Freie Universität Berlin

Lecture: Machine Learning for Data Science

Winter semester 2021/22

Lectures 7: Classification (Support vector machines)

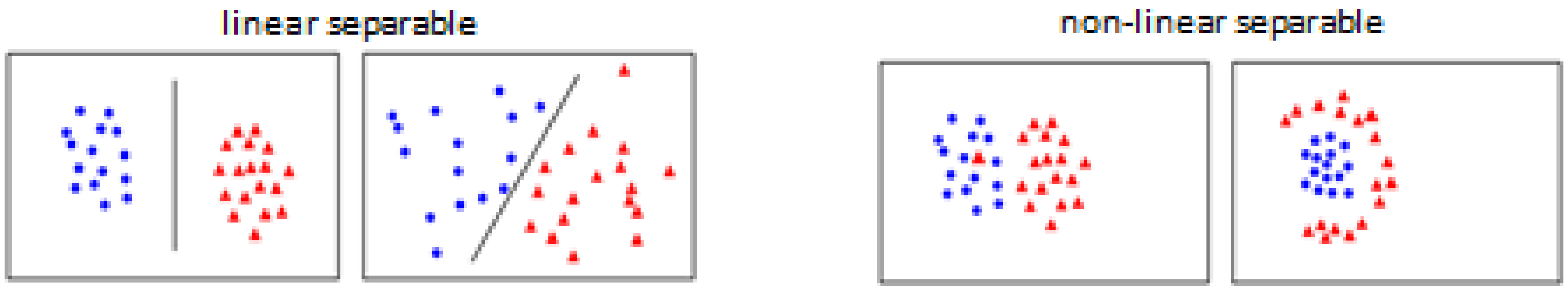
Prof. Dr. Eirini Ntoutsi

Outline

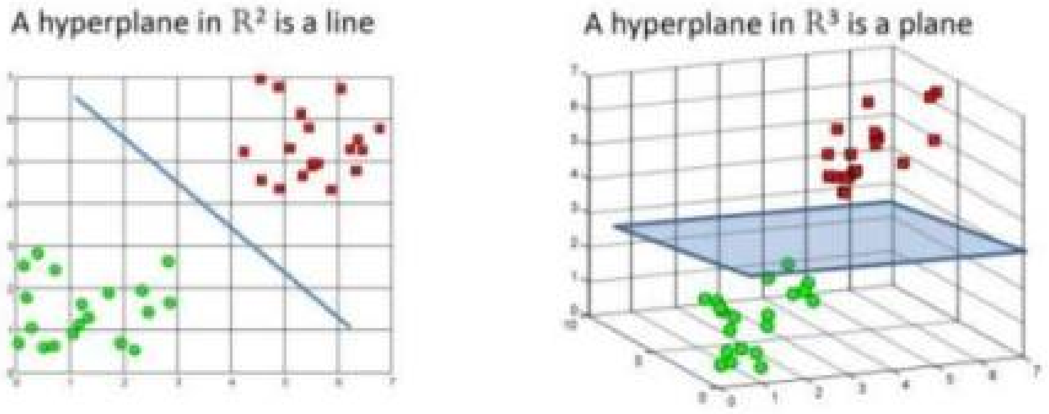
- Class of linear classifiers
- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

Basic idea

- Assumption: The classes are linearly separable



- Goal: find a classifier that will separate the data based on their class
  - In 2D, this is just a straight line
  - In higher dimensions, a hyperplane



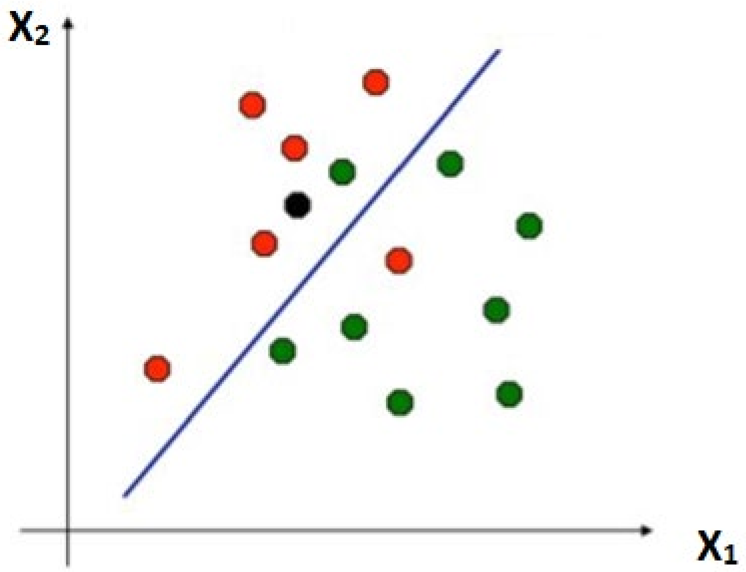
A hyperplane in $\mathbb{R}^n$ is an n-1 dimensional subspace

Linear classifiers

- Consider a simple binary classification problem.
  - Let training set, $D=\{(\vec{x}_i, y_i)\}$ and each instance is described in the d-dimensional feature space: $(X_1, X_2, \ldots, X_d)$
  - Let class Y = {-1,1}
- The decision function is linear in the features:

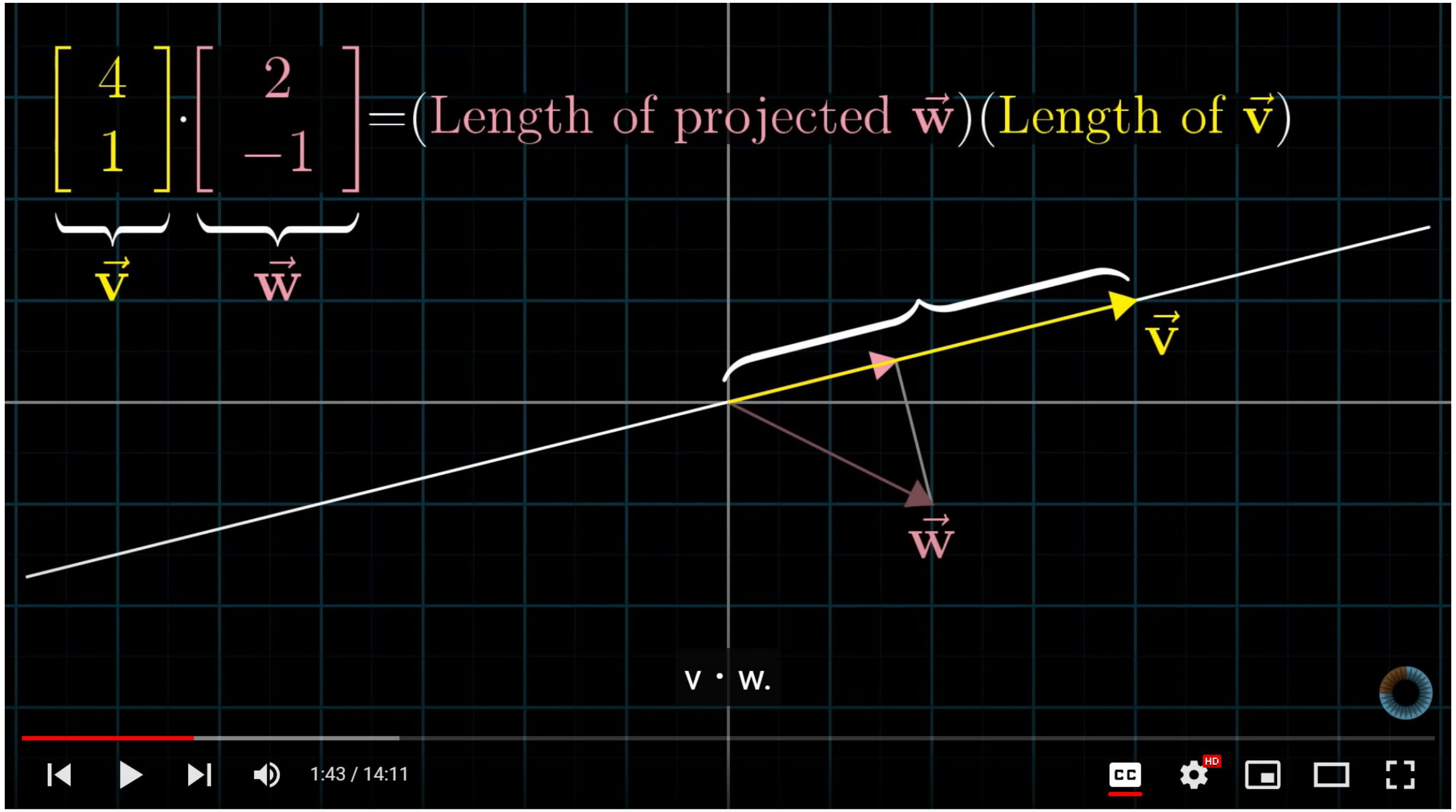
$$f(x) = w \cdot x + b = \sum_{1}^{d} w_i x_i + b$$

  - orientation is defined by *w*
    - *w* is a *d*-dimensional vector (the weight vector)
  - offset from origin is defined by *b* (bias)
    - *b* is a scalar
  - *w*•*x* is the dot product
  - *w* and *b* are the parameters of the model (to be learned)



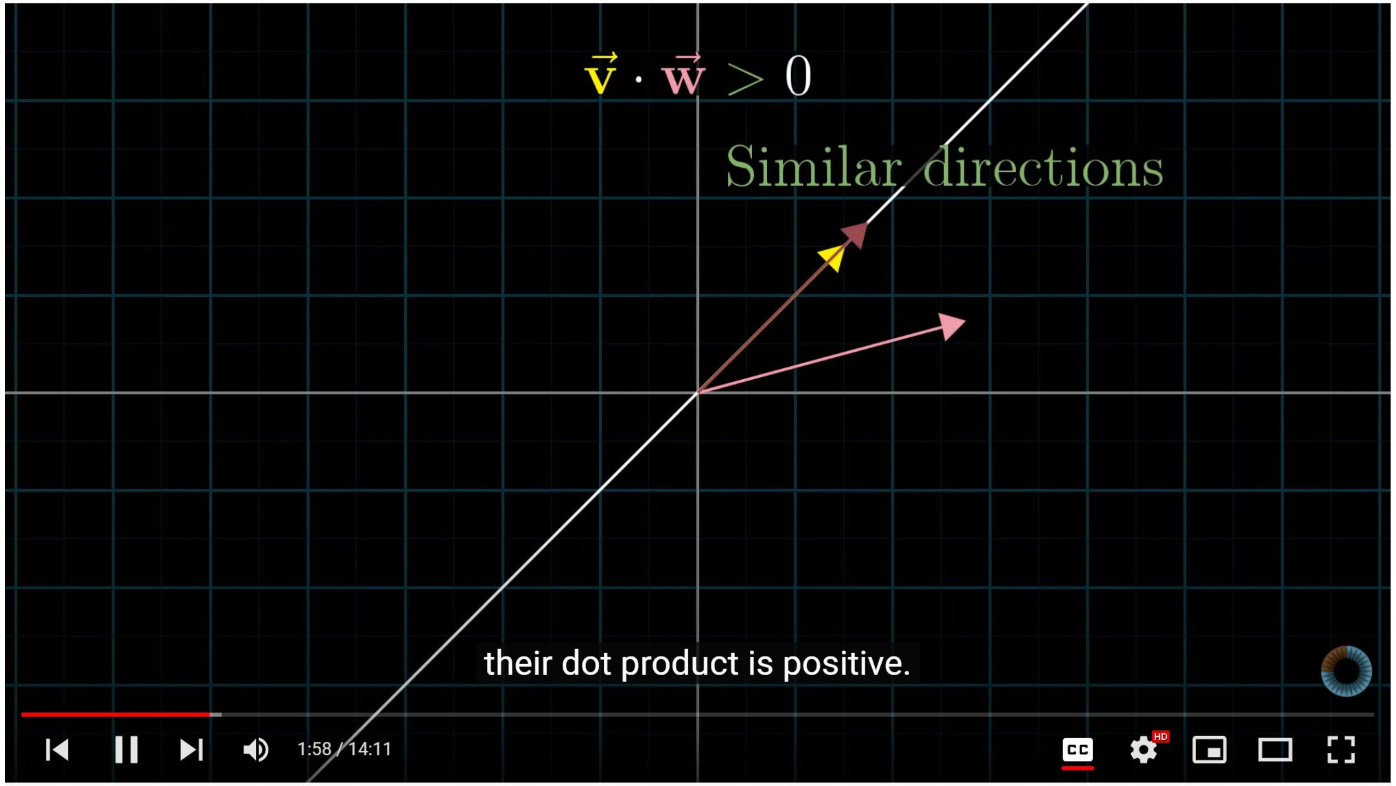
Linear classifiers

- More on dot product and its geometric interpretation [youtube link]



