数据分析及实践

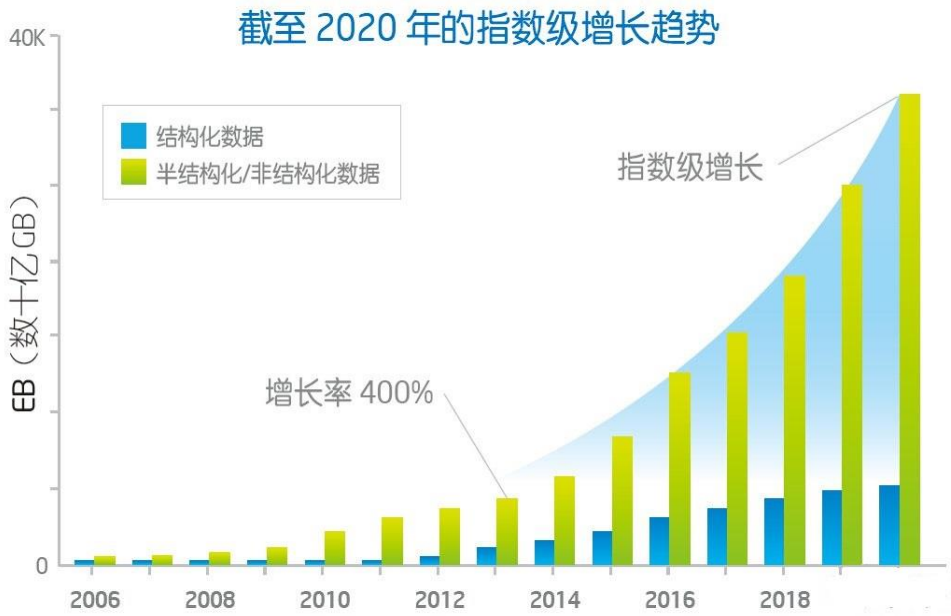
Analysis and Practice of the Data

第二章 数据入门

刘 淇
Email: qiliuql@ustc.edu.cn

Data

- 数据采集 Data Acquisition
- 数据预处理 Data Preprocessing
- 特征工程 Feature engineering



数据采集

- 获得数据的方式多种多样

网页

测量

数据库

监控

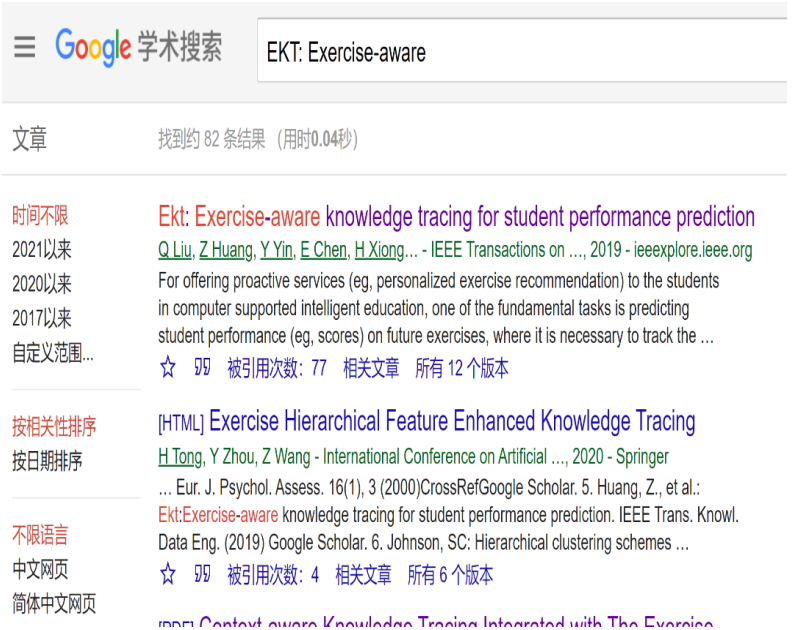
传统媒体

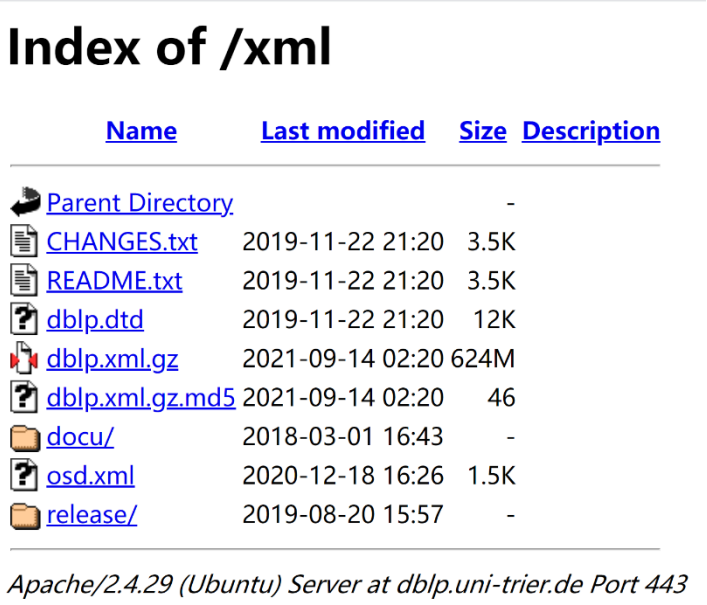
数据采集：数据检索

- 最简单、最灵活的数据获取方式就是依靠检索
- 学会使用搜索引擎
  - 百度：适合于搜索中文信息
  - Google：更适合搜索英文信息

数据采集：数据检索

- 最简单、最灵活的数据获取方式就是依靠检索
- 学会使用搜索引擎
  - Google：更适合搜索英文信息
    - Google Scholar，DBLP





数据采集：数据检索

- 最简单、最灵活的数据获取方式就是依靠检索
- 学会使用搜索引擎

数据采集：数据检索

- 最简单、最灵活的数据获取方式就是依靠检索
- 学会使用搜索引擎
  - Google：更适合搜索英文信息
    - 2018.9，Google Dataset Search（Google 数据集搜索）

      网址：https://toolbox.google.com/datasetsearch

数据采集：公开数据

- 国内常见公开数据渠道
  - 国家相关部门统计信息
  - 中国银行业监督管理委员会
  - 中国国家统计局

- 国际公开数据集
  - 1400万的图像数据
    - http://www.image-net.org/
  - Amazon从2008年开始就为开发者提供几十TB的开发数据
    - http://aws.amazon.com/datasets
  - YouTube视频的统计与社交网络数据
    - http://netsg.cs.sfu.ca/youtubedata/

数据采集：公开数据

- 代表性公开数据集
  - 用户评分MovieLens：https://grouplens.org/datasets/movielens/
  - 文本数据-头条：https://github.com/aceimnorstuvwxz/toutiao-text-classfication-dataset
  - 金融数据-股票：https://github.com/asxinyu/Stock
  - 网络数据-Large scale network：https://snap.stanford.edu/data/
  - 教育数据:
    - ASSISTmentsData-学业: https://sites.google.com/site/assistmentsdata/home/
    - BASEGroup: https://github.com/bigdata-ustc/EduData
  - 阿里天池数据-数据平台：https://tianchi.aliyun.com/dataset/
  - 公开大数据竞赛的数据：KDDCup，NeurIPS Challenge

数据采集：公开数据

github.com/bigdata-ustc

bigdata-ustc

BigData Lab @USTC 中科大大数据实验室

中国科学技术大学大数据实验室

Hefei 合肥 http://bigdata.ustc.edu.cn/index.html

Repositories 28 Packages People 1 Projects

Find a repository...

Type Language Sort

EduData

Forked from tswsxk/EduData

Edudata: Datasets in Education and convenient interface for downloading and preprocessing dataset in education

education dataset psychometrics knowledge-tracing cognitive-diagnosis datasets-education

EduData

pypi v0.0.13 build passing codecov 79% downloads 30/month

Convenient interface for downloading and preprocessing dataset in education.

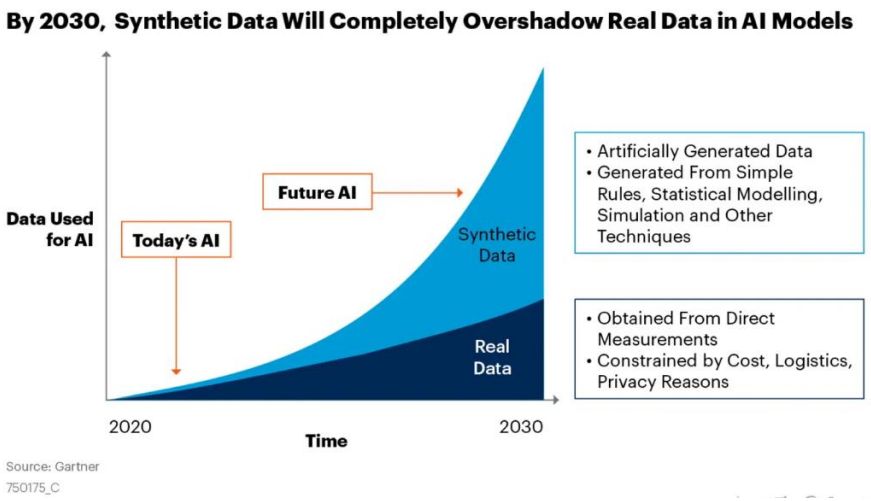
The dataset includes:

- KDD Cup 2010
- ASSISTments
- OLI Engineering Statics 2011
- JunyiAcademy Math Practicing Log [Annotation]
- slepemapy.cz
- synthetic
- math2015
- EdNet
- pisa2015math
- workbankr
- critlangacq
- math23k

数据采集：数据生成

2021年，尼日利亚数据科学的研究人员在训练计算机视觉算法的时候，发现现有的真实世界里的数据集都是大量以西方服装为特征的数据，但是没有非洲服饰。最终这个团队生成了对应非洲人的全新合成数据集，最终解决了缺少非洲服饰的问题。

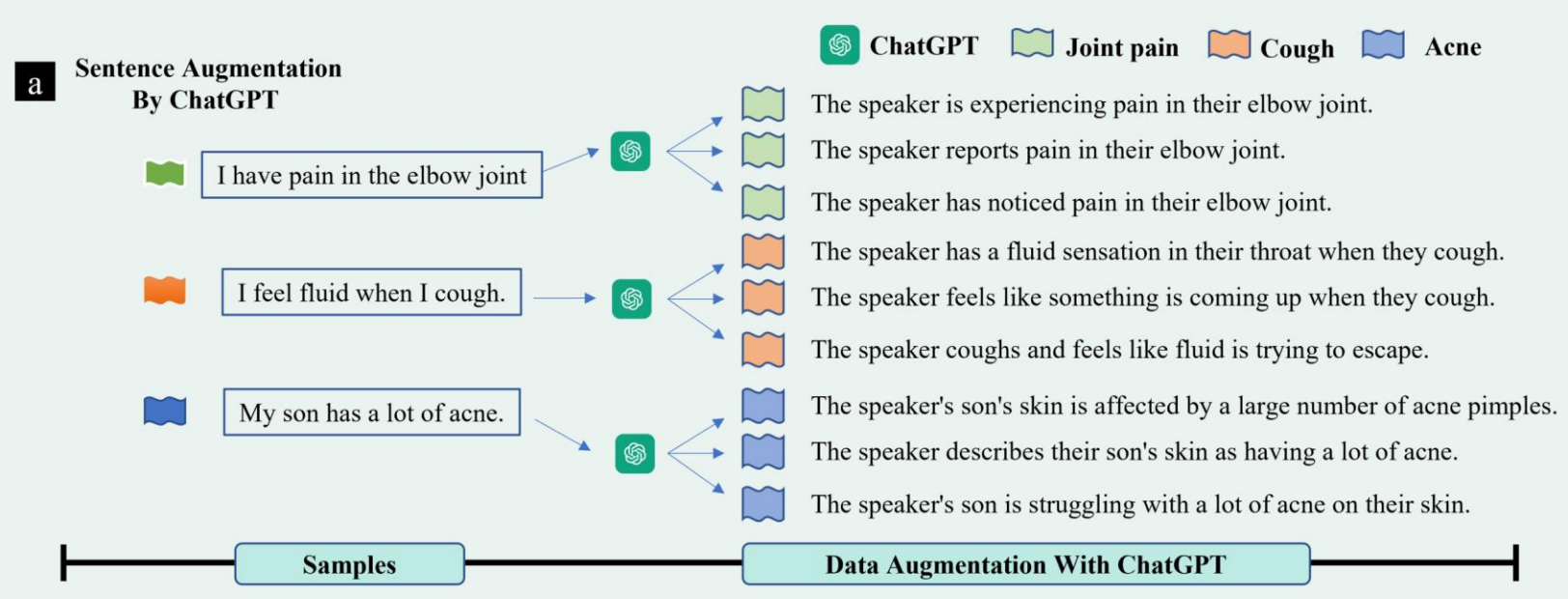
- 大数据的未来：合成数据（Synthetic Data）
  - 人为标注数据的缺点：
    - 耗时耗力，代价昂贵
    - 一些任务的数据难以获取和标注
  - 合成数据：
    - 定义：基于计算机模拟活算法生成模仿现实世界观察的人造数据
    - **优势：**大大降低数据获取成本；可控；没有隐私安全问题；数据丰富多样，减少真实世界中存在的偏见…
    - 代表工作：
      - 麻省理工学院搭建的合成数据工具库：The Synthetic Data Vault. Put synthetic data to work! (sdv.dev)



图：世界著名技术研究和分析公司 Gartner 预测，在 2030 年，绝大多数的人工智能训练数据集会使用合成数据集

数据采集：数据生成

- 利用大模型做数据增强
  - 解决目标领域数据少且质量差的问题
    - 合成数据是让模型“无中生有”，直接生成某个领域的数据（有一定质量问题）
    - 数据增强是让模型“**有中生更多**”，根据已标注好的数据，举一反三（更安全可靠）
  - 利用大模型卓越的语言理解能力，为每个训练样本的句子重新构造概念类似，但语义不同的样本



Dai, Haixing, et al. "AugGPT: Leveraging ChatGPT for Text Data Augmentation.“ arXiv preprint (2023)

数据采集：数据众包

- 数据众包： 一种利用大众力量来收集、分析和处理数据的模式
  - 需求方将一批数据标注任务分成多个子任务，分发在数据众包平台
  - 标注者通过互联网在众包平台接受任务
  - 平台收集标注结果，质量评估合格后返回给需求方

专家标注

- 单人准确性高
- 专家少
- 耗时久
- 费用高

众包标注

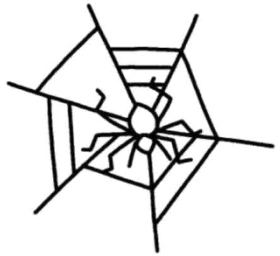
- 单人准确性低
- 标注者多
- 效率高
- 费用低

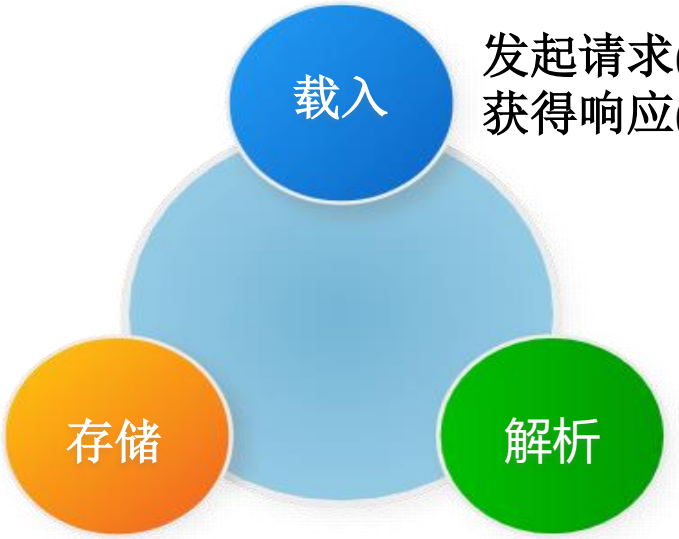
数据采集：批量数据获取

- 大量数据的获取难以手动实现，需借助**爬虫程序**
  - 也有可能通过交易（购买）“数据”而得
- 网络爬虫是一个自动在网上抓取数据的程序
  - 爬虫本质上就是**下载**特定网站网页的HTML/JSON/XML数据，并对数据进行**解析**、**提取**与**存储**
  - 通常先定义一组**入口URL**，根据页面中的其他URL，**深度优先**或**广度优先**的遍历访问，逐一抓取数据

数据采集：网络爬虫

- 网络爬虫是什么？
  - 网络爬虫（又被称为网页蜘蛛，网络机器人，网页追逐者），是一种按照一定的规则，自动的抓取万维网信息的程序或者脚本。
    - 请求网站并提取数据的自动化程序
  - 爬虫的行为可以划分为：载入、解析、存储，
    - 最复杂的部分为载入





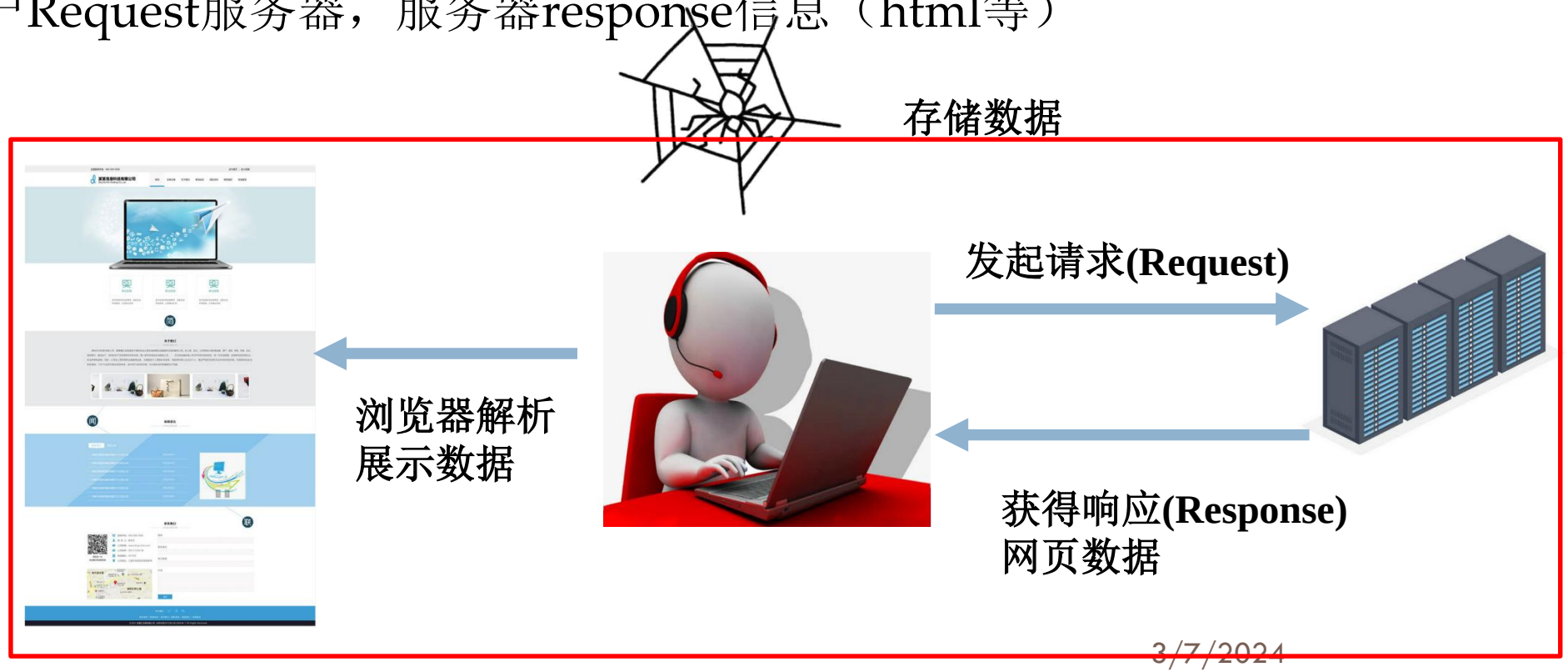
发起请求(Request)
获得响应(Response)-类型多样

获得数据
存储数据-类型多样

解析内容
提取数据-类型多样

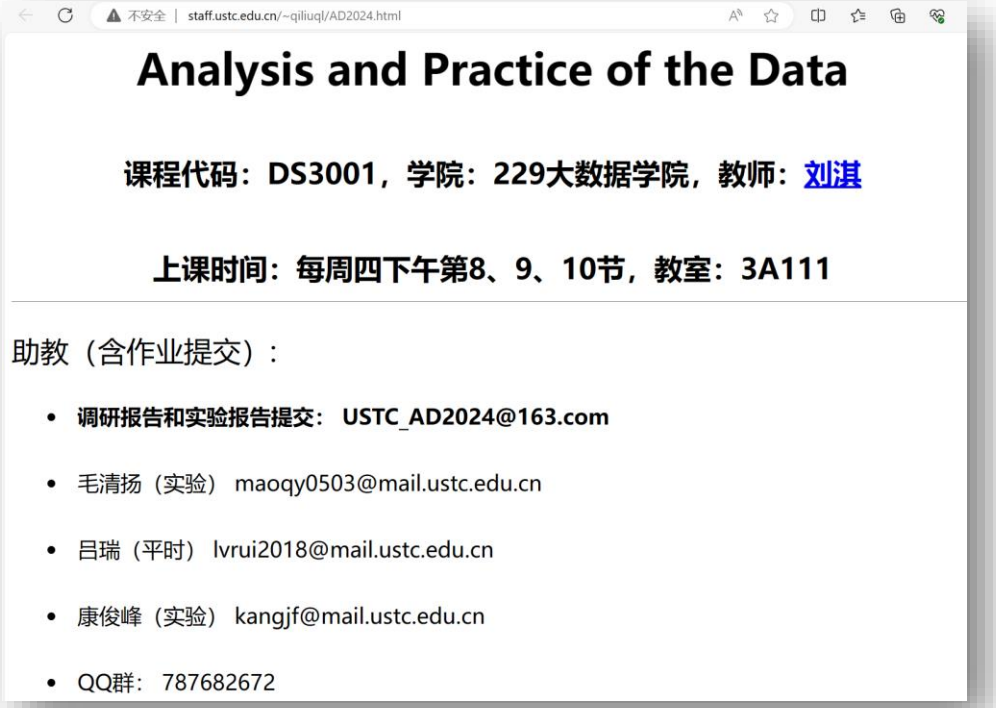
数据采集：网站数据

- 访问网页示例
  - 网站数据主要依托于网页（html, 超文本标记语言）展示
  - 用户Request服务器，服务器response信息（html等）



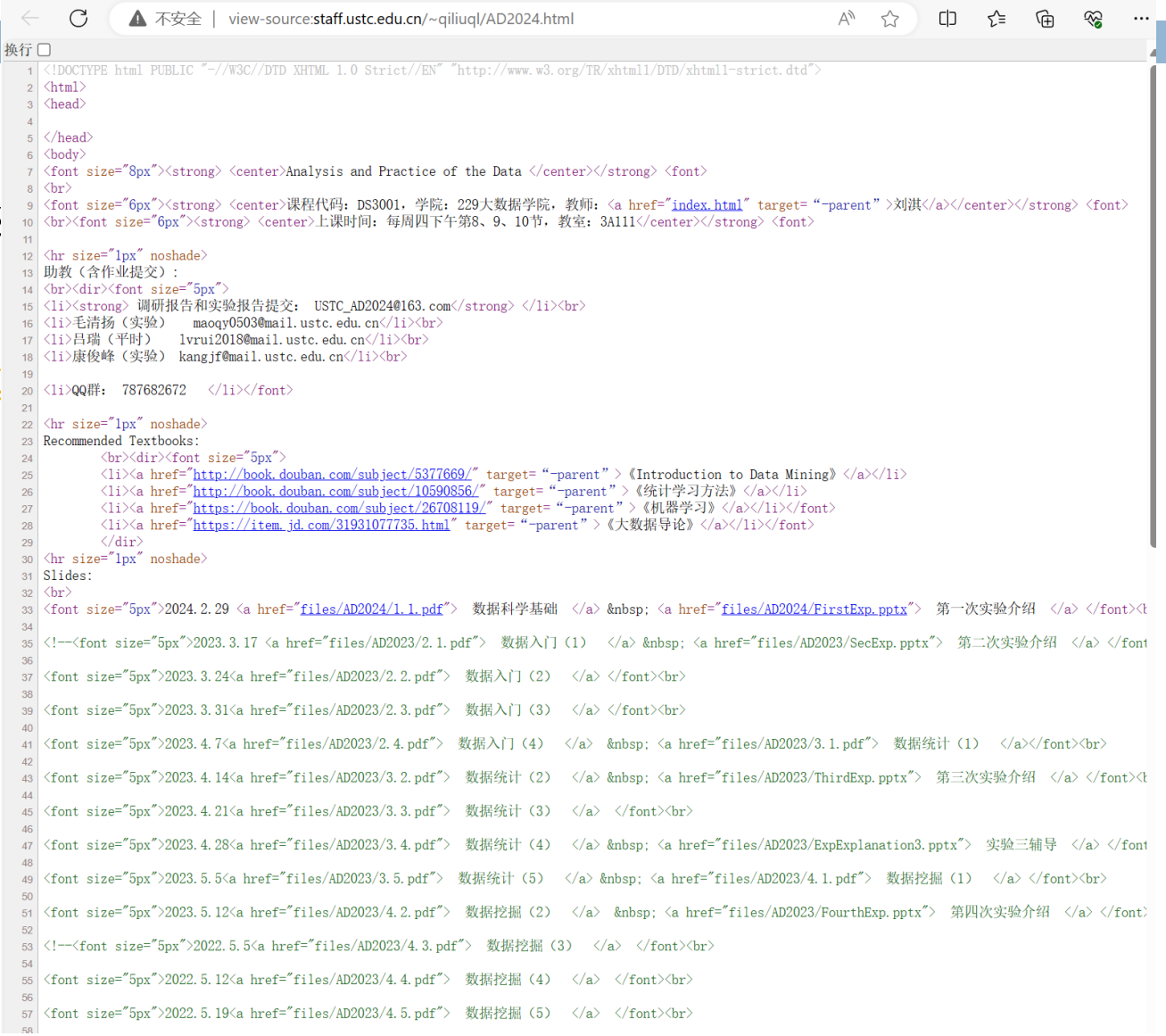
数据采集：网站数据

- 网页示例
  - 网站数据主要依托于网页（html, 超文本标记语言）展示
  - 用户Request服务器，服务器response信息（html等）
  - 课程主页：http://staff.ustc.edu.cn/~qiliuql/AD2024.html
  - 右键“检查”查看网页源代码



数据采集：网站数据

- 网页示例
  - 网站数据主要依托于网页（html, 超文
  - 用户Request服务器，服务器respons



网络爬虫：载入

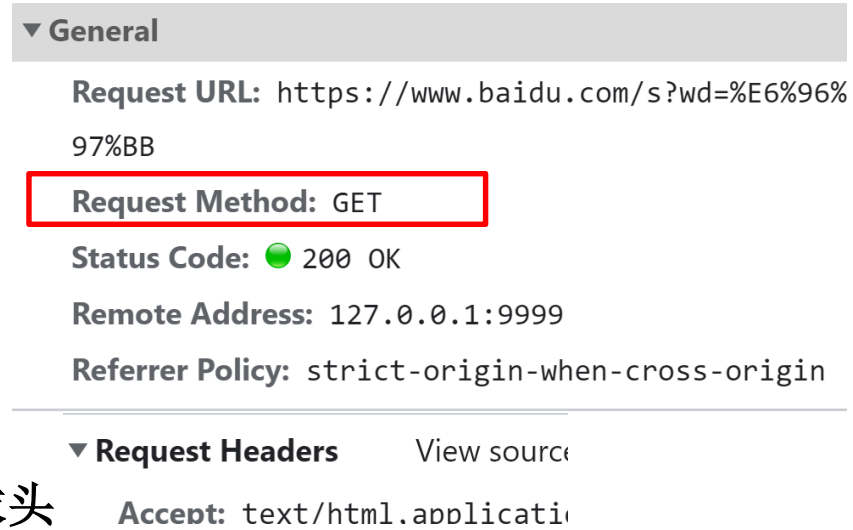
- 载入：将目标网站数据下载到本地
  - Html，HyperText Markup Language
  - 爬虫程序向服务器发送网络请求 Request，获取相应的网页
    - 网站常用网络协议：http，https
    - 数据常用请求方式：get，post
      - get：参数常放置在**URL**中
        - http://www.adc.com?p=1&q=2&r=3，
        - 问号后为参数
      - post：参数常放置在一个表单中（报文头（header））
        - 在向目标URL发送请求时，将参数放置在一个网络请求的报文头中
        - 更安全

HTTP：是互联网上应用最为广泛的一种网络协议，是一个客户端和服务器端请求和应答的标准（TCP），用于从WWW服务器传输超文本到本地浏览器的传输协议。

HTTPS：是以安全为目标的HTTP通道，简单讲是HTTP的安全版，即HTTP下加入SSL层(一类安全协议)，HTTPS的安全基础是SSL，因此加密的详细内容就需要SSL。

网络爬虫：载入

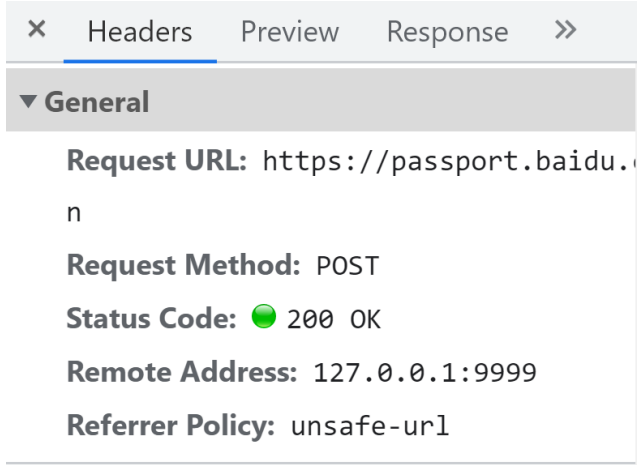
- 载入：将目标网站数据下载到本地
  - 数据常用请求方式：get，post
    - **get：** 参数常放置在**URL**中
      - http://www.adc.com?p=1&q=2&r=3，问号后为参数
      - 例如，https://www.baidu.com/s?wd=图片



请求头

网络爬虫：载入

- 载入：将目标网站数据下载到本地
  - 数据常用请求方式：get，post
    - **post：** 参数常放置在一个表单中
      - 在向目标URL发送请求时，将参数放置在一个网络请求的报文头中
      - 相比于Get，多了Form Data部分（请求体）
      - 更安全：登录操作常用（不会放在URL后面）



网络爬虫：载入

- 载入：将目标网站数据下载到本地
  - 数据常用请求方式：get，post
  - 获得服务器的响应：Response，即获取网页源代码

响应头

▼ **Response Headers** View source

Bdpagetype: 2

Bdqid: 0xdabb3fd4000073b3

响应状态：200， 404等

HTTP状态码

响应体，即网页源代码

```
<html>
<head>

</head>
<body>
<font size="8px"><strong> <center>Analysis and Practice of the Data </center></strong> <font>
<br>
<font size="6px"><strong> <center>课程代码：DS3001，学院：229大数据学院，教师：<a href="index.html" target="-parent">刘淇</a></center></strong> <font>
<br><font size="6px"><strong> <center>上课时间：每周四下午第8、9、10节，教室：3A113</center></strong> <font>

<hr size="1px" noshade>
助教（含作业提交）：
<br><dir><font size="5px">
<li>张航 （实验）     yzwz@mail.ustc.edu.cn</li><br>
<li>李佳桐（平时）    satosasara@mail.ustc.edu.cn</li><br>
<li>QQ群： 575874232  </li></font>

<hr size="1px" noshade>
Recommended Textbooks:
        <br><dir><font size="5px">
        <li><a href="http://book.douban.com/subject/5377669/" target="-parent">《Introduction to Data Mining》</a></li>
        <li><a href="http://book.douban.com/subject/10590856/" target="-parent">《统计学习方法》</a></li>
        <li><a href="https://book.douban.com/subject/26708119/" target="-parent">《机器学习》</a></li></font>
        <li><a href="https://item.jd.com/31931077735.html" target="-parent">《大数据导论》</a></li></font>
        </dir>
<hr size="1px" noshade>
Slides:
<br>
```

网络爬虫：载入

- 实际操作：抓取一个静态网页步骤
  - 首先确定URL，例如：http://www.baidu.com
  - 其次确定请求的方式以及相关参数：
    - 直接用浏览器实现：chrome, firefox浏览器抓包工具，详见
      - http://jingyan.baidu.com/article/3c343ff703fee20d377963e7.html
    - 或者抓包工具：charles等，详见
      - http://blog.csdn.net/jiangwei0910410003/article/details/41620363/
  - 最后在代码中按照特定的请求方式（get，post）向URL发送参数，即可收到网页的结果

网络爬虫：载入

- 但部分页面的数据是动态加载的
  - Ajax异步请求
    - 网页中的部分数据需要浏览器渲染 (JavaScript调用接口获取数据)
    - 用户的某些点击、下拉的操作触发才能获得
  - 解决方案：
    - 借助抓包工具，分析Ajax某次操作所触发的请求，通过代码实现相应的请求
      - 有技术难度，但抓取速度快。
    - 利用智能化的工具：selenium webdriver
      - 用程序控制驱动浏览器，模拟浏览器
      - 可以模拟实现人的所有操作
      - 操作简单，但是速度慢
        - 因为爬虫需要启动浏览器，浏览器需要渲染页面，所以速度比较慢
    - 其他：Splash，Pyv8等

