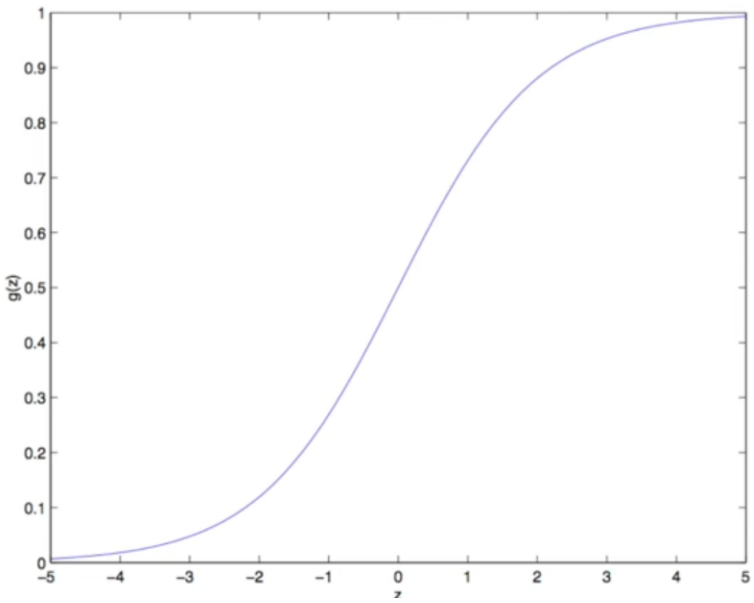
机器学习算法

- 逻辑回归
  - 什么是逻辑回归：
    - LR是Logistic Regression Classifier，本质上是线性回归，特殊之处在于特征到结果的映射加入了一层逻辑函数g(z)，即先把特征线性求和，然后使用函数g(z)作为假设函数来预测。g(z)可以讲连续值映射到0到1.逻辑回归使用的g(z)函数是sigmoid函数。因此逻辑回归=线性回归 + sigmoid。
    - 逻辑回归的表达式为：$sigma(w^T x) = \frac{1}{1+e^{-w^T x}}$；图像如下



  - 逻辑回归的优势
    - 他是直接对分类可能性进行建模，无需实现假设数据分布，这样就避免了假设分布不准确所带来的问题；
    - 它不是仅预测处“类别”，而是可得到近似概率预测，这对许多需利用概率辅助决策的任务很有用；
    - 逻辑回归函数是任意阶可导的凸函数，有很多好的数学性质，先有的许多数值优化算法都可直接用于求解最优解。
    - 对于线性数据，大部分时候逻辑回归的拟合和计算都非常快，计算效率优于SVM和随机森林
  - 逻辑回归推导

- 假设数据集为 $Data:(x_i,y_i)_{i=1}^N\ x_i\in\mathbb{R}^p,y_i\in 0,1$ sigmoid函数为 $sigmoid:\sigma(z)=\frac{1}{1+e^{-z}}$ 在线性回归中有 $y=w^Tx+b$为了方便，我们将其中的权值向量w和输入向量x进行扩充，即$w=(w_1,w_2,...,w_n,b);\ x=(x_1,x_2,...,x_n,1)$，则式(3)可以改写为$y=w^Tx$
- 线性回归是将向量x映射为具体的数值y（连续），而逻辑回归是用来解决分类问题（通常为二分类问题），希望得到0或1的概率（区间$[0,1]$），即通过某种方式将数值y映射到区间$[0,1]$范围内。逻辑回归采用sigmoid函数来完成这样的映射，从而建立y与x之间的概率判别模型 $P(Y|X)$ 有 $p_1=P(y=1|x)=\frac{1}{1+e^{-(w^Tx)}}$；PS：在x的特征下是y类别大概率
- $p_0=P(y=0|x)=1-P(y=1|x)=\frac{e^{-(w^Tx)}}{1+e^{-(w^Tx)}}$
- 得到 $P(Y|X)=p_1^Yp_0^{1-Y},Y\in 0,1$ 对应的似然函数为 $\prod_{i=1}^N P(y_i|x_i)$ 取对数，得到对数似然函数

$$
\begin{aligned}
L(w)&=\sum_{i=1}^N\log P(y_i|x_i)\\
&=\sum_{i=1}^N(y_i\log p_1+(1-y_i)\log p_0)\\
&=\sum_{i=1}^N(y_i\log p_1+(1-y_i)\log(1-p_1))\\
&=\sum_{i=1}^N(y_i(\log p_1-\log(1-p_1))+\log(1-p_1))\\
&=\sum_{i=1}^N(y_i\log(\frac{p_1}{1-p_1})+\log(1-p_1))\\
&=\sum_{i=1}^N(y_i(w^Tx_i)+\log\frac{e^{-(w^Tx_i)}}{1+e^{-(w^Tx_i)}})\\
&=\sum_{i=1}^N(y_i(w^Tx_i)+\log\frac{1}{1+e^{w^Tx_i}})\\
&=\sum_{i=1}^N(y_i(w^Tx_i)-\log(1+e^{w^Tx_i}))
\end{aligned}
$$

- 对$L(w)$求极大值（即极大似然估计），即可得到w的估计值 $\hat{w}=\arg\max_w L(w)$
- 这样，问题就变成了以对数似然函数为目标的最优化问题，可采用梯度下降法或拟牛顿法。

- ps：最大似然估计理解：
  - 假如有一个罐子，里面有黑白两种颜色的球，数目多少不知，两种颜色的比例也不知。我 们想知道罐中白球和黑球的比例，但我们不能把罐中的球全部拿出来数。现在我们可以每次任意从已经摇匀的罐中拿一个球出来，记录球的颜色，然后把拿出来的球 再放回罐中。这个过程可以重复，我们可以用记录的球的颜色来估计罐中黑白球的比例。假如在前面的一百次重复记录中，有七十次是白球，请问罐中白球所占的比例最有可能是多少?
  - **很多人马上就有答案了：70%。而其后的理论支撑是什么呢?**
  - 我们假设罐中白球的比例是p，那么黑球的比例就是1-p。因为每抽一个球出来，在记录颜色之后，我们把抽出的球放回了罐中并摇匀，**所以每次抽出来的球的颜 色服从同一独立分布。**
  - 这里我们把一次抽出来球的颜色称为一次抽样。题目中在一百次抽样中，七十次是白球的,三十次为黑球事件的概率是P(样本结果|Model)。
  - 如果第一次抽象的结果记为x1,第二次抽样的结果记为x2....那么样本结果为(x1,x2.....,x100)。这样，我们可以得到如下表达式：
  - P(样本结果|Model)
  - = P(x1,x2,…,x100|Model)
  - = P(x1|Mel)P(x2|M)…P(x100|M)
  - = p^70(1-p)^30.
  - 好的，我们已经有了观察样本结果出现的概率表达式了。那么我们要求的模型的参数，也就是求的式中的p。
  - 那么我们怎么来求这个p呢？不同的p，直接导致P（样本结果|Model）的不同。
  - 好的，我们的p实际上是有无数多种分布的。p=50%那么求出 p^70(1-p)^30为 7.8 * 10^(-31)

- p的分布也可以是如下：p=70%那么也可以求出p^70(1-p)^30为2.95* 10^(-27)
- **那么问题来了，既然有无数种分布可以选择，极大似然估计应该按照什么原则去选取这个分布呢?**
- **答：采取的方法是让这个样本结果出现的可能性最大，也就是使得p^70(1-p)^30值最大，那么我们就可以看成是p的方程，求导即可!**
- 那么既然事情已经发生了，为什么不让这个出现的结果的可能性最大呢？**这也就是最大似然估计的核心。**
- **我们想办法让观察样本出现的概率最大，转换为数学问题就是使得:**
- p^70(1-p)^30最大，这太简单了，未知数只有一个p，我们令其导数为0，即可求出p为70%，与我们一开始认为的70%是一致的。其中蕴含着我们的数学思想在里面。

