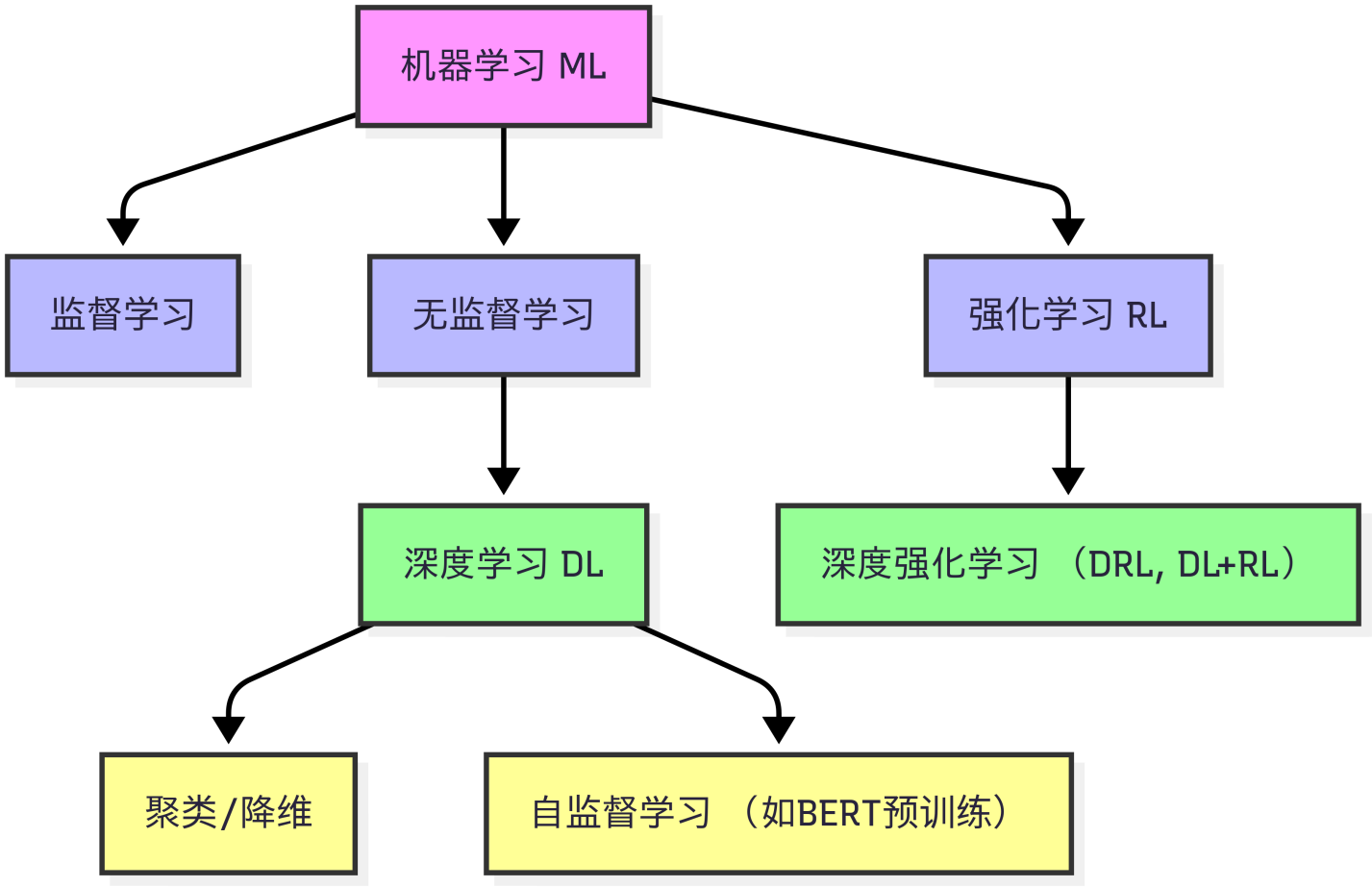
+++
date = '2025-06-16T19:29:56+08:00'
draft = true
title = '学习笔记-机器学习'
description = "The process of studying machine learning"
tags = ["Machine learning"]
categories = [
"学习笔记"
]
+++



