CSCI567 Machine Learning (Fall 2017)

Prof. Fei Sha

U of Southern California

Lecture on Oct. 19, 2017

Outline

Outline

Schedule change

- Quiz 2 is to be moved to 11/2. But please wait for the official announcement.
- Next Tuesday's lecture is dedicated to regrading. Please come for either the standard solution if you cannot make discussion section or regrading.
- DEN students will need to schedule regrading sessions separately with TAs.

Outline

Formal definition of Naive Bayes

General case

Given a random variable $X \in \mathbb{R}^{\mathsf{D}}$ and a dependent variable $Y \in [\mathsf{C}]$, the Naive Bayes model defines the joint distribution

$$P(X = x, Y = y) = P(Y = y)P(X = x|Y = y) \quad (1)$$

$$= P(Y = y)\prod_{d=1}^{\mathsf{D}} P(X_d = x_d|Y = y) \quad (2)$$

Special case (i.e., our model of spam emails)

Assumptions

- All X_d are categorical variables from the same domain — $x_d \in [\mathsf{K}]$, for example, the index to the unique words in a dictionary.
- $P(X_d = x_d | Y = y)$ depends only on the value of x_d, not d itself, namely, orders are not important (thus, we only need to count).

Simplified definition

$$P(X = x, Y = c) = P(Y = c) \prod_k P(k|Y = c)^{z_k} = \pi_c \prod_k \theta_{ck}^{z_k}$$

where z_k is the number of times k in x.

Note that we only need to enumerate in the product, the index to the x_d's possible values. On the previous slide, however, we enumerate over d as we do not have the assumption there that order is not important.

Learning problem

Training data

$$\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{\mathsf{N}} \rightarrow \mathcal{D} = \{(\{z_{nk}\}_{k=1}^{\mathsf{K}}, y_n)\}_{n=1}^{\mathsf{N}}$$

Goal

Learn $\pi_c, c = 1, 2, \cdots, \mathsf{C}$, and $\theta_{ck}, \forall c \in [\mathsf{C}], k \in [\mathsf{K}]$ under the constraint

$$\sum_c \pi_c = 1$$

and

$$\sum_k \theta_{ck} = \sum_k P(k|Y = c) = 1$$

as well as those quantities should be nonnegative.

Estimating $\{\pi_c\}$

We want to maximize

$$\sum_c \log \pi_c \times (\#\text{of data points labeled as c})$$

Intuition

- Similar to roll a dice (or flip a coin): each side of the dice shows up with a probability of π_c (total C sides)
- And we have total N trials of rolling this dice

Solution

$$\pi_c^* = \frac{\#\text{of data points labeled as c}}{\text{N}}$$

Estimating $\{\theta_{ck}, k = 1, 2, \cdots, \mathsf{K}\}$

We want to maximize

$$\sum_{n:y_n=c,k} z_{nk} \log \theta_{ck}$$

Intuition

- Similar to roll a dice with color c: each side of the dice shows up with a probability of θ_{ck} (total K slides)
- And we have total $\sum_{n:y_n=c,k} z_{nk}$ trials.

Solution

$$\theta^*_{ck} = \frac{\text{\#of side-k shows up in data points labeled as c}}{\text{\#of all slides in data points labeled as c}}$$

Classification rule

Given an unlabeled data point $x = \{z_k, k = 1, 2, \cdots, \mathsf{K}\}$, label it with

$$
\begin{aligned}
y^* &= \arg\max\nolimits_{c \in [\mathsf{C}]} P(y = c|x) && (3)\\
&= \arg\max\nolimits_{c \in [\mathsf{C}]} P(y = c)P(x|y = c) && (4)\\
&= \arg\max\nolimits_{c} [\log \pi_c + \sum_k z_k \log \theta_{ck}] && (5)
\end{aligned}
$$

Naive Bayes is a linear classifier

Fundamentally, what really matters in deciding decision boundary is

$$w_0 + \sum_k z_k w_k$$

This is the same as logistic regression's decision boundary. However, we estimate *parameters* differently.

Difference and similarity: have you filled the blank yet?

