COMP5211: Machine Learning

Lecture 5

Minhao Cheng

Adagrad

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\varepsilon I + diag(G_t)}} \cdot g_t, \tag{1}$$

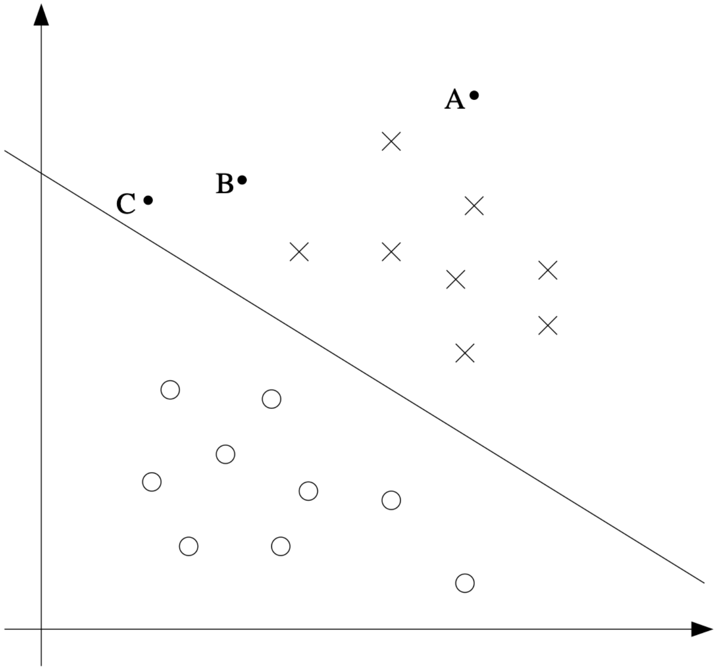
$$g_t = \frac{1}{n}\sum_{i=1}^{n} \nabla_\theta \mathcal{L}(x^{(i)}, y^{(i)}, \theta_t), \tag{2}$$

$$G_t = \sum_{\tau=1}^{t} g_\tau g_\tau^\top. \tag{3}$$

$$\begin{bmatrix} \theta_{t+1}^{(1)} \\ \theta_{t+1}^{(2)} \\ \vdots \\ \theta_{t+1}^{(m)} \end{bmatrix} = \begin{bmatrix} \theta_t^{(1)} \\ \theta_t^{(2)} \\ \vdots \\ \theta_t^{(m)} \end{bmatrix} - \begin{bmatrix} \frac{\eta}{\sqrt{\varepsilon + G_t^{(1,1)}}} & 0 & \cdots & 0 \\ 0 & \frac{\eta}{\sqrt{\varepsilon + G_t^{(2,2)}}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{\eta}{\sqrt{\varepsilon + G_t^{(m,m)}}} \end{bmatrix} \cdot \begin{bmatrix} g_t^{(1)} \\ g_t^{(2)} \\ \vdots \\ g_t^{(m)} \end{bmatrix}. \tag{5}$$

Support Vector Machine
Margin

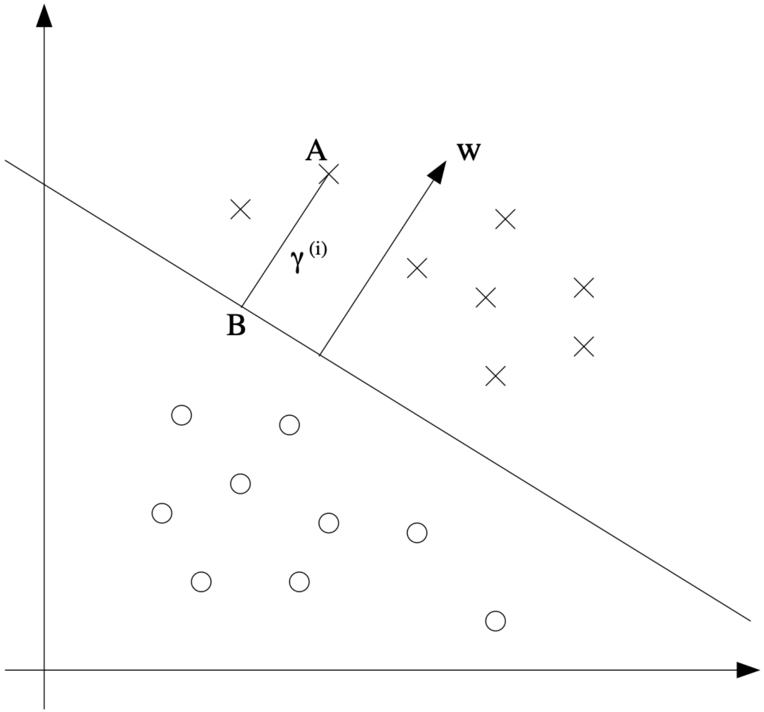
- Intuition
  - Confidence of A,B,C
- Let our decision function as
  - $h_{w,b} = g(w^T x + b)$
  - $g(z) = 1$ if $z \geq 0$, $g(z) = -1$ otherwise
- Function margins: $\hat{\gamma}^{(i)} = y^{(i)}(w^T x + b)$
- However, it will double just replace w with 2w, b with 2b



