Linear classifiers: Parameter learning

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington

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

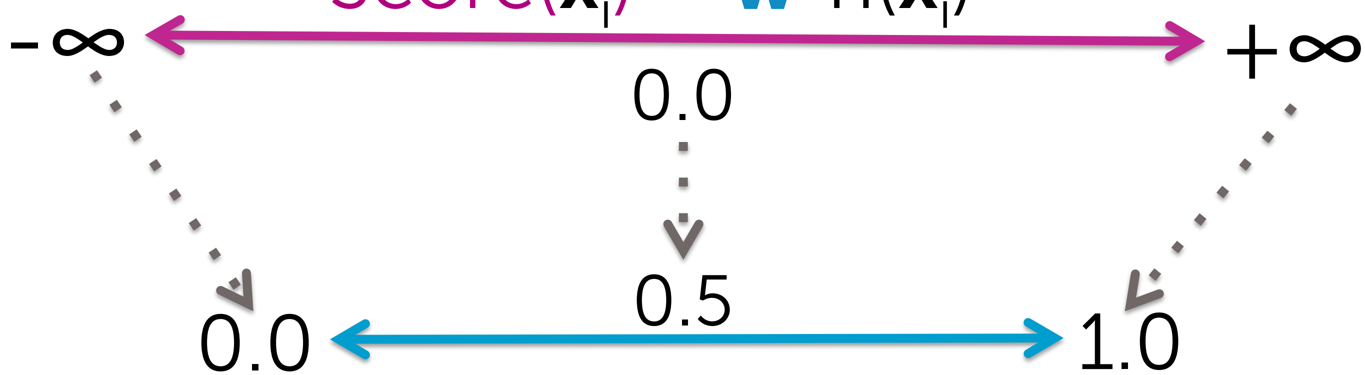
A (linear) classifier

- 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

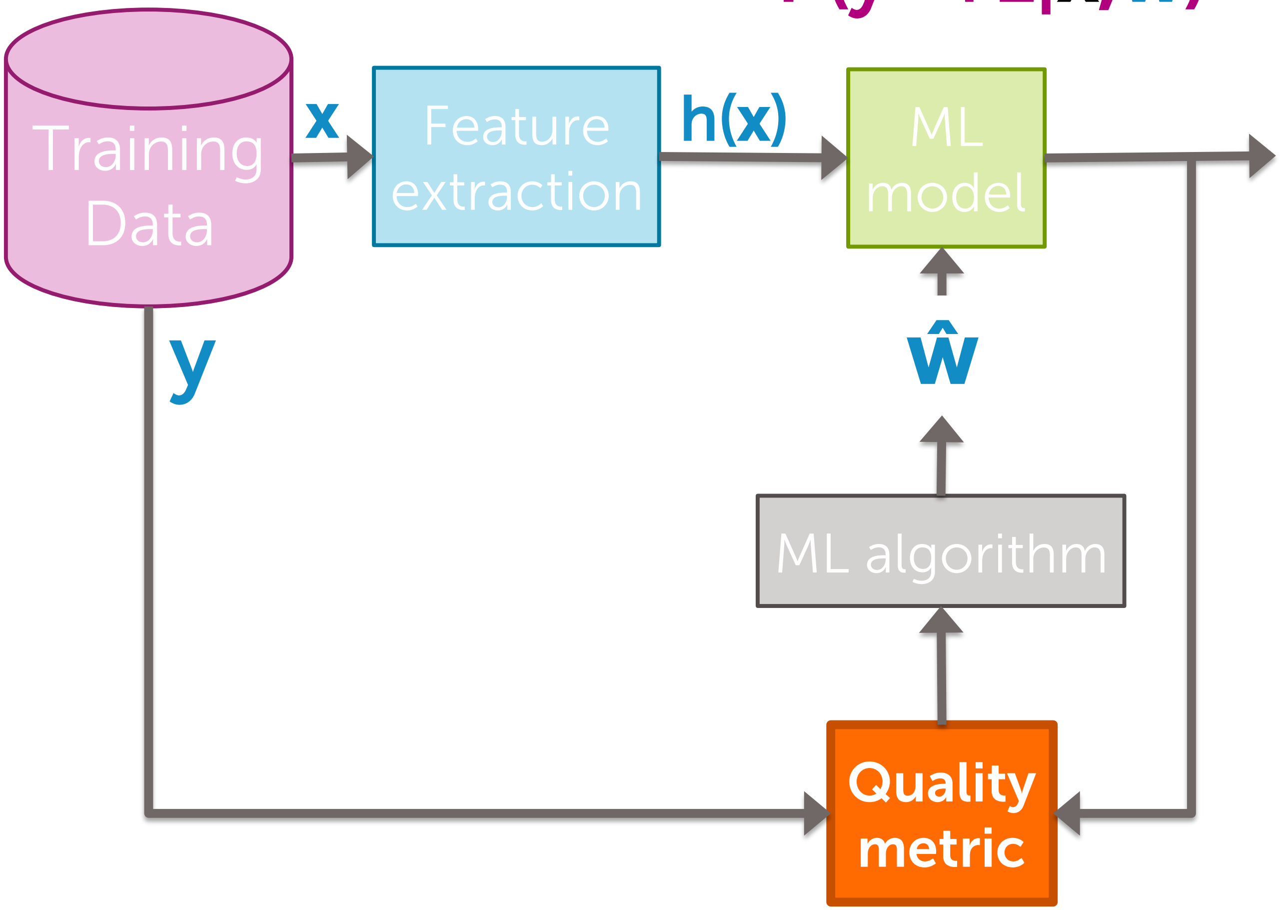
$\text{Score}(\mathbf{x}_i) = \hat{\mathbf{w}}^T h(\mathbf{x}_i)$



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

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

$\hat{\mathbf{w}}$

MOVE TO
HEAD SHOT

Finding best coefficients

x[1] = #awesome	**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

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
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
1	1	+1
2	1	+1

