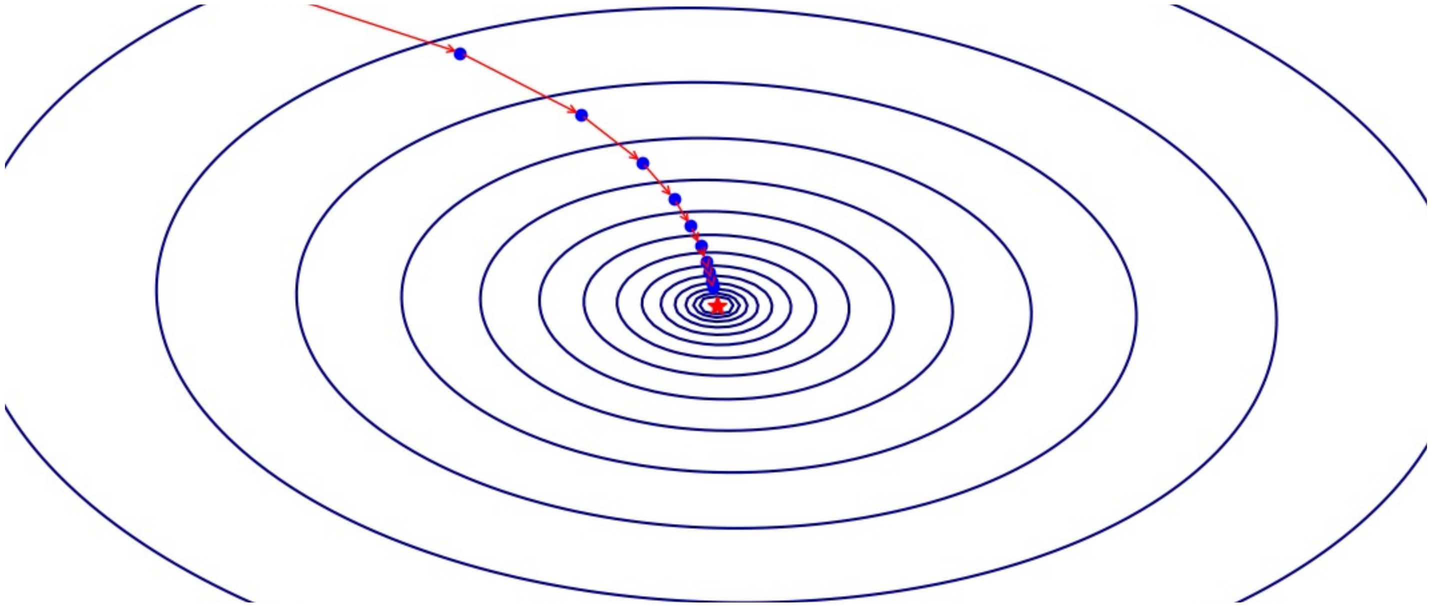


Machine Learning in Signal Processing

Winter Semester 2023/24

12. Support Vector Machines

06.02.2024

Prof. Dr. Vasileios Belagiannis

Chair of Multimedia Communications and Signal Processing

Friedrich-Alexander-Universität
Technische Fakultät

Course Topics

1. Introduction.
2. Basics and terminology.
3. Linear regression.
4. Linear classification.
5. Performance evaluation.
6. Neural networks.
7. Deep neural networks.
8. Decision trees.
9. Ensemble Learning.
10. Clustering.
11. Dimensionality reduction.
12. **Support vector machines.**
13. Recap and Q&A.

- The exam will be written.
- Test (ungraded) around the middle of the lectures.
- We will have an exam preparation test before the end of the year.

Acknowledgements
Ideas and inspiration from:
- CSC311 Introduction to Machine Learning, University of Toronto.
- Introduction to Machine Learning: LMU Munich.
- Introduction to Machine Learning, CSAIL, MIT.
- CSE 574 Introduction to Machine Learning, University of Buffalo.
- Special thanks Arij Bouazizi, Julia Hornauer, Julian Wiederer, Adrian Holzbock and Youssef Dawoud for contributing to the lecture preparation.

Last Lecture Recap

- Dimensionality reduction.
- PCA.
- Autoencoders.
- Stacked Autoencoders.
- t-SNE for visualisation.

