Classification:
Analyzing Sentiment

STAT/CSE 416: Machine Learning
Emily Fox
University of Washington
April 17, 2018

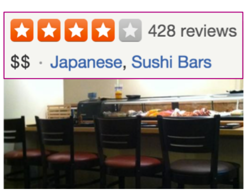
Predicting sentiment by topic:
An intelligent restaurant review system

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.

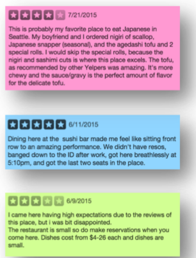
My wife tried their ramen and it was pretty forgettable.

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

Experience

From reviews to topic sentiments

All reviews
for restaurant



Novel intelligent
restaurant review app

Experience
★★★★

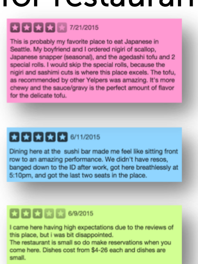
Ramen
★★★

Sushi
★★★★★

Easily best <u>sushi</u>
in Seattle.

Intelligent restaurant review system

All reviews
for restaurant



Break all reviews
into sentences

The seaweed salad was just OK,
vegetable salad was just ordinary.

I like the interior decoration and
the blackboard menu on the wall.

All the sushi was delicious.

My wife tried their ramen and
it was pretty forgettable.

The sushi was amazing, and
the rice is just outstanding.

The service is somewhat hectic.

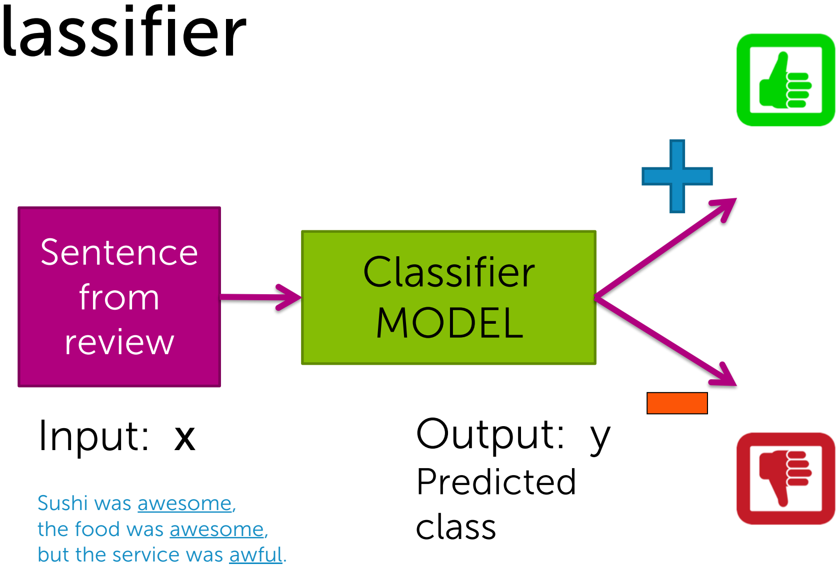
Easily best sushi in Seattle.

