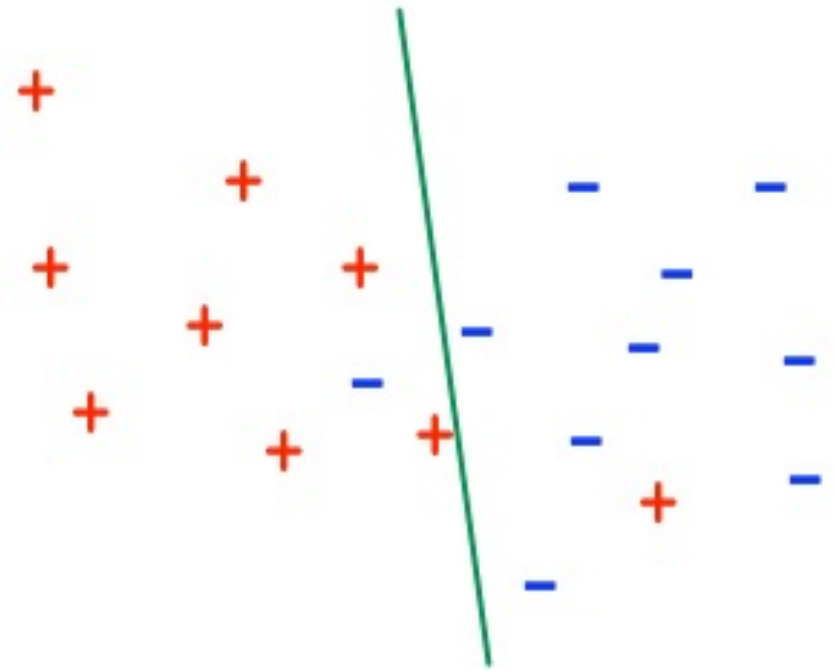
# More linear classification

# The decision boundary



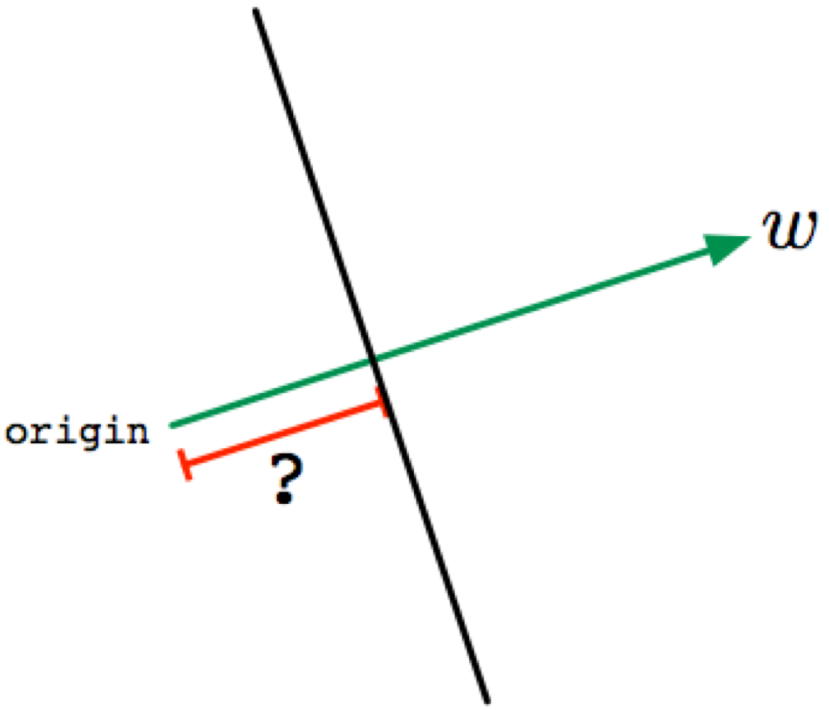
Decision boundary in $R^p$ is a **hyperplane**.

- How is this boundary parametrized?
- How can we learn a hyperplane from training data?

# Hyperplanes

Hyperplane $\{x : w \cdot x = b\}$

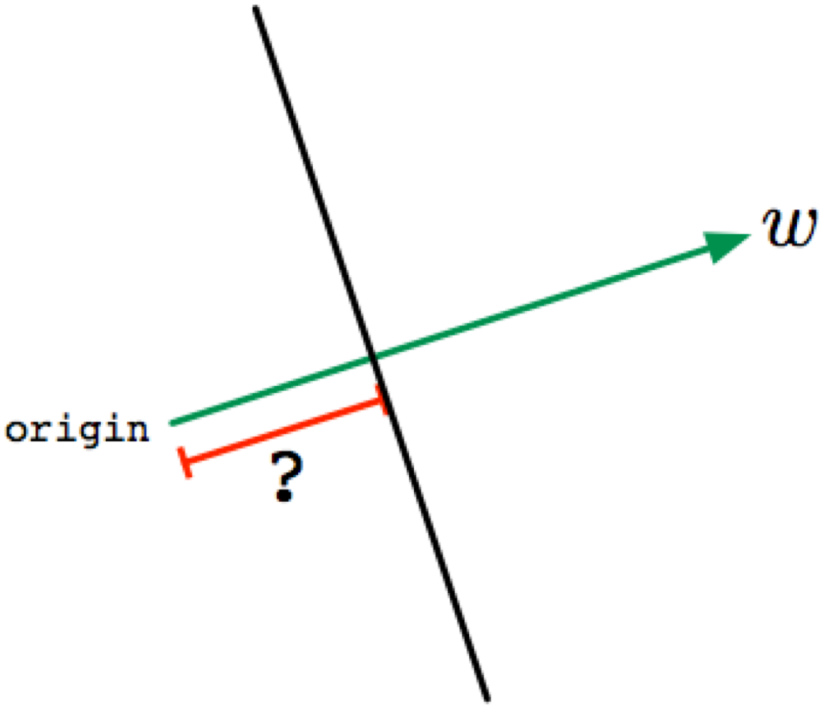
- orientation $w \in \mathrm{R}^p$
- offset $b \in \mathrm{R}$



# Hyperplanes

Hyperplane $\{x : w \cdot x = b\}$

- orientation $w \in \mathrm{R}^p$
- offset $b \in \mathrm{R}$




Can always normalize $w$ to unit length:

$$(w, b) \quad \longleftrightarrow \quad \left(\widehat{w} = \frac{w}{\|w\|}, \frac{b}{\|w\|}\right)$$

$$w \cdot x = b \quad \longleftrightarrow \quad \widehat{w} \cdot x = \frac{b}{\|w\|}$$

Equivalently: all points whose projection onto $\widehat{w}$ is $b/\|w\|$.

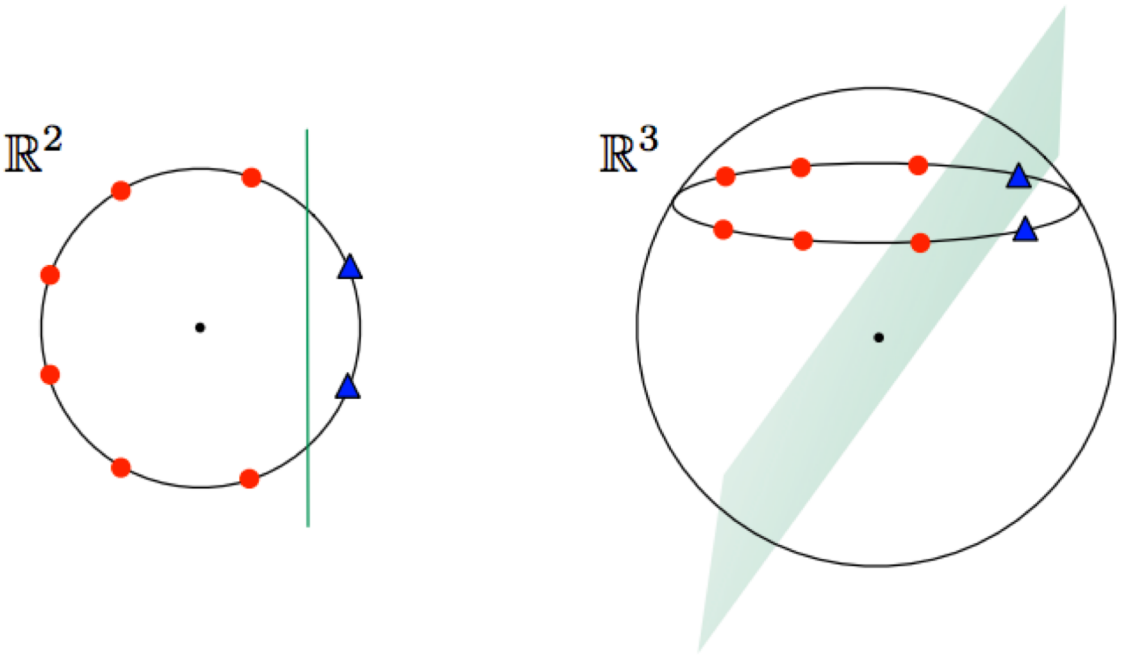
# Homogeneous linear separators

Hyperplanes that pass through the origin have no offset, $b = 0$.

Reduce to this case by adding an extra feature to $x$ :

$$\tilde{x} = (x, 1) \in \mathbb{R}^{p+1}$$

Then $\{x : w \cdot x = b\} \equiv \{x : \tilde{w} \cdot \tilde{x} = 0\}$ where $\tilde{w} = (w, -b)$.



# The learning problem: separable case

*Input:* training data $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^p \times \{-1, +1\}$

*Output:* linear classifier $w \in \mathbb{R}^p$ such that

$$y^{(i)}(w \cdot x^{(i)}) > 0 \quad \text{for } i = 1, 2, \ldots, n$$

This is linear programming:

- Each data point is a linear constraint on $w$
- Want to find $w$ that satisfies all these constraints

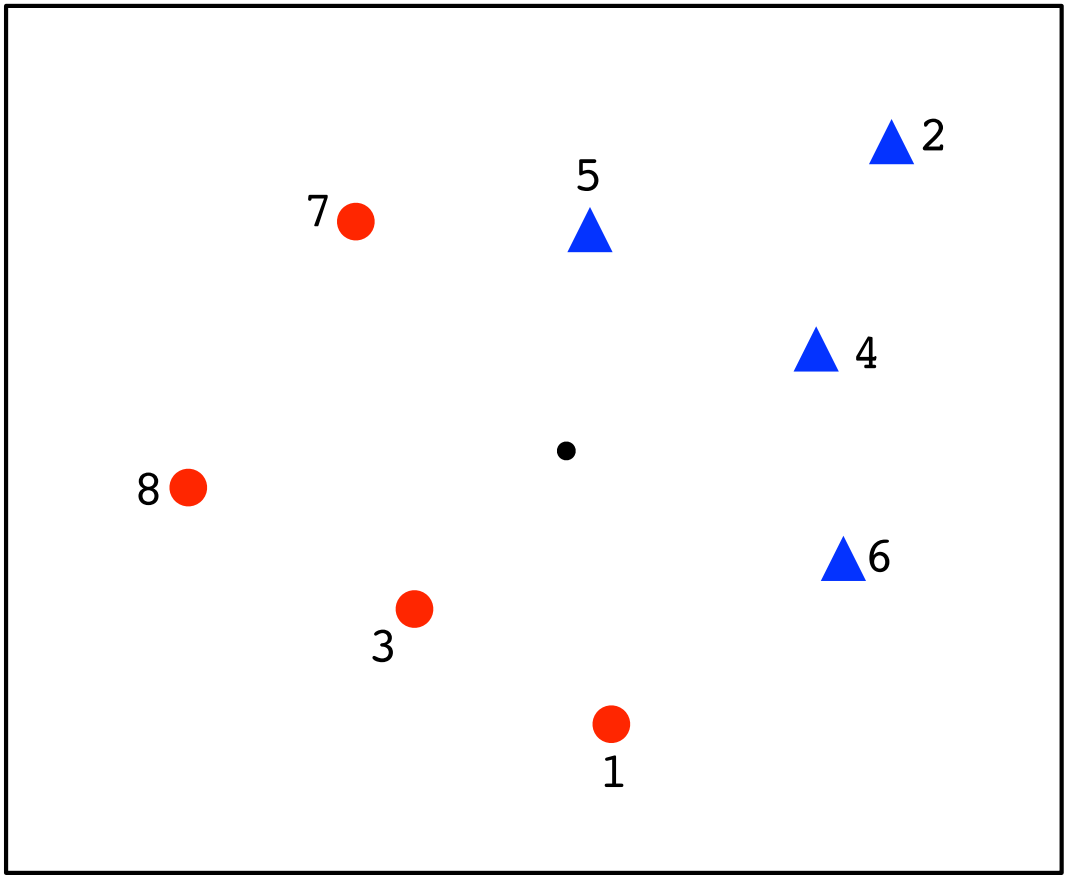
But we won't use generic linear programming methods, such as simplex.

