线性分类器（3）

叶山 中国地质大学（北京）

yes@cugb.edu.cn

本教案部分参考以下课程：斯坦福大学 CS 231、密歇根大学 EECS 442 & 498、威斯康辛大学麦迪逊分校 CS 766

上期回顾

正则项

数据损失（data loss）　　正则损失（regularization）

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

分类器模型预测结果应和训练数据集相符

避免模型在训练数据集上的表现过于优秀

正则项又称惩罚因子，被用来抑制模型在训练集上的性能。

正则项$R(W)$函数的输出值只和W有关，和图像x无关。在添加常见的正则项以后，当总损失一定时（如$L = 0$），W通常有唯一解。

超参数λ控制正则损失$R(W)$占总损失L的占比。

正则项

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

$$L_i = \sum_{j \neq y_i} \max(0, s_{ij} - s_{y_i} + 1)$$

$$R(W) = \sum_k \sum_l W_{k,l}^2$$ L2正则项

作用：抑制较大权值的作用，从而促使权值的分散，鼓励模型综合使用所有的特征，而非过度依赖少数权值较大的特征，从而让模型更稳健，减少噪声的影响。

鼓励分类器2的权值分布（权值平均分布，分类时考虑所有特征维度），不鼓励分类器1的权值分布（权值集中，分类时只参考了少量特征维度）。

L2正则项计算举例

样本数据 $x = [1,1,1,1]$

分类器1权值 $W_1 = [1,0,0,0]$

分类器2权值 $W_2 = [0.25,0.25,0.25,0.25]$

分类器的输出 $W_1^T x = W_2^T x = 1$

忽略偏置，假设$\lambda = 1$

两个分类器数值损失相同

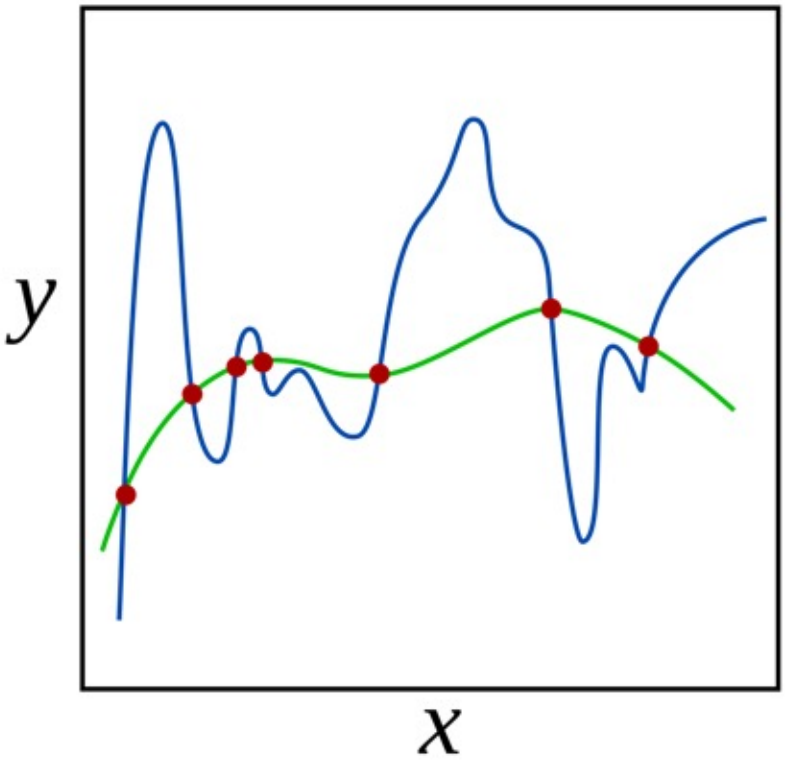
分类器1的正则损失：
$R(W_1) = 1^2 + 0^2 + 0^2 + 0^2 = 1$

