

Eight more Classic Machine Learning algorithms

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials:
<http://www.cs.cmu.edu/~awm/tutorials>.
 Comments and corrections gratefully received.

Andrew W. Moore
Associate Professor
School of Computer Science
Carnegie Mellon University
www.cs.cmu.edu/~awm
awm@cs.cmu.edu
 412-268-7599

Copyright © 2001, Andrew W. Moore

Oct 29th, 2001

8: Polynomial Regression

So far we've mainly been dealing with linear regression

X_1	X_2	Y
3	2	7
1	1	3
:	:	:

X	Y
3	7
1	3
:	:

$y_1 = 7..$

Z	y
1	7
3	3
2	:
1	:
1	:
:	:

$z_1 = (1, 3, 2).. \quad y_1 = 7..$
 $z_k = (1, x_{k1}, x_{k2})$

$$b = (Z^T Z)^{-1} (Z^T y)$$

$$y^{est} = b_0 + b_1 X_1 + b_2 X_2$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 2

Quadratic Regression

It's trivial to do linear fits of fixed nonlinear basis functions

X_1	X_2	Y
3	2	7
1	1	3
⋮	⋮	⋮

X	Y
3	7
2	3
1	⋮

Z	y
1	7
3	3
2	⋮
9	
6	
4	
1	
1	
⋮	

$$b = (Z^T Z)^{-1} (Z^T y)$$

$$y^{est} = b_0 + b_1 X_1 + b_2 X_2 + b_3 X_1^2 + b_4 X_1 X_2 + b_5 X_2^2$$

$z = (1, X_1, X_2, X_1^2, X_1 X_2, X_2^2)$

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 3

Quadratic Regression

It's trivial

X_1	X_2
3	2
1	1
⋮	⋮

Z
1
3
2
9
6
4
1
1
⋮

Each component of a z vector is called a term.

Each column of the Z matrix is called a term column

How many terms in a quadratic regression with m inputs?

- 1 constant term
- m linear terms
- $(m+1)\text{-choose-}2 = m(m+1)/2$ quadratic terms
- $(m+2)\text{-choose-}2$ terms in total = $O(m^2)$

Note that solving $b = (Z^T Z)^{-1} (Z^T y)$ is thus $O(m^6)$

$z = (1, X_1, X_2, X_1^2, X_1 X_2, X_2^2)$

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 4

Qth-degree polynomial Regression

X_1	X_2	Y
3	2	7
1	1	3
⋮	⋮	⋮

$x =$	3	2	$y =$	7
	1	1		3
	⋮	⋮		⋮

$z =$	1	3	2	9	6	⋯	$y =$	7
	1	1	1	1	1	⋯		3
	⋮	⋮	⋮	⋮	⋮	⋯		⋮

$$b = (Z^T Z)^{-1} (Z^T y)$$

$$y^{est} = b_0 + b_1 X_1 + \dots$$

$z =$ (all products of powers of inputs in which sum of powers is q or less)

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 5

m inputs, degree Q: how many terms?

= the number of unique terms of the form

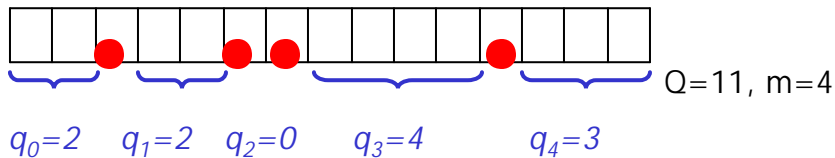
$$x_1^{q_1} x_2^{q_2} \dots x_m^{q_m} \text{ where } \sum_{i=1}^m q_i \leq Q$$

= the number of unique terms of the form

$$1^{q_0} x_1^{q_1} x_2^{q_2} \dots x_m^{q_m} \text{ where } \sum_{i=0}^m q_i = Q$$

= the number of lists of non-negative integers $[q_0, q_1, q_2, \dots, q_m]$ in which $\sum q_i = Q$

= the number of ways of placing Q red disks on a row of squares of length Q+m = (Q+m)-choose-Q



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 6

7: Radial Basis Functions (RBFs)

X_1	X_2	Y
3	2	7
1	1	3
⋮	⋮	⋮

x	y
3	7
2	3
1	⋮

z	y
⋮	7
⋮	3
⋮	⋮

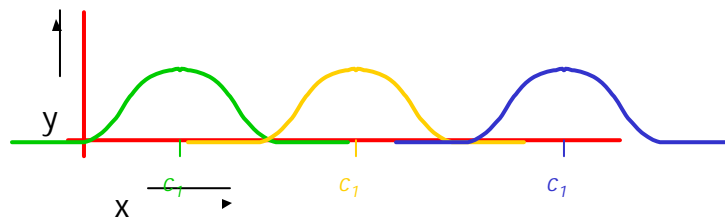
$z = (\text{list of radial basis function evaluations})$

$$b = (Z^T Z)^{-1} (Z^T y)$$

$$y^{est} = b_0 + b_1 X_1 + \dots$$

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 7

1-d RBFs

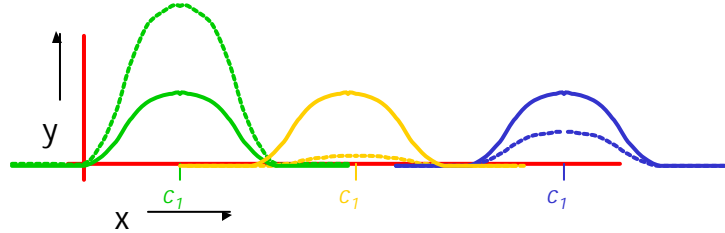


$$y^{est} = b_1 f_1(x) + b_2 f_2(x) + b_3 f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

Example



$$y^{est} = 2f_1(x) + 0.05f_2(x) + 0.5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 9

RBFs with Linear Regression

All c_i 's are held constant (initialized randomly or on a grid in m-dimensional input space)

KW also held constant (initialized to be large enough that there's decent overlap between basis functions*)

*Usually much better than the crappy overlap on my diagram

$$y^{est} = 2f_1(x) + 0.05f_2(x) + 5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 10

RBFs with Linear Regression

All c_i 's are held constant (initialized randomly or on a grid in m -dimensional input space)

KW also held constant (initialized to be large enough that there's decent overlap between basis functions*)

*Usually much better than the crappy overlap on my diagram

$$y^{est} = 2f_1(x) + 0.05f_2(x) + 5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

then given Q basis functions, define the matrix Z such that $Z_{kj} = \text{KernelFunction}(|x_k - c_j| / KW)$ where x_k is the k th vector of inputs

$$\text{And as before, } b = (Z^T Z)^{-1} (Z^T y)$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 11

RBFs with NonLinear Regression

Allow the c_i 's to adapt to the data (initialized randomly or on a grid in m -dimensional input space)

KW allowed to adapt to the data. (Some folks even let each basis function have its own KW_j , permitting fine detail in dense regions of input space)

$$y^{est} = 2f_1(x) + 0.05f_2(x) + 5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

But how do we now find all the b_j 's, c_i 's and KW ?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 12

RBFs with NonLinear Regression

Allow the c_i 's to adapt to the data (initialized randomly or on a grid in m-dimensional input space)

KW allowed to adapt to the data. (Some folks even let each basis function have its own KW_j , permitting fine detail in dense regions of input space)

$$y^{est} = 2f_1(x) + 0.05f_2(x) + 5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

But how do we now find all the b_j 's, c_i 's and KW?

Answer: Gradient Descent

RBFs with NonLinear Regression

Allow the c_i 's to adapt to the data (initialized randomly or on a grid in m-dimensional input space)

KW allowed to adapt to the data. (Some folks even let each basis function have its own KW_j , permitting fine detail in dense regions of input space)

$$y^{est} = 2f_1(x) + 0.05f_2(x) + 5f_3(x)$$

where

$$f_i(x) = \text{KernelFunction}(|x - c_i| / KW)$$

But how do we now find all the b_j 's, c_i 's and KW?

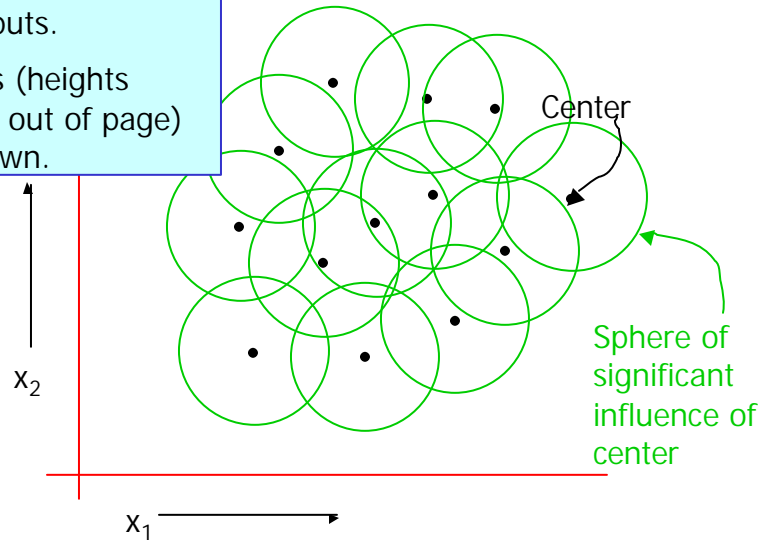
(But I'd like to see, or hope someone's already done, a hybrid, where the c_i 's and KW are updated with gradient descent while the b_j 's use matrix inversion)

Answer: Gradient Descent

Radial Basis Functions in 2-d

Two inputs.

Outputs (heights sticking out of page) not shown.

