機械学習原論（体験版）

2018 年 2 月 24 日

1　はじめに

世は空前の人工知能ブーム。猫も杓子も AI だディープラーニングだと、なんでも機械学習に任せてしまえばいいという考えが蔓延している。

機械学習エンジニアの待遇の良さや需要の高さを耳にする機会は多いことだろう。かつて数学徒の目指す職種としてありがちだったクオンツやアクチュアリーと違い、スーツを着ることを強要されるのが少ないという点も魅力的だ。そうなると「自分も機械学習を学んで AI エンジニアになり、ガッポガッポ大儲け。タワーマンションの上層階でたくさん女侍らして、高級ワインを飲むんじゃあ！」という思考になり、機械学習の本に手が伸びる数学徒も少なくないことかと思う。

しかし、勉強をし始めた多くの数学徒は匙を投げてしまう。その理由の大半が「あまりに数学的に適当すぎて読んでいられない」というものだ。

その気持ちはよくわかる。かく言う私もそうだった。世間には「一般向け」と称した、あまりにも粗雑な機械学習の入門書が溢れかえっており、数学的にまともな文献は少数派だ。もちろん一部存在はするが、とてもじゃないが初学者が読んで役立てることはあまりに難しい。

数学的に適当、とは具体的にどういうことなのか。これは大きく分けて 3 つあるように思う。

1. 確率測度と確率密度関数が区別されていない
2. 確率変数であるものと確率変数でないものが区別されていない
3. 目の前の関数がどこからどこへの写像かわからない

ある程度わかってくるとなんとなく忖度して雰囲気で読んでいくことができるようになるが、厳密に読んでいく訓練を受けてきた機械学習初学者の数学徒にはなかなか酷な話だ。

そのため、本書の前半ではなるべく数学科の解析学の授業で習う用語、流儀を用いて、機械学習の基礎を書くことを心掛けた。本書の内容を理解すれば、そう苦労せず一般向けの機械学習資料を読んでいくことができるはずだ。

前半部分の前提知識は数学科学部レベルの関数解析学、確率論、数理統計学とする。後半はその限りではない。

体験版追記：本書は体験版ということで、ページ数も限られるのもあり、具体的な手法についてあまり詳細な解説はできていない。特にベイズを用いた推定の具体的なやり方についてはほとんど解説できなかったと言っていい。興味のある人は自力で調べてみてほしい。本書を読みこなせた人なら難なく学べるはずだ。

2　機械学習入門

機械学習とは、ひたすらに「関数近似」の機構である。

2.1　問題設定

2.1.1　大枠設定

まずおおもとの完備な確率空間 $(\Omega, \mathcal{F}, P)$ を設定する。

特に表記がなければ、すべての可測空間の σ 加法族はボレル集合族、線形空間に対応する体は実数体であるものとする。

線形位相空間 $\mathcal{X}$ を特徴量空間、線形位相空間 $\mathcal{Y}$ をラベル空間と呼び、それぞれの元 x, y を「特徴量」「ラベル」と呼ぶ。

ここで、独立同分布な確率変数列 $\mathcal{D} \coloneqq \{D_i(\omega)\}_{i=1}^n$ を考える。各 D_i は $D_i \colon \Omega \to \mathcal{X} \times \mathcal{Y}$ となる可測関数である。この確率変数列 $\mathcal{D}(= \{D_i\}_{i=1}^n)$ を入力データと呼び、確率変数 D_i の $\mathcal{X}, \mathcal{Y}$ への射影をそれぞれ X_i, Y_i と表記する。

増大情報系 $\mathcal{F}_0 \subset \mathcal{F}_1 (\subset \mathcal{F})$ を考える。$\mathcal{F}_0$ はデータ観測前の保持情報、$\mathcal{F}_1$ はデータ観測後の保持情報である。すなわち

$$\mathcal{F}_1 \coloneqq \mathcal{F}_0 \vee \sigma([D_i]_{i=1}^n) \tag{1}$$

$\mathcal{Y}$ 上の L^2 確率変数の集合を $\tilde{\mathcal{Y}}$ と表記する。また、そのうちガウス型確率変数になるものの集合を $\mathcal{Y}_G, \mathcal{F}_1$-可測で確率 1 で定数になるものの集合を $\mathcal{Y}_{ae}$ と表記する。