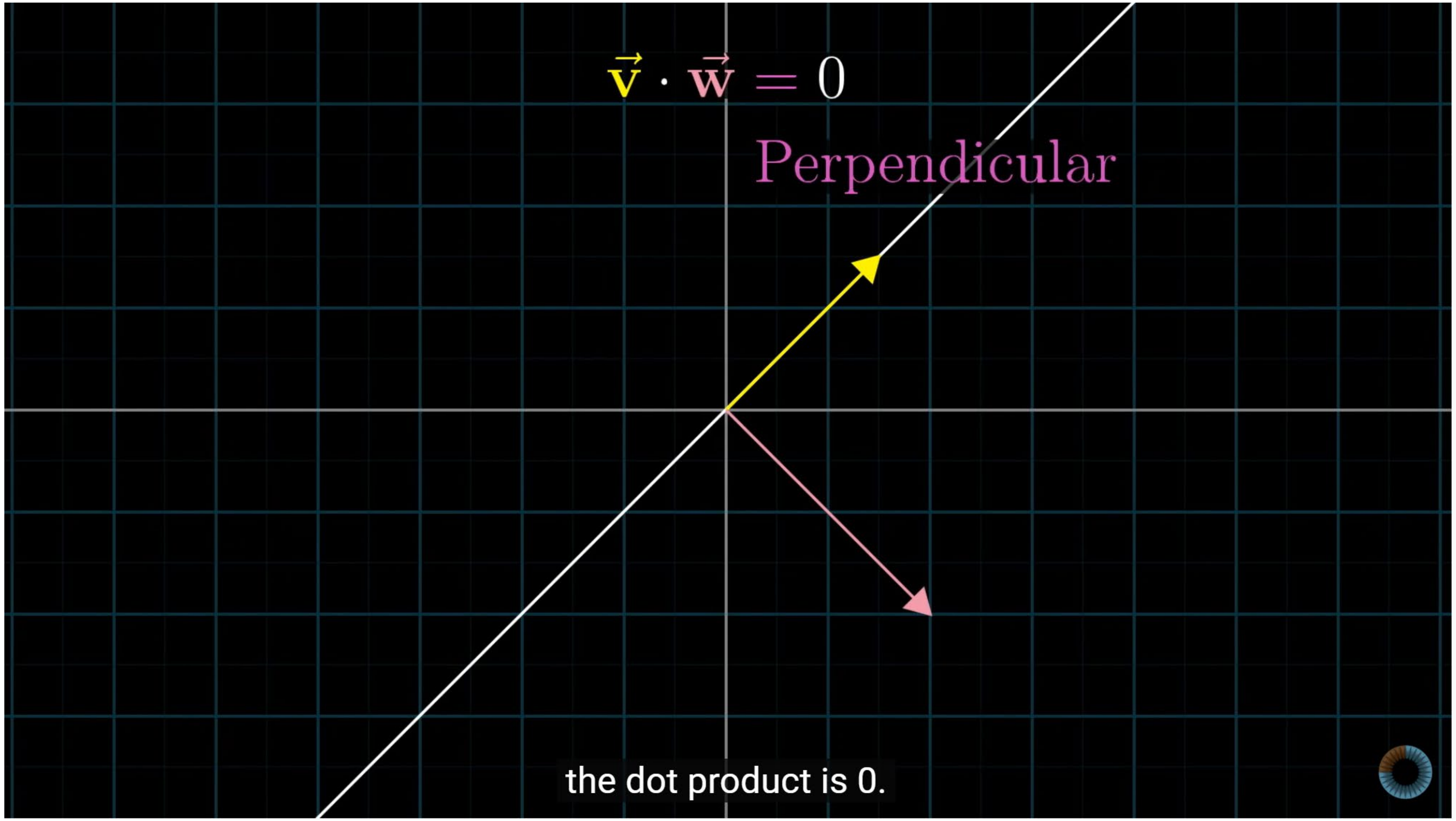
Linear classifiers

- More on dot product and its geometric interpretation [youtube link]



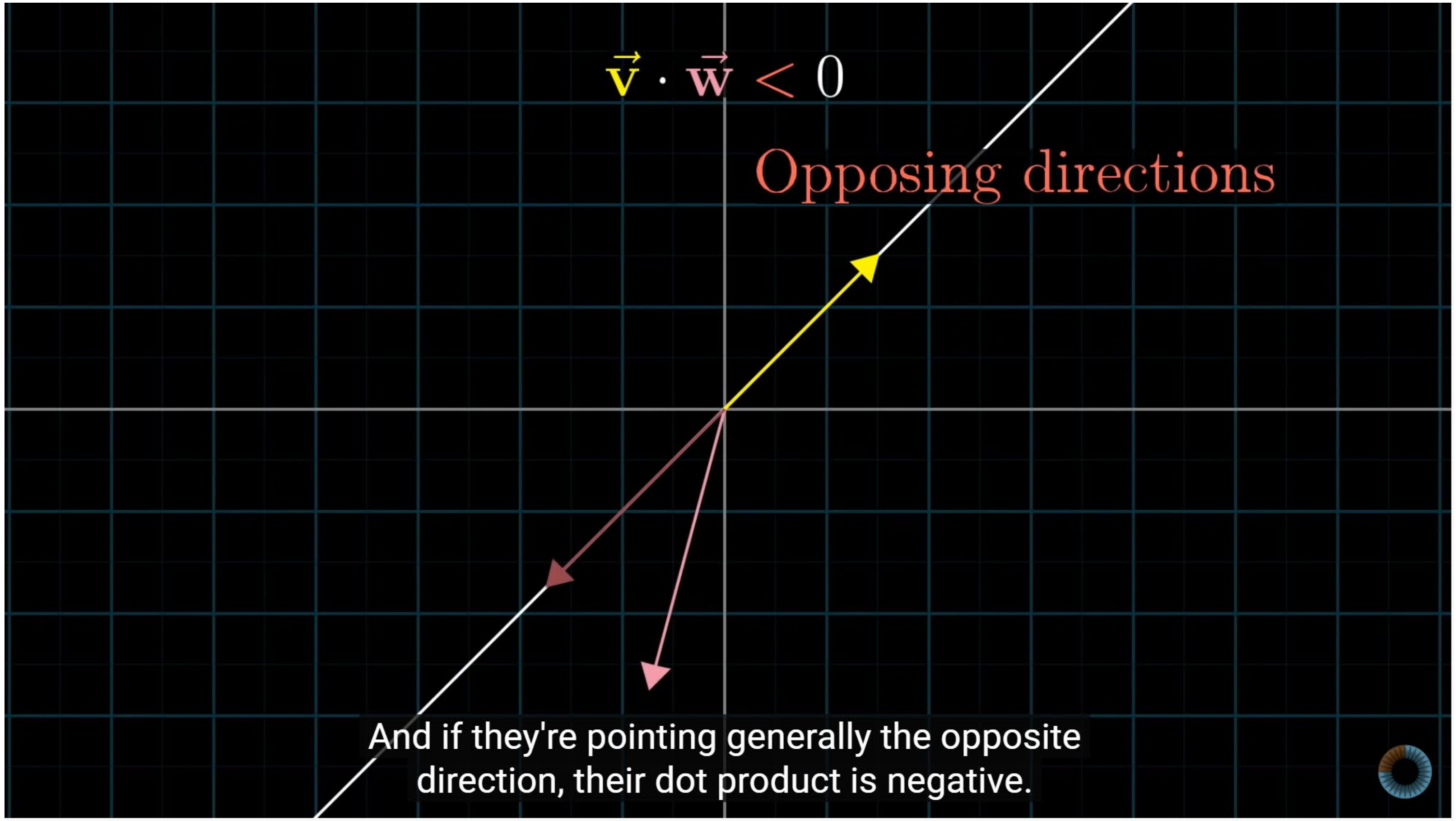
Linear classifiers

- More on dot product and its geometric interpretation [youtube link]



Linear classifiers

- More on dot product and its geometric interpretation [youtube link]



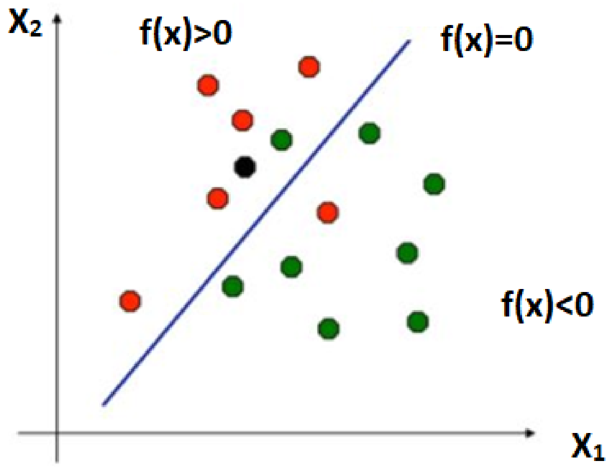
Linear classifiers

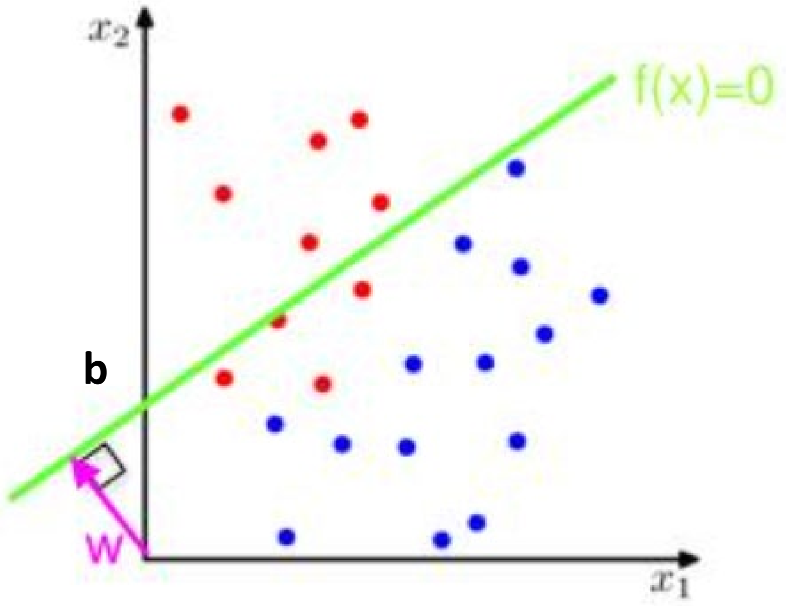
- Classification is based on the sign of *f(x)*

$$f(x) = \begin{cases} +1, if \sum_{1}^{d} w_i x_i + b > 0 \\ -1, if \sum_{1}^{d} w_i x_i + b < 0 \end{cases}$$

- The decision boundary is a (*d*-1)-dimensional hyper-plane orthogonal to *w* given by

$$f(x) = 0 \Rightarrow w \cdot x + b = 0 \Rightarrow \sum_{1}^{d} w_i x_i + b = 0$$



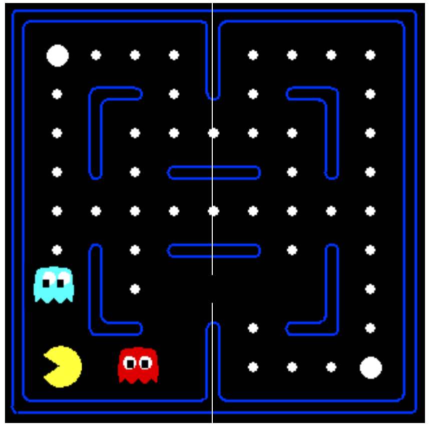


Class of linear classifiers

- The class of linear classifiers (also called, halfspaces) H is a class of functions of the form

$$f(x) = w \cdot x + b = \sum_{1}^{d} w_i x_i + b$$

- A very important class of predictors as the weights indicate the importance of the different features in the prediction
- Remember the Pacman example from the AI lecture
  - Instances refer to states and we could describe a state through different features
    - Distance to closest ghost → f_1
    - Distance to closest dot → f_2
    - Number of ghosts → f_3
    - ….
  - Goal was to find whether a state is good or not



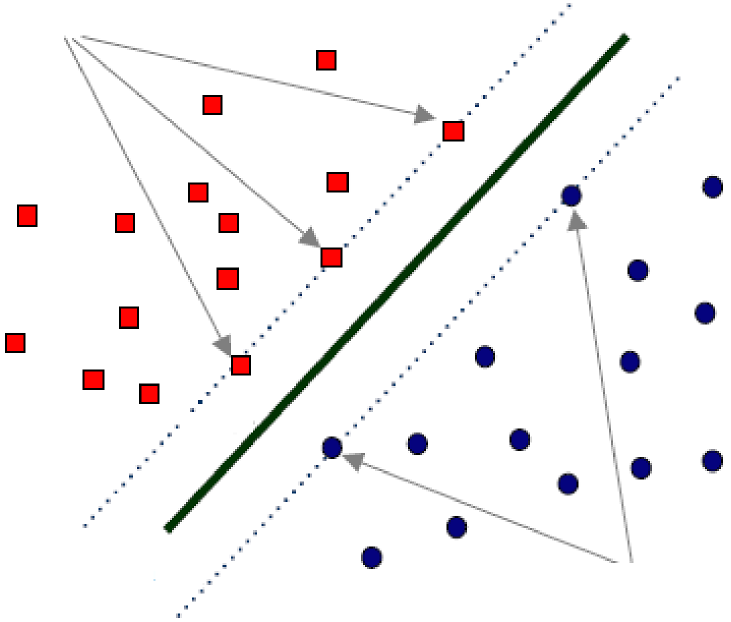
$$V(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s) \quad = \sum_{i=1}^{n} w_i f_i(s)$$

Outline

- Class of linear classifiers
- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

Support Vector Machines (SVM)

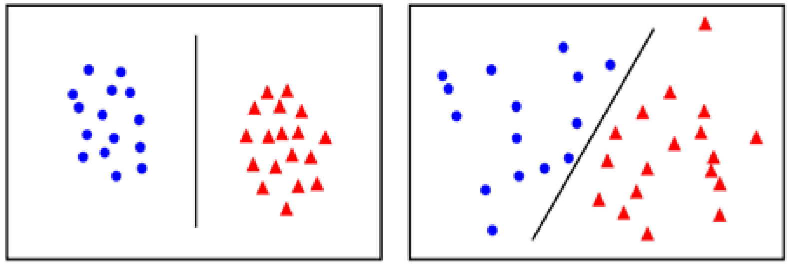
- A popular classification method
- Its roots are in statistical learning theory
- Promising results in many applications, e.g., handwritten text classification, text categorization
- The decision boundary is represented using a subset of the training instances, the so-called support vectors.



