**8:** Polynomial Regression

So far we’ve mainly been dealing with linear regression

**X**= **y**=

:

***x****1=(3,2)..*

**y**= 7

*y1=7..*

*******=(****Z****T****Z****)-1(****Z****T****y****)*

***z****1=(1,3,2)..*

***z****k=(1,xk1,xk2)*

Copyright © 2001, Andrew W. Moore

*y1=7..*

*yest = 0+ 1 x1+ 2 x2*

Machine Learning Favorites: Slide 2

:

3

:

3

7

:

:

1

1

1

1

2

3

**Z**=

:

:

3

1

1

7

2

3

*Y*

*X2*

*X1*

Oct 29th, 2001

Copyright © 2001, Andrew W. Moore

**Carnegie Mellon University**

[www.cs.cmu.edu/~awm](http://www.cs.cmu.edu/%7Eawm) [awm@cs.cmu.edu](mailto:awm@cs.cmu.edu)

412-268-7599

Comments and corrections gratefully received.

[http://www.cs.cmu.edu/~awm/tutorials .](http://www.cs.cmu.edu/%7Eawm/tutorials) **School of Computer Science**

Andrew’s tutorials:

**algorithms**

**Andrew W. Moore Associate Professor**

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

**Eight more Classic Machine Learning**

|  |  |
| --- | --- |
| 3 | 2 |
| 1 | 1 |
| : | : |

vial to do linear fits of fixed nonlinear basis functions

*Y* **X**= 3 2 **y**= 7

7 1 1 3

3 : : :

: ***x****1=(3,2).. y1=7..*

3 2 9 6 4 **y**=

1 1 1 1 1 7

: 3 *******=(****Z****T****Z****)-1(****Z****T****y****)*

:

*x1, x2 , x1 , x1x2,x2 ,) y =  +  x +  x +*

*est*

*2 2 0 1 1 2 2*

* x 2 +  x x +  x 2*

Quadratic Regression

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

**X**= **y**=

*(3,2)..*

**y**=

*y1=7..*

*******=(****Z****T****Z****)-1(****Z****T****y****)*

***z****=(1 , x , x , x 2, x x ,x 2 )*

*1 2 1 1 2 2 ,*

*yest = 0+ 1 x1+ 2 x2+*

* x 2 +  x x +  x 2*

*3 1*

*4 1 2 5 2*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 3

:

3

7

:

3

7

|  |  |
| --- | --- |
| 3 | 2 |
| 1 | 1 |
| : | : |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X1* | *X2* | *Y* |  | | | |
| 3 | 2 | 7 |
| 1 | 1 | 3 |
| : | : | : |
|  |  | 2 | 9 | ***x****1=*  6 | 4 |
| **Z**= | 1 | 3 |
| 1 | 1 | 1 | 1 | 1 | 1 |
| : |  |  |  |  | : |

Machine Learning Favorites: Slide 4

Copyright © 2001, Andrew W. Moore

*5 2*

*4 1 2*

*3 1*

***z****=(1 ,* Note that solving *******=(****Z****T****Z****)-1(****Z****T****y****)* is thus *O(m6)*

(m+2)-choose-2 terms in total *= O(m2)*

:

**Z**= 1 •m linear terms

1 •(m+1)-choose-2 = m(m+1)/2 quadratic terms

* 1 constant term

:

*X2* Each column of the Z matrix is called a term column

2 How many terms in a quadratic regression with *m*

1 inputs?

*X1*

3

1

:

Each component of a z vector is called a term.

It’s tri

Quadratic Regression

Qth-degree polynomial Regression

**X**=

**y**=

*(3,2)..*

**y**=

*y1=7..*

*******=(****Z****T****Z****)-1(****Z****T****y****)*

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

*yest = 0+*

*1 x1+…*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 5

:

3

7

:

3

7



m inputs, degree Q: how many terms?

= the number of unique terms of the*m* form

*x x* ...*x* where *qi*  *Q*

*q q qm*

1 2

1 2 *m*

= the number of unique terms of the *m*form

*i* 1

1 *x x* ...*x* where *qi*  *Q*

*q q q qm*

0 1 2

1 2 *m*

= the number of lists of non-negative integers *[q ,q ,q ,..q ]*

*i* 0

in which *qi = Q*

*0 1 2 m*

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

Q=11, m=4

*q0=2 q1=2 q2=0*

*q3=4*

Copyright © 2001, Andrew W. Moore

*q4=3*

Machine Learning Favorites: Slide 6

|  |  |
| --- | --- |
| 3 | 2 |
| 1 | 1 |
| : | : |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X1* | *X2* | *Y* |  | | | |
| 3 | 2 | 7 |
| 1 | 1 | 3 |
| : | : | : |
|  |  | 2 | 9 | ***x****1=*  6 | … |
| **Z**= | 1 | 3 |
| 1 | 1 | 1 | 1 | 1 | … |
| : |  |  |  |  | … |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

**7:** Radial Basis Functions (RBFs)

**X**=

**y**=

*(3,2)..*

**y**=

*y1=7..*

*******=(****Z****T****Z****)-1(****Z****T****y****)*

***z****=(list of radial basis function evaluations)*

*yest = 0+*

*1 x1+…*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 7

:

3

7

:

3

7

|  |  |
| --- | --- |
| 3 | 2 |
| 1 | 1 |
| : | : |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X1* | *X2* | *Y* |  | | | |
| 3 | 2 | 7 |
| 1 | 1 | 3 |
| : | : | : |
|  |  | … | … | ***x****1=*  … | … |
| **Z**= | … | … |
| … | … | … | … | … | … |
| … | … | … | … | … | … |

Machine Learning Favorites: Slide 8

Copyright © 2001, Andrew W. Moore

*yest = 1 1(x) + 2 2(x) + 3 3(x)*

where

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

*c1*

*c1*

*c1*

x

y

1-d RBFs

Machine Learning Favorites: Slide 9

Copyright © 2001, Andrew W. Moore

*yest = 1(x) +  2(x) + 3(x)*

where

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

*c1*

*c1*

*c1*

x

y

Example

Machine Learning Favorites: Slide 10

Copyright © 2001, Andrew W. Moore

*yest = 1(x) +  2(x) + 3(x)*

where

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

overlap between basis fun*c1*ctions\*

\*Usually much better than the crappy overlap on my diagram

*c1*

on a ~~g~~*c*~~r~~*~~1~~*~~id in~~ m-

dimensioxnal input space)

(initialized randomly or

Ally*ci* ’s are held constant enough that there’s decent

RBFs with Linear Regression

*KW* also held constant (initialized to be large

Machine Learning Favorites: Slide 11

Copyright © 2001, Andrew W. Moore

*yest = 1(x) +  2(x) + 3(x)*

where

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

then given *Q* basis functions, define the matrix *Z* such that *Zkj* =

*KernelFunction(* | *xk - ci* | */ KW)* where *xk* is the kth vector of inputs

And as before, *******=(****Z****T* ***Z****)-1(****Z****T****y****)*

overlap between basis

fun*c1*ctions\*

\*Usually much better than the crappy overlap on my diagram

*c1*

on a g*c*r*1*id in m-

dimensioxnal input space)

Ally*ci* ’s are held constant (initialized randomly or

RBFs with Linear Regression

*KW* also held constant (initialized to be large enough that there’s decent

Machine Learning Favorites: Slide 12

Copyright © 2001, Andrew W. Moore

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

But how do we now find all the *j*’s, *ci* ’s and *KW* ?

*KW* allowed to adapt to the data.

(Some folks even let each basis function have its own *KWj*,permittin*c*g*1* fine detail in dense regions of input space)

*1*

space)

*yest = 1(x) +  2(x) + 3(x)*

where

m-dimxens*1*ional input

randomly o*c*r on a grid in*c*

ythe data (initialized

Allow the *ci* ’s to adapt to

RBFs with NonLinear Regression

Machine Learning Favorites: Slide 13

Copyright © 2001, Andrew W. Moore

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

But how do we now find all the *j*’s, *ci* ’s and *KW* ?

Answer: Gradient Descent

function have its own *KWj*,permittin*c*g*1* fine detail in dense regions of input space)

*1*

space)

*yest = 1(x) +  2(x) + 3(x)*

where

m-dimxens*1*ional input

randomly o*c*r on a grid in*c*

*KW* allowed to adapt to the data.

ythe data (initialized (Some folks even let each basis

Allow the *ci* ’s to adapt to

RBFs with NonLinear Regression

Machine Learning Favorites: Slide 14

*i*

descent while the *j*’s use matrix inversion)

Copyright © 2001, Andrew W. Moore

hybrid, where the *c* ’s and *KW* are updated with gradient Answer: Gradient Descent

(But I’d like to see, or hope someone’s already done, a

*i(x) = KernelFunction(* | *x - ci* | */ KW)*

But how do we now find all the *j*’s, *ci* ’s and *KW* ?

*KW* allowed to adapt to the data.

(Some folks even let each basis function have its own *KWj*,permittin*c*g*1* fine detail in dense regions of input space)

*1*

space)

