CS 232: Artificial Intelligence

Fall 2023

Prof. Carolyn Anderson
Wellesley College

Classification Methods

Slides borrowed from Jurafsky & Martin Edition 3

Classification Methods: Supervised Machine Learning

Lots of kinds!

- Naïve Bayes
- Logistic regression
- Neural networks
- k-Nearest Neighbors
- random forests
- …

Classification Methods: Supervised Machine Learning

Input:

- a document d
- a fixed set of classes $C = \{c_1, c_2 \dots c_i\}$
- A training set of *m* hand-labeled documents

$(d_1, c_1), \dots, (d_m, c_m)$ gold standard "true"

Output:

- a learned classifier $y: d \rightarrow c$

document class

Components of a probabilistic machine learning classifier

Given *m* input/output pairs *($x^{(i)}$, $y^{(i)}$)*:

1. A **feature representation** of the input
   For each input $x^{(i)}$, a vector of features $[x_1, x_2 \ldots x_n]$
   Feature j for input $x^{(i)}$ is x_j.
2. A **classification function** that computes $\hat{y}$, the estimated class via $p(y|x)$
3. An objective function for learning, like **cross-entropy loss**
4. An algorithm for optimizing the objective function: **stochastic gradient descent**.

Logistic Regression Classifiers

Slides borrowed from Jurafsky & Martin Edition 3

Is this spam?

Subject: **Important notice!**
From: Stanford University <newsforum@stanford.edu>
Date: October 28, 2011 12:34:16 PM PDT
To: undisclosed-recipients:;

Greats News!

You can now access the latest news by using the link below to login to Stanford University News Forum.

http://www.123contactform.com/contact-form-StanfordNew1-236335.html

Click on the above link to login for more information about this new exciting forum. You can also copy the above link to your browser bar and login for more information about the new services.

© Stanford University. All Rights Reserved.

Who wrote which Federalist papers?

Anonymous essays try to convince New York to ratify U.S Constitution. Authorship of 12 of the letters in dispute.

Solved by Mosteller and Wallace (1963) using Bayesian methods

What is the subject of this research article?

MEDLINE Article

?

Antogonists and Inhibitors
Blood Supply
Chemistry
Drug Therapy
Embryology
Epidemiology
…

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:

doc (x^i, y^i) class label

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

- We represent $x^{(i)}$ $[x_1, x_2 \ldots x_n]$
- We compute an output $\hat{y} \in \{0, 1\}$

Features in logistic regression

Bag of Words Features

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

- x_i ="review contains '`awesome`'": $w_1 = +10$
- x_j ="review contains '`abysmal`'": $w_2 = -10$
- x_k ="review contains '`mediocre`'": $w_3 = -2$

Logistic Regression for one observation x

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

Weights (one per feature): $W = [w_1, w_2 \ldots w_n]$

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

How to do classification

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

Bias : b

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

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

$$z = wx + b$$

$$+10 \cdot 1 + -10 \cdot 1 + -2 \cdot 1 = -2$$

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:

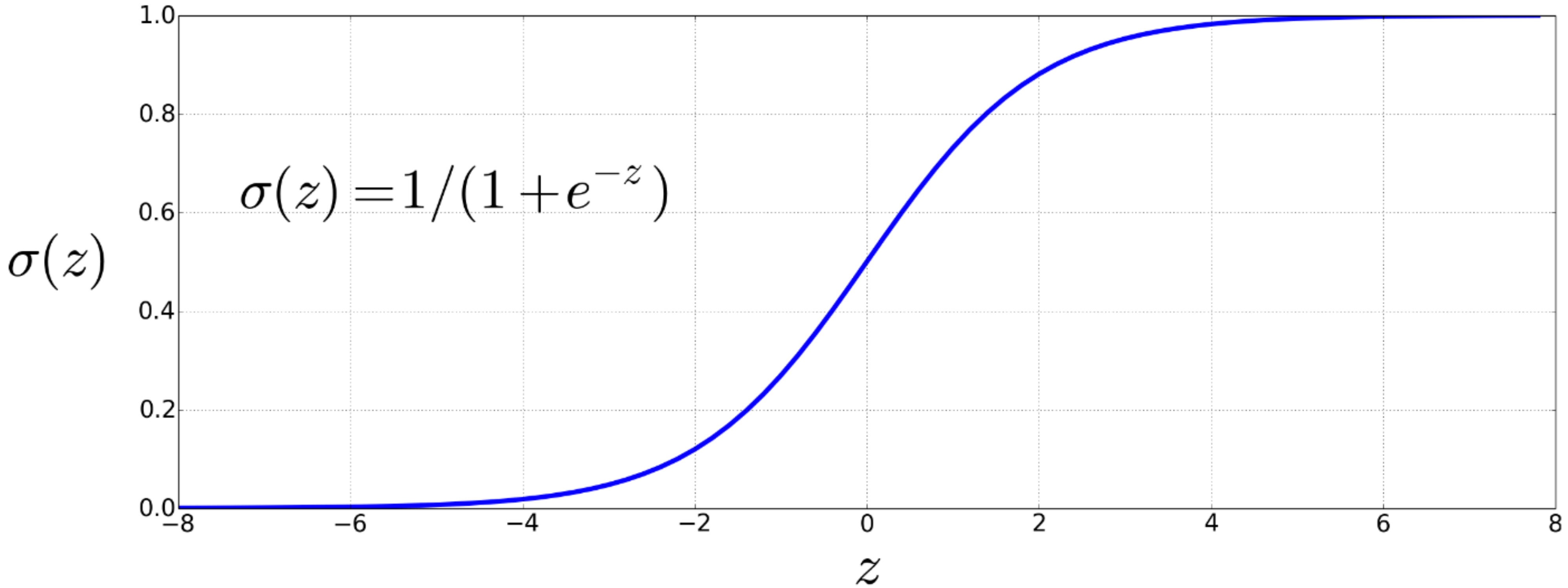
$$p(y=1 \mid x; w)$$
$$p(y=0 \mid x; w)$$

