Constrained Optimization and Support Vector Machines

Man-Wai MAK

Dept. of Electronic and Information Engineering,
The Hong Kong Polytechnic University

enmwmak@polyu.edu.hk
http://www.eie.polyu.edu.hk/∼mwmak

References:

C. Bishop, "*Pattern Recognition and Machine Learning*", Appendix E, Springer, 2006.
S.Y. Kung, M.W. Mak and S.H. Lin, *Biometric Authentication: A Machine Learning Approach*, Prentice Hall, 2005, Chapter 4.
Lagrange multipliers and constrained optimization, www.khancademy.org
M. J. Kochenderfer and T. A. Wheeler, "*Algorithms for optimization*", MIT Press, 2019.

October 10, 2019

Overview

1. Motivations
   - Why Study Constrained Optimization
   - Why Study SVM
2. Constrained Optimization
   - Hard-Constrained Optimization
   - Lagrangian Function
   - Inequality Constraint
   - Multiple Constraints
   - Software Tools
3. Support Vector Machines
   - Linear SVM: Separable Case
   - Linear SVM: Fuzzy Separation (Optional)
   - Nonlinear SVM
   - SVM for Pattern Classification
   - Software Tools

Why Study Constrained Optimization?

- Constrained optimization is used in almost every discipline:
  - **Power Electronics**: "Design of a boost power factor correction converter using optimization techniques," *IEEE Transactions on Power Electronics*, vol. 19, no. 6, pp. 1388-1396, Nov. 2004.
  - **Wireless Communication**: "Energy-constrained modulation optimization," *IEEE Transactions on Wireless Communications*, vol. 4, no. 5, pp. 2349-2360, Sept. 2005
  - **Photonics**: "Module Placement Based on Resistive Network Optimization," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 3, no. 3, pp. 218-225, July 1984.
  - **Multimedia**: "Nonlinear total variation based noise removal algorithms." *Physica D: Nonlinear Phenomena* 60.1-4 (1992): 259-268.

Why Study SVM?

- SVM is a typical application of constraint optimization.
- SVMs are used everywhere:
  - **Power Electronics**: "Support Vector Machines Used to Estimate the Battery State of Charge," *IEEE Transactions on Power Electronics*, vol. 28, no. 12, pp. 5919-5926, Dec. 2013.
  - **Wireless Communication**: "Localization In Wireless Sensor Networks Based on Support Vector Machines," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 7, pp. 981-994, July 2008.
  - **Photonics**: "Development of robust calibration models using support vector machines for spectroscopic monitoring of blood glucose." *Analytical chemistry* 82.23 (2010): 9719-9726.
  - **Multimedia**: "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308-311, May 2006.
  - **Bioinformatics**: "Gene selection for cancer classification using support vector machines." *Machine Learning*, 46.1-3 (2002): 389-422.

Constrained Optimization

- Constrained optimization is the process of optimizing an objective function with respect to some variables in the presence of constraints on those variables.
- The objective function is either
  - a cost function or energy function which is to be minimized, or
  - a reward function or utility function, which is to be maximized.
- Constraints can be either
  - hard constraints which set conditions for the variables that are **required** to be satisfied, or
  - soft constraints which have some variable values that are **penalized** in the objective function if the conditions on the variables are not satisfied.

Constrained Optimization

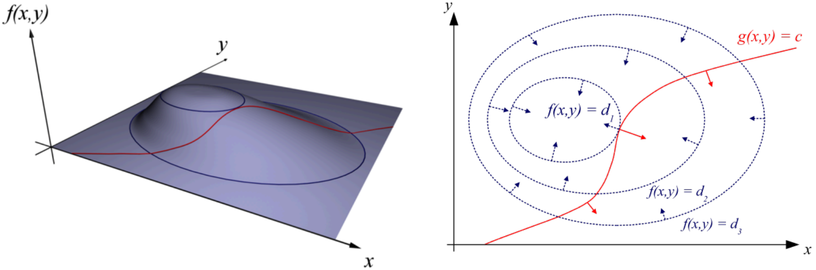
- A general constrained minimization problem:

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{subject to} & g_i(\mathbf{x}) = c_i \text{ for } i = 1, \ldots, n \text{ (Equality constraints)} \\ & h_j(\mathbf{x}) \geq d_j \text{ for } j = 1, \ldots, m \text{ (Inquality constraints)} \end{array} \tag{1}$$

  where $g_i(\mathbf{x}) = c_i$ and $h_j(\mathbf{x}) \geq d_j$ are called *hard constraints*.
- If the constrained problem has only equality constraints, the method of *Lagrange multipliers* can be used to convert it into an unconstrained problem whose number of variables is the original number of variables plus the original number of equality constraints.

