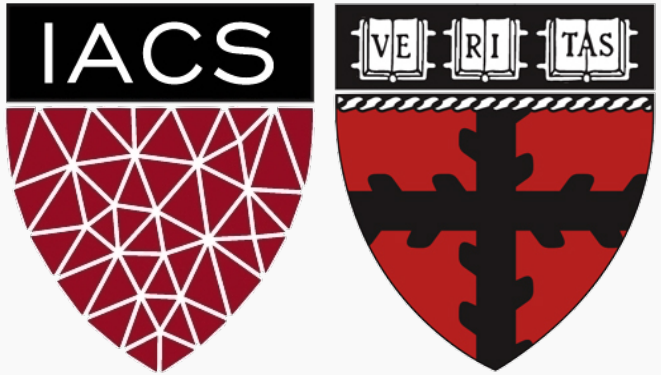
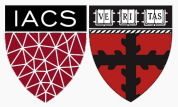
Lecture 1: Perceptron and Back Propagation

Pavlos Protopapas
Institute for Applied Computational Science
Harvard



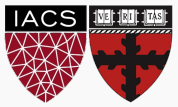
Outline

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



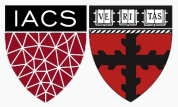
Schedule

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 2nd midnight, due on Nov 9th midnight
2. To be released Nov 9th midnight, due on Nov 16th 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.

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!

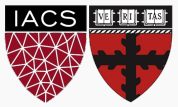
Policies

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.

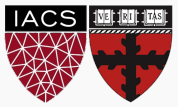
➔ Failing to cite the origins of code or techniques that you discover outside of the course's own lessons and integrate into your own work.

➔ 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.

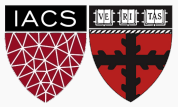
➔ Searching for or soliciting outright solutions to problem sets online or elsewhere.

➔ Splitting a problem set's workload with another individual and combining your work.



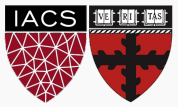
Outline

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



Artificial Neural Networks

Today's news

An AI just beat top lawyers at their own game

Share on F Share on +

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 consultati Share Quote ssors 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

Today's news

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

Share on F Share on +

IMAGE: BEN BRAIN/DIGITAL CAMERA MAGAZINE VIA GETTY IMAGES

BY MONICA CHIN

FEB 20 2018

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

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

AlphaGo (2015)

First program to beat a professional Go player

AlphaZero (2017)

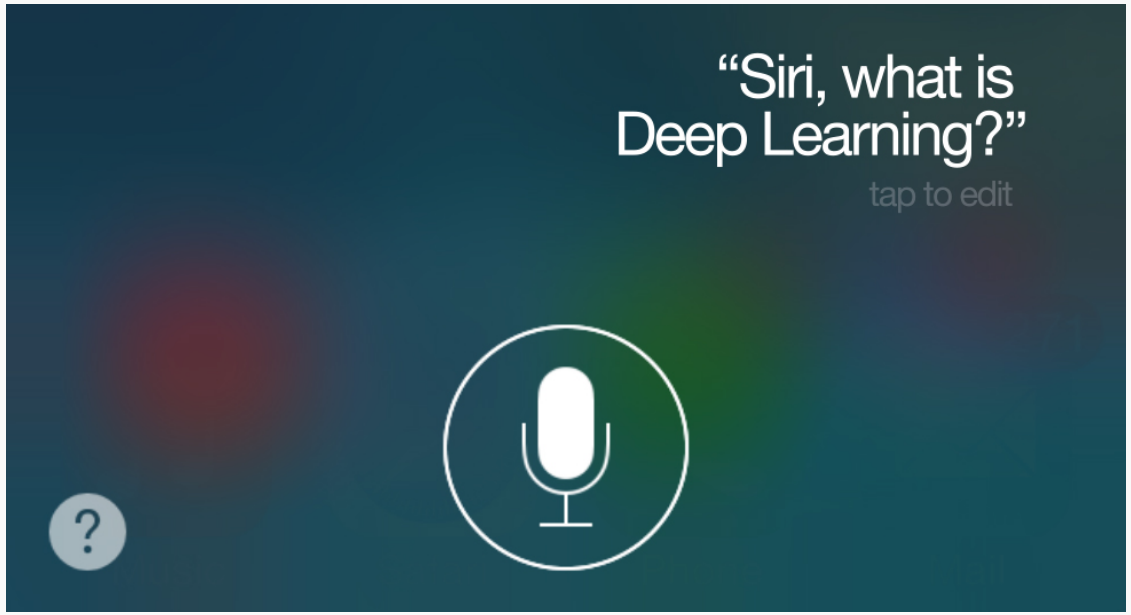
DeepMind

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

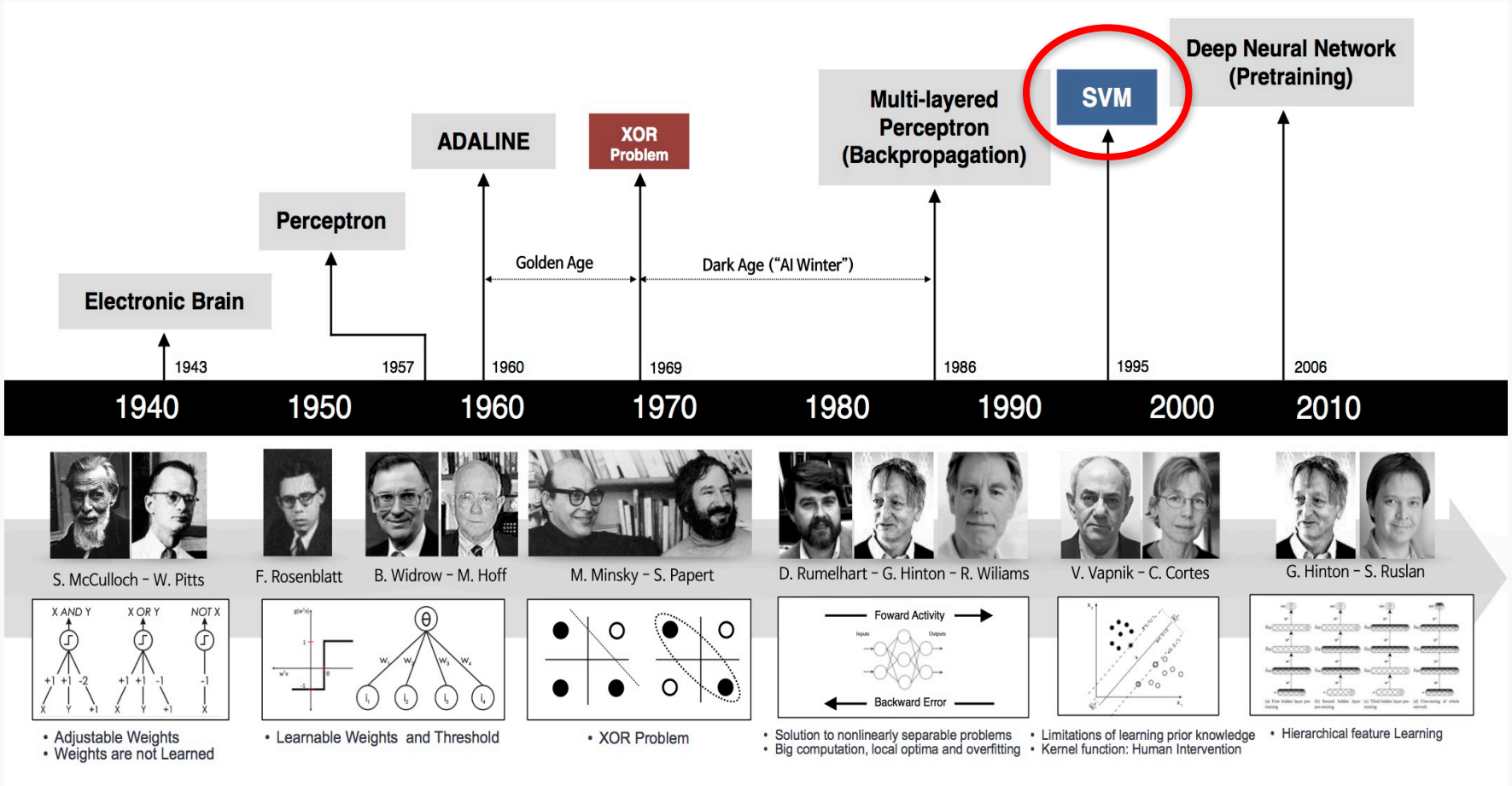
iOS Speech Synthesis (2016-)

Trained from 20 hours of high quality speech



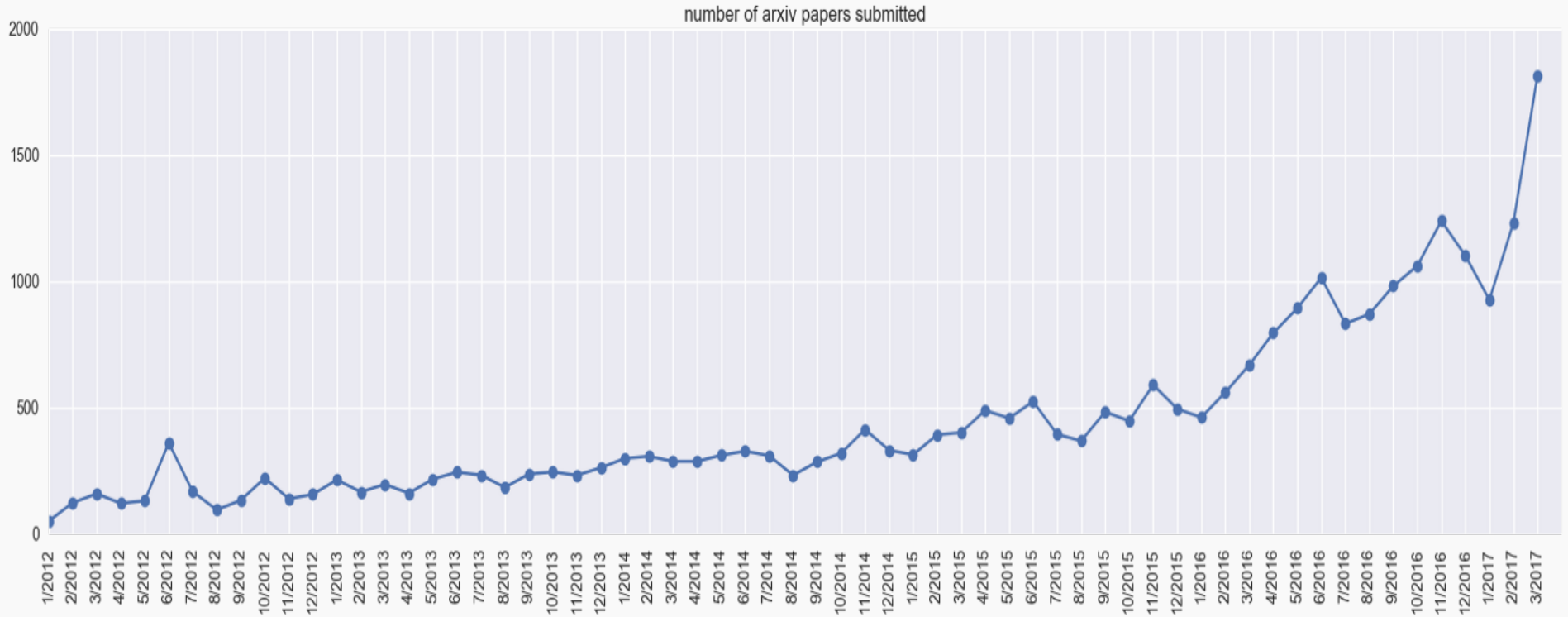
machinelearning.apple.com

Historical Trends



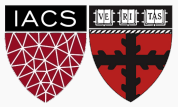
Historical Trends

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

X
predictors
features
covariates

Y
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

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	reversable	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

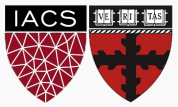
These data contain a binary outcome HD for 303 patients who presented with chest pain. An outcome value of:

- ***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.



