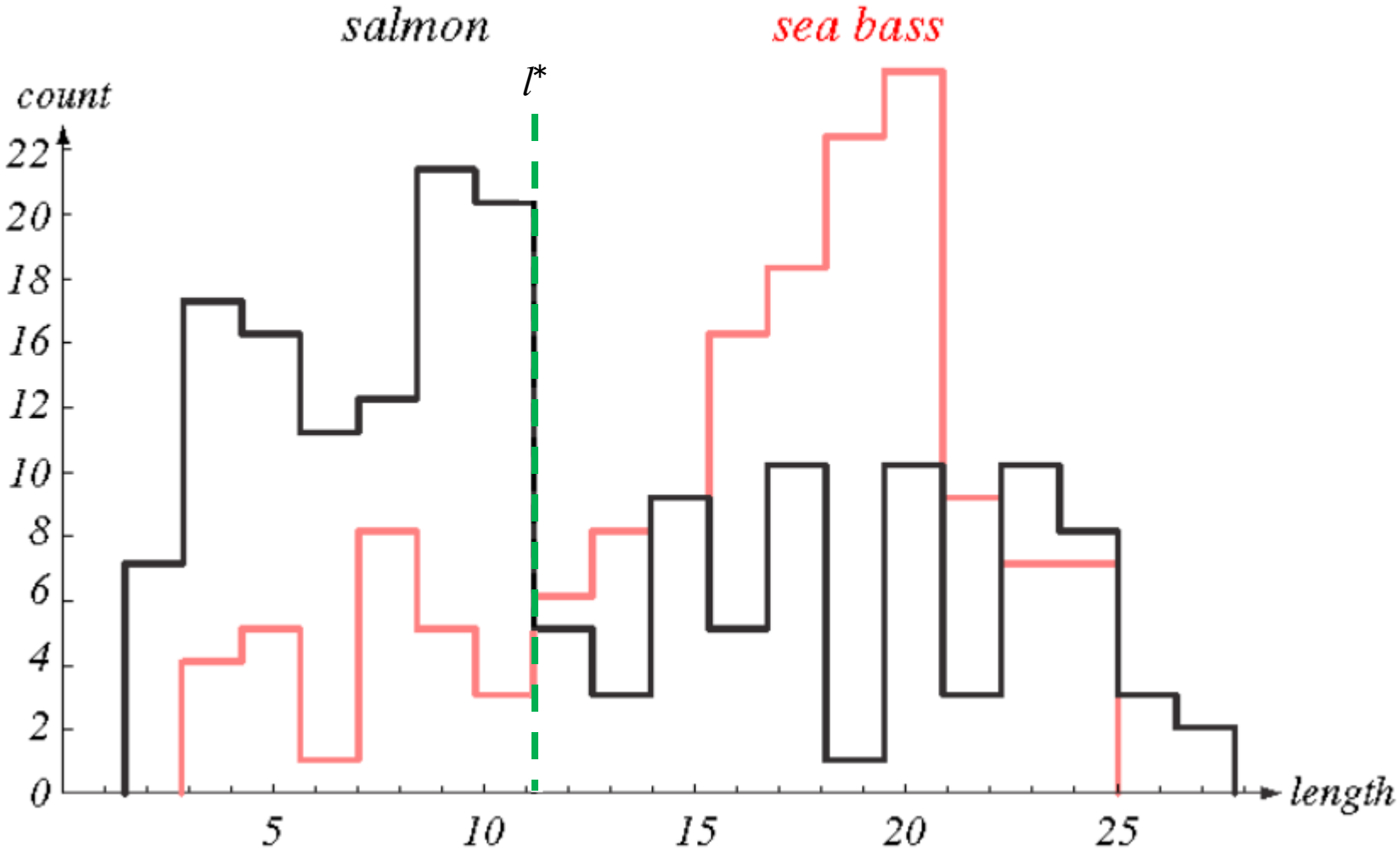
# 机器学习概述

主讲：刘丽珏

# 例——鱼的分类

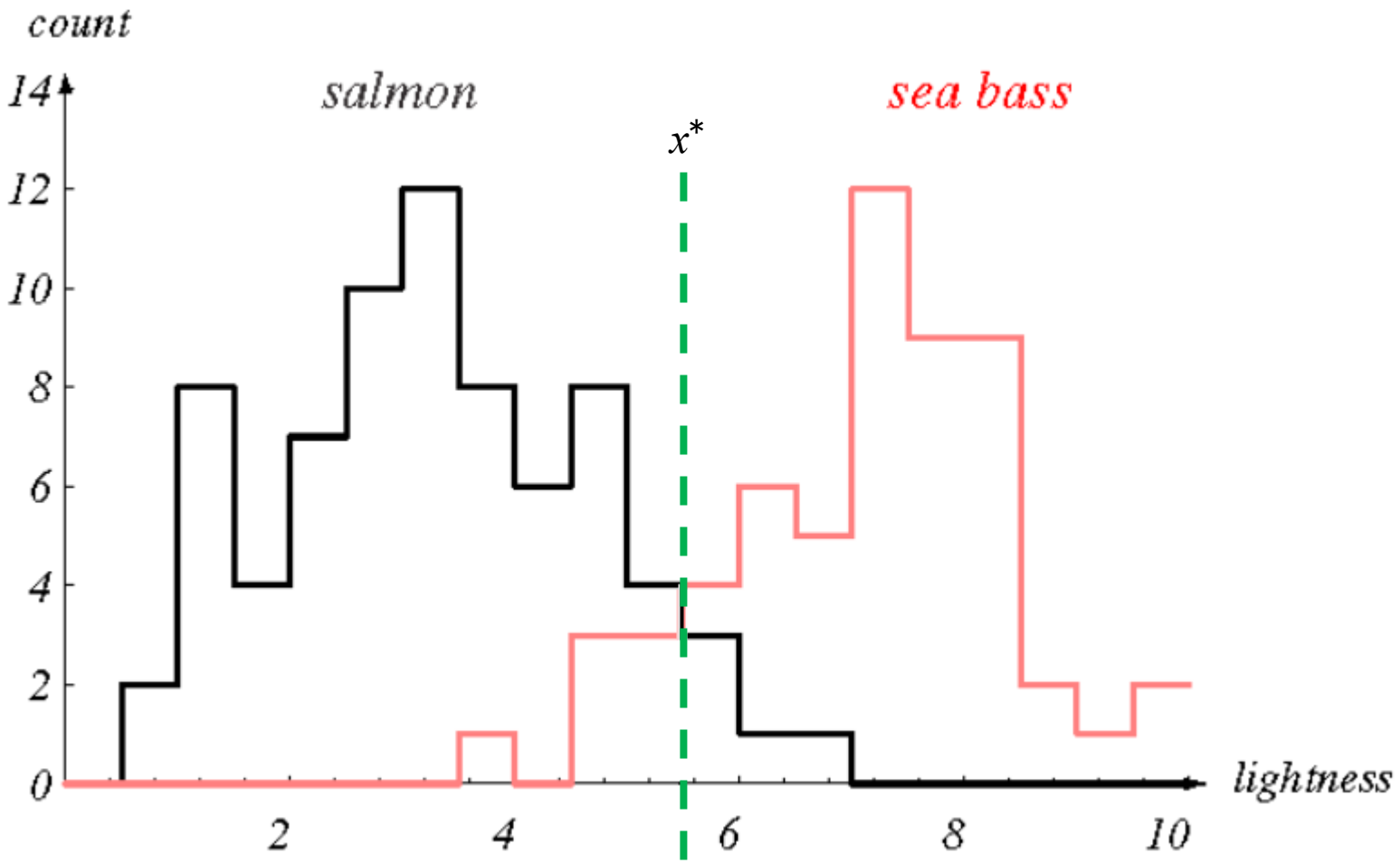
- 选择特征
  - 假设一位渔夫告诉我们，鲈鱼通常比鲑鱼更长
  - 可以将长度作为特征，并根据长度阈值在鲈鱼和鲑鱼之间进行选择
  - 如何选择阈值？




样本集中两种鱼的长度特征直方图，
如何选择阈值$l^*$

# 例——鱼的分类

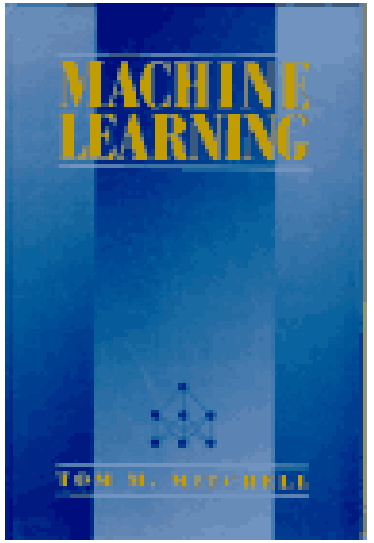
- 尽管鲈鱼平均比鲑鱼长，但仍有许多鱼的不符合这个观察结果
- 换一个特征
  - 鱼鳞的亮度




样本集中两种鱼的亮度特征直方图，
如何选择阈值$x^*$

亮度阈值更容易区分两种鱼，但仍不能做出完美的决策

# 什么是机器学习




- 对于某类任务$T$和性能度量$P$，一个计算机程序被认为可以从经验$E$中学习是指，通过经验$E$改进后，它在任务$T$上由性能度量$P$衡量的性能有所提升。

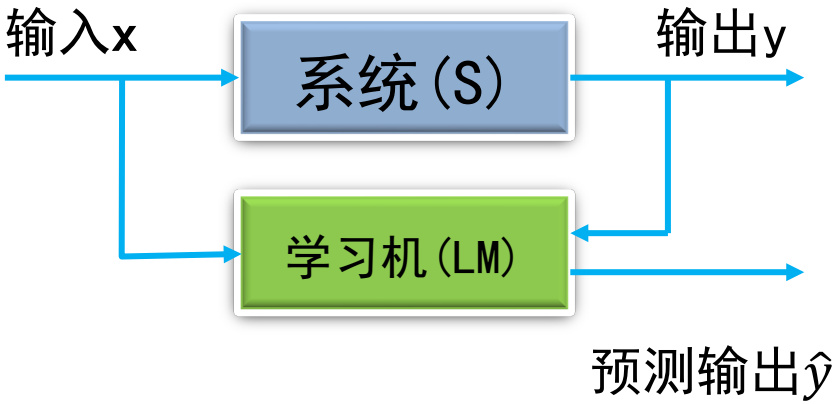
——— Tom Mitchell，1997

- **<T,P,E>**
  - T(任务)
  - P(性能)
  - E(经验)

汤姆·米切尔
(Tom M. Mitchell)

# 什么是机器学习

- 现代机器学习方法主要是建立在统计学习（Statistical learning ）理论之上的
  - learning from data
  - 基于观测数据(样本)，发现规律，预测新数据
  - 数据驱动



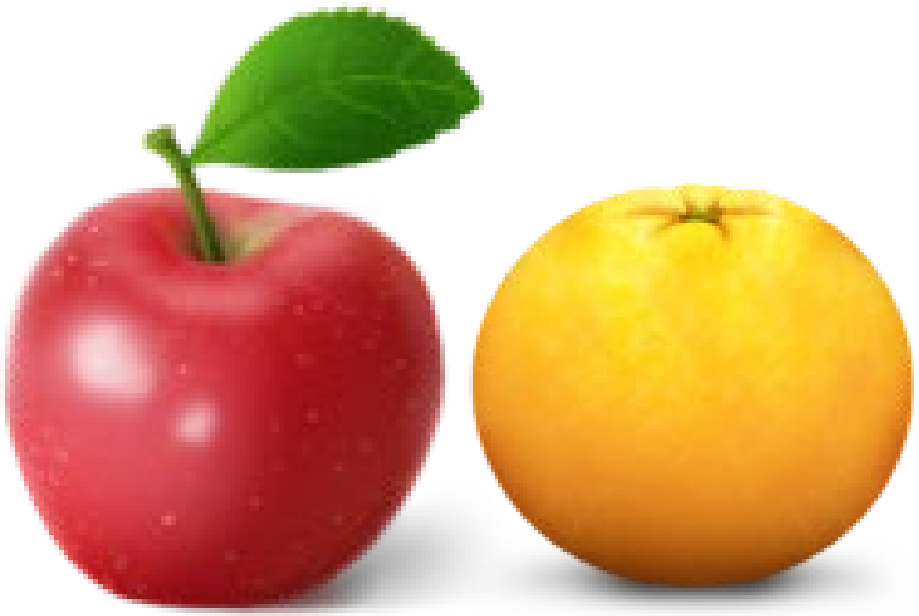

其中系统(S)是研究对象，LM是所求学习机

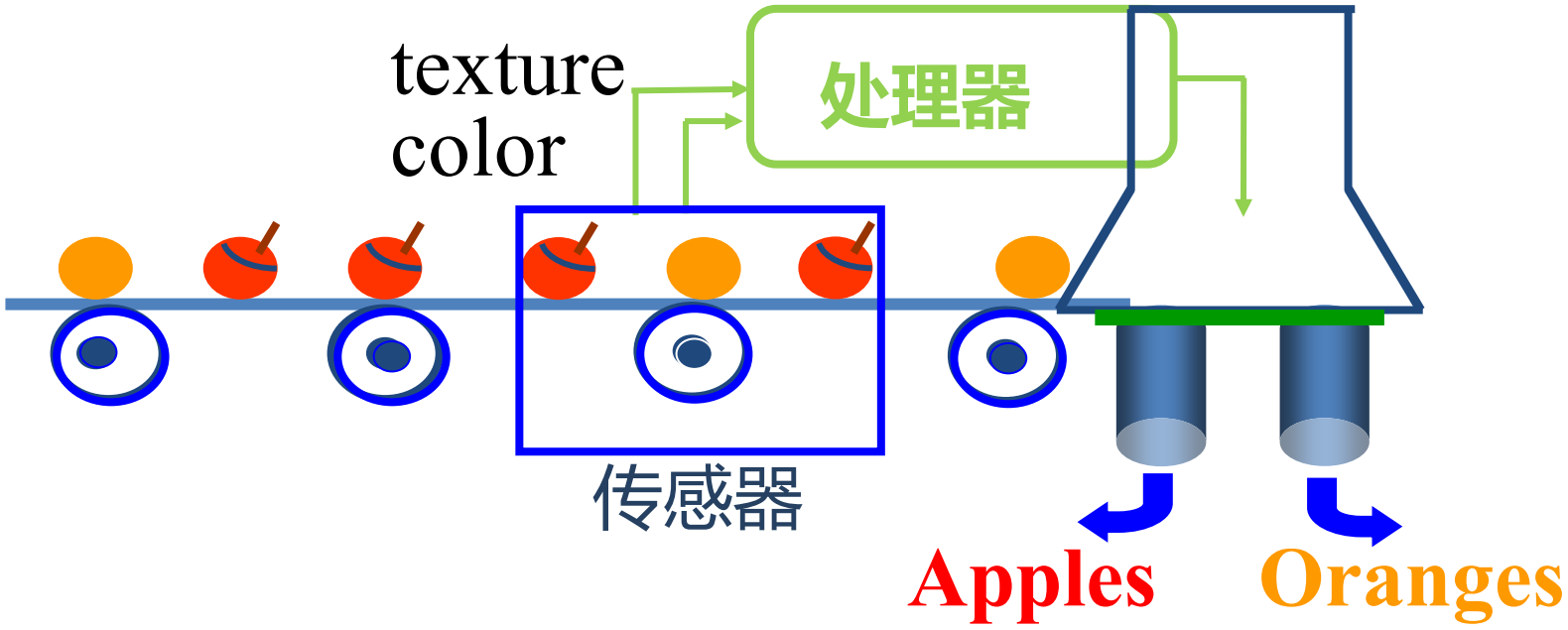
# 机器学习的求解思路

- 模拟人类识别物体需要解决两个困难问题
  - 特征选择与提取
  - 根据样本特征进行分类决策
- 特征选择与提取
  - 早期的机器学习技术采用人工提取的方法
  - 特征工程
- 分类决策
  - 利用训练样本建立起决策面来划分不同类别

# 例：水果分拣

- 回到水果分类






Texture sensor: 1 —— 光滑, –1 —— 粗糙
Color sensor: 1—— 红色, –1 —— 橙色

任务：设计一个水果自动分拣系统

# 例：水果分拣

- 传统程序

  If *texture*=1 && *color*=1 then *apple*

  else *orange*

  OR

  If *texture*=1 && *color*=1 then *apple*

  elseif If *texture*=-1 && *color*=-1 *orange*

- 可能会出现什么问题？
  - 如果传送带上来了一个黄的苹果，或者一个表面光滑的橘子会怎么样？

# 例：水果分拣

- 换一种思路
  - 简便起见传感器信息用一个2维向量$p$表示，则

$$p = \begin{bmatrix} texture \\ color \end{bmatrix} \qquad p(apple) = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \qquad p(orange) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

  - 令$w = [1 \quad 1]$
  - 代码改写成

if $wp$>=1 then *apple*
else *orange*

会发生什么？

# 思考题

- 若将苹果$p=[1\ 1]^T$和桔子$p=[-1\ -1]^T$代入$wp$得到什么结果？
  - $w=[1\ 1]$ 权向量
- 假设传送带上送来了以下一些水果
  - 光滑的桔子$p=[1\ -1]^T$
  - 颜色偏红的桔子$p=[-1\ 1]^T$

