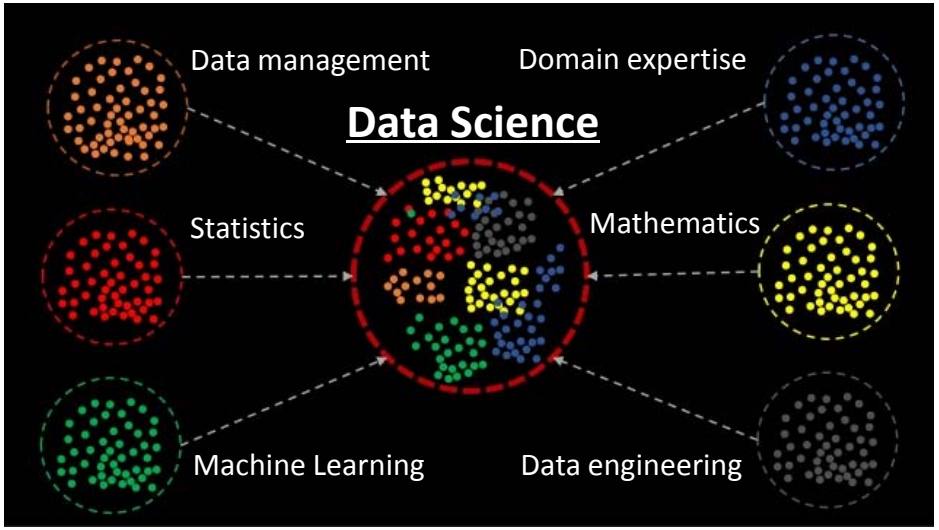
如何建设数据专业的第一门导论课程？

王 伟
wwang@dase.ecnu.edu.cn
華東師範大學

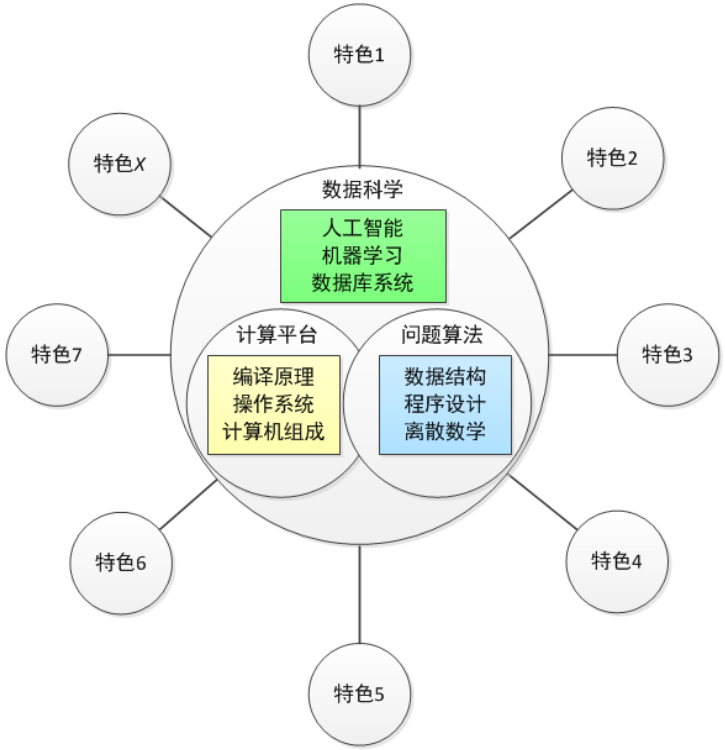
Outline

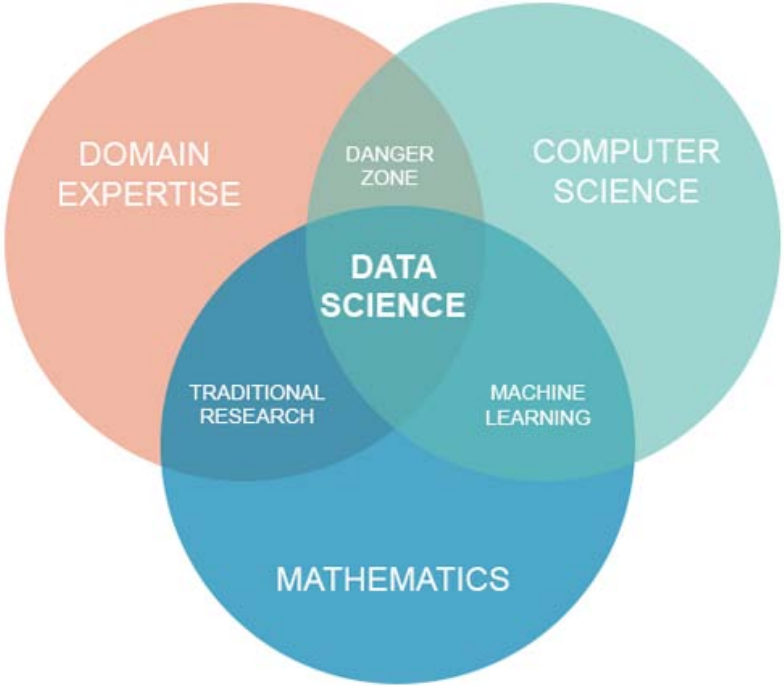
- 背景与观点
- 体系与内容
- 实训与工具
- 结论与展望



背景1：科学范式与数据思维

背景2：计算机专业 vs. 数据专业



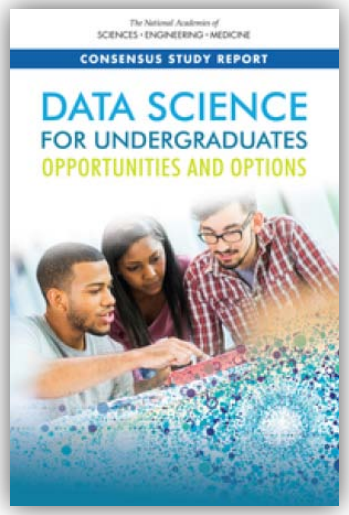


CCF, 计算机科学与技术专业培养方案编制指南, 2018.9.

Drew Conway's Venn Diagram of Data Science

背景3： 数据科学本科体系与能力模型

- Mathematical foundations
- Computational foundations
- Statistical foundations
- Data management and curation
- Data description and visualization
- Data modeling and assessment
- Workflow and reproducibility
- Communication and teamwork
- Domain-specific considerations
- Ethical problem solving



数据能力模型

背景4：IT + Community + Education

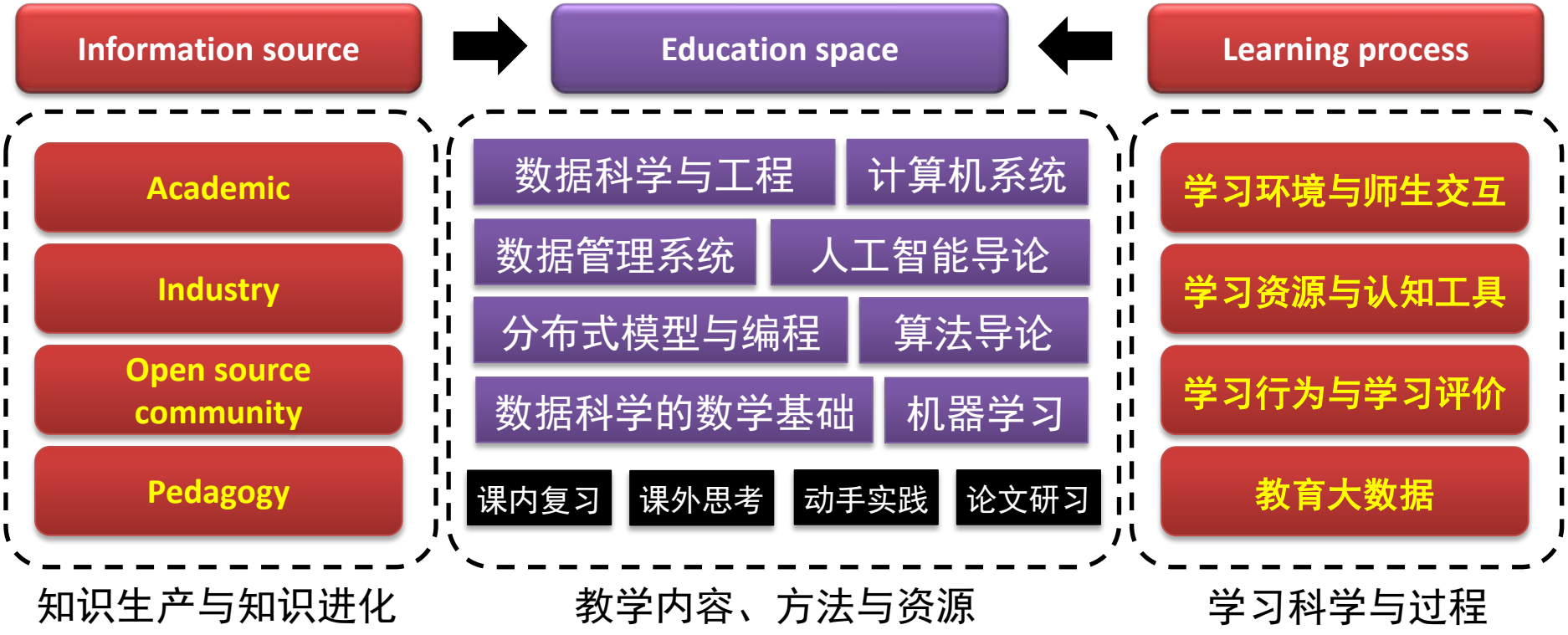
企业 — **开源社区** — **高校**

Information technology | **Organization & Community** | **Education technology**

- Big Data
- Cloud computing
- Blockchain
- Artificial intelligence

- MOOX + 新工科
- 教育信息科学与技术
- 人工智能/CS4All教育
- 知识经济/共享经济

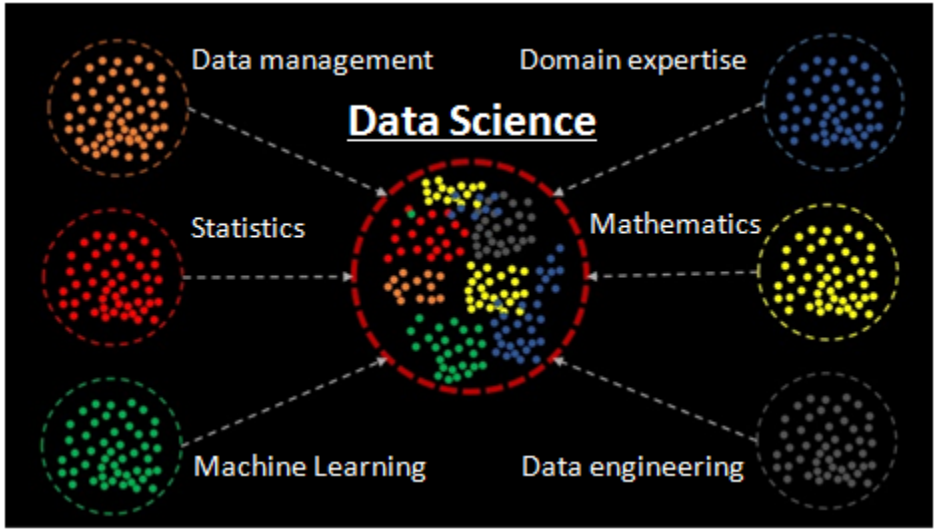
背景5：Education × IT = New Learning Science



Remark

- 数据科学全世界范围内的普及
- 学科之间在不断的持续融合（CS/DS/Math/Stat/…）
- 数据专业建设的持续深入
- 数据科学专业的建设需要“学校-企业-社区”深度结合
- 作为新工科下的新专业需要信息技术与基础设施的支持