- **Example**: Maximization of a function of two variables with equality constraints:

$$\begin{array}{ll} \max & f(x,y) \\ \text{subject to} & g(x,y) = 0 \end{array} \tag{2}$$

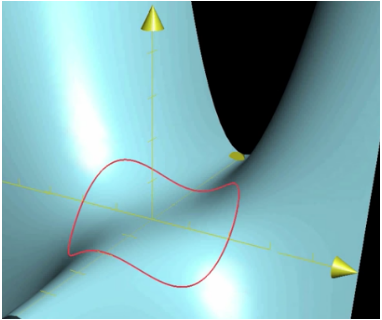


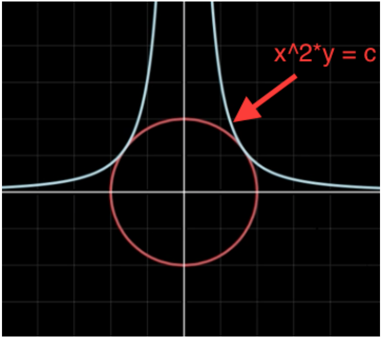
- At the optimal point (x^*, y^*), the gradient of $f(x,y)$ and $g(x,y)$ are anti-parallel, i.e., $\nabla f(x^*, y^*) = -\lambda \nabla g(x^*, y^*)$, where λ is called the Lagrange multiplier. (See Tutorial for explanation.)

Constrained Optimization

- Example:

$$\begin{array}{rl} \max & f(x,y) = x^2 y \\ \text{subject to} & x^2 + y^2 = 1 \end{array}$$

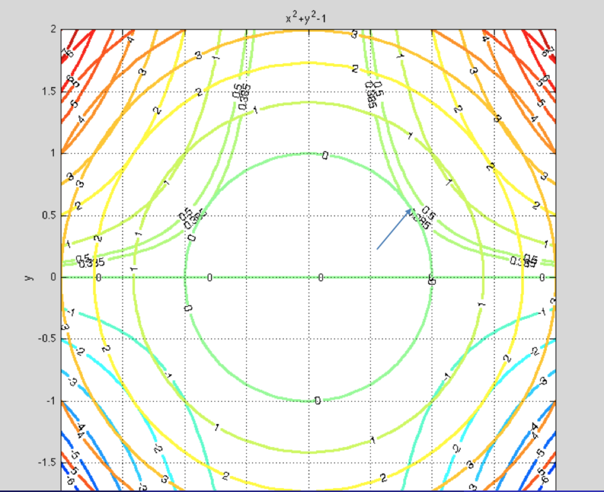




- Note that the red curve ($x^2 + y^2 = 1$) is of 2-dimension.

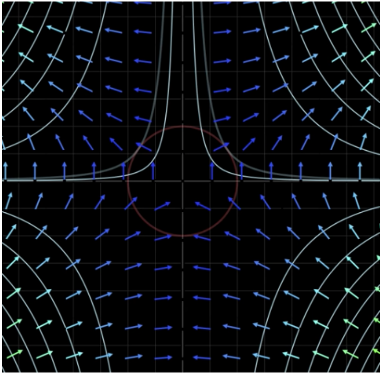
Constrained Optimization

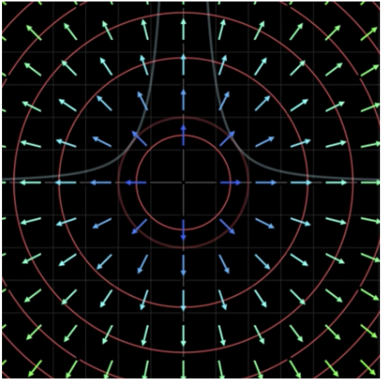
- $f(x, y) = x^2 y$ and $x^2 + y^2 = 1$
- Solution: $x^* = \sqrt{\frac{2}{3}}$; $y^* = \sqrt{\frac{1}{3}}$



Constrained Optimization

- *Left*: Gradients of the objective function $f(x,y) = x^2 y$
- *Right*: Gradients of $g(x,y) = x^2 + y^2$.





