機械学習原論

2017 年 12 月 28 日

1　確率論の基礎の復習

この章では、機械学習の数学的定式化に必要な確率論の基礎中の基礎を復習する。

目的はあくまで機械学習への応用であるため、細々とした議論は行わず、機械学習の基礎を定式化するために必要な最低限の項目だけ解説する。

ここの内容で対応できるのは 3 章まで。4 章以降の数学に関しては、各自関数解析や確率解析の書物を参照すること

1. 確率空間と確率変数

標本空間 Ω に対し、次を満たす部分集合族 $\mathcal{F}$ を σ 加法族と呼ぶ

$$1.\phi, \Omega \in \mathcal{F} \tag{1}$$
$$2.A \in \mathcal{F} \to A^c \in \mathcal{F} \tag{2}$$
$$3.\text{可算個の}\Omega\text{の部分集合 } A_1, A_2, ... \text{, があり、任意の } n \text{ に対して } A_n \in \mathcal{F} \text{ なら、} \cup_{n=1}^{\infty} A_n \in \mathcal{F} \tag{3}$$

写像 $P : \mathcal{F} \to [0, 1]$ が次を満たすとき、P を確率測度という

$$1.P(\Omega) = 1 \tag{4}$$
$$2.\text{互いに素な可算個の集合 } A_1, A_2, ..., \in \mathcal{F} \text{ に対し、} P(\cup_{n=1}^{\infty} A_n) = \Sigma_{n=1}^{\infty} P(A_n) \tag{5}$$

この $(\Omega, \mathcal{F}, P)$ の組を、確率空間と呼ぶ

$\mathcal{B}(\mathbb{R})$ で、$\mathbb{R}$ の開集合をすべて含む最小の σ 加法族を表し、ボレル集合族と呼ぶ。

写像 $X : \Omega \to \mathbb{R}$ が次の条件を満たすとき、（1 次元実）確率変数であるという

$$\text{任意の } B \in \mathcal{B}(\mathbb{R}) \text{ に対して、} X^{-1}(B) \in \mathcal{F} \tag{6}$$

ここで、$(\mathbb{R}, \mathcal{B}(\mathbb{R}), P \circ X^{-1})$ は確率空間となる

確率変数 X に対して、分布関数 $F_X : \mathbb{R} \to [0, 1]$ を次のように定義する

$$F_X(x) := P(X \leq x) \tag{7}$$

今回扱う確率変数は、その確率変数に対応する確率測度 $P \circ X^{-1}$ が $\mathbb{R}$ 上のボレル測度 μ に対して絶対連続であるものとし、ボレル測度とのラドンニコディム導関数を確率密度関数と呼ぶ

すなわち

$$f_X(x) := \frac{d(P \circ X^{-1})(x)}{d\mu(x)} \tag{8}$$

Ω の部分集合族 $\mathcal{G}$ が、σ 加法族の条件を満たし、さらに $\mathcal{G} \subset \mathcal{F}$ が成り立つなら、$\mathcal{G}$ を部分 σ 加法族という
Ω の部分集合族 $\mathcal{H}$ に対して、$\sigma(\mathcal{H})$ で $\mathcal{H}$ を含む最小の σ-加法族を表すものとする。
また、確率変数 Y に対して、$\sigma(Y)$ で Y を可測にする最小の σ-加法族を表すとする。

2. 独立性と条件付き期待値

集合 $A, B \in \mathcal{F}$ が独立であるとは、$P(A \cup B) = P(A)P(B)$ が成り立つことである。
部分 σ-加法族 $\mathcal{G}_1, \mathcal{G}_2$ が独立であるとは、任意の集合 $A_1(\in \mathcal{G}_1), A_2(\in \mathcal{G}_2)$ が独立となることである。
確率変数 X, Y が独立であるとは、 σ-加法族 $\sigma(X), \sigma(Y)$ が独立となることである。
集合 $A \in \mathcal{F}$ の、$B \in \mathcal{F}$ に対する条件付き確率を次のように定義する。（確率論入門は後日書く）

2　機械学習入門

ここからは本格的に機械学習の内容を学んでいく。

工学への応用であるため、数学的な厳密性に関しては数学書に比べればかなり適当なものになるがご容赦願いたい。

1. 問題設定

機械学習のタスクの多くは、次のような問題設定で一般化できる。

「分類」や「回帰」、「次元削減」などの違いについてはこの段階では区別しないのであしからず。

ちなみに、これで書けない機械学習のタスクも、その大半はこの問題設定のわずかな改変で記述できる。

確率空間 $(\Omega, \mathcal{F}, P)$ を定義する。

考察に用いたい特徴量の集合を特徴量ベクトル $\boldsymbol{x}$。それに対して教師データを y と表記する。

$$\boldsymbol{x} \in \mathbb{R}^d$$
$$y \in \mathbb{R}^n$$

これに対して、$\boldsymbol{x}$ と y が紐づけられた m 個のデータ $\mathcal{D} := [X_1, X_2, ..., X_m] := [(\boldsymbol{x}_1, y_1), ..., (\boldsymbol{x}_m, y_n m)]$ が与えられた上で、できるだけ良い分類関数 $\hat{y} : \mathbb{R}^d \to \mathbb{R}^n$ を見繕いたい。