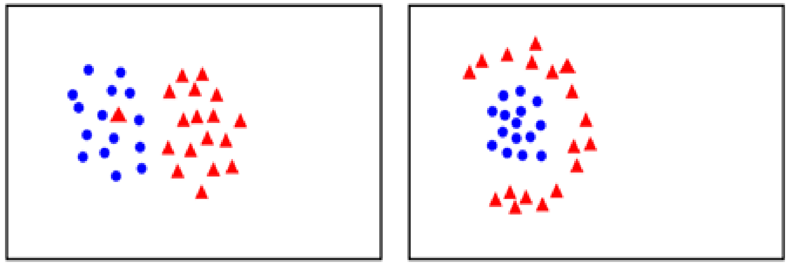
Basic idea: Class separation by hyperplanes

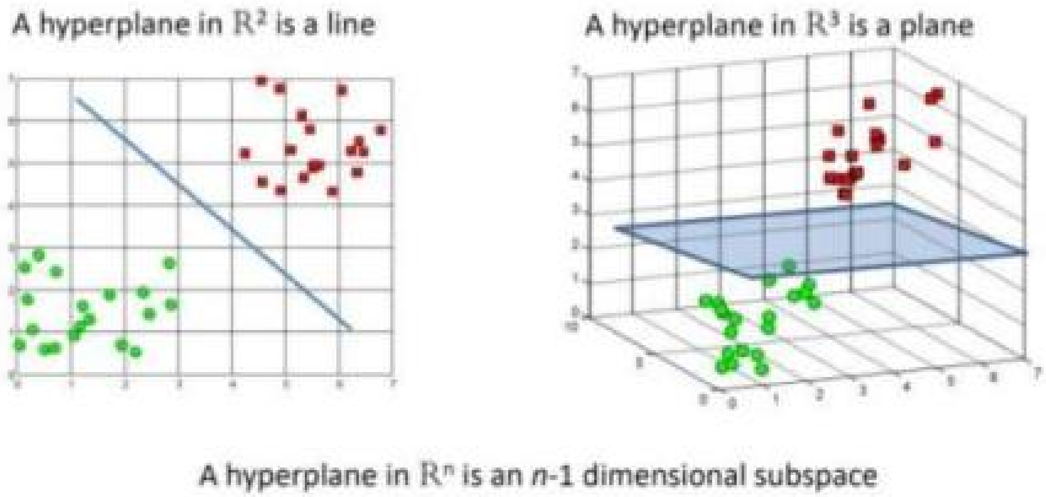
- Lets start with a simple 2 class problem and lets assume linear separability (we will relax this later)
- **Goal**: find a decision boundary (hyperplane) that will separate the data based on their class
  - In 2D, this is just a straight line
  - In higher dimensions, a hyperplane

linear separable



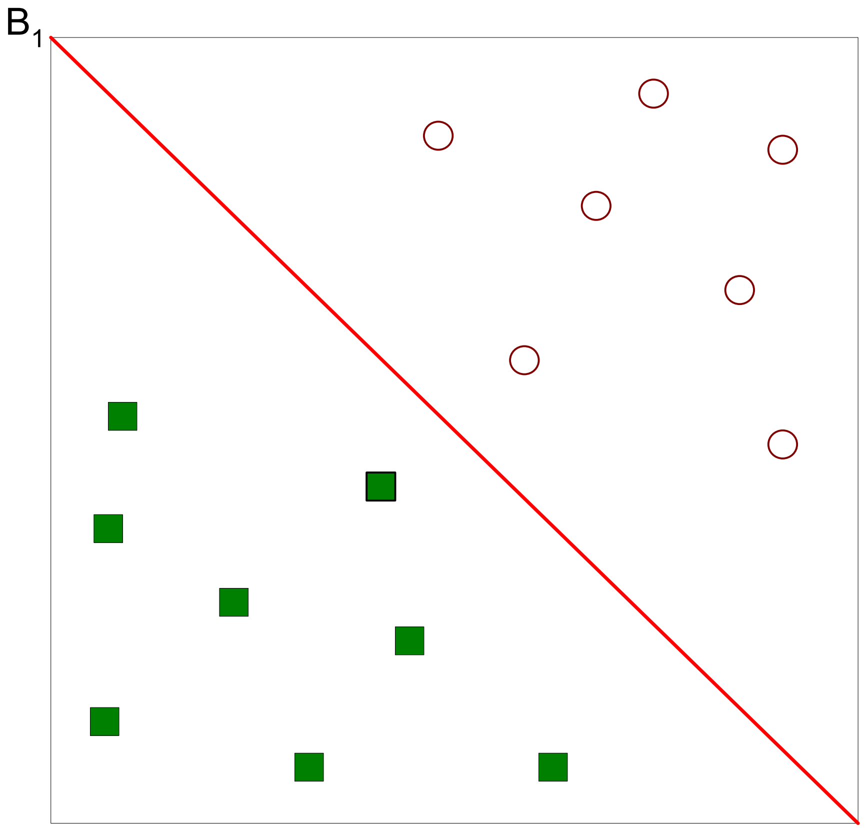
non-linear separable





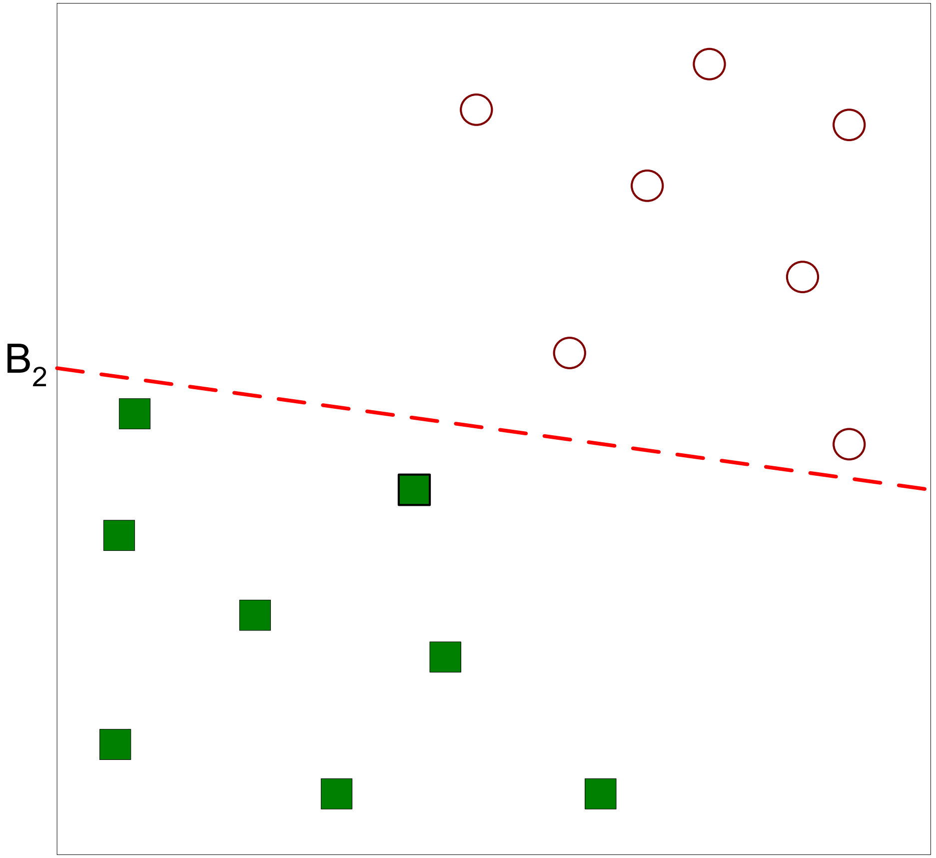
Finding a separating hyperplane

- Consider the following running example (2D for simplicity, so the hyperplane is a line)
- A possible solution is the following:



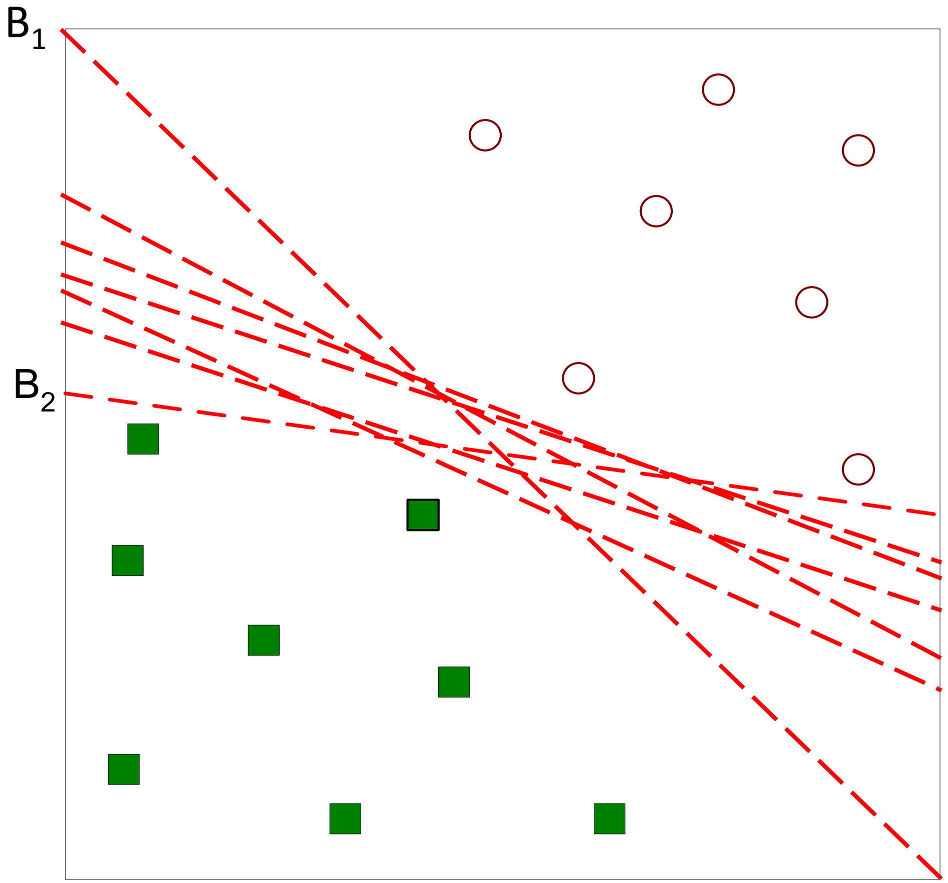
Finding a separating hyperplane

- Another possible solution



Finding a separating hyperplane

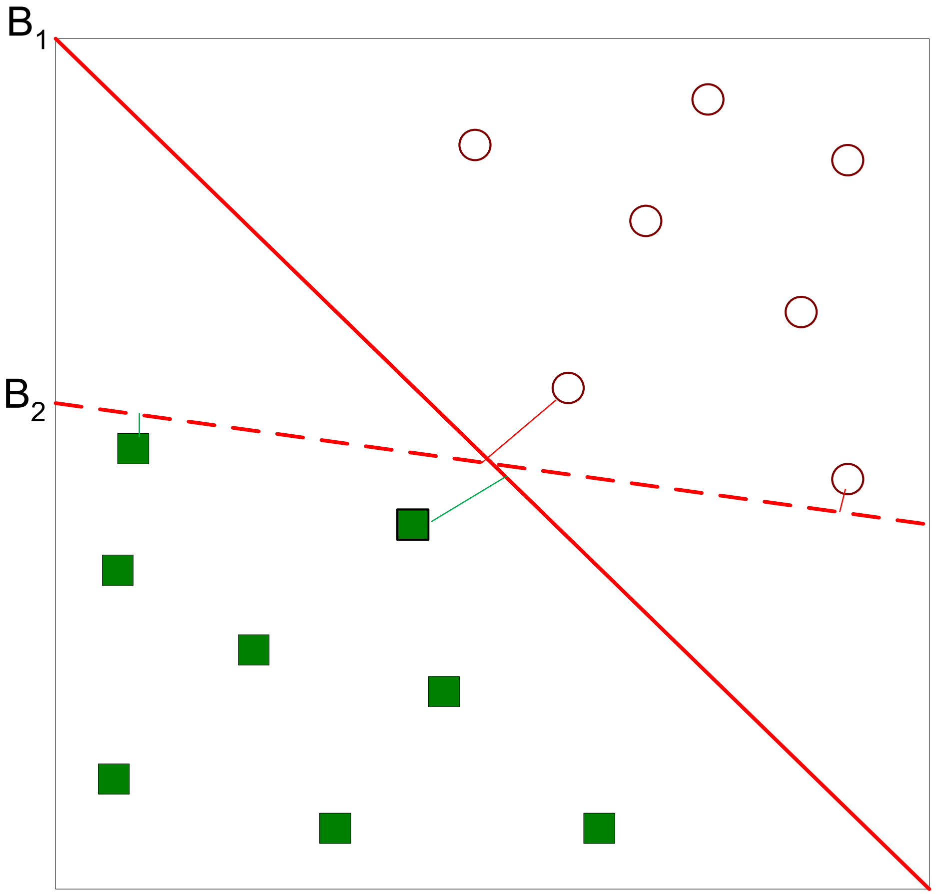
- Lots of possible solutions
- All with 0 error in the training data
- But we want to choose a hyperplane that is expected to work well on future unseen instances of the population.
  - i.e., a hyperplane that can generalize