- Note that $\lambda < 0$ in this example, which means that the gradients of $f(x,y)$ and $g(x,y)$ are parallel at the optimal point.

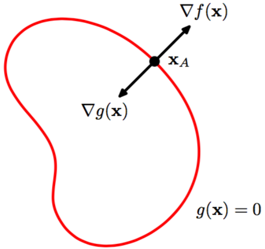
Constrained Optimization

- Extension to function of D variables:

$$\begin{array}{rl} \max & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) = 0 \end{array} \tag{3}$$

where $\mathbf{x} \in \Re^D$. Optimal occurs when

$$\nabla f(\mathbf{x}) + \lambda \nabla g(\mathbf{x}) = 0. \tag{4}$$



- Note that the red curve is of dimension $D - 1$.

- Define the Lagrangian function as

$$L(\mathbf{x}, \lambda) \equiv f(\mathbf{x}) + \lambda g(\mathbf{x}) \tag{5}$$

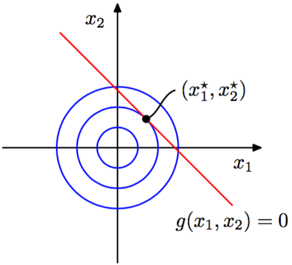
where $\lambda \neq 0$ is the Lagrange multiplier.

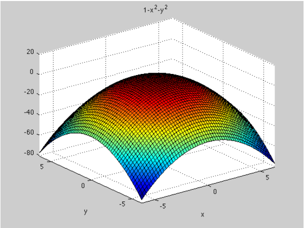
- The optimal condition (Eq. 4) will be satisfied when $\nabla_{\mathbf{x}} L = 0$.
- Note that $\partial L / \partial \lambda = 0$ leads to the constrained equation $g(\mathbf{x}) = 0$.
- The constrained maximization can be written as:

$$\begin{array}{rl} \max & L(\mathbf{x}, \lambda) = f(\mathbf{x}) + \lambda g(\mathbf{x}) \\ \text{subject to} & \lambda \neq 0, g(\mathbf{x}) = 0 \end{array} \tag{6}$$

- Find the stationary point of the function $f(x_1, x_2)$:

$$\begin{array}{ll} \max & f(x_1, x_2) = 1 - x_1^2 - x_2^2 \\ \text{subject to} & g(x_1, x_2) = x_1 + x_2 - 1 = 0 \end{array} \tag{7}$$





- Lagrangian function:

$$L(\mathbf{x}, \lambda) = 1 - x_1^2 - x_2^2 + \lambda(x_1 + x_2 - 1)$$

Lagrangian Function: 2D Example

- Differenting $L(\mathbf{x}, \lambda)$ w.r.t. x_1, x_2, and λ and set the results to 0, we obtain

$$
\begin{aligned}
-2x_1 + \lambda &= 0 \\
-2x_2 + \lambda &= 0 \\
x_1 + x_2 - 1 &= 0
\end{aligned}
$$

- The solution is $(x_1^*, x_2^*) = (\frac{1}{2}, \frac{1}{2})$, and the corresponding $\lambda = 1$.
- As $\lambda > 0$, the gradients of $f(x_1, x_2)$ and $g(x_1, x_2)$ are anti-parallel at (x_1^*, x_2^*).

Inequality Constraint

- Maximization with *inequality* constraint

$$\begin{array}{ll} \max & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \geq 0 \end{array} \tag{8}$$

- Two possible solutions for the max of $L(\mathbf{x}, \mu) = f(\mathbf{x}) + \mu g(\mathbf{x})$:

$$\begin{array}{llll} \textit{Inactive}\text{ Constraint}: & g(\mathbf{x}) > 0, & \mu = 0, & \nabla f(\mathbf{x}) = 0 \\ \textit{Active}\text{ Constraint}: & g(\mathbf{x}) = 0, & \mu > 0, & \nabla f(\mathbf{x}) = -\mu \nabla g(\mathbf{x}) \end{array} \tag{9}$$

- Therefore, the maximization can be rewritten as

$$\begin{array}{ll} \max & L(\mathbf{x}, \mu) = f(\mathbf{x}) + \mu g(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \geq 0, \mu \geq 0, \mu g(\mathbf{x}) = 0 \end{array} \tag{10}$$

which is known as the Karush-Kuhn-Tucker (KKT) condition.