*yest = 1(x) +  2(x) + 3(x)*

where

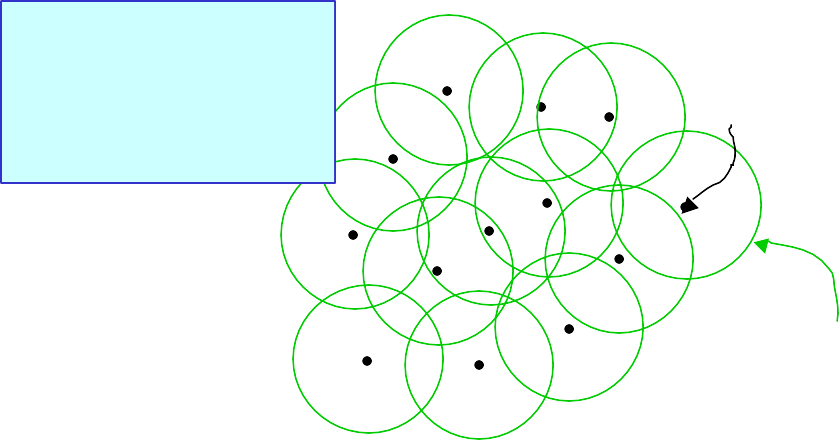
m-dimxens*1*ional input

randomly o*c*r on a grid in*c*

ythe data (initialized

Allow the *ci* ’s to adapt to

RBFs with NonLinear Regression



Machine Learning Favorites: Slide 15

x1

Copyright © 2001, Andrew W. Moore

Sphere of significant influence of center

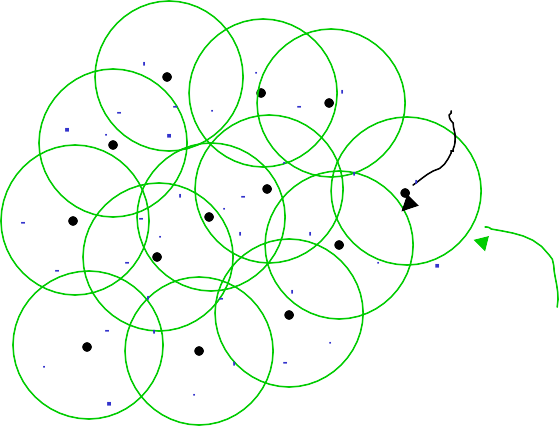
x2

Center

sticking out of page) not shown.

Radial Basis Functions in 2-d

Two inputs. Outputs (heights



Machine Learning Favorites: Slide 16

x1

Copyright © 2001, Andrew W. Moore

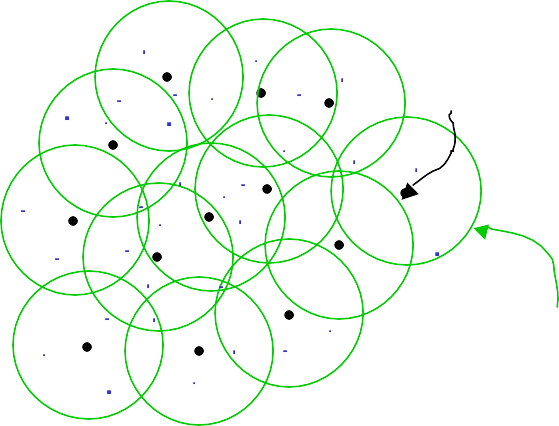
Sphere of significant influence of center

x2

Center

Blue dots denote coordinates of input vectors

Happy RBFs in 2-d



Machine Learning Favorites: Slide 17

x1

Copyright © 2001, Andrew W. Moore

Sphere of significant influence of center

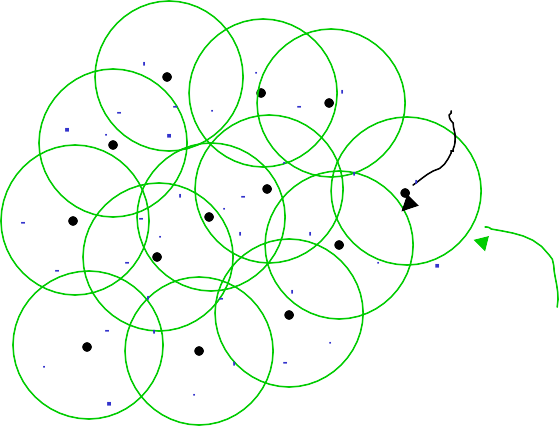
Center

x2

What’s the problem in this example?

Crabby RBFs in 2-d

Blue dots denote coordinates of input vectors



Machine Learning Favorites: Slide 18

x1

Copyright © 2001, Andrew W. Moore

Sphere of significant influence of center

Center

x2

And what’s the problem in this example?

More crabby RBFs

Blue dots denote coordinates of input vectors

Machine Learning Favorites: Slide 19

x1

Copyright © 2001, Andrew W. Moore

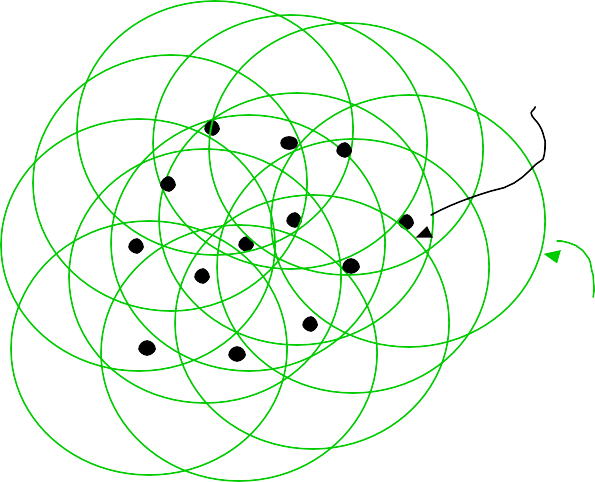
Sphere of significant influence of center

x2

Center

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

Hopeless!



