https://ttpoll.com/p/911058

COSC 325: Introduction to Machine Learning

Dr. Hector Santos-Villalobos

THE UNIVERSITY OF
TENNESSEE
KNOXVILLE

Lecture 06: Gradient Descent and Linear Regression

THE UNIVERSITY OF TENNESSEE KNOXVILLE

https://ttpoll.com/p/911058

Class Announcements

Homework:

We reduced homework assignments from seven to six.
The deadline for homework #2 has shifted by a week.

Lectures:

Absences: In your email's subject, include the following text "[COSC325 ABSENCE]"

Course Project:

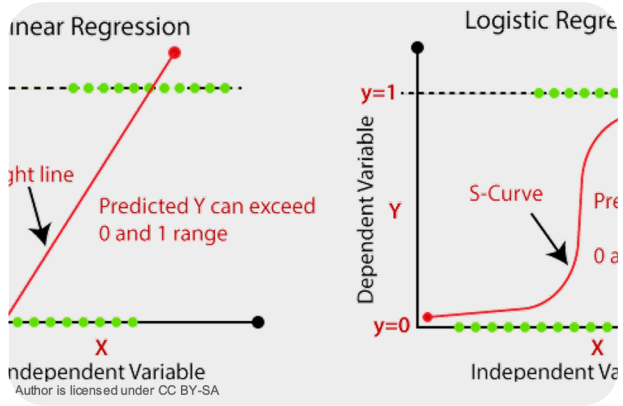
Check groups in Canvas.

https://ttpoll.com/p/911058

Today's Topics

Gradient Descent

Regression Techniques



https://ttpoll.com/p/911058

Last Lecture

- A model cannot make a better hypothesis than one provided by the sample distribution and within the limits of the learning category and technique.
- Bayes Optimal Classifier is the best solution when the data distribution f is known.
- Gradient descent
  - An iterative process to minimize model error
  - Simplicity is King
  - Needs the first derivative of the cost w.r.t the parameters

https://ttpoll.com/p/911058

Gradient Descent

https://ttpoll.com/p/911058

Pop Quiz

1 | MULTIPLE CHOICE

How familiar are you with Gradient Descent?

A. 1 - First time I hear about Gradient Descent

B. 2

C. 3

D. 4

E. 5 - I implemented and used GD before.

Gradient Descent

- First-order optimization (find minimum or maximum) technique
  - Only the first derivative is needed.
- Moves in the direction of steepest descent/accent
- It is the most popular method to minimize the error in the cost $J(\theta)$
- Types of GD
  - Batch: all samples are used for each update (i.e., iteration)
  - Stochastic (SGD): one sample per parameter update
  - Mini-Batch: a subset of the batch is used per iteration
    - Typical values: **32, 64, 128**, 256

Gradient Descent Algorithm

$X :=$ data features

$y :=$ data targets

$\theta = \theta_0$

Repeat:

$\hat{y} = h_\theta(X)$

$cost = J_\theta(y, \hat{y})$

$d\theta = \frac{\partial J_\theta(y,\hat{y})}{\partial \theta}$

$\theta := \theta - \alpha(d\theta)$

Until a fixed number of iterations or $d\theta$ very small.

Gradient Descent Algorithm

Random initialization of parameters.

Typically, very small, non-zero values.

$X :=$ data features

$y :=$ data targets

$\theta = \theta_0$

Repeat:

Use current hypothesis to obtain predictions.

$$\hat{y} = h_\theta(X)$$

Compute residuals (i.e., error between targets and predictions)

$$cost = J_\theta(y, \hat{y})$$

Partial derivative of J w.r.t. θ

$$d\theta = \frac{\partial J_\theta(y,\hat{y})}{\partial \theta}$$

Update parameters

$$\theta := \theta - \alpha(d\theta)$$

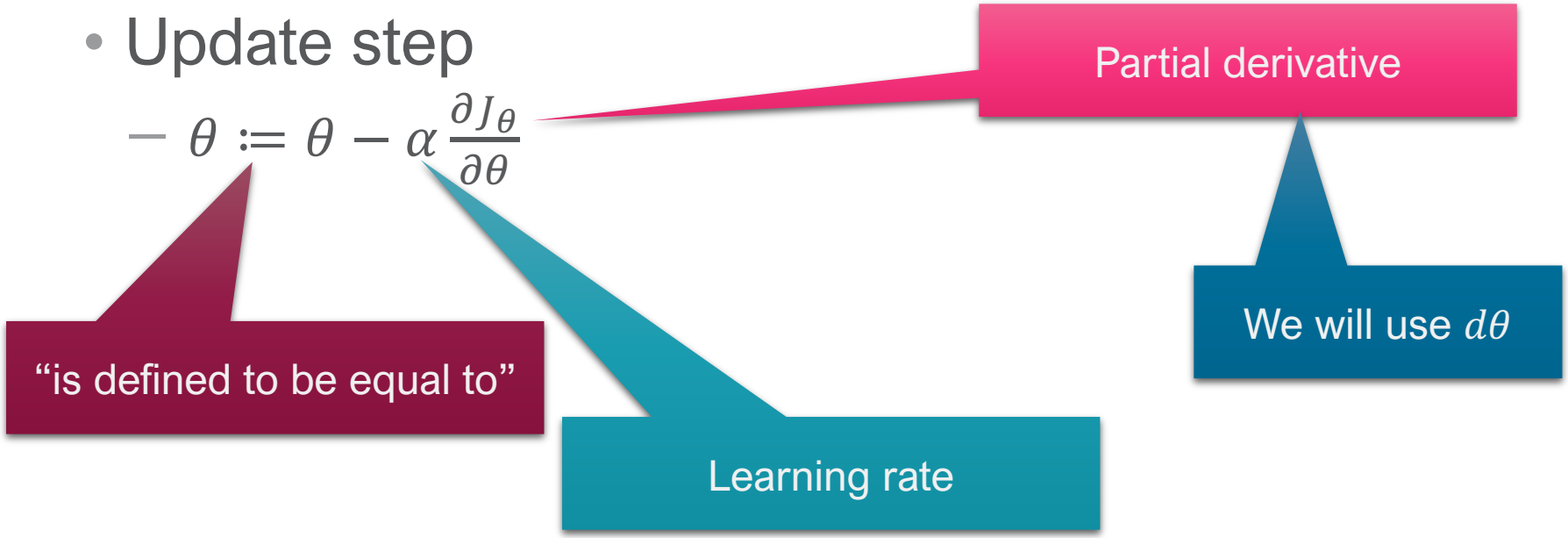
Until a fixed number of iterations or dθ very small.

Gradient Descent (GD)

- We want to find θ that minimizes J_θ

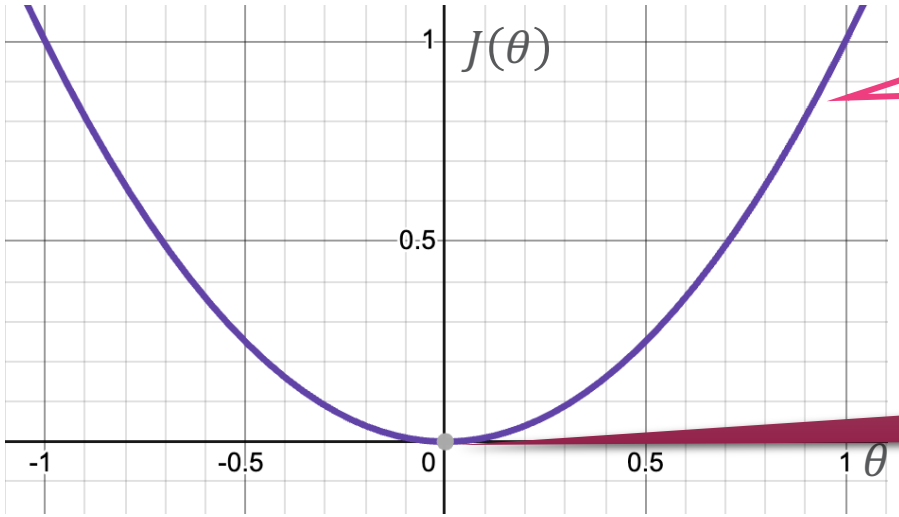
- Update step

$$-\ \theta := \theta - \alpha \frac{\partial J_\theta}{\partial \theta}$$



https://ttpoll.com/p/911058

Intuition Behind GD



$J(\theta) = \theta^2$

$d\theta = \frac{\partial J(\theta)}{\partial \theta} = 2\theta$

Zero minimizes $J(\theta)$; the solution.

