CS-E4710 Machine Learning: Supervised Methods

Lecture 6: Support vector machines

Juho Rousu

10. October, 2023

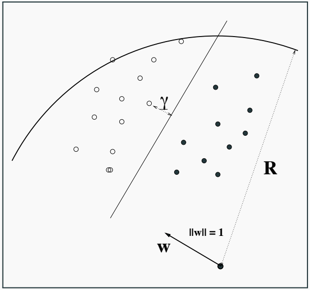
Department of Computer Science
Aalto University

Finding optimal separating hyperplanes

Recall: Perceptron algorithm on linearly separable data

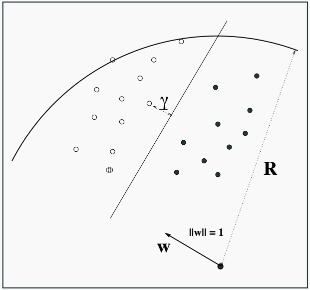
Recall the upper bound $(\frac{2R}{\gamma})^2$ of iterations of perceptron algorithm on linearly separable data

- γ: The largest achievable geometric margin in the training set, $\frac{y_i \mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|} \geq \gamma$ for all $i = 1 \dots, m$
- $R = \max_i \|\mathbf{x}_i\|$: The smallest radius of the d-dimensional ball that encloses the training data



Finding optimal separating hyperplanes

- The perceptron algorithm is guaranteed find a consistent hyperplane if one exists
- All training data are on the correct side of the hyperplane
- However, typically there are several hyperplanes that are consistent
- Which one is the best?



Maximum margin hyperplane

One good solution is to choose the hyperplane $\mathbf{w}^T\mathbf{x} = 0$ that lies furthest away from the training data (maximizing the minimum geometric margin of the training examples):

$$\begin{aligned} &\textit{Maximize } \gamma \\ &\quad w.r.t. \;\; \text{variables } \mathbf{w} \in \mathbb{R}^d \\ &\textit{Subject to } \frac{y_i \mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|} \geq \gamma, \text{for all } i = 1, \ldots, m, \end{aligned}$$

The maximum margin hyperplane has good properties:

- Robustness: small change in the training data will not change the classifications too much
- Theoretically a large margin is tied to a low generalization error
- It can be found efficiently through incremental optimization

Support vector machines (SVM) are based on this principle

How to Maximize the Margin?

- However, the optimization problem

$$\begin{aligned} &\textit{Maximize } \gamma \\ &\quad w.r.t. \text{ variables } \mathbf{w} \in \mathbb{R}^d \\ &\textit{Subject to } \frac{y_i \mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|} \geq \gamma, \text{for all } i = 1, \ldots, m \end{aligned}$$

  does not give us a **unique** optimal weight vector $\mathbf{w}^*$

- This is because if $\mathbf{w}^*$ is a solution, then so is any vector $c\mathbf{w}^*, c > 0$ since

$$\frac{y_i (c\mathbf{w})^T \mathbf{x}_i}{\|c\mathbf{w}\|} = \frac{c y_i \mathbf{w}^T \mathbf{x}_i}{\sqrt{c^2 \mathbf{w}^T \mathbf{w}}} = \frac{c y_i \mathbf{w}^T \mathbf{x}_i}{c \|\mathbf{w}\|} = \frac{y_i \mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|}$$

- We can make the functional margin $y_i \mathbf{w}^T \mathbf{x}_i$ arbitrarily high just by scaling the norm of $\mathbf{w}$

How to Maximize the Margin?

