Introduction to AI

Reminder

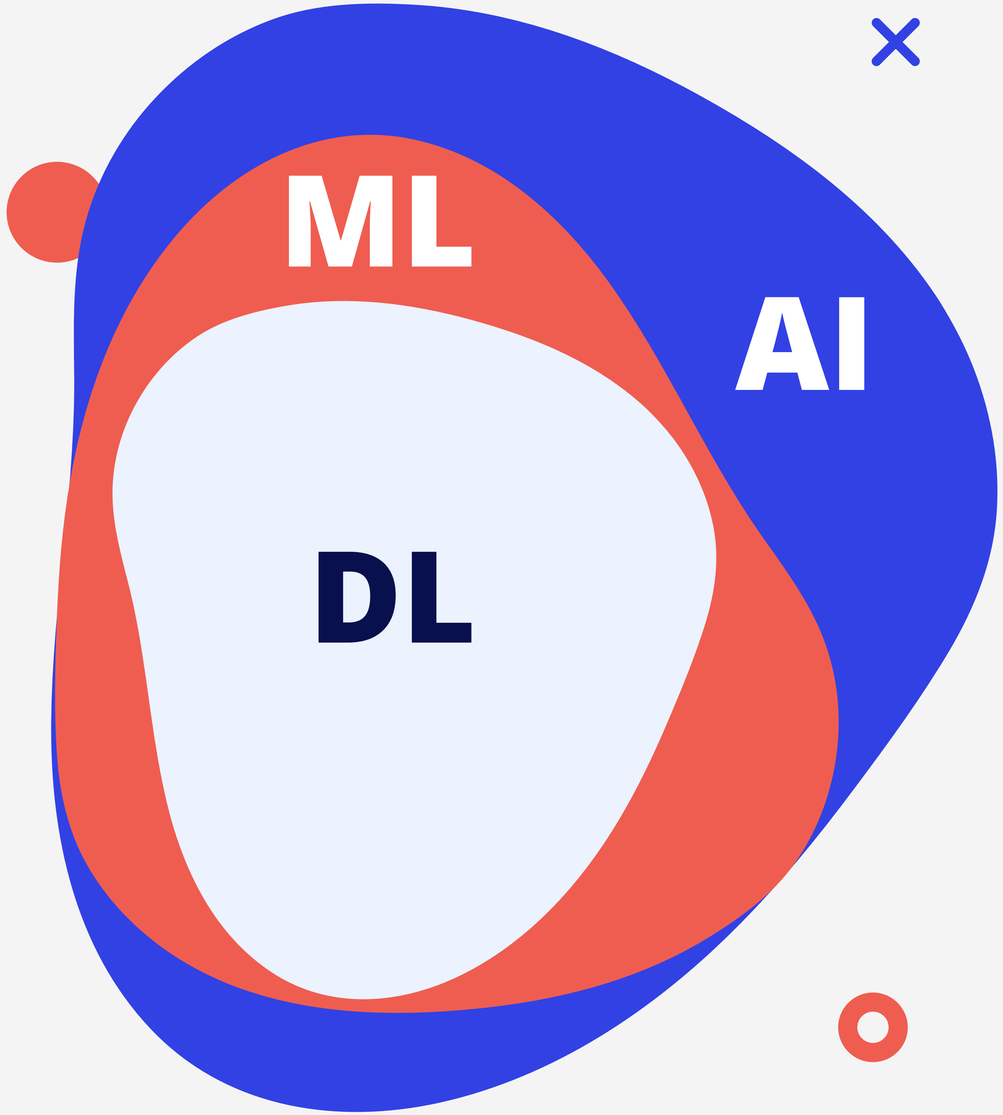
What is AI?

Artificial intelligence (AI) is a wide-ranging **branch of computer science** concerned with building smart machines capable of performing tasks that typically require **human intelligence**.”

-Wikipedia

Hierarchy

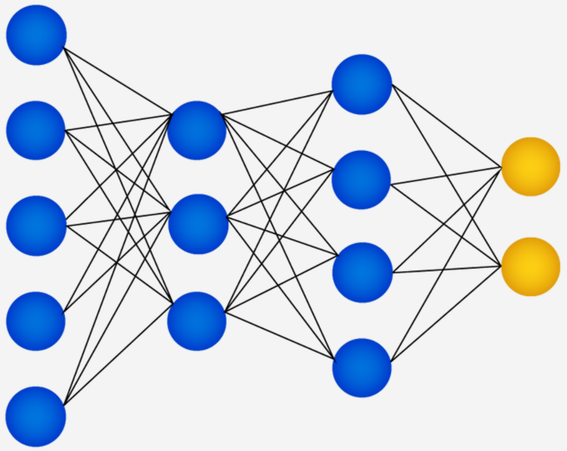
AI : Artificial Intelligence
ML : Machine Learning
DL : Deep Learning



What is Machine Learning?

