感知机

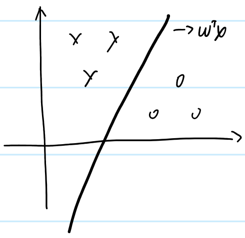
2020年5月11日 星期一 21:11

这一部分主要是希望将深度学习中的历史沿革中的每一部分都做个总结归纳

MCP模型 ⟶ 感知机 ⟶ 多层感知机(BP算法及sigmoid) ⟶ SVM ⟶ 深度学习 { LeNet, AlexNet, Inception, Resnet

1943   1958   1986   20th 90s

感知机：



思想：错误驱动

模型：$f(x)=\text{sign}(w^Tx+b)$ $x\in\mathbb{R}^p$ $w\in\mathbb{R}^p$

$$\text{sign}(a)=\begin{cases}+1 & a\geq 0\\ -1 & a<0\end{cases}$$

先给定 w 初值 w_0，此时的 w_0^Tx 必然存在分类错误的情况，样本集 $\{(x_i,y_i)\}_{i=1}^N$

将分类错误的样本称为 D，用 D 去不断修正 w

策略：loss function：

$$L(w)=\sum_{i=1}^N I\{y_i(w^Tx_i+b)<0\}$$

分类不对的情况

括号内为真时输出1，为否时输出0

其实就是统计分类不正确的数量

因为 $w^Tx+b>0$ 时 $y_i=1$，$w^Tx+b<0$ 时 $y_i=-1$ ⟹ $y_i(w^Tx_i+b)>0$ 此时 x_i 被正确分类

但是这种 $L(w)$ 是不可导的，无法直接解决

所以直接以 $y_iw^Tx_i$ 为 loss

Loss：$L(w)=\sum_{x_i\in D}-y_i(w^Tx_i+b)$

梯度：$\nabla_w L=\sum_{x_i\in D}-y_ix_i$ $\nabla_b L=\sum_{x_i\in D}-y_i$

解法：随机梯度下降 SGD，遍历所有样本，如果当前样本出现错误，则按照下面更新权重

更新：$w^{(t+1)}\leftarrow w^{(t)}-\lambda\nabla_w L = w^{(t)}+\lambda y_ix_i$ λ为学习率

$$b^{t+1}\leftarrow b^t-\lambda\nabla_b L=b^t+\lambda y_i$$

观察上式可发现，每一次检测到错误，更新权重时，都需要算一遍 $y_i x_i$，这可能会拖慢速度

因此有一种对偶形式可以简化运算

先假设 w_0 和 b_0 一开始都设为0，则优化到最后 w，b 分别为

$$w=\sum_{i=1}^{N} n_i \lambda y_i x_i$$

$$b=\sum_{i=1}^{N} n_i \lambda y_i$$

这里的 n_i 表示在SGD的过程中每个数据被错误分类的次数

这个 n_i 从含义上来讲，越大表明这个数据被分错的次数越多，表明它离超平面其实非常近（就像svm中的支持向量）

这样一来其实只要训练 n_i 即可。

只要 $y_i(w^T x_i+b) = y_i(\sum_{j=1}^{N} n_j \lambda y_j x_j \cdot x_i + \sum_{j=1}^{N} n_j \lambda y_j)$ 出现小于1的情况，则当前这个数据的 n_i 就加1

这样一来最后就训练了一个 $n_1 n_2 \cdots n_N$ 的一个向量，而每个 $y_i x_i$ 只要提前计算好，到算的时候拿来用就行（即GRAM矩阵）