1. **机器学习概述**

**机器学习、深度学习、人工智能之间的关系：**

1. 人工智能（Artificial intelligence），简称AI。人工智能是计算机科学的一个分支，是利用数字计算机或数字计算机控制机器、模拟、延伸和扩展人类的智能，感知环境、获取知识获得最佳结果的理论、方法、技术和应用系统。
2. 机器学习（Machine Learning），简称ML。机器学习属于人工智能的一个重要分支，也是人工智能的核心。
3. 深度学习（Deep Learning），简称DL。深度学习是机器学习研究中的一个新的领域。

**机器学习=算力+数据+算法**

**深度学习发展的条件：**

1. 庞大的数据集
2. 算法理论演进，有很多层的网络
3. 算力提升，更快的硬件（如GPU）

**机器学习与数据挖掘的关系：**

1. 机器学习是数据挖掘的重要工具
2. 数据挖掘不仅要研究、扩展、应用一些机器学习方法
3. 数据挖掘可以视为机器学习和数据库的交叉
4. 数据挖掘主要利用机器学习的技术分析海量数据，利用数据库的技术来管理海量数据
5. 机器学习偏理论，数据挖掘偏应用

**机器学习的分类：**

按照学习方式的不同，可以将机器学习分为：

1. 监督学习
2. 非监督学习，也称为无监督学习
3. 半监督学习

监督学习与非（无）监督学习的区别，分类问题与回归问题的区别，参考下图理解：



总结：监督学习：有y标签，非（无）监督学习：没有y标签。

监督学习中又分为两大类：分类算法（y值是有限个类别的离散值）、回归算法（是连续型变量）

**机器学习中数据集的划分：**

包括训练集、验证集（也称为开发集，有时候可以没有）、测试集。

训练集：用于模型拟合的数据样本，进行训练模型。

验证集：模型训练过程中，用于调整模型的超参数和用于对模型能力进行初步评估。

测试集：用来评估最终模型的泛化能力。但是不能作为调参、选择特征等算法相关的选择的依据。

数据集划分没有固定的比例，比如6:2:2或7:2:1都是可以的，具体根据数据集的规模来决定。

**解决机器学习项目的基本流程：**

1. 将项目中的实际问题抽象成机器学习问题，是分类问题、回归问题、或者聚类问题等。
2. 获取数据集。

（3）数据分析，对数据进行描述（统计、可视化等手段搞清数据的分布以及可能隐含的规律，描述性的统计包括平均值、中位数、标准差等）。

（4）数据预处理。对缺失值和异常值的处理，可以采用均值或中位数等填补，消除量纲差异，对数据进行归一化处理，数据去重，离群点/异常值的检测。

（5）特征提取和特征选择。构造特征组合或其他高阶特征，对高维数据进行降维。

（6）数据集的划分。将数据划分为训练集、验证集（也可以没有）、测试集，其中训练集用于模型的训练，验证集进行评估和调参，测试集对模型进行评价。

（7）模型训练和调优，尝试调整各种超参数来训练模型以达到较好的效果。

（8）模型诊断，通过模型评价机制对模型进行评估。

**数据标准化的常用方法：**

1. min-max标准化：

本质上是把数变为[0,1]之间的小数。

转换函数为(X-min)/(max-min),若将数映射到[-1,1],则(x-mean)/(max-min)

1. Z-score标准化(规范化)：

计算原始数据的均值和标准差，经过处理的数据符合标准正态分布，即均值为0，方差为1。

转换函数为：(X-mean)/(Standard deviation)

其中mean为所有样本数据的均值，Standard deviation为所有样本数据的标准差。

**特征提取**

为了提高复杂关系的拟合能力，在特征工程中经常会把一些离散特征两两组合，构造高阶特征。比如房屋的价格，与房屋的长短和宽度等属性，可以通过长度乘以宽度构造面积属性。

**机器学习中的数据不平衡问题解决？**

采样主要用于处理不均衡数据集。包括欠采样和过采样。

随机过采样是从少数类样本中有放回地随机重复抽取样本，而欠采样是从多数类样本中随机选取较少样本。上述两种方法均存在一定的缺陷，随机过拟合采样容易造成过拟合，随机欠采样可能损失部分有用信息，造成欠拟合。方法的选择应视样本规模实际情况而定。

为了解决上述问题，通常在随机过拟合时并不是简单复制样本，而是采取一定方法生成新的样本。比如空间坐标增加少许噪声扰动，或者对于图像数据而言，通过空间变换增加样本（如旋转、平移、裁剪、翻转、缩放、改变亮度等）。

**第二章 K最近邻算法**

**KNN算法的原理和一般步骤：**

1. 设定K值
2. 计算测试数据与各个训练数据之间的距离；
3. 按照距离的递增关系进行排序；
4. 选取距离最小的K个点；
5. 确定前K个点所在类别的出现次数（频率）；
6. 返回前K个点中出现次数（频率）最高的类别作为测试数据的预测分类。

**KNN算法中为什么要做数据归一化？**

通常来说，数据可能存在多个维度，每个维度数值之间可能存在较大差异，对于存在量纲差异的数据，数值差值最大的属性对计算结果起到决定性的影响。

**数据归一化方法：**

（1）min-max标准化：

也称为离差标准化，是对原始数据的线性变换，使结果值映射到[0 ,1]之间。

x ＝ (x - min)/(max - min) 其中，max: 样本数据的最大值

min: 样本数据的最小值

（2）Z-score标准化方法：

给予原始数据的均值（mean）和标准差（standard deviation）进行数据标准化。经过处理的数据符合标准正态分布，即均值为0，标准差为1。

x = (x - u)/σ 其中，u: 所有样本数据的均值

σ: 为所有样本数据的标准差

**K值选择对结果的影响：**

（1）K值较小，则模型复杂度较高，容易发生过拟合，学习的估计误差会增大，预测结果对近邻的实例点非常敏感。

（2）K值选择太大，模型就变得过于泛化，无法准确预测训练和测试集中的数据点，这种情况称为欠拟合。

（3）在应用中，k值一般取一个比较小的值，通常采用交叉验证法来来选取最优的K值。

（4）K值选择一个奇数，通常不超过20。

**K最近邻算法中的样本不平衡问题：**

如果数据存在严重的不平衡，预测得出的结论往往也是有偏的，即分类结果会偏向于较多观测的类。

欠采样：

将多的那一类砍掉一部分（即欠采样），但是砍掉的数据会导致某些隐含信息的丢失。

过采样：

有放回的抽样形成的简单复制，又会使模型产生过拟合。

**距离度量常见的几种方法：**

1. 欧式距离（较为常用）
2. 曼哈顿距离
3. 余弦距离

**简述K近邻算法与Kmeans算法的区别和联系？**

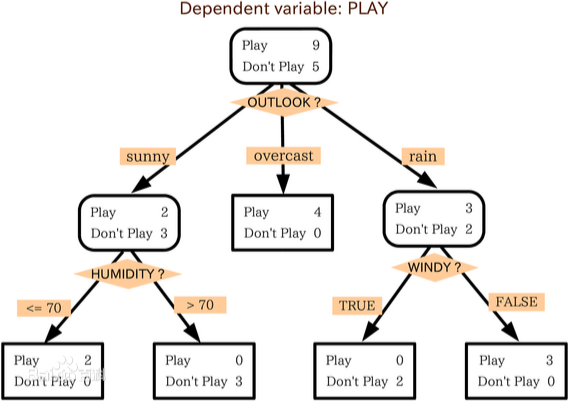
参考课堂板书笔记和实验报告部分。

**第三章 决策树算法**

**决策树基本概念**

决策树又称为判定树，它所使用的知识包含概率、期望以及信息熵。

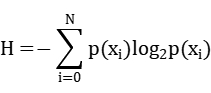
决策树类似于流程图的树结构，一般一棵决策树包含一个根结点、若干个内部结点和若干个叶结点，叶结点代表类或类分布，对应于决策结果，其他每个结点则对应于一个属性测试，根结点包含样本全集。



