



Samueli
Computer Science



CS M146 Discussion: Week 7

Kernels, SVM

Junheng Hao
Friday, 02/19/2021

- Announcement
- Kernels
- SVM
- PyTorch Q&A (PS3)

- **5:00 pm PST, Feb 19 (Friday):** Weekly Quiz 7 released on Gradescope.
- **11:59 pm PST, Feb 21 (Sunday):** Weekly quiz 4 closed on Gradescope!
 - Start the quiz before **11:00 pm PST, Feb 21** to have the full 60-minute time
- **Problem set 3** released on CCLE, submission on Gradescope.
 - Please assign pages of your submission with corresponding problem set outline items on GradeScope.
 - You need to submit code, similar to PS2
 - Due on next week, **11:59pm PST, Feb 26 (Friday)**

Late Submission of PS will NOT be accepted!

About Quiz 7



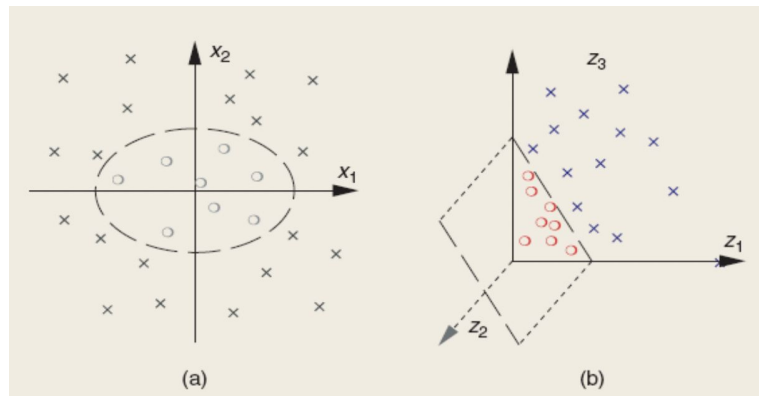
- Quiz release date and time: **Feb 19, 2021 (Friday) 05:00 PM PST**
- Quiz due/close date and time: **Feb 21, 2021 (Sunday) 11:59 PM PST**
- You will have up to **60 minutes** to take this exam. → Start before **11:00 PM** Sunday
- You can find the exam entry named "Week 7 Quiz" on GradeScope.
- Topics: **Kernels, SVM**
- Question Types
 - True/false, multiple choices
 - Some questions may include several subquestions.
- Some light calculations are expected. Some scratch paper and one scientific calculator (physical or online) are recommended for preparation.

- Motivation: Transformed feature space
- Basic idea: Define K , called kernel, such that:

$$K: X \times X \rightarrow \mathbb{R} \quad \Phi(x) \cdot \Phi(y) = K(x, y)$$

which is often as a similarity measure.

- Benefit:
 - Efficiency: is often more efficient to compute than and the dot product.
 - Flexibility: can be chosen arbitrarily so long as the existence of is guaranteed (Mercer's condition).