《数据科学与工程导论》体系与内容



数据科学与大数据教学

数据系

多维数据、
图形图像、
自然语言、
Web…

博

平台系

Hadoop、
Hbase、
NoSQL、
Spark…

大

算法系

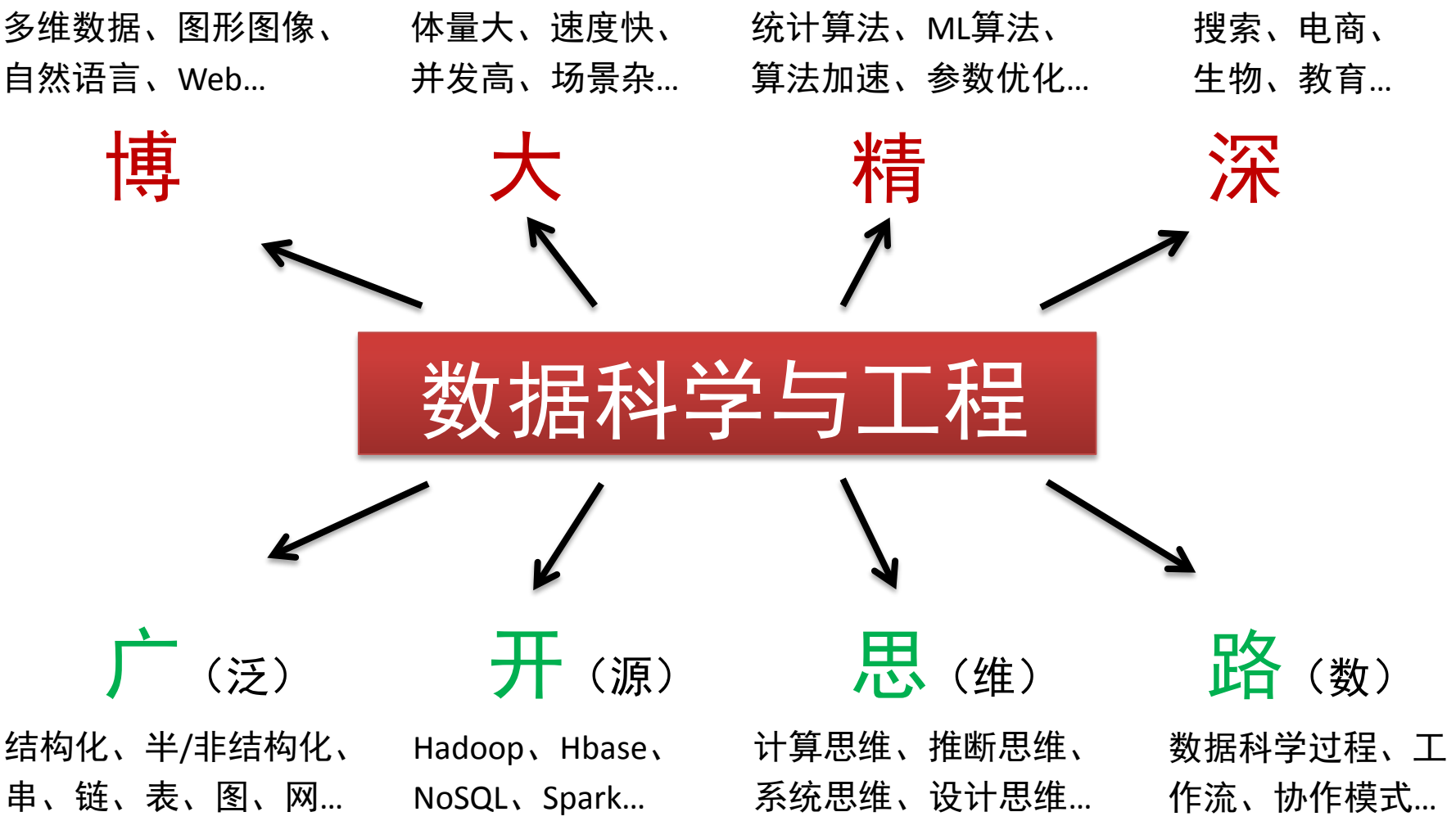
数据挖掘、
机器学习、
算法加速、
参数优化…

精

应用系

搜索大数据、
电商大数据、
生物大数据、
教育大数据…

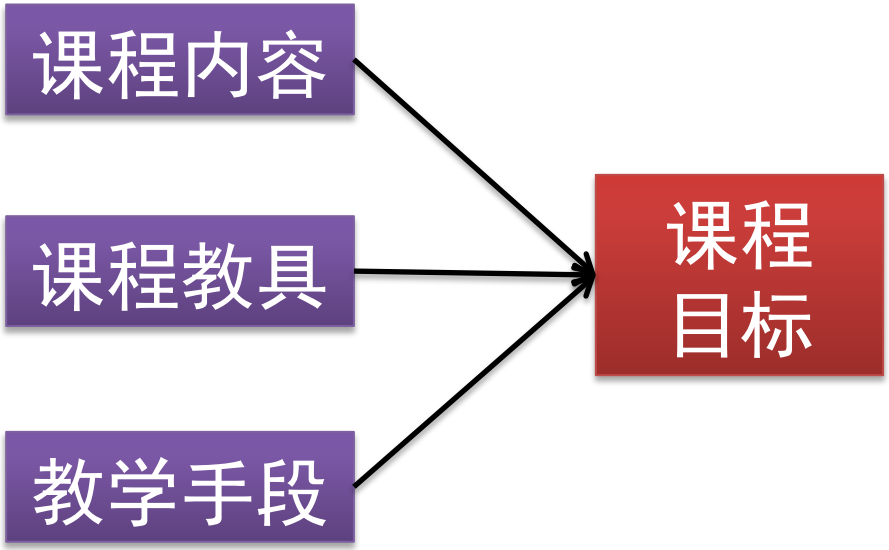
深


多维数据、图形图像、
自然语言、Web…
体量大、速度快、
并发高、场景杂…
统计算法、ML算法、
算法加速、参数优化…
搜索、电商、
生物、教育…
博
大
精
深
数据科学与工程
广（泛）
开（源）
思（维）
路（数）
结构化、半/非结构化、
串、链、表、图、网…
Hadoop、Hbase、
NoSQL、Spark…
计算思维、推断思维、
系统思维、设计思维…
数据科学过程、工
作流、协作模式…

课程目标

- 了解数据专业全貌，建立数据思维的意识；
- 掌握数据科学与工程的基本内涵和应用模式；
- 培养以数据为中心的问题求解能力，系统性的学习数据科学与工程的核心原理与关键技术；
- 培养开源开放的精神，建立基于开源工具的数据分析与处理意识，并做到初步的数据编程训练；
- 让大家感受到数据与计算的美，数据与计算的愉悦；
- 点燃大家对数据专业的热情与兴趣！

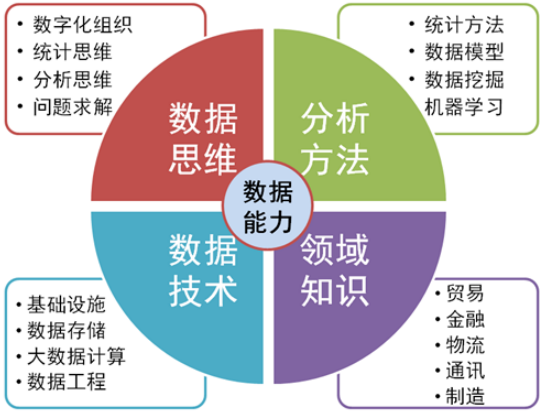
如何达成这样一个教学目标？



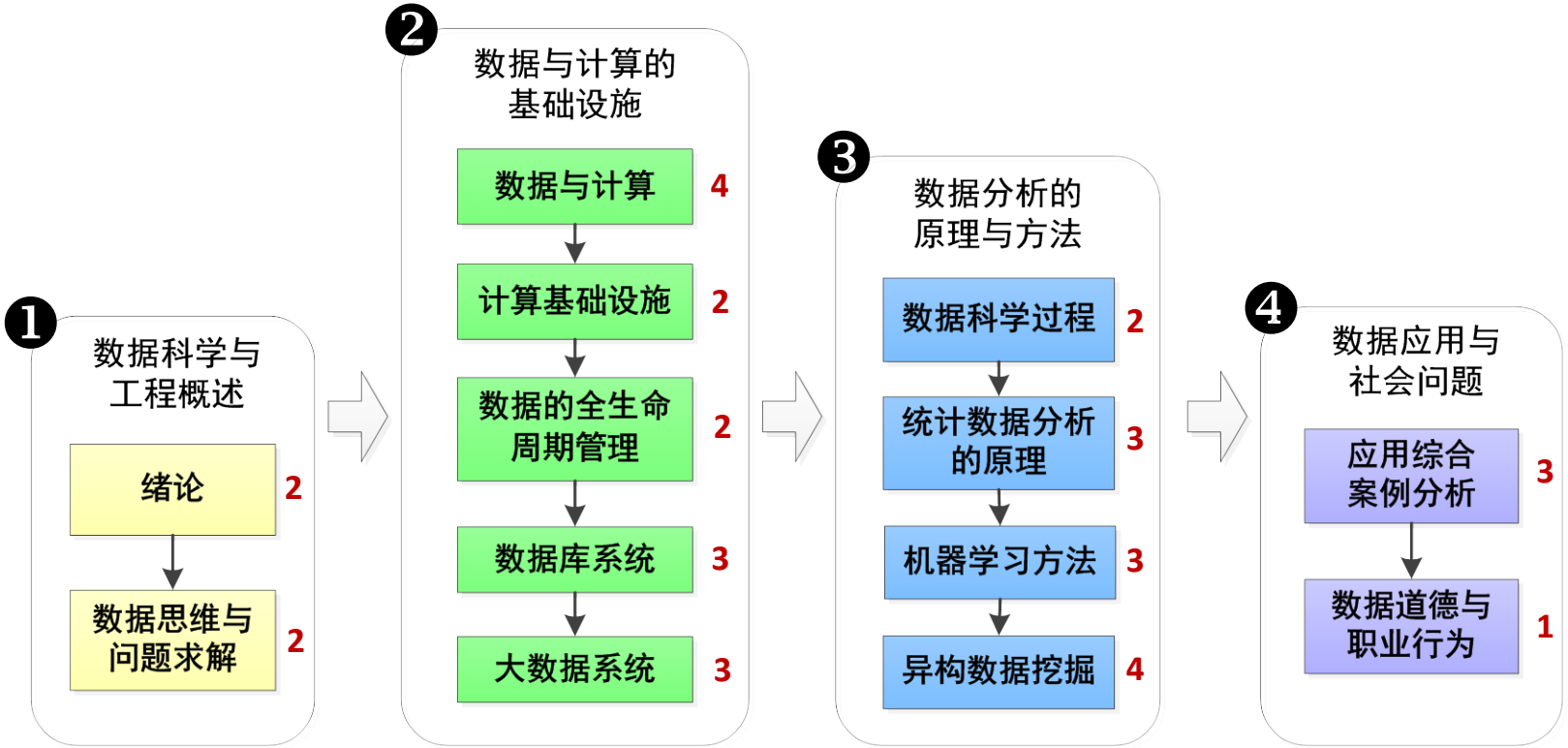
关键1：完整的知识框架

四条线贯穿起来：

1. 数据思维：以数据为中心的问题求解
   - 计算思维 + 分析思维
2. 基础设施：数据管理的全生命周期技术
   - 采集、存储、计算、分析、展示
3. 分析方法：统计与算法重新定义世界
   - 基本分析方法：统计、算法
   - 进阶工具平台：数据挖掘、机器学习、工程平台
4. 开源实践：Python语言（生态）、SQL、Hadoop/Spark、KNIME

