From Last Meeting

Studied Fisher Linear Discrimination

- Mathematics
- “Point Cloud” view
- Likelihood view
- Toy examples
- Extensions (e.g. Principal Discriminant Analysis)

Polynomial Embedding

Aizerman, Braverman and Rozoner (1964) *Automation and Remote Control*, **15**, 821-837.

Motivating idea: extend "scope" of linear discrimination,

by adding "nonlinear components" to data

(better use of name "nonlinear discrimination"????)

E.g. In 1d, "linear separation" splits the domain

$$\{x : x \in \Re\}$$

into only 2 parts

Show PolyEmbed/Poly1Embed1d.mpg

Polynomial Embedding (cont.)

But in the "quadratic embedded domain"

$$\left\{\left(x, x^2\right) : x \in \Re\right\} \subset \Re^2$$

linear separation can give 3 parts

Show PolyEmbed/Poly1Embed2d.mpg

- original data space lies in 1d manifold
- very sparse region of $\Re^2$
- curvature of manifold gives better linear separation
- can have *any* 2 break points (2 points $\Rightarrow$ line)

Polynomial Embedding (cont.)

Stronger effects for higher order polynomial embedding:

E.g. for cubic, $\{(x, x^2, x^3) : x \in \Re\} \subset \Re^3$

linear separation can give 4 parts (or fewer)

Show PolyEmbed/Poly1Embed3d.mpg

- original space lies in 1d manifold, even sparser in $\Re^3$
- higher d curvature gives improved linear separation
- can have *any* 3 break points (3 points $\Rightarrow$ plane)?
- relatively few “interesting separating planes”

Polynomial Embedding (cont.)

General View: for original data matrix:

$$\begin{pmatrix} x_{11} & & x_{1n} \\ \vdots & \cdots & \vdots \\ x_{d1} & & x_{dn} \end{pmatrix}$$

"add rows":

$$\begin{pmatrix} x_{11} & & x_{1n} \\ \vdots & & \vdots \\ x_{d1} & & x_{dn} \\ x_{11}^2 & \cdots & x_{1n}^2 \\ \vdots & & \vdots \\ x_{d1}^2 & & x_{dn}^2 \\ x_{11}x_{21} & & x_{1n}x_{2n} \\ \vdots & & \vdots \end{pmatrix}$$

Polynomial Embedding (cont.)

Fisher Linear Discrimination: Choose Class 1 for $\underrightarrow{x}^0$ when:

$$\underrightarrow{x}^{0t}\hat{\Sigma}^{w-1}\left(\underrightarrow{\overline{X}}^{(1)}-\underrightarrow{\overline{X}}^{(2)}\right)\leq\frac{1}{2}\left(\underrightarrow{\overline{X}}^{(1)}+\underrightarrow{\overline{X}}^{(2)}\right)\hat{\Sigma}^{w-1}\left(\underrightarrow{\overline{X}}^{(1)}-\underrightarrow{\overline{X}}^{(2)}\right)$$

in *embedded* space.

- image of class boundaries in original space is *nonlinear*
- allows much more *complicated* class regions
- can also do Gaussian Likelihood Ratio (or others)

Polynomial Embedding Toy Examples

E.g. 1:   Parallel Clouds

Show PolyEmbed\PEod1Raw.ps

- PC1 always bad (finds “embedded greatest var.” only)

  show PolyEmbed\PEod1PC1combine.pdf

- FLD stays good

  show PolyEmbed\PEod1FLDcombine.pdf

- GLR OK discrimination at data, but ∃ overfitting problems

  show PolyEmbed\PEod1GLRcombine.pdf

Polynomial Embedding Toy Examples (cont.)

E.g. 2: Two Clouds

Show PolyEmbed\PEtclRaw.ps

- FLD good, generally improves with higher degree

  show PolyEmbed\PEtclFLDcombine.pdf

- GLR mostly good, some overfitting

  show PolyEmbed\PEtclGLRcombine.pdf

- $x_1, x_2, x_1^2, x_2^2, x_1 x_2$ similar in shape to x_1, x_2???