- We could add a constraint $\|\mathbf{w}\| = 1$ to the optimization problem to get an unique answer.
- However, optimization would become more difficult to solve
- Instead, let us multiply the constraint on the geometric margin

$$\frac{y_i \mathbf{w}^T \mathbf{x}_i}{\|\mathbf{w}\|} \geq \gamma$$

  by $\|\mathbf{w}\|$ to obtain a an equivalent constraint on the functional margin

$$y_i \mathbf{w}^T \mathbf{x}_i \geq \gamma \|\mathbf{w}\|$$

- Now fix the functional margin to 1: $\gamma \|\mathbf{w}\| = 1$ which gives $\gamma = \frac{1}{\|\mathbf{w}\|}$
- To maximize γ, we should minimize $\|\mathbf{w}\|$ with the constraint of having functional margin of at least 1

Support vector machine (SVM)

The so called hard margin support-vector machine (SVM, Cortes & Vapnik, 1995) solves the margin maximization as follows:

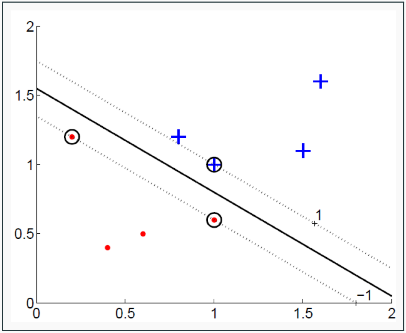
$$\begin{aligned} &\textit{Minimize } \frac{1}{2}\left\|\mathbf{w}\right\|^2 \\ &w.r.t. \;\text{ variables } \mathbf{w} \in \mathbb{R}^d \\ &\text{Subject to } y_i \mathbf{w}^T \mathbf{x}_i \geq 1, \text{for all } i = 1, \ldots, m \end{aligned}$$

- We are minimizing the half of the squared norm of the weight vector, which gives the same answer as minimizing the norm, but easier to optimize
- This is equivalent of finding the maximal geometric margin over the same data

Geometrical interpretation

The maximum margin hyperplane separates the positive and negative examples with a minimum functional margin of 1

- The points that have exactly margin $y\mathbf{w}^T\mathbf{x} = 1$ are called the support vectors
- The position of the hyperplane only depends on the support vectors, its position does not change if points with $y\mathbf{w}^T\mathbf{x} > 0$ are added or removed



Generalization capability of the maximum margin hyperplane

- The maximum margin hyperplane has significant theoretical backup
- Consider the hypothesis class

$$\mathcal{H} = \{h(\mathbf{x}) = \text{sgn}\left(\mathbf{w}^T\mathbf{x}\right) \mid \min_{i=1}^{m} y_i\mathbf{w}^T\mathbf{x}_i = 1, \|\mathbf{w}\| \leq B, \|\mathbf{x}_i\| \leq R\}$$

- The VC dimension satisfies $VCdim(\mathcal{H}) \leq B^2R^2$
- Rademacher complexity satisfies: $\mathcal{R}(H) \leq \frac{RB}{\sqrt{m}}$
- Thus a small norm ($\leq B$) translates to low complexity of the hypothesis class
- A better generalization error is thus likely if we can find a consistent hyperplane with a small norm (or, equivalently, a large margin)

Non-separable data

The so called hard margin support-vector machine assumes linearly separable data

$$\textit{Minimize } \frac{1}{2}\|\mathbf{w}\|^2$$
$$w.r.t. \;\text{ variables } \mathbf{w} \in \mathbb{R}^d$$
$$\text{Subject to } y_i\mathbf{w}^T\mathbf{x}_i \geq 1, \text{for all } i = 1, \ldots, m$$

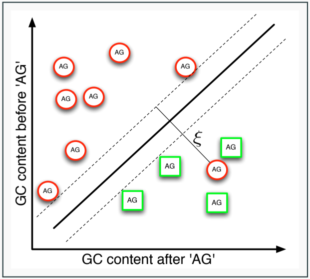
- In the non-separable case, for any hyperplane, there will be an example with a negative margin $y_i\mathbf{w}^T\mathbf{x}_i < 0$ which violates the constraint $y_i\mathbf{w}^T\mathbf{x}_i \geq 1$
- Our optimization problem has no feasible solution
- We need to extend our model to allow misclassified training points

Non-separable data

- To allow non-separable data, we allow the functional margin of some data points to be smaller than 1 by a slack variable $\xi_i \geq 0$
- The relaxed margin constraint will be expressed as

$$y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i, \xi_i \geq 0$$

- $\xi_i = 0$ corresponds to having large enough margin > 1
- $\xi_i > 1$ corresponds to negative margin, misclassified point
- The set of support vectors includes all $\mathbf{x}_i$ that have non-zero slack ξ_i (functional margin ≤ 1)