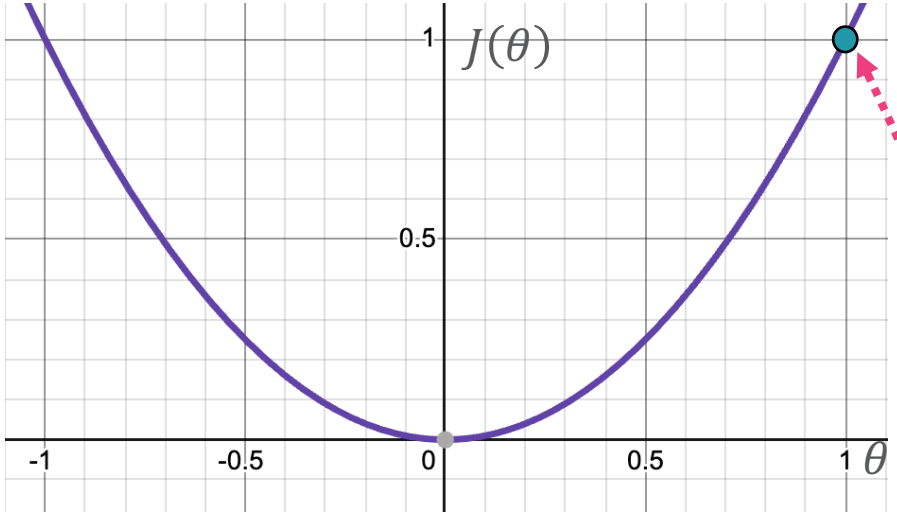
In this case, zero is boring.

Let's pick a more interesting initialization point.

You can initialize θ to zero or with a random assignment.

https://ttpoll.com/p/911058

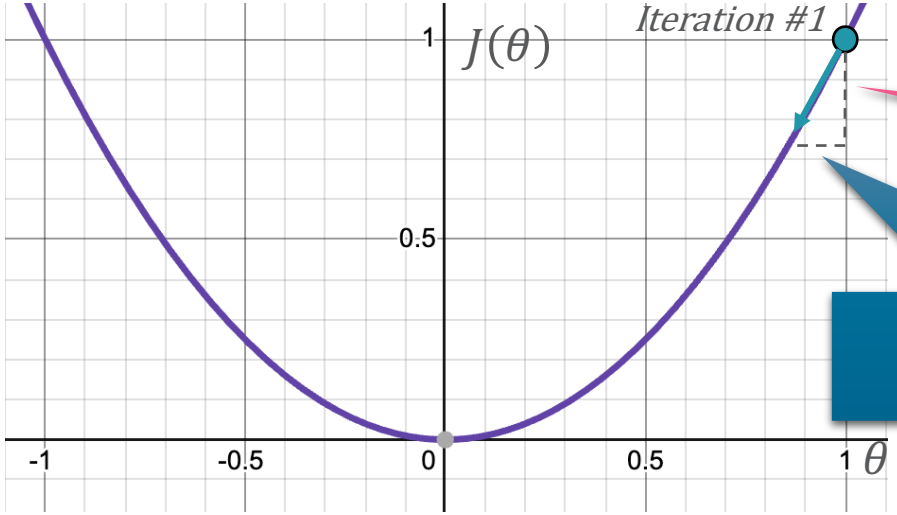
Intuition Behind GD



Let's assume $\theta = 1$.

Intuition Behind GD

Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.



Change of $J(\theta)$…

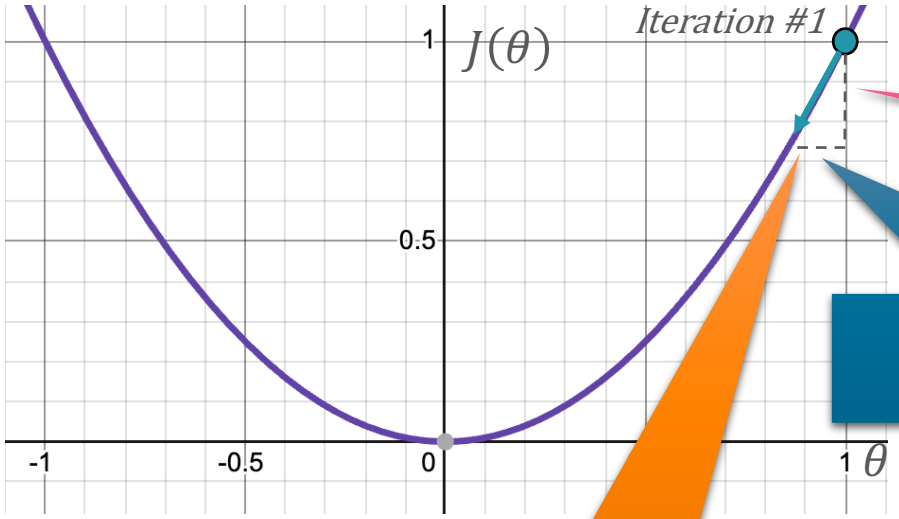
…as a consequence of a change in θ

…and α controls the magnitude of the update of w

Let's assume $w = 1$.

Intuition Behind GD

Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.



Change of $J(\theta)$…

…as a consequence of a change in θ

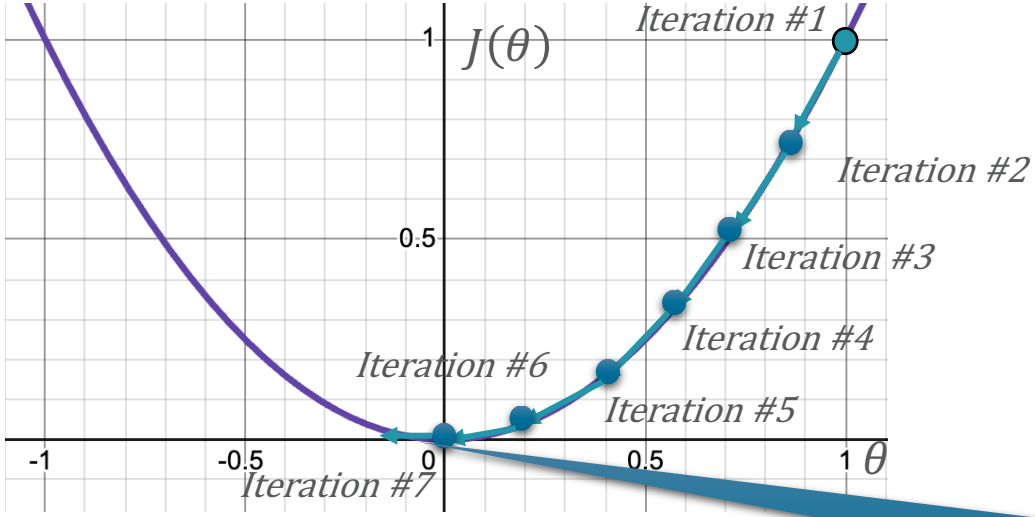
…and α controls the magnitude of the update of w

Note the positive slope

$$\theta := \theta - \alpha d\theta \quad (d\theta > 0) \Rightarrow \theta_{new} < \theta_{old}$$

https://ttpoll.com/p/911058

Intuition Behind GD

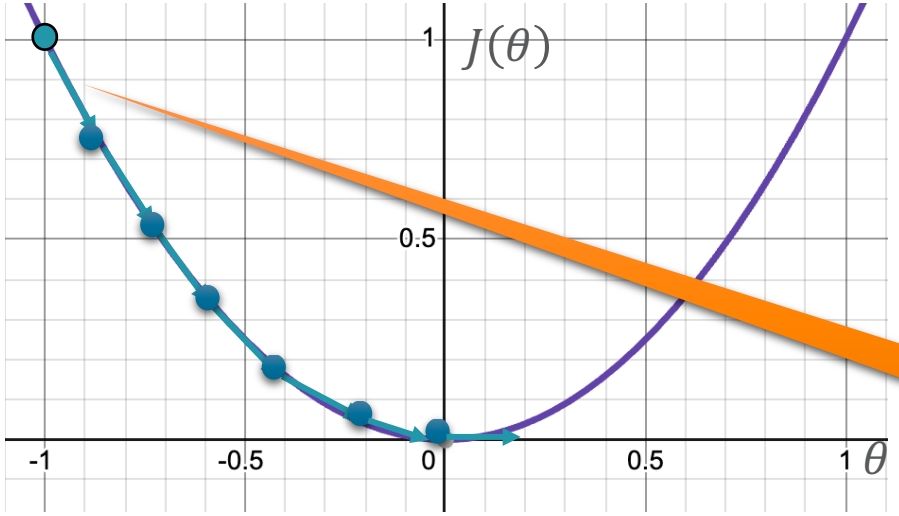


Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

Note a flat slope ($d\theta \approx 0$) at $\theta = 0$. We stop here.

https://ttpoll.com/p/911058

Intuition Behind GD



Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

Note negative slope

< 0

$\theta := \theta - \alpha d\theta$

$\theta_{new} > \theta_{old}$

Let's assume we start at $\boldsymbol{\theta = -1}$.

https://ttpoll.com/p/911058

Pop Quiz

2 | MULTIPLE CHOICE

If the average gradient for parameter Z_3, dZ_3, is negative, then the update step will make Z_3...

A. Smaller

B. Unchange

C. Larger

https://ttpoll.com/p/911058

Pop Quiz

2 | MULTIPLE CHOICE

If the average gradient for parameter Z_3, dZ_3, is negative, then the update step will make Z_3...