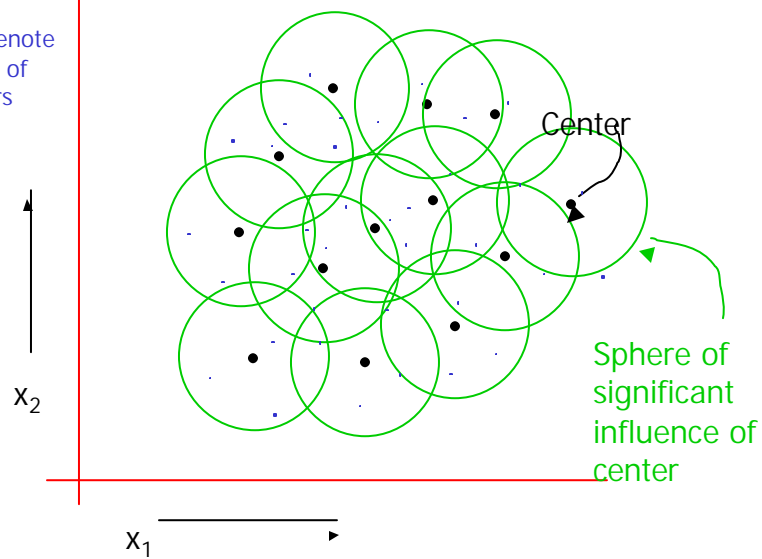


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 15

Happy RBFs in 2-d

Blue dots denote coordinates of input vectors



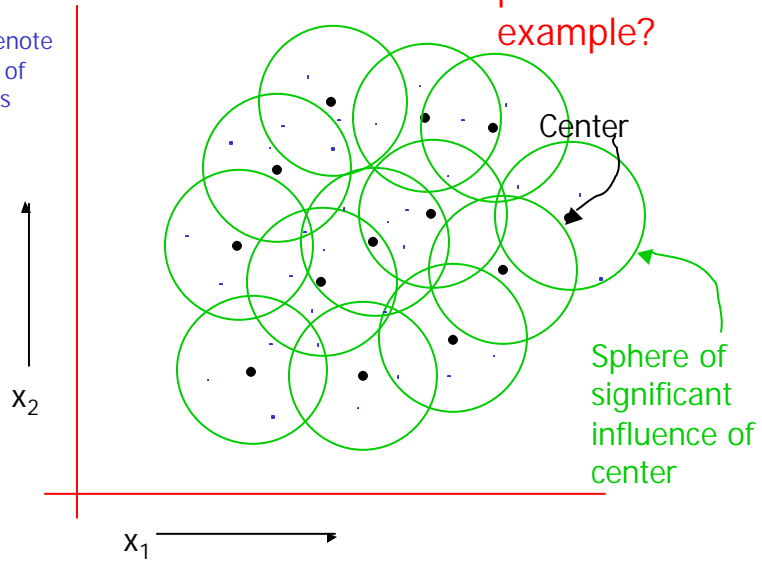
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 16

Crabby RBFs in 2-d

What's the problem in this example?

Blue dots denote coordinates of input vectors



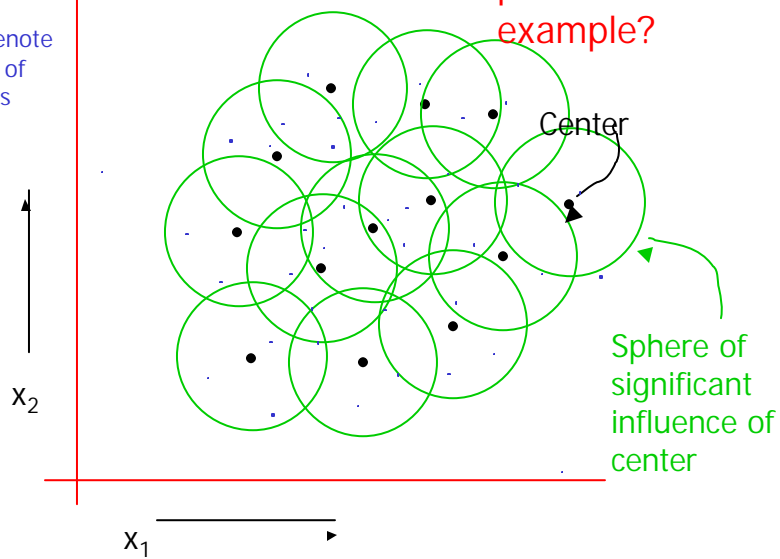
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 17

More crabby RBFs

And what's the problem in this example?

Blue dots denote coordinates of input vectors

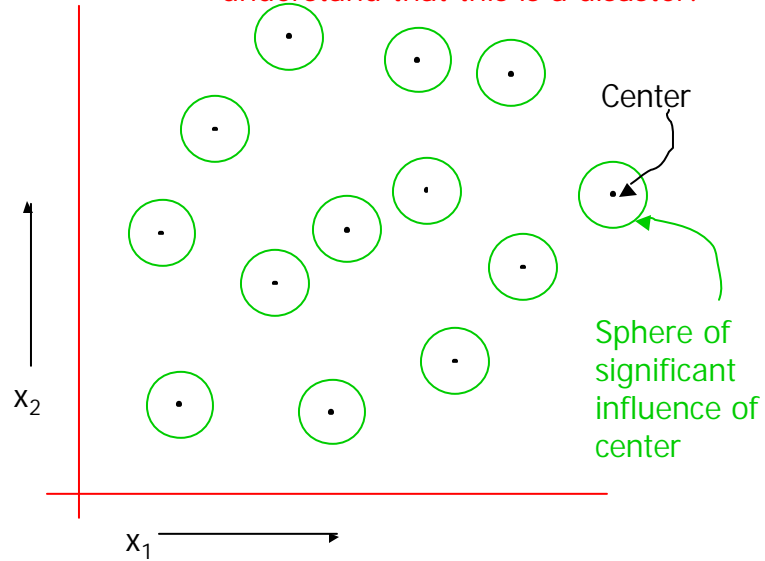


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 18

Hopeless!

Even before seeing the data, you should understand that this is a disaster!

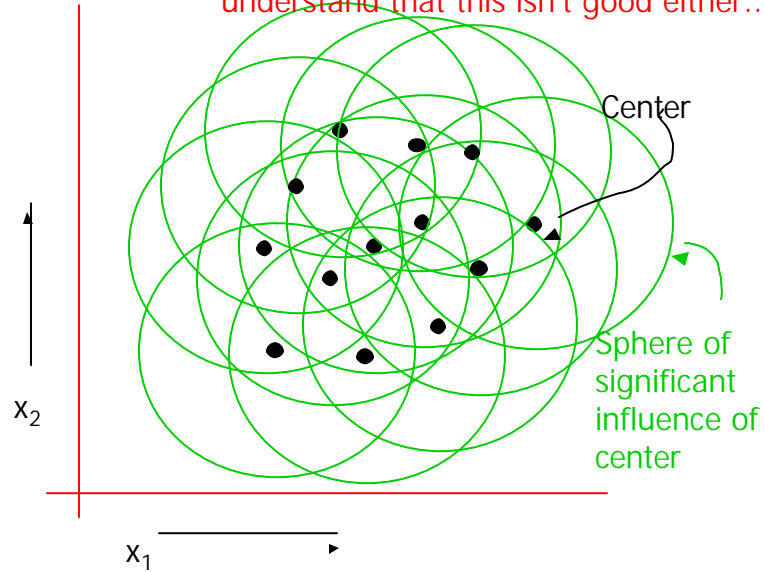


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 19

Unhappy

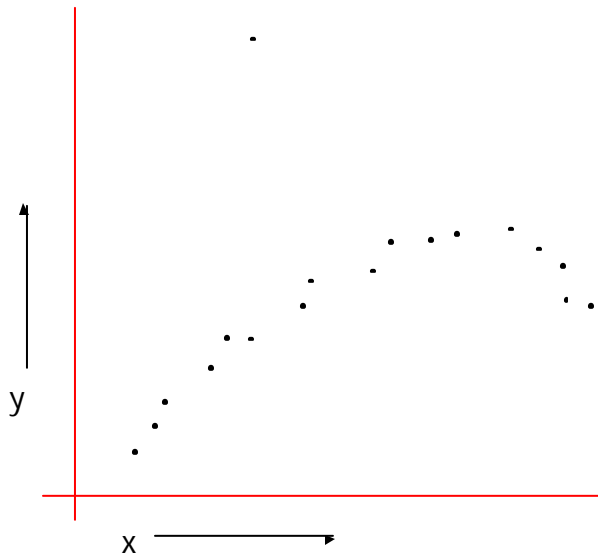
Even before seeing the data, you should understand that this isn't good either..



