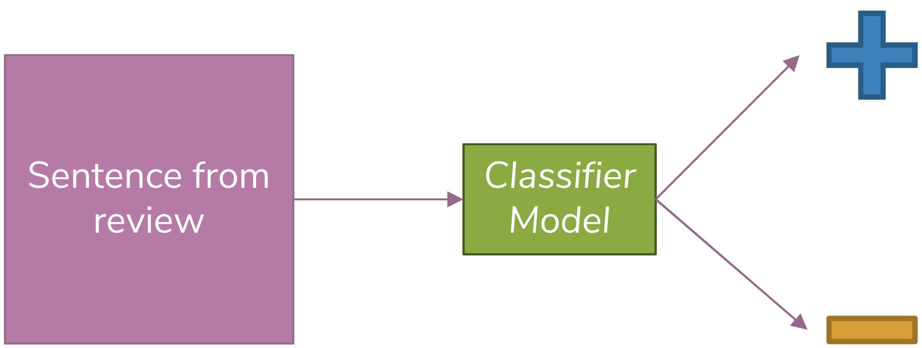
CSE/STAT 416

Logistic Regression

Hunter Schafer
University of Washington
July 10, 2019

Sentiment Classifier

In our example, we want to classify a restaurant review as positive or negative.



Input: x

Output: y
Predicted class

Implementation 2: Linear Classifier

Idea: Use labelled training data to learn a weight for each word. Use weights to score a sentence.

- See last slide for example weights and scoring.

Linear Classifier

Input x: Sentence from review

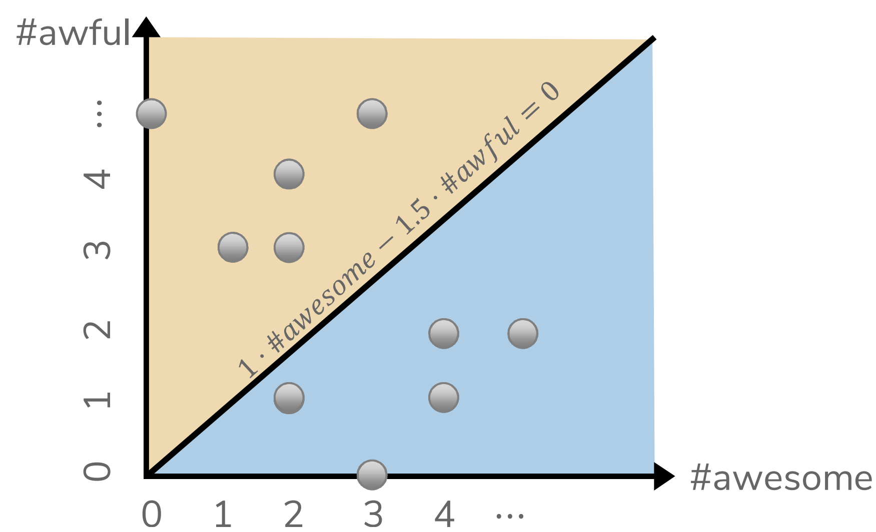
- Compute $Score(x)$
- If $Score(x) > 0$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$

Decision Boundary

Consider if only two words had non-zero coefficients

Word	Coefficient	Weight
	w_0	0.0
awesome	w_1	1.0
awful	w_2	-1.5

$$\hat{s} = 1 \cdot \#awesome - 1.5 \cdot \#awful$$



Learning $\hat{w}$

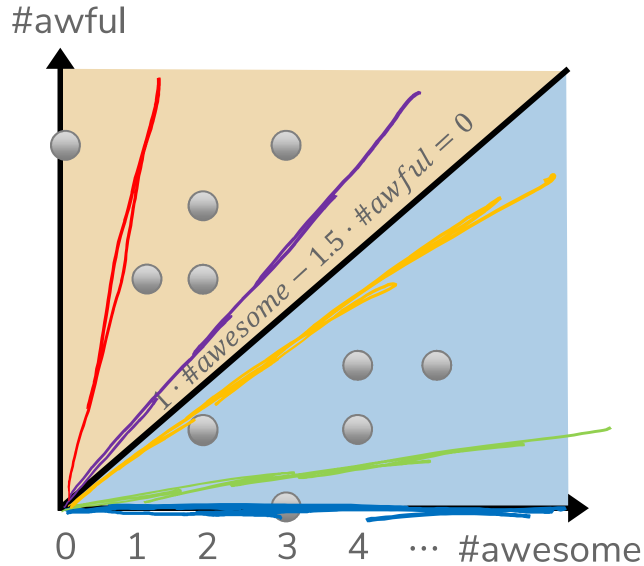
All the Same?

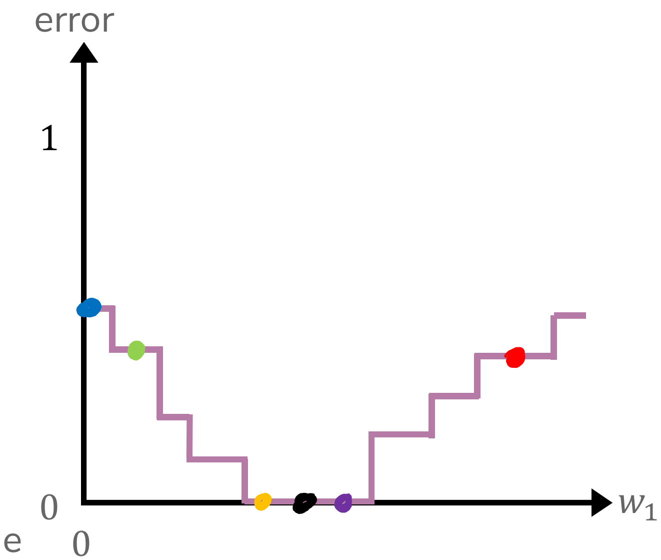
One idea is to just model the processing of finding $\hat{w}$ based on what we discussed in linear regression

$$\hat{w} = \min_{w} \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}\{y_i \neq \hat{y}_i\}$$

Will this work?

Assume $h_1(x) = \#awesome$ so w_1 is its coefficient and w_2 is fixed.





Minimizing Error

Minimizing classification error is totally the most intuitive thing to do given all we have learned from regression. However, it just doesn't work in this case with classification.

We aren't able to use a method like gradient descent here because the function isn't "nice" (it's not continuous, it's not differentiable, etc.).

We will use a stand-in for classification error that will allow us to use an optimization algorithm. But first, we have to change the problem we care about a bit.

Instead of caring about the classifications, let's look at some probabilities

$P(y=+1|x) + P(y=-1|x) = 1$

Probabilities

$$P(y|x) = \begin{cases} P(y=+1|x) & \text{if } y=+1 \\ P(y=-1|x) & \text{otherwise} \end{cases}$$

Assume that there is some randomness in the world, and instead will try to model the probability of a positive/negative label.

Examples:

"The sushi & everything else were awesome!"

- Definite positive (+1)
- $P\left(y = +1 \mid x = \text{“}The\ sushi\ \&\ everything\ else\ were\ awesome!\text{”}\right) = 0.99$

"The sushi was alright, the service was OK"

- Not as sure
- $P\left(y = -1 \mid x = \text{“}The\ sushi\ alright, the\ service\ was\ okay!\text{”}\right) = 0.5$

Use probability as the measurement of certainty

$$P(y|x)$$

Probability Classifier

Idea: Estimate probabilities $\hat{P}(y|x)$ and use those for prediction

Probability Classifier

Input x: Sentence from review

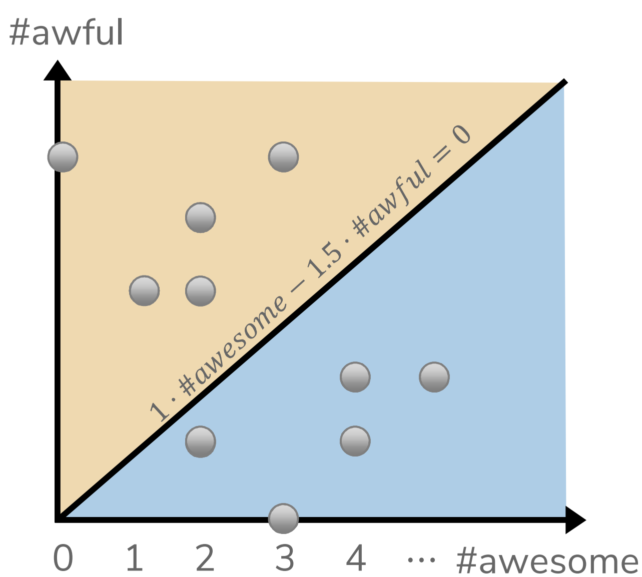
- Estimate class probability $\hat{P}(y = +1|x)$
- If $\hat{P}(y = +1|x) > 0.5$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$

Notes:

- Estimating the probability improves **interpretability**

Score Probabilities?

