Perceptron

Perceptron

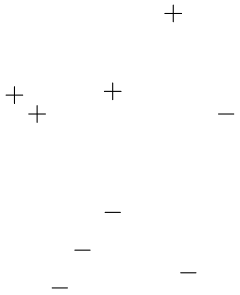
Support Vector Machines

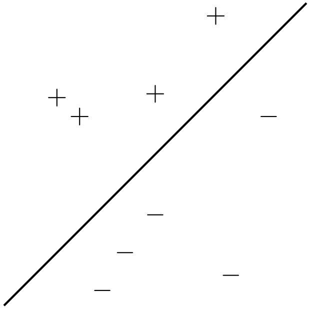
Perceptron

Support Vector Machines

Lagrange Multiplier

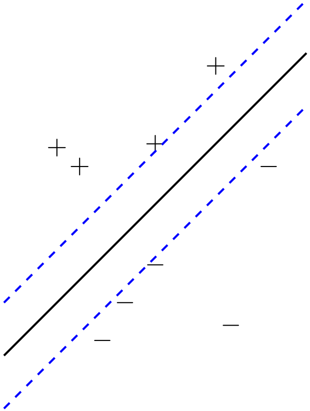
Labelled points with $x_1, \ldots, x_n$.





Labelled points with $x_1, \ldots, x_n$.

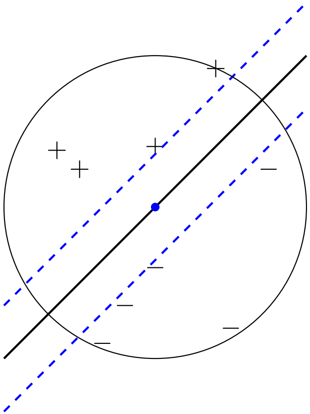
Hyperplane separator.



Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

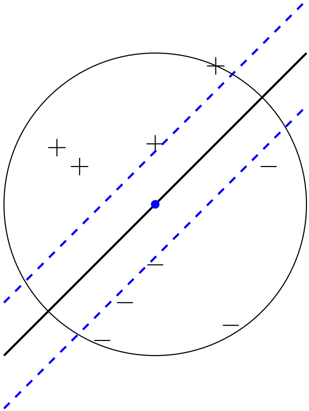


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.

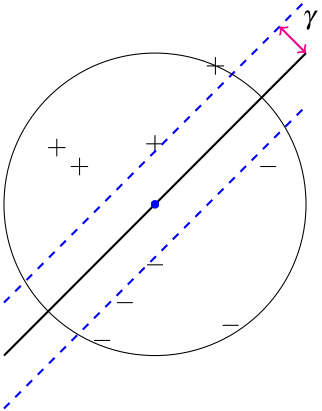


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.

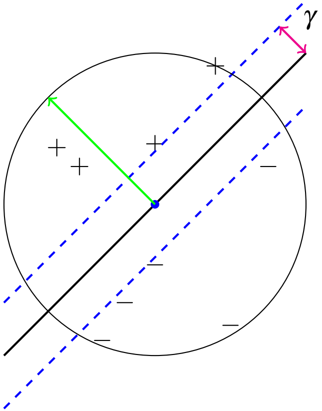


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ

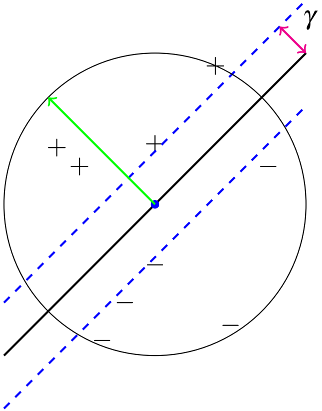


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:

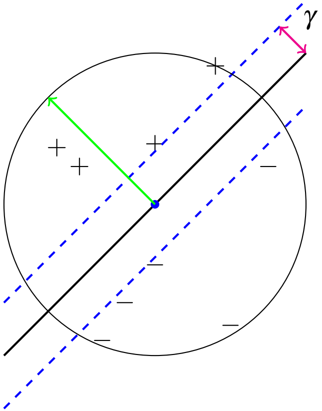


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
$w \cdot x \geq \gamma$ for $+$ points.

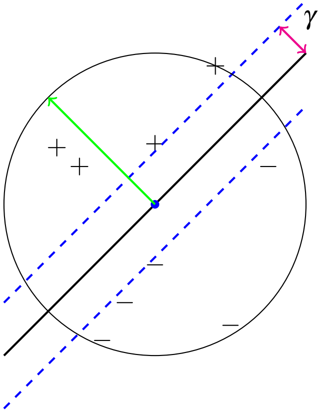


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
$w \cdot x \geq \gamma$ for $+$ points.
$w \cdot x \leq -\gamma$ for $-$ points.