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 20

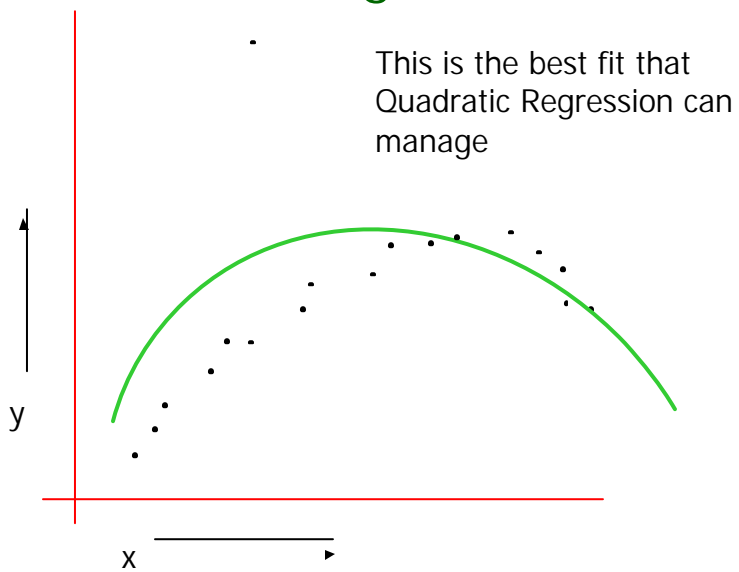
6: Robust Regression



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 21

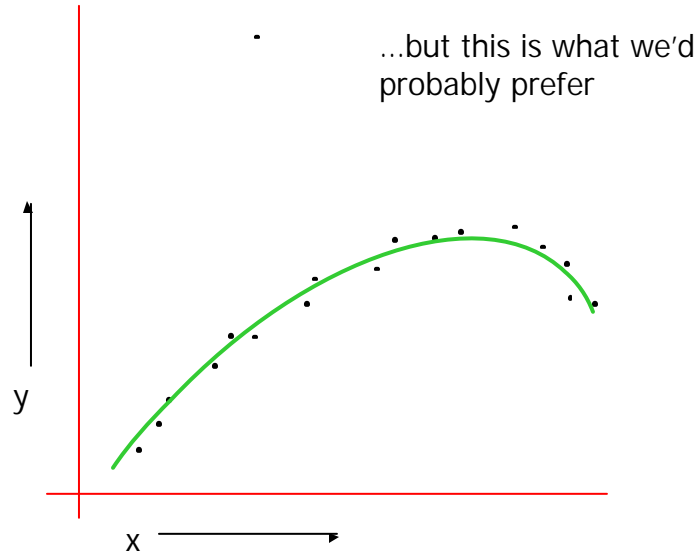
Robust Regression



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 22

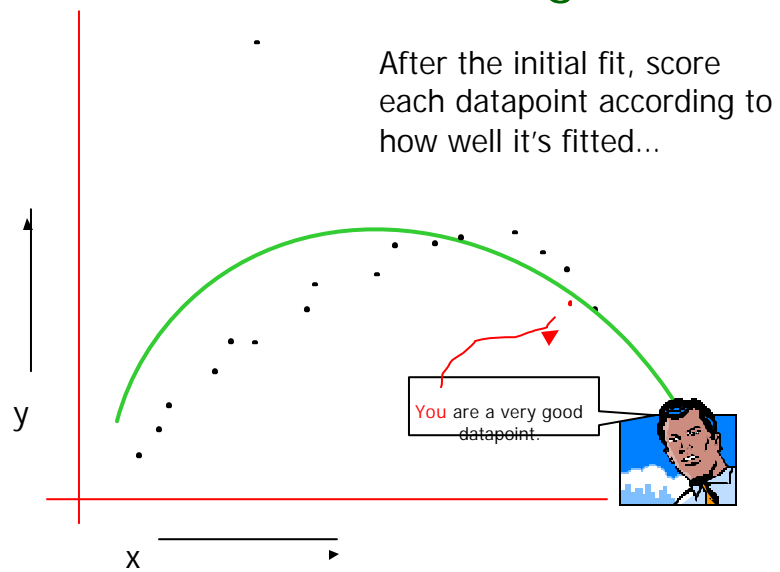
Robust Regression



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 23

LOESS-based Robust Regression

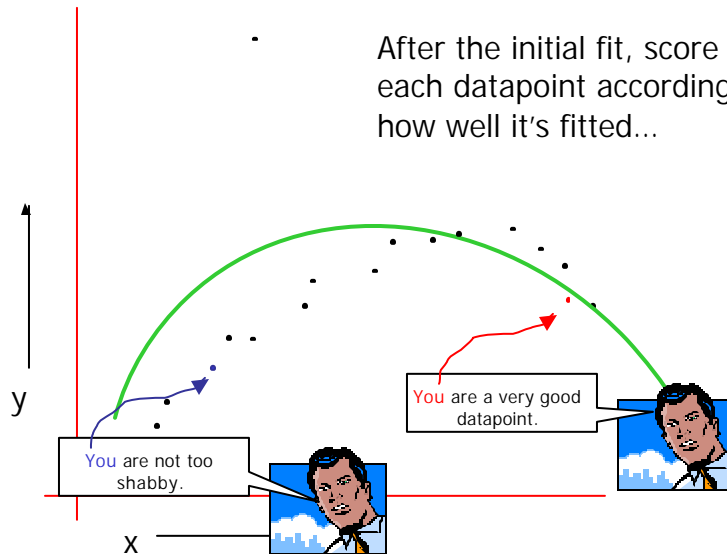


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 24

LOESS-based Robust Regression

After the initial fit, score each datapoint according to how well it's fitted...

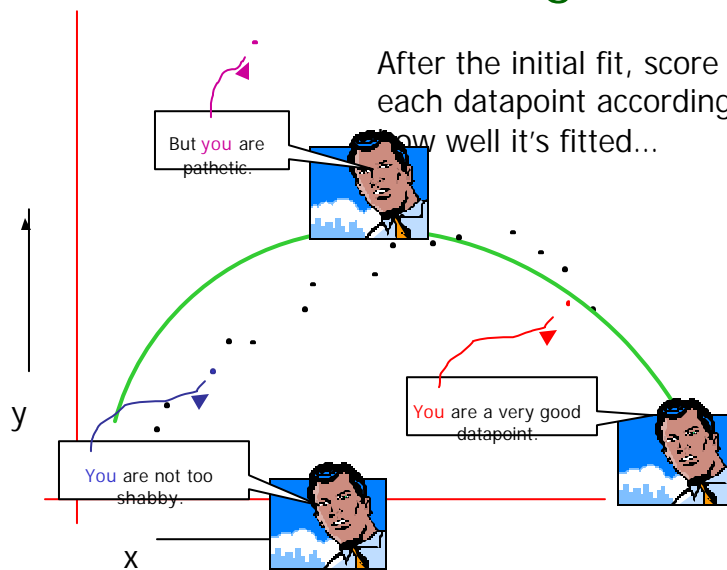


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 25

LOESS-based Robust Regression

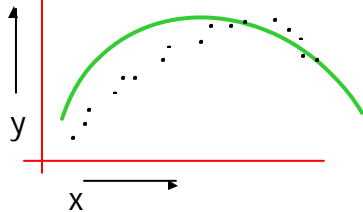
After the initial fit, score each datapoint according to how well it's fitted...



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 26

Robust Regression



For $k = 1$ to $R...$

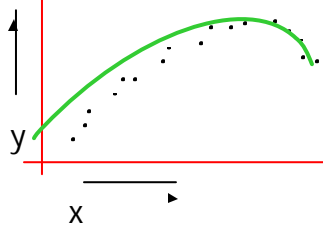
- Let (x_k, y_k) be the k th datapoint
- Let y_k^{est} be predicted value of y_k
- Let w_k be a weight for datapoint k that is large if the datapoint fits well and small if it fits badly:

$$w_k = \text{KernelFn}([y_k - y_k^{est}]^2)$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 27

Robust Regression



For $k = 1$ to $R...$

- Let (x_k, y_k) be the k th datapoint
- Let y_k^{est} be predicted value of y_k
- Let w_k be a weight for datapoint k that is large if the datapoint fits well and small if it fits badly:

$$w_k = \text{KernelFn}([y_k - y_k^{est}]^2)$$

Then redo the regression using weighted datapoints.

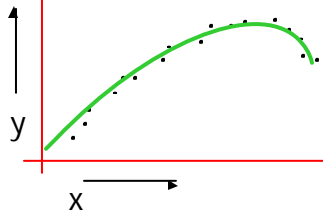
I taught you how to do this in the "Instance-based" lecture (only then the weights depended on distance in input-space)

Guess what happens next?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 28

Robust Regression



For $k = 1$ to $R...$

- Let (x_k, y_k) be the k th datapoint
- Let y_k^{est} be predicted value of y_k
- Let w_k be a weight for datapoint k that is large if the datapoint fits well and small if it fits badly:

$$w_k = \text{KernelFn}([y_k - y_k^{est}]^2)$$

Then redo the regression using weighted datapoints.

I taught you how to do this in the "Instance-based" lecture (only then the weights depended on distance in input-space)

Repeat whole thing until converged!

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 29

Robust Regression---what we're doing

What regular regression does:

Assume y_k was originally generated using the following recipe:

$$y_k = \mathbf{b}_0 + \mathbf{b}_1 x_k + \mathbf{b}_2 x_k^2 + N(0, \mathbf{s}^2)$$

Computational task is to find the Maximum Likelihood \mathbf{b}_0 , \mathbf{b}_1 and \mathbf{b}_2

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 30

Robust Regression---what we're doing

What LOESS robust regression does:

Assume y_k was originally generated using the following recipe:

With probability p :

$$y_k = \mathbf{b}_0 + \mathbf{b}_1 x_k + \mathbf{b}_2 x_k^2 + N(0, \mathbf{s}^2)$$

But otherwise

$$y_k \sim N(\mathbf{m}, \mathbf{s}_{huge}^2)$$

Computational task is to find the Maximum Likelihood $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, p, \mathbf{m}$ and \mathbf{s}_{huge}

Robust Regression---what we're doing

What LOESS robust regression does:

Assume y_k was originally generated using the following recipe:

With probability p :

$$y_k = \mathbf{b}_0 + \mathbf{b}_1 x_k + \mathbf{b}_2 x_k^2 + N(0, \mathbf{s}^2)$$

But otherwise

$$y_k \sim N(\mathbf{m}, \mathbf{s}_{huge}^2)$$

Computational task is to find the Maximum Likelihood $\mathbf{b}_0, \mathbf{b}_1, \mathbf{b}_2, p, \mathbf{m}$ and \mathbf{s}_{huge}

Mysteriously, the reweighting procedure does this computation for us.

Your first glimpse of two spectacular letters:

E.M.

5: Regression Trees

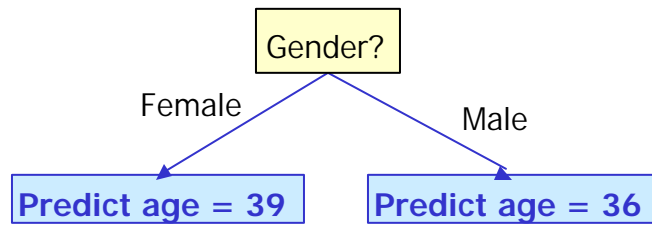
- “Decision trees for regression”

A regression tree leaf

Predict age = 47

Mean age of records
matching this leaf node

A one-split regression tree



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 35

Choosing the attribute to split on

Gender	Rich?	Num. Children	Num. Beany Babies	Age
Female	No	2	1	38
Male	No	0	0	24
Male	Yes	0	5+	72
.