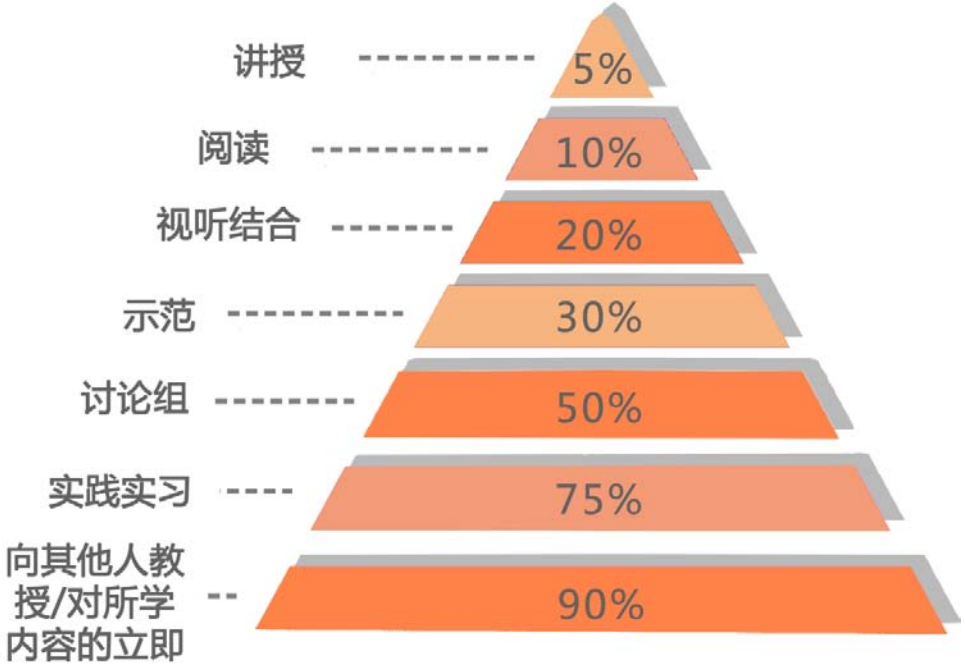


课程大纲（34学时理论课）

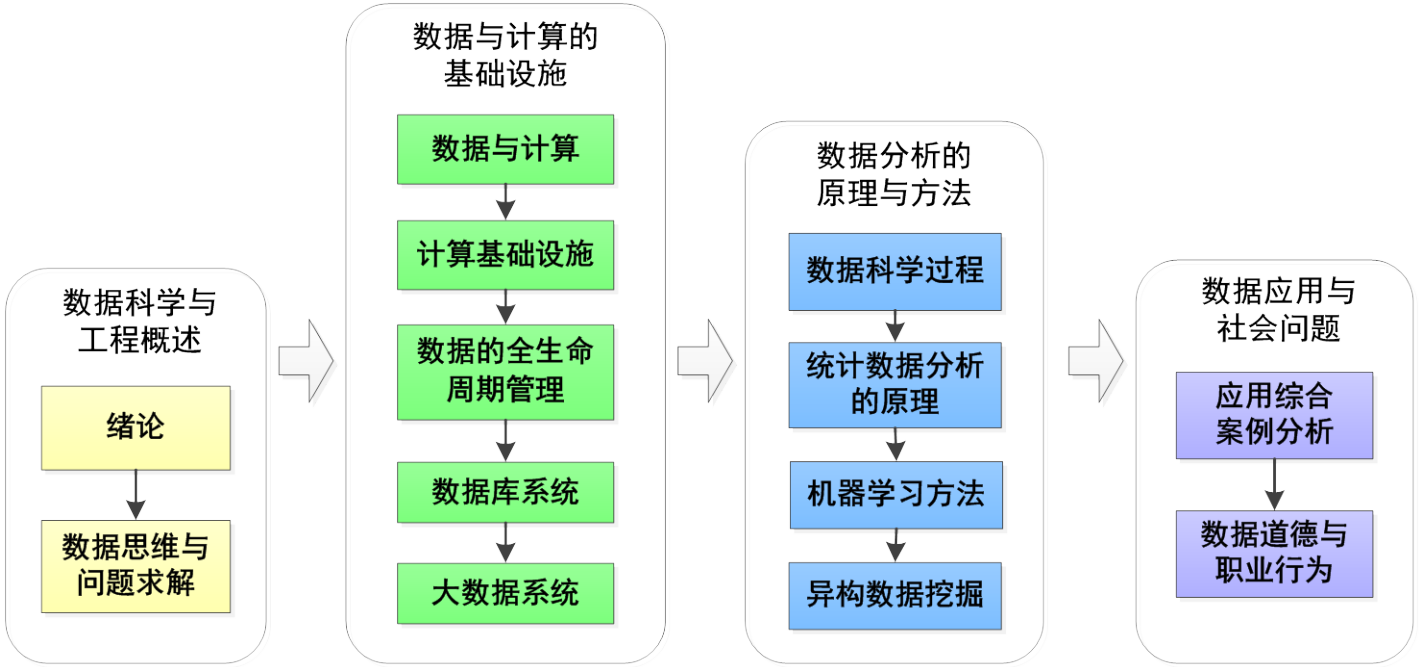


教学主体思路

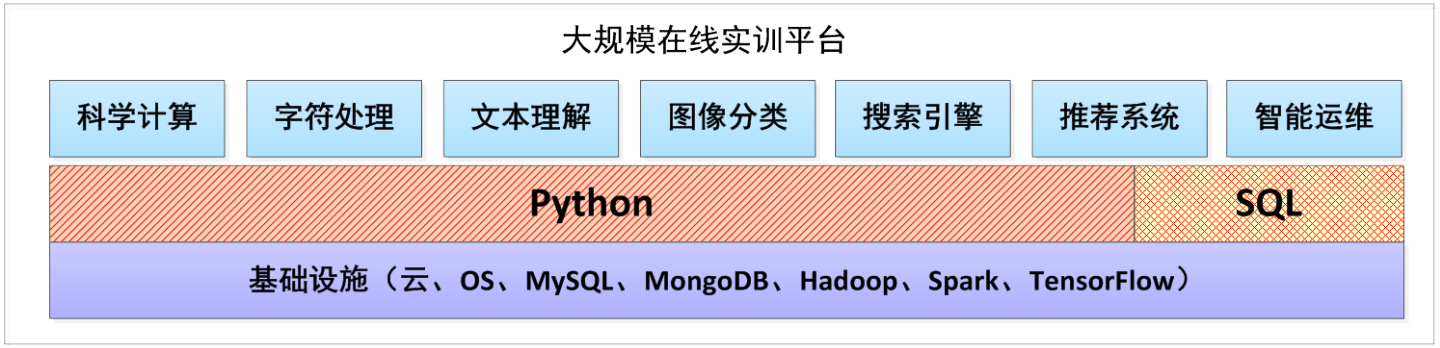
激发思辨，建立意识



动手实践，训练思维


数据科学与
工程概述
绪论
数据思维与
问题求解
数据与计算的
基础设施
数据与计算
计算基础设施
数据的全生命
周期管理
数据库系统
大数据系统
数据分析的
原理与方法
数据科学过程
统计数据分析
的原理
机器学习方法
异构数据挖掘
数据应用与
社会问题
应用综合
案例分析
数据道德与
职业行为

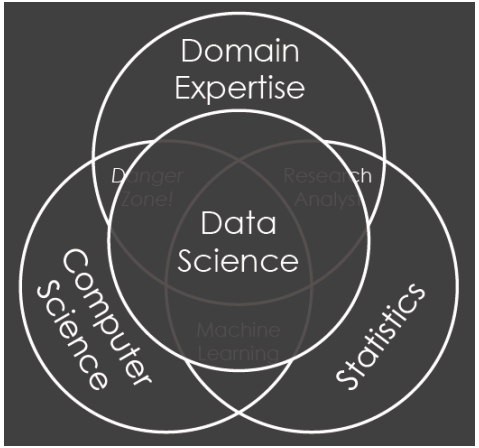
34学时


大规模在线实训平台
科学计算
字符处理
文本理解
图像分类
搜索引擎
推荐系统
智能运维
Python
SQL
基础设施（云、OS、MySQL、MongoDB、Hadoop、Spark、TensorFlow）

34学时

第1章 绪论

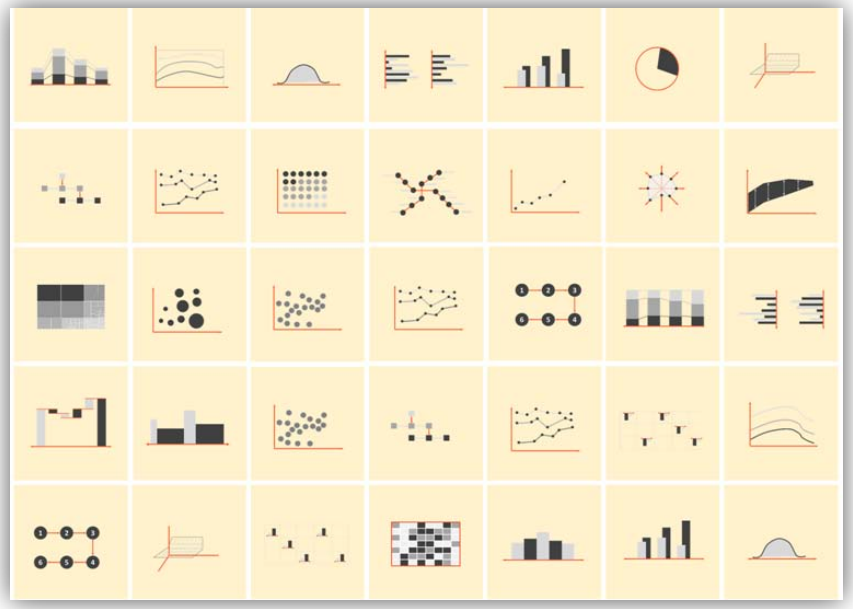
- 信息文明与数据简史
- 数据科学与工程的基本内涵
- 第四范式：数据密集型科学
- 比特与算法重构我们的世界
- 实训基础：Git、Python与KFCoding
- 实训Lab-01：Git与Python基础



什么是数据专业？

- 我们认为，数据专业（或数据学科）至少包括下面四个方面的内容：
  1. **数据学（Dataology）**：研究探索赛博空间中数据界（data nature）奥秘的理论、方法和技术，研究的对象是数据界中的数据，研究认识数据的各种类型、状态、属性及变化形式和变化规律，即用科学的方法研究数据；
  2. **数据科学（Data science ）**：是以数据为中心，通过计算思维与数据思维的方法，来理解我们所处的世界，并实现问题的求解，即用数据的方法研究科学；
  3. **数据工程（ Data engineering ）**：支持上述两类活动的工程实现，包括数据基础设施、数据全生命周期管理过程、数据科学过程方法论和工具、数据处理与分析系统、数据分析编程语言、可视化工具等；
  4. **数据道德与职业行为准则（ Data of Ethics & Professional Conduct）.**

Gastronomy（美食学）和Dataology（数据学）



什么是数据科学？

- 定义： The application of **data centric**, **computational**, and **data thinking** to

understand the world **Science**

&

solve problems **Engineering**

Data science is fundamentally interdisciplinary

什么是数据工程？

- 支持数据学和数据科学研究和活动的工程实现，包括：
  - 数据基础设施
  - 数据全生命周期管理过程
  - 数据科学过程方法论和工具
  - 数据处理与分析系统
  - 数据分析编程语言
  - 可视化工具
  - ……

数据道德与职业行为准则

- 数据隐私与安全
- 数据道德与数据伦理
- 开放数据
- 数据关联的社会问题
- 数据相关的职业规划
- Code and data ethics

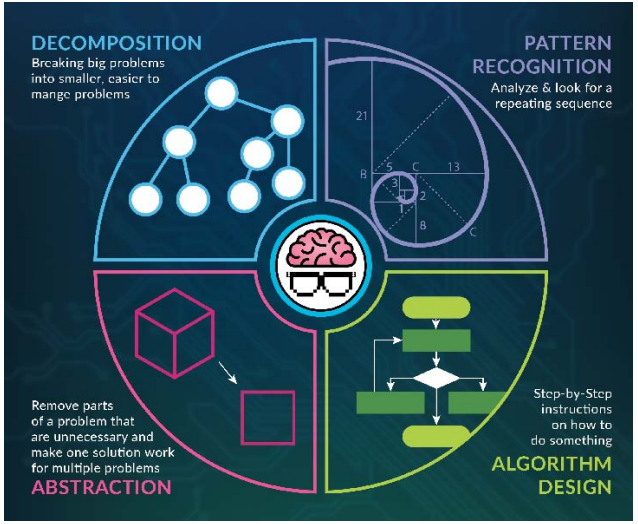
第2章 数据思维与问题求解

- 问题求解与思维方式
- 计算思维与数据思维
- 问题求解的实例
- 实训Lab-02：Python问题求解

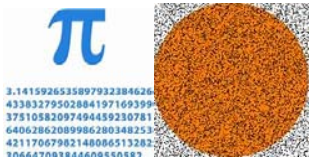
常见的计算思维方法

Abstraction（抽象）和**Automation（自动化）**是计算思维的两大核心特征，常见的计算思维方法包括：

- 分而治之：把数据、过程或问题分解成更小的、易于管理或解决的部分；
- 模式识别：观察问题的模式、趋势和规律；
- 抽象：识别模式形成背后的一般原理；
- 算法设计：为解决某一类问题撰写一系列详细的步骤。



计算思维与数据思维



Machine Learning

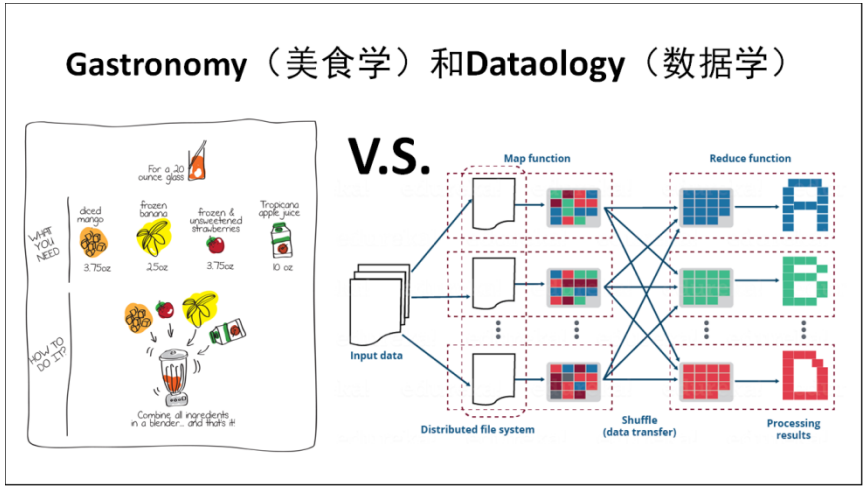
（数据为中心的）问题求解

数据思维

计算思维

第3章 数据与计算

- 比特与数据
- 数据的二进制表示
- 数据的模型
- 数据的结构
- 数据的计算：算法
- 算法分析与局限性
- 数据结构与算法的关系
- 计算机编程语言
- 实训Lab-03：数据的Python程序表达与计算



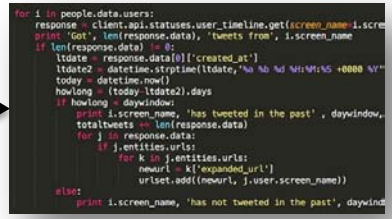
比特世界
(机械化)

程序表示

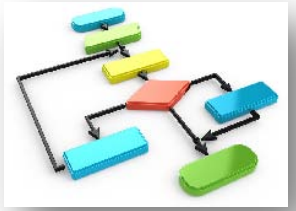
(符号程序)

代码实现

(编码工程)



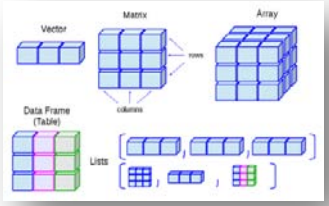
计算空间
(自动化)



算法结构 + 数据结构

(算法设计)

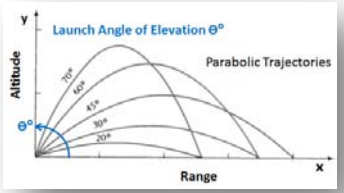
(编程语言)



模式数据

(数据化)

问题空间
(抽象化)



算法模型 + 数据模型

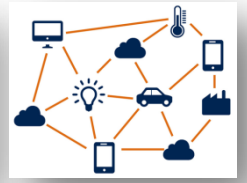
(问题求解)

(数据抽象)

数据

(数字化)

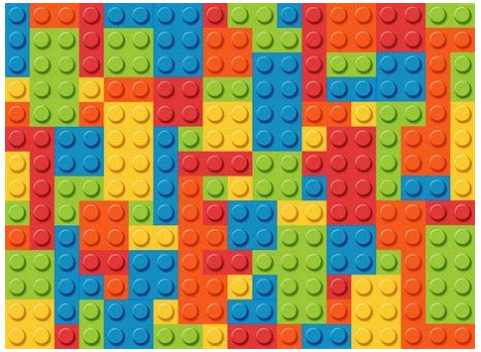
现实世界



模式化数据（Patterned Data）

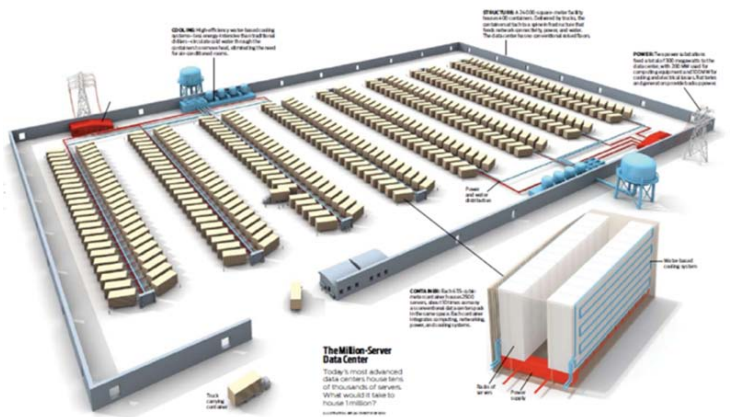
- 数据的多媒体类型：文本、图形、图像、音频、视频、……
- 数据的文件类型：.doc、.pdf、.ppt、.txt、……
- 数据的结构化类型：结构化、半结构化、非结构化
- 数据的程序表达类型：数值、字符、文本、向量、树、表、集合、关系、图、有限自动机、正则表达式、……

