発表予定日：2019 年 10 月 28 日
慶應義塾大学理工学部物理学科
岡崎健人

卒研ゼミ「深層学習」

5　機械学習の基礎

5.2　容量，過剰適合，過少適合

5.2.1　ノーフリーランチ定理（要約）

機械学習アルゴリズムは、有限の訓練用のデータセット（training dataset）からそこに潜む一般的なパターンを見つけ出すが、このことは論理的に妥当[*1]でないように思われる。機械学習では確率的なパターンのみを吐き出すことでこの問題を部分的に回避している。つまり、たとえば「カラス 1、カラス 2、…、カラス n は黒い」というデータセットを与えれば、「ほとんどすべてのカラスは黒い」というほぼ正しいルールを得ることが約束される。

これでもまだ問題は残っている。ノーフリーランチ定理（no free lunch theorem, NFLT）は次のように表現される。「すべての評価関数に適用できる（他の方法よりも）効率のよいアルゴリズムは存在しない [1]」「コスト関数の極値を探索するあらゆるアルゴリズムは、全ての可能なコスト関数に適用した結果を平均すると同じ性能となる [2]」「与えられた課題に独立などんな学習アルゴリズムの優劣判定方法はない。どのようなモデルが最も優れているのかは、問題の種類や、付随する情報によって決まる [3]」など。つまり、**すべての評価関数に対して平均したとき**に、普遍的で絶対的に最良なアルゴリズムなど存在せず、個々の問題の特徴を把握して、それに最適な機械学習アルゴリズムを理解・選択すべきであるということだ[*2]。ノーフリーランチ定理の証明は参考文献 [1] を参照されたい。

5.2.2　正則化

結局のところノーフリーランチ定理は、特定のタスクに対してうまく機能するアルゴリズムを設計しなければならないということをほのめかしているが、これは学習アルゴリズムに特定の設計を組み込んで修正すれば解決する。

最小二乗法での関数の推定においては、データの個数よりも近似する曲線の次数のほうが大きいときや、係数のベクトル $\boldsymbol{w} = [w_0, w_1, \dots, w_n]^\mathsf{T}$ の成分の値が大きくなると、過学習に陥りやすくなる。このことを防ぐために平均二乗誤差 $\mathrm{MSE}(\boldsymbol{w})$ に重み減衰 $\lambda \boldsymbol{w}^\mathsf{T}\boldsymbol{w}$ を加えたもの $J(\boldsymbol{w}) = \mathrm{MSE}(\boldsymbol{w}) + \lambda \boldsymbol{w}^\mathsf{T}\boldsymbol{w}$ を最小化することを考える。データの数を m，データを (x_i, y_i)，目的関数を n 次多項式 $\hat{y} = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n =$

[*1] 論理学の文脈で valid は妥当であると訳されるのが普通（翻訳ミス？）。妥当であるとは、たとえば「A である。A ならば B である。ゆえに B である」という演繹的推論は、前提が真であれば結論は必ず真であり、このことを妥当であるという。一方で「a_1 は P である。a_2 は P である。ゆえに（おそらく）すべての a は P である」という帰納的推論において、前提が真であっても結論が真であるとは保障されず、妥当ではない。

[*2] 定理の名称は英語の格言 “There ain’t no such thing as a free lunch.”（TANSTAAFL と略される）から。かつてアメリカのサルーンでは、「飲みにきた客にはランチを無料でふるまう」という宣伝文句が用いられていた。うまい話のように思えるが、ランチ代は酒代に上乗せされているだけだった [4]。タダ飯のような素敵なものなんて存在せず、結局代金は支払わなければならない。同じように「すべての目的関数に使える万能で性能の良いアルゴリズム」のような素敵なものは存在せず、データに対して前提条件を設けるという代金を支払わなければならない。