A simple alternative: **Perceptron algorithm** (Rosenblatt, 1958)

- $w = 0$
- while some $(x, y)$ is misclassified:
  - $w = w + yx$

# Perceptron: example

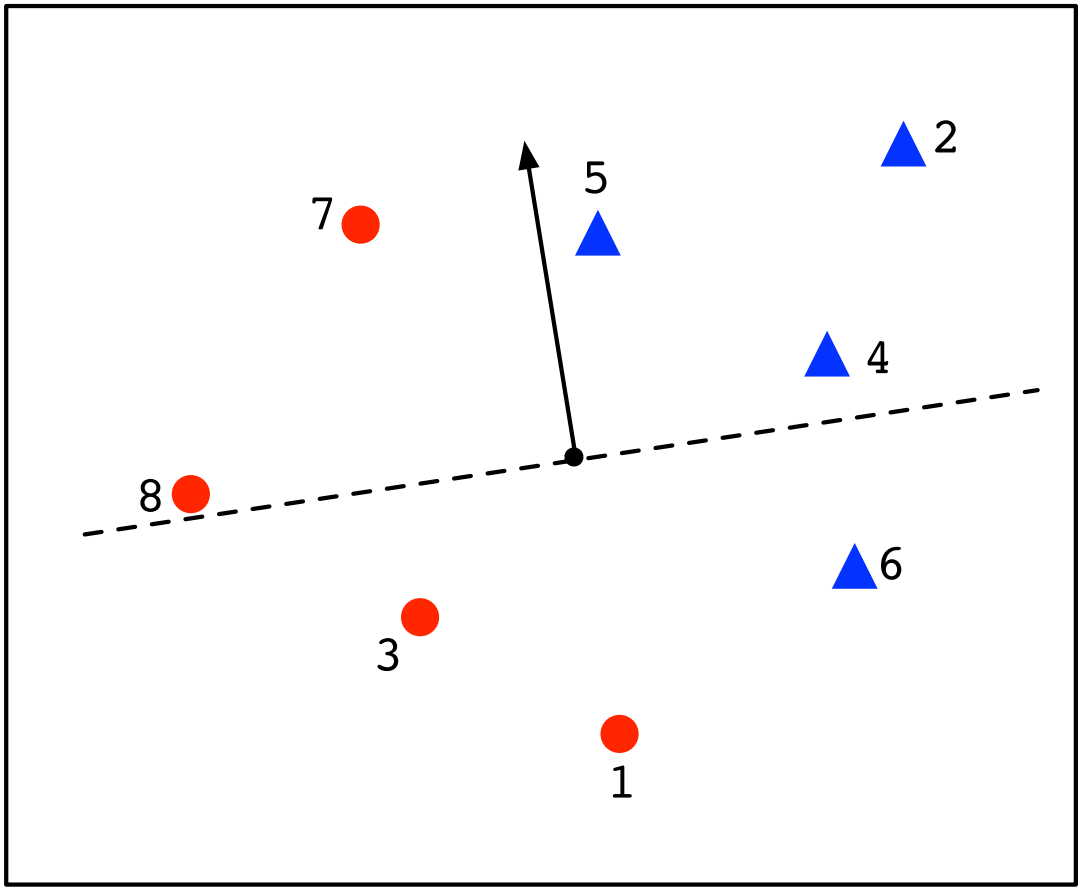
- $w = 0$
- while some $(x, y)$ is misclassified:
  - $w = w + yx$




