NLP 勉強会資料

渡辺秀行

2025 年 1 月 22 日

1　機械学習の復習

1.1　機械学習とは

Wikipedia の定義：コンピュータに明示的に命令を与えずに学習する能力を与えること。これだと抽象的過ぎるので具体的にすると、

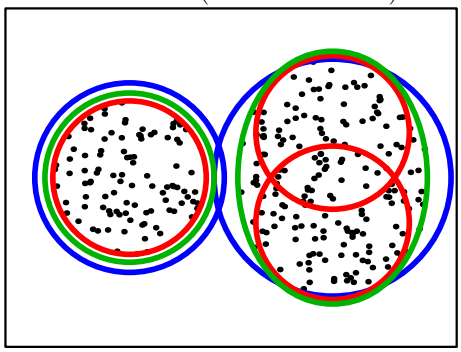
普通の機械学習[1]：既知のデータが持っている「規則性」「構造」を「発見」または「抽出」し (訓練:training/learning)、その結果 (仮説：hypothesis) を未知のデータに適用して予測 (predict/prediction)・推論 (infer/inference) するための計算技術、

となります。仮説 (hypothesis) は固定された部分であるモデル (model) と可変部分であるパラメータ (parameter) があります。

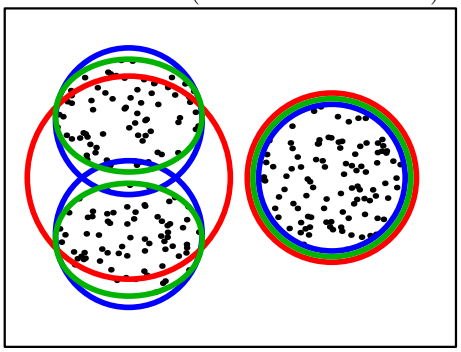
1.2　機械学習・モデルの分類

機械学習・モデルにはいろいろな分類の方法があります。

- 教師あり学習 (supervised learning) と教師なし学習 (unsupervised learning)
  正解のデータがある (教師あり) かない (教師なし) か?
- パラメトリックモデル (parametric model)、ノンパラメトリックモデル (non-parametric model)
  パラメータ数が固定 (パラメトリック) か、データに合わせて可変か (ノンパラメトリック)?



k-NN で $k = 3$ だと無理に分割、DPGMM だと綺麗に 2 分割



k-NN で $k = 2$ だと無理に結合、DPGMM だと綺麗に 3 分割

青：k-NN($k = 2$)、赤：k-NN($k = 3$)、緑：DPGMM

これはノンパラメトリックのイメージ図です。k-NN の境界は直線 (一般には超平面) です。

[1] この定義では「強化学習」などの一部の重要な機械学習をカバーしていません

- (通常の) モデル、ベイズモデル (Bayesian model)
  通常は何かの値を予測するときには 1 点を予測しますが、予測結果に幅を持たせる、すなわち予測結果自体が確率分布となるモデルもあり、ベイズモデルと呼ばれます (教師なし学習と相性が良い)。

- 識別モデル (discriminative model) と生成モデル (generative model)
  観測値から目的変数を分類、識別、予測するモデルを「識別モデル」と言い、目的変数の値から観測値をランダムに生成するモデルを「生成モデル」と言います。例えば、前者は「画像」から「犬か猫かを分類」、後者は「猫」と判定される画像を生成します。2 つのモデルを競争させる GAN(generative adversarial network) という技術もあります。

- 入力の種類 (モーダル： modal) 入力が数値か、テキストか、画像か、音声か、あるいは複数の入力を受けつけるか (マルチモーダル： multi modal)?