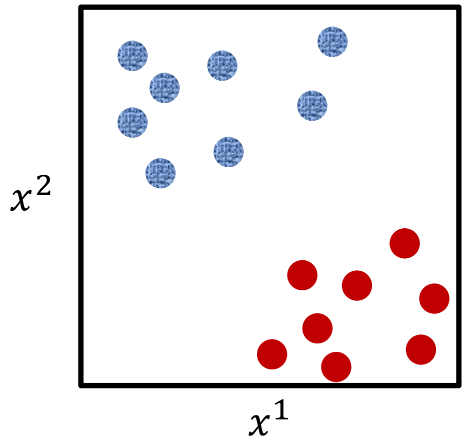
Today’s Agenda and Objectives

- Binary classification
- Separating hyperplanes
- Primal Support Vector Machines
- Dual Support Vector Machines
- Kernels
- Numerical optimisation

Acknowledgement
Our lecture today is based on Chapter 12 from the book Deisenroth, Marc Peter, A. Aldo Faisal, and Cheng Soon Ong. Mathematics for machine learning. Cambridge University Press, 2020.

Motivation

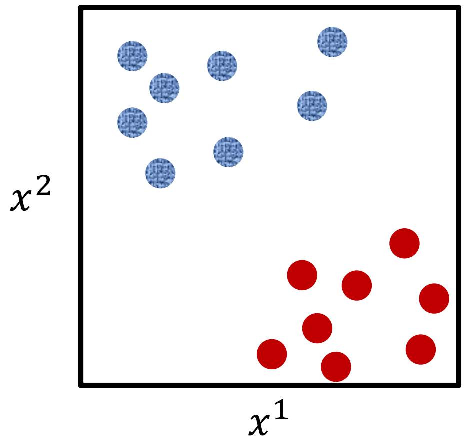
- We consider the problem of *binary classification*. For example, our task is to classify whether an incoming email is spam.
- We define our prediction function as:
  - $f: \mathbb{R}^D \longrightarrow \{+1, -1\}$.
  - There are positive and negative classes (true or false. It depends on the problem configuration.
- A support vector machine (SVM) is a supervised learning approach to preform classification, regression or anomaly detection.
- Here, we will formulate it only for the binary classification problem.
  - Given the training set $\{\{\boldsymbol{x_1}, \boldsymbol{y_1}\}, \dots, \{\boldsymbol{x_N}, \boldsymbol{y_N}\}\}$ with $\boldsymbol{x_n} \in \mathbb{R}^D$ and $\boldsymbol{y_n} \in \{+1, -1\}$, we would like to estimate the parameters $\boldsymbol{w}$ of our model f that will minimize the classification error.



Our data is linearly separable.

Motivation (Cont.)

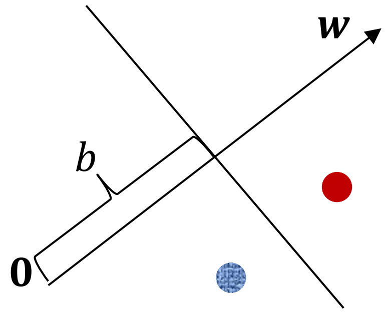
- The SVM will allows to reason a in geometric way. For that reason, we will design a loss function that has a geometric interpretation. The main concept in SVMs is the hyperplane.
- The hyperplane is a subspace with dimensions $D - 1$.
  - It has one less dimension than our feature space.
  - By considering our feature space to be affine, then we have an affine subspace and thus affine hyperplanes.
- *What is the hyperplane for a 2-dimensional feature space?*
- We seek for the hyperplane to separate the input samples into two categories, e.g., red / blue classes in the illustration.
  - It is also assumed that we have linearly separable samples.



Our data is linearly separable.

Separating Hyperplanes

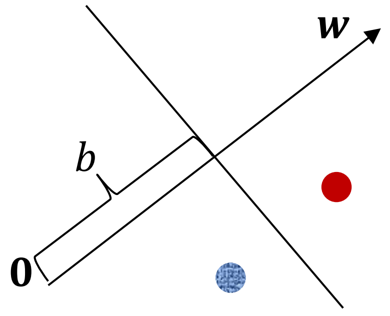
- Consider the similarity between the samples $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$, that can be as the inner product $\langle \boldsymbol{x_i}, \boldsymbol{x_j} \rangle$.
  - We will make use of the operation for the hyperplane separation.
- Our goal is to partition the <u>feature space </u>in a way that samples with the same class lie together in the same partition.
  - A simple approach is to linearly split the feature space into two halves with a hyperplane.



- We define the following function:
  - $\boldsymbol{x} \longmapsto f(\boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$
  - Where the input feature $\boldsymbol{x} \in \mathbb{R}^D$, parameters $\boldsymbol{w} \in \mathbb{R}^D$ and $b \in \mathbb{R}$.
  - The function f return a scalar.
- Also we define the hyperplane to separate the samples as:
  - $\{\boldsymbol{x} \in \mathbb{R}^D : f(\boldsymbol{x}) = 0\}$

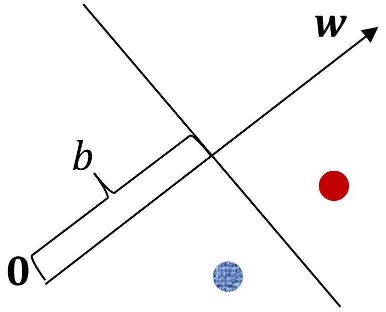
Separating Hyperplanes (Cont.)

