The Evalution of One Variable Nonparamteric Linear Regression

Yihong Gu
Department of Computer Science
Tsinghua University
gyh15@mails.tsinghua.edu.cn

Abstract

In this paper, we proposed an one variable nonparamteric linear regression method called 'the k-nearest neighbourhood estimate'. We firstly carefully define the model and describe our method. Afterwards we use the simple case and some calculus to demonstrate its large sample properties. Then we proposed the hybrid determination of hyperparamter h using both the large sample property as well as pratical method called 'search and fix'. We then design the experiments, especially in data generation. We expose the estimator to different kind of pseudo-data designed above and generated by R and anaylse its performance using L^2 error. Using the results of experiments, we discuss its performance and use some sample plots to illustrate them.

1 Model

We evaluate the performance of nonparametric estimator for One Variable Linear Model. We consider the simplest case, in which we are given the data set $\{(x_i, y_i)\}_{i=1}^n$, where the x_i's and y_i's are all real number in $\mathbb{R}^1$, the probabilistic model is

$$Y_i = m(X_i) + \epsilon_i \tag{1}$$

where $i \in \{1, 2, \cdots, n\}$, Y_i, ϵ_i are all random variables, and X_i is a constant when i is fixed. At the same time, $m(\cdot)$ is the **unknown function** in $\mathcal{C}^0(\mathbb{R})$ we want to estimate. We call X_i **explanatory variable** and call Y_i **response variable**, while ϵ_i is a random error that can't be measured exactly.

We regard this model as a discriminant model instead of a generative one (although in reality the explanatory variable X might has distribution itself but for simplicity we ignore it).

In the perspective of expectation, we add more constraints on the model

$$\begin{aligned} \mathbb{E}[Y_i|X = X_i] &= m(X_i) + \mathbb{E}[\epsilon_i|X = X_i] \quad (2) \\ &= m(X_i) \quad (3) \end{aligned}$$

Here, we assume $\mathbb{E}[\epsilon_i|X = X_i] = 0$ for any X_i and ϵ_i are i.i.d.

For summary, we define our model through 4 steps:

- (constant) explanatory variable: X_i
- unknown (but truly exsits) function: $m(\cdot) \in \mathcal{C}^0(\mathbb{R})$
- i.i.d. noise random variable: $\epsilon_1, \cdots, \epsilon_n$
- response variable: $Y_i = m(X_i) + \epsilon_i$

When the model is clearly defined, we are given the data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$, and use $\mathcal{D}$ to estimate the unknown function $m(\cdot)$.

2 Method

We use the classic method call '**the k-nearest neighbourhood estimate**' to estimate the unknown function $m(\cdot)$.

2.1 Algorithm

In the subsection, we describe how can we estimate the particular value of Y given X.

RLab homework for course Linear Regression, Spring 17, Tsinghua University, Student ID: 2015011249
The code can be found in Github: https://github.com/wmyw96/LinearRegression-Spring17/

If we are given $X = x$ and want to estimate $m(x)$, we re-sorted all the data set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ according to x and get the sorted data set $\mathcal{D}(x) = \{(x_{(i)}(x), y_{(i)}(x))\}_{i=1}^n$, which satisfies the following condition:

$$|x_{(i)}(x) - x| \leq |x_{(i+1)}(x) - x| \tag{4}$$

since the order might not be unqiue, if $|x_{(i)}(x) - x| = |x_{(i+1)}(x) - x|$, it must also satisty $x_{(i)}(x) \leq x_{(i+1)}(x)$

we define the estimate:

$$\hat{m}_{n,h}(x) = \frac{1}{2h+1} \sum_{i=1}^{2h+1} y_{(i)}(x) \tag{5}$$

2.2 Large Samples Property

We consider the large sample property of our estimator. Considering the most simple case, we assume $x_1, \cdots, x_n$ satisty $x_1 \leq x_2 \leq \cdots \leq x_n$, $x_1 = a$, $x_n = b$ and $x_i - x_{i-1} = \frac{b-a}{n}$, i.e. $x_1, \cdots, x_n$ is point of n-section of interval $[a, b]$.

