なぜ Local Regression？

なぜ、Inverse PM で Local regression を使おうと思うのか？

俺の直感では、Inverse PM では、high-level indicator と PM parameter の関係は一般的に非線形であり、その関係を quadratic だとか、polynomial とかで表現できるとは限らない。そもそも、general なアプローチを提案したいので、quadratic だとか polynomial のような前提はしたくない。

Artificial Neural Network や Deep Learning は、そのような前提なしに、input と output の関係を学習できる。しかし、関係が非常にに複雑な場合は、新しい値に対する interpolation、extrapolation は、やはり難しい。※実際に deep leaning を実装して試す必要アリ。

俺の直感では、local regression が最も最適な solution ではないかと思う。非常にたくさんの sample を用意し、locally linear （もしくは、locally quadratic）を前提として output の推定を行う。非常に dense な sample があれば、local regression は一般的に効果的みたい。しかも、dense な sample があれば、どのような関係も表現できる。

つまり、感覚的には、deep learning と同様に表現力があり、しかも、deep learning よりも robust というイメージかな。

Local Regression

データ$X = \{x_1, x_2, \cdots, x_N\}$、そのアウトプット$Y = \{y_1, y_2, \cdots, y_N\}$が与えられた時、データ$x$のアウトプット$\mu(x)$を予測したい。**※アウトプットはスカラー値とする！**

Local linear regression の場合、各アウトプットを以下のように線形で表現する。

$$\mu(x_i) \approx a_0 + a_1(x_i - x) \tag{1}$$

式(1)を、多次元データに一般化すると、

$$\mu(x_i) \approx [(\boldsymbol{x_i} - \boldsymbol{x})^T \quad 1]\boldsymbol{a} \tag{2}$$

この時、以下の誤差を最小化するようなa_0、a_1を求める。

$$\sum_{i=1}^{N} W_i[y_i - \mu(x_i)]^2 \tag{3}$$

W_iは、距離$\|x_i - x\|$に基づいた重みだ。以下のようなガウス分布などが良く使われる。

$$W_i = \exp\left[-\frac{\|x_i - x\|^2}{2\sigma^2}\right]$$

結局、上の式は、xがx_iに近ければ近いほど、誤差の重みが大きくなる。つまり、これを最小化することで、xがx_iに近ければ、アウトプット$\mu(x)$はy_iに近づくわけだ。すごくリーズナブルなアイデアだね。

この時、予測される値は、式(1)に$x_i = x$を代入して、$\mu(x) = a_0$となる。

式(3)は、

$$W_1(y_1 - \mu_1)^2 + W_2(y_2 - \mu_2)^2 + \cdots + W_N(y_N - \mu_N)^2$$

と表せる。これって、例の「二次形式」を使うと、以下のように行列で表現できる。

$$[y_1 - \mu_1 \quad y_2 - \mu_2 \quad \cdots \quad y_N - \mu_N] \begin{bmatrix} W_1 & & & \\ & W_2 & & \\ & & \ddots & \\ & & & W_N \end{bmatrix} \begin{bmatrix} y_1 - \mu_1 \\ y_2 - \mu_2 \\ \vdots \\ y_N - \mu_N \end{bmatrix}$$

つまり、

$$(\boldsymbol{y} - \boldsymbol{\mu})^T W (\boldsymbol{y} - \boldsymbol{\mu}) \tag{4}$$

ただし、

$$\mu_i = [(\boldsymbol{x}_i - \boldsymbol{x})^T \quad 1]\boldsymbol{a}$$

よって、

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_N \end{bmatrix} = \begin{bmatrix} [(\boldsymbol{x}_1 - \boldsymbol{x})^T & 1]\boldsymbol{a} \\ \vdots & \\ [(\boldsymbol{x}_N - \boldsymbol{x})^T & 1]\boldsymbol{a} \end{bmatrix} = X\boldsymbol{a}$$

ただし、

$$X = \begin{bmatrix} (\boldsymbol{x}_1 - \boldsymbol{x})^T & 1 \\ \vdots & \vdots \\ (\boldsymbol{x}_N - \boldsymbol{x})^T & 1 \end{bmatrix} \tag{5}$$

従って、式(4)は、

$$(\boldsymbol{y} - X\boldsymbol{a})^T W (\boldsymbol{y} - X\boldsymbol{a}) \tag{6}$$

つまり、weighted least squares は、式(6)の形で表現できる。

この解は、以下のようになるらしい（なんで？？）

$$\boldsymbol{a} = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} = (X^T W X)^{-1} X^T W Y \tag{7}$$

■まとめ：

ある点$\boldsymbol{x}$が与えられた時、重み行列Wを計算し、式(5)で行列Xを作成し、式(7)を使って$\boldsymbol{a}$を求めたら、その最後の要素（a_1）が、予想されるアウトプットだ。

うまくいけば、こんな結果が得られるはず。

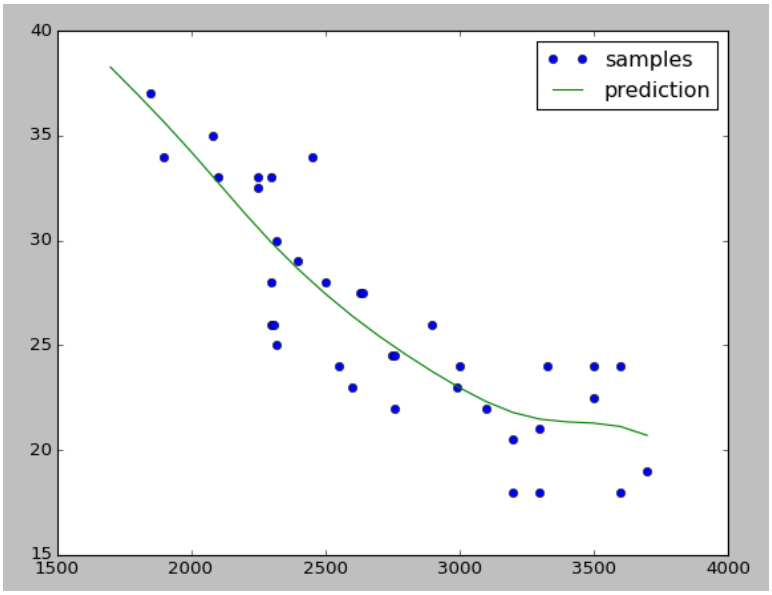

Figure 1. ガウスカーネルを使用した場合の、local regression の結果。$\boldsymbol{\sigma = 250}$を使用した。

多次元アウトプット

ここまでの議論では、アウトプットはスカラー値と見なしてきた。つまり、

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

多次元アウトプットの場合は、式(7)で、Yの列数が 1 から複数に増えるだけ。しかも、Yは、式(7)の最後に出てくるだけなので、Yの列数が増えても、そのまま OK ってことだ！つまり、

$$A = (X^T W X)^{-1} X^T W Y$$

でAを求めたら、Aの最後の行が答えだ！