网络爬虫：载入

- 反爬虫：随着网络爬虫对目标网站访问频率的加大，网站禁止爬虫程序继续访问
- 常见反爬手段：
  - 出现用户登录界面，需要验证码
  - 禁止某个固定帐号或ip一段时间内访问网站
  - 更有甚者，直接返回错误的无用数据
- 应对措施：
  - 优化爬虫程序，尽量减少访问次数，尽量不抓取重复内容
  - 使用多个cookie（网站用来识别用户的手段，每个用户登录会生成一个cookie）
  - 使用多个ip（可以用代理实现）

网络爬虫：解析

- 解析：在载入的结果中抽取特定的数据，载入的结果主要分成三类html、json、xml 结构化方法用以标记数据
  - html
    - Java工具包：jsoup等
    - Python工具包：beautifulSoup等
  - json
    - Java工具包：json-lib、org-json、jackson等
    - Python工具包：json、demjson等
  - Xml
    - Java工具包：dom4j等
    - Java工具包：xml、libxml2等

解析内容
提取数据-
类型多样

网络爬虫：解析(对比JSON与XML)


```
{
  "name": "中国",
  "province": [{
    "name": "黑龙江",
    "cities": {
      "city": ["哈尔滨", "大庆"]
    }
  },
   {
    "name": "广东",
    "cities": {
      "city": ["广州", "深圳", "珠海"]
    }
  },
  .....
  }]
}
```

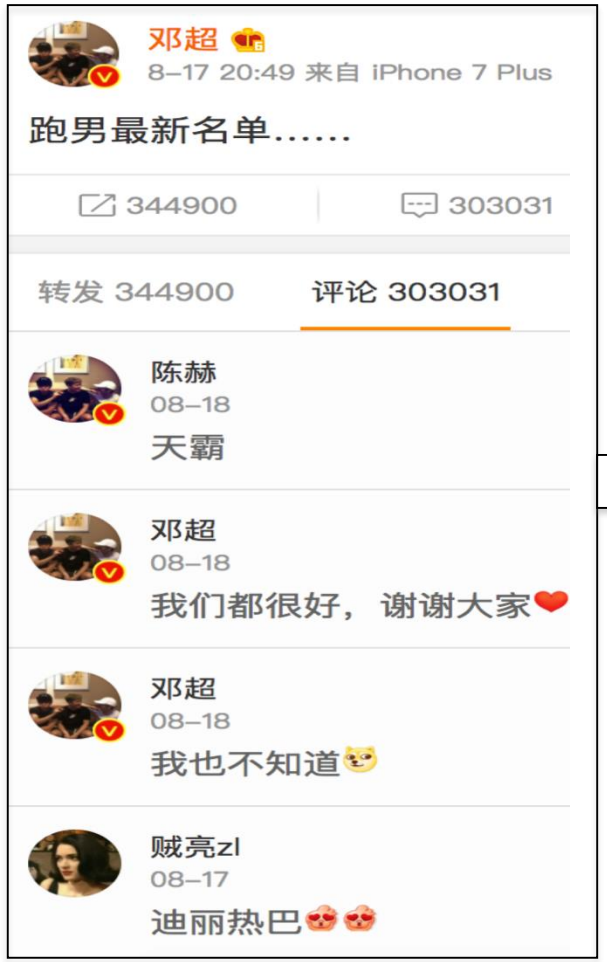
对象，成员：键值对

```
<?xml version="1.0" encoding="utf-8"?>
<country>
  <name>中国</name>
  <province>
    <name>黑龙江</name>
    <cities>
      <city>哈尔滨</city>
      <city>大庆</city>
    </cities>
  </province>
  <province>
    <name>广东</name>
    <cities>
      <city>广州</city>
      <city>深圳</city>
      <city>珠海</city>
    </cities>
  </province>
 .........
</country>
```

网络爬虫：解析(对比JSON与XML)

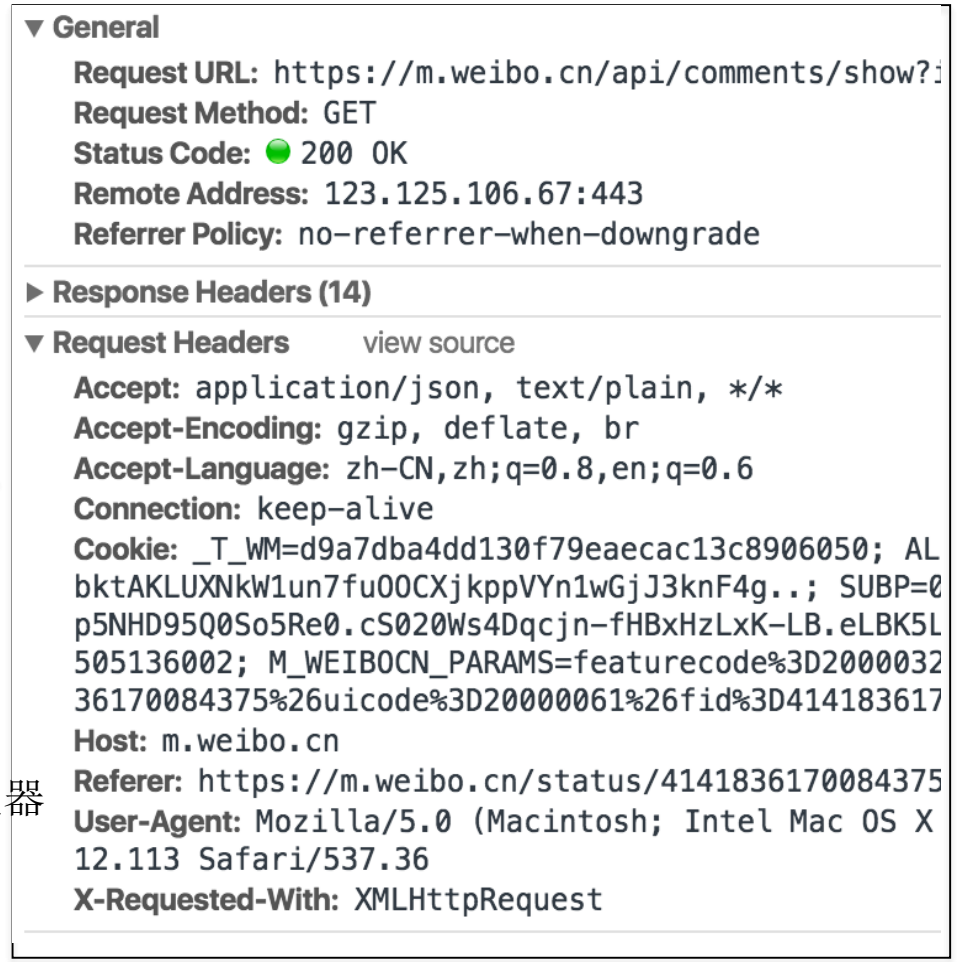
- 可读性
  - Json简洁，XML规范，xml比较好
- 可扩展性
  - 均很好
- 数据体积
  - Json数据量少，传输快。Xml数据量大，传输慢
- 编码解码
  - Json容易，xml复杂（树结构，父子节点）
- 数据描述
  - Xml数据描述更好
- 数据交互
  - Json与JavaScript交互更方便，易于解析。XML更适合跨平台共享

网络爬虫：抓取微博评论



抓包工具
获取请求

可以使用浏览器
开发者工具



抓取单博主的所有微博及其评论
https://zhuanlan.zhihu.com/p/26793251

浏览器开发者工具基本使用教程
https://blog.csdn.net/changjiang366/article/details/80512474

网络爬虫：抓取微博评论

获得评论的json格式

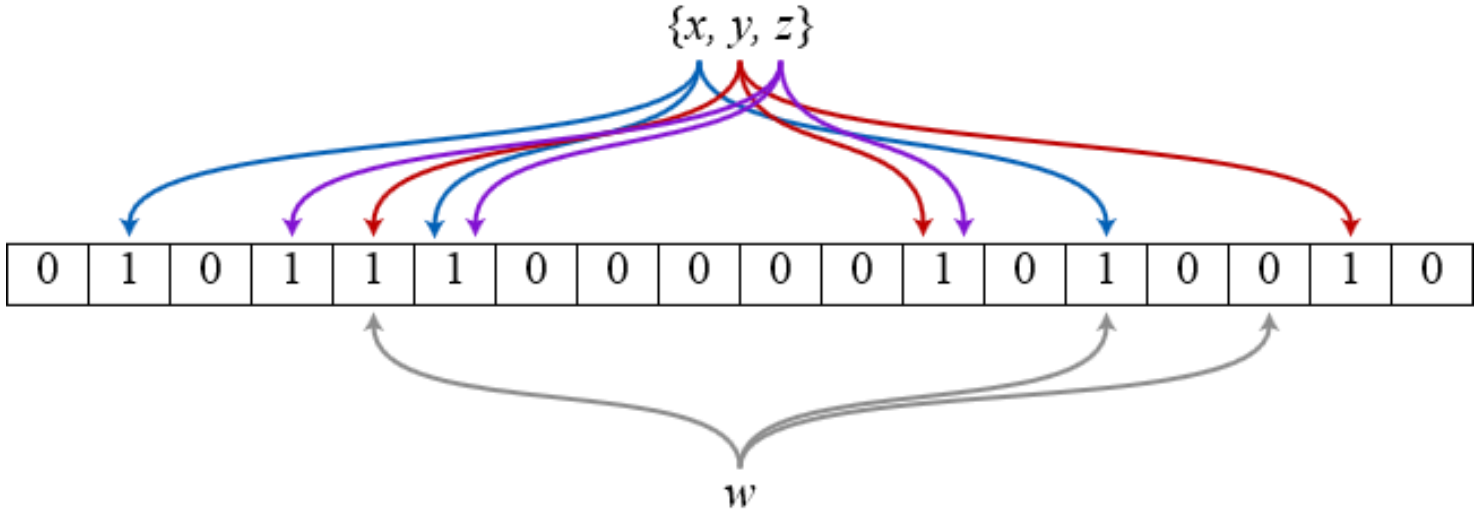

```
豆ICP备15025187号-1 邮箱: service@json.cn
{
    "mod_type":"mod/pagelist",
    "previous_cursor":"",
    "next_cursor":"",
    "card_group":[
        {
            "id":4142016554789113,
            "created_at":"08-18 08:46",
            "source":"柔光自拍vivo X7",
            "user":Object{...},
            "text":"回复<a
href="/n/%E9%82%93%E8%B6%85">@邓超</a>:不管是谁,
请大家记住陈赫的话，他们很好，感情都很好。恳请各家粉丝不要
戏太多就好<i class="face face_1 icon_1">[微笑]</i>
没准你们那么嫌弃骂的那么难听，人家正主还是感情好的时不时去
吃火锅呢，你们不累吗？别用自己对他的爱去给他造成困扰",
            "reply_id":4142015488402958,
            "reply_text":"<a
href='/u/'5187664653>@邓超</a>: 我也不知道<i
class="face face_1 icon_20">[doge]</i>",
            "like_counts":10811,
            "liked":false,
            "mod_type":"mod/single/infobox"
```


解析出需要的字段

用户id	时间	内容
陈赫	08-18	天霸
邓超	08-18	我们都很好，谢谢大家
邓超	08-18	我也不知道
贼亮zl	08-17	迪丽热巴
……	……	……

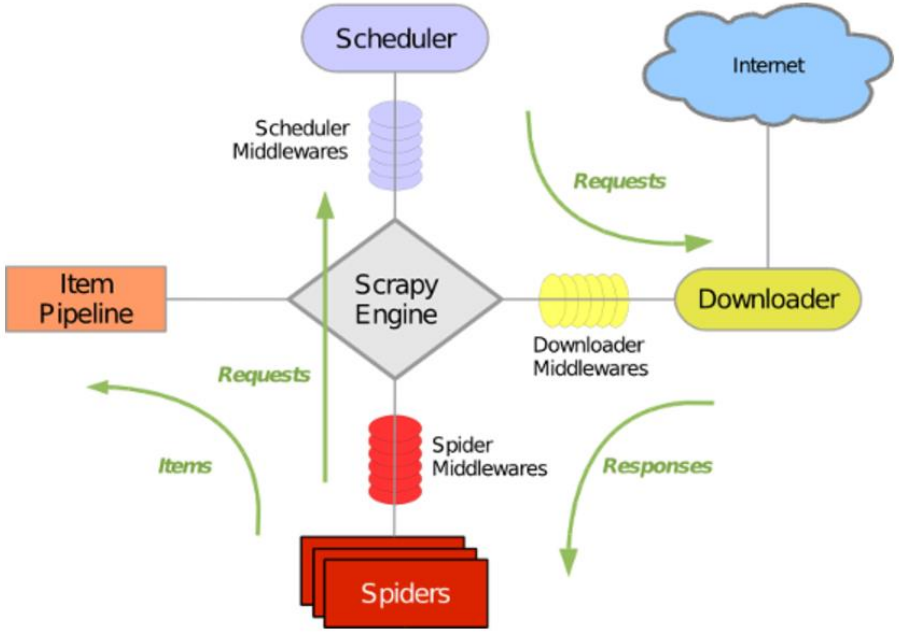
网络爬虫：去重服务

- 去重服务：避免信息的重复抓取，减小存储空间
  - **Bloom过滤器**：由一个很长的二进制向量和一系列随机映射函数组成，通过多个hash函数将一个元素映射成一个位阵列（Bit Array）中的多个点
    - 只有多个hash结果都一样时，才说明数据是重复的



网络爬虫：现有技术

- 基于Java的工具
  - HttpClient
  - Jsoup
- 基于Python的工具
  - **网页爬取URLLIB库(无需安装)**
  - 网页解析Beautiful-soup库
  - **网页爬取+解析Scrapy框架(需要安装)**



现有的爬虫框架很成熟，能够合理的控制爬取的过程，并有效的处理爬取过程中出现的各种异常

网络爬虫：现有技术

- Urllib是Python内置的HTTP请求库
  - 包含request, error, parse, robotparse等模块
  - Urllib.request —— 用于打开 URL 的可扩展模块（官方文档）
    - 定义了适用于在各种复杂情况下打开 URL（主要为HTTP）的函数和类
  - 常用函数
    - urlopen(url, data=None, [timeout,]*)：**打开统一资源定位符 url**，可以是一个字符串或一个 Request 对象。
    - Request(url, data=None, headers={}, method=None)：**URL请求对象的抽象类**（可用于对头信息headers进行模拟）

```
import urllib.request
import urllib.parse

url = 'http://httpbin.org/post'
# 添加请求头
headers = {'user-agent': 'Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36(
dict = {
    'name':'WenAn'
}
data = bytes(urllib.parse.urlencode(dict), encoding = 'utf-8')
request = urllib.request.Request(url, data=data, headers=headers, method='POST')
response = urllib.request.urlopen(request)
print(response.read().decode('utf-8'))
```

网络爬虫：现有技术

- ItSucks工具
  - 支持通过下载模板和正则表达式来定义下载规则
  - 提供swing GUI操作界面
- Spidernet工具
  - 以递归树为模型的多线程web爬虫程序
  - 存储于sqlite数据文件
- 完整解决方案
  - 基于用户浏览器的爬虫（插件）
  - 八爪鱼
  - 火车采集器（合肥乐维）
  - 数据堂

乐维公司的核心产品——火车采集器是国内使用人数最多的互联网数据采集软件，曾多次被网易新闻、电脑报、安徽商报等知名媒体报道，广受业界关注。

网络爬虫：协议

- 域名 + /robots.txt
  - Robots协议（也称为爬虫协议、机器人协议等）的全称是“网络爬虫排除标准”（Robots Exclusion Protocol），网站通过Robots协议告诉爬虫哪些页面可以抓取，哪些页面不能抓取。

taobao.com/robots.txt

指定对哪些爬虫生效

```
User-agent: Baiduspider
Disallow: /

User-agent: baiduspider
Disallow: /
```

指定要屏蔽的网址

禁止百度爬虫爬取所有页面

- 用户协议/版权声明/法律条款

为了保护版权人的合法权益，“[redacted]系统（[redacted]）”（下称“[redacted]”）发表如下版权声明：

一、[redacted]平台题库内容，为[redacted]全力打造的咨询研究和分享平台，其目的在于共同提高和一起分享，提升受众对于市场的及时和动态把握，并提高专业素养；

二、[redacted]平台的所有页面内容的著作权人，未经[redacted]授权，任何个人或者组织不得抄袭、转载、摘编、修改本网站内容；

三、本网站所载文章或所提供的下载文件数据仅供参考，使用者务请核实，风险自负；

四、凡[redacted]网站及相关网站平台所提供下载的文件中注明“来源：XXX”的作品，均转载自公开资源，转载目的在于传递更多信息或提供研究机会，并不代表[redacted]赞同其观点和对其真实性负责。如其他媒体、网站或个人从本文件转载使用，必须保留本文件注明的“稿件来源”，并自负版权等法律责任。如擅自篡改稿件来源，其产生一切后果与[redacted]无关；

禁止爬取后公开

数据采集

- 注意网站规定
- 注意法律规定
  - 2021年6月《中华人民共和国数据安全法》
  - 2023年7月《生成式人工智能服务管理暂行办法》
- 注意数据使用规范
- etc

第七条 生成式人工智能服务提供者（以下称提供者）应当依法开展预训练、优化训练等训练数据处理活动，遵守以下规定：

（一）使用具有合法来源的数据和基础模型；

（二）涉及知识产权的，不得侵害他人依法享有的知识产权；

（三）涉及个人信息的，应当取得个人同意或者符合法律、行政法规规定的其他情形；

（四）采取有效措施提高训练数据质量，增强训练数据的真实性、准确性、客观性、多样性；

（五）《中华人民共和国网络安全法》、《中华人民共和国数据安全法》、《中华人民共和国个人信息保护法》等法律、行政法规的其他有关规定和有关主管部门的相关监管要求。

第八条 在生成式人工智能技术研发过程中进行数据标注的，提供者应当制定符合本办法要求的清晰、具体、可操作的标注规则；开展数据标注质量评估，抽样核验标注内容的准确性；对标注人员进行必要培训，提升尊法守法意识，监督指导标注人员规范开展标注工作。

数据采集

- 注意网站规定
- 注意法律规定
  - 2021年6月《中华人民共和国数据安全法》
  - 2023年7月 《生成式人工智能服务管理暂行办法》
- 注意数据使用规范
- etc

https://code.bdaa.pro/privacy

课达编程 (CODIA) · 隐私政策

引言

课达编程重视用户的隐私。您在使用我们的服务时，我们可能会收集和使用您的相关信息。我们希望通过本《隐私政策》向您说明，在使用我们的服务时，我们如何收集、使用、储存和分享这些信息，以及我们为您提供的访问、更新、控制和保护这些信息的方式。本《隐私政策》与您所使用的课达编程服务息息相关，希望您仔细阅读，在需要时，按照本《隐私政策》的指引，作出您认为适当的选择。本《隐私政策》中涉及的相关技术词汇，我们尽量以简明扼要的表述，并提供进一步说明的链接，以便您的理解。

您使用或继续使用我们的服务，即意味着同意我们按照本《隐私政策》收集、使用、储存和分享您的相关信息。

如对本《隐私政策》或相关事宜有任何问题，请通过 codia-support@bdaa.pro 与我们联系。

网络爬虫：注意事项

- 重点关注：下列情况下，爬虫有可能违法，严重的甚至构成犯罪。
  - 爬虫程序规避网站经营者设置的反爬虫措施或者破解服务器防抓取措施，非法获取相关信息，情节严重的，有可能构成“**非法获取计算机信息系统数据罪**”。
  - 爬虫程序干扰被访问的网站或系统正常运营，后果严重的，触犯刑法，构成“**破坏计算机信息系统罪**”
  - 爬虫采集的信息属于公民个人信息的，有可能构成非法获取公民个人信息的违法行为，情节严重的，有可能构成“**侵犯公民个人信息罪**”。

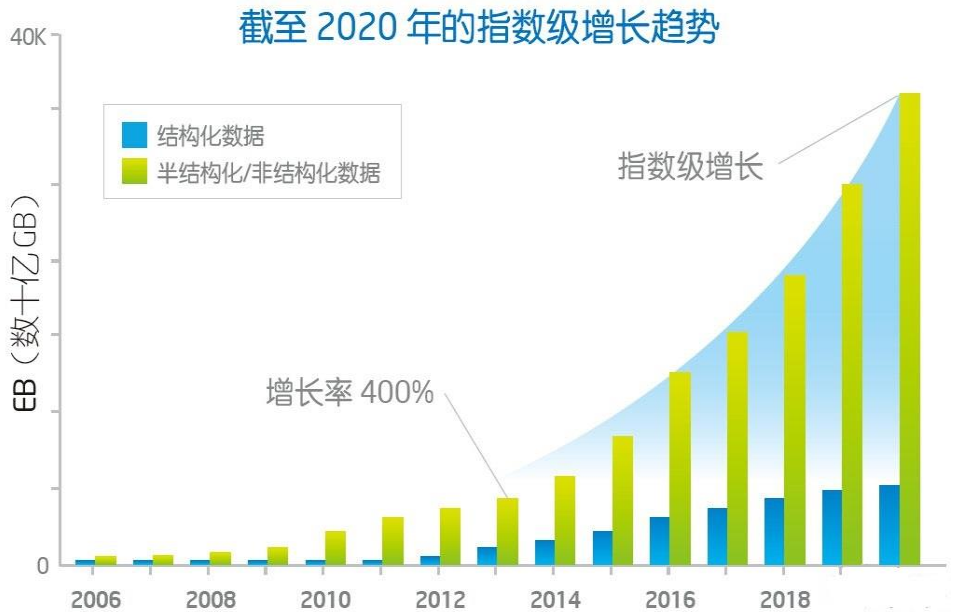
网络爬虫：合规建议

- 非法获取计算机信息系统数据罪，合规建议：
  - 严格**禁止通过技术手段绕过服务器的访问限制**，或**破解被爬网站为保护数据而采取的加密算法及技术保护措施**，从而对被爬网站受保护的计算机信息系统中的数据进行爬取。若被爬网站设定了获取数据信息的措施，爬虫企业应**避免通过伪造实名认证或窃取账号密码、内部权限**的形式获取数据。
- 非法侵入计算机信息系统罪，合规建议：
  - 对大数据公司，特别是大数据风控企业来说，获取“裁判文书网”“执行信息公开网”的数据非常普遍且重要，但爬取这类国家事务网站的信息时应当尤为审慎，特别是在网站已采取相关“反爬措施”的情况下，仍**强行恶意突破防护措施爬取数据，对网站运行造成影响的**，均可能构成本罪。
- 侵犯公民个人信息罪，合规建议：
  - 利用爬虫技术获取公民个人信息的，应该严格遵守相关法律、行政法规、部门规章的规定，否则极易落入“非法获取”公民个人信息的法律风险范畴。**收集已合法公开的个人信息应不属于违法，但在立法尚不完善的阶段，仍建议谨慎使用爬虫技术抓取公开的个人信息**。

严格遵循《中华人民共和国网络安全法》等法律法规，只采集公开数据，不涉及第三方隐私泄露等问题

数据分析基础

- 数据采集 Data Collection
- 数据存储 Data Storage
- 数据预处理 Data Preprocessing
- 特征工程 Feature engineering



数据存储

- 我们面对的数据有哪些？

MovieLens

*ratings.csv - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
userId,movieId,rating,timestamp
1,1,4.0,964982703
1,3,4.0,964981247
1,6,4.0,964982224
1,47,5.0,964983815
1,50,5.0,964982931
1,70,3.0,964982400
1,101,5.0,964980868
```

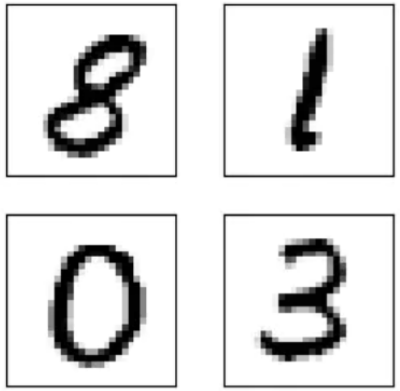
SQuAD

Context:
Computational complexity theory is a branch of the theory of computation in theoretical computer science that focuses on classifying computational problems according to their inherent difficulty, and relating those classes to each other. A computational problem is understood to be a task that is in principle amenable to being solved by a computer, which is equivalent to stating that the problem may be solved by mechanical application of mathematical steps, such as an algorithm.

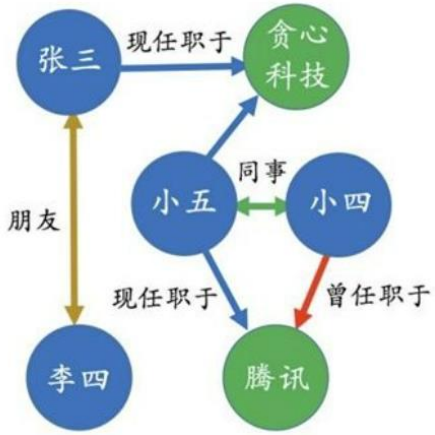
Question:
By what main attribute are computational problems classified using computational complexity theory?

Answer:
inherent difficulty

Image



Graph

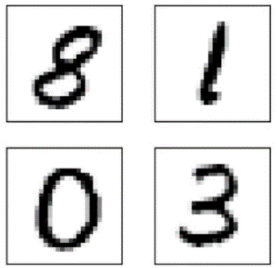


数据存储

- 结构化数据
  - 可以使用关系型数据库表示和存储的数据，拥有固定结构
- 半结构化数据
  - 弱结构化，虽然不符合关系型数据模型的要求，但是含有相关的标记(自描述结构)，分割实体及期属性 。如：XML，JSON等
- 非结构化数据
  - 没有固定数据结构，或很难发现统一数据结构的数据
  - 文档，文本，图片，视频，音频等

ID	时间	内容
陈赫	08-18	天霸
邓超	08-18	我们都很好
邓超	08-18	我也不知道

```
<province>
  <name>黑龙江</name>
  <cities>
    <city>哈尔滨</city>
    <city>大庆</city>
  </cities>
</province>
```

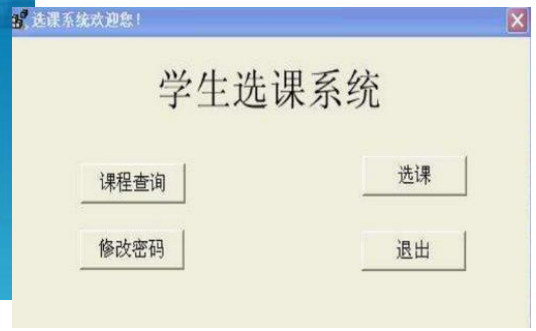


数据存储

- 人脑，书籍，文件
- 硬盘文件系统（Excel, word, txt…）

数据存储：数据库

- 数据(Data)
- 数据库(Database, DB)
  - 长期储存在计算机内、有组织的、可共享的大量数据的集合。
- **数据库管理系统**(Database Management System, DBMS)
  - 位于用户与操作系统之间的一层数据管理软件
- 数据库系统(Database System，简称DBS)
  - 在计算机系统中引入数据库后的系统构成
  - DB、DBMS（及其开发工具）、应用系统、数据库管理员



数据存储：数据库

- 数据库有哪些

数据存储：关系型数据库

- 关系型数据库系统（SQL）
  - 关系模型中数据的逻辑结构是一张二维表，行与列
  - 特点：结构化
  - 概念：关系，属性，元组
  - **优点1：**管理效率高（十万，百万，千万…）

记录三个学生信息：

- 学号：001，姓名：张三，年龄：20，性别：男，专业：计算机
- 学号：002，姓名：李四，年龄：19，性别：女，专业：管理
- 学号：003，姓名：王五，年龄：21，性别：男，专业：数学
- 。。。

关系

学生信息表

属性

学号	姓名	年龄	性别	专业
001	张三	20	男	计算机
002	李四	19	女	管理
003	王五	21	男	数学
…	…	…	…	…

元组

数据存储：关系型数据库

- 关系必须是规范化的，满足一定的规范条件（范式）
  - 最基本的规范条件（第一范式）：关系的每一个分量必须是一个不可分的数据项, 不允许表中还有表

图中工资和扣除是可分的数据项 ,不符合关系模型要求

职工号	姓名	职 称	应发工 资			扣 除		实发工资
			基 本	津 贴	职务	房 租	水 电	
86051	陈 平	讲 师	1305	1200	50	160	112	2283
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

图 一个工资表(表中有表)实例

数据存储：关系型数据库

- 关系型数据库系统（SQL）
  - 数据库操作：增、删、改、查，通过SQL语言完成
  - **优点2：**存取路径对用户隐蔽，用户只要指出“干什么”，不必详细说明“怎么干”

学生信息表

学号	姓名	年龄	性别	专业
001	张三	20	男	计算机
002	李四	19	女	管理
003	王五	21	男	数学
…	…	…	…	…

增加学生信息：

- 学号：004，姓名：赵六，年龄：22，性别：男，专业：计算机

删除002号学生的信息

修改003号学生的专业为计算机

查询年龄大于20的学生信息

数据存储：关系型数据库

- 实现：SQL（Structured Query Language）结构化查询语言，是关系数据库的标准语言

［例2］ 将学生张成民的信息插入到Student表中。

```
INSERT
        INTO  Student
        VALUES (‘200215126’,  ‘张成民’,  ‘男’, 18，'CS');
```

```
1 • INSERT
2   INTO  Student
3   VALUES ("200215126", "张成民", "男", 18, "CS");
4 • SELECT * FROM student;
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Sno	Sname	Ssex	Sage	Sdept
200215126	张成民	男	18	CS
200215128	陈冬	男	18	IS

数据存储：关系型数据库

- 有没有不足？—冗余

Student表

学号	所在系	系主任	课程名	成绩
S1	计算机系	张明	C1	95
S2	计算机系	张明	C1	90
S3	计算机系	张明	C1	88
S4	计算机系	张明	C1	70
S5	计算机系	张明	C1	78
….	…	…	…	…

关系模式
Student<U, F>中
存在的问题

1. 数据冗余太大：系主任
2. 更新异常：更新系主任
3. 插入异常：数学系没有学生，插入系主任
4. 删除异常：学生毕业了

数据存储：关系型数据库

- **数据库的完整性：** 实体完整性，参照完整性，用户定义完整性

主码不为空

Student表

学 号 Sno	姓 名 Sname	性 别 Ssex	年 龄 Sage	所 在 系 Sdept
200215121	李勇	男	20	CS
200215122	刘晨	女	19	CS
200215123	王敏	女	18	MA
200515125	张立	男	19	IS

主码　　外码

Course表

课程号 Cno	课程名 Cname	先行课 Cpno	学分 Ccredit
1	数据库	5	4
2	数学		2
3	信息系统	1	4
4	操作系统	6	3
5	数据结构	7	4
6	数据处理		2
7	PASCAL语言	6	4

Student-Course表

学 号 Sno	课程号 Cno	成绩 Grade
200215121	1	92
200215121	2	85
200215121	3	88
200215122	2	90
200215122	3	80

成绩范围在[0,100]

数据存储：关系型数据库

- 数据库的安全性：自主存取控制与强制存取控制
  - 自主存取控制（Discretionary Access Control，简称DAC）
    - 用户可“自主”地决定将数据的存取权限授予何人、决定是否也将“授予”的权限授予别人
  - 强制存取控制（Mandatory Access Control，简称 MAC）
    - 系统“强制”地给用户和数据标记安全等级

(1)仅当主体(如用户)的许可证级别大于或等于客体（数据，表，索引等）的密级时，该主体才能读取相应的客体

(2)仅当主体的许可证级别小于或等于客体（数据）的密级时，该主体才能写相应的客体

数据存储：关系型数据库

- 数据库的效率：索引
  - 建立索引（Index）的目的：加快查询速度
  - 谁可以建立索引
    - DBA 或 表的属主（即建立表的人）
    - DBMS一般会自动建立以下列上的索引

      PRIMARY KEY

      UNIQUE
  - 谁维护索引

    DBMS自动完成
- 使用索引

  DBMS自动选择是否使用索引以及使用哪些索引

数据存储：关系型数据库

- 数据库的效率：索引
- **RDBMS中索引一般采用B+树、HASH索引来实现**
  - **B+/B-树索引具有动态平衡的优点**
  - **HASH索引具有查找速度快的特点**

数据存储：NoSQL

- NoSQL（Not Only SQL）
  - 非关系型的DBMS，不同于传统关系型DB的DBMS的统称
  - 里程碑：2009年，强调KV和文档数据库优点，不反对RDMBS
  - 超大规模数据存储，不需要固定模式，无需多余操作横向扩展

- 为什么NoSQL？（大数据）
  - 数据量的飞速增长：GB-TB-PB
  - RDBMS的范式约束、事务特性、磁盘IO等特点的限制
  - 开发需求的前期不明确。例如，直播带货

数据存储：NoSQL

- NoSQL（Not Only SQL），泛指非关系型的DBMS

分类	举例	典型应用场景	数据模型	优点	缺点
文档型数据库	**MongoDb** CouchDB	Web应用（与Key-Value类似，Value是结构化的，不同的是数据库能够了解Value的内容）	文档形式存储，类似JSON的KV存储 Value：简单类型(字符串)，复杂类型（表格，对象）	数据结构要求不严格，表结构可变，不需要像关系型数据库一样需要预先定义表结构	查询性能不高，而且缺乏统一的查询语法。
图形(Graph)数据库	**Neo4J,** InfoGrid, Infinite Graph	社交网络，推荐系统等。专注于构建关系图谱	图结构	利用图结构相关算法。比如最短路径寻址，N度关系查找等	经常需对整个图做计算才能得出需要的信息，且这种结构不太好做分布式集群方案
键值（key-value）	**Redis,** Cabinet/Tyrant, Voldemort, Oracle BDB	内容缓存，用户信息等，如会话，日志等。主要用于处理大量数据的高访问负载	Key 指向 Value 的键值对，通常用hash table来实现	查找速度快，可以通过key快速查询到其value。一般来说，存储不管value的格式，照单全收。	数据无结构化，通常只被当作字符串或者二进制数据
列存储数据库	**HBase,** Cassandra	日志，分布式的文件系统，时空数据等	以**列簇式存储**，将同一列数据存在一起	查找速度快，可扩展性强，方便做数据压缩，对针对某一列或者某几列的查询有IO优势。	功能相对局限缺乏统一查询语言

数据存储：文档数据存储-MongoDB

- MongoDB简介
  - 基于分布式文件存储，由 C++ 语言编写，旨在为 WEB 应用提供可扩展的高性能数据存储解决方案。
  - 介于关系数据库和非关系数据库之间的产品，是非关系数据库当中功能最丰富，最像关系数据库的。
  - 将数据存储为一个文档，数据结构由键值(key=>value)对组成，类似于 JSON 对象。Value可以包含其他文档，数组及文档数组
  - 几乎可以支持RDBMS的绝大多数功能，还支持索引

```
{
  name: "sue",                    ← field: value
  age: 26,                        ← field: value
  status: "A",                    ← field: value
  groups: [ "news", "sports" ]    ← field: value
}
```

与sql不同，各个字段不需要预先定义好数据类型，所以mongodb很灵活

什么时候使用MongoDB?

