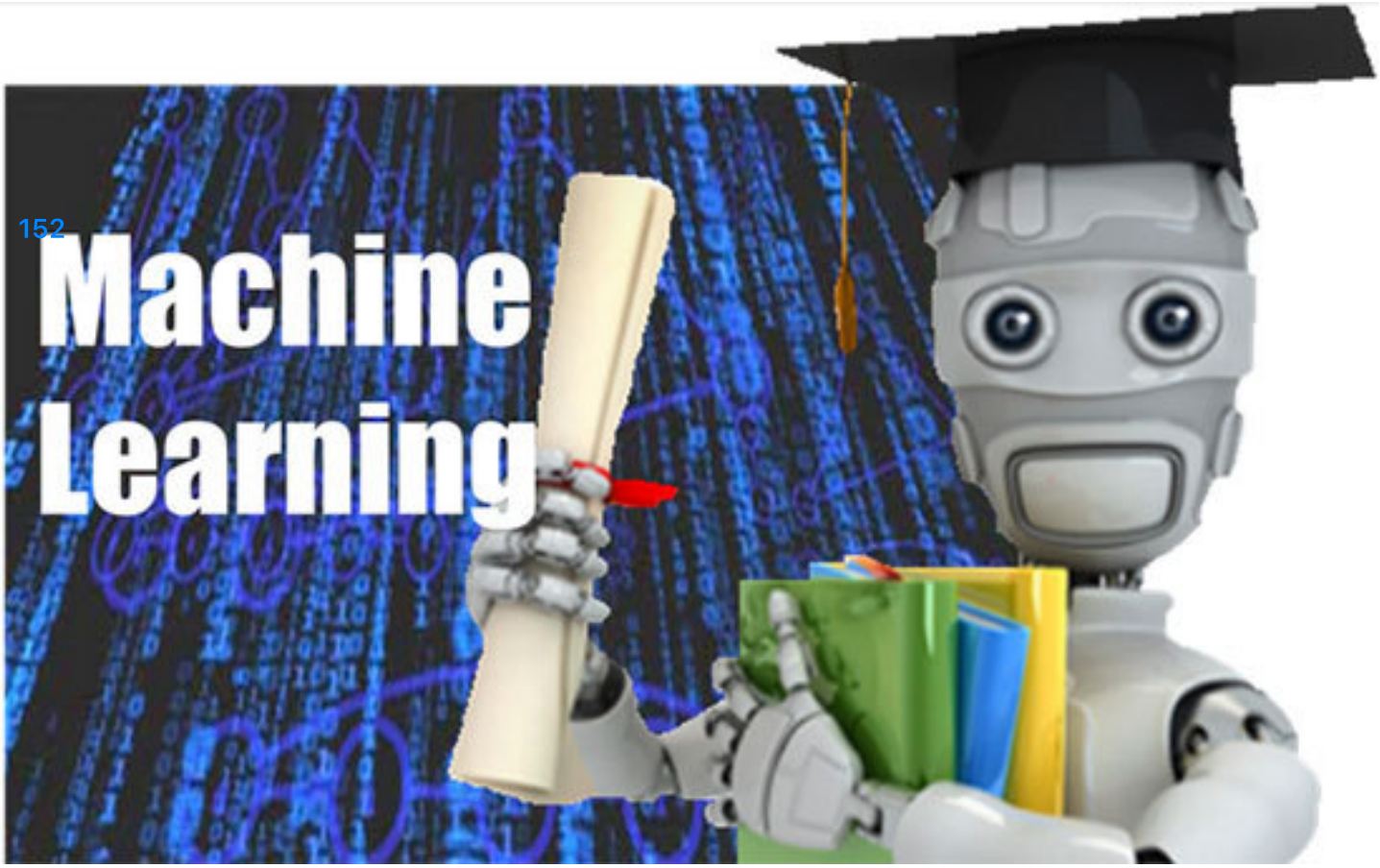


机器学习算法面试小结

dragon fi...
过气的CV从业者

关注他

152 人赞了该文章

目录：

知乎 首发于 机器学习与算法

- 优化问题的求解方法
  - 梯度下降法
    - 优化思想
    - 缺点
    - 批量梯度下降法
    - 随机梯度下降法

- 牛顿法
  - 牛顿法比梯度下降法快
  - 拟牛顿法

- 拉格朗日法

- 机器学习算法选择
  - 贝叶斯
  - K近邻
    - 三要素：
    - k值的选择
    - 分类决策规则
    - 优缺点：

- KD树
  - 构造KD树
  - KD树的搜索

- 决策树
- 随机森林
  - 基本概念

- RF的优点
- 缺点
- RF的学习算法

152

- GBDT
  - 基本概念
  - GBDT与传统Boosting（AdaBoost）的区别
  - GBDT正则化的方式
  - GBDT的优缺点
  - GBDT预测时每一棵树是否能并行?
  - GBDT和RF的区别与联系
  - XGBOOST相比于GBDT有何不同？XGBOOST为什么快？XGBOOST如何支持并行?
  - ababoost

- 集成学习与方差偏差
  - 为什么说bagging是减少variance，而boosting是减少bias?
  - 为什么把特征组合之后还能提升

- 总体性问题
  - 分类与回归的区别
  - 生成模型与判别模型的区别
  - 精确率、召回率、F1 值、ROC、AUC 各自的优缺点是什么?
  - 过拟合
  - 线性分类器与非线性分类器的区别以及优劣
  - 样本不均衡如何解决
    - 重采样（resampling）技术：

- BP算法为什么不能适应于深度学习
- CNN卷基层和pooling层的作用
- DNN常用的激活函数有哪些，各有什么特点
- 解决relu神经元坏死问题
- 什么样的资料不适合用深度学习?
- 什么是共线性，跟过拟合有何关联?
- pooling技术有哪些，有什么作用,有什么区别
- pooling的反向传播

152

- 特征选择的方法
  - 特征选择方法举例
  - 特征选择方法分类
    - Filter过滤法
    - Embedded 集成方法
    - 降维

- LR相关问题
  - LR与BP
  - LR为什么使用sigmoid函数
  - 为什么LR把特征离散化后效果更好
  - 如何用LR建立一个广告点击的模型：
  - LR的过拟合
  - 关于LR的多分类：softmax

- SVM相关问题
  - SVM的主要特点
  - 缺点：
  - 为什么要引入对偶问题
  - 样本失衡的影响
  - 样本失衡时，如何评价分类器的性能好坏?

- 拉格朗日乘子法 和KKT条件
  - 凸函数
  - 等式条件约束
  - 不等式约束与KKT条件

152

- SVM的原问题和对偶问题
- 为什么要引入对偶算法
- SVM解决过拟合的方法
- SVM优缺点
- SMO算法
- SVM多分类问题

- reference

机器学习是做NLP和计算机视觉这类应用算法的基础，虽然现在深度学习模型大行其道，但是懂一些传统算法的原理和它们之间的区别还是很有必要的。可以帮助我们做一些模型选择。本篇博文就总结一下各种机器学习算法的特点和应用场景。本文是笔者结合自身面试中遇到的问题和总结网络上的资源得到的，所有引用已给出链接，如侵删。

152

机器学习

SVM与LR的区别

从模型解决问题的方式来看

Linear SVM直观上是trade-off两个量

a large margin，就是两类之间可以画多宽的gap；不妨说是正样本应该在分界平面向左gap/2（称正分界），负样本应该在分解平面向右gap/2（称负分界）

L1 error penalty，对所有不满足上述条件的点做L1 penalty

给定一个数据集，一旦完成Linear SVM的求解，所有数据点可以被归成两类

一类是落在对应分界平面外并被正确分类的点，比如落在正分界左侧的正样本或落在负分界右侧的负样本

第二类是落在gap里或被错误分类的点。

假设一个数据集已经被Linear SVM求解，那么往这个数据集里面增加或者删除更多的一类点并不会改变重新求解的Linear SVM平面。不受数据分布的影响。

求解LR模型过程中，**每一个数据点对分类平面都是有影响的**，它的影响力远离它到分类平面的距离指数递减。换句话说，LR的解是**受数据本身分布**影响的。在实际应用中，如果数据维度很高，LR模型都会配合参数的L1 regularization。

