NYU Center for Data Science

DS-GA 3001.007
Introduction to Machine Learning

Lecture 5
Linear Regression

Reminders

- Survey 2
  - Please respond by October 7

Reminders

- Survey 2
  - Please respond by October 7
- Homework 2
  - Please submit by October 5
  - Contact Ravi and Raghav through Messages

Reminders

- Survey 2
  - Please respond on Qualtrics by October 7
- Homework 2
  - Please submit to Gradescope by October 3
  - Contact Ravi and Raghav through Messages
- Final
  - The final exam is scheduled for December 18 12-1:50pm.

Reminders

- Project
  - Proposal due October 31
  - Milestone due November 28
  - Report due December 15
- Post to Forum about groups…or random assignment.

Agenda

- Review
- Lesson
- Demo

Agenda

- Review
  - Generalizing from In Sample to Out of Sample
  - How many samples needed for hypothesis class?
- Lesson
- Demo

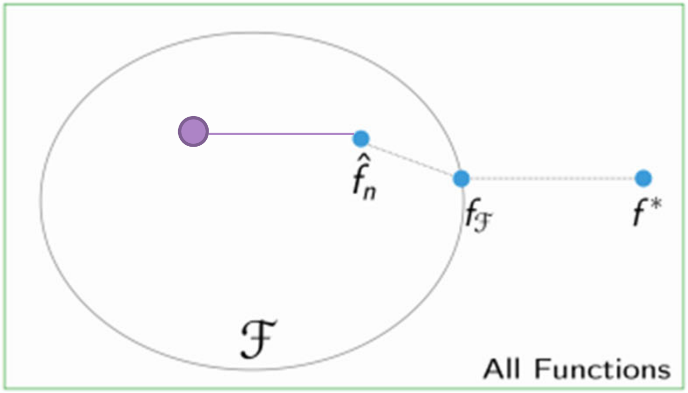
Agenda

- Review
- Lesson
  - Minimize In Sample Error
  - Square Loss, Absolute Loss, 0-1 Loss
- Demo

Agenda

- Review
- Lesson
- Demo
  - Gradient Descent
  - Stochastic Gradient Descent

Risk Decomposition



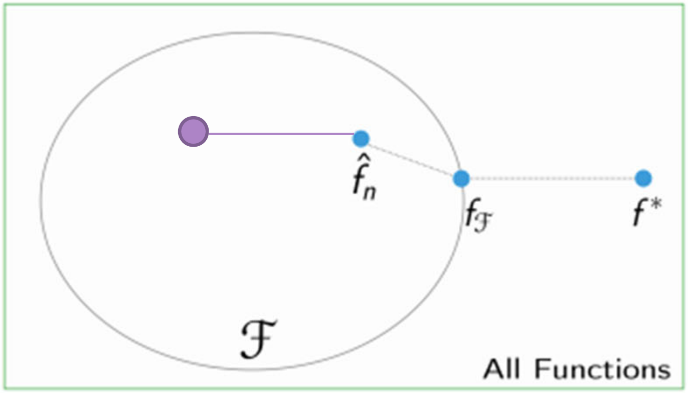
$f_{\text{opt}} = \text{Hypothesis determined by algorithm}$

$$f^* = \arg\min_{f} \mathbb{E}\ell(f(x), y)$$

$$f_{\mathcal{F}} = \arg\min_{f \in \mathcal{F}} \mathbb{E}\ell(f(x), y))$$

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

Risk Decomposition



$$f_{\text{opt}} = \text{Hypothesis determined by algorithm}$$

$$f^* = \arg\min_{f} \mathbb{E}\ell(f(x), y)$$

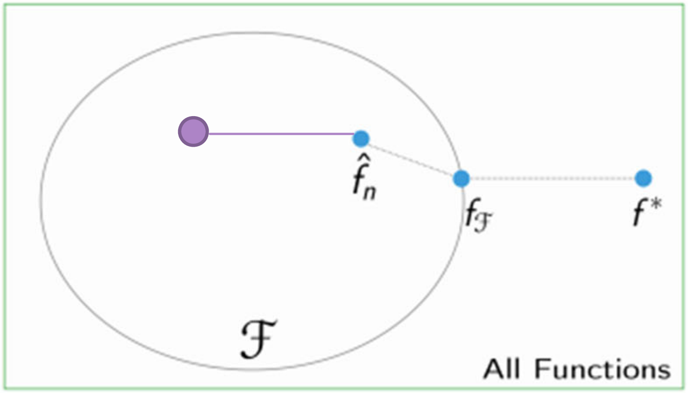
$$f_{\mathcal{F}} = \arg\min_{f \in \mathcal{F}} \mathbb{E}\ell(f(x), y))$$

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

- **Approximation Error** (of $\mathcal{F}$) $= \ R(f_{\mathcal{F}}) - R(f^*)$
- **Estimation error** (of $\hat{f}_n$ in $\mathcal{F}$) $= \ R(\hat{f}_n) - R(f_{\mathcal{F}})$

$$\textbf{Optimization Error} = R(f_{\text{opt}}) - R(\hat{f}_n)$$

Risk Decomposition



$f_{\text{opt}} = \text{Hypothesis determined by algorithm}$

$$f^* = \arg\min_{f} \mathbb{E}\ell(f(x), y)$$

$$f_{\mathcal{F}} = \arg\min_{f \in \mathcal{F}} \mathbb{E}\ell(f(x), y))$$

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), y_i)$$

- **Approximation Error** (of $\mathcal{F}$) $= \; R(f_{\mathcal{F}}) - R(f^*)$
- **Estimation error** (of $\hat{f}_n$ in $\mathcal{F}$) $= \; R(\hat{f}_n) - R(f_{\mathcal{F}})$

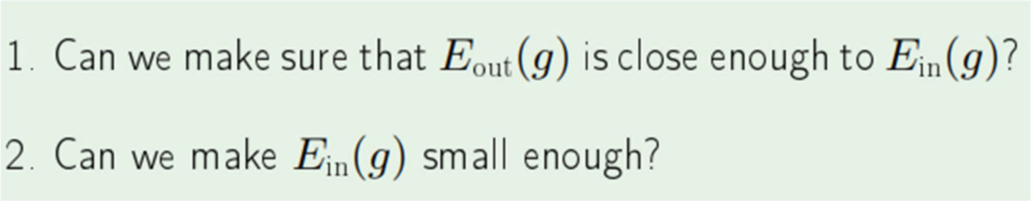
$\text{Optimization Error} = R(f_{\text{opt}}) - R(\hat{f}_n)$

$$\begin{aligned} \text{Excess Risk} &= R(f_{\text{opt}}) - R(f^*) \\ &= R(f_{\text{opt}}) - R(\hat{f}_n) \\ &\quad + R(\hat{f}_n) - R(f_{\mathcal{F}}) \\ &\quad + R(f_{\mathcal{F}}) - R(f^*) \end{aligned}$$

Question

- Large Scale and Small Scale
- Try to best characterize each of the following in terms of one or more of optimization error, approximation error, and estimation error.
  - Overfitting.
  - Underfitting.
  - Precise empirical risk minimization for your hypothesis space is computationally intractable.
  - Not enough data.

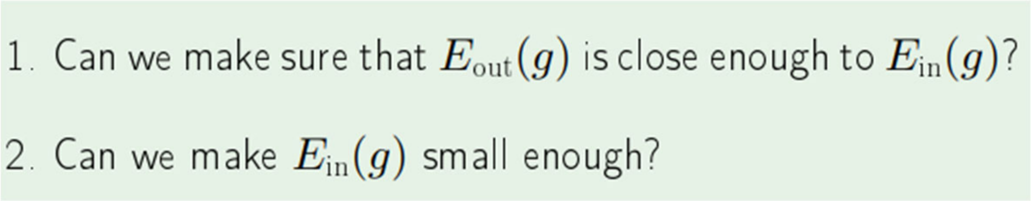
Risk Decomposition



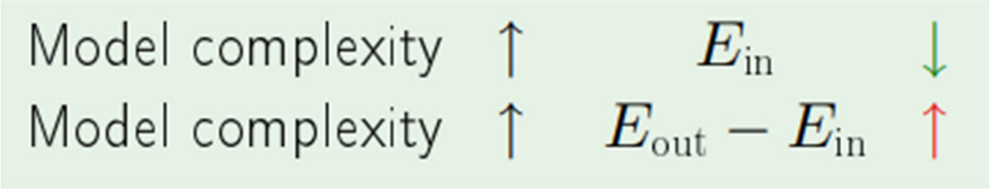
1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?