1.3 NLP における機械学習モデルの例

NLP における、機械学習モデルの例をいくつか挙げます。

- ベイジアンフィルター
  電子メールのテキスト (ヘッダを含むのが望ましい) から SPAM かどうかを判定。教師は人間の操作 (SPAM フォルダへの移動、SPAM フォルダからの復活)。メールが増えると語彙が増えていくのでノンパラメトリック。

- 商品の推薦 (recommendation)
  「この商品を買った人は、あの商品も買ってます」を出すモデル。教師なし。パラメトリック。入力は選択した商品の ID。

- トークナイズ (tokenize) もしくはサブワード区切 (subword segmentation)
  テキストをトークンに区切るモデル。未知語の問題を防ぐために単語より短く区切る (subword) のが現代では一般的。BPE、wordpiece、sentencepiece などの手法があり、言語モデルや翻訳モデルの入力に使われます。教師なし。点推定識別モデル。パラメトリック。

- 単語区切+タグ (品詞) 付け トークナイズでトークンを単語とする場合は、品詞を付けることが出来ます。

- 埋め込みモデル
  単語または文章を入力とし、ベクトルを出力します。言語モデルや翻訳モデルの入出力に使われます。また、文章の分類、マッチングにも使われます。教師なし (教師付きのファインチューニングあり)。点推定識別モデル。パラメトリック。

- 言語モデル
  文章の出現確率をモデルにします。文章の途中までの入力があれば、その続きの文章の出現確率がモデルされます。教師あり、生成モデル、パラメトリック、入力は部分テキスト。

- 翻訳モデル
  文章が与えられたとき、それに対応する文章の出現確率をモデルにします。教師あり、生成モデル、パラメトリック、入力は原文テキスト。

- 感情分類
  文章の感情を分類するモデル。教師あり、点推定識別モデル、パラメトリック、入力はテキスト。

- トピックモデル
  文章がいくつかのトピックの混合でなっているとして文書を分類するモデル。クラスタリングの一種と捉えることも可能。教師なし。点推定識別モデル、パラメトリック、入力はテキスト。

- ランキングモデル
  検索語に対して関連のある URL を順位付けするモデル。教師あり。教師なし。点推定識別モデル、パラメトリック、入力はテキスト。

その他重要な例を挙げます。

- クラスタリング
  「近いもの」同士をグループ化するモデル。教師なし。パラメトリック/ノンパラメトリック。

- PCA (主成分解析)
  次元削減の手法。線型性を持つモデル。

- EM(期待値、最大値) アルゴリズム
  教師なし学習の手法の一つ。トピックモデルや GMM(ガウス混合モデル：Gaussian mixture model) で用いられます。

- 変分ベイズ法
  EM アルゴリズムの Bayes 推定版。

- Auto-encoder：自己エンコーダ
  次元削減。PCA の非線形版。入力より次元の低い潜在変数 (latent variables) を通して元の入力と同じものを出力するように学習。入力→潜在変数の部分をエンコーダ、潜在変数→出力の部分をデコーダと呼びます。

- VAE (variational auto-encoder：変分自己エンコーダ)
  次元削減以外に、欠損値予測 (画像の欠損部分の復元) にも使えます。GAN と組み合わせたモデルも。自己エンコーダと流れは共通ですが、エンコーダ部分を Bayes 推定にし、デコーダ部分は推定からサンプリングを行って出力します。その為線型性を持たせることが出来て、潜在変数の空間で操作が可能になったり、生成の能力が上がったりします。

1.4 基本的な教師あり点推定識別モデル

ここでは、教師あり点推定識別モデルの基本的な 3 つのモデル・タスクを見ていきます。すなわち、回帰 (regression)、分類 (classification)、順位付け (ranking) です。

1.4.1 回帰

回帰とは連続値の出力を予測するモデル・タスクです。入力 $x \in \mathbb{R}^m$ に対して、仮説は $f(x) = y \in \mathbb{R}^n$ の形を取ります。すなわちベクトル値関数を予測することに相当します。最も単純な回帰問題は $m = n = 1$ のときの補間 (interpolation) です。図は 2 次曲線による補間の例です。入力 $x_i (i = 1, \ldots, N)$ における誤差は $|y_i - f(x_i)|$ で表わされますが、絶対値は微分可能 (differentiable) でないので誤差の自乗 (square) の平均が損失関数 (loss function) としてよく使われます。すなわち、

