从wx+b开始--Logistic 回归,支持向量机和神经网络(pre)


KAMIWei (/u/05735d080ba8) ＋关注
2017.11.11 23:47* 字数 3962 阅读 93 评论 0 喜欢 2
(/u/05735d080ba8)

前言

人工智能近年来可以说是一门显学. 作为人工智能的一个子集, 机器学习的新成果新应用也是穷出不尽, 各种资料和教程也可谓是汗牛充栋. 然而, 这些资料或多或少给人一种照本宣科, 千篇一律的感觉, 而且常常让初学者读后似懂未懂.

在这里, 我会尝试从几个耳熟能详的模型的相似处作为切入点, 穿插相关理论和概念, 希望读者能从中窥视一斑机器学习的原理, 过程和局限.

wx+b

我们常说, 机器学习有三大任务: 回归, 分类, 聚类. 从某种意义上讲, 实际这三种任务的"目标"和"内涵"是相似的, 区别主要在于是否有监督, 值是否连续等. 下面我们会选取分类作为我们的讨论场景.

在许多机器学习模型中, 我们往往会看到 wx+b 这么一个表达式. x 表示样本, 样本一般有多个属性, 所以 x 一般是向量. 从几何角度来说, wx+b 可以看做是一个超平面. 从代数的角度讲, 是一个"加权+偏置" 的操作, w为权重, b为偏置. 在往后的章节, 我们对以这个为切入点, 看 wx+b 是怎么样因为思想和策略不同, 衍生成各种不同的机器学习模型.

顺便一提, 通过简单的变换, 位移项/偏置项 b 是能够收入到 w 和 x 中. 为表述方便, 我们有时会讨论 wx, 有时会讨论 wx+b, 两者的含义是一样的.

回归

原始二分类模型和损失函数

首先, 我们来考虑一个简单的二分类问题: 给出一组已标记为正例或负例的样本, 要求从这组样本中学习出一个二分类器, 可以用来预测未知标记样本.

一个非常直观自然的想法就是: 设法根据样本找到一个超平面, 使得尽可能多的正例位于该超平面的一侧, 尽可能多的负例都位于超平面的另一侧. 那么, 这个超平面就可以作为一个分类器来使用. 若人为地定一下超平面方向, 那问题可转化为: 求一个超平面, 使得

$$\mathbf{w}^T\mathbf{x} + b$$

对于尽可能多的正例大于0, 尽可能多的负例小于0.
进一步地, 若将正例标签记为1, 负例标签记为0. 那么条件等价于尽可能多的样本满足

$$(2y - 1)(\mathbf{w}^T\mathbf{x} + b)$$

分类错误会造成"损失". 如果我们把分类正确的损失记为0, 分类错误的损失记为样本在错误的一侧到平面的距离(离得越远, 说明这个超平面对该样本越不合适). 那么样本 x^(i) 的分类损失有

$$\max(0, (1 - 2y)(\mathbf{w}^T\mathbf{x} + b))$$

自然, 模型造成的损失越小越好. 换言之, 我们希望二分类器模型的参数 w, 使得下面函数的取值越小越好.

$$\sum_{i=1}^{m} \max(0, (1 - 2y^{(i)})(\mathbf{w}^T\mathbf{x}^{(i)} + b))$$

这样的函数就是机器学习领域中常常讨论的"损失函数". 我们要做的是, 通过学习给出的样本, 寻找使得损失函数越尽可能小的模型.

优化器和训练

由上节我们得知, 给出样本后, 损失函数是一个关于模型参数的函数. 既然如此, 那么求损失函数的极小值点不就可以了吗? 理论上的是这样, 然而理想很丰满, 现实很骨感. 绝大部分的机器学习模型, 都很难找求出其极小点解析解, 或者不会去求其解析解, 固中原因无非是老生常谈的几点:

- 关于参数的方程组超定(噪声, 异常点, etc)
- 损失函数无法求导
- 运算代价太大

好在, 我们知道, 并不一定要通过求解析解才能求出其极小值点, 有梯度下降法, 最小二乘法, 牛顿法等各种优化方法可以利用. 以梯度下降法为例. 我们知道, 函数的梯度描述了函数值变化的"方向"和"陡峭程度". 那么, 我们先指定模型参数的一个初始值, 求其梯度, 然后让模型参数w的取值往损失函数的梯度的负方向走一小步, 再求其梯度. 如此循环, 收敛后的结果便是我们我们想要的模型参数.