Inequality Constraint

- For minimization,

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \geq 0 \end{array} \tag{11}$$

- We can also express the minimization as

$$\begin{array}{ll} \min & L(\mathbf{x}, \mu) = f(\mathbf{x}) - \mu g(\mathbf{x}) \\ \text{subject to} & g(\mathbf{x}) \geq 0, \mu \geq 0, \mu g(\mathbf{x}) = 0 \end{array} \tag{12}$$

Multiple Constraints

- Maximization with multiple equality and inequality constraints:

$$\begin{aligned} \max \quad & f(\mathbf{x}) \\ \text{subject to} \quad & g_j(\mathbf{x}) = 0 \text{ for } j = 1, \ldots, J \\ & h_k(\mathbf{x}) \geq 0 \text{ for } k = 1, \ldots, K. \end{aligned} \tag{13}$$

- This maximization can be written as

$$\begin{aligned} \max \quad & L(\mathbf{x}, \{\lambda_j\}, \{\mu_k\}) = f(\mathbf{x}) + \sum_{j=1}^{J} \lambda_j g_j(\mathbf{x}) + \sum_{k=1}^{K} \mu_k h_k(\mathbf{x}) \\ \text{subject to} \quad & \lambda_j \neq 0, g_j(\mathbf{x}) = 0 \text{ for } j = 1, \ldots, J \text{ and} \\ & \mu_k \geq 0, h_k(\mathbf{x}) \geq 0, \mu_k h_k(\mathbf{x}) = 0 \text{ for } k = 1, \ldots, K. \end{aligned} \tag{14}$$

Software Tools for Constrained Optimization

- **Matlab Optimization Toolbox:** `fmincon` can find the minimum of a function subject to nonlinear multivariable constraints.
- **Python:** `scipy.optimize.minimize` provides a common interface to unconstrained and constrained minimization algorithms for multivariate scalar functions

Linear SVM: Separable Case

- Consider a training set $\{\mathbf{x}_i, y_i;\ i = 1, \ldots, N\} \in \mathcal{X} \times \{+1, -1\}$ shown below, where $\mathcal{X}$ is the set of input data in $\Re^D$ and y_i are the labels.

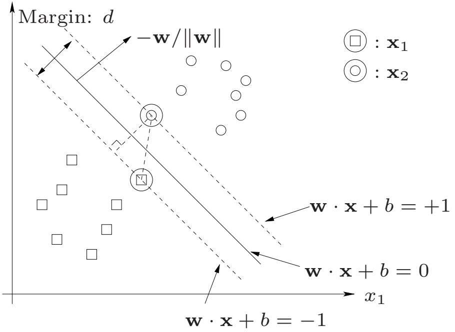


Figure: Linear SVM on 2-D space

- $\square$: $y_i = +1$; $\circ$: $y_i = -1$.

Linear SVM: Separable Case

- A linear support vector machine (SVM) aims to find a decision plane (a line for the case of 2D)

$$\mathbf{x} \cdot \mathbf{w} + b = 0$$

that maximizes the margin of separation (see Fig. 1).

- Assume that all data points satisfy the constraints:

$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for} \quad i \in \{1, \ldots, N\} \text{ where } y_i = +1. \tag{15}$$
$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for} \quad i \in \{1, \ldots, N\} \text{ where } y_i = -1. \tag{16}$$

- Data points $\mathbf{x}_1$ and $\mathbf{x}_2$ in previous page satisfy the equality constraint:

$$\begin{aligned} \mathbf{x}_1 \cdot \mathbf{w} + b &= +1 \\ \mathbf{x}_2 \cdot \mathbf{w} + b &= -1 \end{aligned} \tag{17}$$

Linear SVM: Separable Case

- Using Eq. 17 and Fig. 1, the distance between the two separating hyperplane (also called the margin of separation) can be computed:

$$d(\mathbf{w}) = (\mathbf{x}_2 - \mathbf{x}_1) \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|} = \frac{2}{\|\mathbf{w}\|}$$

- Maximizing $d(\mathbf{w})$ is equivalent to minimizing $\|\mathbf{w}\|^2$. So, the constrained optimization problem in SVM is

$$\begin{array}{cl} \min & \frac{1}{2}\|\mathbf{w}\|^2 \\ \text{subject to} & y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 \; \forall i = 1, \ldots, N \end{array} \tag{18}$$

