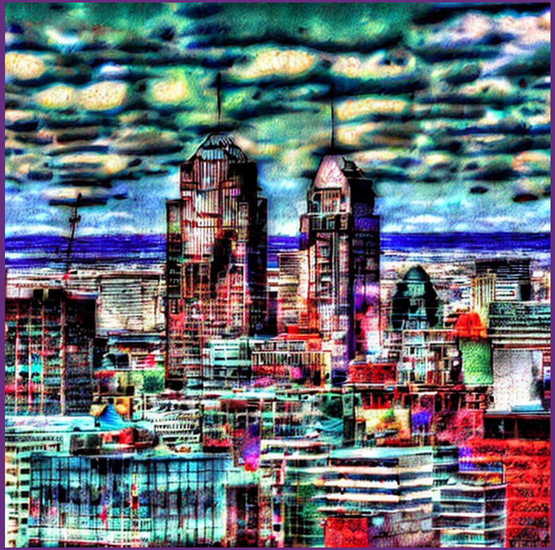


Mila

10-06-2024

Introduction to Machine Learning

Session 4

Prakhar Ganesh

About me

Prakhar Ganesh (he/him)

PhD candidate in Computer Science
at McGill University / Mila

Research in Responsible AI:
Fairness and Privacy in AI; Model Multiplicity

- Introduction to ML (session 4)
- Natural Language Processing (x2)
- Data Privacy

Mila

Before we start …

Ask questions anytime!

Contact Prakhar: prakhar.ganesh@mila.quebec

Before we start …

Bootcamp Week 2 Pulse Check - How’s everyone doing?

Mila

Before we start …

Bootcamp Week 2 Pulse Check - How’s everyone doing?

Any questions from the last 3 “intro to ML” sessions you want to clarify before moving on?

Before we start ...

Bootcamp Week 2 Pulse Check - How’s everyone doing?

Any questions from the last 3 “intro to ML” sessions you want to clarify before moving on?

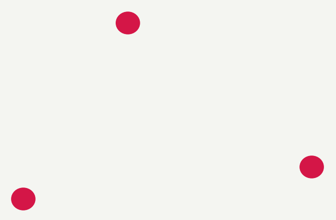
We’ll also do a recap on the go!

Some slides borrowed from Intro to ML sessions 1, 2, 3 by Mélisande and Niki

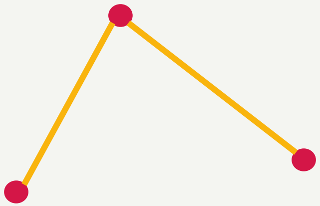
Goals today...

- Understand the importance of iterative learning.
- Derivatives
- Gradient Descent
- Why nonlinear models?
- Bringing it all together: We'll follow the training of a neural network from start to end!

Example: Draw a circle through 3 points



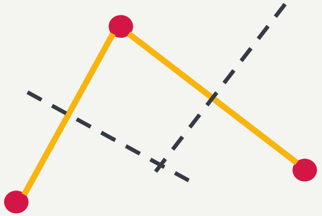
Example: Draw a circle through 3 points



Connect the points

Mila

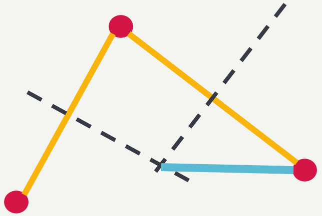
Example: Draw a circle through 3 points



Draw perpendiculars from the center

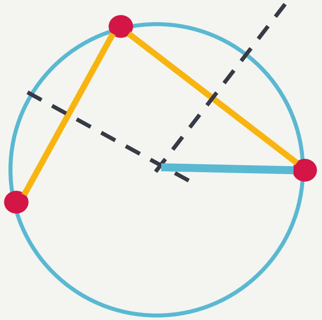
Mila

Example: Draw a circle through 3 points



Intersection is the center.
Distance is the radius.

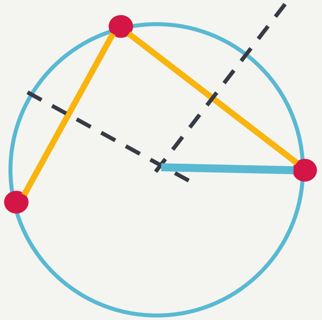
Example: Draw a circle through 3 points



We got the circle.
Perfect!

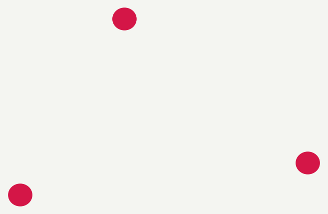
Mila

Example: Draw a circle through 3 points

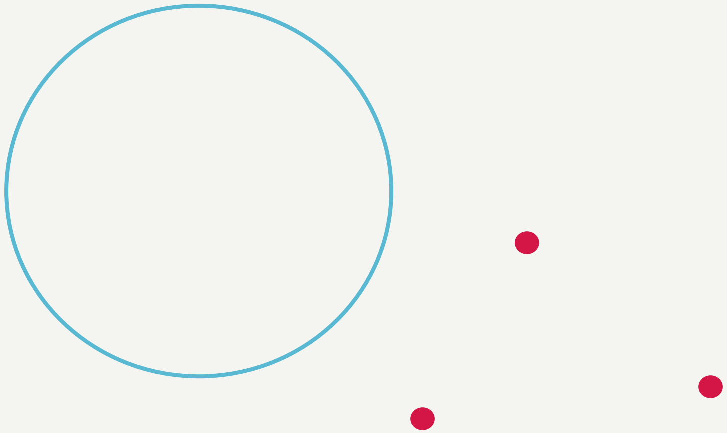


Closed-form solution

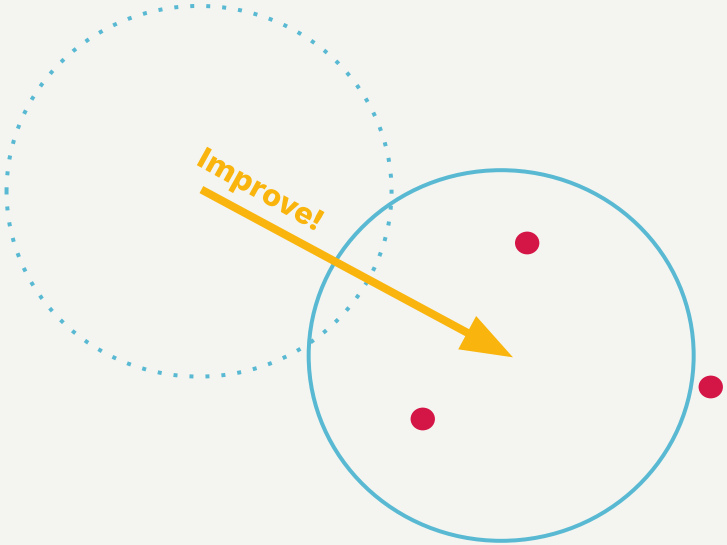
Example: Draw a circle through 3 points



Example: Draw a circle through 3 points

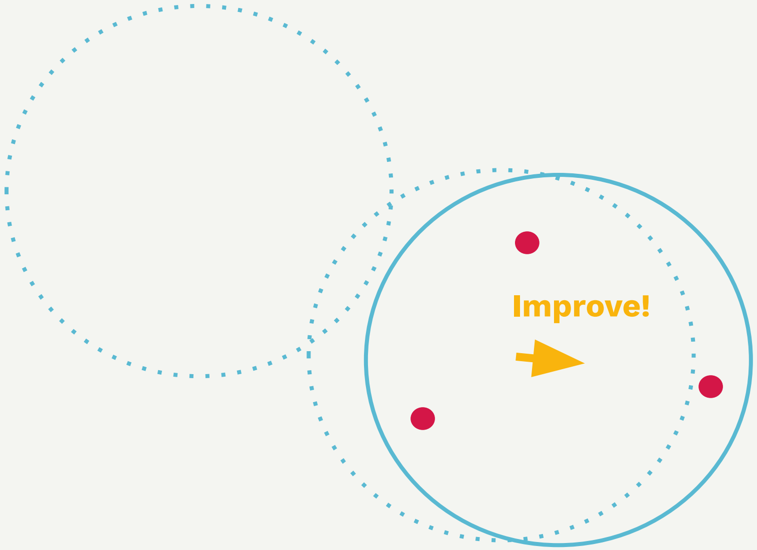


Example: Draw a circle through 3 points



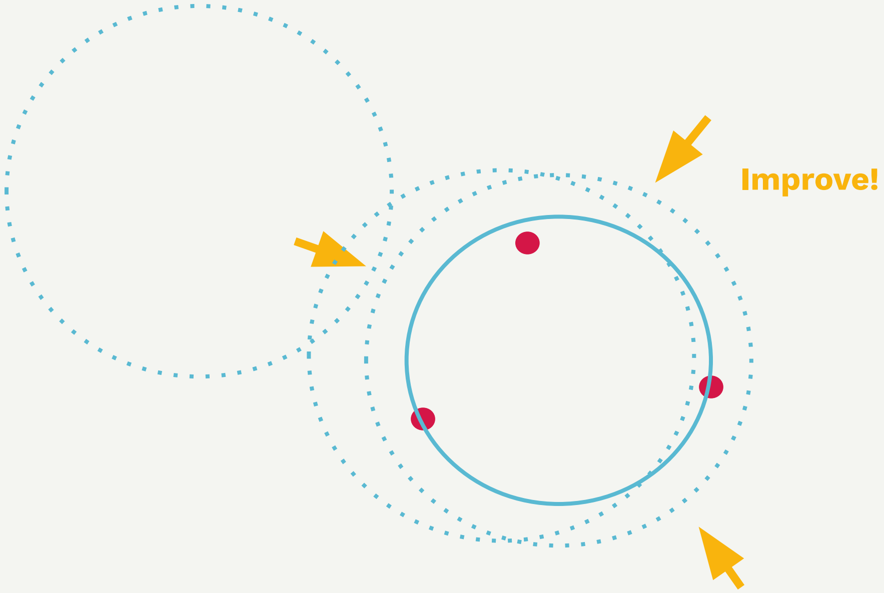
Mila

Example: Draw a circle through 3 points

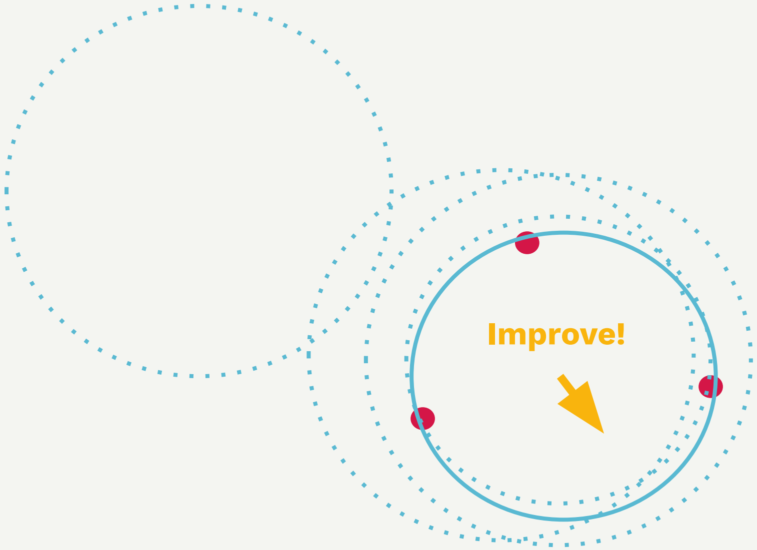


Mila

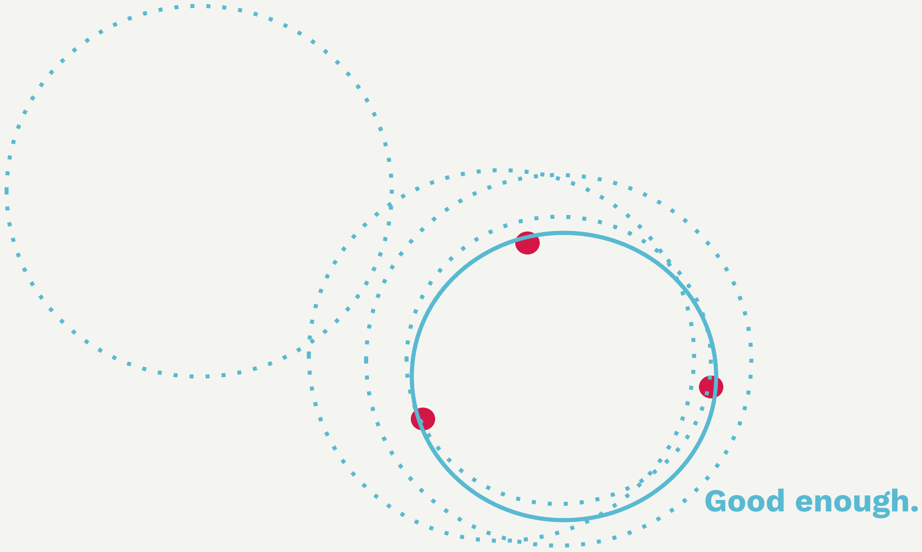
Example: Draw a circle through 3 points



Example: Draw a circle through 3 points

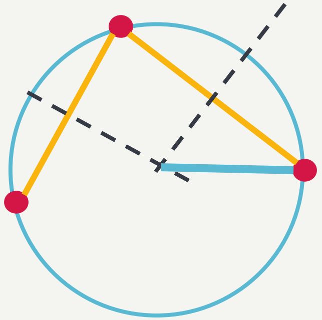


Example: Draw a circle through 3 points

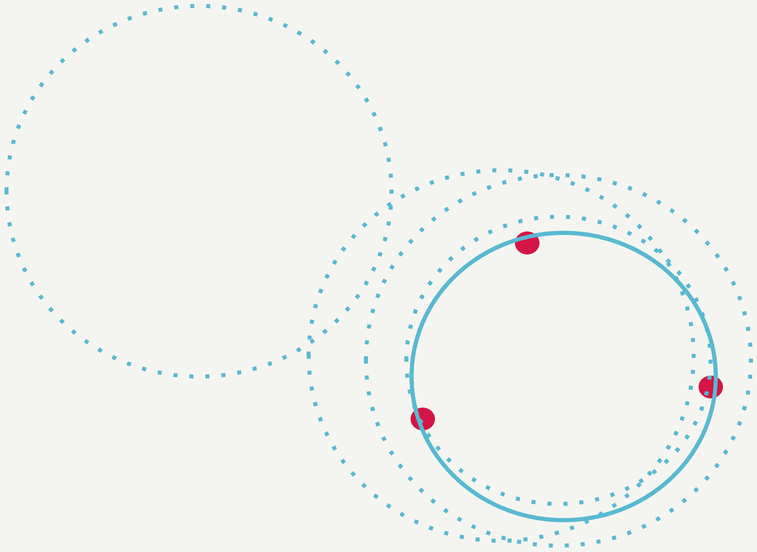


Mila

Example: Draw a circle through 3 points



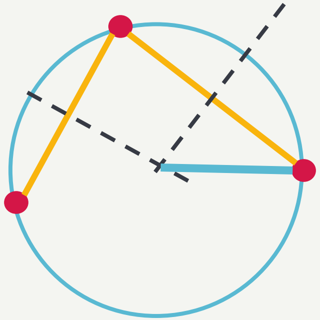
Closed-form solution

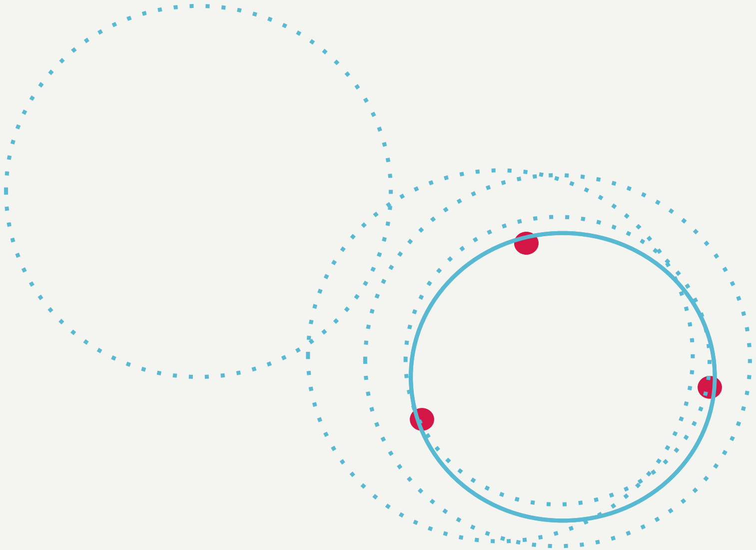


Iterative learning

Mila

Example: Draw a circle through 3 points





Closed-form solution

- Gives you the exact solution.
- Can be quick (only once a method has been defined!)
- Can be too complex to solve!

Iterative learning

- Can only give you a good enough solution.
- Can be time consuming.
- (Almost) always works!

