Kernels (SVMs, Logistic Regression, Ridge Regression)

Aarti Singh
Co-instructor: Pradeep Ravikumar

Machine Learning 10-701
Feb 27, 2017

SVM dual formulation

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i . \mathbf{x}_j$$

$$\sum_i \alpha_i y_i = 0$$
$$C \geq \alpha_i \geq 0$$

$$\text{maximize}_\alpha \quad \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}_j)$$
$$\sum_i \alpha_i y_i = 0$$
$$C \geq \alpha_i \geq 0$$

$\Phi(\mathbf{x})$ – High-dimensional feature space, but never need it explicitly as long as we can compute the dot product fast using some Kernel K

Common Kernels

- Polynomials of degree d

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d$$

- Polynomials of degree up to d

$$K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$$

- Gaussian/Radial kernels (polynomials of all orders – recall series expansion of exp)

$$K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||^2}{2\sigma^2}\right)$$

- Sigmoid

$$K(\mathbf{u}, \mathbf{v}) = \tanh(\eta \mathbf{u} \cdot \mathbf{v} + \nu)$$

SVMs with Kernels

- Choose a set of features and kernel function
- Solve dual problem to obtain support vectors α_i
- At classification time, compute:

$$\mathbf{w} \cdot \Phi(\mathbf{x}) = \sum_i \alpha_i y_i K(\mathbf{x}, \mathbf{x}_i)$$

