INTRODUCTION TO DATA SCIENCE

This lecture is
based on course by E. Fox and C. Guestrin, Univ of Washington

Classification

- **An inteligent restaurant review system**

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

What is a sentiment of the review

Positive reviews not positive about everything

Experience

Sample review:

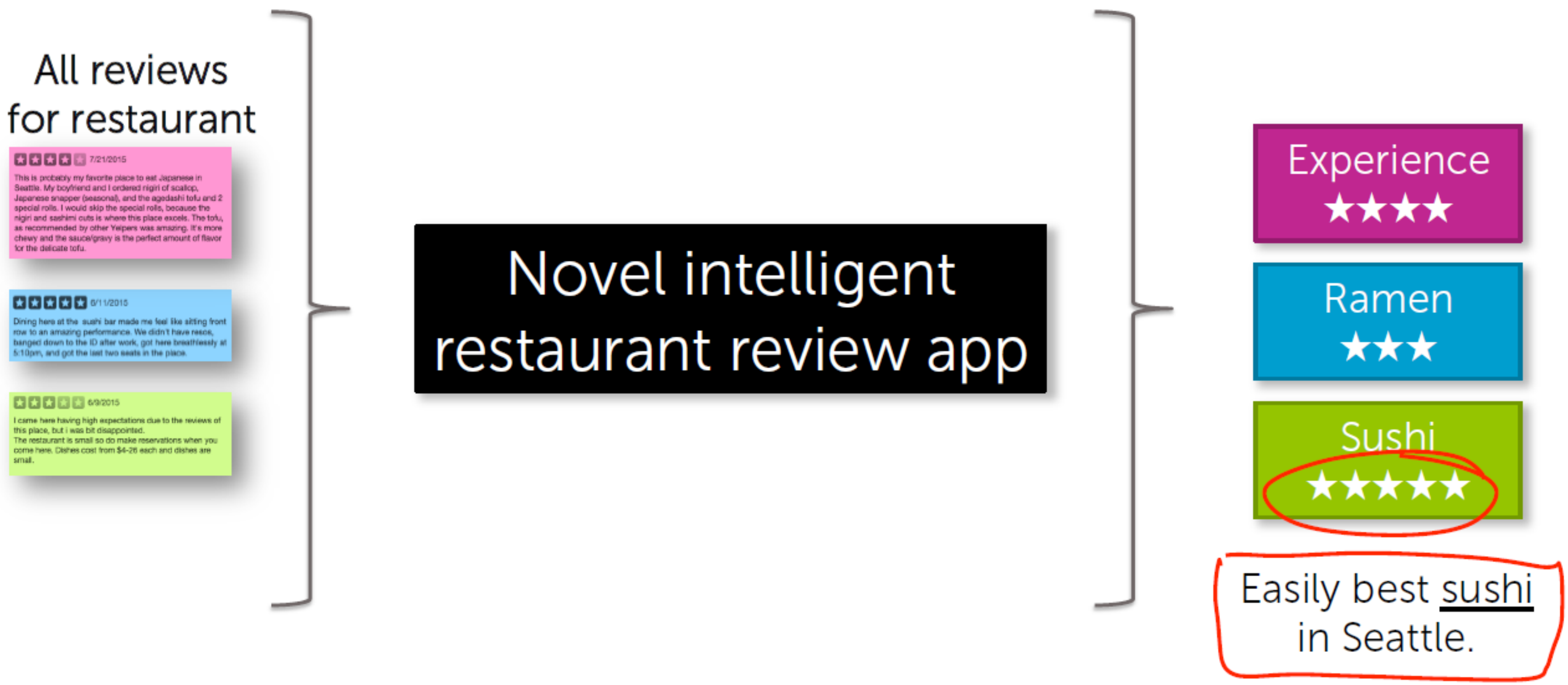
Watching the chefs create incredible edible art made the experience very unique.

My wife tried their ramen and it was pretty forgettable.

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

Topic sentiments

From reviews to topic sentiments



Inteligent 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.

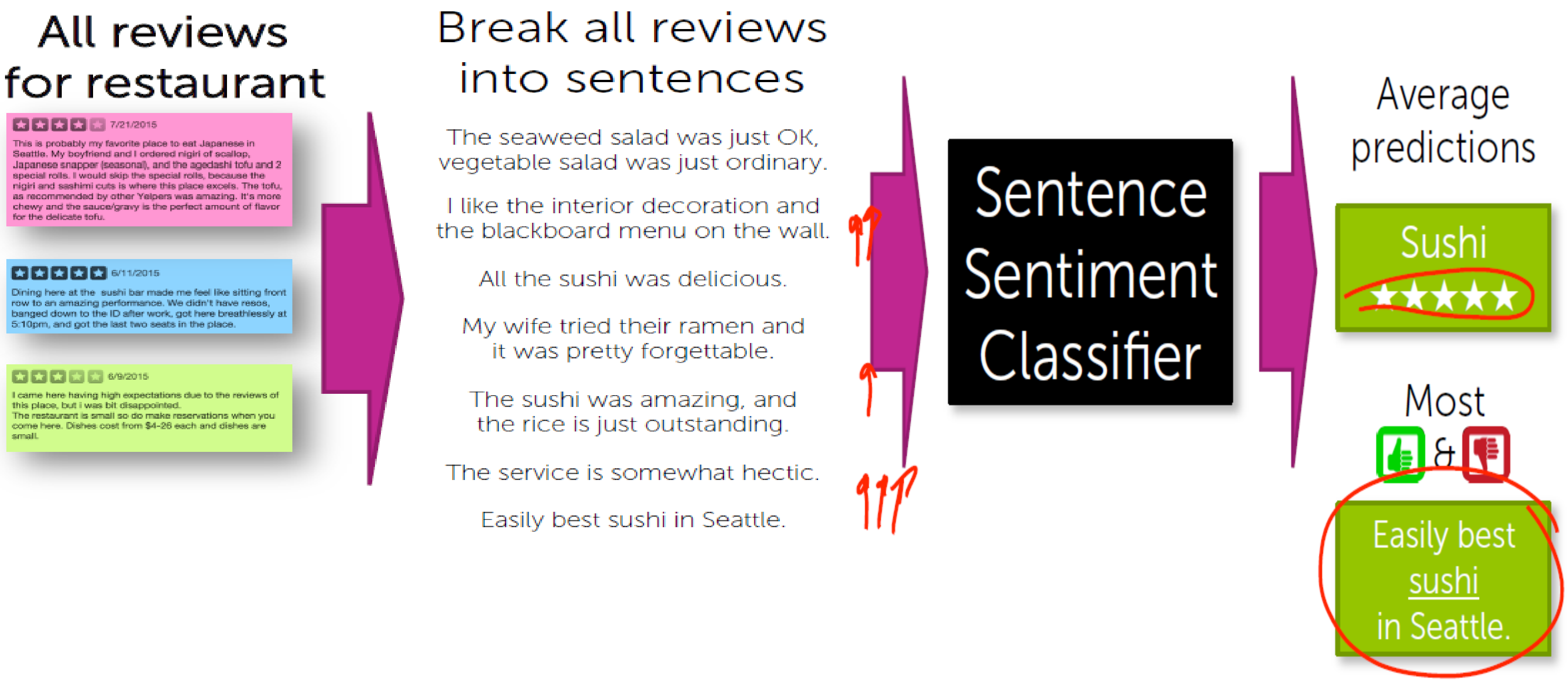
The sushi was amazing, and the rice is just outstanding.