熵与信息熵：

在信息论与概率统计中，**熵(entropy)是表示随机变量不确定性的度量**。设X是一个取有限个值的离散随机变量，其概率分布为：

则随机变量X的熵定义为：



因此，熵只依赖于X的分布，而与X的取值无关。

熵表征的是一个东西的混乱程度、不确定性。

该物体越混乱、越不确定，它的熵越大；越整齐、越确定，熵越小，纯度也越高。

**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。

特征A对训练数据集D的信息增益g(D,A)，定义为集合D的经验熵H(D)与特征A给定条件D的经验条件熵H(D|A)之差，即：

g(D,A)=H(D)-H(D|A)

决策树学习应用信息增益准则选择特征。

具体特征选择方法为：对训练数据集（或子集）D，计算其每个特征的信息增益，并比较它们的大小，选择信息增益最大的特征。

**特征选择算法计算过程：**

学习P198-p204特征选择的计算过程，或者PPT案例题中对“天气”、“满员”、“工作”三个特征，分别计算信息增益，以及特征选择的过程。

**特征选择方法：**

**（1）ID3算法**

ID3算法的核心是在决策树各结点上应用信息增益准则选择特征，递归地构建决策树。从根结点开始，计算所有可能特征的信息增益，选择信息增益最大的特征作为结点的特征，再对子结点递归地调用以上方法。

ID3算法终止条件：（1）直到所有特征的信息增益均很小，小于设定阈值（2）没有特征可以选择。

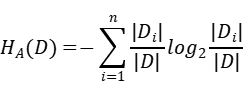
ID3算法缺陷：采用信息增益作为特征选择标准，存在偏向于选择取值较多的特征的问题。**（2）C4.5算法**

C4.5算法与ID3算法相似，C4.5算法对ID3算法进行了改进，C4.5算法在生成决策树的过程中，用信息增益比来选择特征。

信息增益比

特征A对训练数据集D的信息增益比gR(D,A)定义为其信息增益g(D,A)与训练数据集D关于特征A的值的熵之比，即：



其中，，n是特征A取值的个数。

**（3）CART算法（了解）**

分类与回归树（classifation and regession tree, CART）。

CART既可以用于分类也可以用于回归。

分类树使用基尼指数（Gini index）最小化准则进行特征选择。

回归树使用平方误差最小化准则。

概率模型**不需要归一化**,因为它们不关心变量的值，而是关心变量的分布和变量之间的条件概率，**树模型一般不需要做归一化。如决策树、随机森林等。**

机器学习算法需要做归一化处理的，如最优化问题：像K近邻算法、逻辑回归、Kmeans等。

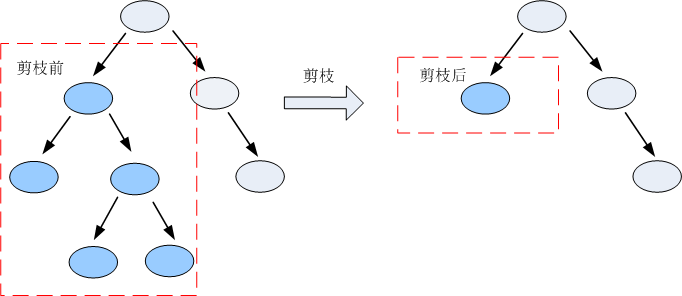
**决策树剪枝：**

决策树生成算法递归地产生决策树，直到不能继续下去为止，这样产生的树往往对训练数据的分类很准确，但对未知的测试数据的分类却没有那么准确，即出现**过拟合**现象。

**过拟合的原因**在于学习时过多地考虑如何提高对训练数据的正确分类，从而构建出**过于复杂的决策树**，对于解决新问题效果并不好。

在决策树学习中，将已经生成的树进行简化的过程称为剪枝。具体地，剪枝从已生成的树上裁掉一些子树或叶结点，并将其根节点或父结点作为新的叶结点，从而**简化分类树的模型**。

考虑极端的情况，如果我们令所有的叶子节点都只含有一个数据点，那么我们能够保证所有的训练数据都能准确分类，强化了噪声数据的作用。



两种剪枝方案：

（1）预剪枝（先剪枝）：

提前结束决策树的增长，在决策树生成的过程中，限制树的最大深度、限制叶结点上的最小样本数。

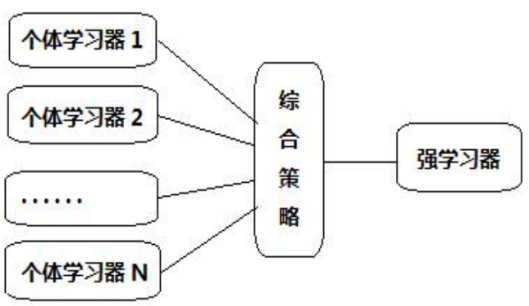
（2）后剪枝：

递归地从叶结点向上会所，根据剪枝前后判断损失函数。

**第四章 集成学习与随机森林**

**集成学习：**

通过构建并结合多个机器学习器（基学习器）来完成学习任务。基本思想：将多个弱学习器模型组合，得到一个更好更全面的强模型。“三个臭皮匠，赛过一个诸葛亮”。



**如何得到若干个个体学习器（基学习器）？**

**（1）同质学习器**

多个学习器都是一个种类的，比如各个体学习器（基学习器）都是决策树模型。如：bagging和boosting系列。

**（2）异质学习器**

所有个体学习器不全是一个种类的，比如分类问题，有的学习器采用LR，有的采用决策树，再通过某种结合策略组合。

**同质学习器按个体学习器之间是否存在依赖关系可分为两大类：**

**第一种：Bagging系列**

Bagging系列算法，从训练集进行随机抽样组成每个基模型所需要的子训练集，对所有基模型预测的结果进行综合产生最终的预测结果。

Bagging工作机制：

1. 样本随机抽取。---有放回的抽取获得k个训练集。
2. 每个训练集训练一个模型。---k个训练集训练k个模型。
3. K个模型通过综合策略得到最终结果。对于分类问题：采用k个模型的投票，少数服从多数得到最终的决策结果。对于回归问题：k个模型预测结果的均值作为最终预测结构。

**第二种Boosting系列**

每次在前一个基础模型之上，针对误分类样本提高权重，对正确分类的样本权重降低，相对于每次都针对误分类样本做了“强化训练”，类似于我们平时学习中的“错题集”。针对做错的题目，加强练习。

