Data Science Training

November 2017

Support Vector Machine Techniques

Xavier Bresson

Data Science and AI Research Centre
NTU, Singapore

http://data-science-optum17.tk

Outline

- **Learning techniques**
- Linear SVM
- Soft-Margin SVM
- Kernel Techniques
- Non-Linear/Kernel SVM
- Graph SVM
- Conclusion

Classes of Learning Techniques

Q: Do you know the difference between unsupervised, supervised, and semi-supervised learning?

- **Three classes of learning algorithms** (for clustering, classification, recommendation, visualization, feature extraction, etc):
  - *(1) Unsupervised learning:* Algorithms that do **not** use any prior information.
  - *(2) Supervised learning:* Algorithms that **only** use labeled data.
  - *(3) Semi-supervised learning:* Algorithms that use **both** labeled and unlabeled data.
- **Labeled data** are gold data but they are generally expensive to produce, while **unlabeled data** are usually cheap to get. Ex: Image recognition in Computer Vision, millions of images are available with Google, but they are unlabeled. How much time/money to label 1,000 images?
- This lecture focuses on **supervised and semi-supervised data classification,** particularly SVM techniques.

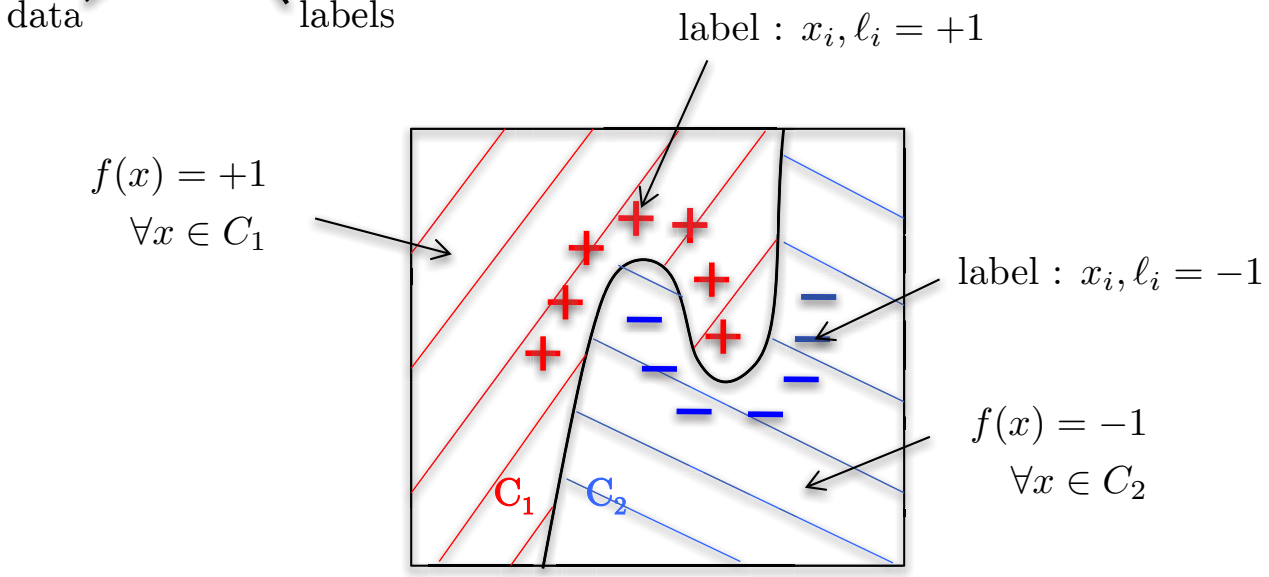
Support Vector Machine (SVM)

- SVM is a very **popular** classification technique (among top 10 algorithms in data science). SVM is used in deep learning as loss function (later discussed).

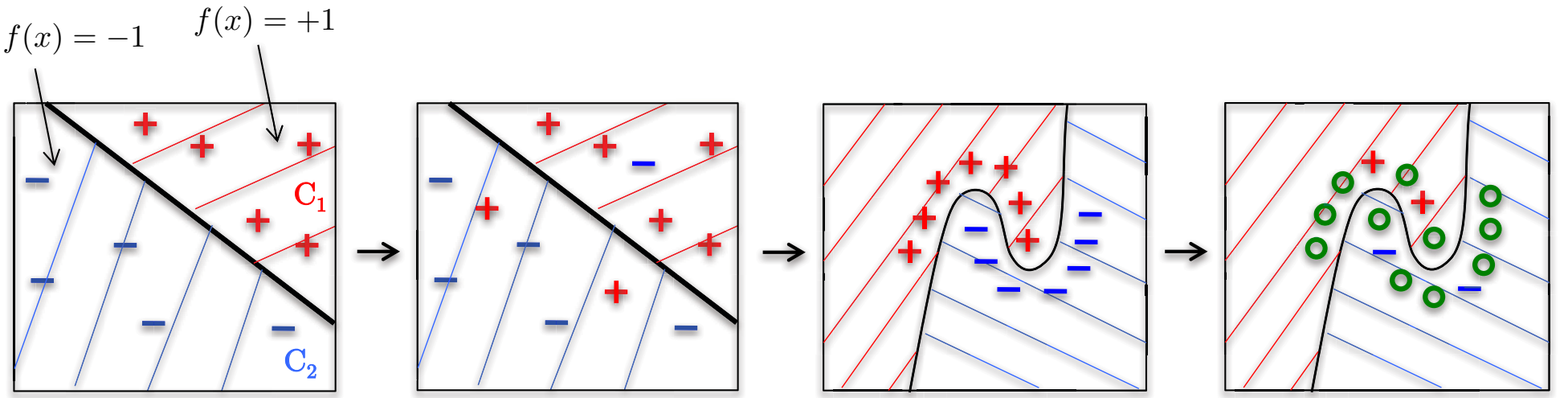
- We will cover the supervised and semi-supervised **binary SVM classification** techniques: *Given a set of labeled data that belongs to two classes, construct a classification function that outputs the class of any new data.*

$$V = \{x_i, \ell_i\}_{i=1}^{n},\ x_i \in \mathbb{R}^d,\ \ell_i \in \{-1, +1\}$$

data ↗ ↖ labels



History of SVM Techniques



Linear SVM
Supervised Learning
[Vapnik-Chervonenkis'63]

Soft-Margin SVM
Supervised Learning
[Cortes-Vapnik'95]

Non-Linear/Kernel SVM
Supervised Learning
[Boser,Guyon,Vapnik-Chervonenkis'92]

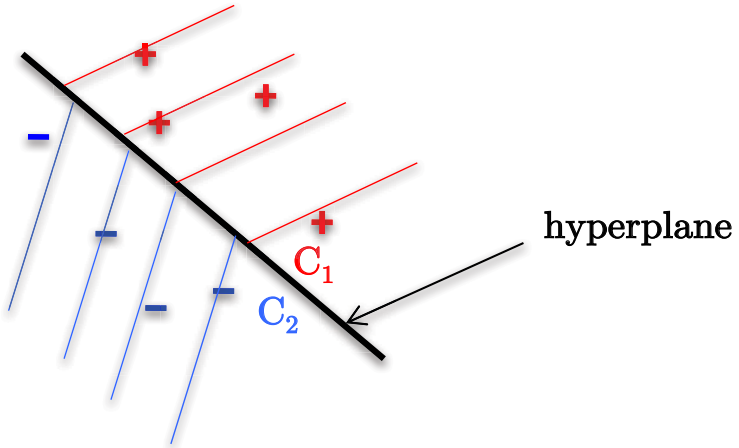
Laplacian SVM
Semi-Supervised Learning
[Belkin,Niyogi,Sindhwari'06]

Outline

- Learning techniques
- **Linear SVM**
- Soft-Margin SVM
- Kernel Techniques
- Non-Linear/Kernel SVM
- Graph SVM
- Conclusion

Linear SVM [Vapnik, Chervonenkis '63]

- **Assumption:** Training (and test) data are **linearly separable**, i.e. data can be perfectly separated by a hyperplane:



- **Linear SVM:** *Given a training dataset* $V=\{x_i,l_i\}$, *design a classification function that assigns any new data* x *to the class that is best consistent with* V.