2. Can we make $E_{\text{in}}(g)$ small enough?

Risk Decomposition

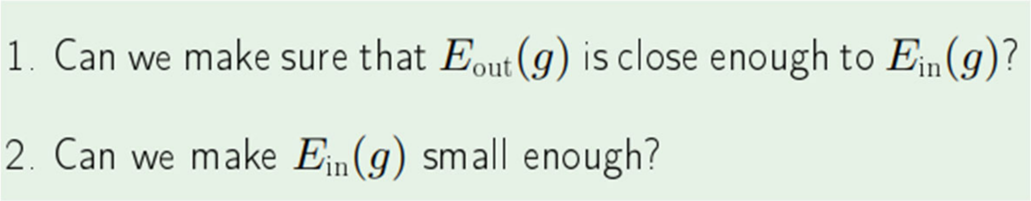


1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?
2. Can we make $E_{\text{in}}(g)$ small enough?

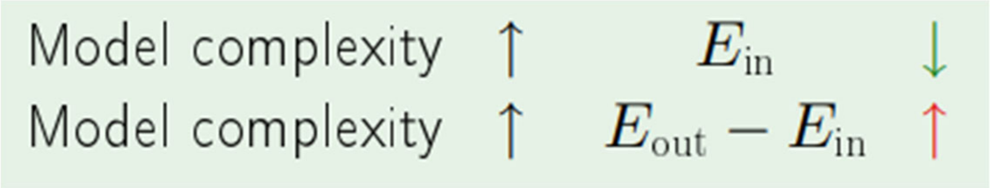


Model complexity	↑	E_{in}	↓
Model complexity	↑	$E_{\text{out}} - E_{\text{in}}$	↑

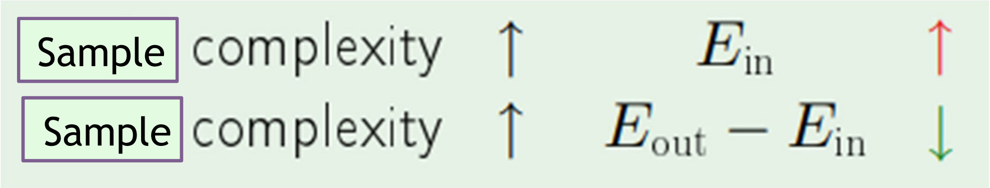
Risk Decomposition



1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?
2. Can we make $E_{\text{in}}(g)$ small enough?



Model complexity	↑	E_{in}	↓
Model complexity	↑	$E_{\text{out}} - E_{\text{in}}$	↑



Sample complexity	↑	E_{in}	↑
Sample complexity	↑	$E_{\text{out}} - E_{\text{in}}$	↓

Risk Decomposition

1. Can we make sure that $E_{\text{out}}(g)$ is close enough to $E_{\text{in}}(g)$?
2. Can we make $E_{\text{in}}(g)$ small enough?

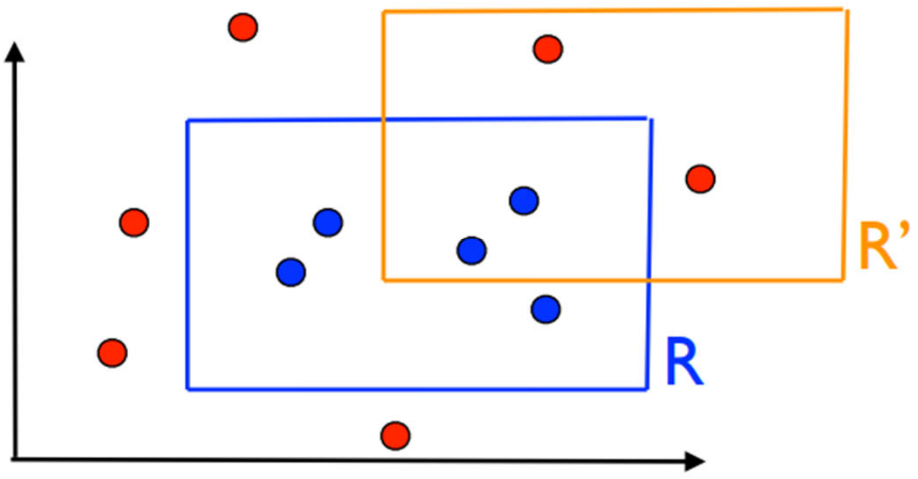
Model complexity	↑	E_{in}	↓
Model complexity	↑	$E_{\text{out}} - E_{\text{in}}$	↑

Sample complexity	↑	E_{in}	↑
Sample complexity	↑	$E_{\text{out}} - E_{\text{in}}$	↓

Comp complexity	↑	E_{in}	↑
Comp			

Bound Difference In Sample and Out of Sample

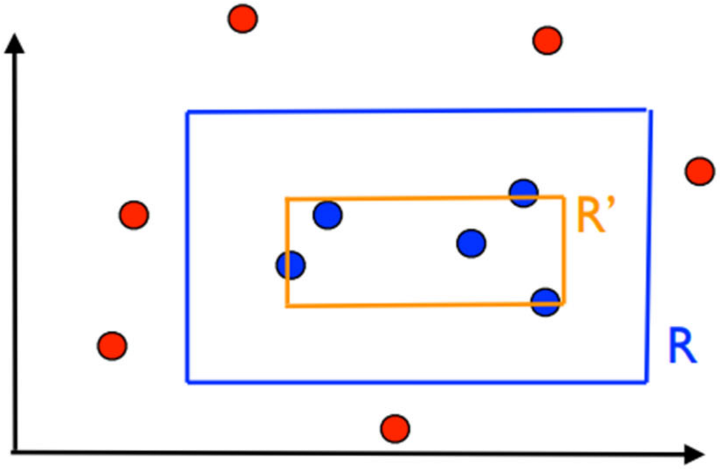
- Problem: learn unknown axis-aligned rectangle R using as small a labeled sample as possible.



- Hypothesis: rectangle R'. In general, there may be false positive and false negative points.

Bound Difference In Sample and Out of Sample

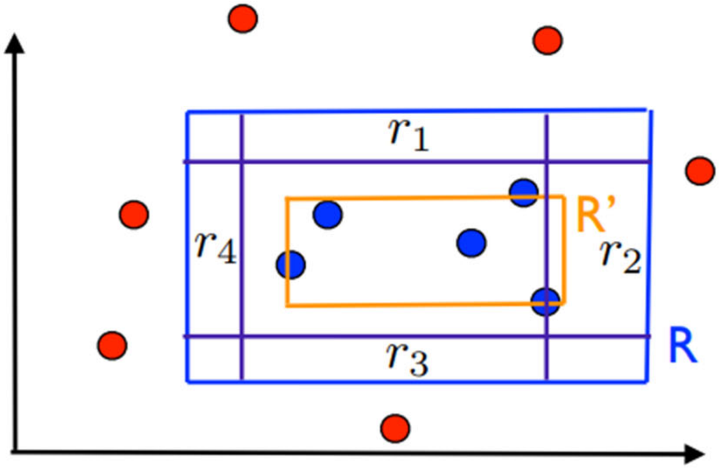
- Simple method: choose tightest consistent rectangle R’ for a large enough sample. How large a sample?



- What is the probability that $R(\mathrm{R}') > \epsilon$?

Bound Difference In Sample and Out of Sample

- Fix $\epsilon > 0$ and assume $\Pr_D[\mathsf{R}] > \epsilon$ (otherwise the result is trivial).
- Let r_1, r_2, r_3, r_4 be four smallest rectangles along the sides of R such that $\Pr_D[r_i] \geq \frac{\epsilon}{4}$.



$$\mathsf{R} = [l, r] \times [b, t]$$
$$r_4 = [l, s_4] \times [b, t]$$
$$s_4 = \inf\{s \colon \Pr\left[[l, s] \times [b, t]\right] \geq \tfrac{\epsilon}{4}\}$$
$$\Pr_D\left[[l, s_4[\times [b, t]\right] < \tfrac{\epsilon}{4}$$

Bound Difference In Sample and Out of Sample

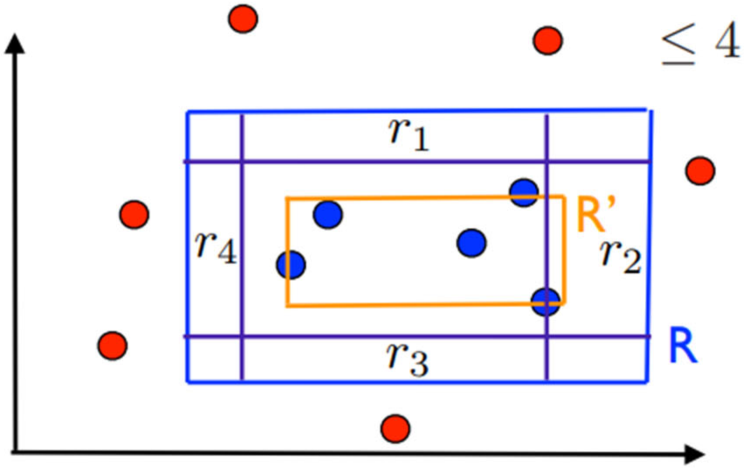
- Errors can only occur in $\mathsf{R}-\mathsf{R}'$. Thus (geometry),

  $R(\mathsf{R}') > \epsilon \Rightarrow \mathsf{R}'$ misses at least one region r_i.