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

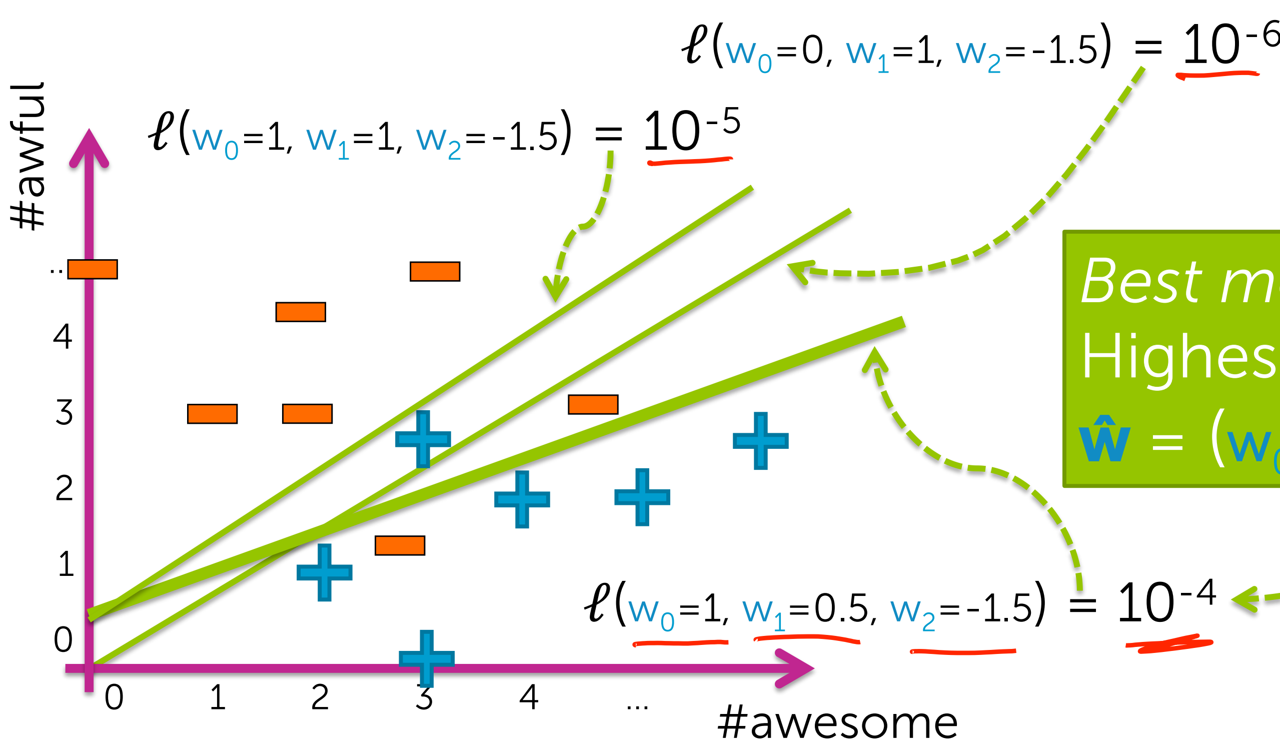
$P(y=+1|\mathbf{x}_i,\mathbf{w}) = 1.0$

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	$P(y=+1\|\mathbf{x}_1,\mathbf{w}) = P(y=+1\|x[1]=2, x[2]=1, \mathbf{w})$
$\mathbf{x}_2,y_2$	0	2	-1	$P(y=-1\|\mathbf{x}_2,\mathbf{w})$
$\mathbf{x}_3,y_3$	3	3	-1	$P(y=-1\|\mathbf{x}_3,\mathbf{w})$
$\mathbf{x}_4,y_4$	4	1	+1	$P(y=+1\|\mathbf{x}_4,\mathbf{w})$
$\mathbf{x}_5,y_5$	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

$P(y=+1|\mathbf{x}_1,\mathbf{w})\, P(y=-1|\mathbf{x}_2,\mathbf{w})\, P(y=-1|\mathbf{x}_3,\mathbf{w}) \ldots$

The reason you multiply is that you assume that every row is independent of each other.

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	$P(y=+1 \mid \mathbf{x}[1]=2, \mathbf{x}[2]=1, \mathbf{w})$
$\mathbf{x}_2, y_2$	0	2	-1	$P(y=-1 \mid \mathbf{x}[1]=0, \mathbf{x}[2]=2, \mathbf{w})$
$\mathbf{x}_3, y_3$	3	3	-1	$P(y=-1 \mid \mathbf{x}[1]=3, \mathbf{x}[2]=3, \mathbf{w})$
$\mathbf{x}_4, y_4$	4	1	+1	$P(y=+1 \mid \mathbf{x}[1]=4, \mathbf{x}[2]=1, \mathbf{w})$

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

Num. of data points

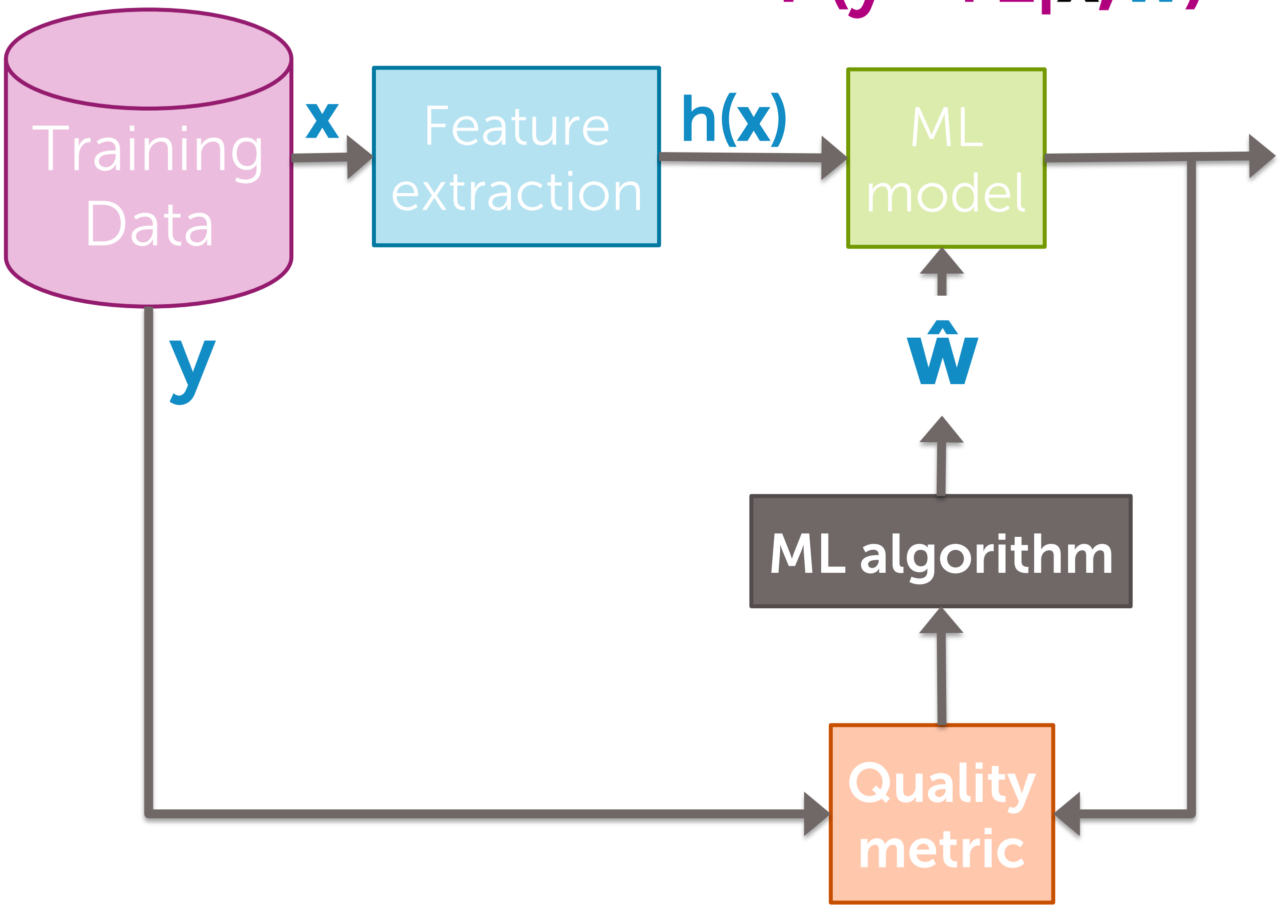
$$\ell(w) = \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})$$