**Separator:** $w = 0$

# Perceptron: example

- $w = 0$
- while some $(x, y)$ is misclassified:
  - $w = w + yx$

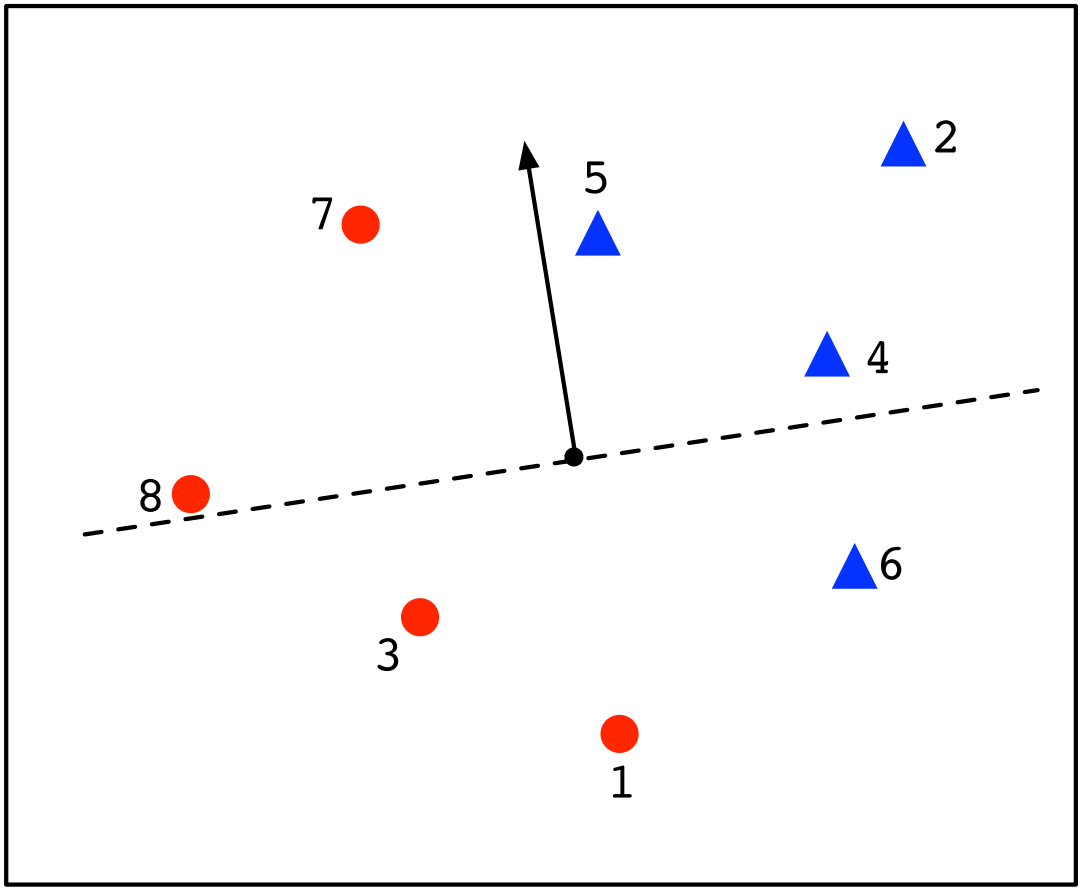



**Separator:** $w = -x^{(1)}$

# Perceptron: example

- $w = 0$
- while some ($x$ , $y$ ) is misclassified:
  - $w = w + yx$

Points 2-5 are correct



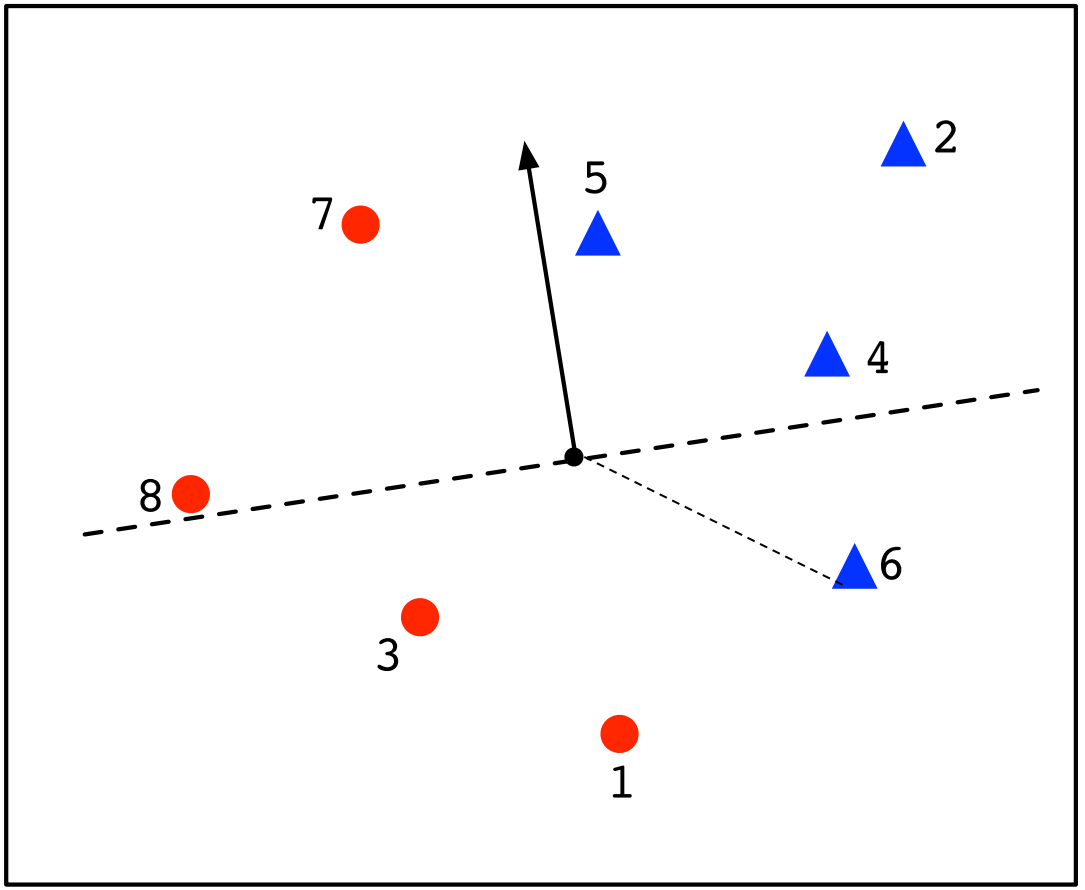

**Separator:** $w = -x^{(1)}$

# Perceptron: example

- $w = 0$
- while some ($x$ , $y$ ) is misclassified:
  - $w = w + yx$

Six is misclassified

→ Add vector in direction of 6



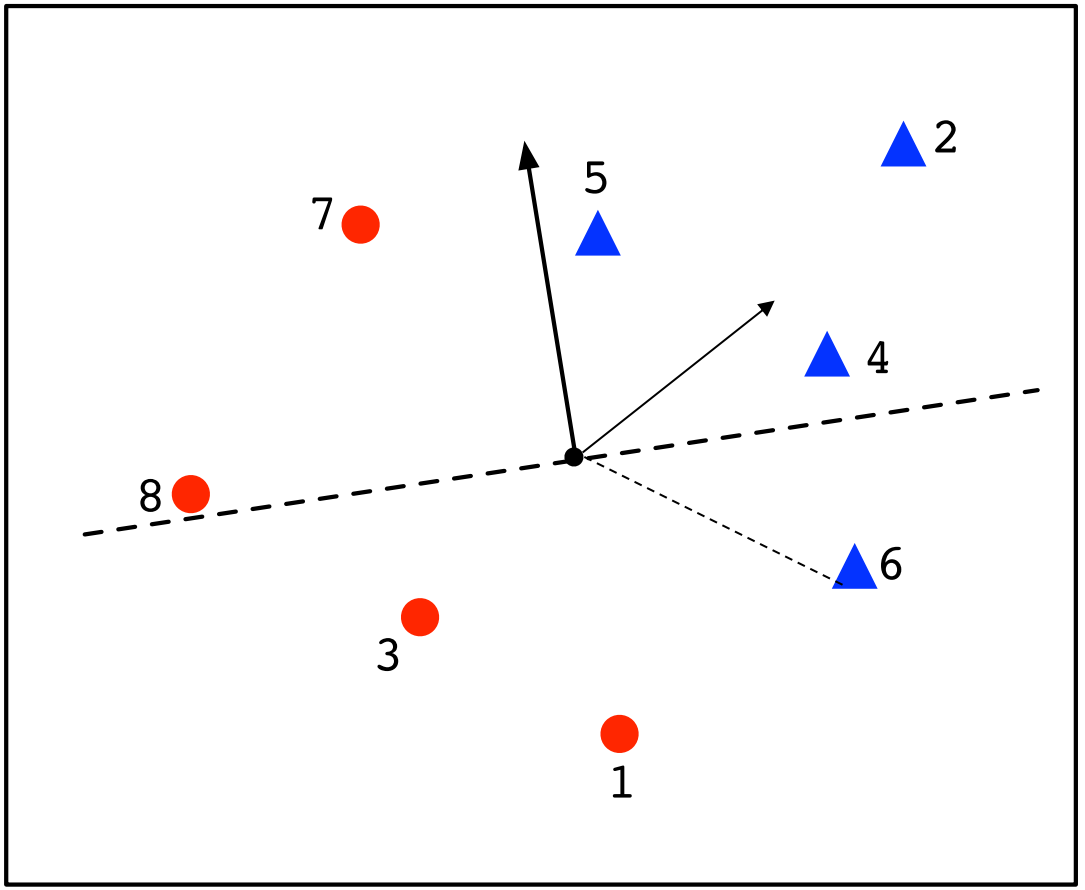

**Separator:** $w = -x^{(1)}$

# Perceptron: example

- $w = 0$
- while some ($x$ , $y$ ) is misclassified:
  - $w = w + yx$

Six is misclassified

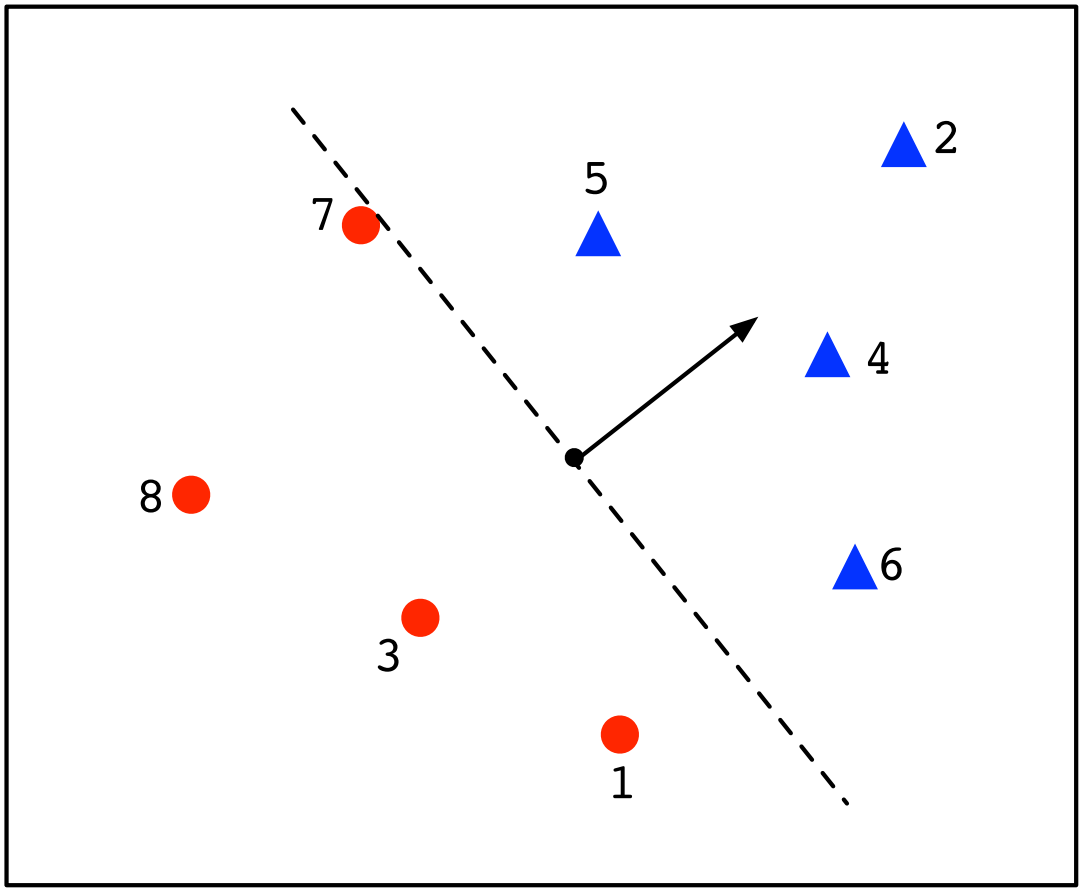
→ Add vector in direction of 6




**Separator:** $w = -x^{(1)}$

# Perceptron: example

- $w = 0$
- while some ($x$ , $y$ ) is misclassified:
  - $w = w + yx$




**Separator:** $w = 0 w = -x^{(1)} w = -x^{(1)} + x^{(6)}$

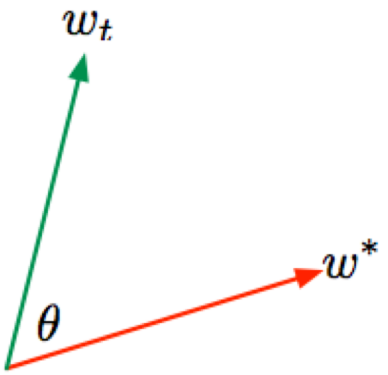
# Perceptron: convergence

**Theorem:** Let $R = \max \|x^{(i)}\|$. Suppose there is a unit vector $w^*$ and some (margin) $\gamma > 0$ such that

$$y^{(i)}(w^* \cdot x^{(i)}) \geq \gamma \quad \text{for all } i.$$

Then the Perceptron algorithm converges after at most $R^2/\gamma^2$ updates.

**Proof idea.** Let $w_t$ be the classifier after $t$ updates.




**Track angle between $w_t$ and $w^*$:**

$$\cos(\angle(w_t, w^*)) = \frac{w_t \cdot w^*}{\|w\|}.$$

On each mistake, when $w_t$ is updated to $w_{t+1}$,

- $w_t \cdot w^*$ grows significantly.
- $\|w_t\|$ does not grow much.

# Perceptron convergence, cont'd

Perceptron update: if $y(w_t \cdot x) < 0$ (misclassified) then $w_{t+1} = w_t + yx$.
Target vector $w^*$ has unit length, and margin condition $y(w^* \cdot x) \geq \gamma$.

1. Initial vector $w_0 = 0$.
2. When updating $w_t$ to $w_{t+1}$:

$$w_{t+1} \cdot w^* = (w_t + yx) \cdot w^* = w_t \cdot w^* + y(w^* \cdot x) \geq w_t \cdot w^* + \gamma$$
$$\|w_{t+1}\|^2 = \|w_t + yx\|^2 = \|w_t\|^2 + \|x\|^2 + 2y(w_t \cdot x) \leq \|w_t\|^2 + R^2$$

3. After $T$ updates, we have

$$w_T \cdot w^* \geq T\gamma$$
$$\|w_T\|^2 \leq TR^2$$

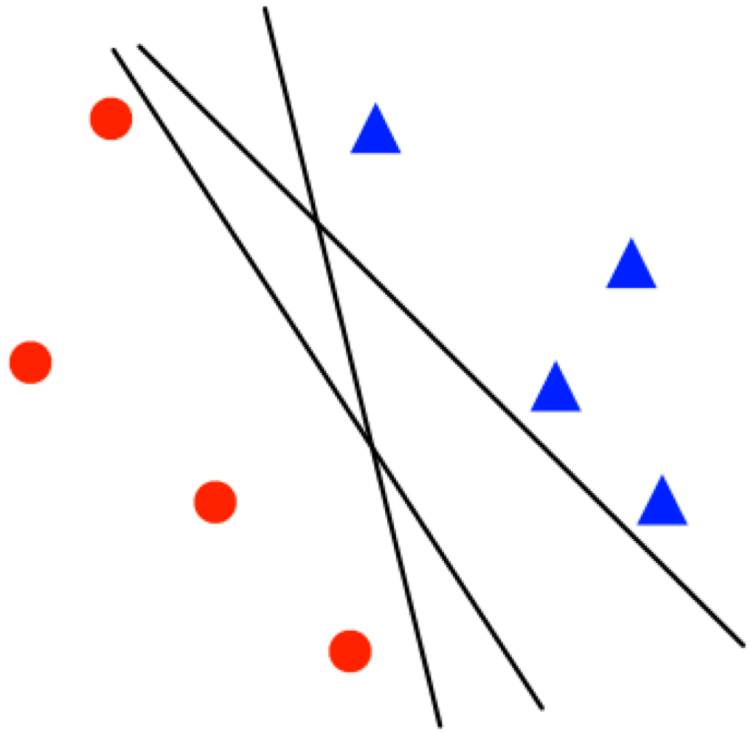
4. The angle between $w_T$ and $w^*$ is given by

$$\cos(\angle(w_T, w^*)) = \frac{w_T \cdot w^*}{\|w\|} \geq \frac{T\gamma}{R\sqrt{T}}.$$

**This is at most 1, so $T \leq R^2/\gamma^2$.**

# A better separator?

For a linearly separable data set, there are in general many possible separating hyperplanes, and Perceptron is guaranteed to find one of them.



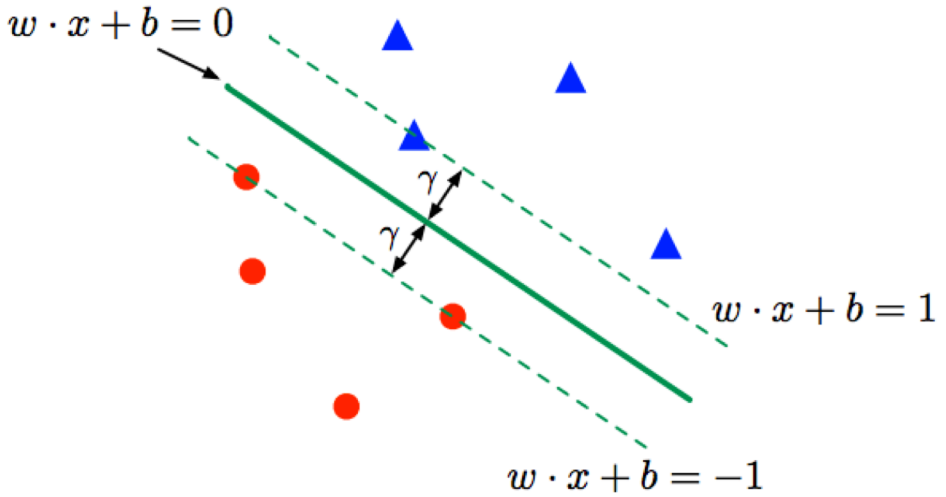
But is there a better, more systematic choice of separator? The one with the most buffer around it, for instance?

# Maximizing the margin

Given training data $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^p \times \{-1, +1\}$, find $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$ such that $y^{(i)}(w \cdot x^{(i)} + b) > 0$ for all $i$.

By scaling $w, b$, can equally ask for

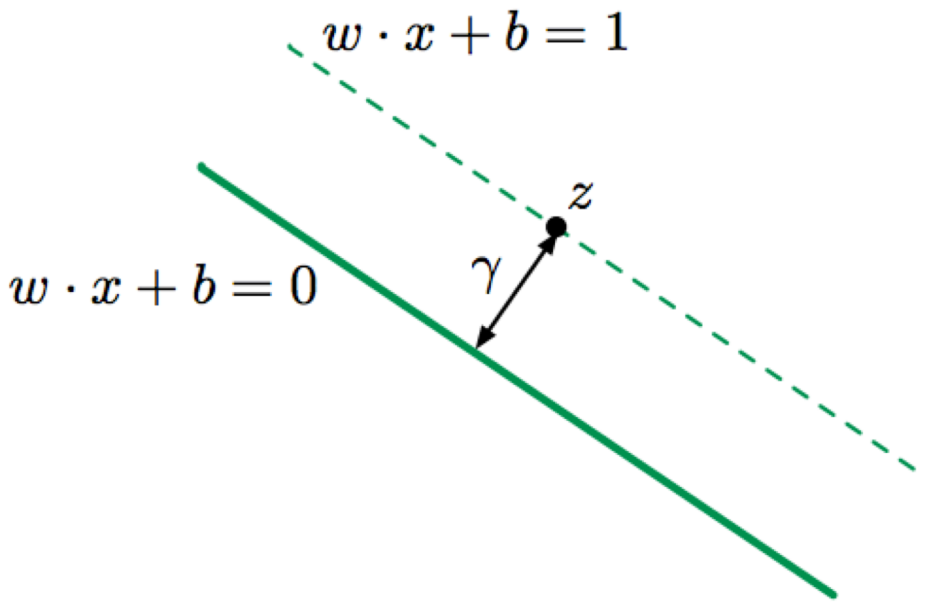
$$y^{(i)}(w \cdot x^{(i)} + b) \geq 1 \quad \text{for all } i.$$




Maximize the **margin** $\gamma$.