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - f(x_i))^2$$

の値が小さくなるように関数 $f(x)$ を決定することが学習となります。

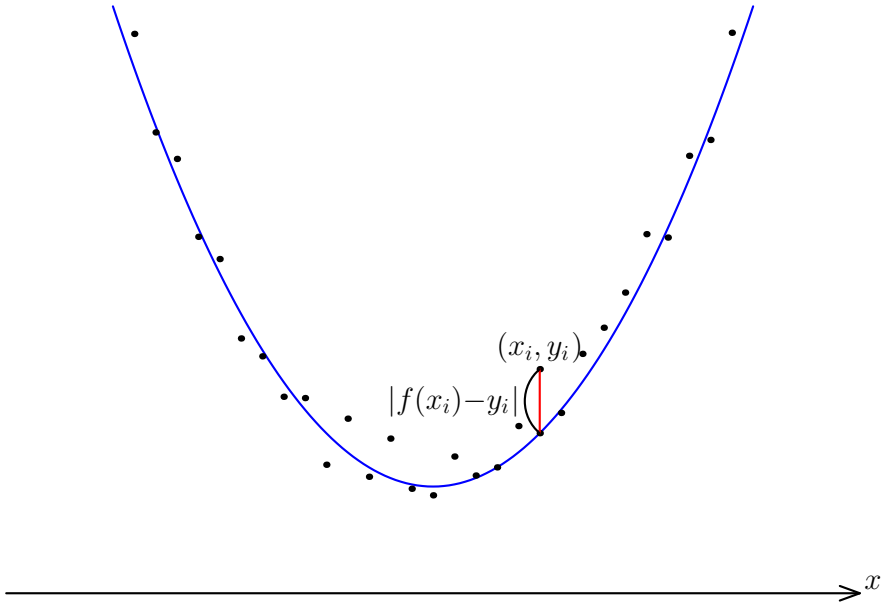


図 1 補間の例

1.4.2 分類

分類とは有限個の値の出力を予測するモデル・タスクです。まずは 2 値分類 (binary classification) について考えます。2 値分類はある分類 (class)C に属するか否かを予測するタスクです。これは入力 $x \in \mathbb{R}^m$ に対してモデルとなる関数 $f(x) = y \in \mathbb{R}$ が与えられたとき、仮説は関数 $h(x) = \sigma\ (f(x))$ を分類 C に属する確率と見做して予測を行います。確率が閾値 (threshold) より高ければ C に属するとし低ければ C に属さないとします。閾値は PR(precision と recall) 値を参照しながら決定されることが多いですが、単純に 0.5 とすることもあります。ここで

$$\sigma(x) := \frac{1}{1+e^{-x}} = \frac{1}{2}(1 + \tanh x)$$

はシグモイド (sigmoid) 関数と呼ばれ、実数 $\mathbb{R}$ を開区間 $(0, 1)$ に写像しています。

2 値分類の損失関数は、「予測の外れたデータの割合」とすると理想的なのですが、想像がつくようにこれだと微分可能となりません。そこで入力 $x_i (i = 1, \ldots, N)$ の正解の分類を y_i とすると

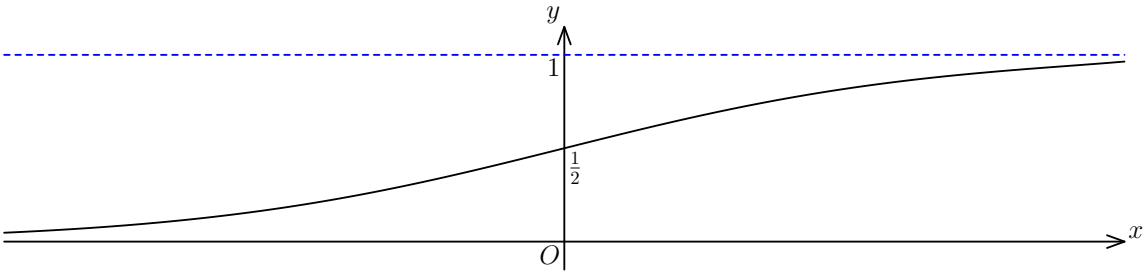


図 2 シグモイド関数

き、確率 $h(x_i)$ で C に属すると予測したと考えた場合、予測の当たる確率 $p(x_i)$ は次のようになります。

$$p(x_i) = \begin{cases} h(x_i) & (y_i = 1) \\ 1 - h(x_i) & (y_i = 0) \end{cases} = h(x_i)^{y_i}(1 - h(x_i)^{1-y_i}$$