会出现什么结果？

- 如果写成传统的IF-ELSE结构的代码需要写多少？
- 在苹果和桔子的分拣中起关键作用的因素是什么？

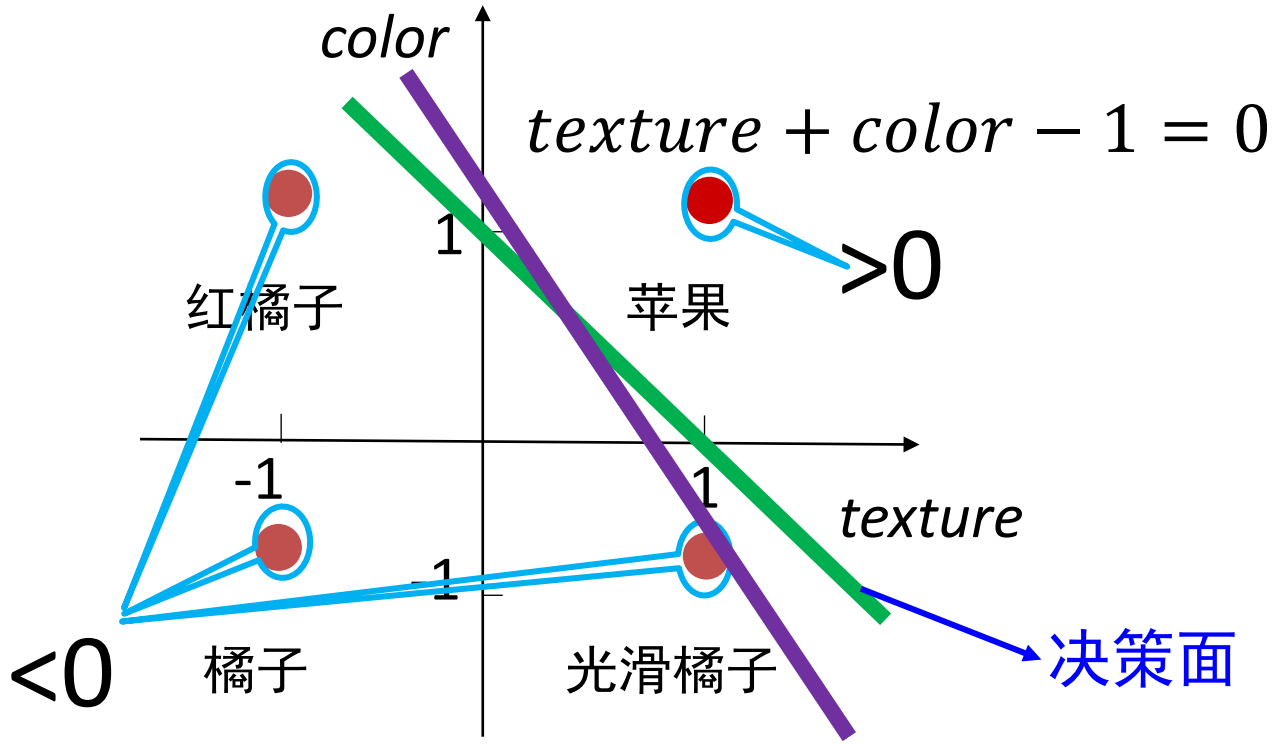
# 思考题解析

- 显然在分拣中起关键作用的是$w$和$wp >= 1$中的1

$$p = \begin{bmatrix} texture \\ color \end{bmatrix}$$

$$p(apple) = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$p(orange) = \begin{bmatrix} -1 \\ -1 \end{bmatrix}$$

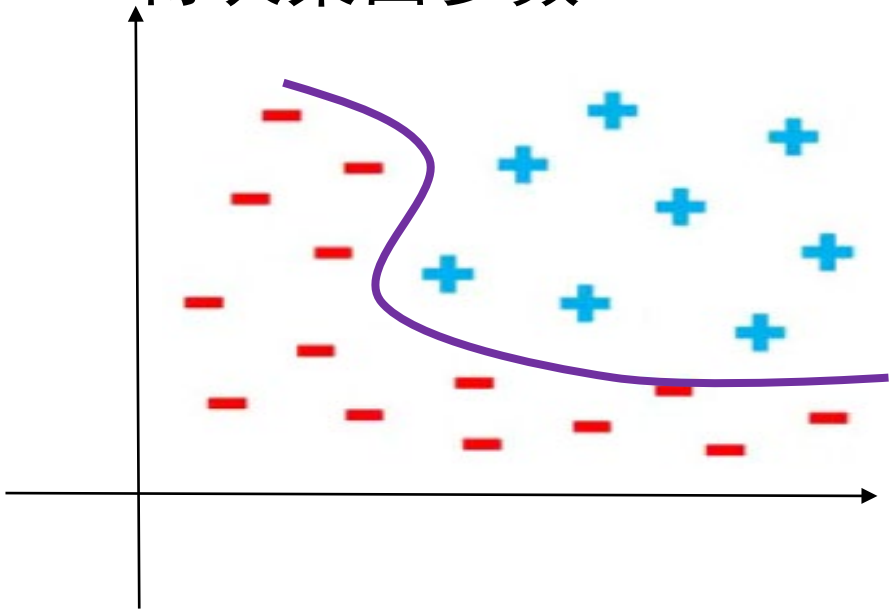


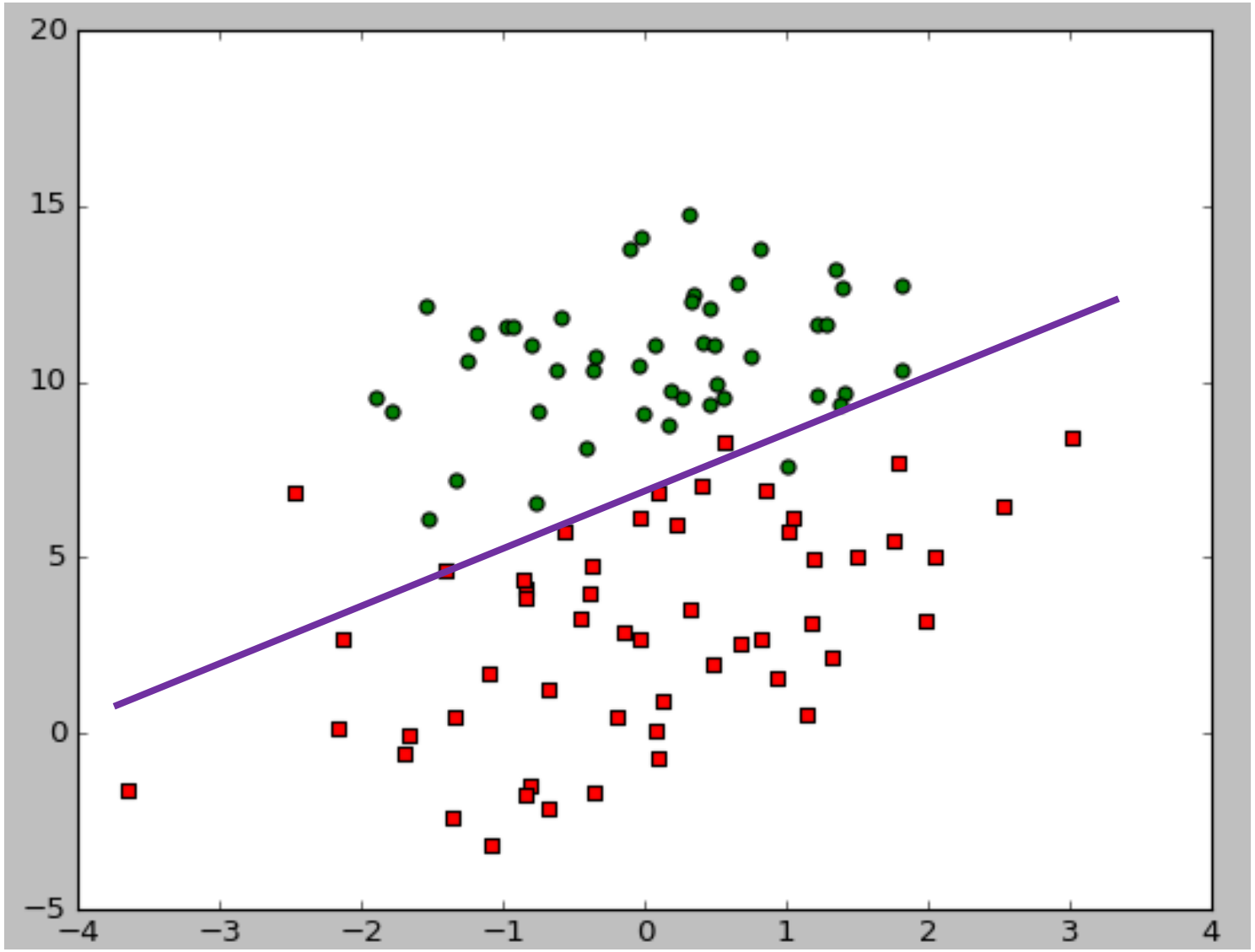

$$w_1 texture + w_2 color + w_3 = 0$$

利用建立决策面的方法代替精确模型

# 决策面在哪里？

- 也会存在错误
- 近似人的错误率
- 机器学习的任务之一
  - 从大量数据中学习获得决策面参数



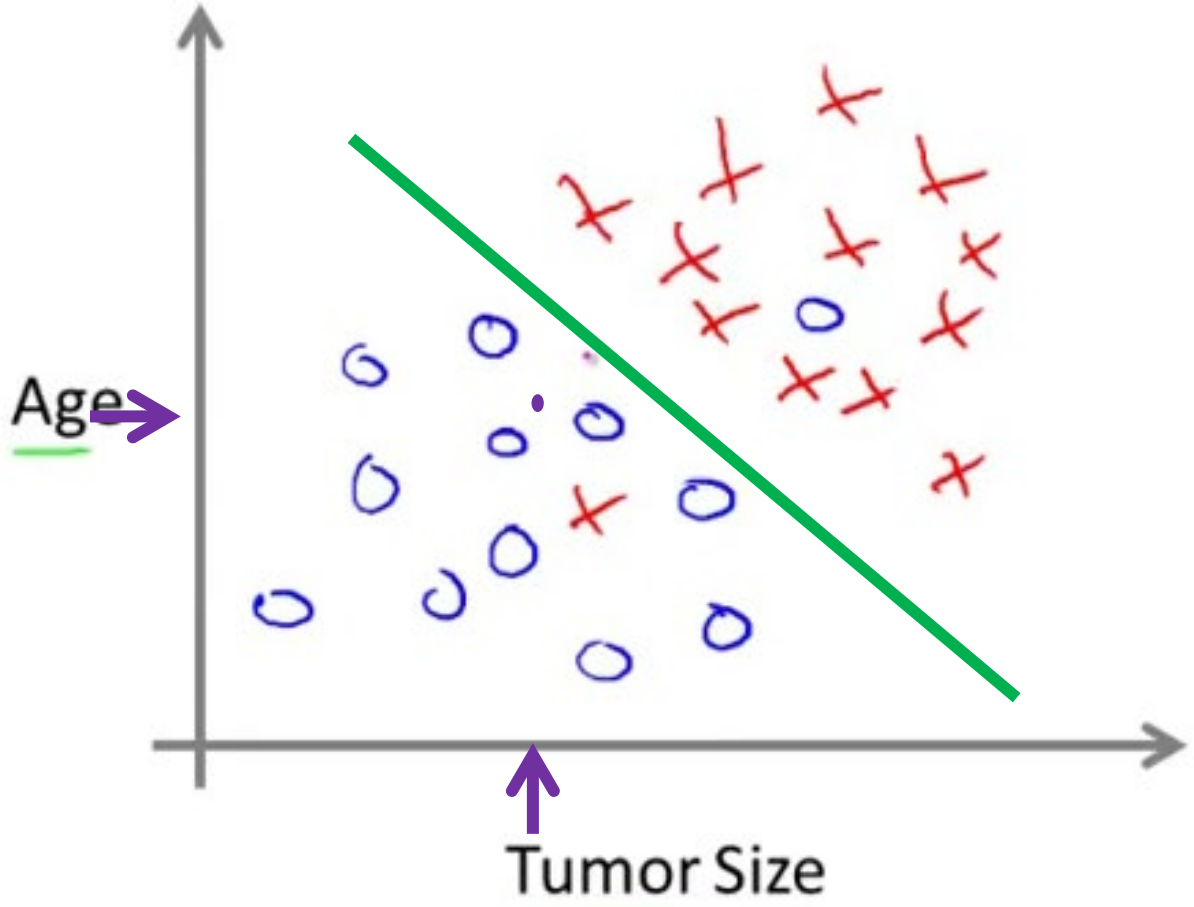


# 有监督学习（Supervised learning）

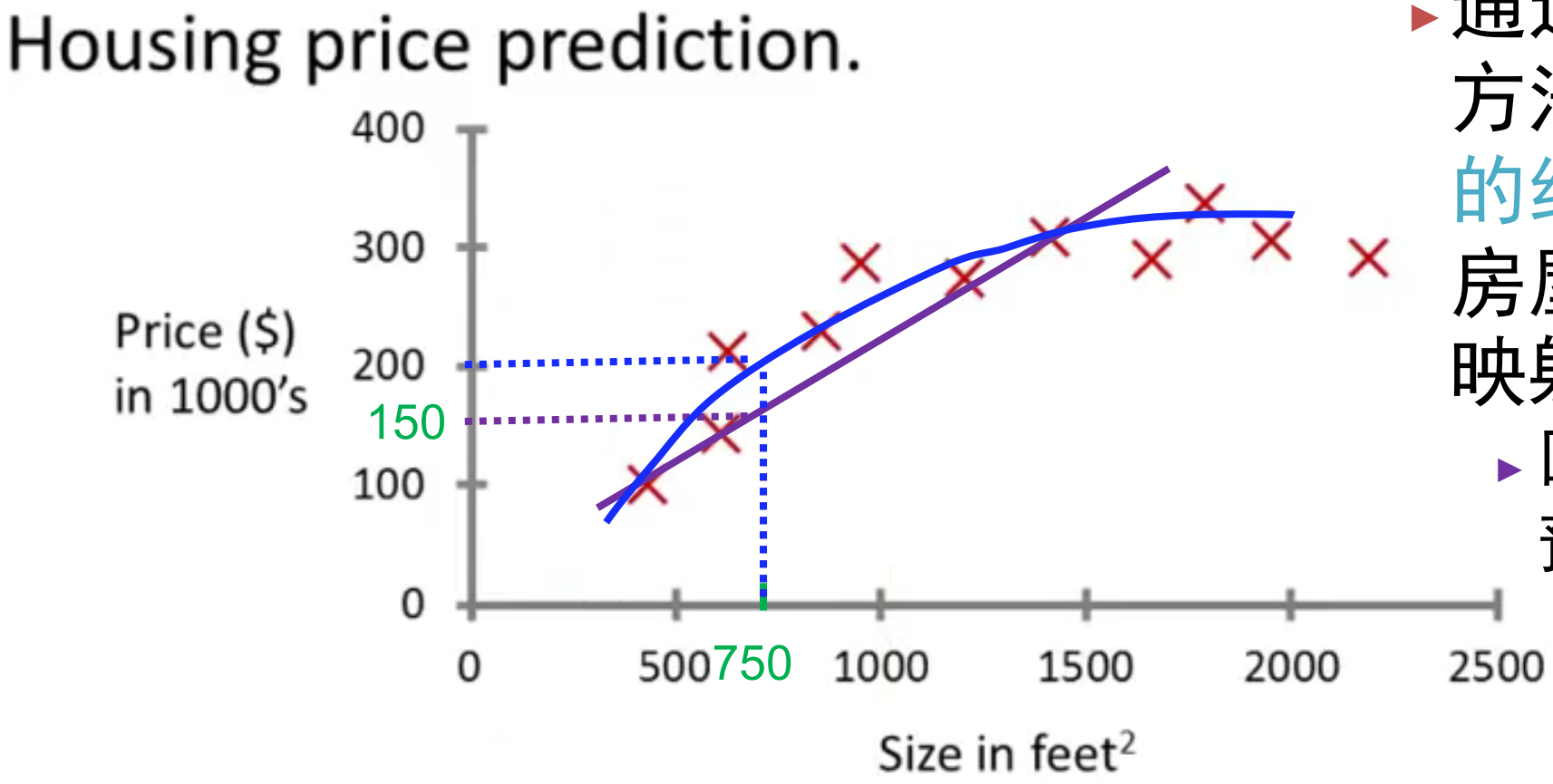
- 从给定输入和输出的训练数据集中学习输入和输出之间的映射函数，然后利用该映射函数预测出测试样本的输出值，其中训练集中的每个样本都由输入和对应的输出(也称之为label)组成(labeled data)
  - 回归(Regression)——预测的目标值是连续变量
  - 分类(Classification)——预测的目标值是离散的

# 例1：肿瘤预测

- 任务：预测肿瘤是良性还是恶性
- 通过机器学习的相关方法，可以根据历史的经验数据自动找到决策面
  - 分类问题：预测离散的输出量



# 例2：房价预测




- 通过机器学习的相关方法，可以根据历史的经验数据自动找到房屋面积与价格间的映射函数
  - 回归问题(Regression)：预测连续的输出量“价格”

# 房价预测数据举例

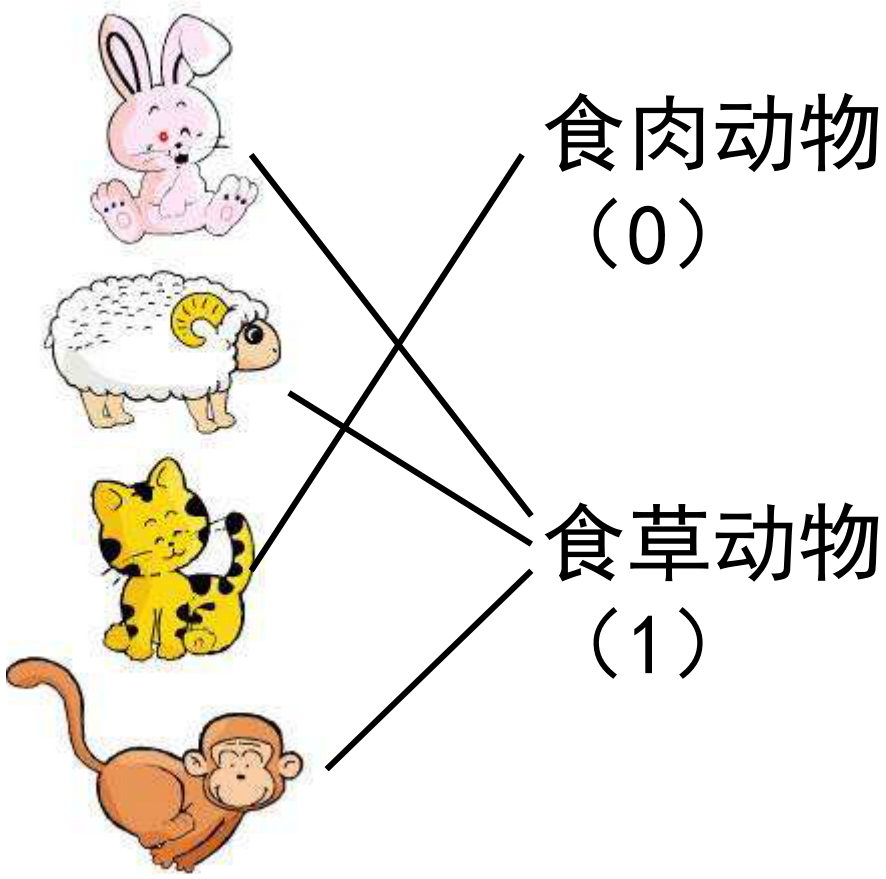
| 面积 | 价格 |
| --- | --- |
| 6.1101 | 17.592 |
| 5.5277 | 9.1302 |
| 8.5186 | 13.662 |
| 7.0032 | 11.854 |
| 5.8598 | 6.8233 |
| … | … |