在计算领域，我们将现实世界中的事实或信息用编程语言提供的符号化手段进行表示，这种符号化表示称为数据（data）。



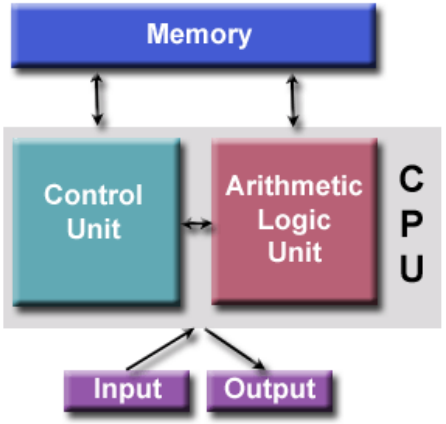
第4章 计算基础设施

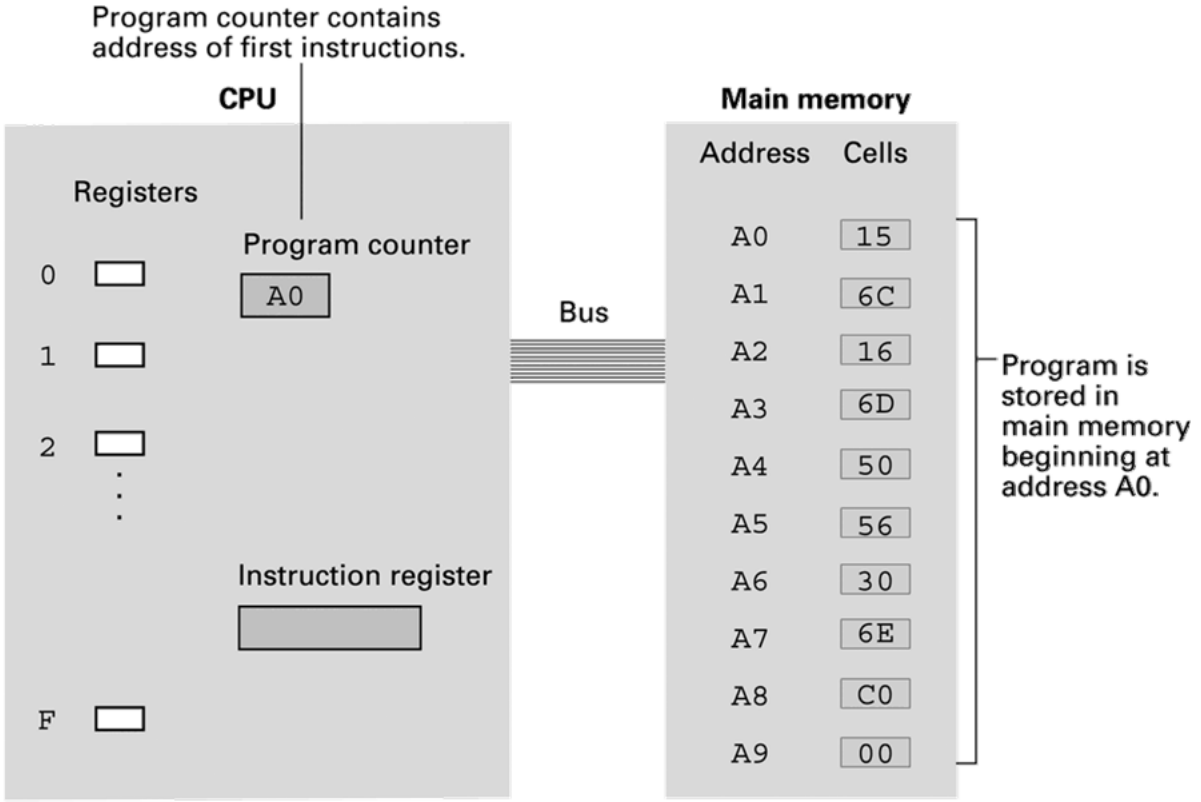
- 通用机器的思想
- 程序是如何执行的
- 计算机系统结构
- 云计算与数据中心
- 实训Lab-04：Python程序性能评测
  - 利用Python采集与分析CPU的信息
  - Python Performance基准测试



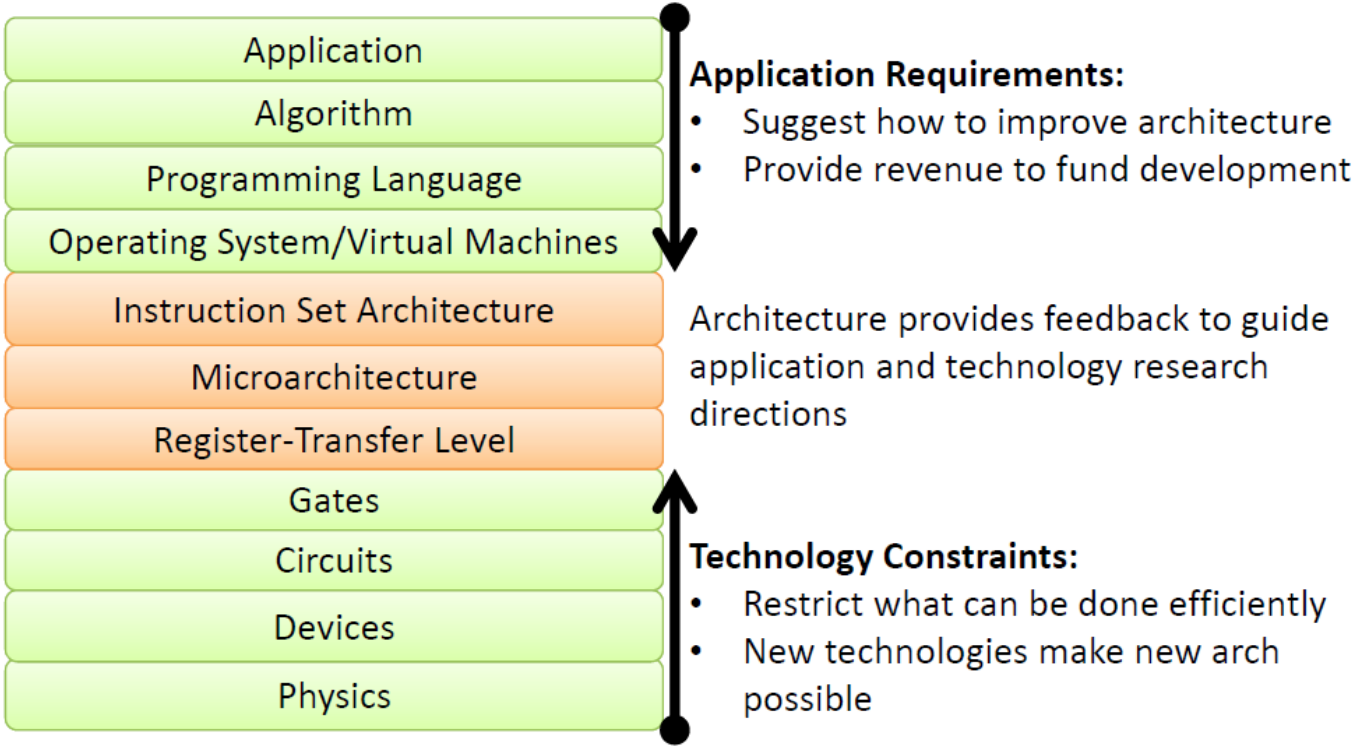
How to Make a Program “Run”?

- 存储程序控制
- 程序和数据都用二进制数表示
- 机器以CPU为中心





计算机系统结构


Application
Algorithm
Programming Language
Operating System/Virtual Machines
Instruction Set Architecture
Microarchitecture
Register-Transfer Level
Gates
Circuits
Devices
Physics
Application Requirements:
• Suggest how to improve architecture
• Provide revenue to fund development
Architecture provides feedback to guide application and technology research directions
Technology Constraints:
• Restrict what can be done efficiently
• New technologies make new arch possible

云计算的本质

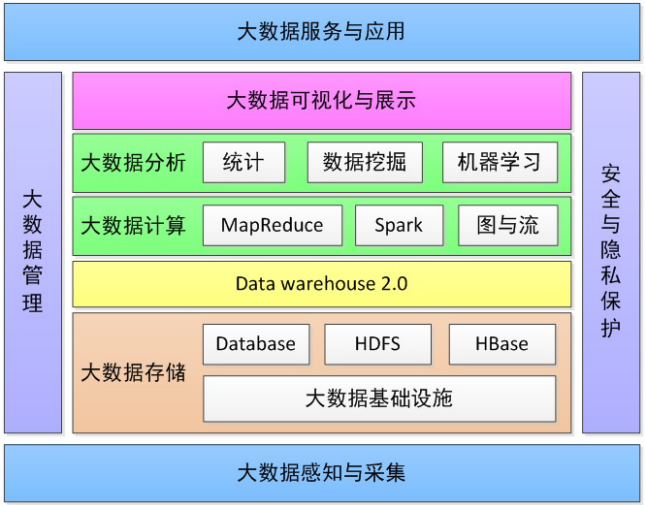
- 云计算服务模式（云服务的商业模式）
  - 商业模式：IaaS，PaaS，SaaS
  - 部署模式：公有云，私有云，社区云，混合云
- 云计算逻辑平台（云平台的逻辑组成）
  - 纵向架构：IaaS，PaaS， SaaS（ technically ）
  - 横向架构：计算，存储，网络（processing, storage, communication）
- 云计算系统实现（云系统的物理实现）
  - 数据中心（Datacenter）

The datacenter *is* the new computer!

Google's datacenter at Dalles, USA

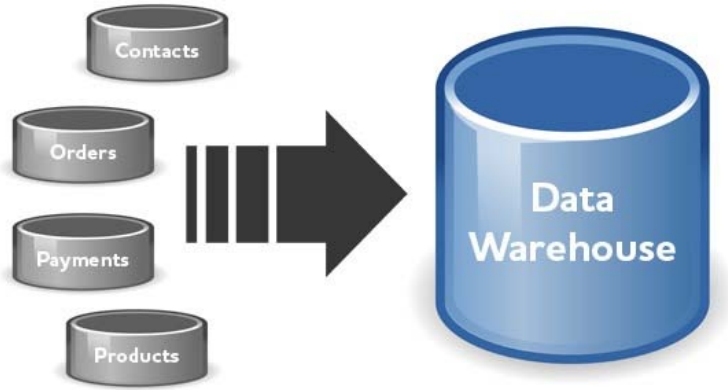
第5章 数据的全生命周期管理

- 数据采集
- 数据存储
- 数据计算
- 数据分析
- 数据展示
- 实训Lab-05：Python数据采集与存储



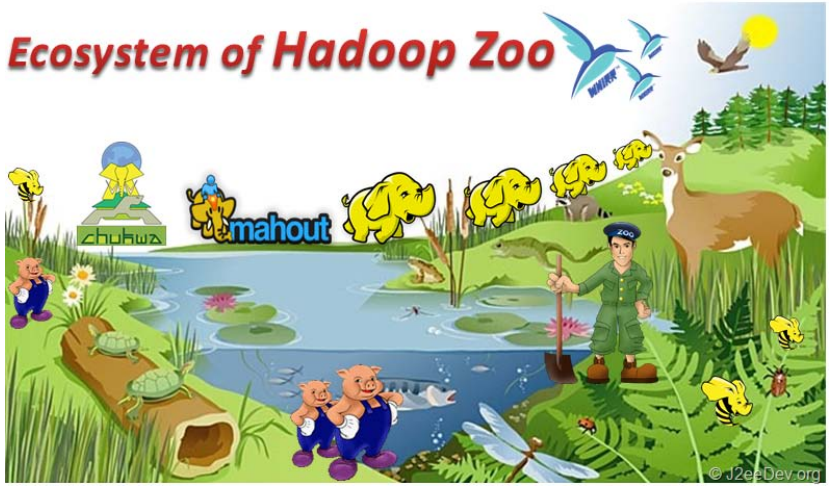
第6章 数据库系统

- 关系数据库与SQL语言
- 数据仓库与OLAP
- NoSQL和NewSQL
- 实训Lab-06：SQL数据处理与分析



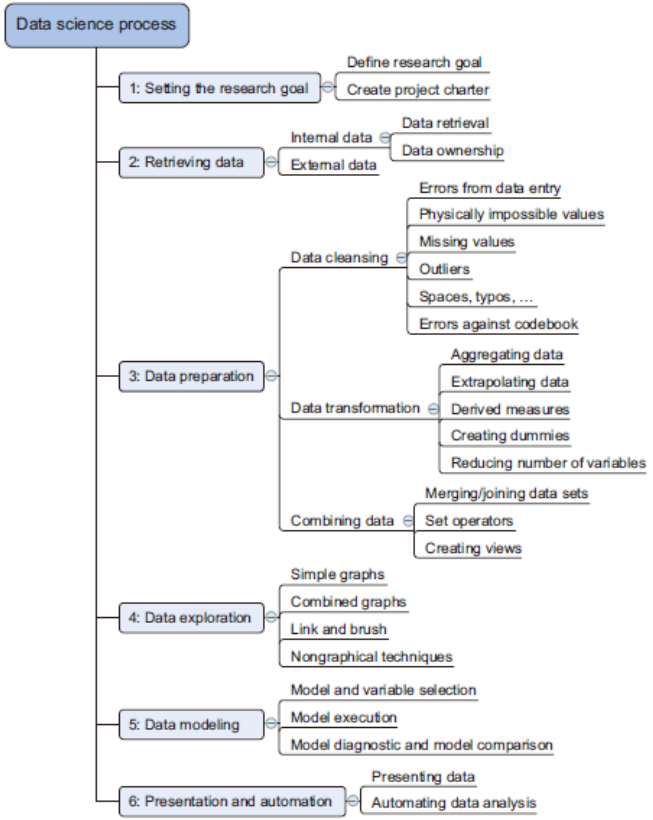
第7章 大数据系统

- Hadoop和Spark生态
- SQL on Hadoop
- 大数据系统实例
- 实训Lab-07：SQL大数据分析与评测
- 实训Lab-07：Mapreduce数据处理