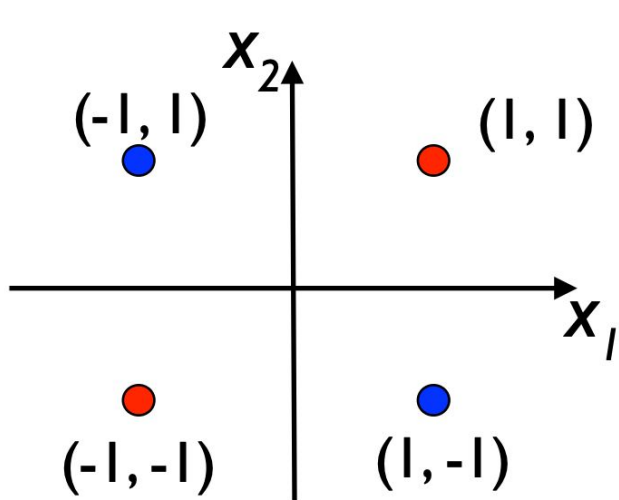


■ **Definition:**

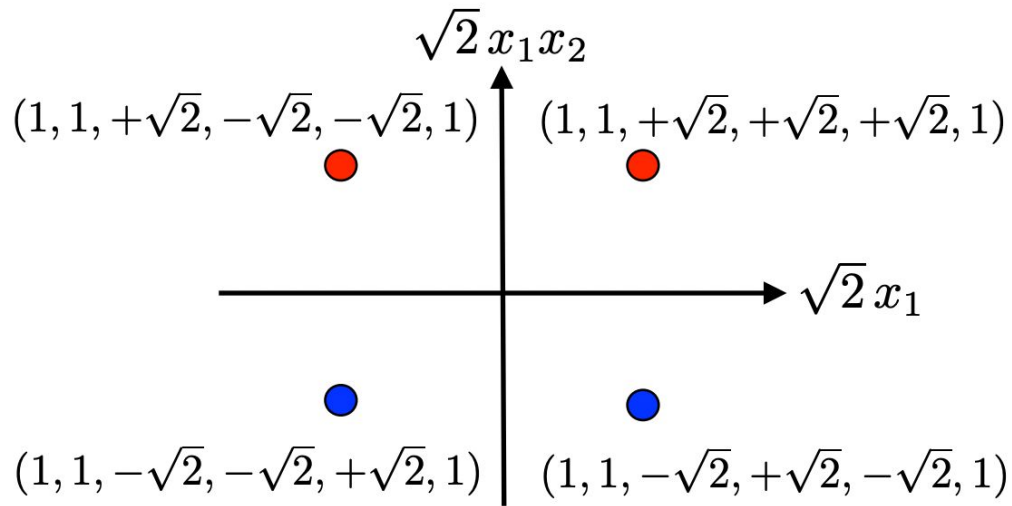
$$\forall x, y \in \mathbb{R}^N, \quad K(x, y) = (x \cdot y + c)^d, \quad c > 0.$$

■ **Example:** for $N=2$ and $d=2$,

$$\begin{aligned} K(x, y) &= (x_1 y_1 + x_2 y_2 + c)^2 \\ &= \begin{bmatrix} x_1^2 \\ x_2^2 \\ \sqrt{2} x_1 x_2 \\ \sqrt{2c} x_1 \\ \sqrt{2c} x_2 \\ c \end{bmatrix} \cdot \begin{bmatrix} y_1^2 \\ y_2^2 \\ \sqrt{2} y_1 y_2 \\ \sqrt{2c} y_1 \\ \sqrt{2c} y_2 \\ c \end{bmatrix}. \end{aligned}$$



Linearly non-separable



Linearly separable by
 $x_1x_2 = 0$.

Gaussian kernels:

$$K(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right), \sigma \neq 0.$$

Also known as “Radial Basis Function Kernel”

Sigmoid Kernels:

$$K(x, y) = \tanh(a(x \cdot y) + b), a, b \geq 0.$$

Note: The RBF/Gaussian kernel as a projection into infinite dimensions, commonly used in kernel SVM.

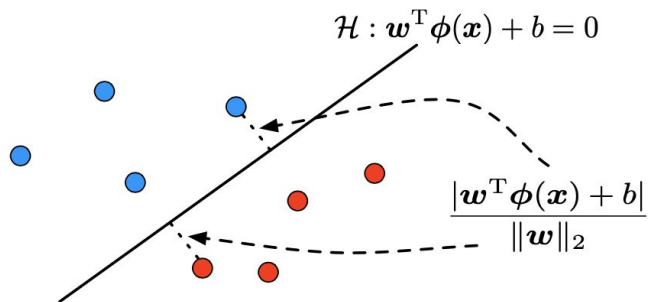
$$\begin{aligned} K(x, x') &= \exp\left(-(x - x')^2\right) \\ &= \exp(-x^2) \exp(-x'^2) \underbrace{\sum_{k=0}^{\infty} \frac{2^k (x)^k (x')^k}{k!}}_{\exp(2xx')} \end{aligned}$$

Taylor Expansion

- Links: <https://cs.stanford.edu/people/karpathy/svmjs/demo/>



$$\text{MARGIN}(\mathbf{w}, b) = \min_n \frac{y_n [\mathbf{w}^T \phi(\mathbf{x}_n) + b]}{\|\mathbf{w}\|_2}$$



SVM: Start with the margin



Margin Lines

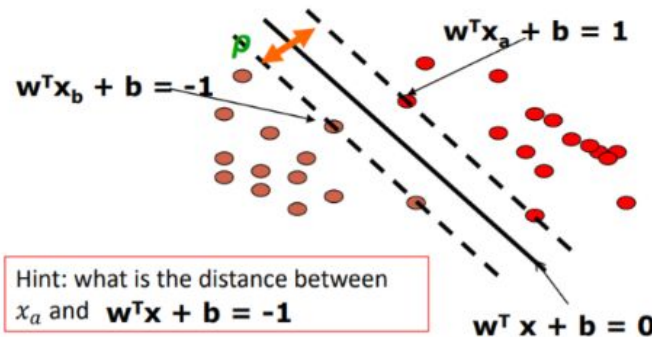
$$\mathbf{w}^T \mathbf{x}_a + b = 1 \quad \mathbf{w}^T \mathbf{x}_b + b = -1$$

Distance between parallel lines of $ax_1 + bx_2 = c_1/c_2$

$$d = \frac{|c_2 - c_1|}{\sqrt{a^2 + b^2}}$$

Margin

$$\rho = \frac{|(b+1) - (b-1)|}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$



1. Formulation of the Linear SVM problem: maximizing margin
2. Formulation of Quadratic Programming (optimization with linear constraints) → Primal problem
3. Solving linear SVM problem with “great” math*
 - a. (Generalized) Lagrange function, lagrange multiplier
 - b. Identify primal and dual problem (duality) → KKT conditions
 - c. Solution to w and b regarding α
4. Support Vectors, SVM Classifier Inference
5. Non-linear SVM, Kernel tricks

- Slides: <http://people.csail.mit.edu/dsontag/courses/ml13/slides/lecture6.pdf>
- Notes: <https://see.stanford.edu/materials/aimlcs229/cs229-notes3.pdf>

**To show in hand notes*

This slide is intentionally left blank.