特征　　标签

# 打标签

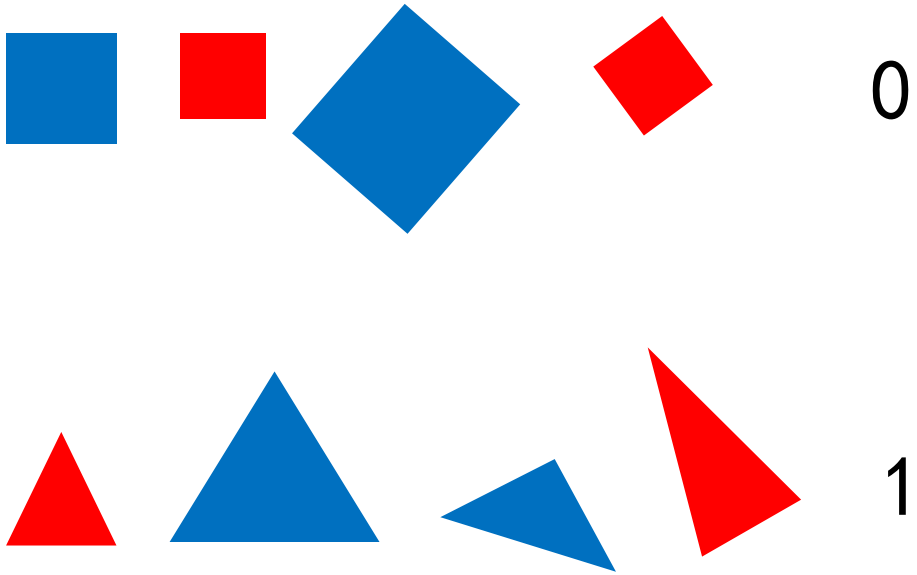
- 有监督学习中工作量最大的数据准备过程

例：请将右图的动物与它的类别连起来

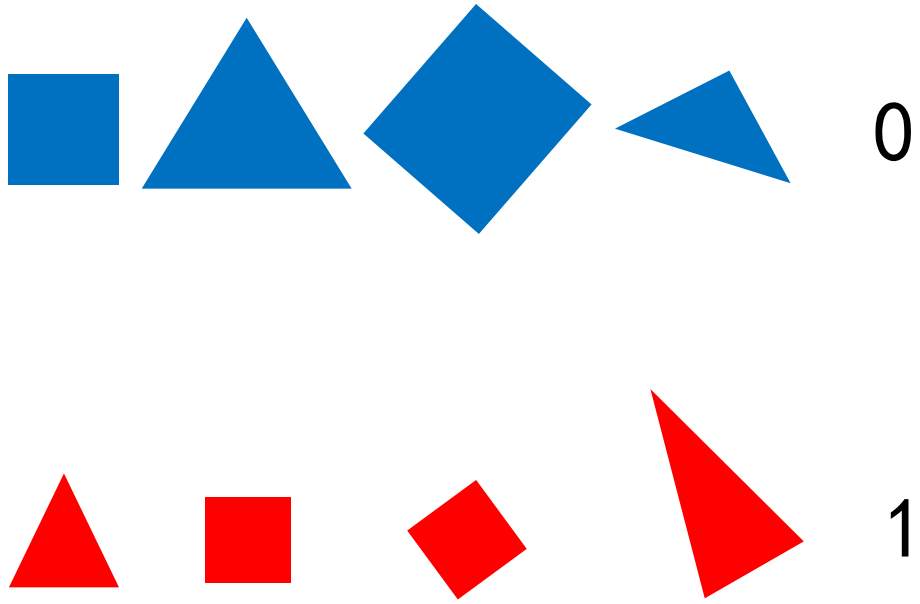



- 打标签的过程即给每个样本数据赋予正确答案的过程
- 有监督学习利用预测结果与标签的误差来调整决策面参数，最终达到正确分类的目的
- 标签同时反映了分类的偏好

# 标签数据举例



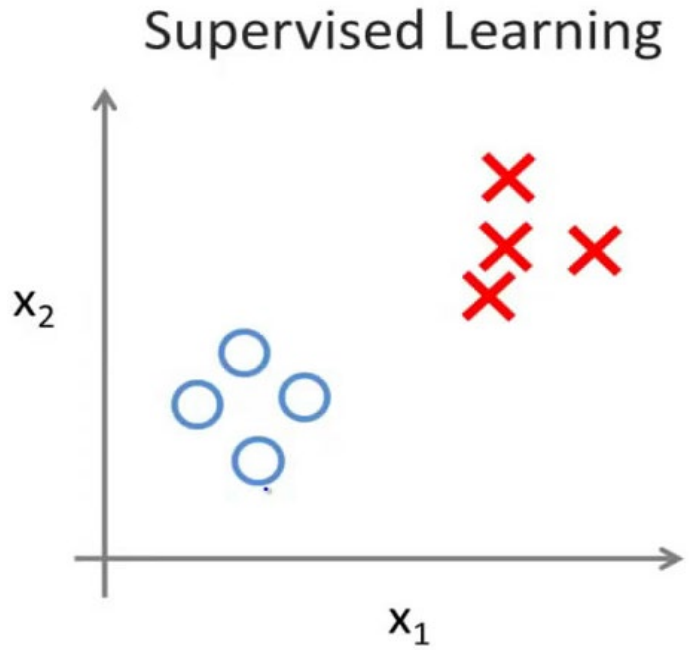

分形状

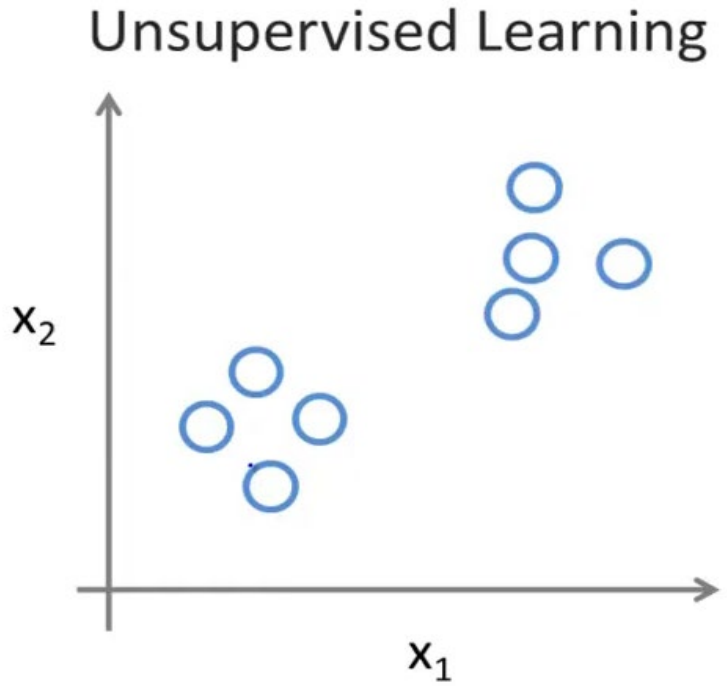



分颜色

# 无监督学习(Unsupervised learning)

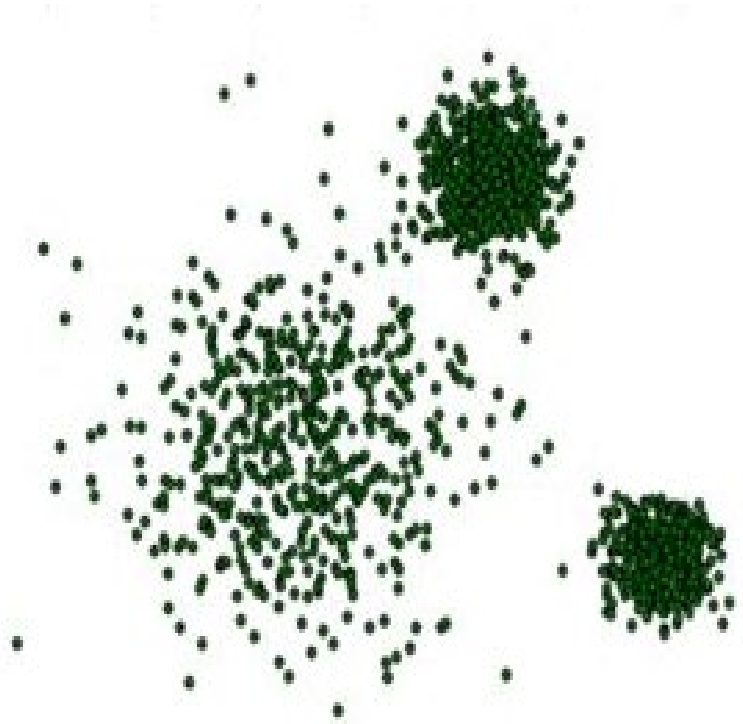
- 发现数据中的隐含结构，其中的数据样本未给定对应的目标值(unlabeled data)。
  - 聚类(Clustering)：将相似的样本划分为不同的组
  - 密度估计(Density Estimation)：找出输入空间的数据分布状况
  - 特征提取、降维
- 在有监督学习中，给定的训练集是一些已标定的数据(labled data)，如正例和反例，而无监督学习的数据却没有标定



# 例3：鸢尾花聚类

- 花园里种了很多鸢尾花，但我们没有每株花是什么品种的数据，能否将一样的鸢尾花放在一起做成花束呢



# 鸢尾花的数据举例

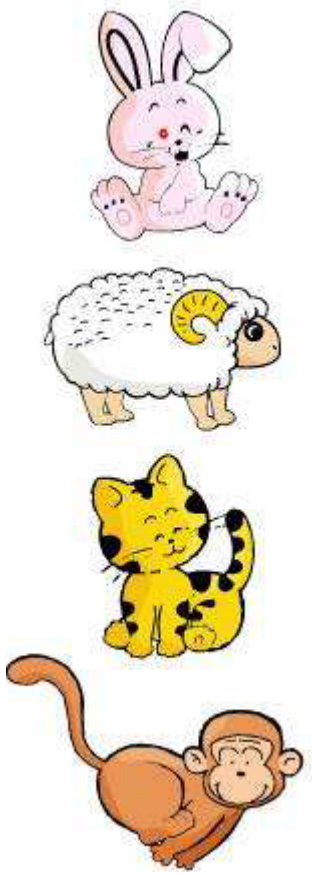
| 萼片长度 | 萼片宽度 | 花瓣长度 | 花瓣宽度 |
|---|---|---|---|
| 5.3cm | 3.7cm | 1.5cm | 0.2cm |
| 7cm | 3.2cm | 4.7cm | 1.4cm |
| 6.3cm | 3.3cm | 6cm | 2.5cm |
| … | … | … | … |

- 通过机器学习的相关方法，可以根据历史的经验数据自动将具有类似特征的对象聚集在一起
  - 聚类问题：历史数据没有标签，发现数据之间的相似性，将类似数据聚集在一起

# 无监督学习不需要标签

- 找出样本数据之间的关联、相似性、重要程度等
- 发现数据中的隐含结构
  - 如：聚类(Clustering)：将相似的样本划分为不同的组
- 经常与有监督学习结合使用，用于为有监督学习的样本数据自动打标签

例：请将同类型的动物圈在一起



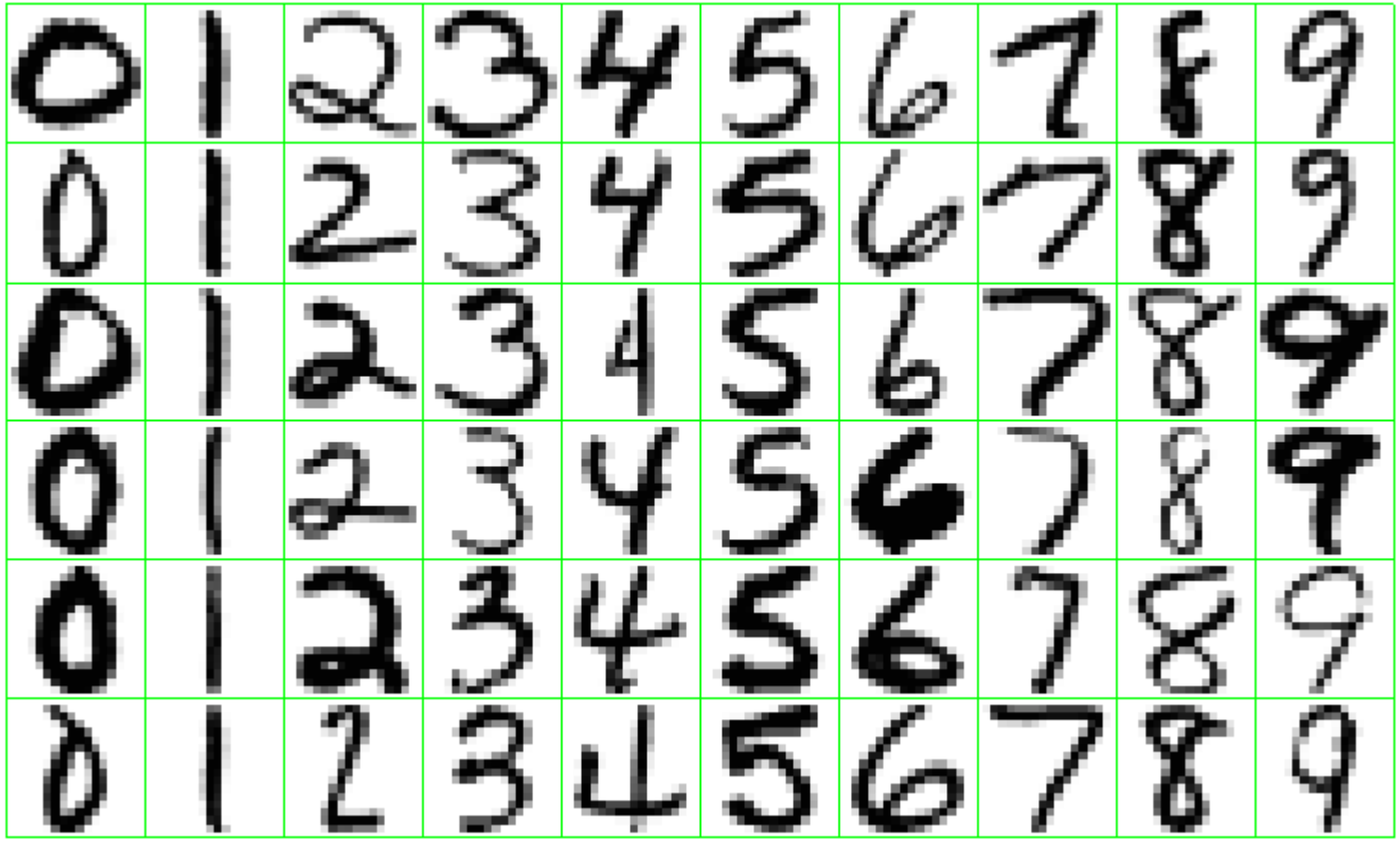
聚类的偏好取决于特征

# 关于特征

- 前面的例子都是建立在已经提取了样本特征的基础之上的
- 但是，哪些特征是真正对分类有贡献的？怎么把它们找到呢？
  - 传统机器学习技术的一大难点
  - 特征工程解决这一问题
    - 通常需要领域专家参与
    - 同时采用统计学的分析方法：方差选择、卡方检验、相关系数、互信息...
  - 这类方法只适用于结构化数据
  - 非结构化数据很难确定特征
    - 图像
    - 声音
    - 自然语言...