**Bagging与Boosting的区别：**

|  |  |  |
| --- | --- | --- |
| 区别 | Bagging系列 | Boosting系列 |
| 样本选择  策略 | 训练集是在原始数据集中有放回的抽取，各子模型的训练集是独立的，各不相同 | 每一轮训练新的子模型，训练集不变 |
| 样本权重 | 训练集中的每个样本权重相同 | 初始时各样本权重相同，后续根据错误率不断调整样本的权重，错误率越大则权重越大。 |
| 预测函数 | 所有预测函数（子模型）的权重相等。 | 每个弱分类器都有相应的权重，对于分类器误差小的分类器会有更大的权重。 |
| 并行计算 | 可以并行 | 各个预测函数只能顺序生成，因为后一个模型需要针对前面的  模型，做针对性训练和提升。串行处理。 |

**随机森林：**

采用了bagging的思想，通过多个学习器并行学习，共同预测结果。“集思广益”

随机森林有多棵决策树，每个决策树并行独立训练，预测时每个决策树都预测出一个结果，大家共同决策。

对于分类问题：多棵决策树投票少数服从多数

对于回归问题：多棵决策树求平均值

随机森林=bagging + 决策树

随机森林之所以叫做“随机”森林，是因为建立时有两个随机。

（1）每棵树的样本集随机

为了保证训练数据的相对独立性。引入了样本扰动。

（2）每棵树的属性集随机

为了保证训练过程的相对独立性。引入属性扰动。

**样本集随机**

假设原始样本集有N个样本，则每棵树从中有放回的选择N个样本作为自己的样本集，这样每棵树的样本集仍然是N个。

但由于是有放回的，所以对于一棵决策树而言，有些样本被选择了多次，有些样本没有被选择到。

既保证了每棵树样本集个数和分布的一致，又保证了每个树的样本集相对独立。

属性随机

假设每条数据有M个属性，每棵树随机选取其中的m个属性进行训练（m<<M），一般取m=logM.

保证决策树训练过程的相对独立性。

**第五章 贝叶斯算法**

**贝叶斯算法**

是机器学习中的一个重要的分支，在较小的数据集中其分类效果非常好，而且原理也十分简单。

贝叶斯定理为：

它要解决的问题就是，已知P(A|B)如何求得P(B|A)，或者已知P(B|A)如何求得P(A|B)。

**算法计算过程：**

掌握PPT中案例题的解题思路。

**朴素贝叶斯分类算法：**

是贝叶斯分类算法中最简单的一种，朴素的含义：条件概率独立性。

P(A|x1x2x3x4)=p(A|x1)\*p(A|x2)p(A|x3)p(A|x4)则为条件概率独立。

朴素贝叶斯的思想：

如果一个事物在一些属性条件发生的情况下，事物属于A的概率>属于B的概率，则判定事物属于A。

通俗举例：你在街上看到一个黑人，我让你猜他是哪里来的，你十有八九猜非洲。为什么呢？

在你的脑海中，有这么一个判断流程：

1、这个人的肤色是黑色 <特征>

2、黑人是非洲人的概率最高 <条件概率：黑色条件下是非洲人的概率>

3、没有其他辅助信息的情况下，最好的判断就是非洲人

这就是朴素贝叶斯的思想基础。

再扩展一下，假如在街上看到一个黑人讲英语，那我们是怎么去判断他来自于哪里？

提取特征：

肤色： 黑

语言： 英语

黑色人种来自非洲的概率： 80%

黑色人种来自于美国的概率：20%

讲英语的人来自于非洲的概率：10%

讲英语的人来自于美国的概率：90%

在我们的自然思维方式中，就会这样判断：

这个人来自非洲的概率：80% \* 10% = 0.08

这个人来自美国的概率：20% \* 90% =0.18

我们的判断结果就是：此人来自美国！

其蕴含的数学原理如下：

p(A|xy)=p(Axy)/p(xy)=p(Axy)/p(x)p(y)=p(A)/p(x)\*p(A)/p(y)\* p(xy)/p(xy)=p(A|x)p(A|y)

P(类别 | 特征)=P(特征 | 类别)\*P(类别) / P(特征)

**朴素贝叶斯算法步骤：**

1、分解各类先验样本数据中的特征

2、计算各类数据中，各特征的条件概率

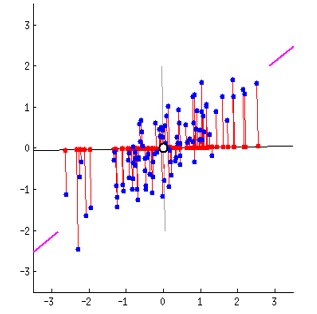
（比如：特征1出现的情况下，属于A类的概率p(A|特征1)，属于B类的概率p(B|特征1)，属于C类的概率p(C|特征1)......）

3、分解待分类数据中的特征（特征1、特征2、特征3、特征4......）

**第六章 PCA降维算法**

**PCA概念**

在实际生产生活中，我们获得的数据集在特征上往往具有很高的维度，对高维度的数据进行处理时消耗的时间很多，并且过多的特征变量也会妨碍寻找数据规律。如何在最大程度上保留数据集信息量的前提下进行数据维度的降低，是我们需要解决的问题。



一般来说，这些样本点在坐标图中任意一条向量上都可以投影成一维的，那么我们如何选择最佳投影向量呢？

需要在最大程度上保留数据集的信息量的前提下进行数据维度的降低，因此需要有优化目标来对图中的向量进行选择。

**而PCA的优化目标为：**

（1）对2维降到1维，找到一个投影方向，使得投影误差和最小；

（2）对于n维降到k维，找到k个向量定义的k维投影平面，使得投影误差和最小。

**那么投影误差又是什么？**

投影误差，即为每一个样本点到投影向量或者投影平面的距离。而投影误差和即为所有样本点到投影平面的距离的和。

**如何寻找投影误差最小的方向呢？**

寻找到方差最大的方向即可。方差最大与投影误差最小这两个优化目标其实本质上是一样的。

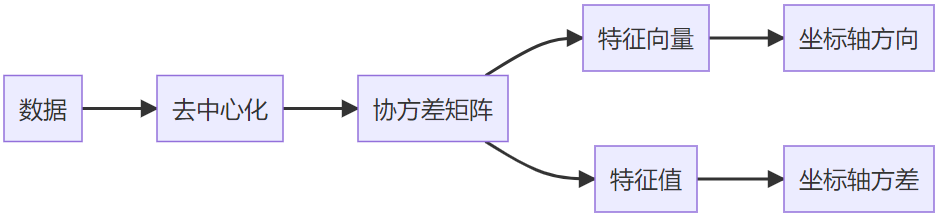
数据从原来的坐标系转换到新的坐标系，由数据本身决定。转换坐标系时，以方差最大的方向作为坐标轴方向，因为数据的最大方差给出了数据的最重要的信息。第一个新坐标选择的是原始数据中方差最大的方向，第二个新坐标轴选择的是与第一个新坐标轴正交且方差次大的方向。重复该过程，重复次数为原始数据的特征维数。

通过这种方式获得的新坐标系，大部分方差都包含在前面几个坐标轴中，后面的坐标轴所含的方差几乎为0。于是，通常可以忽略余下的坐标轴，只保留前面几个含有绝大部分方差的坐标轴。事实上，这样就相当于只保留含有绝大部分方差的维度特征，而忽略包含方差几乎为0的特征维度，也就实现了对数据特征的降维处理。

**数据中心化：**

首先要计算各特征维度的均值，然后所有数据特征对应减去各维度均值。

**PCA流程图：**



**PCA主成分数量选择：**

PCA用于对数据做降维，我们一般用PCA把m维的数据降到k维（k < m）。

K值选择的原则:

通常选择K值，比如假定能使误差小于0.01（即99%的信息被保留）或0.05（即95%的信息被保留）的k值。

对于特征值分解获得对角阵D，根据特征值从大到小排列。

PCA误差表达式：

也就是方差保留度大于99%，即：

**第七章 线性判别算法与矩阵分解算法**

**线性判别分析**

线性判别也称为Fisher线性判别，它经常被用于分类和数据预处理中的降维步骤。其最主要的使用场景是处理维数灾难而造成的过度拟合问题，少数情况下也用于处理分类问题。

线性判别就是降维，通过线性变换将高维空间的数据降到低维空间。线性判别算法最核心的知识是方差和投影。

**线性判别的本质要求：**

1. 投影后每组数据的方差足够小。
2. 投影后组与组之间的距离足够大。
3. 将数据从高维映射到低维，映射之后仍然保留明显的分类特性。

**算法对比学习：**

学习PCA和LDA部分的教程章节和相应的PPT，**理解PCA和LDA算法之间的区别和联系**。

LDA用于降维，和PCA有很多相同，也有很多不同的地方，因此值得好好的比较一下两者的降维异同点。

相同点：

1）两者均可以对数据进行降维。

2）两者在降维时均使用了矩阵特征分解的思想。

3）两者都假设数据符合高斯分布。

不同点：

1）LDA是有监督的降维方法，而PCA是无监督的降维方法

2）LDA除了可以用于降维，还可以用于分类。

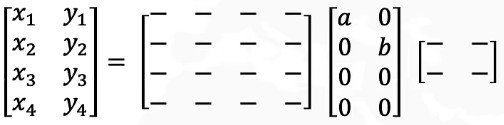
3）LDA选择分类性能最好的投影方向（确保分类信息尽可能不丢失的前提下降维），而PCA选择样本点投影具有最大方差的方向。

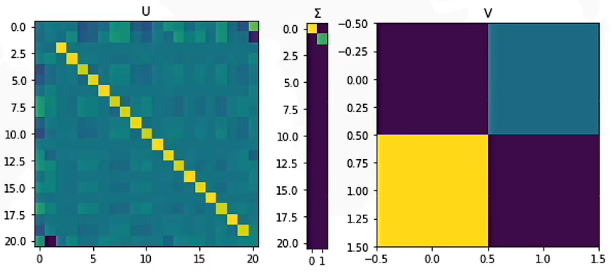
**SVD奇异值分解：**

SVD(Singular Value Decomposition)奇异值分解是一个强有力的降维工具，它经常被用在图像处理和推荐系统中，可以将SVD奇异值分解看成是PCA降维的加强版。PCA降维是压缩了特征的维度，而SVD奇异值分解不仅压缩了特征的维度，而且压缩了样本的维度。

一个矩阵可以通过矩阵分解成三个矩阵的乘积形式：



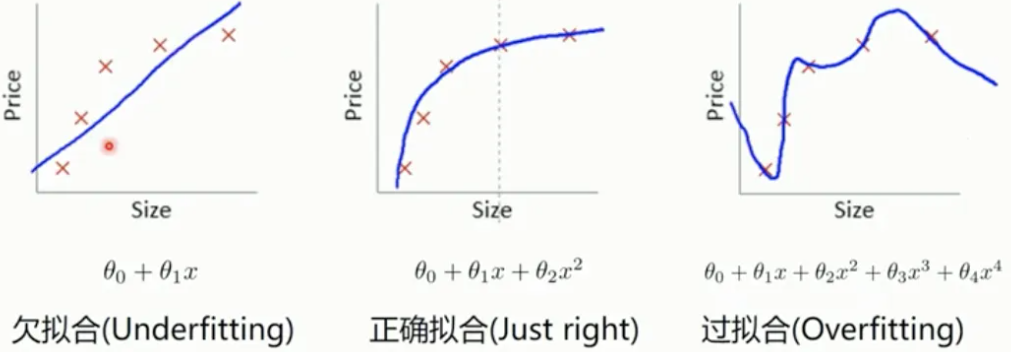




SVD奇异值分解后，S矩阵即为奇异值矩阵，奇异值代表特征的重要性。按照重要性从大到小进行排列。

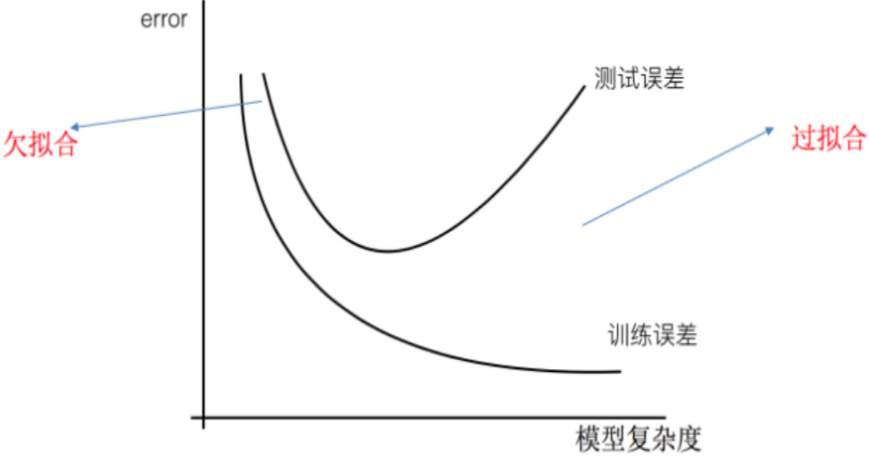
**第八章 欠拟合与过拟合以及评价指标**

**欠拟合与过拟合**



**欠拟合：**是模型过于简单，并不能很好地拟合训练数据。现象为模型在训练集上表现较差，在测试集上表现也较差。

**过拟合：**模型过于复杂，模型特征过多，导致模型参数太多，没有更多的数据来约束变量。模型千方百计地拟合训练数据，对训练集“死记硬背”，没有理解数据背后的规律，泛化能力差。现象为模型在训练集上表现很好，但是在测试集上却表现很差。



**欠拟合解决：**

模型过于简单，增加特征或构建高阶特征来增加模型复杂度。

**过拟合的原因：**

1. 训练样本不足
2. 训练数据中噪声干扰较大
3. 模型过于复杂

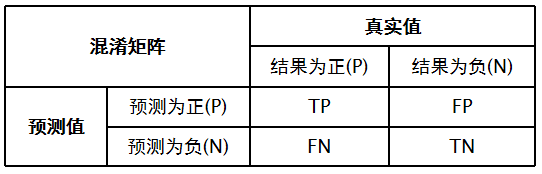
**过拟合如何解决：**

1. 增加训练数据集。
2. 采用合适的模型，适当控制模型的复杂度。
3. 减少特征数量。可以通过特征工程降低特征数量，删除冗余特征，人工保留一些有用的特征。也可以通过特征选择方法降维进行特征选择。
4. 正则化。包括L1正则化和L2正则化。其中L1正则化用于稀疏化，能够让部分特征的权重为0，从而实现特征选择的目的。L2正则化是在原始的损失函数后面加入L2正则化项，从而使得权重参数w变小，让模型变得平滑，降低模型复杂度，从而提高模型的泛化能力。

