

Deep Learning for Vision & Language

SGD and Regularization, Softmax, MLPs



RICE UNIVERSITY

Stochastic Gradient Descent

- How to choose the right batch size B ?
- How to choose the right learning rate λ ?
- How to choose the right loss function, e.g. is least squares good enough?
- How to choose the right function/classifier, e.g. linear, quadratic, neural network with 1 layer, 2 layers, etc?

Linear Regression

Example: Hollywood movie data

input variables x

output variables y

training
data

	production costs	promotional costs	genre of the movie	box office first week	total book sales	total revenue USA	total revenue international
	$x_1^{(1)}$	$x_2^{(1)}$	$x_3^{(1)}$	$x_4^{(1)}$	$x_5^{(1)}$	$y_1^{(1)}$	$y_2^{(1)}$
	$x_1^{(2)}$	$x_2^{(2)}$	$x_3^{(2)}$	$x_4^{(2)}$	$x_5^{(2)}$	$y_1^{(2)}$	$y_2^{(2)}$
	$x_1^{(3)}$	$x_2^{(3)}$	$x_3^{(3)}$	$x_4^{(3)}$	$x_5^{(3)}$	$y_1^{(3)}$	$y_2^{(3)}$
test data	$x_1^{(4)}$	$x_2^{(4)}$	$x_3^{(4)}$	$x_4^{(4)}$	$x_5^{(4)}$	$y_1^{(4)}$	$y_2^{(4)}$
	$x_1^{(5)}$	$x_2^{(5)}$	$x_3^{(5)}$	$x_4^{(5)}$	$x_5^{(5)}$	$y_1^{(5)}$	$y_2^{(5)}$

Training, Validation (Dev), Test Sets



The diagram consists of three vertical rectangular boxes of different widths, all filled with a solid orange color. The leftmost box is the widest, the middle box is narrower, and the rightmost box is the narrowest. Each box contains white text centered horizontally and vertically. The text in the boxes reads 'Training Set', 'Validation Set', and 'Testing Set' respectively from left to right.

Training Set

Validation
Set

Testing Set

Training, Validation (Dev), Test Sets



Training Set

Validation
Set

Testing Set

Used during development

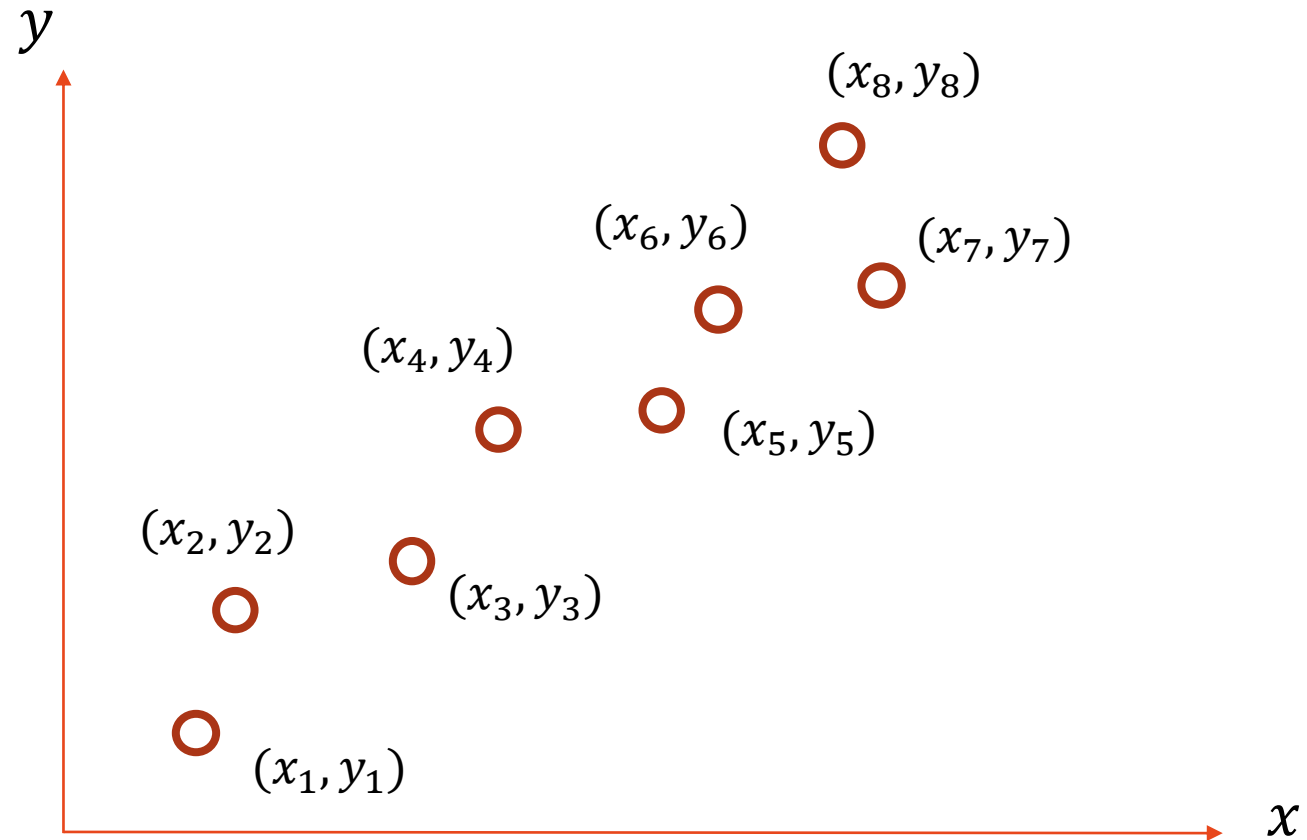
Training, Validation (Dev), Test Sets



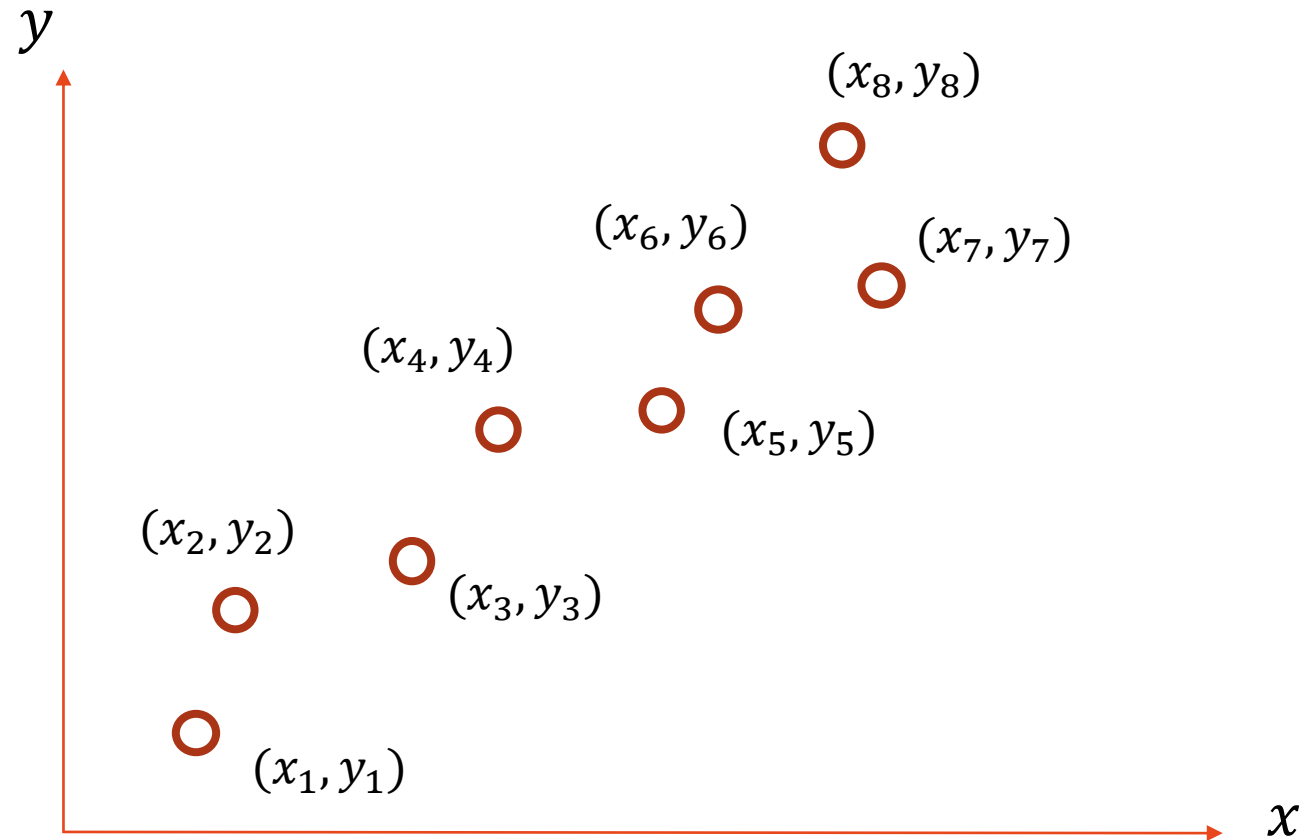
Only to be used for evaluating the model at the very end of development and any changes to the model after running it on the test set, could be influenced by what you saw happened on the test set, which would invalidate any future evaluation.

