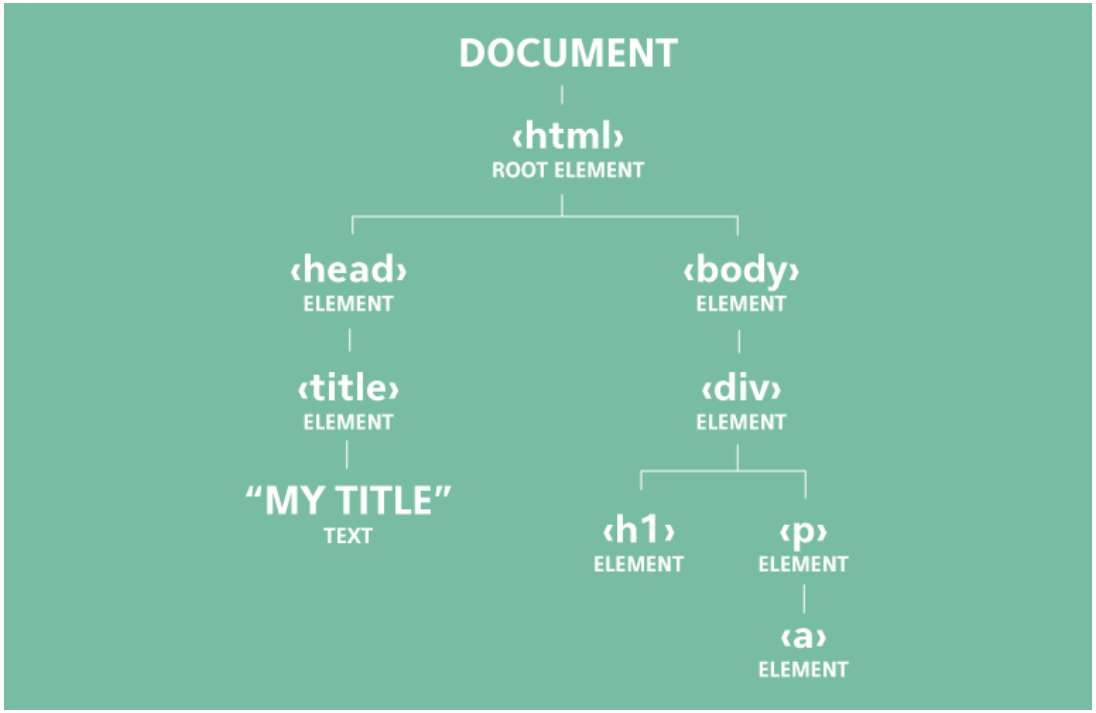
网页解析模块

- 从已下载的网页中爬取数据
- 实现方式
  - 正则表达式，字符串的模糊匹配
  - html.parser
  - BeautifulSoup，结构化的网页解析
  - lxml

爬虫简介

网页解析模块 (续)

- 结构化解析
- DOM (Document Object Model)，树形结构



目录

BeautifulSoup解析网页

BeautifulSoup

- 用于解析HTML或XML
- pip install beautifulsoup4
- import bs4
- 步骤

  1. 创建BeautifulSoup对象
  2. 查询节点

     find，找到第一个满足条件的节点

     find_all, 找到所有满足条件的节点

BeautifulSoup解析网页

创建对象

- 创建BeautifulSoup对象
- bs = BeautifulSoup(

    url,

    html_parser, 指定解析器

    enoding 指定编码格式（确保和网页编码格式一致）

)

示例代码： 07_bs4_basic.ipynb

BeautifulSoup解析网页

查找节点

- next page
- 可按节点类型、属性或内容访问
- 按类型查找节点
  - bs.find_all('a')
- 按属性查找节点
  - bs.find_all('a', href='a.html')
  - bs.find_all('a', href='a.html', string='next page')
  - bs.find_all('a', class_='a_link')
    - 注意：是class_

示例代码： 07_bs4_basic.ipynb

BeautifulSoup解析网页

获取节点信息

- node是已查找到的节点
- node.name
  - 获取节点标签名称
- node['href']
  - 获取节点href属性
- node.get_text()
  - 获取节点文字

