



南京财经大学  
NANJING UNIVERSITY OF FINANCE & ECONOMICS

## 第2章：数据采集

# 电子商务导论

朱桂祥 (9120201070@nufe.edu.cn)

南京财经大学信息工程学院

江苏省电子商务重点实验室

电子商务信息处理国家级国际联合研究中心

电子商务交易技术国家地方联合工程实验室



### 为什么要有数据采集和预处理？

- (1)数据是大数据分析的原材料，但原始数据(raw data)往往不能直接使用。面临着不~~规范~~，不~~完整~~的问题。
- (2)数据~~采集~~可以解决数据不~~完整~~的问题，而~~预处理~~可以解决不~~规范~~的问题。
- (3)数据采集，预处理是数据分析的前期工作，并在整个大数据分析中起到非常重要的基础作用。如果数据出错了，后面的分析工作往往受重要影响。~~浮沙之上，难建高楼~~。
- (4)数据~~采集~~面临着6V中的~~规模性~~(Volume)，~~多样性~~(Variety)，~~高速性~~(Velocity)，~~价值高~~(Value)等诸多挑战。

~~预处理~~面临着大数据6V中的Veracity(真实性)的问题。



### 数据从哪里来？

#### (1) 系统的内部数据

系统内部的数据库，各种文档，图片，音频和视频。  
系统内部的业务数据，人员数据，日程事务数据等。

#### (2) 系统的外部数据

政府公开的数据，竞争对手的情报数据，社交网站的舆情数据，与业务相关的外部支撑数据，聘用新员工所需的人力资源数据等。

#### (3) 大数据分析需要解决数据外部性的痛点。

只有内外互补，才可能解决问题。



### 怎样采集数据？

- (1) 对于结构化的日常事务数据，在线表单，在线调查，线下问卷调查，存储在数据库。
- (2) 对于用户消费者行为数据，网络日志采集。
- (3) 互联网上面海量的公开数据，网络爬虫
- (4) 物联网上的海量数据。各种设备感知，存储。
- (5) 第三方数据库。



### 怎样采集数据？

(1) 对于结构化的日常事务数据，在线表单，在线调查，线下问卷调查，存储在数据库。一个大学就业调查的例子[1]。

#### 当代大学生就业意向的调查分析问卷



据了解，本次调查的目的在于分析、研究大学生就业发展中存在的问题及企业对大学生素质的要求，找出人才培养与人才需求的分歧，让大学生认清现状，有针对性地加强对自身的培养。

\* 1. 您的性别：

- ☐ 男  
☐ 女

\* 2. 请问你是大几的学生

- ☐ 大一  
☐ 大二  
☐ 大三  
☐ 大四  
☐ 已毕业

\* 3. 大学的规划

- ☐ 很明确  
☐ 一般般  
☐ 没规划

\* 17. 毕业前能做到那些准备 【多选题】

- ☐ 计算机二级证  
☐ 英语四级证  
☐ 英语六级证  
☒ 初级会计资格证  
☐ 教师资格证  
☐ 心理师  
☐ 其他

18. 你对未来的就业有什么想法？

提交



### 怎样采集数据？

(2) 对于用户消费者行为数据，网络日志采集。

#### 2.1 浏览器数据采集(BS模式(Browser Server))

通过html5, javascript用于收集用户通过上网泄漏的各种信息，包括地理位置，IP地址，照片，语音，浏览器版本等信息。结合大数据，可实现广告定向投放，用户追踪，用户行为分析，用户群体调研等一系列更人性化的服务。

#### 2.2 户端数据采集(CS模式(Client Server))

Client/Server结构(C/S结构)是大家熟知的客户机和服务器结构。它是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到Client端和Server端来实现，降低了系统的通讯开销。

