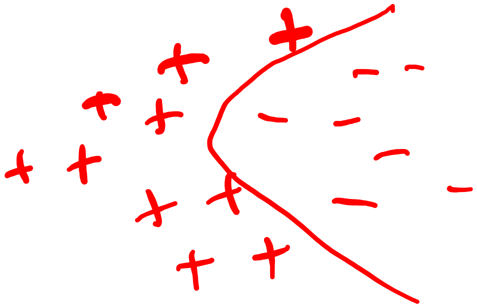
Kernel Trick



$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\downarrow$$

$$\phi(x) = \begin{bmatrix} x_1 \\ x_1^2 \\ x_2 \\ x_2^2 \\ x_1 x_2 \end{bmatrix}$$

$\Downarrow$

Aarti Singh

Machine Learning 10-701
Feb 8, 2023

The Kernel Trick!

SVM dual

$z^T K z$

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

$\alpha = (\alpha_1 .. \alpha_n)$

$x_i \cdot x_j$ ✓

$$\boxed{K(\mathbf{x}_i,\mathbf{x}_j) = \Phi(\mathbf{x}_i)\cdot\Phi(\mathbf{x}_j)}$$

$\phi(x_i)\cdot\phi(x_j)$

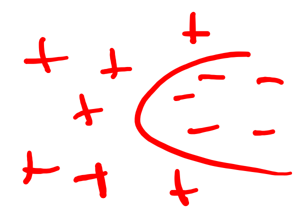
$$\sum_i \alpha_i y_i = 0$$
$$C \geq \alpha_i \geq 0$$

- Never represent features explicitly
  - Compute dot products in closed form
- Constant-time high-dimensional dot-products for many classes of features

Dot Product of Polynomial features

$$\Phi(\mathbf{x}) = \text{polynomials of degree exactly d}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \mathbf{z} = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix}$$



d=1 $\Phi(\mathbf{x})\cdot\Phi(\mathbf{z}) = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \cdot \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} = x_1 z_1 + x_2 z_2 = \mathbf{x}\cdot\mathbf{z}$

d=2 $\Phi(\mathbf{x})\cdot\Phi(\mathbf{z}) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \cdot \begin{bmatrix} z_1^2 \\ \sqrt{2}z_1z_2 \\ z_2^2 \end{bmatrix}$

9 multiplications

$$\begin{aligned} &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2x_1x_2z_1z_2 \\ &= (x_1z_1 + x_2z_2)^2 \\ &= (\mathbf{x}\cdot\mathbf{z})^2 \end{aligned}$$

3 multiplication

d $\Phi(\mathbf{x})\cdot\Phi(\mathbf{z}) = K(\mathbf{x},\mathbf{z}) = (\mathbf{x}\cdot\mathbf{z})^d$

Common Kernels

- Polynomials of degree d

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

- Polynomials of degree up to d

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

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

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

- Sigmoid

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

Mercer Kernels

What functions are valid kernels that correspond to feature vectors $\varphi(\mathbf{x})$?

$K(u,v) = \underline{\phi(u) \cdot \phi(v)}$

Answer: **Mercer kernels** K

- K is continuous ✓
- K is symmetric ✓
- K is positive semi-definite, i.e. $\mathbf{z}^T K \mathbf{z} \geq 0$ for all $\mathbf{z}$ ✓

$$z^T \begin{bmatrix} \phi(u)\cdot\phi(u) & \phi(u)\cdot\phi(v) \\ \phi(v)\cdot\phi(u) & \phi(v)\cdot\phi(v) \end{bmatrix} z$$

Ensures optimization is concave maximization

Overfitting

- Huge feature space with kernels, what about overfitting???
  - Maximizing margin leads to sparse set of support vectors α sparse
  - Some interesting theory says that SVMs search for simple hypothesis with large margin
  - Often robust to overfitting

What about classification time?

$\{x_i\}_{i=1}^{n}$ $K(x_i, x_j)$

- For a new input **x**, if we need to represent Φ(**x**), we are in trouble!
- Recall classifier: sign(**w**.Φ(**x**)+b)

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

$$b = y_k - \mathbf{w}.\Phi(\mathbf{x}_k)$$

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

$w \cdot \phi(x) = \sum_i \alpha_i y_i \underbrace{\phi(x_i) \cdot \phi(x)}_{K(x_i, x)}$

- Using kernels we are cool!

