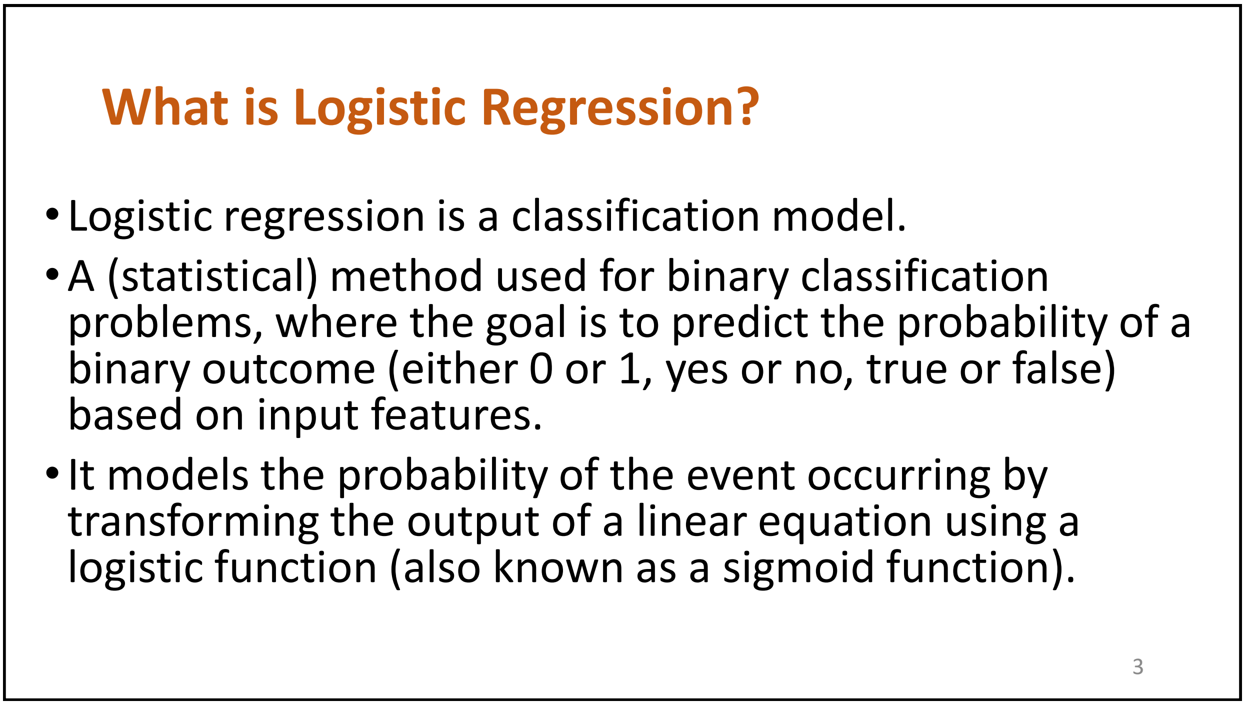
EECE-6820 Machine Learning

Classification and Logistic Regression

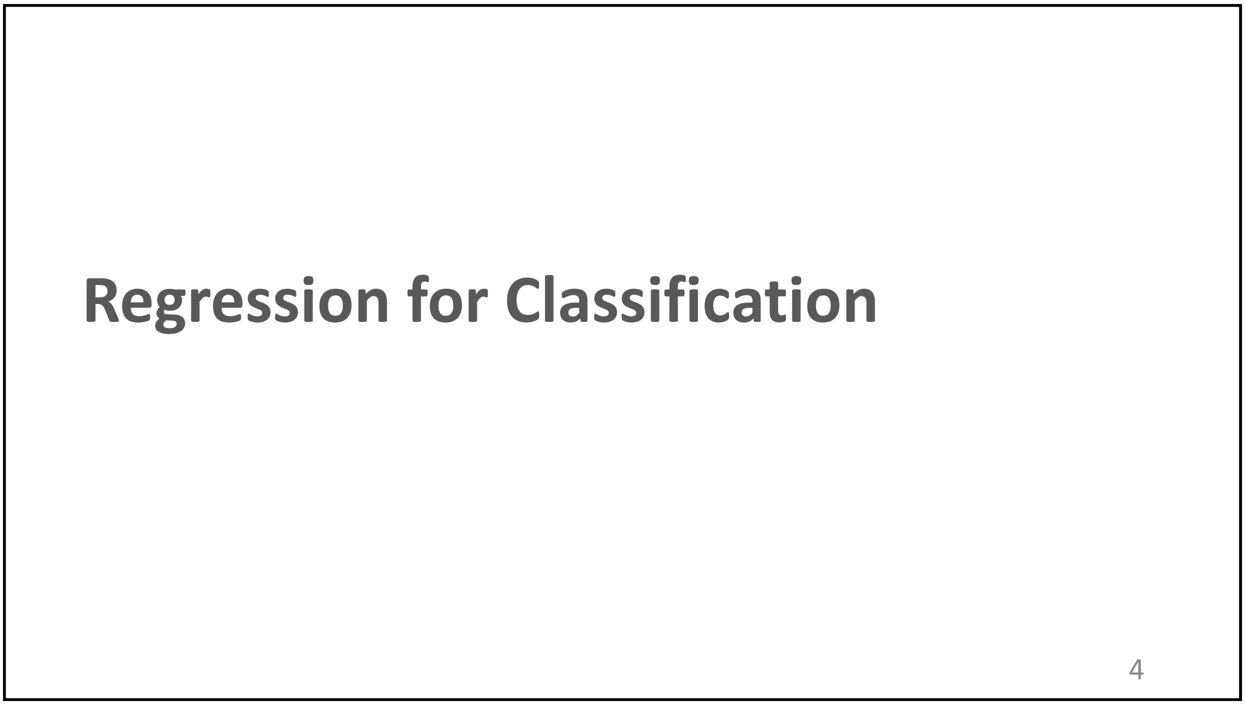
Cris Ababei
Dept. of Electrical and Computer Engineering
MARQUETTE UNIVERSITY
BE THE DIFFERENCE.

PART 1
A Lighter Presentation

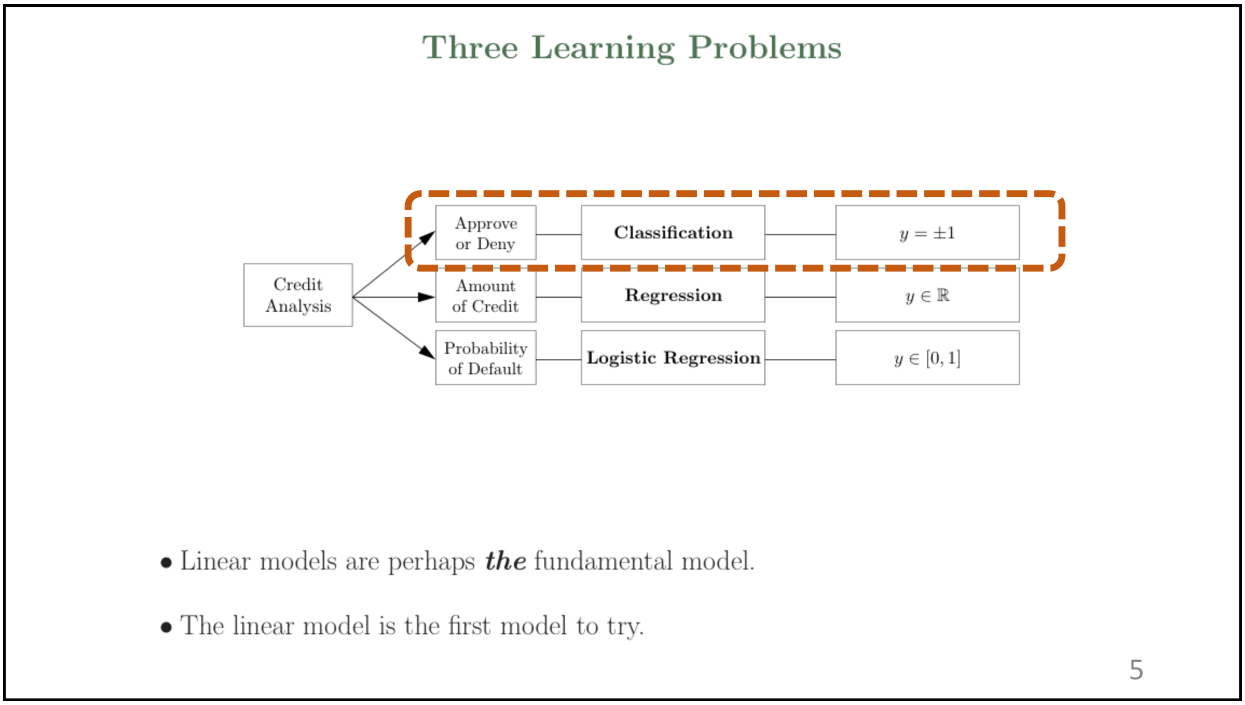


What is Logistic Regression?

- Logistic regression is a classification model.
- A (statistical) method used for binary classification problems, where the goal is to predict the probability of a binary outcome (either 0 or 1, yes or no, true or false) based on input features.
- It models the probability of the event occurring by transforming the output of a linear equation using a logistic function (also known as a sigmoid function).