异常处理

- 网络资源或URL是经常变动的
- 需要处理异常

示例代码： 07_bs4_basic.ipynb

BeautifulSoup解析网页

BeautifulSoup 进阶

- 使用CSS方式、正则表达式查找节点
- 保存解析的内容
- DOM树形结构
  - children 只返回“孩子”节点
  - desecdants 返回所有“子孙”节点
  - next_siblings 返回下一个“同辈”节点
  - previous_siblings 返回上一个“同辈”节点
  - parent 返回“父亲”节点

示例代码： 08_bs4_advanced.ipynb

BeautifulSoup解析网页

BeautifulSoup 进阶（续）

- 正则表达式
- 简单的字符串匹配可以使用字符串方法完成
- 复杂、模糊的字符串匹配使用正则表达式
  - 如：电子邮箱格式匹配
- 通过使用单个字符串描述匹配一系列符合某个语法规则的字符串
- 字符串操作的逻辑公式
- 常用于处理文本数据
- 匹配过程：依次拿出表达式和文本中的字符作比较，如果每个字符都能匹配，则匹配成功；否则失败

示例代码： 08_bs4_advanced.ipynb

BeautifulSoup解析网页

BeautifulSoup 进阶 (续)

- 正则表达式
- import re
- pattern = re.compile(‘str’) 返回pattern对象
  - 推荐使用 r’str’ 无需考虑转义字符
- pattern.match()
- 基本语法
  - https://msdn.microsoft.com/zh-cn/library/ae5bf541(v=vs.90).aspx

示例代码： 08_bs4_advanced.ipynb

目录

爬虫框架Scrapy基础

Scrapy简介

- 开源的爬虫框架
- 快速强大，只需编写少量代码即可完成爬取任务
- 易扩展，添加新的功能模块
- 用户群
  - https://scrapy.org/companies/

爬虫框架Scrapy基础

Scrapy抓取过程

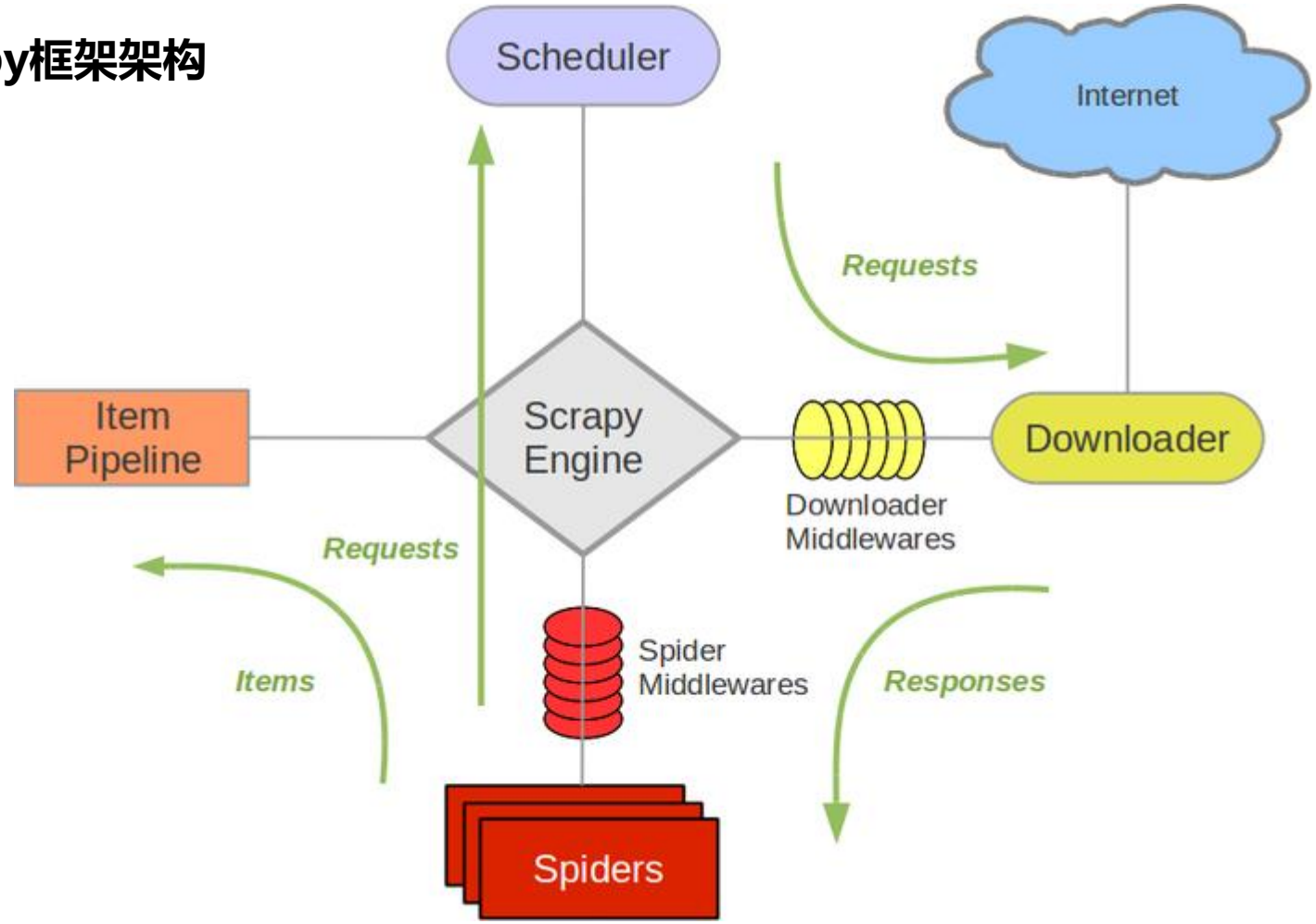
- 使用start_urls作为初始url生成Request，默认将parse作为他的回调函数
- 在parse函数中解析目标url

