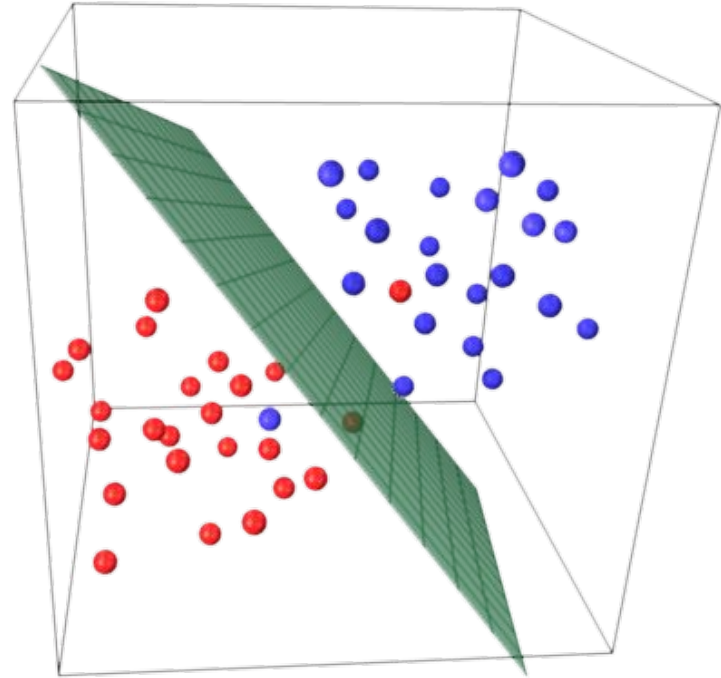


Support Vector Machines

DSI SEA5, jf.omhover, Sep 30 2016

a priori version
(for “solutions” use a posteriori version)



Support Vector Machines

DSI SEA5, jf.omhover, Sep 30 2016



STANDARDS

- Compute a hyperplane as a decision boundary in SVC
- Explain what a support vector is in plain english
- Tune a SVC or SVM using their hyperparameters
- State what happens to bias and variance if we tune these hyperparameters
- State how “one-vs-one” and “one-vs-rest” approaches for multi-class problems are implemented.

Support Vector Machines

DSI SEA5, jf.omhover, Sep 30 2016



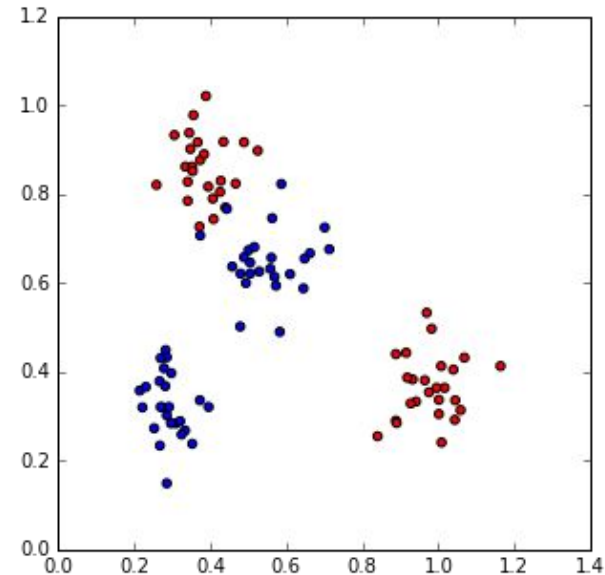
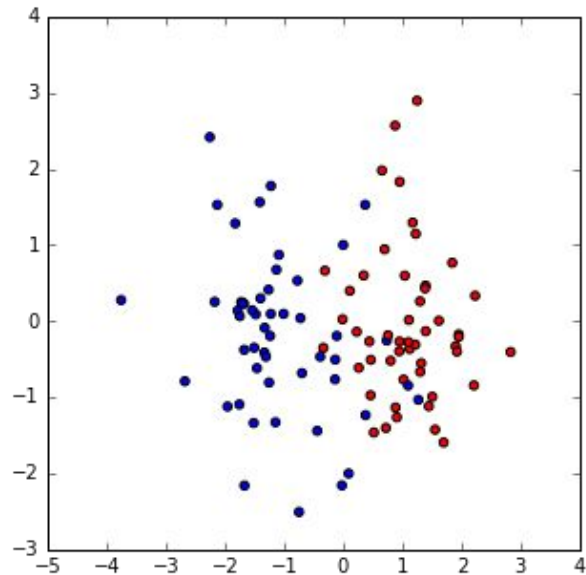
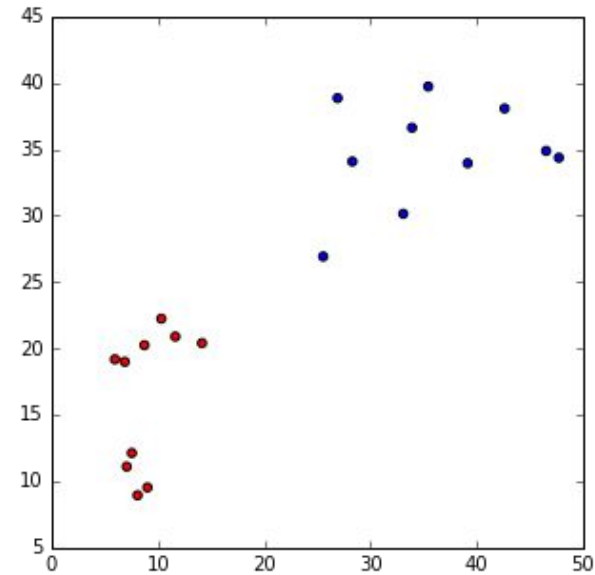
OBJECTIVES

- **Understand** the notion of decision boundaries
- **Describe** the function and parameters of SVMs
- **Investigate** some of the maths behind SVMs
- **Extend** SVMs by soft margins and kernel tricks
- **Investigate** how SVMs perform in terms of Bias-Variance
- Get your **mind blown**

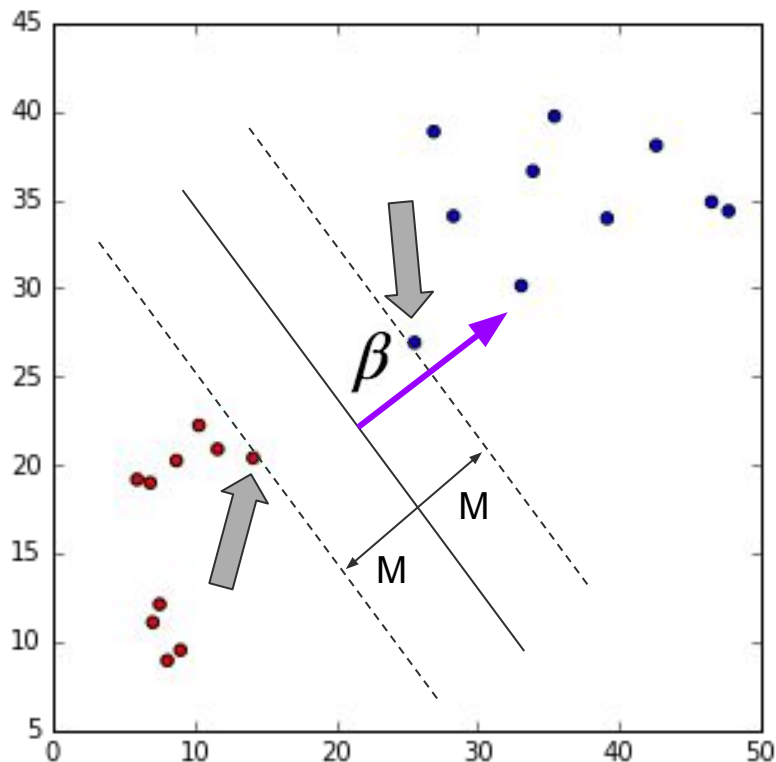
Brainstorm : what's a good decision boundary ?



MMC



Maximum Margin Classification



$$\max_{\beta_0, \dots, \beta_p} M$$

$$\text{subject to } \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i \cdot (\beta_0 + \beta_1 \cdot x_{i,1} + \dots + \beta_p \cdot x_{i,p}) \geq M$$

$$y_i \cdot (\beta_0 + x_i^T \cdot \beta) \geq M$$

Points that condition the margin.

Points that have a direct influence on the margin.

Points that end up being the closest to the hyperplane.

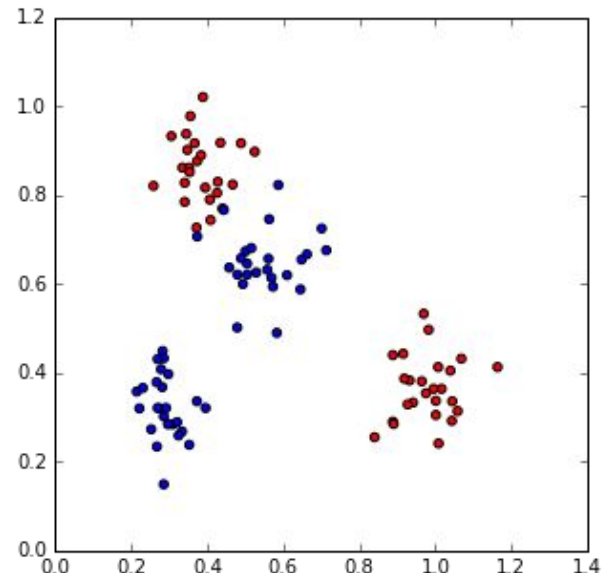
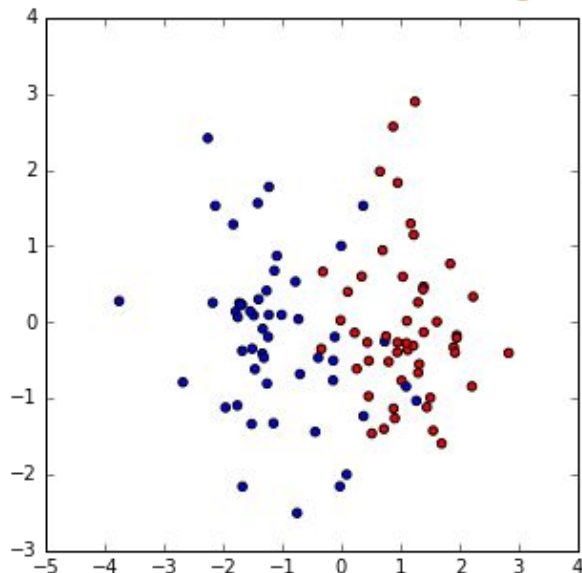
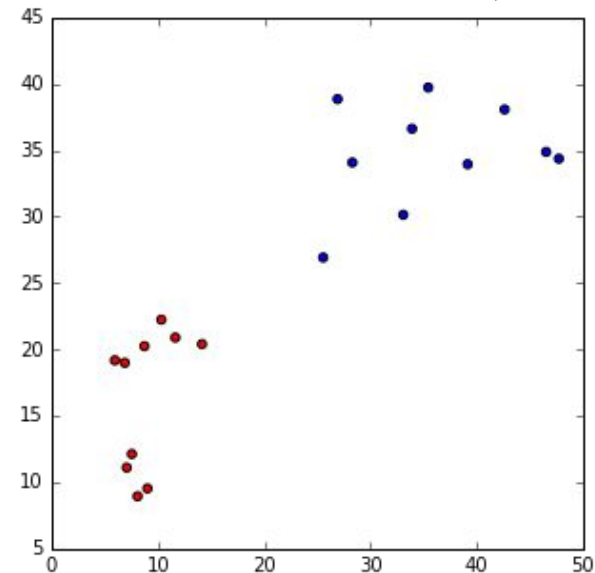
Brainstorm : what's a good decision boundary ?



MMC



SVC

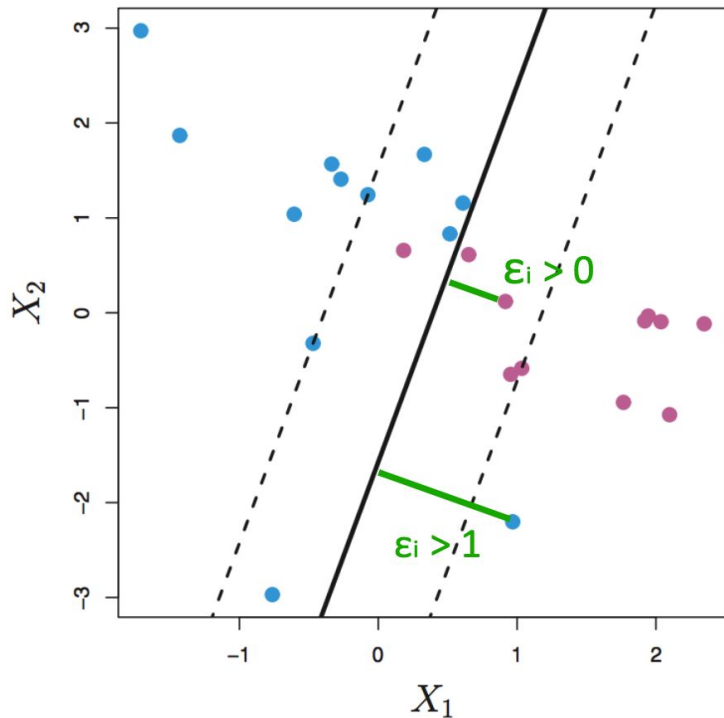




Support Vectors Classifier

introducing soft margins

Soft Margins / Support Vectors Classifier



Relax the constraints for a limited number of vectors

C : our “budget” to spend on relaxing this constraint

Each vector can expense ϵ_i on the margin (slack)

$$\max_{\beta_0, \dots, \beta_p, \epsilon_1, \dots, \epsilon_p} M$$

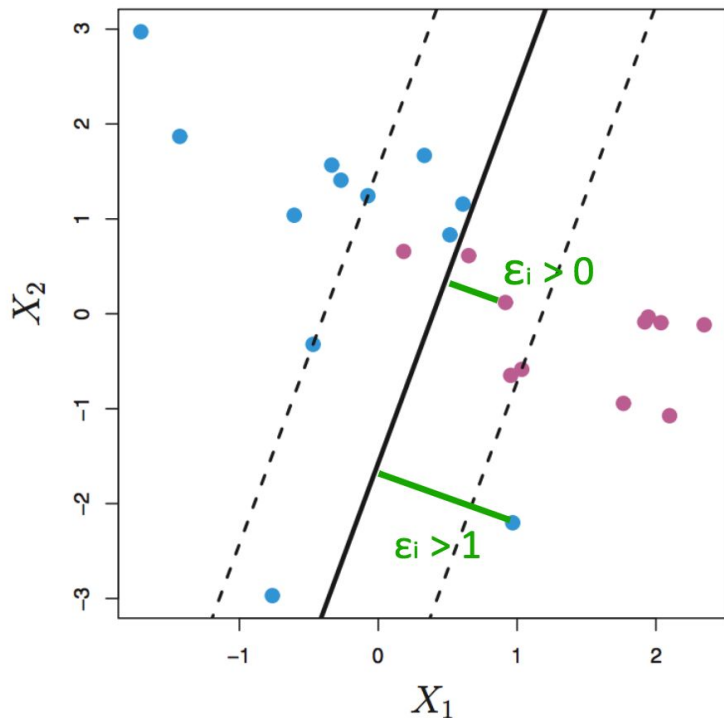
$$\text{subject to } \sum_{j=1}^p \beta_j^2 = 1$$

$$y_i \cdot (\beta_0 + \beta_1 \cdot x_{i,1} + \dots + \beta_p \cdot x_{i,p}) \geq M(1 - \epsilon_i)$$

$$y_i \cdot (\beta_0 + x_i^T \cdot \beta) \geq M(1 - \epsilon_i)$$

$$\text{subject to } \forall i, \epsilon_i \geq 0 \text{ and } \sum_{i=1}^n \epsilon_i \leq C$$

How to solve that ?



Namedropping : Lagrange dual objective function

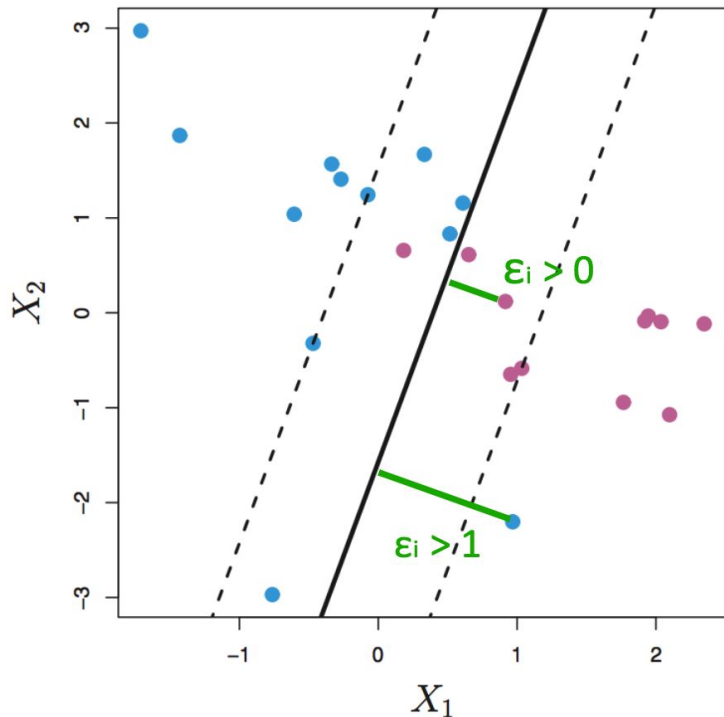
$$L_D = \sum_i^n \alpha_i - 1/2 \sum_i^n \sum_{i'}^n \alpha_i \alpha_{i'} y_i y_{i'} \langle x_i, x_{i'} \rangle$$

$f_{\beta_0, \beta_1, \dots, \beta_p}$ is a solution to that maximization problem

$$f_{\beta_0, \beta_1, \dots, \beta_p}(x) = \beta_0 + \sum_{i=1}^p \alpha_i \langle x, x_i \rangle$$

Alpha is non zero only for support vectors

Soft Margins / Support Vectors



“Support vectors” :

Only the observations that lie on the margin or violate it will affect the hyperplane

Large C : many support vectors

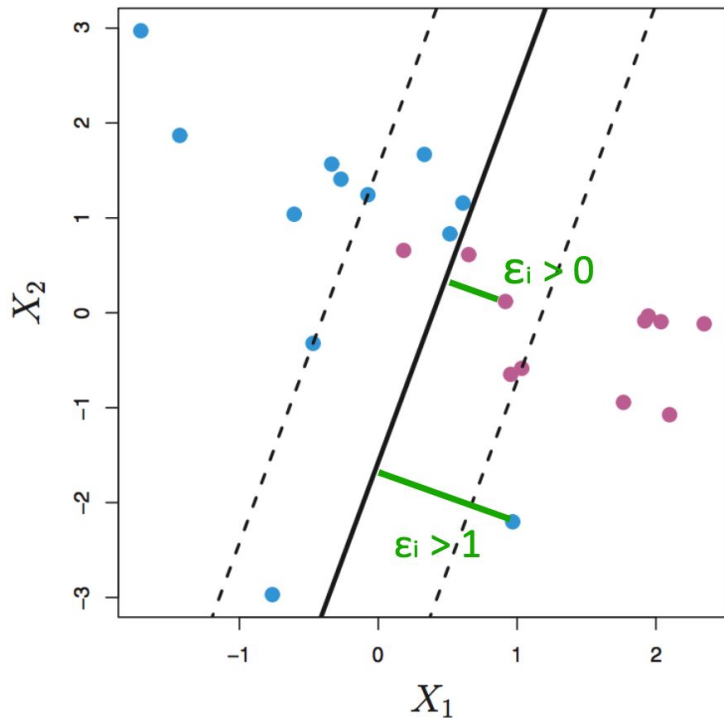
Small C : few support vectors

$f_{\beta_0, \beta_1, \dots, \beta_p}$ is a solution to that maximization problem

$$f_{\beta_0, \beta_1, \dots, \beta_p}(x) = \beta_0 + \sum_{i=1}^p \alpha_i \langle x, x_i \rangle$$

Alpha is non zero only for support vectors

Soft Margins / SVC / Hyperparameter C



Relax the constraints for a limited number of vectors

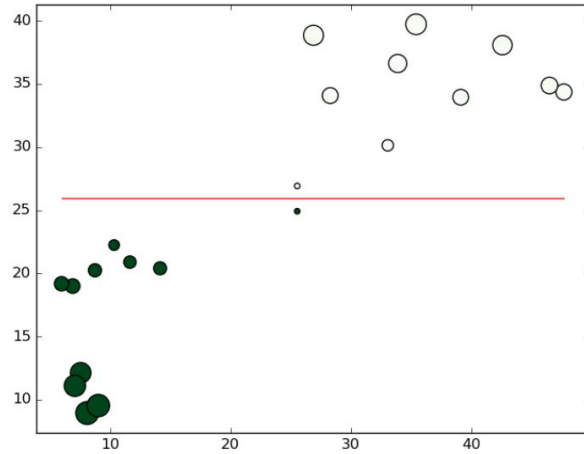
C : our “budget” to spend on relaxing this constraint

Each vector can expense ϵ_i on the margin (slack)

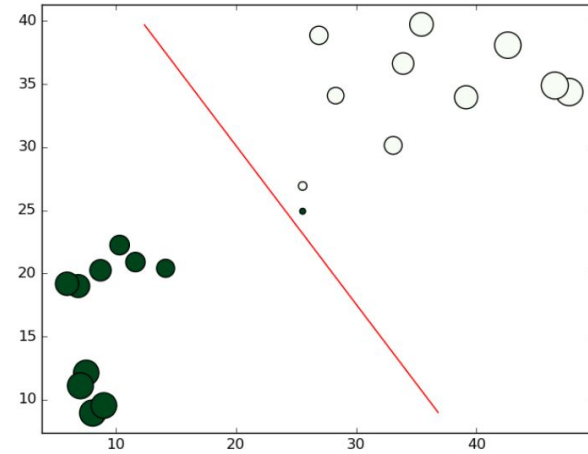
Small C : narrow margins, highly fit,
low bias, high variance

Large C : wider margins, fitting less
hard, high bias, low variance

Soft Margins / Hard Margins



Hard Margin



Soft Margin

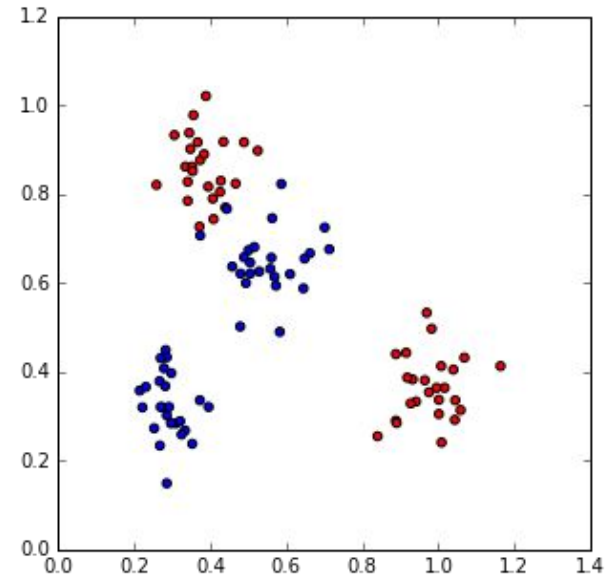
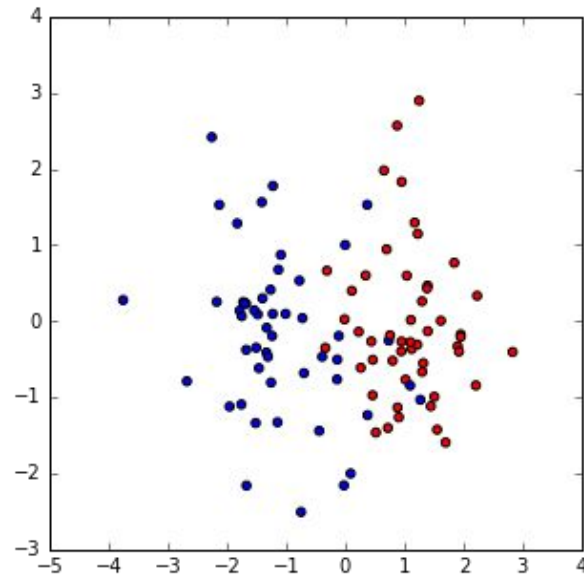
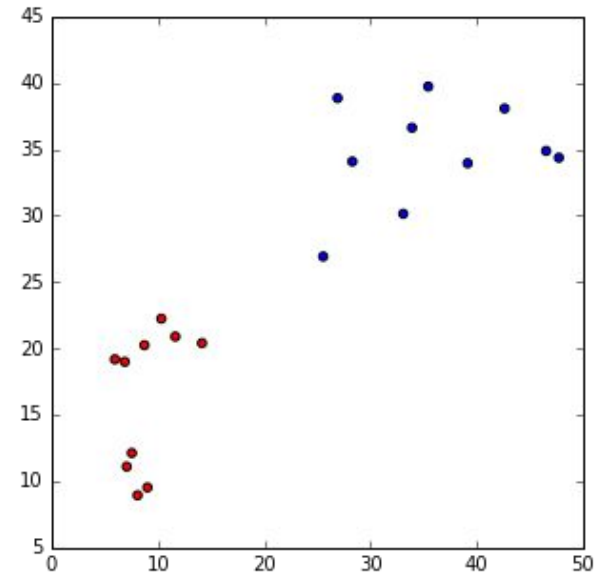
Brainstorm : what's a good decision boundary ?



MMC



SVC





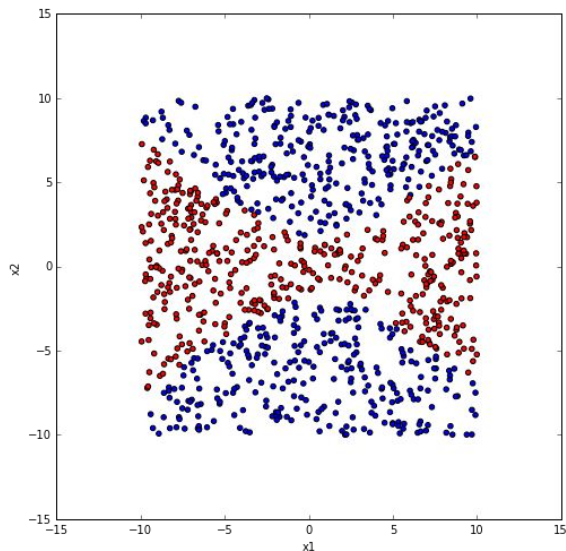
Support Vectors Machines

introducing kernels

Lyrics of the Kernel Trick song

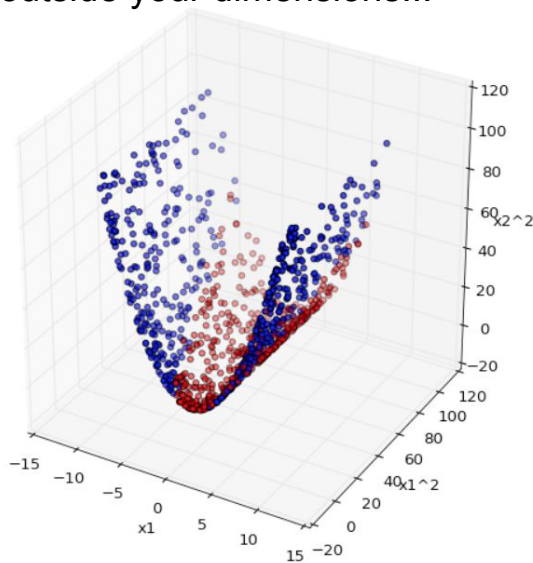


Life is not linearly separable...



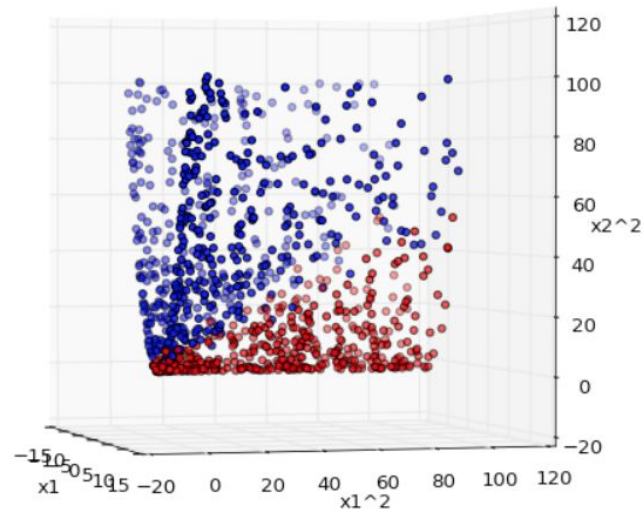
(x_1, x_2)

But maybe by thinking
outside your dimensions...



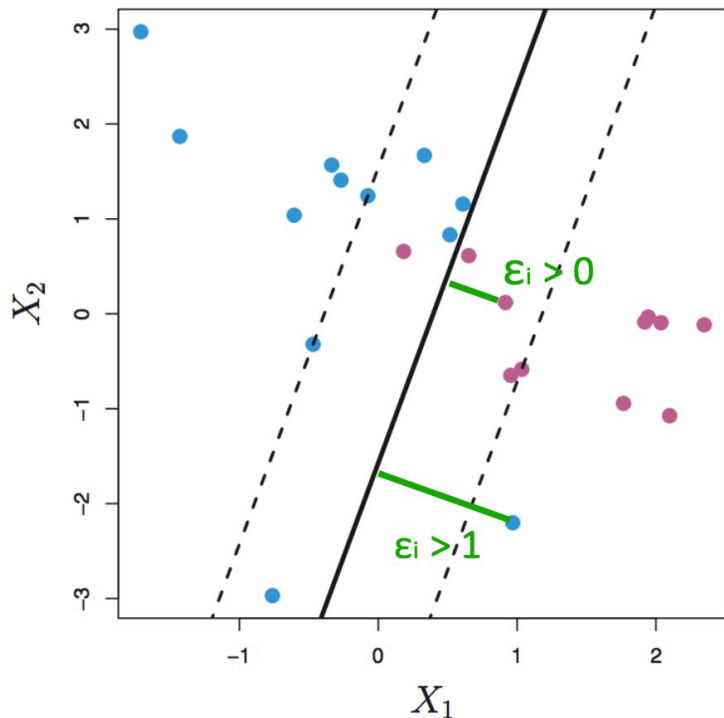
(x_1, x_2, x_1^2, x_2^2)

It is...



$(x_1, x_2, x_1^2, x_2^2) > 0$

Sol 1: Expanding in the polynomial realm...



Adding to the input space...

$$\max_{\beta_0, \dots, \beta_p, \beta_{2,1}, \dots, \beta_{2,p}, \epsilon_1, \dots, \epsilon_p} M$$

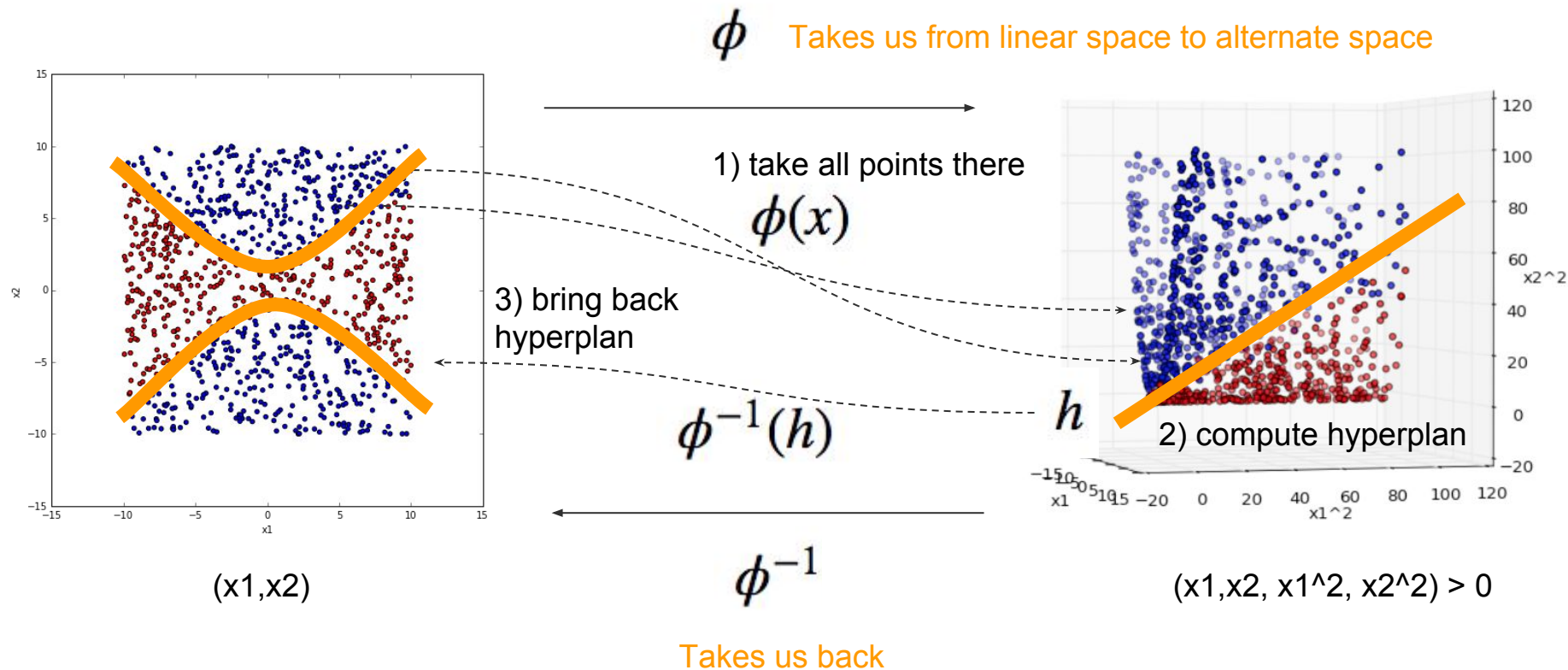
$$\text{subject to } \sum_{j=1}^p \beta_j^2 + \sum_{j=1}^p \beta_{2,j}^2 = 1$$

$$y_i \cdot (\beta_0 + \sum_{j=1}^p \beta_j \cdot x_{i,j} + \sum_{j=1}^p \beta_{2,j} \cdot x_{i,j}^2) \geq M(1 - \epsilon_i)$$

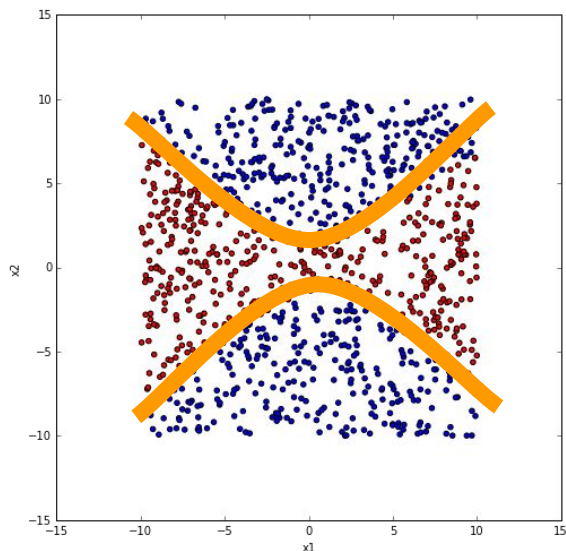
$$y_i \cdot (\beta_0 + x_i^T \cdot \beta) \geq M(1 - \epsilon_i)$$

$$\text{subject to } \forall i, \epsilon_i \geq 0 \text{ and } \sum_{i=1}^n \epsilon_i \leq C$$

Sol 2: Let's devise an alternate space



How to solve that ?



Namedropping : Lagrange dual objective function

$$L_D = \sum_i^n \alpha_i - 1/2 \sum_i^n \sum_{i'}^n \alpha_i \alpha_{i'} y_i y_{i'} K(x_i, x_{i'})$$

$$K(x, x_i) = \langle \phi(x), \phi(x_i) \rangle$$

$f_{\beta_0, \beta_1, \dots, \beta_p}$ is a solution to that maximization problem

$$f_{\beta_0, \beta_1, \dots, \beta_p}(x) = \beta_0 + \sum_{i=1}^p \alpha_i \langle \phi(x), \phi(x_i) \rangle$$

Alpha is non zero only for support vectors

Kernels we use...



Solution to SVC only involves inner product of observations

$$\langle x_i, x_{i'} \rangle = \sum_{j=1}^p x_{ij} x_{i'j}$$

$$f(x) = \beta_0 + \sum_{i=1}^n \alpha_i \langle x, x_i \rangle \leftarrow \text{SVC}$$

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle \quad \leftarrow \text{Only requires support vectors}$$

More generally, instead of just taking inner product, we can use *Kernels*

$$f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i K(x, x_i) \quad \leftarrow \text{SVM, since using Kernels now}$$

$$K(x_i, x_{i'}) = \sum_{j=1}^p x_{ij} x_{i'j} \quad \text{Linear Kernel}$$

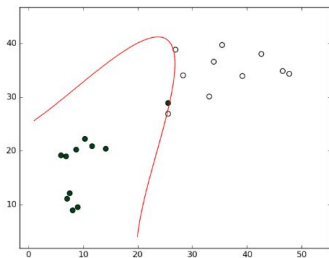
$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^p x_{ij} x_{i'j}\right)^d \quad \text{Polynomial Kernel}$$

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^p (x_{ij} - x_{i'j})^2\right) \quad \text{Radial Basis Function Kernel ("Gaussian")}$$

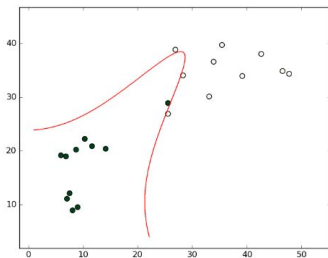
$$K(x^{(i)}, x^{(j)}) = (1 + x^{(i)} \cdot x^{(j)})^d$$

- ▶ equivalent to the dot product in the d -order ϕ space
- ▶ requires an extra hyper-parameter, d , for “degree”

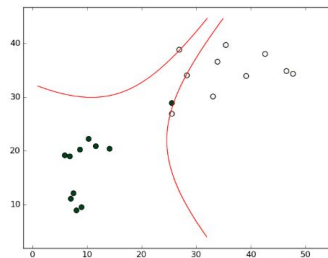
Polynomial Kernel



SVC =
SVC(C=10000.0,
kernel='poly',
degree=3)



SVC =
SVC(C=10000.0,
kernel='poly',
degree=5)



SVC =
SVC(C=10000.0,
kernel='poly',
degree=10)

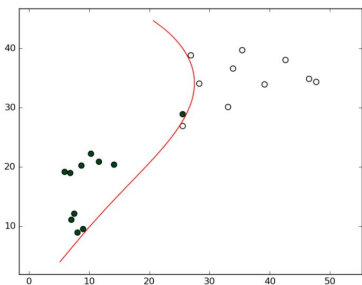
RBF Kernel (Radial Basis Function)



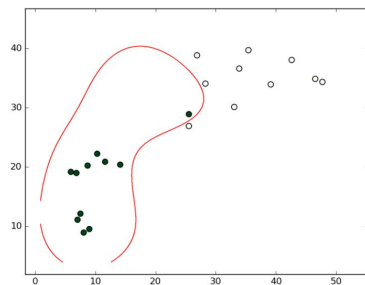
$$K(x^{(i)}, x^{(j)}) = \exp(-\gamma \|x^{(i)} - x^{(j)}\|^2)$$

- ▶ equivalent to the dot product in the Hilbert space of infinite dimensions
- ▶ requires an extra hyper-parameter, γ , “gamma”

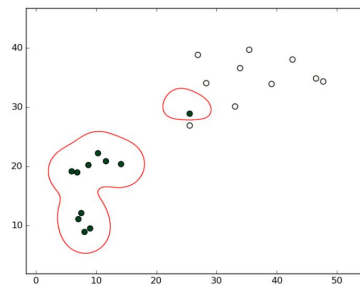
RBF Kernel (Radial Basis Function)



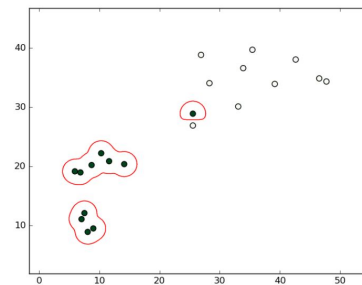
`svc = SVC(C=10000.0,
kernel='rbf', gamma=0.001)`



`svc = SVC(C=10000.0,
kernel='rbf', gamma=0.01)`

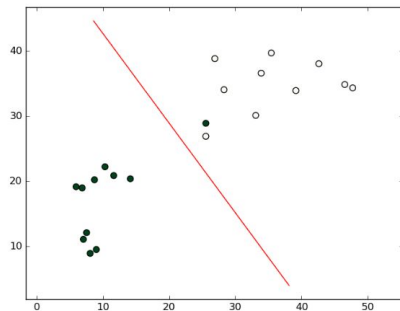


`svc = SVC(C=10000.0,
kernel='rbf', gamma=0.1)`

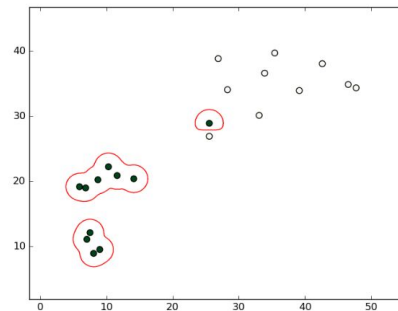


`svc = SVC(C=10000.0,
kernel='rbf', gamma=1.0)`

Best fit ?



```
svc = SVC(C=0.01,  
kernel="linear")
```



```
svc = SVC(C=10000.0,  
kernel='rbf', gamma=1.0)
```

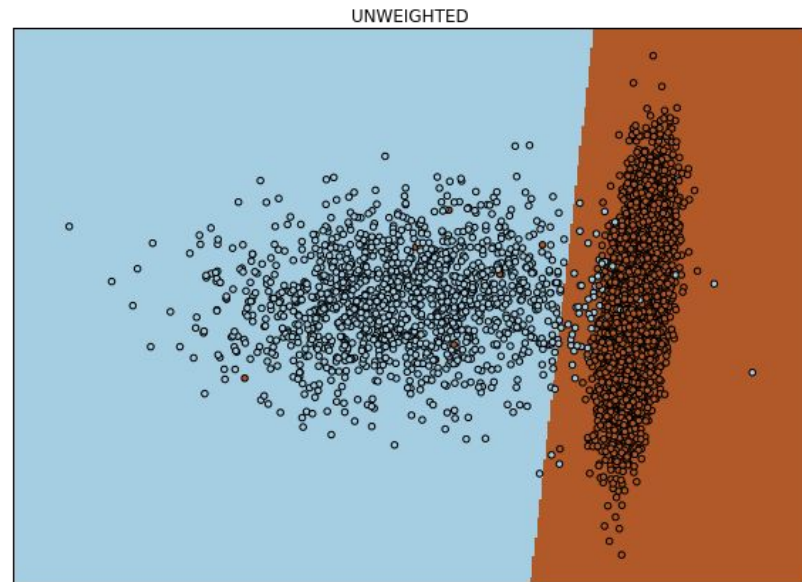



Adaptation of SVM to real-world problems

Unbalanced classes



Adjusting weights (beta)
so that they are inversely proportional
to class representativity



One-versus-One :

Train a classifier on all pairs of classes

Use each pair for determining the class of an observation (aggregating)

One-versus-All / One-versus-Rest :

Train a classifier on each class (+1), considering all other as the rest (-1)

Use this classifier for determining if an observation fits in that class



Pair Assignment

```
from sklearn.svm import SVC
```

```
svm = SVC(kernel='linear').fit(X, y)
```

```
svm = SVC(kernel='linear', C=x).fit(X,y)
```

```
svm = SVC(kernel='poly', C=c, degree=5).fit(X,y)
```

```
svm = SVC(kernel='rbf', C=c, gamma=0.5).fit(X,y)
```