- The $\boldsymbol{w}$ is normal to the hyperplane and b is the intercept.
- For $\boldsymbol{w}$ to be a normal vector to the hyperplane (perpendicular), we need to find two training samples $\boldsymbol{x_a}$ and $\boldsymbol{x_b}$ and show that the vector between them is orthogonal to $\boldsymbol{w}$. This can be defined as:
  - $f(\boldsymbol{x_a}) - f(\boldsymbol{x_b}) = (\langle \boldsymbol{w}, \boldsymbol{x_a} \rangle + b) - (\langle \boldsymbol{w}, \boldsymbol{x_b} \rangle + b) = \langle \boldsymbol{w}, \boldsymbol{x_a} - \boldsymbol{x_b} \rangle.$
  - Since $\boldsymbol{x_a}$ and $\boldsymbol{x_b}$ are on the hyperplane, we have $f(\boldsymbol{x_a}) = 0$ and $f(\boldsymbol{x_b}) = 0$.
  - Then we have $\langle \boldsymbol{w}, \boldsymbol{x_a} - \boldsymbol{x_b} \rangle = \boldsymbol{0}$. Since the inner produce is zero, the two vectors are orthogonal.
  - Finally $\boldsymbol{w}$ is orthogonal to any vector on the hyperplane.
- From the geometric view, we can think of $\boldsymbol{w}$ to be a vector and $\boldsymbol{x_a}$ and $\boldsymbol{x_b}$ as data points.
- The hyperplane separation $\{\boldsymbol{x} \in \mathbb{R}^D : f(\boldsymbol{x}) = 0\}$ also indicates a direction. We, thus, have the positive and genitive side of the hyperplane.
- Our problem become to classify whether a sample lies on the positive (above the hyperplane) or negative (below the hyperplane) side.
  - We expect then a value of to be positive $f(\boldsymbol{x}) \geq 0$ for the class $+1$ and negative for -1.



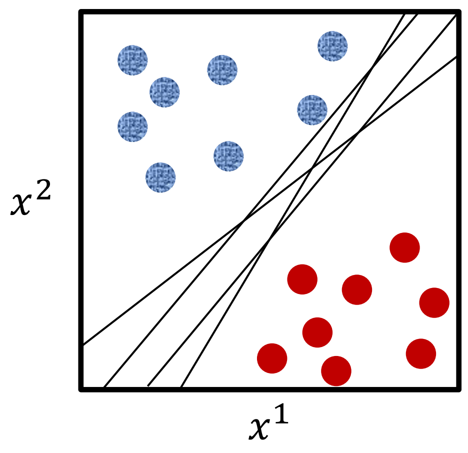
Separating Hyperplanes (Cont.)

- We formulate our learning problem as:
  - $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b \geq 0$ when $y_n = +1$
  - $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b < 0$ when $y_n = -1$
- Both conditions can be written in a single equation as:
  - $y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 0$
- Our classifier is based on the hyperplane for deciding the binary class of the input sample.
- *What parameters do we need to learn from the training set to perform the classification task?*



Primal SVM

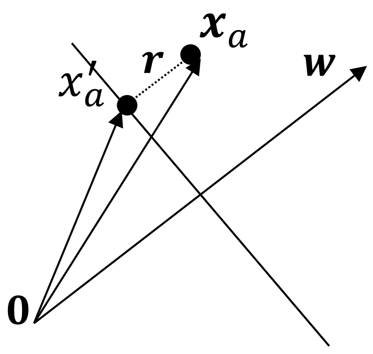
- Given the training set $\{\{\boldsymbol{x}_1, \boldsymbol{y}_1\}, \dots, \{\boldsymbol{x}_N, \boldsymbol{y}_N\}\}$, we assume that the features are linearly separable.
- We look for the splitting hyperplane that give the lowest classification error.
  - To find the unique solution, we can try to maximise the margin between the positive and negative samples.
  - The margin corresponds to the distance of the splitting hyperplane to the closest training samples.
  - We would like to find the maximum margin between positive and negative samples.



There are multiple splitting hyperplanes. Each split a different linear classifier.