Polynomial Embedding Toy Examples (cont.)

E.g. 3: Split X

Show PolyEmbed\PEexd3Raw.ps

- FLD rapidly improves with higher degree

  show PolyEmbed\Pexd3FLDcombine.pdf

- GLR always good, but never “ellipse around blues”?

  show PolyEmbed\Pexd3GLRcombine.pdf

- Should apply ICA first?

  Show HDLSS\HDLSSxd3ICA.ps

Polynomial Embedding Toy Examples (cont.)

E.g. 4: Split X, parallel to Axes

Show PolyEmbed\Pexd4Raw.ps

- FLD fine with more embedding

  show PolyEmbed\Pexd4FLDcombine.pdf

- GLR OK for all, no overfitting.

  show PolyEmbed\Pedx4LGLRcombine.pdf

- never found “ellipse” (maybe “hyperbola” is right?)

- ICA helped FLD (better for lower degree).

Polynomial Embedding Toy Examples (cont.)

E.g. 5: Donut

Show PolyEmbed\PEdonRaw.ps

- FLD: poor for low degree, then good, no overfit

  Show PolyEmbed\ PEdonFLDcombine.pdf

- GLR: best with no embed, “square shape” for overfitting?

  Show PolyEmbed\ PEdonGLRcombine.pdf

Polynomial Embedding Toy Examples (cont.)

E.g. 6: Target

Show PolyEmbed\PEtarRaw.ps

- Similar lessons

Show PolyEmbed\PEtarFLDcombine.pdf, PolyEmbed\PEtarGLRcombine.pdf

- Hoped for better performance from cubic…

Polynomial Embedding (cont.)

Drawback to polynomial embedding:

- too many extra terms create spurious structure
- i.e. have “overfitting”
- High Dimension Low Sample Size problems worse

Kernel Machines

Idea: replace polynomials by other “nonlinear functions”

e.g. 1: “sigmoid functions” from neural nets

e.g. 2: “radial basis functions” – Gaussian kernels

Related to “kernel density estimation” (smoothed histogram)

Show SiZer\EGkdeCombined.pdf

Kernel Machines (cont.)

Radial basis functions: at some "grid points" $\underset{\rightarrow}{g}_1, \ldots, \underset{\rightarrow}{g}_k$,

For a "bandwidth" (i.e. standard deviation) σ,

Consider (d dim'al) functions: $\varphi_\sigma\left(\underset{\rightarrow}{x} - \underset{\rightarrow}{g}_1\right), \ldots, \varphi_\sigma\left(\underset{\rightarrow}{x} - \underset{\rightarrow}{g}_k\right)$

Replace data matrix with:

$$\begin{pmatrix} \varphi_\sigma\left(\underset{\rightarrow}{X}_1 - \underset{\rightarrow}{g}_1\right) & & \varphi_\sigma\left(\underset{\rightarrow}{X}_n - \underset{\rightarrow}{g}_1\right) \\ \vdots & \cdots & \vdots \\ \varphi_\sigma\left(\underset{\rightarrow}{X}_1 - \underset{\rightarrow}{g}_k\right) & & \varphi_\sigma\left(\underset{\rightarrow}{X}_n - \underset{\rightarrow}{g}_k\right) \end{pmatrix}$$

Kernel Machines (cont.)

For discrimination: work in radial basis function domain,

With new data vector $\underrightarrow{X}_0$ represented by: $\begin{pmatrix} \varphi_\sigma\left(\underrightarrow{X}_0 - \underrightarrow{g}_1\right) \\ \vdots \\ \varphi_\sigma\left(\underrightarrow{X}_0 - \underrightarrow{g}_1\right) \end{pmatrix}$

Kernel Machines (cont.)