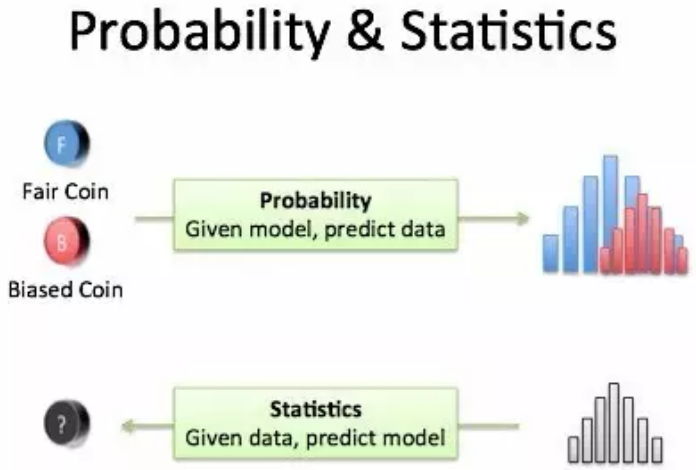
第8章 数据科学过程

- 数据获取
- 数据预处理
- 探索性分析
- 数据建模
- 数据可视化
- Python数据工程生态
- 实训Lab-08：Python数据科学过程



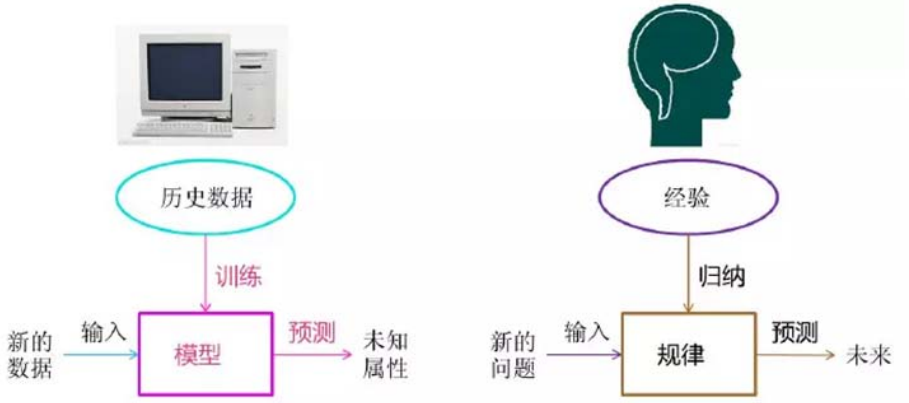
第9章 统计数据分析的原理

- 概率与统计
- 模型和估计
- 假设检验
- 统计可视化
- 实训Lab-09：Python探索性统计分析



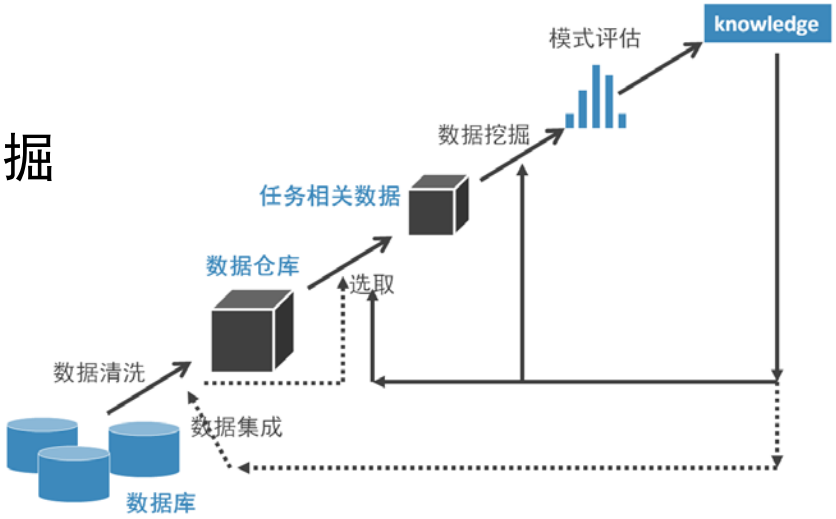
第10章 机器学习方法

- 机器学习简史
- 机器学习的原理与方法
- 深度学习技术
- 实训Lab-10：Python机器学习



第11章 异构数据挖掘

- 多维数据挖掘
- 自然语言分析
- 图像数据处理
- Web信息提取
- 社交关系网络
- 实训Lab-11：推荐系统与社交网络挖掘
- Python vs. Knime



IV 数据应用与数据伦理

- 第12章 应用综合案例分析
  - 智慧城市、人工智能、教育大数据
  - 实训Lab-12：搜索引擎和推荐系综合实践
  - 实训Lab-12：智能运维综合实践

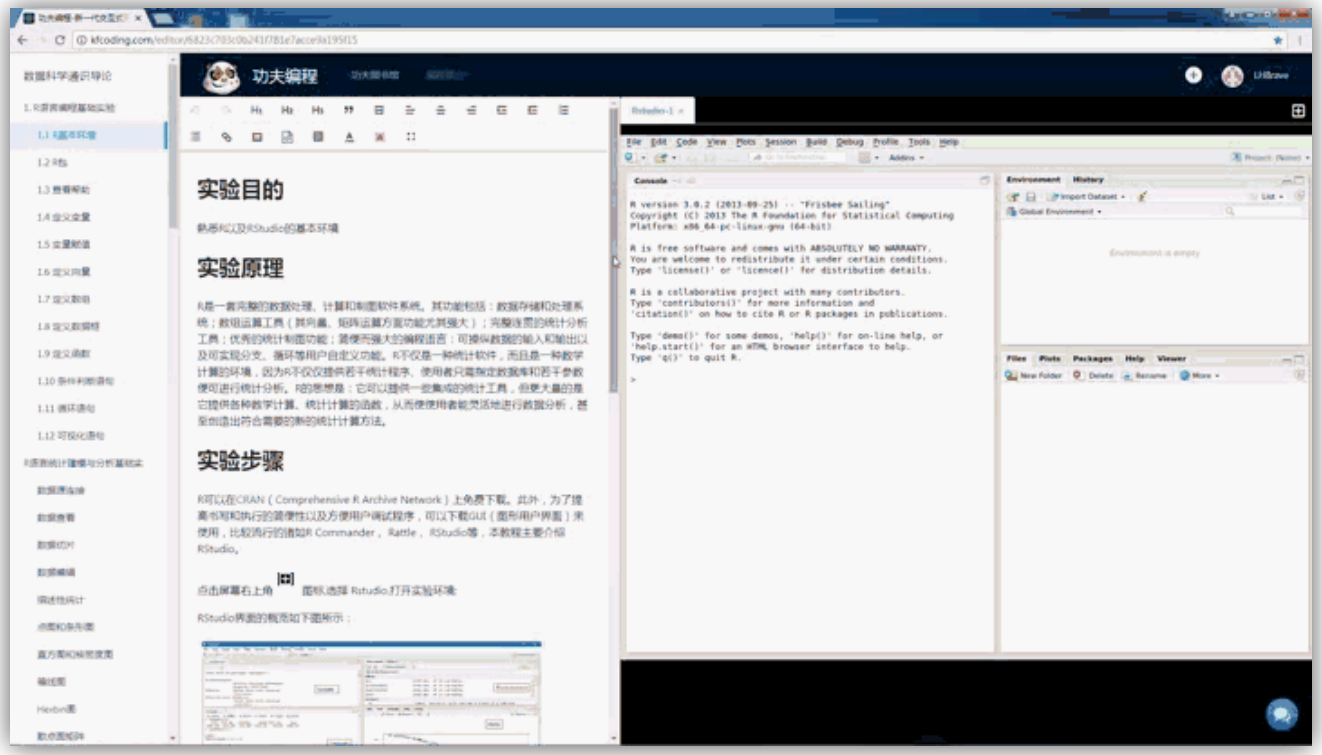
- 第13章 数据道德与职业行为准则

关键2：实训平台的构建

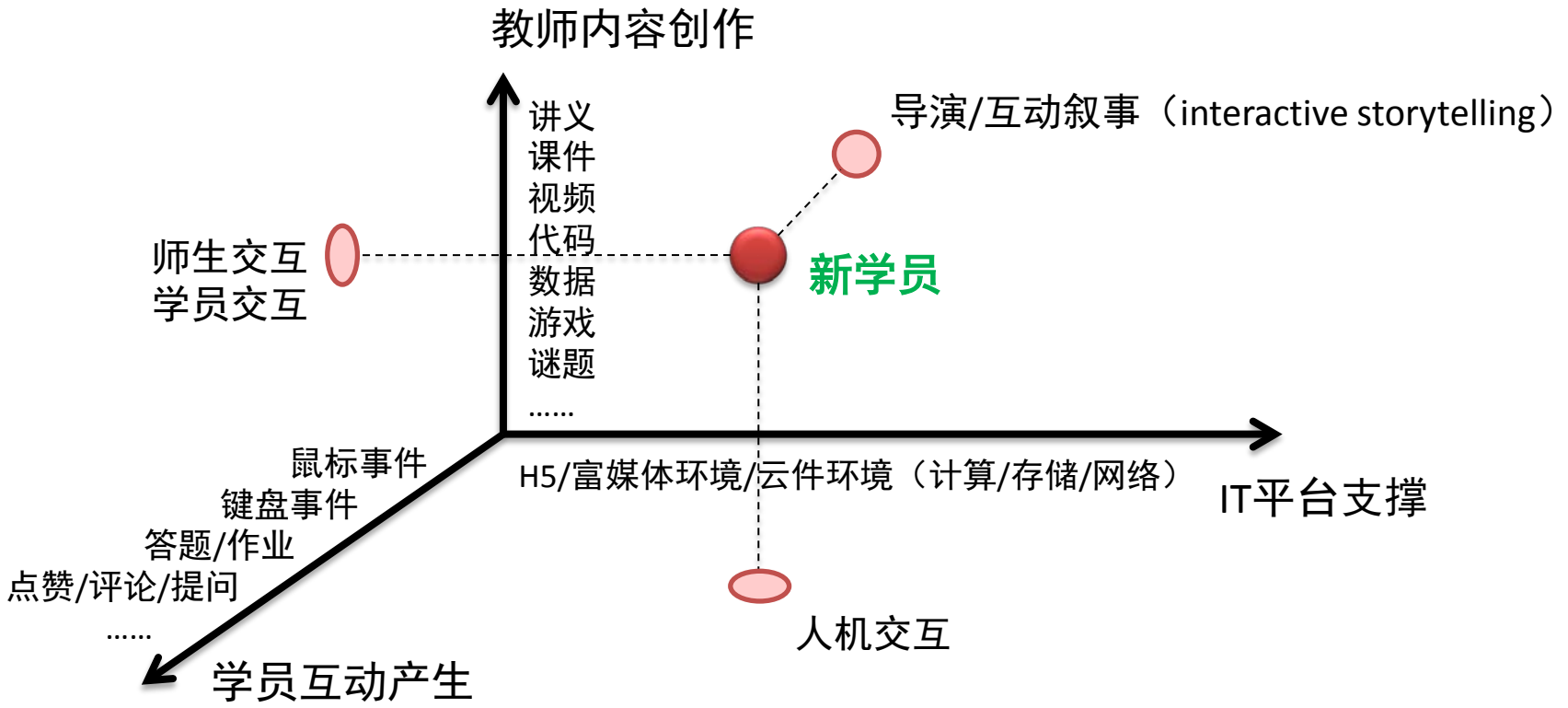
- 对实训平台的诉求：
  - 要是基于云的互联网实训平台
    - 7×24、large scale、全平台访问、……
  - 要能够方便满足各种编程语言和编程环境
    - Python、R、SQL、Julia、Hadoop、Spark、TensorFlow、……
  - 要能够方便的和课程内容与课程资源整合
    - PPT、讲义、视频、实验指导、实验报告、作业、课程设计、……
  - 要能够满足学生的互动需求
    - 老师和学生、学生和学生、学生和机器、……
  - 要能够留存学生的学习行为数据（教育大数据）
    - 在线学习、点评、疑问、学习效果、……

新一代交互式开发者学习社区

新一代交互式开发者学习平台演示

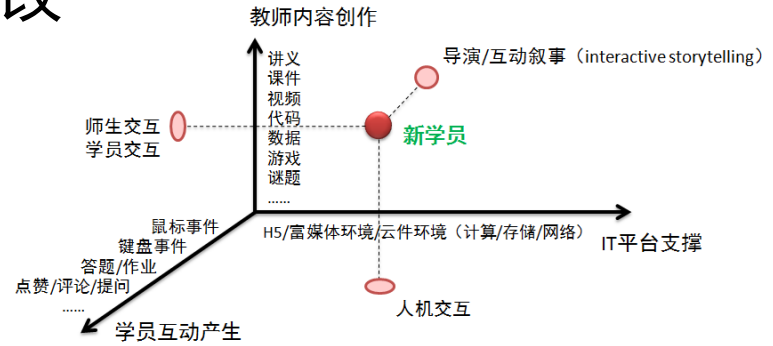


交互式学习资源空间的构建（教师视角）



教师的诉求

- 完整与强大的实训课程制作
- 课堂上能方便演示
- 学生课下方便复现
- 学生课后方便操练
- 作业的布置、提交、与自动化批改
- 能够拓展到数据分析的全栈
- 学生的学习行为分析