**模型评价**

**分类模型评价指标**

1. **混淆矩阵**



TP: 将正例预测为正例（预测正确）

FN: 将正例预测为负例（预测错误）

FP: 将负例预测为正例（预测错误）

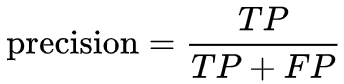
TN: 将负例预测为负例（预测正确）

其中：T、F表示预测结果是否正确

P表示Positive、N表示Negative

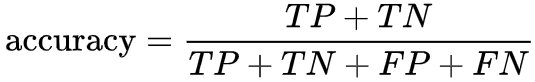
1. **精确率**

所有预测为正样本的集合中真实也为正的样本所占比例。



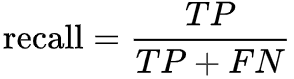
1. **准确率**

模型预测正确的结果所占的比例。



1. **召回率**

使用模型正确预测多少实际正样本。



1. **F1值**

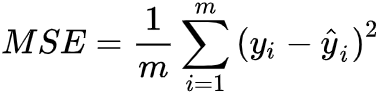
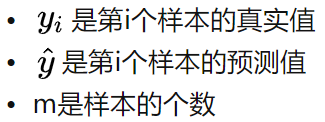
是统计学中用来衡量二分类模型精确度的一种指标。它兼顾了分类模型的准确率和召回率。F1分数看做是模型准确率和召回率的一种加权平均。当precision等于recall时，F1值最大。

F1 = 2\*(precision\*recall)/precision+recall)

**回归模型的评价指标**

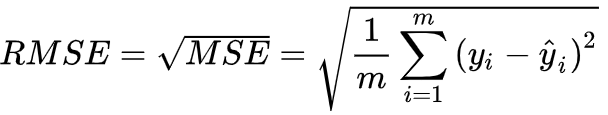
1. **均方误差(MSE)**

MSE是最常用的回归损失函数，计算方法是求预测值与真实值之间距离的平方和。

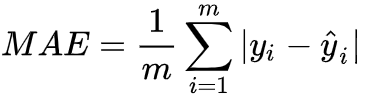
1. **平方根误差(RMSE)**

平方根误差是均方误差结果的开方。



1. **平均绝对误差(RAE)**

平均绝对误差(Mean Absolute Error)，预测值与真实值的误差绝对值的平均值。

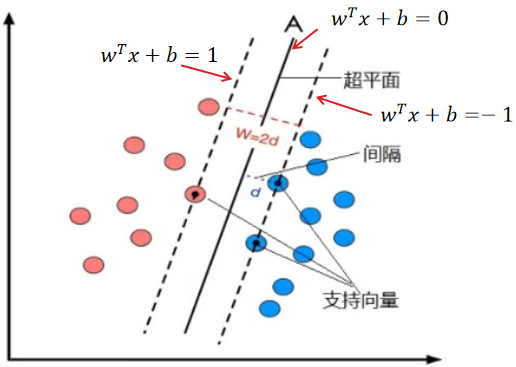


MAE对于异常值比MSE更稳定，相对于使用MAE计算损失，使用MSE的模型会赋予异常点更大的权重。

**第九章 支持向量机算法**

**支持向量机简介**

把一个数据集正确分开的超平面可能有多个，而那个具有“最大间隔”的超平面就是SVM的最优解。而这个真正的最优解对应的两侧虚线所穿过的样本点，就是SVM中的支持样本点，称为“支持向量”。支持向量到超平面的距离被称为间隔（margin）。

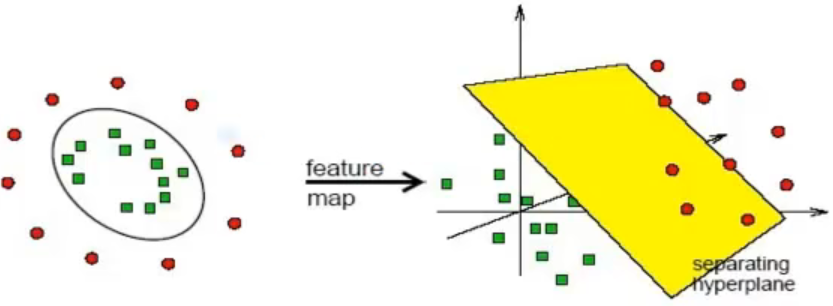


**SVM一些特性：**

训练好的模型的算法复杂度由支持向量的个数决定，而不是由数据的维度决定。所以SVM不太容易产生过拟合问题。

SVM训练出来的模型完全依赖于支持向量，即使训练集里所有非支持向量的点都被去除，重复训练过程，结果仍然会得到完全一样的模型。

**线性不可分场景：**



两个步骤来解决：

（1）利用一个非线性的映射把原数据集中的向量点转化到一个更高维度的空间中

（2）在高维度的空间中找到一个线性的超平面来按照线性可分的情况处理

**核函数：**

SVM核函数的选择对于其性能的表现至关重要，尤其是针对线性不可分的数据，核函数通过将空间内线性不可分的数据映射到一个高维的特征空间，使得数据在特征空间内是可分的。

**常用的核函数：**

（1）线性核函数（Liner）

用于线性可分的情况，参数少，速度快。对线性可分数据，分类效果好。因此通常首先尝试线性核函数，不行再尝试其他。

（2）多项式核函数（POLY）

实现将低维的输入空间映射到高维特征空间，但多项式核函数参数多，计算复杂度相对较高。

（3）径向基核函数（RBF）或 高斯核函数

是一种局部性强的核函数，其可以将一个样本映射到更高的空间内，该核函数应用广泛。对于大样本或小样本性能都较好，相对多项式核函数参数少，因此不知道如何选时优先使用高斯核函数。

（4）神经元的非线性核函数Sigmoid

采用Sigmoid核函数，支持向量机实现的就是一种多层神经网络。

**SVM扩展到解决多分类问题：**

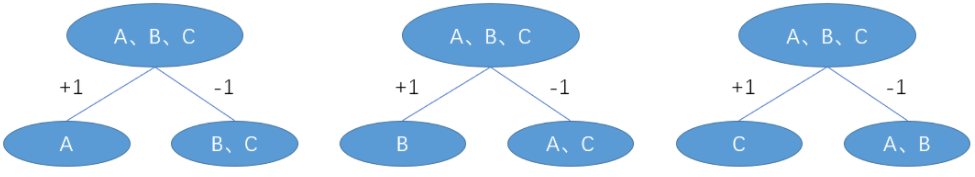
**SVM本身是一个二分类器**，最初是为二值分类问题设计的，当处理多类问题时，就需要构造合适的多类分类器。**SVM通过组合**其实也是**可以处理多分类问题**的。

如何处理？

基于一类对余类的思想，即通过组合多个二分类器来实现多分类器的构造。

在该分类方法中对n个类别仅需构造n个支持向量机‚每一个支持向量机分别将某一类的数据从其他类别中分离出来。

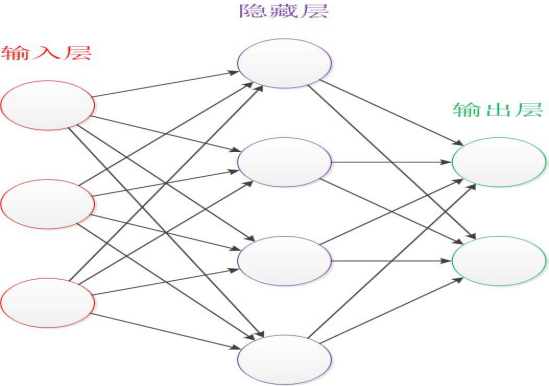
举例：使用这三个训练集分别进行训练，然后的得到三个训练结果文件。在测试的时候，把对应的测试向量分别利用这三个训练结果文件进行测试。最后每个测试都有一个结果f1(x),f2(x),f3(x)。于是最终的结果便是这三个值中最大的一个作为分类结果。