Scrapy高级特性

- 内置数据抽取器css/xpath/re
- 交互式控制台用于调试
- 结果输出的格式支持，JSON，CSV，XML等
- 自动处理编码
- 支持自定义扩展

爬虫框架Scrapy基础

Scrapy框架架构

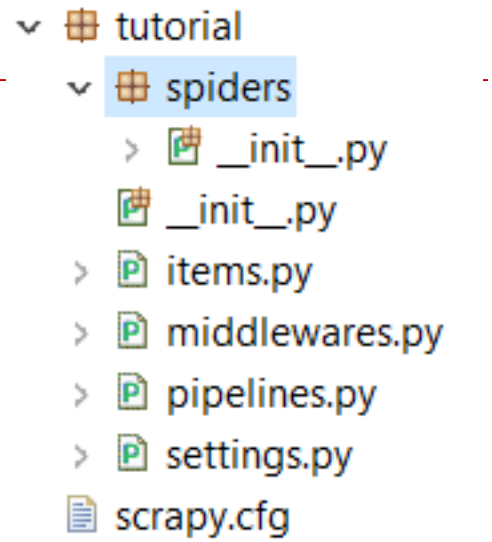


爬虫框架Scrapy基础

Scrapy使用步骤

- 安装：pip install scrapy (可能需要额外安装visual c++ build tools)
- 1．创建工程
- 2．定义Item，构造爬取的对象（可选）
- 3．编写Spider，爬虫主体
- 4．编写配置和Pipeline，用于处理爬取的结果（可选）
- 5．执行爬虫

爬虫框架Scrapy基础



Scrapy使用步骤

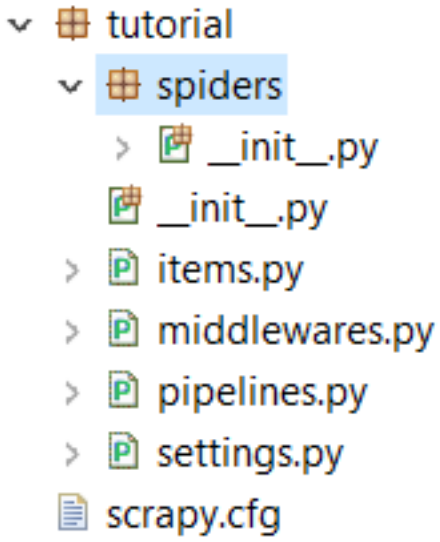
1. 创建工程
   - scrapy startproject air_quality
   - 目录结构
3. 编写Spider
   - scrapy genspider aqi_history_spider https://www.aqistudy.cn/historydata/index.php
5. 运行Spider
   - scrapy crawl aqi_history_spider