# 例：手写数字识别

- 此示例中的数据来自美国邮政信封上的手写邮政编码
- 每个数字图像都是从五位数邮政编码图像中分割得到



*Examples of handwritten digits from U.S. postal envelopes*

# 手写数字识别的数据举例

- 每个数字图像是16×16灰度图
- 每个像素范围：0到255

手写数字1对应的矩阵

图像

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 255 | 253 | 255 | 253 | 255 | 251 | 255 | 255 | 255 | 253 | 253 | 255 | 255 | 253 | 255 | 255 |
| 253 | 255 | 252 | 255 | 254 | 255 | 255 | 251 | 254 | 255 | 253 | 255 | 254 | 255 | 254 | 255 |
| 254 | 255 | 255 | 252 | 255 | 251 | 254 | 255 | 251 | 204 | 255 | 251 | 255 | 252 | 255 | 255 |
| 255 | 255 | 254 | 255 | 254 | 255 | 253 | 255 | 204 | 255 | 254 | 255 | 252 | 255 | 255 | 253 |
| 255 | 254 | 255 | 255 | 253 | 255 | 255 | 249 | 212 | 248 | 255 | 255 | 253 | 254 | 255 | 255 |
| 254 | 254 | 255 | 255 | 252 | 254 | 251 | 187 | 254 | 255 | 255 | 255 | 255 | 255 | 255 | 251 |
| 254 | 252 | 255 | 254 | 255 | 255 | 198 | 255 | 254 | 255 | 251 | 255 | 254 | 254 | 255 | 255 |
| 255 | 255 | 253 | 255 | 255 | 254 | 211 | 254 | 254 | 255 | 254 | 255 | 255 | 255 | 255 | 253 |
| 255 | 255 | 253 | 255 | 255 | 188 | 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 252 | 255 | 251 | 251 | 211 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 254 | 255 | 251 | 255 | 212 | 248 | 252 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 254 | 255 | 255 | 255 | 189 | 255 | 255 | 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 254 | 255 | 255 | 253 | 200 | 255 | 255 | 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 246 | 243 | 253 | 255 | 254 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 251 | 255 | 254 | 255 | 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 255 | 254 | 252 | 255 | 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |

1

标签

如何提取特征？

# 一种朴素的方法——模板匹配

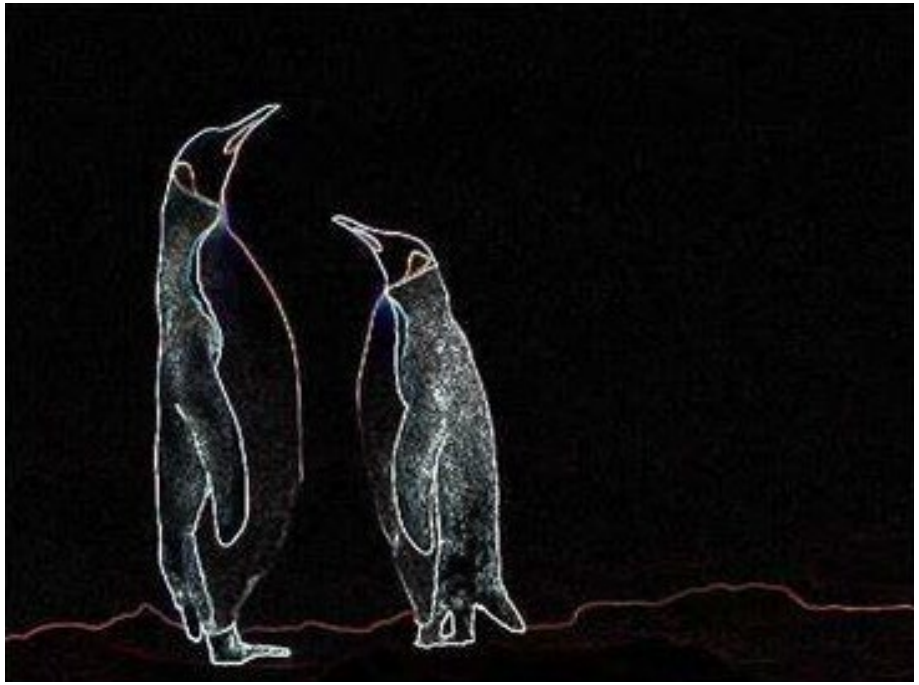
- 事先建立各种目标和场景的模板——目标对象的一个或多个二值或灰度图
  - 也可以是由提取的特征构成的特征向量，如图像的轮廓、灰度、纹理和边缘
- 识别时将模板图像在图像中平移
- 计算模板和图像的相似度
- 找到相似度最大的区域作为识别结果

模板

原图

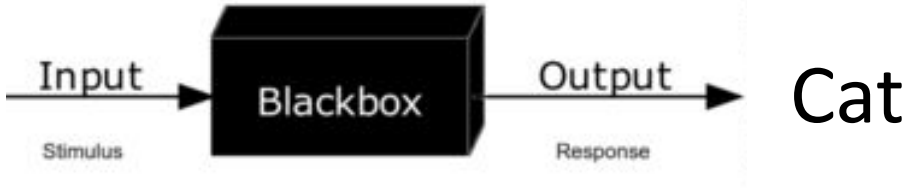
# 常用图像特征举例

- 颜色
- 形状（边缘、角点…）
- 纹理
- 空间…

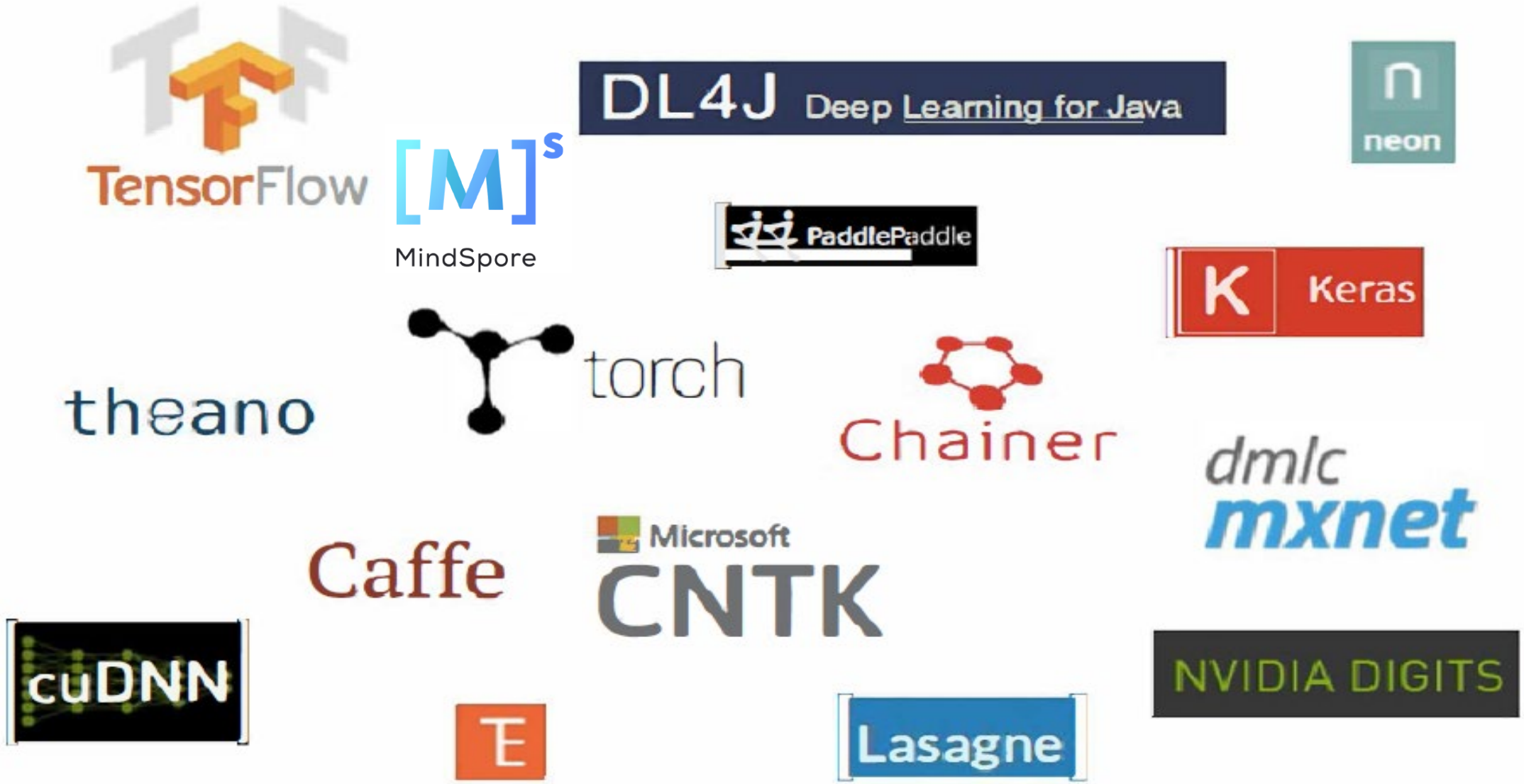


# 深度学习与特征提取

- 深度神经网络
  - 目前网络深度已经超过千层
- 主要应用领域
  - 图像识别
    - 手写数字识别
    - 人脸识别
    - 猫识别
  - 语音识别
    - 讯飞输入法
- 最大的优势
  - 建立了端到端的识别过程
  - 不需要人工提取特征



# 深度学习框架


TensorFlow
MindSpore
DL4J Deep Learning for Java
neon
PaddlePaddle
Keras
theano
torch
Chainer
dmlc
mxnet
Caffe
Microsoft
CNTK
cuDNN
NVIDIA DIGITS
Lasagne

# 国产AI框架：华为AI

**行业应用**

能源、金融、公共、交通、运营商、制造、教育等更多行业应用

**应用使能**

ModelArts　　HiAi Service　　第三方平台

**MindX 使能应用**

MindX DL
深度学习使能

MindX Edge
智能边缘使能

ModelZoo
优选模型库

MindX SDK
行业SDK

**AI框架**

昇思MindSpore

匹配昇腾AI处理器算力的全场景深度学习框架

**AI异构计算框架**

CANN

统一异构计算架构，释放昇腾硬件澎湃算力

**Atlas系列硬件**

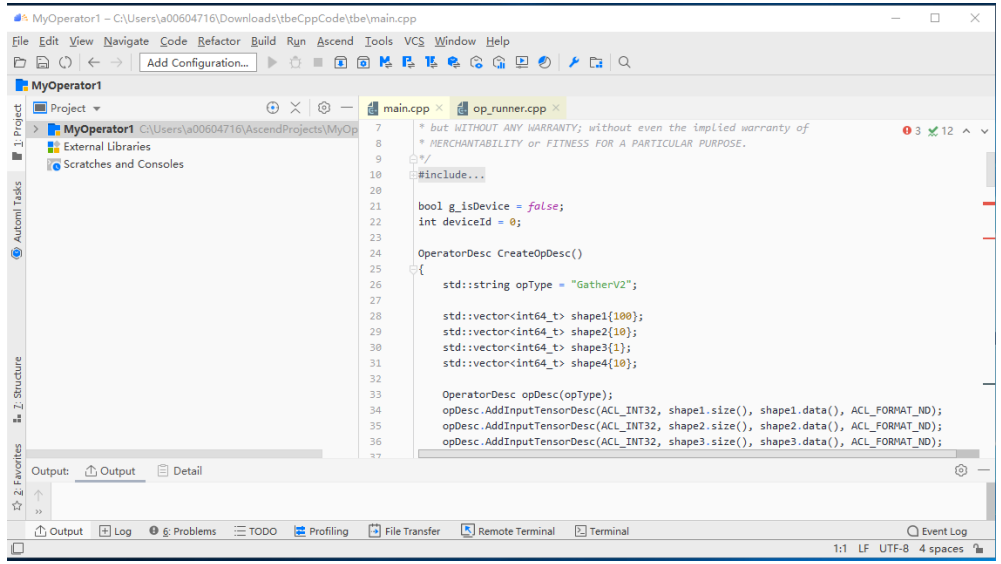

全流程开发工具链 MindStudio

管理运维工具 FusionDirector/Smart

昇腾社区 ascend.huawei.com

# 一站式开发环境MindStudio

MindStudio是一套基于IntelliJ框架的开发工具平台。提供了应用开发、调试、模型转换功能，同时还提供了网络移植、优化和分析等功能




辅助功能

- 训练脚本转换
- 模型转换
- 精度比对
- Profiling性能分析
- System Profiling工具
- AI Core Error分析工具

**应用开发**

- 基于MindX SDK开发应用
- 基于新工程开发应用
- 应用工程调试

**模型开发**

- 查询模型
- 模型训练
- 模型可视化

**算子开发**

- 查询算子
- 开发流程
- 算子分析
- 工程创建
- TBE算子开发（TensorFlow）
- TBE算子开发（PyTorch）
- TBE算子开发（MindSpore）
- AI CPU算子开发（TensorFlow）

# 开源AI框架MindSpore

## 简单的开发体验

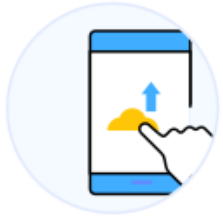
帮助开发者实现网络自动切分，只需串行表达就能实现并行训练，降低门槛，简化开发流程。

## 灵活的调试模式

具备训练过程静态执行和动态调试能力，开发者通过变更一行代码即可切换模式，快速在线定位问题。

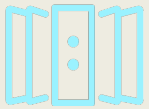
## 充分发挥硬件潜能

最佳匹配昇腾处理器，最大程度地发挥硬件能力，帮助开发者缩短训练时间，提升推理性能。



## 全场景快速部署

支持云、边缘和手机上的快速部署，实现更好的资源利用和隐私保护，让开发者专注于AI应用的创造。



## 全自动并行

静态图自动混合并行
训练**性能提升40%**

动态图优化
性能**超越业界60%**



## 全场景协同

云端分布式推理

边缘AI加速

超轻量IoT设备推理



## 全流程极简

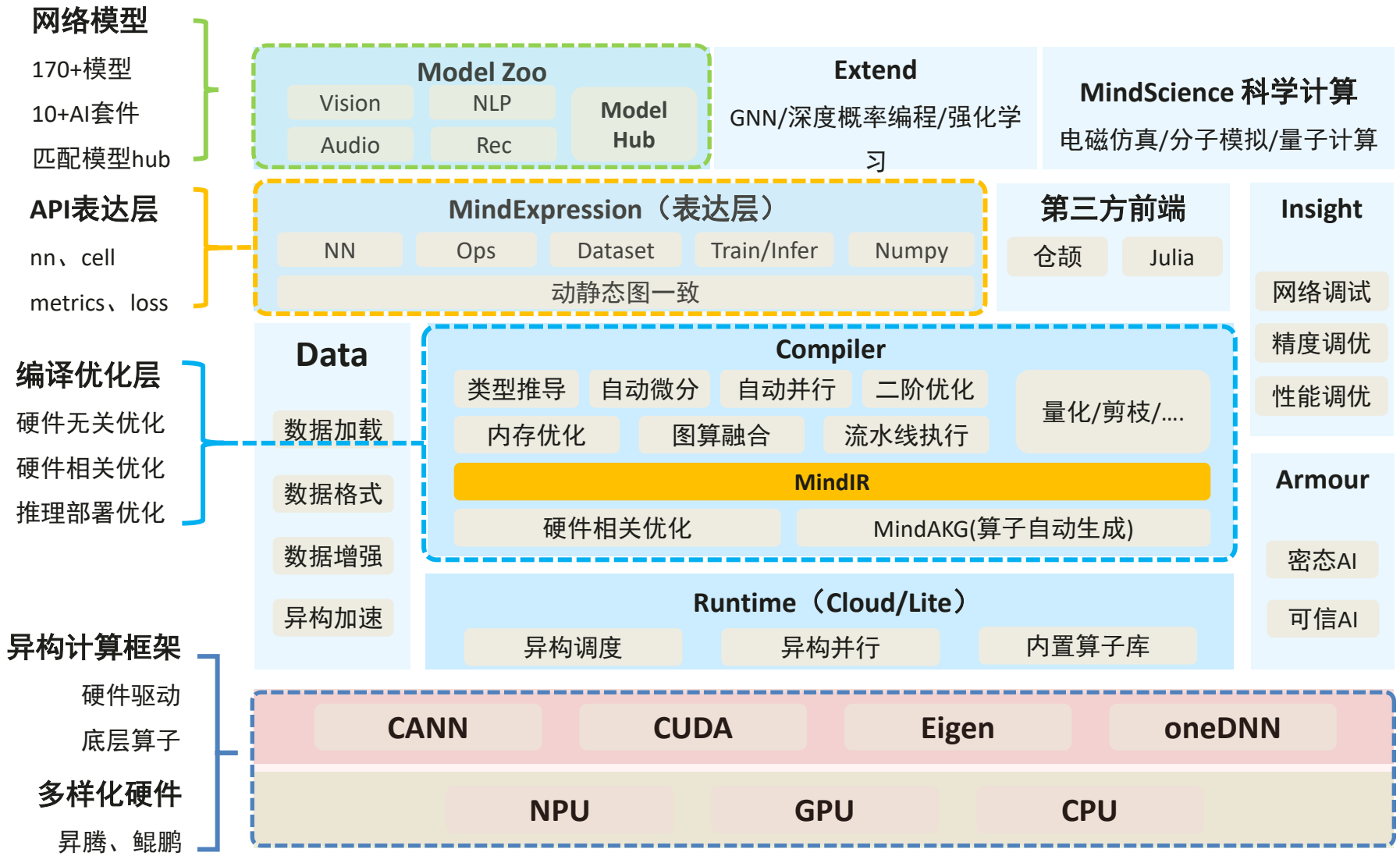
第三方框架转换工具
业务快速迁移

### 开发者生态

**51万+**
下载量

**2300+**
社区贡献者

# MindSpore 框架架构图



# 全场景AI计算框架MindSpore

## MindSpore 架构特点：

用户态易用；
运行态高效；
部署态灵活；

## MindSpore 特性：

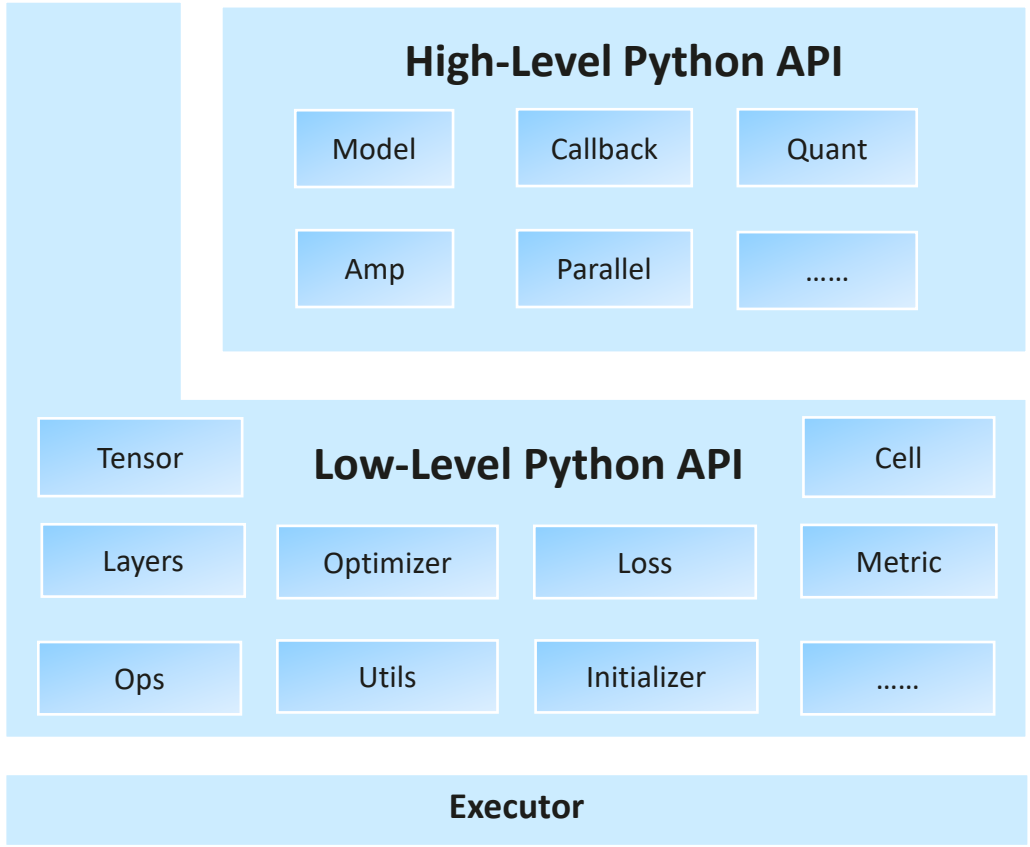
**自动并行：**通过自动并行机制、数据pipeline处理等手段降低超大模型训练门槛。