Idea: Let's try to relate the value of $Score(x)$ to $\hat{P}(y = +1|x)$



What if $Score(x)$ is positive?

$P(y=+1|x) > 1/2$

What if $Score(x)$ is negative?

$P(y=-1|x) > 1/2$

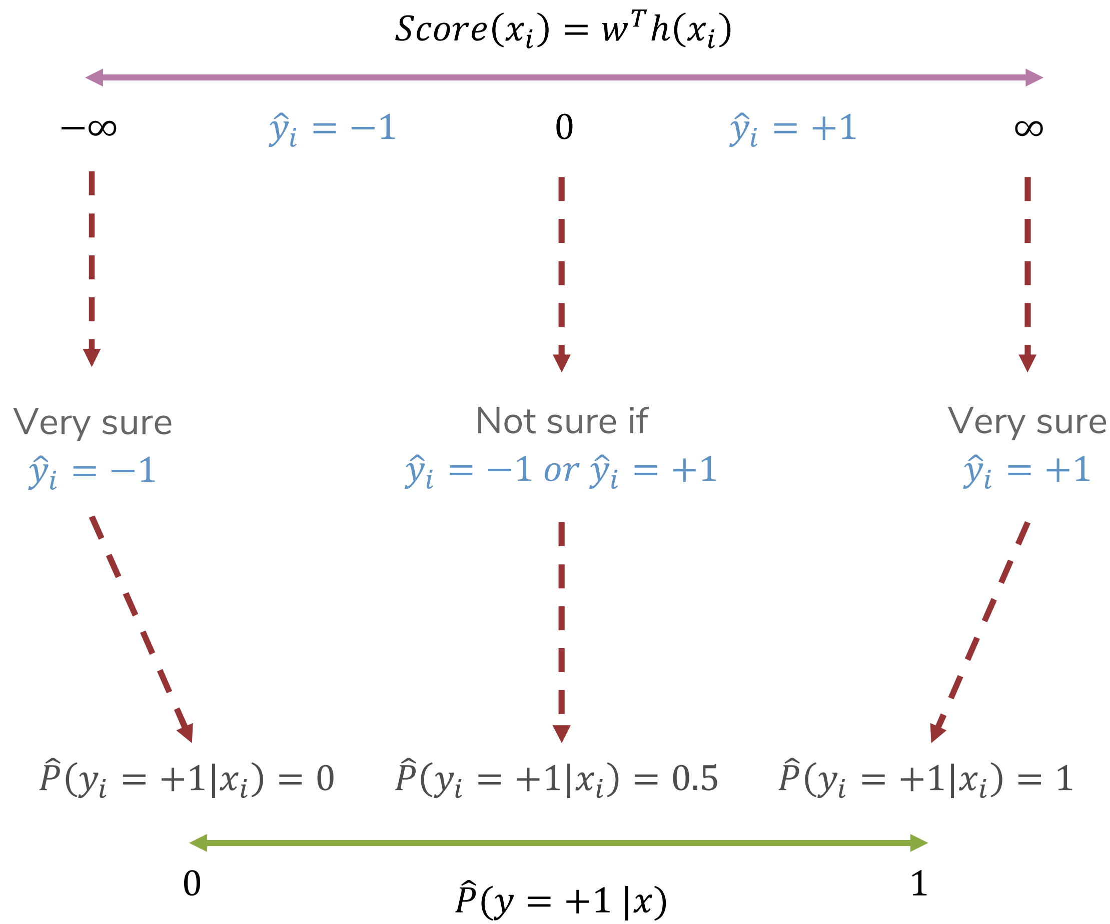
equiv.

$P(y=+1|x) < 1/2$

What if $Score(x)$ is 0?

$P(y=+1|x) = 1/2$

Interpreting Score

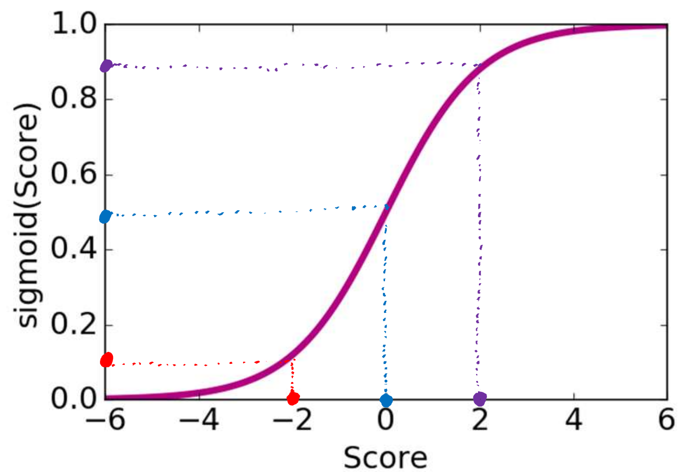


Logistic Function

Use a function that takes numbers arbitrarily large/small and maps them between 0 and 1.

$$sigmoid(Score(x)) = \frac{1}{1 + e^{-Score(x)}}$$

$Score(x)$	$sigmoid(Score(x))$
$-\infty$	$\frac{1}{1+e^{--\infty}} = \frac{1}{1+e^{\infty}} = \frac{1}{\infty} = 0$
-2	0.12
0	$\frac{1}{1+e^{0}} = \frac{1}{1+1} = \frac{1}{2}$
2	0.88
∞	$\frac{1}{1+e^{-\infty}} = \frac{1}{1+0} = 1$



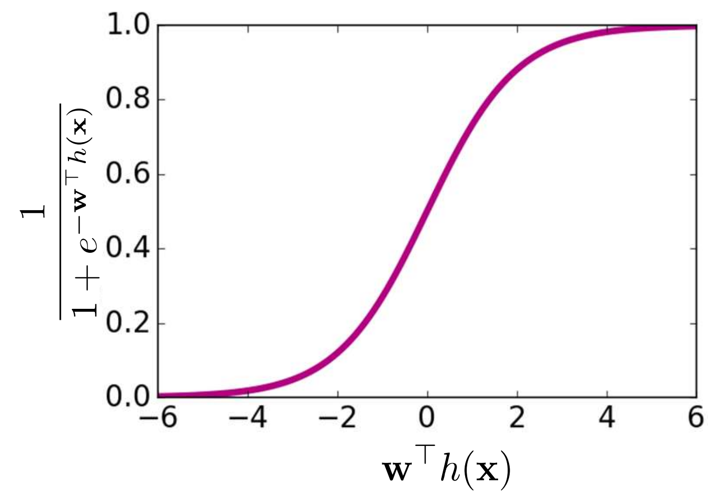
Logistic Regression Model

$$P(y_i = +1|x_i, w) = sigmoid\big(Score(x_i)\big) = \frac{1}{1 + e^{-w^T h(x_i)}}$$