Core building block

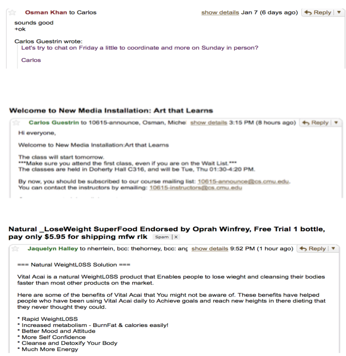
Intelligent restaurant review system

Classifier applications

Classifier



Spam filtering



Text of email, sender, IP,…

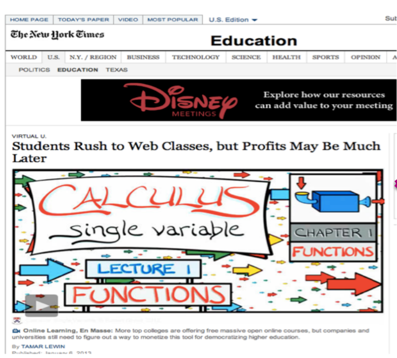
Not spam

Spam

Input: **x** Output: y

Multiclass classifier

Output y has more than 2 categories



Education

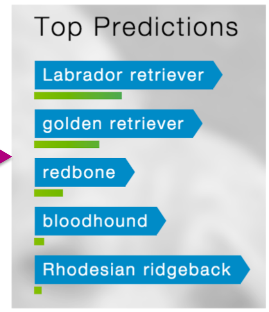
Finance

Technology

Input: **x**
Webpage

Output: y

Image classification



Input: x
Image pixels

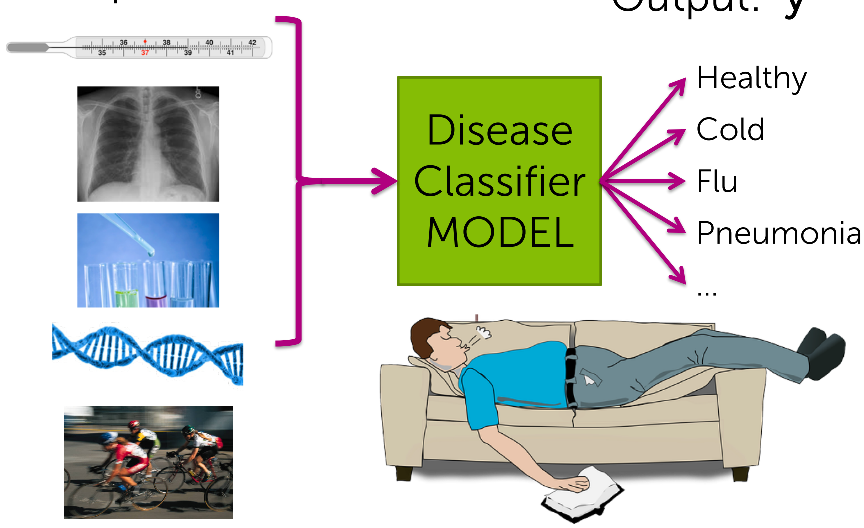
Output: y
Predicted object

13 ©2018 Emily Fox STAT/CSE 416: Intro to Machine Learning

Personalized medical diagnosis

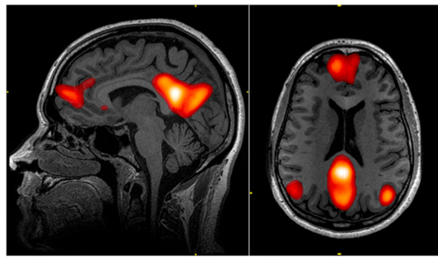
Input: x

Output: **y**



14 ©2018 Emily Fox STAT/CSE 416: Intro to Machine Learning

Reading your mind



Output y

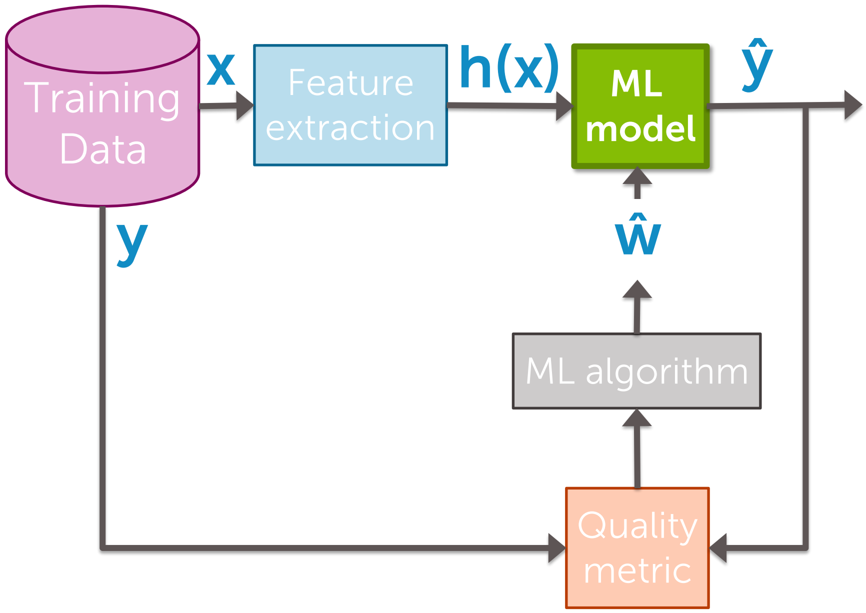
"Hammer"

"House"