A. Smaller

B. Unchange

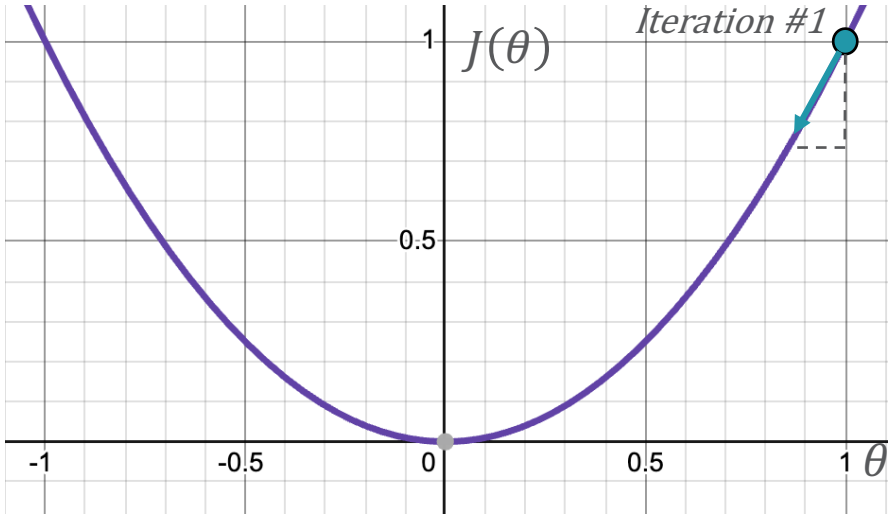
C. Larger

https://ttpoll.com/p/911058

The derivative of a function w.r.t to a variable measures the influence of that variable.

Intuition Behind GD

Let's assume $w = 1$.



Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

$$J(\theta) = \theta^2, \qquad \frac{dJ}{d\theta} = 2\theta$$

So, a change of 0.001 in θ leads to a change of 0.002 in $J(\theta)$ or a 2x change.

$$J(1.0) = 1.0^2 = 1.0,$$

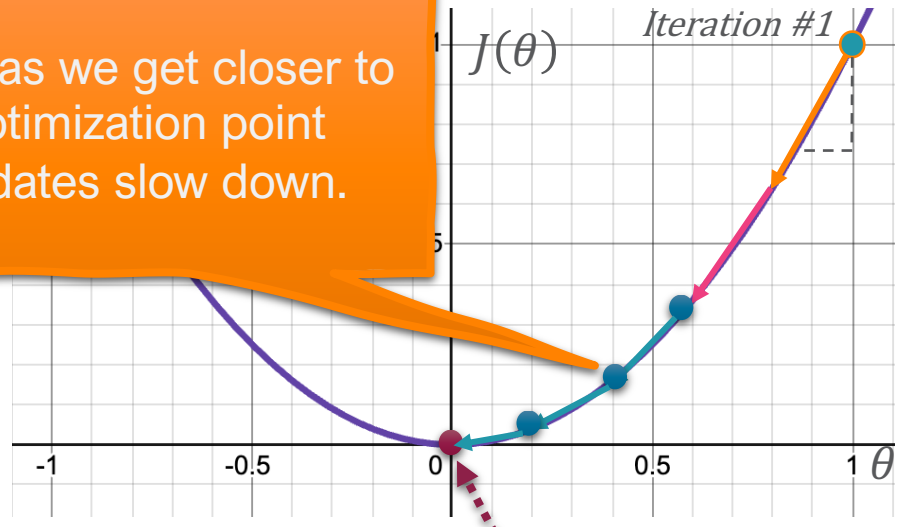
$$J(1.001) = 1.001^2 = 1.002,$$

Tiny change in θ.

https://ttpoll.com/p/911058

Intuition Behind α

Note as we get closer to optimization point updates slow down.



Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

$$J(\theta) = \theta^2, \qquad \frac{dJ}{d\theta} = 2\theta$$

If $\alpha = 0.1$:

$\theta_1 := 1 - 0.1 \times 2.0 = 1 - 0.2 = 0.8$

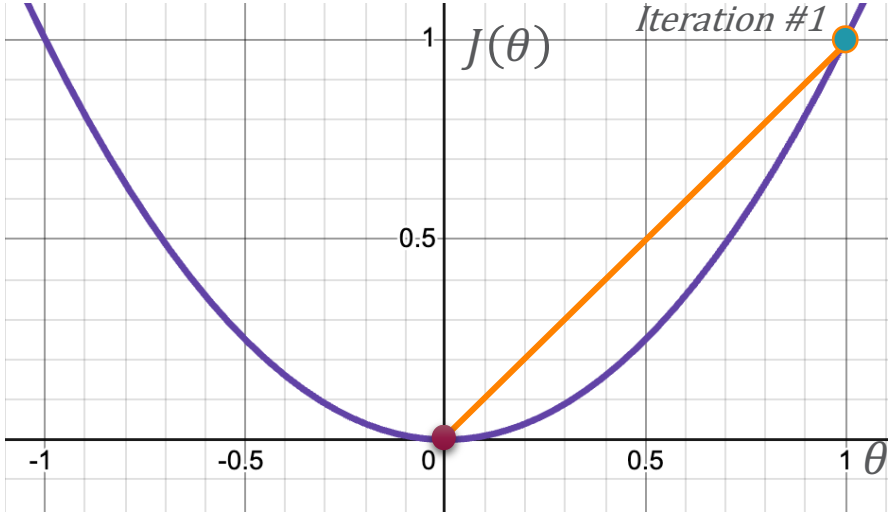
$\theta_2 := 0.8 - 0.1 \times 1.6 = 0.64$

$\vdots$

$\theta_{46} := 4 \times 10^{-5} - 0.1 \times 4 \times 10^{-5} = 3 \times 10^{-5}$

https://ttpoll.com/p/911058

Intuition Behind α



Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

$$J(\theta) = \theta^2, \qquad \frac{dJ}{d\theta} = 2\theta$$

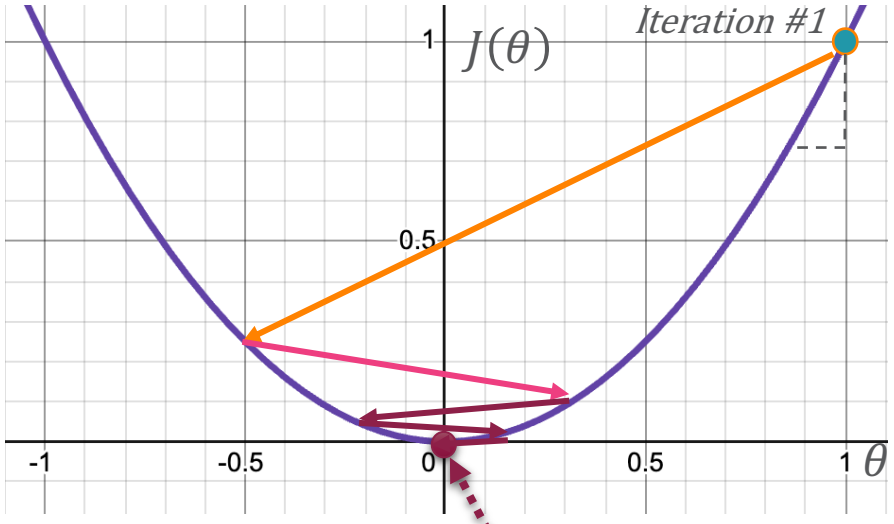
If $\alpha = 0.5$:

$\theta_1 := 1 - 0.5 \times 2.0 = 1 - 1 = 0$

$\theta_2 := 0 - 0.5 \times 0 = 0$

https://ttpoll.com/p/911058

Intuition Behind α



Repeat: $\theta := \theta - \alpha d\theta$,
until $d\theta$ is small enough.

$$J(\theta) = \theta^2, \qquad \frac{dJ}{d\theta} = 2\theta$$

If $\alpha = 0.75$:

$\theta_1 := 1 - 0.75 \times 2.0 = 1 - 1.5 = -0.5$

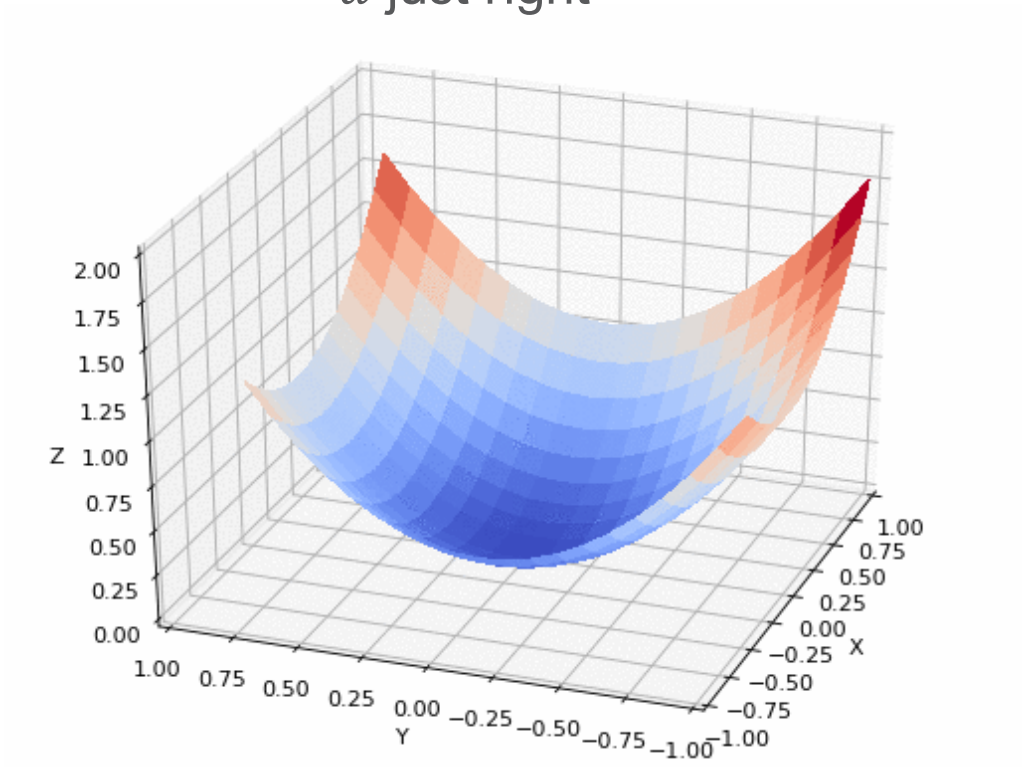
$\theta_2 := -0.5 - 0.75 \times (-1) = 0.25$

$\theta_5 := -3 \times 10^{-5} - 0.75 \times 3 \times 10^{-5} = 2 \times 10^{-5}$

https://ttpoll.com/p/911058

GD Learning Rate

α just right



α too large

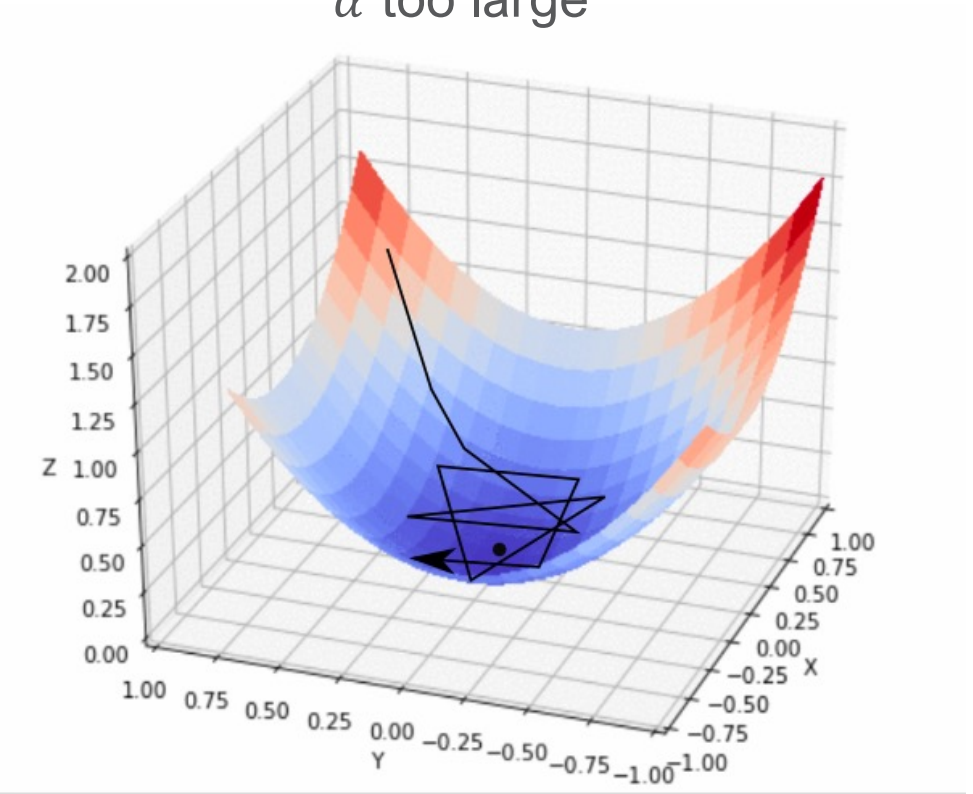


Image source: https://blog.paperspace.com/intro-to-optimization-in-deep-learning-gradient-descent/

https://ttpoll.com/p/911058

Pop Quiz

Challenges

- The method can get stuck in local minima and saddle points.
  - Sensitive to initialization point

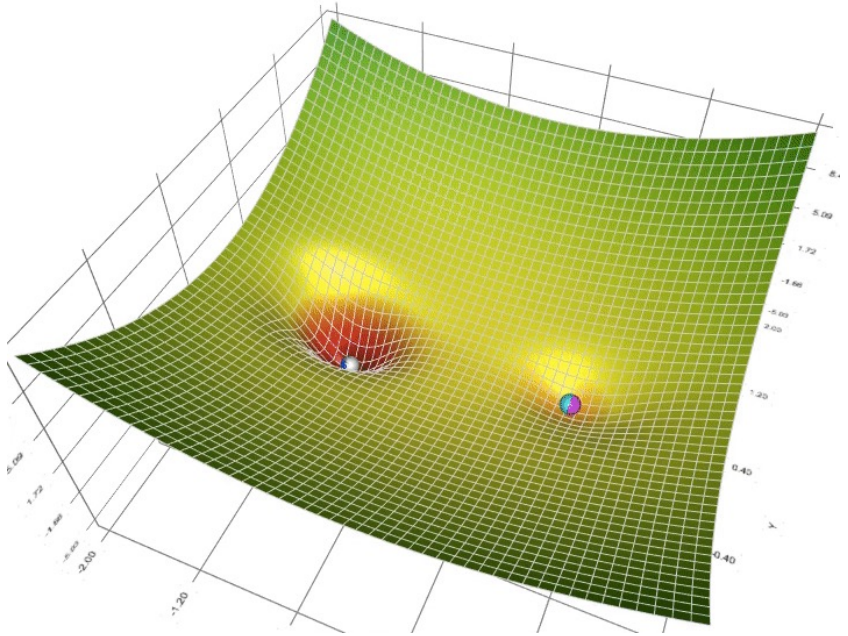


Image source: https://towardsdatascience.com/a-visual-explanation-of-gradient-descent-methods-momentum-adagrad-rmsprop-adam-f898b102325c

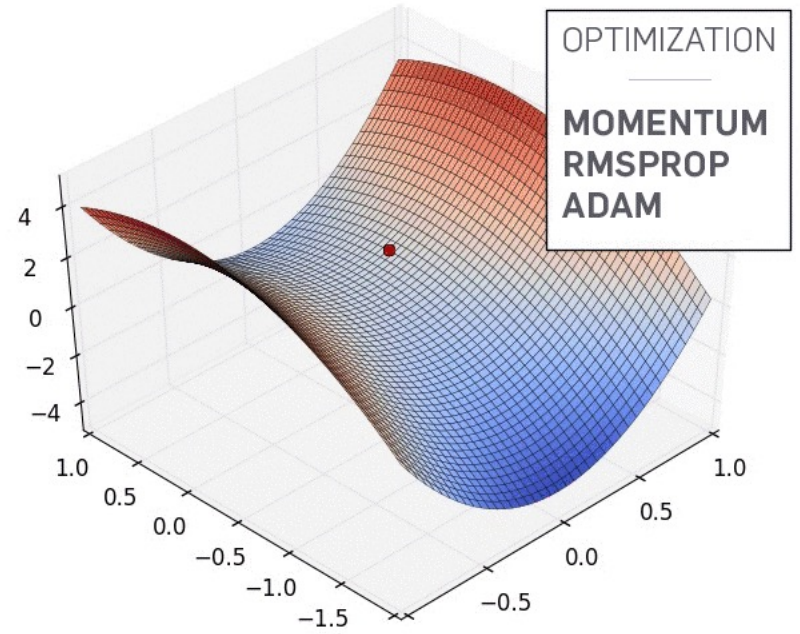


Image source: https://blog.paperspace.com/intro-to-optimization-momentum-rmsprop-adam/

Other Optimization Alternatives

- Newton Method
  - Requires 2nd derivative (i.e., Hessian)
  - Computational expensive
  - Many functions do not have a second derivative

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

- Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS)
  - Approximates the second derivative
    - Faster convergence
    - Robustness for ill-conditioned problems
  - Automated learning rate
  - It uses lots of memory, and implementation is complex

https://ttpoll.com/p/911058

Pop Quiz

3 | MULTIPLE CHOICE

If we have a function f(x) with parameter x, and f(1) = 2.00, and f(1.01)=1.5; what is the derivative of f(x) at x=1.0?