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

SVMs with Kernels

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

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

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

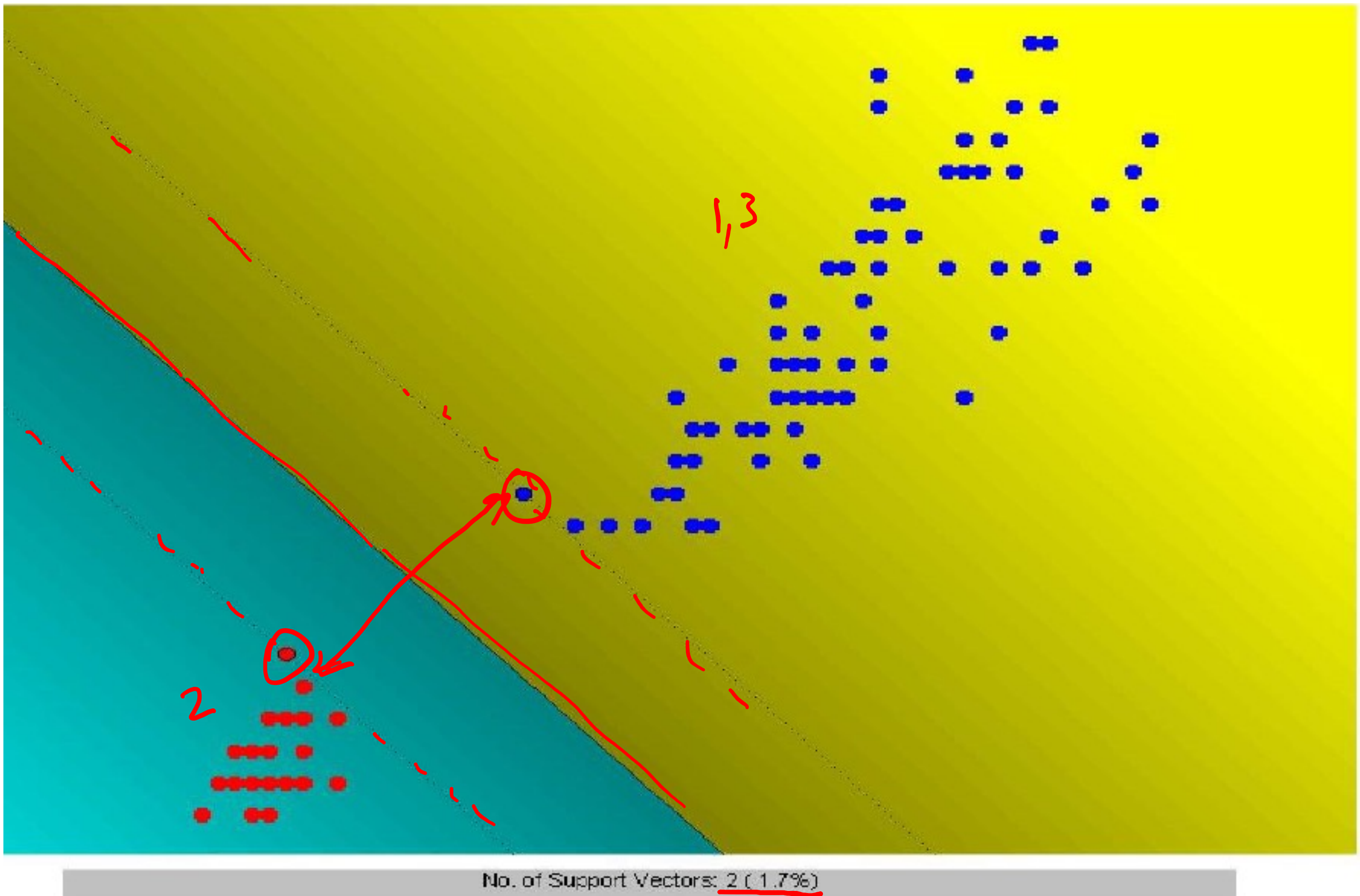
for any k where $C > \alpha_k > 0$

Classify as