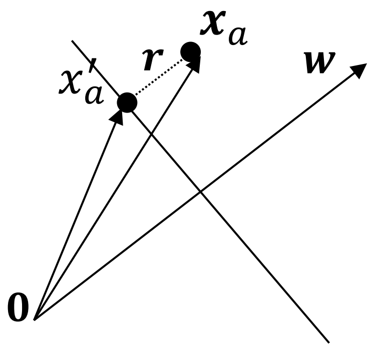
SVM Margin

- Consider a hyperplane $\langle \boldsymbol{w}, \boldsymbol{x} \rangle + b$. We can consider the sample $\boldsymbol{x}_a$ to be on the positive side such that $\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b > 0$.
  - The distance $r > 0$ of $\boldsymbol{x}_a$ from the hyperplane is illustrated in the figure. We can compute it by taking the orthogonal projection x_a' of $\boldsymbol{x}_a$ on the hyperplane.
  - $\boldsymbol{w}$ is orthogonal to the hyperplane and thus the distance r is a scaling of the vector $\boldsymbol{w}$.
  - We can obtain $\boldsymbol{x}_a$ as $\boldsymbol{x}_a = x_a' + r \frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$.
  - If $\boldsymbol{x}_a$ is the closest point to the hyperplane, then the distance r is the margin which we are looking for.



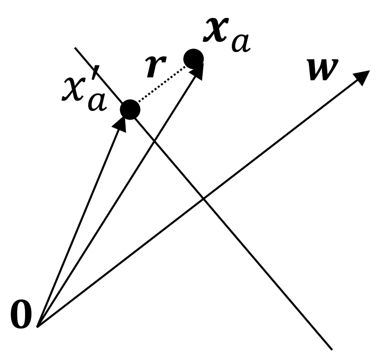
SVM Margin (Cont.)

- We aim for the positive samples to be further than the distance r from the hyperplane in the positive direction.
- Also, the negative samples should further than the distance r from the hyperplane in the negative direction.
- We can reformulate the inequality $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 0$ and now write it as:
  - $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq r.$
- We assume that the parameter vector $\boldsymbol{w}$ is of unit length and use the Euclidean norm as:
  - $\|\boldsymbol{w}\| = 1.$



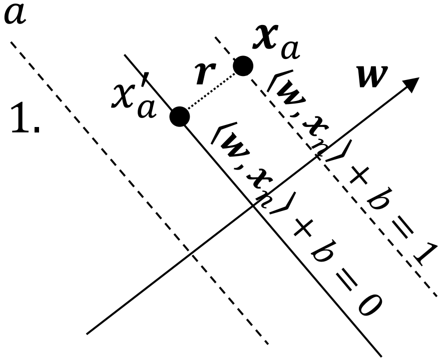
SVM Margin (Cont.)

- If we combine our requirements, we obtain:
  - $max_{\boldsymbol{w},b,r} r$
  - Subject to $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq r$, $\|\boldsymbol{w}\| = 1, r > 0.$
- The objective tell us that we want to maximise the margin while keeping the training samples on the right hyperplane side.
- We derived our objective function by assuming $\|\boldsymbol{w}\| = 1$ (unit length) and being interested only for the direction $\boldsymbol{w}$.



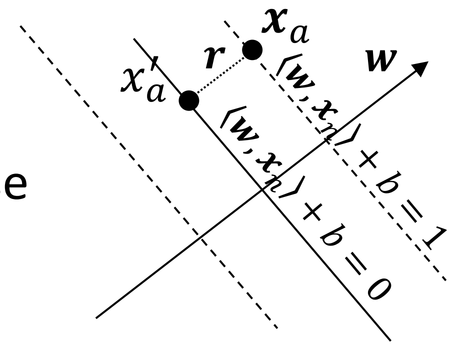
SVM Margin Derivation

- Instead of assuming that $\|\boldsymbol{w}\| = 1$, we can scale the data.
- The prediction $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b$ at the closest training sample $\boldsymbol{x}_a$ to the hyperplane is 1.
  - The sample $\boldsymbol{x}_a$ lies exactly on the margin and thus $\langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b = 1$.
  - The orthogonal projection x_a' lies on the hyperplane and thus $\langle \boldsymbol{w}, x_a' \rangle + b = 0$.
  - Recall that we have $\boldsymbol{x}_a = x_a' + r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|}$
  - By substitution we obtain $\left\langle \boldsymbol{w}, \boldsymbol{x}_a - r\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} \right\rangle + b = 0 \Rightarrow \langle \boldsymbol{w}, \boldsymbol{x}_a \rangle - r\frac{\langle \boldsymbol{w}, \boldsymbol{w} \rangle}{\|\boldsymbol{w}\|} + b = 0 \Rightarrow \langle \boldsymbol{w}, \boldsymbol{x}_a \rangle + b - r\frac{\|\boldsymbol{w}\|^2}{\|\boldsymbol{w}\|} = 0 \Rightarrow r = \frac{1}{\|\boldsymbol{w}\|}$.
  - $r = \frac{1}{\|\boldsymbol{w}\|}$ means that we express the distance w.r.t the normal vector $\boldsymbol{w}$.
  - Finally, the positive and negative training samples should be at least 1 aways from the hyperplane $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1$.


