

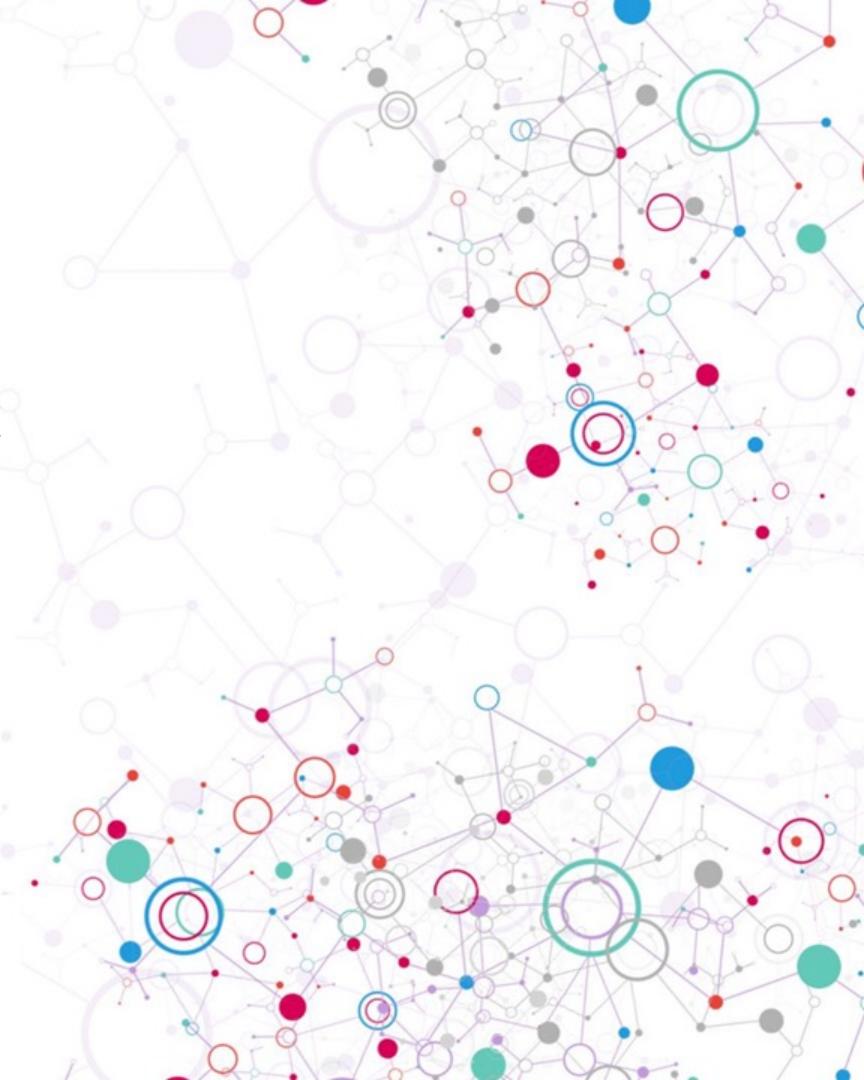
数据思维与实践

王伟

华东师范大学

数据科学与工程学院

全民数字素养与技能培训基地



开篇实例：无处不在的推荐系统

■ 推荐系统成为连接数据特征与用户需求的桥梁

- 普通用户难以直接从大数据中获取所需信息
- 推荐系统将大数据从单纯的数据层面转化到用户可以理解的信息层面，满足客户的需求



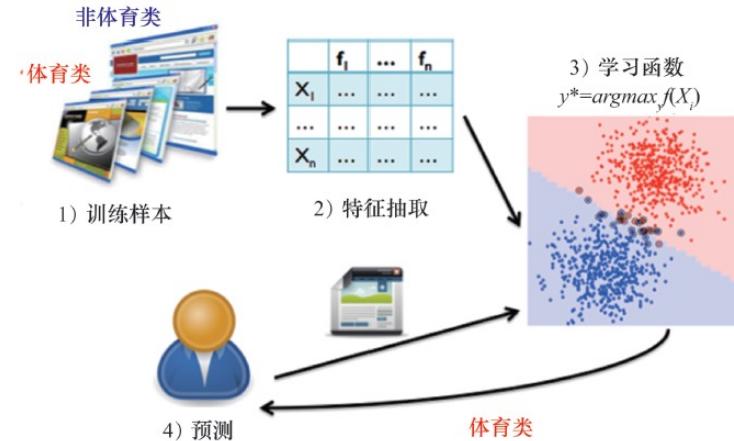
■ 推荐系统为企业带来**巨大价值**

- 2016年“双十一”淘宝交易额破千万
- 今日头条app人均日使用时间超40分钟
- UC浏览器个性化推荐月活跃用户超过3.3亿



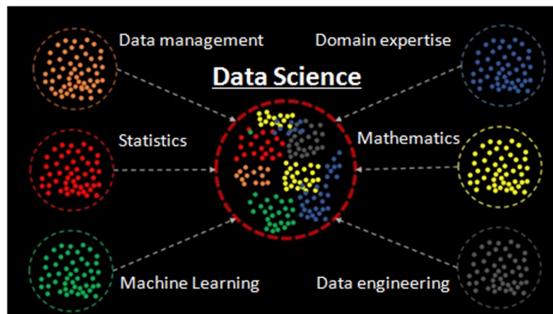
我们正在离开信息的时代，进入推荐的时代。

——Chris Anderson in *The Long Tail*



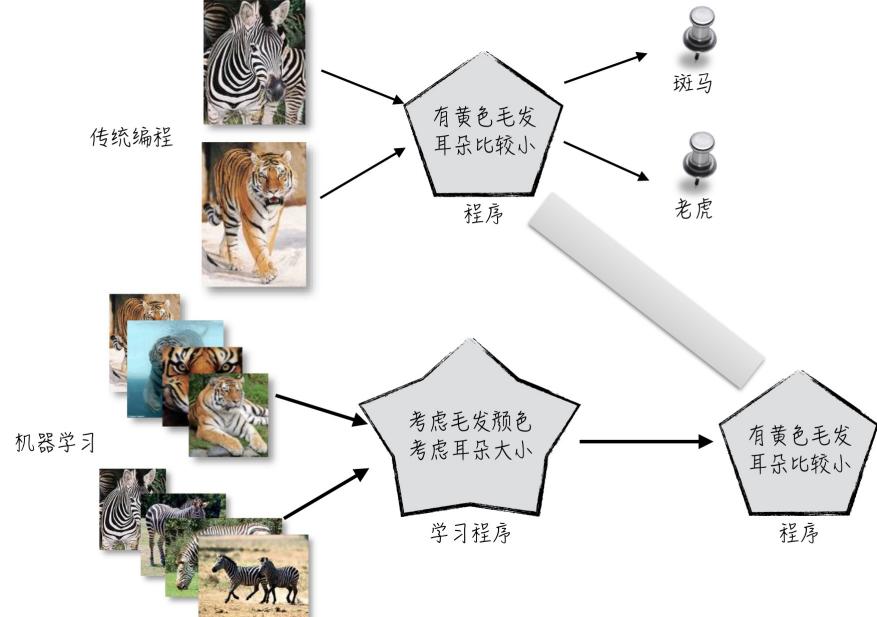
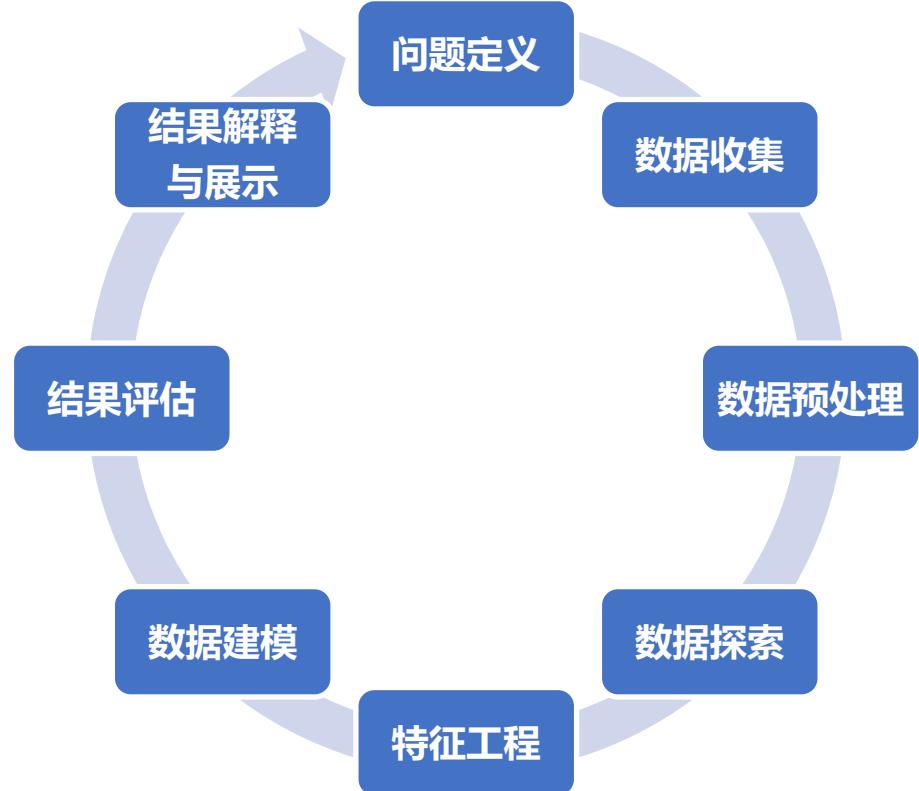
数据思维与实践

