Natural Language Processing

Machine learning foundations: Logistic regression

Ana Marasović
University of Utah

Welcome!

(Professor) Ana (Marasović) [she/her]

- Joined University of Utah in 2022
- *My research:* **NLP/LLMs**, with a focus on **interpretability** and **AI-assisted** *decision making*, *communication*, and *creativity*

Postdoc in **Computer Science** (Seattle, USA)

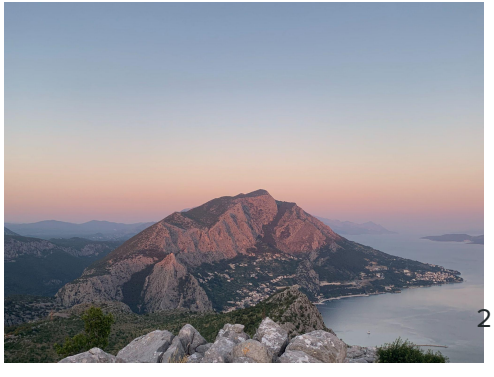
UNIVERSITY of WASHINGTON

Ai2 Allen Institute for AI

PhD in **Comp. Linguistics** (Heidelberg, Germany)

MSc/BSc in **Math** (Zagreb, Croatia)

Grew up here (Omiš, Croatia)



TAs

Fateme Hashemi Chaleshtori [she/her]

3rd year PhD student

She has researched:

- Measuring the usefulness of LLM explanations for improving AI-assisted decision making
- Connecting explanations generated in plain language with internal computations
- Legal NLP

Purbid Bambroo [he/him]

2nd year MS student

Served as a TA for NLP in Spring 2024

He has worked on:

- Supporting research on finding influential train examples for LLM predictions
- Is starting to research RLHF

Review this syllabus carefully – students are responsible for understanding everything there!

https://utah.instructure.com/courses/983222

Office hours

Mondays 11am–12pm with Fateme (2426 WEB)

Tuesdays 2–3pm with Ana (Zoom; link in the Syllabus)

Wednesdays 4:30–5:30pm with Purbid (2426 WEB)

Thursday 11am–12pm with Fateme (2426 WEB)

Fridays 3–4pm with Purbid (Zoom; link in the Syllabus)

- The first office hours are tomorrow (Aug 22)
- We are open to appointments outside of our office hours, but make sure to reach out a few days in advance

ML/DL Background

- I will cover all the necessary ML/DL concepts required to solve the assignments
- The practical intro to ML/DL will be fast-paced, so it's important to think about whether you can keep up. Here are some things to consider:
  - Are you also taking other demanding courses this semester?
  - How quickly do you typically code?
  - How much time do you usually need to grasp new concepts?
  - How comfortable are you with taking derivatives, understanding matrix/vector notation and operations, and the basics of probability, mean, and standard deviation?
  - Why do you want to take this course?
- Talk to undergraduate/graduate advisors
- See feedback from students who were in the same position I shared in the announcement
- Start working on A1 soon after this lecture

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

Sentiment Classification

IMDb

This film is interesting as an experiment but **tells no cogent story**. One might feel virtuous for sitting thru it because it touches on so many IMPORTANT issues but it does so **without any discernable motive**. The viewer comes away with **no new perspectives** (unless one comes up with one while one's mind wanders, as it will invariably do during this **pointless film**). **One might better spend one's time staring out a window at a tree growing**.

How to approach building a sentiment classifier with machine learning?

4 components of a supervised ML for binary classification

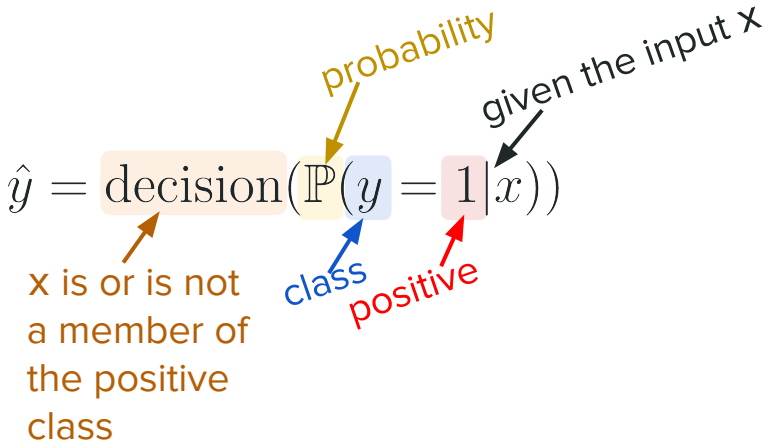
1. **A feature representation of the input**

$$x \mapsto f(x) = [f(x)_1, \ldots, f(x)_n] \in \mathbb{R}^n$$

model input (movie review)

high-dimensional feature vector

2. **A classification function that decides which class to apply to an instance:**