we also assume $x_0 = x_k$, where k satisfy $h + 1 \leq k \leq n - h$, which means that the point is not so 'skewed', we re-write the estimator [5] as followings:

$$\hat{m}_{n,h}(x_0) = \frac{1}{2h+1} \sum_{i=1}^{2h+1} m(x_{(i)}(x_0)) + \epsilon_{(i)}(x_0)$$

then subtract the true value from it, we get

$$\hat{m}_{n,h}(x_0) \quad = \quad \frac{1}{2h+1} \sum_{i=1}^{2h+1} m(x_{(i)}(x_0)) - m(x_0) + \mathrm{N}(x_0)$$

we define term $\mathrm{N}(x_0) = \frac{1}{2h+1} \sum_{i=1}^{2h+1} \epsilon_{(i)}(x_0)$, using talyor expansion and some obvious approximation, we get

$$\begin{aligned} \hat{m}_{n,h}(x_0) &\approx \frac{1}{2h+1} (\frac{b-a}{n-1}^2)^2 m''(x_0) h^3 C + \mathrm{N}(x_0) \\ &\approx C' \frac{h^2}{n^2} + N(x_0) \end{aligned} \tag{6}$$

where C and C' are both constants we don't care, the variance of the estimate is

$$\mathrm{var}(N(x_0)) = \frac{\sigma^2}{2n+1} \tag{7}$$

where σ^2 is the variance of the noise. At the same time, the bias of the estimator is

$$\text{bias} \quad \approx \quad C' \frac{h^2}{n^2} \tag{8}$$

using the approximation [7] and [8], we can write the MSE of the esitmator as followings:

$$\text{MSE} \sim C_1 (\frac{h}{n})^4 + C_2 \frac{1}{h} \tag{9}$$

to determine the best h, (using the basic inequation) we get $h = Cn^{4/5}$, so

$$\text{MSE} \sim C' \frac{1}{n^{4/5}} \tag{10}$$

2.3 Determine hyperparameter h

In the previous subsection, we discuss the property of the estimator and claim that in order to get the best MSE error, we should make h satisfy $h = Cn^{4/5}$. However, C is unknown and is according to the true function $m(\cdot)$ and x_0, so it's impossible to find the value of C analytically. Therefore, we use 'search and fix' method to determine the parameter C, then determine the hyperparamter h according to the equation $h = [Cn^{4/5}]$.

Confronted with particular data set, we use search C in $\{0.01n + 0.1\}_{n=0}^{290}$ and use few samples (repeat $T = 10$ times) to get the optimal C^* which minimize the loss function [15]. When C^* is fixed, we use C^* to do the following evaluation process and get the results.

Actually, the method above well determine the hypyerparamter h when x_0 is in the 'middle' of the data x, but works quite poorly when $x_0 \to a$ and $x_0 \to b$. We re-determine the actual h^* in the following method:

$$h^*(x_0) \quad = \quad \min(h_l(x_0), h_r(x_0), h) \tag{11}$$

$$h_l(x_0) \quad = \quad \sum_{i=1}^{n} 1_{x_i < x_0} \tag{12}$$

$$h_l(x_0) \quad = \quad \sum_{i=1}^{n} 1_{x_i > x_0} \tag{13}$$

3 Evaluation

3.1 Data Variants

Following the setting of our model, we perform our experiment in the following steps:

1. We use two true function $m_1(\cdot), m_2(\cdot)$ and use the true function to generate pseudo data, and then use the pseudo data and different estimators to estimate the parameters, we perform the following steps separately for each function, the 2 functions are as followings:

- $m_1(x) = x\sin(x)$
- $m_2(x) = \log(1 + (1/x))$

2. We set the data size $n = 30, 100, 1000$, and see how the estimators performed under diffenent scales of data.

3. We design $\{x_i\}_{i=1}^n$. We can generate x randomly or fixedly, here we consider two classic methods:

- Let $x_i \sim$ i.i.d $\mathcal{U}[0, 1]$
- Make x_i have same distance, i.e. let $x_i = \frac{i}{n}$.