两者的区别

两个模型对**数据和参数**的敏感程度不同，Linear SVM比较依赖penalty的系数和**数据表达空间的测度**，而（带正则项的）LR**比较依赖对参数做L1 regularization的系数**。但是由于他们或多或少都是线性分类器，所以实际上对低维度数据overfitting的能力都比较有限，相比之下对高维度数据，LR的表现会更加稳定，为什么呢？因为Linear SVM在计算margin有多“宽”的时候是依赖数据表达上的距离测度的，换句话说如果这个测度不好（badly scaled，这种情况在高维数据尤为显著），所求得的所谓Large margin就没有意义了，这个问题即使换用kernel trick（比如用Gaussian kernel）也无法完全避免。所以使用Linear SVM之前一般都需要先对数据做normalization，而求解LR（without regularization）时则不需要或者结果不敏感。

Linear SVM和LR都是线性分类器
Linear SVM不直接依赖数据分布，分类平面不受一类点影响；**LR则受所有数据点的影响，如果数**

Linear SVM依赖penalty的系数，实验中需要做validation
Linear SVM和LR的performance都会收到outlier的影响，其敏感程度而言，谁更好很难下明确结论。

152

balance的方法

调整正、负样本在求cost时的权重，比如按比例加大正样本cost的权重。然而deep learning的训练过程是on-line的因此你需要按照batch中正、负样本的比例调整。

做训练样本选取：如hard negative mining，只用负样本中的一部分。

做训练样本选取：如通过data augmentation扩大正样本数量。

过拟合方面

LR容易欠拟合，准确度低。

SVM不太容易过拟合：松弛因子+损失函数形式

注意SVM的求解方法叫拉格朗日乘子法，而对于均方误差的优化方法是最小二乘法。

方法的选择

在Andrew NG的课里讲到过：

如果Feature的数量很大，跟样本数量差不多，这时候选用LR或者是Linear Kernel的SVM

如果Feature的数量比较小，样本数量一般，不算大也不算小，选用SVM+Gaussian Kernel

如果Feature的数量比较小，而样本数量很多，需要手工添加一些feature变成第一种情况
152

当你的数据非常非常非常非常非常大然后完全跑不动SVM的时候，跑LR。SVM适合于小样本学习。多大算是非常非常非常非常非常大？ 比如几个G，几万维特征，就勉强算大吧...而实际问题上几万个参数实在完全不算个事儿，太常见了。随随便便就得上spark。读一遍数据就老半天，一天能训练出来的模型就叫高效了。所以在新时代，LR其实反而比以前用的多了=. =

应用场景方面不同

拟合程度，样本量，

距离测度，数据balance

模型简单易解释

如果数据特征维度高，svm要使用核函数来求解

Note：拉格朗日对偶没有改变最优解，但改变了算法复杂度：原问题—样本维度；对偶问题–样本数量。所以 线性分类&&样本维度<样本数量：原问题求解（liblinear默认）； 非线性–升维—一般导致 样本维度>样本数量：对偶问题求解

SVM适合处理什么样的数据?

152

高维稀疏，样本少。【参数只与支持向量有关，数量少，所以需要的样本少，由于参数跟维度没有关系，所以可以处理高维问题】

机器学习常见算法总结

机器学习常见算法个人总结（面试用）

朴素贝叶斯

朴素贝叶斯的**优点**:
对小规模的数据表现很好，适合多分类任务，适合增量式训练。
缺点:
对输入数据的表达形式很敏感（离散、连续，值极大极小之类的）

线性回归

线性回归试图学得一个线性模型以尽可能准确地预测实值输出标记。均方误差是回归任务中最常用的性能度量，基于均方误差最小化来进行模型求解的方法成为最小二乘法。在线性回归中，最小二

优化方法

152

当x矩阵是列满秩的时候，可以用最小二乘法，但是求矩阵的逆比较慢

$$\theta = (X^T X)^{-1} X^T \vec{y}.$$

梯度下降法，以最大似然估计的结果对权值求梯度，sigmoid函数也是如此

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)}) \right) x_j^{(i)}.$$

enter description here

均方无法的概率解释

假设根据特征的预测结果与实际结果有误差∈ (i) ,那么预测结果θ T x (i) 和真实结果y (i) 满足下式:

$$y^{(i)} = \theta^T x^{(i)} + \epsilon^{(i)},$$

一般来讲,误差满足平均值为 0 的高斯分布,也就是正态分布。那么 x 和 y 的条件概率也就是

152
用条件概率最大似然估计法得到：

$$\frac{1}{2}\sum_{i=1}^{m}(y^{(i)}-\theta^T x^{(i)})^2,$$

enter description here

LR回归

$$h_\theta(x)=g(\theta^T x)=\frac{1}{1+e^{-\theta^T x}},$$

$$g(z)=\frac{1}{1+e^{-z}}$$

enter description here

152

回归用来分类 0/1 问题,也就是预测结果属于 0 或者 1 的二值分类问题

$$\begin{aligned} P(y=1 \mid x;\theta) &= h_\theta(x) \\ P(y=0 \mid x;\theta) &= 1-h_\theta(x) \end{aligned}$$

仍然求的是最大似然估计,然后求导,得到迭代公式结果为，梯度下降法：

$$\theta_j := \theta_j + \alpha \left(y^{(i)} - h_\theta(x^{(i)})\right) x_j^{(i)}$$

enter description here

优化问题的求解方法

优化，从而训练出最好的模型。常见的最优化方法有梯度下降法、牛顿法和拟牛顿法、共轭梯度法等等。

152

梯度下降法

优化思想

当目标函数是凸函数时，梯度下降法的解是全局解。一般情况下，其解不保证是全局最优解，梯度下降法的速度也未必是最快的。梯度下降法的优化思想是用当前位置负梯度方向作为搜索方向，因为该方向为当前位置的最快下降方向，所以也被称为是”最速下降法“。最速下降法越接近目标值，步长越小，前进越慢。

缺点

梯度下降法的最大问题就是会陷入局部最优，靠近极小值时收敛速度减慢。

批量梯度下降法

152

随机梯度下降法

最小化每条样本的损失函数，虽然不是每次迭代得到的损失函数都向着全局最优方向， 但是大的整体的方向是向全局最优解的，最终的结果往往是在全局最优解附近，适用于**大规模训练样本**情况。

随机梯度下降是通过每个样本来迭代更新一次，如果样本量很大的情况（例如几十万），那么可能**只用其中几万条或者几千条的样本**，就已经将theta迭代到最优解了，对比上面的批量梯度下降，迭代一次需要用到十几万训练样本，一次迭代不可能最优，如果迭代10次的话就需要遍历训练样本10次。但是，SGD伴随的一个问题是**噪音**较BGD要多，使得SGD并不是每次迭代都向着整体最优化方向。

牛顿法

牛顿法是一种在实数域和复数域上近似求解方程的方法。方法使用函数f (x)的**泰勒级数的前面几项**来寻找方程f (x) = 0的根。牛顿法最大的特点就在于它的收敛速度很快。

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

迭代公式

152

牛顿法比梯度下降法快

牛顿法是二阶收敛，梯度下降是一阶收敛，所以牛顿法就更快。如果更通俗地说的话，比如你想找一条最短的路径走到一个盆地的最底部，梯度下降法每次只从你当前所处位置选一个坡度最大的方向走一步，牛顿法在选择方向时，不仅会考虑坡度是否够大，还会考虑你走了一步之后，坡度是否会变得更大。所以，可以说牛顿法比梯度下降法看得更远一点，能更快地走到最底部。

但是牛顿法要**算hessian矩阵的逆**，比较费时间。

拟牛顿法

拟牛顿法的本质思想是改善牛顿法每次需要求解复杂的Hessian矩阵的逆矩阵的缺陷，它使**用正定矩阵来近似Hessian矩阵的逆**，从而简化了运算的复杂度。拟牛顿法和最速下降法一样只要求每一步迭代时知道目标函数的梯度。通过测量梯度的变化，构造一个目标函数的模型使之足以产生超线性收敛性。这类方法大大优于最速下降法，尤其对于困难的问题。另外，因为拟牛顿法不需要二阶导数的信息，所以有时比牛顿法更为有效。