# What is the margin?

Close-up of a point $z$ on the positive boundary.




Let $\widehat{w}$ be the unit vector in the direction of $w$, i.e. $\widehat{w} = w/\|w\|$.
Then $z - \gamma\widehat{w}$ is on the separator, so

$$w \cdot (z - \gamma\widehat{w}) + b = 0 \quad \Rightarrow \quad \gamma w \cdot \widehat{w} = w \cdot z + b = 1 \quad \Rightarrow \quad \gamma = 1/\|w\|$$

In short: to maximize the margin, minimize $\|w\|$.

# Duality

$$\begin{aligned} \min_{x,y} \quad & px + qy \\ & x \geq 0 \\ & y \leq 1 \\ \text{subject to} \quad & 3x + y = 2 \end{aligned}$$

# Duality

$$\begin{aligned} \min_{x,y} \quad & px + qy \\ & x \geq 0 \\ & y \leq 1 \\ \text{subject to} \quad & 3x + y = 2 \end{aligned}$$

Introduce coefficients and scale all the inequalities

| | | |
|---|---|---|
| $a \geq 0$: | $x \geq 0$ | $ax \geq 0$ |
| $b \geq 0$: | $-y \geq -1$ | $-by \geq -b$ |
| c: | $3x + y - 2 = 0$ | $c(3x + y - 2) = 0$ |

# Duality

## Add these equations

$$x(a+3c)+y(-b+c)-2c \geq -b \tag{1}$$

$$x(a+3c)+y(-b+c) \geq 2c-b \tag{2}$$

# Duality

Add these equations

$$x(a+3c)+y(-b+c)-2c \geq -b \tag{1}$$

$$x(a+3c)+y(-b+c) \geq 2c-b \tag{2}$$

Compare above equation with

$$\min_{x,y} \qquad px+qy \geq 2c-b \tag{3}$$

$$a+3c=p \tag{4}$$

$$-b+c=q \tag{5}$$

# Duality

## Primary LP

$$\begin{aligned} \min_{x,y} \quad & px + qy \\ & x \geq 0 \\ & y \leq 1 \\ \text{subject to} \quad & 3x + y = 2 \end{aligned}$$

# Duality

## Primary LP

$$\begin{aligned} \min_{x,y} \quad & px + qy \\ & x \geq 0 \\ & y \leq 1 \\ \text{subject to} \quad & 3x + y = 2 \end{aligned}$$

## Dual LP

$$\begin{aligned} \max_{a,b,c} \quad & 2c - b \\ & a \geq 0 \\ & b \geq 0 \\ & a + 3c = p \\ & -b + c = q \\ \text{subject to} \quad & c : \text{unconstrained} \end{aligned}$$

# Maximum-margin linear classifier

- Given $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^p \times \{-1, +1\}$.

$$\text{(PRIMAL)} \quad \min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \frac{1}{2}\|w\|^2$$
$$\text{s.t.:} \quad y^{(i)}(w \cdot x^{(i)} + b) \geq 1 \quad \text{for all } i = 1, 2, \ldots, n$$

- This is a convex optimization problem:
  - Convex objective function
  - Linear constraints

- It has a dual maximization problem with the same optimum value.

$$\text{(DUAL)} \quad \max_{\alpha \in \mathbb{R}^n} \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)} \cdot x^{(j)})$$
$$\text{s.t.:} \quad \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$
$$\alpha \geq 0$$

# Complementary slackness

$$\text{(PRIMAL)} \qquad \min_{w \in \mathbb{R}^p, b \in \mathbb{R}} \ \frac{1}{2}\|w\|^2$$
$$\text{s.t.:} \ \ y^{(i)}(w \cdot x^{(i)} + b) \geq 1 \quad \text{for all } i = 1, 2, \ldots, n$$

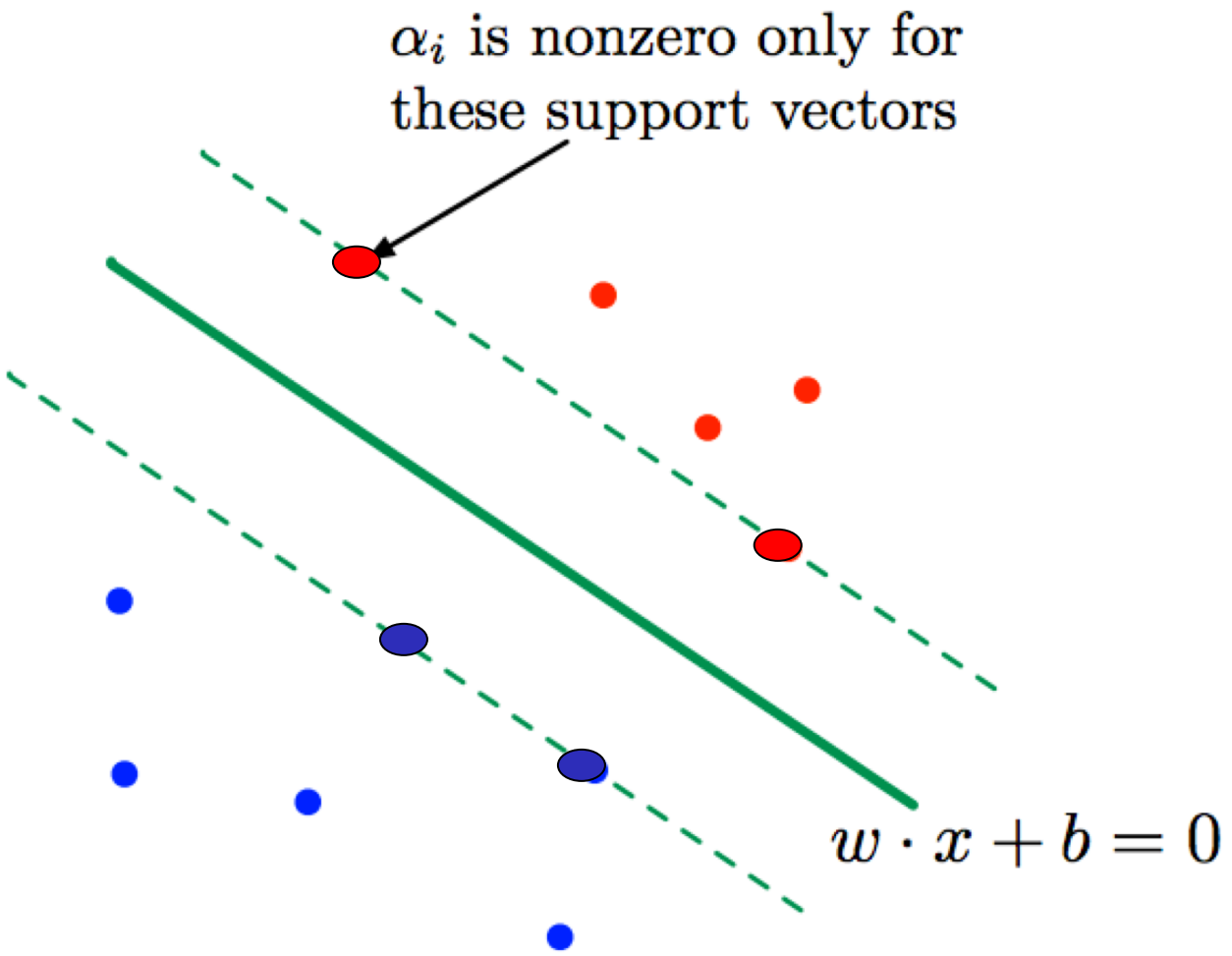
$$\text{(DUAL)} \qquad \max_{\alpha \in \mathbb{R}^n} \ \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y^{(i)} y^{(j)} (x^{(i)} \cdot x^{(j)})$$
$$\text{s.t.:} \ \ \sum_{i=1}^{n} \alpha_i y^{(i)} = 0$$
$$\alpha \geq 0$$

At optimality, $w = \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)}$ and moreover

$$\alpha_i > 0 \quad \Rightarrow \quad y^{(i)}(w \cdot x^{(i)} + b) = 1$$

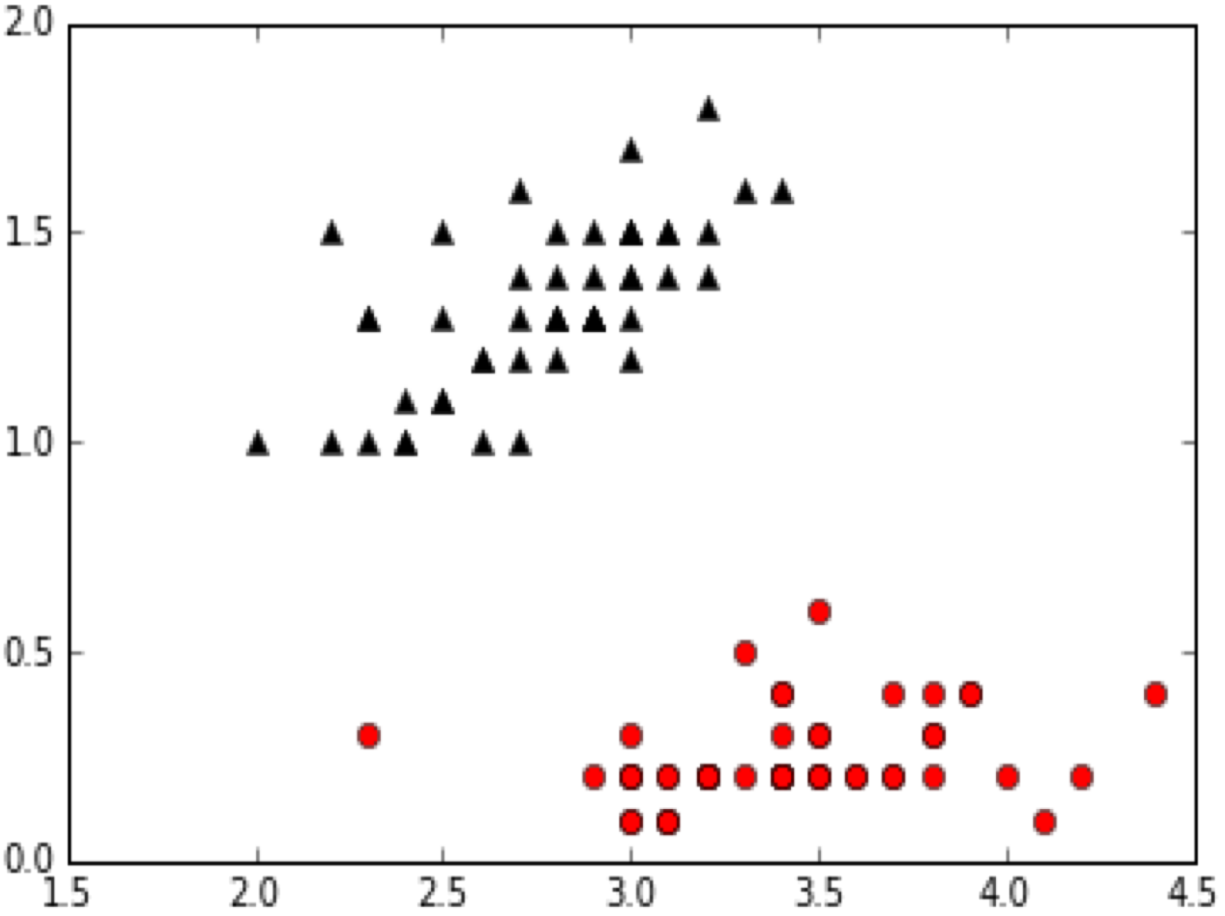
Points $x^{(i)}$ with $\alpha_i > 0$ are called **support vectors**.