pick w to make this fn. as large as possible

MOVE TO FULL BODY SHOT

Finding best linear classifier with gradient ascent

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

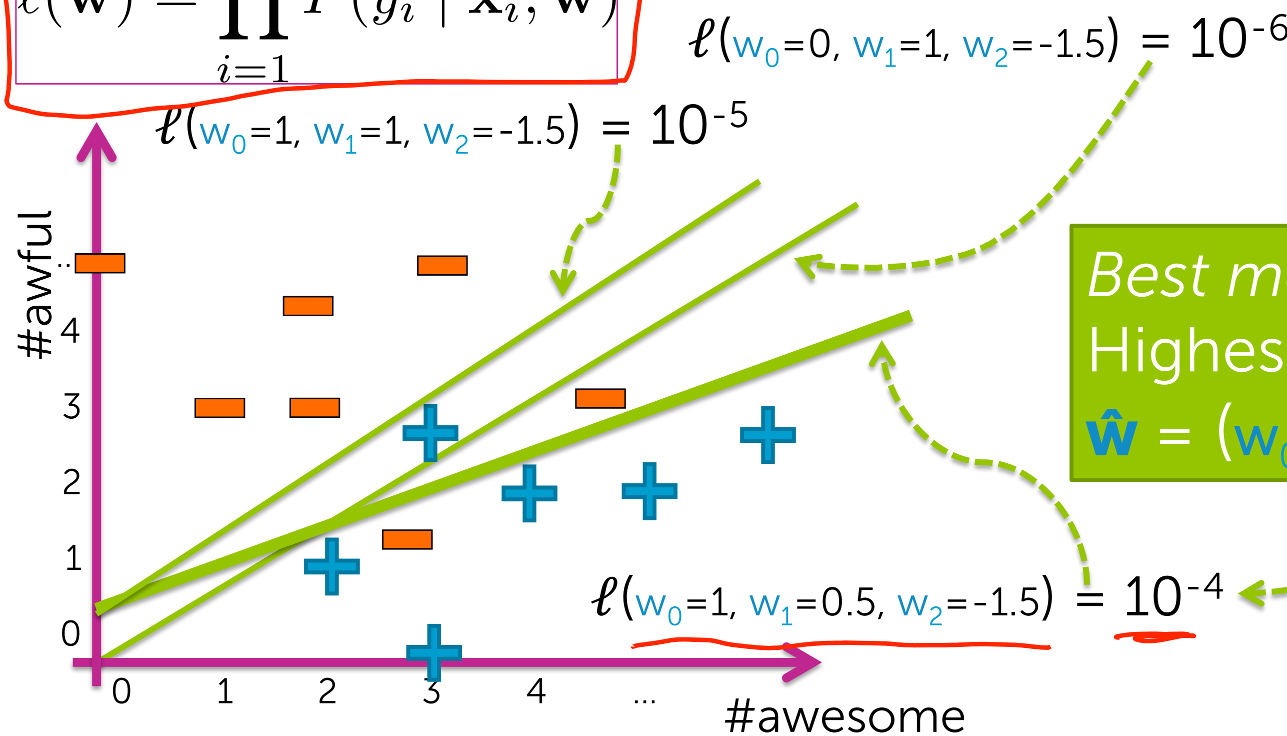


MOVE TO HEAD SHOT

Find "best" classifier

Maximize likelihood over all possible w_0, w_1, w_2

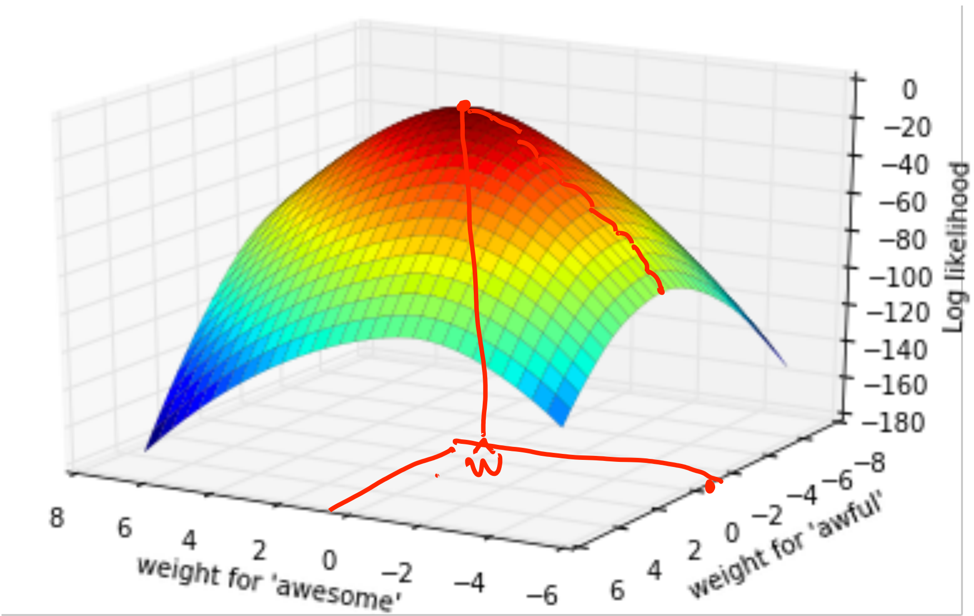
$$\boxed{\ell(\mathbf{w}) = \prod_{i=1}^{N} P(y_i \mid \mathbf{x}_i, \mathbf{w})}$$



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

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

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

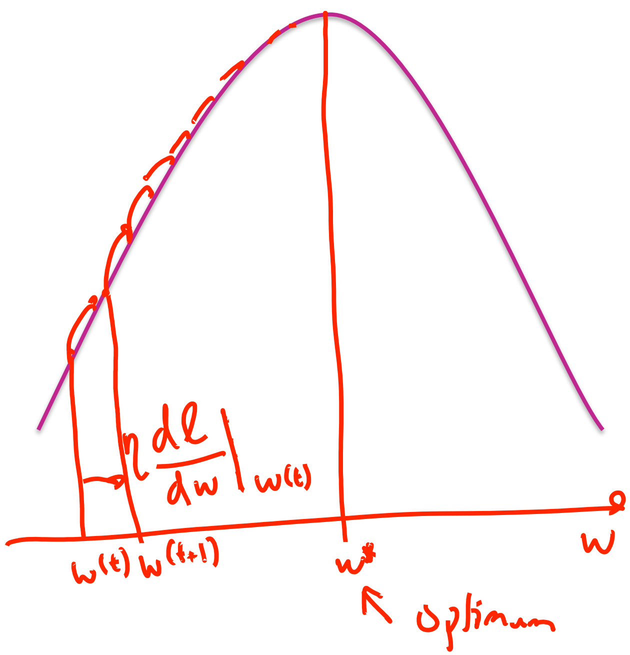
No closed-form solution ➔ use gradient ascent

MOVE TO FULL BODY SHOT

Review of gradient ascent

MOVE TO HEAD SHOT

Finding the max via hill climbing



Algorithm:

while not converged

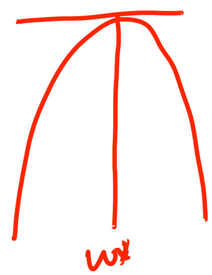
$$w^{(t+1)} \leftarrow w^{(t)} + \eta \frac{d\ell}{dw}\Big|_{w^{(t)}}$$

step size

Convergence criteria

For convex functions,
optimum occurs when

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



In practice, stop when

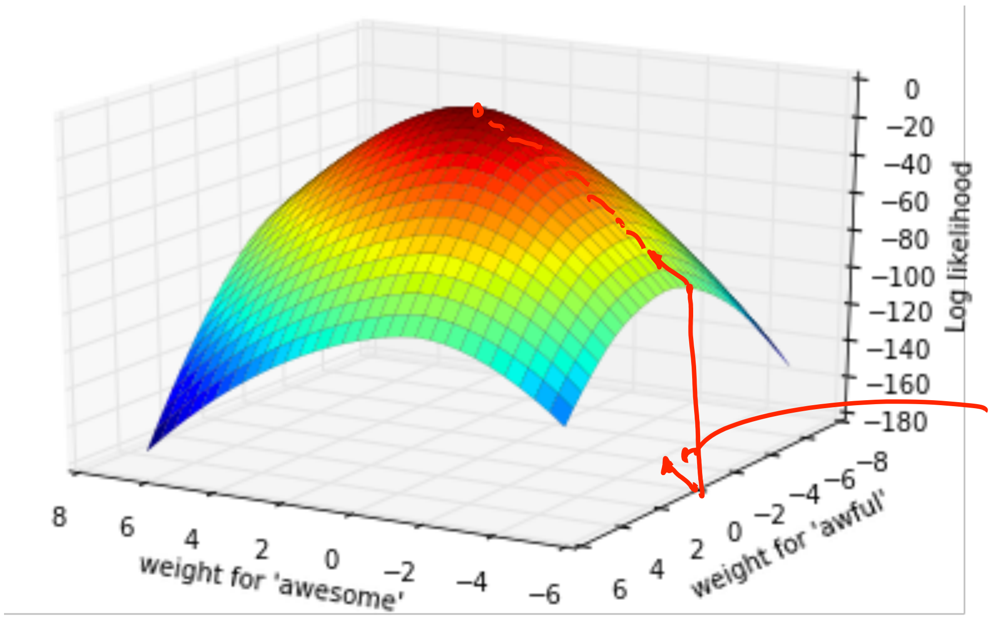
$$\left.\frac{d\ell}{dw}\right|_{w^{(t)}} < \varepsilon$$

tolerance

Algorithm:

while not converged
$$w^{(t+1)} \leftarrow w^{(t)} + \eta \left.\frac{d\ell}{dw}\right|_{w^{(t)}}$$