分类器2的正则损失：
$R(W_2) = 0.25^2 + 0.25^2 + 0.25^2 + 0.25^2 = 0.25$

分类器2的总损失小

正则项

抑制过拟合现象：不要“死记硬背”训练数据集中的局部规律，鼓励模型去学习宏观规律。



正则项

- 让权值参数W形成唯一解
- 鼓励权值分散，增加模型的冗余度
- 抑制过拟合现象

优化算法

参数优化

利用损失函数的输出值（损失值）作为反馈信号来调整分类器的模型参数，从而提升分类器对训练样本的预测性能。

目标：通过调整模型参数，让损失值尽量变小。

数学意义：机器学习中，需要用一个数学模型来解释训练数据集的规律。该数学模型过于复杂，难以直接求得最优解，但可以通过多次迭代，逐渐靠近最优解。

参数优化

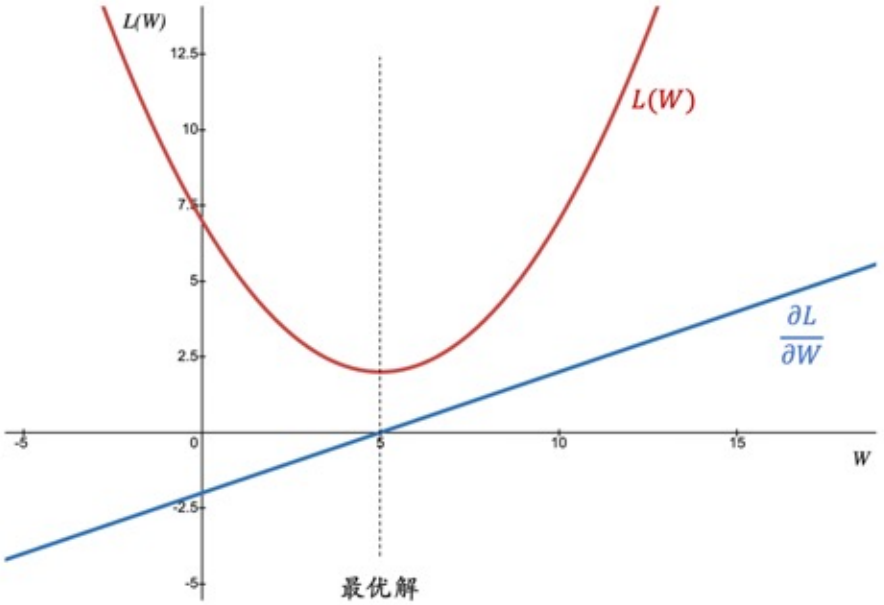
损失函数是一个与参数W（即权值矩阵，并且融合了偏置b）有关的函数，而参数优化的目标是找到能让损失函数L达到最小的参数W。

损失函数 $$L = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

参数优化 $$W^* = arg\min_W L(W)$$

当损失函数L为凸函数时（常见情况），优化寻找的目标为让L的一阶导数为0的W：

$$\frac{dL}{dW} = 0$$



参数优化

通常情况下，L很复杂，难以直接求出让其导数为0的参数W（以及偏置b）。

以CIFAR-10的分类任务为例：

$$\begin{cases} \frac{dL}{dW_1} = 0 \\ \frac{dL}{dW_2} = 0 \\ \vdots \\ \frac{dL}{dW_{10}} = 0 \end{cases} \quad \begin{cases} \frac{dL}{db_1} = 0 \\ \frac{dL}{db_2} = 0 \\ \vdots \\ \frac{dL}{db_{10}} = 0 \end{cases}$$