連続写像 $f \colon \mathcal{X} \to \tilde{\mathcal{Y}}$ の満たす集合全体を $\mathcal{Z}$ と表記する。モデルに対応した $\mathcal{Z}$ の部分集合 (詳細は後述) を $\mathcal{Z}_M$ と書く。

連続汎関数 $F \colon \mathcal{Z} \to \mathbb{R}$ の集合を $\mathcal{Z}^*$ と書く。$\mathcal{Z}^*$ の位相は弱位相であるとし、$\mathcal{F}_1$-可測な確率変数 $F^* \colon \Omega \to \mathcal{Z}^*$ によって定義される評価関数 $F_{\mathcal{D}} \coloneqq F^*(\omega)$ に対して

$$\hat{f} \coloneqq argmin_{f \in \mathcal{Z}_M} F_{\mathcal{D}}(f) \tag{2}$$

を満たす $\hat{f} \in \mathcal{Z}_M$ を見つける。これが機械学習における大枠の問題設定である。

2.1.2　パラメータを通した $\hat{f}$ の構成方法

パラメータを導入する。パラメータの集合となる 2 つの線形位相空間を Θ_0, Θ と表記する。写像 $\theta^* : \Theta_0 \times \Theta \to \mathcal{Z}$ はあらかじめ一つ固定しておき、確率変数 $\hat{\theta}_0(\omega) \colon \Omega \to \Theta_0, \hat{\theta}(\omega) \colon \Omega \to \Theta_1$ を考える。実際に機械学習問題を解くにあたっては、この $\hat{\theta}_0, \hat{\theta}$ を構成することを通して関数 $\hat{f}$ を構成する。今後は、分かりやすくするために $\theta^*(\theta_0, \theta)$ を $f(\cdot, \theta_0, \theta)$ と表記する。

今後本書では、θ_0 をハイパーパラメータ、θ をパラメータと呼び、単にパラメータといった場合は後者のみを指すものとする。ハイパーパラメータはパラメータの数や後述の事前分布の分散など、パラメータにまつわる数値を決定するものである。

(ここで、聡明な読者は「$\mathcal{Z}_M = Im(\theta^*)$ でいいのでは？」と思うかもしれない。しかし、これではベイズ的な機械学習おいて非常に困る事態が起こってしまう。$Im(\theta^*) \subset \mathcal{Z}_M$ は常に成り立つが、逆は頻度論でしか成り立たない)

$\hat{\theta}$ の $\mathcal{F}_0$-条件付き確率、$\mathcal{F}_1$-条件付き確率をそれぞれ事前確率、事後確率と呼ぶ。またそれらの確率測度に Θ 上のルベーグ測度とのラドンニコディム導関数が存在すれば $p(\theta), p(\theta|\mathcal{D})$ と書き、事前分布、事後分布と呼ぶ。また、$\hat{\theta_0}$ の事前分布を超事前分布と呼ぶ。

2.1.3 問題設定

ここでいよいよ統一的な問題設定を定義する。

特徴量空間 $\mathcal{X}$、ラベル空間 $\mathcal{Y}$、データ $\mathcal{D}$、評価関数 $F_{\mathcal{D}}$、事前分布 $p(\theta)$、事後分布 $p(\theta|\mathcal{D})$、超事前分布 $p(\theta_0)$、パラメータ $\hat{\theta}$ の可測性をまとめて $(\mathcal{X}, \mathcal{Y}, \mathcal{D}, F_{\mathcal{D}}, p(\theta), p(\theta|\mathcal{D}), p(\theta_0), \hat{\theta} - mesurable)$ と表記し、これを今後「機械学習問題」と呼ぶことにする。

機械学習問題において、$\hat{f}$ は次のように計算できる。

$$\hat{f}(x) := E[f(x, \hat{\theta_0}, \hat{\theta})|\mathcal{F}_1] \tag{3}$$

厳密に言えば、これが最適な $\hat{f}$ となるように $\hat{\theta}$ を構成するのが機械学習である。このとき $\hat{f}$ が実現できる範囲の集合こそが $\mathcal{Z}_M$ であり、一般には「仮説集合」と呼ばれる。(ちなみに、$\hat{\theta_0}, \hat{\theta}$ が共に $\mathcal{F}_1$-可測であれば、これは $Im(\theta^*)$ と一致する)