- 由此理解上述计算过程，逻辑回归就没有求解p的过程，而是求解$P(Y|X) = p_1^Y p_0^{1-Y}, Y \in 0,1$的似然函数$\prod_{i=1}^{N} P(y_i|x_i)$最大值求解，那么就是对$P(y_i|x_i)$求导然后取等于0值，因为$p_1 = P(y=1|x) = \frac{1}{1+e^{-(w^T x)}}$，实际上也就是对$y = w^T x$的$w$求导。
- 理解整个逻辑回归过程，也就是给定相应的特征X和标签Y，然后有多少样本就代表对**整个该类别样本**（这个是高维度概念，就是世界上所有的样本组成的，实际上达不到那种，模型就是模拟这个空间）**空间**取多少次样本，然后构成(x1,x2.....,x100,...xn)，然后求解这个模型取出这么些样本且对应这些标签的最大可能性的参数，也就是似然函数$\prod_{i=1}^{N} P(y_i|x_i)$的最大可能性的$w$参数。跟上面一个例子不一样的是，逻辑回归要经过一个sigmoid函数。
- 简而言之，**模型要最好拟合训练集样本空间，就是求解似然函数的最大值。**这是一个判别模型$P(Y|X)$
- 求解优化：
- 令

$$g(w^T x_i) = \frac{1}{1+e^{-(w^T x_i)}}$$

- 此时，

$$\begin{aligned}\frac{\partial}{\partial w_j}L(w) &= \sum_{i=1}^{N}(y_i\frac{1}{g(w^T x_i)} - (1-y_i)\frac{1}{1-g(w^T x_i)})\frac{\partial}{\partial w_j}g(w^T x_i) \\ &= \sum_{i=1}^{N}(y_i\frac{1}{g(w^T x_i)} - (1-y_i)\frac{1}{1-g(w^T x_i)})g(w^T x_i)(1-g(w^T x_i))\frac{\partial}{\partial w_j}w^T x_i) \\ &= \sum_{i=1}^{N}(y_i(1-g(w^T x_i)) - (1-y)g(w^T x_i))x_i^j \\ &= \sum_{i=1}^{N}(y_i - g(w^T x_i))x_i^j\end{aligned}$$

- 因为这里是求最大值，采用梯度上升法：

$$w_j := w_j + \alpha(\sum_{i=1}^{N}(y_i - g(w^T x_i))x_i^j)$$

- 支持向量机
  - 什么是SVM?
    - 数学模型：支持向量机的提出基于统计学习理论和结构风险最小化准则，统计学习理论避免分类模型受样本量的限制，结构风险最小化准则避免模型训练时出现的模型问题。在这样的背景下，支持向量机技术的推广及其判别能力较强，其最终的目的是根据输入数据样本寻找到一个最优分类超平面。
    - 支持向量机最优分类面的求解问题可转化为求数据样本分类间隔最大化的二次函数的解，关键是求得分类间隔最大值的目标解。以两类线性可分数据为例，一类数据用圆形表示，另一类数据用菱形代表，则最优分类线示例图如下图所示：

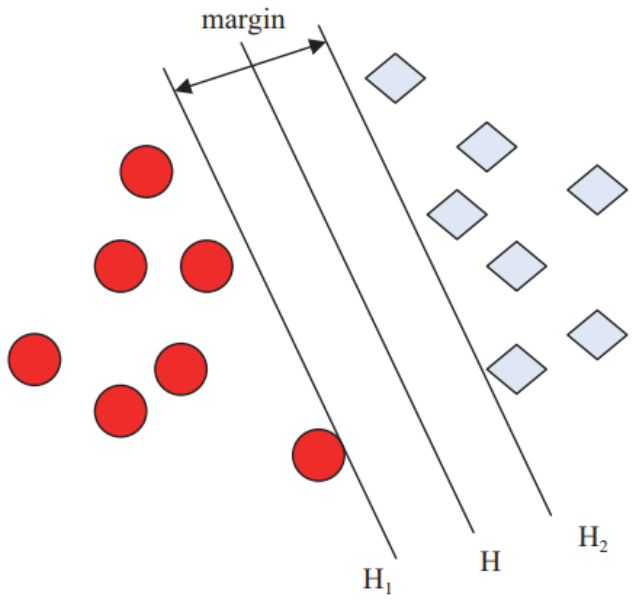


图 1　最优分类线示例图

- 图1中， margin代表分类平面间的最大分类间隔，处于分类线两侧的数据点为待分类的样本。在该例图中，基本分类判别面方程如公式(1)所示，若对线性可分的样本集进行数据归一化处理，分类间隔表达式如公式(2)所示。

$$w^{\mathrm{T}}x+b=0 \qquad (1)$$

$$2/\|w\| \qquad (2)$$

- 基于上述分析，通过加入有效约束条件，引入拉格朗日乘子后，解得最优分类判别函数，且其参数的确定依赖于支持向量。实际应用中，核函数结合最优分类判别面形成的支持向量机模型解决了其只处理线性可分样本的弊端，两者结合形成最终的支持向量机模型。相应的通用支持向量机分类函数表达式如公式(3)所示。公式(3)中, a*i和b*i是调控支持向量机确定最优分类平面的参数。

$$f(x)=\operatorname{sgn}\left\{\sum_{i=1}^{k}a_i^*y_iK(x_i\cdot x)+b^*\right\} \qquad (3)$$

- 核支持向量机
  - 核函数的价值在于它虽然也是将特征进行从低维到高维的转换，但核函数事先在低纬度进行计算，而将实质上

的分类效果表现在了高维度上，也就避免了直接在高维度空间中的复杂计算。

- 原本从低维空间映射到高维空间的时候，是先映射到高维再进行内积计算，但是核函数是在低纬度进行计算，而不需要显式地写出映射后的结果。
- 支持向量机是基于两类线性可分的样本数据发展而来，但是在实际应用中，需要识别和分类的数据大多数情况下都处于非线性不可分状态，并非理想化状态。由此，研究人员设计一个核函数应用于支持向量机的分类过程中解决该问题，其主要目的是将原低维空间中非线性不可分数据映射到高维空间中，即解决低维特征空间无法构造分类超平面的问题。支持向量机的应用性能关键在于核函数方法的选取。核函数方法计算公式如下所示:
公式(4)表示在特征空间直接计算内积，ϕ代表x映射到内积特征空间的过程。研究人员在解决不同的数据分类问题的时候需选择不同的参数，简单来说就是选择不同的核函数。核函数主要分为线性核、多项式核、Sigmoid核和Gauss径向基核。

$$K(X,Z)=\langle\phi(x)\cdot\phi(z)\rangle \qquad (4)$$

- (1)**线性核**：公式(5)代表数据所处的原空间中的内积计算。其作用是统一两空间数据形式，即数据处于原空间的形式与数据经映射后所处空间的形式。ps：就是经过一个线性全连接层将数据转化成高维数据。

$$K(x_i,x)=x_i^{\mathrm{T}}x \qquad (5)$$

- 这实际上就是原始空间中的内积。这个核存在的主要目的是使得“映射空间中的问题”核“映射前空间中的问题”两者在形式上统一起来，意思是说，咱们有的时候，写代码，或写公式的时候，只要写个模版或通用表达式，然后再代入不同的核，便可以了，于此，便在形式上统一了起来，不同再分别写一个线性，核一个飞线性的。

- (2)**多项式核**：公式(6)代表多项式空间中的内积计算,注重数据的全局性。其计算过程不同于线性核,这是由于直接在多项式空间计算会造成维数灾难，所以其计算包含一个转换过程,即从高维空间转到低维空间，利用低维空间计算其内积值。

$$K(x_i, x) = \left(\gamma x_i^{\mathrm{T}} x + r\right)^d \qquad (6)$$

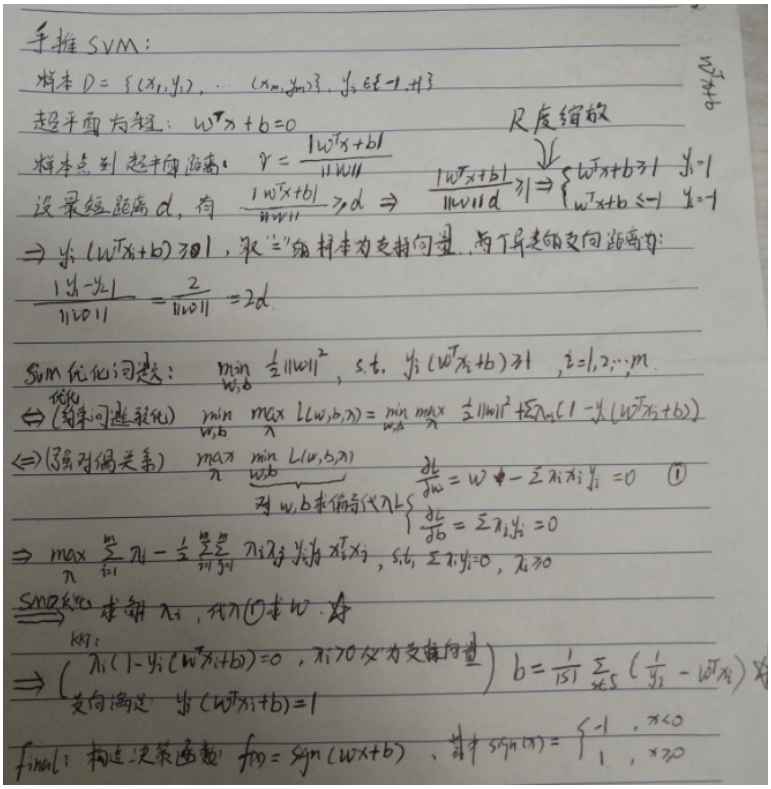
- **(3) Sigmoid核**：公式(7)实现将 Sigmoid函数作为核函数，其近似为多层感知器神经网络，注重样本数据的全局最优值。