第05讲 数据建模与分析



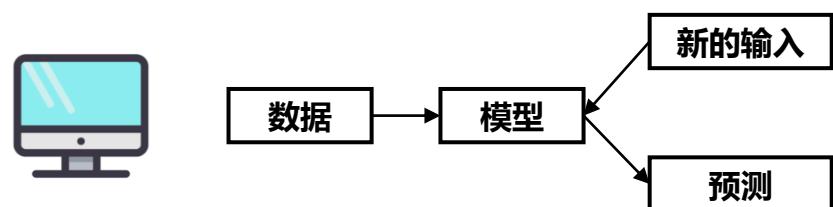
- 什么是数据建模?
- 机器学习基础
- Python 机器学习
- 开源数字王国中的建模与分析

回顾：数据科学过程



数据建模

- **数据**是人类经验的数字化形式。
- **数据建模**: 捕捉数据的本质特征，根据数据的特征形成模型。
 - **按任务属性**: 分类模型、聚类模型、推荐模型、
 - **按数据属性**: 图像模型、语音模型、文字模型、
- **数据思维**是一种通过数据驱动决策的思维模式，包括：
 - 数据、模型、算力和业务模式

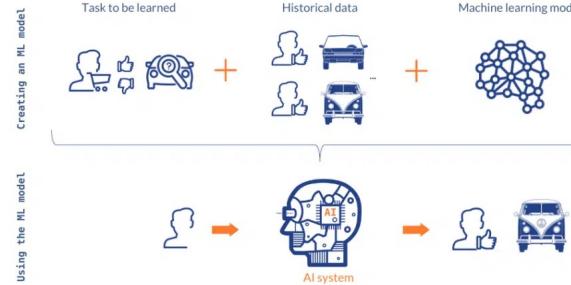
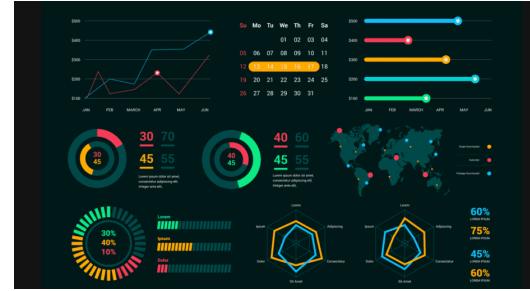


数据分析（建模）的方法

统计方法
可视化方法
机器学习方法



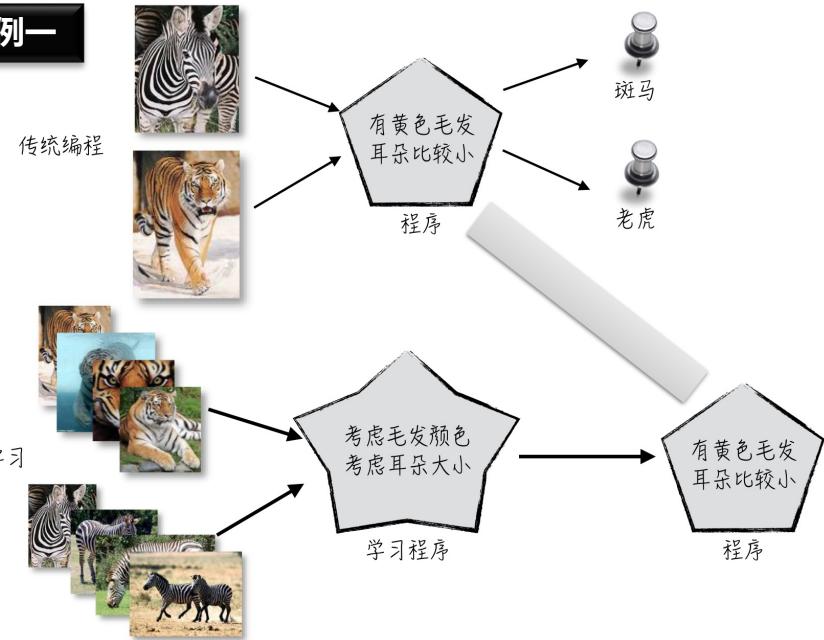
A statistical model



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**。

实例一



传统编程

通过身高预测体重



$$y = 0.9x - 90$$

机器学习

通过身高预测体重



身高和体重的数据

$$y = 0.8x - 100$$

实例二

传统编程

通过身高预测体重



$$y = 0.9x - 90$$

机器学习

通过身高预测体重



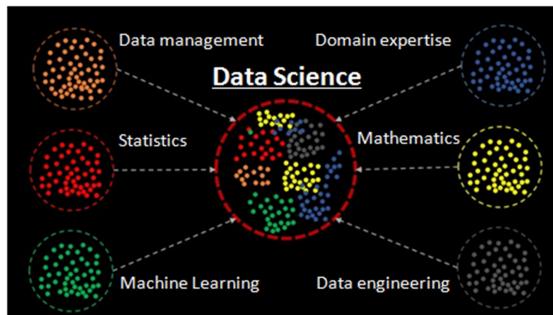
身高和体重的数据

$$y = 0.8x - 100$$

模型训练

数据思维与实践

第05讲 数据建模与分析



- 什么是数据建模?
- **机器学习基础**
- Python 机器学习
- 开源数字王国中的建模与分析

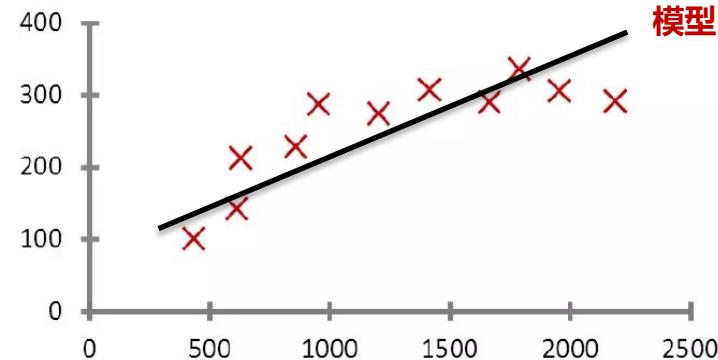
什么是机器学习？

- **学习**是人类具有的一种重要智能行为，但究竟什么是学习，长期以来却众说纷纭。
 - 社会学家、逻辑学家和心理学家都各有其不同的看法。
 - 至今，还没有统一的“机器学习”定义，而且也很难给出一个公认的和准确的定义。
- **机器学习**是研究如何使用机器来模拟人类学习活动的一门学科。稍为严格的提法是：**机器学习是一门研究机器获取新知识和新技能，并识别现有知识的学问。**