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

图-1-a 梯度下降算法

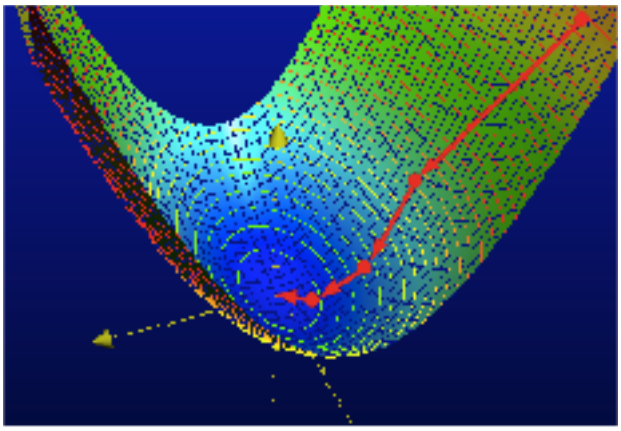


图-1-b 梯度下降算示意

这样的一组优化算法便是机器学习常讨论的"优化器". 该过程也称之为"训练", 用于优化过程的样本, 则称之为训练样本.

下降步长 α 也称为学习率. 注意到学习率和模型本身没有直接, 我们也很难从模型和样本本身中获得关于它的经验值. 这样的参数便是机器学习领域常讨论的超参. 超参如何设置和调节是未决问题, 某种程度上依赖使用者的经验, 甚至是直觉.

Logisic回归

回到我们的二分类模型. 我们发现, 原始二分类模型的损失函数不可导, 这给模型的训练带来了困难.
下面让我们来找那么一个变换, 来完成这么两个任务:

1. 将wx+b映射到值域[0,1], 使得其与标签 y 有比较的意义(wx+b 和 y 是没有比较的意义的, 可以这么理解: 分类正确后, 距离无关要紧), 以移除min, max, sign等运算
2. 损失函数可导

sigmod是个单调可微函数, 值域在 [0, 1]. 十分合适.

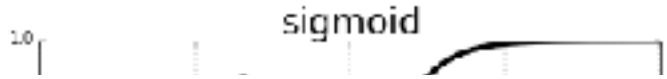


图-2 sigmod function

代入后, 有

$$y - \frac{1}{1 + e^{-(w^T x + b)}}$$

实际上等价于通过线性回归拟合

$$\ln \frac{y}{1 - y}$$

如果将 y 视为 样本为正例的概率. 几率为事件发生的概率与其不发生的概率的比. 也就是说, 我们用线性回归预测分类为正例的对数几率(logit). 这便是Logistic回归这个名称的由来.

GO DEEP

- 指数家族和广义线性模型
- 殊归同途: 交叉熵视角
- 非线性数据
- 超参与网格搜索
- 欠拟合, 过拟合与模型评价指标

支持向量机

Logistic回归的缺陷

Logistic回归十分优雅. 然而, 我们进一步对其损失函数分析可以发现其一个很明显的缺陷, Logistic回归对训练集的局部扰动很敏感. 如下图所示. 训练集发生局部扰动后, logistic回归产生的超平面被"拽"歪了, 还出现了原来不存在的分类错误. 由于训练集的局限性和噪声, 这样的现象非常普遍.