The service is somewhat hectic.

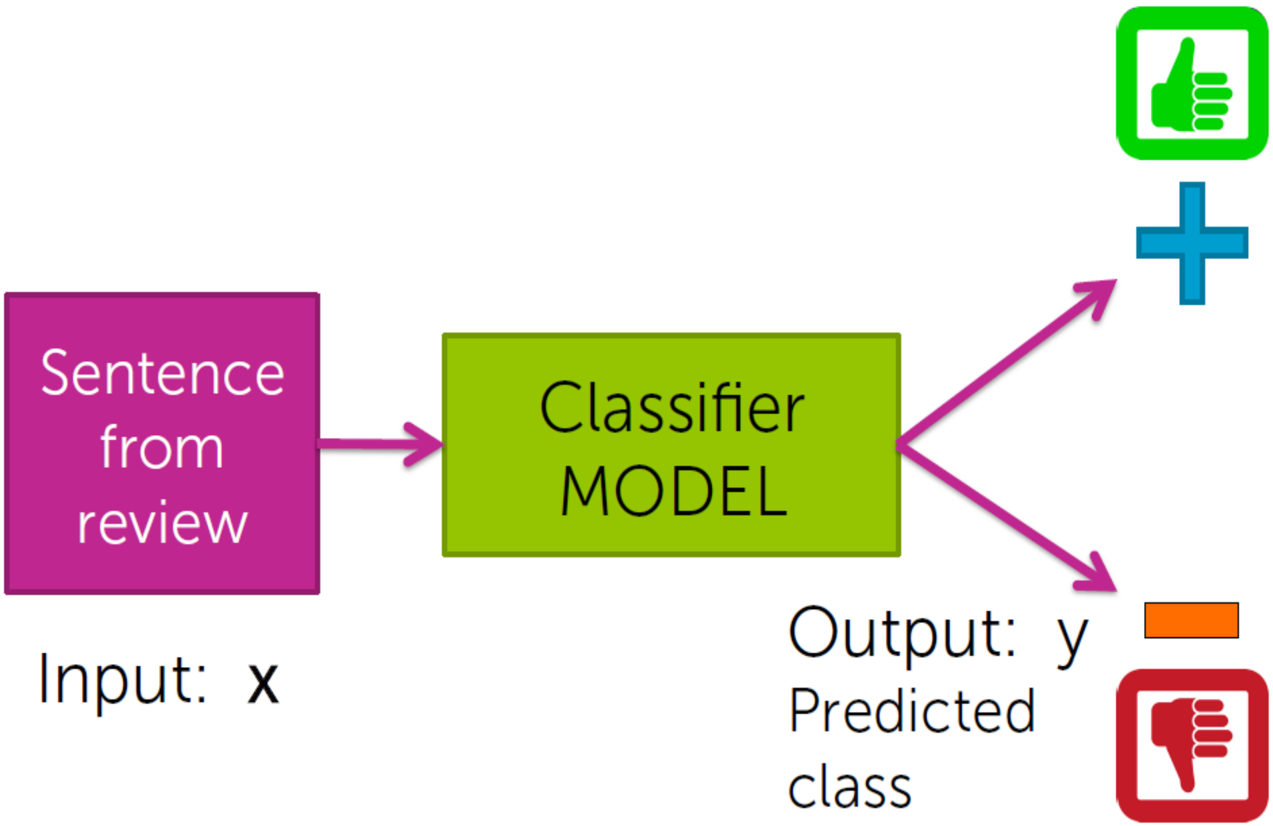
Easily best sushi in Seattle.