Given labeled data and alpha values

- Positively labeled data points (1 to 4)

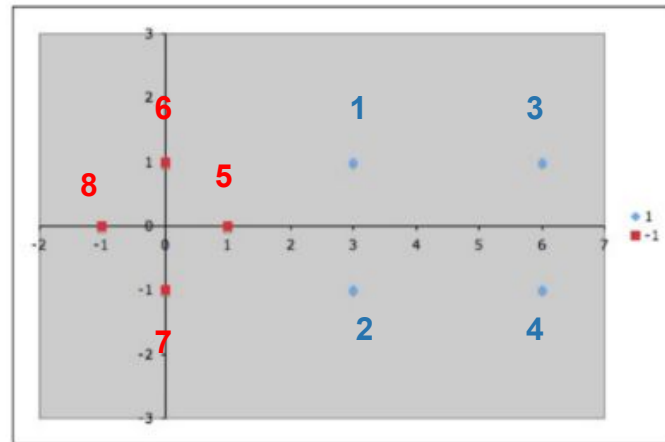
$$\left\{ \begin{pmatrix} 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ -1 \end{pmatrix}, \begin{pmatrix} 6 \\ 1 \end{pmatrix}, \begin{pmatrix} 6 \\ -1 \end{pmatrix} \right\}$$

- Negatively labeled data points (5 to 8)

$$\left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 0 \end{pmatrix} \right\}$$

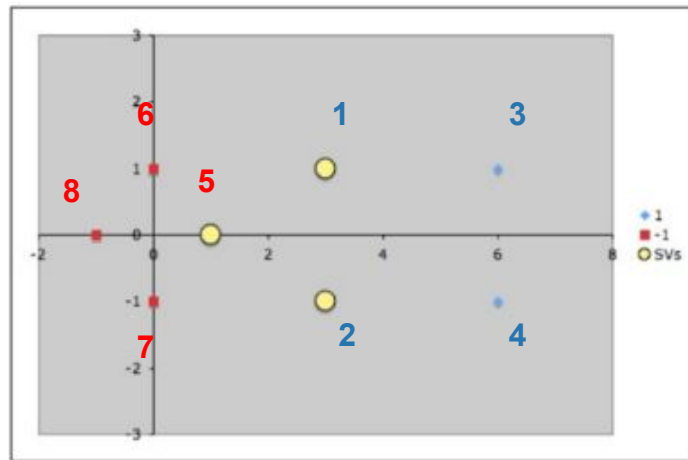
- Alpha values

- $\alpha_1 = 0.25$
- $\alpha_2 = 0.25$
- $\alpha_5 = 0.5$
- Others = 0



- Which points are support vectors?
- Calculate normal vector of hyperplane: \mathbf{w}
- Calculate the bias term
- What is the decision boundary?
- Predict class of new point (4, 1)

$$\mathbf{w} = \sum \alpha_i y_i \mathbf{x}_i \quad b = \sum_{k:\alpha_k \neq 0} (y_k - \mathbf{w}^T \mathbf{x}_k) / N_k$$



Linear SVM: Example for Practice

Predictions for new data



$$y \leftarrow \text{sign}(\vec{w} \cdot \vec{x} + b)$$



Using dual solution

$$y \leftarrow \text{sign} \left[\sum_i \alpha_i y_i (\underbrace{\vec{x}_i \cdot \vec{x}}_{\text{dot product}}) + b \right]$$