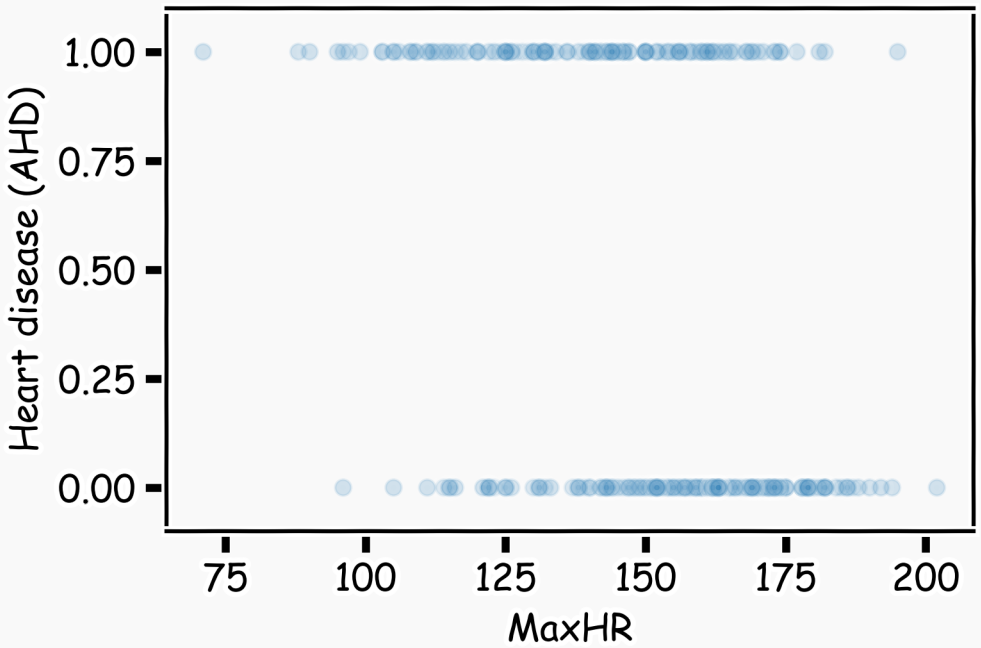
Logistic Regression

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

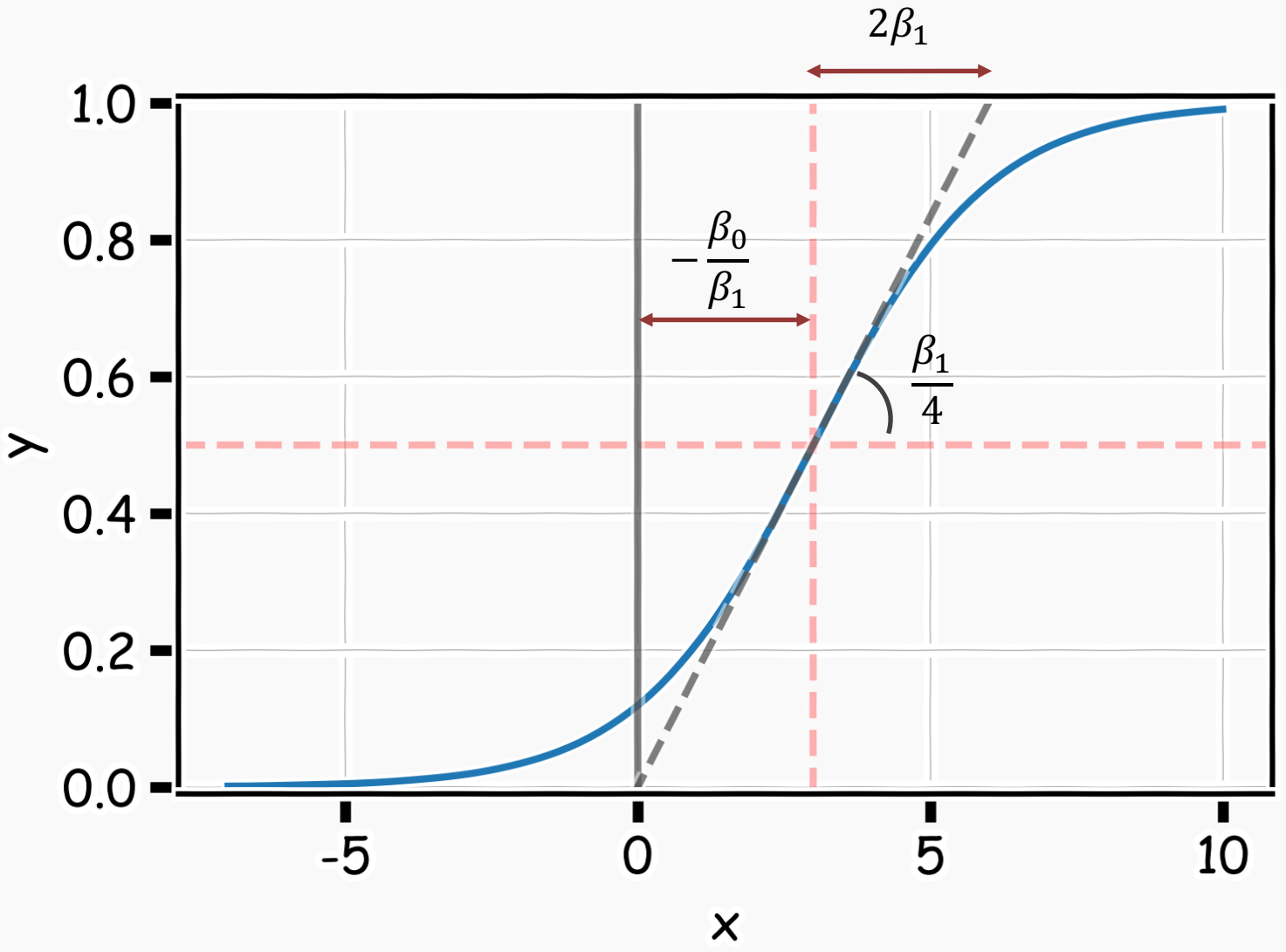
As a result the model will predict $P(y = 1)$ with an S-shaped curve, which is the general shape of the logistic function.

β_0 shifts the curve right or left by $c = -\frac{\beta_0}{\beta_1}$.

β_1 controls how steep the S-shaped curve is distance from ½ to ~1 or ½ to ~0 to ½ is $\frac{2}{\beta_1}$

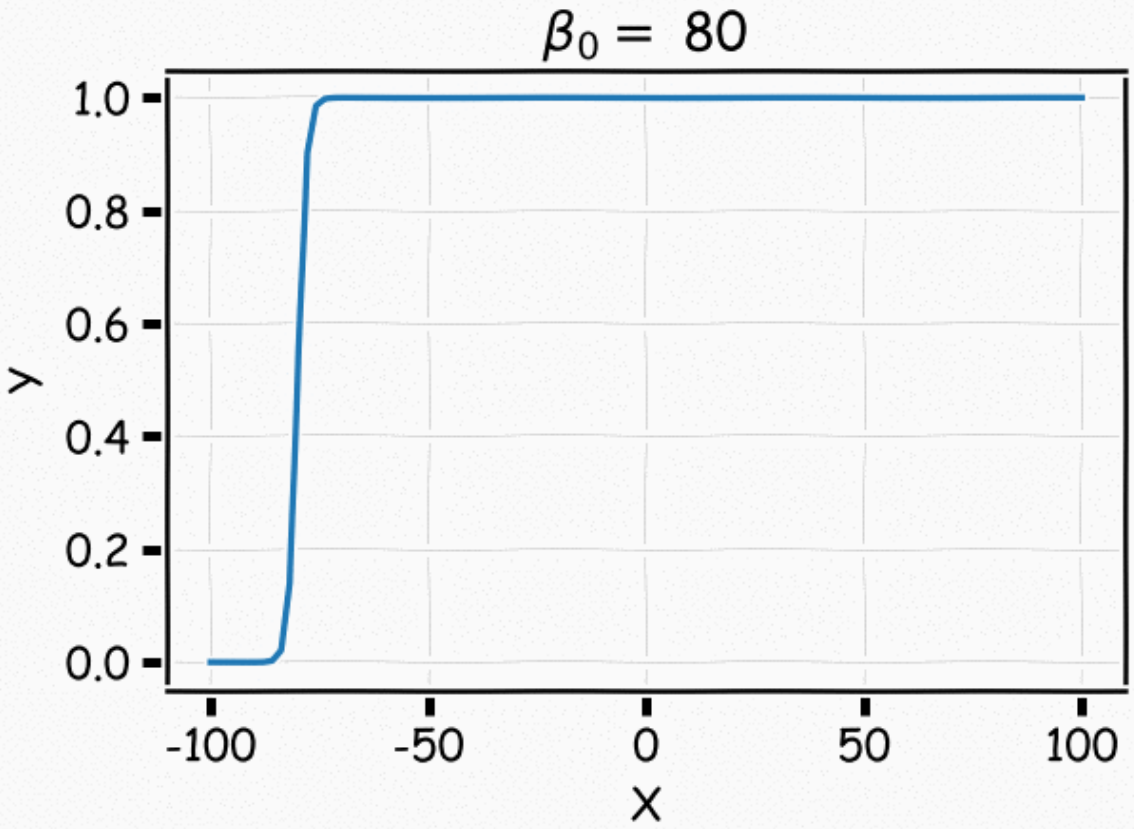
Note: if β_1 is positive, then the predicted $P(y = 1)$ goes from zero for small values of X to one for large values of X and if β_1 is negative, then has the $P(y = 1)$ opposite association.