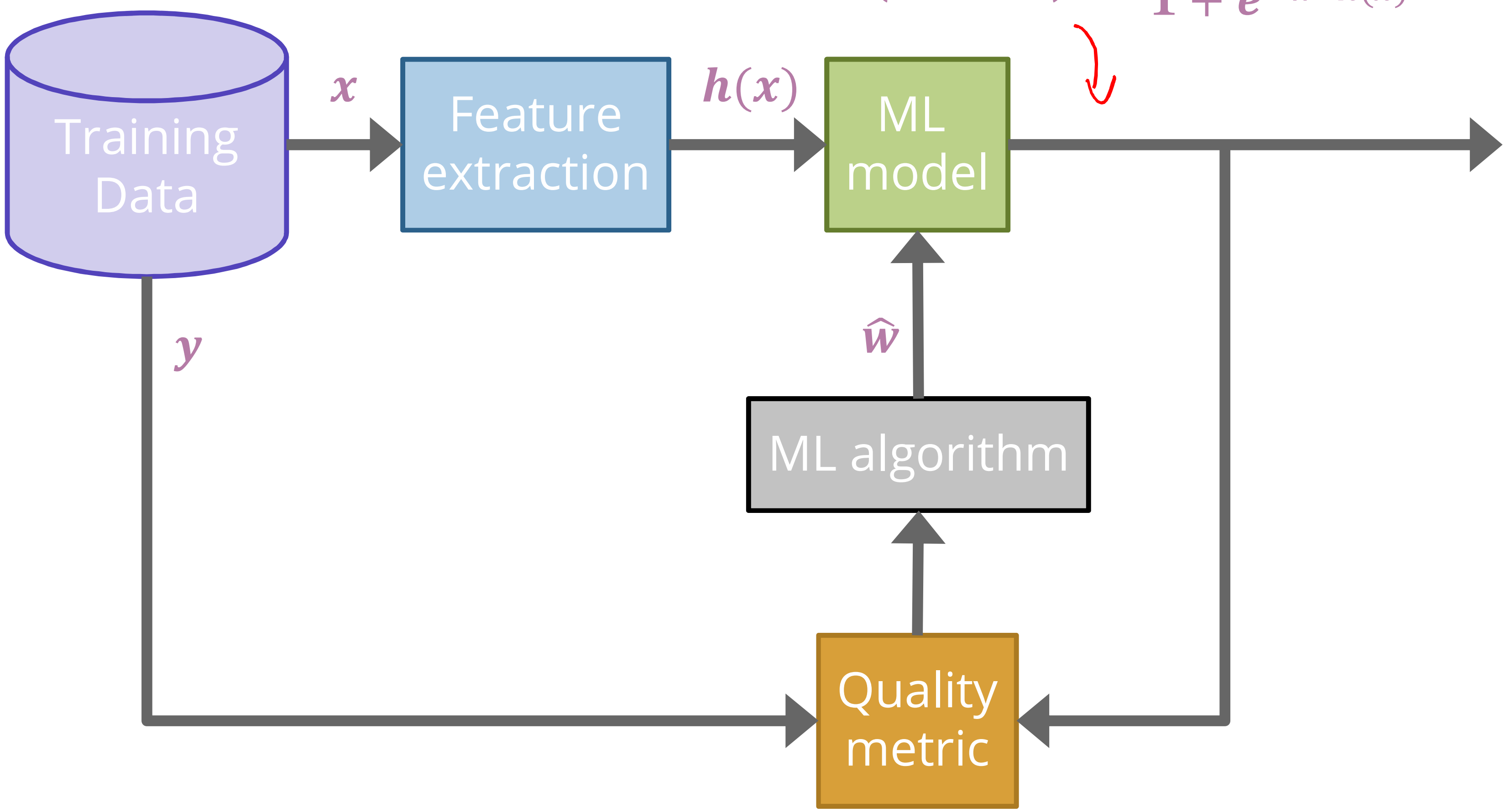
Logistic Regression Classifier

Input x: Sentence from review

- Estimate class probability $\hat{P}(y = +1|x, \hat{w}) = sigmoid(\underbrace{\hat{w}^T h(x_i)}_{Score(x_i)})$
- If $\hat{P}(y = +1|x, \hat{w}) > 0.5$:
  - $\hat{y} = +1$
- Else:
  - $\hat{y} = -1$



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



Brain Break

10:19

Quality Metric = Likelihood

$P(y_i \mid x_i, w) = \begin{cases} P(y_i = +1 \mid x_i, w) & \text{if } y_i = +1 \\ P(y_i = -1 \mid x_i, w) & \text{otherwise} \end{cases}$

$P(y_i = -1 \mid x_i, w) = 1 - P(y_i = +1 \mid x_i, w)$

$P(y_i = +1 \mid x_i, w) = \frac{1}{1 + e^{-w^T h(x_i)}}$ $\quad P(y_i = -1 \mid x_i, w) = \frac{e^{-w^T h(x_i)}}{1 + e^{-w^T h(x_i)}}$

Want to compute the probability of seeing our dataset for every possible setting for w. Find w that makes data most likely!

Data Point	$h_1(x)$	$h_2(x)$	y	Choose w to maximize
x_1, y_1	2	1	$+1$	$P(y_1 = +1 \mid x_1, w)$
x_2, y_2	0	2	-1	$P(y_2 = -1 \mid x_2, w)$
x_3, y_3	3	3	-1	$P(y_3 = -1 \mid x_3, w)$
x_4, y_4	4	1	$+1$	$P(y_4 = +1 \mid x_4, w)$

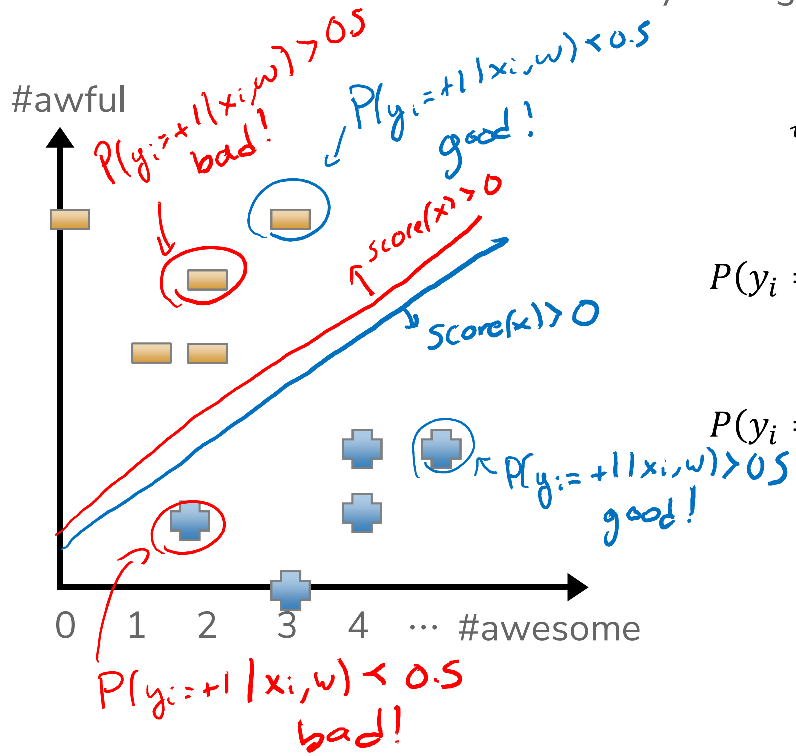
$\ell(w) = P(y_1 \mid x_1, w) P(y_2 \mid x_2, w) P(y_3 \mid x_3, w) P(y_4 \mid x_4, w)$

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

Learn $\hat{w}$

Now that we have our new model, we will talk about how to choose $\hat{w}$ to be the "best fit".

