



# Linear classifiers:



# Logistic regression



Emily Fox & Carlos Guestrin

Machine Learning Specialization

University of Washington

# Predicting sentiment by topic: *An intelligent restaurant review system*



# It's a big day & I want to book a table at a nice Japanese restaurant

Seattle has many  
★★★★★  
sushi restaurants



What are people  
saying about  
the food?  
the ambiance?...





## Positive reviews not positive about everything



Sample review:

Watching the chefs create incredible edible art made the experience very unique.



My wife tried their ramen and it was pretty forgettable.



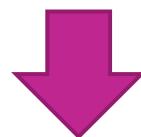
All the sushi was delicious! Easily best sushi in Seattle.



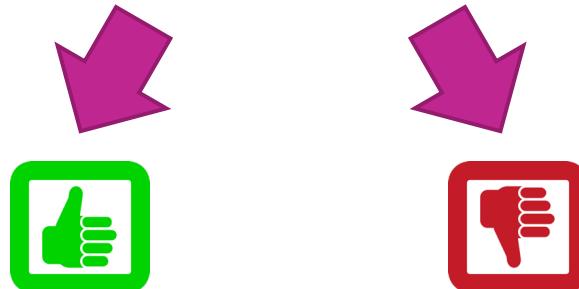
---

# Classifying sentiment of review

Easily best sushi in Seattle.



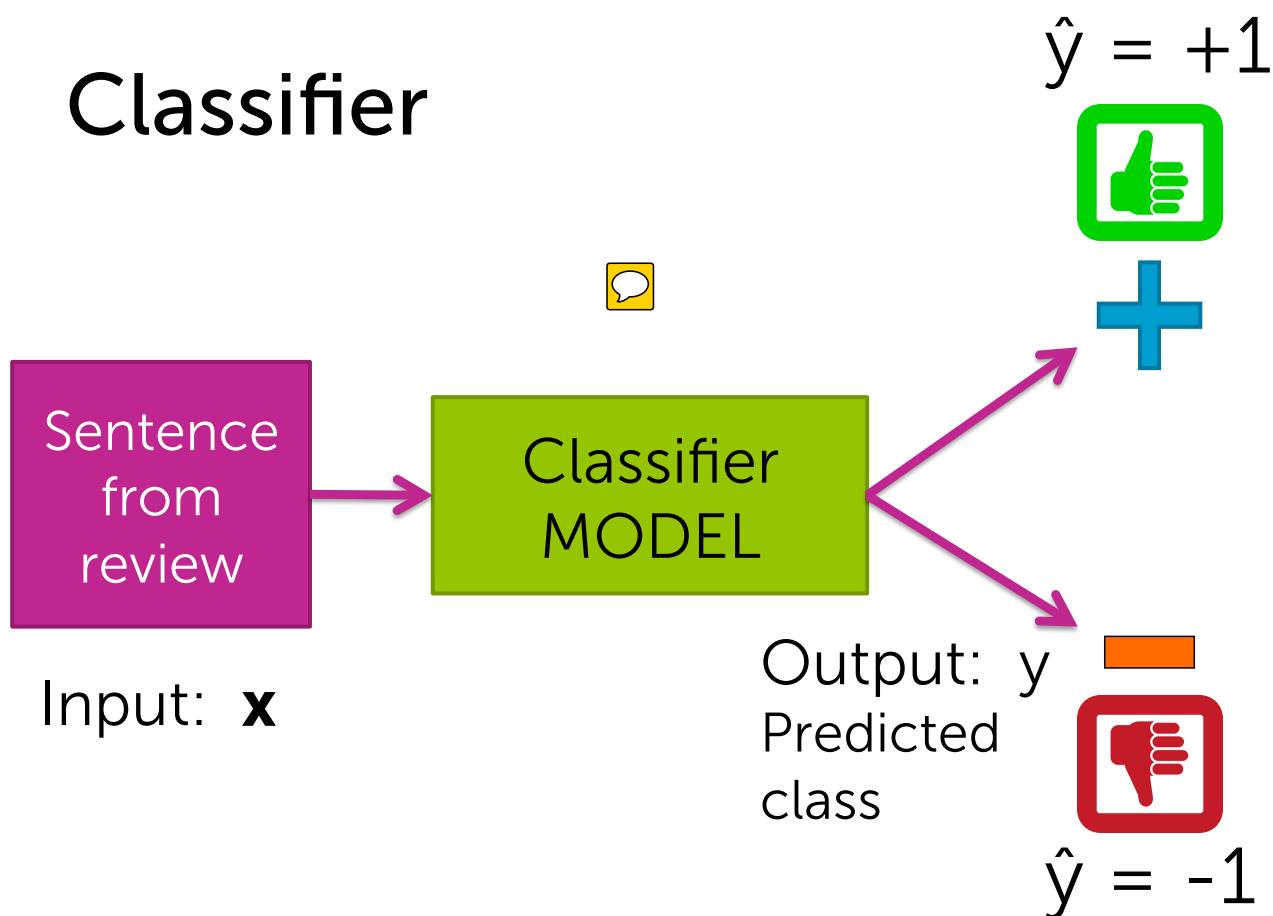
Sentence Sentiment  
Classifier



# Linear classifier: Intuition



# Classifier



Note: we'll start talking about 2 classes, and address multiclass later

# A (linear) classifier

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

Word	Coefficient
good	1.0
great	1.5
awesome	2.7
bad	-1.0
terrible	-2.1
awful	-3.3
restaurant, the, we, where, ...	0.0
...	...

# Scoring a sentence

Word	Coefficient
good	1.0
great	1.2
awesome	1.7
bad	-1.0
terrible	-2.1
awful	-3.3
restaurant, the, we, where, ...	0.0
...	...

Input  $\mathbf{x}_i$ :

Sushi was great,  
the food was awesome,  
but the service was terrible.



Called a linear classifier, because output is weighted sum of input.

Sentence  
from  
review

Input:  $\mathbf{x}$

Word	Coefficient
...	...



## Simple linear classifier

$\text{Score}(\mathbf{x})$  = weighted count of  
words in sentence

If  $\text{Score}(\mathbf{x}) > 0$ :

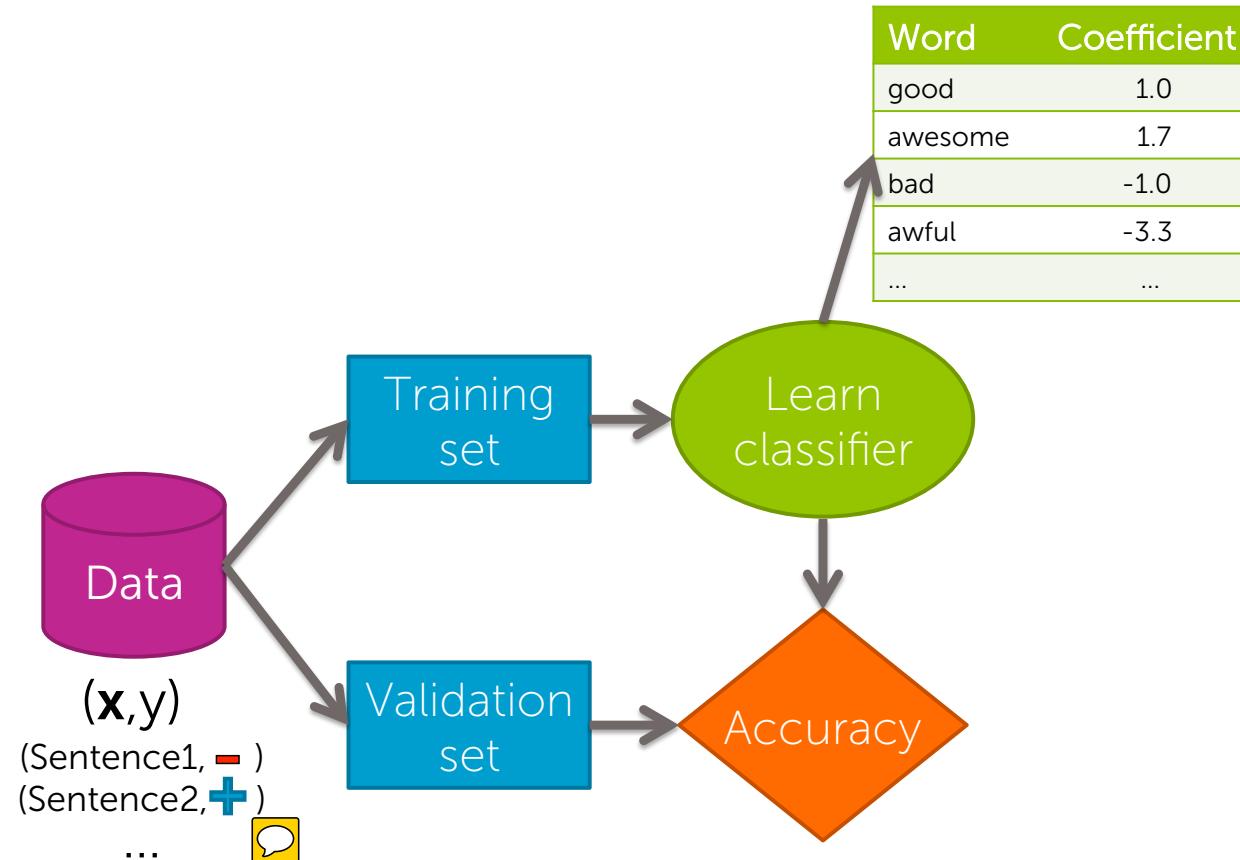
$$\hat{y} = +1$$



Else:

$$\hat{y} = -1$$

# Training a classifier = Learning the coefficients



# Decision boundaries



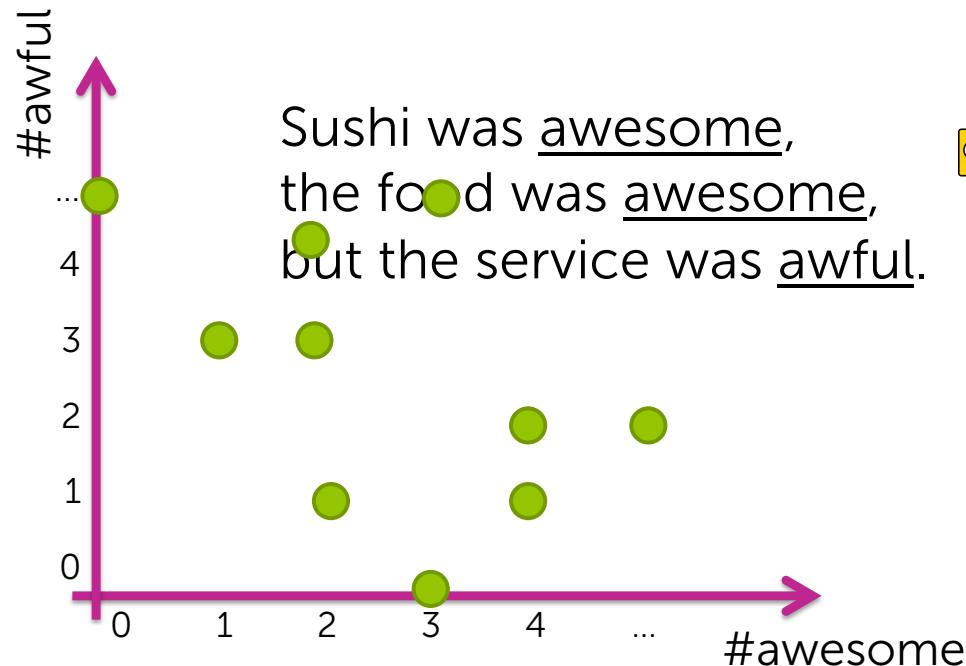
# Suppose only two words had non-zero coefficient



Word	Coefficient
#awesome	1.0
#awful	-1.5



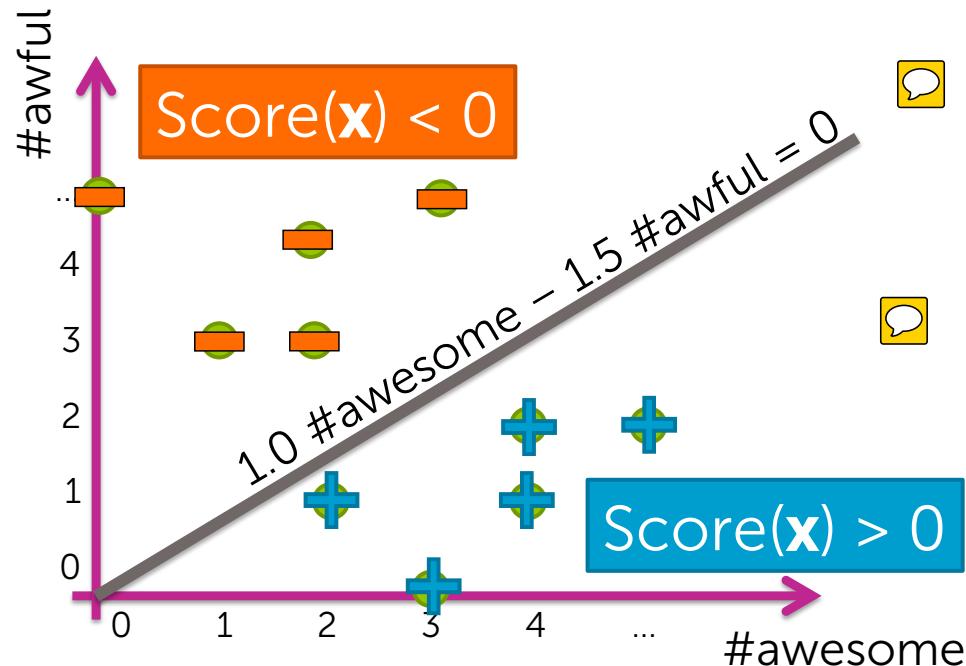
$$\text{Score}(x) = 1.0 \text{ #awesome} - 1.5 \text{ #awful}$$