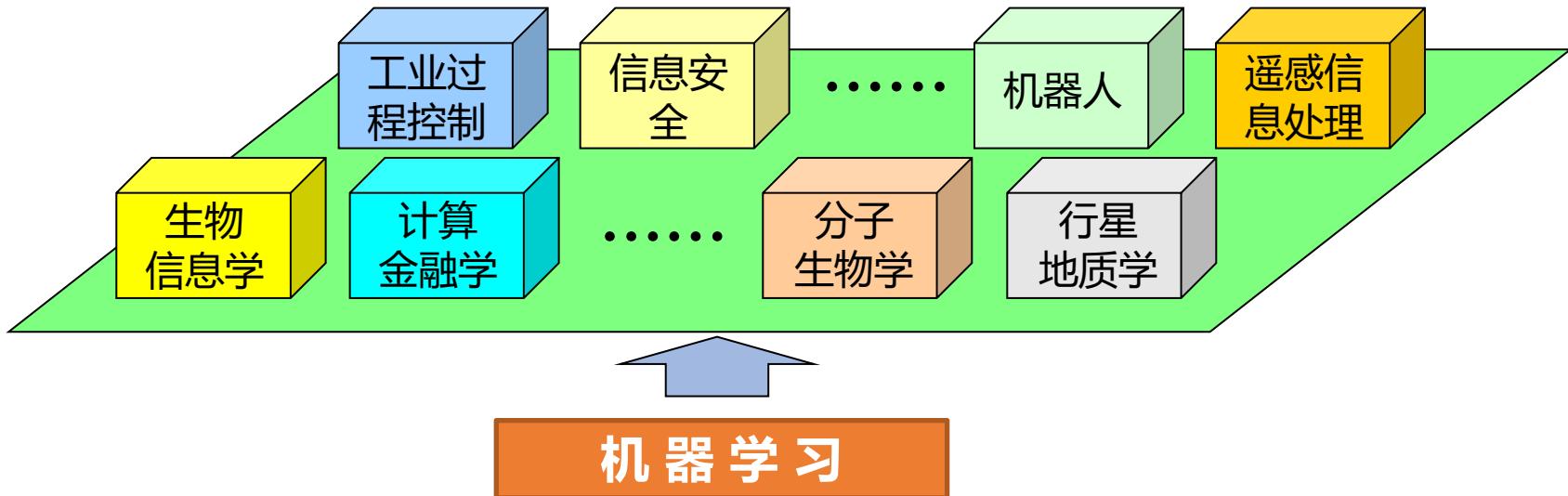
什么是机器学习？

- 从广义上来说，**机器学习**是一种能够赋予机器学习的能力以此让它完成直接编程无法完成的功能的方法。
- 但从实践的意义上来说，机器学习是一种通过**利用数据，训练出模型**，然后**使用模型预测**的一种方法。

问题：现在我手里有一栋房子需要售卖，我应该给它标上多大的价格？房子的面积是 100 平方米，价格是 100 万，120 万，还是 140 万？



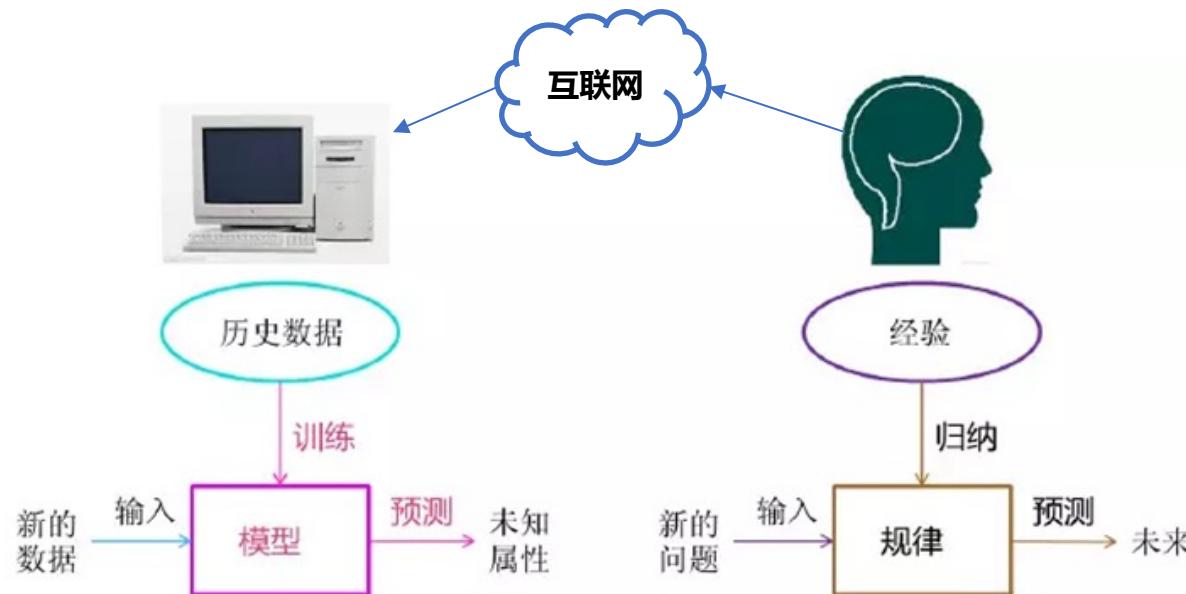
机器学习的重要性



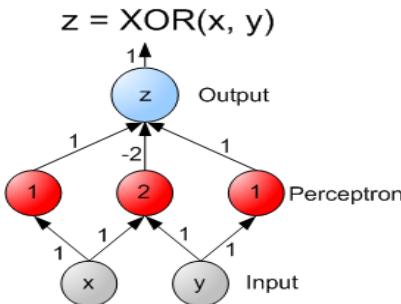
美国 JPL 实验室的科学家在《Science》(2001年9月) 上撰文指出：

- 机器学习对科学的研究的整个过程正起到越来越大的支持作用，.....，该领域在今后的若干年内将取得稳定而快速的发展，.....

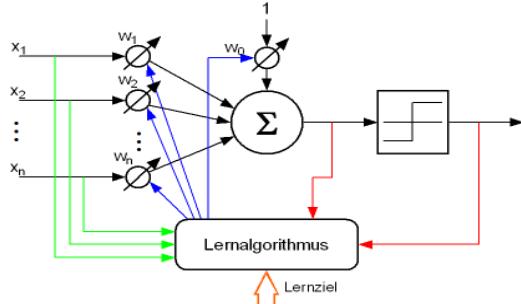
机器学习与人类思考的类比



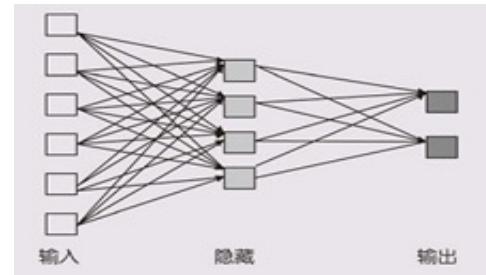
机器学习的发展史



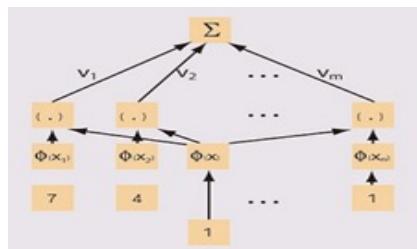
感知机



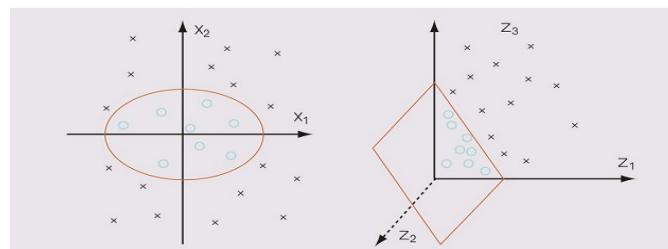
线性适应元



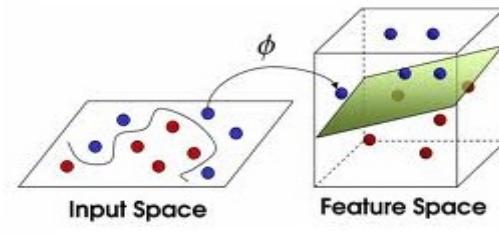
连接主义学习模型



统计学习模型



“核方法” 机器学习



支持向量机

开启人工智能的新时代

谷歌DeepMind挑战赛



vs.  AlphaGo

100万美金挑战赛（5局3胜制）
首尔，3月9日至15日

Lee Sedol
李世石

Google DeepMind Challenge Match

游戏是最佳测试平台



Google DeepMind Challenge Match



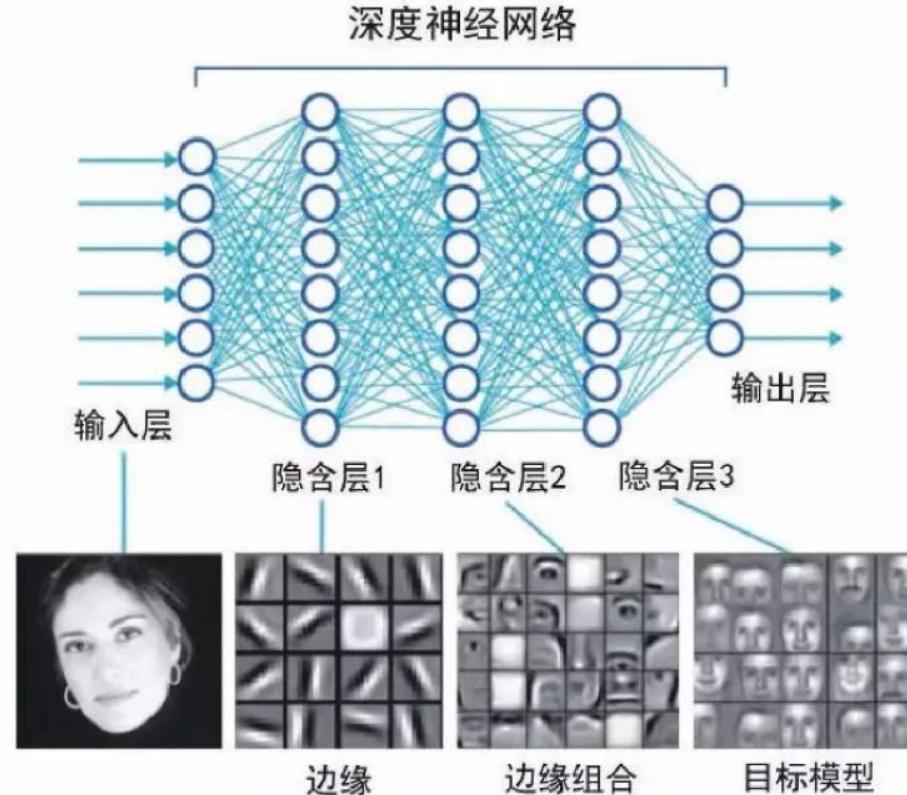
谷歌DeepMind团队在2016年1月《Nature》上发表论文称，他们研发的人工智能算法击败了欧洲围棋冠军Fan Hui，同时也击败了目前最好的围棋程序中99.8%的对手。