Regression for Classification



Three Learning Problems

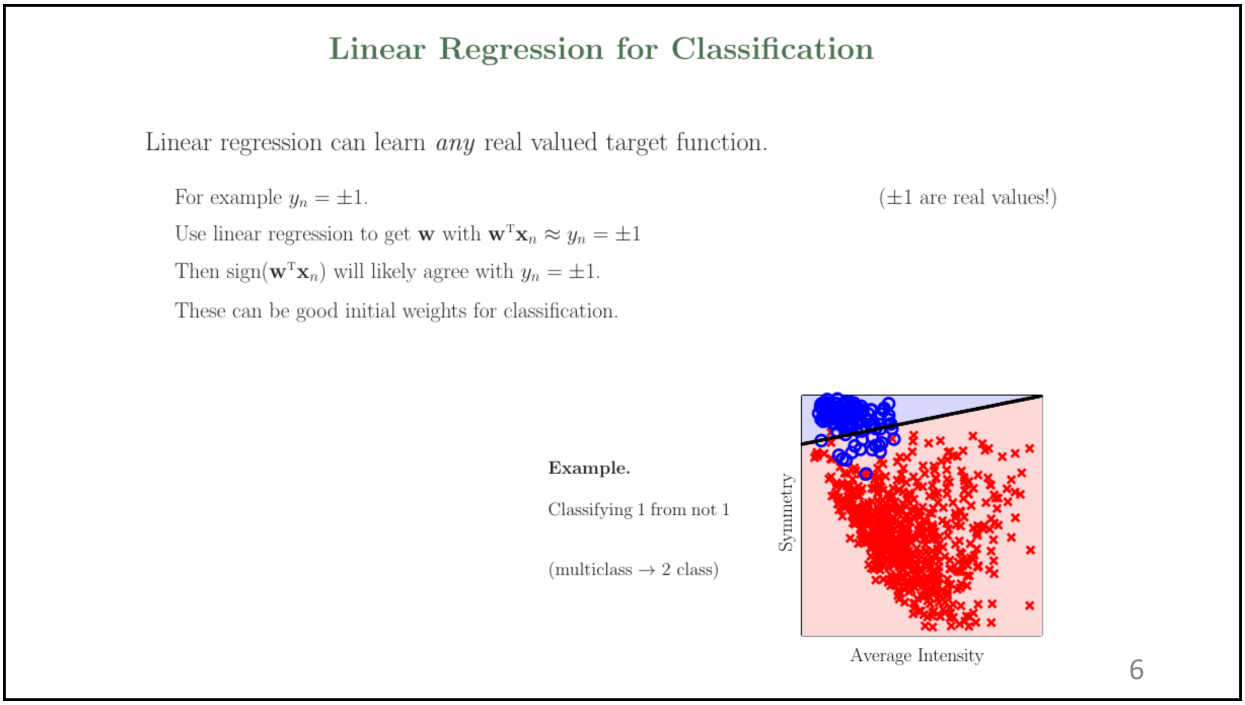
Credit Analysis

Approve or Deny — **Classification** — $y = \pm 1$

Amount of Credit — **Regression** — $y \in \mathbb{R}$

Probability of Default — **Logistic Regression** — $y \in [0, 1]$

- Linear models are perhaps ***the*** fundamental model.
- The linear model is the first model to try.



Linear Regression for Classification

Linear regression can learn *any* real valued target function.

For example $y_n = \pm 1$. (± 1 are real values!)

Use linear regression to get $\mathbf{w}$ with $\mathbf{w}^{\mathrm{T}}\mathbf{x}_n \approx y_n = \pm 1$

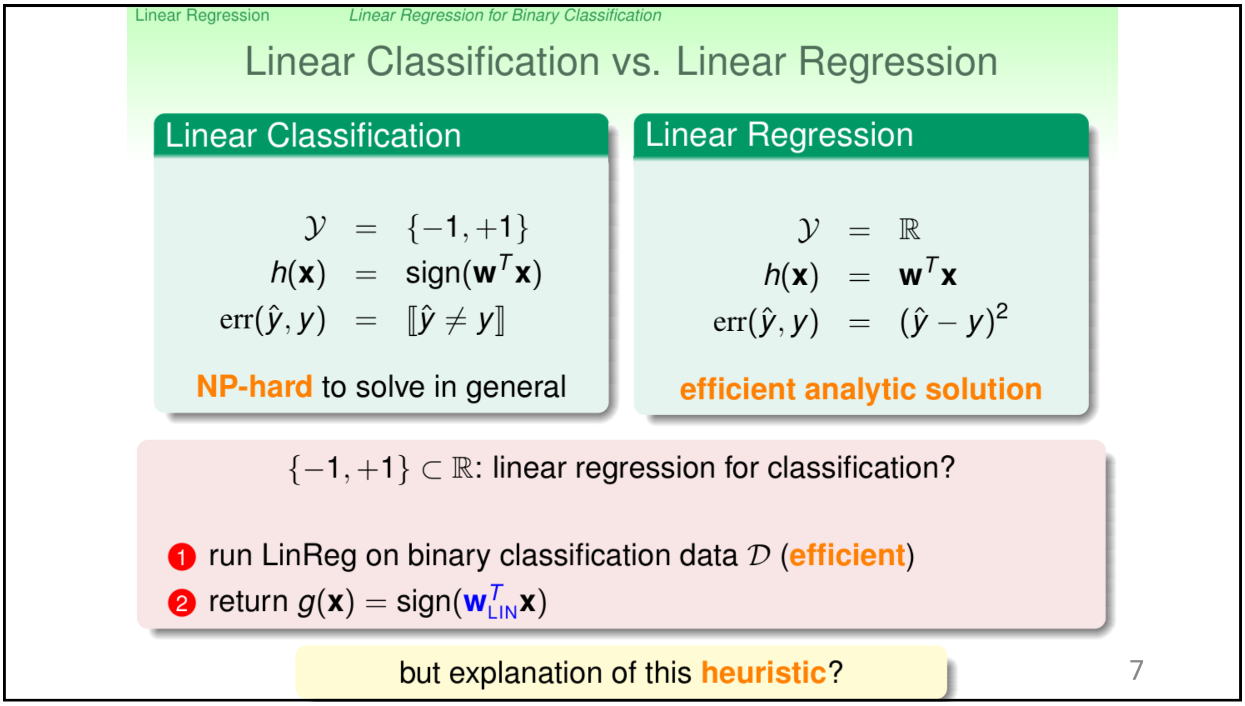
Then $\text{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x}_n)$ will likely agree with $y_n = \pm 1$.

These can be good initial weights for classification.

Example.

Classifying 1 from not 1

(multiclass → 2 class)



Linear Classification vs. Linear Regression

Linear Classification

$$\begin{aligned} \mathcal{Y} &= \{-1,+1\} \\ h(\mathbf{x}) &= \text{sign}(\mathbf{w}^T\mathbf{x}) \\ \text{err}(\hat{y},y) &= [\![\hat{y} \neq y]\!] \end{aligned}$$

NP-hard to solve in general

Linear Regression

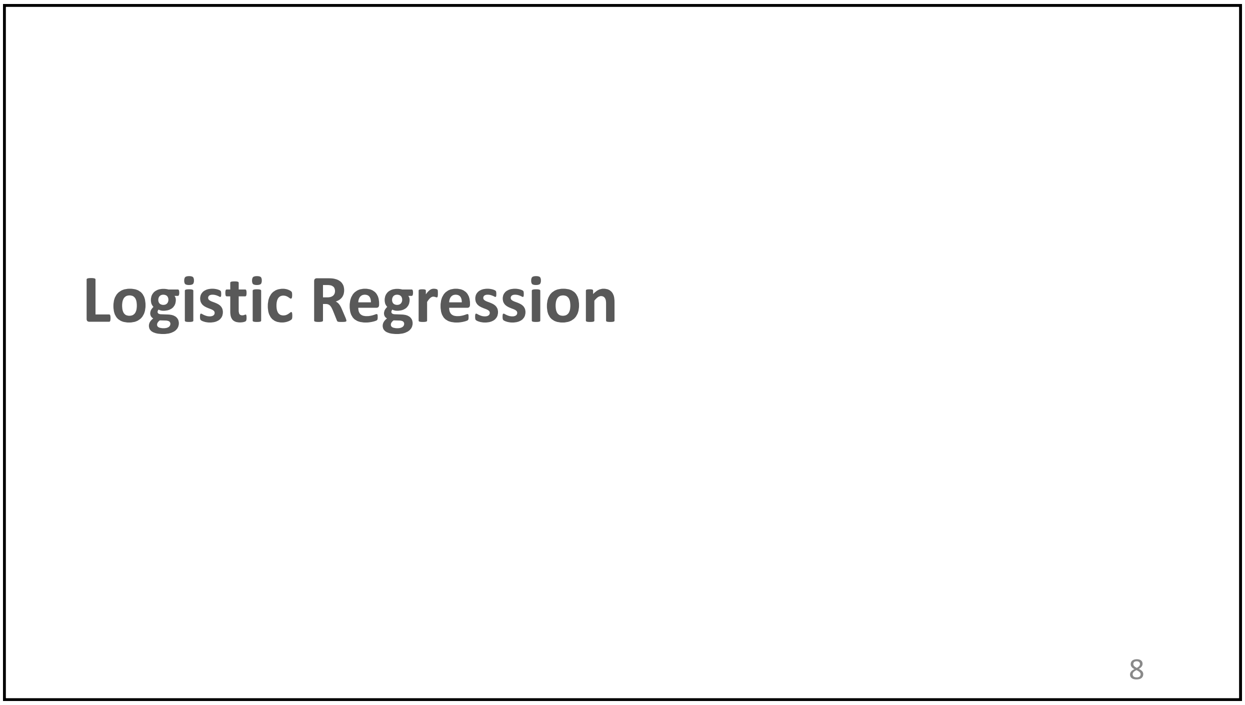
$$\begin{aligned} \mathcal{Y} &= \mathbb{R} \\ h(\mathbf{x}) &= \mathbf{w}^T\mathbf{x} \\ \text{err}(\hat{y},y) &= (\hat{y}-y)^2 \end{aligned}$$