**AI+科学计算：**支持AI+科学计算的高阶/高纬、多范式编程。

**通用计算+DSA：**通过图算融合对性能进行优化，自动算子生成技术简化异构（ DSA ）编程，发挥多样性算力的性能。

**端边云统一的可信架构：**解决企业级部署和可信的挑战。

# MindExpress子系统




**设计目标：**

- 两层用户API设计（包括High-Level与Low-Level），支撑用户进行网络构建、整图执行、子图执行以及单算子执行
- 向用户提供统一的模型训练、推理和导出等接口，满足端、边、云等不同场景
- 动态图和静态图统一的编码方式
- 单机和分布式训练统一的编码方式

**功能模块：**

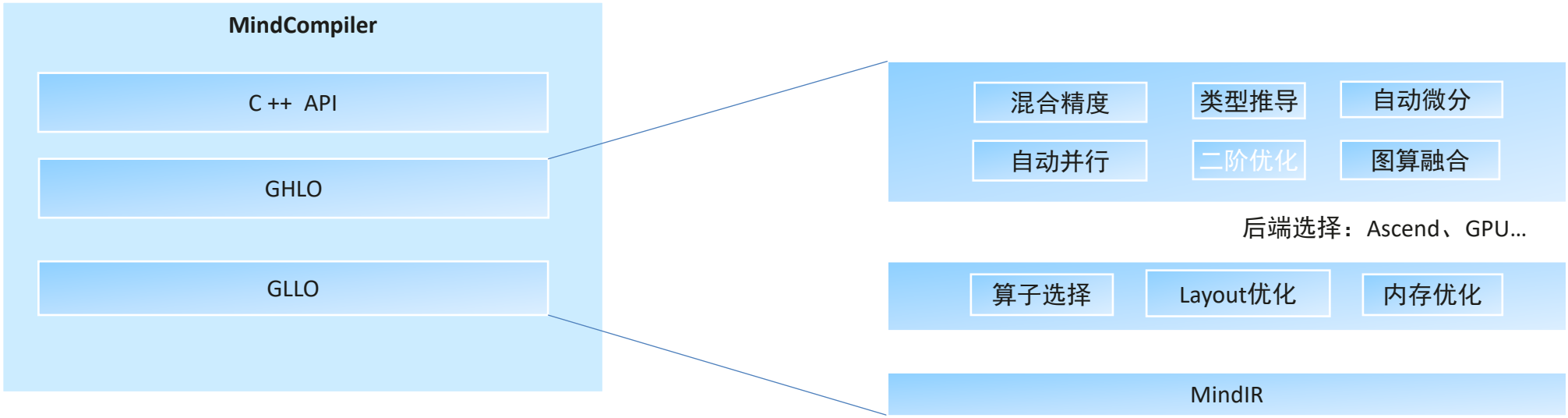
- High-Level API提供训练推理的管理接口、Callback、量化、混合精度、并行等控制接口，易于用户实现整网流程的控制
- Low-Level API提供基础的Tensor、Cell、NN-Layers、优化器、初始化等，易于用户灵活构建网络和控制执行流程
- Executor提供计算的执行控制，与MindSpore backend交互

# MindCompiler子系统

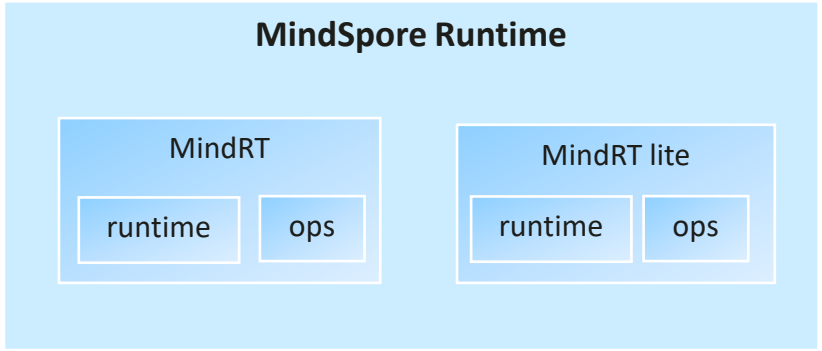
MindCompiler提供面向Mind IR的图级即时编译能力：

1）Graph High Level Optimization (GHLO)面向应用，进行偏前端的优化和功能，如类型推倒、自动微分、二阶优化、自动并行等

2） Graph Low Level Optimization (GLLO)面向硬件，进行偏底层的优化，如算子融合、layout优化、冗余消除、内存优化等



# MindSpore Runtime子系统




## 统一的运行时（RunTime）系统：

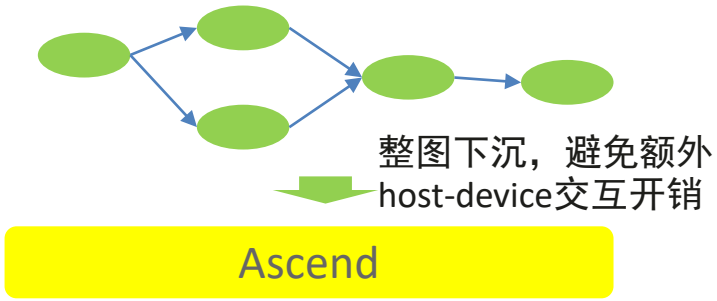
1、支持端、云多种设备形态要求

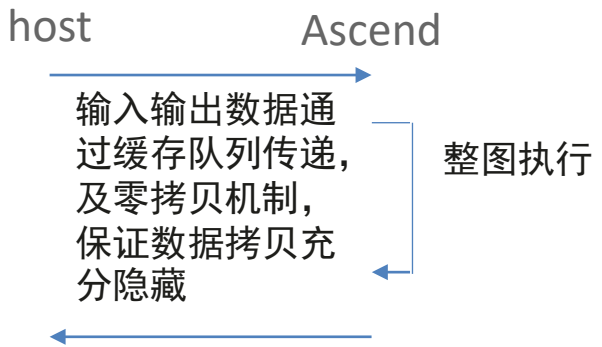
2、支持多种硬件设置的调度管理，如Ascend、GPU、CPU

3、内存池化管理，高效内存复用

4、算子异步、异构执行，多流并发

特色技术点



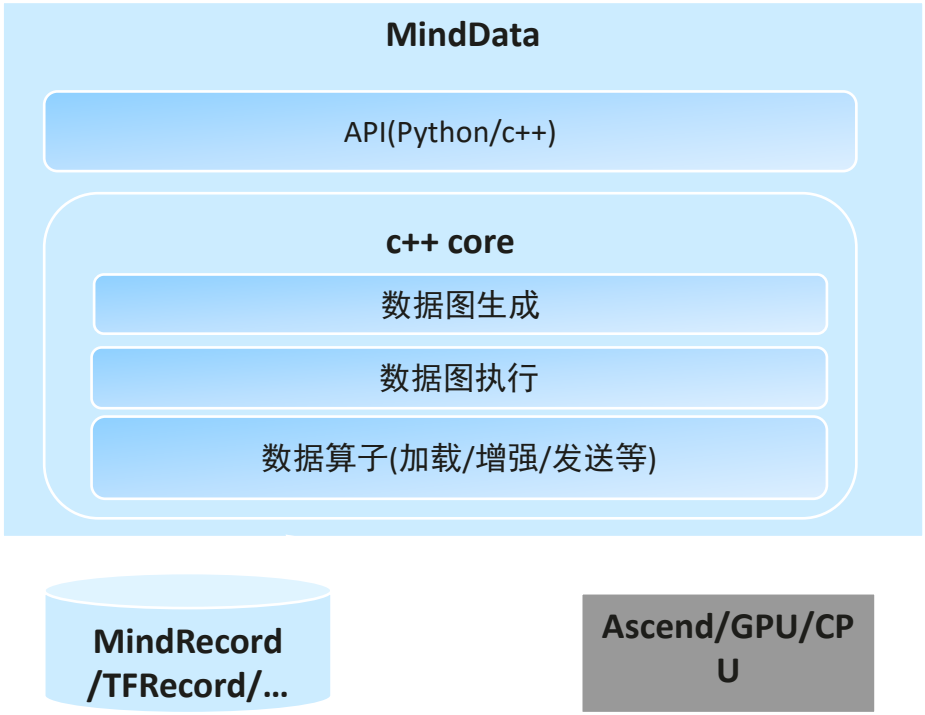



# MindData子系统

**MindData负责高效执行训练数据处理pipeline，与计算形成流水，数据及时导入训练。**

加载　　shuffle　　map　　batch　　repeat

典型训练数据处理pipeline



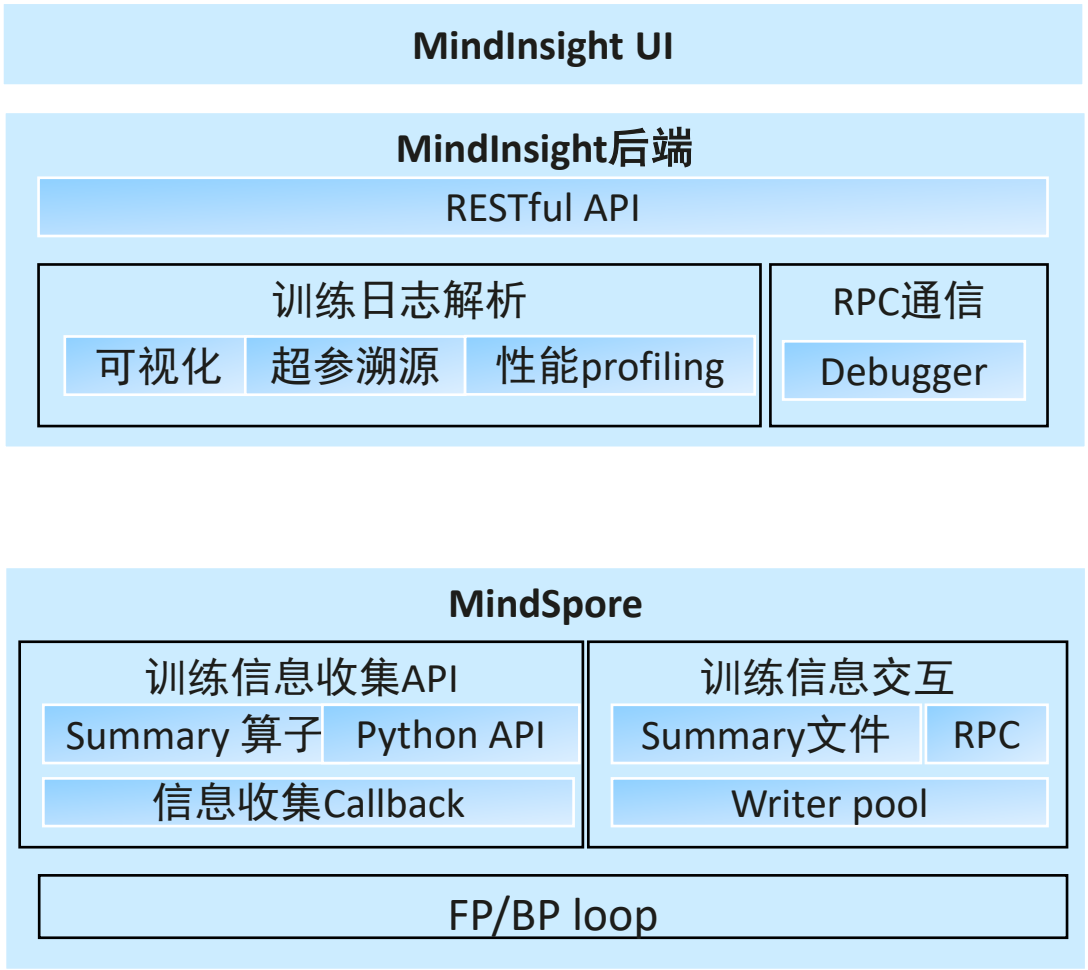

**关键功能：**

- 流水线+并行方式执行，提高数据处理吞吐量；
- 丰富的数据算子；
- 用户自定义python算子，以及用户灵活定制pipeline (数据加载、采样、增强等)；
- 异构硬件加速(Ascend/GPU/CPU)；
- MindRecord：自带元数据、聚合存储；

**运行流程：**

1. 数据图生成：根据用户的Python API调用，生成数据图；
2. 数据图执行：Pipeline并行执行数据图中的数据算子完成数据集加载、shuffle、数据增强、batch等处理；
3. 数据导入Device：处理后数据导入Device训练；

# MindInsight子系统




**MindInsight是MindSpore的调试调优子系统，提供训练过程可视化、模型溯源、debugger和性能profiling功能。**

**关键功能：**

- 易用的API接口，在训练过程中，用户可以方便的收集训练过程指标，包括计算图、标量数据(loss/accuracy…）、直方图数据(梯度/权重…)、性能数据等，并通过Web UI界面进行展示。
- 通过收集训练的超参，数据集、数据增强信息实现模型溯源，并可在多次训练间进行对比

**运行流程：**

1. 训练信息收集：用户可通过callback接口，收集常用训练指标。用户也可以按需要收集自定义信息，如通过summary算子收集计算图中信息，通过Python接口收集Python层信息。
2. 训练日志生成：用户在训练过程中收集到的过程信息，最终会生成训练日志。
3. 训练信息展示：MindInsight通过打开并解析训练日志，以图形化方式为用户展示训练过程信息。

# 学习推荐

官方网站

MindSpore

https://www.mindspore.cn

代码托管平台

gitee

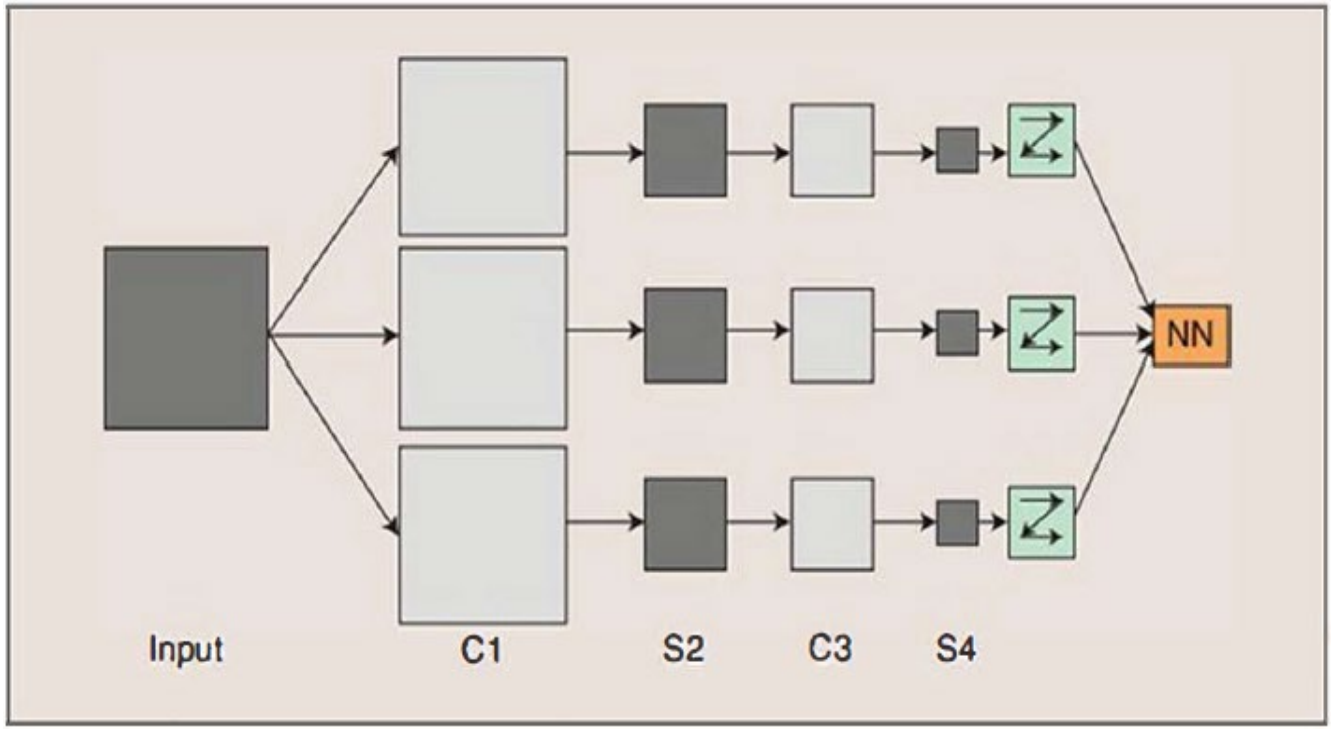
https://gitee.com/mindspore

- MindSpore社区：https://gitee.com/mindspore/community
- 优秀开发者及布道师招募：https://www.mindspore.cn/evang
- 高校开发者微信群：添加微信mindspore0328，备注“开发者”
- 官方交流QQ群：871543426
- 官方微信公众号：MindSpore