Logistic Regression



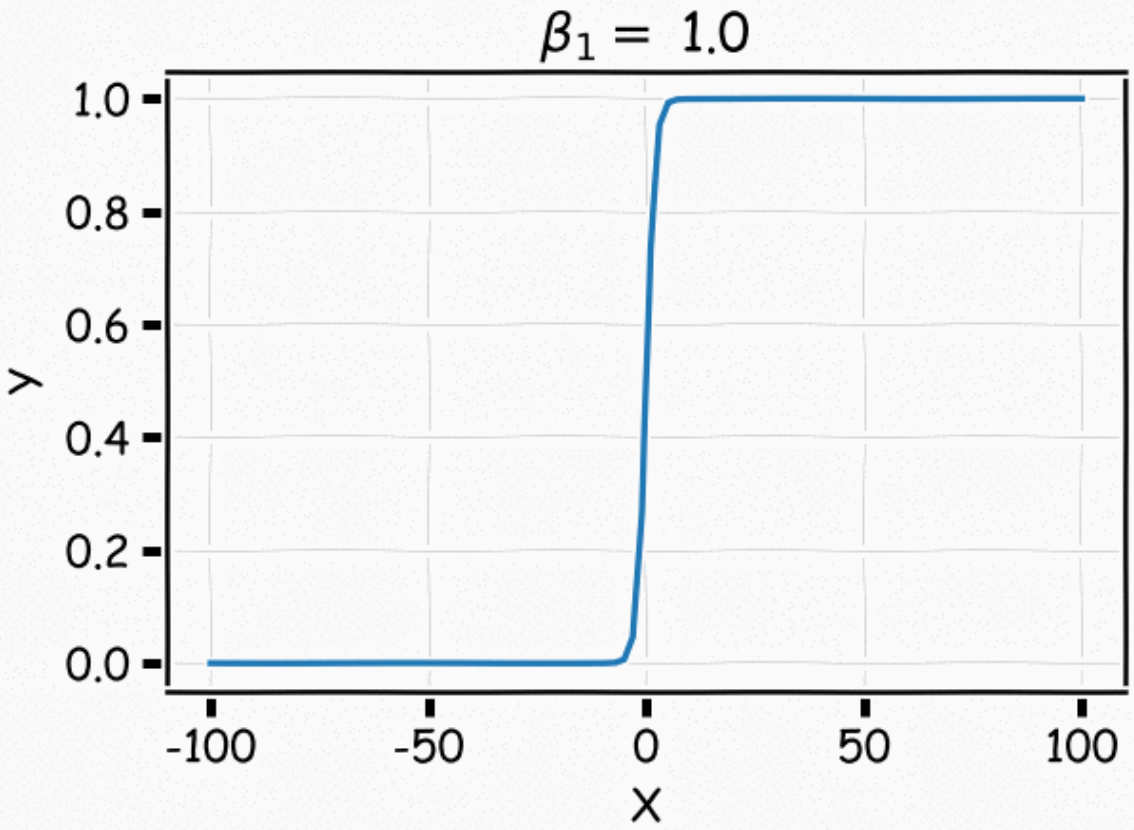
Logistic Regression

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



Logistic Regression

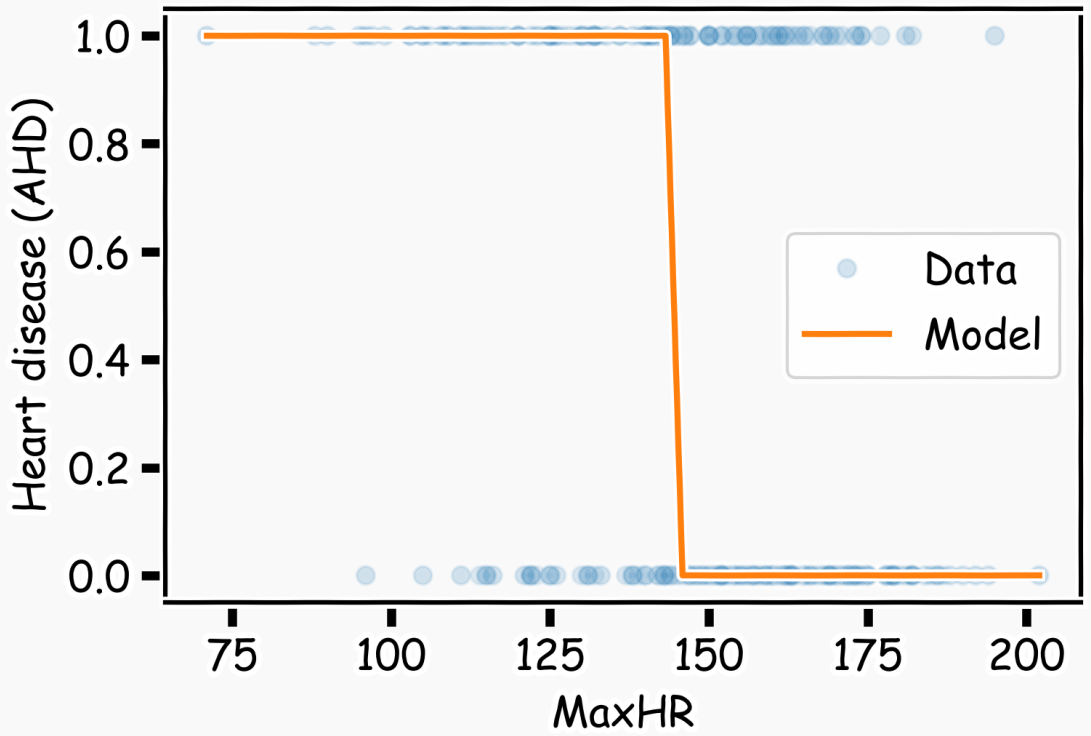
$$P(Y=1) = \frac{1}{1+e^{-(\beta_0+\beta_1 X)}}$$



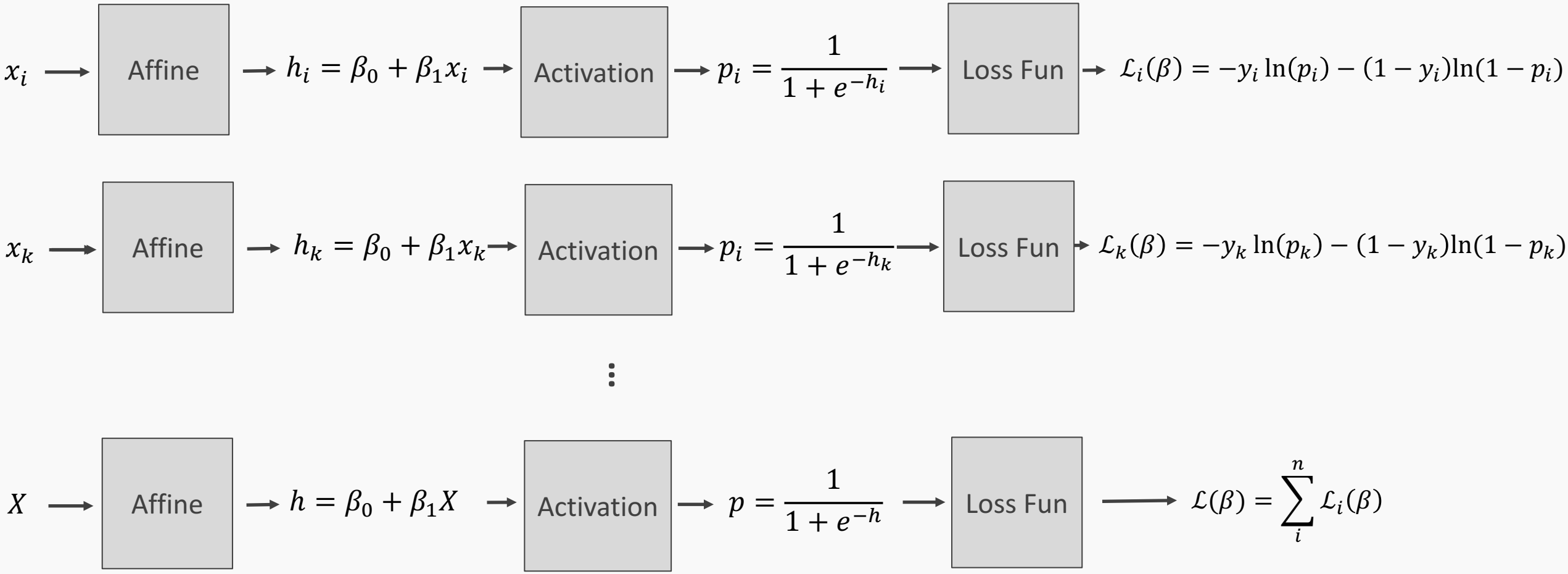
Estimating the coefficients for Logistic Regression

Find the coefficients that minimize the loss function

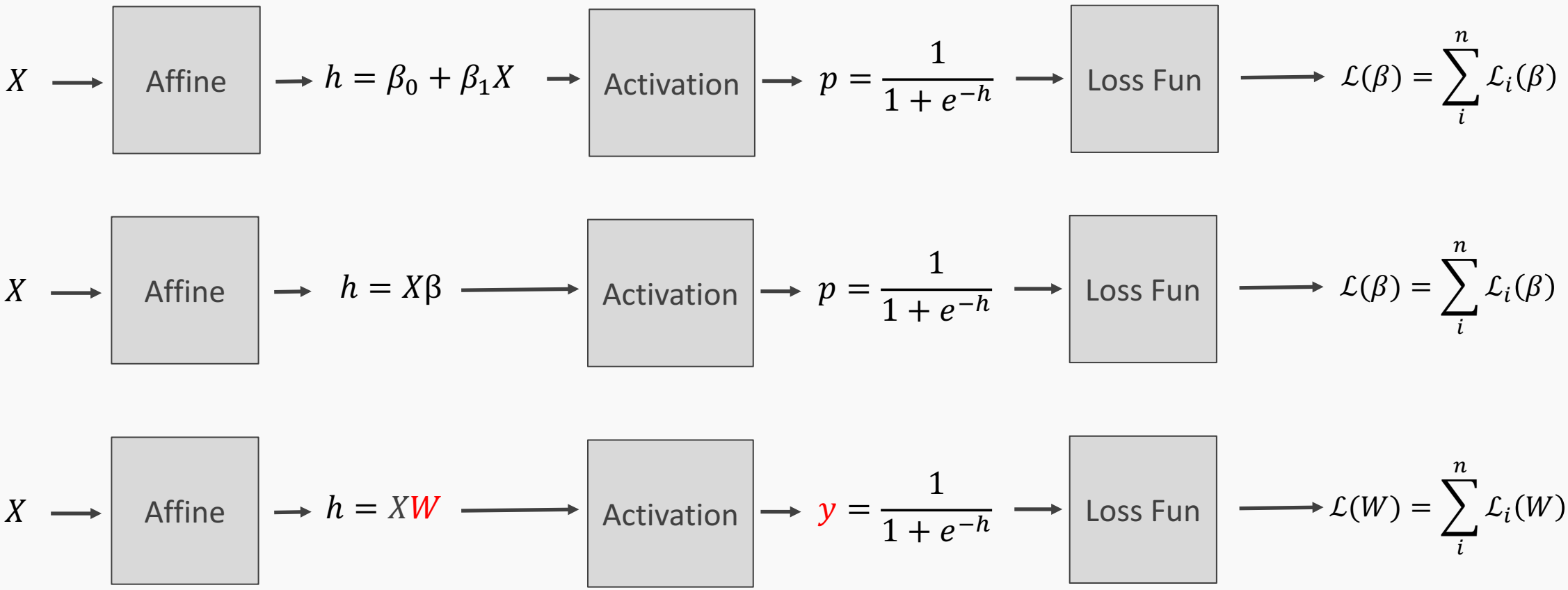
$$\mathcal{L}(\beta_0, \beta_1) = -\sum_i [y_i \log p_i + (1 - y_i) \log(1 - p_i)]$$