- We can't use information gain.
- What should we use?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 36

Choosing the attribute to split on

Gender	Rich?	Num. Children	Num. Beany Babies	Age
Female	No	2	1	38
Male	No	0	0	24
Male	Yes	0	5+	72
:	:	:	:	:

$MSE(Y|X)$ = The expected squared error if we must predict a record's Y value given only knowledge of the record's X value

If we're told $x=j$, the smallest expected error comes from predicting the mean of the Y-values among those records in which $x=j$. Call this mean quantity $\mu_y^{x=j}$

Then...

$$MSE(Y | X) = \frac{1}{R} \sum_{j=1}^{N_x} \sum_{(k \text{ such that } x_k = j)} (y_k - \mu_y^{x=j})^2$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 37

Choosing the attribute to split on

Gender	Rich?	Num.	Num. Beany Babies	Age
Female	No			
Male	No			
Male	Yes			
:	:			

Regression tree attribute selection: greedily choose the attribute that minimizes $MSE(Y|X)$

Guess what we do about real-valued inputs?

Guess how we prevent overfitting.

$MSE(Y|X)$ = The expected squared error if we must predict a record's Y value given only knowledge of the record's X value

If we're told $x=j$, the smallest expected error comes from predicting the mean of the Y-values among those records in which $x=j$. Call this mean quantity $\mu_y^{x=j}$

Then...

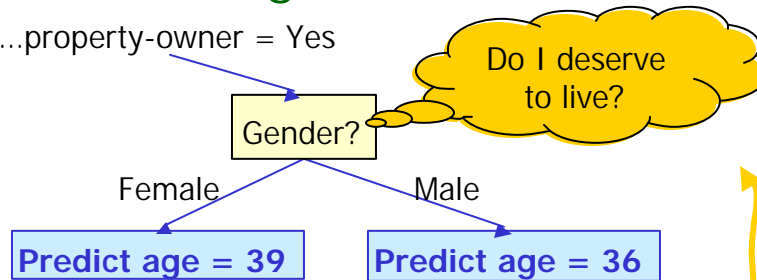
$$MSE(Y | X) = \frac{1}{R} \sum_{j=1}^{N_x} \sum_{(k \text{ such that } x_k = j)} (y_k - \mu_y^{x=j})^2$$

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 38

Pruning Decision

...property-owner = Yes



property-owning females = 56712
 Mean age among POFs = 39
 Age std dev among POFs = 12

property-owning males = 55800
 Mean age among POMs = 36
 Age std dev among POMs = 11.5

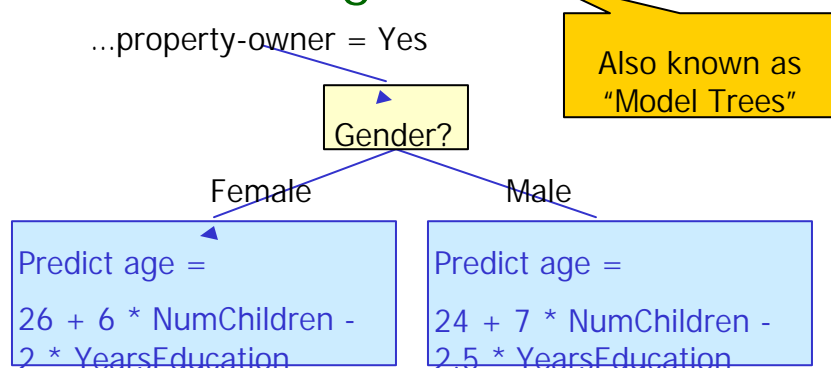
Use a standard Chi-squared test of the null-hypothesis "these two populations have the same mean" and Bob's your uncle.

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 39

Linear Regression Trees

...property-owner = Yes



Leaves contain linear functions (trained using linear regression on all records matching that leaf)

Split attribute chosen to minimize MSE of regressed children.

Pruning with a different Chi-squared

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 40

Linear Regression Trees

...property-owner = Yes

Gender?

Female

Predict age =

$$26 + 6 * M + 2 * Years$$

Also known as
"MART Trees"

Detail: You typically ignore any categorical attribute that has been tested on higher up in the tree during the regression. But use all untested attributes, and use real-valued attributes even if they've been tested above

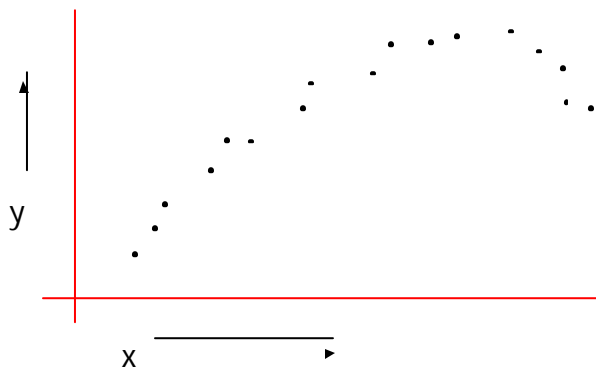
Leaves contain linear regression functions (trained on records matching the criteria)

Attributes are chosen to minimize regression error.

Pruning with a different Chi-squared

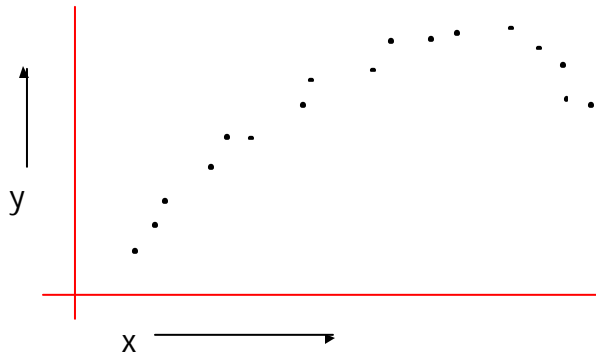
Test your understanding

Assuming **regular** regression trees, can you sketch a graph of the fitted function $y^{est}(x)$ over this diagram?



Test your understanding

Assuming **linear** regression trees, can you sketch a graph of the fitted function $y^{est}(x)$ over this diagram?



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 43

4: GMDH (c.f. BACON, AIM)

- Group Method Data Handling
- A very simple but very good idea:
 1. Do linear regression
 2. Use cross-validation to discover whether any quadratic term is good. If so, add it as a basis function and loop.
 3. Use cross-validation to discover whether any of a set of familiar functions (log, exp, sin etc) applied to any previous basis function helps. If so, add it as a basis function and loop.
 4. Else stop

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 44

GMDH (c.f. BACON, AIM)

- Group Method Data Handling
- A very simple but very good idea:
 1. Determine typical learned function:
 2. Usage^{est} = height - 3.1 sqrt(weight) +
4.3 income / (cos (NumCars))
 3. Use cross-validation to discover whether any of a set of familiar functions (log, exp, sin etc) applied to any previous basis function helps. If so, add it as a basis function and loop.
 4. Else stop

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 45

3: Cascade Correlation

- A super-fast form of Neural Network learning
- Begins with 0 hidden units
- Incrementally adds hidden units, one by one, doing ingeniously little recomputation between each unit addition

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 46

Cascade beginning

Begin with a regular dataset

Nonstandard notation:

- $x^{(i)}$ is the i 'th attribute
- $x^{(i)}_k$ is the value of the i 'th attribute in the k 'th record

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2
:	:	:	:	:

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 47

Cascade first step

Begin with a regular dataset

Find weights $w^{(0)}_i$ to best fit Y .

I.E. to minimize

$$\sum_{k=1}^R (y_k - y_k^{(0)})^2 \text{ where } y_k^{(0)} = \sum_{j=1}^m w_j^{(0)} x_k^{(j)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2
:	:	:	:	:

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 48

Consider our errors...

Begin with a regular dataset

Find weights $w^{(0)}_i$ to best fit Y.

I.E. to minimize

$$\sum_{k=1}^R (y_k - y_k^{(0)})^2 \text{ where } y_k^{(0)} = \sum_{j=1}^m w_j^{(0)} x_k^{(j)}$$

$$\text{Define } e_k^{(0)} = y_k - y_k^{(0)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 49

Create a hidden unit...

Find weights $u^{(0)}_i$ to define a new basis function $H^{(0)}(x)$ of the inputs.

Make it specialize in predicting the errors in our original fit:

Find $\{u^{(0)}_i\}$ to maximize correlation between $H^{(0)}(x)$ and $E^{(0)}$ where

$$H^{(0)}(\mathbf{x}) = g\left(\sum_{j=1}^m u_j^{(0)} x^{(j)}\right)$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 50

Cascade next step

Find weights $w^{(1)}_i p^{(1)}_0$ to better fit Y.

I.E. to minimize

$$\sum_{k=1}^R (y_k - y_k^{(1)})^2 \text{ where } y_k^{(1)} = \sum_{j=1}^m w_j^{(0)} x_k^{(j)} + p_j^{(0)} h_k^{(0)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 51

Now look at new errors

Find weights $w^{(1)}_i p^{(1)}_0$ to better fit Y.

$$\text{Define } e_k^{(1)} = y_k - y_k^{(1)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$	$E^{(1)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$	$e^{(1)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$	$e^{(1)}_2$
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 52

Create next hidden unit...

Find weights $u^{(1)}_i, v^{(1)}_0$ to define a new basis function $H^{(1)}(x)$ of the inputs.

Make it specialize in predicting the errors in our original fit:

Find $\{u^{(1)}_i, v^{(1)}_0\}$ to maximize correlation between $H^{(1)}(x)$ and $E^{(1)}$ where
$$H^{(1)}(\mathbf{x}) = g\left(\sum_{j=1}^m u_j^{(1)} x^{(j)} + v_0^{(1)} h^{(0)}\right)$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$	$E^{(1)}$	$H^{(1)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$	$e^{(1)}_1$	$h^{(1)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$	$e^{(1)}_2$	$h^{(1)}_2$
:	:	:	:	:	:	:	:	:	:	:

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 53

Cascade n'th step

Find weights $w^{(n)}_i, p^{(n)}_j$ to better fit Y .

