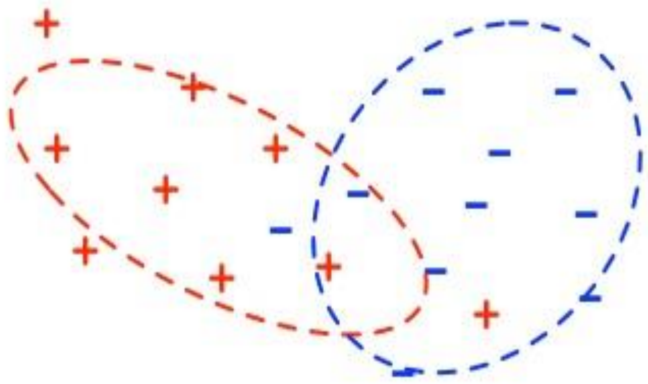
# Classification with discriminative models

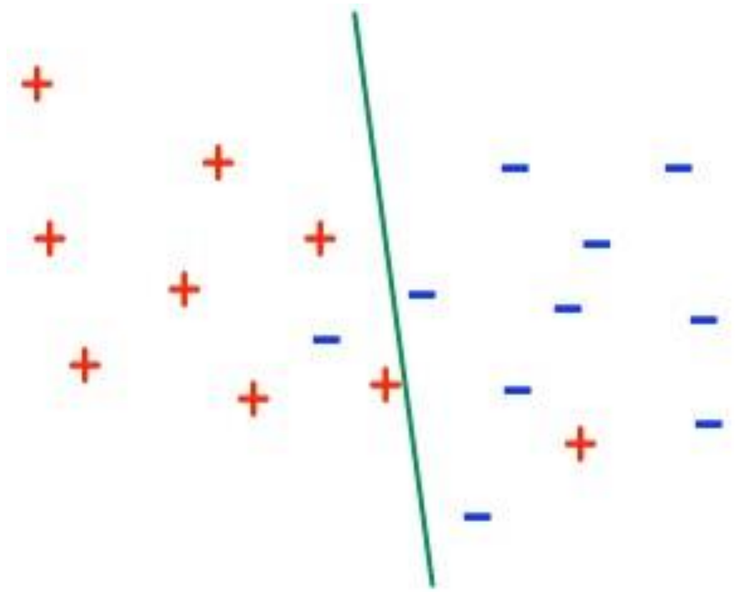
DSE 220

# Classification with parametrized models

Classifiers with a fixed number of parameters can represent a limited set of functions. Learning a model is about picking a good approximation.

Typically the *x* ’s are points in *p*-dimensional Euclidean space, $\mathbf{R}^p$





Two ways to classify:

- **Generative**: model the individual classes
- **Discriminative**: mode the decision boundary between the classes

# Generative models: pros and cons

Advantages:

- Multiclass is a breeze
- For many common models: converges fast
- Returns not just a classification but also a confidence $\Pr(y|x)$

# Generative models: pros and cons

Advantages:

- Multiclass is a breeze
- For many common models: converges fast
- Returns not just a classification but also a confidence $\Pr(y|x)$

Disadvantages:

Formula for $\Pr(y|x)$ assumes the class specific density models are perfect but this is never true

# Generative models: pros and cons

Advantages:

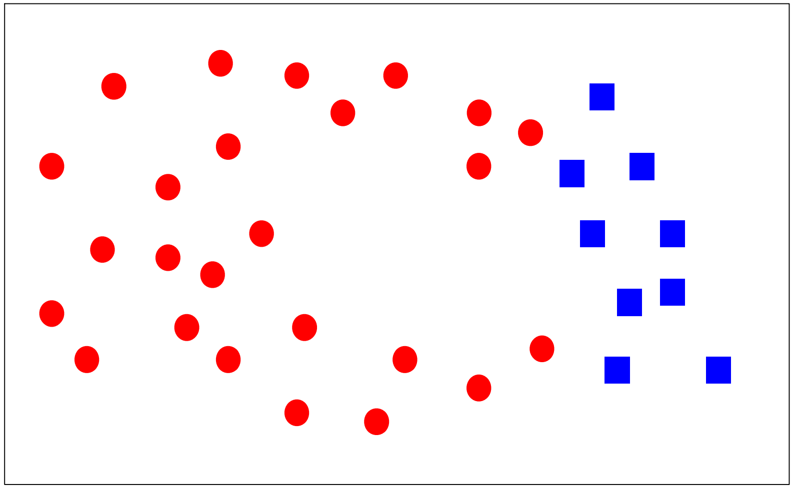
- Multiclass is a breeze
- For many common models: converges fast
- Returns not just a classification but also a confidence $\Pr(y|x)$

Disadvantages:

Formula for $\Pr(y|x)$ assumes the class specific density models are perfect but this is never true

If we only care about classification, shouldn't we focus on the decision boundary rather than trying to model other aspects of the distribution of $x$ ?

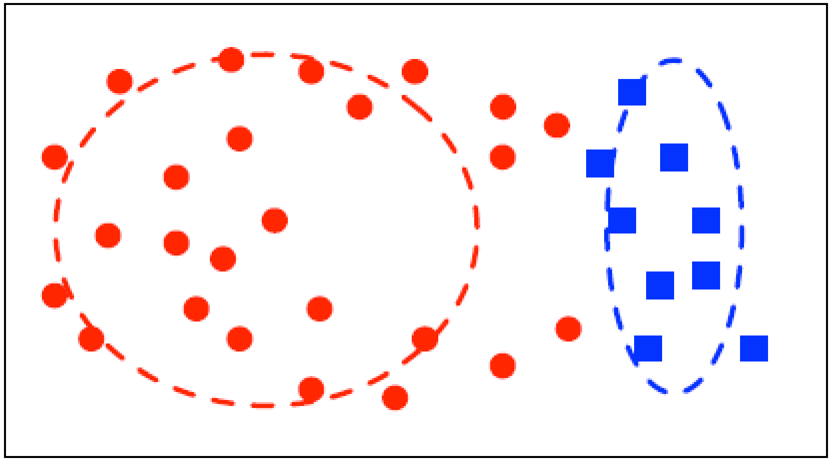
# Generative versus discriminative



The generative way:

- Fit: $\pi_0$, $\pi_1$, $P_0$,$P_1$
- This determines a full joint distribution Pr($x$ , $y$ )
- Use Bayes' rule to obtain Pr ($y|x$)

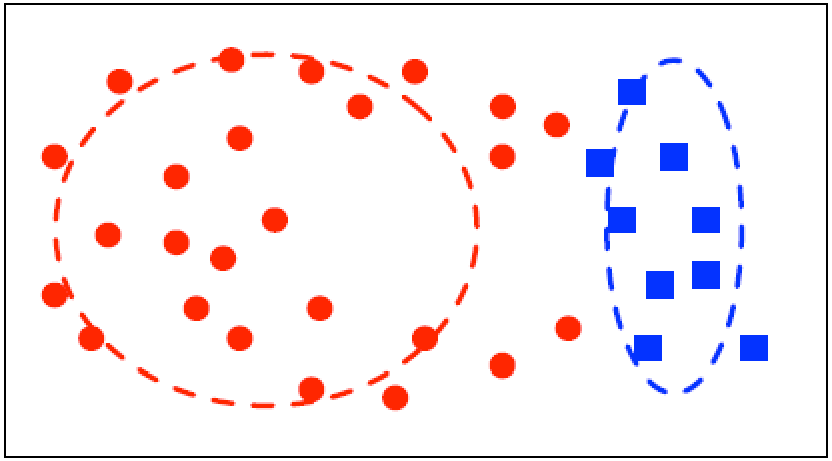
# Generative versus discriminative



The generative way:

- Fit: $\pi_0$, $\pi_1$, $P_0$,$P_1$
- This determines a full joint distribution Pr($x$ , $y$ )
- Use Bayes' rule to obtain Pr ($y|x$)

# Generative versus discriminative



The generative way:

- Fit: $\pi_0$, $\pi_1$, $P_0$,$P_1$
- This determines a full joint distribution Pr($x$ , $y$ )
- Use Bayes' rule to obtain Pr ($y|x$)

The generative way: model Pr($y|x$) directly.
In our earlier terminology: forget about the $\mu$ (Prob. Distribution), just learn the $\eta$ (likelihood)

# The logistic regression model

What model to use for $\Pr(y \mid x)$?

# The logistic regression model

What model to use for $\Pr(y \mid x)$?

- Say $\mathcal{Y} = \{-1, 1\}$. Recall: for Gaussians with common covariance,

$$\ln \frac{\Pr(y = 1 \mid x)}{\Pr(y = -1 \mid x)} = \underbrace{w \cdot x + \theta}_{\text{linear}}$$

# The logistic regression model

What model to use for Pr(*y* |*x* )?

- Say $\mathcal{Y} = \{-1, 1\}$. Recall: for Gaussians with common covariance,

$$\ln \frac{\Pr(y = 1 \mid x)}{\Pr(y = -1 \mid x)} = \underbrace{w \cdot x + \theta}_{\text{linear}}$$

- Can drop $\theta$ by adding an extra feature to $x$.

# The logistic regression model

What model to use for Pr($y$ | $x$ )?

- Say $\mathcal{Y} = \{-1, 1\}$. Recall: for Gaussians with common covariance,

$$\ln \frac{\Pr(y = 1 \mid x)}{\Pr(y = -1 \mid x)} = \underbrace{w \cdot x + \theta}_{\text{linear}}$$

- Can drop $\theta$ by adding an extra feature to $x$.
- Then $\Pr(y = 1 \mid x) = \Pr(y = -1 \mid x)\, e^{w \cdot x}$, where $\Pr(y = 1 \mid x) = 1 - \Pr(y = -1 \mid x)$

$$\Pr(y = -1 \mid x) = \frac{1}{1 + e^{w \cdot x}}$$

$$\Pr(y = 1 \mid x) = 1 - \frac{1}{1 + e^{w \cdot x}} = \frac{e^{w \cdot x}}{1 + e^{w \cdot x}} = \frac{1}{1 + e^{-w \cdot x}}$$

# The logistic regression model

What model to use for Pr($y$ | $x$ )?