4. Set the noise, we also consider 2 major settings:

- $\epsilon_1, \cdots, \epsilon_n \sim$ i.i.d. $\mathcal{N}(0, \sigma^2)$, where $\sigma = 0.1$
- $\epsilon_1, \cdots, \epsilon_n \sim$ i.i.d. t_1

5. We use the model $Y_i = m(X_i) + \epsilon_i$ and the generated value $\{x_i\}_{i=1}^n$ and $\{\epsilon_i\}_{i=1}^n$ to calculate $\{y_i\}_{i=1}^n$, so here now we have the data $\{(x_i, y_i)\}_{i=1}^n$.

6. We use the data and the estimator [5] and see how they vary from the true function

We repeat [2]$\sim$[6] T times separately for $m_1(\cdot)$ and $m_2(\cdot)$ and generate $\hat{m}(\cdot)$, using the loss function

$$\mathcal{L} = \int_0^1 |\hat{m}(x) - m(x)|^2 dx \tag{14}$$

to estimate the performance of the estimator, actually, it can't be analytically integrated, so we approximate if using the following approximation:

$$\mathcal{L} \approx \sum_{i=1}^{100} \frac{1}{100} |\hat{m}(\frac{i}{100}) - m(\frac{i}{100})|^2 \tag{15}$$

3.2 Implemention Details

We fixed the random seed to be 123469 and use R-package 'ggplot2' to generate plots and 'xtable' to directly convert data.frame in R to table format in Latex.

4 Experiments and Results

4.1 $m_1(x) = x\sin(x)$

We first plot performance of estimators versus the hyperparamter C in Figure 1

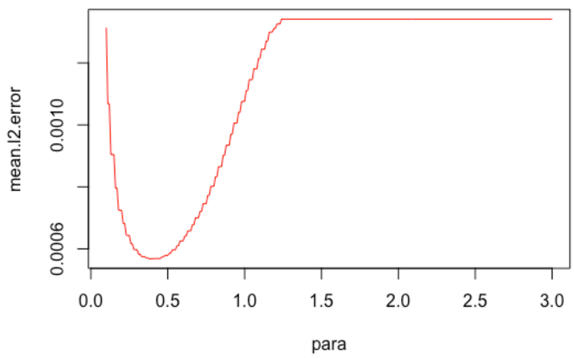

Figure 1: L^2 error of different hyperparameters C

we found that the optimial C might be approximately about 0.45 in the condition $T = 10, n = 100$, normal noise, the plot of other conditions are similar.

Normal Noise

The summary results are as followings:

Table 1: Summary reuslts of Normal Noise, $m(x) = x\sin(x)$

n	F_X	C	$\mathcal{L}$
30	Fixed	0.43	0.001503
30	Uniform	0.30	0.003098
100	Fixed	0.39	0.000220
100	Uniform	0.27	0.001112
1000	Fixed	0.43	0.000139
1000	Uniform	0.38	0.000284

From the data we can draw some conclusions:

1. Fixed distribution X perform much better than Uniform distributed X according to the value of $\mathcal{L}$, and its advanage over Uniform distributed X might be reduced when n becoming large due to the large sample property.

2. The $\mathcal{L}$ might subject to large sample property: when n becomes larger, $\mathcal{L}$ might become smaller.

3. The best C might be different for different conditions because we want to minimize $\mathcal{L}$ instead of bias^2.

Because fixed distribution don't have siginificant differences over uniform distribution, so we only give some examples of fixed distribution in Figure [2], [3] and [4].