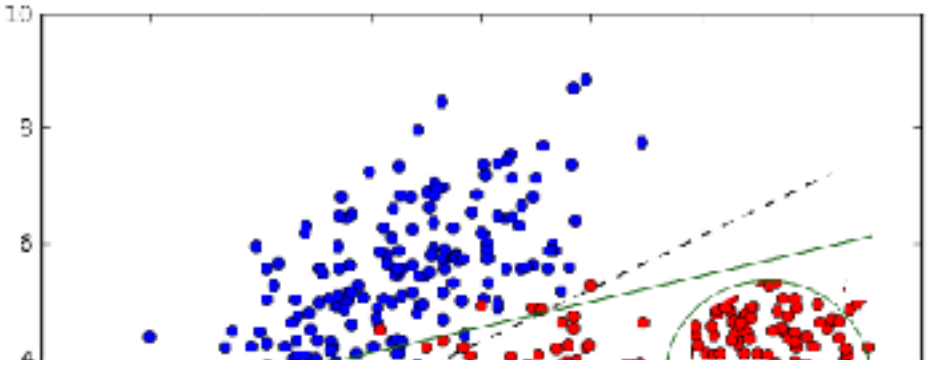
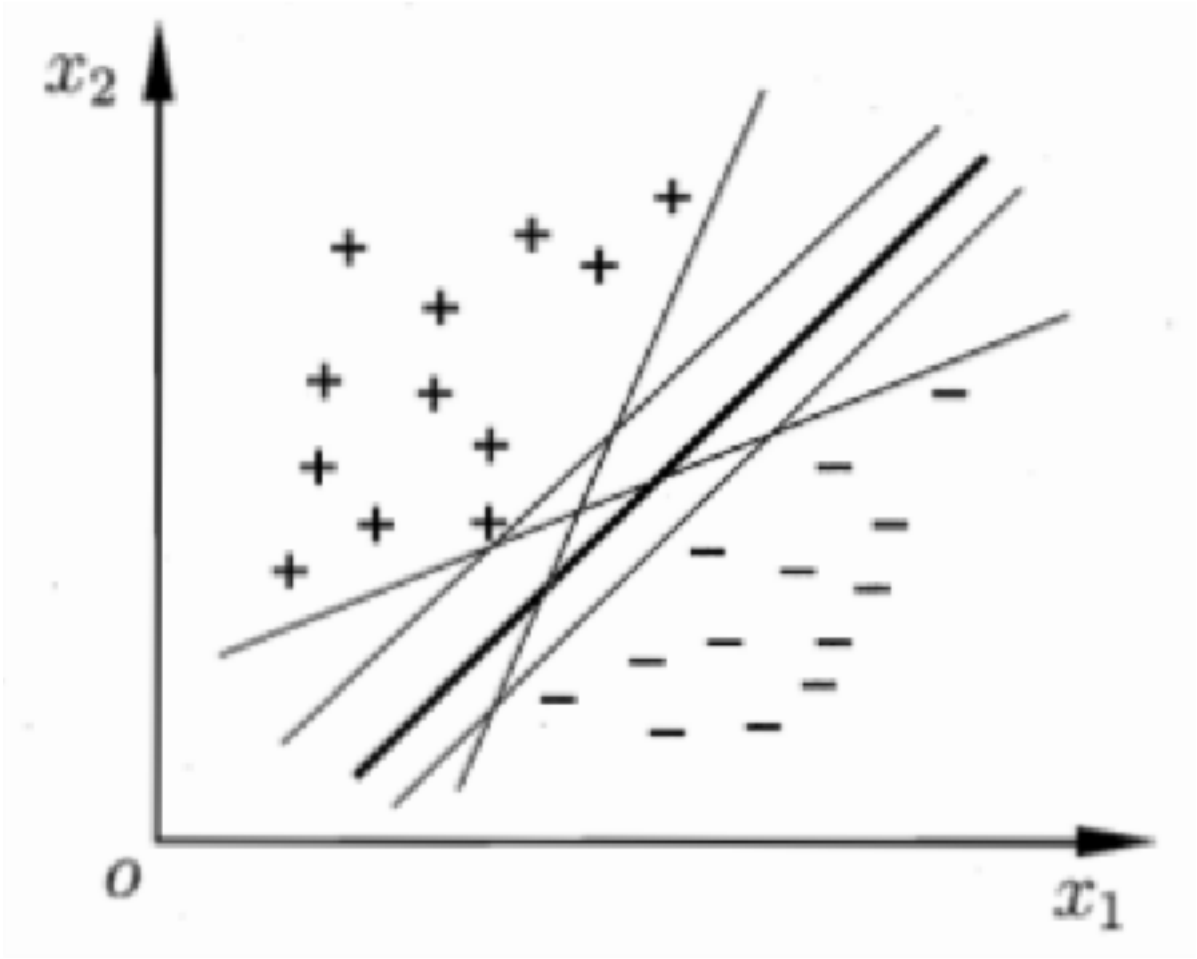


图2-1 Logistic 回归与局部扰动

SVM基本型

直观上看, 我们应该去找位于两类训练样本的"正中间"的划分超平面, 这样的超平面收局部扰动的影响最小, 也就是拥有对未知样本有最强的泛化性能和最好的Robust表现. 这便是SVM的基本思想.