开放（Open）的内容创作工具

课程资源资源整合

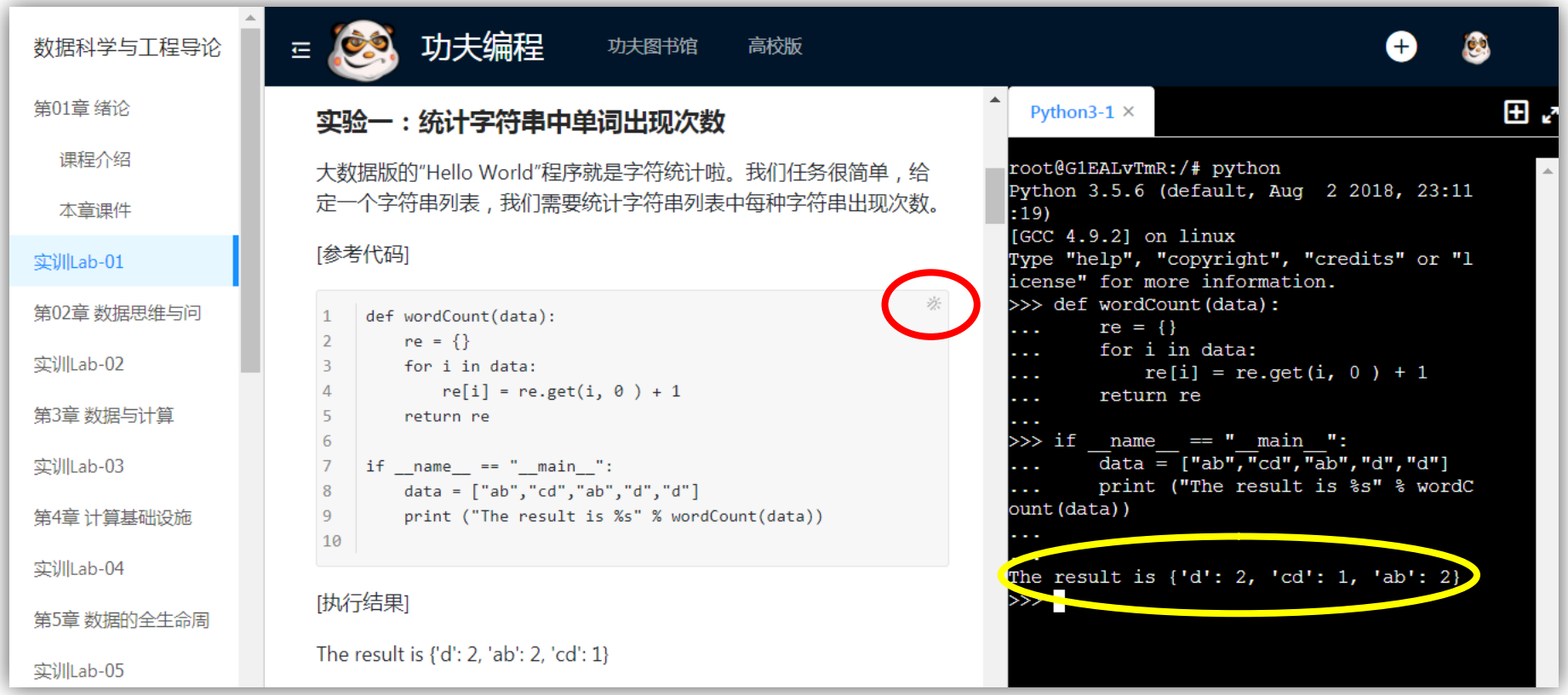

数据科学与工程导论
第01章 绪论
课程介绍
本章课件
实训Lab-01
习题解答
第02章 数据思维与
本章课件
实训Lab-02
习题解答
第3章 数据与计算
本章课件A
本章课件B
实训Lab-03
习题解答
第4章 计算基础设施
本章课件
功夫编程
功夫图书馆
高校版
willtongji
YOUKU
01:54 / 02:10
高清
YOUKU
数据科学与工程导论
WebIDE-1
语言：
python
样式：
monokai
def print_welcome(name):
print("Welcome to", name)
print_welcome("Kfcoding")

实训演练（workbench）

进入Python环境

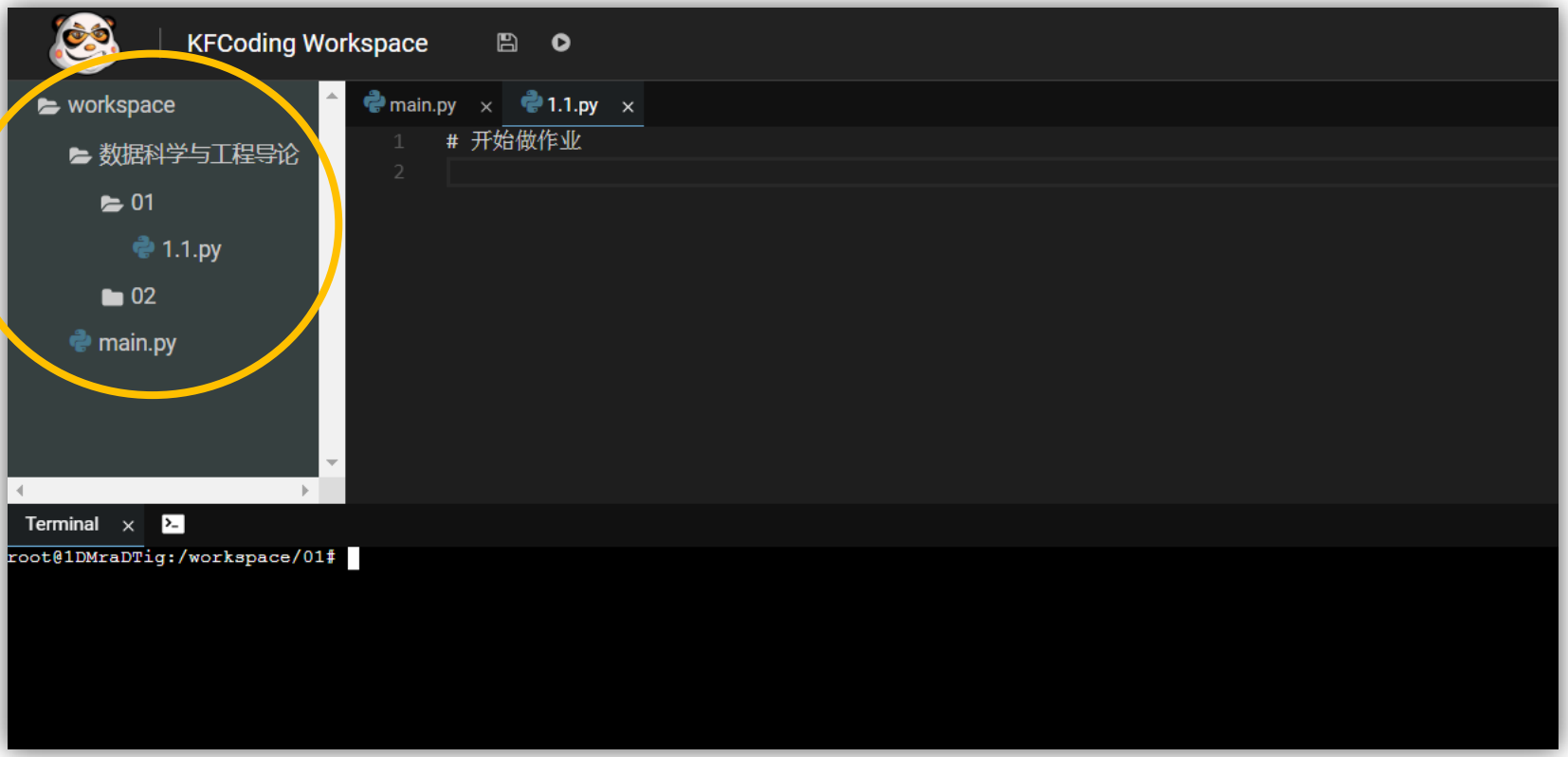

数据科学与工程导论
第01章 绪论
课程介绍
本章课件
实训Lab-01
第02章 数据思维与问
实训Lab-02
第3章 数据与计算
实训Lab-03
第4章 计算基础设施
实训Lab-04
第5章 数据的全生命周
实训Lab-05
功夫编程
功夫图书馆
高校版
实验一：统计字符串中单词出现次数
大数据版的"Hello World"程序就是字符统计啦。我们任务很简单，给定一个字符串列表，我们需要统计字符串列表中每种字符串出现次数。
[参考代码]
def wordCount(data):
re = {}
for i in data:
re[i] = re.get(i, 0) + 1
return re
if __name__ == "__main__":
data = ["ab","cd","ab","d","d"]
print ("The result is %s" % wordCount(data))
[执行结果]
The result is {'cd': 1, 'd': 2, 'ab': 2}
Python3-1
root@hka0BDomR:/# python3
Python 3.5.6 (default, Aug 2 2018, 2
3:11:19)
[GCC 4.9.2] on linux
Type "help", "copyright", "credits" o
r "license" for more information.
>>>

代码运行

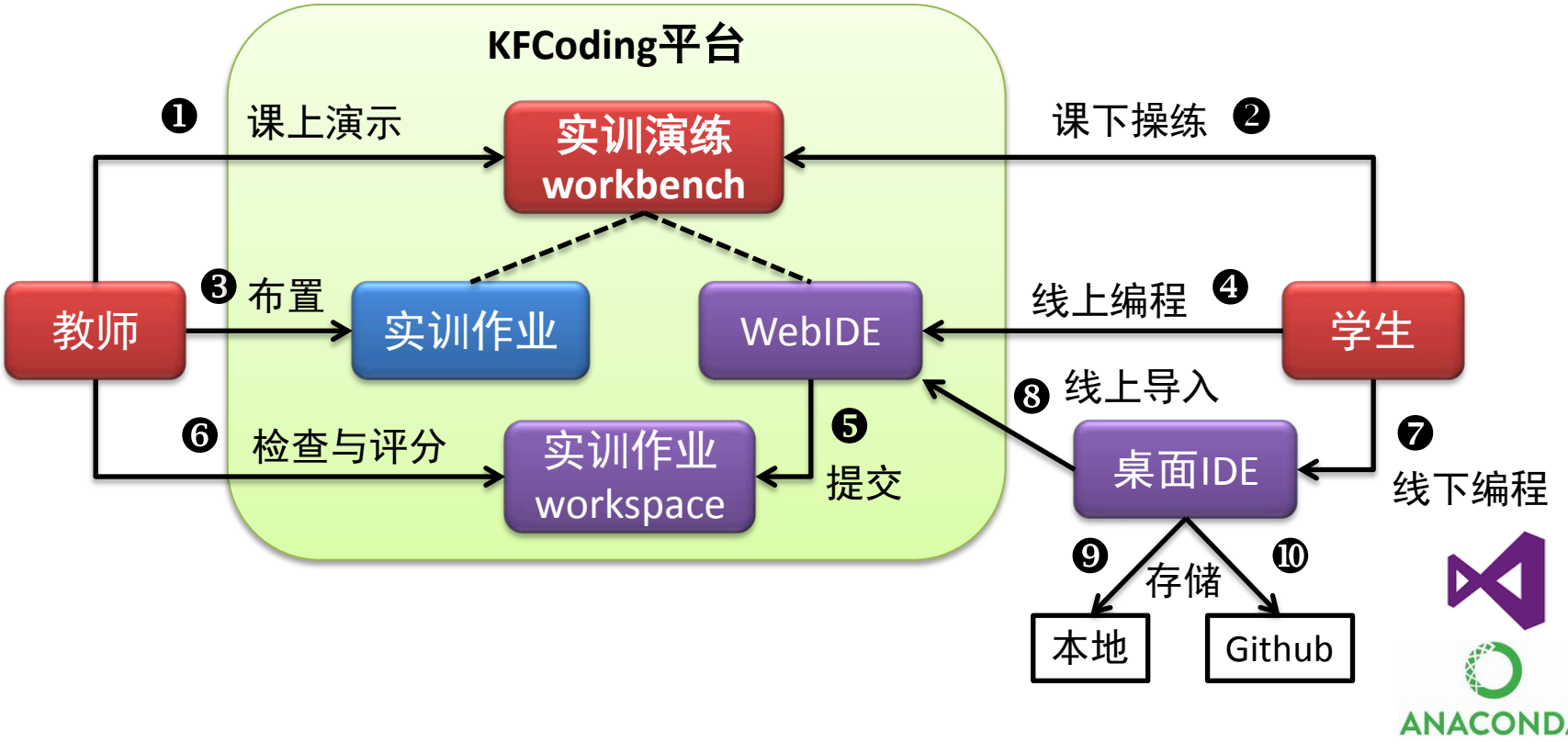

数据科学与工程导论
第01章 绪论
课程介绍
本章课件
实训Lab-01
第02章 数据思维与问
实训Lab-02
第3章 数据与计算
实训Lab-03
第4章 计算基础设施
实训Lab-04
第5章 数据的全生命周
实训Lab-05
功夫编程
功夫图书馆
高校版
实验一：统计字符串中单词出现次数
大数据版的"Hello World"程序就是字符统计啦。我们任务很简单，给定一个字符串列表，我们需要统计字符串列表中每种字符串出现次数。
[参考代码]
1 def wordCount(data):
2 re = {}
3 for i in data:
4 re[i] = re.get(i, 0) + 1
5 return re
6
7 if __name__ == "__main__":
8 data = ["ab","cd","ab","d","d"]
9 print ("The result is %s" % wordCount(data))
10
[执行结果]
The result is {'d': 2, 'ab': 2, 'cd': 1}
Python3-1
root@G1EALvTmR:/# python
Python 3.5.6 (default, Aug 2 2018, 23:11
:19)
[GCC 4.9.2] on linux
Type "help", "copyright", "credits" or "l
icense" for more information.
>>> def wordCount(data):
... re = {}
... for i in data:
... re[i] = re.get(i, 0) + 1
... return re
...
>>> if __name__ == "__main__":
... data = ["ab","cd","ab","d","d"]
... print ("The result is %s" % wordC
ount(data))
...
The result is {'d': 2, 'cd': 1, 'ab': 2}
>>>