$\sum_{k=0}^{n} w_k x^k$ とし、この係数 $\boldsymbol{w}$ を求める。行ベクトル $\boldsymbol{x}_i{}^\mathsf{T} = \left[1, x_i, {x_i}^2, \ldots, {x_i}^n\right]$ を縦に並べた行列

$$X = \begin{bmatrix} \boldsymbol{x}_1{}^\mathsf{T} \\ \boldsymbol{x}_1{}^\mathsf{T} \\ \vdots \\ \boldsymbol{x}_m{}^\mathsf{T} \end{bmatrix} = \begin{bmatrix} 1 & x_1 & {x_1}^2 & \ldots & {x_1}^n \\ 1 & x_2 & {x_2}^2 & \ldots & {x_2}^n \\ & & \vdots & & \\ 1 & x_m & {x_m}^2 & \ldots & {x_m}^n \end{bmatrix}$$

と（これは計画行列（design matrix）というのだった。日本語版教科書 [5] の p. 77.)、ベクトル $\boldsymbol{y} = [y_1, \ldots, y_m]^\mathsf{T}$ を用いれば、$\partial(\boldsymbol{w}^\mathsf{T}\boldsymbol{w})/\partial\boldsymbol{w} = 2\boldsymbol{w}$ より

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial \mathrm{MSE}(\boldsymbol{w})}{\partial \boldsymbol{w}} + \frac{\partial\left(\lambda \boldsymbol{w}^\mathsf{T}\boldsymbol{w}\right)}{\partial \boldsymbol{w}} = \frac{2}{m}\left(X^\mathsf{T}X\boldsymbol{w} - X^\mathsf{T}\boldsymbol{y}\right) + 2\lambda\boldsymbol{w}.$$

これを $\boldsymbol{0}$ とすれば、$\boldsymbol{w}$ を変数とみた $(n+1)$ 本の連立方程式 $\left(X^\mathsf{T}X + m\lambda I\right)\boldsymbol{w} = X^\mathsf{T}\boldsymbol{y}$ が得られる。λ の値を大きくしていくと、連立方程式におけるすべての対角成分 w_i の係数が大きくなってゆく。しかし右辺の $X^\mathsf{T}\boldsymbol{y}$ はデータから得られる定ベクトルであるため、w_i の値は小さくならなければならない。ゆえに目的関数は多項式の丸みを生かせなくなり、直線に近い形となる。逆に λ の値が小さいときは w_i の値が大きくなるため目的関数はよく曲がるような、柔軟な曲線となる。このことはデータ点をすべて曲線で無理やり繋いでしまうような、過学習の状態になることを示している。中程度の値の λ を用いれば、過少適合も過剰適合も回避できるようになる。

いま正則化項を $\lambda\|\boldsymbol{w}\|_2^2$ としていたが、この線形回帰をリッジ回帰（Ridge regression）という。ほかにも正則化項を $\lambda\|\boldsymbol{w}\|_1^1$ としたものや、$\|\boldsymbol{w}\|_2^2$ と $\|\boldsymbol{w}\|_1^1$ の線型結合としたものもあり、それぞれラッソ回帰（LASSO regression)、Elastic Net という [6]。正則化は最適化と並んで重要であるため、詳しくはまとめて第 7 章で勉強する。

なぜ $J(\boldsymbol{w}) = \mathrm{MSE}(\boldsymbol{w}) + \lambda\boldsymbol{w}^\mathsf{T}\boldsymbol{w}$ か？