Soft-Margin SVM (Cortes & Vapnik, 1995)

The soft-margin SVM allows non-separable data by using the relaxed the margin constraints

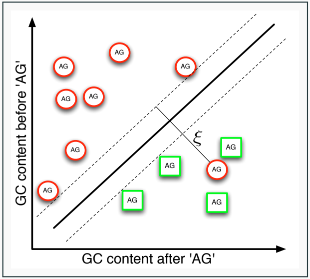
$$Minimize \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{m}\sum_{i=1}^{m}\xi_i$$

$$w.r.t \text{ variables } \mathbf{w}, \boldsymbol{\xi}$$

$$\text{Subject to } y_i\mathbf{w}^T\mathbf{x}_i \geq 1 - \xi_i$$

$$\text{for all } i = 1, \ldots, m.$$

$$\xi_i \geq 0, \text{for all } i = 1, \ldots, m.$$



- The sum (or average) of slack variables appear as a penalty in the objective
- The coefficient $C > 0$ controls the balance between model complexity (low C) and empirical error (high C)

The loss function in SVM

- We can interpret the soft-margin SVM in terms of minimization of a loss function
- Observe the relaxed margin constraint:

$$y_i \mathbf{w}^T \mathbf{x}_i \geq 1 - \xi_i, \xi_i \geq 0$$

- By rearranging, the same can be expressed as

$$\xi_i \geq 1 - y_i \mathbf{w}^T \mathbf{x}_i, \xi_i \geq 0$$

and further

$$\xi_i \geq \max(1 - y_i \mathbf{w}^T \mathbf{x}_i, 0)$$

- The right-hand side is so called Hinge loss:

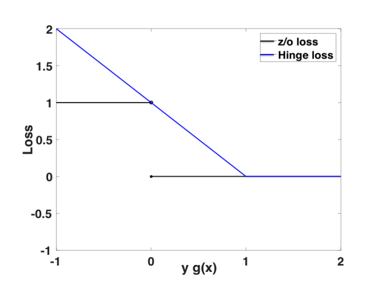
$$L_{Hinge}(y, \mathbf{w}^T \mathbf{x}) = \max(1 - y \mathbf{w}^T \mathbf{x}, 0)$$

Loss functions: Hinge loss

Hinge loss can be written for $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ as

$$L_{Hinge}(y, f(\mathbf{x})) = \max(1 - yf(\mathbf{x}), 0)$$

- Hinge loss is a convex upper bound of zero-one loss
- Hinge loss is zero if margin $y_i f(\mathbf{x}) \geq 1$
- For a misclassified example, margin is negative and Hinge loss is $\mathcal{L}_{Hinge}(f(\mathbf{x}), y_i)) > 1$
- The loss grows linearly in the margin violation $1 - yf(\mathbf{x})$, for margins < 1



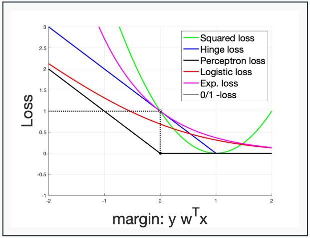
Interlude: Convex loss functions for classification

We have so far seen three different convex loss functions for classification:

- Perceptron loss
- Hinge loss
- Logistic loss

In addition there are multiple other convex loss functions:

- Squared loss - used for regression, not optimal for classification
- Exponential loss - used in boosting

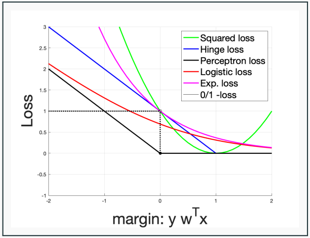


Because of convexity, all of the functions support fast optimization through gradient based appraoches, unlike the zero-one loss

Interlude: Convex loss functions for classification

- Three of the loss functions in the figure are not only convex, but also upper-bounds of zero-one loss: Exponential loss, Hinge loss, Squared loss

- Logistic loss can be scaled by a constant to make it an upper bound of zero-one loss
- Perceptron loss is not an upper bound of zero-one loss