A. -50

B. -5

C. 5

D. 50

https://ttpoll.com/p/911058

Pop Quiz

3 | MULTIPLE CHOICE

If we have a function f(x) with parameter x, and f(1) = 2.00, and f(1.01)=1.5; what is the derivative of f(x) at x=1.0?

A. -50

B. -5

C. 5

D. 50

https://ttpoll.com/p/911058

Notebook Time

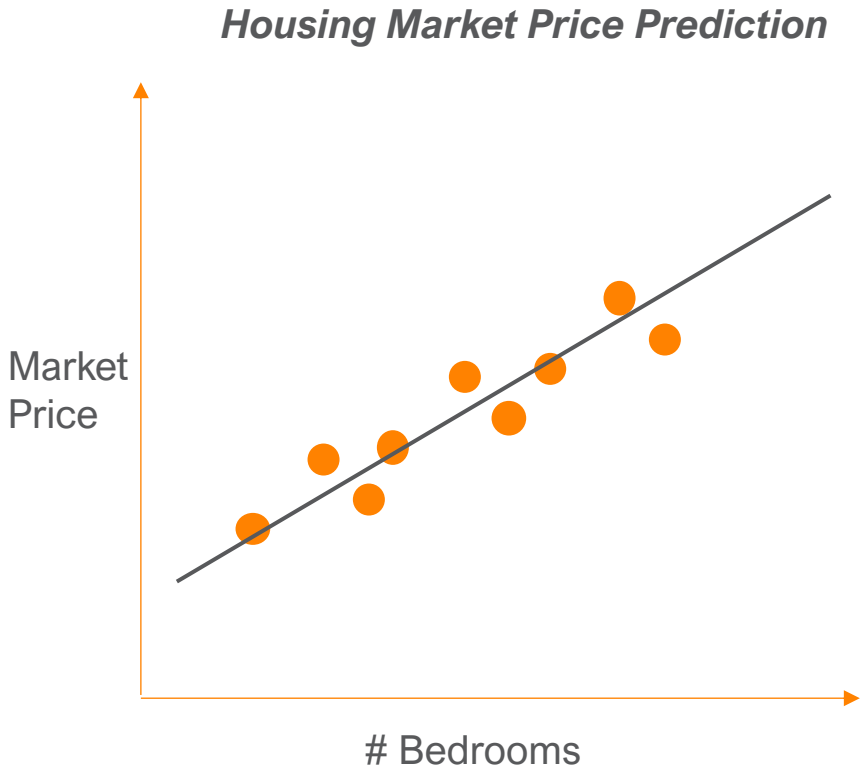
https://ttpoll.com/p/911058

Linear Regression

Linear Regression

- Supervised learning category
- Model Purpose: Establishes a linear relationship between the independent variable (i.e., input feature vector) x and the dependent variable (i.e., target/label) y.

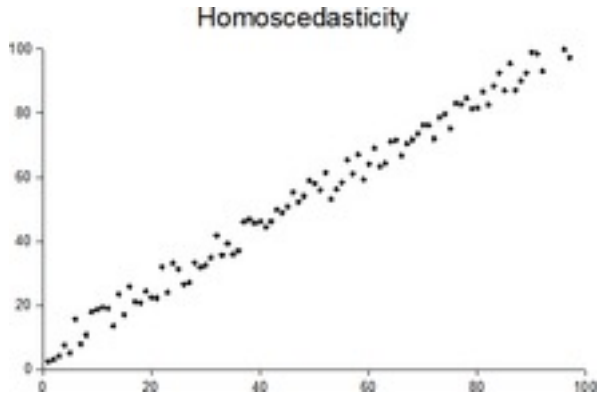
$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_m x_m$$

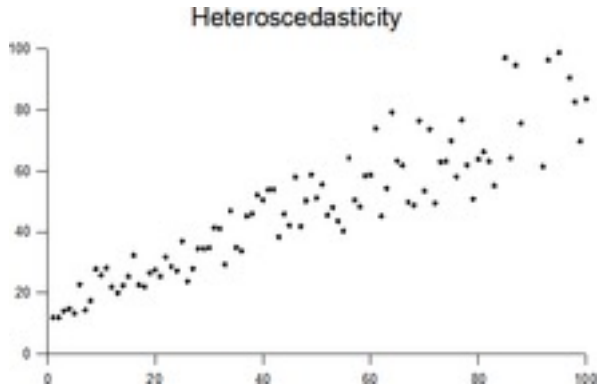


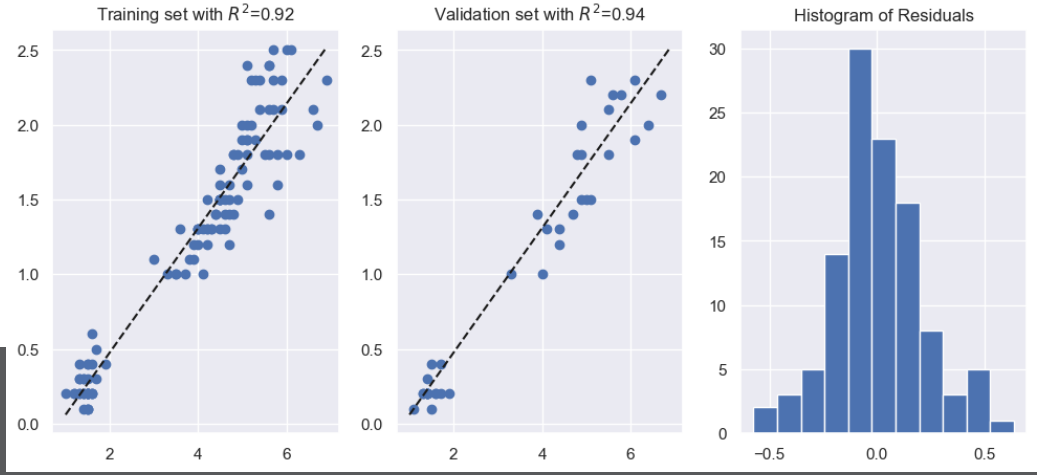
https://ttpoll.com/p/911058

Assumptions

- Linear relationship between features and targets.
- Observations are independent of each other.
- The variance of errors is consistent across all levels of the independent variables (i.e., Homoscedasticity).
- The errors between targets and predictions are normally distributed (i.e., normality).







Linear Regression

- For $m = 1$:

$$y^{(i)} = \theta_1 x^{(i)} + \theta_0$$

θs are the parameters to learn

For $m = n_x$:

$$y^{(i)} = \theta_0 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \cdots + \theta_m x_m^{(i)}$$

$$y^{(i)} = x^{(i)T}\theta$$

Shape: $(1,1)$

Shape: $(1, m)$

?

Linear Regression

- For $m = 1$:

$$y^{(i)} = \theta_1 x^{(i)} + \theta_0$$

θs are the parameters to learn

For $m = n_x$:

$$y^{(i)} = \theta_0 1 + \theta_1 x_1^{(i)} + \theta_2 x_2^{(i)} + \cdots + \theta_m x_m^{(i)}$$

$$y^{(i)} = [1, x^{(i)T}]\theta$$

Shape: $(1,1)$

Shape: $(1, m+1)$

Shape: $(m+1, 1)$

Strengths

- Simplicity and Interpretability:
  - Linear regression is easy to understand and implement
  - The model's coefficients provide insights into the relationship between each feature and the target, allowing for straightforward interpretability.
- Computational Efficiency:
  - Linear regression is computationally inexpensive, making it suitable for large datasets with many features.
- Closed-form Solution:
  - The parameters of a linear regression model can be estimated using a closed-form solution via the normal equation, eliminating the need for iterative methods like gradient descent (in most cases).
- Versatility:
  - It can be extended to polynomial regression by adding polynomial features, allowing the model to capture non-linear relationships.