深度学习：它的核心是人工神经网络（Artificial Neural Networks, ANN），尤其是拥有多个隐藏层的“深度”神经网络，通过模拟人脑神经元的工作方式，能够自动从数据中学习出极其复杂和抽象的特征。

强化学习：它关注的是如何在一个环境中，让一个智能体（Agent）通过“试错”的方式学习，以采取一系列行动来最大化其获得的累积奖励。

0 算法速览

监督学习算法：

- 线性回归（Linear Regression）：用于回归任务，预测连续的数值。
- 逻辑回归（Logistic Regression）：用于二分类任务，预测类别。
- 决策树（Decision Tree）：基于树状结构进行决策的分类或回归方法。
- 支持向量机（SVM）：用于分类任务，构建超平面进行分类。
- K近邻算法
- 集成学习

无监督学习算法：

- K-means 聚类：通过聚类中心将数据分组。
- 主成分分析（PCA）：用于降维，提取数据的主成分。

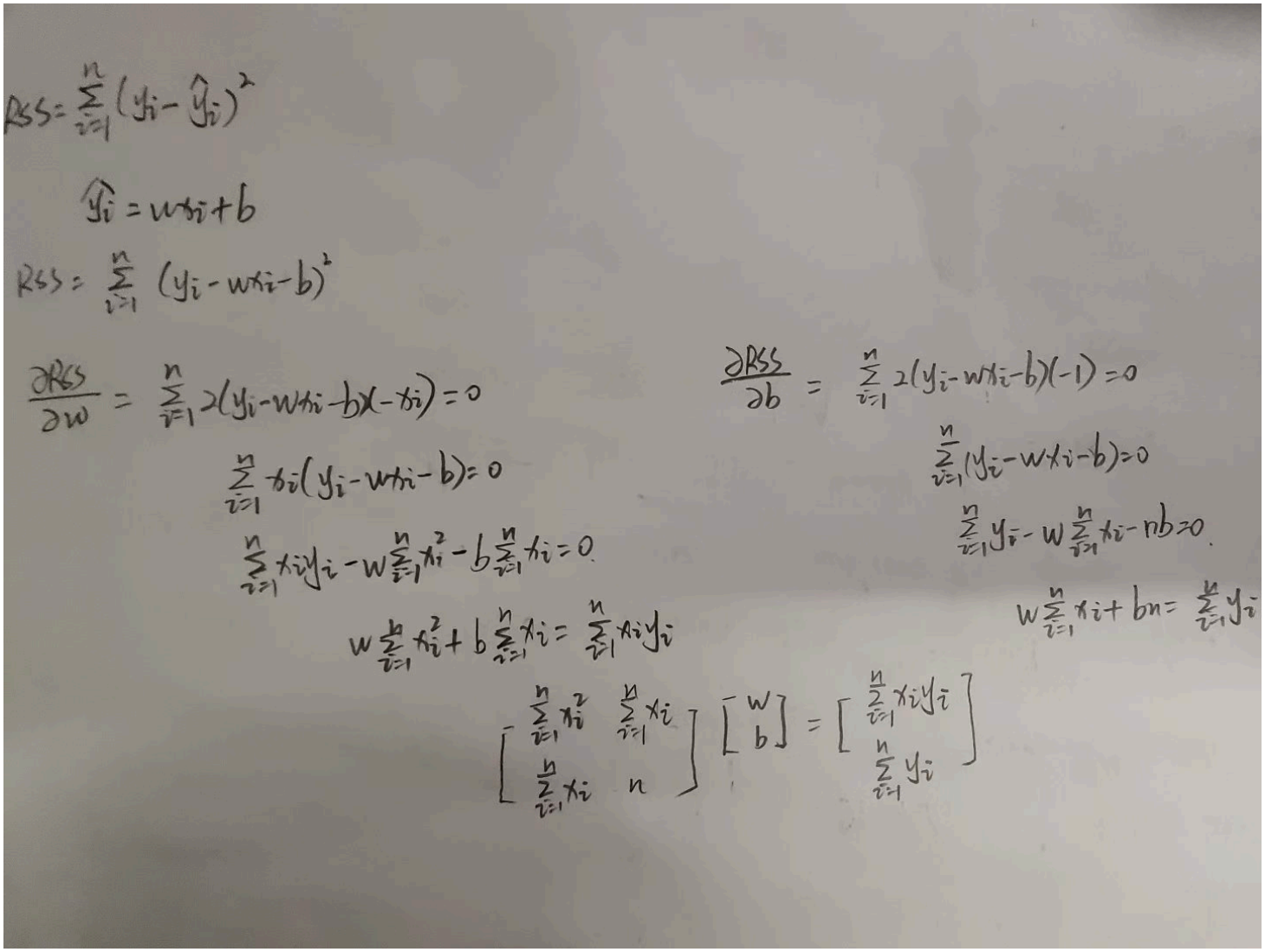
0.1 线性回归

线性回归 (Linear Regression) 是一种用于预测连续值的最基本的机器学习算法，它假设目标变量 y 和特征变量 x 之间存在线性关系，并试图找到一条最佳拟合直线来描述这种关系。

常用的误差函数是均方误差 (MSE)：$MSE = 1/n * \Sigma(y_i - y_{pred_i})^2$

求解方法-最小二乘法

最小二乘法的目标是最小化残差平方和（RSS），其公式为：$RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$



得到最佳的 w, b

$$\begin{bmatrix} w \\ b \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{n} x_i^2 & \sum_{i=1}^{n} x_i \\ \sum_{i=1}^{n} x_i & n \end{bmatrix}^{-1} \begin{bmatrix} \sum_{i=1}^{n} x_i y_i \\ \sum_{i=1}^{n} y_i \end{bmatrix}$$

求解方法-梯度下降法

梯度下降法的目标是最小化损失函数 $J(w, b)$ 。对于线性回归问题，通常使用均方误差（MSE）作为损失函数：

$$J(w, b) = \frac{1}{2m} \sum_{i=1}^{m} (y_i - \hat{y}_i)^2$$