4 components of a supervised ML for binary classification

3. **An objective function that we want to optimize for to learn appropriate model parameters**
    - Usually a function corresponding to error on example
    - Consider:

      The difference between

      the predicted probability for the true class label of the example

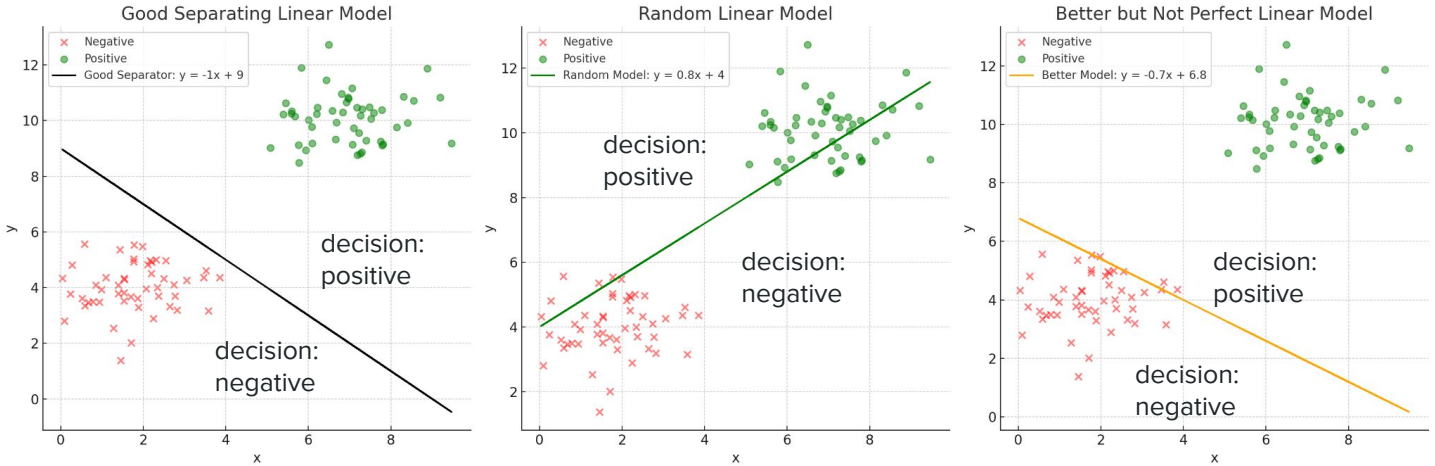
      vs.

      what the probability for the true class label should be (= 1)
    - Optimization aims to find model parameters/weights that minimize a function corresponding to error
    - We use information about what the true class label of the example is, hence *supervised machine learning*

      ☀️ Requires that human annotators label examples

4 components of a supervised ML for binary classification

3. An objective function that we want to optimize for to learn appropriate model parameters



error = 0 | error = 17+ | error = 2+

Plots generated with ChatGPT 4o

4 components of a supervised ML for binary classification

4. **An algorithm for finding optimal model parameters/weights for the objective function**
    - We don’t have formulae for identifying optima
    - Instead, we use iterative methods for moving towards an optimum
    - For `y=ax+b` example, start with random `a` and `b`, and iteratively change them until there is little to no error

Two phases

1. **Training**: We use a set of *labeled* training examples $\mathcal{D} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$ to make decisions, compute the decision error, & change the model parameters/weights to minimize the error
2. **Testing:** We use a held-out labeled set of test instances (not used for training) and measure the model accuracy

Notation notes:

- {} denotes a set
- Subscript and superscript next } shorten writing $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$
- Superscripts in parentheses refer to individual instances
- (,) denotes a pair
- x denotes input
- y denotes the class label

4 components of a supervised ML for binary classification

1. A feature representation of the input
2. A classification function that decides which class to apply to an instance
3. An objective function that we want to optimize for to learn appropriate model parameters
4. An algorithm for finding optimal model parameters/weights for the objective function