Support Vector Machine

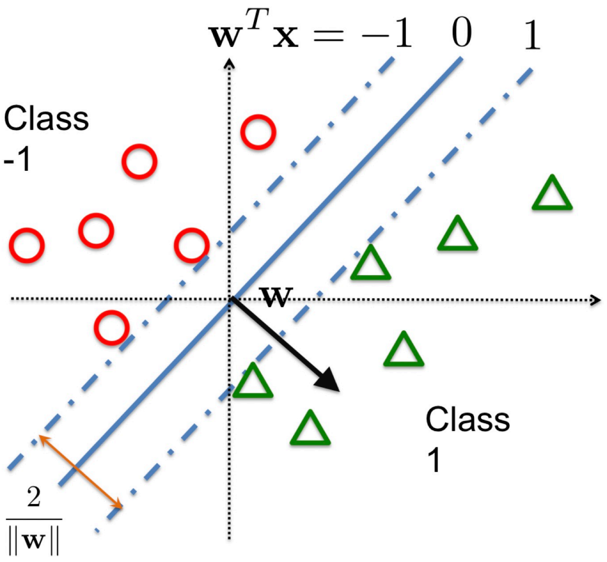
Margin

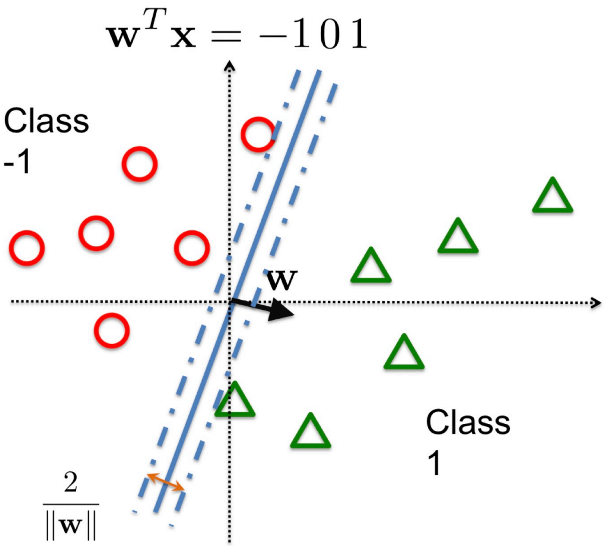
- Find point B:
  - $w^T(x^{(i)} - \gamma^{(i)} \frac{w}{\|w\|}) + b = 0$
- Geometric margin: $\gamma^{(i)} = \frac{w^T x^{(i)} + b}{\|w\|}$
- $\gamma = \min_{i=1,\ldots,N} \gamma^{(i)}$



Support Vector Machine

Max margin

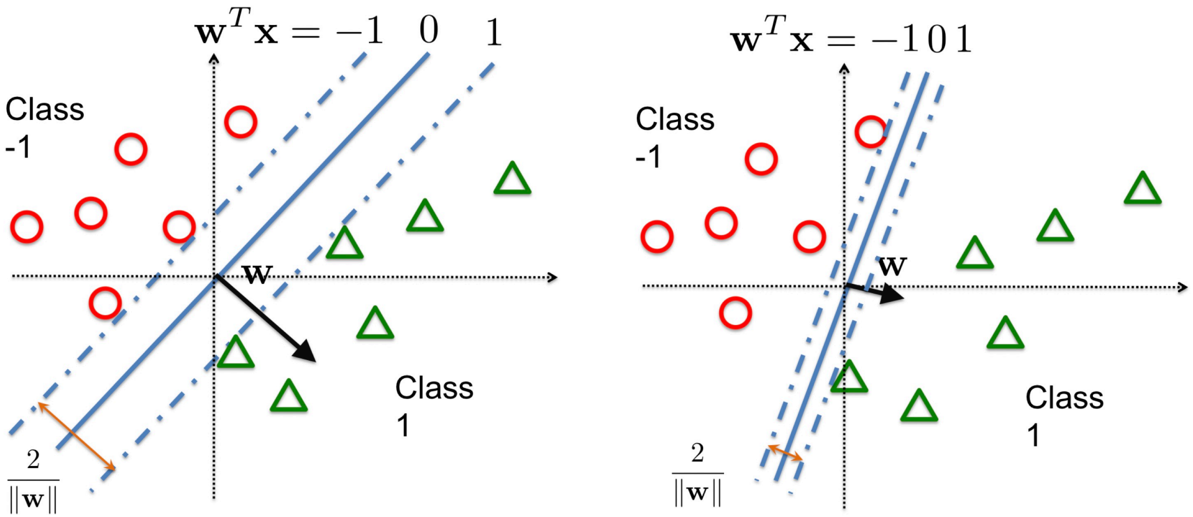




Support Vector Machine

Linear SVM

- Goal: Find a hyperplane to separate these two classes of data: if $y_i = 1, w^T x_i \geq 1$; if $y_i = -1, w^T x_i \geq -1$



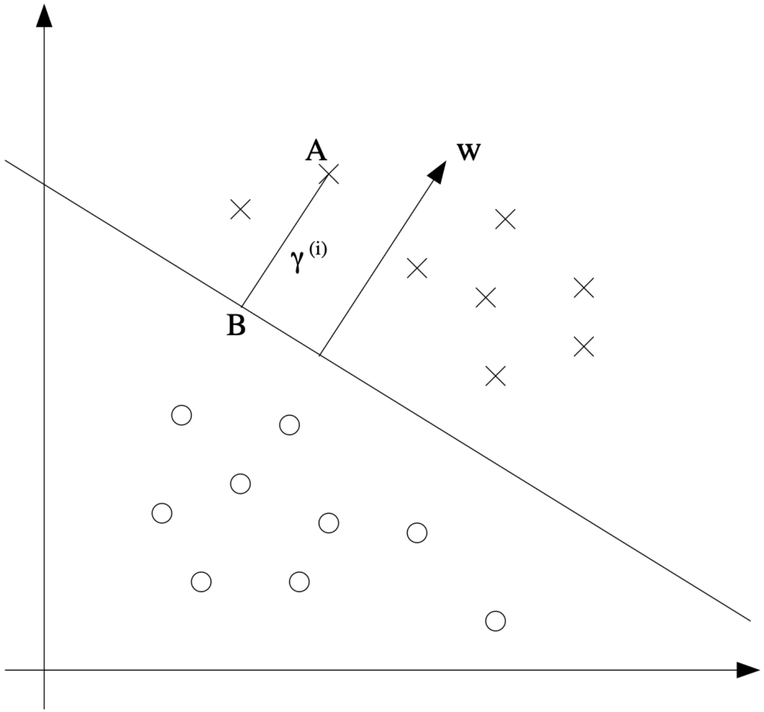
- Prefer a hyperplane with maximum margin

Support Vector Machine

Margin

- $\gamma^{(i)} = \dfrac{w^T x^{(i)} + b}{\|w\|}$
- $\gamma = \min\limits_{i=1,\ldots,m} \gamma^{(i)}$
- $\max\limits_{\gamma,w,b} \quad \dfrac{\gamma}{\|w\|}$