拉格朗日乘数法

152

拉格朗日乘子法主要用于解决约束优化问题，它的基本思想就是通过引入拉格朗日乘子来将含有n个变量和k个约束条件的约束优化问题转化为含有（n+k）个变量的无约束优化问题。拉格朗日乘子背后的数学意义是其为约束方程梯度线性组合中每个向量的系数。

通过引入拉格朗日乘子建立极值条件，对n个变量分别求偏导对应了n个方程，然后加上k个约束条件（对应k个拉格朗日乘子）一起构成包含了（n+k）变量的（n+k）个方程的方程组问题，这样就能根据求方程组的方法对其进行求解。

机器学习算法选择

机器学习算法选择

随机森林平均来说最强，但也只在9.9%的数据集上拿到了第一，优点是鲜有短板。SVM的平均水平紧随其后，在10.7%的数据集上拿到第一。神经网络（13.2%）和boosting（~9%）表现不错。数据维度越高，随机森林就比AdaBoost强越多，但是整体不及SVM2。数据量越大，神经网络就越强。

贝叶斯

是相对容易理解的一个模型，至今依然被垃圾邮件过滤器使用。

K近邻

152

典型的例子是KNN，它的思路就是——对于待判断的点，找到离它最近的几个数据点，根据它们的类型决定待判断点的类型。

它的特点是完全跟着数据走，没有数学模型可言。

三要素：

k值的选择

距离的度量（常见的距离度量有欧式距离，马氏距离等）

分类决策规则 （多数表决规则）

k值的选择

k值越小表明模型越复杂，更加容易过拟合

但是k值越大，模型越简单，如果k=N的时候就表明无论什么点都是训练集中类别最多的那个类

> 所以一般k会取一个较小的值，然后用过交叉验证来确定
> 这里所谓的交叉验证就是将样本划分一部分出来为预测样本，比如95%训练，5%预测，然后k分别取1，2，3，4，5之类的，进行预测，计算最后的分类误差，选择误差最小的k

分类决策规则

152

找到最近的k个实例之后，可以计算平均值作为预测值，也可以给这k个实例加上一个权重再求平均值，这个权重与度量距离成反比（越近权重越大）

优缺点：

优点

思想简单

可用于非线性分类

训练时间复杂度为O(n)

准确度高，对outlier不敏感

缺点

计算量大

样本不平衡问题不适用

需要大量的内存

KD树

KD树是一个二叉树，表示对**K维空间**的一个划分，可以进行快速检索

152

构造KD树

在k维的空间上循环找子区域的中位数进行划分的过程。

假设现在有K维空间的数据集： ，

首先构造根节点，以坐标的中位数b为切分点，将根结点对应的矩形局域划分为两个区域，区域1中,区域2中

构造叶子节点，分别以上面两个区域中的中位数作为切分点，再次将他们两两划分，作为深度1的叶子节点，（如果a2=中位数，则a2的实例落在切分面）

不断重复2的操作，深度为j的叶子节点划分的时候，索取的 的，直到两个子区域没有实例时停止

KD树的搜索

首先从根节点开始递归往下找到包含x的叶子节点，每一层都是找对应的xi

将这个叶子节点认为是当前的“近似最近点”

递归向上回退，如果以x圆心，以“近似最近点”为半径的球与根节点的**另一半子区域**边界相交，则说明另一半子区域中存在与x更近的点，则进入另一个子区域中查找该点并且更新”近似最近点“

重复3的步骤，直到另一子区域与球体不相交或者退回根节点

log(n)

152

决策树

决策树的特点是它总是在沿着特征做切分。随着层层递进，这个划分会越来越细。

因为它能够生成清晰的基于特征(feature)选择不同预测结果的树状结构

随机森林

器学习岗位面试问题汇总 之 集成学习

基本概念

天池离线赛 - 移动推荐算法（四）： 基于LR, RF, GBDT等模型的预测

它首先随机选取不同的特征(feature)和训练样本(training sample)**bagging**，生成大量的决策树，然后综合这些决策树的结果来进行最终的分类。

随机森林在现实分析中被大量使用，它相对于决策树，在准确性上有了很大的提升

适用场景：**数据维度相对低**（几十维），同时对准确性有较高要求时。
152

参数调节

是一种基于决策树基模型的集成学习方法，其核心思想是通过**特征采样来降低训练方差**，提高集成泛化能力。

max_depth 属于基学习器参数，它控制着每个决策树的深度，一般来说，**决策树越深，模型拟合的偏差越小**，但同时拟合的开销也越大。一般地，需要保证足够的树深度，但也不宜过大。

RF与传统bagging的区别

（1）**样本采样**：RF**有放回**选取和整体样本数目相同的样本，一般bagging用的样本<总体样本数
（2）**特征采样**：RF对特征进行采样，BAGGING用全部特征

RF的优点

（3）防止过拟合

（4）能够处理高维特征，且不用做特征选择，可以给出**特征重要性的评分**，训练过程中，可以检测到feature的相互影响

152

缺点

①树越多，随机森林的表现才会越稳定。所以在实际使用随机森林的时候需要注意如果树不够多的时候，可能会导致不稳定的情况。

②不平衡数据集。分类结果会倾向于样本多的类别，所以训练样本中各类别的数据必须相同。Breiman在实际实现该算法的时候有考虑到了这个问题，采取了根据样本类别比例对决策树的判断赋予不同权值的方法

RF的学习算法

ID3：离散
C4.5：连续
CART：离散或连续

GBDT

基本概念

152

GBDT（梯度迭代决策树）是一种基于**决策回归树的Boosting**模型，其核心思想是将提升过程建立在对“**之前残差的负梯度表示**”的回归拟合上，通过不断的迭代实现降低偏差的目的。

GBDT设置大量基学习器的目的是为了集成来**降低偏差**，所以 n_estimators （基决策器的个数）一般会设置得大一些。

对于GBDT模型来说，其每个基学习器是一个弱学习器(欠拟合)，**决策树的深度一般设置得比较小，以此来降低方差**（模型复杂度低），之后在经过残差逼近迭代来降低偏差，从而形成强学习器。

GBDT与传统Boosting（AdaBoost）的区别

Boosting算法，但与传统boosting有区别、**拟合上一步的残差**，传统意义上说不能并行，只能用CART回归树，降低偏差

迭代思路不同：传统boosting对训练样本进行加权，GBDT则是**拟合残差**，下一棵树沿残差梯度下降的方向进行拟合

GBDT正则化的方式

（3）子抽样，**不放回**，SGBT，可以实现一定程度上的并行

152

GBDT的优缺点

优点：（1）调参少的情况下，**准确率也高**（SVM）
（2）灵活处理各种数据，包括连续和离散，无需归一化处理（LR）
（3）模型非线性变换多，特征不用经过复杂处理即**可表达复杂信息**
（4）从一定程度上可以防止过拟合，小步而非大步拟合
缺点：（1）一般来说传统的GBDT只能串行，但是也可以通过子采样比例（0.5~0.8）实现某种意义上的并行，但一般这就不叫GBDT了。
（2）**对异常值敏感**，但是可以采取一些健壮的损失函数缓解，如Huber./Quantile损失函数

GBDT预测时每一棵树是否能并行?