- Therefore, $\Pr[R(\mathsf{R}') > \epsilon] \leq \Pr[\cup_{i=1}^{4} \{\mathsf{R}' \text{ misses } r_i\}]$

$$\leq \sum_{i=1}^{4} \Pr[\{\mathsf{R}' \text{ misses } r_i\}]$$

$$\leq 4(1-\tfrac{\epsilon}{4})^m \leq 4e^{-\frac{m\epsilon}{4}}.$$



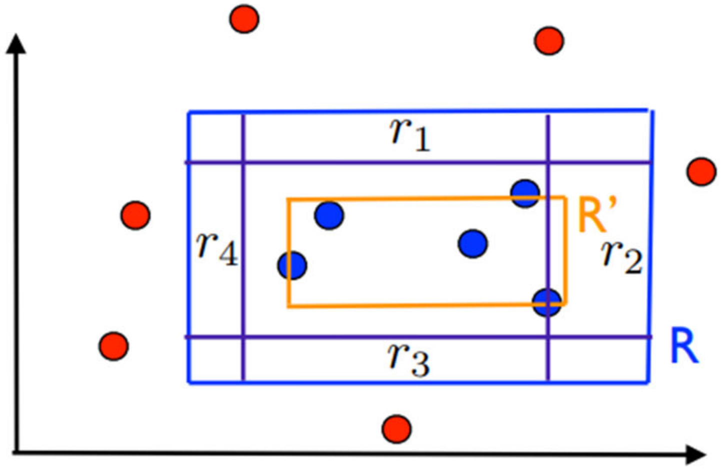
Bound Difference In Sample and Out of Sample

- Set $\delta > 0$ to match the upper bound:

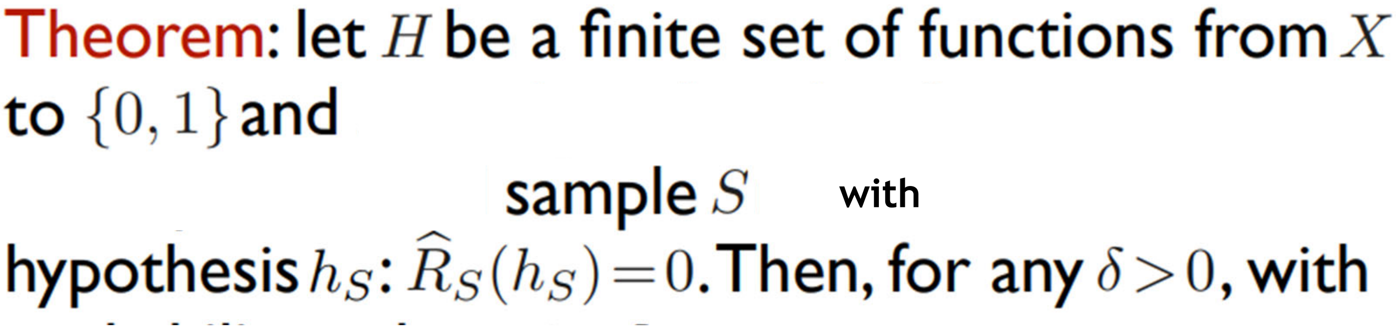
$$4e^{-\frac{m\epsilon}{4}} \leq \delta \Leftrightarrow m \geq \frac{4}{\epsilon} \log \frac{4}{\delta}.$$

- Then, for $m \geq \frac{4}{\epsilon} \log \frac{4}{\delta}$, with probability at least$1 - \delta$,

$$R(\mathrm{R}') \leq \epsilon.$$



Bound Difference In Sample and Out of Sample

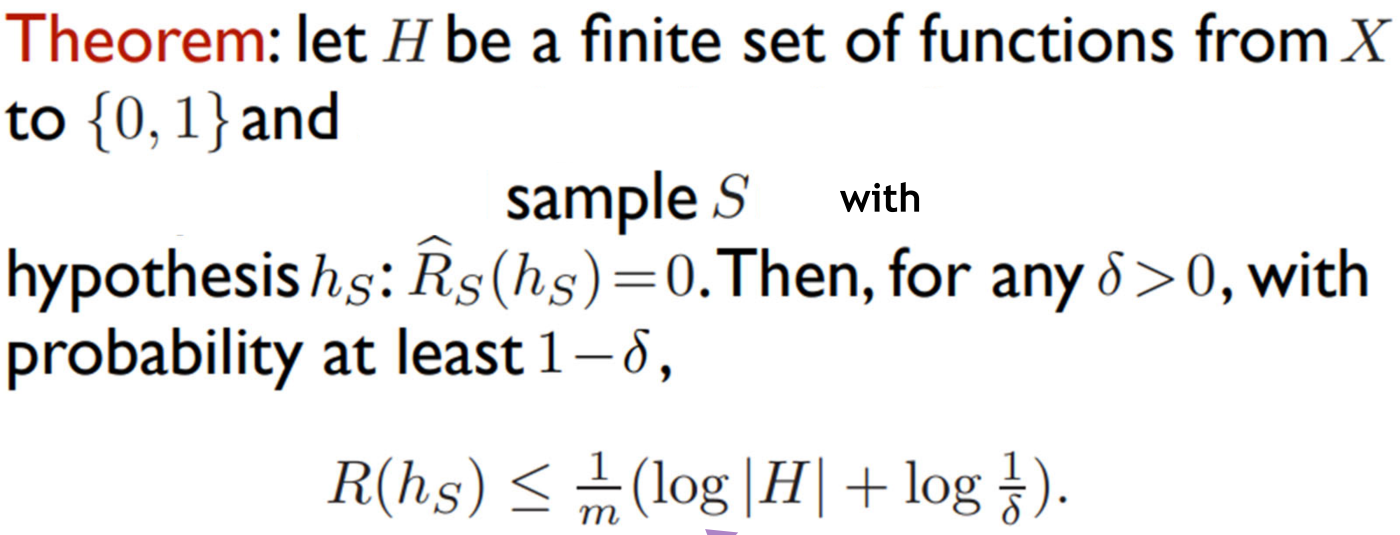


Theorem: let H be a finite set of functions from X to $\{0,1\}$ and

sample S with

hypothesis h_S: $\widehat{R}_S(h_S)=0$. Then, for any $\delta>0$, with

Bound Difference In Sample and Out of Sample



Theorem: let H be a finite set of functions from X to $\{0, 1\}$ and

sample S with

hypothesis h_S: $\widehat{R}_S(h_S) = 0$. Then, for any $\delta > 0$, with probability at least $1 - \delta$,

$$R(h_S) \leq \frac{1}{m}(\log |H| + \log \frac{1}{\delta}).$$

Number of functions in the collection of hypothesis – think the model complexity

Bound Difference In Sample and Out of Sample

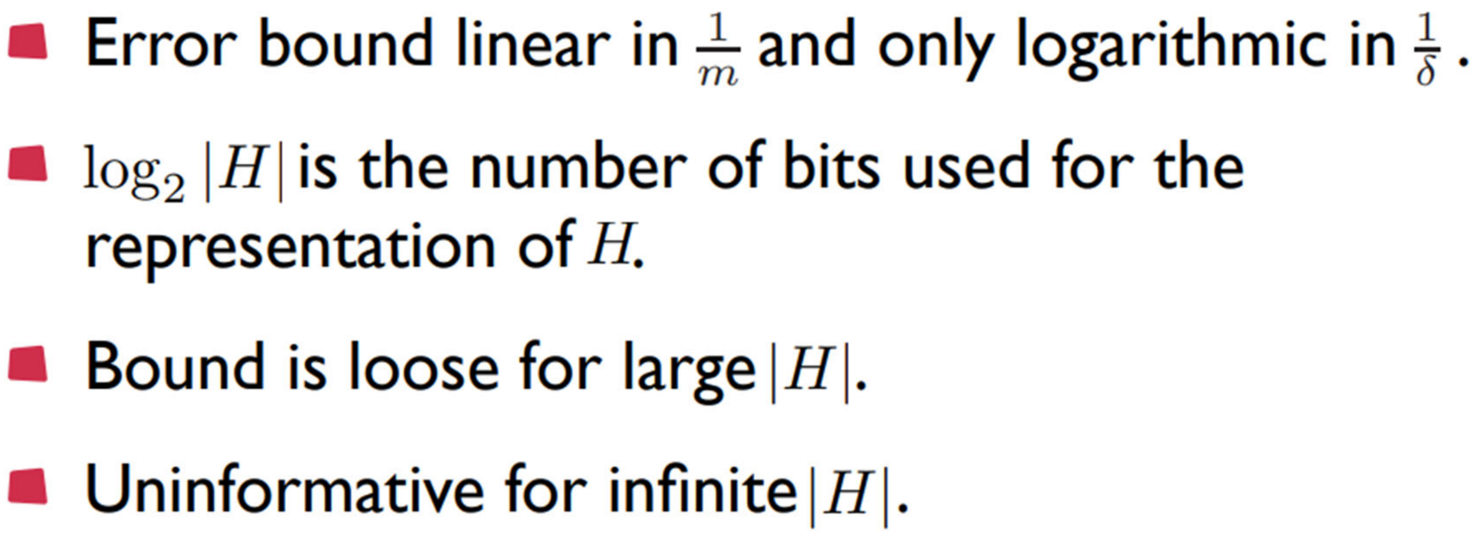
Proof: for any $\epsilon > 0$, define $H_\epsilon = \{h \in H : R(h) > \epsilon\}$.
Then,

$$
\begin{aligned}
&\Pr\left[\exists h \in H_\epsilon \colon \widehat{R}_S(h) = 0\right] \\
&\quad = \Pr\left[\widehat{R}_S(h_1) = 0 \vee \cdots \vee \widehat{R}_S(h_{|H_\epsilon|}) = 0\right] \\
&\quad \leq \sum_{h \in H_\epsilon} \Pr\left[\widehat{R}_S(h) = 0\right] && \text{(union bound)} \\
&\quad \leq \sum_{h \in H_\epsilon} (1 - \epsilon)^m \leq |H|(1 - \epsilon)^m \leq |H| e^{-m\epsilon}.
\end{aligned}
$$

