Lecture 14 Regularization

整理者：LobbyBoy* 2020年2月27日

1. Regularized Hypothesis Set

下图中右侧的图像表示我们的学习过程发生了过拟合，原因是：在有限的训练样本上运用了过于复杂的假说集合$\mathcal{H}_{10}$。如果我们想要提高学习效果，就要使用简单一点的假说集合，如$\mathcal{H}_2$，这样就可以得到左边图像中的红线了。也就是说，我们需要step back from $\mathcal{H}_{10}$ to $\mathcal{H}_2$(这里需要回忆我们此前所提到的nested hypothesis set，10次多项式的假说集合包含2次多项式的假说集合)。

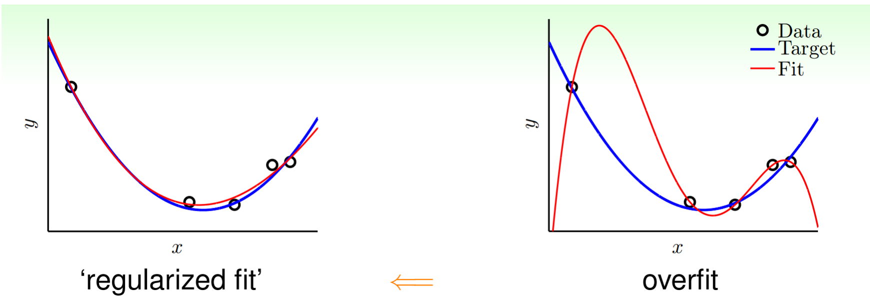

图 1: Step back

如何step back？考虑：仍然以$\mathcal{H}_{10}$为假说集合，但是加上约束条件：

$$w_3 = w_4 = \cdots = w_{10} = 0$$

这样就等价于以$\mathcal{H}_2$为假说集合，因为：

$$\mathcal{H}_2 = \left\{ \mathbf{w} \in R^{10+1} \ while \ w_3 = w_4 = \cdots = w_{10} = 0 \right\}$$

在该假说集合上进行线性回归：

*本笔记根据台湾大学林轩田教授于线上教育平台Coursera开设的“机器学习基石”课程整理而成(课程内容见：https://www.coursera.org/learn/ntumlone-mathematicalfoundations/home/welcome)。笔记内的大多数图片来自于林老师的课程slides。感谢林老师能够将如此精彩的课程通过线上平台同所有人分享，thanks!

$$\min_{\mathbf{w}\in R^{10+1}} \quad E_{in}(\mathbf{w})$$
$$s.t. \quad w_3 = w_4 = \cdots = w_{10} = 0$$

现在，我们考虑稍微放松我们的约束条件$w_3 = w_4 = \cdots = w_{10} = 0$。原本的约束条件表示，后8个高次项的系数必须为0；现在我们放松成：有至少8个系数为0，但并不指定是那些次方项的系数，即：

$$\mathcal{H}_2^{'} = \left\{\mathbf{w} \in R^{10+1} \ while \ \geq 8 \ of \ w_q = 0\right\}$$

在该假说集合上进行线性回归：

$$\min_{\mathbf{w}\in R^{10+1}} \quad E_{in}(\mathbf{w})$$
$$s.t. \quad \sum_{q=0}^{10} I(w_q \neq 0) \leq 3$$

易知，$\mathcal{H}_2 \subset \mathcal{H}_2^{'} \subset \mathcal{H}_{10}$。但换成$\mathcal{H}_2^{'}$，线性回归变成了一个NP-Hard的问题，因为引入了boolean运算。因为我们再考虑稍微变换约束条件：刚刚已经变成了，至少8个系数为0；现在我们变换成，这11个系数的平方和不大于某个设定值。这就好像是，原来是非常“硬”的约束条件，一定要某些系数为0，另一些不为0；现在我们不要求非黑即白，而是说大家都可以不为0，但是都必须比较小才行，即：

$$\mathcal{H}(C) = \left\{\mathbf{w} \in R^{10+1} \ while \ ||\mathbf{w}||^2 \leq C\right\}$$