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

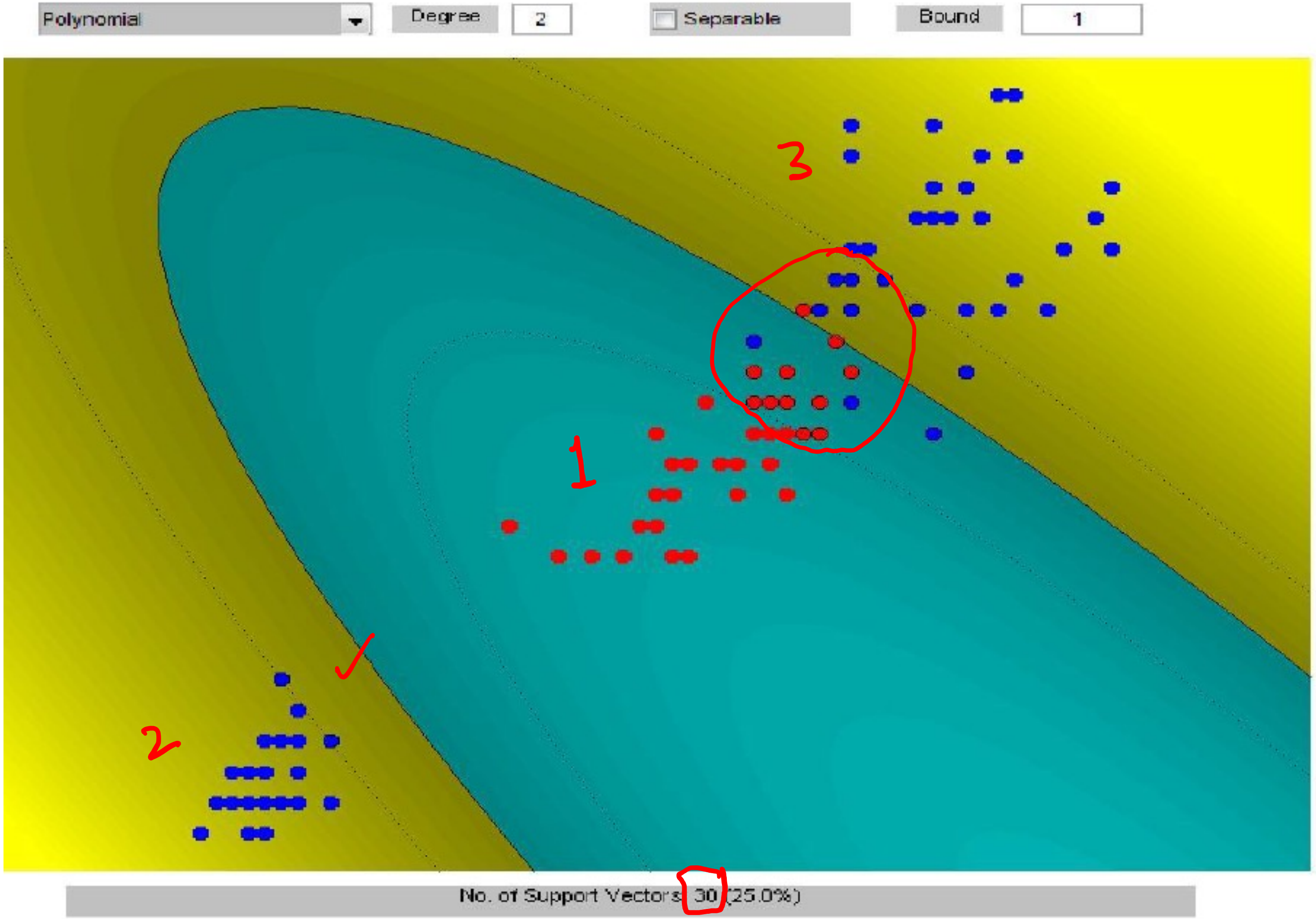
SVMs with Kernels

- Iris dataset, 2 vs 13, Linear Kernel



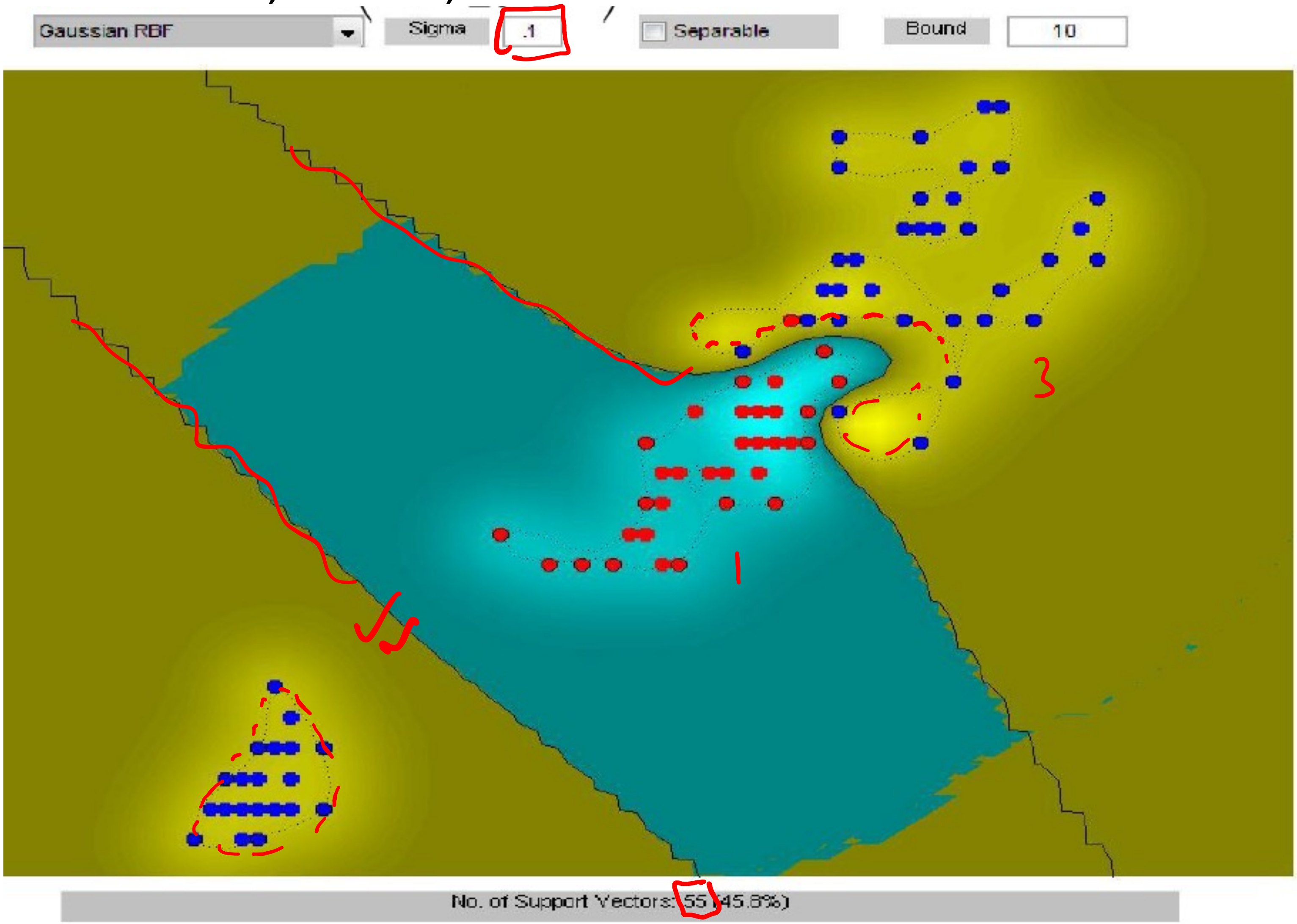