I.E. to minimize

$$\sum_{k=1}^R (y_k - y_k^{(n)})^2 \text{ where } y_k^{(n)} = \sum_{j=1}^m w_j^{(n)} x_k^{(j)} + \sum_{j=1}^{n-1} p_j^{(n)} h_k^{(j)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$	$E^{(1)}$	$H^{(1)}$...	$Y^{(n)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$	$e^{(1)}_1$	$h^{(1)}_1$...	$y^{(n)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$	$e^{(1)}_2$	$h^{(1)}_2$...	$y^{(n)}_2$
:	:	:	:	:	:	:	:	:	:	:	:	:

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 54

Now look at new errors

Find weights $w^{(n)}_i p^{(n)}_j$ to better fit Y.

I.E. to minimize

$$\text{Define } e_k^{(n)} = y_k - y_k^{(n)}$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$	$E^{(1)}$	$H^{(1)}$...	$Y^{(n)}$	$E^{(n)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$	$e^{(1)}_1$	$h^{(1)}_1$...	$y^{(n)}_1$	$e^{(n)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$	$e^{(1)}_2$	$h^{(1)}_2$...	$y^{(n)}_2$	$e^{(n)}_2$
:	:	:	:	:	:	:	:	:	:	:	:	:	:

Create n'th hidden unit...

Find weights $u^{(n)}_i, v^{(n)}_j$ to define a new basis function $H^{(n)}(x)$ of the inputs.

Make it specialize in predicting the errors in our previous fit:

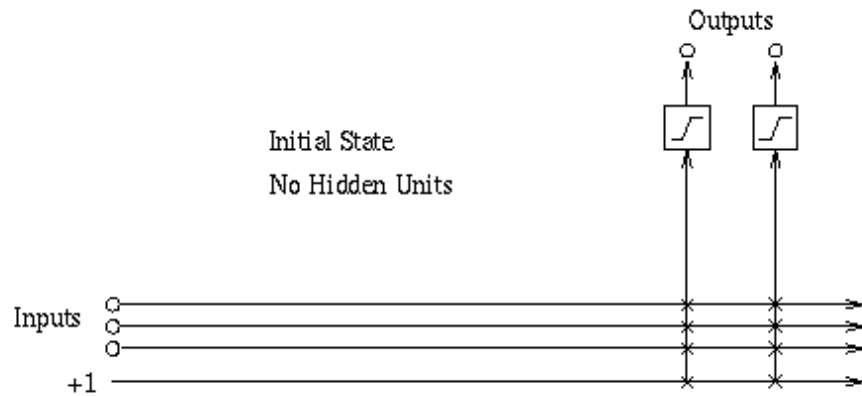
Find $\{u^{(n)}_i, v^{(n)}_j\}$ to maximize correlation between $H^{(n)}(x)$ and $E^{(n)}$ where

$$H^{(n)}(\mathbf{x}) = g \left(\sum_{j=1}^m u_j^{(n)} x^{(j)} + \sum_{j=1}^{n-1} v_j^{(n)} h^{(j)} \right)$$

$X^{(0)}$	$X^{(1)}$...	$X^{(m)}$	Y	$Y^{(0)}$	$E^{(0)}$	$H^{(0)}$	$Y^{(1)}$	$E^{(1)}$	$H^{(1)}$...	$Y^{(n)}$	$E^{(n)}$	$H^{(n)}$
$x^{(0)}_1$	$x^{(1)}_1$...	$x^{(m)}_1$	y_1	$y^{(0)}_1$	$e^{(0)}_1$	$h^{(0)}_1$	$y^{(1)}_1$	$e^{(1)}_1$	$h^{(1)}_1$...	$y^{(n)}_1$	$e^{(n)}_1$	$h^{(n)}_1$
$x^{(0)}_2$	$x^{(1)}_2$...	$x^{(m)}_2$	y_2	$y^{(0)}_2$	$e^{(0)}_2$	$h^{(0)}_2$	$y^{(1)}_2$	$e^{(1)}_2$	$h^{(1)}_2$...	$y^{(n)}_2$	$e^{(n)}_2$	$h^{(n)}_2$
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:

Continue until satisfied with fit...

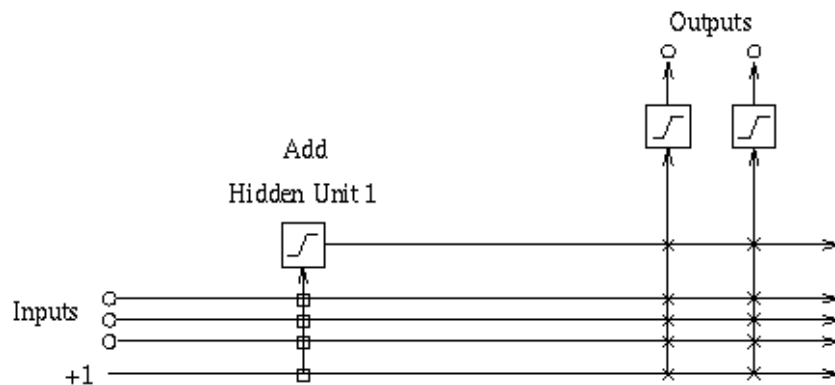
Visualizing first iteration



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 57

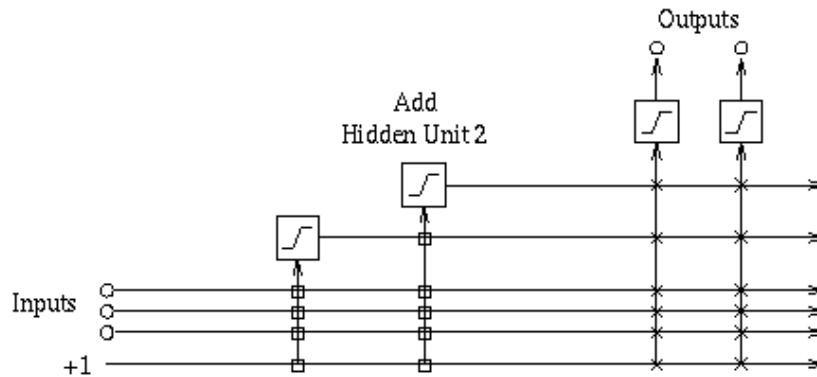
Visualizing second iteration



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 58

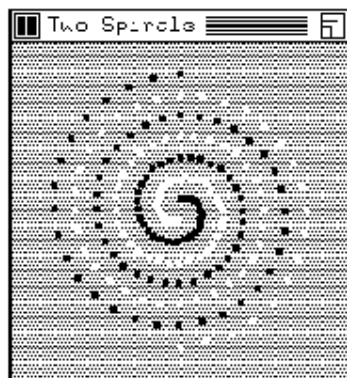
Visualizing third iteration



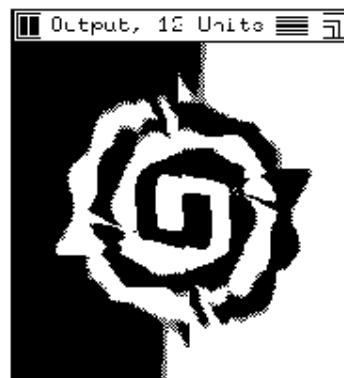
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 59

Example: Cascade Correlation for Classification

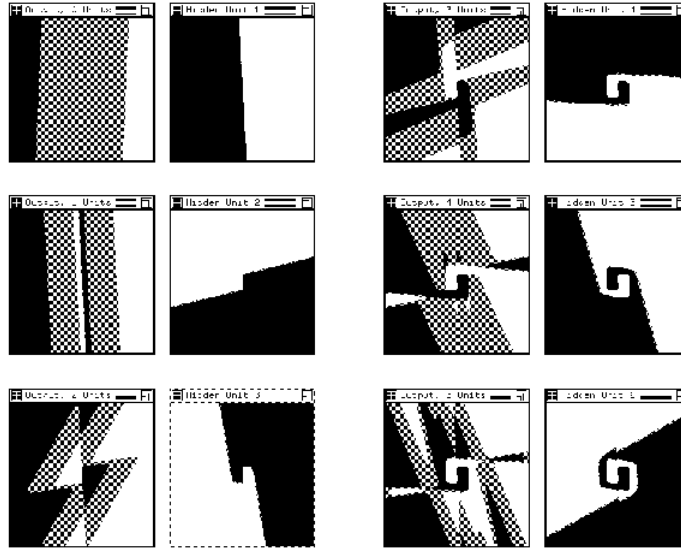


Copyright © 2001, Andrew W. Moore



Machine Learning Favorites: Slide 60

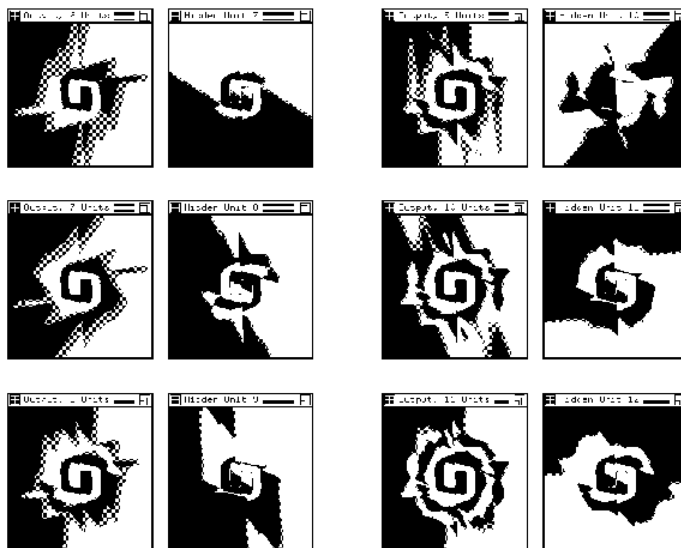
Training two spirals: Steps 1-6



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 61

Training two spirals: Steps 2-12



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 62

If you like Cascade Correlation...