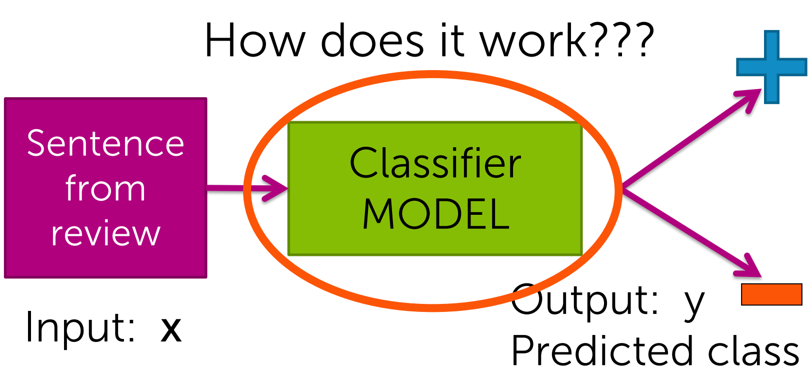
Inputs x are brain region intensities

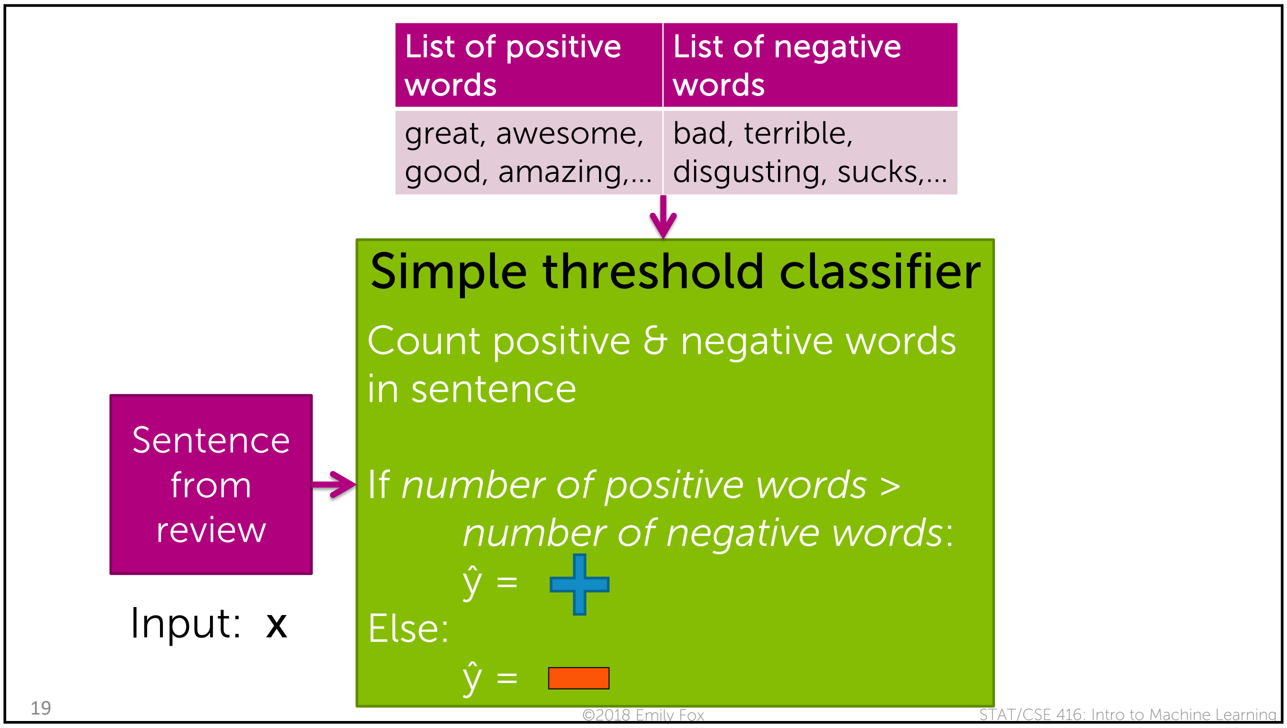
Linear classifiers

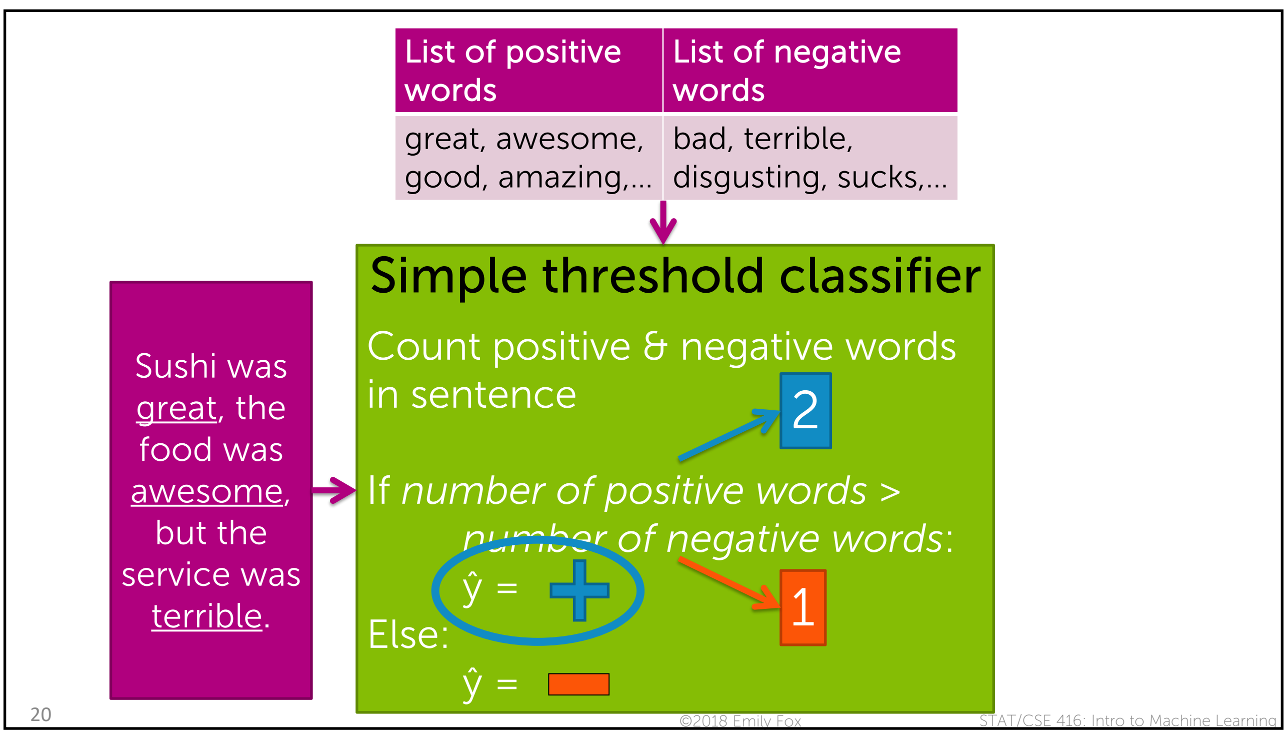
©2018 Emily Fox
STAT/CSE 416: Intro to Machine Learning