深蓝
人工输入的象棋知识
全局搜索
每秒2亿次局面

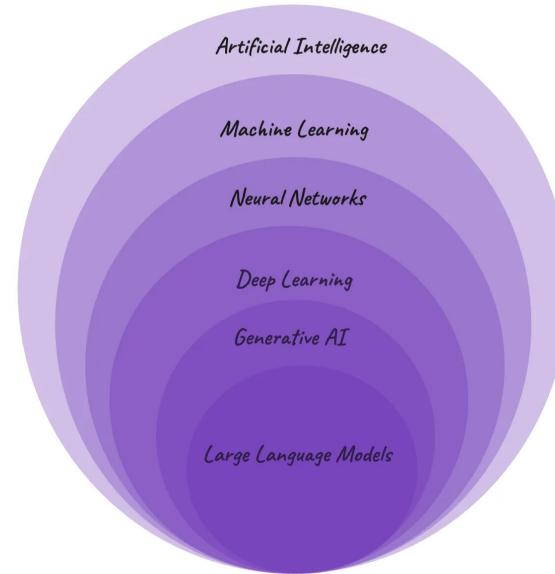
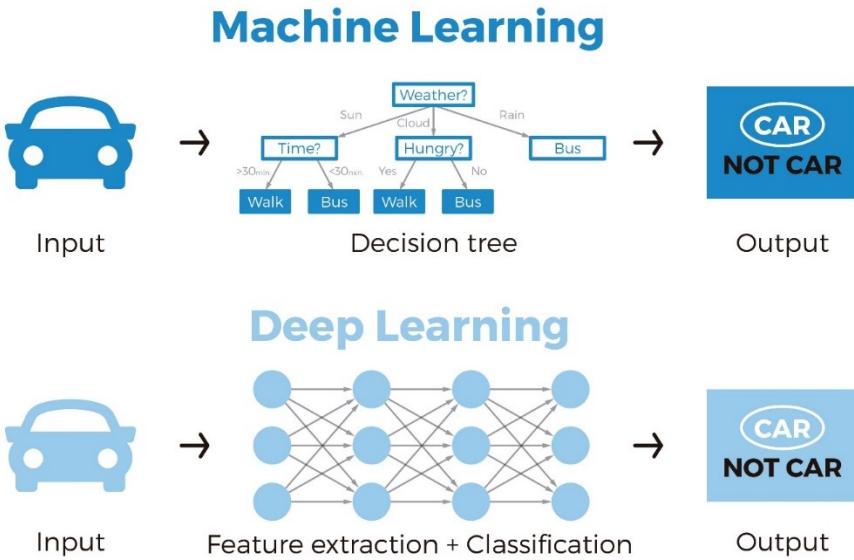
AlphaGo
从专家对弈和自我对弈中学到的知识
由策略和价值网络引导的高度选择性搜索
每秒10万次局面

Google DeepMind Challenge Match

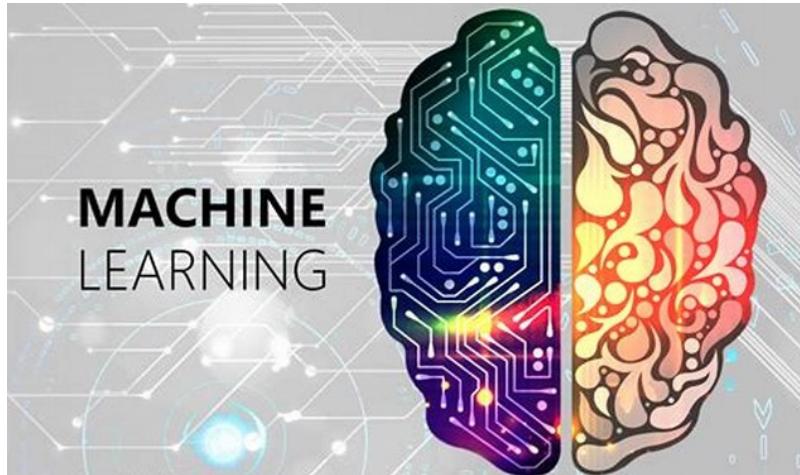
典型深度学习工作流程



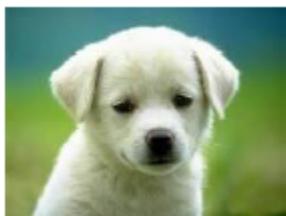
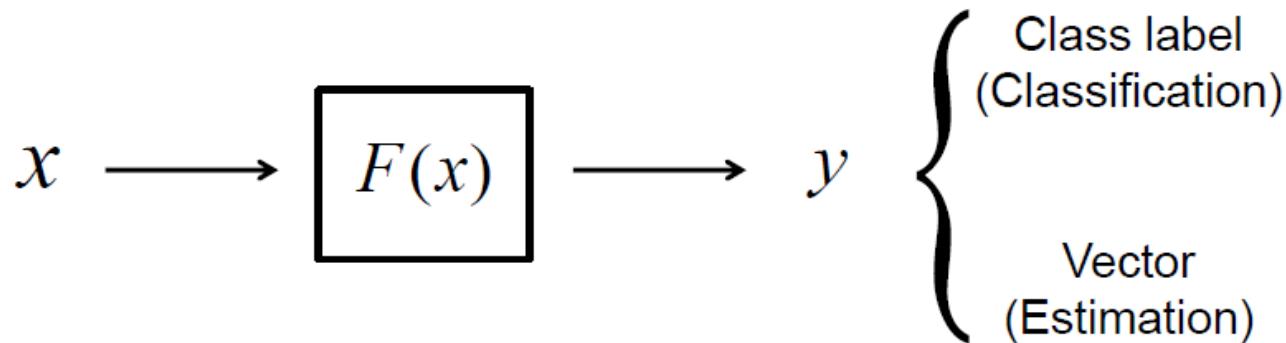
机器学习的最新发展



机器学习的方法



机器学习的基本任务



Object recognition $\longrightarrow \{\text{dog, cat, horse, ...}\}$

基本术语

求职者	笔试成绩	语言	工作经验	是否录取
1	A	Java	2年	录取
2	B	Python	2年	录取
3	C	C++	1年	录取
4	A	C	0年	不录取

- **数据集**: 记录的集合，例如上表中4个求职者的所有信息。
- **样本 (示例)** : 描述一个对象或者事件的记录。例如表中的描述每个求职者的记录就是一个样本。
- **特征**: 反应对象在某方面的表现或者性质的事项，例如笔试成绩，语言，工作经验。
- **属性 (样本) 空间**: 属性张成的空间。例如我们把笔试成绩，语言，工作经验作为3个坐标轴，每个求职者都能在这3维空间中找到自己的位置。
- 由于空间中每个点对应一个向量，因此我们可以把每一个样本称为一个**特征向量**。

基本术语

求职者	笔试成绩	语言	工作经验	是否录取
1	A	Java	2年	录取
2	B	Python	2年	录取
3	C	C++	1年	录取
4	A	C	0年	不录取

- 从数据中学得模型的过程称为**学习 (learning)**或**训练 (training)**，这个过程通过执行某个**学习算法**来完成。
- 比如给定上述求职者的信息，面试后再给每个求职者一个**标记**“录取”或者“不录取”。
- 我们通过给定标记（录取或者不录取）的数据中**训练模型**（例如决策树），从而不用面试新的求职者，直接用决策树模型就可以判断是否录取。

