Classifying with Regression

Slides borrowed from Jurafsky & Martin Edition 3

Generative and Discriminative Classifiers

Naive Bayes is a **generative** classifier.

Logistic regression is a **discriminative** classifier.

Generative and Discriminative Classifiers

Suppose we're distinguishing cat from dog images

me

imagenet

Generative Classifier:

- Build a model of what's in a cat image
  - Knows about whiskers, ears, eyes
  - Assigns a probability to any image:
    - how cat-y is this image?

Also build a model for dog images

Now given a new image:

Run both models and see which one fits better

Discriminative Classifier

Just try to distinguish dogs from cats

Dogs have collars! Let's ignore everything else

Not easy to ask questions like: *given that I observe a bone, how likely is it to be a dog?*

But: discriminative classifiers generally perform better.

Finding the correct class c from a document d with Generative vs Discriminative Classifiers

Naive Bayes:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(d \mid c) P(c)$$

Likelihoods Prior

Logistic Regression:

$$c_{MAP} = \underset{c \in C}{\operatorname{argmax}} P(c \mid d)$$

Logistic Regression Classifiers

Slides borrowed from Jurafsky & Martin Edition 3

Text Classification: definition

Input:

- a document x
- a fixed set of classes $C = \{c_1, c_2, \ldots, c_J\}$

Output: a predicted class $\hat{y} \in C$

Binary Classification in Logistic Regression

Given a series of input/output pairs:

- $(x^{(i)}, y^{(i)})$

For each observation $x^{(i)}$

- We represent $x^{(i)}$ by a **feature vector** $[x_1, x_2, \ldots, x_n]$
- We compute an output: a predicted class $\hat{y}^{(i)} \in \{0,1\}$

Features in logistic regression

For feature x_i, weight w_i tells is how important is x_i

- x_i ="review contains '`awesome`'": $w_i = +10$
- x_j ="review contains '`abysmal`'": $w_j = -10$
- x_k ="review contains '`mediocre`'": $w_k = -2$

1. Randomly initialize
2. In training, we update weights to learn good weights for the task

Logistic Regression for one observation x

Input observation: vector $x = [x_1, x_2 \dots x_n]$

Weights: one per feature: $W = [w_1, w_2 \dots w_n]$

$\theta = [\theta_1, \theta_2 \dots \theta_3]$

Output: a predicted class $\hat{y} \in \{0, 1\}$

If more than 2 classes: $\hat{y} \in \{0, 1, 2, \dots n\}$

How to do classification

For each feature x_i, weight w_i tells us importance of x_i

- (Plus we'll have a bias b)

We'll sum up all the weighted features and the bias

$$z = \left(\sum_{i=1}^{n} w_i x_i \right) + b$$

bias

$$y = mx + b$$

$$z = wx + b$$

But we want a probabilistic classifier

We need to formalize "sum is high".

We'd like a principled classifier that gives us a probability, just like Naive Bayes did

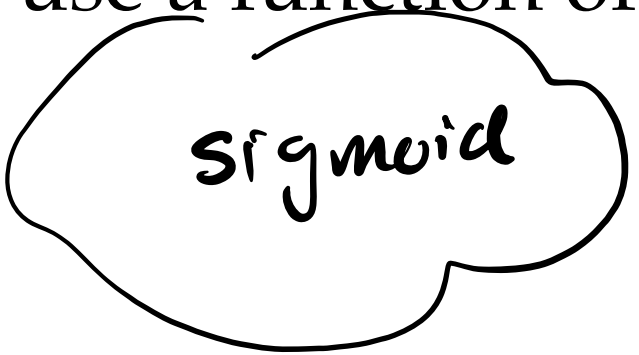
We want a model that can tell us:

$$p(y=1 \mid x; \theta)$$
$$p(y=0 \mid x; \theta)$$

The problem: z isn't a probability, it's just a number!