dot product of feature vectors of
new example with support vectors

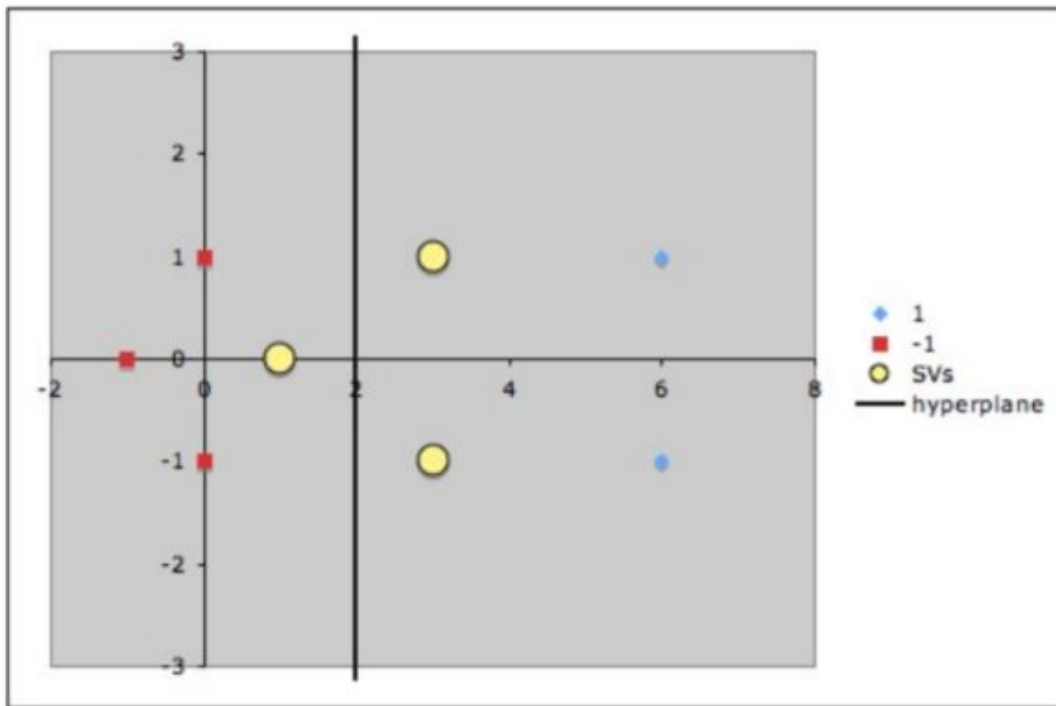
$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i$$

$$b = y_k - \mathbf{w} \cdot \mathbf{x}_k$$

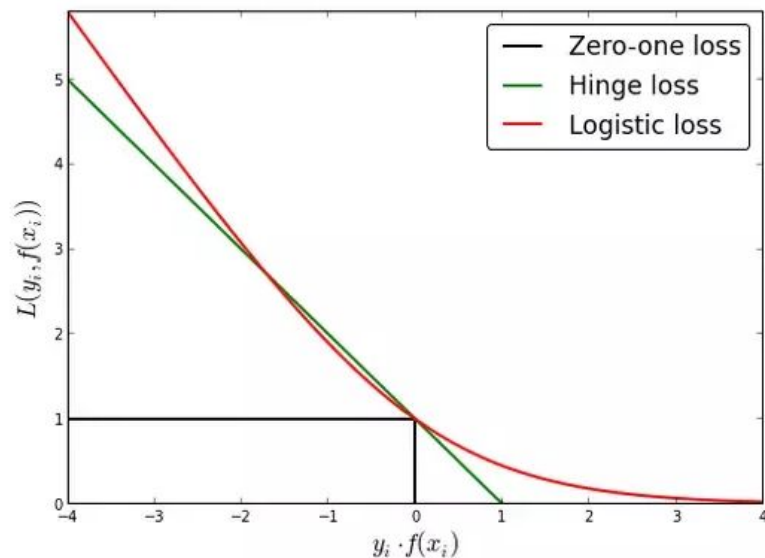
for any k where $C > \alpha_k > 0$

Linear SVM: Example for Practice

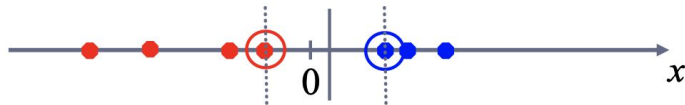
Plot



- Decision boundaries?
- Loss functions?



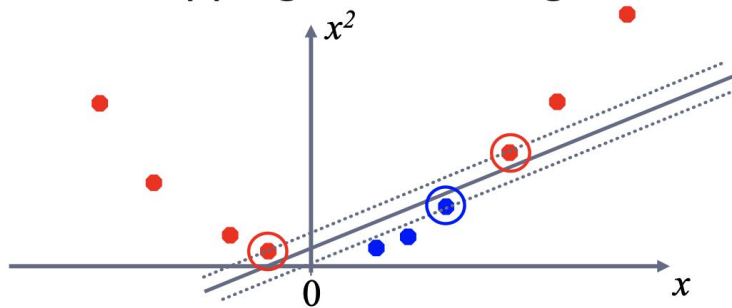
- Datasets that are linearly separable (with some noise) work out great:



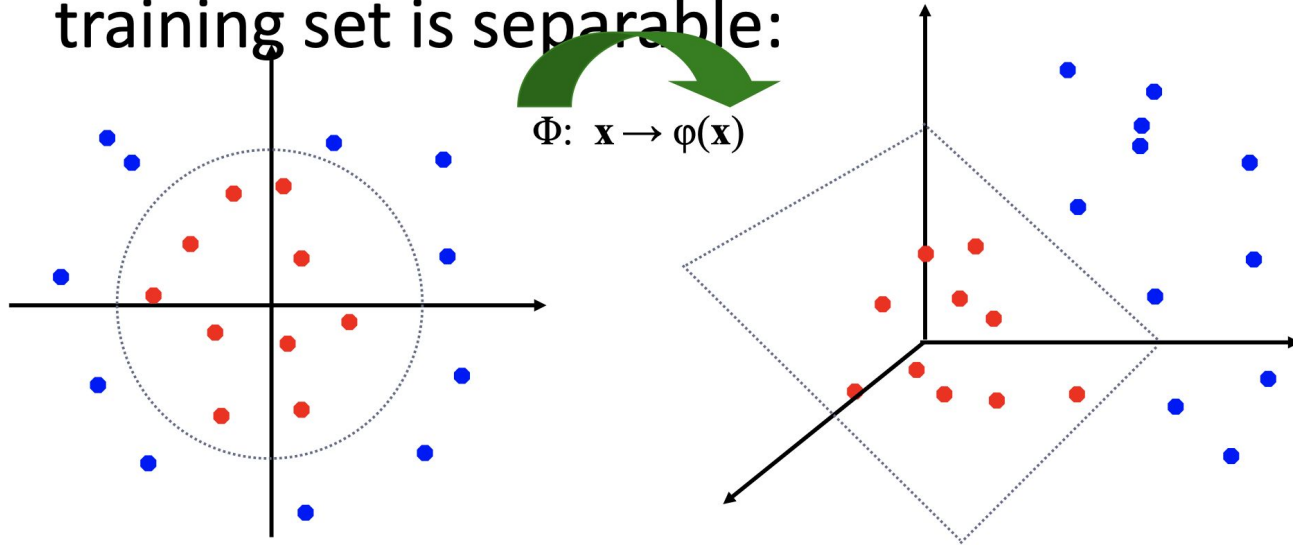
- But what are we going to do if the dataset is just too hard?



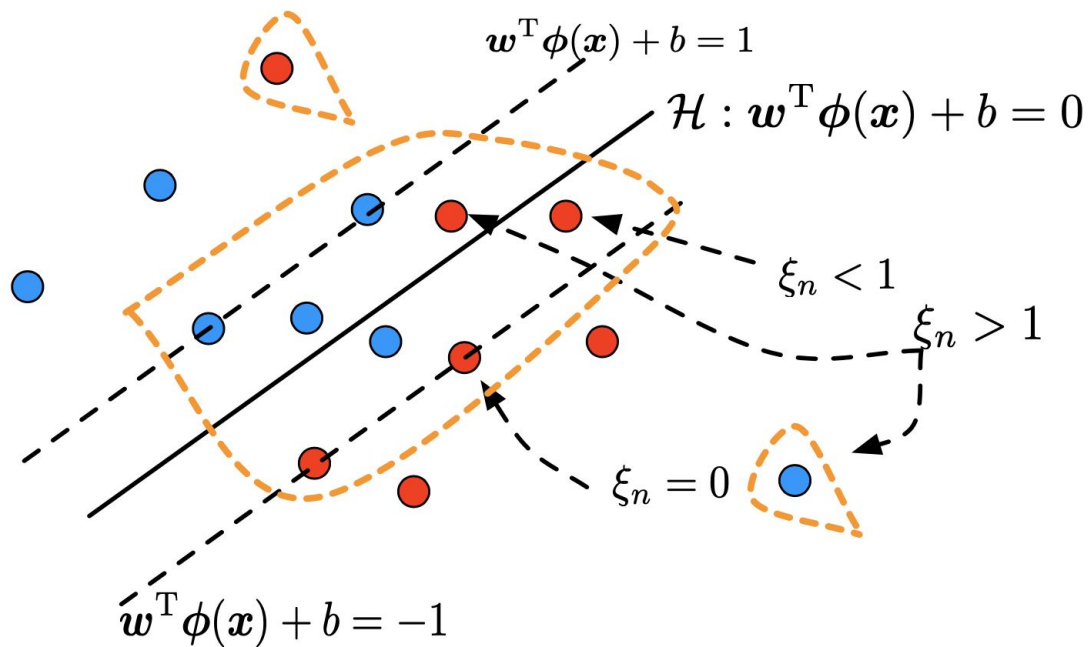
- How about ... mapping data to a higher-dimensional space:



- General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable:



Support Vectors



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

$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

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

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

$$\sum_i \alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0$$

- The linear SVM relies on an inner product between data vectors,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$$

- If every data point is mapped into high-dimensional space via transformation, the inner product becomes,

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi^T(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$$

