先端データ解析論レポート 第4回

荻野 聖也（37-196323）

2019年5月9日

宿題1

線形モデル

$$f_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sum_{j=1}^{b} \theta_j \phi_j(\boldsymbol{x})$$

に対する重み付き最小二乗法

$$\min_{\boldsymbol{\theta}} \frac{1}{2} \sum_{i=1}^{n} \tilde{w}_i (f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$$

について，

$$\begin{aligned}
\sum_{i=1}^{n} \tilde{w}_i (f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2 &= \tilde{w}_1 (f_{\boldsymbol{\theta}}(\boldsymbol{x}_1) - y_1)^2 + \tilde{w}_2 (f_{\boldsymbol{\theta}}(\boldsymbol{x}_2) - y_2)^2 + \ldots + \tilde{w}_n (f_{\boldsymbol{\theta}}(\boldsymbol{x}_n) - y_n)^2 \\
&= \tilde{w}_1 \left(\begin{bmatrix} \phi_1 & \ldots & \phi_b \end{bmatrix}_{\boldsymbol{x}=\boldsymbol{x}_1} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_b \end{bmatrix} - y_1 \right)^2 + \tilde{w}_2 \left(\begin{bmatrix} \phi_1 & \ldots & \phi_b \end{bmatrix}_{\boldsymbol{x}=\boldsymbol{x}_2} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_b \end{bmatrix} - y_2 \right)^2 \\
&\quad + \ldots + \tilde{w}_n \left(\begin{bmatrix} \phi_1 & \ldots & \phi_b \end{bmatrix}_{\boldsymbol{x}=\boldsymbol{x}_n} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_b \end{bmatrix} - y_n \right)^2
\end{aligned}$$

ここで，

$$Y_i \equiv \left(\begin{bmatrix} \phi_1 & \ldots & \phi_b \end{bmatrix}_{\boldsymbol{x}=\boldsymbol{x}_i} \begin{bmatrix} \theta_1 \\ \vdots \\ \theta_b \end{bmatrix} - y_i \right)$$

とすると，

$$\tilde{w}_1 Y_1^2 + \ldots + \tilde{w}_n Y_n^2 = \begin{bmatrix} Y_1 & \ldots & Y_n \end{bmatrix} \begin{bmatrix} \tilde{w}_1 & & 0 \\ & \ddots & \\ 0 & & \tilde{w}_n \end{bmatrix} \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}$$

なので，

$$\begin{aligned}
\sum_{i=1}^{n} \tilde{w}_i (f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2 &= (\boldsymbol{\Phi}\boldsymbol{\theta} - \boldsymbol{y})^T \tilde{\boldsymbol{W}} (\boldsymbol{\Phi}\boldsymbol{\theta} - \boldsymbol{y}) \\
&= (\boldsymbol{\theta}^T \boldsymbol{\Phi}^T - \boldsymbol{y}^T) \tilde{\boldsymbol{W}} (\boldsymbol{\Phi}\boldsymbol{\theta} - \boldsymbol{y}) \\
&= \boldsymbol{\theta}^T \boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{\Phi} \boldsymbol{\theta} - \boldsymbol{\theta}^T \boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{y} - \boldsymbol{y}^T \tilde{\boldsymbol{W}} \boldsymbol{\Phi} \boldsymbol{\theta} + \boldsymbol{y}^T \tilde{\boldsymbol{W}} \boldsymbol{y}
\end{aligned}$$

のように表すことができる。これより，

$$J(\boldsymbol{\theta}) \equiv \frac{1}{2} \sum_{i=1}^{n} \tilde{w}_i (f_{\boldsymbol{\theta}}(\boldsymbol{x}_i) - y_i)^2$$

にたいして，$\boldsymbol{\theta}$ に対して停留点をもとめる。

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{\Phi} \boldsymbol{\theta} - \boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{y}$$

なので，$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}=0$ に対して，

$$\hat{\boldsymbol{\theta}}=(\boldsymbol{\Phi}^T\tilde{\boldsymbol{W}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\tilde{\boldsymbol{W}}\boldsymbol{y}$$

となる。

宿題 2

損失 $\rho(r)$ は対称であることから，この 2 次上界は，

$$\tilde{\rho}(r)=ar^2+b$$

と表せる。ただし，a，b は定数。2 曲線 $\rho(r)$，$\tilde{\rho}(r)$ は点 $r=\tilde{r}$ で接するので，

$$\begin{aligned}&\left.\frac{d\tilde{\rho}}{dr}\right|_{r=\tilde{r}}=\left.\frac{d\rho}{dr}\right|_{r=\tilde{r}}\\&\Longleftrightarrow 2a\tilde{r}=\rho'(\tilde{r})\\&\Longleftrightarrow a=\frac{1}{2}\frac{\rho'(\tilde{r})}{\tilde{r}}\end{aligned}$$