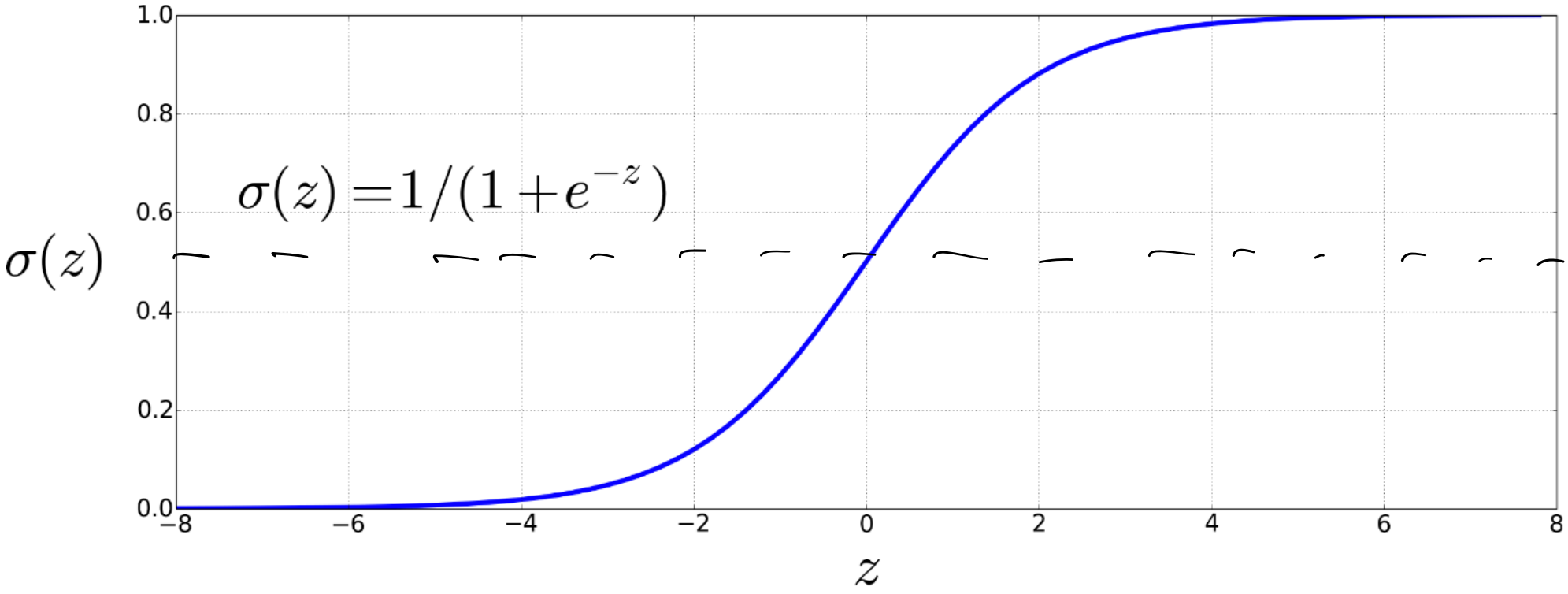
$$z = w \cdot x + b$$

Solution: use a function of z that goes from 0 to 1



$$\sigma(z) = \frac{1}{1+e^{-z}} = \frac{1}{1+\exp(-z)}$$

The very useful sigmoid or logistic function



Idea of logistic regression

Compute $w \cdot x+b$, then pass it through the sigmoid function: $\sigma(w \cdot x+b)$.

Treat the result as a probability.