参数更新：

$$w := w - \alpha \frac{\partial J}{\partial w} \quad b := b - \alpha \frac{\partial J}{\partial b}$$

梯度下降法的步骤

1. 初始化参数：初始化 w 和 b 的值（通常设为 0 或随机值）。
2. 计算损失函数：计算当前参数下的损失函数值 $J(w, b)$ 。
3. 计算梯度：计算损失函数对 w 和 b 的偏导数。
4. 更新参数：根据梯度更新 w 和 b。
5. 重复迭代：重复步骤 2 到 4，直到损失函数收敛或达到最大迭代次数。

0.2 逻辑回归

逻辑回归（Logistic Regression）是一种广泛应用于分类问题的统计学习方法，尽管名字中带有"回归"，但它实际上是一种用于二分类或多分类问题的算法。

逻辑回归通过使用逻辑函数（也称为 Sigmoid 函数）将线性回归的输出映射到 0 和 1 之间，从而预测某个事件发生的概率。建立模型：

$$p(y = 1|X) = \sigma(w^T X + b)$$

其中：

- X 是输入特征（可以是多个特征组成的向量）。
- w 是权重向量。
- b 是偏置项。
- $\sigma(z) = \frac{1}{1+e^{-z}}$ 是Sigmoid函数。Sigmoid函数将模型的输出值 $(w^T X + b)$ 映射到0到1之间，因此它可以看作是属于类别1的概率。注意 $\sigma'(z) = \sigma(z)(1 - \sigma(z))$ 。

使用对数损失函数

$$L(\theta) = \begin{cases} -\log(p), & \text{if } y = 1 \\ -\log(1-p), & \text{if } y = 0 \end{cases}$$

合并得到单个样品的损失函数

$$L(\theta) = -ylog(p) - (1-y)log(1-p)$$

因此总体损失函数（也就是交叉熵损失函数）

$$L(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[y^{(i)}\log(p^{(i)}) + (1-y^{(i)})\log(1-p^{(i)})\right]$$

求解方法-梯度下降法

对 w 的梯度：

$$\frac{\partial J(w,b)}{\partial w} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x^{(i)}$$

