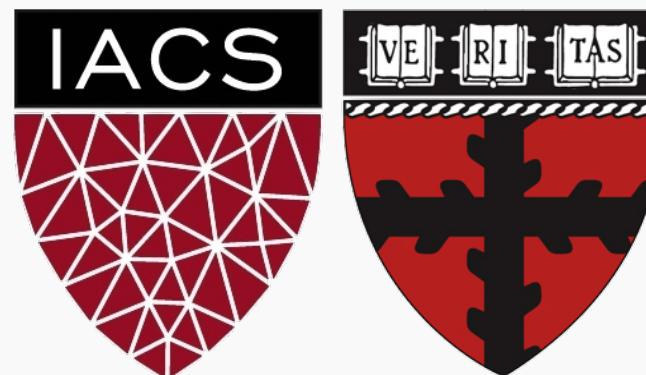


# Lecture 1: Perceptron and Back Propagation

Pavlos Protopapas

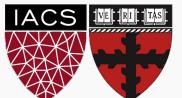
Institute for Applied Computational Science  
Harvard



# Outline

---

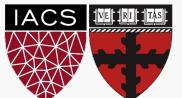
1. Description of the course
2. Introduction to Artificial Neural Networks
3. Review of Classification and Logistic Regression
4. Single Neuron Network ('Perceptron')
5. Introduction to Optimization
  - Gradient Descent
  - Stochastic Gradient Descent
6. Back Propagation
7. Multi-Layer Perceptron



# Outline

---

1. Description of the course
2. Introduction to Artificial Neural Networks
3. Review of Classification and Logistic Regression
4. Single Neuron Network ('Perceptron')
5. Introduction to Optimization
  - Gradient Descent
  - Stochastic Gradient Descent
6. Back Propagation
7. Multi-Layer Perceptron



# Schedule

---

Six Lectures @Saturdays 8:00am EST by Pavlos Protopapas (**check the schedule**):

1. Introduction
2. Design choices for NN, number of layers, number of nodes, activation function
3. Solver and Regularization
4. Convolutional Neural Networks, basic concepts and architectures
5. Convolutional Neural Networks part2 and state of the art networks
6. Auto-encoders

Six Labs @Sundays 8:00am EST (David Sondak)

Six Office hours @Mondays 8:00am EST (David Sondak)



# Assignments

---

**Pre-lecture reading assignment:** 10%

**Participation:** 25%

- Piazza/Ed: 5%
- Lectures: 10%
- Labs: 10%

**Homework :** 35%

- HW1: Release week 4, due week 5
- HW2: Release week 5 due week 6

**Final Project** (release 6, due week 7?): 30%



# Assignments

---

Two assignments:

1. To be released Nov 2<sup>nd</sup> midnight, due on Nov 9<sup>th</sup> midnight
2. To be released Nov 9<sup>th</sup> midnight, due on Nov 16<sup>th</sup> midnight

Final Project:

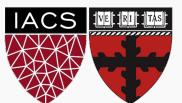
The final project will be challenging! We will give you a problem and you should apply some innovative method besides the standard methods we have learned in class. You will have a chance to present your approach to us and the rest of the class.



# Who

---

**Pavlos Protopapas**, Scientific Director of the Institute for Applied Computational Science (IACS). Teaches CS109a, CS109b and the Capstone course for the Data Science masters program. He is a leader in astrostatistics and he is excited about the new telescopes coming online in the next few years. He loves cooking and eating.



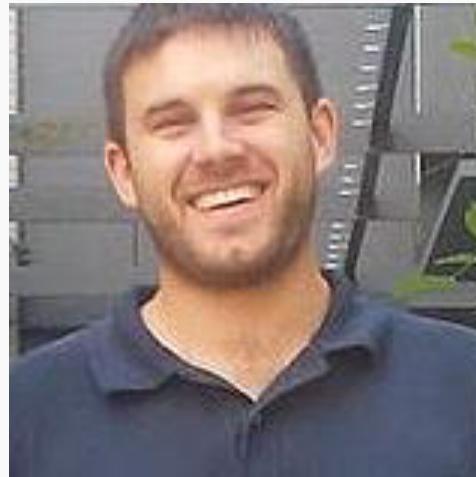
PAVLOS PROTOPAPAS



# Who

---

**David Sondak**, computational mathematician and lecturer at the Institute for Applied Computational Science. He teaches CS207 and extreme computing. He loves maths and dogs!

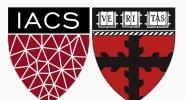


# Policies

---

We encourage you to talk and discuss the assignments with your fellow students (and on Piazza), but you are not allowed to look at any other students assignment or code outside of your group (max 3).

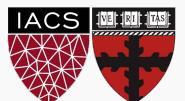
**YOU CAN SUBMIT AS A GROUP.**



# Policies

---

Ethical behavior is an important trait of a Data Scientist, from ethically handling data to attribution of code and work of others. Thus, in this course we give a strong emphasis to Academic Honesty. As a student your best guidelines are to be reasonable and fair. We encourage teamwork for problem sets, but you should not split the homework and you should work on all the problems together. We have included some ideas below of acceptable and not acceptable behaviors. Engaging in not acceptable behavior regarding academic honesty will be handled harshly. Please be responsible and when in doubt ask either Pavlos or David.



# Policies

---

ACCEPTABLE :

- Discussing materials and engaging in OH.
- Helping debug.
- Using a few lines of code found online or other forum as long as you cite the origin and attribute authorship of code.
- Searching online to expand your knowledge and for debugging, but not for outright solutions to HWs.
- Using a tutor, provided the tutor does not do your work for you.

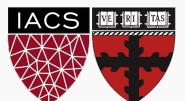


# Policies

---

NOT ACCEPTABLE :

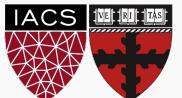
- Accessing a solution to some problem prior to submitting your own.
- Failing to cite the origins of code or techniques that you discover outside of the course's own lessons and integrate into your own work.
- Paying or offering to pay an individual for work that you may submit as your own.
- Providing or making available solutions to problem sets to individuals who might take this course in the future.
- Searching for or soliciting outright solutions to problem sets online or elsewhere.
- **Splitting a problem set's workload with another individual and combining your work.**



# Outline

---

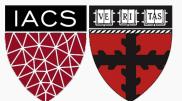
1. Description of the course
2. **Introduction to Artificial Neural Networks**
3. Review of Classification and Logistic Regression
4. Single Neuron Network ('Perceptron')
5. Introduction to Optimization
  - Gradient Descent
  - Stochastic Gradient Descent
6. Back Propagation
7. Multi-Layer Perceptron



# Watch this!

---

<http://video.arstechnica.com/watch/sunspring-sci-fi-short-film>



PAVLOS PROTOPAPAS



# Artificial Neural Networks



# Today's news

## An AI just beat top lawyers at their own game



IMAGE: BOB AL-GREEN/MASHABLE



BY  
**MONICA  
CHIN**

FEB  
26  
2018

The nation's top lawyers recently battled artificial intelligence in a competition to interpret contracts — and they lost.

A new study, conducted by legal AI platform LawGeex in consultation with scholars from Stanford University, Duke University School of Law, and University of Southern California, pitted twenty experienced lawyers against an AI trained to evaluate legal contracts.

Competitors were given four hours to review five non-disclosure agreements (NDAs) and identify 30 legal issues, including arbitration, confidentiality of relationship, and indemnification. They were scored by how accurately they identified each issue.

**SEE ALSO:** [Google's new AI can predict heart disease by simply scanning your eyes](#)



PAVLOS PROTOPAPAS



16

# Today's news

## Google's new AI can predict heart disease by simply scanning your eyes

[Share on F](#) [Share on T](#) [+](#)



IMAGE: BEN BRAIN/DIGITAL CAMERA MAGAZINE  
VIA GETTY IMAGES

The secret to identifying certain health conditions may be hidden in our eyes.

BY

MONICA  
CHIN

FEB  
2018

Researchers from Google and its health-tech subsidiary Verily announced on Monday that they have successfully created algorithms to predict whether someone has high blood pressure or is at risk of a heart attack or stroke simply by scanning a person's eyes, the *Washington Post* reports.

SEE ALSO: [This fork helps you stay healthy](#)

Google's researchers trained the algorithm with images of scanned retinas from more than 280,000 patients. By reviewing this massive database, Google's algorithm trained itself to recognize the patterns that designated people as at-risk.

This algorithm's success is a sign of exciting developments in healthcare on the horizon. As Google fine-tunes the technology, it could one day



PAVLOS PROTOPAPAS



17

# AlphaGo (2015)

First program to beat a professional Go player

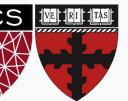


# AlphaZero (2017)

DeepMind

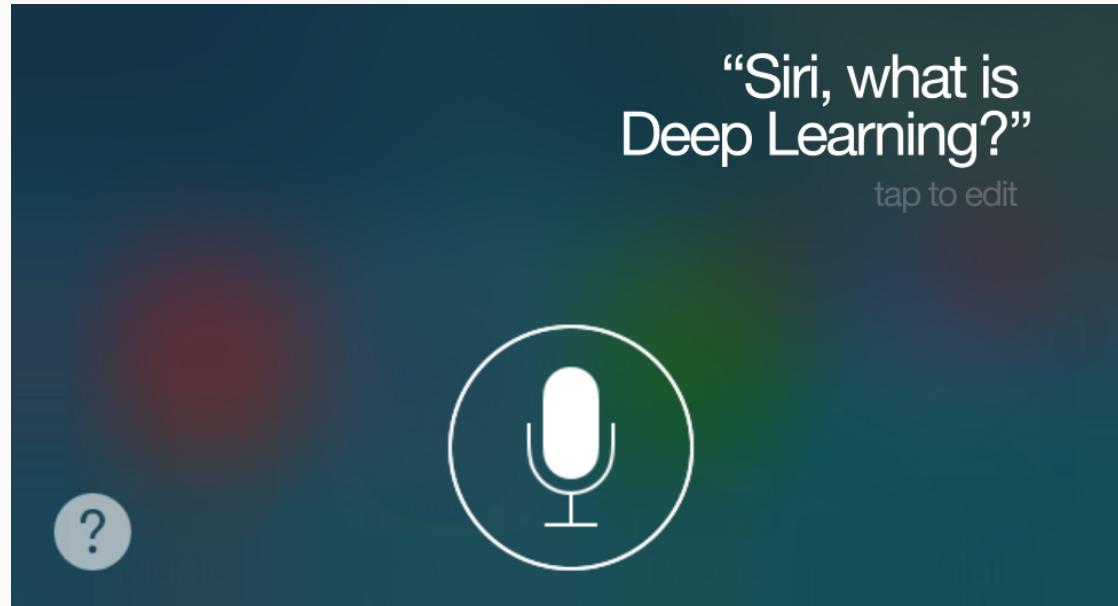
## AlphaZero AI beats champion chess program after teaching itself in four hours

Google's artificial intelligence sibling DeepMind repurposes Go-playing AI to conquer chess and shogi without aid of human knowledge



# iOS Speech Synthesis (2016-)

Trained from 20 hours of high quality speech



[machinelearning.apple.com](https://machinelearning.apple.com)

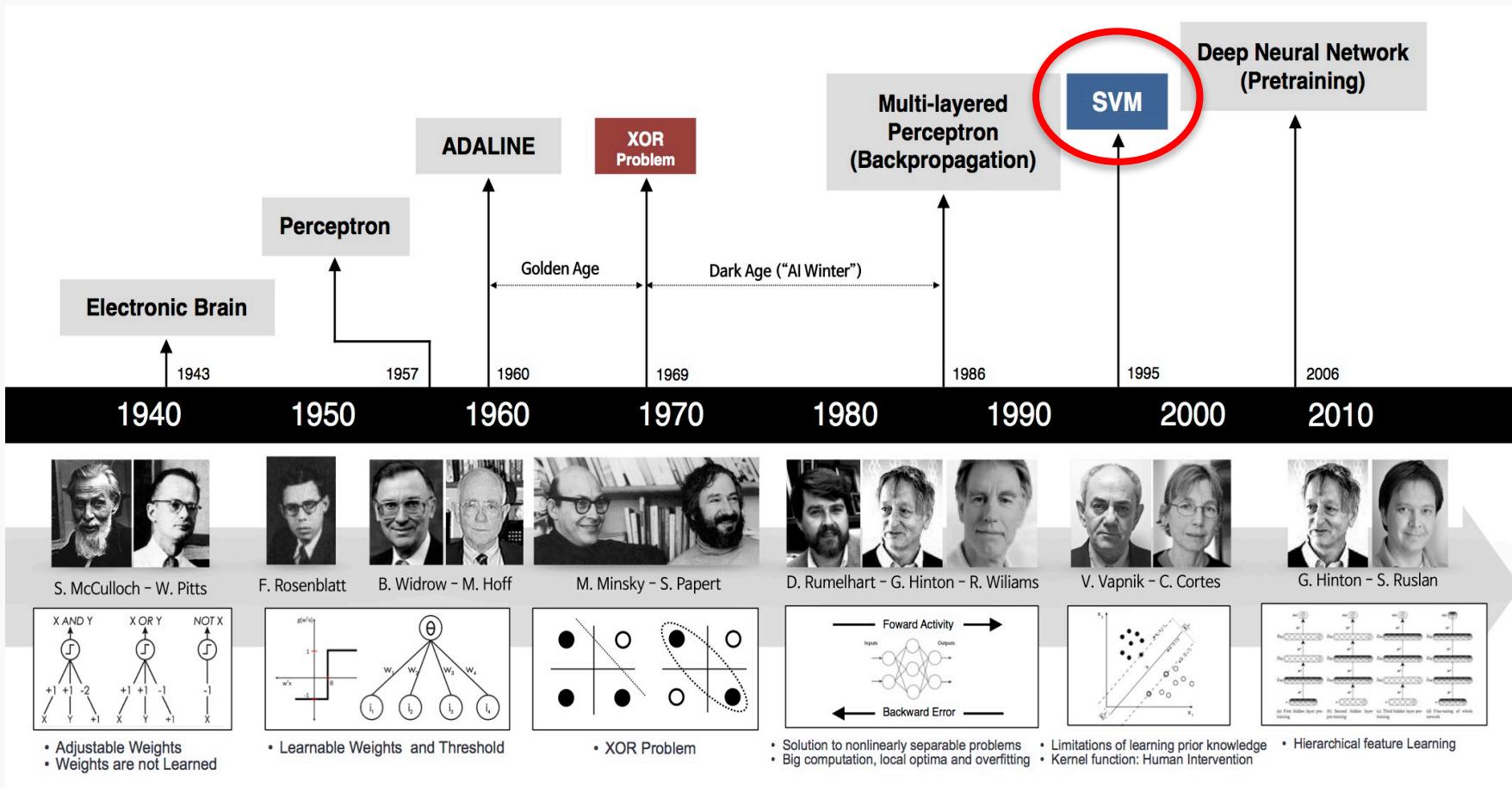


PAVLOS PROTOPAPAS



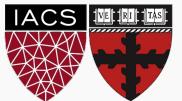
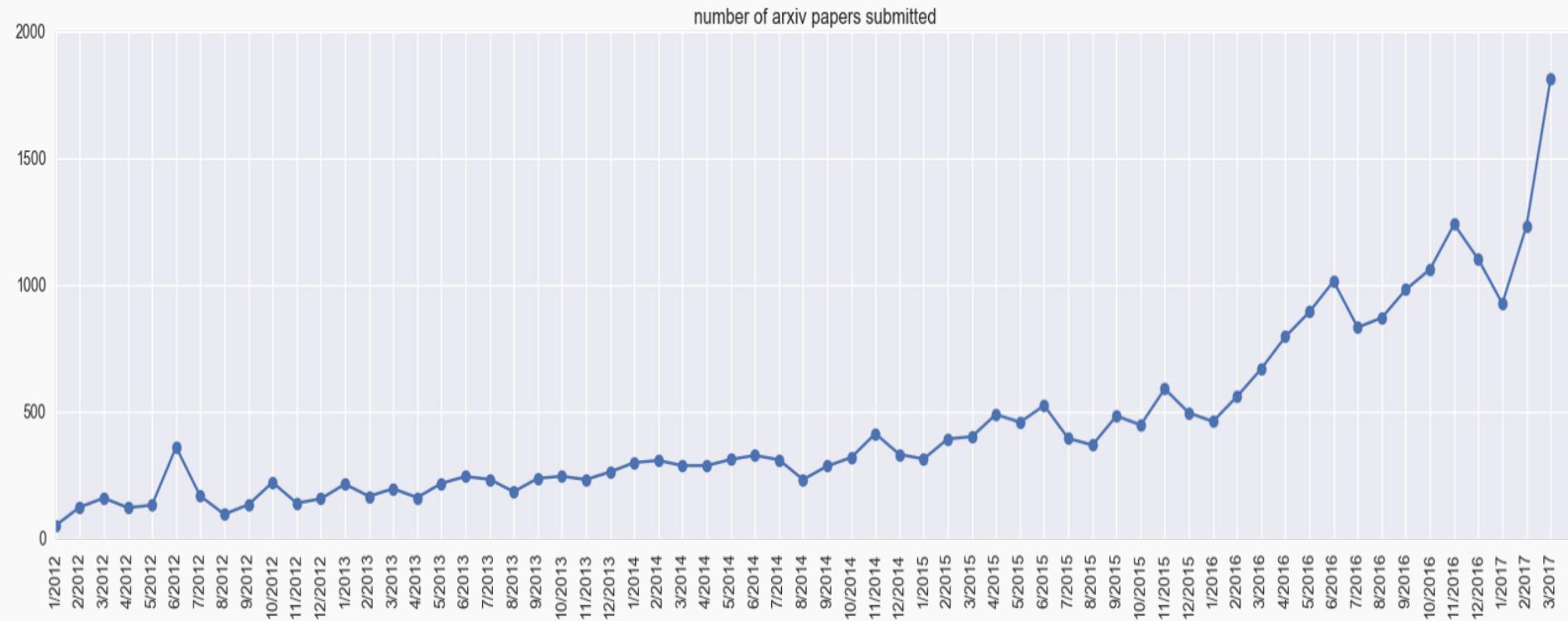
20

# Historical Trends



# Historical Trends

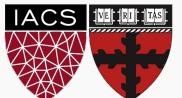
## ArXiv papers on deep learning: 2012-2017



# Outline

---

1. Description of the course
2. Introduction to Artificial Neural Networks
- 3. Review of Classification and Logistic Regression**
4. Single Neuron Network ('Perceptron')
5. Introduction to Optimization
  - Gradient Descent
  - Stochastic Gradient Descent
6. Back Propagation
7. Multi-Layer Perceptron



# Classification and Logistic Regression

# Regression and Classification

---

Methods that are centered around modeling and prediction of a **quantitative** response variable (ex, number of taxi pickups, number of bike rentals, etc) are called **regressions** (and Ridge, LASSO, etc).