$\boldsymbol{x}_a$
$\boldsymbol{r}$
x_a'
$\boldsymbol{w}$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 1$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 0$

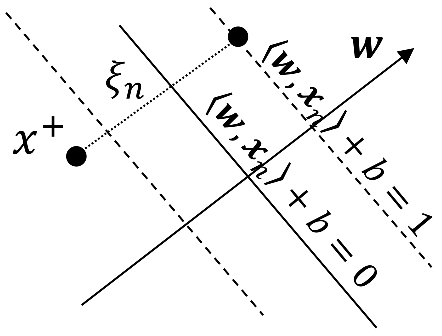
Hard Margin SVM

- We can now combine $r = \frac{1}{\|\boldsymbol{w}\|}$ and $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1$ to obtain:
  - $max_{\boldsymbol{w},b} \frac{1}{\|\boldsymbol{w}\|}$
  - Subject to $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1$.
- Instead of maximising the reciprocal norm, we can minimise the the square norm. We rewrite our optimisation then as:
  - $min_{\boldsymbol{w},b} \frac{1}{2} \|\boldsymbol{w}\|^2$
  - Subject to $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1$
  - The term $\frac{1}{2}$ adds numerical stability and does not change the solution.
  - This is known as hard margin SVM.
- The hard margin SVM formulation does not allow to violate the the margin condition. However, we can have cases where the data is not always linearly separable. We will need then a soft margin.


$\boldsymbol{x}_a$
$\boldsymbol{r}$
x_a'
$\boldsymbol{w}$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 1$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 0$

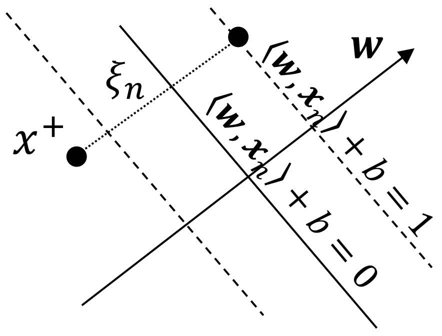
Soft Margin SVM

- When our data (the input features) are not linearly separable, we aim to relax our constraints and let samples to cross the margin to the wrong side or lie within the margin region.
  - The soft margin SVM formulation allows for this kind of errors.
- We introduce a slack variable ξ_n for each training sample $\boldsymbol{x}_n$ with label $\boldsymbol{y}_n$. It allows each sample be on the wrong side of the hyperplane or within the margin.
  - We can subtract the slack variable from the margin and constrain it to be positive as $y_n\,(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b) \geq 1 - \xi_n$.
- We add ξ_n to our objective function to optimise for the correct classification of the training samples.


$\boldsymbol{w}$
ξ_n
x^+
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 1$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 0$

Soft Margin SVM (Cont.)

- Our objective now becomes:
  - $min_{\boldsymbol{w},b} \frac{1}{2} ||\boldsymbol{w}||^2 + C \sum_{n=1}^{N} \xi_n$
  - Subject to $y_n \left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1 - \xi_n$ and $\xi_n \geq 0$.
- $C > 0$ indicates the influence of the slack variables to the minimization. It is called <u>regularisation parameter</u>.
- The margin term $||\boldsymbol{w}||^2$ is called regulariser.


$\boldsymbol{w}$
ξ_n
x^+
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 1$
$\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b = 0$

Dual SVM

- We derived the formulation for the primal SVM for the variables $\boldsymbol{w}$ and b.
  - We perform the operation $\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle$ which required the dimensions of $\boldsymbol{w}$ to be the same with the input features. The dimensions of $\boldsymbol{w}$ thus grows linearly with the dimensions of the input $\boldsymbol{x}_n$.
- The dual view of the problem detaches the optimisation from the dimensions of the input features. Instead, the dimensions of $\boldsymbol{w}$ will be affected from the size of the training set.
- The dual SVM allows to apply kernels to the input features for non-linear cases.

Convex Duality

- On primal SVM, the soft margin objective is given by:
  - $min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n$
  - Subject to $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) \geq 1 - \xi_n$ and $\xi_n \geq 0$.
  - The variables $\boldsymbol{w}, b, \xi$ are called primal variables.
- Now, we use the Langrage multipliers a_n and γ_n to reformulate the objective function as following:
  - $\mathfrak{L}(\boldsymbol{w}, b, \xi, a, \gamma) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N} a_n(y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n\rangle + b\right) - 1 + \xi_n) - \sum_{n=1}^{N}\gamma_n\xi_n.$
- Next we differentiate $\mathfrak{L}(\boldsymbol{w}, b, \xi, a, \gamma)$ w.r.t each primal variables to obtain:
  - $\frac{\partial \mathfrak{L}}{\partial \boldsymbol{w}} = \boldsymbol{w}^{\mathrm{T}} - \sum_{n=1}^{N} a_n y_n \boldsymbol{x}_n^{\mathrm{T}}$
  - $\frac{\partial \mathfrak{L}}{\partial b} = -\sum_{n=1}^{N} a_n y_n$
  - $\frac{\partial \mathfrak{L}}{\partial \xi_n} = C - a_n - \gamma_n$

Convex Duality (Cont.)