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

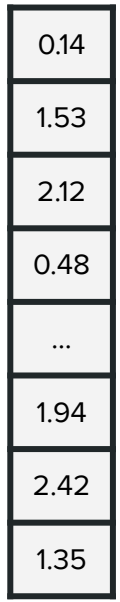
A feature representation of the input

This film is interesting as an experiment but **tells no cogent story**. One might feel virtuous for sitting thru it because it touches on so many IMPORTANT issues but it does so **without any discernable motive**. The viewer comes away with **no new perspectives** (unless one comes up with one while one's mind wanders, as it will invariably do during this **pointless film**). **One might better spend one's time staring out a window at a tree growing**.

$$x \mapsto f(x) = [f(x)_1, \ldots, f(x)_n] \in \mathbb{R}^n$$

model input (movie review)

high-dimensional feature vector



Tokenization

Splitting a string into a sequence of *tokens*

Tokens: “basic units which need not be decomposed in a subsequent processing” [Webster and Kit, 1992]

Whitespace tokenizer

- This film is interesting as an experiment but tells no cogent story.
- [This, film, is, interesting, as, an, experiment, but, tells, no, cogent, story.]

Whitespace tokenizer failures

- *conjunctions:* isn’t ⇒ is, n’t
- *hyphenated phrases:* prize-winning ⇒ prize, -, winning
- *punctuation:* great movie! ⇒ great, movie, !

For now, let’s use whitespace tokenizer but in the next lecture we will cover more