- The benefit of being an upper bound of zero-one loss: if we minimize the upper bound, the zero-one loss will also usually be small
- Because of the convexity, minimization of the upper bound can be done fast, unlike minimization of zero-one loss

Optimization

Quadratic programming

The soft-margin SVM corresponds to a Quadratic program (QP)

$$Minimize \frac{1}{2}\left\|\mathbf{w}\right\|^2 + \frac{C}{m}\sum_{i=1}^{m}\xi_i$$

$$w.r.t \text{ variables } \mathbf{w}, \boldsymbol{\xi}$$

$$\text{Subject to } y_i\mathbf{w}^T\mathbf{x}_i \geq 1 - \xi_i$$

$$\text{for all } i = 1, \ldots, m.$$

$$\xi_i \geq 0, \text{for all } i = 1, \ldots, m.$$

- A QP is a convex optimization problem (with a unique optimum)
- The QP objective is a quadratic function of the variables
- The QP constraints are linear functions of the variables
- When data is small, QP solvers in optimization libraries can be used to solve the soft-margin SVM problem

Soft-margin SVM as a regularised learning problem

- We can rewrite the soft-margin SVM problem

$$\begin{aligned} \textit{Minimize} \quad & \tfrac{1}{2}||\mathbf{w}||^2 + \tfrac{C}{m}\textstyle\sum_{i=1}^{m} \xi_i \\ \text{Subject to} \quad & \xi_i \geq \max(1 - y_i \mathbf{w}^T \mathbf{x}_i, 0) \\ & \text{for all } i = 1, \ldots, N. \\ & \xi_i \geq 0 \end{aligned}$$

  equivalently in terms of Hinge loss as

$$\min_{\mathbf{w}} \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}_{Hinge}(\mathbf{w}^T \mathbf{x}_i, y_i) + \frac{\lambda}{2} ||\mathbf{w}||^2$$

- This is a so called **regularized learning problem**
  - First term minimizes a loss function on training data
  - Second term, called the **regularizer**, controls the complexity of the model
  - The parameter $\lambda = \frac{1}{C}$ controls the balance between the two terms

Optimization on big data

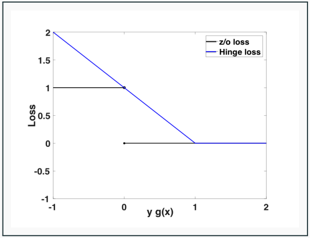
On big data, a stochastic gradient descent procedure is a good option

Rewrite the regularized learning problem as an average:

$$J(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m} J_i(\mathbf{w}) = \frac{1}{m}\sum_{i=1}^{m}\left(\mathcal{L}_{Hinge}(\mathbf{w}^T\mathbf{x}_i, y_i) + \frac{\lambda}{2}||\mathbf{w}||^2\right)$$

where $J_i(\mathbf{w}) = L_{Hinge}(\mathbf{w}^T\mathbf{x}_i, y_i) + \frac{\lambda}{2}||\mathbf{w}||^2$

However, Hinge loss is not differentiable at 1 (because of the 'Hinge' at 1), so cannot simply compute the gradient $\nabla J_i(\mathbf{w})$



Gradients of the Hinge loss

- We can differentiate the linear pieces of the loss separately

$$L_{Hinge}(\mathbf{w}^T\mathbf{x}_i, y_i) = \begin{cases} 1 - y_i\mathbf{w}^T\mathbf{x}_i, & \text{if } y_i\mathbf{w}^T\mathbf{x}_i < 1 \\ 0, & \text{if } y_i\mathbf{w}^T\mathbf{x}_i \geq 1 \end{cases}$$

- We get

$$\nabla L_{Hinge}(\mathbf{w}^T\mathbf{x}_i, y_i) = \begin{cases} -y_i\mathbf{x}_i & \text{if } y_i\mathbf{w}^T\mathbf{x}_i < 1 \\ \mathbf{0} & \text{if } y_i\mathbf{w}^T\mathbf{x}_i > 1 \end{cases}$$

