Day-23, Oct 25, 2024

→ Continuing Deep learning Foundations

→ Overfitting problems & possible remedies

Deep learning Foundations

① Sigmoid functions motivations & usage

② Optimization in Logistic Regression

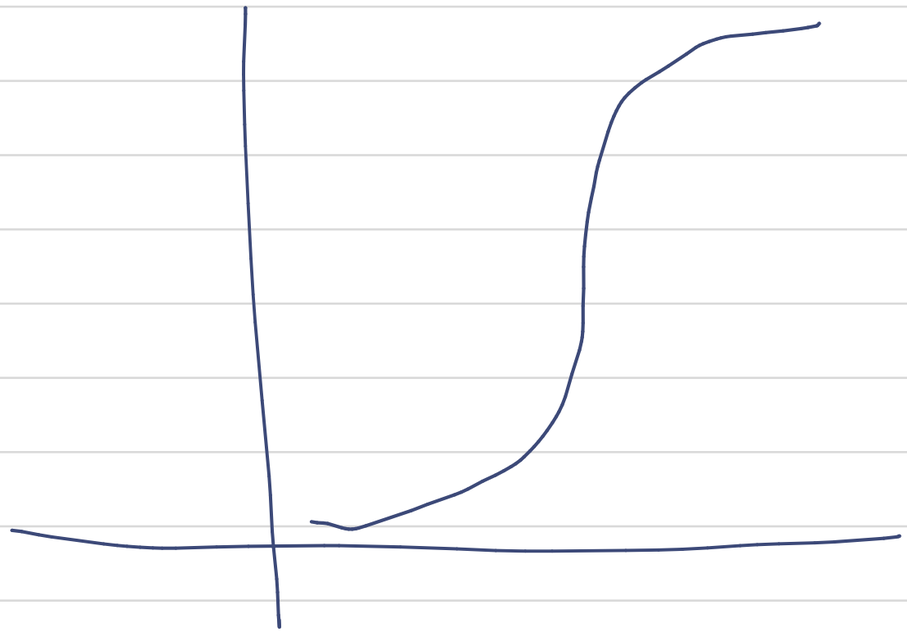
③ Methods to Compute Gradient Descend

④ Example of Chain Rule & Composite Derivative function.

Sigmoid Function

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

→ ranges [0,1]



Using

$z = \theta^T x$

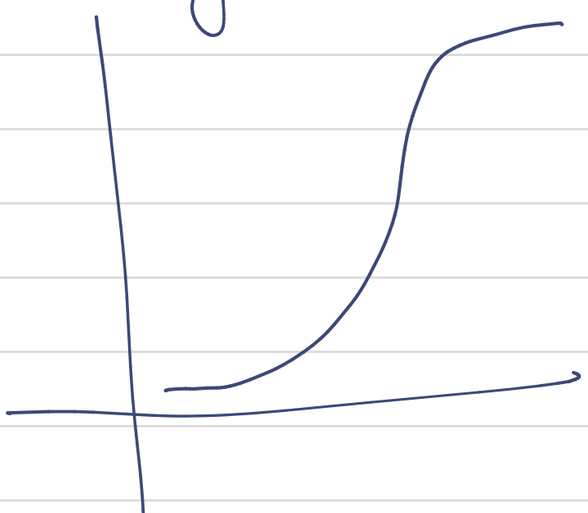
$$g(z) = \frac{1}{1+e^{-(\theta^T x)}}$$

→ Change θ randomly (θ^T) → Transposed θ

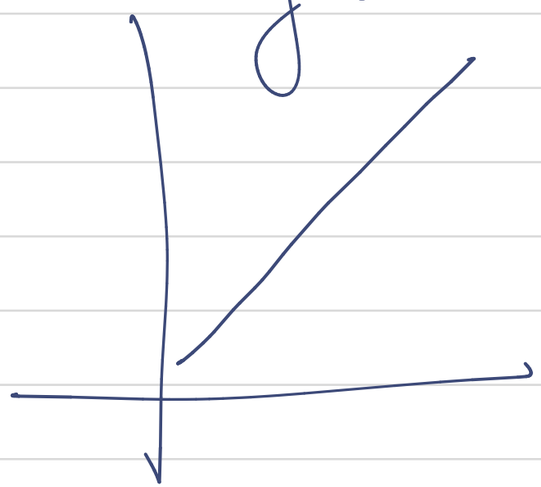
→ Sigmoid shows probabilities of classification

→ There is no nonlinear decision boundary using Sigmoid.

Using $\theta = 0.73$



Using $\theta = 0.10$



our goal must be finding the optimal parameter theta(θ) that maximize the likelihood of given (observed dataset).