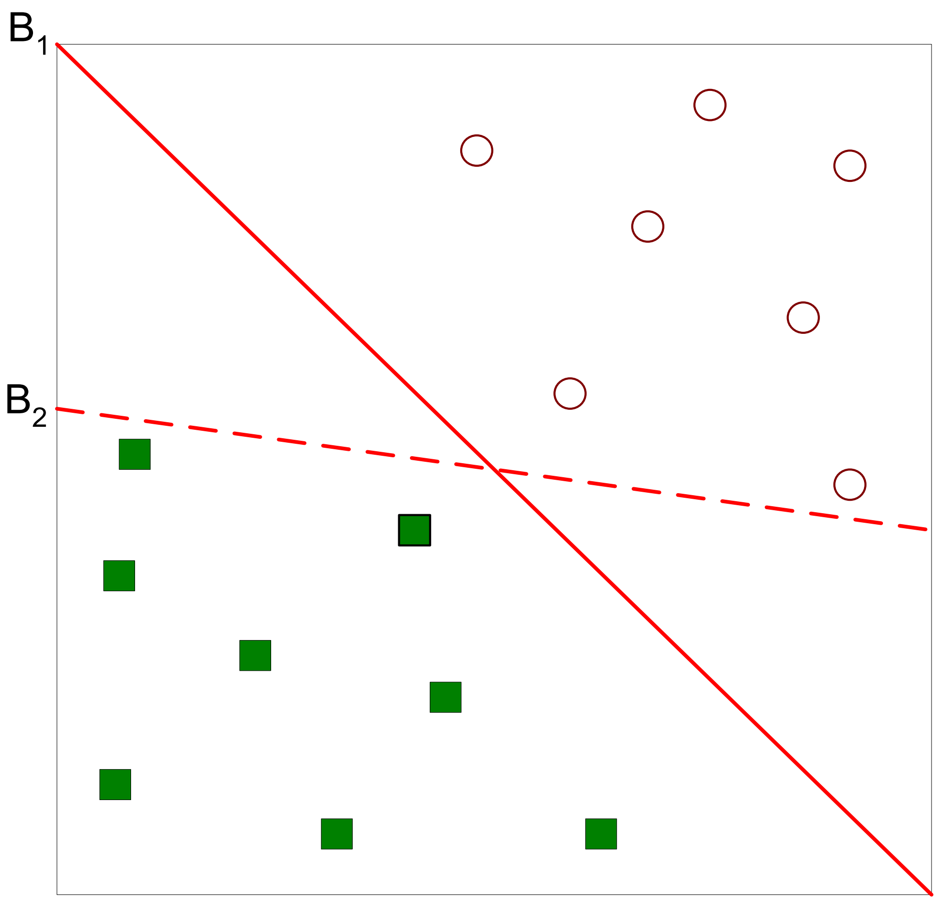
Choosing the best hyperplane

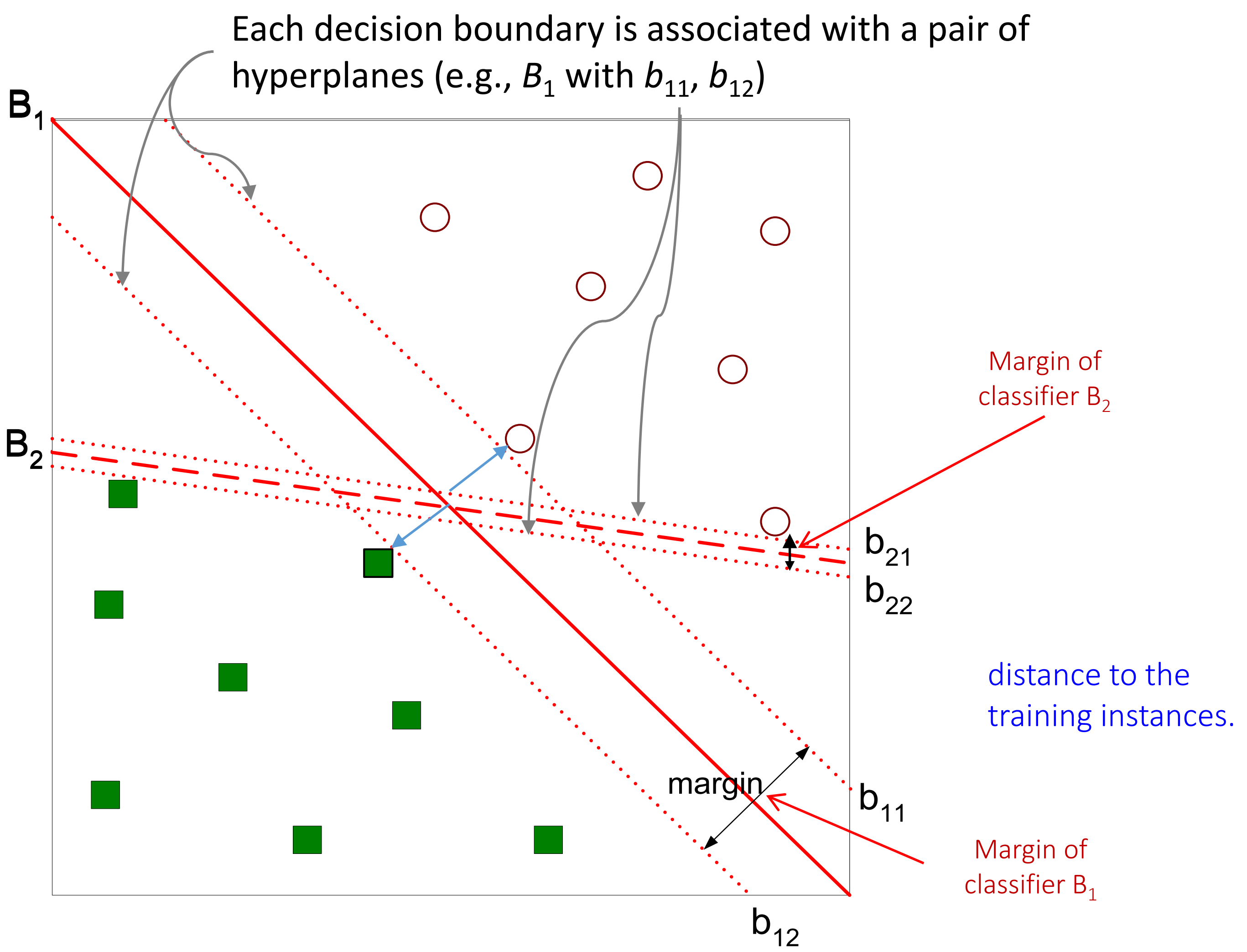
- Which hyperplane is better B_1 or B_2?

- How to define better? We need to evaluate somehow the hyperplanes/ lines.
  - Intuitively, a line is bad if it passes too close to the training instances
    - because it will be noise sensitive and will generalize poorly.
  - Therefore, our goal should be to find the line (hyperplane in general) that results in the largest minimum distance to the training instances.
  - Twice this distance is called margin.



Choosing the best hyperplane



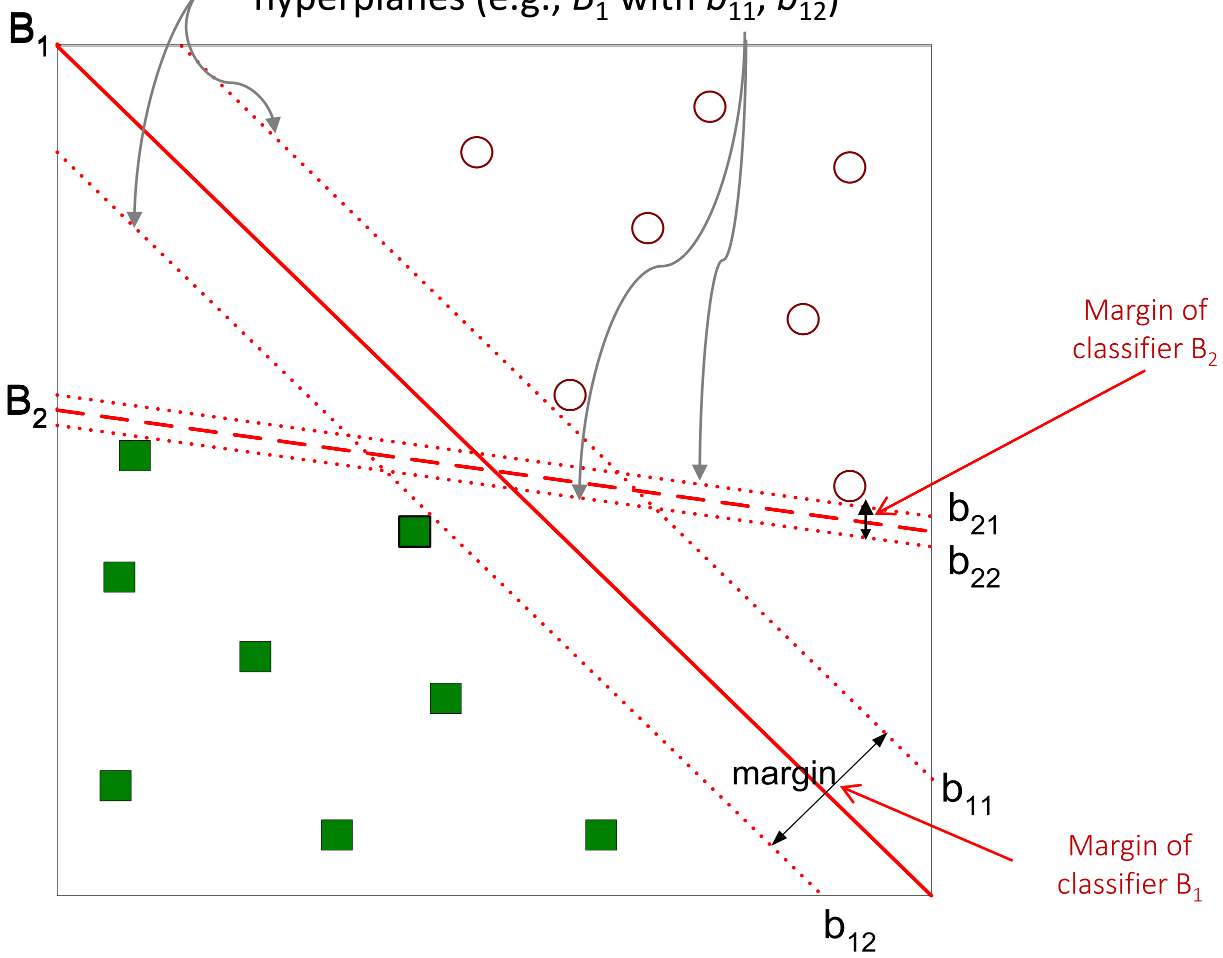


Choosing the best hyperplane

- Goal: Find the hyperplane that <u>maximizes</u> the margin (represents largest class separation)

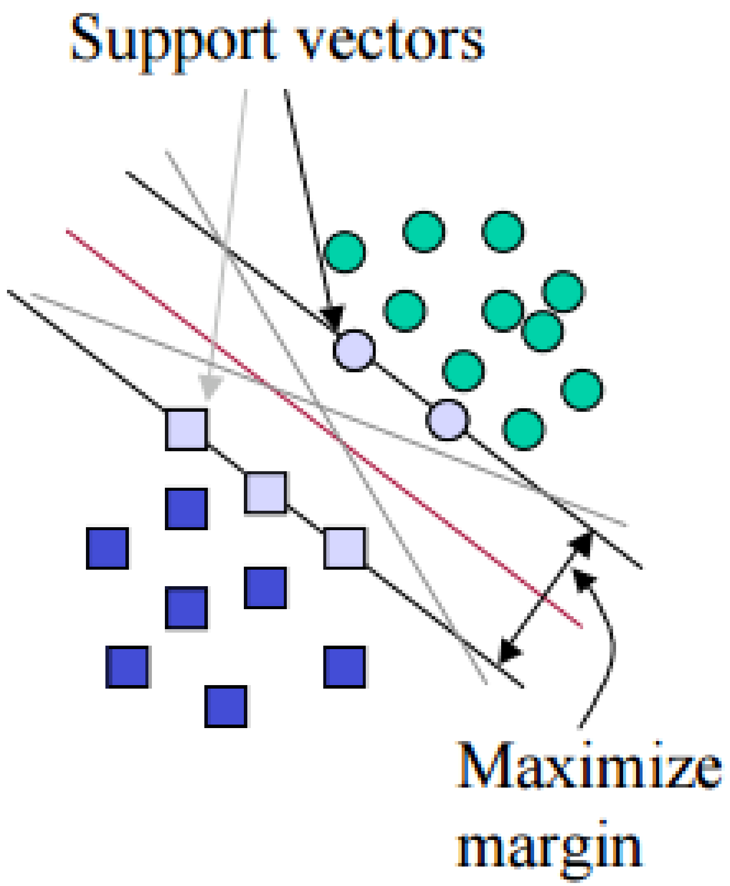
- In our example, which hyperplane is better B_1 or B_2?
  - B_1 is better than B_2

Each decision boundary is associated with a pair of hyperplanes (e.g., B_1 with b_{11}, b_{12})