efficient analytic solution

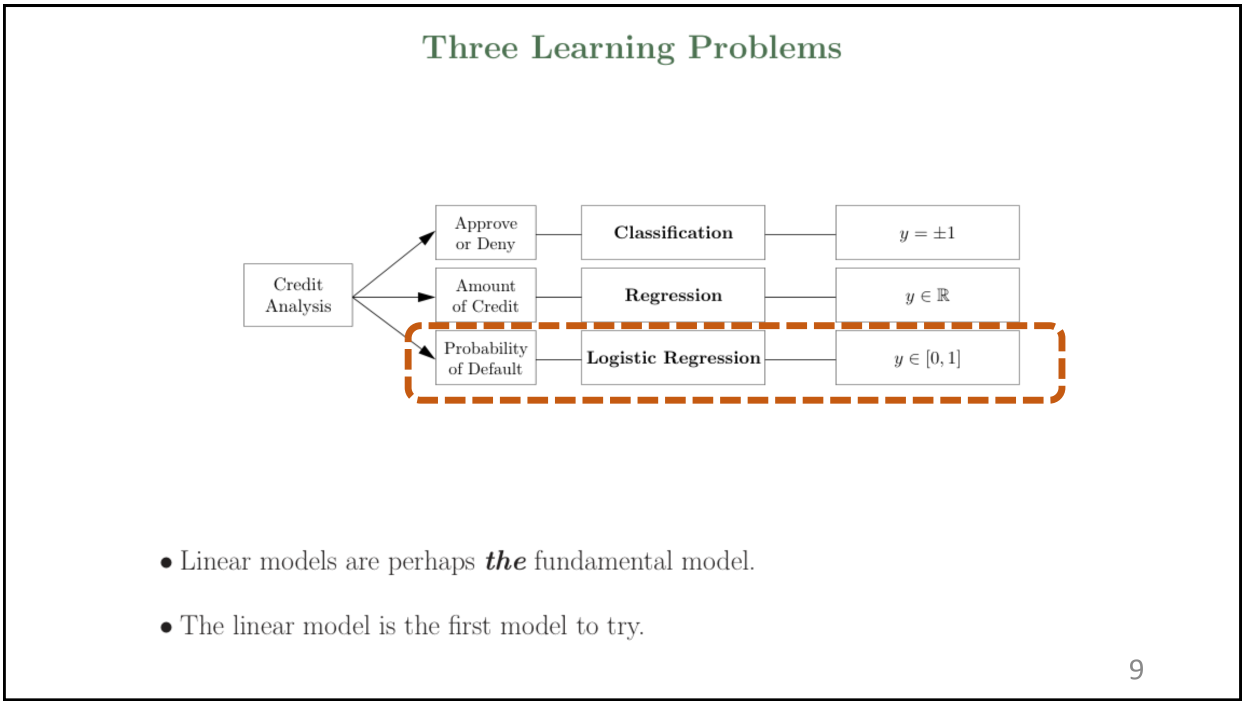
$\{-1,+1\} \subset \mathbb{R}$: linear regression for classification?

1. run LinReg on binary classification data $\mathcal{D}$ (**efficient**)
2. return $g(\mathbf{x}) = \text{sign}(\mathbf{w}_{\text{LIN}}^T\mathbf{x})$

but explanation of this **heuristic**?



Logistic Regression



Three Learning Problems

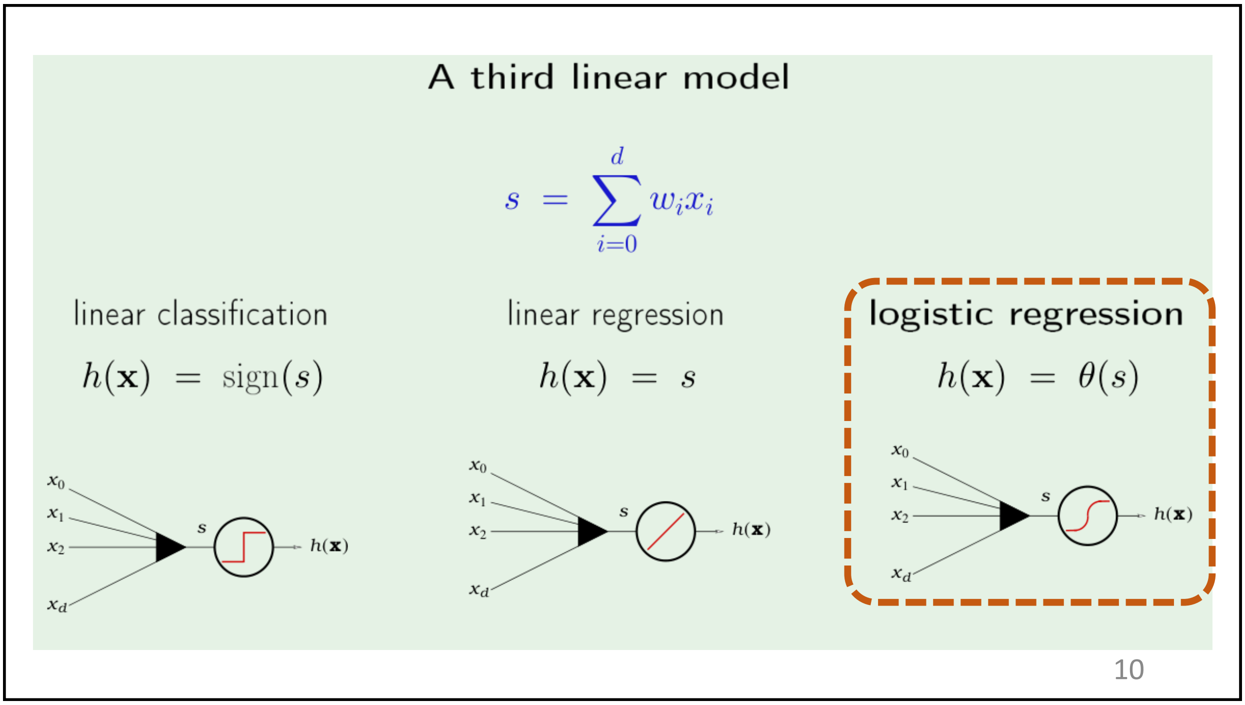
Credit Analysis

Approve or Deny	**Classification**	$y = \pm 1$
Amount of Credit	**Regression**	$y \in \mathbb{R}$
Probability of Default	**Logistic Regression**	$y \in [0, 1]$

- Linear models are perhaps ***the*** fundamental model.
- The linear model is the first model to try.