目前大多数应用软件系统都是Client/Server形式的两层结构，由于现在的软件应用系统正在向分布式的Web应用发展，Web和Client/Server



### 怎样采集数据？

#### (3) 互联网上面海量的公开数据，网络爬虫（八爪鱼[1]）



##### 云采集

5000台云服务器，24\*7高效稳定采集，结合API可无缝对接内部系统，定期同步爬取数据



##### 智能防封

自动破解多种验证码，模拟真实用户访问，提供全球最大代理IP池，结合UA切换，不怕防采集



##### 海量模板

内置400+网站数据爬虫模版，全面覆盖多个行业，只需简单设置，就可快速准确获取数据

[1]<http://www.bazhuayu.com/>





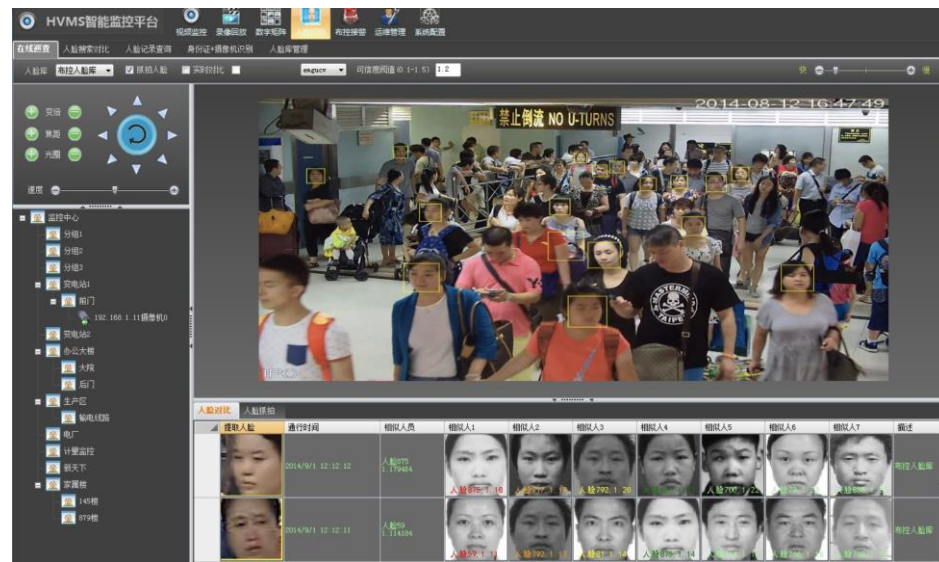
### 怎样采集数据？

(4) 物联网上的海量数据。各种感知，存储设备。

视频数据：高清防抖摄像头，获得关键信息(人脸，车牌等)。

语音数据：麦克风阵列，消除背景噪声。

传感器数据采集：智慧农业中的温度传感器，湿度传感器等。







### 怎样采集数据？

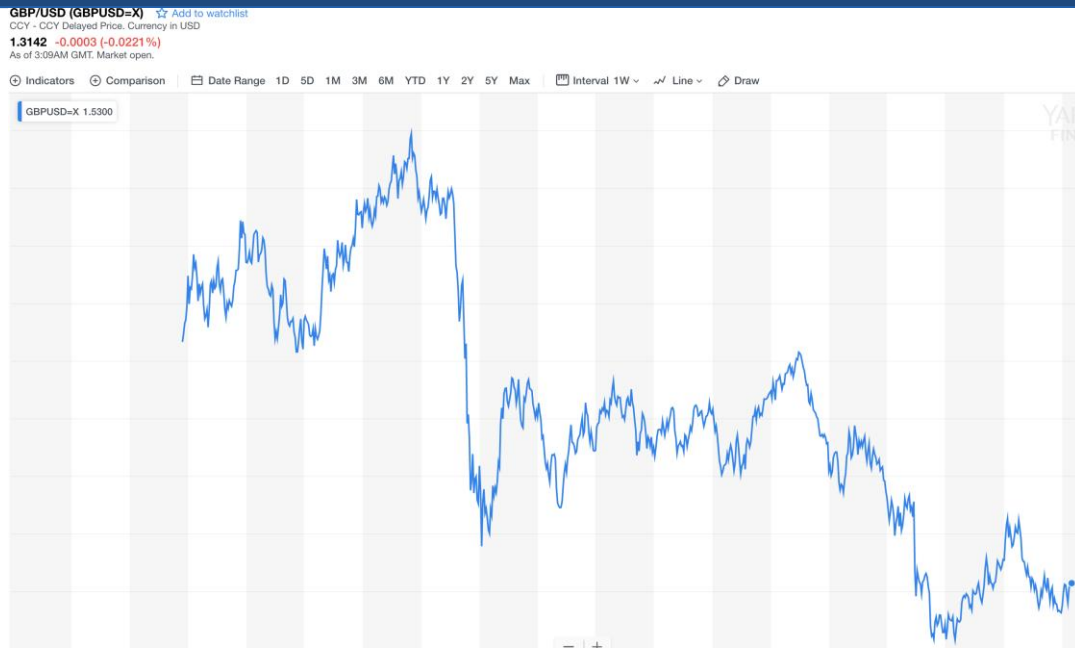
(4) 第三方数据库公开数据

金融数据：雅虎财经

(如右图，从2013年12月至今  
英镑和美元的汇率波动数据)