また、$\hat{\theta_0}$ が $\mathcal{F}_0$-可測で $\hat{\theta}$ が $\mathcal{F}_1$-可測である場合を頻度論的機械学習問題と呼び、$\hat{\theta_0}$ が $\mathcal{F}_0$-可測だが $\hat{\theta}$ が $\mathcal{F}_1$-可測でない場合をベイズ論的機械学習問題と呼ぶ。$\hat{f}$ の値域は、ベイズ論の場合 $\mathcal{Y}_G$, 頻度論の場合 $\mathcal{Y}_{ae}$ となる

また、$\hat{\theta_0}, \hat{\theta}$ が共に $\mathcal{F}_1$-可測でない場合を階層ベイズ論的機械学習問題と呼ぶが、本書では扱わない。つまり $\hat{\theta_0}$ はすべてデータ観測前の段階であらかじめ決まっている定数であるものとする。

2.2 ベイズの定理と共役事前分布

この項では、ベイズ論的機械学習についてもう少し掘り下げる。ベイズを用いるため、$\hat{\theta_0}$ は $\mathcal{F}_1$-可測だが、$\hat{\theta}$ は $\mathcal{F}_1$-可測ではない。

$$\hat{f}(x) = \int_{\Theta} f(x, \theta_0, \theta) p(\theta|\mathcal{D}) d\theta \tag{4}$$

となるわけだが、果たしてこんなものを解析的に計算するなどということが、本当にできるのだろうか。

もちろん一般には現実的なリソースで計算できるものではない。実際にこれを近似計算する方法のうち代表的なものは 2 つあり、

1. MCMC を利用する
2. 共役事前分布を利用する

2.2 項ではこの 2 つの手法について解説する。

定理 2.1. ベイズの定理

$\hat{\theta} = \theta$ である時のデータ D_i の尤度を $p(D_i|\theta)$ と表記する。

このとき、任意の $\theta \in \Theta$ に対して、ある定数 W が存在し、次の式が成り立つ

$$p(\theta|\mathcal{D}) = \frac{1}{W} p(\theta) \Pi_{i=1}^{n} p(D_i|\theta) \tag{5}$$

この W を解析的に表現するのは簡単だが、実際にパソコン上で求めるとなると非常に難しい。ここでの 2 つの手法は、どちらもこの定数を計算せずに済む方法である。

2.2.1　マルコフ連鎖モンテカルロ法

まず、基本的な考え方として「比がわかればその具体的な値を知る必要はない」というものがある。

例えば、箱の中にたくさんの青いボールと赤いボールが入っており、赤いボールを引けば 100 万円もらえるが、青いボールを引くとなにももらえないという状況があったとする。このとき、引く人間の関心ごとは「赤いボールと青いボールの個数の比は」であって、それさえわかれば「赤いボールと青いボールがそれぞれ何個入っているか」は気にならないはずだ。そして、ボールを一個引き出したときにもらえる金額の期待値を求めたいときにはどうすればいいか。最も手っ取り早いのは、「適当に大量のボールを取り出して、その中の赤と青の比を見る」ことであろう。

このモンテカルロ法はそういった発想に基づいている。サンプリングされる θ の確率分布が $p(\theta|\mathcal{D})$ に近くなるような方法で、多数の θ をサンプリングしてくるのである。

$\{\theta_1, \theta_2, ..., \theta_N\}$ と N 個の θ を事後分布に従うようサンプリングしてきたとする。そうしたら、平均値 $\frac{1}{N}\Sigma_{i=1}^{N} f(x, \theta_0, \theta_i)$ は、$N \to \infty$ により $\int_\Theta f(x, \theta_0, \theta) p(\theta|\mathcal{D}) d\theta$ に $\mathcal{X}$ 上ほとんど至るところ確率 1 で収束する。

事後分布からの具体的なサンプリング方法については、後の実装部分にて解説する。

2.2.2　共役事前分布

実は、事前分布と尤度関数の間にある関係を仮定すれば、MCMC のような複雑な手法を使うよりも圧倒的に簡単な方法で事後分布を直接計算できる。実際には分析対象がよほどアレなデータでない限り、こちらの方法を用いることが多い。