Weaknesses

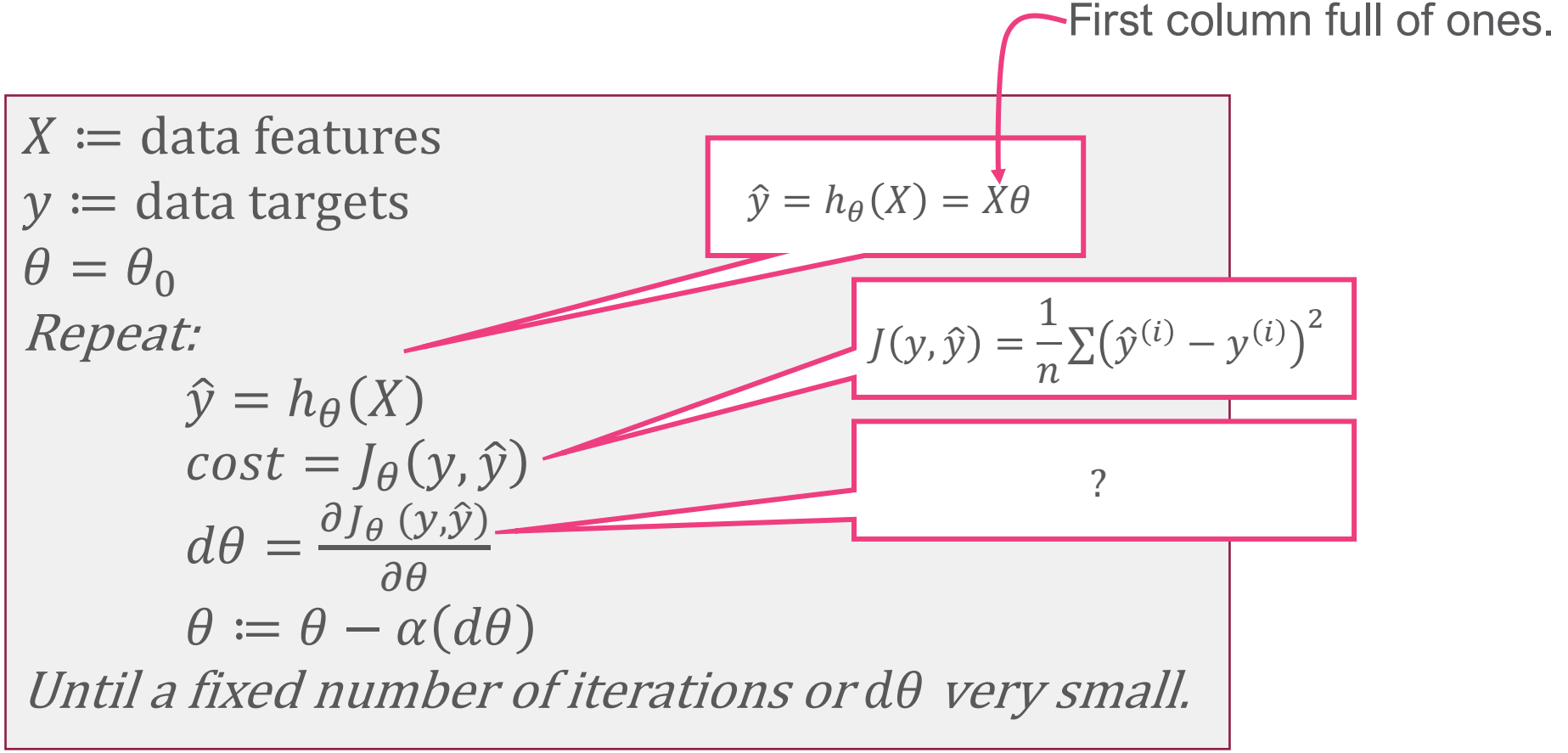
- Limited Flexibility:
  - Linear regression can only capture linear relationships. It may perform poorly when the true relationship between features and target is non-linear.
- Sensitive to Outliers:
  - Linear regression is highly sensitive to outliers, which can disproportionately affect the model's performance and skew the results.
- Assumptions Limitations:
  - The model's effectiveness depends on fulfilling its assumptions. Violations can lead to inaccurate predictions.
- Overfitting with High-Dimensional Data:
  - In the presence of many features, especially when the number of features approaches the number of observations, linear regression can overfit, leading to poor generalization of new data.

https://ttpoll.com/p/911058

Gradient Descent for a Linear Regressor

Computing $dZ = d\theta$

Gradient Descent Algorithm



https://ttpoll.com/p/911058

Computing the derivative of the cost J_θ w.r.t. to parameters θ

- Recall: the derivative of θ measures the influence of θ on the expected error J_θ.
- How do we compute $\frac{\partial J_\theta(y,\hat{y})}{\partial \theta}$?
  - Computation graphs
  - Calculus Chain Rule
- Need to know
  - $\hat{y} = h_\theta(X)$
  - Loss function $\mathcal{L}_\theta$

$= X\theta = \theta_0 + x_1\theta_1 + \cdots + x_m\theta_m$

$= (\hat{y} - y)^2$

If we change any of these two, the derivative equation changes.

https://ttpoll.com/p/911058

Chain Rule Refresher

You need to identify the inner and outer functions of a composite function.

- What is it?

$$\frac{d}{dx}[f(g(x))] = \frac{d}{dx}[f(g(x))]\frac{d}{dx}[g(x)]$$

$$u = g(x)$$

$$\frac{d}{dx}[f(g(x))] = \frac{d}{du}[f(u)]\frac{d}{dx}[g(x)] = \frac{df}{du}\frac{du}{dx}$$

If we can break g(x) further, we can add that inner function to the chain.

$$\frac{d}{dx}[f(g(h(x)))] = \frac{d}{dx}[f(g(x))]\frac{d}{dx}[g(h(x))]\frac{d}{dx}[h(x)] = \frac{df}{du}\frac{du}{db}\frac{db}{dx}$$

$$u = g(x) \text{ and } b = h(x)$$

Chain Rule Refresher

- Example

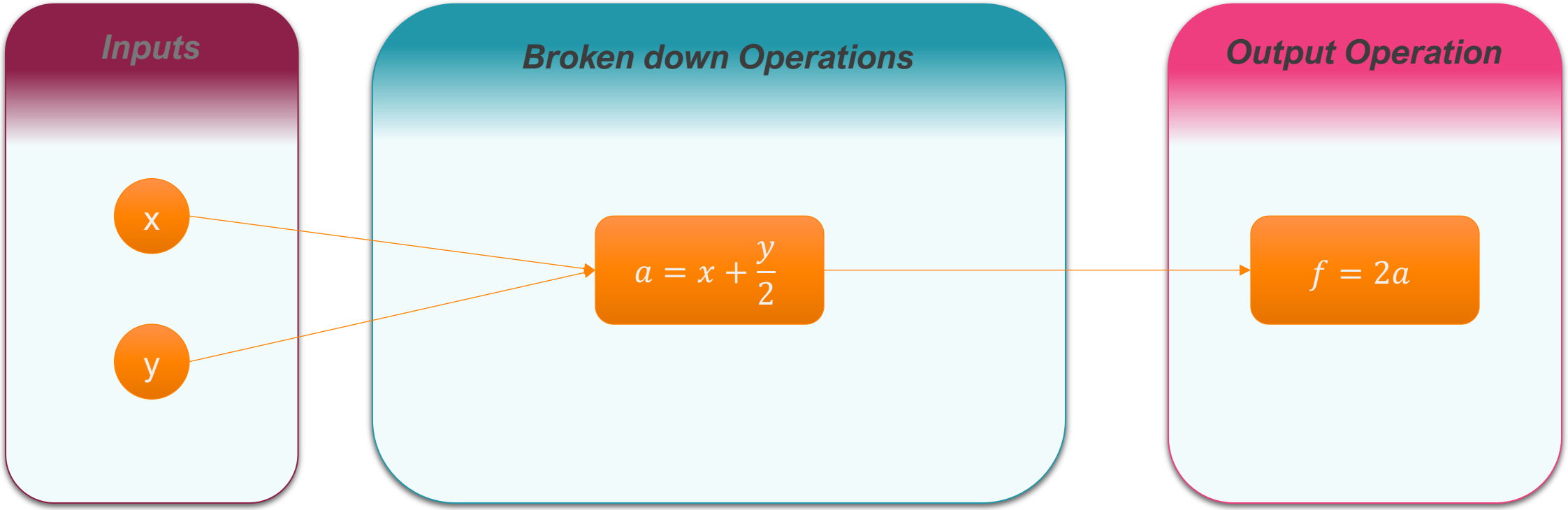
$$f(x) = \frac{1}{(1-x)^2} \qquad g(x) = u = 1 - x \qquad f(u) = \frac{1}{u^2}$$

$$\frac{du}{dx} = -1 \qquad \frac{df}{du} = \frac{-2}{u^3}$$

$$\frac{df}{dx} = \frac{df}{du}\frac{du}{dx} = \frac{-2}{(1-x)^3} \times -1 = \frac{2}{(1+x)^3}$$