Representing classifiers




List of positive words
great, awesome, good, amazing,…
List of negative words
bad, terrible, disgusting, sucks,…
Simple threshold classifier
Count positive & negative words in sentence
If number of positive words > number of negative words:
ŷ =
Else:
ŷ =
Sentence from review
Input: x
19
©2018 Emily Fox
STAT/CSE 416: Intro to Machine Learning


List of positive words
great, awesome, good, amazing,…
List of negative words
bad, terrible, disgusting, sucks,…
Simple threshold classifier
Count positive & negative words in sentence
2
If number of positive words > number of negative words:
ŷ =
1
Else:
ŷ =
Sushi was great, the food was awesome, but the service was terrible.
20
©2018 Emily Fox
STAT/CSE 416: Intro to Machine Learning

Problems with threshold classifier

- How do we get list of positive/negative words?
- Words have different degrees of sentiment:
  - Great > good
  - How do we weigh different words?

Addressed by learning a classifier

- Single words are not enough:
  - *Good* ➔ Positive
  - *Not good* ➔ Negative

Addressed by more elaborate features

A (linear) classifier

Will use training data to learn a weight for each word

Word	Weight
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	Weight
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 x_i:
Sushi was great,
the food was awesome,
but the service was terrible.

count of "great"

$$\text{Score}(x_i) = 1.2\,(1) + 1.7(1) - 2.1(1)$$
$$= 0.8 > 0$$
$$\Rightarrow \hat{y}_i = +1$$

positive review

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

Word	Weight
…	…

Sentence from review

Input: **x**

Simple linear classifier

Score(x) = weighted count of words in sentence

If *Score (x) > 0*:
ŷ = +
Else:
ŷ = –

More generically…

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

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

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

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

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

feature 3 = $h_2(\mathbf{x})$ … e.g., $\mathbf{x}[2]$ = #awful

or, $\log(\mathbf{x}[7])\ \mathbf{x}[2]$ = log(#bad) x #awful

or, tf-idf("awful")

…

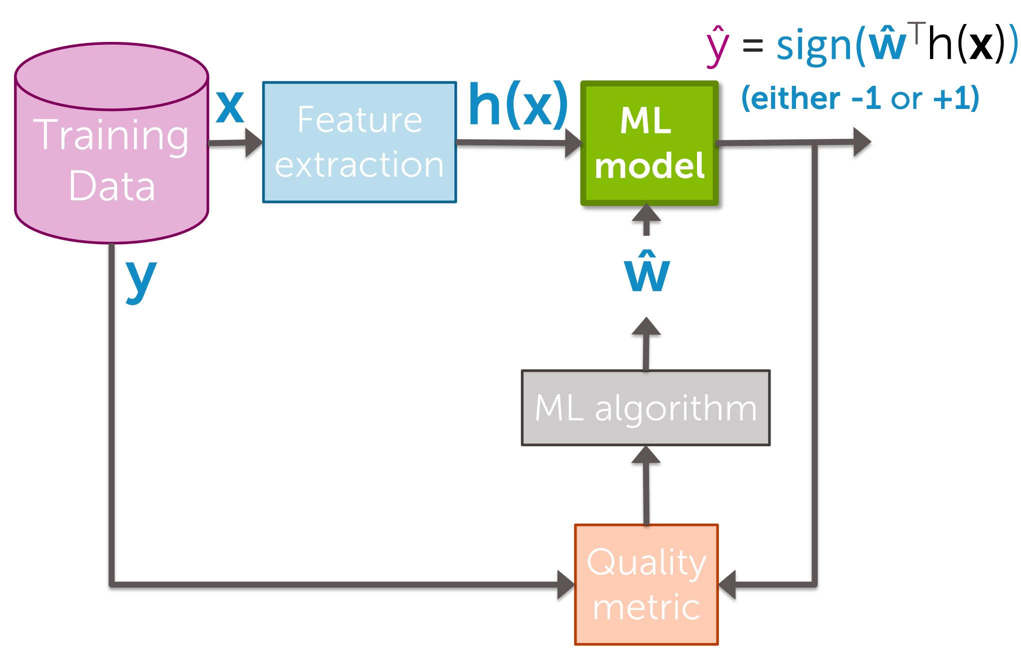
feature D+1 = $h_D(\mathbf{x})$ … some other function of $\mathbf{x}[1]$,…, $\mathbf{x}[d]$