$$K(x_i, x) = \tanh\left(\gamma x_i^{\mathrm{T}} x + r\right) \qquad (7)$$

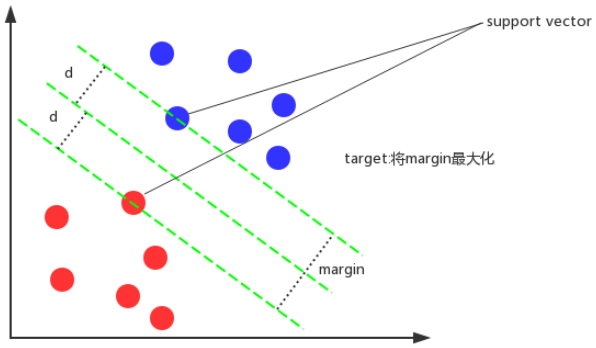
- **(4)Gauss径向基核(RBF)**：公式(8)可将原始特征空间映射到无穷维特征空间中，其性能好坏在于y参数的调控,局部性较强。y参数选取的值较小,映射后的特征空间近似一个低维空间；y参数选取的值较大,易造成过拟合问题.正因为其具有较强的可调控性,其在实际应用中更为广泛。实际应用中，研究者通过权衡各个核函数的优势与劣势，通常将最佳的核函数应用于特定数据分类领域中。

$$K(x_i, x) = \exp\left(-\gamma \|x_i - x\|^2\right) \qquad (8)$$

- **面试常见题：**
  - 【机器学习】支持向量机 SVM（非常详细） - 知乎 (zhihu.com)多看几次，很全的SVM
  - **那你讲一下SVM吧/ 讲一下SVM的推导/ 手推一下SVM吧**
    - hard margin， soft margin， 几何距离、函数距离，最优化，拉格朗日乘子法，对偶问题，KKT条件，损失函数，惩罚系数等。



- 手推SVM：
  - 首先SVM的意义实际在空间中找到一个超平面，把空间点中的两个线性可分的点集完全分开，我们把这个超平面定义成$w \cdot x_i + b = 0$，其中w是一个n维的向量x_i表示空间中的点。与此同时，我们不仅需要用超平面将空间中的两个点集分开，同时需要超平面两侧距离超平面最近的样本点到超平面的距离最大化。



  - 从二维平面可以看出，我们最终的目的就是最大化margin，两个类别中存在距离超平面最近的点我们称作支持向量，支持向量所在的于超平面平行的两个平面之间的距离被称为margin。在n维空

间中，点x到直线$w \cdot x_i + b = 0$的距离为$\frac{|wTx+b|}{\|w\|}$，其中$\|w\| = \sqrt{w_1^2 + w_2^2 + ... + w_d^2}$，根据定义每个类别到超平面最近的点和超平面的距离为d，那么这个类别其他所有的点到超平面的距离都是大于d，可以得到如下的式子：

$$\begin{cases} \frac{w^T x_i + b}{\|w\|} \geqslant d & y_i = 1 \\ \\ \frac{w^T x_i + b}{\|w\|} \leqslant -d & y_i = 1 \end{cases}$$

- 同时也可以知道

$$\begin{cases} w^T x_i + b > 0 & y_i = 1 \\ \\ w^T x_i + b < 0 & y_i = -1 \end{cases}$$

- 所以可以得到

$$y_i(w^T x_i + b) > 0$$

- 因为margin就是最小的距离，所以可以得到以下式子：

$$\max \quad margin = \max \quad \min \frac{|w^T x_i + b|}{\|w\|}; \qquad s.t. \quad y_i(w^T x_i + b) > 0$$

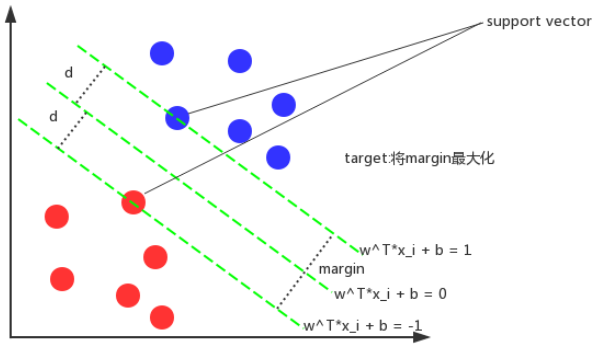
- 上述式子等价于：

$$\max \quad margin = \max \quad \min \frac{(w^T x_i + b) y_i}{\|w\|}; \qquad s.t. \quad y_i(w^T x_i + b) > 0$$

- 我们可以将上述公式拆开来看：

$$\max \quad margin = \max \frac{1}{\|w\|} min(w^T x_i + b) y_i; \qquad s.t. \quad y_i(w^T x_i + b) > 0$$

- 我们再来看看分类的图，我们要将所有的点分成+1和-1两个类，那么如图所示：



- 1

由上图可知，所有的点一定会在平面$w^T x_i + b = 1$和$w^T x_i + b = -1$两侧，所以$\min y_i(w^T x_i + b) = 1$，再结合上述公式我们可以得到：

$$\max margin = \max \frac{1}{\|w\|}; \quad s.t. \quad y_i(w^T x_i + b) \geqslant 1$$

这个问题同样可以等价于：

$$\max marg\, in = \min \frac{1}{2}\|w\|^2; \quad s.t. \quad y_i(w^T x_i + b) \geqslant 1$$

原来的目标函数为：

$$\max \quad margin = \min_{w,b} \quad \max_{\lambda} L(w, b, \lambda)$$

上述式子我们可以用对偶关系转化一下：

$$\max \quad margin = \max_{\lambda} \quad \min_{w,b} L(w, b, \lambda); \quad s.t. \quad \lambda_i \geqslant 0$$

- **讲一讲SVM,知道多少说多少？为什么要用对偶问题求解？（今日头条面试题）**
  - 首先是我们有不等式约束方程，这就需要我们写成min max的形式来得到最优解。而这种写成这种形式对x不能求导，所以我们需要转换成max min的形式，这时候，x就在里面，这样就能对x求导了。而为了满足这种对偶变换成立，就需要满足KKT条件（KKT条件是原问题与对偶问题等价的必要条件，当原问题是凸优化问题时，变为充要条件）。
  - 对偶问题将原始问题中的约束转为了对偶问题中的等式约束

- 方便核函数的引入
- 改变了问题的复杂度。由求特征向量w转化为求比例系数a，在原始问题下，求解的复杂度与样本的维度有关，即w的维度。在对偶问题下，只与样本数量有关。

- **支持向量机中到底什么是支持向量，即支持向量中的向量是指什么**
  - 支持向量本质是向量，而这些向量却起着很重要的作用，如果做分类，他们就是离分界线最近的向量。也就是说分界面是靠这些向量确定的，他们支撑着分类面。即就是离最优分类平面最近的离散点，也可以成为向量。
- **既然有很多的核函数，针对具体问题该怎么选择？如果使用核函数向高维空间映射后，问题仍然是线性不可分的，那怎么办？**
  - 对核函数的选择，现在还缺乏指导原则，但是一般来说，径向基核函数是不会出太大偏差的一种首选。（在做文本分类系统的时候，使用径向基核函数，没有参数调优的情况下，绝大部分类别的准确和召回都在85%以上，可见。虽然libSVM的作者林智仁认为文本分类用线性核函数效果更佳，待考证）
  - 对于松弛变量来说。他是控制近似可分样本的对错分样本惩罚程度c，而这个参数没有一定的公式作参考，智能凭借经验核实验选取。
- **SVM中惩罚参数C的理解（正则化参数对支持向量数的影响）**这个问题非常不熟悉，甚至参数c在哪都不知道
  - C理解为调节优化方向中两个指标（间隔大小，分类准确度）偏好的权重。soft-margin SVM针对hard-margin SVM容易出现的过度拟合问题，适当放宽了margin的大小，容忍一些分类错误（violation），把这些样本当做噪声处理，本质上是间隔大小和噪声容忍度的一种trade-off，至于具体怎么trade-off，对哪个指标要求更高，那就体现在C这个参数上了。