各 X_i は、確率空間上の独立同分布な確率変数 $X_i : \Omega \to \mathbb{R}^d \times \mathbb{R}^m$ である。ただし、値域となる空間の σ-加法族はボレル集合族をとるものとする。

ここで、データ観測前の所持情報を $\mathcal{F}_0$, データ観測後の所持情報を $\mathcal{F}_1$ と置く。これはどちらも $\mathcal{F}$ の部分 σ-加法族で、$\mathcal{F}_0 \subset \mathcal{F}_1$ であれば、これらは増大情報系の条件を満たし、次のように記述できる。

$$\mathcal{F}_1 = \mathcal{F}_0 \vee \sigma([X_i]_{i=1}^m) \tag{9}$$

写像 $\hat{y} : \mathbb{R}^d \to \mathbb{R}^n$ の満たす集合を Y とおく。Y の位相は後述の F を連続関数にする最弱の位相と定義する。

入力データ $\mathcal{D}$ に対して、最適な写像 $\hat{y}$ を導き出したい。

ただし、Y の任意の元を取ってこれるかと言えばそうではなく、ある程度制約をつけなければ役に立たない。

選択されたモデル M(詳細は後述) に対して、$Y_M (\subset Y)$ でモデルに対応する関数の集合とし、この上で最適化を行っていく。

$$\tilde{y} := argmax_{y \in Y_M} [F_{\mathcal{D}}(y)] \tag{10}$$

ただし、$F_{\mathcal{D}}$ はモデルと入力データから定まる $F_{\mathcal{D}} : Y \to \mathbb{R}$ の汎関数である。

そして、後述の様々な技法の中から使用されるものを「選択されたモデル M」と呼ぶものとする。$(\mathbb{R}^d, \mathbb{R}^n, \mathcal{D}, M)$ の組を「汎化機械学習問題」と呼び、そこから導き出された最適な写像 $\hat{y}$ を、「汎化機械学習問題の解」と呼ぶ。

2. モデル設定

ここでは、ベイズとサポートベクターマシンとニューラルネットワークを除いたモデルについて解説していく。(サポートベクターマシンとニューラルネットワークが絡むと、絡んでいない場合と比べて数学的な議論が圧倒的に濃くなるため。ベイズは浅い範囲に限っても語る内容が多い)

「これ機械学習じゃなくて普通の統計学じゃないか」と思われるであろう要素を多分に含んでいるが、そもそも機械学習と統計学の明確な線引きは不可能であることに留意していただきたい。

1. 線形識別 (2 クラス)

特徴量ベクトルの空間 $\mathbb{R}^d$ を 2 つに分類する。教師データの集合は $\mathbb{R}^n, n=1$ として、Y_M を次のように定義する。

$$Y_M := \{\tilde{y} \in Y | A^t \boldsymbol{x} + b = 0 \text{ を満たすアフィン部分空間が } d-1 \text{ 次元になる } A \in \mathbb{R}^d \times \mathbb{R}^n, b \in \mathbb{R}^n\} \quad (11)$$

また、これらの関数の評価関数はこのように定義できる。

$$\begin{aligned} F_{\mathcal{D}}(\tilde{y}) :&= \{ \text{ 各データの、上述のアフィン部分空間までの距離の和 } \} \\ &= -\frac{1}{m}\Sigma_{i=1}^{m}|X_i - \rho(X_i)|^2 \end{aligned} \quad (12)$$

ただし、$\rho(X)$ で、X のアフィン部分空間への射影であるとする。

こうして求められた最適な $A.b$ に対して、$Ax+b>0$ か $Ax+b<0$ かでデータを二つのクラスに分類する

2. ロジスティック回帰

基本的には 2 クラスの線形識別と同じく、$\mathbb{R}^d$ を 2 つにわけるのだが、今回はすっぱりと分けてしまうわけではなく、片方のクラスに属する「確率」を求める。

線形回帰は $\tilde{y}(x) := Ax+b$ の A, b はすべて定数倍に対して同値だったが、今回はそうもいかない。
Y_M は前回と同じでいいが、評価関数が大きく変わる。

$$\begin{aligned} z_i :&= \tilde{y}(x_i) \\ \sigma(z) :&= \frac{1}{1+exp(-z)} \\ g(y) :&= 1_{y>0} \\ F_{\mathcal{D}}(\tilde{y}) :&= \{ \text{ 交差エントロピー } \} \\ &:= \frac{1}{m}\Sigma_{i=1}^{m} - (g(y_i)log(\sigma(z_i)) + (1-g(y_n)log(1-\sigma(z_i)))) \end{aligned} \quad (13)$$

これによって、未知のデータ x に対して、それに対応する $y>0$ である確率が $\sigma(\hat{y}(x))$ で求められる。

3. 線形識別 (多クラス)