对 b 的梯度：

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})$$

$$\frac{\partial J(w,b)}{\partial w} = \frac{\partial J(w,b)}{\partial p^{(i)}} \frac{\partial p^{(i)}}{\partial w^T x} \frac{\partial w^T x}{\partial w}$$

$$\frac{\partial J(w,b)}{\partial p^{(i)}} = -\frac{1}{m} \sum_{i=1}^{m} \left(\frac{y^{(i)}}{p^{(i)}} - \frac{1-y^{(i)}}{1-p^{(i)}} \right)$$

$$\frac{\partial p^{(i)}}{\partial w^T x} = p^{(i)}(1-p^{(i)})$$

$$\frac{\partial w^T x}{\partial w} = x$$

故 $$\frac{\partial J(w,b)}{\partial w} = \frac{1}{m} \sum_{i=1}^{m} (p^{(i)} - y^{(i)}) x^{(i)}$$

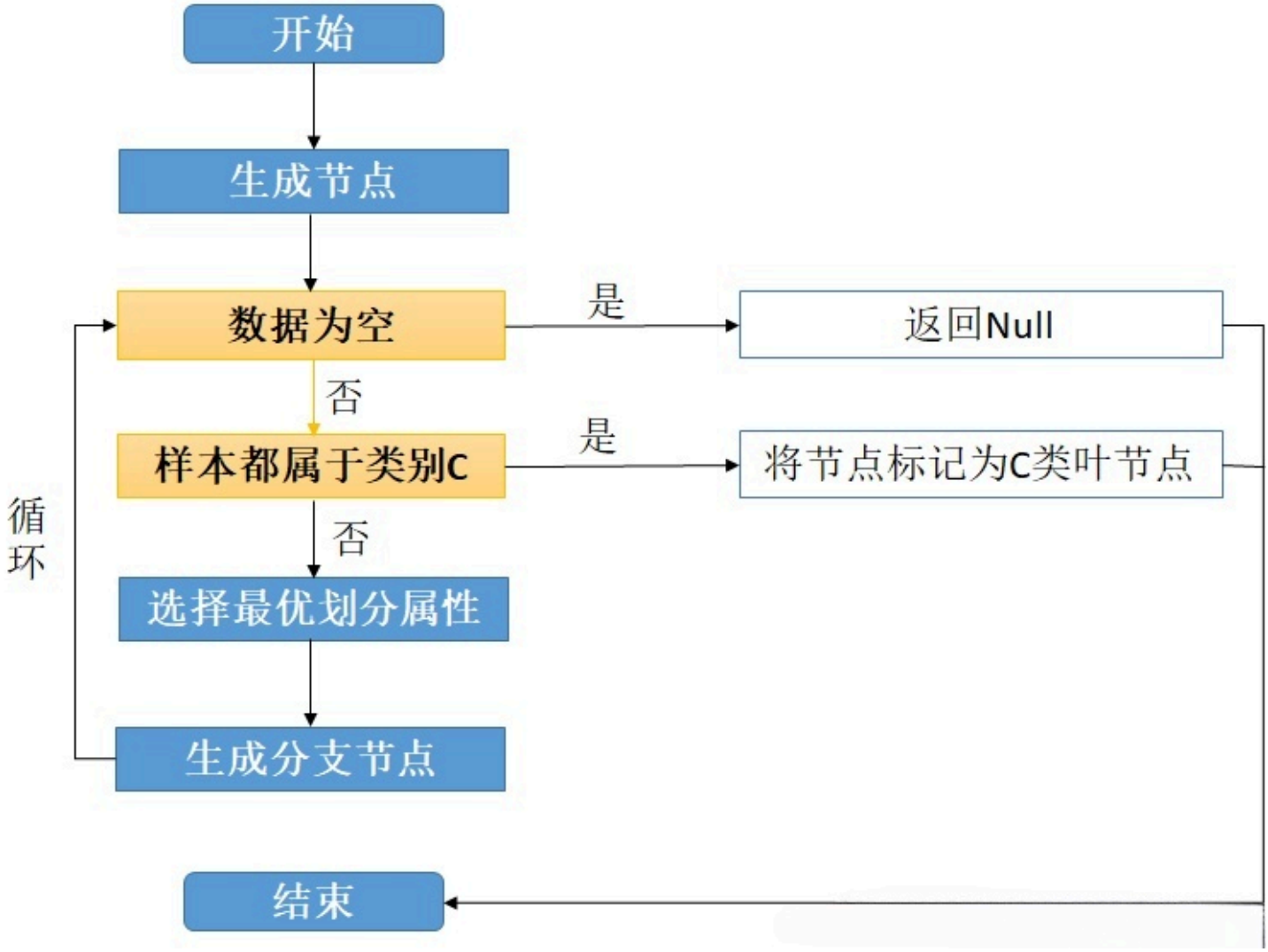
$$\frac{\partial J(w,b)}{\partial b} = \frac{\partial J(w,b)}{\partial p^{(i)}} \frac{\partial p^{(i)}}{\partial b}$$

$$\frac{\partial p^{(i)}}{\partial b} = p^{(i)}(1-p^{(i)})$$

故 $$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m} \sum_{i=1}^{m} (p^{(i)} - y^{(i)})$$