- 一般情况
  - Sql能存储的一般都可以用mongodb存储（事务除外）
- 非常适合日志、博客等比较杂乱的系统的存储
  - 种类较多、范围较大、内容也比较杂乱
  - 原因：在数据集合collection中，document对字段没有强约束
- 大文件存储
  - 二进制存储，可以用pickle等工具包对其进行压缩
  - GridFS：是MongoDB规范，用于存储和检索图片、音频、视频等大文件，可以存储超过16M的文件

集合(collection)就是一组文档（document），如果说MongoDB中的文档（document）类似于关系型数据库中的行，那么集合就如同表

MongoDB存储举例——存储字段


```
> db.student.insert({"_id":"BA18011000", "name":"zhang san",
"sex":"male","age":18,"introduction":"Zhang San is a handsome guy!"})
```

```
db.getCollection('student').find({})
```

显示该数据集合

student 0.001 sec.

	_id	name	sex	age	introduction
1	BA18011000	zhang san	male	18.0	Zhang San is a handsome guy!

```
> db.student.insert({ “_id” : “BA18011001” , “name” : “Li Si", "sex":"female",
"age":17,"introduction":"Li Si is a beautiful girl!", "class":["Math","Music","Physics"]})
```

```
db.getCollection('student').find({})
```

document对字段没有强约束，Value可以是各种类型的文档

student 0.001 sec.

	_id	name	sex	age	introduction	class
1	BA18011000	zhang san	male	18.0	Zhang San is a handsome guy!	
2	BA18011001	Li Si	female	17.0	Li Si is a beautiful girl!	[3 elements]

MongoDB存储举例——存储图片

```
file_path = "E:\\USTC\\store_data\\photo"
files = os.listdir(file_path)
# print(files)

pics =[]
#遍历图片目录集合
for index, file in enumerate(files):
    filename = file_path + '\\' + file
    print(filename)
    with open(filename, "rb") as b_image:
        content = b_image.read()
        pics.append(content)
```

二进制读取图片

```
dic = {
    "_id" : "BA18011002",
    "name" : "Wang Wu",
    "sex" : "female",
    "age" : 19.0,
    "introduction" : "Wang Wu studies very hard!",
    "class" : [
        "Math",
        "Physics",
        "chemistry"
    ],
    "picture": pics
}
student.insert_one(dic)
```

构造图片文档

```
db.getCollection('student').find({})
```

student 0.003 sec.

	_id	name	sex	age	introduction	class	picture
1	BA18011000	zhang san	male	18.0	Zhang San i...	"picture" : [{ "$binary" : "/9j/4AAQSkZJRgABAQEASA { "$binary" : "/9j/4AAQSkZJRgABAQEASA { "$binary" : "/9j/4AAQSkZJRgABAQEASA { "$binary" : "/9j/4AAQSkZJRgABAQEASA { "$binary" : "/9j/4AAQSkZJRgABAQEASA	
2	BA18011001	Li Si	female	17.0	Li Si is a be...		
3	BA18011002	Wang Wu	female	19.0	Wang Wu s...	[3 element...	[5 element...

MongoDB存储举例


```
db.getCollection('student').find({})
```

student 0.005 sec.

```
/* 1 */
{
    "_id" : "BA18011000",
    "name" : "zhang san",
    "sex" : "male",
    "age" : 18.0,
    "intorduction" : "Zhang San is a handsome guy!"
}

/* 2 */
{
    "_id" : "BA18011001",
    "name" : "Li Si",
    "sex" : "female",
    "age" : 17.0,
    "intorduction" : "Li Si is a beautiful girl!",
    "class" : [
        "Math",
        "Music",
        "Physics"
    ]
}
/* 3 */
{
    "_id" : "BA18011002",
    "name" : "Wang Wu",
    "sex" : "female",
    "age" : 19.0,
    "intorduction" : "Wang Wu studies very hard!",
    "class" : [
        "Math",
        "Physics",
        "chemistry"
    ],
    "picture" : [
        { "$binary" : "/9j/4AAQSkZJRgABAQEASABIAAD/4TeGRXhpZgAATU0AKgAAAAgABgALAAIAAAAmAAAIYgESAAMAAAABAAEAA
        { "$binary" : "/9j/4AAQSkZJRgABAQEASABIAAD/4Tg6RXhpZgAATU0AKgAAAAgABgALAAIAAAAmAAAIYgESAAMAAAABAAEAA
        { "$binary" : "/9j/4AAQSkZJRgABAQEASABIAAD/4V70RXhpZgAATU0AKgAAAAgABgALAAIAAAAmAAAIYgESAAMAAAABAAEAA
        { "$binary" : "/9j/4AAQSkZJRgABAQEASABIAAD/4TjeRXhpZgAATU0AKgAAAAgABgALAAIAAAAmAAAIYgESAAMAAAABAAEAA
        { "$binary" : "/9j/4AAQSkZJRgABAQEASABIAAD/4SucRXhpZgAATU0AKgAAAAgABgALAAIAAAAmAAAIYgESAAMAAAABAAEAA
    ]
}
```

正如前面所说的非关系数据库MongoDB是文档型存储，数据集合中存储的正是这样的三条文档记录，与json非常的类似。

MongoDB基本语句

- 创建数据库
  - use DATABASE_NAME
- 查看数据库
  - show dbs
- 插入文档
  - db.COLLECTION_NAME.insert(document)
- 查看文档
  - db. COLLECTION_NAME.find()
- 删除文档
  - db. COLLECTION_NAME.remove(<query>, <justOne>)

只有数据库非空，
才会显示该数据库

document可以是已经
定义好的Bson文档

可以设置一定
的查询条件

只想删除第一条找到的记录
可以设置 justOne 为 1

MongoDB VS SQL

SQL术语/概念	MongoDB术语/概念	解释/说明
database	database	数据库
table	collection	数据库表/集合
row	document	数据记录行/文档
column	field	数据字段/域
index	index	索引
table joins	embedded document	表连接，MongoDB不支持，但支持文档嵌套
primary key	primary key	主键，MongoDB自动将_id字段设置为主键

MongoDB VS MySQL

对比角度	MongoDB	MySQL
数据库模型	非关系型	关系型
存储方式	虚拟内存+持久化	根据引擎有不同
查询语句	独特的mongodb查询语句	传统的sql语句
架构特点	通过副本集和分片可实现高可用	单点，M-S，MHA，MMM，Cluster等
数据处理方式	将热数据存储于内存，高速读写	根据引擎有不同
成熟度	新兴，成熟度较低	较为成熟的体系
广泛度	在Nosql中较为完善，使用人群也在不断增长	开源数据库的份额在增加，mysql的份额也在增长
优势	在适量级内存的Mongodb的性能是非常迅速的；高扩展性，存储的数据格式是json格式；非常适合日志、博客等比较杂乱的系统的存储	拥有较为成熟的体系，成熟度很高，支持事务
劣势	不支持事务，而且开发文档不是很完全，完善	在海量数据处理的时候效率会显著变慢

数据存储：数据库

- DB-Engines: https://db-engines.com/en/ranking

418 systems in ranking, March 2024

Rank			DBMS	Database Model	Score		
Mar 2024	Feb 2024	Mar 2023			Mar 2024	Feb 2024	Mar 2023
1.	1.	1.	Oracle	Relational, Multi-model	1221.06	-20.39	-40.23
2.	2.	2.	MySQL	Relational, Multi-model	1101.50	-5.17	-81.29
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	845.81	-7.76	-76.20
4.	4.	4.	PostgreSQL	Relational, Multi-model	634.91	+5.50	+21.08
5.	5.	5.	MongoDB	Document, Multi-model	424.53	+4.18	-34.25
6.	6.	6.	Redis	Key-value, Multi-model	157.00	-3.71	-15.45
7.	7.	↑ 8.	Elasticsearch	Search engine, Multi-model	134.79	-0.95	-4.28
8.	8.	↓ 7.	IBM Db2	Relational, Multi-model	127.75	-4.47	-15.17
9.	9.	↑ 11.	Snowflake	Relational	125.38	-2.07	+10.98
10.	10.	↓ 9.	SQLite	Relational	118.16	+0.88	-15.66
11.	11.	↓ 10.	Microsoft Access	Relational	107.93	-5.24	-24.13
12.	12.	12.	Cassandra	Wide column, Multi-model	104.59	-4.69	-9.20
13.	13.	13.	MariaDB	Relational, Multi-model	95.03	-2.20	-1.81

数据存储：图像(Image)存储

- 如何存储图形图像数据？两种存储方式

存储图片路径

将图片存在本地，比如windows系统中，在数据库中写入图片的路径，用来索引图片

id	path
1	/image/apple.jpg
2	/image/car.jpg
3	/image/cat.jpg

存储图像数据

如，MINIST数据

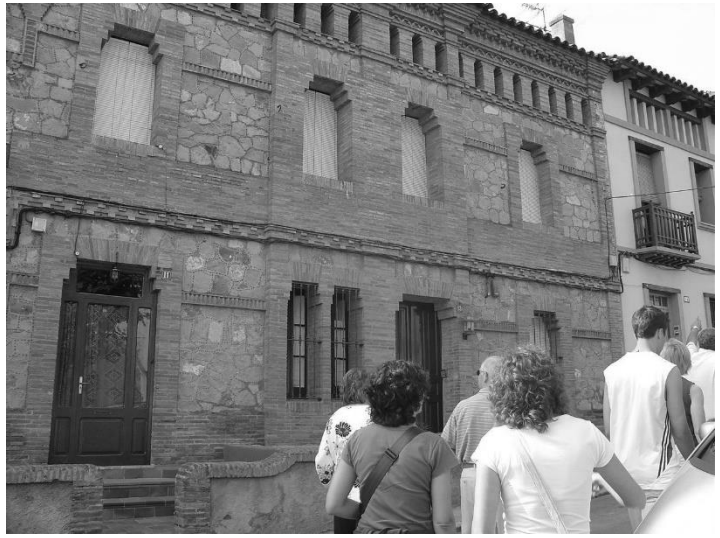
直接将图像数据存入数据库系统中。可以直接提取图像的像素值，存为numpy, json等格式数据，写入数据库系统中。也可以将其以二进制文件的格式写入。

id	data
1	[[137 124 143 111 62 248 253] [133 116 160 125 133 153 85] [102 122 123 137 142 128 130] [116 52 121 121 56 124 98] [99 116 118 36 127 134 169] [67 119 89 253 158 222 204] [100 54 110 62 202 213 184]]

数据存储：图像(Image)存储

彩色图

height: 960
width: 1280

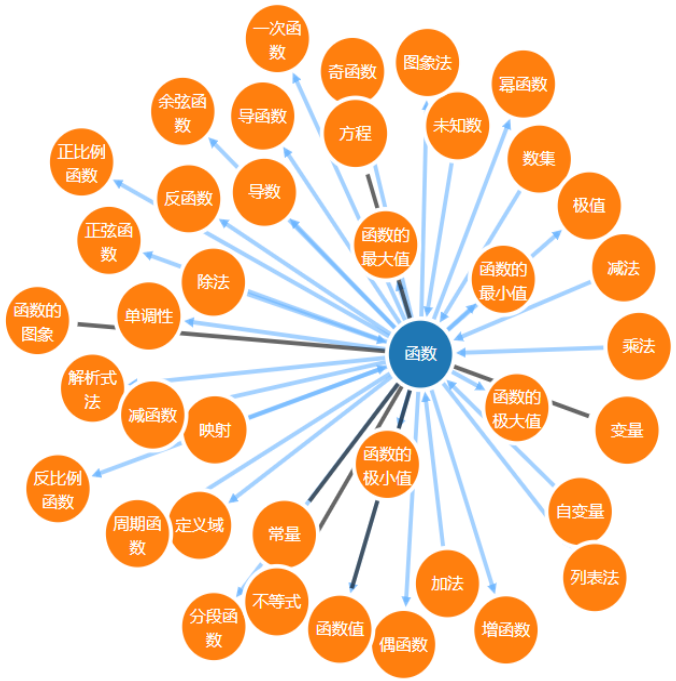


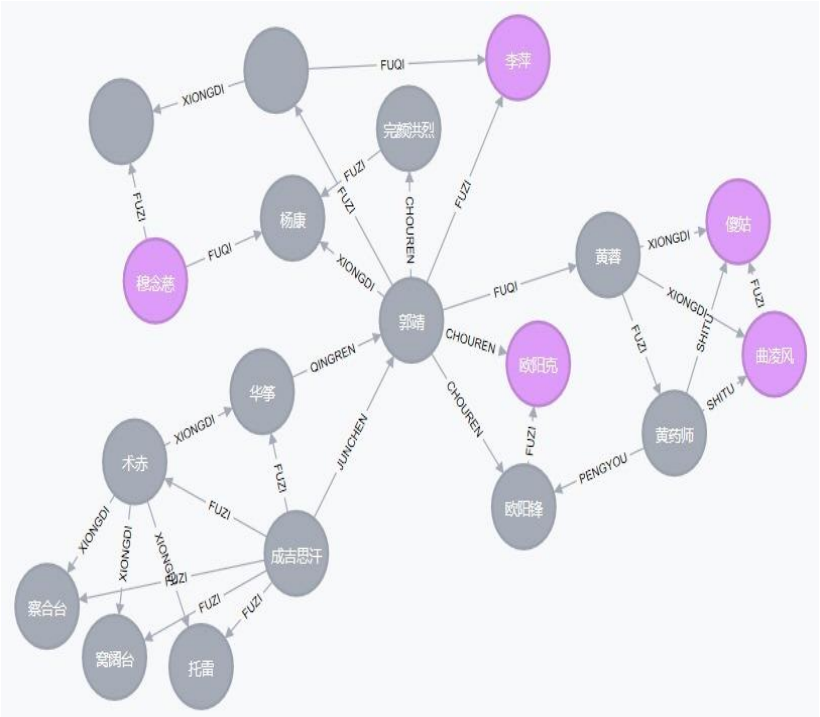
灰度图

当计算机看到一张图像时，它看到的是一堆像素值。
对于左边彩色图，它将看到一个960*1280 *3的数组，3指代的是RGB三个通道；
对于右边灰度图，它将看到一个960*1280的数组。
其中，每个数字的值从0到255不等，其描述了对应那一点的像素灰度。
所以，计算机对图像做处理时，实际上就是对这些数组中的像素值做处理。

数据存储：图形(Graph)数据存储

- 图数据（关系型数据）
  - 社交网络
  - 知识图谱
  - 推荐系统
  - 欺诈检测
  - …



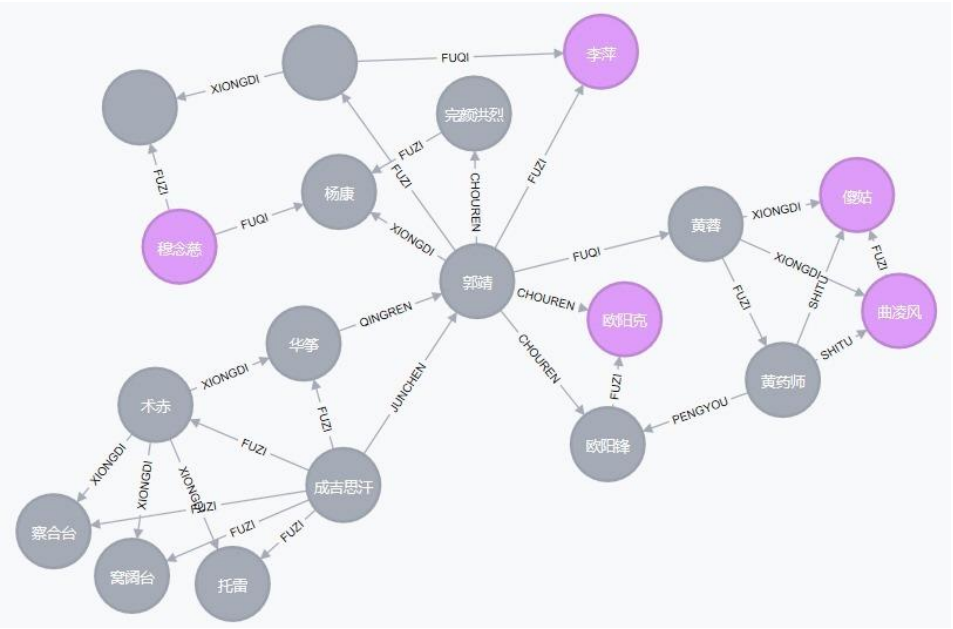


数据存储：图形(Graph)数据存储

- 如何表示图数据？
- SQL
  - 冗余数据
  - 大量空白

名字	子女	兄弟	父亲	年龄
郭靖	破虏	杨康		33
杨康	\	郭靖	完颜洪烈	
破虏	\	\	郭靖	10

- Neo4J
  - 以“图形”的形式存储数据，符合图数据特性
  - 方便删改



Create(n:person{name:”郭靖”,age:”33” })
Create(m:person{name:”破虏”,age:”10” })
create (n)-[:R{type:"父子"}]->(m)

cypher是neo4j官网提供的声明式查询语言

数据存储：图形(Graph)数据存储

- 删去郭靖

SQL

名字	子女	兄弟	父亲	年龄
~~郭靖~~	~~破虏~~	~~杨康~~		~~33~~
杨康	\	~~郭靖~~	完颜洪烈	
破虏	\	\	~~郭靖~~	10

Neo4j

match(n:person{name:”郭靖”}) delete n

- 杨康和破虏是什么关系？

名字	子女	兄弟	父亲	年龄
郭靖	破虏	杨康		33
杨康	\	郭靖	完颜洪烈	
破虏	\	\	郭靖	10

match (n:Person{name:“杨康"}),
(m:Person{name:”破虏”}) ,
r=(n)-[]-(m) ,
return n,m,r

数据存储：图形(Graph)数据存储

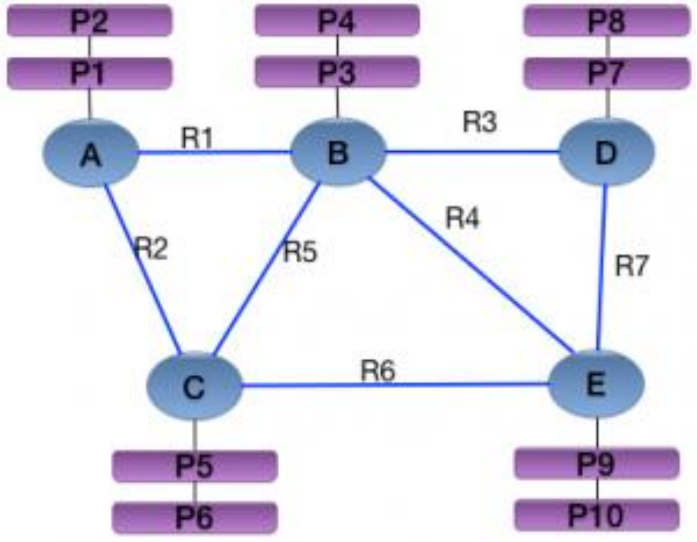
- Neo4J vs MySQL性能对比
  - 具有关系的数据
  - 例：在一个社交网络中找到深度为5的朋友，数据集约为100万人，平均每人50个朋友

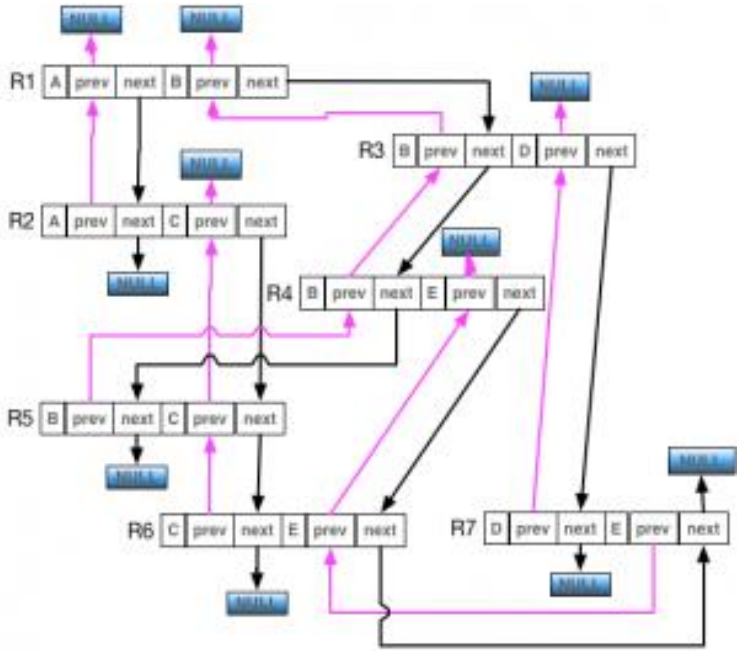
实验结果如下:

深度	MySQL执行时间(s)	Neo4J执行时间(s)	返回记录数
2	0.016	0.01	~2500
3	30.267	0.168	~110 000
4	1543.505	1.359	~600 000
5	未完成	2.132	~800 000

数据存储：图形(Graph)数据存储

- 为什么Neo4j适合存储图数据？
  - **存储结构和数据查询方式，均基于图论**
  - 使用指针遍历所有节点及关系
  - 符合节点+关系的图结构





直接在点和边中保存相应点/边/属性的物理地址，直接进行寻址的遍历方法，免去了基于索引进行扫描查找的开销，实现从O(logn)到O(1)的性能提升。这种图处理方式就叫做**免索引邻接**。它不会随着图数据量的增加影响。仅跟遍历所涉及的数据集有关。

数据存储：键值数据库

- 键值数据库：Redis
  - 数据必须存储在某个key下
  - 数据可以是：
    - 整数
    - 字符串
    - 列表
    - 哈希表
    - 二进制
  - 只提供数据的基本操作
- 因为简单，所以快！

Key Value

```
SET counter 10
INCR counter => 11
GET counter => 11

RPUSH names “taylor”
RPUSH names “swift”
LLEN names => 2
LPOP names => “taylor”

HSET user:1000 name "John"
HSET user:1000 password "s3cret”
HGET user:1000 name => "John"
```

第一部分是存储数字（设置，自增，获取）
第二部分是存储队列（入队，获取长度，出队）
第三部分是存储字典（设置值、获取值）

数据存储：键值数据库

- Redis (REmote Dlctionary Server)
  - key-value存储系统
- 应用场景
  - 发布订阅，地图信息，计数器等
  - 电商“秒杀”：
    - 短时间内极大量访问
    - 避免“超抢”、“超卖”

```
HMGET id Total Booked
HINCRBY id Booked 1
```

库存总量
订单总量

```
"goodsId" : {
    "Total": 100
    "Booked": 100
}
```

下单

秒杀成功/失败

数据存储：NoSQL

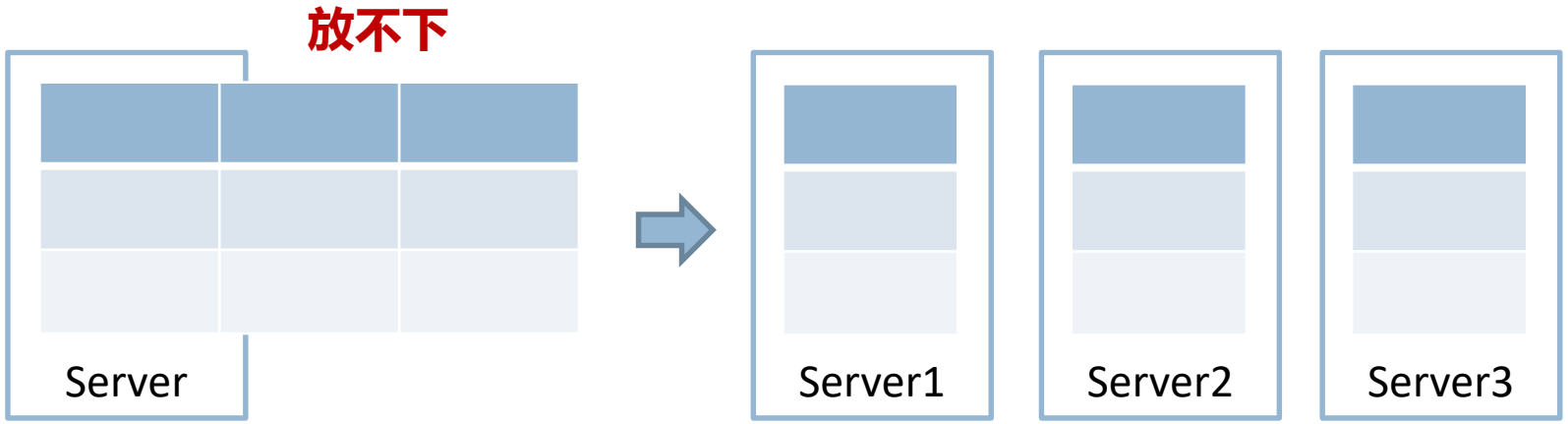
- NoSQL（Not Only SQL），泛指非关系型的DBMS

分类	举例	典型应用场景	数据模型	优点	缺点
文档型数据库	**MongoDb** CouchDB	Web应用（与Key-Value类似，Value是结构化的，不同的是数据库能够了解Value的内容）	文档形式存储，类似JSON的KV存储 Value：简单类型(字符串)，复杂类型（表格，对象）	数据结构要求不严格，表结构可变，不需要像关系型数据库一样需要预先定义表结构	查询性能不高，而且缺乏统一的查询语法。
图形(Graph)数据库	**Neo4J,** InfoGrid, Infinite Graph	社交网络，推荐系统等。专注于构建关系图谱	图结构	利用图结构相关算法。比如最短路径寻址，N度关系查找等	经常需对整个图做计算才能得出需要的信息，且这种结构不太好做分布式集群方案
键值（key-value）	**Redis,** Cabinet/Tyrant, Voldemort, Oracle BDB	内容缓存，用户信息等，如会话，日志等。主要用于处理大量数据的高访问负载	Key 指向 Value 的键值对，通常用hash table来实现	查找速度快，可以通过key快速查询到其value。一般来说，存储不管value的格式，照单全收。	数据无结构化，通常只被当作字符串或者二进制数据
列存储数据库	**HBase,** Cassandra	日志，分布式的文件系统，时空数据等	以**列簇式存储**，将同一列数据存在一起	查找速度快，可扩展性强，方便做数据压缩，对针对某一列或者某几列的查询有IO优势。	功能相对局限缺乏统一查询语言

数据存储：列存储数据库

- 列存储数据库：HBase
  - 分布式的，同一列数据存在一起
  - 应对超大量数据（十亿及百亿行）
  - 将数据表分布式存储
  - 原生仅支持对数据表的简单操作（NoSQL）
  - 可通过扩展模块支持：SQL、图查询、Hadoop、搜索引擎等

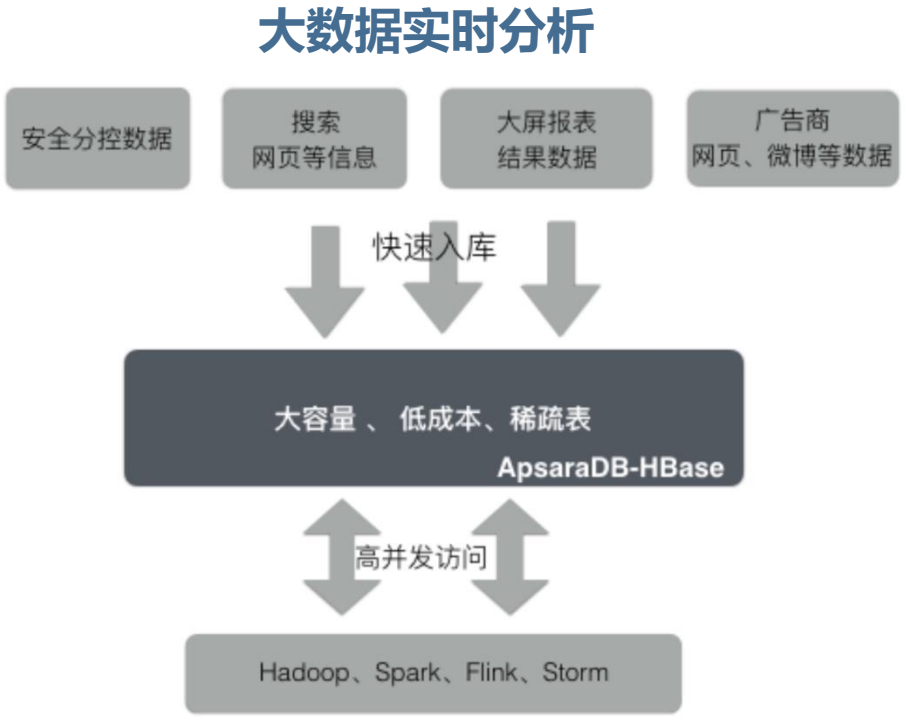
只要数据量传统数据库能够解决，就不用HBase，分布式增加性能损耗。

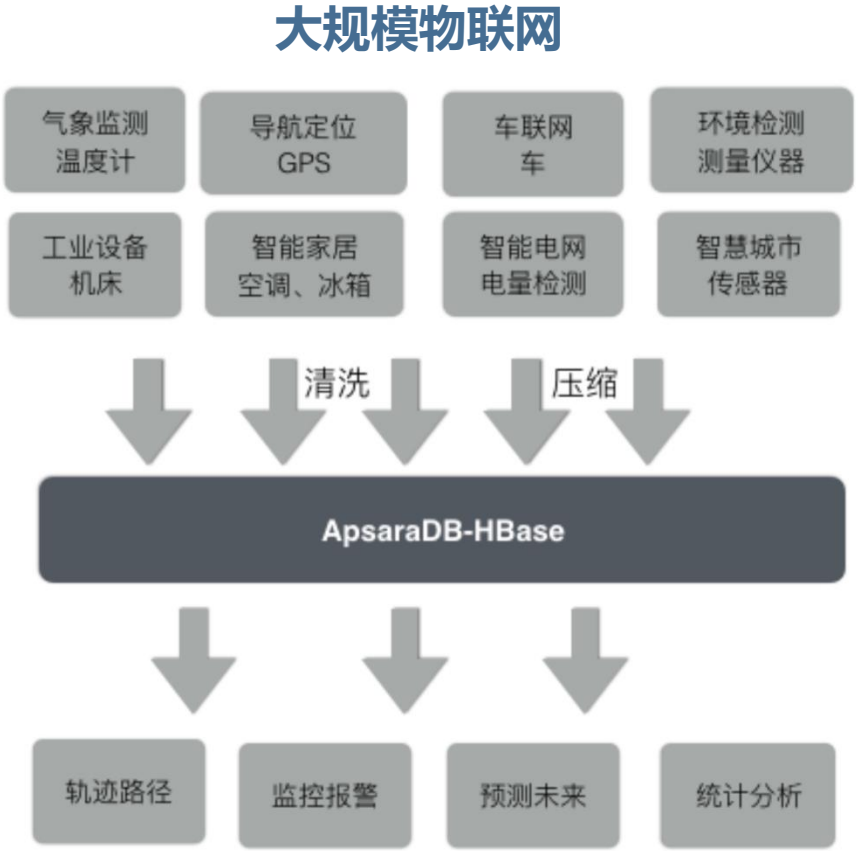


数据存储：列存储数据库

- HBase应用场景
  - 针对需要高容量、高性能的场景

高容量：支持大规模实时数据快速入库
高性能：支持大数据分析应用（如hadoop、spark）高速读取、分析





数据存储：NoSQL

- NoSQL适用于
  - 数据模型比较简单；
  - 需要灵活性更强的IT系统；
  - 对DBMS性能要求较高；
  - 不需要高度的数据一致性；
  - 对于给定key，比较容易映射复杂值的环境
- 比较
  - NoSQL DBMS的产生是为了解决大规模数据集合多重数据种类带来的挑战，尤其是大数据应用难题
  - NoSQL结构比较简单，逻辑控制相对较少，同等存量下数据量超过关系型DBMS，但是处理能力不一定高
  - 对于数据间有固定模式且紧密联系的，还是建议选择关系型DBMS。