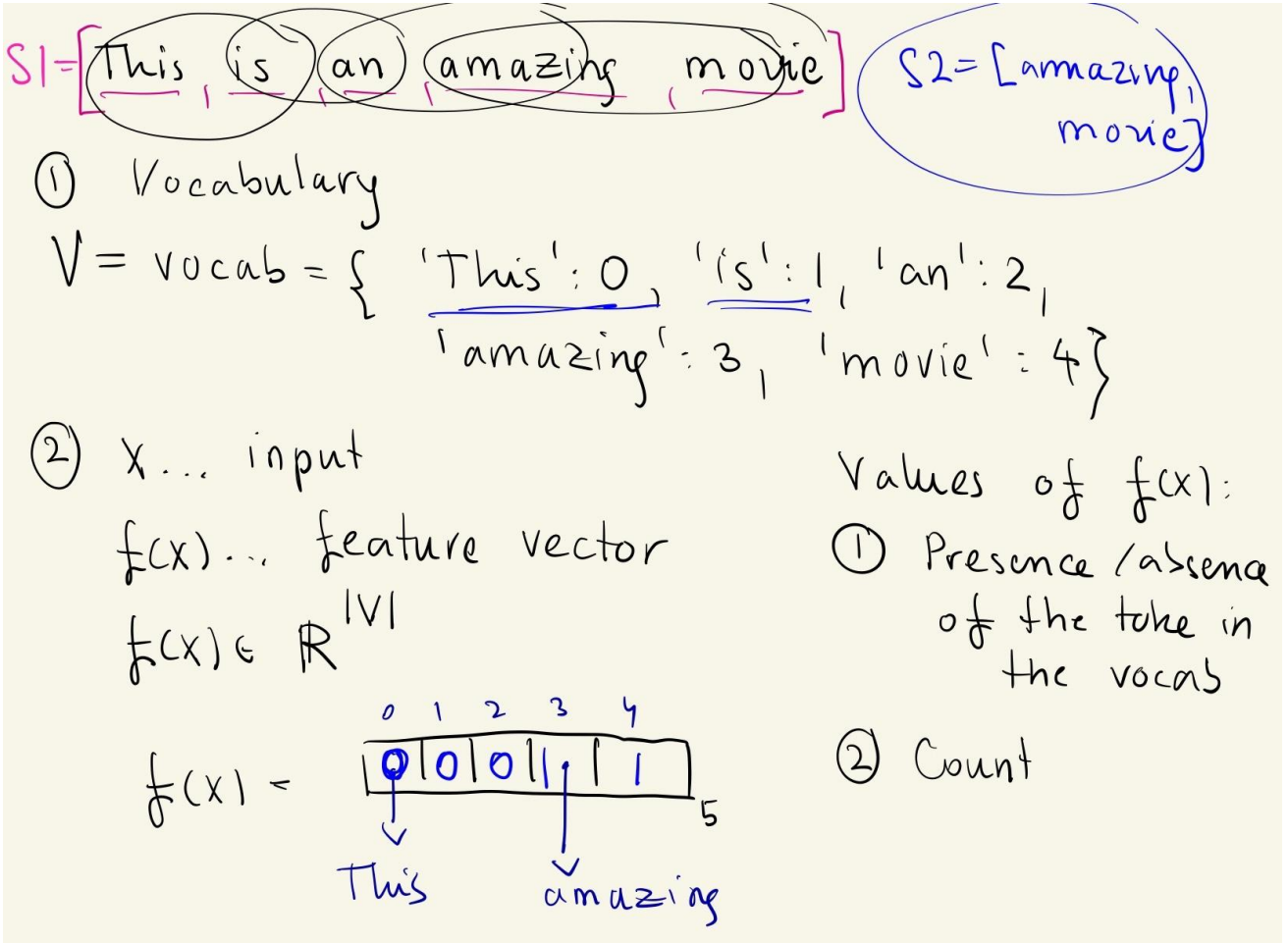
Preprocessing / Text normalization

- **Lemmatization:** determining that two words have the same root, despite their surface differences
  - sang, sung, and sings are forms of sing
- **Stemming**: strip suffixes from the end of the word
  - sings ↦ sing
- **Sentence segmentation:** Breaking up a text into individual sentences
- **Stopword removal:** Remove commonly used words in a language
  - a, the, is, are
- **Casing:** Lowercase all words or not

With LLMs, we need to take care of almost none of these

Feature vector

Bag-of-word



Feature vector

Bag-of-word

- Vocabulary
- The vector size
- Connection between a feature vector dimension and the vocabulary
- The value in each dimension of the feature vector (presence/absence, count, etc.)

Bag-of-ngrams

- **n-gram**: a sequence of n tokens
- Bag-of-words = bag-of-unigrams

Feature vector

Bag-of-word

Bag-of-ngrams

TF-IDF

```
t = token
d = document (movie review)
term_frequency(t,d) = number of times t occurs in d
document_frequency(t) = # documents t occurs in
N = number of documents
inverse_document_frequency(t) = N / document_frequency(t)
tf-idf(t,d) = tf(t,d) x inverse_document_frequency(t)
```

A feature vector for `d` is produced by stacking `tf-idf(t,d)` in an order determined by the vocabulary

4 components of a supervised ML for binary classification

1. **A feature representation of the input**

$$x \mapsto f(x) = [f(x)_1, \ldots, f(x)_n] \in \mathbb{R}^n$$

model input (movie review)

high-dimensional feature vector

Options for feature vectors:

- Bag-of-word
- Bag-of-ngrams
- TF-IDF
- We will introduce more...

4 components of a supervised ML for binary classification

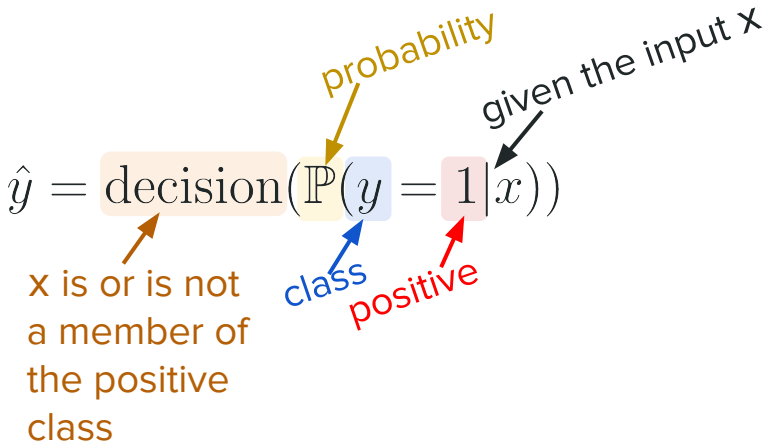
1. **A feature representation of the input**

$$x \mapsto f(x) = [f(x)_1, \ldots, f(x)_n] \in \mathbb{R}^n$$

model input (movie review)

high-dimensional feature vector

2. **A classification function that decides which class to apply to an instance:**