Example: Linear Regression

Recall Linear Regression

→ regression problem

→ input: feature vector **x** = (x_1, x_2, ..., x_n)$\in\mathbb{R}^n$
→ target: scalar $y\in\mathbb{R}$

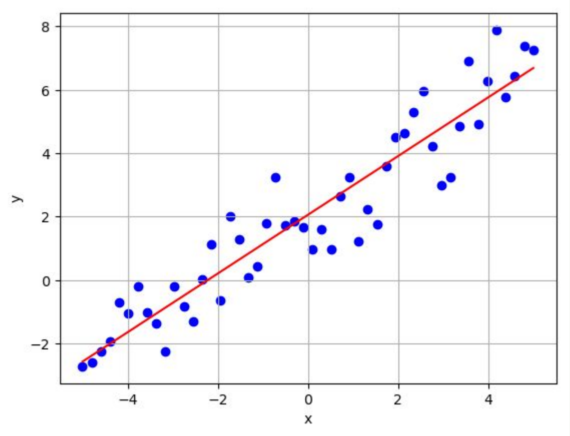
Linear regression implies that its output is a linear function of the input.

$$\hat{y} = w_1 x_1 + w_2 x_2 + \ + w_n x_n = \mathbf{w}^T \mathbf{x}$$ +b

+b

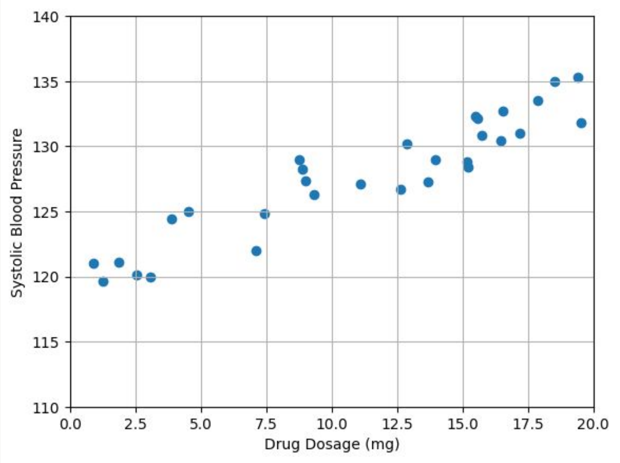
w = (w, w_2, ..., w_n) $\in\mathbb{R}^n$ is a vector of **parameters.**

and b is also a parameter.



Example: Linear Regression

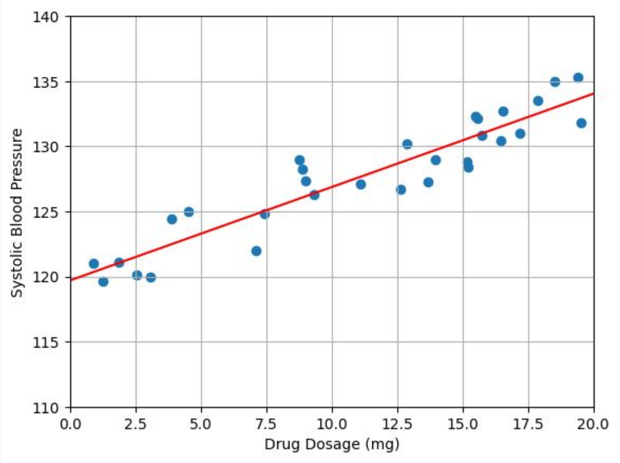
Blood pressure = w*Dosage + b



Mila

Example: Linear Regression

Blood pressure = w*Dosage + b



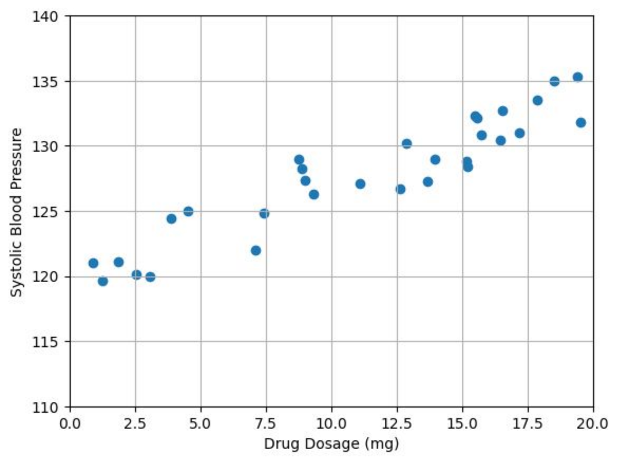
Closed-form solution

$$w = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

$$b = \frac{\sum y_i - w\sum x_i}{n}$$

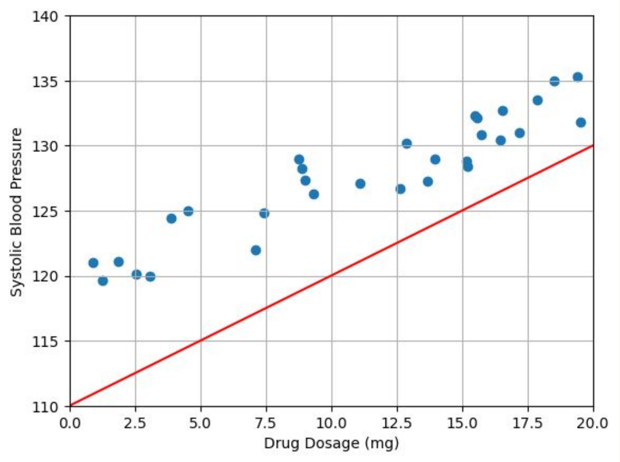
Example: Linear Regression

Blood pressure = w*Dosage + b



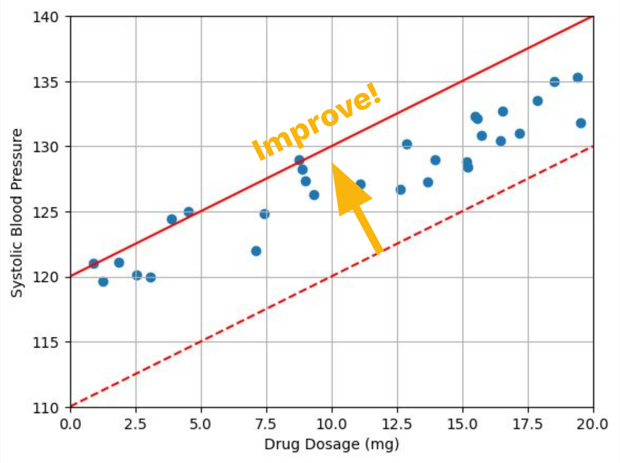
Example: Linear Regression

Blood pressure = w*Dosage + b



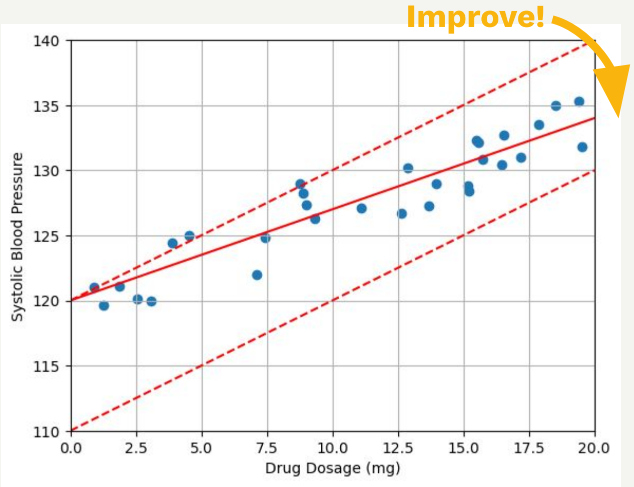
Example: Linear Regression

Blood pressure = w*Dosage + b



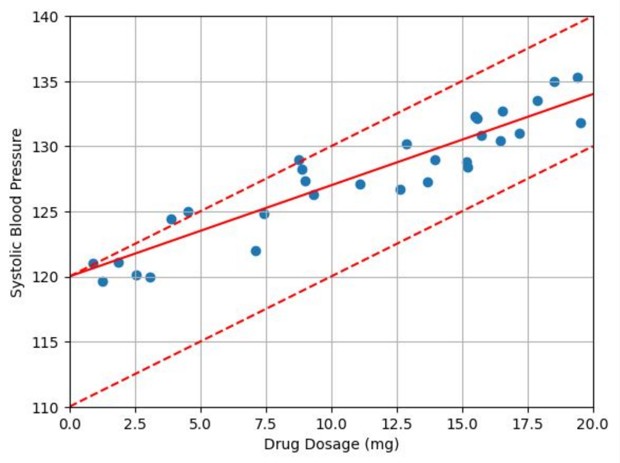
Example: Linear Regression

Blood pressure = w*Dosage + b



Example: Linear Regression

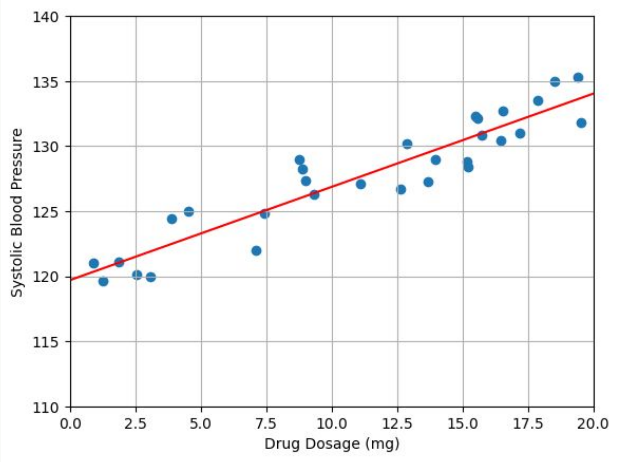
Blood pressure = w*Dosage + b

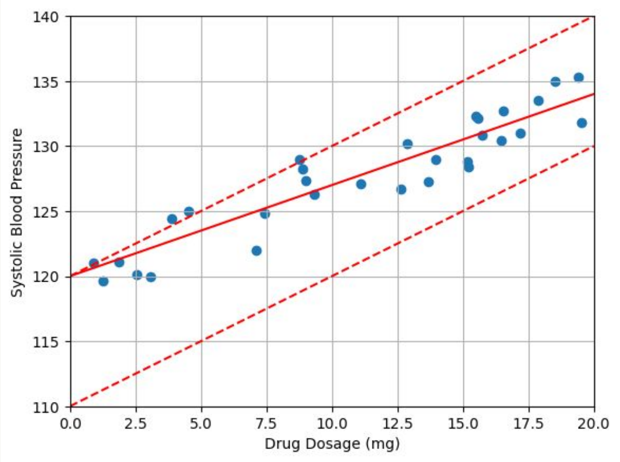


Good enough.

Example: Linear Regression

Blood pressure = w*Dosage + b





Closed-form solution

Iterative learning

$$w = \frac{n \sum (x_i y_i) - \sum x_i \sum y_i}{n \sum x_i^2 - (\sum x_i)^2}$$

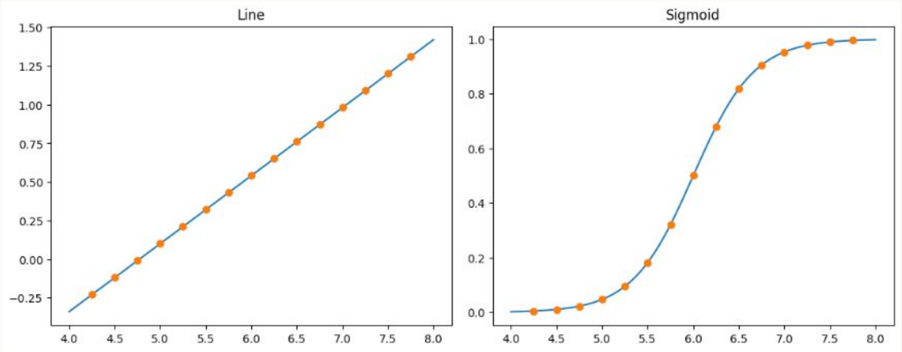
$$b = \frac{\sum y_i - w \sum x_i}{n}$$

Mila

Example: Logistic Regression

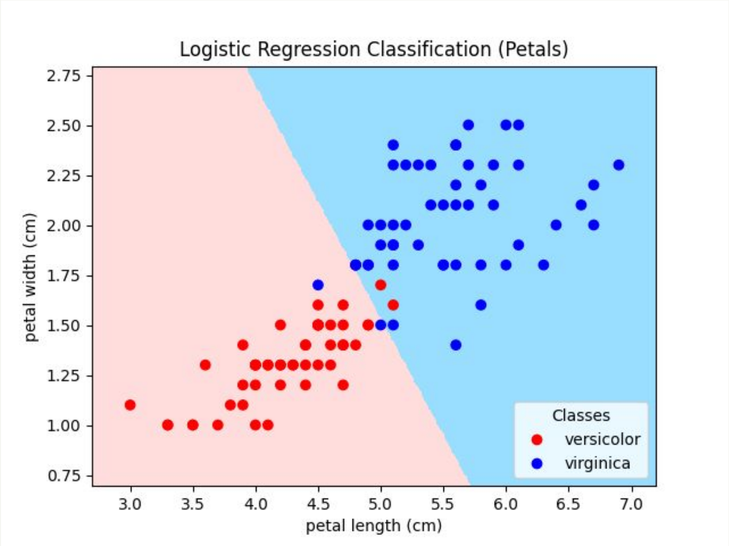
Recall Logistic Regression

$$y = wx + b \qquad y = \frac{1}{1 + e^{-(wx+b)}}$$



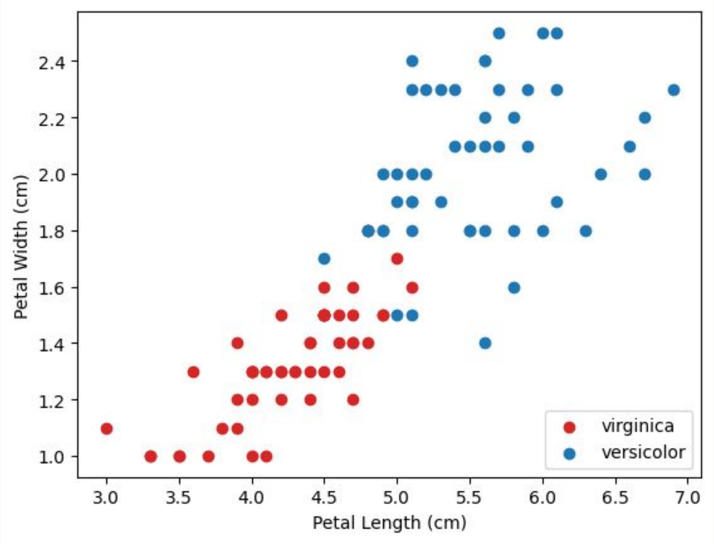
y = wx + b

y = σ(wx + b)



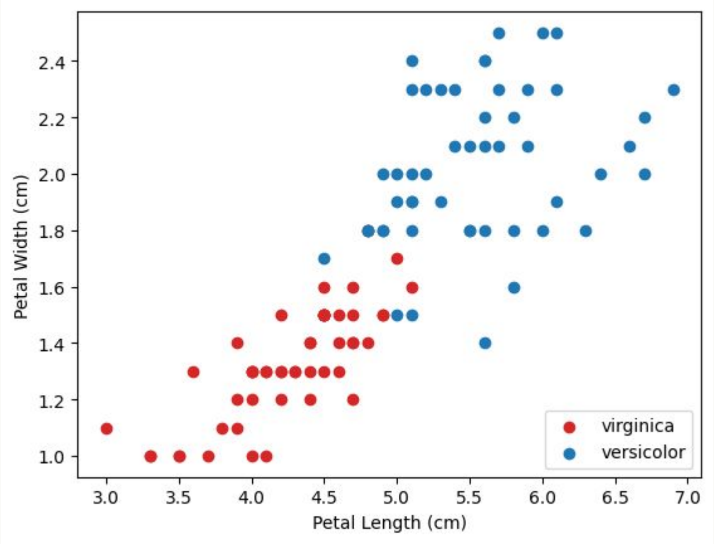
Example: Logistic Regression

Classify iris flowers



Example: Logistic Regression

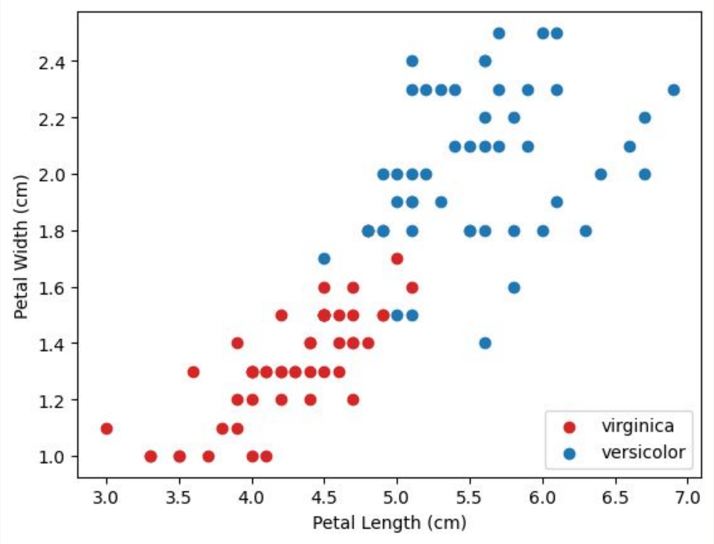
Classify iris flowers



Closed-form solution

Example: Logistic Regression

Classify iris flowers

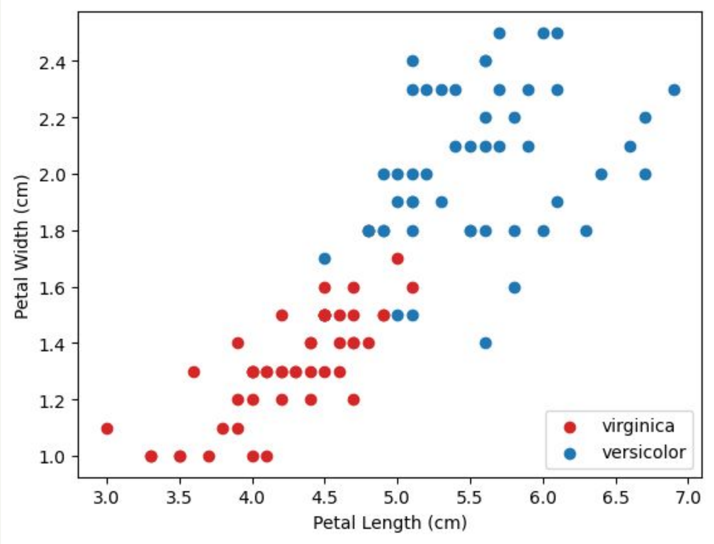


~~**Closed form solution**~~

Doesn’t exist!

