Duality

Duality is a powerful algorithm design tool that allows us to explore different algorithmic alternatives based on the inherent constraints of the underlying problem.

In our case the constraints on the solution (the position of the decision surface) are the observations close to the class boundaries where the classes meet.

We find that the dual solution is dominated by the constraints represented by the points close to the class boundaries.

Perceptron Learning

$\vdots$

repeat

  for $i = 1$ **to** l

    if $\text{sign}(\overline{w} \bullet \overline{x}_i - b) \neq y_i$ **then**

      $\overline{w} \leftarrow \overline{w} + \eta y_i \overline{x}_i$

      $b \leftarrow b - \eta y_i r^2$

    end if

  end for

until $\text{sign}(\overline{w} \bullet \overline{x}_j - b) = y_j$ with $j = 1, \ldots, l$

$\vdots$

Observation: $\overline{w}$ will contain many copies of the scaled position vector $\overline{x}_i$ for “difficult” points; points that are difficult to classify.

Idea: Instead of updating the normal vector of the decision surface, count the number of times a particular point is being misclassified.

Duality

Introduce a new variable that keeps track of the misclassification counts:

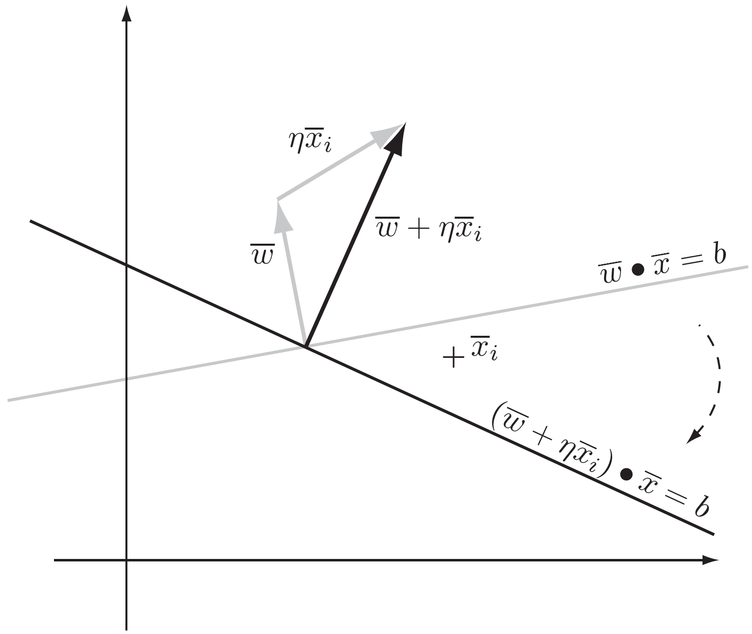
$$\overline{\alpha} = (\alpha_1, \ldots, \alpha_l)$$

Then we can write our learning algorithm as,

Initialize $\overline{\alpha}$ and b to 0.
repeat
  for each $(\overline{x}_i, y_i) \in D$ **do**
    if $\hat{f}(\overline{x}_i) \neq y_i$ **then**
      Increment α_i by 1
      Update b
    end if
  end for
until D is perfectly classified
return $\overline{\alpha}$ and b

Problem: How do we represent our free parameter $\overline{w}$ using $\overline{\alpha}$? Or $\hat{f}$ for that matter?

Duality



$$\overline{w} = \sum_{i=1}^{l} \eta\alpha_i y_i \overline{x}_i$$
$$= \eta \sum_{i=1}^{l} \alpha_i y_i \overline{x}_i.$$

Duality

This means that we can rewrite the normal vector $\overline{w}$ as the following sum,

$$\overline{w} = \sum_{i=1}^{l} \alpha_i y_i \overline{x}_i$$

where

$\alpha_i \approx 0$ for “easy” points,

$\alpha_i \gg 1$ for “difficult” points.

Note: The learning rate η is simply a scaling constant of the resulting normal vector $\overline{w}$ and since we are primarily interested in the orientation of the decision surface it is customary to drop this constant.