HOW TO PICK THE RIGHT MODEL?

Linear Regression – 1 output, 1 input

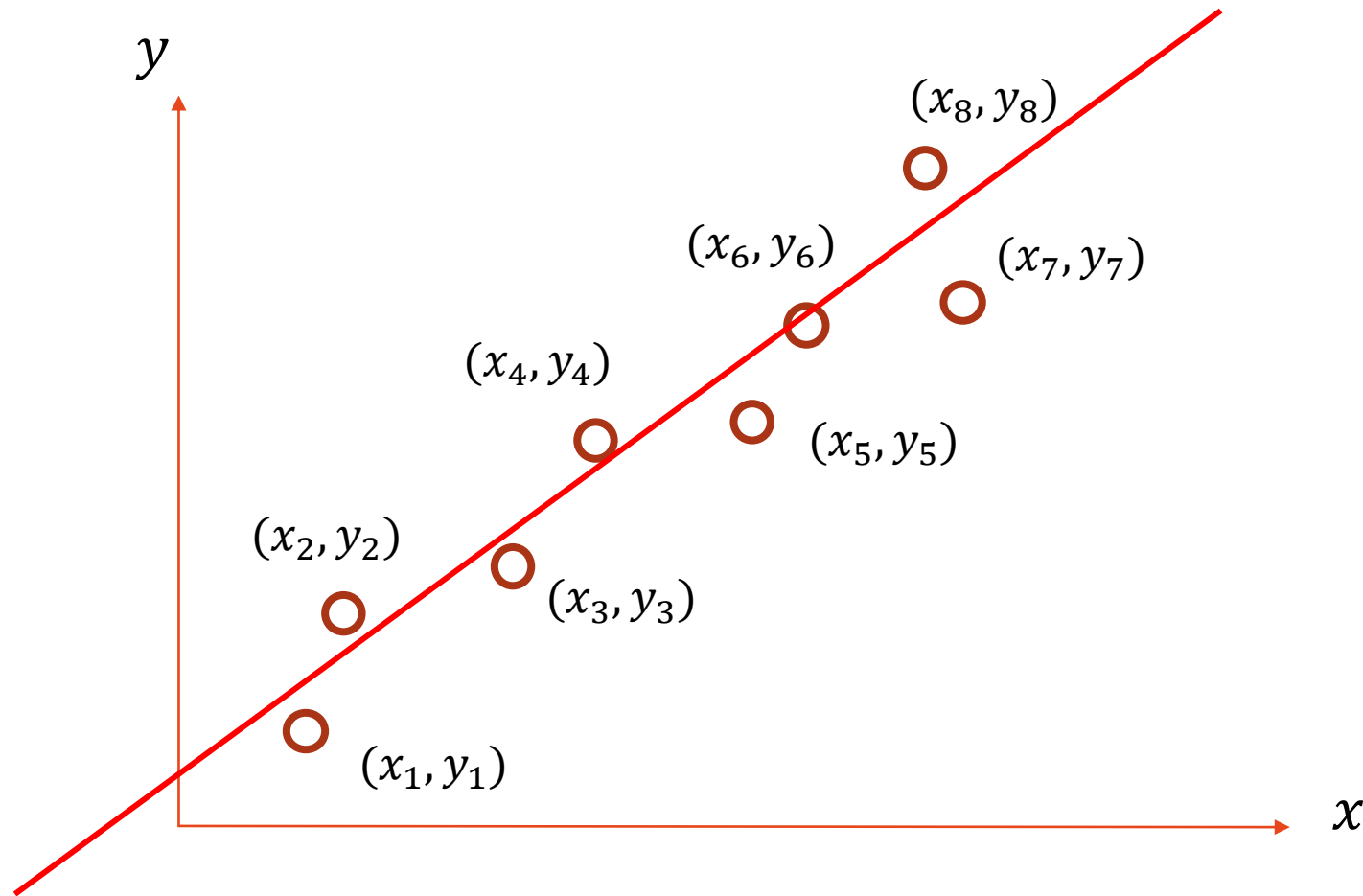


Linear Regression – 1 output, 1 input



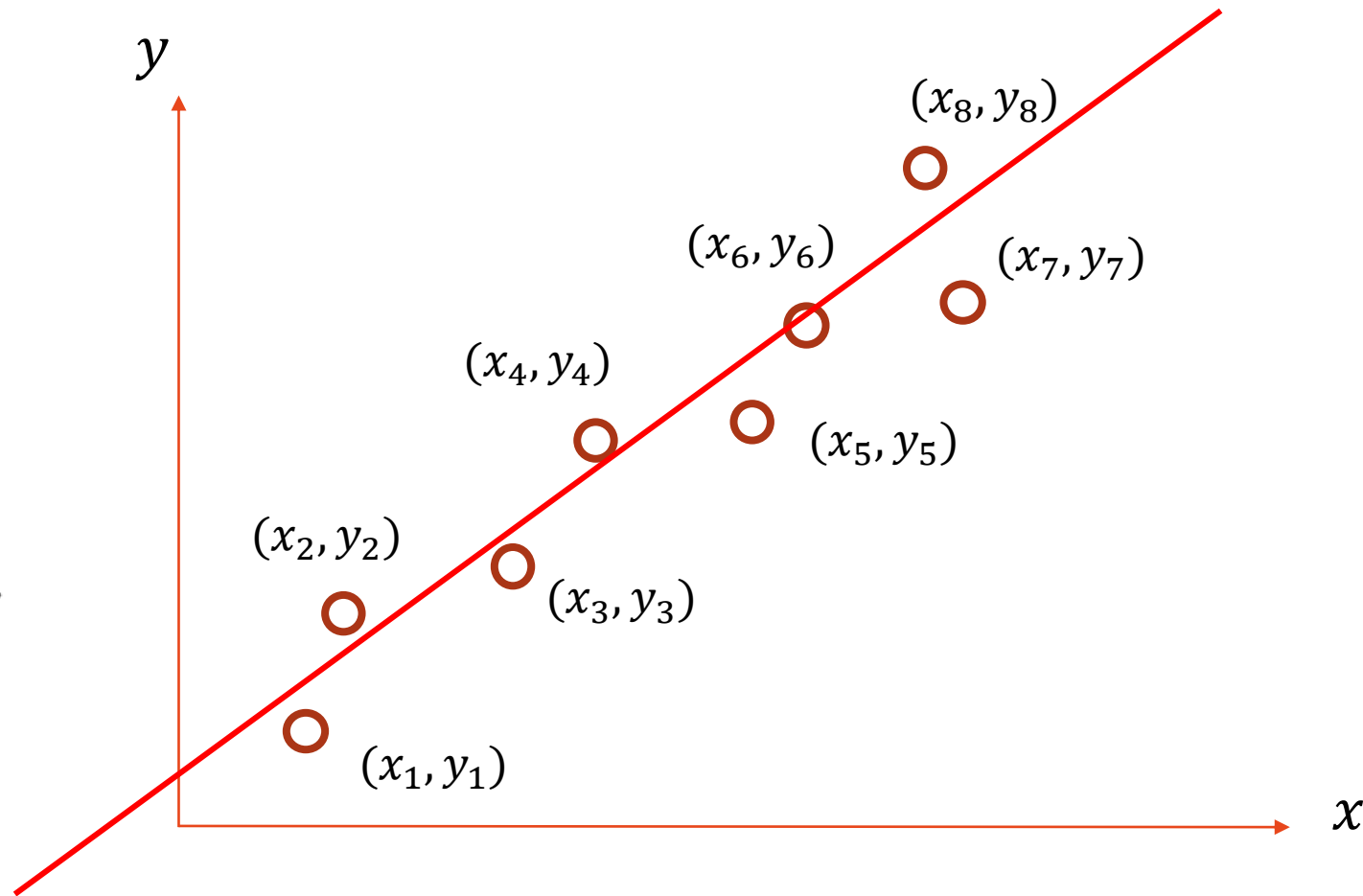
Model: $\hat{y} = wx + b$

Linear Regression – 1 output, 1 input