        - 当C趋于无穷大时，这个问题也就是不允许出现分类误差的样本存在，那这就是一个hard-margin SVM问题（过拟合）
        - 当C趋于0时，我们不再关注分类是否正确，只要求间隔越大越好，那么我们将无法得到有意义的解且算法不会收敛。（欠拟合）
- **SVM加核函数用过吗？讲一讲其中的差别？**
    - 训练样本多，维度大就可以用核函数，如果样本少，用核函数比肩容易过拟合
- **SVM在训练的时候有没有遇到hard example？**
    - SVM对hard example的应用很重要，先用一部分训练集训练一个模型，然后用剩下的一部分训练集的一部分做测试，把出错的再送入重新训练，重复二三次，效果会比较好
- **核函数是什么？高斯核映射到无穷维是怎么回事？**
- **SVM在训练的时候有没有遇到hard example？**
    - SVM对hard example的应用很重要，先用一部份训练集训练一个模型，然后用剩下的一部份做测试，把出错的在送入重新训练，重复二三次，效果会比较好
- **SVM的主要面试问题：**硬间隔最大化（几何间隔）、函数间隔、学习的对偶问题，软间隔最大化（引入松弛变量）、非线性支持向量机（核技巧）、Hinge Loss（面试题）
    - 函数间隔就是空间点到超平面的相对距离
    - 几何间隔就是函数间隔乘以标签
- **SVM中硬间隔和软间隔**
    - 硬间隔分类即线性可分支持向量机，软间隔分类即线性不可分支持向量机，利用软间隔时是因为存在一些训练集样本不满足函数间隔大于等于1的条件，于是加入一个非负的参数（松弛变量），让得出的函数间隔加上满足条件。
- **SVM为什么采用间隔最大化？**

- 当训练数据线性可分时，存在无穷个分离超平面可以将两类数据正确分开。感知机利用误分类最小策略，求得分离超平面，不过此时的解有无穷多个。线性可分支持向量机利用间隔最大化求得最优分离超平面，这时，解是唯一的。另一方面，此时的分隔超平面所产生的分类结果是最鲁棒的，对未知实例的泛化性能最强。
- 然后应该借此阐述，几何间隔，函数间隔，及从函数间隔——》求解最小化1/2||w||^2时的w和b。即线性可分支持向量机学习算法—最大间隔发的由来。

- **SVM存在什么问题?**
  - SVM算法对大规模训练样本难以实施。
    - SVM的空间消耗主要是存储寻来呢样本和核矩阵，由于SVM是借助二次规划来求解支持向量，而求解二次规划将涉及m阶矩阵的计算（m为样本个数），当m数目很大时该矩阵的存储核计算将耗费大量的机器内存核运算时间。
    - 用SVM解决多分类问题存在困难
      - 经典的支持向量机算法只给出了二类分类算法，而在数据挖掘的实际应用中，一般要解决多分类问题。可以通过多个二类支持向量机的组合来解决。
    - 对缺失数据敏感，对参数核核函数的选择敏感
      - 支持向量机性能分优劣主要取决于核函数的选取，所以对于一个实际问题而言，都是根据经验来选取，带有一定的随意性。在不同问题领域，核函数应当具有不同形式和参数，所以在选取时候应该将领域知识引入进来，但是目前还没有好的方法来解决核函数的选取问题。

- 支持向量机算法篇
  - 基本的支持向量机算法基本思想是在二次规化的基础上不断迭代寻找支持向量，主要有块算法、分解算法、序列最小优化算法、增量算法等。

- **块算法**
  - Chunking算法的出发点是删除矩阵中对应Lagrange乘数为零的行和列将不会影响最终的结果。对于给定的样本，Chunking算法的目的是通过某种迭代方式逐步排除非支持向量，从而降低训练过程对储存容量的要求。具体的做法是，讲一个大型QP问题分解为一系列较小规模的QP问题，然后找到所有非零的Lagrange乘数并删除。在算法的每步中Chunking都解决一个QP问题，其样本为上一步所剩的具有非零Lagrange乘数的样本以及M个步满足KKT条件的最差样本。如果在某一步中，不满足KKT条件的样本数不足M个，则这些样本全部加入到新的QP问题。每个QP子问题都采用上一个子问题的结果作为初始值。在算法进行到最后一步时，多有Lagrange乘数都被找到，从而解决的出事的大型QP问题。
  - Chunking算法将矩阵规模从训练样本数的平方减少到具有非零Lagrange乘数的样本数的平方，在很大程度上降低了训练过程对储存容量的要求。Chunking算法能够大大提高训练速度，尤其是当支持向量的数目远远小于训练样本的数目时。然而，如果支持向量个数比较多，随着算法迭代次数的增多，所选的块也会越来越大，算法的训练速度依旧会变得十分缓慢。
- 什么是分解算法?
  - 分解算法最早在1997年OSUNA等人中提出，是目前有效解决大规模问题的主要方法。分解算法将二次规划问题分解成一系列规模较小的二次规划子问题，进行迭代求解。在每次迭代中，选取拉格朗日乘子分量的一个子集做为工作集，利用传统优化算法求解一个二次规划的子问题。以分类SVM为例，分解算法的主要思想是将训练样本分成工作集B和非工作集N，工作集B中的样本个数为q，q远小于训练样本总数。每次只针对工作集B中的样本进行训练，而固定N中的训练样本。该算法的关键在于选择一种最优工作集选择算法，而在工作集的选取中采用了随机的方法，因此限制了算法的收敛速度。1998年，JOACHIMS等人在分解算法的基础上对工作集的选择

做了重要改进。采用类似可行方向法的策略确定工作集B。如果存在不满足KTT条件的样本，利用最速下降法，在最速下降方向中存在q个样本，然后以这q个样本构成工作集,在该工作集上解决QP问题，直到所有样本满足KTT条件。如此改进提高了分解算法的收敛速度，并且实现了sVMlieht算法。1998年，PLATT等人提出的序列最小优化(sequentialminimal optimization，SMO)算法是分解算法的一个特例，工作集中只有2个样本，其优点是针对2个样本的二次规划问题可以有解析解的形式，从而避免多样本情况下的数值解不稳定及耗时问题，且不需要大的矩阵存储空间，特别适合稀疏样本。工作集的选择不是传统的最陡下降法，而是启发式。通过两个嵌套的循环寻找待优化的样本，然后在内环中选择另一个样本，完成一次优化，再循环，进行下一次优化，直到全部样本都满足最优条件。SMO算法主要耗时在最优条件的判断上，所以应寻找最合理即计算代价最低的最优条件判别式。

- 什么是序列最小优化算法?
  - PLATT等人在1998年提出的序列最小优化(sequential minimal optimization,SMO)算法是在分解算法的基础上发展起来的,它将工作集减少为只有2个样本。通过两个嵌套的循环寻找待优化的两个样本,外层循环主要寻找工作集的第一个样本;然后采用启发式规则选择第二个样本,选择的原则是使目标函数靠近最优点的速度达到最快;最后用解析的方法快速对选定的样本进行优化。工作集的选择采用启发式选择策略加快了算法的收敛速度，同时减小了矩阵存储空间,适合于稀疏样本。
- 什么是增量算法
  - 增量学习是机器学习系统在处理新增样本时，能够只对原学习结果中与新样本有关的部分进行增加修改或删除操作，与之无关的部分则不被触及。增量训练算法的一个突出特点是支持向量机的学习不是一次离线进行的，而是一个数据逐一加入反复优化的过程。SYED等人在1999年最早提出了SVM增量训练算法，每次只选一小批

常规二次算法能处理的数据作为增量，保留原样本中的支持向量和新增样本混合训练，直到训练样本用完。GAUWENBERGHS等人提出了增量训练的精确解，即增加一个训练样本或减少一个样本对Lagrange系数和支持向量的影响。RALAIVOLA等人提出了另一种增量式学习方法，其思想是基于高斯核的局部特性，只更新对学习机器输出影响最大的Lagrange系数，以减少计算复杂度。孔锐等人提出了一种“快速增量学习算法”，该算法依据边界向量不一定是支持向量，但支持向量一定是边界向量的原理，首先选择那些可能成为支持向量的边界向量，进行SVM的增量学习，找出支持向量，最终求出最优分类面，提高训练速度。孔波等人提出了基于中心距离比值的增量运动向量机，利用中心距离比值，在保证训练和测试准确率没有改变的情况下，提高收敛速度。李东晖等人提出了基于壳向量的线性SVM增量学习算法，通过初始样本集求得壳向量，进行SVM训练，得到支持向量集降低二次规划过程的复杂度，提高整个算法的训练速度和分类精度。以上几种基本算法本质上都是将一个大规模的二次规划问题分解为小的二次规划问题，不同的是分解策略和工作集的选取方法,这也是导致算法收敛速度快慢的原因。