Question

- What is the probability of flipping a fair coin 10 times and getting all heads?
- What is the probability of flipping 1000 fair coins 10 times and getting all heads for at least one coin?

Bound Difference In Sample and Out of Sample



- Error bound linear in $\frac{1}{m}$ and only logarithmic in $\frac{1}{\delta}$.
- $\log_2|H|$ is the number of bits used for the representation of H.
- Bound is loose for large $|H|$.
- Uninformative for infinite $|H|$.

Square Loss

Hypothesis space: $\mathcal{F} = \left\{ f : \mathbf{R}^d \to \mathbf{R} \mid f(x) = w^T x,\ w \in \mathbf{R}^d \right\}$

Given data set $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$,

$$\hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \left(w^T x_i - y_i \right)^2,$$

$$\nabla \hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

Normal Equations

$$\frac{2}{m}\sum_{i=1}^{m}(\langle \mathbf{w}, \mathbf{x}_i\rangle - y_i)\mathbf{x}_i = 0.$$

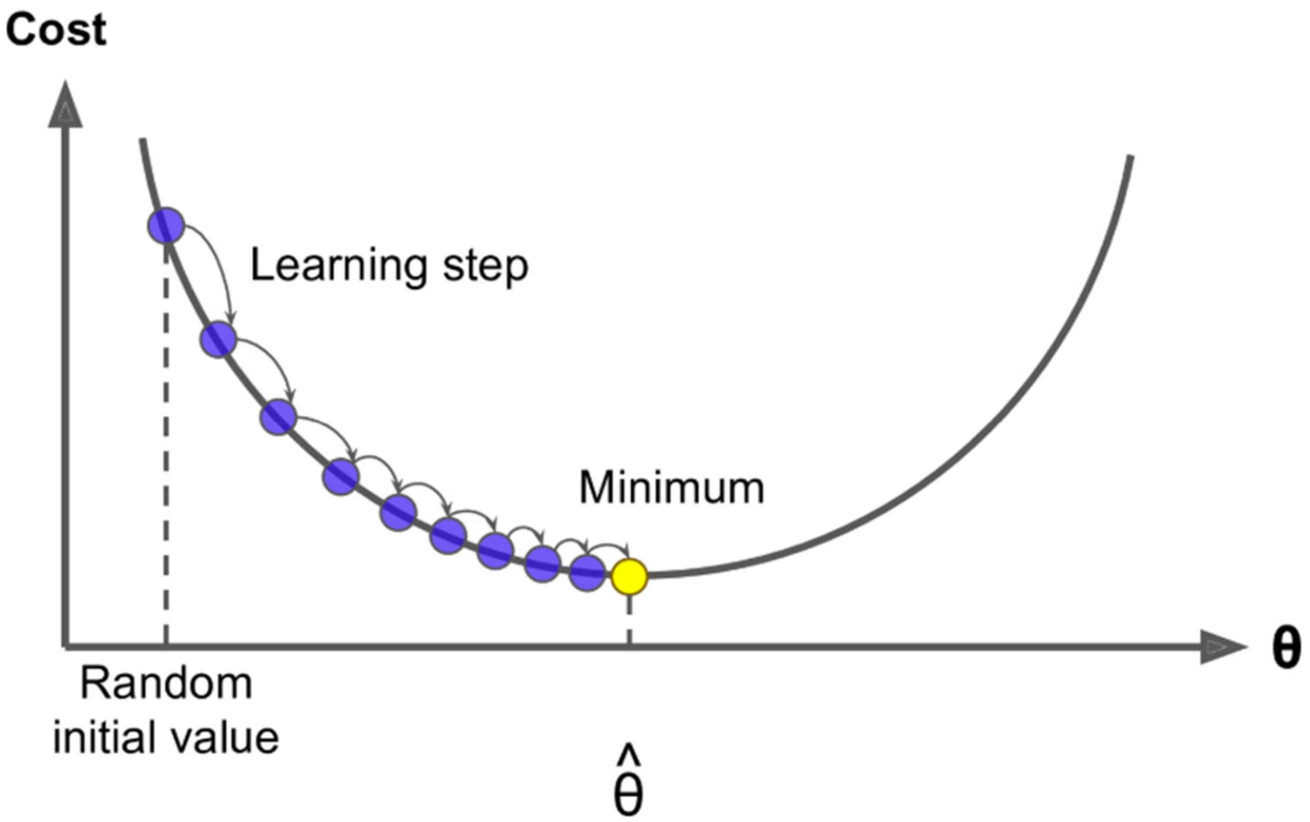
$$A = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \ldots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix} \begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \ldots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix}$$

$$\mathbf{b} = \begin{pmatrix} \vdots & & \vdots \\ \mathbf{x}_1 & \ldots & \mathbf{x}_m \\ \vdots & & \vdots \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix} .$$

$$A = \left(\sum_{i=1}^{m} \mathbf{x}_i\, \mathbf{x}_i^{\top}\right) \quad \text{and} \quad \mathbf{b} = \sum_{i=1}^{m} y_i \mathbf{x}_i .$$

$$\mathbf{w} = A^{-1}\,\mathbf{b}.$$

Gradient Descent



Gradient Descent

- Use derivatives to approximate a function by a linear function

$$f(\mathbf{w}) \approx f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle$$

Gradient Descent

- Use derivatives to approximate a function by a linear function

$$f(\mathbf{w}) \approx f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle$$

- Approximation inaccurate for far away points

$$\mathbf{w}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}^{(t)}\|^2 + \eta \left(f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle \right) .$$

Gradient Descent

- Use derivatives to approximate a function by a linear function

$$f(\mathbf{w}) \approx f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle$$

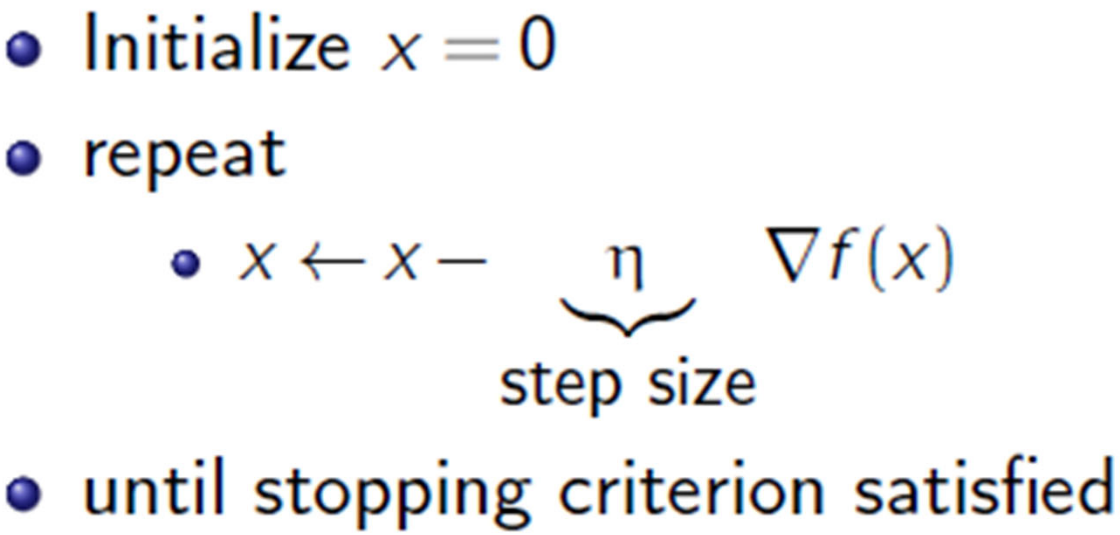
- Approximation inaccurate for far away points

$$\mathbf{w}^{(t+1)} = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}^{(t)}\|^2 + \eta \left(f(\mathbf{w}^{(t)}) + \langle \mathbf{w} - \mathbf{w}^{(t)}, \nabla f(\mathbf{w}^{(t)}) \rangle \right).$$

