scikit-learn を用いた機械学習入門ハンズオン ~回帰モデル編~

福嶋竜希

アジェンダ

- 回帰モデルによる機械学習の概論 (60分)

  フレームワークの説明

  特徴量

  モデル

  ハイパーパラメータチューニング

  モデルの評価方法

  回帰モデル

  2 乗和誤差関数

  線形回帰

  Ridge 回帰

  ニューラルネットワーク回帰

- 実装編(60分)

  ハンズオンを通して理解を深め、

  モデルの比較を行いましょう！

アジェンダ

- 回帰モデルによる機械学習の概論 (60分)

    フレームワークの説明
        特徴量
        モデル
        ハイパーパラメータチューニング
        モデルの評価方法

    回帰モデル
        2 乗和誤差関数
        線形回帰
        Ridge 回帰
        ニューラルネットワーク回帰

- 実装編(60分)

    ハンズオンを通して理解を深め、
    モデルの比較を行いましょう！

フレームワーク

- 機械学習は、与えられたデータから教師あり学習 (Supervised Learning) や教師なし学習 (Unsupervised Learning) などを行う。
- 本セミナーでは、教師あり学習、とりわけ、回帰というタスクに着目する。

特徴選択

生の入力変数から、遂行しようとしているタスクにとって重要な変数を選び出す、または(変数 1 を対数変換するなど)作り出すことをいう。

- このようにして得られた変数が実際の入力になり、これを特徴量と呼ぶ。
- 線形回帰における多重共線性など、無い方がよい変数も存在し得るので、特徴選択は機械学習において重要なタスクである。
- 一般に、特徴選択は非常に困難である。
(変数が夥しい数与えられることもあるから。)

フレームワーク

- 機械学習では、様々なモデルが提案されている。
- モデルは、入力を与えると、出力を返すもの。
- このセミナーでは、次の3つのモデルを扱う。

・線形回帰

・正則化回帰

・ニューラルネットワーク

- どのモデルで、どれほどの精度が出るかは問題に依存する。
- モデルは、パラメータを持つ。学習により最適な値が決まるパラメータと、人間が値を与えなくてはならないハイパーパラメータがある。
- ハイパーパラメータは、多くの場合、予め値の候補を用意しておき、それらで学習させてみて、最も精度が良いものを選ぶ。
- 値の候補の例: $\{2^{-10}, 2^{-9}, \dots, 2^{10}\}$ など。決まりはない。

フレームワーク

- 確率モデルでは、情報量基準。そうでない場合は誤差関数で評価することが多い。
- 本セミナーで扱うモデルはすべて、確率モデルでないため、誤差関数で評価。
- 誤差関数を説明するために必要な定義を述べておく。

訓練データ集合: (入力, 正しい出力)の組の集合

$$D = \{(\boldsymbol{x}_1, \boldsymbol{t}_1), (\boldsymbol{x}_2, \boldsymbol{t}_2), \dots, (\boldsymbol{x}_N, \boldsymbol{t}_N)\}$$

モデル: $y = f(\boldsymbol{x})$

- 回帰モデルの場合は、2乗和誤差関数または、2乗誤差の平均などが用いられる。
- 2乗和誤差関数:

$$E = \frac{1}{2}\sum_{n=1}^{N}(t_n - f(\boldsymbol{x}_n))^2$$

- 2乗誤差の平均:

$$E = \frac{1}{2N}\sum_{n=1}^{N}(t_n - f(\boldsymbol{x}_n))^2$$

- これらを最小化するように、モデルのパラメータは学習される。

フレームワーク

- モデルの選択(ハイパーパラメータ選択など)には、検証を行う。
- まず、与えられたすべてのデータを訓練データとテストデータに分ける。
- k-fold cross validation について述べる。