sign(pos. #) = +1

sign(neg. #) = −1

sign(0.8) = +1

sign(−1,627,000) = −1

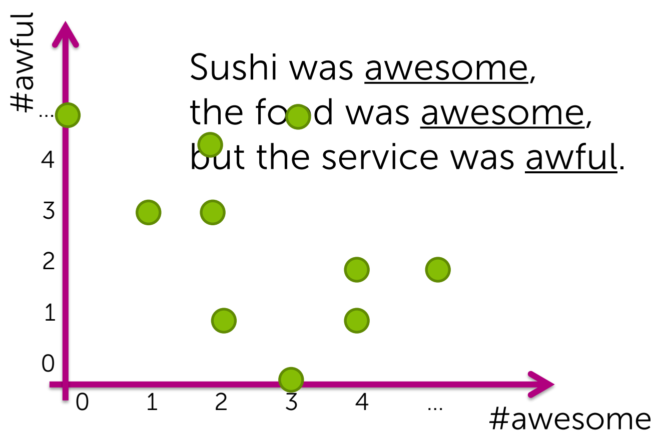


Decision boundaries

Suppose only two words had non-zero coefficient

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

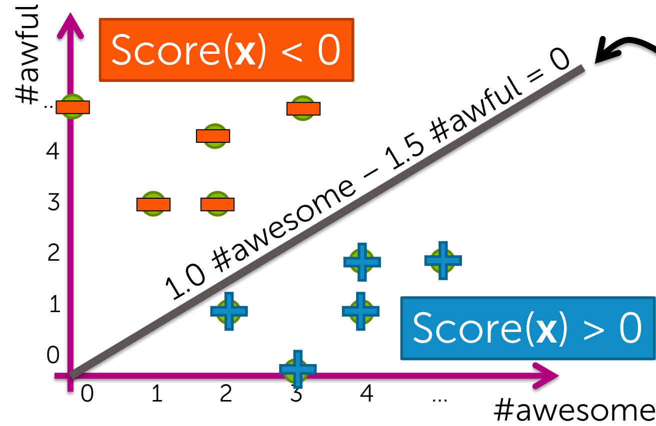
Score(x) = 1.0 #awesome – 1.5 #awful



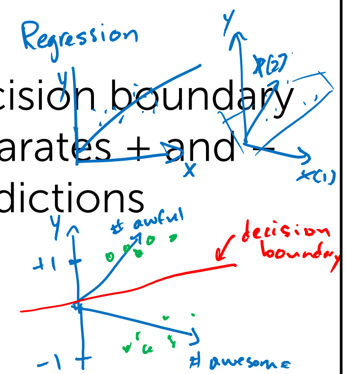
Decision boundary example

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

Score(x) = 1.0 #awesome – 1.5 #awful



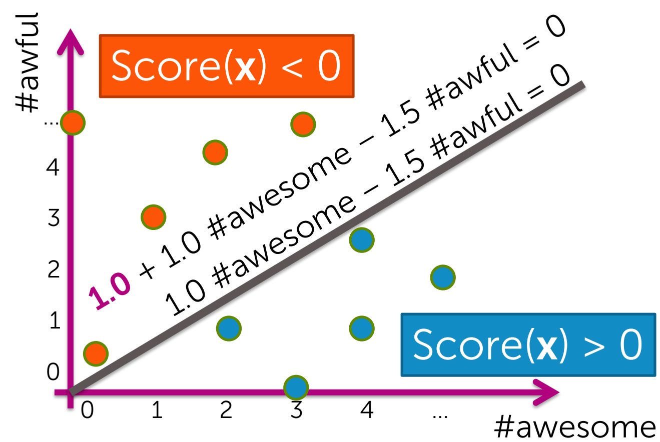
Decision boundary separates + and – predictions



Decision boundary: effect of changing coefficients

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

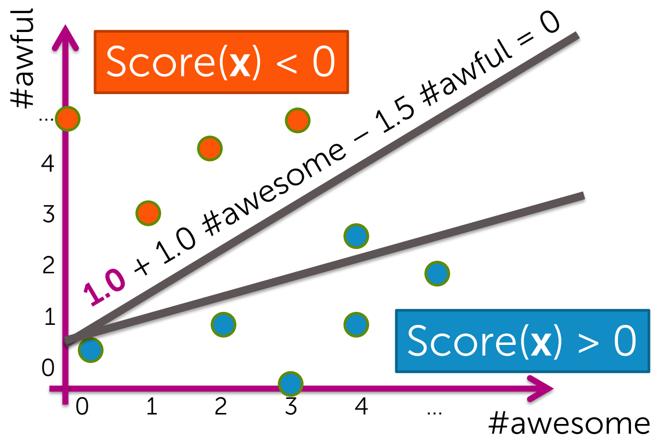
Score(x) = **1.0** #awesome – 1.5 #awful