$$b = y_k - \sum_i \alpha_i y_i K(\mathbf{x}_k, \mathbf{x}_i)$$

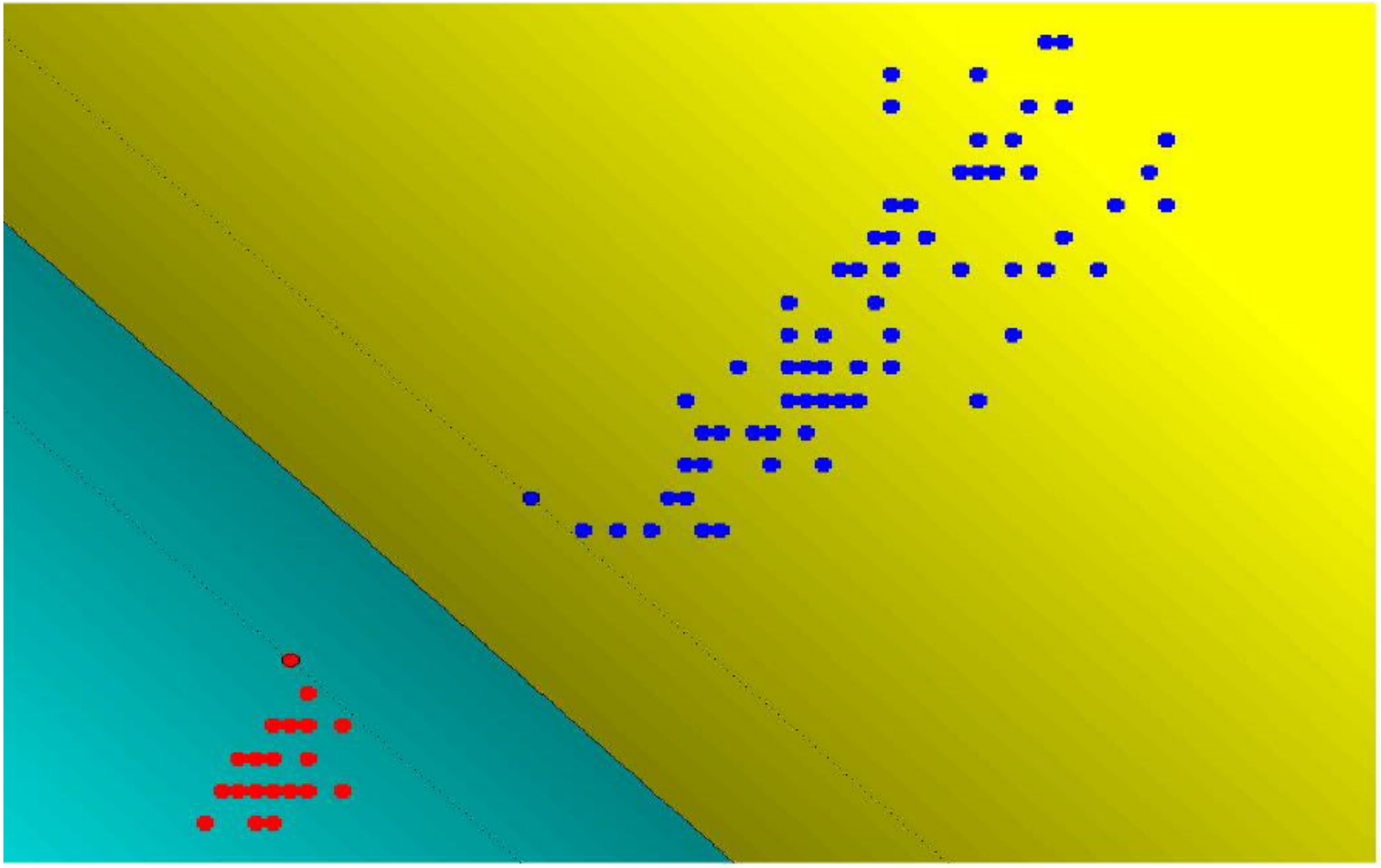
for any k where $C > \alpha_k > 0$

Classify as

$$sign\left(\mathbf{w} \cdot \Phi(\mathbf{x}) + b\right)$$

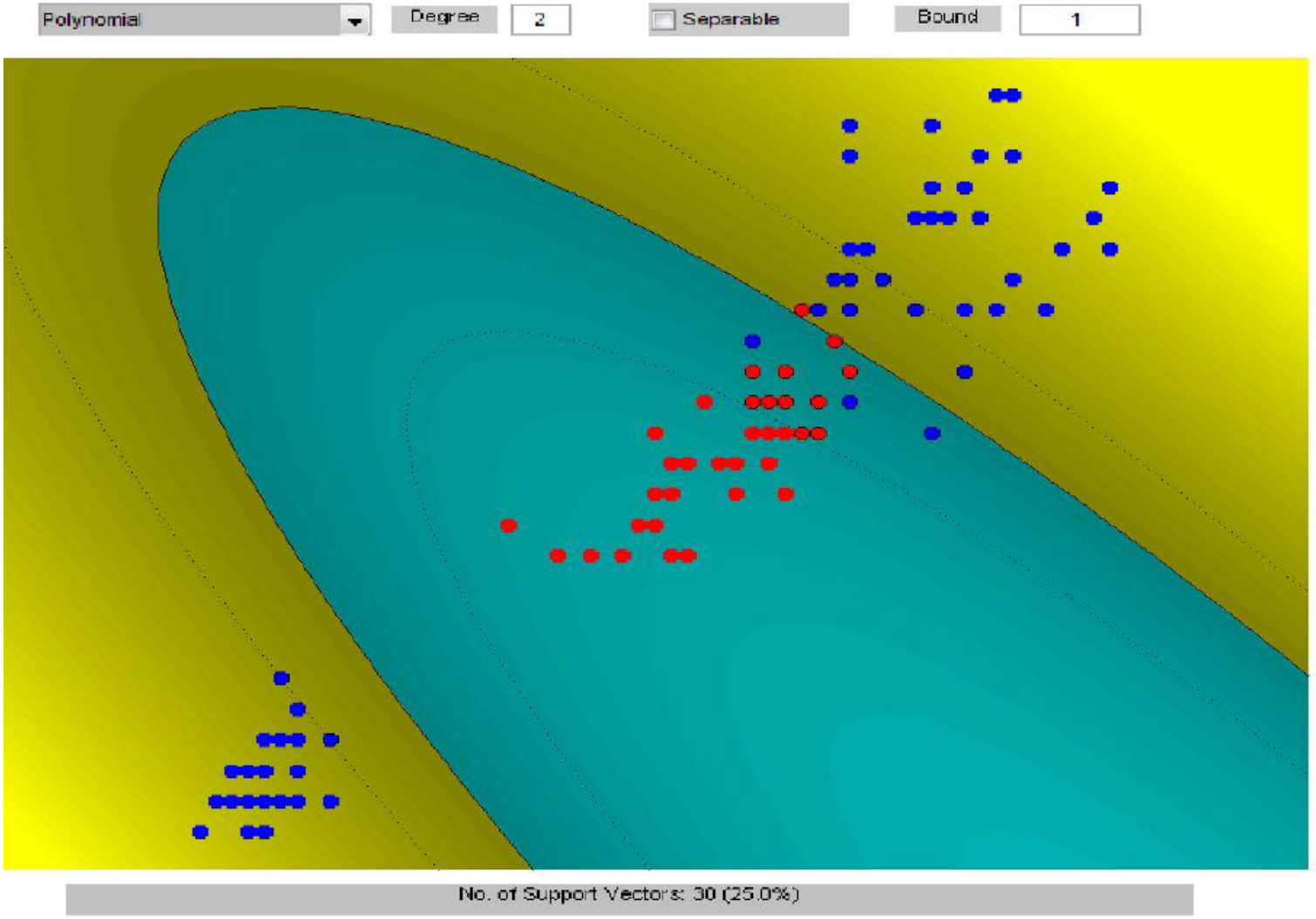
SVMs with Kernels

- Iris dataset, 2 vs 13, Linear Kernel



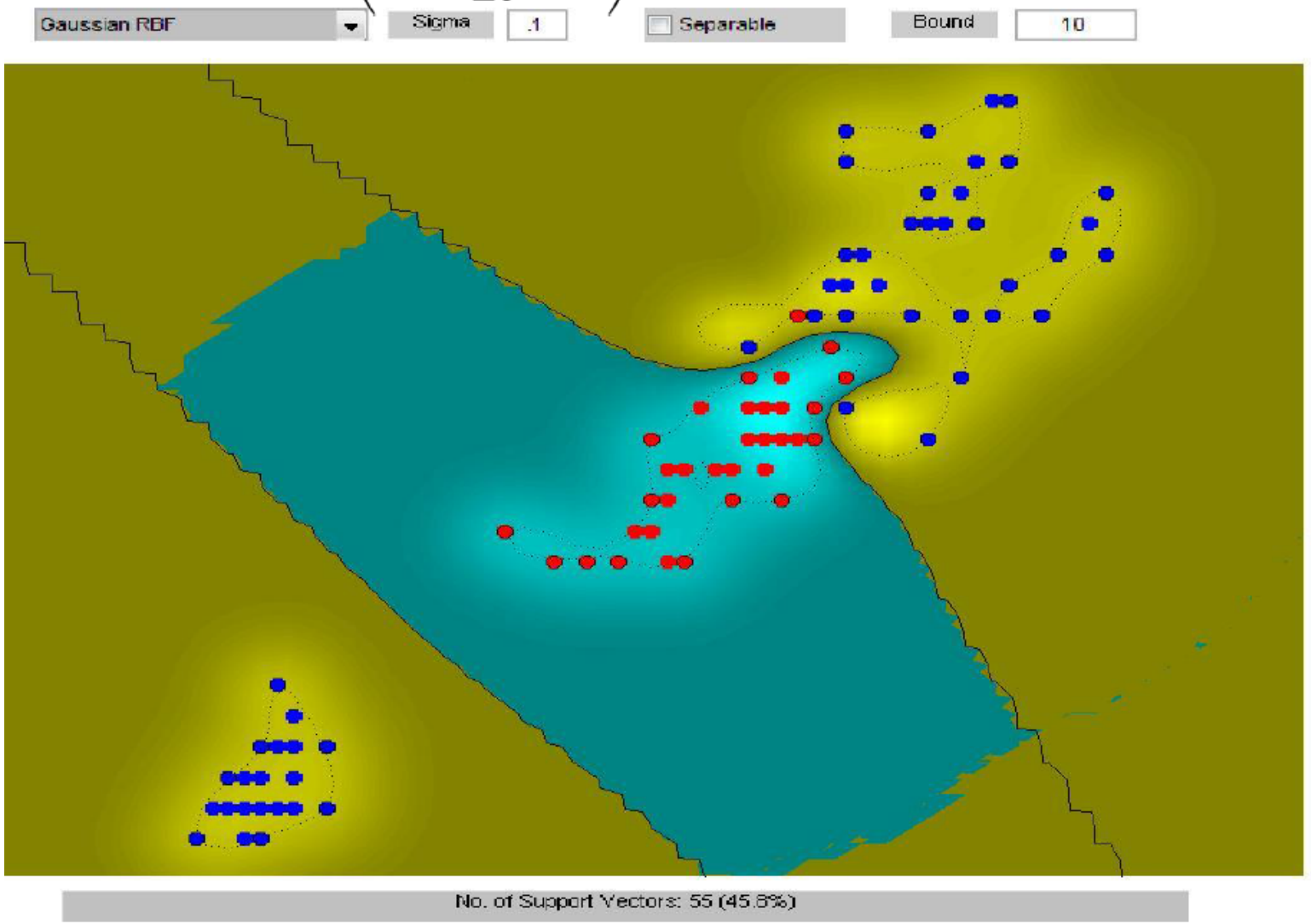
SVMs with Kernels

- Iris dataset, 1 vs 23, Polynomial Kernel degree 2



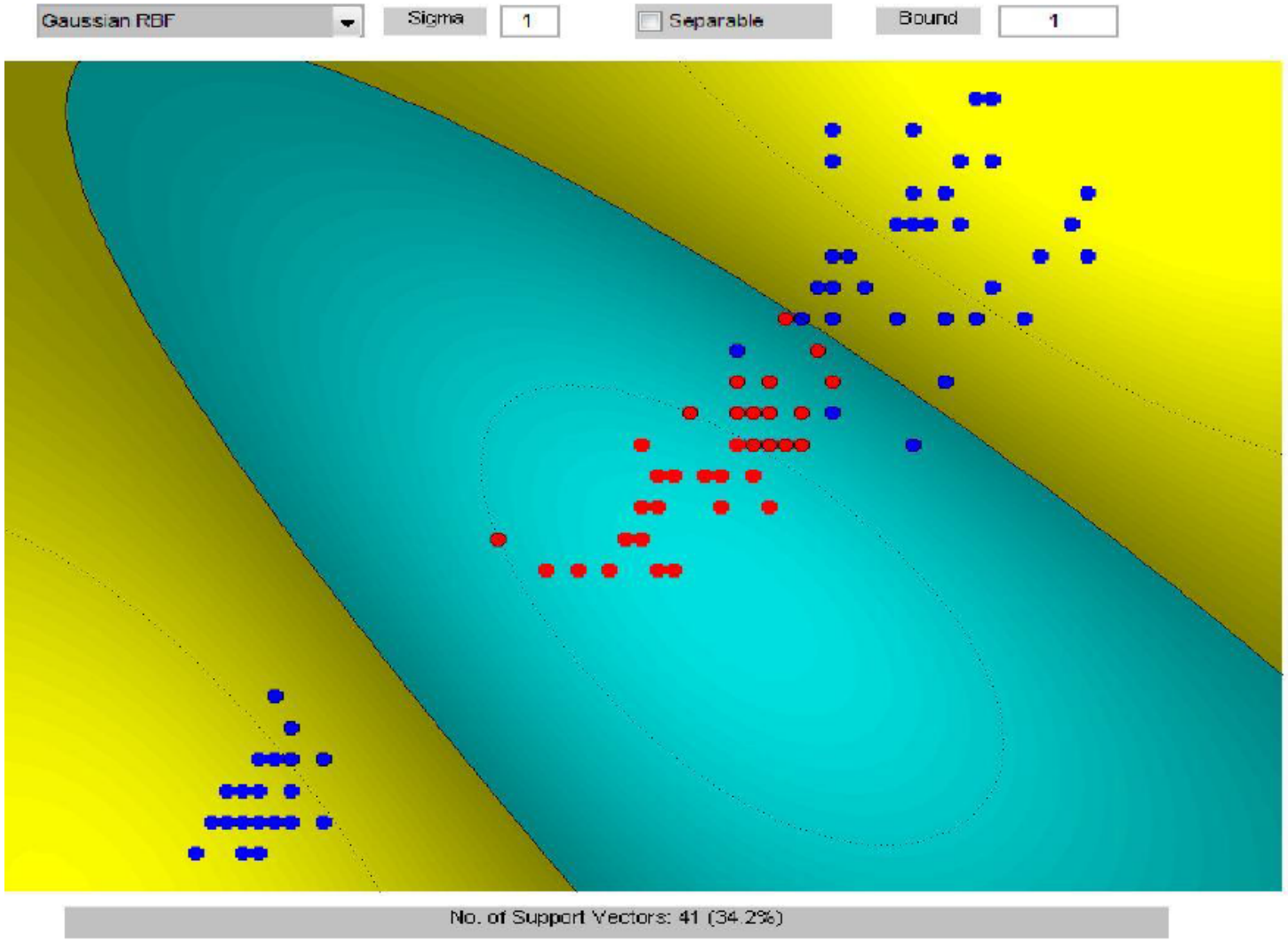
SVMs with Kernels

- Iris dataset, 1 vs 23, Gaussian RBF kernel



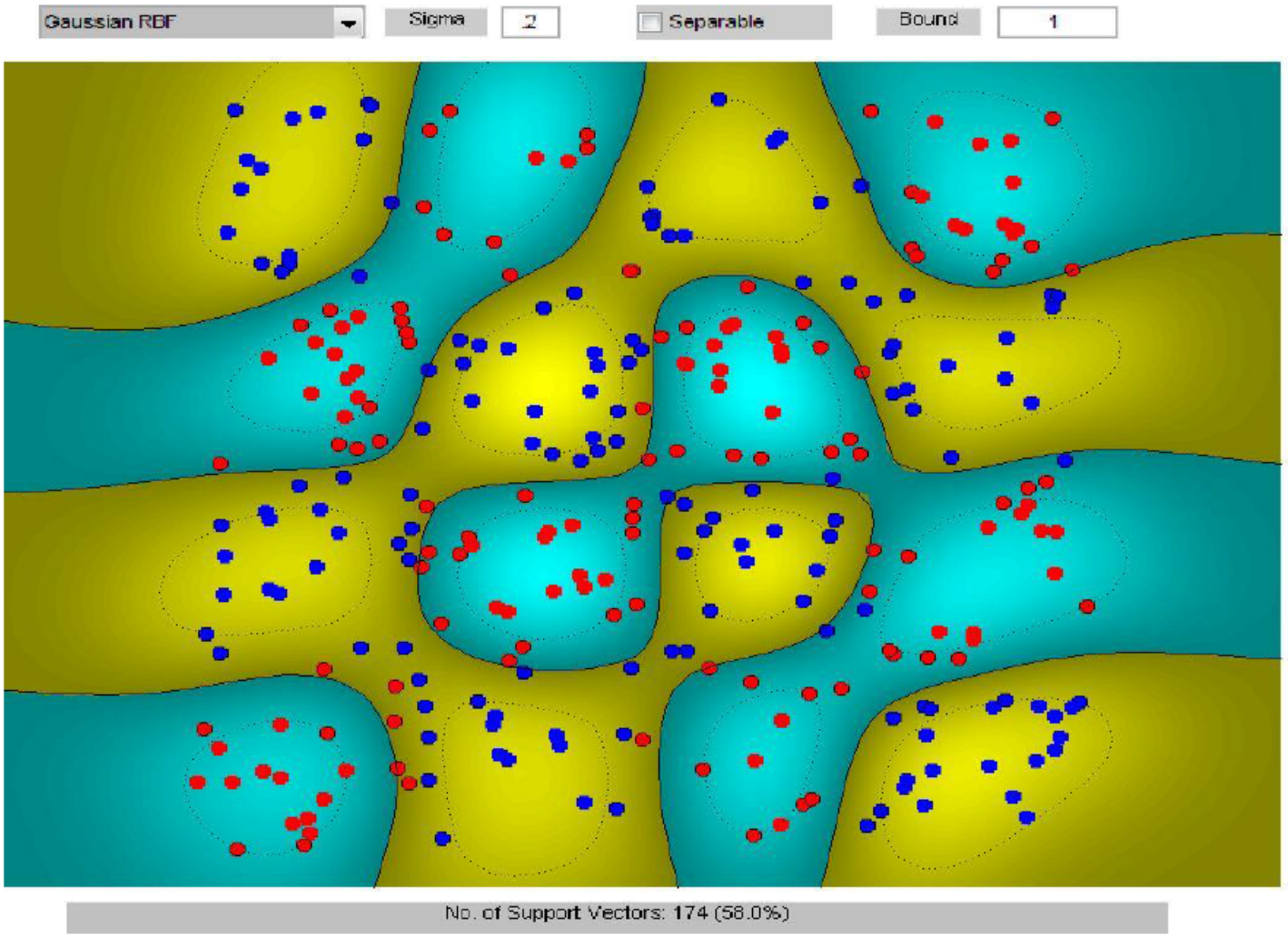
SVMs with Kernels

- Iris dataset, 1 vs 23, Gaussian RBF kernel



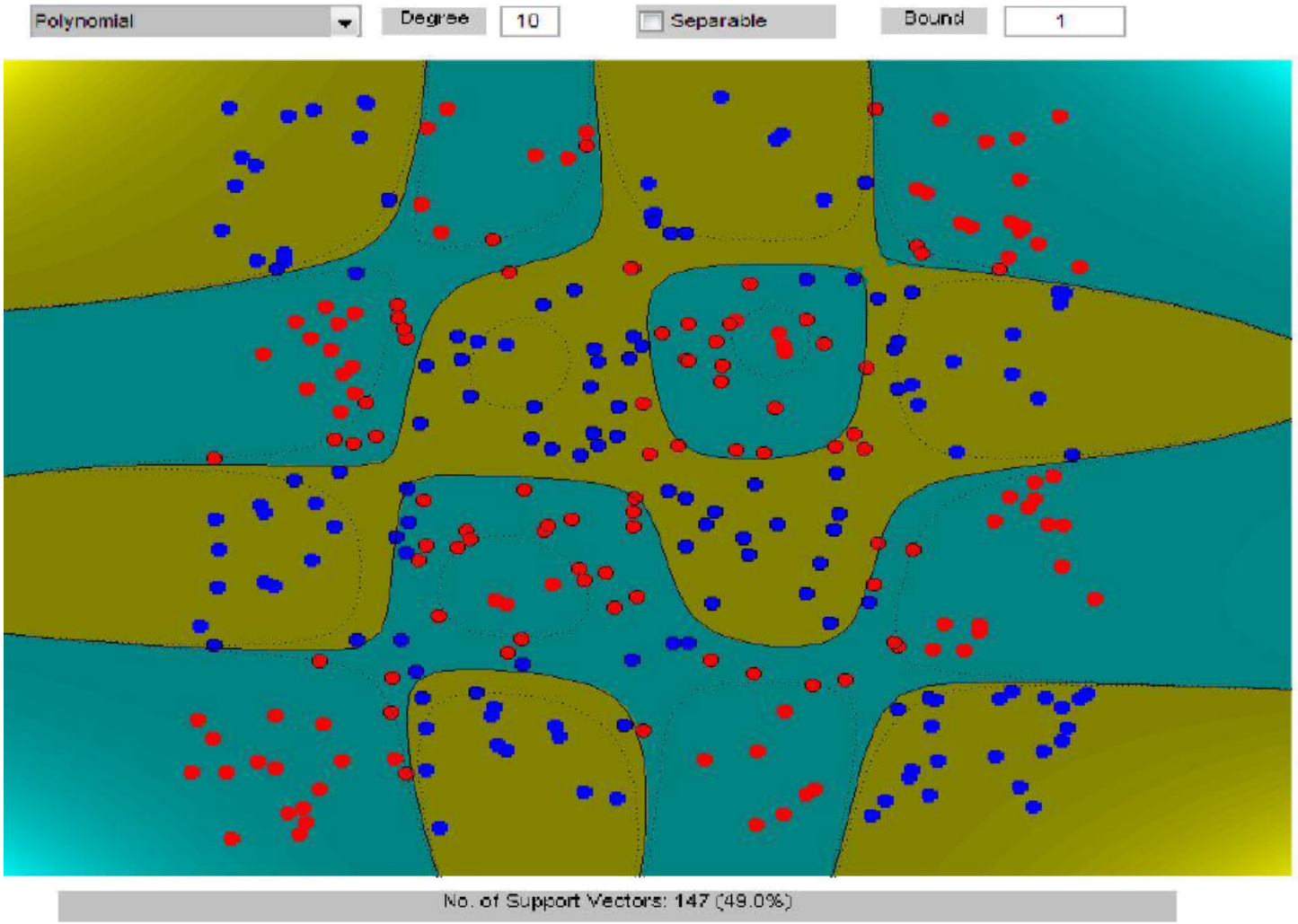
SVMs with Kernels

- Chessboard dataset, Gaussian RBF kernel



SVMs with Kernels

- Chessboard dataset, Polynomial kernel



Corel Dataset

Air shows