	Logistic regression	Naive Bayes
Similar	Linear classifier	Linear classifier
Difference	?	?

Outline

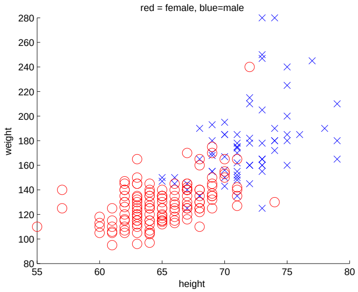
Naive Bayes and logistic regression: two different modeling paradigms

- Setup of the learning problem
  Suppose the training data is from an *unknown* joint probabilistic model $p(\boldsymbol{x}, y)$
- Differences in *assuming* models for the data
  - the generative approach requires we specify the model for the joint distribution (such as Naive Bayes), and thus, maximize the *joint* likelihood $\sum_n \log p(\boldsymbol{x}_n, y_n)$
  - the discriminative approach (discriminative) requires only specifying a model for the conditional distribution (such as logistic regression), and thus, maximize the *conditional* likelihood $\sum_n \log p(y_n|\boldsymbol{x}_n)$

Naive Bayes and logistic regression: two different modeling paradigms

- Setup of the learning problem
  Suppose the training data is from an *unknown* joint probabilistic model $p(\boldsymbol{x}, y)$
- Differences in *assuming* models for the data
  - the generative approach requires we specify the model for the joint distribution (such as Naive Bayes), and thus, maximize the *joint* likelihood $\sum_n \log p(\boldsymbol{x}_n, y_n)$
  - the discriminative approach (discriminative) requires only specifying a model for the conditional distribution (such as logistic regression), and thus, maximize the *conditional* likelihood $\sum_n \log p(y_n|\boldsymbol{x}_n)$
- Differences in computation
  - Sometimes, modeling by discriminative approach is easier
  - Sometimes, parameter estimation by generative approach is easier

Determining sex (man or woman) based on measurements

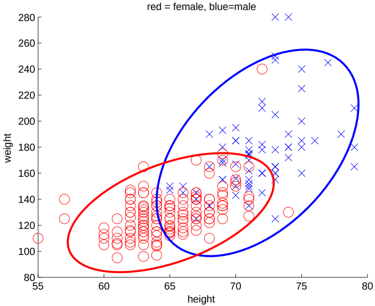


Generative approach

Propose a model of the joint distribution of ($x =$ height, $y =$sex)

our data

Sex	Height
1	$6'$
2	$5'2"$
1	$5'6"$
1	$6'2"$
2	$5.7"$
…	…



Intuition: we will model how heights vary (according to a Gaussian) in each sub-population (male and female).

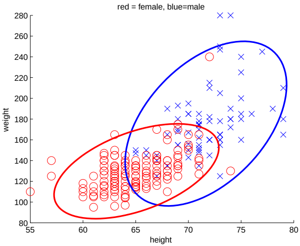
Note: This is similar to Naive Bayes for detecting spam emails.

Model of the joint distribution

$$p(x, y) = p(y)p(x|y) \tag{6}$$

$$= \begin{cases} p_1 \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} & \text{if } y = 1 \\ p_2 \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} & \text{if } y = 2 \end{cases} \tag{7}$$

where $p_1 + p_2 = 1$ represents two *prior* probabilities that x is given the label 1 or 2 respectively. $p(x|y)$ is called *class distributions*, which we have assumed to be Gaussians.



Parameter estimation

Likelihood of the training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$ with $y_n \in \{1, 2\}$

$$
\begin{aligned}
\log P(\mathcal{D}) &= \sum_n \log p(x_n, y_n) \\
&= \sum_{n:y_n=1} \log\left(p_1 \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x_n-\mu_1)^2}{2\sigma_1^2}}\right) \\
&+ \sum_{n:y_n=2} \log\left(p_2 \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x_n-\mu_2)^2}{2\sigma_2^2}}\right)
\end{aligned}
$$