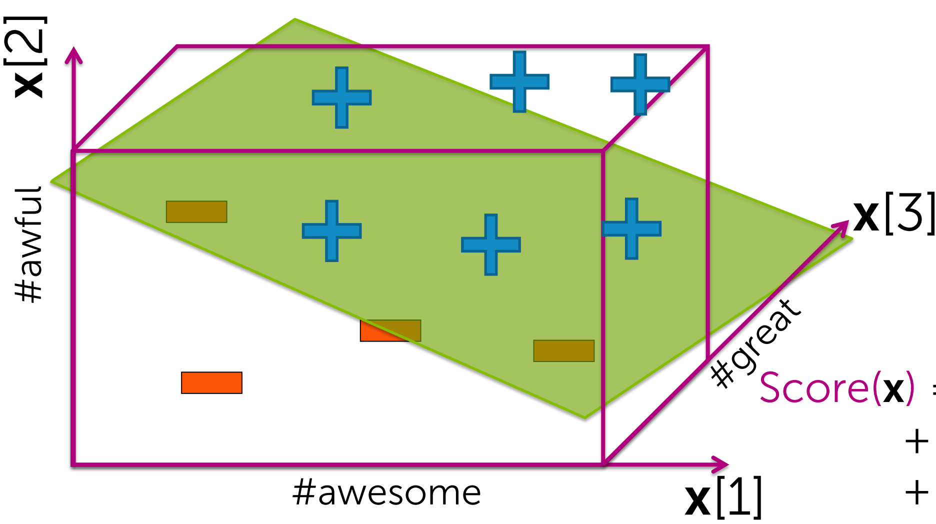
Decision boundary: effect of changing coefficients

Input	Coefficient	Value
	w_0	**1.0**
#awesome	w_1	1.0
#awful	w_2	**–3.0**

Score(x) = **1.0** + 1.0 #awesome **– 3.0** #awful



For more inputs (linear features)…

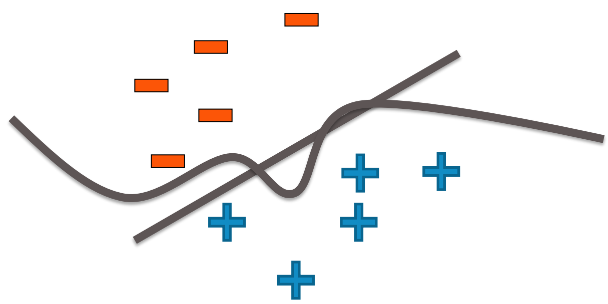


Score(**x**) = w_0
+ w_1 #awesome
+ w_2 #awful
+ w_3 #great

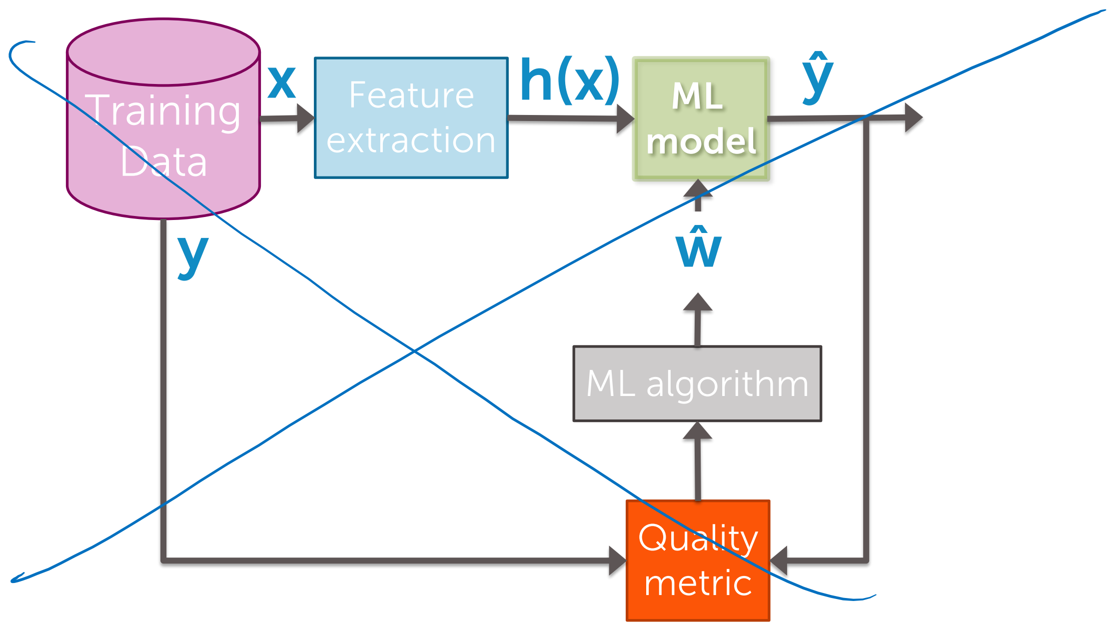
For general features…

For more general classifiers (not just linear features)

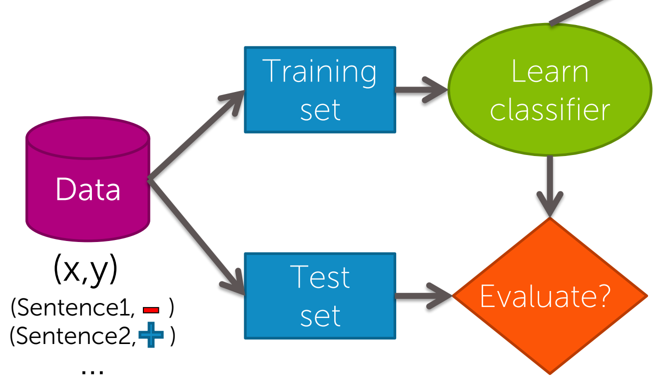
➔ more complicated shapes



~~Training and~~ evaluating a classifier



Training a classifier = Learning the weights



Word	Weight
good	1.0
awesome	1.7
bad	-1.0
awful	-3.3
...	...