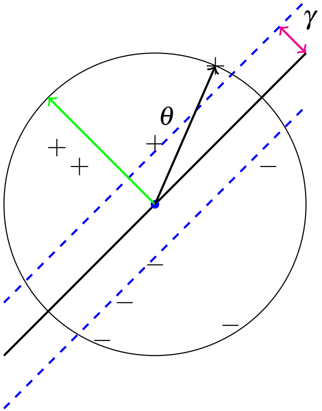


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.

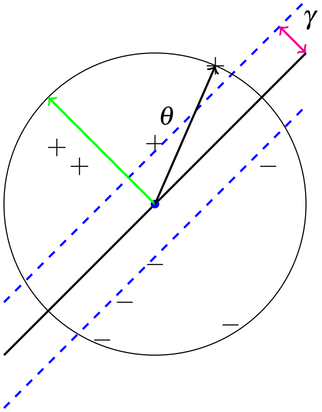


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.
  $w \cdot x = cos\theta$

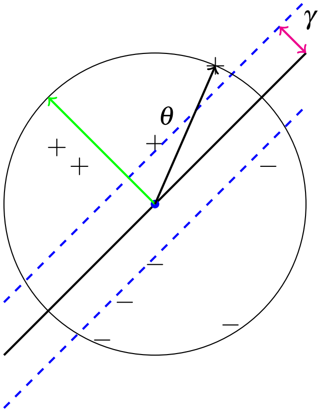


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.
  $w \cdot x = cos\theta$
Will assume positive labels!

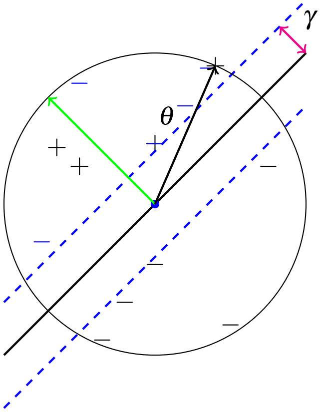


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.
  $w \cdot x = cos\theta$
Will assume positive labels!
  negate the negative.

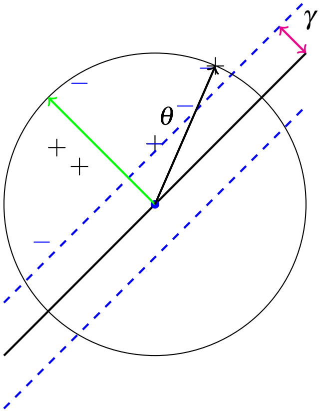


Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.
  $w \cdot x = cos\theta$
Will assume positive labels!
  negate the negative.



Labelled points with $x_1, \ldots, x_n$.

Hyperplane separator.

Margins.

Inside unit ball.
Margin γ
Hyperplane:
  $w \cdot x \geq \gamma$ for $+$ points.
  $w \cdot x \leq -\gamma$ for $-$ points.
Put points on unit ball.
  $w \cdot x = cos\theta$
Will assume positive labels!
  negate the negative.

Perceptron Algorithm

An aside: a hyperplane is a perceptron.

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network,

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see?

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:

$$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$$

A step in the right direction!

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$
A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:

$$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$$

A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.
A γ in the right direction!

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1$.
A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.
A γ in the right direction!

Mistake on positive x_i;

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1$.
A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.
A γ in the right direction!

Mistake on positive x_i;
$w_{t+1} \cdot w = (w_t + x_i) \cdot w = w_t \cdot w + x_i \cdot w$

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$
A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.
A γ in the right direction!

Mistake on positive x_i;

$$w_{t+1} \cdot w = (w_t + x_i) \cdot w = w_t \cdot w + x_i \cdot w$$
$$\geq w_t \cdot w + \gamma.$$

Perceptron Algorithm

An aside: a hyperplane is a perceptron.
(single layer neural network, do you see? Linear programming!)

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Theorem: Algorithm only makes $\frac{1}{\gamma^2}$ mistakes.

Idea: Mistake on positive x_i:
$$w_{t+1} \cdot x_i = (w_t + x_i) \cdot x_i = w_t x_i + 1.$$
A step in the right direction!

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.
A γ in the right direction!

Mistake on positive x_i;
$$w_{t+1} \cdot w = (w_t + x_i) \cdot w = w_t \cdot w + x_i \cdot w$$
$$\geq w_t \cdot w + \gamma.$$

$\square$

Alg: Given $x_1, \ldots, x_n$.

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.

For each x_i where $w_t \cdot x_i$ has wrong sign (negative)

$w_{t+1} = w_t + x_i$

$t = t + 1$

Alg: Given $x_1, \ldots, x_n$**.**

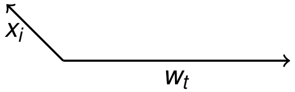
Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

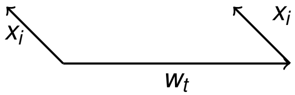
Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$



Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

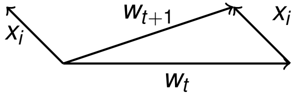


Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

$$w_{t+1} = w_t + x_i$$

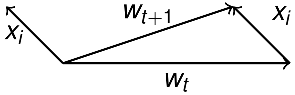


Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

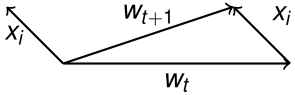
$w_{t+1} = w_t + x_i$
Less than a right angle!



Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

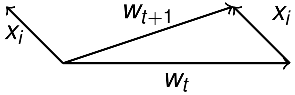


$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1.$

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$

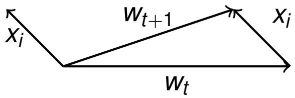


$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1.$
Algebraically.

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$



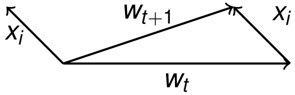
$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1$.

Algebraically.
Positive x_i, $w_t \cdot x_i \leq 0$.

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$



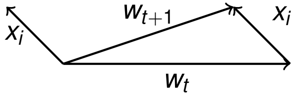
$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1.$

Algebraically.
Positive x_i, $w_t \cdot x_i \leq 0$.
$(w_t + x_i)^2 = |w_t|^2 + 2w_t \cdot x_i + |x_i|^2$.

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \le |w_t|^2 + 1$



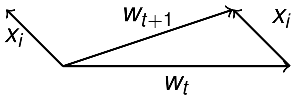
$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \le |w_t|^2 + |x_i|^2 \le |w_t|^2 + 1$.

Algebraically.
Positive x_i, $w_t \cdot x_i \le 0$.
$(w_t + x_i)^2 = |w_t|^2 + 2w_t \cdot x_i + |x_i|^2$.
$\le |w_t|^2 + |x_i|^2 = |w_t|^2 + 1$.

Alg: Given $x_1, \ldots, x_n$**.**

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$



$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1.$

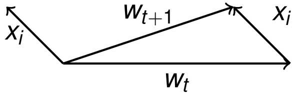
Algebraically.
Positive x_i, $w_t \cdot x_i \leq 0$.
$(w_t + x_i)^2 = |w_t|^2 + 2w_t \cdot x_i + |x_i|^2$.
$\leq |w_t|^2 + |x_i|^2 = |w_t|^2 + 1.$

$\square$

Alg: Given $x_1, \ldots, x_n$.

Let $w_1 = x_1$.
For each x_i where $w_t \cdot x_i$ has wrong sign (negative)
$w_{t+1} = w_t + x_i$
$t = t + 1$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$



$w_{t+1} = w_t + x_i$
Less than a right angle!
$\rightarrow |w_{t+1}|^2 \leq |w_t|^2 + |x_i|^2 \leq |w_t|^2 + 1.$
Algebraically.
Positive x_i, $w_t \cdot x_i \leq 0$.
$(w_t + x_i)^2 = |w_t|^2 + 2w_t \cdot x_i + |x_i|^2$.
$\leq |w_t|^2 + |x_i|^2 = |w_t|^2 + 1.$

$\square$

Claim 2 holds even if no separating hyperplane!

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$. $\implies w_t \cdot w \geq t\gamma$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1.$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$. $\implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1$. $\implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

γM

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

$\gamma M \leq w_M \cdot w$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

$$
\begin{aligned}
\gamma M &\leq w_M \cdot w \\
&\leq ||w_M||
\end{aligned}
$$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

$$\begin{aligned} \gamma M &\leq w_M \cdot w \\ &\leq ||w_M|| \leq \sqrt{M}. \end{aligned}$$

Putting it together...

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma. \implies w_t \cdot w \geq t\gamma$

Claim 2: $|w_{t+1}|^2 \leq |w_t|^2 + 1. \implies |w_t|^2 \leq t$

M-number of mistakes in algorithm.
Let $t = M$.

$$\gamma M \leq w_M \cdot w$$
$$\leq ||w_M|| \leq \sqrt{M}.$$

$\rightarrow M \leq \frac{1}{\gamma^2}$

Hinge Loss.

Most of data has good separator.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
  Total rotation: TD_γ.
    Analysis: subtract bad tilting part.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
  Total rotation: TD_γ.
    Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow$

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
  Total rotation: TD_γ.
    Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Uh...

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Uh...

One implication: $M \leq \frac{1}{\gamma^2} + \frac{2}{\gamma} TD_\gamma$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Uh...

One implication: $M \leq \frac{1}{\gamma^2} + \frac{2}{\gamma} TD_\gamma$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
  Total rotation: TD_γ.
    Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Uh...

One implication: $M \leq \frac{1}{\gamma^2} + \frac{2}{\gamma} TD_\gamma$.

The extra is (twice) the amount of rotation in units of $1/\gamma$.

Hinge Loss.

Most of data has good separator.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Don't make progress or tilt the wrong way.

How much bad tilting?

Rotate points to have γ-margin.
Total rotation: TD_γ.
Analysis: subtract bad tilting part.

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma -$ rotation for x_{i_t}.