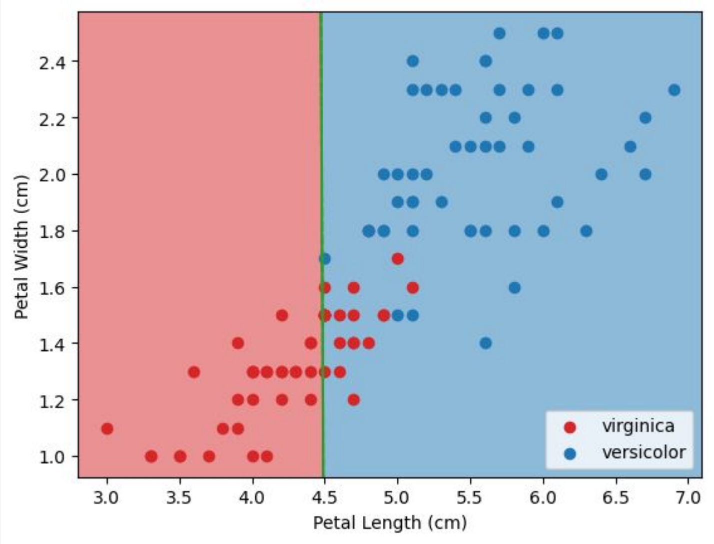
Example: Logistic Regression

Classify iris flowers



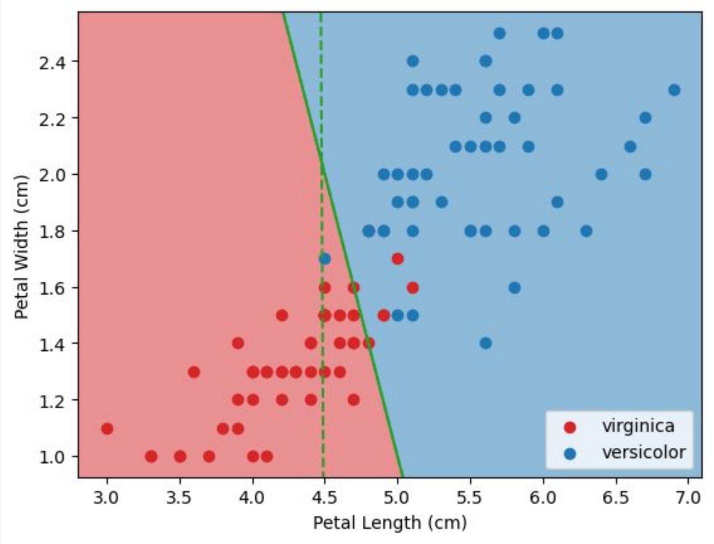
Example: Logistic Regression

Classify iris flowers



Example: Logistic Regression

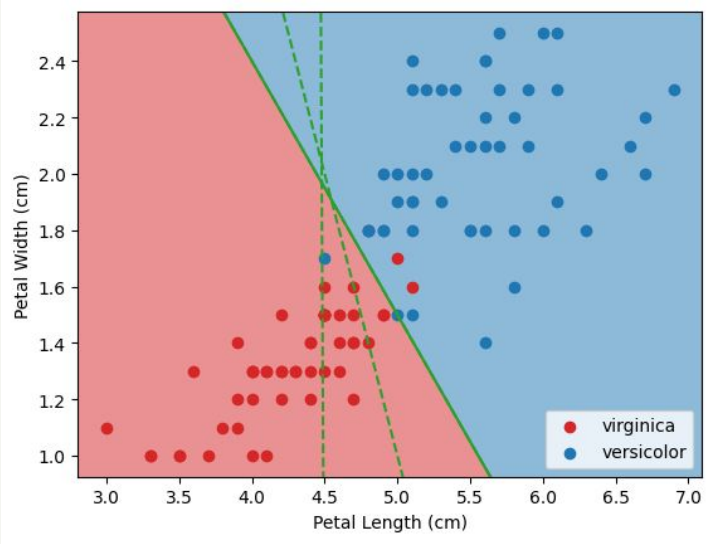
Classify iris flowers



Mila

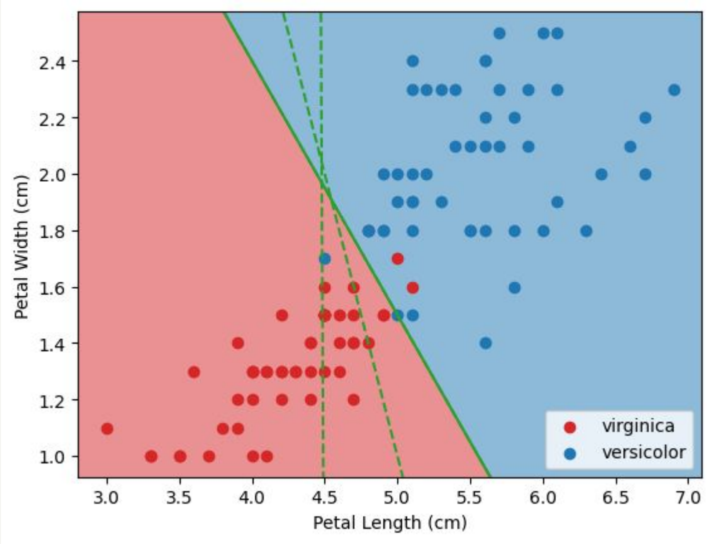
Example: Logistic Regression

Classify iris flowers



Example: Logistic Regression

Classify iris flowers

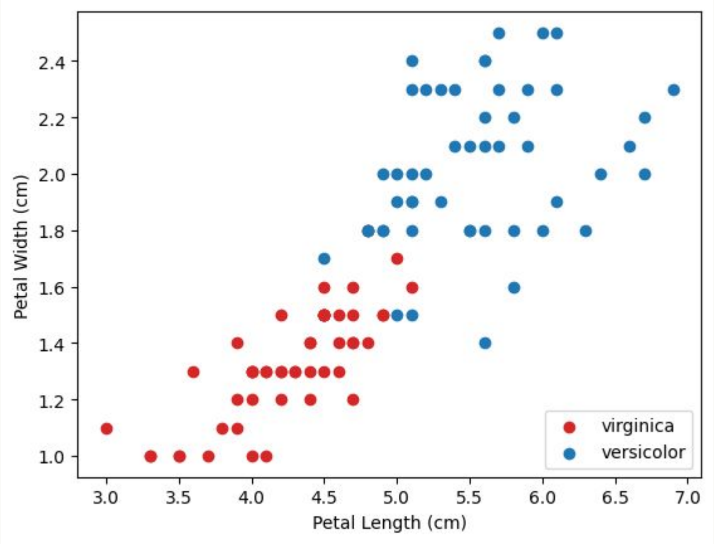


Good enough.

Mila

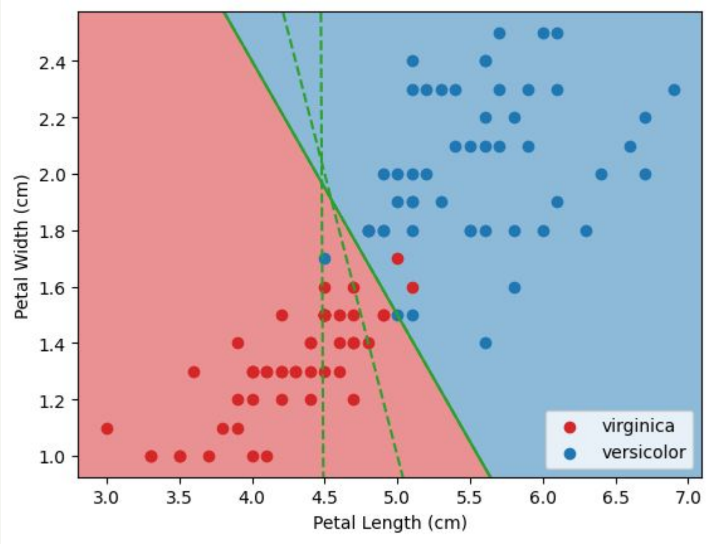
Example: Logistic Regression

Classify iris flowers



~~**Closed form solution**~~

Doesn’t exist!



Iterative learning

Mila

Closed-form vs Iterative Learning

Closed-form Solution

- Can provide an exact answer without approximations.
- Once derived, the solution can be computed very quickly, often in constant time.
- **Deriving a closed-form solution can be complex and is not always possible for every problem.**

Iterative Learning

- Only provides good enough answers.
- Each iteration can be expensive, and the total time depends on the number of iterations.
- **Highly flexible and can be applied to a wide range of problems, including those without closed-form solutions.**

Closed-form vs Iterative Learning

Closed-form Solution

- Can provide an exact answer without approximations.
- Once derived, the solution can be computed very quickly, often in constant time.
- **Deriving a closed-form solution can be complex and is not always possible for every problem.**

Iterative Learning

- Only provides good enough answers.
- Each iteration can be expensive, and the total time depends on the number of iterations.
- **Highly flexible and can be applied to a wide range of problems, including those without closed-form solutions.**

Moving towards a more 'general' form of learning!

But how do we ‘quantify’ getting better?

Recall Loss Functions

→ During training, we want to measure the discrepancy between the target variables y and the outcome of the hypotheses $h(x)$.

→ **Loss function**: $L(y, h(x))$

A loss function quantifies how poorly $h(x)$ approximates y

→ smaller values of $L(y, h(x))$ are better
→ generally, $L(y, y)=0$ and $L(y, h(x)) > 0$ for all (x,y)

But how do we ‘quantify’ getting better?

Recall Empirical Risk Minimization

The **empirical risk** is the average loss over all observed data points in the dataset.

$$R_N(h) = \frac{1}{N}\sum_{i=1}^{N} L(y^i, \hat{y}^i) = \frac{1}{N}\sum_{i=1}^{N} L(y^i, h(x^i, \theta))$$

If the empirical risk is small, we say that **the predictor fits the data well** (according to the loss *L*).

Note: we used θ because that is how you will see the empirical risk written in most textbooks, but it corresponds to our **w**!

Mila

But how do we ‘quantify’ getting better?

- ➔ Loss functions/Empirical Risk are a measure of how ‘good’ is our solution (Lower is better!)
- ➔ To get ‘better’ is to reduce loss.
- ➔ How to reduce loss? How to minimize the empirical risk?

Derivatives

How familiar is everyone with derivatives?

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by Δ**x** (which is really small)?

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Consider **f(x) = x²**

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(x + \Delta x)^2 - x^2}{\Delta x}$$

Mila

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Consider **f(x) = x^2**

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(x + \Delta x)^2 - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{x^2 + (\Delta x)^2 + 2x(\Delta x) - x^2}{\Delta x}$$

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Consider **f(x) = x²**

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(x + \Delta x)^2 - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{x^2 + (\Delta x)^2 + 2x(\Delta x) - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(\Delta x)^2 + 2x(\Delta x)}{\Delta x}$$

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Consider **f(x) = x²**

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(x + \Delta x)^2 - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{x^2 + (\Delta x)^2 + 2x(\Delta x) - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(\Delta x)^2 + 2x(\Delta x)}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \Delta x + 2x$$

Derivatives

Derivative of a function **f(x)** with respect to **x** is - How much would **f(x)** change (rate of change) if we changed **x** by **Δx** (which is really small)?

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Consider **f(x) = x²**

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(x + \Delta x)^2 - x^2}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{x^2 + (\Delta x)^2 + 2x(\Delta x) - x^2}{\Delta x}$$

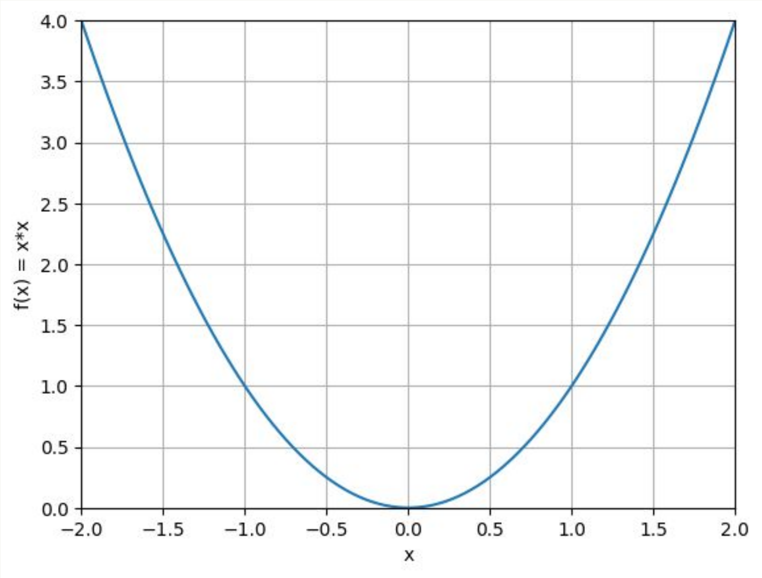
$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \frac{(\Delta x)^2 + 2x(\Delta x)}{\Delta x}$$