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

$$z = wx + b$$

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

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 it as a probability

Making probabilities with sigmoids

$$P(y=1 \mid x; W) = \sigma(Wx+b)$$

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

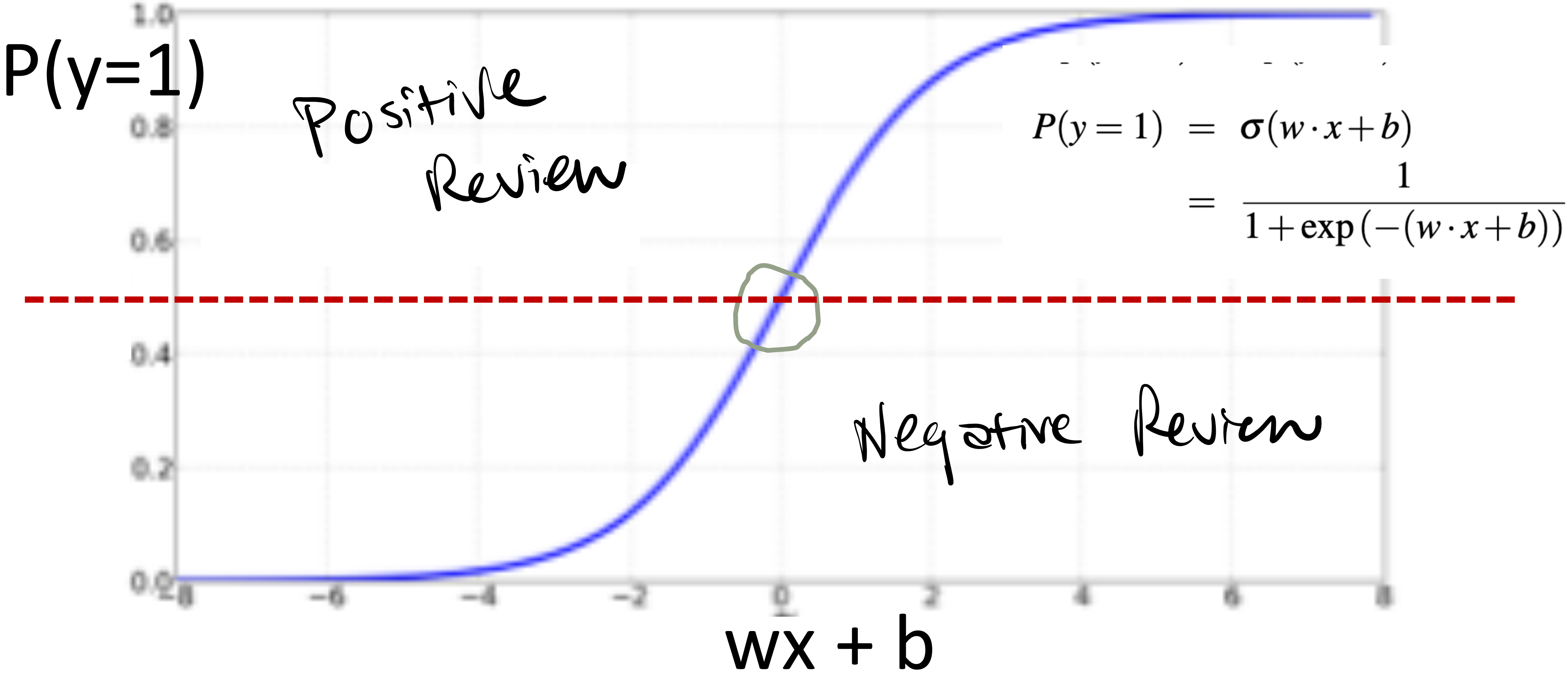
$$P(y=0 \mid x; W) = 1 - \sigma(Wx+b)$$

Turning a probability into a classifier

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

0.5 here is the decision boundary

The probabilistic classifier



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

Turning a probability into a classifier

$$\text{decision}(x) = \begin{cases} 1 & \text{if } wx + b > 0 \\ 0 & \text{if } wx + b \leq 0 \end{cases}$$

The two phases of logistic regression

Training: We learn weights W & b using stochastic gradient descent & cross-entropy loss

Test: Given an example x we compute p(y|x) using learned weights

Return whichever label 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 .

Features

Weights

Classifying sentiment for input 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

$$
\begin{aligned}
P(y=1) &= \sigma(w \cdot x + b) \\
&= \frac{1}{1+\exp\left(-(w \cdot x + b)\right)}
\end{aligned}
$$

Evaluating Classifiers

Slides borrowed from Jurafsky & Martin Edition 3

Evaluation

Consider a binary text classification task:

Is this passage from a book a "smell experience" or not?

Towards Olfactory Information Extraction from Text: A Case Study on Detecting Smell Experiences in Novels

Ryan Brate and **Paul Groth**
University of Amsterdam
Amsterdam, the Netherlands
r.brate@gmail.com
p.t.groth@uva.nl

Marieke van Erp
KNAW Humanities Cluster
Digital Humanities Lab
Amsterdam, the Netherlands
marieke.van.erp@dh.huc.knaw.nl

Abstract

Environmental factors determine the smells we perceive, but societal factors factors shape the importance, sentiment and biases we give to them. Descriptions of smells in text, or as we call them 'smell experiences', offer a window into these factors, but they must first be identified. To the best of our knowledge, no tool exists to extract references to smell experiences from text. In

Evaluation

Consider a binary text classification task:

Is this passage from a book a "smell experience" or not?

You build a "smell" detector

- Positive class: paragraph that involves a smell experience
- Negative class: all other paragraphs

The 2-by-2 confusion matrix

		Truth	
Prediction			