示例代码：lect02_proj.zip

爬虫框架Scrapy基础

Scrapy使用步骤

2. 定义Item
   - scrapy.Field()
3. 编写Spider
   - 调用自定义的Item
4. pipelines
   - 默认return item
5. 运行Spider
   - scrapy crawl aqi_history_spider



示例代码：lect02_proj.zip

爬虫框架Scrapy基础

Scrapy常用命令

- help: 查看帮助，scrapy --help
- version: 查看版本信息，
  - scrapy version，查看scrapy版本
  - scrapy version –v，查看相关模块的版本
- startproject， 新建工程，scrapy startproject porj_name
- genspider，生成spider模板，scrapy genspider spider_name url

爬虫框架Scrapy基础

Scrapy常用命令 (续)

- list，列出所有的spider，scrapy list
- view，返回网页源代码并在浏览器中打开，scrapy view url
  - 有时页面渲染的结果和查看结果是不同的
- parse，调用工程spider中的parse解析url，scrapy parse url
- shell，进入交互式调试模式，scrapy shell url
- bench，可以用来检测scrapy是否安装成功
- …

目录

Logistic 回归

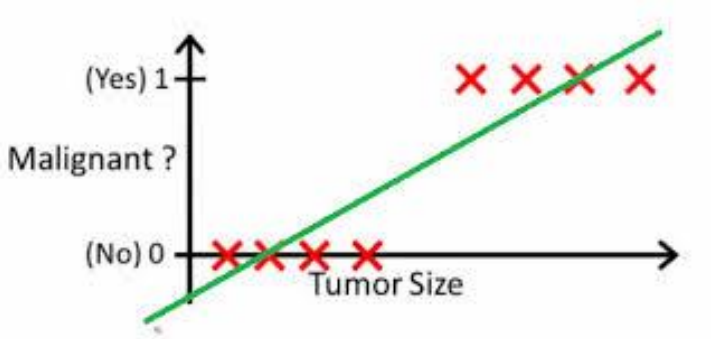
概率(probability)