学术数据库：

知网，Elsevier，arXiv等。





### 网络爬虫

为什么要使用网络爬虫？

- (1) 网络爬虫，搜索引擎背后的基础技术。百度和谷歌搜索显示的页面，都源自于网络爬虫每天不停的工作。
- (2) 网络爬虫可以一次下载大量网页。
- (3) 网络爬虫和各种网站开放的API有什么不同？
- (4) 多源异构的互联网开放数据, 经过预处理，数据融合以后的有价值数据。



### 网络爬虫

网络爬虫 (Web Crawler) 的定义：



(1) 也叫网络蜘蛛 (Web Spider)，利用 HTTP 协议，根据超链接和 Web 文档检索的方法遍历 Web 空间的程序，是一种“自动化浏览网络”的程序，或者说是一种网络机器人[1]。

(2) A Web crawler, sometimes called a spider or spiderbot and often shortened to crawler, is an Internet bot that systematically browses the World Wide Web, typically for the purpose of Web indexing[2].

[1] 大数据分析，曹杰等编著

[2] 维基百科：[https://en.wikipedia.org/wiki/Web\\_crawler](https://en.wikipedia.org/wiki/Web_crawler)



### 网络爬虫

网络爬虫的分类：

- (1) 全网爬虫：搜集整个互联网的网页（百度，谷歌，搜狗等）
- (2) 主题网络爬虫：特定需求的爬虫，比如八爪鱼
- (3) 增量式网络爬虫：不抓取重复的数据，保证新数据和旧数据的唯一性。
- (4) 深层网络爬虫：深层网络爬虫对应深层网络数据。

表层网络数据：网页显示的内容。

深层网络数据：藏在网页背后的数据库的内容，并没有完全通过网页展示出来。或者是特定用户才有权限看到的内容。



### 网络爬虫

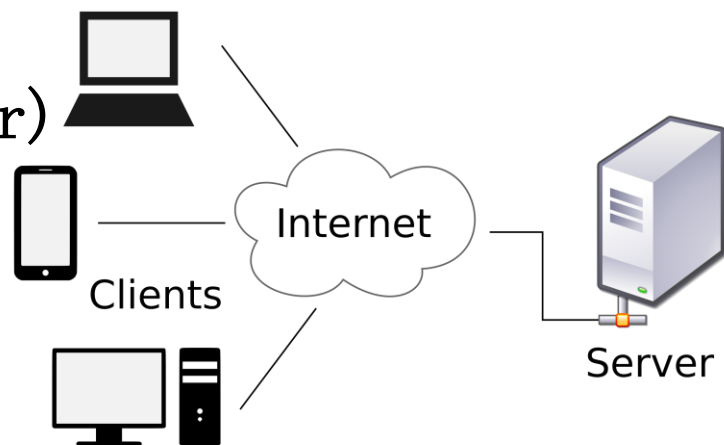
网络爬虫的预备基础知识：

客户端-服务器CS模式(Client - Server)

客户端向服务器发起**请求**(request),  
服务端向客户端给予**响应**(response)。

根据具体的业务不同，不同的请求  
和响应规则由不同的**协议**来定义。

包括HTTP(Hypertext Transfer Protocol)  
协议, HTTPS(Hypertext Transfer  
Protocol Secure)等等。