“Machine learning (ML) is the study of computer algorithms that can improve **automatically** through **experience** and by the **use of data.”**

-Wikipedia

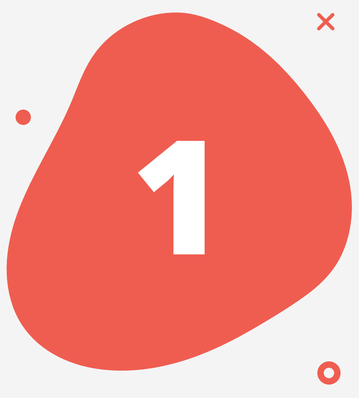


What about Deep Learning?

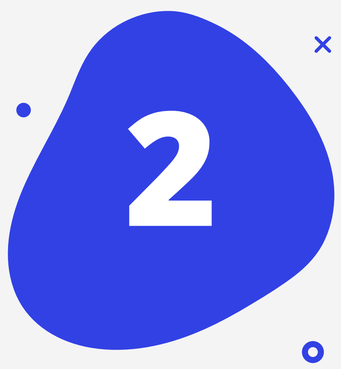
“Deep learning models introduce an extremely **sophisticated** approach to machine learning and are set to tackle these challenges because they've been specifically modeled after the **human brain.”**

-Flatiron School

Types of Machine Learning Algorithms



Supervised Learning



Unsupervised Learning

Reinforcement Learning

Supervised Learning

Labeled data :

The training data contains the "answer"

Apple

Banana

Unsupervised Learning

Unlabeled data :

The training data doesn't contain the "answer"

?

?

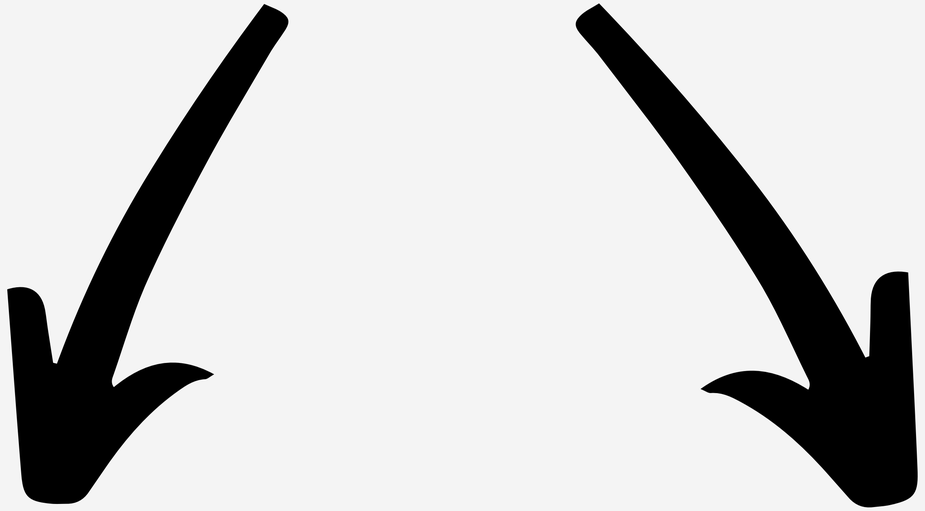
Reinforcement Learning

No data :