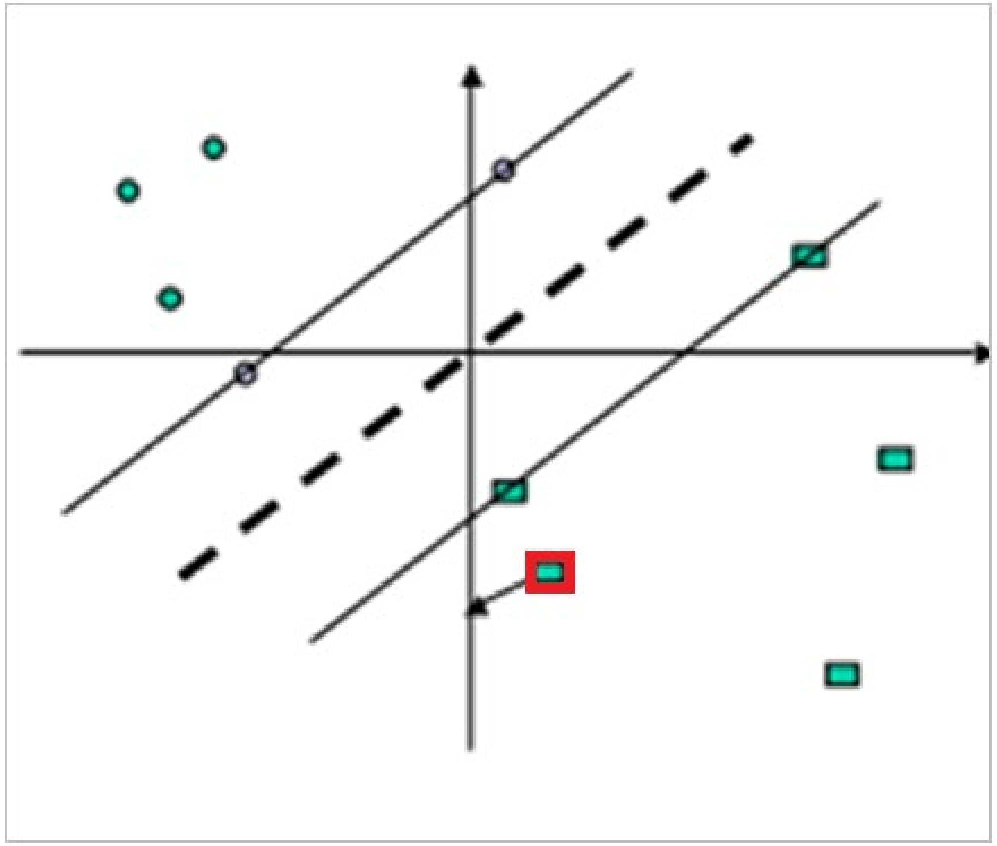
Support vectors

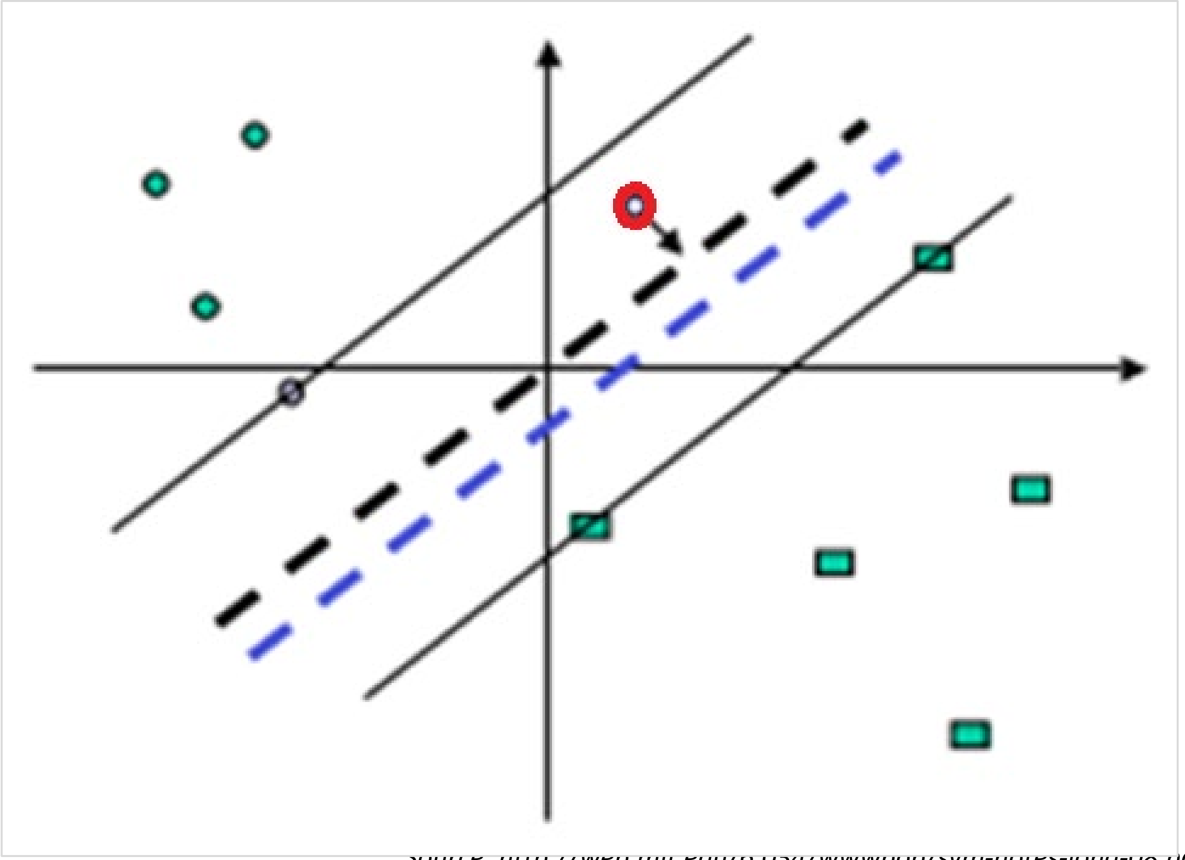
- SVMs maximize the margin (Winston terminology: the ‘street’) around the separating hyperplane
- The decision function is fully specified by a (usually very small) subset of training samples, the support vectors.
- Support vectors are the data points that lie closest to the decision boundary
  - They are the most difficult to classify
- Support vectors are critical elements
  - If removed, they would change the position of the dividing hyperplane



The criticality of support vectors

- Moving the non-support vectors has no effect on the decision boundary
- Moving a support vector moves the decision boundary





Source: http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf

Outline

- Class of linear classifiers
- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

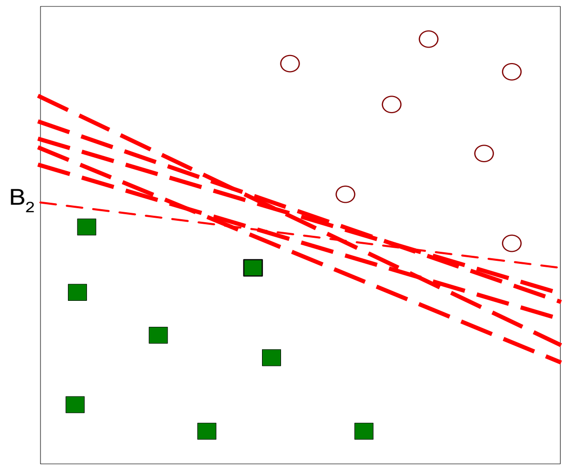
Linear SVM

- A linear SVM searches for a hyperplane that maximizes the margin (maximal margin classifier)
- Consider a simple binary classification problem. Let training set, $D=\{(\vec{x_i}, y_i)\}$ and
  - Each instance is described in the d-dimensional feature space: $(X_1, X_2, \ldots, X_d)$
  - Let class Y = {-1,1}

- The decision boundary of a linear classifier can be written as follows:

$$w \bullet x_i + b = 0$$

  - w is a weight vector
  - x_i is the input vector (instance)
  - b is a scalar (bias)
  - w and b are the parameters of the model



Linear SVM

- The linear classifier is the hyperplane *H* characterized by parameters *w* and *b*

$$w \bullet x_i + b = 0$$

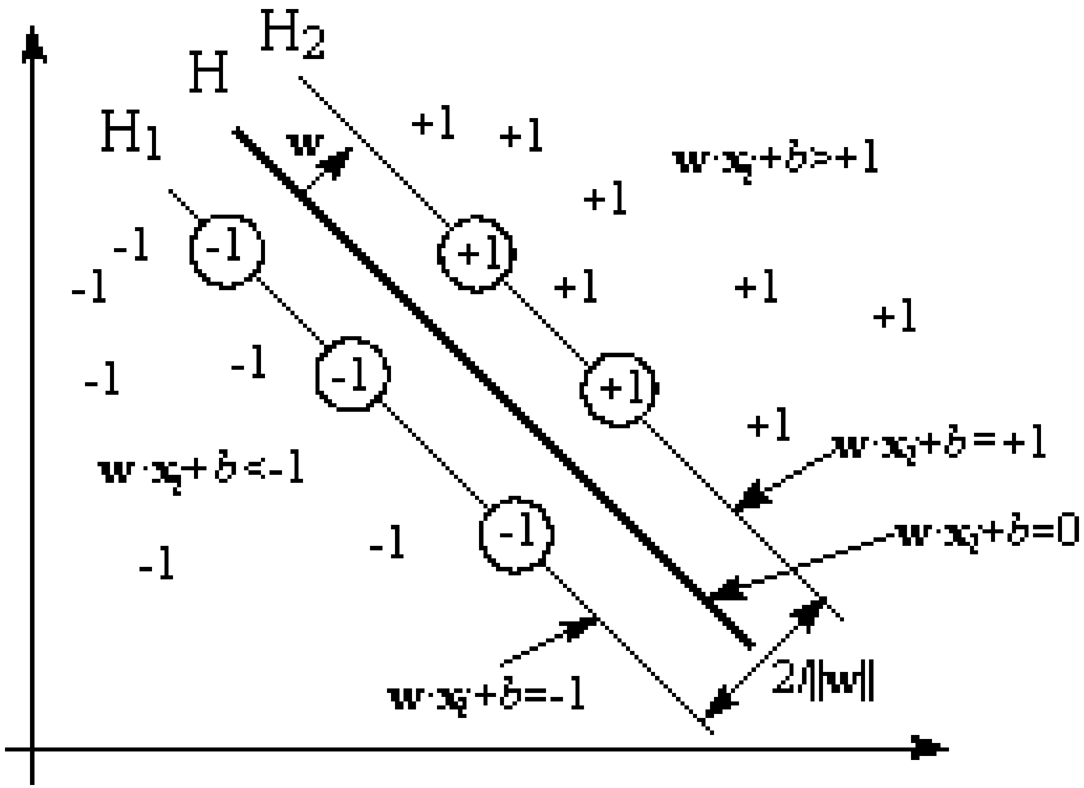
- For any instance x_i in the training set, the decision boundary *H* must satisfy the following inequalities:

$$w \bullet x_i + b \geq +1 \text{ when } y_i = +1$$
$$w \bullet x_i + b \leq -1 \text{ when } y_i = -1$$

- For the associated hyperplanes H_1 and H_2 it holds:

$$H_2 : w \bullet x_i + b = +1$$
$$H_1 : w \bullet x_i + b = -1$$

  - The points on the planes H_1, H_2 are the support vectors

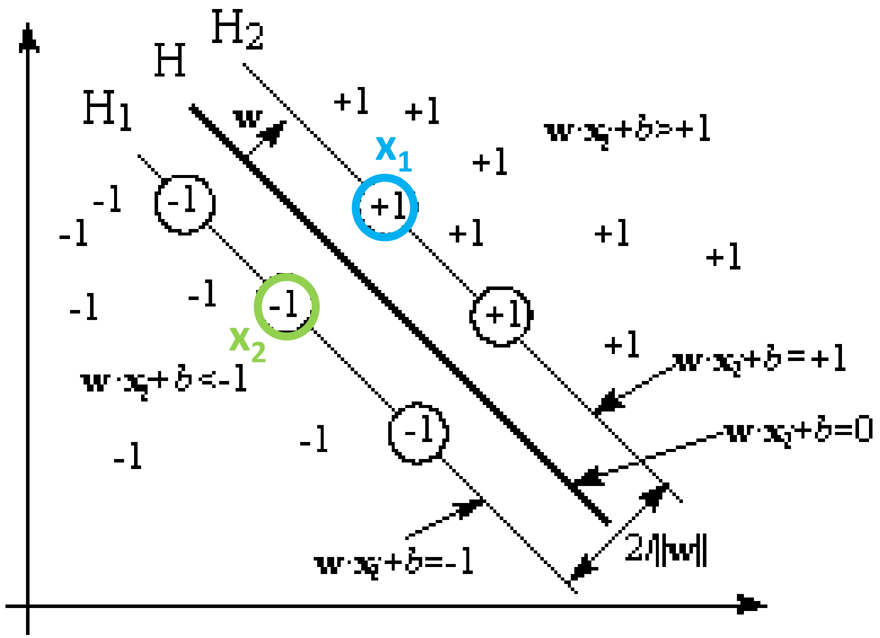


Source: http://www.support-vector-machines.org/SVM_osh.html

- The aforementioned conditions are also applicable to *any* linear classifiers, SVMs imposes an *additional* requirement regarding the margin

Linear SVM

- The margin of hyperplane H is given by the distance between the two hyperplanes H_1,H_2.
- Let x_1, x_2 be two points in H_2, H_1 respectively.
  - $w \cdot x_1 + b = +1$
  - $w \cdot x_2 + b = -1$
- From geometry, distance of point x_i from hyperplane: $\frac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$
- Distance of x_1 from H: $= \frac{1}{\|w\|}$
- Distance of x_2 from H: $= \frac{1}{\|w\|}$



Source: http://www.support-vector-machines.org/SVM_osh.html

- So the margin which we want to maximize is given by
  - $Margin\ d = \frac{1}{\|w\|} + \frac{1}{\|w\|} = \frac{2}{\|w\|}$

$\| \boldsymbol{w} \| = \sqrt{\boldsymbol{w} \cdot \boldsymbol{w}}$ is the Euclidean norm/length:

Linear SVM

- So, we want to maximize

  Margin $d = \frac{2}{\|w\|}$

- In order to maximize *d*, we need to minimize $\| w \|$
  - actually, we minimize the following:

    $$\min_{w} \frac{\|\boldsymbol{w}\|}{2} \Leftrightarrow \min_{w} \frac{\|w\|^2}{2}$$

  - but, subject to the following constraints

//This allows us to perform quadratic programming optimization latter on

$$y_i = \begin{cases} +1, & if\ \boldsymbol{w} \cdot \boldsymbol{x_i} + b \geq +1 \\ -1, & if\ \boldsymbol{w} \cdot \boldsymbol{x_i} + b \leq -1 \end{cases}$$

(i.e., there are no *data points between H_1 and H_2)*

or, alternatively $y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$

Linear SVM

Definition: (Hard Margin) Linear SVM: Separable case

$$\min_{w} \frac{\|\boldsymbol{w}\|^2}{2}$$

subject to

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1, x_i \in D$$

- This is a constrained optimization problem
  - The objective function is quadratic
  - The constraints are linear on model parameters *w, b*
  - This is convex optimization problem, which can be solved using the [Lagrange multiplier method]
    - A technique that lets you find the maximum or minimum of a multivariable function when there is some constraint on the input values you are allowed to use.

Linear SVM

- Goal: Minimize $\min_{w} \frac{\|\boldsymbol{w}\|^2}{2}$ subject to $y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$
- Introduce Langrange multipliers $a_i \geq 0\ \forall i$; there is an a_i for each training instance
- Define the auxiliary objective function L_P

$$L_P = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i (y_i(\boldsymbol{x}_i . \boldsymbol{w} + b) - 1)$$

- Minimize L_P w.r.t. w, b.

$$\frac{\partial L}{\partial \boldsymbol{w}} = \left(\boldsymbol{w} - \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \right) = 0$$

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0$$

- Points with $\alpha_i > 0$ are called support vectors and lie on H_1 or H_2.
- Points with $\alpha_i = 0$ lie beyond margin planes and are irrelevant to the solution.