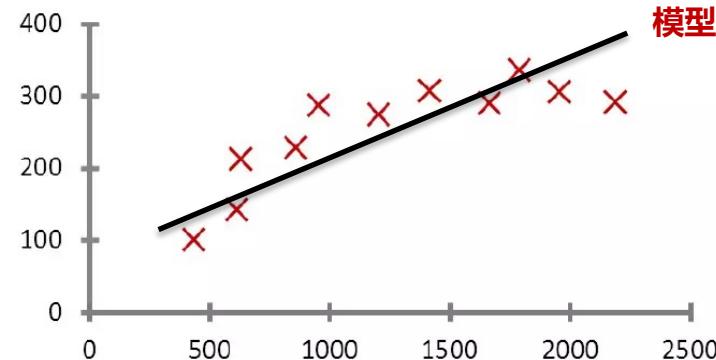
基本术语

- 简单讲，**模型是样本空间到输入空间的映射**，一般由模型的**假设函数与参数 w**组成，例如房价示例中简单的线性模型：

$$\text{房子价格} = \text{beta} * \text{房屋面积} + \text{alpha}$$

- 模型参数**为 beta 和 alpha 。当训练好模型参数 beta 与 alpha 后，就可以输入一个房屋面积，进而预测房屋价格了。

问题：现在我手里有一栋房子需要售卖，我应该给它标上多大的价格？房子的面积是 100 平方米，价格是 100 万，120 万，还是 140 万？



基本术语

- 训练过程中使用的数据称为 “**训练数据**” (**training data**)，其中每一个样本称为一个**训练样本**(**training sample**)，训练样本组成的集合称为**训练集**(**training set**)。
- 学习到的模型对应了数据的某种潜在规律，因此称模型为 “**假设**” (**hypothesis**)。这种 “潜在规律” 称为 “**真相**” 。学习的过程就是为了找出或者逼近真相。
- 若我们欲预测的是离散值，例如 “生存” 或者 “死亡”，此类学习任务称为 “**分类**” ；若预测的值是连续值，例如预测房价，则此类学习任务称为 “**回归**” 。

机器学习方法的分类

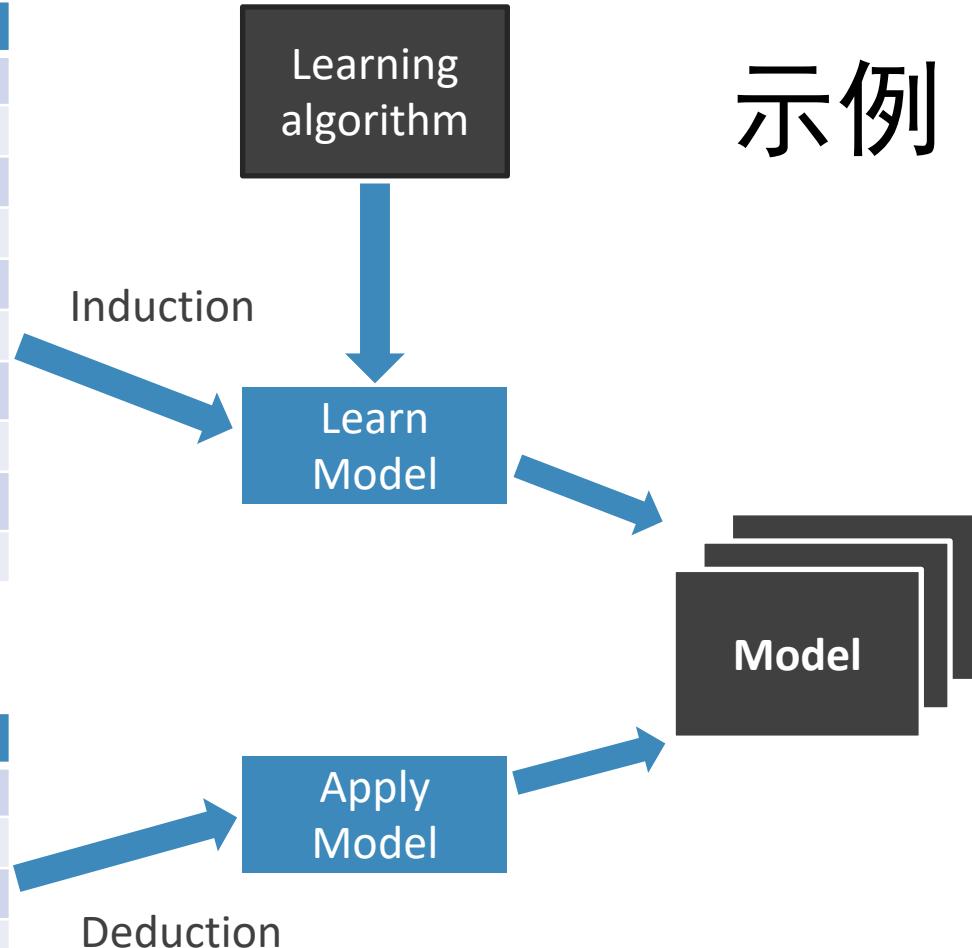
- 按照训练的数据有无**标记**，将机器学习分为**监督学习**和**无监督学习**。
- 根据输出的数据来分，可以分为**离散输出**（通常用来表示个体的类别，比如把人分为两类，男人和女人）和**连续输出**（指那些在一定区间内可以任意取值的数据，如公司收入、客户的价值、生产流程的能耗等）
- 将以上四类排列组合，就形成了四种不同类型的机器学习问题：
 - 通过监督学习，得到离散输出，**分类问题**
 - 通过无监督学习，得到离散输出，**聚类问题**
 - 通过监督学习，得到连续输出，**回归问题**
 - 通过无监督学习，得到连续输出，**降维问题**

输出类型	监督类型	
	监督学习	无监督学习
离散输出	分类问题	聚类问题
连续输出	回归问题	降维问题

分类的定义

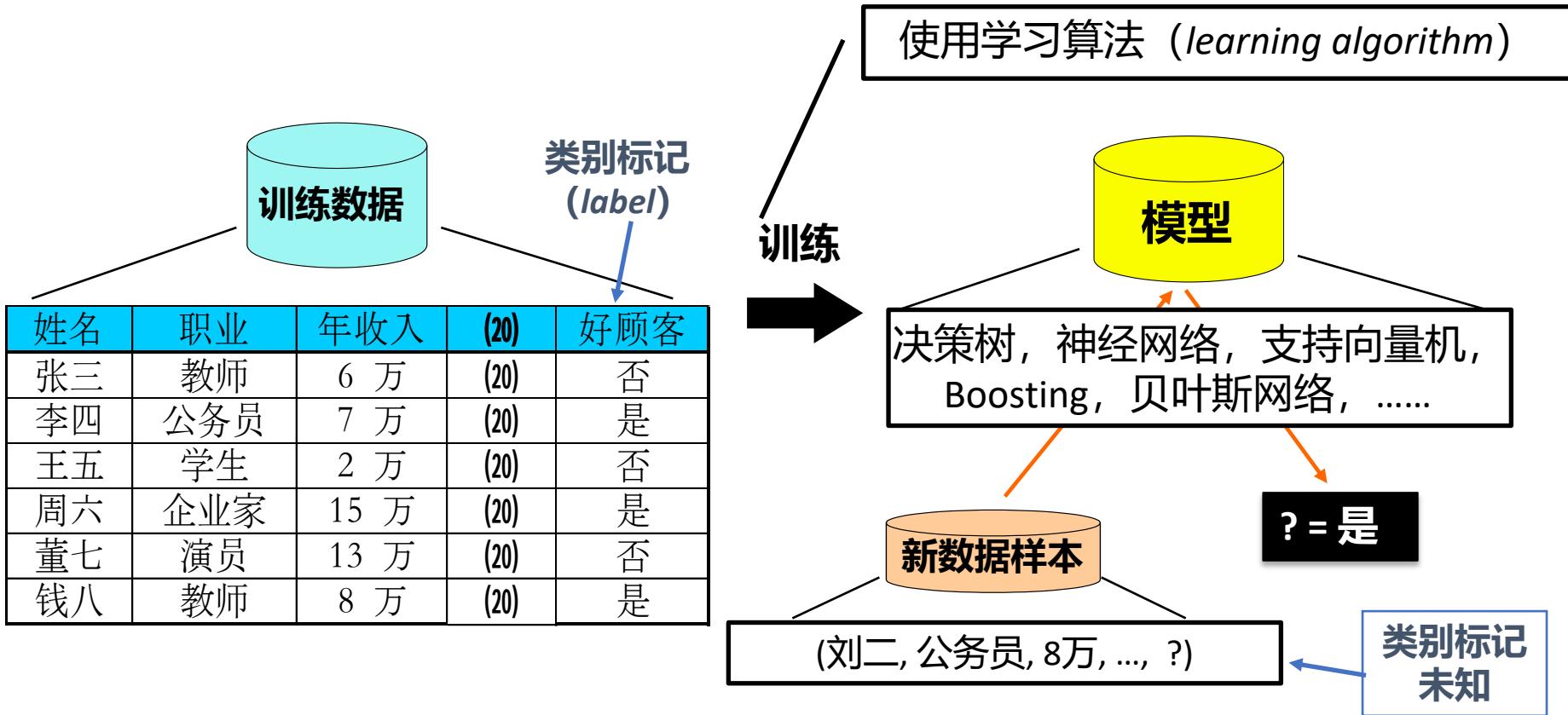
- 给定一个记录(样本)集合 (**训练集**)
 - 每条记录有一些**属性**组成, 其中一个属性为**类别**
 - $(x_1, x_2, \dots, x_n, c)$
- 找到一个将类别属性表示为其他属性的函数的模型。 (如 $c = f(x)$)
- **目标**: 未见过的记录尽可能准确地被分类.
 - 一个**测试集**用来确定模型的精度.
 - 通常, 给定的数据集被分成训练集和测试集, 训练集用于建立模型, 而测试集用于检验该模型.