事前分布と尤度関数が (図 1) のいずれかを満たすとき、事後分布は事前分布のパラメータを変更したものになる。

2.3　代表的なモデル

いよいよここから機械学習における代表的なモデルについて説明する。ここまでの問題設定が難解だった分、ここからはかなりあっさり進んでいくことができる。

解説するモデルは大きく分けて 3 つ、「線形回帰」「カーネル回帰」「ニューラルネットワーク」である。カーネル回帰とニューラルネットワークは後の章で詳細な数学的研究について解説するため、この章では触りだけで済ませる。

2.3.1　線形回帰

最も基本的なモデルである。

$\mathcal{X} := \mathbb{R}^d, \mathcal{Y} := \mathbb{R}, \Theta_0 := \mathbb{N} \times \mathbb{R}, \Theta := \mathbb{R}^{N \times d}$ というのはベイズ・頻度論共通である。ただし、θ_0 の自然数部分を N と表記する。

また、線形回帰においては基底関数というものを定義する。これは有限個の一次独立な関数 $\phi_k : \mathcal{X} \to \mathcal{Y}$ の

組で、$\{\phi_k\}_{k=1}^N$ で表記する。基底関数としてよく使われるのは次の二種類。

$$\text{多項式基底：} \quad (\phi_k(x))_j \coloneqq x_j^k \tag{6}$$

$$\text{ガウス基底：} \quad (\phi_k(x))_j \coloneqq exp\{-\frac{x_j - \mu_{kj}}{2s_{kj}^2}\}(s_{kj}, \mu_{kj} \in \mathbb{R}) \tag{7}$$

ここからは頻度論とベイズ論を分けて解説する。

頻度論の場合：

写像 $\theta^* : \Theta \to \mathcal{Z}$ は次のように定義する。

$$\theta^*(\theta)(x) \coloneqq \Sigma_{j=1}^d \Sigma_{k=1}^N \theta_{kj}\phi_k(x_j) \tag{8}$$

ただし、θ_{kj} は、行列 $\theta \in \mathbb{R}^{N\times d}$ の kj 成分である。

そして、入力データ $\mathcal{D}$ に対して、評価関数 $F_{\mathcal{D}}$: $\mathcal{Z} \to \mathbb{R}$ は次のように定義する。

$$F_{\mathcal{D}}(f) \coloneqq \frac{1}{2}\Sigma_{i=1}^n (Y_i - f(X_i))^2 \tag{9}$$

これを最小化するような行列 $\theta \in \mathbb{R}^{N\times d}$ を求めればよい。

ところで、聡明な読者はこう疑問に思ったかもしれない。「N を際限なく大きくとれば、どんなデータに対しても完璧に近似できるのでは？」と

その予測は正しい。確かに $n < N$ であれば、基底関数がよほど歪でない限り損失関数はかならず 0 にできてしまう。しかし、そのようにして作った関数が、未知のデータに対して力を発揮できるだろうか（未知のデータへの予測性能を汎化性能という）

もちろんそんなことはない。強引な近似を行えば、データのノイズにもフィッティングしてしまい、とてもじゃないが未知のデータに対する予測性能などもてるはずもない（これを過学習という）

かといって、表現能力の低い近似だとろくな予測性能を発揮できない（これを未学習という）

未学習を無理に解決すると過学習を起こし、過学習を無理に解決すると未学習を起こす。（これを、バイアスとバリアンスのトレードオフという）

では、その過学習と未学習の折り合いをつけるにはどうすればよいか。その代表例が「正則化」と呼ばれる手法である。正則化は、簡単に言うと「近似の複雑さにペナルティを与える」手法だ。基本的には次のように定義する。

$$F_{\mathcal{D}}(f) \coloneqq \frac{1}{2}\Sigma_{i=1}^n (Y_i - f(X_i))^2 + \lambda\Sigma_k\Sigma_j|\theta_{kj}|^l \tag{10}$$

ただし、l は自然数 1 もしくは 2 で、$l = 1$ のときこれを「Lasso 回帰」,$l = 2$ のときこれを「Ridge 回帰」という。λ は θ_0 の実数部分となるハイパーパラメータで、0.01 や 0.001 くらいの数字がセットされる場合が多い。

Lasso 回帰にはパラメータが疎になりやすい（値が 0 になるパラメータがたくさん存在する可能性が高い）、また Ridge 回帰は数学的な扱いが楽で解析的に最適化できる、という特徴があり、どちらも一長一短である。