数据存储：NoSQL

- NoSQL vs. RDBMS

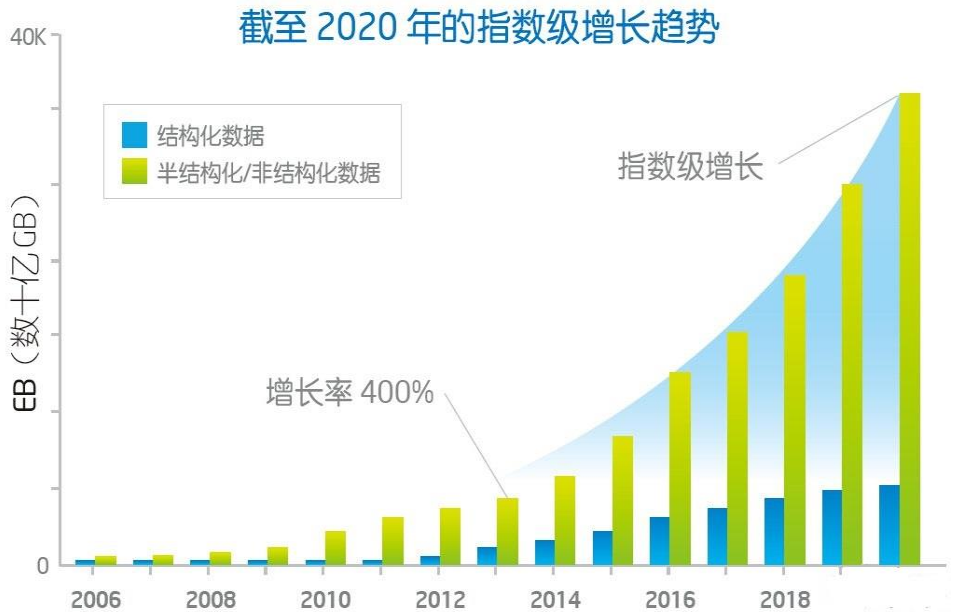
NoSQL	RDBMS
代表着不仅仅是SQL	高度组织化结构化数据
没有声明性查询语言	结构化查询语言（SQL）（SQL）
没有预定义的模式	数据和关系都存储在单独的表中。
键 - 值对存储，列存储，文档存储，图形数据库	数据操纵语言，数据定义语言
最终一致性，而非ACID属性	严格的一致性
非结构化和不可预知的数据	基础事务
CAP定理	
高性能，高可用性和可伸缩性	

数据存储

Beautiful Soup	http://beautifulsoup.readthedocs.io/zh_CN/latest/
MongoDB 教程	http://www.runoob.com/mongodb/mongodb-tutorial.html
CSS选择器教程	http://www.w3school.com.cn/cssref/css_selectors.asp
jsoup教程	http://blog.csdn.net/column/details/jsoup.html
scrapy教程	http://scrapy-chs.readthedocs.io/zh_CN/0.24/intro/tutorial.html

数据分析基础

- 数据采集 Data Collection
- 数据存储 Data Storage
- 数据预处理 Data Preprocessing
- 特征工程 Feature engineering

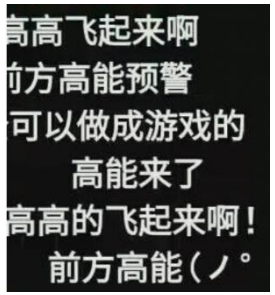


数据预处理

- 大数据环境下的数据特点
- 为什么需要进行预处理
- 预处理的基本方法
  - 数据清理
  - 数据集成
  - 数据变换
  - 数据规约

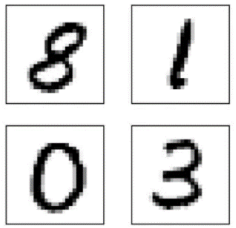
为什么进行数据预处理

- 直接收集的数据通常是“脏的”—数据来源不同
  - 应用需求
    - 微博
    - 淘宝
    - ……
  - 收集手段
    - 传感器，扫描仪
    - 摄像，照相
    - App收集
    - 爬虫写错了
    - ……
  - 数据格式
    - 结构化
    - 半结构化
    - 非结构化
    - ……



ID	时间	内容
陈赫	08-18	天霸
邓超	08-18	我们都很好
邓超	08-18	我也不知道

```
<province>
  <name>黑龙江</name>
  <cities>
    <city>哈尔滨</city>
    <city>大庆</city>
  </cities>
</province>
```



为什么进行数据预处理

- 直接收集的数据通常是“脏的”
  - 不完整
    - 有些数据属性的值丢失或不确定
    - 缺失必要的数据，例：缺失学生成绩
    - ……
  - 不准确
    - 数据错误，属性值错误，例：成绩 = -10
    - 噪声数据：包含孤立（偏离期望）的离群
    - ……
  - 不一致
    - 数据结构有较大差异，例，编码或者命名上存在差异
    - 数据需求改动，例，评价等级:“百分制”与“A, B, C”
    - 存在数据重复和信息冗余现象
    - ……

学 号 Sno	课程号 Cno	成绩 Grade
200215121	1	92
200215121	2	85
200215121	3	88
200215122	2	90
200215122	3	80

为什么进行数据预处理

- 现实世界的数据是“脏的”——举例
  - 滥用缩写词　例：中科大，科大，中国科大，USTC
  - **数据输入错误　例：裤子大、国科大…**
  - 数据中的内嵌控制信息　例：E3=F3*C3
  - 不同的惯用语　例：南七技校
  - **重复记录**
  - **缺失值**
  - 拼写变化与时态，例：propose, proposed, proposing
  - **不同的计量单位**
  - **噪声**
  - **UGC数据，例：弹幕，颜文字(*^▽^*)**

为什么进行数据预处理

- 数据错误的不可避免性
  - 数据输入和获得过程数据错误的不可避免性
  - 数据集成所表现出来的错误
  - 数据传输过程所引入的错误
  - 据统计，有错误的数据占总数据的5%左右

为什么进行数据预处理

- 没有高质量的数据，就没有高质量的结果
  - 高质量的决策必须依赖高质量的数据
    - 例. 数据重复或者缺失将会产生不正确的分析结果，误导决策

- 数据质量的含义
  - 正确性（Correctness）
  - 一致性（Consistency）
  - 完整性（Completeness）
  - 可靠性（Reliability）

- **数据预处理是进行大数据的分析和挖掘的工作中占工作量最大的一个步骤（80%）**

数据预处理

- 大数据环境下的数据特征
- 为什么需要进行预处理
- **预处理的基本方法**
  - **数据清理**
  - 数据集成
  - 数据变换
  - 数据规约

数据预处理：数据清理

- 数据清理的目标
  - 解决数据质量问题
  - 让数据更适合分析、建模
- 数据清理基本任务
  - 处理缺失值
  - 清洗噪声数据
  - 纠正不一致数据
  - 根据需求进行清理
  - ……

ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	C
04	A区	硕士	C	CS	B
05	A区	博士	A	DS	A
…	…	…	…	…	…

ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	**0**
04	A区		C	CS	B
	A区	博士	A	DS	
…	…	…	…	…	…

数据清理-处理缺失值

- 造成数据缺失的原因
  - 信息无法获取，或获取代价大。
    - 反爬虫，加密
  - 信息遗漏
    - 需求不明确
    - 采集故障，存储故障，传输故障
    - 人为因素
  - 数据的某些属性不可用，或不存在（与设计有关）
    - 如：学生的收入，老师的成绩等

数据清理-处理缺失值

- 数据缺失的类型
  - 完全随机缺失：不依赖其他属性/变量，不影响样本的无偏性
  - 随机缺失：缺失与其他完全属性/变量有关系
    - “期末成绩”的依赖于“平时表现”
    - “工资”与“人群背景”的关系
  - 非随机缺失: 数据缺失与属性/变量自身的取值有关
    - 工资问卷

ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	0
04	A区		C	CS	B
	A区	博士	A	DS	
…	…	…	…	…	…

数据清理-处理缺失值

- 处理缺失数据的方法：首先确认缺失数据的影响
  - 数据删除（可能丢失信息，或改变分布）
    - 删除数据
    - 删除属性
    - 改变权重
  - 数据填充
    - 特殊值填充
      - **空值填充，**不同于任何属性值。例，NLP词表补0，DL补mask
      - 样本/属性的均值、中位数、众数填充
    - 使用最可能的数据填充
      - 热卡填充（就近补齐）
      - K最近距离法（KNN）
      - 利用回归等估计方法
      - 期望值最大化方法（EM算法）

模型预测：建立模型预测缺失值

数据清理-处理缺失值

- 热卡填充
  - 完整数据中找到**1个**与它最相似的样本，然后用该样本的值来进行填充
- K最近距离法
  - 根据相关分析(距离)来确定距离缺失数据样本的最近**K**
  - 将这K个值加权平均估计样本缺失数据
- 回归法
  - 基于**数据集**，建立回归模型
  - 将已知属性值代入模型来估计未知属性值，以此预测

ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	**0**
04	A区		C	CS	B
	A区	博士	A	DS	
…	…	…	…	…	…

数据清理-处理缺失值

- 期望值最大化方法（EM算法）
  - 在缺失类型为**随机缺失**的条件下，通过观测数据的边际分布可以对未知参数进行极大似然估计
  - EM算法是一种在不完全数据情况下计算极大似然估计或者后验分布的迭代算法
  - EM算法在每一迭代循环过程中交替执行两个步骤：
    - E步（Excepctaion step, 期望步），在给定完全数据和前一次迭代所得到的隐含参数估计的情况下计算完全数据对应的对数似然函数的条件期望
    - M步（Maximzation step，极大化步），用极大化对数似然函数以确定参数(更新隐含参数)的值，并用于下步的迭代

数据清理–清洗噪声

- 噪声是测量误差的随机部分
  - 包括错误值，或偏离期望的孤立点值
  - 需要对数据进行平滑
- 常用的处理方法
  - 分箱(binning)
    - 利用近邻数据对数据进行平滑
  - 回归(Regression)
    - 让数据适应回归函数来平滑数据
  - 识别离群点，常用聚类方法
    - 监测并且去除孤立点

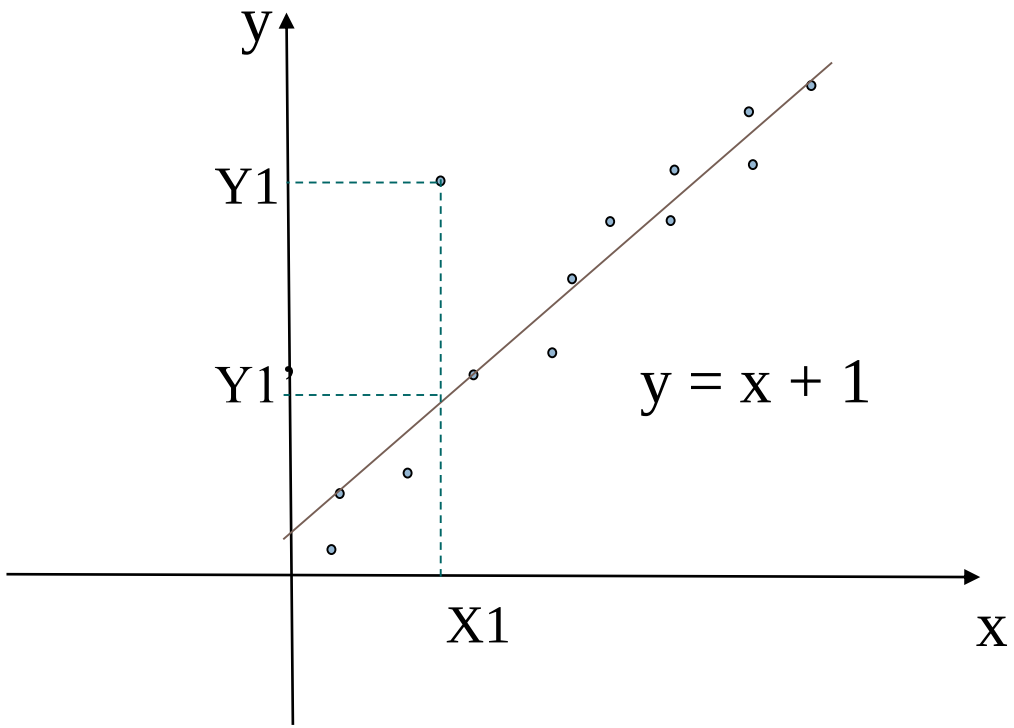
ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	0
04	A区		C	CS	B
	A区	博士	A	DS	
…	…	…	…	…	…

数据清理-清洗噪声

- **分箱：利用近邻数据进行数据平滑**
  - 1. 排序数据，并将他们分到等深的箱中
  - 2. 按箱平均值平滑、按箱中值平滑、按箱边界平滑等**(离散化)**
- 例：排序后数据：4，8，15，21，21，24，25，28，34
  - 1.划分为（等深的）箱
    - 箱1：4，8，15
    - 箱2：21，21，24
    - 箱3：25，28，34
  - 2.用箱平均值平滑
    - 箱1：9，9，9
    - 箱2：22，22，22
    - 箱3：29，29，29
  - 2.用箱边界平滑
    - 箱1：4，4，15
    - 箱2：21，21，24
    - 箱3：25，25，34

数据清理–清洗噪声

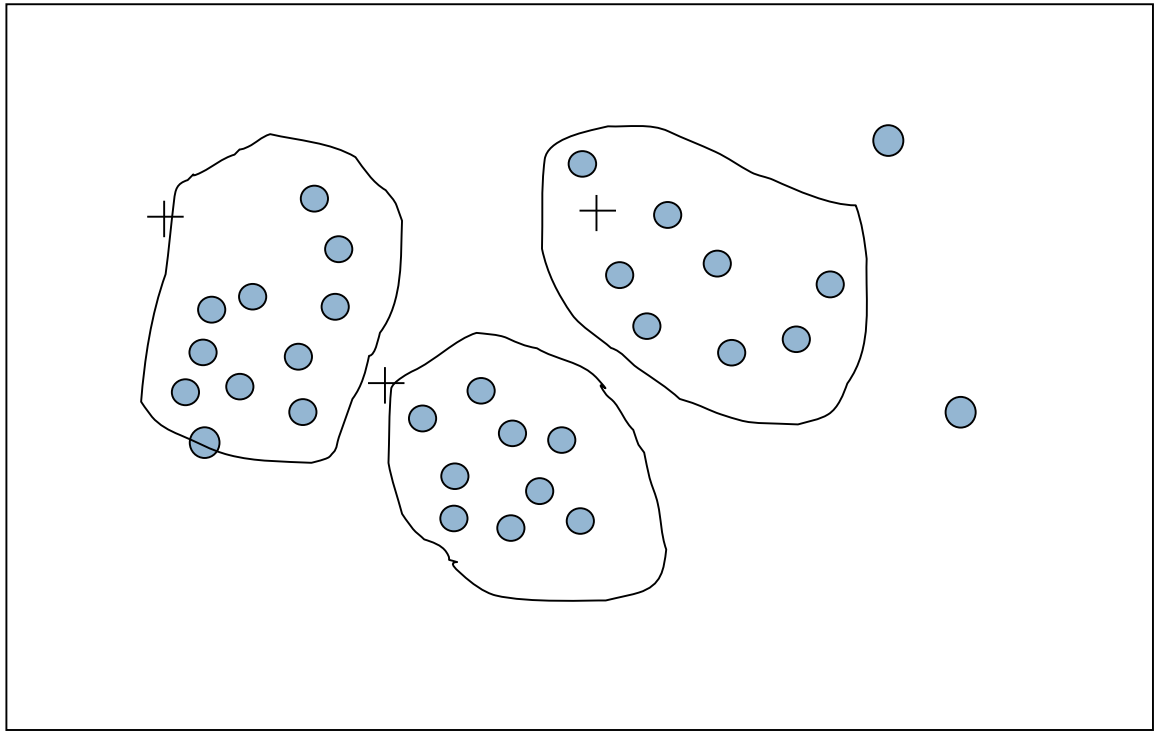
- 回归：让数据适应回归函数来平滑数据
  - 通过线性回归模型，对不符合回归的数据进行平滑处理
  - 用某些属性预测其他属性



ID	住址	学历	单位	专业	收入
01	A区	本科	A	CS	C
02	B区	本科	B	EE	C
03	A区	本科	A	CS	0
04	A区		C	CS	B
	A区	博士	A	DS	
…	…	…	…	…	…

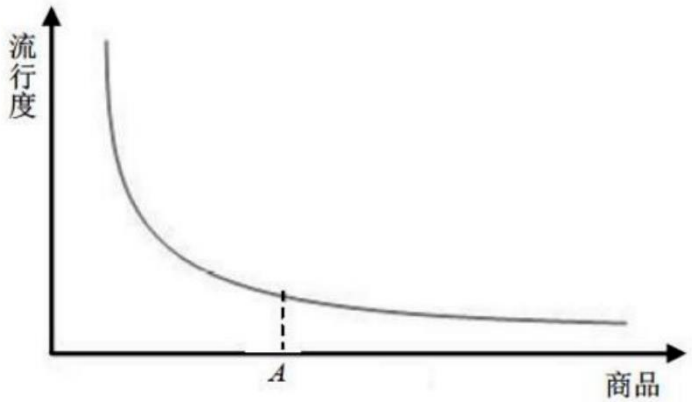
数据清理-清洗噪声

- **识别离群点：聚类分析检测离群点，消除噪声**
  - 聚类将类似的值聚成簇
  - 落在簇集合之外的值被视为离群点



数据清理-根据需求清理数据

- 在特定的应用任务中，根据目标不同，需要特殊的数据清理方法
  - 推荐系统
    - 通用推荐问题
    - 冷启动问题
  - 教育大数据
  - 社交网络
  - POI任务：Point of interest



数据预处理

- 大数据环境下的数据特征
- 为什么需要进行预处理
- **预处理的基本方法**
  - 数据清理
  - **数据集成**
  - 数据变换
  - 数据规约

数据预处理：数据集成

- 数据集成
  - 将多个数据源的数据整合到一个一致的数据存储中
- 数据集成的目标
  - 获得更多的数据
  - 获得更完整的数据
  - 获得更全面的数据画像，如用户画像
- 例：要求电商推荐
  - 用户的购物记录：淘宝，美团，拼多多 等
  - 用户的社交网络：微博，facebook等
  - 用户的视频记录：爱奇艺，抖音等
  - 。。。

数据预处理：数据集成

- 数据集成
  - 将多个数据源的数据整合到一个一致的数据存储中
  - 集成数据（库）时，经常出现冗余数据
    - 冗余数据带来的问题: 浪费存储、重复计算
    - 冗余的属性
    - 冗余的样本
  - 例如：
    - 用户的电商记录出现在很多app中
    - 用户的个人信息在多个app中
    - 。。。

数据预处理：数据集成

- 检测冗余属性
  - 分析属性之间的相关性
  - 相关性分析检测冗余

$$r_{A,B} = \frac{\sum (A-\bar{A})(B-\bar{B})}{(n-1)\sigma_A \sigma_B}$$

Pearson积矩相关系数，取值范围为 [-1; 1]

- **值大于 0，则属性 A 和 B 是正相关的，值越大相关性越强**
  因此，表明两个属性中有一个可以作为冗余删除
- **值为 0，则 A 和 B 是独立的，它们不存在相关性**
- **值小于 0，则 A 和 B 是负相关的。**

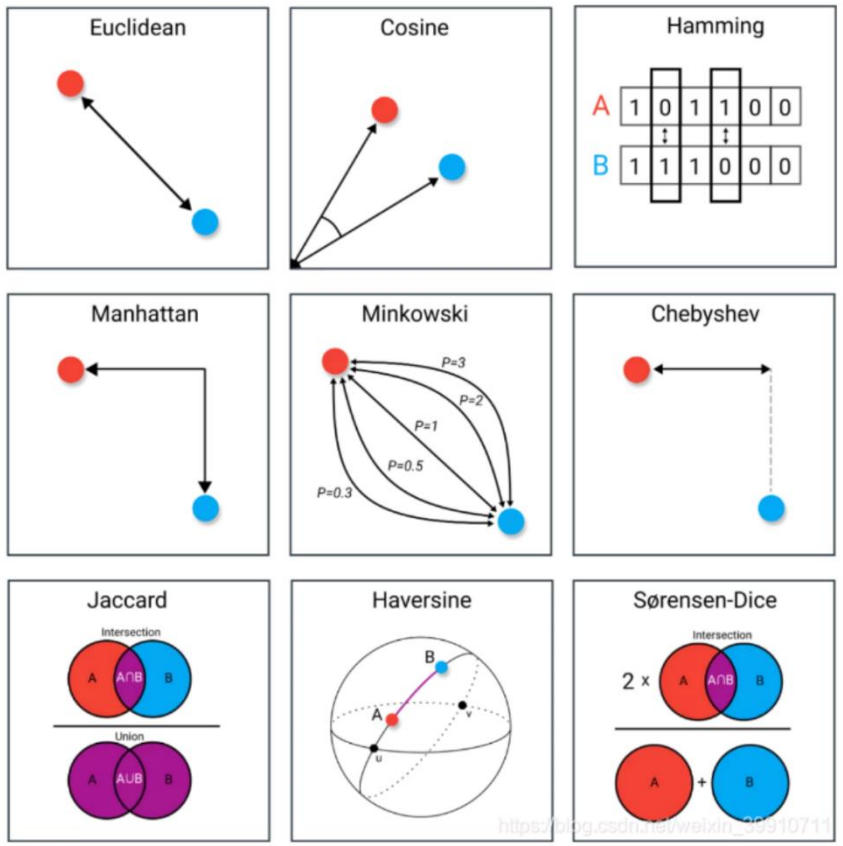
  - 卡方检验：值越大，两个变量相关的可能性越大

$$\chi^2 = \sum_{i=1}^{c}\sum_{j=1}^{r}\frac{(o_{ij}-e_{ij})}{e_{ij}}$$

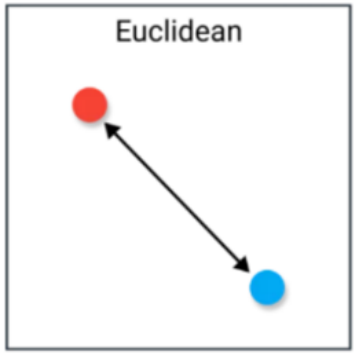
卡方检验：*oij* 是联合事件 (*Ai; Bj*) 的观测频度（即实际计数），而 *eij* 是 (*Ai; Bj*)的期望频度。 卡方检验的原假设是 A 和 B 两个属性相互独立，如果可以拒绝该原假设，则我们说 A 和 B 是显著相关的。

数据预处理：数据集成

- 检测冗余样本
  - 思想：数据样本之间的相关性，数据融合、去除冗余
  - 方法：距离度量
    - 欧几里得距离
    - 曼哈顿距离
    - 汉明距离
    - 明氏距离
    - ……
  - 方法：相似度计算
    - 余弦相似度
    - Jaccard相似度
    - ……

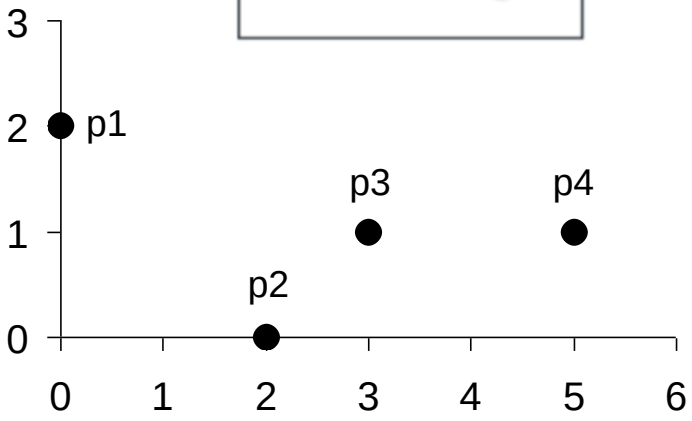


数据预处理：数据集成



- 数据的距离度量
  - 欧几里得距离(Euclidean Distance)

$$dist = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}$$



n 表示数据p和q维度数
p_k 和 q_k 表示数据p和q的第k个属性

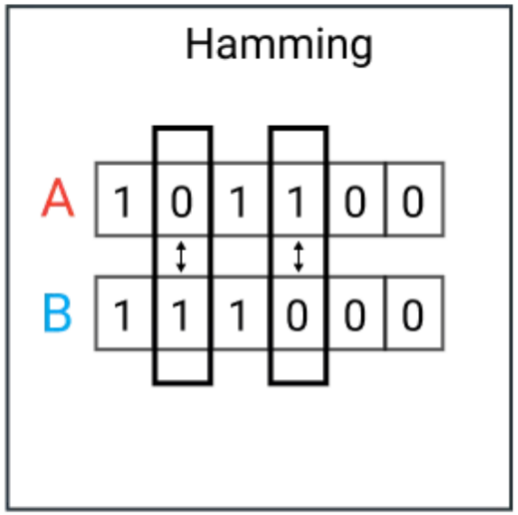
	p1	p2	p3	p4
p1	0	2.828	3.162	5.099
p2	2.828	0	1.414	3.162
p3	3.162	1.414	0	2
p4	5.099	3.162	2	0

point	x	y
p1	0	2
p2	2	0
p3	3	1
p4	5	1

对数据标准化非常重要

数据预处理：数据集成

- 数据的距离度量
  - 汉明距离(Hamming Distance)
  - 两个向量之间不同值的个数。
    - 字符串比较：比较两个相同长度的二进制字符串
  - 要求：向量长度相同

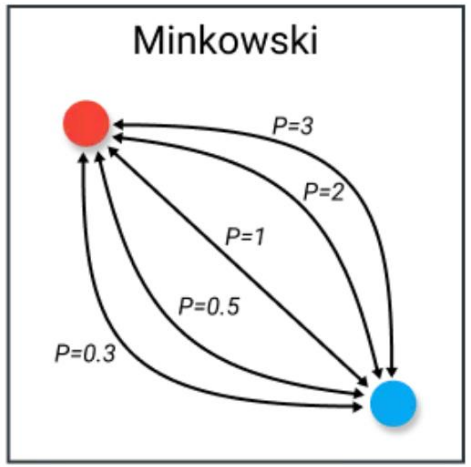


Defu Lian, Haoyu Wang, Enhong Chen, Xing Xie. LightRec: a Memory and Search-Efficient Recommender System. WWW 2020.

数据预处理：数据集成

- 数据的距离度量
  - 明氏距离（Minkowski Distance）
    - 距离度量：通用表达形式

$$dist = \left(\sum_{k=1}^{n} | p_k - q_k |^r \right)^{\frac{1}{r}}$$



r 是参数

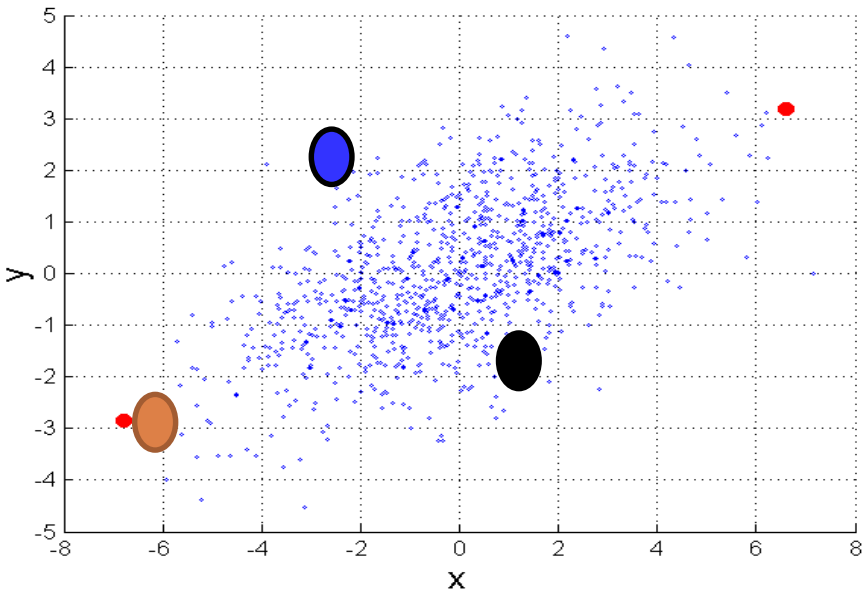
n 表示数据p和q维度数，p_k 和 q_k 表示数据p和q的第k个属性

- r=1：曼哈顿距离
- r=2：欧氏距离
- r=∞：切比雪夫距离

数据预处理：数据集成

- 数据的距离度量
  - 马氏距离：数据的协方差距离
    - 欧氏距离的扩展，考虑到各种特性之间的联系（协方差）

$$mahalanobis(p,q) = (p-q)\Sigma^{-1}(p-q)^T$$



Σ 是总体样本 X的协方差矩阵

$$\Sigma_{j,k} = \frac{1}{n-1}\sum_{i=1}^{n}(X_{ij}-\overline{X}_j)(X_{ik}-\overline{X}_k)$$

- 确定未知样本集与已知样本集的相似度
- 它考虑了数据集的相关性，并且是比例不变的

红色的数据点, 欧氏距离为14.7, 马氏距离为6

数据预处理：数据集成

- 马氏距离 vs 欧氏距离
  - **假设：**以厘米为单位测量人的身高，以克（g）为单位测量人的体重。每个人被表示为一个两维向量。如：一个人身高173cm，体重50000g，表示为（173, 50000），根据身高体重来判断人的体型的相似程度
  - **已知：**小明(160, 60000)；小王(160, 59000)；小李(170，60000)。小明与谁的体型更相似？

分析：根据常识可以知道小明和小王体型相似。但是如果根据欧氏距离来判断，小明和小王的距离要远大于小明和小李之间的距离，即小明和小李体型相似

原因：不同特征的度量标准之间存在差异而导致判断出错

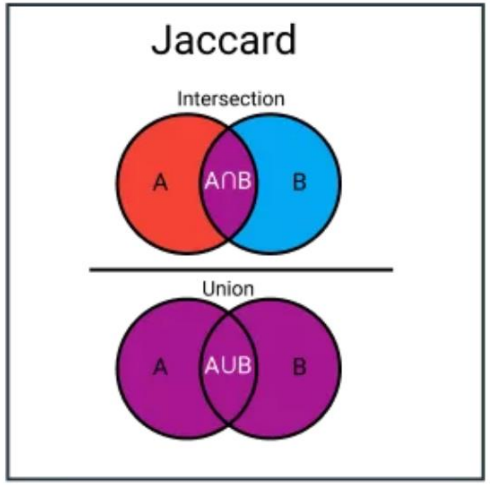
- 以克（g）为单位测量人的体重，数据分布比较分散，即方差大，
- 以厘米为单位来测量人的身高，数据分布就相对集中，方差小

马氏距离把方差归一化，使得特征之间的关系更加符合实际情况。

数据预处理：数据集成

- 数据的相似度计算
  - **简单匹配 Simple Matching VS Jaccard相关系数**
  - 离散数据，属性的取值表示为0或1
  - 例：数据p和q，定义如下4个变量
    - F01：p为0，q为1的属性数量
    - F10：p为1，q为0的属性数量
    - F00：p为0，q为0的属性数量
    - F11：p为1，q为1的属性数量

$$p = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)$$
$$q = (0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1)$$



SMC = number of matches / number of attributes
= (F11 + F00) / (F01 + F10 + F11 + F00)

Jaccard = number of 11 matches / number of non-zero attributes
= (F11) / (F01 + F10 + F11)

数据预处理：数据集成

- 数据的相似度计算
  - **简单匹配 Simple Matching VS Jaccard相关系数**

$p = (1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0)$

$q = (0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1)$

p和q是否相关?