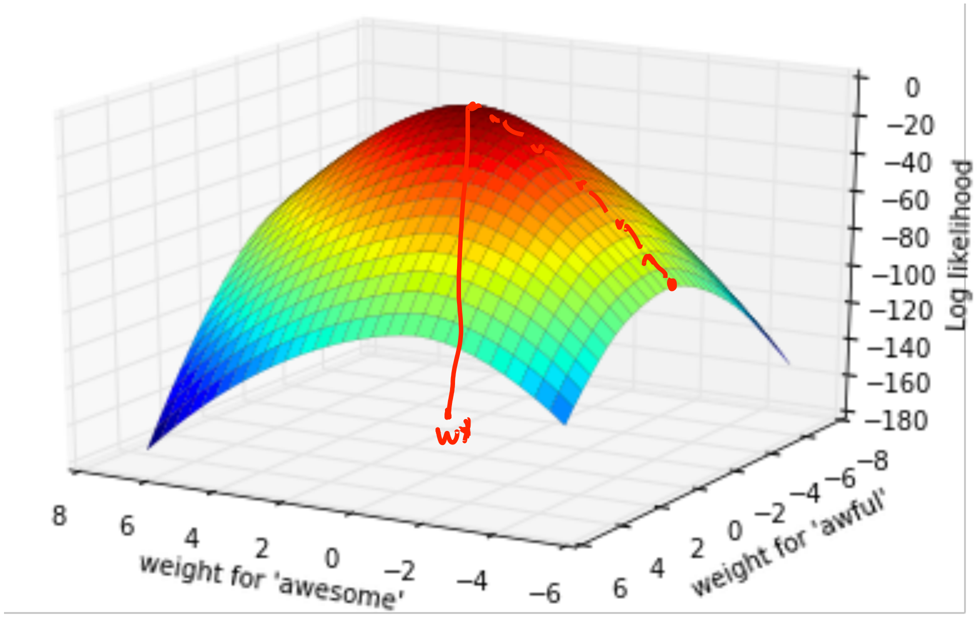
Moving to multiple dimensions: Gradients

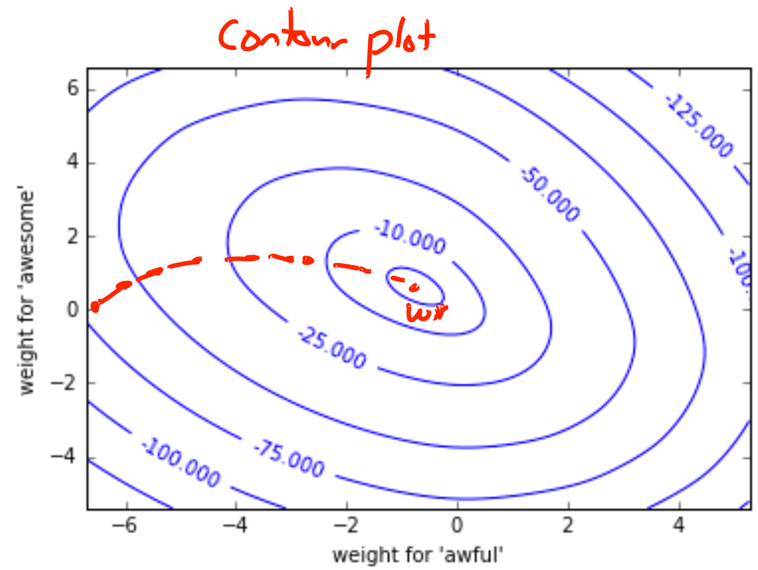


$$\nabla \ell(\mathbf{w}) = \begin{bmatrix} \frac{\partial \ell}{\partial w_0} \\ \frac{\partial \ell}{\partial w_1} \\ \vdots \\ \frac{\partial \ell}{\partial w_D} \end{bmatrix}$$

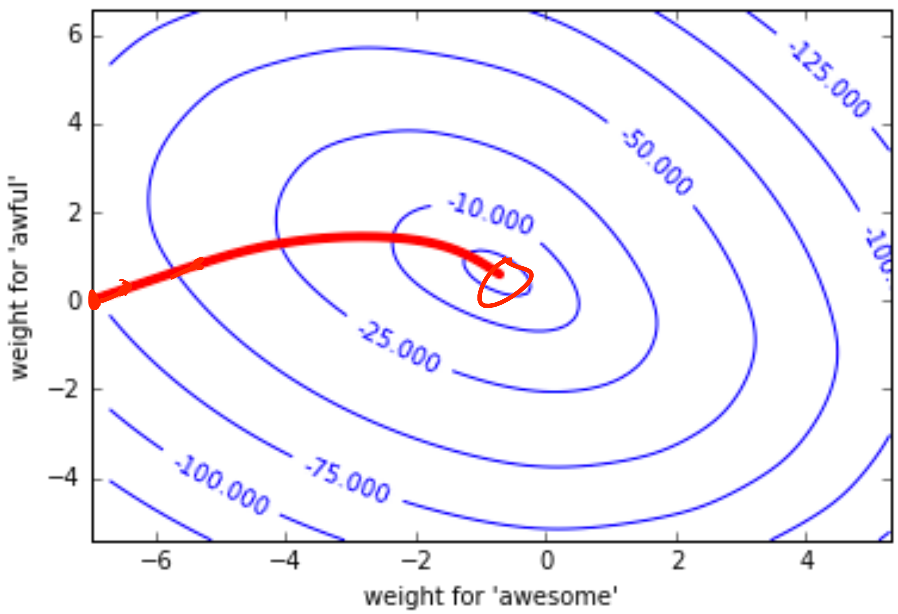
D+1 dim vector

Contour plots





Gradient ascent



Algorithm:

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

while not converged

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

step size

MOVE TO FULL BODY SHOT

Learning algorithm for logistic regression

MOVE TO HEAD SHOT

Derivative of (log-)likelihood

Sum over data points

Feature value

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

j ranges from 0 ~ D, num of features
i ranges from 1~N, num of data points

- output = 1 if yi positive
- output = 0 if yi negative

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

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

$w^{(t)}$:

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

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

$h_1(x)$ = # awesome parameter w1 multiplies 1st feature h1(x)