- Learning Rate controls the trade-off by determining the step size

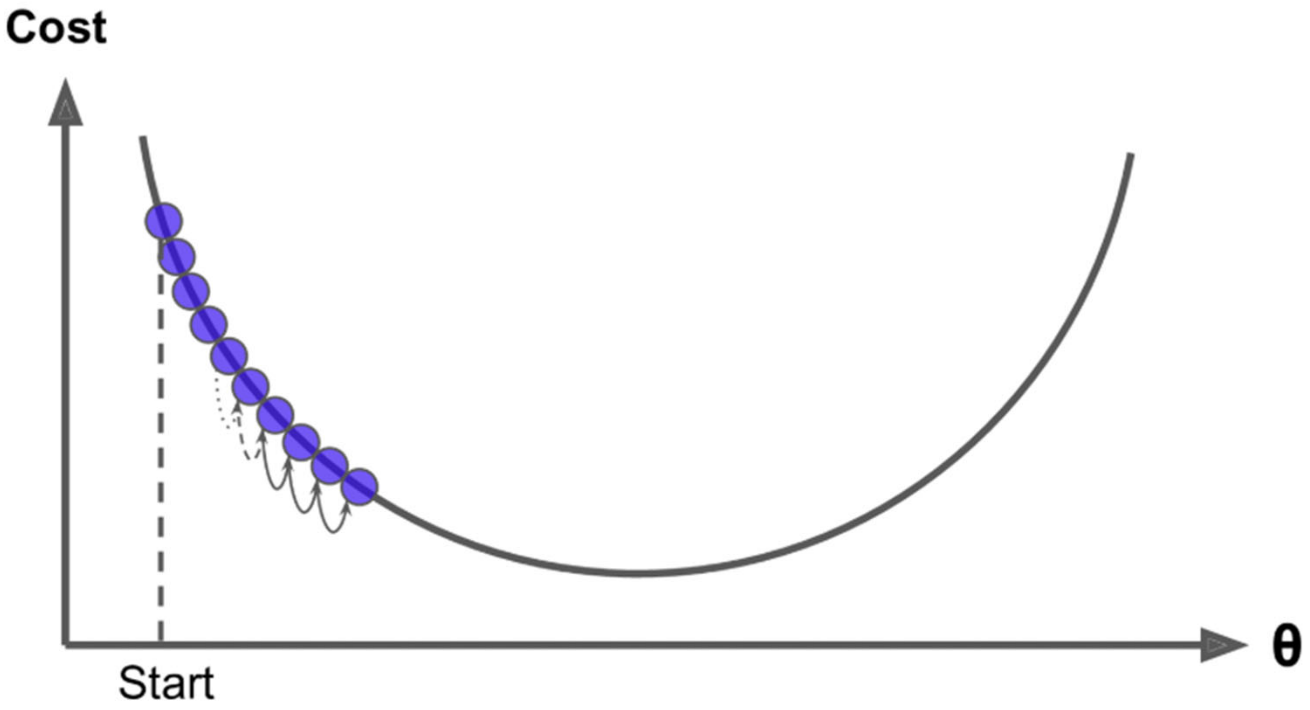
$$\mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla f(\mathbf{w}^{(t)}),$$

Gradient Descent

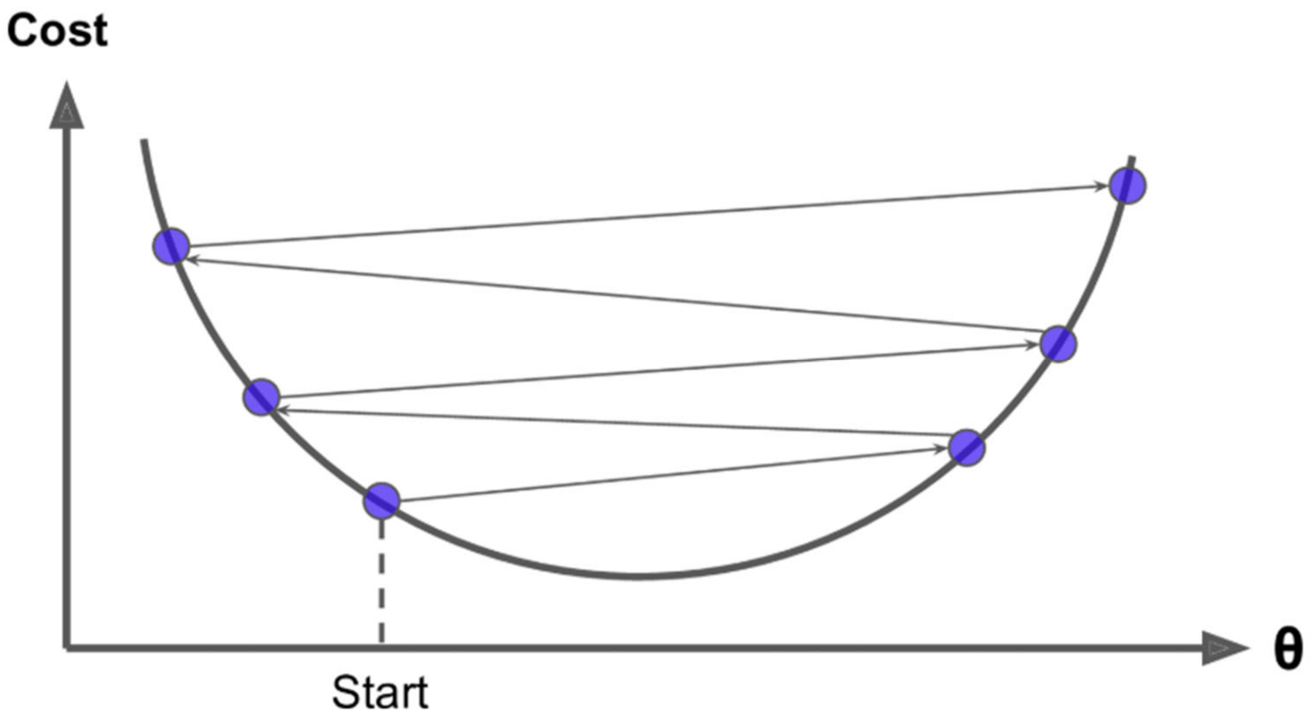


- Initialize $x = 0$
- repeat
  - $x \leftarrow x - \underbrace{\eta}_{\text{step size}} \nabla f(x)$
- until stopping criterion satisfied

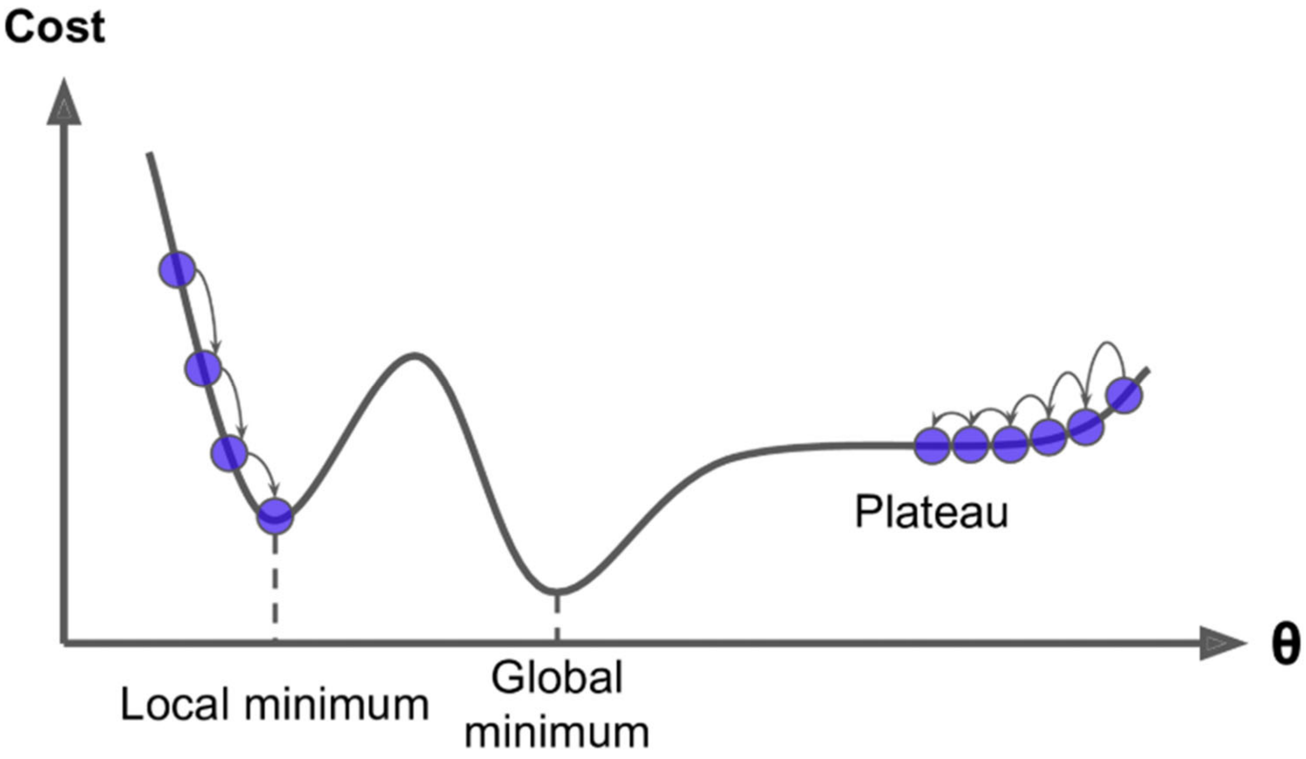
Learning Rate?