その逆数を perplexity と呼び、その対数を交差エントロピー (cross entropy) と呼びます。N 個の点での交差エントロピーの平均

$$-\frac{1}{N}\sum_{i=1}^{N}\{y_i \log h(x_i) + (1 - y_i)\log(1 - h(x_i))\}$$

を損失関数[2]とします。

一般の n 値分類の場合、すなわち分類 C_1、C_2、...、C_n のどれに属するかを予測するモデルについても考え方は同様です。これは入力 $x \in \mathbb{R}^m$ に対してモデルとなる関数 $f(x) = y \in \mathbb{R}^n$ が与えられたとき、仮説は関数 $h(x) = \zeta(f(x))$ を各分類に属する確率として予測を行います。ここで $\zeta(x)$ は softmax 関数と呼ばれ、x の成分表示 $x = (x_1, x_2, \ldots, x_n)$ に対して

$$\zeta(x) := \frac{1}{e^{x_1} + e^{x_2} + \cdots + e^{x_n}}(e^{x_1}, e^{x_2}, \ldots, e^{x_n})$$

となります。シグモイド関数 $\sigma(x)$ は $\zeta(x, 0)$ の第 1 成分で表せますのでこれは 2 値分類のときの自然な拡張になっています。各 $x_i (i = 1, \ldots, N)$ の正解の分類を $y_i = (0, 0, \ldots, 1, \ldots, 0)$（正解の分類のところだけ 1）とします。また $h(x_i)$ および y_i の成分表示をそれぞれ $h(x_i) = (h(x_i)^{(1)}, h(x_i)^{(2)}, \ldots, h(x_i)^{(n)})$ および $y_i = (y_i^{(1)}, y_i^{(2)}, \ldots, y_i^{(n)})$ とします。右肩の数字に括弧が着いているのは巾乗ではなく、成分番号であることを強調しています。確率 $h(x_i)^{(j)}$ で分類 C_j に属すると予測したと考えた場合、予測の当たる確率 $p(x_i)$ は 2 値分類のときと同様に次のようになります。

$$p(x_i) = \prod_{j=1}^{n}\left(h(x_i)^{(j)}\right)^{y_i^{(j)}}$$

この値の逆数が perplexity[3]で、perpexity の対数が交差エントロピーです。交差エントロピーの平均を損失関数とします。

$$-\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i^{(j)} \log h(x_i)^{(j)}$$

[2]これを交差エントロピーと呼ぶこともあります。perplexity も同様に N 点での相乗平均の場合もあります。
[3]多値分類でも 2 値分類と同様、平均を perplexity や交差エントロピーと呼ぶ場合もあります

なお、分類器には確率ベクトルを出力としないものもあり、その場合は異なる損失関数が用いられます。例えば埋め込みモデルを triplet margin loss を損失関数としてファインチューニングすることが出来ます。

1.4.3 順位付け

順位付け (ranking) とは沢山のものから良いもの順に並べるモデル・タスクです。例えば検索エンジンのメインの仕事がそれに当ります。検索語に対して URL を最も関係の深い順に並べるからです。その並べるためのスコア付けが順位付けにおける仮説となります。順位付けの訓練は常に対 (pair) に対して行なわれます。検索エンジンの場合、2 つの URL のどちらがある検索語に対して優れているか、という教師データを用います。詳しくは”learning to rank”や”mart”, ”s-mart”で論文を当ってみると良いでしょう。