# Support vectors

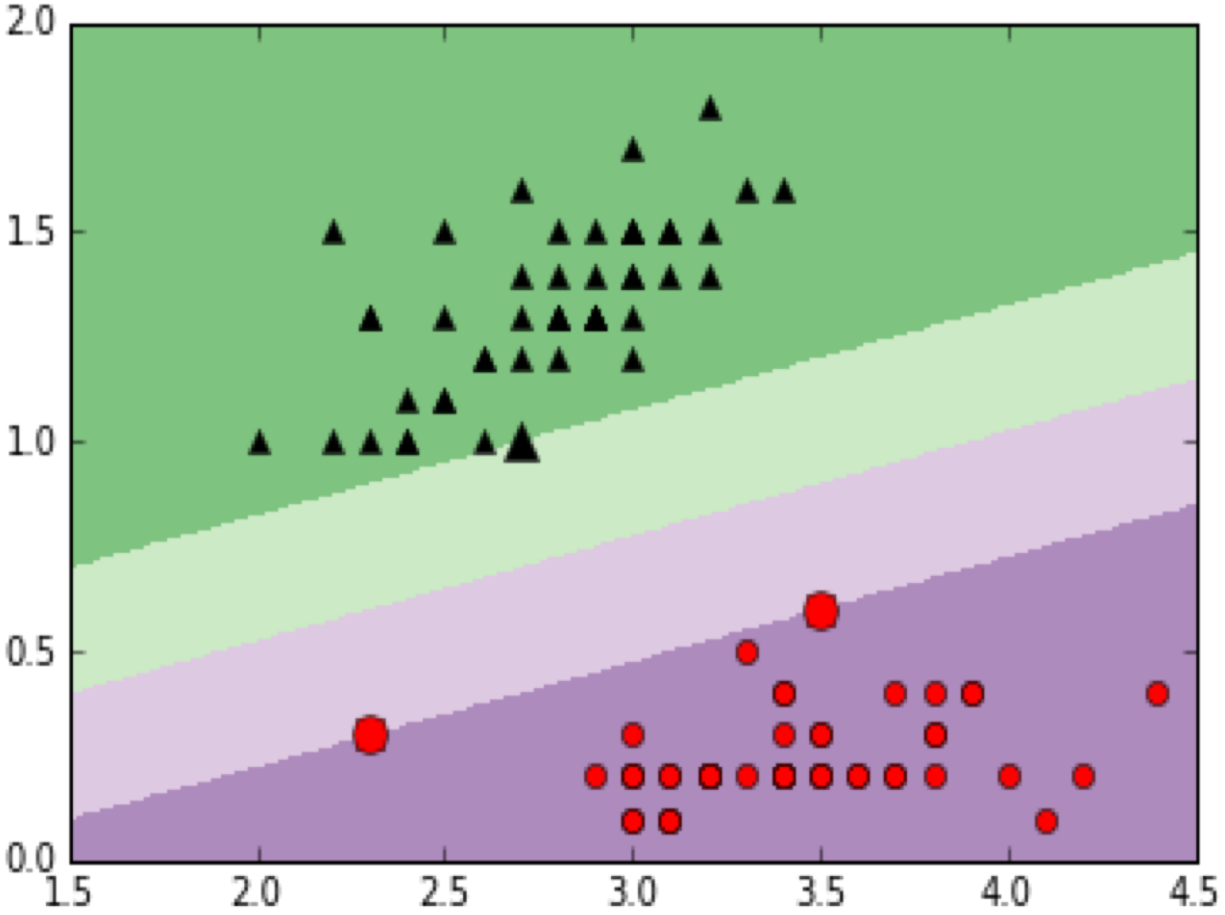



Linear classifier $w = \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)}$ is a function of just the support vectors.

# Small example: Iris data set



# Small example: Iris data set



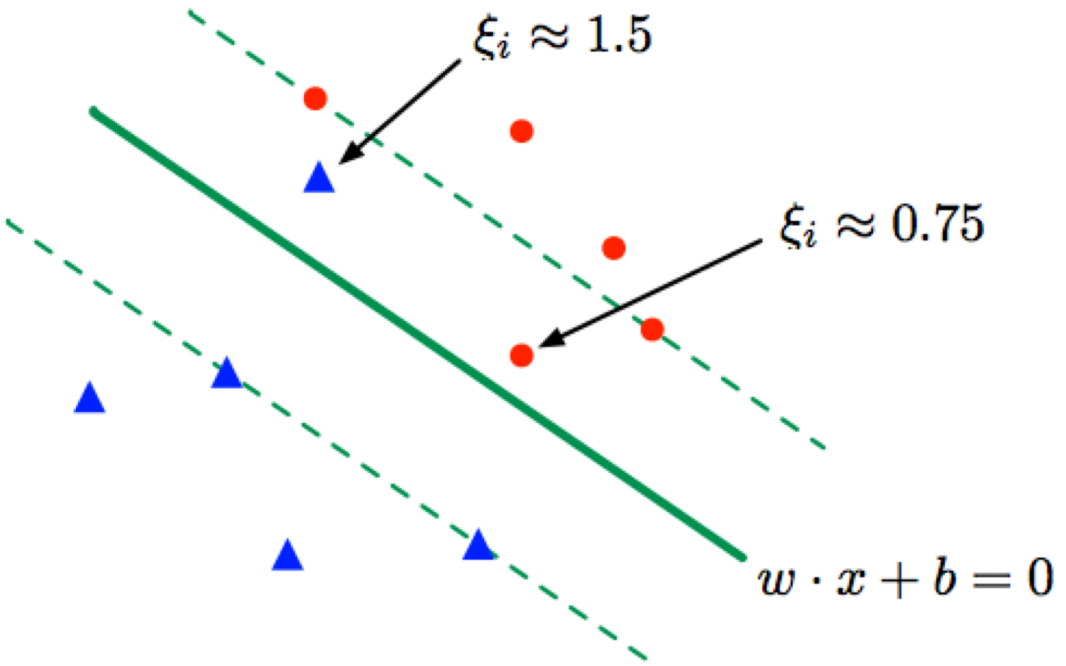
# The non-separable case

Idea: allow each data point $x^{(i)}$ some slack $\xi_i$.

$$\text{(PRIMAL)} \qquad \min_{w \in \mathbb{R}^p, b \in \mathbb{R}, \xi \in \mathbb{R}^n} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n} \xi_i$$

$$\text{s.t.:} \quad y^{(i)}(w \cdot x^{(i)} + b) \geq 1 - \xi_i \quad \text{for all } i = 1, 2, \ldots, n$$

$$\xi \geq 0$$



# Dual for general case

$$\text{(PRIMAL)} \qquad \min_{w\in\mathbb{R}^p, b\in\mathbb{R}, \xi\in\mathbb{R}^n} \ \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{s.t.:} \quad y^{(i)}(w\cdot x^{(i)} + b) \geq 1-\xi_i \quad \text{for all } i = 1, 2, \ldots, n$$

$$\xi \geq 0$$

$$\text{(DUAL)} \qquad \max_{\alpha\in\mathbb{R}^n} \ \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y^{(i)}y^{(j)}(x^{(i)}\cdot x^{(j)})$$

$$\text{s.t.:} \quad \sum_{i=1}^{n}\alpha_i y^{(i)} = 0$$

$$0 \leq \alpha_i \leq C$$

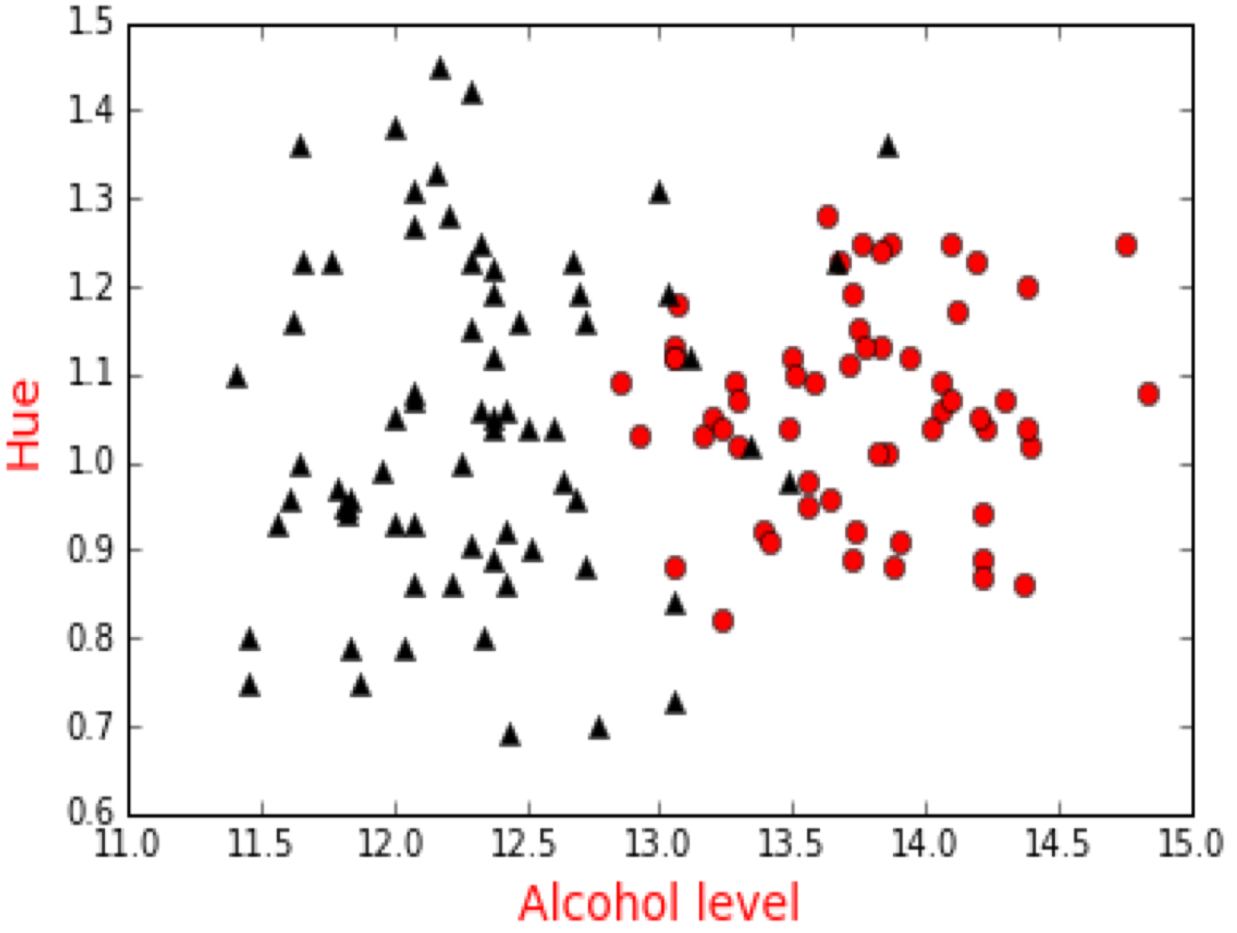
At optimality, $w = \sum_i \alpha_i y^{(i)} x^{(i)}$, with