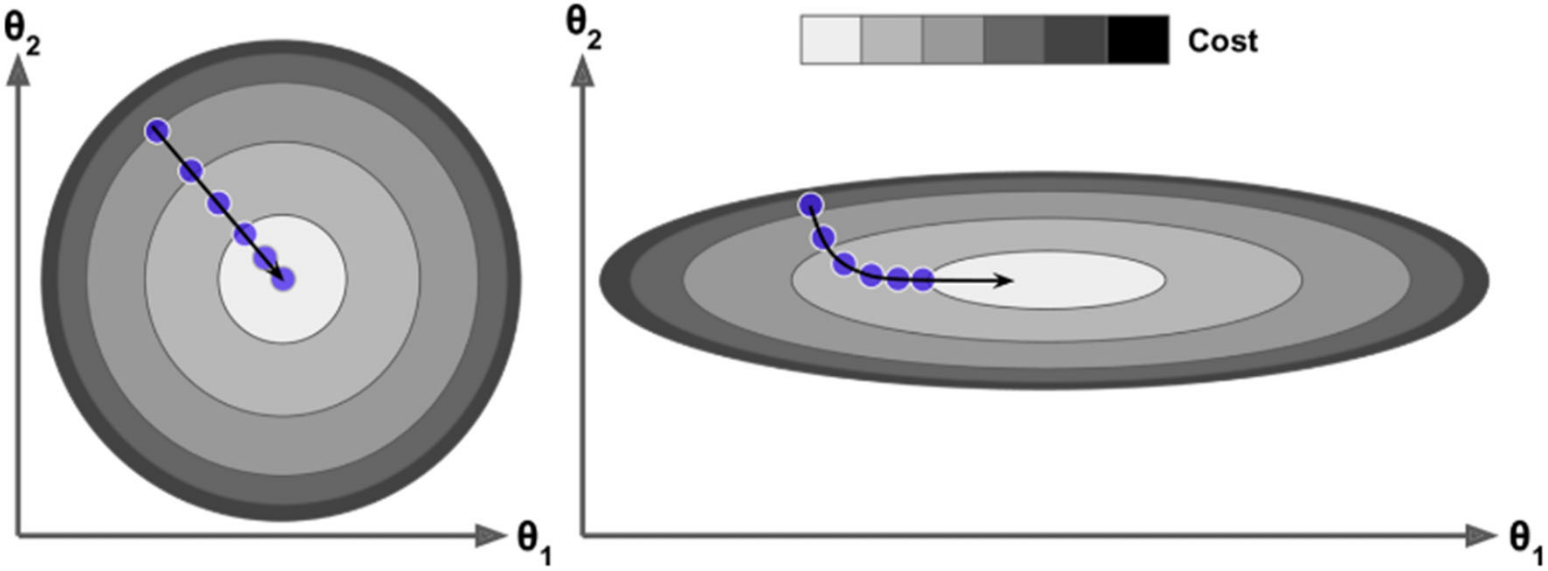
Learning Rate?



Issues with Gradient Descent



Issues with Gradient Descent



Changing the Learning Rate

- Fixed Learning Rate
- Changing Learning Rate
  - Determine before Iteration
  - Determine at each Iteration

Backtracking Line Search

- First fix a parameter $0 < \beta < 1$
- Then at each iteration, start with $t = 1$, and while

$$f(x - t\nabla f(x)) > f(x) - \frac{t}{2}\|\nabla f(x)\|^2,$$

update $t = \beta t$

Stochastic Gradient Descent

The **full gradient** is

$$\nabla \hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

It's an average over the **full batch** of data $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.

Stochastic Gradient Descent

The **full gradient** is

$$\nabla \hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

It's an average over the **full batch** of data $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.

Let's take a random subsample of size N (called a **minibatch**):

$$(x_{m_1}, y_{m_1}), \ldots, (x_{m_N}, y_{m_N})$$

Stochastic Gradient Descent

The **full gradient** is

$$\nabla \hat{R}_n(w) = \frac{1}{n}\sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i)$$

It's an average over the **full batch** of data $\mathcal{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$.

Let's take a random subsample of size N (called a **minibatch**):

$$(x_{m_1}, y_{m_1}), \ldots, (x_{m_N}, y_{m_N})$$

The **minibatch gradient** is

$$\nabla \hat{R}_N(w) = \frac{1}{N}\sum_{i=1}^{N} \nabla_w \ell(f_w(x_{m_i}), y_{m_i})$$

What’s the expected value of minibatch gradient

$$\mathbb{E}\left[\nabla \hat{R}_N(w)\right] = \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left[\nabla_w \ell(f_w(x_{m_i}), y_{m_i})\right]$$

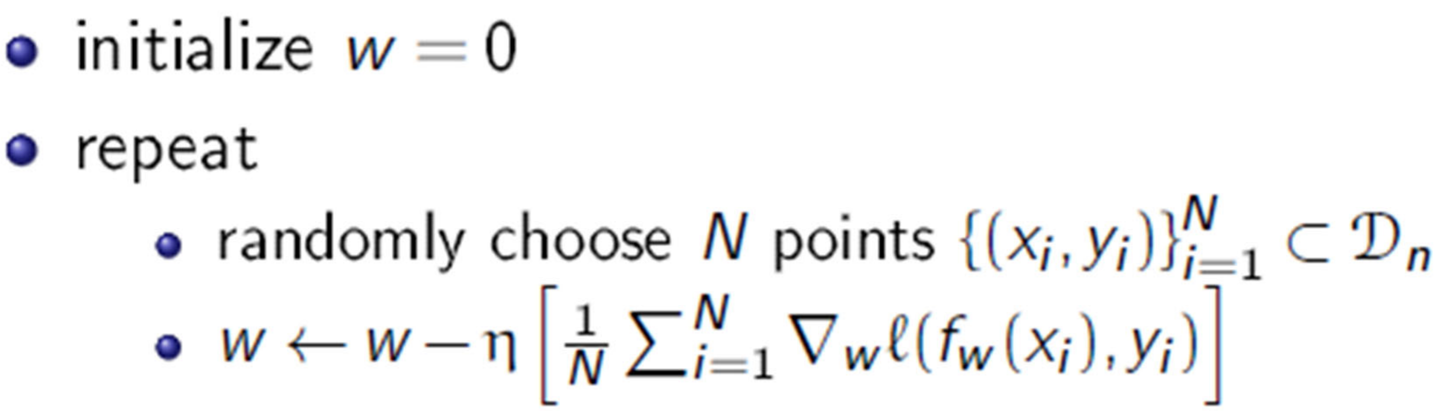
What’s the expected value of minibatch gradient

$$
\begin{aligned}
\mathbb{E}\left[\nabla \hat{R}_N(w)\right] &= \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}[\nabla_w \ell(f_w(x_{m_i}), y_{m_i})] \\
&= \mathbb{E}[\nabla_w \ell(f_w(x_{m_1}), y_{m_1})] \\
&= \sum_{i=1}^{n}\mathbb{P}(m_1 = i)\nabla_w \ell(f_w(x_i), y_i)
\end{aligned}
$$

