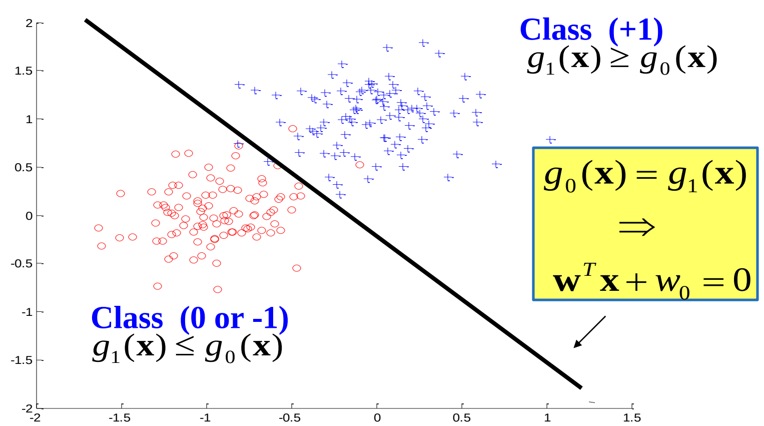
CS 2750 Machine Learning
Lecture 11b

Support vector machines

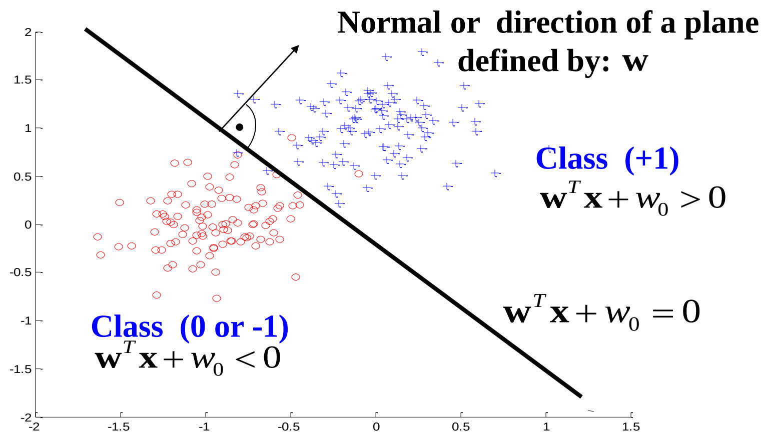
Milos Hauskrecht
milos@cs.pitt.edu
5329 Sennott Square

Linear decision boundaries

- What models define linear decision boundaries?



Linear decision boundaries



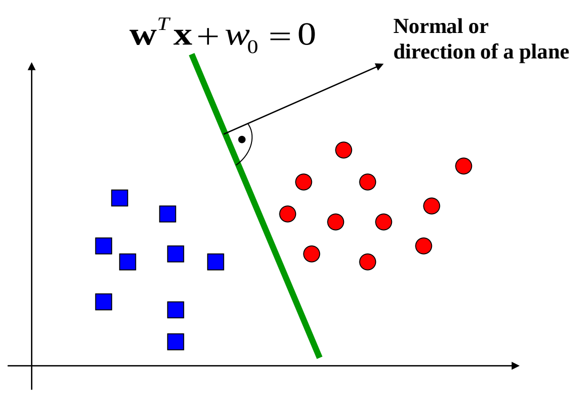
Linearly separable classes

Linearly separable classes:

There is a **hyperplane** $\mathbf{w}^T\mathbf{x} + w_0 = 0$

that separates training instances with no error

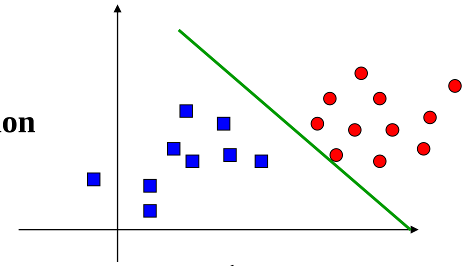
Class (+1) $\mathbf{w}^T\mathbf{x} + w_0 > 0$
Class (-1) $\mathbf{w}^T\mathbf{x} + w_0 < 0$



Learning linearly separable sets

Finding weights for linearly separable classes:

- **Linear program (LP) solution**
- It finds weights that satisfy the following constraints:



$\mathbf{w}^T \mathbf{x}_i + w_0 \geq 0$ For all i, such that $y_i = +1$

$\mathbf{w}^T \mathbf{x}_i + w_0 \leq 0$ For all i, such that $y_i = -1$

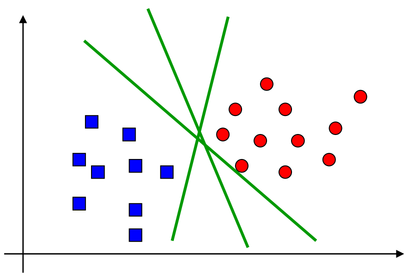
Together: $y_i(\mathbf{w}^T \mathbf{x}_i + w_0) \geq 0$

Property: if there is a hyperplane separating the examples, the linear program finds the solution

Optimal separating hyperplane

Problem:

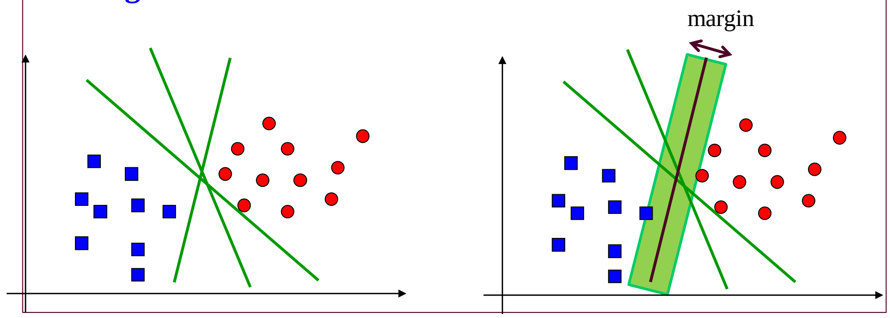
- There are multiple hyperplanes that separate the data points
- Which one to choose?



Optimal separating hyperplane

Problem:

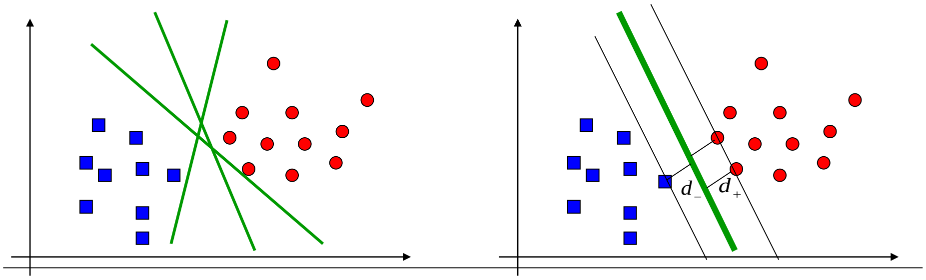
- There are multiple hyperplanes that separate the data points
- Which one to choose?
- **One solution**: hyperplane that maximizes **the width of the margin**



Optimal separating hyperplane

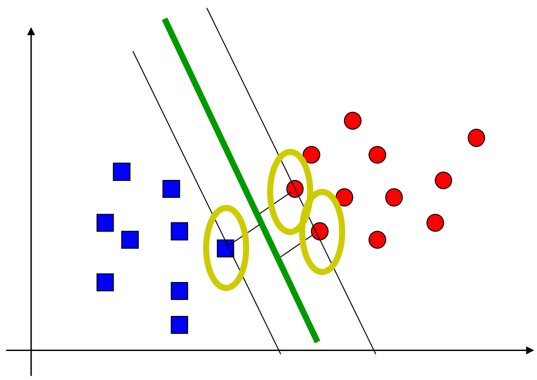
- **Problem:** multiple hyperplanes that separate the data exists
- **Maximum margin** choice: maximum distance of $d_+ + d_-$
  - where d_+ is the shortest distance of a positive example from the hyperplane (similarly d_- for negative examples)

Note: a margin classifier is a classifier for which we can calculate the distance of each example from the decision boundary



Maximum margin hyperplane

- For the maximum margin hyperplane only examples on the margin matter (only these affect the distances)
- These are called **support vectors**