# Decision boundary example

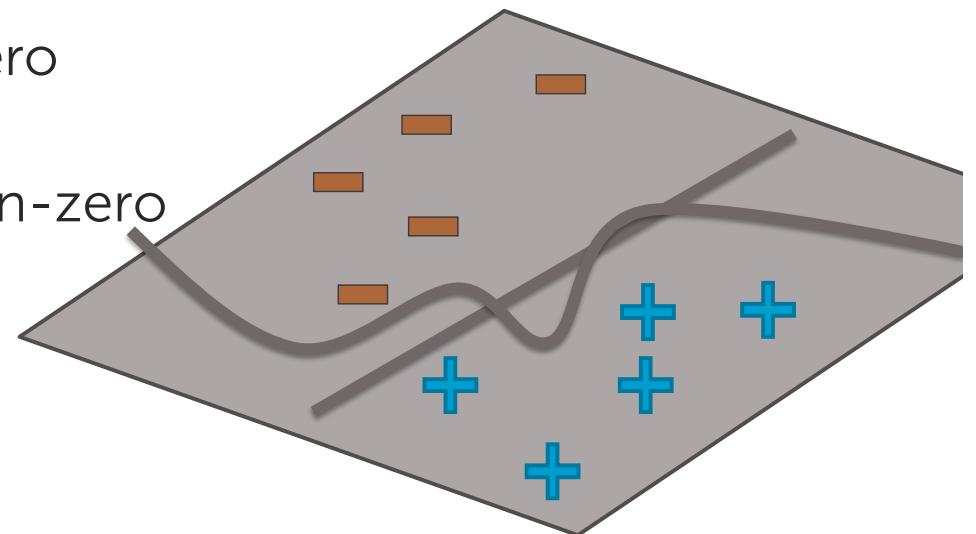
Word	Coefficient
#awesome	1.0
#awful	-1.5

$$\rightarrow \text{Score}(x) = 1.0 \text{ #awesome} - 1.5 \text{ #awful}$$



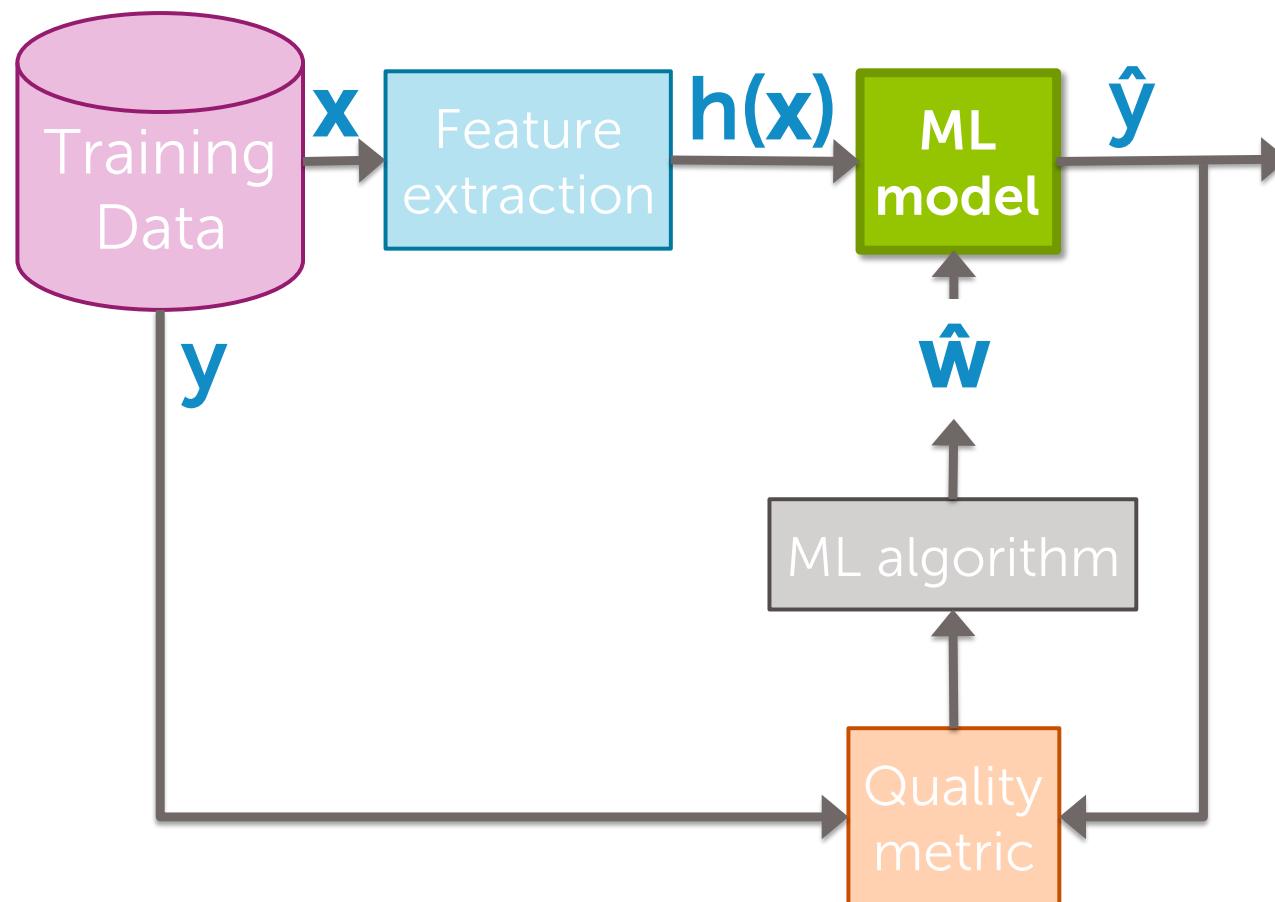
# Decision boundary separates positive & negative predictions

- For linear classifiers:
  - When 2 coefficients are non-zero  
→ line
  - When 3 coefficients are non-zero  
→ plane
  - When many coefficients are non-zero  
→ hyperplane
- For more general classifiers  
→ more complicated shapes

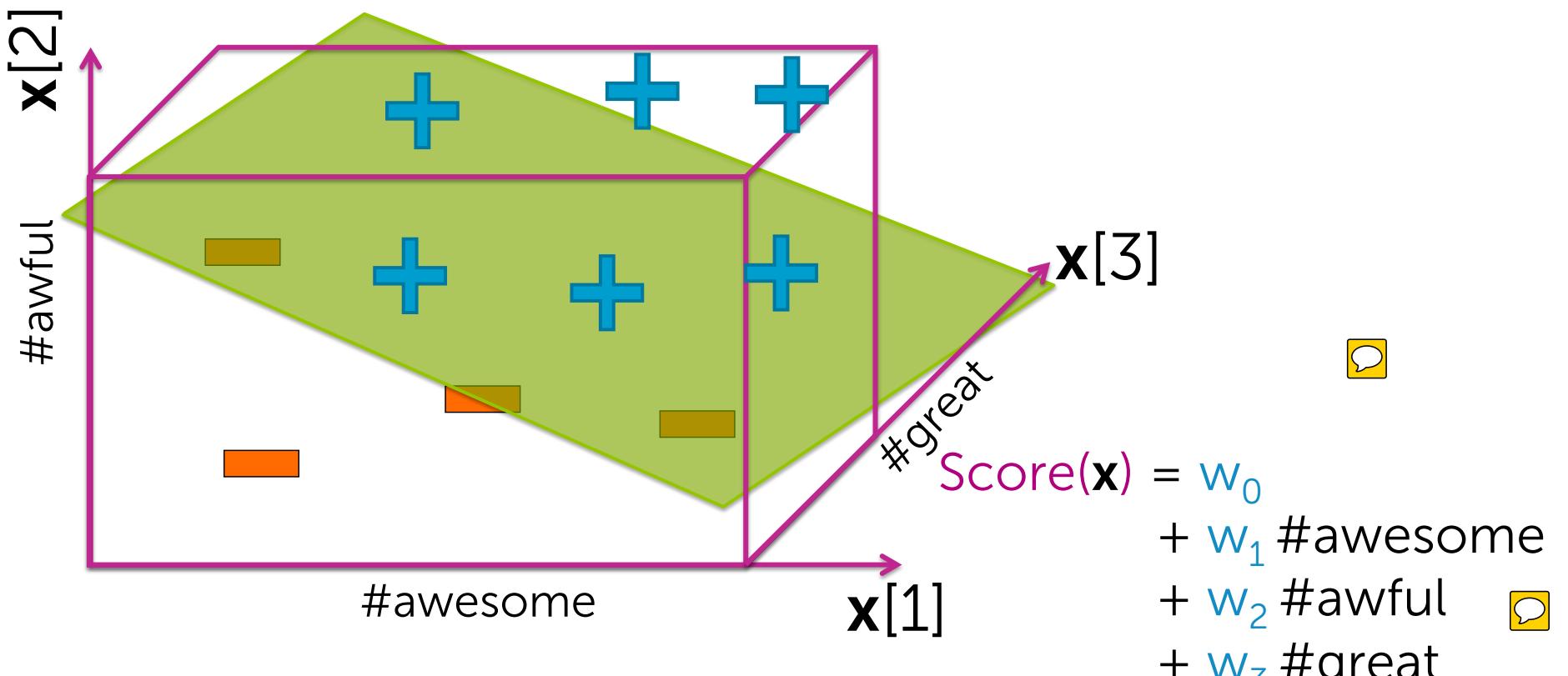


# Linear classifier: Model





# Coefficients of classifier



# General notation

Output:  $y \leftarrow \{-1, +1\}$

Inputs:  $\mathbf{x} = (\mathbf{x}[1], \mathbf{x}[2], \dots, \mathbf{x}[d])$

    ↑  
    d-dim vector



Notational conventions:

$\mathbf{x}[j]$  = j<sup>th</sup> input (*scalar*)

$h_j(\mathbf{x})$  = j<sup>th</sup> feature (*scalar*)

$\mathbf{x}_i$  = input of i<sup>th</sup> data point (*vector*)

$\mathbf{x}_i[j]$  = j<sup>th</sup> input of i<sup>th</sup> data point (*scalar*)

# Simple hyperplane

Model:  $\hat{y}_i = \text{sign}(\text{Score}(x_i))$  

$\text{Score}(x_i) = w_0 + w_1 x_i[1] + \dots + w_d x_i[d] = \mathbf{w}^\top \mathbf{x}_i$  

*feature 1 = 1*

*feature 2 =  $x[1]$  ... e.g., #awesome* 

*feature 3 =  $x[2]$  ... e.g., #awful*

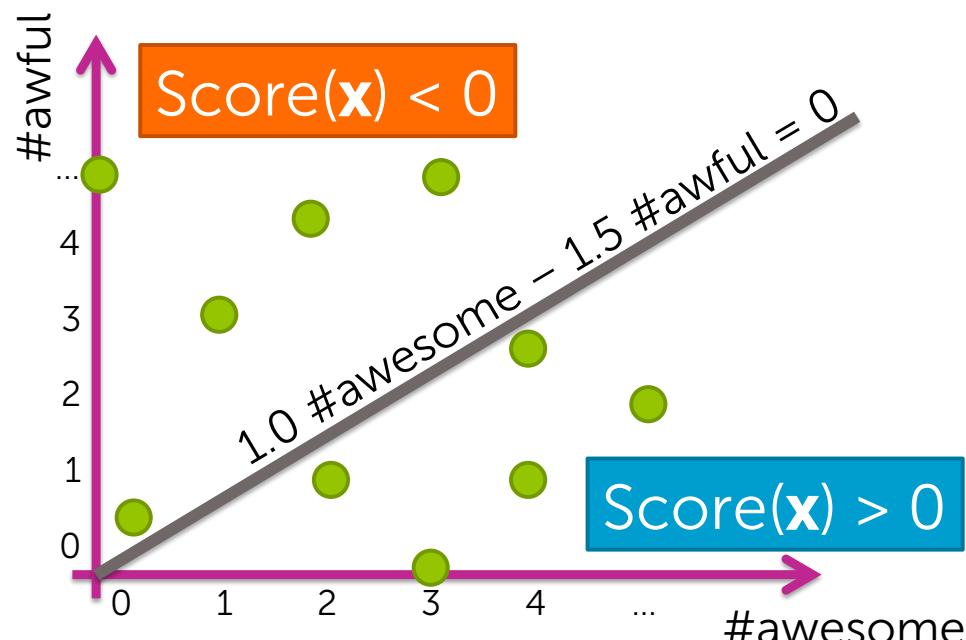
...