action => reward/punishment system

Supervised Learning

Supervised Learning

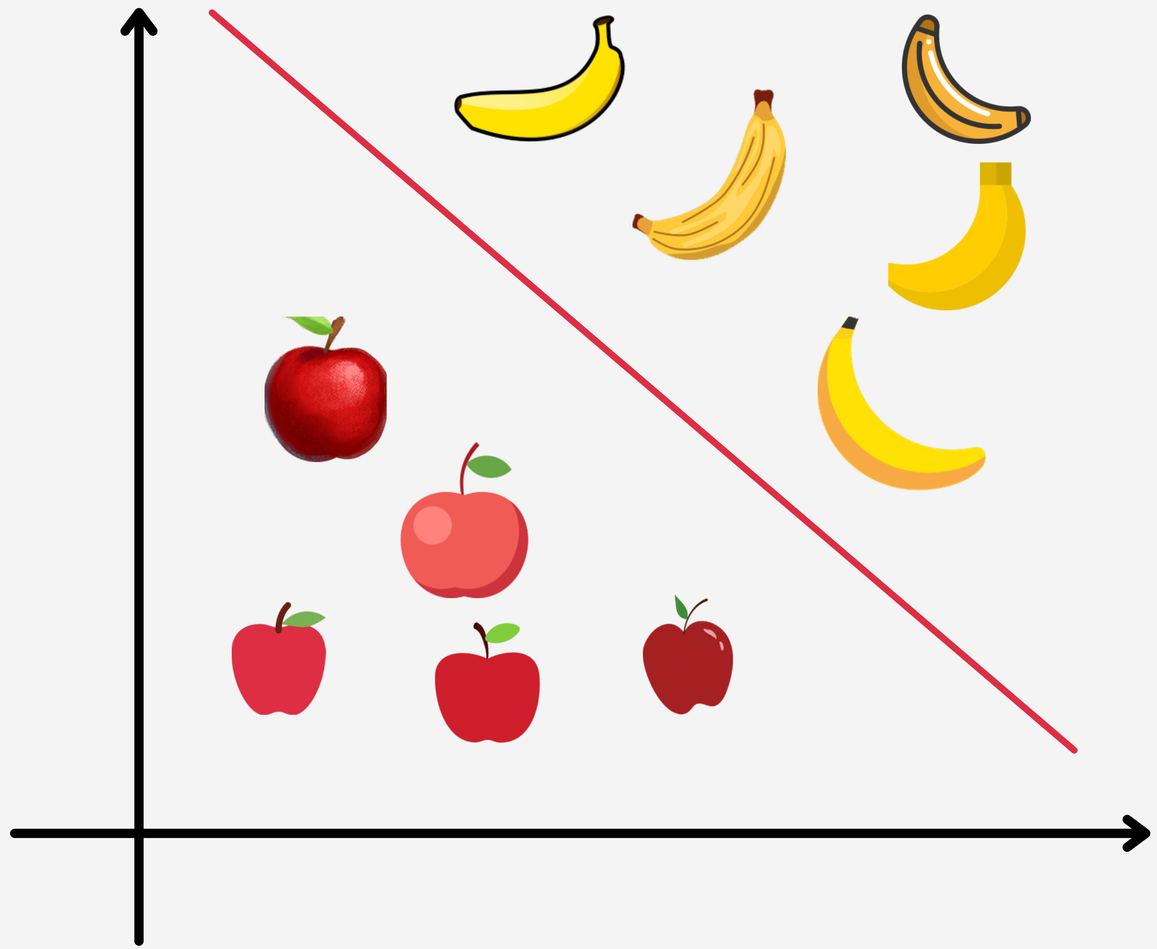


Classification

Regression

Classification

Example 1 :
Banana/Apple
classification



Classification

Example 2 :
Movie Reviews
Classification

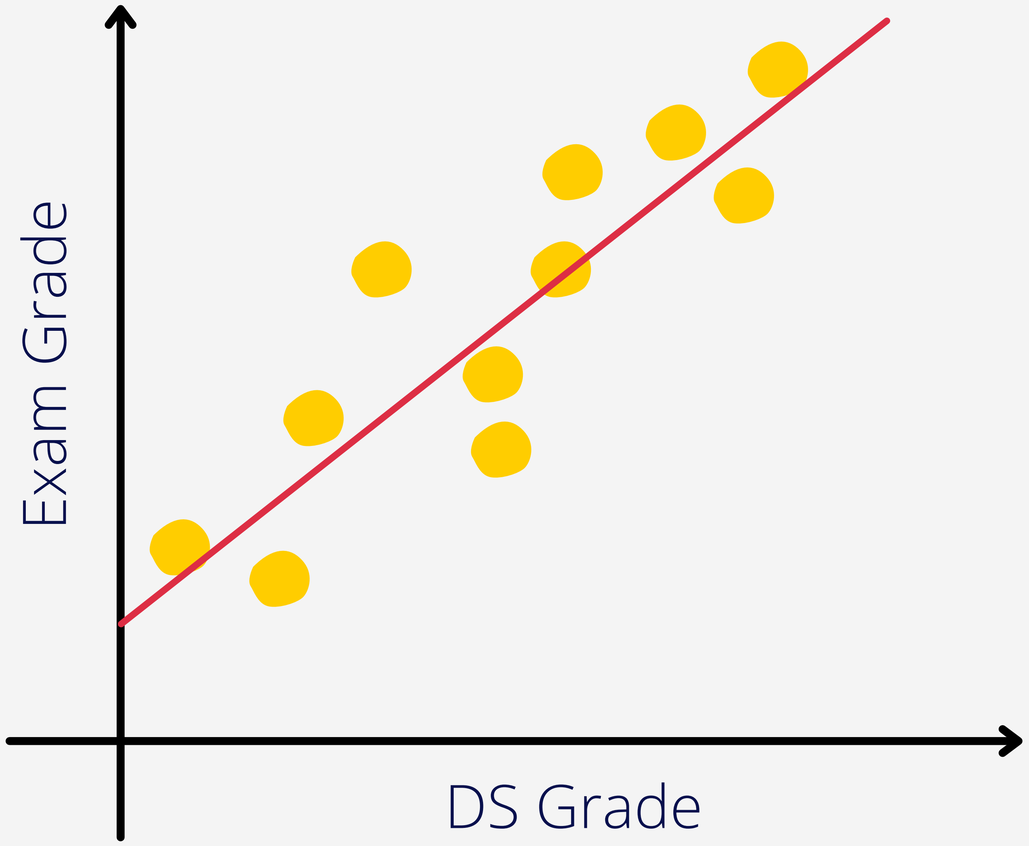
Text	Prediction	Reality	Supervision
•••	Bad	Bad	✓
•••	Good	Good	✓
•••	Neutral	Bad	✗
•••	Neutral	Neutral	✓
•••	Good	Good	✓