- The choice of w affects how likely seeing our dataset is



$$\ell(w) = \prod_{i}^{n} P(y_i|x_i, w)$$

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

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

Maximum Likelihood Estimate (MLE)

Find the w that maximizes the likelihood

$$\hat{w} = \max_w \ell(w) = \max_w \prod_{i=1}^{n} P(y_i | x_i, w)$$

Generally we maximize the log-likelihood which looks like

$$\hat{w} = \max_w \underbrace{\sum_{i=1:y_i=+1}^{n} \ln\left(\frac{1}{1+e^{-w^T h(x)}}\right)}_{\text{Positive examples}} + \underbrace{\sum_{i=1:y_i=-1}^{n} \ln\left(\frac{e^{-w^T h(x)}}{1+e^{-w^T h(x)}}\right)}_{\text{Negative examples}}$$

Penalizes small sigmoid$(w^T h(x))$

Penalizes large sigmoid$(w^T h(x))$

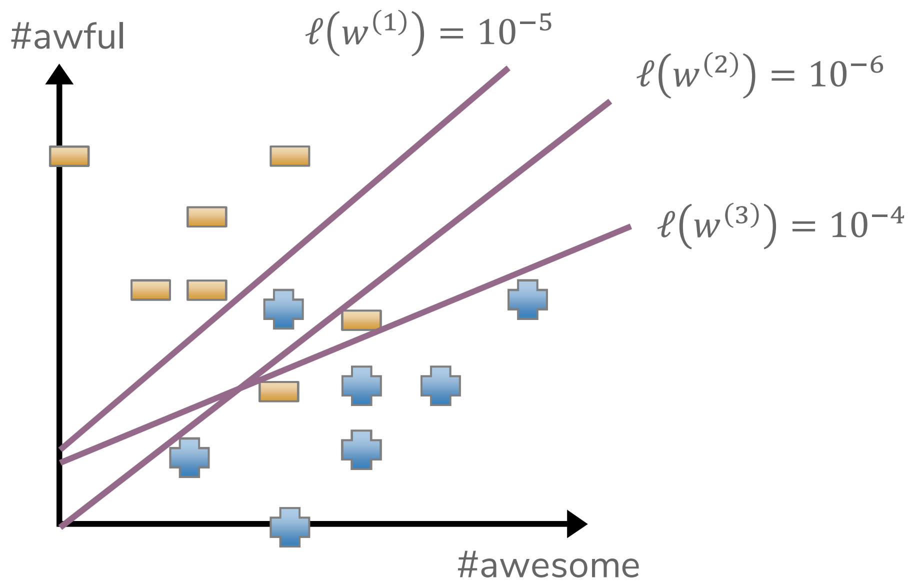
Poll Everywhere

Think

1 min

pollev.com/cse416

Which setting of w should we use?



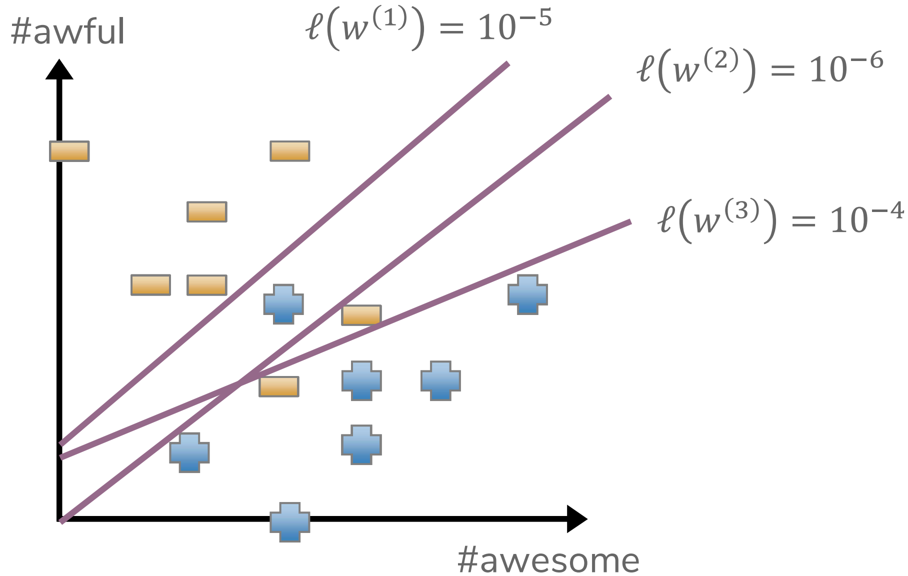
Poll Everywhere

Pair

2 min

pollev.com/cse416

Which setting of w should we use?

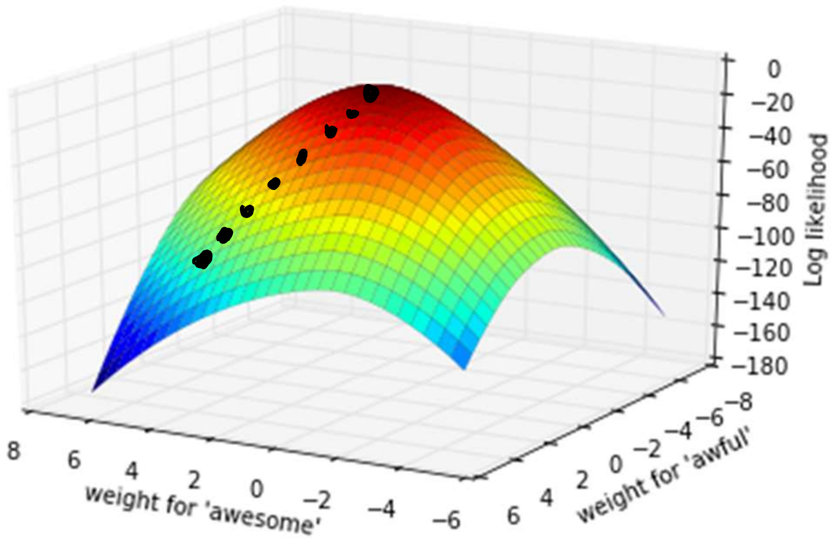


2:00

Finding MLE

No closed-form solution, have to use an iterative method like gradient **ascent**!

$$\hat{w} = \max_{w} \prod_{i=1}^{n} P(y_i | x_i, w)$$



Gradient Ascent

Gradient ascent is the same as gradient descent, but we go ”up the hill”.

start at some (random) point $w^{(0)}$ when $t = 0$

while we haven't converged

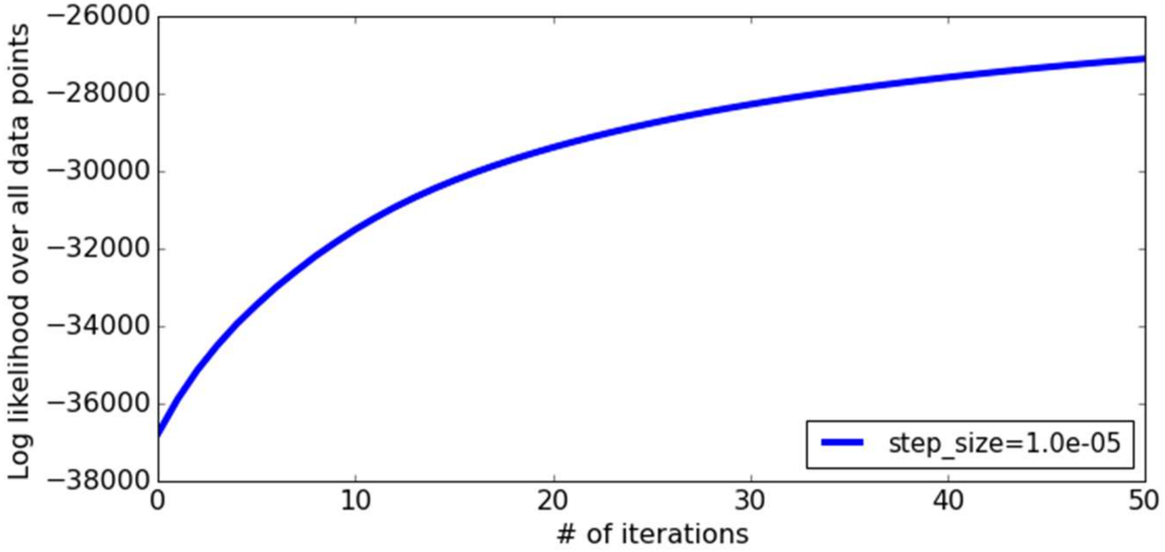
$$w^{(t+1)} \leftarrow w^{(t)} + \eta \nabla \ell(w^{(t)})$$

$$t \leftarrow t + 1$$

This is just describing going up the hill step by step.