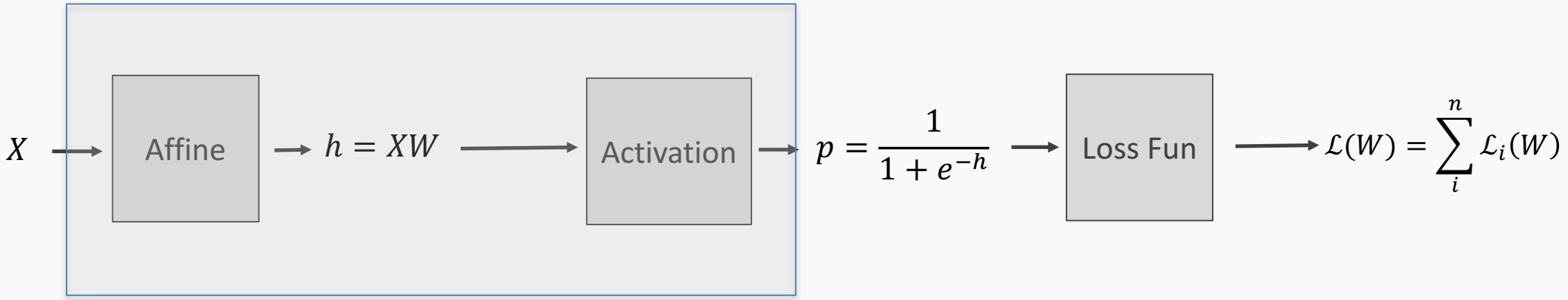
Logistic Regression Revisited

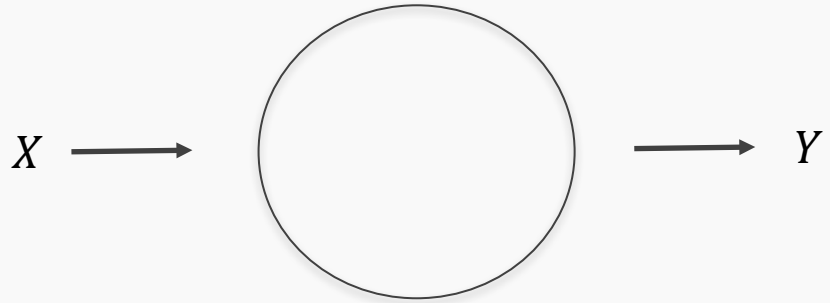


Build our first ANN



Build our first ANN



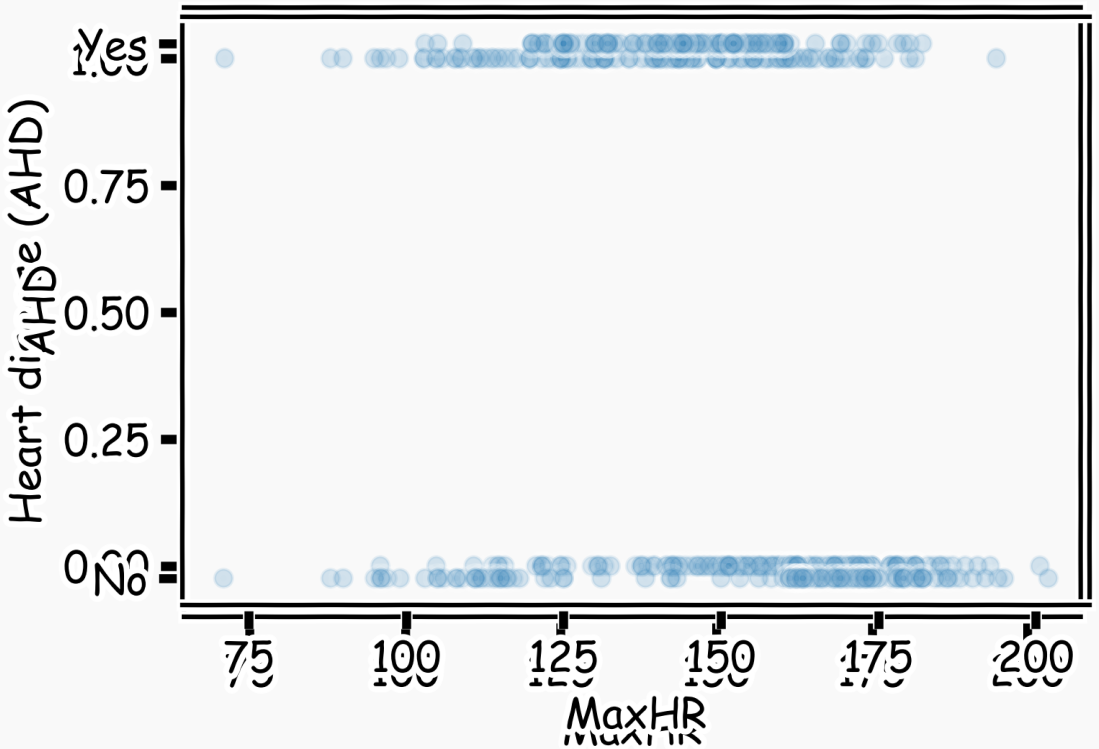


Single Neuron Network

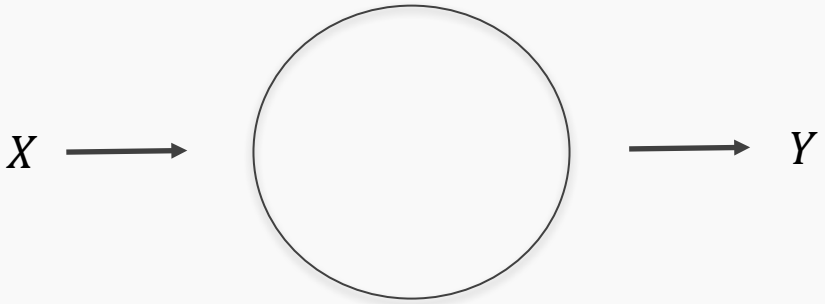
Very similar to Perceptron

Example Using Heart Data

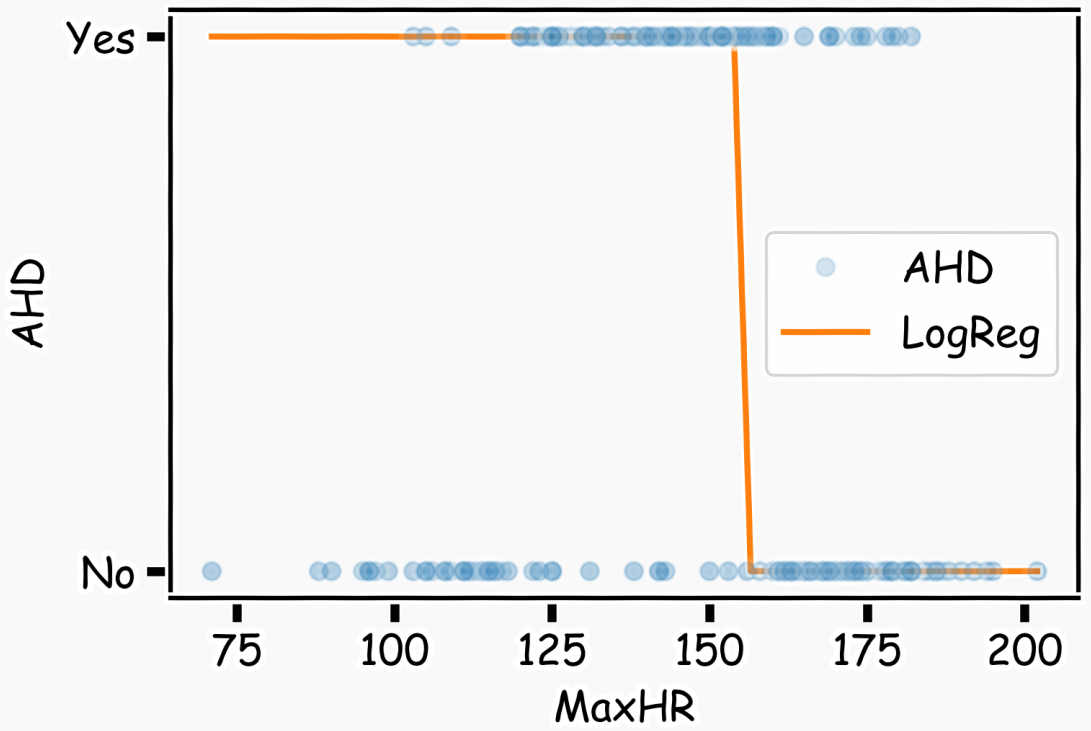
Slightly modified data to illustrate concepts.



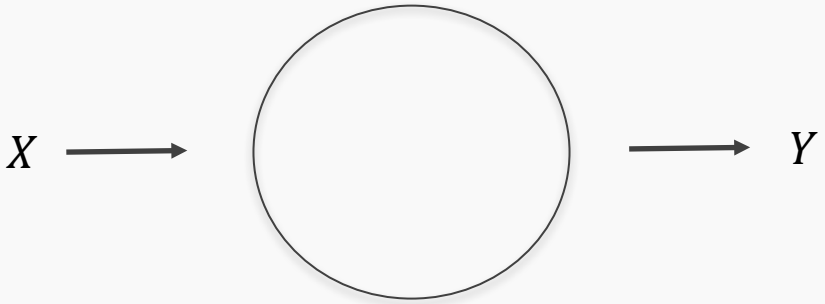
Example Using Heart Data



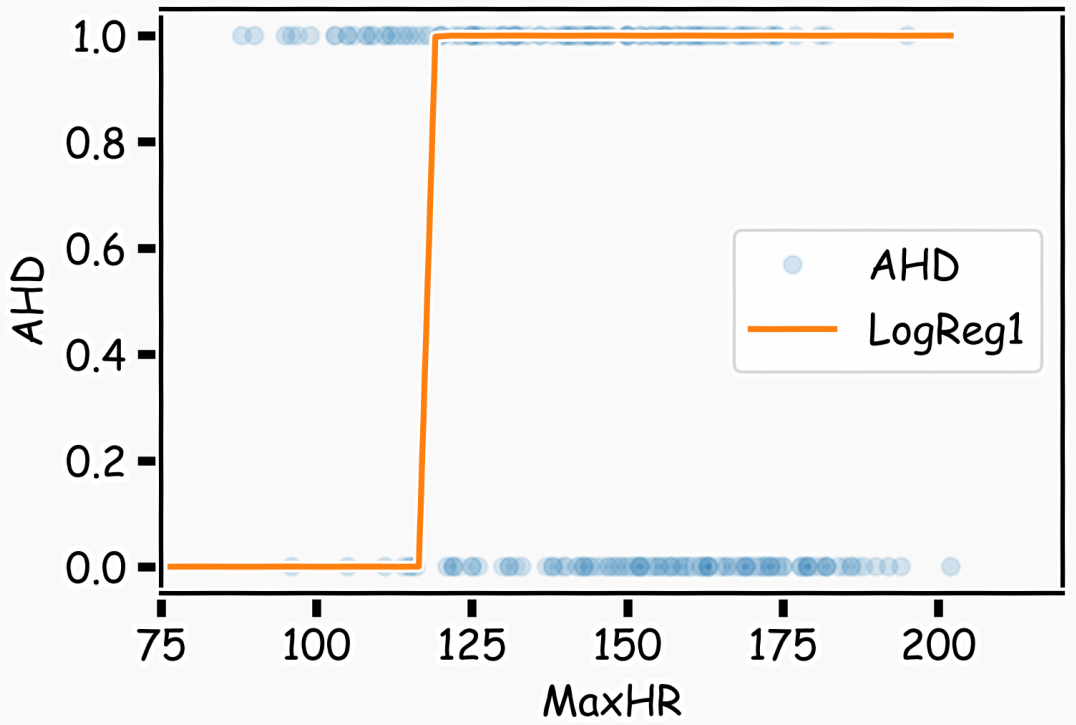
Choose W such as



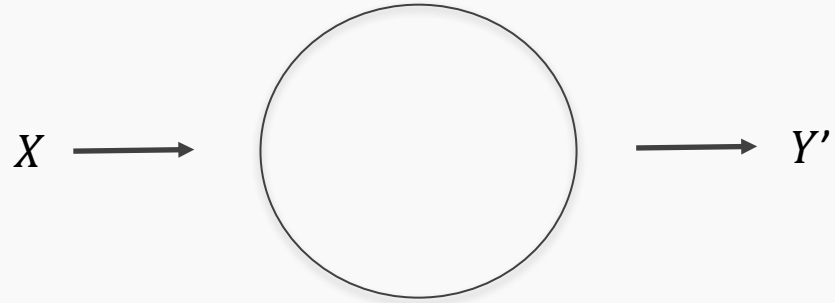
Example Using Heart Data

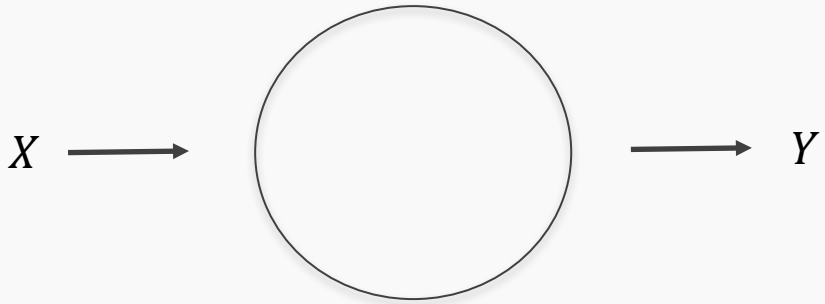


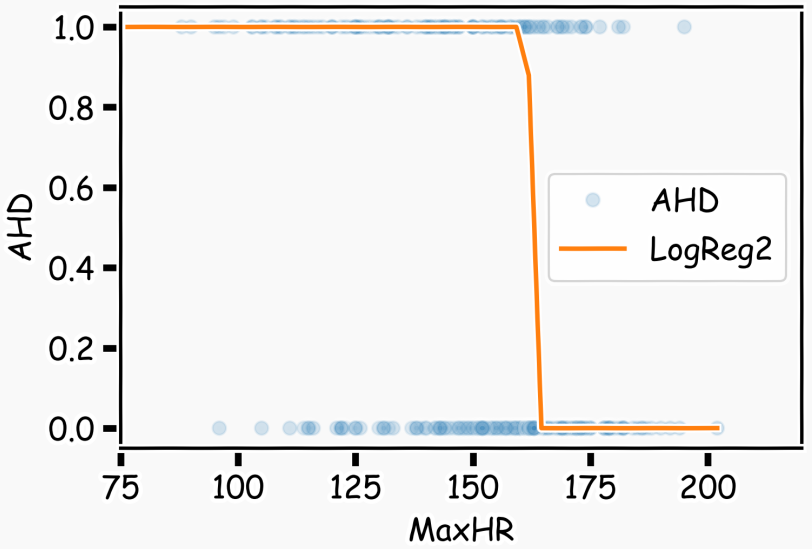
Choose W such as

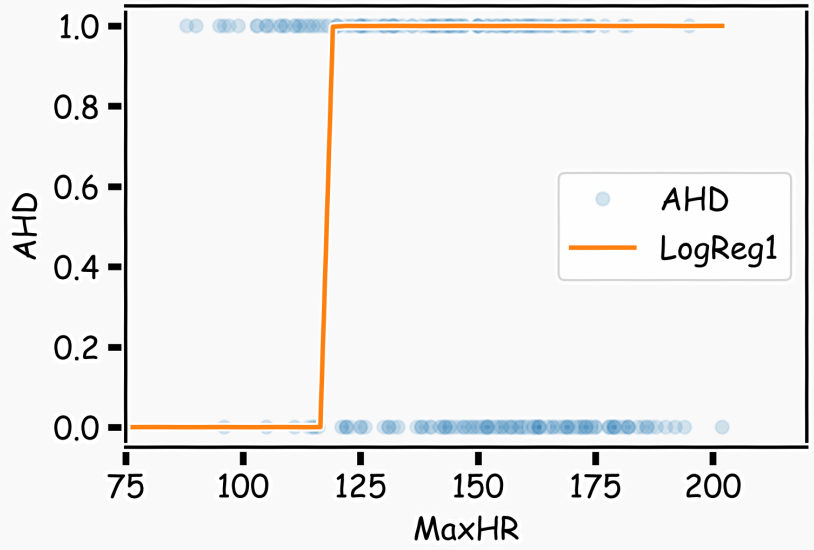


Example

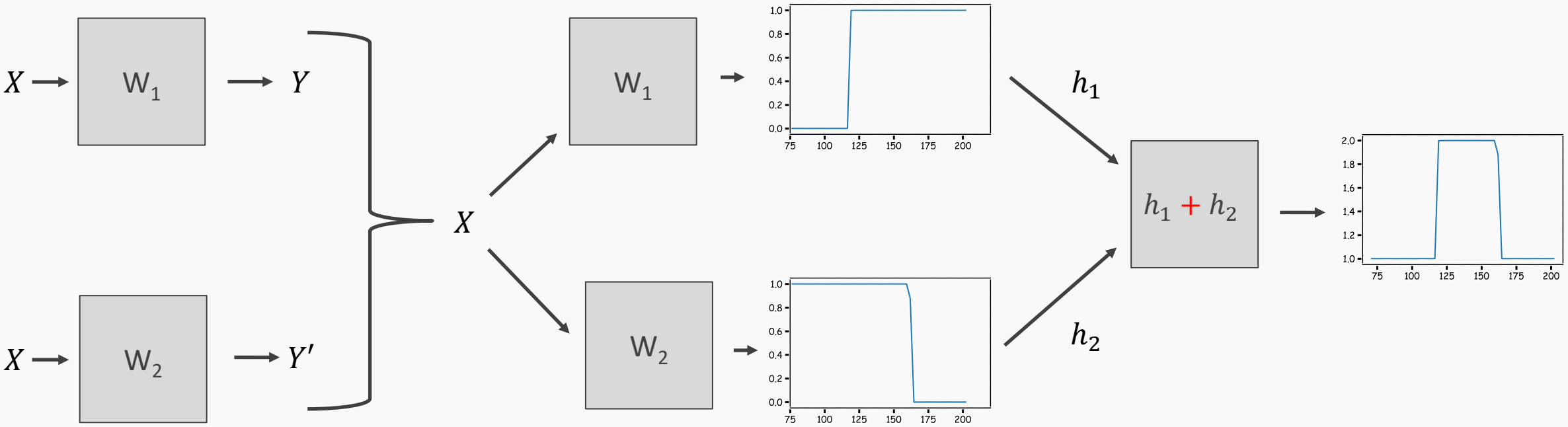

X
Y'