Accuracy : 80%

Regression

Example :

Grade prediction



How does the algorithm train?

An example on a regression problem

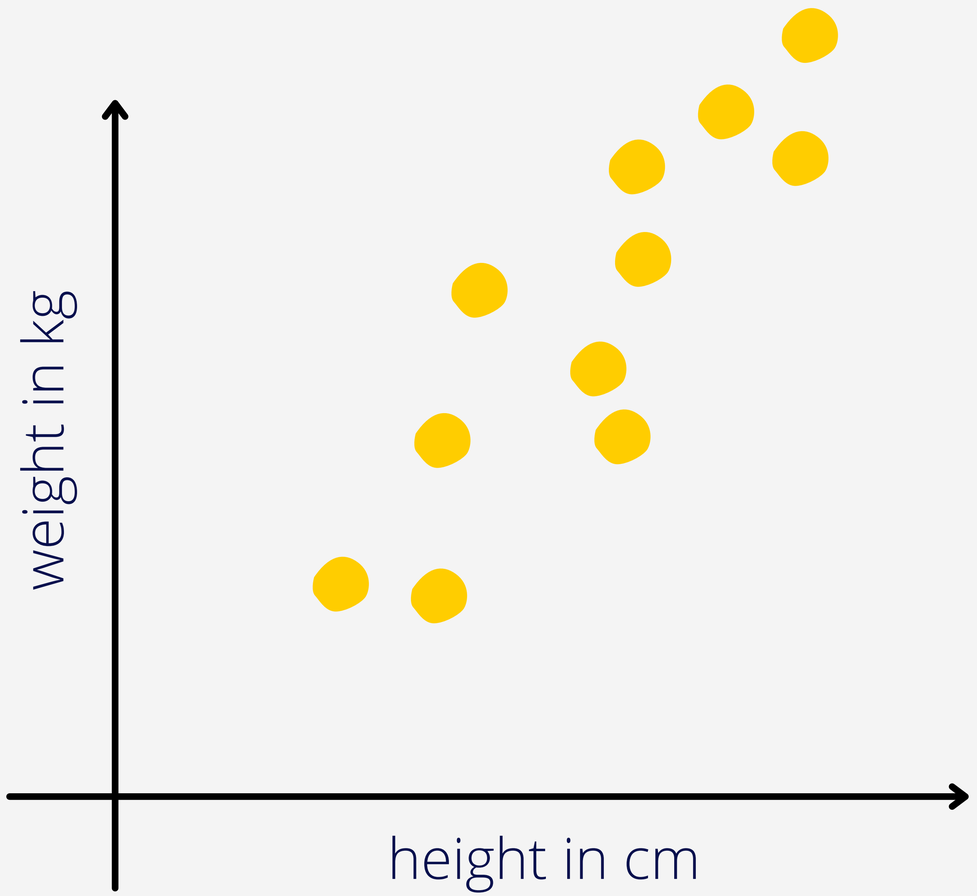
Get the data

1

Feature

Target / Label

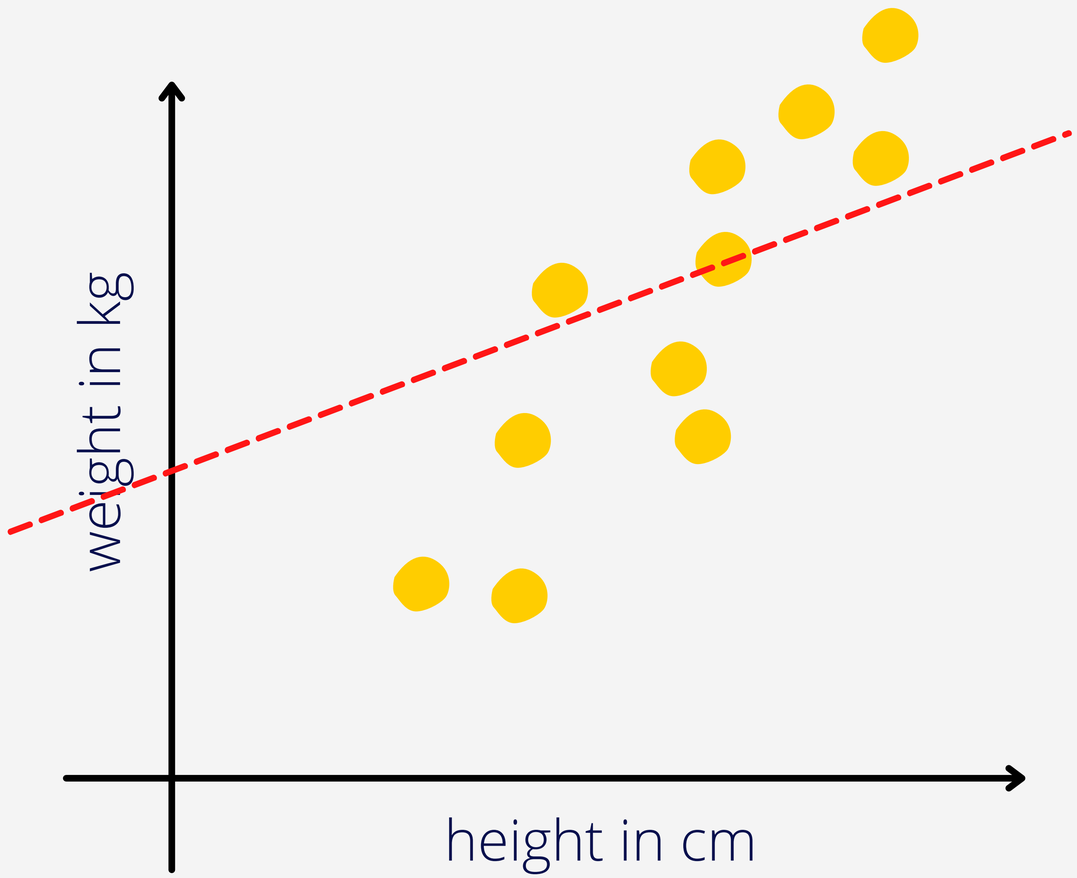
X : Height	Y : Weight
161	50
190	87
173	80