So we have loglikelihood function and likelihood function

log-likelihood(f): Sum of individual log-likelihood terms for each input.

likelihood(f): Product of probabilities of each data point belonging to respective classes.

$(y=1)$ $\log \pi_0(x)$: Head → term contribute to likelihood of actual class '1'

$(y=0)$ $\log(1-\pi_0(x))$: Tail → term contribute to likelihood of actual class '0'.

$$So,\ h_\theta(x) \Rightarrow \frac{1}{1 + e^{-(\theta^T x)}}$$

Once we find the optimal θ it is time for decision boundary that separate two classes.

In Logistic Regression, optimization works by iteratively updating θ values in the direction of the gradient of the log-likelihood function to maximize it.

Logistic Regression

- Logistic Regression → Finding good θ's for this likelihood (i.e., the ones that maximize it)

$$\log p_\theta(y|x) = \sum_{i=1}^{n} \mathbb{I}(y_i = 1) \log \pi_\theta(x) + \mathbb{I}(y_i = 0) \log(1 - \pi_\theta(x))$$

- These are the directions like those in the previous figure that do a good job separating heads from tails

Sigmoid Function

- Consider the collection of functions (one for each θ)

$$\pi_\theta(x) = \frac{1}{1 + \exp(-\theta^T x)} \overset{\text{defined}}{=} \sigma(\theta^T x)$$

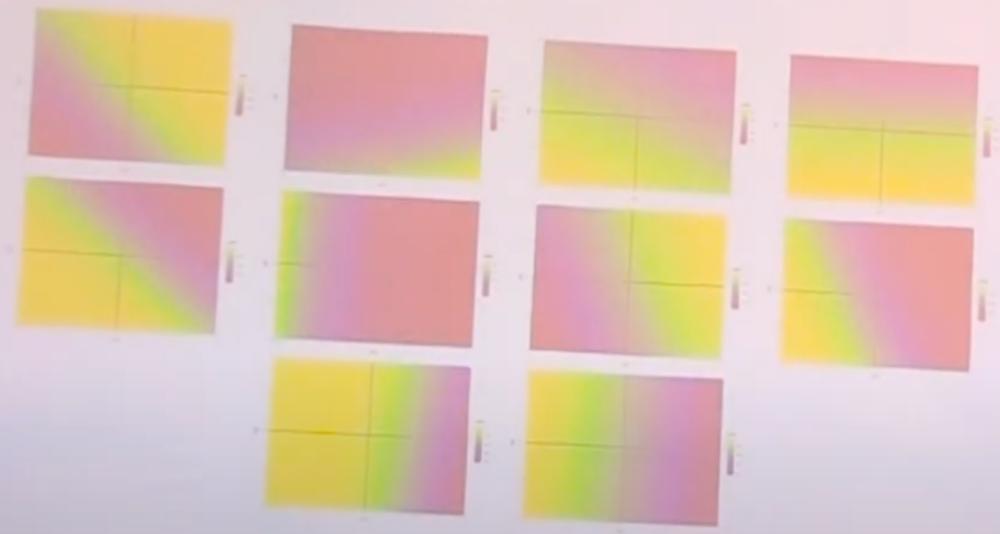


Figure: Example sigmoid functions π_θ for random draws of θ, when we assume x has an intercept term. Notice that there are no nonlinear boundaries...

$$\log p_\theta(y|x) \Rightarrow \sum_{i=1} \mathbb{I}(y_i = 1) \log \pi_\theta(x) + \mathbb{I}(y_i = 0) \log(1 - \pi_\theta(x))$$

$$\pi_\theta(x) \Rightarrow \frac{1}{1 + e^{-\theta^T x}} \Rightarrow \sigma(\theta^T x) \quad \#.$$

Courageously differentiate

① $\frac{\partial}{\partial \theta} \log p_\theta(y|x)$

②

⇒ Sum of log-likelihood terms for each point

③

⇒ then the derivative of log-likelihood is calculated using Chain Rule and the derivative of Sigmoid function

Optimization

- Need to find some θ's that maximize $\log p_\theta(y|s)$
- Use gradient descent on $-\log p_\theta(y|x)$
- How to get the gradients?
- Approach 1: Courageously differentiate.

$$\frac{\partial}{\partial \theta} \log p_\theta(y|x)$$

$$= \frac{\partial}{\partial \theta} \left[\sum_{i=1}^{n} y_i \log \sigma(\theta^T x_i) + (1 - y_i) \log \left(1 - \sigma(\theta^T x_i)\right) \right]$$

$$= \frac{\partial}{\partial \theta} \left[\sum_{i=1}^{n} y_i \log \left(\frac{1}{1 + \exp(-\theta^T x_i)} \right) + (1 - y_i) \log \left(\frac{\exp(-\theta^T x_i)}{1 + \exp(-\theta^T x_i)} \right) \right]$$