基本的には 2 クラスの線形識別と同じだが、$n=1$ ではなく、分類したいクラスの数が m なら、$m \leq 2^n$ となるように n を取り、n 個のアフィン部分空間を生成する。

4. 決定木

基本的には多クラス線形識別と同じだが、n はあらかじめ決めておかず、必要に応じて増やしていく。

どういうことかというと、

5. ランダムフォレスト

決定木の改良である。データ $\mathcal{D}$ からランダムに部分集合をいくつも取り出し（重複はもちろん可)、それぞれに決定木学習を行う。

そして最後に、各点に対して部分集合ごとの結果を多数決もしくは平均値をとることで、学習結果とする。

6. 主成分分析

別名、次元削減。後述のオートエンコーダと数学的には同値である。

超高次元のデータに対して、いきなり機械学習的な分析を行うのは計算量や精度の問題からなかなか無謀な話であり、いったんこの手法で前処理をして次元を削減してから分析を行うことが多い。

$\mathbb{R}^d$ 上のデータに対してこれを $\tilde{d}$ 次元まで削減したいとする。

まず次のような 1 次元アフィン部分空間 A_1 を考える。

$$A_1 := \{x \in \mathbb{R}^d : \hat{y}_1(x) = 0\} \tag{14}$$

$$\tilde{y}_1(x) := a_1 \cdot x + b_1 (a_1, b_1 \in \mathbb{R}^d) \tag{15}$$

評価関数は

$$F^1_{\mathcal{D}}(\tilde{y}_1) := \frac{1}{m}\Sigma_{i=1}^m \rho_1(X_i)^2 - (\frac{1}{m}\Sigma_{i=1}^m \rho_1(X_i))^2 \tag{16}$$

ただし、ρ_1 は A_1 への射影である。

評価関数を最大化するということは、「データの射影の分散が最大となるような 1 次元アフィン部分空間を求める」ということである。（分散が大きいとなぜいいか。それは、分散が大きいほど元のデータの情報を残していると考えられるためである）

この射影ベクトルを「第一主成分」と呼ぶ。

次に、任意の元が A_1 と直交するような線形部分空間 A_2 を考える。評価関数や定義の仕方は上記と同じ。

これを何度も $A_{\tilde{d}}$ が計算できるまで繰り返す。

元の情報をこのアフィン部分空間の族がどの程度残しているかについて測る指標が「累積寄与率」と呼ばれる数値である。多くの場合、$\tilde{d}$ 次元までの主成分分析でこれが 80 %を超えれば、それなりにいい近似であるとされる。

3. 実装について

3　ベイズ学習

　ベイズ統計に深入りすると一生機械学習の話に戻れなくなるので、とりあえず機械学習の場でよく使われるパラメトリックなベイズ学習の入り口を簡単に解説する。

　ノンパラベイズは魔境なのでそれなりの覚悟を持って臨むこと。本書では解説しない。

1. ベイズ学習とは

　2 章のモデル設定で、入力データを独立同分布な確率変数として扱ったが、その確率変数の分布をデータから推定するのがベイズ学習である。

　真の確率密度関数を $f(x)$ とおき、データを元に構成したそれの推定量を $\hat{f}_{\mathcal{D}}(x)$ と書く。任意の x に対して、なるべく真の値 $f(x)$ に近い $\mathcal{F}_1$-可測な推定量 $\hat{f}_{\mathcal{D}}(x)$（これを予測分布と呼ぶ）を見繕いたい。

　パラメータを θ、Θ をパラメータが取りうる値の集合であるとする。これにより、分布を求めたい確率変数 X から生成される確率測度の族 $\{P_\theta^X\}_{\theta\in\Theta}$ を考えることができる。

4　サポートベクターマシン

ニューラルネットワークほどではないが、数学的に濃く、未解決の部分も多い領域である。

5　ニューラルネットワーク

本書のメインである。この章では、中間層が一つの場合を扱う。

1. ニューラルネットワークとは

2. ニューラルネットワークの積分表現理論

前項で解説したニューラルネットの Σ を用いた表現について掘り下げてみよう。これに対して中間層のノード数を無限に増やすとどうなるか。これは積分を用いて表記することができる。

$$g(x) = \int_{\mathbb{Y}^{d+1}} T(a,b)\eta(a \cdot x - b)dadb \tag{17}$$

このようなニューラルネットワークを「積分的ニューラルネットワーク」と呼び、そのパラメータは後述の「リッジレット変換」により解析的に決定することができる。

もちろん、実際にニューラルネットワークを実装する場合は中間層のノード数を実数濃度にするわけにはいかないので、有限個のノードで近似する。この近似に用いるべき (a,b) は、後述の「† オラクルサンプリング †」により決定するといい値を取り出せる確率が高い。

重みづけの初期値を、この † オラクルサンプリング † とリッジレット変換により決定すると、誤差逆伝播を使わなくとも初期状態で非常に高い精度を出せる。局所解にもまずハマらない。