1.4.4 3 つのタスクの違い

それぞれのタスクの違いを考えます。仮説が完璧であれば、それは 3 つのタスクのどれに用いても完璧な仕事をします。しかし実際は完璧ということはないのでタスク毎に有効な仮説は異なります。例で考えてみましょう。6 つの商品 A、B、C、D、E、 Fがあり、それらに対するユーザの評価が 0 から 1 の値で与えられているとします。0.5 より大きければ良い、小さければ悪いとします。また、それを予測する 3 つの仮説 h_a, h_b, h_c があるとします。

商品	ユーザ評価	h_a	h_b	h_c
A	1.0	1.0	0.7	0.7
B	0.8	0.7	0.8	0.6
C	0.6	0.4	0.9	0.3
D	0.4	0.6	0.2	0.2
E	0.2	0.2	0.3	0.1
F	0.0	0.1	0.4	0.0

表 1 タスクの違い

このときに各仮説の性能を考えましょう。回帰を行いたい場合は、

$$\begin{aligned} \mathrm{loss}(h_a) &= \frac{1}{6}\left\{(1.0-1.0)^2+(0.8-0.7)^2+(0.6-0.4)^2+(0.4-0.6)^2+(0.2-0.2)^2+(0.1-0.1)^2\right\} \\ &= 0.017 \end{aligned}$$

同様に $\mathrm{loss}(h_b) = 0.065, \mathrm{loss}(h_c) = 0.045$ となり h_a が最も優れています。良い・悪いの分類を行ないたい場合には、外れの数が順に $2, 0, 1$ となり、h_b が最も優れています。順位の場合、誤りの量 (順位差の合計) [4]は順に $2, 8, 0$ となり、h_c が最も優れています。

[4]ランキングにおいては実際は順位差の合計ではなく NDCG と呼ばれる指標が使われます。NDCG でも h_c が優れています。

1.5　ニューラルネットワーク

実装が単純で能力が高いため良く使われます。様々なモデルの部品にもなっています。もともと人間の神経をモデルにしたからこういう名前が付いていますが、線型演算と非線形変換を交互に組み合わせたものです。線型演算の数を層 (layer) と呼びます。入力を $x = x_0 \in \mathbb{R}^m$ とし、l 層目の出力を $x_l \in \mathbb{R}^{m_l}$ とします。$l = 1, \dots, L-1$(最終層以外) の x_l を潜在変数 (latent variable) と呼びます。これは直前の層の潜在変数を用いて計算され、

$$x_l = f(W_l x_{l-1} + b_l)$$

となります。x_l, b_l は縦ベクトル ($1 \times m_l$ 行列) で、W_l は $m_l \times m_{l-1}$ 行列。f は各成分に同じ非線形関数を適用します。これを活性化 (activation) 関数、と呼びます。活性化関数として昔は tanh が使われていましたが、今では ReLU と呼ばれる関数 $f(x) = \max\{x, 0\}$ が使われます。[5]最終層では

$$y = W_L x_{L-1} + b_L$$

となります。ここで $n = m_L$ として $y \in \mathbb{R}^n$ となります。最終層では活性化関数は使わないことが多いです。例えば分類モデルとして使う場合は、最終層に対して softmax をとってクロスエントロピーを計算して損失関数とします。