$w_M \geq \gamma M - TD_\gamma$ + Claim 2. $\rightarrow \gamma M - TD_\gamma \leq \sqrt{M}$

Quadratic equation: $\gamma^2 M^2 - (2\gamma TD_\gamma + 1)M + TD_\gamma^2 \leq 0$.

Uh...

One implication: $M \leq \frac{1}{\gamma^2} + \frac{2}{\gamma} TD_\gamma$.

The extra is (twice) the amount of rotation in units of $1/\gamma$.

Hinge loss: $\frac{1}{\gamma} TD_\gamma$.

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it!

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,
if $w_t \cdot x_i < \gamma/2$, $w_{t+1} = w_t + x_i$,

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,
if $w_t \cdot x_i < \gamma/2$, $w_{t+1} = w_t + x_i$, $t = t + 1$

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,
if $w_t \cdot x_i < \gamma/2$, $w_{t+1} = w_t + x_i$, $t = t + 1$

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,
if $w_t \cdot x_i < \gamma/2$, $w_{t+1} = w_t + x_i$, $t = t + 1$

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Same

Approximately Maximizing Margin Algorithm

There is a γ separating hyperplane.

Find it! (Kind of.)

Any point within $\gamma/2$ is still a mistake.

Let $w_1 = x_1$,

For each $x_2, \ldots x_n$,
if $w_t \cdot x_i < \gamma/2$, $w_{t+1} = w_t + x_i$, $t = t + 1$

Claim 1: $w_{t+1} \cdot w \geq w_t \cdot w + \gamma$.

Same (ish) as before.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

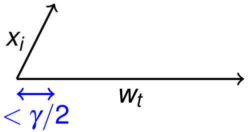
Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

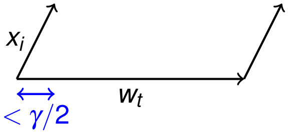
Adding x_i to w_t even if in correct direction.



Margin Approximation: Claim 2

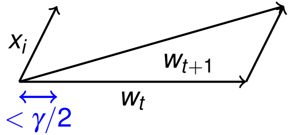
Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

Adding x_i to w_t even if in correct direction.



Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

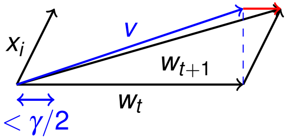


Adding x_i to w_t even if in correct direction.

Obtuse triangle.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??

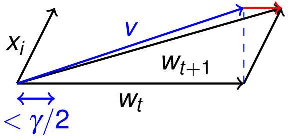


Adding x_i to w_t even if in correct direction.

Obtuse triangle.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



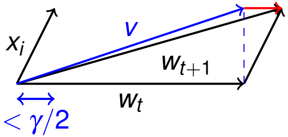
Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



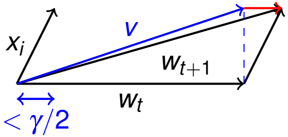
Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

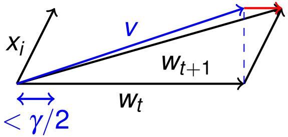
$|v|^2 \leq |w_t|^2 + 1$

$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$

(square right hand side.)

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

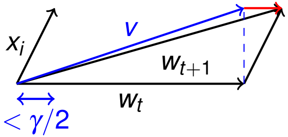
Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

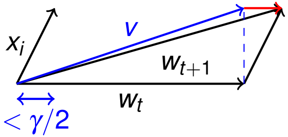
$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

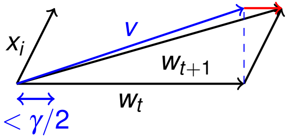
Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

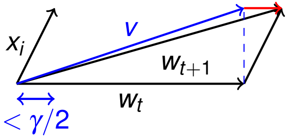
Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

M updates

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

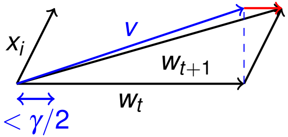
Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

M updates $|w_M| \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

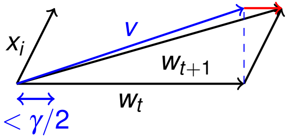
If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

M updates $|w_M| \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$.

Claim 1:

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

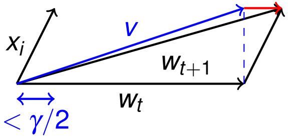
If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

M updates $|w_M| \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$.

Claim 1: Implies $|w_M| \geq \gamma M$.

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

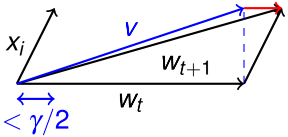
M updates $|w_M| \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$.

Claim 1: Implies $|w_M| \geq \gamma M$.

$\gamma M \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$

Margin Approximation: Claim 2

Claim 2(?): $|w_{t+1}|^2 \leq |w_t|^2 + 1$??



Adding x_i to w_t even if in correct direction.

Obtuse triangle.