$$0 < \alpha_i < C \quad \Rightarrow \quad y^{(i)}(w\cdot x^{(i)} + b) = 1$$

$$\alpha_i = C \quad \Rightarrow \quad y^{(i)}(w\cdot x^{(i)} + b) = 1 - \xi_i$$

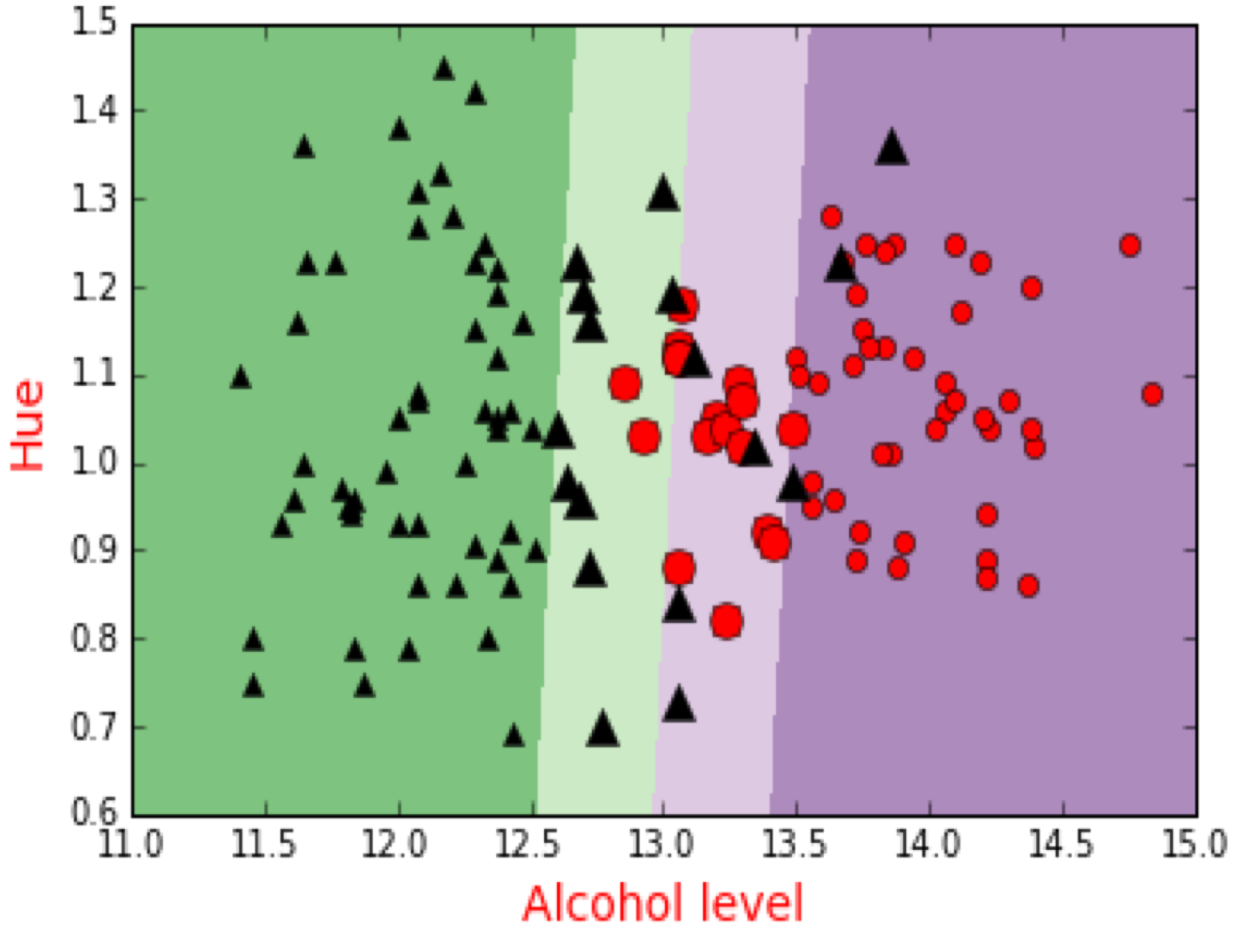
# Wine data set

Here $C = 1.0$



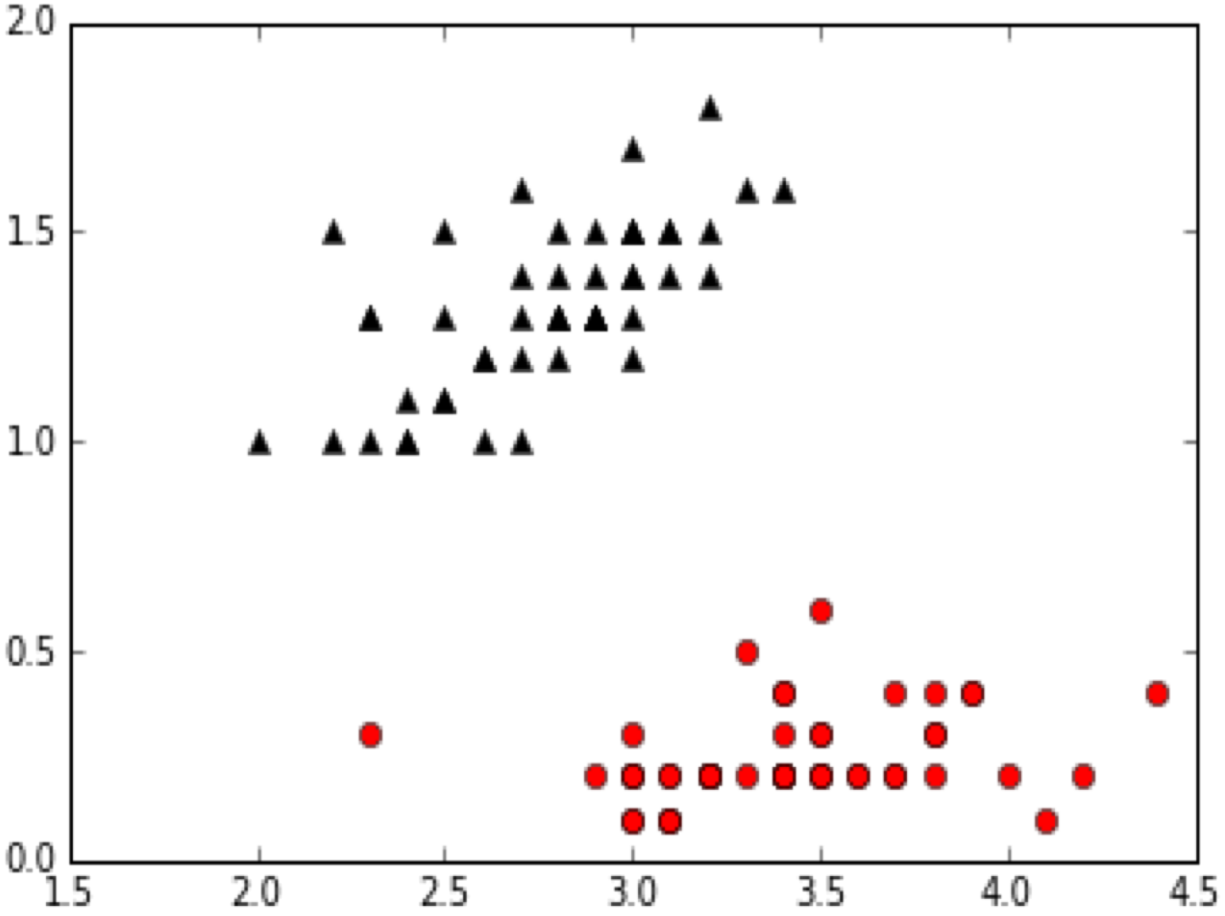
# Wine data set

Here $C = 1.0$



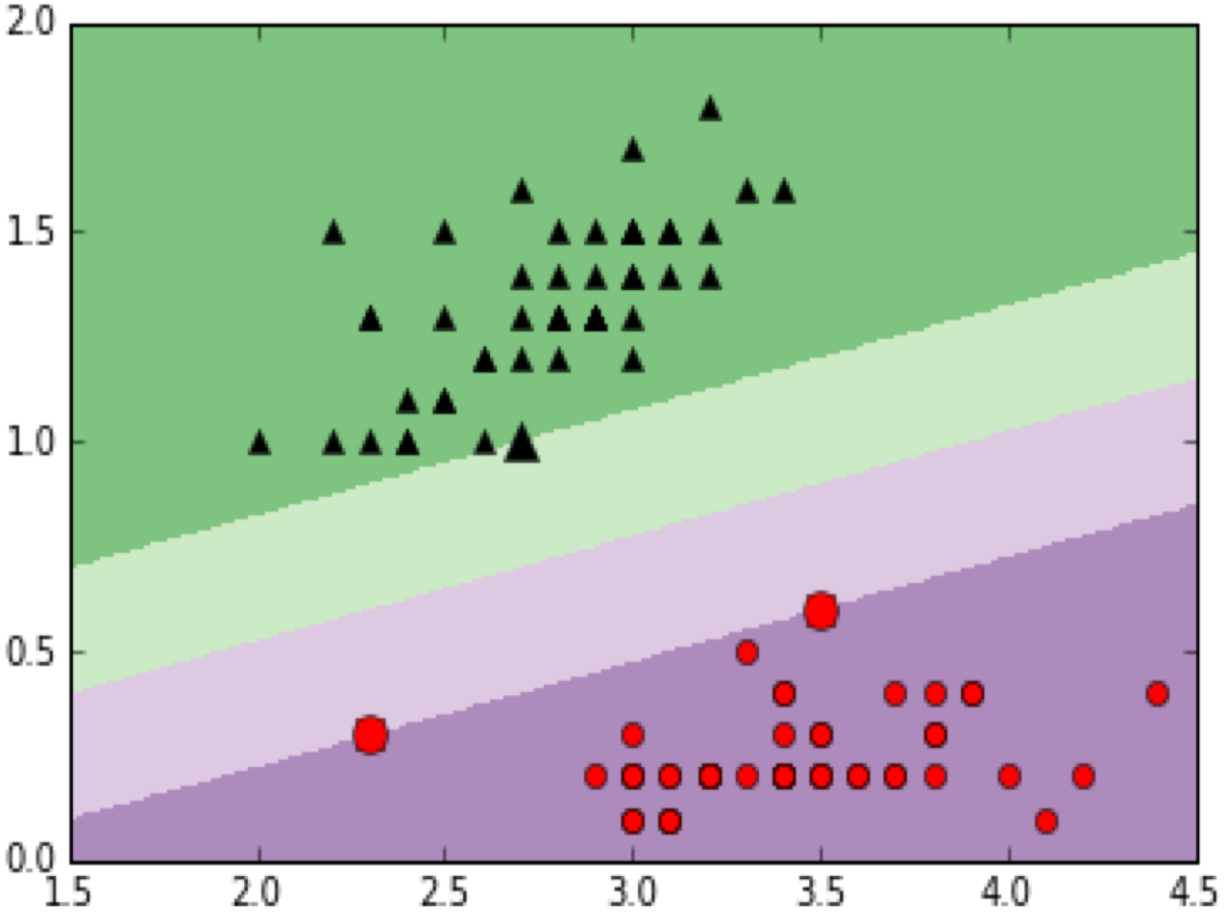
# Back to Iris

$C = 10$



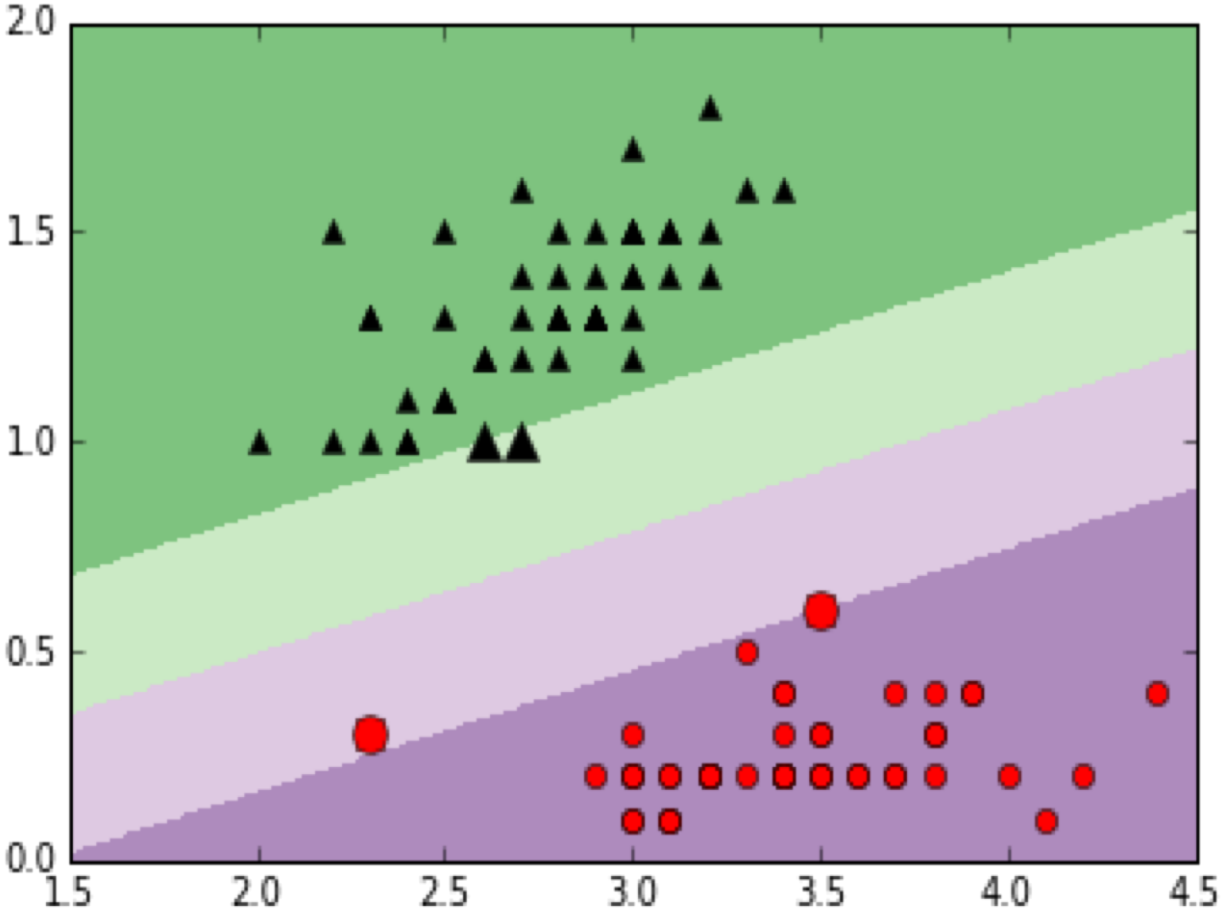
# Back to Iris

$C = 10$



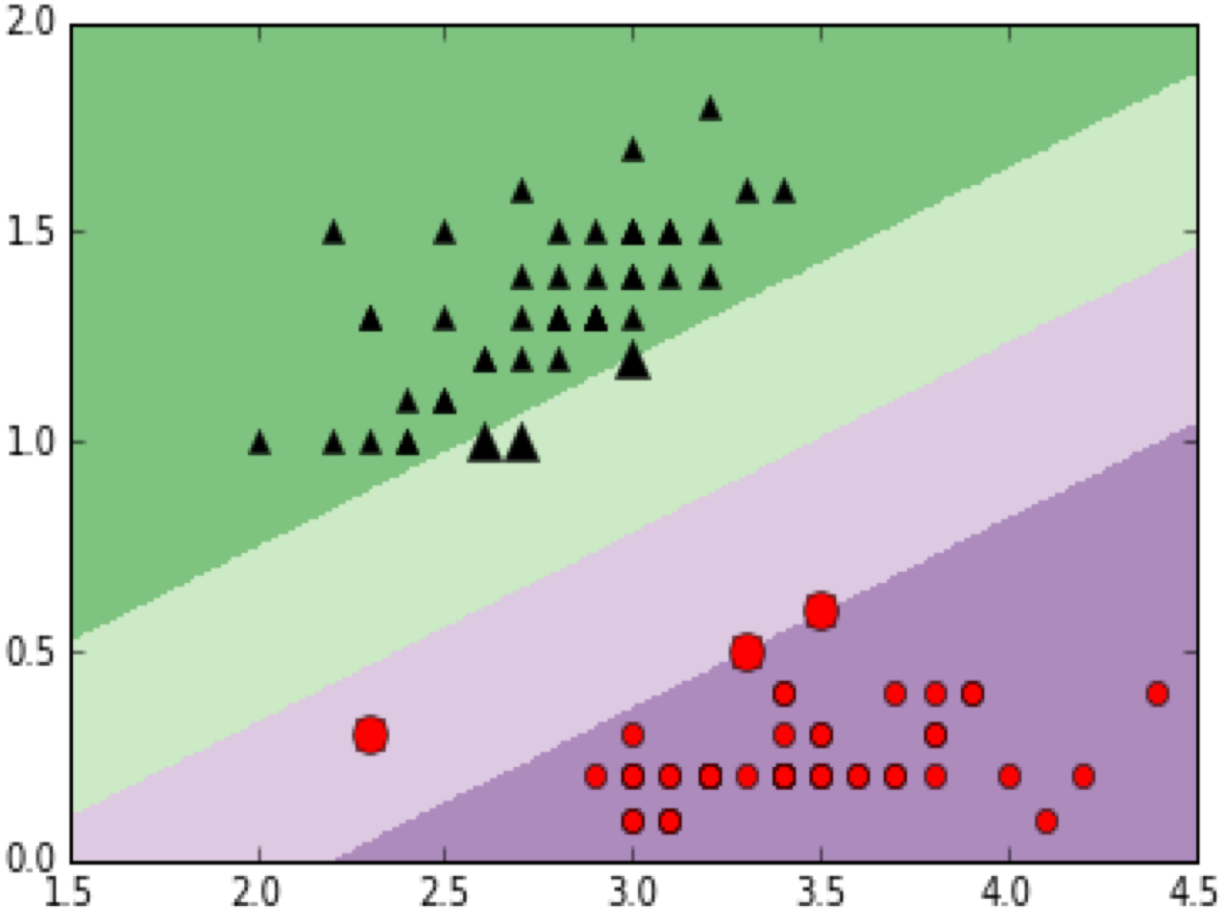
# Back to Iris

$C = 3$



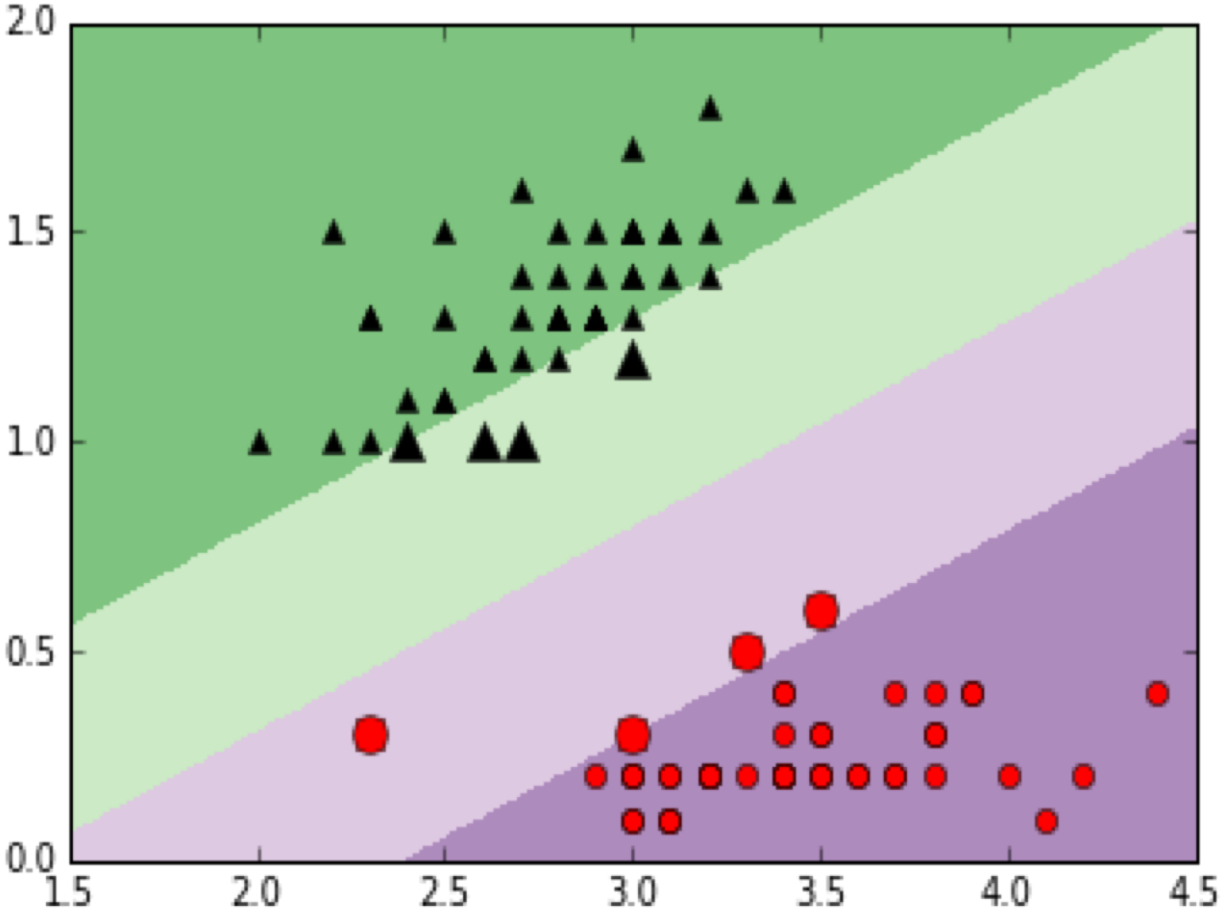
# Back to Iris

$C = 2$



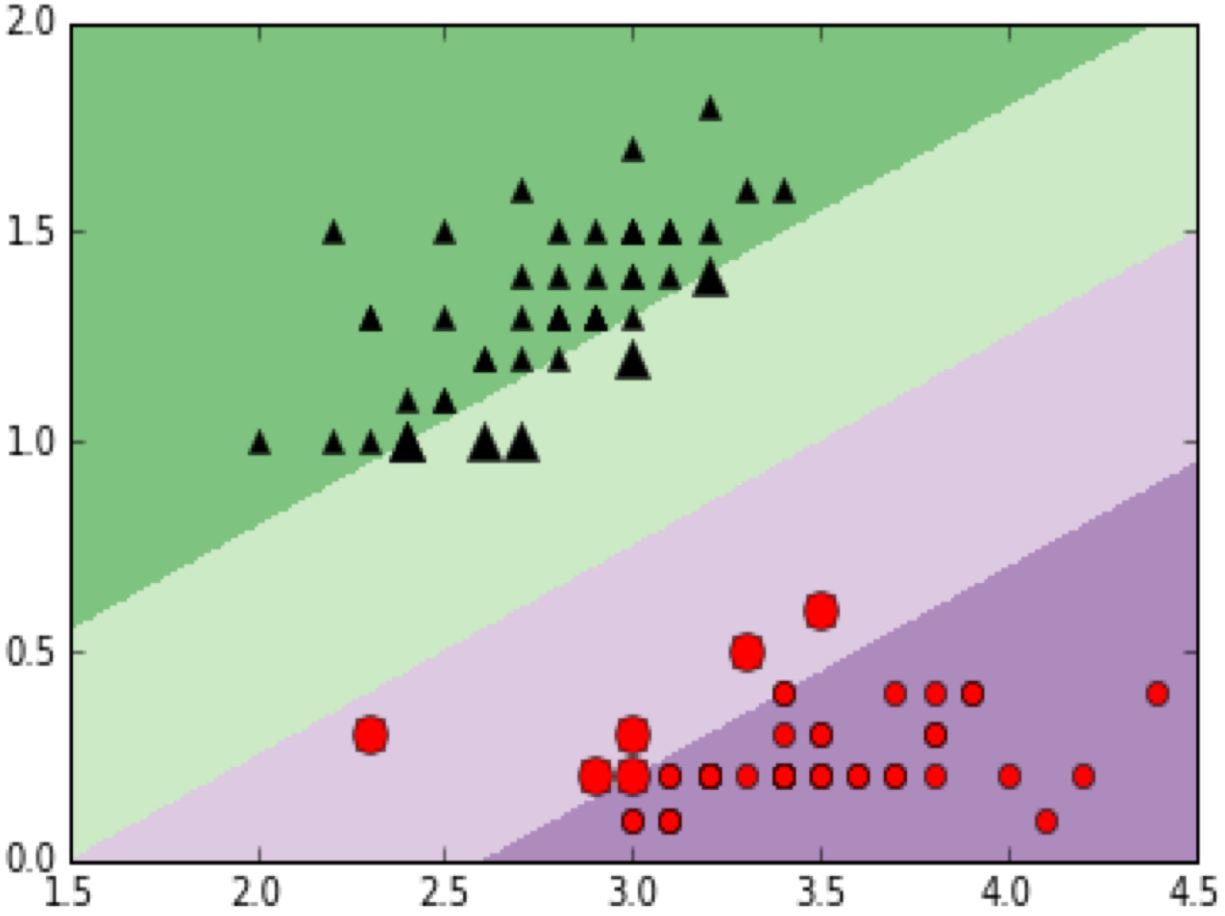
# Back to Iris

$C = 1$



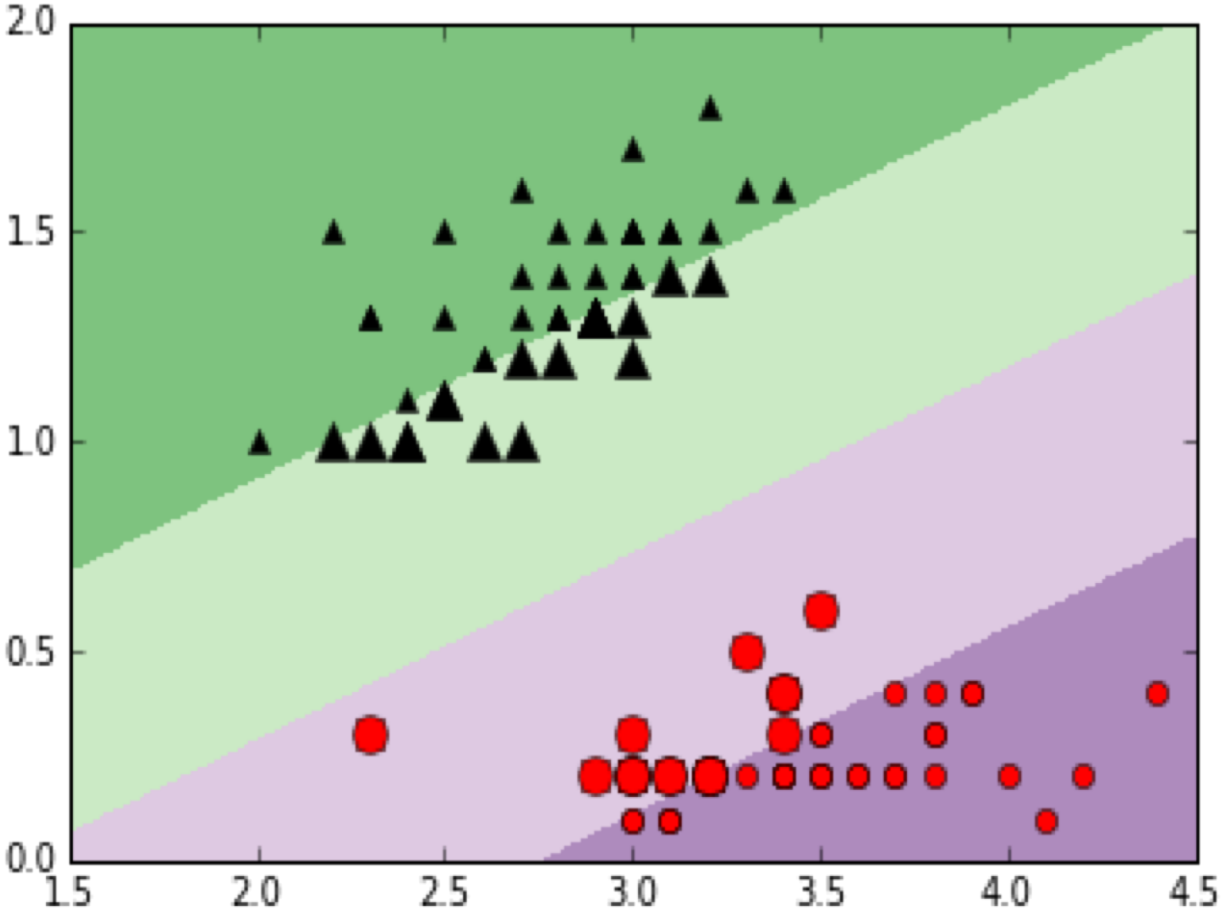
# Back to Iris

$C = 0.5$



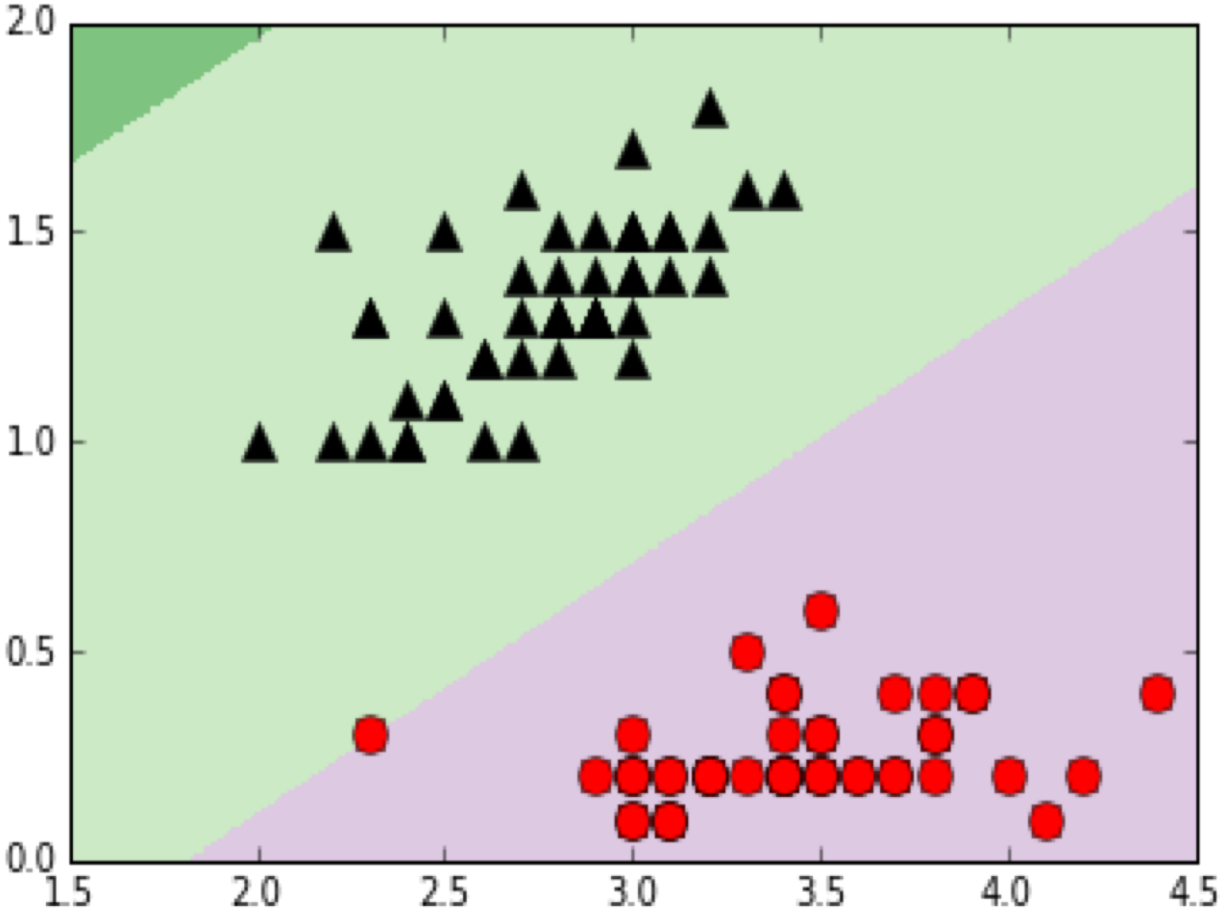
# Back to Iris

$C = 0.1$



# Back to Iris

$C = 0.01$



# Convex surrogates for 0-1 loss

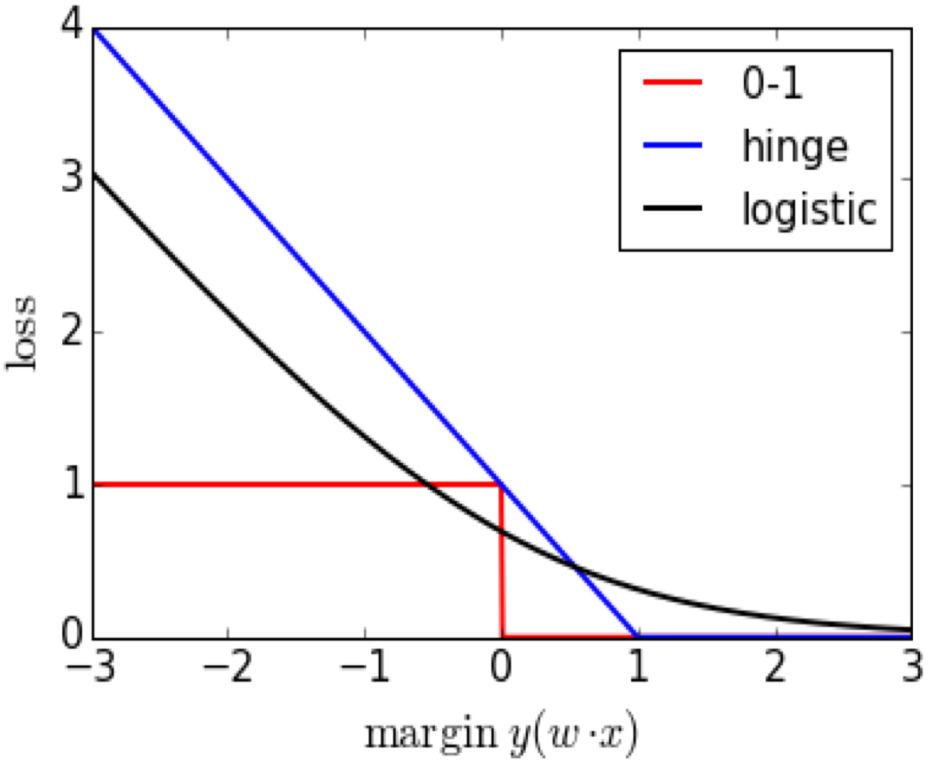
Want a separator $w$ that misclassifies as few training points as possible.

- 0-1 loss: charge $1(y\,(w \cdot x) < 0)$ for each $(x, y)$

Problem: this is NP-hard.

Instead, use **convex** loss functions.

- Hinge loss (SVM): charge $(1 - y\,(w \cdot x))_+$
- Logistic loss: charge $\ln(1 + e^{-y\,(w \cdot x)})$



# A high-level view of optimization

**Unconstrained optimization**

Logistic regression: find the vector $w \in \mathrm{R}^p$ that minimizes