*feature  $d+1 = x[d]$  ... e.g., #ramen*

## Decision boundary: effect of changing coefficients

Input	Coefficient	Value
	$w_0$	0.0
#awesome	$w_1$	1.0
#awful	$w_2$	-1.5

$$\rightarrow \text{Score}(x) = 1.0 \# \text{awesome} - 1.5 \# \text{awful}$$

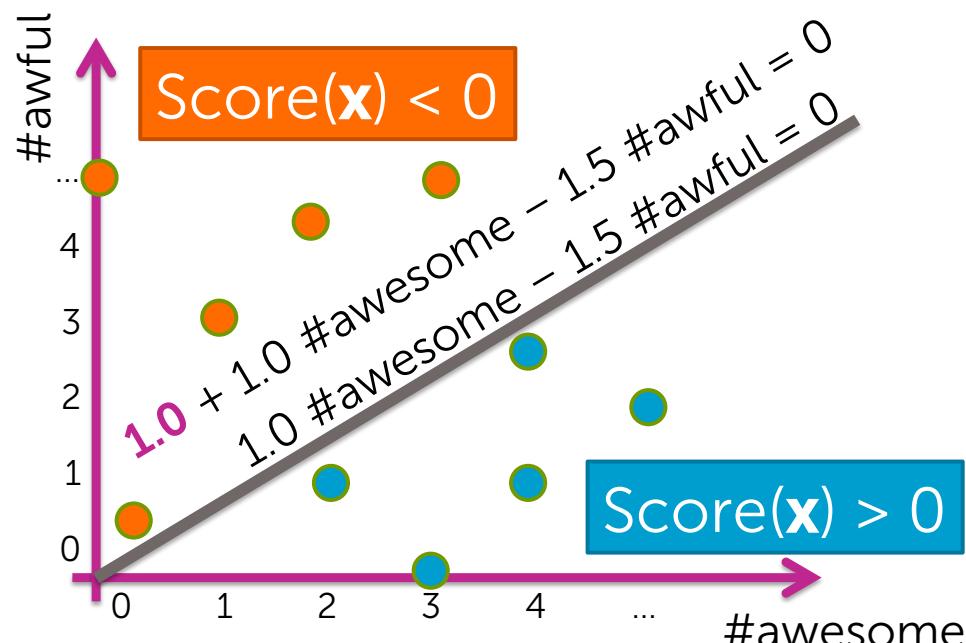


# Decision boundary: effect of changing coefficients

Input	Coefficient	Value
	$w_0$	1.0
#awesome	$w_1$	1.0
#awful	$w_2$	-1.5



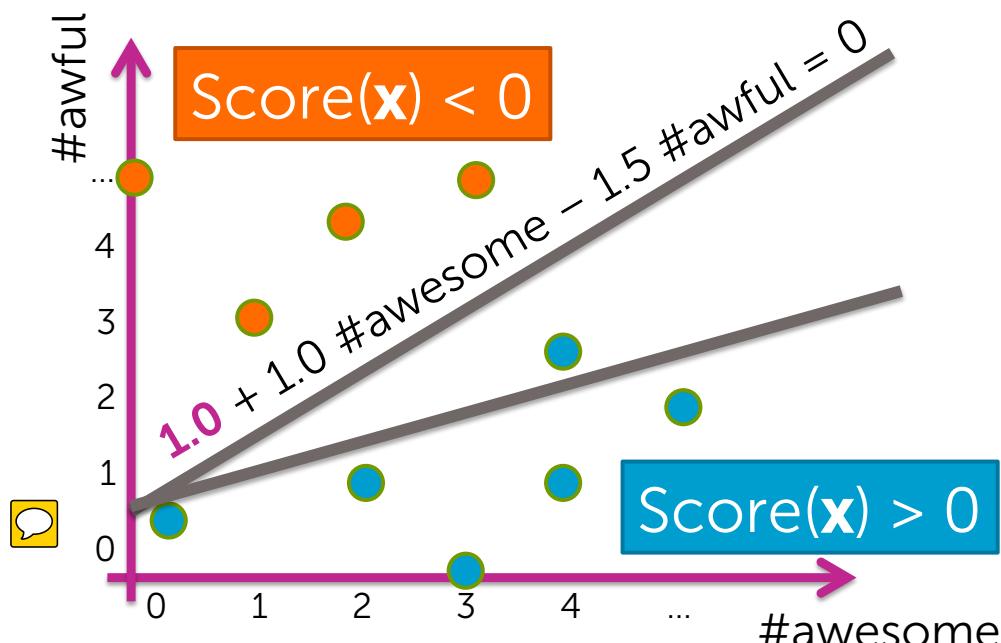
$$\rightarrow \text{Score}(x) = 1.0 \text{ #awesome} - 1.5 \text{ #awful}$$



# Decision boundary: effect of changing coefficients

Input	Coefficient	Value
	$w_0$	1.0
#awesome	$w_1$	1.0
#awful	$w_2$	-3.0

$$\rightarrow \text{Score}(x) = 1.0 + 1.0 \text{ #awesome} - 3.0 \text{ #awful}$$



# More generic features... D-dimensional hyperplane

Model:  $\hat{y}_i = \text{sign}(\text{Score}(\mathbf{x}_i))$

$$\text{Score}(\mathbf{x}_i) = w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \dots + w_D h_D(\mathbf{x}_i)$$

$$= \sum_{j=0}^D w_j h_j(\mathbf{x}_i) = \mathbf{w}^\top \mathbf{h}(\mathbf{x}_i)$$



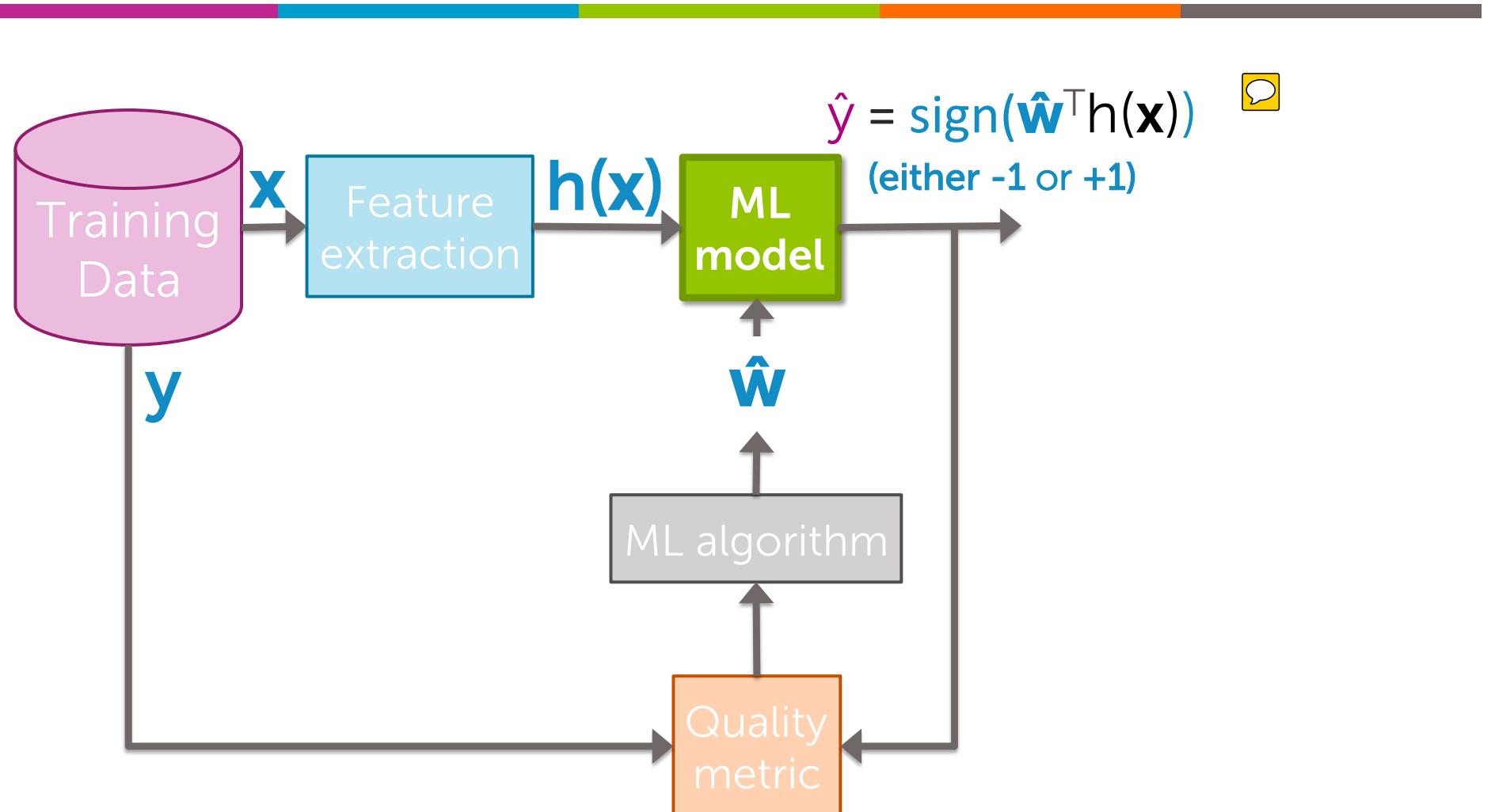
feature 1 =  $h_0(\mathbf{x})$  ... e.g., 1

feature 2 =  $h_1(\mathbf{x})$  ... e.g.,  $\mathbf{x}[1] = \text{\#awesome}$

feature 3 =  $h_2(\mathbf{x})$  ... e.g.,  $\mathbf{x}[2] = \text{\#awful}$   
or,  $\log(\mathbf{x}[7]) \mathbf{x}[2] = \log(\text{\#bad}) \times \text{\#awful}$   
or, tf-idf("awful")

...

feature  $D+1 = h_D(\mathbf{x})$  ... some other function of  $\mathbf{x}[1], \dots, \mathbf{x}[d]$

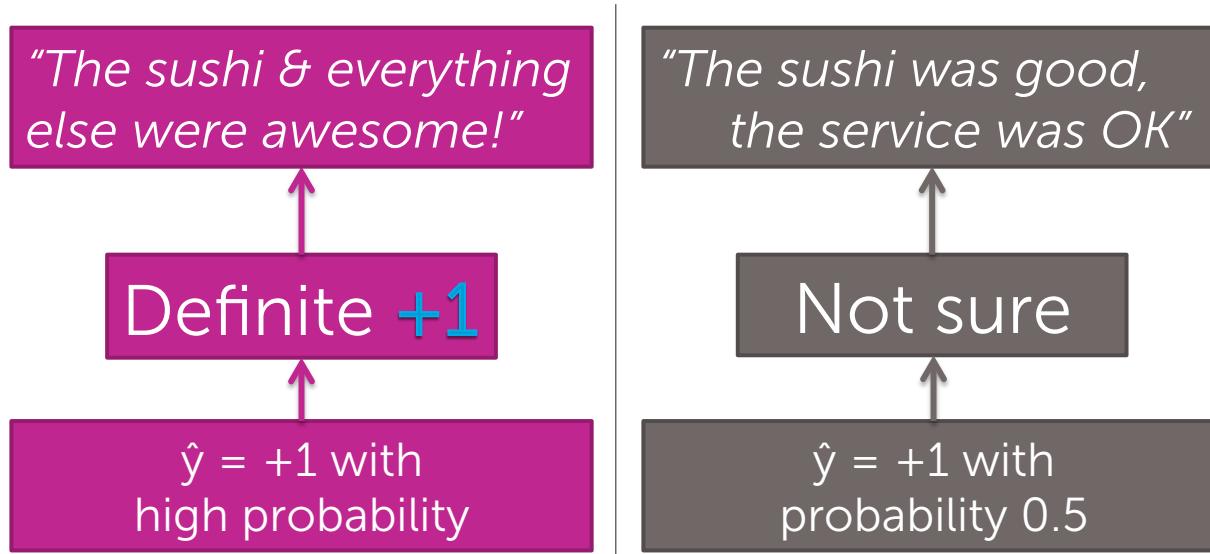




Are you sure about the prediction?  
**Class probability** 

# How confident is your prediction?

- Thus far, we've outputted a prediction **+1** or **-1**
- But, **how sure** are you about the prediction?