- 其他SVM
  - 什么是最小二次支持向量机?
    - 在求解大型QP问题时，基本支持向量机算法中的块算法和分解算法会存在维度灾难和求解速度过慢等问题，在支持向量机的求解过程中约束条件为不等式约束，为简化寻优过程并保证一定的学习速率，用等式约束来代替式（1）中的不等式约束，用最小乘损失函数代替不敏感损失函数来简化求解过程，从而得到最小二乘支持向量机算法。
  - 什么是模糊支持向量机?
    - 由于实际样本检测时存在不同程度的噪声，需要从样本数据中将非正常数据筛以除降低其影响。具体

实施方法为:通过在训练样本数据集中增加每个样本的隶属度项,如对样本中的噪声数据和孤立点给予较小的隶属度,正常的样本赋予较大的隶属度,从而对不同的样本起到惩罚作用，降低噪声数据对分类最优平面的影响。模糊支持向量机算法结合模糊数学方法通过在基本支持向量机算法的基础上增加隶属度值对最优平面起到调节作用,提高分类精度。但样本数据中如果异常数据较多时,会影响支持向量机的泛化学习能力,另外由于增加了隶属度项,使得核函数计算复杂,训练速度降低。为了克服噪声和野值点对支持向量机的影响，LIN等人将模糊数学和支持向量机相结合，提出了模糊支持向量机，主要用于处理训练样本中的噪声数据。其主要思想是针对支持向量机对训练样本内的噪音和孤立点的敏感性，在训练样本集中增加一项隶属度，并赋予支持向量较高的隶属度，而非支持向量及噪声野值点赋予较小的隶属度，从而降低非支持向量、噪声和野值点对最优超平面的影响。FSVM中存在的问题是如何确定隶属度值，即如何确定各个样本的权重。LIN等人提出了基于类中心的隶属度确定方法，将样本点到其类中心的距离的线性函数作为隶属度函数，但是隶属函数会严重依赖训练样本集的几何形状，降低支持向量的隶属度。HUANG和张翔等人也提出相似的隶属度确定方法。张贵香等人提出一种基于类内超平面距离的隶属度确定方法，将样本点到类内平面距离的线性函数作为隶属度函数，降低了隶属度函数对训练样本集几何形状的依赖，提高了支持向量的隶属度。李苗苗等人提出了一种新的核隶属度函数，该新的隶属度函数不仅依赖于每个样本点到类型中心的距离，还依赖于该样本点最邻近的K个其他样本点的距离。孙名松等人通过对每个样本赋予不同的双隶属度，得到最优分类器。吴青和YAN等人提

出了去边缘的FSVM主要是根据训练集的几何形状，将其分成两个子集，认为其中一个子集不包含支持向量并将其舍去;另一子集包含支持向量并在该子集中进行训练，但会人为地去除一些支持向量而导致分类精度降低。针对一般模糊支持向量机训练时间过长，训练效率低下的问题，刘宏冰等人对边缘数据赋予较大的隶属度，而对类中心附近的数据赋予较小的隶属度，体现加大对容易错分样本进行惩罚的改进策略。施其权等人进一步对离分类超平面较远不可能成为支持向量的数据赋予较小的隶属度，使训练样本集中的数据大大减少。

- 什么是排序支持向量机?
  - 排序学习是当前信息检索和机器学习领域的热点问题，广泛应用于许多领域，包括文档检索、协同过滤、关键字提取、定义发现等等。排序学习问题大致分为基于回归的排序学习和基于分类的排序学习两类。从上面的介绍可以看出，SVM的排序学习模型可以基于分类又可以基于顺序回归，所以自RSVM 61提出以来，该排序学习方法便得到了很大的发展。
  - RSVM在应用中大部分用于信息检索，其中最主要的问题是如何针对检索问题建立适用于具体应用的损伤函数。可以与基本的SVM一样，两类数据使用相同的代价函数;也可以针对两类数据分别建立不同的代价函数，对某些应用会大大提高排序的精确度。
  - YAJIMA等人将RSVM用于推荐系统，文中使用1-SVM对给定数据的相关度顺序进行预测，通过使用图核处理数据集，取得了很好的效果。SVM的回归排序模型也能用于推荐系统，与传统的使用启发式的推荐系统相比，该方法在大样本下的性能是相当高效的。

- 在实际应用中，速度永远是人们追求的目标。传统的RSVM都是使用2-SVM进行学习，YU等人使用1-SVM学习排序，该方法可以使用更少的支持向量，大大提高训练速度。参数选择对于SVM的训练速度起着至关重要的作用，使用用于核方法的正则化路径方法可以加速RSVM中的参数选择并使该过程自动化，从而提高训练速度。在分子测序中，由于数据为结构数据，RSVM的训练速度很长，将粒度计算与RSVM结合，使用粒度计算进行属性约简和问题分割，在保证学习质量的情况下提高训练速度。传统的RSVM在应用中有其局限性，即模型过于简单而不能用于复杂的排序;很难将先验知识加入模型中。为了克服这些局限，一种新的排序支持向量机被提出，它将一般RSVM输出的打分函数映射为一个概率sigmoid函数，该函数用交叉检验进行训练，实际上该函数就是输入数据集的后验概率，可以用其进行一些后处理的工作，避免传统RSVM的应用局限性。

- SVM如何解决多分类问题 或者SVM可以用来划分多类别吗？如果可以，要怎么实现？

  - 1.直接法：直接在目标函数上修改，将多个分类面的参数求解合并到一个最优化问题里面。看似简单但是计算量却非常大。
  - 2.间接法：对训练器进行组合。其中比较典型的有一对一，和一对多。
    - 一对多：对每个类都训练出一个分类器，由于SVM是二分类，所以将此二分类器的两类设定为目标类为一类，其余类为零为一类。这样针对k个类可以训练出k个分类器来测试，那个分类器的概率高，那么这个样本就属于哪一类。这种方法效果不太好，bias比较高。

- 一对一：针对任意两个类训练出一个分类器，如果有k类，一共寻来呢出C（2，k）个分类器来测试，每当被判定属于某一类的时候，该类就加一，最后票数最多的类别被认定为该样本的类。

- SVM的应用
  - 模式识别
    - 在手写字体识别方面,当采用5层神经网络算法时,其识别的错误率为5.1%;贝尔实验室最先将SVM应用于手写字体识别研究,选取三种不同的核函数时,得到的误识率分别为4.0% ,4.1%和4.2%,可看出支持向量机方法比神经网络算法具有更好的分类准确性。柳回春等在SVM 的基础上结合与RBF神经网络将其用于UK心理测试自动分析系统的手写体数字识别问题。手写体识别属于多类问题。相关专家学者在研究2类问题的SVM前提下,形成了能够处理多类问题的相关SVM，其中主要核函数就是sigmoid核函数、径向基核函数、多项式核函数。不但可以支持比较其他分类和支持向量机，还能够支持比较不同形式的SVM,经过大量实践可以发现，存在很大优势。
    - 在人脸识别方面,由于人脸图像存储和SVM训练需要大量的存储空间,周志明等人将小波变换与支持向量机相结合,由小波变换提取人脸特征,减小特征向量维数并将其输入到SVM中,并结合最近邻分类器进行分类,提高了分类器的鲁棒性和分类精度。相关学者和专家经过不断研究和分析以后形成以层次结构形式的支持向量机分类器,由一个非线性和线性支持向量机构成，上述方式不但具备比较低的误差率和比较高的检测率，还存在比较快的速度。此后，人们利用SVM方式来有效判断人脸姿态，并且合理分为6个类别，手工标定方式在多姿态人脸库中发现测试样本和训练样

本集，在SVM基础上的训练集姿态分类器,可以降低到1.67%的错误率。在支持向量机和小波技术上形成的识别人脸技术,压缩提取人脸特征的时候应用小波技术,然后结合支持向量机技术和邻近分类器进行分类，确保具备比较好的鲁棒性能和分类性能。