$|v|^2 \leq |w_t|^2 + 1$
$\rightarrow |v| \leq |w_t| + \frac{1}{2|w_t|}$
(square right hand side.)

Red bit is at most $\gamma/2$.

Together: $|w_{t+1}| \leq |w_t| + \frac{1}{2|w_t|} + \frac{\gamma}{2}$

If $|w_t| \geq \frac{2}{\gamma}$, then $|w_{t+1}| \leq |w_t| + \frac{3}{4}\gamma$.

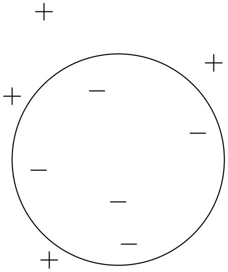
M updates $|w_M| \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M$.

Claim 1: Implies $|w_M| \geq \gamma M$.

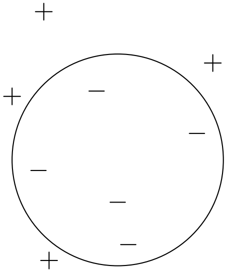
$\gamma M \leq \frac{2}{\gamma} + \frac{3}{4}\gamma M \rightarrow M \leq \frac{8}{\gamma^2}$

Support Vector Machines.

Other fat separators?

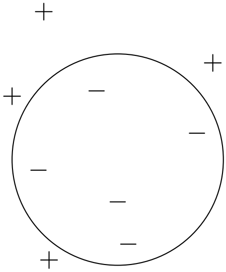


Other fat separators?



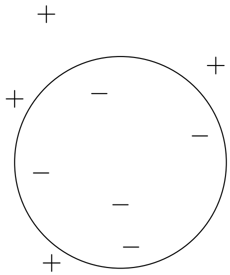
No hyperplane separator.

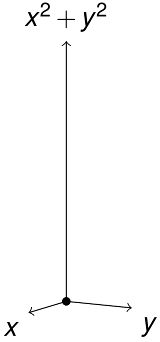
Other fat separators?



No hyperplane separator.
Circle separator!

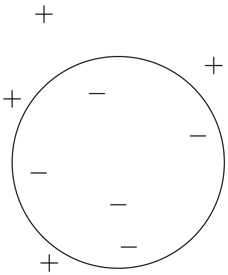
Other fat separators?

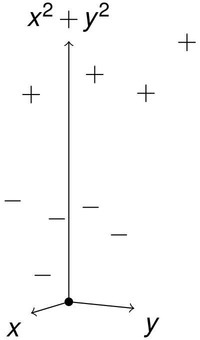




No hyperplane separator.
Circle separator!
Map points to three dimensions.

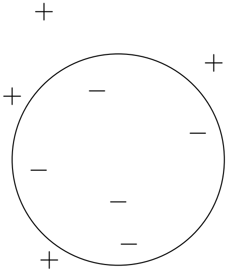
Other fat separators?

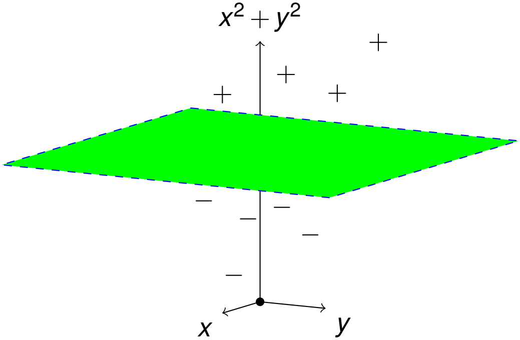




No hyperplane separator.
Circle separator!
Map points to three dimensions.
map point (x, y) to point $(x, y, x^2 + y^2)$.

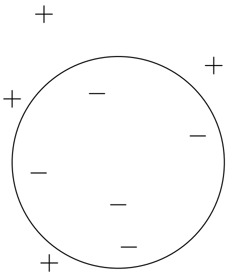
Other fat separators?

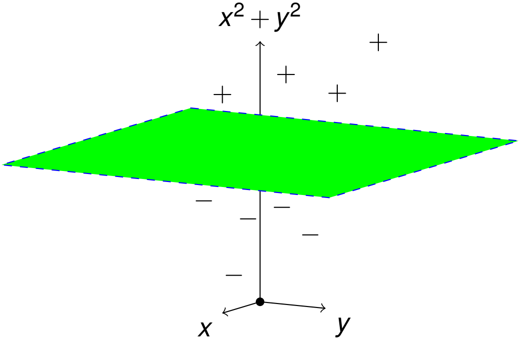




No hyperplane separator.
Circle separator!
Map points to three dimensions.
  map point (x, y) to point $(x, y, x^2 + y^2)$.
Hyperplane separator in three dimensions.

Other fat separators?





No hyperplane separator.
Circle separator!
Map points to three dimensions.
  map point (x, y) to point $(x, y, x^2 + y^2)$.
Hyperplane separator in three dimensions.

Kernel Functions.

Map x to $\phi(x)$.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
  Test: $w_t \cdot x_i > \gamma$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!