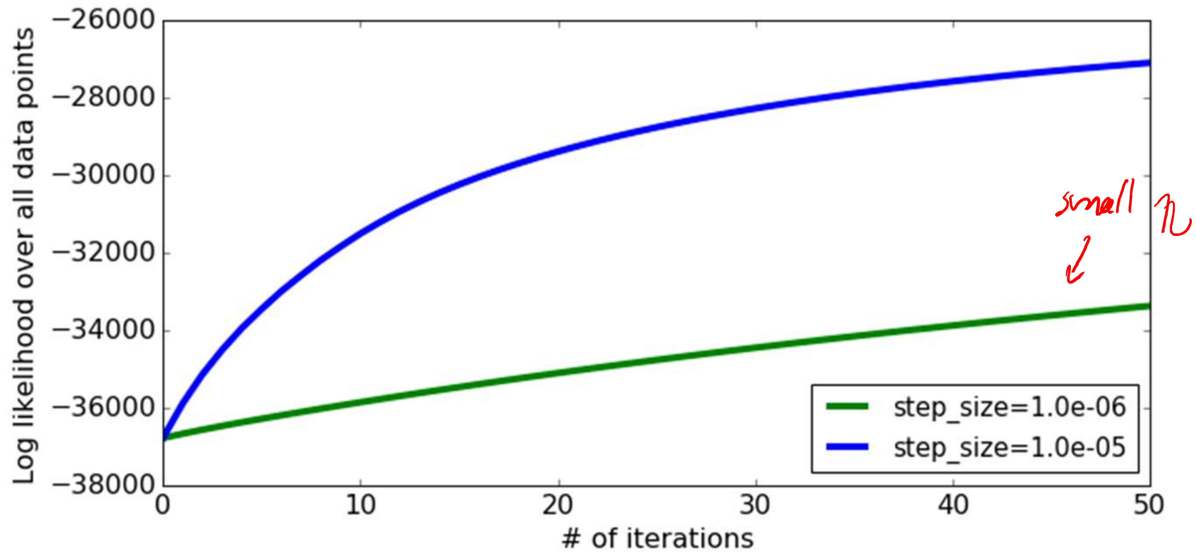
η controls how big of steps we take, and picking it is crucial for how well the model you learn does!

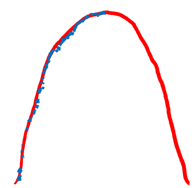
Learning Curve



Choosing η

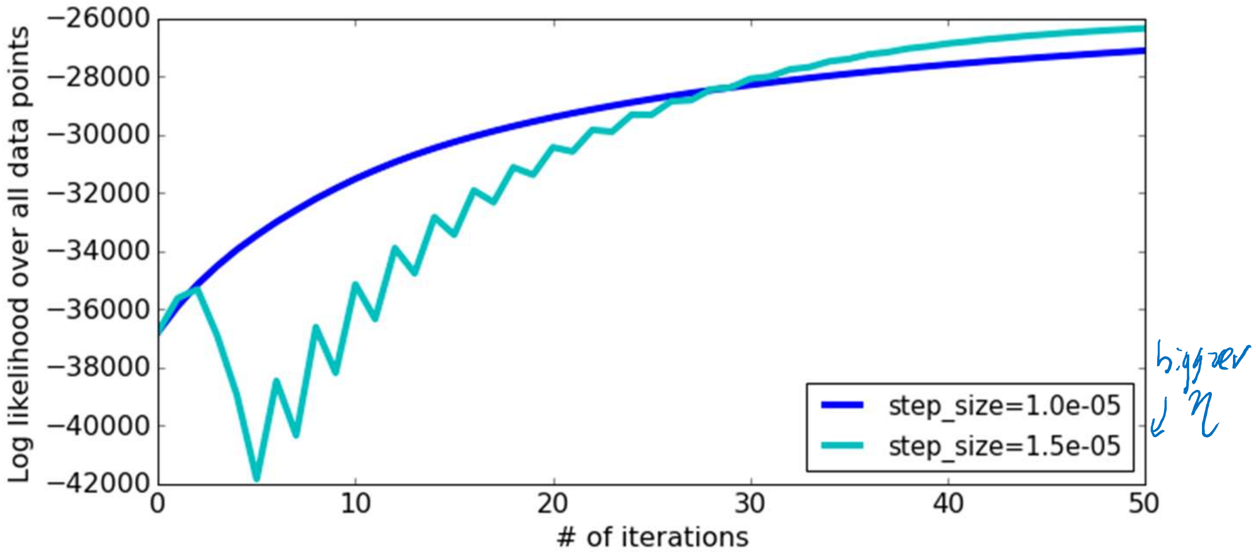
Step-size too small

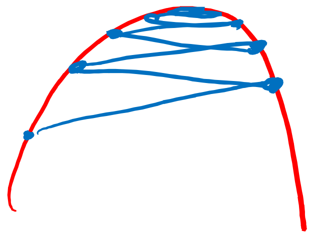




Choosing η

What about a larger step-size?

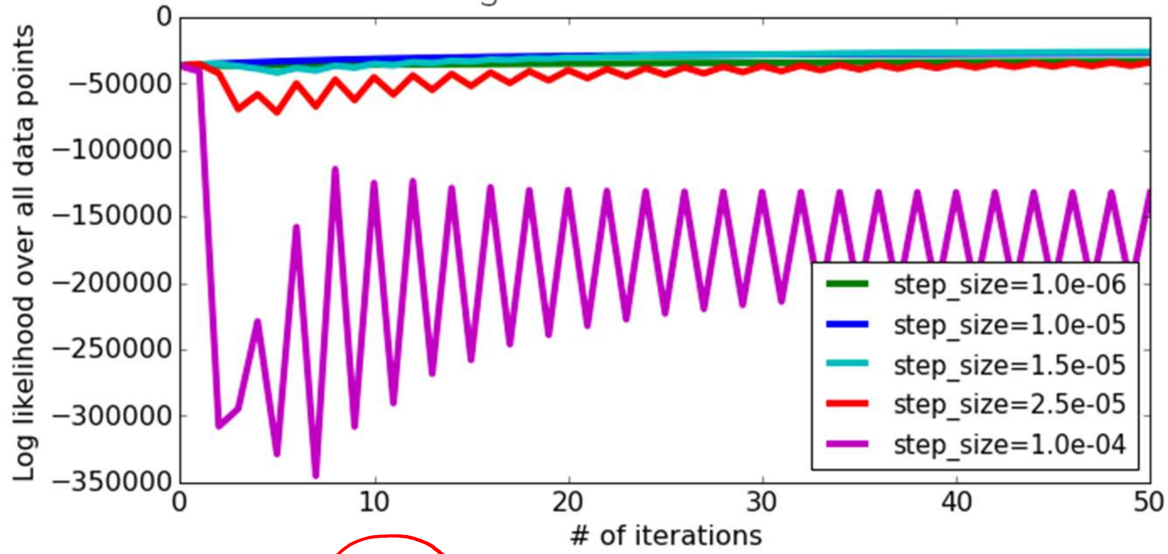


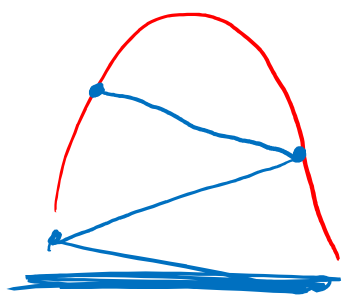


Choosing η

What about a larger step-size?

Can cause divergence!





Choosing η

Unfortunately, you have to do a lot of trial and error ☹

Try several values (generally exponentially spaced)

- Find one that is too small and one that is too large to narrow search range. Try values in between!