Parameter estimation

Likelihood of the training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ with $y_n \in \{1, 2\}$

$$\begin{aligned}
\log P(\mathcal{D}) &= \sum_n \log p(x_n, y_n) \\
&= \sum_{n:y_n=1} \log \left(p_1 \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(x_n-\mu_1)^2}{2\sigma_1^2}} \right) \\
&+ \sum_{n:y_n=2} \log \left(p_2 \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(x_n-\mu_2)^2}{2\sigma_2^2}} \right)
\end{aligned}$$

Maximize the likelihood function

$$(p_1^*, p_2^*, \mu_1^*, \mu_2^*, \sigma_1^*, \sigma_2^*) = \arg\max \log P(\mathcal{D})$$

Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

$$p(y = 1|x) \geq p(y = 2|x)$$

which is equivalent to

$$p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2)$$

Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

$$p(y = 1|x) \geq p(y = 2|x)$$

which is equivalent to

$$p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2)$$

Namely,

$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

$$p(y = 1|x) \geq p(y = 2|x)$$

which is equivalent to

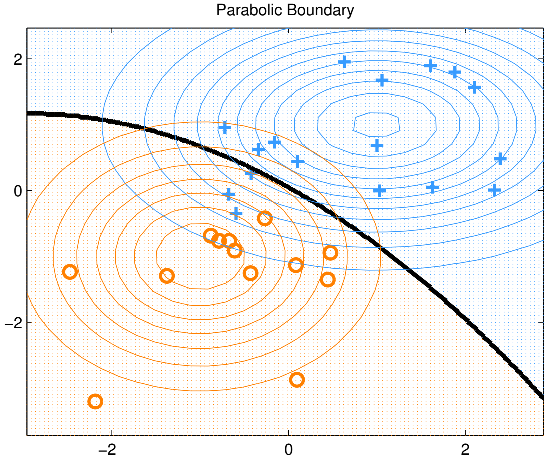
$$p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2)$$

Namely,

$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

$\Rightarrow ax^2 + bx + c \geq 0$ $\quad\leftarrow$ the decision boundary not *linear*!

Example of nonlinear decision boundary



Note: the boundary is characterized by a quadratic function, giving rise to the shape of parabolic curve.

A special case: what if we assume the two Gaussians have the same variance?

We will get a linear decision boundary

$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

with $\sigma_1 = \sigma_2$, we have

$$bx + c \geq 0$$

A special case: what if we assume the two Gaussians have the same variance?

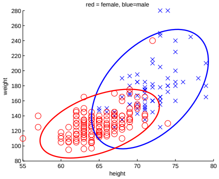
We will get a linear decision boundary

$$-\frac{(x-\mu_1)^2}{2\sigma_1^2} - \log\sqrt{2\pi}\sigma_1 + \log p_1 \geq -\frac{(x-\mu_2)^2}{2\sigma_2^2} - \log\sqrt{2\pi}\sigma_2 + \log p_2$$

with $\sigma_1 = \sigma_2$, we have

$$bx + c \geq 0$$

Note: equal variances across two different categories could be a very strong assumption.



For example, from the plot, it does seem that the *male* population has slightly bigger variance (i.e., bigger eclipse) than the *female* population. So the assumption might not be applicable.

Mini-summary

Gaussian discriminant analysis

- A generative approach, assuming the data modeled by

$$p(x, y) = p(y)p(x|y)$$

  where $p(x|y)$ is a Gaussian distribution.
- Parameters (of those Gaussian distributions) are estimated by maximizing the likelihood
  - Computationally, estimating those parameters are very easy — it amounts to computing sample mean vectors and covariance matrices
- Decision boundary
  - In general, nonlinear functions of x — in this case, we call the approach *quadratic discriminant analysis*
  - In the special case we assume equal variance of the Gaussian distributions, we get a linear decision boundary — we call the approach *linear discriminant analysis*

So what is the discriminative counterpart?

Intuition

The decision boundary in Gaussian discriminant analysis is

$$ax^2 + bx + c = 0$$

Let us model the conditional distribution analogously

$$p(y|x) = \sigma[ax^2 + bx + c] = \frac{1}{1 + e^{-(ax^2+bx+c)}}$$

Or, even simpler, going after the decision boundary of linear discriminant analysis

$$p(y|x) = \sigma[bx + c]$$

Both look very similar to logistic regression — i.e. we focus on writing down the *conditional* probability, *not* the joint probability.