- At $\mathbf{w}^T\mathbf{x}_i = 1$, the function is not differentiable but we can we choose $\mathbf{0}$ as the value, since the Hinge loss is zero so no update is needed to decrease loss
- (Formally $\mathbf{0}$ is one of the subgradients of the Hinge loss at 1, so can be justified from optimization theory)

Stochastic gradient descent algorithm for SVM

To find the update direction we express $J_i(\mathbf{w})$ as a piecewise differentiable function

$$J_i(\mathbf{w}) = L_{Hinge}(\mathbf{w}^T\mathbf{x}_i, y_i) + \frac{\lambda}{2}||\mathbf{w}||^2 = \begin{cases} 1 - y_i\mathbf{w}^T\mathbf{x}_i + \frac{\lambda}{2}||\mathbf{w}||^2, & \text{if } y_i\mathbf{w}^T\mathbf{x}_i < 1 \\ 0 + \frac{\lambda}{2}||\mathbf{w}||^2 & \text{if } y_i\mathbf{w}^T\mathbf{x}_i \geq 1 \end{cases}$$

Computing the derivatives piecewise gives the gradient:

$$\nabla J_i(\mathbf{w}) = \begin{cases} -y_i\mathbf{x}_i + \lambda\mathbf{w} & \text{if } y_i\mathbf{w}^T\mathbf{x}_i < 1 \\ \mathbf{0} + \lambda\mathbf{w} & \text{if } y_i\mathbf{w}^T\mathbf{x}_i \geq 1 \end{cases}$$

Update direction is the negative gradient $-\nabla J_i(\mathbf{w})$

Stochastic gradient descent algorithm for soft-margin SVM

Initialize $\mathbf{w} = 0$

repeat

Draw a training example (x_i, y_i) uniformly at random

Compute the update direction corresponding to the training example:

$$\nabla J_i(\mathbf{w})) = \begin{cases} -y_i \mathbf{x}_i + \lambda \mathbf{w} & \text{if } y_i \mathbf{w}^T \mathbf{x}_i < 1 \\ \lambda \mathbf{w} & \text{if } y_i \mathbf{w}^T \mathbf{x}_i \geq 1 \end{cases}$$

Determine a stepsize η

Update $\mathbf{w} = \mathbf{w} - \eta \nabla J_i(\mathbf{w})$

until stopping criterion satisfied

Output $\mathbf{w}$

- For the stepsize, diminishing stepsize of $\eta = 1/\lambda t$, has been suggested in the literature
- As the stopping criterion, one can use, e.g. the relative improvement of the objective between two successive iterations – stop iterations once it goes below given threshold

Interpreting the update

$$\mathbf{w} = \mathbf{w} - \eta \left(\lambda \mathbf{w} + \begin{cases} -y_i \mathbf{x}_i & \text{if } y_i \mathbf{w}^T \mathbf{x}_i < 1 \\ \mathbf{0} & \text{otherwise} \end{cases} \right)$$

- Each update shrinks the weight vector by $\eta\lambda \implies$ increases the geometric margin and adds regularization
- If the example has positive Hinge loss (functional margin < 1), we add $\eta y_i \mathbf{x}_i$ to the weight vector
- This has an effect of decreasing the Hinge loss on that example, similarly to the perceptron update

Interpreting the update

$$\mathbf{w} = \mathbf{w} - \eta \left(\lambda \mathbf{w} + \begin{cases} -y_i \mathbf{x}_i & \text{if } y_i \mathbf{w}^T \mathbf{x}_i < 1 \\ \mathbf{0} & \text{otherwise} \end{cases} \right)$$

- Compare to the perceptron update:

$$\mathbf{w} = \mathbf{w} + \begin{cases} y_i \mathbf{x}_i & \text{if } y_i \mathbf{w}^T \mathbf{x}_i < 0 \\ \mathbf{0} & \text{otherwise} \end{cases}$$

- Both share the idea of adding to $\mathbf{w}$ the training example multiplied by the label $y_i \mathbf{x}_i$, SVM does this to all examples that have too small margin, not only misclassified ones
- SVM shrinks the weight vector by fraction of λ on all examples, to regularize