Evaluation: Accuracy

Why don't we use **accuracy** as our metric?

Imagine we saw 1 million paragraphs

- 100 of them mention smells
- 999,900 talk about something else

We could build a classifier that labels every paragraph "not about smell"

Evaluation: Accuracy

Why don't we use **accuracy** as our metric?

Imagine we saw 1 million paragraphs

- 100 of them mention smells
- 999,900 talk about something else

We could build a classifier that labels every paragraph "not about smell"

- It would get 99.99% accuracy!
- But the whole point of the classifier is to help literary scholars find passages about smell to study--- so this is useless!
- That's why we use **precision** and **recall** instead

Evaluation: Precision

% of items the system detected (i.e., items the system labeled as positive) that are in fact positive (according to the human gold labels)

PRECISION =

Evaluation: Recall

% of items actually present in the input that were correctly identified by the system.

RECALL =

Why Precision and recall

Our no-smells classifier

- Labels nothing as "about smell"

Accuracy =

Recall =

Precision =

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

The softmax function

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

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

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

$$[0.055, 0.090, 0.006, 0.099, 0.74, 0.010]$$

Softmax in multinomial logistic regression

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

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.

Features in binary versus multinomial logistic regression

Binary: positive weight

$$x_5 = \begin{cases} 1 & \text{if “!”} \in \text{doc} \\ 0 & \text{otherwise} \end{cases} \qquad w_5 = 3.0$$

Multinominal: separate weights for each class:

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

Computing with Probabilities

Numerical Underflow

So far we've been working with relatively small sample spaces. This means our probabilities have been decently large.

As we go on in this class, our sample spaces are going to get much larger. We want to be able to reason about the probabilities of things like:

- All words in English
- All pixels in a photo
- All possible game states for Pacman

Numerical Underflow

Problem: when our probabilities get really really small, programming languages start making mistakes.

There is a **bound on precision** in numerical computing.

This is because of the limitations on space allocation for (floating point) numbers.