X
Y

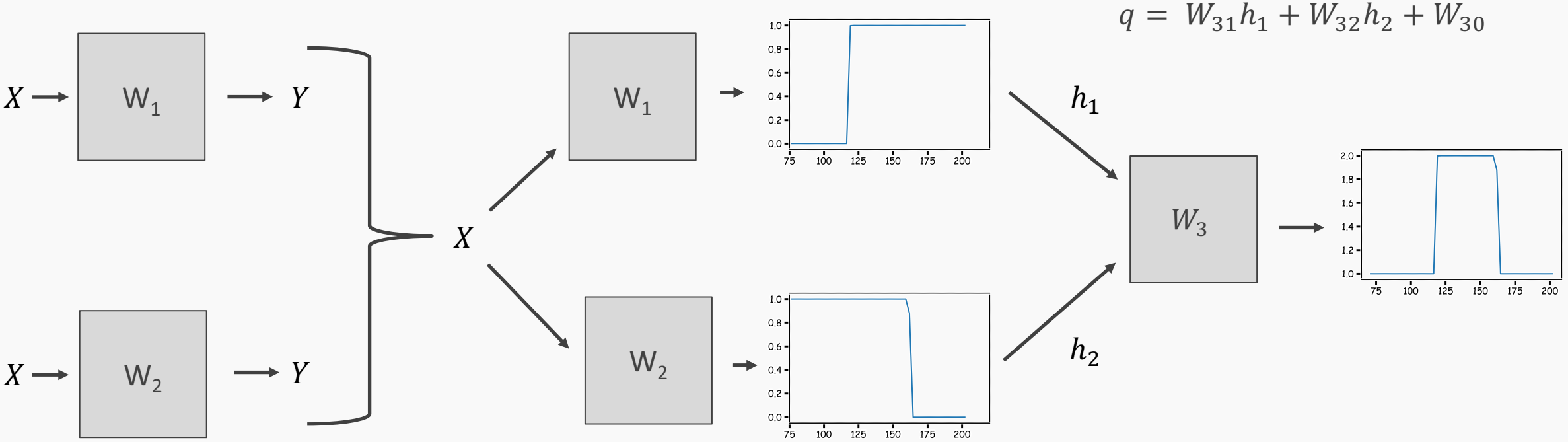

1.0
0.8
0.6
0.4
0.2
0.0
AHD
75
100
125
150
175
200
MaxHR
AHD
LogReg2


1.0
0.8
0.6
0.4
0.2
0.0
AHD
75
100
125
150
175
200
MaxHR
AHD
LogReg1

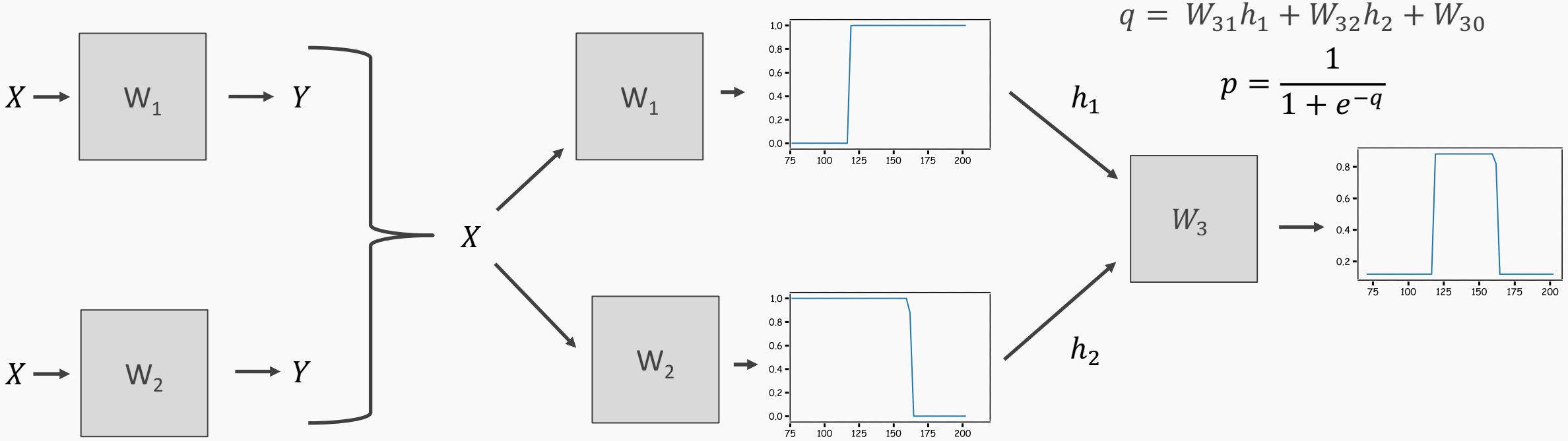
Pavlos game #232



Pavlos game #232



Pavlos game #232



$$q = W_{31}h_1 + W_{32}h_2 + W_{30}$$

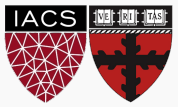
$$p = \frac{1}{1+e^{-q}}$$

$$L = -y\ln(\mathrm{p}) - (1-\mathrm{y})\ln(1-\mathrm{p})$$

Need to learn W1, W2 and W3.