Machine Learning Favorites: Slide 20

x1

Copyright © 2001, Andrew W. Moore

Sphere of significant influence of center

x2

Center

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

Unhappy

Machine Learning Favorites: Slide 21

x

Copyright © 2001, Andrew W. Moore

y

**6:** Robust Regression

Machine Learning Favorites: Slide 22

x

Copyright © 2001, Andrew W. Moore

y

Robust Regression

This is the best fit that Quadratic Regression can manage

Machine Learning Favorites: Slide 23

x

Copyright © 2001, Andrew W. Moore

y

Robust Regression

…but this is what we’d probably prefer



Machine Learning Favorites: Slide 24

x

Copyright © 2001, Andrew W. Moore

You are a very good datapoint.

y

LOESS-based Robust Regression

After the initial fit, score each datapoint according to how well it’s fitted…



You are not too shabby.

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 25

You are a very good datapoint.

y

LOESS-based Robust Regression

After the initial fit, score each datapoint according to how well it’s fitted…



You are not too shabby.

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 26

You are a very good datapoint.

y

LOESS-based Robust Regression

After the initial fit, score each datapoint according to

But you are how well it’s fitted…

pathetic.

Machine Learning Favorites: Slide 27

Copyright © 2001, Andrew W. Moore

*k*

*wk = KernelFn([yk- yest ]2)*

*yk*

* Let *wk* be a weight for datapoint k that is large if the datapoint fits well and small if it fits badly:

x

*k*

* Let *yest* be predicted value of

y

Robust Regression

For k = 1 to R…

* Let *(xk,yk)* be the kth datapoint

Machine Learning Favorites: Slide 28

Copyright © 2001, Andrew W. Moore

*est 2*

*wk = KernelFn([yk- y k] )*

based” lecture (only then the weights depended on distance in input-space)

Guess what happens next?

using weighted datapoints. datapoint fits well and small if it

I taught you how to do this in the “Instance- fits badly:

* Let *wk* be a weight for datapoint k that is large if the

*yk*

Then redo the regression

x

*k*

* Let *yest* be predicted value of

y

Robust Regression

For k = 1 to R…

* Let *(xk,yk)* be the kth datapoint

Machine Learning Favorites: Slide 29

Copyright © 2001, Andrew W. Moore

*est 2*

*wk = KernelFn([yk- y k] )*

based” lecture (only then the weights depended on distance in input-space)

Repeat whole thing until converged!

using weighted datapoints. datapoint fits well and small if it

I taught you how to do this in the “Instance- fits badly:

* Let *wk* be a weight for datapoint k that is large if the

*yk*

Then redo the regression

x

*k*

* Let *yest* be predicted value of

y

Robust Regression

For k = 1 to R…

* Let *(xk,yk)* be the kth datapoint

Machine Learning Favorites: Slide 30

Copyright © 2001, Andrew W. Moore

Computational task is to find the Maximum Likelihood *0 , 1 and 2*

*k*

*yk = 0+ 1 xk+ 2 x 2 +N(0,2)*

Robust Regression---what we’re doing

**What regular regression does:**

Assume *yk* was originally generated using the following recipe:

Machine Learning Favorites: Slide 31

Copyright © 2001, Andrew W. Moore

But otherwise

*yk ~ N(,huge2)*

Computational task is to find the Maximum Likelihood *0 , 1 , 2 , p,  and huge*

*k*

*yk = 0+ 1 xk+ 2 x 2 +N(0,2)*

Robust Regression---what we’re doing

**What LOESS robust regression does:**

Assume *yk* was originally generated using the following recipe:

With probability *p*:

Machine Learning Favorites: Slide 32

Copyright © 2001, Andrew W. Moore

But otherwise

*yk ~ N(,huge2)* **E.M.**

Computational task is to find the Maximum Likelihood *0 , 1 , 2 , p,  and huge*

Your first glimpse of two spectacular letters:

*k*

reweighting procedure does this computation for us.

*k*

following recipe:

With probability *p*:

*yk = 0+ 1 xk+ 2 x 2 +N(0,2)*

Assume *y* was originally generated using the

Robust Regression---what we’re doing

**What LOESS robust regression does:**

Mysteriously, the

Machine Learning Favorites: Slide 33

Copyright © 2001, Andrew W. Moore

**5:** Regression Trees

* “Decision trees for regression”

Machine Learning Favorites: Slide 34

Copyright © 2001, Andrew W. Moore

Mean age of records matching this leaf node

**Predict age = 47**

A regression tree leaf

Choosing the attribute to split on

* We can’t use information gain.
* What should we use?

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 36

Machine Learning Favorites: Slide 35

Copyright © 2001, Andrew W. Moore

Male

Female

A one-split regression tree

**Predict age = 36**

**Predict age = 39**

Gender?