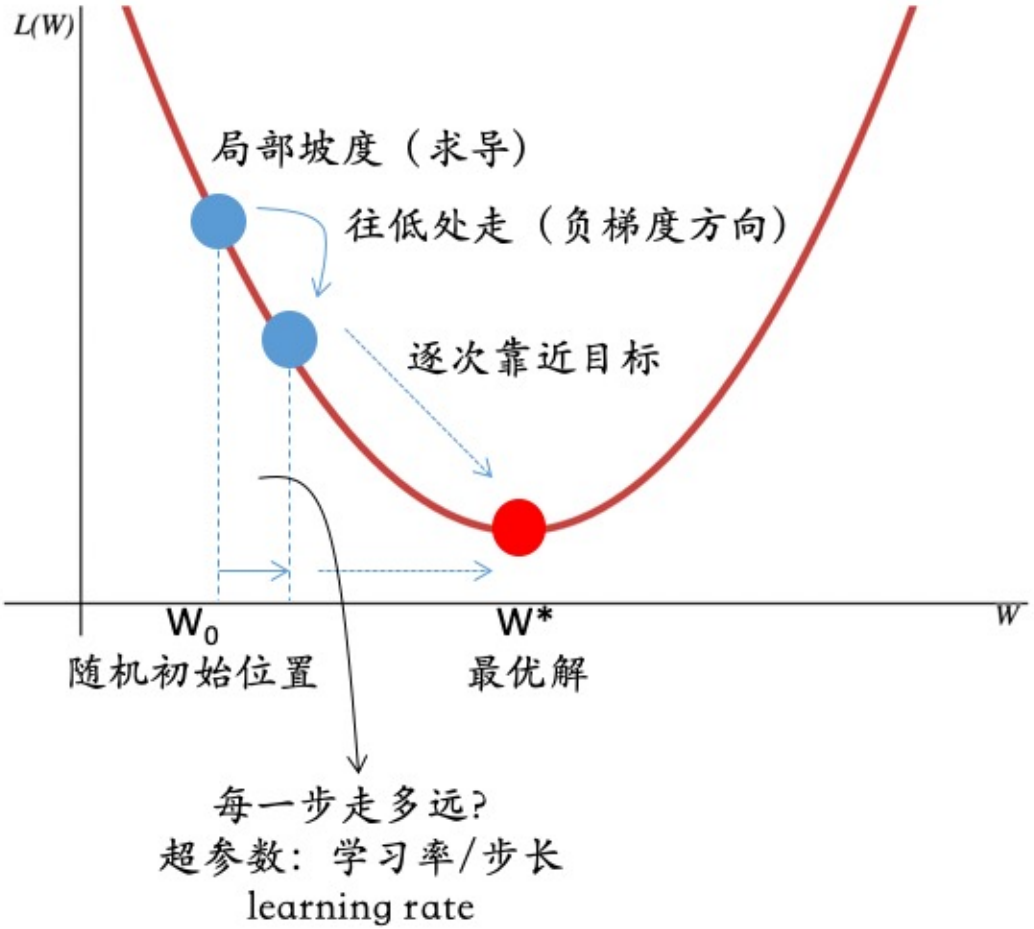
20个式子联立求解
直接求解难度通常很大
需要采用其他手段

梯度下降
gradient descent

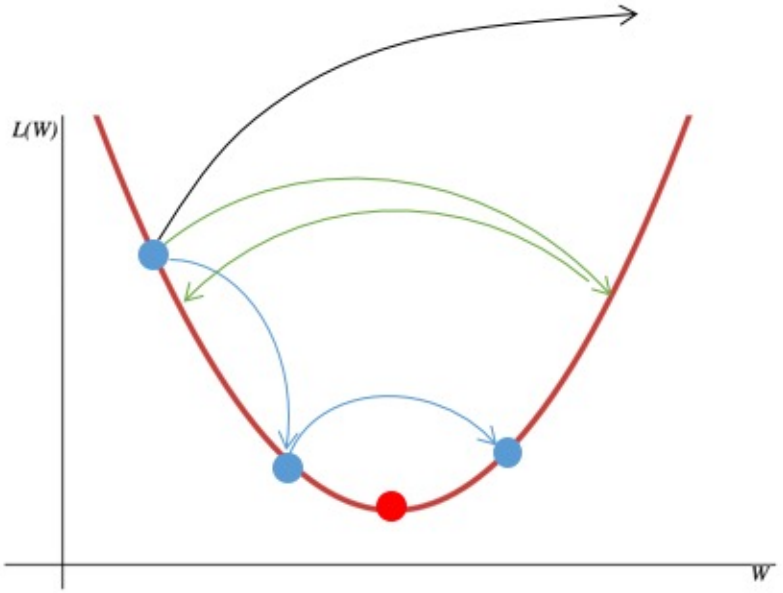
梯度

损失函数 $$L = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

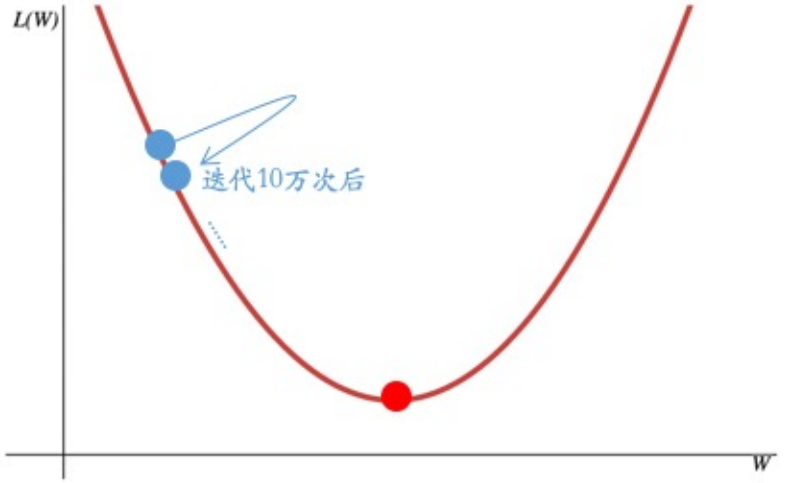
参数优化 $$W^* = arg\min_W L(W)$$



梯度



学习率过大
模型不稳定、难以收敛



学习率过小
训练速度慢、可能被困于局部最优解

梯度的计算：数值近似法

$$\frac{dL(w)}{dw} = \lim_{h \to 0} \frac{L(w+h) - L(w)}{h}$$

数值梯度 numeric gradient
优点：计算简单易懂
缺点：不精确、计算量大

例题：求损失函数$L(w) = w^2 + 3w + 5$在$w = 0.5$处的梯度。

解：假设h为0.0001，代入$w = 0.5$可得

$$\frac{dL(w)}{dw} \approx \frac{L(0.5 + 0.0001) - L(0.5)}{0.0001}$$

$$= \frac{(0.5001^2 + 3 \times 0.5001 + 5) - (0.5^2 + 3 \times 0.5 + 5)}{0.0001}$$