ベイズ論の場合：

ベイズ論手法の特徴は、実際の y の誤差がどの程度になるか x 毎に見積もれる点にある。

（ベイズの話はあとで書き足す）

$$p(y|x) = \int_\Theta p(y|x,\theta)p(\theta|\mathcal{D})d\theta \tag{11}$$

2.3.2　カーネル回帰

一昔前に流行ったサポートベクターマシンなどにつながっていく方法である。

カーネル関数 $k : \mathcal{X}^2 \to \mathcal{Y}$ を決めておく。データ数をこれまでと同じく n とおいて

$$f(x) := \Sigma_{i=1}^{n} a_i k(x, X_j) \tag{12}$$

という形であらわされる関数に対してパラメータを決定する。詳細は 3 章。

2.3.3　ニューラルネットワーク

最近流行りの手法である。まずはシャロウニューラルネットについて解説する。

$\mathcal{X} := \mathbb{R}^d, \mathcal{Y} := \mathbb{C}, \Theta_0 := \mathbb{N} \times \mathbb{R}$, この実数部分を α と書き学習率と呼び、自然数部分を N と書き中間層のノード数と呼ぶ。また、$\Theta := \mathbb{Y}^{d+1 \times N} + \mathbb{C}^N$ で、$\mathbb{Y} := \mathbb{R}$ で、入力層→中間層のパラメータを表す。

活性化関数 $\eta : \mathbb{R} \to \mathbb{C}$ を定義し、それを用いてニューラルネットで構成される関数はこのように書ける。

$$f(x) := \Sigma_{j=1}^{N} C_j \eta(a \cdot x - b) \tag{13}$$

(a,b,C がパラメータのどの部分かはちゃんと書いておくこと)

3　再生核ヒルベルト空間とカーネル法

この章では、前章で簡単にしか触れられなかったカーネル法の、関数解析的な理論の入り口を解説する。

3.1　再生核ヒルベルト空間

定義 3.1. 正定値カーネル

写像 $k : \mathcal{X}^2 \to \mathcal{Y}$ が正定値カーネルであることの定義は

4　ニューラルネットワークの積分表現理論

この章では、「中間層のノード数が無限大であれば学習結果を解析的に決定できる」という理論に基づく、ニューラルネットのアルゴリズムについて簡単に解説する。

中間層のノード数が無限大だった場合、ニューラルネットの計算はある関数 $T(a,b)$ を用いて次のように書ける。

$$f(x) = \int_{\mathbb{Y}^{d+1}} T(a,b)\eta(a \cdot x - b)dadb \tag{14}$$

近似したい関数 $f : \mathbb{R}^d \to \mathbb{C}$ に対して、リッジレット関数 ψ と活性化関数 η が適当な条件を満たす場合、リッジレット作用素と呼ばれる線形作用素 $\mathcal{R}_\psi$ を用いて

$$f(x) = \int_{\mathbb{Y}^{d+1}} (\mathcal{R}_\psi f)(a,b)\eta(a \cdot x - b)dadb \tag{15}$$

が $\mathcal{X}$ 上ほとんどいたるところで成り立つ。つまりこのリッジレット作用素によって、学習結果であるはずの $T(a,b)$ を解析的に計算できてしまう。

もちろん実際には有限のノードしか使えないので近似が必要だが、この理論を用いて初期値を決定すると、精度、速度ともに向上がみられる。

4.1　活性化関数が有界な場合

まずは $\eta \in L^\infty(\mathbb{R} \to \mathbb{C})$ が成り立つ場合について解説する。

4.2　活性化関数が有界でない場合

機械学習に取り組んだことのある方は疑問に思ったことだろう。「じゃあ η が Relu や Swish のような、どう見ても L^∞ じゃない場合はどうするのか」と。

5　数学×機械学習

この章では、ある程度機械学習の経験があり、数学知識も豊富な人向けに、機械学習の数値計算への応用や、逆に数値計算の技法を機械学習に応用する方法について解説する。

5.1　ニューラルネットワークの数値解析への応用

ここでは、PDEs（放物型偏微分方程式）の境界値問題への数値計算に応用する手法について解説する。

PDE の数値解析理論は、現状空間が 4 次元以上だと誤差がひどいことになってしまう。株式の銘柄の数だけ次元が増えるファイナンスなど、SDE や PDE の応用分野ではこれが大いに問題になることも多い。