- Do we need to compute $\phi(\mathbf{x})$ explicitly for each data sample? → **Directly compute kernel function $K(x_i, x_j)$**

$$\begin{aligned}
 k(\mathbf{x}, \mathbf{z}) &= (\mathbf{x}^T \mathbf{z} + c)^2 = \left(\sum_{j=1}^n x^{(j)} z^{(j)} + c \right) \left(\sum_{\ell=1}^n x^{(\ell)} z^{(\ell)} + c \right) \\
 &= \sum_{j=1}^n \sum_{\ell=1}^n x^{(j)} x^{(\ell)} z^{(j)} z^{(\ell)} + 2c \sum_{j=1}^n x^{(j)} z^{(j)} + c^2 \\
 &= \sum_{j,\ell=1}^n (x^{(j)} x^{(\ell)}) (z^{(j)} z^{(\ell)}) + \sum_{j=1}^n (\sqrt{2c} x^{(j)}) (\sqrt{2c} z^{(j)}) + c^2,
 \end{aligned}$$

Feature mapping given by:

$$\Phi(\mathbf{x}) = [x^{(1)2}, x^{(1)}x^{(2)}, \dots, x^{(3)2}, \sqrt{2c}x^{(1)}, \sqrt{2c}x^{(2)}, \sqrt{2c}x^{(3)}, c]$$

Polynomial kernel of degree h : $K(\mathbf{X}_i, \mathbf{X}_j) = (\mathbf{X}_i \cdot \mathbf{X}_j + 1)^h$

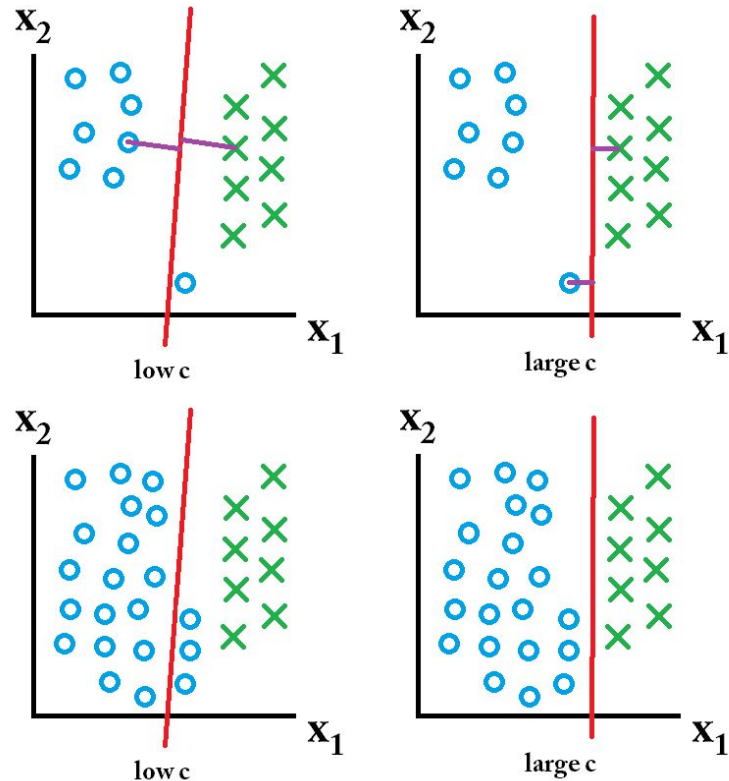
Gaussian radial basis function kernel : $K(\mathbf{X}_i, \mathbf{X}_j) = e^{-\|\mathbf{X}_i - \mathbf{X}_j\|^2 / 2\sigma^2}$

Sigmoid kernel : $K(\mathbf{X}_i, \mathbf{X}_j) = \tanh(\kappa \mathbf{X}_i \cdot \mathbf{X}_j - \delta)$