- 定义：对一件事情发生可能性的衡量
- 范围：0<=p<=1
- 条件概率

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

Logistic Regression（逻辑回归）

- 例子

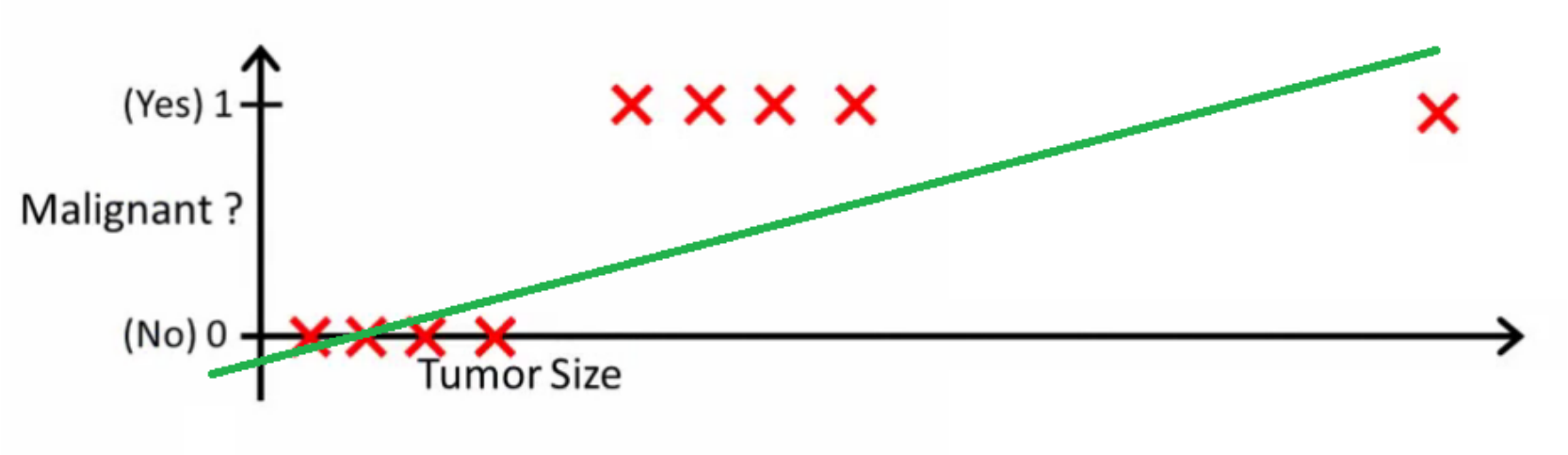


h(x) > 0.5 or h(x) < 0.5

Logistic 回归

Logistic Regression（逻辑回归）（续）

- 例子



$h(x) > 0.2$ or $h(x) < 0.2$

Logistic 回归

基本模型

- 训练样本为： X（x_0，x_1，x_2，…，x_n）
- 学习的参数为：θ（θ_0，θ_1，θ_2，…，θ_n）

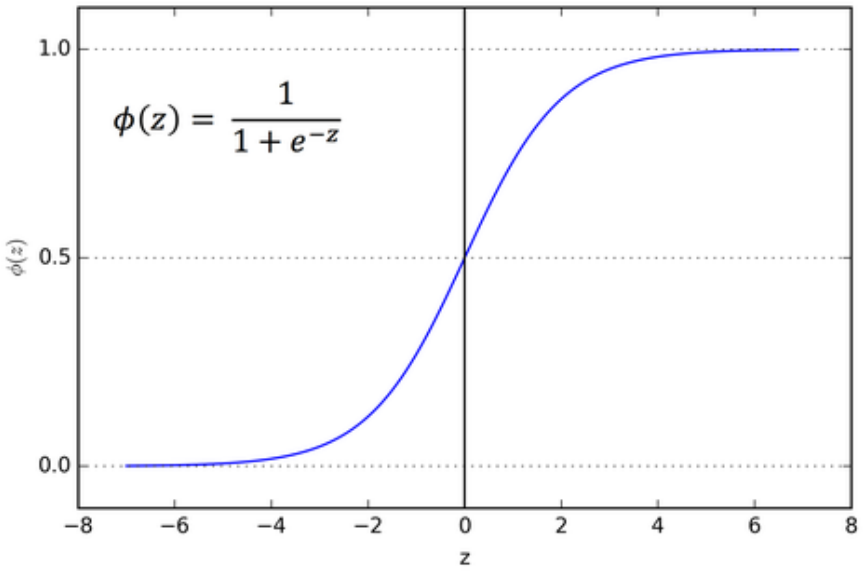
$$Z = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

- 向量表示

$$Z = \Theta^T X$$

- Sigmoid函数将线型转换成非线性

$$g(Z) = \frac{1}{1+e^{-Z}}$$



Logistic 回归

基本模型（续）

- 预测函数 $$h_\theta(X) = g(\Theta^T X) = \frac{1}{1+e^{-\Theta^T X}}$$
- 用概率的形式表示
  - 正样本 $$h_\theta(X) = P(y=1 \mid X;\Theta)$$
  - 负样本 $$1-h_\theta(X) = P(y=0 \mid X;\Theta)$$
- 损失函数

$$Cost(h_\Theta(X),y) = \begin{cases} -\log(h_\Theta(X)) \_ \text{when\_y=1} \\ -\log(1-h_\Theta(X)) \_ when \_ y=0 \end{cases}$$