$$\frac{df(x)}{dx} = \lim_{\Delta x \to 0} \Delta x + 2x$$

$$\frac{df(x)}{dx} = 2x$$

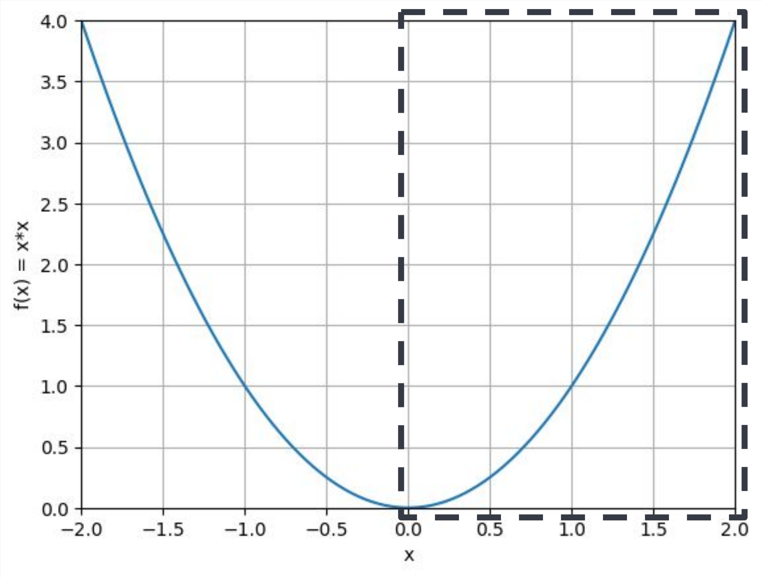
Derivatives

Consider $f(x) = x^2$



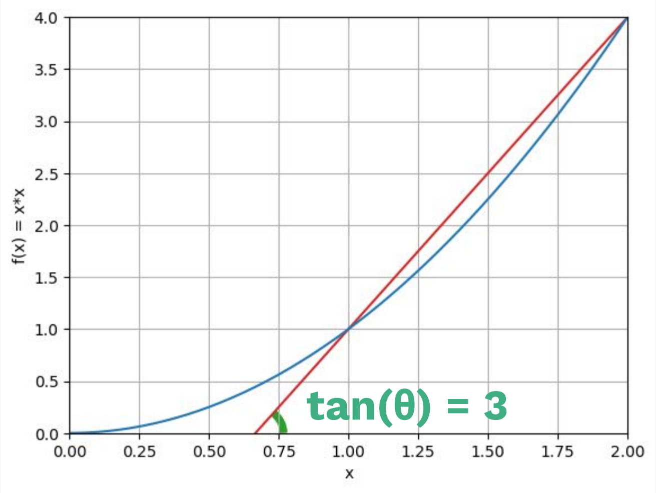
Derivatives

Consider **f(x) = x²**



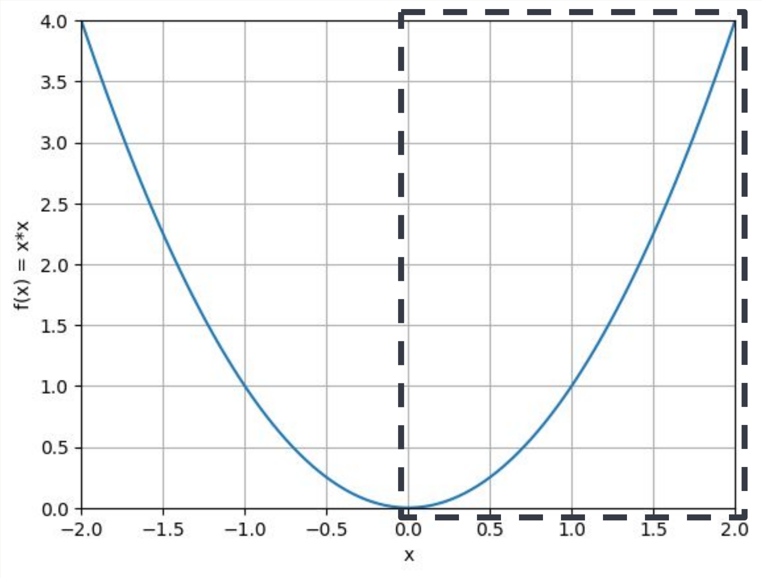
x = 1 and Δx = 1

$$\frac{f(1+1)-f(1)}{1}=3$$



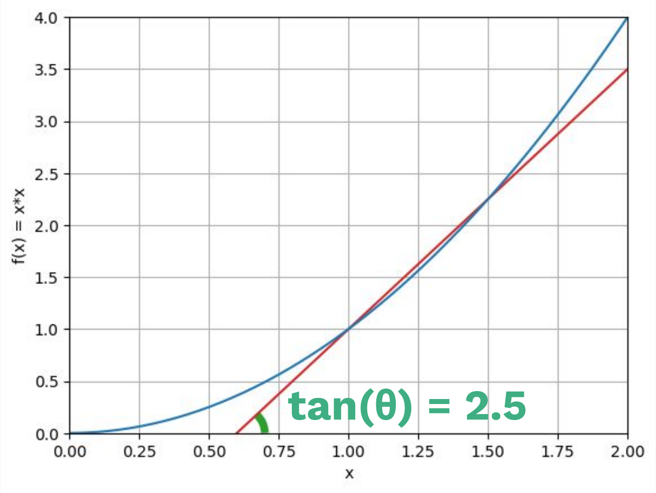
Derivatives

Consider **f(x) = x^2**



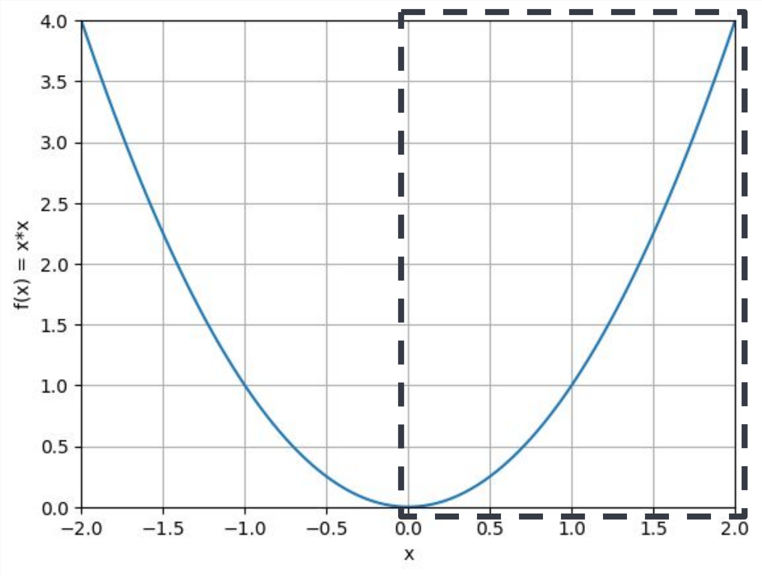
x = 1 and Δx = 0.5

$$\frac{f(1+0.5)-f(1)}{0.5} = 2.5$$



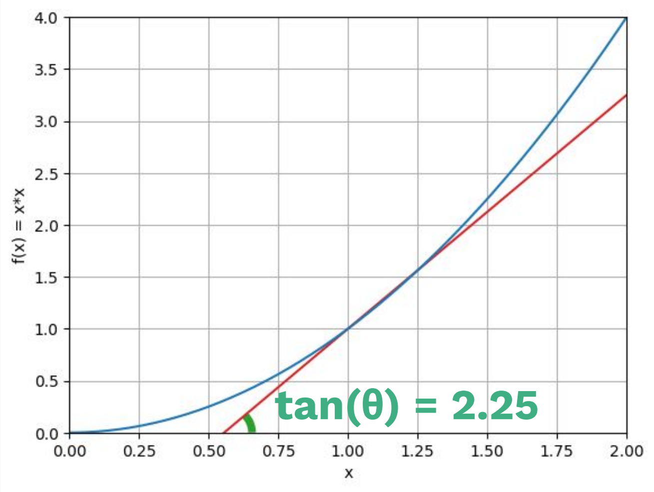
Derivatives

Consider **f(x) = x²**



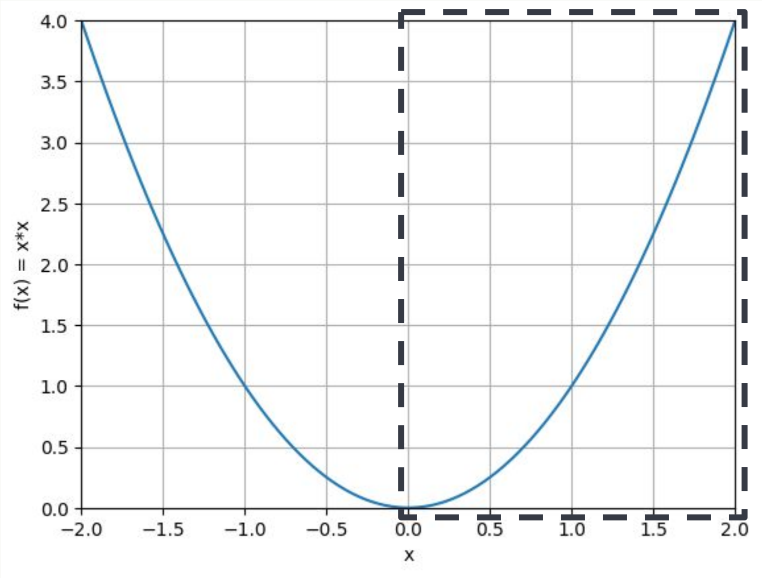
x = 1 and Δx = 0.25

$$\frac{f(1+0.25)-f(1)}{0.25}=2.25$$



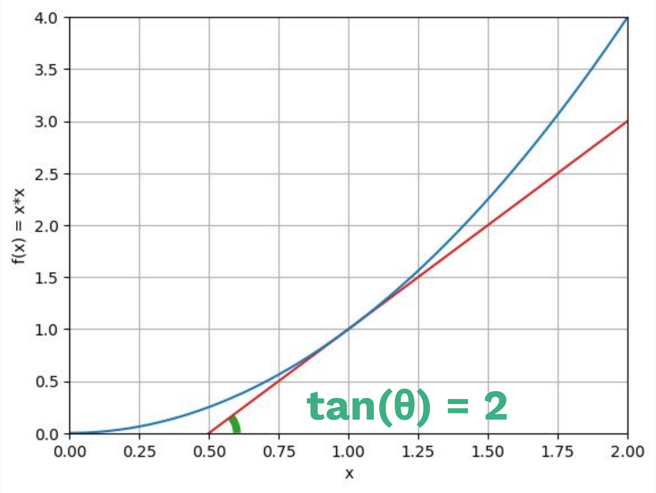
Derivatives

Consider **f(x) = x²**



x = 1 and Δx → 0

$$\lim_{\Delta x \to 0} \frac{f(1+\Delta x) - f(1)}{\Delta x} = 2$$



Mila

Derivatives

- Derivative of a function at a point is the **rate of change** of a function at that point.
- Derivative of a function at a point is the **slope of the tangent line** at that point.
- Also written as **f'(x)**

Break

Derivatives

- ➔ Derivative of a function at a point is the **rate of change** of a function at that point.
- ➔ Derivative of a function at a point is the **slope of the tangent line** at that point.
- ➔ Also written as **f'(x)**

Mila

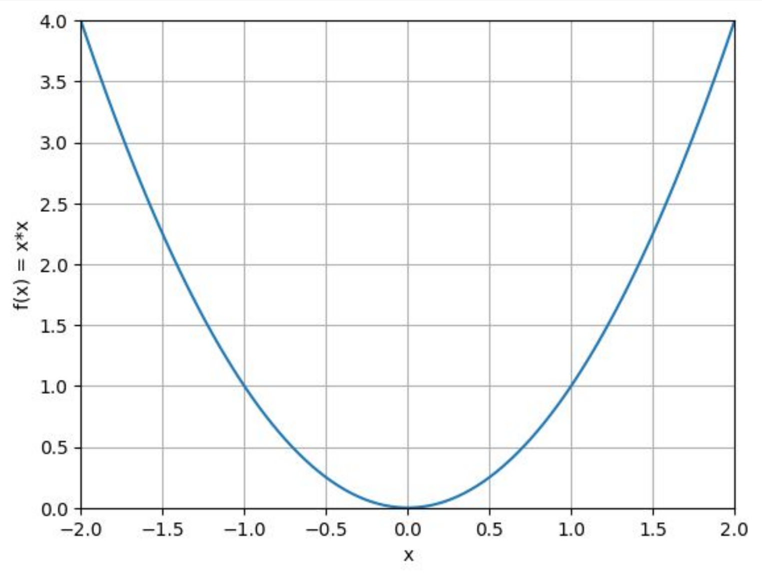
Derivatives

- → Derivative of a function at a point is the **rate of change** of a function at that point.
- → Derivative of a function at a point is the **slope of the tangent line** at that point.
- → Also written as **f’(x)**

But why do we care about derivatives?

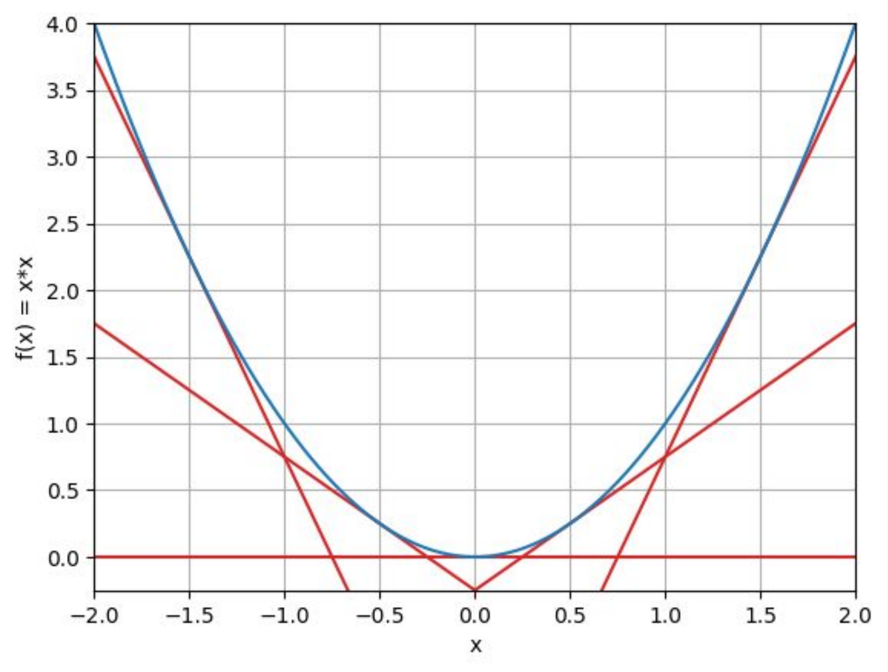
Special Case: Closed-form Solutions

Let's go back to **f(x) = x^2**



Special Case: Closed-form Solutions

Anything special about the derivative at the minimum?

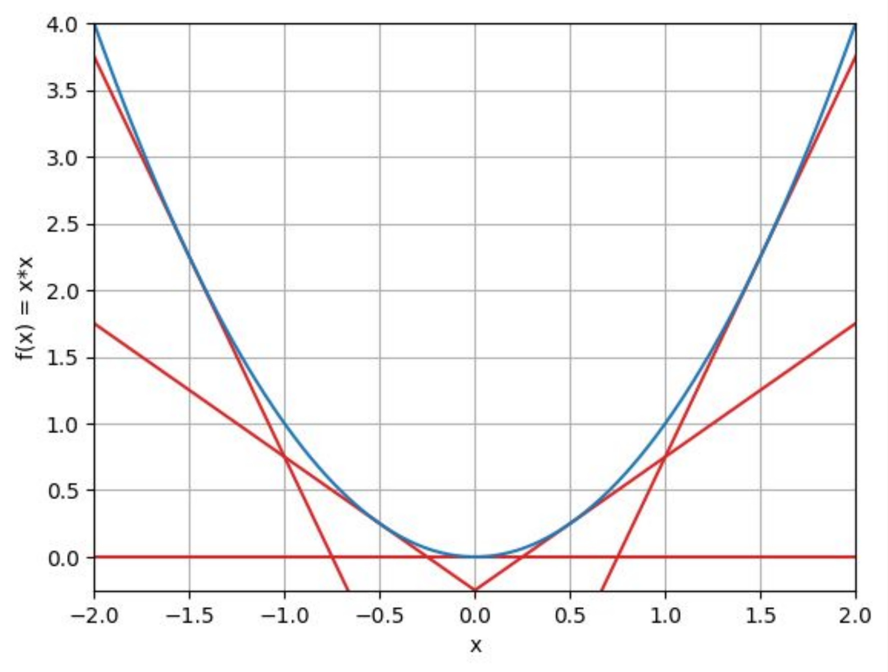


Special Case: Closed-form Solutions

Anything special about the derivative at the minimum?

It's a horizontal line!

That means, derivative is zero!

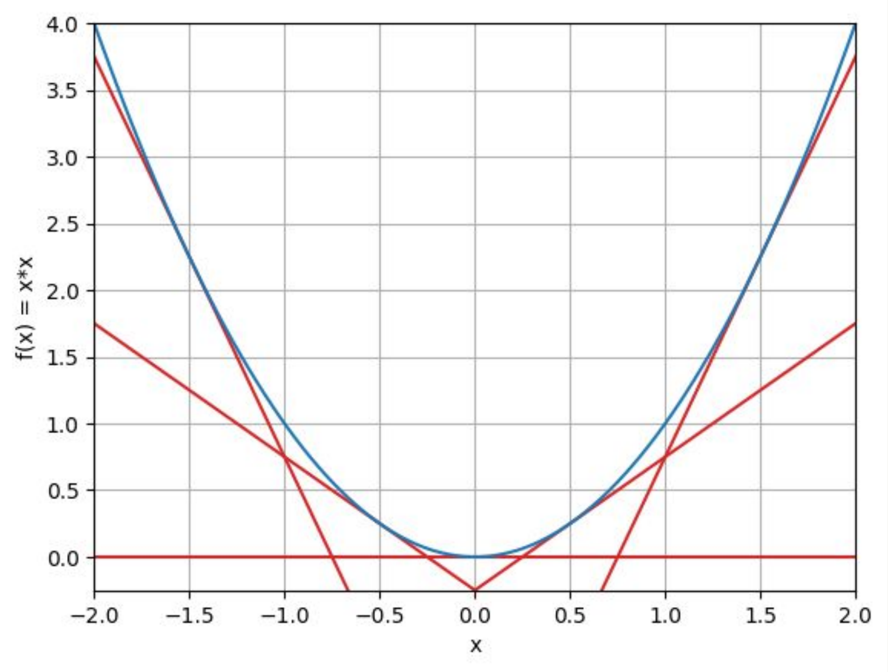


Special Case: Closed-form Solutions

Anything special about the derivative at the minimum?

It's a horizontal line!

That means, derivative is zero!



$f(x) = x^2$

We know $f'(x) = 2x$

$f'(x) = 0$

$\rightarrow 2x = 0$

$\rightarrow x = 0$

Special Case: Closed-form Solutions

➔ **f'(x) = 0** is the point where the rate of change is 0, i.e., the function doesn't change.
This will happen at the minimum value of a function! (although it can also happen at other places)

Mila

Special Case: Closed-form Solutions

➔ **f'(x) = 0** is the point where the rate of change is 0, i.e., the function doesn't change.
This will happen at the minimum value of a function! (although it can also happen at other places)

➔ That's how we got the values for linear regression.

$$w = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2}$$

$$b = \frac{\sum y_i - w\sum x_i}{n}$$

Mila

Special Case: Closed-form Solutions

➔ **f'(x) = 0** is the point where the rate of change is 0, i.e., the function doesn't change.
This will happen at the minimum value of a function! (although it can also happen at other places)

➔ That's how we got the values for linear regression.