TID	Attrib1	Attrib2	Attrib3	class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes



TID	Attrib1	Attrib2	Attrib3	class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

典型的机器学习过程



常用的方法

决策树

(Decision trees)

规则归纳

(Rule induction)

贝叶斯学习

(Bayesian
learning)

神经网络

(Neural networks)

支持向量机

(Support Vector
Machine)

Ensemble方法

(AdaBoost, Bagging...)

k-最近邻

k-最近邻法介绍

下面图片中只有三种豆，有三个豆是未知的种类，如何判定他们的种类？



1968年，Cover 和 Hart 提出了最初的近邻法

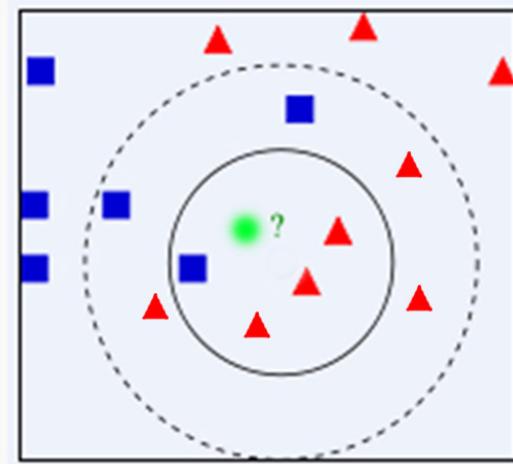
最近邻算法

- 提供一种思路，即：**未知的豆离哪种豆最近就认为未知豆和该豆是同一种类。**
- 由此，我们引出**最近邻算法**的定义：
 - 为了判定未知样本的类别，以全部训练样本作为代表点，计算未知样本与所有训练样本的距离，并以最近邻者的类别作为决策未知样本类别的唯一依据。



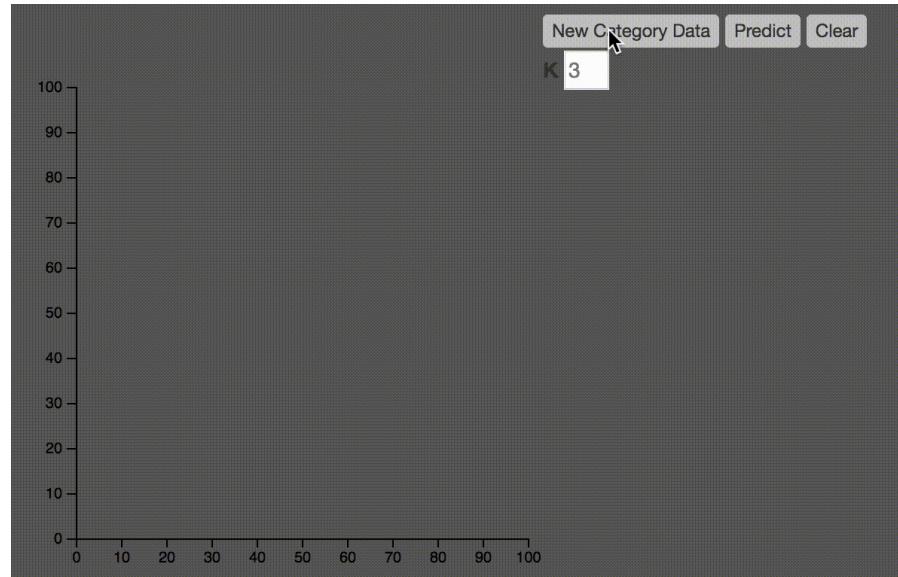
k-最近邻算法

- **k-最近邻算法**是最近邻算法的延伸。
- 基本思路：选择未知样本一定范围内确定个数的 k 个样本，该 k 个样本大多数属于某一类型，则未知样本判定为该类型。



k-最近邻算法的实现步骤

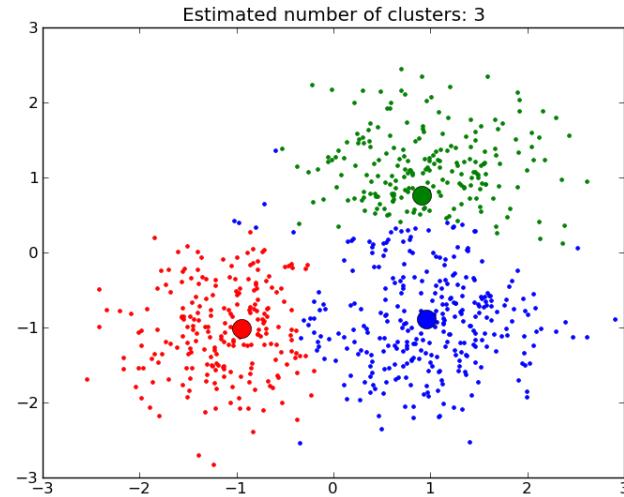
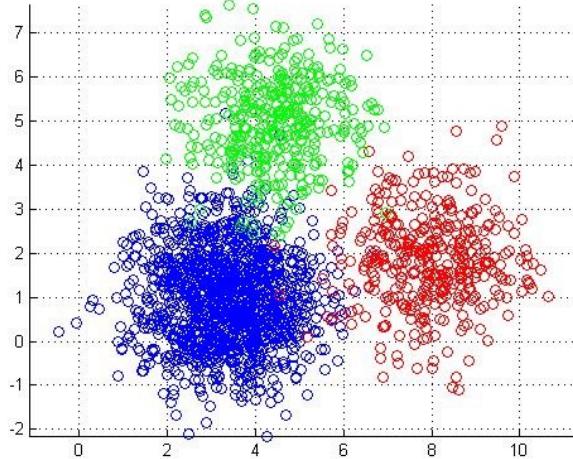
- 1、初始化距离为最大值
- 2、计算未知样本和每个训练样本的距离dist
- 3、得到目前K个最临近样本中的最大距离maxdist
- 4、如果dist小于maxdist，则将该训练样本作为k-最近邻样本
- 5、重复步骤2、3、4，直到未知样本和所有训练样本的距离都算完
- 6、统计K个最近邻样本中每个类别出现的次数
- 7、选择出现频率最大的类别作为未知样本的类别



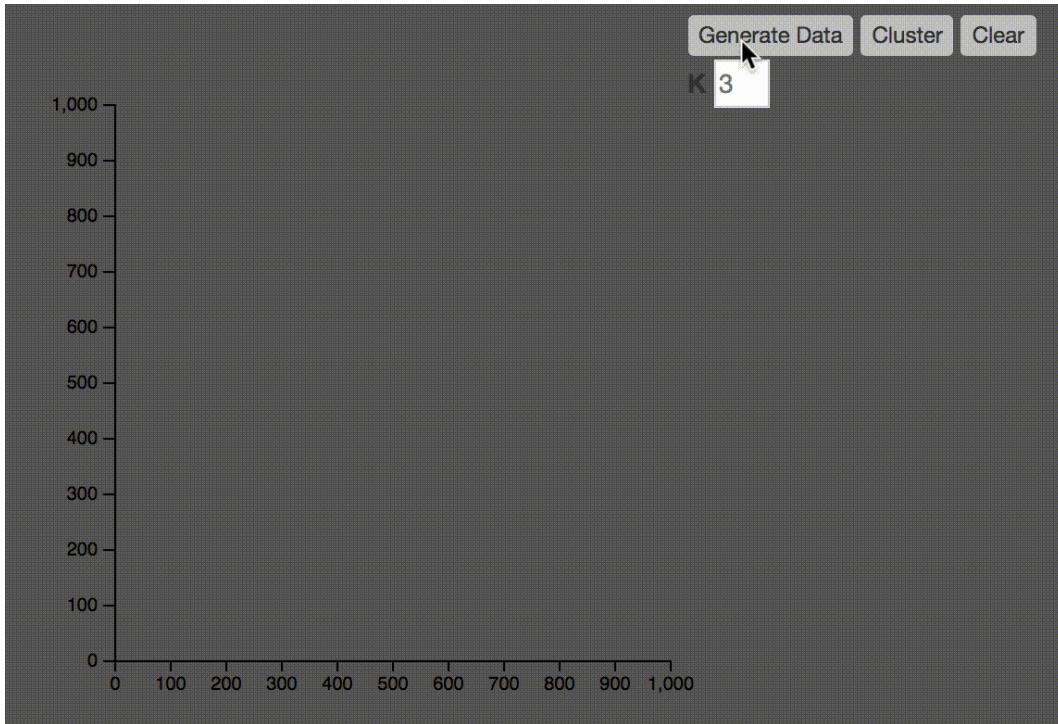
<https://codepen.io/gangtao/pen/ayPVQz>

聚类

- 这类方法有一个统称，即**无监督算法**，其中最典型的代表就是**聚类**。聚类就是计算种群中的距离，根据距离的远近将数据划分为多个族群。
- 聚类算法中最典型的代表就是 **K-Means 算法**。