小助手mindspore0328

# 最常用的深度网络——卷积神经网络(CNN)

- 图像识别中最常用的模型
- 特点
  - 结构简单
  - 训练参数少
  - 高适应性
- 避免了复杂的图像预处理过程
  - 不需人工提取特征
  - 只需直接输入原始图像
- 基本组成部分
  - 卷积层（Convolution Layers）
  - 池化层（Pooling Layers）
  - 全连接层（Fully connected Layers）



# 机器学习中的变量类型

- 机器学习任务中有各种不同的变量
  - 房价预测的输出
    - 定量数据（quantitative data）
    - 值有大有小，值的大小关系与实际问题中的关系一致，即y1>y2，则房价1>房价2
  - 水果分拣的输出
    - 定性数据（qualitative data）
    - 值被假定限制在一个有限集中G={apple, orange}
  - 手写数字识别的输出
    - 10个不同的数字类别G={0，1，2，3，4，5，6，7，8，9}，是定性的
    - 不同值之间没有顺序或大小关系

# 机器学习中的变量类型

- 三种类型
  - 分类变量（*categorical variables*）
    - 定性变量
  - 数值变量（*numerical variables* ）
    - 定量变量
  - 序数变量（*ordinal variables* ， *ordered categorical* ）
    - 定性与定量的混合
    - 如： small(0), medium(1) and large(2)
    - 数据是Categorical 性质的，但是数值本身的大小也是有意义的
    - 不过度量标准与数值大小没有联系，如中小之间的差异不必与大中之间的差异相同，虽然1-0=2-1

# 机器学习中的变量类型

- 定性变量的量化方法
  - 通常以数字表示
  - 最简单的情况
    - 只有两个类别，如“成功”或“失败”，“幸存”或“死亡”
    - 通常由单个二进制数字或位表示为0或1，也可以用 -1或1
  - K个类别的情况（K>2）
    - 最常用的方法——使用虚拟变量 ( *Dummy Variables*) 又称虚设变量、名义变量或哑变量
    - 每一个具有K个类别的分类变量用一个K维二进制向量表示，每个向量只有一维为“1”，代表其属于对应的类别
    - 如手写数字识别，（0，0，1，0，0，0，0，0，0，0）表示该输出是“2”
    - 该方法又称为“独热”（one-hot）编码

# 机器学习的一般思想

- 一般来说，在进行机器学习任务时，使用的每一个算法都有一个目标函数，算法便是对这个目标函数进行优化
  - 目标函数包含衡量预测值与真实值的距离的部分——损失函数/经验风险
  - 训练的目的是缩小这个差距，让预测值与真实值尽可能一致
- 不同的机器学习算法使用的目标函数不同
- 从大量数据中自动搜索学习使得目标函数最小化的参数配置
- 机器学习由参数驱动，大量的时间和精力都花费在参数调整上
- 学习的效果主要取决于所能获得的训练数据

# 机器学习的一般过程

- 数据采集和整理，分析特征
  - 有什么数据非常重要
- 数据集分为训练集和测试集
- 制定机器学习策略
- 用训练集训练机器学习模型，不断调整参数，以期结构风险最小，达到最佳分类效果
- 交叉验证，即用未参与训练的测试集数据来验证模型的效果，根据结果调整模型
- 投入使用，输入待分类数据，得到结果

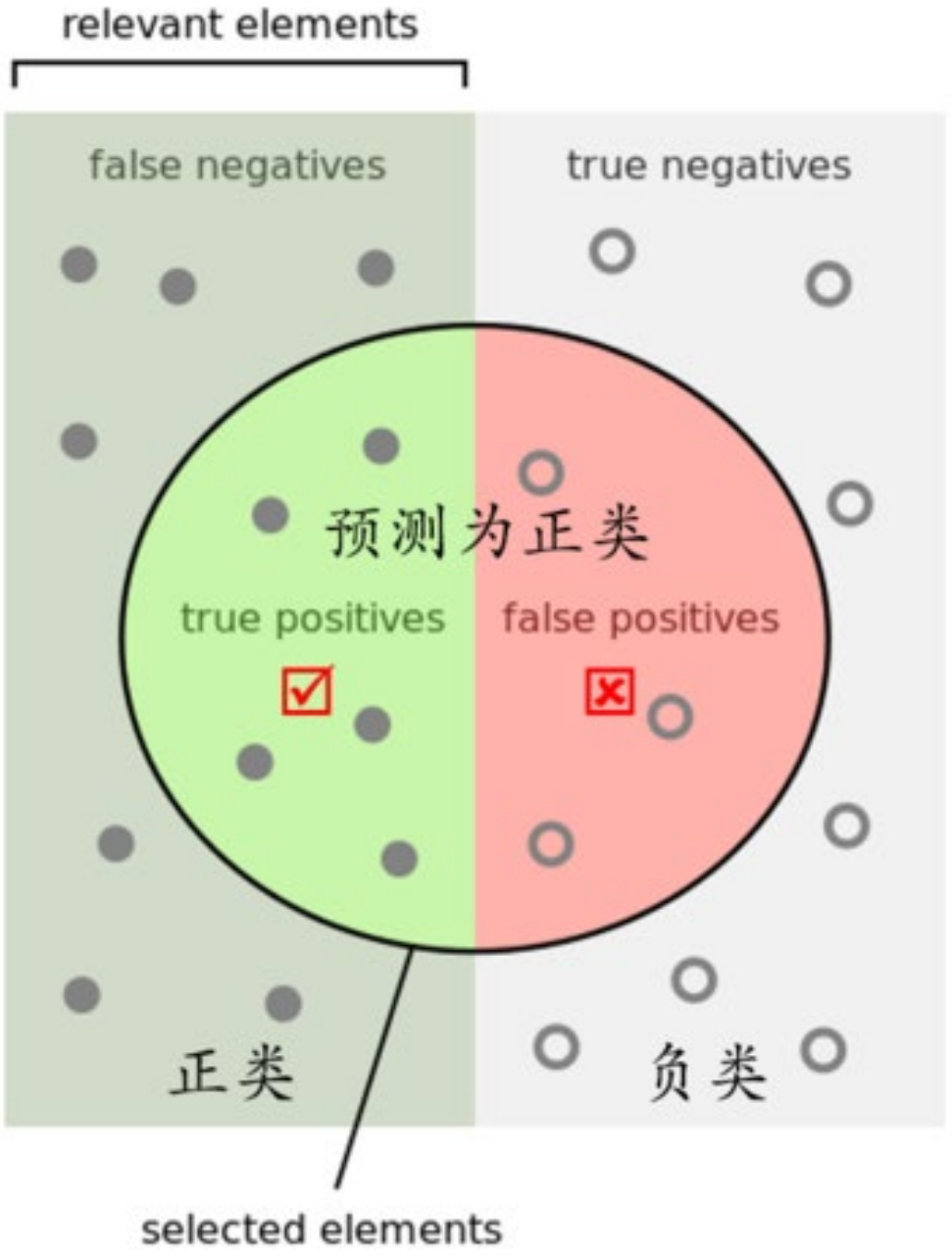
# 交叉验证（Cross validation）

- 基本思想：把给定的数据切分为训练集和测试集，反复进行训练、测试及模型选择
- 简单验证（Hold-Out Method）
  - 随机地将样本数据分为训练集和测试集
  - 不够有说服力
- K折交叉验证（K-fold cross validation）
  - 随机将样本分为K个互不相交的大小相同的子集
  - 用K-1个子集的数据训练模型，剩下的子集做测试
  - 对K个子集重复上述过程，选出K次评测中平均测试误差最小的模型
- 留一法（Leave-One-Out）
  - K等于数据集中数据的个数，每次只用一个作测试集，剩下的全部作为训练集
  - 得出的结果与训练整个测试集的期望值最为接近
  - 成本庞大

# 机器学习的评价指标

- 基本指标（主要用于二分类）
  - *TP*（*true positive*）
    - 真正，实际类别为正，预测类别为正
  - *FP*（*false positive*）
    - 假正，其为实际类别为负，预测类别为正
  - *TN* (*true negative*)
    - 真负，实际类别为负，预测类别为负
  - *FN* (*false negative*)
    - 假负，实际类别为正，预测类别为负
- 混淆矩阵

| | |
|---|---|
| *TP* | *FP* |
| *FN* | *TN* |



# 其他常用评价指标

- 二分类问题的常用评价指标
  - 精确率（*precision*,查准率）

$$precision = \frac{TP}{TP+FP}$$

  - 召回率（*recall*,查全率），灵敏度（*sensitivity*），真正类率（*TPR*）

$$recall = \frac{TP}{TP+FN} = \frac{TP}{P} = sensitivity$$

  - 特异度（*specificity*），真负类率（*TNR*）

$$specificity = \frac{TN}{TN+FP} = \frac{TN}{N}$$

    - 假负类率（*FNR*）

$$FNR = 1 - specificity$$

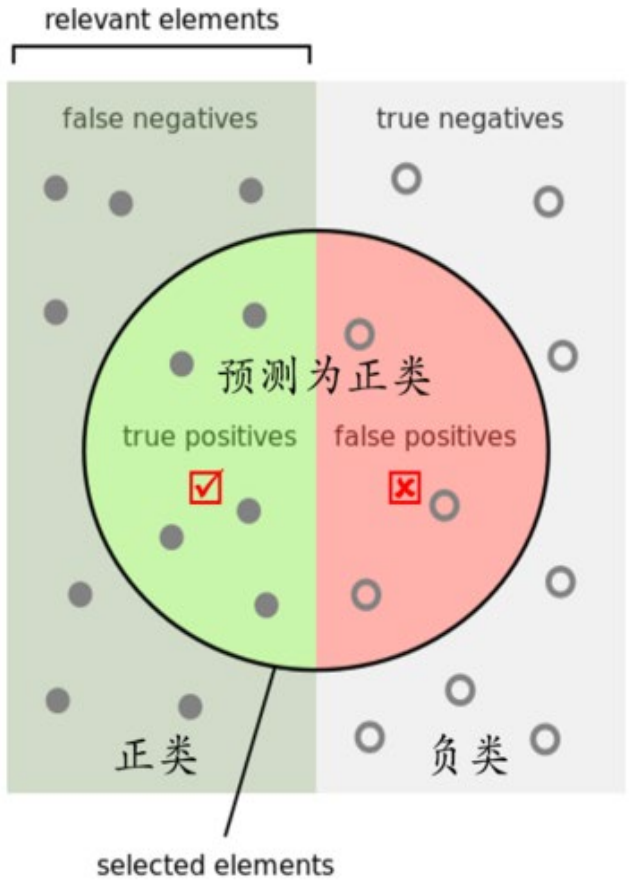
  - $F_1$值
    - 精确率与召回率的调和均值
    - 精确率与召回率都高时，$F_1$值也高

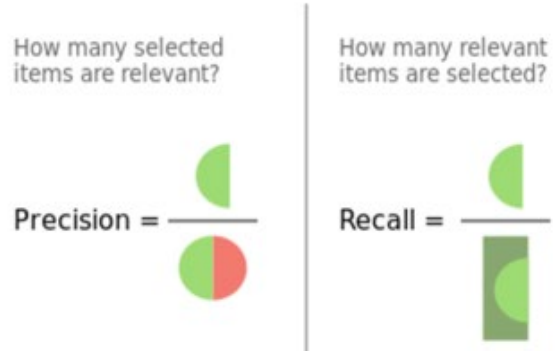
$$\frac{2}{F_1} = \frac{1}{precision} + \frac{1}{recall}$$

$$F_1 = \frac{2TP}{2TP + FP + FN}$$

  - 准确率（*accuracy*）

$$accuracy = \frac{TP+TN}{P+N}$$



# 两类错误率

- 医学领域的疾病诊断是典型的二分类问题
  - 通常用阳性(*positive*, 有症状/异常)和阴性(*negative*, 无症状/正常)来代表两类
- 混淆矩阵中存在两类错误

| 决策 | 状态 | |
|---|---|---|
| | 阳性 | 阴性 |
| 阳性 | *TP* | *FP* |
| 阴性 | *FN* | *TN* |

混淆矩阵，有两种错误分类

**评价指标：**

1. 灵敏度/敏感度(*sensitivity*)

$$S_n = \frac{TP}{TP+FN} = \frac{TP}{P}$$

2. 特异度(*specificity*)

$$S_p = \frac{TN}{TN+FP} = \frac{TN}{N}$$

**两类错误率：**

1. 第一类错误率(*Type* I *Error*，$\alpha$)

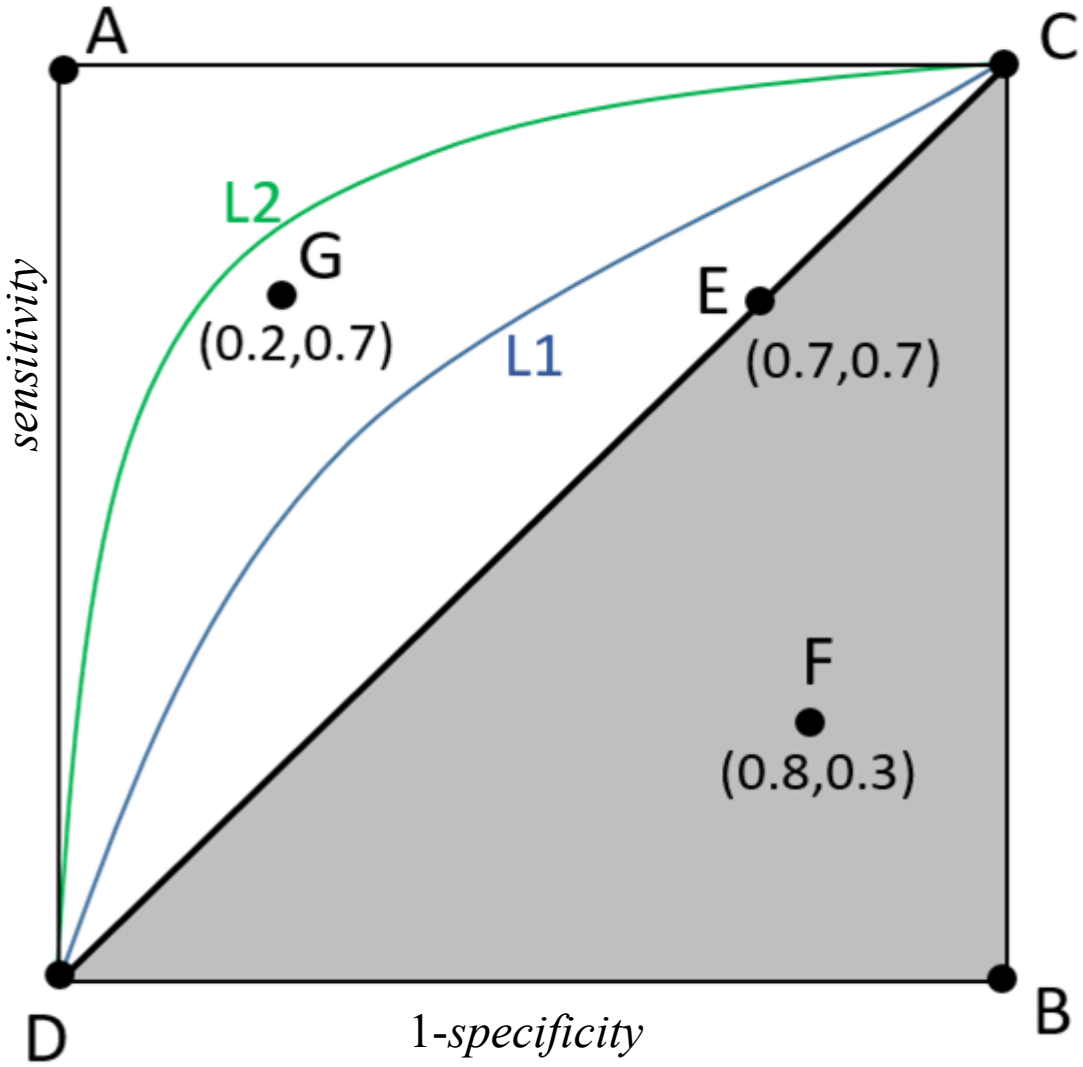
假阳率 $\alpha = 1 - S_p$

2. 第二类错误率(*Type* II *Error*，$\beta$)

假阴率 $\beta = 1 - S_n$

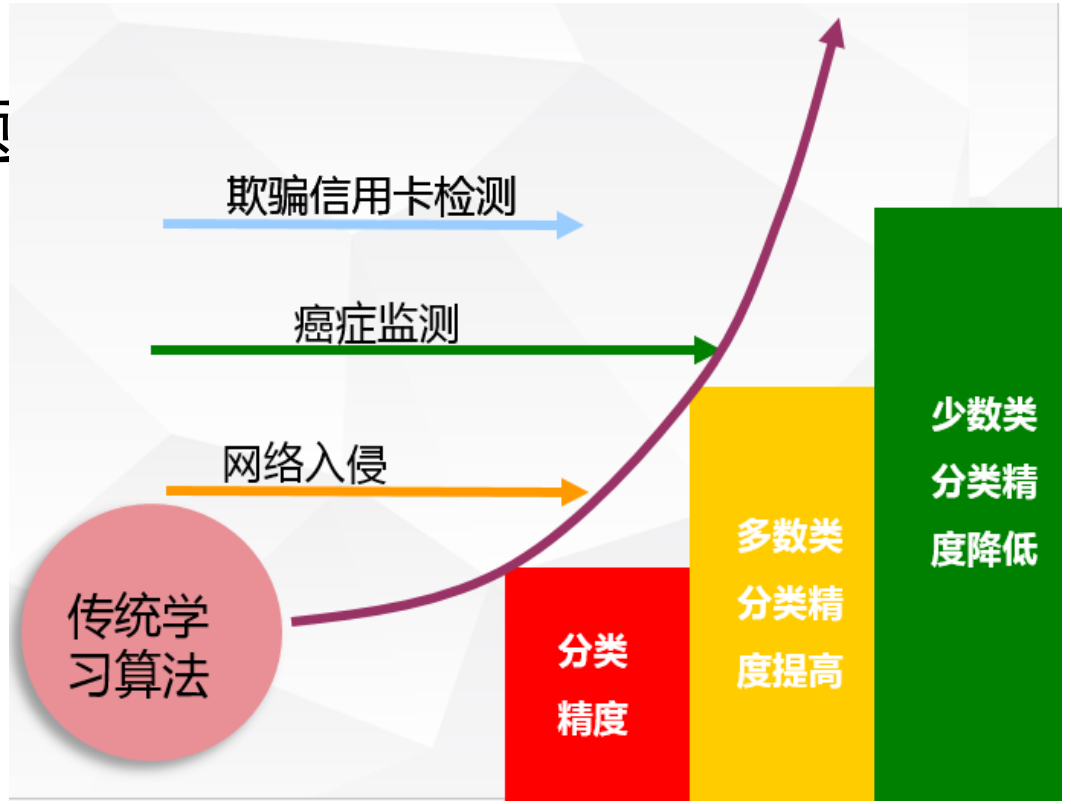
# 其他常用评价指标

- ROC曲线
  - *Receiver Operating Characteristic Curve*，接受者操作特性曲线
  - 以1-*specificity*和*recall*(*sensitivity*)为*x*,*y*轴的曲线
  - ROC曲线越靠近A点性能越好，越靠近B点性能越差
  - C-D线上的点说明算法性能和随机猜测是一样的
- AUC
  - *Area Under the ROC Curve*， ROC 曲线下面积
  - AUC越大说明性能越好