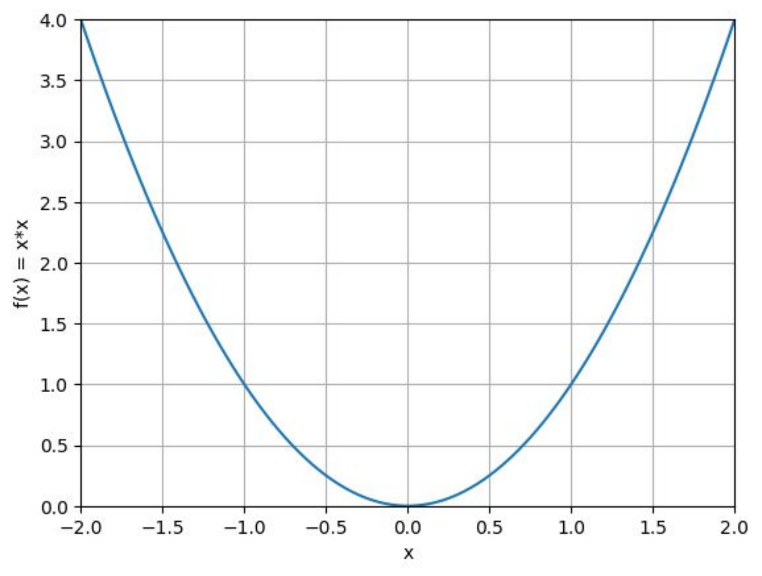
$$w = \frac{n\sum(x_i y_i) - \sum x_i \sum y_i}{n\sum x_i^2 - (\sum x_i)^2} \qquad b = \frac{\sum y_i - w\sum x_i}{n}$$

➔ But this is useful only if we can solve it! (We couldn't solve it for logistic regression)

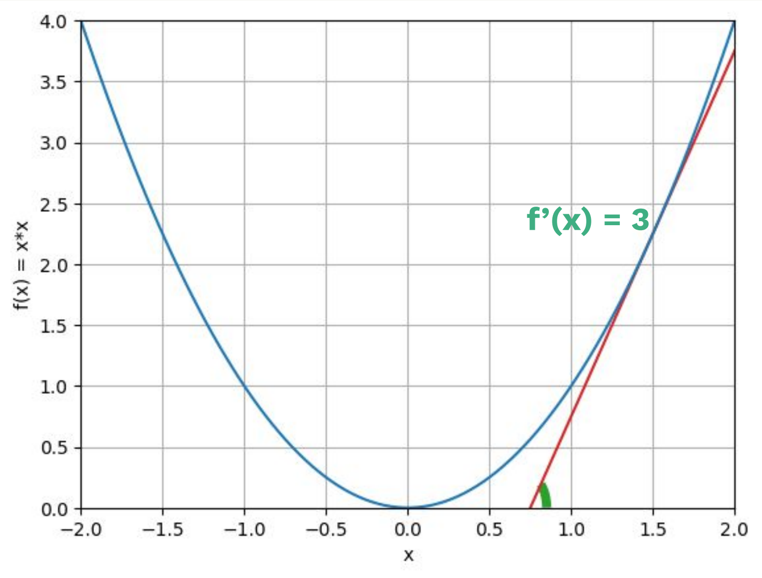
Mila

Gradient Descent

Let’s go back to **$f(x) = x^2$**

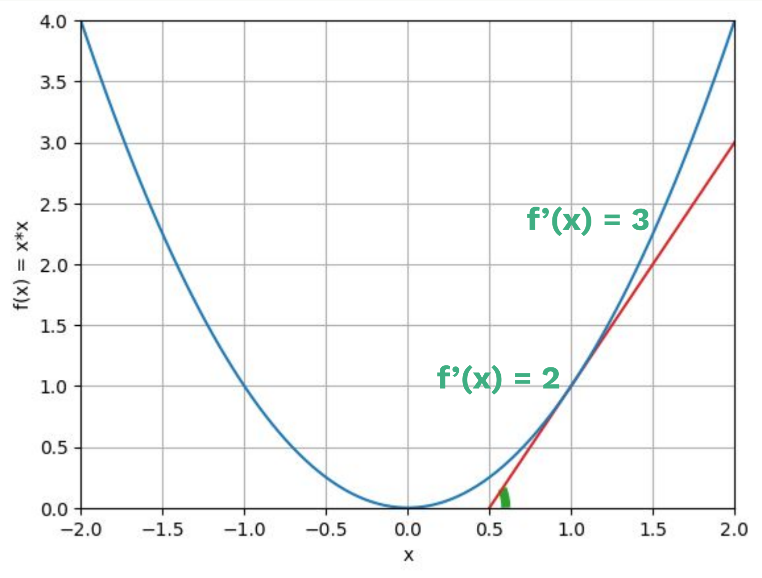


Gradient Descent

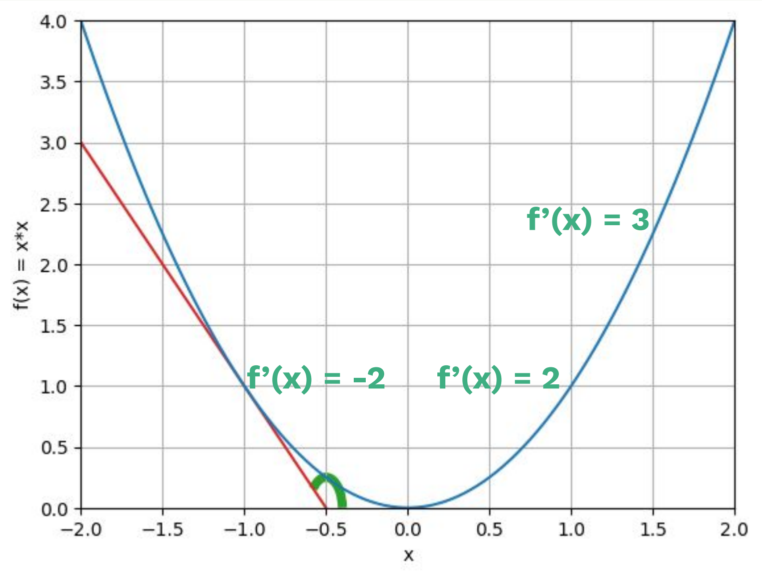


Mila

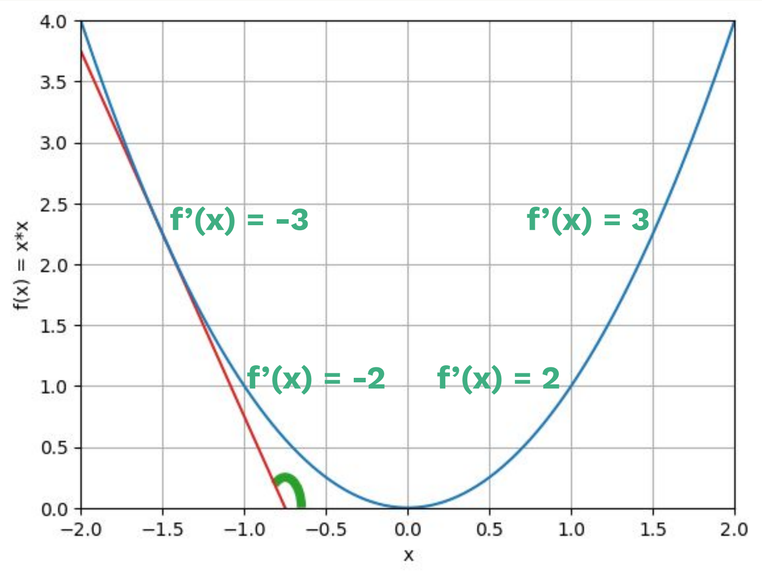
Gradient Descent



Gradient Descent



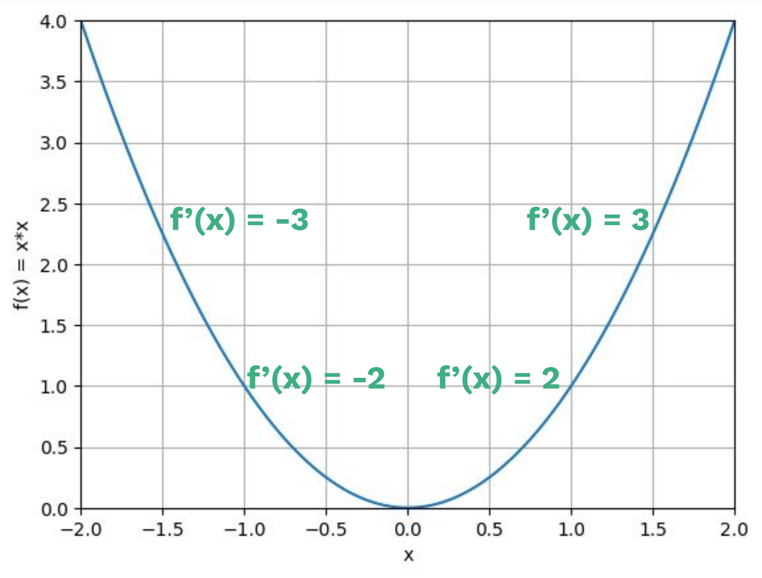
Gradient Descent



Mila

Gradient Descent

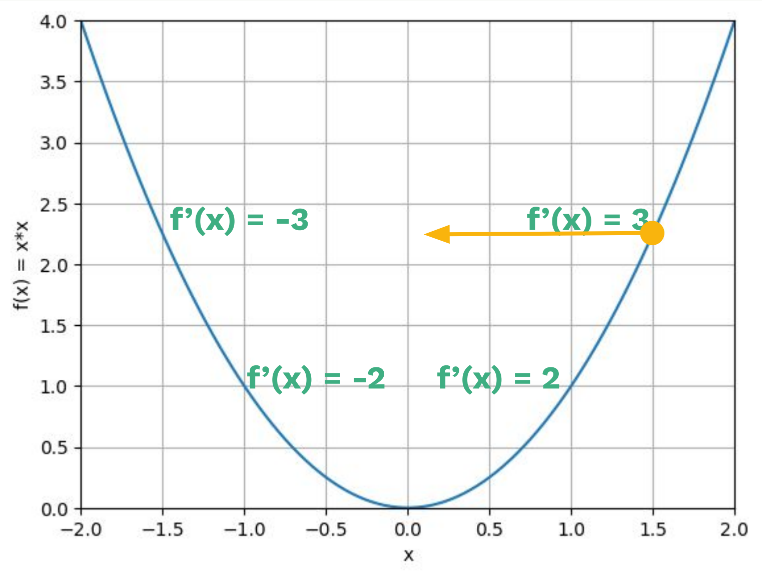
How can we 'minimize' **f(x)**?



Mila

Gradient Descent

How can we ‘minimize’ **f(x)**?

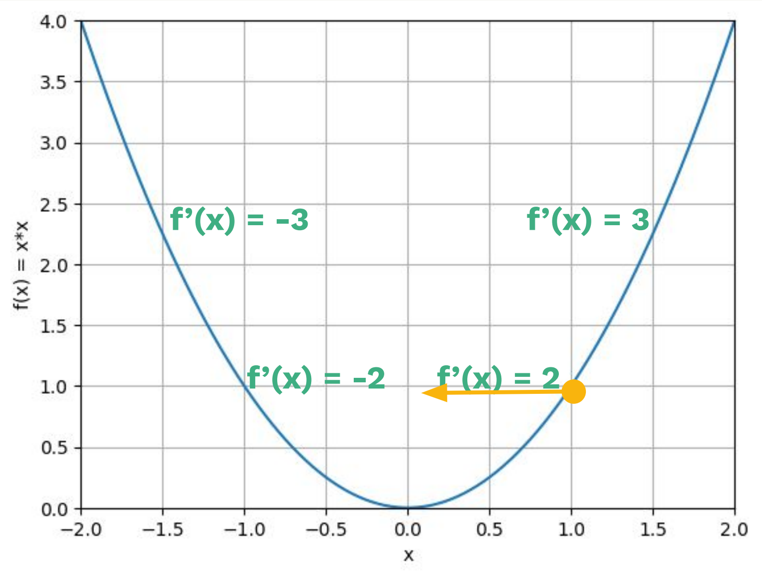


Move in the negative direction; Move a lot

Mila

Gradient Descent

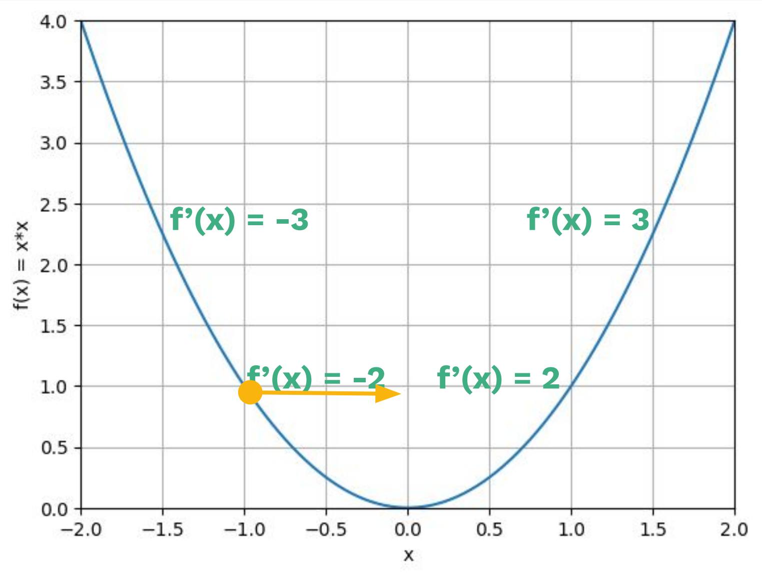
How can we 'minimize' **f(x)**?



Move in the negative direction; Move a little

Gradient Descent

How can we ‘minimize’ **f(x)**?

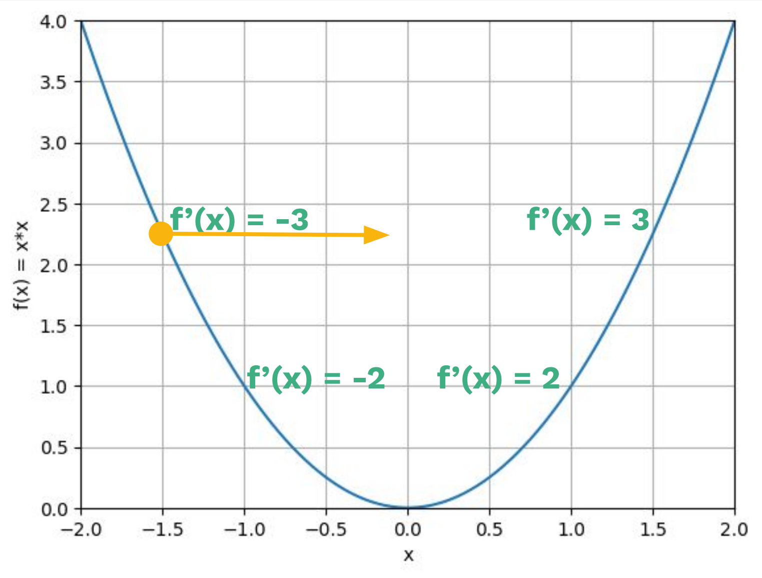


Move in the positive direction; Move a little

Mila

Gradient Descent

How can we ‘minimize’ **f(x)**?

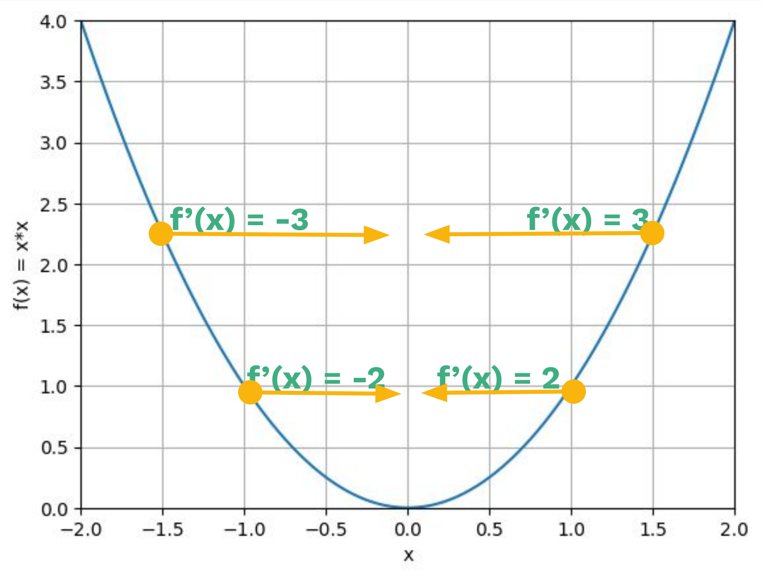


Move in the positive direction; Move a lot

Mila

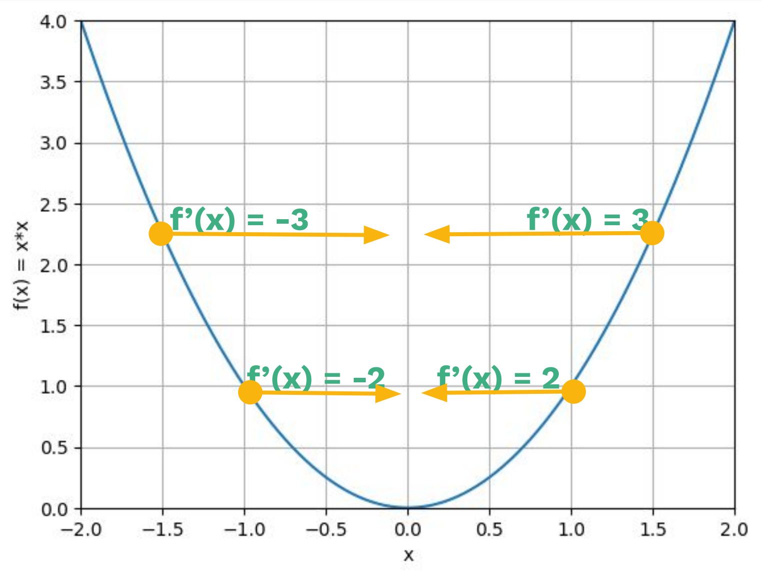
Gradient Descent

How can we 'minimize' **f(x)**?



Gradient Descent

How can we 'minimize' **f(x)**?