- 在语音识别方面,由于背景环境中存在不同程度的噪杂声,根据支持向量机和隐式马尔可夫模型相结合的特点,忻栋等建立SVM和隐式马尔可夫模型两者相结合的混合模型,算法比较复杂,用来解决语音识别问题。

- 网页分类
  - 在中文网页分类问题上,贺海军等在网页文本的特征表示和二分类问题的基础上,把二叉决策树和SVM算法相结合构成多类分类器,实现网页文本的分类,取得了较好的分类效果和训练速度。在网络入侵检测分类方面,网络入侵检测其实也是一种网页分类。徐文龙等提出了一种基于从特殊到特殊的直推式支持向量机,从获取的网络数据中进行归纳式学习和标记样本,从中提出特征输入到TSVM学习机中进行学习,检测其异常的网络入侵,提高了测试样本的分类准确度。

- 系统建模和系统辨识
  - 在未知非线性系统建模方面,张浩然等'利用对象的输入输出数据集,在非线性系统的控制结构设计中采用支持向量机建模来获取对象的动态特性,以SVM作为辨识器在控制器中采用指数梯度法实现控制作用。在非线性系统的辨识方面,崔万照等将最小二乘方法应用于支持向量机算法中并选择小波函数作为支持向量机的核函数,构造最小二乘小波支持向量机来解决单输人单输出

(SISO)非线性系统的辨识问题,仿真结果表明此方法能提高辨识效果,加快系统响应时间。

- 其他
  - 支持向量机具备一定的优越性,已经得到大量应用。专家学者提出了支持向量机基础上的水印算法,在数字水印中合理应用支持向量机，存在十分良好的应用效果。并且入侵监测系统已经是十分重要的网络安全技术之一，在分析入侵检测系统的时候，主要应用的就是SVM基础上的主动学习算法，可以在一定程度上降低学习样本,能够增加入侵监测系统整体分类性能。在处理图像中迷糊噪音的时候,依据SVM模糊推理方式形成的一种噪音检测系统,上述方式能够合理除去检测中的噪音,适当保存图像相关信息。在分析混合气体定量和多维光谱定性的时候，不能应用同一种方式来定性和定量分析组合气体吸收普线重叠、输入光谱的维数,训练样本数目有限，在分析地混合气体多维光谱的时候应用支持向量机,依据核函数有效把重叠光谱数据变为支持向量机回归模型,此时可以定量分析混合气体的组分浓度以及定性分析种类。
- SVM与其他模型的对比
  - SVM和LR的区别和联系？或者SVM和Logistic回归的异同?
    - 相同点:
    - 第一，LR和SVM都是分类算法。看到这里很多人就不会认同了，因为在很大一部分人眼里，LR是回归算法。我是非常不赞同这一点的，因为我认为判断一个算法是分类还是回归算法的唯一标准就是样本label的类型，如果label是离散的，就是分类算法，如果label

是连续的，就是回归算法。很明显，LR的训练数据的label是“0或者1”，当然是分类算法。其实这样不重要啦，暂且迁就我认为它是分类算法吧，再说了，SVM也可以回归用呢。

- 第二，如果不考虑核函数，LR和SVM都是线性分类算法，也就是说他们的分类决策面都是线性的。这里要先说明一点，那就是LR也是可以用核函数的，至于为什么通常在SVM中运用核函数而不在LR中运用，后面讲到他们之间区别的时候会重点分析。总之，原始的LR和SVM都是线性分类器，这也是为什么通常没人问你决策树和LR什么区别，决策树和SVM什么区别，你说一个非线性分类器和一个线性分类器有什么区别？
- 第三，LR和SVM都是监督学习算法。这个就不赘述什么是监督学习，什么是半监督学习，什么是非监督学习了。
- 不同点：
- 第一，本质上是其损失函数（loss function）不同。lr的损失函数是 cross entropy loss，adaboost的损失函数是 expotional loss ,svm是 hinge loss，常见的回归模型通常用 均方误差 loss。SVM 和正则化的逻辑回归它们的损失函数：

$$\text{SVM:} \frac{1}{n}\sum_{i=1}^{n}(1-y_i[w_0+\mathbf{x}_i^T\mathbf{w}_1])^+ + \lambda\|\mathbf{w}_1\|/2 \qquad (1)$$

$$\text{Logistic:} \frac{1}{n}\sum_{i=1}^{n}\overbrace{-\log g(y_i[w_0+\mathbf{x}_i^T\mathbf{w}_1])}^{-\log P(y_i|\mathbf{x},\mathbf{W})} + \lambda\|\mathbf{w}_1\|/2 \qquad (2)$$

其中， $g(z)=(1+\exp(-z))^{-1}$.

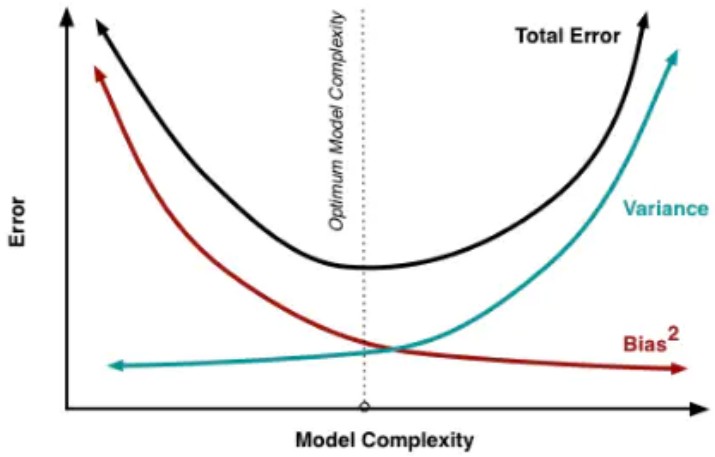
- Hinge损失函数就是目标函数，为了求最大间隔的目标函数。

- 第二，支持向量机只考虑局部的边界线附近的点，而逻辑回归考虑全局（远离的点对边界线的确定也起作用）。
- 第三，在解决非线性问题时，支持向量机采用核函数的机制，而LR通常不采用核函数的方法。
- 第四，线性SVM依赖数据表达的距离测度，所以需要对数据先做normalization，LR不受其影响。
- 第五，SVM的损失函数就自带正则（损失函数中的1/2||w||^2项），这就是为什么SVM是结构风险最小化算法的原因；而LR必须另外在损失函数上添加正则项。

- 集成学习
  - 集成学习的基本分类?
    - GBDT：梯度提升决策树 - 简书 (jianshu.com)
    - 集成学习下有两个重要的策略Bagging核Boosting。
    - Bagging算法是这样做的：每个分类器都随机从原样本中做有放回的采样，然后分别在这些采样后的样本上训练分类器，然后再把这些分类器组合起来。简单的多数投票一般就可以。其代表算法就是随机森林。
    - Boosting的意思是这样，他通过迭代地训练你一系列的分类，每个分类器采用的样本分布都核上一轮的学习结果有关。其代表算法是AdaBoosting， GBDT。
    - Boosting主要关注降低偏差，因此Boosting能基于泛化性能相当弱的学习器构建出很强的集成；Bagging主要关注降低方差，因此它在不剪枝的决策树、神经网络等学习器上效用更为明显。
      - 其实就机器学习算法来说，其泛化误差可以分解为两部分，偏差（bias)和方差(variance)。这个可由下图的式子导出（这里用到了概率论公式D(X)=E(X2)-[E(X)]2）。偏差

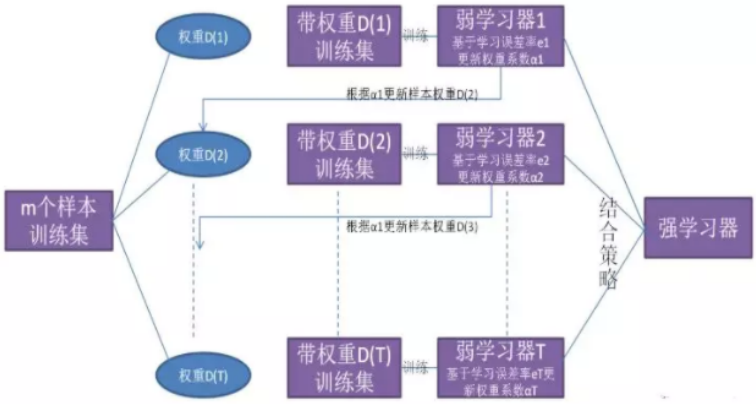
指的是算法的期望预测与真实预测之间的偏差程度，反应了模型本身的拟合能力；方差度量了同等大小的训练集的变动导致学习性能的变化，刻画了数据扰动所导致的影响。这个有点儿绕，不过你一定知道过拟合。