9

9



A third linear model

$$s = \sum_{i=0}^{d} w_i x_i$$

linear classification

$$h(\mathbf{x}) = \text{sign}(s)$$

linear regression

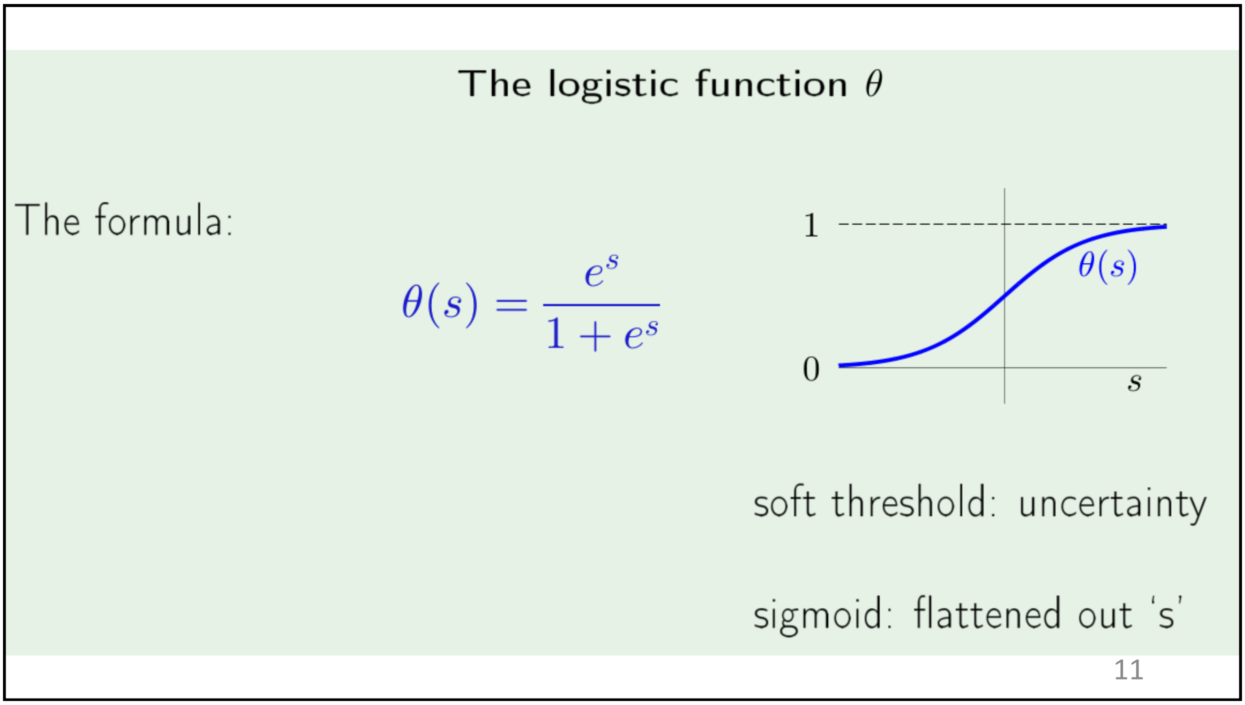
$$h(\mathbf{x}) = s$$

logistic regression

$$h(\mathbf{x}) = \theta(s)$$

10

10



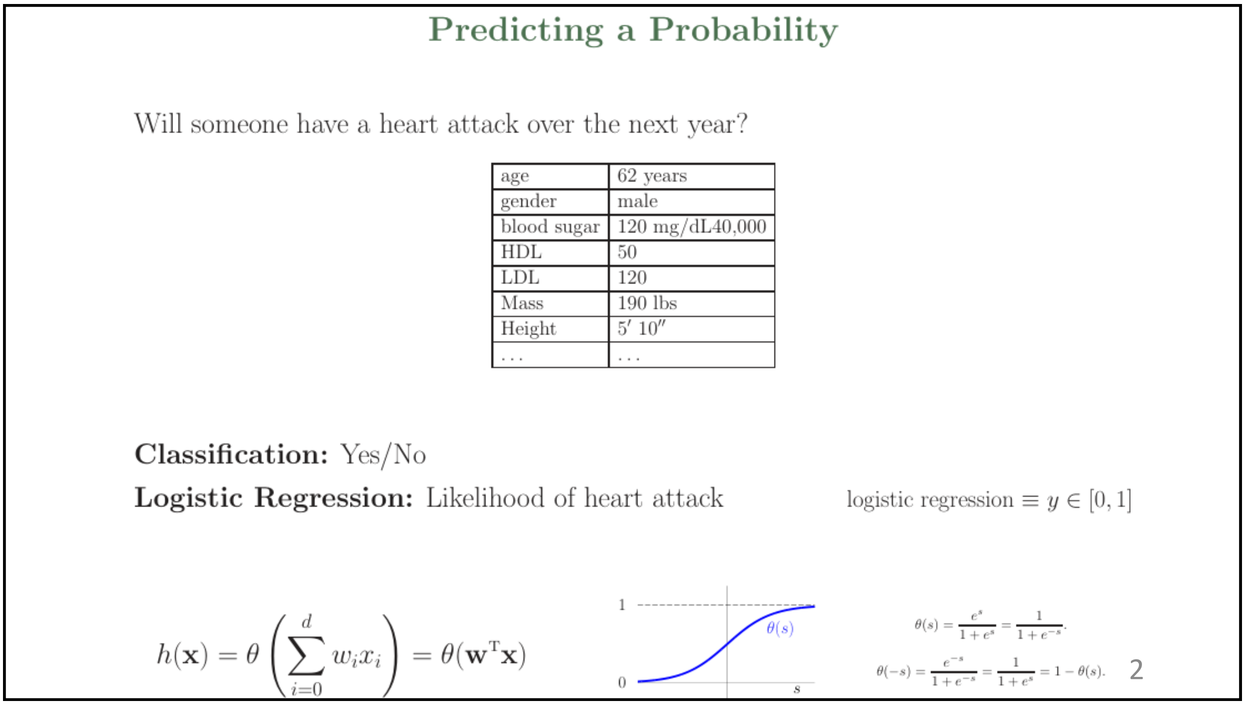
The logistic function θ

The formula:

$$\theta(s) = \frac{e^s}{1+e^s}$$

soft threshold: uncertainty

sigmoid: flattened out 's'



Predicting a Probability

Will someone have a heart attack over the next year?

age	62 years
gender	male
blood sugar	120 mg/dL40,000
HDL	50
LDL	120
Mass	190 lbs
Height	5′ 10″
. . .	. . .

Classification: Yes/No

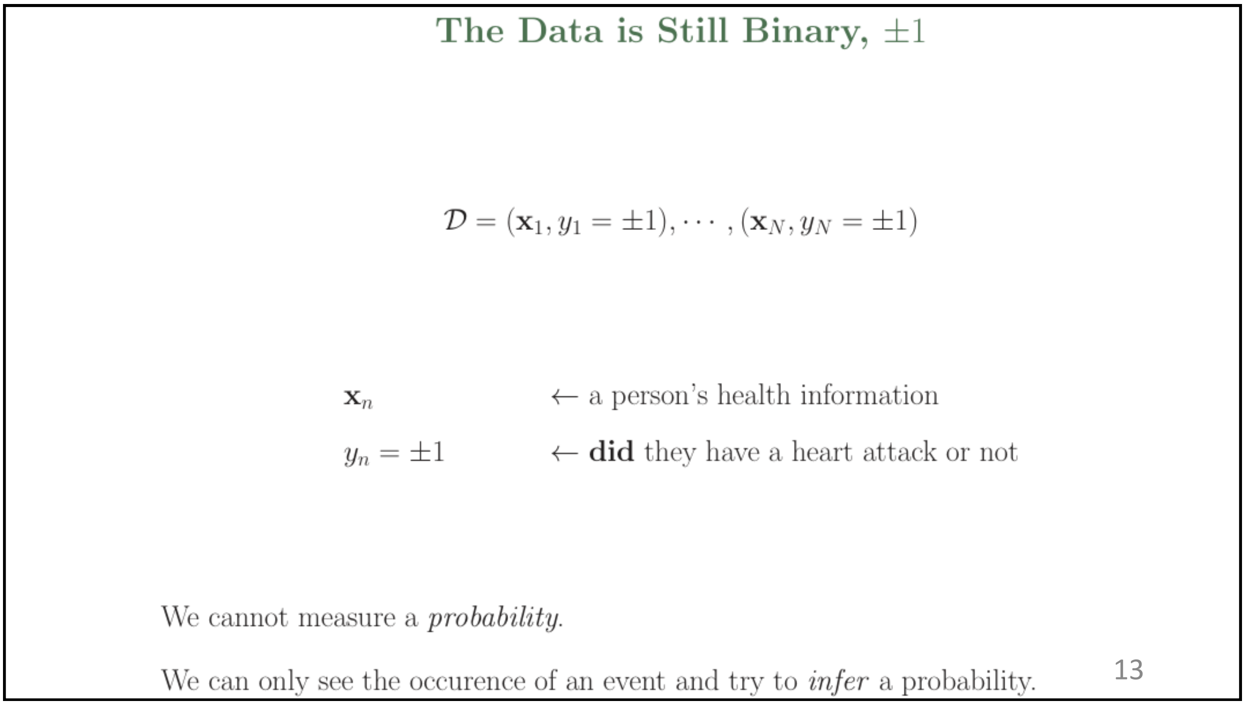
Logistic Regression: Likelihood of heart attack

logistic regression $\equiv y \in [0,1]$

$$h(\mathbf{x}) = \theta\left(\sum_{i=0}^{d} w_i x_i\right) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$$

$$\theta(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}.$$

$$\theta(-s) = \frac{e^{-s}}{1+e^{-s}} = \frac{1}{1+e^{s}} = 1 - \theta(s).$$



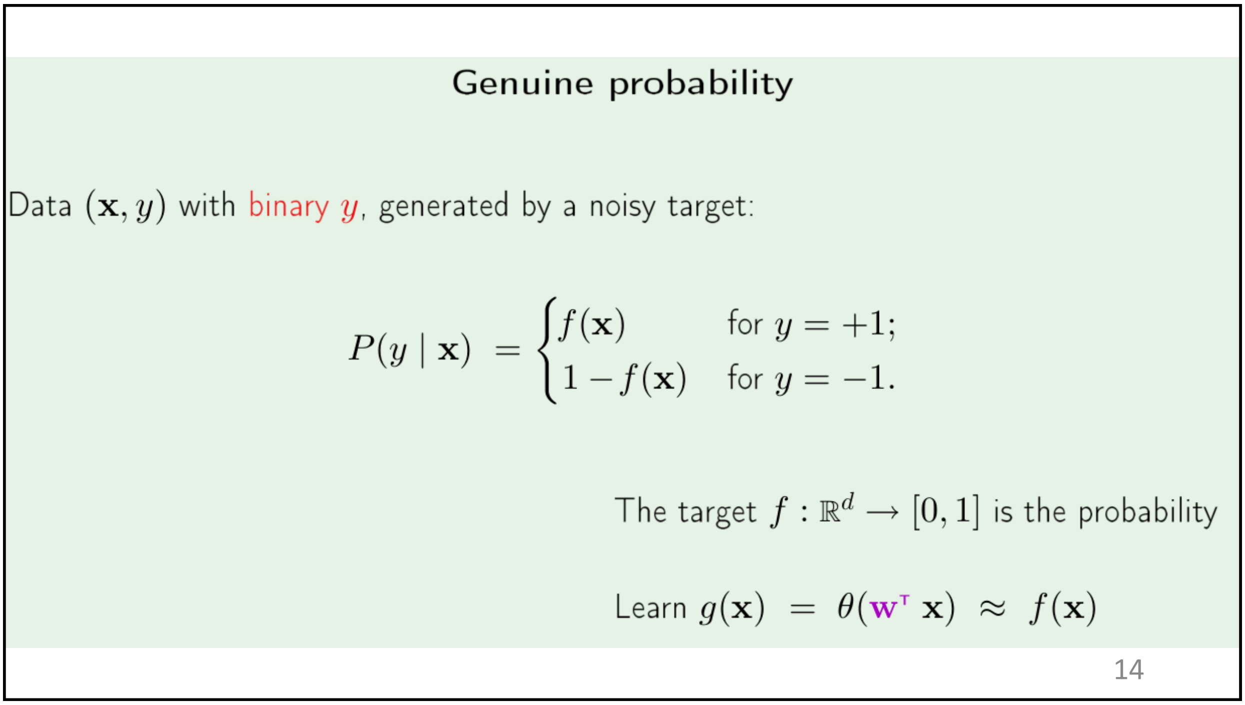
The Data is Still Binary, ±1

$$\mathcal{D} = (\mathbf{x}_1, y_1 = \pm 1), \cdots, (\mathbf{x}_N, y_N = \pm 1)$$

$\mathbf{x}_n$	← a person's health information
$y_n = \pm 1$	← **did** they have a heart attack or not

We cannot measure a *probability.*

We can only see the occurence of an event and try to *infer* a probability.