Advanced: Divergence with large step sizes tends to happen at the end, close to the optimal point. You can use a decreasing step size to avoid this

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

Brain Break

11:10

Overfitting - Classification

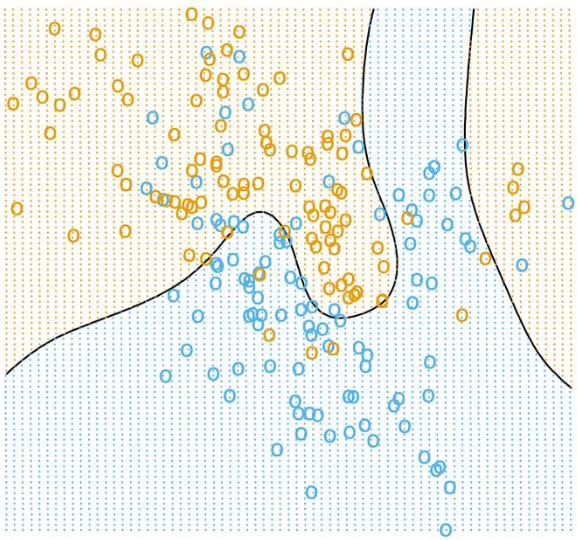
More Features

Like with regression, we can learn more complicated models by including more features or by including more complex features.

Instead of just using

$h_1(x) = \#awesome$

$h_2(x) = \#awful$



We could use

$h_1(x) = \#awesome$

$h_2(x) = \#awful$

$h_3(x) = \#awesome^2$

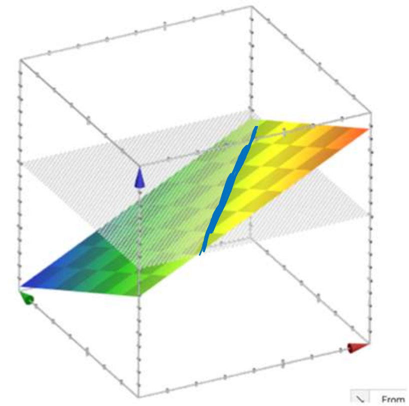
$h_4(x) = \#awful^2$

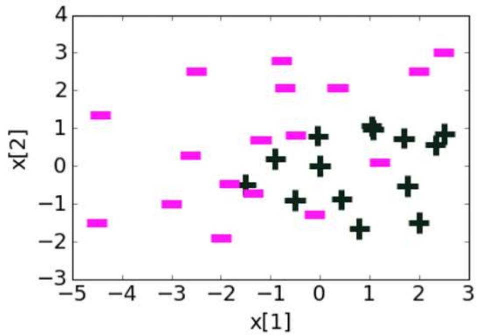
…

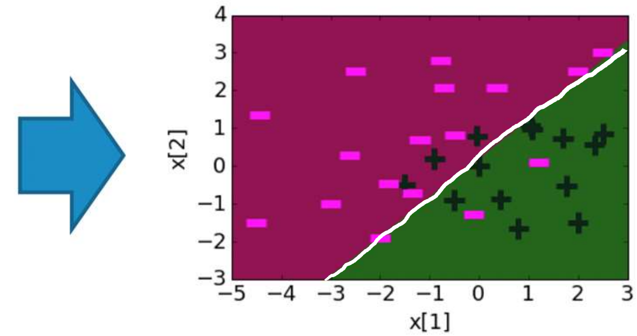
Decision Boundary

$$w^T h(x) = 0.23 + 1.12x[1] - 1.07x[2]$$

Feature	Value	Coefficient learned
$h_0(x)$	1	0.23
$h_1(x)$	x[1]	1.12
$h_2(x)$	x[2]	-1.07



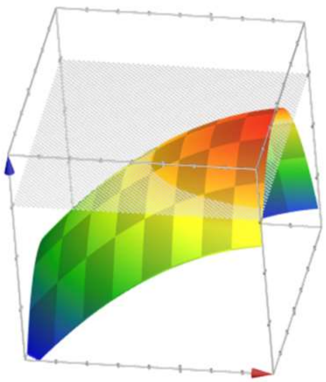


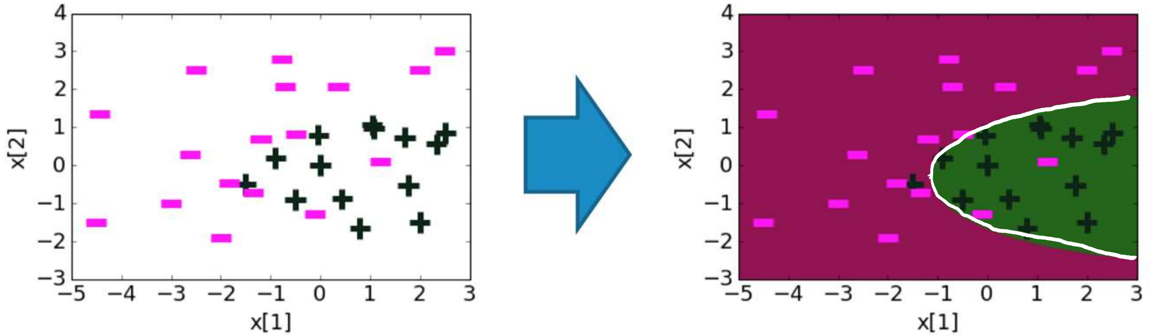


Decision Boundary

$$w^T h(x) = 1.68 + 1.39x[1] - 0.59x[2] - 0.17x[1]^2 - 0.96x[2]^2$$

Feature	Value	Coefficient learned
$h_0(x)$	1	1.68
$h_1(x)$	x[1]	1.39
$h_2(x)$	x[2]	-0.59
$h_3(x)$	$(x[1])^2$	-0.17
$h_4(x)$	$(x[2])^2$	-0.96

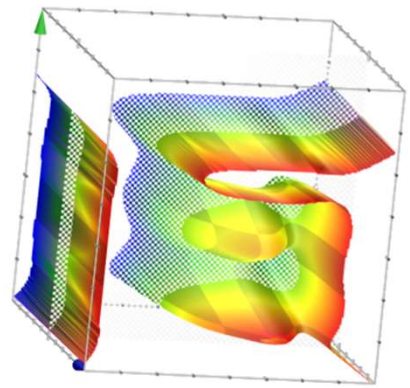




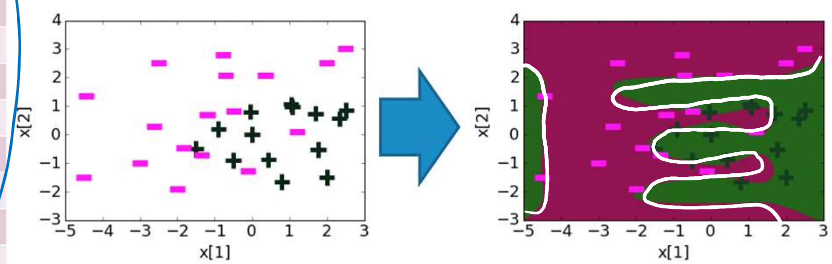
Decision Boundary

$$w^T h(x) = \cdots$$

getting larger



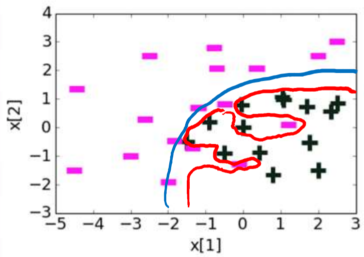
Feature	Value	Coefficient learned
$h_0(x)$	1	21.6
$h_1(x)$	x[1]	5.3
$h_2(x)$	x[2]	-42.7
$h_3(x)$	$(x[1])^2$	-15.9
$h_4(x)$	$(x[2])^2$	-48.6
$h_5(x)$	$(x[1])^3$	-11.0
$h_6(x)$	$(x[2])^3$	67.0
$h_7(x)$	$(x[1])^4$	1.5
$h_8(x)$	$(x[2])^4$	48.0
$h_9(x)$	$(x[1])^5$	4.4
$h_{10}(x)$	$(x[2])^5$	-14.2
$h_{11}(x)$	$(x[1])^6$	0.8
$h_{12}(x)$	$(x[2])^6$	-8.6