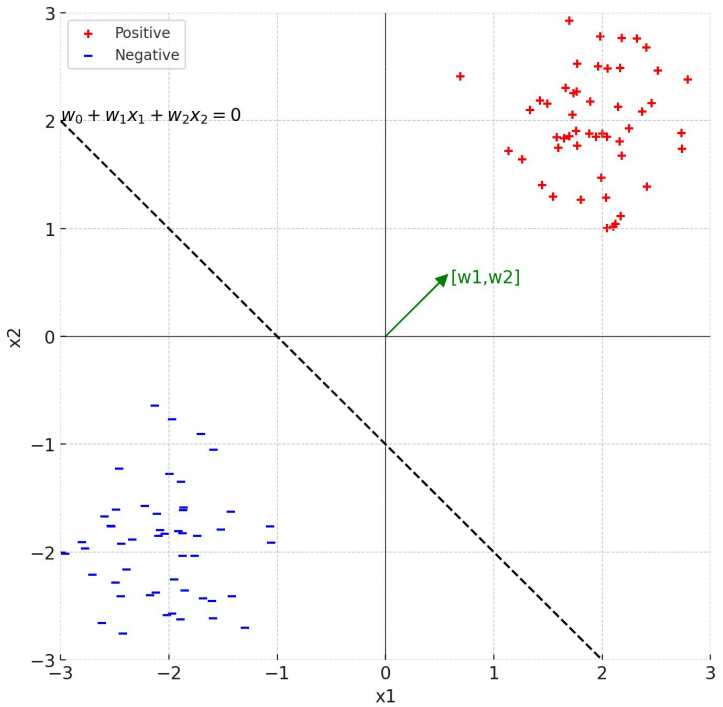


Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

The geometry of a linear classifier



$[x_1, x_2] \ldots \text{feature vector} \qquad [w_1, w_2] \ldots \text{weight vector}$

$w_0 + w_1x_1 + w_2x_2 = 0 \ldots \text{line}$

- The weight vector is of the same size as the feature vector as each weight represents the contribution of the corresponding feature to the model
- The weight vector is orthogonal to the line
- The weight vector points to the positive class
  - Dot product between the feature and weight vector is positive for positive examples
- w_0 controls the position of the decision boundary relative to the origin in the feature space

Plot generated with ChatGPT 4o

Folding in the bias term w_0

$$f(x) = [1, f(x)_1, f(x)_2, \ldots, f(x)_{d-1}] \in \mathbb{R}^d$$

$$f(x)_0 := 1$$

$$w = [w_0, w_1, \ldots, w_{d-1}] \in \mathbb{R}^d$$

$$w^T f(x) = \sum_{i=0}^{d-1} w_i f(x)_i = w_0 + w_1 x_1 + \ldots + w_{d-1} x_{d-1} \ldots \text{a dot product between } w \text{ and } f(x)$$

A general setup for binary linear classification

$$f(x) = [1, f(x)_1, f(x)_2, \ldots, f(x)_{d-1}] \in \mathbb{R}^d$$

$$f(x)_0 := 1$$

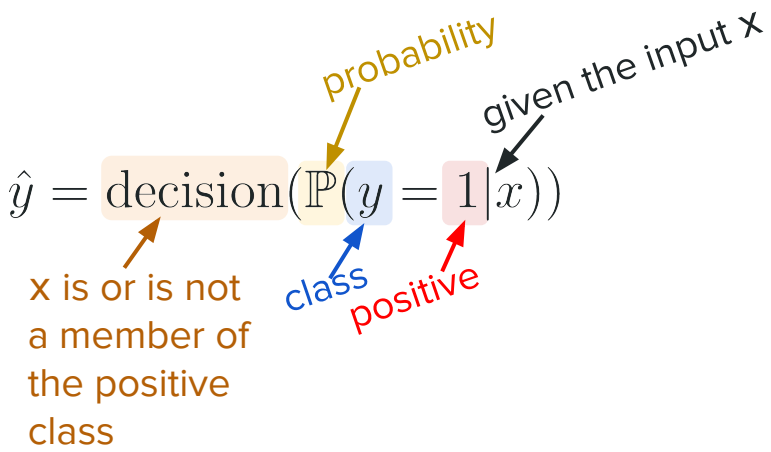
$$w = [w_0, w_1, \ldots, w_{d-1}] \in \mathbb{R}^d$$

$$w^T f(x) = \sum_{i=0}^{d-1} w_i f(x)_i = w_0 + w_1 x_1 + \ldots + w_{d-1} x_{d-1} \ldots \text{a dot product between } w \text{ and } f(x)$$

$$\hat{y} = \text{decision}(x) = \begin{cases} 1, w^T f(x) \geq 0 \\ 0, \text{else} \end{cases}$$

What's missing?