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

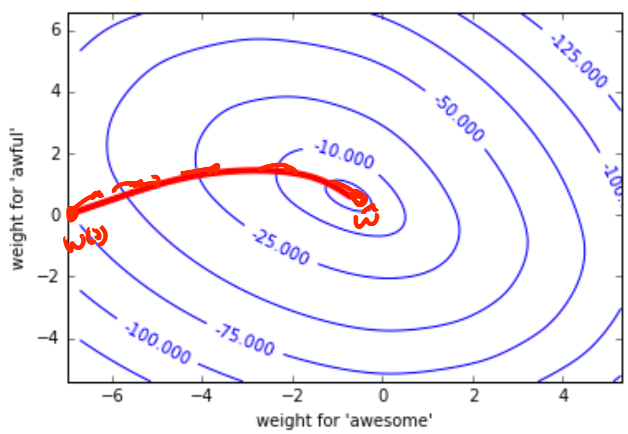
Difference between truth and prediction

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

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

increase parameter w
=> increase score
=> increase probability

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

for j=0,...,D ← coefficient

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

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

t ← t + 1

step size

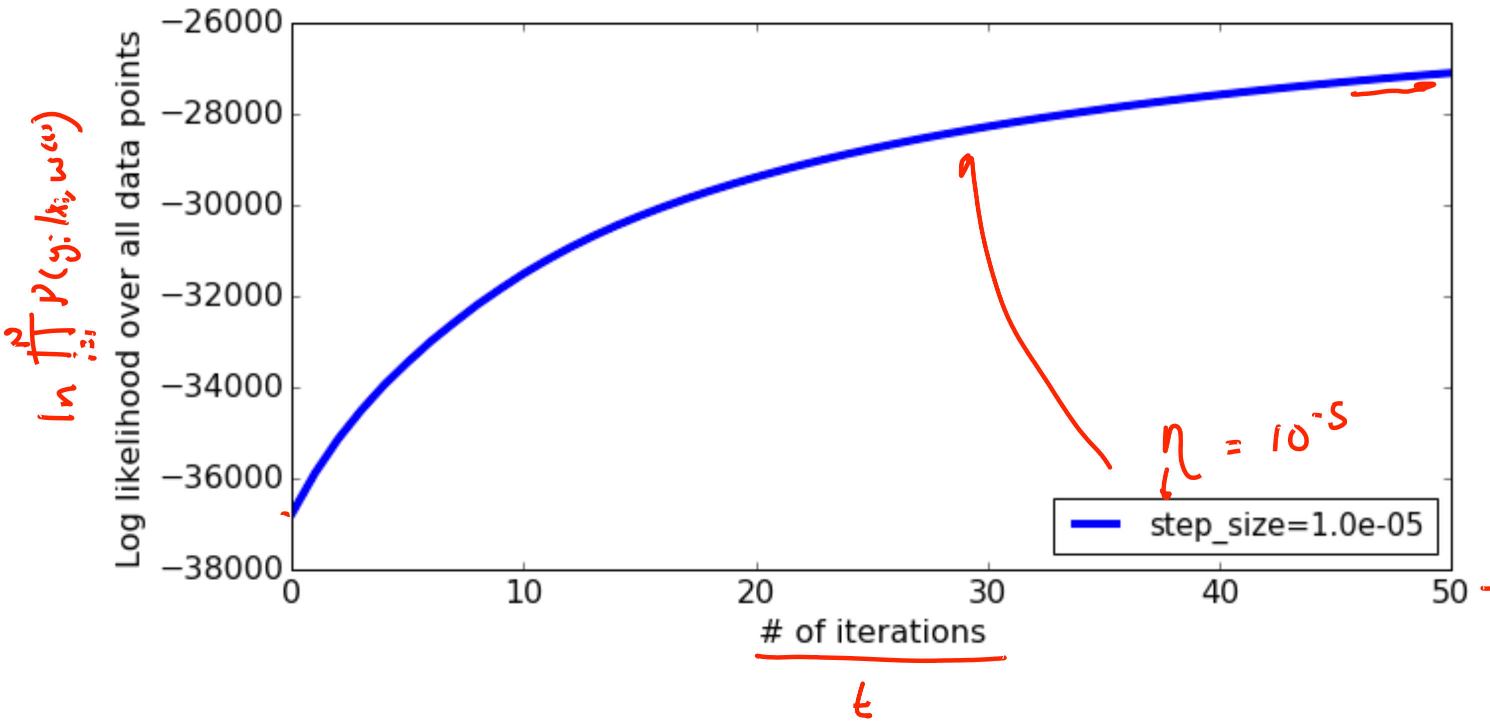
$\frac{\partial \ell(\mathbf{w}^{(t)})}{\partial w_j}$

MOVE TO FULL BODY SHOT

Choosing the step size η

MOVE TO HEAD SHOT

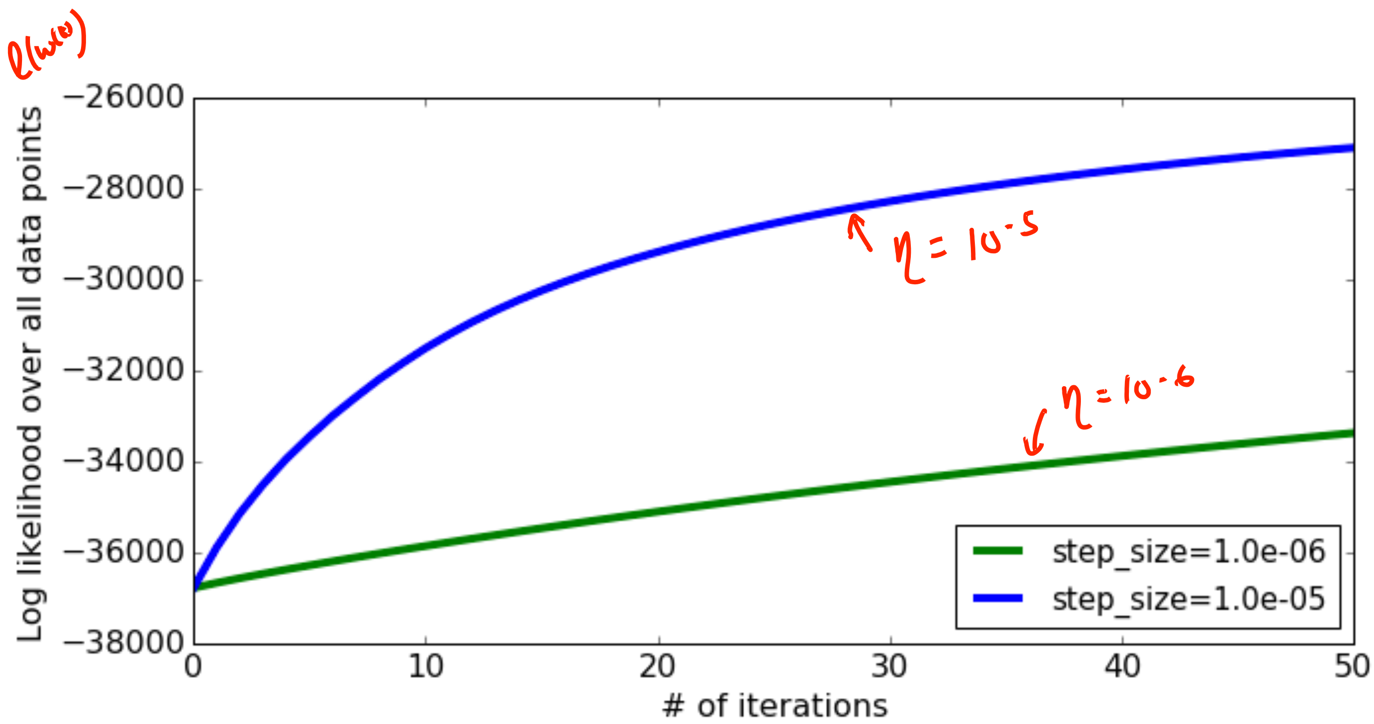
Learning curve:
Plot quality (likelihood) over iterations