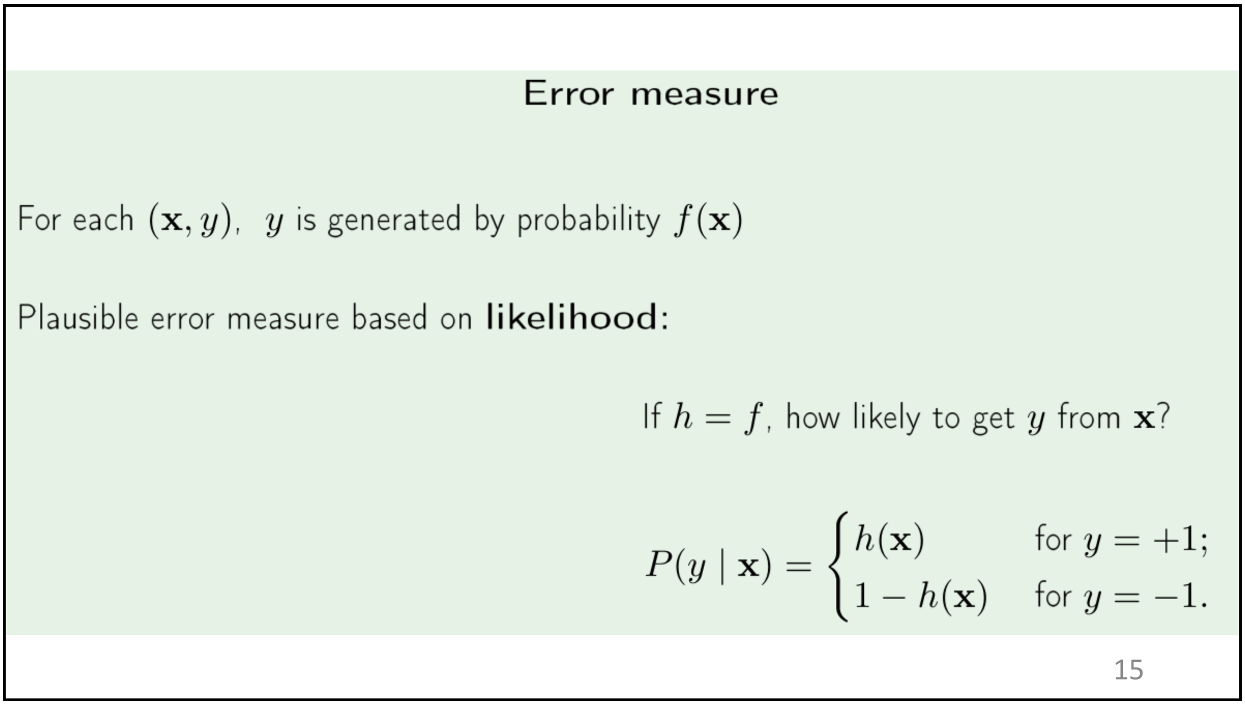
Genuine probability

Data $(\mathbf{x}, y)$ with binary y, generated by a noisy target:

$$P(y \mid \mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{for } y = +1; \\ 1 - f(\mathbf{x}) & \text{for } y = -1. \end{cases}$$

The target $f : \mathbb{R}^d \to [0, 1]$ is the probability

Learn $g(\mathbf{x}) = \theta(\mathbf{w}^{\intercal}\mathbf{x}) \approx f(\mathbf{x})$



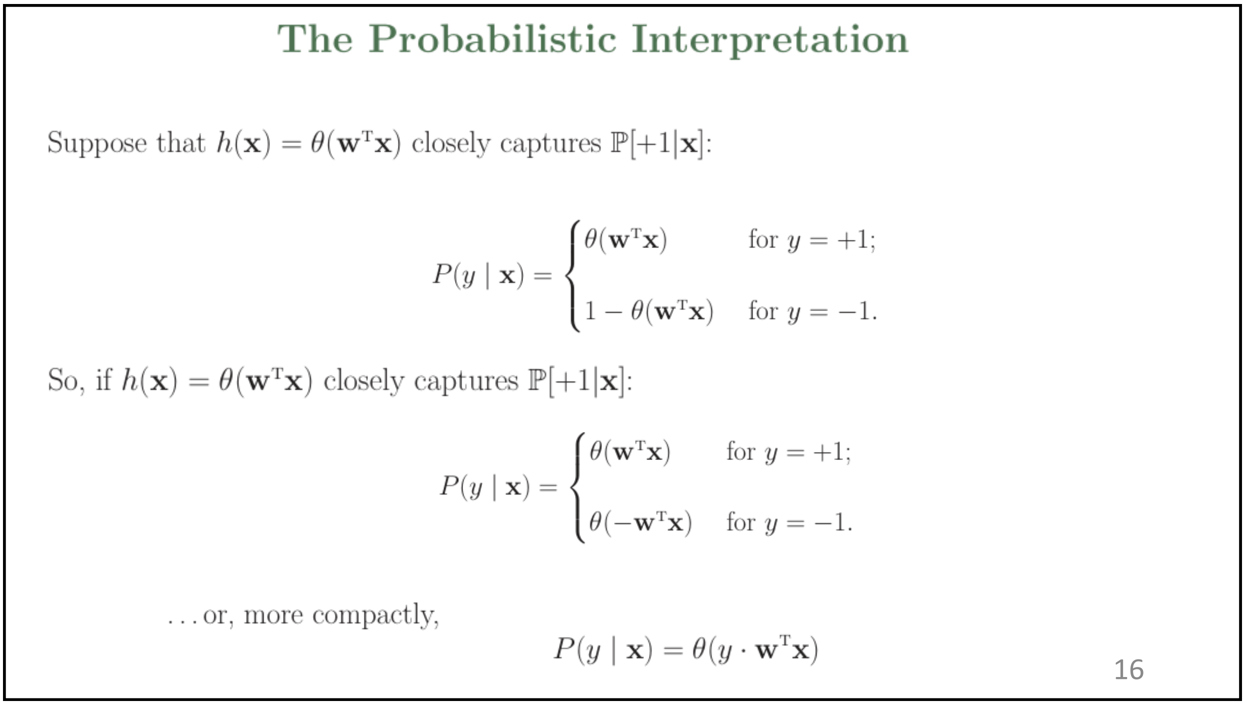
Error measure

For each $(\mathbf{x}, y)$, y is generated by probability $f(\mathbf{x})$

Plausible error measure based on **likelihood:**

If $h = f$, how likely to get y from $\mathbf{x}$?

$$P(y \mid \mathbf{x}) = \begin{cases} h(\mathbf{x}) & \text{for } y = +1; \\ 1 - h(\mathbf{x}) & \text{for } y = -1. \end{cases}$$



The Probabilistic Interpretation

Suppose that $h(\mathbf{x}) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ closely captures $\mathbb{P}[+1|\mathbf{x}]$:

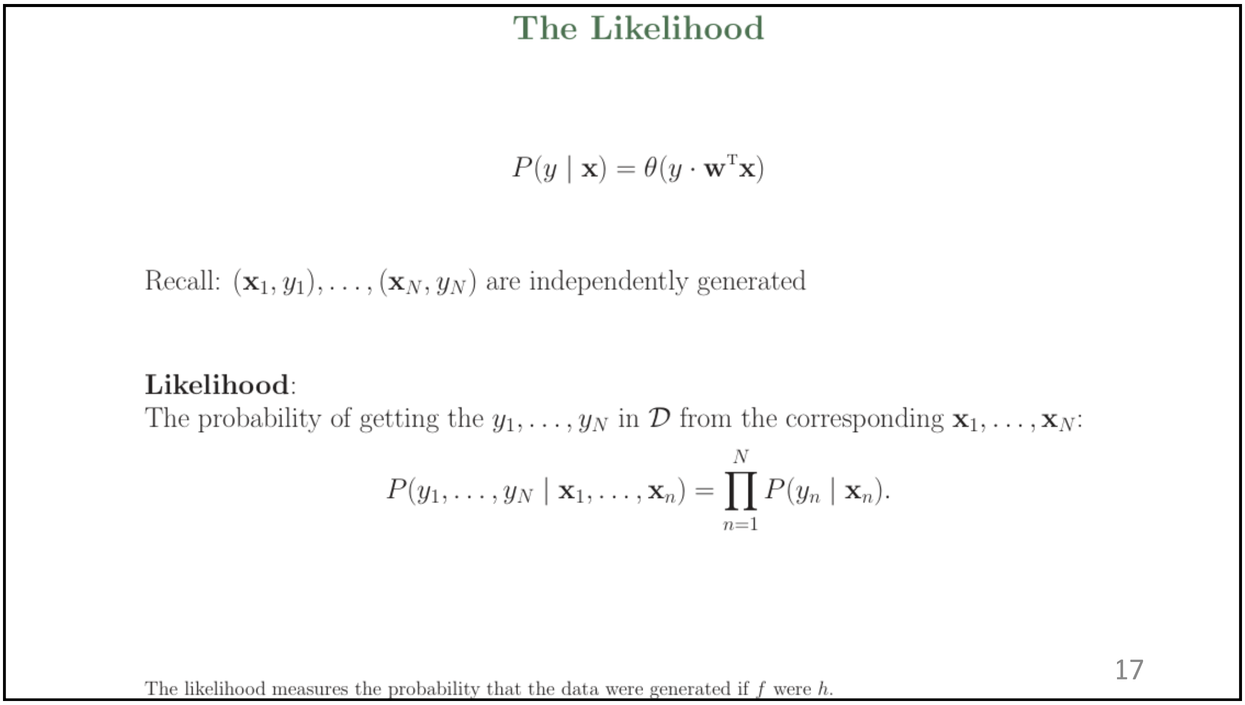
$$P(y \mid \mathbf{x}) = \begin{cases} \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = +1; \\ 1 - \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = -1. \end{cases}$$