Does this change how we estimate the parameters?

First change: a smaller number of parameters to estimate

Our models are only parameterized by a, b and c. There is no prior probabilities (p_1, p_2) or Gaussian distribution parameters (μ_1, μ_2, σ_1 and σ_2).

Second change: we need to maximize the conditional likelihood $p(y|x)$

$$
\begin{aligned}
(a^*, b^*, c^*) = \arg\min -\sum_n \big\{ & y_n \log \sigma(ax_n^2 + bx_n + c) && (8)\\
& + (1 - y_n) \log[1 - \sigma(ax_n^2 + bx_n + c)] \big\} && (9)
\end{aligned}
$$

Computationally, much harder!

How easy for our Gaussian discriminant analysis?

Example

$$p_1 = \frac{\text{\# of training samples in class 1}}{\text{\# of training samples}} \tag{10}$$

$$\mu_1 = \frac{\sum_{n:y_n=1} x_n}{\text{\# of training samples in class 1}} \tag{11}$$

$$\sigma_1^2 = \frac{\sum_{n:y_n=1} (x_n - \mu_1)^2}{\text{\# of training samples in class 1}} \tag{12}$$

Note: detailed derivation is in the books. They can be generalized rather easily to multi-variate distributions as well as multiple classes.

Generative versus discriminative: which one to use?

There is no fixed rule

- Selecting which type of method to use is dataset/task specific
- It depends on how well your modeling assumption fits the data
- Recent trend: big data is always useful for both!
    - Apply very complex discriminative models, such as deep learning methods, for building classifiers
    - Apply very complex generative models, such as nonparametric Bayesian methods, for modeling data

Outline

Motivating example

Suppose we have a sequence of real-valued observation

$$\mathcal{D} = x_1, x_2, x_3, \cdots, x_N$$

drawn from an *unknown* distribution

$$p(x)$$

How do we estimate what is $p(x)$?

First solution

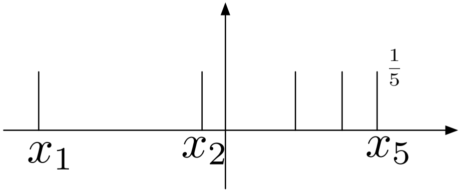
How about the following distribution

$$\hat{p}(x) = \frac{1}{N} \sum_{n=1}^{N} \delta(x - x_n)$$

where the $\delta(\cdot)$ is the Dirac function

$$\delta(z) = 1 \text{ if and only if } z = 0$$

The problem: what does $\hat{p}(x)$ look like?



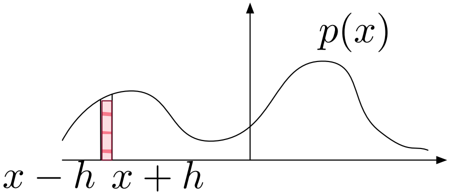
This does not seem good as

$$\hat{p}(x) = 0$$

for any x that is not in the training set!

A better way

Assume our probability density function $p(x)$ is smooth



Then what is the probability a data point falling into the range of $[x-h\ x+h]$ if h is small?
This is

$$P(x' \in [x-h\ x+h]) = \int_{x-h}^{x+h} p(x)dx \approx 2hp(x)$$

where we assume that h is so small such that $p(x)$ is near constant in this interval.

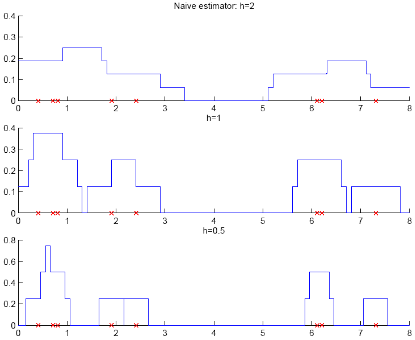
What is a good estimate of this probability?

$$P(x' \in [x-h \;\; x+h]) = \frac{\#\text{training data samples } \in [x-h \;\; x+h]}{N}$$

Thus, we can approximate

$$p(x) \approx \hat{p}(x) = \frac{\#\text{training data samples } \in [x-h \;\; x+h]}{2hN}$$

Naive/Silverman Density Estimator



Fundamentally, we are just computing histogram to count the number of points falling different bins, decided by *bin width h*.