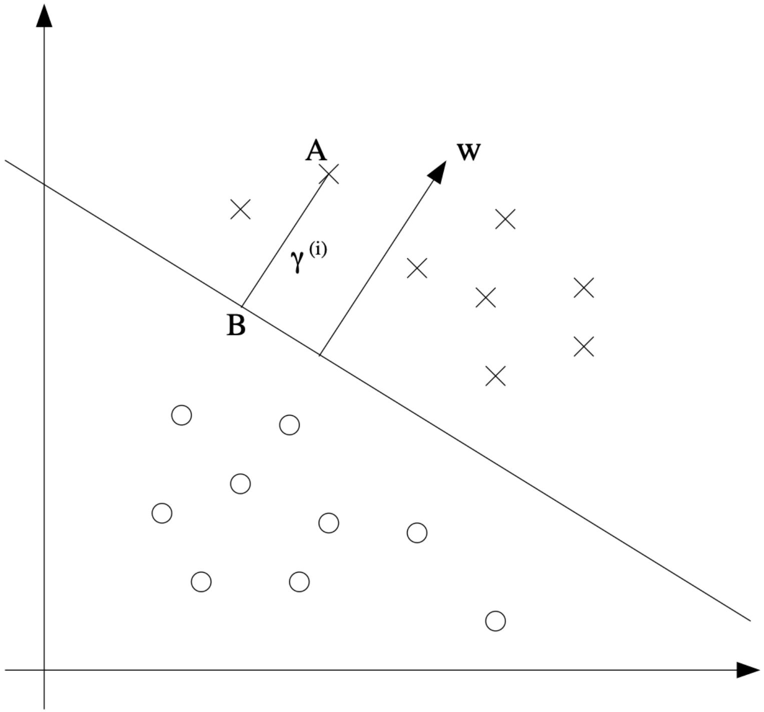
  s.t. $\quad y_i(w^T x_i + b) \geq \gamma, i = 1,\ldots,n$



Support Vector Machine

Margin

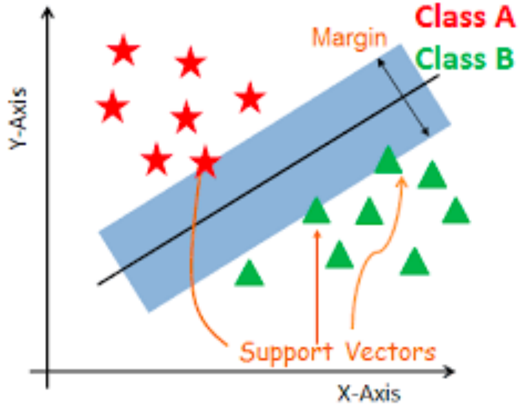
- $$\max_{\gamma,w,b} \quad \frac{\gamma}{\|w\|}$$
  s.t. $\quad y_i(w^T x_i + b) \geq \gamma, i = 1,\ldots,n$
- γ will increase linearly with w, b
- Just set $\gamma = 1$



Support Vector Machine

Margin

- $$\max_{\gamma,w,b} \quad \frac{\gamma}{\|w\|}$$
  $$\text{s.t.} \quad y_i(w^T x_i + b) \geq \gamma, i = 1,\ldots,n$$
- γ will increase linearly with w, b
- Just set $\gamma = 1$
- $$\max_{w,b} \quad \frac{1}{\|w\|}$$
  $$\text{s.t.} \quad y_i(w^T x_i + b) \geq 1, i = 1,\ldots,n$$



Support Vector Machine

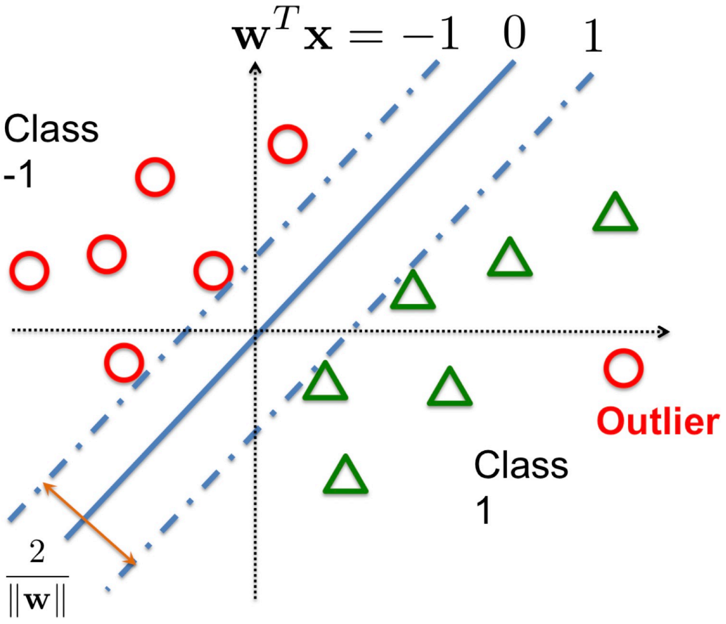
Linear SVM (hard constraint)

- SVM primal problem (with hard constraints):
- $$\min_{w} \quad \frac{1}{2} w^T w$$
  s.t. $\quad y_i(w^T x_i) \geq 1, i = 1, \ldots, n$

Support Vector Machine

Linear SVM (hard constraint)

- SVM primal problem (with hard constraints):
  - $$\min_{w} \quad \frac{1}{2} w^T w$$
    $$\text{s.t.} \quad y_i(w^T x_i) \geq 1, i = 1, \ldots, n$$
- What if there are outliers?



Support Vector Machine

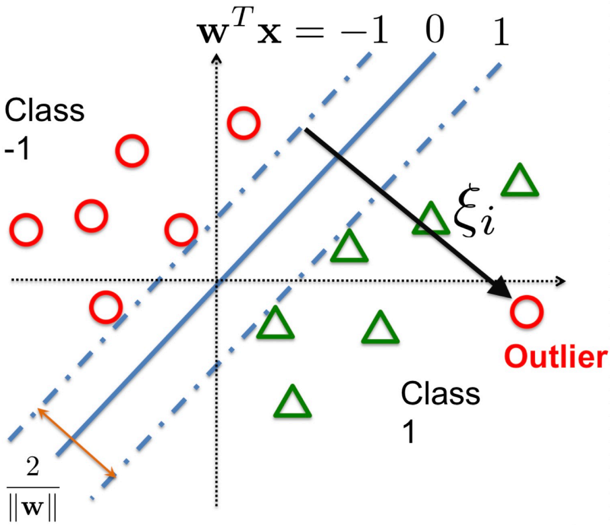
Linear SVM

- Given training examples $(x_1, y_1), \ldots, (x_n, y_n)$
  - Consider binary classification: $y_i \in \{+1, -1\}$
- Linear Support Vector Machine (SVM):

$$\min_{w} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i$$

- s.t. $\quad y_i(w^T x_i) \geq 1 - \xi_i, i = 1, \ldots, n$

  $\xi_i \geq 0$



Support Vector Machine

Linear SVM

- SVM primal problem:

$$\min_{w} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{n} \xi_i$$

- s.t. $\quad y_i(w^T x_i) \geq 1 - \xi_i, i = 1,\ldots,n$

$$\xi_i \geq 0$$

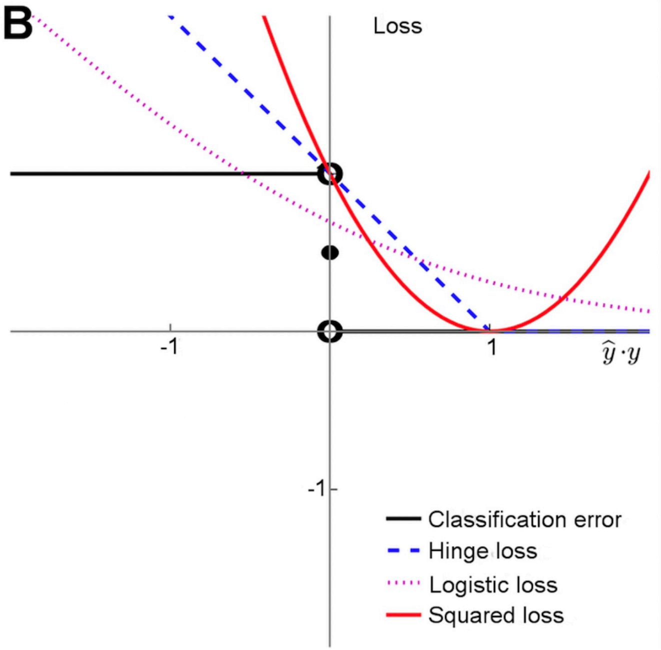
- Equivalent to

- $$\arg\min_{w} \underbrace{C\sum_{i=1}^{n} \max(1 - y_i w^T x_i, 0)}_{\text{hinge loss}} + \underbrace{\frac{1}{2}w^T w}_{\text{L2 regularization}}$$

- None-differentiable when $y_i w^T x_i = 1$ for some i

Support Vector Machine
Linear SVM

- Given training examples $(x_1, y_1), \ldots, (x_n, y_n)$
  - Consider binary classification: $y_i \in \{+1, -1\}$
- Linear Support Vector Machine (SVM):
  - $$\arg\min_{w} C \sum_{i=1}^{n} \max(1 - y_i w^T x_i, 0) + \frac{1}{2} w^T w$$
    - (Hinge loss with l2 regularization)



Support Vector Machine

Stochastic subgradient method for SVM

- A sub gradient of $\ell_i(w) = \max(0, 1 - y_i w^T x_i)$:
- $$\nabla_w \ell_i(w) = \begin{cases} -y_i x_i, & if\ 1 - y_i w^T x_i > 0 \\ 0, & if\ 1 - y_i w^t x_i < 0 \\ 0, & if\ 1 - y_i w^T x_i = 0 \end{cases}$$
- Stochastic subgradient descent for SVM:
- For $t = 1, 2, \ldots$
  Randomly pick an index i
  If $y_i \boldsymbol{w}^T \boldsymbol{x}_i < 1$, then
  $\boldsymbol{w} \leftarrow (1 - \eta_t)\boldsymbol{w} + \eta_t n C y_i \boldsymbol{x}_i$
  Else (if $y_i \boldsymbol{w}^T \boldsymbol{x}_i \geq 1$):
  $\boldsymbol{w} \leftarrow (1 - \eta_t)\boldsymbol{w}$

Support Vector Machine

Linear SVM

- SVM primal problem:

$$\min_{w} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{n} \xi_i$$

- s.t. $\quad y_i(w^T x_i) \geq 1 - \xi_i, i = 1,\ldots,n$
  $\xi_i \geq 0$

- Equivalent to

- $$\arg\min_{w} \underbrace{C\sum_{i=1}^{n} \max(1 - y_i w^T x_i, 0)}_{\text{hinge loss}} + \underbrace{\frac{1}{2}w^T w}_{\text{L2 regularization}}$$

- None-differentiable when $y_i w^T x_i = 1$ for some i

- Alternatively, we show how to derive the dual form of SVM

Support Vector Machine

Linear SVM dual

- SVM primal problem:

$$\min_{w} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{n} \xi_i$$

- s.t. $\quad y_i(w^T x_i) \geq 1 - \xi_i, i = 1,\ldots,n$

  $\xi_i \geq 0$

- Equivalent to: (using Lagrange multiplier):

- $\min_{w,\xi} \max_{\alpha \geq 0, \beta \geq 0} \frac{1}{2}\|w\|^2 + C\sum_{i} \xi_i - \sum_{i} \alpha_i(y_i w^T x_i - 1 + \xi_i) - \sum_{i} \beta_i \xi_i$

Support Vector Machine

Linear SVM dual form

- SVM primal problem:

$$\min_{w} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i$$

- s.t. $\quad y_i(w^T x_i) \geq 1 - \xi_i, i = 1,\ldots,n$

$$\xi_i \geq 0$$

- Equivalent to: (using Lagrange multiplier):

- $\min_{w,\xi} \max_{\alpha\geq 0,\beta\geq 0} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i - \sum_i \alpha_i(y_i w^T x_i - 1 + \xi_i) - \sum_i \beta_i \xi_i$

- Under certain condition (e.g., Slater's condition), exchanging $\min, \max$ will not change the optimal solution:

- $\max_{\alpha\geq 0,\beta\geq 0} \min_{w,\xi} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i - \sum_i \alpha_i(y_i w^T x_i - 1 + \xi_i) - \sum_i \beta_i \xi_i$

Support Vector Machine

Linear SVM dual form

- Reorganize the equation:
    - $\max_{\alpha \geq 0, \beta \geq 0} \min_{w, \xi} \frac{1}{2} \|w\|^2 - \sum_i \alpha_i y_i w^T x_i + \sum_i \xi_i (C - \alpha_i - \beta_i) + \sum_i \alpha_i$
- Now, for any given $\alpha, \beta,$ the minimizer of w will satisfy
    - $\frac{\partial L}{\partial w} = w - \sum_i \alpha_i y_i x_i = 0 \Rightarrow w^* = \sum_i y_i \alpha_i x_i$
    - Also, we have $C = \alpha_i + \beta_i$, otherwise ξ_i can make the objective function $-\infty$
- Subsititue these two equations back we get
    - $\max_{\alpha \geq 0, \beta \geq 0, C = \alpha + \beta} -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j + \sum_i \alpha_i$