可以，训练需串行，**预测可并行**

GBDT和RF的区别与联系

联系：多棵树进行训练+多棵树共同进行预测
区别：（1）取样方式
（2）预测时，RF多数投票，GBDT加权累加
（3）样本的关系—>并行和串行

（6）通过减少方差/偏差提高性能

152

XGBOOST相比于GBDT有何不同？XGBOOST为什么快？XGBOOST如何支持并行？

（1）GBDT只能用CART回归树，而XGBOOST可以用CART树（回归/分类）,还可以用用想LR之类的线性模型，相当于加入L1、L2正则项的LR或线性回归
（2）列抽样，可以并行，不是树粒度上的，是特征粒度上的，block块，并行计算所有信息增益等信息
（3）可处理多种特征，且对缺失值也不用进行处理
（4）GBDT在**残差梯度下降方向拟合，一阶导；XGBOOST泰勒展开至二阶导**
（5）近似直方图算法，高效生产候选分割点
（6）shrink，缩减，叶子节点同时乘，防止过拟合
（7）可以自己定义评价函数
（8）代价函数含正则化项，防止过拟合

ababoost

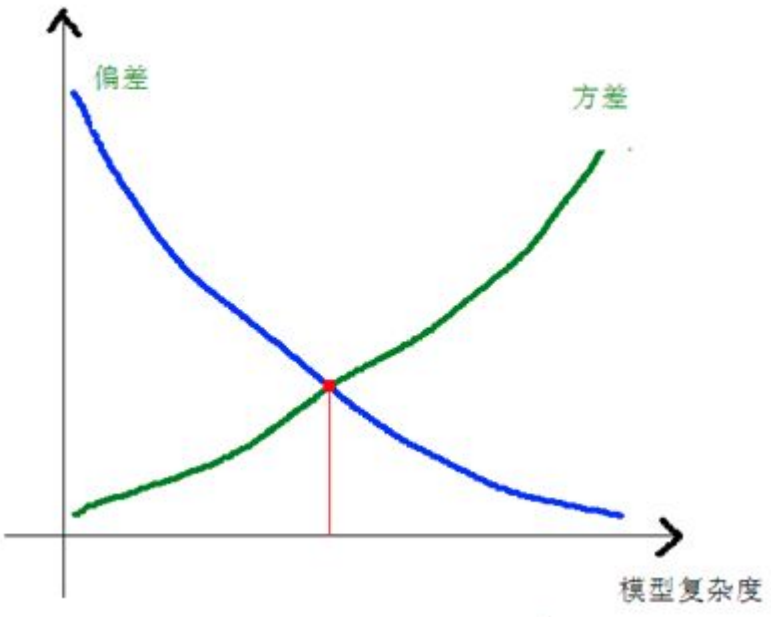
daBoost的优缺点
优点：（1）容易理解、实现简单
（2）易编码
（3）分类精度高
（4）可以使用各种回归模型构建基分类器，非常灵活
（5）作为二元分类器是，构造简单、结果可理解、少参数
（6）相对来说，不宜过拟合

152

集成学习与方差偏差

我觉得，避免偏差的话，首先我们需要尽量选择正确的模型，所谓“对症下药”。我觉得有位同行把机器学习算法的使用比作医生开药方，是非常不错的比喻。我们要根据数据的分布和特点，选择合适的算法。

其次，有了合适的算法，我们还要慎重选择数据集的大小。通常训练数据集越大越好，但是当大到数据集已经对整体所有数据有了一定的代表性之后，再多的数据已经不能提升模型的准确性，反而带来模型训练的计算量增加。但是，训练数据太少的话是一定不好的，这会带来过拟合的问题，过拟合就是模型复杂度太高，方差很大，不同的数据集训练出来的模型变化非常大



偏差与方差

152

从集成学习到模型的偏差和方差的理解

使用sklearn进行集成学习——理论

GBDT算法特征重要程度计算

机器学习中，有哪些特征选择的工程方法?

为什么说bagging是减少variance，而boosting是减少bias?

从机制上讲 为什么说bagging是减少variance，而boosting是减少bias

若各**子模型独立**，则有

，此时可以显著降低variance。若**各子模型完全相同**，则

Bagging 是 Bootstrap Aggregating 的简称，意思就是再取样 (Bootstrap) 然后在每个样本上训练出来的模型取平均。

$$E[\frac{\sum X_i}{n}] = E[X_i]$$

，所以从偏差上看没有降低，但是由于各个子模型是单独训练的，有一定的独立性，所以方差降低比较多,**提高泛化能力**。特别是random forest这种方式，不仅对样本取样，还有特征取样。

boosting从优化角度来看，是用forward-stagewise这种贪心法去最小化损失函数，在这个过程中偏差是逐步减小的，而由于各阶段分类器之间相关性较强，方差降低得少。

举个例子
gbdt是boosting的方式，它的决策树的深度比较小，模型会欠拟合，刚开始偏差大，后来就慢慢变小了。

为什么把特征组合之后还能提升

反正这些基本都是增强了特征的表达能力，或者说更容易线性可分吧

总体性问题

分类和回归的区别在于输出变量的类型。

152
定量输出称为回归，或者说是连续变量预测；
定性输出称为分类，或者说是离散变量预测。

生成模型与判别模型的区别

有监督机器学习方法可以分为生成方法和判别方法（常见的生成方法有混合高斯模型、朴素贝叶斯法和隐形马尔科夫模型等，常见的判别方法有SVM、LR等），生成方法学习出的是生成模型，判别方法学习出的是判别模型。

监督学习，预测时，一般都是在求p(Y|X)生成模型： 从数据中学习联合概率分布p(X,Y)，然后利用贝叶斯公式求：

，比如说朴素贝叶斯

判别模型：直接学习P(Y|X)， 它直观输入什么特征X，就直接预测出最可能的Y; 典型的模型包括：LR, SVM,CRF,Boosting,Decision tree....

生成方法的特点：生成方法可以还原联合概率分布，而判别方法则不能；生成方法的学习**收敛速度更快**，即当样本容量增加的时候，学习的模型可以更快的收敛于真实的模型；当存在**隐变量**时，仍可以用生成方法学习，此时判别方法就不能用。

判别方法的特点：判别方法直接学习的是条件概率或者决策函数，直接面对预测，往往学习的准确率更高；由于直接学习或者，可以对数据进行各种程度上的抽象、定义特征并使用特征，因此可以**简化学习问题**。

152

精确率、召回率、F1 值、ROC、AUC 各自的优缺点是什么?

		实际表现	
		1	0
预测表现	1	True Positive(TP)	False Positive(FP)
	0	False Negative(FN)	True Negative(TN)

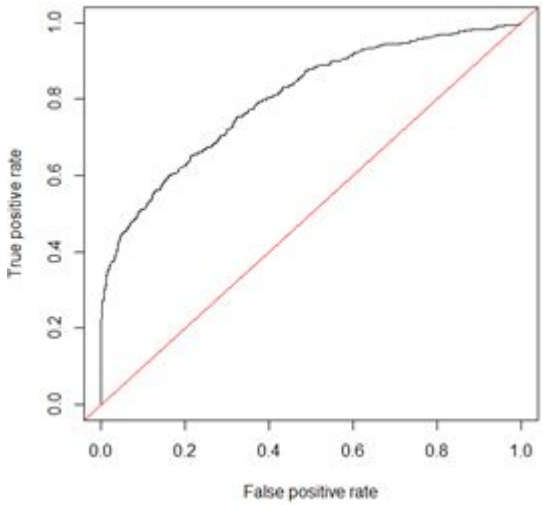
enter description here

精确率（Precision）为TP/(TP+FP)

召回率（Recall）为TP/(TP+FN)

ROC曲线（Receiver operating characteristic curve），ROC曲线其实是多个混淆矩阵的结果组合，如果在上述模型中我们没有定好阈值，而是将模型预测结果从高到低排序，将每个概率值依次作为阈值，那么就有多个混淆矩阵。对于每个混淆矩阵，我们计算两个指标TPR（True positive rate）和FPR（False positive rate），TPR=TP/(TP+FN)=Recall，**TPR就是召回率**，FPR=FP/(FP+TN)。

152



enter description here

在画ROC曲线的过程中，若有一个阈值，高于此阈值的均为坏人，低于此阈值的均为好人，则认为此模型已完美的区分开好坏用户。此时坏用户的预测准确率（TPR）为1，同时好用户的预测错误率（FPR）为0，ROC曲线经过（0,1）点。AUC（Area Under Curve）的值为ROC曲线下面的面积，若如上所述模型十分准确，则AUC为1。但现实生活中尤其是工业界不会有如此完美的模型，一般AUC均在0.5到1之间，AUC越高，模型的区分能力越好

若AUC=0.5，即与上图中红线重合，表示模型的区分能力与随机猜测没有差别。

所以AUC表征的是模型的分类能力。

152

过拟合

如果一味的去提高**训练数据的预测能力**，所选模型的复杂度往往会很高，这种现象称为过拟合。所表现的就是模型训练时候的误差很小，但在测试的时候误差很大。

产生的原因
因为参数太多，会导致我们的模型复杂度上升，容易过拟合

权值学习迭代次数足够多(Overtraining),拟合了训练数据中的噪声和训练样例中没有代表性的特征.