- Say $\mathcal{Y} = \{-1, 1\}$. Recall: for Gaussians with common covariance,

$$\ln \frac{\Pr(y = 1 \mid x)}{\Pr(y = -1 \mid x)} = \underbrace{w \cdot x + \theta}_{\text{linear}}$$

- Can drop $\theta$ by adding an extra feature to $x$.
- Then $\Pr(y = 1 \mid x) = \Pr(y = -1 \mid x)\, e^{w \cdot x}$, where $\Pr(y = 1 \mid x) = 1 - \Pr(y = -1 \mid x)$

$$\Pr(y = -1 \mid x) = \frac{1}{1 + e^{w \cdot x}}$$

$$\Pr(y = 1 \mid x) = 1 - \frac{1}{1 + e^{w \cdot x}} = \frac{e^{w \cdot x}}{1 + e^{w \cdot x}} = \frac{1}{1 + e^{-w \cdot x}}$$

- More concisely,

$$\Pr(y \mid x) = \frac{1}{1 + e^{-y(w \cdot x)}}$$

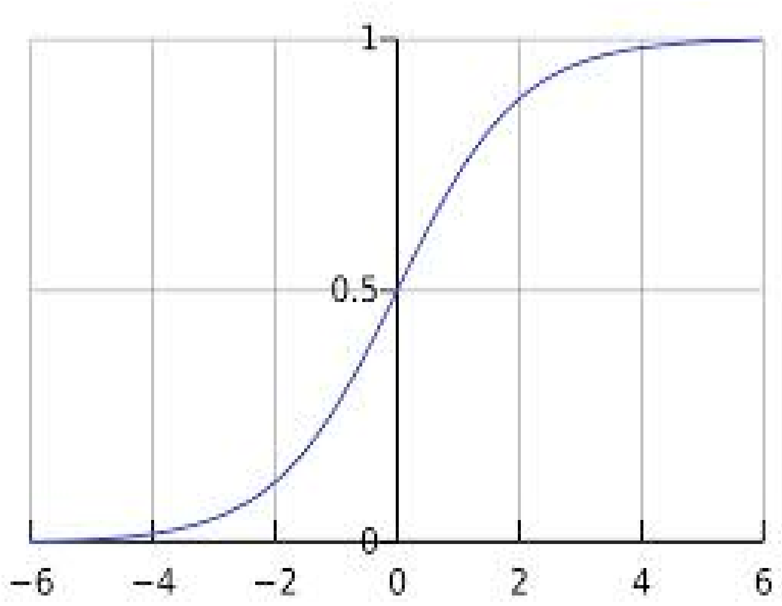
This is the **logistic regression model**, parametrized by $w$.

# The squashing function

Take $X = \mathbf{R}^p$ and $Y = \{-1, 1\}$. The model specified by $w \in \mathbf{R}^p$ is

$$\Pr_w(y \mid x) = \frac{1}{1 + e^{-y\,(w \cdot x)}} = g\,(y\,(w \cdot x)),$$

where $g\,(z) = 1/(1 + e^{-z})$ is the *squashing function*.

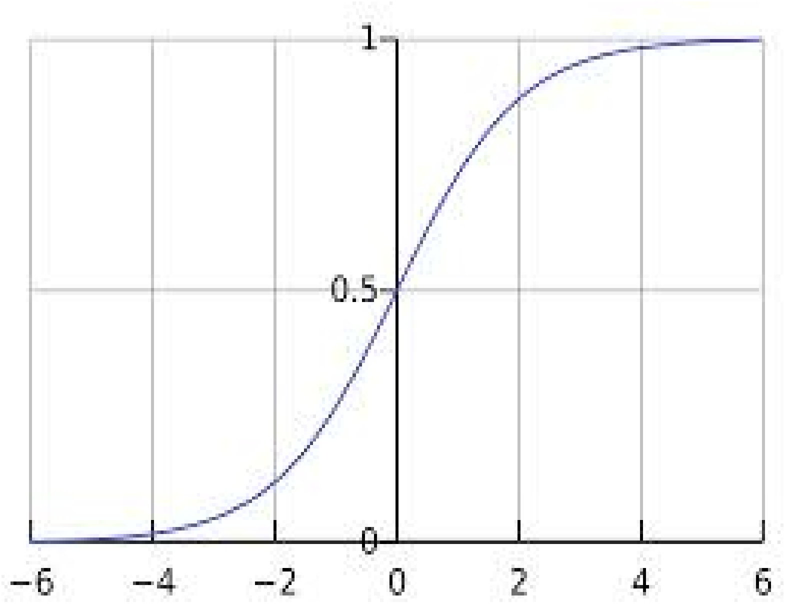


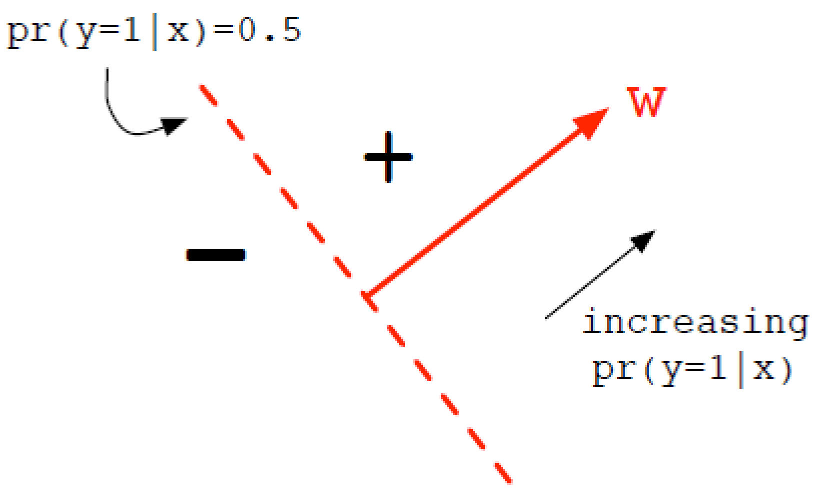
# The squashing function

Take $X = \mathbf{R}^p$ and $Y = \{-1, 1\}$. The model specified by $w \in \mathbf{R}^p$ is

$$\Pr_w(y \mid x) = \frac{1}{1 + e^{-y(w \cdot x)}} = g(y(w \cdot x)),$$

where $g(z) = 1/(1 + e^{-z})$ is the *squashing function*.



# Fitting $w$

The maximum-likelihood principle: given a data set

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

pick the $w \in \mathbb{R}^p$ that maximizes

$$\prod_{i=1}^{n} \Pr_w(y^{(i)} \mid x^{(i)}).$$

# Fitting $w$

The maximum-likelihood principle: given a data set

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

pick the $w \in \mathbb{R}^p$ that maximizes

$$\prod_{i=1}^{n} \Pr_w(y^{(i)} \mid x^{(i)}).$$

Easier to work with sums, so take negative log to get **loss function**

$$L(w) \;=\; -\sum_{i=1}^{n} \ln \Pr_w(y^{(i)} \mid x^{(i)})$$

$$= \; -\sum_{i=1}^{n} \ln\left(\frac{1}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}}\right) \;=\; \sum_{i=1}^{n} \ln(1 + e^{-y^{(i)}(w \cdot x^{(i)})})$$

Our goal is to minimize $L(w)$.