Toy Examples:

E.g. 1: Parallel Clouds – good at data, poor outside

Show PolyEmbed\PEod1FLDe7.ps

E.g. 2: Two Clouds – Similar result

Show PolyEmbed\PEtclFLDe7.ps

E.g. 3: Split X – OK at data, strange outside

Show PolyEmbed\Pexd3FLDe7.ps

Kernel Machines (cont.)

E.g. 4: Split X, parallel to Axes – similar ideas

Show PolyEmbed\Pexd4FLDe7.ps

E.g. 5: Donut – mostly good (slight mistake for one kernel)

Show PolyEmbed\PedonFLDe7.ps

E.g. 6: Target – much better than other examples

Show PolyEmbed\PetarFLDe7.ps

Main lesson: generally good in regions with data,
unpredictable results where data are sparse

Kernel Machines (cont.)

E.g. 7: Checkerboard

Show PolyEmbed\PechbRaw.ps

- Kernel embedding is excellent

  Show PolyEmbed\PechbFLDe7.ps

- Other polynomials lack flexibility

  Show PolyEmbed\PEchbFLDcombine.pdf and PolyEmbed\PEchbGLRcombine.pdf

- Lower degree is worse

Kernel Machines (cont.)

Note: Gaussian Likelihood Ratio had frequent numerical failure

Important point for kernel machines:

High Dimension Low Sample Size problems get worse

This is motivation for “Support Vector Machines”

Kernel Machines (cont.)

$\exists$ generalizations of this idea to other types of analysis,

and some clever computational ideas.

E.g. “Kernel based, nonlinear Principal Components Analysis”

Schölkopf, Smola and Müller (1998) “Nonlinear component analysis as a kernel eigenvalue problem”, *Neural Computation*, **10**, 1299-1319.

Support Vector Machines

Classical References:

Vapnik (1982) *Estimation of dependences based on empirical data*, Springer (Russian version, 1979)

Boser, Guyon & Vapnik (1992) in *Fifth Annual Workshop on Computational Learning Theory*, ACM.

Vapnik (1995) *The nature of statistical learning theory*, Springer.

Recommended tutorial:

Burges (1998) A tutorial on support vector machines for pattern recognition, *Data Mining and Knowledge Discovery*, **2**, 955-974, see also web site:

http://citeseer.nj.nec.com/burges98tutorial.html

Support Vector Machines (cont.)

Motivation: High Dimension Low Sample Size discrimination

(e.g. from doing a nonlinear embedding)

$\exists$ a tendency towards major *over-fitting* problems

Toy Example:

In 1st dimension: Class 1: $N(2,0.8)$ Class 2: $N(-2,0.8)$

($n = 20$ of each, and threw in 4 "outliers")

In dimensions $2,...,d$: independent $N(0,1)$

Show Svm\SVMeg3p1d2m1v1.mpg

Support Vector Machines (cont.)

Toy Example: for linear discrimination:

Top: Proj'n onto (2-d) subspace generated by 1^{st} unit vector (- -) and Discrimination direction vector (----) (shows angle)

For "reproducible (over new data sets) discrimination":

Want these "near each other", i.e. small angle

Bottom: 1-d projections, and smoothed histograms

Support Vector Machines (cont.)

Lessons from Fisher Linear Discrimination Toy Example:

- Great angle for $d = 1$, but substantial overlap
- OK angle for $d = 2,...,10$, still significant overlap
- Angle gets very bad for $d = 11,...,18$, but overlap declines
- No overlap for $d \geq 23$ (perfect discrimination!?!?)
- Completely nonreproducible (with new data)
- Thus useless for real discrimination

Support Vector Machines (cont.)

Main Goal of Support Vector Machines:

Achieve a trade off between:

Discrimination quality for data at hand

vs.

Reproducibility with new data

Approaches:

1. Regularization (bound on “generaliz’n”, via “complexity”)
2. Quadratic Programming (general’n of Linear Prog.)