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

Children Babies

No 2 1 38

No 0 0 24

es 0 5+ 72

: : :

Choosing the attribute to split on

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 *y x=j*

Then…

*MSE*(*Y* | *X* )

1 *NX*

*R*





( *y*  *µ* )

*x* *j* 2

*k*

*y*

*j*1 (*k* such that*x*  *j* )

*k*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 37

Choosing the attribute to split on

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 *y x=j*

Then…

*MSE*(*Y* | *X* )

1 *NX*

*R*





( *y*  *µ* )

*x* *j* 2

*k*

*y*

*j*1 (*k* such that*x*  *j* )

*k*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 38

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

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Gender | Rich? | | Num. | Num. Beany | Age |  |
|  | 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 | | | | |
| Female |  |
| Male |  |
| Male | Y |
| : | : |
|  | |



Machine Learning Favorites: Slide 39

Copyright © 2001, Andrew W. Moore

Pruning Decision

…property-owner = Yes Do I deserve to live?

Gender?

Female Male

**Predict age = 36**

# property-owning females = 56712 # property-owning males = 55800 Mean age among POFs = 39 Mean age among POMs = 36

Age std dev among POFs = 12 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.

**Predict age = 39**

Split attribute chosen to minimize MSE of regressed children.

Pruning with a different Chi- squared

Machine Learning Favorites: Slide 40

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

Copyright © 2001, Andrew W. Moore

Predict age =

24 + 7 \* NumChildren -

2.5 \* YearsEducation

Predict age =

26 + 6 \* NumChildren -

2 \* YearsEducation

Male

Female

Linear Regression Trees

…property-owner = Yes Also known as “Model Trees”

Gender?

Male

Predict age =

24 + 7 \* NumChildren -

2.5 \* YearsEducation

Split attribute chosen to minimize MSE of regressed children.

Pruning with a different Chi- squared

Machine Learning Favorites: Slide 41

Female

Predict age =

26 + 6 \* NumChildren -

2 \* YearsEducation

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

Copyright © 2001, Andrew W. Moore

Linear Regression Trees

…property-owner = Yes Also known as “Model Trees”

Gender?

Machine Learning Favorites: Slide 42

x

Copyright © 2001, Andrew W. Moore

y

Test your understanding

Assuming regular regression trees, can you sketch a graph of the fitted function *yest(x)* over this diagram?

Machine Learning Favorites: Slide 43

x

Copyright © 2001, Andrew W. Moore

y

Test your understanding

Assuming linear regression trees, can you sketch a graph of the fitted function *yest(x)* over this diagram?

Machine Learning Favorites: Slide 44

Copyright © 2001, Andrew W. Moore

**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

o linear regression

se cross-validation to discover whether an uadratic term is good. If so, add it as a ba

Machine Learning Favorites: Slide 45

Copyright © 2001, Andrew W. Moore

1. 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.
2. Else stop

4.3 income / (cos (NumCars))

function and loop.

y sis

q

GMDH (c.f. BACON, AIM)

Group Method Data Handling

A very simple but very good idea: D Typical learned function:

U ageest = height - 3.1 sqrt(weight) +

•

• 1.

2.

Machine Learning Favorites: Slide 46

Copyright © 2001, Andrew W. Moore

**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

Cascade beginning

Begin with a regular dataset

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 47

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

Cascade first step

Begin with a regular dataset

Find weights *w(0)* to best fit Y.

*i*

I.E. to minimize

*R*

*m*

*k*

*k*

*k*

 *j k*

*k* 1 *j* 1

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 48

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* |
| : | : | : | : | : |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* |
|  |  |  |  |  |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* |
| : | : | : | : | : |

Consider our errors…

Begin with a regular dataset

Find weights *w(0)* to best fit Y.

*i*

I.E. to minimize

*R*

*m*

*k*

*k*

*k*  *j k*

*k* 1 *j* 1

Define *e*(0)  *y*  *y*(0)

*k*

*k k*

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* (**x**)  *g u x*

(0 )

 *m*



 *j* 1

 *j*

(0 ) ( *j*) 





Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 50

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* | *Y(0)* | *E(0)* |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* | *y 1*  *(0)* | *e 1*  *(0)* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* | *y 2*  *(0)* | *e 2*  *(0)* |
| : | : | : | : | : | : | : |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* | *Y(0)* | *E(0)* | *H(0)* |
|  |  |  |  |  |  |  |  |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* | *y 1*  *(0)* | *e 1*  *(0)* | *h 1*  *(0)* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* | *y 2*  *(0)* | *e 2*  *(0)* | *h 2*  *(0)* |
| : | : | : | : | : | : | : | : |

Cascade next step

Find weights *w(1) p(1)* to better fit Y.

*i 0*

I.E. to minimize

*R m*

*k*

*k*

*k*

 *j*

*k*

*j k*

*k* 1

*j*1

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 51

Now look at new errors

Find weights *w(1) p(1)* to better fit Y.

