Linear classifiers:
Parameter learning

Learn a probabilistic classification model

"The sushi & everything else were awesome!"

Definite +1

P(y=+1|**x**= *"The sushi & everything else were awesome!"*) = 0.99

"The sushi was good, the service was OK"

Not sure

P(y=+1|**x**= *"The sushi was good, the service was OK"*) = 0.55

Many classifiers provide a degree of certainty:

Output label

Input sentence

P(y|**x**)

Extremely useful in practice

A (linear) classifier

- Will use training data to learn a weight or coefficient for each word

Word	Coefficient	Value
	$\hat{w}_0$	-2.0
good	$\hat{w}_1$	1.0
great	$\hat{w}_2$	1.5
awesome	$\hat{w}_3$	2.7
bad	$\hat{w}_4$	-1.0
terrible	$\hat{w}_5$	-2.1
awful	$\hat{w}_6$	-3.3
restaurant, the, we, …	$\hat{w}_7, \hat{w}_8, \hat{w}_9, \ldots$	0.0
…		…

Logistic regression model

$$\text{Score}(\mathbf{x}_i) = \hat{\mathbf{w}}^T h(\mathbf{x}_i)$$

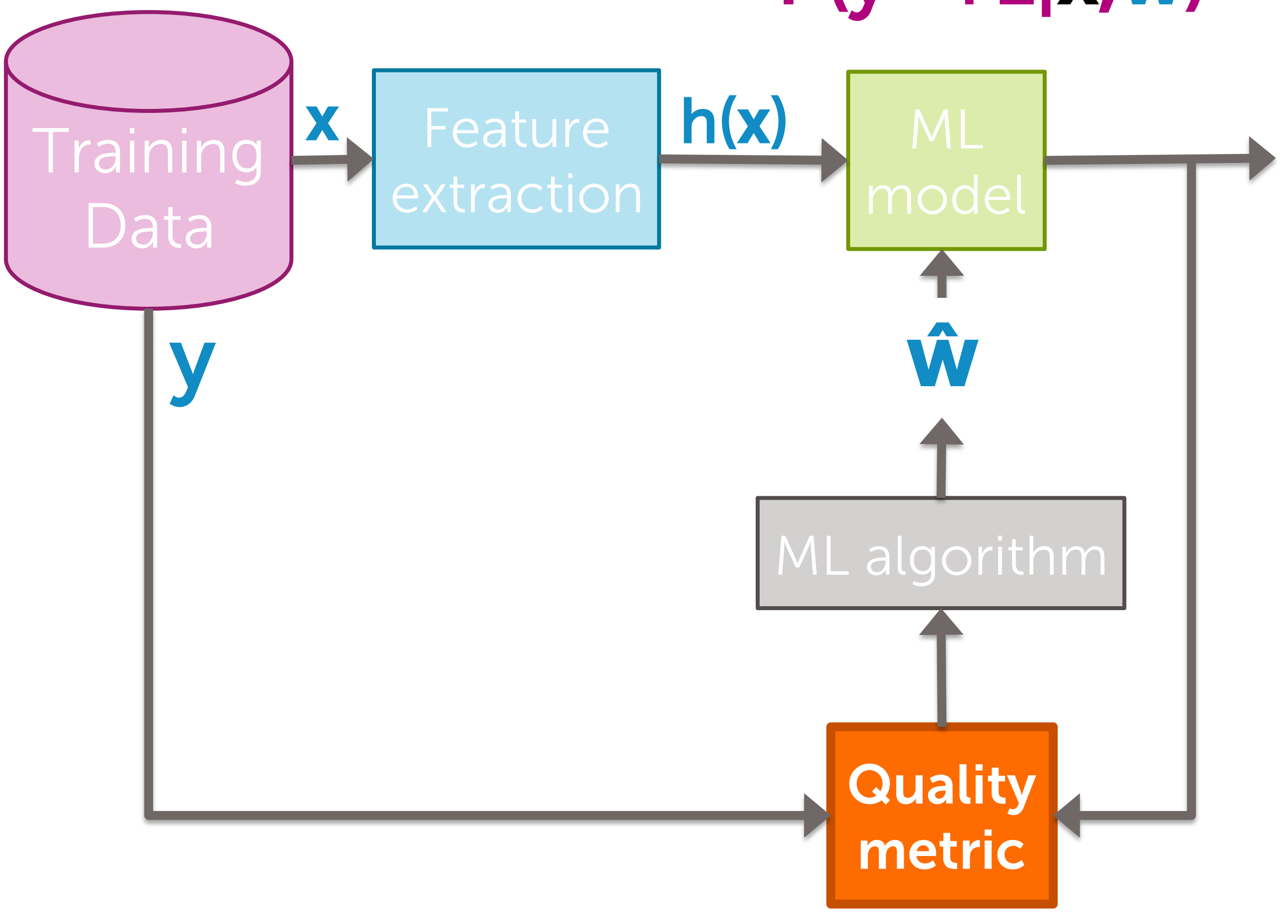
$-\infty$ 0.0 $+\infty$

0.0 0.5 1.0

$$\hat{P}(y=+1|\mathbf{x},\hat{\mathbf{w}}) = \frac{1}{1+e^{-\hat{\mathbf{w}}^T h(\mathbf{x})}}$$

Quality metric for logistic regression: Maximum likelihood estimation

$$\hat{P}(y=+1|\mathbf{x},\hat{\mathbf{w}}) = \frac{1}{1+e^{-\hat{\mathbf{w}}^\top h(\mathbf{x})}}$$



Learning problem

Training data:
N observations ($\mathbf{x}_i$,y_i)

$\mathbf{x}$[1] = #awesome	$\mathbf{x}$[2] = #awful	y = sentiment
2	1	+1
0	2	-1
3	3	-1
4	1	+1
1	1	+1
2	4	-1
0	3	-1
0	1	-1
2	1	+1