Bears

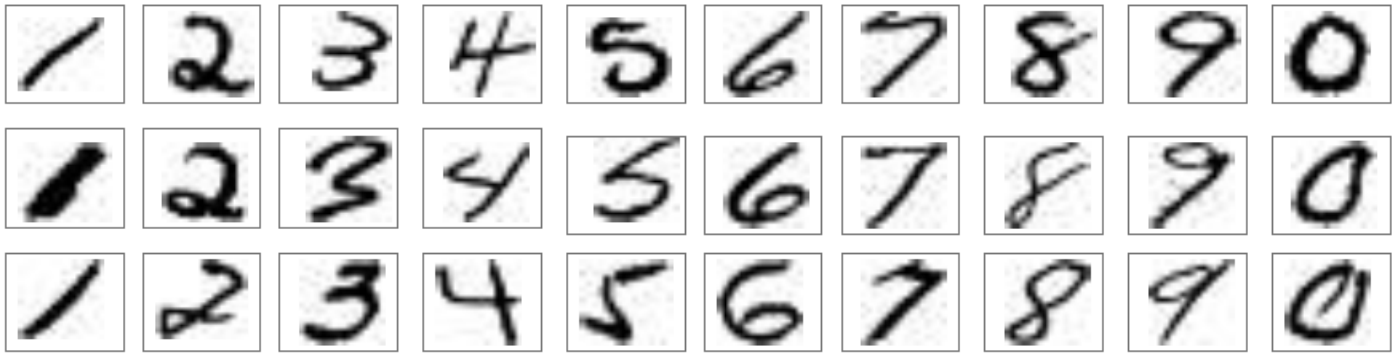
Horses

Corel Dataset

		1	2	3	4	5	6	7	8	9	10	11	12	13	14
air-shows	1	31		1							1	1			
bears	2		26	2	2		2	1		1					
elephants	3		1	27				3						3	
tigers	4			1	32			1							
horses	5					34									
polar-bears	6						30					1		2	1
african-animals	7			1	1			30		1				1	
cheetahs	8						1		32			1			
eagles	9	1								33					
mountains	10	3								1	24	3	3		
fields	11			1				1			2	27	3		
deserts	12						2	1	1	2	1	3	24		
sunsets	13													34	
night scenes	14	1												2	31

Olivier Chapelle 1998

USPS Handwritten digits



- 1000 training and 1000 test instances

Results:

SVM on raw images **~97%** accuracy

SVMs vs. Logistic Regression

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss

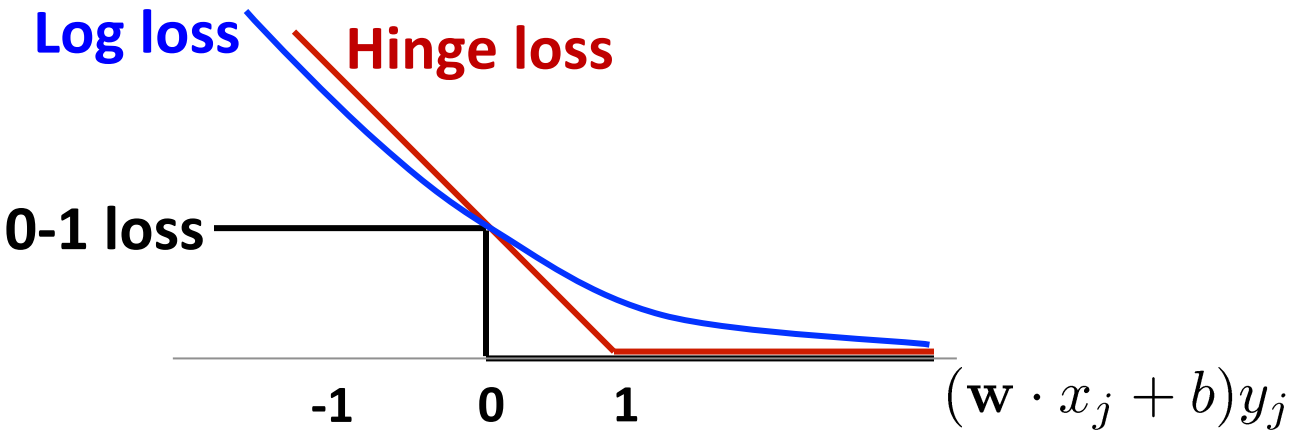
SVMs vs. Logistic Regression

SVM : **Hinge loss**