See Also

- Projection Pursuit

In which you add together many non-linear non-parametric scalar functions of carefully chosen directions
Each direction is chosen to maximize error-reduction from the best scalar function

- ADA-Boost

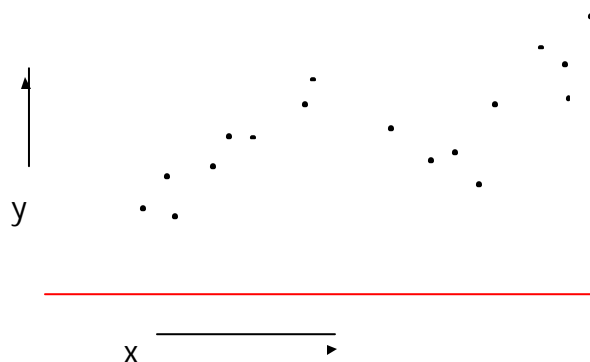
An additive combination of regression trees in which the $n+1$ 'th tree learns the error of the n 'th tree

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 63

2: Multilinear Interpolation

Consider this dataset. Suppose we wanted to create a continuous and piecewise linear fit to the data

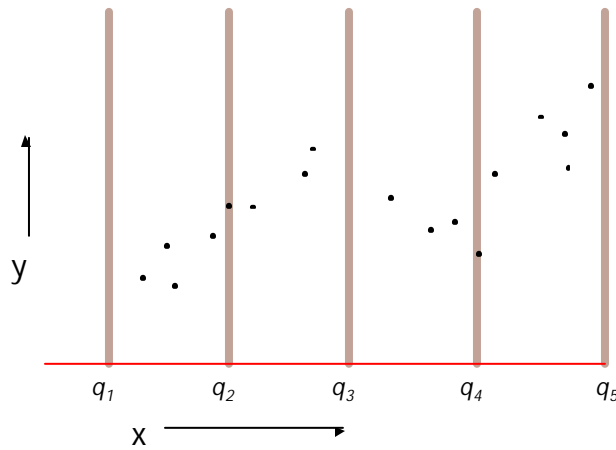


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 64

Multilinear Interpolation

Create a set of knot points: selected X-coordinates (usually equally spaced) that cover the data

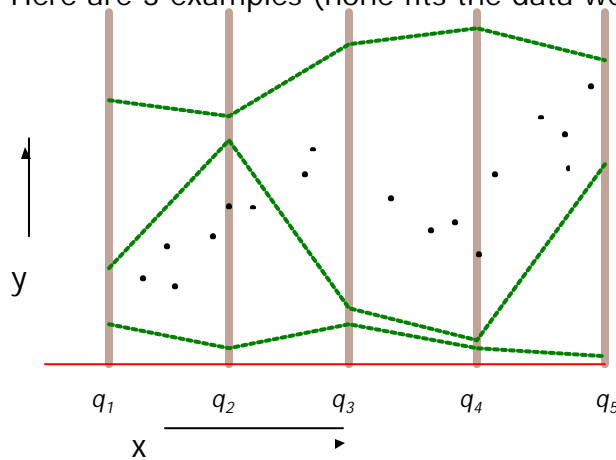


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 65

Multilinear Interpolation

We are going to assume the data was generated by a noisy version of a function that can only bend at the knots. Here are 3 examples (none fits the data well)

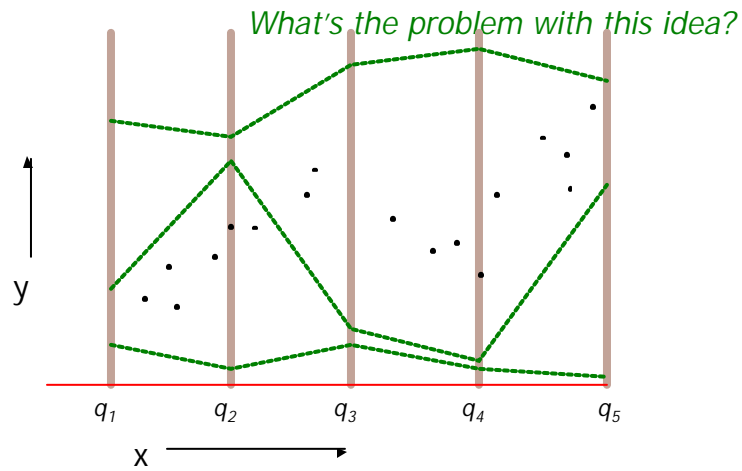


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 66

How to find the best fit?

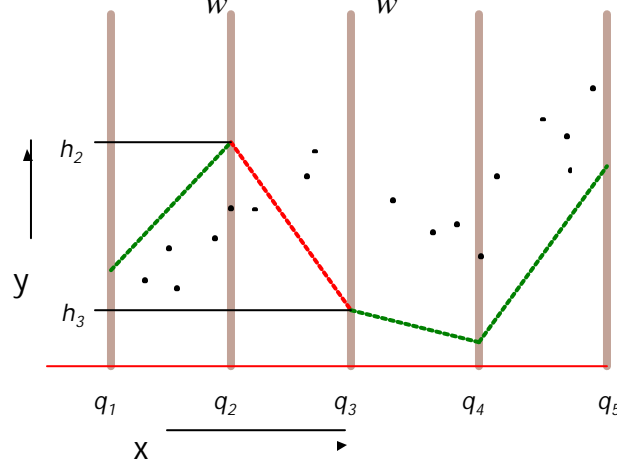
Idea 1: Simply perform a separate regression in each segment for each part of the curve



How to find the best fit?

Let's look at what goes on in the red segment

$$y^{est}(x) = \frac{(q_3 - x)}{w} h_2 + \frac{(x - q_2)}{w} h_3 \text{ where } w = q_3 - q_2$$

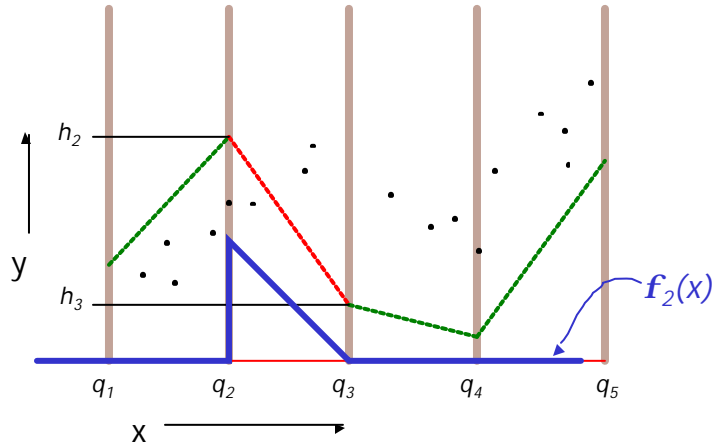


How to find the best fit?

In the red segment...

$$y^{est}(x) = h_2 f_2(x) + h_3 f_3(x)$$

$$\text{where } f_2(x) = 1 - \frac{x - q_2}{w}, f_3(x) = 1 - \frac{q_3 - x}{w}$$



Copyright © 2001, Andrew W. Moore

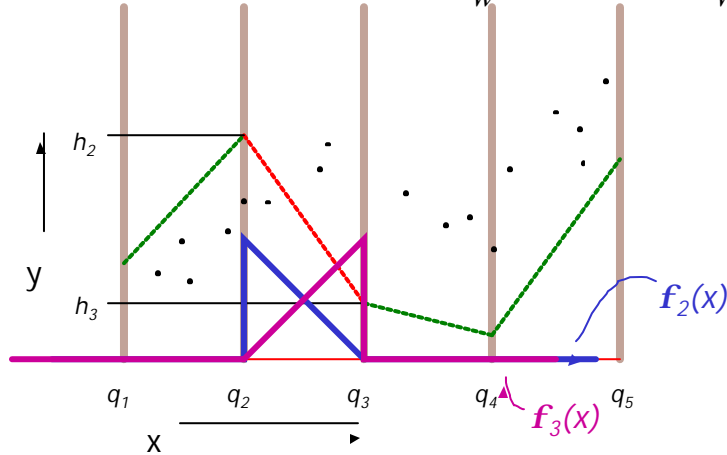
Machine Learning Favorites: Slide 69

How to find the best fit?

In the red segment...

$$y^{est}(x) = h_2 f_2(x) + h_3 f_3(x)$$

$$\text{where } f_2(x) = 1 - \frac{x - q_2}{w}, f_3(x) = 1 - \frac{q_3 - x}{w}$$



Copyright © 2001, Andrew W. Moore

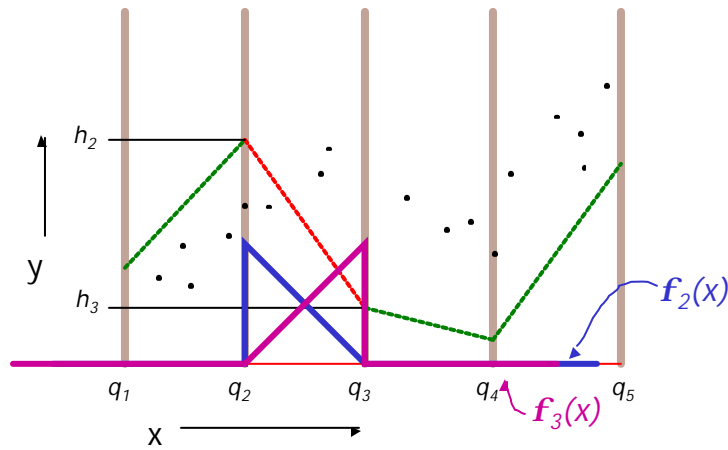
Machine Learning Favorites: Slide 70

How to find the best fit?