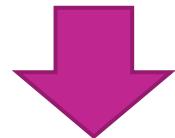




## Basics of probabilities – quick review

# Basic probability

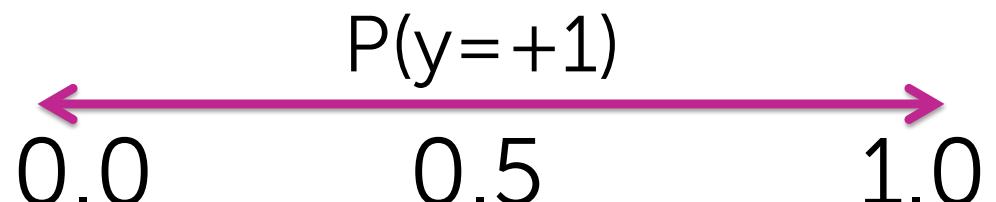
Probability a review is positive is 0.7



$x =$ review text	$y =$ sentiment
All the sushi was delicious! Easily best sushi in Seattle.	+1
The sushi & everything else were awesome!	+1
My wife tried their ramen, it was pretty forgettable.	-1
The sushi was good, the service was OK	+1
...	...

I expect 70% of rows  
to have  $y = +1$   
(Exact number will vary  
for each specific dataset)

## Interpreting probabilities as degrees of belief



Absolutely sure  
reviews are negative

Absolutely sure  
reviews are positive

Not sure if reviews are  
positive or negative

# Key properties of probabilities

Property	Two class (e.g., $y$ is +1 or -1)	Multiple classes (e.g., $y$ is dog, cat or bird)
Probabilities always between 0 & 1		
Probabilities sum up to 1		

# Conditional probability

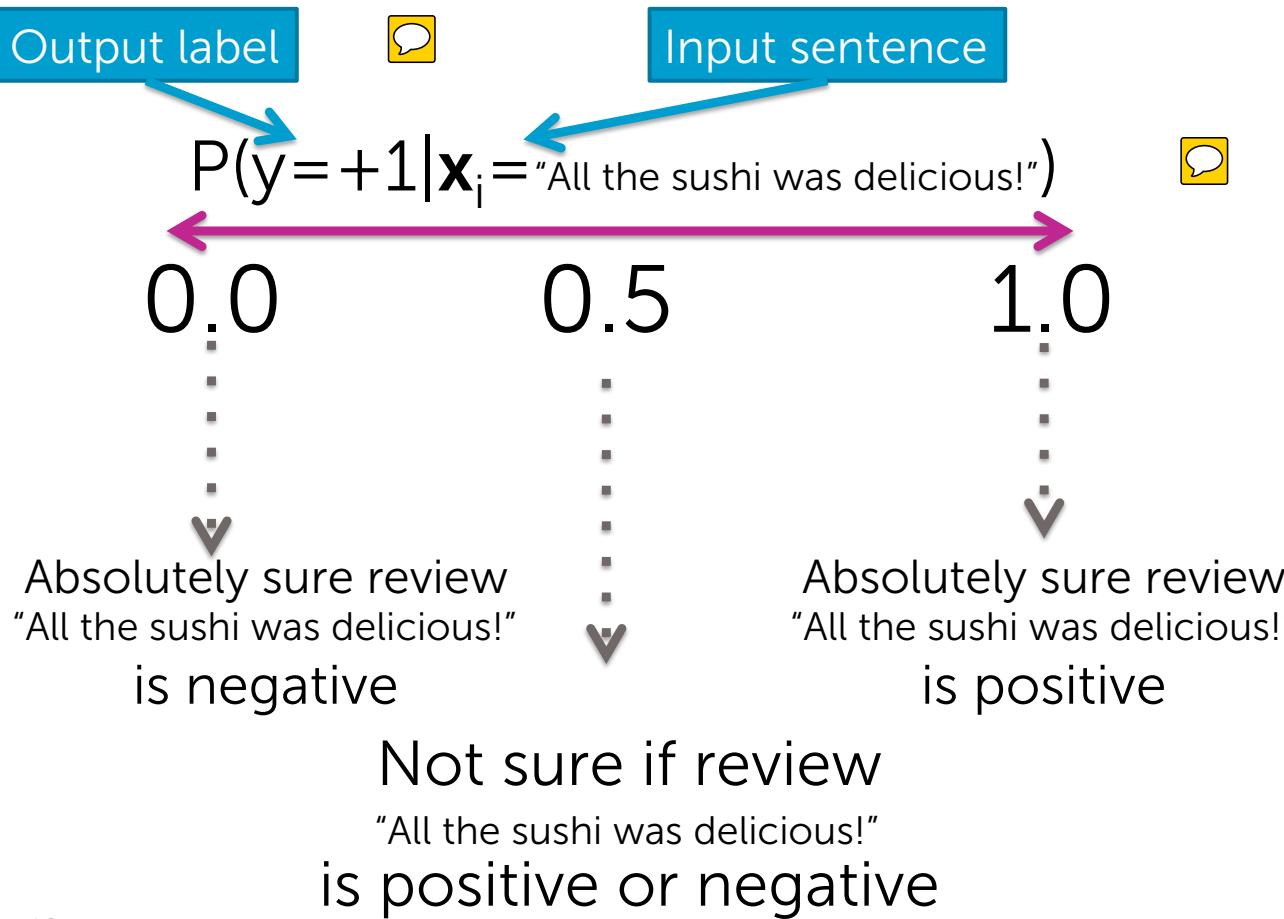
Probability a review with  
3 "awesome" and 1 "awful" is positive is 0.9



x = review text	y = sentiment
All the sushi was delicious! Easily best sushi in Seattle.	+1
Sushi was <b>awesome</b> & everything else was <b>awesome!</b> The service was <b>awful</b> , but overall <b>awesome</b> place!	+1
My wife tried their ramen, it was pretty forgettable.	-1
The sushi was good, the service was OK	+1
...	...
<b>awesome ... awesome ... awful ... awesome</b>	+1
...	...
<b>awesome ... awesome ... awful ... awesome</b>	-1
...	...
...	...
<b>awesome ... awesome ... awful ... awesome</b>	+1

I expect 90% of rows with reviews containing 3 "awesome" & 1 "awful" to have  $y = +1$   
(Exact number will vary for each specific dataset)

# Interpreting conditional probabilities



# Key properties of conditional probabilities

Property	Two class (e.g., $y$ is +1 or -1, $x_i$ is review text)	Multiple classes (e.g., $y$ is dog, cat or bird, $x_i$ is image)
Conditional probabilities always between 0 & 1		
Conditional probabilities sum up to 1 over $y$ , but not over $x$		



# How confident is your prediction?

"The sushi & everything else were awesome!"

Definite +1

$$P(y=+1|x=\text{"The sushi & everything else were awesome!"}) = 0.99$$

"The sushi was good, the service was OK"

Not sure

$$P(y=+1|x=\text{"The sushi was good, the service was OK"}) = 0.55$$

Many classifiers provide a degree of certainty:

Output label

Input sentence

$$P(y|x)$$

(Extremely useful in practice)



# Goal: Learn conditional probabilities from data

Training data:  $N$  observations  $(x_i, y_i)$

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



Optimize **quality metric**  
on training data

Find best model  $\hat{P}$   
by finding best  $\hat{w}$

Useful for  
predicting  $\hat{y}$



Sentence  
from  
review

Input:  $\mathbf{x}$

## Predict most likely class

$\hat{P}(y|x)$  = estimate of class probabilities

If  $\hat{P}(y=+1|x) > 0.5$ :

$\hat{y} = +1$

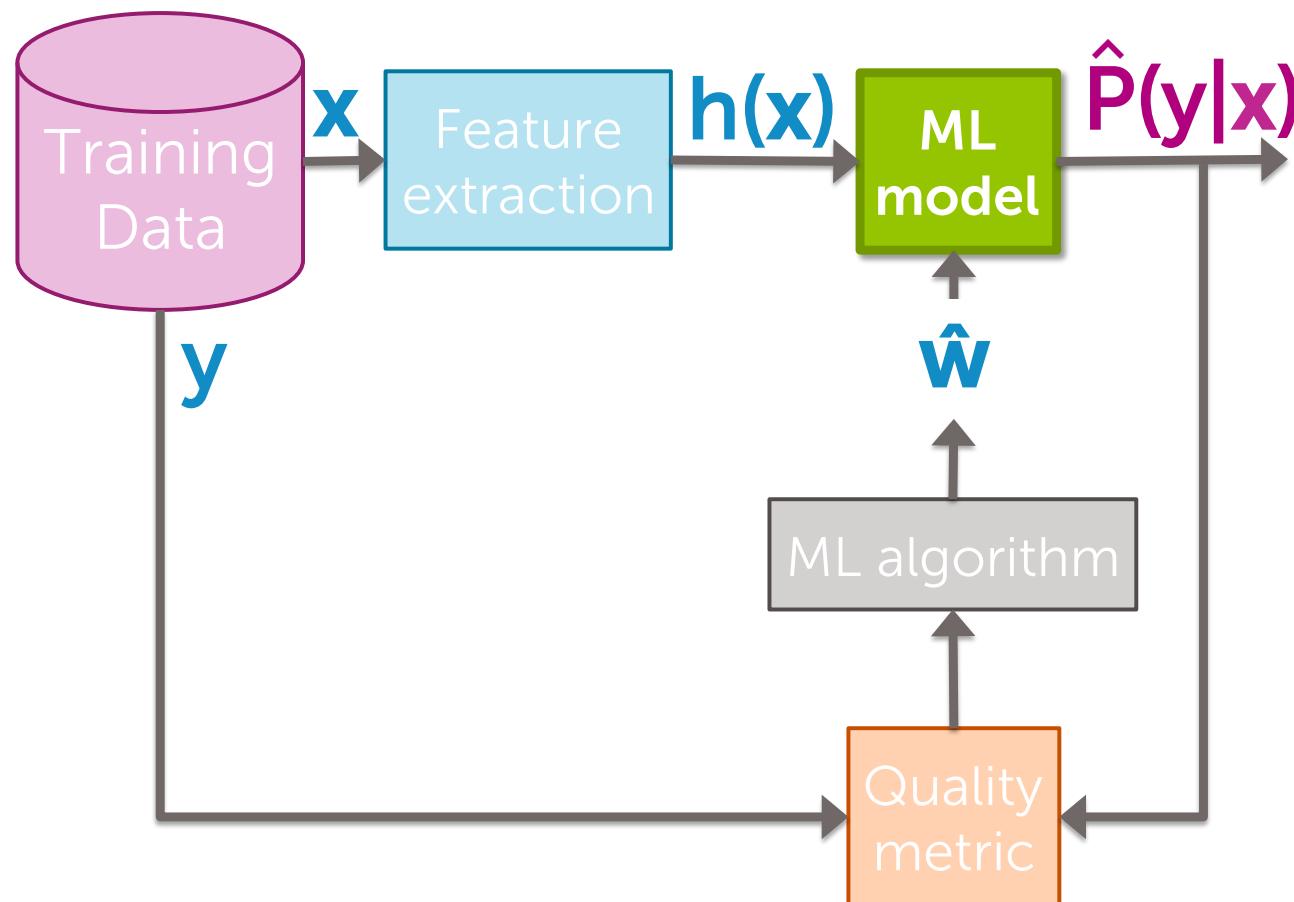
Else:

$\hat{y} = -1$

- Estimating  $\hat{P}(y|x)$  improves **interpretability**:
  - Predict  $\hat{y} = +1$  **and** tell me how sure you are

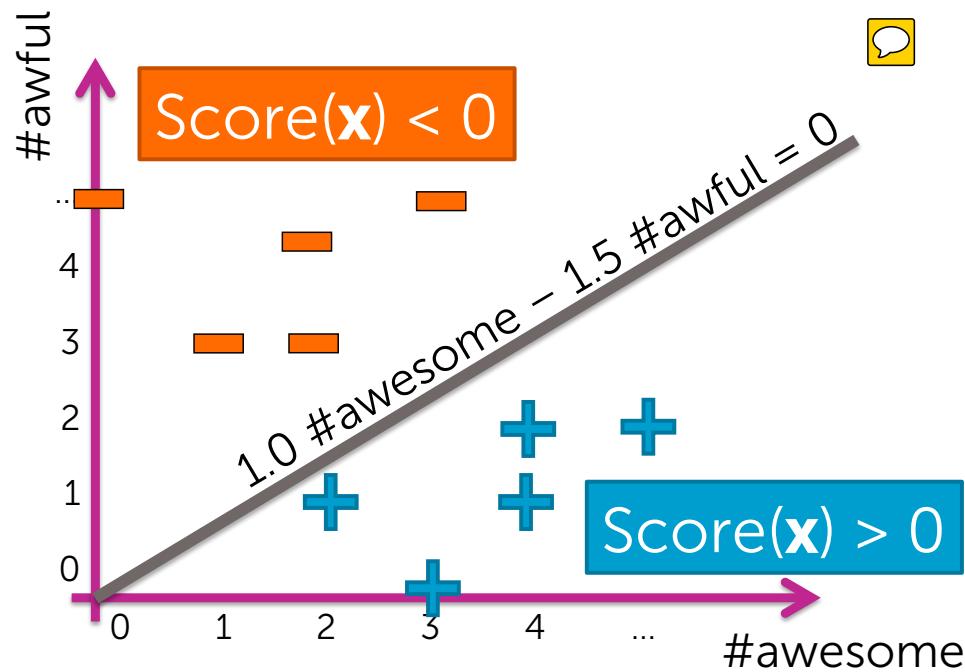
# Predicting class probabilities with generalized linear models