Core building block

Inteligent restaurant review system

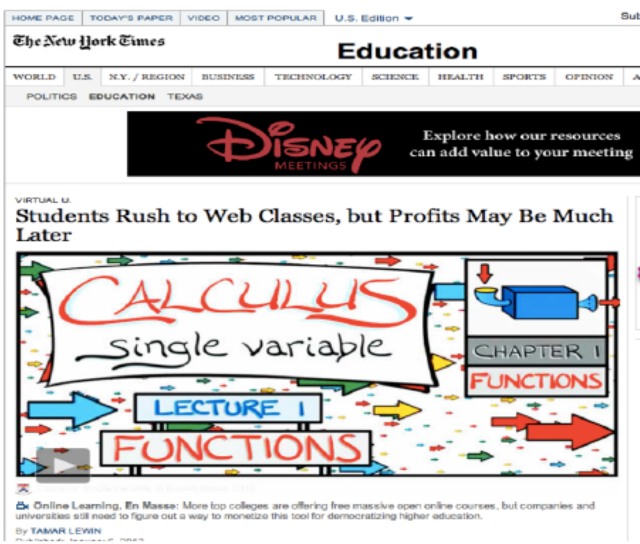


Classifier


Sentence from review
Classifier MODEL
Input: x
Output: y
Predicted class

Multiclass classifier

Output y has more than 2 categories



Education

Finance

Technology

Input: x
Webpage

Output: y

Spam filtering

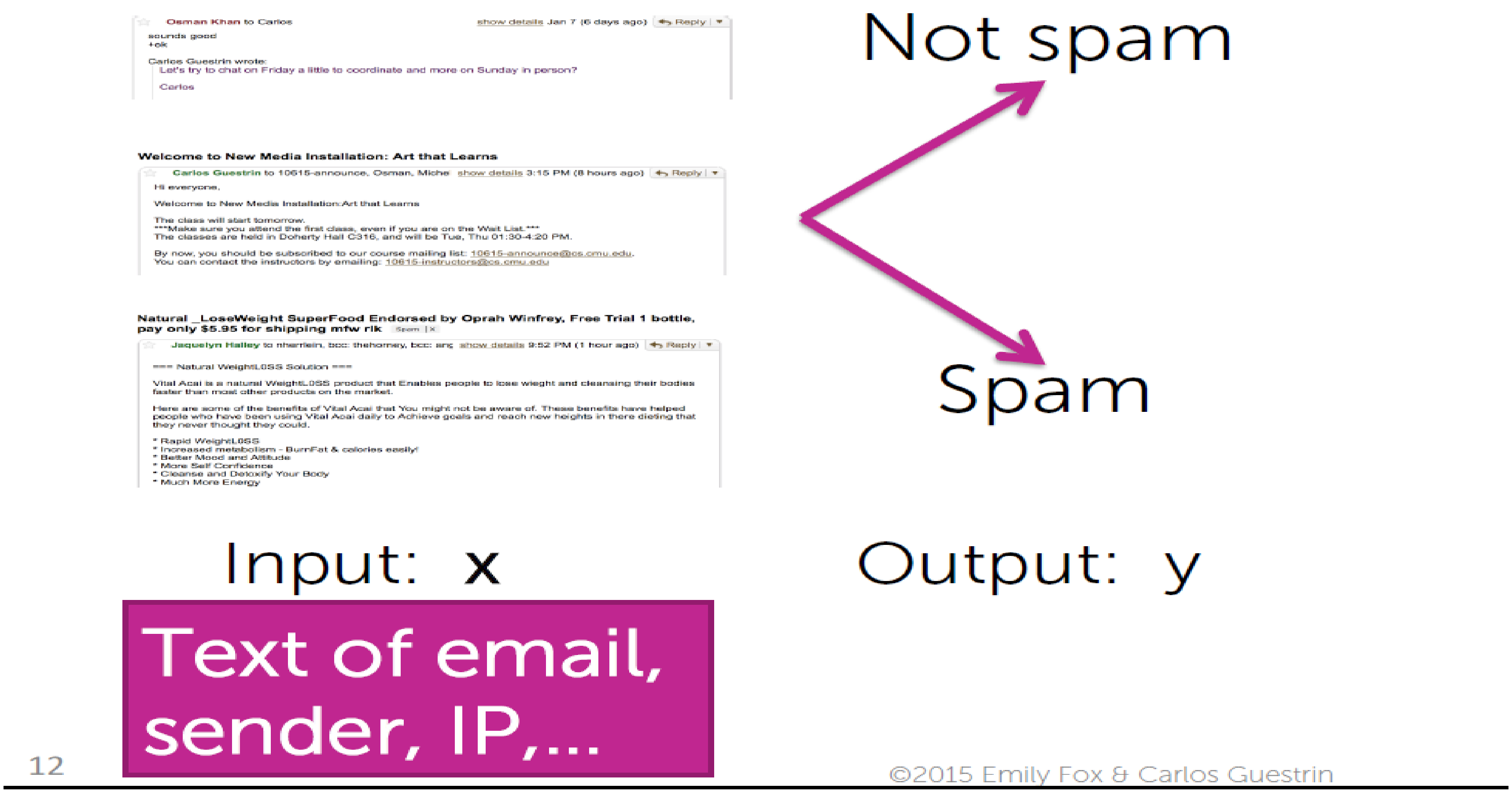
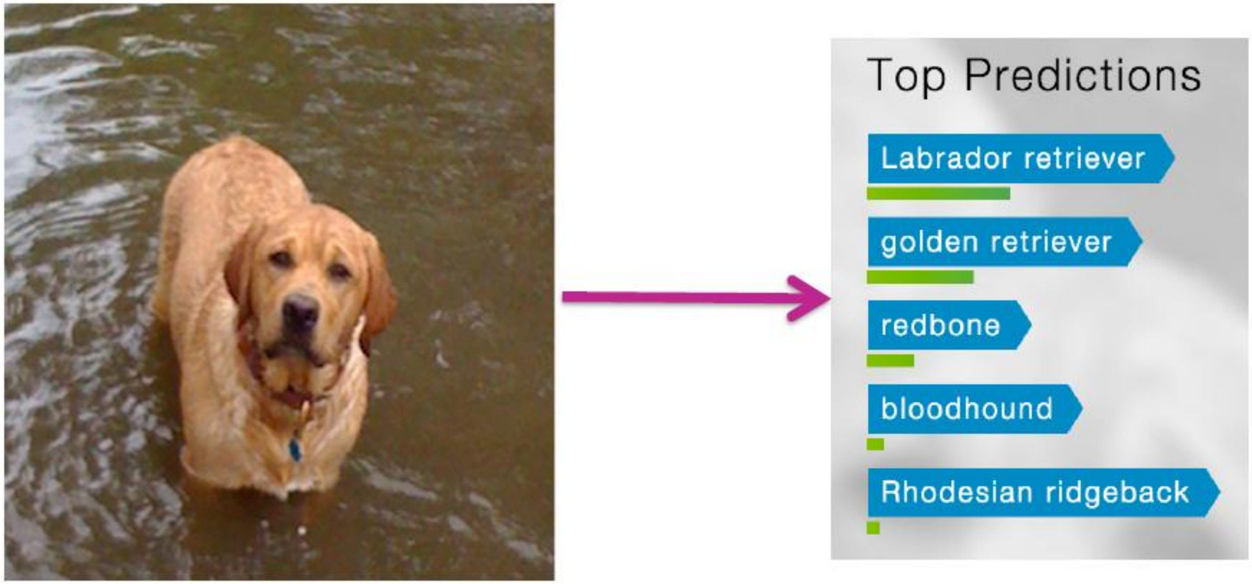