Support Vector Machine

Linear SVM dual

- Therefore, we get the following dual problem
  - $\max_{C \geq \alpha \geq 0} \{-\frac{1}{2}\alpha^T Q \alpha + e^T \alpha\} := D(\alpha),$
  - Where Q is an $n \times n$ matrix with $Q_{ij} = y_i y_j x_i^T x_j$
- Based on the derivations, we know
  - Primal minimum = dual maximum (under Slater's condition)
  - Let α^* be the dual solution and w^* be the primal solution, we have
    - $w^* = \sum_i y_i \alpha_i^* x_i$
  - We can solve the dual problem instead of primal problem

Multi problem

Multi-class

- n data points, L labels, d features
- Input: training data $\{x_i, y_i\}_{i=1}^n$:
  - Each x_i is a d dimensional feature vector
  - Each $y_i \in \{1,\ldots,L\}$ is the corresponding label
  - Each training data belongs to one category
- Goal: find a function to predict the correct label
  - $f(x) \approx y$

Multi problems

Reduction to binary classification

- Many algorithms for binary classification
- Idea: transform multi-class or multi-label problems to multiple binary classification problems
- Two approaches:
    - One versus All (OVA)
    - One versus One (OVO)

Multi problems

One Versus All (OVA)

- Multi-class/multi-label problems with L categories
- Build L different binary classifiers
- For the t-th classifier:
  - Positive samples: all the points in class t ($\{x_i : t \in y_i\}$)
  - Negative samples: all the points not in class t ($\{x_i : t \notin y_i\}$)
  - $f_t(x)$: the decision value for the t-th classifier
    - (Larger $f_t \Rightarrow$ higher probability that x in class t)
- Prediction:
  - $f(x) = \arg\max_t f_t(x)$
- Example: using SVM to train each binary classifier

Multi problems
One Versus One (OVO)

- Multi-class/multi-label problems with L categories
- Build $L(L-1)$ different binary classifiers
- For the (s,t)-th classifier:
  - Positive samples: all the points in class s $(\{x_i : s \in y_i\})$
  - Negative samples: all the points in class t $(\{x_i : t \notin y_i\})$
  - $f_{s,t}(x)$: the decision value for the t-th classifier
    - (Larger $f_{s,t}(x) \Rightarrow$ label s has higher probability than label t)
    - $f_{t,s}(x) = -f_{s,t}(x)$
- Prediction:
  - $f(x) = \arg\max_s \left(\sum_t f_{s,t}(x)\right)$
- Example: using SVM to train each binary classifier

Multi problems

Another approach for multi-class classification

- OVA and OVO: decompose the problem by labels
  - But good binary classifiers may not imply good multi-class prediction
- Design a multi-class loss function and solve a single optimization problem

Multi problems

Another approach for multi-class classification

- OVA and OVO: decompose the problem by labels
  - But good binary classifiers may not imply good multi-class prediction
- Design a multi-class loss function and solve a single optimization problem
- Minimize the regularized training error:
  - $\min_{w_1,\ldots,w_L} \sum_{i=1}^{n} \text{loss}(x_i, y_i) + \lambda \sum_{j=1,\ldots,L} w_j^T w_j$

Multi problems

Main idea

- For simplicity, we assume a linear model
- Model parameters: $w_1, \ldots, w_L$
- For each data point x, compute the decision value for each label:
  - $w_1^T x, w_2^T x, \ldots, w_L^T x$
- Prediction:
  - $y = \arg\max_t w_t^T x$
- For training data x_i, y_i is the true label, so we want
  - $y_i \approx \arg\max_t w_t^T x_i, \quad \forall i$

Multi problems

Softmax

- The predicted score for each class:
  - $w_1^T x_i, w_2^T x_i, \ldots$
- Loss for the i-th data is defined by
  - $-\log(\frac{e^{w_{y_i}^T x_i}}{\sum_j e^{w_j^T x_i}})$ (probability of choosing the correct label)
- Solve a single optimization problem
  - $\min_{w_1,\ldots,w_L} \sum_{i=1}^{n} -\log(\frac{e^{w_{y_i}^T x_i}}{\sum_j e^{w_j^T x_i}}) + \lambda \sum_j w_j^T w_j$

Multi problems

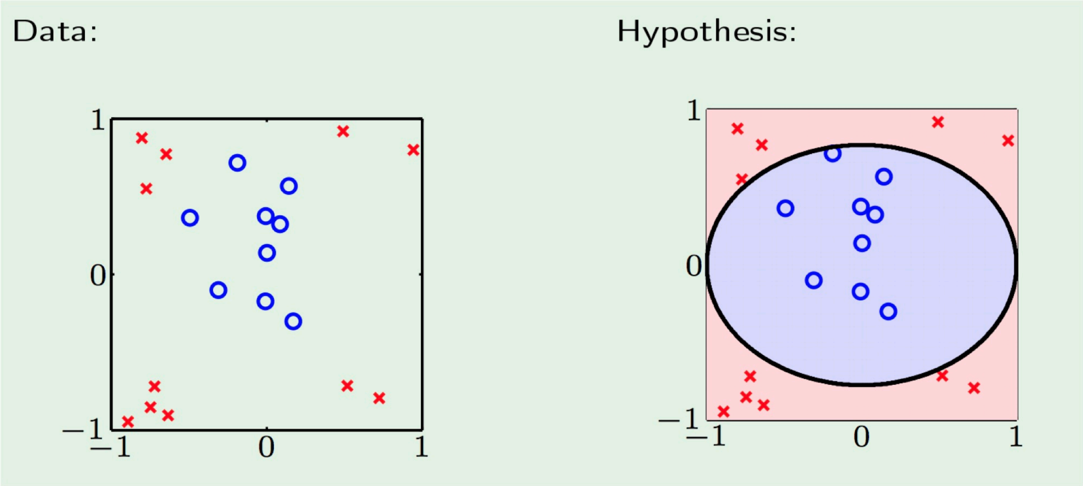
Loss functions for multi-class classification

- Ranking based approaches: directly minimizes the ranking loss:
    - For multi-class classification, the score of y_i should be larger than other labels
- Soft-max loss:
    - Measure the probability of predicting correct class