- We first set $\frac{\partial \mathfrak{L}}{\partial \boldsymbol{w}} = \boldsymbol{w}^{\mathrm{T}} - \sum_{n=1}^{N} a_n y_n {\boldsymbol{x}_n}^{\mathrm{T}}$ to zero for finding the maximum.
  - $\boldsymbol{w} = \sum_{n=1}^{N} a_n y_n \boldsymbol{x}_n$.
  - The optimal weight vector in the primal formulation is a linear combination of the input samples.
- Samples with $a_n = 0$ will not contribute to the final solution $\boldsymbol{w}$. The ones with $a_n > 0$ will contribution and that is why they are called support vectors. They support the hyperplane.
- We can substitute $\boldsymbol{w} = \sum_{n=1}^{N} a_n y_n \boldsymbol{x}_n$ to the objective function $\mathfrak{L}(\boldsymbol{w}, b, \xi, a, \gamma)$ to obtain:
  - $\mathfrak{D}(\xi, a, \gamma) = \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j a_i a_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} y_i a_i \left\langle \sum_{j=1}^{N} y_j a_j \boldsymbol{x}_j, \boldsymbol{x}_i \right\rangle + C \sum_{i=1}^{N} \xi_i - b \sum_{i=1}^{N} y_i a_i + \sum_{i=1}^{N} a_i - \sum_{i=1}^{N} a_i \xi_i - \sum_{i=1}^{N} \gamma_i \xi_i$.
  - The primal variable $\boldsymbol{w}$ is now completely gone.
- By setting $\frac{\partial \mathfrak{L}}{\partial b} = -\sum_{n=1}^{N} a_n y_n$ to zero, we obtain:
  - $\sum_{i=1}^{N} y_i a_i = 0$. As a result the term b vanishes completely.
- The inner products are symmetric and bilinear. That is why the first two terms of $\mathfrak{D}(\xi, a, \gamma)$ are over the same objects. Wen can thus simplify $\mathfrak{D}(\xi, a, \gamma)$ and refactor to:
  - $\mathfrak{D}(\xi, a, \gamma) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j a_i a_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \sum_{i=1}^{N} a_i - \sum_{i=1}^{N} (C - a_i - \gamma_i) \xi_i$.
- By setting $\frac{\partial \mathfrak{L}}{\partial \xi_n} = C - a_n - \gamma_n$ to zero and then substituting to $\mathfrak{D}(\xi, a, \gamma)$, we have:
  - $\mathfrak{D}(\xi, a, \gamma) = -\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j a_i a_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle + \sum_{i=1}^{N} a_i$.
  - This is our objective function to minimise.
  - Also we conclude from the gradient above that $a_i \leq C$.

Convex Duality (Cont.)

- Now we can summarise our optimisation as following:
  - $min_a \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j a_i a_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} a_i$
  - Subject to $\sum_{i=1}^{N} y_i a_i = 0$ and $0 \leq a_i \leq C$.
- $\sum_{i=1}^{N} y_i a_i = 0$ is obtained by setting $\frac{\partial \mathfrak{L}}{\partial b}$ to zero.
- The inequality $a_i \geq 0$ is the condition that is imposed on Lagrange multipliers.
- This set of inequality constraints are called <u>box constraints</u>. This is because they constrain the vector $\boldsymbol{a} = [a_1, \ldots, a_N]^{\mathrm{T}} \in \mathbb{R}^N$ of the Lagrange multipliers to be within a box that is defined by 0 and C on each axis.
- After solving for $\boldsymbol{a} = [a_1, \ldots, a_N]^{\mathrm{T}}$ we go back to primal variables and recover $\boldsymbol{w}^*$ with the equation $\boldsymbol{w} = \sum_{n=1}^{N} a_n y_n \boldsymbol{x}_n$ (previous slide).
- Finally, we need to obtain the parameter optimal parameter for b that we call b^*. We rely on the sample that lies on the margin for that. Consider $\boldsymbol{x}_n$ to be on the margin, then we have:
  - $b^* = y_n - \langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle$.
  - In practice we will not always some sample on the margin. We can instead take $|y_n - \langle \boldsymbol{w}^*, \boldsymbol{x}_n \rangle|$ for all support vectors and then compute the median value as the value of b^*.