解决方法

> 交叉验证法
> 减少特征
> 正则化
> **权值衰减**
> 验证数据

线性分类器与非线性分类器的区别以及优劣

如果模型是参数的线性函数，并且存在线性分类面，那么就是线性分类器，否则不是。
常见的线性分类器有：LR,贝叶斯分类，单层感知机、线性回归
常见的非线性分类器：决策树、RF、GBDT、多层感知机
SVM两种都有(看线性核还是高斯核)
152

线性分类器速度快、编程方便，但是可能拟合效果不会很好
非线性分类器编程复杂，但是效果拟合能力强

特征比数据量还大时，选择什么样的分类器?
线性分类器，因为维度高的时候，数据一般在维度空间里面会比较稀疏，很有可能线性可分
对于维度很高的特征，你是选择线性还是非线性分类器?
理由同上
对于维度极低的特征，你是选择线性还是非线性分类器?
非线性分类器，因为低维空间可能很多特征都跑到一起了，导致线性不可分

样本不均衡如何解决

从重采样到数据合成

主要三个方面，数据，模型和评估方法。

数据上重采样和欠采样，使之均衡；

模型上选对样本不均衡问题不敏感的模型，和算法集成技术，如决策树，不能用KNN；

评估方法，用查全率，查准率之类

重采样（resampling）技术：

(1). **随机欠采样**

152

随机欠采样的目标是通过随机地消除占多数的类的样本来平衡类分布。

优点

它可以提升运行时间；并且当训练数据集很大时，可以通过减少样本数量来解决存储问题。

缺点

它会丢弃对构建规则分类器很重要的有价值的潜在信息。

被随机欠采样选取的样本可能具有偏差。它不能准确代表大多数。

(2). **随机过采样**（Random Over-Sampling）

过采样（Over-Sampling）通过随机复制少数类来增加其中的实例数量，从而可增加样本中少数类的代表性。

优点

与欠采样不同，这种方法不会带来信息损失。

表现优于欠采样。

缺点

由于复制少数类事件，它加大了**过拟合**的可能性。

(3). 信息性过采样：**合成少数类**过采样技术

直接复制少数类实例并将其添加到主数据集时。从少数类中把一个数据子集作为一个实例取走，接着创建相似的新合成的实例。这些合成的实例接着被添加进原来的数据集。新数据集被用作样本以训练分类模型。

优点

通过随机采样生成的合成样本而非实例的副本，可以缓解过拟合的问题。

不会损失有价值信息。

缺点

当生成合成性实例时，SMOTE 并不会把来自其他类的相邻实例考虑进来。这导致了类重叠的增加，并会引入额外的噪音。

深度学习方面的问题

机器学习岗位面试问题汇总 之 深度学习
152

深度学习的实质 及其 与浅层学习的区别

深度学习实质：多隐层+海量数据——>学习有用特征——>提高分类或预测准确性
区别： （1）DL强调模型深度
(2）DL突出特征学习的重要性：特征变换+非人工

BP算法为什么不能适应于深度学习

BP为传统多层感知机的训练方法，<=5层
问题： （1）梯度越来越稀疏（梯度扩散<—-非凸目标函数）
（2）局部最小
（3）一般，有标签
NOTE：解决其中局部最小值的方法： （1）多组不同随机参数，取最好参数 （2）启发式优化算法：模拟退火 或 遗传 （3）随机梯度下降

CNN卷基层和pooling层的作用

152

DNN常用的激活函数有哪些，各有什么特点

（1）sigmoid：易饱和（梯度消失），非0均值 （2）tanh，改进了sigmoid的第二个缺点，即它是0均值的 （3）ReLU，收敛快（不容易饱和），求梯度简单（没有指数计算，只需要阈值就可以），有稀疏特性。缺点是**神经元容易坏死**。

2行

由于ReLU在x<0时梯度为0，这样就导致负的梯度在这个ReLU被置零，而且这个神经元有可能再也不会被任何数据激活。如果这个情况发生了，那么这个神经元之后的梯度就永远是0了，也就是ReLU神经元坏死了，不再对任何数据有所响应。实际操作中，如果你的learning rate 很大，那么很有可能你网络中的40%的神经元都坏死了

解决relu神经元坏死问题

当然，如果你设置了一个合适的较小的learning rate，这个问题发生的情况其实也不会太频繁。

relu的变种 leaky-relu:

，

，

这里的 α 是一个很小的常数。这样，即修正了数据分布，又保留了一些负轴的值，使得负轴信息不会全部丢失。

152

Parametric ReLU： 对于 Leaky ReLU 中的α，通常都是通过先验知识人工赋值的。
然而可以观察到，损失函数对α的导数我们是可以求得的，可不可以将它作为一个参数进行训练呢

Randomized ReLU：
Randomized Leaky ReLU 是 leaky ReLU 的random 版本 ,核心思想就是，在训练过程中，α 是从一个高斯分布 U(l,u) 中 随机出来的，然后再测试过程中进行修正（有点像dropout的用法)

什么样的资料不适合用深度学习?

（1）数据量小 （2）没有**局部相关性**

什么是共线性，跟过拟合有何关联?

共线性：高度相关—>冗余——>过拟合
解决：排除相关、加入权重正则

pooling的结果是使得特征减少，参数减少，但pooling的目的并不仅在于此。pooling目的是为了保持某种不变性（**平移**），常用的有**mean-pooling，max-pooling和Stochastic-pooling**三种。

152

mean-pooling，即对邻域内特征点只求平均，max-pooling，即对邻域内特征点取最大。根据相关理论，特征提取的误差主要来自两个方面：（1）邻域大小受限造成的估计值方差增大；（2）卷积层参数误差造成估计均值的偏移。一般来说，**mean-pooling能减小第一种误差，更多的保留图像的背景信息，max-pooling能减小第二种误差，更多的保留纹理信息**。Stochastic-pooling则介于两者之间，通过对像素点按照数值大小赋予概率，再按照概率进行亚采样，在平均意义上，与mean-pooling近似，在局部意义上，则服从max-pooling的准则。

LeCun的“Learning Mid-Level Features For Recognition”对前两种pooling方法有比较详细的分析对比，如果有需要可以看下这篇论文。

其实pooling的目的就是为了使参数量减少，因为根本不需要那么多参数。pooling也只能做到在极小范围内的平移不变性，旋转和 伸缩是做不到的。其实不变性都是特征工程时代的概念了，现在在数据量极大的情况下，样本覆盖了足够多的variance，dnn自动就会把各种不变性学习出来

使用Pooling的目的之一是获取一定的特征不变性，目前**用的比较多的是Max Pooling**。
max pooling是DCNN的非线性来源之一，然后在现代的深度神经网络中，最大的非线性来源是ReLU类的激活函数。
因此，目前对使用Pooling也存在一定的争议，一些最新的工作已经不在网络的中间层使用pooling层了（或者只在最后一层使用average pooling，比如说network in network)。

缺点在于会丢失信息。

对于mean pooling，真的是好简单：假设pooling的窗大小是2x2, 在forward的时候啊，就是在前面卷积完的输出上依次不重合的取2x2的窗平均，得到一个值就是当前mean pooling之后的值。**backward的时候，把一个值分成四等分放到前面2x2的格子**里面就好了。如下

152

forward: [1 3; 2 2] -> 2

backward: 2 -> [0.5 0.5; 0.5 0.5]

max pooling就稍微复杂一点，forward的时候你只需要把2x2窗子里面那个最大的拿走就好了，**backward的时候你要把当前的值放到之前那个最大的位置，其他的三个位置都弄成0**。如下

forward: [1 3; 2 2] -> 3

backward: 3 -> [0 3; 0 0]

特征选择的方法

机器学习中，有哪些特征选择的工程方法?