Interpreting the update

- Consider the evolution of the weight vector $\mathbf{w}^{(t)}$ by the stochastic gradient optimization
- Assume $\lambda = 0$ and that $(\mathbf{x}^{(t)}, y^{(t)})$ is the t'th training example drawn by the algorithm that has positive Hinge loss, and $\eta^{(t)}$ is the learning rate
- Then we have

$$\begin{aligned}
\mathbf{w}^{(1)} &= \eta^{(1)} y^{(1)} \mathbf{x}^{(1)} \\
\mathbf{w}^{(2)} &= \eta^{(1)} y^{(1)} \mathbf{x}^{(1)} + \eta^{(2)} y^{(2)} \mathbf{x}^{(2)} \\
\mathbf{w}^{(3)} &= \eta^{(1)} y^{(1)} \mathbf{x}^{(1)} + \eta^{(2)} y^{(2)} \mathbf{x}^{(2)} + \eta^{(3)} y^{(3)} \mathbf{x}^{(3)} \\
\mathbf{w}^{(t)} &= \sum_{j=1}^{t} \eta^{(j)} y^{(j)} \mathbf{x}^{(j)}
\end{aligned}$$

- Thus the weight vector is a linear combination of the training examples that have been updated on so far

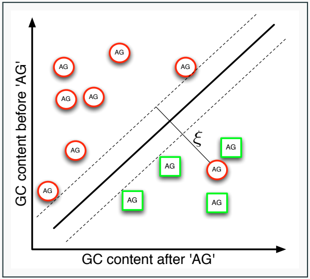
Dual soft-margin SVM

Dual representation of the optimal hyperplane

It can be shown theoretically that the **optimal hyperplane** of the soft-margin SVM has a **dual representation** as the linear combination of the training data

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$$

- The coefficients, also called the **dual variables** are non-negative $\alpha_i \geq 0$
- The positive coefficients $\alpha_i > 0$ appear if and only if $\mathbf{x}_i$ is a support vector, for other training points we have $\alpha_i = 0$



Dual representation of the optimal hyperplane

- Consequently, the functional margin $y\mathbf{w}^T\mathbf{x}$ also can be expressed using the support vectors:

$$y\mathbf{w}^T\mathbf{x} = y\sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i^T \mathbf{x}$$

- The norm of the weight vector can be expressed as

$$\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{m}\alpha_i y_i \mathbf{x}_i^T \sum_{j=1}^{m}\alpha_j y_j \mathbf{x}_j = \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

- Note that the training data appears in pairwise inner products: $\mathbf{x}_i^T\mathbf{x}_j$

Dual representations

- We can replace the explicit inner products with a kernel function

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

  which computes an inner product in the space of the arguments, here $\mathbb{R}^d$
- Plug in:
  - Margin:

$$y\mathbf{w}^T\mathbf{x} = y \sum_{i=1}^{m} \alpha_i y_i \kappa(\mathbf{x}_i, \mathbf{x})$$

  - Squared norm:

$$\|\mathbf{w}\|^2 = \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

Dual Soft-Margin SVM

A dual optimization problem for the soft-margin SVM with kernels is given by

$$\begin{aligned} \textit{Maximize} \quad & OBJ(\boldsymbol{\alpha}) = \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m \sum_{j=1}^m \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) \\ w.r.t \quad & \text{variables } \boldsymbol{\alpha} \in \mathbb{R}^m \\ \text{Subject to} \quad & 0 \leq \alpha_i \leq C/m \\ & \text{for all } i = 1, \ldots, m \end{aligned}$$

- It is a QP with variables α_i, again with a unique optimum
- At optimum, will have implicitly computed the optimal hyperplane $\mathbf{w} = \sum_{i=1}^m \alpha_i y_i \mathbf{x}_i$
- The data only appears through the kernel function $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j$
- Full derivation requires techniques of optimization theory, which we will skip here