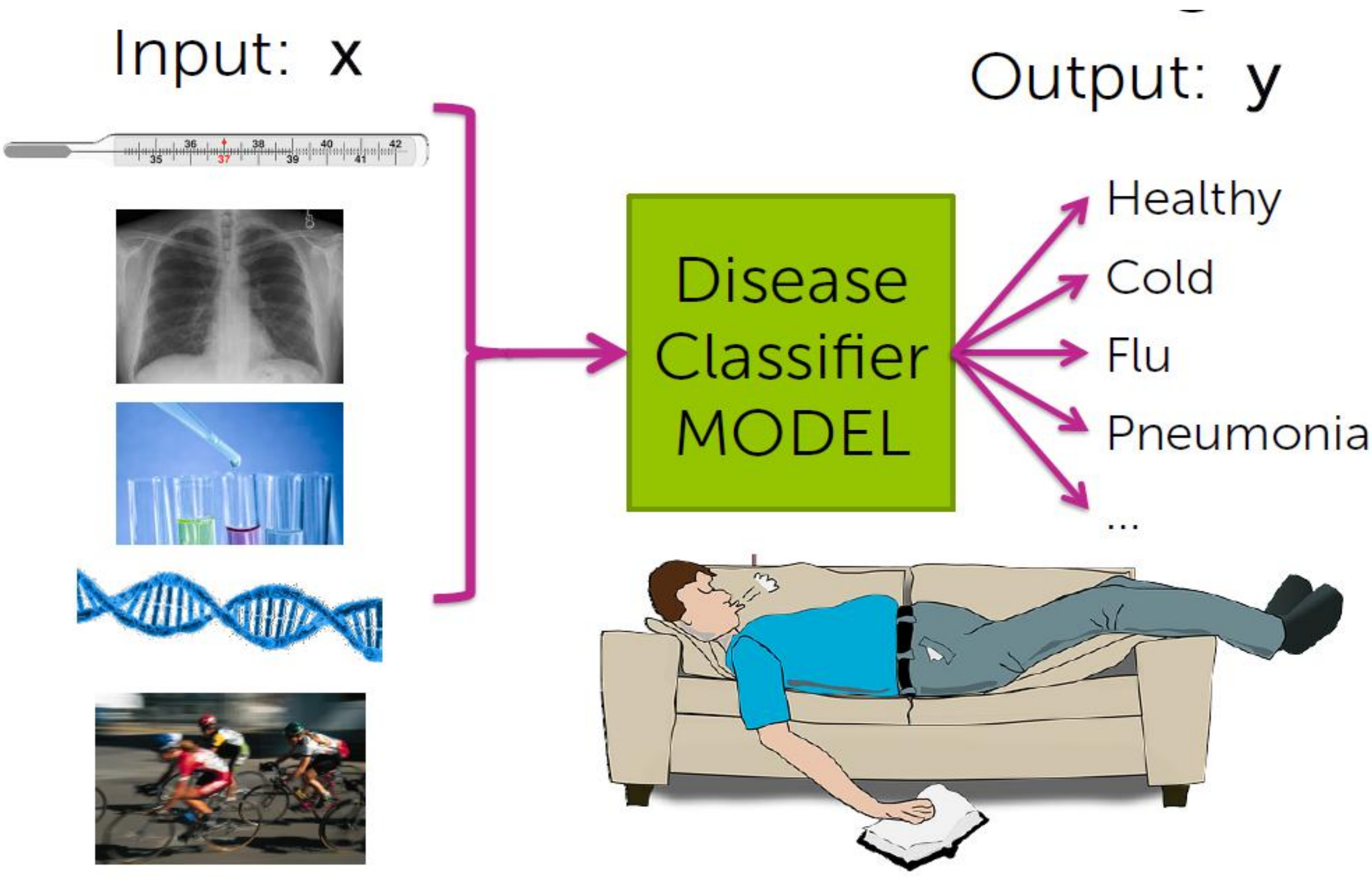
Image classification



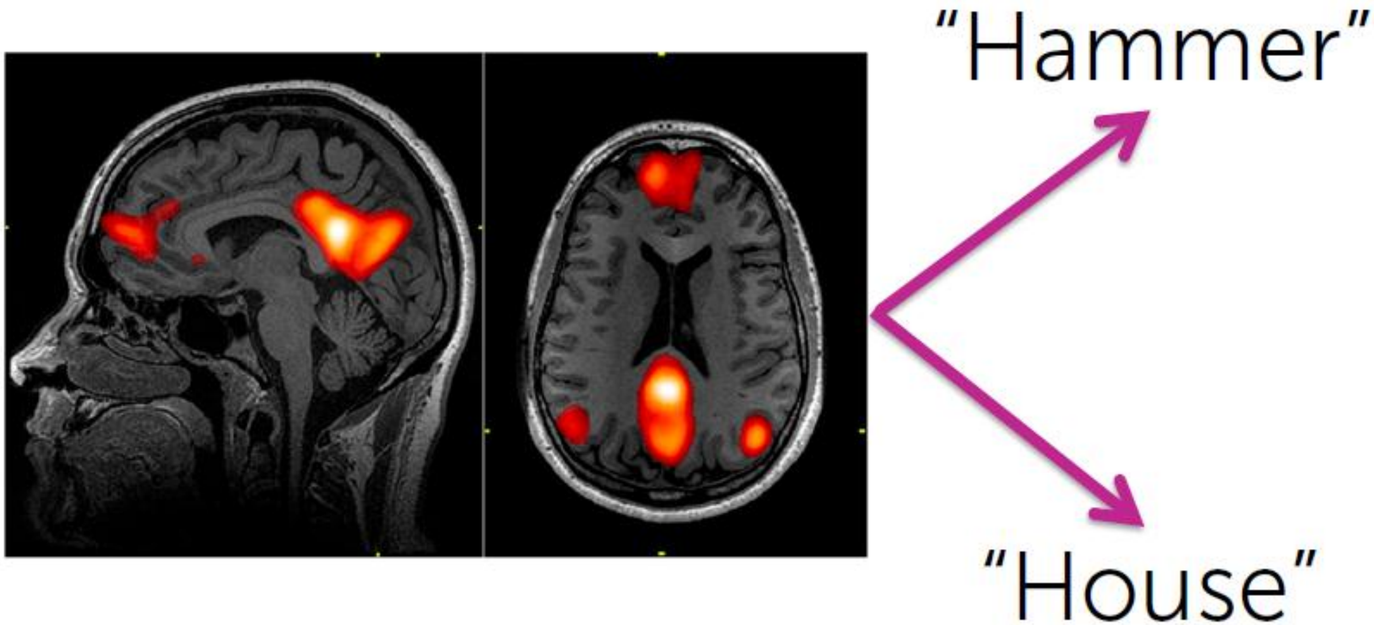
Input: x
Image pixels

Output: y
Predicted object

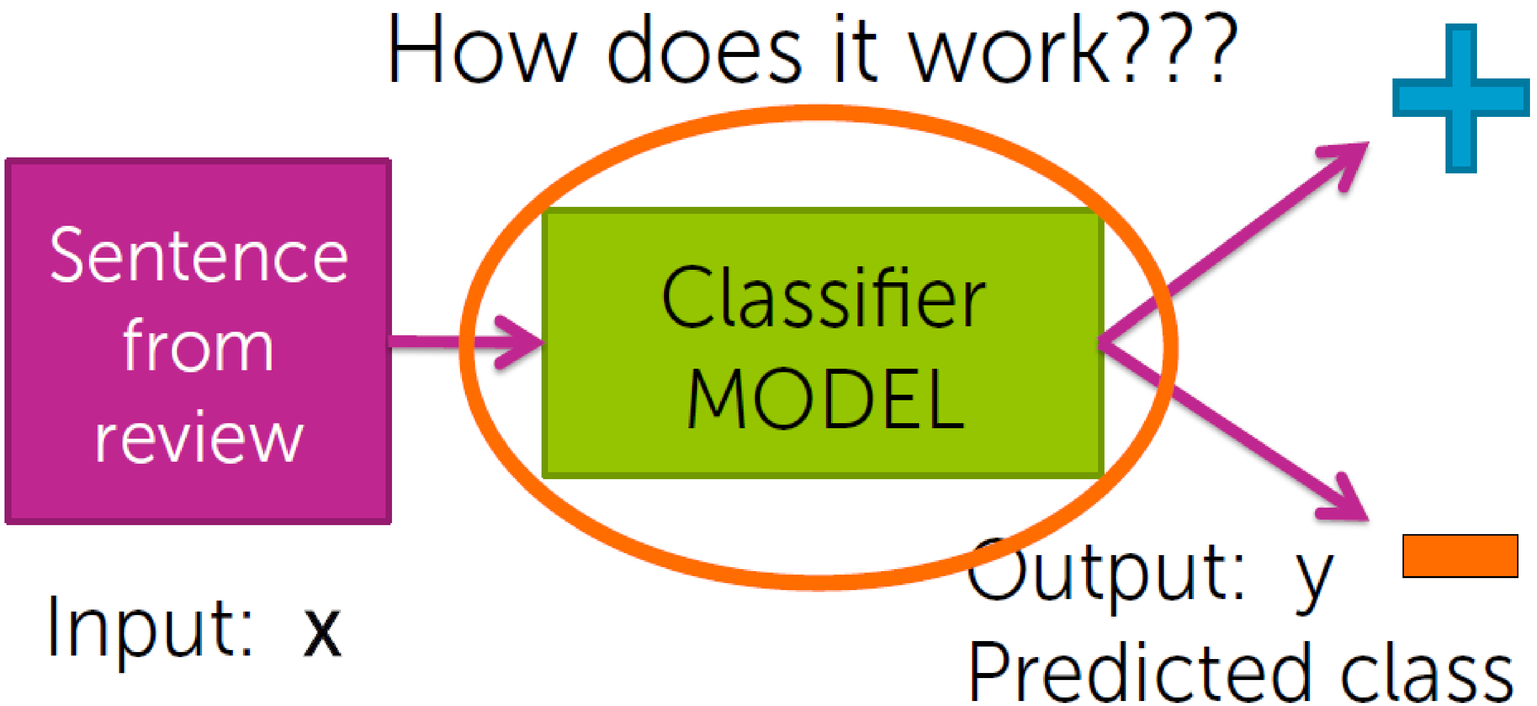
Personalized medical diagnosis



Reading your mind



Representing classifiers



Simple threshold classifier

List of positive words	List of negative words
great, awesome, good, amazing,…	bad, terrible, disgusting, sucks,…

Sentence from review

Input: x

Simple threshold classifier

Count positive & negative words in sentence

If *number of positive words > number of negative words*:

$\hat{y}$ = +

Else:

$\hat{y}$ = −

Simple threshold classifier

List of positive words	List of negative words
great, awesome, good, amazing,…	bad, terrible, disgusting, sucks,…