K-Means聚类算法演示



<https://codepen.io/gangtao/pen/vJaYya>

过拟合与欠拟合

- **训练误差**: 根据训练集学得模型之后，模型在训练集上的误差称为“训练误差”或“经验误差”。若是回归任务，则训练误差为模型的预测值与真实值的差的平方；若是分类任务，则是误分类的训练样本与总样本的比值，称这个比值为**错误率**。
- **泛化误差**: 模型在新样本（没有出现在训练集中）上的误差称为“泛化误差”，
- 泛化误差是我们追求的目标。但实际上我们并不知道新样本是什么，我们能做的是学得一个**训练误差**很小、在训练集上表现良好的学习器。

过拟合与欠拟合

- 由于实际中，我们能做的是降低训练误差。但当模型很复杂时，很多时候能导致过拟合。
- **过拟合**: 当模型把训练集学得“太好”了，即模型在训练集上表现的非常好。这很可能把训练样本自身的一些特点当作了所有潜在样本都会具有的一般性质，这样就会导致泛化能力下降，这种现象称为过拟合。
- **欠拟合**: 当模型过于简单，对训练集的一般性质尚未学好，模型在训练集上都表现的不好，此现象称为欠拟合。欠拟合通常意味着模型不够好，需要继续改进模型。

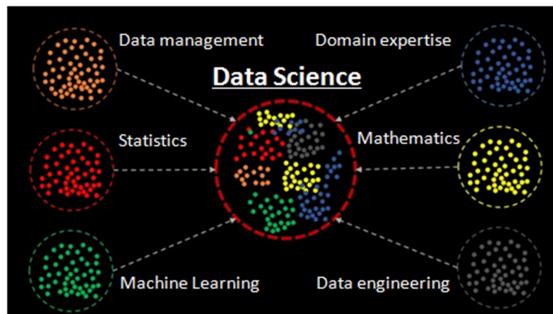
过拟合与欠拟合



过拟合与欠拟合直观类比

数据思维与实践

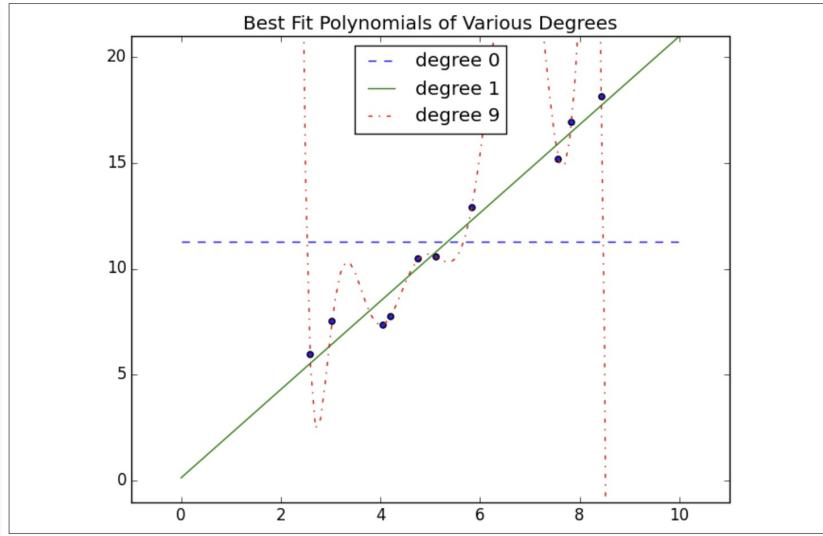
第05讲 数据建模与分析



- 什么是数据建模?
- 机器学习基础
- **Python 机器学习**
- 开源数字王国中的建模与分析

过拟合与欠拟合的处理

方法：拆分数据集



```
import random
from typing import TypeVar, List, Tuple
X = TypeVar('X') # generic type to represent a data point

def split_data(data: List[X], prob: float) -> Tuple[List[X], List[X]]:
    """Split data into fractions [prob, 1 - prob]"""
    data = data[:] # Make a shallow copy
    random.shuffle(data) # because shuffle modifies the list.
    cut = int(len(data) * prob) # Use prob to find a cutoff
    return data[:cut], data[cut:] # and split the shuffled list there.

data = [n for n in range(1000)]
train, test = split_data(data, 0.75)

# The proportions should be correct
assert len(train) == 750
assert len(test) == 250

# And the original data should be preserved (in some order)
assert sorted(train + test) == data
```

```
from sklearn.model_selection import train_test_split
import pandas as pd

# features是特征, labels是目标变量
data = pd.read_csv('your_data.csv')
features = data.drop('target', axis=1)
labels = data['target']

# 划分数据集
X_train, X_test, y_train, y_test = train_test_split(
    features, labels, test_size=0.2, random_state=42)
```

正确性的评估

垃圾邮件混淆矩阵

	Spam	Not spam
Predict "spam"	True positive	False positive
Predict "not spam"	False negative	True negative

名字预测患病混淆矩阵

	Leukemia	No leukemia	Total
"Luke"	70	4,930	5,000
Not "Luke"	13,930	981,070	995,000
Total	14,000	986,000	1,000,000

```
from sklearn.metrics import  
accuracy_score, precision_score,  
recall_score, f1_score,  
confusion_matrix
```

```
def accuracy(tp: int, fp: int, fn: int, tn: int) -> float:  
    correct = tp + tn  
    total = tp + fp + fn + tn  
    return correct / total
```

准确率

```
assert accuracy(70, 4930, 13930, 981070) == 0.98114
```

```
def precision(tp: int, fp: int, fn: int, tn: int) -> float:  
    return tp / (tp + fp)
```

精确率

```
assert precision(70, 4930, 13930, 981070) == 0.014
```

```
def recall(tp: int, fp: int, fn: int, tn: int) -> float:  
    return tp / (tp + fn)
```

召回率

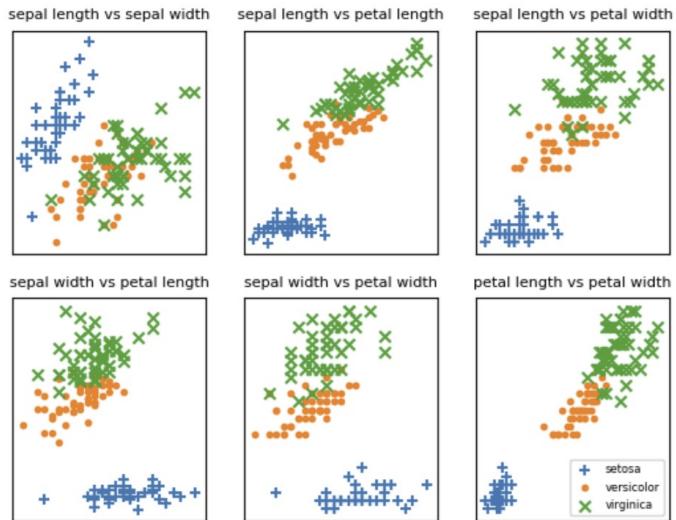
```
assert recall(70, 4930, 13930, 981070) == 0.005
```

```
def f1_score(tp: int, fp: int, fn: int, tn: int) -> float:  
    p = precision(tp, fp, fn, tn)  
    r = recall(tp, fp, fn, tn)
```

F1得分

```
return 2 * p * r / (p + r)
```

k-最近邻算法



鸢尾花数据集

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from sklearn.decomposition import PCA