# 类不平衡性与评价指标的选择

- 针对具体问题，有时需合理选择或设计合适的评价指标
- 其中一类突出的问题：类别数据不平衡问题
  - 假设观测样本分两类
    - N——反例集合
    - P——正例集合
  - 若其中一类样本的数量远大于另一类，如|P|>>|N|，则很多常规指标无法合理评价预测模型的优劣
- 类不平衡性在很多问题中是常见现象
  - 信用卡欺诈，疾病诊断，入侵检测…
  - 常规算法倾向于将不能明确分类的样本归为大类以减少误差



# 训练数据的不平衡可能带来的问题

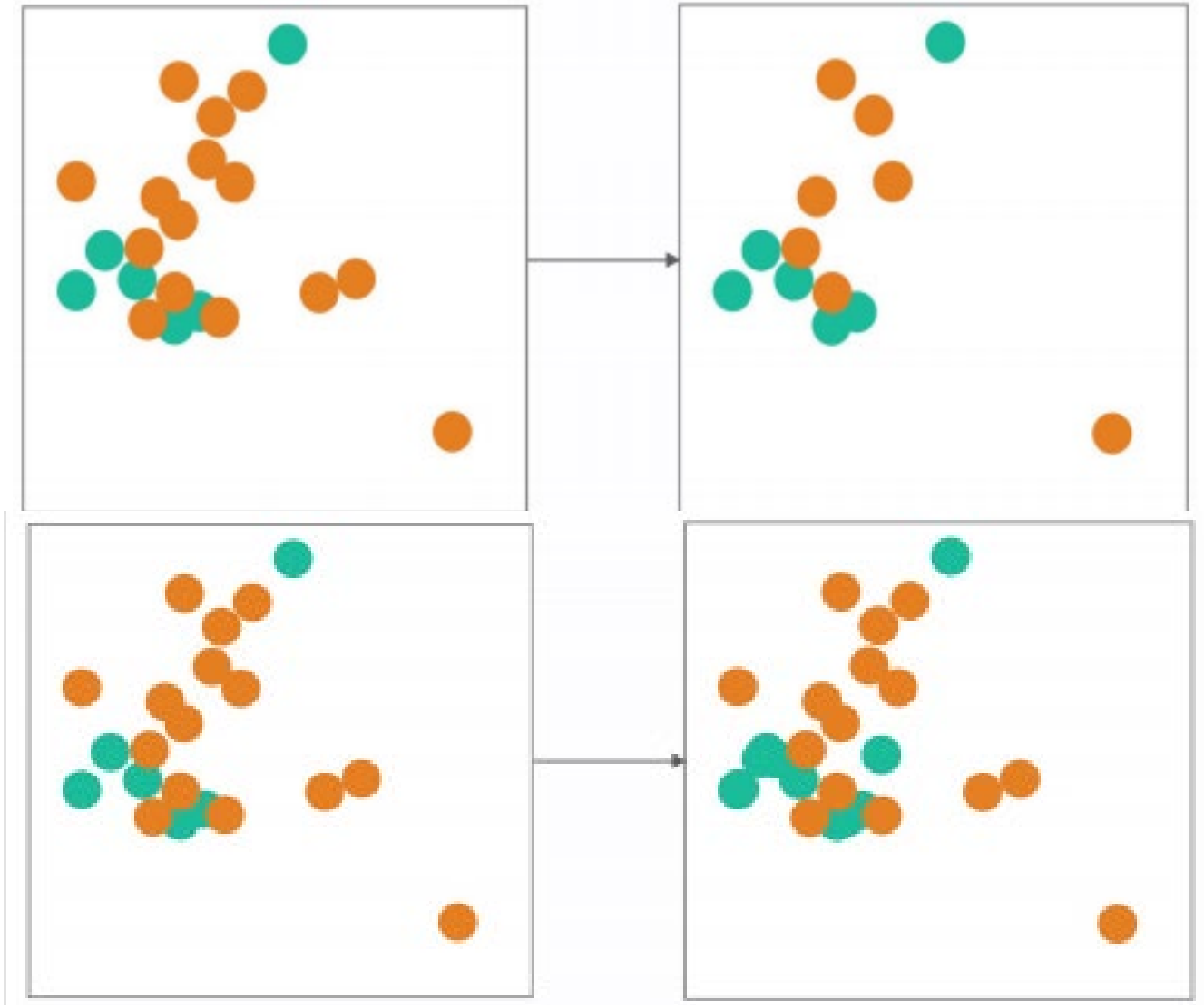

## 人工智能开发应用的道德原则

# 思考题

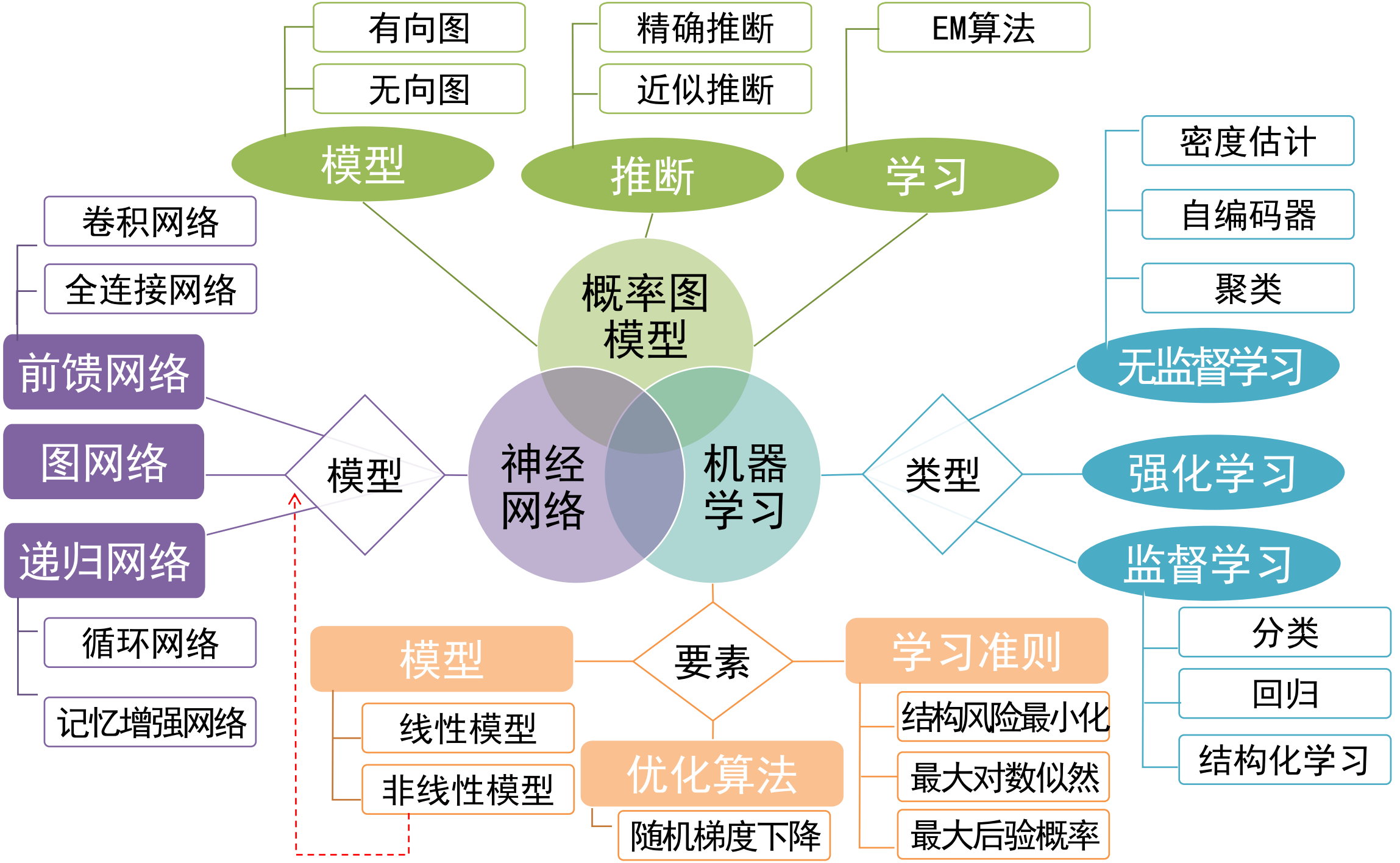
- 若有观测样本100个，其中正例90个，反例10个，现有一预测模型
  - 若该模型将所有样本均识别为正例，则分别计算以下评价指标
    - TP，TN
    - *recall*, *specificity*
    - *accuracy*，*precision*
  - 若该模型能正确识别7个反例， 70个正例，则上述评价指标值分别是多少

# 不平衡数据的处理方法

- 采样层面
  - 下采样（欠采样）
    - 从多数类样本中随机删除一些样本与少数类样本一起作为训练数据集
  - 上采样（过采样）
    - 从少数类样本中多次有放回随机采样，与多数类样本组成训练集，少数类采样的数量要大于原数量



# 机器学习知识图谱


机器学习
神经网络
概率图模型
模型
卷积网络
全连接网络
前馈网络
图网络
递归网络
循环网络
记忆增强网络
有向图
无向图
推断
精确推断
近似推断
学习
EM算法
类型
无监督学习
密度估计
自编码器
聚类
强化学习
监督学习
分类
回归
结构化学习
要素
模型
线性模型
非线性模型
优化算法
随机梯度下降
学习准则
结构风险最小化
最大对数似然
最大后验概率

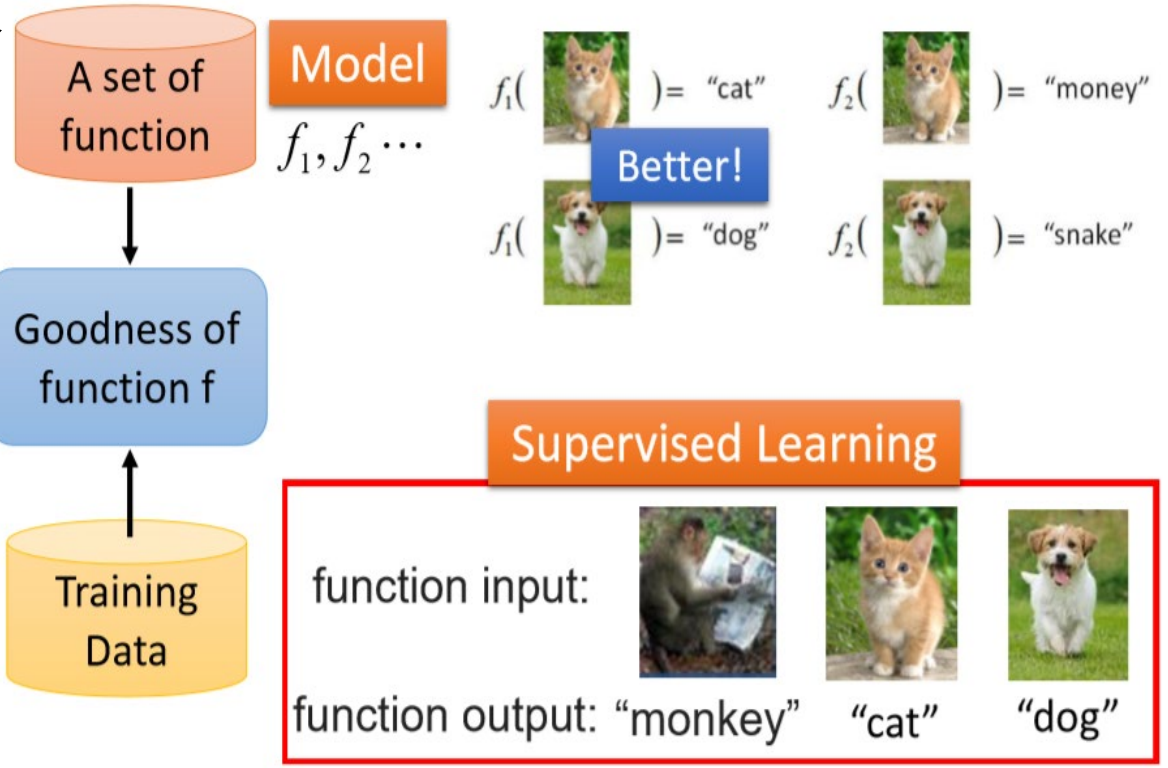
# 统计学习理论

- 传统统计学方法只有在学习样本的数目趋于无穷大时，学习性能才有理论上的保证
- 机器学习的基本问题
  - 根据给定的训练样本求对某系统输入输出之间依赖关系的估计，使它能够对未知输出作出尽可能准确的预测
- 问题的表示：根据n个独立同分布观测样本，假设其联合概率为$F(x,y)$

$$(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)$$

在一组函数$\{f(\boldsymbol{x},w)\}$中求一个最优的函数$f(\boldsymbol{x},w_0)$对依赖关系进行估计，使预测的期望风险最小

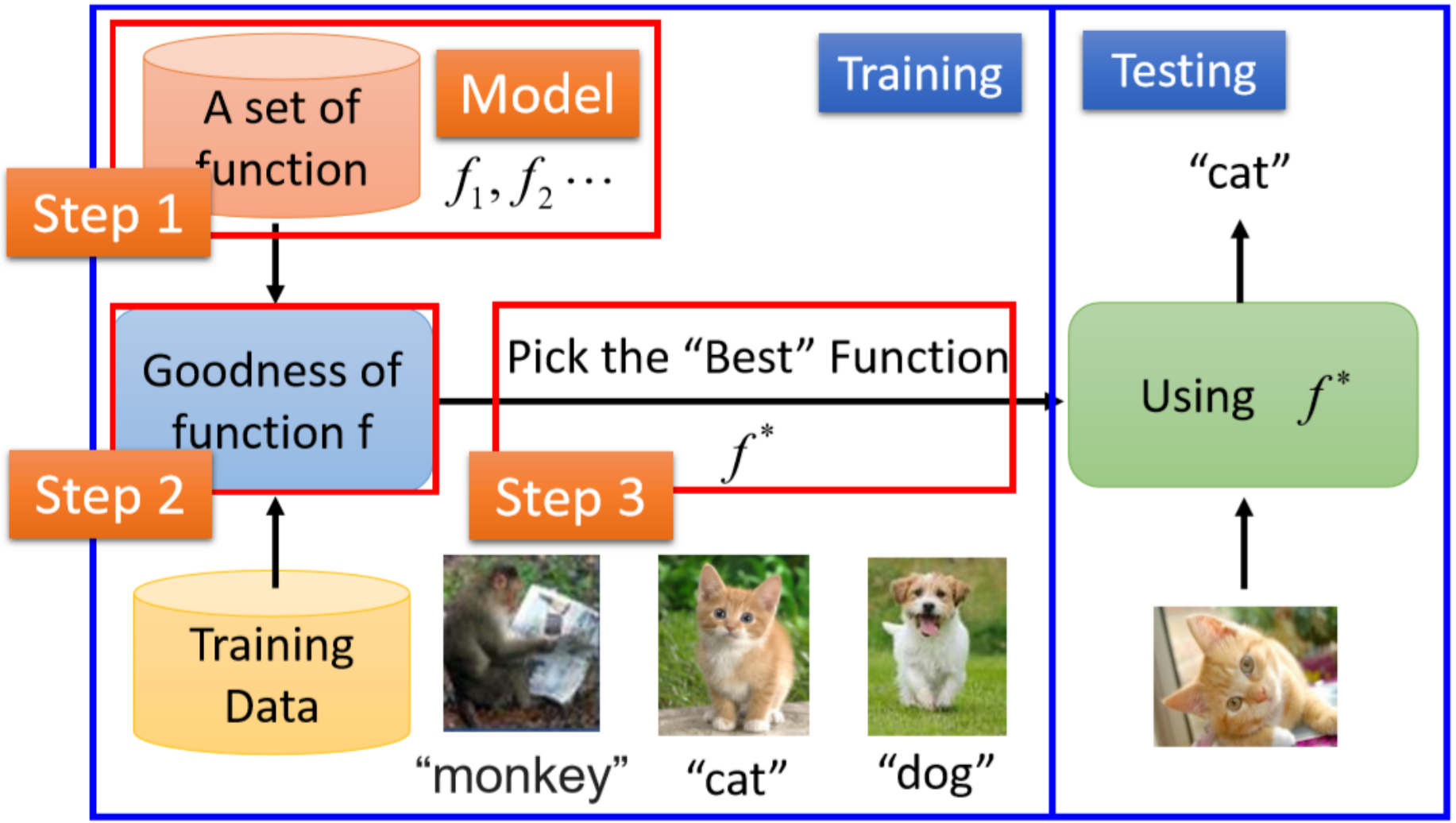
$$R(w)=\int L\big(y,f(x,w)\big)dF(x,y)$$



# 统计学习理论

Image Recognition:

$f($  $)=$ "cat"