What is the problem? We still have zero probability event despite that we think $p(x)$ is smooth. (*Note. Image borrowed from here* `https://www.cmpe.boun.edu.tr/~ethem/i2ml3e/`)

Understand this estimator better

Our naive estimator

$$\hat{p}(x) = \frac{1}{2Nh} \#\text{training data samples} \;\in [x-h \;\; x+h]$$

We can rewrite it as

$$\hat{p}(x) = \frac{1}{Nh} \sum_{n=1}^{N} K\left(\frac{x - x_n}{h}\right)$$

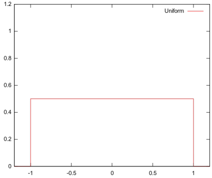
where the *kernel* $K(\cdot)$ is defined as

$$K(u) = \begin{cases} \frac{1}{2} & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases}$$

Note that this kernel is *not* the kernel we have seen before in kernel methods (though they do have connections).

This kernel function is just a weight

$$K(u) = \begin{cases} \frac{1}{2} & \text{if } |u| < 1 \\ 0 & \text{otherwise} \end{cases}$$



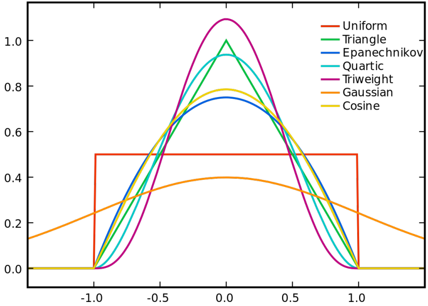
We can see it as a weighted sum of different data points towards x

$$\hat{p}(x) = \sum_{n=1}^{N} \frac{1}{h} K\left(\frac{x - x_n}{h}\right) \frac{1}{N} = \sum_{n=1}^{N} K_h(x - x_n) \frac{1}{N}$$

where $\frac{1}{N}$ can be seen as the probability of data sample x_n.

In other words, our estimator is just a weighted average of training samples' empirical probability. (K_h is called scaled kernel function)

We can use different kernels



The only requirement is

$$\int_{-\infty}^{+\infty} K(u)du = 1, K(u) = K(-u)$$

This type of method is called *Parzen Window* method.

Example

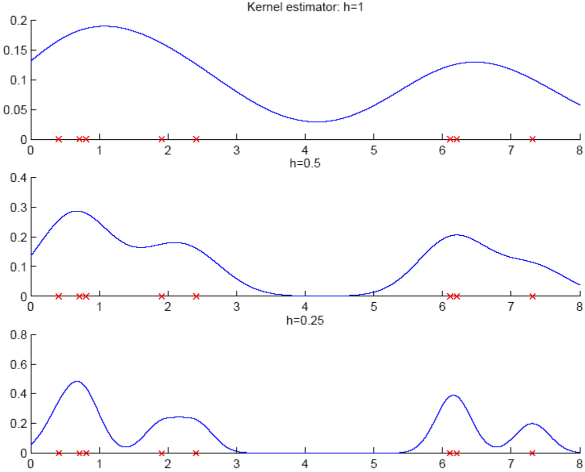
Gaussian kernel

$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$$

The corresponding estimator is

$$\hat{p}(x) = \frac{1}{Nh} \sum_{n=1}^{N} \frac{1}{\sqrt{2\pi}} e^{-\frac{(x-x_n)^2}{2h^2}}$$

Effect of h


Kernel estimator: h=1
0.2
0.15
0.1
0.05
0
0 1 2 3 4 5 6 7 8
h=0.5
0.4
0.3
0.2
0.1
0
0 1 2 3 4 5 6 7 8
h=0.25
0.8
0.6
0.4
0.2
0
0 1 2 3 4 5 6 7 8

8

Choosing optimal h is not easy

We can use cross-validation

- On which dataset?
- Measure what kind of performance metric?