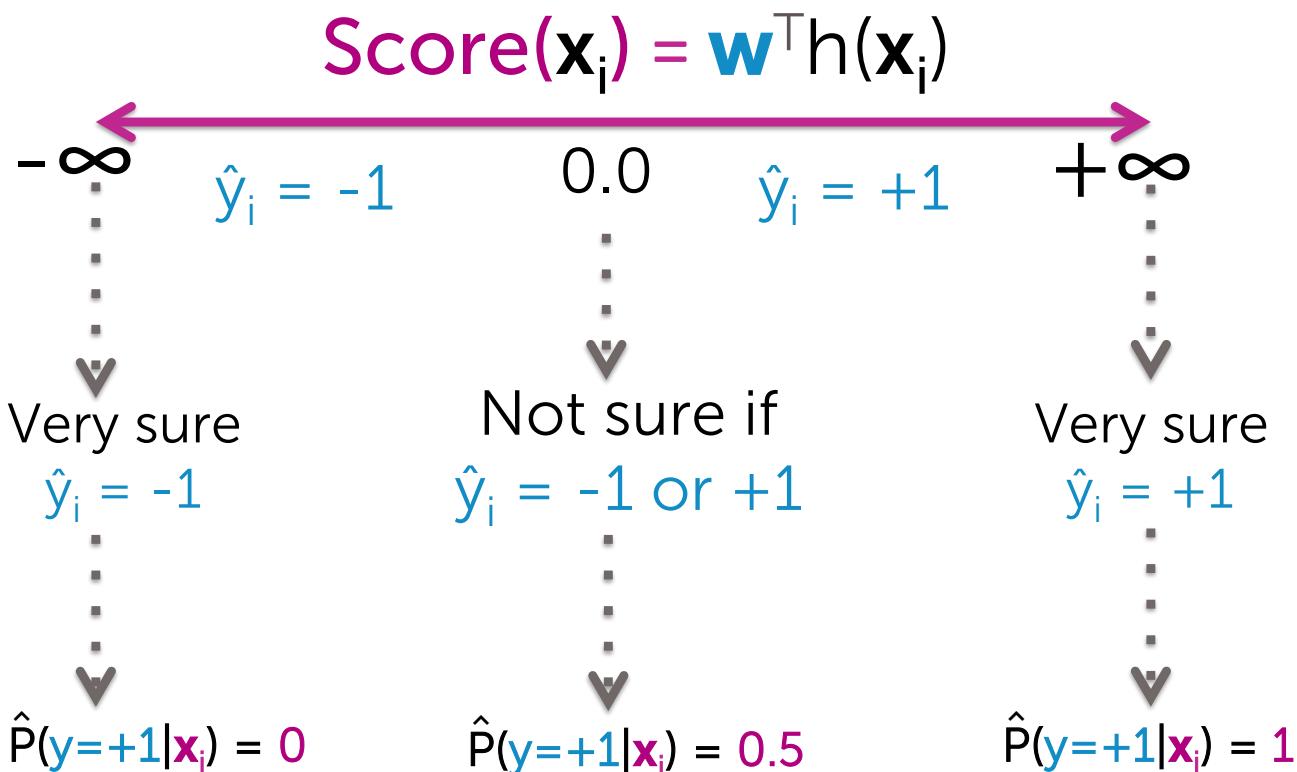
# Thus far, we focused on decision boundaries

$$\begin{aligned}\text{Score}(\mathbf{x}_i) &= w_0 h_0(\mathbf{x}_i) + w_1 h_1(\mathbf{x}_i) + \dots + w_D h_D(\mathbf{x}_i) \\ &= \mathbf{w}^\top \mathbf{h}(\mathbf{x}_i)\end{aligned}$$

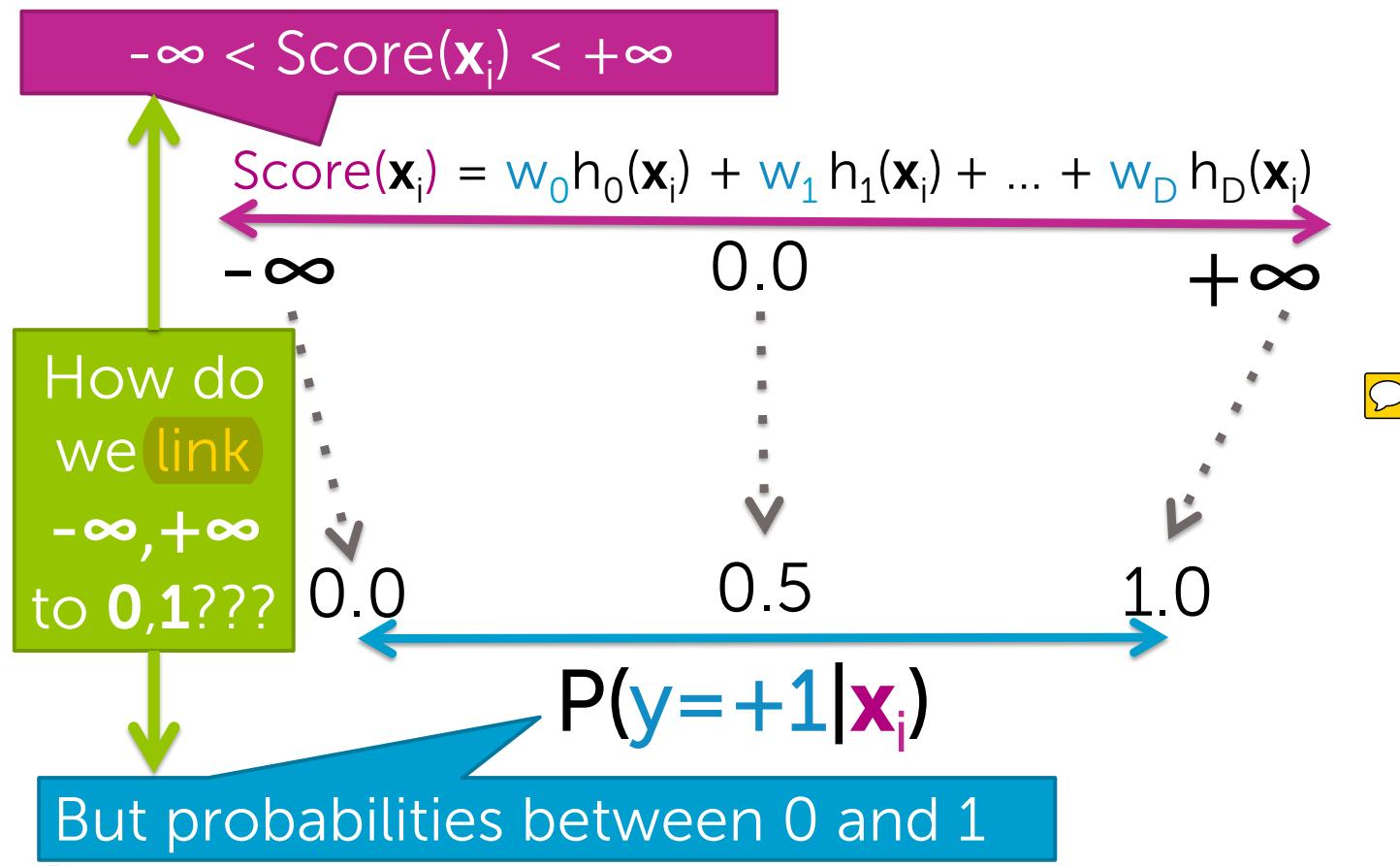


Relate  
Score( $\mathbf{x}_i$ ) to  
 $\hat{P}(y=+1|\mathbf{x}, \hat{\mathbf{w}})$ ?