0.3 决策树

决策树（Decision Tree），它是一种以树形数据结构来展示决策规则和分类结果的模型，作为一种归纳学习算法，其重点是将看似无序、杂乱的已知数据，通过某种技术手段将它们转化成可以预测未知数据的树状模型，每一条从根结点（对最终分类结果贡献最大的属性）到叶子结点（最终分类结果）的路径都代表一条决策的规则。



在这个过程中，寻找最优划分属性是决策树过程中的重点，那么应该如何求解呢？

求解方法-信息增益

首先引入信息熵的概念

> 信息熵：描述随机变量的不确定性（也就是混乱程度）。
> 假设某随机变量的概率分布为：$P(X=x_i)=p_i,\quad i=1,2,\ldots,n$，则它的信息熵计算公式为：$H(X)=-\sum_{i=1}^{n}p_i\log p_i$

在决策树中，信息熵

$$H(D)=-\sum_{k=1}^{K}\frac{|D_k|}{|D|}\log\frac{|D_k|}{|D|}$$

其中：

- D：整个数据集
- D_k：第 k 个类的样本子集
- $\frac{|D_k|}{|D|}$：第 k 类的概率

条件熵

$$H(D|A)=\sum_{i=1}^{n}\frac{|D_i|}{|D|}H(D_i)$$

$H(D_i)$ 是每个子集 D_i 的信息熵

$$H(D_i)=-\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log\frac{|D_{ik}|}{|D_i|}$$

其中：

- A：某个属性
- D_i：属性 A 的第 i 个取值所对应的数据子集
- D_{ik}：在第 i 个子集中属于第 k 类的样本数

因此有

$$H(D|A)=-\sum_{i=1}^{n}\frac{|D_i|}{|D|}\sum_{k=1}^{K}\frac{|D_{ik}|}{|D_i|}\log\frac{|D_{ik}|}{|D_i|}$$

特征 A 对训练数据集 D 的信息增益 $gain(D,A)$ 定义为集合 D 的信息熵 $H(D)$ 与特征 A 给定条件下 D 的信息条件熵 $H(D|A)$ 之差，即公式为：

$$gain(D,A)=H(D)-H(D|A)$$

信息增益表示得知特征 X 的信息而使得类 Y 的信息的不确定性减少的程度，因此信息增益最大的属性就是最优划分属性，标志性算法 $ID3$ 。

求解方法-增益比

信息增益虽然在理论上可以找到最优的划分属性，但在某些情况下会存在问题。信息增益比较偏好可取值较多的属性。因此为了矫正信息增益偏好的问题，使算法不偏向可取值较多的属性，引申出了增益比

的思想。

$$Gain\_ratio(D,A)=\frac{Gain(D,A)}{H(A)}$$

可以看出，增益比就是信息增益除以属性 A 的信息熵，当属性 A 可取值增多的时候，$H(A)$ 一般也增大，因此在一定程度上能抑制信息增益偏好取值多的属性的特点，但是增益比偏好取值较少的属性。

算法 $C4.5$ 是算法 $ID3$ 的改进版，它使用了信息增益和增益比两种选择算法，先选出信息增益高于平均水平的属性，然后再在这些属性中选择增益比最高的，作为最优划分属性。这样综合了信息增益和增益比的优点，可以取得较好的效果。