**第十章/第十一章 神经网络算法**

**神经网络简介**

经典的神经网络，包含三个层次的神经网络。红色是输入层，绿色是输出层，紫色是中间层（也叫隐藏层）。输入层有3个输入单元，隐藏层有4个单元，输出层有2个单元。

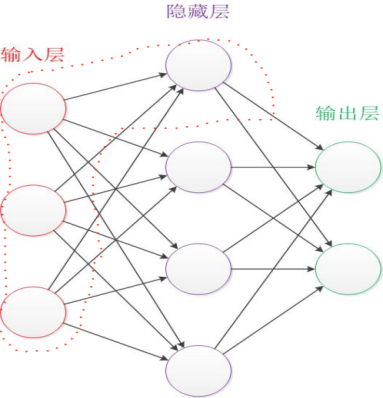


（1）**输入层和输出层的节点往往是固定的**，输入层节点数取决于特征的数量，输出层取决于业务本身，是二分类问题还是多分类问题，从而设定一定数量的输出层节点，**中间层可以自由设定**

（2）结构图中的拓扑与箭头代表着预测过程时数据的流向

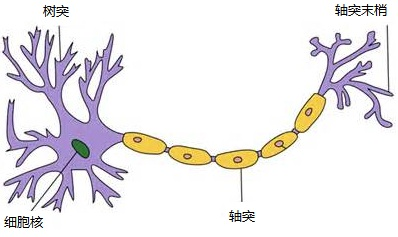
（3）结构图中的关键不是圆圈（神经元），而是连接线。每个连接线对应一个不同的权重（即权值），由训练得到。

**神经网络与逻辑回归的关系：**

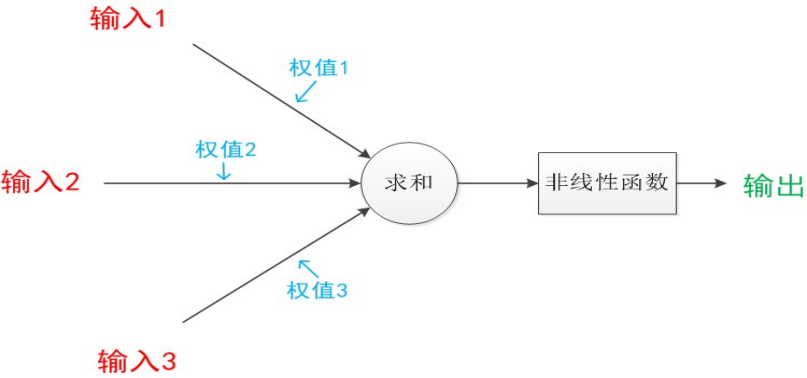
 

**神经元：**

一个神经元通常具有**多个树突**，主要用来接收传入信息；**一个轴突**，轴突尾端有许多轴突末梢可以给其他神经元传递信息。轴突末梢跟其他神经元的树突产生链接，从而传递信号。这个连接的位置在生物学上叫做“突触”。



神经网络的训练算法就是让权重的值调整到最佳，以使得整个网络预测效果最好。



**神经网络正向过程和反向传播过程：**

参考PPT，熟练掌握正向过程和反向传播的详细过程。

正向传播的作用：计算整个网络预测输出。

反向传播的作用：将最终的预测值和真实值之间的误差反向传播，更新网络权重参数。

链式法则：查看PPT理解掌握。

梯度消失：查看PPT理解掌握。

**激活函数的作用是什么？**

神经网络中，不使用激活函数的话，神经网络的每层都只能做线性变换，多层输入叠加后也还是线性变换。因为线性模型的表达能力通常不够，所以需要引入非线性函数，足够多的非线性函数组合可以近似表达任意复杂的函数。所以通过激活函数来引入非线性。

**神经网络的超参数：**

（1）学习率。如果学习率很低，训练会变得更加可靠，但优化耗时较长；如果学习率很高，训练可能根本不会收敛。

（2）层数。如果层数比较多，即为所谓的深度学习神经网络。不过，并不是越深的神经网络效果就越好，因为过多的层数可能会导致梯度消失。

（3）每层节点的个数。设置每层神经节点的个数。

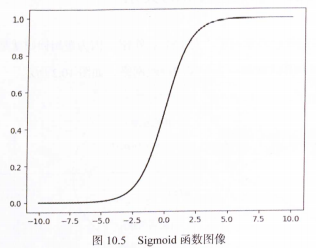
（4）设置节点的激活函数。一般在隐藏层设置为ReLU激活函数，因为该函数比Sigmoid函数和tanh函数的学习效果更好。而在输出层则需要根据最后需要的结果来选择函数。

**神经网络中有哪些激活函数？结合函数图像说说有分别有什么特点？**

（1）Sigmoid 函数

Sigmoid函数定义如下：





第一点：Sigmoid函数可以将任意实数映射到0-1之间，表达概率。

第二点：求导方便，g(z)的导数等于g(z)(1-g(z))

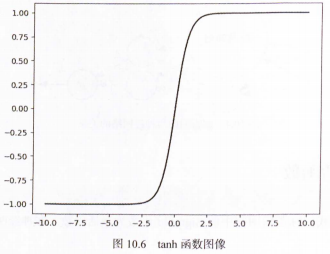
第三点：梯度消失问题，Sigmoid函数在0点附近导数最大，最大值为1/4，远离坐标远点时，导数非常小，引起梯度消失问题。

第四点：不以零点为中心。

（2）tanh 函数

tanh函数的定义如下：





tanh 函数将拟合曲线最后的结果转换到（－1, 1)的区间上。比较大的负数无限接近于－1,比较大的正数无限接近于1。

第一点：将任意实数映射到-1到1的区间内

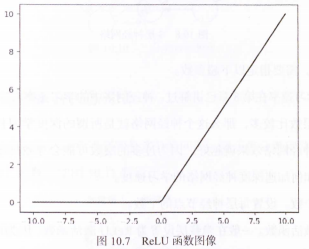
第二点：以零为中心

第三点：在零点附近，梯度较大，远离零点，也会出现梯度消失问题

（3）ReLU函数

ReLU函数的定义如下：





RelU函数将拟合曲线最后的结果转换到［0,+∞)的区间上，其中小于0的结果转换为0,大于0的结果不变。因为Sigmoid函数和tanh函数在训练神经网络的过程中效率比较低，所以现在都默认使用ReLU函数进行训练。