- Equivalently, minimizing a Lagrangian function:

$$\begin{array}{cl} \min & L(\mathbf{w}, b, \{\alpha_i\}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{N} \alpha_i [y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1] \\ \text{subject to} & \alpha_i \geq 0, \quad y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 \geq 0, \\ & \alpha_i [y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1] = 0, \quad \forall i = 1, \ldots, N \end{array} \tag{19}$$

Linear SVM: Separable Case

- Setting

$$\frac{\partial}{\partial b}L(\mathbf{w}, b, \{\alpha_i\}) = 0 \quad \text{and} \quad \frac{\partial}{\partial \mathbf{w}}L(\mathbf{w}, b, \{\alpha_i\}) = 0, \tag{20}$$

subject to the constraint $\alpha_i \geq 0$, results in

$$\sum\nolimits_{i=1}^{N} \alpha_i y_i = 0 \quad \text{and} \quad \mathbf{w} = \sum\nolimits_{i=1}^{N} \alpha_i y_i \mathbf{x}_i. \tag{21}$$

- Substituting these results back into the Lagrangian function:

$$L(\mathbf{w}, b, \{\alpha_i\}) = \frac{1}{2}(\mathbf{w} \cdot \mathbf{w}) - \sum_{i=1}^{N} \alpha_i y_i (\mathbf{x}_i \cdot \mathbf{w}) - \sum_{i=1}^{N} \alpha_i y_i b + \sum_{i=1}^{N} \alpha_i$$

$$= \frac{1}{2}\sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \cdot \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j - \sum_{i=1}^{N} \alpha_i y_i \mathbf{x}_i \cdot \sum_{j=1}^{N} \alpha_j y_j \mathbf{x}_j + \sum_{i=1}^{N} \alpha_i$$

$$= \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j).$$

Linear SVM: Separable Case

- This results in the following *Wolfe dual* formulation:

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j)$$

$$\text{subject to} \tag{22}$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0 \quad \text{and} \quad \alpha_i \geq 0, i = 1, \ldots, N.$$

- The solution contains two kinds of Lagrange multiplier:
  1. $\alpha_i = 0$: The corresponding $\mathbf{x}_i$ are irrelevant
  2. $\alpha_i > 0$: The corresponding $\mathbf{x}_i$ are critical
- $\mathbf{x}_k$ for which $\alpha_k > 0$ are called *support vectors*.

Linear SVM: Separable Case

- The SVM output is given by

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{w} \cdot \mathbf{x} + b \\ &= \sum_{k \in \mathcal{S}} \alpha_k y_k \mathbf{x}_k \cdot \mathbf{x} + b \end{aligned}$$

  where $\mathcal{S}$ is the set of indexes for which $\alpha_k > 0$.

- b can be computed by using the KTT condition, i.e., for any k such that $y_k = 1$ and $\alpha_k > 0$, we have

$$\begin{aligned} &\alpha_k [y_k(\mathbf{x}_k \cdot \mathbf{w} + b) - 1] = 0 \\ \Longrightarrow\ & b = 1 - \mathbf{x}_k \cdot \mathbf{w}. \end{aligned}$$

Linear SVM: Fuzzy Separation (Optional)

- If the data patterns are not separable by a linear hyperplane, a set of slack variables $\{\xi = \xi_1, \ldots, \xi_N\}$ is introduced with $\xi_i \geq 0$ such that the inequality constraints in SVM become

$$y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i \quad \forall i = 1, \ldots, N. \tag{23}$$

- The *slack variables* $\{\xi_i\}_{i=1}^N$ allow some data to violate the constraints in Eq. 18.
- The value of ξ_i indicates the degree of violation of the constraint.
- The minimization problem becomes

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C \sum_i \xi_i, \quad \text{subject to} \quad y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i, \tag{24}$$

where C is a user-defined penalty parameter to penalize any violation of the safety margin for all training data.

Linear SVM: Fuzzy Separation (Optional)

- The new Lagrangian is

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i - \sum_{i=1}^{N} \alpha_i(y_i(\mathbf{x}_i \cdot \mathbf{w} + b) - 1 + \xi_i) - \sum_{i=1}^{N} \beta_i \xi_i, \quad (25)$$

  where $\alpha_i \geq 0$ and $\beta_i \geq 0$ are, respectively, the Lagrange multipliers to ensure that $y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 - \xi_i$ and that $\xi_i \geq 0$.