Classification error

Learned classifier

ŷ = +

Test example

(Sushi was great, +)

Correct!

Correct	0
Mistakes	0

Hide label

Classification error & accuracy

- Error measures fraction of mistakes

$$\text{error} = \frac{\#\text{ of mistakes}}{\text{Total }\#\text{ of sentences}}$$

test set

  – Best possible value is 0.0

error = 1 - accuracy

- Often, measure **accuracy**
  – Fraction of correct predictions

$$\text{accuracy} = \frac{\#\text{ of correct}}{\text{Total }\#\text{ of sentences}}$$

  – Best possible value is 1.0

What's a good accuracy?

What if you ignore the sentence, and just guess?

- For binary classification:
  - Half the time, you'll get it right! (on average)
    ➔ accuracy = 0.5
- For k classes, accuracy = 1/k
  - 0.333 for 3 classes, 0.25 for 4 classes,…

At the very, very, very least,
you should healthily beat random…
Otherwise, it's (usually) pointless…

Is a classifier with 90% accuracy good? Depends…

2010 data shows:
"90% emails sent are spam!"

Predicting every email is spam gets you 90% accuracy!!!

Majority class prediction

Amazing performance when there is class imbalance
(but silly approach)

- One class is more common than others
- Beats random (if you know the majority class)

So, always be digging in and asking the hard questions about reported accuracies

- Is there class imbalance?
- How does it compare to a simple, baseline approach?
  - Random guessing
  - Majority class
  - …
- Most important:
  What accuracy does my application need?
  - What is good enough for my user's experience?
  - What is the impact of the mistakes we make?

False positives, false negatives, and confusion matrices

Types of mistakes

Predicted label

True label	+	−
+	True Positive	False Negative (FN)
−	False Positive (FP)	True Negative

Cost of different types of mistakes can be different (& high) in some applications

	Spam filtering	Medical diagnosis
False negative	Annoying	Disease not treated
		depends
False positive	Email lost higher cost	Wasteful treatment

Confusion matrix – binary classification

100 test examples

Predicted label

True label	+	−
+ 60	50	10
− 40	5	35

$$\text{accuracy} = \frac{50+35}{100} = 0.85$$

Confusion matrix – multiclass classification

100 test examples

Predicted label

True label	Healthy	Cold	Flu
Healthy 70	60	8	2
Cold 20	4	12	4
Flu 10	0	2	8

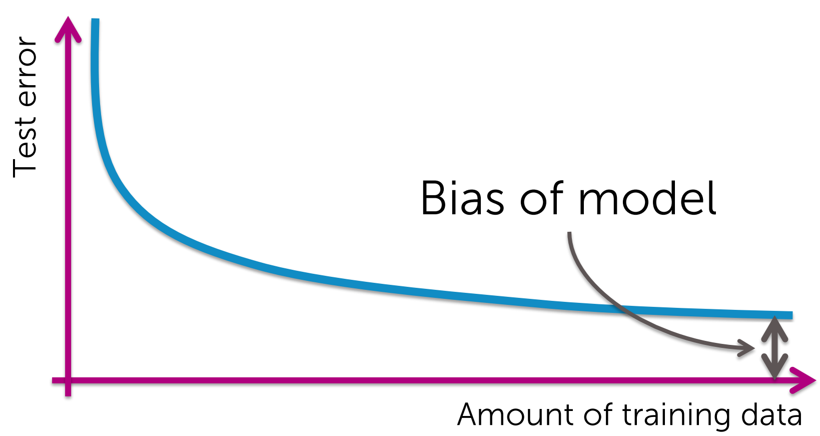
$$\text{Accuracy} = \frac{80}{100} = 0.8$$

Learning curves (again):
How much data do I need?

How much data does a model need to learn?

- The more the merrier ☺
  - But data quality is most important factor
- Theoretical techniques can sometimes bound how much data is needed
  - Typically too loose for practical application
  - But provide guidance
- In practice:
  - **More complex models require more data**
  - Empirical analysis can provide guidance