# Interpreting Score( $\mathbf{x}_i$ )

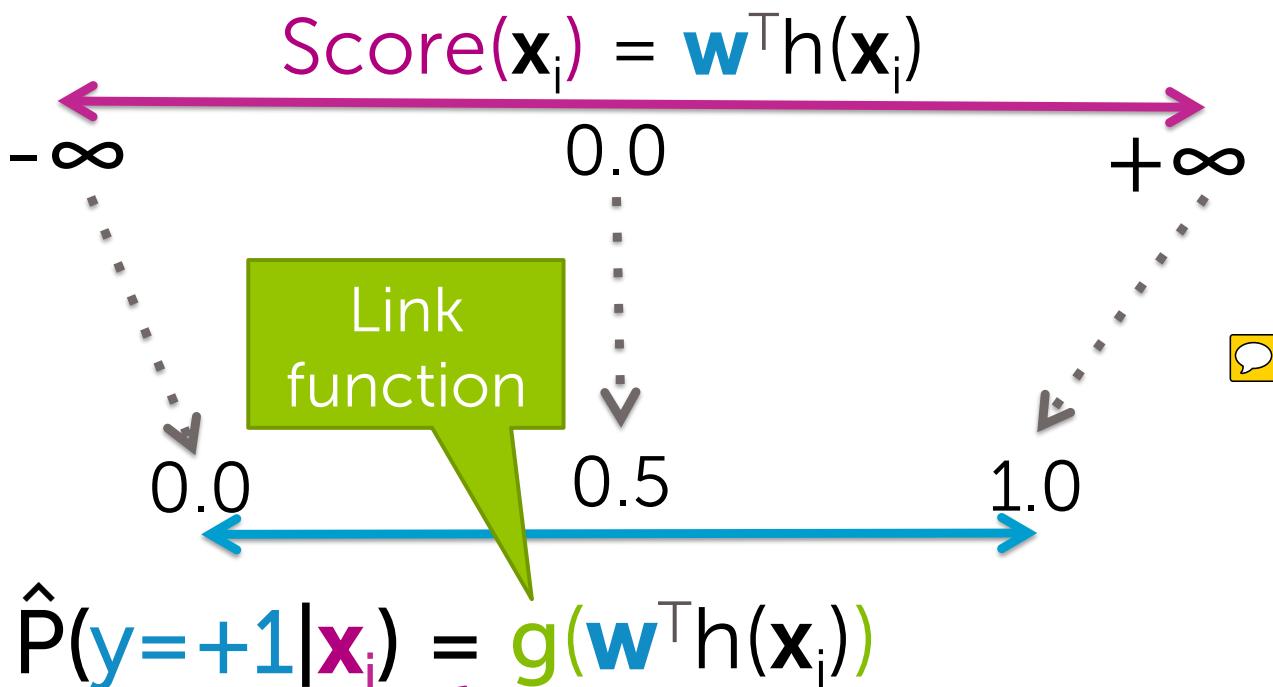


# Why not just use regression to build classifier?

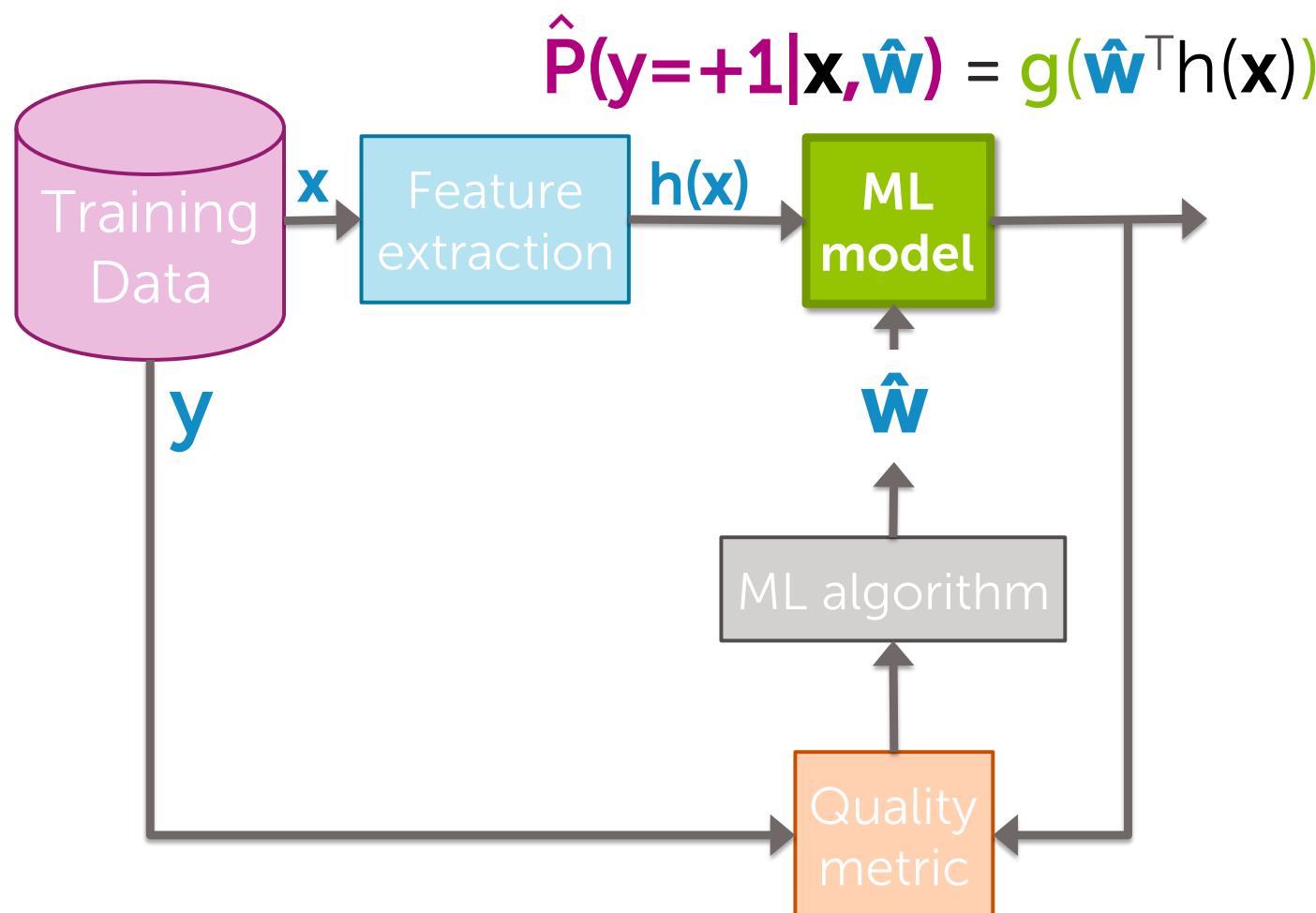


---

## *Link function: squeeze real line into [0,1]*



Generalized linear model



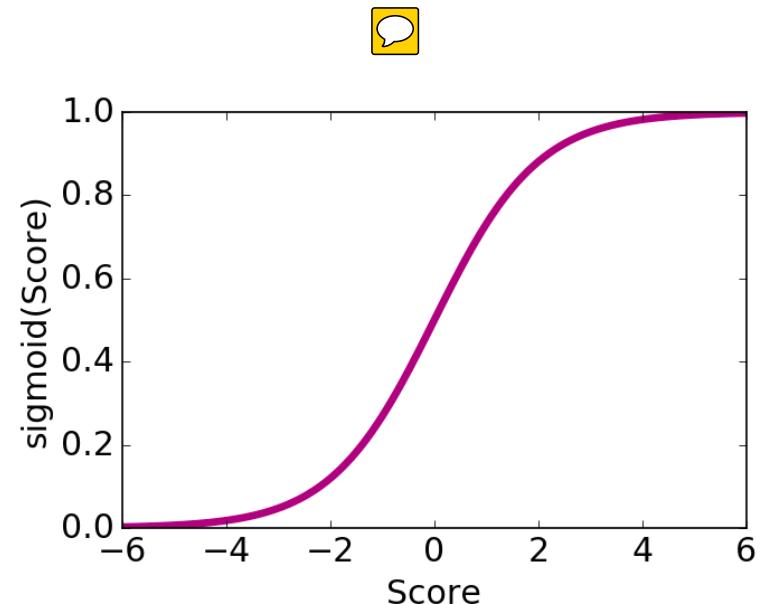
Logistic regression classifier:  
linear score with  
logistic link function



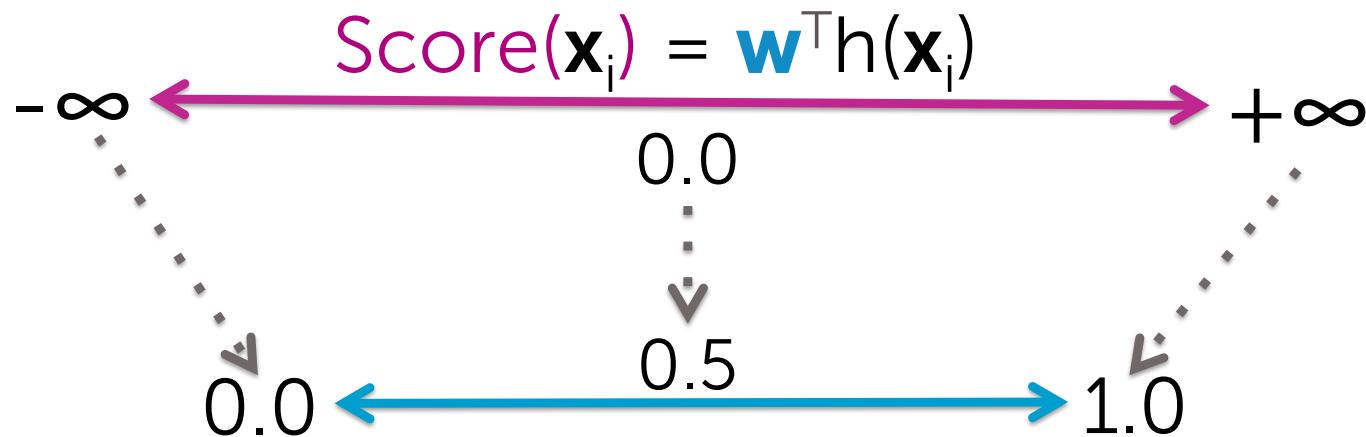
# Logistic function (sigmoid, logit)

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

Score	$-\infty$	-2	0.0	+2	$+\infty$
sigmoid(Score)					



# Logistic regression model

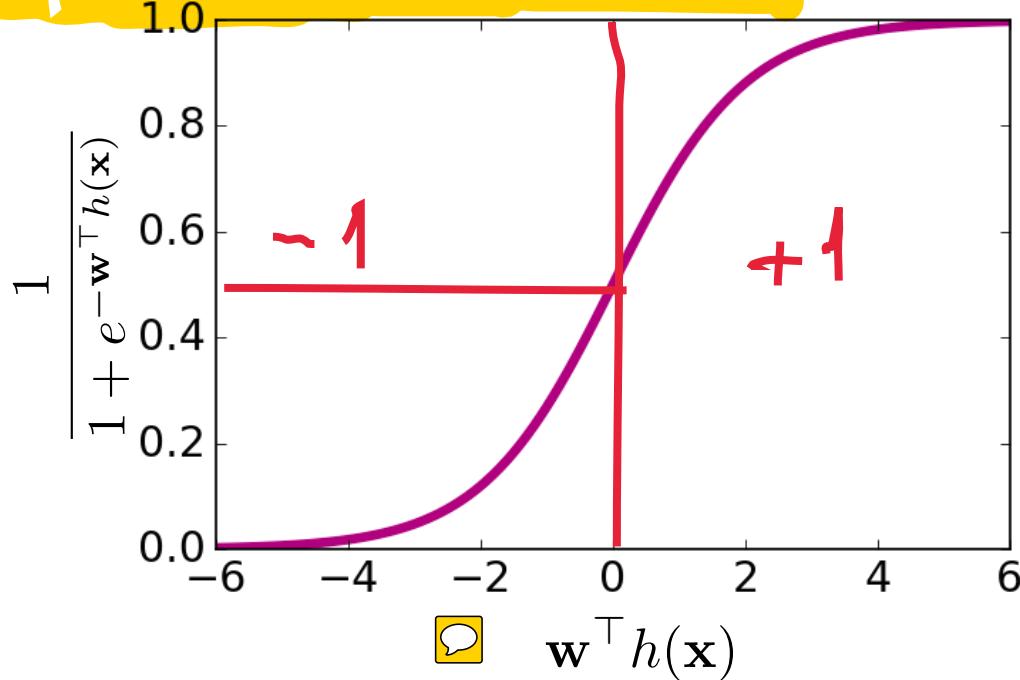


$$P(y=+1|\mathbf{x}_i, \mathbf{w}) = \text{sigmoid}(\text{Score}(\mathbf{x}_i))$$



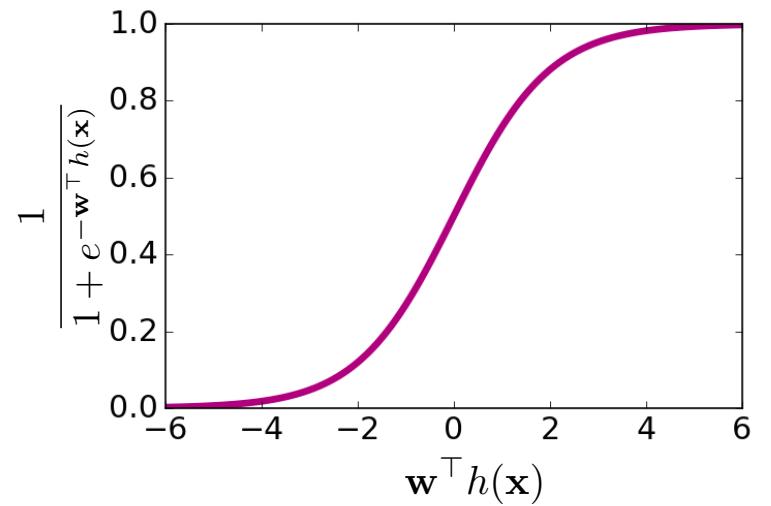
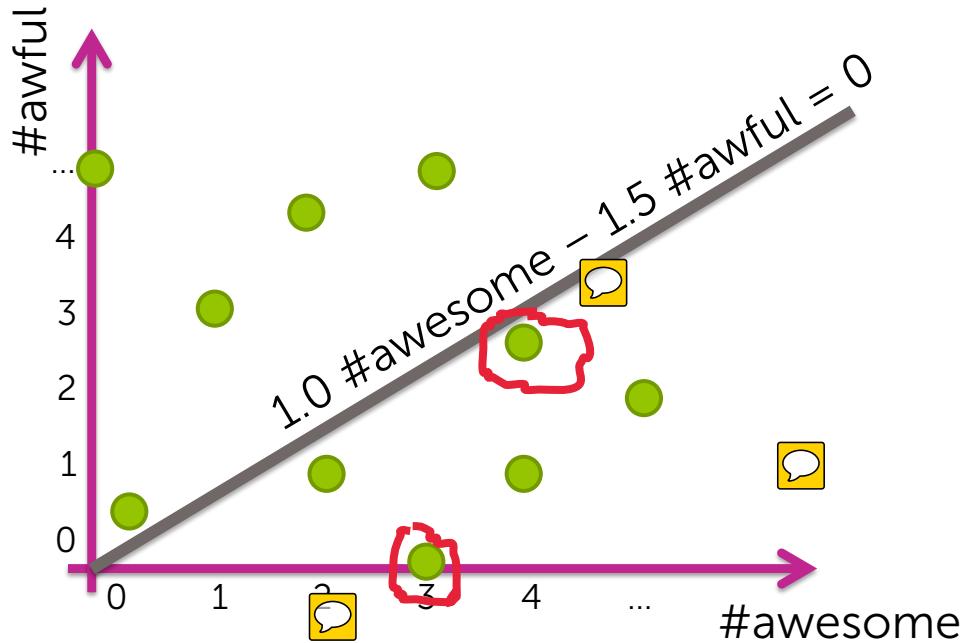
# Understanding the logistic regression model

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



Score( $\mathbf{x}_i$ )	$P(y=+1 \mathbf{x}_i, \mathbf{w})$
0	0.5
-2	0.12
2	0.88
4	0.98

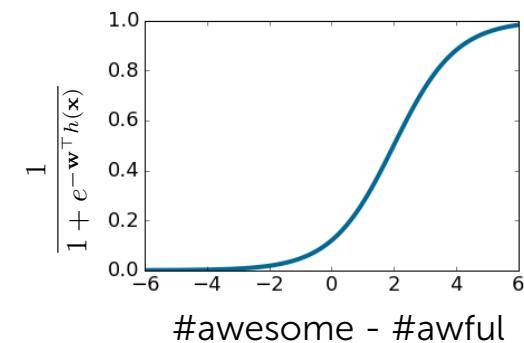
# Logistic regression → Linear decision boundary



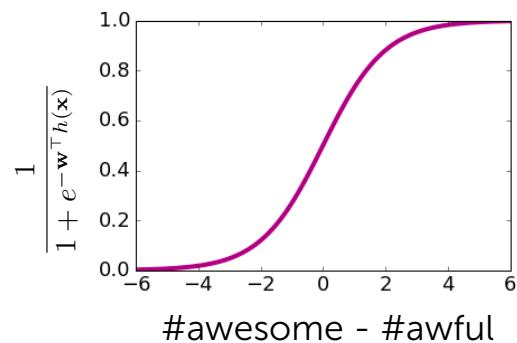
# Effect of coefficients on logistic regression model



