

# Introduction to Machine Learning

Kernel Support Vector Machines

Varun Chandola

March 10, 2021

## Outline

## Contents

1 Support Vector Machines	1
1.1 SVM Learning	1
1.2 Kernel SVM	2

## 1 Support Vector Machines

### 1.1 SVM Learning

#### SVM Optimization

##### Optimization Formulation

$$\underset{\mathbf{w}, b}{\text{minimize}} \quad \frac{\|\mathbf{w}\|^2}{2}$$

$$\text{subject to} \quad y_n(\mathbf{w}^\top \mathbf{x}_n + b) \geq 1, \quad n = 1, \dots, N.$$

- Introducing [Lagrange Multipliers](#),  $\alpha_n, n = 1, \dots, N$

##### Rewriting as a (primal) Lagrangian

$$\underset{\mathbf{w}, b, \alpha}{\text{minimize}} \quad L_P(\mathbf{w}, b, \alpha) = \frac{\|\mathbf{w}\|^2}{2} + \sum_{n=1}^N \alpha_n \{1 - y_n(\mathbf{w}^\top \mathbf{x}_n + b)\}$$

$$\text{subject to} \quad \alpha_n \geq 0 \quad n = 1, \dots, N.$$

## Solving the Lagrangian

- Set gradient of  $L_P$  to 0

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{n=1}^N \alpha_n y_n \mathbf{x}_n$$

$$\frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{n=1}^N \alpha_n y_n = 0$$

- Substituting in  $L_P$  to get the dual  $L_D$

## Dual Lagrangian Formulation

$$\underset{\mathbf{w}, b, \alpha}{\text{maximize}} \quad L_D(\mathbf{w}, b, \alpha) = \sum_{n=1}^N \alpha_n - \frac{1}{2} \sum_{m, n=1}^N \alpha_m \alpha_n y_m y_n (\mathbf{x}_m^\top \mathbf{x}_n)$$

$$\text{subject to} \quad \sum_{n=1}^N \alpha_n y_n = 0, \alpha_n \geq 0 \quad n = 1, \dots, N.$$

## 1.2 Kernel SVM

### Dot Product Formulation

- All training examples ( $\mathbf{x}_n$ 's) occur in *dot/inner products*
- Also recall the prediction using SVMs

$$\begin{aligned} y^* &= \text{sign}(\mathbf{w}^\top \mathbf{x}^* + b) \\ &= \text{sign}\left(\left(\sum_{n=1}^N \alpha_n y_n \mathbf{x}_n\right)^\top \mathbf{x}^* + b\right) \\ &= \text{sign}\left(\sum_{n=1}^N \alpha_n y_n (\mathbf{x}_n^\top \mathbf{x}^*) + b\right) \end{aligned}$$

- Replace the dot products with kernel functions

– Kernel or non-linear SVM

- Kernel SVM with radial basis function kernel (RBF)

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{1}{2\gamma^2}\|\mathbf{x}_i - \mathbf{x}_j\|^2\right)$$

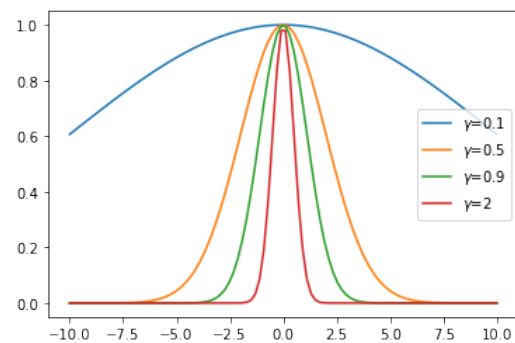
### Setting $\gamma$ and $C$

- $C$  is the regularization parameter

$$L(\mathbf{w}, b) = \|\mathbf{w}\|^2 + C \sum_{i=1}^N \xi_i \quad \text{subject to constraints}$$

- For the role of  $\gamma$ , consider the following two aspects:

$$y^* = \text{sign}\left(\sum_{n=1}^N \alpha_n y_n (\mathbf{x}_n^\top \mathbf{x}^*) + b\right)$$



- $\gamma$  determines the influence of a training example - inverse of the radius of influence of support vectors
- $C$  determines the trade-off between the total slack (errors on training data) and the size of the margin (regularization)

- Setting  $\gamma$  too large makes the decision boundary too complex,  $C$  will not prevent overfitting
- Setting  $\gamma$  very small makes the decision boundary simple (linear)
- Usually a grid search is performed to identify optimal  $C$  and  $\gamma$

### References