$$f : x \in \mathbb{R}^d \rightarrow \{-1, +1\}$$

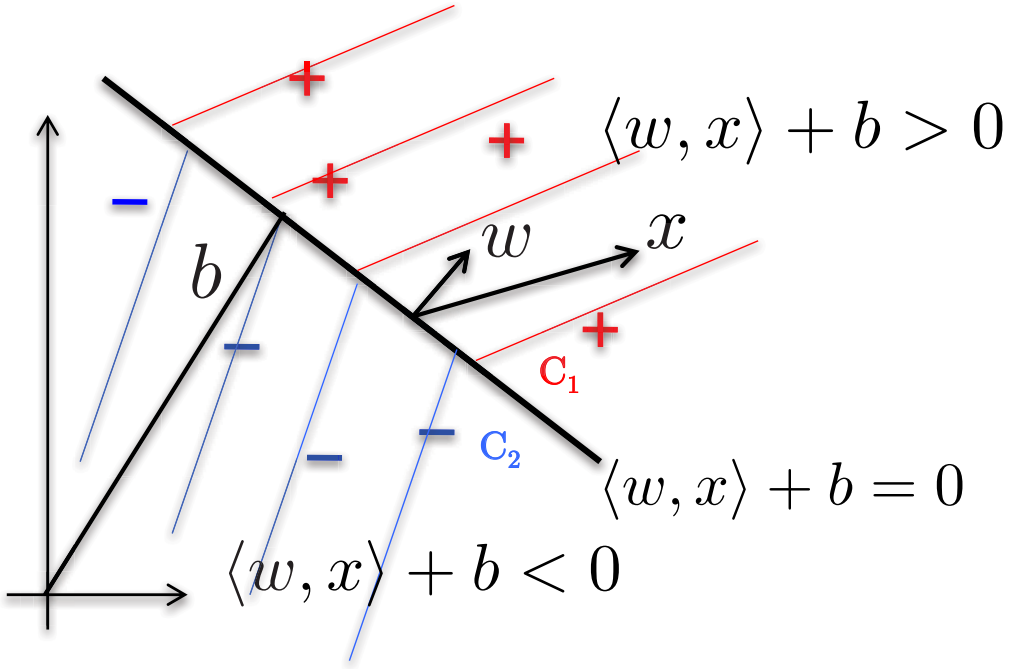
- **Class of (linear) solutions:** Given the assumption, determine the hyperplane that best separates the two classes. Any hyperplane is **parameterized by two variables** ***(w,b)***, where w is the normal vector of hyperplane, and b is the offset value.

Hyperplane equation: $\langle w, x \rangle + b = 0$

Linear SVM Classifier

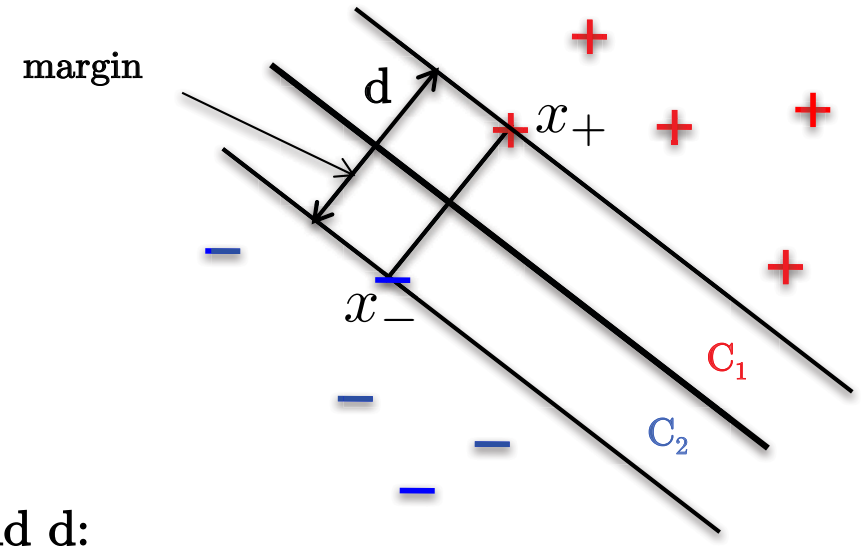
- **Classification function:**

$$f(x) = \text{sign}(\langle w, x \rangle + b) = \begin{cases} +1 & \text{if } x \in C_1 \\ -1 & \text{if } x \in C_2 \end{cases}$$



How to Find (w,b)?

- **SVM idea:** Define the best hyperplane by **maximizing the margin d** between the 2 classes:



- **Relationship between w and d:**

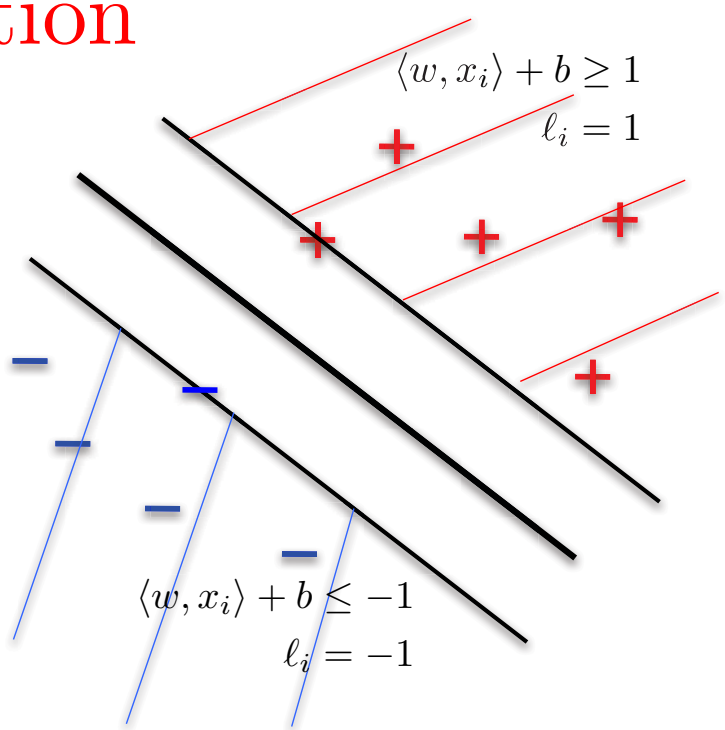
$$\begin{array}{r} \langle w, x_+ \rangle + b = +1 \\ \langle w, x_- \rangle + b = -1 \\ \hline \langle w, x_+ - x_- \rangle = 2 \end{array}$$

$$\vec{d} = x_+ - x_- = \alpha w \quad \rightarrow \quad \alpha = \frac{2}{\|w\|_2^2} \quad \rightarrow \quad d = \frac{2}{\|w\|_2}$$

$\Rightarrow$ **Maximize the margin d:** $\max\ d \Leftrightarrow \max \frac{2}{\|w\|_2} \Leftrightarrow \min\ \|w\|_2^2$

$\Leftrightarrow$ **Minimize the weight w**

Primal Optimization



- Variable w is called **primal variable**.
  Optimization problem w.r.t. w is $\min_w \|w\|_2^2$
  with constraints:

$$f_i = \langle w, x_i \rangle + b = \left\{ \begin{array}{ll} \geq +1 & \text{if } x \in C_1 \\ \leq -1 & \text{if } x \in C_2 \end{array} \right\} \rightarrow \quad \ell_i . f_i \geq 1 \ \forall i \in V$$

$$\ell_i = \left\{ \begin{array}{ll} +1 & \text{if } x \in C_1 \\ -1 & \text{if } x \in C_2 \end{array} \right.$$

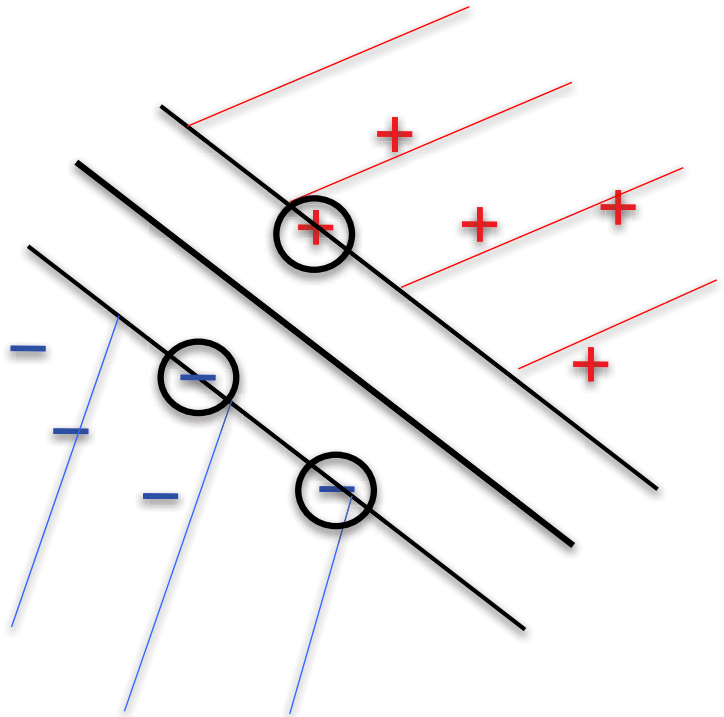
- **Summary:** SVM classifier f is given by the solution of the **Quadratic Programming** (QP) problem:

$$\min_{w,b} \|w\|_2^2 \ \text{ s.t. } \ \ell_i . f_i \geq 1 \ \forall i \in V \quad \rightarrow \quad f(x) = \langle w, x \rangle + b$$

Quadratic function — **Convex set (polytope)** — **SVM classifier**

Support Vectors

- **Definition:** Data that are exactly localized on the margin hyperplanes.