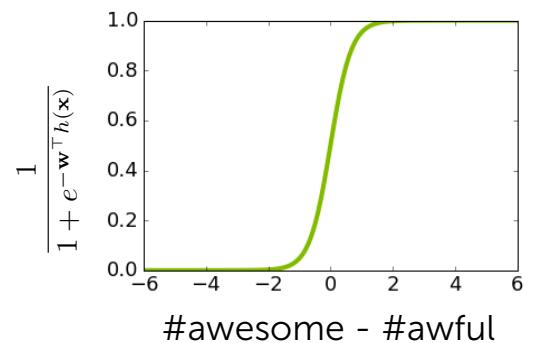
$w_0$	-2
$w_{\text{#awesome}}$	+1
$w_{\text{#awful}}$	-1



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

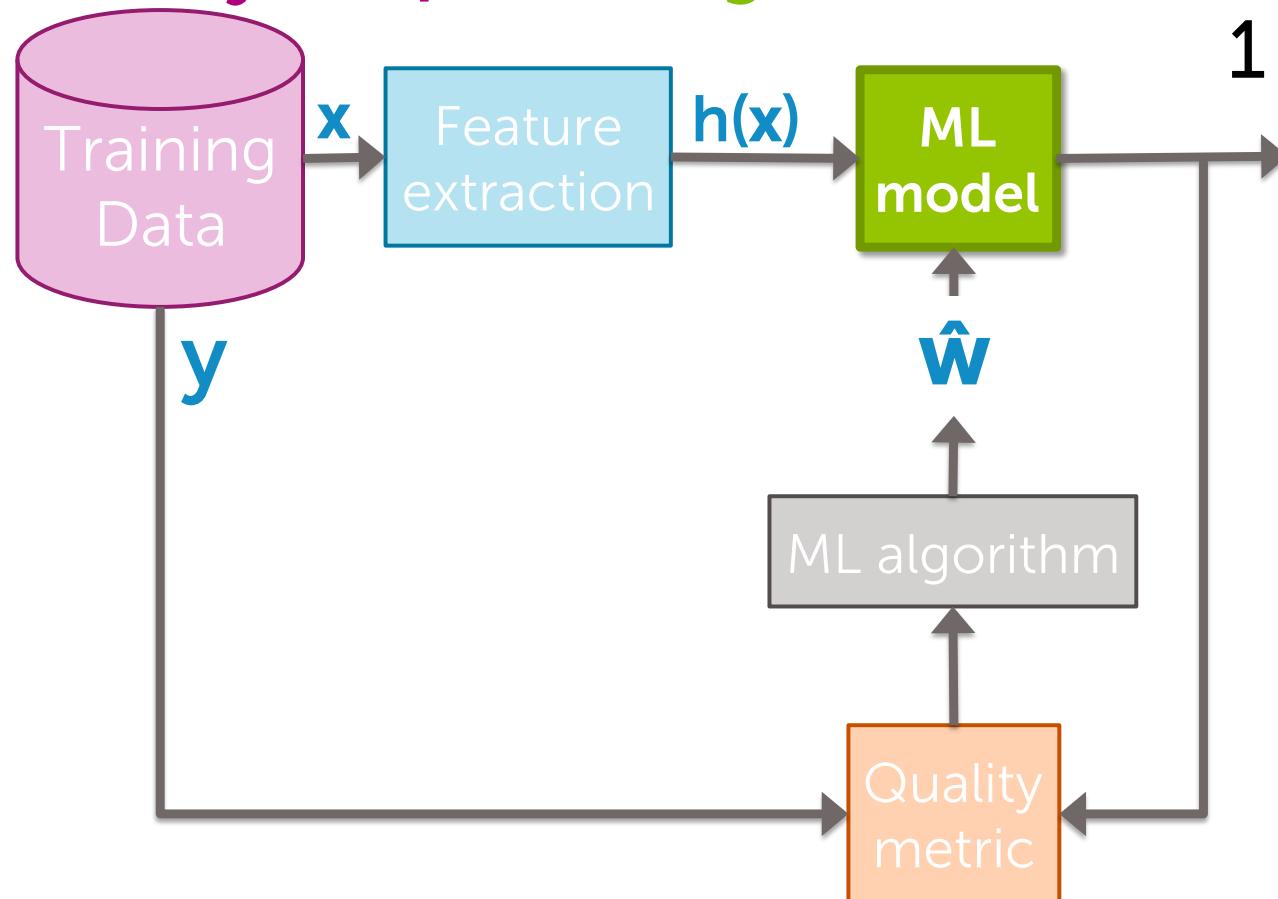


$w_0$	0
$w_{\text{#awesome}}$	+3
$w_{\text{#awful}}$	-3



---

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



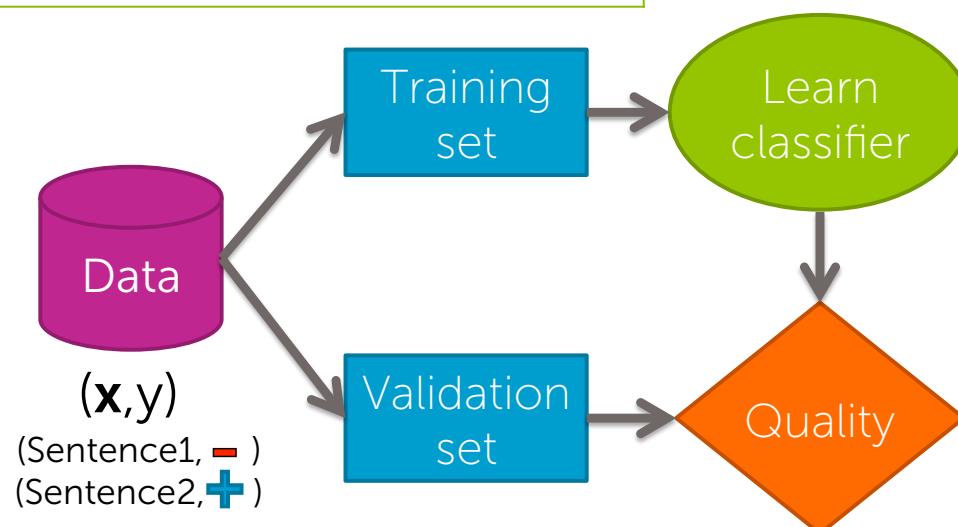
# Overview of learning logistic regression model



# Training a classifier = Learning the coefficients

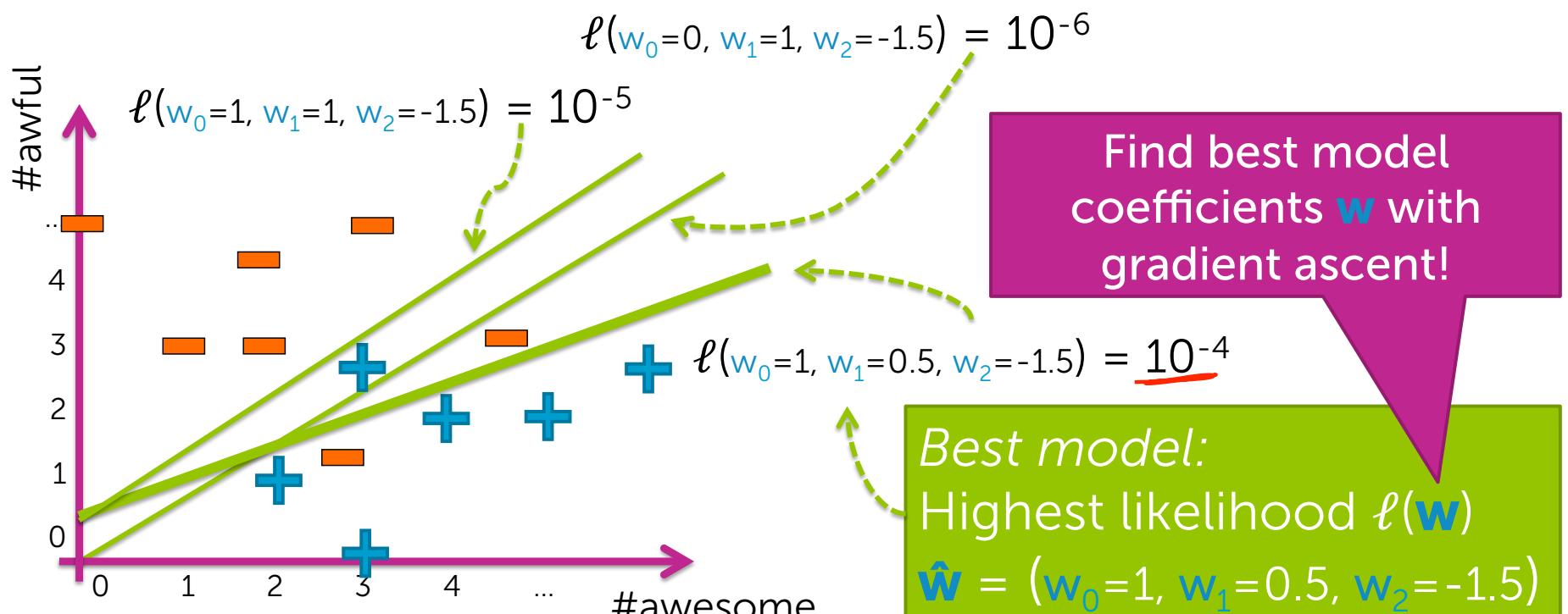
Word	Coefficient	Value
	$\hat{w}_0$	-2.0
good	$\hat{w}_1$	1.0
awesome	$\hat{w}_2$	1.7
bad	$\hat{w}_3$	-1.0
awful	$\hat{w}_4$	-3.3
...	...	...

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



Find “best” classifier =  
Maximize quality metric over all possible  $w_0, w_1, w_2$

Likelihood  $\ell(w)$

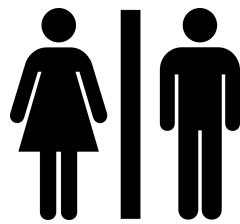


# Encoding categorical inputs

# Categorical inputs

- Numeric inputs:
  - #awesome, age, salary,...
  - Intuitive when multiplied by coefficient
    - e.g., **1.5 #awesome**

Numeric value, but should be interpreted as category  
(98195 not about 9x larger than 10005)



Gender  
(Male, Female,...)



Country of birth  
(Argentina, Brazil, USA,...)

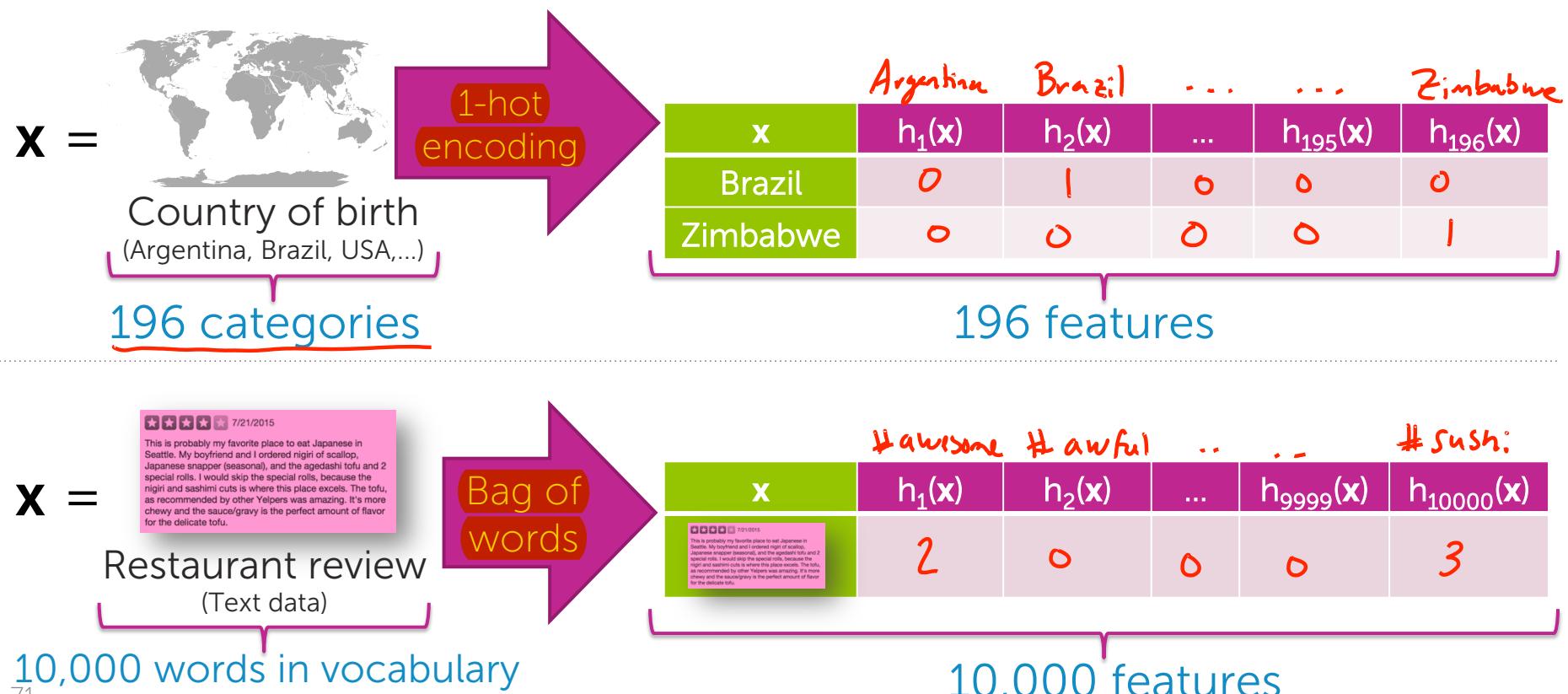


Zipcode  
(10005, 98195,...)