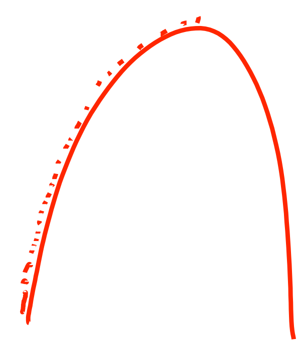


→ need more than 50 iterations

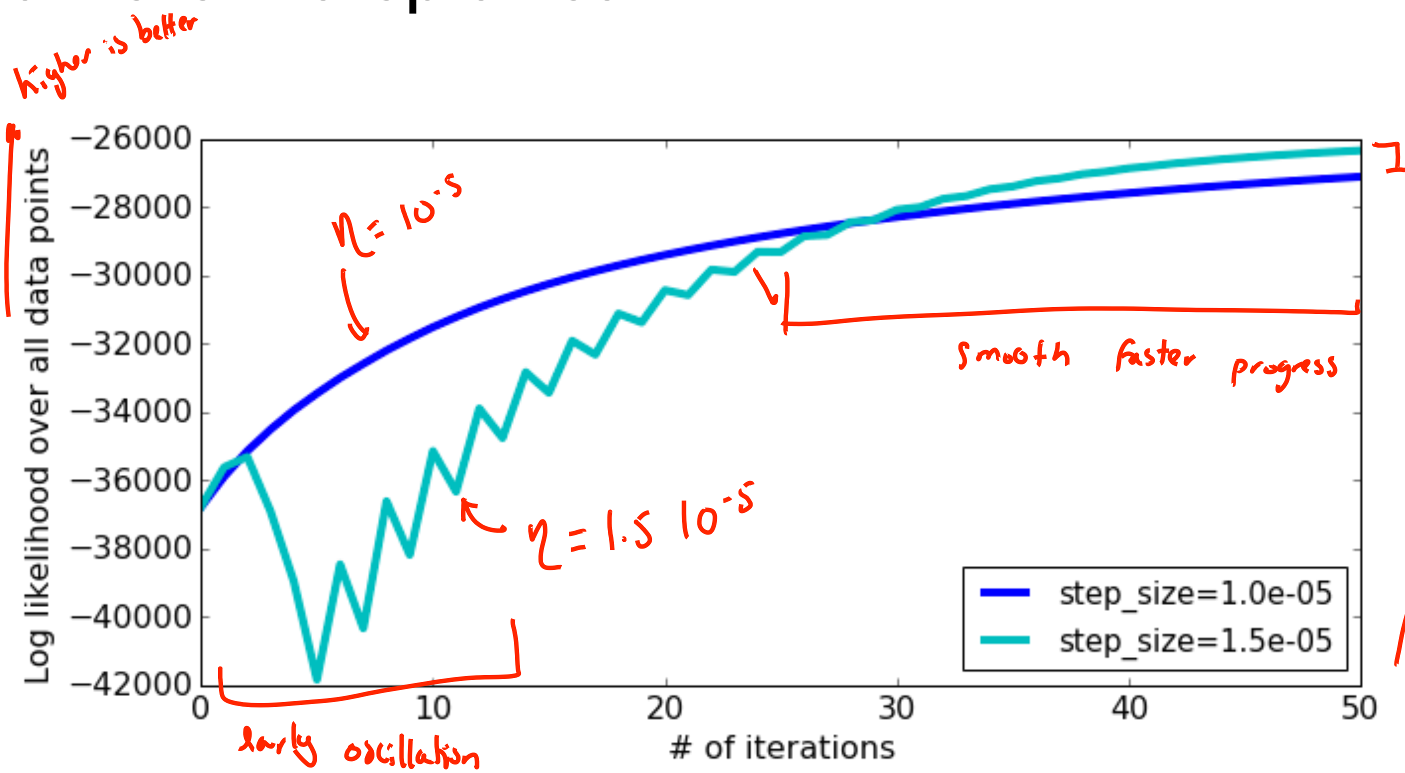
because the curve is still going up

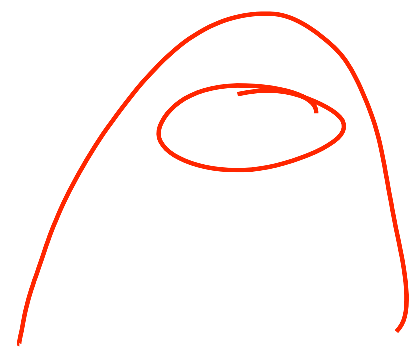
If step size is too small, can take a long time to converge



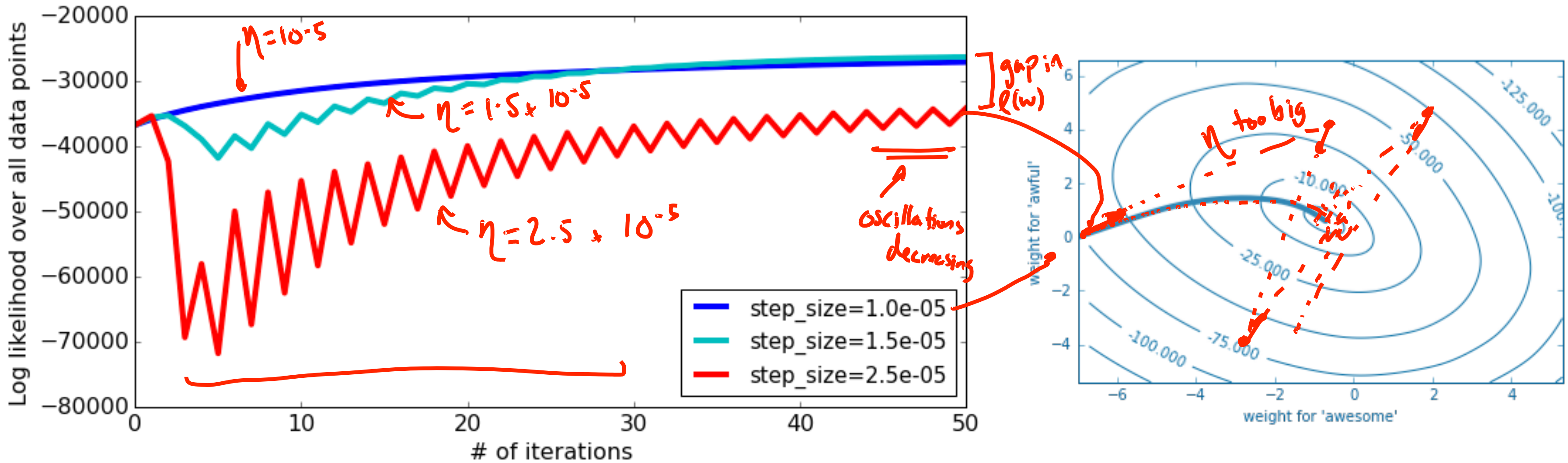


Compare converge with different step sizes

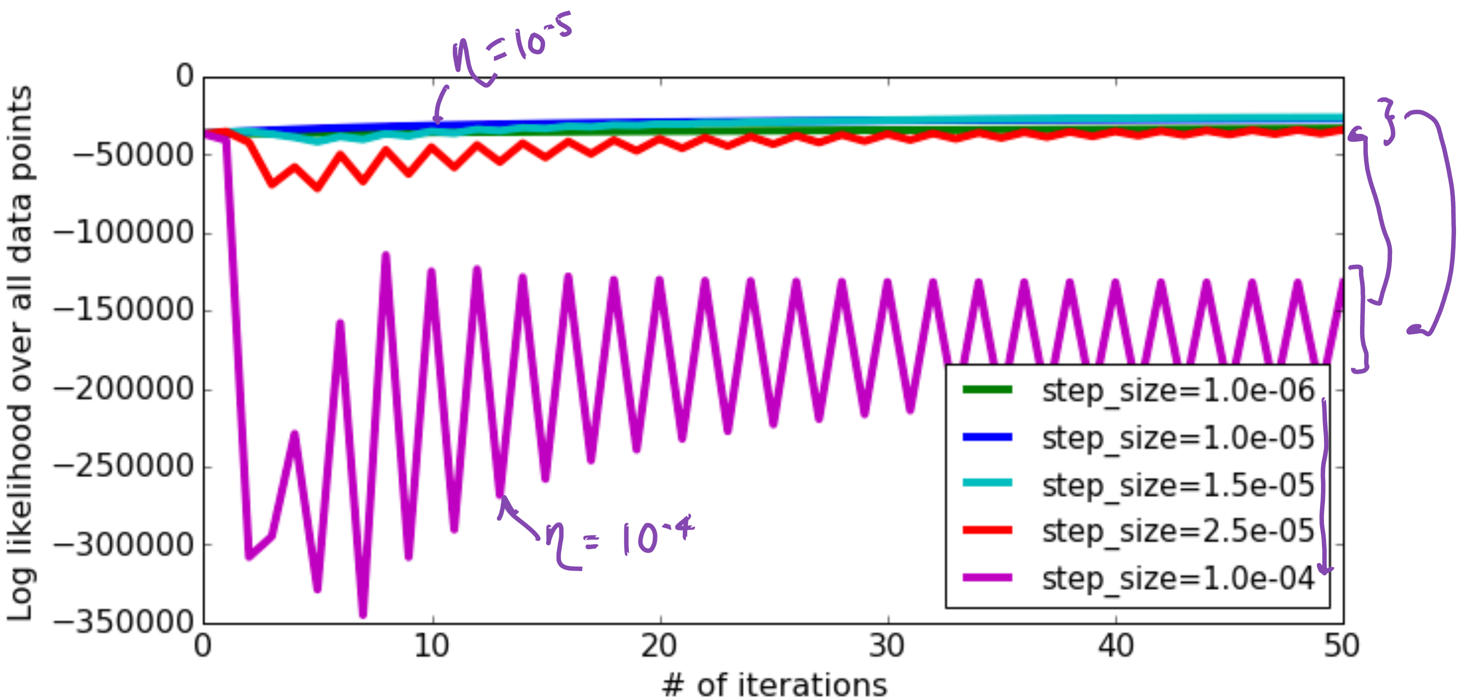




Careful with step sizes that are too large



Very large step sizes can even cause divergence or wild oscillations



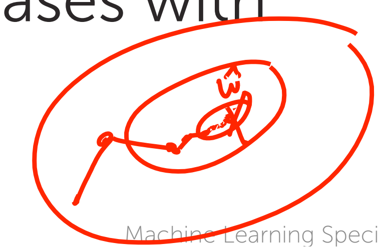
Simple rule of thumb for picking step size η