F01 = 2 (p为0，q为1的属性数量)

F10 = 1 (p为1，q为0的属性数量)

F00 = 7 (p为0，q为0的属性数量)

F11 = 0 (p为1，q为1的属性数量)

$$SMC = (F11 + F00) / (F01 + F10 + F11 + F00)$$

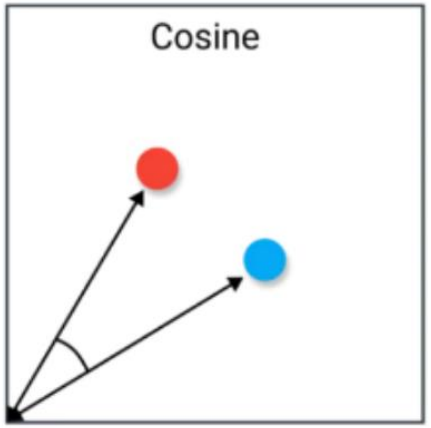
$$= (0+7) / (2+1+0+7) = 0.7$$

$$Jaccard = (F_{11}) / (F_{01} + F_{10} + F_{11}) = 0 / (2 + 1 + 0) = 0$$

数据预处理：数据集成

- 数据的相似度计算
  - **余弦相似性 (Cosine Similarity)**

$$\cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}.$$



  - **例：**

A = 3 2 0 5 0 0 0 2 0 0
B = 1 0 0 0 0 0 0 1 0 2

$A \bullet B = 3*1 + 2*0 + 0*0 + 5*0 + 0*0 + 0*0 + 0*0 + 2*1 + 0*0 + 0*2 = 5$

$||A|| = (3*3+2*2+0*0+5*5+0*0+0*0+0*0+2*2+0*0+0*0)^{0.5} = (42)^{0.5} = 6.481$

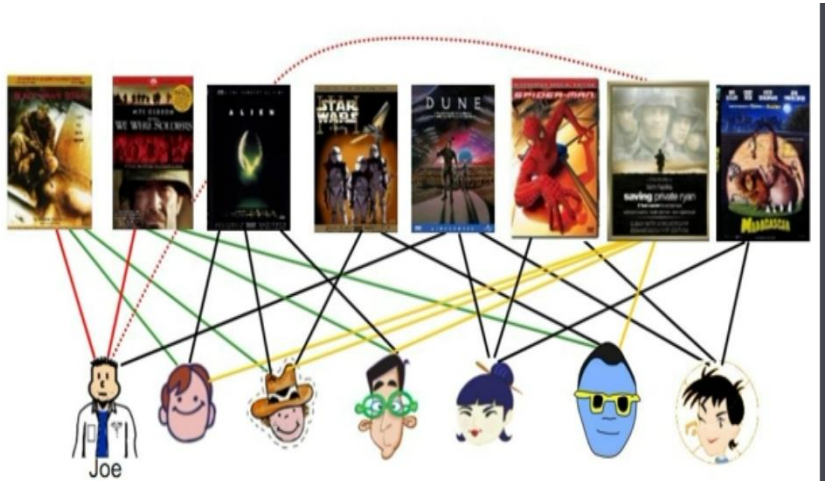
$||B|| = (1*1+0*0+0*0+0*0+0*0+0*0+0*0+1*1+0*0+2*2)^{0.5} = (6)^{0.5} = 2.245$

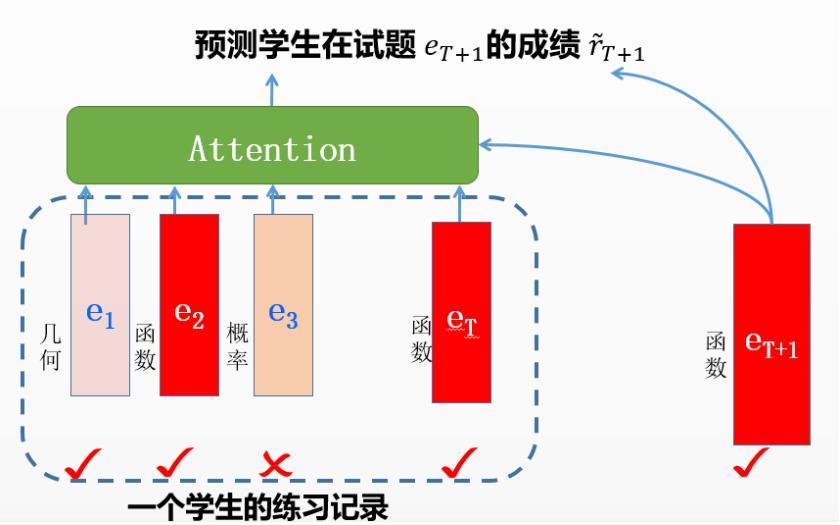
$\cos(A, B) = 0.3150$

思考：余弦相似度是不是一种距离？

数据预处理：数据集成

- 数据的相似度计算
  - **余弦相似度 (Cosine Similarity)**
    - 推荐系统中，协同过滤算法(UCF, ICF)—经典算法
      - 用户(向量)的相似度度量，产品(向量)的相似度度量
    - 深度学习中，训练Attention（注意力机制）的权重
      - 基于注意力机制的学生成绩预测模型





Qi Liu, Zhenya Huang, Enhong Chen,, EKT: Exercise-aware Knowledge Tracing for Student Performance Prediction, TKDE

数据预处理：数据集成

- 数据的相关性分析
  - Pearson相关系数
    - 衡量两个数据对象之间的线性关系
    - 数据标准化
  - 可以简单理解为：p和q的协方差/(p的标准差*q的标准差)

$$p_{X,Y} = \frac{\sum_{i=1}^{n}(X_i - \tilde{X})(Y_i - \tilde{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \tilde{X})^2 \sum_{i=1}^{n}(Y_i - \tilde{Y})^2}}$$

$$\rho_{X,Y} = \mathrm{corr}(X,Y) = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} = \frac{E[(X-\mu_X)(Y-\mu_Y)]}{\sigma_X \sigma_Y}$$

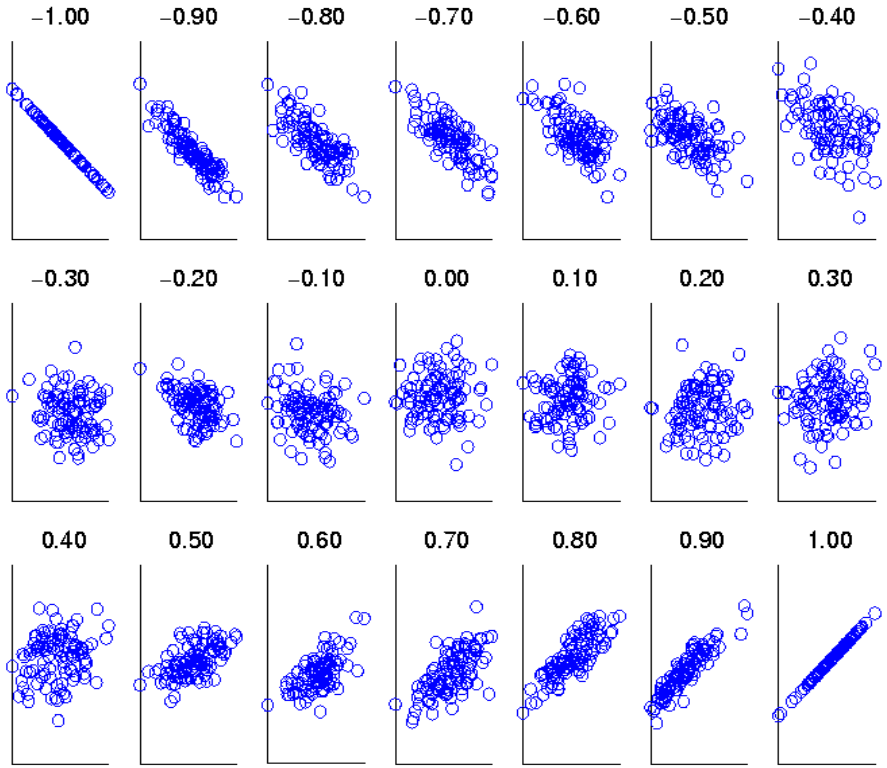
Zhenya Huang, Qi Liu, Enhong Chen, et al, Question Difficulty Prediction for READING Problems in Standard Tests, AAAI’2017

数据预处理：数据集成

- 数据的相关性
  - Pearson相关系数：衡量数据对象之间的线性关系

散点图显示相似度[-1, 1]

两个数据样本x,y各有30个属性，这些属性值随机产生，使得x和y的相关度从-1到1。图中每个小圆圈代表30个属性中的一个，其x坐标是x的一个属性的值，y坐标是y的相同属性的值



数据预处理：数据集成

- 数据的相关性分析
  - Pearson相关系数：衡量数据对象之间的线性关系

$$p_{X,Y} = \frac{\sum_{i=1}^{n}(X_i - \tilde{X})(Y_i - \tilde{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \tilde{X})^2 \sum_{i=1}^{n}(Y_i - \tilde{Y})^2}}$$

- 例：　　**问：X与Y有没有关系？**
  - X = (-3, -2, -1, 0, 1, 2, 3)
  - Y = (9, 4, 1, 0, 1, 4, 9)

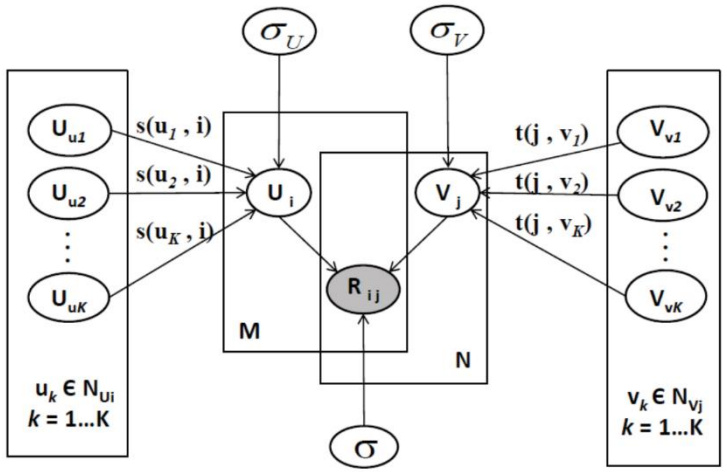
- Mean(X) = 0, Mean(Y) = 4
- Correlation= ？
  - = (-3)(5)+(-2)(0)+(-1)(-3)+(0)(-4)+(1)(-3)+(2)(0)+3(5)= 0

数据预处理：数据集成

- 数据的相关性分析
  - 有时，不同的属性产生的影响不同
  - 在计算距离，相似度时，可以赋予数据属性的权重不同 (w_k)

$$\text{similarity}(\mathbf{x},\mathbf{y}) = \frac{\sum_{k=1}^{n} \boxed{w_k} \delta_k s_k(\mathbf{x},\mathbf{y})}{\sum_{k=1}^{n} \delta_k}$$

$$d(\mathbf{x},\mathbf{y}) = \left(\sum_{k=1}^{n} \boxed{w_k} |x_k - y_k|^r \right)^{1/r}$$



Le Wu, Enhong Chen, Qi Liu, Leveraging Tagging for Neighborhood-aware Probabilistic Matrix Factorization. CIKM2012

数据预处理：数据集成

数据的相关性分析

- **无序数据：每个数据样本的不同维度是没有顺序关系的**
  - 余弦相似度、相关度、欧几里得距离、Jaccard
- **有序数据：对应的不同维度(如特征)是有顺序(rank)要求的**
  - 在信息检索中，如何判断不同检索方法返回的页面序列的优劣
  - 在推荐系统中，如何判断不同推荐序列的好坏
    - **Spearman Rank(斯皮尔曼等级)相关系数**
    - **归一化的折损累计增益(NDCG)**
    - 肯德尔相关性系数
      - kendall correlation coefficient
- 课外阅读：PageRank算法

i	相关度
1	3
2	3
3	2
4	0
5	1
6	2

方法返回结果

i	相关度
1	3
2	3
3	2
4	2
5	1
6	0

真实结果

数据预处理：数据集成

数据的相关性分析—举例

- 已知：6个网页的相关度是3，2，3，0，1，2，所以在信息检索中，最好的返回结果应当如(a)所示。
- 如果我们设计了两个检索算法，返回结果分别是(b)和(c)，请问哪个方法的结果与真实结果更相似？

i	相关度
1	3
2	3
3	2
4	2
5	1
6	0

(a)真实结果

i	相关度
1	3
2	3
3	0
4	2
5	2
6	1

(b)方法1返回结果

i	相关度
1	3
2	3
3	2
4	0
5	2
6	1

(c)方法2返回结果

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **Spearman Rank(斯皮尔曼等级)相关系数**
    - 比较两组变量的相关程度
    - 当关系是非线性时，它是两个变量之间关系评价的更好指标

$$\rho_s = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

    - ρ_s：表示斯皮尔曼相关系数
    - d_i^2：表示每一对样本之间等级的差
    - N：表示n样本容量
  - ρ_s的范围：-1 to 1 (正相关:$\rho_s > 0$，负相关:$\rho_s < 0$，不相关:$\rho_s = 0$)

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **Spearman Rank(斯皮尔曼等级)相关系数**
  - X = (a, b, c, d, e, f)
  - Y = (c, a, e, d, f, b)

$$d_i = Y_i - X_i$$

$$\rho_s = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

  - $d_i^2 = (4, 1, 4, 0, 1, 16)$
  - $\rho = 1 - \frac{6(26)}{6(36-1)} \approx 1 - 0.743 = 0.257$

数据预处理：数据集成

数据的相关性分析

- **无序数据：每个数据样本的不同维度是没有顺序关系的**
  - 余弦相似度、相关度、欧几里得距离、Jaccard
- **有序数据：对应的不同维度(如特征)是有顺序(rank)要求的**
  - 在信息检索中，如何判断不同检索方法返回的页面序列的优劣
  - 在推荐系统中，如何判断不同推荐序列的好坏
    - **Spearman Rank(斯皮尔曼等级)相关系数**
    - **标准化的折损累计增益(NDCG)**
    - 肯德尔相关性系数
      - kendall correlation coefficient
- 课外阅读：PageRank算法

i	相关度
1	3
2	3
3	2
4	0
5	1
6	2

方法返回结果

i	相关度
1	3
2	3
3	2
4	2
5	1
6	0

真实结果

数据预处理：数据集成

数据的相关性分析—举例

- 已知：6个网页的相关度是3，2，3，0，1，2，所以在信息检索中，最好的返回结果应当如(a)所示。
- 如果我们设计了两个检索算法，返回结果分别是(b)和(c)，请问哪个方法的结果与真实结果更相似？

i	相关度
1	3
2	3
3	2
4	2
5	1
6	0

(a)真实结果

i	相关度
1	3
2	3
3	0
4	2
5	2
6	1

(b)方法1返回结果

i	相关度
1	3
2	3
3	2
4	0
5	2
6	1

(c)方法2返回结果

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **Spearman Rank(斯皮尔曼等级)相关系数**
    - 比较两组变量的相关程度
    - 当关系是非线性时，它是两个变量之间关系评价的更好指标

$$\rho_s = \frac{\sum_{i=1}^{N}(R_i-\bar{R})(S_i-\bar{S})}{\left[\sum_{i=1}^{N}(R_i-\bar{R})^2 \sum_{i=1}^{N}(S_i-\bar{S})^2\right]^{\frac{1}{2}}} \Rightarrow \rho_s = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

    - ρ_s：表示斯皮尔曼相关系数
    - d_i^2：表示每一对样本之间等级的差
    - N：表示n样本容量
  - ρ_s的范围：-1 to 1 (正相关:$\rho_s > 0$，负相关:$\rho_s < 0$，不相关:$\rho_s = 0$)

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **Spearman Rank(斯皮尔曼等级)相关系数**
  - X = (a, b, c, d, e, f)
  - Y = (c, a, e, d, f, b)

$$d_i = Y_i - X_i$$

  - $d_i^2 = (4, 1, 4, 0, 1, 16)$
  - $\rho = 1 - \frac{6(26)}{6(36-1)} \approx 1 - 0.743 = 0.257$

$$\rho_s = 1 - \frac{6\sum d_i^2}{n(n^2-1)}$$

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **NDCG(Normalized Discounted cumulative gain)**
    - **CG(累计增益)：** 只考虑到了相关性的关联程度，没有考虑每个推荐结果处于不同位置对整个推荐效果的影响

$$CG_k = \sum_{i=1}^{k} rel_i$$

rel_i表示处于位置 ii 的推荐结果的相关性

    - **DCG(折损累计增益)：** 就是在每一个CG的结果上处以一个折损值，目的就是为了让排名越靠前的结果越能影响最后的结果

$$DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i}-1}{log_2(i+1)}$$

    - i表示推荐结果的位置， i越大， 则推荐结果在推荐列表中排名越靠后推荐效果越差，DCG越小

Qi Liu, Yong Ge, Enhong Chen, and Hui Xiong. Personalized Travel Package Recommendation. ICDM'2011, (Best Research Paper Award)

数据预处理：数据集成

- **有序数据的距离度量(信息检索、推荐系统等)**
  - **NDCG(** Normalized Discounted cumulative gain **)**
    - **NDCG：**由于搜索结果随着检索词的不同，返回的数量不一致，而DCG是一个累加的值，没法针对两个不同的搜索结果进行比较，因此需要标准化处理，这里是除以IDCG:

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

IDCG为理想（ideal）情况下最 荐系统为某一用户返回的最好推荐结果列表(或者，真实的数据序列)

数据预处理：数据集成

$$NDCG_k = \frac{DCG_k}{IDCG_k}$$

- 例，假设搜索返回的6个物品，其相关性分别是 3、2、3、0、1、2
  - CG@6 = 3+2+3+0+1+2
  - DCG@6 = 7+1.89+3.5+0+0.39+1.07 = 13.85
- 假如我们实际召回了8个物品，除了上面的6个，还有两个物品，第7个相关性为3，第8个相关性为0。那么在理想情况下的相关性分数排序应该是
  - 3、3、3、2、2、1、0、0。
- 计算IDCG@6：
  - IDCG = 7+4.42+3.5+1.29+1.16+0.36 = 17.73
- 可以计算NDCG@6：
  - NDCG@6 = 13.85/17.73 = 0.78

$$CG_k = \sum_{i=1}^{k} rel_i \qquad DCG_k = \sum_{i=1}^{k} \frac{2^{rel_i}-1}{log_2(i+1)}$$

i	rel
1	3
2	2
3	3
4	0
5	1
6	2

方法返回结果

i	rel
1	3
2	3
3	3
4	2
5	2
6	1

真实结果

数据预处理：数据集成

课后阅读

- Defu Lian, Haoyu Wang, Enhong Chen, Xing Xie. LightRec: a Memory and Search-Efficient Recommender System. WWW 2020.
- Qi Liu, Zhenya Huang, Enhong Chen,, EKT: Exercise-aware Knowledge Tracing for Student Performance Prediction, TKDE
- Zhenya Huang, Qi Liu, Enhong Chen, et al, Question Difficulty Prediction for READING Problems in Standard Tests, AAAI’2017
- Qi Liu, Yong Ge, Enhong Chen, and Hui Xiong. Personalized Travel Package Recommendation. ICDM'2011, (Best Research Paper Award)
- PageRank算法

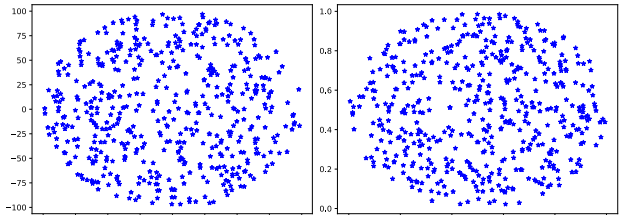
数据预处理

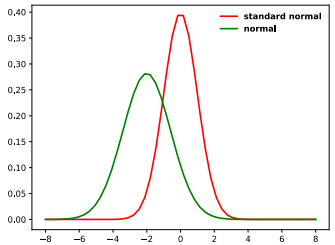
- 大数据环境下的数据特征
- 为什么需要进行预处理
- **预处理的基本方法**
  - 数据清理
  - 数据集成
  - **数据变换**
  - 数据规约

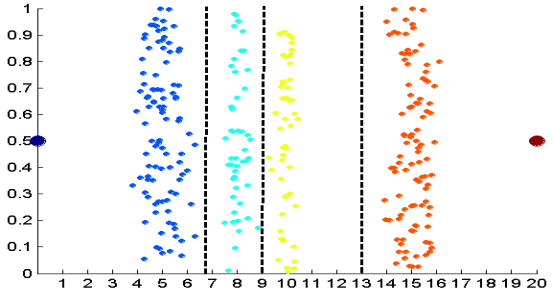
数据预处理：数据变换

131

- 数据变换的目的是将数据转换成适合分析建模的形式
  - 前提条件：尽量不改变原始数据的规律
  - 数据规范化
    - 最小-最大规范化
    - z-score规范化
    - 小数定标规范化
  - 数据离散化
    - 非监督离散化
    - 监督离散化



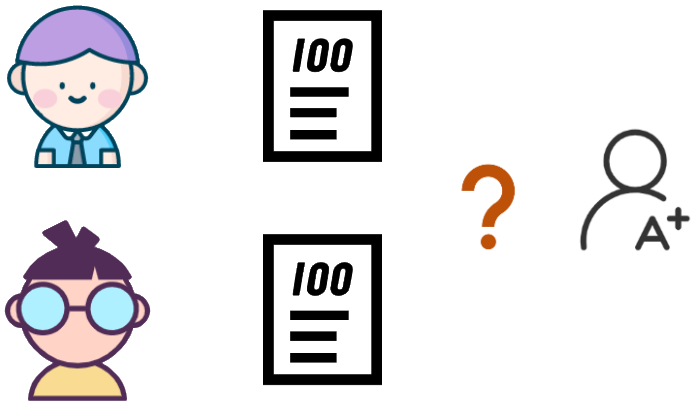




数据预处理：数据变换

132

- 数据规范化
  - 目的：将不同数据（属性）按一定规则进行缩放，使它们具有可比性
  - 例如，我们需要考察学生A和学生B的某门课程成绩。A的考试满分是100分（及格60分），B的考试满分是150分（及格90分）。显然，A和B的100分代表着完全不同的含义。



如何用一个同等的标准来比较A与B的成绩数据呢?

数据变换-规范化

- 最小-最大规范化
  - 对原始数据进行线性变换。把数据A的观察值v从原始的区间[min_A, max_A]映射到新区间 [new_min_A, new_max_A]
    - 0-1规范化又称为归一化

$$v' = \frac{v - min_A}{max_A - min_A}(new_max_A - new_min_A) + new_min_A$$

  - 数理依据：

$$\frac{v' - new_min_A}{new_max_A - new_min_A} = \frac{v - min_A}{max_A - min_A}$$

数据变换-规范化

134

- 最小-最大规范化

$$v' = \frac{v - min_A}{max_A - min_A}(new_max_A - new_min_A) + new_min_A$$

- 例：假设某属性规范化前的取值区间为[－100，100]，规范化后的取值区间为[0，1]，采用最小-最大规范化 66，得

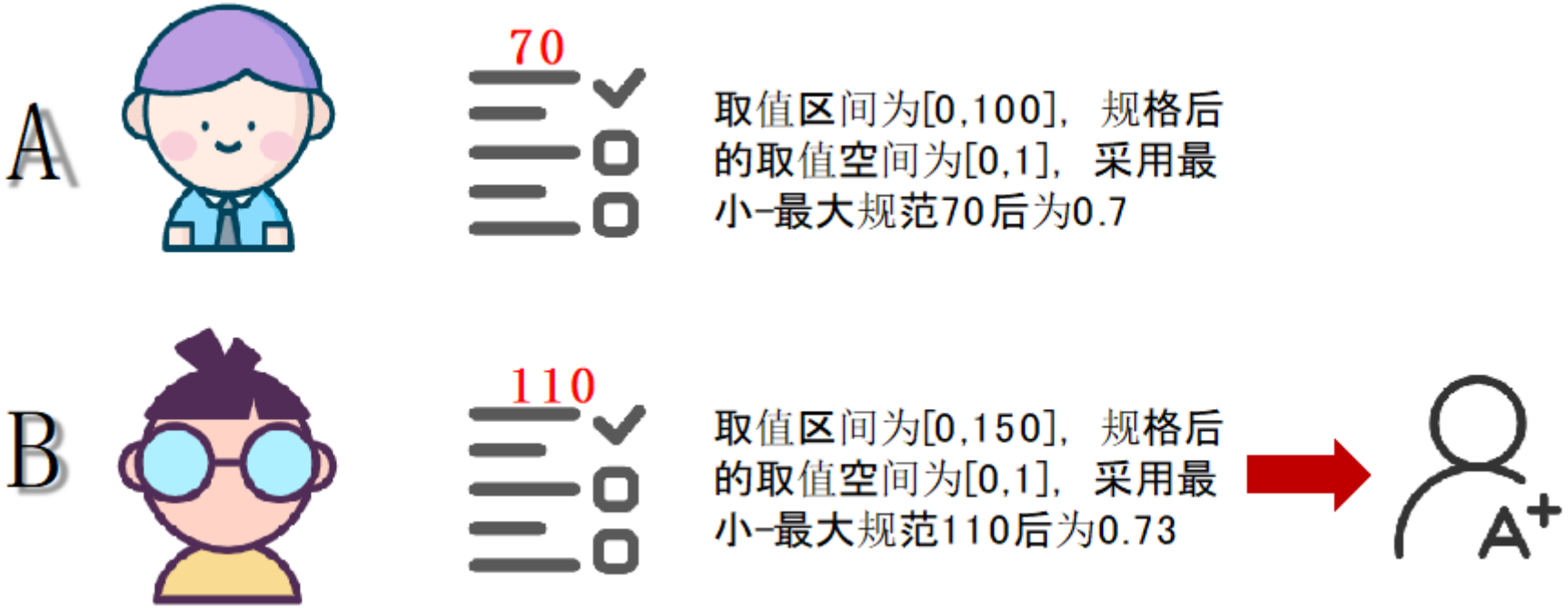
$$v' = \frac{66-(-100)}{100-(-100)}(1-0)+0 = 0.83$$

快速练习：采用最小-最大规范化 -80？

数据变换-规范化

135

假设A的课程成绩为70分（0-100分），B的课程成绩为110分(0-150分)，采用最小-最大规范化来比较A和B的成绩



用最小-最大规范化后得出B的成绩更好

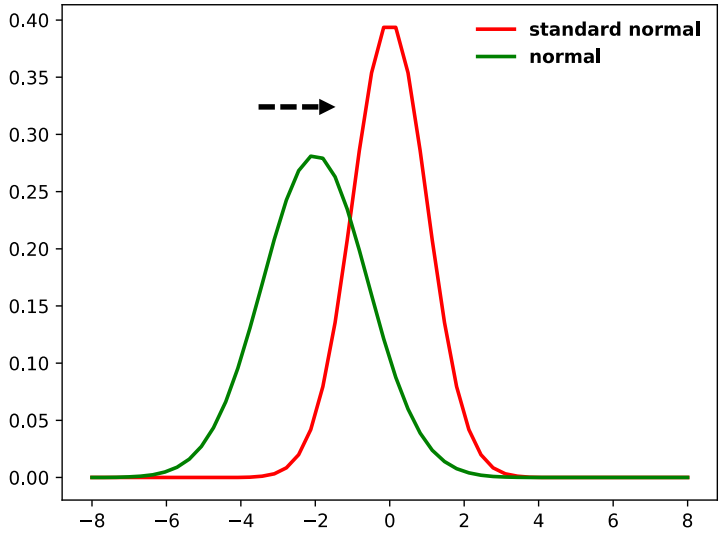
数据变换-规范化

- z-score规范化
  - 最大最小值未知，或者离群点影响较大时，假设数据服从正态分布
    - 某一原始数据(v)与原始均值的差再除以标准差，可以衡量某数据在分布中的相对位置

$$v' = \frac{v - \bar{A}}{\sigma_A}$$

  - 例：假设某属性的平均值、标准差分别为80、25，用z-score规范化 66

$$v' = \frac{66 - 80}{25} = -0.56$$



数据变换-规范化

- z-score规范化
  - 例：假设学生的成绩分布符合正态分布，某素质课考试的平均分为73分，标准差为7分，A得78分；实践课考试的平均分为80分，标准差为6.5分，A得83分。那么A的哪一门考试成绩比较好？

采用z-score规范化得出A的素质课成绩要优于实践课成绩

数据变换-规范化

138

- 小数定标规范化
  - 通过移动小数点的位置来进行规范化。小数点移动多少位取决于属性A的取值中的最大绝对值。

$$v' = \frac{v}{10^j}$$

其中，j 是使 Max($|v'|$)<1的最小整数

  - 比如属性A的取值范围是-999到88，那么最大绝对值为999，小数点就会移动3位，即新数值=原数值/1000。那么A的取值范围就被规范为-0.999到0.088。

数据变换-规范化

139

小结

	优点	缺点	适用场景
最小-最大规范化	保留了原始数据中存在的关系，是消除量纲和数据取值范围影响的最简单方法	对最大最小值敏感，新数据加入时，可能改变最大最小值，需重新计算	适用于原始数据不存在很大/很小的一部分数据的时候
z-score规范化	算法简单方便，结果方便比较，应用于数值型的数据，且不受数据量级的影响	总体平均值和方差不一定可知，在一定程度上要求数据分布，结果没有具体意义，只用于比较	适用于最大最小值未知，或者离群点影响较大的时候
小数定标规范化	算法实现简单	不适用于不同含义数据的比较，无实际意义	使用含义相同的数据，且最大最小相差较大

数据预处理：数据变换

140

- 数据离散化
  - 连续数据过于细致，数据之间的关系难以分析
  - 划分为离散化的区间，发现数据之间的关联，便于算法处理
    - **同学们成绩：**100分制分数使用五分制离散化表示
      - A（大于等于85分），B，C，D，F（小于60分）
    - **人的年龄：**离散化为不同的年龄段（引源自世卫组织）
      - 未成年人：0至17岁；
      - 青年人：18岁至45岁；
      - 中年人：46岁至69岁；
      - 老年人：大于70岁。
    - **一年365天：**离散化表示为12个月份或四个季节

A+	A	A-
B+	B	B-
C+	C	C-
D+	D	D-
F		

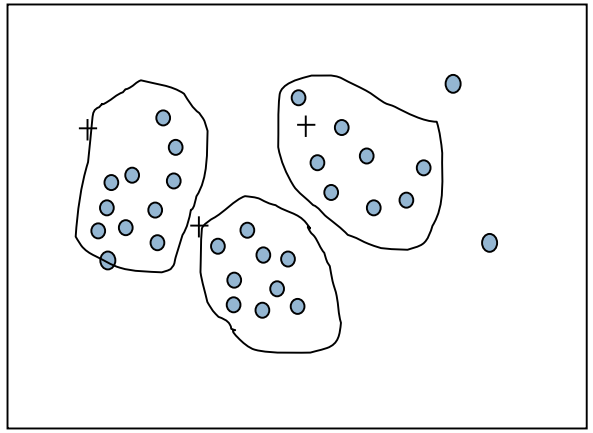
数据变换-离散化

- 数据离散化
  - 连续数据过于细致，数据之间的关系难以分析，将其分段为离散化的区间，发现数据之间的关联，便于算法处理
  - 非监督离散化（无类别信息）
  - 有监督离散化（有类别信息）

数据变换-离散化

142

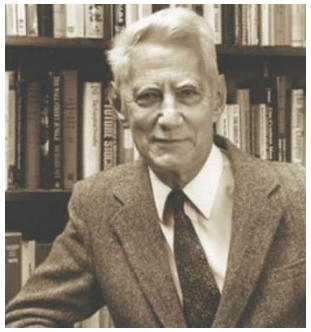
- 非监督离散化（参考上一节内容：**数据清理-噪声数据**）
  - 分箱
    - 1. 排序数据，并将他们分到等深的箱中
    - 2. 按箱平均值平滑、按箱中值平滑、按箱边界平滑等
  - **聚类：**监测并且去除噪声数据
    - 将类似的数据聚成簇
    - 每个簇计算一个值用以将该簇的数据离散化



数据变换-离散化

143

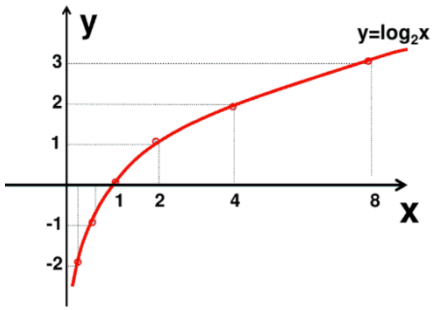
- 有监督离散化—基于熵的离散化
  - 熵用来度量系统的不确定程度
    - 熵是由 克劳德·艾尔伍德·香农 将热力学的熵，引入到信息论，因此它又被称为香农熵



香农提出了信息熵的概念，为**信息论**和**数字通信奠定了基础**，被誉为“**信息论之父**”

数据变换-离散化

- 信息熵：度量系统的不确定程度
  - 信息量
    - 定义一个时间x的概率分布为P(x)
    - 则事件x的自信息量是-logP(x)，取值范围:



[0, +∞]

  - 信息熵
    - 平均而言，发生一个事件我们得到的自信息量大小
    - 即：熵可以表示为自信息量的期望

$$H = -\sum P(x)\log P(x)$$

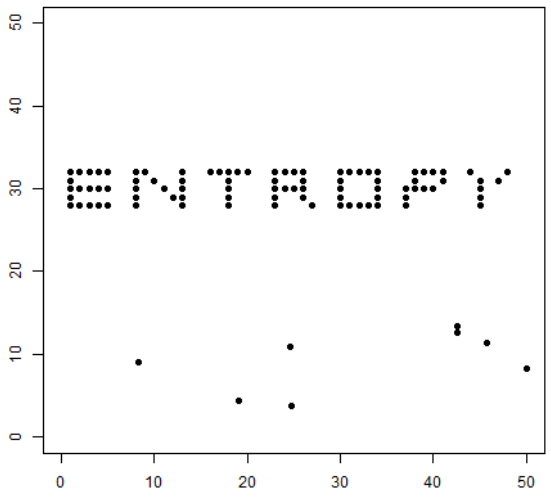
X	0	1
P(x)	0.4	0.6

$$\begin{aligned} H(P) &= -P(X=0)\log_2 P(X=0) - P(X=1)\log_2 P(X=1) \\ &= -0.4 * log_2(0.4) - 0.6 * log_2(0.6) \\ &\approx 0.97 \end{aligned}$$

数据变换-离散化

145

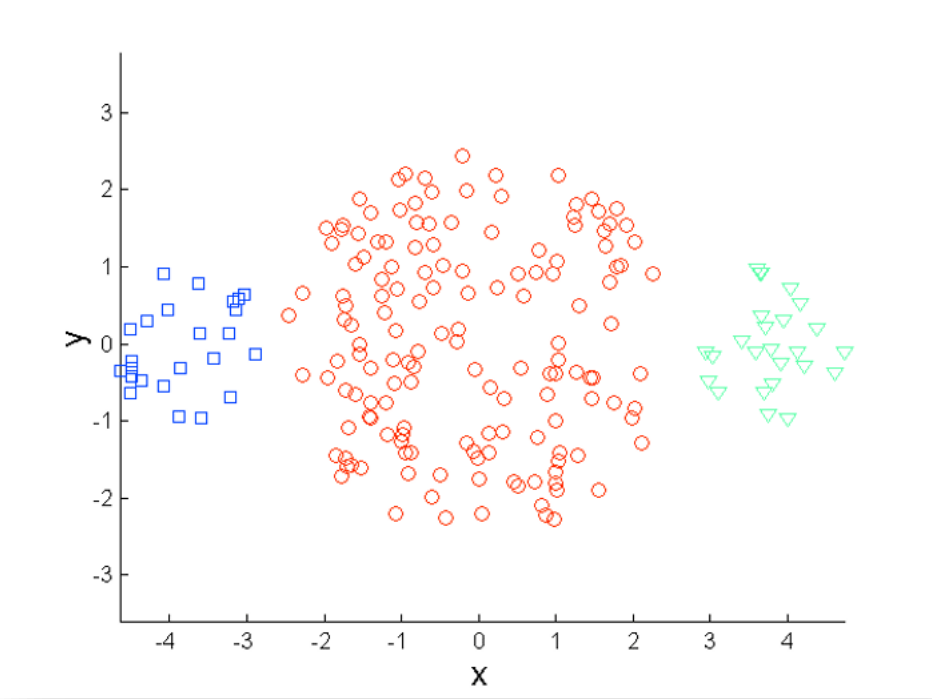
- 熵与数据离散化有什么关系？——**不确定程度**
  - 数据点单词（ENTROPY）**完整**的时候，容易理解表达的意思，**确定程度较高**，对应的**信息熵也较小**。
  - 数据点被完全打乱的时候，难以理解其意思，造成**不确定性**也就多了，对应的**信息熵也变大**了。
  - 目标：对数据进行离散化后，每个区间的数据的确定性（又称“纯度”）更高，因此用熵来对数据进行离散化。