こういった「機械学習で行うことを連続化して、解析的に答えを求める。そしてその解析解やその離散化を初期値として学習を行うことで、通常よりも圧倒的に早い収束が見込める」という手法は、機械学習全般において数学の専門知識を活かしアルゴリズムの改善を行うにあたって極めて基本的な考え方になっていくと思われる。

関数 $f \in L^1(\mathbb{R}^d \to \mathbb{C})$ を、活性化関数 $\eta \in L^\infty(\mathbb{R} \to \mathbb{C})$ を持つニューラルネットワークで近似することを考える。

定義 5-1：リッジレット変換

f のリッジレット関数 $\psi \in L^\infty(\mathbb{R} \to \mathbb{C})$ によるリッジレット変換は、次のように定義される。

$$(\mathcal{R}_\psi f)(a,b) := \int_{\mathbb{R}^d} f(\boldsymbol{x})\overline{\psi(a \cdot \boldsymbol{x} - b)}|a|^s d\boldsymbol{x} \tag{18}$$

ただし、s は非負実数。0 か 1 を使うことが多い。

実際に学習させるにあたって、リッジレット関数をどのように選べばいいのかは次項にて解説する。

3. 積分表現理論を応用した学習高速化

応用数学には予言性が必要である。

2 章は著者自ら行った定式化だが、「別の学問のこういう話を数式で書けた」「解析的に計算できた」だけでは、その研究に学術的な意義はない。応用先の学問に新しい結果をもたらさなければ、せいぜい「数学徒がその学問に入門しやすくなる」以上の意味はないのである。（カタストロフィー理論はそれができずに廃れた）

機械学習で言うと、新しい結果とは大雑把に言えば「高速化ができた」「精度が上がった」「学習がうまくいくことを数学的に厳密に証明した」の３つに分けられる。

この項で解説するのは、†オラクルサンプリング†とリッジレット解析を用いた学習の具体的なアルゴリズムである。離散化に伴いもちらん値はずれるので学習が必要だが、答えの近くから始められるので圧倒的に速い収束が見込める。その速さたるや、もはやバックプロパゲーションが必要ないまである。

4. ボルツマンマシン

非常に名前がかっこいい機構である。本書では、ボルツマンマシンのうち、最もよく使われている「制限ボルツマンマシン」について解説する。

6　ディープラーニング

いよいよここから、第三次人工知能ブームを巻き起こしたとして巷で話題の深層学習に入り込んでいく。数学的なレベルは前章よりも大幅に上がるので、前章で知識不足を感じた方は関数解析や確率解析, 圏論の書籍を片手に読むことを勧める。

深層学習とは

前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

7　強化学習

近年、Deep Zen GO などの、

8　ニューラルネットワークの数値解析への応用

ここでは、強化学習による BSDE（後退確率微分方程式）の確率制御問題や、それをさらに 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。量子の世界に踏み込まない物理学や工学ならそれでもいいかもしれないが、株式の銘柄の数だけ次元が増えるファイナンスや 11 次元の物理学、その他 SDE や PDE の応用分野の多くではこれが大いに問題になる

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

$$\frac{\partial u}{\partial t}(t,x) + \frac{1}{2}(\Delta_x u)(t,x) + f(u(t,x),(\nabla_x u)(t,x)) = 0 \tag{19}$$

$$u(T,x) = g(x) \tag{20}$$

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。

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。

$$Y_t = g(t, \xi + W_T) + \int_t^T f(Y_s, Z_s)ds - \int_t^T < Z_t, dW_t > \tag{21}$$

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

$$Y_t = u(t, X_t), Z_t = \nabla_x u(t, X_t) \tag{22}$$

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、

$$\psi(\theta) = ||Y_T - g(X_T)||^2 \tag{23}$$

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。

$$Y_{t_{n+1}} \approx Y_{t_n} - f(Y_{t_n}, (\nabla_x u)(t_n, \xi + W_{t_n}))(t_{n-2} - t_{n-1}) + < (\nabla_x u)(t_n, \xi + W_{t_n}), W_{t_2} - W_{t_1} > \quad (24)$$

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

$$\Theta_{m+1} := \Theta_m - \nabla_\theta \psi(\Theta_m) \quad (25)$$

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。

9　数値解析の深層学習への応用

今度は逆に、深層学習に対して、微分方程式の数値計算を応用する方法について解説する。

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。

1. 残差学習

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。

3 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。

$$x_{t_{n+1}} = x_{t_n} + F(t_n, x_{t_n}) \tag{26}$$

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。

そう。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学習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

$$ResNet - \text{前進オイラー法} \tag{27}$$
$$RevNet - \text{後退オイラー法} \tag{28}$$
$$PolyNet - \text{前進オイラー法（連立常微分方程式）} \tag{29}$$
$$FractalNet - 2 \text{ 次ルンゲクッタ法} \tag{30}$$

さらに [1] 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。

2. 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。

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] 深層ニューラルネットの積分表現理論, 園田翔 (2017)