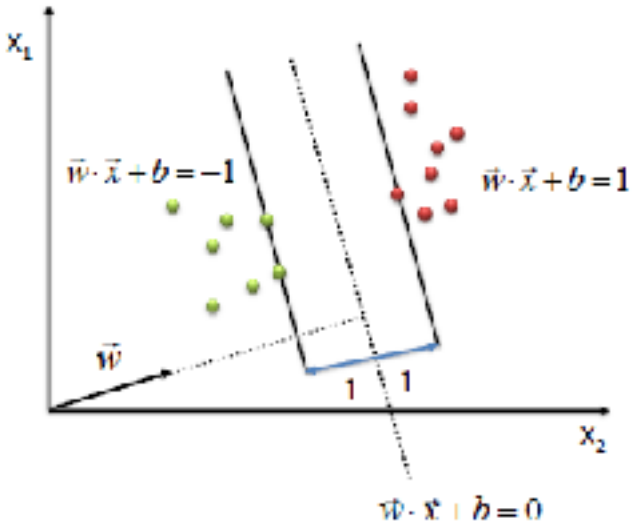
假设超平面能够将训练样本正确分类. 同前面对原始二分类模型的讨论, 一定存在正数, 使得

$$\begin{cases} \mathbf{w}'^T\mathbf{x}^{(i)} + b' \geq +\xi, & y^{(i)} = +1 \\ \mathbf{w}'^T\mathbf{x}^{(i)} + b' \leq -\xi, & y^{(i)} = -1 \end{cases}$$

进行缩放, 有

$$\begin{cases} \mathbf{w}^T\mathbf{x}^{(i)} + b \geq +1, & y^{(i)} = +1 \\ \mathbf{w}^T\mathbf{x}^{(i)} + b < -1, & y^{(i)} = -1 \end{cases}$$

距离超平面最近的几个样本点会使得等号成立, 可以说是由它们"支撑"起了这个分类平面, 它们被称为"支持向量". 这就是"支持向量机(Support Vector Machine, SVM)"这个名词的由来.



$$\max \frac{2}{|w|}$$

s.t.

$(w \cdot x + b) \geq 1, \forall x$ of class 1

$(w \cdot x + b) \leq -1, \forall x$ of class 2

结合点到平面的距离的公式

$$\frac{|\mathbf{w}^T\mathbf{x} + b|}{||\mathbf{w}||}$$

可知, 两个异类支持向量到超平面的距离之和为

$$\frac{2}{||\mathbf{w}||}$$

它被称为"间隔".

两类样本的"正中央"实际就是"间隔"最大处. 换言之, 我们要找到这样的超平面

$$\begin{aligned} &\underset{w,b}{argmax} \quad \frac{2}{||\mathbf{w}||} \\ &st. \quad y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, i = 1, ..., m \end{aligned}$$

等价于

$$\begin{aligned} &\underset{w,b}{argmin} \quad \frac{1}{2}||\mathbf{w}||^2 \\ &st. \quad y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1, i - 1, ..., m \end{aligned}$$

这个便是所谓的SVM基本型, 实质上这是一个凸二次规划问题. 我们注意到, SVM基本型并不是一个典型的损失函数和正则化例子, 实际上 SVM 一开始是作为一种统计学习方法提出的. 不过在后面我们会引入软间隔这个概念, 能够到一般意义上的损失函数.

非线性可分样本与核函数

在此之前, 我们在讨论分类问题时有个隐含的前提条件: 样本是线性可分的. 所谓的样本线性可分, 是指存在一个超平面可将其正确分类. 然而, 在实际场景中, 原始样本空间也许就不存在这样的超平面, 如异或逻辑分布的样本就不是线性可分的.