在该假说集合上进行线性回归：

$$\min_{\mathbf{w}\in R^{10+1}} \quad E_{in}(\mathbf{w})$$
$$s.t. \quad \sum_{q=0}^{10} w_q^2 \leq C$$

易知，$\mathcal{H}(0) \subset \mathcal{H}(1.126) \subset \cdots \mathcal{H}(1126) \subset \cdots \mathcal{H}(\infty)$。我们将最后变成的这个假说集合$\mathcal{H}(C)$称为regularized hypothesis set，将解上述最优化问题得到的最优解称为regularized hypothesis $\mathbf{w}_{REG}$。

2. Weight Decay Regularization

我们将上节中最后得到的regularized regression problem：

$$\min_{\mathbf{w}\in R^{Q+1}} \quad E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{z}_n - y_n)^2$$
$$s.t. \quad \sum_{q=0}^{Q} w_q^2 \le C$$

写成矩阵形式：

$$\min_{\mathbf{w}\in R^{Q+1}} \quad E_{in}(\mathbf{w}) = \frac{1}{N}||\mathbf{Zw} - \mathbf{y}||^2$$
$$s.t. \quad \mathbf{w}^T\mathbf{w} \le C$$

下图中，紫色的$\mathbf{w}$表示其正处于梯度下降法中的某一点；黑色的$\mathbf{w}_{lin}$表示最优解，即“谷底”；蓝色的$-\nabla E_{in}$表示紫色位置处的负梯度方向，也就是下一次$\mathbf{w}$应该前进的方向；红色的圆圈表示我们增加的约束条件$\mathbf{w}^T\mathbf{w} \le C$。

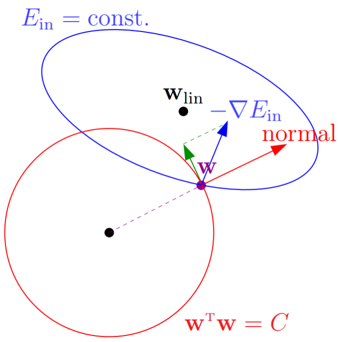

图 2: Step back

也就是说，我们在梯度下降中对$\mathbf{w}$做更新，不可以跑出红色的圆圈边界。比如这个时候，梯度下降法希望$\mathbf{w}$向蓝色方向迈一步，但是约束条件使得$\mathbf{w}$不能出圈，因此只能沿着圆在该点的切线方向前进，即图中的绿色箭头。因为可以想到，当$\mathbf{w}$的更新收敛时，其梯度必然沿着圆的径向方向，没有切线分量，即：

$$-\nabla E_{in}(\mathbf{w}_{REG}) \propto \mathbf{w}_{REG}$$

引入所谓的Lagrange multiplier $\lambda > 0$，将上式写作(想一下，为何可以令$\lambda > 0$)：

$$\nabla E_{in}(\mathbf{w}_{REG}) + \frac{2\lambda}{N}\mathbf{w}_{REG} = \mathbf{0}$$

假设我们已经知道了λ的值，那么解$\mathbf{w}_{REG}$就很简单了：

$$\mathbf{w}_{REG} = (\mathbf{Z}^T\mathbf{Z} + \lambda\mathbf{I})^{-1}\mathbf{Z}^T\mathbf{y}$$

注意到右侧的逆矩阵一定存在(正定矩阵)。这样的回归我们称其为“岭回归”(ridge regression)，是统计学上的名词。

我们回头再看一下：

$$\nabla E_{in}(\mathbf{w}_{REG}) + \frac{2\lambda}{N}\mathbf{w}_{REG} = \mathbf{0}$$