### 常用的网络爬虫工具：

- **Scrapy**: Scrapy 是基于python的一个库，为了抓取网页数据、提取结构性数据而编写的应用框架，该框架是封装的，包含 **request**（异步调度和处理）、下载器（多线程的 **Downloader**）、解析器（**selector**）和 **twisted**（异步处理）等。对于网站的内容爬取，其速度非常快捷。优点：通过管道的方式存入数据库，灵活，可保存为多种形式。缺点：无法用它完成分布式爬取。
- **PySpider**: 一个国人编写的强大的网络爬虫系统并带有强大的WebUI。采用Python语言编写，分布式架构，支持多种数据库后端，强大的WebUI支持脚本编辑器，任务监视器，项目管理器以及结果查看器。Python脚本控制，可以用任何你喜欢的html解析包。
- **Nutch**是为搜索引擎设计的爬虫，Nutch运行的一套流程里，有三分之二是为了搜索引擎而设计的。对精抽取没有太大的意义。也就是说，用Nutch做数据抽取，会浪费很多的时间在不必要的计算上。而且如果你试图通过对Nutch进行二次开发，来使得它适用于精抽取的业务，基本上就要破坏Nutch的框架，把Nutch改的面目全非。





### 常用的网络爬虫工具：

- **Crawler4j**: Crawler4j是一个开源的Java类库提供一个用于抓取Web页面的简单接口。可以利用它来构建一个多线程的Web爬虫。优点：支持多线程采集。缺点：不支持动态网页抓取，例如网页的ajax部分。
- **DotnetSpider**这是国人开源的一个跨平台、高性能、轻量级的爬虫软件，采用C# 开发。目前是.Net开源爬虫最为优秀的爬虫之一。
- **Larbin**: 是一种开源的网络爬虫/网络蜘蛛，由法国的年轻人 Sébastien Ailleret独立开发。larbin目的是能够跟踪页面的url进行扩展的抓取，最后为搜索引擎提供广泛的数据来源。Larbin只是一个爬虫，也就是说larbin只抓取网页，至于如何parse的事情则由用户自己完成。另外，如何存储到数据库以及建立索引的事情 larbin也不提供。







南京财经大学  
NANJING UNIVERSITY OF FINANCE & ECONOMICS

## 第2章：数据采集和预处理

### 网络爬虫的预备基础知识： HTTP 协议

HTTP协议基于CS模式，是一种请求响应的协议。通常请求由客户机上的Web浏览器发出，而服务器上的Web网站收到请求，给出响应。

右边是一个访问Wiki百科的例子。包括Request, Response Header, Response body三个部分。

```
josh@blackbox:~$ telnet en.wikipedia.org 80
Trying 208.80.152.2...
Connected to rr.pmtpa.wikimedia.org.
Escape character is '^]'.
GET /wiki/Main_Page http/1.1
Host: en.wikipedia.org

HTTP/1.0 200 OK
Date: Thu, 03 Jul 2008 11:12:06 GMT
Server: Apache
X-Powered-By: PHP/5.2.5
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: en
Vary: Accept-Encoding, Cookie
X-Vary-Options: Accept-Encoding;list-contains=gzip, Cookie;string-contains=enwikiToken;string-contains=enwikiLoggedOut;string-contains=enwiki_session;string-contains=centralauth_Token;string-contains=centralauth_Session;string-contains=centralauth_LoggedOut
Last-Modified: Thu, 03 Jul 2008 10:44:34 GMT
Content-Length: 54218
Content-Type: text/html; charset=utf-8
X-Cache: HIT from sq39.wikimedia.org
X-Cache-Lookup: HIT from sq39.wikimedia.org:3128
Age: 3
X-Cache: HIT from sq38.wikimedia.org
X-Cache-Lookup: HIT from sq38.wikimedia.org:80
Via: 1.0 sq39.wikimedia.org:3128 (squid/2.6.STABLE18), 1.0 sq38.wikimedia.org:80 (squid/2.6.STABLE18)
Connection: close

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en" dir="ltr">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <meta name="keywords" content="Main Page,1778,1844,1863,1938,1980 Summer Olympics,2008,2008 Guizhou riot,2008 Jerusal
    ...
    ... This content has been removed to save space
    ...
    "Non-profit organization">nonprofit</a> <a href="http://en.wikipedia.org/wiki/Charitable_organization" title="Charitable organization">charity</a>.<b
    r /></li>
    <li id="privacy"><a href="http://wikimediafoundation.org/wiki/Privacy_policy" title="wikimedia:Privacy policy">Privac
    y policy</a></li>
    <li id="about"><a href="/wiki/Wikipedia:About" title="Wikipedia:About">About Wikipedia</a></li>
    <li id="disclaimer"><a href="/wiki/Wikipedia:General_disclaimer" title="Wikipedia:General disclaimer">Disclaimers</a>
  </li>
  </ul>
</div>
</div>

    <script type="text/javascript">if (window.runOnLoadHook) runOnLoadHook();</script>
<!-- Served by srv93 in 0.050 secs. --></body></html>
Connection closed by foreign host.
josh@blackbox:~$
```