# Fitting $w$

The maximum-likelihood principle: given a data set

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

pick the $w \in \mathbb{R}^p$ that maximizes

$$\prod_{i=1}^{n} \Pr_w(y^{(i)} \mid x^{(i)}).$$

Easier to work with sums, so take negative log to get **loss function**

$$L(w) \;=\; -\sum_{i=1}^{n} \ln \Pr_w(y^{(i)} \mid x^{(i)})$$

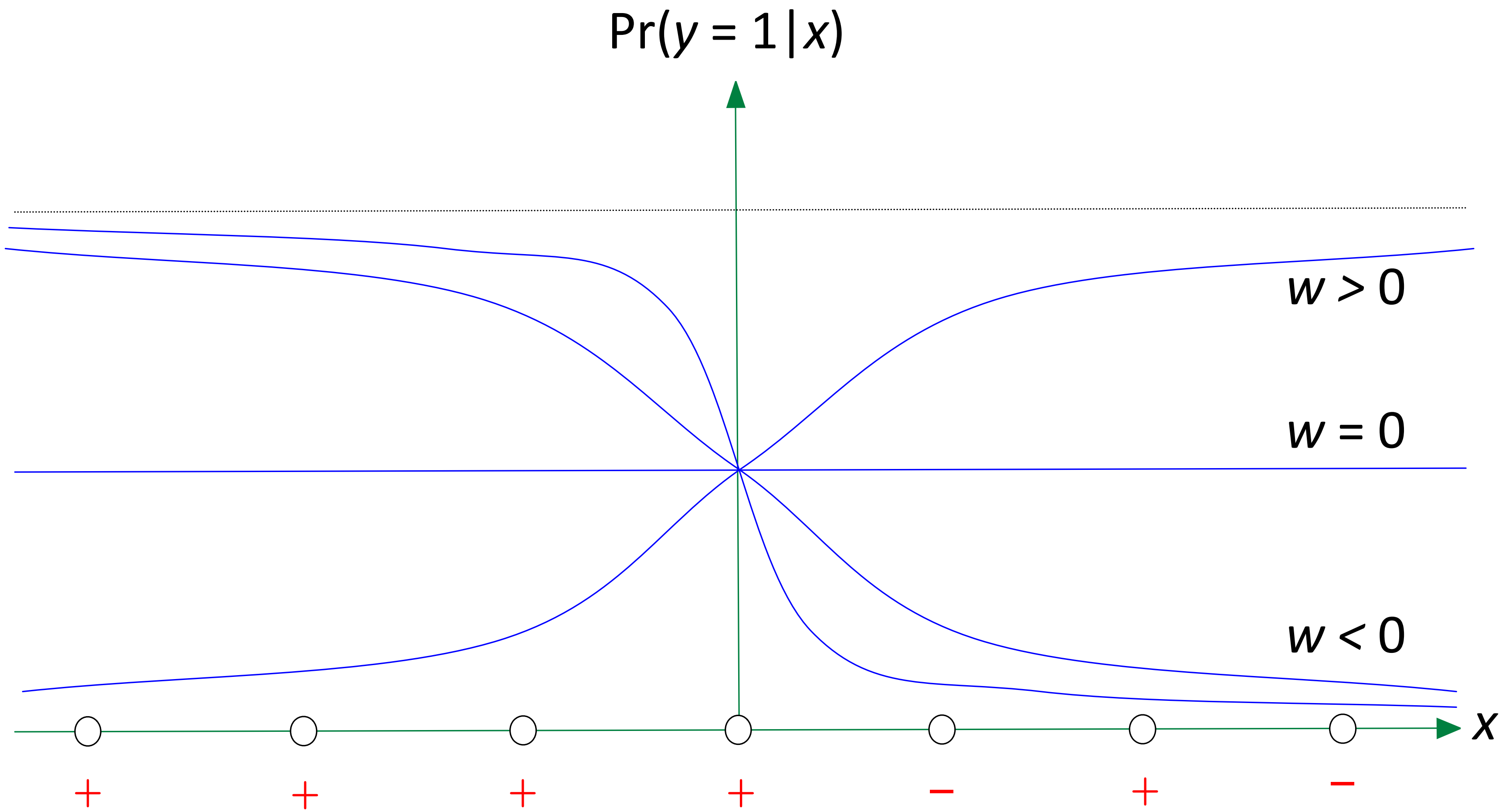
$$=\; -\sum_{i=1}^{n} \ln\left(\frac{1}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}}\right) \;=\; \sum_{i=1}^{n} \ln(1 + e^{-y^{(i)}(w \cdot x^{(i)})})$$

Our goal is to minimize $L(w)$.

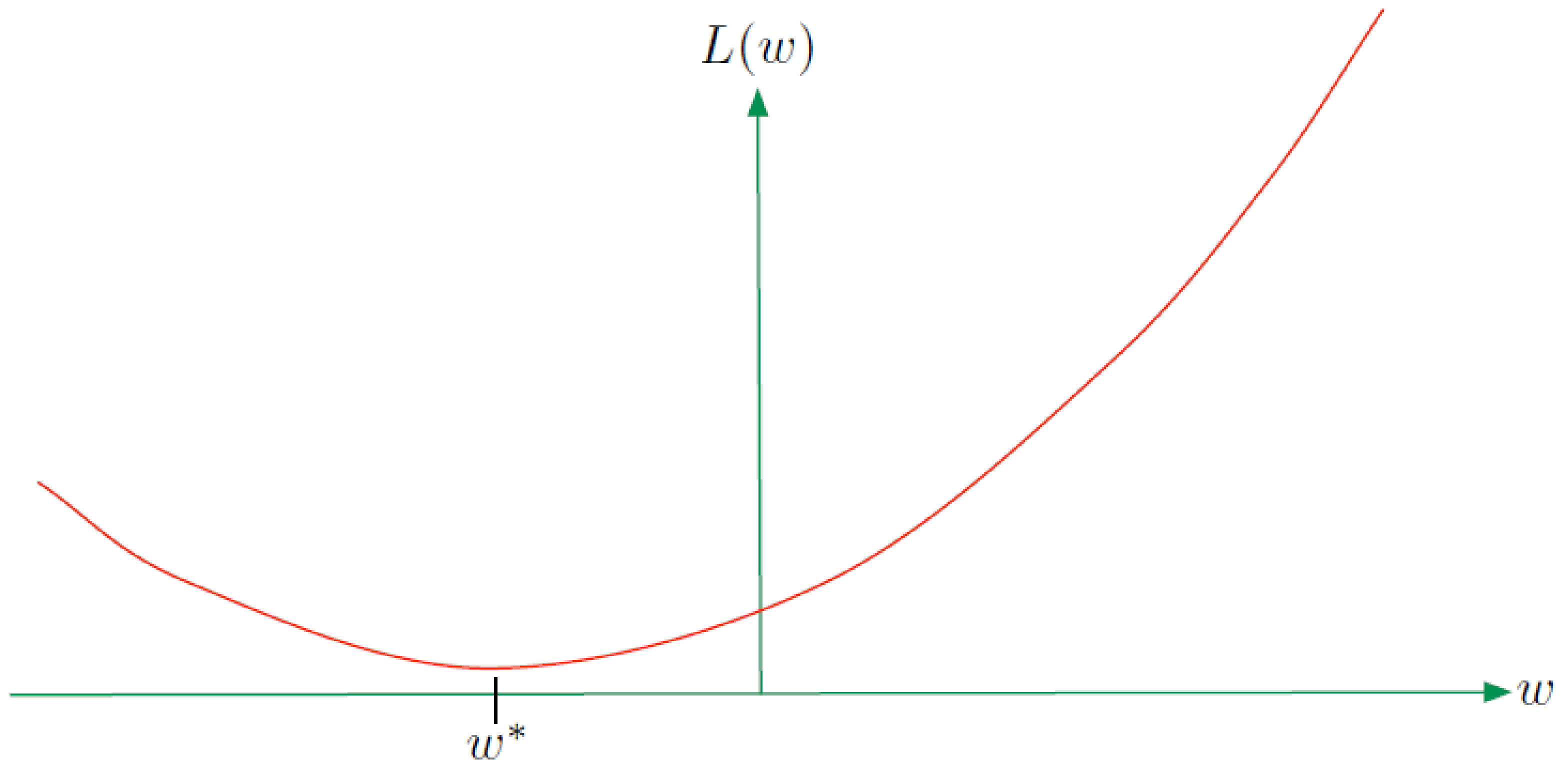
The good news: $L(w)$ is **convex** in $w$.

# One dimensional example

$$\Pr_w(y \mid x) = \frac{1}{1 + e^{-ywx}}, \quad w \in \mathbf{R}$$



# Example, cont'd




How to find the minimum of this convex function? A variety of options:

- Gradient descent
- Newton-Raphson

and many others.

# Gradient descent procedure for logistic regression

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

$$\arg\min_{w \in \mathbb{R}^p} L(w) \;=\; \sum_{i=1}^{n} \ln(1 + e^{-y^{(i)}(w \cdot x^{(i)})})$$

- Set $w_0 = 0$
- For $t = 0, 1, 2, \ldots,$ until convergence:

$$w_{t+1} \;=\; w_t \;+\; \eta_t \sum_{i=1}^{n} y^{(i)} x^{(i)} \underbrace{\Pr_{w_t}(-y^{(i)} | x^{(i)})}_{\text{doubt}_t(x^{(i)}, y^{(i)})},$$

where $\eta_t$ is a step size chosen by line search to minimize $L(w_{t+1})$.

# Newton-Raphson procedure for logistic regression

- Set $w_0 = 0$
- For $t = 0, 1, 2, \ldots$, until convergence:

$$w_{t+1} \;=\; w_t \;+\; \eta_t \, (X^T D_t X)^{-1} \sum_{i=1}^{n} y^{(i)} x^{(i)} \mathrm{Pr}_{w_t}(-y^{(i)} | x^{(i)}),$$

  where
  - $X$ is the $n \times p$ data matrix with one point per row
  - $D_t$ is an $n \times n$ diagonal matrix with $(i, i)$ entry