(2) Chain Rule Approach (Compute Gradient Descend)

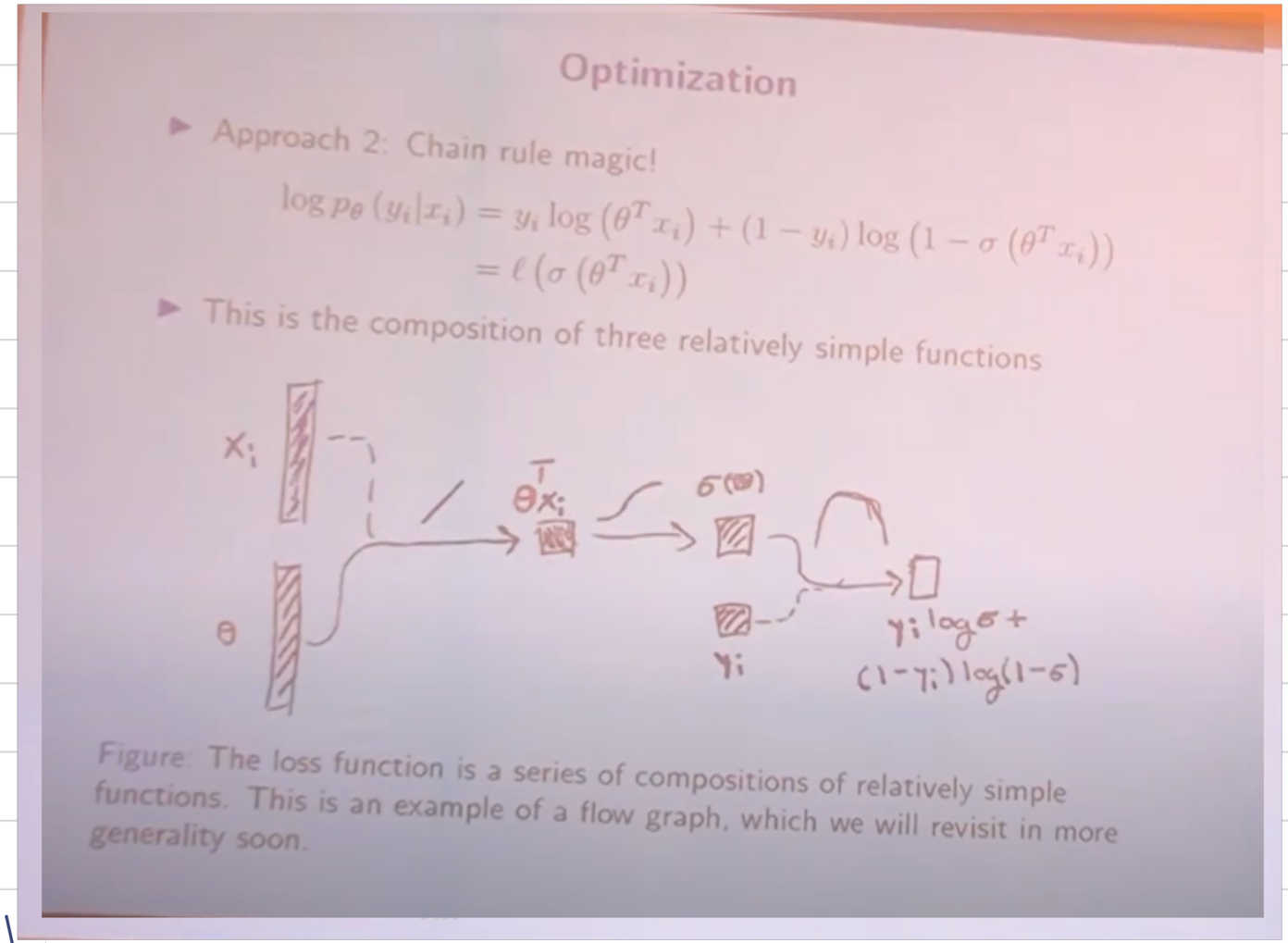
See how Chain Rule does following things

(1) Function Decomposition

(2) Linear Combination

(3) Function differentiable

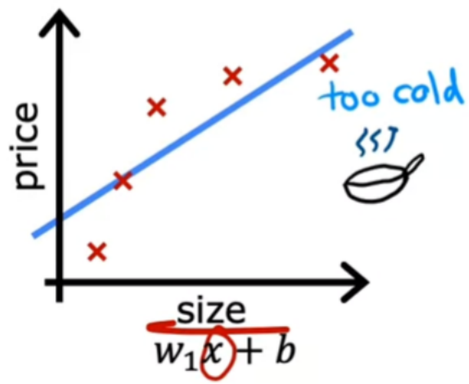
(4) Derivative & chain Rule breaks down Complex problems functions into Simpler ones.