Test error vs. amount of training data



even with infinite data, test error will not go to 0

More complex models tend to have less bias…

Sentiment classifier using single words can do OK, but…

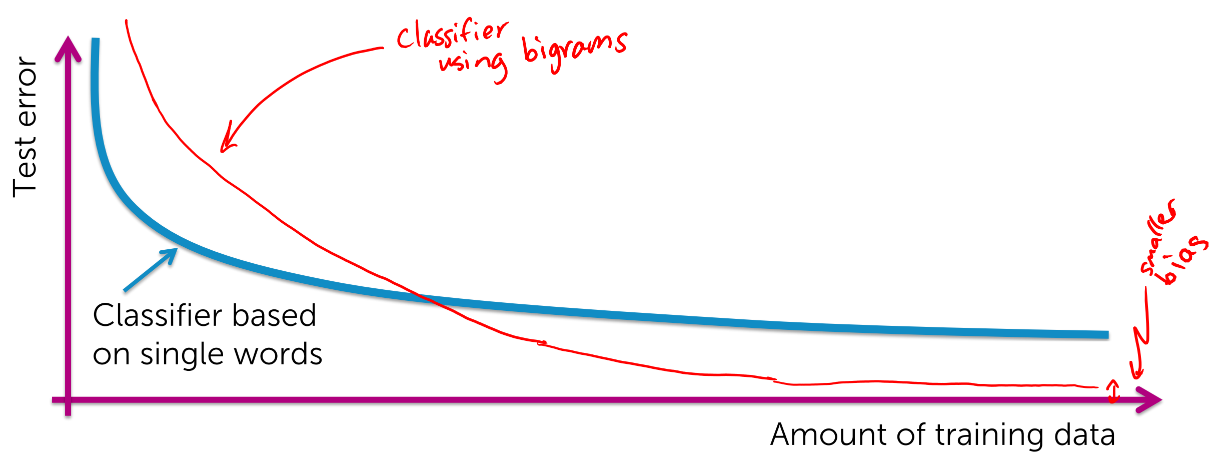
Never classifies correctly: "The sushi was not good."

More complex model: consider pairs of words (bigrams)

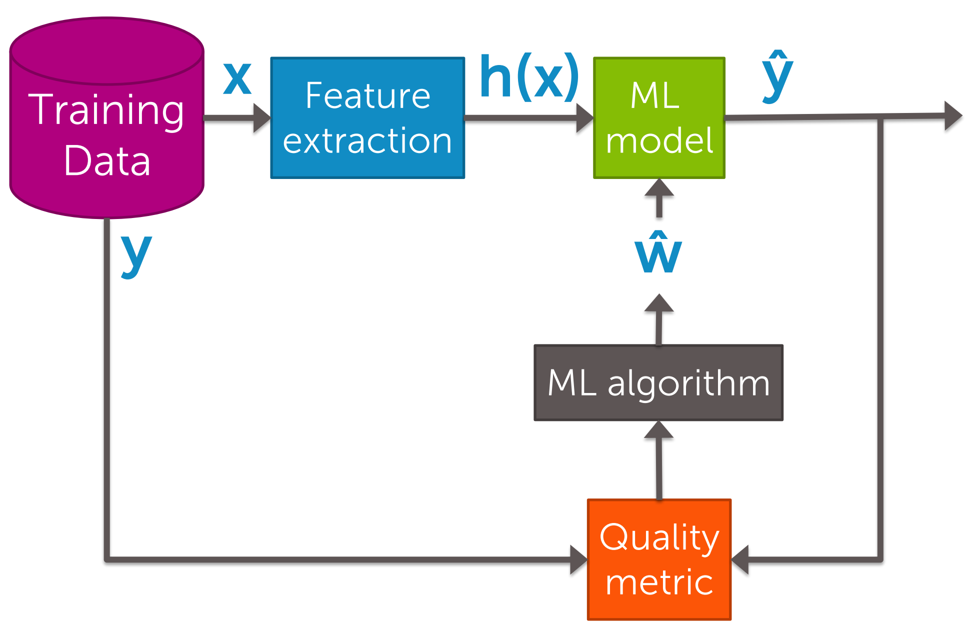
Word	Weight
good	+1.5
not good	-2.1

Less bias → potentially more accurate, needs more data to learn

Models with less bias tend to need more data to learn well, but do better with sufficient data



Summary of classification intro



What you can do now…

- Identify a classification problem and some common applications
- Describe decision boundaries and linear classifiers
- Train a classifier
- Measure its error
  - Some rules of thumb for good accuracy
- Interpret the types of error associated with classification
- Describe the tradeoffs between model bias and data set size

Linear classifiers: Logistic regression

STAT/CSE 416: Machine Learning
Emily Fox
University of Washington
April 17, 2018

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?

"The sushi & everything else were awesome!"

Definite **+1**

"The sushi was good, the service was OK"

Not sure

Using probabilities in classification

How confident is your prediction?

"The sushi & everything else were awesome!"

Definite +1

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

"The sushi was good, the service was OK"

Not sure

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

Many classifiers provide a degree of certainty:

Output label → $P(y|\mathbf{x})$ ← Input sentence

Extremely useful in practice

Goal: Learn conditional probabilities from data

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

x[1] = #awesome	x[2] = #awful	y = 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 $\hat{\mathbf{w}}$

Useful for predicting ŷ

Predict most likely class

Sentence from review

Input: **x**

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

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

ŷ = **+1**

Else:

ŷ = **-1**

Estimating $\hat{P}(y|x)$ improves **interpretability**:

– Predict ŷ = +1 **and** tell me how sure you are