今回扱う半線形放物型 PDE の境界値問題は次のような形をしている。

$$\frac{\partial u}{\partial t}(t,x) + \frac{1}{2}(\Delta_x u)(t,x) + f(u(t,x),(\nabla_x u)(t,x)) = 0 \tag{16}$$

$$u(T,x) = g(x) \tag{17}$$

ただし、x の取りうる空間は $\mathbb{R}^d$,$f : \mathbb{R} \times \mathbb{R}^d \to \mathbb{R}, g : \mathbb{R}^d \to \mathbb{R}$ は既知の連続関数であるとし、$u : [0,T] \times \mathbb{R}^d \to \mathbb{R}$ は $C^{1,2}$ 級とする。

定理 5.1. 非線形ファインマンカッツの公式

W を d 次元標準ブラウン運動。確率空間をそこから生成されるものとし、Y, Z は適合過程で、f, g は先ほどと同じで、$\xi \in \mathbb{R}^d$ とする。

$$Y_t = g(t, \xi + W_T) + \int_t^T f(Y_s, Z_s)ds - \int_t^T < Z_t, dW_t > \tag{18}$$

は、上記の半線形熱方程式終端値問題の解 u に対して次を満たす

$$Y_t = u(t, X_t), Z_t = \nabla_x u(t, X_t) \tag{19}$$

ただし、$X_t := \xi + W_t$

このブラウン運動は、学習の反復 1 回ごとにオイラー丸山法で実装する。今回、このニューラルネットワークを用いた強化学習で学習させる対象は、写像 $Z_t = F_t(X_t)$。通常の強化学習での評価関数にあたるものは $Y_t = Q(t, Z_t)$ と定義される。

そしてこの評価関数は、終端時刻 T にて、

$$\psi(\theta) = ||Y_T - g(X_T)||^2 \tag{20}$$

で評価され、通常の強化学習と同じく更新されていく。

つまり、時刻が N 分割の場合、「毎回 N 回の行動をとった後に一度だけ報酬が与えられる状況での強化学習」と見なして、X_t に対する最適戦略 Z_t と Y_t を導き出して、それを偏微分方程式の数値解析結果にするという考えである。

通常の Q-学習と違い、状態 X_{t_n} と方策 Z_{t_n} と前時刻での評価 $Y_{t_{n-1}}$ から一意的にこの式で Y_{t_n} が定まるため、その意味では楽だといえる。

$$Y_{t_{n+1}} \approx Y_{t_n} - f(Y_{t_n}, (\nabla_x u)(t_n, \xi + W_{t_n}))(t_{n-2} - t_{n-1}) + < (\nabla_x u)(t_n, \xi + W_{t_n}), W_{t_2} - W_{t_1} > \quad (21)$$

ニューラルネットは各時刻で用意する。つまりこれは RNN ではなく、単純に N 個のニューラルネットが存在するのみである。時刻が違う中間層同士のつながりはない。

損失関数 $\psi(\theta)$ で計算された結果から、新たなパラメータを決定する。すなわち、学習 m 回目のパラメータを Θ_m とおくと

$$\Theta_{m+1} := \Theta_m - \nabla_\theta \psi(\Theta_m) \quad (22)$$

これは要するに確率的勾配法による学習で、ニューラルネットワークに偏微分方程式の正しい関数を近似させようとする取り組みだ。

ただし、普通の確率的勾配法はデータの集合 $\mathcal{D}$ 上の一様測度の元で行われていたが、今回は $C([0,T])$ 上の Wiener 測度の下で確率的勾配法を行っているのである。

「ならば Adam 法などをこの手法に適用してはどうか」「ということは確率過程になるので、抽象 Wiener 空間上の解析を離散化したものと捉えて収束証明ができるのではないか」「カメロン-マルティン部分空間が絡んでくるのではないか」という当然の疑問が浮かんでくる読者もいるだろうが、それは今後の課題である。

5.2 数値解析の深層学習への応用

5.2.1 深層学習とは

これまでの章で扱ったニューラルネットワークは、入力層と隠れ層と出力層が 1 つずつであったが、ここからは隠れ層が 2 つ以上のもの、つまり n 層パーセプトロン $(n > 3)$ を扱っていく。