数据变换-离散化

146

- 基于熵的离散化
  - 在x轴上对数据划分

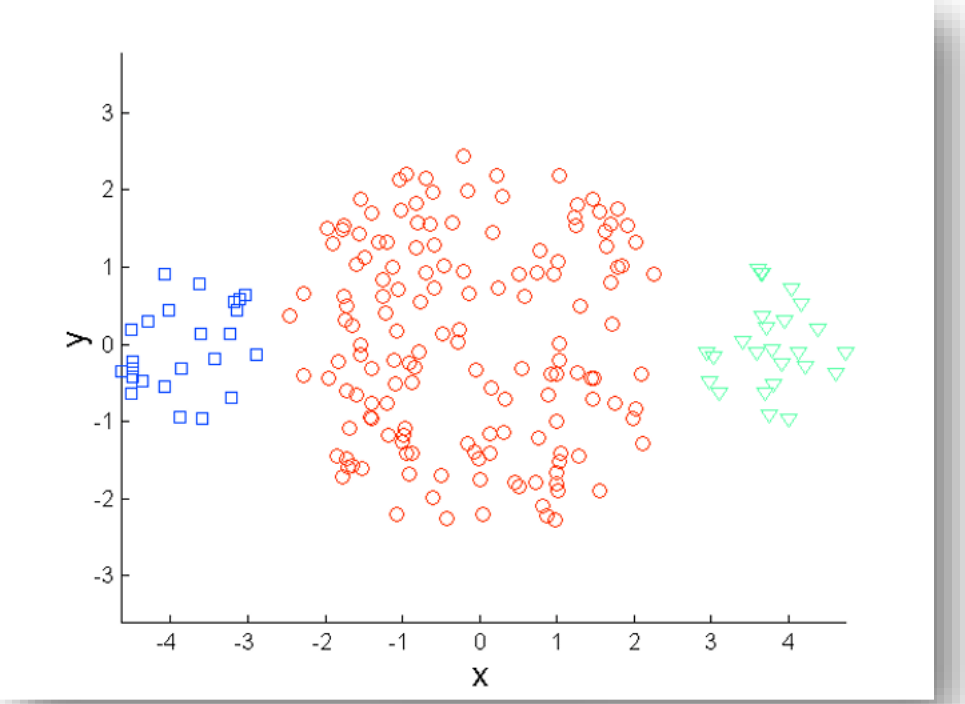


原始数据

数据变换-离散化

147

- 基于熵的离散化
  - 在x轴上对数据划分



原始数据

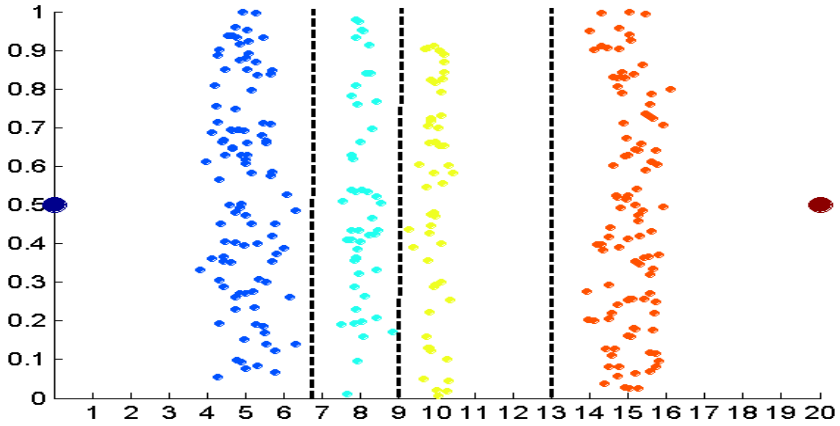
数据变换-离散化

148

- 熵—计算不确定性以及不纯性
  - 假设数据已经离散，计算离散后的某个区间 ***t*** 中的熵:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

- 其中，***p*(*j* | *t*)** 表示 第 ***j*** 类在区间 ***t*** 中的概率；一般对数log以2为底



数据变换-离散化

- 计算 单个区间 的 Entropy

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	0
C2	6

$P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1$

$Entropy = -0 \log_2 0 - 1 \log_2 1 = -0 - 0 = 0$

C1	1
C2	5

$P(C1) = 1/6 \quad P(C2) = 5/6$

$Entropy = -(1/6) \log_2 (1/6) - (5/6) \log_2 (5/6) = 0.65$

C1	2
C2	4

$P(C1) = 2/6 \quad P(C2) = 4/6$

$Entropy = -(2/6) \log_2 (2/6) - (4/6) \log_2 (4/6) = 0.92$

练习: (1) 假设区里t里面C1和C2的样本数各为3，Entropy是多少? **1**

(2) 假设区间t里面有4个类，且样本数一样，Entropy是多少? **2**

(3) 假设区间t里面有C个类，且样本数一样，Entropy是多少? **logC**

数据变换-离散化

- 熵—计算不确定性以及不纯性
  - 假设数据已经离散，计算离散后的某个区间 ***t*** 中的熵:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

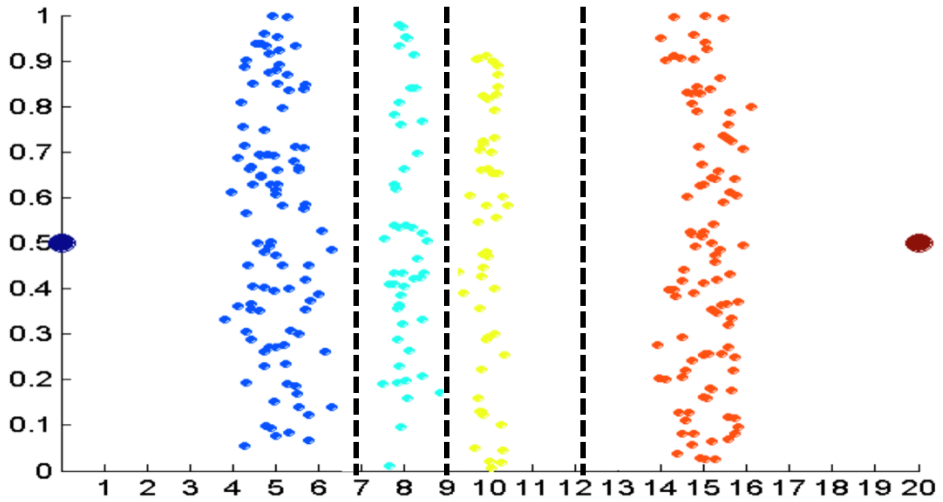
- 其中，$p(j \mid t)$ 表示 第 j 类在区间 t 中的概率；一般对数log以2为底

- **区间里面不同类别的样本均匀分布时，熵值最大（最不确定、最不纯），熵值为：logC**
- **区间结面只有一类样本时，熵值最小（最确定、最纯）**
- **熵的论值范围：[0, logC]**

数据变换-离散化

151

- 根据Entropy进行二分离散化
  - 先找到一个分隔点（属性值），把所有数据分到两个区间
  - 分别对两个子区间的数据进行二分隔
  - 重复以上步骤



数据变换-离散化

152

- **如何确定分隔点？**--计算分隔后的信息增益
  - 信息增益（Information Gain）：

$$GAIN_{\text{split}} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)\right)$$

    - 信息增益：表示在某个条件下，信息不确定性减少的程度。
    - 父节点 P 被分隔为 K 个区间
    - n 表示总记录数，n_i表示区间 i 中的记录数
  - 确定分隔点 j ：
    - 选择信息增益最大的分隔点，即

$$j = max(GAIN_{\text{split}})$$

数据变换-离散化

- 如何确定分隔点？--计算分隔后的信息增益
  - 信息增益（Information Gain）
    - 表示在某个条件下，信息不确定性减少的程度

$Ent(p)$

属性	x_1	x_2	x_3	……	x_{n-2}	x_{n-1}	x_n
类别	C0	C0	C1	……	C1	C1	C0

x_1	x_2
C0	C0

x_3	……	x_{n-2}	x_{n-1}	x_n
C1	……	C1	C1	C0

$Ent(m_{11})$ $Ent(m_{12})$

$$Ent(m_1) = \frac{n1}{n} Ent(m11) + \frac{n2}{n} Ent(m12)$$

$$Gain1 = Ent(p) - Ent(m_1) > 0$$

数据变换-离散化

- 如何确定分隔点？--计算分隔后的信息增益
  - 信息增益（Information Gain）
    - 表示在某个条件下，信息不确定性减少的程度

$Ent(p)$

属性	x_1	x_2	x_3	……	x_{n-2}	x_{n-1}	x_n
类别	C0	C0	C1	……	C1	C1	C0

x_1	x_2
C0	C0

$Ent(m_{11})$

x_3	……	x_{n-2}	x_{n-1}	x_n
C1	……	C1	C1	C0

$Ent(m_{12})$

x_1	x_2	x_3	……	x_{n-2}
C0	C0	C1	……	C1

$Ent(m_{21})$

x_{n-1}	x_n
C1	C0

$Ent(m_{22})$

$$Ent(m_1) = \frac{n1}{n} Ent(m11) + \frac{n2}{n} Ent(m12)$$

$Ent(m_2)$

数据变换-离散化

- 如何确定分隔点？--计算分隔后的信息增益
  - 信息增益（Information Gain）
    - 表示在某个条件下，信息不确定性减少的程度

***Ent*(*p*)**

属性	x_1	x_2	x_3	……	x_{n-2}	x_{n-1}	x_n
类别	C0	C0	C1	……	C1	C1	C0

x_1	x_2
C0	C0

x_3	……	x_{n-2}	x_{n-1}	x_n
C1	……	C1	C1	C0

x_1	x_2	x_3	……	x_{n-2}
C0	C0	C1	……	C1

x_{n-1}	x_n
C1	C0

Ent(m11) *Ent*(m12)

***Ent*(m1)**

Ent(m21) *Ent*(m22)

***Ent*(m2)**

***Gain1* = *Ent*(*p*) - *Ent*(m1)** **Vs** ***Gain2*= *Ent*(*p*) - *Ent*(m2)**

数据变换-离散化

156

- **如何确定分隔点？**--计算分隔后的信息增益
  - 信息增益（Information Gain）：

$$GAIN_{\text{split}} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)\right)$$

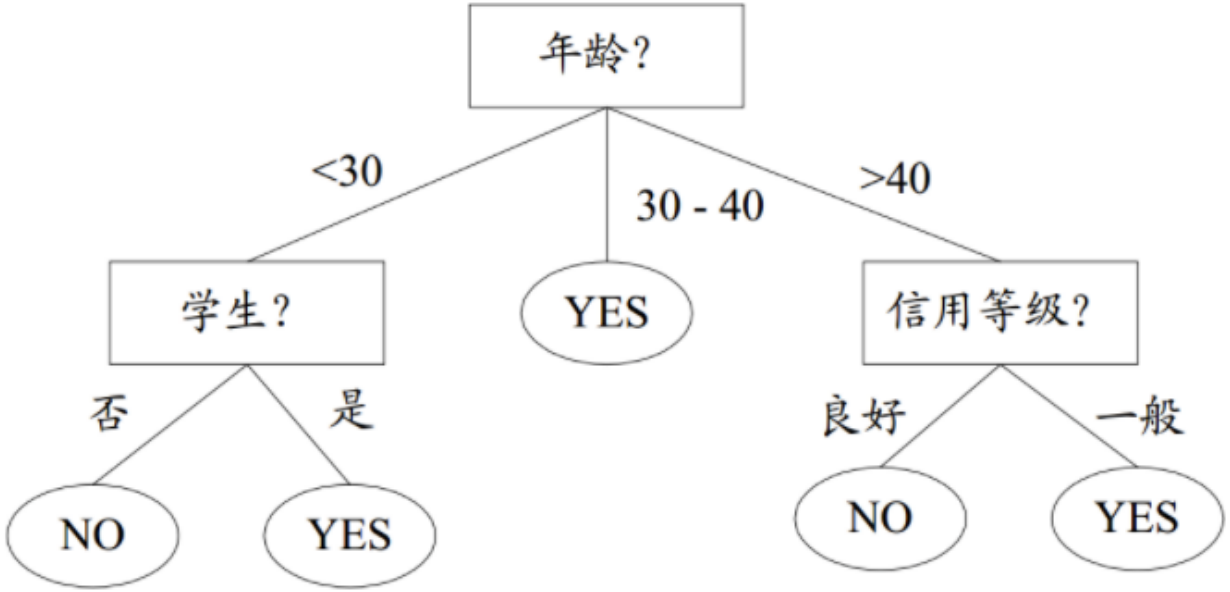
    - 信息增益：表示在某个条件下，信息不确定性减少的程度。
    - 父节点 P 被分隔为 K 个区间
    - n 表示总记录数，n_i表示区间 i 中的记录数
  - 确定分隔点 j ：
    - 选择信息增益最大的分隔点，即

$$j = max(GAIN_{\text{split}})$$

数据变换-离散化

157

- 十大经典机器学习算法
  - 决策树 (第四章：数据挖掘)



数据变换-离散化

158

- 熵（*Entropy*）的应用举例
  - 使用熵进行旅游季节（Travel Season）的划分
    - 假设不同的景点适合不同的季节进行旅游

- 根据景点的类别分布，计算区间（季节）中的熵:

$$WAE(i;S^P) = \frac{|S_1^P(i)|}{|S^P|} Ent(S_1^P(i)) + \frac{|S_2^P(i)|}{|S^P|} Ent(S_2^P(i))$$

Qi Liu, Yong Ge, Zhongmou Li, **Enhong Chen,** and Hui Xiong. Personalized Travel Package Recommendation. **ICDM'2011**. **Best Research Paper**

课后学习

159

- 前沿文献调研： “熵在数据科学中的应用"
  - 推荐1：基于技术分布的熵值预测公司发展前景
    - 技术的发展一般处于5个阶段(萌芽期、过热期、低谷期、复苏期和成熟期)，如果公司的技术发展在以上阶段分布越均衡，可能它的发展前景就越好
      - Bo Jin, Yong Ge, Hengshu Zhu, Li Guo, Hui Xiong and Chao Zhang. Technology Prospecting for High Tech Companies through Patent Mining ICDM'2014
  - 推荐2：基于交叉熵的机器学习目标函数设计
    - 信息熵、交叉熵和相对熵： https://charlesliuyx.github.io/2017/09/11/

参考资料

- Qi Liu, Yong Ge, Zhongmou Li, Enhong Chen, and Hui Xiong. Personalized Travel Package Recommendation. ICDM'2011
- Bo Jin, Yong Ge, Hengshu Zhu, Li Guo, Hui Xiong and Chao Zhang. Technology Prospecting for High Tech Companies through Patent Mining ICDM'2014
- 数据规范化的几种方法： https://www.jianshu.com/p/55aee18b3fbc
- Z-score（Z值）的意义： http://blog.sina.com.cn/s/blog_72208a6a0101cdt1.html
- 信息熵是什么： https://www.zhihu.com/question/22178202
- 交叉熵损失函数的优点： https://blog.csdn.net/qq_41853758/article/details/82826820
- 信息熵、交叉熵和相对熵： https://charlesliuyx.github.io/2017/09/11/
- 常见的三种数据规范化方法及其python实现： https://joshuaqyh.github.io/2019/02/24/
- 一种基于信息熵的离散化方法（MDLP）python实现：https://zhuanlan.zhihu.com/p/74839156

数据预处理

- 大数据环境下的数据特征
- 为什么需要进行预处理
- **预处理的基本方法**
  - 数据清理
  - 数据集成
  - 数据变换
  - **数据规约**

数据预处理：数据规约

- 为什么需要进行数据规约？
  - **数据清理、数据集成之后，获得多源且质量完好的数据集**
  - 但数据规模很大：大规模数据样本，使得在整个数据集上进行复杂的数据分析与建模需要**很多计算资源和很长的时间**

- 例：电子商务中的商品类别

CVPR 2021 AliProducts Challenge: Large-scale Product Recognition

- 背景：电商企业面临的大规模、细粒度商品图像识别问题
- 数据量：**300万张图片**，涵盖了**5万个SKU级商品类别**

数据预处理：数据规约

- **数据归约**
  - **目标：缩小数据挖掘所需的数据集规模**
  - 得到数据集的归约表示，它小得多，但可以产生相同的（或几乎相同的）分析结果
  - 用于数据归约的时间不应当超过或“抵消”在归约后的数据上挖掘节省的时间
- 常用的数据规约方法
  - 维度归约
    - 减少所考虑的随机变量或属性的个数
  - 数值归约
    - 用较小的数据表示形式替换原始数据
    - e.g. 使用模型来表示数据

数据规约-维度规约

- **维度规约**
  - 大数据应用包含几万至几百万的属性，其中大部分属性与挖掘任务不相关，是冗余的，
  - **数据降维：**删除不相关的属性，并保证信息的损失最小
- **维度归约方法**
  - 主成分分析
  - 特征子集选择

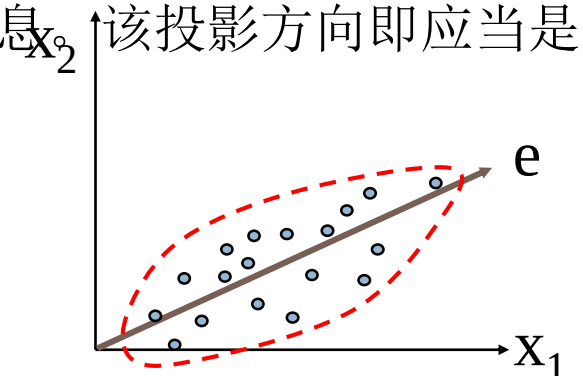
个人借贷数据

loan_id	119262	贷款记录唯一标识
user_id	0	用户唯一标识
total_loan	12000.0	贷款金额
year_of_loan	5	贷款期限（year）
interest	11.53	贷款利率
…	…	…

NLP中的Glove词表
https://nlp.stanford.edu/projects/glove/

数据规约-维度规约

- 主成分分析(principal component analysis, PCA)
  - 目的：数据降维、数据去噪、数据压缩（去除冗余属性）
  - 思想：将原高维（如维度为N）数据向一个较低维度（如维度为K）的空间投影，同时使得数据之间的区分度变大。这K维空间的每一个维度的基向量（坐标）就是一个主成分
  - **问题：**如何找到这K个主成分
    - **消除原始数据不同属性间的相关性**，要求：K个维度间相互独立
    - **最大化保留K维度上的数据多样性**，要求：最大化每个维度内的样本方差。使用方差信息，若在一个方向上发现数据分布的方差越大，则说明该投影方向越能体现数据中的主要信息，该投影方向即应当是一个主成分

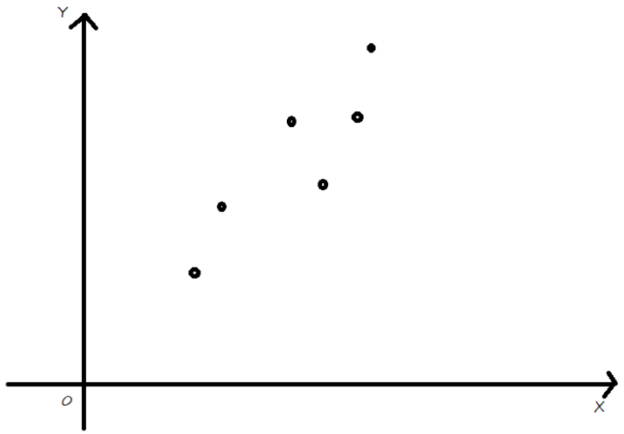


$$D = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{bmatrix} \Rightarrow D' = \begin{bmatrix} x_1' & x_2' & x_3' & x_4' \\ \cancel{0} & \cancel{0} & \cancel{0} & \cancel{0} \end{bmatrix}$$

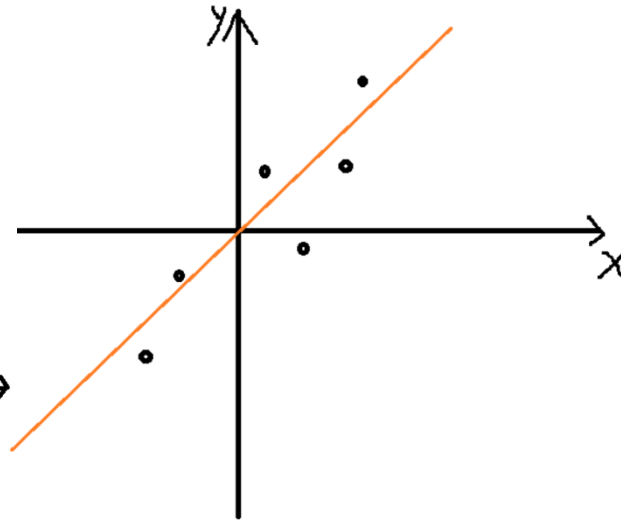
数据规约-维度规约

- 主成分分析(principal component analysis, PCA)
  - 例：如何把二维数据降维至一维
    - 思想：方差越大，数据间的**差异**越大。让新数据的**方差**尽可能地大，使得新数据尽可能地不丢失原有数据的信息

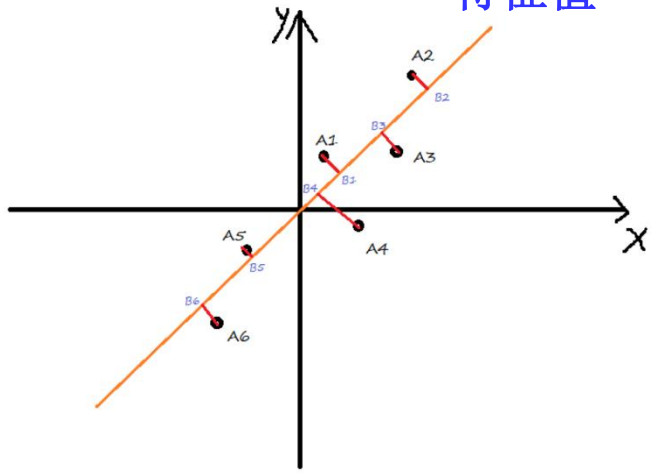
原始二维数据



去中心化



协方差矩阵 { 特征向量 / 特征值



课后思考：一元线性回归与PCA的关系？

数据规约-维度规约

- 主成分分析(principal component analysis, PCA)
  - 原数据为X，降维后新数据为Y
    - **不同维度相互独立：**要求数据Y的协方差矩阵为对角阵 $B = \frac{1}{m}Y^\top Y \in \mathrm{R}^{k\times k}$

$$B = \frac{1}{m}Y^\top Y = \frac{1}{m}(XP)^\top(XP) = \frac{1}{m}P^\top X^\top XP = P^\top CP \qquad C = \frac{1}{m}X^\top X \in \mathrm{R}^{n\times n}$$

    - **最大化每个维度内的样本方差：**Y的协方差矩阵中的对角元素越大越好
      - 对 C 进行特征值分解，将求解得到的特征值从大到小依次作为协方差矩阵的对角线元素，特征向量组成变换矩阵 (P)
    - 保留最大的 k 个特征值：第K个主成分就是第K大的特征值对应的特征向量

  - 存储空间：
    - 对于原始的N*M（N维M个样本）的数据，原始存储空间是N*M
    - PCA以后为：K*M（M个K维样本）+N*K(K个特征向量)

数据规约-维度规约

主成分分析(principal component analysis, PCA)

Algorithm 5 PCA 算法

Input: 原始的样本矩阵 $X \in \mathrm{R}^{m \times n}$

Output: 压缩后的样本矩阵 $Y \in \mathrm{R}^{m \times k}$

1: 对样本矩阵进行去均值化 $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i - \frac{1}{m}\sum_{i=1}^{m} \boldsymbol{x}_i, \forall i \in \{1, 2, \cdots, m\}$

2: 计算协方差矩阵 $C = \frac{1}{m} X^\top X$

3: 通过特征值分解求解 C 的特征值和特征向量

4: 将特征值从大到小排序，取最大的 k 个特征值对应的特征向量作为列向量构成变换矩阵 $P \in \mathrm{R}^{n \times k}$

5: 将原始数据转换到新的空间中 $Y = XP$

- 不足之处
  - 当原始数据的维度n特别大的时候，计算协方差时的 X⊤X 已经具有相当大的计算量
  - 针对协方差矩阵C的特征值求解过程计算效率不高

数据规约-维度规约

样本矩阵(10个城市样本，8个属性)的转置X^T

省份	GDP X_1	居民消费水平 X_2	固定资产投资 X_3	职工平均工资 X_4	货物周转量 X_5	居民消费价格指数 X_6	商品零售价格指数 X_7	工业总产值 X_8
北京	1394.89	2505	519.01	8144	373.9	117.3	112.6	843.43
天津	920.11	2720	345.46	6501	342.8	115.2	110.6	582.51
河北	2849.52	1258	704.87	4839	2033.3	115.2	115.8	1234.85
山西	1092.48	1250	290.9	4721	717.3	116.9	115.6	697.25
内蒙	832.88	1387	250.23	4134	781.7	117.5	116.8	419.39
辽宁	2793.37	2397	387.99	4911	1371.1	116.1	114	1840.55
吉林	1129.2	1872	320.45	4430	497.4	115.2	114.2	762.47
黑龙江	2014.53	2334	435.73	4145	824.8	116.1	114.3	1240.37
上海	2462.57	5343	996.48	9279	207.4	118.7	113	1642.95
江苏	5155.25	1926	1434.95	5943	1025.5	115.8	114.3	2026.64

数据归一化并得到协方差矩阵

	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
X_1	1.000	.267	.951	.191	.617	−.274	−.264	.874
X_2	.267	1.000	.426	.718	−.151	−.234	−.593	.363
X_3	.951	.426	1.000	.400	431	−.282	−.359	.792
X_4	.191	.718	.400	1.000	−.356	−.134	−.539	.104
X_5	.617	−.151	.431	−.356	1.000	−.255	.022	.659
X_6	−.274	−.234	−.282	−.134	−.255	1.000	.760	−.126
X_7	−.264	−.593	−.359	−.539	.022	.760	1.000	−.192
X_8	.874	.363	.792	−.104	.659	−.126	−.192	1.000

特征值分解

三个主成分 (8*3)

第一特征向量 a_1	第二特征向量 a_2	第三特征向量 a_3
0.470641	0.107995	0.19241
0.456708	0.258512	0.109819
0.424712	0.287536	0.19241
−0.31944	0.400931	0.397525
0.312729	−0.40431	0.24505
0.250802	0.498801	−0.24777
0.240481	−0.48868	0.332179
−0.26267	0.167392	0.723351

数据规约-维度规约

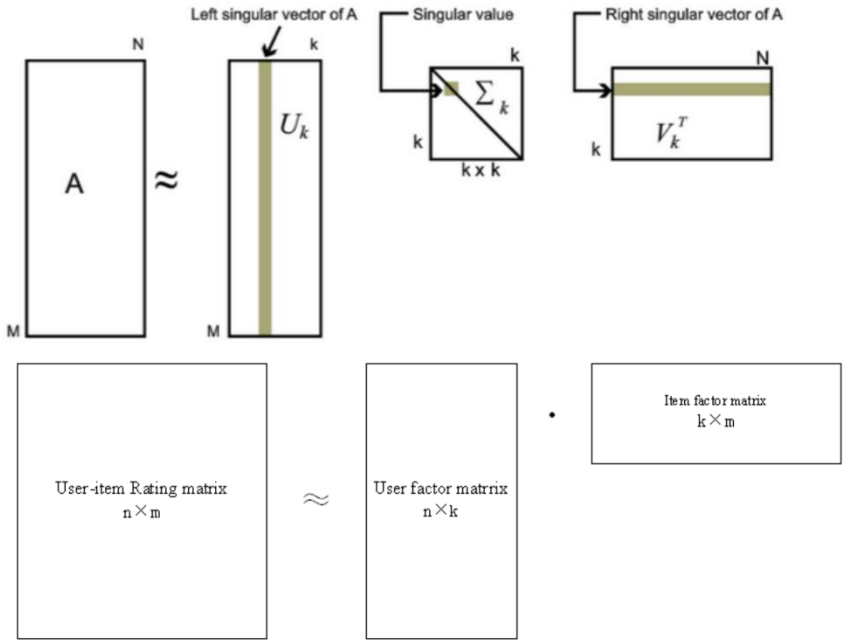
先把大图像分成16*16（256）的小图像块，把小图像块当成一个256维的向量，所有256维向量拼接成新的数据矩阵，对其进行归一化和PCA压缩（取前四个特征值，取前八个，前16个，一直到前256个特征值），压缩完以后需要重构图像就会得到以上的效果

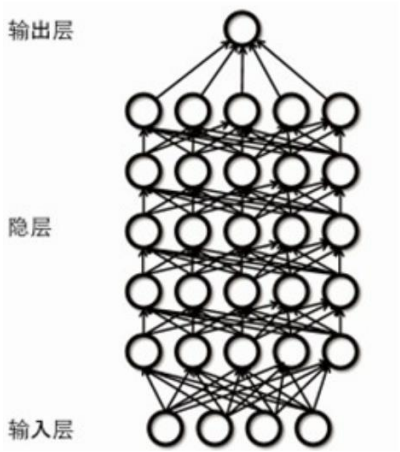
数据规约-维度规约

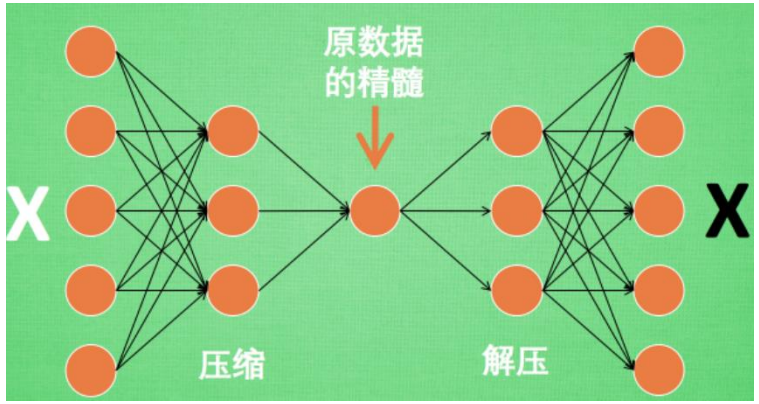
- 维度规约
  - 属性子集选择（特征子集）
    - 做法：删除不相关或冗余的属性来减少维度与数据量
    - 目标：找到最小属性集，使得数据的概率分布尽可能接近使用所有属性得到的原分布
    - 理解：从全部属性中选取一个特征属性子集，使构造出来的模型更好
  - 启发式步骤
    - 建立子集集合
    - 构造评价函数
    - 构建停止准则
    - 验证有效性
  - 例如：决策树

数据规约-维度规约

- 维度规约
  - 奇异值分解(SVD)
  - 矩阵分解(PMF)
  - 深度学习(Deep Learning)
    - DNN，AutoEncoder, etc







数据规约-数值规约

- **数值规约**
  - 通过选择替代的、较小的数据表示形式来减少数据量
- 参数化方法
  - 使用一个参数模型估计数据，最后只要存储参数即可，不用存储数据（除了可能的离群点）
  - 常用方法
    - 线性回归方法；多元回归；对数线性模型；
- 非参数化方法
  - 不使用模型的方法存储数据
  - 常用方法：直方图，聚类，抽样

资料推荐

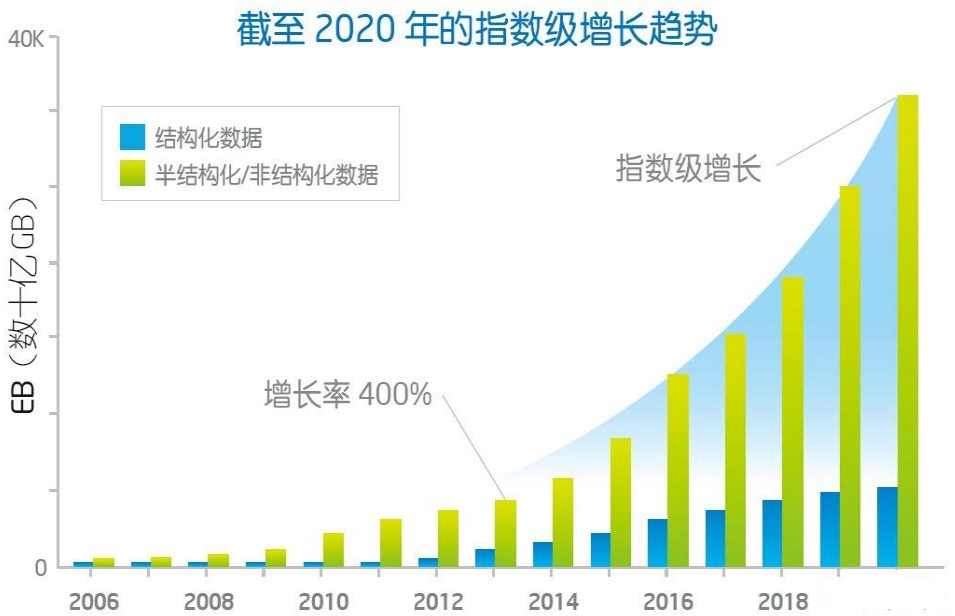
- 数据挖掘导论 第2章：数据，人民邮电出版社
- 数据挖掘原理与算法 第2章， 清华大学出版社
- T.C. Redman Data Quality: The Field Guide. January 2001
- I.T.Jolliffe. Principal Component Analysis. Springer Verlag, 2nd edition, October 2002.
- Feature selection algorithms: A survey and experimental evaluation, ICDM 2003
- Zhenya Huang, Qi Liu, Enhong Chen, Learning or Forgetting? A Dynamic Approach for Tracking the Knowledge Proficiency of Students, ACM TOIS
- Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang,, Neural Cognitive Diagnosis for Intelligent Education Systems, AAAI'2020