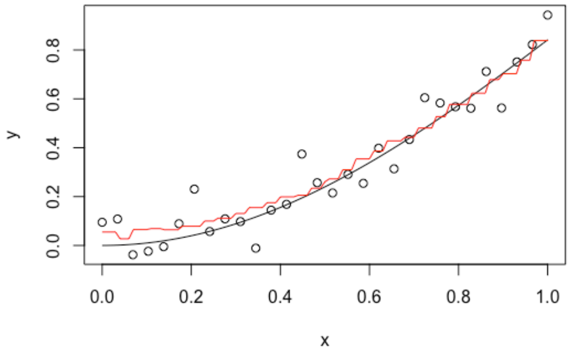

Figure 2: Sample: m_1, normal noise, fixed distributed X, $n = 30$

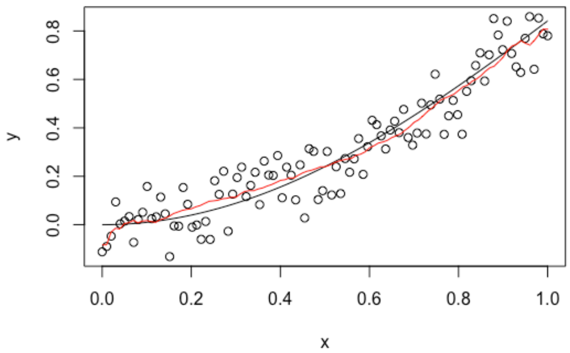

Figure 3: Sample: m_1, normal noise, fixed distributed X, $n = 100$

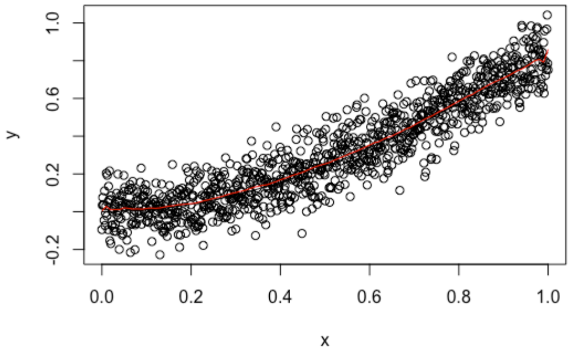

Figure 4: Sample: m_1, normal noise, fixed distributed X, $n = 1000$

From the figures above, we see that when n is small, the curve might be like step-like function and when n is large, the curve might be approximately equal to the true function.

Student's t Noise

The summary results are as followings:

Table 2: Summary reuslts of T Noise, $m(x) = x\sin(x)$

n	F_X	C	$\mathcal{L}$
30	Fixed	2.00	1211.892
30	Uniform	0.63	1568.185
100	Fixed	2.12	4014.363
100	Uniform	0.24	10587.75
1000	Fixed	1.94	3741.85
1000	Uniform	1.98	24833.35

From the data we can draw some conclusions:

1. Fixed distribution X perform much better than Uniform distributed X according to the value of $\mathcal{L}$, and its advanage over Uniform distributed X might be enhanced when n becoming large due to the non-existence of variance of noise.

2. The $\mathcal{L}$ might not subject to large sample property: when n becomes larger, $\mathcal{L}$ might become larger due to the non-existence of variance of noise.

3. The best C might be different for different conditions. Moreover, we can see that when X is fixed, the best C indicated that $2h + 1$ might be larger (or equal) than n, it might imply that the estimator tend to average all the y due to the extreme instability of the noise.

We can see a simple example in Figure [5]