When the response variable is **categorical**, then the problem is no longer called a regression problem but is instead labeled as a **classification problem**.

The goal is to attempt to classify each observation into a category (aka, class or cluster) defined by Y, based on a set of predictor variables X.



# Typical Classification Examples

The motivating examples for this lecture(s), homeworks and labs are based on classification. Classification problems are common in these domains:

- Trying to determine where to set the cut-off for some diagnostic test (pregnancy tests, prostate or breast cancer screening tests, etc...)
- Trying to determine if cancer has gone into remission based on treatment and various other indicators
- Trying to classify patients into types or classes of disease based on various genomic markers



# Data: Response vs. Predictor Variables

The diagram illustrates the structure of a dataset. A horizontal line represents the data, divided into two main sections: 'X predictors' and 'Y outcome'. The 'X predictors' section contains three columns labeled 'TV', 'radio', and 'newspaper'. The 'Y outcome' section contains one column labeled 'sales'. A bracket on the left indicates 'n observations' across all columns. A bracket at the bottom indicates 'p predictors' for the first three columns.

	TV	radio	newspaper	sales
1	230.1	37.8	69.2	22.1
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9

n observations

p predictors

X  
predictors  
features  
covariates

Y  
outcome  
response variable  
dependent variable

# Response vs. Predictor Variables

$$X = X_1, \dots, X_p$$

$$X_j = x_{1j}, \dots, x_{ij}, \dots, x_{nj}$$

**predictors**

features

covariates

$$Y = y_1, \dots, y_n$$

outcome

**response** variable

dependent variable

TV	radio	newspaper	sales
230.1	37.8	69.2	22.1
44.5	39.3	45.1	10.4
17.2	45.9	69.3	9.3
151.5	41.3	58.5	18.5
180.8	10.8	58.4	12.9

**n observations**

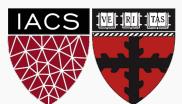
**p predictors**



# Heart Data

response variable Y  
is Yes/No

Age	Sex	ChestPain	RestBP	Chol	Fbs	RestECG	MaxHR	ExAng	Oldpeak	Slope	Ca	Thal	AHD
63	1	typical	145	233	1	2	150	0	2.3	3	0.0	fixed	No
67	1	asymptomatic	160	286	0	2	108	1	1.5	2	3.0	normal	Yes
67	1	asymptomatic	120	229	0	2	129	1	2.6	2	2.0	reversible	Yes
37	1	nonanginal	130	250	0	0	187	0	3.5	3	0.0	normal	No
41	0	nontypical	130	204	0	2	172	0	1.4	1	0.0	normal	No



# Heart Data

---

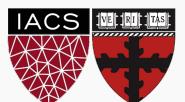
These data contain a binary outcome HD for 303 patients who presented with chest pain. An outcome value of:

- **Yes** indicates the presence of heart disease based on an angiographic test,
- **No** means no heart disease.

There are 13 predictors including:

- Age
- Sex
- Chol (a cholesterol measurement),
- MaxHR
- RestBP

and other heart and lung function measurements.



# Logistic Regression

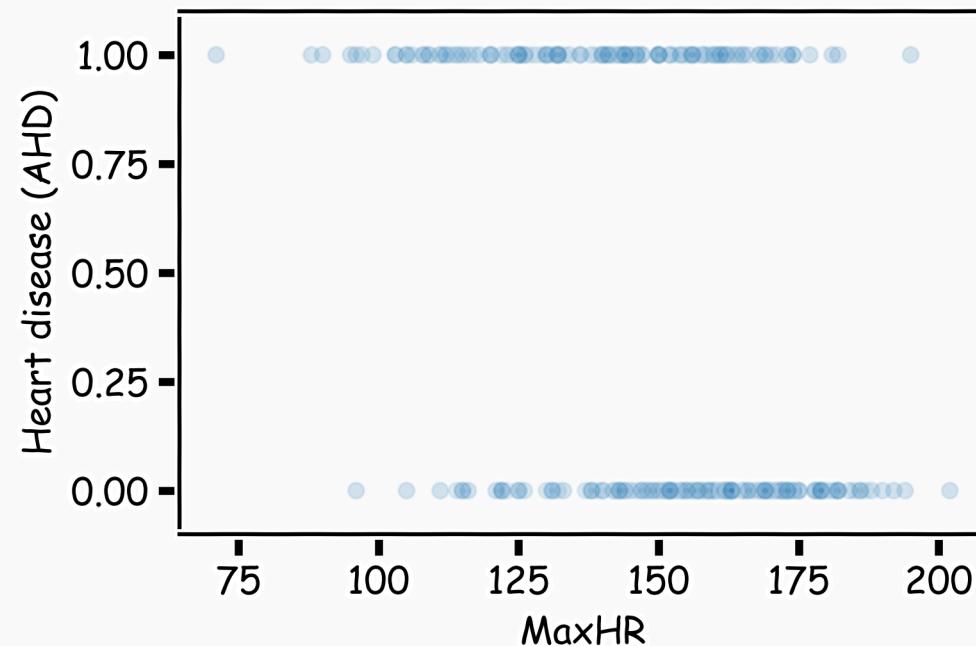
Logistic Regression addresses the problem of estimating a probability,  $P(y = 1)$ , given an input  $X$ . The logistic regression model uses a function, called the **logistic** function, to model  $P(y = 1)$ :

$$P(Y = 1) = \frac{e^{\beta_0 + \beta_1 X}}{1 + e^{\beta_0 + \beta_1 X}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



# Heart Data: logistic estimation

We'd like to predict whether or not a person has a heart disease. And we'd like to make this prediction, for now, just based on the MaxHR.



# Logistic Regression

As a result the model will predict  $P(y = 1)$  with an *S*-shaped curve, which is the general shape of the logistic function.

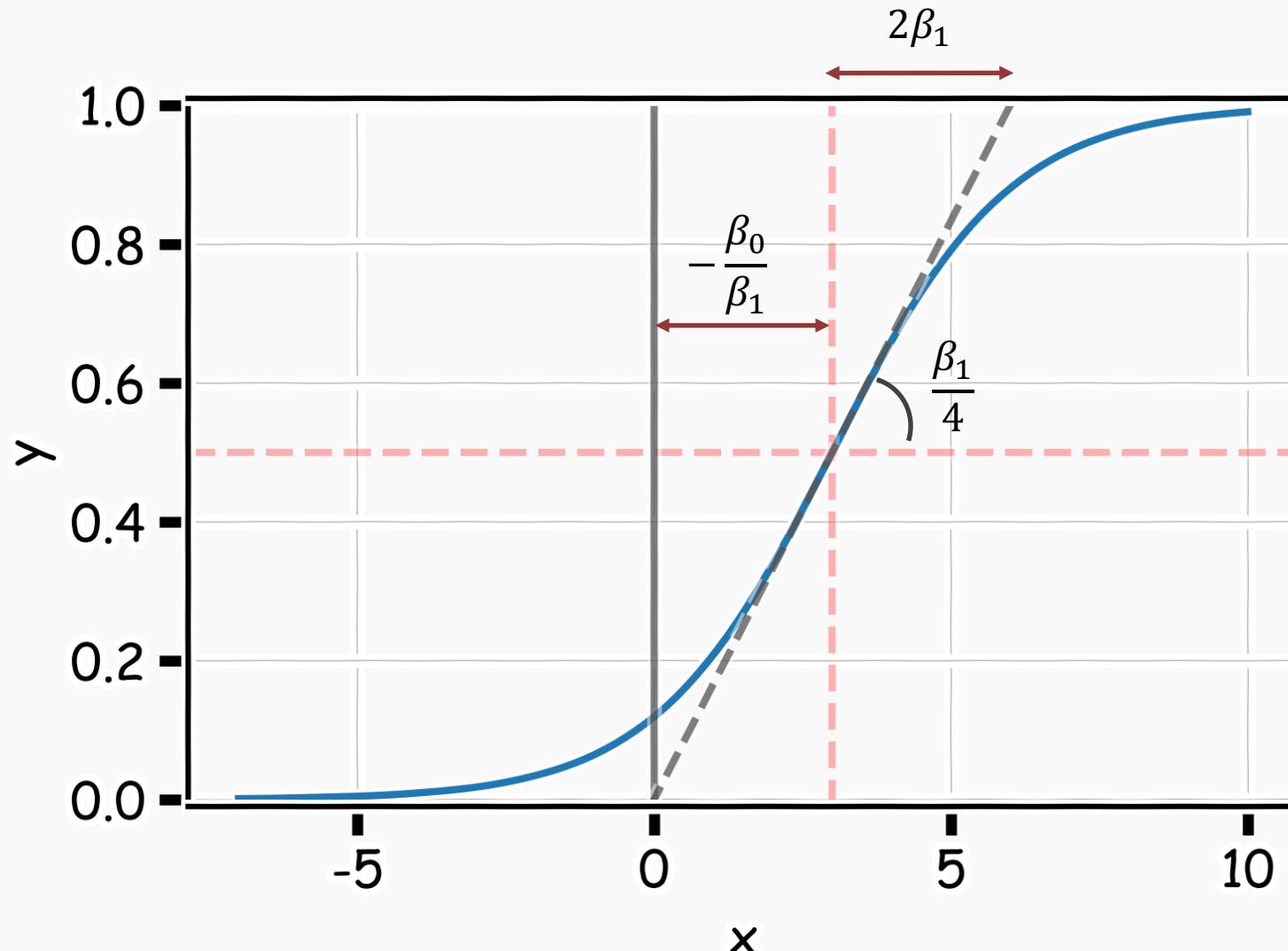
$\beta_0$  shifts the curve right or left by  $c = -\frac{\beta_0}{\beta_1}$ .

$\beta_1$  controls how steep the *S*-shaped curve is distance from  $\frac{1}{2}$  to  $\sim 1$  or  $\frac{1}{2}$  to  $\sim 0$  to  $\frac{1}{2}$  is  $\frac{2}{\beta_1}$

Note: if  $\beta_1$  is positive, then the predicted  $P(y = 1)$  goes from zero for small values of  $X$  to one for large values of  $X$  and if  $\beta_1$  is negative, then has the  $P(y = 1)$  opposite association.

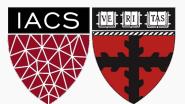
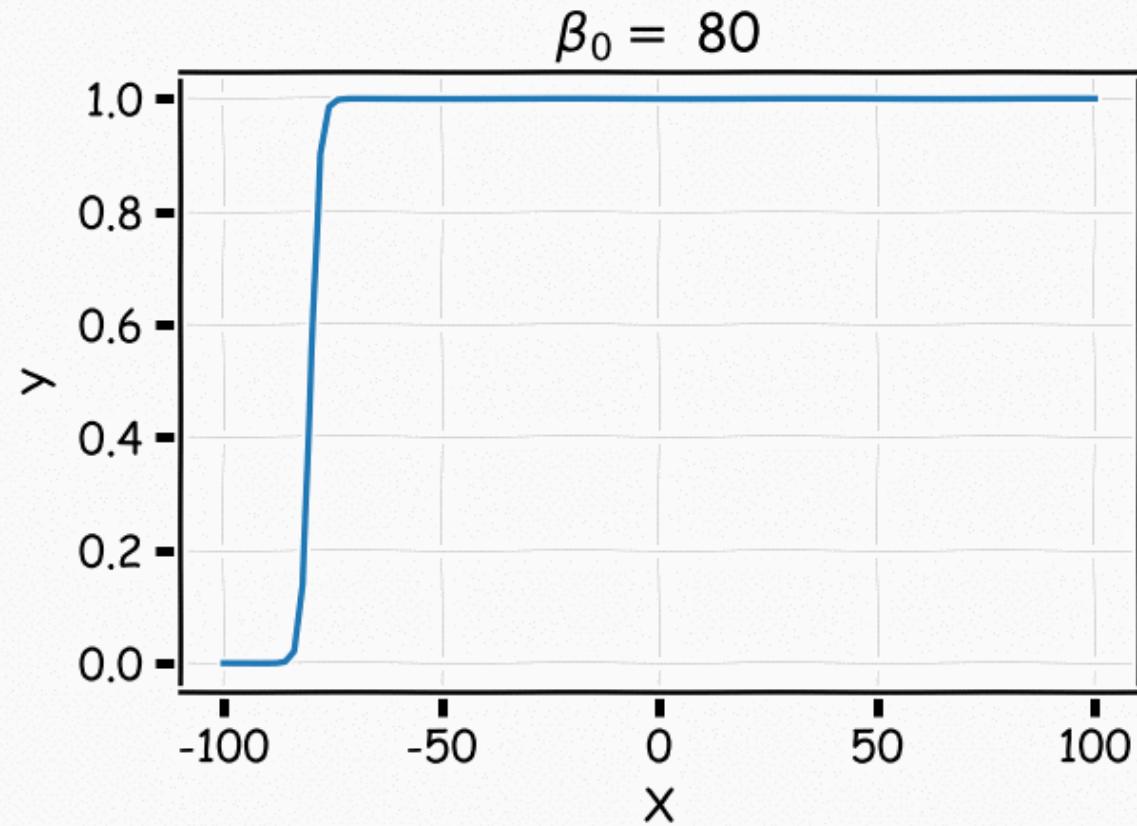


# Logistic Regression



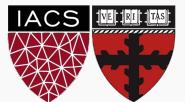
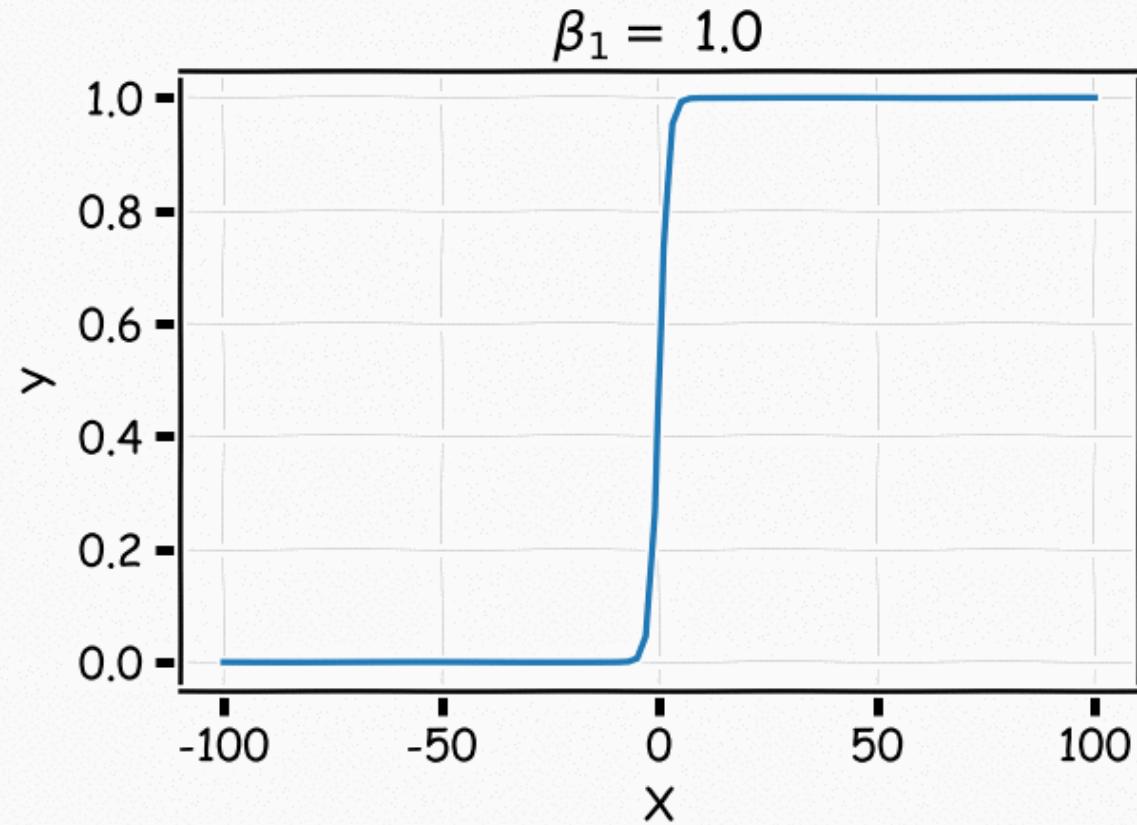
# Logistic Regression

$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



# Logistic Regression

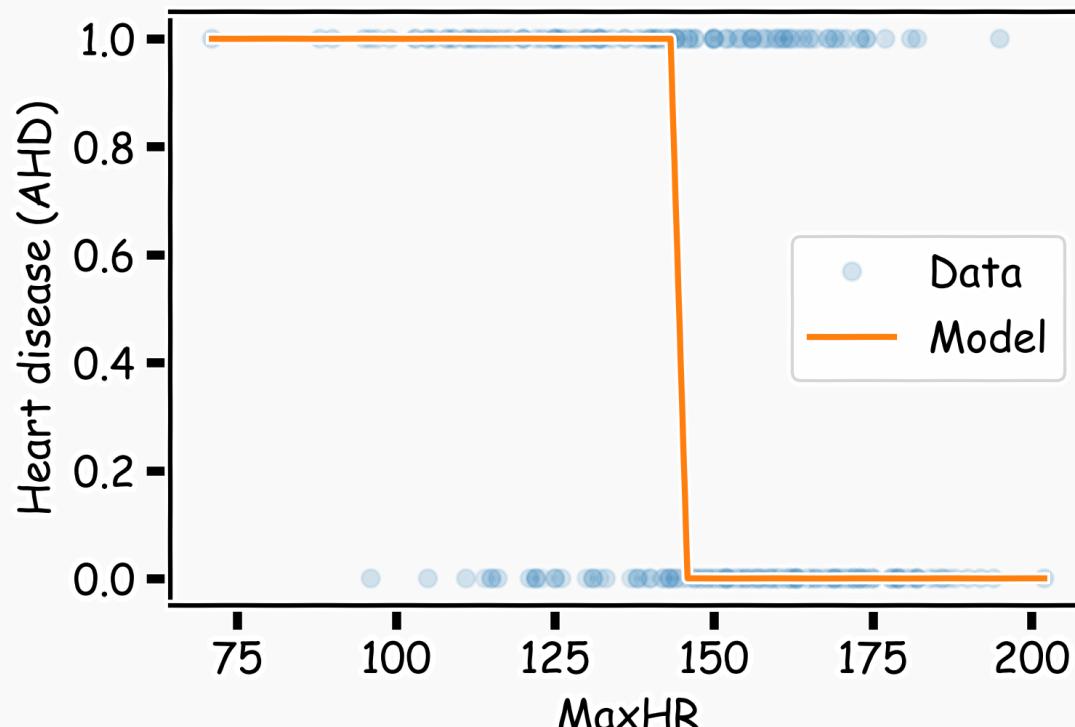
$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



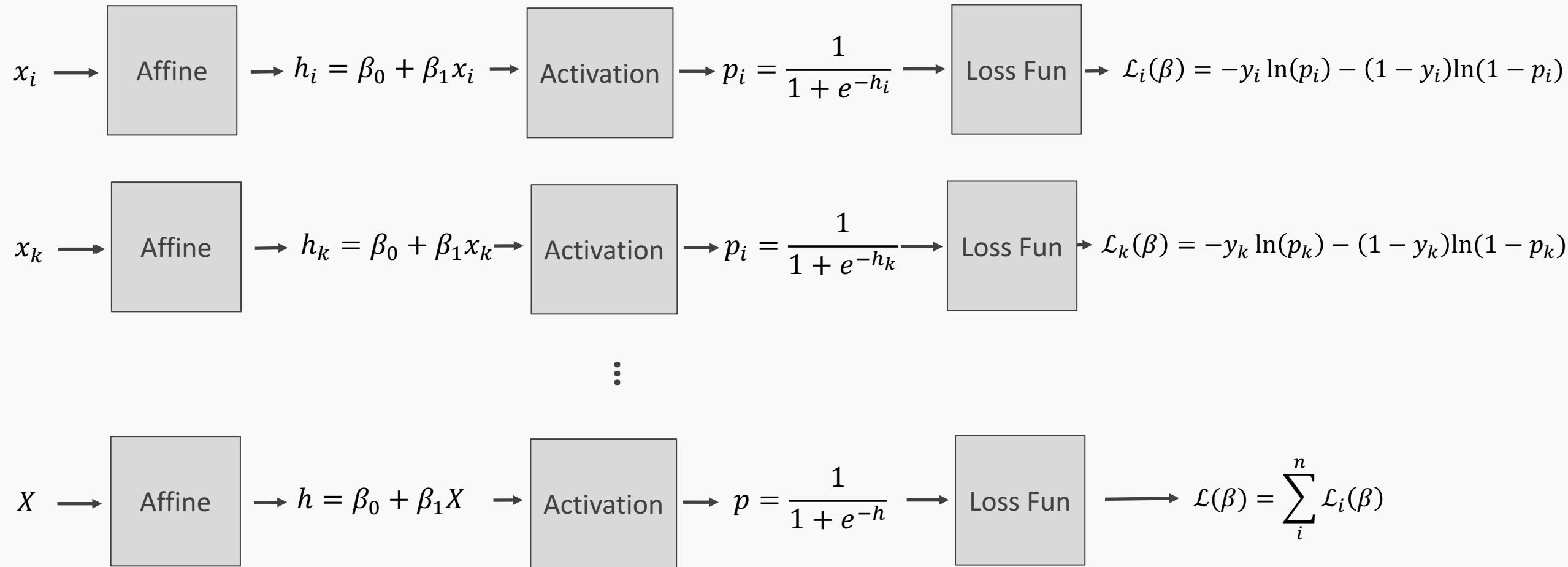
# Estimating the coefficients for Logistic Regression

Find the coefficients that minimize the loss function

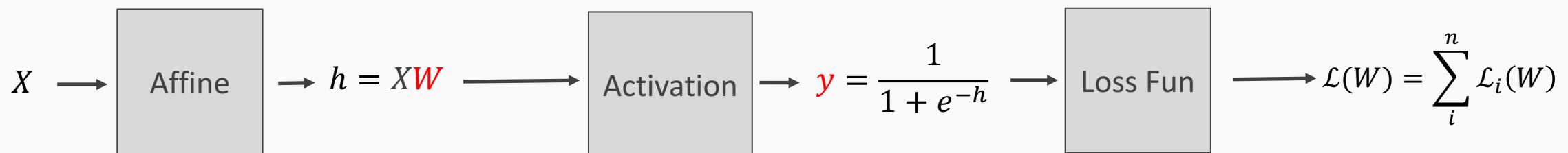
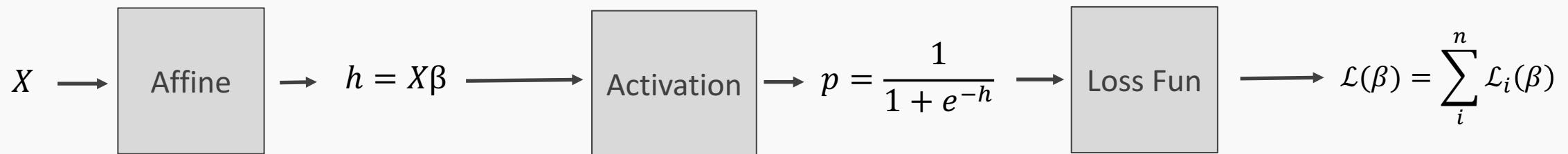
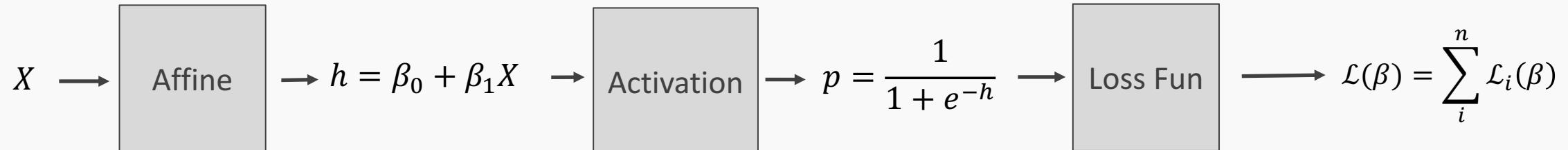
$$\mathcal{L}(\beta_0, \beta_1) = - \sum_i [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$



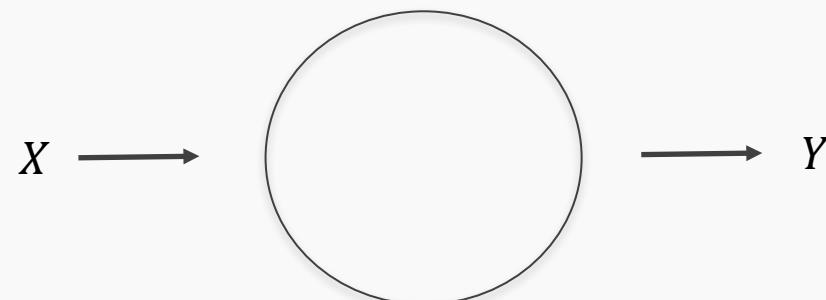
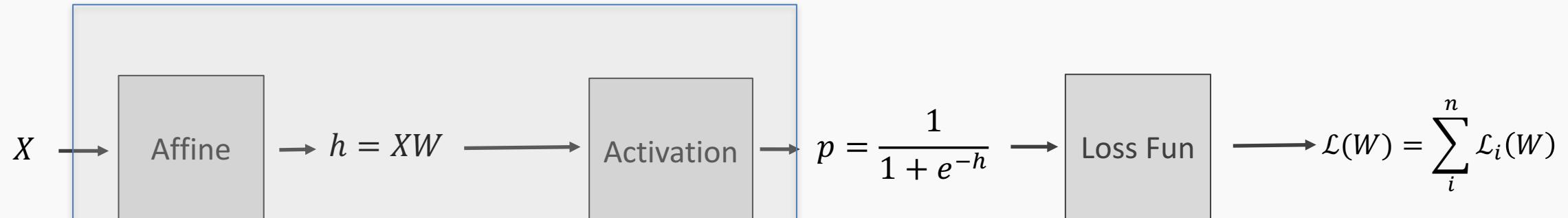
# Logistic Regression Revisited



# Build our first ANN



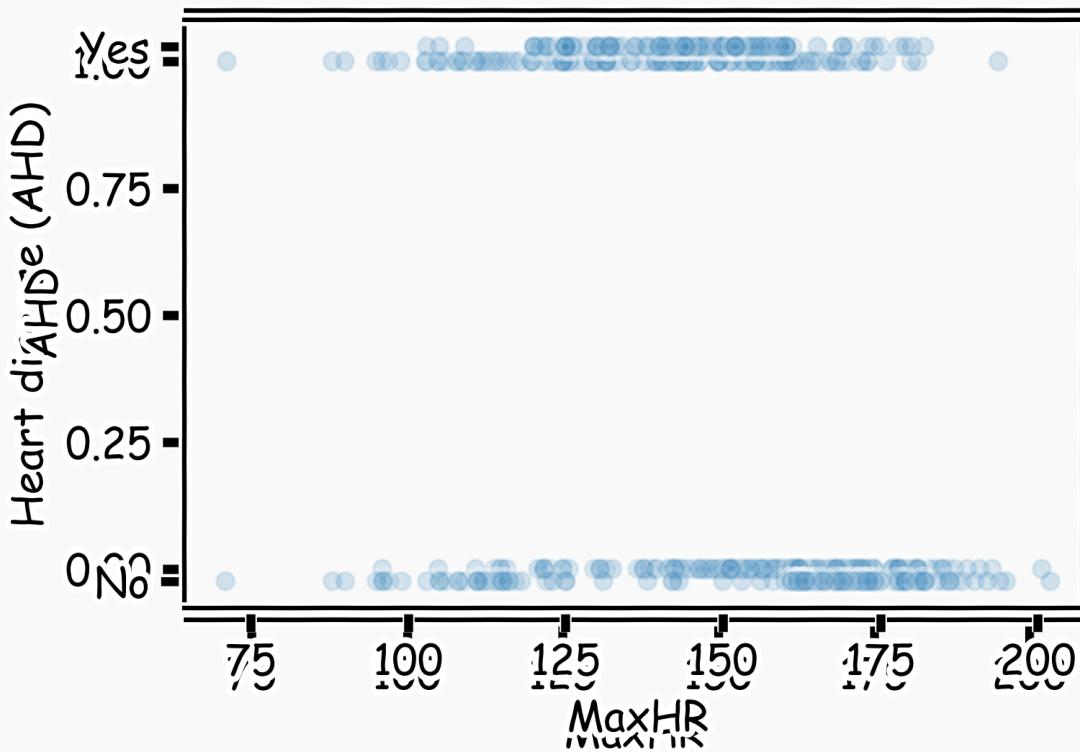
# Build our first ANN



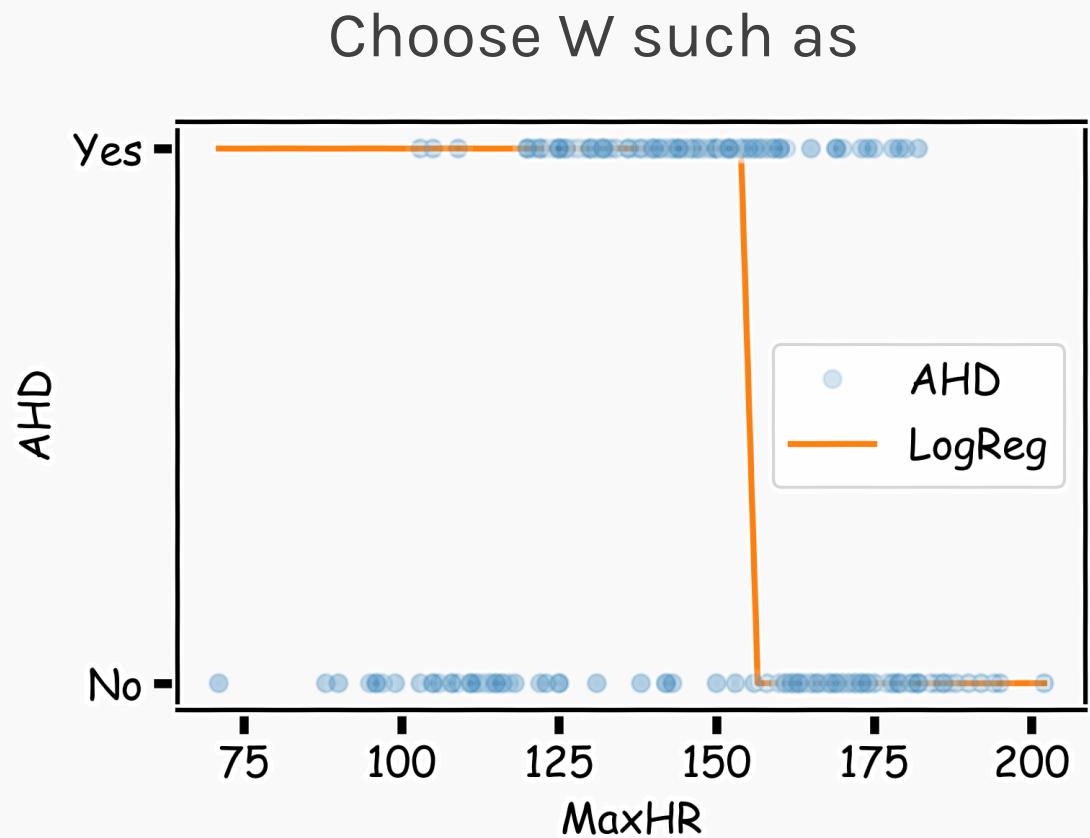
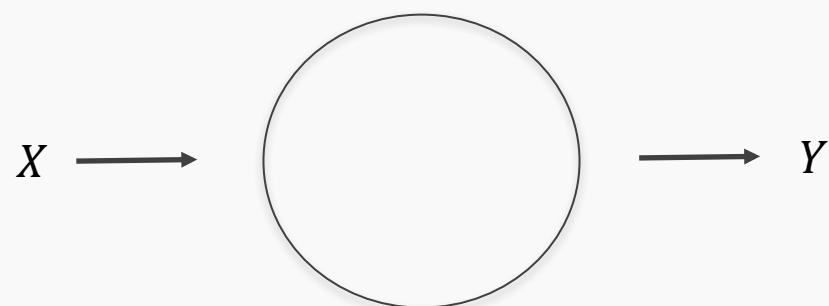
**Single Neuron Network**  
**Very similar to Perceptron**

# Example Using Heart Data

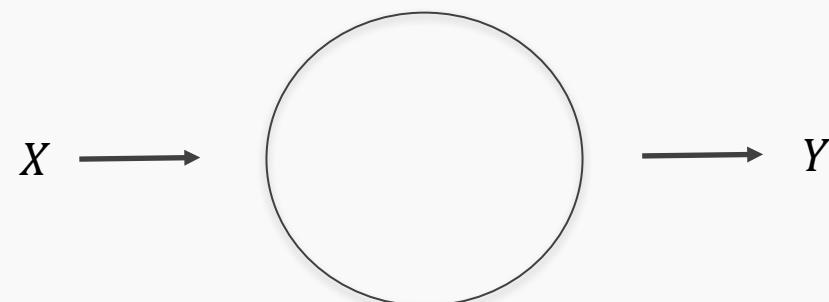
Slightly modified data to illustrate concepts.



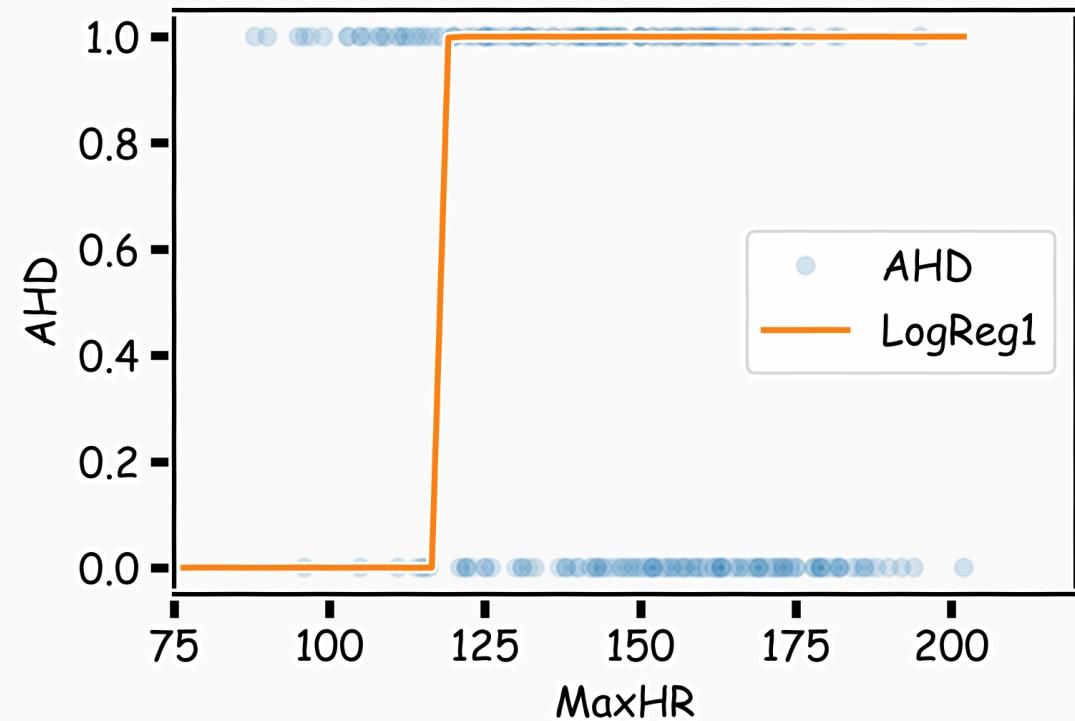
# Example Using Heart Data



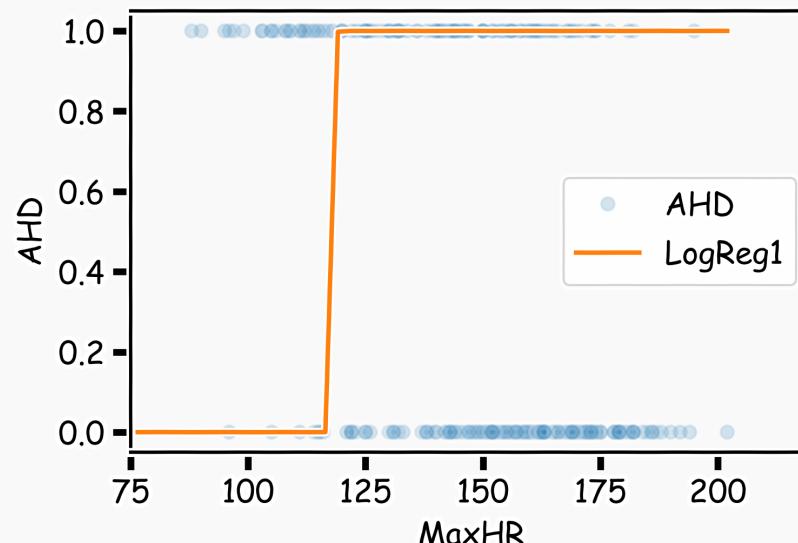
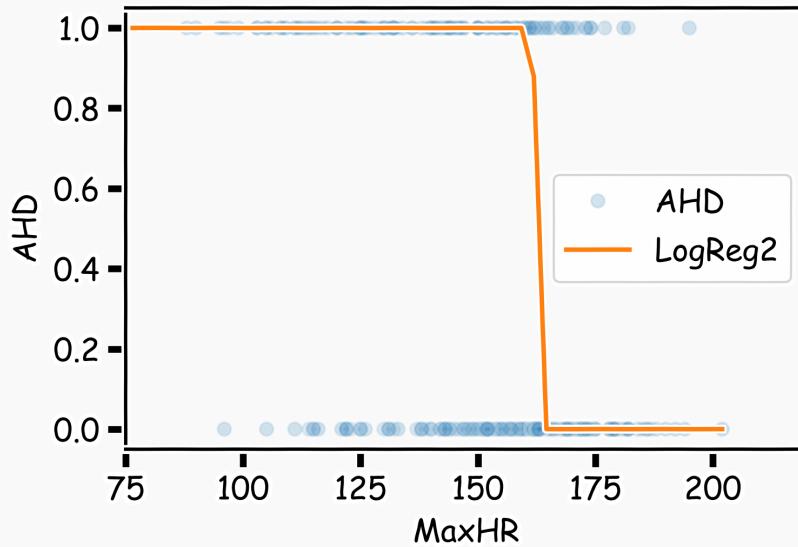
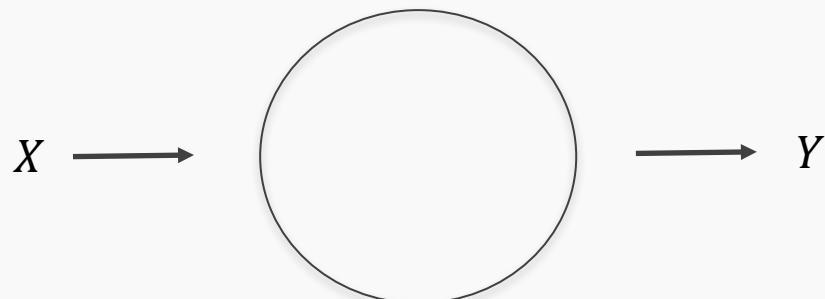
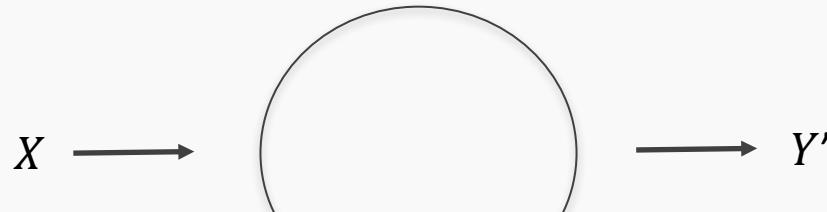
# Example Using Heart Data



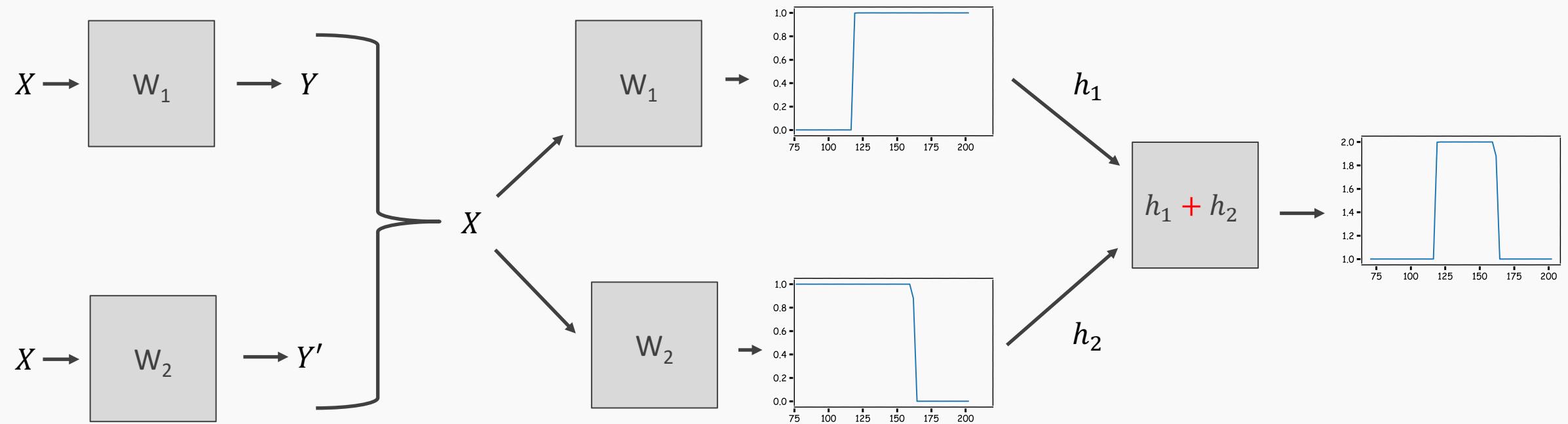
Choose  $W$  such as



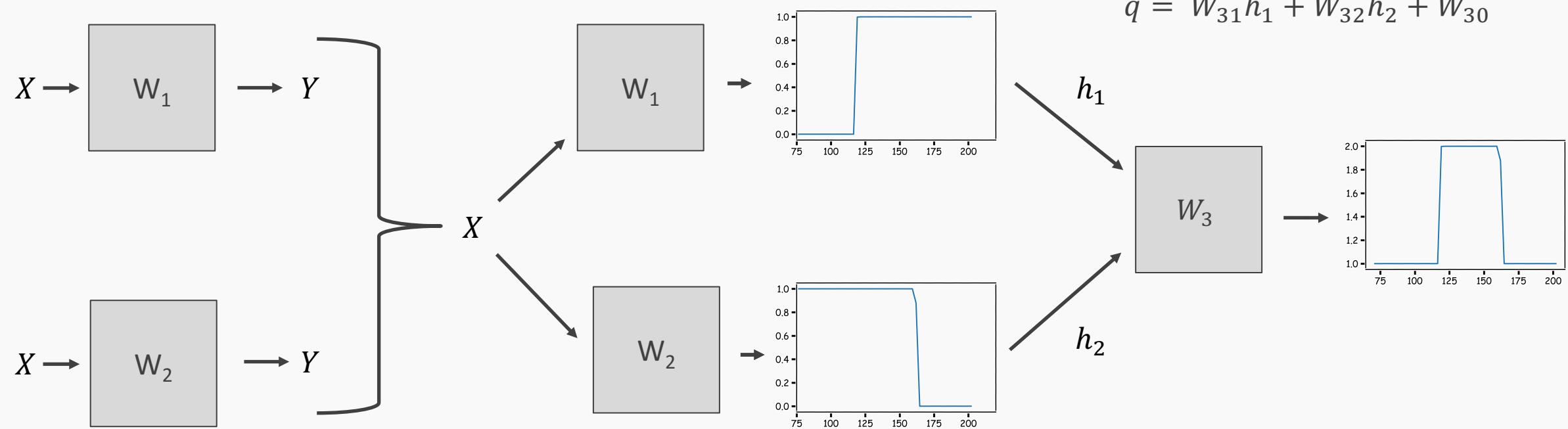
# Example



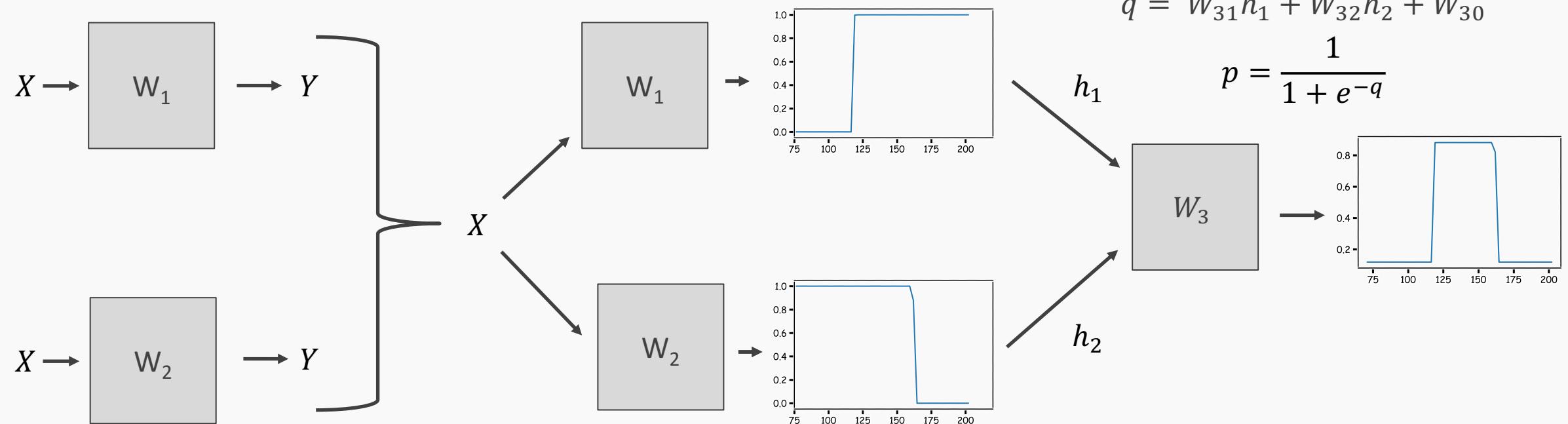
# Pavlos game #232



# Pavlos game #232



# Pavlos game #232



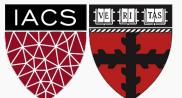
$$L = -y \ln(p) - (1 - y) \ln(1 - p)$$

Need to learn  $W_1$ ,  $W_2$  and  $W_3$ .

# Outline

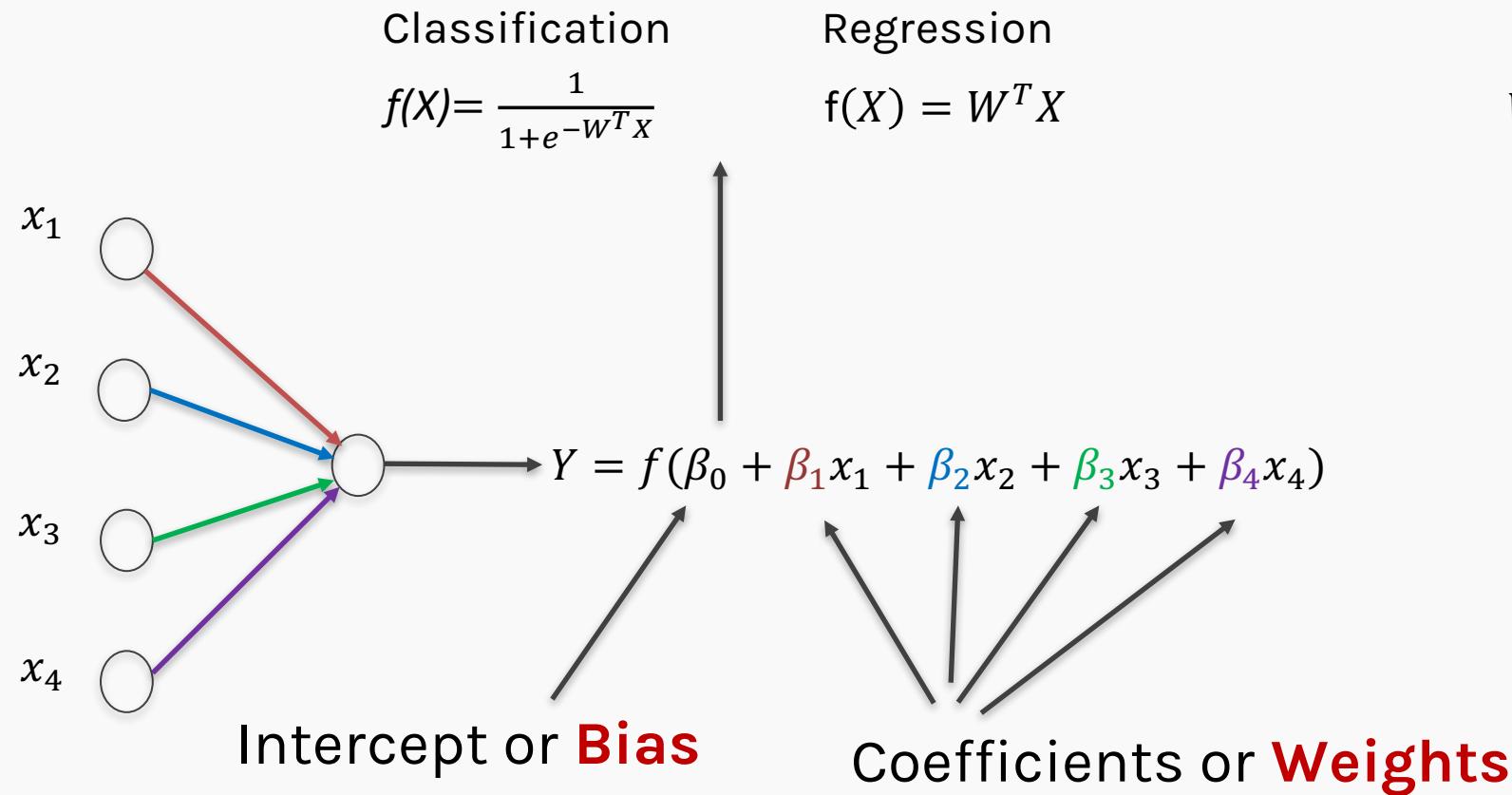
---

1. Description of the course
2. Introduction to Artificial Neural Networks
3. Review of Classification and Logistic Regression
4. Single Neuron Network ('Perceptron')
5. **Introduction to Optimization**
  - Gradient Descent
  - Stochastic Gradient Descent
6. Back Propagation
7. Multi-Layer Perceptron



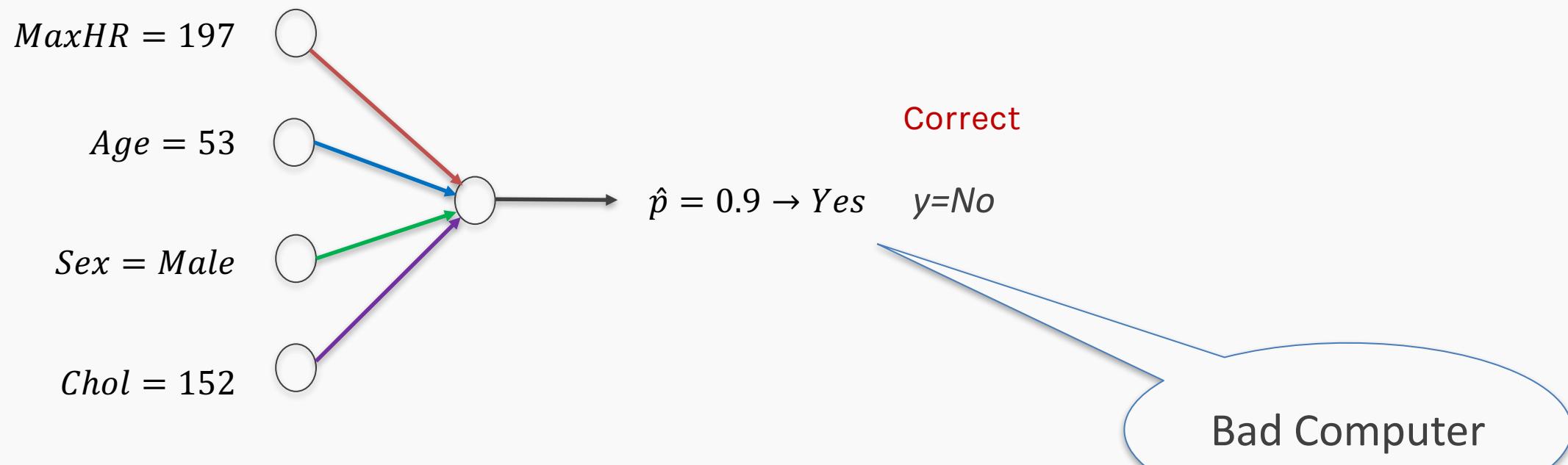
# Learning the coefficients

Start with Regression or Logistic Regression



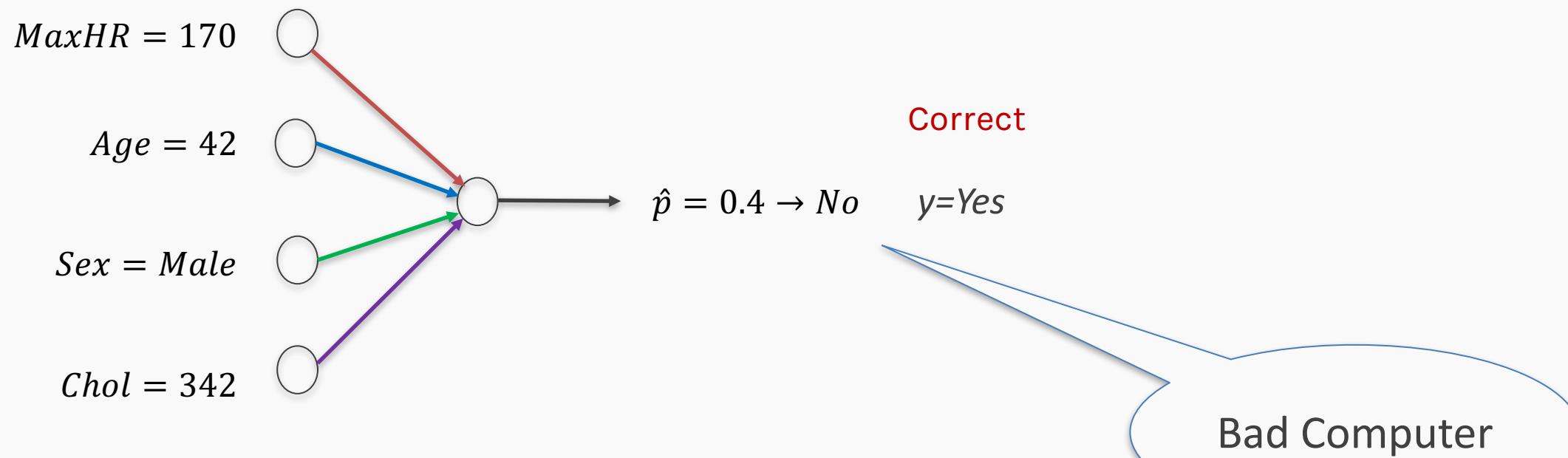
# But what is the idea?

Start with all randomly selected weights. Most likely it will perform horribly. For example, in our heart data, the model will be giving us the wrong answer.



# But what is the idea?

Start with all randomly selected weights. Most likely it will perform horribly. For example, in our heart data, the model will be giving us the wrong answer.



# But what is the idea?

---

- **Loss Function:** Takes all of these results and averages them and tells us how bad or good the computer or those weights are.
- Telling the computer how **bad** or **good** is, does not help.
- You want to tell it how to change those weights so it gets better.

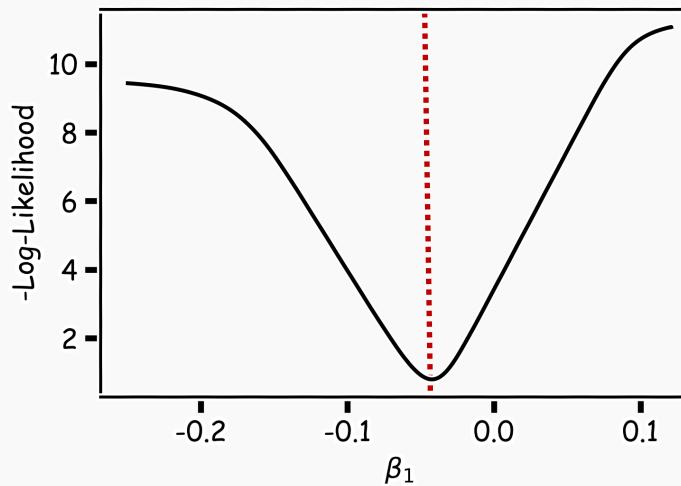
Loss function:  $\mathcal{L}(w_0, w_1, w_2, w_3, w_4)$

For now let's only consider one weight,  $\mathcal{L}(w_1)$



# Minimizing the Loss function

Ideally we want to know the value of  $w_1$  that gives the minimum  $\mathcal{L}(W)$

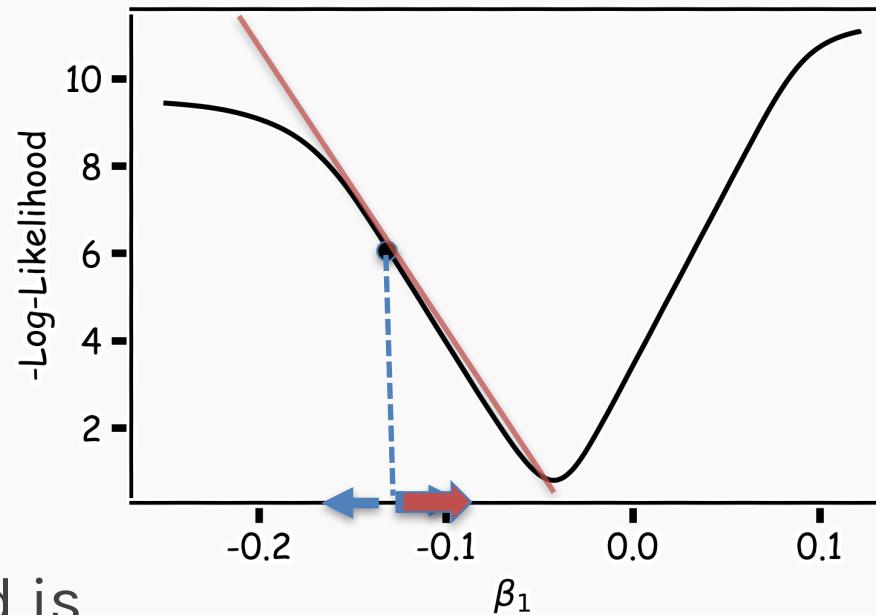


To find the optimal point of a function  $\mathcal{L}(W)$

$$\frac{d\mathcal{L}(W)}{dW} = 0$$

And find the  $W$  that satisfies that equation. Sometimes there is no explicit solution for that.

# Estimate of the regression coefficients: gradient descent



A more flexible method is

- Start from a random point
  1. Determine which direction to go to reduce the loss (left or right)
  2. Compute the slope of the function at this point and step to the right if slope is negative or step to the left if slope is positive
  3. Goto to #1

# Minimization of the Loss Function

If the step is proportional to the slope then you avoid overshooting the minimum.

**Question:** What is the mathematical function that describes the slope?

**Question:** How do we generalize this to more than one predictor?

**Question:** What do you think it is a good approach for telling the model how to change (what is the step size) to become better?



# Minimization of the Loss Function

If the step is proportional to the slope then you avoid overshooting the minimum.

**Question:** What is the mathematical function that describes the slope?

**Derivative**

**Question:** How do we generalize this to more than one predictor?

**Take the derivative with respect to each coefficient and do the same sequentially**

**Question:** What do you think it is a good approach for telling the model how to change (what is the step size) to become better?

# Let's play the Pavlos game

We know that we want to go in the opposite direction of the derivative and we know we want to be making a step proportionally to the derivative.

Making a step means:

$$w^{new} = w^{old} + step$$

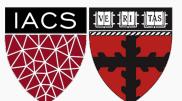
Opposite direction of the derivative means:

$$w^{new} = w^{old} - \lambda \frac{d\mathcal{L}}{dw}$$

Learning  
Rate

Change to more conventional notation:

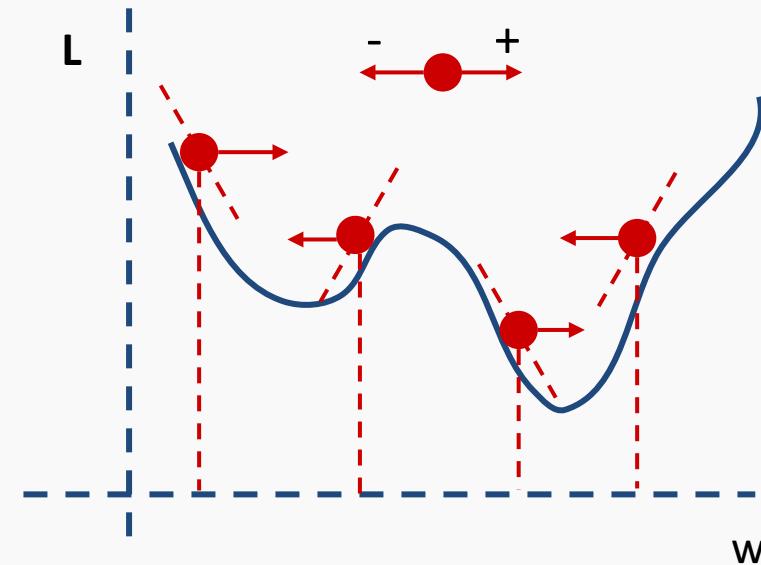
$$w^{(i+1)} = w^{(i)} - \lambda \frac{d\mathcal{L}}{dw}$$



# Gradient Descent

- Algorithm for optimization of first order to finding a minimum of a function.
- It is an iterative method.
- $L$  is decreasing much faster in the direction of the negative derivative.
- The learning rate is controlled by the magnitude of  $\lambda$ .

$$w^{(i+1)} = w^{(i)} - \lambda \frac{d\mathcal{L}}{dw}$$



# Considerations

---

- We still need to calculate the derivatives.
- We need to know what is the learning rate or how to set it.
- Local vs global minima.
- The full likelihood function includes summing up all individual ‘errors’. Unless you are a statistician, sometimes this includes hundreds of thousands of examples.



# Logistic Regression Derivatives

Can we do it?

Wolfram Alpha can do it f



We need a formalism to deal with these derivatives.

# Chain Rule

- Chain rule for computing gradients:

$$\bullet \quad y = g(x) \quad z = f(y) = f(g(x)) \quad y = g(\mathbf{x}) \quad z = f(\mathbf{y}) = f(g(\mathbf{x}))$$

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y} \frac{\partial y}{\partial x}$$

$$\frac{\partial z}{\partial x_i} = \sum_j \frac{\partial z}{\partial y_j} \frac{\partial y_j}{\partial x_i}$$

- For longer chains

$$\frac{\partial z}{\partial x_i} = \sum_{j_1} \dots \sum_{j_m} \frac{\partial z}{\partial y_{j_1}} \dots \frac{\partial y_{j_m}}{\partial x_i}$$



# Logistic Regression derivatives

For logistic regression, the -ve log of the likelihood is:

$$\mathcal{L} = \sum_i \mathcal{L}_i = - \sum_i \log L_i = - \sum_i [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

$$\mathcal{L}_i = -y_i \log \frac{1}{1 + e^{-W^T X}} - (1 - y_i) \log \left(1 - \frac{1}{1 + e^{-W^T X}}\right)$$

To simplify the analysis let us split it into two parts,

$$\mathcal{L}_i = \mathcal{L}_i^A + \mathcal{L}_i^B$$

So the derivative with respect to  $W$  is:

$$\frac{\partial \mathcal{L}}{\partial W} = \sum_i \frac{\partial \mathcal{L}_i}{\partial W} = \sum_i \left( \frac{\partial \mathcal{L}_i^A}{\partial W} + \frac{\partial \mathcal{L}_i^B}{\partial W} \right)$$



$$\mathcal{L}_i^A = -y_i \log \frac{1}{1 + e^{-W^T X}}$$

Variables	Partial derivatives	Partial derivatives
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	$\frac{\partial \xi_1}{\partial W} = -X$
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{-W^T X}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{(1 + e^{-W^T X})^2}$
$\xi_5 = \log \xi_4 = \log p = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	$\frac{\partial \xi_5}{\partial \xi_4} = 1 + e^{-W^T X}$
$\mathcal{L}_i^A = -y \xi_5$	$\frac{\partial \mathcal{L}_i^A}{\partial \xi_5} = -y$	$\frac{\partial \mathcal{L}_i^A}{\partial \xi_5} = -y$
$\frac{\partial \mathcal{L}_i^A}{\partial W} = \frac{\partial \mathcal{L}_i^A}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$		$\frac{\partial \mathcal{L}_i^A}{\partial W} = -y X e^{-W^T X} \frac{1}{(1 + e^{-W^T X})}$

$$\mathcal{L}_i^B = -(1 - y_i) \log\left[1 - \frac{1}{1 + e^{-W^T X}}\right]$$

Variables	derivatives	Partial derivatives wrt to X,W
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	$\frac{\partial \xi_1}{\partial W} = -X$
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{-W^T X}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$\frac{\partial \xi_3}{\partial 2} = 1$
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{(1 + e^{-W^T X})^2}$
$\xi_5 = 1 - \xi_4 = 1 - \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = -1$	$\frac{\partial \xi_5}{\partial \xi_4} = -1$
$\xi_6 = \log \xi_5 = \log(1 - p) = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_6}{\partial \xi_5} = \frac{1}{\xi_5}$	$\frac{\partial \xi_6}{\partial \xi_5} = \frac{1 + e^{-W^T X}}{e^{-W^T X}}$
$\mathcal{L}_i^B = (1 - y)\xi_6$	$\frac{\partial \mathcal{L}}{\partial \xi_6} = 1 - y$	$\frac{\partial \mathcal{L}}{\partial \xi_6} = 1 - y$
$\frac{\partial \mathcal{L}_i^B}{\partial W} = \frac{\partial \mathcal{L}_i^B}{\partial \xi_6} \frac{\partial \xi_6}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$		$\frac{\partial \mathcal{L}_i^B}{\partial W} = (1 - y)X \frac{1}{(1 + e^{-W^T X})}$



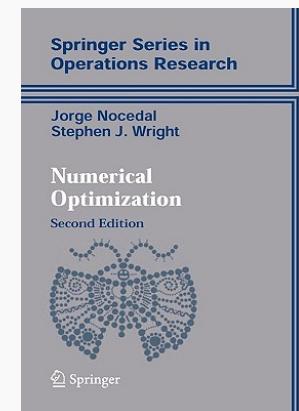
# Learning Rate

# Learning Rate

## Trial and Error.

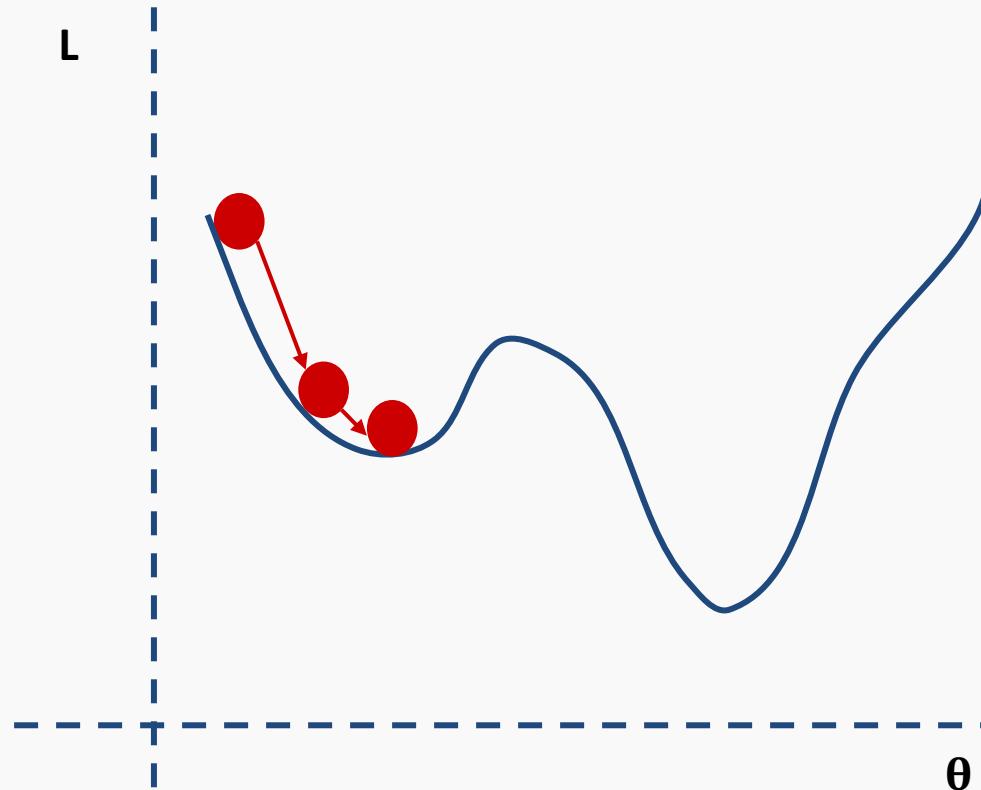
There are many alternative methods which address how to set or adjust the learning rate, using the derivative or second derivatives and or the momentum. To be discussed in the next lectures on NN.

- \* J. Nocedal y S. Wright, “Numerical optimization”, Springer, 1999 
- \* *TLDR*: J. Bullinaria, “Learning with Momentum, Conjugate Gradient Learning”, 2015 

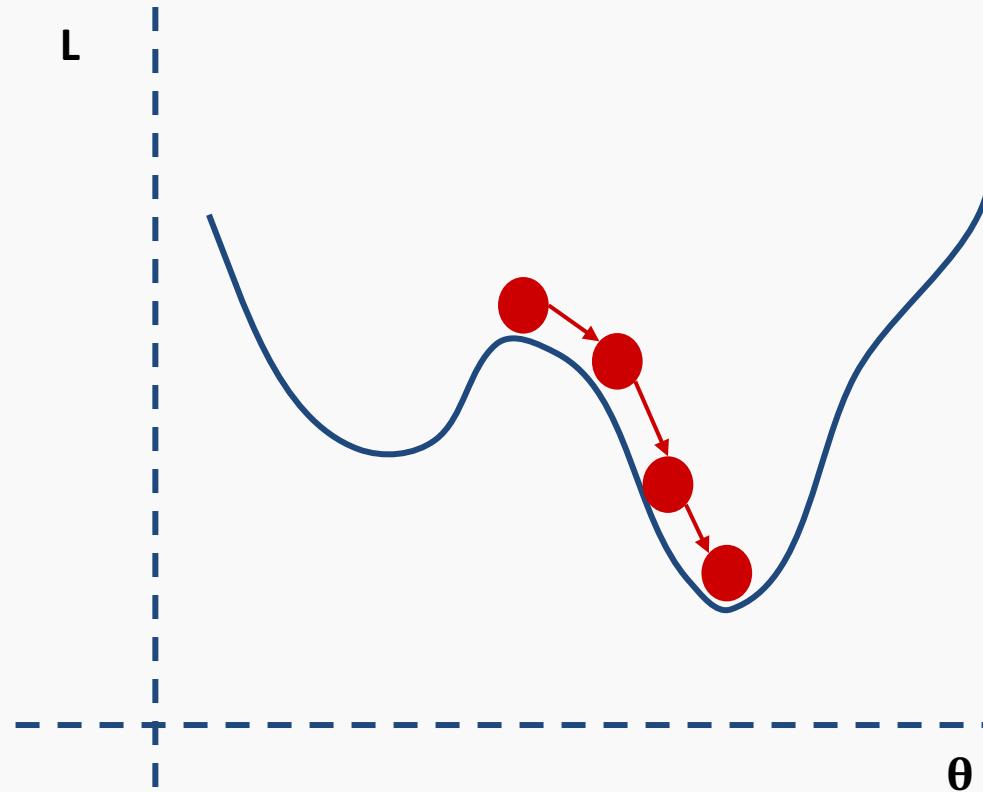


# Local and Global minima

# Local vs Global Minima



# Local vs Global Minima



# Local vs Global Minima

---

No guarantee that we get the global minimum.

**Question:** What would be a good strategy?



# Large data



# Batch and Stochastic Gradient Descent

$$\mathcal{L} = - \sum_i [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

Instead of using all the examples for every step, use a subset of them (batch).

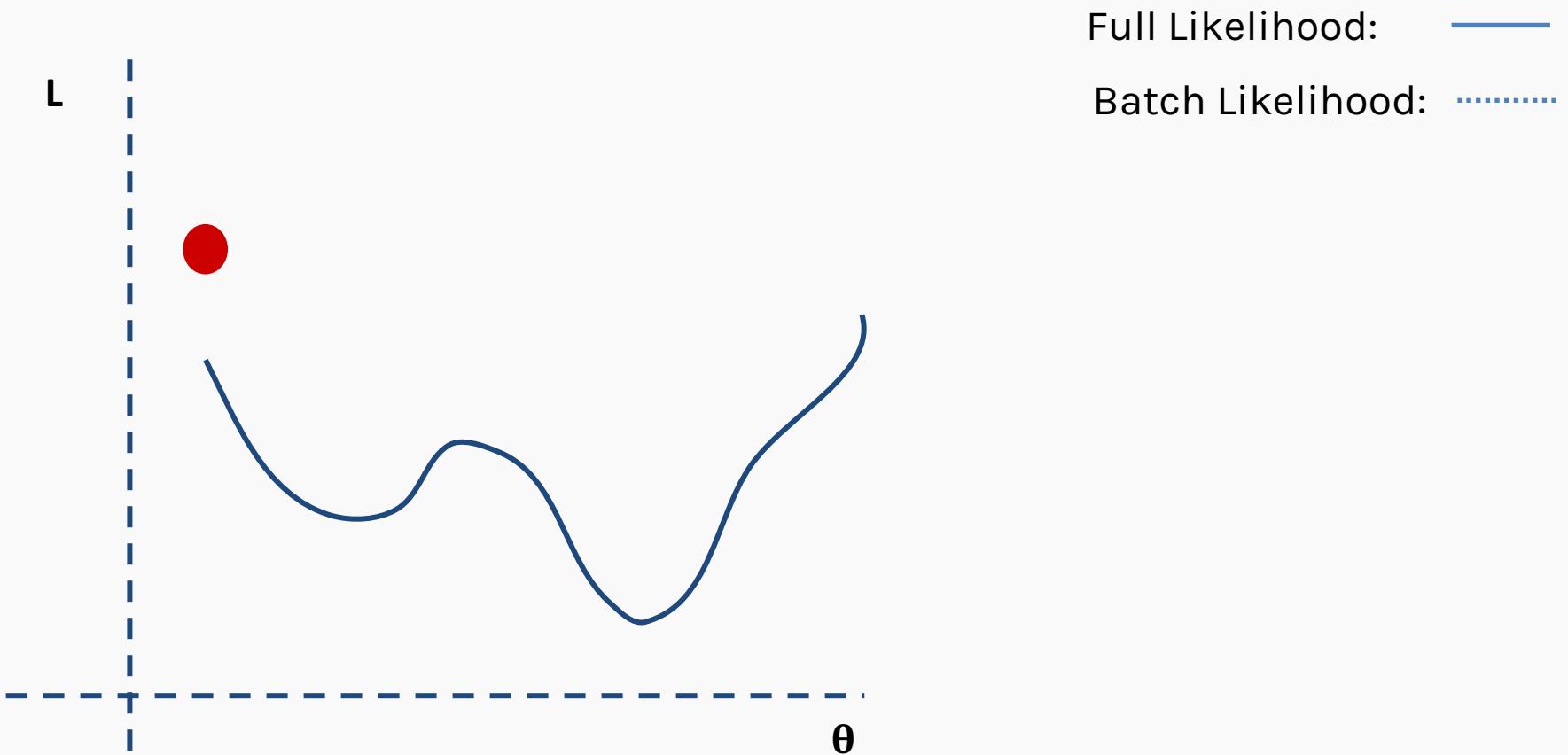
For each iteration  $k$ , use the following loss function to derive the derivatives:

$$\mathcal{L}^k = - \sum_{i \in b^k} [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$

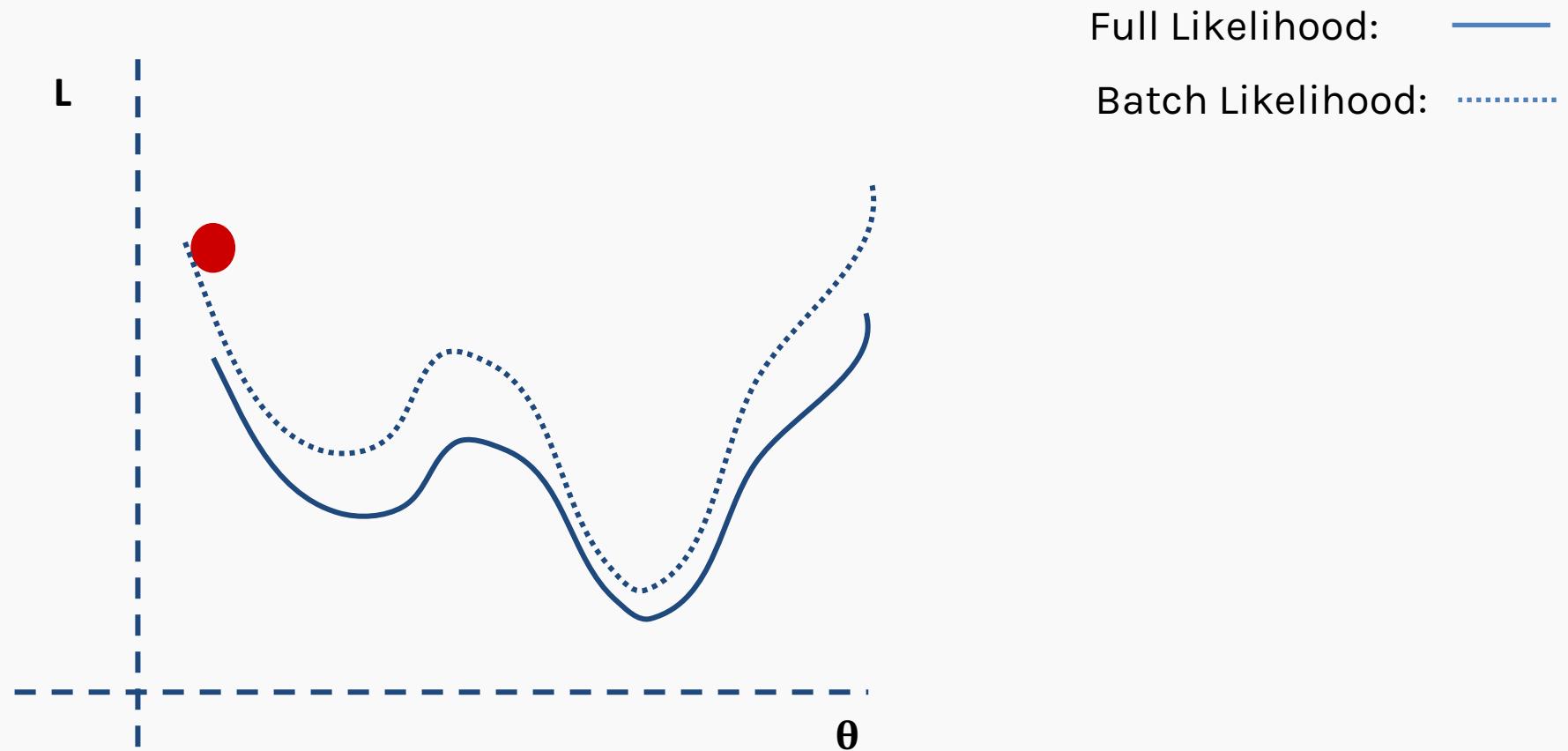
which is an **approximation** to the full Loss function.



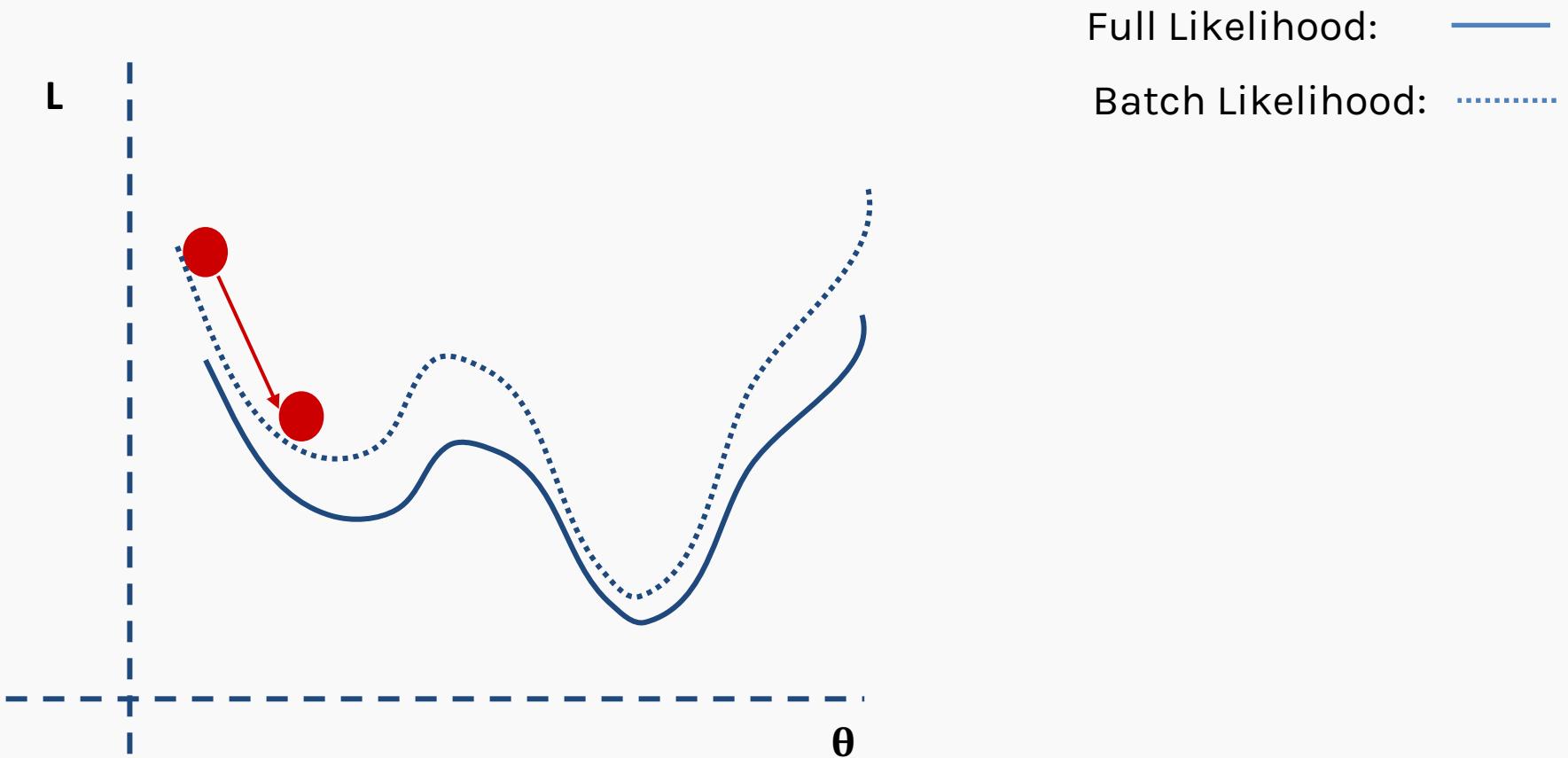
# Batch and Stochastic Gradient Descent



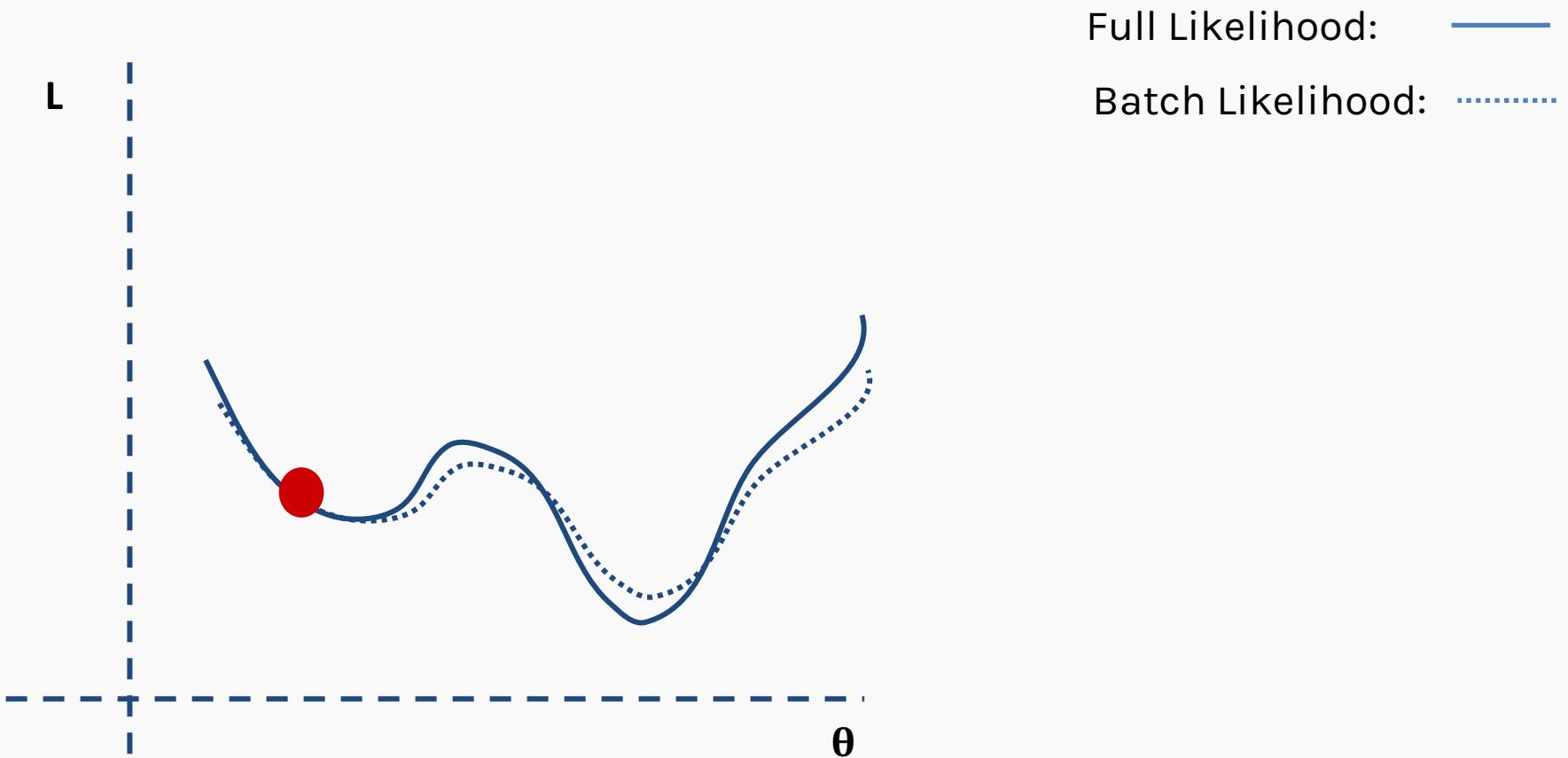
# Batch and Stochastic Gradient Descent



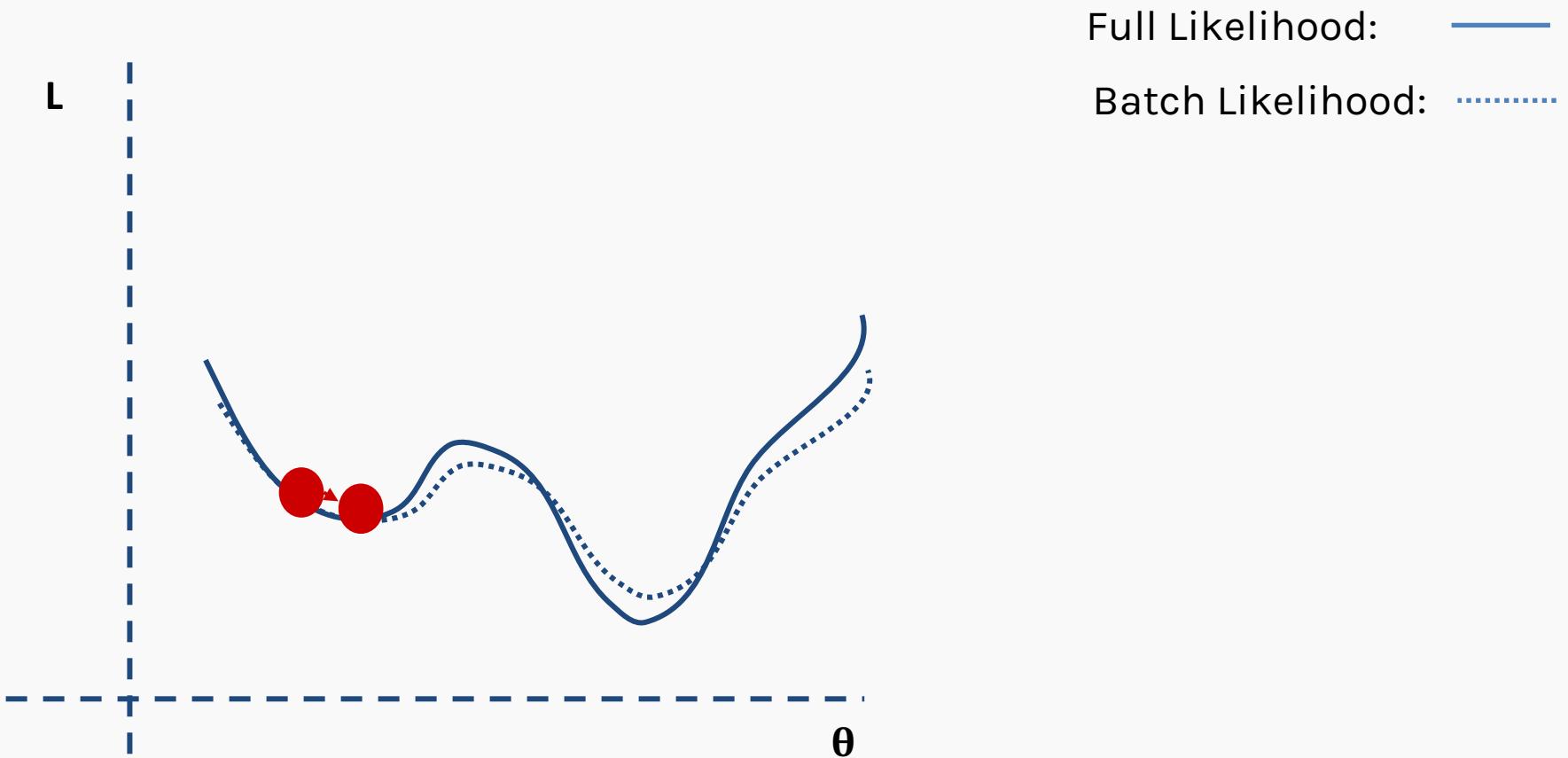
# Batch and Stochastic Gradient Descent



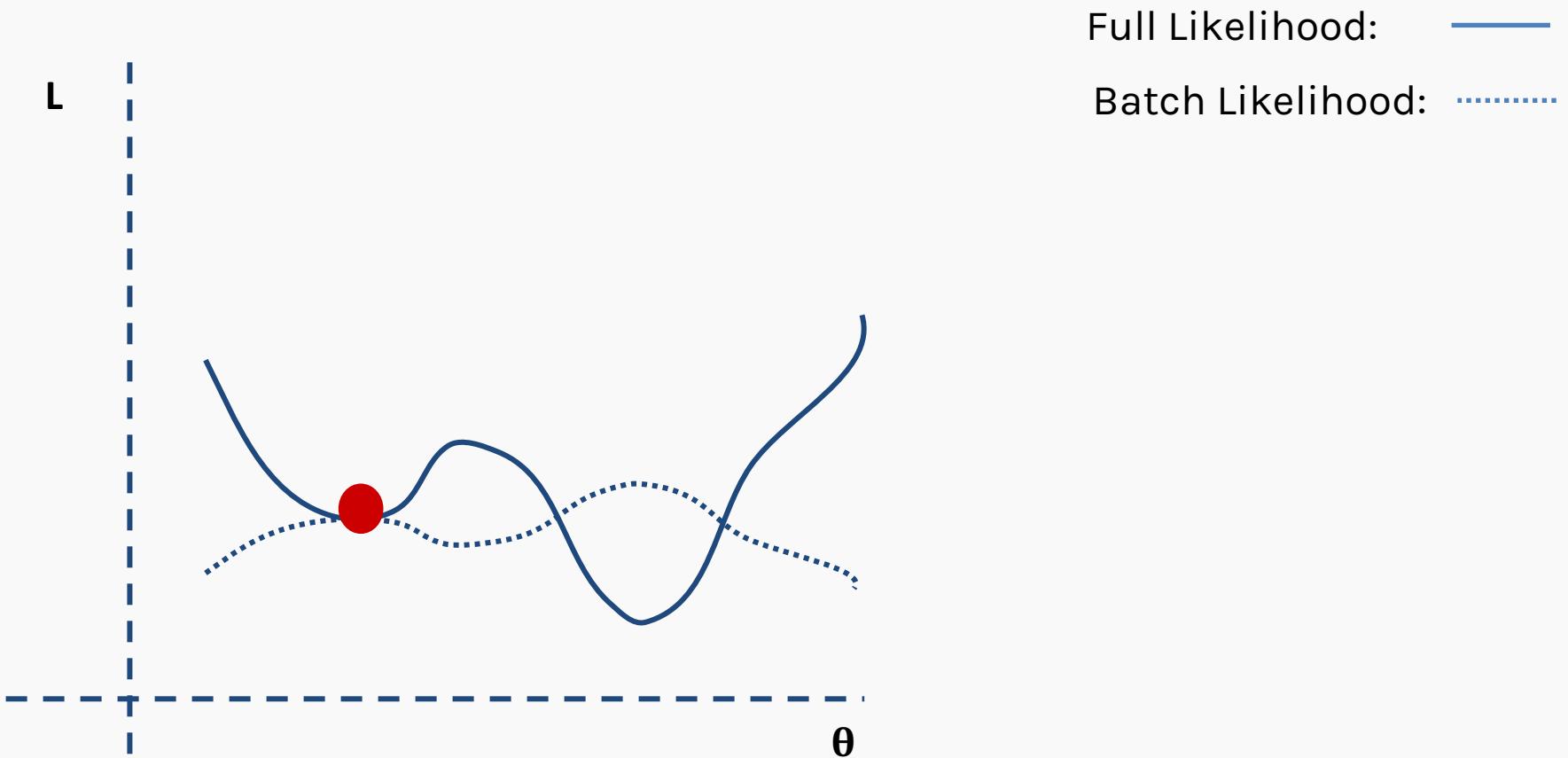
# Batch and Stochastic Gradient Descent



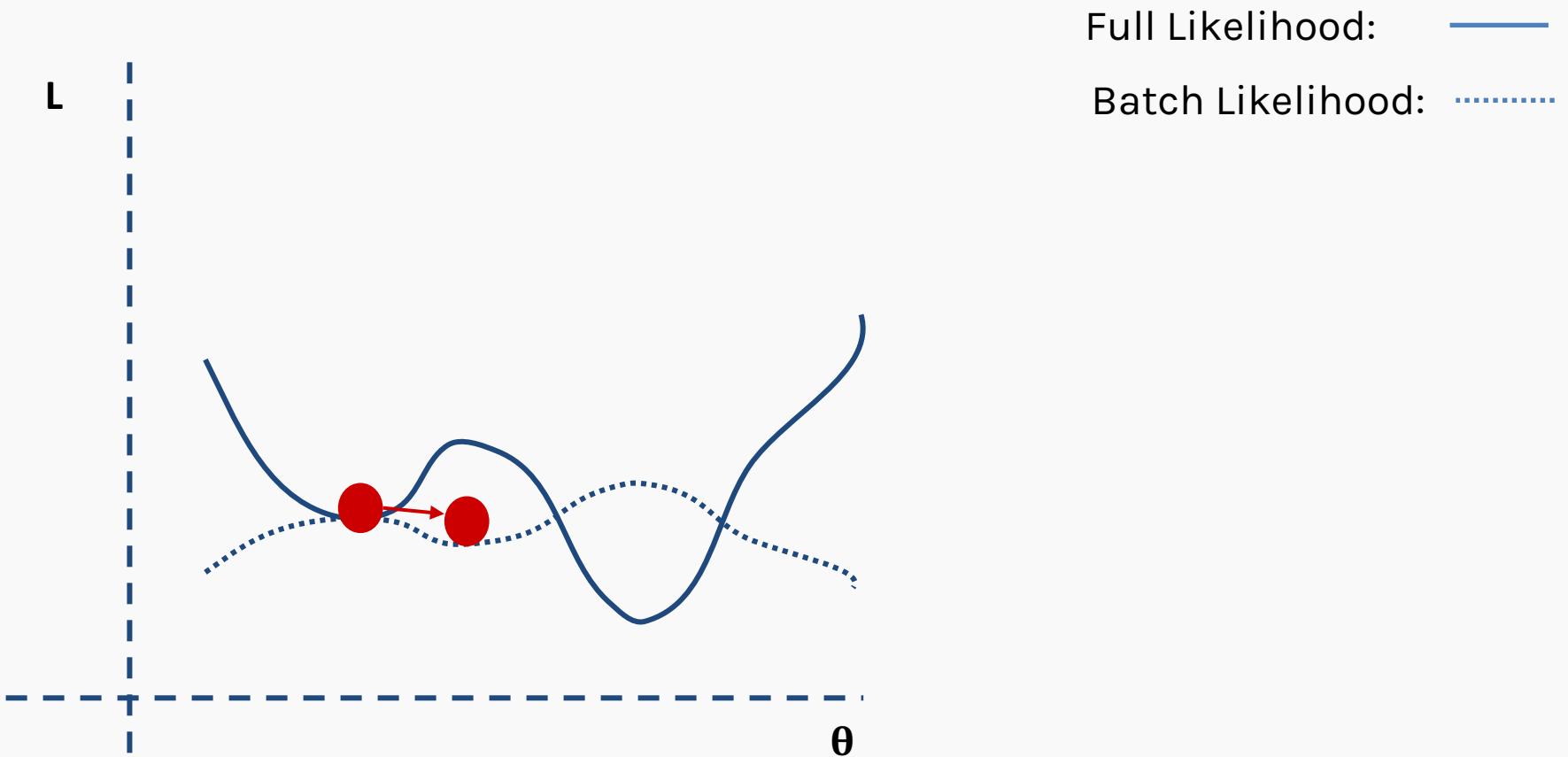
# Batch and Stochastic Gradient Descent



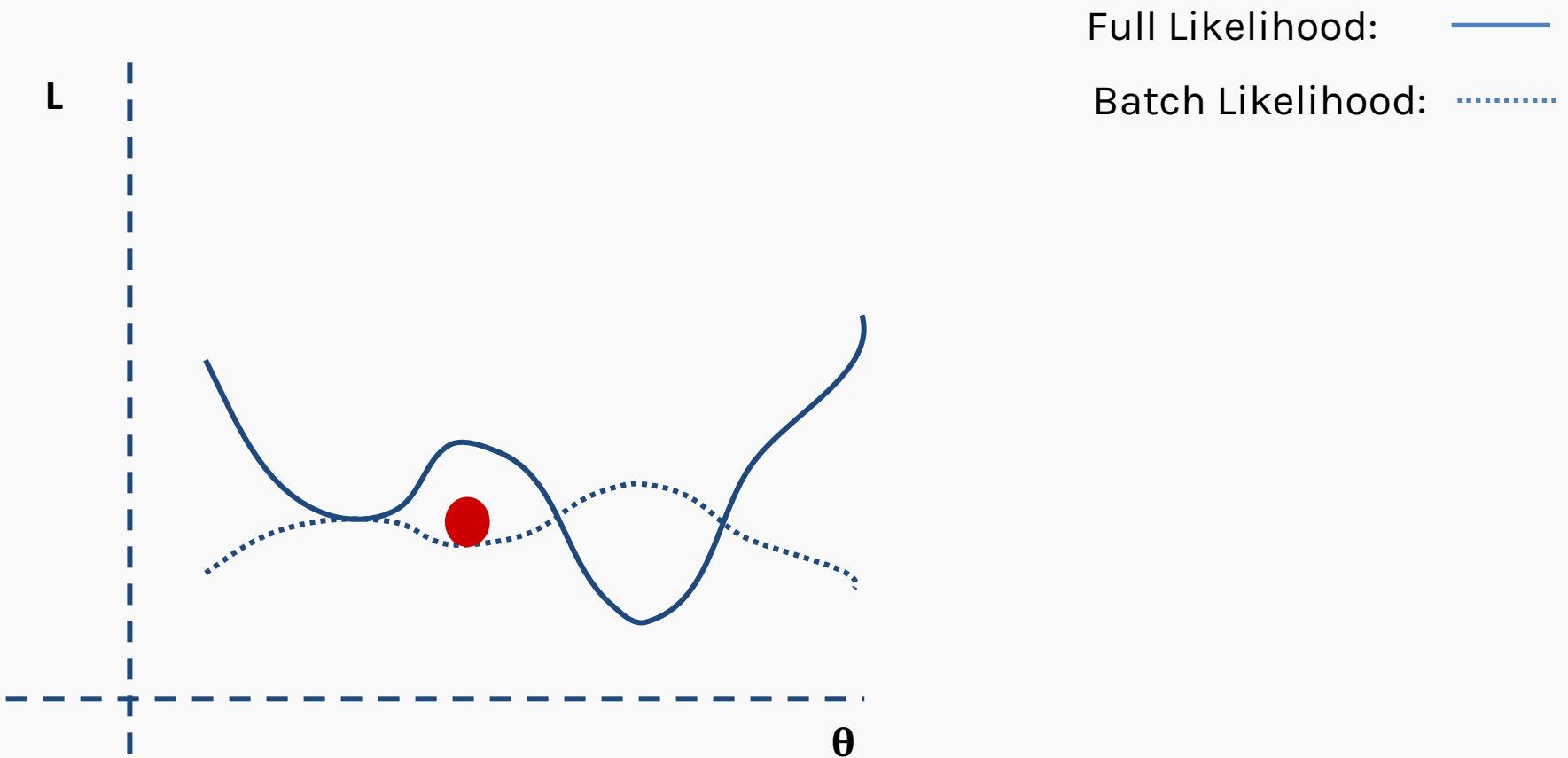
# Batch and Stochastic Gradient Descent



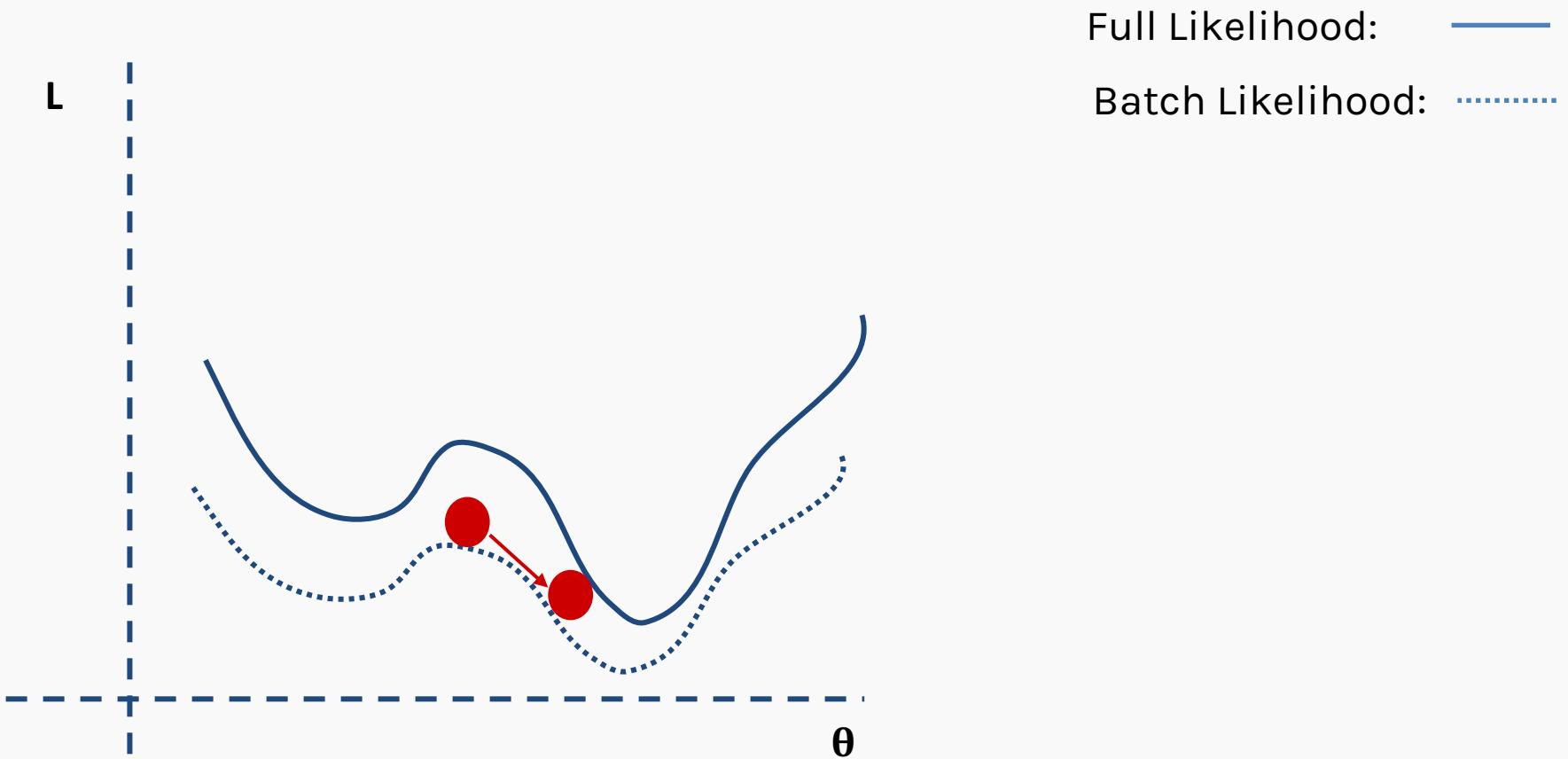
# Batch and Stochastic Gradient Descent



# Batch and Stochastic Gradient Descent



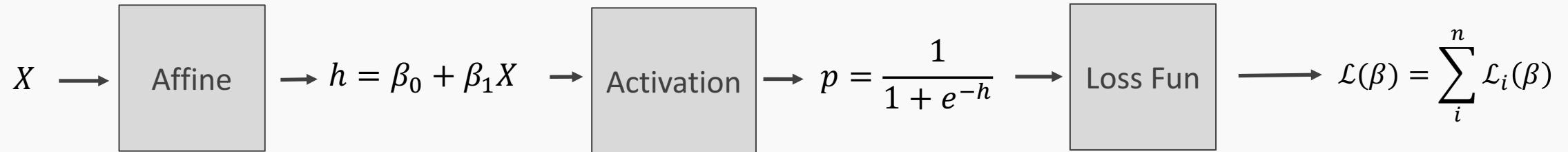
# Batch and Stochastic Gradient Descent



# Backpropagation



# Backpropagation: Logistic Regression Revisited



$$\frac{\partial \mathcal{L}}{\partial p} \frac{\partial p}{\partial h} \frac{\partial h}{\partial \beta}$$