- **Offset value *b*:** b is defined by:

$$\ell_i.(\langle w, x_i^{SP}\rangle + b) - 1 = 0,\ \forall x_i^{SP} \rightarrow \quad b_i = \ell_i - \langle w, x_i^{SP}\rangle$$

$$\downarrow$$

$$b = \mathbb{E}(\{b_i\})$$

Dual Problem

- **Primal problem:** $\min_{w,b} \|w\|_2^2 \quad \text{s.t.} \quad \ell_i . f_i \geq 1 \ \forall i \in V$

$$\Updownarrow$$

- **Dual problem:** The *dual variable* is α.

  Motivation: Work with data products $\langle x_i, x_j \rangle$ (kernel trick).

After some computations... $\min_{\alpha \geq 0} \frac{1}{2} \alpha^T Q \alpha - \langle \alpha, 1 \rangle \quad \text{s.t.} \quad \langle \alpha, 1 \rangle = 0$ *QP problem*

With: $Q = LKL$

$L = \text{diag}(\ell_1, ..., \ell_n)$

$K_{ij} = \langle x_i, x_j \rangle$

Linear kernel

Optimization Algorithm

- **Classification function:**

$$f(x) = \text{sign}(\langle w^\star, x\rangle + b^\star)$$
$$= \text{sign}(\alpha^{\star T} LK(x) + b^\star)$$

- **Optimization scheme:** Solution α^* given by iterative scheme:

Initalization: $\alpha^{l=0} = y^{l=0} = 0 \qquad \tau = \frac{1}{\|Q\|},\ \sigma = \frac{1}{\|L\|}$

Iterate until convergence: l=0,1,2,...

$$\alpha^{l+1} = P_{.\geq 0}[(\tau Q + I_n)^{-1}(\alpha^k + \tau - \tau L y^k)]$$
$$y^{l+1} = y^k + \sigma L \alpha^{l+1}$$

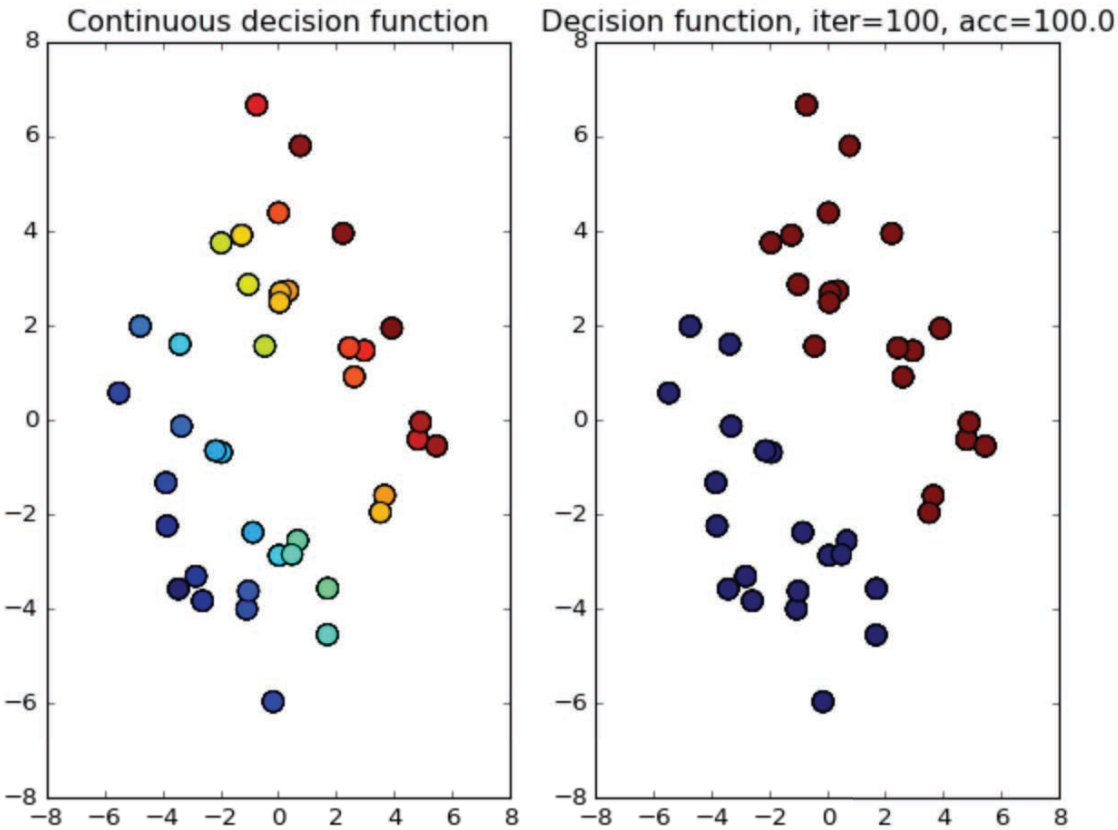
$$\rightarrow \quad \alpha^\star = \alpha^\infty$$

Demo: Standard/Linear SVM

- Run **code01.ipynb**

```
In [4]: # Run Linear SVM

        # Compute linear kernel, J, Q
        Ker = Xtrain.dot(Xtrain.T)
        L = np.diag(l_train)
        l = l_train
        Q = L.dot(Ker.dot(L))
```



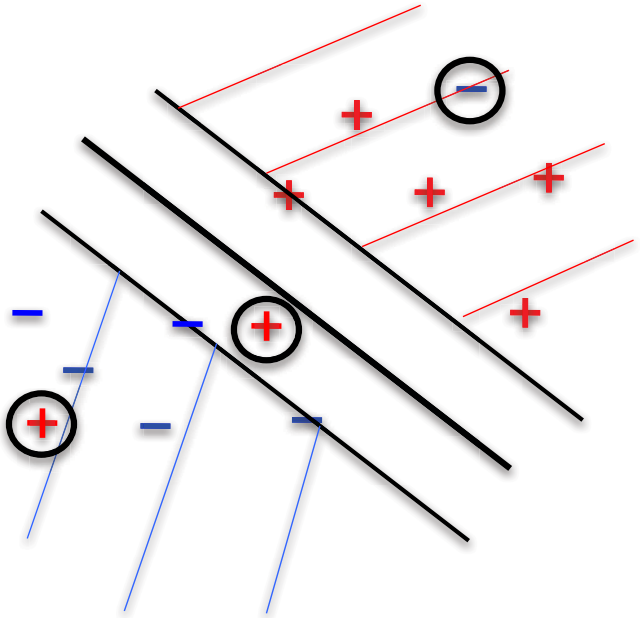
Outline

- Learning techniques
- Linear SVM
- **Soft-Margin SVM**
- Kernel Techniques
- Non-Linear/Kernel SVM
- Graph SVM
- Conclusion

Soft-Margin SVM [Cortes-Vapnik '95]

- **Motivation:** Linear SVM suppose data are linearly separable, i.e. there exist an hyperplane separating **perfectly** the data.

