Math in Information Age

Course Review

Mingdong Wu

EECS, PKU

Outline

感知机算法和 SVM

线性分类器

给定 d 维空间中一组 sample 集合 $S = (x_1, y_1), (x_2, y_2)...(x_n, y_n)$, 考虑一个线性分类器：$y = (x, w)$, 也就是 x 所有分量的加权和。同时考虑一个阈值 t，使得：

- $(w, x_i) > t$ 如果 x_i 是正例, 即 $y_i = +1$
- $(w, x_i) < t$ 如果 x_i 是负例，即 $y_i = -1$

而这样的一组 (w, t) 就叫线性分类器

感知机算法

感知机算法：

- $w \leftarrow 0$
- 依次考虑 sample(x_i, y_i)
- 如果 $y_i(x_i, w) \leq 0$,$w \leftarrow w + x_i * y_i$

显然，我们有两个结论：

- 如果算法停止，那学出来的 w 就是 sample 中 x_i 的线性组合
- 如果 sample 中 x_i 线性不可分，则算法无法停止

感知机算法收敛性

如果存在一个 w^* 使得 $(w^*, x_i) * l_i \geq 1$, 那么感知机算法至多在 $r^2|w^*|^2$ 步内停止，其中 $r = max_i|x_i|$

- 一次迭代后，$(w^*, w + x_i * l_i) \geq (w^*, w) + 1$
- 一次迭代，$(w + x_i * l_i, w + x_i * l_i) \leq |w|^2 + r^2$

那么设更新的步数是 m，则有:

- $(w, w^*) \geq m \rightarrow |w||w^*| \geq m$
- $|w|^2 \leq mr^2 \rightarrow |w| \leq r\sqrt{m}$

得到 $m^2|w^*|^2$

线性可分和 margin

我们知道感知机算法能停止需要训练集本身线性可分，那么如何判断训练集的点线性可分呢？

- 直接看出一个特定的分类平面
- 对正例和负例样本分别求凸包，判断凸包是否相交

对于给定的线性分类器 (w, t)，sample 中离分类平面最近的距离就是 margin

SVM

SVM(supported vector machine), 就是在训练集已经线性可分时，直接优化 margin

$$
\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2
$$

$$
\text{subject to: } y_i(\mathbf{w}\cdot\mathbf{x}_i + b) \geq 1,\ \forall i \in [1, m]\,.
$$

求 SVM 和解感知机都能找到可以分开所有 sample 的线性分类器，但 SVM 优化了 margin，有更好的泛化性

Kernel Function

Kernel Function

对于线性不可分的 sample，我们怎么用感知机算法或 SVM 找一个决策平面呢？考虑将 sample 映射到另一个空间，使它们在另一个空间中可分，再跑感知机或 SVM 算法

- e.g.$(x, y) \to (x, y, x^2 + y^2)$

设 ϕ 将 $x_1, x_2 ... x_n$ 映射到 $\phi(x_1), \phi(x_2)..., \phi(x_n)$ 如果要跑感知机算法，首先得判断点有没有分对：

- $(w, \phi(x_j)) = \sum_i^n c_i(\phi(x_i), \phi(x_j))$

还得算更新之后的 w：

- 如果 samplex_i 要更新 w, 那就在相应的 c_i 处 +1

综上，我们发现不用具体计算每个 $\phi(x_i)$, 只用保留 $k(x_i, x_j) = (\phi(x_i), \phi(x_j))$ 即可，而这样的 $k(x_i, x_j)$ 就是 kernel function

Kernel Function 性质

定理：k 是 kernel function **当且仅当矩阵** $k_{ij} = k(x_i, x_j)$ **是半正定的。假定** k_1, k_2 **是** kernel，c **是常数**，$f(x)$ **是从** d **维空间到** R **的映射**，**那么**

- $k_3 = ck_1$
- $k_3 = k_1 + k_2$
- $k_3 = f(x)f(y)k1(x, y)$
- $k_3 = k_1 * k_2$

都是合法的 kernel

kernel function 合法性判断实例

考虑 $k(x,y)=e^{-c|x-y|^2}$?