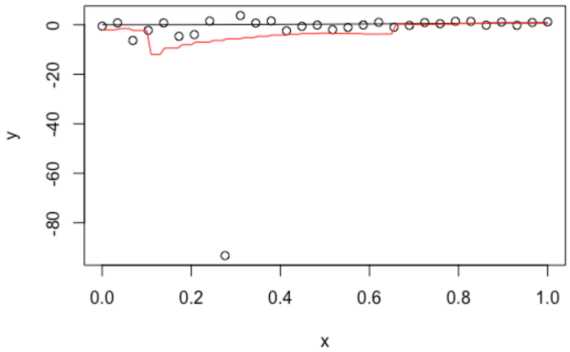

Figure 5: Sample: m_1, t noise, fixed distributed X, $n = 30$

the outlier make the prediction perform so poorly

4.2 $m_2(x) = \log(1 + 1/x)$

Normal Noise

We first plot performance of estimators versus the hyperparamter C in Figure 6

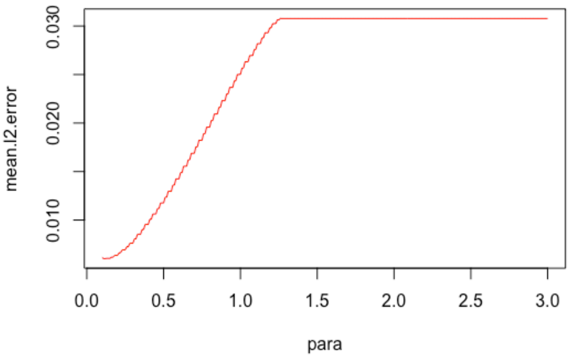

Figure 6: L^2 error of different hyperparameters C, m_2

we found that the optimial C might be approximately about 0.10 in the condition $T = 10, n = 100$, normal noise, the plot of other conditions are similar.

Moreover, from the plot we can see that the smaller C is, the better the performance is, that might because it will capture the approximation when $x \to 0$.

From the data we can draw some conclusions:

1. Fixed distribution X perform much better than Uniform distributed X according to the value of $\mathcal{L}$, and its advanage over Uniform distributed X might be reduced when n becoming large due to the large sample property.

Table 3: Summary reuslts of Normal Noise, $m(x) = \log(1 + 1/x)$

n	F_X	C	$\mathcal{L}$
30	Fixed	0.26	0.0224016
30	Uniform	0.12	0.047907
100	Fixed	0.12	0.005969
100	Uniform	0.07	0.011058
1000	Fixed	0.10	0.001595
1000	Fixed	0.03	0.000828
1000	Uniform	0.03	0.001405

2. The $\mathcal{L}$ might subject to large sample property: when n becomes larger, $\mathcal{L}$ might become smaller.

3. The best C might be different for different conditions because we want to minimize $\mathcal{L}$ instead of bias^2. Moreover, The best C will approximate 0 when n becomes larger.

4. The performance of estimator for m_2 is a little bit worse than that for m_1(might because the non-convergence of function around 0), but it also perform well.

Because fixed distribution don't have siginificant differences over uniform distribution, so we only give some examples of fixed distribution in Figure [7], [8] and [9].