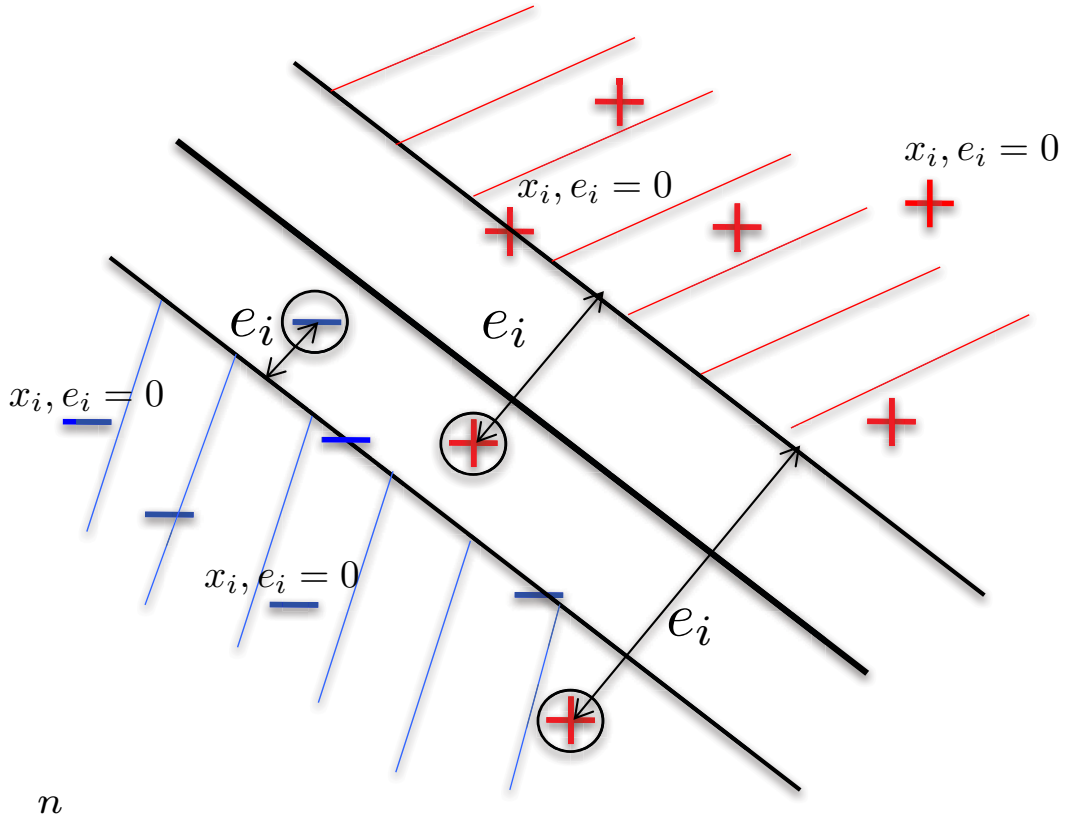
Q: What happens in the presence of outliers?



- **Soft SVM:** Find an hyperplane that best separate the data (by maximizing the margin) while **allowing as few outliers as possible**.

Modeling Errors with Slack Variables

- **Slack variables e_i:** Measure the error of each data x_i to be an outlier.



- **New optimization:**

$$\min_{w,b} \underbrace{\|w\|_2^2 + \gamma \sum_{i=1}^{n} e_i}_{\text{Trade-off between large margin and small errors}} \quad \text{s.t.} \quad \ell_i . f_i \geq 1 - e_i, \; e_i \geq 0 \; \forall i \in V$$

Dual Problem

- **Primal problem:** $\min_{w,b} \|w\|_2^2 + \gamma \sum_{i=1}^{n} e_i \quad \text{s.t.} \quad \ell_i . f_i \geq 1 - e_i, \; e_i \geq 0 \; \forall i \in V$

- **Dual problem:**

After some computations... $\min_{0 \leq \alpha \leq \gamma} \frac{1}{2} \alpha^T Q \alpha - \langle \alpha, 1 \rangle \quad \text{s.t.} \quad \langle \alpha, 1 \rangle = 0$

Only this trivial modification!

With: $Q = LKL$

$L = \text{diag}(\ell_1, ..., \ell_n)$

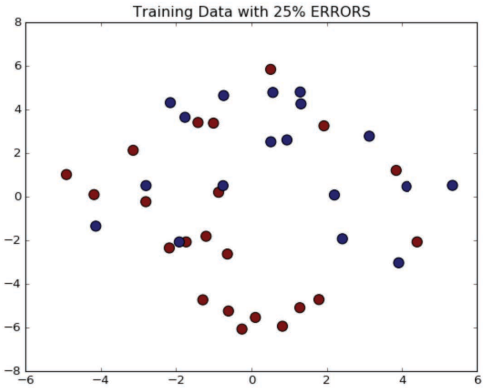
$K_{ij} = \langle x_i, x_j \rangle$

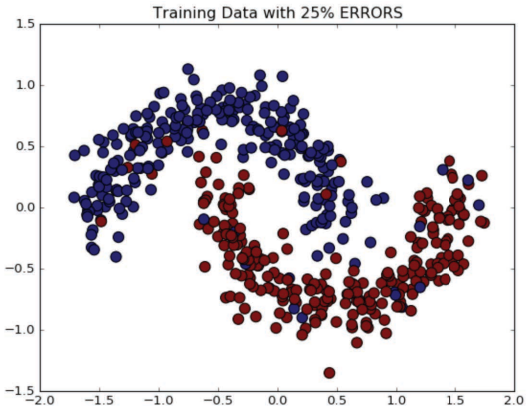
Demo: Soft-Margin SVM

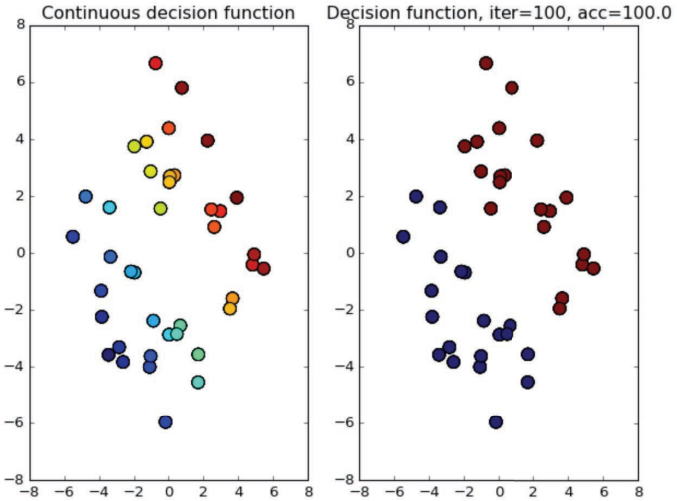
- Run **code02.ipynb**

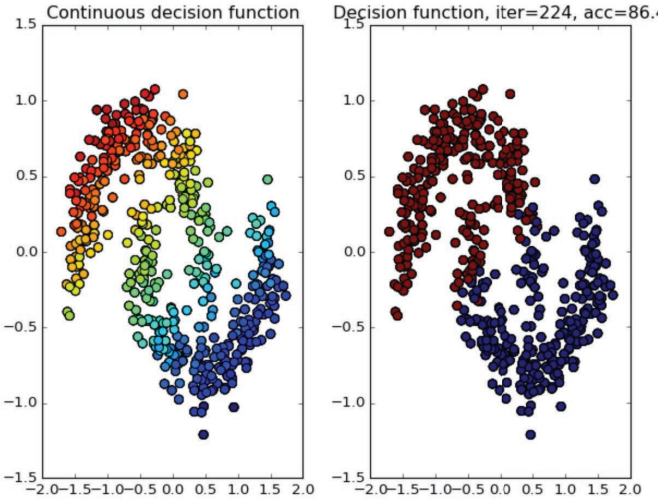
```
In [5]: # Run soft SVM
        _,_,_,_ = compute_SVM(Xtrain,Cgt_train,l_train,'soft_linear',[0.1],Xtest,Cgt_test,[3,100])

Run Linear SVM
Construct Linear Kernel
```









Hinge Loss Function

- **Primal problem:**

$$\min_{w,b} \|w\|_2^2 + \gamma \sum_{i=1}^{n} e_i \quad \text{s.t.} \quad \ell_i.f_i \geq 1 - e_i, \; e_i \geq 0 \; \forall i \in V$$

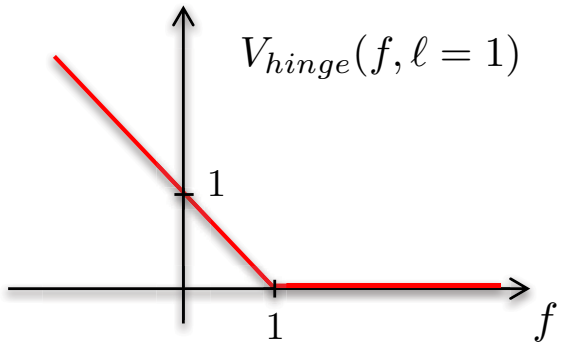
$$\Updownarrow$$

Q: Do you know the hinge loss?

$$\min_{w,b} \|w\|_2^2 + \gamma \sum_{i=1}^{n} V_{hinge}(f_i, \ell_i)$$

$$V_{hinge}(f_i, \ell_i) = \max(0, 1 - f_i.\ell_i)$$

Popular SVM loss function



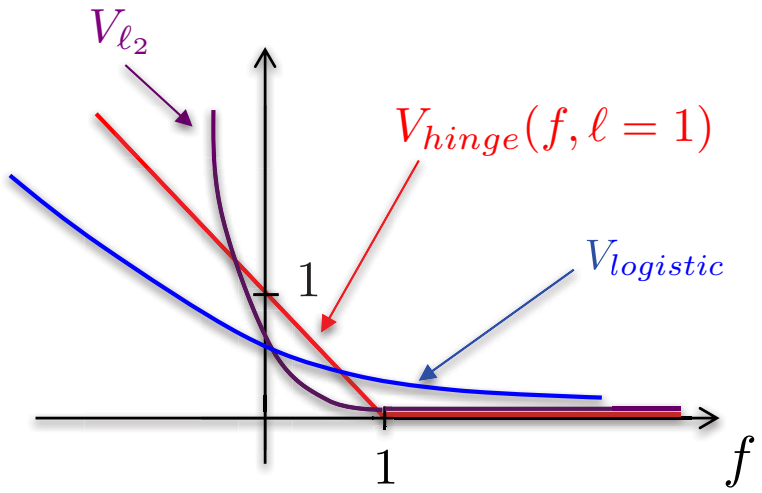
Other Loss Functions

- **Quadratic/L2 loss:**
$$V_{\ell_2}(f_i, \ell_i) = \begin{cases} (1 - f_i.\ell_i)^2 & \text{if } f_i.\ell_i \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