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

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

10　以下、第一稿のメモ (スルー推奨)

ニューラルネットワーク２章３の分類に含めても構わなかったのだが、あまりにも長くなるため、ニューラルネットワーク（ディープラーニング）に関してだけは、２個の章に分けて解説する。数学的に最も濃厚かつ難解で、数学科生にとっては一番楽しめる内容になることかと思う。

1. ニューラルネットワークとは

（人工）ニューラルネットワークとは、脳回路の人工的再現である。一つ以上の入力を受けて、値を出力する。ここで言う入力の数とは、先述のモデルで言えば d のことである。（以降、他の記号はすべてリセットされたし。$K = \mathbb{R}$ とする）

まずは隠れ層が 1 つである場合を考える

入力層から入力された値のある重みづけを、隠れ層では活性化関数 $\eta : \mathbb{R} \to \mathbb{C}$ の引数にする。また、この先にも重みづけを行う。すなわち、入力データを $\boldsymbol{x}$, 入力後の重みづけベクトルの各成分を $a_j, b_j \in \mathbb{R}$, 関数で写像を飛ばしたあとの重みづけベクトルの各成分を $C_j \in \mathbb{C}$ とおくと、このニューラルネットワークが示す関数 $g : \mathbb{R} \to \mathbb{C}$ は

$$g(\boldsymbol{x}) := \Sigma_{j=1}^{n} C_j \eta(a_j x_j - b_j) \tag{31}$$

と表せる。ただし n は隠れ層のノード数である。

入力に対して、正しい出力がわかっている状況で正しい出力が出るようにするにはこの重みづけのパラメータを調整すればよい。そのようにして、正しい関数 $f : \mathbb{R}^d \to \mathbb{C}$ の近似 g を探索するのが「ニューラルネットワーク」である。

ところで、上記の式は、ノード数 $n \to \infty$ とすることで、積分にすることができる。すなわち

$$g(\boldsymbol{x}) := \int_{\mathbb{Y}^{d+1}} T(a,b)\eta(a \cdot x - b) d\mu(a,b) \tag{32}$$

ただし、$T(a,b)$ は、複素数列 C_j の連続版に当たる写像 $T : \mathbb{Y}^{d+1} \to \mathbb{C}$ であり、$a \cdot x$ はユークリッド内積

このような積分の存在は仮定する（T, η に対してこの積分が収束するような測度 μ のみを扱う）。また、μ にルベーグ測度との絶対連続性を課さなければ、離散化はこの特別な場合として書ける。

2. リッジレット作用素

ではその関数 $T : \mathbb{Y}^{d+1} \to \mathbb{C}$ を求めるにはどうすればいいか。これは実は既存研究で解析的に書ける。

関数 $f \in L^1(\mathbb{R}^d; \mathbb{C})$ の、$\psi \in L^\infty(\mathbb{R}; \mathbb{C})$ に対するリッジレット作用素 $\mathcal{R}_\psi$ を次のように定義する

$$(\mathcal{R}_\psi f)(a,b) := \int_{\mathbb{R}^d} f(\boldsymbol{x})\overline{\psi(a \cdot \boldsymbol{x} - b)}|a|^s d\boldsymbol{x} \tag{33}$$

次に、双対リッジレット作用素を定義する。

$\mathcal{R}_\eta^* \cdot : L^1(\mathbb{Y}^{d+1}; \mathbb{C}, |a|^{-s} da db) \times L^\infty(\mathbb{R}; \mathbb{C}) \to L^\infty(\mathbb{R}^d)$ を定義する

$$(\mathcal{R}_\eta^* T)(\boldsymbol{x}) := \int_{Y^{d+1}} T(a,b)\eta(a \cdot \boldsymbol{x} - b)|a|^{-s} d\mu(a,b) \tag{34}$$

これは適当な条件（要検証）の元で、リッジレット作用素 $\mathcal{R}_\eta$ の双対作用素となる。

定義：
次の積分が有界で非零のとき、ψ, η に対する核が許容的であるという

$$K_{\psi,\eta} := (2\pi)^{d-1} \int_{\mathbb{R}} \frac{\overline{\hat{\psi}(\zeta)}\eta\zeta}{|\zeta|^d} d\zeta \tag{35}$$

定理：再構成公式
ψ, η は上述の関数と同一集合上の元であるとし、また核は許容的であるとする。このとき、次の式が成り立つ。

$$f(\boldsymbol{x}) = \int_{\mathbb{Y}^{d+1}} (\mathcal{R}_\psi f)(a,b)\eta(a \cdot \boldsymbol{x} - b) da db \tag{36}$$

よって、$T := \mathcal{R}_\psi f$ と置くと、この積分的ニューラルネットワークは関数 f と同じになる。実際のニューラルネットワークではこれを離散化するため、関数 f の近似となる。

3. 離散化
ここまで、連続的に三層パーセプトロンを考察してきたが、実際のニューラルネットワークにおいて、隠れ層のノード数は有限である。そのため、中間層素子のの離散化を考える必要がある。