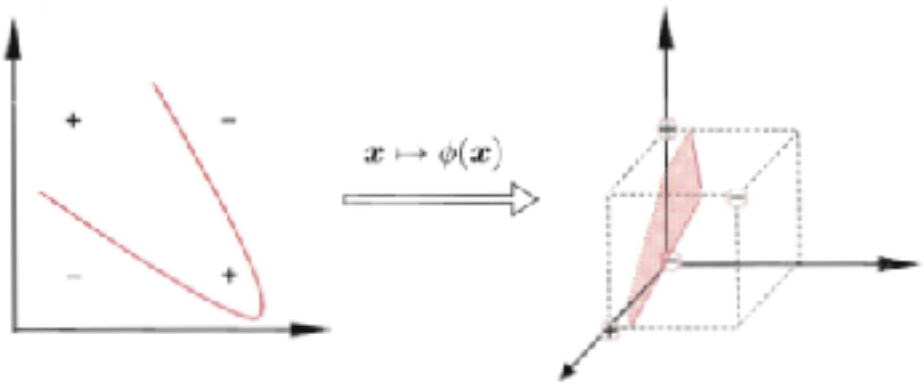


图2-3 异或问题

线性分类器并不能处理非线性可分的数据. 不过, 如果原始样本是有限维, 那一定存在一个高维空间, 在这个空间里线性可分. 令φ(x)为样本点映射到更高维的特征空间的特征向量, 我们改为在特征空间对其进行分类,

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b$$

那么在特征空间对应的SVM模型可表示为

$$\begin{aligned} &\underset{w,b}{argmin} \quad \frac{1}{2}||\mathbf{w}||^2 \\ &st. \quad y^{(i)}(\mathbf{w}^T \phi(\mathbf{x}^{(i)}) + b) \geq 1, i = 1, ..., m \end{aligned}$$

然而, 通过其对偶问题, 求解上式涉及计算样本在映射后空间的内积, 这通常是困难的. 我们设想这么一个函数, 样本点在特征空间的内积等于在原始样本空间中通过该函数进行计算.

$$\kappa(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \langle \phi(\mathbf{x}^{(i)}), \phi(\mathbf{x}^{(j)}) \rangle = \phi(\mathbf{x}^{(i)})^T \phi(\mathbf{x}^{(j)})$$

这样的函数便是所谓的核函数. 通过求解对偶函数, 模型有

$$f(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b = \sum_{i=1}^{m} \alpha_i y^{(i)} \kappa(\mathbf{x}, \mathbf{x}^{(i)}) + b$$

α_i 为对偶问题解出的拉格朗日乘子.

我们把这种方法称之为"核技巧"(kernal trick). 不仅是SVM, 其它线性分类算法也可以引入核函数进行映射, 以处理非线性可分样本.

软间隔

从对SVM的描述可知, 其对噪声是很敏感的. 在极端一点的情况下, 一个噪声点就能吧超平面给推偏. 在我们的讨论中, 我们假定样本在样本空间或者特征空间是线性可分的. 然而, 在现实任务中, 往往很难确定合适的核函数使得样本在特征空间线性可分. 退一步讲, 即使找到了这个核函数, 也很难判断貌似可分的结果是不是由于过拟合造成的. 样本在样本空间的线性不可分, 会不会只是噪声的影响.

缓解这样问题的一个办法是允许SVM在在一些样本上出错, 即引入"软间隔"的概念. 于此对应的"硬间隔", 即是要求所有样本都分类正确.