- **Elastic Net loss:**
$$V_{ElasticNet}(f_i, \ell_i) = \begin{cases} (1 - f_i.\ell_i)^2 + \beta|1 - f_i.\ell_i| & \text{if } f_i.\ell_i \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

- **Huber loss:**
$$V_{Huber}(f_i, \ell_i) = \begin{cases} \frac{1}{2} - f_i.\ell_i & \text{if } f_i.\ell_i \leq 0 \\ \frac{1}{2}(1 - f_i.\ell_i)^2 & \text{if } 0 < f_i.\ell_i \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

- **Logistic loss:**
$$V_{Logistic}(f_i, \ell_i) = e^{1 - f_i.\ell_i}$$



Outline

- Learning techniques
- Linear SVM
- Soft-Margin SVM
- **Kernel Techniques**
- Non-Linear/Kernel SVM
- Graph SVM
- Conclusion

Kernel Techniques

- Very popular techniques (until deep learning)

- **Reproducing Kernel Hilbert Space (RKHS):** A space associated to bounded, symmetric, PSD operators called kernels K that can reproduce any smooth function f.

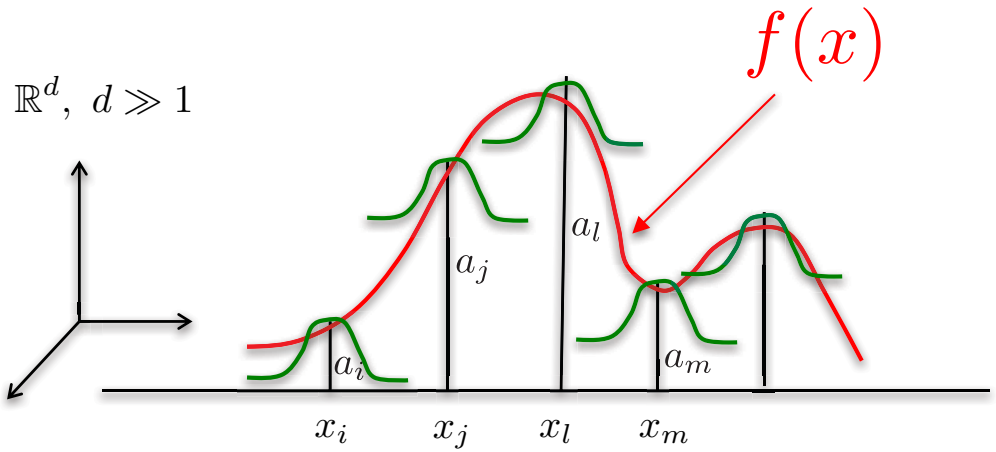
- **Representer Theorem** [Scholkopf, Herbrich, Smola, '01]: Any continuous and smooth function in a RKHS can be represented as a linear combination of the kernel function K evaluated at the data points:

$$f(x) = \sum_{i=1}^{n} a_i K(x, x_i) + b$$

Interpretation of Representer Theorem

- **Interpretation:** The Representer Theorem is a powerful tool for *function interpolation in high-dim spaces*:

$$f(x) = \sum_{i=1}^{n} a_i K(x, x_i) + b$$



with $K(x, x_i) = e^{-\|x - x_i\|_2^2 / \sigma^2}$

- **Popular kernels:**

(1) *Linear kernel:* $K(x, y) = \langle x, y \rangle$

(2) *Gaussian kernel:* $K(x, y) = e^{-\|x - y\|_2^2 / \sigma^2}$

(3) *Polynomial kernel:* $K(x, y) = (a\langle x, y \rangle + b)^c$

Feature Maps and Kernel Trick

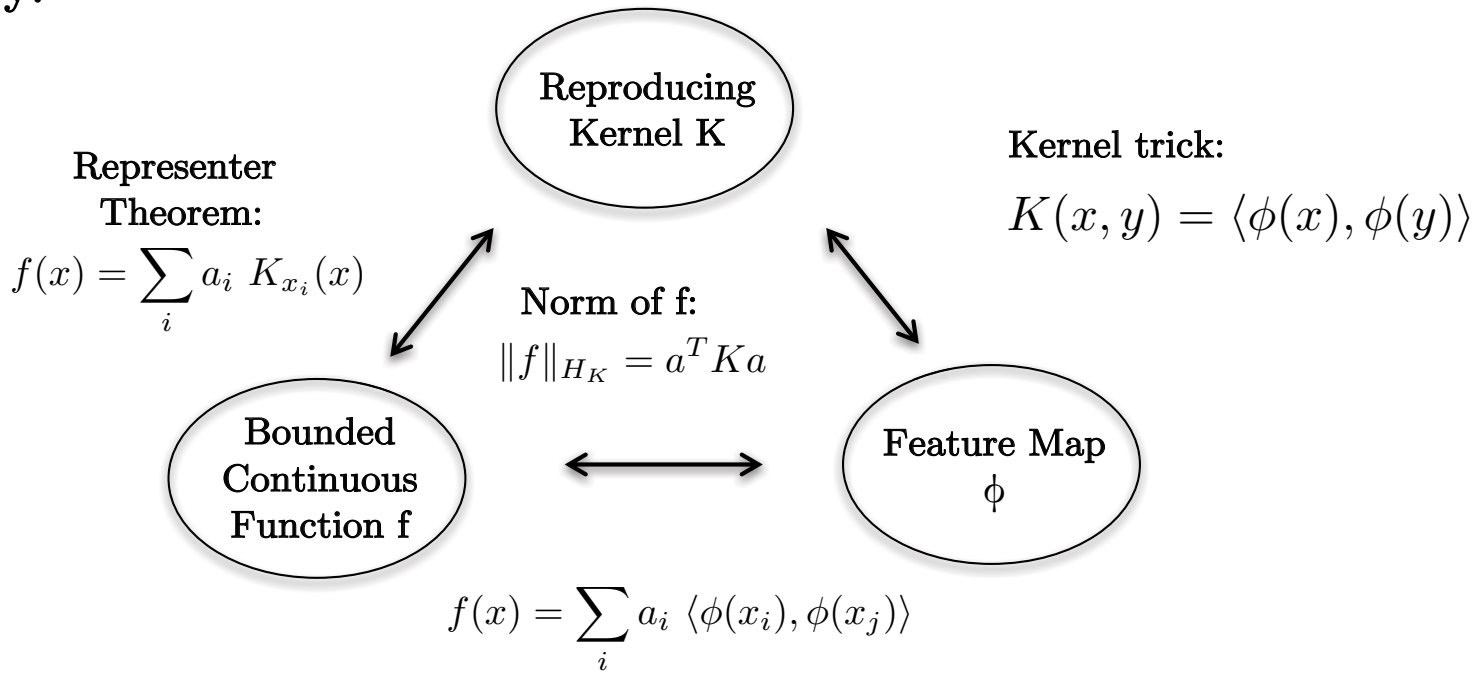
- **Definition:** Any feature map ϕ defines a reproducing kernel K:

$$\langle \phi(x), \phi(y) \rangle \overset{\text{def}}{=} K(x, y)$$

and inversely:

$$K'(x, y) \overset{\text{def}}{=} \langle \phi'(x), \phi'(y) \rangle$$

- **Summary:**



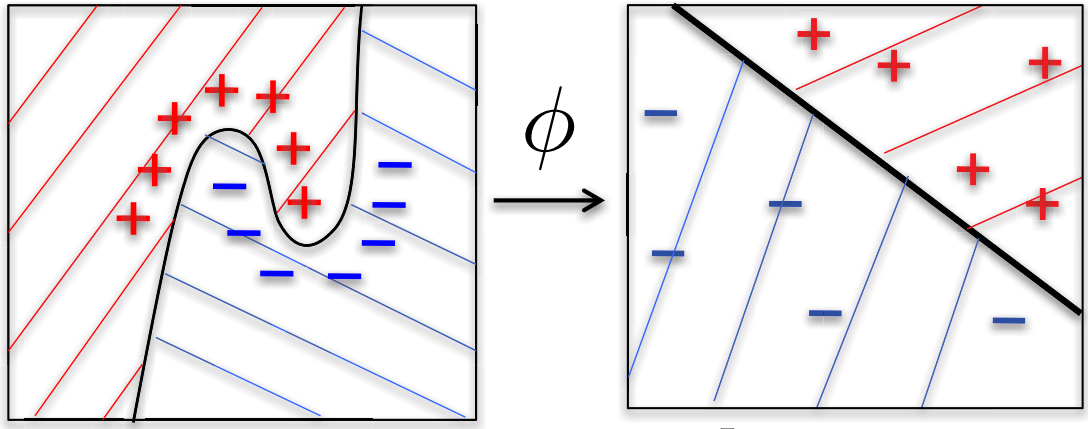
Outline

- Learning techniques
- Linear SVM
- Soft-Margin SVM
- Kernel Techniques
- **Non-Linear/Kernel SVM**
- Graph SVM
- Conclusion

Non-Linear/Kernel SVM
[Bosser, Guyon, Vapnik '92]

- **Motivation:** Linear/soft SVM assume data are linearly separable (up to a few outliers). For several real-world data, the hyperplane assumption is **not** satisfied. A better separation is a non-linear hyperplane, that is a **hypersurface**.