Overfitting:-

① Model learns noise bw & have high variance.

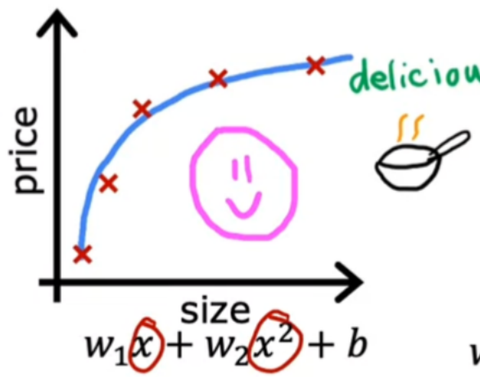
Regression example



underfit

- Does not fit the training set well

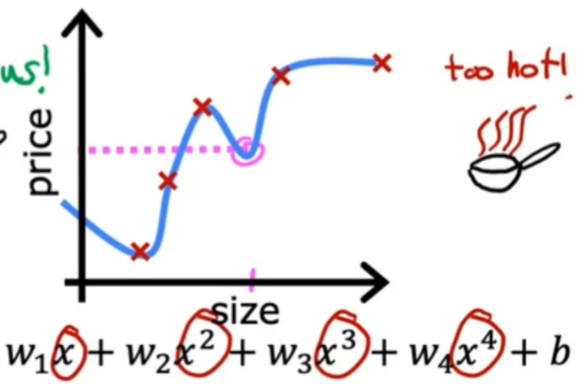
high bias



just right

- Fits training set pretty well

generalization



overfit

- Fits the training set extremely well

high variance

→ $w_1x + b$
→ High bias
Underfit

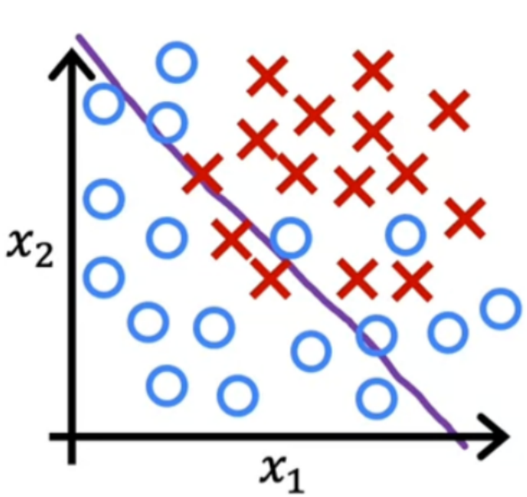
$w_1x + w_2x^2 + b$
generalized.

→ $w_1x + w_2x^2 + w_3x^3 + w_4x^4 + b$
→ overfit
→ High Variance.

See how polynomial features play role in overfit.