回顾：数据预处理

- 大数据环境下的数据特征
- 为什么需要进行预处理
- 预处理的基本方法
  - 数据清理
  - 数据集成
  - 数据变换
  - 数据规约

数据分析基础

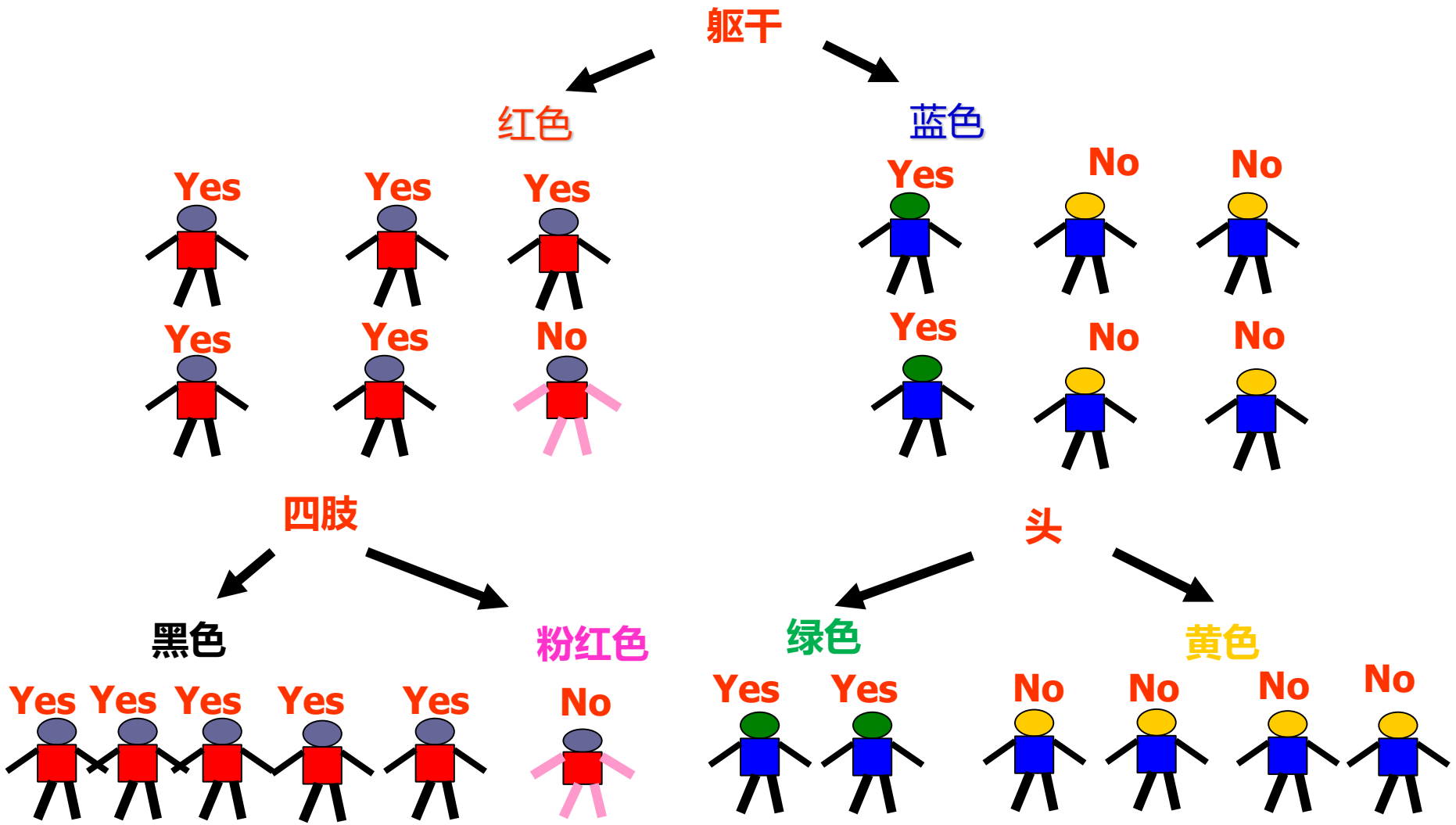
- 数据采集 Data Collection
- 数据存储 Data Storage
- 数据预处理 Data Preprocessing
- **特征工程 Feature Engineering**



特征工程

- 特征工程的定义
- 特征工程的流程
- 特征学习
- 案例学习
- 参考文献

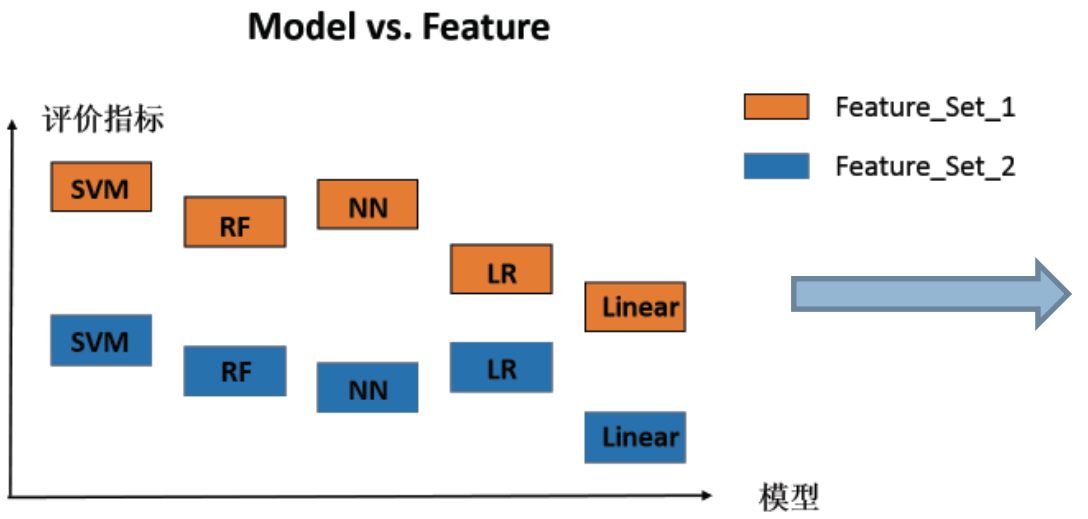
特征工程


躯干
红色
蓝色
Yes
Yes
Yes
Yes
Yes
No
Yes
No
No
Yes
No
No
四肢
头
黑色
粉红色
绿色
黄色
Yes Yes Yes Yes Yes
No
Yes
Yes
No
No
No
No

特征工程

特征工程是什么？ (Feature Engineering)

在数据预处理以后（或者数据预处理过程中），如何从数据中提取有效的特征，使这些特征能够尽可能的表达原始数据中的信息，使得后续建立的数据模型能达到更好的效果，就是特征工程所要做的工作。



数据特征决定了模型的上限

- Feature 决定模型 UpperBound
- Model 决定接近 UpperBound的程度
- 不同的问题下Model的表现的不同

特征工程

特征工程的意义

著名数据科学家Andrew Ng 对特征工程这样描述的：“虽然提取数据特征是非常困难、耗时并且需要相关领域的专家知识，但是机器学习应用的基础就是特征工程”

- **特征越好，灵活性越强**

  好的特征能使一般的模型也能获得很好的性能，在不复杂的模型上运行速度很快，并且容易理解和维护。

- **特征越好，构建的模型越简单**

  好的特征不需要花太多的时间去寻找最优参数，降低了模型的复杂度，使模型趋于简单。

- **特征越好，模型的性能越出色**

  好的特征能够使模型表现越出色是毫无疑问的，而特征 的性能。

如何去做特征工程?

特征工程的流程

- 特征工程（**重复迭代**）的流程

1. 对特征进行头脑风暴

   深入分析问题，观察数据的基本统计信息，结合问题的相关领域知识和参考其他问题的相关特征工程的方法并应用到自身的问题中来。

2. **特征的设计—基础且重要的步骤**

   人工设计特征、自动提取特征，或两者结合，得到模型使用的特征。

3. **特征的选择**

   使用不同的特征重要性评分方法或者特征选择方法，对特征的有效性进行分析，选出有效的特征。

4. 评估模型

   利用所选择的特征对测试数据进行预测，评估模型的性能。

5. 上线测试

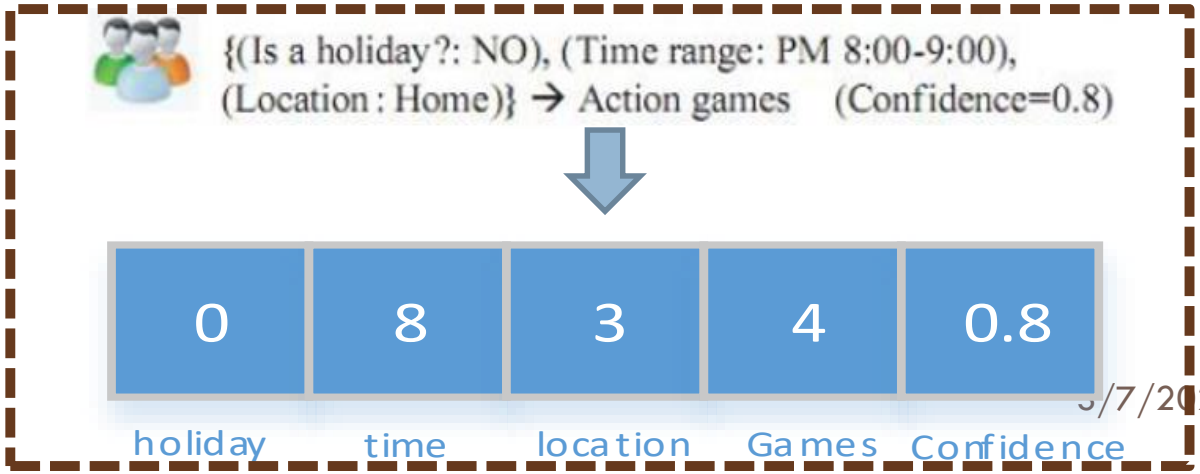
   通过在线测试的效果判断特征是否有效，若不能达到要求，则重复2-5 步骤，直到模型的性能达到要求。

特征的设计

- 从原始数据中如何设计特征？
  - **基本特征的提取**

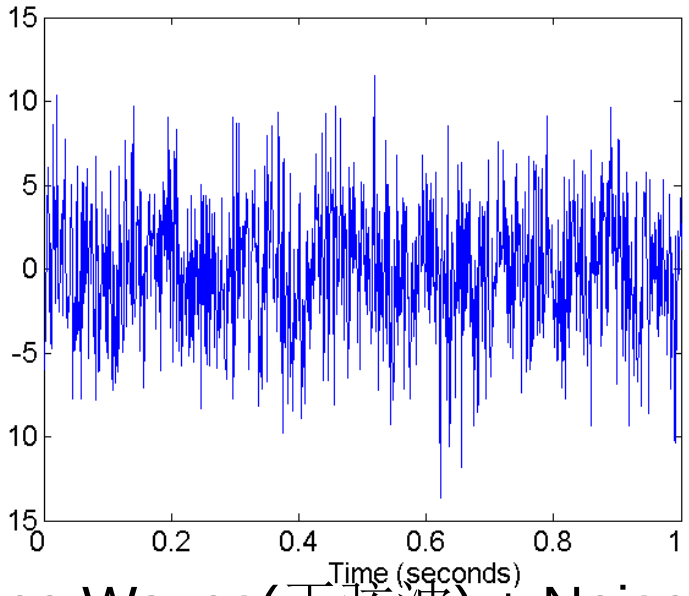
    基本特征的提取过程就是对原始数据进行预处理，将其转化成可以使用的数值特征。常见的方法有：数据的**归一化**、**离散化**、**缺失值补全**和**数据变换**等。
  - **创建新的特征**

    根据对应的领域知识，在基本特征的基础上进行特征之间的**比值**和**交叉变化**来构建新的特征。

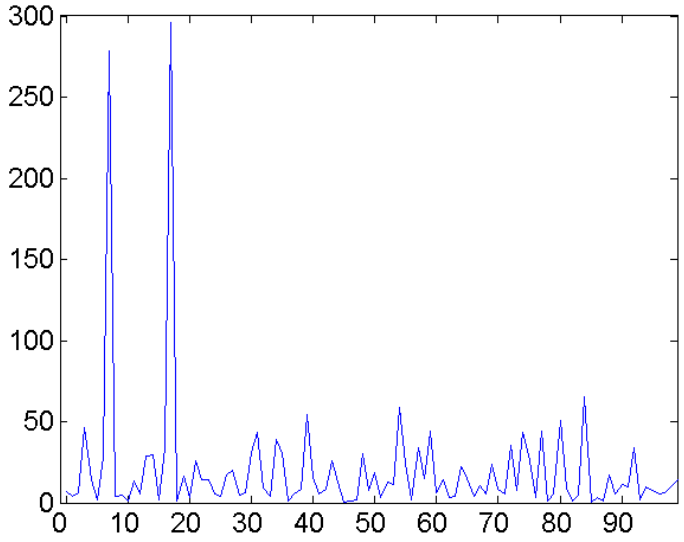


特征的设计

- 从原始数据中如何设计特征？
  - **函数变换特征**
    - 左图是根据两个Sin函数（分别是每秒7个和17个周期），以及一些噪声数据得到的序列图；
    - 右图是由**傅立叶变换**得到了频率图，可以看出变换后成功得到了两个概率最大的频率7和17（其中纵坐标是振幅，即概率值）



Two Sine Waves(正弦波) + Noise



Frequency

特征的设计

推荐系统中的“用户”和“商品”

*ratings.csv - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
userId,movieId,rating,timestamp
1,1,4.0,964982703
1,3,4.0,964981247
1,6,4.0,964982224
1,47,5.0,964983815
1,50,5.0,964982931
1,70,3.0,964982400
1,101,5.0,964980868
```

- 从原始数据中如何设计特征？
  - 独热特征表示 One-hot Representation
  - 将每个属性表示成一个很长的向量（每维代表一个属性值，如词语）
    - **函数：**[0, 0, 1, 0, 0, …, 0, 0, 0, 0]
    - **图像：**[0, 0, 0, 0, 0, …, 0, 0, 0, 1]
  - **优点：**直观，简洁
  - **缺陷：**
    - **“维度灾难”问题：**尤其是我们所构建的语料库包含的词语数据非常多的时候，独热表征在空间和时间上的开销都是十分巨大的
    - **“语义鸿沟”现象：**任意两个词之间都是完全孤立的，是无法刻画句子中词语的语序信息的（之前提到的词袋模型也是如此）。例如，我们是无法通过独热表征来判断“函数”与“偶函数”之间的联系的（但实际上这两个词语是非常相关的）。

特征的设计

- 从原始数据中如何设计特征？
  - 独热特征表示 One-hot Representation
    - **“维度灾难” 问题**
    - **“语义鸿沟” 现象**

Download pre-trained word vectors

- Pre-trained word vectors. This data is made available under the Pu
  http://www.opendatacommons.org/licenses/pddl/1.0/.
  - Wikipedia 2014 + Gigaword 5 (6B tokens, 400K vocab, uncas
  - Common Craw (42B tokens, 1.9M vocab, uncased, 300d vec
  - Common Craw (840B tokens, 2.2M vocab, cased, 300d vect
  - Twitter (2B twe ts, 27B tokens, 1.2M vocab, uncased, 25d, 50

“dog”	“canine”
3	399,999
$\begin{pmatrix}0\\0\\1\\0\\\vdots\\0\\0\end{pmatrix}$	$\begin{pmatrix}0\\0\\0\\0\\\vdots\\1\\0\end{pmatrix}$

特征的设计

- 从原始数据中如何设计特征？
  - **数据的统计特征，**如：文档中的词频统计

John likes to watch movies. Mary likes too.

John also likes to watch football games.

  - 字典

{"John": 1, "likes": 2, "to": 3, "watch": 4, "movies": 5, "also": 6, "football": 7, "games": 8, "Mary": 9, "too": 10}

  - 文档词频特征

[1, 2, 1, 1, 1, 0, 0, 0, 1, 1]

[1, 1, 1, 1, 0, 1, 1, 1, 0, 0]

特征的设计

- 从原始数据中如何设计特征？
  - **TF-IDF（词频-逆文档率）**
    - 算法简单高效,工业界用于最开始的**数据预处理**
  - 主要思想：找到能代表该文档中的**“关键词”**
  - 词频 （TF, Term Frequency）
    - TF = 某个词(特征值)在句子(数据)中出现的频率

$$TF_{w,D_i} = \frac{count(w)}{|D_i|}$$

  - 逆文档率 （IDF, Inverse Document Frequency）
    - IDF = log(语料库(数据库)的句子(数据)总数 / 包含该词(特征值)

$$IDF_w = \log \frac{N}{1 + \sum_{i=1}^{N} I(w, D_i)}$$

https://blog.csdn.net/xiaoyu

  - 每个特征值（词）的重要性
    - w_{ij}= tf*idf = $TF_{ij}*\log(N/DF_i)$

可能会有微小变型

特征的设计

$$TF_{w,D_i} = \frac{count(w)}{|D_i|} \qquad IDF_w = \log \frac{N}{1+\sum_{i=1}^{N} I(w, D_i)}$$

https://blog.csdn.net/xiaosyu

- 从原始数据中如何设计特征？—计算 TF-IDF
  - d_1 (A, B, C, C, S, D, A, B, T, S, S, S, T, W, W)
  - d_2 (C, S, S, T, W, W, A, B, S, B)
  - d_3(不含ABCDSTW)

文档中词总数=25

TF

	d_1	d_2
A	0.08	0.04
B	0.08	0.08
C	0.08	0.04
D	0.04	0.00
S	0.16	0.12
T	0.08	0.04
W	0.08	0.08

IDF

	IDF
A	0.4
B	0.4
C	0.4
D	1.1
S	0.4
T	0.4
W	0.4

TF-IDF

	d_1	d_2
A	0.032	0.016
B	0.032	0.032
C	0.032	0.016
D	0.044	0.000
S	0.064	0.048
T	0.032	0.016
W	0.032	0.032

特征的设计

- 从原始数据中如何设计特征？
  - **TF-IDF（词频-逆文档率）** w_{ij}= tf*idf = $TF_{ij}*\log(N/DF_i)$
- 优点
  - 简单快速的词（特征）重要性表示方法，结果比较符合实际情况
  - 应用广泛：不仅限于文本数据
- 缺点
  - 单纯以“词频”衡量一个词的重要性，不够全面，有时重要的词可能出现次数并不多
  - 无法体现词的**位置信息、顺序信息，**出现位置靠前的词与出现位置靠后的词，都被视为重要性相同
  - 无法发现词（特征）的隐含联系，如同义词等

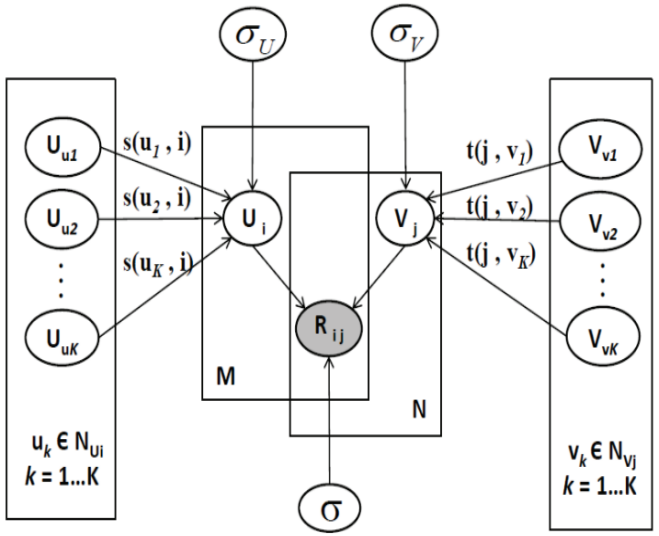
特征的设计

- 从原始数据中如何设计特征？
  - **TF-IDF（词频-逆文档率）—应用**
    - 搜索引擎；关键词提取；文本相似性；文本摘要
  - 推荐系统
    - 可以计算“用户-标签-商品”的特征
    - 用户-标签的TF-IDF

$$P_{il} = tf(i,l) \times \ln\left(\frac{M}{df(l)}\right)$$

- 用户：i。标签：l。用户总数：M。

$$s(i,j) = \cos(\vec{i},\vec{j}) = \frac{\vec{i}\cdot\vec{j}}{\|\vec{i}\|_F * \|\vec{j}\|_F}$$



Le Wu, Enhong Chen, Qi Liu, Leveraging Tagging for Neighborhood-aware Probabilistic Matrix Factorization. CIKM'2012

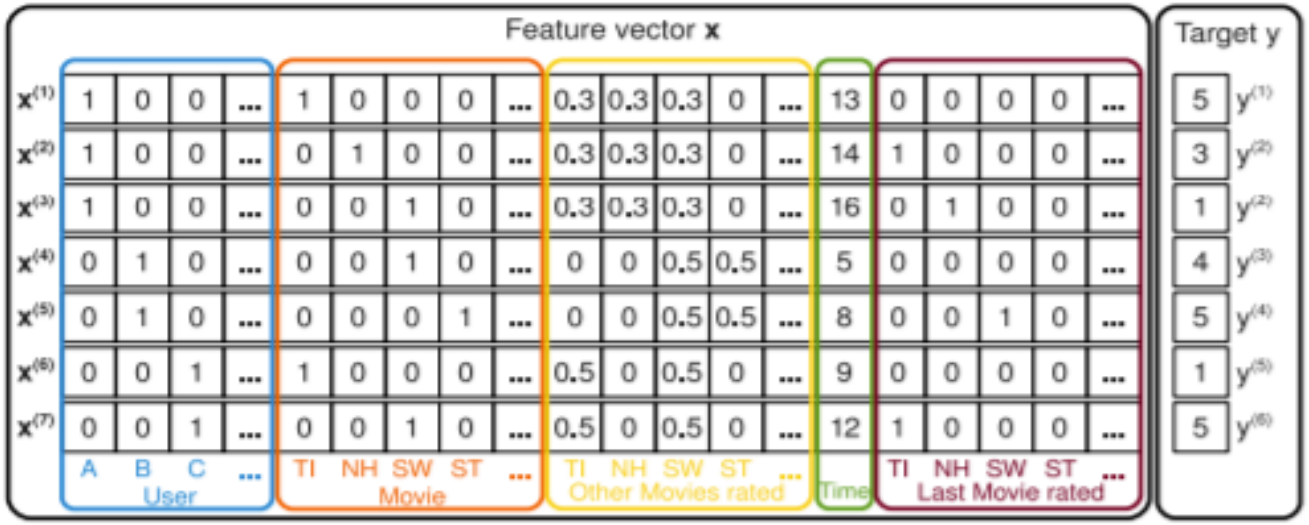
特征的设计

*ratings.csv - 记事本

文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)