So, if $h(\mathbf{x}) = \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x})$ closely captures $\mathbb{P}[+1|\mathbf{x}]$:

$$P(y \mid \mathbf{x}) = \begin{cases} \theta(\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = +1; \\ \theta(-\mathbf{w}^{\mathrm{T}}\mathbf{x}) & \text{for } y = -1. \end{cases}$$

. . . or, more compactly,

$$P(y \mid \mathbf{x}) = \theta(y \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x})$$



The Likelihood

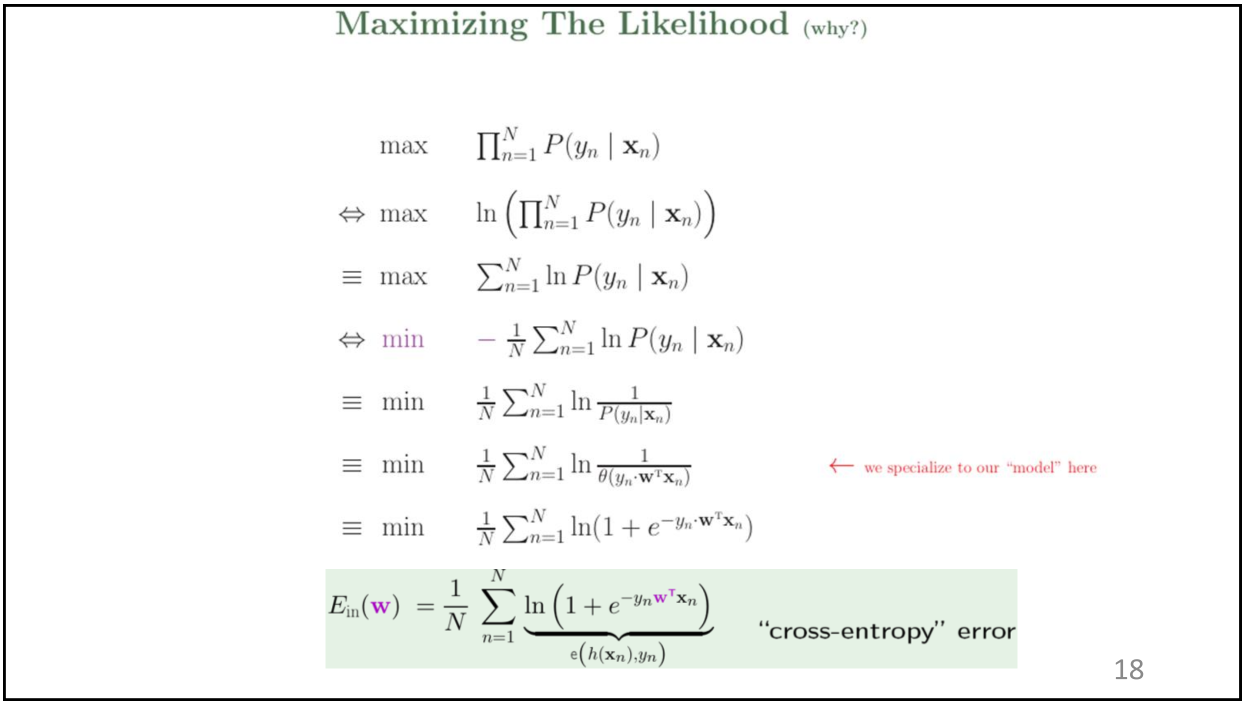
$$P(y \mid \mathbf{x}) = \theta(y \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x})$$

Recall: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ are independently generated

Likelihood:
The probability of getting the $y_1, \ldots, y_N$ in $\mathcal{D}$ from the corresponding $\mathbf{x}_1, \ldots, \mathbf{x}_N$:

$$P(y_1, \ldots, y_N \mid \mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n).$$

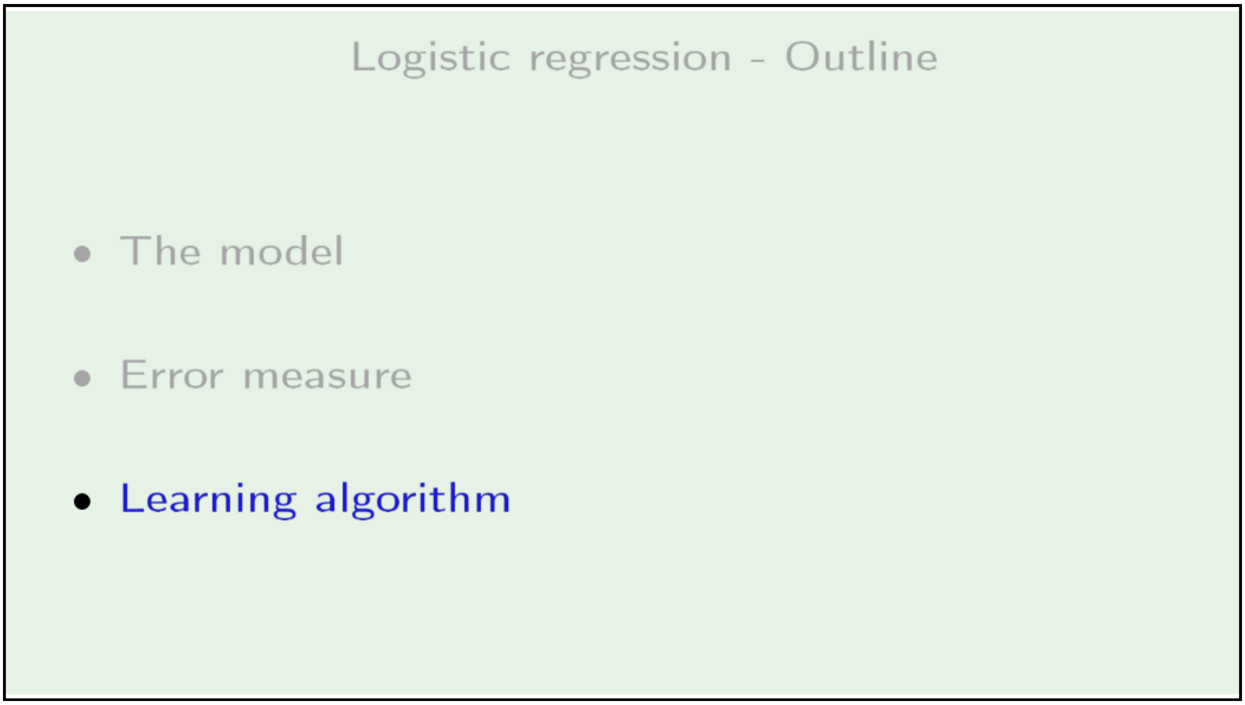
The likelihood measures the probability that the data were generated if f were h.



Maximizing The Likelihood (why?)

$$\begin{array}{lll}
 & \max & \prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow & \max & \ln\left(\prod_{n=1}^{N} P(y_n \mid \mathbf{x}_n)\right) \\
\equiv & \max & \sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\Leftrightarrow & \min & -\frac{1}{N}\sum_{n=1}^{N} \ln P(y_n \mid \mathbf{x}_n) \\
\equiv & \min & \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{P(y_n \mid \mathbf{x}_n)} \\
\equiv & \min & \frac{1}{N}\sum_{n=1}^{N} \ln \frac{1}{\theta(y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n)} \qquad \leftarrow \text{we specialize to our ``model'' here} \\
\equiv & \min & \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}_n})
\end{array}$$

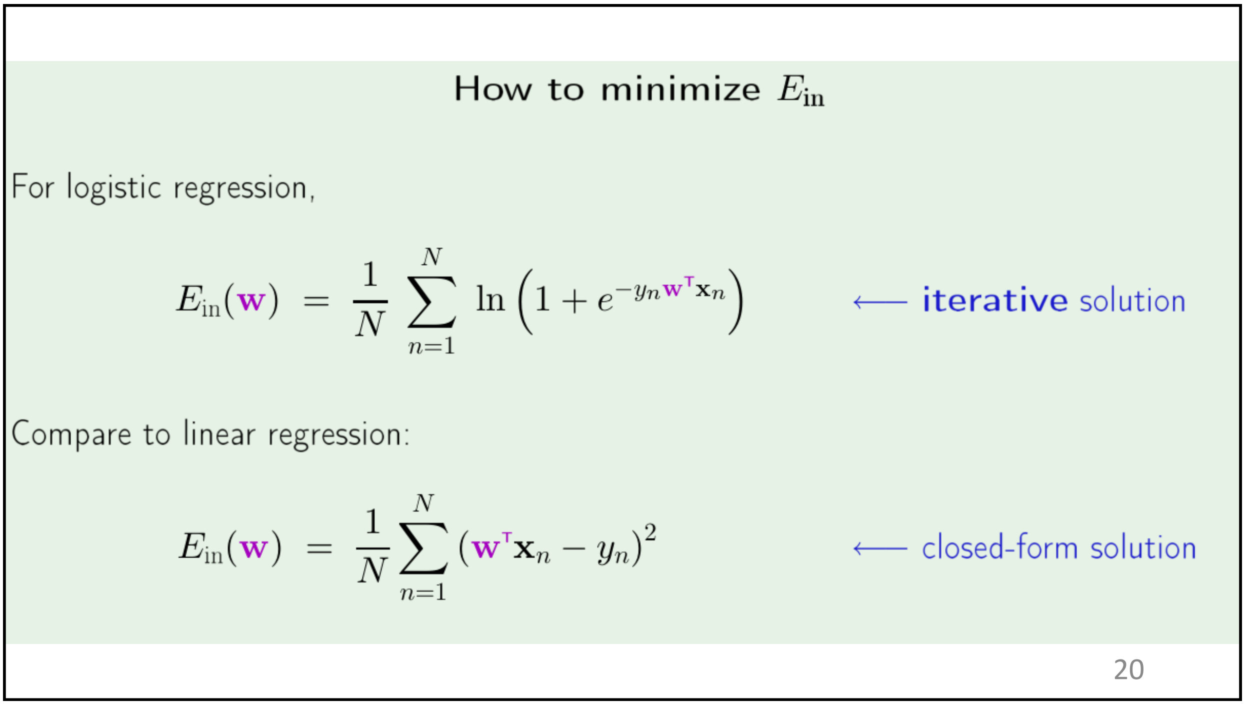
$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \underbrace{\ln\left(1 + e^{-y_n \mathbf{w}^{\mathrm{T}}\mathbf{x}_n}\right)}_{\mathrm{e}\left(h(\mathbf{x}_n), y_n\right)} \qquad \text{``cross-entropy'' error}$$