$$= 4.0001$$

梯度的计算：梯度解析法


解析梯度：analytical gradient
用微积分的方法算梯度
优点：精确值、计算量小
缺点：计算复杂、易错

例题：求损失函数$L(w) = w^2 + 3w + 5$在$w = 0.5$处的梯度。

解：对$L(w) = w^2 + 3w + 5$求导

$\nabla L(w) = 2w + 3$

代入$w = 0.5$可得

$\nabla L(0.5) = 2\times0.5 + 3 = 4$

梯度检验

数值梯度：简单不易错的近似值，速度慢

解析梯度：复杂易错的精确值，速度快

梯度检验：计算解析梯度，但用数值梯度对其计算的正确性进行验算。

模型参数和梯度

权值向量W

模型参数

[0.34,
-1.11,
0.78,
0.12,
0.55,
2.81,
-3.1,
-1.5,
0.33,...]

loss 1.25347

梯度值$\frac{\partial L}{\partial W}$

[-2.5,
0.6,
0,
0.2,
0.7,
-0.5,
1.1,
1.3,
-2.1,...]

通常在反向传播时计算梯度
每一个模型参数都可计算梯度
梯度指导模型对参数进行优化

梯度小于0时，该模型参数需增大
梯度大于0时，该模型参数需减小
梯度等于0时，该模型参数达到最优解

随机梯度下降

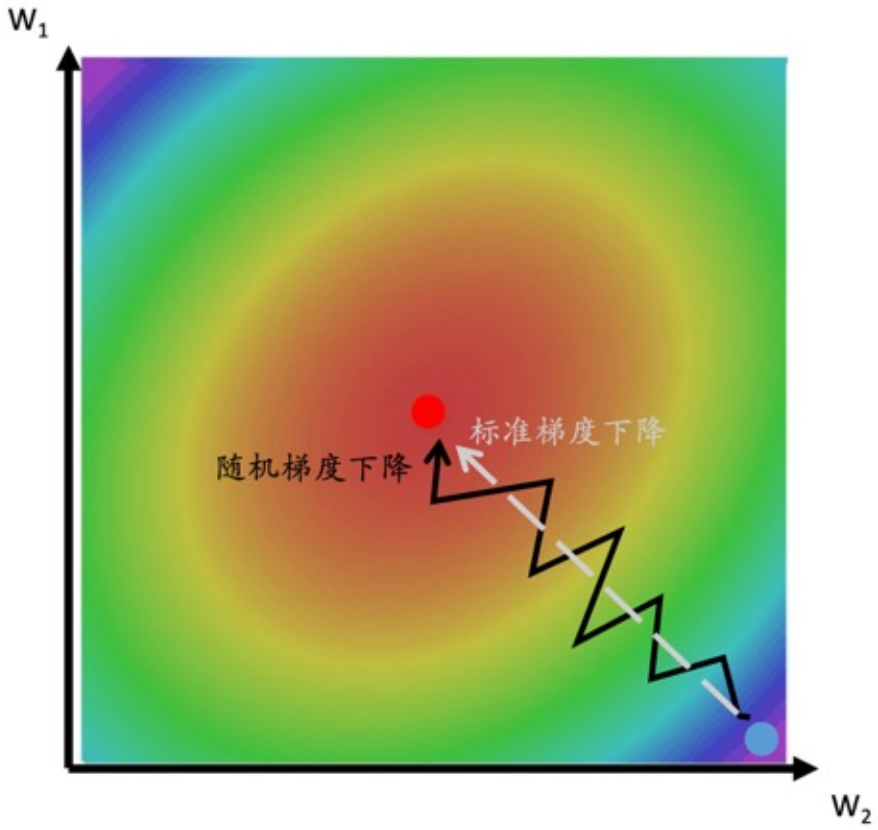
100万个样本做梯度下降

方案1（全样本/标准/批量梯度下降）：
每次迭代需完成每个样本的梯度计算，找到最小损失。

方案2（随机梯度下降）：
每次只针对一个随机样本做梯度下降，近似计算样本集梯度，从而找到最小损失。

$$L = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i) + \lambda R(W)$$

随机梯度下降



随机梯度下降（stochastic gradient descent，SGD）广泛应用于机器学习和数据挖掘领域，在处理大规模数据集时表现出了高效性。

在全样本（标准）梯度下降中，每次迭代都需要计算整个数据集的梯度。然而，在大规模数据集上，这可能会非常耗时。随机梯度下降在每次迭代中仅使用一个样本来近似计算梯度，从而大幅提高计算效率，且不会明显影响优化结果。