$$\text{loss}(f(x_j), y_j) = (1 - (\mathbf{w} \cdot x_j + b)y_j))_+$$

Logistic Regression : **Log loss** (-ve log conditional likelihood)

$$\text{loss}(f(x_j), y_j) = -\log P(y_j \mid x_j, \mathbf{w}, b) = \log(1 + e^{-(\mathbf{w} \cdot x_j + b)y_j})$$



SVMs vs. Logistic Regression

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!

Kernels in Logistic Regression

$$P(Y = 1 \mid x, \mathbf{w}) = \frac{1}{1 + e^{-(\mathbf{w} \cdot \Phi(\mathbf{x}) + b)}}$$

- Define weights in terms of features:

$$\mathbf{w} = \sum_i \alpha_i \Phi(\mathbf{x}_i)$$

$$\begin{aligned} P(Y = 1 \mid x, \mathbf{w}) &= \frac{1}{1 + e^{-(\sum_i \alpha_i \Phi(\mathbf{x}_i) \cdot \Phi(\mathbf{x}) + b)}} \\ &= \frac{1}{1 + e^{-(\sum_i \alpha_i K(\mathbf{x}, \mathbf{x}_i) + b)}} \end{aligned}$$

- Derive simple gradient descent rule on α_i

SVMs vs. Logistic Regression

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!

SVMs vs. Logistic Regression

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!
Solution sparse	Often yes!	Almost always no!

SVMs vs. Logistic Regression

	SVMs	Logistic Regression
Loss function	Hinge loss	Log-loss
High dimensional features with kernels	Yes!	Yes!
Solution sparse	Often yes!	Almost always no!
Semantics of output	"Margin"	Real probabilities

What you need to know

- Maximizing margin
- Derivation of SVM formulation
- Slack variables and hinge loss
- Relationship between SVMs and logistic regression
  - 0/1 loss
  - Hinge loss
  - Log loss
- Tackling multiple class
  - One against All
  - Multiclass SVMs
- Dual SVM formulation
  - Easier to solve when dimension high d > n
  - Kernel Trick

Can we use kernels in regression?