$$D_{t,ii} \;=\; \mathrm{Pr}_{w_t}(1|x^{(i)}) \, \mathrm{Pr}_{w_t}(-1|x^{(i)})$$

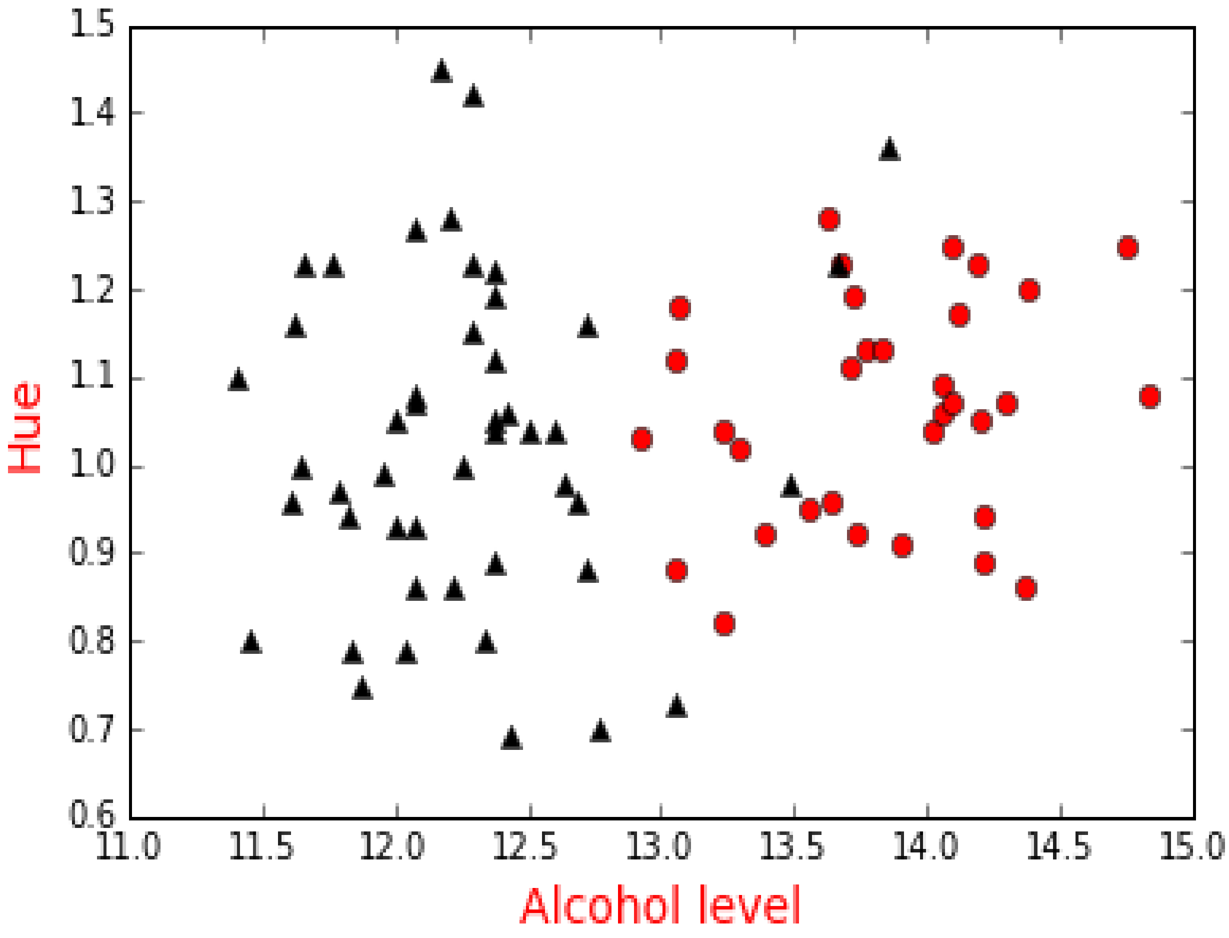
  - $\eta_t$ is a step size that is either fixed to 1 ("iterative reweighted least squares") or chosen by line search to minimize $L(w_{t+1})$.

# Example: “wine” data set

Recall: data from three wineries from the same region of Italy.

- 13 attributes: hue, color intensity, flavanoids, ash content, ...
- 178 instances in all: split into 118 train, 60 test

Pick two classes and just two attributes (hue, alcohol content).

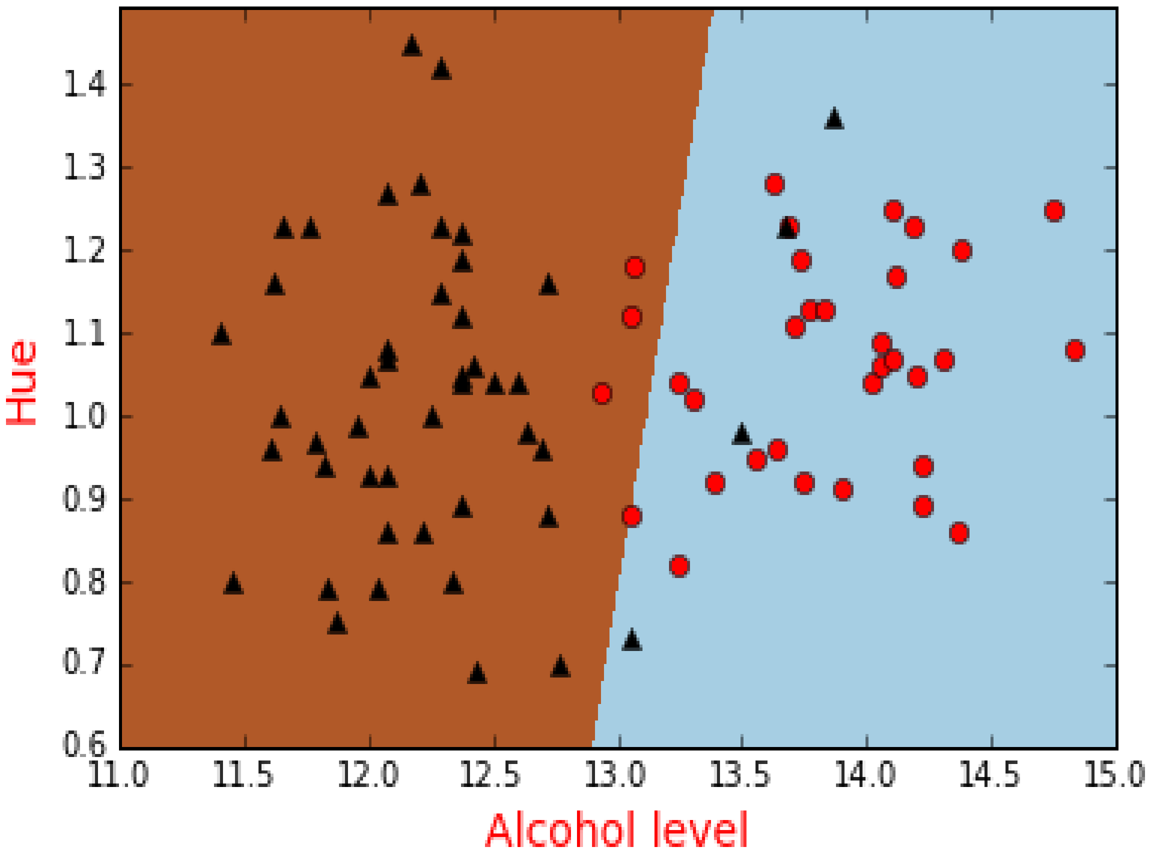


# Example: "wine" data set

Recall: data from three wineries from the same region of Italy.

- 13 attributes: hue, color intensity, flavanoids, ash content, ...
- 178 instances in all: split into 118 train, 60 test

Pick two classes and just two attributes (hue, alcohol content).




Test error using logistic regression: 10%.

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

**First derivative**. There are $p$ partial derivatives $\partial L / \partial w_j$. Put into a vector:

$$\nabla L(w) = \left( \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_p} \right).$$

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

**First derivative**. There are $p$ partial derivatives $\partial L / \partial w_j$. Put into a vector:

$$\nabla L(w) = \left( \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_p} \right).$$

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^{n} \frac{e^{-y^{(i)}(w \cdot x^{(i)})}}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}} (-y^{(i)} x_j^{(i)}) = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

$$\nabla L(w) = -\sum_{i=1}^{n} y^{(i)} x^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

**First derivative**. There are $p$ partial derivatives $\partial L / \partial w_j$. Put into a vector:

$$\nabla L(w) = \left( \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_p} \right).$$

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^{n} \frac{e^{-y^{(i)}(w \cdot x^{(i)})}}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}} (-y^{(i)} x_j^{(i)}) = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

$$\nabla L(w) = -\sum_{i=1}^{n} y^{(i)} x^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

We know $\nabla L(w) = 0$ iff $w$ is a local optimum.

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

**First derivative**. There are $p$ partial derivatives $\partial L / \partial w_j$. Put into a vector:

$$\nabla L(w) = \left( \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_p} \right).$$

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^{n} \frac{e^{-y^{(i)}(w \cdot x^{(i)})}}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}} (-y^{(i)} x_j^{(i)}) = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

$$\nabla L(w) = -\sum_{i=1}^{n} y^{(i)} x^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

We know $\nabla L(w) = 0$ iff $w$ is a local optimum. But we want a **global** optimum.

# A closer look at the logistic loss

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

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

Can't we just minimize this by calculus?

**First derivative**. There are $p$ partial derivatives $\partial L / \partial w_j$. Put into a vector:

$$\nabla L(w) = \left( \frac{\partial L}{\partial w_1}, \ldots, \frac{\partial L}{\partial w_p} \right).$$

$$\frac{\partial L}{\partial w_j} = \sum_{i=1}^{n} \frac{e^{-y^{(i)}(w \cdot x^{(i)})}}{1 + e^{-y^{(i)}(w \cdot x^{(i)})}} (-y^{(i)} x_j^{(i)}) = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

$$\nabla L(w) = -\sum_{i=1}^{n} y^{(i)} x^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}$$

Good news: $L(w)$ is **convex**, so local optimum implies global optimum.

# Quick quiz

Compute the first derivative of the following functions on $\mathbf{R}^p$.

1. $F(w) = u \cdot w$, where $u \in \mathbf{R}^p$

# Quick quiz

Compute the first derivative of the following functions on $\mathbf{R}^p$

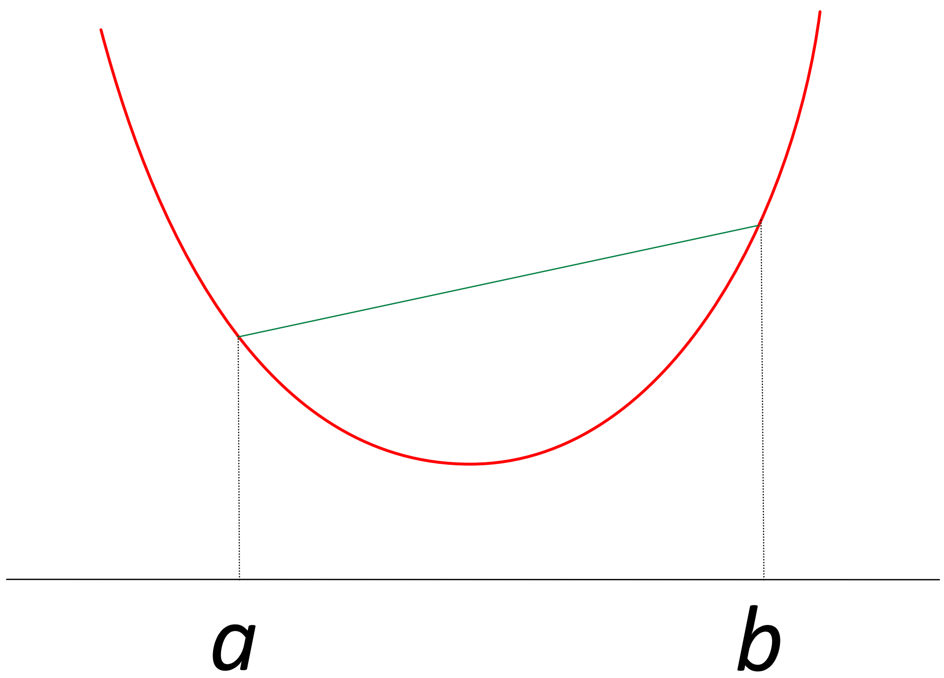
1. $F(w) = u \cdot w$, where $u \in \mathbf{R}^p$
2. $F(w) = w^T w$

# Quick quiz: Answers

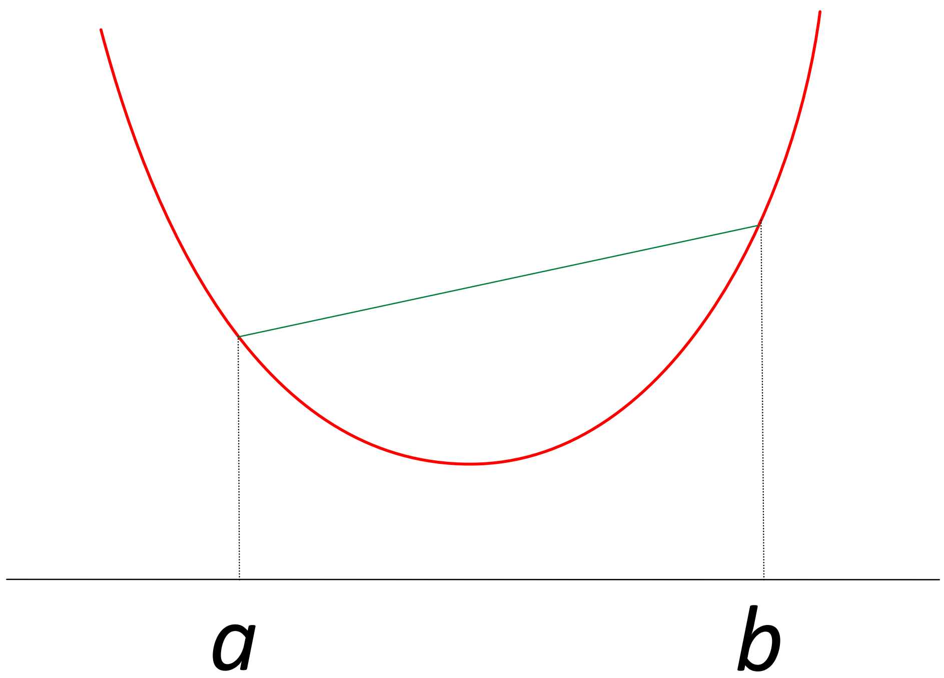
Compute the first derivative of the following functions on $\mathbf{R}^p$

1. $F(w) = u \cdot w$, where $u \in \mathbf{R}^p \rightarrow \mathbf{u}$
2. $F(w) = w^T w \rightarrow \boldsymbol{2w}$

# Convexity



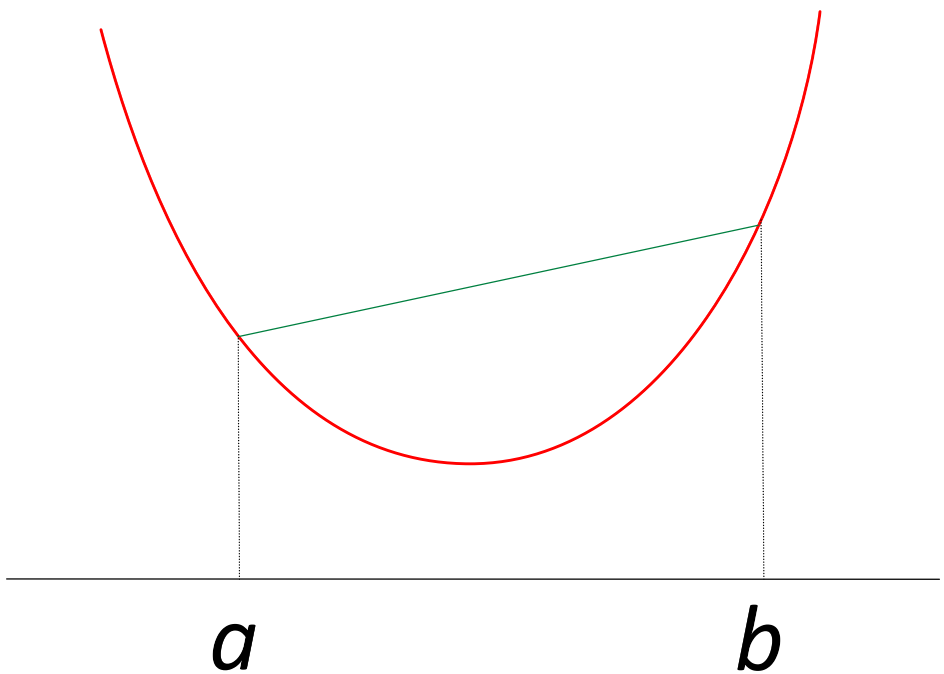
# Convexity




A function $f : \mathbb{R}^p \to \mathbb{R}$ is **convex** if for all $a, b \in \mathbb{R}^p$ and $0 < \theta < 1$,

$$f(\theta a + (1-\theta)b) \;\leq\; \theta f(a) + (1-\theta) f(b).$$

# Convexity

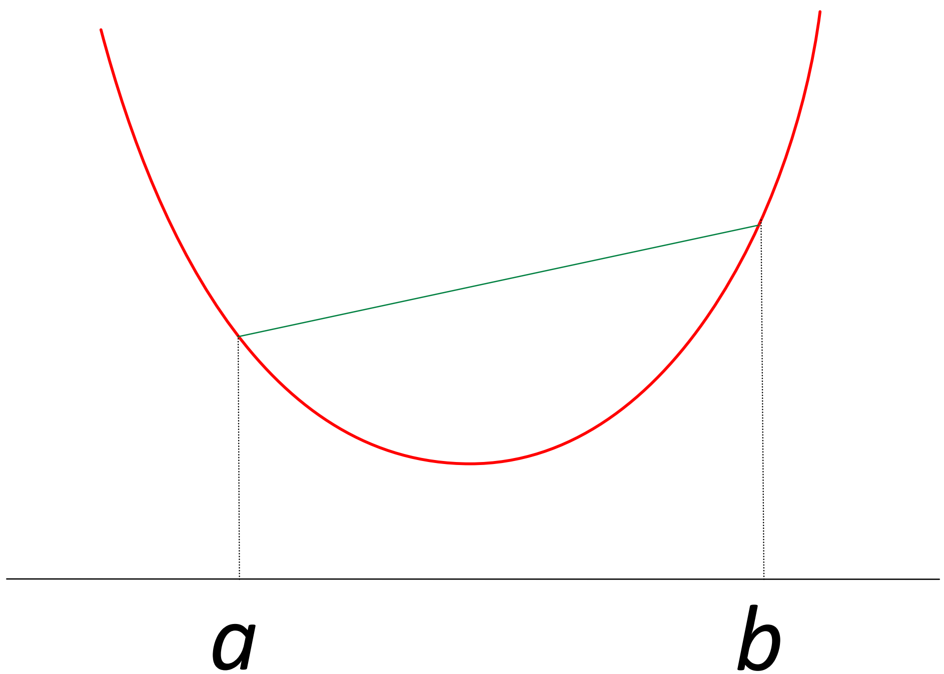



A function $f : \mathbb{R}^p \to \mathbb{R}$ is **convex** if for all $a, b \in \mathbb{R}^p$ and $0 < \theta < 1$,

$$f(\theta a + (1-\theta)b) \ \le \ \theta f(a) + (1-\theta)f(b).$$

It is **strictly convex** if strict inequality holds for all $a \neq b$.

# Convexity




A function $f : \mathbb{R}^p \to \mathbb{R}$ is **convex** if for all $a, b \in \mathbb{R}^p$ and $0 < \theta < 1$,

$$f(\theta a + (1-\theta)b) \ \leq \ \theta f(a) + (1-\theta)f(b).$$

It is **strictly convex** if strict inequality holds for all $a \neq b$.

$f$ is **concave** (resp., **strictly concave**) iff $-f$ is convex (resp., strictly convex).

# Second-derivative test for convexity

A function $f : \mathbb{R}^p \to \mathbb{R}$ has $p^2$ second partial derivatives $\frac{\partial^2 f}{\partial z_j \partial z_k}$ at any $z \in \mathbb{R}^p$ (assuming these exist).

# Second-derivative test for convexity

A function $f : \mathbb{R}^p \to \mathbb{R}$ has $p^2$ second partial derivatives $\frac{\partial^2 f}{\partial z_j \partial z_k}$ at any $z \in \mathbb{R}^p$ (assuming these exist).

Assemble into a $p \times p$ matrix, the **Hessian** $H(z)$:

$$H_{jk} \;=\; \frac{\partial^2 f}{\partial z_j \partial z_k}.$$

# Second-derivative test for convexity

A function $f : \mathbb{R}^p \to \mathbb{R}$ has $p^2$ second partial derivatives $\frac{\partial^2 f}{\partial z_j \partial z_k}$ at any $z \in \mathbb{R}^p$ (assuming these exist).

Assemble into a $p \times p$ matrix, the **Hessian** $H(z)$:

$$H_{jk} = \frac{\partial^2 f}{\partial z_j \partial z_k}.$$

**Useful fact.** Suppose that for $f : \mathbb{R}^p \to \mathbb{R}$, the second partial derivatives exist everywhere and are continuous functions of $z$. Then:

1. $H(z)$ is a symmetric matrix.
2. $f$ is convex if and only if $H(z)$ is positive semidefinite for all $z \in \mathbb{R}^p$.

# Quick quiz

Is this function $f : \mathbb{R}^p \to \mathbb{R}$ convex?

$$f(z) = (u \cdot z)^2$$

(Here $u$ is some fixed vector in $\mathbb{R}^p$.)

# Quick quiz: Answers

Is this function $f : \mathbb{R}^p \to \mathbb{R}$ convex?

$$f(z) = (u \cdot z)^2$$

(Here $u$ is some fixed vector in $\mathbb{R}^p$.)

Yes

Calculate Hessian. It will be of the form $UU^T$

# Convexity of logistic regression loss function

Recall the loss function: for data $(x^{(i)}, y^{(i)}) \in \mathbb{R}^p \times \{-1, +1\}$,

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

We already know the first derivative:

$$\frac{\partial L}{\partial w_j} \;=\; -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}.$$

# Convexity of logistic regression loss function

Recall the loss function: for data $(x^{(i)}, y^{(i)}) \in \mathbb{R}^p \times \{-1, +1\}$,

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

We already know the first derivative:

$$\frac{\partial L}{\partial w_j} \;=\; -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}.$$

Second derivative: the $(j, k)$ entry of the Hessian $H(w)$ is

$$\begin{aligned}
\frac{\partial L}{\partial w_k \partial w_j} &= -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} (-1) \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} y^{(i)} x_k^{(i)} \\
&= \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} = \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \, \frac{1}{1 + e^{w \cdot x^{(i)}}} \, \frac{1}{1 + e^{-w \cdot x^{(i)}}}
\end{aligned}$$

# Convexity of logistic regression loss function

Recall the loss function: for data $(x^{(i)}, y^{(i)}) \in \mathbb{R}^p \times \{-1, +1\}$,

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

We already know the first derivative:

$$\frac{\partial L}{\partial w_j} = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}.$$

Second derivative: the $(j,k)$ entry of the Hessian $H(w)$ is

$$\begin{aligned}\frac{\partial L}{\partial w_k \partial w_j} &= -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} (-1) \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} y^{(i)} x_k^{(i)} \\ &= \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} = \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \frac{1}{1 + e^{w \cdot x^{(i)}}} \frac{1}{1 + e^{-w \cdot x^{(i)}}}\end{aligned}$$

This is $u_j \cdot u_k$, where vectors $u_1, \ldots, u_p \in \mathbb{R}^n$ are defined as follows:

$$u_j \text{ has } i\text{th coordinate } x_j^{(i)} \sqrt{\frac{1}{(1 + e^{w \cdot x^{(i)}})(1 + e^{-w \cdot x^{(i)}})}}.$$

# Convexity of logistic regression loss function

Recall the loss function: for data $(x^{(i)}, y^{(i)}) \in \mathbb{R}^p \times \{-1,+1\}$,

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

We already know the first derivative:

$$\frac{\partial L}{\partial w_j} = -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} \frac{1}{1 + e^{y^{(i)}(w \cdot x^{(i)})}}.$$

Second derivative: the $(j,k)$ entry of the Hessian $H(w)$ is

$$\begin{aligned}
\frac{\partial L}{\partial w_k \partial w_j} &= -\sum_{i=1}^{n} y^{(i)} x_j^{(i)} (-1) \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} y^{(i)} x_k^{(i)} \\
&= \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \frac{e^{y^{(i)}(w \cdot x^{(i)})}}{(1 + e^{y^{(i)}(w \cdot x^{(i)})})^2} = \sum_{i=1}^{n} x_j^{(i)} x_k^{(i)} \frac{1}{1 + e^{w \cdot x^{(i)}}} \frac{1}{1 + e^{-w \cdot x^{(i)}}}
\end{aligned}$$

This is $u_j \cdot u_k$, where vectors $u_1, \ldots, u_p \in \mathbb{R}^n$ are defined as follows:

$$u_j \text{ has } i\text{th coordinate } x_j^{(i)} \sqrt{\frac{1}{(1 + e^{w \cdot x^{(i)}})(1 + e^{-w \cdot x^{(i)}})}}.$$

Therefore $H(w) = UU^T$, where $U$ is the matrix with rows $u_j$ $\Rightarrow$ convex.

# Gradient descent

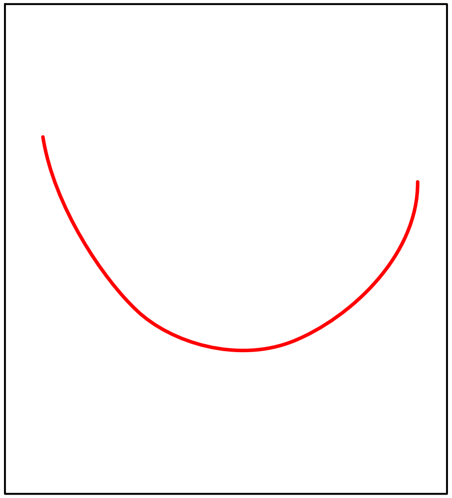
For minimizing a function $L(w)$:

- $w_o = 0,\ t = 0$
- while $\nabla L(w_t) \not\approx 0$:
  - $w_{t+1} = w_t - \eta_t \nabla L(w_t)$
  - $t = t + 1$

Here $\eta_t$ is the *step size* at time $t$.

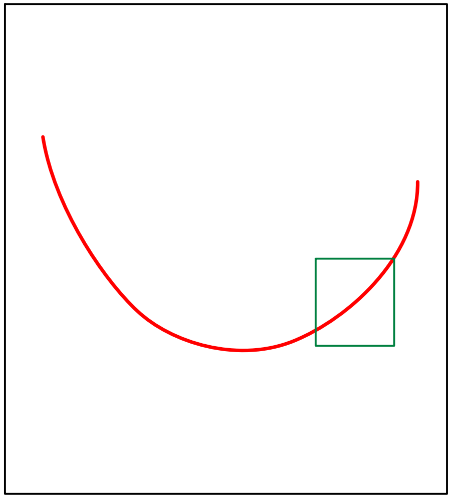
# Gradient descent: rationale

"Differentiable" means "locally linear".



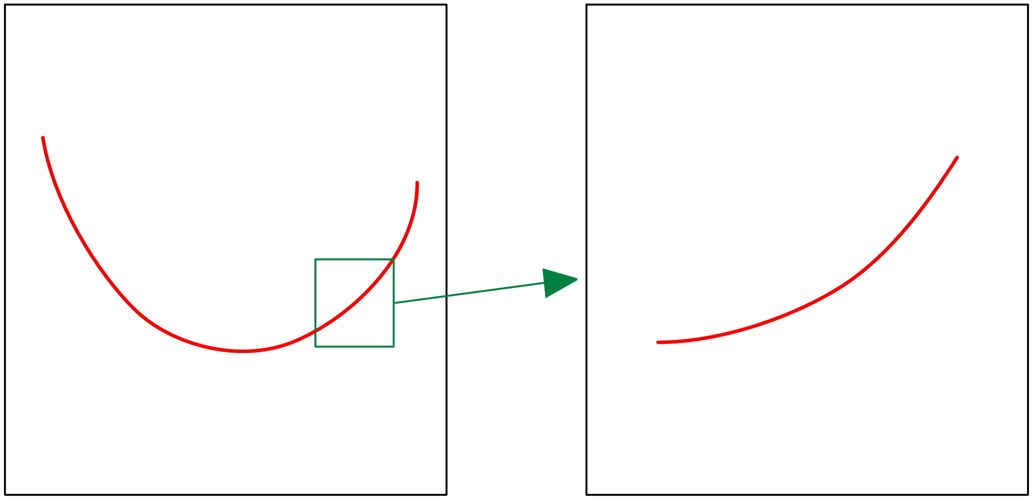
# Gradient descent: rationale

"Differentiable" means "locally linear".



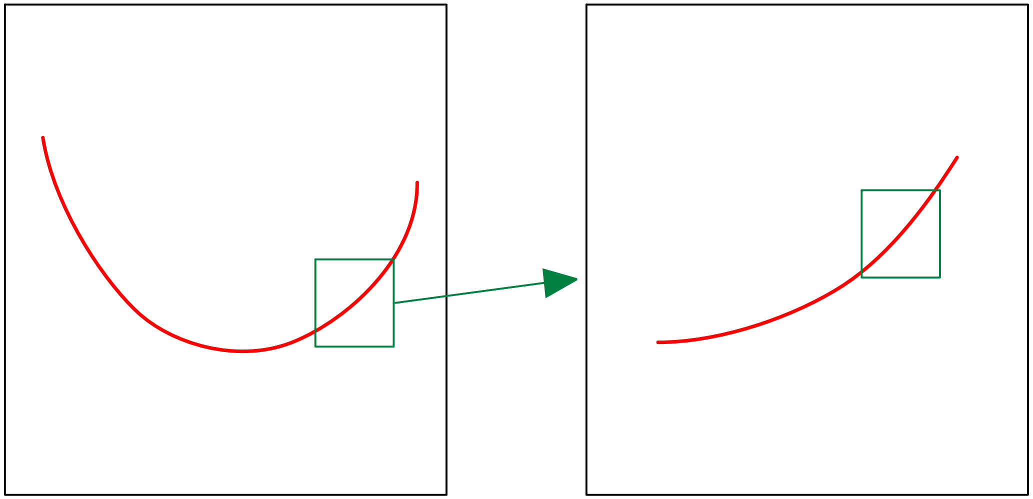
# Gradient descent: rationale

"Differentiable" means "locally linear".



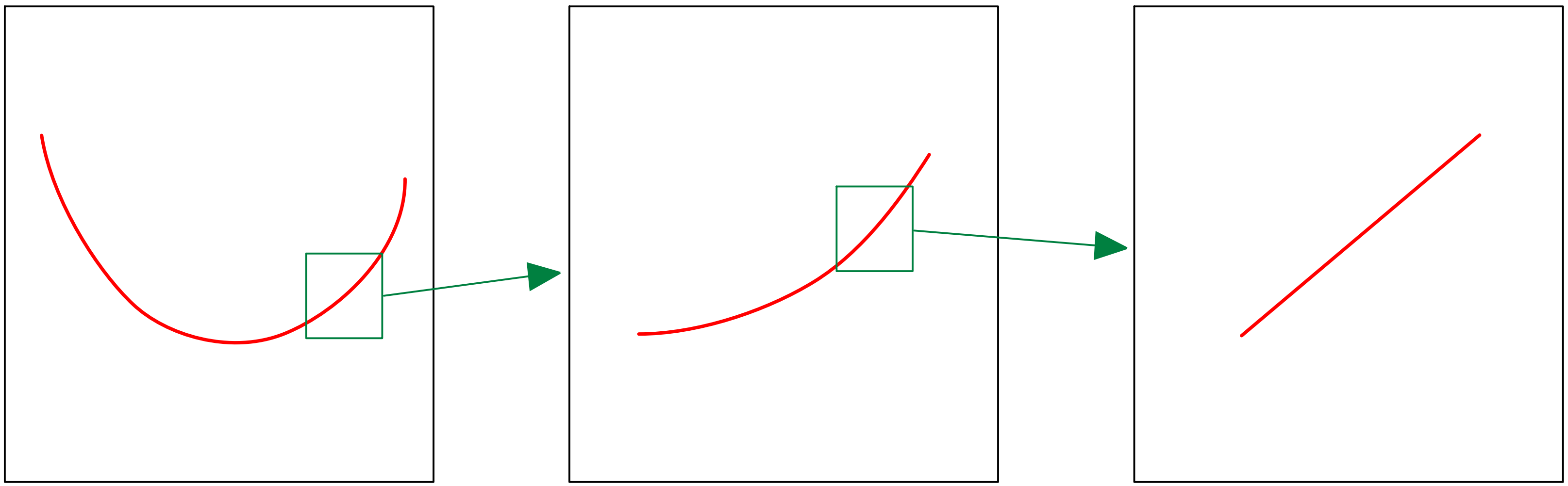
# Gradient descent: rationale

"Differentiable" means "locally linear".



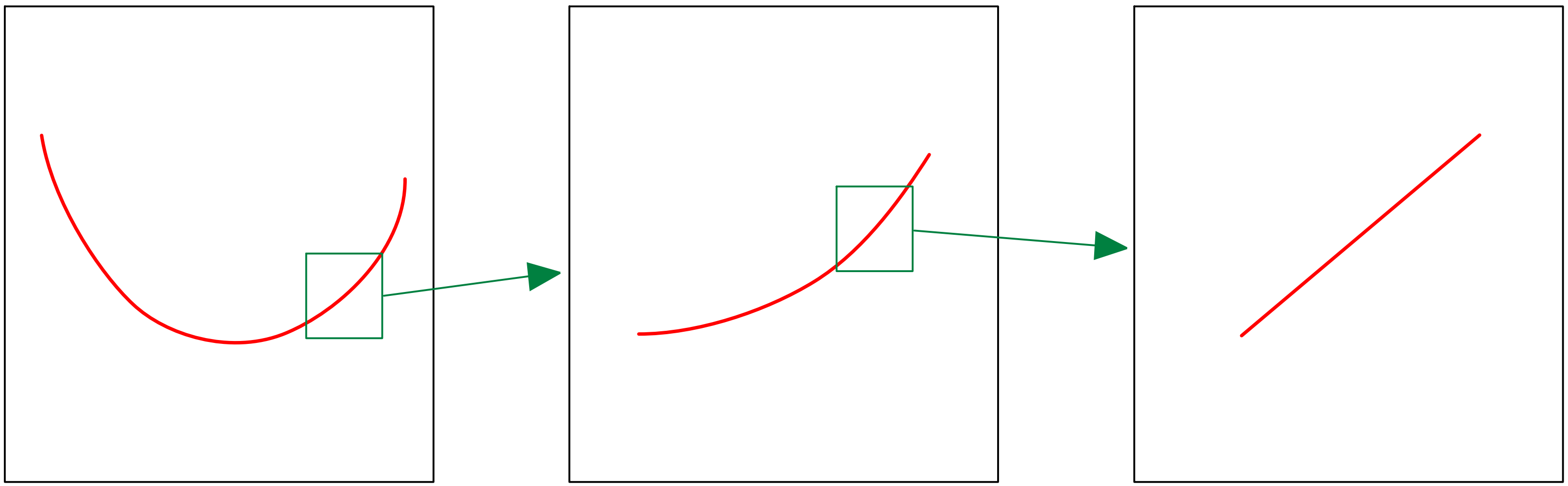
# Gradient descent: rationale

"Differentiable" means "locally linear".



# Gradient descent: rationale

"Differentiable" means "locally linear".



For *small* displacements $u \in \mathbb{R}^p$,

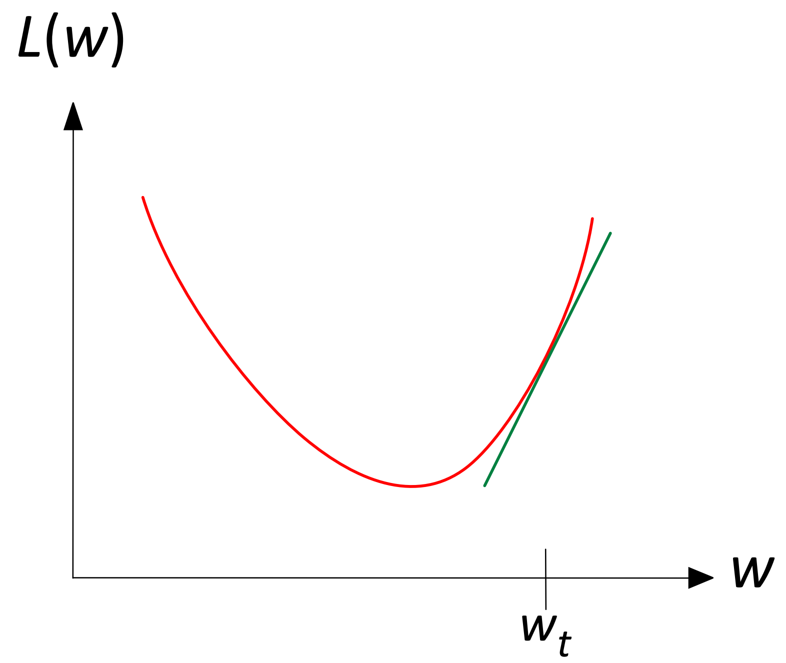
$$L(w+u) \;\approx\; L(w) + u \cdot \nabla L(w).$$

Therefore, if $u = -\eta \nabla L(w)$ is small,

$$L(w+u) \;\approx\; L(w) - \eta \|\nabla L(w)\|^2 \;<\; L(w).$$

# The step size matters

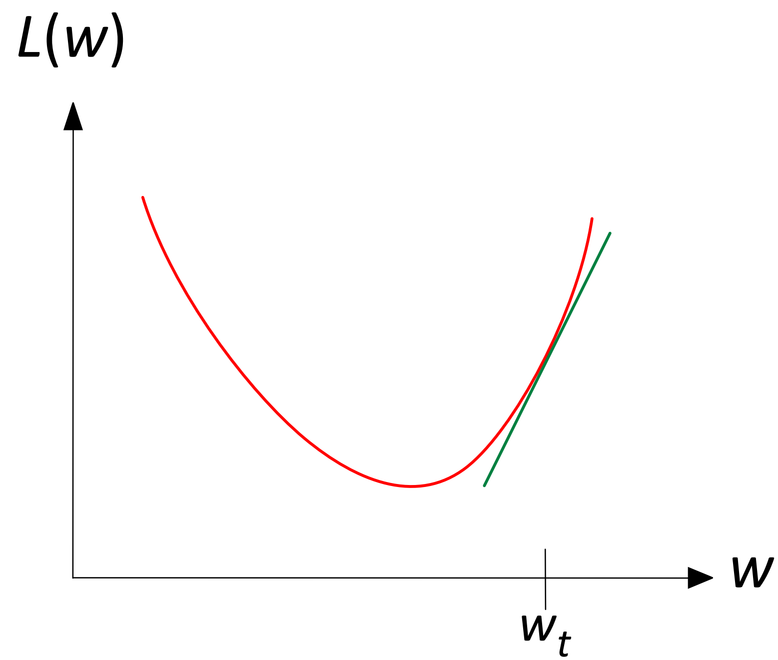
Gradient descent update: $w_{t+1} = w_t - \eta_t \nabla L(w_t)$.




- Step size $\eta_t$ too small: not much progress
- Too large: overshoot the mark

# The step size matters

Gradient descent update: $w_{t+1} = w_t - \eta_t \nabla L(w_t)$.