- $k(x,y)=e^{-c|x-y|^2}=f(x)f(y)e^{2c(x,y)}$
- $e^{2c(x,y)}$ 再用 Taylor 展开处理

注意，我们假定了“前述性质可以用于无穷维 kernel”的前提

PAC 学习理论

同分布假设

在 PAC learning 这个 section 中，我们之后考虑的都是 (二) 分类问题

- 给定样例集 $S = (x_1, y_1), ...(x_n, y_n)$, 我们认为所有的样例 x_i 都相互独立地服从一个隐含的未知分布 $x_i \sim \mathcal{D}$
- 这几乎是之后分析最重要的基础

下面解释几个通用概念

- concept 可以看做是在 $\mathcal{D}$ 支撑集上的指示函数，也就是把数据分布支撑集的某些部分判断成正类，另外的判定为负类
- hypothesis 就是我们用自己的某些学习算法得到的一个在 $\mathcal{D}$ 支撑集上的指示函数，跟 concept 类似，只不过它是“不完美的” concept
- hypothesis space(class) 我们的学习算法可能输出的所有 hypothesis

泛化误差和经验误差

我们知道对于一个 hypothesis$h$, 它很可能和目标概念 c 不同，那么如何刻画这种差异呢？我们有泛化误差和经验误差两种角度。

- 泛化误差 (True error) $err_{\mathcal{D}} = Prob(h \triangle c)$
- 经验误差 (Training error)$err_S = Prob(x \in S, x \in (h \triangle c)) = |S \bigcap (h \triangle c)|/|S|$

由于我们通常不知道 concept$c$ 的隐藏分布，比如 “猫图分布”，因此泛化误差是无法直接计算的，我们只能计算机器学习算法输出假设 h 的经验误差。

Overfitting

我们通常采取尽可能降低经验误差，也就是训练集上误差的策略，但在极低的经验误差往往带来较高的泛化误差，这就是 Overfitting

如何学出和 concept 尽可能接近的 hypothesis？

如果 conceptc 就在假设空间 $\mathcal{H}$ 内，那我们就一个个剔除和 conceptc 不一致的假设就好，但仅凭在训练集上的观测，我们会有很多假设是无法区分的，也就是在训练集上表现一致，但泛化能力可能不一致。

这种情形下，我们可以通过增加 training set 大小 |S|，以 $1-\delta$ 概率学到目标 concept 的 ϵ 近似

hypothesis space 有限的情形

那么 training set 大小 $|S|$ 究竟要达到什么程度（阶）才能以 $1-\delta$ 概率学到目标 concept 的 ϵ 近似呢？

- 当 $n \geq \frac{1}{\epsilon}(ln(|\mathcal{H}|)+ln(1/\delta))$ 时
- 存在经验误差 =0, 而泛化误差 $\geq \epsilon$ 的假设的概率小于 δ

因此按照前述“剔除”算法，不断剔除在训练集上和目标 concept 不一致的假设，就有 $1-\delta$ 的概率学到目标 concept 的 ϵ 近似

但注意上一个定理的前提是，假设空间 $\mathcal{H}$ 中存在和目标 conceptc 在训练集 S 上一致的假设 h^*。而一般地，机器学习算法假设空间没那么“强”，可能最好的假设在训练集上的表现也做不到经验误差为 0(炼丹)，这是我们还能通过增大训练集来以 $1-\delta$ 概率学到目标 concept 的 ϵ 近似吗？

- 当 $n \geq \frac{1}{2\epsilon^2}(ln(|\mathcal{H}|) + ln(2/\delta))$ 时
- 每个假设空间 $\mathcal{H}$ 中的额假设 h 的经验误差和泛化误差之差的绝对值不超过 ϵ
- $Prob(|err_D(h) - err_S(h)| \leq \epsilon) \geq 1-\delta$