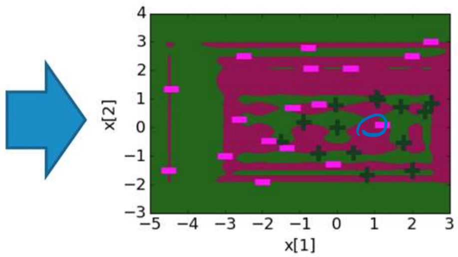


Decision Boundary

$$w^T h(x) = \cdots$$

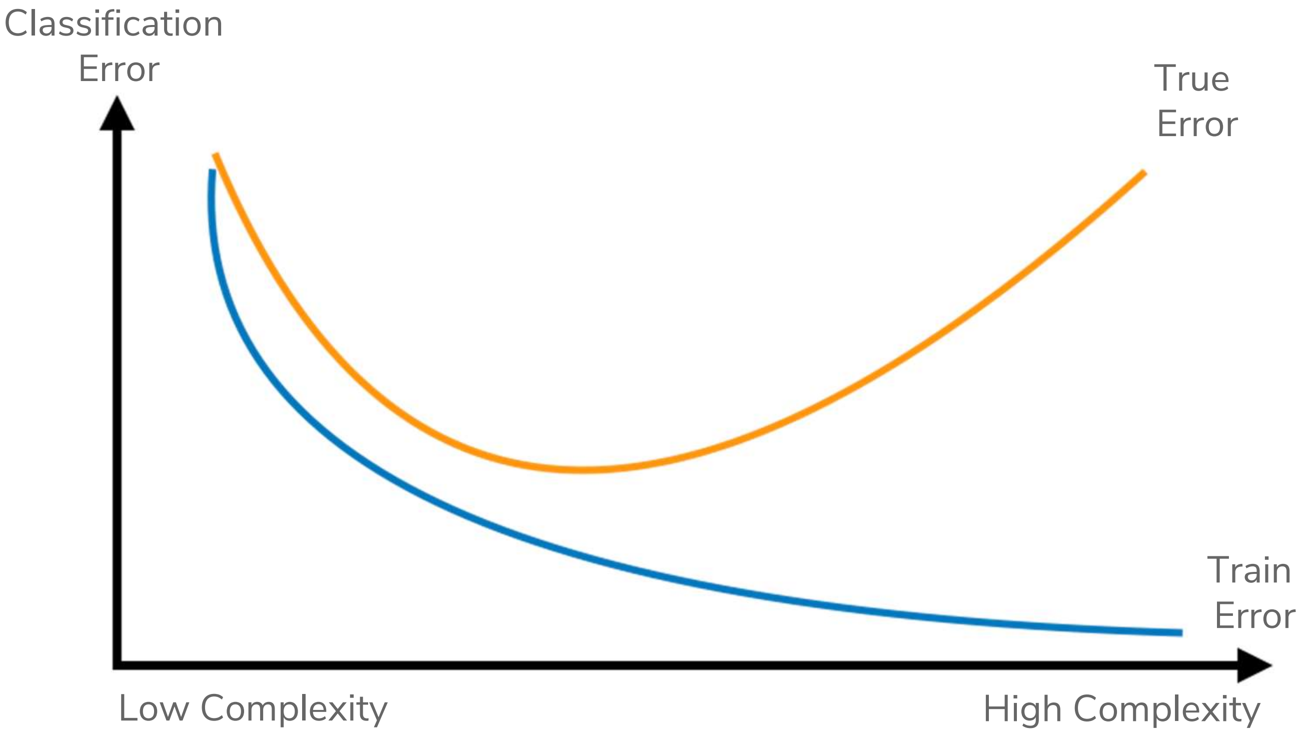
Feature	Value	Coefficient learned
$h_0(x)$	1	8.7
$h_1(x)$	x[1]	5.1
$h_2(x)$	x[2]	78.7
…	…	…
$h_{11}(x)$	$(x[1])^6$	-7.5
$h_{12}(x)$	$(x[2])^6$	3803
$h_{13}(x)$	$(x[1])^7$	21.1
$h_{14}(x)$	$(x[2])^7$	-2406
…	…	…
$h_{37}(x)$	$(x[1])^{19}$	$-2*10^{-6}$
$h_{38}(x)$	$(x[2])^{19}$	-0.15
$h_{39}(x)$	$(x[1])^{20}$	$-2*10^{-8}$
$h_{40}(x)$	$(x[2])^{20}$	0.03





Overfitting

Just like with regression, we see a similar pattern with complexity



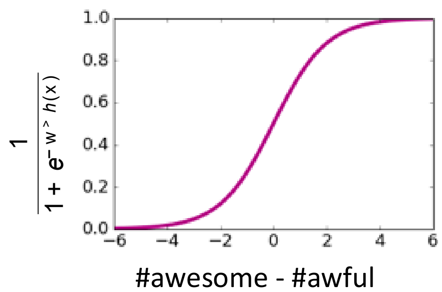
Effects of Overfitting

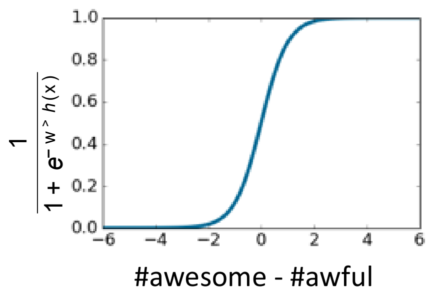
Remember, we say the logistic function become "sharper" with larger coefficients.

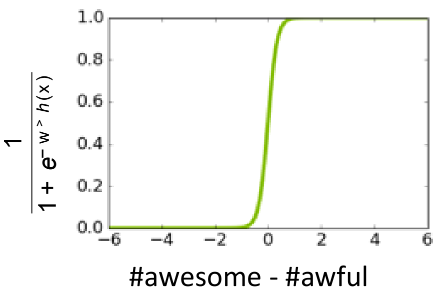
w_0	0
$w_{\#awesome}$	+1
$w_{\#awful}$	-1

w_0	0
$w_{\#awesome}$	+2
$w_{\#awful}$	-2

w_0	0
$w_{\#awesome}$	+6
$w_{\#awful}$	-6





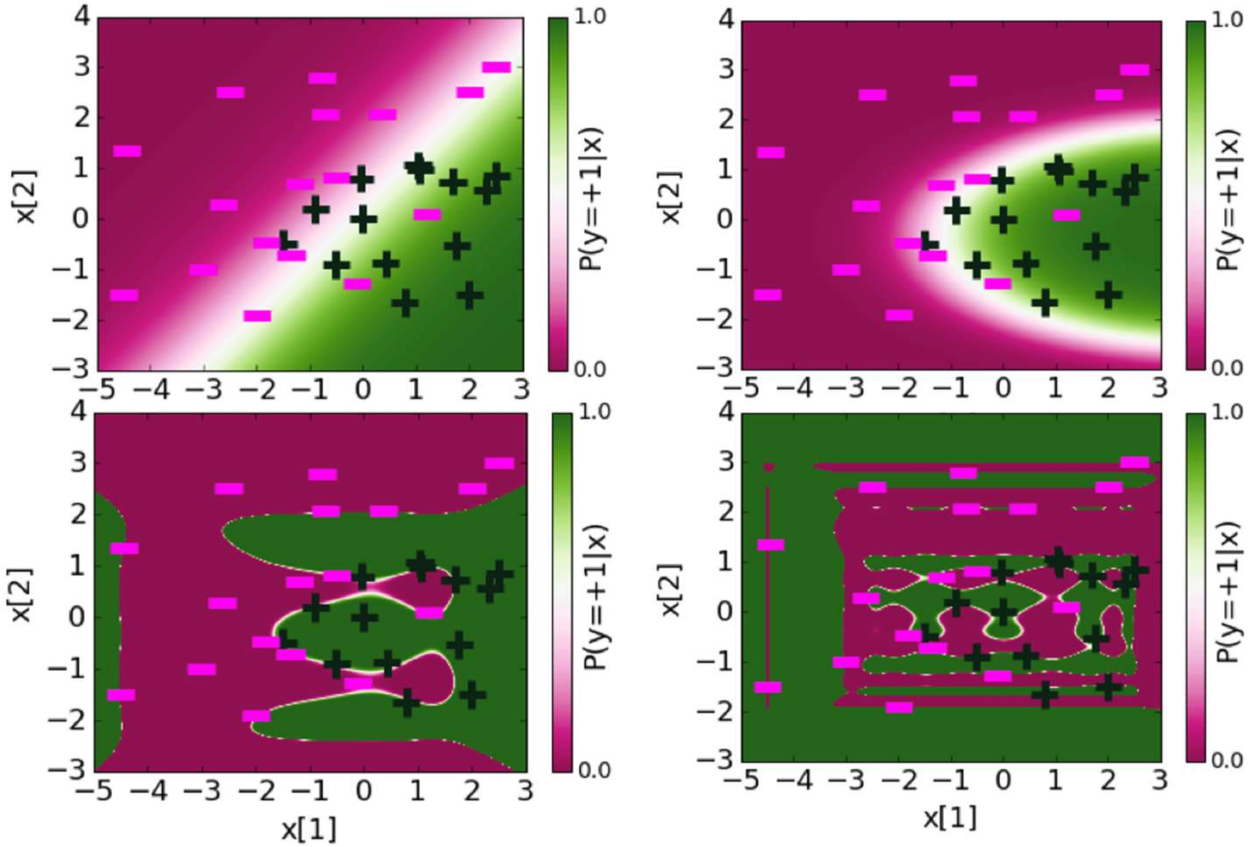


What does this mean for our predictions?