目的関数を多項式とするときの係数のベクトル $\boldsymbol{w} = [w_0, w_1, \ldots, w_n]^\mathsf{T}$ の成分が大きいと、目的関数の曲がり方が激しくなって、過学習に陥ってしまう。そこで t をある定数として、$\|\boldsymbol{w}\|_2^2 = \boldsymbol{w}^\mathsf{T}\boldsymbol{w} \leq t$ という制約のもとでの平均二乗誤差 $\mathrm{MSE}(\boldsymbol{w}) = \frac{1}{m}\|X\boldsymbol{w} - \boldsymbol{y}\|_2^2 = \frac{1}{m}(X\boldsymbol{w} - \boldsymbol{y})^\mathsf{T}(X\boldsymbol{w} - \boldsymbol{y})$ の最小化を考える。これは KKT 法を用いれば解ける。一般化ラグランジュ関数を

$$L(\boldsymbol{w}, \lambda) = \frac{1}{m}(X\boldsymbol{w} - \boldsymbol{y})^\mathsf{T}(X\boldsymbol{w} - \boldsymbol{y}) + \lambda\left(\boldsymbol{w}^\mathsf{T}\boldsymbol{w} - t\right)$$

として KKT 条件は

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{w}} = \dfrac{2}{m}\left(X^\mathsf{T}X\boldsymbol{w} - X^\mathsf{T}\boldsymbol{y}\right) + 2\lambda\boldsymbol{w} = \boldsymbol{0}, \\ \dfrac{\partial L}{\partial \lambda} = \boldsymbol{w}^\mathsf{T}\boldsymbol{w} - t \leq 0, \\ \quad \lambda\left(\boldsymbol{w}^\mathsf{T}\boldsymbol{w} - t\right) = 0, \\ \quad \lambda \geq 0 \end{cases}$$

となる。$\lambda = 0$ の場合は最適な解 $\boldsymbol{w}$ がもともと $\boldsymbol{w}^\mathsf{T}\boldsymbol{w} \leq t$ を満たしていたことを表す。もし $\lambda > 0$ ならば $\boldsymbol{w}^\mathsf{T}\boldsymbol{w} = t$ だが、$\boldsymbol{w}$ は第 1 式より $\boldsymbol{w} = \left(X^\mathsf{T}X + m\lambda I\right)^{-1}X^\mathsf{T}\boldsymbol{y}$ である。したがって

$$t = \boldsymbol{w}^\mathsf{T}\boldsymbol{w} = \left[\left(X^\mathsf{T}X + m\lambda I\right)^{-1}X^\mathsf{T}\boldsymbol{y}\right]^\mathsf{T}\left(X^\mathsf{T}X + m\lambda I\right)^{-1}X^\mathsf{T}\boldsymbol{y}.$$

これより λ の値を決めれば t の値は自動的に決まることがわかる [7]。したがって、λ をパラメータとして最初から一般化ラグランジュ関数 $L(\boldsymbol{w}, \lambda)$ から $-\lambda t$ の項を省いた $J(\boldsymbol{w}) = \mathrm{MSE}(\boldsymbol{w}) + \lambda \boldsymbol{w}^{\mathsf{T}} \boldsymbol{w}$ という関数の最小化をすればよい。

参考文献

[1] 伊藤斉志,「進化論的計算手法」, 2005 年. pp. 107–122.

[2]「ノーフリーランチ定理」, Wikipedia, https://ja.wikipedia.org/wiki/ ノーフリーランチ定理, 最終閲覧 2019 年 10 月 27 日

[3] 浅川伸一,「結局どのアルゴリズムが良いのか？」, https://www.cis.twcu.ac.jp/~asakawa/waseda2002/nofreelunch.pdf

[4] "There's no such thing as a free lunch", The Phrase Finder, https://www.phrases.org.uk/meanings/tanstaafl.html, 最終閲覧 2019 年 10 月 27 日.

[5] Ian Goodfellow *et al.* 著, 岩澤有祐ほか訳,「深層学習」, 株式会社ドワンゴ, 2018 年.

[6] Aurélien Géron, 長尾高弘訳.「scikkit–learn と TensorFlow による実践機械学習」, 株式会社オライリージャパン, 2018 年. pp. 128–133.

[7] "Lasso (statistiques)", Wikipédia, https://fr.wikipedia.org/wiki/Lasso_(statistiques). 最終閲覧 2019 年 10 月 27 日.