活性化関数 η に対して、すべての中間層素子の集合を「辞書」と呼ぶ。

$$\mathcal{D} := \{h(\cdot; a, b) | (a, b) \in \mathbb{Y}^{d+1}\} \tag{37}$$

ただし、$h(x; a, b) := \eta(a \cdot \boldsymbol{x} - b)$

辞書の部分集合 D（以下、部分辞書と呼称する）と、関数 $T : \mathbb{Y}^{d+1} \to \mathbb{C}$ の内積を次のように定義する

$$(DT)(\boldsymbol{x}) := \int_D T(a,b)h(\boldsymbol{x}; a, b) d\mu(a,b) \tag{38}$$

ただし、この積分は実際には適宜局所座標系をとらないと面倒である。
ここで、このような形でかけるとき、これは実際に実装されるニューラルネットワークの挙動になる。

$$(DT)(\boldsymbol{x}) = \Sigma_D T(a,b)h(\boldsymbol{x}; a, b) \tag{39}$$

部分辞書 D が可算集合であるとき、これを「離散化」という。
離散化の評価は必要に応じて様々な関数空間のノルムで行われる。
困ったことに、$\mathcal{D}$ の閉方はヒルベルト空間であるとは限らないで、一般には正規直交基底をとれない。

4. 一般化されたリッジレット作用素

実際のニューラルネットワークでは、様々な関数が活性化関数として使われる。
ここで問題になってくるのは、ReLU（$\eta(x) := x_+$）のような、「実際によく使われるが上記の定義では定式化できない（L^∞ の元ではない）」場合である。
そこで、この章ではリッジレット作用素を超関数論で一般化し、活用範囲を大幅に広げる

まず、上述の $(a, b) \in \mathbb{Y}^{d+1}$ を極座標変換する

$$u := a/|a|, \alpha := 1/|a|, \beta := b/|a| \tag{40}$$

特に誤解の恐れがない状況では、座標系の取り方によらずパラメータの空間を $\mathbb{Y}$ と表記する

リッジレット変換を次のように超関数として定義する
$f \in \mathcal{X}(\mathbb{R}^d)$ の $\psi \in \mathcal{Z}(\mathbb{R})$（空間の定義は後述）によるリッジレット変換を、こう定義する

$$(\mathcal{R}_\psi f)(u, \alpha, \beta) := \overline{\int_{\mathbb{R}} \overline{Rf(u, az + \beta)}\psi(z)dz} \tag{41}$$

表 5.2: リッジレット変換が定義できる $\mathcal{X}$ と $\mathcal{Z}$ の組み合わせおよび対応する値域。$\mathcal{B}, \mathcal{A}, \mathcal{Y}$ は $\mathscr{R}_\psi f(u, \alpha, \beta)$ をそれぞれ $\beta, (\alpha, \beta), (u, \alpha, \beta)$ の関数とみなした場合のクラスを表す。

$f(x)$	$\mathrm{R}f(u, p)$	$\psi(z)$	$\mathscr{R}_\psi f(u, \alpha, \beta)$		
$\mathcal{X}(\mathbb{R}^m)$	$\mathcal{X}(\mathbb{S}^{m-1} \times \mathbb{R})$	$\mathcal{Z}(\mathbb{R})$	$\mathcal{B}(\mathbb{R})$	$\mathcal{A}(\mathbb{H})$	$\mathcal{Y}(\mathbb{Y}^{m+1})$
$\mathcal{D}$	$\mathcal{D}$	$\mathcal{D}'$	$\mathcal{E}$	$\mathcal{E}$	$\mathcal{E}$
$\mathcal{E}'$	$\mathcal{E}'$	$\mathcal{D}'$	$\mathcal{D}'$	$\mathcal{D}'$	$\mathcal{D}'$
$\mathcal{S}$	$\mathcal{S}$	$\mathcal{S}'$	$\mathcal{O}_\mathcal{M}$	$\mathcal{O}_\mathcal{M}$	$\mathcal{O}_\mathcal{M}$
$\mathcal{O}'_\mathcal{C}$	$\mathcal{O}'_\mathcal{C}$	$\mathcal{S}'$	$\mathcal{S}'$	$\mathcal{S}'$	$\mathcal{S}'$
L^1	L^1	$L^p \cap C$	$L^p \cap C$	$\mathcal{S}'$	$\mathcal{S}'$

証明は各クラスの定義に従って“積分”の収束性を確認する。詳細は 付録 A.2 を見よ。表において $\mathcal{Z}$ の大きさは畳み込み $\mathcal{B} = \mathcal{X} * \mathcal{Z}$ が存在す

ただし、R はラドン変換
（ここに自分で図を書いておく）

定理
この組み合わせにおいて、作用素 $\mathcal{R}_\cdot(\cdot) : \mathcal{X}(\mathbb{R}^d) \times \mathcal{Z}(\mathbb{R}) \to \mathcal{Y}(\mathbb{Y}^{d+1})$ は、双線形写像である