How do we multiply category by coefficient???

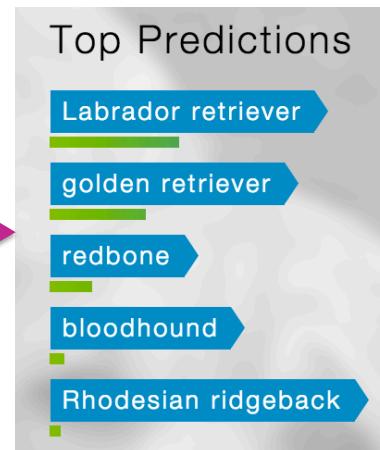
Must convert categorical inputs into numeric features

# Encoding categories as numeric features



# Multiclass classification using **1 versus all**

# Multiclass classification



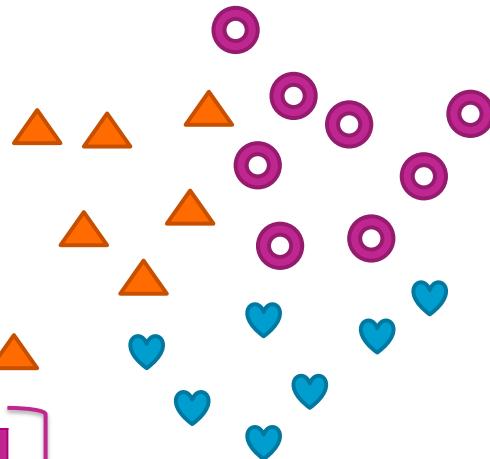
Input:  $x$   
Image pixels

Output:  $y$   
Object in image

# Multiclass classification formulation

- C possible classes:
  - $y$  can be 1, 2, ..., C
- N datapoints:

Data point	$x[1]$	$x[2]$	$y$
$\mathbf{x}_1, y_1$	2	1	▲
$\mathbf{x}_2, y_2$	0	2	♥
$\mathbf{x}_3, y_3$	3	3	○
$\mathbf{x}_4, y_4$	4	1	○



Learn:

$$\hat{P}(y=\blacktriangle | \mathbf{x})$$

$$\hat{P}(y=\heartsuit | \mathbf{x})$$

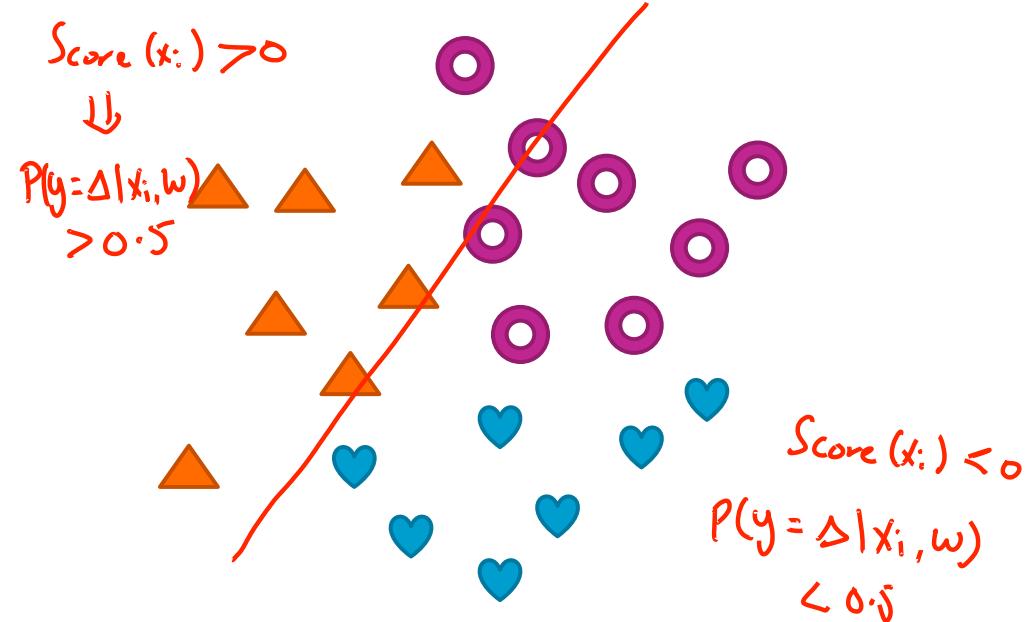
$$\hat{P}(y=\circlearrowleft | \mathbf{x})$$

# 1 versus all: Estimate $\hat{P}(y=\Delta|x)$ using 2-class model

+1 class: points with  $y_i = \Delta$   
-1 class: points with  $y_i = \text{blue heart} \text{ OR } \text{purple circle}$

Train classifier:  $\hat{P}(y=+1|x)$

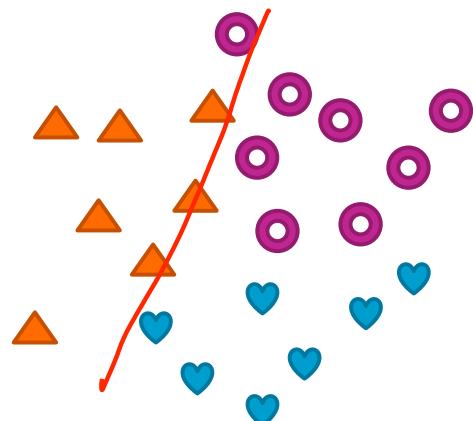
Predict:  $\hat{P}(y=\Delta|x_i) = \hat{P}(y=+1|x_i)$



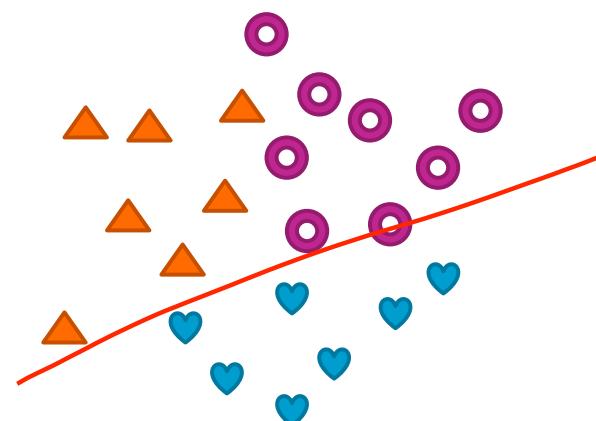
# 1 versus all: simple multiclass classification using C 2-class models



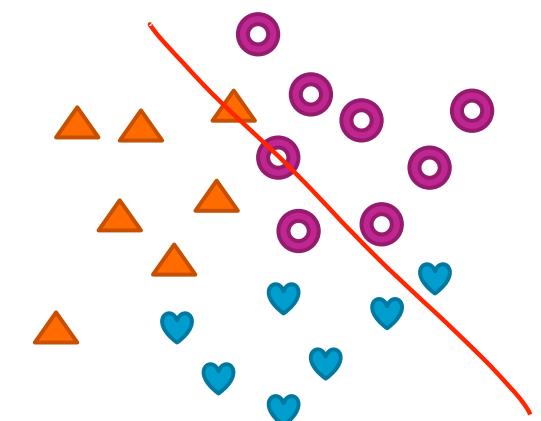
$$\hat{P}(y=\triangle | \mathbf{x}_i) = \hat{P}_{\alpha}(y=+1 | \mathbf{x}_i, w)$$



$$\hat{P}(y=\heartsuit | \mathbf{x}_i) = \hat{P}_{\beta}(y=+1 | \mathbf{x}_i, w)$$



$$\hat{P}(y=\circlearrowleft | \mathbf{x}_i) = \hat{P}_{\theta}(y=+1 | \mathbf{x}_i, w)$$



## Multiclass training

$\hat{P}_c(y=+1|\mathbf{x})$  = estimate of  
1 vs all model for each class



Input:  $\mathbf{x}_i$

Predict most likely class

`max_prob = 0;  $\hat{y} = 0$`

For  $c = 1, \dots, C$ :

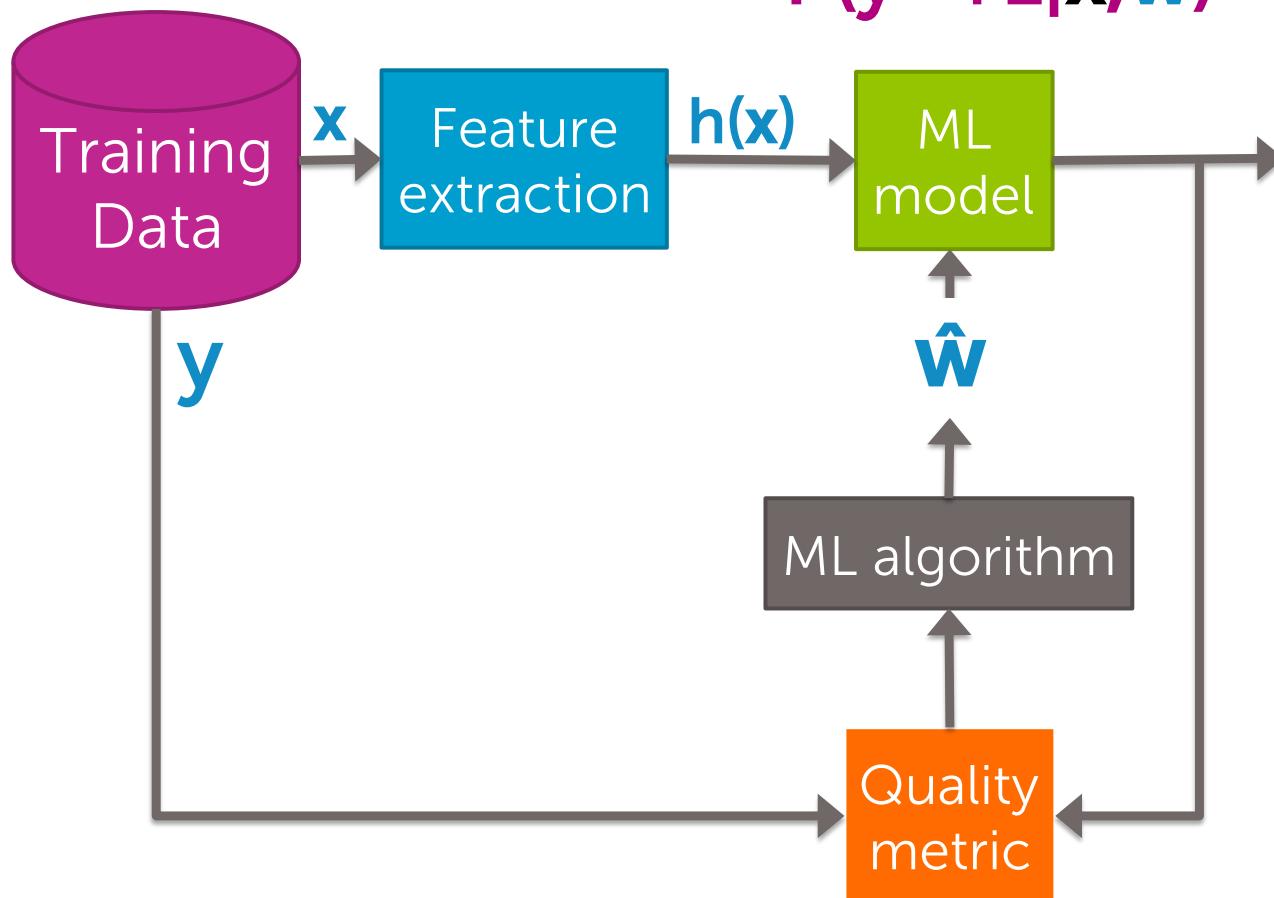
If  $\hat{P}_c(y=+1|\mathbf{x}_i) > max\_prob$ :

$\hat{y} = c$

`max_prob =  $\hat{P}_c(y=+1|\mathbf{x}_i)$`



# Summary of logistic regression classifier





# What you can do now...

- Describe decision boundaries and linear classifiers
- Use class probability to express degree of confidence in prediction
- Define a logistic regression model
- Interpret logistic regression outputs as class probabilities
- Describe impact of coefficient values on logistic regression output
- Use 1-hot encoding to represent categorical inputs
- Perform multiclass classification using the 1-versus-all approach