SVMs with Kernels

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



SVMs with Kernels

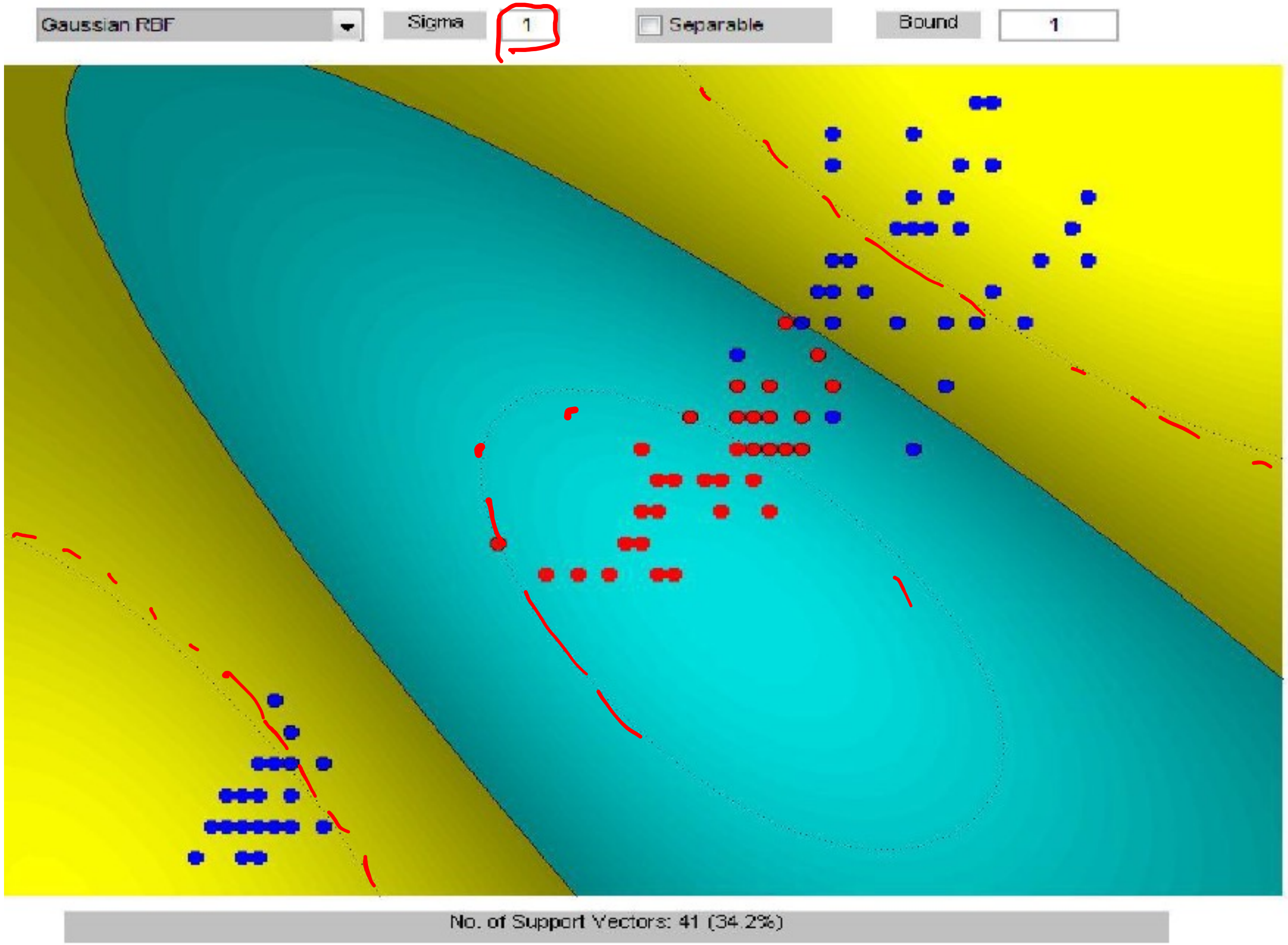
- Iris dataset, 1 vs 23, Gaussian RBF kernel