Optimize **quality metric** on training data

$\hat{\mathbf{w}}$

Finding best coefficients

x[1] = #awesome	**x**[2] = #awful	y = sentiment
0	2	-1
3	3	-1
2	4	-1
0	3	-1
0	1	-1

x[1] = #awesome	**x**[2] = #awful	y = sentiment
2	1	+1
4	1	+1
1	1	+1
2	1	+1

$$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 0.0$$

$$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 1.0$$

Pick $\hat{\mathbf{w}}$ that makes

Quality metric = Likelihood function

Negative data points

$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 0.0$

Positive data points

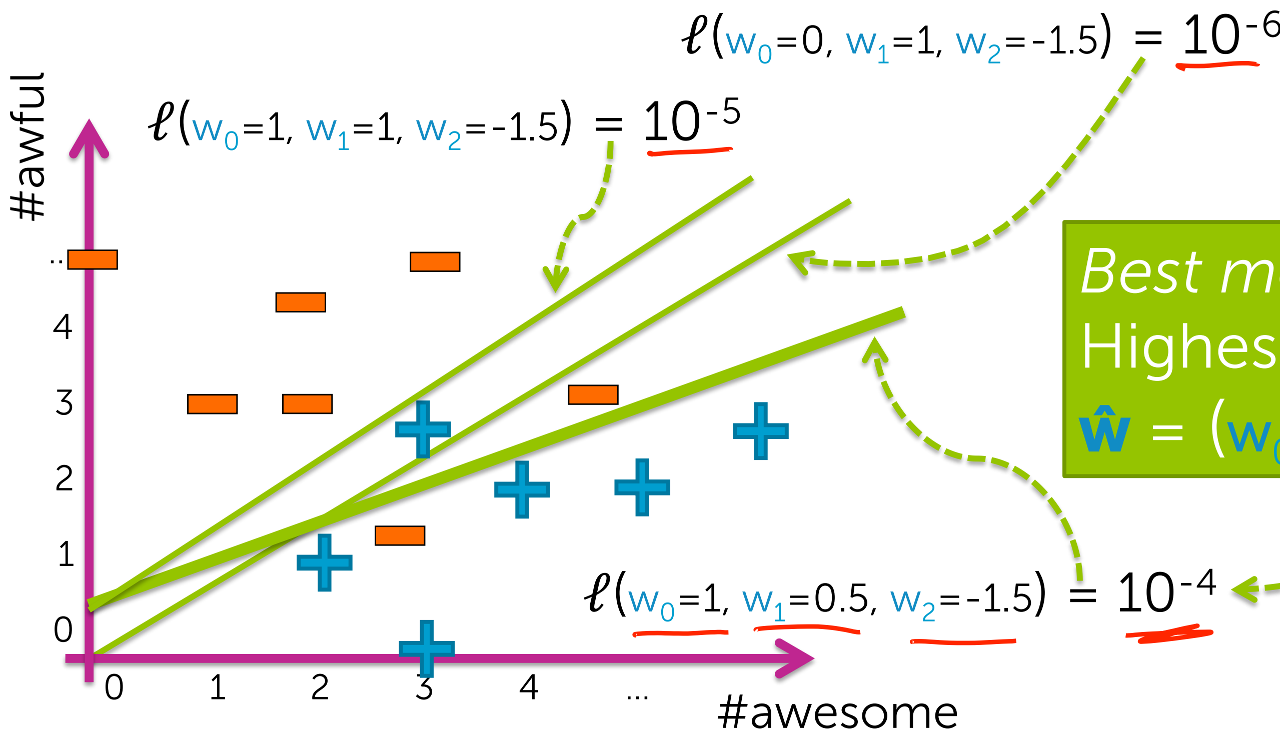
$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 1.0$

No $\hat{\mathbf{w}}$ achieves perfect predictions (usually)

Likelihood $\ell(\mathbf{w})$: Measures quality of fit for model with coefficients $\mathbf{w}$

Find "best" classifier

Maximize likelihood over all possible w_0, w_1, w_2



Best model:

Highest likelihood $\ell(\mathbf{w})$

$\hat{\mathbf{w}} = (w_0=1, w_1=0.5, w_2=-1.5)$

gradient ascent to find $\hat{w}$

Data likelihood

Quality metric: probability of data

x[1] = #awesome	**x**[2] = #awful	y = sentiment
2	1	+1

x_1 y_1 =

If model good, should predict:

$\hat{y}_1 = +1$

Pick **w** to maximize:

$P(y=+1 \mid x_1, w) = P(y=+1 \mid x[1]=2, x[2]=1, w)$

x[1] = #awesome	**x**[2] = #awful	y = sentiment
0	2	-1

x_2 = y_2 =

If model good, should predict:

$\hat{y}_2 = -1$

Pick **w** to maximize:

$P(y=-1 \mid x_2, w)$

Maximizing likelihood (probability of data)

Data point	x[1]	x[2]	y	Choose **w** to maximize
$\mathbf{x}_1,y_1$	2	1	+1	$P(y=+1 \mid x_1, w) = P(y=+1 \mid x[1]=2, x[2]=1, w)$
$\mathbf{x}_2,y_2$	0	2	-1	$P(y=-1 \mid x_2, w)$
$\mathbf{x}_3,y_3$	3	3	-1	$P(y=-1 \mid x_3, w)$
$\mathbf{x}_4,y_4$	4	1	+1	$P(y=+1 \mid x_4, w)$
$\mathbf{x}_5,y_5$	1	1	+1	
$\mathbf{x}_6,y_6$	2	4	-1	
$\mathbf{x}_7,y_7$	0	3	-1	
$\mathbf{x}_8,y_8$	0	1	-1	
$\mathbf{x}_9,y_9$	2	1	+1	

Must combine into single measure of quality ?

Multiply probabilities

$P(y=+1 \mid x_1, w)\, P(y=-1 \mid x_2, w)\, P(y=-1 \mid x_3, w) \ldots$

Learn logistic regression model with maximum likelihood estimation (MLE)

Data point	**x**[1]	**x**[2]	y	Choose **w** to maximize
$\mathbf{x}_1, y_1$	2	1	$y_i =$ +1	P(y=+1|**x**[1]=2, **x**[2]=1,**w**)
$\mathbf{x}_2, y_2$	0	2	-1	P(y=-1|**x**[1]=0, **x**[2]=2,**w**)
$\mathbf{x}_3, y_3$	3	3	-1	P(y=-1|**x**[1]=3, **x**[2]=3,**w**)
$\mathbf{x}_4, y_4$	4	1	+1	P(y=+1|**x**[1]=4, **x**[2]=1,**w**)

$$\ell(\mathbf{w}) = \underbrace{P(y_1|\mathbf{x}_1,\mathbf{w})\; P(y_2|\mathbf{x}_2,\mathbf{w})\; P(y_3|\mathbf{x}_3,\mathbf{w})\; P(y_4|\mathbf{x}_4,\mathbf{w})}$$