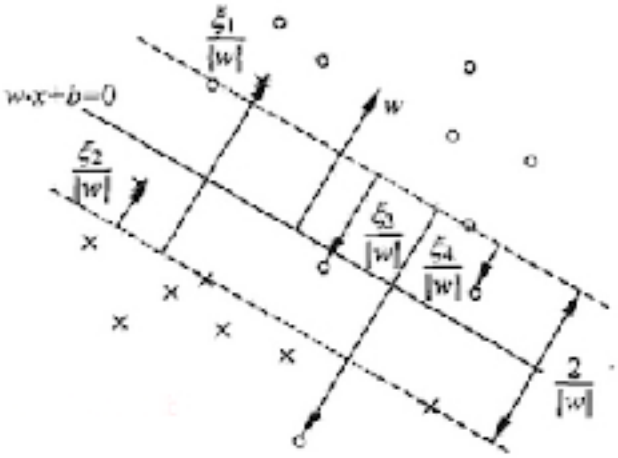


图-? SVM软间隔

$$\begin{aligned} \underset{w,b}{argmin} \quad & \frac{1}{2}||\mathbf{w}|^2 + C\sum_{i=1}^{m}\varepsilon_i \\ st. \quad & y^{(i)}(\mathbf{w}^T\mathbf{x}^{(i)} + b) \geq 1 - \varepsilon_i, i - 1, .., m \end{aligned}$$

当然, 在要求间隔最大化的同时, 分类出错的样本应该尽可能少, 于是我们有损失函数

$$L_{\mathrm{SVM}} - \sum_{i=1}^{n}\max(0, 1 - w^{j'}x_i y_i) + \frac{C}{2}||w||_2^2$$

logistic回归的损失函数作对比, 可以看到两者是很相似的. 区别在于, 一个为log loss, 一个为hinge loss.

实际上, 机器学习的损失函数一般可以写成以下形式

$$\min_f \Omega(f) + C\sum_{i=1}^{n} l(f(\mathbf{x}^{(i)}), y^{(i)})$$

第一项称为"结构风险", 用来描述模型的某些特性, 如复杂程度等; 第二项称为"经验风险", 用来描述模型与训练样本的契合程度; 常数C用于对两者进行折中.

GO DEEP

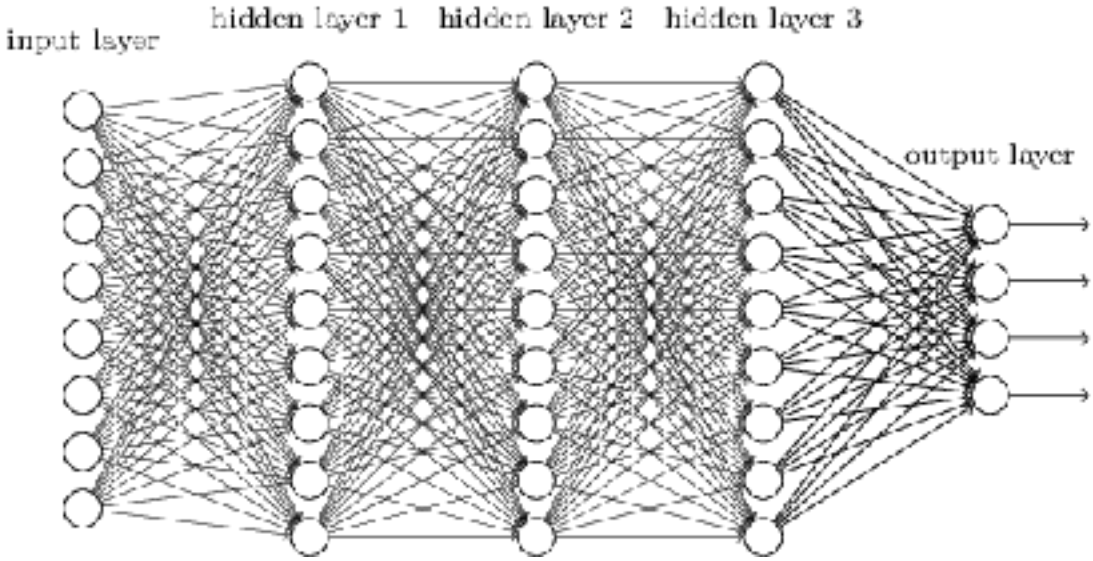
- 核函数需要满足怎么样的性质?
- 核函数怎么选择?

人工神经网络

在经历了几起几落后, 在用户数据的积累和硬件能力的提高的助力下, 人工神经网络今年来又成为显学. AlphaGo的惊人表现, 更是令热度由业内蔓延到普罗大众中. 然而, 神经网络的涵盖的内容非常广和深, 难以在一篇文章中面面俱到. 所以, 下面主要介绍与wx+b关系密切, 在神经网络非常常见与通用的全连接层.