Test: $w_t \cdot x_i > \gamma$

$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$

$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!

Test: $w_t \cdot x_i > \gamma$

$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$

$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$:

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!

Test: $w_t \cdot x_i > \gamma$

$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$

$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
  Test: $w_t \cdot x_i > \gamma$
  $w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$
$\phi(x)$ - products of all subsets.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
  Test: $w_t \cdot x_i > \gamma$
  $w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$
  $\phi(x)$ - products of all subsets. Boolean Fourier basis.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$
$\phi(x)$ - products of all subsets. Boolean Fourier basis.

$K(x,y) = exp(C|x-y|^2)$

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$
$\phi(x)$ - products of all subsets. Boolean Fourier basis.

$K(x,y) = exp(C|x-y|^2)$ Infinite dimensional space.
Expansion of e^z.

Kernel Functions.

Map x to $\phi(x)$.

Hyperplane separator for points under $\phi(\cdot)$.

Problem: complexity of computing in higher dimension.

Recall perceptron. Only compute dot products!
Test: $w_t \cdot x_i > \gamma$
$w_t = x_{i_1} + x_{i_2} + x_{i_3} \cdots$

Support Vectors: $x_{i_1}, x_{i_2}, \ldots$
$\rightarrow$ Support Vector Machine.

Kernel trick: compute dot products in original space.

Kernel function for mapping $\phi(\cdot)$: $K(x,y) = \phi(x) \cdot \phi(y)$

$K(x,y) = (1 + x \cdot y)^d$ $\phi(x) = [1, \ldots, x_i, \ldots, x_i x_j \ldots]$. Polynomial.

$K(x,y) = (1 + x_1 y_1)(1 + x_2 y_2) \cdots (1 + x_n y_n)$
$\phi(x)$ - products of all subsets. Boolean Fourier basis.

$K(x,y) = exp(C|x-y|^2)$ Infinite dimensional space.
Expansion of e^z. Gaussian Kernel.

Video

“http://www.youtube.com/watch?v=3liCbRZPrZA”

Support Vector Machine

Pick Kernel.

Support Vector Machine

Pick Kernel.

Run algorithm that:

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:

Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.

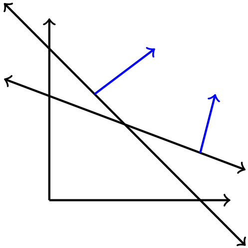
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



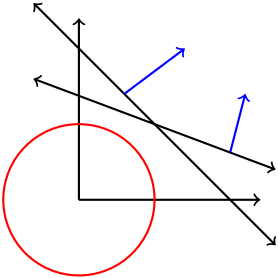
Support Vector Machine

Pick Kernel.

Run algorithm that:
  (1) Uses dot products.
  (2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
  $\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



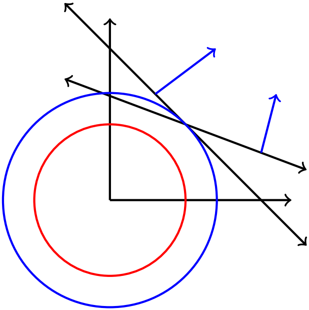
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



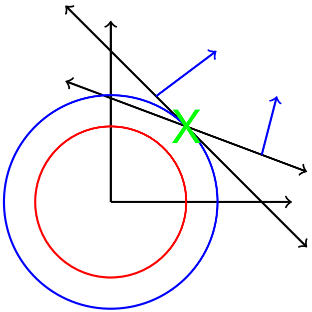
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



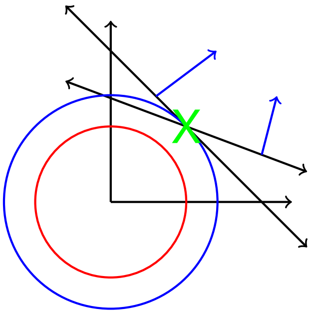
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i\ w \cdot x_i \geq 1$.



Algorithms output:

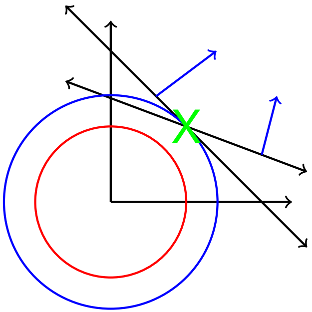
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

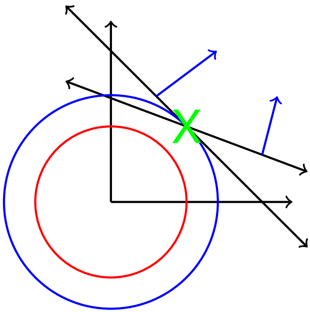
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

Solution is linear combination of hyperplanes

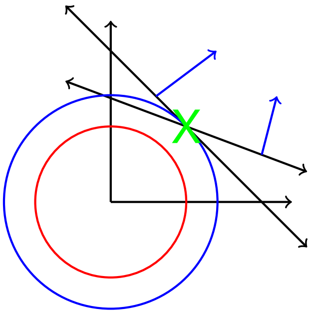
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \ w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

Solution is linear combination of hyperplanes
$w = \alpha_1 x_1 + \alpha_2 x_2 + \cdots$.