Making probabilities with sigmoids

$$P(y=1) = \sigma(wx+b)$$

$$= \frac{1}{1+\exp(-(wx+b))}$$

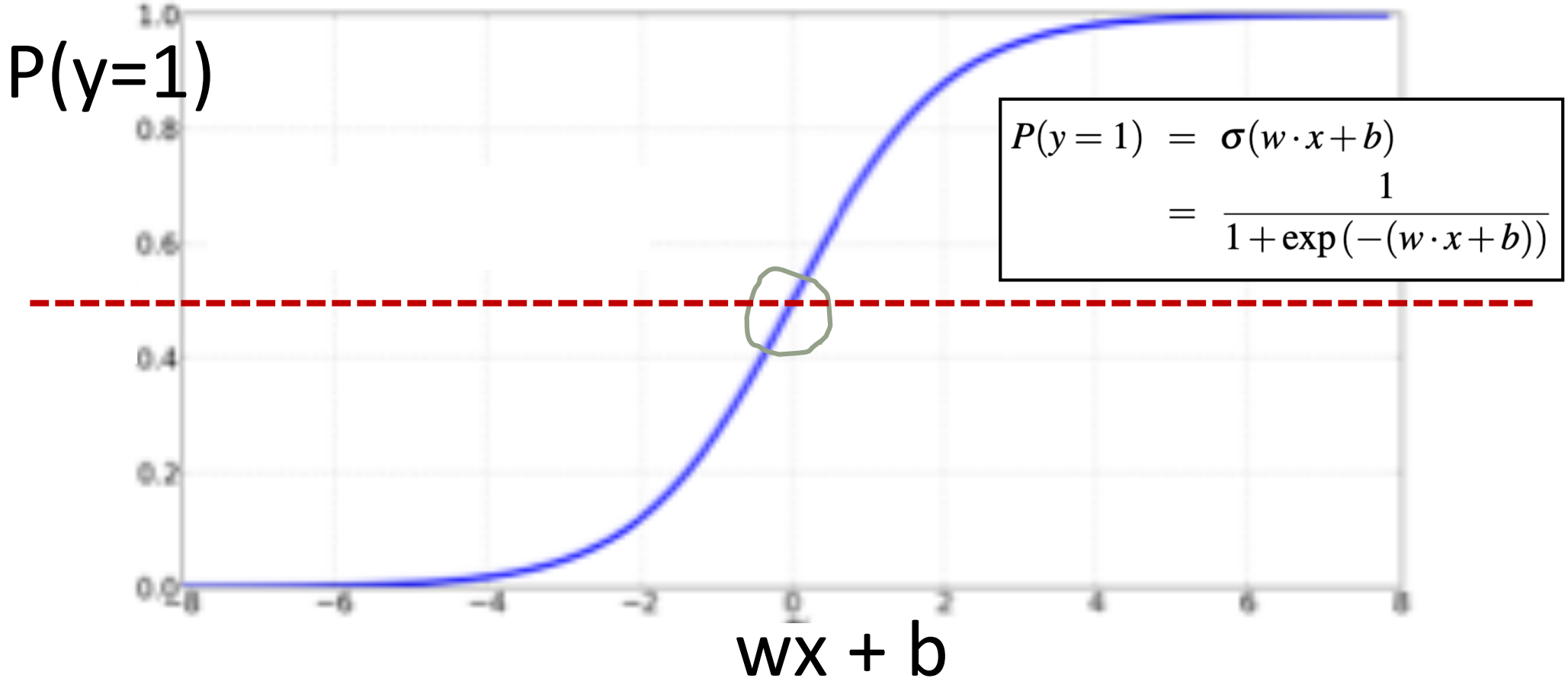
$$P(y=0) = 1-\sigma(wx+b)$$

Turning a probability into a classifier

$$\text{decision}(x) = \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

0.5 here is called the **decision boundary**

The probabilistic classifier



Turning a probability into a classifier

$$\text{decision}(x) = \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

if $w \cdot x + b > 0$

if $w \cdot x + b \leq 0$

The two phases of logistic regression

Training: we learn weights *w* and *b* using **stochastic gradient descent** and **cross-entropy loss**.

Test: Given a test example *x* we compute $p(y|x)$ using learned weights *w* and *b*, and return whichever label ($y = 1$ or $y = 0$) is higher probability

Logistic Regression Example: Text Classification

Slides borrowed from Jurafsky & Martin Edition 3

Sentiment example: does y=1 or y=0?

It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ?

For one thing , the cast is great .

Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

It's hokey . There are virtually no surprises , and the writing is second-rate . So why was it so enjoyable ?

For one thing , the cast is great .

Another nice touch is the music . I was overcome with the urge to get off the couch and start dancing . It sucked me in , and it'll do the same to you .