随机梯度下降

1. **初始化参数**：设定好模型的初始参数值。
2. **随机选择样本**：从数据集中随机选择一个样本。
3. **计算梯度**：计算选中样本的损失函数相对于当前参数的梯度。
4. **更新参数**：根据计算出的梯度，按照学习率，向负梯度方向更新参数。
5. **重复迭代**：重复步骤2-4，直到满足停止条件（如达到预设的迭代次数或损失函数值收敛）。

```
w = initialize_weights()
for t in range(num_steps):
  dw = compute_gradient(loss_fn, data, w)
  w -= learning_rate * dw
```

随机梯度下降

优点	缺点
• **训练速度快**：由于每次只处理一个样本，SGD的训练速度非常快。 • **适用于大规模数据集**：SGD的高效性使其特别适用于大规模数据集。 • **可能跳出局部最优**：每次下降的路径不同，多次重复能有效绕开局部最优解，找到全局更优的解。	• **更新不稳定**：由于每次只考虑一个样本，SGD的更新可能具有较大的随机性，导致更新不稳定。 • **学习率的设置**：SGD的收敛速度受学习率影响较大，调参时需要仔细调整学习率，以达到最佳性能。

小批量梯度下降

小批量梯度下降（mini-batch gradient descent，MBGD）是介于标准梯度下降和随机梯度下降两者之间的折中方案，它每次使用一个以上而又并非全部的训练样本来代表全部数据集，进行梯度计算。

- 每次从训练集中提取一批数量为m的样本（m为批量大小/batch size，通常为2的倍数）进行学习，而不是对所有样本进行学习。
- 可以在保证训练精度的同时，也提高训练速度、降低计算成本。

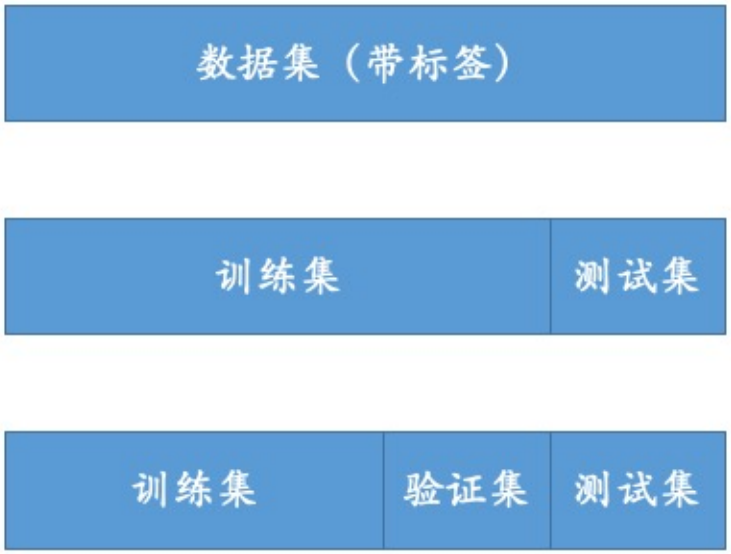

```
w = initialize_weights()
for t in range(num_steps):
  minibatch = sample_data(data, batch_size)
  dw = compute_gradient(loss_fn, minibatch, w)
  w -= learning_rate * dw
```

梯度下降小结

全样本梯度下降（批量下降）	• 需要让所有样本参与梯度更新的运算。
随机下降	• 从数据集里随机采样一个样本，来代表全部数据集，进行梯度更新。
小批量下降	• 把样本分批运行，每次用其中一批代表全部数据。