What’s the expected value of minibatch gradient

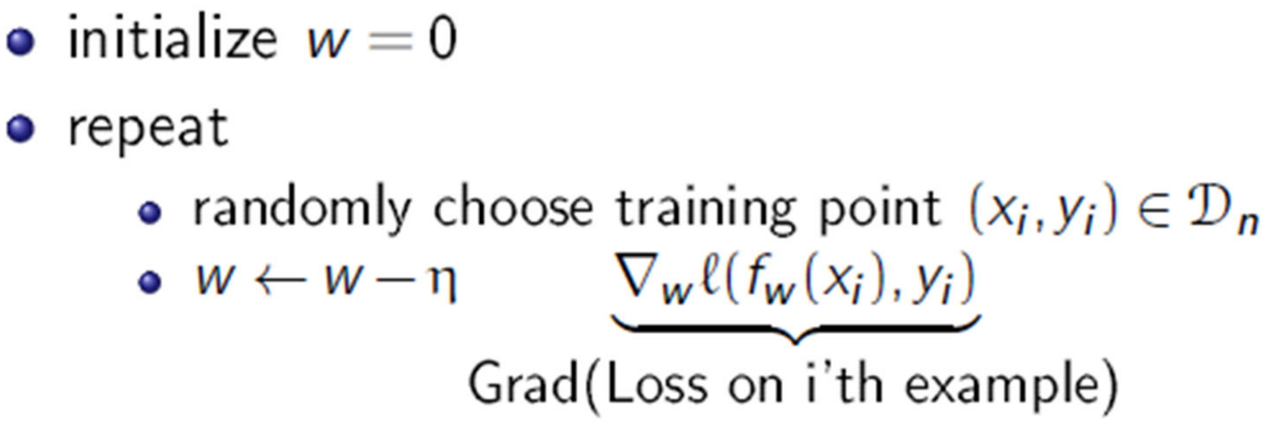
$$\begin{aligned}
\mathbb{E}\left[\nabla \hat{R}_N(w)\right] &= \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}[\nabla_w \ell(f_w(x_{m_i}), y_{m_i})] \\
&= \mathbb{E}[\nabla_w \ell(f_w(x_{m_1}), y_{m_1})] \\
&= \sum_{i=1}^{n} \mathbb{P}(m_1 = i)\nabla_w \ell(f_w(x_i), y_i) \\
&= \frac{1}{n}\sum_{i=1}^{n} \nabla_w \ell(f_w(x_i), y_i) \\
&= \nabla \hat{R}_n(w)
\end{aligned}$$

Minibatch gradient descent



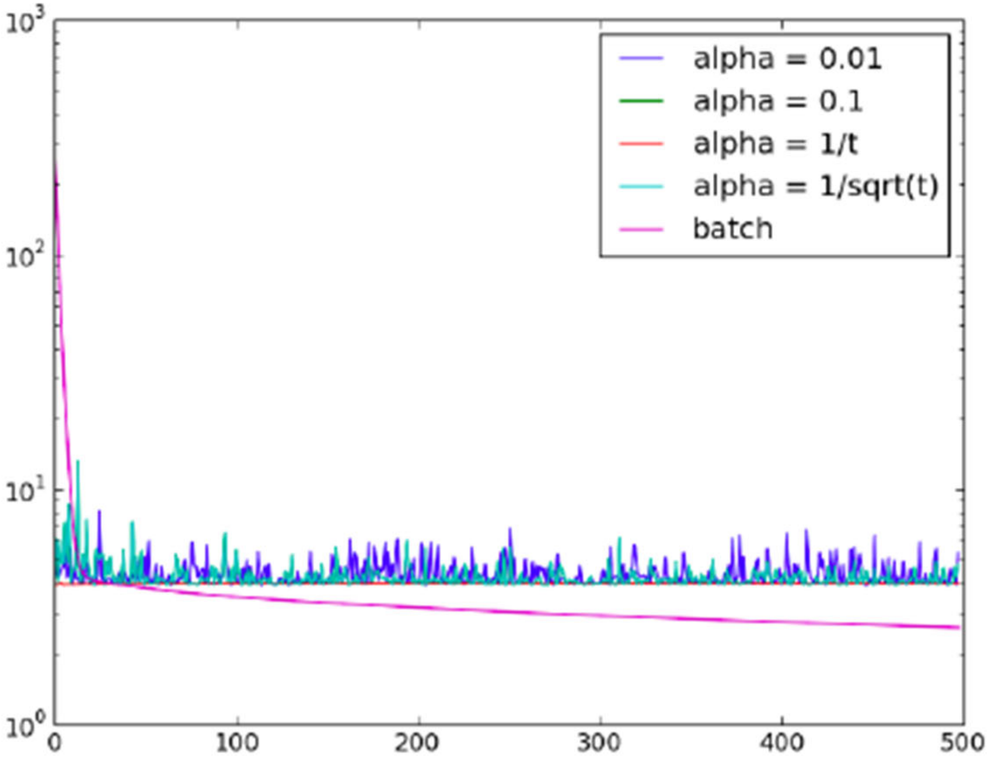
- initialize $w = 0$
- repeat
  - randomly choose N points $\{(x_i, y_i)\}_{i=1}^{N} \subset \mathcal{D}_n$
  - $w \leftarrow w - \eta \left[\frac{1}{N} \sum_{i=1}^{N} \nabla_w \ell(f_w(x_i), y_i) \right]$

Stochastic gradient descent



- initialize $w = 0$
- repeat
  - randomly choose training point $(x_i, y_i) \in \mathcal{D}_n$
  - $w \leftarrow w - \eta \underbrace{\nabla_w \ell(f_w(x_i), y_i)}_{\text{Grad(Loss on i'th example)}}$

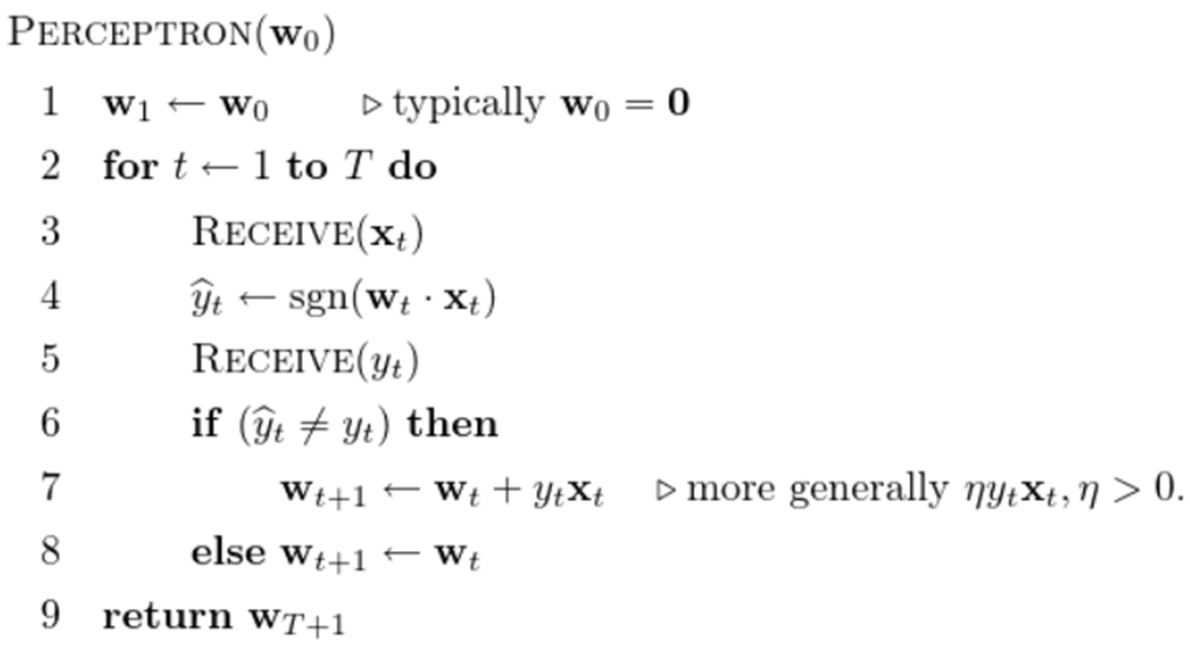
Gradient descent vs Stochastic Gradient Descent



Question

- Suppose you have been successfully running mini-batch gradient descent with a full training set size of 105 and a mini-batch size of 100.
- After receiving more data your full training set size increases to 109.
- Give a hand-wavy argument as to why the mini-batch size need not increase even though we have 10000 times more data.