第一点：解决了梯度消失问题。

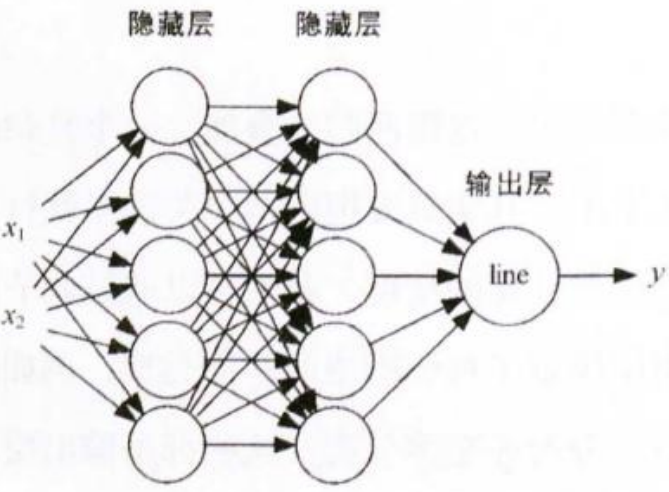
第二点：梯求导方便，计算速度快。

第三点：不以零点为中心。

第四点：负半轴死亡。

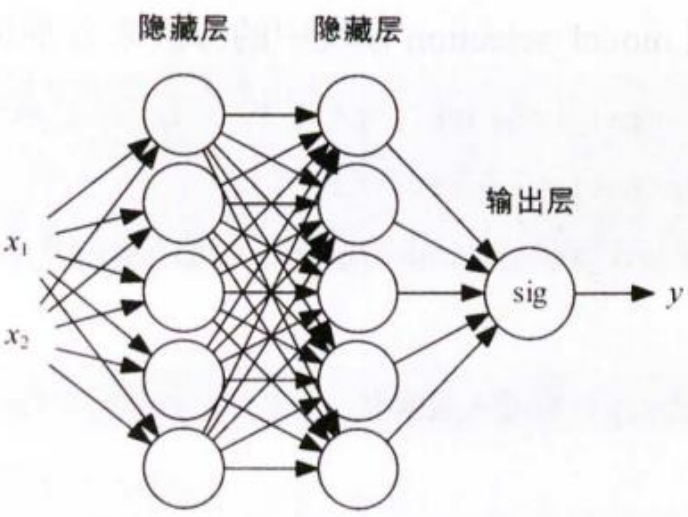
**神经网络应用于回归问题：**

输出层使用线性，也就是不需要设置任何的激活函数。



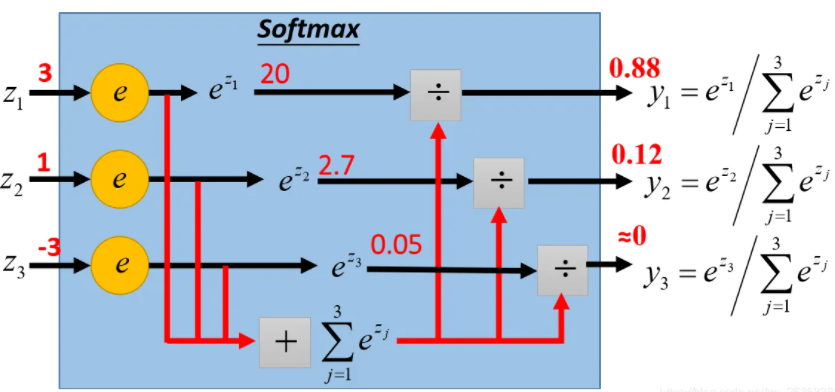
**神经网络应用于二分类问题：**

输出层使用Sigmoid函数。



**神经网络应用于多分类问题：**

输出层使用Softmax函数。softmax函数又称为归一化指数函数，将多个神经元的输出映射到(0,1)区间内，并且累计和为1。可理解为概率，从而进行多分类。



softmax直白来说就是将原来输出是3,1,-3的数，通过softmax函数映射成为(0,1)的值，而这些值的累和为1（满足概率的性质），那么我们就可以将它理解成概率，在最后选取输出结点的时候，我们就可以选取概率最大（也就是值对应最大的）结点，作为我们的**预测目标。**

=========================== 补充 ===========================

**梯度下降法与线性回归**

**梯度下降法的形象理解**

梯度下降可理解为你站在山的某处，想要下山，此时最快的下山方式就是你环顾四周，哪里最陡峭，朝哪里走一步，一直执行该策略，在第N个循环后，你就到达了山的最低处。

**学习率对梯度下降法的影响：**

该部分学习教材P134-142，深刻理解梯度下降法的计算过程，以及学习率选择的不同对结果产生的影响。

**梯度下降法优化的巧妙之处：**

（1）当学习率较小时，即便初始点位置距离最小值点较远，但是迭代足够多次，均能到达最优解附近。

（2）初始点位置可以在任意位置，正半轴或负半轴均可。

（3）距离最小值点较远时，导数较大，参数更新较快；当距离最小值点较近时，导数较小，参数更新变慢，该特性可以保证迭代优化又快又稳。

**线性回归：**

定义(x(i), y(i))为第i个样本，X表示输入值，y表示输出值，下标表示样本的序号。

复习一下机器学习过程：



线性回归（linear regression）是一种线性模型，它假设输入变量x和单个输出变量y之间存在线性关系。

一般地，假设有一组二维数据，我们可以用一条直线描述，用线性函数来拟合这个模型。即：

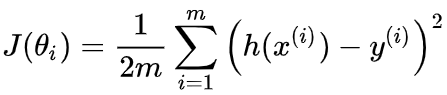
其中，和为模型参数，和学得之后，模型就得以确定。

设想：对于训练集(x,y),选择参数和使得尽可能的接近y。

如何做？

计算训练集的平方误差函数（Square error function）。基于预测值和真实值的均方差最小化的方法来估计线性回归学习器的参数。其中，平方误差损失也称为最小二乘法。

代价函数（损失函数）：



其中，m:样本数， 为预测值，为实际值，i为样本序号。

学习PPT中线性回归算法的求解过程，熟练掌握损失函数的表达公式及代表的含义，掌握线性回归利用梯度下降法优化求解的过程。

**逻辑回归**

**基本概念**

虽然**线性回归**算法和**逻辑回归**算法都有回归一词，但是二者的理论内容是截然不同的。前者**解决回归问题**，后者**解决分类问题**。

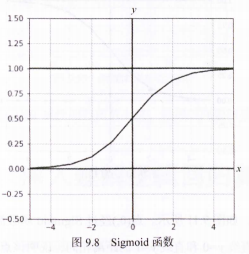
逻辑回归包含了两个部分，第一部分是线性部分，第二部分是非线性Sigmoid部分，也就是将线性回归部分的结果，通过sigmoid进行非线性映射，将线性部分任意实数的值域范围映射到0-1之间，表示概率的概念，从而解决分类问题。

**逻辑函数**

逻辑函数（Logistic Function)又称为Sigmoid函数。



它的特性是所有值都在(0, 1)之间，如下图。



逻辑函数的作用是判断不同属性的样本属于某个类别的概率。在二分类过程中，用1表示正向的类别，用0表示负向的类别，也就是说经过Sigmoid函数转换，如果值越靠近1则其属于正向类别的概率越大；如果值越靠近0,则其属于负向类别的概率越大。