- Differentiating $L(\mathbf{w}, b, \alpha)$ w.r.t. $\mathbf{w}$, b, and ξ_i, we obtain the Wolfe dual:

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \quad (26)$$

  subject to $0 \leq \alpha_i \leq C$, $i = 1, \ldots, N$, $\sum_{i=1}^{N} \alpha_i y_i = 0$.

Linear SVM: Fuzzy Separation (Optional)

Three types of support vectors:

1. On the margin:

$C > \alpha_i > 0, \xi_i = 0$

$y_i(\mathbf{w}^T\mathbf{x}_i + b) = 1$

$\alpha_{11} = 0.44; \xi_{11} = 0$

$\alpha_1 = 2.85; \xi_1 = 0$

2. Inside the margin:

$\alpha_i = C; 0 < \xi_i < 2$

$y_i(\mathbf{w}^T\mathbf{x}_i + b) \le 1$

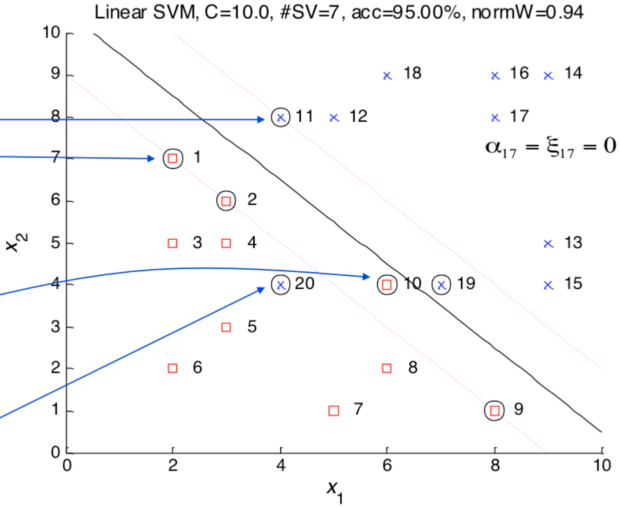
$\alpha_{10} = 10; \xi_{10} = 0.667$

3. Outside the margin:

$\alpha_i = C; \xi_i \ge 2$

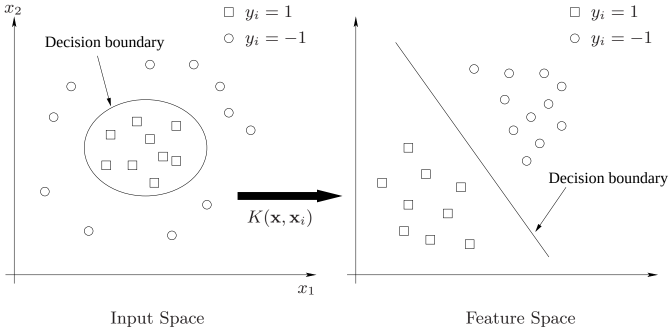
$y_i(\mathbf{w}^T\mathbf{x}_i + b) \le 1$

$\alpha_{20} = 10; \xi_{20} = 2.667$



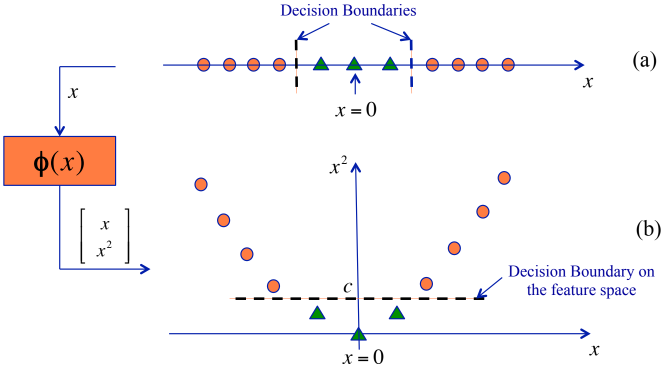
Nonlinear SVM

- Assume that we have a nonlinear function $\phi(\mathbf{x})$ that map $\mathbf{x}$ from the input space to a much higher (possibly infinite) dimensional space called the feature space.
- While data are not linearly separable in the input space, they will become linearly separable in the feature space.



Nonlinear SVM

- A 1-D problem requiring two decision boundaries (thresholds).
- 1-D linear SVMs could not solve this problem because they can only provide one decision threshold.