In the red segment...

$$y^{est}(x) = h_2 f_2(x) + h_3 f_3(x)$$

$$\text{where } f_2(x) = 1 - \frac{|x - q_2|}{w}, f_3(x) = 1 - \frac{|x - q_3|}{w}$$



Copyright © 2001, Andrew W. Moore

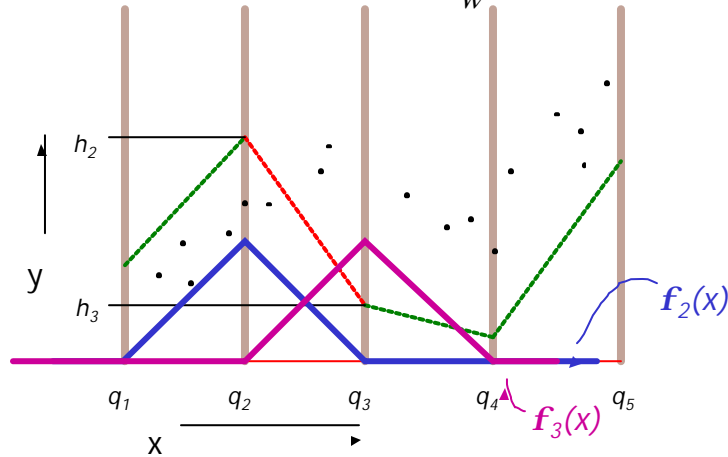
Machine Learning Favorites: Slide 71

How to find the best fit?

In the red segment...

$$y^{est}(x) = h_2 f_2(x) + h_3 f_3(x)$$

$$\text{where } f_2(x) = 1 - \frac{|x - q_2|}{w}, f_3(x) = 1 - \frac{|x - q_3|}{w}$$



Copyright © 2001, Andrew W. Moore

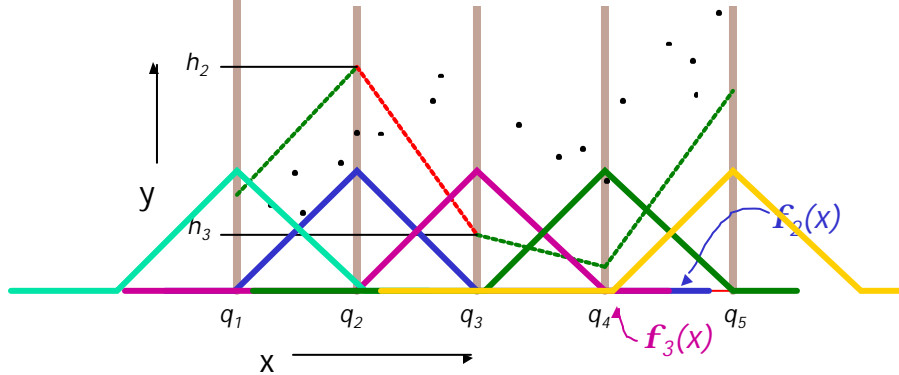
Machine Learning Favorites: Slide 72

How to find the best fit?

In **general**

$$y^{est}(x) = \sum_{i=1}^{N_k} h_i f_i(x)$$

$$\text{where } f_i(x) = \begin{cases} 1 - \frac{|x - q_i|}{w} & \text{if } |x - q_i| < w \\ 0 & \text{otherwise} \end{cases}$$



Copyright © 2001, Andrew W. Moore

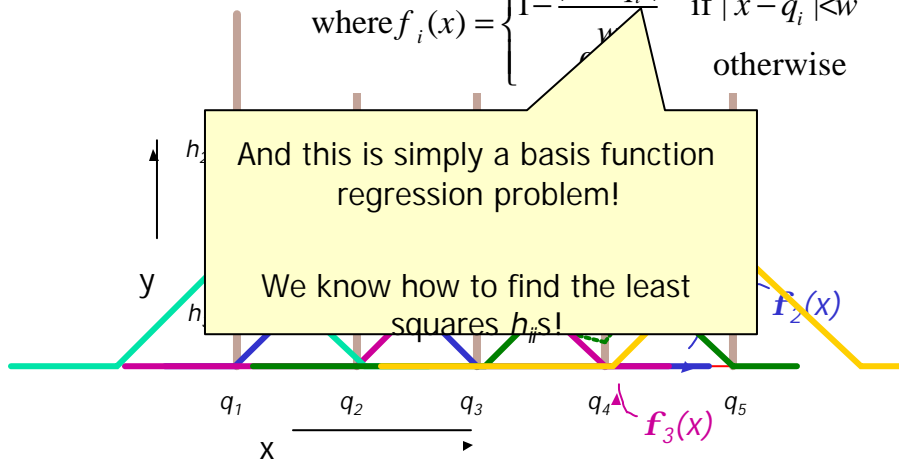
Machine Learning Favorites: Slide 73

How to find the best fit?

In **general**

$$y^{est}(x) = \sum_{i=1}^{N_k} h_i f_i(x)$$

$$\text{where } f_i(x) = \begin{cases} 1 - \frac{|x - q_i|}{w} & \text{if } |x - q_i| < w \\ 0 & \text{otherwise} \end{cases}$$

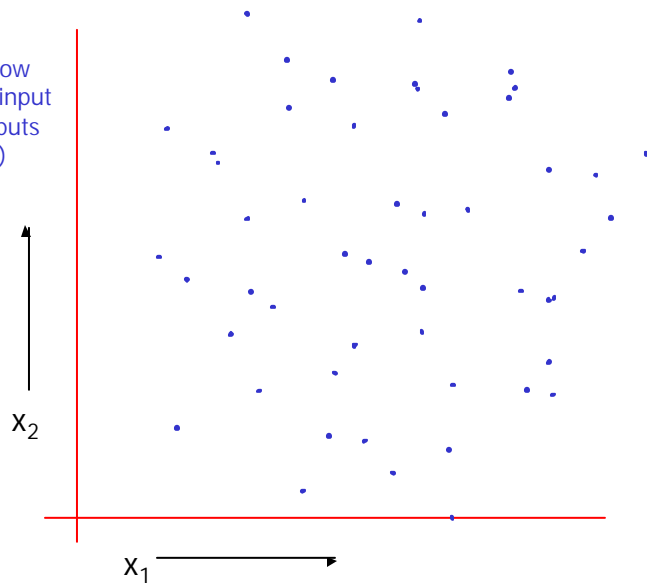


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 74

In two dimensions...

Blue dots show locations of input vectors (outputs not depicted)



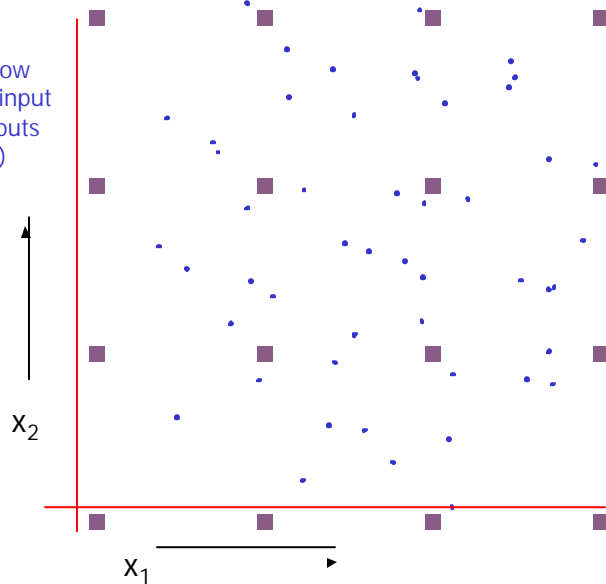
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 75

In two dimensions...

Blue dots show locations of input vectors (outputs not depicted)

Each purple dot is a knot point. It will contain the height of the estimated surface

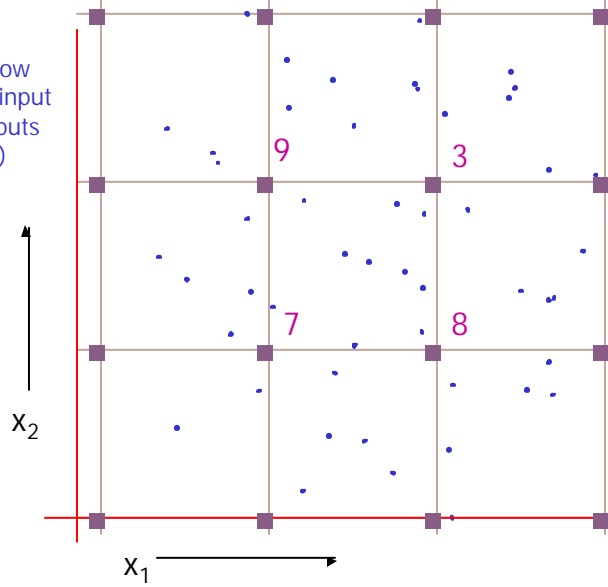


Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 76

In two dimensions...

Blue dots show locations of input vectors (outputs not depicted)



Each purple dot is a knot point. It will contain the height of the estimated surface

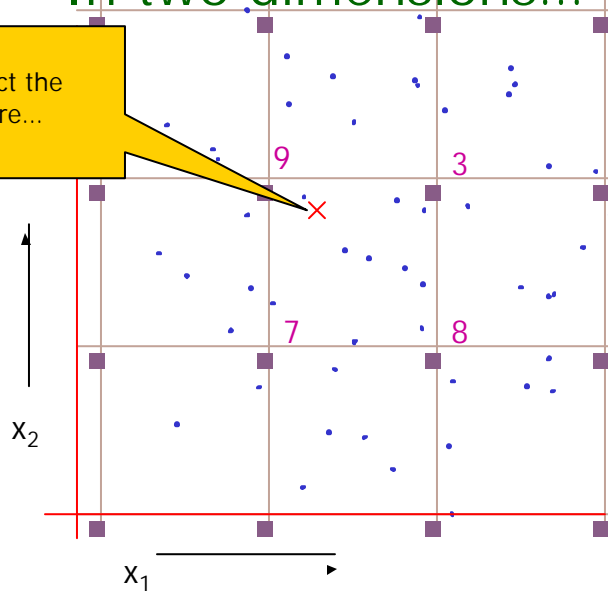
But how do we do the interpolation to ensure that the surface is continuous?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 77

In two dimensions...

To predict the value here...



Each purple dot is a knot point. It will contain the height of the estimated surface

But how do we do the interpolation to ensure that the surface is continuous?