求解方法-基尼指数

基尼指数是在样本集中随机抽出两个样本不同类别的概率。当样本集越不纯的时候，这个概率也就越大，即基尼指数也越大。这个规律与信息熵的相同。

$$Gini(D)=\sum_{k=1}^{n}\sum_{k'\neq k}p_kp_{k'}=1-\sum_{k=1}^{n}p_k^2$$

使用基尼指数来选择最优划分属性也是对比不同属性划分后基尼指数的差值，选择使样本集基尼指数减小最多的属性。

$$Gain(D,a)=Gini(D)-\sum_{i=1}^{n}\frac{|D^i|}{|D|}Gini(D^i)$$

著名的 $CART$ 决策树就是使用基尼指数来作为划分准则， $CART$ 决策树与 $ID3$ 和 $C4.5$ 的区别：

1. 划分准则不同，CART决策树使用基尼指数， $ID3$ 和 $C4.5$ 使用信息熵。
2. $ID3$ 和 $C4.5$ 划分时，一个节点可以划分为多个子结点，子结点数量根据属性可取值的数量决定。而 $CART$ 决策树是严格的二叉树结构，就是说 1 个节点最多划分为 2 子结点。

0.4 支持向量机

支持向量机（Support Vector Machine，简称 SVM）是一种监督学习算法，主要用于分类和回归问题。

SVM 的核心思想是找到一个最优的超平面，将不同类别的数据分开。这个超平面不仅要能够正确分类数据，还要使得两个类别之间的间隔（margin）最大化。

> 支持向量：支持向量是离超平面最近的样本点。这些支持向量对于定义超平面至关重要。支持向量机通过最大化支持向量到超平面的距离（即最大化间隔）来选择最佳的超平面。

当训练样本**线性可分**时，通过**硬间隔最大化**，学习一个**线性可分支持向量机**；

当训练样本**近似线性可分**时，通过**软间隔最大化**，学习一个**线性支持向量机**；

当训练样本**线性不可分**时，通过**核技巧和软间隔最大化**，学习一个**非线性支持向量机**。

求解方法-间隔最大化和支持向量

好难……

0.5 K近邻算法

K 近邻算法（K-Nearest Neighbors，简称 KNN）是一种简单且常用的分类和回归算法。

K 近邻算法属于监督学习的一种，核心思想是通过计算待分类样本与训练集中各个样本的距离，找到距离最近的 K 个样本，然后根据这 K 个样本的类别或值来预测待分类样本的类别或值。

求解方法

基本步骤：

1. 计算距离：计算待分类样本与训练集中每个样本的距离。常用的距离度量方法有欧氏距离、曼哈顿距离等。
2. 选择 K 个最近邻：根据计算出的距离，选择距离最近的 K 个样本。
3. 投票或平均：对于分类问题，K 个最近邻中出现次数最多的类别即为待分类样本的类别；对于回归问题，K 个最近邻的值的平均值即为待分类样本的值。

0.6 集成学习

0.7 K-means 聚类

K-means 聚类是一种常用的基于距离的聚类算法，旨在将数据集划分为 K 个簇。算法的目标是最小化簇内的点到簇中心的距离总和。

求解方法

基本步骤：

1. 选择 K 值：设定簇的数量 。
2. 初始化簇中心：随机选择 K 个数据点作为初始簇中心（centroids）。
3. 分配步骤（Assignment Step）：对于数据集中的每个点，将它分配到最近的簇中心对应的簇。这里的“距离”通常使用欧氏距离（Euclidean distance）。
4. 更新步骤（Update Step）：根据当前的簇分配，重新计算每个簇的中心，即计算簇内所有点的均值作为新的簇中心。

5. 重复 3 和 4 步：不断重复分配和更新步骤，直到簇中心不再发生变化（收敛）或达到指定的最大迭代次数。

确定最佳的簇数 K 是 $K-means$ 聚类中的一个难点。聚类的目标是使得每个样本点到距离其最近的聚类中心的总误差平方和（也即聚类的代价函数，记作 SSE ）尽可能小。