训练数据

数据集的划分



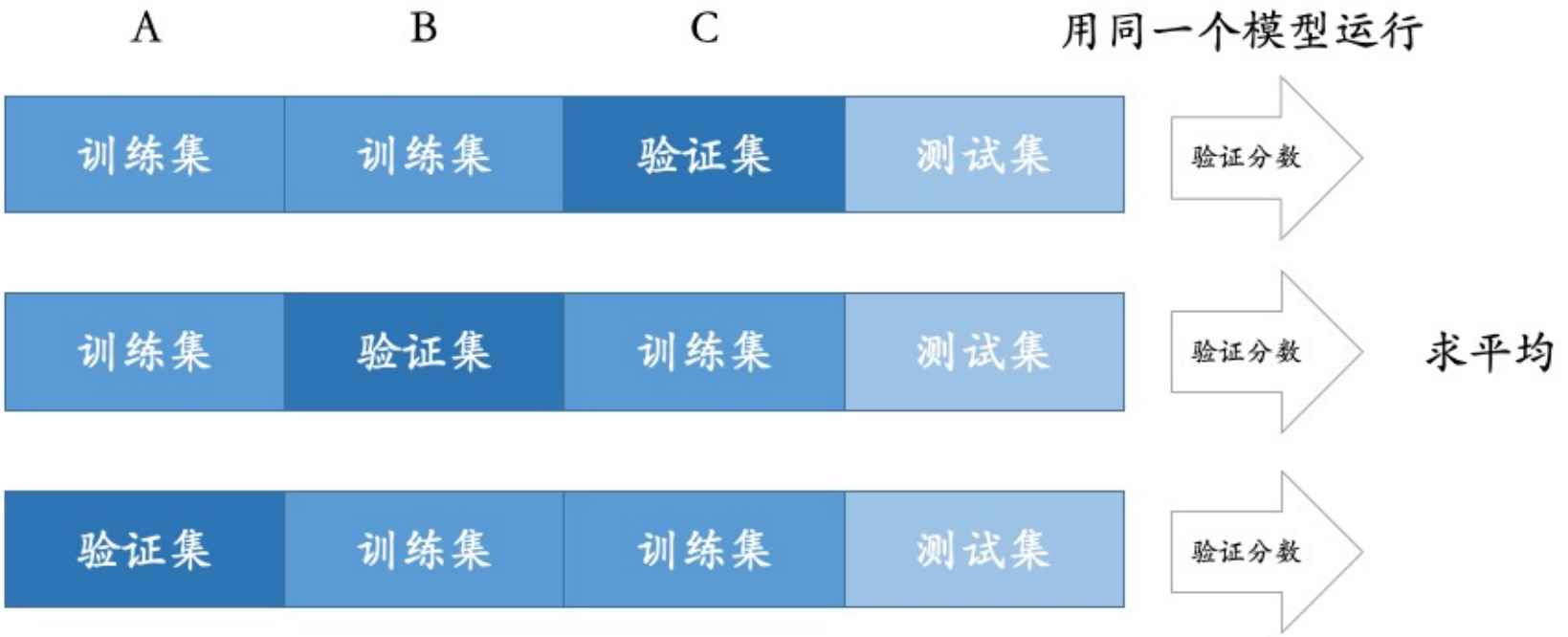
训练集 training set：训练模型
验证集 validation set：选择超参数
测试集 testing set：评估性能和泛化能力

交叉验证

对比模型选择超参数时，如果验证集中的数据量不够多，无法带来统计学上的显著意义，或者发现随机打乱数据会得到明显不同的结果，应该怎么办?