定理
$\mathcal{X} = L^1$ とし、$\psi \in \mathcal{S}(\mathbb{R})$ を固定する。このときリッジレット作用素 $\mathcal{R}_\psi$ は有界作用素

定義
関数 $T \in \mathcal{Y}(\mathbb{Y}^{d+1})$ の、活性化関数 $\eta \in \mathcal{W}(\mathbb{R})$ による双対リッジレット変換を次のように定義する

$$(\mathcal{R}^*_\eta T)(\boldsymbol{x}) := lim_{\delta \to \infty, \varepsilon \to 0} \int_{\mathbb{S}^{d-1}} \int_\varepsilon^\delta \int_{\mathbb{R}} T(u, \alpha, u \cdot x - \alpha z)\eta(z) \frac{dz d\alpha du}{\alpha^d} \tag{42}$$

$\mathcal{R}_\eta^*$ は、存在すれば $\mathcal{R}_\eta$ の双対作用素
活性化関数は超関数でもよい。次に許容条件について定義する

定義
$(\psi, \eta) \in \mathcal{S} \times \mathcal{S}'$ が許容的であるとは、任意の原点を含む近傍 Ω に対し、次の積分が 0 でない値に収束し $\hat{\eta}$ が局所可積分であることを言う

$$K_{\psi,\eta} := (2\pi)^{d-1}\left(\int_{\Omega/\{0\}} + \int_{\mathbb{R}/\Omega}\right)\frac{\overline{\hat{\psi}(\zeta)}\hat{\eta}(\zeta)}{|\zeta|} \tag{43}$$

また、この二つの関数が等しいとき、これを自己許容的という

定理（リッジレット変換の L^2 拡張性）
$\psi \in \mathcal{S}$ は自己許容的であるとし、$K_{\psi,\psi} = 1$ とする。このとき、$L^1 \cap L^2(\mathbb{R}^d)$ 上のリッジレット作用素は $L^2(\mathbb{R}^d)$ 上の作用素に一意に拡大でき、$||\mathcal{R}_\psi f||^2 = ||f||^2$（等距離写像）

具体的な再構成公式や離散化などは参考文献を参照されたし
5. 実装について
一般には、誤差電波法が用いられる。リッジレット関数の選び方は

6. 強化学習
これはニューラルネットワークの一分野ではないのだが、近年 DQN（ヤンキーのことではない）などによりニューラルネットワークを利用した強化学習が進んでおり、また終盤の章の「ニューラルネットの PDE 数値解析への応用」はそのタイプの強化学習を用いているので、ここで紹介しておく
探索と行動の二律背反まるでデュエルマスターズの多色呪文（おそらく水、自然でコスト 3 くらい。ドローとマナ加速を選べて、色基盤にもなり状況に応じて使えるデッキの潤滑油的な存在なのだろう）のような名前だが、強化学習の用語である。
7. ボルツマンマシン
非常に名前がかっこいい機構である。ここまでは決定論的なニューラルネットワークだったが、ここでは確率的ニューラルネットワークについて論じていく。

ディープラーニングいよいよここから、第三次人工知能ブームを巻き起こしたとして巷で話題の深層学習に入り込んでいく。数学的なレベルは前章よりも大幅に上がるので、前章で知識不足を感じた方は関数解析や確率解析, 圏論の書籍を片手に読むことを勧める。

深層学習とは
前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

深層学習の積分表現の課題点
積分が入れ子構造になり非常にややこしい

デノイジング・オートエンコーダ（DAE）とは
まず、オートエンコーダについて解説する。ニューラルネットは往々にして「次元削減をしたうえで復元する機構」と言われる。次元削減する機構を「エンコーダ」、次元復元する機構を「デコーダ」という。
学習の前に、恒等写像を学習させる機構を「オートエンコーダ」という
すなわち、関数 $f : x \to x$ を近似する。デノイジングでは、この入力データにあえてノイズを付与して学習させることで、ノイズに対して頑健なニューラルネットワークを作ることができる。

まずは 3 層パーセプトロンについて考察する。入力データ $X \in \mathbb{R}^d$ は確率密度関数 $\pi_0(x)$ で定まる確率変数であるとする。ここに d 次元独立正規分布 (分散 t) の確率変数 ε を加える。

$$\tilde{X} := X + \varepsilon \tag{44}$$

そして、2 章のモデルに倣い、次の関数を最大化することを考えたい。（2 乗誤差最小化）

$$F(g) := -E[|X - g(\tilde{X})|^2] \tag{45}$$

中間層の写像 h と、出力層の写像 k を用いて、$g = h \circ k$ と書ける。この記法は後々深層学習を考えるにあたって非常に重要になる

定理：
上記の最適化問題の解は次の輸送写像になる（ジェームスシュタイン推定量？）

$$g(x) := x + t\nabla log[K(x, \tilde{x}, t/2) * \pi_0(x)] \tag{46}$$

ただし、K は熱核で、$K(x, y, t) := (4\pi t)^{-d/2} exp[-|x - y|^2/4t]$ となる。つまり、$K(\cdot, \cdot, t/2)$ で、時刻 t のブラウン運動の推移確率密度となる
実のところ、この畳み込みは偏微分作用素 $e^{t/2\Delta}$ の作用と同じである（要検証）
この輸送写像は、もっと拡張して楕円型作用素 L に対して定義できる