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

Simple threshold classifier

Count positive & negative words in sentence

2

If *number of positive words > number of negative words*:

ŷ = +

1

Else:

ŷ = −

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

Addressed by learning a classifier

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:

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

$$\text{Score}(x) = 1.2 + 1.7 - 2.1$$
$$= 0.8$$

$\text{Score}(x) > 0 \Rightarrow +$

if

$\text{Score}(x) < 0 \Rightarrow -$

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

Simple linear classifier

Word	Weight
…	…

Sentence from review

Input: x

Simple linear classifier

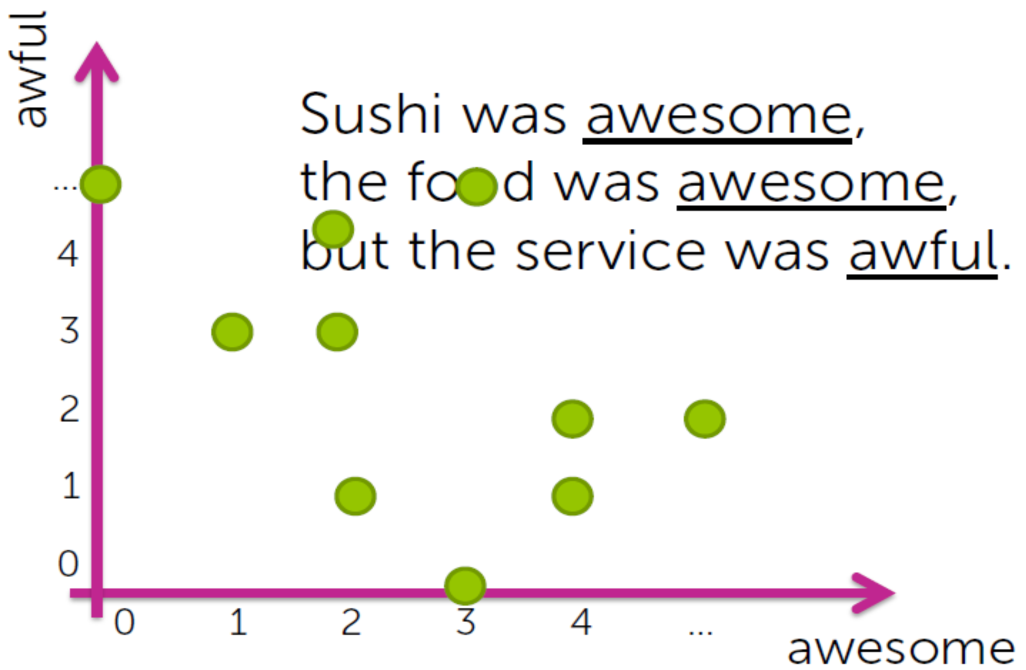
Score(x) = weighted count of words in sentence