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

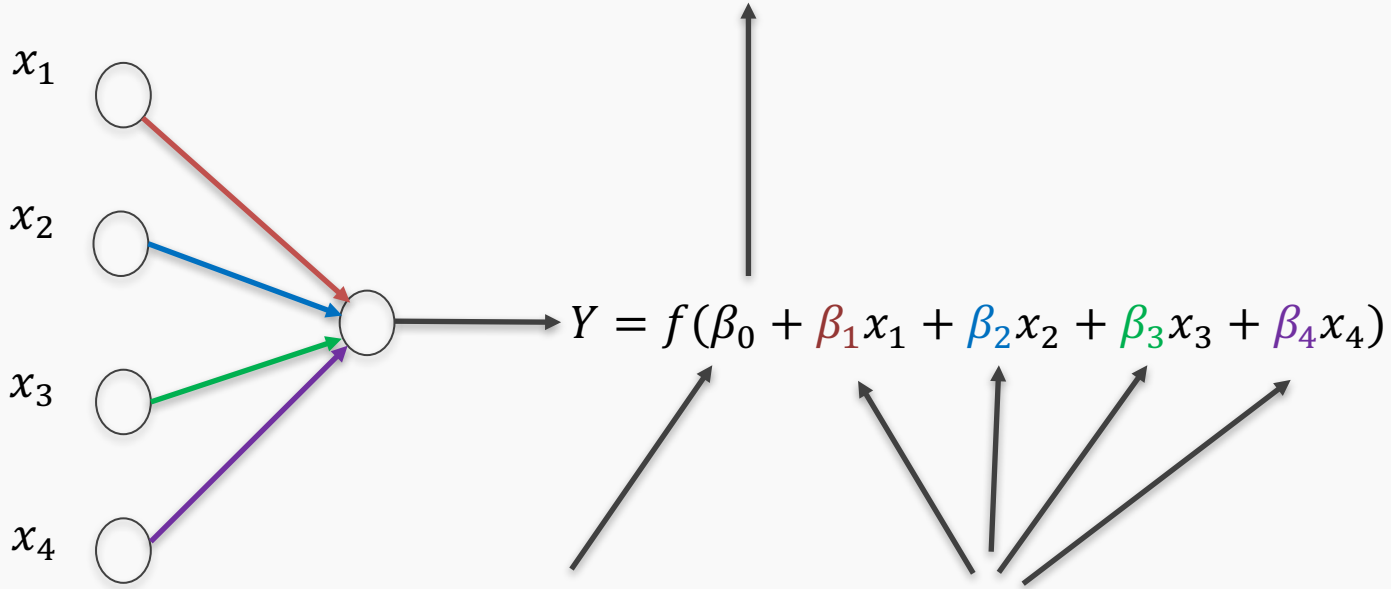
Classification

$f(X) = \frac{1}{1+e^{-W^T X}}$

Regression

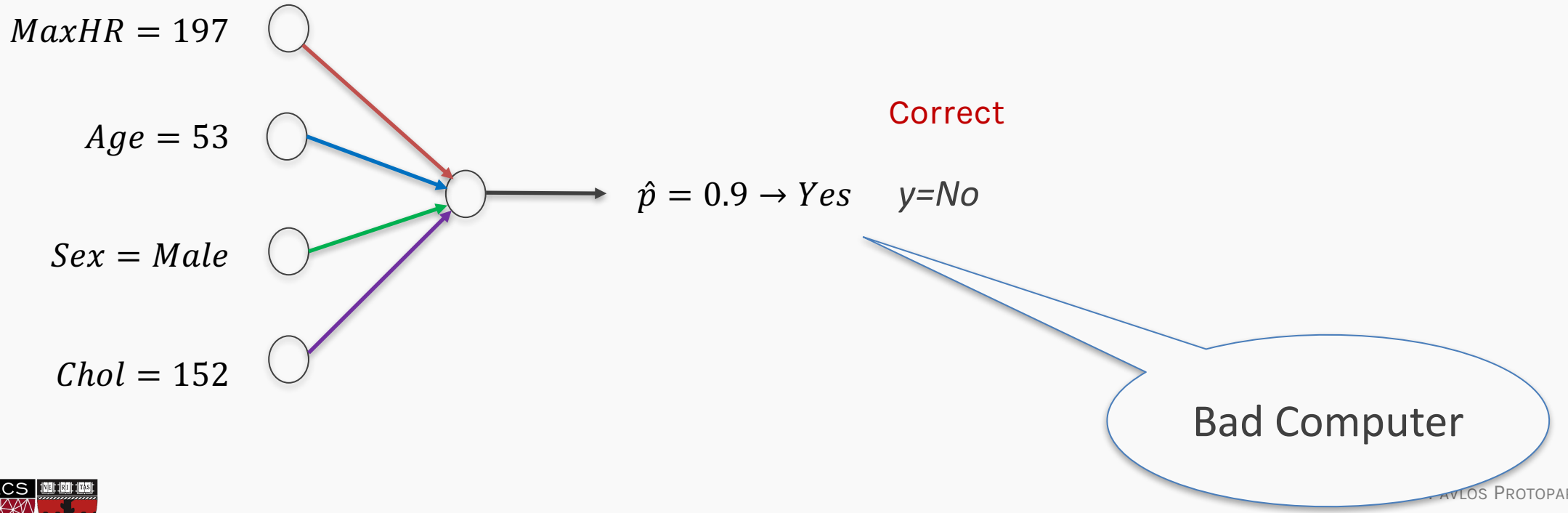
$f(X) = W^T X$

$W^T = [W_0, W_1, \ldots, W_4]$
$= [\beta_0, \beta_1, \ldots, \beta_4]$



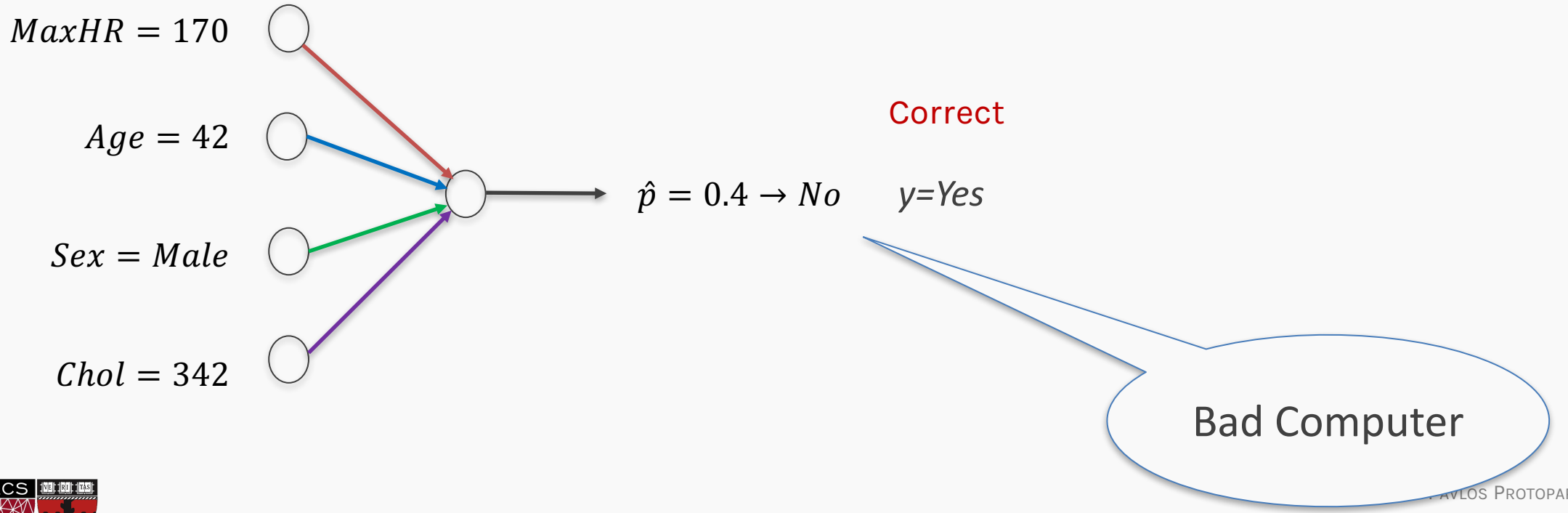
But what is the idea?

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?

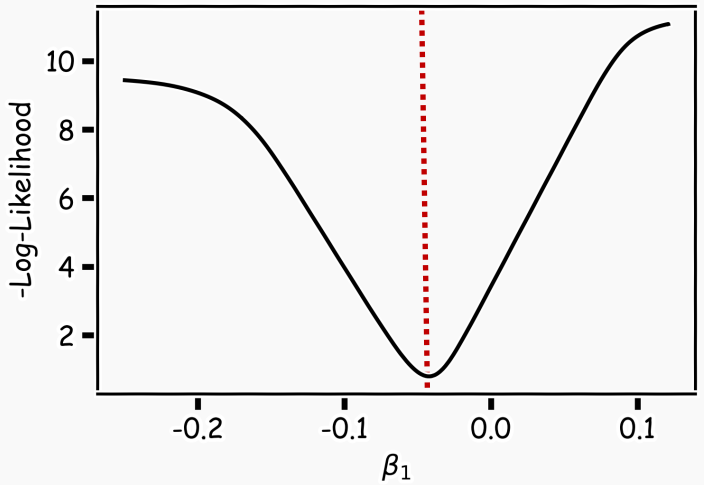
- **Loss Function:** Takes all of these results and averages them and tells us how bad or good the computer or those weights are.
- Telling the computer how **bad** or **good** is, does not help.
- You want to tell it how to change those weights so it gets better.

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 minimul $\mathcal{L}(W)$

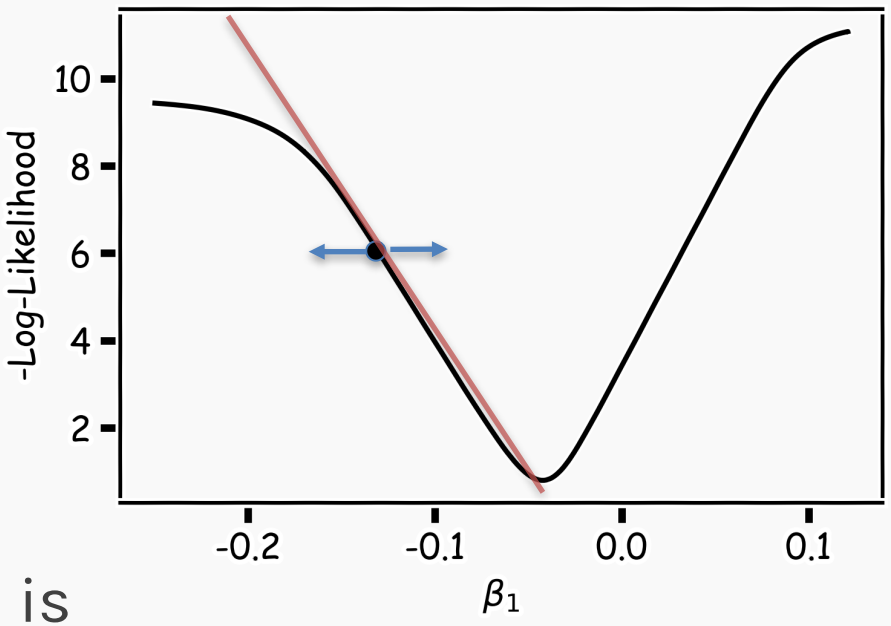


To find the optimal point of a function $\mathcal{L}(W)$

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

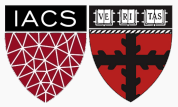
And find the W that satisfies that equation. Sometimes there is no explicit solution for that.

Minimizing the Loss function



A more flexible method is

- Start from any point
  - Determine which direction to go to reduce the loss direction (left or right)
  - Specifically if you can calculate the slope of the function at this point
  - Shift to the right if slope is negative or shift to the left if slope is positive
- Repeat



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:

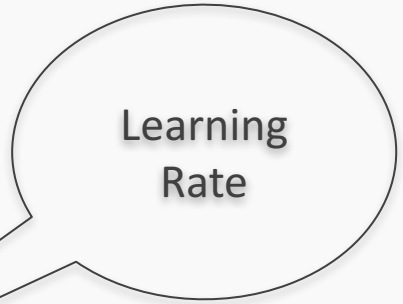
$$w^{new} = w^{old} + step$$

Opposite direction of the derivative means:

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

Change to more conventional notation:

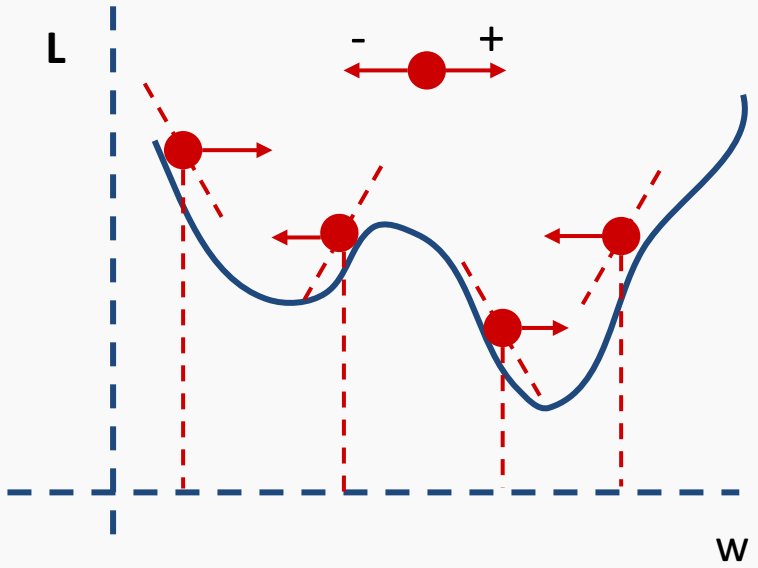
$$w^{(i+1)} = w^{(i)} - \lambda \frac{d\mathcal{L}}{dw}$$



Gradient Descent

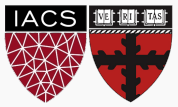
- Algorithm for optimization of first order to finding a minimum of a function.
- It is an iterative method.
- L is decreasing much faster in the direction of the negative derivative.
- The learning rate is controlled by the magnitude of λ.

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



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:

- $y = g(x) \qquad z = f(y) = f\big(g(x)\big)$

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

$$\boldsymbol{y} = g(\boldsymbol{x}) \qquad z = f(\boldsymbol{y}) = f\big(g(\boldsymbol{x})\big)$$

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

- For longer chains

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

Logistic Regression derivatives

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

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

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

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:

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

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

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

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

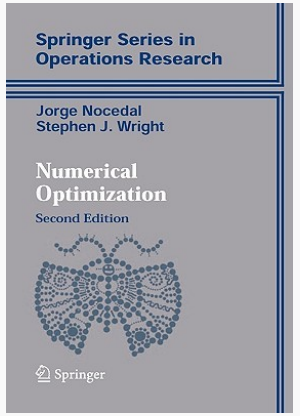
Variables	derivatives	Partial derivatives wrt to X,W
$\xi_1 = -W^TX$	$\frac{\partial \xi_1}{\partial W} = -X$	$\frac{\partial \xi_1}{\partial W} = -X$
$\xi_2 = e^{\xi_1} = e^{-W^TX}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{\xi_1}$	$\frac{\partial \xi_2}{\partial \xi_1} = e^{-W^TX}$
$\xi_3 = 1+\xi_2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

Learning Rate

Learning Rate

Trial and Error.

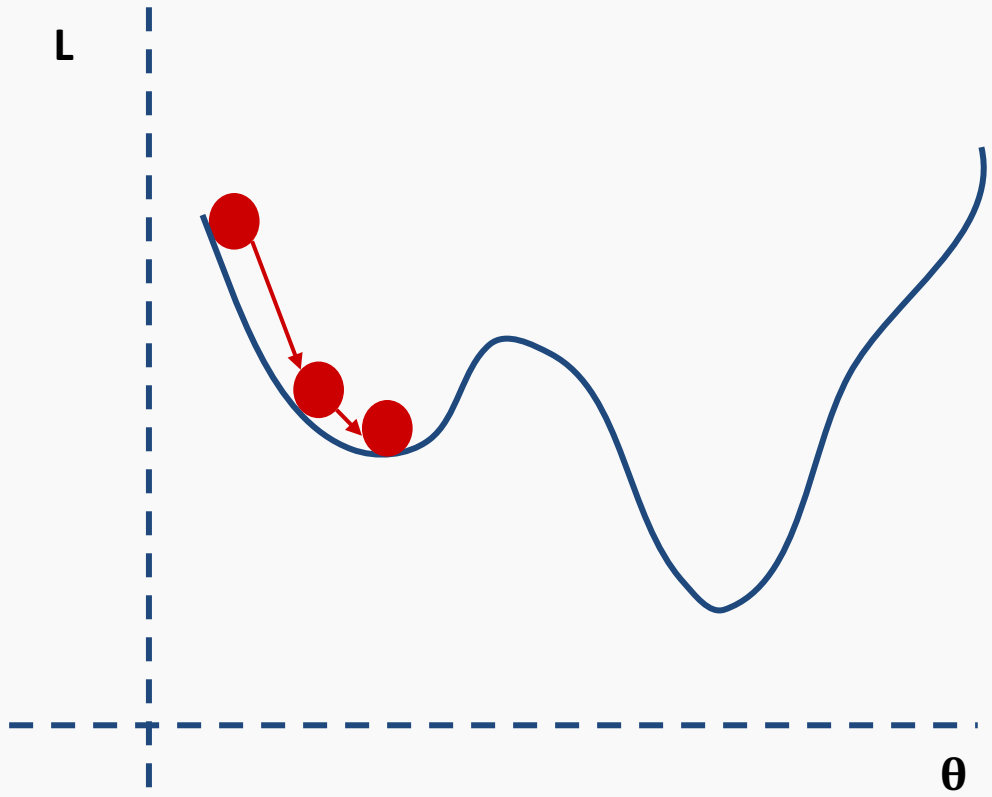
There are many alternative methods which address how to set or adjust the learning rate, using the derivative or second derivatives and or the momentum. To be discussed in the next lectures on NN.