(i) 訓練データを(大体) k 等分し、それぞれの塊を $D_1, D_2, \ldots, D_k$ とする。

(ii) 候補のハイパーパラメータの値を1つ用いて、 $D_2, \ldots, D_k$ で学習を行い、D_1 を用いて誤差を測定する。

(iii) ハイパーパラメータの値は固定して、 D_2以外で学習を行い、D_2を用いて誤差を測定する。次は D_3 で誤差測定... を計 k 回繰り返す。

(iv) このハイパーパラメータの値を用いた時の精度は、

$$\frac{1}{k}\sum_{i=1}^{k}(D_i\text{での誤差})$$

で測定する。

(v) (iv) の精度が最も良いものを適切なハイパーパラメータとする。

線形回帰

- モデルは次の通り。

$$y = w_0 + w_1 x_1 + \ldots + w_M x_M = \boldsymbol{w}^T \boldsymbol{x}$$

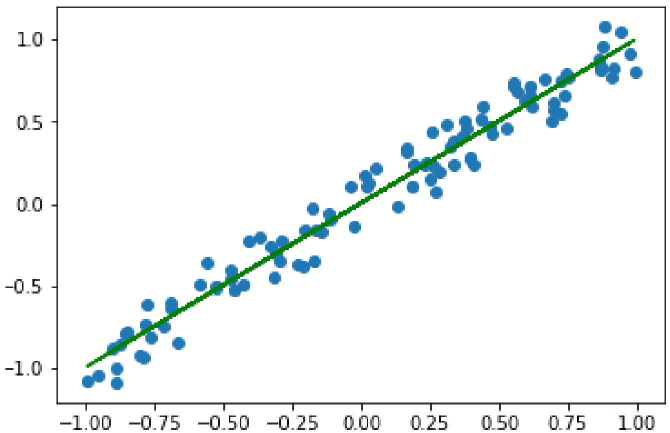
- このモデルの出力は y であり、正しい出力 t との誤差は、$t - y$ となる。
- 2乗和誤差関数

$$E = \sum_{n=1}^{N} (t_n - y_n)^2$$

を最小化すると、次の解が得られる。

$$\begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_N \end{bmatrix} = \left([\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \cdot \begin{bmatrix} \boldsymbol{x}_1^T \\ \vdots \\ \boldsymbol{x}_N^T \end{bmatrix} \right)^{-1} [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \begin{bmatrix} t_1 \\ \vdots \\ t_N \end{bmatrix}$$

- 右図のように、データによく当てはまる直線(超平面)を求めることで予測を行う。



正則化線型回帰

汎化能力

未知のデータに対して、精度よく予測を行う能力のこと。

- 機械学習の主なタスクは、予測であるから、汎化能力の向上が重要である。
- 先の線形回帰は、単純なモデルだが、解が求まらなかったり、
  (ただし sklearn では自動的にそれを回避するように設計されているようである)
  訓練データに当てはまりすぎて、予測精度が落ちる過学習を引き起こしやすい。
  (過学習は、線形回帰に限った話ではない。)
- こうした問題に対処する手法の1つが正則化である。
- 正則化を線形回帰に取り入れたものを正則化線形回帰というが、
  本セミナーでは、Ridge回帰を取り上げる。

正則化線型回帰

- Ridge回帰では、2乗和誤差関数にパラメータの2乗和を加える。つまり、

$$E_{Ridge} = \frac{1}{2}\sum_{n=1}^{N}(t_n - y_n)^2 + \frac{\lambda}{2}\sum_{i=0}^{M} w_i^2$$