This is what I advertised...
(missing probabilities)



How this connects to:

$$\hat{y} = \text{decision}(x) = \begin{cases} 1, w^T f(x) \geq 0 \\ 0, \text{else} \end{cases}$$

The sigmoid function

$$\sigma(z) = \frac{1}{1 + e^{-z}} \ldots \text{sigmoid function}$$

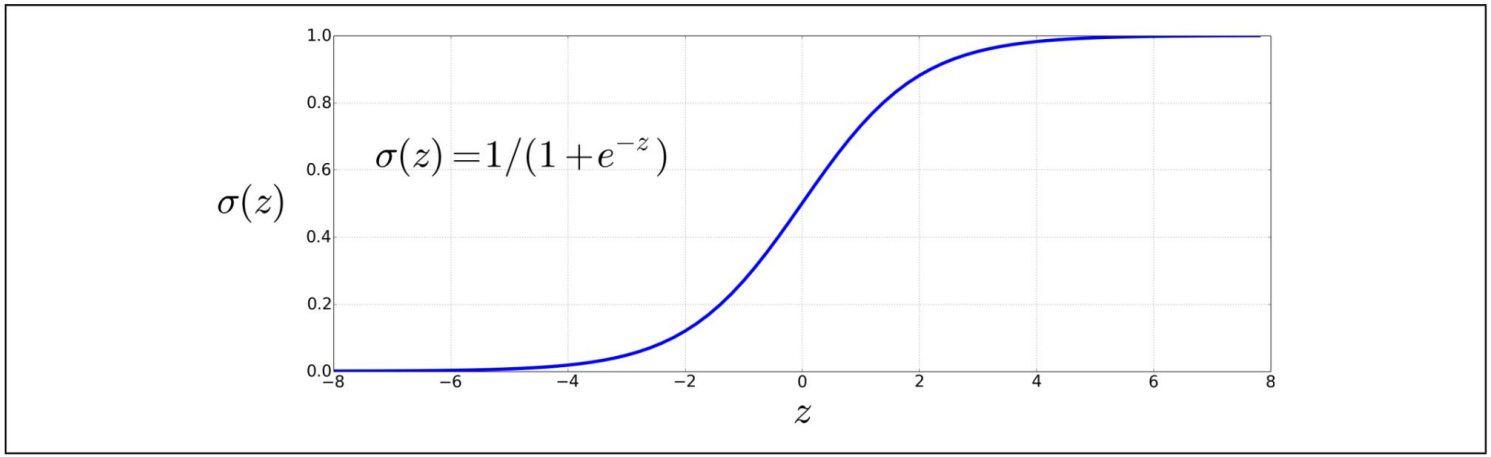


Figure 5.1 The sigmoid function $\sigma(z) = \frac{1}{1+e^{-z}}$ takes a real value and maps it to the range $(0, 1)$. It is nearly linear around 0 but outlier values get squashed toward 0 or 1.

Source: Jurafsky & Martin

Logistic Regression

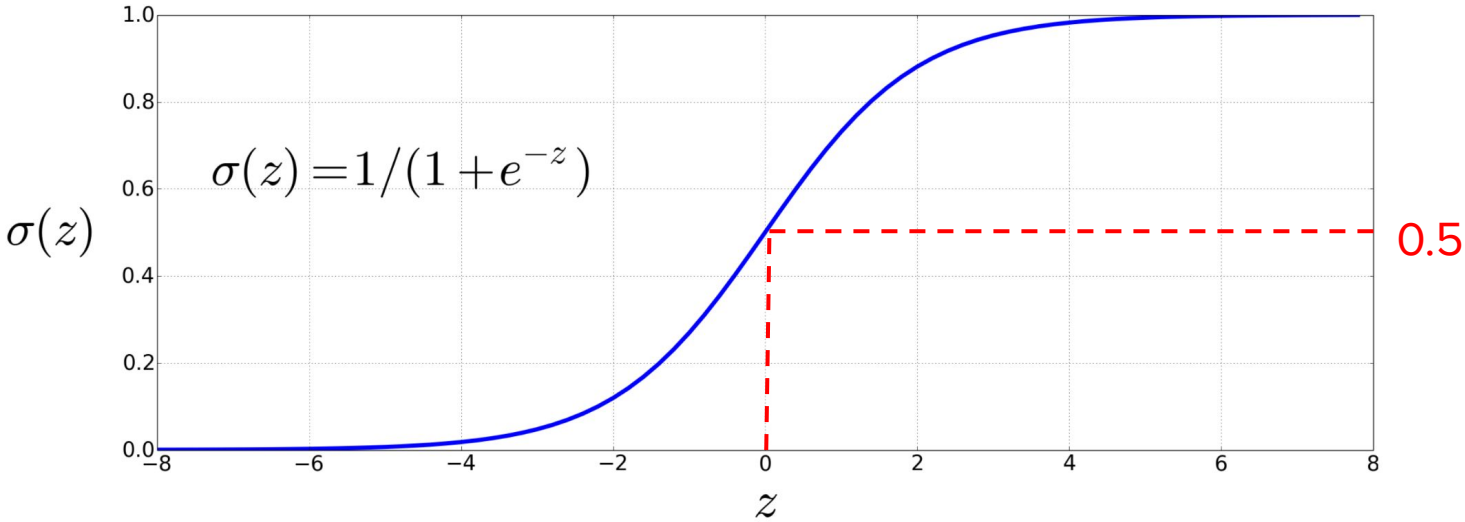
Foundation for many techniques in this course, including neural networks