Model: $\hat{y} = wx + b$

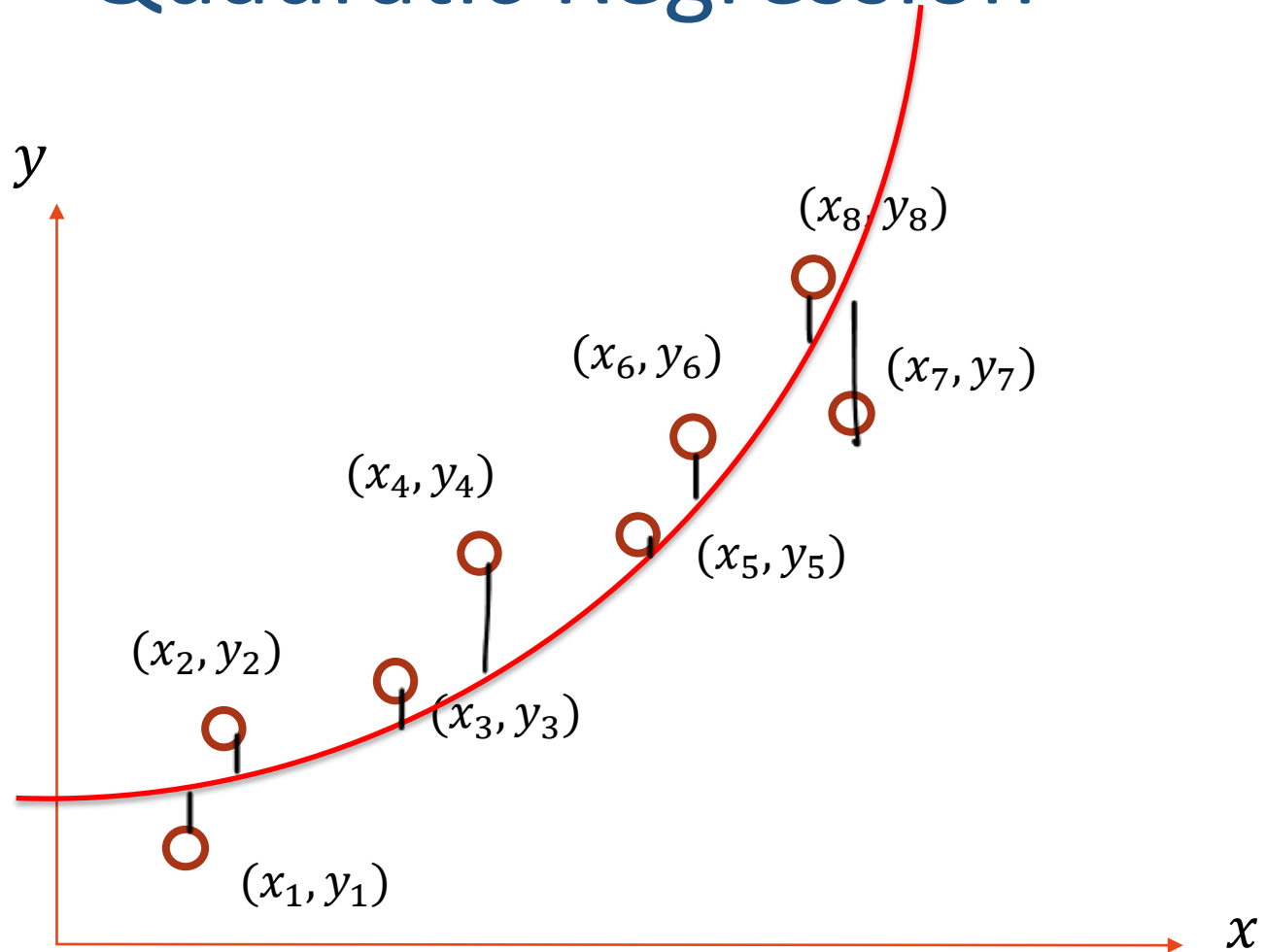
Linear Regression – 1 output, 1 input



Model: $\hat{y} = wx + b$

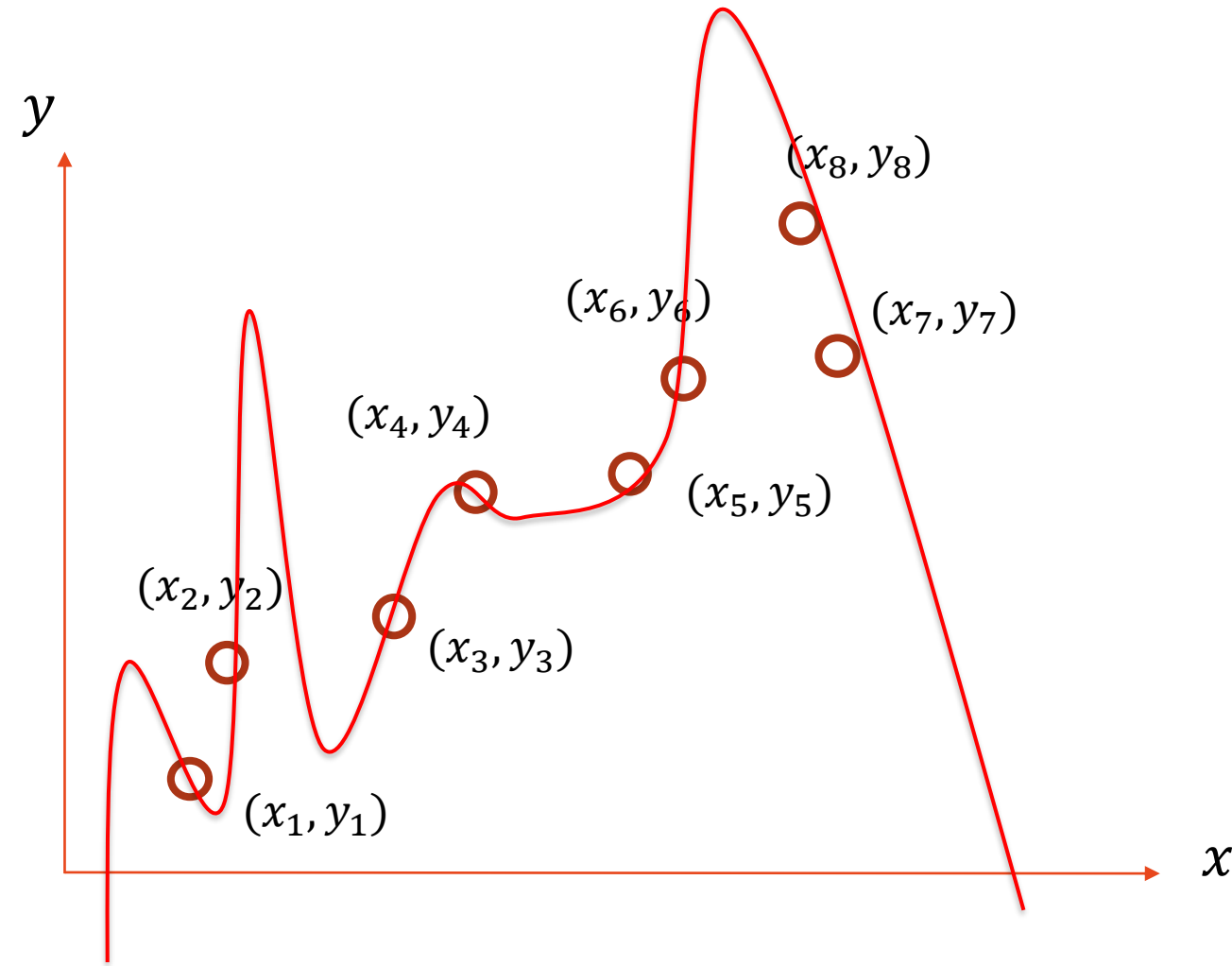
Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

Quadratic Regression



Model: $\hat{y} = w_1x^2 + w_2x + b$ Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

n-polynomial Regression

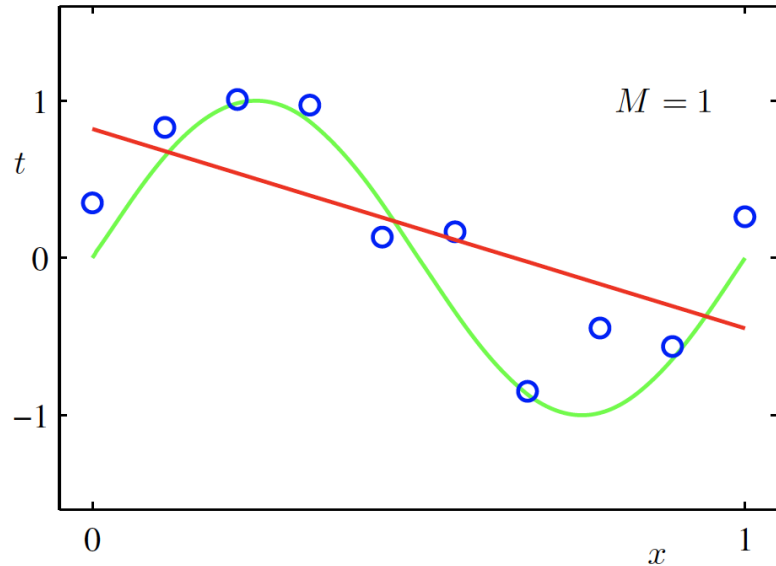


Model: $\hat{y} = w_n x^n + \dots + w_1 x + b$

Loss: $L(w, b) = \sum_{i=1}^{i=8} (\hat{y}_i - y_i)^2$

Overfitting

f is linear

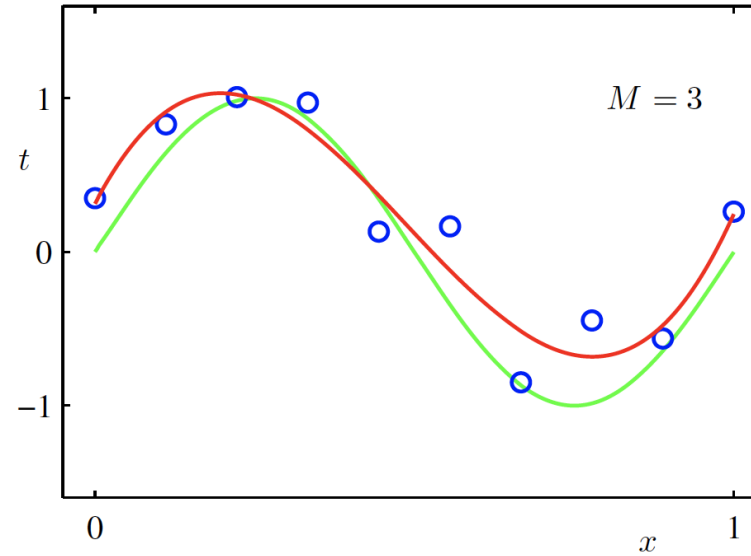


$Loss(w)$ is high

Underfitting

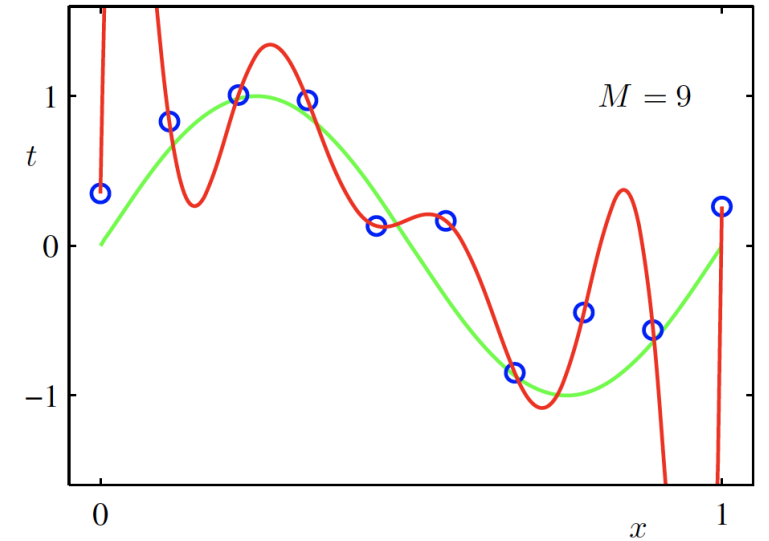
High Bias

f is cubic



$Loss(w)$ is low

f is a polynomial of degree 9



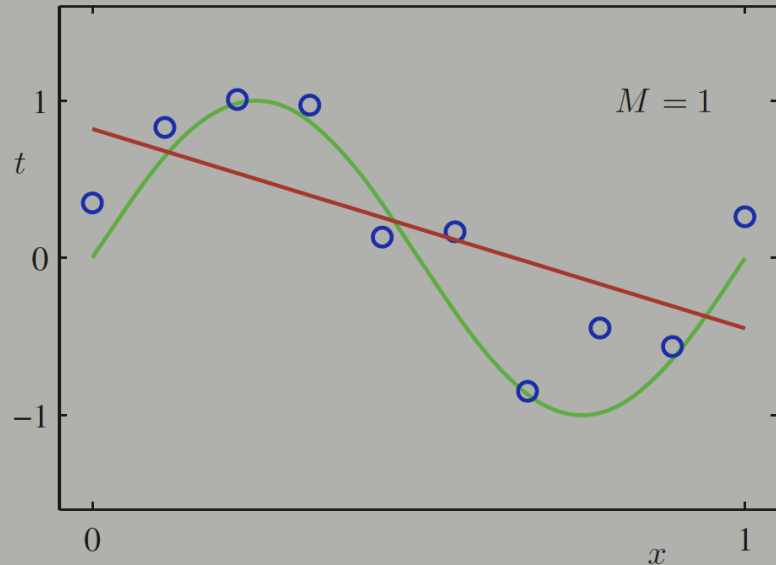
$Loss(w)$ is zero!

Overfitting

High Variance

Overfitting

f is linear

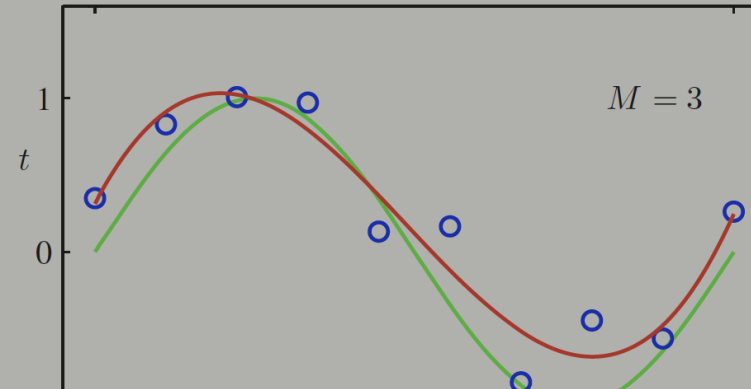


$Loss(w)$ is high

Underfitting

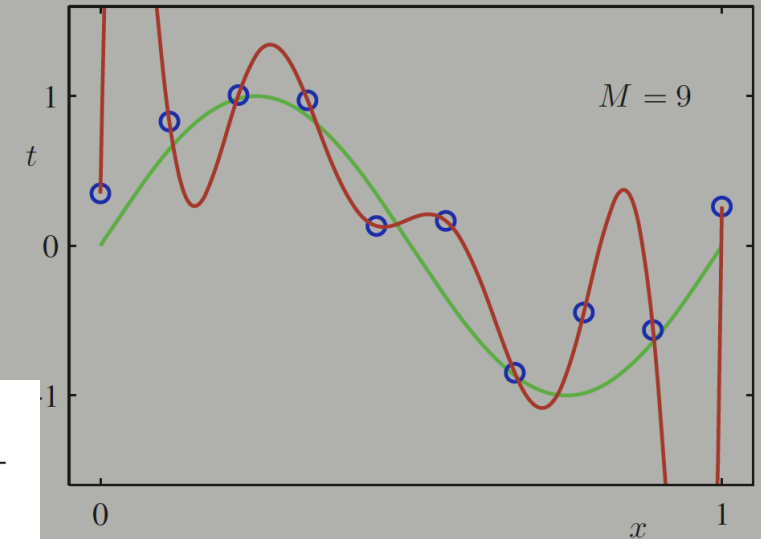
High Bias

f is cubic



	$M = 1$	$M = 6$	$M = 9$
w_0^*	0.82	0.31	0.35
w_1^*	-1.27	7.99	232.37
w_2^*		-25.43	-5321.83
w_3^*		17.37	48568.31
w_4^*			-231639.30
w_5^*			640042.26
w_6^*			-1061800.52
w_7^*			1042400.18
w_8^*			-557682.99
w_9^*			125201.43

f is a polynomial of degree 9

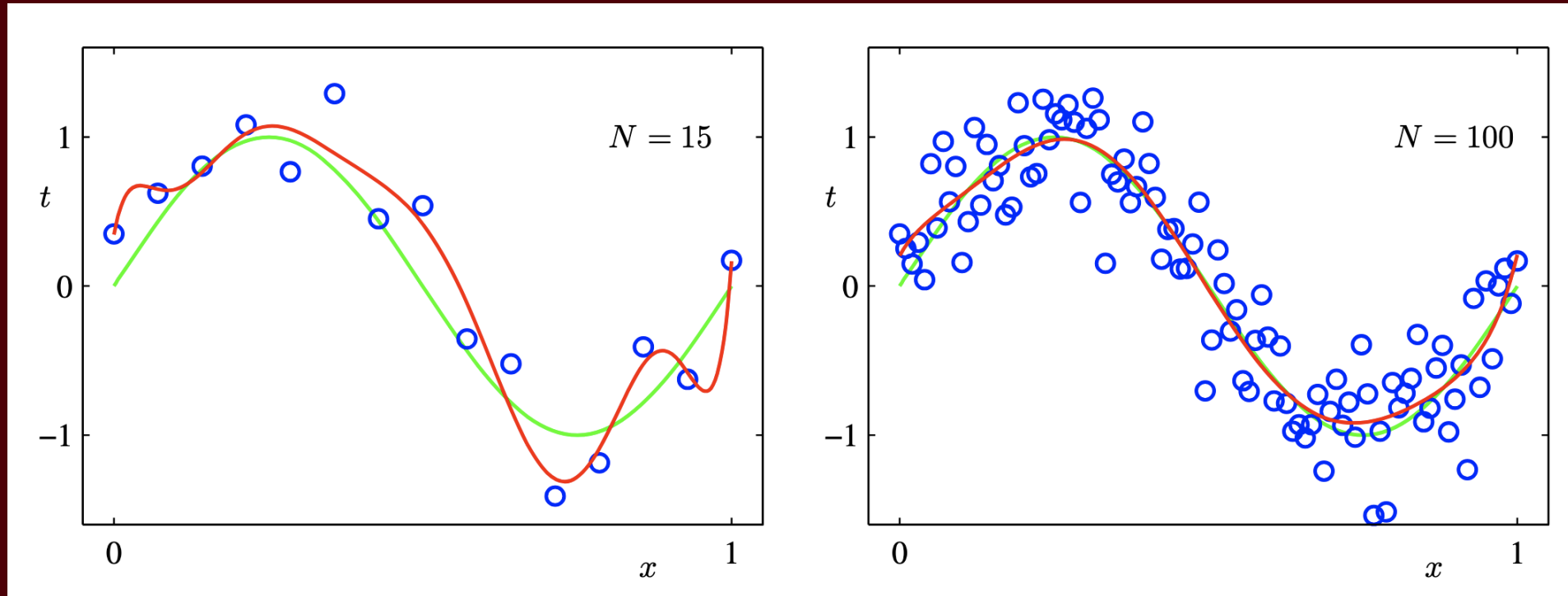


$Loss(w)$ is zero!

Overfitting

High Variance

What to do about overfitting?



Add more training data

(mini-batch) Stochastic Gradient Descent (SGD)

$\lambda = 0.01$

Initialize w and b randomly

$$l(w, b) = \sum_{i \in B} Cost(w, b)$$

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

 Compute: $dl(w, b)/dw$ and $dl(w, b)/db$

 Update w : $w = w - \lambda dl(w, b)/dw$

 Update b : $b = b - \lambda dl(w, b)/db$

 Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

Regularization

- Large weights lead to large variance. i.e. model fits to the training data too strongly.
- Solution: Minimize the loss but also try to keep the weight values small by doing the following:

$$\text{minimize} \quad L(w, b) + \alpha \sum_i |w_i|^2$$

Regularization

- Large weights lead to large variance. i.e. model fits to the training data too strongly.
- Solution: Minimize the loss but also try to keep the weight values small by doing the following:

minimize $L(w, b) + \alpha \sum_i |w_i|^2$

Regularizer term
e.g. L2- regularizer

SGD with Regularization (L-2)

$\lambda = 0.01$

$$l(w, b) = l(w, b) + \alpha \sum_i |w_i|^2$$

Initialize w and b randomly

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

Compute: $dl(w, b)/dw$ and $dl(w, b)/db$

Update w : $w = w - \lambda dl(w, b)/dw - \lambda \alpha w$

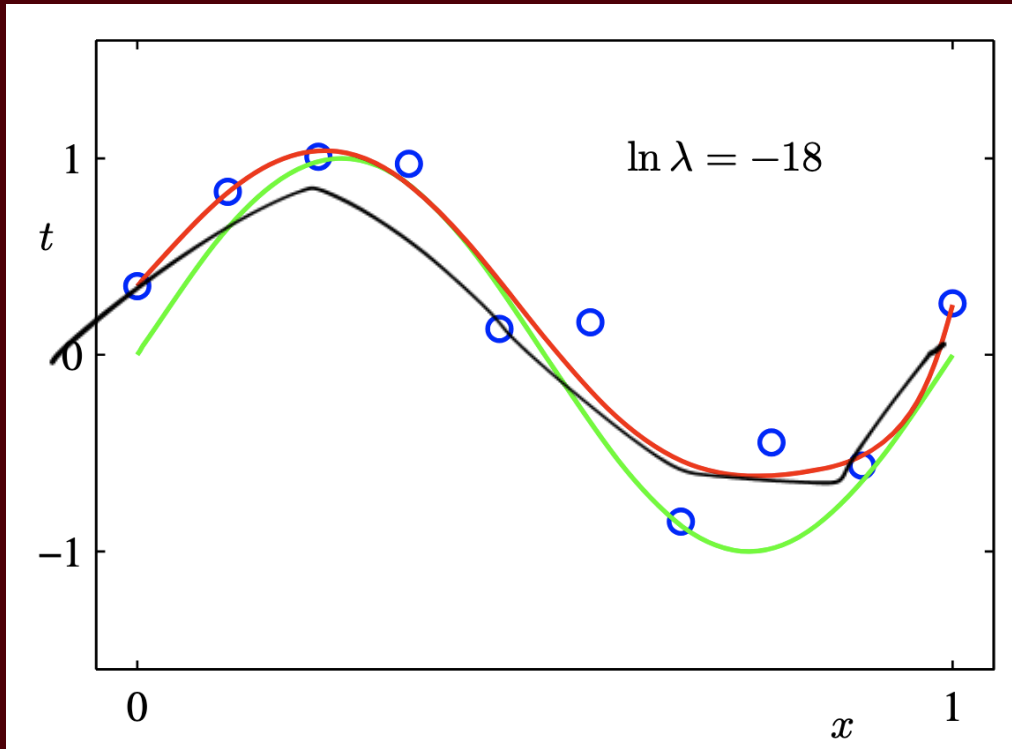
Update b : $b = b - \lambda dl(w, b)/db - \lambda \alpha w$

Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

Regularization: As Shown in Bishop's ML Book



	ln $\lambda = -18$
w_0^*	0.35
w_1^*	4.74
w_2^*	-0.77
w_3^*	-31.97
w_4^*	-3.89
w_5^*	55.28
w_6^*	41.32
w_7^*	-45.95
w_8^*	-91.53
w_9^*	72.68

$$\tilde{E}(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^N \{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\lambda}{2} \|\mathbf{w}\|^2$$

Revisiting Another Problem with SGD

$$\lambda = 0.01$$

$$l(w, b) = l(w, b) + \alpha \sum_i |w_i|^2$$

Initialize w and b randomly

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

Compute: $dl(w, b)/dw$ and $dl(w, b)/db$

Update w : $w = w - \lambda dl(w, b)/dw - \lambda \alpha w$

Update b : $b = b - \lambda dl(w, b)/db - \lambda \alpha w$

Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

These are only approximations to the true gradient with respect to $L(w, b)$

Revisiting Another Problem with SGD

$$\lambda = 0.01$$

$$l(w, b) = l(w, b) + \alpha \sum_i |w_i|^2$$

Initialize w and b randomly

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

Compute: $dl(w, b)/dw$ and $dl(w, b)/db$

Update w : $w = w - \lambda dl(w, b)/dw - \lambda \alpha w$

Update b : $b = b - \lambda dl(w, b)/db - \lambda \alpha w$

Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

This could lead to “un-learning” what has been learned in some previous steps of training.

Solution: Momentum Updates

$$\lambda = 0.01$$

$$l(w, b) = l(w, b) + \alpha \sum_i |w_i|^2$$

Initialize w and b randomly

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

Compute: $dl(w, b)/dw$ and $dl(w, b)/db$

Update w : $w = w - \lambda dl(w, b)/dw - \lambda \alpha w$

Update b : $b = b - \lambda dl(w, b)/db - \lambda \alpha w$

Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

Keep track of previous gradients in an accumulator variable! and use a weighted average with current gradient.

Solution: Momentum Updates

$$\lambda = 0.01 \quad \tau = 0.9$$

Initialize w and b randomly

$$l(w, b) = l(w, b) + \alpha \sum_i |w_i|^2$$

global v

for $e = 0, \text{num_epochs}$ **do**

for $b = 0, \text{num_batches}$ **do**

Compute: $dl(w, b)/dw$

Compute: $v = \tau v + dl(w, b)/dw + \alpha w$

Update w : $w = w - \lambda v$

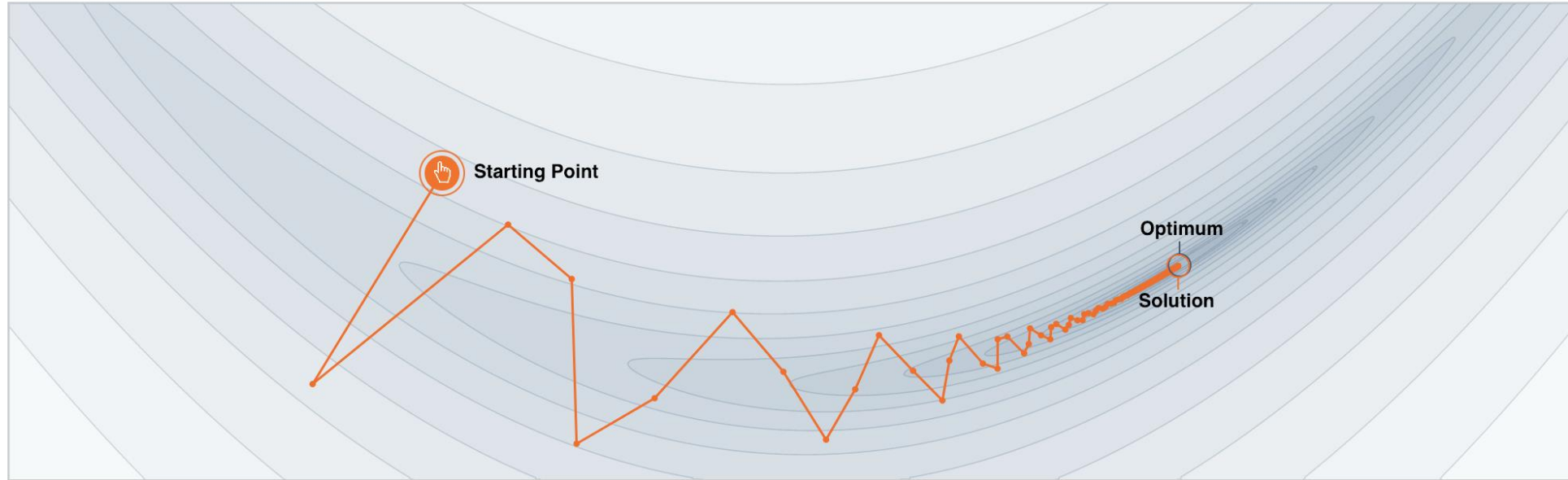
Print: $l(w, b)$ // Useful to see if this is becoming smaller or not.

end

end

Keep track of previous gradients in an accumulator variable! and use a weighted average with current gradient.

More on Momentum



Step-size $\alpha = 0.0050$



Momentum $\beta = 0.77$



We often think of Momentum as a means of dampening oscillations and speeding up the iterations, leading to faster convergence. But it has other interesting behavior. It allows a larger range of step-sizes to be used, and creates its own oscillations. What is going on?

<https://distill.pub/2017/momentum/>

Supervised Learning - Classification

Training Data



cat



dog



cat

-
-
-



bear

Test Data



-
-
-



Supervised Learning - Classification

Training Data

$$x_1 = [\text{img}] \quad y_1 = [\text{cat}]$$

$$x_2 = [\text{img}] \quad y_2 = [\text{dog}]$$

$$x_3 = [\text{img}] \quad y_3 = [\text{cat}]$$

•
•
•

$$x_n = [\text{img}] \quad y_n = [\text{bear}]$$

Supervised Learning - Classification

Training Data

inputs	targets / labels / ground truth	predictions
$x_1 = [x_{11} \ x_{12} \ x_{13} \ x_{14}]$	$y_1 = 1$	$\hat{y}_1 = 1$
$x_2 = [x_{21} \ x_{22} \ x_{23} \ x_{24}]$	$y_2 = 2$	$\hat{y}_2 = 2$
$x_3 = [x_{31} \ x_{32} \ x_{33} \ x_{34}]$	$y_3 = 1$	$\hat{y}_3 = 2$
⋮		
⋮		
⋮		
$x_n = [x_{n1} \ x_{n2} \ x_{n3} \ x_{n4}]$	$y_n = 3$	$\hat{y}_n = 1$

We need to find a function that maps x and y for any of them.

$$\hat{y}_i = f(x_i; \theta)$$

How do we "learn" the parameters of this function?

We choose ones that makes the following quantity small:

$$\sum_{i=1}^n Cost(\hat{y}_i, y_i)$$

Supervised Learning - Classification

Training Data



cat



dog



cat

-
-
-



bear

Test Data



-
-
-



Supervised Learning - Classification

Training Data

$$x_1 = [\text{img}] \quad y_1 = [\text{cat}]$$

$$x_2 = [\text{img}] \quad y_2 = [\text{dog}]$$

$$x_3 = [\text{img}] \quad y_3 = [\text{cat}]$$

•
•
•

$$x_n = [\text{img}] \quad y_n = [\text{bear}]$$

Supervised Learning - Classification

Training Data

inputs	targets / labels / ground truth	predictions
$x_1 = [x_{11} \ x_{12} \ x_{13} \ x_{14}]$	$y_1 = 1$	$\hat{y}_1 = 1$
$x_2 = [x_{21} \ x_{22} \ x_{23} \ x_{24}]$	$y_2 = 2$	$\hat{y}_2 = 2$
$x_3 = [x_{31} \ x_{32} \ x_{33} \ x_{34}]$	$y_3 = 1$	$\hat{y}_3 = 2$
⋮		
⋮		
⋮		
$x_n = [x_{n1} \ x_{n2} \ x_{n3} \ x_{n4}]$	$y_n = 3$	$\hat{y}_n = 1$

We need to find a function that maps x and y for any of them.

$$\hat{y}_i = f(x_i; \theta)$$

How do we "learn" the parameters of this function?

We choose ones that makes the following quantity small:

$$\sum_{i=1}^n Cost(\hat{y}_i, y_i)$$

Supervised Learning – Linear Softmax

Training Data

inputs

targets /
labels /
ground truth

$$x_1 = [x_{11} \ x_{12} \ x_{13} \ x_{14}] \quad y_1 = 1$$

$$x_2 = [x_{21} \ x_{22} \ x_{23} \ x_{24}] \quad y_2 = 2$$

$$x_3 = [x_{31} \ x_{32} \ x_{33} \ x_{34}] \quad y_3 = 1$$

•
•
•

$$x_n = [x_{n1} \ x_{n2} \ x_{n3} \ x_{n4}] \quad y_n = 3$$

Supervised Learning – Linear Softmax

Training Data

inputs	targets / labels / ground truth	predictions
$x_1 = [x_{11} \ x_{12} \ x_{13} \ x_{14}]$	$y_1 = [1 \ 0 \ 0]$	$\hat{y}_1 = [0.85 \ 0.10 \ 0.05]$
$x_2 = [x_{21} \ x_{22} \ x_{23} \ x_{24}]$	$y_2 = [0 \ 1 \ 0]$	$\hat{y}_2 = [0.20 \ 0.70 \ 0.10]$
$x_3 = [x_{31} \ x_{32} \ x_{33} \ x_{34}]$	$y_3 = [1 \ 0 \ 0]$	$\hat{y}_3 = [0.40 \ 0.45 \ 0.15]$
•		
•		
•		
$x_n = [x_{n1} \ x_{n2} \ x_{n3} \ x_{n4}]$	$y_n = [0 \ 0 \ 1]$	$\hat{y}_n = [0.40 \ 0.25 \ 0.35]$

Supervised Learning – Linear Softmax

$$x_i = [x_{i1} \ x_{i2} \ x_{i3} \ x_{i4}] \quad y_i = [1 \ 0 \ 0] \quad \hat{y}_i = [f_1 \ f_2 \ f_3]$$

$$a_1 = w_{11}x_{i1} + w_{12}x_{i2} + w_{13}x_{i3} + w_{14}x_{i4} + b_c$$

$$a_2 = w_{21}x_{i1} + w_{22}x_{i2} + w_{23}x_{i3} + w_{24}x_{i4} + b_d$$

$$a_3 = w_{31}x_{i1} + w_{32}x_{i2} + w_{33}x_{i3} + w_{34}x_{i4} + b_b$$

$$f_1 = e^{a_1} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_2 = e^{a_2} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_3 = e^{a_3} / (e^{a_1} + e^{a_2} + e^{a_3})$$

How do we find a good w and b ?

$$x_i = [x_{i1} \ x_{i2} \ x_{i3} \ x_{i4}] \quad y_i = [1 \ 0 \ 0] \quad \hat{y}_i = [f_1(w, b) \ f_2(w, b) \ f_3(w, b)]$$

We need to find w , and b that minimize the following:

$$L(w, b) = \sum_{i=1}^n \sum_{j=1}^3 -y_{i,j} \log(\hat{y}_{i,j}) = \sum_{i=1}^n -\log(\hat{y}_{i,label}) = \sum_{i=1}^n -\log f_{i,label}(w, b)$$

Why?

Computing Analytic Gradients

This is what we have:

$$L(w, b) = \sum_{i=1}^n \sum_{j=1}^3 -y_{i,j} \log(\hat{y}_{i,j}) = \sum_{i=1}^n -\log(\hat{y}_{i,label}) = \sum_{i=1}^n -\log f_{i,label}(w, b)$$

To simplify let's assume $n = 1$

$$\ell(W, b) = -\log(\hat{y}_{label}(W, b)) = -\log\left(\frac{\exp(a_{label}(W, b))}{\sum_{k=1}^3 \exp(a_k(W, b))}\right)$$

Supervised Learning – Linear Softmax

$$x = [x_1 \ x_2 \ x_3 \ x_4] \quad y = [1 \ 0 \ 0] \quad \hat{y} = [f_1 \ f_2 \ f_3]$$

$$a_1 = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + w_{14}x_4 + b_c$$

$$a_2 = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + w_{24}x_4 + b_d$$

$$a_3 = w_{31}x_1 + w_{32}x_2 + w_{33}x_3 + w_{34}x_4 + b_b$$

$$f_1 = e^{a_1} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_2 = e^{a_2} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_3 = e^{a_3} / (e^{a_1} + e^{a_2} + e^{a_3})$$

Computing Analytic Gradients

This is what we have:

$$\ell(W, b) = -\log(\hat{y}_{label}(W, b)) = -\log\left(\frac{\exp(a_{label}(W, b))}{\sum_{k=1}^3 \exp(a_k(W, b))}\right)$$

Computing Analytic Gradients

This is what we have:

$$\ell(W, b) = -\log(\hat{y}_{label}(W, b)) = -\log\left(\frac{\exp(a_{label}(W, b))}{\sum_{k=1}^3 \exp(a_k(W, b))}\right)$$

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

Reminder: $a_i = (w_{i,1}x_1 + w_{i,2}x_2 + w_{i,3}x_3 + w_{i,4}x_4) + b_i$

Computing Analytic Gradients

This is what we have:

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

Computing Analytic Gradients

This is what we have:

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

This is what we need:

$$\frac{\partial \ell}{\partial w_{ij}} \quad \text{for each } w_{ij} \qquad \frac{\partial \ell}{\partial b_i} \quad \text{for each } b_i$$

Computing Analytic Gradients

This is what we have:

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

Step 1: Chain Rule of Calculus

$$\frac{\partial \ell}{\partial w_{ij}} = \frac{\partial \ell}{\partial a_i} \frac{\partial a_i}{\partial w_{ij}}$$

$$\frac{\partial \ell}{\partial b_i} = \frac{\partial \ell}{\partial a_i} \frac{\partial a_i}{\partial b_i}$$

Computing Analytic Gradients

This is what we have:

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

Step 1: Chain Rule of Calculus

Let's do these first

$$\frac{\partial \ell}{\partial w_{ij}} = \frac{\partial \ell}{\partial a_i} \boxed{\frac{\partial a_i}{\partial w_{ij}}}$$

$$\frac{\partial \ell}{\partial b_i} = \frac{\partial \ell}{\partial a_i} \boxed{\frac{\partial a_i}{\partial b_i}}$$

Computing Analytic Gradients

$$\frac{\partial a_i}{\partial w_{ij}}$$

$$\frac{\partial a_i}{\partial b_i}$$

$$a_i = (w_{i,1}x_1 + w_{i,2}x_2 + w_{i,3}x_3 + w_{i,4}x_4) + b_i$$

$$\frac{\partial a_i}{\partial w_{i,3}} = \frac{\partial}{\partial w_{i,3}} (w_{i,1}x_1 + w_{i,2}x_2 + w_{i,3}x_3 + w_{i,4}x_4) + b_i$$

$$\frac{\partial a_i}{\partial w_{i,3}} = x_3$$

$$\frac{\partial a_i}{\partial w_{i,j}} = x_j$$

Computing Analytic Gradients

$$\frac{\partial a_i}{\partial w_{i,j}} = x_j$$

$$\frac{\partial a_i}{\partial b_i}$$

$$a_i = (w_{i,1}x_1 + w_{i,2}x_2 + w_{i,3}x_3 + w_{i,4}x_4) + b_i$$

$$\frac{\partial a_i}{\partial b_i} = \frac{\partial}{\partial b_i} (w_{i,1}x_1 + w_{i,2}x_2 + w_{i,3}x_3 + w_{i,4}x_4) + b_i$$

$$\frac{\partial a_i}{\partial b_i} = 1$$

Computing Analytic Gradients

$$\frac{\partial a_i}{\partial w_{i,j}} = x_j$$

$$\frac{\partial a_i}{\partial b_i} = 1$$

Computing Analytic Gradients

This is what we have:

$$\ell = -\log\left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)}\right)$$

Step 1: Chain Rule of Calculus

Now let's do this one (same for both!)

$$\frac{\partial \ell}{\partial w_{ij}} = \boxed{\frac{\partial \ell}{\partial a_i}} \frac{\partial a_i}{\partial w_{ij}}$$

$$\frac{\partial \ell}{\partial b_i} = \boxed{\frac{\partial \ell}{\partial a_i}} \frac{\partial a_i}{\partial b_i}$$

Computing Analytic Gradients

$$\begin{aligned}\frac{\partial \ell}{\partial a_i} &= \frac{\partial}{\partial a_i} \left[-\log \left(\frac{\exp(a_{label})}{\sum_{k=1}^3 \exp(a_k)} \right) \right] \\ &= \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{label} \right]\end{aligned}$$

In our cat, dog, bear classification example: $i = \{1, 2, 3\}$

Computing Analytic Gradients

$$\begin{aligned}\frac{\partial \ell}{\partial a_i} &= \frac{\partial}{\partial a_i} \left[-\log \left(\frac{\exp(a_{\text{label}})}{\sum_{k=1}^3 \exp(a_k)} \right) \right] \\ &= \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]\end{aligned}$$

In our cat, dog, bear classification example: $i = \{1, 2, 3\}$

Let's say: label = 2

We need:

$$\frac{\partial \ell}{\partial a_1} \quad \frac{\partial \ell}{\partial a_2} \quad \frac{\partial \ell}{\partial a_3}$$

Computing Analytic Gradients

$$= \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_1} \quad \frac{\partial \ell}{\partial a_3} \quad \text{when } i \neq \text{label:}$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\partial}{\partial a_i} \log \left(\sum_{k=1}^3 \exp(a_k) \right)$$

$$\frac{\partial \ell}{\partial a_i} = \left(\frac{1}{\sum_{k=1}^3 \exp(a_k)} \right) \left(\frac{\partial}{\partial a_i} \sum_{k=1}^3 \exp(a_k) \right)$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\exp(a_i)}{\sum_{k=1}^3 \exp(a_k)} = \hat{y}_i$$

Supervised Learning – Linear Softmax

$$x_i = [x_{i1} \ x_{i2} \ x_{i3} \ x_{i4}] \quad y_i = [1 \ 0 \ 0] \quad \hat{y}_i = [f_1 \ f_2 \ f_3]$$

$$a_1 = w_{11}x_{i1} + w_{12}x_{i2} + w_{13}x_{i3} + w_{14}x_{i4} + b_c$$

$$a_2 = w_{21}x_{i1} + w_{22}x_{i2} + w_{23}x_{i3} + w_{24}x_{i4} + b_d$$

$$a_3 = w_{31}x_{i1} + w_{32}x_{i2} + w_{33}x_{i3} + w_{34}x_{i4} + b_b$$

$$f_1 = e^{a_1} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_2 = e^{a_2} / (e^{a_1} + e^{a_2} + e^{a_3})$$

$$f_3 = e^{a_3} / (e^{a_1} + e^{a_2} + e^{a_3})$$

Computing Analytic Gradients

$$= \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_1} \quad \frac{\partial \ell}{\partial a_3} \quad \text{when } i \neq \text{label:}$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\partial}{\partial a_i} \log \left(\sum_{k=1}^3 \exp(a_k) \right)$$

$$\frac{\partial \ell}{\partial a_i} = \left(\frac{1}{\sum_{k=1}^3 \exp(a_k)} \right) \left(\frac{\partial}{\partial a_i} \sum_{k=1}^3 \exp(a_k) \right)$$

$$\frac{\partial \ell}{\partial a_i} = \frac{\exp(a_i)}{\sum_{k=1}^3 \exp(a_k)} = \hat{y}_i$$

Computing Analytic Gradients

$$= \frac{\partial}{\partial a_i} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_2}$$

when $i = \text{label}$:

$$\frac{\partial \ell}{\partial a_{\text{label}}} = \frac{\partial}{\partial a_{\text{label}}} \left[\log \left(\sum_{k=1}^3 \exp(a_k) \right) - a_{\text{label}} \right]$$

$$\frac{\partial \ell}{\partial a_{\text{label}}} = \frac{\partial}{\partial a_{\text{label}}} \log \left(\sum_{k=1}^3 \exp(a_k) \right) - 1$$

$$\frac{\partial \ell}{\partial a_{\text{label}}} = \left(\frac{1}{\sum_{k=1}^3 \exp(a_k)} \right) \left(\frac{\partial}{\partial a_{\text{label}}} \sum_{k=1}^3 \exp(a_k) \right) - 1$$

$$\frac{\partial \ell}{\partial a_{\text{label}}} = \frac{\exp(a_{\text{label}})}{\sum_{k=1}^3 \exp(a_k)} - 1 \quad \hat{y}_i - 1$$

Computing Analytic Gradients

label = 2

$$\frac{\partial \ell}{\partial a_1} = \hat{y}_1 \qquad \frac{\partial \ell}{\partial a_2} = \hat{y}_2 - 1 \qquad \frac{\partial \ell}{\partial a_3} = \hat{y}_3$$

$$\frac{\partial \ell}{\partial a} = \begin{bmatrix} \frac{\partial \ell}{\partial a_1} \\ \frac{\partial \ell}{\partial a_2} \\ \frac{\partial \ell}{\partial a_3} \end{bmatrix} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 - 1 \\ \hat{y}_3 \end{bmatrix} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \hat{y}_3 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \hat{y} - y$$

$$\frac{\partial \ell}{\partial a_i} = \hat{y}_i - y_i$$

Computing Analytic Gradients

$$\frac{\partial \ell}{\partial w_{ij}} = \frac{\partial \ell}{\partial a_i} \frac{\partial a_i}{\partial w_{ij}}$$

$$\frac{\partial \ell}{\partial b_i} = \frac{\partial \ell}{\partial a_i} \frac{\partial a_i}{\partial b_i}$$

$$\frac{\partial a_i}{\partial w_{i,j}} = x_j$$

$$\frac{\partial a_i}{\partial b_i} = 1$$

$$\frac{\partial \ell}{\partial a_i} = \hat{y}_i - y_i$$

$$\frac{\partial \ell}{\partial w_{i,j}} = (\hat{y}_i - y_i) x_j$$

$$\frac{\partial \ell}{\partial b_i} = (\hat{y}_i - y_i)$$

Questions?