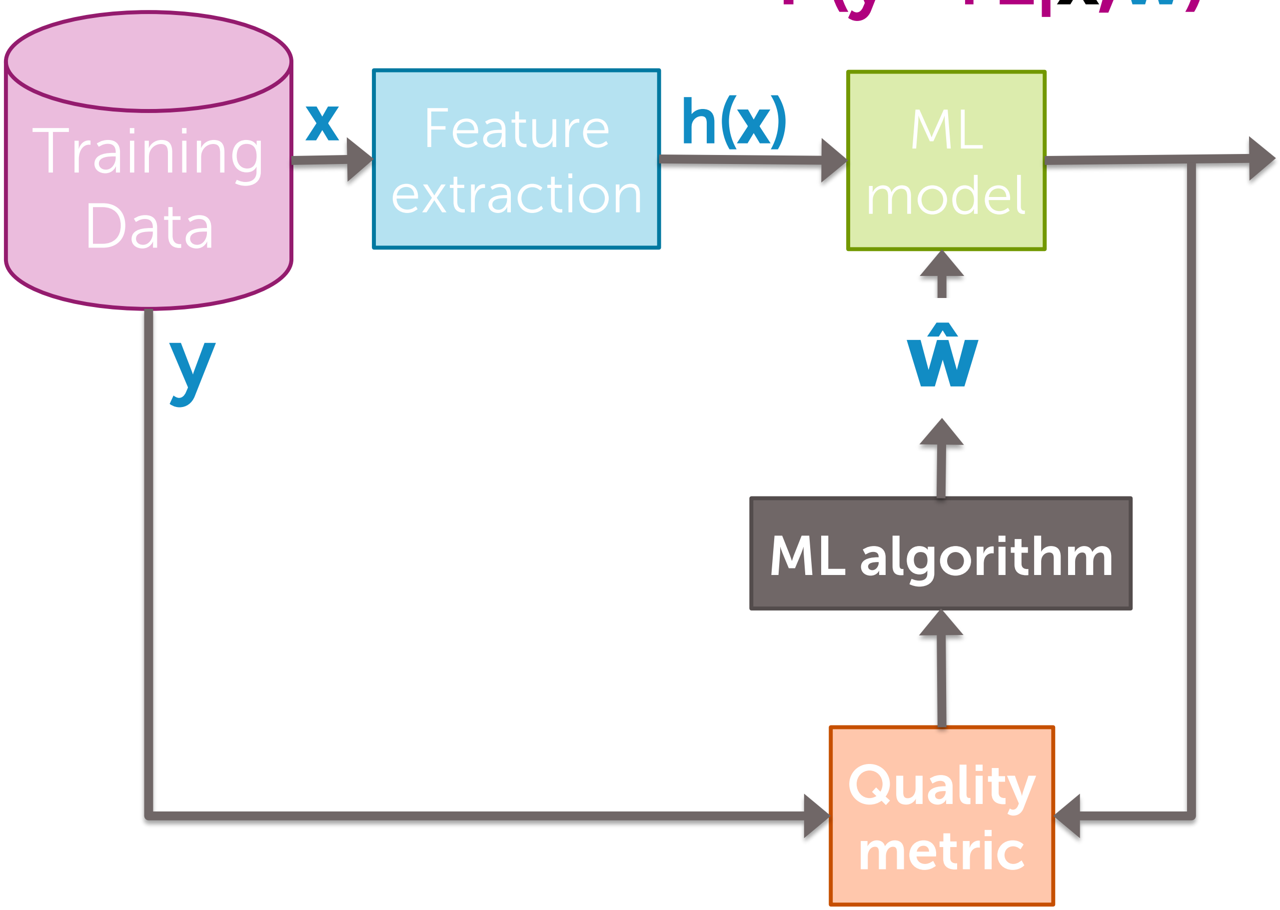
Num. of data points → N

$$\ell(w) = \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})$$

← pick w to make this fn. as large as possible

Finding best linear classifier with gradient ascent

$$\hat{P}(y=+1|\mathbf{x},\hat{\mathbf{w}}) = \frac{1}{1+e^{-\hat{\mathbf{w}}^\top h(\mathbf{x})}}$$

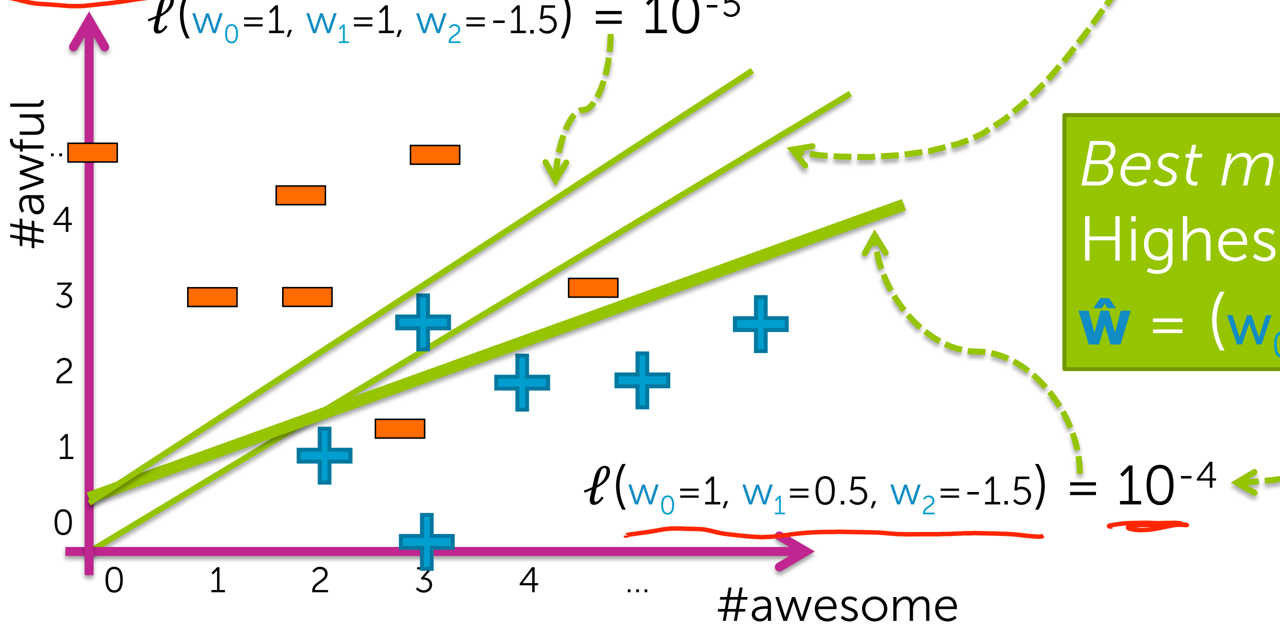


Find “best” classifier

Maximize likelihood over all possible w_0, w_1, w_2

$$\boxed{\ell(\mathbf{w}) = \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})}$$