Move in the direction OPPOSITE of the derivative

Move the amount proportional to the derivative

Mila

Gradient Descent

$$\theta_i^{new} = \theta_i - \eta \frac{\partial L(.)}{\partial \theta_i}$$

Gradient Descent

Rate of change of loss
w.r.t. the parameter

$$\theta_i^{new} = \theta_i - \eta \frac{\partial L(.)}{\partial \theta_i}$$

Mila

Gradient Descent

Rate of change of loss
w.r.t. the parameter

$$\theta_i^{new} = \theta_i - \eta \frac{\partial L(.)}{\partial \theta_i}$$

Learning Rate

Mila

Gradient Descent

Current value of the parameter

Rate of change of loss w.r.t. the parameter

$$\theta_i^{new} = \theta_i - \eta \frac{\partial L(.)}{\partial \theta_i}$$

Learning Rate

Mila

Gradient Descent

$$\theta_i^{new} = \theta_i - \eta \frac{\partial L(.)}{\partial \theta_i}$$

Current value of the parameter

Rate of change of loss w.r.t. the parameter

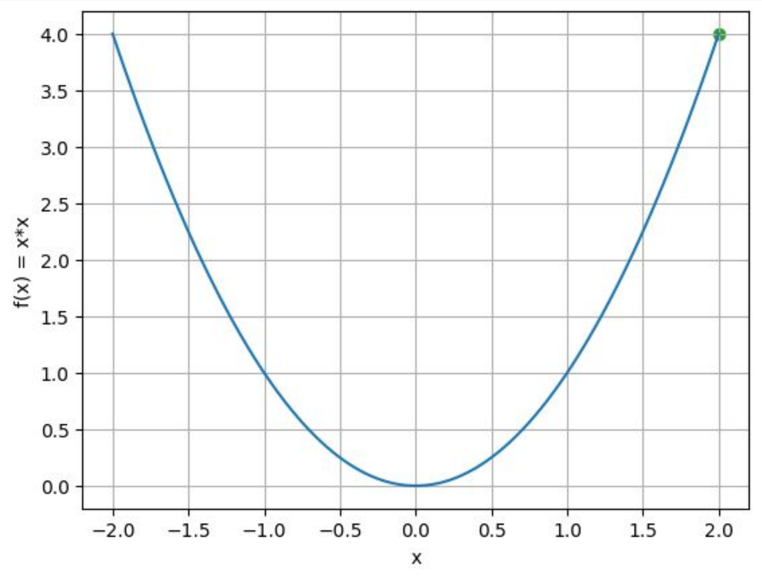
Improved value of the parameter

Learning Rate

Mila

Gradient Descent

Learning rate = 0.2



x^{new} = x - 0.2*f'(x)

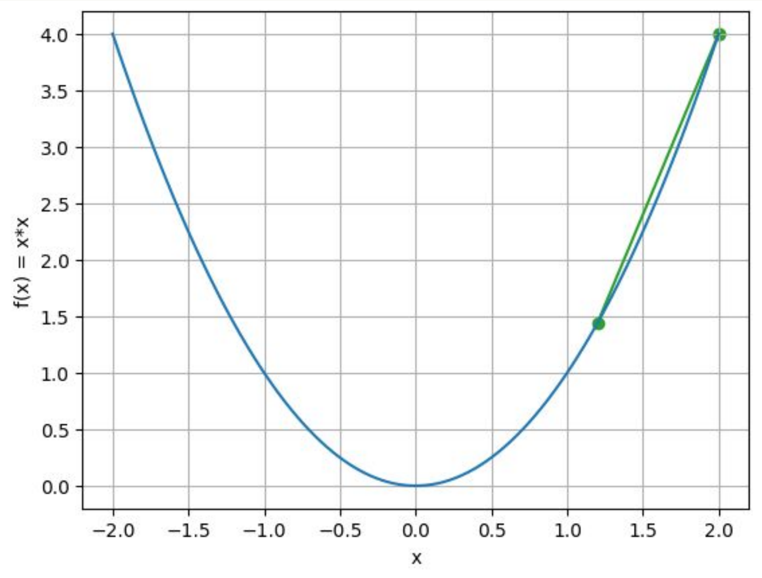
f'(2) = 4

x^{new} = 2 - 0.2*4

x^{new} = 1.2

Gradient Descent

Learning rate = 0.2



x^{new} = x - 0.2*f’(x)

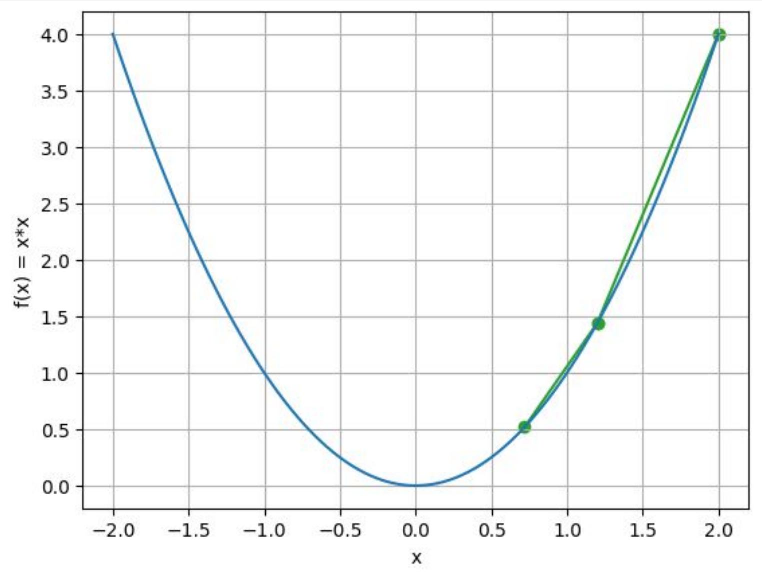
f’(1.2) = 2.4

x^{new} = 1.2 - 0.2*2.4

x^{new} = 0.72

Gradient Descent

Learning rate = 0.2



x^{new} = x - 0.2*f'(x)

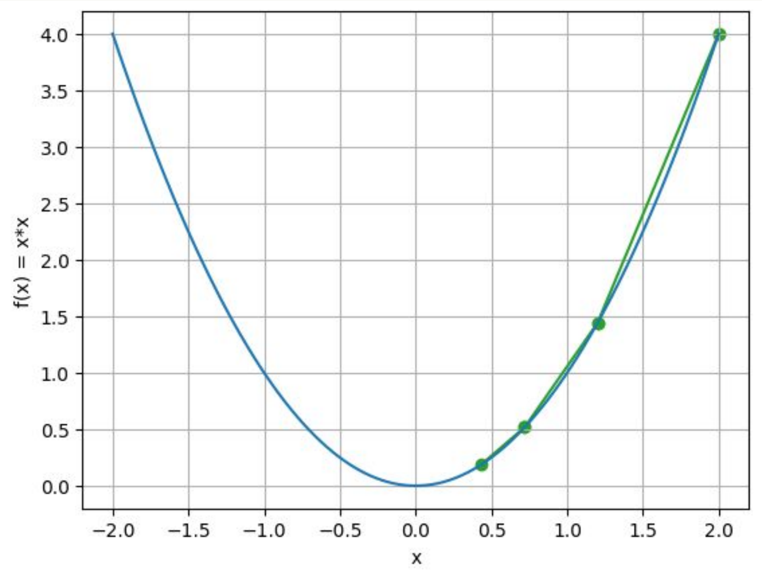
f'(0.72) = 1.44

x^{new} = 0.72 - 0.2*1.44

x^{new} = 0.432

Gradient Descent

Learning rate = 0.2



$x^{new} = x - 0.2*f'(x)$

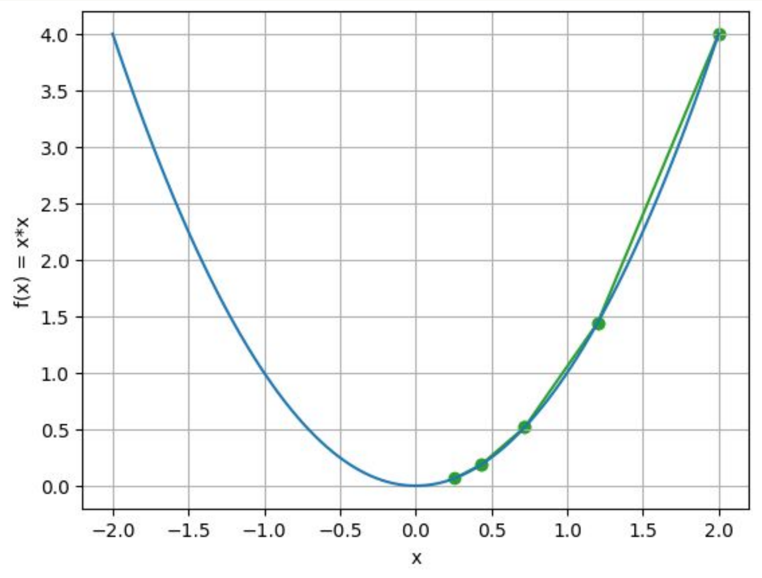
$f'(0.432) = 0.864$

$x^{new} = 0.432 - 0.2*0.864$

$x^{new} = 0.2592$

Gradient Descent

Learning rate = 0.2



Mila

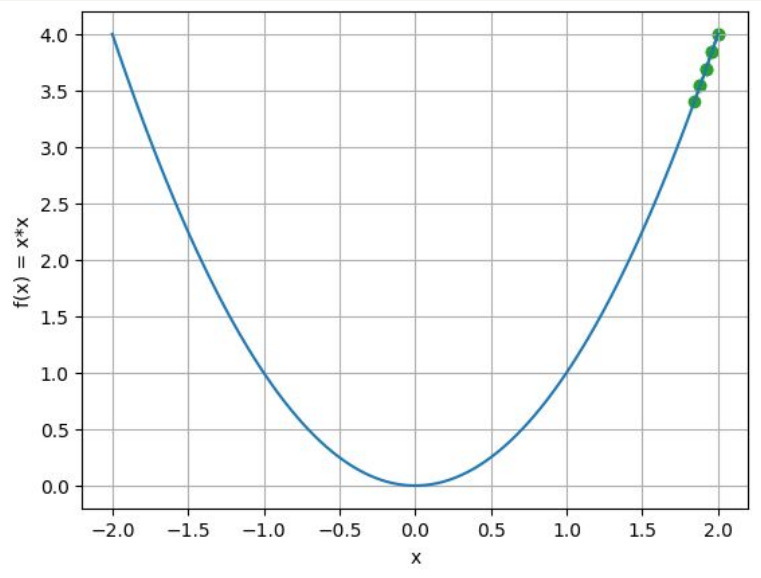
Problems with Gradient Descent

- ➔ Sensitivity to the learning rate
- ➔ Doesn't work with non-differentiable functions
- ➔ Can get stuck in local minima

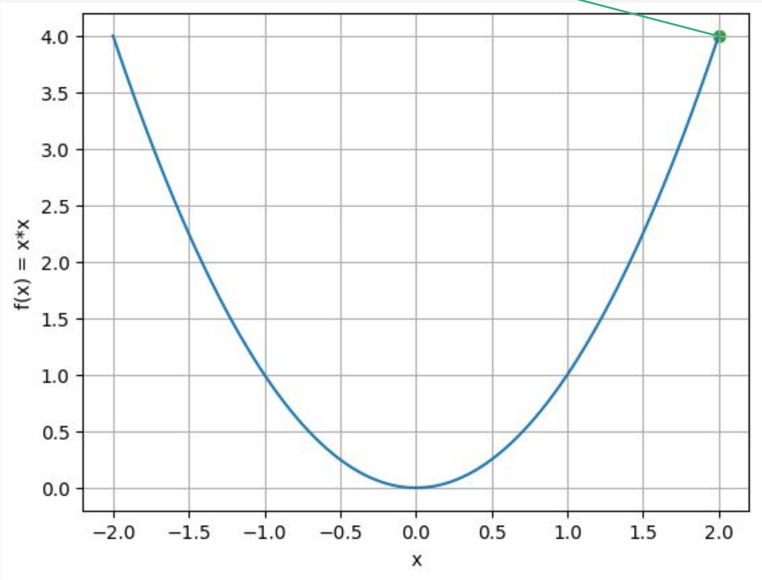
Mila

Sensitivity to Learning Rate

Learning rate = 0.01



Learning rate > 1



Mila

Needs differentiable functions

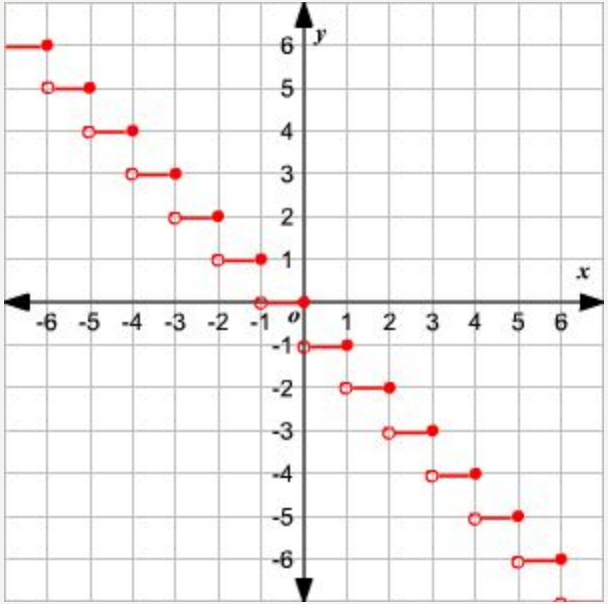


Image source: https://www.varsitytutors.com/hotmath/hotmath_help/topics/step-function

Mila

Can get stuck in local minima

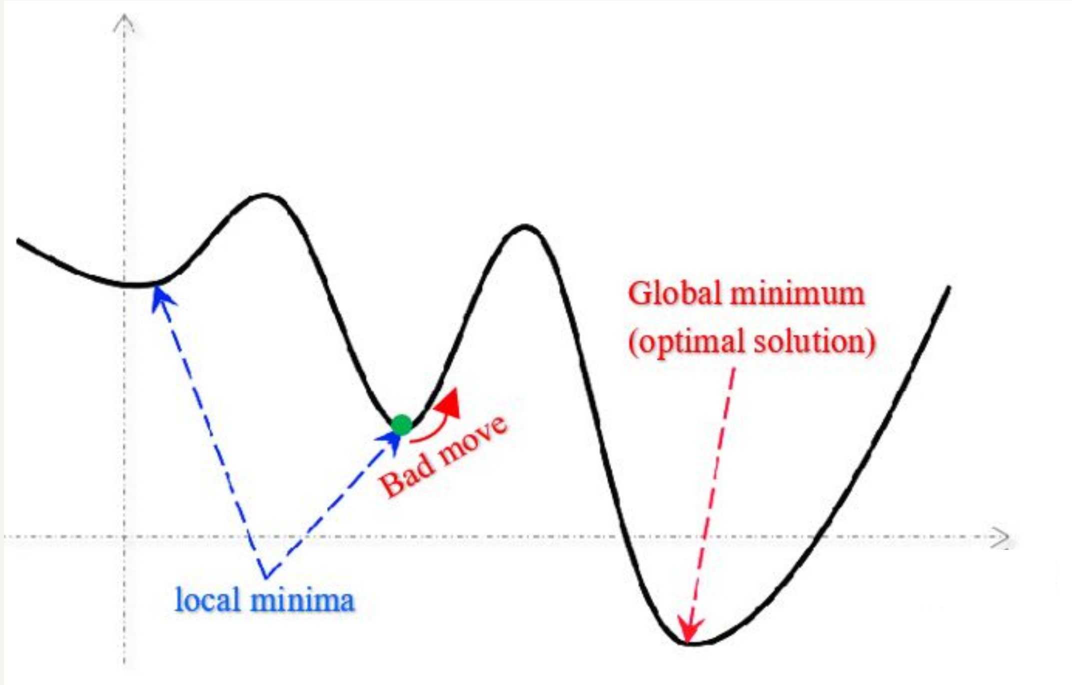


Image source: Elkarashily, Ahmed, et al. "VLSI Placement using Modified Parallel Simulated Annealing."

Mila

Why nonlinear learning?

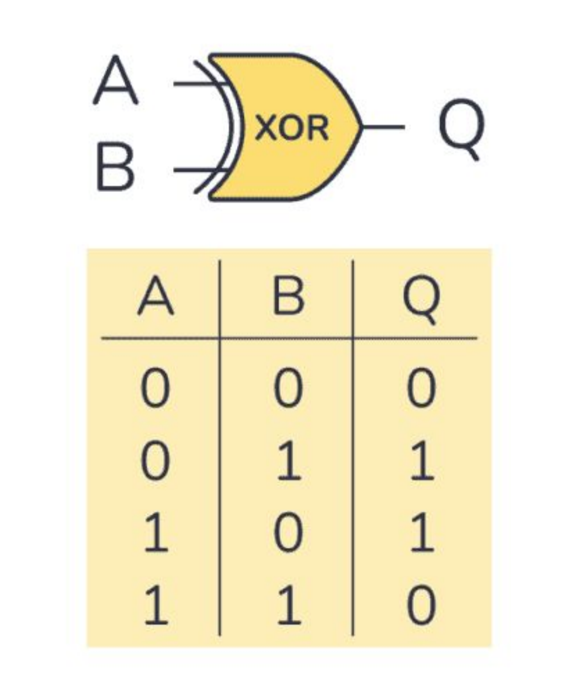
- ➔ We saw iterative learning (gradient descent) is needed when we want to learn nonlinear functions.
- ➔ But why do we need nonlinear functions? Is linear not enough?