- **Kernel Trick:** Project data into a higher-dim space with feature map ϕ where the data are linearly separable.



- **Decision function in high-dim:**

$$\left.\begin{array}{rcl} f(x) = \langle w, x\rangle + b & \xrightarrow{\phi} & f(x) = \langle w, \phi(x)\rangle + b \\ w = \sum_i \alpha_i \ell_i x_i & \xrightarrow{\phi} & w = \sum_i \alpha_i \ell_i \phi(x_i) \end{array}\right\}$$

$$\longrightarrow \quad f(x) = \sum_i \alpha_i \ell_i \langle \phi(x), \phi(x_i)\rangle + b = \sum_i \alpha_i \ell_i K(x_i, x) + b$$

Kernel

Optimization

- **Dual problem:**

$$\min_{0\leq\alpha\leq\gamma} \frac{1}{2}\alpha^T Q\alpha - \langle\alpha, 1\rangle \quad \text{s.t.} \quad \langle\alpha, 1\rangle = 0$$

With: $Q = LKL$

$L = \text{diag}(\ell_1, ..., \ell_n)$

$$K_{ij} = \langle x_i, x_j\rangle \quad \overset{\phi}{\rightarrow} \quad K_{ij} = \langle\phi(x_i), \phi(x_j)\rangle$$

$\Rightarrow$ Same optimization algorithm ☺

Recall: We never compute explicitly ϕ and the products $\langle\phi(x_i),\phi(x_j)\rangle$, only the Kernel matrix:

$$K(x, y) = \begin{cases} (a\langle x, y\rangle + b)^c \\ e^{-\|x-y\|_2^2/\sigma^2} \end{cases}$$

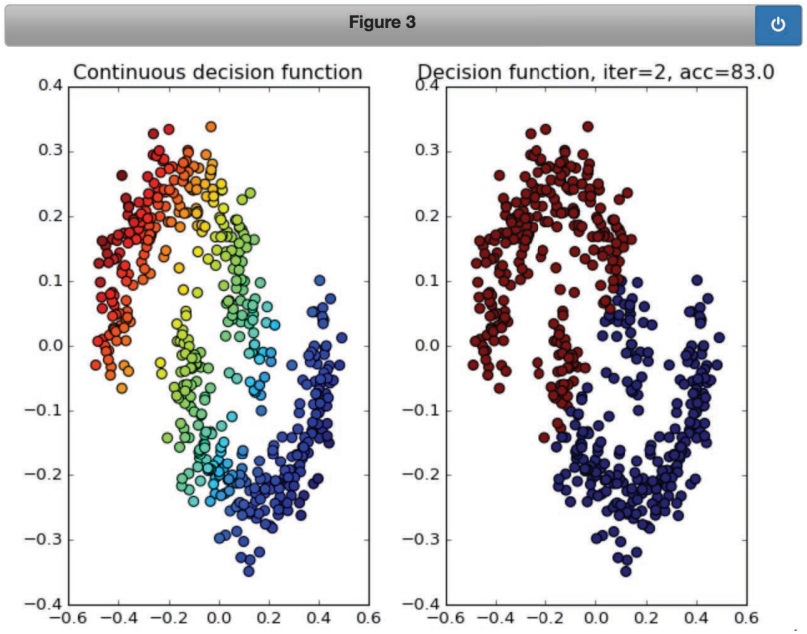
Demo: Kernel/Non-Linear SVM

- Run **code03.ipynb**

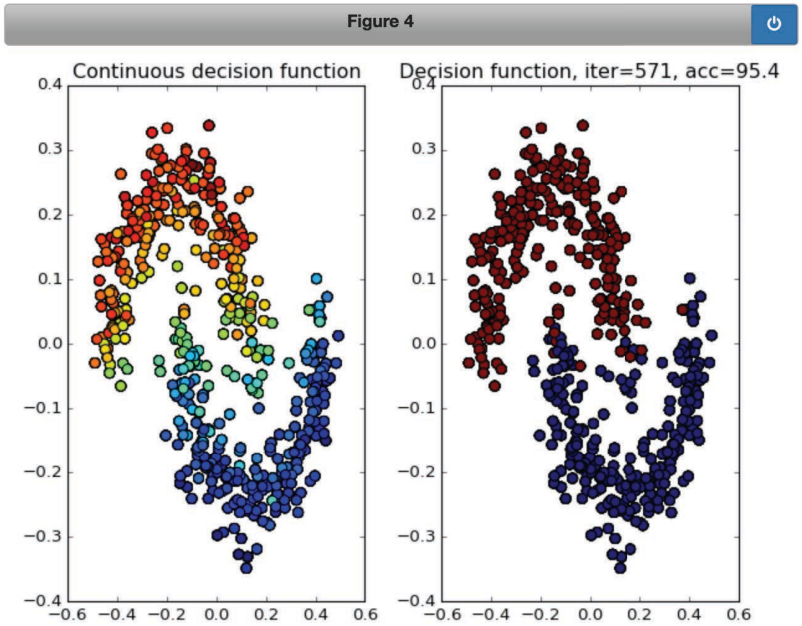
```
In [6]: # Run Kernel SVM
        _,_,_,_ = compute_SVM(Xtrain,Cgt_train,l_train,'gaussian_kernel',[1.0,0.5],Xtest,Cgt_test,[4,50])

Run Kernel SVM
Construct Gaussian Kernel
```

```
Run Linear SVM
Construct Linear Kernel
```




```
Run Kernel SVM
Construct Gaussian Kernel
```



General Supervised Learning

- **Generalization:**

$$\min_{w} \ \|w\|_2^2 + \gamma \sum_{i=1}^{n} V_{loss}(f_i, \ell_i)$$

$$\downarrow$$

$$\min_{f \in H_K} \ \|f\|_{H_K}^2 + \gamma \sum_{i=1}^{n} V_{loss}(f_i, \ell_i)$$

Regularity of f — ***Error for inaccurate predictions*** — ***Trade-off***

$$\|f\|_{H_K}^2 = \|w\|_2^2 \quad \text{for} \quad f(x) = \langle w, x \rangle$$

Representer theorem:

$$f(x) = \sum_{i=1}^{n} a_i K(x, x_i)$$

Norm of in RKHS:

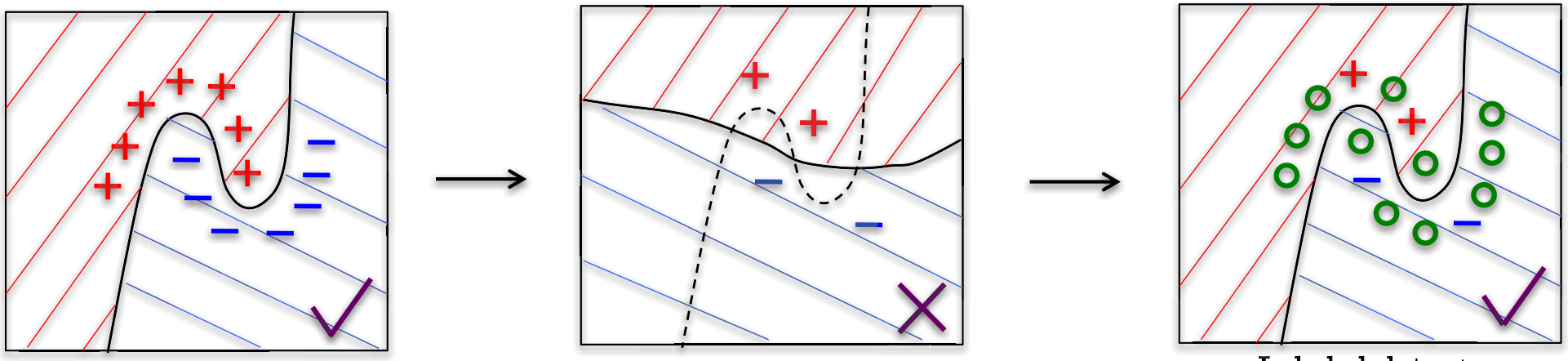
$$\|f\|_{H_K}^2 = \langle f, f \rangle_{H_K} = \sum_{ij} f_i f_j K_{ij} = f^T K f$$

Outline

- Learning techniques
- Linear SVM
- Soft-Margin SVM
- Kernel Techniques
- Non-Linear/Kernel SVM
- **Graph SVM**
- Conclusion

Semi-Supervised Learning – Graph SVM

- **Motivation:** It is *time consuming* to *label* data, but it is *cheap* to collect *unlabeled* data. Besides, unlabeled data are without information: they hold the *geometric structure* of data ⇒ **Best design of classification function is with simultaneously labeled and unlabeled data.**



Labeled data +,-
Supervised Learning

Labeled data +,-
Supervised Learning

Labeled data +,-
Unlabeled data o
Semi-Supervised Learning

Semi-Supervised Learning – Graph SVM

- **Case of few labels:** Semi-supervised learning is particularly wanted in the case of **few labels** (extreme case is *one label/class*). Let us call

  n: the number of labeled data

  m: the number of unlabeled data

  Then, *semi-supervised plays an important role in the case of* $n \ll m$.

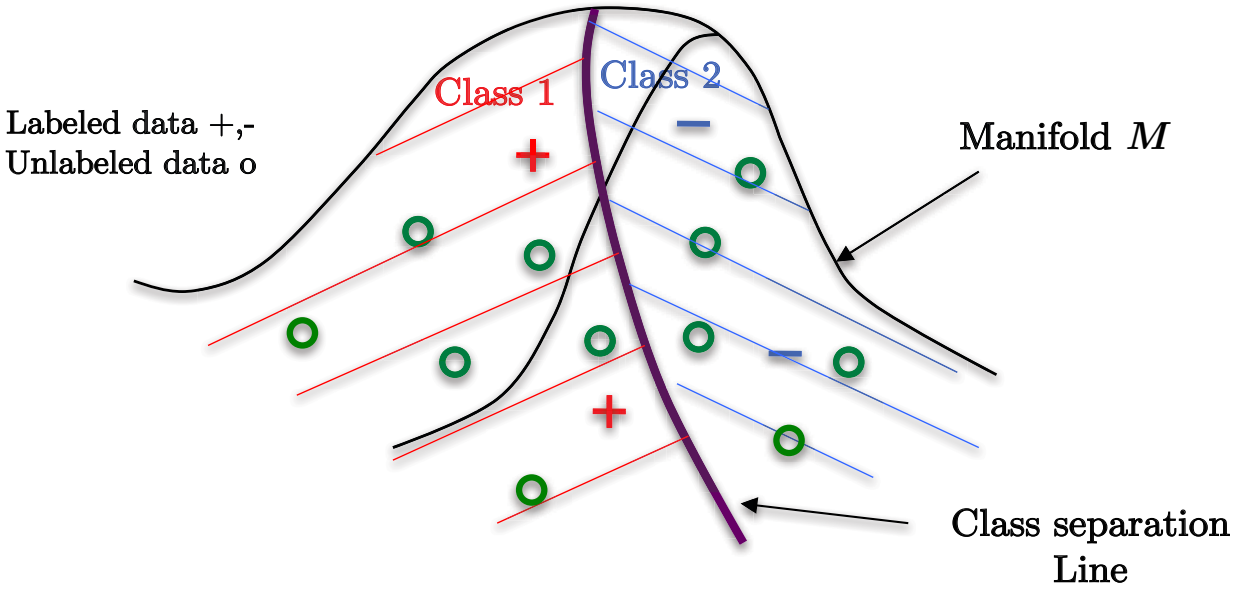
Q: How to design SVM to deal simultaneously with labeled and unlabeled data?

$\Rightarrow$ SVM on graphs

Manifold Assumption

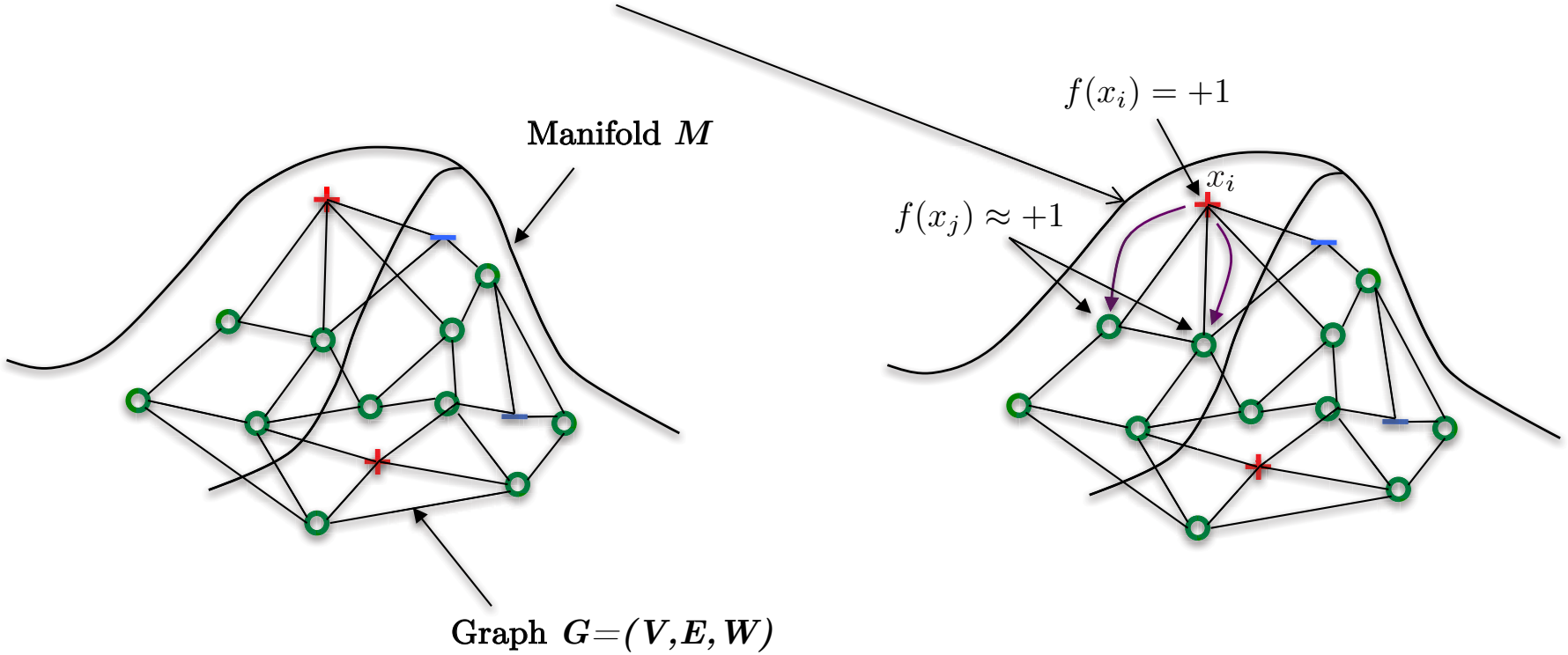
- **Observation: Geometry of data is independent of labels.**

  (Labeled and unlabeled) **data are assumed to lie on a manifold**, where the classification will be carried out.



Manifold Assumption

- **How to introduce the manifold geometry in SVM?**
  - First, we approximate the manifold M with a neighborhood graph, **i.e. a k-NN graph**.
  - Second, we add a **regularization term** that forces the classification **function f to be smooth on the manifold** (/graph).



Optimization

- **Optimization problem:**

$$\min_{f \in H_K} \|f\|_{H_K}^2 + \gamma_l \sum_{i=1}^{n} V_{loss}(f_i, \ell_i) + \gamma_G \underbrace{\int_{\mathcal{M}} |\nabla f|^2}$$

Dirichlet energy:
(1) It forces f to be smooth on M
(2) Derivative is $\Delta f=0$ (heat diffusion)

- **Discretization of Dirichlet on graphs:**

$$\int_{\mathcal{M}} |\nabla f|^2 \approx \sum_{ij} W_{ij} |f(x_i) - f(x_j)|^2 = f^T \mathcal{L} f$$

Laplacian operator

Algorithm

- **Semi-supervised SVM or Laplacian SVM** [Belkin, Niyogi, Sindhwani'06]:

$$\min_{f \in H_K} \; \|f\|_{H_K}^2 + \gamma_l \sum_{i=1}^{n} V_{hinge}(f_i, \ell_i) + \gamma_G f^T \mathcal{L} f$$

$$\Downarrow$$

$$f(x) = \text{sign}\Big(\sum_{i=1}^{n} a_i^\star K(x, x_i) \Big)$$

$$a^\star = (I + \gamma_G \mathcal{L} K)^{-1} H L \alpha^\star$$

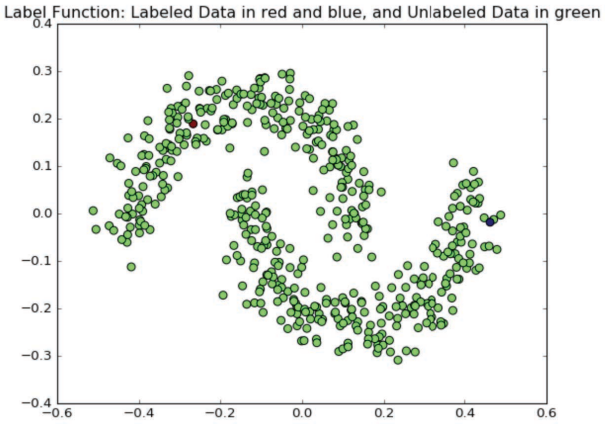
$$\alpha^\star = \arg \min_{0 \leq \alpha \leq \gamma_l} \; \frac{1}{2} \alpha^T Q \alpha - \langle \alpha, 1 \rangle \quad \text{s.t.} \quad \langle \alpha, 1 \rangle = 0$$

$$\text{with} \quad Q = LHK(I + \gamma_G \mathcal{L} K)^{-1} HL$$

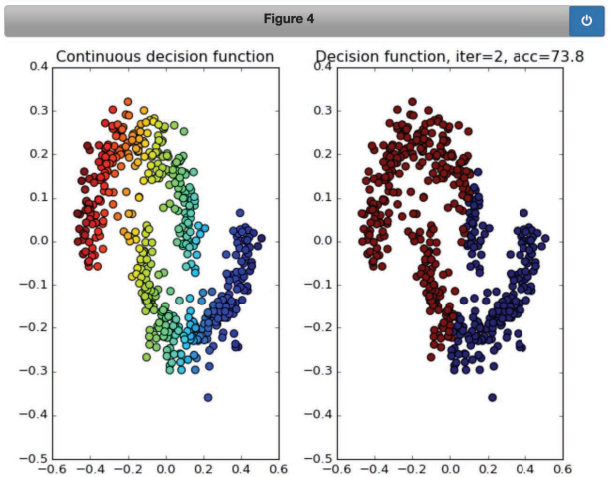
Demo: Graph SVM

- Run **code04.ipynb**

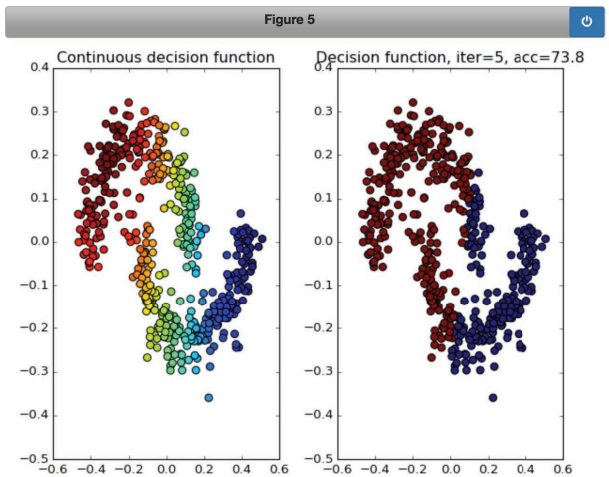
```
# Run Graph SVM
_,_,_,_ = compute_SVM(Xtrain,Cgt_train,l_train,'gaussian_kernel',[1.0,0.114],Xtest,Cgt_test,[6,50],
                      'euclidean',[10,24.0])
```



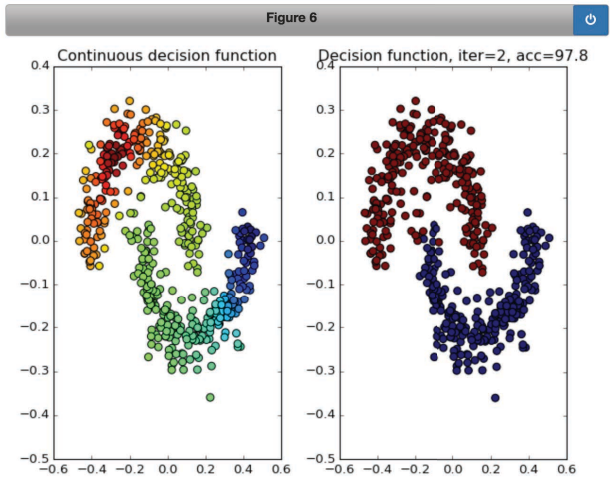
Run Linear SVM
Construct Linear Kernel



Run Kernel SVM
Construct Gaussian Kernel



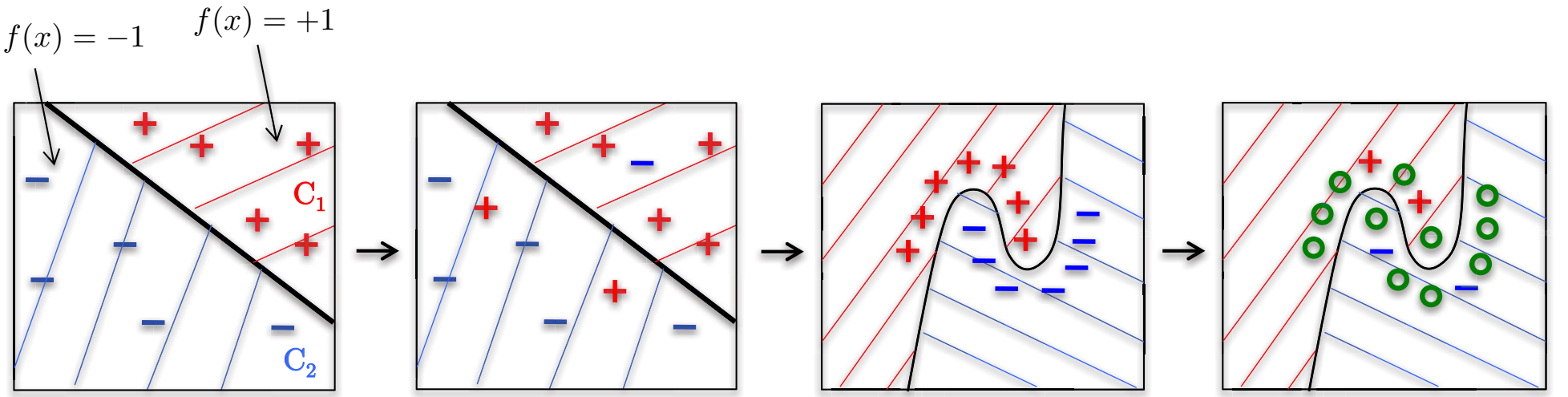
Run Graph SVM with Gaussian Kernel
Construct Gaussian Kernel
k-NN graph with euclidean distance



Outline

- Learning techniques
- Linear SVM
- Soft-Margin SVM
- Kernel Techniques
- Non-Linear/Kernel SVM
- Graph SVM
- **Conclusion**

Summary



Linear SVM
Supervised Learning
[Vapnik-Chervonenkis'63]

Soft-Margin SVM
Supervised Learning
[Cortes-Vapnik'95]

Non-Linear/Kernel SVM
Supervised Learning
[Boser,Guyon,Vapnik-Chervonenkis'92]

Laplacian SVM
Semi-Supervised Learning
[Belkin,Niyogi,Sindhwari'06]

Summary

- **General** supervised and semi-supervised optimization techniques:

$$\min_{f \in H_K} \|f\|_{H_K}^2 + \gamma_l \sum_{i=1}^{n} V_{loss}(f_i, \ell_i) + \gamma_G R_{graph}(f)$$

Regularity of f — ***Error for inaccurate predictions*** — ***Graph regularization for unlabeled data***

$$V_{loss} = \begin{cases} \text{Hinge} \\ L_2 \\ L_1 \\ \text{Huber} \\ \text{Logistic} \end{cases}$$

$$R_{graph}(\cdot) = \begin{cases} \text{Dirichlet: } \|\nabla_G \cdot\|_2^2 \\ \text{Total Variation: } \|\nabla_G \cdot\|_1 \\ \text{Wavelets: } \|D_{wavelets} \cdot\|_2^2 \end{cases}$$

Questions?