Train the model

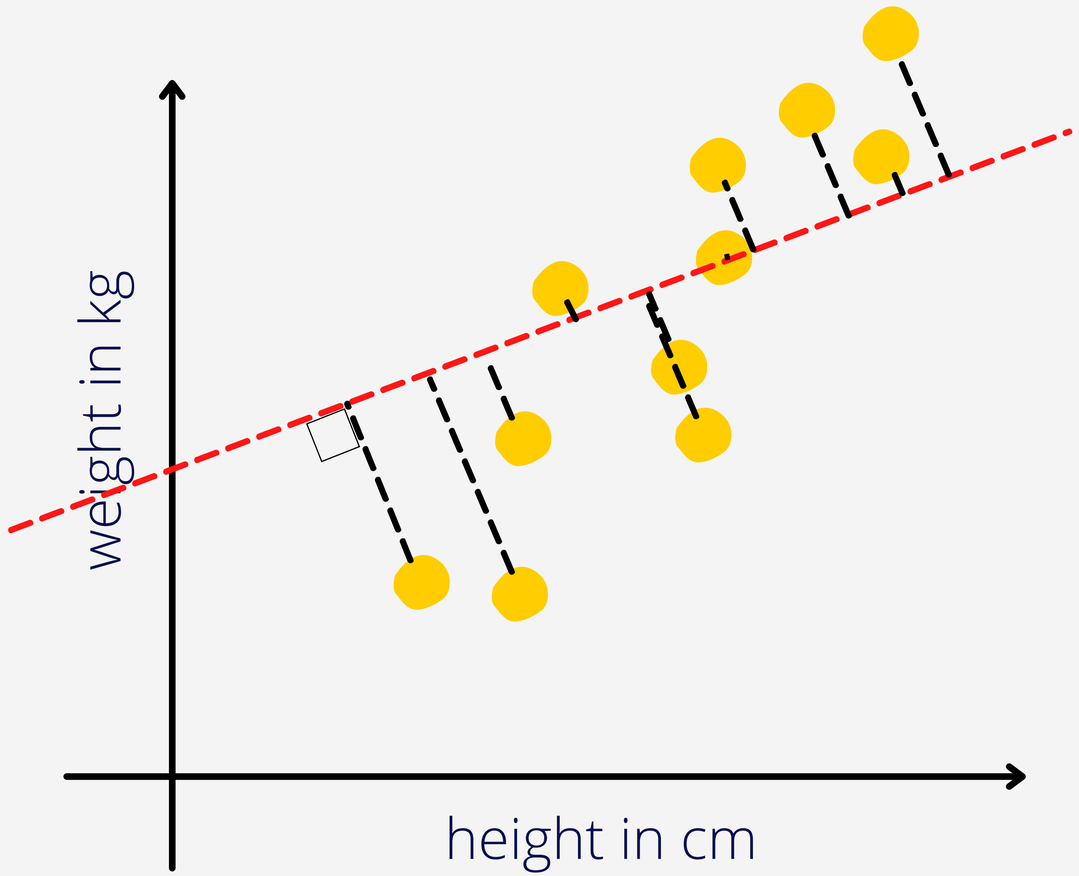
2

W = H +130



1. Make a random prediction

Our goal is to reduce the distance between estimated value and actual value (the error).