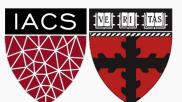
$$\frac{\partial h}{\partial \beta_1} = X, \frac{d\mathcal{L}}{d\beta_0} = 1$$

$$\frac{\partial p}{\partial h} = \sigma(h)(1 - \sigma(h))$$

$$\frac{\partial \mathcal{L}}{\partial p} = -y \frac{1}{p} - (1 - y) \frac{1}{1 - p}$$

$$\frac{\partial \mathcal{L}}{\partial \beta_1} = -X\sigma(h)(1 - \sigma(h))\left[y \frac{1}{p} + (1 - y) \frac{1}{1 - p}\right]$$

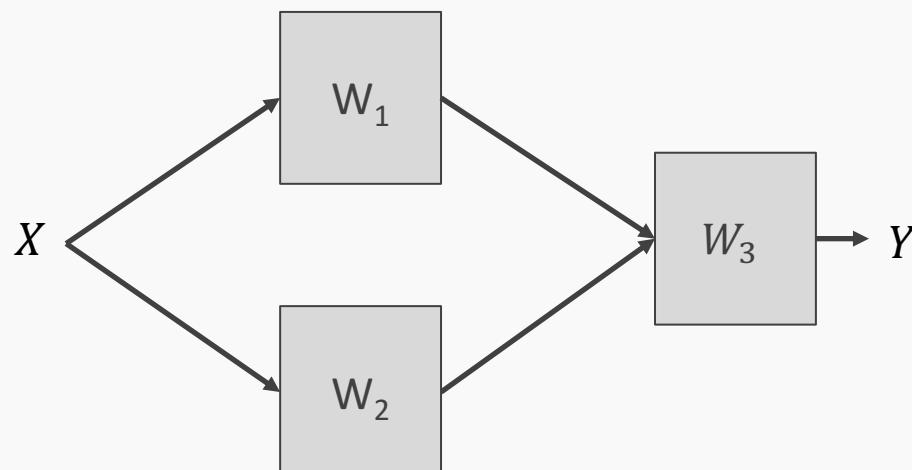
$$\frac{\partial \mathcal{L}}{\partial \beta_0} = -\sigma(h)(1 - \sigma(h))\left[y \frac{1}{p} + (1 - y) \frac{1}{1 - p}\right]$$



# Backpropagation

1. Derivatives need to be evaluated at some values of  $X, y$  and  $W$ .
2. But since we have an expression, we can build a function that takes as input  $X, y, W$  and returns the derivatives and then we can use gradient descent to update.
3. This approach works well but it does not generalize. For example if the network is changed, we need to write a new function to evaluate the derivatives.

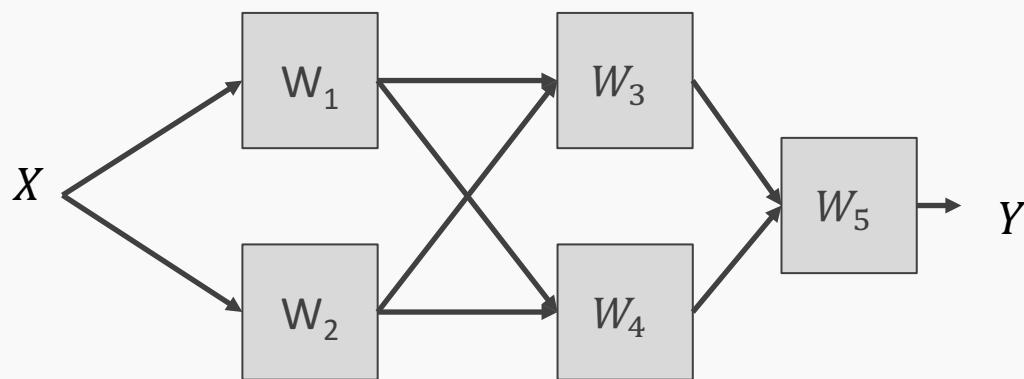
For example this network will need a different function for the derivatives



# Backpropagation

1. Derivatives need to be evaluated at some values of X,y and W.
2. But since we have an expression, we can build a function that takes as input X,y,W and returns the derivatives and then we can use gradient descent to update.
3. This approach works well but it does not generalize. For example if the network is changed, we need to write a new function to evaluate the derivatives.

For example this network will need a different function for the derivatives



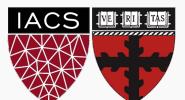
# Backpropagation. Pavlos game #456

Need to find a formalism to calculate the derivatives of the loss wrt to weights that is:

1. Flexible enough that adding a node or a layer or changing something in the network won't require to re-derive the functional form from scratch.
2. It is exact.
3. It is computationally efficient.

Hints:

1. Remember we only need to evaluate the derivatives at  $X_i, y_i$  and  $W^{(k)}$ .
2. We should take advantage of the chain rule we learned before



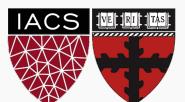
# Idea 1: Evaluate the derivative at: $X=\{3\}$ , $y=1$ , $W=3$

Variables	derivatives	Value of the variable	Value of the partial derivative	$\frac{d\xi_n}{dW}$
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	-9	-3	-3
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$e^{-9}$	$e^{-9}$	$-3e^{-9}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$1+e^{-9}$	1	$-3e^{-9}$
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{1}{1 + e^{-9}}$	$\left(\frac{1}{1 + e^{-9}}\right)^2$	$-3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)^2$
$\xi_5 = \log \xi_4 = \log p = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	$\log \frac{1}{1 + e^{-9}}$	$1 + e^{-9}$	$-3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)$
$\mathcal{L}_i^A = -y \xi_5$	$\frac{\partial \mathcal{L}}{\partial \xi_5} = -y$	$-\log \frac{1}{1 + e^{-9}}$	-1	$3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)$
$\frac{\partial \mathcal{L}_i^A}{\partial W} = \frac{\partial \mathcal{L}_i}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$			-3	0.00037018372

# Basic functions

We still need to derive derivatives ☹

Variables	derivatives	Value of the variable	Value of the partial derivative	$\frac{d\xi_n}{dW}$
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	-9	-3	-3
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$e^{-9}$	$e^{-9}$	$-3e^{-9}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$1+e^{-9}$	1	$-3e^{-9}$
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{1}{1 + e^{-9}}$	$\left(\frac{1}{1 + e^{-9}}\right)^2$	$-3e^{-9} \left(\frac{1}{1+e^{-9}}\right)^2$
$\xi_5 = \log \xi_4 = \log p = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	$\log \frac{1}{1 + e^{-9}}$	$1 + e^{-9}$	$-3e^{-9} \left(\frac{1}{1+e^{-9}}\right)$
$\mathcal{L}_i^A = -y \xi_5$	$\frac{\partial \mathcal{L}_i^A}{\partial \xi_5} = -y$	$-\log \frac{1}{1 + e^{-9}}$	-1	$3e^{-9} \left(\frac{1}{1+e^{-9}}\right)$
$\frac{\partial \mathcal{L}_i^A}{\partial W} = \frac{\partial \mathcal{L}_i}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$			-3	0.00037018372



# Basic functions

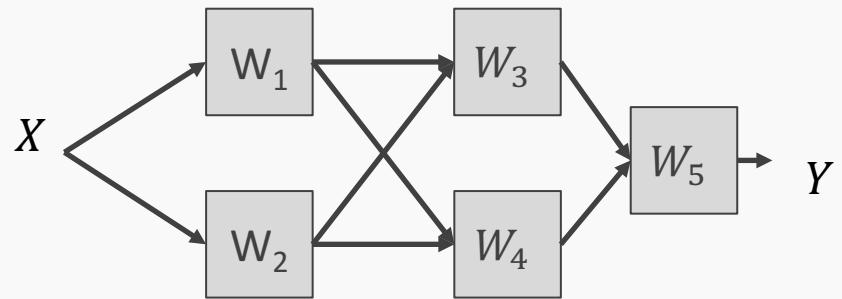
Notice though those are basic functions that my grandparent can do

$\xi_0 = X$	$\frac{\partial \xi_0}{\partial X} = 1$	<pre>def x0(x):     return x</pre>	<pre>def derx0():     return 1</pre>
$\xi_1 = -W^T \xi_0$	$\frac{\partial \xi_1}{\partial W} = -X$	<pre>def x1(a,x):     return -a*x</pre>	<pre>def derx1(a,x):     return -a</pre>
$\xi_2 = e^{\xi_1}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	<pre>def x2(x):     return np.exp(x)</pre>	<pre>def derx2(x):     return np.exp(x)</pre>
$\xi_3 = 1 + \xi_2$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	<pre>def x3(x):     return 1+x</pre>	<pre>def derx3(x):     return 1</pre>
$\xi_4 = \frac{1}{\xi_3}$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	<pre>def der1(x):     return 1/(x)</pre>	<pre>def derx4(x):     return -(1/x)**(2)</pre>
$\xi_5 = \log \xi_4$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	<pre>def der1(x):     return np.log(x)</pre>	<pre>def derx5(x):     return 1/x</pre>
$\mathcal{L}_i^A = -y \xi_5$	$\frac{\partial \mathcal{L}}{\partial \xi_5} = -y$	<pre>def der1(y,x):     return -y*x</pre>	<pre>def derL(y):     return -y</pre>



# Putting it altogether

1. We specify the network structure

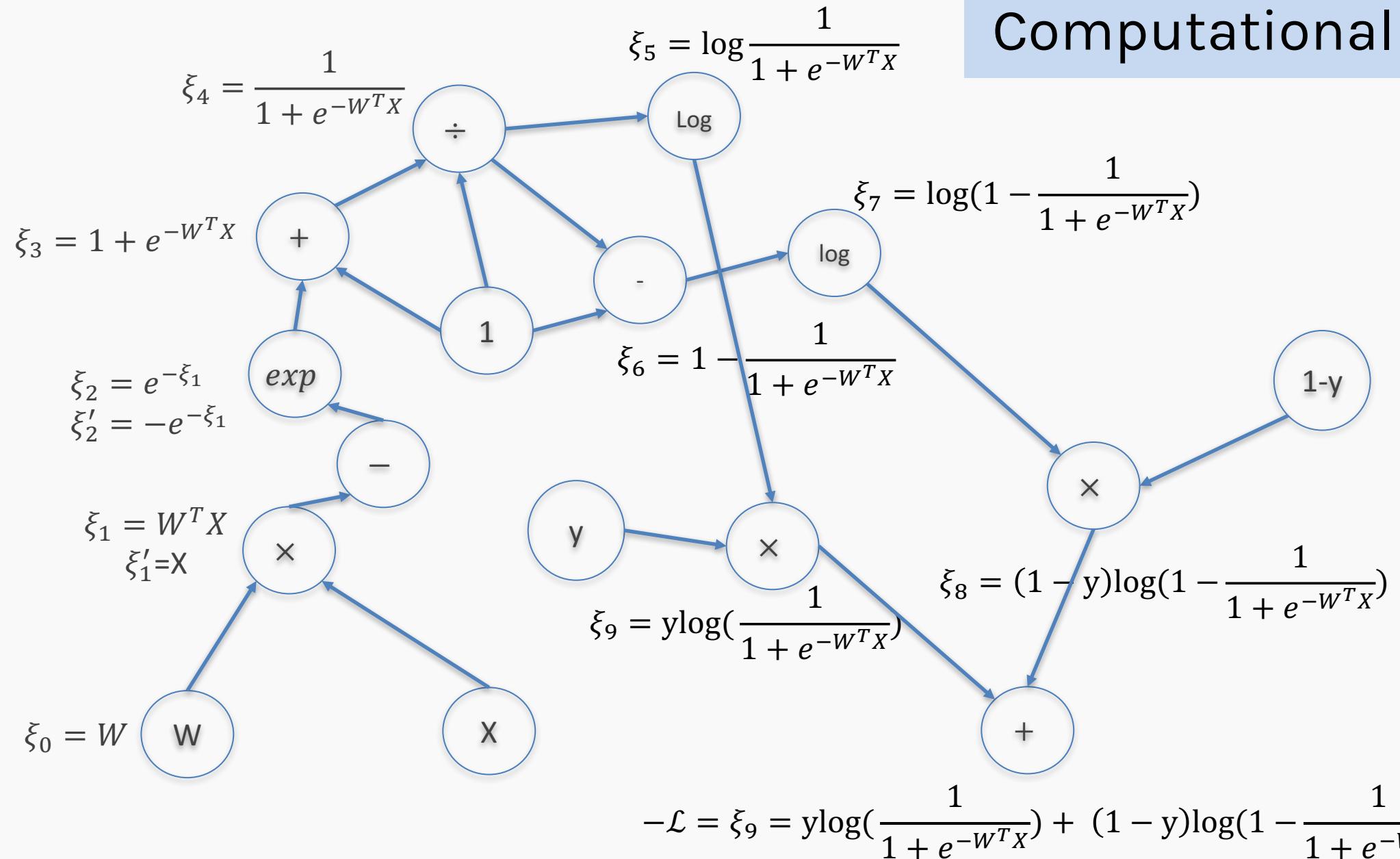


2. We create the computational graph ...

What is computational graph?



# Computational Graph

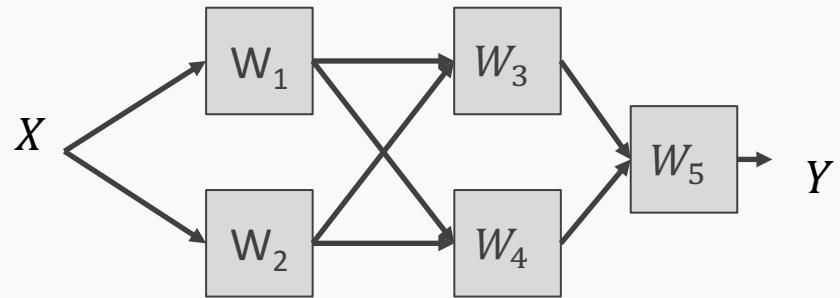


$$-\mathcal{L} = \xi_9 = y \log\left(\frac{1}{1 + e^{-W^T X}}\right) + (1 - y) \log\left(1 - \frac{1}{1 + e^{-W^T X}}\right)$$

PAVLOS PROTOPAPAS

# Putting it altogether

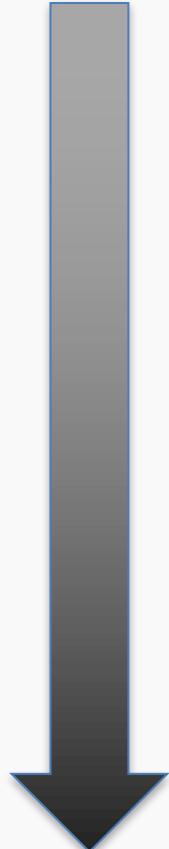
## 1. We specify the network structure



- We create the computational graph.
- At each node of the graph we build two functions: the evaluation of the variable and its partial derivative with respect to the previous variable (as shown in the table 3 slides back)
- Now we can either go forward or backward depending on the situation.  
In general, forward is easier to implement and to understand. The difference is clearer when there are multiple nodes per layer.

# Forward mode: Evaluate the derivative at: $X=\{3\}$ , $y=1$ , $W=3$

Variables	derivatives	Value of the variable	Value of the partial derivative	$\frac{d\mathcal{L}}{d\xi_n}$
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	-9	-3	-3
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$e^{-9}$	$e^{-9}$	$-3e^{-9}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$1+e^{-9}$	1	$-3e^{-9}$
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{1}{1 + e^{-9}}$	$\left(\frac{1}{1 + e^{-9}}\right)^2$	$-3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)^2$
$\xi_5 = \log \xi_4 = \log p = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	$\log \frac{1}{1 + e^{-9}}$	$1 + e^{-9}$	$-3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)$
$\mathcal{L}_i^A = -y \xi_5$	$\frac{\partial \mathcal{L}}{\partial \xi_5} = -y$	$-\log \frac{1}{1 + e^{-9}}$	-1	$3e^{-9} \left(\frac{1}{1 + e^{-9}}\right)$
$\frac{\partial \mathcal{L}_i^A}{\partial W} = \frac{\partial \mathcal{L}_i}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$			-3	0.00037018372



# Backward mode: Evaluate the derivative at: $X=\{3\}$ , $y=1$ , $W=3$

Variables	derivatives	Value of the variable	Value of the partial derivative
$\xi_1 = -W^T X$	$\frac{\partial \xi_1}{\partial W} = -X$	-9	-3
$\xi_2 = e^{\xi_1} = e^{-W^T X}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$e^{-9}$	$e^{-9}$
$\xi_3 = 1 + \xi_2 = 1 + e^{-W^T X}$	$\frac{\partial \xi_3}{\partial \xi_2} = 1$	$1+e^{-9}$	1
$\xi_4 = \frac{1}{\xi_3} = \frac{1}{1 + e^{-W^T X}} = p$	$\frac{\partial \xi_4}{\partial \xi_3} = -\frac{1}{\xi_3^2}$	$\frac{1}{1 + e^{-9}}$	$\left(\frac{1}{1 + e^{-9}}\right)^2$
$\xi_5 = \log \xi_4 = \log p = \log \frac{1}{1 + e^{-W^T X}}$	$\frac{\partial \xi_5}{\partial \xi_4} = \frac{1}{\xi_4}$	$\log \frac{1}{1 + e^{-9}}$	$1 + e^{-9}$
$\mathcal{L}_i^A = -y\xi_5$	$\frac{\partial \mathcal{L}}{\partial \xi_5} = -y$	$-\log \frac{1}{1 + e^{-9}}$	-1
$\frac{\partial \mathcal{L}_i^A}{\partial W} = \frac{\partial \mathcal{L}_i}{\partial \xi_5} \frac{\partial \xi_5}{\partial \xi_4} \frac{\partial \xi_4}{\partial \xi_3} \frac{\partial \xi_3}{\partial \xi_2} \frac{\partial \xi_2}{\partial \xi_1} \frac{\partial \xi_1}{\partial W}$			Type equation here.

Store all these values