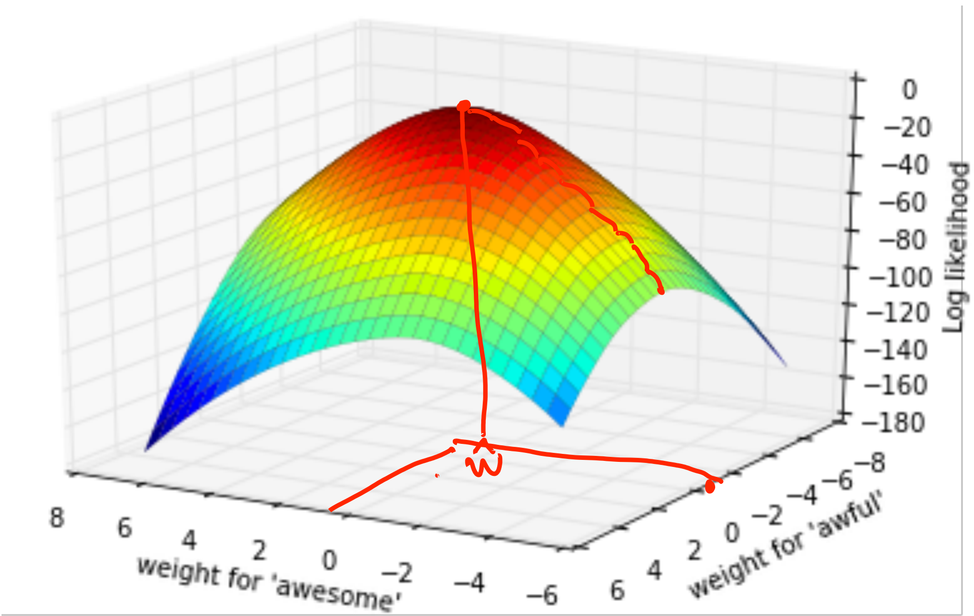
$\ell(w_0=0,\ w_1=1,\ w_2=-1.5) = 10^{-6}$



Best model:
Highest likelihood $\ell(\mathbf{w})$
$\hat{\mathbf{w}} = (w_0=1,\ w_1=0.5,\ w_2=-1.5)$

optimize with gradient ascent

Maximizing likelihood



Maximize function over all possible w_0, w_1, w_2

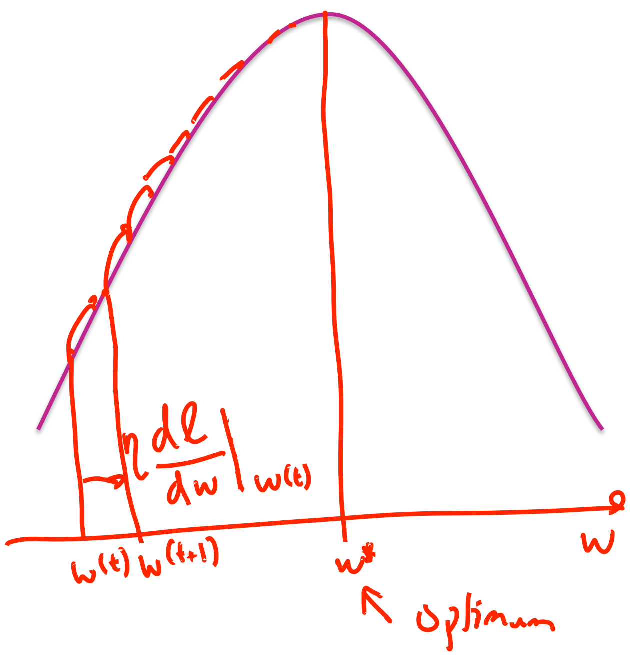
$$\max_{w_0,w_1,w_2} \underbrace{\prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})}_{\ell(w_0,w_1,w_2)}$$

$\ell(w_0,w_1,w_2)$ is a function of 3 variables

No closed-form solution ➔ use gradient ascent

Review of gradient ascent

Finding the max via hill climbing



Algorithm:

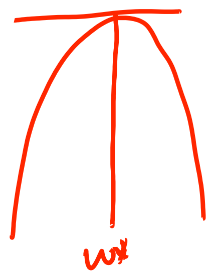
while not converged

$$w^{(t+1)} \leftarrow w^{(t)} + \eta \frac{d\ell}{dw}\Big|_{w^{(t)}}$$

step size

Convergence criteria

For convex functions, optimum occurs when



$$\frac{d\ell}{dw} = 0$$

In practice, stop when

$$\left.\frac{d\ell}{dw}\right|_{w^{(t)}} < \varepsilon$$

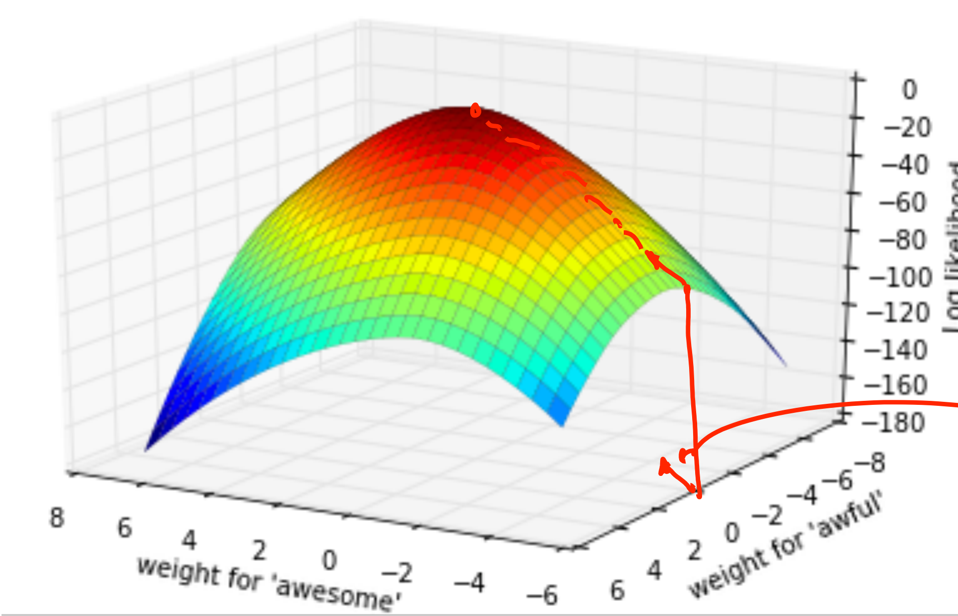
↑ tolerance

Algorithm:

while not converged

$$w^{(t+1)} \leftarrow w^{(t)} + \eta \left.\frac{d\ell}{dw}\right|_{w^{(t)}}$$

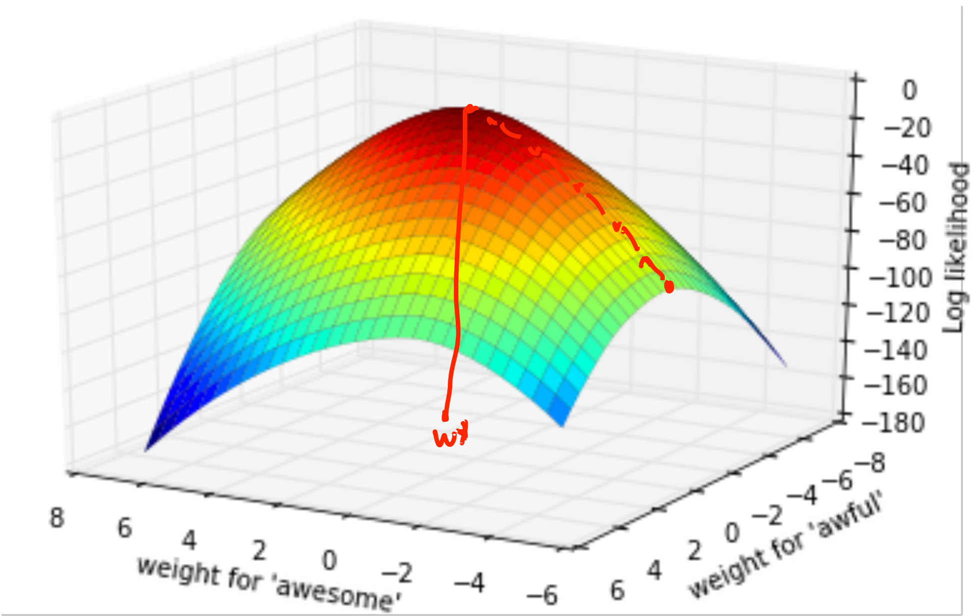
Moving to multiple dimensions: Gradients

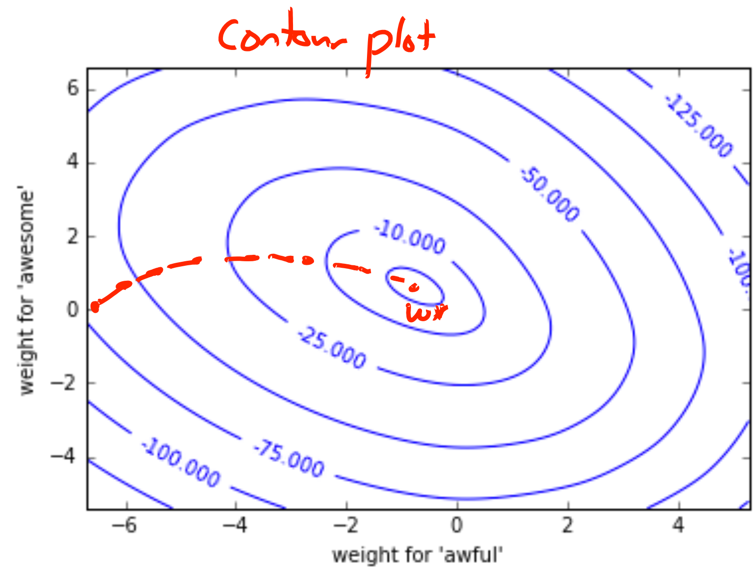


$$\nabla \ell(\mathbf{w}) = \begin{bmatrix} \frac{\partial \ell}{\partial w_0} \\ \frac{\partial \ell}{\partial w_1} \\ \vdots \\ \frac{\partial \ell}{\partial w_D} \end{bmatrix}$$

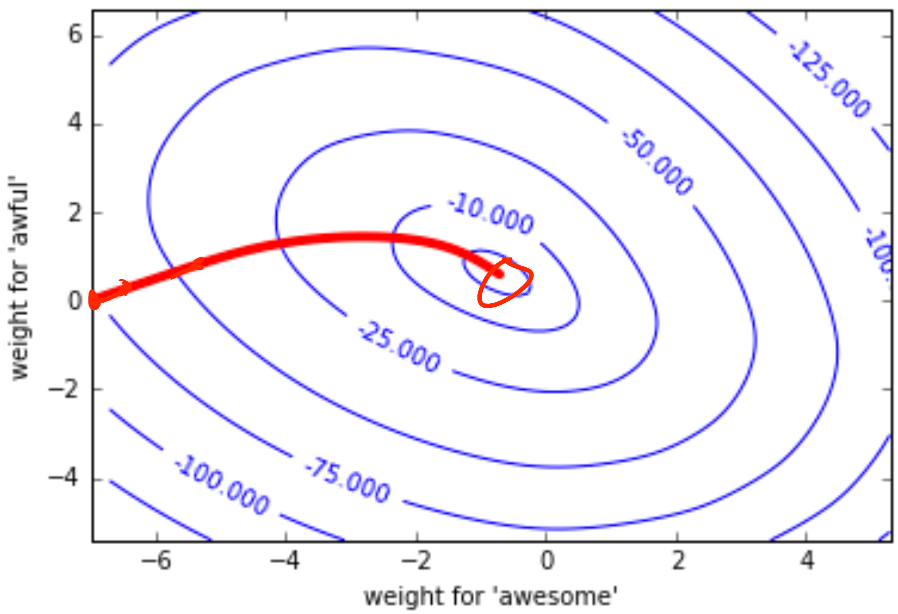
D+1 dim vector

Contour plots





Gradient ascent



Algorithm:

$w^{(0)} = 0$, random , or something smart.

while not converged

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} + \eta \nabla \ell(\mathbf{w}^{(t)})$$

step size

Learning algorithm for logistic regression

Derivative of (log-)likelihood

Sum over data points

Feature value

Difference between truth and prediction

$$\frac{\partial \ell(\mathbf{w})}{\partial \mathbf{w}_j} = \sum_{i=1}^{N} h_j(\mathbf{x}_i)\overbrace{\left(\mathbb{1}[y_i = +1] - \underbrace{P(y = +1 \mid \mathbf{x}_i, \mathbf{w})}_{\text{predict } x_i \text{ is positive}}\right)}^{\text{Difference between truth and prediction}}$$

Indicator function:

$$\mathbb{1}[y_i = +1] = \begin{cases} 1 & \text{if } y_i = +1 \\ 0 & \text{if } y_i = -1 \end{cases}$$

Computing derivative

$$\frac{\partial \ell(\mathbf{w}^{(t)})}{\partial \mathbf{w}_j} = \sum_{i=1}^{N} h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w}^{(t)})\Big)$$

$w^{(t)}$:

$w_0^{(t)}$	0
$w_1^{(t)}$	1
$w_2^{(t)}$	-2

$\frac{\partial \ell}{\partial w_1}$

$h_1(x) = \#$ awesome

x[1]	x[2]	y	P(y=+1|x_i,w)	Contribution to derivative for w_1
2	1	+1	0.5	$2(1-0.5) = 1$
0	2	-1	0.02	$0(0-0.02) = 0$
3	3	-1	0.05	$3(0-0.05) = -0.15$
4	1	+1	0.88	$4(1-0.88) = 0.48$

Total derivative:

$\frac{\partial \ell(w^{(t)})}{\partial w_1} = 1 + 0 - 0.15 + 0.48 = 1.33$

$w_1^{(t+1)} = w_1^{(t)} + \eta \frac{\partial \ell(w^{(t)})}{\partial w_1} \quad \Big| \quad \eta = 0.1$