Logistic regression - Outline

- The model
- Error measure
- Learning algorithm

19



How to minimize E_{in}

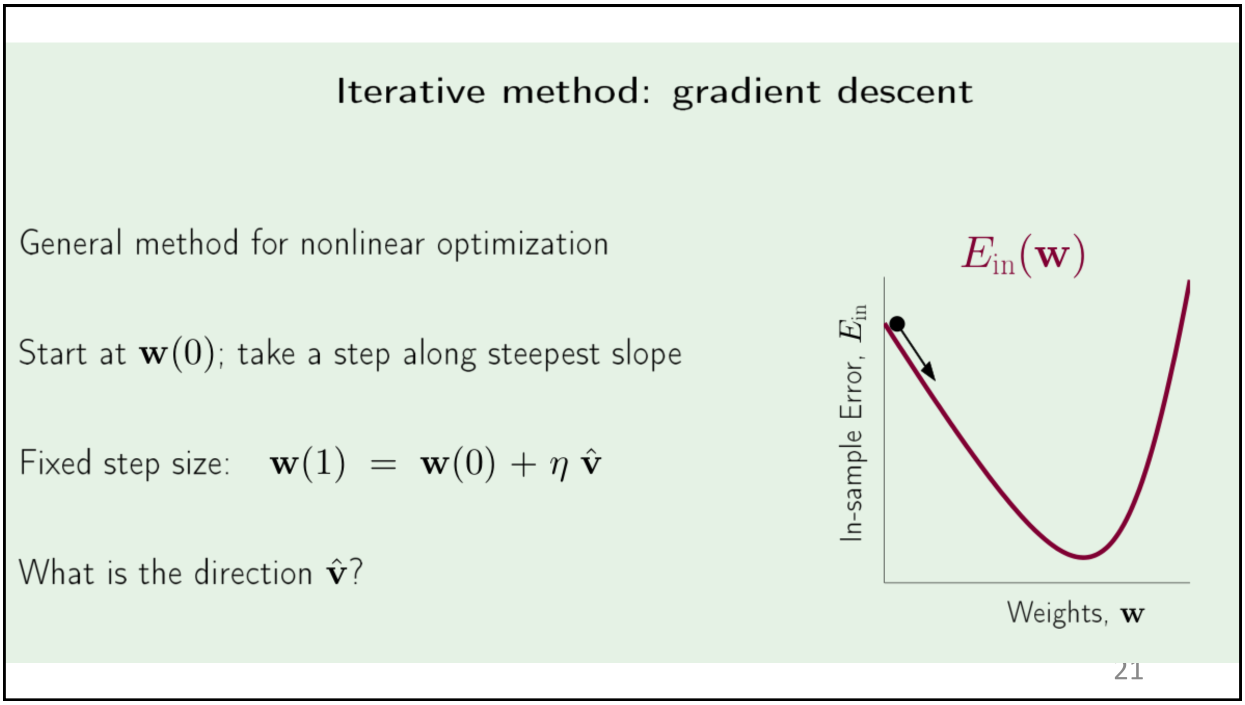
For logistic regression,

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + e^{-y_n \mathbf{w}^{\intercal}\mathbf{x}_n}\right) \qquad \longleftarrow \text{ iterative solution}$$

Compare to linear regression:

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} (\mathbf{w}^{\intercal}\mathbf{x}_n - y_n)^2 \qquad \longleftarrow \text{ closed-form solution}$$

20



Iterative method: gradient descent

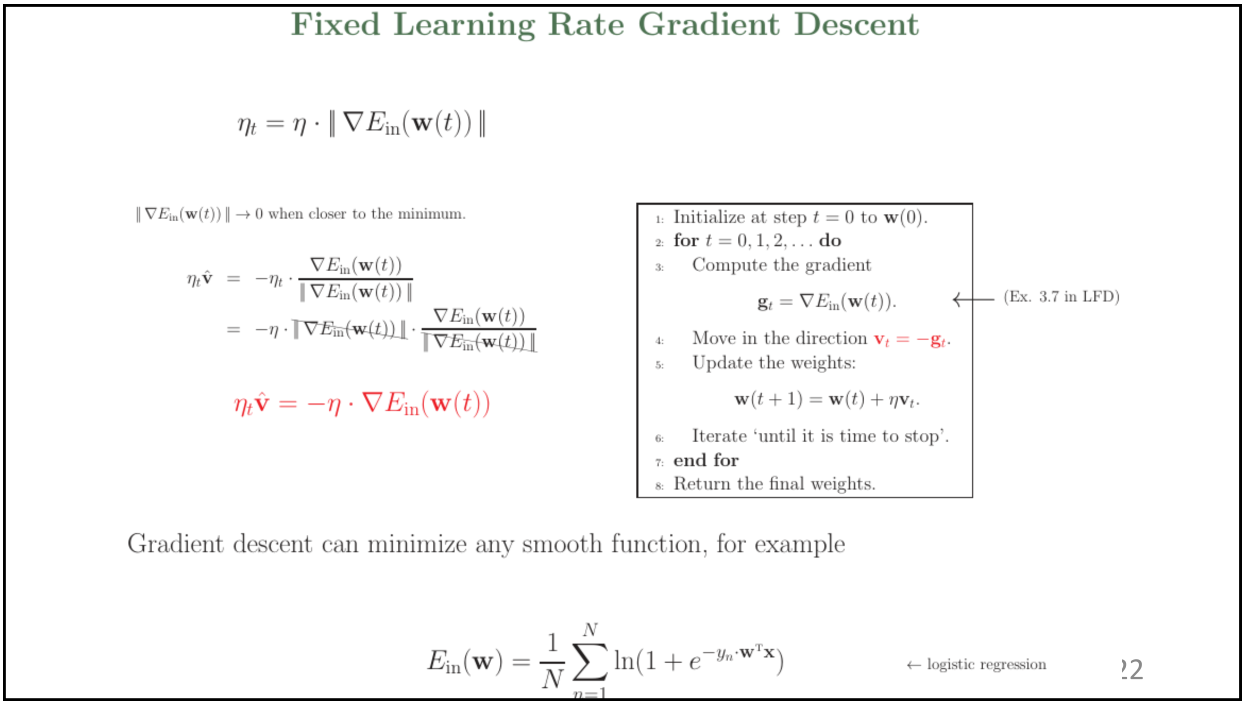
General method for nonlinear optimization

Start at $\mathbf{w}(0)$; take a step along steepest slope

Fixed step size: $\mathbf{w}(1) = \mathbf{w}(0) + \eta\,\hat{\mathbf{v}}$

What is the direction $\hat{\mathbf{v}}$?

$E_{\text{in}}(\mathbf{w})$
In-sample Error, E_{in}
Weights, $\mathbf{w}$



Fixed Learning Rate Gradient Descent

$$\eta_t = \eta \cdot \|\nabla E_{\text{in}}(\mathbf{w}(t))\|$$

$\|\nabla E_{\text{in}}(\mathbf{w}(t))\| \to 0$ when closer to the minimum.