特征选择是特征工程中的重要问题（另一个重要的问题是特征提取），坊间常说：**数据和特征决定了机器学习的上限**，而模型和算法只是逼近这个上限而已。由此可见，特征工程尤其是特征选择在机器学习中占有相当重要的地位。

特征选择方法举例

计算每一个特征与响应变量的**相关性**：工程上常用的手段有计算皮尔逊系数和互信息系数，皮尔逊系数只能衡量线性相关性而互信息系数能够很好地度量各种相关性

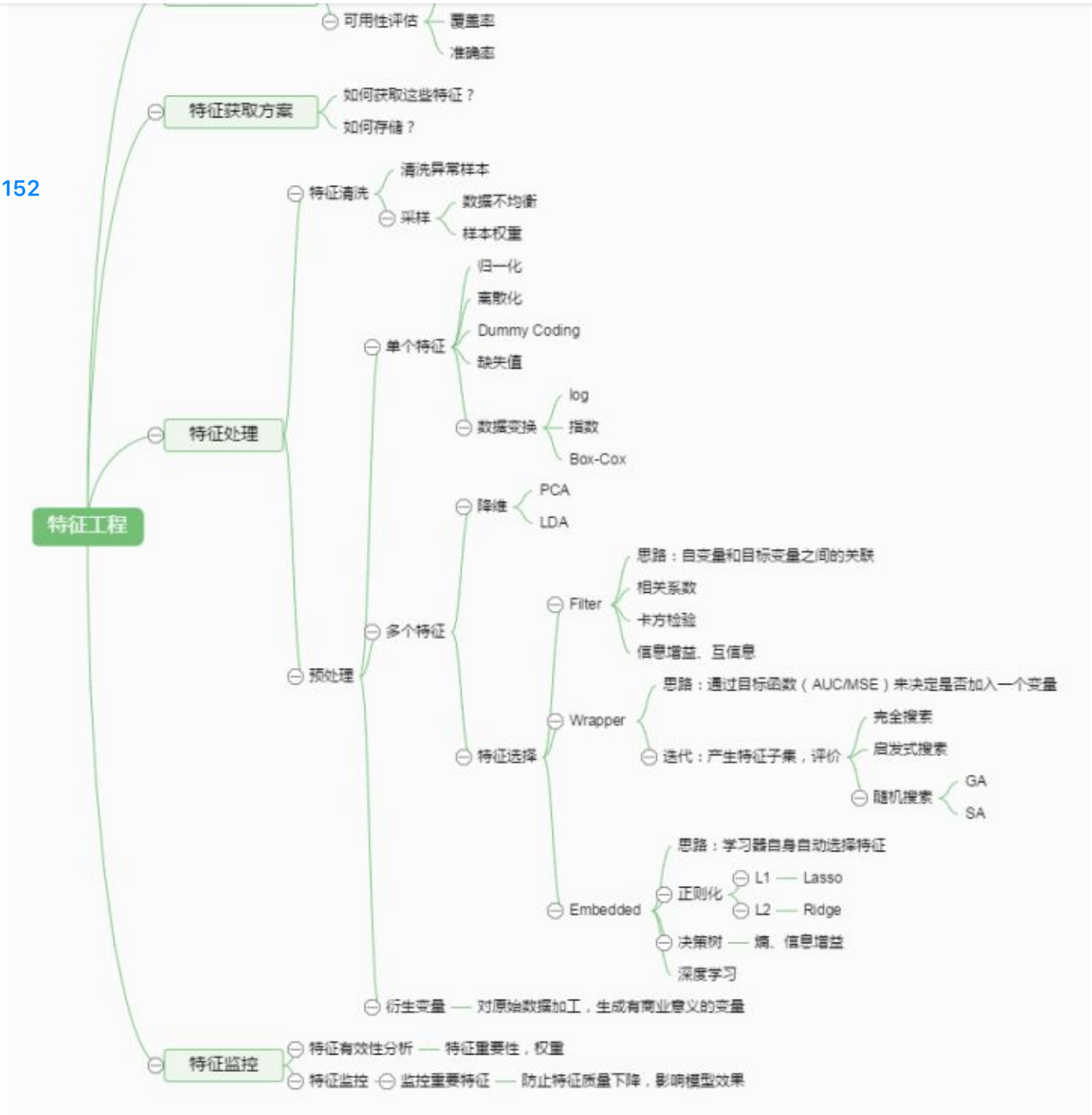
通过**L1正则项**来选择特征：L1正则方法具有稀疏解的特性，因此天然具备特征选择的特性，但是要注意，L1没有选到的特征不代表不重要，原因是两个具有高相关性的特征可能只保留了一个，如果要确定哪个特征重要应再通过L2正则方法交叉检验；

训练能够**对特征打分的预选模型**：RandomForest和Logistic Regression等都能对模型的特征打分，通过打分获得相关性后再训练最终模型；

通过深度学习来进行特征选择：目前这种手段正在随着深度学习的流行而成为一种手段，尤其是在计算机视觉领域，原因是深度学习具有自动学习特征的能力.

特征选择方法分类

152



特征选择思维导图

Wrapper：包装法，根据目标函数（通常是预测效果评分），每次选择若干特征，或者排除若干特征。

Embedded：集成方法，先使用某些机器学习的算法和模型进行训练，**得到各个特征的权值系数**，根据系数从大到小选择特征。类似于Filter方法，但是是通过训练来确定特征的优劣。

降维：PCA LDA等。

Filter过滤法

方差选择法

使用方差选择法，先要计算各个**特征的方差**，然后根据阈值，选择方差大于阈值的特征

相关系数法

使用相关系数法，先要计算各个特征对目标值的相关系数以及相关系数的P值

卡方检验

经典的卡方检验是检验定性自变量对定性因变量的**相关性**

互信息法

经典的互信息也是评价定性自变量对定性因变量的相关性的

Embedded 集成方法

基于**树模型**的特征选择法
树模型中GBDT也可用来作为基模型进行特征选择

深度学习方法
152

降维

将原始的样本映射到维度更低的样本空间中。

PCA是为了让映射后的样本具有最大的**发散性**；而LDA是为了让映射后的样本有最好的**分类性能**。所以说PCA是一种无监督的降维方法，而LDA是一种有监督的降维方法。

LR相关问题

LR与BP

BP神经网络是否优于logistic回归?

首先，神经网络的最后一层，也就是输出层，是一个 Logistic Regression （或者 Softmax Regression），也就是一个线性分类器，中间的隐含层起到特征提取的作用，把隐含层的输出当

神经网络的训练，实际上就是同时训练特征提取算法以及最后的 Logistic Regression的参数。为什么要特征提取呢，因为 Logistic Regression 本身是一个线性分类器，所以，通过特征提取，我们可以把原本线性不可分的数据变得线性可分。要如何训练呢，最简单的方法是**（随机，Mini batch）梯度下降法**

152

LR为什么使用sigmoid函数

源于sigmoid，或者说exponential family所具有的最佳性质，即**maximum entropy**的性质。maximum entropy给了logistic regression一个很好的数学解释。为什么maximum entropy好呢？entropy翻译过来就是熵，所以maximum entropy也就是最大熵。熵用在概率分布上可以表示这个**分布中所包含的不确定度**，熵越大不确定度越大。均匀分布熵最大，因为基本新数据是任何值的概率都均等。而我们现在关心的是，给定某些假设之后，熵最大的分布。也就是说这个分布应该在满足我假设的前提下越均匀越好。比如大家熟知的正态分布，正是假设已知mean和variance后熵最大的分布。首先，我们在建模预测 Y|X，并认为 Y|X 服从**bernoulli distribution**，所以我们只需要知道 P(Y|X)；其次我们需要一个线性模型，所以 P(Y|X) = f(wx)。接下来我们就只需要知道 f 是什么就行了。而我们可以通过最大熵原则推出的这个 f，就是sigmoid。

面试问了如何在海量数据中查找给定部分数据最相似的top200向量，向量的维度也很高. 因为之前了解过其他面蚂蚁金服的朋友，也有问到这个题目的
所以反应比较快，直接就说可以用KD树，聚类，hash,
一天之内两连面，还是问了很多机器学习算法的东西 为什么LR需要归一化或者取对数，为什么LR把特征离散化后效果更好 为什么把特征组合之后还能提升，反正这些基本都是增强了特征的表达能力，或者更容易线性可分吧

在logistic regression （LR）中，这个目标是什么呢？最大化条件似然度。考虑一个二值分类问题，训练数据是一堆（特征，标记）组合， (x1,y1), (x2,y2), ... 其中x是特征向量，y是类标记（y=1表示正类，y=0表示反类）。LR首先定义一个条件概率p(y|x；w) 。 **p(y|x；w）表示给定特**

152

为什么LR把特征离散化后效果更好

逻辑回归属于广义线性模型，表达能力受限；单变量离散化为N个后，**每个变量有单独的权重**，相当于为模型**引入了非线性，能够提升模型表达能力**，加大拟合；(哑变量)
特征离散化以后，起到了简化了逻辑回归模型的作用，降低了模型过拟合的风险。

连续特征的离散化：在什么情况下将连续的特征离散化之后可以获得更好的效果?