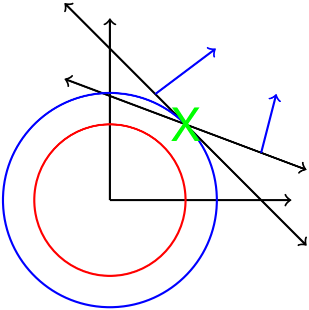
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

Solution is linear combination of hyperplanes
$w = \alpha_1 x_1 + \alpha_2 x_2 + \cdots$.
With Kernel: $\phi(\cdot)$

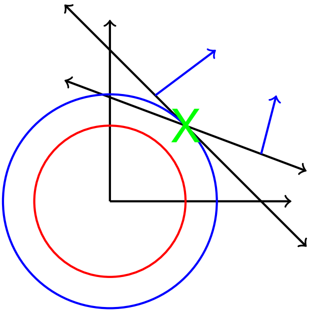
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

Solution is linear combination of hyperplanes
$w = \alpha_1 x_1 + \alpha_2 x_2 + \cdots$.

With Kernel: $\phi(\cdot)$
Problem is to find α_i where

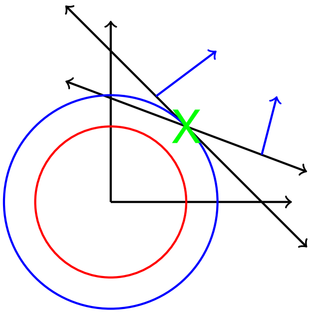
Support Vector Machine

Pick Kernel.

Run algorithm that:
(1) Uses dot products.
(2) Outputs hyperplane that is linear combination of points.

Perceptron.

Max Margin Problem as Convex optimization:
$\min |w|^2$ where $\forall i \; w \cdot x_i \geq 1$.



Algorithms output: tight hyperplanes!

Solution is linear combination of hyperplanes
$w = \alpha_1 x_1 + \alpha_2 x_2 + \cdots$.
With Kernel: $\phi(\cdot)$
Problem is to find α_i where
$\forall i (\sum_j \alpha_j \phi(x_j)) \cdot \phi(x_i) \geq 1$

Lagrange Multipliers.

Lagrangian Dual.

Find x, subject to

Lagrangian Dual.

Find x, subject to

$$f_i(x) \leq 0, i = 1, \ldots m.$$

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian:

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x, \lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x, \lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

(C) non-positive for any valid λ.

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \dots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

(C) non-positive for any valid λ.

If $\exists \lambda \geq 0$, where $L(x,\lambda)$ is positive for all x

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

(C) non-positive for any valid λ.

If $\exists \lambda \geq 0$, where $L(x,\lambda)$ is positive for all x

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \dots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

(C) non-positive for any valid λ.

If $\exists \lambda \geq 0$, where $L(x,\lambda)$ is positive for all x

(A) there is no feasible x.

Lagrangian Dual.

Find x, subject to

$f_i(x) \leq 0, i = 1, \ldots m.$

Remember calculus (constrained optimization.)

Lagrangian: $L(x,\lambda) = \sum_{i=1}^{m} \lambda_i f_i(x)$

$\lambda_i \geq 0$ - Lagrangian multiplier for inequality i.

For feasible solution x, $L(x,\lambda)$ is

(A) non-negative in expectation

(B) positive for any λ.

(C) non-positive for any valid λ.

If $\exists \lambda \geq 0$, where $L(x,\lambda)$ is positive for all x

(A) there is no feasible x.

(B) there is no x, λ with $L(x,\lambda) < 0$.

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, & \qquad i = 1, ..., m \end{aligned}$$

Lagrangian function:

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian function:

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1,...,m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$
For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$

For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$

Maximizing λ, only positive λ_i when $f_i(x) = 0$

Lagrangian:constrained optimization.

$$\begin{aligned} &\min \quad f(x) \\ &\text{subject to } f_i(x) \le 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \le 0$
For all $\lambda \ge 0$ have $L(x,\lambda) \le v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x,\lambda) \ge f(x)$

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$

For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$

Maximizing λ, only positive λ_i when $f_i(x) = 0$

which implies $L(x,\lambda) \geq f(x) = v$

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$
For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x,\lambda) \geq f(x) = v$

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, & \qquad i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$
For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x,\lambda) \geq f(x) = v$

If there is λ with $L(x,\lambda) \geq \alpha$ for all x

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian function:

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$
For all $\lambda \geq 0$ have $L(x, \lambda) \leq v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x, \lambda) \geq f(x) = v$

If there is λ with $L(x, \lambda) \geq \alpha$ for all x
Optimum value of program is at least α

Lagrangian:constrained optimization.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \le 0, \quad & i = 1, ..., m \end{aligned}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \le 0$
For all $\lambda \ge 0$ have $L(x,\lambda) \le v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x,\lambda) \ge f(x) = v$

If there is λ with $L(x,\lambda) \ge \alpha$ for all x
Optimum value of program is at least α

Primal problem:
x, that minimizes $L(x,\lambda)$ over all $\lambda \ge 0$.