Nonlinear transformation

Linear hypotheses

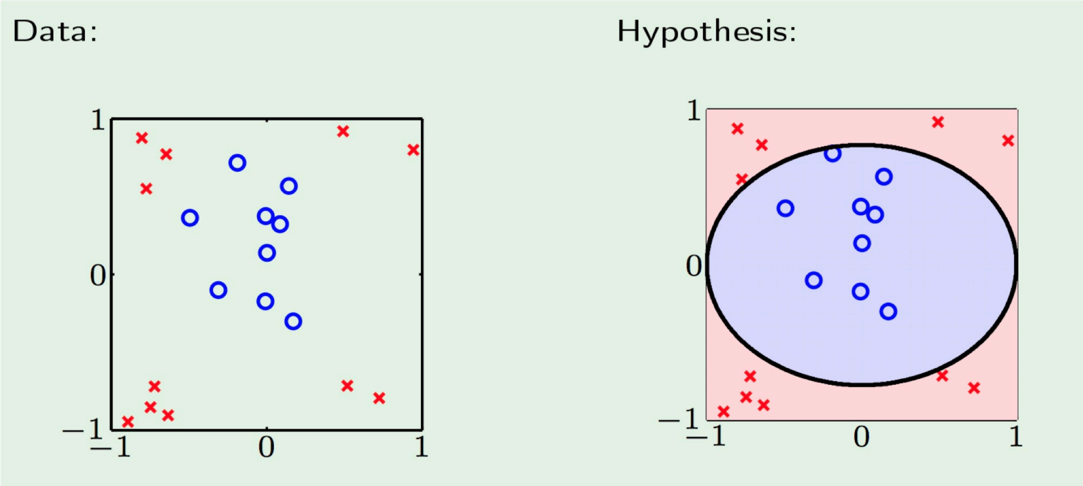
- Up to now: linear hypotheses
  - Perception, Linear regression, Logistic regression, …
- Many problems are not linearly separable



Nonlinear transformation

Circular Separable

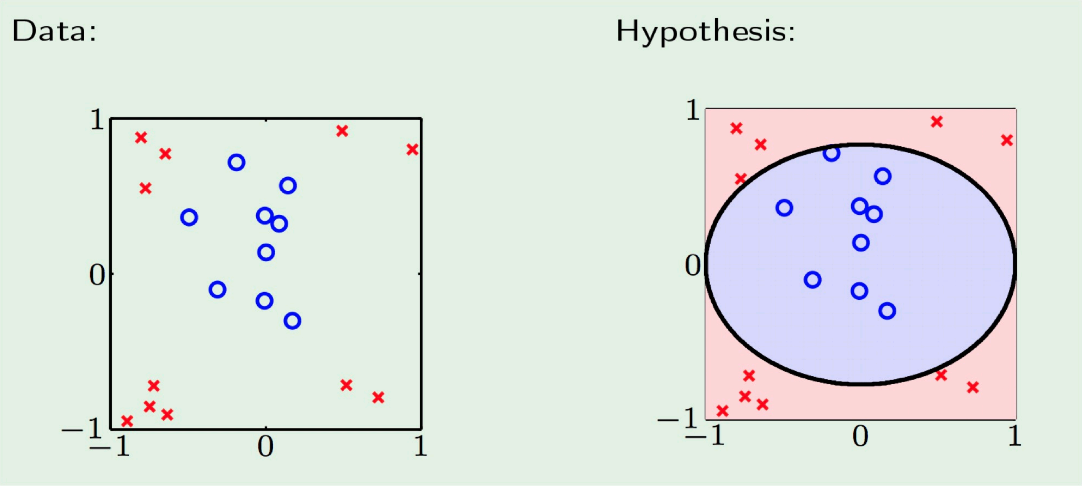
- Data is not linear separable



Nonlinear transformation

Circular Separable

- Data is not linear separable
- But circular separable by a circle of radius $\sqrt{0.6}$ centered at origin:
  - $h_{\text{SEP}}(x) = \text{sign}(-x_1^2 - x_2^2 + 0.6)$



Nonlinear transformation

Circular Separable and Linear Separable

- $h(x) = \text{sign}(0.6 \cdot 1 + (-1) \cdot x_1^2 + (-1) \cdot x_2^2)$

Nonlinear transformation

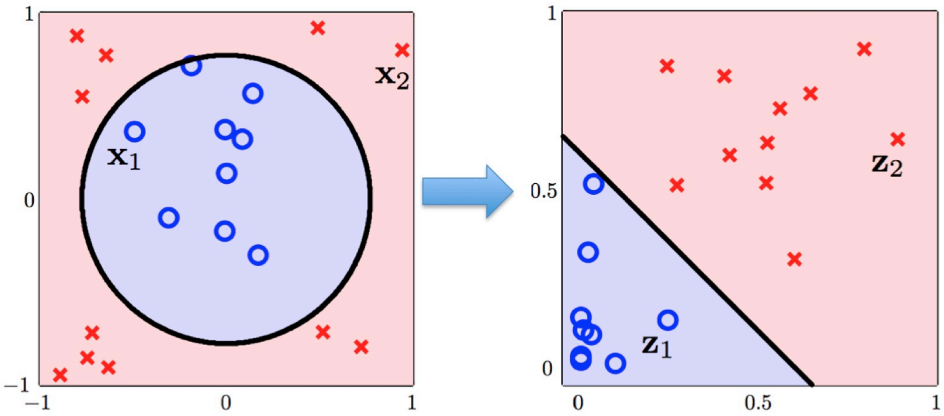
Circular Separable and Linear Separable

- $$h(x) = \text{sign}(\underbrace{0.6}_{\tilde{w}_0} \cdot \underbrace{1}_{\tilde{z}_0} + \underbrace{(-1)}_{\tilde{w}_1} \cdot \underbrace{x_1^2}_{\tilde{z}_1} + \underbrace{(-1)}_{\tilde{w}_2} \cdot \underbrace{x_2^2}_{\tilde{z}_2})$$
  $$= \text{sign}(\tilde{w}^T z)$$