一般に L が 5 以上の場合、深層学習 (deep learning) と呼ばれます。

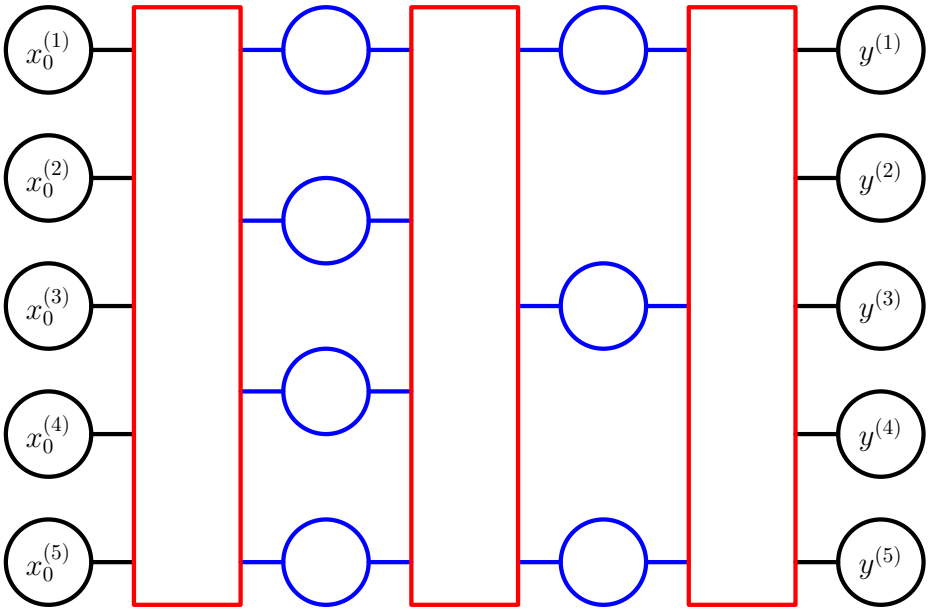


図 3 ニューラルネット：青が潜在変数、赤が訓練されるパラメータ (行列とバイアス)

1.6　モデルの訓練

モデルの訓練は損失関数を微分することで行います。モデルのパラメータを $p = (p_1, \dots, p_n)$ とします。入力変数と教師データの組を $(x_i, y_i)(i = 1, \dots, N)$ とし、1 点 (x_i, y_i) での損失を $L(x_i, y_i; p)$

[5]深層学習において傾きが 1 未満の場合、沢山掛けると小さくなる問題 (勾配消失問題) が生じます。ReLU だと傾きは 0 か 1 なのでこの問題が解決されます。

とすると、損失関数は

$$L(p) = \sum_{i=1}^{N} L(x_i, y_i; p)$$

という形で書けます。これを最小化する p を求めるのがモデルの訓練の目標です。p に関する勾配 (gradient) は、

$$\nabla_p L(p) = \left(\frac{\partial}{\partial p_1}, \ldots, \frac{\partial}{\partial p_n} \right) \sum_{i=1}^{N} L(x_i, y_i; p)$$

と書けます。すなわち p の付近で

$$L(p + p_\varepsilon) = L(p) + p_\varepsilon \cdot \nabla_p L(p) + o(|p_\varepsilon|)$$

と近似できるので、p を $-\nabla_p L(p)$ の向きに少し動かせば $L(p)$ が減ります。これを繰り返すことでモデルの訓練が出来ます。これを gradient descent(勾配降下法) と言います。

しかし、N が大きい場合に N 個の和を計算するのは大変なので、ランダムサンプルして勾配降下を繰り返す方法がとられます。これを SGD(stochastic gradient descent：確率的勾配降下法) と言います。さらに訓練の履歴を用いて収束を早める工夫した Adam, Adamax, Nadam, AMSGrad などのアルゴリズムもあります。[6]

問 手書の数字 (MNIST データ) を簡単なニューラルネットで認識させる例が置いてあります。興味があれば Python3.12 と各種ライブラリ (torch, torchvision, jupyter, notebook) をインストールして動かしてみて下さい。ニューラルネットの形を変えたり、オプティマイザのアルゴリズムを変えるなどやってみると何か分かるかもしれません。必要なライブラリはすべて持ち込んであります。GPU マシンにはこれらがインストール済です。

2　文章をトークン列として扱うモデル

ここでは文章はトークナイズ (tokenize：トークンの列に区切る) 済として考えます。文章を単語の集まり (BoW：Bag of Words) として扱うモデル (ベイジアンフィルターなど) やトークナイズそのものを扱うモデル、商品のレコメンデーションのモデルは扱いません。

2.1　埋め込みモデルとシーケンスモデル

[6] しかし、十分に時間をかけて訓練した SGD が最も成績が良い、というのが私の経験です。