Kernels

- In our dual optimisation, our objective function is given by:
  - $min_a \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j a_i a_j \langle \boldsymbol{x}_i, \boldsymbol{x}_j \rangle - \sum_{i=1}^{N} a_i$
  - Subject to $\sum_{i=1}^{N} y_i a_i = 0$ and $0 \leq a_i \leq C$.
  - Note that the <u>inner product</u> is between training samples $\boldsymbol{x}_i, \boldsymbol{x}_j$.
- The model parameters $\boldsymbol{w}$ are not involved in the minimisation of the dual variables.
  - We could then consider any kind of input features including non-linear representations.
- Consider the function $\varphi(\boldsymbol{x})$ to be non-linear and transform the input feature $\boldsymbol{x}$. We can use the SVM formulation to build a classifier with input $\varphi(\boldsymbol{x})$ that is non-linear in the examples.
  - In this way, we can deal with data that are not linearly separable. At the same time, we do not have to make changes at the objective function other than <u>replacing</u> the inner product.

Kernels (Cont.)

- Instead of the inner product, we now define a similarity function $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The similarity function can implicitly define a <u>non-linear mapping function</u> so that we don't have to define the non-linear function $\varphi(\cdot)$.
  - A kernel is function itself given by $k\colon \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$.
  - For a kernel there is a Hilbert space $\mathcal{H}$.
  - We have the feature map $\varphi\colon \mathcal{X} \rightarrow \mathcal{H}$ such that $k\left(\boldsymbol{x}_i, \boldsymbol{x}_j\right) = \left\langle \varphi(\boldsymbol{x}_i), \varphi\left(\boldsymbol{x}_j\right)\right\rangle_{\mathcal{H}}$.
  - Every kernel k is associated with a unique* Hilbert space $\mathcal{H}$ that is called canonical feature map.
- The reformulation of the inner product to a kernel function is known as the <u>kernel trick</u>** because it allows to use (hide) the non-linear feature map within the linear minimisation of the SVM.

*Berlinet, Alain, and Christine Thomas-Agnan. Reproducing kernel Hilbert spaces in probability and statistics. Springer Science & Business Media, 2011.

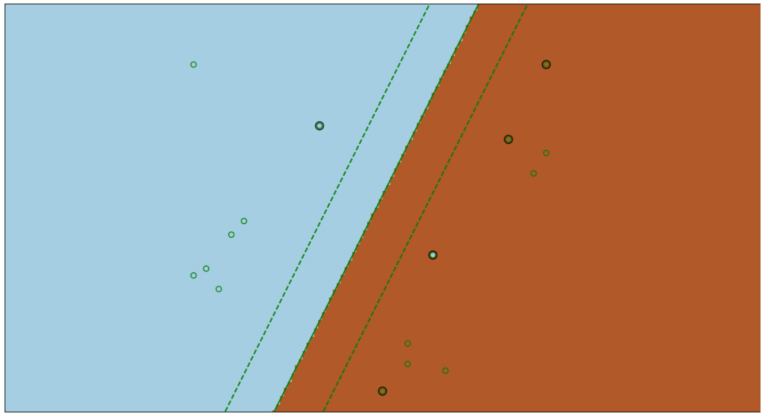
**Schölkopf, Bernhard, Alexander J. Smola, and Francis Bach. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

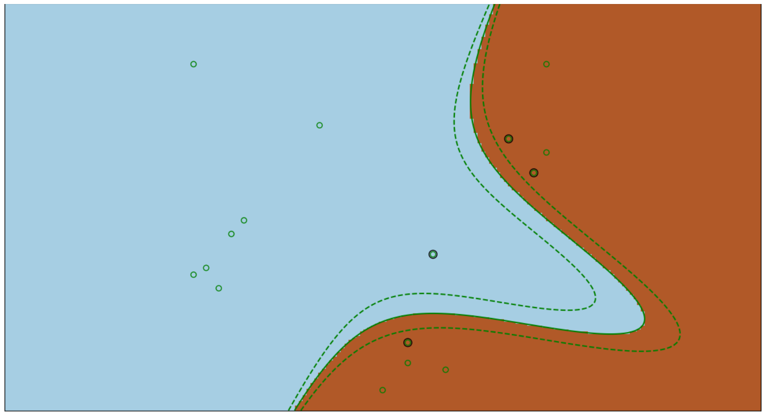
Kernels (Cont.)