实训作业（workspace）

用Git组织你的作业


KFCoding Workspace
workspace
数据科学与工程导论
01
1.1.py
02
main.py
main.py
1.1.py
1 # 开始做作业
2
Terminal
root@1DMraDTig:/workspace/01#

课程实训与作业流程

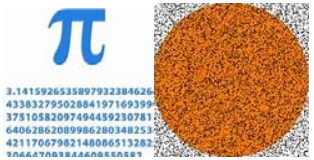

KFCoding平台
❶ 课上演示
实训演练
workbench
课下操练 ❷
教师
❸ 布置
实训作业
WebIDE
线上编程 ❹
学生
❻ 检查与评分
实训作业
workspace
❺ 提交
❽ 线上导入
桌面IDE
❼ 线下编程
❾
存储
❿
本地
Github
ANACONDA

关键3：教学方法

- 线下教学
  - 基本概念、思维方式、原理技术、应用案例
- 线上微信公众号互动平台
  - 师生互动、课件同步、重点解释、补充阅读
- 课内外实践Lab
  - 动手实践
- 综合创新

课程公众号

计算思维与数据思维



Machine Learning

（数据为中心的）问题求解

数据思维

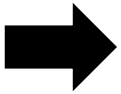
计算思维

问题1：求解√2的不同方法

方法1：循环迭代，逐步逼近

```python
#开平方1
def Square_root_1():
    c = 2
    i = 0
    g = 0
    for j in range(0, c+1):
        if (j * j > c and g == 0):
            g = j -1
    while(abs(g * g - c) > 0.0001):
        g += 0.00001
        i = i+1
        print ("%d:g = %.5f" % (i,g))


Square_root_1()
```



```
41411:g = 1.41411
41412:g = 1.41412
41413:g = 1.41413
41414:g = 1.41414
41415:g = 1.41415
41416:g = 1.41416
41417:g = 1.41417
41418:g = 1.41418
>>>
```

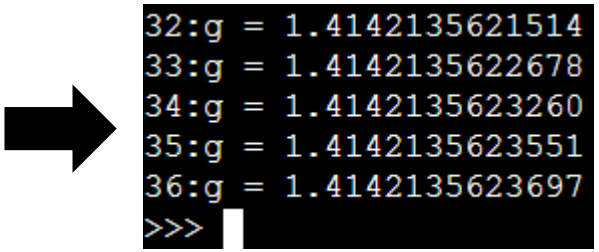
循环次数迭代了414818次，
精度达到0.00001

问题1：求解√2的不同方法

方法2：二分查找法

```
# 开平方2  二分法
def Square_root_2():
    i = 0
    c = 2
    m_max = c
    m_min = 0
    g = (m_min + m_max)/2
    while (abs(g * g - c)> 0.00000000001):
        if (g*g<c):
            m_min = g
        else :
            m_max = g
        g = (m_min + m_max)/2
        i = i + 1
        print ("%d:g = %.13f" % (i,g))

Square_root_2()
```

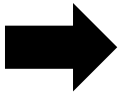


循环次数迭代了36次，精度达到0.00000000001

问题1：求解√2的不同方法

方法3：牛顿法

```
1   # 开平方3  牛顿法
2   def Square_root_3():
3       c = 2
4       g = c/2
5       i = 0
6       while (abs(g*g-c)>0.00000000001):
7           g = (g + c/g)/2
8           i = i+1
9           print ("%d:%.13f"%(i,g))
10
11  Square_root_3()
```



```
>>> Square_root_3()
1:1.5000000000000
2:1.4166666666667
3:1.4142156862745
4:1.4142135623747
>>>
```

循环次数仅4次，精度达到0.00000000001

Remark 1

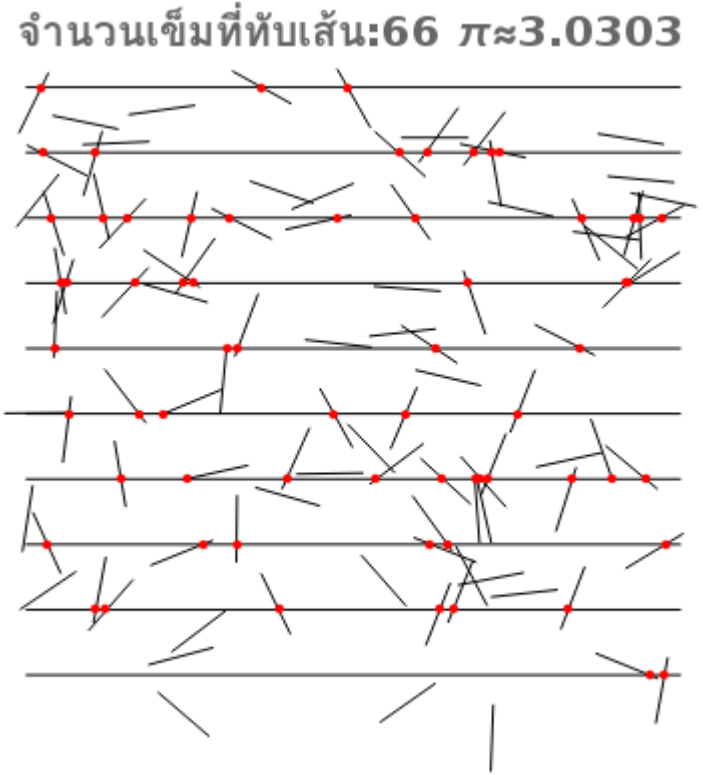
- 上面实例的精妙有趣之处，就是它对于“算法”的研究，解决同一个问题可以设计出各种不同的算法，不是获得解就结束了，而是要进一步分析不同算法之间对程序执行效率的影响，然后选择最好的算法。
- “设计”就是算法研究中的最重要的问题，针对一个问题，设计出高效的算法，而不单单是解决给定的一个问题。
- 这就是计算之美！

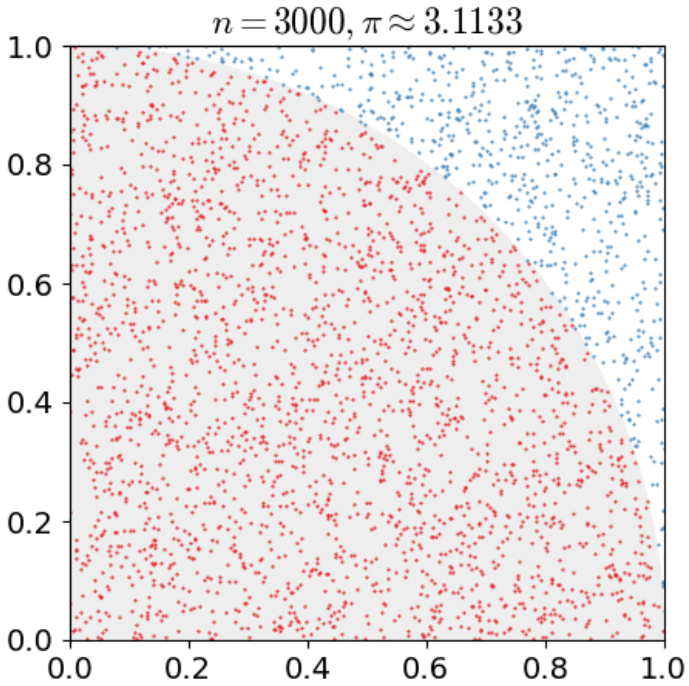
问题2：求解Pi的值

- 常规的解析方法：

$$\frac{\pi}{2} = \frac{2}{1} \cdot \frac{2}{3} \cdot \frac{4}{3} \cdot \frac{4}{5} \cdot \frac{6}{5} \cdot \frac{6}{7} \cdot \frac{8}{7} \cdot \frac{8}{9} \cdots \qquad \frac{\pi^2}{6} = \frac{1}{1^2} + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \cdots$$

Buffon’s needle与蒙特卡洛法



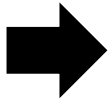


使用蒙特卡洛法求Pi的值

```python

# 蒙特卡洛法求Pi
import random
def Pi(times):
    sum = 0
    for i in range(times):
        x = random.random()
        y = random.random()
        d2 = x*x + y*y
        if d2 <= 1 :
            sum+=1
    return (sum/times*4)


times = 100000000
x = Pi(times)
print ("Pi = %.8f"%(x))
```

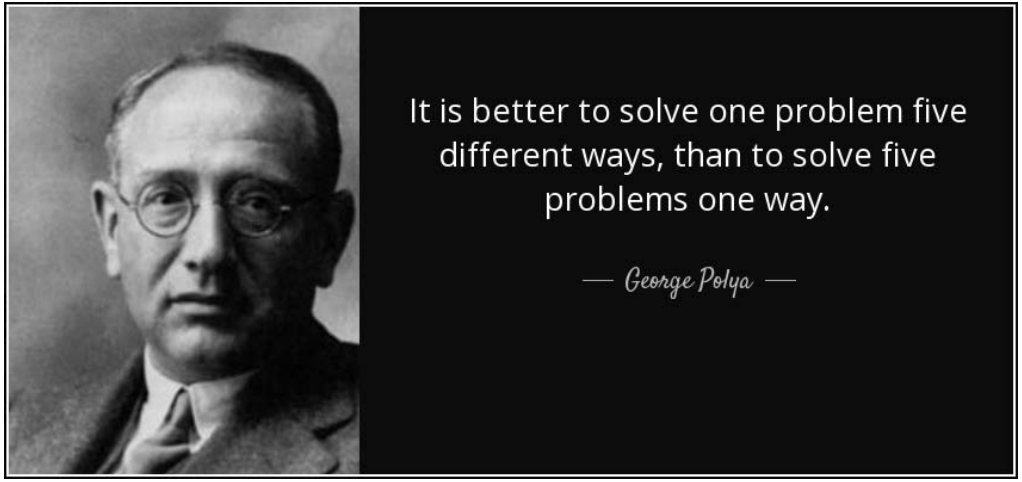


```
>>> # 蒙特卡洛法求Pi
... import random
>>> def Pi(times):
...     sum = 0
...     for i in range(times):
...         x = random.random()
...         y = random.random()
...         d2 = x*x + y*y
...         if d2 <= 1 :
...             sum+=1
...     return (sum/times*4)
...
>>>
>>> times = 100000000
>>> x = Pi(times)
>>> print ("Pi = %.8f"%(x))
Pi = 3.14172316
```