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Image source: https://www.build-electronic-circuits.com/xor-gate/

Example: XOR Problem



A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

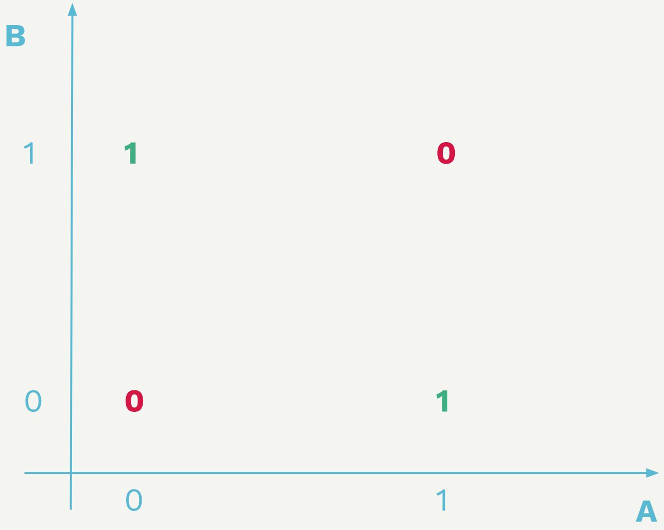
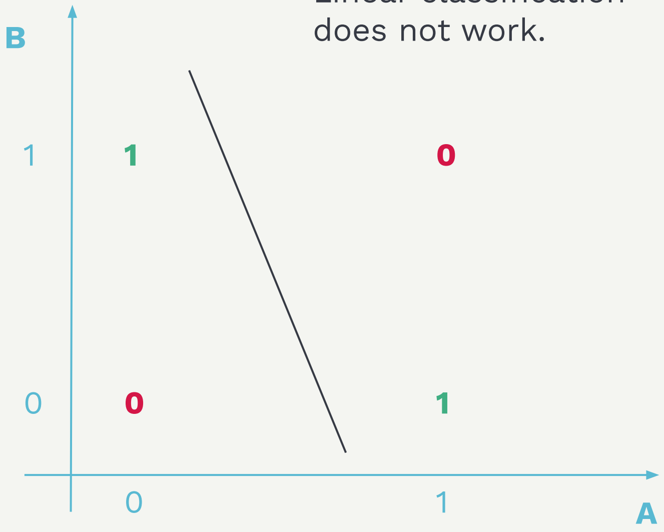


Image source: https://www.build-electronic-circuits.com/xor-gate/

Mila

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0



Linear classification does not work.

Image source: https://www.build-electronic-circuits.com/xor-gate/

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Linear classification does not work.

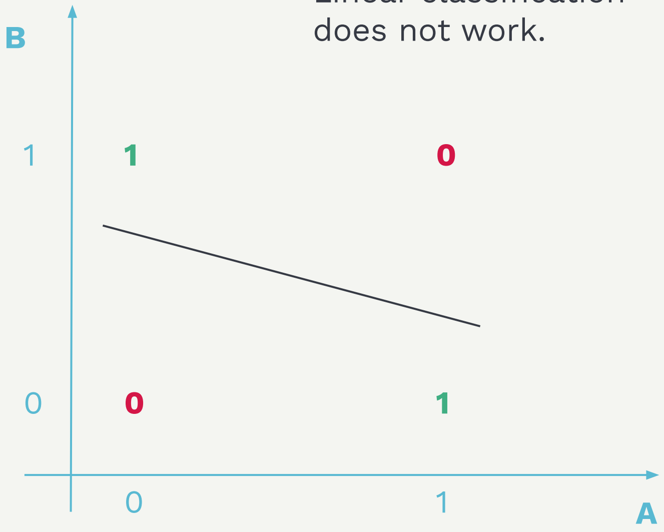
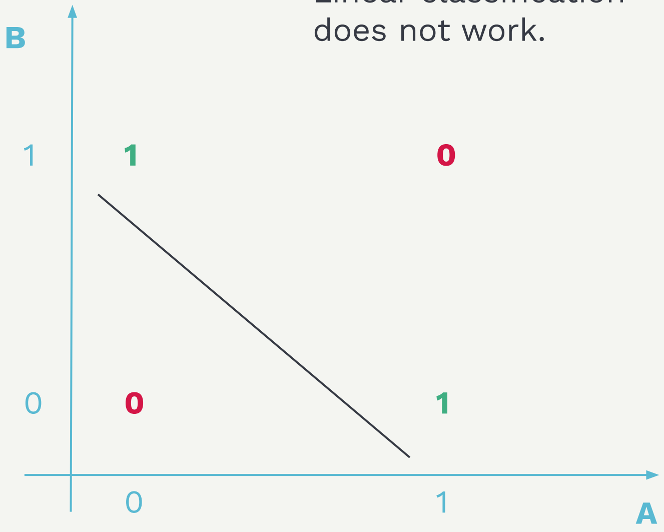


Image source: https://www.build-electronic-circuits.com/xor-gate/

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0



Linear classification does not work.

Image source: https://www.build-electronic-circuits.com/xor-gate/

Mila

How to do nonlinear learning?

→ Can we combine linear boundaries to perform nonlinear learning?

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

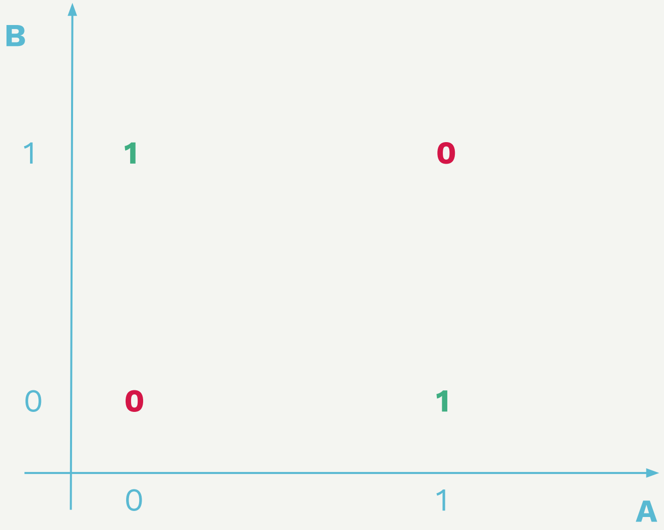


Image source: https://www.build-electronic-circuits.com/xor-gate/

Mila

Example: XOR Problem

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

New variables

C = A AND (NOT B)

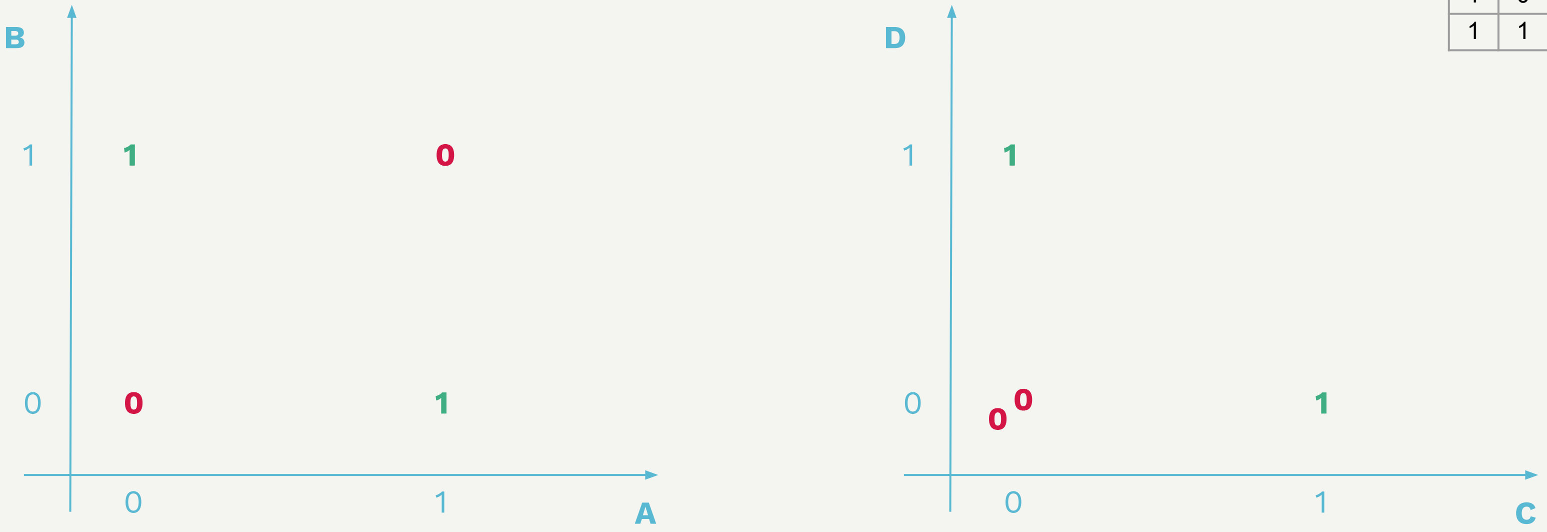
D = (NOT A) AND B

A	B	C	D
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0

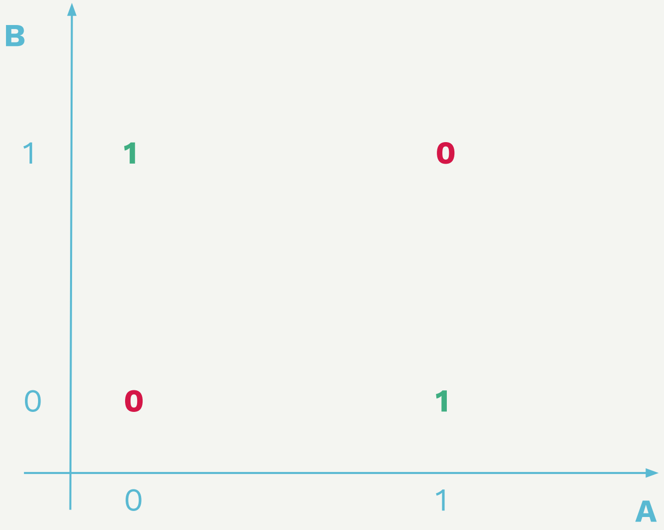
Image source: https://www.build-electronic-circuits.com/xor-gate/

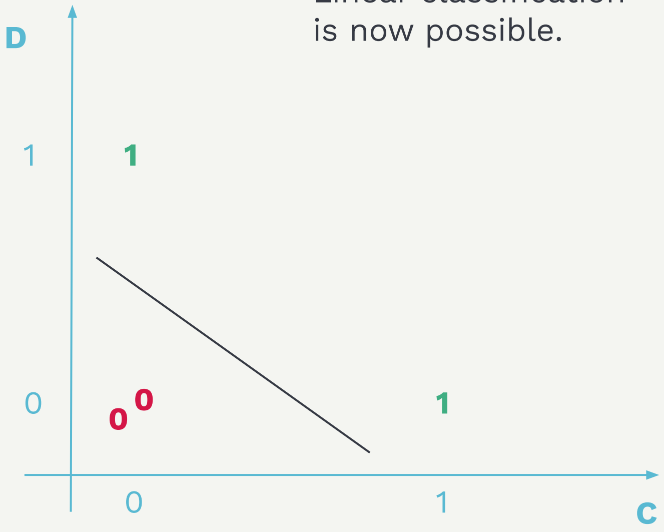
Example: XOR Problem

A	B	C	D
0	0	0	0
0	1	0	1
1	0	1	0
1	1	0	0



Example: XOR Problem

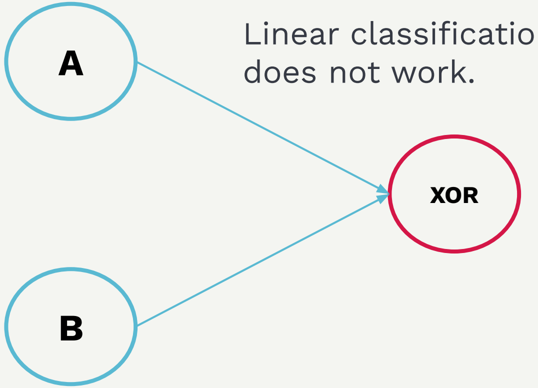




Linear classification
is now possible.

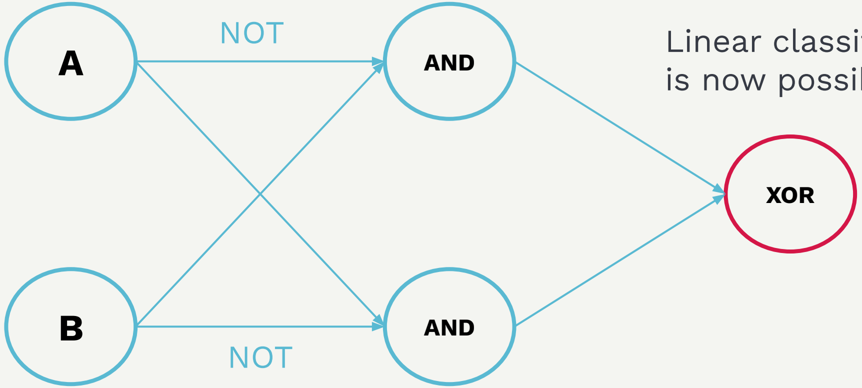
Mila

Example: XOR Problem



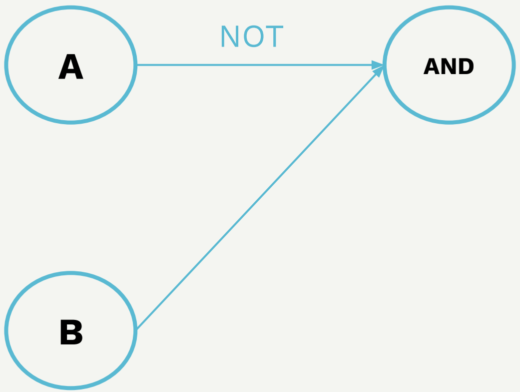
Linear classification does not work.

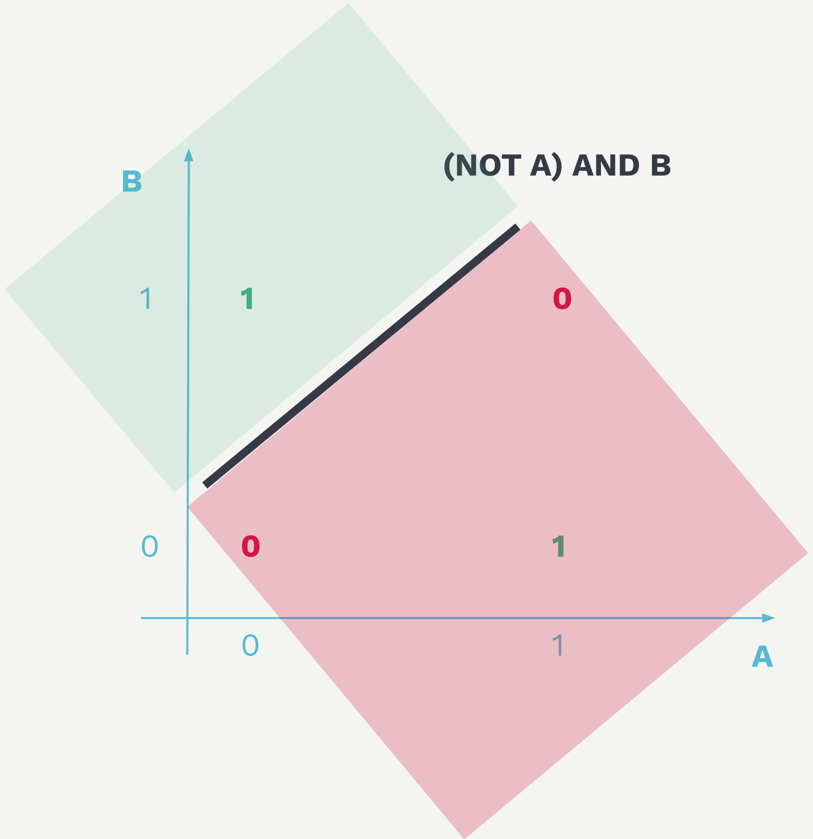
Example: XOR Problem



Linear classification
is now possible.