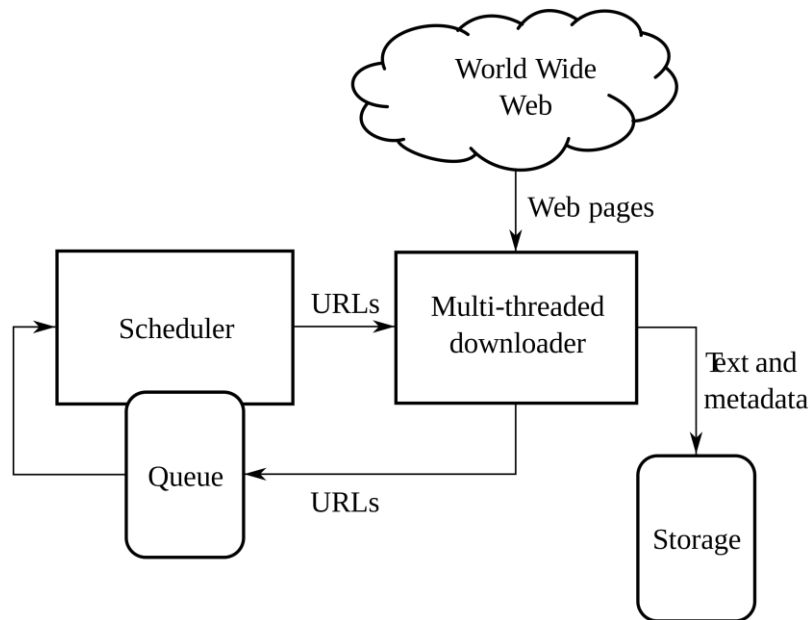


## 第2章：数据采集和预处理

### 网络爬虫

网络爬虫的技术原理（如右图所示）：

- (1) 调度器(Scheduler)输入种子URL
- (2) 下载器(Downloader)下载相关页面。
- (3) 下载器从下载的页面中，由挖网页解析器抽取关联的URLs, 放入调度器的队列Queue, 等待下一轮处理。
- (4) 下载器把下载得到的页面进行存储。
- (5) 调度器从增量的URLs开始新一轮任，重复步骤(1)-(4)。
- (6) 整个过程迭代，直到队列中的URLs列表为空，停止下载。





### 网络爬虫

网络爬虫实战的常用工具。

C/C++等语言一般用于百度等搜索引擎公司，用于设计通用的搜索引擎，但是由于实现比较复杂，不适合初学者。

Python相比较C/C++而言，具有简单易学，功能较全的特点。

Python的url和urllib：由URLs列表得到网页内容，

Re库：通过正则表达式解析下载网页中的URLs，并放到队列Queue中。

整个爬虫框架Scrapy = Scraach(抓取)+Python



网络爬虫：一个Scrapy的例子：抓取某网站的内容(1).

```
import scrapy

class QuotesSpider(scrapy.Spider):
    name = 'quotes'
    start_urls = [
        'http://quotes.toscrape.com/tag/humor/',
    ]

    def parse(self, response):
        for quote in response.css('div.quote'):
            yield {
                'text': quote.css('span.text::text').get(),
                'author': quote.xpath('span/small/text()').get(),
            }

        next_page = response.css('li.next a::attr("href")').get()
        if next_page is not None:
            yield response.follow(next_page, self.parse)
```

```
scrapy runspider quotes_spider.py -o quotes.json
```



网络爬虫：一个Scrapy的例子：抓取某网站的内容(2).

```
[{
  "author": "Jane Austen",
  "text": "\u201cThe person, be it gentleman or lady, who has not pleasure in a good novel, i
},
{
  "author": "Groucho Marx",
  "text": "\u201cOutside of a dog, a book is man's best friend. Inside of a dog it's too dar
},
{
  "author": "Steve Martin",
  "text": "\u201cA day without sunshine is like, you know, night.\u201d"
},
...]
```