Nonlinear SVM

- We may use a nonlinear function ϕ to perform the mapping:

$$\phi : x \rightarrow [x \quad x^2]^\mathsf{T}.$$

- The decision boundary in the previous slide is a straight line that can perfectly separate the two classes.
- We may write the decision function as

$$x^2 - c = [0 \quad 1] \begin{bmatrix} x \\ x^2 \end{bmatrix} - c = 0$$

- Or equivalently,

$$\mathbf{w}^\mathsf{T} \phi(x) + b = 0, \tag{27}$$

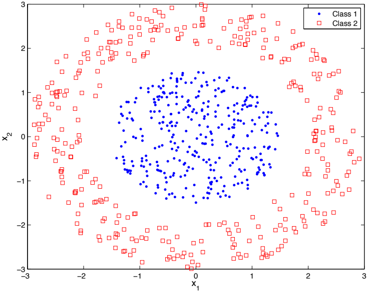
where $\mathbf{w} = [0 \quad 1]^\mathsf{T}$, $\phi(x) = [x \quad x^2]^\mathsf{T}$, and $b = -c$.

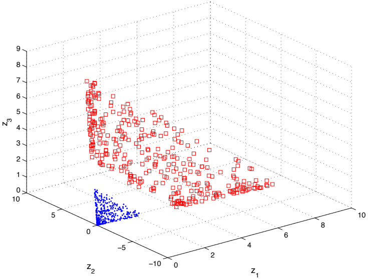
Nonlinear SVM

- *Left*: A 2-D example in which linear SVMs will not be able to perfectly separate the two classes.
- *Right*: By transforming $\mathbf{x} = [x_1 \;\; x_2]^\mathsf{T}$ to:

$$\boldsymbol{\phi} : \mathbf{x} \rightarrow [x_1^2 \;\; \sqrt{2}x_1x_2 \;\; x_2^2]^\mathsf{T}, \tag{28}$$

we will be able to use a linear SVM to separate the 2 classes in three dimensional space





Nonlinear SVM

- The SVM's decision function has the form

$$
\begin{aligned}
f(\mathbf{x}) &= \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)^{\mathsf{T}} \boldsymbol{\phi}(\mathbf{x}) + b \\
&= \mathbf{w}^{\mathsf{T}} \boldsymbol{\phi}(\mathbf{x}) + b,
\end{aligned}
$$

  where $\mathcal{S}$ is the set of support vector indexes and $\mathbf{w} = \sum_{i \in \mathcal{S}} \alpha_i y_i \boldsymbol{\phi}(\mathbf{x}_i)$.
- In this simple problem, the dot products $\boldsymbol{\phi}(\mathbf{x}_i)^{\mathsf{T}} \boldsymbol{\phi}(\mathbf{x}_j)$ for any $\mathbf{x}_i$ and $\mathbf{x}_j$ in the input space can be easily evaluated

$$
\boldsymbol{\phi}(\mathbf{x}_i)^{\mathsf{T}} \boldsymbol{\phi}(\mathbf{x}_j) = x_{i1}^2 x_{j1}^2 + 2 x_{i1} x_{i2} x_{j1} x_{j2} + x_{i2}^2 x_{j2}^2 = (\mathbf{x}_i^{\mathsf{T}} \mathbf{x}_j)^2. \quad (29)
$$

Nonlinear SVM

- The SVM output becomes

$$f(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i y_i \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}) + b$$

- However, the dimension of $\phi(\mathbf{x})$ is very high and could be infinite in some cases, meaning that this function may not be implementable.
- Fortunately, the dot product $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x})$ can be replaced by a kernel function:

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

which can be efficiently implemented.