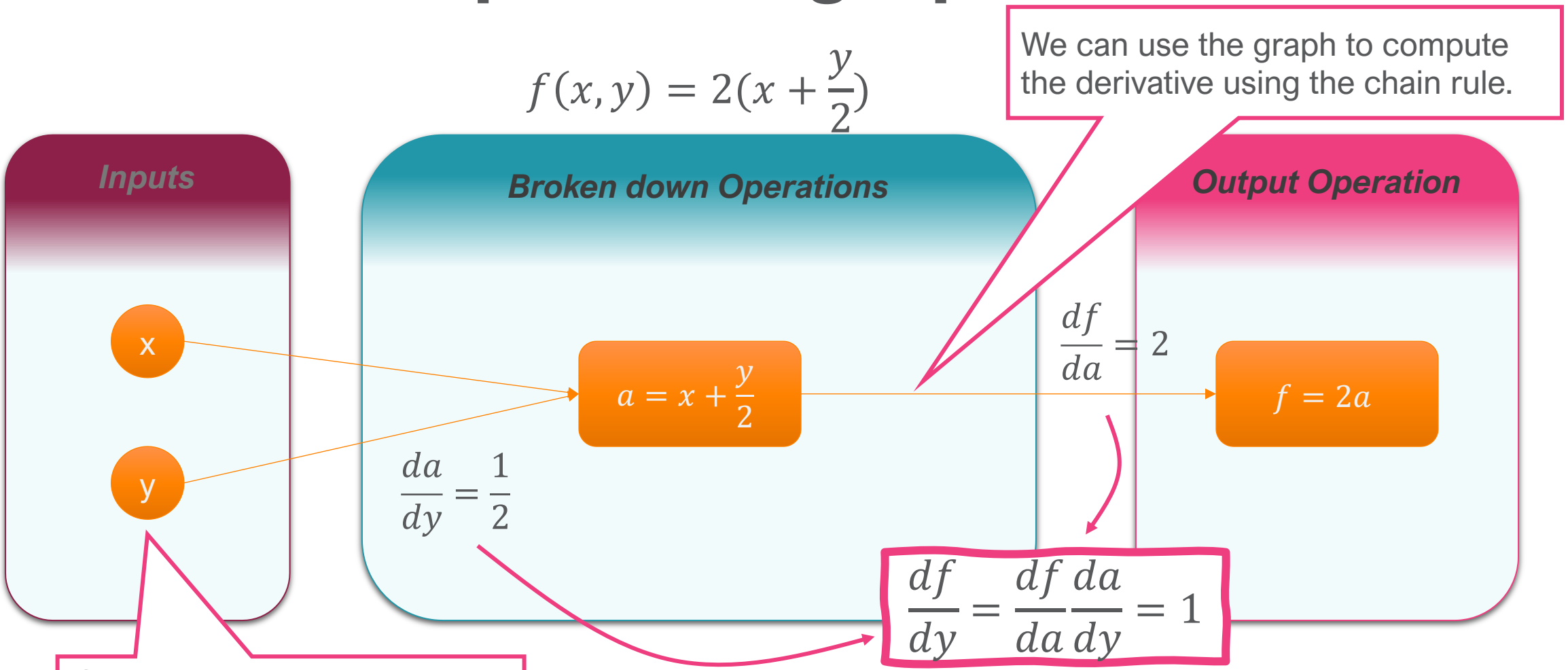
What is a computation graph?

$$f(x, y) = 2(x + \frac{y}{2})$$



What is a computation graph?

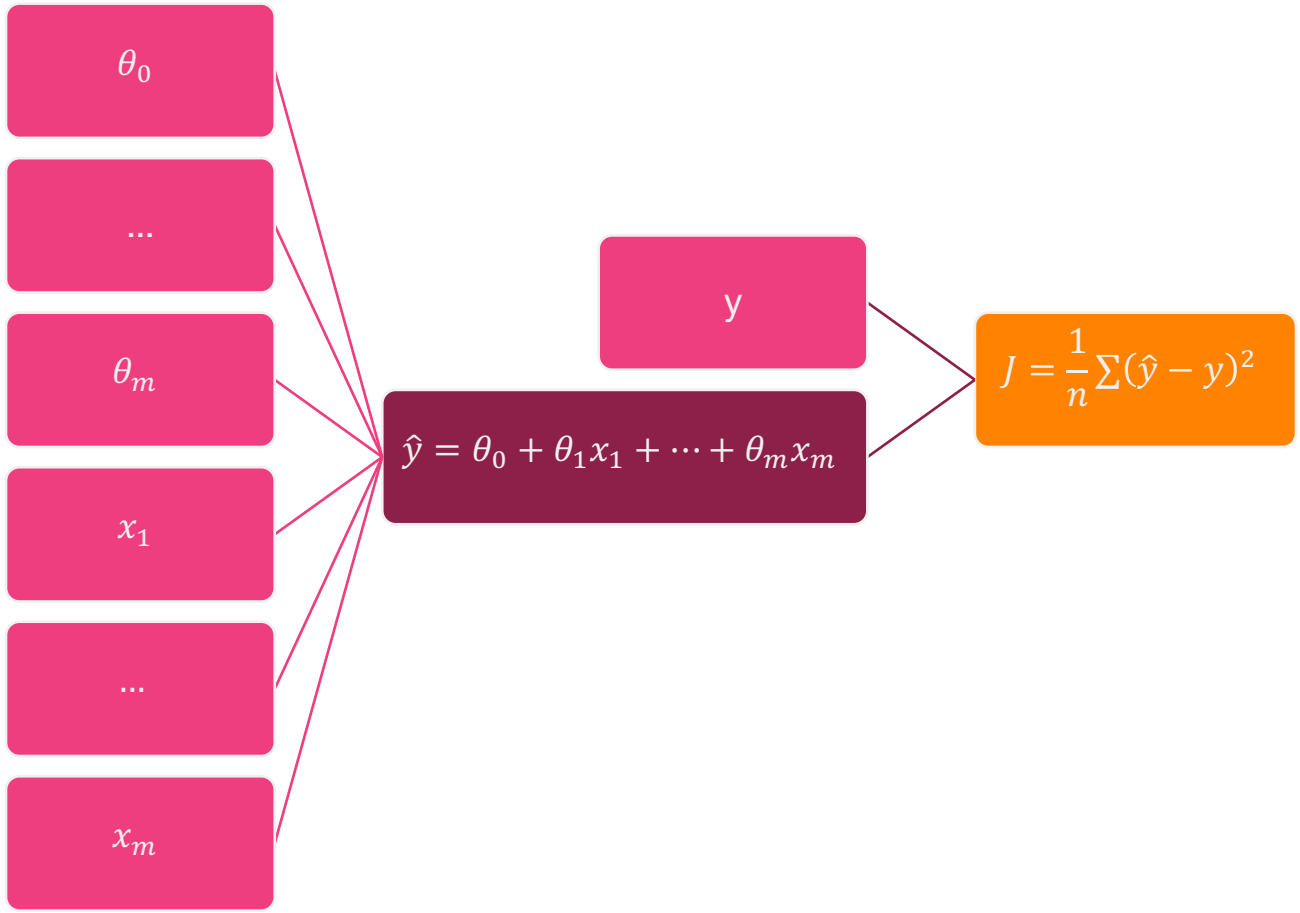
$$f(x, y) = 2(x + \frac{y}{2})$$



https://ttpoll.com/p/911058

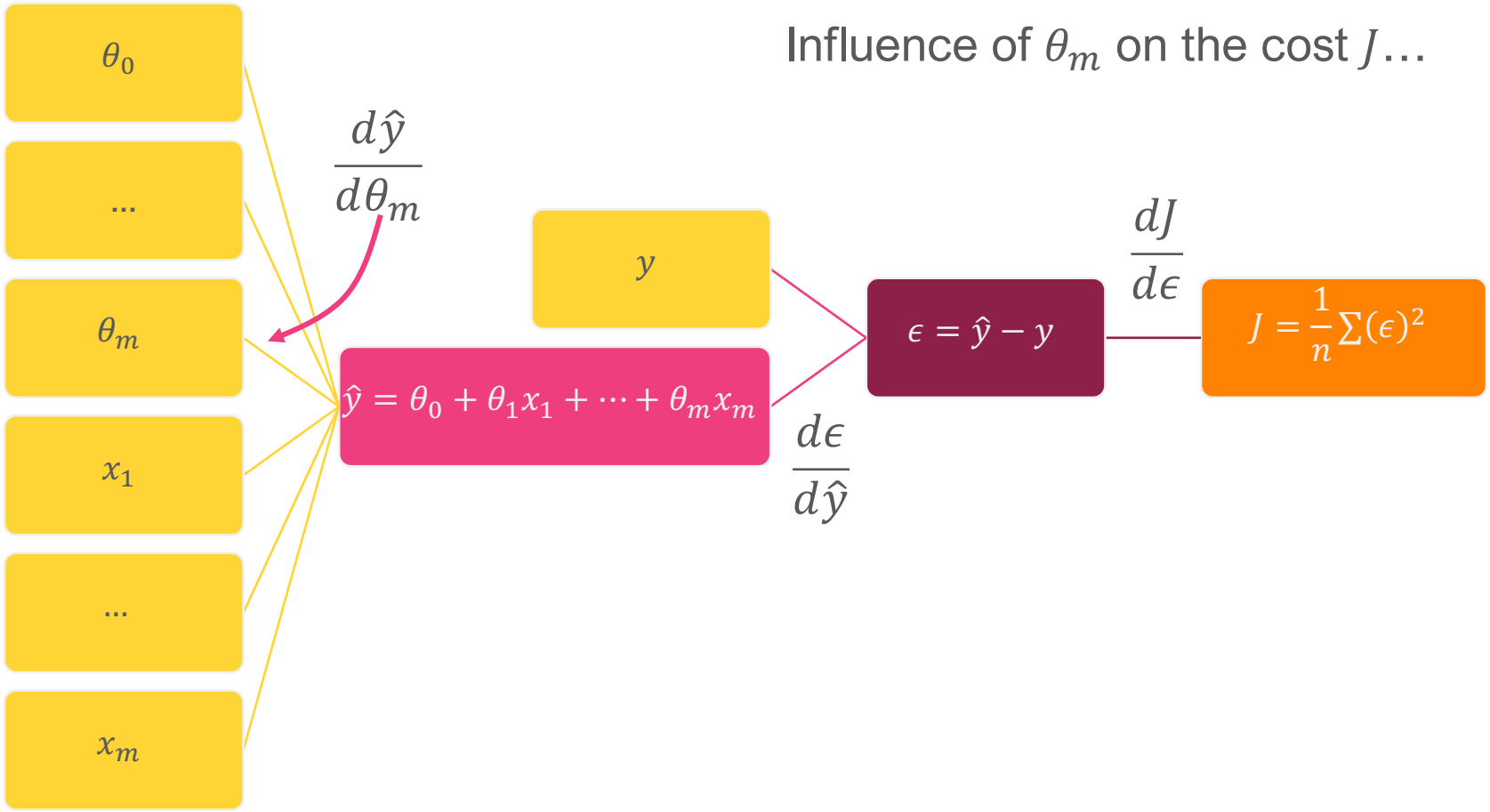
Computation Graph for Linear Regression

Assumes MSE as the cost function.