Linear SVM

- The solution (trained SVM) consists of:
  - The support vectors x_i (those instances with $\alpha_i > 0$)
- Based on which, we can also compute the parameters *w, b* of the decision boundary
- The plane is a linear combination of the training vectors:

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i$$

- *b* can be computed using the plane equation for each support vector:

$$f(x_i) = wx_i + b \Rightarrow b = f(x_i) - wx_i$$

The support vectors

x_1	x_2	y	Lagrange Multiplier
0.3858	0.4687	1	65.5261
0.4871	0.611	−1	65.5261
0.9218	0.4103	−1	0
0.7382	0.8936	−1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	−1	0
0.2146	0.0099	1	0

Linear SVM – an example of computing model parameters

- Find a linear SVM for the given 2D dataset:
  - After we find the support vectors α_i:
  - We compute the parameters of the plane:

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i$$

  - Computing w:

$$w_1 = \sum_{i=1}^{N} \alpha_i y_i x_{i1} = 65.5621 * 1 * 0{,}3858 + 65.5621 * -1 * 0{,}4871 = -6{,}64$$

$$w_2 = \sum_{i=1}^{N} \alpha_i y_i x_{i2} = 65.5621 * 1 * 0{,}4687 + 65.5621 * -1 * 0{,}611 = -9{,}32$$

  - Computing b:
    - $f(x_i) = wx_i + b$ ➔ $b = f(x_i) - wx_i$

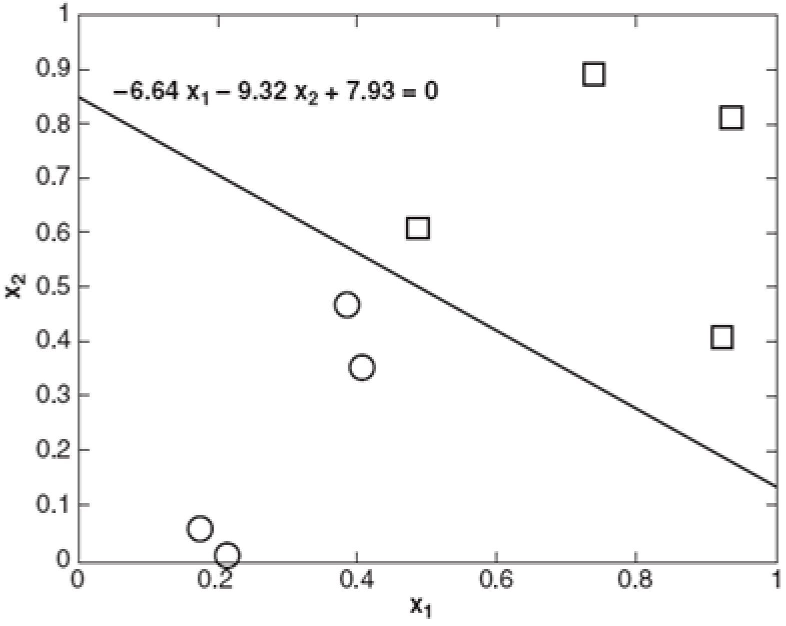
$$b^{(1)} = 1 - wx_1 = 1 - (-6{,}64)(0{,}3858) - (-9{,}32)(0{,}4687)$$

$$b^{(2)} = -1 - wx_2 = \ldots$$

$$b = avg(b^{(1)}, b^{(2)})$$

The support vectors

x_1	x_2	y	Lagrange Multiplier
0.3858	0.4687	1	65.5261
0.4871	0.611	−1	65.5261
0.9218	0.4103	−1	0
0.7382	0.8936	−1	0
0.1763	0.0579	1	0
0.4057	0.3529	1	0
0.9355	0.8132	−1	0
0.2146	0.0099	1	0



Linear SVM – Classifying new instances

- How can we classify a new instance x_j?
- Let $Ns \subseteq D$ be the set of support vectors, with $s_i = x_i$

Note that the decision relies on the inner product between x_j and the support vectors $s_i = x_i$

- The classification of a new point x_j is given by the sign of:

$$\left.\begin{aligned} f(x_j) &= sign(wx_j + b) \\ \mathbf{w} &= \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \end{aligned}\right\} \quad f(\mathbf{x}_j) = sign\left(\sum_{i=1}^{N_s} \alpha_i y_i \mathbf{s}_i . \mathbf{x}_j + b \right)$$

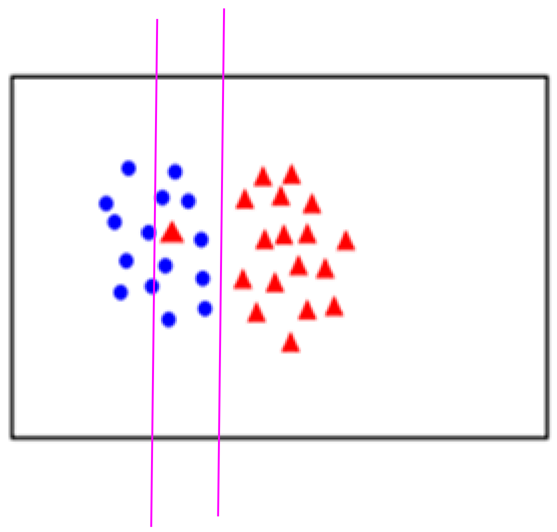
  - a_i: Lagrange multipliers
  - s_i: is the support vector
  - y_i: is the class of s_i

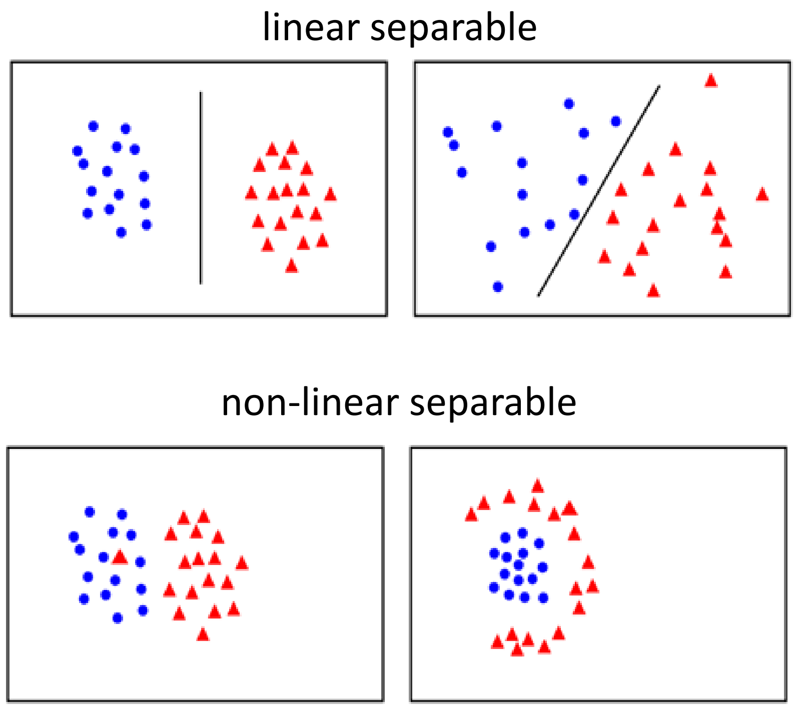
Outline

- Class of linear classifiers
- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

Linear SVM nonseparable cases

- What if the problem is not linearly separable?
  - There is no hyperplane that makes no mistakes on training data



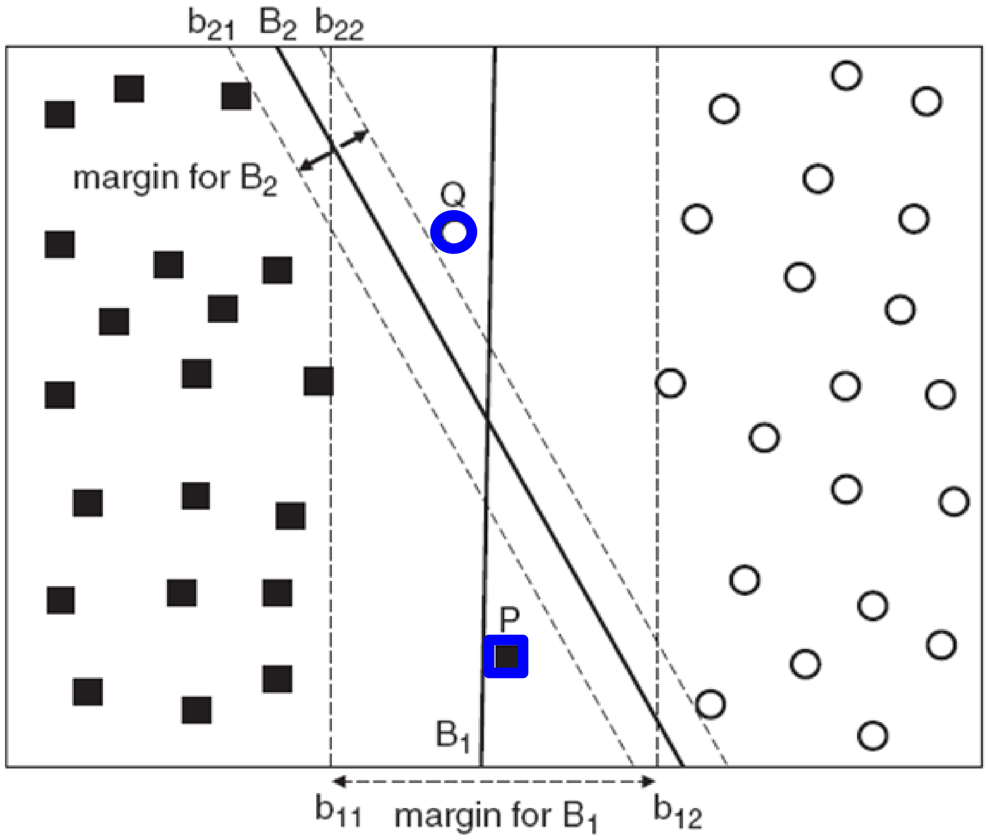


Even in linear separable cases

Which hyperplane is better B_1 or B_2?

- We might prefer a solution that better separates the bulk of the data while ignoring a few weird noise points.
- B_1 should be preferred over B_2
  - it has a wider margin → less susceptible to overfitting
- but, the so far SVM formulation is error free

  → Soft margin approach
  - Allows the margin to make some mistakes



Soft margin approach

- Learn a decision boundary that is tolerable to small training errors
- Allows SVM to construct a decision boundary even in cases where the classes are not linearly separable
- Idea: trade-off between the width of the margin and the misclassification errors committed by the linear decision boundary.

Original optimization problem

$$\min_{w} \frac{\|\boldsymbol{w}\|2}{2}$$

subject to

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$$

violated

Idea:

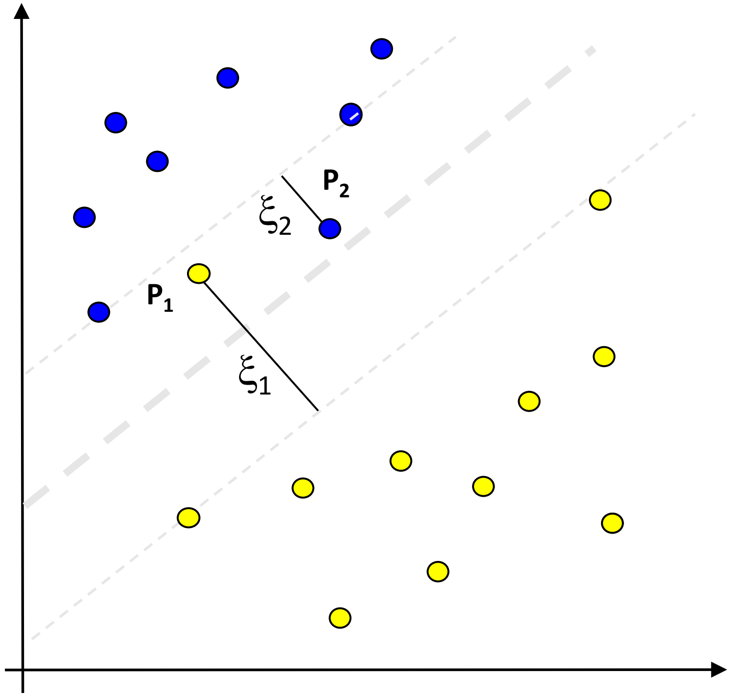
- Relax the constraints to accommodate small training errors
- Introduce positive-valued slack variables ξ_i

Soft margin approach

- Relaxing by introducing slack variables ξ_i, $\xi_i \geq 0$

$$y_i=\begin{cases} +1, & if\ \boldsymbol{w}\cdot\boldsymbol{x_i}+b \geq +1 \\ -1, & if\ \boldsymbol{w}\cdot\boldsymbol{x_i}+b \leq -1 \end{cases} \Rightarrow y_i=\begin{cases} +1, & if\ \boldsymbol{w}\cdot\boldsymbol{x_i}+b \geq +1-\xi_i \\ -1, & if\ \boldsymbol{w}\cdot\boldsymbol{x_i}+b \leq -1+\xi_i \end{cases}$$

- The slack variable ξ_i provide an estimate of the error of the decision boundary on training instance x_i
- Intuitively, data points on the incorrect side of the margin boundary have a penalty that increases with the distance from it.
  - This allows x_i to not meet the margin requirement at a cost proportional to the value of ξ_i.
- Goal: find a line that penalize points in the "wrong side".



Soft margin approach

Original optimization problem

$$\min_{w} \frac{\|w\|^2}{2}$$

subject to

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x_i} + b) \geq 1$$

- Definition: Soft margin linear SVM

- Need to minimize:

$$\min_{w} \frac{\|w\|^2}{2} + C\left(\sum_{i=1}^{n} \xi_i^k\right)$$

subject to the following constraints:

$$y_i = \begin{cases} +1, & if\ \boldsymbol{w} \cdot \boldsymbol{x_i} + b \geq +1 - \xi_i \\ -1, & if\ \boldsymbol{w} \cdot \boldsymbol{x_i} + b \leq -1 + \xi_i \end{cases}$$

If no constrains on # mistakes, we might end up with a very wide margin with many misclassification errors

C, *k* are user-specified parameters representing the penalty of misclassifying the training instances

Relaxed constraints

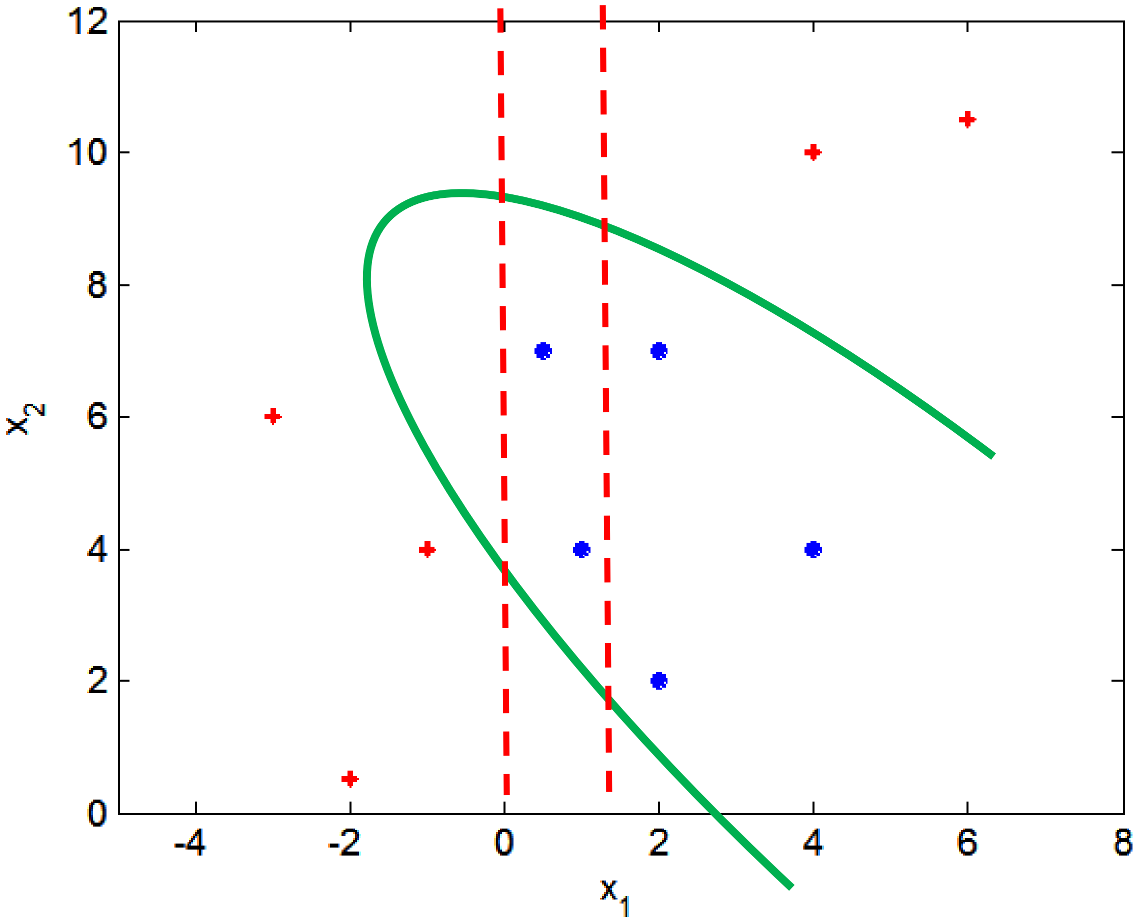
- Can be solved used quadratic programming
  - This way we can learn the parameters *w, b* of the decision boundary