If *Score (x) > 0*:
$\hat{y}$ = +
Else:
$\hat{y}$ = −

Suppose only two words had non-zero weight

Word	Weight
awesome	1.0
awful	-1.5

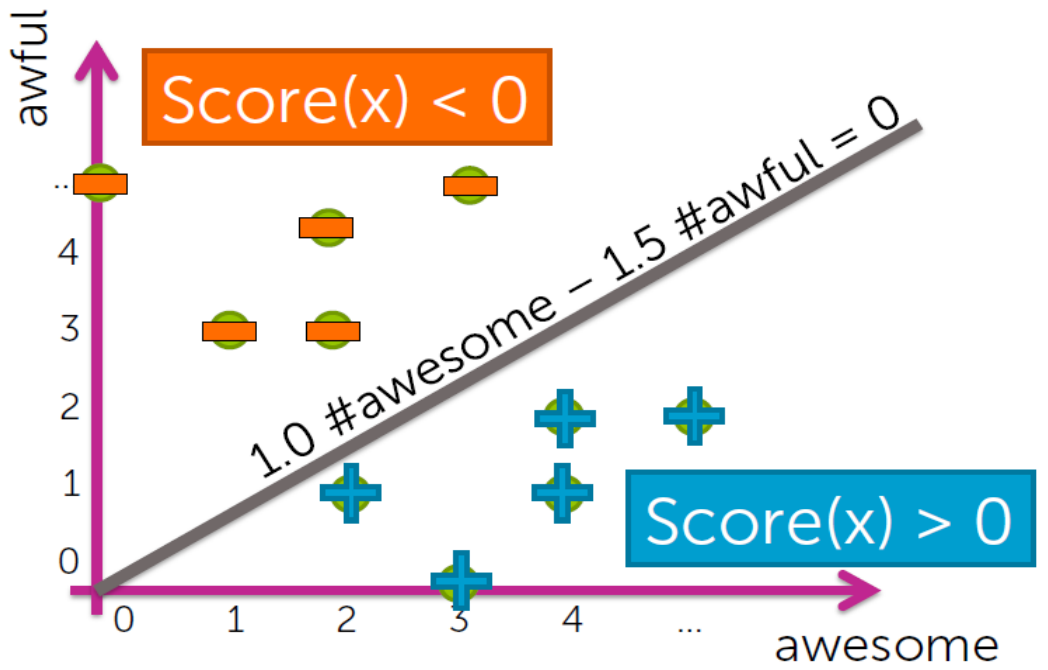
→ Score(x) = 1.0 #awesome – 1.5 #awful



Decision boundary example

Word	Weight
awesome	1.0
awful	-1.5

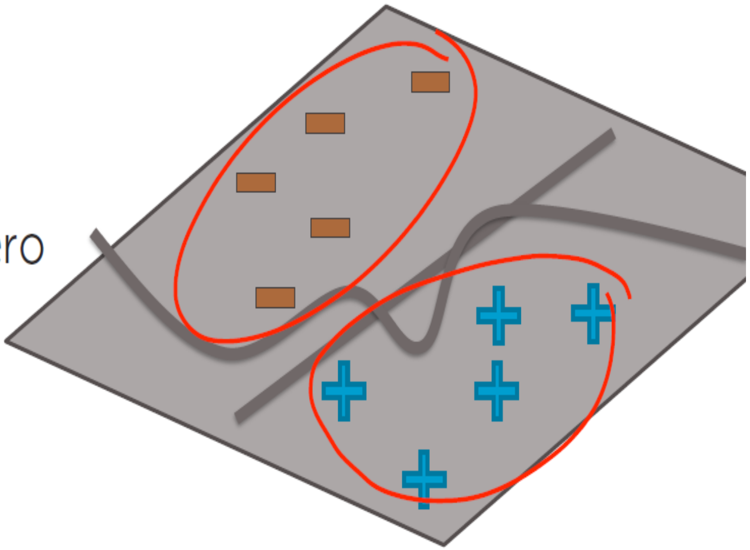
Score(x) = 1.0 #awesome – 1.5 #awful



Decision boundary

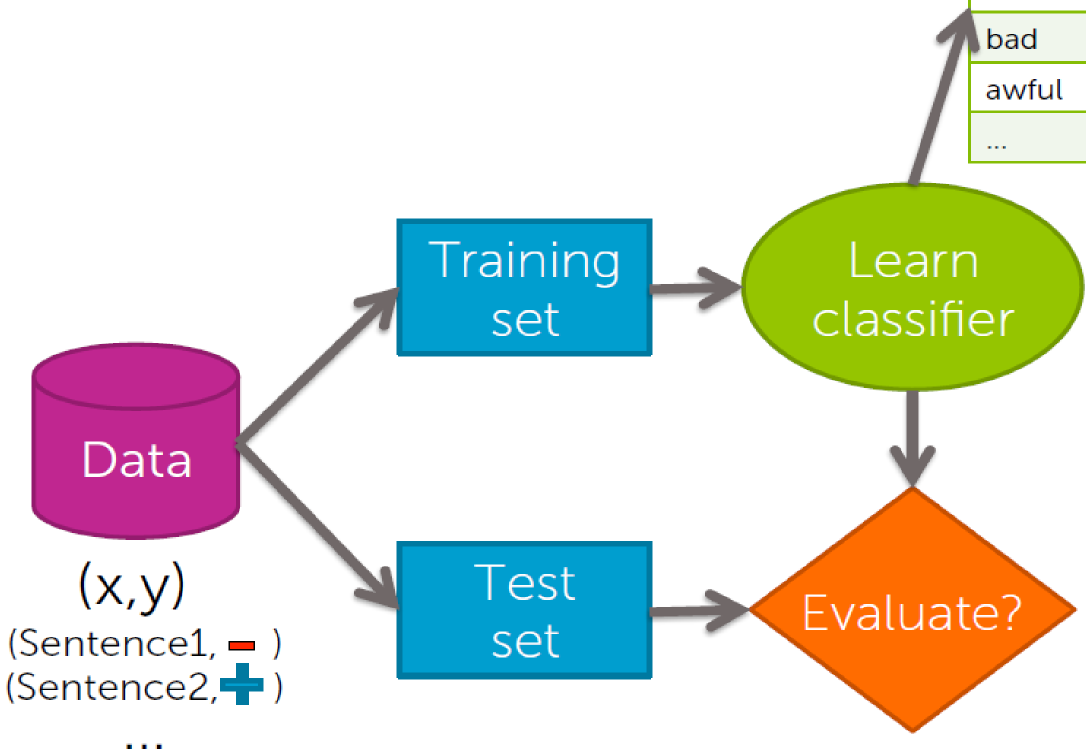
Separates positive & negative predictions

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



Training a classifier = Learning the weights

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



Classification error & accuracy

- Error measures fraction of mistakes

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

  - Best possible value is 0.0

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

$$\text{accuracy} = \frac{\text{\# of Correct}}{\text{Total \# of Sentences}}$$

  - Best possible value is 1.0

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)