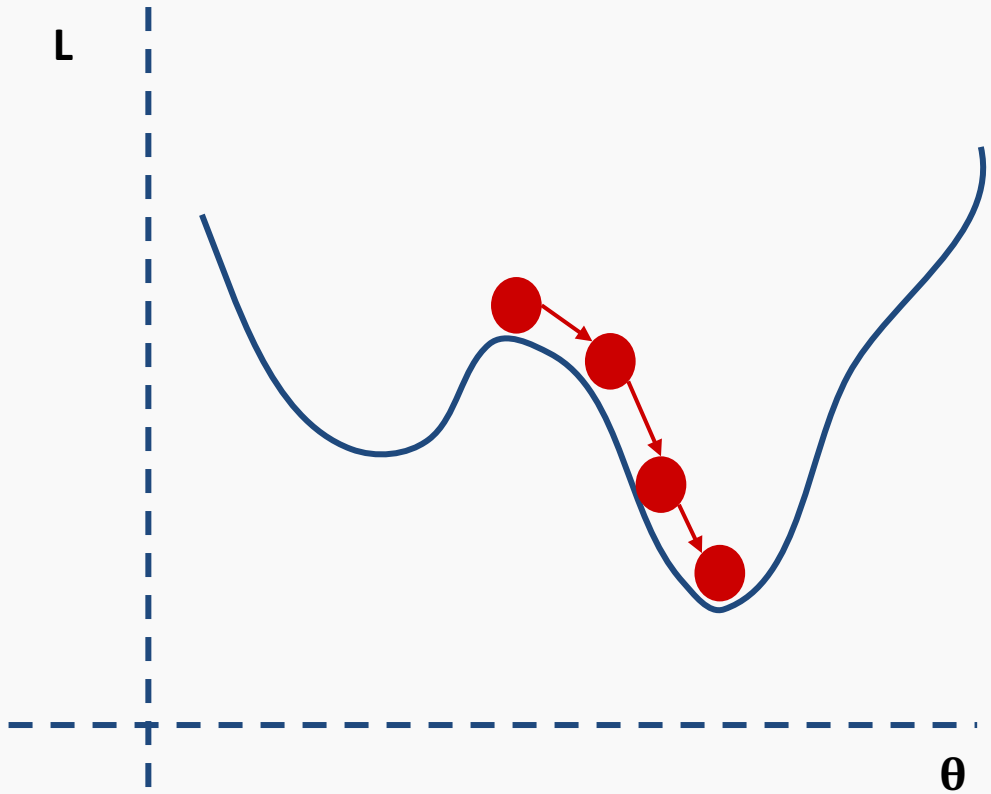
* J. Nocedal y S. Wright, "Numerical optimization", Springer, 1999 🔗
* *TLDR*: J. Bullinaria, "Learning with Momentum, Conjugate Gradient Learning", 2015 🔗

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

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

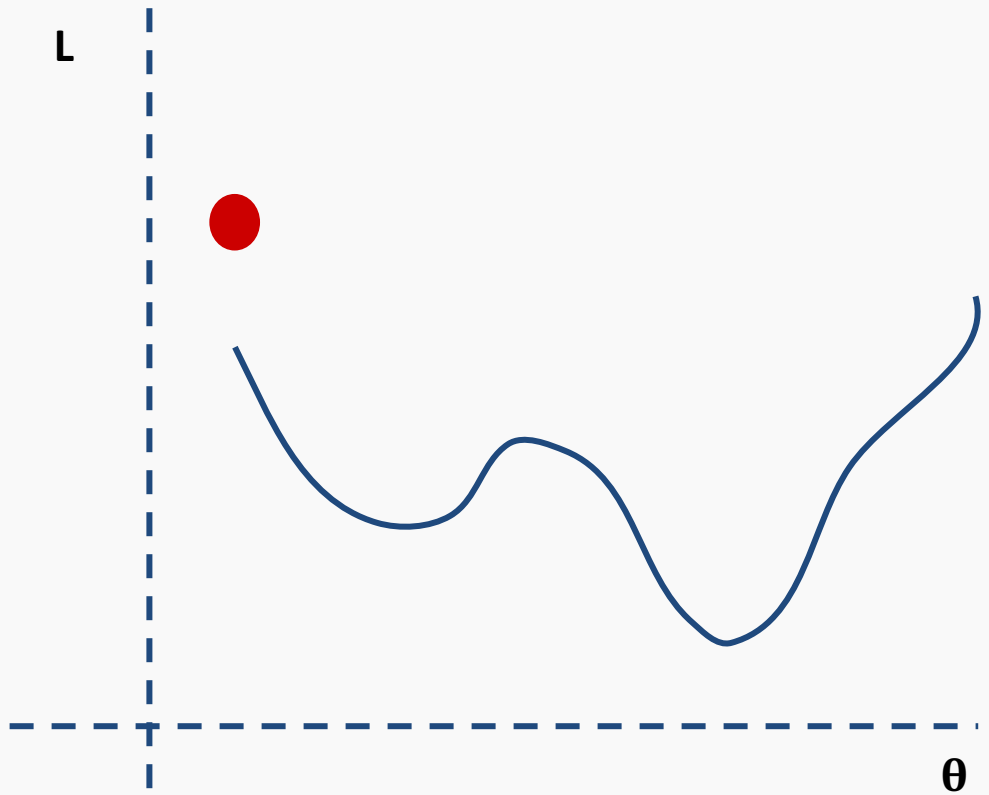
Instead of using all the examples for every step, use a subset of them (batch).

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.

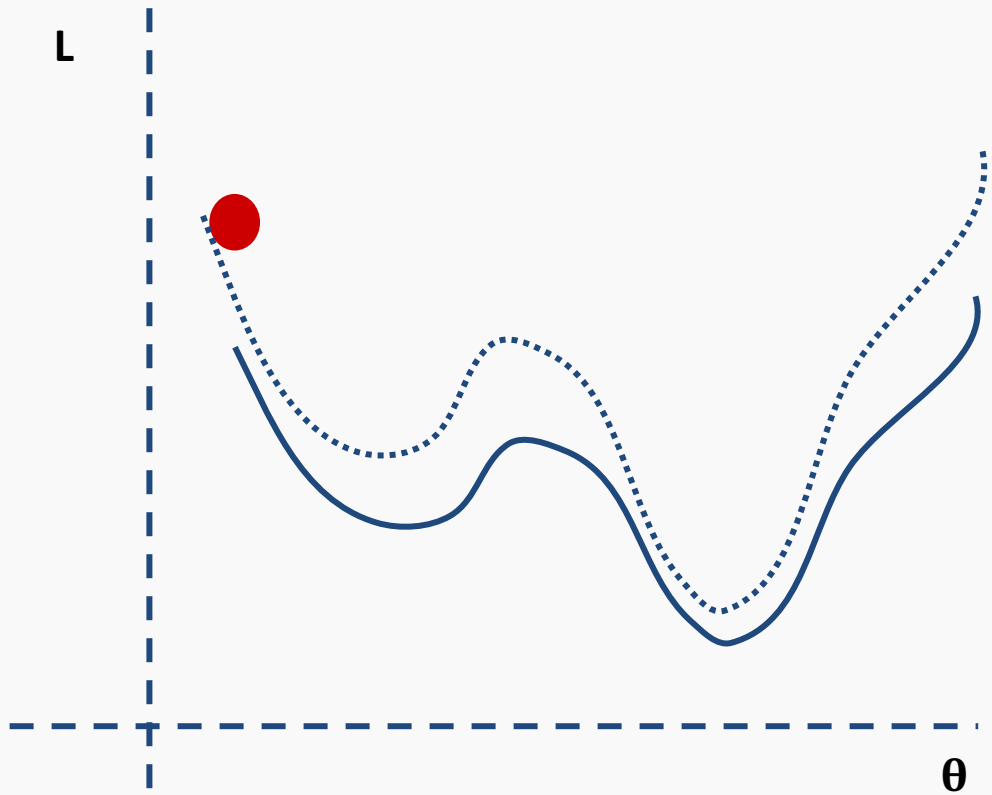
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

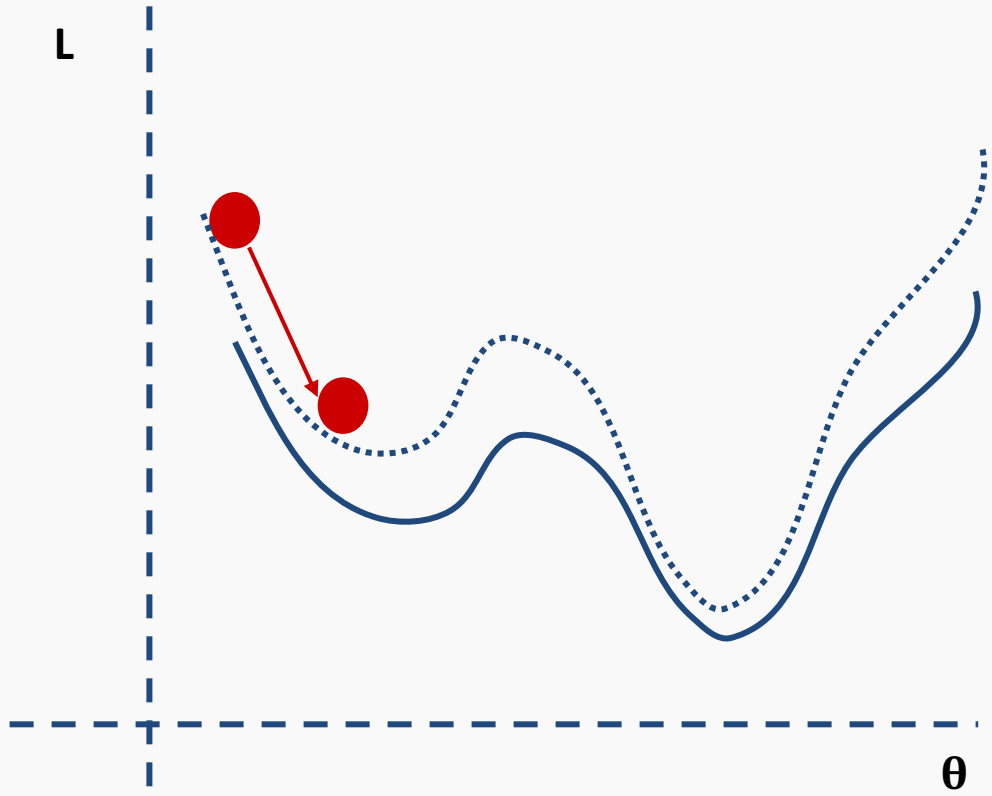
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

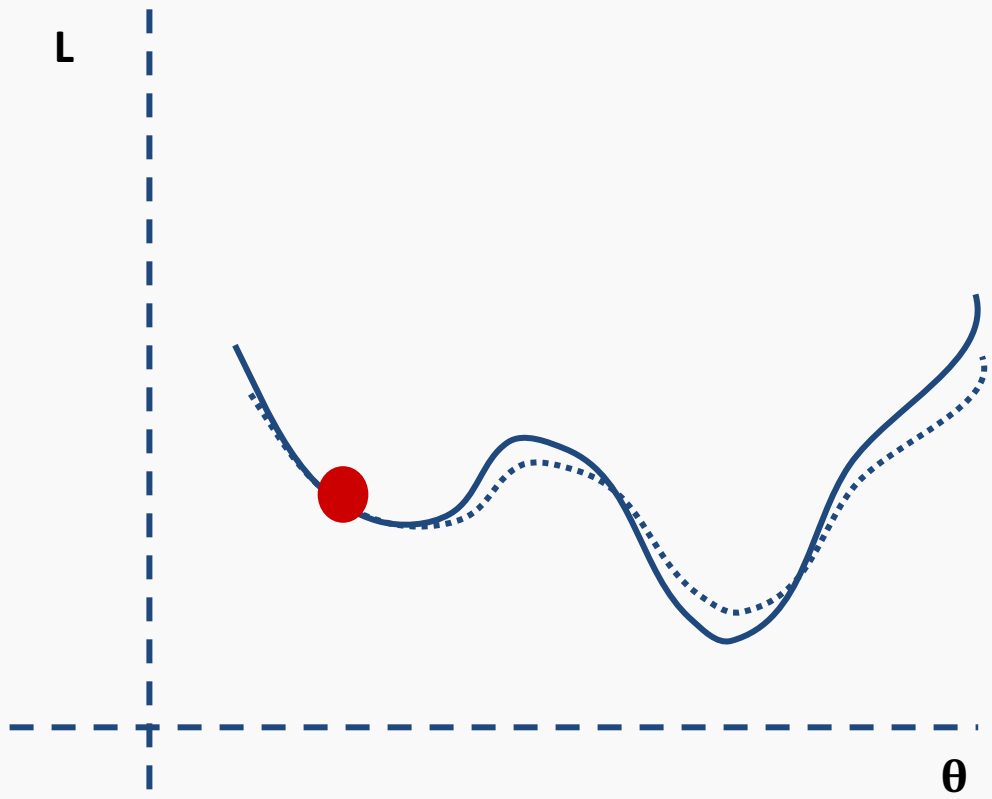
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