近年、結果をあげている深層学習アルゴリズムの層数は圧倒的に深くなっている。数年前は結果を出しているのはせいぜい 8 層、22 層程度だったが、2015 年に某大会で 152 層のニューラルネットワークが成果を上げたのを皮切りに、現在では 1000 層を超えるネットワークも使われている。

5.2.2 残差学習とは

なぜそのような超深層学習が可能になったのか、そのカギを握るのは。残差学習という概念だ。

これまでの章を読んでもらえばわかる通り、通常のニューラルネットワークは入力 x に対して、学習させたい写像 $H(x)$ を直接学習させる。

しかしこの手法は 3〜6 層程度のパーセプトロンなら有効だが、層の数が劇的に増えてくると意味をなさなくなってくる（勾配消失・発散)。それを突破するためにいろいろな方法が考えられたが、どれも中間層の数が数百、数千層単位になる超深層学習には対応できなかった。

そこで登場するのが残差学習である。学習させる対象は、$H(x)$ ではなく、残差 $F(x) := H(x) - x$、これを学習させ、そこに x を加えた値を次の層の入力とする。

ここで、この残差関数 F の、n 番目の層での値を $F(t_n, \cdot)$ と表記する。また、n 番目の層の入力を x_{t_n} と表記する。

すると、この残差学習はこう書ける。

$$x_{t_{n+1}} = x_{t_n} + F(t_n, x_{t_n}) \quad (23)$$

これまで数学を学んできた読者ならすでにお気づきだろう。これは常微分方程式のオイラー近似に他ならない。

すなわち、残差学習による超深層ニューラルネットで行われていることは、常微分方程式の離散近似なのである。

近年の主な超深層学習と常微分方程式数値計算の関係を簡単にまとめると

$$ResNet- \quad \text{前進オイラー法} \tag{24}$$
$$RevNet- \quad \text{後退オイラー法} \tag{25}$$
$$PolyNet- \quad \text{前進オイラー法（連立常微分方程式）} \tag{26}$$
$$FractalNet- \quad \text{2 次ルンゲクッタ法} \tag{27}$$

さらに [5] の著者は、別の常微分方程式の数値計算技法を使うことで、さらなる計算の高速化及び精度の向上を可能にした。

5.2.3 確率的超深層学習と確率微分方程式

ここまで書かれたのは、決定論的な超深層学習である。ここにノイズ付加やドロップアウトなどの確率要素を加えると、必然これらの議論は確率微分方程式の話に突入する。

6　あとがき

如何だっただろうか。正直、（原稿締め切りと就活の狭間で慌てふためいたせいで）まだまだ解説しきれていない部分はたくさんある。

しかし、ここまで読んでくれた方なら、一体機械学習とは何をやっているのか、機械学習の背後にはどれだけ数学的な議論が隠れているのか、その一端を知ることができているはずだ。

本書の内容を理解した方であれば、きっと一般向けの機械学習書籍を読んでいくこともできるようになっているかと思う。是非とも、機械学習の勉強を進め、積極的に実践経験を積んでいってほしい。この手の分野は自分で手を動かしてナンボだ。本書はそこに辿り着くまでの手助けをしているに過ぎない。

では、読者の数学徒諸君と、機械学習の議論ができる日を楽しみにしながら、いったん筆を置くことにする。本書は常に修正・加筆に取り組んでいるため、意見等があったら気軽にメッセージを送ってほしい。面白い機械学習の数学的研究の資料の紹介なども大歓迎だ。

参考文献

[1] 確率論 (実教理工学全書), 西尾真紀子 (1978)

[2] Machine Learning: A Probabilistic Perspective (Adaptive Computation and Machine Learning series),Kevin.P.Murphy (2012)

[3] ルベーグ積分入門-使うための基礎と応用-, 吉田伸生 (2006)

[4] 深層ニューラルネットの積分表現理論, 園田翔 (2017)

[5] Deep learning-based numerical methods forhigh-dimensional parabolic partial differential equationsand backward stochastic differential equations,Weinan E,Jiequn Han,Arnulf Jentzen(2017)

[6] BEYOND FINITE LAYER NEURALNETWORKS:BRIDGING DEEP ARCHITECTURES AND NUMERICAL DIFFERENTIAL EQUATIONS,Yiping Lu,Aoxiao Zhong,Quanzheng Li(2017)

[7] カーネル法の新展開ーその理論と応用ー, 福水健次 (2012)