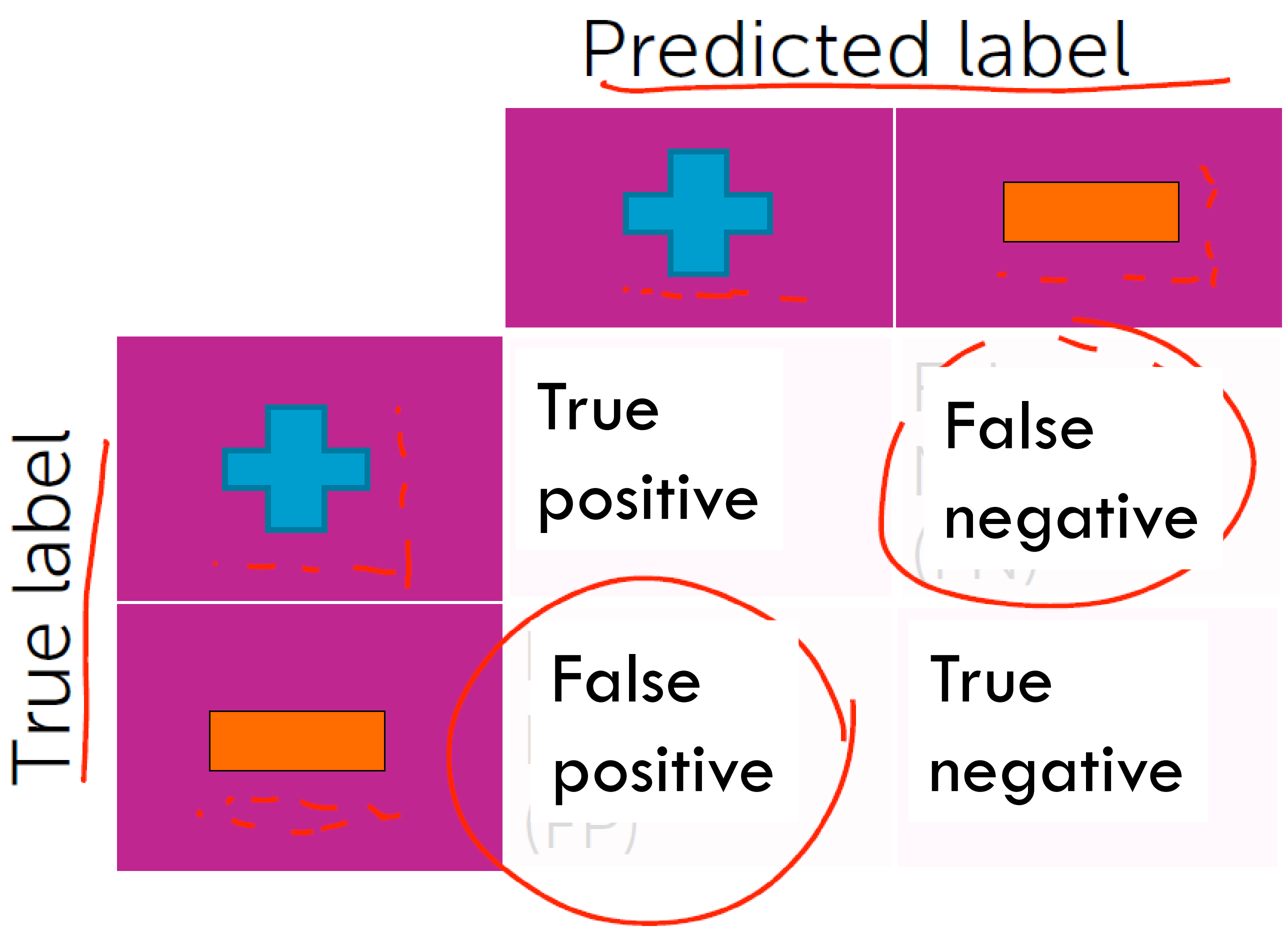
- One class is more common than others
- Beats random (if you know the majority class)

What is a good accuracy?

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 importantly:
  what accuracy does my application need?
  - What is good enough for my user's experience?
  - What is the impact of the mistakes we make?

Types of mistakes



Cost of mistakes

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

	Spam filtering	Medical diagnosis
False negative	Annoying	Disease not treated
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

$$\text{accuracy} = \frac{85}{100} = 0.85$$

Confusion matrix: multiclass classification

100 test examples

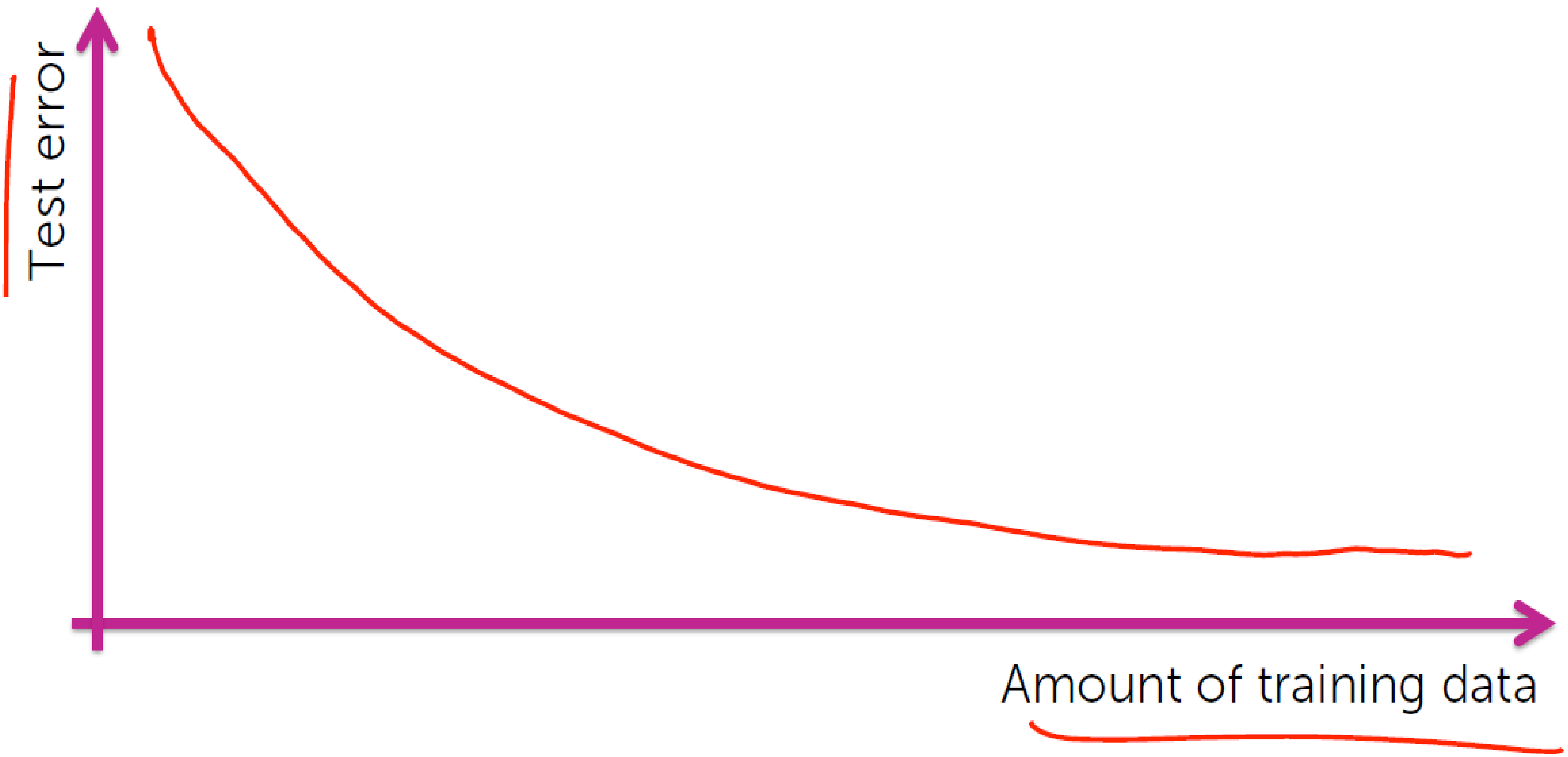
		Predicted label		
		Healthy	Cold	Flu
True label	Healthy 70	60	8	2
	Cold 20	4	12	4
	Flu 10	0	2	8

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

How much data does a model need to learn?