$= 1 + 0.1 \times 1.33 = 1.133$

Derivative of (log-)likelihood: Interpretation

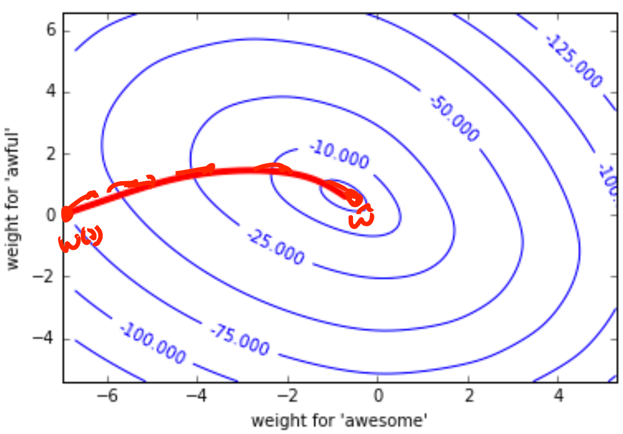
Sum over data points

Feature value

$$\frac{\partial \ell(\mathbf{w})}{\partial \mathbf{w}_j} = \sum_{i=1}^{N} \underbrace{h_j(\mathbf{x}_i)\overbrace{\left(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w})\right)}^{\text{Difference between truth and prediction}}}_{\Delta_i}$$

If $h_j(\mathbf{x}_i)=1$:	$P(y=+1\mid\mathbf{x}_i,\mathbf{w}) \approx 1$	$P(y=+1\mid\mathbf{x}_i,\mathbf{w}) \approx 0$
$y_i=+1$	$\Delta_i = (1-1) \approx 0$ ↳ don't change anything!	$\Delta_i \approx 1 \Rightarrow$ increase w_j $\Rightarrow$ Score(x_i) becomes larger $\Rightarrow P(y=+1\mid x_i,w)$ increases
$y_i=-1$	$\Delta_i = -1 \Rightarrow w_j$ to decrease $\Rightarrow$ Score(x_i) decreases $\Rightarrow P(y=+1\mid x_i,w)$ decrease	$\Delta_i \approx 0 \Rightarrow$ don't change anything

Summary of gradient ascent for logistic regression



init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), t=1

while $||\nabla \ell(\mathbf{w}^{(t)})|| > \varepsilon$ tolerance

$$\frac{1}{1+e^{-\mathbf{w}^{(t)T}h(x_i)}}$$

for j=0,…,D coefficient

partial[j] = $\sum_{i=1}^{N} h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i=+1] - P(y=+1 \mid \mathbf{x}_i, \mathbf{w}^{(t)})\Big)$

$w_j^{(t+1)} \leftarrow w_j^{(t)} + \eta$ partial[j]

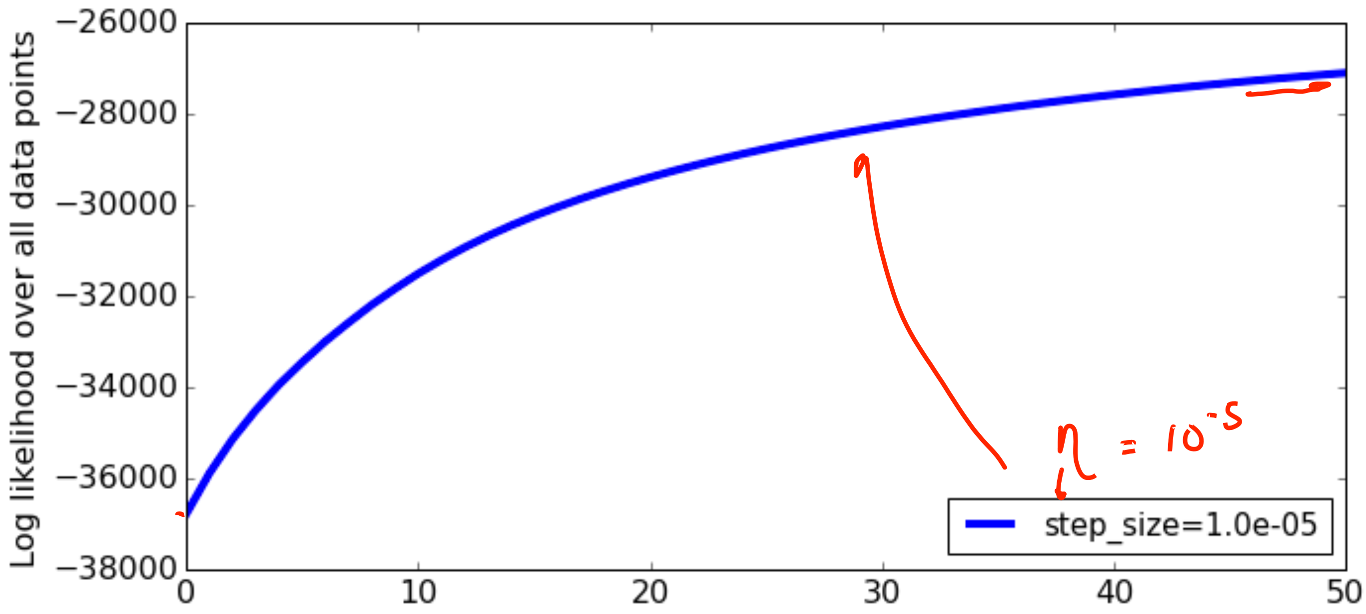
t ← t + 1

step size

$$\frac{\partial \ell(w^{(t)})}{\partial w_j}$$

Choosing the step size η

Learning curve:
Plot quality (likelihood) over iterations



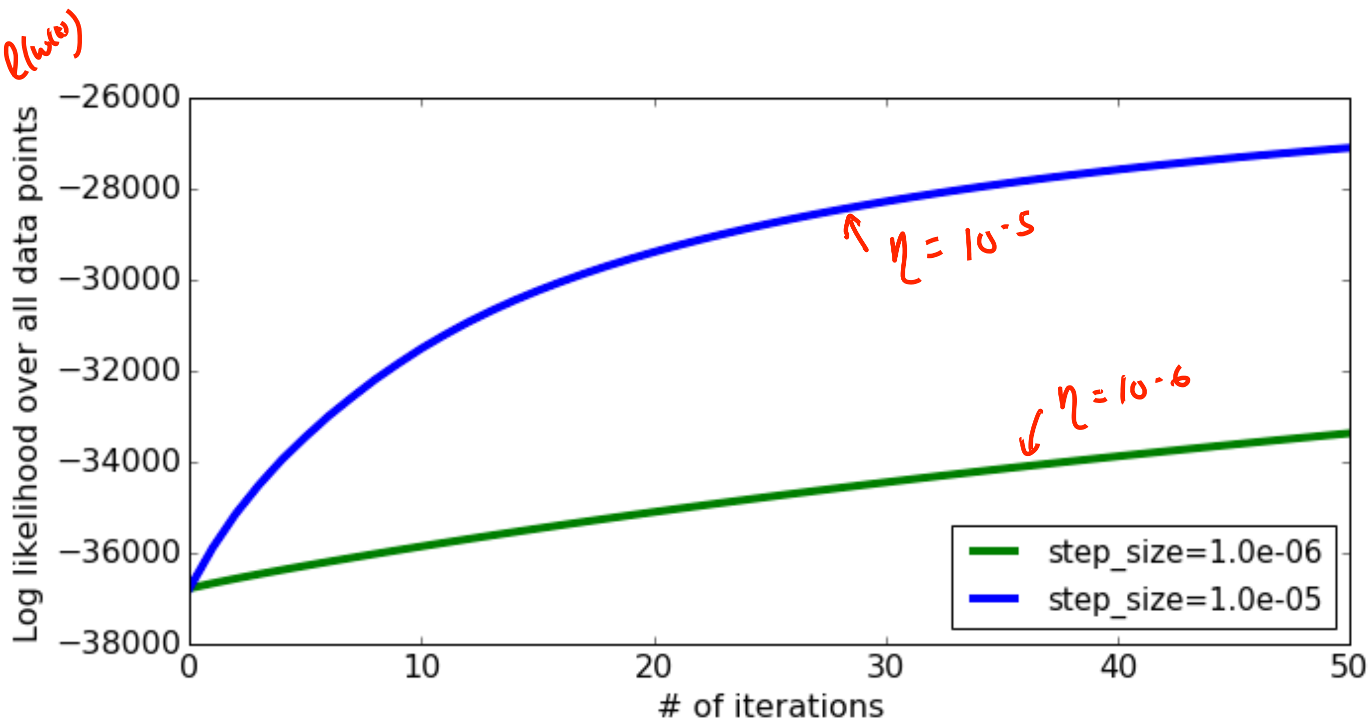
$\ln \prod_{i=1}^{N} P(y_i \mid x_i, w^{(t)})$

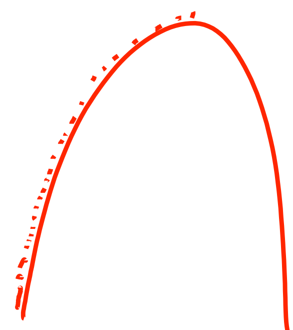
$\eta = 10^{-5}$

t

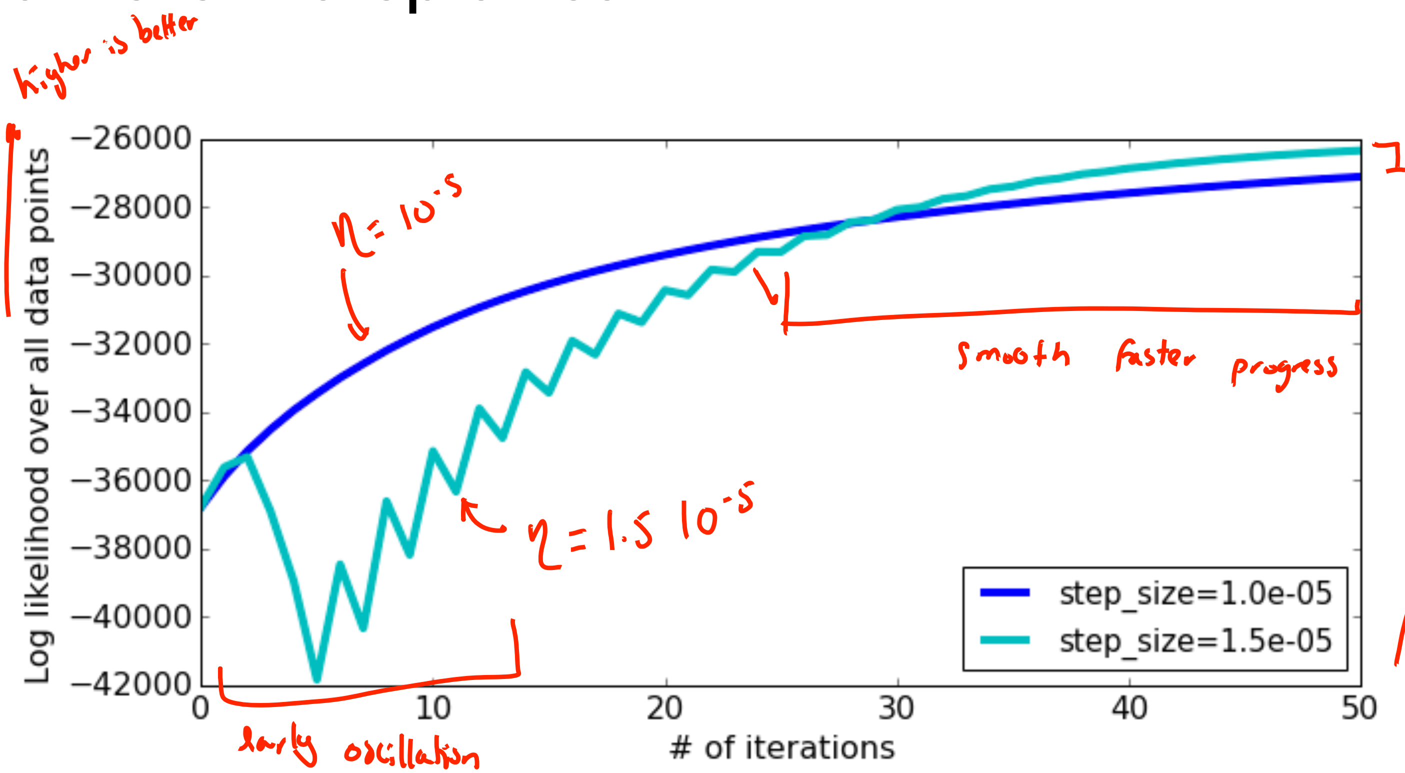
→ need more than 50 iterations

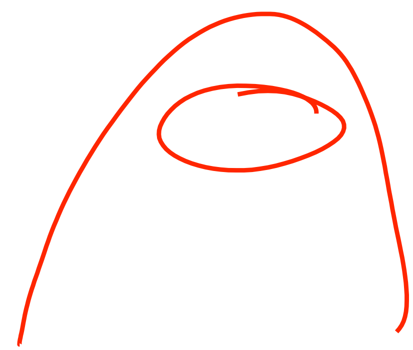
If step size is too small, can take a long time to converge



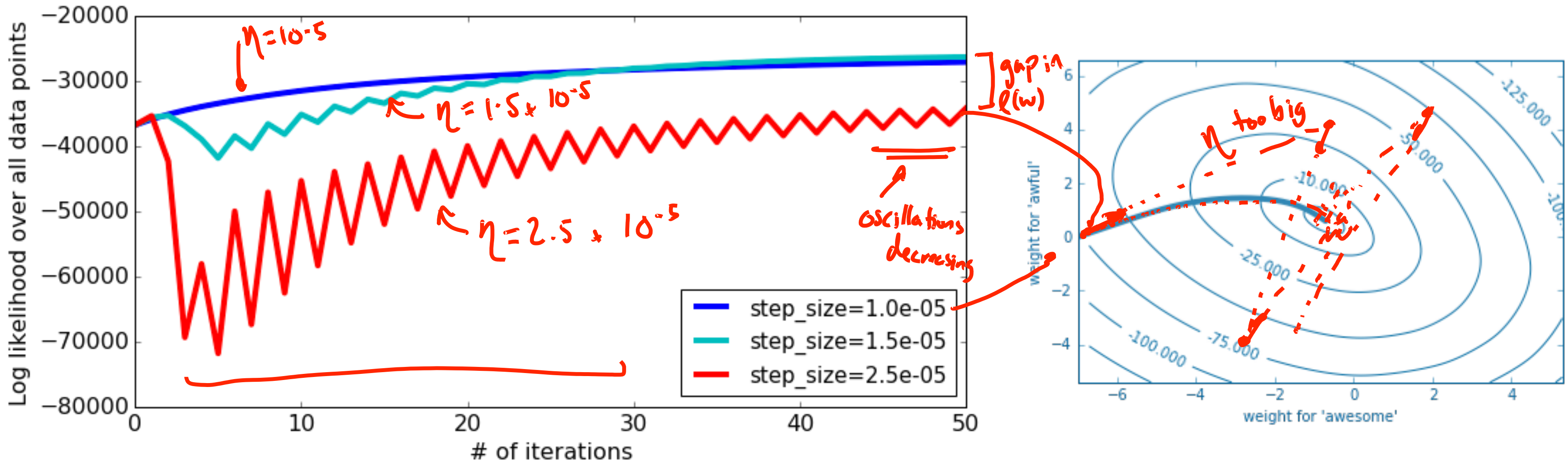


Compare converge with different step sizes

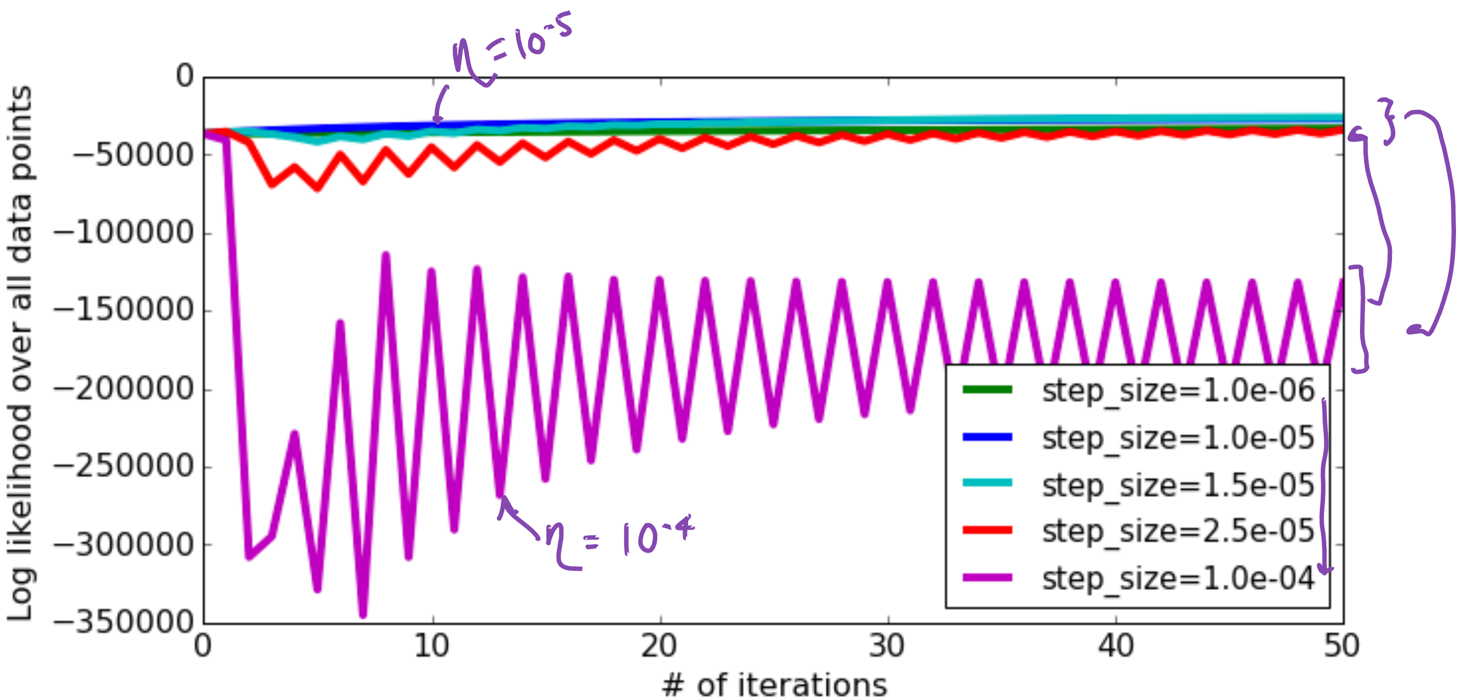




Careful with step sizes that are too large



Very large step sizes can even cause divergence or wild oscillations



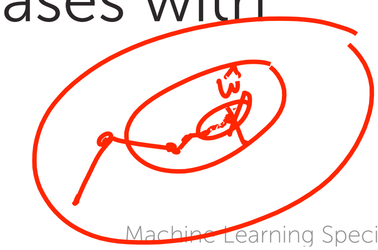
Simple rule of thumb for picking step size η

- Unfortunately, picking step size requires a lot of trial and error ☹
- Try a several values, exponentially spaced
  - **Goal**: plot learning curves to
    - find one η that is too small (smooth but moving too slowly)
    - find one η that is too large (oscillation or divergence)
- Try values in between to find "best" η

↳ exponentially space, pick one that leads best training data likelihood

- *Advanced tip*: can also try step size that decreases with iterations, e.g.,

$$\eta_t = \frac{\eta_0}{t}$$



Summary of logistic regression classifier

What you can do now…

- Measure quality of a classifier using the likelihood function
- Interpret the likelihood function as the probability of the observed data
- Learn a logistic regression model with gradient descent