Nonlinear transformation

Circular Separable and Linear Separable

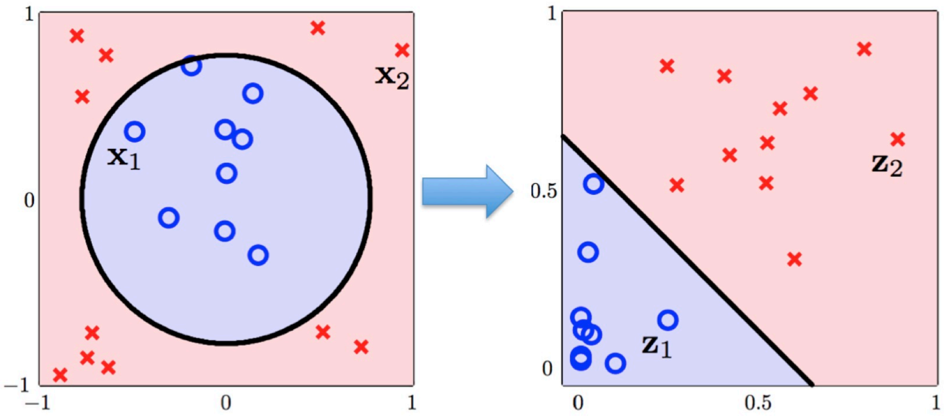
- $$h(x) = \text{sign}(\underbrace{0.6}_{\tilde{w}_0} \cdot \underbrace{1}_{\tilde{z}_0} + \underbrace{(-1)}_{\tilde{w}_1} \cdot \underbrace{x_1^2}_{\tilde{z}_1} + \underbrace{(-1)}_{\tilde{w}_2} \cdot \underbrace{x_2^2}_{\tilde{z}_2})$$
  $$= \text{sign}(\tilde{w}^T z)$$
- $\{(x_n, y_n)\}$ circular separable $\Rightarrow$ $\{(z_n, y_n)\}$ linear separable



Nonlinear transformation

Circular Separable and Linear Separable

- $$h(x) = \text{sign}(\underbrace{0.6}_{\tilde{w}_0} \cdot \underbrace{1}_{\tilde{z}_0} + \underbrace{(-1)}_{\tilde{w}_1} \cdot \underbrace{x_1^2}_{\tilde{z}_1} + \underbrace{(-1)}_{\tilde{w}_2} \cdot \underbrace{x_2^2}_{\tilde{z}_2})$$
  $$= \text{sign}(\tilde{w}^T z)$$
- $\{(x_n, y_n)\}$ circular separable $\Rightarrow$ $\{(z_n, y_n)\}$ linear separable
- $x \in \mathcal{X} \rightarrow x \in \mathcal{Z}$ (using a nonlinear transformation ϕ)



Nonlinear Transformation

Definition

- Define nonlinear transformation
  - $\phi(\mathbf{x}) = (1, x_1^2, x_2^2) = (z_0, z_1, z_2) = \mathbf{z}$

Nonlinear Transformation

Definition

- Define nonlinear transformation
  - $\phi(\mathbf{x}) = (1, x_1^2, x_2^2) = (z_0, z_1, z_2) = \mathbf{z}$
- Linear hypotheses in $\mathcal{Z}$ space:
  - $\text{sign}(\tilde{h}(\mathbf{z})) = \text{sign}(\tilde{h}(\phi(\mathbf{x}))) = \text{sign}(w^T \phi(\mathbf{x}))$

Nonlinear Transformation

Definition

- Define nonlinear transformation
  - $\phi(\mathbf{x}) = (1, x_1^2, x_2^2) = (z_0, z_1, z_2) = \mathbf{z}$
- Linear hypotheses in $\mathcal{Z}$-space:
  - $\text{sign}(\tilde{h}(\mathbf{z})) = \text{sign}(\tilde{h}(\phi(\mathbf{x}))) = \text{sign}(w^T \phi(\mathbf{x}))$
- Line in $\mathcal{Z}$-space $\Leftrightarrow$ some quadratic curves in $\mathcal{X}$-space

Nonlinear Transformation

General Quadratic Hypothesis Set

- A “bigger ” $\mathcal{Z}$ space:
  - $\phi_2(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$

Nonlinear Transformation

General Quadratic Hypothesis Set

- A "bigger " $\mathcal{Z}$-space:
    - $\phi_2(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$
- Linear in $\mathcal{Z}$-space $\Leftrightarrow$ quadratic hypotheses in $\mathcal{X}$-space

Nonlinear Transformation

General Quadratic Hypothesis Set

- A “bigger ” $\mathscr{Z}$-space:
  - $\phi_2(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$
- Linear in $\mathscr{Z}$-space ⇔ quadratic hypotheses in $\mathscr{X}$-space
- The hypotheses space:
  - $\mathscr{H}_{\phi_2} = \{h(x) : h(x) = \tilde{w}^T \phi_2(x)$ for some $\tilde{w}\}$ (quadratic hypotheses)

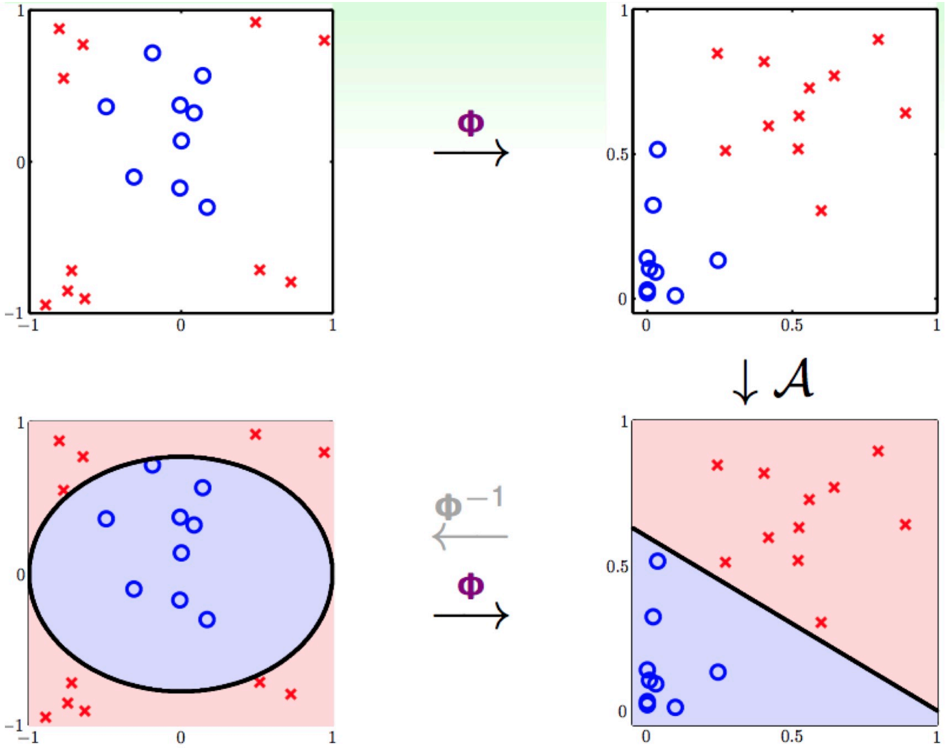
Nonlinear Transformation
General Quadratic Hypothesis Set