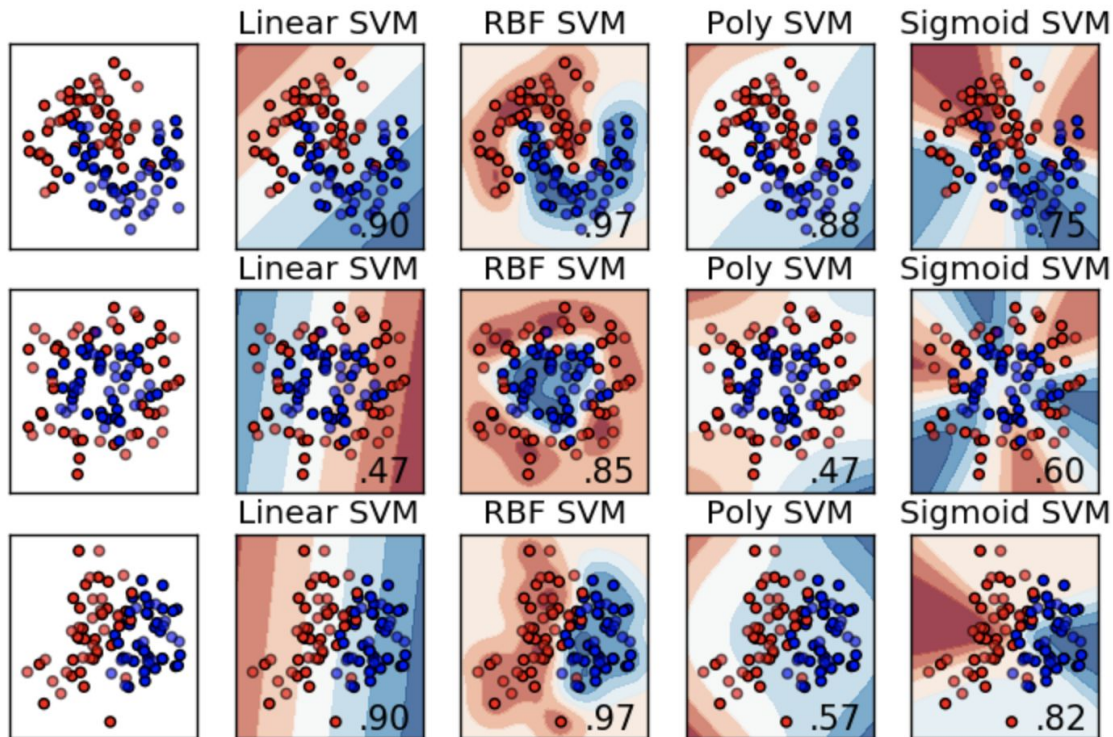
- Given the same data samples, what is the difference between linear kernel and non-linear kernel? Is the decision boundary linear (in original feature space)?

- Huge feature space with kernels: should we worry about overfitting?
 - SVM objective seeks a solution with large margin.
 - Theory says that large margin leads to good generalization.
 - But everything overfits sometimes.
 - Can control by:
 - Setting C
 - Choosing a better Kernel
 - Varying parameters of the Kernel (width of Gaussian, etc.)

- The C parameter tells the SVM optimization how much you want to avoid misclassifying each training example.
- For large values of C, the optimization will choose **a smaller-margin hyperplane** if that hyperplane does a better job of getting all the training points classified correctly.
- Conversely, a very small value of C will cause the optimizer to look for **a larger-margin hyperplane**, even if that hyperplane misclassified more points.



SVM: Demo of different kernels



- Positively labeled data points (1 to 4)

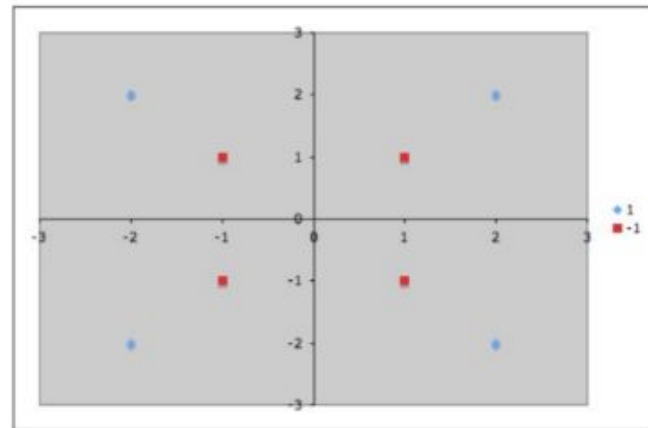
$$\left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ -2 \end{pmatrix}, \begin{pmatrix} -2 \\ 2 \end{pmatrix} \right\}$$

- Negatively labeled data points (5 to 8)

$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

- Non-linear mapping

$$\Phi_1 \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{cases} \begin{pmatrix} 4 - x_2 \\ 4 - x_1 \\ x_1 \\ x_2 \end{pmatrix} & \text{if } \sqrt{x_1^2 + x_2^2} > 2 \\ \text{otherwise} \end{cases}$$



- New positively labeled data points (1 to 4)

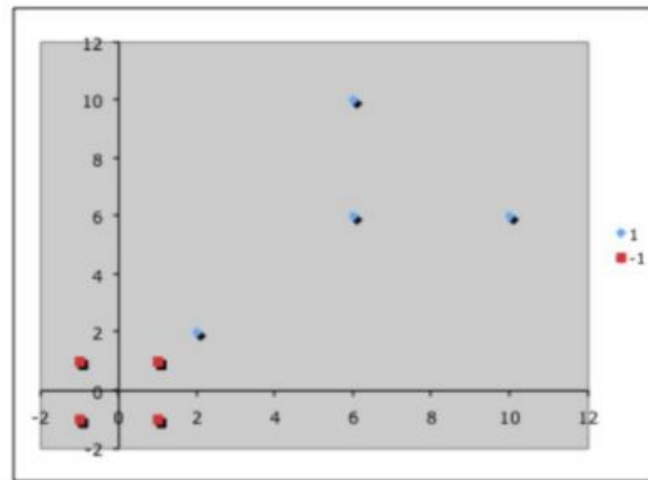
$$\left\{ \begin{pmatrix} 2 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 2 \end{pmatrix}, \begin{pmatrix} 6 \\ 6 \end{pmatrix}, \begin{pmatrix} 2 \\ 6 \end{pmatrix} \right\}$$