*i 0*

Define*e*(1)  *y*  *y*(1)

*k*

*k k*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 52

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* | *Y(0)* | *E(0)* | *H(0)* | *Y(1)* |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* | *y 1*  *(0)* | *e 1*  *(0)* | *h 1*  *(0)* | *y 1*  *(1)* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* | *y 2*  *(0)* | *e 2*  *(0)* | *h 2*  *(0)* | *y 2*  *(1)* |
| : | : | : | : | : | : | : | : | : |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* | *Y(0)* | *E(0)* | *H(0)* | *Y(1)* | *E(1)* |
|  |  |  |  |  |  |  |  |  |  |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* | *y 1*  *(0)* | *e 1*  *(0)* | *h 1*  *(0)* | *y 1*  *(1)* | *e 1*  *(1)* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* | *y 2*  *(0)* | *e 2*  *(0)* | *h 2*  *(0)* | *y 2*  *(1)* | *e 2*  *(1)* |
| : | : | : | : | : | : | : | : | : | : |

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* (**x**)  *g u x*  *v h*

(1)

 *m*



 *j*

(1) ( *j* ) (1) (0 ) 

 *j*1

0





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

*R*

*m*

*n* 1

*k k*

*k*

 *j k*

 *j k*

*k* 1 *j* 1 *j* 1

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 54

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *X(0)* | *X(1)* | … | *X(m)* | *Y* | *Y(0)* | *E(0)* | *H(0)* | *Y(1)* | *E(1)* | *H(1)* |
| *x 1*  *(0)* | *x 1*  *(1)* | … | *x 1*  *(m)* | *y1* | *y 1*  *(0)* | *e 1*  *(0)* | *h 1*  *(0)* | *y 1*  *(1)* | *e 1*  *(1)* | *h 1*  *(1)* |
| *x 2*  *(0)* | *x 2*  *(1)* | … | *x 2*  *(m)* | *y2* | *y 2*  *(0)* | *e 2*  *(0)* | *h 2*  *(0)* | *y 2*  *(1)* | *e 2*  *(1)* | *h 2*  *(1)* |
| : | : | : | : | : | : | : | : | : | : | : |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| *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* | *y1* | *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* | *y2* | *y(0)2* | *e(0)2* | *h(0)2* | *y(1)2* | *e(1)2* | *h(1)2* | *…* | *y(n)2* |
| : | : | : | : | : | : | : | : | : | : | : | : | : |

Now look at new errors

Find weights *w(n)i p(n)j* to better fit Y.

I.E. to minimize

Define*e*(*n*)  *y*  *y*(*n*)

*k*

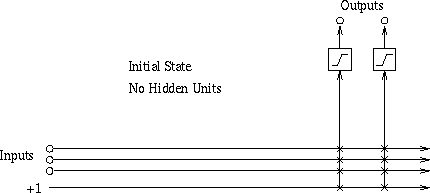
*k k*

Copyright © 2001, Andrew W. Moore

Machine Learning Favorites: Slide 55

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| *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* | *y1* | *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* | *y2* | *y(0)2* | *e(0)2* | *h(0)2* | *y(1)2* | *e(1)2* | *h(1)2* | *…* | *y(n)2* | *e(n)2* |
| : | : | : | : | : | : | : | : | : | : | : | : | : | : |

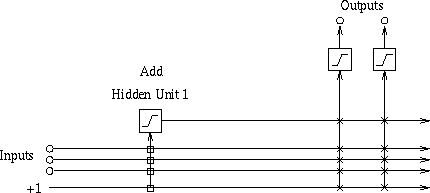
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Create n’th hidden unit…  Find weights *u(n)i v(n)i* to define a new basis function *H(n)(x)* of the inputs.  Make it specialize in predicting the errors in our previous fit:  Find {*u(n)i , v(n)j*} to maximize correlation between *H(n)(x) and E(n)* where  *m n* 1    *j*  *j*    *j*1 *j* 1  | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| *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* | *y1* | *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* | *y2* | *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…*  Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 56 | | | | | | | | | | | | | | | |