连接与激活

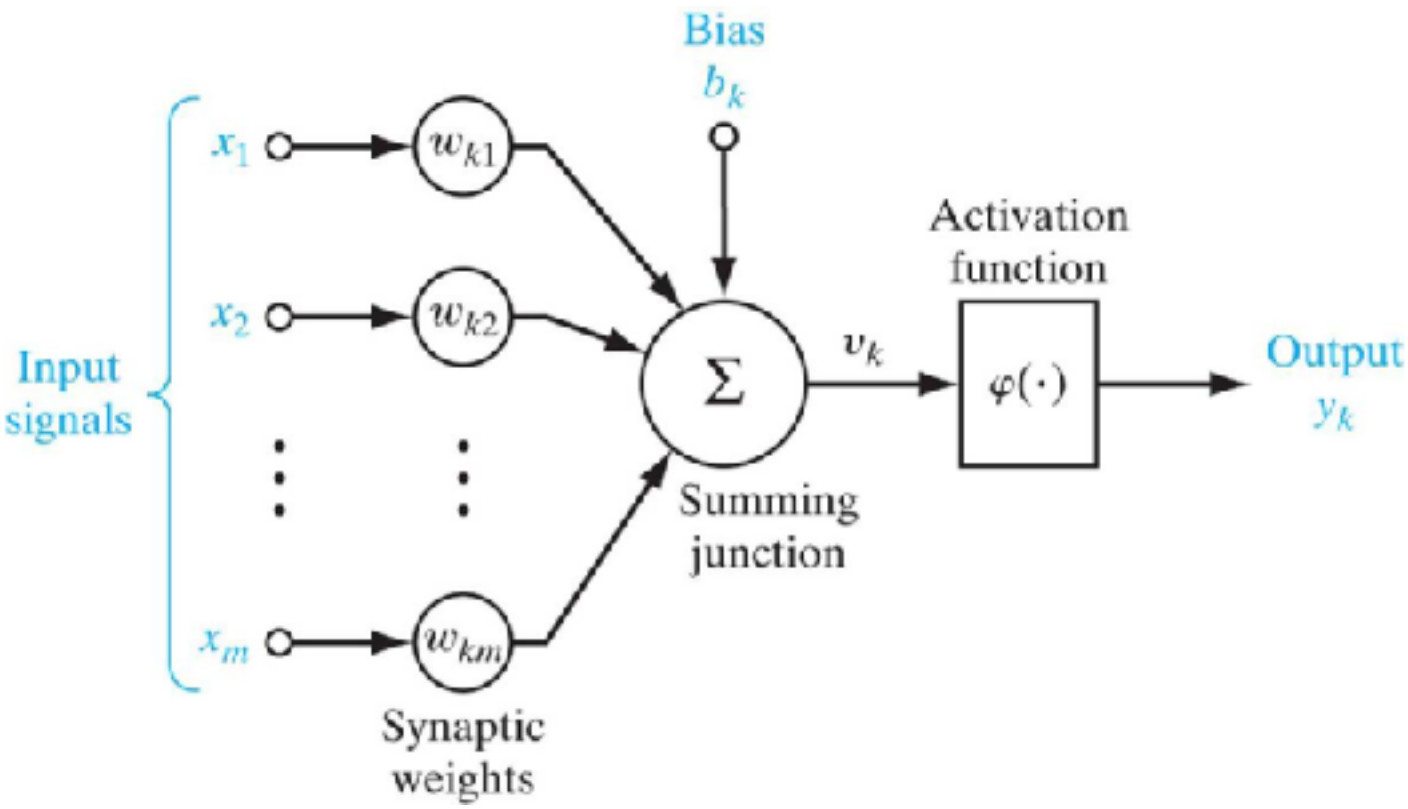
说起神经网络, 大家的第一个念头就是很多个神经元, 像大脑一样, 相互交错连接成一个复杂的网络, 复杂的相互连接孕育出各种神奇的能力.



神经网络

此话不错, 但只对了一半. 人工神经网络是否真的模仿了生物神经网络的工作方式仍是学术界的一大争论点, 但人工神经网络和生物神经网络的确有"连接"这个相似点. 但是, 他们之间还有一个更为重要的相似点, 那就是 **激活**.