交叉验证

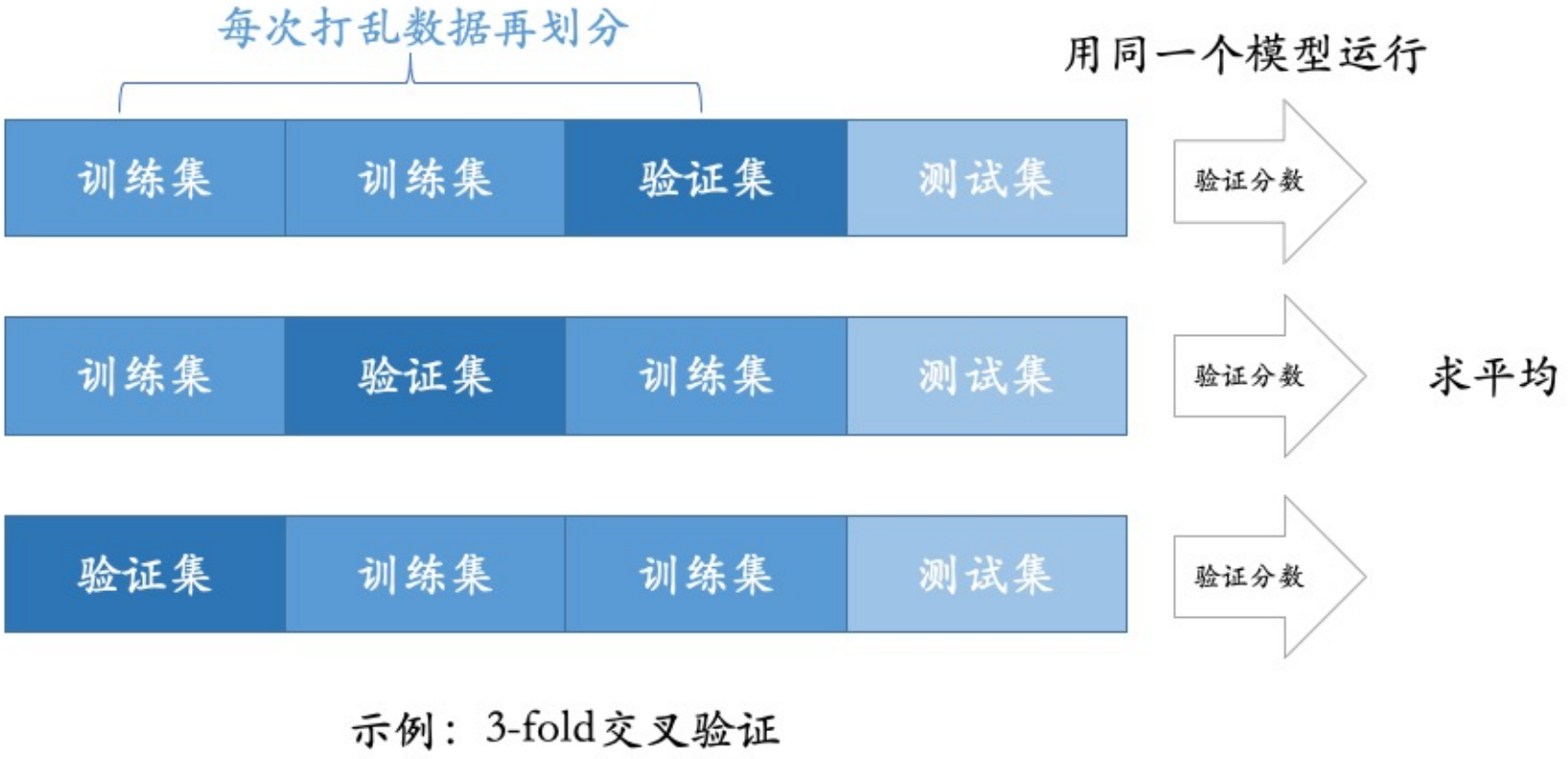
K-fold cross validation



示例：3-fold交叉验证

交叉验证

K-fold cross validation



示例：3-fold交叉验证

数据预处理

- 缺失值填充：当数据集中存在缺失值时，可以使用插值、回归、分类等方法进行填充。
- 异常值检测与处理：异常值是指在数据集中出现的不符合正常规律的数值，可以使用统计方法（如箱线图、Z-score法等）进行检测，并通过删除、替换等方式进行处理。
- 特征缩放：特征缩放是指将特征的取值范围缩放到一个较小的范围内，以避免某些特征对模型训练产生过大的影响。常见的特征缩放方法有最小-最大标准化（min-max scaling）和Z-score标准化。
- 特征选择：特征选择是指从原始特征中选择出一部分最有价值的特征，以减少计算量和提高模型性能。
- 特征编码：特征编码是指将非数值型特征转换为数值型特征，以便于后续的机器学习算法进行处理。常见的特征编码方法有独热编码（one hot encoding）、标签编码（label encoding）等。

独热编码 One hot encoding

状态寄存器

类别标签：[狗, 猫, 鸟, 猪, 熊]