- The more the merrier ☺
  - But data quality is most important factor
- Theoretical techniques sometimes can 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

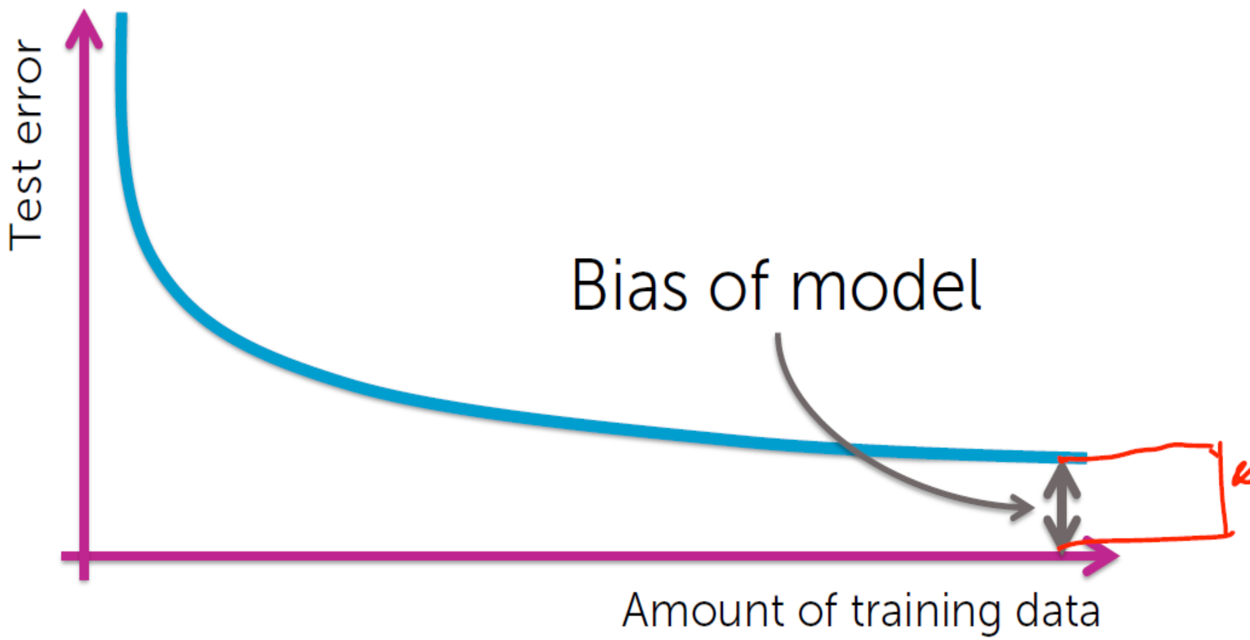
Learning curves



Learning curves

Is there a limit?

Yes, for most models…



even with infinite data, testerror will not go to zero

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."

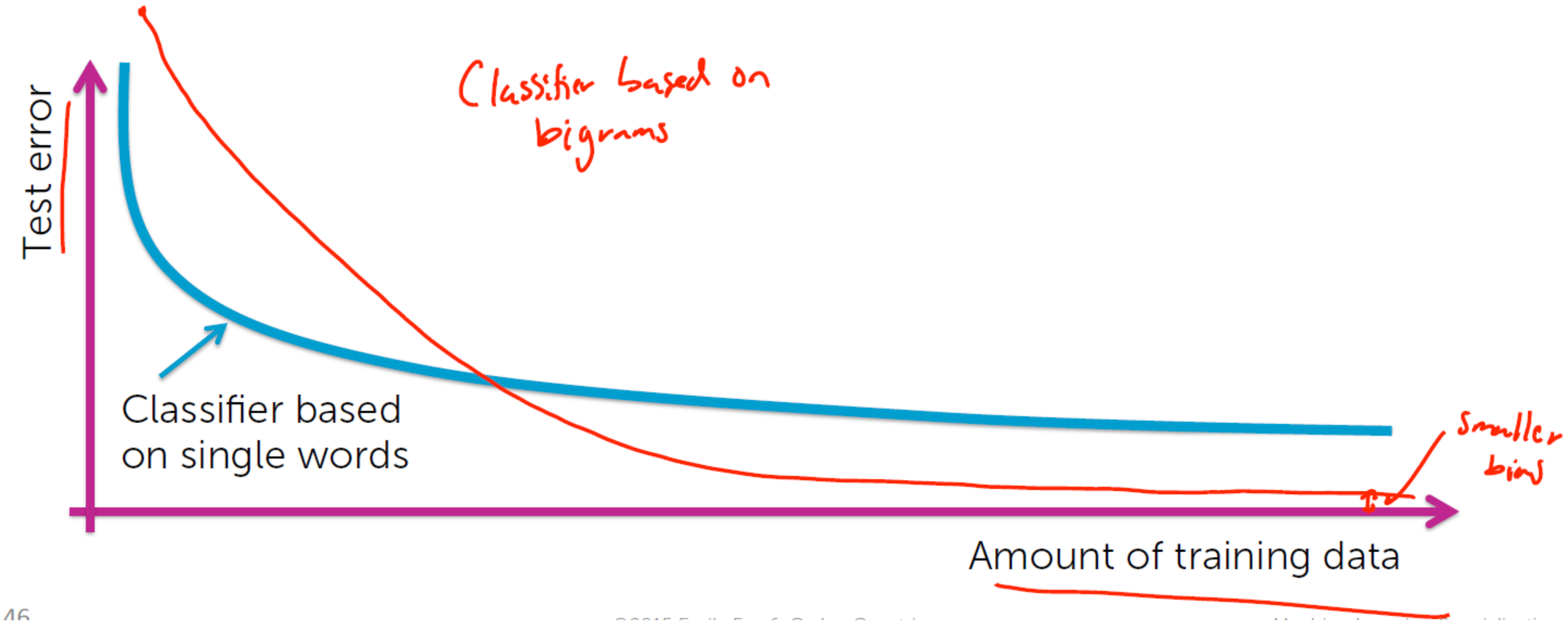
More complex model:
consider pairs of words (bigrams)

Word	Weight
good	+1.5
not good	-2.1

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

Classification based on bigrams

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



How confident is your prediction?

- Thus far, we've outputted a prediction + / −

- But, how sure are you about the prediction?
  - *"The sushi & everything else were awesome!"* ← $P(y=+|x) = 0.99$
  - *"The sushi was good, the service was OK."* ← $P(y=+|x) = 0.55$

Many classifiers provide a confidence level:

$$P(y|x)$$

Output label → y; Input sentence → x

Extremely useful in practice

We have discussed how to

- 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
- Use class probability to express degree of confidence in prediction