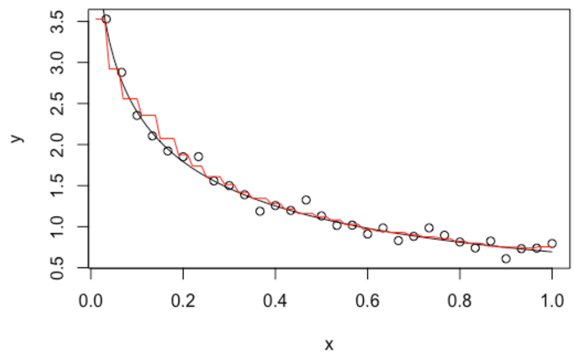

Figure 7: Sample: m_2, normal noise, fixed distributed X, $n = 30$

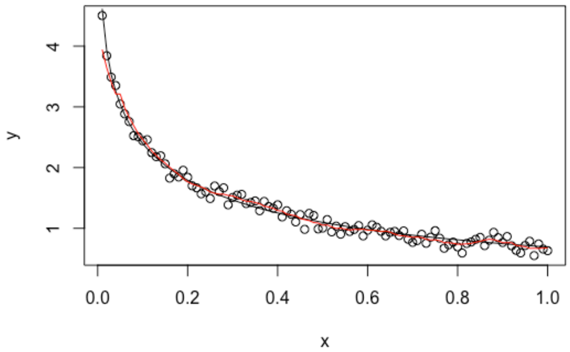

Figure 8: Sample: m_2, normal noise, fixed distributed X, $n = 100$

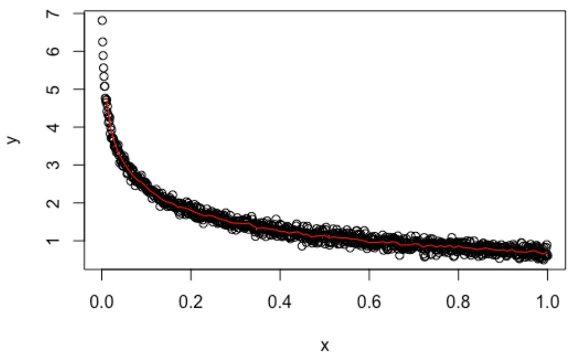

Figure 9: Sample: m_2, normal noise, fixed distributed X, $n = 1000$

Compare Figure [9] with Figure [10], which uses $C = 0.10$, we found the curve of latter figure is smoother than that of the former.

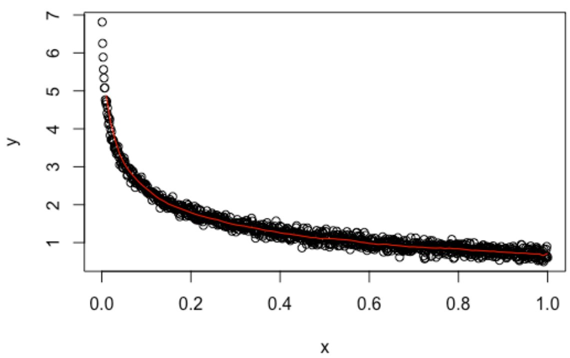

Figure 10: Sample: m_2, normal noise, fixed distributed X, $n = 1000$, with hyperparamter $C = 0.1$

Student's t Noise

The summary results are as followings:

Table 4: Summary reuslts of T Noise, $m(x) = \log(1 + 1/x)$

n	F_X	C	$\mathcal{L}$
30	Fixed	0.67	2202.761
30	Uniform	0.63	1566.381
100	Fixed	2.02	3545.114
100	Uniform	0.24	10583.19
1000	Fixed	1.97	3529.085
1000	Uniform	1.98	24833.23

From the data we can draw some conclusions:

1. Fixed distribution X perform much better than Uniform distributed X according to the value of $\mathcal{L}$, and its advanage over Uniform distributed X might be enhanced when n becoming large due to the non-existence of variance of noise.

2. The $\mathcal{L}$ might not subject to large sample property: when n becomes larger, $\mathcal{L}$ might become larger due to the non-existence of variance of noise.

3. The best C might be different for different conditions. Moreover, we can see that when X is fixed, the best C indicated that $2h + 1$ might be larger (or equal) than n, it might imply that the estimator tend to average all the y due to the extreme instability of the noise.

4. We can see that the results of m_2, Uniformly distributed X is approximately same with that of m_1, Uniformly distributed X. It might because that the variance dominate the loss error.

We can see a simple example in Figure [11]

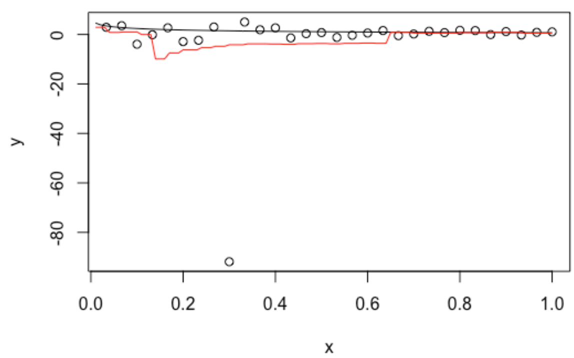

Figure 11: Sample: m_2, t noise, fixed distributed X, $n = 30$

the outlier make the prediction perform so poorly, too.