Perceptron Algorithm as SGD



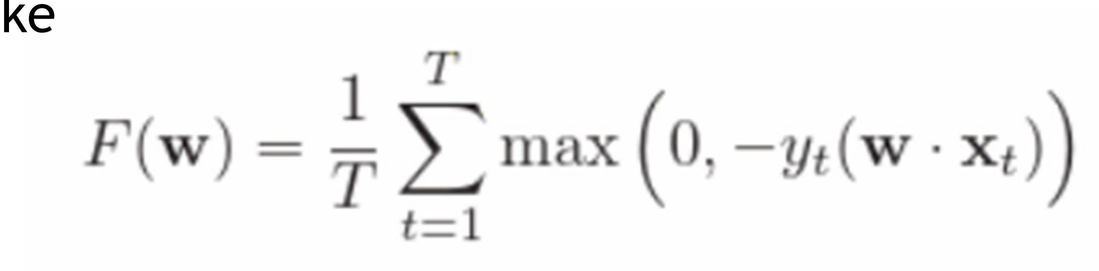
```
PERCEPTRON($\mathbf{w}_0$)
$\mathbf{w}_1 \leftarrow \mathbf{w}_0$    $\triangleright$ typically $\mathbf{w}_0 = \mathbf{0}$
for $t \leftarrow 1$ to $T$ do
    RECEIVE($\mathbf{x}_t$)
    $\widehat{y}_t \leftarrow \mathrm{sgn}(\mathbf{w}_t \cdot \mathbf{x}_t)$
    RECEIVE($y_t$)
    if $(\widehat{y}_t \neq y_t)$ then
        $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + y_t \mathbf{x}_t$    $\triangleright$ more generally $\eta y_t \mathbf{x}_t, \eta > 0$.
    else $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t$
return $\mathbf{w}_{T+1}$
```

Perceptron Algorithm as SGD

y Sign $\langle \mathbf{w}, \mathbf{x} \rangle$	$+1$	-1
$+1$	$+1$	-1
-1	-1	$+1$

Perceptron Algorithm as SGD

- Take



$$F(\mathbf{w}) = \frac{1}{T}\sum_{t=1}^{T} \max\Big(0, -y_t(\mathbf{w}\cdot\mathbf{x}_t)\Big)$$

Perceptron Algorithm as SGD

- Take

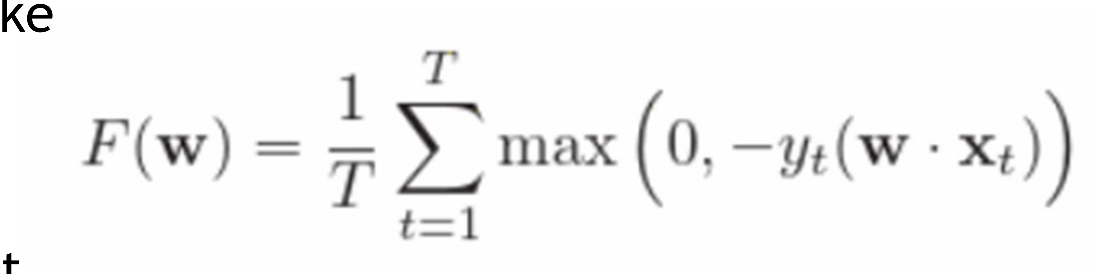
$$F(\mathbf{w}) = \frac{1}{T}\sum_{t=1}^{T} \max\Big(0, -y_t(\mathbf{w}\cdot\mathbf{x}_t)\Big)$$

- Set

$$\widetilde{F}(\mathbf{w},\mathbf{x}) \;=\; \max\,(0, -f(\mathbf{x})(\mathbf{w}\cdot\mathbf{x}))$$

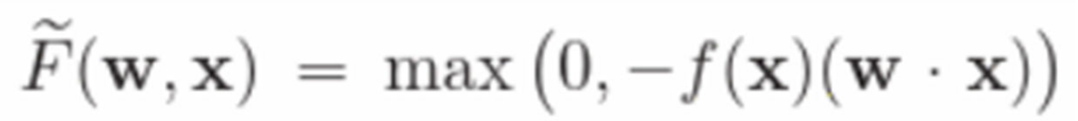
Perceptron Algorithm as SGD

- Take



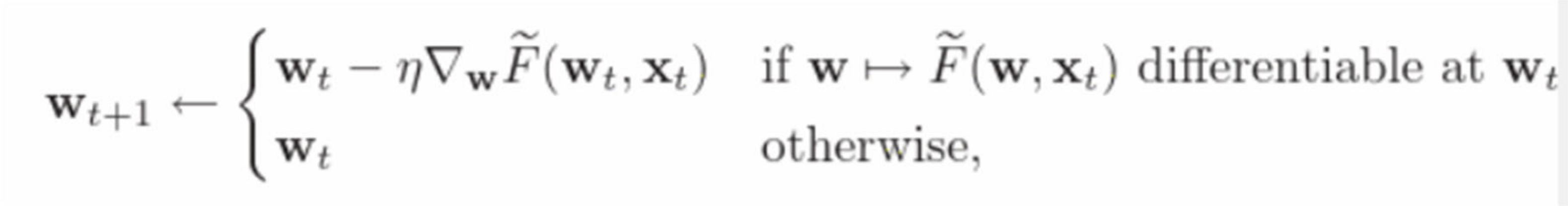
$$F(\mathbf{w}) = \frac{1}{T}\sum_{t=1}^{T} \max\Big(0, -y_t(\mathbf{w}\cdot\mathbf{x}_t)\Big)$$

- Set



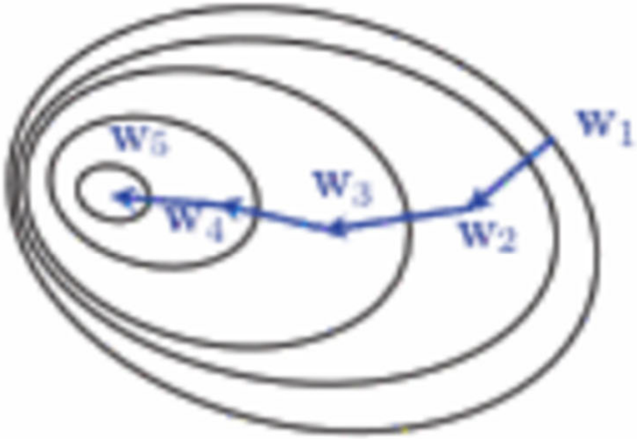
$$\widetilde{F}(\mathbf{w},\mathbf{x}) \;=\; \max\big(0, -f(\mathbf{x})(\mathbf{w}\cdot\mathbf{x})\big)$$

- Update Guess



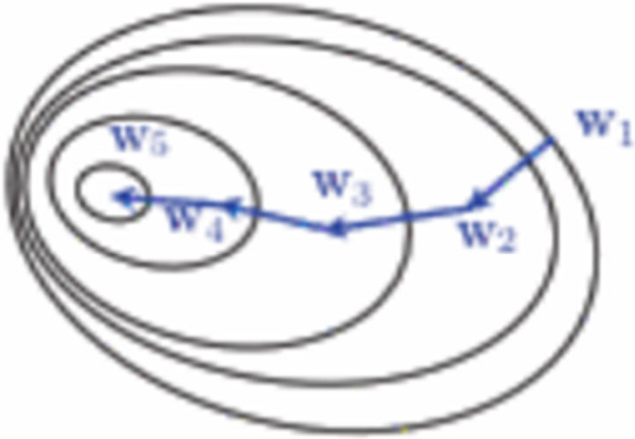
$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t - \eta\nabla_{\mathbf{w}}\widetilde{F}(\mathbf{w}_t,\mathbf{x}_t) & \text{if } \mathbf{w}\mapsto\widetilde{F}(\mathbf{w},\mathbf{x}_t) \text{ differentiable at } \mathbf{w}_t \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

Perceptron Algorithm as SGD



$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

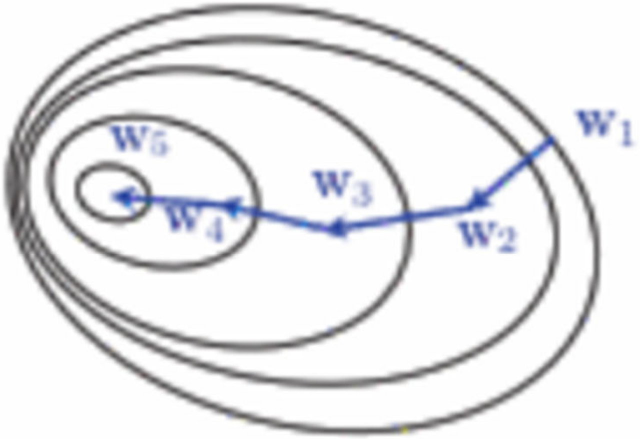
Perceptron Algorithm as SGD



$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

$$\nabla_{\mathbf{w}} \widetilde{F}(\mathbf{w}, \mathbf{x}_t) = -y\mathbf{x}_t \quad \text{if} \quad y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0$$

Perceptron Algorithm as SGD



$$\mathbf{w}_{t+1} \leftarrow \begin{cases} \mathbf{w}_t + \eta y_t \mathbf{x}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0; \\ \mathbf{w}_t & \text{if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0; \\ \mathbf{w}_t & \text{otherwise,} \end{cases}$$

$$\nabla_{\mathbf{w}} \widetilde{F}(\mathbf{w}, \mathbf{x}_t) \;=\; -y\mathbf{x}_t \quad \text{if} \quad y_t(\mathbf{w} \cdot \mathbf{x}_t) < 0$$

$$\nabla_{\mathbf{w}} \widetilde{F}(\mathbf{w}, \mathbf{x}_t) = 0 \text{ if } y_t(\mathbf{w} \cdot \mathbf{x}_t) > 0.$$

Take-Aways

- Write the empirical risk for a particular loss function over a particular hypothesis space, such as
  - square loss over a hypothesis space of linear functions
  - 0-1 loss over a hypothesis space of perceptron classifiers?
- What are the normal equations? Can we solve them...then why use an iterative method?
- Compare and contrast gradient descent and stochastic gradient descent.