SVMs with Kernels

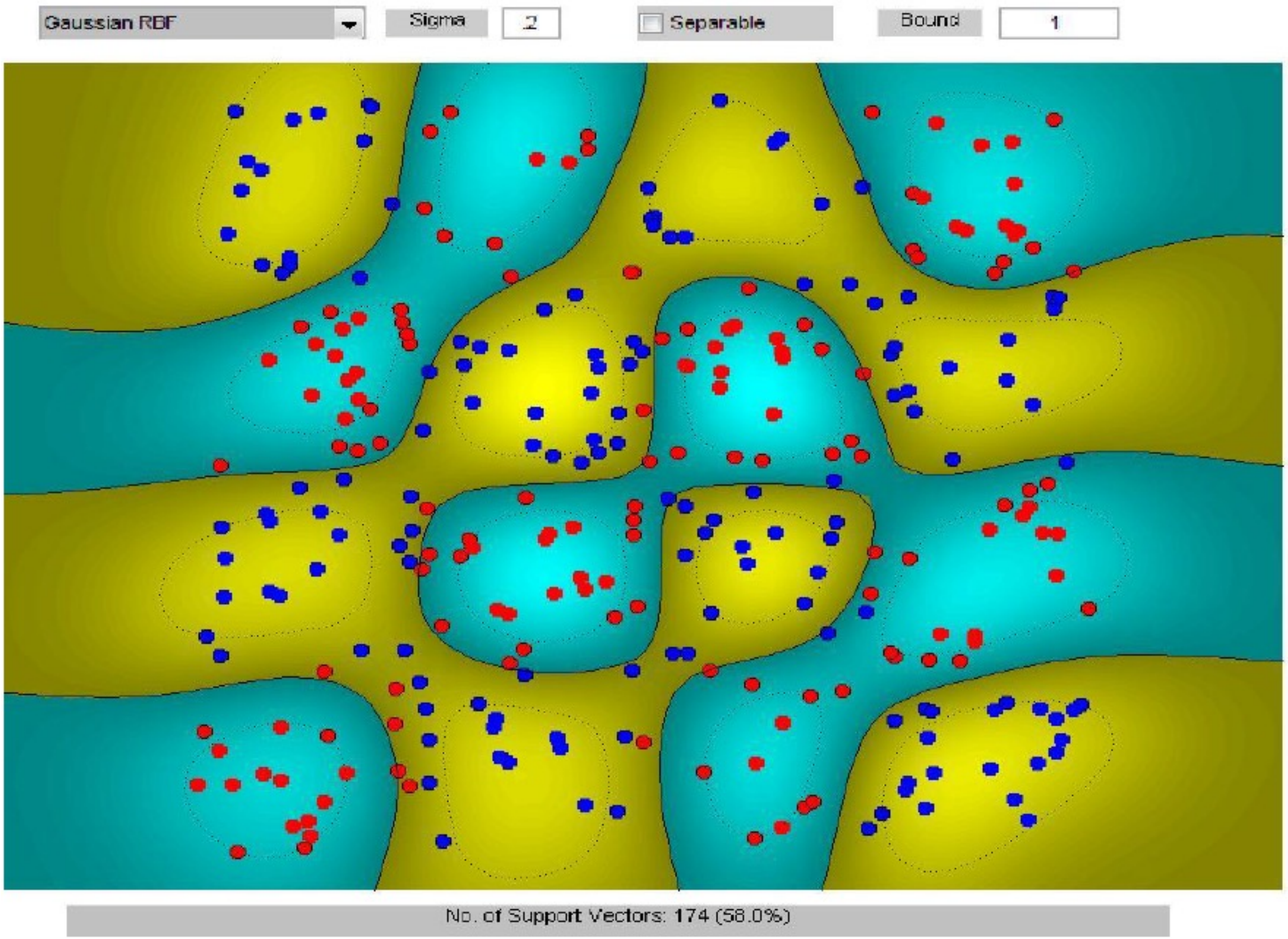
$\exp\left(\frac{-\|u-v\|^2}{2\sigma^2}\right)$

- Iris dataset, 1 vs 23, Gaussian RBF kernel



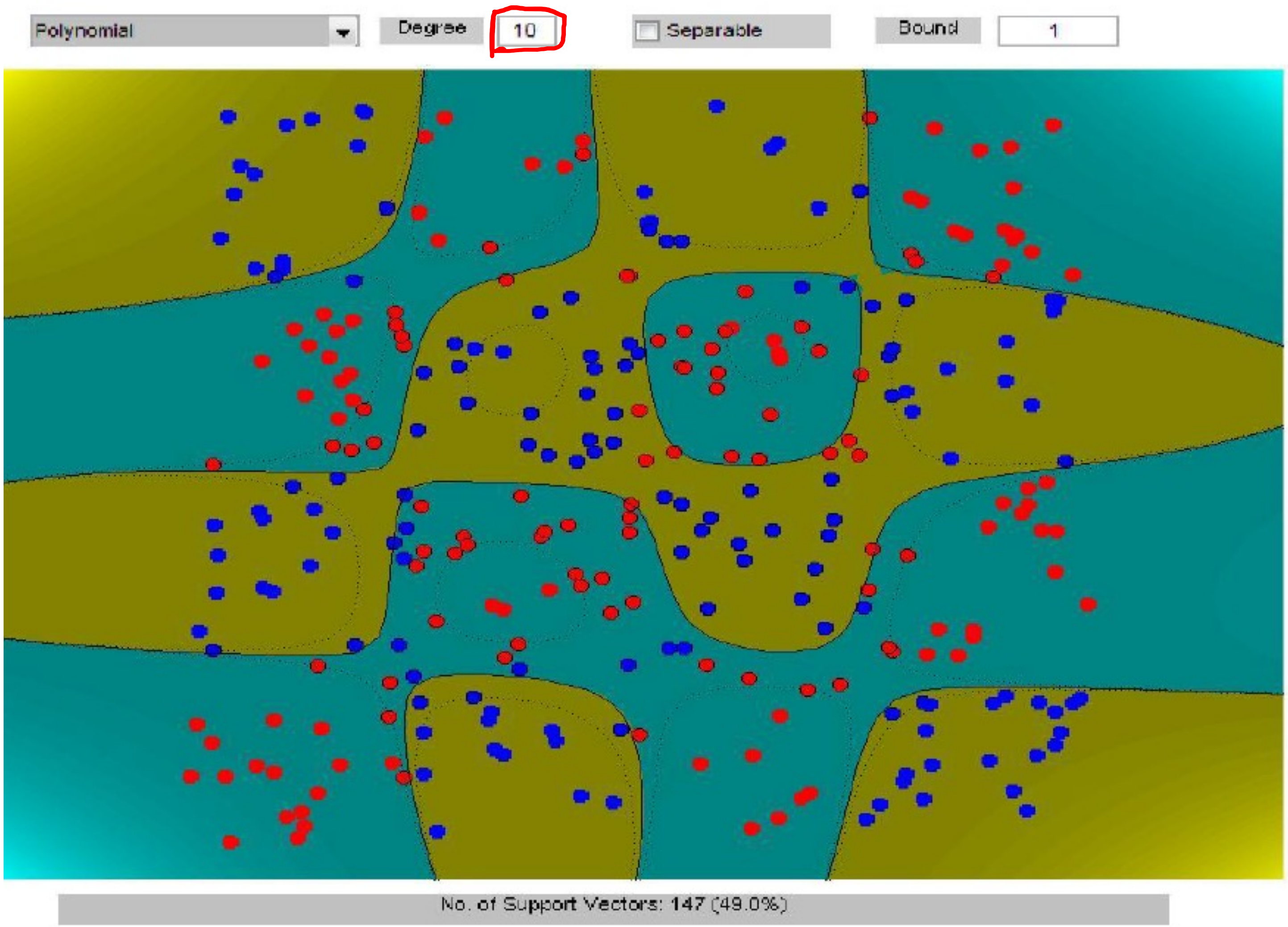
SVMs with Kernels

- Chessboard dataset, Gaussian RBF kernel

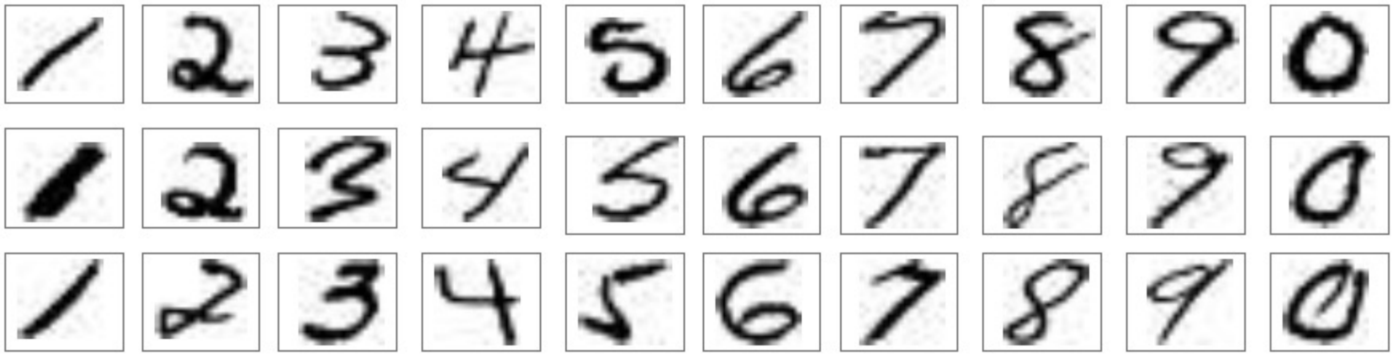


SVMs with Kernels

- Chessboard dataset, Polynomial kernel



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

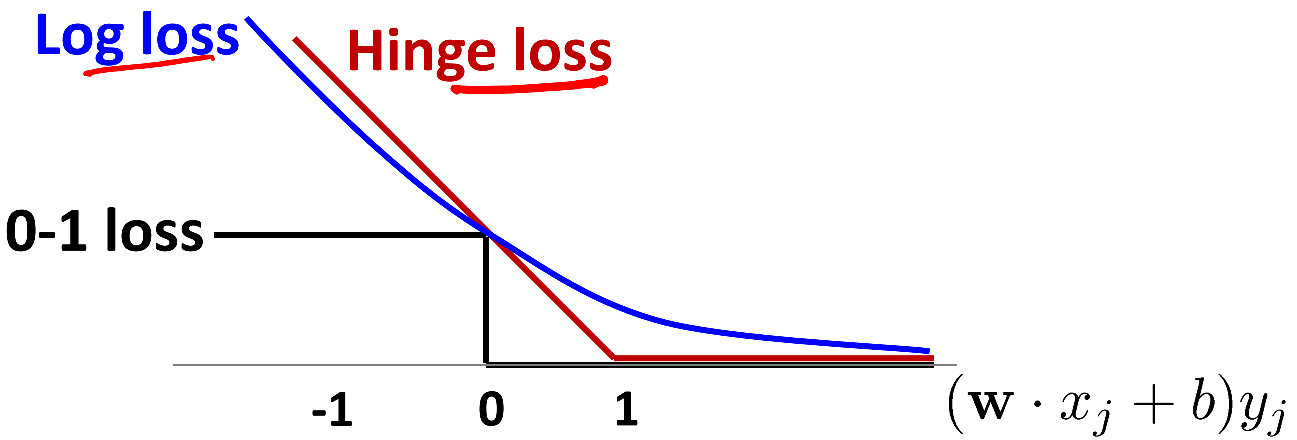
SVMs vs. Logistic Regression

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

$P(Y=1|X) = \frac{1}{1+\exp(-w\cdot x - w_0)}$

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

Kernels in Logistic Regression

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

logistic model ✓

Regularized log likelihood: $-\log \prod_{i=1}^{n} P(y_i \mid x_i, w)$

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \log(1 + e^{y_i(\mathbf{w}\cdot\Phi(x_i)+b)}) + \frac{\lambda}{2}\|\mathbf{w}\|^2$$

Equivalent constrained optimization problem:

$\mathcal{L}(w, z_i, \alpha_i)$

$$\min_{w, z_i} \sum_{i=1}^{n} \log(1 + e^{z_i}) + \frac{\lambda}{2}\|w\|^2$$

s.t. $z_i = y_i(w \cdot \phi(x_i) + b)$ $\quad i = 1..n$ $\quad \alpha_i$ ✓ $\quad \frac{\partial \mathcal{L}}{\partial w}$

$g(\,) = 0$

Kernels in Logistic Regression

Lagrangian: $\underbrace{\mathcal{L}(w, z_i, \alpha_i)}_{d(\alpha)} = \sum_{i=1}^{n} \log(1+e^{z_i}) + \frac{\lambda}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left(z_i - y_i(w\cdot\phi(x_i)+b)\right)$

Derivatives: $\frac{\partial \mathcal{L}}{\partial w} = \frac{\cancel{2}\lambda}{\cancel{2}} w + \sum_{i=1}^{n} \alpha_i y_i \phi(x_i) = 0$

$\Rightarrow \quad w = -\frac{1}{\lambda} \underline{\sum_{i=1}^{n} \alpha_i y_i \phi(x_i)}$

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:

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

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

$$= \frac{1}{1+e^{-(\sum_i y_i \alpha_i K(\mathbf{x},\mathbf{x}_i)+b)}}$$

- 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!
Solution sparse	Often yes!	Almost always no!
Semantics of output	"Margin"	Real probabilities $P(Y=1 \mid X)$

Kernel Trick

- Only dot products between data points appear in optimization
- Replace with kernel
- Valid kernels aka Mercer kernels
- Can apply to other methods such as linear regression, PCA (principal component analysis), … etc.