Outline

- Class of linear classifiers
- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

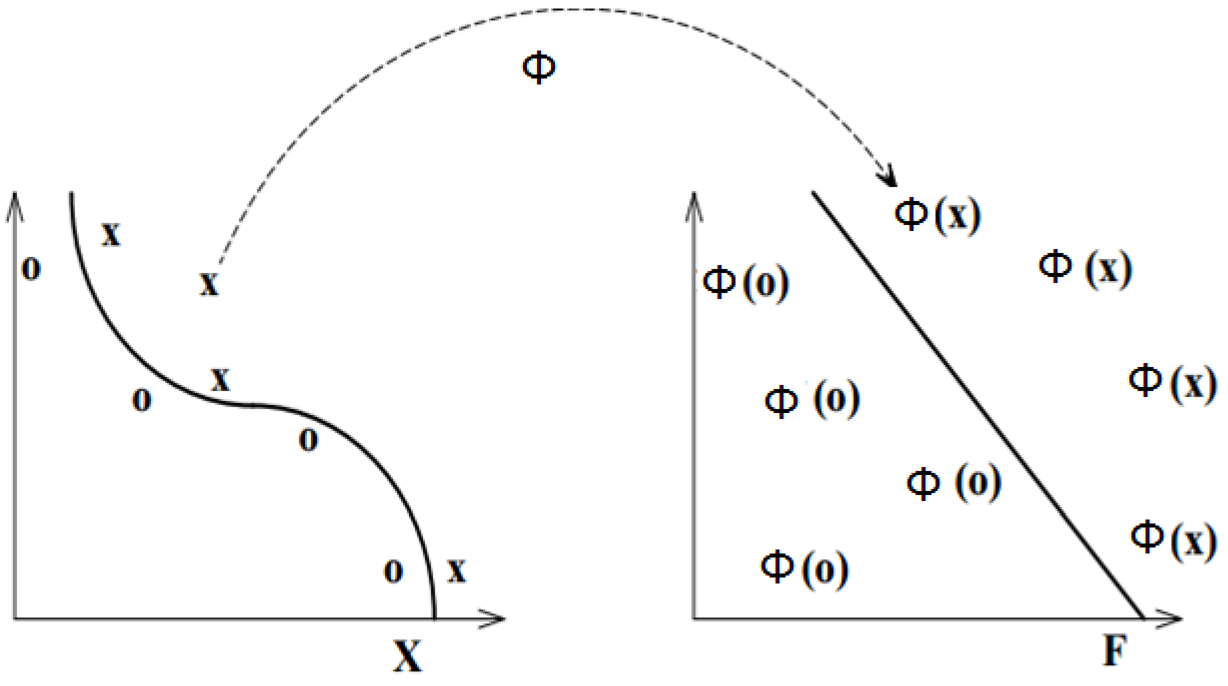
Nonlinear SVM

- What if the decision boundary is not linear?



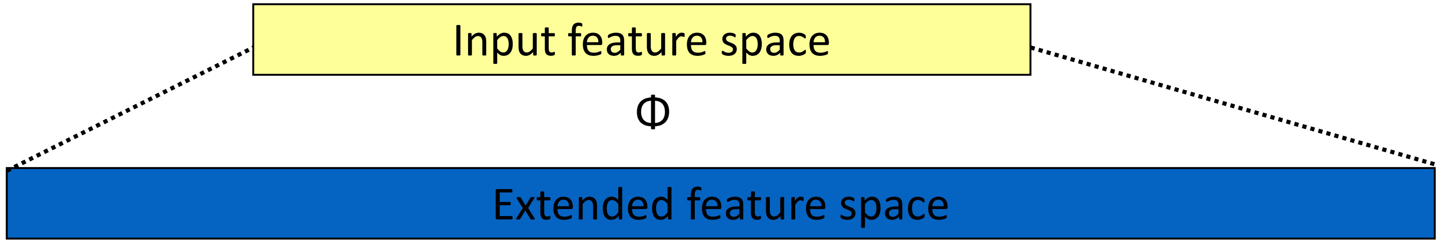
Transformation to separate

- Trick: transform the data from its original space X into a new space $\Phi(X)$ so that a linear decision boundary can be used to separate the instances in the transformed space
- In $\Phi(X)$, we can apply the same methodology as before to find a linear decision boundary

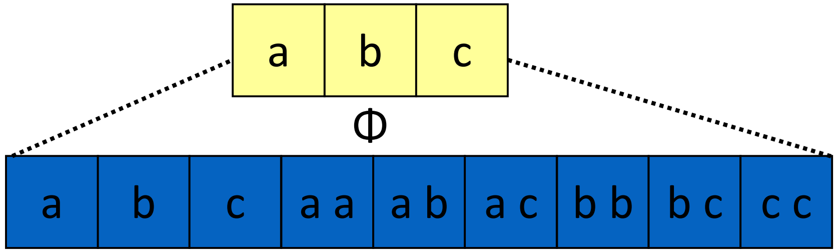


Transformation

- Intuitively, we extend the hypothesis space



- e.g.,



An example 1/2

- Data is generated as follows:

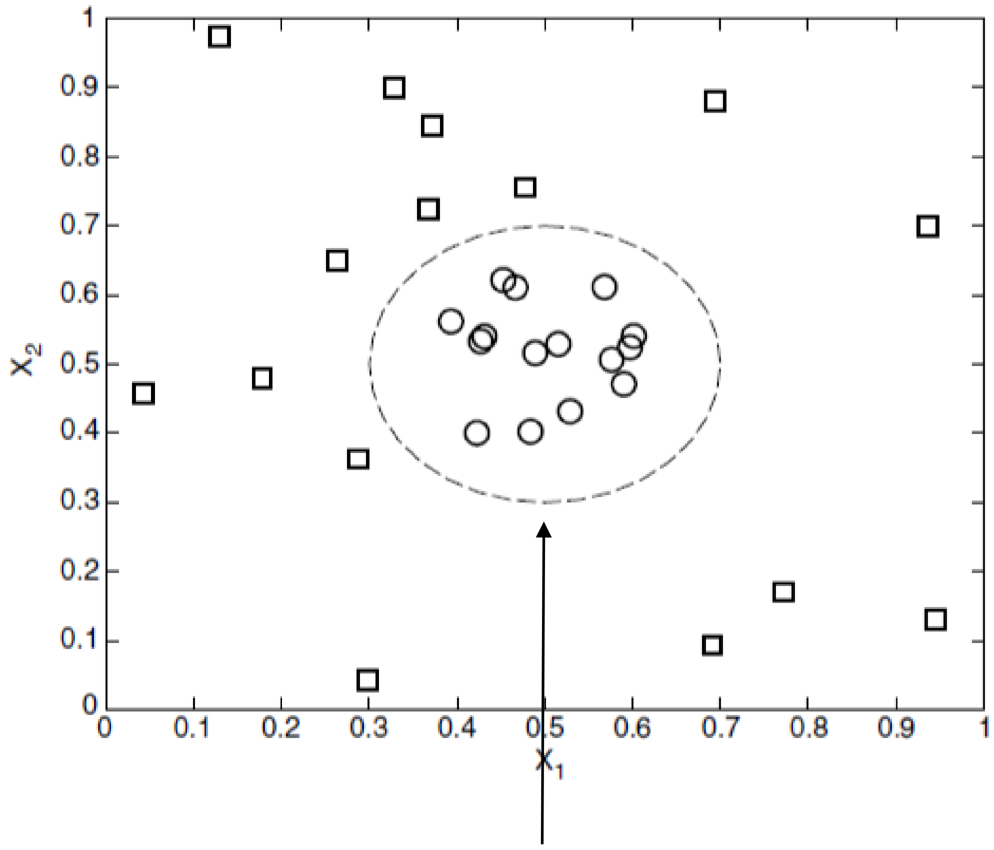
$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2, \\ -1 & \text{otherwise.} \end{cases}$$

- The decision boundary can be written as:

$$\sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} = 0.2,$$

- or, alternatively:

$$x_1^2 - x_1 + x_2^2 - x_2 = -0.46.$$



Decision boundary in the original space is elliptical

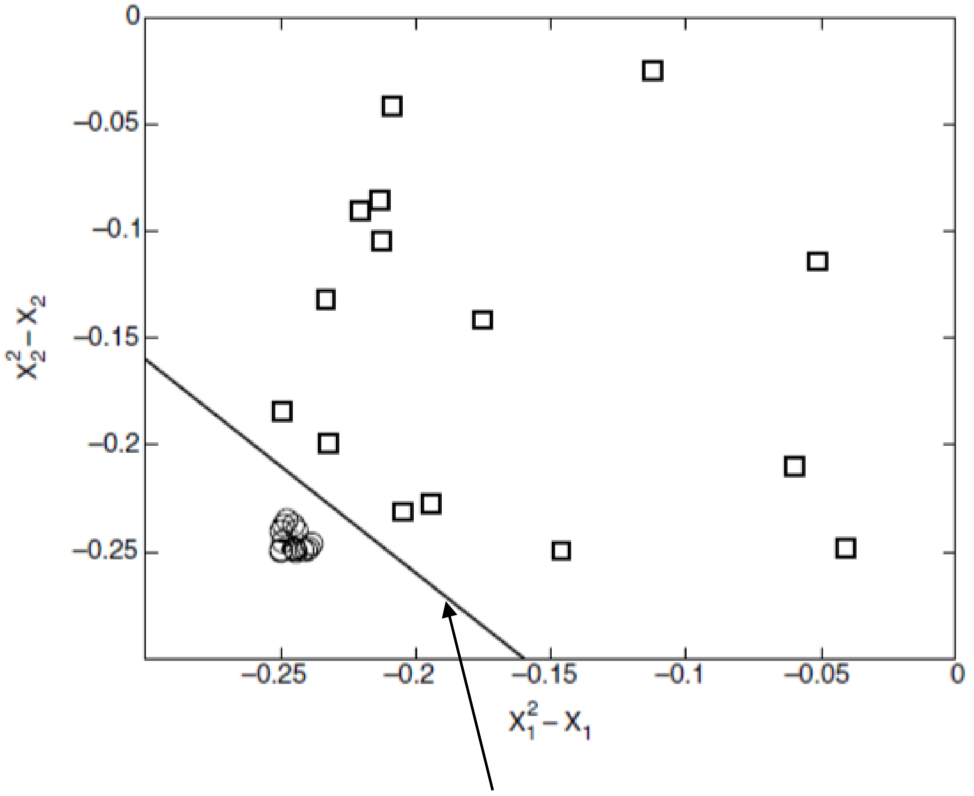
An example 2/2

- A non-linear transformation Φ maps the data into a new space:

$$\Phi : (x_1, x_2) \longrightarrow (x_1^2, x_2^2, \sqrt{2}x_1, \sqrt{2}x_2, 1)$$

- In the transformed space, we can find the parameters of the boundary *w = (w₁,w₂,w₃,w₄)*