上面的等式很像是某个函数的导数置零。我们可以积分一下，得到：

$$E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$$

也就是说，最小化上面的函数，就等价于解$\nabla E_{in}(\mathbf{w}_{REG})+\frac{2\lambda}{N}\mathbf{w}_{REG} = \mathbf{0}$；而解$\nabla E_{in}(\mathbf{w}_{REG})+\frac{2\lambda}{N}\mathbf{w}_{REG} = \mathbf{0}$，就等价于解(对于某个C)：

$$\min_{\mathbf{w}\in R^{Q+1}} \quad E_{in}(\mathbf{w}) = \frac{1}{N}||\mathbf{Z}\mathbf{w} - \mathbf{y}||^2$$
$$s.t. \quad \mathbf{w}^T\mathbf{w} \le C$$

其中，我们如果给定C，那么λ也就确定了。因此，我们就直接给定λ好了，这样直接可以解没有约束的最优化问题，该最优化问题对应回原来有约束的最优化问题中的某个C，但是我们并不care。

最小化的目标函数$E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$，我们称之为augmented error，记作$E_{aug}(\mathbf{w})$。其中的$\mathbf{w}^T\mathbf{w}$项我们称之为regularizer。综上：

$$\mathbf{w}_{REG} = \underset{\mathbf{w}}{argmin}\ \ E_{aug}(\mathbf{w})$$

注意$\lambda >= 0$，等于0时等价于没有任何正则化。

调整λ的大小看看学习结果如何：

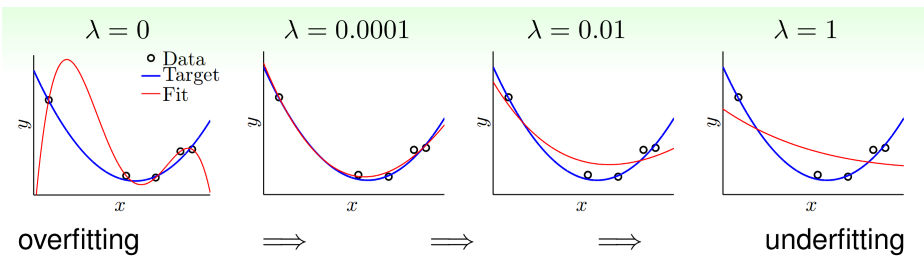

图 3: Step back

可见，一点点的λ就可以发挥很大的作用，使得模型避免了overfitting。我们将加上$\frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$的这种正则化方式称为weight-decay regularization。λ越大，表示我们越希望各个weight都比较小，即short的$\mathbf{w}$，也就等价于小的C。