在工业界，很少直接将连续值作为逻辑回归模型的特征输入，而是将连续特征离散化为一系列0、1特征交给逻辑回归模型，这样做的优势有以下几点：

离散特征的增加和减少都很容易，易于模型的快速迭代；

稀疏向量内积乘法运算速度快，计算结果方便存储，容易扩展；

1. 离散化后的特征对**异常数据**有很强的鲁棒性：比如一个特征是年龄>30是1，否则0。如果特征没有离散化，一个异常数据“年龄300岁”会给模型造成很大的干扰；

1. 逻辑回归属于广义线性模型，表达能力受限；**单变量离散化为N个后，每个变量有单独的权重，相当于为模型引入了非线性**，能够提升模型表达能力，加大拟合；

152

1. **特征离散化后，模型会更稳定**，比如如果对用户年龄离散化，20-30作为一个区间，不会因为一个用户年龄长了一岁就变成一个完全不同的人。当然处于区间相邻处的样本会刚好相反，所以怎么划分区间是门学问；

1. 特征离散化以后，起到了简化了逻辑回归模型的作用，降低了模型过拟合的风险。

李沐曾经说过：模型是使用离散特征还是连续特征，其实是一个“**海量离散特征+简单模型**” 同 “少量连续特征+复杂模型”的权衡。既可以离散化用线性模型，也可以用连续特征加深度学习。就看是喜欢折腾特征还是折腾模型了。通常来说，前者容易，而且可以n个人一起并行做，有成功经验；后者目前看很赞，能走多远还须拭目以待。

如何用LR建立一个广告点击的模型：

特征提取—>特征处理（**离散化**、**归一化**、onehot等）—>找出候选集—->模型训练，得到结果

LR的过拟合

正则化（为了方便求解，L2使用较多）

添加正则化后的损失函数变为：

152

同时w的更新变为：

关于LR的多分类：softmax

这里会输出当前样本下属于哪一类的概率，并且满足全部概率加起来=1

关于softmax和k个LR的选择

如果类别之间是否互斥（比如音乐只能属于古典音乐、乡村音乐、摇滚月的一种）就用softmax
否则类别之前有联系（比如一首歌曲可能有影视原声，也可能包含人声，或者是舞曲），这个时候

实现简单；

分类时计算量非常小，速度很快，存储资源低；

缺点：

容易欠拟合，一般准确度不太高

152

只能处理两分类问题

SVM相关问题

解密SVM系列（一）：关于拉格朗日乘子法和KKT条件

svmw问题整理

SVM的主要特点

（1）非线性映射-理论基础

（2）最大化分类边界-方法核心

（3）支持向量-计算结果

（4）小样本学习方法，最终的决策函数只有少量支持向量决定，避免了“维数灾难”，少数支持向量决定最终结果—->可“剔除”大量冗余样本+算法简单+具有鲁棒性

（7）学习问题可表示为凸优化问题—->全局最小值

（8）可自动通过最大化边界控制模型，但需要用户指定核函数类型和引入松弛变量

（9）适合于小样本，**优秀泛化能力（因为结构风险最小）**

（10）泛化错误率低，分类速度快，结果易解释

缺点：

152

（1）**大规模训练样本**（m阶矩阵计算）

（2）传统的不适合多分类

（3）对缺失数据、参数、核函数敏感

为什么要引入对偶问题

（1）容易求解 （2）核函数

Note：拉格朗日对偶没有改变最优解，但改变了算法复杂度：原问题—样本维度；对偶问题–样本数量。所以 线性分类&&样本维度<样本数量：原问题求解（liblinear默认）；

非线性–升维—一般导致 样本维度>样本数量：对偶问题求解

样本失衡的影响

超平面会靠近样本少的类别。因为使用的是软间隔分类，而如果对所有类别都是使用同样的惩罚系数，则由于优化目标里面有最小化惩罚量，所以靠近少数样本时，其惩罚量会少一些。

对正例和负例赋予不同的C值，例如正例远少于负例，则正例的C值取得较大，这种方法的缺点是可能会偏离原始数据的概率分布；

152

样本失衡时，如何评价分类器的性能好坏?

使用ROC曲线

样本没有规范化对SVM有什么影响?

对偶问题的优化目标函数中有向量的内积计算(优化过程中也会有内积计算的，见SMO)，径向基核函数中有向量的距离计算，存在值域小的变量会被忽略的问题，影响算法的精度。参考

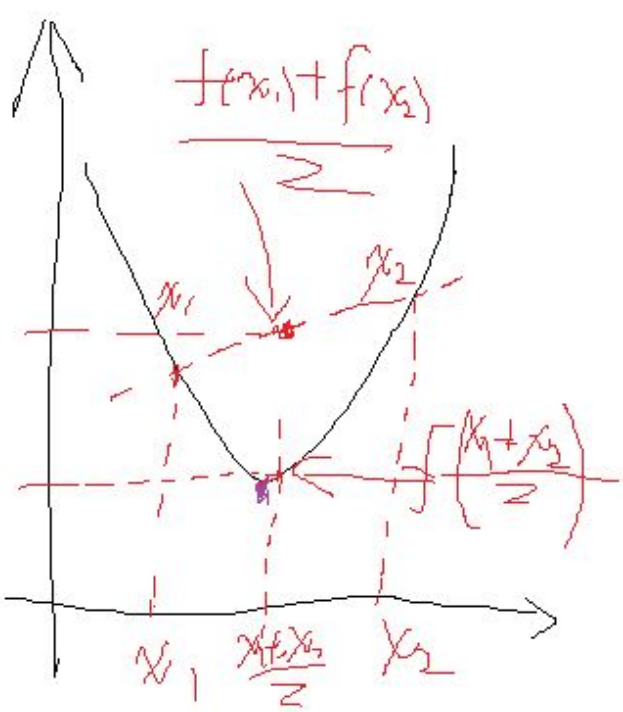
数据维度大于数据量的对SVM的影响?

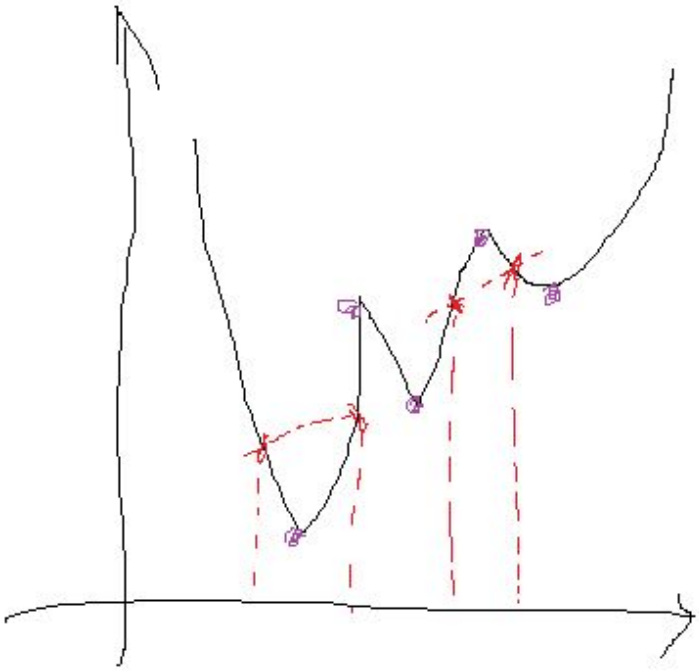
这种情况下一般采用线性核(即无核)，因为此时特征够用了(很大可能是线性问题)，没必要映射到更高维的特征空间。