Because the $Score(x)$ is getting larger in magnitude, the probabilities are closer to 0 or 1!

Plotting Probabilities

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

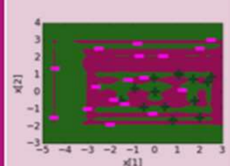
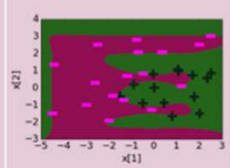
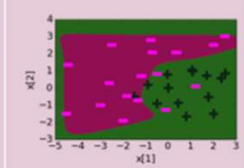
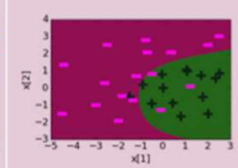
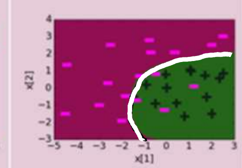
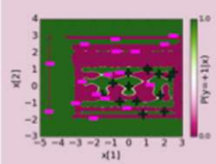
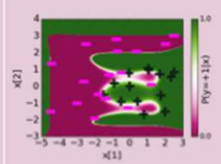
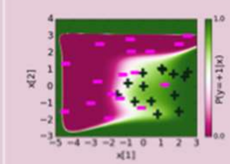
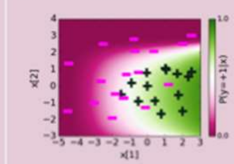


Regularization

L2 Regularized Logistic Regression

Just like in regression, can change our quality metric to avoid overfitting when training a model

$$\hat{w} = \max_{w} \ell(w) - \lambda \|w\|_2^2$$

Regularization	λ = 0	λ = 0.00001	λ = 0.001	λ = 1	λ = 10
Range of coefficients	-3170 to 3803	-8.04 to 12.14	-0.70 to 1.25	-0.13 to 0.57	-0.05 to 0.22
Decision boundary					
Learned probabilities					

Some Details

Why do we subtract the L2 Norm?

$$\hat{w} = \max_{w} \ell(w) - \lambda ||w||_2^2$$

How does λ impact the complexity of the model?

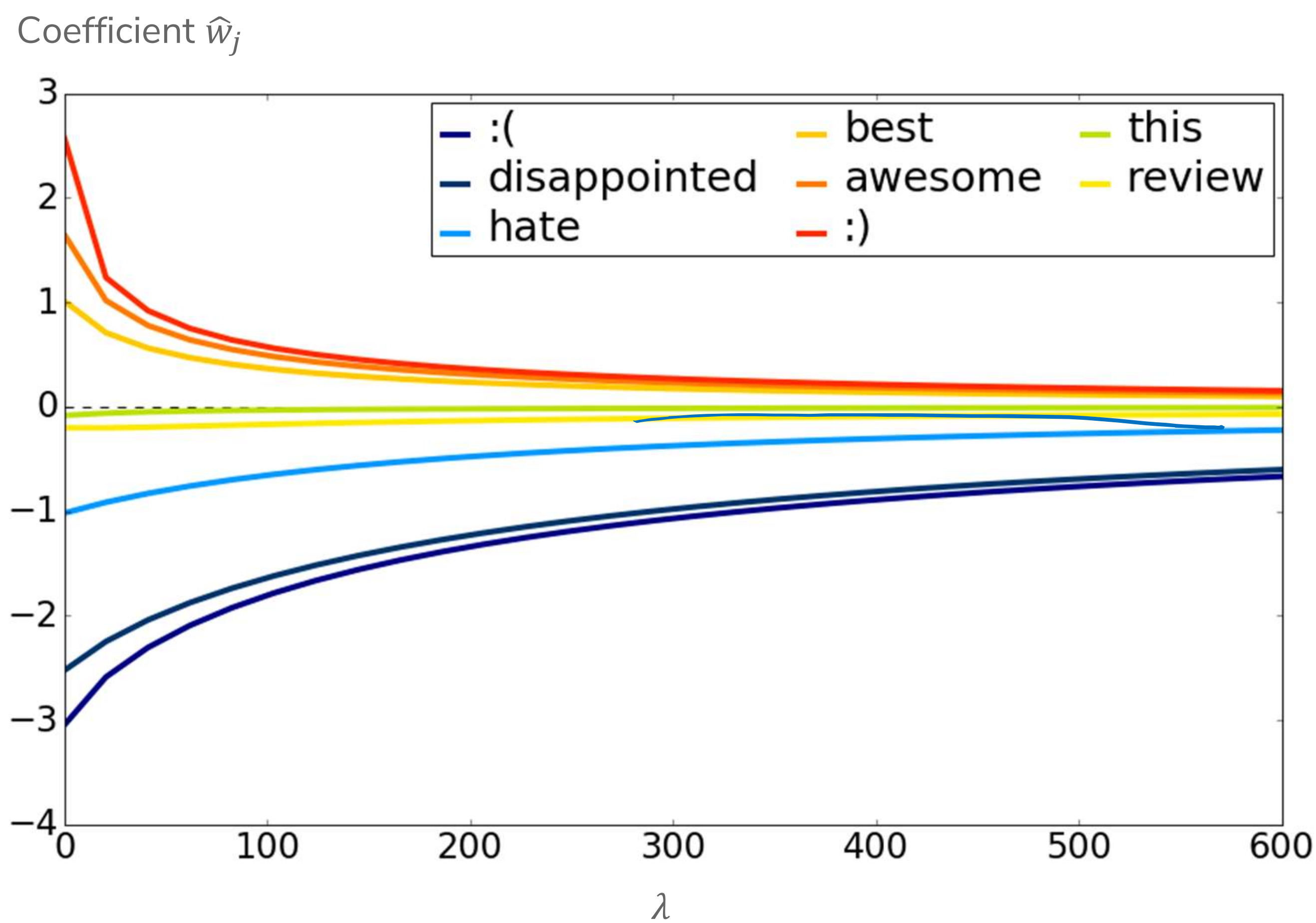
Same as Ridge

How do we pick λ?

Validation set

Cross validation

Coefficient Path: L2 Penalty



Other Penalties?

Could you use the L1 penalty instead? Absolutely!

$$\hat{w} = \max_{w} \ell(w) - \lambda ||w||_1$$

This is **L1 regularized logistic regression**

It has the same properties as the LASSO

- Increasing λ decreases $||\hat{w}||_1$
- The LASSO favors sparse solutions

Recap

Theme: Details of the linear model for classification and how to train it

Ideas:

- Minimizing error vs maximizing likelihood
- Predict with probabilities
- Using the logistic function to turn Score to probability
- Logistic Regression
- Gradient Ascent
- Step size
- Overfitting with logistic regression
  - Over-confident
  - Regularization

Reflection

Spend 10 minutes writing a reflection on a piece of paper

1) What did you learn this week? How does it relate to what we learned earlier in the quarter?

2) Are there any topics you are still finding a bit confusing?

It helps if you clearly indicate which questions you are answering on the paper!