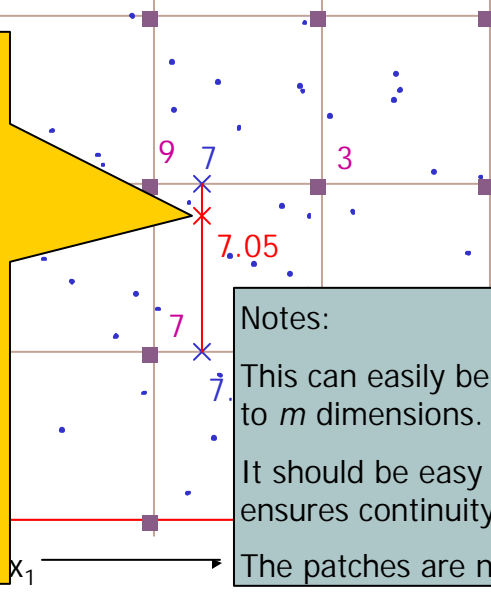
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 78

In two dimensions...

Blue
local
vertical
no

To predict the value here...
First interpolate its value on two opposite edges...
Then interpolate between those two values



Each purple dot is a knot point. It will contain the height of the estimated surface

But how do we do the interpolation to ensure that the

Notes:

This can easily be generalized to m dimensions.

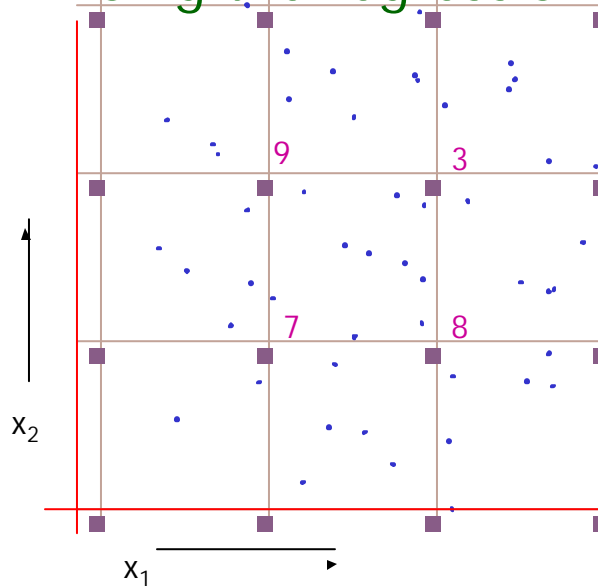
It should be easy to see that it ensures continuity

The patches are not linear

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 81

Doing the regression



Given data, how do we find the optimal knot heights?

Happily, it's simply a two-dimensional basis function problem.

(Working out the basis functions is tedious, unilluminating, and easy)

What's the problem in higher dimensions?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 82

1: MARS

- Multivariate Adaptive Regression Splines
- Invented by Jerry Friedman (one of Andrew's heroes)
- Simplest version:

Let's assume the function we are learning is of the following form:

$$y^{est}(\mathbf{x}) = \sum_{k=1}^m g_k(x_k)$$

Instead of a linear combination of the inputs, it's a linear combination of non-linear functions of *individual* inputs

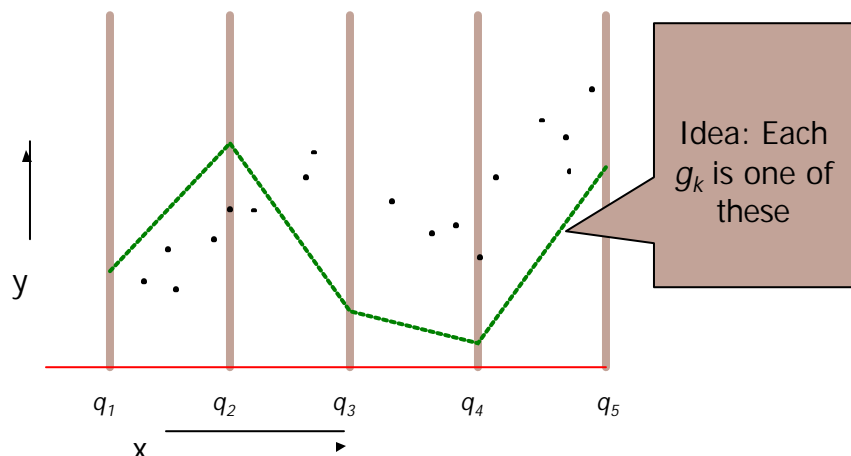
Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 83

MARS

$$y^{est}(\mathbf{x}) = \sum_{k=1}^m g_k(x_k)$$

Instead of a linear combination of the inputs, it's a linear combination of non-linear functions of *individual* inputs



Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 84

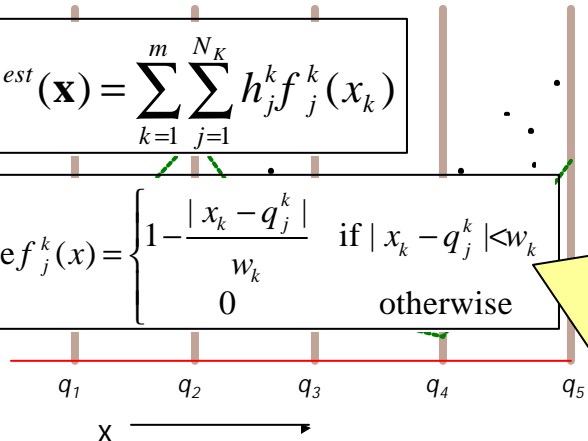
MARS

$$y^{est}(\mathbf{x}) = \sum_{k=1}^m g_k(x_k)$$

Instead of a linear combination of the inputs, it's a linear combination of non-linear functions of *individual* inputs

$$y^{est}(\mathbf{x}) = \sum_{k=1}^m \sum_{j=1}^{N_k} h_j^k f_j^k(x_k)$$

$$\text{where } f_j^k(x) = \begin{cases} 1 - \frac{|x_k - q_j^k|}{w_k} & \text{if } |x_k - q_j^k| < w_k \\ 0 & \text{otherwise} \end{cases}$$



q_j^k : The location of the j 'th knot in the k 'th dimension
 h_j^k : The regressed height of the j 'th knot in the k 'th dimension
 w^k : The spacing between knots in the k 'th dimension

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 85

That's not complicated enough!

- Okay, now let's get serious. We'll allow arbitrary "two-way interactions":

$$y^{est}(\mathbf{x}) = \sum_{k=1}^m g_k(x_k) + \sum_{k=1}^m \sum_{t=k+1}^m g_{kt}(x_k, x_t)$$

The function we're learning is allowed to be a sum of non-linear functions over all one-d and 2-d subsets of attributes

Can still be expressed as a linear combination of basis functions

Thus learnable by linear regression

Full MARS: Uses cross-validation to choose a subset of subspaces, knot resolution and other parameters.

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 86

If you like MARS...

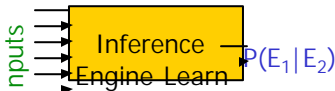
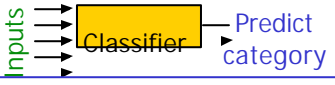
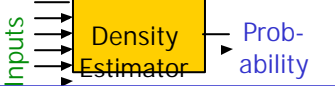
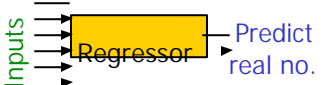
...See also CMAC (Cerebellar Model Articulated Controller) by James Albus (another of Andrew's heroes)

- Many of the same gut-level intuitions
- But entirely in a neural-network, biologically plausible way
 - (All the low dimensional functions are by means of lookup tables, trained with a delta-rule and using a clever blurred update and hash-tables)

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 87

Where are we now?

	Joint DE, Bayes Net Structure Learning
	Dec Tree, Sigmoid Perceptron, Sigmoid N.Net, Gauss/Joint BC, Gauss Naive BC, N.Neigh, Bayes Net Based BC, Cascade Correlation
	Joint DE, Naive DE, Gauss/Joint DE, Gauss Naive DE, Bayes Net Structure Learning
	Linear Regression, Polynomial Regression, Perceptron, Neural Net, N.Neigh, Kernel, LWR, RBFs, Robust Regression, Cascade Correlation, Regression Trees, GMDH, Multilinear Interp, MARS

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 88

What You Should Know

- For each of the eight methods you should be able to summarize briefly what they do and outline how they work.
- You should understand them well enough that given access to the notes you can quickly re-understand them at a moments notice
- But you don't have to memorize all the details
- In the right context any one of these eight might end up being really useful to you one day! You should be able to recognize this when it occurs.

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 89

Citations

Radial Basis Functions

T. Poggio and F. Girosi, Regularization Algorithms for Learning That Are Equivalent to Multilayer Networks, *Science*, 247, 978--982, 1989

LOESS

W. S. Cleveland, Robust Locally Weighted Regression and Smoothing Scatterplots, *Journal of the American Statistical Association*, 74, 368, 829-836, December, 1979

GMDH etc

<http://www.inf.kiev.ua/GMDH-home/>

P. Langley and G. L. Bradshaw and H. A. Simon, Rediscovering Chemistry with the BACON System, *Machine Learning: An Artificial Intelligence Approach*, R. S. Michalski and J. G. Carbonell and T. M. Mitchell, Morgan Kaufmann, 1983

Regression Trees etc

L. Breiman and J. H. Friedman and R. A. Olshen and C. J. Stone, *Classification and Regression Trees*, Wadsworth, 1984

J. R. Quinlan, Combining Instance-Based and Model-Based Learning, *Machine Learning: Proceedings of the Tenth International Conference*, 1993

Cascade Correlation etc

S. E. Fahlman and C. Lebiere. The cascade-correlation learning architecture. Technical Report CMU-CS-90-100, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1990.
<http://citeseer.nj.nec.com/fahlman91cascadecorrelation.html>

J. H. Friedman and W. Stuetzle, Projection Pursuit Regression, *Journal of the American Statistical Association*, 76, 376, December, 1981

MARS

J. H. Friedman, Multivariate Adaptive Regression Splines, Department for Statistics, Stanford University, 1988, Technical Report No. 102

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 90