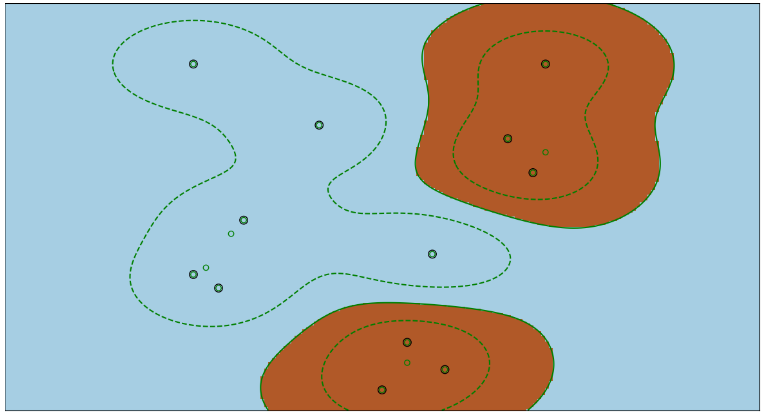
- The resulted matrix $\boldsymbol{K} \in \mathbb{R}^{N \times N}$ after applying $k(\cdot)$ is called kernel matrix.
- A kernel $k(\cdot)$ should be a symmetric and positive semidefinite so that every kernel matrix $\boldsymbol{K}$.
- Some popular kernels for SVM are the polynomial kernel, the Gaussian radial basis function (RBF) kernel, and the rational quadratic kernel.
- Important: Even if we use non-linear kernels, we are still optimising for the hyperplanes. This is still a linear problem. The non-linear features are only because of the kernels.

Kernels (Cont.)

- For non-linearly separable features, we need a kernel.


Linear kernel


Polynomial kernel


Gaussian radial basis function (RBF) kernel

Numerical Optimisation

- The primal SVM has variables with dimensions of D. While the dual SVM has variables with dimensions of the dataset size.
  - Both problems results in a convex quadratic programming problem. This is constrained optimisation and be solved with some existing solver.
- The constrained optimisation standard form can be defined as:
  - $\min f(x)$
  - Subject to $g_i(x) = c_i$ (equality constraint) and $h_j(x) \geq d_j$ (inequality constraint).
  - Brining the problem to this form allows us to use several tools for obtaining our solution.
- Recall the primal formulation:
  - $min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N} \xi_n$
  - Subject to $y_n\left(\langle \boldsymbol{w}, \boldsymbol{x}_n \rangle + b\right) \geq 1 - \xi_n$ and $\xi_n \geq 0$.
- We rearrange it in the following:
  - $min_{\boldsymbol{w},b} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{n=1}^{N} \xi_n$
  - Subject to $-y_n\boldsymbol{x}_n{}^T\boldsymbol{w} - y_n b - \xi_n \leq -1$ and $-\xi_n \leq 0$.
  - The optimization variables are all on the right side.
  - We have the standard form for constrained optimization.

Numerical Optimisation (Cont.)

- We can then put all the variables on a single vector and solve for the unknowns.
  - $min_{\boldsymbol{w},b,\xi} \frac{1}{2} \begin{bmatrix} \boldsymbol{w} \\ b \\ \xi \end{bmatrix}^T \begin{bmatrix} \boldsymbol{I}_D & \boldsymbol{0}_{D,N+1} \\ \boldsymbol{0}_{N+1,D} & \boldsymbol{0}_{N+1,N+1} \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ b \\ \xi \end{bmatrix} + \begin{bmatrix} \boldsymbol{0}_{D+1,1} & C\boldsymbol{1}_{N,1} \end{bmatrix}^T \begin{bmatrix} \boldsymbol{w} \\ b \\ \xi \end{bmatrix}$
  - Subject to $\begin{bmatrix} -\boldsymbol{YX} & -\boldsymbol{y} & -\boldsymbol{I}_N \\ \boldsymbol{0}_{N,D+1} & & -\boldsymbol{I}_N \end{bmatrix} \begin{bmatrix} \boldsymbol{w} \\ b \\ \xi \end{bmatrix} \leq \begin{bmatrix} -\boldsymbol{1}_{N,1} \\ \boldsymbol{0}_{N,1} \end{bmatrix}$
- $\boldsymbol{I}_m$ is the identity matrix, $\boldsymbol{0}_{m,n}$ matrix with zeros and $\boldsymbol{1}_{m,n}$ ones.
- Similar for the dual SVM, we can bring our problem to the standard optimisation form and solve for the unknowns.
- We can use any numerical optimisation technique to solve for the unknowns*.

*Nocedal, Jorge, and Wright, Stephen J. 2006. Numerical Optimization. Springer.

Study Material

- *Chapter 12, Deisenroth, Marc Peter, A. Aldo Faisal, and Cheng Soon Ong. Mathematics for machine learning. Cambridge University Press, 2020.*
- *Chapter 3, Cristianini, Nello, and John Shawe-Taylor. An introduction to support vector machines and other kernel-based learning methods. Cambridge university press, 2000.*

Next Lecture

Q&A