となる。これより，2 次上界は，

$$\tilde{\rho}(r)=\frac{1}{2}\frac{\rho'(\tilde{r})}{\tilde{r}}r^2+b$$

これより，$\tilde{w}=\frac{\rho'(\tilde{r})}{\tilde{r}}$ とするとき，

$$\tilde{\rho}(r)=\frac{\tilde{w}}{2}r^2+const$$

となる。

宿題 3

テューキー損失 $\rho(r)$ は，

$$\rho(r)=\begin{cases}\frac{1-\left(1-\frac{r^2}{\eta^2}\right)^3}{6} & (|r|\leq\eta)\\ \frac{1}{6} & (|r|>\eta)\end{cases}$$

と表され，さらにこの時の重み w は，

$$w=\begin{cases}\left(1-\frac{r^2}{\eta^2}\right)^2 & (|r|\leq\eta)\\ 0 & (|r|>\eta)\end{cases} \tag{1}$$

と表される。

サンプルデータについて

講義内と同様に，直線モデル

$$f_{\boldsymbol{\theta}}(x)=\theta_1+\theta_2x$$

とする。つまり，基底関数 $\{\phi_j(x)\}_{j=1}^b$ は，

$$\begin{cases}\phi_1(x)=1\\ \phi_2(x)=x\end{cases}$$

さらに，パラメータ θ_1，θ_2 をそれぞれ，

$$\begin{cases}\theta_1=0\\ \theta_2=1\end{cases}$$

とする。サンプル数を 10 個とし，

$$y_i = f(x_i) + \varepsilon_i$$

と表されるものとする。この時，

$$-3 \leq x_i \leq 3$$

を満たし，ノイズ ε_i は正規分布

$$\varepsilon_i \sim N(0, \sigma^2 = 0.2^2)$$

に従うものとする。さらに，今回，ロバスト回帰であることから，外れ値を設ける必要がある。このことから，

$$y_2 = y_9 = y_{10} = -4$$

とする。以上の条件で生成した訓練標本を図 1 に示す。

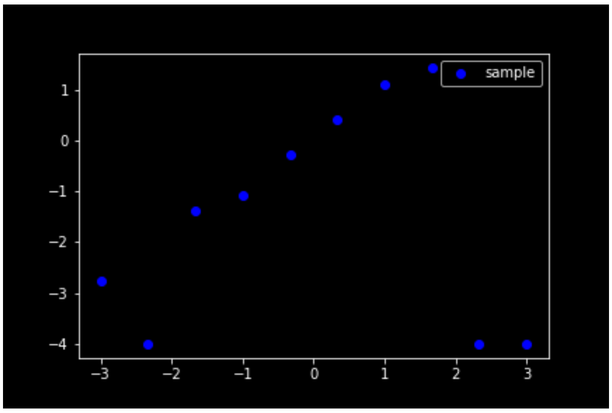

図 1　訓練サンプル. 今回，外れ値を数点設けた.

アルゴリズムについて

テューキー回帰の繰り返し最小二乗アルゴリズムを簡単に示す。基底関数については，1 次関数型の回帰であるので，

$$\begin{cases} \phi_1(x) = 1 \\ \phi_2(x) = x \end{cases}$$

である。

1. $\boldsymbol{\theta}$ を初期化する。
2. $\boldsymbol{\theta}$ に対して，

$$r_i = |f_{\boldsymbol{\theta}}(x_i) - y_i|$$

を求め，(1) から，$W = diag(w_1, \ldots, w_n)$ を用いて，

$$\boldsymbol{\theta} = (\boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T \tilde{\boldsymbol{W}} \boldsymbol{y}$$

により，$\boldsymbol{\theta}$ を更新する。

3. 解が習得するまで，2 を繰り返す。

結果

結果を以下の図 2 に示す。ただし，$\boldsymbol{\theta}$ の初期値は，

$$\boldsymbol{\theta}_{init} = (1.0, 1.0)$$

とした。図からもわかるように 3 点設けた外れ値の影響を受けず，うまく回帰できていることがわかる。なお，コードについては補填あるいは添付ファイルを参照してください。

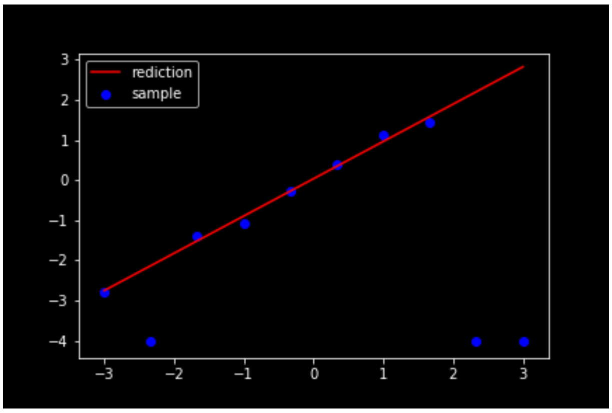

図 2　テューキー回帰を用いた直線モデルの予測結果. 外れ値の影響を受けていないことがわかる.