3. Regularization and VC Theory

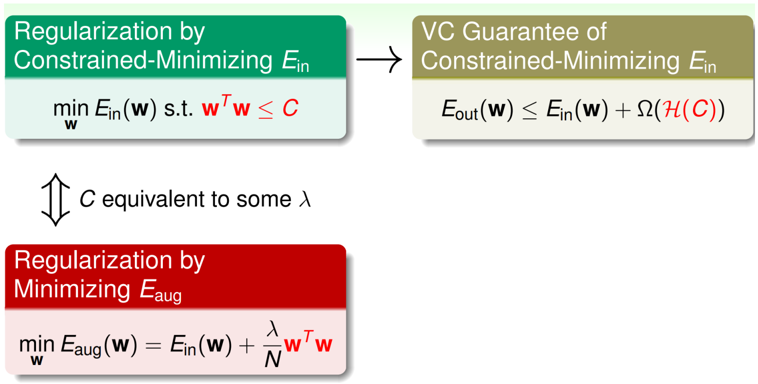

图 4: Regularization and VC

第一种观点：正则化中最小化augmented error，某种程度上相当于在最小化泛化误差(上界)。由VC bound我们知道：

$$E_{out}(\mathbf{w}) \le E_{in}(\mathbf{w}) + \Omega(\mathcal{H})$$

观察augmented error的表达式：

$$E_{aug}(\mathbf{w}) = E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$$

可以发现，$E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$与$E_{in}(\mathbf{w}) + \Omega(\mathcal{H})$很像，只是$\frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$与$\Omega(\mathcal{H})$不同。$\Omega(\mathcal{H})$是表示整个假说集合的复杂度，而$\frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$可以看作是一个hypothesis的复杂度，即$\Omega(\mathbf{w})$。一个hypothesis的复杂度某种程度上可以反映其所属假说集合的复杂度，也就是说假说集合的复杂度某种程度上可以被其中的某个hypothesis所代。这样的话，$\frac{\lambda}{N}\Omega(\mathbf{w}) \approx \Omega(\mathcal{H})$，那么最小化$E_{aug}$就相当于直接最小化$E_{out}$的上界，效果会比只最小化$E_{in}$更好。也可以说，$E_{aug}$较$E_{in}$来说是更好的针对$E_{out}$的“代理”(proxy)。

另一种观点是：正则化算法使得我们真正的做选择的假说集合并没有实际上那么大。例如，我们看似选择了10次多项式为假说集合，但是由于正则化项的存在，我们实际上把那些weight很大的hypothesis排除在选择之外，因此“有效假说集合”会变小，我们记之VC dimension为$d_{VC}(\mathcal{H}(C))$。然而，由于我们仍然拿来的是10次多项式假说集合，只是演算法让我们忽略了该假说集合中很多hypotheses。因此我们定义一种“有效的vc dimension”：该effective vc dimension不仅考虑假说集合的复杂度，还考虑演算法如何在这个假说集合中做选择——记作$d_{EFF}(\mathcal{H}, \mathcal{A})$。因此，对于正则化，$d_{VC}(\mathcal{H})$依然很大，但是$d_{EFF}(\mathcal{H}, \mathcal{A})$比较小($\mathcal{A}$为正则化算法)。

4. General Regularizers

Weight-decay regularization以$\frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$为正则项，使得最后训练出的$\mathbf{w}$不会太“长”($\lambda > 0$)。

考虑更一般的正则项$\Omega(\vec{w})$。正则项的作用是，对$\mathcal{H}$进行隐性的约束，使训练出的hypothesis往target function的方向靠近——constraint in the 'direction' of target function，因此我们选择正则项的第一个原则就是target-dependent——根据对目标函数的一些了解设计正则项。例如，我们知道目标函数很可能是一个偶函数，那么我们就希望奇数次方项的系数比较小，那么我们可以设计如下的正则项：

$$\sum I(q\ is\ odd)w_q^2$$

其次，正则项的设计也可以基于plausible。比如，我们往往认为smoother或simpler(因为stochastic/deterministic noise都是non-smooth，结果往往overfit)的训练结果更好，因此我们可以设计L1 regularizer：

$$\sum |w_q|$$

最后，我们也可以设计friendly的正则项，即容易进行最优化。例如，L2 regularizer：

$$\sum w_q^2$$

L1正则化的解一般是稀疏的，如下图所示，一般落在正方形的顶点处：

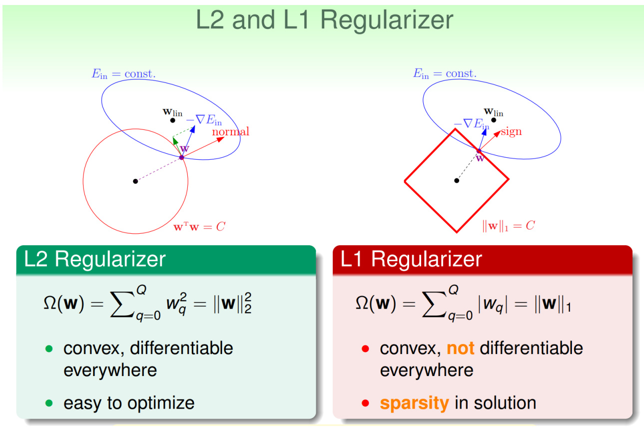

图 5: L1 regularizer - sparse solution

一般来说，more noise就需要more regularization。但noise水平一般是不知道的，那么就需要选择proper的regularization。下一章将介绍validation的方法，帮助我们选择一个合适的λ。