$$L(w) = \sum_{i=1}^{n} \ln(1 + \exp(-y^{(i)}(w \cdot x^{(i)})).$$

We know how to check such problems for convexity and how to solve using gradient descent or Newton methods.

**Constrained optimization**

Support vector machine: find $w \in \mathrm{R}^p$ and $b \in \mathrm{R}$ that minimize

$$L(w) = ||w||^2$$

subject to the constraints

$$y^{(i)}(w \cdot x^{(i)} + b) \geq 1$$

# Constrained optimization

Write the optimization problem in a standardized form:

$$\min f_o(z)$$
$$f_i(z) \leq 0 \quad \text{for } i = 1, .., m$$
$$h_i(z) = 0 \quad \text{for } i = 1, .., n$$

Special cases that can be solved (relatively) easily:

- **Linear programs.**
  $f_o, f_i, h_i$ are all linear functions.
- **Convex programs.**
  $f_o, f_i$ are convex functions. The $h_i$ are linear functions.

# Example: regression with $\ell_1$ loss

Given $(x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)}) \in \mathbb{R}^p \times \mathbb{R}$, find $w \in \mathbb{R}^p$ minimizing

$$L(w) = \sum_{i=1}^{n} |y^{(i)} - (w \cdot x^{(i)})|.$$

Equivalently: let $X$ be the $n \times p$ matrix with rows $x^{(i)}$, and $y = (y^{(1)}, \ldots, y^{(n)})$. Then $L(w) = \|Xw - y\|_1$.

Optimization problem in $p + n$ variables, $w \in \mathbb{R}^p$ and $z \in \mathbb{R}^n$:

$$\min \sum_{i=1}^{n} z_i$$

$$y^{(i)} - w \cdot x^{(i)} \leq z_i, \quad i = 1, 2, \ldots, n$$