大约40秒以后

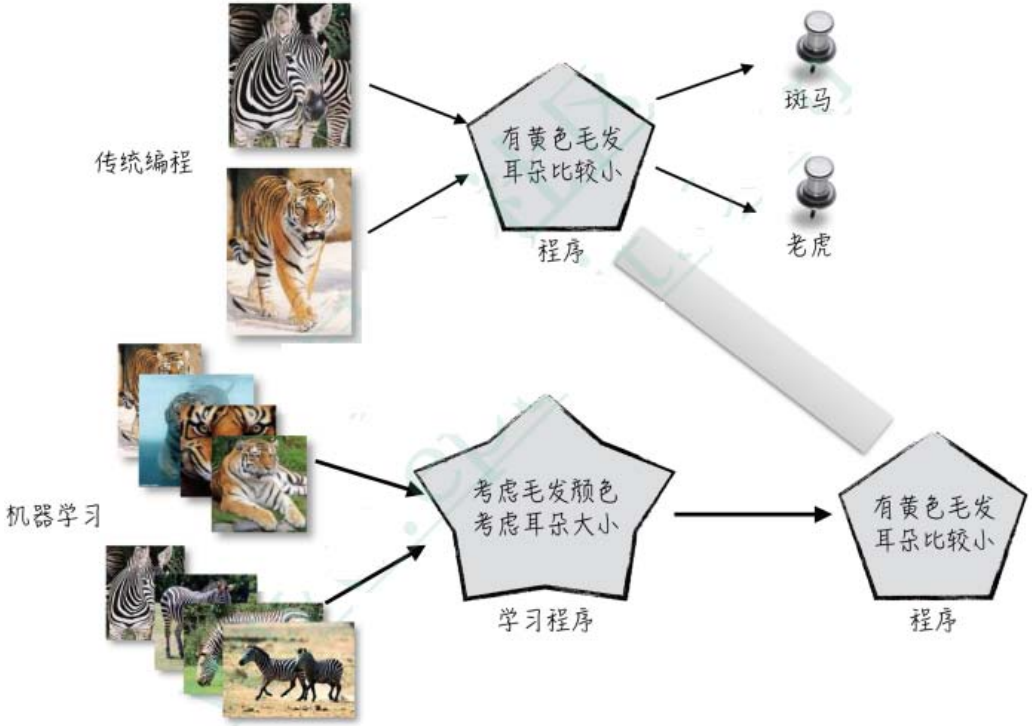
习 题

1. 根据蒙特卡洛法的思想，设计求解根号2的第四种方法。
2. 至少用3种方法求解Pi的值，并比较它们的效率（精度保留到小数点后10位）。

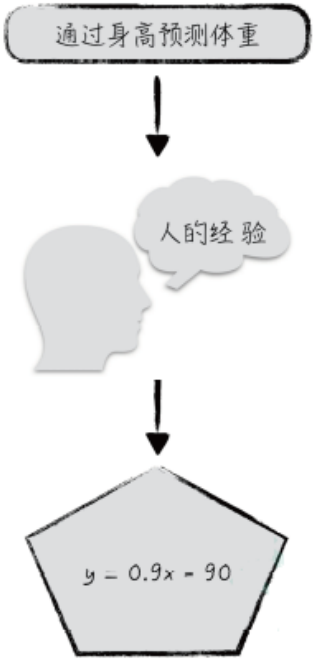


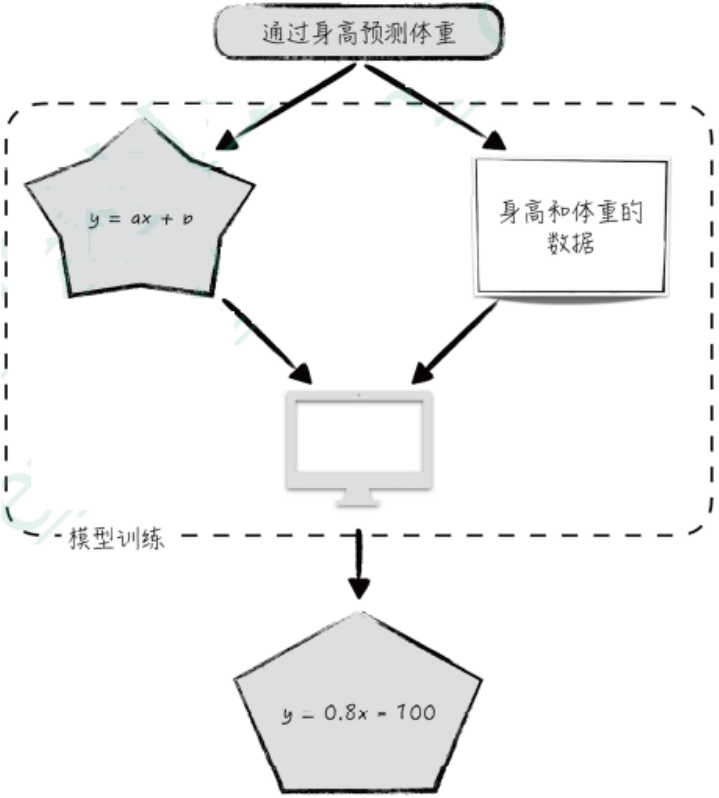
机器学习方法

从编程的角度来看，机器学习是一种能自动生成程序的特殊程序。



通过身高（x）来预测体重（y）

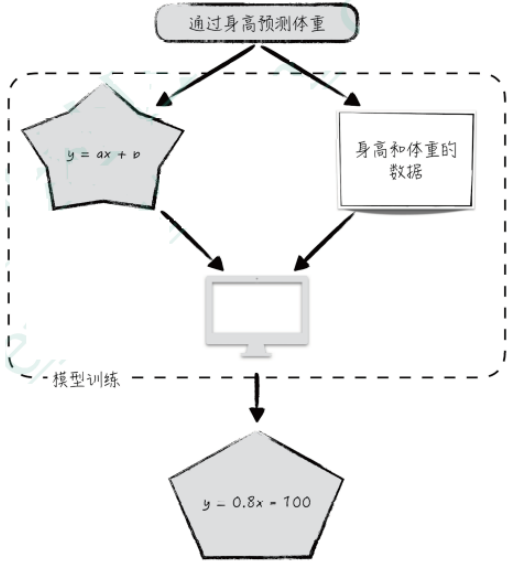




机器学习：数据驱动的问题求解

经验驱动的算法模型

数据驱动的算法模型



经验 → 算法模型 ← 数据

Remark 2

- 上面实例的精妙有趣之处，就是在于以数据为中心或数据驱动的问题求解过程，传统算法无法或很难解决的问题，通过引入相关数据，往往就变得迎刃而解了。
- 这类问题除了需要考虑高效的算法之外，还要生产或采集到与之相关的关键数据，才能保证问题的高质量求解；
- 这就是计算与数据之美！

什么是数据模型？

- A **data model** is an abstract model that organizes elements of data and standardizes how they relate to one another and to properties of the real world entities.

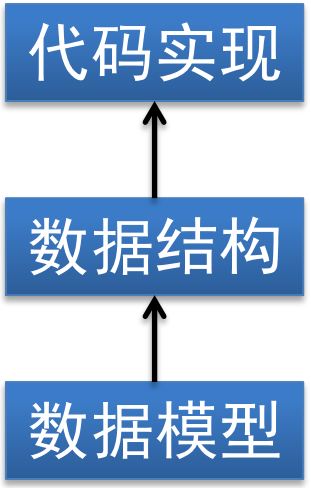
- 程序语言中的数据模型
  - 数据类型、数据结构
- 系统软件中的数据模型
  - 文件、目录、进程
- 集成电路中的数据模型
  - 命题逻辑

程序语言中的数据模型

- 基本模型
  - 数值、字符、向量、文本
- 高级模型
  - 树、表、集合、关系、图、有限自动机、正则表达式
- 数据模型的实现举例：
  - 树（Tree）：指针数组
  - 列表（List）：数组、链表；
  - 集合（Set）：链表、特征向量、散列表
  - 关系（Relation）：数据库、模式、健、索引
  - 图（Graph）：邻接表、邻接矩阵

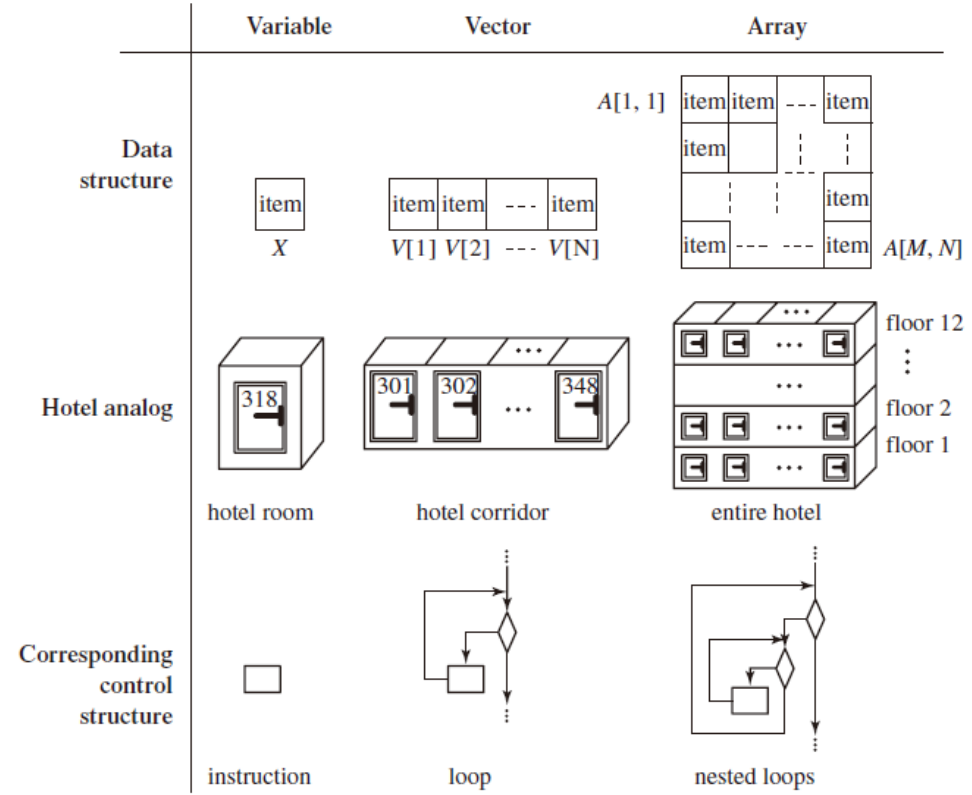
数据模型和数据结构的关系

- 数据模型是数学抽象
- 数据结构是程序表达
- 例如：
  - 列表list（数据模型）
  - 链表linked list（数据结构）



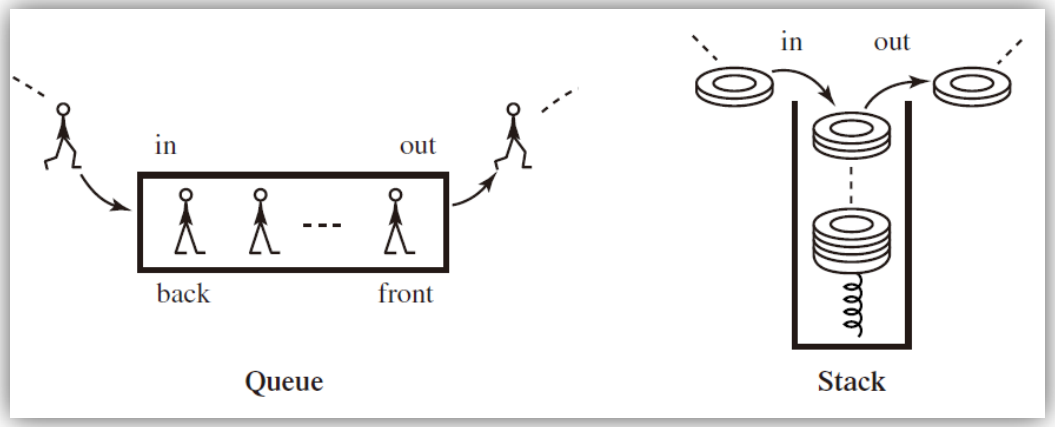
数据结构与算法结构的关系

例如：数组（向量）和循环的关系



数据结构与算法结构的关系

例如：队列与栈




```
Exercise (N)
{ Print N
  if (N < 3) Exercise (N + 1)
  print N
}

Exercise (1)
{ Print 1
  if (1 < 3) Exercise (1+1)
  print 1
}

Exercise (2)
{ Print 2
  if (2<3) Exercise (2+1)
  print 2
}

Exercise (3)
{ Print 3
  if (3 < 3) false
  print 3
}
```

数据结构与算法实例：用树排序

- 第1步：将数组表示为“二分搜索树”

128	76	106	402	100	46	354	1018	112	28	396	35

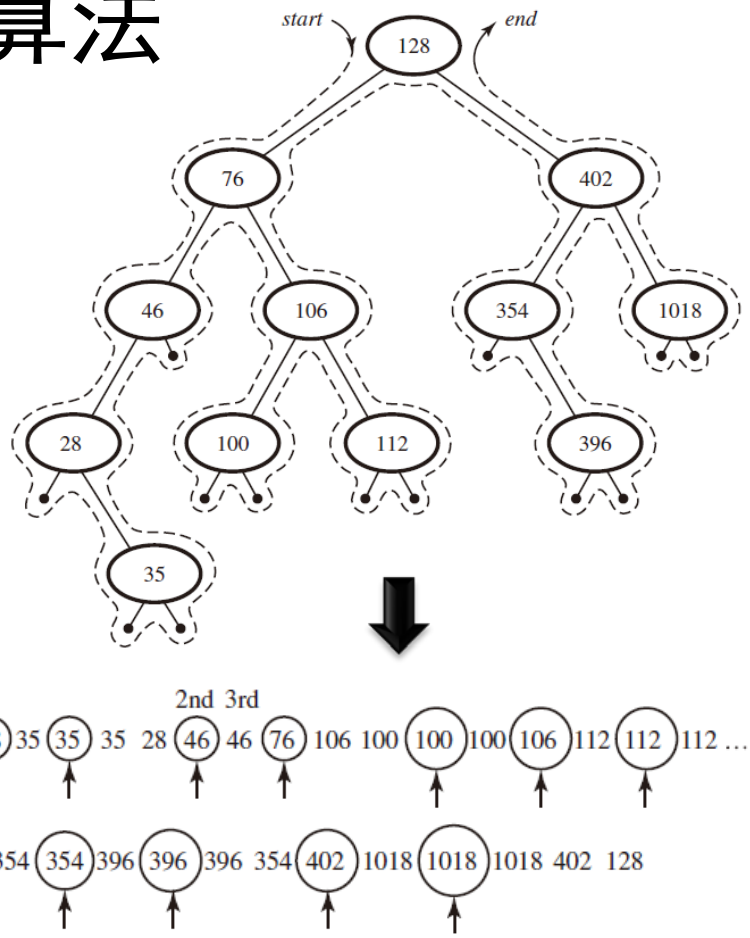
first *last*

你能看出这个树是如何生成的吗?

不同数据结构上排序算法

- 第2步：以“某种”方式遍历树
- 规则如下：
  - left-first traversal
  - second-visit output

为什么输出肯定是从小到大的?



比特世界

计算空间

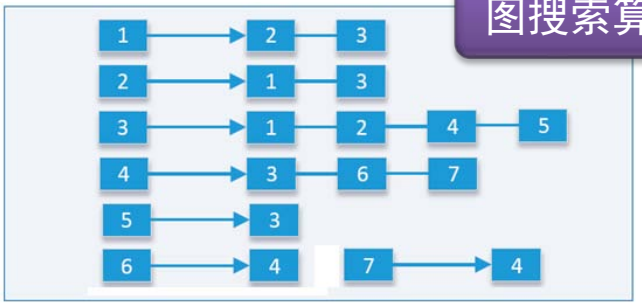
问题空间

现实世界

程序设计与代码实现

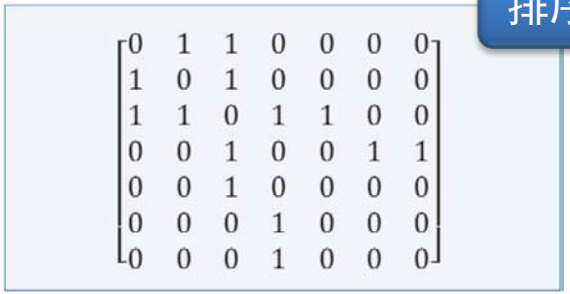
程序设计与代码实现

图搜索算法

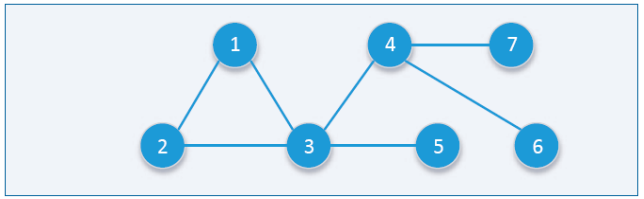


图数的邻接表数据结构

排序算法



图数的邻接矩阵数据结构



无向图数据模型

Web
链接网页

结论与展望

- 应该为所有的本科生开设基本的数据科学课程；
- 应该吸引不同专业背景的学生共同来学习数据科学这门课程；
- 应该不断的改进课程体系和课程内容，随着数据学科的发展；
- 应该给予学生足够的语言、平台、数据等工具，促进其数据科学实践；
- 应该在一个学术界、产业界、开源社区联合的空间下进行数据实践；
- 应该关心code of ethics和data of ethics的问题。

Thanks

课程公众号