# 统计学习理论

- 期望风险

损失函数

$$R(w) = \int L\big(y, f(x,w)\big) dF(x,y)$$

  - 对所有样本（包含已知和未知样本）的预测能力
  - 是全局概念
- 理想的模型（决策）函数应该是要使风险最小化，这依赖于联合概率$F(x,y)$的信息
- 真实的联合概率很难获得，只能利用样本数据的信息
  - 警惕“以偏概全”
- 期望风险无法直接计算和最小化

# 统计学习理论

- 根据大数定律，人们想到用算术平均代替数学期望

$$R_{emp}(w) = \frac{1}{n}\sum_{i=1}^{n} L\left(y_i, f(x_i, w)\right)$$

- 经验风险
  - 由已知的训练样本（即经验数据）确定
  - 是局部的
- 经验风险和期望风险之间的关系
  - 经验风险是局部的，基于训练集所有样本点损失函数最小化。经验风险是局部最优，是现实的可求的
  - 期望风险是全局的，基于所有样本点损失函数最小化。期望风险是全局最优，是理想化的不可求的

# 统计学习理论

- 损失函数
  - 针对具体样本，表示模型预测值与真实样本值之间的差距
  - 常用损失函数
    - 平方损失（最小二乘）

$$\min(\sum(f(WX_i) - y_i\ )^2)$$

    - 对数损失（最大似然）

$$\max\prod p(y_i|X_i, W) = max\sum log(p(y_i|X_i, W))$$

    - 交叉熵损失

$$-\sum_{c=1}^{C} y_i^c log(\hat{y}_i{}^c) \qquad (Softmax，C为类别数)$$

$$-(y_i log(\hat{y}_i) + (1 - y_i) log(1 - \hat{y}_i)) \qquad (Sigmoid)$$

# 统计学习理论

- 常用损失函数
  - Zero-one Loss

$$L(y_i, \hat{y}_i) = \begin{cases} 1 & y_i \neq \hat{y}_i \\ 0 & y_i = \hat{y}_i \end{cases}$$

  - Perceptron Loss

$$L(y_i, \hat{y}_i) = \begin{cases} 1 & |y_i - \hat{y}_i| > t \\ 0 & |y_i - \hat{y}_i| \leq t \end{cases}$$

$t$—超参

  - Hinge Loss（SVM）

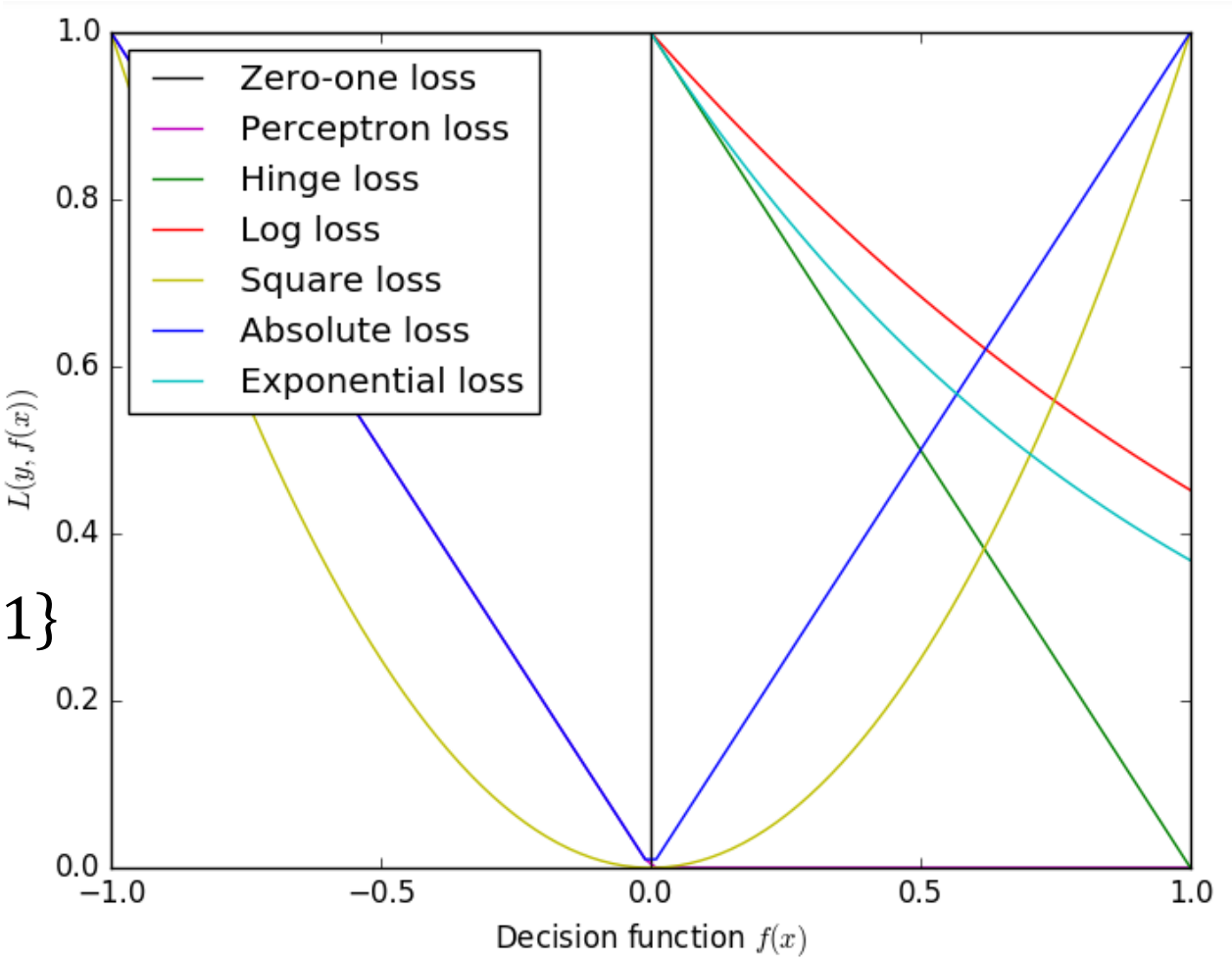
$$L(y_i, \hat{y}_i) = \max(0, 1 - y_i \cdot \hat{y}_i), \quad y_i \in \{-1, 1\}$$

  - Exponential Loss（Adaboost）

$$L(y_i, \hat{y}_i) = \exp(-y_i \cdot y_i), \quad y_i \in \{-1, 1\}$$

  - Absolute Loss

$$L(y_i, \hat{y}_i) = |y_i - \hat{y}_i|$$



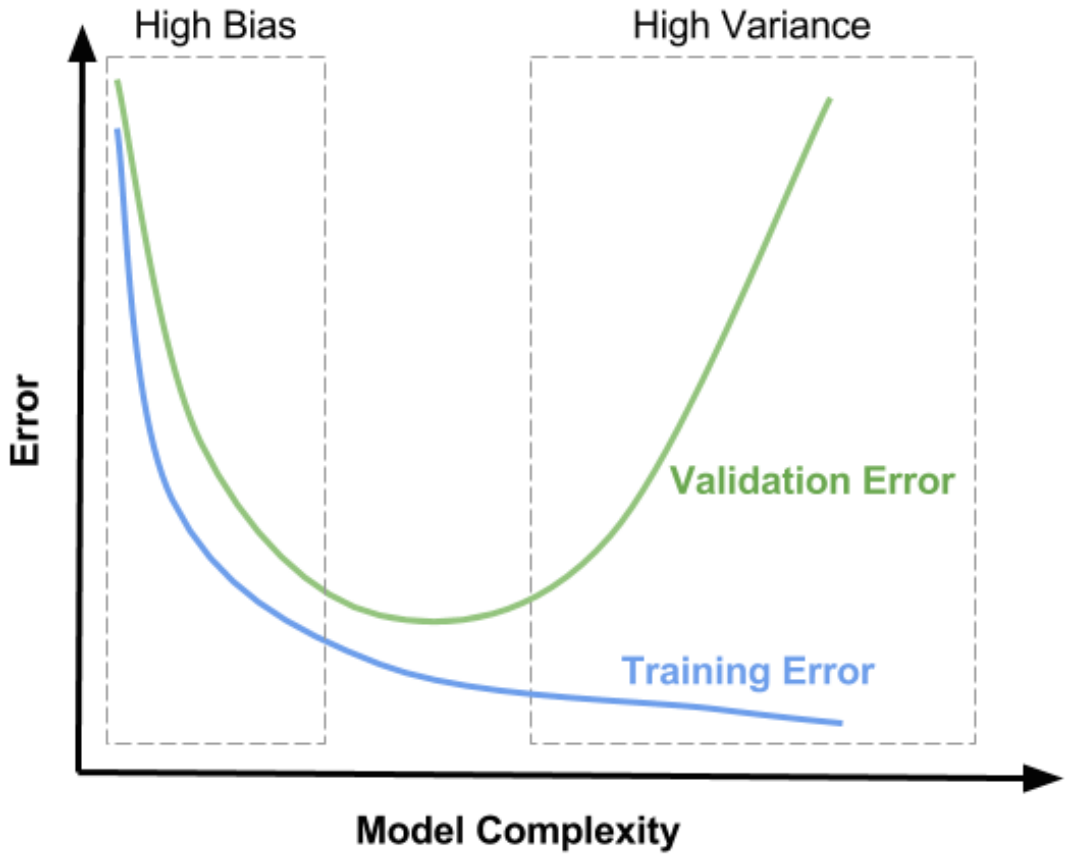
# 统计学习理论

- 结构风险
  - 只考虑经验风险，会出现过拟合现象，即模型$f(x)$对训练集中所有的样本点都有最好的预测能力，但是对于非训练集中的样本数据，模型的预测能力非常不好
  - 结构风险是对经验风险和期望风险的折中，在经验风险函数后面加一个正则化项（惩罚项），正则化项表示的是模型的复杂度
  - 经验风险越小，模型决策函数越复杂，其包含的参数越多，当经验风险函数小到一定程度就出现了过拟合现象
  - 模型决策函数的复杂程度是过拟合的必要条件，要想防止过拟合现象需要降低决策函数的复杂度。

# 经验风险与模型复杂度

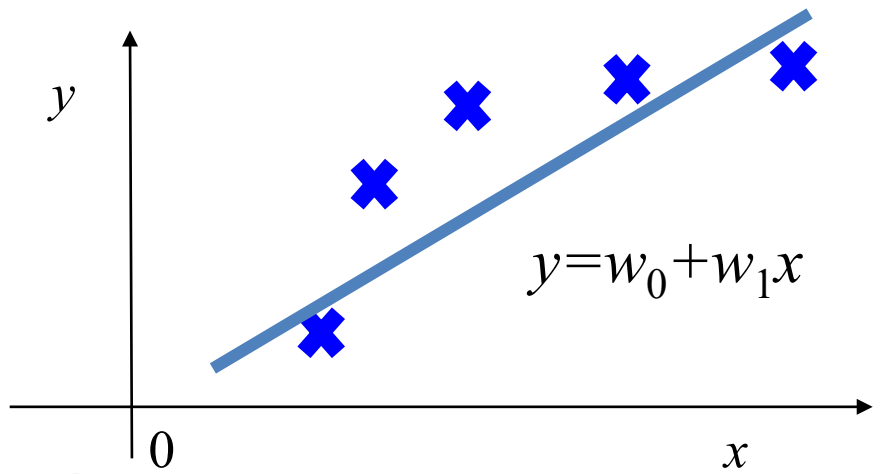
过犹不及

- 训练误差
  - 学习到的模型的预测值与真实值（标签）在训练数据集上的平均误差
  - 本质上不重要，仅对判断给定问题是否为容易学习的问题有意义
- 测试误差
  - 预测值与真实值在测试数据集上的平均误差
  - 反映了学习方法对未知测试数据的预测能力，是重要指标
  - 通常将学习方法对未知数据的预测能力称为泛化能力

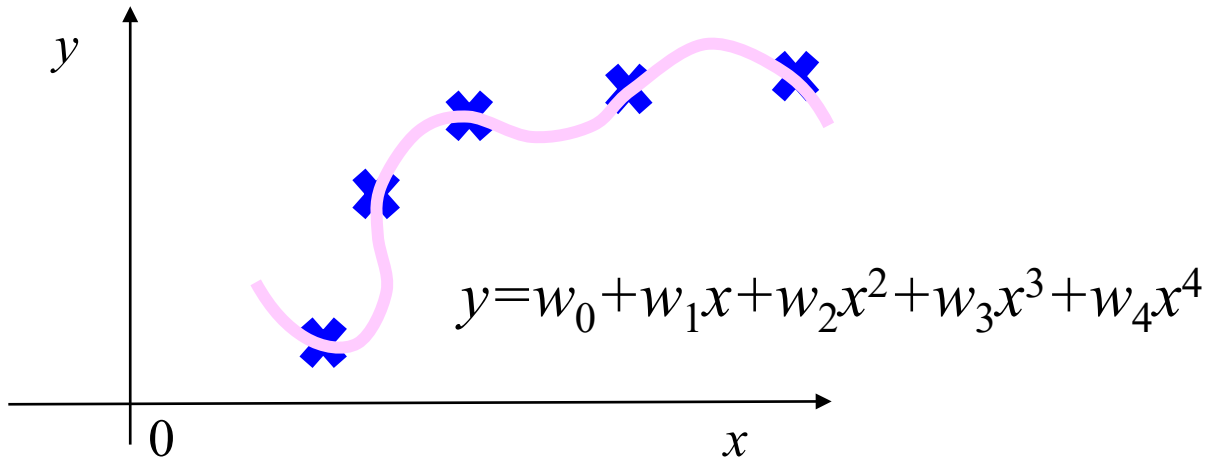


# 欠拟合与过拟合

- 欠拟合——模型没有很好地捕捉到数据特征，不能够很好地拟合数据
- 过拟合——一味追求对训练数据的预测能力，模型的复杂程度高于真实模型，泛化能力差



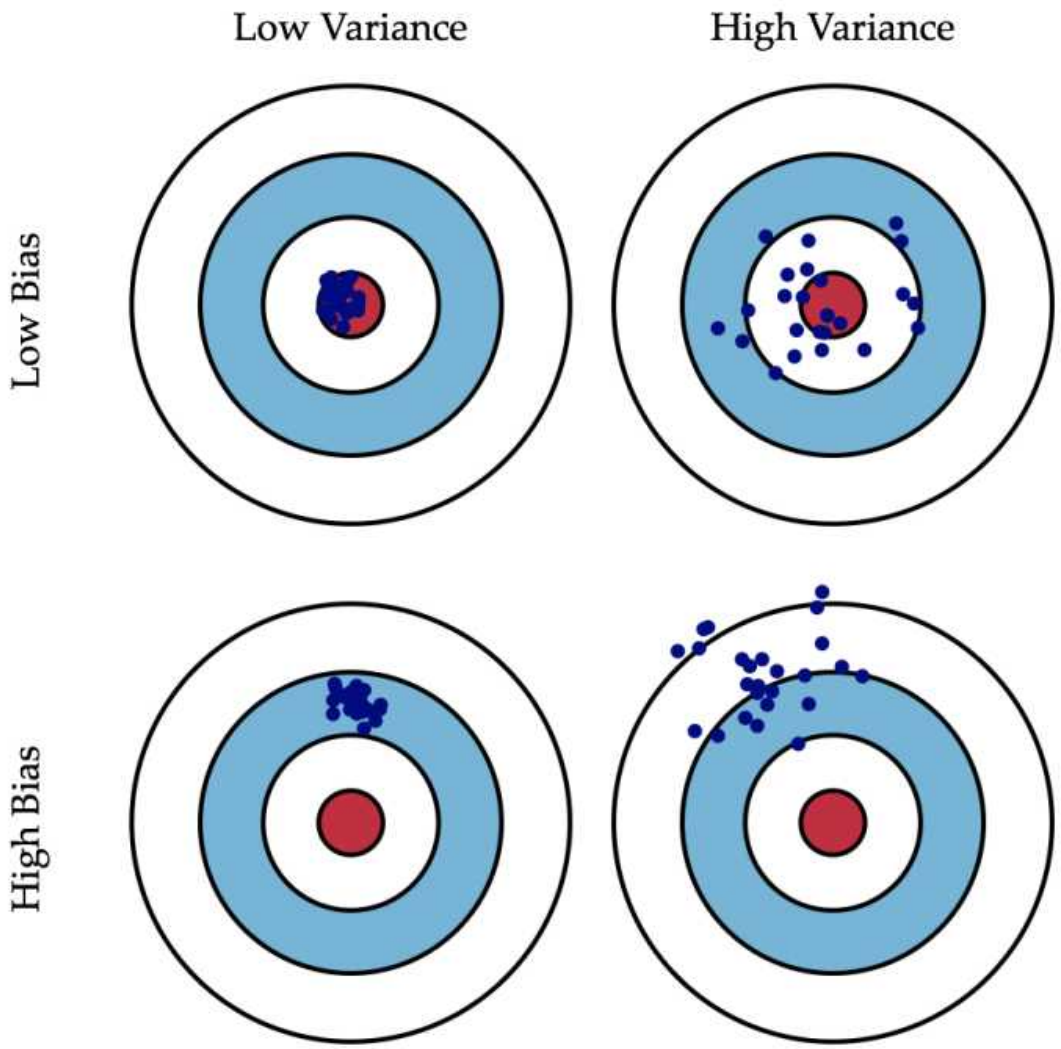

欠拟合(underfitting)
高偏差 (high bias)：与真实值偏差大




过拟合(overfitting)
高方差(high variance)：波动大，分散

# 方差与偏差

- Error = Bias + Variance
- Bias反映的是模型在样本上的输出与真实值之间的误差，即模型本身的精准度
- Variance反映的是模型每一次输出结果与模型输出期望之间的误差，即模型的稳定性



# 常用机器学习方法

- 十大经典算法
  - 线性回归（回归问题）
  - 朴素贝叶斯
  - K近邻（KNN）
  - 支持向量机（SVM）
  - 逻辑回归
  - 决策树
  - 神经网络（分类、聚类、深度学习）
  - K均值（KMEANS）（无监督，聚类）
  - 提升法和Adaboost（集成学习）
  - Bagging和随机森林（集成学习）
- 深度学习