coursera.org - To exit full screen, press esc

Classification

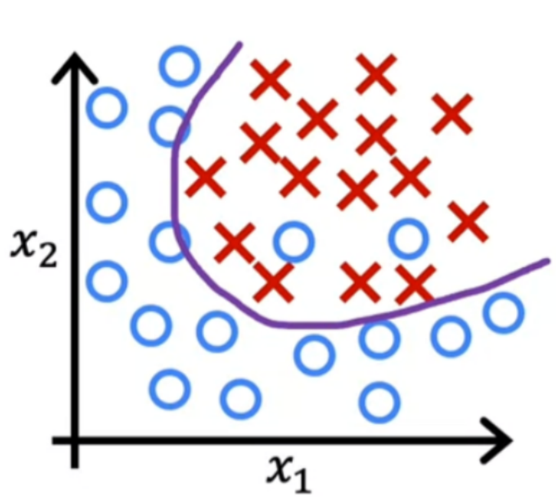


$z = w_1 x_1 + w_2 x_2 + b$

$f_{\vec{w},b}(\vec{x}) = g(z)$

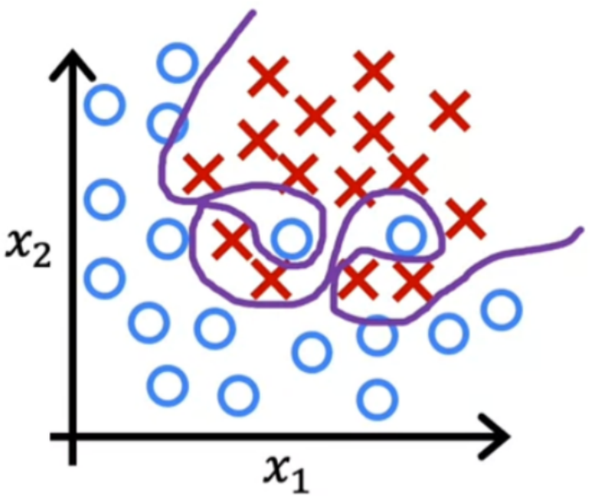
g is the sigmoid function

underfit high bias



$z = w_1 x_1 + w_2 x_2 + w_3 x_1^2 + w_4 x_2^2 + w_5 x_1 x_2 + b$

just right

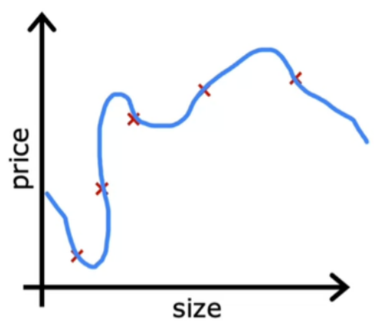


$z = w_1 x_1 + w_2 x_2 + w_3 x_1^2 x_2 + w_4 x_1^2 x_2^2 + w_5 x_1^2 x_2^3 + w_6 x_1^3 x_2 + \cdots + b$

overfit

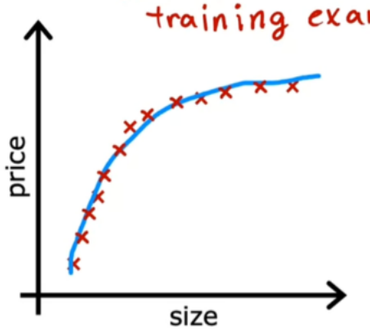
It is likely that irrelevant features will aid in overfitting while less data or not enough features will result in underfitting.

Collect more training examples



overfit

collect more training examples



Select features to include/exclude

size	bedrooms	floors	age	avg income	...	distance to coffee shop	price
x_1	x_2	x_3	x_4	x_5		x_{100}	y

all features

\+

insufficient data

↓

overfit

selected features

size
bedrooms
age
just right

feature selection

course 2

disadvantage

↓

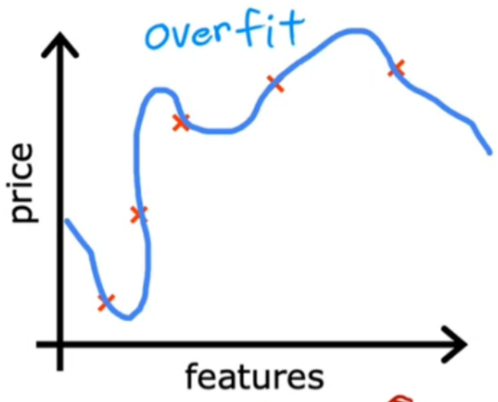
useful features could be lost

How to prevent Overfitting? (Possible Solutions)

→ Collect More training Examples
→ Remove Irrelevant Features
→ Select best features
→ Regularization
→ Cross-Validations
→ training & test Split.

Regularization

Reduce the size of parameters w_j

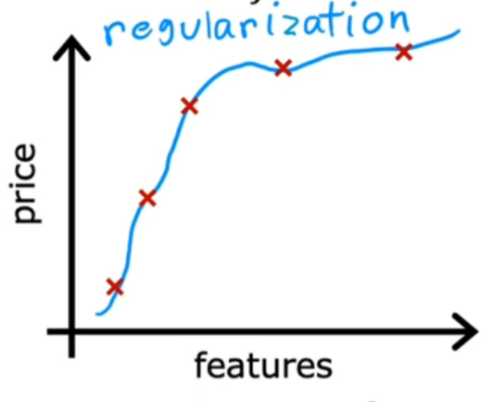


$f(x) = 28x - 385x^2 + 39x^3 - 174x^4 + 100$

0

eliminate feature

large values for w_j



$f(x) = 13x - 0.23x^2 + 0.000014x^3 - 0.0001x^4 + 10$

small values for w_j

Feature Selection

1) Wrapper
2) Embedded
3) RLE

Regularization:

L1, L2, Elastic Net

Addressing overfitting

Options

1. Collect more data

2. Select features
   - Feature selection in course 2

3. Reduce size of parameters
   - "Regularization" next videos!