Ridge regression

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2 \qquad \widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

Similarity with SVMs

Primal problem:

$$\min_{\beta, z_i} \sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|_2^2$$
$$\text{s.t. } z_i = Y_i - X_i\beta$$

SVM Primal problem:

$$\min_{w, \xi_i} C\sum_{i=1}^{n} \xi_i + \frac{1}{2}\|w\|_2^2$$
$$\text{s.t. } \xi_i = \max(1 - Y_i X_i\, w, 0)$$

Lagrangian:

$$\sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|^2 + \sum_{i=1}^{n} \alpha_i(z_i - Y_i + X_i\beta)$$

α_i – Lagrange parameter, one per training point

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2 \qquad \widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$

Dual problem:

$$\max_{\alpha} \min_{\beta, z_i} \sum_{i=1}^{n} z_i^2 + \lambda\|\beta\|^2 + \sum_{i=1}^{n} \alpha_i(z_i - Y_i + X_i\beta)$$

$\alpha = \{\alpha_i\}$ for i = 1,…, n

Taking derivatives of Lagrangian wrt β and z_i we get:

$$\beta = -\frac{1}{2\lambda}\mathbf{A}^\top\alpha \qquad z_i = -\frac{\alpha_i}{2}$$

Dual problem: $$\max_{\alpha} \; -\frac{\alpha^\top\alpha}{4} - \frac{1}{4\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y}$$

n-dimensional optimization problem

Ridge regression (dual)

$$\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|_2^2$$

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y}$$
$$= \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

Dual problem:

$$\max_{\alpha} \; -\frac{\alpha^\top\alpha}{4} - \frac{1}{4\lambda}\alpha^\top\mathbf{A}\mathbf{A}^\top\alpha - \alpha^\top\mathbf{Y} \qquad \Rightarrow \widehat{\alpha} = -\left(\frac{\mathbf{A}\mathbf{A}^\top}{\lambda} + \mathbf{I}\right)^{-1} 2\,\mathbf{Y}$$

can get back $\hat{\beta} = -\frac{1}{2\lambda}\mathbf{A}^\top\hat{\alpha} = \mathbf{A}^\top(\mathbf{A}\mathbf{A}^\top + \lambda\mathbf{I})^{-1}\mathbf{Y}$

Weighted average of training points

Weight of each training point (but typically not sparse)

Kernelized ridge regression

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I})^{-1}\mathbf{A}^T\mathbf{Y} \qquad \widehat{f}_n(X) = \mathbf{X}\widehat{\beta}$$

Using dual, can re-write solution as:

$$\widehat{\beta} = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I})^{-1}\mathbf{Y}$$

How does this help?

- Only need to invert n x n matrix (instead of p x p or m x m)
- More importantly, kernel trick!

$$\widehat{f}_n(X) = \mathbf{K}_X(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{Y} \quad \text{where} \quad \begin{aligned} \mathbf{K}_X(i) &= \boldsymbol{\phi}(X) \cdot \boldsymbol{\phi}(X_i) \\ \mathbf{K}(i,j) &= \boldsymbol{\phi}(X_i) \cdot \boldsymbol{\phi}(X_j) \end{aligned}$$

Work with kernels, never need to write out the high-dim vectors

Kernelized ridge regression

$$\widehat{f}_n(X) = \mathbf{K}_X(\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{Y}$$ where $$\mathbf{K}_X(i) = \boldsymbol{\phi}(X) \cdot \boldsymbol{\phi}(X_i)$$ $$\mathbf{K}(i,j) = \boldsymbol{\phi}(X_i) \cdot \boldsymbol{\phi}(X_j)$$

Work with kernels, never need to write out the high-dim vectors

Examples of kernels:

Polynomials of degree exactly d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v})^d$

Polynomials of degree up to d $K(\mathbf{u}, \mathbf{v}) = (\mathbf{u} \cdot \mathbf{v} + 1)^d$

Gaussian/Radial kernels $K(\mathbf{u}, \mathbf{v}) = \exp\left(-\frac{||\mathbf{u} - \mathbf{v}||^2}{2\sigma^2}\right)$

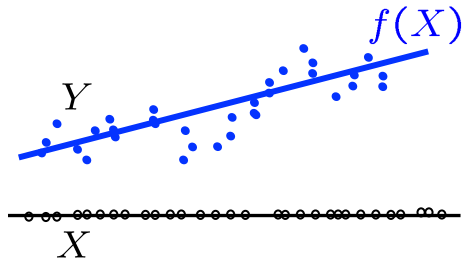
Ridge Regression with (implicit) nonlinear features $\boldsymbol{\phi}(X)$! $f(X) = \phi(X)\beta$

Prior for kernelized ridge regression

- Recall ridge regression was obtained as MAP under a Gaussian prior for a linear model with Gaussian noise

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = X\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$



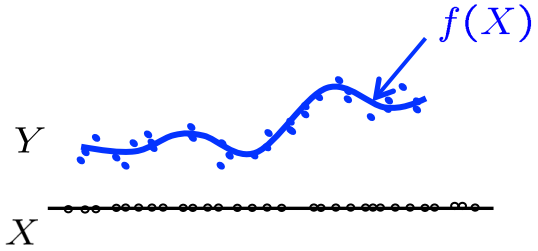
- For kernelized ridge regression, what's the prior?

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = \phi(X)\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$

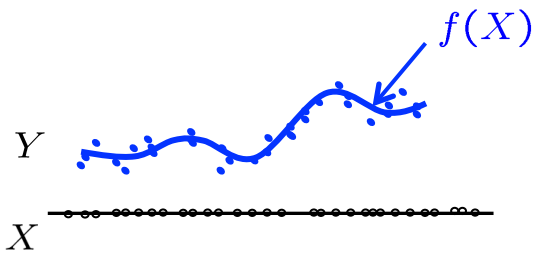
$$\Rightarrow f(X) \sim \mathcal{N}(0, \tau^2 \phi(X) \cdot \phi(X))$$

$$\sim \mathcal{N}(0, \tau^2 K_{X,X})$$



Prior for kernelized ridge regression

- For kernelized ridge regression, what's the prior?