Solution: make the numbers bigger

- Intuition: we care about how big probabilities are relative to the other probabilities in our distribution, not the actual value.

Probabilities:
p(heart) = 0.1
p(rainbow) = 0.2
p(letter) = 0.7

Interpretation: a letter is **7 times more likely** than a heart!

Solution: make the numbers bigger

- Intuition: we care about how big probabilities are relative to the other probabilities in our distribution, not the actual value.

<u>Probabilities:</u>
p(heart) = ~~0.1~~ 100
p(rainbow) = ~~0.2~~ 200
p(letter) = ~~0.7~~ 700

What if we just multiply all our probs by 100?

This preserves the ratio.

Solution: make the numbers bigger

- What if we just multiply all our probs by 100? This preserves the ratio.

<u>Probabilities:</u>

p(heart) = ~~0.1~~ 100

p(rainbow) = ~~0.2~~ 200

p(letter) = ~~0.7~~ 700

However, if we want to recover the probabilities later, we'll need to **renormalize** them. This means **remembering that we multiplied by 100**.

Solution: log-transform the numbers

- Instead, we use a log transformation. This changes the range from [0,1] to [-∞, 0].

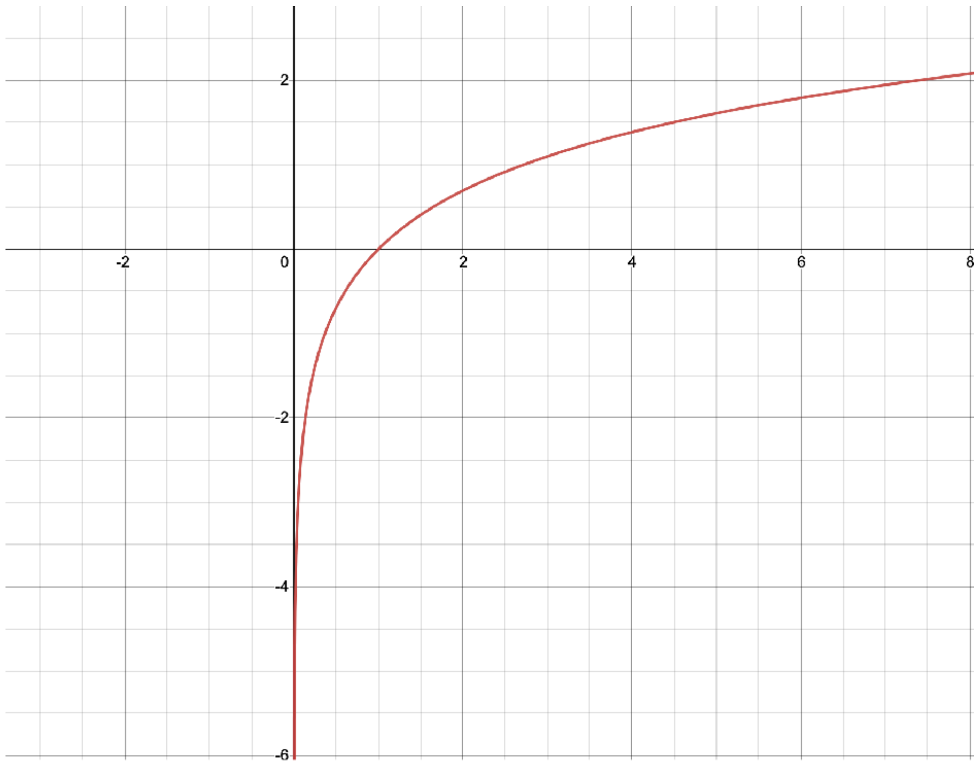
Log base doesn't matter much but we usually use natural log (base e):

Probabilities:

p(heart) = ~~0.1~~ -2.3

p(rainbow) = ~~0.2~~ -1.6

p(letter) = ~~0.7~~ -0.36



www.desmos.com/calculator/aczt76asao

Bonus

Generative and Discriminative Classifiers

Slides borrowed from Jurafsky & Martin Edition 3

Logistic Regression

- Important analytic tool in natural and social sciences
- Baseline supervised machine learning tool for classification
- Is also the foundation of neural networks

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

Discriminative Classifier

Just try to distinguish dogs from cats

Dogs have collars! Let's ignore everything else

A combined measure: F

F measure:

a single number that combines P and R

A combined measure: F

F measure:

a single number that combines P and R

$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P + R}$$

We almost always use balanced F_1 (i.e., β = 1)

$$\mathrm{F}_1 = \frac{2PR}{P+R}$$