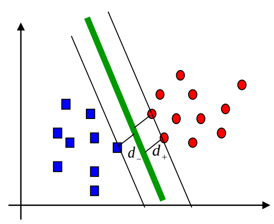


Finding maximum margin hyperplanes

- **Assume** that examples in the training set are $(\mathbf{x}_i, y_i)$ such that $y_i \in \{+1, -1\}$
- **Assume** that all data satisfy:

$$\mathbf{w}^T \mathbf{x}_i + w_0 \geq 1 \quad \text{for} \quad y_i = +1$$

$$\mathbf{w}^T \mathbf{x}_i + w_0 \leq -1 \quad \text{for} \quad y_i = -1$$



- The inequalities can be combined as:

$$y_i(\mathbf{w}^T \mathbf{x}_i + w_0) - 1 \geq 0 \quad \text{for all} \quad i$$

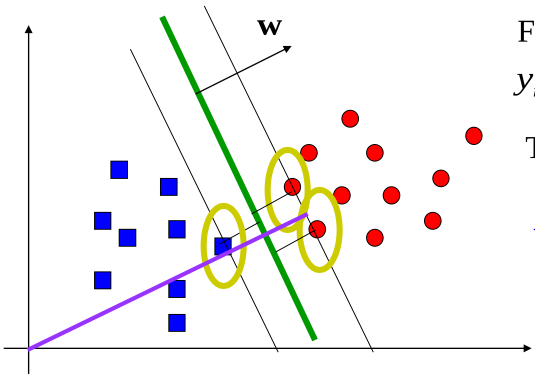
- Equalities define two hyperplanes:

$$\mathbf{w}^T \mathbf{x}_i + w_0 = 1 \qquad \mathbf{w}^T \mathbf{x}_i + w_0 = -1$$

Finding the maximum margin hyperplane

- **Geometric margin:** $\rho_{\mathbf{w},w_0}(\mathbf{x}, y) = y(\mathbf{w}^T\mathbf{x} + w_0)/\|\mathbf{w}\|_{L2}$
  - measures the distance of a point **x** from the hyperplane

    $\mathbf{w}$ - normal to the hyperplane $\|..\|_{L2}$ - Euclidean norm



For points satisfying:

$y_i(\mathbf{w}^T\mathbf{x}_i + w_0) - 1 = 0$

The distance is $\frac{1}{\|\mathbf{w}\|_{L2}}$

Width of the margin:

$$d_+ + d_- = \frac{2}{\|\mathbf{w}\|_{L2}}$$

Maximum margin hyperplane

- **We want to maximize** $d_+ + d_- = \frac{2}{\|\mathbf{w}\|_{L2}}$

- We do it by **minimizing**

$$\|\mathbf{w}\|_{L2}^{2}/2 = \mathbf{w}^T\mathbf{w}/2$$

$\mathbf{w}, w_0$ - variables

- But we also need to enforce the constraints on data instances: $(\mathbf{x}_i, y_i)$

$$\left[y_i(\mathbf{w}^T\mathbf{x}_i + w_0) - 1\right] \geq 0$$

Maximum margin hyperplane

- **Solution: Incorporate constraints into the optimization**
- **Optimization problem** (Lagrangian)

Data instances $(\mathbf{x}_i, y_i)$

$$J(\mathbf{w}, w_0, \alpha) = \|\mathbf{w}\|^2 / 2 - \sum_{i=1}^{n} \alpha_i \left[y_i (\mathbf{w}^T \mathbf{x}_i + w_0) - 1 \right]$$

$\alpha_i \geq 0$ - **Lagrange multipliers**

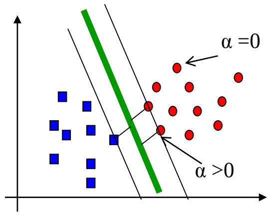
- **Minimize** with respect to $\mathbf{w}, w_0$ (primal variables)
- **Maximize** with respect to $\boldsymbol{\alpha}$ (dual variables)

What happens to α:

if $y_i(\mathbf{w}^T \mathbf{x}_i + w_0) - 1 > 0 \Longrightarrow \alpha_i \to 0$

else $\Longrightarrow \alpha_i > 0$

Active constraint



Max margin hyperplane solution

- Set derivatives to 0 (Kuhn-Tucker conditions)

$$\nabla_{\mathbf{w}} J(\mathbf{w}, w_0, \alpha) = \mathbf{w} - \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i = \bar{0} \quad \Rightarrow \quad \boxed{\mathbf{w} = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i}$$

$$\frac{\partial J(\mathbf{w}, w_0, \alpha)}{\partial w_0} = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

- Now we need to solve for Lagrange parameters (Wolfe dual)

$$J(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i^T \mathbf{x}_j) \quad \Leftarrow \quad \textbf{maximize}$$

Subject to constraints

$\alpha_i \geq 0$ for all i, and $\sum_{i=1}^{n} \alpha_i y_i = 0$

- **Quadratic optimization problem:** solution $\hat{\alpha}_i$ for all i

Maximum margin solution

- The resulting parameter vector $\hat{\mathbf{w}}$ can be expressed as:

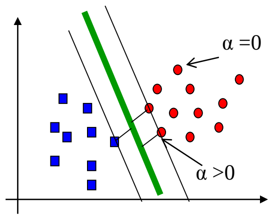
$$\hat{\mathbf{w}} = \sum_{i=1}^{n} \hat{\alpha}_i y_i \mathbf{x}_i \qquad \hat{\alpha}_i \text{ is the solution of the optimization}$$

- The parameter w_0 is obtained from $\hat{\alpha}_i [y_i (\hat{\mathbf{w}}\mathbf{x}_i + w_0) - 1] = 0$

Solution properties

- $\hat{\alpha}_i = 0$ for all points that are not on the margin
- **The decision boundary:**

$$\hat{\mathbf{w}}^T \mathbf{x} + w_0 = \sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0 = 0$$



The decision boundary defined by support vectors only

Support vector machines: solution property

- **Decision boundary defined by a set of support vectors SV and their alpha values**
  - **Support vectors = a subset of datapoints in the training data that define the margin**

$$\hat{\mathbf{w}}^T \mathbf{x} + w_0 = \sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0$$

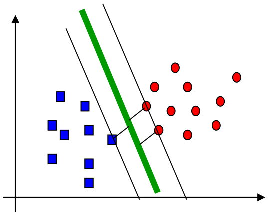
Lagrange multipliers

- **Classification decision for new x:**

$$\hat{y} = \operatorname{sign}\left[\sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0 \right]$$

- **Note that we do not have to explicitly compute** $\hat{\mathbf{w}}$
  - This will be important for the nonlinear (kernel) case

Support vector machines



- **The decision boundary:**

$$\hat{\mathbf{w}}^T\mathbf{x} + w_0 = \sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0$$

- **Classification decision:**

$$\hat{y} = \operatorname{sign}\left[\sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0\right]$$

Support vector machines: inner product

- Decision on a new **x** depends on the **inner product between two examples**
- **The decision boundary:**

$$\hat{\mathbf{w}}^T\mathbf{x} + w_0 = \sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0$$

- **Classification decision:**

$$\hat{y} = \operatorname{sign}\left[\sum_{i \in SV} \hat{\alpha}_i y_i (\mathbf{x}_i{}^T \mathbf{x}) + w_0\right]$$

- Similarly, the optimization depends on $(\mathbf{x}_i{}^T \mathbf{x}_j)$

$$J(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i{}^T \mathbf{x}_j)$$

Inner product of two vectors

- The decision boundary for the SVM and its optimization depend on the inner product of two datapoints (vectors):

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

$$\mathbf{x}_i = \begin{pmatrix} 2 \\ 5 \\ 6 \end{pmatrix} \qquad \mathbf{x}_j = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}$$

$$(\mathbf{x}_i^T \mathbf{x}_j) = \quad ?$$

Inner product of two vectors

- The decision boundary for the SVM and its optimization depend on the inner product of two data points (vectors):

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

$$\mathbf{x}_i = \begin{pmatrix} 2 \\ 5 \\ 6 \end{pmatrix} \qquad \mathbf{x}_j = \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}$$

$$(\mathbf{x}_i^T \mathbf{x}_j) = \begin{pmatrix} 2 & 5 & 6 \end{pmatrix} * \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix} = 2*2+5*3+6*1 = 25$$

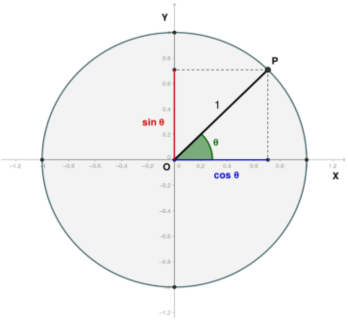
Inner product of two vectors

- **The decision boundary for the SVM and its optimization depend on the inner product of two data points (vectors):**

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

- **The inner product is equal**

$$(\mathbf{x}_i^T \mathbf{x}_j) = \|\mathbf{x}_i\| * \|\mathbf{x}_j\| \cos\theta$$



If the angle in between them is 0 then:

$$(\mathbf{x}_i^T \mathbf{x}_j) = \|\mathbf{x}_i\| * \|\mathbf{x}_j\|$$

If the angle between them is 90 then:

$$(\mathbf{x}_i^T \mathbf{x}_j) = 0$$

The inner product measures how similar the two vectors are