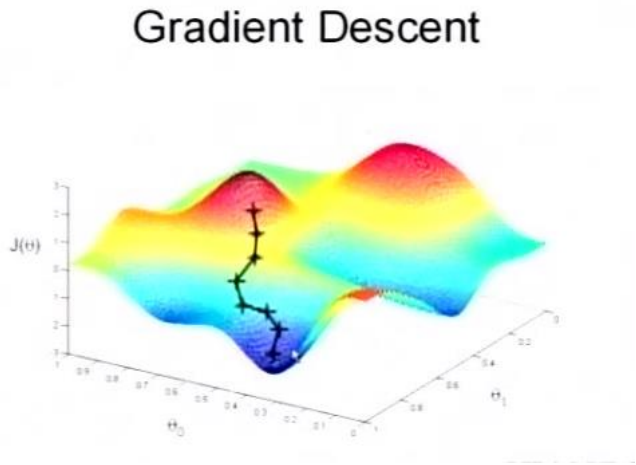
$$J(\Theta) = \frac{1}{m}\sum_{i=1}^{m} Cost(h_\Theta(x^{(i)}), y^{(i)}) = -\frac{1}{m}[\sum_{i=1}^{m}(y^{(i)}\log(h_\Theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\Theta(x^{(i)})))]$$

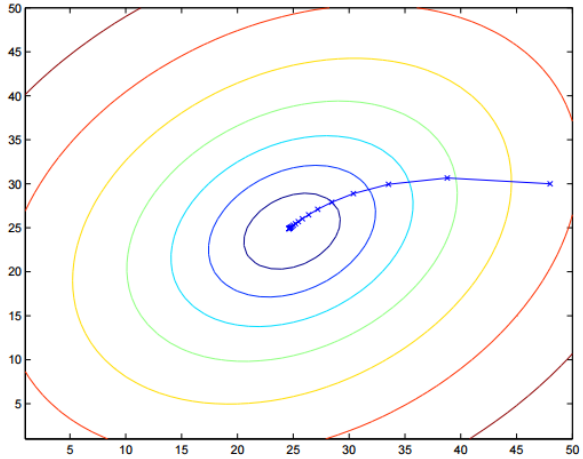
- 目标：通过训练样本求出参数theta使损失函数最小化

Logistic 回归

基本模型（续）

- 解法：梯度下降（gradient descent）





$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta), (j = 0 \ldots n)$$

更新方式：

$$\theta_j = \theta_j - \alpha \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) x_j^{(i)}, (j = 0 \ldots n)$$

- alpha：学习率
- 同时更新所有theta
- 迭代更新直到收敛

目录

实战案例

项目介绍

- 爬取中国各城市历史空气质量记录
- https://www.aqistudy.cn/historydata/

- **项目任务**

1．掌握Scrapy框架的基本操作

2．能够爬取简单的页面数据

3．掌握深度数据的爬取及广度数据的爬取

示例代码： lect02_proj_scrapy

实战案例

涉及知识点

- Python面向对象编程
- Scrapy框架
- xpath
- 数据保存
  - CSV
  - JSON
  - XML

示例代码： lect02_proj_scrapy

参考

- Python的文件读写

  https://docs.python.org/2/tutorial/inputoutput.html
- Pandas的IO工具

  http://pandas.pydata.org/pandas-docs/stable/io.html
- SQLite中的多表连接

  https://www.tutorialspoint.com/sqlite/sqlite_using_joins.htm
- sqlite3模块

  https://docs.python.org/2/library/sqlite3.html
- Python的中文编码

  http://blog.csdn.net/liuxincumt/article/details/8183391

参考

- BeautifulSoup

  https://www.crummy.com/software/BeautifulSoup/bs4/doc/
- 正则表达式

  http://www.regexlab.com/zh/regref.htm
- Scrapy

  https://scrapy.org/
- Xpath教程

  http://www.w3school.com.cn/xpath/
- Scrapy命令行

  https://doc.scrapy.org/en/latest/topics/commands.html

参考

- 验证码识别

  https://code.google.com/archive/p/pytesser/

- Scrapy中的IP代理

  https://doc.scrapy.org/en/latest/topics/downloader-middleware.html#scrapy.downloadermiddlewares.httpproxy.HttpProxyMiddleware

疑问

- 问题答疑： http://www.xxwenda.com/
  - 可邀请老师或者其他人回答问题

小象问答 @Robin_TY

联系我们

小象学院：互联网新技术在线教育领航者

- 微信公众号：小象
- 新浪微博：ChinaHadoop