Machine Learning Favorites: Slide 57

Copyright © 2001, Andrew W. Moore

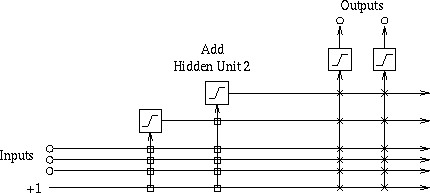
Visualizing first iteration



Machine Learning Favorites: Slide 58

Copyright © 2001, Andrew W. Moore

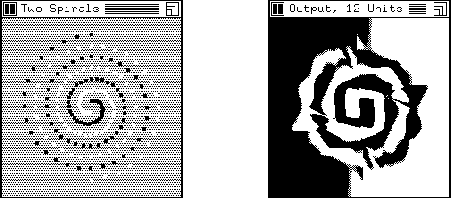
Visualizing second iteration



Machine Learning Favorites: Slide 59

Copyright © 2001, Andrew W. Moore

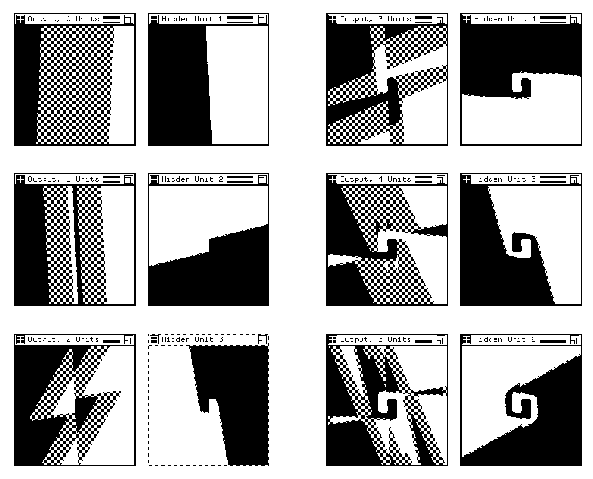
Visualizing third iteration



Machine Learning Favorites: Slide 60

Copyright © 2001, Andrew W. Moore

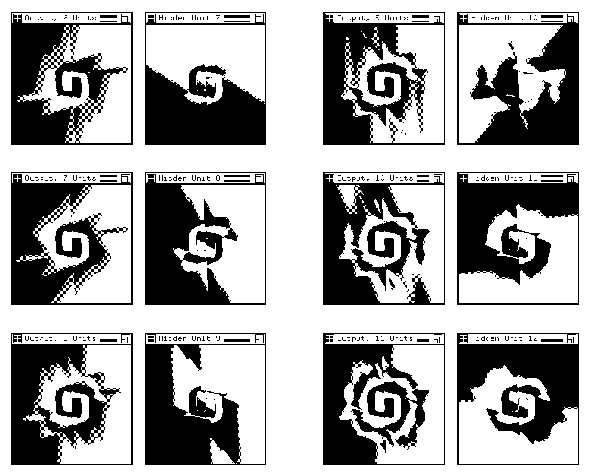
Example: Cascade Correlation for Classification



Machine Learning Favorites: Slide 61

Copyright © 2001, Andrew W. Moore

Training two spirals: Steps 1-6



Machine Learning Favorites: Slide 62

Copyright © 2001, Andrew W. Moore

Training two spirals: Steps 2-12

Machine Learning Favorites: Slide 63

Copyright © 2001, Andrew W. Moore

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

Machine Learning Favorites: Slide 64

x

Copyright © 2001, Andrew W. Moore

y

**2:** Multilinear Interpolation

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

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 65

*q5*

*q4*

*q3*

*q2*

*q1*

y

Multilinear Interpolation

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

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 66

*q5*

*q4*

*q3*

*q2*

*q1*

y

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)

How to find the best fit?

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

*y* (*x*) 

*est*

(*q*  *x*)

3

*h* 

(*q*  *x*)

2

*w*

2

*w*

*h* where *w*  *q*  *q*

3

3 2

y

*q1*

*q2*

*q3*

*q4*

*q5*

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 68

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 67

*q5*

*q4*

*q3*

*q2*

*q1*

y

How to find the best fit?

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

*What’s the problem with this idea?*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *h2*  *h3* |  |  |  |  |
|  |
|  |  |

x

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 69

*q5*

*q4*

*q3*

*q2*

*q1*

*2(x)*

*h3*

y

*h2*

*w*

3

2

3

*q*  *x*

, *f* (*x*)  1

2

*x*  *q*

*w*

3 3

where *f* ( *x*)  1

2 2

*yest* (*x*)  *h f* (*x*)  *h f* (*x*)

In the red segment…

How to find the best fit?



Machine Learning Favorites: Slide 70

*3*

* (x)*

x

Copyright © 2001, Andrew W. Moore

*5*

*q*

*q4*

*q3*

*q2*

*q1*

*2(x)*

*h3*

y

*h2*

*w*

3

3

, *f* (*x*)  1

2

*q*  *x*

2

*x*  *q*

*w*

3 3

where *f* ( *x*)  1

2 2

*yest* (*x*)  *h f* (*x*)  *h f* (*x*)

In the red segment…

How to find the best fit?



Machine Learning Favorites: Slide 71

*3*

*5*

* (x)*

x

Copyright © 2001, Andrew W. Moore

*q*

*q4*

*q3*

*q2*

*q1*

*2(x)*

*h3*

y

*h2*

*w*

3

2

3

, *f* (*x*)  1

*w*

| *x*  *q* |

2

| *x*  *q* |

3 3

where *f* (*x*)  1

2 2

*yest* (*x*)  *h f* (*x*)  *h f* (*x*)

In the red segment…

How to find the best fit?



Machine Learning Favorites: Slide 72

*3*

*5*

* (x)*

x

Copyright © 2001, Andrew W. Moore

*q*

*q4*

*q3*

*q2*

*q1*

*2(x)*

*h3*

y

*h2*

*w*

3

3

, *f* (*x*)  1

*w*

2

| *x*  *q* |

2

| *x*  *q* |

3 3

where *f* (*x*)  1

2 2

*yest* (*x*)  *h f* (*x*)  *h f* (*x*)

In the red segment…

How to find the best fit?



Machine Learning Favorites: Slide 73

*3*

* (x)*

x

Copyright © 2001, Andrew W. Moore

*5*

*q*

*q4*

*q3*

*q2*

*q1*

*2(x)*

*h3*

y

*h2*

*i*

otherwise

*w*

0

if | *x*  *q* |*w*

1 | *x*  *qi* |

where *f i* (*x*)  



*i* 1

*i i*

*NK*



*est*

*y* (*x*)  *h f* (*x*)

In ***general***

How to find the best fit?