  という誤差関数を最小化する。

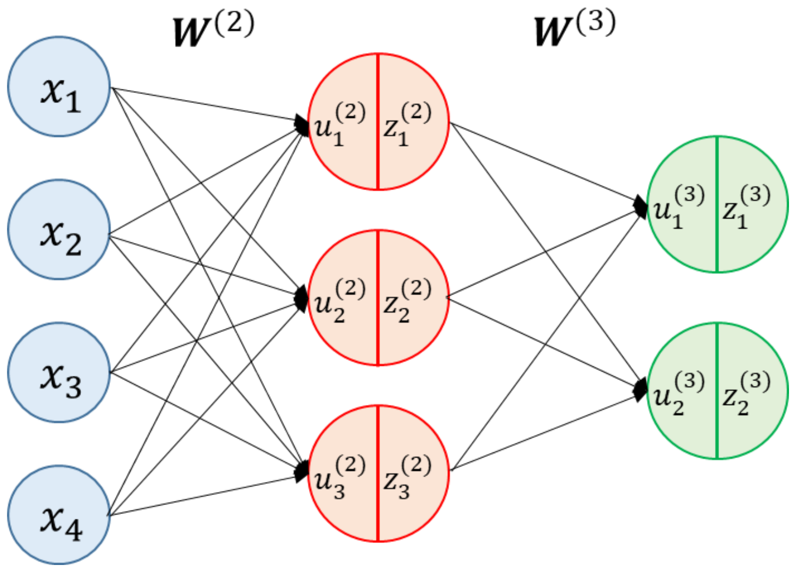
- これは、パラメータの値が大きくなりすぎないように学習を行うということ。

- λはハイパーパラメータであり、λが大きいほどパラメータが小さくなるように学習される。

- λ<0 となるように値を選んではいけない。

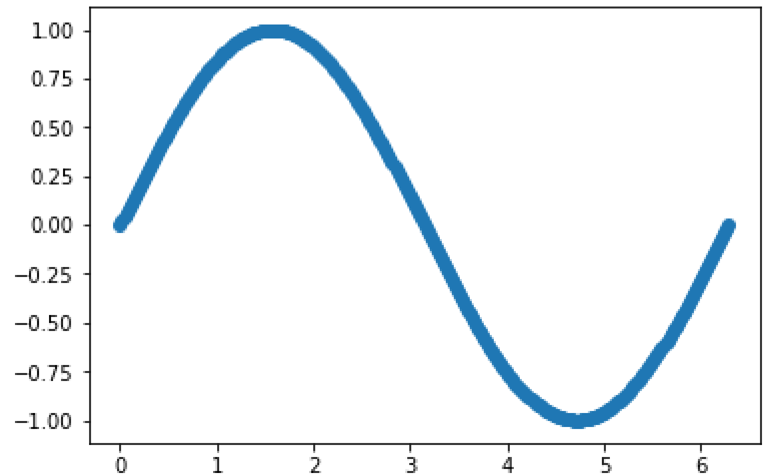
ニューラルネットワーク回帰

- ニューラルネットワークの構造は次の通り。



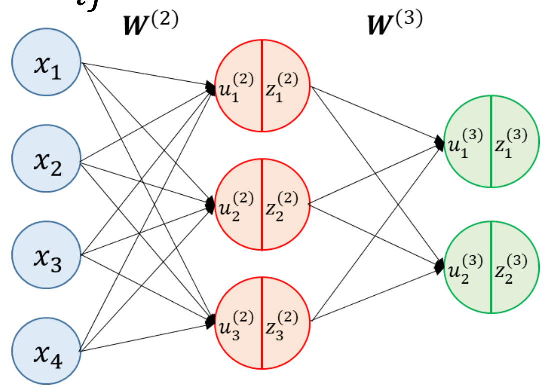
- 図のような複雑なデータでもしっかり対処できる。

- 線形回帰だと、図のようなデータに上手くあてはまる直線が無いので、精度が低い。



ニューラルネットワーク回帰

- 各ユニットの入出力を書き下してみる。
- 1つの重み(パラメータ)は$w_{ij}^{(l)}$と書くが、i が矢印の終点、j が始点とする。



$$\boxed{w_{To,\ From}^{(l)}}$$

$l = 2$

$for\ i\ in\ \{1,2,3\}$:

$$u_i^{(2)} = w_{i1}^{(2)} x_1 + w_{i2}^{(2)} x_2 + w_{i3}^{(2)} x_3 + w_{i4}^{(2)} x_4 + b_i^{(2)}$$
$$z_i^{(2)} = f(u_i^{(2)})$$

$l = 3$

$for\ k\ in\ \{1,2\}$:

$$u_k^{(3)} = w_{k1}^{(3)} z_k^{(2)} + w_{k2}^{(3)} z_k^{(2)} + w_{k3}^{(3)} z_k^{(2)} + b_k^{(3)}$$
$$y_k = z_k^{(3)} = f(u_k^{(3)}) \quad \text{(モデルの出力)}$$

まとめて、$\boldsymbol{y} = \boldsymbol{f}\left(\boldsymbol{u}^{(3)}\right) = \boldsymbol{f}\left(\boldsymbol{W}^{(3)}\boldsymbol{z}^{(2)} + \boldsymbol{b}^{(3)}\right) = \boldsymbol{f}\left(\boldsymbol{W}^{(3)}\boldsymbol{f}\left(\boldsymbol{W}^{(2)}\boldsymbol{x} + \boldsymbol{b}^{(2)}\right) + \boldsymbol{b}^{(3)}\right)$

一般に、$\boldsymbol{z}^{(l+1)} = \boldsymbol{f}\left(\boldsymbol{u}^{(l+1)}\right) = f\left(\boldsymbol{W}^{(l+1)}\boldsymbol{z}^{(l)} + \boldsymbol{b}^{(l+1)}\right)$