$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f(X) = \phi(X)\beta \quad \beta \sim \mathcal{N}(0, \tau^2 \mathbf{I})$$

$$\Rightarrow f(X) \sim \mathcal{N}(0, \tau^2 \phi(X) \cdot \phi(X)) \sim \mathcal{N}(0, \tau^2 K_{X,X})$$

$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} = \begin{bmatrix} \phi(X_i) \\ \phi(X_j) \end{bmatrix} \beta$$

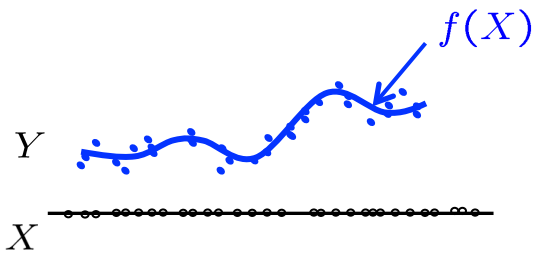
$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} \sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} \phi(X_i) \cdot \phi(X_i) & \phi(X_i) \cdot \phi(X_j) \\ \phi(X_j) \cdot \phi(X_i) & \phi(X_j) \cdot \phi(X_j) \end{bmatrix}\right)$$

$$\sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} K(i,i) & K(i,j) \\ K(j,i) & K(j,j)) \end{bmatrix}\right)$$

Prior for kernelized ridge regression

- For kernelized ridge regression, what's the prior?

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$



$$\begin{bmatrix} f(X_i) \\ f(X_j) \end{bmatrix} \sim \mathcal{N}\left(0, \tau^2 \begin{bmatrix} K(i,i) & K(i,j) \\ K(j,i) & K(j,j)) \end{bmatrix}\right)$$

Then $\tau^2 K(i,j) = \Sigma(i,j)$ i.e. we can interpret Kernel function as the covariance function under a Gaussian process prior.

$$f \sim \mathcal{N}(0, \mathbf{\Sigma})$$

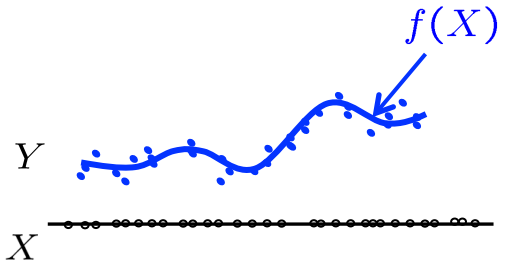
$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i,j)$$

Gaussian process regression

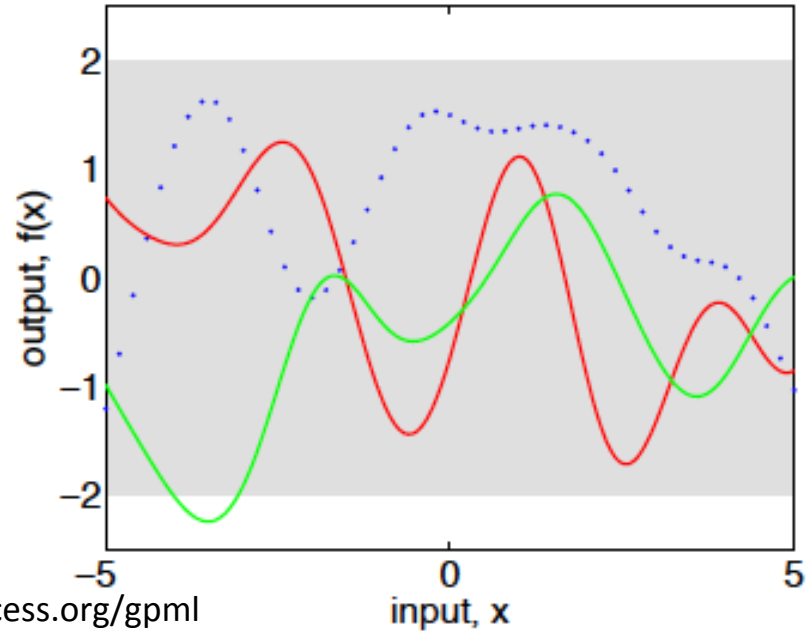
$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f \sim \mathcal{N}(0, \mathbf{\Sigma})$$ Gaussian process prior

$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i, j)$$



3 functions drawn at random from GP **prior**

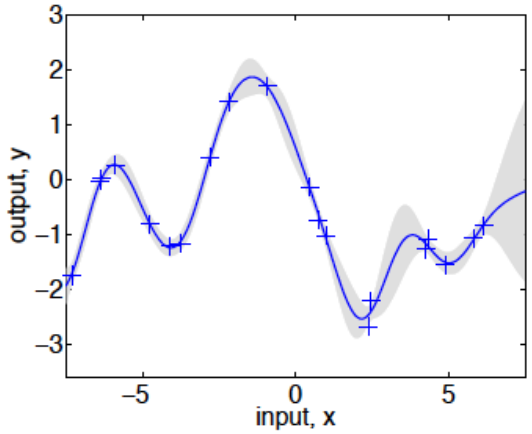


$$\Sigma(i, j) = \exp\left(-\frac{1}{2}\|X_i - X_j\|^2\right)$$

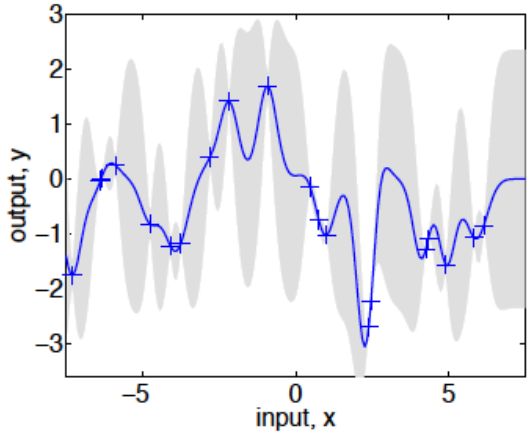
Shaded region represents pointwise mean +/- 2 standard deviation

guassianprocess.org/gpml

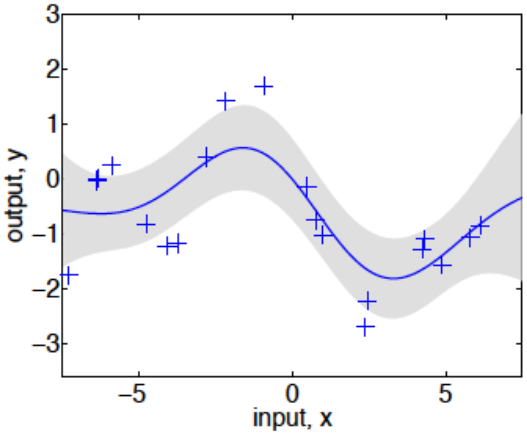
Gaussian process regression



(a), $\ell = 1$



(b), $\ell = 0.3$



(c), $\ell = 3$

Figure 2.5: (a) Data is generated from a GP with hyperparameters $(\ell, \sigma_f, \sigma_n) = (1, 1, 0.1)$, as shown by the + symbols. Using Gaussian process prediction with these hyperparameters we obtain a 95% confidence region for the underlying function f (shown in grey). Panels (b) and (c) again show the 95% confidence region, but this time for hyperparameter values $(0.3, 1.08, 0.00005)$ and $(3.0, 1.16, 0.89)$ respectively.