Python通过pyodbc连接Sql Server数据库进行数据操作：

```
1 # -*- encoding:utf-8 -*-
2 import pyodbc
3 import sys
4 import csv
5 import datetime
6 import time
7 import re
8 import os
9 reload(sys)
10 sys.setdefaultencoding('utf-8')
11 #conn=pymssql.connect(host='192.168.0.184',user='sa',password='pwd',database='ShcemDW')
12 starttime=datetime.datetime.now()
13 #conn=pymssql.connect(host='.',database='TUNIU-BI198',charset="utf8")
14 conn = pyodbc.connect('DRIVER={SQL Server};SERVER=localhost;PORT=1433;DATABASE=TUNIU-BI198',charset="utf8")
15 cur=conn.cursor()
16 #file1='ID CtiyName Type=0 1 3 19.txt'
17 #file1='ticket type=19 ID City.csv'
18 file1='final City jwd.txt'
19 file1='new.txt'
20 Dir=os.path.abspath('')
21 #print Dir+'提取全球目的地经纬度/'+file1
22 #finl=open(Dir+'提取全球目的地经纬度/'+file1,'r')
23 finl=open(file1,'r')
24 content=finl.readlines()
25 for line in content:
26     line=line.strip('\n')
27     #line=line.encode('utf-8')
28     list1=line.split(' ')
29     CityName=list1[0].encode('utf-8')
30     wd=list1[1]
31     jd = list1[2]
32     #print CityName,wd,jd
33     #sql="insert into [Tuniu purchase prediction training data].[dbo].[Type=0 1 3 19 TypeID CityName] VALUES (" + str(TypeID) + "," + str(CityName) + ")"
34     sql = "insert into [Distinct City JWD made by 20180125] VALUES ('" + unicode(CityName) + "','" + str(wd) + "','" + str(jd) + "')"
35     print sql
36     cur.execute(sql)
37 finl.close()
38 conn.commit()
39 cur.close()
40 conn.close()
41 #[Tuniu purchase prediction training data].[dbo].[Type=0 1 3 19 TypeID CityName]
```



### 网络爬虫与反爬虫技术：

#### 为什么要反爬虫？

- (1) 爬虫消耗了大量的服务器响应资源，使得正常的响应变慢。
- (2) 爬虫会盗用一部分网站不想公开的数据和信息。

#### 反爬虫的主要技术有哪些？

- (1) 基于Headers反爬虫：浏览器访问服务器，在Headers中有User-Agent-Referer字段。爬虫可以模拟浏览器绕过此限制。
  - (2) 基于用户行为反爬虫：同一用户短时间大量访问某网站。
- 爬虫应对策略：使用代理IP或降低访问频率。

- (3) 动态页面反爬虫：模拟AJAX请求，或是模拟浏览器发送动态请求。
- (4) Cookie限制：Cookie检验。
- (5) 验证码限制：拖动某个图片，或输入某个字母进行手动验证。





网络爬虫的法律与道德约束：

**合理合法**地获得网上的数据

- (1) 未经授权，不得擅自将有版权的数据公布，供人下载。
- (2) 不得擅自下载，或者**暴力破解**数据。
- (3) 不得违规下载**涉密**数据。
- (4) 遵循robots协议，那些页面能够被抓取，那些页面不能被抓取。



网络爬虫的法律与道德约束：

**合理合法**地获得网上的数据

- (1) 未经授权，不得擅自将有版权的数据公布，供人下载。
- (2) 不得擅自下载，或者**暴力破解**数据。
- (3) 不得违规下载**涉密**数据。
- (4) 遵循robots协议，那些页面能够被抓取，那些页面不能被抓取。



电子商务数据的采集：

数据的来源及分类：

- (1) 电子商务数据平台的基础数据
- (2) 电商专业网站的研究数据
- (3) 基于电商媒体的数据
- (4) 基于电商评论的数据

电商平台的数据采集：

- (1) HTML网页文本，图片-爬虫
- (2) JSON或XML文本-API

电商平台数据采集的困难：

**V**olume(数据量大)

**V**ariety(种类太多)：包括平台，研究数据，媒体数据等等。

**V**elocity(流式数据，高速性)

反爬虫技术，数据孤岛等等。