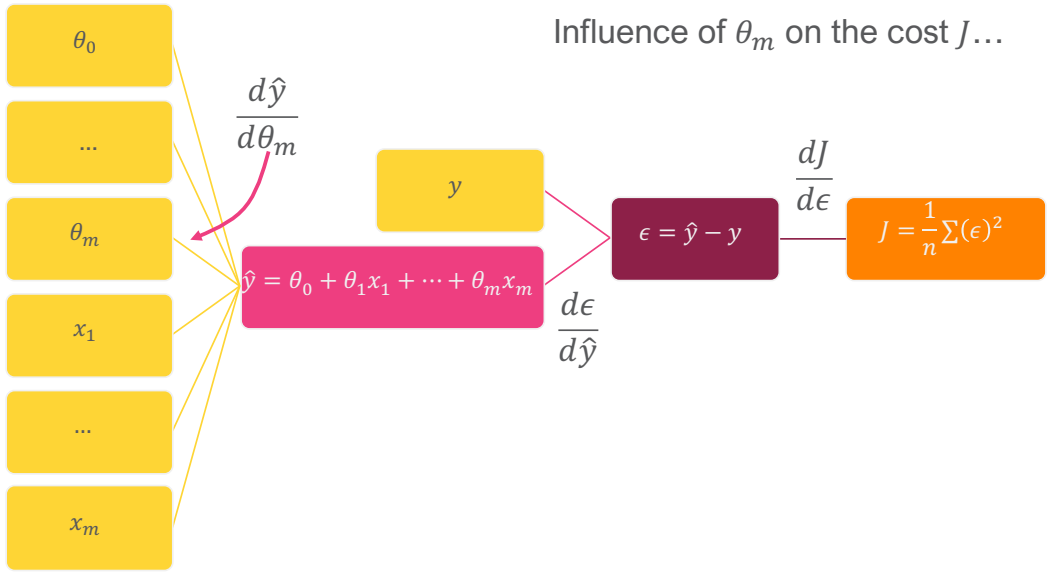
Computation Graph for Linear Regression

Assumes MSE as the cost function.



Computation Graph for Linear Regression

Assumes MSE as the cost function.



$$\frac{dJ}{d\epsilon} = \frac{1}{n}\sum\left(2\epsilon^{(i)}\right) = \frac{2}{n}\sum\left(\epsilon^{(i)}\right)$$

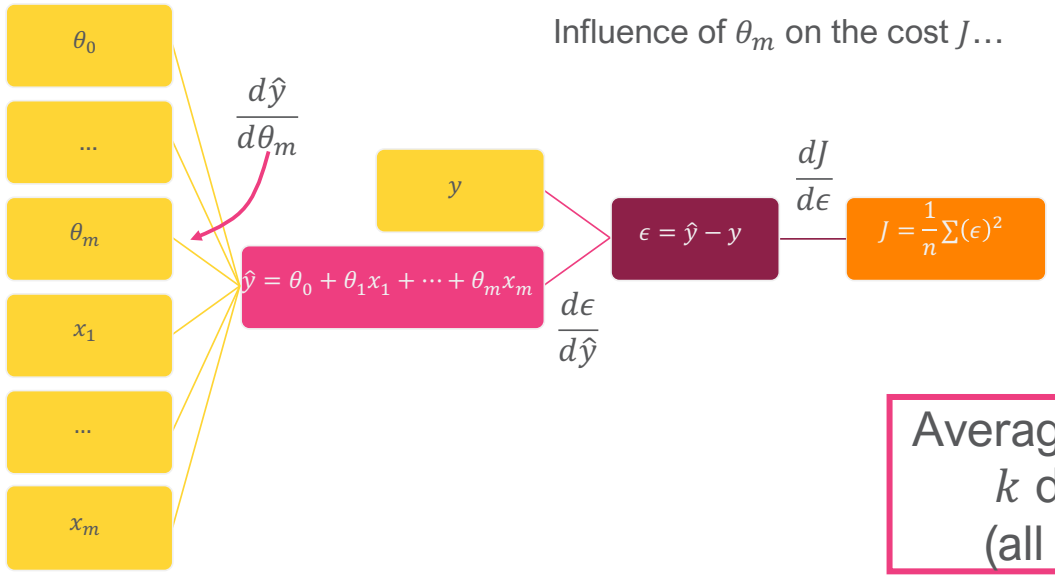
$$\frac{d\epsilon}{d\hat{y}} = 1$$

$$\frac{d\hat{y}}{d\theta_m} = x_m^{(i)}$$

$$\frac{dJ}{d\theta_m} = \frac{dJ}{d\epsilon}\frac{\mathrm{d}\epsilon}{d\hat{y}}\frac{d\hat{y}}{d\theta_m} = \left(\frac{2}{n}\sum\left(\epsilon^{(i)}\right)\right)(1)\left(x_m^{(i)}\right) = \frac{2}{n}\sum\left(\hat{y}^{(i)} - y^{(i)}\right)x_m^{(i)}$$

Computation Graph for Linear Regression

Assumes MSE as the cost function.



Influence of θ_m on the cost J…

$$\frac{dJ}{d\theta_m} = \frac{2}{n}\sum\left(\hat{y}^{(i)} - y^{(i)}\right)x_m^{(i)}$$

$$\frac{dJ}{d\theta_k} = \frac{2}{n}\sum\left(\hat{y}^{(i)} - y^{(i)}\right)x_k^{(i)}$$

Average parameter k derivative (all samples)

Vectorize

$$\frac{dJ}{d\theta} = \frac{2}{n}\left(X^T(\hat{y} - y)\right)$$

Exact Solution vs. Gradient Descent

$$\theta = (X^T X)^{-1} X^T y$$

- This gives an exact solution (modulo numerical inaccuracy for inverting the matrix)
- Gradient descent gives you progressively better solutions and eventually gets to an optimum

Exact Solution vs. Gradient Descent

- Is calculating the exact solution more efficient?
- Running gradient descent for one step is $O(n * m)$ time with a small time constant
  - Running it for k iterations is $O(k * n * m)$
- The closed form solutions requires:
  - Constructing $X^T X$, which takes $O(n * m^2)$
  - Inversion, which takes $O(n * m^2)$
  - Multiplications, which take $O(m^3)$
  - Overall run time: $O(m^3 + n * m^2)$
  - Note: In most cases, $N > D$, so this is dominated by $O(D\ N)$

https://ttpoll.com/p/911058

Exact Solution vs. Gradient Descent

- GD Solution: $O(n * m)$
- Close solution: $O(m^3 + n * m^2)$
- Guidance:
  - Typically $n > m$
  - Will you need to run more than m iterations of gradient descent?
    - Yes? Close form solution may be faster
    - No? Gradient descent may be faster
  - For m $\leq$ 100, it's probably faster to do a closed form solution
  - For m $\geq$ 10000, it's probably faster to do gradient descent
  - For in between...it's unclear

https://ttpoll.com/p/911058

Notebook Time

Review

- Gradient descent algorithm and concepts
  - Convexity, learning rate, saddle point, global vs. local minimum, etc.
- Derivatives measure the influence of a variable on the function output.
- Computational graphs and the chain rule
- Linear regression
  - Close form vs. GD solutions
  - Python implementation.

Next Lectures

- Regression Techniques
  - Polynomial
  - Logistic
- Logistic classification
- Cross Entropy Loss

What: UTK Machine Learning Club

Where: **MK 525**

When: **Tuesday** at **5:00**
(including today)

Who: Any experience level

Everyone is welcome to the first meeting of ML club today. Whether you are a beginner looking to learn from our intro to ML lesson series, experienced practitioner who wants to learn from and discuss with other enthusiasts in our reading groups, or you just want to hear from our industry guest speakers and seminars, utkML can help you scratch your machine learning itch!