- 如下图所示，当模型越复杂时，拟合的程度就越高，模型的训练偏差就越小。但此时如果换一组数据可能模型的变化就会很大，即模型的方差很大。所以模型过于复杂的时候会导致过拟合。



- 集成学习的基本思想是什么?
  - 结合多个学习器组合成一个性能更好的学习器
- 集成学习为什么有效?
  - 不同的模型通常会在测试集上产生不同的误差；如果成员的误差是独立的，集成学习将显著地比其他成员表现更好。
- **Boosting**：Boosting应用迭代式学习的方式进行学习。
- Boosting的特点是什么
  - Boosting分类器间存在依赖关系，基于学习器之间存在依赖关系，新的学习器需要根据上一个学习器生成。
  - 每次学习都会使用全部训练样本
- Boosting的基本思想是什么?
  - 先从初始训练集训练一个基学习器；初始训练集中各样本的权重是相同的；
  - 根据上一个基学习器的表现，调整样本权重，使分类错误的样本得到更多的关注；
  - 基于调整后的样本分布，训练下一个基学习器；

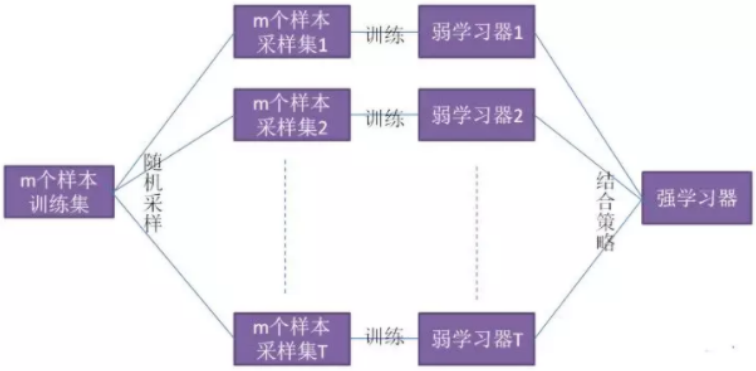
- 测试时，对各基学习器加权得到最终结果；



- GBDT是什么？GBDT：梯度提升决策树
  - 思想：每一颗树学习的是之前所有树的整体预测核标签的误差；
  - GBDT(Gradient Boosting Decision Tree) 又叫 MART （Multiple Additive Regression Tree)，是一种迭代的决策树算法，该算法由多棵决策树组成，所有树的结论累加起来做最终答案。采用平方误差损失函数时，每一颗回归树学习的是之前所有树的结论核残差，拟合得到一个当前残差回归树，残差的的意义如公式：残差 = 真实值 - 预测值
  - 举例说明：假如有个人30岁，我们首先用20岁去拟合，发现损失由10岁，这时我们用6岁去拟合剩下的损失，发现距离还有4岁，第三轮我们用3岁拟合剩下的差距，差距就只有一岁了。如果我们的迭代轮数还没有完，可以继续迭代下面，每一轮迭代，拟合的岁数误差都会减小。
- Xgboost是什么？
  - 思想：不断地添加树，不断地进行特征分裂来生长一棵树，每次添加一个树，其实是学习一个新函数，去拟合上次预测的残差。当我们训练完成得到k棵树，我们要预测一个样本的分数，其实就是根据这个样本的特征，在每棵树中会落到对应的一个叶子节点，每个叶子节点就对应一个分数，最后只需要将每棵树对应的分数加起来就是该样本的预测值。
- **Bagging：**Bagging应用基于并行策略的方式进行学习
- Bagging的特点是什么？
  - 基学习器之间不存在依赖关系，可同时生成。

- 训练每个基学习器时只使用一部分样本；
- 偏好不稳定的学习器作为基学习器
- 注：所谓不稳定的学习器，指的是对样本分布较为敏感的学习器

- Bagging的基本思想是什么？
  - 利用自主采样法对训练集随机采样，重复进行T次；
  - 基于每个采样集寻来你一个基学习器，并得到T个基学习器
  - 与测试是，集体投票策略。

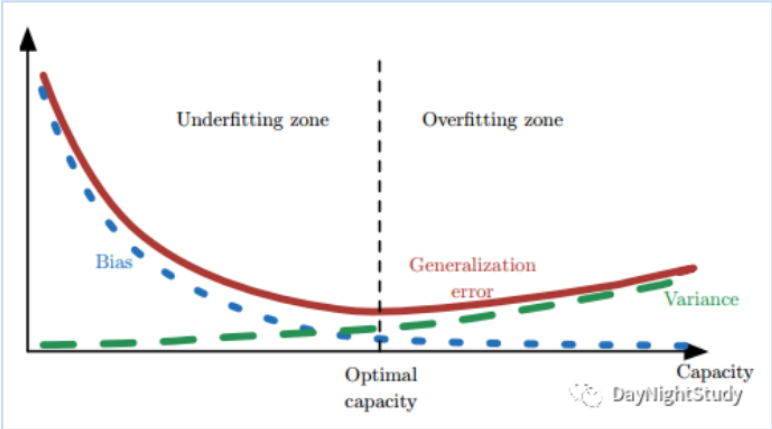


- Bagging的基分类器如何选择？
  - 所用基分类器最好本身对样本分布较为敏感（不稳定性）
- Bagging的优点是什么？
  - 集成后分类器方差比基分类器的小
- 随机森林是什么？
  - 思想：用随机的方式建立一个森林，森林里面有很多的决策树组成，随机森林的每一棵决策树之间是没有关联的。在得到森林之后，当有一个新的输入样本进入的时候，就让森林中的每一棵决策树分别进行一下判断，对于分类算法，看看这个样本应该属于哪一类，然后看看哪一类被选择最多，就预测这个样本为那一类。对回归问题，计算k的模型的均值作为最后的结果。
- Stacking：Stacking应用基于串行策略的方式进行学习
- Stacking的特点是什么？
  - 初级学习器与次级学习器自建存在依赖关系，初学习器的输出作为次级学习器的输入

- Stacking的基本思路是什么?
  - 先从初始训练集寻来你T个不同的初级学习器;
  - 利用每个初级学习器的输出构建一个次级数据集，该数据集依然使用初始数据集的标签;
  - 根据新的数据集寻来你次级学习器;
  - 多级学习器的构建过程类似;
- 常见问题:
  - **为什么使用决策树作为基学习器?**
    - 决策树的表达能力和泛化能力，可以通过剪枝快速调整;
    - 决策树可以方便地将样本的权重整合到训练过程中;（适合Boosting策略）
    - 决策树是一种不稳定的学习器;（适合Bagging策略）
      - 注：所谓不稳定，指的是数据样本的扰动会对决策树的结果产生较大的影响;
    - 类似问题：基学习器有什么特点？基学习器有什么要求?
  - **为什么不稳定的学习器更适合作为基学习器?**
    - 不稳定的学习器容易受到样本分布的影响（方差大），很好的引入了随机性；这有助于在集成学习（特别是采用 Bagging策略）中提升模型的泛化能力。
    - 为了更好的引入随机性，有时会随机选择一个属性子集中的最优分裂属性，而不是全局最优（随机森林）
  - **哪些模型适合作为基学习器?**
    - 决策树
    - 神经网络----神经网络也属于不稳定的学习器；通过调整神经元的数量、网络层数，连接方式初始权重也能很好的引入随机性和改变模型的表达能力和泛化能力
  - **Bagging 方法中能使用线性分类器作为基学习器吗? Boosting 呢?**
    - Bagging 方法中不推荐:

- 线性分类器都属于稳定的学习器（方差小），对数据不敏感；
- 甚至可能因为 Bagging 的采样，导致在训练中难以收敛，增大集成分类器的偏差

- Boosting 方法中可以使用：
  - Boosting 方法主要通过降低偏差的方式来提升模型的性能，而线性分类器本身具有方差小的特点，所以两者有一定相性
  - XGBoost 中就支持以线性分类器作为基学习器
- **Boosting/Bagging 与 偏差/方差 的关系?**
  - 提升弱分类器性能的原因：
    - Boosting： 降低了偏差
    - Bagging： 降低了方差
  - Boosting 方法
    - 基本思路：减小模型的训练误差拟合残差或者加大错类的权重），加强模型的学习能力，减小偏差
    - 缺点：但 Boosting 不会显著降低方差，因为其训练过程中各基学习器是强相关的，缺少独立性。
  - Bagging 方法
    - 对 n 个独立不相关的模型预测结果取平均，方差是原来的 1/n
    - 假设所有基分类器出错的概率是独立的，超过半数基分类器出错的概率会随着基分类器的数量增加而下降
  - 泛化误差、偏差、方差、过拟合、欠拟合、模型复杂度（模型容量）的关系图