$$\sigma(z) = \frac{1}{1+e^{-z}} \ldots \text{sigmoid function}$$

$$\mathbb{P}(y=1|x) = \sigma(w^T f(x)) = \frac{1}{1+e^{-w^T f(x)}}$$

$$\mathbb{P}(y=0|x) = 1 - \sigma(w^T f(x)) = 1 - \frac{1}{1+e^{-w^T f(x)}} = \frac{e^{-w^T f(x)}}{1+e^{-w^T f(x)}}$$

Logistic Regression



$$w^T f(x) > 0 \quad \Rightarrow \quad \mathbb{P}(y=1|x) = \sigma(w^T f(x)) > 0.5$$

$$\hat{y} = \text{decision}(x) = \begin{cases} 1, \mathbb{P}(y=1|x) > 0.5 \\ 0, \text{else} \end{cases}$$

4 components of a supervised ML for binary classification

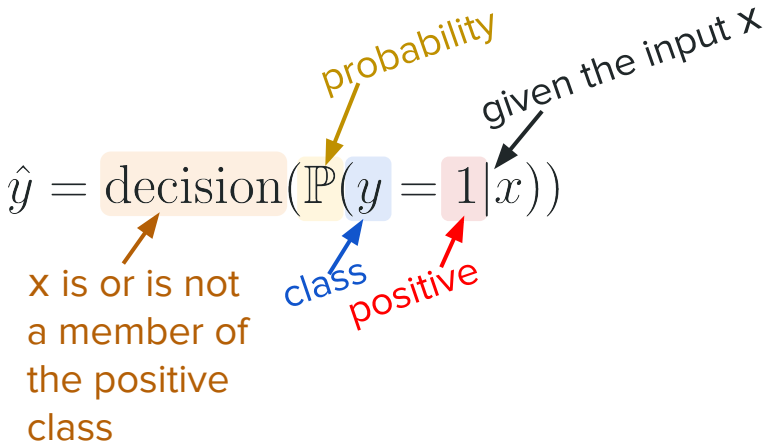
1. **A feature representation of the input**

$$x \mapsto f(x) = [f(x)_1, \ldots, f(x)_n] \in \mathbb{R}^n$$

model input (movie review)

high-dimensional feature vector

2. **A classification function that decides which class to apply to an instance:**



4 components of a supervised ML for binary classification

How do we find the weight vectors?

3. An objective function that we want to optimize for to learn appropriate model parameters
4. An algorithm for finding optimal model parameters/weights for the objective function

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

Training Logistic Regression

$(x^{(i)}, y^{(i)})_{i=1}^{D} \ldots \text{training dataset}$

$$\max_w \prod_{i=1}^{D} \mathbb{P}(y^{(i)}|x^{(i)}, w) \ldots \text{max. likelihood}$$

$$\max_w \sum_{i=1}^{D} \log\mathbb{P}(y^{(i)}|x^{(i)}, w) \ldots \text{max. log-likelihood}$$

$$\min_w \sum_{i=1}^{D} -\log\mathbb{P}(y^{(i)}|x^{(i)}, w) \ldots \text{min. negative log-likelihood (NLL)}$$

$$\min_w \sum_{i=1}^{D} \text{loss}(x^{(i)}, y^{(i)}, w) \ldots \text{training objective}$$

4 components of a supervised ML for binary classification

1. A feature representation of the input
2. A classification function that decides which class to apply to an instance
3. An objective function that we want to optimize for to learn appropriate model parameters
4. An algorithm for finding optimal model parameters/weights for the objective function