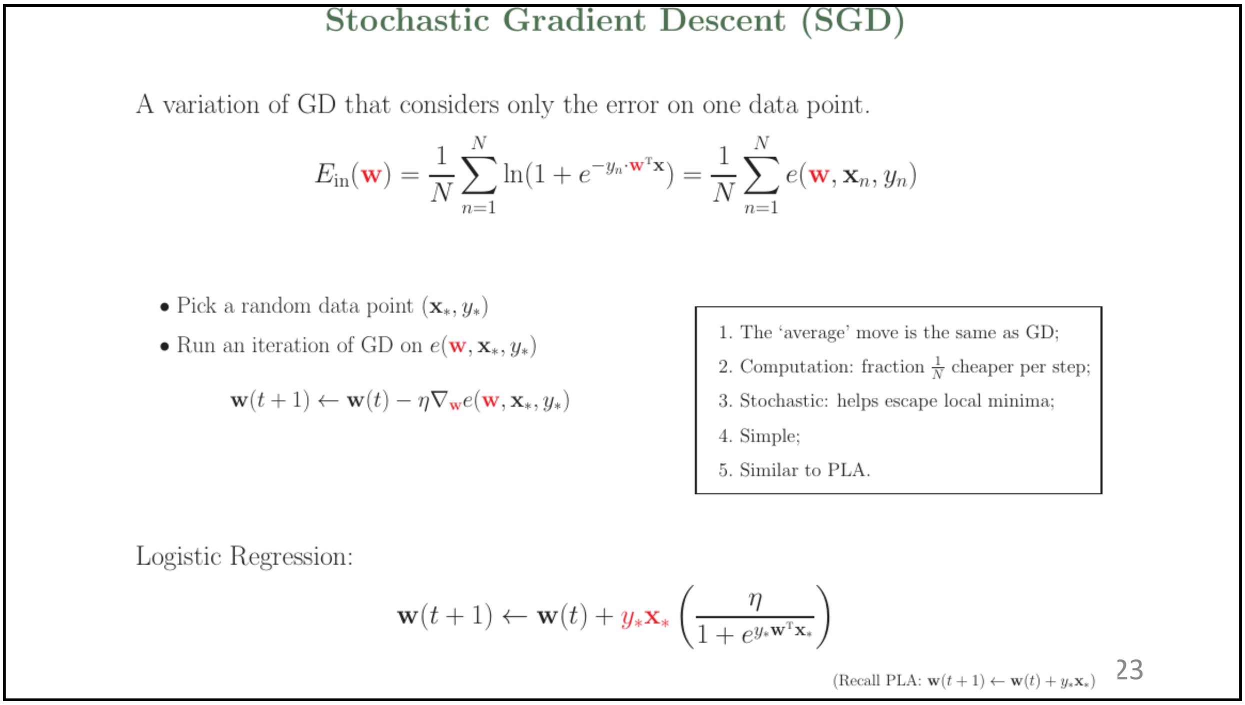
$$\begin{aligned}\eta_t\hat{\mathbf{v}} &= -\eta_t \cdot \frac{\nabla E_{\text{in}}(\mathbf{w}(t))}{\|\nabla E_{\text{in}}(\mathbf{w}(t))\|}\\ &= -\eta \cdot \cancel{\|\nabla E_{\text{in}}(\mathbf{w}(t))\|} \cdot \frac{\nabla E_{\text{in}}(\mathbf{w}(t))}{\cancel{\|\nabla E_{\text{in}}(\mathbf{w}(t))\|}}\end{aligned}$$

$$\eta_t\hat{\mathbf{v}} = -\eta \cdot \nabla E_{\text{in}}(\mathbf{w}(t))$$

```
Initialize at step $t = 0$ to $\mathbf{w}(0)$.
for $t = 0, 1, 2, \ldots$ do
    Compute the gradient
        $\mathbf{g}_t = \nabla E_{\text{in}}(\mathbf{w}(t))$.        ← (Ex. 3.7 in LFD)
    Move in the direction $\mathbf{v}_t = -\mathbf{g}_t$.
    Update the weights:
        $\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\mathbf{v}_t$.
    Iterate 'until it is time to stop'.
end for
Return the final weights.
```

Gradient descent can minimize any smooth function, for example

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}}) \qquad \leftarrow \text{logistic regression}$$



Stochastic Gradient Descent (SGD)

A variation of GD that considers only the error on one data point.

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n \cdot \mathbf{w}^{\mathrm{T}}\mathbf{x}}) = \frac{1}{N}\sum_{n=1}^{N} e(\mathbf{w}, \mathbf{x}_n, y_n)$$

- Pick a random data point $(\mathbf{x}_*, y_*)$
- Run an iteration of GD on $e(\mathbf{w}, \mathbf{x}_*, y_*)$

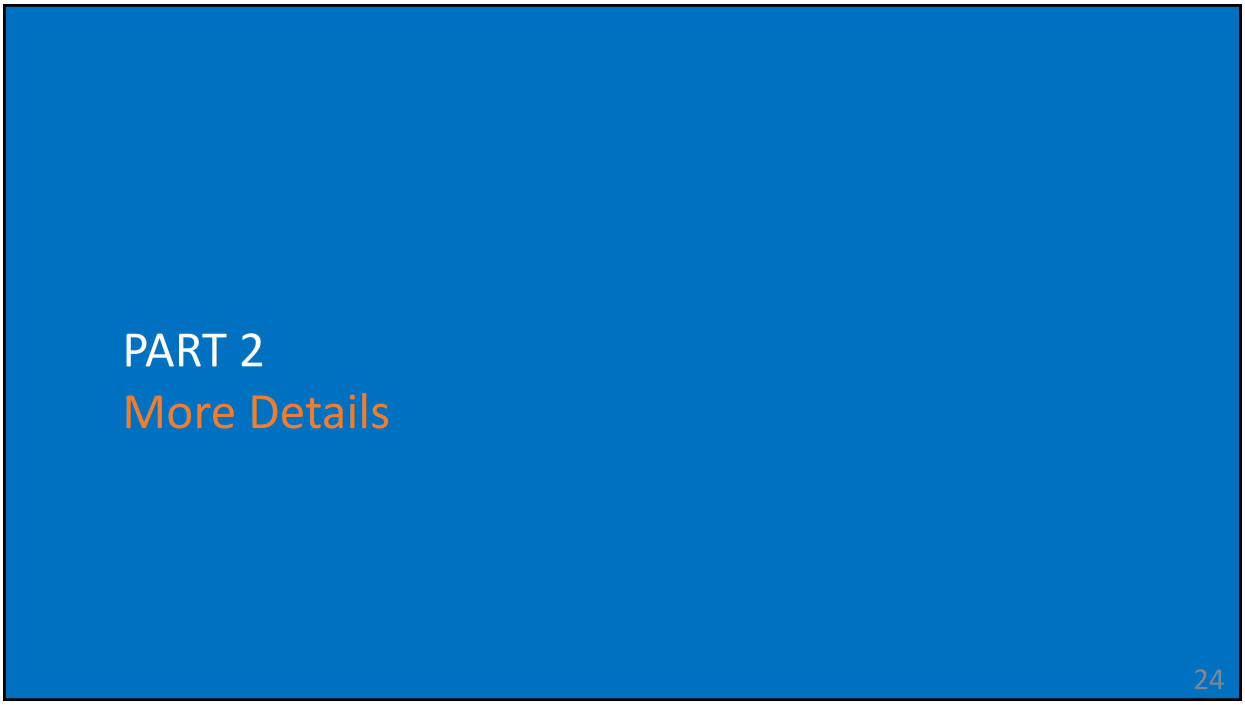
$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) - \eta \nabla_{\mathbf{w}} e(\mathbf{w}, \mathbf{x}_*, y_*)$$

1. The 'average' move is the same as GD;
2. Computation: fraction $\frac{1}{N}$ cheaper per step;
3. Stochastic: helps escape local minima;
4. Simple;
5. Similar to PLA.

Logistic Regression:

$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_*\mathbf{x}_* \left(\frac{\eta}{1 + e^{y_* \mathbf{w}^{\mathrm{T}}\mathbf{x}_*}} \right)$$

(Recall PLA: $\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_*\mathbf{x}_*$)



PART 2

More Details

Details

- See hand-written notes and discussion in class.

PART 3
Code Time!

Code Time

- See demonstration and discussion in class.
- See also links in the “Code Examples” for this Lecture.

Conclusion

Takeaways

Logistic regression vs. Linear regression

- **Logistic regression** - like linear regression - is a type of linear model that examines the relationship between predictor variables (independent variables) and an output variable (the response, target or dependent variable).
- Key difference is:
  - Logistic regression is used when the outcome is categorical, such as whether a loan is approved or not.
  - Linear regression is used when the output is a continuous value - for example, predicting someone's credit score.
- In **logistic regression**, the model predicts the probability that a specific outcome occurs.
  - For instance, given someone's financial profile, we might predict the probability that their loan is approved.
  - The output of the model is a value between 0 and 1. Based on a threshold—often 0.5—we classify the outcome as either "approved" or "not approved."
  - Instead of drawing a straight line through the data as we would in linear regression, logistic regression fits an S-shaped curve to map input values to a probability.

Summary

- Classification problems (where y is categorical) are everywhere
- Regression losses are not typically appropriate
- Logistic regression: model conditional probability $\mathrm{P}(y \mid x)$ as sigmoid (then apply usual MLE machinery)
- Softmax classification: generalized to $k > 2$ classes
- Regularization is still important

Recap: The ML Pipeline

1. Define the **task** (what type of data, what type of eval metrics?)
2. Collect and preprocess **data**
3. Choose **model** family/parameterization
4. Choose **training loss**
4. For each choice of hyperparameters:
   - **Optimize** model (minimize loss) on training data
   - **Evaluate** on validation data
5. Pick best hyperparameters according to validation performance
6. **Evaluate** final model on test data

References and Credits

Many of the teaching materials for this course have been adapted from various sources. We are very grateful and thank the following professors, researchers, and practitioners for sharing their teaching materials (in no particular order):

- Yaser S. Abu-Mostafa, Malik Magdon-Ismail and Hsuan-Tien Lin. https://amlbook.com/slides.html
- Ethem Alpaydin. https://www.cmpe.boun.edu.tr/~ethem/i2ml3e/
- Natasha Jaques. https://courses.cs.washington.edu/courses/cse446/25sp/
- Lyle Ungar. https://alliance.seas.upenn.edu/~cis520/dynamic/2022/wiki/index.php?n=Lectures.Lectures
- Aurelien Geron. https://github.com/ageron/handson-ml3
- Sebastian Raschka. https://github.com/rasbt/machine-learning-book
- Trevor Hastie. https://www.statlearning.com/resources-python
- Andrew Ng. https://www.youtube.com/playlist?list=PLoROMvodv4rMiGQp3WXShtMGgzqpfVfbU
- Richard Povineli. https://www.richard.povinelli.org/teaching
- … and many others.