y

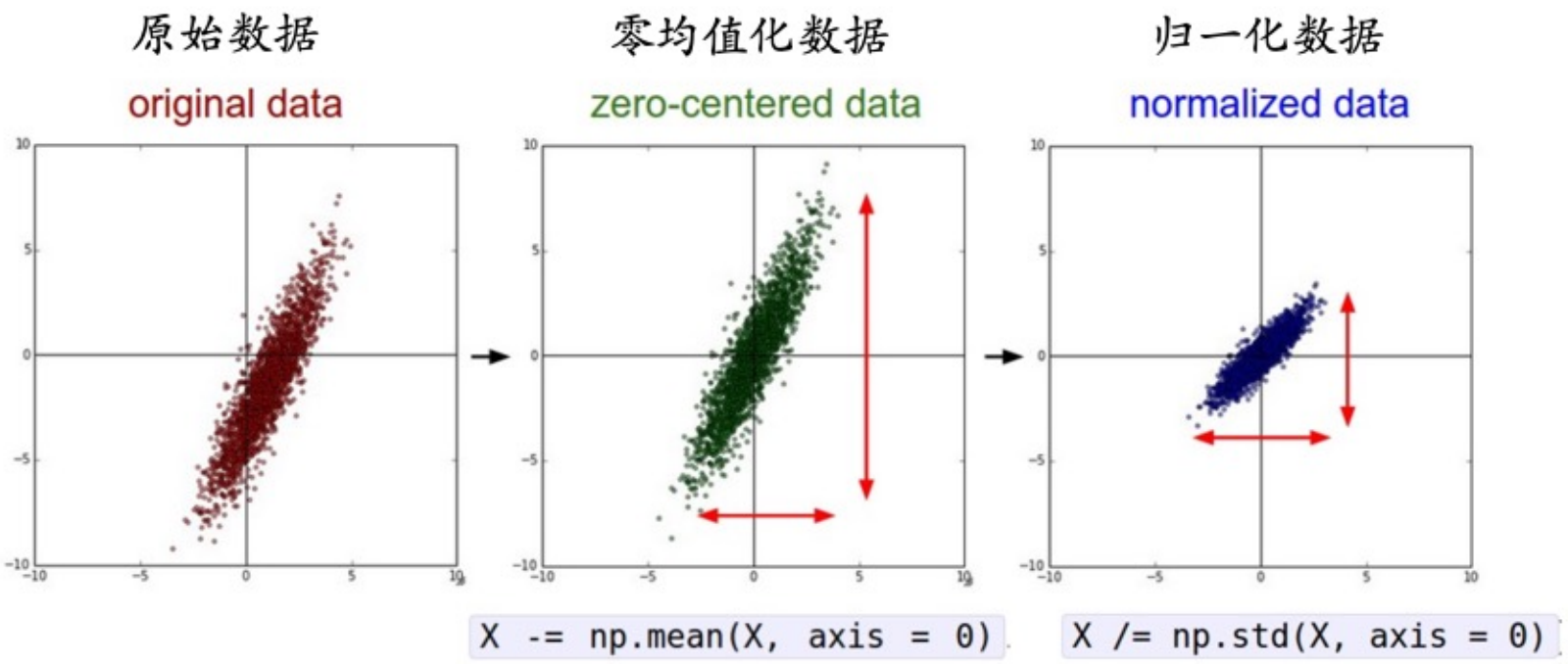
[1, 0, 0, 0, 0]

[0, 1, 0, 0, 0]

机器可读
易于量化

[0, 0, 1, 0, 0]

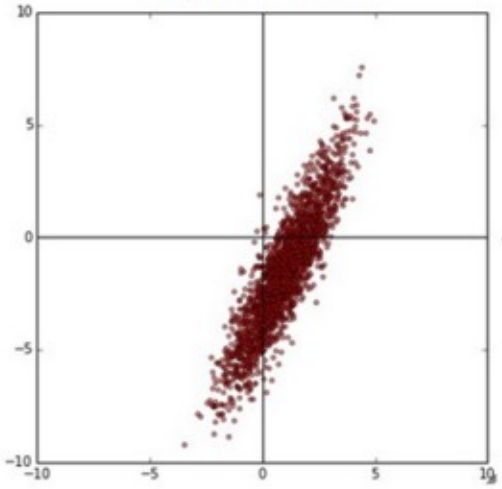
数据预处理


原始数据
original data
零均值化数据
zero-centered data
归一化数据
normalized data
X -= np.mean(X, axis = 0)
X /= np.std(X, axis = 0)
统一均值
统一方差
数据标准化（standardization）处理

数据预处理

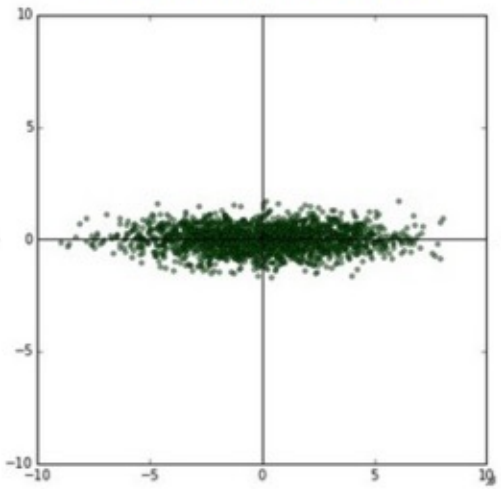
原始数据

original data



去相关化数据

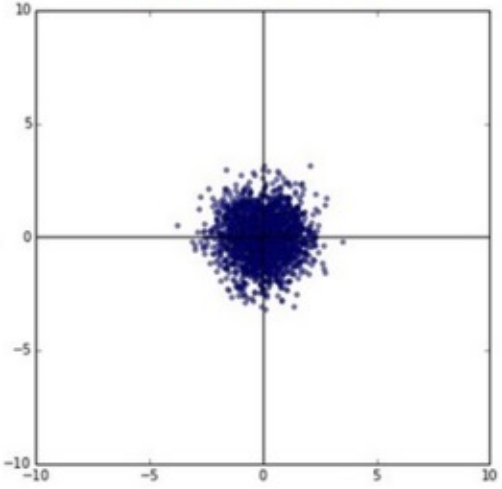
decorrelated data



协方差矩阵为对角线

白化数据

whitened data



协方差矩阵为单位矩阵