拉格朗日乘子法 和KKT条件

凸函数

前提条件凸函数：下图左侧是凸函数。





左侧是凸函数

凸的就是开口朝一个方向（向上或向下）。更准确的数学关系就是：

$$\frac{f(x_1)+f(x_2)}{2} > f(\frac{x_1+x_2}{2})$$

152

enter description here

或者

$$\frac{f(x_1)+f(x_2)}{2} < f(\frac{x_1+x_2}{2})$$

对于凸问题，你去求导的话，是不是只有一个极点，那么他就是最优点，很合理。

等式条件约束

当带有约束条件的凸函数需要优化的时候，一个带等式约束的优化问题就通过拉格朗日乘子法完美的解决了。

可以使用

152

这里可以看到与α1,α2相乘的部分都为0，根原来的函数是等价的。所以α1,α2的取值为全体实数。现在这个优化目标函数就没有约束条件了吧。然后求导数。

不等式约束与KKT条件

任何原始问题约束条件无非最多3种，等式约束，大于号约束，小于号约束，而这三种最终通过将约束方程化简化为两类：约束方程等于0和**约束方程小于0**。

现在将约束拿到目标函数中去就变成：

其中g是不等式约束，h是等式约束（像上面那个只有不等式约束，也可能有等式约束）。那么KKT条件就是函数的最优值必定满足下面条件：

(1) L对各个x求导为零；

(2) h(x)=0;

(3) ，

152

第三个式子不好理解，因为我们知道在约束条件变完后，所有的g(x)<=0，且αi≥0，然后求和还要为0，无非就是告诉你，**要么某个不等式gi(x)=0,要么其对应的αi=0**。那么为什么KKT的条件是这样的呢?

SVM的原问题和对偶问题

原问题

$$\max_{w,b} \frac{1}{||w||} = \min_{w,b} \frac{1}{2}||w||^2$$
$$\text{s,t} \quad y_i(w^T x_i + b) \geq 1, i \in \{1, ..., N\}$$

原问题

拉格朗日乘子法结果

$$\max_{\alpha,\beta,\alpha_i\geq 0}\min_{w} L(w,\alpha,\beta)=\max_{\alpha,\alpha_i\geq 0}\min_{w,b}\{\frac{1}{2}||w||^2-\sum_{i=1}^{N}\alpha_i[y_i(w^Tx_i+b)-1]\}$$

152

对偶问题

求导得到

$$\frac{\partial L}{\partial w}=w-\sum_{i=1}^{N}\alpha_i y_i x_i=0\Rightarrow w=\sum_{i=1}^{N}\alpha_i y_i x_i$$
$$\frac{\partial L}{\partial b}=-\sum_{i=1}^{N}\alpha_i y_i=0\Rightarrow\sum_{i=1}^{N}\alpha_i y_i=0$$

求导得到

代入乘子算式得到

$$W(\alpha) = L(w,b,\alpha) = \frac{1}{2}(\sum_{i=1} \alpha_i y_i x_i)^T (\sum_{j=1} \alpha_j y_j x_j) -$$

$$\sum_{i=1}^{N} \alpha_i y_i ((\sum_{i=1}^{N} \alpha_i y_i x_i) x_i + b) + \sum_{i=1}^{N} \alpha_i$$

152

$$= \frac{1}{2}(\sum_{i,j=1}^{N} \alpha_i y_i \alpha_j y_j x_i * x_j) - \sum_{i,j=1}^{N} \alpha_i y_i \alpha_j y_j x_i * x_j + b \sum_{i=1}^{N} \alpha_i y_i + \sum_{i=1}^{N} \alpha_i$$

$$= -\frac{1}{2}(\sum_{i,j=1}^{N} \alpha_i y_i \alpha_j y_j x_i * x_j) + \sum_{i=1}^{N} \alpha_i$$

对偶结果

就得到的原问题的对偶问题

$$max \quad W(\alpha) = -\frac{1}{2}(\sum_{i,j=1}^{N} \alpha_i y_i \alpha_j y_j x_i * x_j) + \sum_{i=1}^{N} \alpha_i$$

$$s.t. \quad \alpha_i \geq 0$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

对偶问题

为什么要引入对偶算法

对偶问题往往更加容易求解(结合拉格朗日和kkt条件)

152

可以很自然的引用**核函数**（拉格朗日表达式里面有内积，而核函数也是通过内积进行映射的）

SVM解决过拟合的方法

决定SVM最优分类超平面的恰恰是那些占少数的支持向量，如果支持向量中碰巧存在异常点就会过拟合，解决的方法是加入松弛变量。

另一方面从损失函数角度看，引入了L2正则。

为什么要把原问题转换为对偶问题?

因为原问题是凸二次规划问题，转换为对偶问题更加高效。

为什么求解对偶问题更加高效?

因为只用求解alpha系数，而alpha系数只有支持向量才非0，其他全部为0.

alpha系数有多少个?

样本点的个数

L1还可以用来选择特征
152

A 为什么**L1可以用来选择特征**

B 因为L1的话会把某些不重要的特征压缩为0

A 为什么L1可以把某些特征压缩为0

B 因为（画图）L1约束是正方形的，经验损失最有可能和L1的正方形的顶点相交，L1比较有棱角。所以可以把某些特征压缩为0

SVM优缺点

优点:

使用核函数可以向高维空间进行映射

使用核函数可以解决非线性的分类

分类思想很简单，就是将样本与决策面的间隔最大化

分类效果较好

对大规模数据训练比较困难

无法直接支持多分类，但是可以使用间接的方法来做
152

SMO算法

SMO
SMO是用于快速求解SVM的
它选择凸二次规划的两个变量，其他的变量保持不变，然后根据这两个变量构建一个二次规划问题，这个二次规划关于这两个变量解会更加的接近原始二次规划的解，通过这样的子问题划分可以大大增加整个算法的计算速度

SVM多分类问题

间接法
一对多

其中某个类为一类，其余n-1个类为另一个类，比如A,B,C,D四个类，第一次A为一个类，{B,C,D}为一个类训练一个分类器，第二次B为一个类,{A,C,D}为另一个类,按这方式共需要训练4个分类器，最后在测试的时候将测试样本经过这4个分类器f1(x),f2(x),f3(x)和f4(x),取其最大值为分类器(这种方式由于是1对M分类，会存在偏置，很不实用)

一对一(libsvm实现的方式)

任意两个类都训练一个分类器，那么n个类就需要个svm分类器。

还是以A,B,C,D为例,那么需要{A,B},{A,C},{A,D},{B,C},{B,D},{C,D}为目标共6个分类器，然后在预测的时候将测试样本通过这6个分类器之后进行投票选择最终结果。（这种方法虽好，但是需要个分类器代价太大，不过有好像使用循环图来进行改进）

reference

Linear SVM 和 LR 有什么异同?

SVM和logistic回归分别在什么情况下使用?

百度 – 机器学习面试

svmw问题整理

各种机器学习的应用场景分别是什么？例如，k近邻,贝叶斯，决策树，svm，逻辑斯蒂回归

机器学习面试问题汇总

机器学习面试

天池离线赛 - 移动推荐算法（四）： 基于LR, RF, GBDT等模型的预测

152

机器学习常见算法个人总结（面试用）

机器学习面试问题汇总

cs229机器学习笔记及代码

腾讯17届校招面经合集

编辑于 2017-09-26

机器学习　深度学习（Deep Learning）　技术面

文章被以下专栏收录

机器学习与算法

关注专栏

推荐阅读

152
4 条评论

切换为时间排序

写下你的评论...

jasonfzs 2 个月前

好像链接打不开哦?

赞

斗战胜佛 2 个月前

我以为只有我一个人打不开，一个都不行

赞

alancheg 2 个月前

可能这只是个目录

1

WW吴小兀 2 个月前

什么都看不到啊，题主

赞