一个人工神经元模型如下图所示. 在这里, wx+b的几何意义就不是特别明显了. 人工神经元参考了生物神经元的工作原理, 对输入进行加权求和, 其求和结果与一个阈值做比较, 最后通过一个非线性的激活函数, 输出其状态.



人工神经元

多个人工神经元构成一个神经网络层. 若每个神经元都与上一层的节点有连接, 这样的神经元层被称为全连接层.

神经元 k 可表示为

$$\varphi(\mathbf{w}_k{}^T\mathbf{x} + b_k)$$

那么, 一个由n个神经元构成全连接层的输出为

$$\sigma(\mathbf{x}) = \varphi(\mathbf{W}^T\mathbf{x} + \mathbf{B})$$

可见, 它的功能实际上是对输入进行了到n维空间的的非线性变换.

为什么在前面强调激活是神经网络非常重要的一部分呢. 这是因为, 神经网络强大的学习和表达能力正源于每层的非线性激活函数. 由前面对核函数的讨论可知, 非线性变换可以将样本从非线性可分的样本空间变换到线性可分的特征空间. 然而, 真正可以将样本变换到线性可分的空间的非线性函数是未知的. 模型使用的非线性函数为人为构造, 复杂程度有限, 不一定能一次变换到位. 如果我们将多个全连接层组合起来, 进行多次非线性变换, 那自然可以更接近目标特征空间. 实际上, Hornik等人(Hornik et al., 1989; Cybenko, 1989)已经证明, 多层全连接层理论上可以模拟任何非线性变换.

反之, 若没有非线性的激活函数, 全连接层的功能就退化为线性变换. 一万次线性变换跟一次线性变换没有区别. 对应地, 神经网络退化成与线性回归无异. 再多的层数, 也只是浪费计算资源.

多个全连接层构成的神经网络也叫多隐层前馈网络. 除去输入层, 最后的输出层,
中间的全连接层被称为隐层. 到这里, 我们已经能理解神经网络的工作原理了: 一个或多个隐层将样本非线性变换至特征空间, 输出层的每个节点根据特征空间中的样本, 预测目标的对应属性上预测分类.

导数的链式法则和前向反馈算法.

(待完善)

深度学习

(待完善)

GO DEEP

- 相邻关系:卷积层
- 序列关系: 循环层
- 层数过深: 梯度消失
- (太多了)

Beyond Model

- NFL定理
    - No Free Lunch Theorem.
    - "人工智能, 人工智能, 先人工再智能"
    - 模型和算法是有偏好的.
    - 数据本身性质是一个先验知识. 定性<定序<定量<定比.
    - 模型的超参是未决问题: LR的正则化系数; SVM核函数的选择; 神经网络的层数和单元个数
- 环境搭建: tensorflow作为后端, 模型通过keras + sklearn实现
- Nothing is more practical than a good theory.
  -- Vladimir N. Vapnik.

```
\underset{w,b}{argmin}\quad\frac{1}{2}||\mathbf{w}||^2+C\sum_{i=1}^{m}\varepsilon_i\\st.\qua

\underset{w,b}{argmin}\quad\frac{1}{2}||\mathbf{w}||^2\\st.\quad y^{(i)}(\mathbf{w}^T\phi(\m

\kappa((\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = \left \langle\phi(\mathbf{x}^{(i)}), \phi(\mat

f(\mathbf{x})=\mathbf{w}^{T}\phi(\mathbf{x})+b = \sum_{i=1}^{m}\alpha_i y^{(i)}\kappa(\mathb

\min_f\Omega(f)+C\sum_{i=1}^{m}l (f(\mathbf{x}^{(i)}),  y^{(i)})

\underset{w,b}{argmax}\quad\frac{2}{||\mathbf{w}||}\\st.\quad y^{(i)}(\mathbf{w}^T\mathbf{x}

\left\{\begin{matrix}
\mathbf{w}^T\mathbf{x}^{(i)}+b\geq +\xi ,
&  y^{(i)}=+1
& \\ \mathbf{w}^T\mathbf{x}^{(i)}+b\leq -\xi ,
& y^{(i)}=-1
\end{matrix}\right.
```