空间中数据对象与聚类中心间的欧式距离计算公式为：

$$d(x, C_i) = \sqrt{\sum_{j=1}^{m}(x_j - C_{ij})^2}$$

其中：

- x 为数据对象，
- C_i 为第 i 个聚类中心，
- m 为数据对象的维度，
- x_j，C_{ij} 为 x 和 C_i 的第 j 个属性值。

整个数据集的误差平方和 SSE 计算公式为：

$$SSE = \sum_{i=1}^{k}\sum_{x \in C_i}|d(x, C_i)|^2$$

其中：

- SSE 的大小表示聚类结果的好坏，
- k 为簇的个数。

理论上随着 K 的增加， SSE 会单调递减，当 K 超过某一个数后，每个类簇的聚合程度不再获得显著提升，此时我们就可以认为已找到最佳 K 的取值（肘部法）。

K-means++

K-means++ 是一种改进的初始化方法，可以帮助选择更合理的初始中心，优先选择“距离最远”的点作为初始质心，减少陷入局部最优的风险。

基本步骤：

1. 从数据集 $\mathcal{X}$ 中随机（均匀分布）选取一个样本点作为第一个初始聚类中心;
2. 接着计算每个样本与当前已有聚类中心之间的最短距离，用 $D(x)$ 表示；然后计算每个样本点被选为下一个聚类中心的概率 $P(x) = \frac{D(x)^2}{\sum_{x \in \mathcal{X}} D(x)^2}$，最后选择最大概率值（或者概率分布）所对应的样本点作为下一个簇中心；

3. 重复步骤 2，直到选择 K 个聚类中心

优势：避免随机初始化，加快收敛速度，聚类结果更加稳定。

0.8 主成分分析

主成分分析（PCA）是一种无监督学习方法，旨在通过线性变换将原始的高维数据映射到一个低维空间，同时尽可能保留数据的方差（即信息量）。简单来说，PCA 的目标是找到一组新的坐标轴（称为主成分），这些坐标轴能够捕捉数据中最大的变异性，并用更少的维度来近似表示原始数据。

求解方法

1. 数据中心化：首先将数据中心化，即让每个特征的均值变为 0。
2. 计算协方差矩阵：

$$\mathrm{Cov}(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})$$

协方差为正时，说明X和Y是正相关关系；协方差为负时，说明X和Y是负相关关系；协方差为0时，说明X和Y是相互独立。

3. 特征值分解：对协方差矩阵进行特征值分解，得到主成分的方向（特征向量）和重要性（特征值）。令A 是协方差矩阵，λ 是特征值，I 是单位矩阵，求解

$$\det(A - \lambda I) = 0$$

得到特征值 λ 以后代入 $(A - \lambda I)v_1 = 0$ 解得特征向量 v_1 。

4. 排序和选择主成分：将特征值从大到小排序，特征值最大的为第一个主成分，捕捉了数据中最大的变化，也就是数据分布中最显著的变化方向。第二个主成分与第一个主成分正交（相互垂直），且在正交约束下方差次大的方向。后续主成分：依此类推，每个主成分都与前面的主成分正交，并按特征值大小递减排列。
5. 投影数据：将中心化后的数据投影到第一个主成分上，得到降维后的结果。

一些问题

1. 为什么要计算协方差矩阵?

PCA 的目标是找到一组新的坐标轴（称为主成分），使得数据在这些轴上的投影方差最大化，同时这些轴相互正交（不相关）。协方差矩阵正好量化了数据中的变异性和变量间的相关性，我们可以了解到哪些变量变化较大，哪些变量之间存在较强的关联，为找到这样的轴提供了基础。

2. 为什么要进行特征值分解?

特征值表示每个特征向量方向上的方差大小。特征值越大，说明该方向捕捉的变异性越多。通过特征值分解，我们可以将原始数据投影到这些特征向量上，从而实现降维，同时尽可能保留数据的信息。

1 引言

1.1 什么是机器学习

一个好的学习问题定义如下：一个程序被认为能从经验 E 中学习，解决任务 T，达到性能度量值 P，当且仅当，有了经验 E 后，经过 P 评判，程序在处理 T 时的性能有所提升。

目前存在几种不同类型的学习算法，其中主要的两种类型被我们称之为：**监督学习**和**无监督学习**。

1.2 监督学习

监督学习：给定带有标签的数据，模型通过学习输入和标签之间的关系来做预测。

回归 (regression) 问题：推测出这一系列连续值属性。

分类 (classification) 问题：推测出离散的输出值。

1.3 无监督学习

无监督学习：没有标签的数据，模型通过探索数据中的结构或模式来进行学习。

聚类算法：将数据集划分成两个不同的簇。

鸡尾酒算法：分离两种声音。（一个具体实例，仅仅只需要一行代码实现）

```
[W,s,v] = svd((repmat(sum(x.*x,1),size(x,1),1).*x)*x');
```

2 单变量线性回归 (Linear Regression with One Variable)