- New negatively labeled data points (5 to 8)

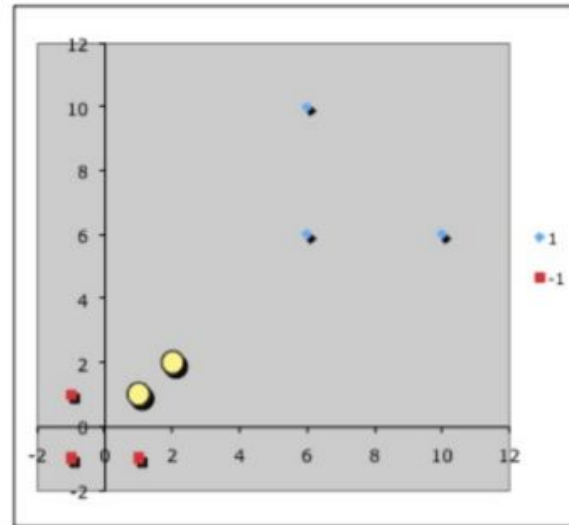
$$\left\{ \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ -1 \end{pmatrix}, \begin{pmatrix} -1 \\ 1 \end{pmatrix} \right\}$$

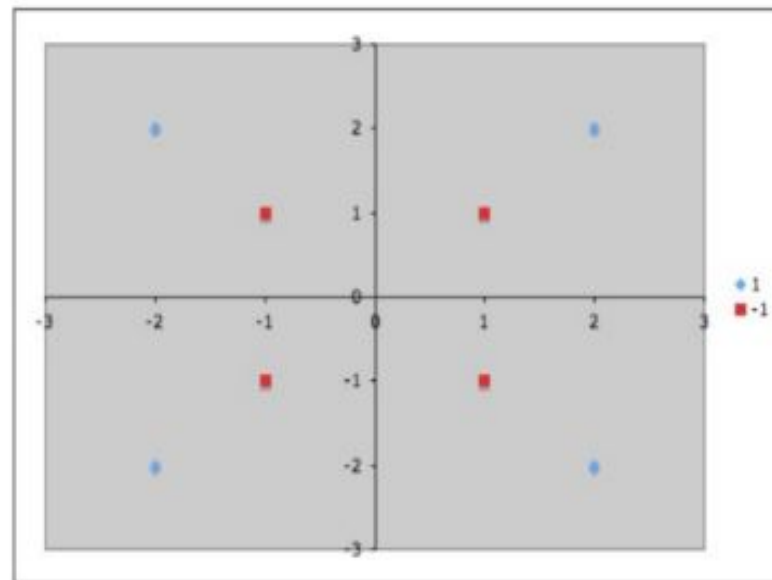
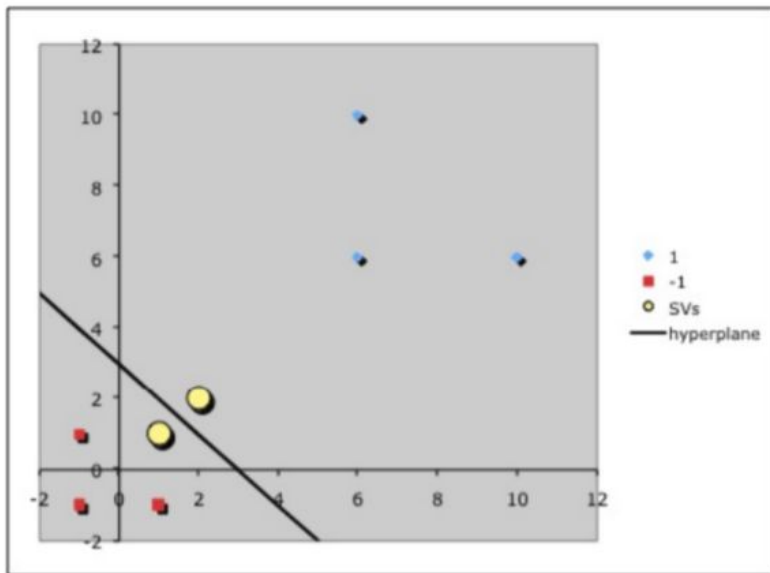
- Alpha values

- $\alpha_1 = 1.0$
- $\alpha_5 = 1.0$
- Others = 0



- Which points are support vectors?
- Calculate normal vector of hyperplane: \mathbf{w}
- Calculate the bias term
- What is the decision boundary?
- Predict class of new point (4, 5)





- Decision Boundary

$$y \leftarrow \text{sign} \left[\sum_i \alpha_i y_i K(x_i, x) + b \right]$$



Samueli
Computer Science



Thank you!

Q & A

-
- Content