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score

Gradient Descent: Intuition

Goal: $w^* = \text{argmin}_w \ \text{loss}(w)$

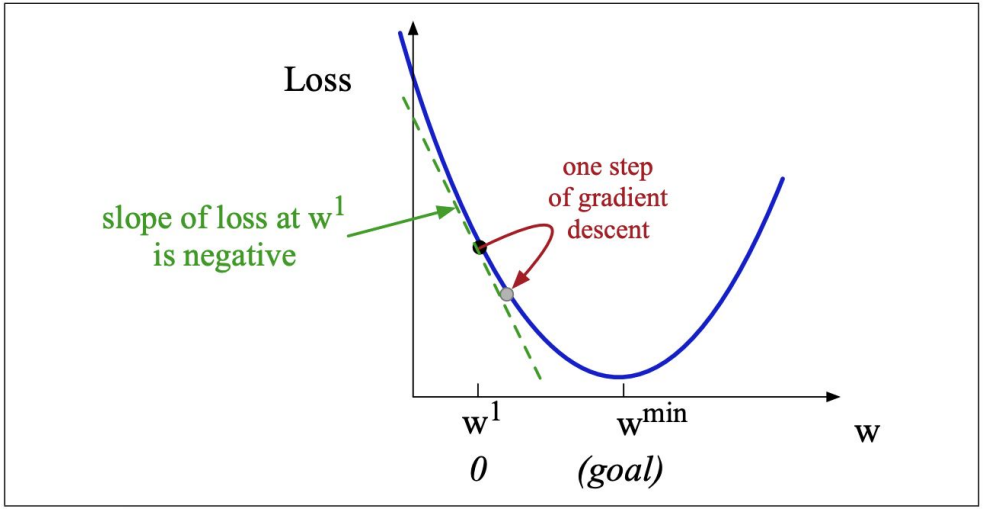


Figure 5.4 The first step in iteratively finding the minimum of this loss function, by moving w in the reverse direction from the slope of the function. Since the slope is negative, we need to move w in a positive direction, to the right. Here superscripts are used for learning steps, so w^1 means the initial value of w (which is 0), w^2 the value at the second step, and so on.

Source: Jurafsky & Martin

Gradient Descent: Algorithm

```
init parameters w (e.g. to zeros)
choose a learning rate α (e.g. to 0.1)
for _ in n_epochs:
    for (x_i, y_i) in shuffled_train_data:
        Optionally tweak the learning rate
        Compute gradient ∇w of the loss function L(x_i, y_i, w) with respect to
w
        Update the parameters: w = w - α * ∇_w L
```

Logistic Regression Gradients

$$\frac{\partial}{\partial w}\text{loss}(x^{(i)}, y^{(i)}, w) = \frac{\partial}{\partial w}\Big(-\log \mathbb{P}(y^{(i)}|x^{(i)}, w)\Big)$$

$$y^{(i)} = +1 \Rightarrow \frac{\partial}{\partial w}\Big(-\log \mathbb{P}(y^{(i)}|x^{(i)}, w)\Big) = \frac{\partial}{\partial w}\Big(-\log \frac{1}{1+e^{-w^T f(x)}}\Big) =$$

$$= \frac{\partial}{\partial w}\Big(\log(1+e^{-w^T f(x)})\Big) =$$

$$= \frac{1}{1+e^{-w^T f(x)}} \cdot e^{-w^T f(x)} \cdot (-f(x)) =$$

$$= -\mathbb{P}(y^{(i)} = 0|x^{(i)}) f(x) =$$

$$= \Big(\mathbb{P}(y^{(i)} = +1|x^{(i)}) - 1\Big) f(x)$$

Logistic Regression GD Updates

$$y^{(i)} = +1 \Rightarrow \frac{\partial}{\partial w}\Big(-\log \mathbb{P}(y^{(i)}|x^{(i)}, w)\Big) = f(x)\Big(\mathbb{P}(y^{(i)} = +1|x^{(i)}) - 1\Big)$$

$$y^{(i)} = 0 \Rightarrow \frac{\partial}{\partial w}\big(-\log \mathbb{P}(y^{(i)}|x^{(i)}, w)\big) = \ldots = \big(1 - \mathbb{P}(y^{(i)} = 0|x^{(i)})\big) f(x)$$

```
init parameters w (e.g. to zeros)
choose a learning rate α (e.g. to 0.1)
for _ in n_epochs:
    for (x_i, y_i) in shuffled_train_data:
        Optionally tweak the learning rate
        Compute gradient ∇w of the loss function L(x_i, y_i, w) with respect to
w
        (different equations based on what the true label y_i is)
        Update the parameters: w = w - α * ∇wL
```

4 components of a supervised ML for binary classification

1. A feature representation of the input
2. A classification function that decides which class to apply to an instance
3. An objective function that we want to optimize for to learn appropriate model parameters
4. An algorithm for finding optimal model parameters/weights for the objective function

Goals & Overview of Today's Lecture

Goal: Learn components of a supervised machine learning system for binary text classification

- Components of a supervised machine learning system for classification
- Feature representations of the input
- The sigmoid function
- Negative log-likelihood / cross-entropy loss
- Stochastic gradient descent
- Accuracy & F1-score (next time ☺️)