定義：
L を楕円型偏微分作用素。関数 $K_t(x, y; L)$ を、偏微分方程式 $\partial_t u(t, x - y) = Lu(t, x - y)$ の解とする。この

時、異方性 DAE を次のように定義できる

$$\begin{aligned}\Phi_t(x;L) :&= x + t\nabla log[e^{tL}\pi_0(x)] \\ &= x + t\nabla log[\int_{\mathbb{R}^d} K_t(x,y;L)\pi_0(y)dy]\end{aligned} \tag{47}$$

これを「ニューラルネットワークの輸送表現」という。つまり、三層パーセプトロンのデノイジングオートエンコーダーによる学習をこの関数で表現できる。

確率密度関数 π_0 から生成される確率測度を P_0 と置く（この確率測度は a.e で一意）。これがルベーグ測度に絶対連続なら、下記の式で定義される P_t もすべてルベーグ測度と絶対連続である。

$$P_t := P_0 \circ \Phi_t^{-1} \tag{48}$$

ここで、P_0 が正規分布、L がラプラシアンであるとき、連続 DAE に対する測度 μ（詳細は後述）は Wiener 測度である。
ちなみに、P_t のルベーグ測度とのラドンニコディム導関数を π_t と置くと、次の式が成り立つ

$$\partial_t \pi_t|_{t=0} = -\Delta\pi_0 \tag{49}$$

2. 深層デノイジング・オートエンコーダーについて
いよいよ深層学習を対象として論理展開をしていく。以後、使用する楕円型偏微分作用素はラプラシアンとし、L は隠れ層の数を表す整数であるものとする
終端時刻は T、これを L 分割し、時間の変分は τ_l（停止時刻とは無関係であることに注意）であらわされるとする。そして、現在時刻を $t_l(:= \tau_0 + \tau_1 + ... + \tau_{l-1})$ と表記する。各 DAE で加えられるノイズの確率分布は、$\mathcal{N}(0, \tau_l I)$ であるとする

そして、$l+1$ 番目の DAE で計算された写像を $\Phi_l : \mathbb{R}^d \to \mathbb{R}^d$ と表記する
ここで、最初の入力データを X_0 と置くと、これは初記の通り確率密度関数 π_0 に従う $\mathbb{R}^d$ 上の確率変数となる。ここで、確率測度 $P_{t_1} := P_0 \circ \Phi_0$ を定義し、これと同様に帰納的に P_{L+1} まで定義していく
また、次のように合成 DAE を定義する

$$\Phi_{0;L}^T := \Phi_L \circ ..... \circ \Phi_1 \circ \Phi_0 \tag{50}$$

これを用いて、次のように連続 DAE が定義できる
定義:連続 DAE
次の連続力学系の解作用素 ϕ_t を連続 DAE と呼ぶ

$$\begin{aligned}dX_t &= \nabla log\pi_t[X_t]dt \\ \text{ただし、}&\pi_t\text{は確率測度 } P_t := P_0 \circ \phi_t\end{aligned} \tag{51}$$

極限の存在と一意性は次の定理よりいえる

定理：
$log\pi_0$ は局所リプシッツ連続であるとする。このとき

$$\Phi_{0:L}^{T} \to \phi_T, L \to \infty \tag{52}$$

結局のところ、連続 DAE とは、標準ブラウン運動 B_t を用いて、$x + B_{T-t}$ と表記される逆向きブラウン運動に対する、x のジェームスシュタイン推定をニューラルネットワークに学習させる機構である。実際の深層 DAE はその離散化となる。
深層化する意味もここで説明できる。単純に DAE のサンプリングによるドリフト推定するにあたって、観測点を増やしたほうが精度が上がるに決まっている。それと同じことが DAE で起きているのである。

3. 積層 DAE と合成 DAE
ここからは、深層 DAE について圏論的・微分幾何学的に考察する

参考文献

http://www2.itc.kansai-u.ac.jp/ afujioka/2014/ig/141112ig.pdf https://datumstudio.jp/blog/
http://sinhrks.hatenablog.com/entry/2014/12/15/081113 実際の伊藤西尾の定理とはだいぶ形が違う。

実際には、「$[0,T]$ 上のカメロン-マルティン部分空間の正規直交基底列 h_k と、標準正規分布に従う確率変数列 X_k に対して、$S_n := \Sigma_{k=0}^{n} h_k X_k$ から誘導される測度が Wiener 測度に弱収束する」という定理である。
http://watanabe-www.math.dis.titech.ac.jp/users/swatanab/Bayestheory.html フィードフォワードニューラルネット http://ibisforest.org/index.php
残差学習のほうが近似できる関数のクラスが広がるのでは