学习PPT，熟练掌握逻辑回归的算法过程，掌握逻辑回归为什么使用交叉熵损失函数，而不是使用平方误差损失函数，熟练掌握两者的差异，结合代码理解损失函数的含义和区别。

**线性回归与逻辑回归对比：**

结合实验报告和代码，理解线性回归与逻辑回归之间的区别和联系。

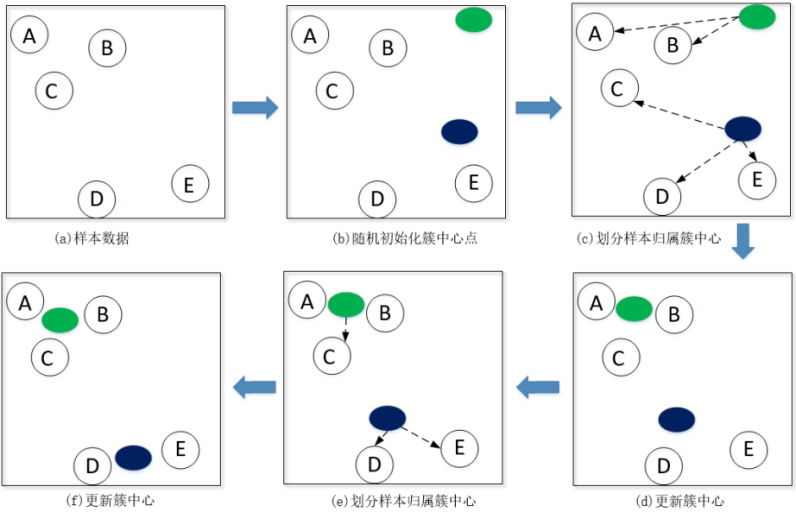
**聚类算法**

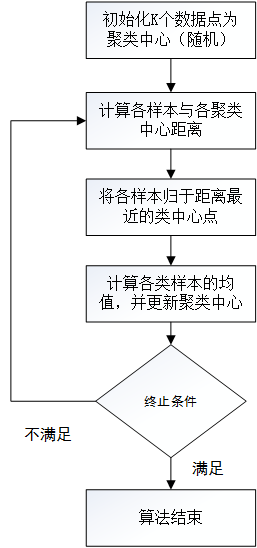
全部的无标签数据，将相近的数据聚成同一类。簇内距离最小化，簇间距离最大化。

常用聚类算法

Kmeans（最常用）、层次聚类、密度聚类等。

kmeans算法基本过程：





**Kmeans算法步骤：**

（1）K值是Kmeans最重要的选择参数，直接决定着数据聚类的类别数量，在选择K值后，会在数据中随机选择K个数据样本最为初始中心点，如K=2；

（2）计算所有样本点到各中心点距离，将其划分到距离最近中心点的类别；

（3）每个类别中有了两个以上的数据时，类的中心就会发生变化，因此类中一旦有新的数据被划入时就需要重新计算整个类的中心点，这一步的计算也是整个算法的核心，因此称为K均值算法；

（4）通过几步计算之后的结果，能够更直观的展示出类的聚合情况和中心点的位置情况；

（5）判断聚类过程结束的标准有两个，一是中心点的位置不再发生变化，即结果收敛；二是执行了足够多次的迭代次数（通俗可以理解为计算了几次中心点位置）。

**算法对比理解：**

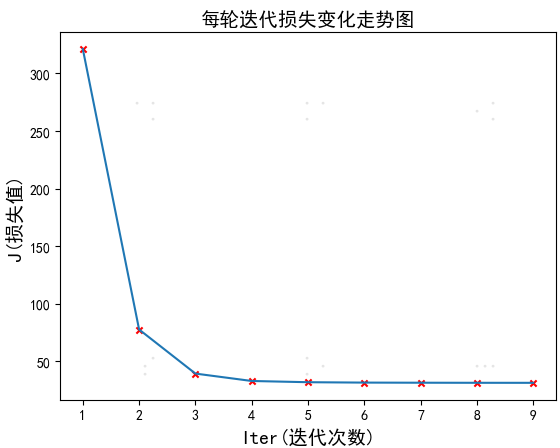
Kmeans 算法和KNN 算法的区别和联系。

**算法计算过程：**

熟练掌握PPT中Kmeans算法的案例题的解题过程。

**Kmeans算法的关键技术：**

**检验算法是否正常工作，通过折线图**



**终止条件**

（1）聚类中心不再变动

（2）达到设定的迭代次数

（3）前后平均误差小于设定的阈值

**簇中心初始化**

Kmeans算法中对簇中心初始化的不同，结果可能相差较大，极端的初始化结果，可能导致算法不能正常工作。

簇中心初始化方法：

（1）初始聚类中心点设置为彼此尽可能远的样本点

（2）首先所有样本中靠近中心的样本点作为第一个聚类中心点，然后选择距离第一个中心点最远的样本点作为第二个中心点，然后再选择距离前两个点的最短距离最大的那个点作为第三个初始类聚类中心点，以此类推，直至选出K个初始类簇中心点（通用场景）。

（3）随机初始化选择多次，取算法迭代损失值最小的（适用于K较小场景）

**K值选择**

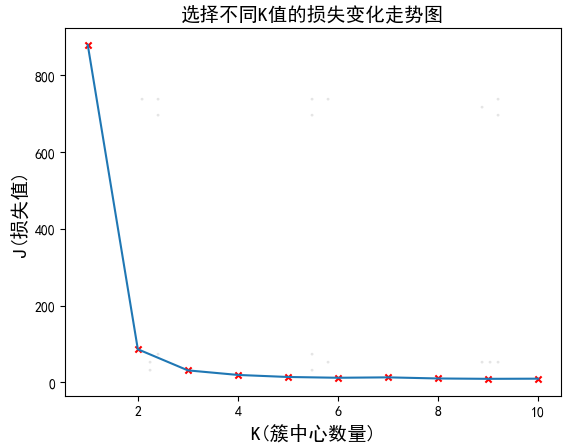
聚类的K值，通常根据业务场景人工确定。当业务目标不明确时，则借助如下几种方法来辅助选择。

（1）肘部法则

（2）轮廓系数

（3）Canopy算法

**最常使用的是肘部法则：**



**Apriori算法**

**关联分析**

关联分析又称关联挖掘，即从大规模数据中，发现对象之间隐含关系与规律的过程，也称为关联规则学习。

**关联关系的两种形式：**

（1）频繁项集：经常出现在一块的物品的集合。

（2）关联规则：暗示两种物品之间可能存在很强的关系。

**关联分析的三大指标：**

**（1）支持度**

支持度是个百分比，指某商品或商品组合出现的次数与总次数之间的比例。支持度越高，代表这个组合出现的频率越大。

**（2）置信度**

置信度即条件概率，也称为可信度。指的是当你购买了商品A，会有多大的概率购买商品B。

表示在发生X的项集中，同时发生Y的可能性，即X和Y同时发生的个数占仅X发生个数的比例，公式为：Confidence(X→Y) = P(Y|X) = P(X,Y)/P(X) = P(X∩Y)/P(X)

**（3）提升度**

提升度，指的是商品A的出现，对商品B的出现概率提升的程度。

Lift(X→Y) = P(Y|X)/P(Y)

提升度反映了关联规则中的X与Y的相关性。

提升度>1且越高表明正相关性越高

提升度<1且越低表明负相关性越高

提升度=1表明没有相关性，即相互独立。

提升度要求至少大于1。

因为关联分析主要目的是寻找频繁项集，如果通过暴力寻找，结果是几何性增长运算量大。对于较大数据集，如果每个项集都要这么遍历的话效率太低。 为减少频繁项集的计算引入了Apriori算法。

**Apriori算法要点：**

1. 如果某个项集是频繁的，那么它的所有子集也是频繁的
2. 如果一个项集是非频繁集，那么它的所有超集也是非频繁
3. 在寻找关联规则中，如果前项可信度较低，那么其后项的可信度也会低