$$w \cdot x^{(i)} - y^{(i)} \leq z_i, \quad i = 1, 2, \ldots, n$$

A linear program.

# The dual of an optimization problem

Take any optimization problem, convex or not:

$$\begin{aligned} &\min f_o(z) \\ f_i(z) &\leq 0 \quad \text{for } i = 1, .., m \\ h_i(z) &= 0 \quad \text{for } i = 1, .., n \end{aligned}$$

Call this the *primal* problem.

There is a *dual* optimization problem over $n + m$ variables:

- $\lambda \in \mathbb{R}^m$, one variable for each primal inequality
- $\nu \in \mathbb{R}^n$, one variable for each primal equality

It is a maximization problem:

$$\begin{aligned} &\max g(\lambda, \nu) \\ &\lambda \geq 0 \end{aligned}$$

Constructing the dual is straightforward. But interpreting it is not.

# Duality and complementary slackness

Let $z^*$ and $\lambda^*, \nu^*$ be the optimal primal and dual solutions.

- Weak duality.
  Dual solution is at most the primal solution: $g(\lambda^*, \nu^*) \leq f_o(z^*)$.
- Strong duality.
  If primal problem is convex then (almost always, with an easily checkable condition) the primal and dual solutions are equal.
- Complementary slackness.
  If primal and dual solutions are equal, then for any $i = 1, \ldots, m$,

$$\lambda_i^* > 0 \;\Rightarrow\; f_i(z^*) = 0.$$

- KKT (Karush-Kuhn-Tucker) conditions.
  If the $f_i$ and $h_i$ are differentiable, these are first-derivative-equals-zero conditions that hold at optimality.