# 加载鸢尾花数据集
iris = datasets.load_iris()
X = iris.data # 特征
y = iris.target # 标签

# 使用PCA进行数据降维，以便可视化
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X)

# 将数据集分为训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)

# 初始化K近邻分类器，选择K值（这里选择K=3）
knn_classifier = KNeighborsClassifier(n_neighbors=3)
# 在训练集上训练模型
knn_classifier.fit(X_train, y_train)
# 在测试集上进行预测
y_pred = knn_classifier.predict(X_test)
# 计算分类准确率
accuracy = accuracy_score(y_test, y_pred)
```

朴素贝叶斯

Accuracy: 0.8503978779840848

	precision	recall	f1-score	support
alt.atheism	0.84	0.89	0.86	151
comp.graphics	0.63	0.91	0.74	202
comp.os.ms-windows.misc	0.93	0.22	0.36	195
comp.sys.ibm.pc.hardware	0.60	0.86	0.71	183
comp.sys.mac.hardware	0.91	0.87	0.89	205
comp.windows.x	0.84	0.83	0.84	215
misc.forsale	0.94	0.62	0.75	193
rec.autos	0.88	0.93	0.91	196
rec.motorcycles	0.96	0.92	0.94	168
rec.sport.baseball	0.98	0.96	0.97	211
rec.sport.hockey	0.96	0.96	0.96	198
sci.crypt	0.87	0.96	0.91	201
sci.electronics	0.90	0.83	0.86	202
sci.med	0.94	0.92	0.93	194
sci.space	0.90	0.98	0.94	189
soc.religion.christian	0.78	0.99	0.87	202
talk.politics.guns	0.88	0.93	0.90	188
talk.politics.mideast	0.92	1.00	0.96	182
talk.politics.misc	0.78	0.90	0.84	159
talk.religion.misc	0.97	0.45	0.61	136
accuracy			0.85	3770
macro avg	0.87	0.85	0.84	3770
weighted avg	0.87	0.85	0.84	3770

新闻分类结果

```
from sklearn.datasets import fetch_20newsgroups
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn import metrics

# 加载数据集
newsgroups = fetch_20newsgroups(subset='all')

# 将文本数据转换为词袋模型
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(newsgroups.data)

# 将数据集分为训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(
    X, newsgroups.target, test_size=0.2, random_state=42)

# 创建朴素贝叶斯分类器
classifier = MultinomialNB()

# 在训练集上训练分类器
classifier.fit(X_train, y_train)

# 在测试集上进行预测
y_pred = classifier.predict(X_test)

# 评估分类器性能
accuracy = metrics.accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy}')
```

聚类分析



图像 (像素) 聚类

```
import cv2
import numpy as np
from sklearn.cluster import KMeans
from sklearn.utils import shuffle

# 读取图像
image_path = "lena.png"
image = cv2.imread(image_path)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB) # 转换颜色通道顺序

# 将图像转为一维数组
original_shape = image.shape
image_array = image.reshape((-1, 3))

# 对颜色进行标准化
image_array = np.array(image_array, dtype=float) / 255.0
# 打乱数据
image_array_sample = shuffle(image_array, random_state=0)[:1000]

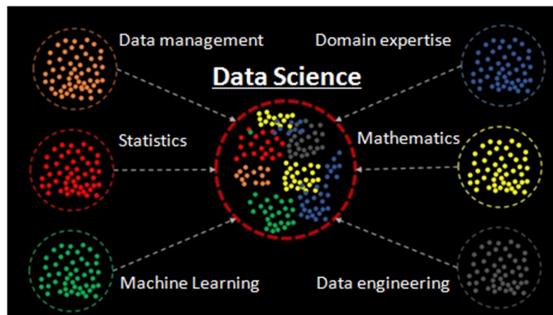
# 使用K均值聚类
kmeans = KMeans(n_clusters=50, random_state=0)
kmeans.fit(image_array_sample)

# 将聚类结果应用到整个图像
segmented_image = kmeans.cluster_centers_[kmeans.predict(image_array)]
# 将一维数组转为图像形状
segmented_image = segmented_image.reshape(original_shape)

# 显示聚类后的图像
plt.imshow(segmented_image)
```

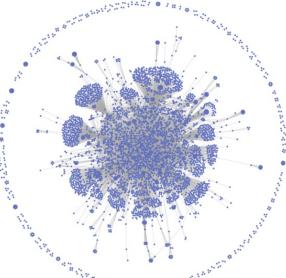
数据思维与实践

第05讲 数据建模与分析

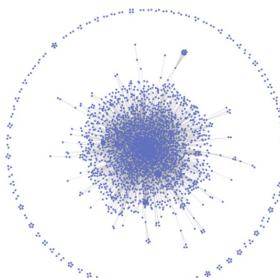


- 什么是数据建模?
- 机器学习基础
- Python 机器学习
- **开源数字王国中的建模与分析**

分类问题：开发者分类



(a) VSCode



(b) Kubernetes



(c) Spark

项目	分类	开发者角色	规模 (占比)	提交量	评论量
VSCode	外围节点	外围开发者	3643(19%)	236	1840
	最大连通子图节点	边缘开发者	12237(65%)	2213	15996
		核心开发者	3058(16%)	38675	56572
Kubernetes	外围节点	外围开发者	778(16%)	62	992
	最大连通子图节点	边缘开发者	3261(67%)	14413	9966
		核心开发者	815(17%)	42217	93118
Spark	外围节点	外围开发者	65 (10%)	5340	51
	最大连通子图节点	边缘开发者	476 (72%)	20257	3284
		核心开发者	118 (18%)	29484	85596

分类问题：项目多标签分类

Explore **Topics** Trending Collections Events GitHub Sponsors

Topics

Browse popular topics on GitHub.



Ruby

Ruby is a scripting language designed for simplified object-oriented programming.



Unity

Unity is a game engine used to create 2D/3D video games, and simulations for computers, consoles, and mobile devices.



Ansible

Ansible is a simple and powerful automation engine.

All featured topics



3D

3D refers to the use of three-dimensional graphics, modeling, and animation in various industries.



Ajax

Ajax is a technique for creating interactive web applications.

Popular topics

javascript css config
python html bioinformatics
typescript advent-of-code
github-config crack-f1-studio

聚类问题

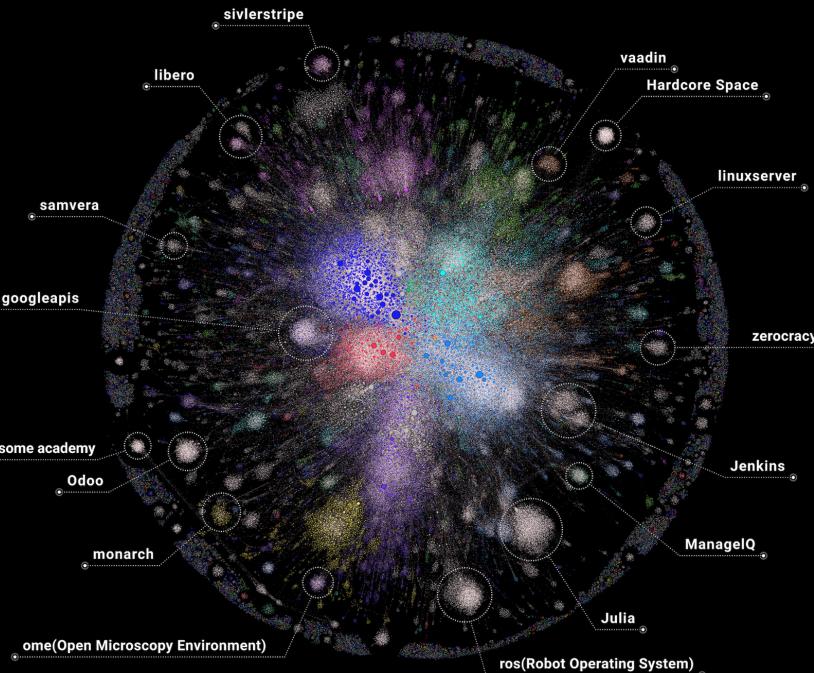
OpenGalaxy 2019

OpenGalaxy is generated by collaboration network of all active GitHub repos in 2019. This graph contains 171,141 nodes and 2,811,489 edges. The generate method can be found in here [1] and the data is from GHArchive [2].

OpenGalaxy 是通过 GitHub 2019 年全域所有活跃项目的协作网络生成的。本图共包含 171,141 个节点和 2,811,489 条边。具体生成方法请参见这里 [1]，数据来自于 GHArchive [2]。

Area/领域	Top Repos/顶级项目	Count/项目数量
ts & frontend	VSCode, TypeScript, react, jest	23,254
cloud native	kubernetes, go, helm, ansible	15,787
AI libs	pandas, numpy, conda, openjournals	14,971
tools	rust, nextcloud, godotengine	13,361
PHP	symfony, laravel, wordpress, magento	8,158
Microsoft	azure-docs, AspNetCore, WSL	6,276
system	homebrew, systemd	6,193
biotech	rstudio, bioconda	6,102
blockchain	bitcoin, ethereum, ipfs	5,141

- [1] http://blog.frankzhao.cn/open_rank_and_open_galaxy/
[2] <http://www.gharchive.org/>



THANK

YOU