Lagrangian:constrained optimization.

$$\begin{array}{rll} \min & f(x) & \\ \text{subject to } f_i(x) \leq 0, & & i = 1, ..., m \end{array}$$

Lagrangian function:

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

If (primal) x has value v $f(x) = v$ and all $f_i(x) \leq 0$
For all $\lambda \geq 0$ have $L(x,\lambda) \leq v$
Maximizing λ, only positive λ_i when $f_i(x) = 0$
which implies $L(x,\lambda) \geq f(x) = v$

If there is λ with $L(x,\lambda) \geq \alpha$ for all x
Optimum value of program is at least α

Primal problem:
x, that minimizes $L(x,\lambda)$ over all $\lambda \geq 0$.

Dual problem:
λ, that maximizes $L(x,\lambda)$ over all x.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{array}{ll} \min & f(x) \\ \text{subject to } f_i(x) \leq 0, & i = 1, ..., m \end{array}$$

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & \quad i = 1, ..., m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} &\min \quad f(x) \\ &\text{subject to } f_i(x) \leq 0, \qquad i = 1, ..., m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } & f_i(x) \leq 0, \qquad i = 1, ..., m \end{aligned}$$

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Feasible primal, $f_i(x^*) \leq 0$, and feasible dual $\lambda_i \geq 0$.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, ..., m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Feasible primal, $f_i(x^*) \leq 0$, and feasible dual $\lambda_i \geq 0$.

Complementary slackness: $\lambda_i f_i(x^*) = 0$.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \leq 0, \quad & i = 1, \ldots, m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Feasible primal, $f_i(x^*) \leq 0$, and feasible dual $\lambda_i \geq 0$.

Complementary slackness: $\lambda_i f_i(x^*) = 0$.

Launched nonlinear programming!

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{array}{ll} \min & f(x) \\ \text{subject to } f_i(x) \leq 0, & i = 1, \ldots, m \end{array}$$

$L(x,\lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Feasible primal, $f_i(x^*) \leq 0$, and feasible dual $\lambda_i \geq 0$.

Complementary slackness: $\lambda_i f_i(x^*) = 0$.

Launched nonlinear programming! See paper.

Why important: KKT.

Karash, Kuhn and Tucker Conditions.

$$\begin{aligned} \min \quad & f(x) \\ \text{subject to } f_i(x) \le 0, \quad & i = 1, ..., m \end{aligned}$$

$L(x, \lambda) = f(x) + \sum_{i=1}^{m} \lambda_i f_i(x)$

Local minima for feasible x^*.

There exist multipliers λ, where

$\nabla f(x^*) + \sum_i \lambda_i \nabla f_i(x^*) = 0$

Feasible primal, $f_i(x^*) \le 0$, and feasible dual $\lambda_i \ge 0$.

Complementary slackness: $\lambda_i f_i(x^*) = 0$.

Launched nonlinear programming! See paper.

Linear Program.

$\min cx, Ax \geq b$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, \ldots, m \end{aligned}$$

Lagrangian (Dual):

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, \ldots, m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad\qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Linear Program.

$\min cx, Ax \geq b$

$$\begin{array}{rll} \min & c \cdot x & \\ \text{subject to } & b_i - a_i \cdot x \leq 0, & i = 1, ..., m \end{array}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x).$

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda.$

Best λ?

$\max b \cdot \lambda$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

$\max b \cdot \lambda$ where $a_j \lambda = c_j$.

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

$\max b \cdot \lambda$ where $a_j \lambda = c_j$.

$\max b\lambda,$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

$\max b \cdot \lambda$ where $a_j \lambda = c_j$.

$\max b\lambda, \lambda^T A = c,$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

$\max b \cdot \lambda$ where $a_j \lambda = c_j$.

$\max b\lambda, \lambda^T A = c, \lambda \geq 0$

Linear Program.

$\min cx, Ax \geq b$

$$\begin{aligned} \min \quad & c \cdot x \\ \text{subject to } & b_i - a_i \cdot x \leq 0, \qquad i = 1, ..., m \end{aligned}$$

Lagrangian (Dual):

$L(\lambda, x) = cx + \sum_i \lambda_i (b_i - a_i x)$.

or

$L(\lambda, x) = -(\sum_j x_j (a_j \lambda - c_j)) + b\lambda$.

Best λ?

$\max b \cdot \lambda$ where $a_j \lambda = c_j$.

$\max b\lambda, \lambda^T A = c, \lambda \geq 0$

Duals!