Predict

bottom-up prediction

……… learning, least-squares and function approximation

…………. prediction, optimization and control

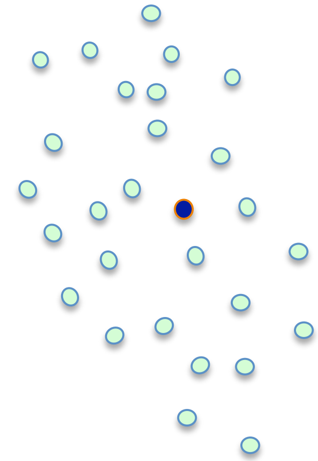
………………… hierarchical temporal memory: prediction

……………………… top-down/bottom-up blackboard architecture

…………………………. web-intelligence; brains; adaptive BI

………………………………. challenge problems

learning and prediction



m data points each having (i) *features* $x_1 \ldots x_{n-1} = \mathbf{x}$ and (ii) *output variable(s)* $y_1 .. y_k$.

e.g. *prices* (numbers for Y); x_i can be numbers or categories

for now assume k=1, i.e. just one output variable y

linear prediction:

$f(\mathbf{x}) = E[y|\mathbf{x}]$ also minimizes*:

$\varepsilon = E[\text{error}] = E[y-f(\mathbf{x})]^2 \approx \frac{1}{m}\Sigma_m(y_i-f(\mathbf{x_i}))^2$

suppose $f(\mathbf{x}) = [\mathbf{x};1]^T\mathbf{f} = \mathbf{x'}^T\mathbf{f}$

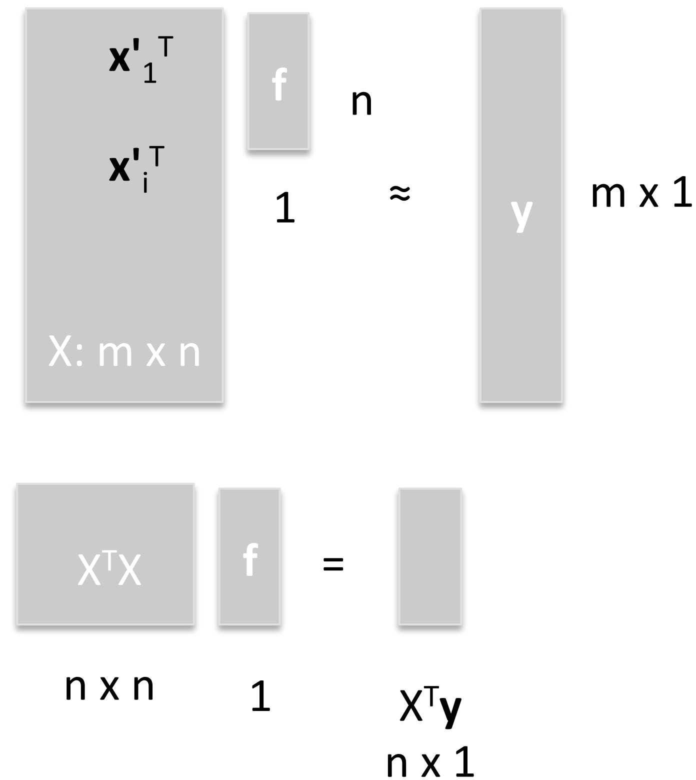
i.e. *linear* in **x**; so we want $X\,\mathbf{f} \approx \mathbf{y}$

$\Sigma_m(y_i - \mathbf{x'_i}^T\mathbf{f})^2 = (X\,\mathbf{f} - \mathbf{y})^T (X\,\mathbf{f} - \mathbf{y})$

minimized if derivative = 0, i.e.

$X^TX\,\mathbf{f} - X^T\mathbf{y}$ .. "normal equations"

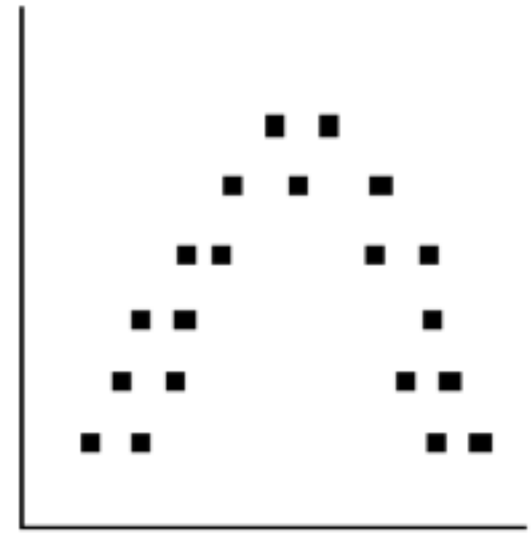
once we have **f**, our "least-squares" estimate of $y|\mathbf{x}$ is $f^{LS}(\mathbf{x}) = \mathbf{x'}^T\mathbf{f}$



some examples

x	y
10	1.2
22	1.8
42	4.6
15	1.3

$$\begin{array}{ccc} X & f & y \\ \begin{bmatrix} 10 & 1 \\ 22 & 1 \\ 42 & 1 \\ 15 & 1 \end{bmatrix} & \begin{bmatrix} 0.11 \\ -0.26 \end{bmatrix} \approx & \begin{bmatrix} 1.2 \\ 1.8 \\ 4.6 \\ 1.3 \end{bmatrix} \end{array}$$



how good is the ‘fit’ ? $R^2 \equiv 1 - \frac{\sum_i (f^T x_i - y_i)^2}{\sum_i (\bar{y} - y_i)^2}$ = .95

example 2*:

[y, **x**] = [wine-quality, winter-rainfall, avg-temp, harvest-rainfall]

$f^{LS}(\mathbf{x})$ = 12.145 + 0.00117 × winter-rainfall + 0.0614 × avg-temperature − 0.00386 × harvest rainfall

*Super-crunchers, Ian Aryes 2007: Orley Ashenfelter

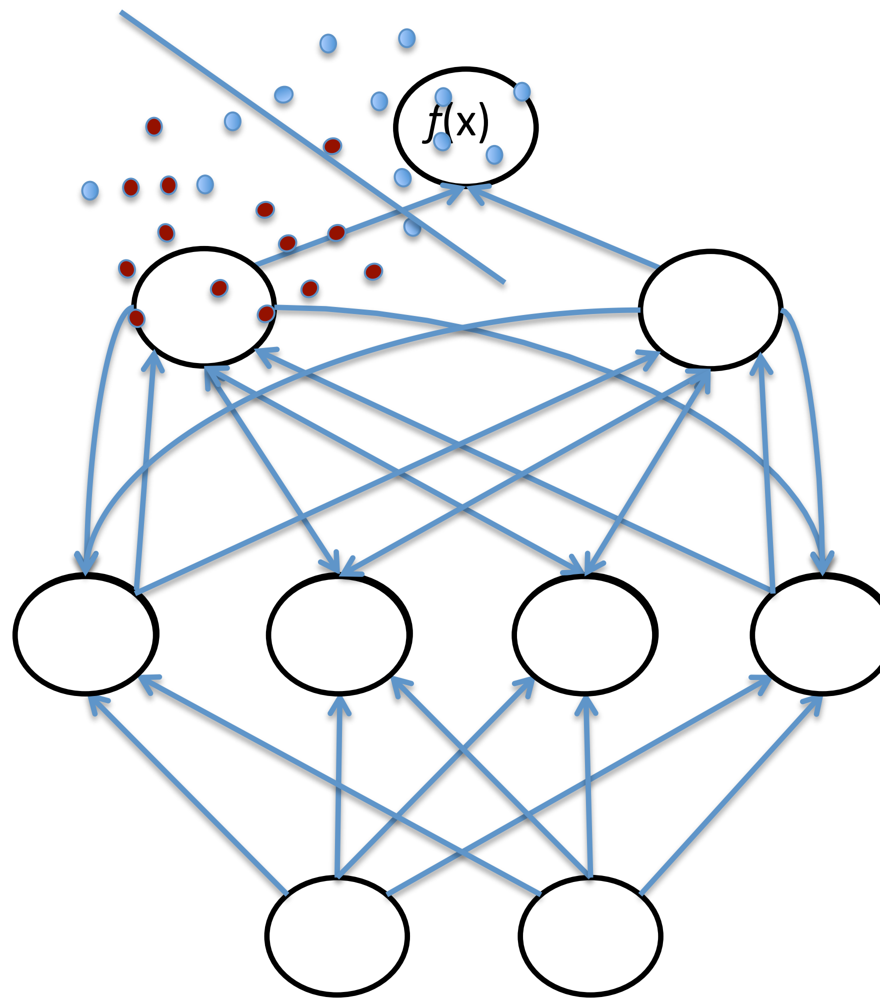
beyond least-squares

categorical data

- logistic regression
- support-vector-machines
  - complex f :
    - 'kernel'-parameters also learn

neural networks

- linear = least-squares
- non-linear
  - like logistic etc.
- feed-forward, multi-layer
  - more complex f
- feed-back
  - like a belief n/w;
    - "explaining-away" effect



deep-belief network

learning parameters

whatever be the model:

need to minimize $|f(x) - y| = \varepsilon(\mathbf{f})$

complex f => no formula

so, iterative method ; start with $\mathbf{f}^0$

$\mathbf{f}^1 = \mathbf{f}^0 + \delta\mathbf{f}$

$f^{i+1} = f^i - \alpha\nabla_f \varepsilon(f^i)$ gradient-descent

use $\varepsilon(\mathbf{f}^i)-\varepsilon(\mathbf{f}^{i-1})$ to approximate derivative

works fine with numbers, i.e. x in $\mathbf{R}^n$

.. *caveats*: local minima, constraints

for categorical data:

convert to binary, i.e. $\{0,1\}^N$

"fuzzyfication": convert to $\mathbf{R}^n$

neighborhood-search; heuristic search, genetic algorithms ..

probabilistic models, i.e. deal with probabilities instead

related matters

"best" solution

$\mathbf{w}$: maximize $\phi(\mathbf{w})$

control actions: $\boldsymbol{\theta}^i$: $\mathbf{s}^{i+1}=S(\boldsymbol{\theta}^i)$

minimize $|\mathbf{s} - \Xi|$