Duality

Now our perceptron decision function now looks like

$$\hat{f}(\overline{x}) = \text{sign}(\overline{w} \bullet \overline{x} - b)$$
$$= \text{sign}\left((\sum_{i=1}^{l} \alpha_i y_i \overline{x}_i \bullet \overline{x}) - b \right)$$

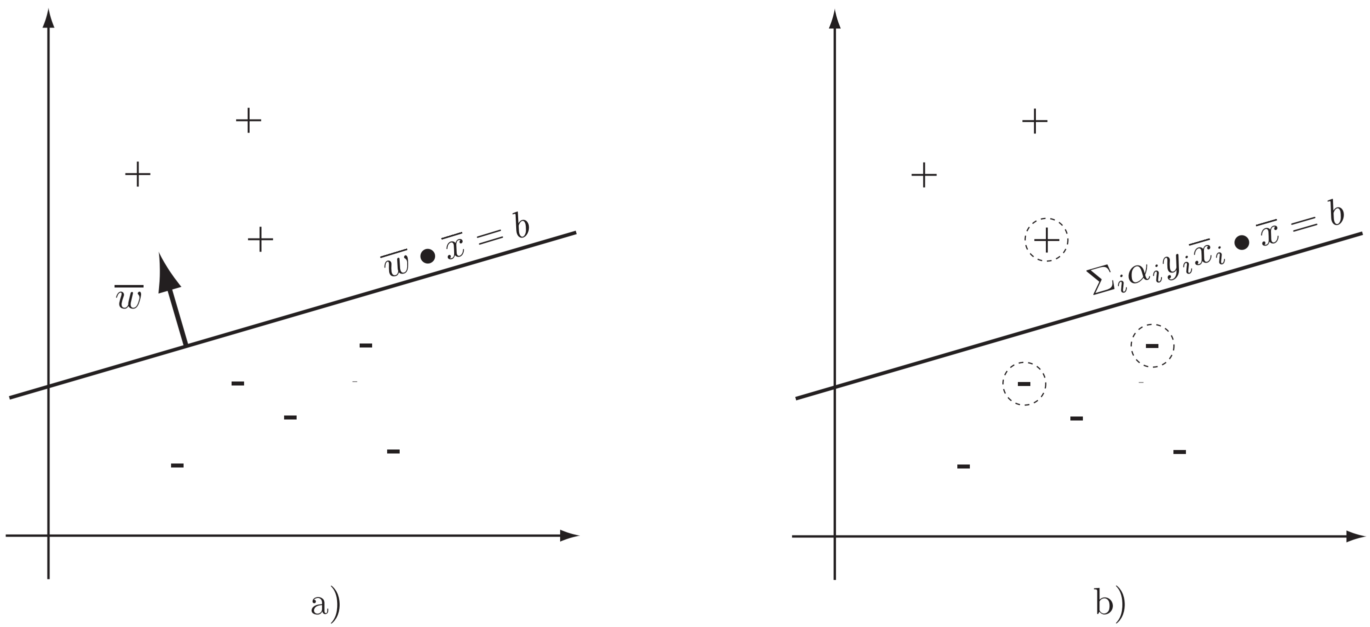
where l is the number of instances in D.

Duality

let $D = \{(\overline{x}_1, y_1), (\overline{x}_2, y_2), \ldots, (\overline{x}_l, y_)\} \subset \mathbb{R}^n \times \{+1, -1\}$
let $0 < \eta < 1$
$\overline{\alpha} \leftarrow \overline{0}$
$b \leftarrow 0$
$r \leftarrow \max\{|\overline{x}| \mid (\overline{x}, y) \in D\}$
repeat
  for $i = 1$ **to** l
    if $\mathrm{sign}(\sum_{j=1}^{l} \alpha_j y_j \overline{x}_j \bullet \overline{x}_i - b) \neq y_i$ **then**
      $\alpha_i \leftarrow \alpha_i + 1$
      $b \leftarrow b - \eta y_i r^2$
    end if
  end for
until $\mathrm{sign}(\sum_{j=1}^{l} \alpha_j y_j \overline{x}_j \bullet \overline{x}_k - b) = y_l$ with $k = 1, \ldots, l$
return $(\overline{\alpha}, b)$

Duality

The two perceptron learning algorithms give rise to different representations of the induced decision surface.



Duality

The dual algorithm still can give rise to degenerate decision surfaces.