補填

今回宿題 3 で用いたソースコードを以下に示す。

Listing 1　宿題 3 のソースコード

```

## Requirement
import numpy as np
import matplotlib.pyplot as plt

np.random.seed(114514)

def get_sample(x_min=-3., x_max=3., sample_num=10, theta_1=0., theta_2=1.):サンプルを生成する。
    """講義資料にあるように

        y = theta_1 + theta_2 xから点を作る。この時，ノイズを生成させる。ただし，外れ値を作る必要があるので，数点ぶっ飛ばす。。。




    Arg:
        x_min(float) 座標の最小点 x default=-3.
        x_max(float) 座標の最大点 x default=3.
        sample_num(int) (x,y)の生成個数 default=10
        theta_1(float) theta_1(切片 y) defalut=0.
        theta_2(float) theta_2傾き () default=1.

    Return:
         X, Y(tuple(list(float),list(float))) サンプル (x,y)の
    """
    X = np.linspace(x_min, x_max, sample_num)
    # ノイズを載せて座標をサンプル y
    Y = theta_1 + theta_2*X + np.random.normal(loc=0, scale=0.2, size=sample_num)
    # 外れ値を設定
    Y[-1] = Y[-2] = Y[1] = -4
```

```
    return X, Y

def tukey_weight(r, eta):テューキー損失に対する重み
    """
    Arg:
        r(float) 予測値とサンプルの残差 y

    Return:
        w(float) 重み
    """
    if abs(r) <= eta:
        return (1 - r**2/eta**2)**2
    else:
        return 0

def get_basal(x):今回の基底関数の定義
    """
    Arg:
        x
    Return:
        phi_1, phi_2 = 1, x
    """
    return np.array([1, x])

def calc_tukey_regression(X, Y, theta_init=[1.,1.], eta=1., n_iter_max=1000):テューキー回帰を解く。
    """の初期値によっては，非凸性により悲しみを帯びるので注意。
    theta
    Arg:
        X(ndarray(float)) サンプルの座標 x
        Y(ndarray(float)) サンプルの座標 y
        theta_init(list[float, float]) の初期値←←重要 theta
        eta(float) 外れ値をテキトーに除外してくれるパラメータdefault=1.
        n_iter_max(int) イテレーションの最大数default=1000

    """
    n = len(X) サンプル数#
    b = len(get_basal(X[0])) 基底関数数#
    # 計画行列
    Phi_mat = np.empty((n, b))
    for row in range(n):
        Phi_mat[row] = get_basal(X[row])
     #の初期値 theta
    theta_vec = np.array(theta_init)
    # 以下繰り返し再重みづけ最小二乗
    for _ in range(n_iter_max):
        r = np.abs(np.dot(Phi_mat, theta_vec)-Y)対角成分を抽出
        #
        w_array = np.array([tukey_weight(r_i,eta) for r_i in r])
        W = np.diag(w_array)
        phit_w_phi = Phi_mat.T.dot(W).dot(Phi_mat)
```

```
        phit_w_y = Phi_mat.T.dot(W).dot(Y)
        theta_vec_pred = np.linalg.solve(phit_w_phi,phit_w_y)
        if np.linalg.norm(theta_vec_pred-theta_vec) < 1e-4:
            theta_vec = theta_vec_pred
            break
        else:
            theta_vec = theta_vec_pred
    return theta_vec

if __name__ == "__main__":
    X, Y = get_sample()
    theta_vec = calc_tukey_regression(X,Y)
    X_detail = np.linspace(-3., 3., 100)
    Y_detail = np.empty_like(X_detail)
    for i in range(len(X_detail)):
        Y_detail[i] = get_basal(X_detail[i]).dot(theta_vec)

    sample_filename = "sample.png"
    plt.scatter(X, Y, c="blue", marker="o", label="sample")
    plt.legend()
    plt.savefig(sample_filename)
    plt.show()

    filename = "tukey_output.png"
    plt.plot(X_detail, Y_detail, color="red", label="rediction")
    plt.scatter(X, Y, c="blue", marker="o", label="sample")
    plt.legend()
    plt.savefig(filename)
    plt.show()
```