```
userId,movieId,rating,timestamp
1,1,4.0,964982703
1,3,4.0,964981247
1,6,4.0,964982224
1,47,5.0,964983815
1,50,5.0,964982931
1,70,3.0,964982400
1,101,5.0,964980868
```

- 从原始数据中如何设计特征？
  - **特征组合：构造高阶特征**
  - 上述所有构造的特征均可以：两两、三三、… 进行组合
    - Factorization Machine (2012)



$$S = \{(\mathrm{A}, \mathrm{TI}, 2010\text{-}1, 5), (\mathrm{A}, \mathrm{NH}, 2010\text{-}2, 3), (\mathrm{A}, \mathrm{SW}, 2010\text{-}4, 1), (\mathrm{B}, \mathrm{SW}, 2009\text{-}5, 4), (\mathrm{B}, \mathrm{ST}, 2009\text{-}8, 5), (\mathrm{C}, \mathrm{TI}, 2009\text{-}9, 1), (\mathrm{C}, \mathrm{SW}, 2009\text{-}12, 5)\}$$

$$\tilde{y}(x) = w_0 + \sum_{i=1}^{n} w_i \boxed{x_i} + \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij} \boxed{x_i x_j}$$

特征的设计

- 举例：第二届“中国高校计算机大赛-大数据挑战赛”
- 赛题描述/数据：http://bdc.saikr.com/vse/bdc/2017
- 简单的说，该赛题的求解目标是利用数据分析将人工的鼠标轨迹和代码生成的鼠标轨迹区分开来。这里的鼠标轨迹是指一种完成一种验证手段——拖动滑块到指定区域时鼠标的轨迹。

- 原始数据格式：一系列连续点的坐

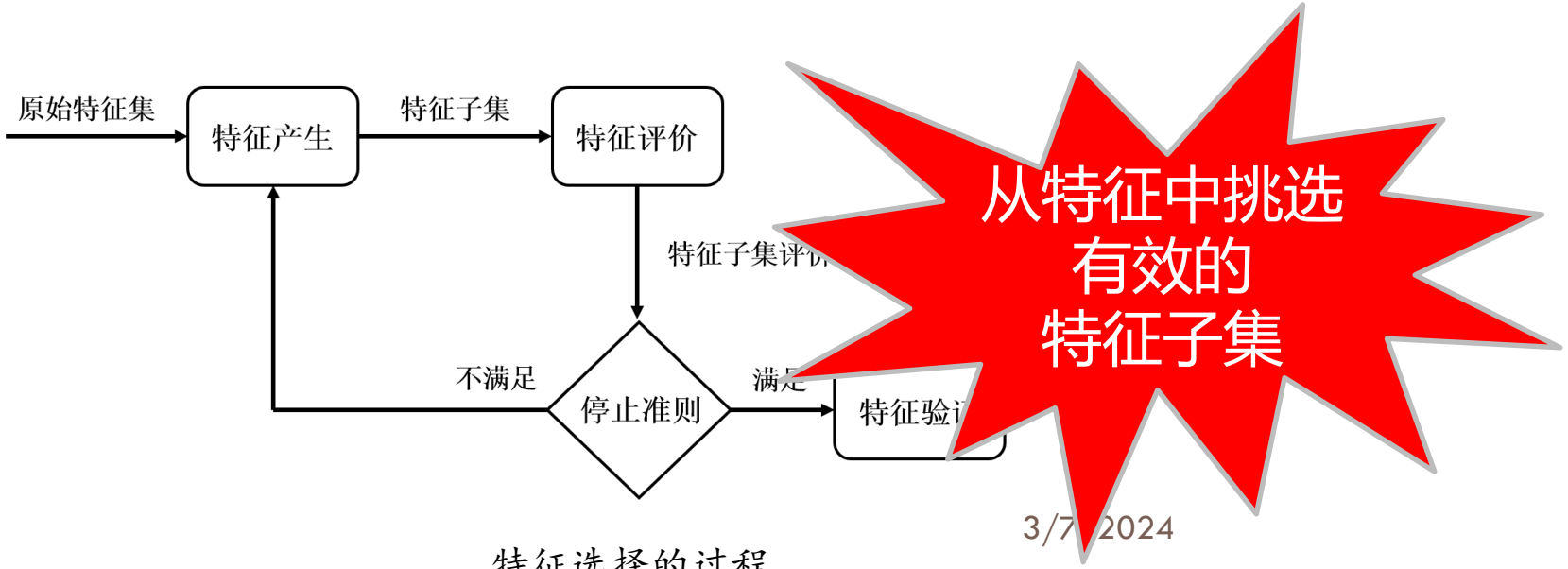
例如：(2,3,4),(2,5,6)(4,3,7) (4,3)，该轨迹中含有三个点的坐标，以(x,y, time)的时间表示，终点坐标为(4,3)

特征的设计

- 从原始数据中如何设计特征？
  - **基本特征的提取**
    - 轨迹运动数据的统计值：运动速度/加速度/角加速度/角速度的均值/极值/最值/中位数 等
    - 轨迹的描述：运动在x轴方向是否为单向，曲线平滑程度， 等
  - **创建新的特征**
    - 基本特征的简单二元运算， 加/减/乘/除/平方和/和平方/倒数和
    - 运动数据在某一维上的偏导
    - **领域专家知识**

特征的选择

- 如何挑选有效的特征（**Subset Selection问题**）？
  - 在实际应用中，特征的数量往往比较多，其中可能会存在不相关的特征。
  - 特征数量越多，分析特征、训练模型所需要的时间就越长，同时容易引起**“维度灾难”**，使得模型更加复杂。
  - **特征选择**通过剔除不相关的特征或冗余的特征来**减少特征数量**，从而简化了模型并且提升了模型的泛化能力。



特征选择的过程

特征子集产生过程

- 如何生成特征子集？

特征选择的本质上是一个组合优化的问题，求解组合优化问题最直接的方法就是搜索.根据不用的搜索策略，可以将搜索的算法分为**完全搜索**(Complete), **启发式搜索**(Heuristic) 和**随机搜索**(Random) 三大类。

1. 采用**全局最优搜索策略**的过程产生方法

全局最优搜索策略可以分为穷举搜索与非穷举搜索两类。穷举搜索策略有遍历所有特征和以广度优先搜索的策略，这两种搜索策略都枚举了所有的特征组合，复杂度为2^n。

2. 采用**启发式搜索策略**的过程产生方法

启发式搜索的基本思想是增加关于要解决问题的解某些特征，以便指导搜索**向最有希望的方向**发展。启发式搜索是搜索是在搜索的最优性和计算量之间做一个折中的搜索策略。

3. 采用**随机算法搜索策略**的产生方法

特征选择本质上是一个组合优化问题，求解这类问题可采用非全局最优目标搜索方法，其实现是依靠带一定**智能的**随机搜索策略。(如模拟退火，遗传算法等)

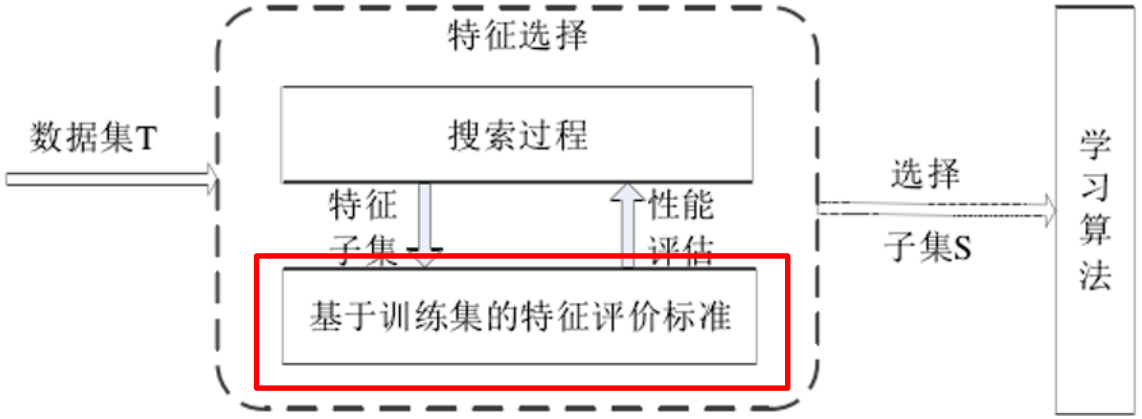
特征子集评价

如何评价特征子集？

不同的特征选择算法不仅对特征子集评价标准不同，有的还需要结合后续的学习算法模型。因此根据特征选择中子集评价标准和后续算法的结合方式主要分为**过滤式(Filter)**、**封装式(Wrapper)** 和**嵌入式(Embedded)**三种

1. 过滤式(Filter)评价策略方法

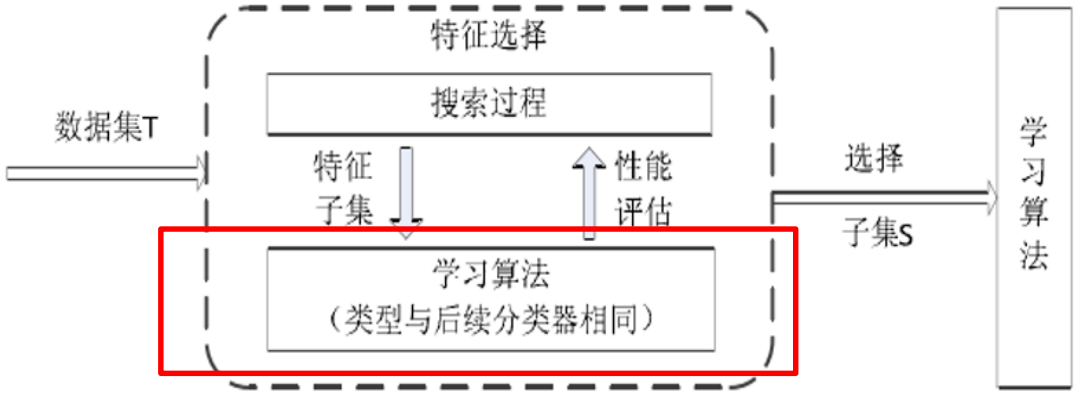
- 独立于后续的学习算法模型来分析数据集的固有的属性
- 采用一些基于信息统计的启发式准则来评价特征子集
- 启发式的评价函数：距离度量、信息度量、依赖性度量、一致性度量



特征子集评价

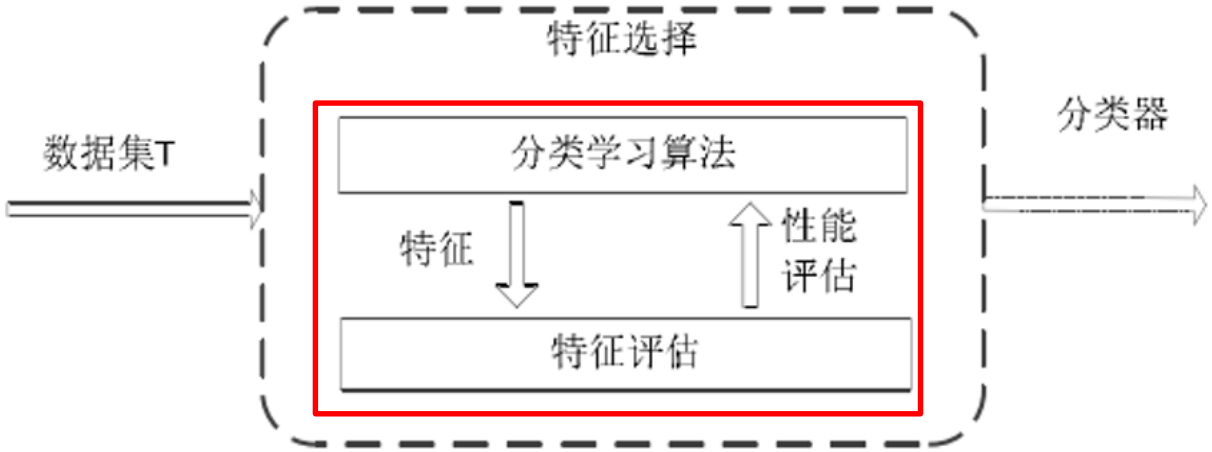
- 如何评价特征子集？
  - **2. 封装式(Wrapper)评价策略方法**
    - 将特征选择作为学习算法一个组成部分，需要结合后续的学习算法，并直接将学习算法的分类性能作为特征重要性的评价标准
    - 直接使用**分类器的性能作为评价的标准**，选出来的特征子集对分类一定有最好的性能

相对于Filter 选择方法，Wrapper 方法所选择的特征子集的规模要小得多，有利于关键特征的辨识，模型的分类性能更好。但Wrapper 方法泛化能力较差，当改变学习算法时，需要针对该学习算法重新进行特征选择，算法的计算**复杂度高**.



特征子集评价

- 如何评价特征子集？
  - **3. 嵌入式(Embedded)评价策略方法**
    - 特征选择算法嵌入到学习和分类算法中，**特征选择是算法模型中的一部分**
    - 算法模型训练和特征选择同时进行，互相结合。即，算法具有自动进行特征选择的功能



特征子集评价

如何评价特征子集？

3. 嵌入式(Embedded)评价策略方法

1). 带惩罚项的特征选择方法

- 在模型损失函数上加上一个惩罚项，模型训练时通过惩罚项来**对特征的系数进行惩罚处理**，而在特征选择方法中常使用的是L1 正则化(regularization)项

线性回归的损失函数：$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2$

岭回归的损失函数：$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2+\lambda\sum_{j=1}^{n}\theta_j^2$

Lasso回归的损失函数：$J(\theta)=\frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)^2+\lambda\sum_{j=1}^{n}\left|\theta_j\right|$

http://blog.csdn.net/hzw19920329

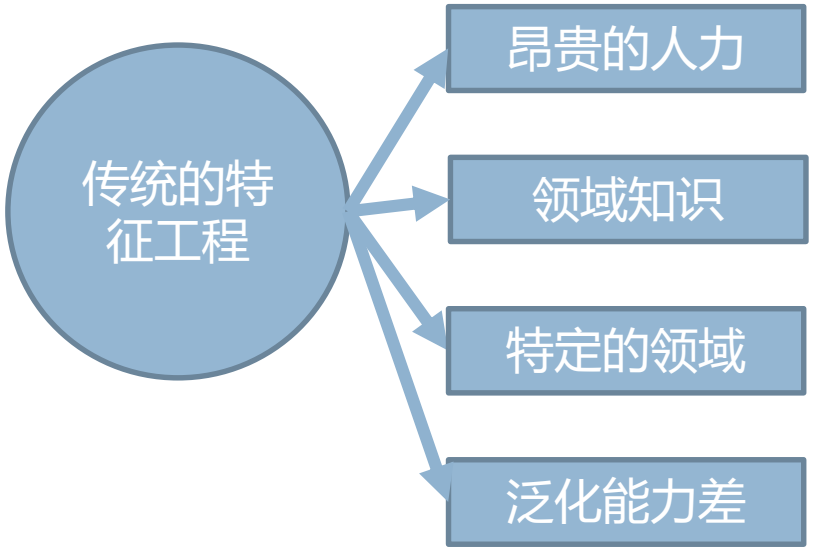
- **2范数，让系数的取值变得平均。对于关联特征，能够获得更相近的对应系数**
- **1范数，系数w的l1范数作为惩罚项加到损失函数上，由于非零，迫使那些弱的特征所对应的系数变成0，达到特征选择**

2). 基于树模型的特征选择方法

- 每一步都必须选择一个特征，将样本集划分为纯度更高的子集，而每次选择出的都是使划分效果最佳的特征。**决策树、迭代决策树(GBDT)、随机森林**
- **第四章：数据挖掘算法**

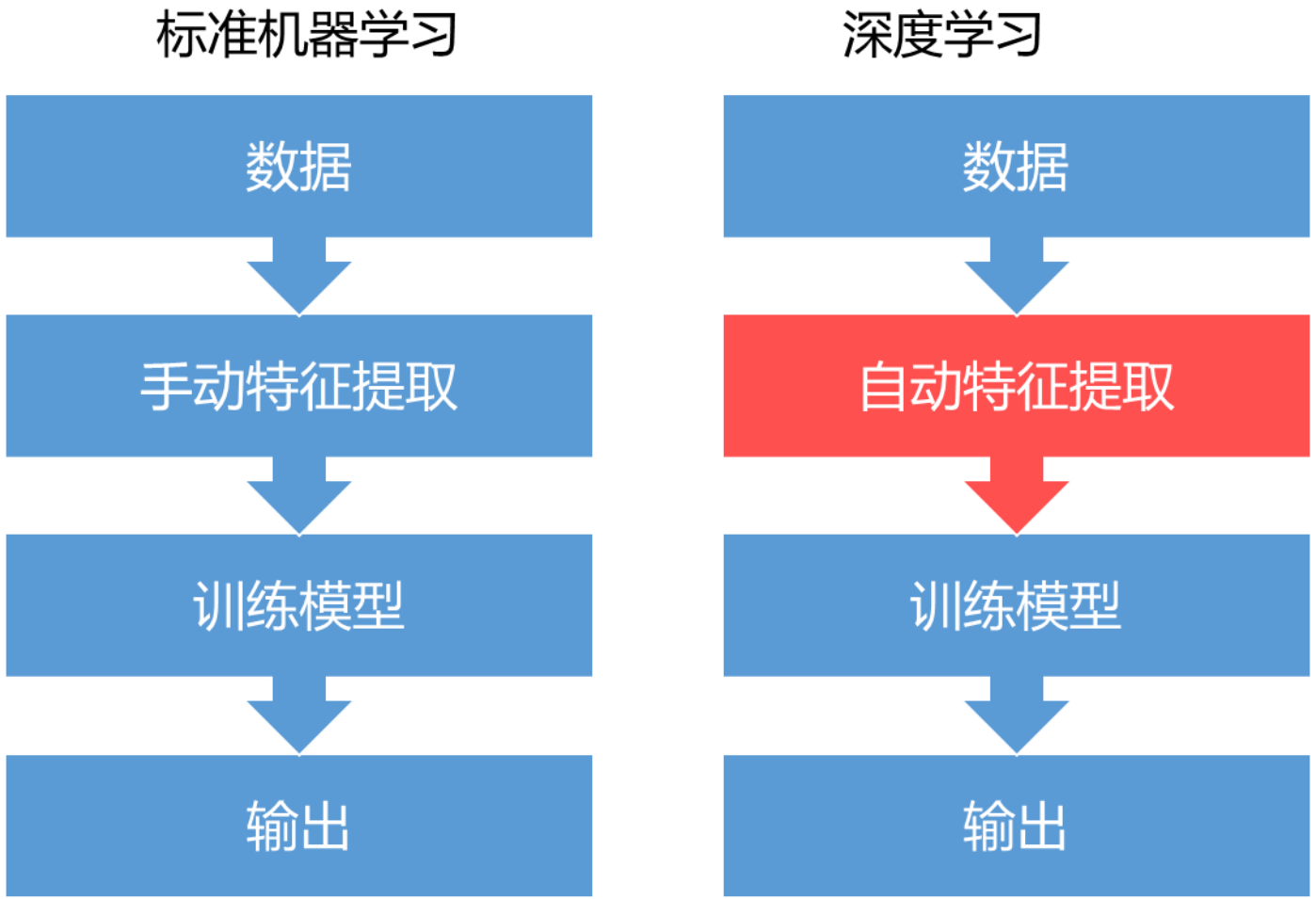
传统特征工程的缺点

- 传统特征工程的缺点



传统特征工程的缺点

- 传统特征工程的缺点

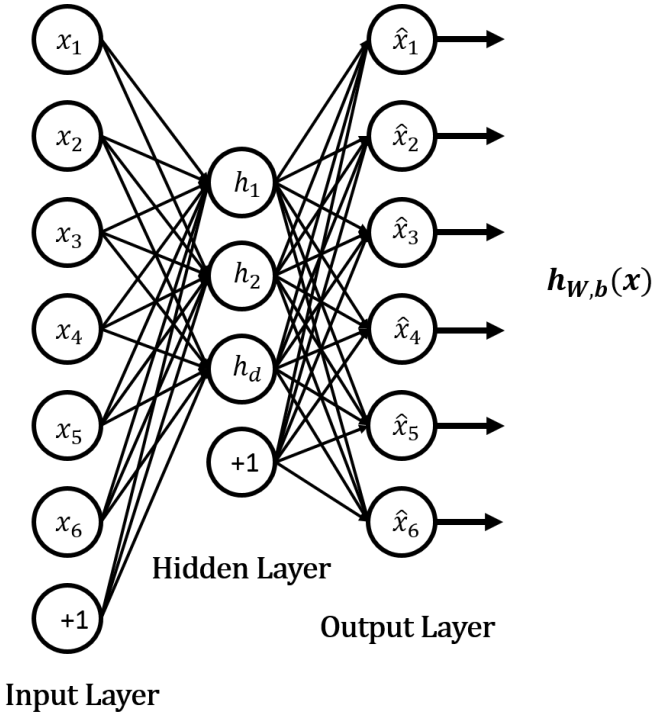


特征学习

- 特征学习

  如何从数据中能够自主的学习特征，在这里我们主要介绍在深度学习中常用的三种网络结构。

  - 自编码结构(Auto-Encoder)



将数据的特征X作为Input Layer输入同样将原始数据特征X′作为Output Layer的输出来重构出原数据。

Encoder: $H = f(A * X + b)$

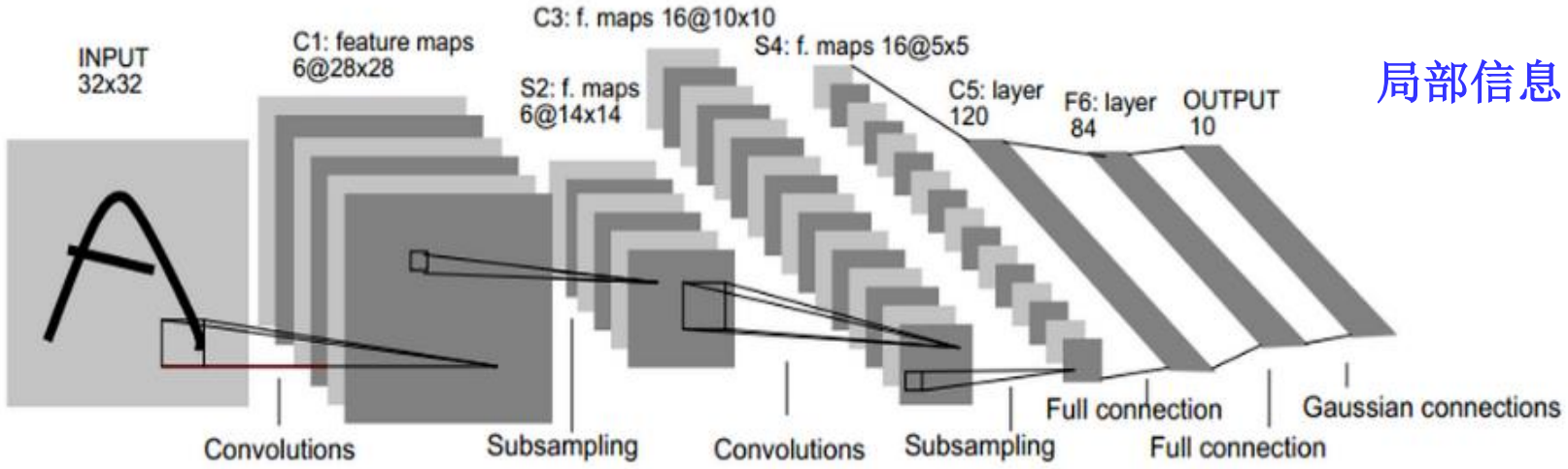
Decoder: $X' = f(A' * H + b')$

将中间的隐含层H的输出作为学习到的数据特征。

特征学习

- 卷积神经网络(CNN): 常用于图像特征提取

LeNet



局部信息

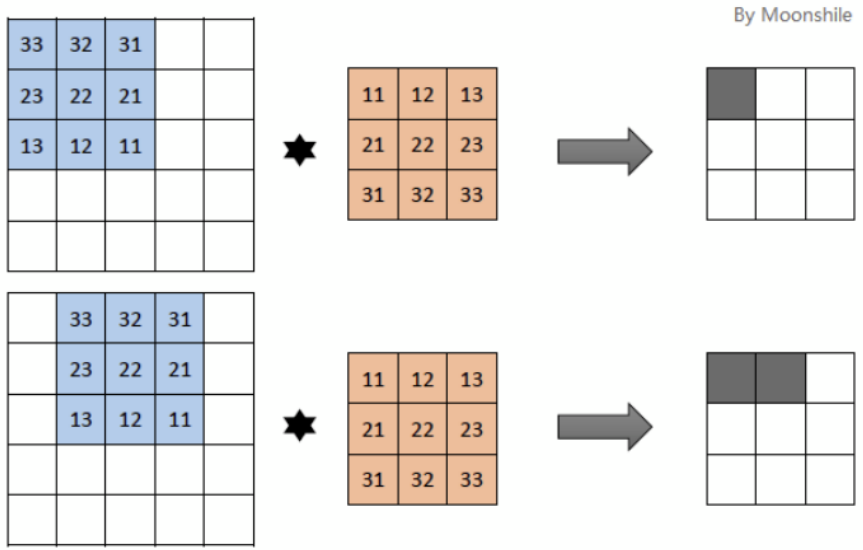
卷积层：通过局部平移，利用不同的卷积核来提取图像中不同的特征
池化层：计算某个区域的特征，提高模型的泛化能力
全连接层：通过多层的神经网络，抽取更高阶的特征。

最终全连接层的输出即为该图像的特征向量表示。

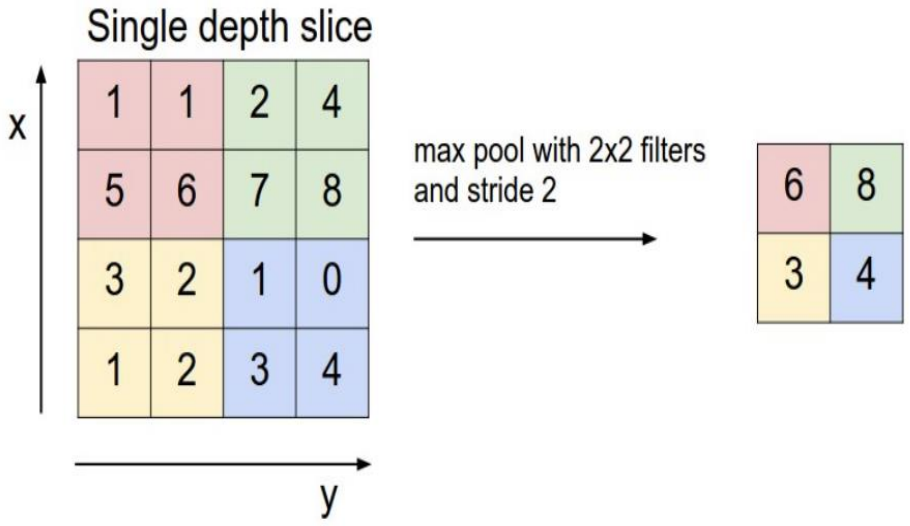
特征学习

- 卷积神经网络(CNN): 常用于图像特征提取

卷积操作

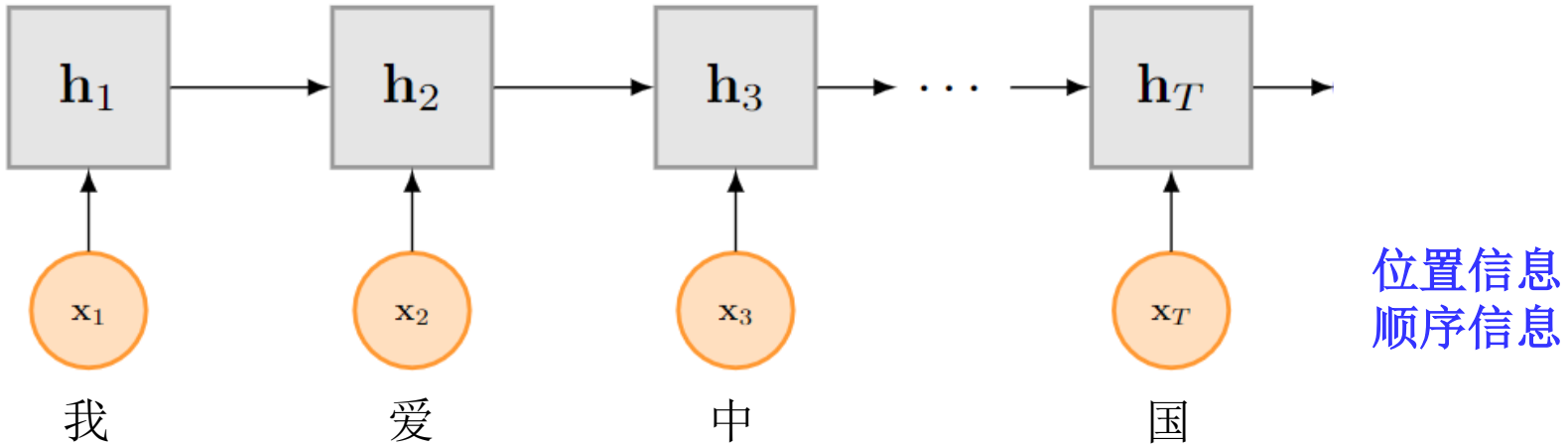


池化操作



特征学习

- 循环神经网络(RNN): 常用于序列数据的特征提取



将序列中的每个数据依次作为RNN的输入，如上图中的文本数据‘我’、‘爱’、‘中’、‘国’，并将最后一层网络的输出h_T作为最终序列数据的特征向量

Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, Guoping Hu, EKT: Exercise-aware Knowledge Tracing for Student Performance Prediction, IEEE TKDE), 33(1): 100-115,2021

特征学习

- 利用标准数据集进行特征学习（特征预训练）
  - 作用：模型效果验证 & 应用问题中的模型预训练
  - 图像数据预训练：ImageNet
    - http://www.image-net.org/
    - 1400万张图片数据，2万类别，已标注
    - 常用模型：ResNet，AlexNet，VGG等
    - 常见应用：图像分类、目标检测、目标定位，场景分类等
  - 文本数据预训练：Twitter，Wiki
    - https://nlp.stanford.edu/projects/glove/
    - 2 Billon tweets, 27 Billion 词数，1.2M 词表
    - 常用模型：CBOW，Skip-gram，Glove等Word2Vec模型
    - 常见应用：文本分类，文本推理，翻译等

训练好的特征即可直接作为其它模型的输入来使用

特征学习

Efficient estimation of word representations in vector space

T Mikolov, K Chen, G Corrado, J Dean - arXiv preprint arXiv:1301.3781, 2013 - arxiv.org

We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing …

☆ 99 被引用次数：24421 相关文章 所有 43 个版本 »

- Word2Vec—自然语言处理的预训练
  - 哪句话更像自然语句

S1: 语言模型的本质是对一段自然语言的文本进行预测概率的大小
S2: 语言模型的本质是对自然一段语言的文本进行预测概率的大小
S3: 语言模型的本质是对自然语言一段的文本进行预测概率的大小

  - 计算词构成句子的概率—最大化

$$P(w_1, w_2, \dots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2) \dots P(w_n|w_1, \dots, w_{n-1})$$

$$L = \sum_{w \in C} \log P(w|context(w))$$

特征工程：可解释性

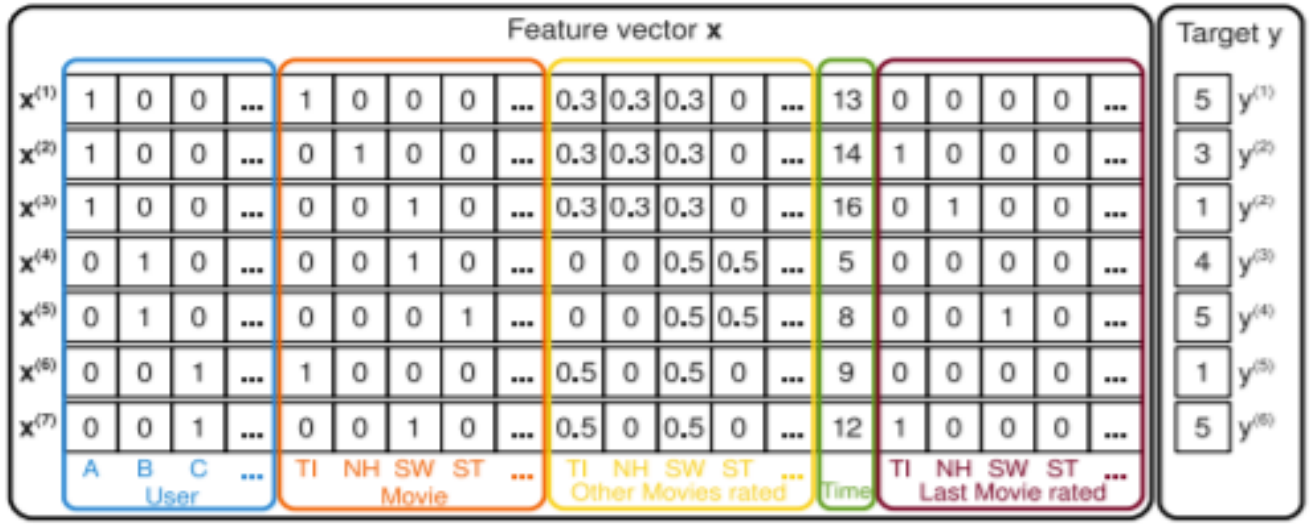
Factorization machines

S Rendle - 2010 IEEE International confere

In this paper, we introduce Factorization Ma
combines the advantages of Support Vecto
SVMs, FMs are a general predictor working

☆ 99 被引用次数：1862 相关文章

- 特征的含义
  - 人工设计的特征
  - 特征的构造基于先验知识，天然具有可解释性

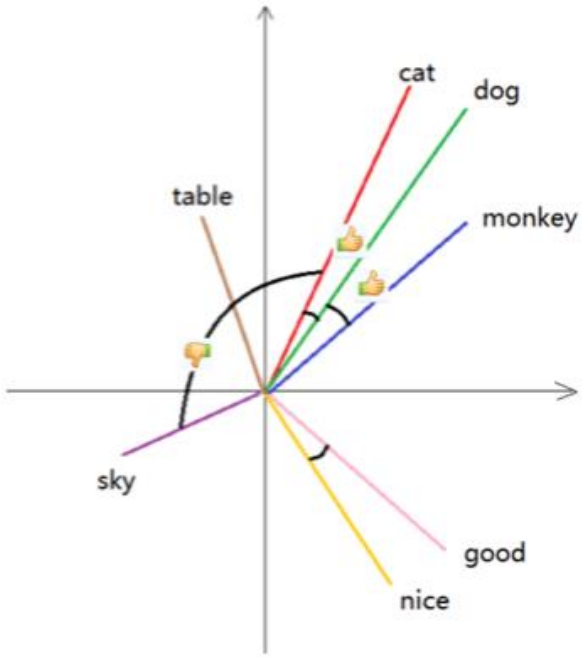


$$S = \{(\mathrm{A}, \mathrm{TI}, 2010\text{-}1, 5), (\mathrm{A}, \mathrm{NH}, 2010\text{-}2, 3), (\mathrm{A}, \mathrm{SW}, 2010\text{-}4, 1),$$
$$(\mathrm{B}, \mathrm{SW}, 2009\text{-}5, 4), (\mathrm{B}, \mathrm{ST}, 2009\text{-}8, 5),$$
$$(\mathrm{C}, \mathrm{TI}, 2009\text{-}9, 1), (\mathrm{C}, \mathrm{SW}, 2009\text{-}12, 5)\}$$

特征工程：可解释性

- 特征的含义
  - Word2Vec—自然语言处理的预训练
  - 学习语言的语义特性：解决“语义鸿沟”

词的相似性



词的类比性

King – Queen ~= Man – Woman

China – Beijing ~= UK – London ~= Capital

Efficient estimation of word representations in vector space

T Mikolov, K Chen, G Corrado, J Dean - arXiv preprint arXiv:1301.3781, 2013 - arxiv.org

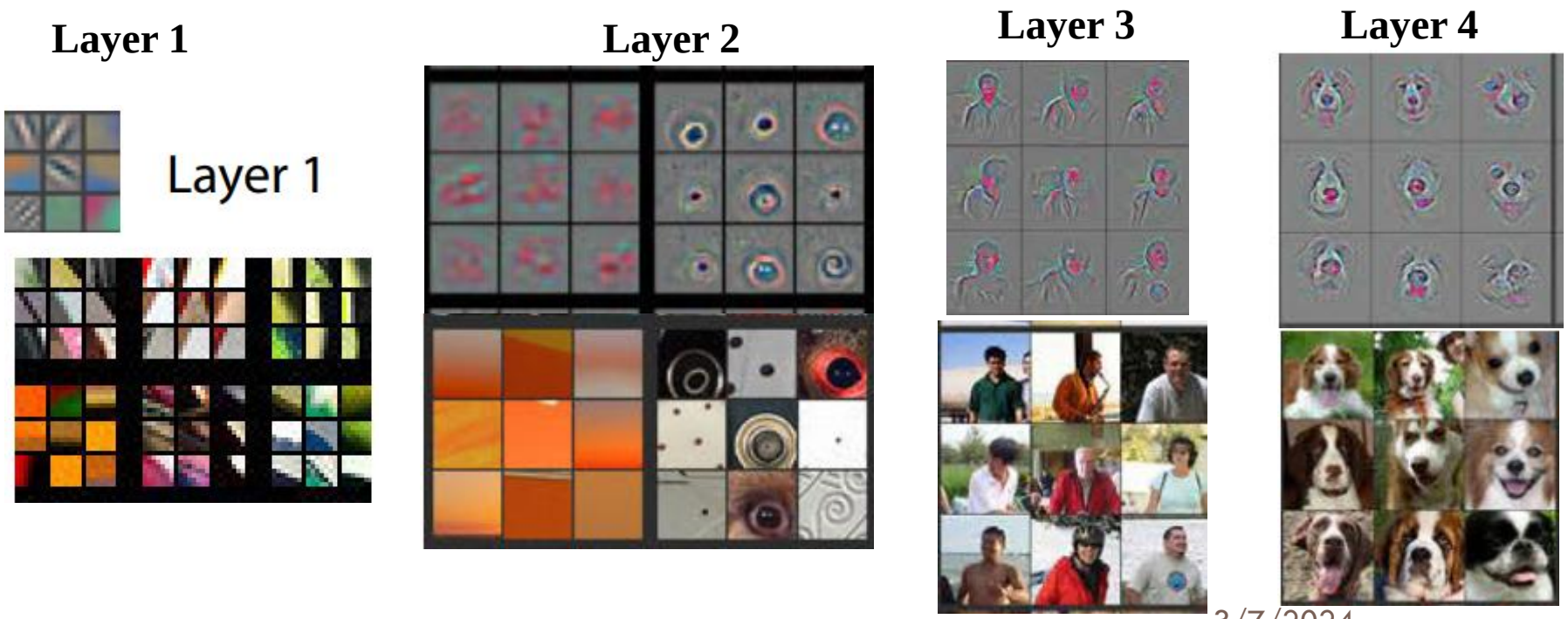
We propose two novel model architectures for computing continuous vector representations of words from very large data sets. The quality of these representations is measured in a word similarity task, and the results are compared to the previously best performing …

☆ 99 被引用次数：24421 相关文章 所有 43 个版本 »

特征工程：可解释性

Visualizing and understanding convolutional networks
MD Zeiler, R Fergus - European conference on computer vision, 2014 -
Abstract Large Convolutional Network models have recently demonstra
classification performance on the ImageNet benchmark Krizhevsky et a
is no clear understanding of why they perform so well, or how they migh
paper we explore both issues. We introduce a novel visualization techn
insight into the function of intermediate feature layers and the operation
Used in a diagnostic role, these visualizations allow us to find model arc
☆ 99 被引用次数：13612 相关文章 所有 18 个版本

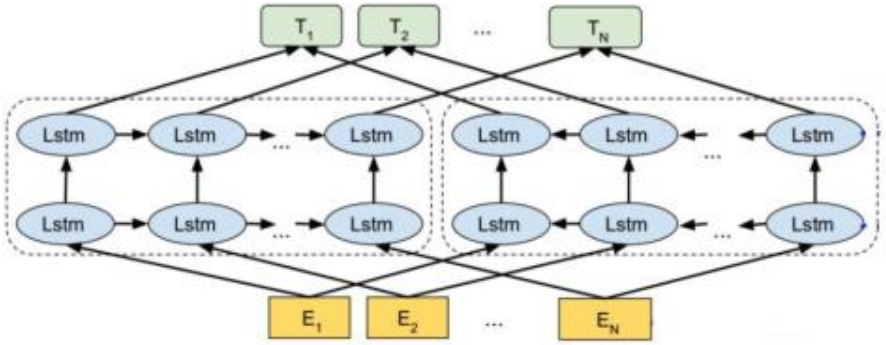
- 特征的含义—可解释性
  - CNN Feature map——特征图可视化
  - 学习图像的特点：图形图像的“颜色，边角，轮廓，图形”等



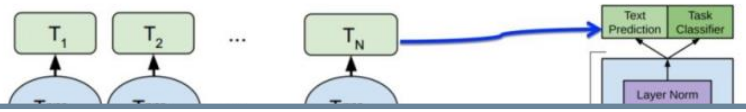
Zeiler, Matthew D., and Rob Fergus. "Visualizing and understanding convolutional networks." European conference on computer vision. Springer, Cham, 2014.

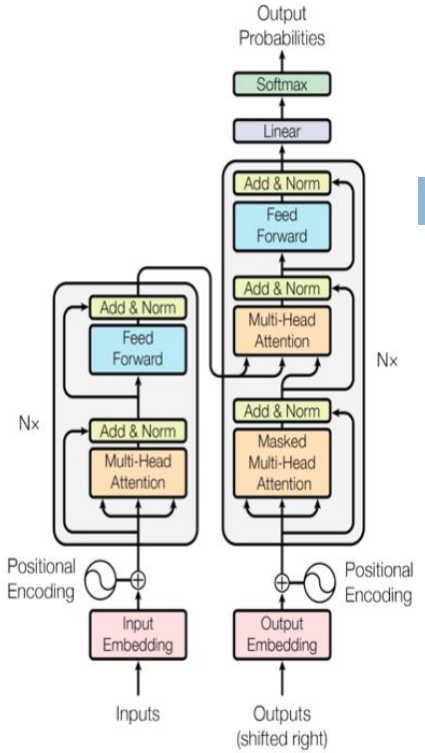
特征学习

- 自然语言处理的预训练模型
  - Elmo，GPT，Transformer, BERT

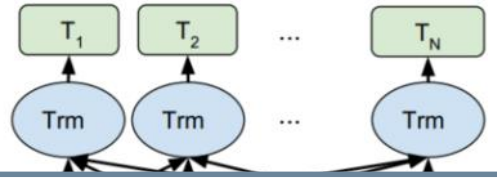


OpenAI GPT





BERT (Ours)



特征学习过程已经不局限于人工的思考、构造、统计等方法。它已经成为一个重要的研究方向，专门的特征学习模型已经在CV、NLP等领域取得重要的突破。

参考文献

- 书籍
  - 数据挖掘导论
  - 机器学习
- 论文
  - 《An Introduction to Variable and Feature Selection》
  - 《特征选择常用算法综述》
- 实战经验
  - Sklearn官方文档
  - Kaggle和天池比赛论坛

第二章数据分析基础小结

- 数据采集 Data Collection
  - 信息检索
  - 网络爬虫
- 数据存储 Data Storage
- 数据预处理 Data Preprocessing
  - 数据清洗
  - 数据集成
  - 数据变换
  - 数据规约
- 特征工程 Feature Engineering
  - 特征设计
  - 特征理解