*3*

Machine Learning Favorites: Slide 74

*5*

*q*

* (x)*

x

Copyright © 2001, Andrew W. Moore

*q4*

*q3*

*q2*

*q1*

*2(x)*

We know how to find the least squares *hii*s!

*h3*

y

And this is simply a basis function regression problem!

*h2*

*i*

otherwise

*w*

0

if | *x*  *q* |*w*

1 | *x*  *qi* |

where *f i* (*x*)  



*i* 1

*i i*

*NK*



*est*

*y* (*x*)  *h f* (*x*)

In ***general***

How to find the best fit?

Machine Learning Favorites: Slide 75

x1

Copyright © 2001, Andrew W. Moore

x2

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

In two dimensions…

Machine Learning Favorites: Slide 76

x1

Copyright © 2001, Andrew W. Moore

x2

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

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

In two dimensions…

Machine Learning Favorites: Slide 77

x1

Copyright © 2001, Andrew W. Moore

x2

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?

8

7

3

9

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

In two dimensions…

Machine Learning Favorites: Slide 78

x1

Copyright © 2001, Andrew W. Moore

x2

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?

8

7

3

9

vectors (outputs not depicted)

locavtaiolunes ohferinep…ut

In two dimensions…

BluTeodoptrsedshicotwthe

Machine Learning Favorites: Slide 79

x1

Copyright © 2001, Andrew W. Moore

7.33

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?

8

7

To predict the value here…

First interpolate its value on two opposite edges…

x2

3

7

9

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

In two dimensions…

Machine Learning Favorites: Slide 80

7.33

x1

Copyright © 2001, Andrew W. Moore

two valuexs 2

between those

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?

8

7

7.05

3

7

9

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

To predict the value here… First interpolate

its value on two

opposite edges…

Then interpolate

In two dimensions…

8 surface is

continuous?

.33

It should be easy to see that it ensures continuity

x1 The patches are not linear

Copyright © 2001, Andrew W. Moore Machine Learning Favorites: Slide 81

This can easily be generalized

7 to *m* dimensions.

two valuexs 2

between those

Notes:

7

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

7.05

3

7

9

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

To predict the value here… First interpolate

its value on two

opposite edges…

Then interpolate

In two dimensions…

Machine Learning Favorites: Slide 82

Copyright © 2001, Andrew W. Moore

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?

1

x

x2

8

7

3

9

Doing the regression

Machine Learning Favorites: Slide 83

Copyright © 2001, Andrew W. Moore

*m*

 *k k*

*k* 1

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

*est*

*y* (**x**)  *g* (*x* )

**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:

Machine Learning Favorites: Slide 84

x

Copyright © 2001, Andrew W. Moore

*q5*

*q4*

*q3*

*q2*

*q1*

y

Idea: Each *gk* is one of these

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

*k* 1

*k k*

*m*

*y est* (**x**)   *g* (*x* )

MARS

y

Machine Learning Favorites: Slide 85

x

Copyright © 2001, Andrew W. Moore

*qkj* : The location of the j’th knot in the k’th dimension

*hkj* : The regressed height of the j’th knot in the k’th dimension

*wk*: The spacing between knots in

*q5* the kth dimension

*q4*

*q3*

*q2*

*q1*

otherwise

0



*k*

*k j k*

if | *x*  *qk* |*w*

*w*

*k j*

1 





*j*



*k*

where *f* ( *x*) 

*k* 1 *j* 1

 | *x*  *qk* |

*k k*

*h f* (*x* )

*NK*

*m*

 *j j k*

*est*

*y* (**x**) 

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

*k* 1

*k k*

*m*

*y est* (**x**)   *g* (*x* )

MARS

Machine Learning Favorites: Slide 86

Copyright © 2001, Andrew W. Moore

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.

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

*k* 1 *t* *k* 1

*k* 1

*kt k t*

*g* (*x* , *x* )

*m m*

 

*k k*

*m*



*est*

*y* (**x**)  *g* (*x* ) 

That’s not complicated enough!

* Okay, now let’s get serious. We’ll allow arbitrary “two-way interactions”:

Where are we now?

Inference Joint DE, Bayes Net Structure Learning

Engine Learn P(E1|E2)

Predict category

Prob- ability

Dec Tree, Sigmoid Perceptron, Sigmoid N.Net, Gauss/Joint BC, Gauss Naïve BC, N.Neigh, Bayes Net Based BC, Cascade Correlation

Joint DE, Naïve DE, Gauss/Joint DE, Gauss Naïve DE, Bayes Net Structure Learning

Predict real no.

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

Regressor

Classifier

Inputs Inputs Inputs Inputs

Machine Learning Favorites: Slide 87

Copyright © 2001, Andrew W. Moore

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)

|  |  |  |
| --- | --- | --- |
|  | Density Estimator |  |
|  |
|  |  |

Machine Learning Favorites: Slide 89

Copyright © 2001, Andrew W. Moore

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.

Machine Learning Favorites: Slide 90

Copyright © 2001, Andrew W. Moore

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/fahlman91cascadec> orrelation.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

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. Carbonnell and T. M. Mitchell, Morgan Kaufmann, 1983

Citations