- A “bigger ” $\mathscr{Z}$-space:
  - $\phi_2(\mathbf{x}) = (1, x_1, x_2, x_1^2, x_1 x_2, x_2^2)$
- Linear in $\mathscr{Z}$-space ⇔ quadratic hypotheses in $\mathscr{X}$-space
- The hypotheses space:
  - $\mathscr{H}_{\phi_2} = \{h(x) : h(x) = \tilde{w}^T \phi_2(x)$ for some $\tilde{w}\}$ (quadratic hypotheses)
- Also include linear model as a degenerate case

Nonlinear transformation

Learning a good quadratic function

- Transform original data $\{x_n, y_n\}$ to $\{z_n = \phi(x_n), y_n\}$
- Solve a linear problem on $\{z_n, y_n\}$ using your favorite algorithm $\mathscr{A}$ to get a good model $\tilde{w}$
- Return the model $h(x) = \text{sign}(\tilde{w}^T \phi(x))$



Nonlinear transformation

Polynomial mappings

- Can now freely do quadratic classification, quadratic regression

Nonlinear transformation

Polynomial mappings

- Can now freely do quadratic classification, quadratic regression
- Can easily extend to any degree of polynomial mappings
  - E.g.,
    $\phi(x) = (x_1, x_2, x_3, x_1x_2, x_1x_3, x_2x_3, x_1x_2^2, x_1x_3^2, x_1x_2^2, x_2^2x_3, x_2^2x_3, x_1^3, x_2^3, x_3^3)$

Support Vector Machine

SVM with nonlinear mapping

- SVM with nonlinear mapping $\varphi(\,\cdot\,)$:

$$\min_{w} \quad \frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i$$

- s.t. $\quad y_i(w^T\varphi(x_i)) \geq 1 - \xi_i, i = 1,\ldots,n$

$$\xi_i \geq 0$$

- Hard to solve if $\varphi(\,\cdot\,)$ maps to very high or infinite dimensional space

Support Vector Machine

SVM with nonlinear mapping

- Similarly, we could derive
  - $\max_{C \geq \alpha \geq 0} \{-\frac{1}{2}\alpha^T Q \alpha + e^T \alpha\} := D(\alpha),$
  - Where Q is an $n \times n$ matrix with $Q_{ij} = y_i y_j \varphi(x_i)^T \varphi(x_j)$
- Based on the derivations, we know
  - Primal minimum = dual maximum (under Slater's condition)
  - Let α^* be the dual solution and w^* be the primal solution, we have
    - $w^* = \sum_i y_i \alpha_i^* \varphi(x_i)$

Nonlinear Transformation

The price we pay: computational complexity

- Q-th oder polynomial transform:

$$\phi(x) = (1, x_1, x_2, \ldots, x_d,$$
$$x_1^2, x_1 x_2, \ldots, x_d^2, \ldots, x_d^2,$$
$$\cdots$$
$$x_1^Q, x_1^{Q-1} x_2, \ldots, x_d^Q)$$

- $O(d^Q)$ dimensional vector $\Rightarrow$ High computational cost

Support Vector Machine

Kernel trick

- Do not directly define $\varphi(\,\cdot\,)$
- Instead, define "kernel"
  - $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$

Support Vector Machine
Kernel trick

- Do not directly define $\varphi(\cdot)$
- Instead, define "kernel"
  - $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$
- Examples:
  - Gaussian kernel: $K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$
  - Polynomial kernel: $K(x_i, x_j) = (\gamma x_i^T x_j + c)^d$
  - Other kernels for specific problems:
    - Graph kernels (Vishwanathan et al., "Graph Kernels", JMLR, 2010)
    - Pyramid kernel for image matching (Grauman and Darrell, "The Pyramid Match Kernel: Discriminative Classification with Sets of Image Features". In ICCV, 2005)
    - String kernel (Lodhi et al., "Text classification using string kernels". JMLR, 2002)

Support Vector Machine

SVM with kernel

- Training: compute $\alpha = [\alpha_1, \ldots, \alpha_n]$ by solving the quadratic optimization problem:
  - $\min_{0 \le \alpha \le C} \frac{1}{2}\alpha^T Q \alpha - e^T \alpha$
  - Where $Q_{ij} = y_i y_j K(x_i, x_j)$

Support Vector Machine

SVM with kernel

- Training: compute $\alpha = [\alpha_1, \ldots, \alpha_n]$ by solving the quadratic optimization problem:
  - $\min\limits_{0 \leq \alpha \leq C} \frac{1}{2} \alpha^T Q \alpha - e^T \alpha$
  - Where $Q_{ij} = y_i y_j K(x_i, x_j)$
- Prediction: for a test data x,
  - $w^T \varphi(x) = \sum_{i=1}^{n} y_i \alpha_i \varphi(x_i)^T \varphi(x) = \sum_{i=1}^{n} y_i \alpha_i K(x_i, x)$

Kernel trick

Kernel Ridge Regression

- Actually, this "kernel method" works for many different losses
- Example: ridge regression
- $$\min_{w} \frac{1}{2}\|w\|^2 + \frac{1}{2}\sum_{i=1}^{n}(w^T\varphi(x_i) - y_i)^2$$
- Dual problem:
- $$\min_{\alpha} \alpha^T Q\alpha + \|\alpha\|^2 - 2\alpha^T y$$

Kernel trick

Scalability

- Challenge for solving kernel SVMs (for dataset with n samples):
  - Space: $O(n^2)$ for storing the $n \times n$ kernel matrix (can be reduced in some cases);
  - Time: $O(n^3)$ for computing the exact solution

Kernel trick

Scalability

- Challenge for solving kernel SVMs (for dataset with n samples):
  - Space: $O(n^2)$ for storing the $n \times n$ kernel matrix (can be reduced in some cases);
  - Time: $O(n^3)$ for computing the exact solution
- Good packages available:
  - LIBSVM (can be called in scikit-learn)
  - LIBLINEAR (for linear SVM, can be called in scikit-learn)