Kernel trick

- We can consider transformations of the input with some basis functions $\phi : \mathbb{R}^d \mapsto \mathbb{R}^k$
- The optimal hyperplane $\mathbf{w} \in \mathbb{R}^k$ will satisfy: $\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \phi(\mathbf{x}_i)$
- Assume κ_ϕ computes an inner product in the space $\kappa_\phi(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$
- Then can compute the hyperplane in the transformed space

$$\mathbf{w}^T \phi(\mathbf{x}) = \sum_{i=1}^{m} \alpha_i y_i \kappa_\phi(\mathbf{x}_i, \mathbf{x})$$

  and the squared norm of the weight vector $\|\mathbf{w}\|^2 = \mathbf{w}^T \mathbf{w} = \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \kappa_\phi(\mathbf{x}_i, \mathbf{x}_j)$
- We do not need to explicitly refer to the transformed data $\phi(\mathbf{x})$ or the weight vector $\mathbf{w}$, both of which could be high-dimensional
- This is sometimes called the **kernel trick**

Stochastic Dual Coordinate Ascent

Stochastic Dual Coordinate Ascent for dual SVM

- Consider an algorithm updating one randomly selected dual variable (i.e. coordinate, hence the name of the method) α_i at a time , while keeping the other dual variables fixed
- We take the direction of the positive gradient of the dual SVM objective $OBJ(\boldsymbol{\alpha})$

$$\Delta\alpha_i = \frac{\partial}{\partial\alpha_i} OBJ(\boldsymbol{\alpha}) = \frac{\partial}{\partial\alpha_i}\left(\sum_{k=1}^{m}\alpha_k - \frac{1}{2}\sum_{k=1}^{m}\sum_{j=1}^{m}\alpha_k\alpha_j y_k y_j \kappa(\mathbf{x}_k, \mathbf{x}_j)\right)$$

$$= 1 - y_i \sum_{j=1}^{m}\alpha_j y_j \kappa(\mathbf{x}_i, \mathbf{x}_j) = 1 - y_i f(\mathbf{x}_i)$$

where we used the dual representation
$f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} = \sum_{j=1}^{m}\alpha_j y_j \kappa(\mathbf{x}_j, \mathbf{x})$

Stochastic Dual Coordinate Ascent for dual SVM

- The update direction thus depends on the margin:

$$\Delta\alpha_i = \begin{cases} < 0 & \text{if } y_i f(\mathbf{x}_i) > 1 \\ 0 & \text{if } y_i f(\mathbf{x}_i) = 1 \\ > 0 & \text{if } y_i f(\mathbf{x}_i) < 1 \end{cases}$$

- If the margin is too small (Hinge loss is positive), α_i is increased, if there is more than the required margin, α_i is decreased
- Note the analogy to updating $\mathbf{w} = \mathbf{w} + \eta y_i \mathbf{x}_i$, when $\mathbf{x}_i$ has too small margin : η and α_i have similar roles

Stepsize

- We can easily find the optimal update direction and step-size by setting
$$\frac{\partial}{\partial \alpha_i} OBJ(\boldsymbol{\alpha}) = 0,$$
it will give:
$$\alpha_i = \frac{1 - y_i \sum_{j \neq i} \alpha_j y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)}{\kappa(\mathbf{x}_i, \mathbf{x}_i)}$$
- Finally, the bounds $0 \leq \alpha_i \leq C/m$ need to be adhered
$\alpha_i = \min(C/m, \max(\alpha_i, 0))$

Stochastic Dual Coordinate Ascent for SVM

Initialize $\boldsymbol{\alpha} = \mathbf{0}$
repeat
Select a random training example (x_i, y_i)
Update the dual variable: $\alpha_i = \frac{1 - y_i \sum_{j \neq i} \alpha_j y_j \kappa(\mathbf{x}_i, \mathbf{x}_j)}{\kappa(\mathbf{x}_i, \mathbf{x}_i)}$
Clip to satisfy the constraints: $\alpha_i = \min(C/m, \max(0, \alpha_i))$
until stopping criterion is satisfied
return $\boldsymbol{\alpha}$

Summary

- Support vector machines are classification methods based on the principle of margin maximization
- SVMs can be efficiently optimized using Stochastic gradient techniques specially developed for piecewise differentiable functions, such as the higge loss
- Dual representation of SVM allows the use of kernel functions (more on kernels next lecture)