$$w_4 x_1^2 + w_3 x_2^2 + w_2\sqrt{2}x_1 + w_1\sqrt{2}x_2 + w_0 = 0.$$

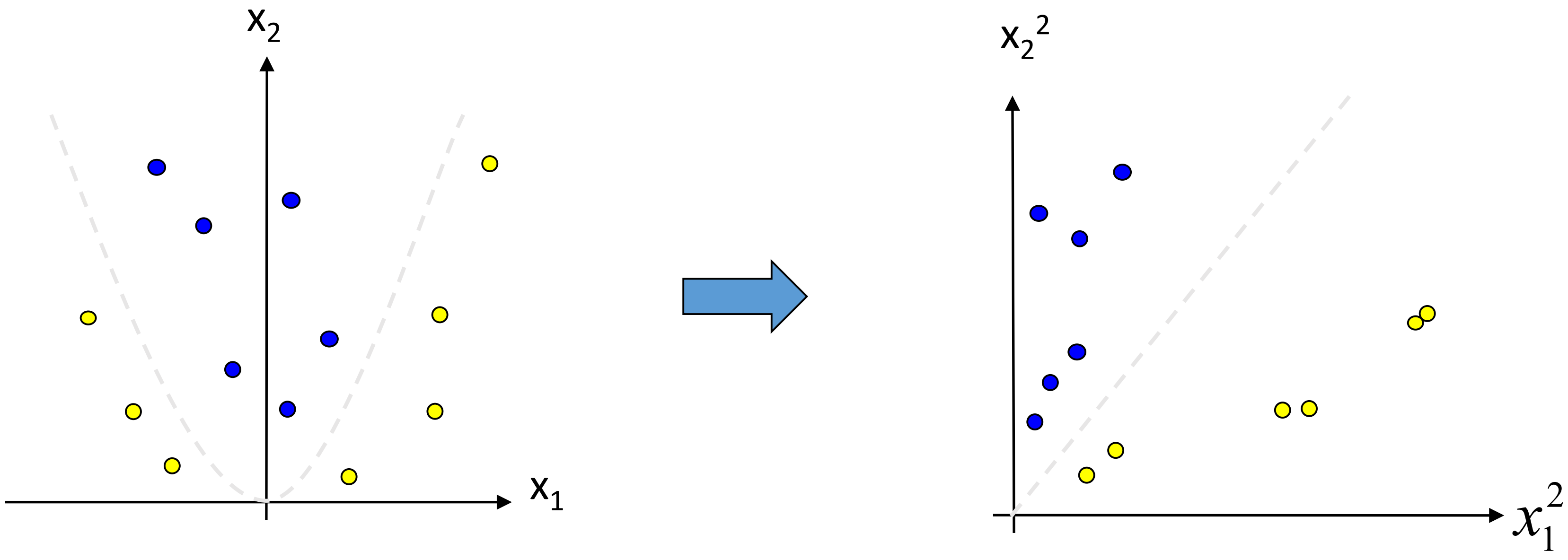


In the transformed space, the decision boundary becomes linear

Another example

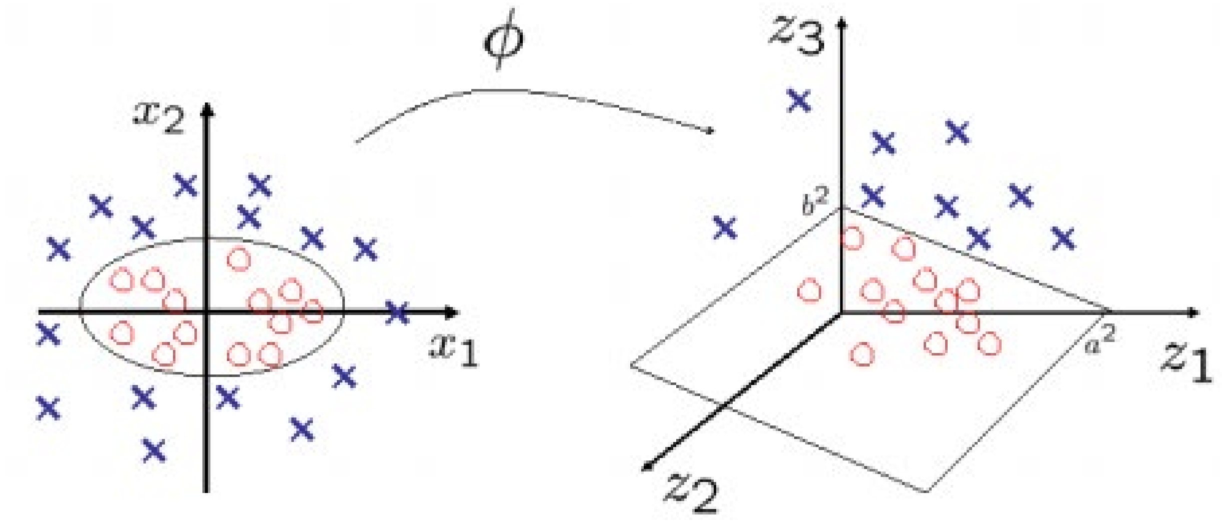
Input space (2D): $\vec{x} = (x_1, x_2)$

Extended space (6D): $\phi(\vec{x}) = \left(x_1^2, x_2^2, \sqrt{2}\cdot x_1, \sqrt{2}\cdot x_2, \sqrt{2}\cdot x_1 \cdot x_2, 1\right)$



Another example

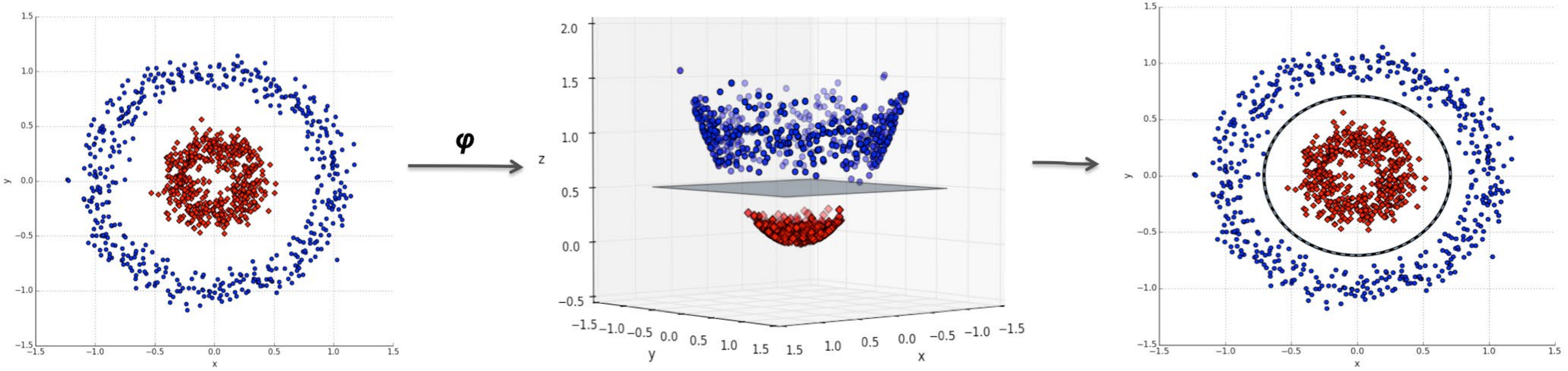
- Elliptical boundary in the input space becomes linear in the transformed space



$$\phi : [x_1, x_2]^T \rightarrow [x_1^2, \sqrt{2}x_1x_2, x_2^2]^T$$

Another example

- The data is linearly separable in the transformed space



Source: http://www.eric-kim.net/eric-kim-net/posts/1/kernel_trick.html

Nonlinear SVM definition

- Updated definition

Need to minimize:

$$\min_{w} \frac{\|\boldsymbol{w}\|^2}{2}$$

subject to the following constraints:

$$y_i(\boldsymbol{w} \cdot \Phi(\boldsymbol{x}_i) + b) \geq 1$$

We work in the transformed space

Original optimization problem

$$\min_{w} \frac{\|\boldsymbol{w}\|^2}{2}$$

subject to

$$y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \geq 1$$

Original boundary

wx *+ b = 0*

- Can be solved used quadratic programming
  - This way we can learn the parameters *w, b* of the decision boundary

Transformed boundary

wΦ(x) *+ b = 0*

Nonlinear SVM definition

The original prediction

$$f(\boldsymbol{x}_j) = sign\left(\sum_{i=1}^{N_s} \alpha_i y_i \boldsymbol{s}_i . \boldsymbol{x}_j + b \right)$$

- How can we classify a new instance x_j?
- Main idea: Use the transformed space
- $f(\boldsymbol{x}_j) = sign(\boldsymbol{w} \cdot \boldsymbol{\Phi}(\boldsymbol{x}_j) + b) = sign(\sum_{i=1}^{n} a_i y_i \boldsymbol{\Phi}(\boldsymbol{x}_i)\boldsymbol{\Phi}(\boldsymbol{x}_j) + b)$
- Involves calculating of the dot product(similarity) in the transformed space
  - computational problem (very large vectors)
  - curse of dimensionality

Kernel trick

- The kernel trick is a method for computing similarity between two instances in the transformed feature space using the original attribute set.
  - e.g., consider transformation

$$\Phi : (u_1, u_2) \rightarrow \left(u_1^2, u_2^2, \sqrt{2}u_1, \sqrt{2}u_2, \sqrt{2}u_1u_2 1\right)$$

  - The dot product between 2 input vectors *u, v* in the transformed space is:

$$\begin{aligned}
\Phi(\mathbf{u}) \cdot \Phi(\mathbf{v}) &= (u_1^2, u_2^2, \sqrt{2}u_1, \sqrt{2}u_2, \sqrt{2}u_1u_2, 1) \cdot (v_1^2, v_2^2, \sqrt{2}v_1, \sqrt{2}v_2, \sqrt{2}v_1v_2, 1) \\
&= u_1^2v_1^2 + u_2^2v_2^2 + 2u_1v_1 + 2u_2v_2 + 2u_1u_2v_1v_2 + 1 \\
&= (\mathbf{u} \cdot \mathbf{v} + 1)^2
\end{aligned}$$

- So, we can express the dot product in *Φ(x)* in terms of a similarity function in the original feature space

kernel function

$$K(u, v) = \Phi(u)\Phi(v) = (uv + 1)^2$$

A function that returns the dot product between the images of two vectors.

Kernels

- The main requirement for a kernel function in nonlinear SVM
- There must exist a transformation such that the kernel function computed for two vectors is equivalent to the dot product between these vectors in the transformed space.

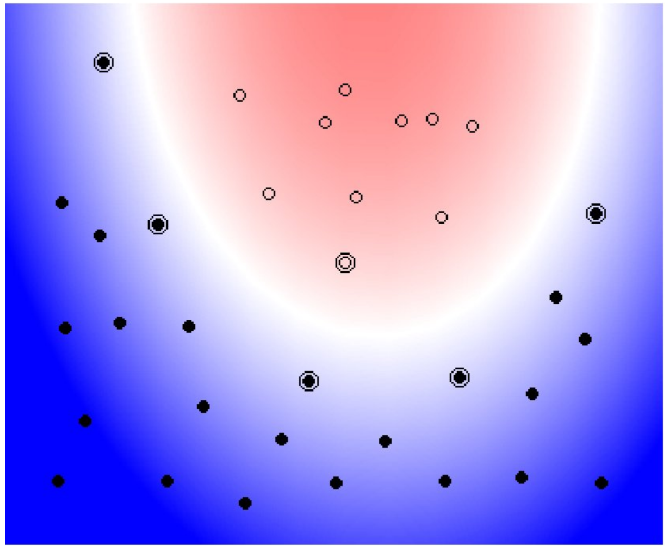
$$K(u,v) = \Phi(u)\Phi(v)$$

Popular kernel functions

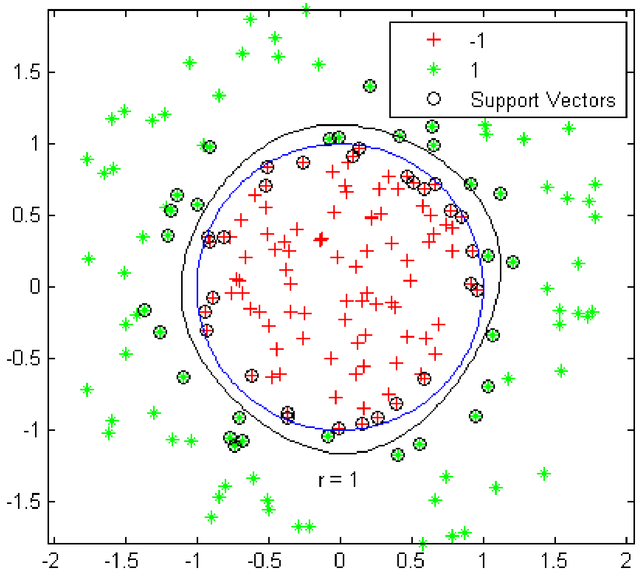
- Linear $$K(\vec{x},\vec{y}) = \langle \vec{x},\vec{y} \rangle$$
- Polynomial $$K(\vec{x},\vec{y}) = \left(\langle \vec{x},\vec{y} \rangle + c\right)^d$$
- Gaussian kernel $$K(\vec{x},\vec{y}) = \exp\left(-\frac{\left\|\vec{x}-\vec{y}\right\|^2}{2\sigma^2}\right)$$
- Radial basis function (RBF) kernel $$K(\vec{x},\vec{y}) = \exp\left(-\gamma \cdot \left|\vec{x}-\vec{y}\right|^2\right)$$
- ....

Choosing the right kernel depends on the problem at hand

Polynomial kernel (degree 2)



Gaussian kernel



SVM overview

- High accuracy classifiers
- Relatively weak tendency to overfitting
- Efficient classification of new objects
- Compact models

- Costly implementation
  - sometimes long training times
  - learned models difficult to interpret

Outline

- Support vector machines - Basic intuition and basic notions
- (Hard-margin) Linear SVM
- Soft-margin linear SVM
- Non-linear SVM
- Things you should know from this lecture & reading material

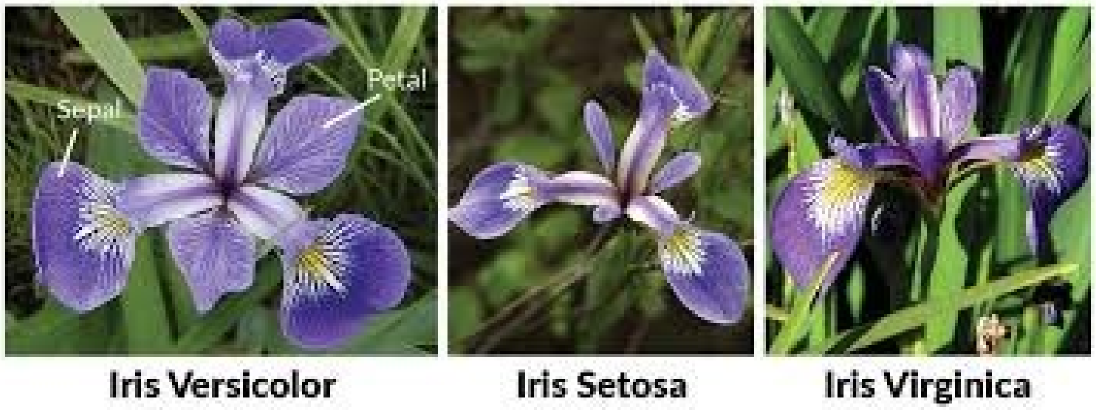
Overview and Reading

- Overview
  - Hard-margin linear SVMs
  - Soft-margin linear SVMs
  - Non-linear SVMs

- Reading
  - Chapter 15: Support Vector Machines, Understanding Machine Learning book by Shai Shalev-Schwartz and Shai Ben-David
  - Dot products and duality | Chapter 9, Essence of linear algebra Youtube

Hands on experience

- Consider the favorite Iris dataset (or choose your favorite dataset)
- Experiment with different kernels and their parameters (if any)
- Plot the decision boundary
- Compare the results



Acknowledgements

- The slides are based on
  - KDD I lecture at LMU Munich (Johannes Aßfalg, Christian Böhm, Karsten Borgwardt, Martin Ester, Eshref Januzaj, Karin Kailing, Peer Kröger, Eirini Ntoutsi, Jörg Sander, Matthias Schubert, Arthur Zimek, Andreas Züfle)
  - Introduction to Data Mining book slides at http://www-users.cs.umn.edu/~kumar/dmbook/
  - Pedro Domingos Machine Lecture course slides at the University of Washington
  - Machine Learning book by T. Mitchel slides at http://www.cs.cmu.edu/~tom/mlbook-chapter-slides.html
  - C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, DMKD 1998
  - Thank you to all TAs contributing to their improvement, namely Vasileios Iosifidis, Damianos Melidis, Tai Le Quy, Han Tran.

Thank you

Questions/Feedback/Wishes?