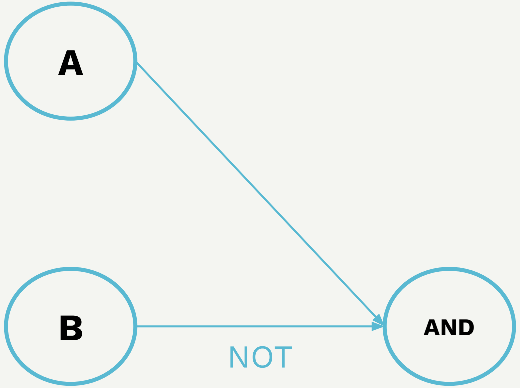
Example: XOR Problem

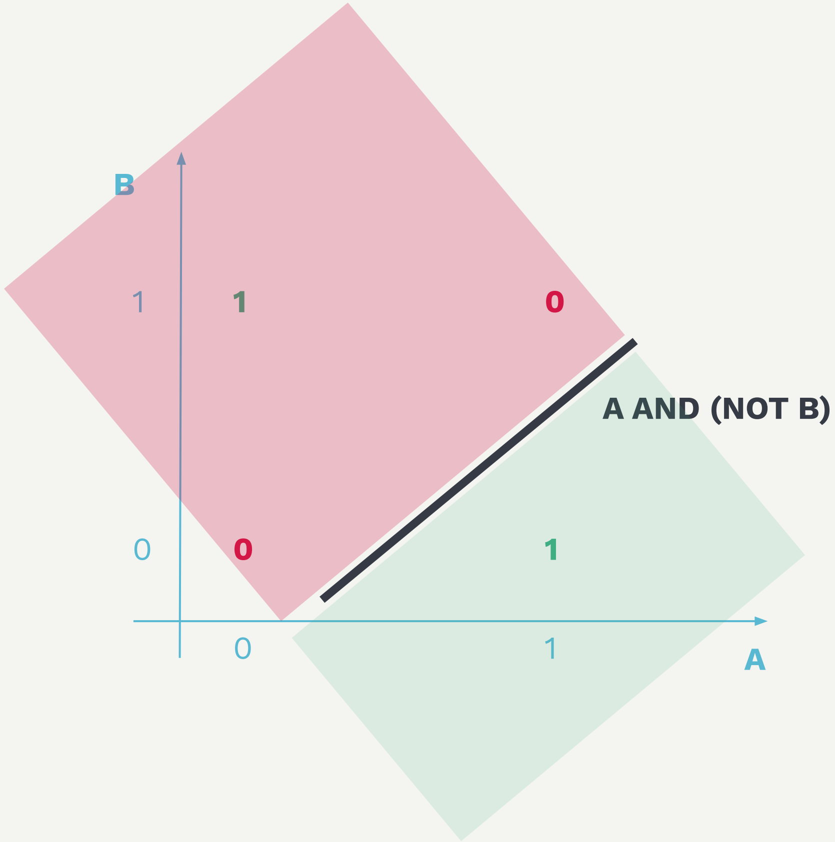




Mila

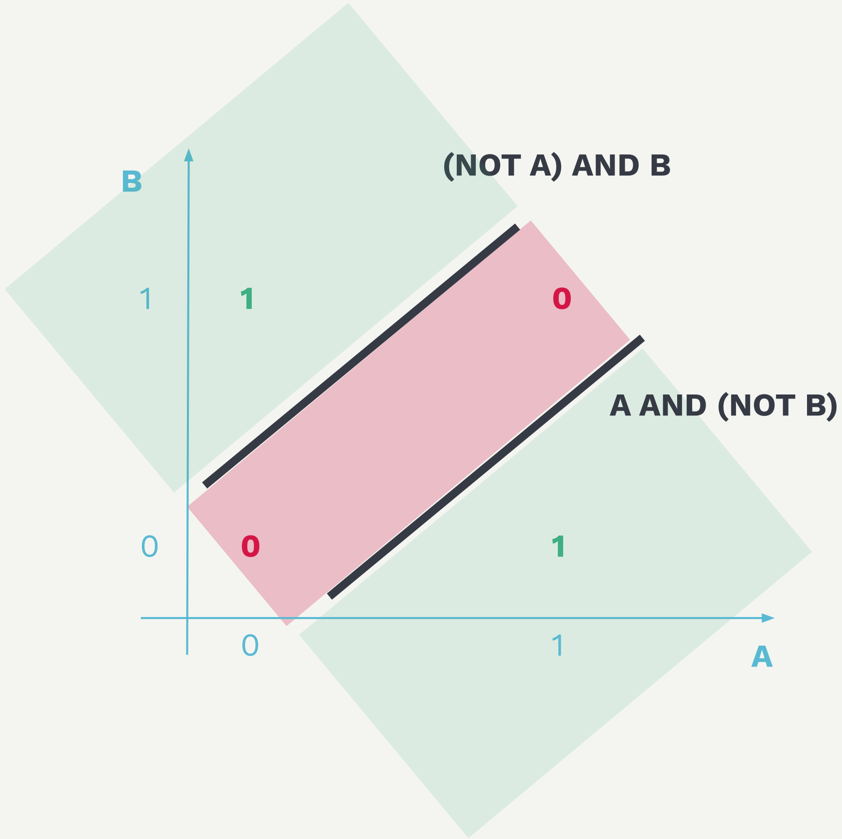
Example: XOR Problem





Example: XOR Problem

We combined two linear boundaries to form a more complicated boundary!!



Mila

How to do nonlinear learning?

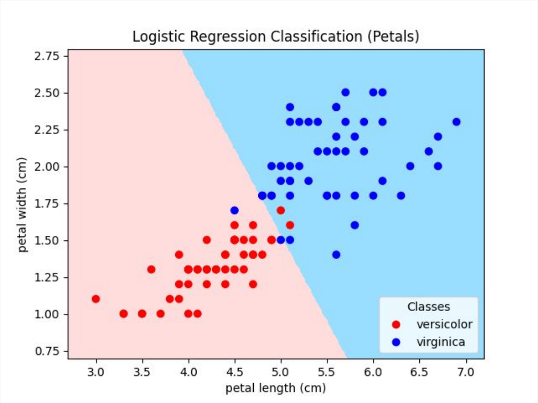
➔ Can we combine linear boundaries to perform nonlinear learning? **YES!**

How to do nonlinear learning?

➔ Can we combine linear boundaries to perform nonlinear learning? **YES!**

Recall Logistic regression

$$y = \frac{1}{1 + e^{-(wx+b)}}$$



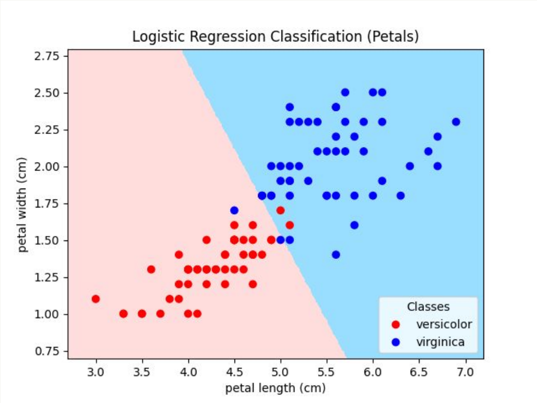
How to do nonlinear learning?

➔ Can we combine linear boundaries to perform nonlinear learning? **YES!**

Recall Logistic regression

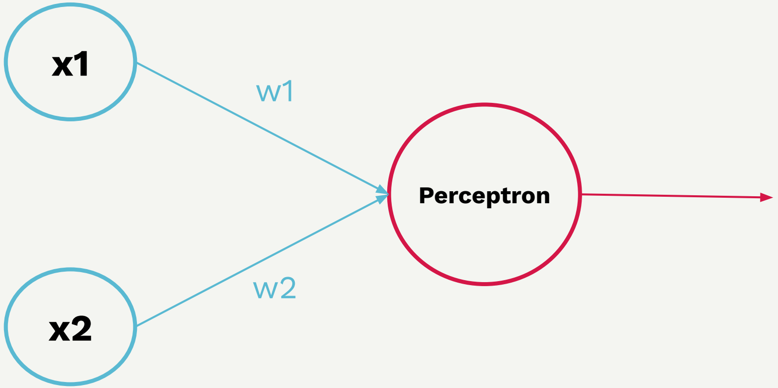
$$y = \frac{1}{1 + e^{-(wx+b)}}$$

We'll call this a 'perceptron'



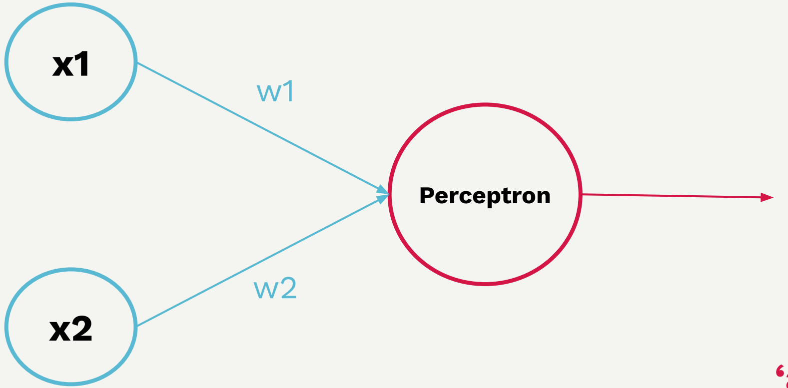
Mila

Perceptron



$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

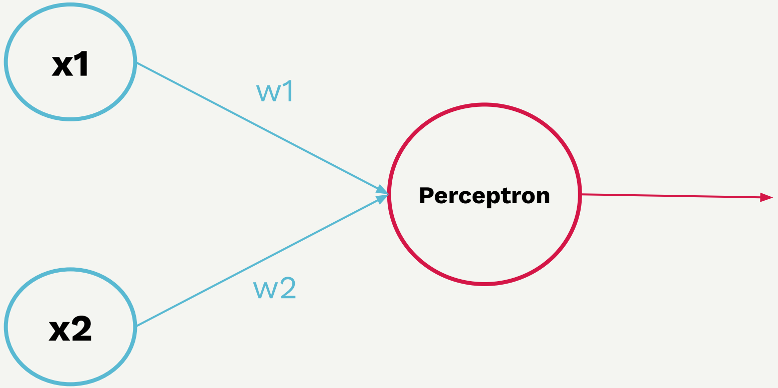
Perceptron



$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

Note: Perceptrons are defined for various 'activation functions'. We are using σ(x) for our example (hence, logistic regression).

Perceptron



$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

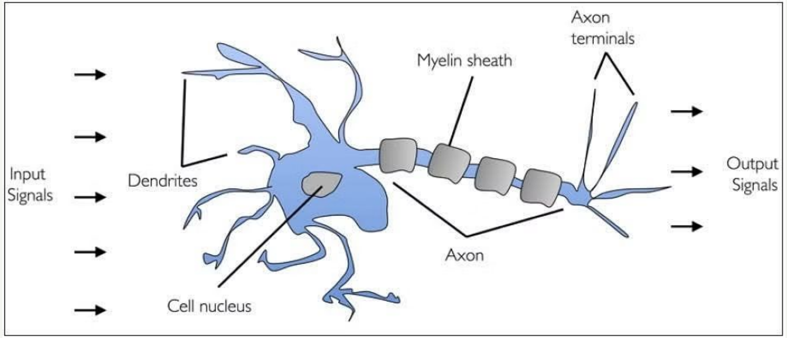
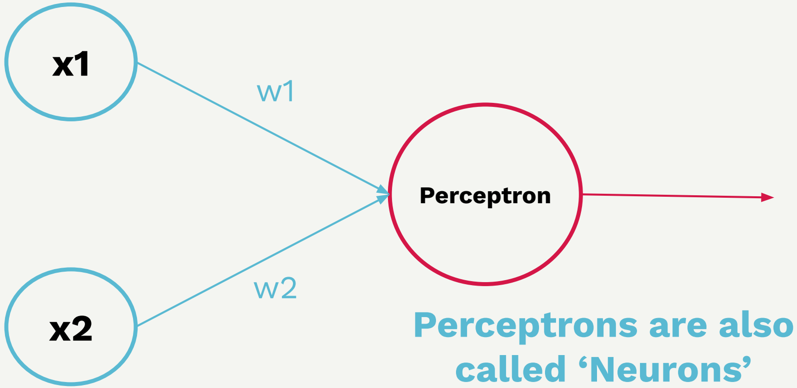


Image source: https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron

Perceptron



$$y = \frac{1}{1 + e^{-(w_1 x_1 + w_2 x_2 + b)}}$$

Perceptrons are also called 'Neurons'

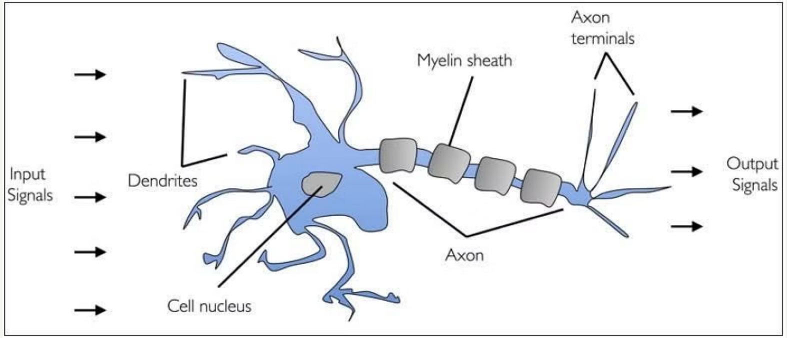


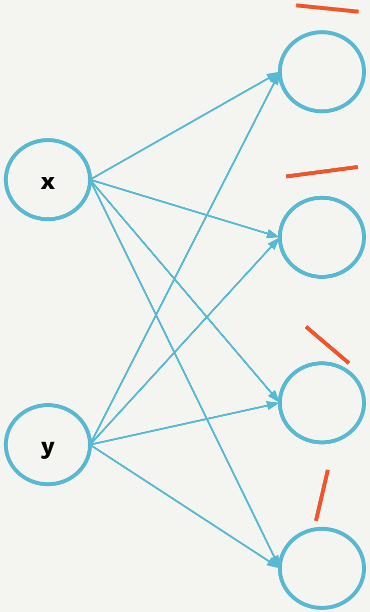
Image source: https://www.simplilearn.com/tutorials/deep-learning-tutorial/perceptron

Multi-layer Perceptron

Each **perceptron** can only create a linear boundary.
But together, they can do so much more!!

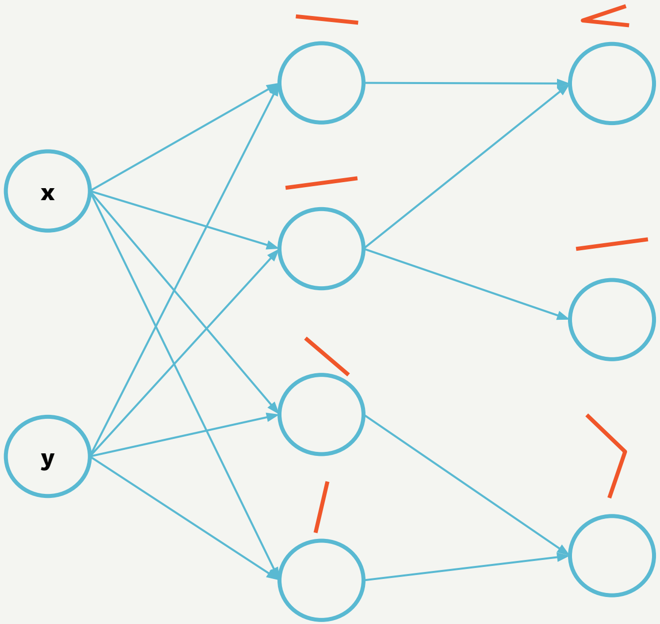
Multi-layer Perceptron

Each **perceptron** can only create a linear boundary.
But together, they can do so much more!!



Multi-layer Perceptron

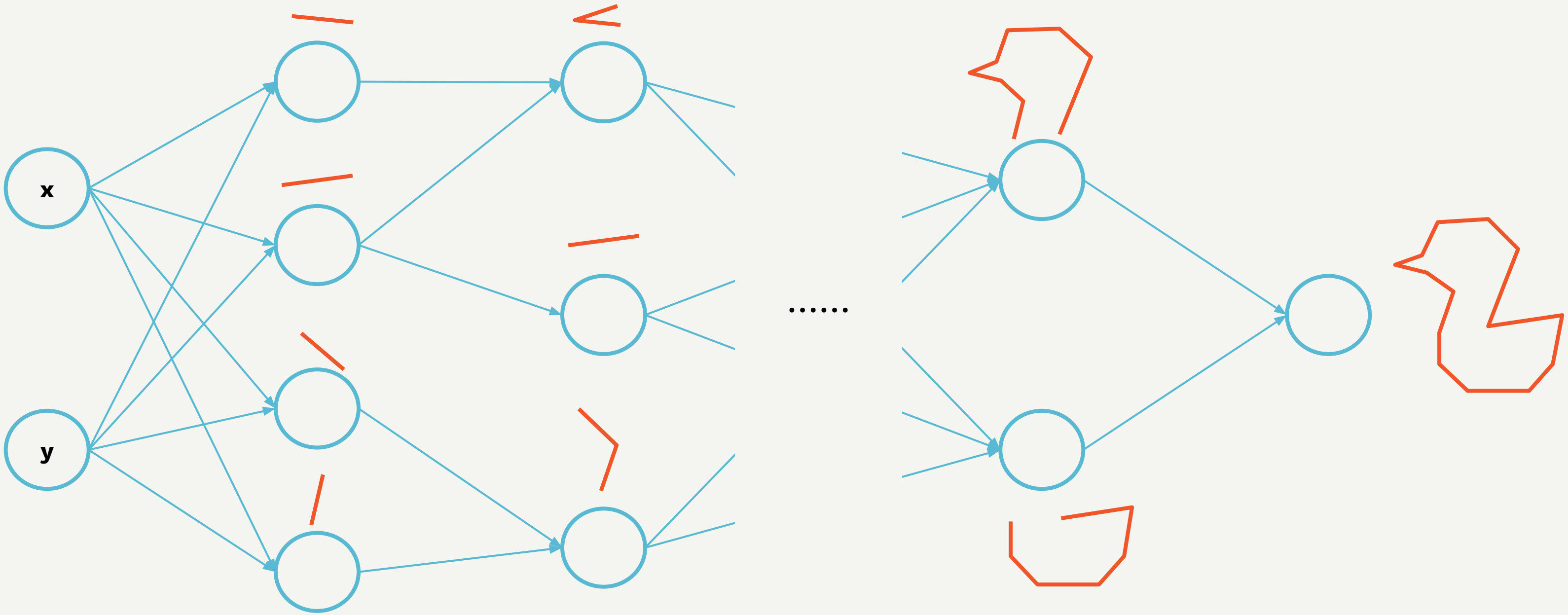
Each **perceptron** can only create a linear boundary.
But together, they can do so much more!!



Mila

Multi-layer Perceptron

Each **perceptron** can only create a linear boundary.
But together, they can do so much more!!



Multi-layer Perceptron

Each **perceptron** can only create a linear boundary.
But together, they can do so much more!!

Multi-layer perceptrons (of infinite width) are universal approximators!!

Break

Training our own MLP

Move to Colab

Mila