- 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.,

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



MOVE TO FULL BODY SHOT

Deriving the gradient for logistic regression

VERY OPTIONAL

MOVE TO HEAD SHOT

Log-likelihood function

- Goal: choose coefficients **w** maximizing likelihood:

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

- Math simplified by using log-likelihood – taking (natural) log:

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

natural log

The log trick, often used in ML…

- Products become sums:

$\ln a \cdot b = \ln a + \ln b$ | $\ln \frac{a}{b} = \ln a - \ln b$

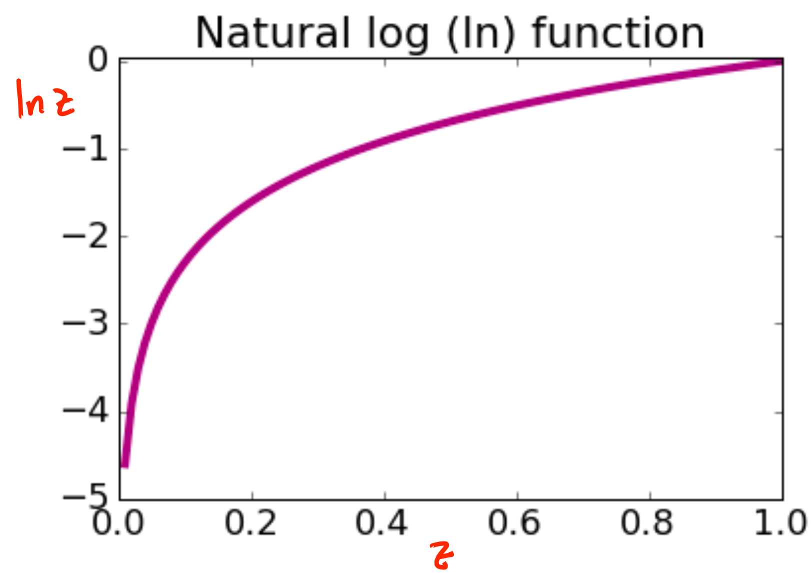
- Doesn't change maximum!
  - If $\hat{\mathbf{w}}$ maximizes f($\mathbf{w}$):

$$\hat{w} = \underbrace{\arg\max_{w}}_{\text{the } w \text{ that makes } f(w) \text{ largest}} f(w)$$

  - Then $\hat{\mathbf{w}}_{\ln}$ maximizes ln(f($\mathbf{w}$)):

$$\hat{w}_{\ln} = \arg\max_{w} \ln(f(w))$$

$$\hat{w} = \hat{w}_{\ln}$$



Insert next title slide before Slide 52, around 4:55 in PL7_DerivingtheGradient_1stEdit

Expressing the log-likelihood

VERY OPTIONAL

Using log to turn products into sums

$$\ln \prod_{i=1}^{N} f_i = \sum_{i=1}^{N} \ln f_i$$

- The log of the product of likelihoods becomes the sum of the logs:

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

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

Rewriting log-likelihood

- For simpler math, we'll rewrite likelihood with indicators:

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

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

if $y_i = +1$ ✓ 0

if $y_i = -1$ 0 ✓

Insert next title slide before Slide 54, around 7:33 in PL7_DerivingtheGradient_1stEdit

Deriving probability that y=-1 given x

VERY OPTIONAL

Logistic regression model: $P(y=-1|\mathbf{x},\mathbf{w})$

- Probability model predicts y=+1:

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

- Probability model predicts y=-1:

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

$$= \frac{1+e^{-w^T h(x)} - 1}{1+e^{-w^T h(x)}} = \frac{e^{-w^T h(x)}}{1+e^{-w^T h(x)}}$$

Insert next title slide before Slide 55, around 9:15 in PL7_DerivingtheGradient_1stEdit

Rewriting the log-likelihood

Plugging in logistic function for 1 data point

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

$$\ell\ell(\mathbf{w}) = \mathbb{1}[y_i=+1]\ln P(y=+1 \mid \mathbf{x}_i,\mathbf{w}) + \mathbb{1}[y_i=-1]\ln P(y=-1 \mid \mathbf{x}_i,\mathbf{w})$$

$$= \mathbb{1}[y_i=+1]\ln\frac{1}{1+e^{-w^T h(x_i)}} + (1-\mathbb{1}[y_i=+1])\ln\frac{e^{-w^T h(x_i)}}{1+e^{-w^T h(x_i)}}$$

$$= -\mathbb{1}[y_i=+1]\ln(1+e^{-w^T h(x_i)}) + (1-\mathbb{1}[y_i=+1])[-w^T h(x_i) - \ln(1+e^{-w^T h(x_i)})]$$

$$\underbrace{= -(1-\mathbb{1}[y_i=+1])\,w^T h(x_i) - \ln(1+e^{-w^T h(x_i)})}_{\text{Simpler form}}$$

$$\ln e^a = a$$

$$\mathbb{1}[y_i=-1] = 1-\mathbb{1}[y_i=+1]$$

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

$$\ln\frac{e^{-w^T h(x_i)}}{1+e^{-w^T h(x_i)}} = \underbrace{\ln e^{-w^T h(x_i)}}_{-w^T h(x_i)} - \ln(1+e^{-w^T h(x_i)})$$

Insert next title slide before Slide 56, around 16:56 in PL7_DerivingtheGradient_1stEdit

Deriving gradient of log-likelihood

VERY OPTIONAL

Gradient for 1 data point

$$\ell\ell(\mathbf{w}) = -(1 - \mathbb{1}[y_i = +1])\mathbf{w}^\top h(\mathbf{x}_i) - \ln\left(1 + e^{-\mathbf{w}^\top h(\mathbf{x}_i)}\right)$$

$$\frac{\partial \ell\ell}{\partial w_j} = -(1 - \mathbb{1}[y_i = +1]) \frac{\partial}{\partial w_j} w^\top h(x_i) - \frac{\partial}{\partial w_j} \ln\left(1 + e^{-w^\top h(x_i)}\right)$$

$$= -(1 - \mathbb{1}[y_i = +1]) h_j(x_i) + h_j(x_i) P(y=-1 \mid x_i, w)$$

$$= h_j(x_i)\left[\mathbb{1}[y_i = +1] - P(y=+1 \mid x_i, w)\right]$$

$$\frac{\partial}{\partial w_j} w^\top h(x_i) = h_j(x_i)$$

$$\frac{\partial}{\partial w_j} \ln\left(1 + e^{-w^\top h(x_i)}\right)$$

$$= -h_j(x_i) \underbrace{\frac{e^{-w^\top h(x_i)}}{1 + e^{-w^\top h(x_i)}}}_{P(y=-1 \mid x_i, w)}$$

Finally, gradient for all data points

- Gradient for one data point:

$$h_j(\mathbf{x}_i)\Big(\mathbb{1}[y_i = +1] - P(y = +1 \mid \mathbf{x}_i, \mathbf{w})\Big)$$

- Adding over data points:

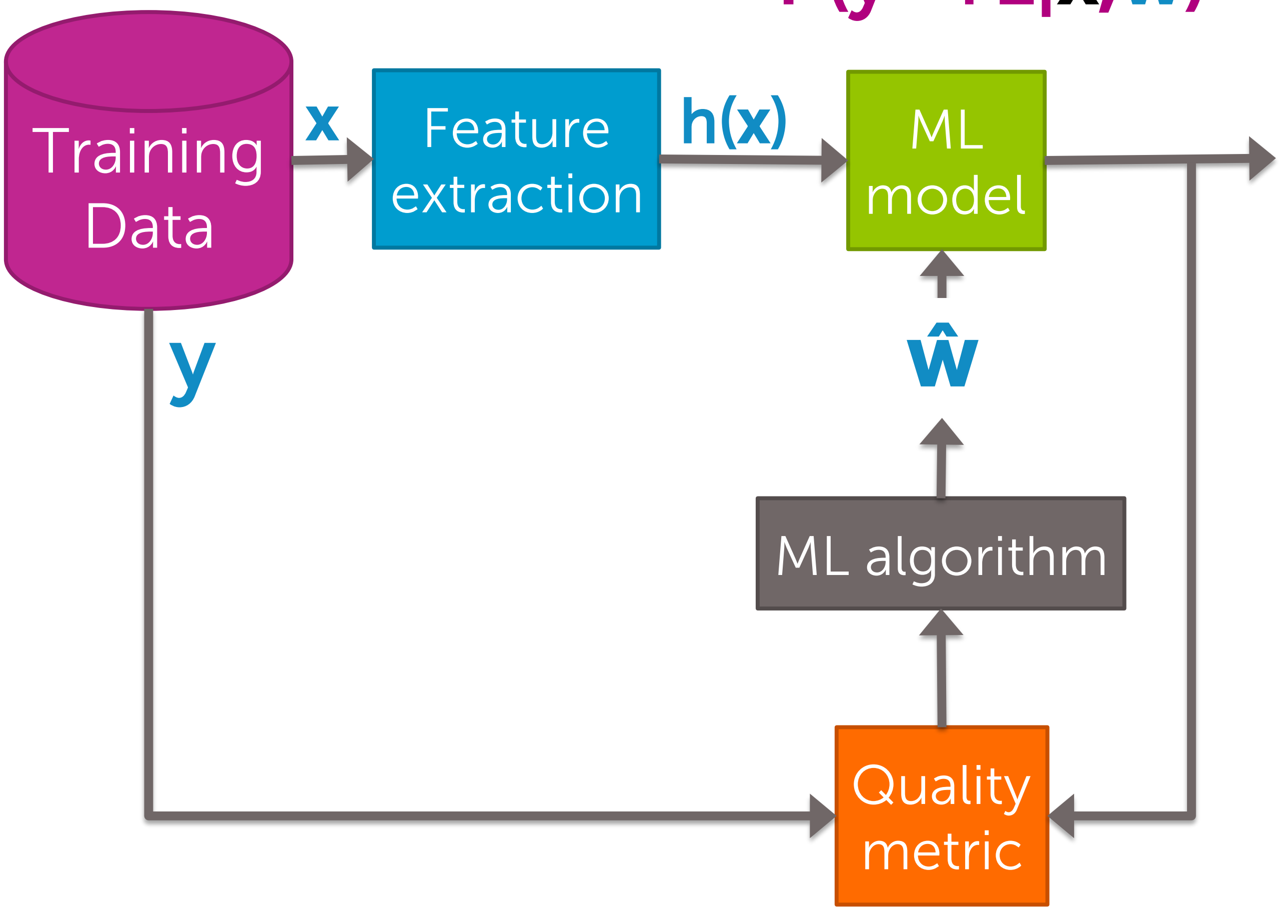
$$\frac{\partial \ell\ell}{\partial w_j} = \sum_{i=1}^{N} h_j(x_i)\Big(\mathbb{1}[y_i = +1] - \underline{P(y = +1 \mid x_i, w)}\Big)$$

MOVE TO FULL BODY SHOT

Summary of logistic regression classifier

MOVE TO HEAD SHOT

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

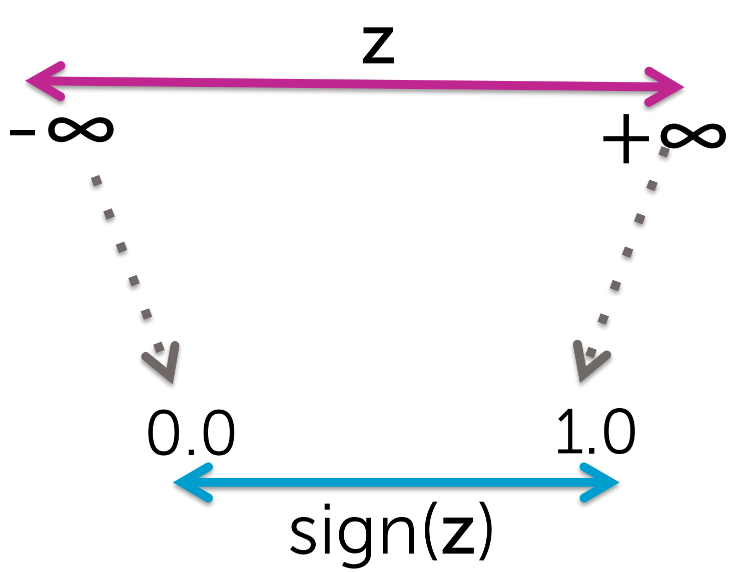


MOVE TO FULL BODY SHOT

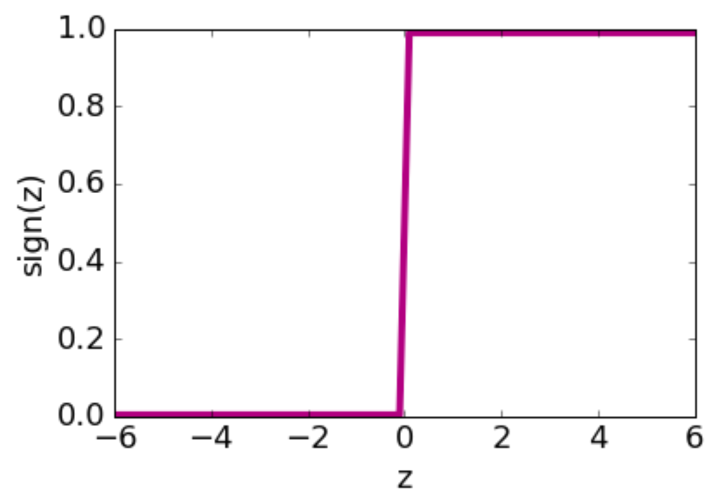
What you can do now…

- 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
- (Optional) Derive the gradient descent update rule for logistic regression

Simplest link function: sign(z)



$$sign(z) = \begin{cases} +1 & \text{if } z \geq 0 \\ -1 & \text{otherwise} \end{cases}$$



But, sign(z) only outputs -1 or +1, no probabilities in between

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

$$0.0 \longleftarrow P(y=+1|\mathbf{x}_i, \hat{\mathbf{w}}) \longrightarrow 1.0$$

$$-\infty \longleftarrow \text{Score}(\mathbf{x}_i) = \hat{\mathbf{w}}^T h(\mathbf{x}_i) \longrightarrow +\infty$$

Quality metric: probability of data

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

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

If model good, should predict

Increase probability y=+1 when

Choose **w** to make

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

If model good, should predict

Increase probability y=-1 when

Choose **w** to make

Maximizing likelihood (probability of data)

Data point	$\mathbf{x}$[1]	$\mathbf{x}$[2]	y	Choose **w** to maximize
$\mathbf{x}_1,y_1$	2	1	+1	
$\mathbf{x}_2,y_2$	0	2	-1	
$\mathbf{x}_3,y_3$	3	3	-1	
$\mathbf{x}_4,y_4$	4	1	+1	
$\mathbf{x}_5,y_5$	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

Learn logistic regression model with maximum likelihood estimation (MLE)

- Choose coefficients **w** that maximize likelihood:

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

- No closed-form solution ➔ use gradient ascent