Var	Definition	
x_1	count(positive lexicon words $\in$ doc)	3
x_2	count(negative lexicon words $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(66) = 4.19$

x_2 1

Classifying sentiment for input x

Var	Definition	
x_1	count(positive lexicon words $\in$ doc)	3
x_2	count(negative lexicon words $\in$ doc)	2
x_3	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
x_4	count(1st and 2nd pronouns $\in$ doc)	3
x_5	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
x_6	log(word count of doc)	$\ln(66) = 4.19$
x_7		1

Suppose w = $[2.5, -5.0, -1.2, 0.5, 2.0, 0.7]$, 0.1

~~b = 0.1~~

Classifying sentiment for input x

$w = [2.5, -5, -1.2, 0.5, 2, 0.7]$ $b = 0.1$

$x = [3, 2, 1, 3, 0, 4.19]$

$$p(+|x) = P(y=1|x) = \sigma(wx+b)$$

$$= \sigma\left([2.5, -5, -1.2, 0.5, 2, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1\right)$$

$$= \sigma(0.8333) = 0.7$$

$$p(-|x) = P(y=0|x) = 1 - \sigma(wx+b)$$

$$= 1 - 0.7 = 0.3$$

Classifying sentiment for input x

$w = [2.5, -5, -1.2, 0.5, 2, 0.7]$ $b = 0.1$

$x = [0, 0, 0, 0, 0, 0]$

$wx = 0$

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

$\sigma(0.1)$

$$p(-|x) = P(y = 0|x) =$$

Classification in (binary) logistic regression: summary

Given:

- a set of classes: (+ sentiment,- sentiment)
- a vector **x** of features `[x1, x2, …, xn]`
  - x1= count("awesome")
  - x2 = log(number of words in review)
- A vector **w** of weights `[w1, w2, …, wn]`
  - w_i for each feature f_i

$$P(y=1) = \sigma(w \cdot x + b)$$
$$= \frac{1}{1+\exp(-(w \cdot x + b))}$$

Multi-class Regression

Slides borrowed from Jurafsky & Martin Edition 3

Multinomial Logistic Regression

Often we need more than 2 classes

- Positive/negative/neutral
- Parts of speech (noun, verb, adjective, adverb, preposition, etc.)
- Classify emergency SMSs into different actionable classes

If >2 classes we use **multinomial logistic regression**

= Softmax regression

= Multinomial logit

= Maximum entropy modeling or MaxEnt

So "logistic regression" means binary (2 classes)

Multinomial Logistic Regression

The probability of everything must still sum to 1

```
P(positive|doc) + P(negative|doc) +
P(neutral|doc) = 1
```

Need a generalization of the sigmoid called **softmax**

- Takes a vector $z = [z1, z2, ..., zk]$ of k arbitrary values
- Outputs a probability distribution

The softmax function

Turns a vector $z = [z_1, z_2, \ldots, z_k]$ of k arbitrary values into probabilities

$$\text{softmax}(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)} \qquad 1 \le i \le k$$

normalization term

The softmax function

Turns a vector $z = [z_1, z_2, \ldots, z_k]$ of k arbitrary values into probabilities :

$$z = [0.6, 1.1, -1.5, 1.2, 3.2, -1.1]$$

$$\text{softmax}(z) = \left[\frac{\exp(z_1)}{\sum_{i=1}^{k} \exp(z_i)}, \frac{\exp(z_2)}{\sum_{i=1}^{k} \exp(z_i)} \cdots \frac{\exp(z_n)}{\sum_{i=1}^{k} \exp(z_i)} \right]$$

$$= [0.055, 0.09, 0.006, 0.099, 0.74, 001]$$

$$[p(\text{Romance}|d), p(\text{Horror}|d) \ldots p(\text{Fantasy}|d)]$$

Softmax in multinomial logistic regression

$$p(y=c|x) = \frac{\exp(w_c \cdot x + b_c)}{\sum_{j=1}^{K} \exp(w_j \cdot x + b_j)}$$

weights for that class

bias for that class

Input is still the dot product between weight vector w and input vector x, but now we need separate weight vectors for each of the K classes.

W_{Pos} W_{Neg} $W_{Neutral}$

Features in binary versus multinomial logistic regression

Binary: positive weight

sigmoid

$$x_5 = \begin{cases} 1 & \text{if } \text{“!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$$

$$w_5 = 3.0$$

Multinominal: separate weights for each class:

softmax

Feature	Definition	$w_{5,+}$	$w_{5,-}$	$w_{5,0}$
$f_5(x)$	1 if “!” $\in$ doc 0 otherwise	3.5	3.1	−5.3