There are several theoretically-motivated ways of choosing the bandwidth

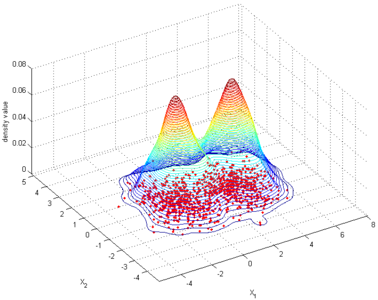
Please check out the wikipedia page
`https://en.wikipedia.org/wiki/Kernel_density_estimation`
as well as free implementation of this method.

Extension to multivariate distribution

Example of using Gaussian kernel

$$\hat{p}(\boldsymbol{x}) = \frac{1}{N}\sum_{n=1}^{n}\frac{1}{\sqrt{(2\pi)^D|\boldsymbol{\Sigma}|}}e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{x}_n)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{x}_n)}$$

Applying to a two-mixture Gaussian distribution



Please see the wikipedia page `https://en.wikipedia.org/wiki/Multivariate_kernel_density_estimation` for more details and sample codes (you do not need to read about how the optimal bandwidth matrix is selected.)

Applications

- Outlier detection
  Please read the following paper https://link.springer.com/chapter/10.1007/978-3-540-73499-4_6 . This is considered supplementary reading material and is not required.
- Nonparametric regression

Nonparametric regression

Consider the supervised learning problem for regression, we are given the training data

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \cdots, (\boldsymbol{x}_N, y_N)\}$$

How to estimate the corresponding value of y for arbitrary $\boldsymbol{x}$?

We will see how kernel density estimator can be helpful

Probabilistic models

Let us start with the joint model

$$p(\boldsymbol{x}, y) = p(y|\boldsymbol{x})p(\boldsymbol{x})$$

We have a way to estimate $p(\boldsymbol{x})$ from the kernel density estimator.

But we do not know the joint probability $p(\boldsymbol{x}, y)$. We are making the following *modeling assumption*

$$p(\boldsymbol{x}, y) = \sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n) K_{h'}(y - y_n)$$

What is the optimal value assign to $\boldsymbol{x}$ then?

It turns out to be the expectation of (reminiscent of Bayes optimal classifier?)

$$p(y|\boldsymbol{x})$$

Thus, we need to compute this conditional probability

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}, y)}{p(\boldsymbol{x})} = \frac{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n) K_{h'}(y - y_n)}{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n)}$$

Now that we can compute its expectation

$$\mathbb{E}[p(y|\boldsymbol{x})] = \frac{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n) \mathbb{E}[K_{h'}(y - y_n)]}{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n)} = \frac{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n) y_n}{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n)}$$

why the last step is true? This has left as a take-home exercise. (Hint: please check the properties of the kernel)

Nadaraya – Watson (kernel) regression

Given $\boldsymbol{x}$ and training dataset $\mathcal{D}$ we predict

$$y = \frac{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n) y_n}{\sum_n K_h(\boldsymbol{x} - \boldsymbol{x}_n)}$$

Note that this is a non-parametric method.

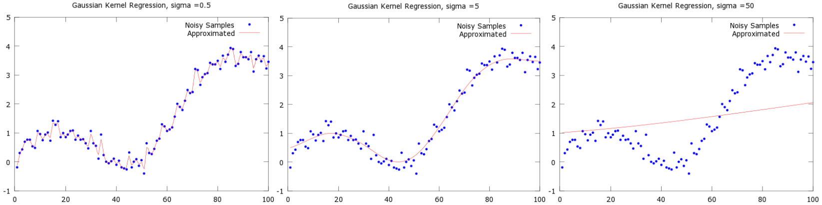
It is easy to see this is a weighted average

$$y = \sum_n \frac{K_h(\boldsymbol{x} - \boldsymbol{x}_n)}{\sum_{n'} K_h(\boldsymbol{x} - \boldsymbol{x}_{n'})} y_n = \sum_n w(x, x_n) y_n$$

with

$$\sum_n w(x, x_n) = 1$$

Examples



See `http://mccormickml.com/2014/02/26/kernel-regression/` for details and demo codes.