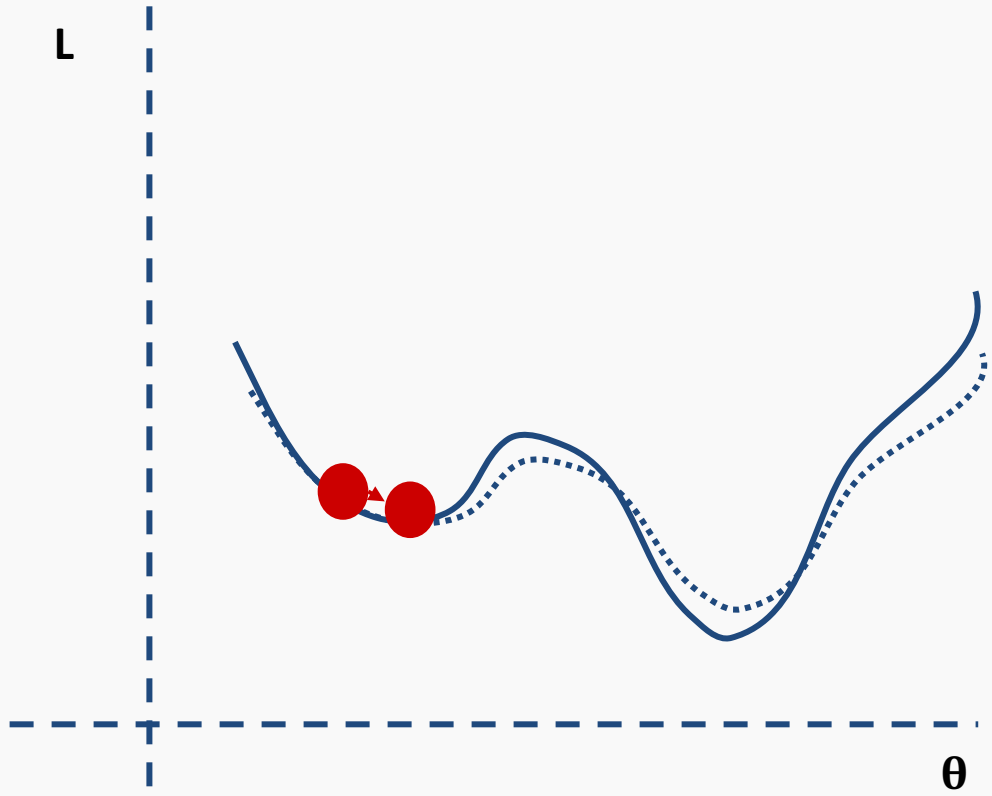
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

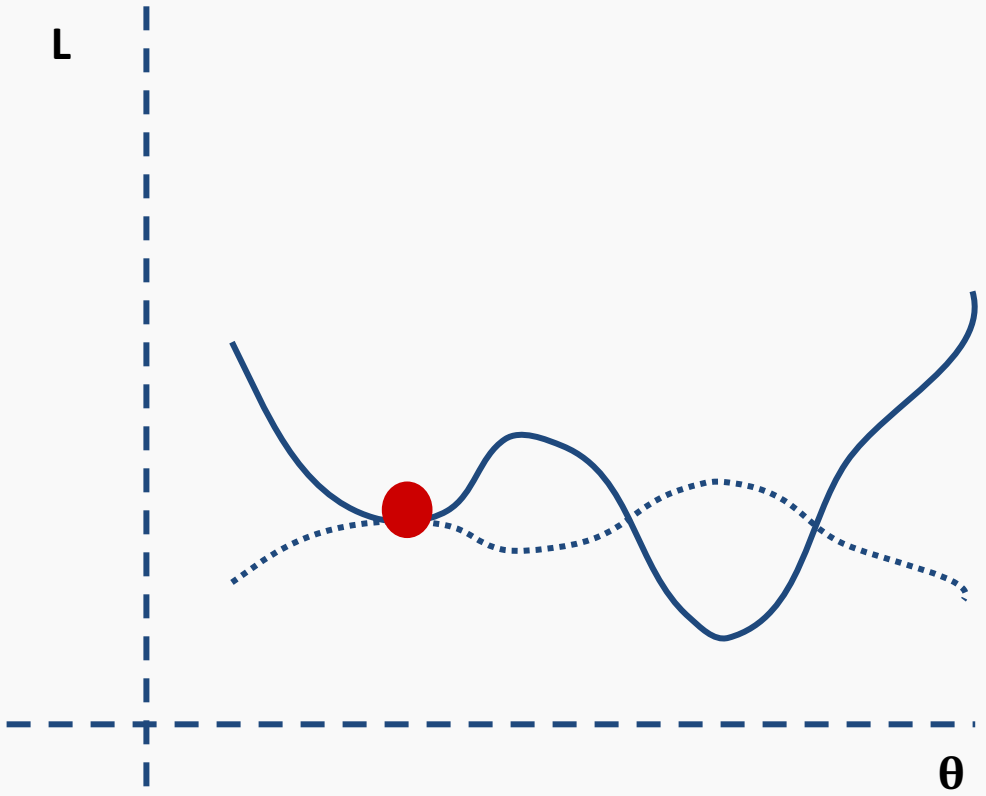
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

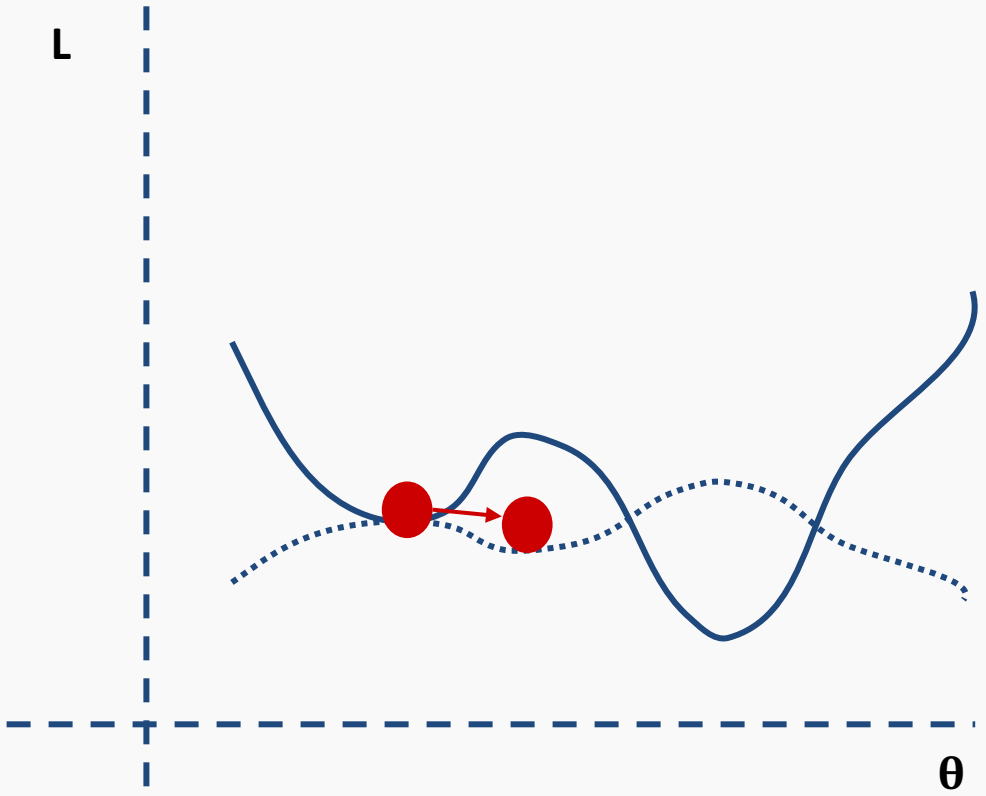
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

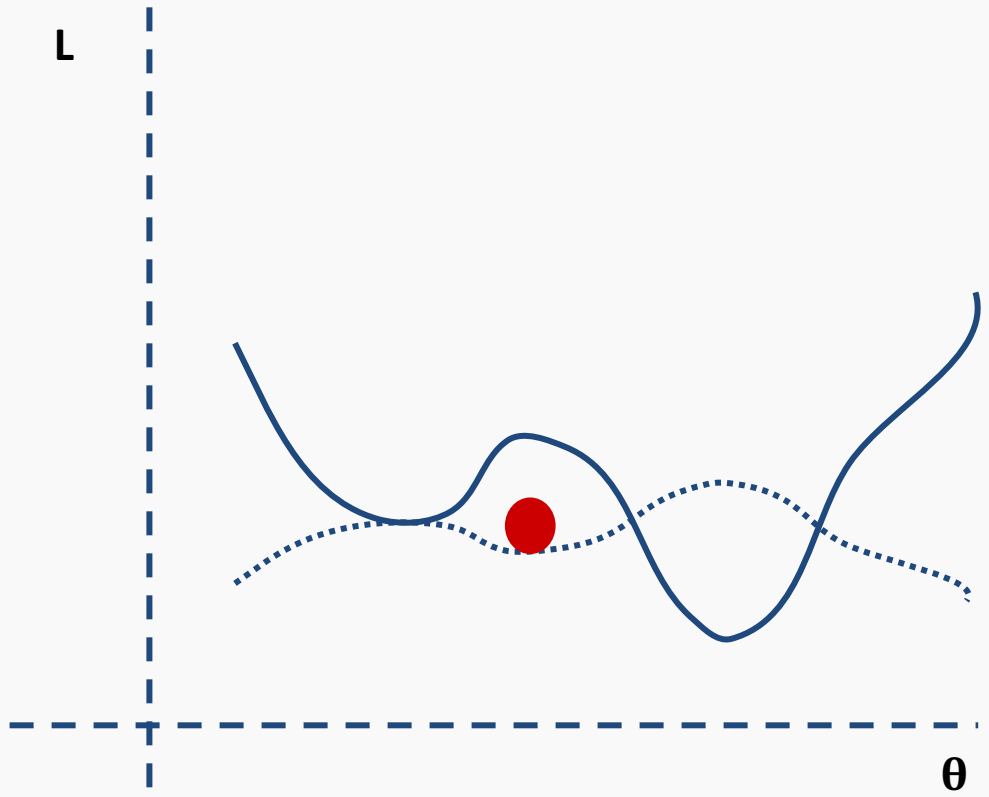
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

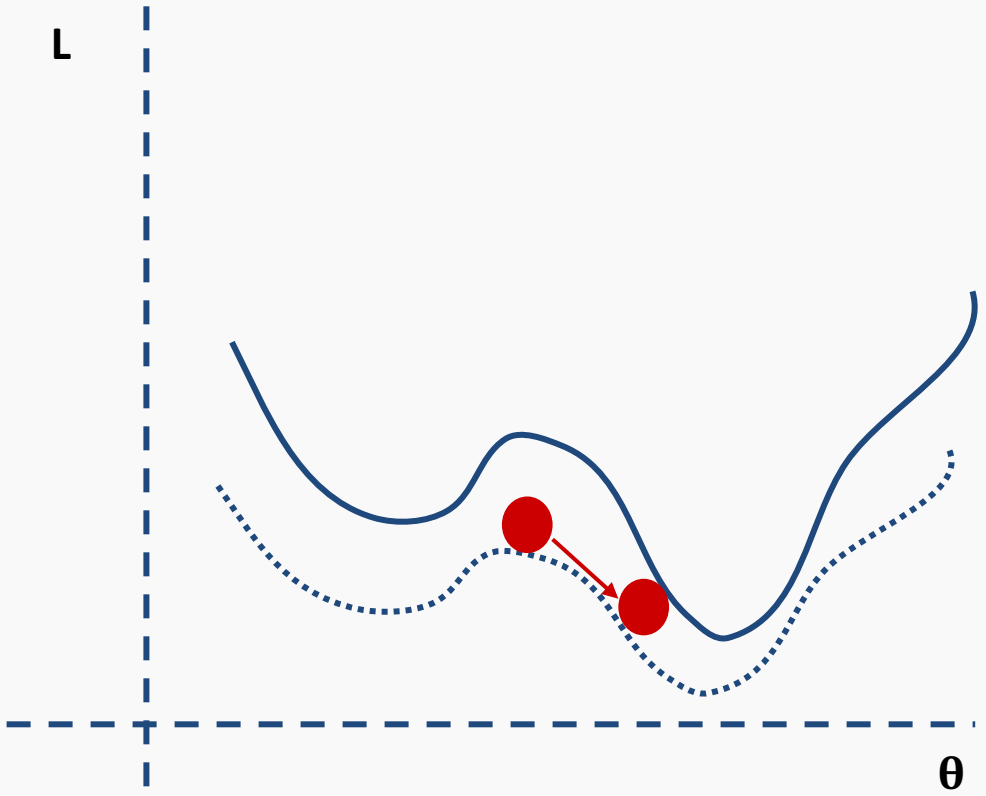
Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood:

Batch and Stochastic Gradient Descent



Full Likelihood:

Batch Likelihood: