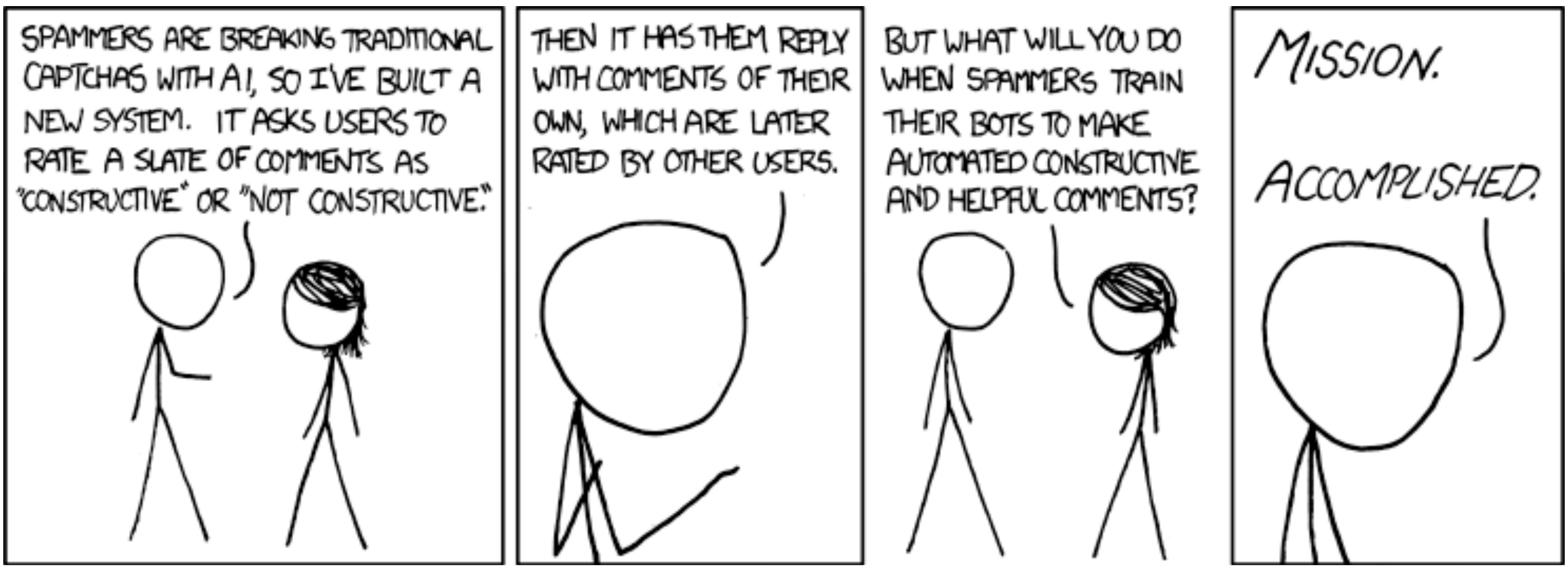
Linear models: Logistic regression

Chapter 3.3



Predicting probabilities

Objective: learn to predict a probability $P(y \mid x)$ for a binary classification problem using a linear classifier

The target function: $$\mathbb{P}[y = +1 \mid \mathbf{x}].$$

For positive examples $P(y = +1 \mid x) = 1$ whereas $P(y = +1 \mid x) = 0$ for negative examples.

Predicting probabilities

Objective: learn to predict a probability $P(y \mid x)$ for a binary classification problem using a linear classifier

The target function: $$\mathbb{P}[y = +1 \mid \mathbf{x}].$$

For positive examples $P(y = +1 \mid x) = 1$ whereas $P(y = +1 \mid x) = 0$ for negative examples.

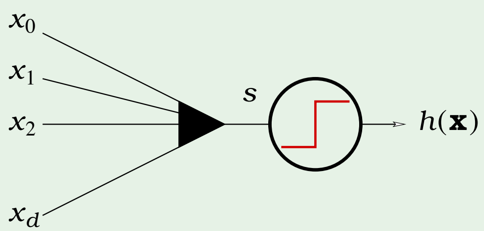
Can we assume that $P(y = +1 \mid x)$ is linear?

Logistic regression

The signal $s = \mathbf{w}^\mathsf{T}\mathbf{x}$ is the basis for several linear models:

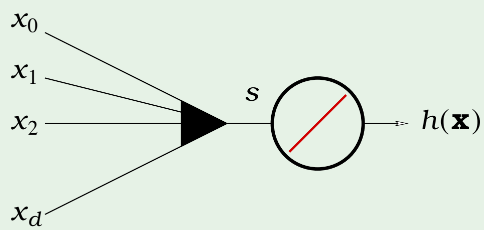
linear classification

$$h(\mathbf{x}) = \text{sign}(s)$$



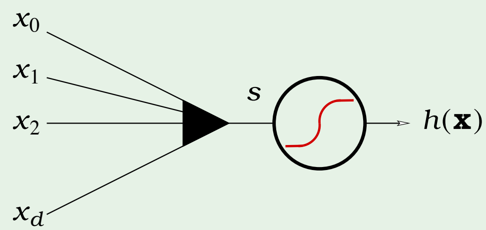
linear regression

$$h(\mathbf{x}) = s$$



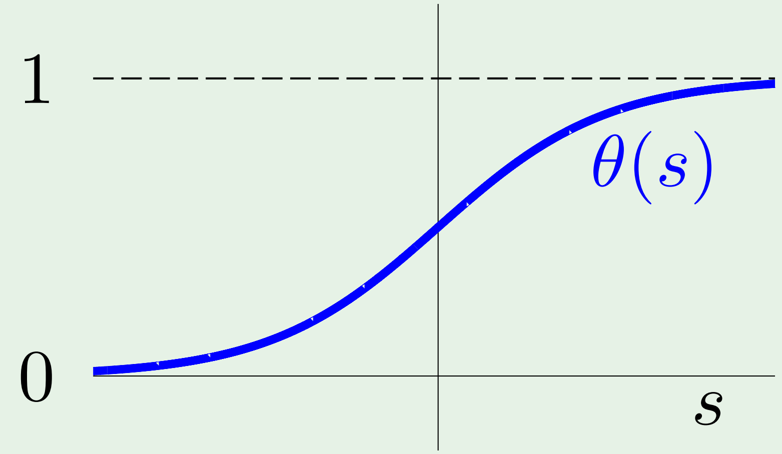
logistic regression

$$h(\mathbf{x}) = \theta(s)$$



The logistic function (aka squashing function):

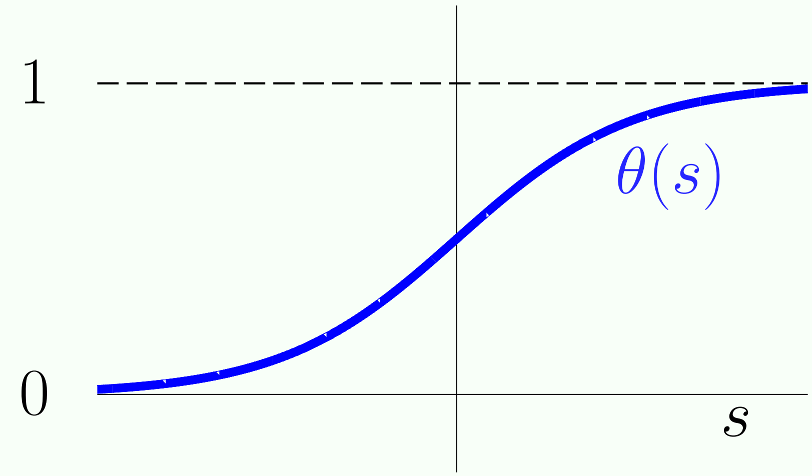
$$\theta(s) = \frac{e^s}{1 + e^s}$$



Properties of the logistic function

$$\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}.$$

$$\theta(-s) = \frac{e^{-s}}{1+e^{-s}} = \frac{1}{1+e^s} = 1 - \theta(s).$$



Predicting probabilities

Fitting the data means finding a good hypothesis h

$$h \text{ is good if: } \begin{cases} h(\mathbf{x}_n) \approx 1 & \text{whenever } y_n = +1; \\ h(\mathbf{x}_n) \approx 0 & \text{whenever } y_n = -1. \end{cases}$$

Suppose that $h(\mathbf{x}) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ closely captures $\mathbb{P}[+1|\mathbf{x}]$:

$$P(y \mid \mathbf{x}) = \begin{cases} \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = +1; \\ 1 - \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = -1. \end{cases}$$

Predicting probabilities

Fitting the data means finding a good hypothesis h

$$h \text{ is good if: } \quad \begin{cases} h(\mathbf{x}_n) \approx 1 & \text{whenever } y_n = +1; \\ h(\mathbf{x}_n) \approx 0 & \text{whenever } y_n = -1. \end{cases}$$

Suppose that $h(\mathbf{x}) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ closely captures $\mathbb{P}[+1|\mathbf{x}]$:

$$P(y \mid \mathbf{x}) = \begin{cases} \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = +1; \\ \theta(-\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = -1. \end{cases}$$

More compactly: $P(y \mid \mathbf{x}) = \theta(y \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x})$

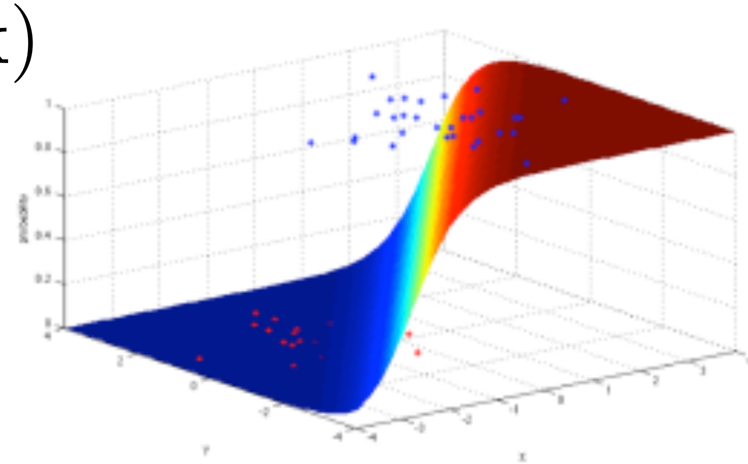
Is logistic regression really linear?

$$P(y=+1|\mathbf{x}) = \frac{\exp(\mathbf{w}^\intercal \mathbf{x})}{\exp(\mathbf{w}^\intercal \mathbf{x})+1}$$

$$P(y=-1|\mathbf{x}) = 1 - P(y=+1|\mathbf{x}) = \frac{1}{\exp(\mathbf{w}^\intercal \mathbf{x})+1}$$

To figure out how the decision boundary looks like set

$$P(y=+1|\mathbf{x}) = P(y=-1|\mathbf{x})$$



solving for x we get:

$$\exp(\mathbf{w}^\intercal \mathbf{x}) = 1$$

i.e. $\mathbf{w}^\intercal \mathbf{x} = 0$

Maximum likelihood

We will find **w** using the principle of maximum likelihood.

Likelihood:
The probability of getting the $y_1, \ldots, y_N$ in $\mathcal{D}$ from the corresponding $\mathbf{x}_1, \ldots, \mathbf{x}_N$:

$$P(y_1, \ldots, y_N \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n).$$

Valid since $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ are independently generated

Maximizing the likelihood

$$
\begin{aligned}
& \max && \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \max && \ln\left(\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n)\right) \\
\equiv\ & \max && \sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow\ & \min && -\frac{1}{N}\sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{P(y_n|\mathbf{x}_n)} \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{\theta(y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n)} \\
\equiv\ & \min && \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)
\end{aligned}
$$

Maximizing the likelihood

Summary: maximizing the likelihood is equivalent to

minimize $$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \underbrace{\ln\left(1 + e^{-y_n \mathbf{w}^{\mathsf{T}}\mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n), y_n\right)}$$

Cross entropy error

Maximizing the likelihood

Summary: maximizing the likelihood is equivalent to

minimize

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \underbrace{\ln\left(1 + e^{-y_n \mathbf{w}^{\mathsf{T}} \mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n), y_n\right)}$$

Cross entropy error

Exercise: check that this is equivalent to:

$$E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} I(y_n = +1)\ln\frac{1}{h(\mathbf{x}_n)} + I(y_n = -1)\ln\frac{1}{1 - h(\mathbf{x}_n)}$$

Digression: information theory

I am thinking of an integer between 0 and 1,023. You want to guess it using the fewest number of questions.

Most of us would ask "*is it between 0 and 512?*"

This is a good strategy because it provides the most information about the unknown number.

It provides the first binary digit of the number.

Initially you need to obtain $log_2(1024) = 10$ bits of information. After the first question you only need $log_2(512) = 9$ bits.

Information and Entropy

By halving the search space we obtained one bit.

In general, the information associated with a probabilistic outcome:

$$I(p) = -\log p$$

Why the logarithm?

Assume we have two independent events x, and y. We would like the information they carry to be additive. Let's check:

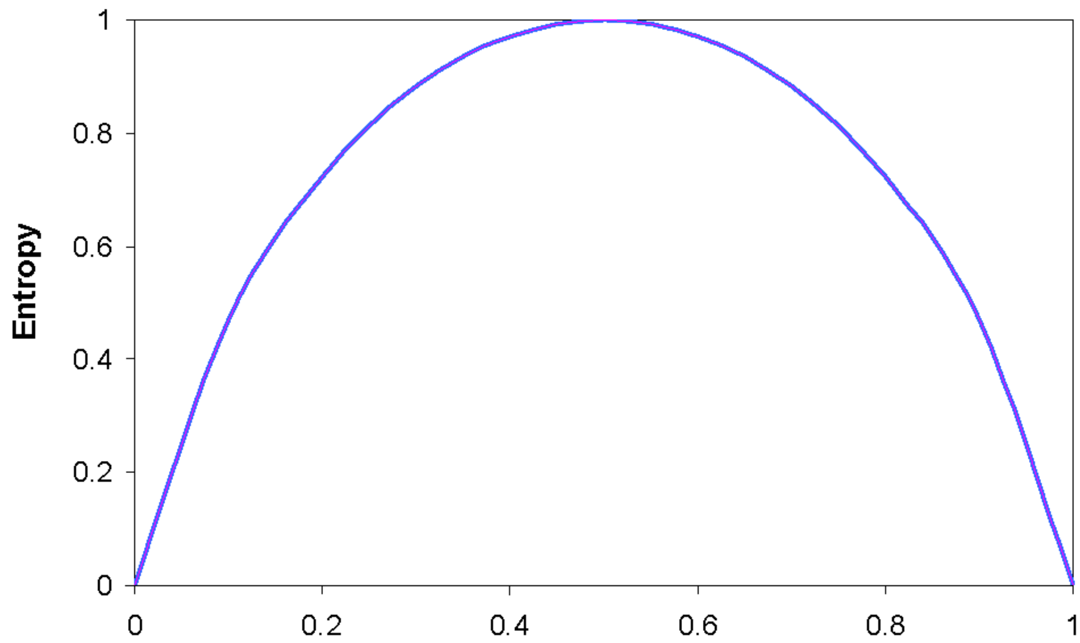
$$\begin{aligned} I(x,y) &= -\log P(x,y) = -\log P(x)P(y) \\ &= -\log P(x) - \log P(y) = I(x) + I(y) \end{aligned}$$

Entropy: $$H(P) = -\sum_x P(x) \log P(x)$$

Entropy

For a Bernoulli random variable:

$$H(p) = -p \log p - (1-p) \log(1-p)$$



Maximal when p = ½.

KL divergence

The KL divergence between distributions P and Q:

$$D_{KL}(P||Q) = -\sum_x P(x) \log \frac{Q(x)}{P(x)}$$

Properties:

- Non-negative, equal to 0 iff P = Q
- It is not symmetric

KL divergence

The KL divergence between distributions P and Q:

$$D_{KL}(P||Q) = -\sum_x P(x) \log \frac{Q(x)}{P(x)}$$

$$D_{KL}(P||Q) = -\sum_x P(x) \log Q(x) + \sum_x P(x) \log P(x)$$

cross entropy - entropy

Cross entropy and logistic regression

The logistic regression cost function:

$$E_{in}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} I(y_n = +1)\ln\frac{1}{h(\mathbf{x}_n)} + I(y_n = -1)\ln\frac{1}{1-h(\mathbf{x}_n)}$$

It is the average cross entropy between the learned $P(y \mid x)$ and the observed probabilities

Cross entropy $$H(P,Q) = -\sum_x P(x)\log Q(x)$$

And for binary variables:

$$H(y,\hat{y}) = -y\log\hat{y} - (1-y)\log(1-\hat{y})$$

In-sample error

The in-sample error for logistic regression

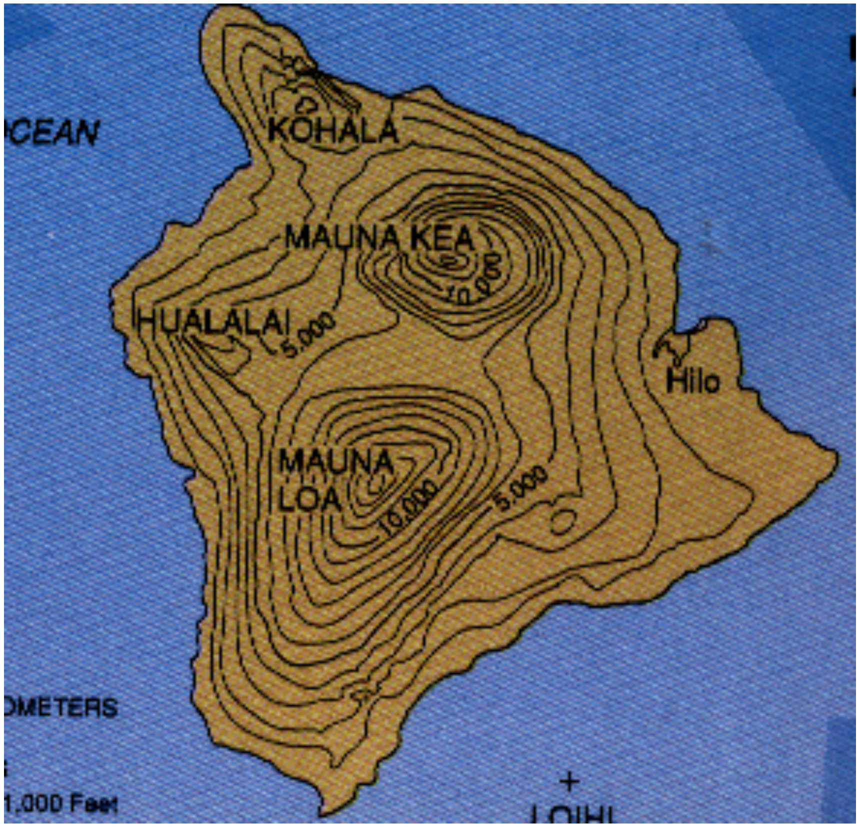
minimize $$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \underbrace{\ln\left(1 + e^{-y_n \mathbf{w}^{\mathsf{T}} \mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n), y_n\right)}$$

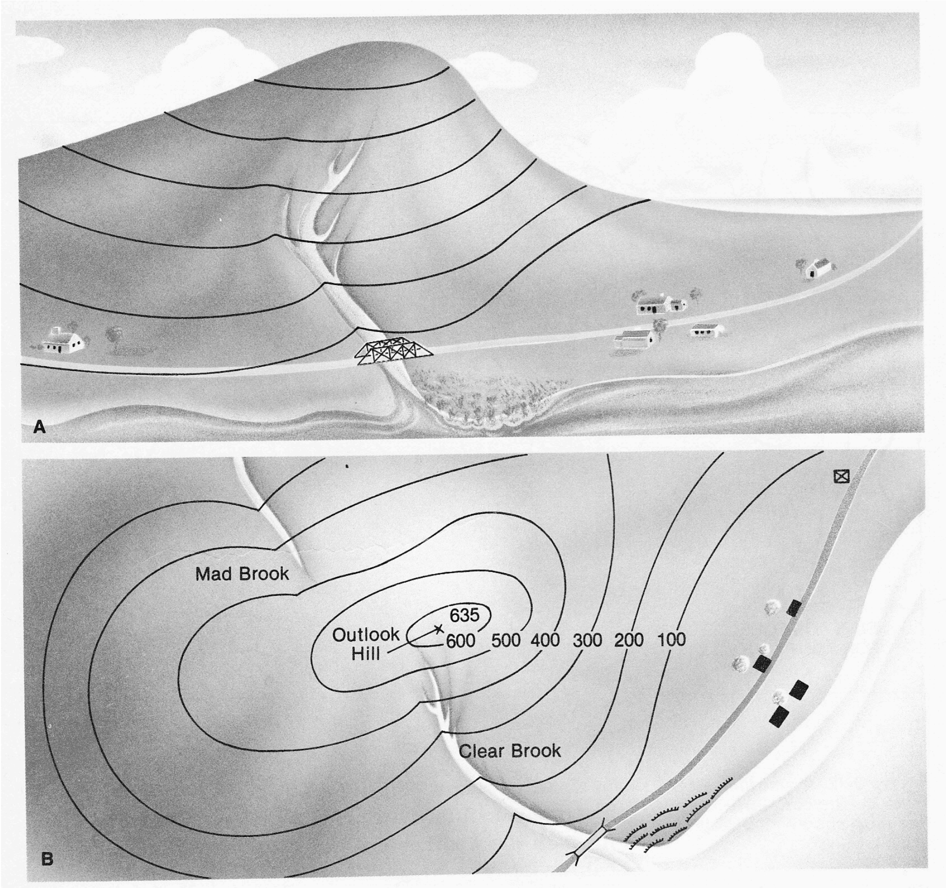
Cross entropy error

$$\nabla E_{\text{in}} = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \mathbf{x}_n}{1 + e^{y_n \mathbf{w}^{\mathsf{T}} \mathbf{x}_n}}$$

Digression: gradient ascent/descent

Topographical maps can give us intuition on how to optimize a cost function



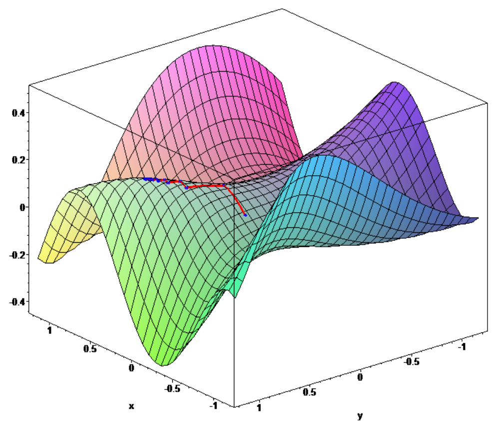


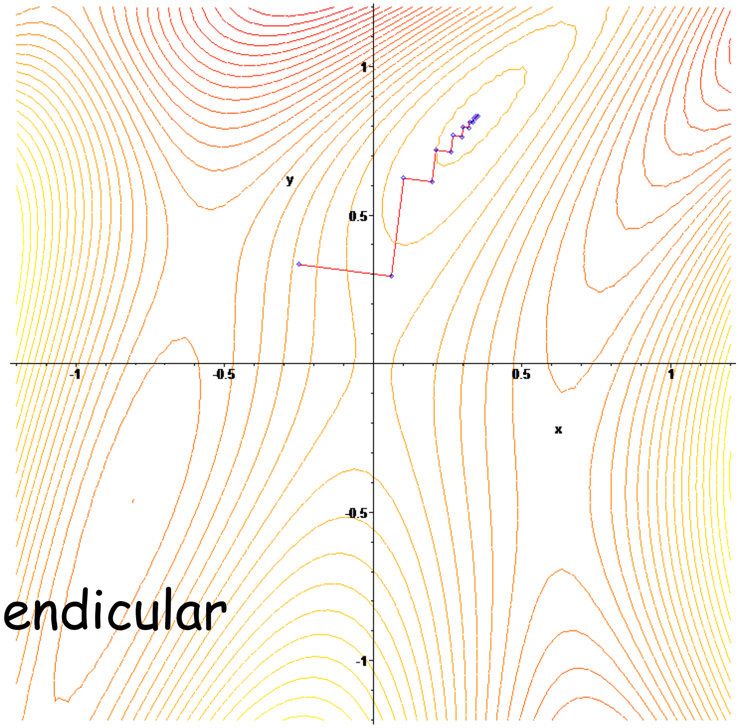
http://www.csus.edu/indiv/s/slaymaker/archives/geol10l/shield1.jpg

http://www.sir-ray.com/touro/IMG_0001_NEW.jpg

Digression: gradient descent

Given a function E(**w**), the gradient is the direction of steepest ascent
Therefore to minimize E(**w**), take a step in the direction of the negative of the gradient





Notice that the gradient is perpendicular to contours of equal E(**w**)

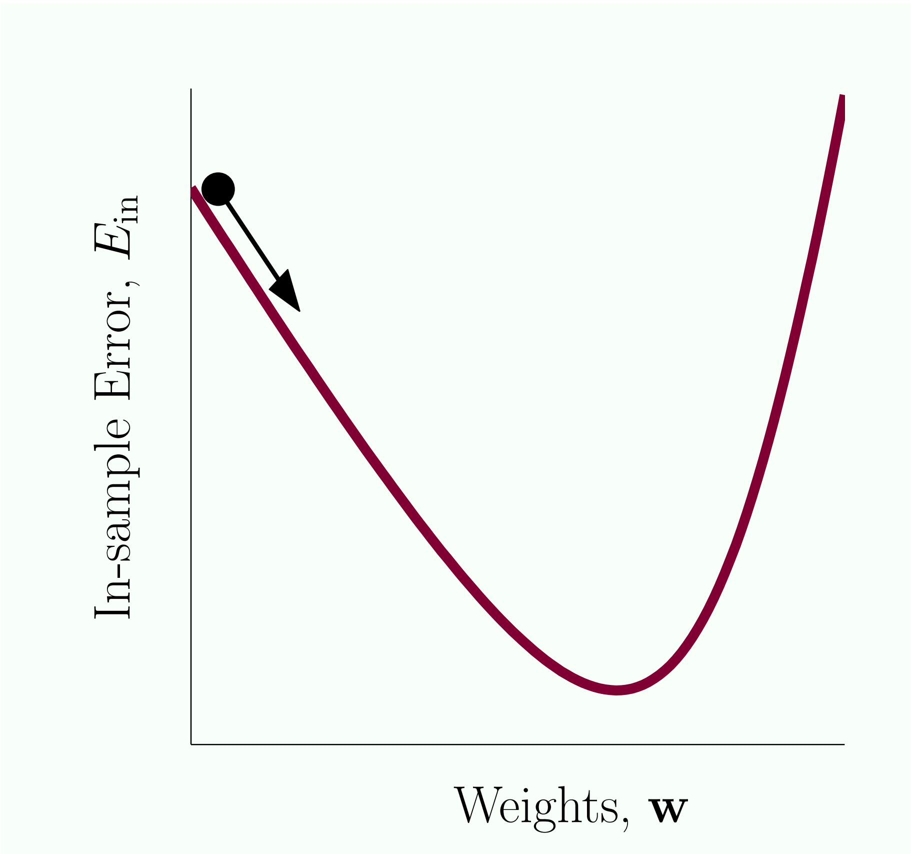
Images from http://en.wikipedia.org/wiki/Gradient_descent

Gradient descent

Gradient descent is an iterative process

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\hat{\mathbf{v}}$$

How to pick $\hat{\mathbf{v}}$?



Gradient descent

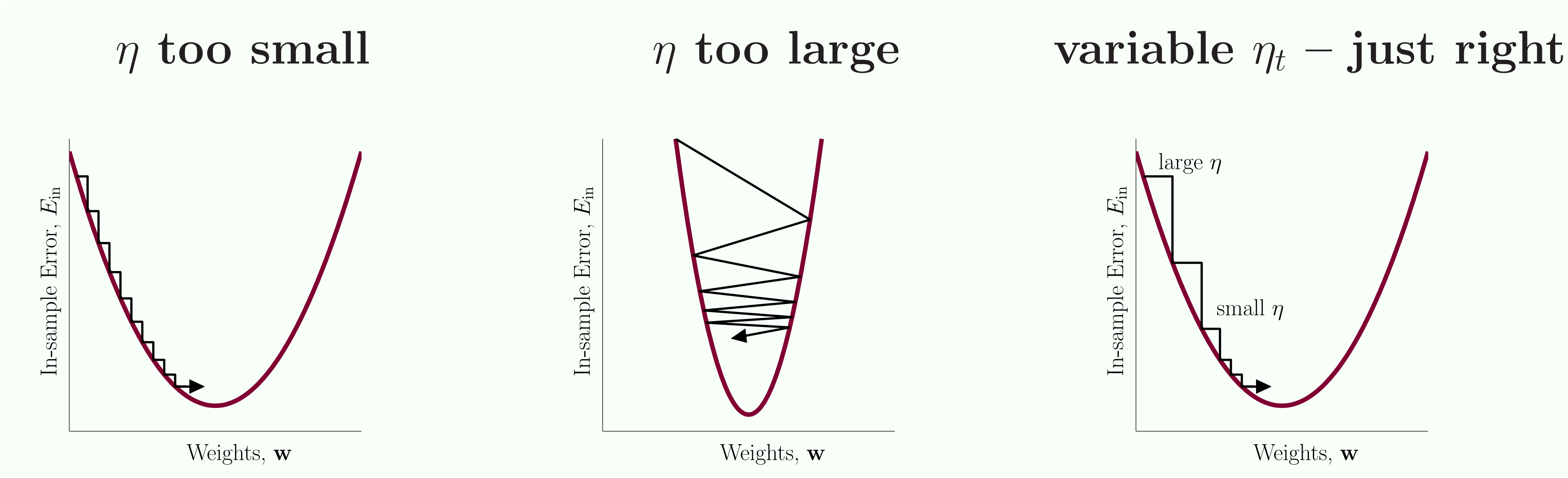
The gradient is the best direction to take to optimize $E_{in}(\mathbf{w})$:

$$\begin{aligned} \Delta E_{\text{in}} &= E_{\text{in}}(\mathbf{w}(t+1)) - E_{\text{in}}(\mathbf{w}(t)) \\ &= E_{\text{in}}(\mathbf{w}(t) + \eta\hat{\mathbf{v}}) - E_{\text{in}}(\mathbf{w}(t)) \\ &= \eta \, \underbrace{\nabla E_{\text{in}}(\mathbf{w}(t))^{\mathrm{T}}\hat{\mathbf{v}}} + O(\eta^2) \end{aligned}$$

$$\text{minimized at } \hat{\mathbf{v}} = -\frac{\nabla E_{\text{in}}(\mathbf{w}(t))}{\| \nabla E_{\text{in}}(\mathbf{w}(t)) \|}$$

Choosing the step size

The choice of the step size affects the rate of convergence:



Let's use a variable learning rate:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta_t \hat{\mathbf{v}}$$

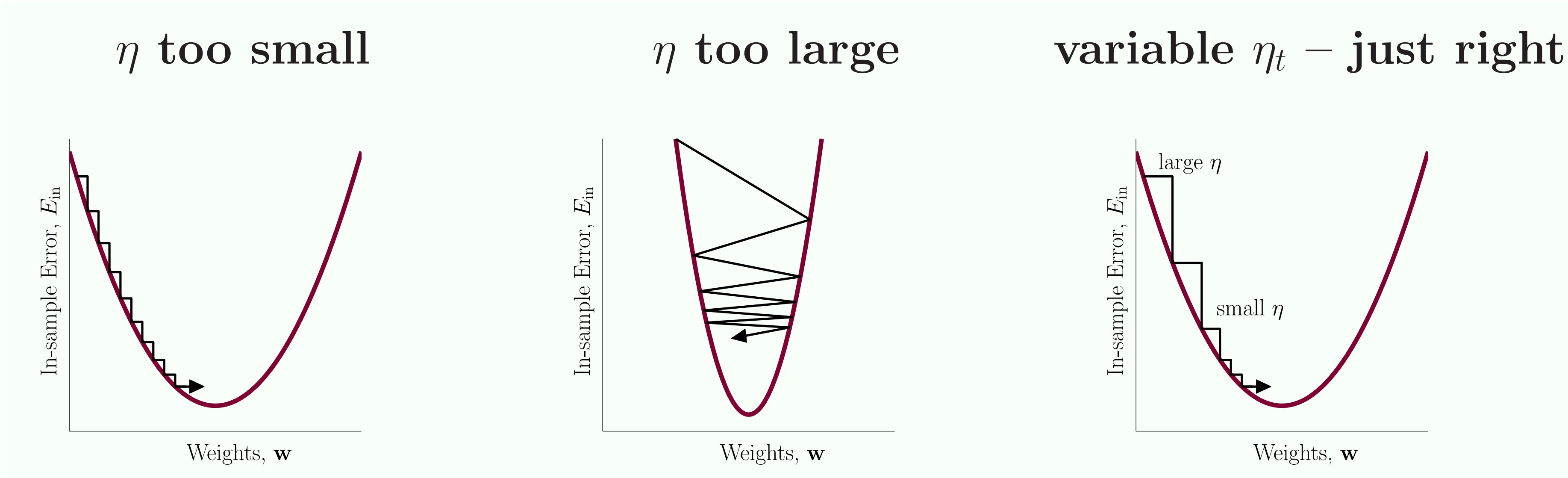
$$\eta_t = \eta \cdot ||\nabla E_{\text{in}}(\mathbf{w}(t))||$$

When approaching the minimum:

$$||\nabla E_{\text{in}}(\mathbf{w}(t))|| \to 0$$

Choosing the step size

The choice of the step size affects the rate of convergence:



Let's use a variable learning rate:

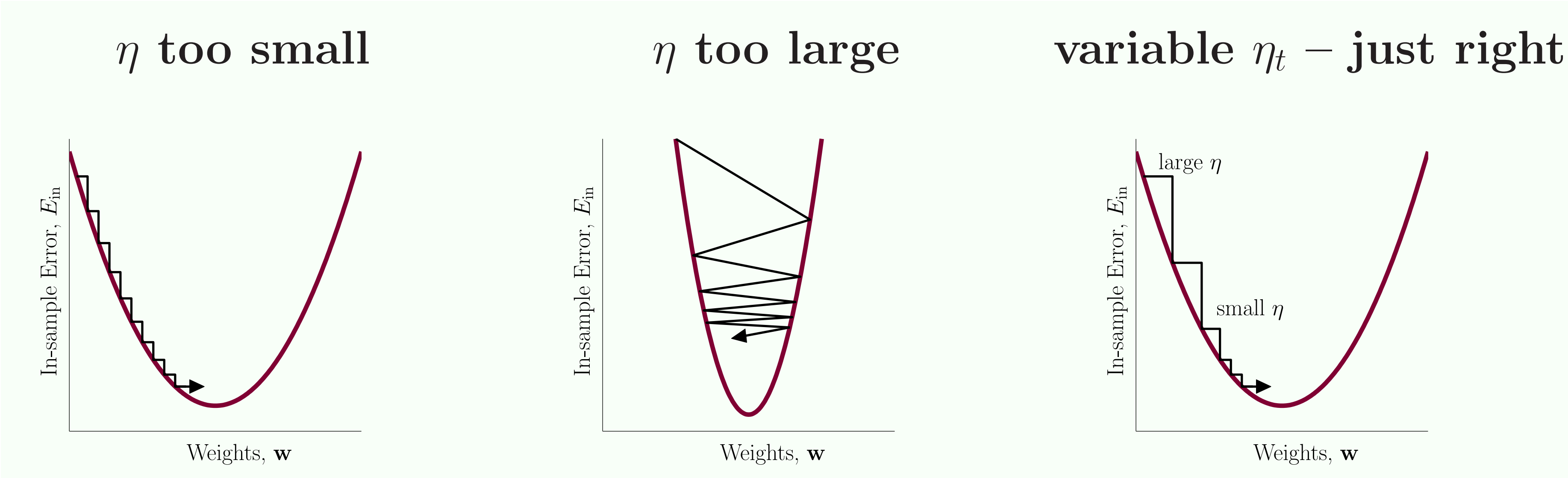
$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta_t \hat{\mathbf{v}}$$

$$\eta_t = \eta \cdot ||\nabla E_{\text{in}}(\mathbf{w}(t))||$$

$$\eta_t \hat{\mathbf{v}} = -\eta \cdot ||\nabla E_{\text{in}}(\mathbf{w}(t))|| \cdot \frac{\nabla E_{\text{in}}(\mathbf{w}(t))}{||\nabla E_{\text{in}}(\mathbf{w}(t))||} = -\eta \nabla E_{\text{in}}(\mathbf{w}(t))$$

The final form of gradient descent

The choice of the step size affects the rate of convergence:

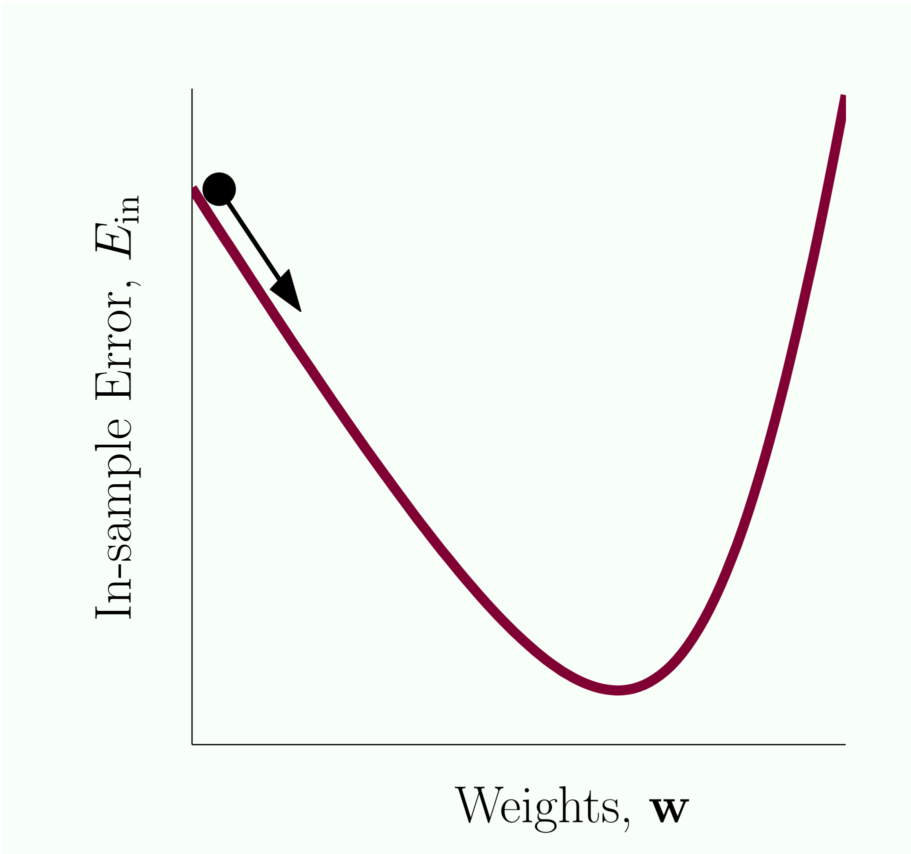


$$\mathbf{w}(t+1) = \mathbf{w}(t) - \eta \nabla E_{\text{in}}(\mathbf{w}(t))$$

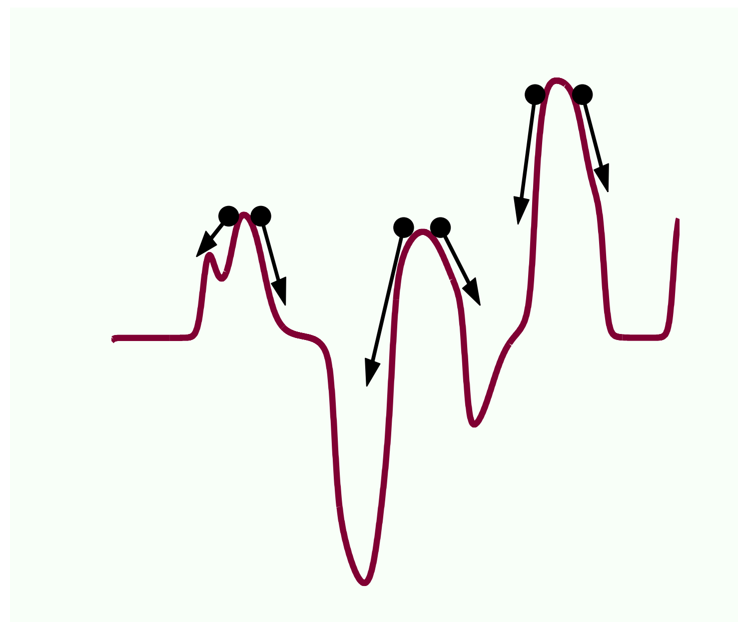
Logistic regression using gradient descent

We will use gradient descent to minimize our error function.

Fortunately, the logistic regression error function has a single global minimum:



So we don't need to worry about getting stuck in local minima



Logistic regression using gradient descent

Putting it all together:

Initialize at step $t = 0$ to $\mathbf{w}(0)$.
for $t = 0, 1, 2, \ldots$ **do**
Compute the gradient

$$\mathbf{g}_t = \nabla E_{\text{in}}(\mathbf{w}(t)).$$

Move in the direction $\mathbf{v}_t = -\mathbf{g}_t$.
Update the weights:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta \mathbf{v}_t.$$

Iterate 'until it is time to stop'.
end for
Return the final weights.

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} \underbrace{\ln\left(1 + e^{-y_n \mathbf{w}^{\intercal} \mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n), y_n\right)}$$

$$\nabla E_{\text{in}} = -\frac{1}{N} \sum_{n=1}^{N} \frac{y_n \mathbf{x}_n}{1 + e^{\,y_n \mathbf{w}^{\intercal}(t) \mathbf{x}_n}}$$

This is called batch gradient descent

Logistic regression

Comments:

- In practice logistic regression is solved by faster methods than gradient descent
- There is an extension to multi-class classification

Stochastic gradient descent

Variation on gradient descent that considers the error for a single training example:

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}}) = \frac{1}{N}\sum_{n=1}^{N} e(\mathbf{w}, \mathbf{x}_n, y_n)$$

Pick a random data point $(\mathbf{x}_*, y_*)$

Run an iteration of GD on $e(\mathbf{w}, \mathbf{x}_*, y_*)$

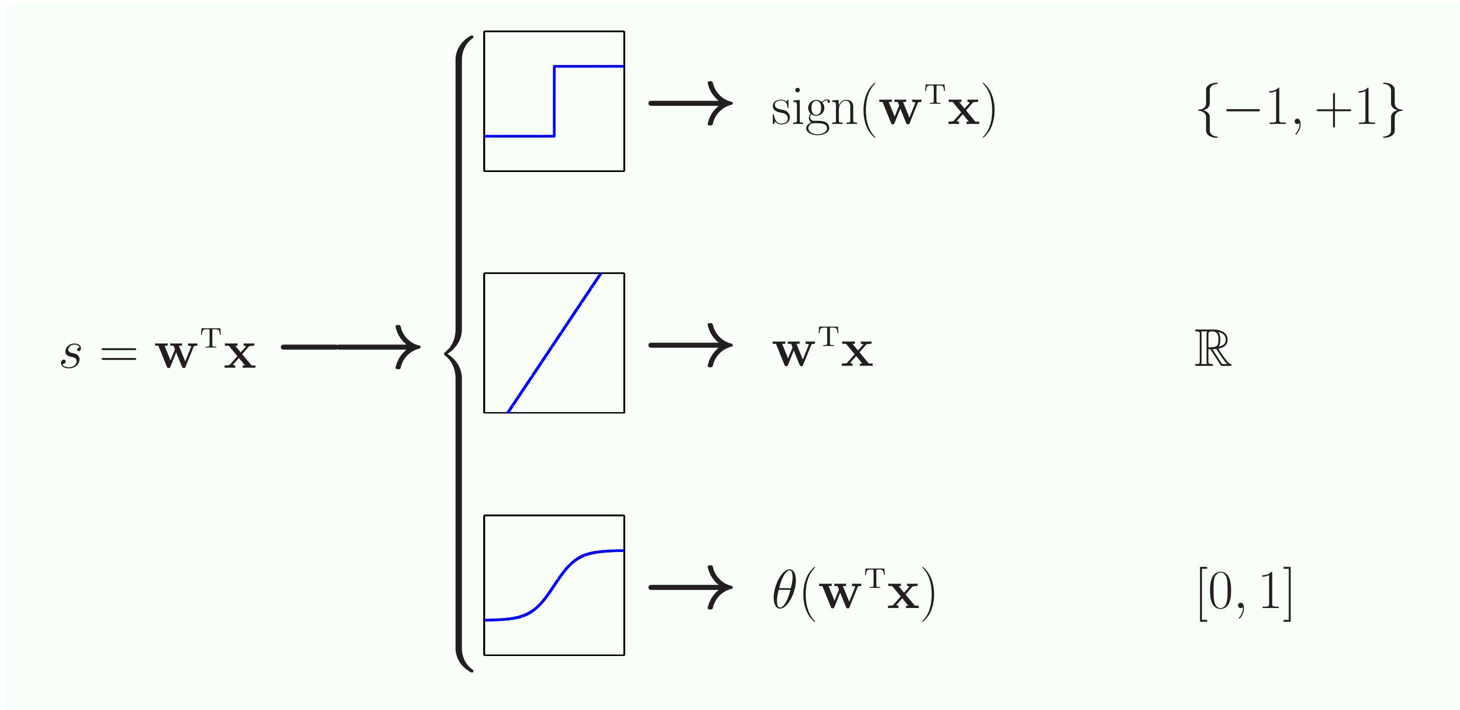
$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) - \eta \nabla_{\mathbf{w}} e(\mathbf{w}, \mathbf{x}_*, y_*)$$

$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_* \mathbf{x}_* \left(\frac{\eta}{1 + e^{y_* \mathbf{w}^{\mathrm{T}} \mathbf{x}_*}} \right)$$

Tends to converge faster than the batch version.

Summary of linear models

Linear methods for classification and regression:



More to come!