Controls "smoothness" of functions

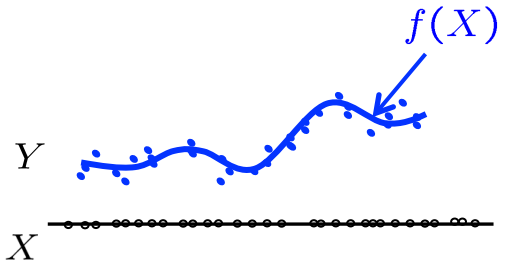
$$k_y(x_p, x_q) = \sigma_f^2 \exp\left(-\frac{1}{2\ell^2}(x_p - x_q)^2\right) + \sigma_n^2 \delta_{pq}$$

Gaussian process regression

$$Y = f(X) + \epsilon, \;\; \epsilon \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

$$f \sim \mathcal{N}(0, \mathbf{\Sigma}) \qquad \text{Gaussian process prior}$$

$$\text{i.e. } \mathbb{E}[f(X_i) f(X_j)] = \Sigma(i,j)$$



Lets derive the MAP estimate under this prior:

$$\begin{bmatrix} \mathbf{Y} \\ f(X) \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} \mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I} & \mathbf{\Sigma}_{nX} \\ \mathbf{\Sigma}_{Xn} & \mathbf{\Sigma}_{XX} \end{bmatrix}\right)$$

Posterior distribution

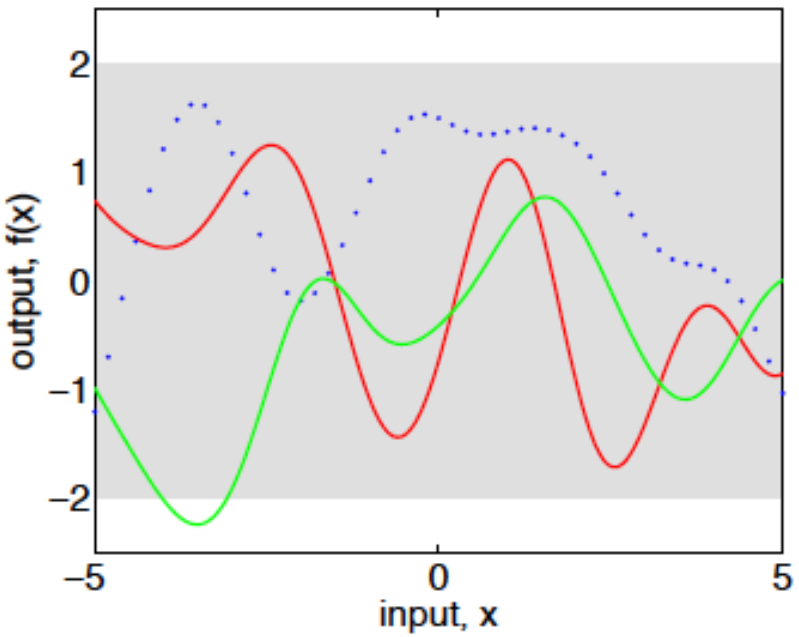
$$\Rightarrow f(X)|\mathbf{Y} \sim \mathcal{N}(\mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{Y}, \mathbf{\Sigma}_{XX} - \mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{\Sigma}_{nX})$$

MAP estimate $\widehat{f}(X) = \mathbf{\Sigma}_{Xn}(\mathbf{\Sigma}_{nn} + \sigma^2 \mathbf{I})^{-1}\mathbf{Y} \equiv \mathbf{K}_X(\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{Y}$

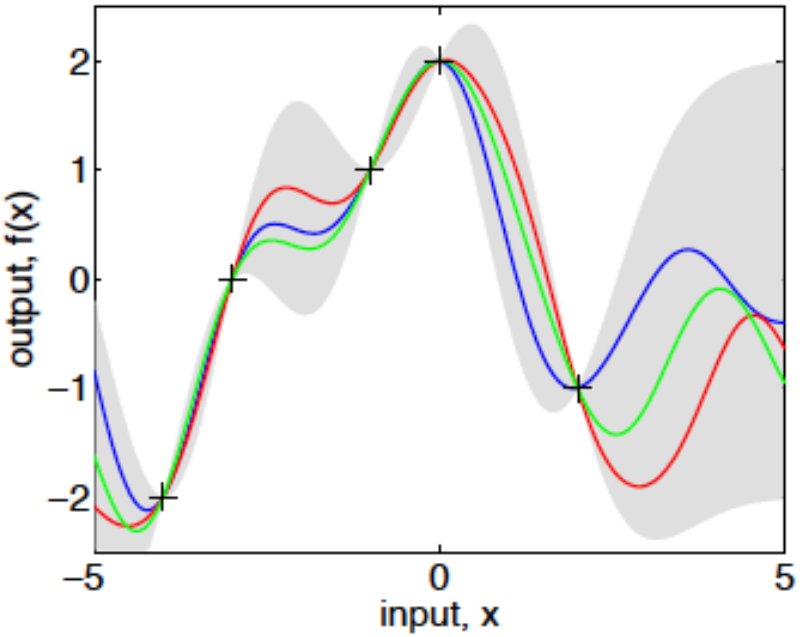
Kernel Ridge Regression!

Gaussian process regression

3 functions drawn at random from GP **prior**



3 functions drawn at random from **posterior** based on five noiseless observations



Shaded region represents pointwise mean +/- 2 standard deviation

http://www.gaussianprocess.org/gpml/