- 不同模型之间的对比
  - **LR vs GBDT?**
    - 从机器学习三要素的角度
      - 机器学习·总览篇 IV 机器学习的三要素 - 知乎 (zhihu.com)
      - 机器学习三大要素：1.模型；2.策略；3.算法
        - 模型：谈到机器学习，经常会谈到机器学习的“模型”。在机器学习中，**模型的实质是一个假设空间（hypothesis space），这个假设空间是“输入空间到输出空间所有映射”的一个集合，这个空间的假设属于我们的先验知识。**然后，机器学习通过“数据+三要素”的训练，目标是获得假设空间的一个最优解，翻译一下就是求模型的最优参数。
        - 策略：在模型部分，机器学习的学习目标是获得假设空间（模型）的一个最优解，那么问题来了，如何评判优还是不优？**策略部分就是评判“最优模型”（最优参数的模型）的准则或方法。**
          - 了解机器学习的策略，最关键是掌握10个名词：**欠拟合（Underfitting）、过拟合（Overfitting）、经验风险（Empirical risk）、经验风险最小化（Empirical risk minimization, ERM）、结构风险（Structural risk）、结构风险最小化**

（Structural risk minimization, SRM）、**损失函数**（Loss function）、**代价函数**（Cost function）、**目标函数**（Object function）、**正则化**（Regularization）

- 关于训练集的平均损失称作**经验风险**(Empirical risk)
- 我们希望经验风险最小，称为**经验风险最小化**（Empirical risk minimization, ERM）
- 这个度量经验风险的函数称为**代价函数**（Cost function）
- $J(f)$这个函数专门用来度量模型的复杂度，表示模型的**结构风险**（Structural risk），在机器学习中也叫**正则化**(Regularization)
- 为了让模型尽可能结构简单，我们的优化目的又多一个，即**结构风险最小化**（Structural risk minimization）
- 经验结构风险的度量函数，该函数就被称为**目标函数**（Object function）

经验风险-代价函数　结构风险-正则化

$$\frac{\sum_{i=1}^{N}|y_i - f(x_i)|^2}{N} + [L_1|L_2]$$

- 算法：在策略部分，机器学习的学习目标转换成了求目标函数的最小值，而**算法部分就是对函数最优解的求解方法。**机器学习求解目标函数常用的算法有最小二乘法、梯度下降法（属于迭代法的一种），最小二乘法针对线性模型，而梯度下降法适用于任意模型，适用最为广泛。
- 最小二乘法：把极小化 *F(x)* 称为最小二乘问题。最小二乘法分为线性最小二乘问题和非

线性最小二乘问题，解决线性最小二乘问题的最常用方法就是最小二乘法。

- 对于坐标为(x,y)的n个样本，假设拟合直线为y=ax+b, 通过使下面式子最小，然后对a和b分别求偏导，得到的了a和b关于n个样本的表示，这种方法就叫做最小二乘法，这是一种基于高斯分布假设的回归方法。

$$\sum_{i=1}^{n}(y_i - ax_i - b)^2$$

- 从模型的角度
  - 相同点：
    - 监督学习；
    - 判别模型；
    - 直接对数据的分布建模；
    - 不尝试挖掘隐含变量；
  - 不同点：
    - Logistic Regression：
      - 线性模型；
      - 分类器：线性分类器；
      - VC 维度： d+1;
        - VC Dimension，其定义是hypothesis set能够shatter的最多inputs的个数，即最大完全正确的分类能力
        - 对一个指示函数集，如果存在H个样本能够被函数集中的函数按所有可能的2的H次方种形式分开，则称函数集能够把H个样本打散；函数集的VC维就是它能打散的最大样本数目H。若对任意数目的样本都有函数能将它们打散，则函数集的VC维是无穷大，有界实函数的VC维可以通过用一定的阈值将它转化成[指示函数]来定义。

- GBDT:
  - 非线性模型;
  - boosting 模型，可以无限分类，具有无限逼近样本 VC 维的特点;
  - VC 维度：远远大于 d+1;
- 从策略角度
  - Loss(经验风险最小化) + 正则(结构风险最小化)
  - 从 Loss 角度
  - Logistic Regression：
    - 输出： y = 1 的概率;
    - Loss 损失函数：交叉熵;
    - 准则：最大熵原理，“为了追求最小分类误差，追求最大熵 Loss”;
    - 本质：分类器算法，而且对数据的噪声具有高斯假设;
  - GBDT：
    - 基分类器：CART，其无论是处理分类还是回归均是将采用回归拟合（将分类问题通过 softmax 转换为回归问题），用当前轮 CART 树拟合前一轮目标函数与实际值的负梯度 ht = -g;
    - 本质：回归算法;
  - 也正是因为 GBDT 采用的 CART 树模型作为基分类器进行负梯度拟合，（负梯度拟合是残差近似）其是一种对特征样本空间进行划分的策略，不能使用 SGD 等梯度优化算法，而是 CART 树自身的节点分裂策略：均方差(回归) 也带来了算法上的不同； GBDT 损失函数指的是前一轮拟合模型与实际值的差异，而树节点内部分裂的特征选择则是固定为 CART 的均方差，目标损失函数可以自定义，当前轮 CART 树旨在拟合负梯度。

- 决策树的生成主要分一下两步，1.节点的分裂：一般当一个节点所代表的属性无法给出判断时，则选择将一节点分成2个子节点。2.阈值的确定：选择适当的阈值使得分类错误率最小。
- 决策树(Decision Tree)：通俗易懂之介绍 - 知乎 (zhihu.com)
- 比较常用的决策树有ID3、C4.5和CART（Classification And Regression Tree），CART的分类效果一般优于其他决策树。
    - ID3: 由增熵（Entropy）原理来决定那个做父节点，那个节点需要分裂。对于一组数据，熵越小说明分类结果越好。熵定义如下：$Entropy = -\sum[p(x_i) * log_2(p(x_i))]$ 其中$p(x_i)$为x_i出现的概率。假如是2分类问题，当A类和B类各占50%的时候，$Entropy = -(0.5 * log_2(0.5) + 0.5 * log_2(0.5)) = 1$当只有A类，或只有B类的时候，$Entropy = -(1 * log_2(1) + 0) = 0$所以当Entropy最大为1的时候，是分类效果最差的状态，当它最小为0的时候，是完全分类的状态。因为熵等于零是理想状态，一般实际情况下，熵介于0和1之间。
    - C4.5：通过对ID3的学习，可以知道ID3存在一个问题，那就是越细小的分割分类错误率越小，所以ID3会越分越细，但是这种分割显然只对训练数据有用，对于新的数据没有意义，这就是所说的过度学习（Overfitting）。所以为了避免分割太细，c4.5对ID3进行了改进，C4.5中，优化项要除以分割太细的代价，这个比值叫做信息增益率，显然分割太细分母增加，信息增益率会降低。除此之外，其他的原理和ID3相同。

- CART只能将一个父节点分为2个子节点。CART用GINI指数来决定如何分裂：GINI指数：总体内包含的类别越杂乱，GINI指数就越大（跟熵的概念很相似）。选择GINI指数最小的方案，例如：
  - a. 比如出勤率大于70%这个条件将训练数据分成两组：大于70%里面有两类：【好学生】和【不是好学生】，而小于等于70%里也有两类：【好学生】和【不是好学生】。
  - b. 如果用分数小于70分来分：则小于70分只有【不是好学生】一类，而大于等于70分有【好学生】和【不是好学生】两类。比较a和b，发现b的凌乱程度比a要小，即GINI指数b比a小，所以选择b的方案。
- CART可以对每个叶节点里的数据分析其均值方差，当方差小于一定值可以终止分裂，以换取计算成本的降低。
- CART和ID3一样，存在偏向细小分割，即过度学习（过度拟合的问题），为了解决这一问题，对特别长的树进行剪枝处理，直接剪掉。
  - 预剪枝是指在生成决策树的过程中，对每个结点划分前进行模拟，如果划分后不能带来决策树泛化性能的提升，则停止划分并将当前结点标记为叶节点。后剪枝则是指在生成一棵决策树后，自下而上地对非叶结点进行考察，如果将该节点对应的子树替换为叶节点能带来泛化性能的提升，则进行替换。
- 以上的决策树训练的时候，一般会采取Cross-Validation法：比如一共有10组数据：

以上内容整理于 幕布文档