VC 维

以上我们考虑的都是假设空间有限的情形，否则无法用 union bound。但对于机器学习算法，其输出的假设空间往往是无穷维的，这时我们还能有之前那样的“好 bound”吗？有，下面我们先引入刻画假设空间 $\mathcal{H}$ 复杂度的新参数，VC 维，利用这个参数我们也能得到很好的 bound。而作为 VC 维的铺垫，我们先引入 growth function 的概念。

- 对正整数 m, 定义假设空间 $\mathcal{H}$ 的增长函数 $\prod_{\mathcal{H}} = max_{x_1,x_2,...x_m \subset \mathcal{X}}|(h(x_1),h(x_2),...h(x_m)|h \in \mathcal{H})|$

$\mathcal{H}$ 的增长函数刻画了 $\mathcal{H}$ 的表达能力

引入 Growth Function 后的 bound

对假设空间 $\mathcal{H}$, 正整数 m，$0 < \epsilon < 1$ 和任意 $h \in \mathcal{H}$:

- $P(|E(h) - \hat{E}(h)| > \epsilon) \leq \prod_{\mathcal{H}}(2m)exp(-\frac{m\epsilon^2}{8})$

但仔细看，在未知 $\prod_{\mathcal{H}}$ 的性质之前，这个 bound 没有意义

VC 维

下面简单估计下 growth function，希望能得到它的上界从而使前述的 bound 更有意义，我们会发现当 m 比较小时 $\prod_{\mathcal{H}} = 2^m$, 而从某个 threshold 开始 $\prod_{\mathcal{H}(m)}$ 就稳定在了某个多项式的阶上，比如 $O(m^4)$ 而从指数退化到多项式的这个 threshold，就是 VC 维

- $VC(\mathcal{H}) = max m : \prod_{\mathcal{H}}(m) = 2^m$

VC 维例子

$VC(\mathcal{H}) = d$ 表明存在，注意，是存在，大小为 d 的 sample set 能被假设空间，注意，是假设空间，打散。算 VC 维，就只用构造出最大的能被打散的实例集，大小为 d，再证明大小为 d+1 的任意实例集都无法被打散。

- 板书
- 会几个简单的例子掌握 idea 就好，证明都是 tricky 的

VC 维和 Growth Function

注意我们的问题是要使之前的 bound 有意义，也就是希望 bound 住 $\prod_{\mathcal{H}(m)}$ 的阶，引入了 VC 维后，我们有如下定理：

- 若假设空间 $\mathcal{H}$VC 维是 d，对于任意的 m 有：
- $\prod_{\mathcal{H}(m)} \leq \sum_{i=0}^{d} \binom{m}{i}$

可见当 m 很大时，$\prod_{\mathcal{H}(m)}$ 关于 m 是 d 次多项式的，回过头 check 之前的 bound： $P(|E(h) - \hat{E}(h)| > \epsilon) \leq \prod_{\mathcal{H}}(2m)exp(-\frac{m\epsilon^2}{8})$

发现 $\prod_{\mathcal{H}}(2m)exp(-\frac{m\epsilon^2}{8})$ 终于无穷小了

$\prod_{\mathcal{H}(m)}$ 关于 VC 维的 bound 可以稍稍变形一下:

$\prod_{\mathcal{H}(m)} \leq (\frac{e*m}{d})^d$

那么令:

- $P(|E(h) - \hat{E}(h)| > \epsilon) \leq \prod_{\mathcal{H}}(2m)exp(-\frac{m\epsilon^2}{8}) < \delta$

我们就得到了精确的 m-bound, 也就是以 $1-\delta$ 概率学到目标 concept 的 ϵ 近似至少需要的 training sample

VC 维和 growth function 练习

证明:

- 往假设空间 $\mathcal{H}$ 中添加一个假设 h 至多使 VC 维加一
- 构造上述命题的紧实例？
- $VC(A \bigcup B) \leq VC(A) + VC(B) + 1$