- Step size $\eta_t$ too small: not much progress
- Too large: overshoot the mark

One option: pick $\eta_t$ using a line search

$$\eta_t = \arg\min_{\alpha > 0} L(w_t - \alpha \nabla L(w_t)).$$

# Line Search

- Set $t = 0$, $\alpha = 0.1$ (or anything random between 0 and 1)

# Line Search

- Set $t = 0$, $\alpha = 0.1$ (or anything random between 0 and 1)
- Repeat:
  - Compute descent direction $\nabla L(w_t)$
  - Choose $\alpha_t$ to minimize $h(\alpha) = L(w_t - \alpha(\nabla L(w_t)))$ [calculus]
  - Update $w_{t+1} = w_t + \alpha(\nabla L(w_t))$

# Line Search

- Set $t = 0$, $\alpha = 0.1$ (or anything random between 0 and 1)
- Repeat:
    - Compute descent direction $\nabla L(w_t)$
    - Choose $\alpha_t$ to minimize $h(\alpha) = L(w_t - \alpha(\nabla L(w_t)))$ [calculus]
    - Update $w_{t+1} = w_t + \alpha(\nabla L(w_t))$
- Until $||\nabla L(w_t)|| <$ some tolerance

# Line Search

- Set $t = 0$, $\alpha = 0.1$ (or anything random between 0 and 1)
- Repeat:
  - Compute descent direction $\nabla L(w_t)$
  - Choose $\alpha_t$ to minimize $h(\alpha) = L(w_t - \alpha(\nabla L(w_t)))$ [calculus]
  - Update $w_{t+1} = w_t + \alpha(\nabla L(w_t))$
- Until $|| \nabla L(w_t)|| <$ some tolerance

This makes sure that $\alpha$ has a value that can lead us to the minimum value of $L(w_t)$.

# Variant: stochastic gradient descent

Recall gradient descent update for logistic regression: at time $t$

$$w_{t+1} = w_t + \eta_t \sum_{i=1}^{n} y^{(i)} x^{(i)} \underbrace{\Pr_{w_t}(-y^{(i)} | x^{(i)})}_{\text{doubt}_t(x^{(i)}, y^{(i)})}.$$

Each update involves the entire data set, which is inconvenient.

# Variant: stochastic gradient descent

Recall gradient descent update for logistic regression: at time $t$

$$w_{t+1} = w_t + \eta_t \sum_{i=1}^{n} y^{(i)} x^{(i)} \underbrace{\Pr_{w_t}(-y^{(i)}|x^{(i)})}_{\text{doubt}_t(x^{(i)},y^{(i)})}.$$

Each update involves the entire data set, which is inconvenient.

**Stochastic gradient descent** makes updates based on just one point:

- Get next data point $(x, y)$ (e.g., keep cycling through the data set)
- $w_{t+1} = w_t + \eta_t \, y \, x \Pr_{w_t}(-y|x)$.

Convenient for very large data sets.

# Variant: stochastic gradient descent

Recall gradient descent update for logistic regression: at time $t$

$$w_{t+1} = w_t + \eta_t \sum_{i=1}^{n} y^{(i)} x^{(i)} \underbrace{\Pr_{w_t}(-y^{(i)} | x^{(i)})}_{\text{doubt}_t(x^{(i)}, y^{(i)})}.$$

Each update involves the entire data set, which is inconvenient.

**Stochastic gradient descent** makes updates based on just one point:

- Get next data point $(x, y)$ (e.g., keep cycling through the data set)
- $w_{t+1} = w_t + \eta_t \, y \, x \Pr_{w_t}(-y|x)$.

Convenient for very large data sets.

Another option: make updates based on "mini-batches" of data points.

# Newton-Raphson

For minimizing a function $L(w)$:

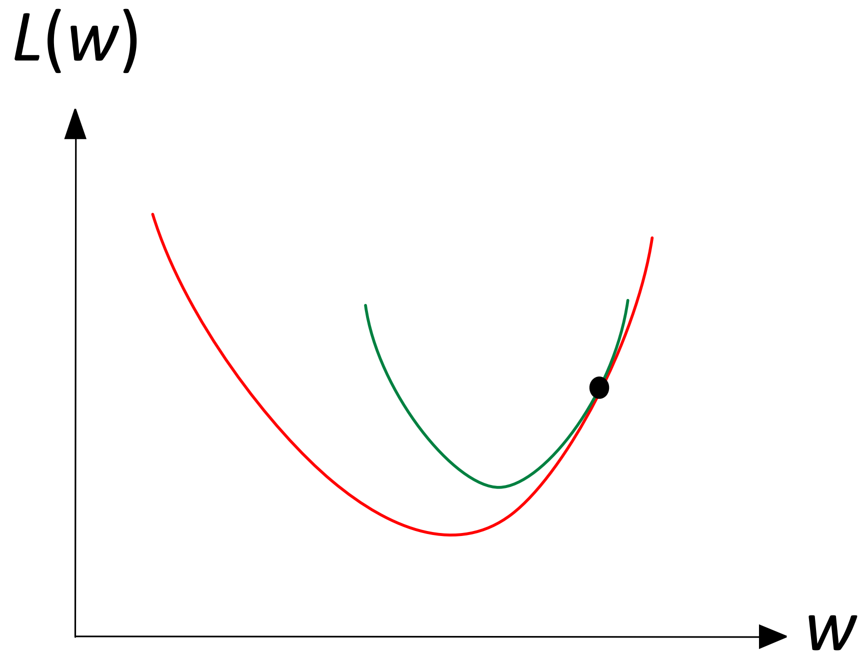
- $w_o = 0$, $t = 0$
- while $\nabla L(w_t) \not\approx 0$:
  - $w_{t+1} = w_t - \eta_t H^{-1}(w_t) \nabla L(w_t)$
  - $t = t + 1$

$H^{-1}(w)$ is the inverse of the Hessian at $w$

# Newton-Raphson: rationale

Second-order Taylor expansion: for small *u*,

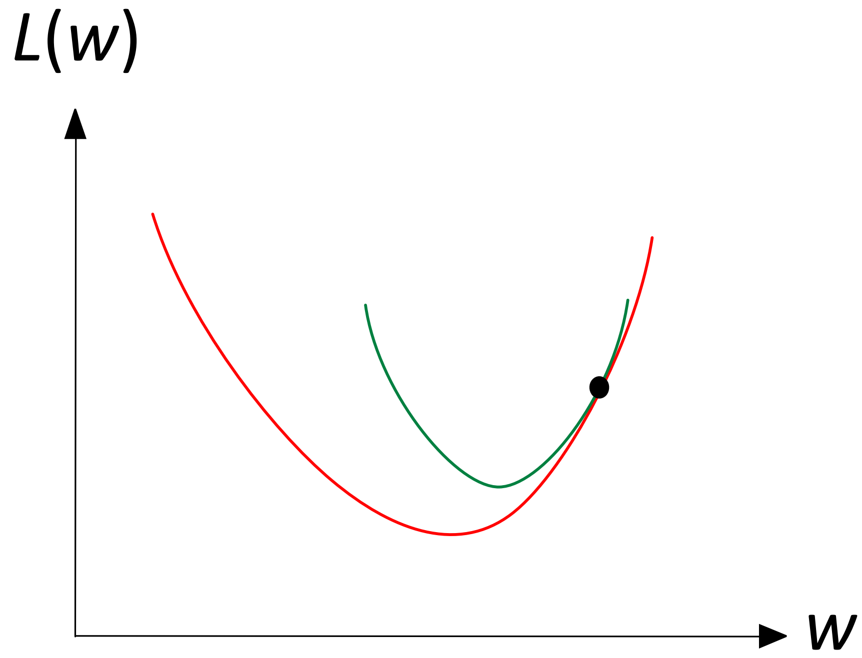
$$L(w+u) \;\approx\; L(w) + u \cdot \nabla L(w) + \frac{1}{2} u^T H(w) u.$$



# Newton-Raphson: rationale

Second-order Taylor expansion: for small *u*,

$$L(w+u) \approx L(w) + u \cdot \nabla L(w) + \frac{1}{2} u^T H(w) u.$$




To minimize this quadratic approximation, set derivative to zero:

$$\nabla L(w) + H(w)u = 0 \;\Rightarrow\; u = -H^{-1}(w)\nabla L(w).$$

# Variant: quasi-Newton methods

For optimizing a function over $\mathbf{R}^p$ , the Newton update involves computing the $p \times p$ Hessian at each time step:

$$w_{t+1} = w_t - \eta_t H^{-1}(w_t) \nabla L(w_t).$$

# Variant: quasi-Newton methods

For optimizing a function over $\mathbf{R}^p$ , the Newton update involves computing the $p \times p$ Hessian at each time step:

$$w_{t+1} = w_t - \eta_t H^{-1}(w_t)\nabla L(w_t).$$

Speed things up by using an approximation $B_t$ of $H^{-1}(w_t)$:

- Initialize it to the identity, say.
- Efficiently update $B_t \rightarrow B_{t+1}$ using a second-order approximation based on $w_{t+1}$, $w_t$, $\nabla L(w_t), \nabla L(w_{t+1})$.

Example: the BFGS (Broyden-Fletcher-Goldfarb-Shanno) procedure.

# Variant: quasi-Newton methods

For optimizing a function over $\mathbf{R}^p$ , the Newton update involves computing the $p \times p$ Hessian at each time step:

$$w_{t+1} = w_t - \eta_t H^{-1}(w_t) \nabla L(w_t).$$

Speed things up by using an approximation $B_t$ of $H^{-1}(w_t)$:

- Initialize it to the identity, say.
- Efficiently update $B_t \rightarrow B_{t+1}$ using a second-order approximation based on $w_{t+1}$, $w_t$, $\nabla L(w_t), \nabla L(w_{t+1})$.

Example: the BFGS (Broyden-Fletcher-Goldfarb-Shanno) procedure.

Even better: instead of maintaining a dense matrix $B_t$, use an $O(p)$ – Sized approximation to it. Example: L-BFGS ("Limited memory BFGS").