Nonlinear SVM

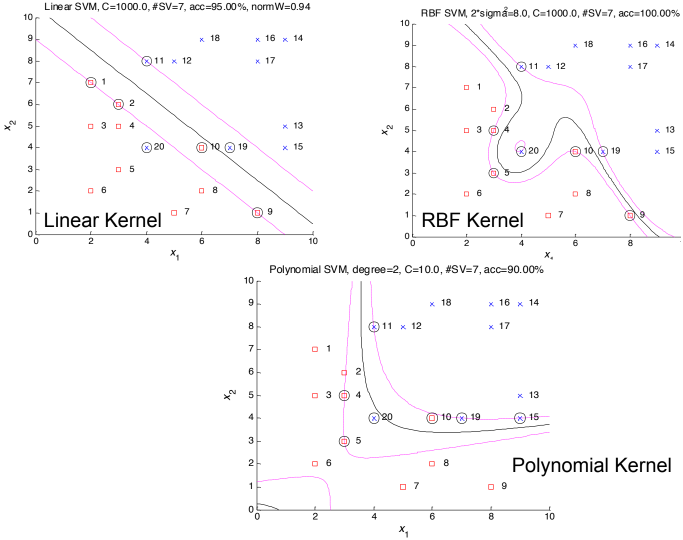
- Common kernel functions include

$$\text{Polynomial Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \left(1 + \frac{\mathbf{x} \cdot \mathbf{x}_i}{\sigma^2}\right)^p, \quad p > 0 \tag{30}$$

$$\text{RBF Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \exp\left\{-\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{2\sigma^2}\right\} \tag{31}$$

$$\text{Sigmoidal Kernel} \quad : \quad K(\mathbf{x}, \mathbf{x}_i) = \frac{1}{1 + e^{-\frac{\mathbf{x} \cdot \mathbf{x}_i + b}{\sigma^2}}} \tag{32}$$

- Comparing kernels:



Nonlinear SVM

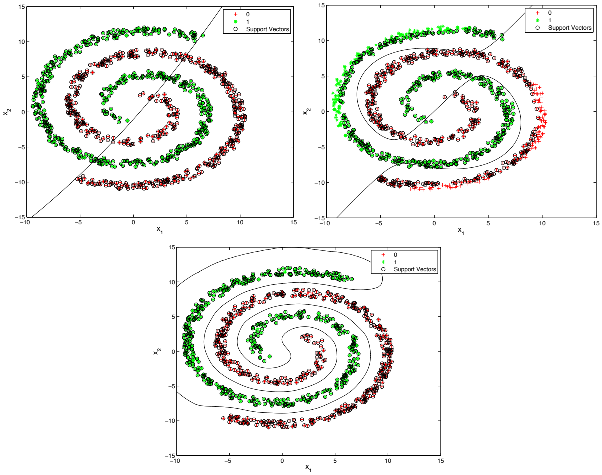
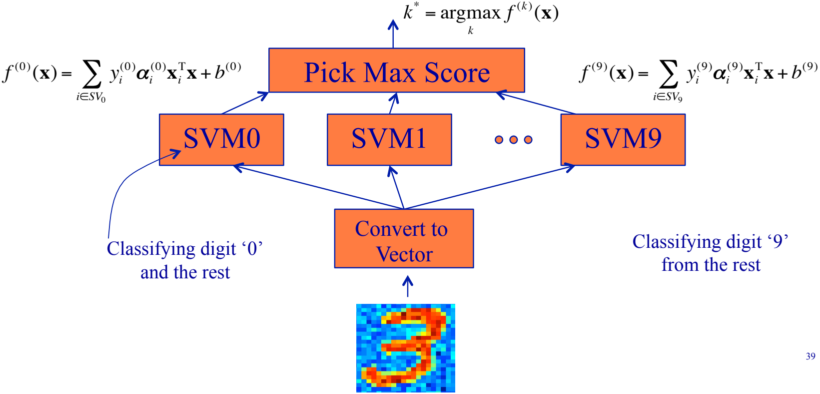


Figure: Decision boundaries produced by a 2nd-order polynomial kernel (top), a 3rd-order polynomial kernel (left), and an RBF kernel (right).

- SVM is good for binary classification:
  $f(\mathbf{x}) > 0 \Rightarrow \mathbf{x} \in$ Class 1; $f(\mathbf{x}) \leq 0 \Rightarrow x \in$ Class 2
- To classify multiple classes, we use the one-vs-rest approach to converting K binary classifications to a K-class classification:



Software Tools for SVM

- **Matlab**: `fitcsvm` trains an SVM for two-class classification.
- **Python**: `svm` from the `sklearn` package provides a set of supervised learning methods used for classification, regression and outliers detection.
- **C/C++**: `LibSVM` is a library for SVM. It also has Java, Perl, Python, Cuda, and Matlab interface.
- **Java**: `SVM-JAVA` implements sequential minimal optimization for training SVM in Java.
- **Javascript**: http://cs.stanford.edu/people/karpathy/svmjs/demo/