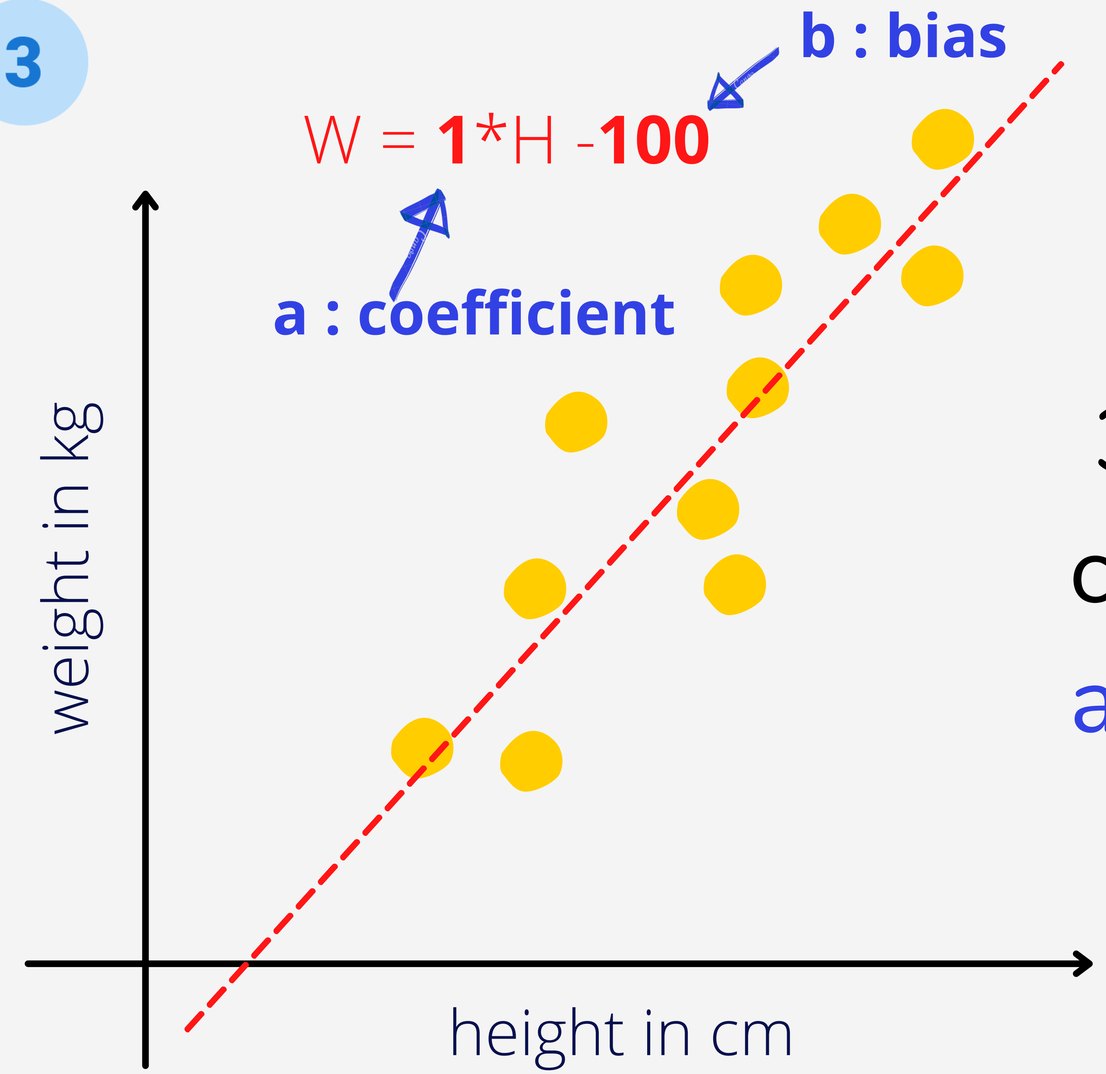


$W = \mathbf{1}*H + \mathbf{130}$

2. How much are we far from the correct answer?

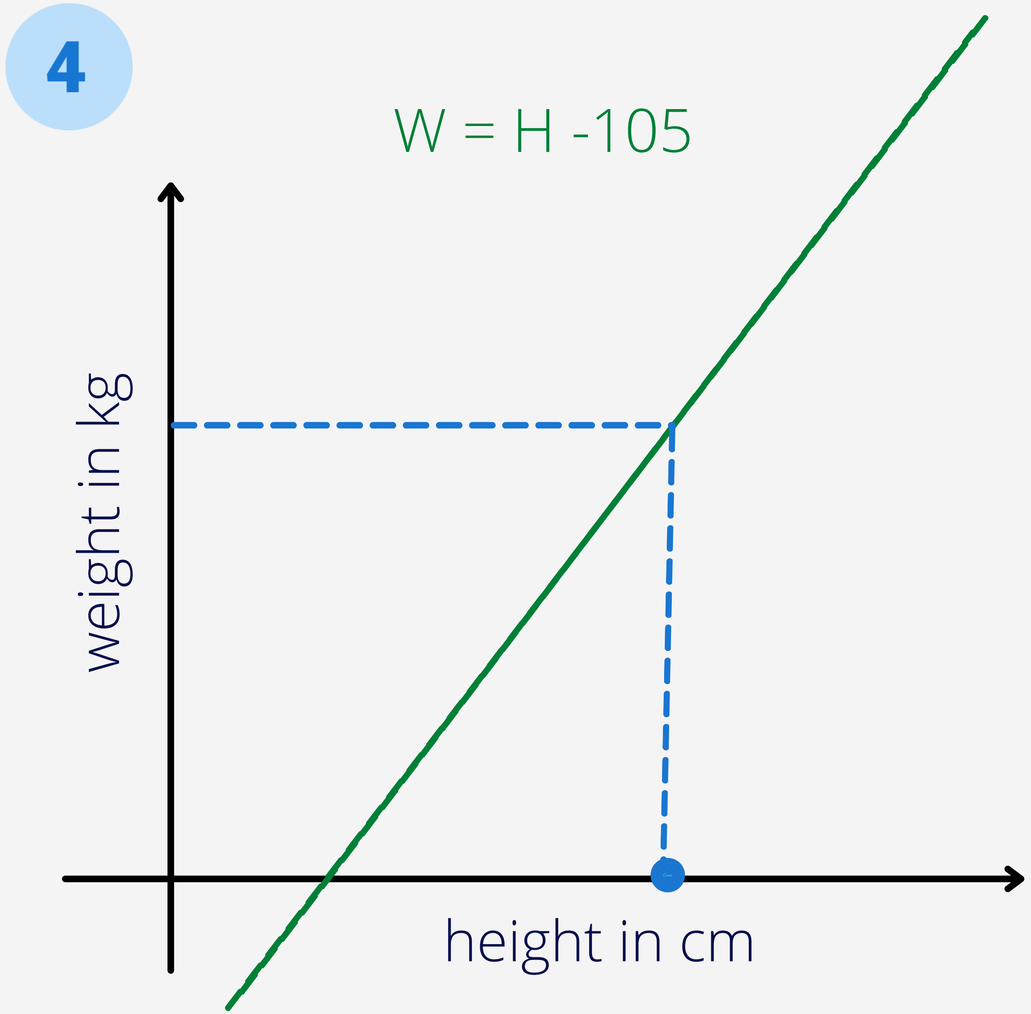
Improve the model

3



3. Find a better line by changing the coefficients a and b.

Iterations / Epochs

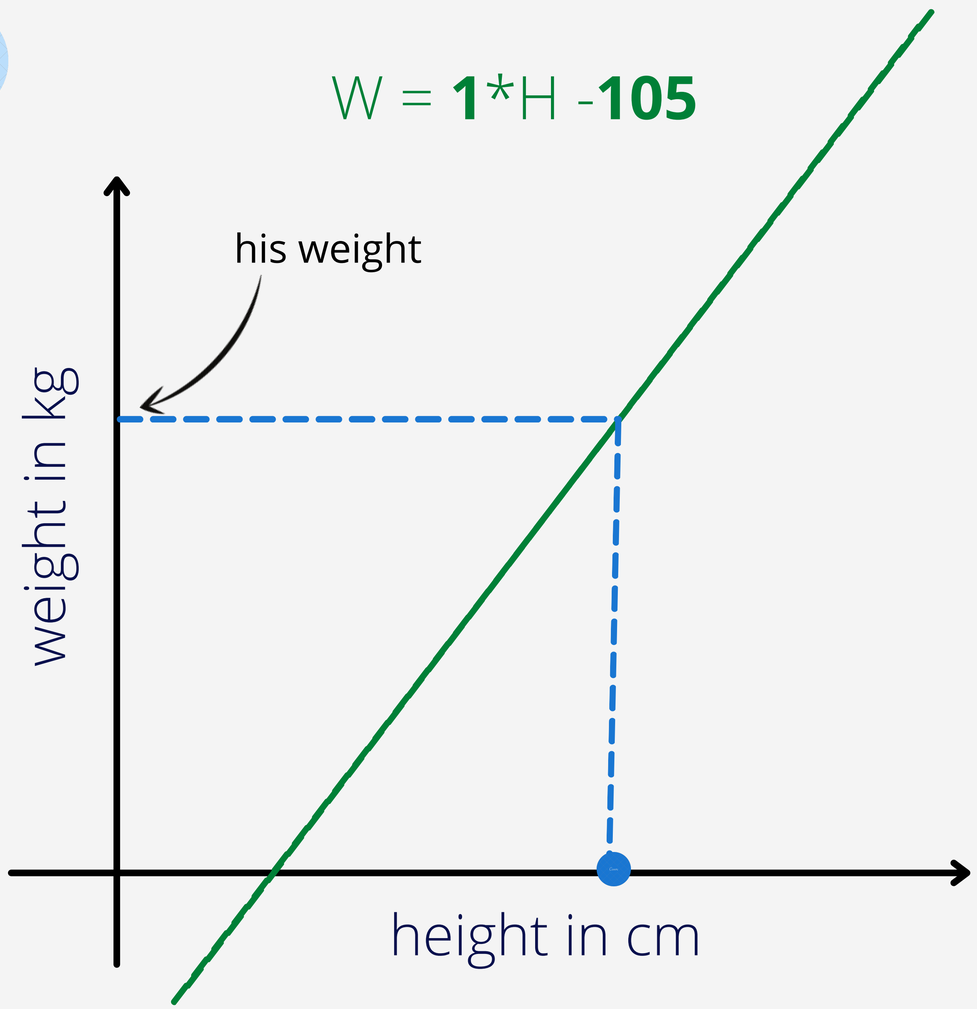


4. Continue the process :
   - Making a prediction (based on your latest a and b values).
   - Seeing how far your prediction is from the correct answer.
   - Improving your coefficients.

Until you find the line of best fit.

Make predictions

5



5. You can now use your model to make a prediction : given a certain height, you can determine the approximative weight of a person.

Regression

Regression Algorithms

- Linear Regression.
- Lasso Regression
- Polynomial Regression
- Ridge Regression

etc ...

Linear Regression

House price prediction problem

Given a house features , we want to train a model that determines its price.
First, we'll suppose we only have one feature, the number of bedrooms.
We collect the data and start the training.
We make our first prediction (randomly) given the input data.

Linear Regression

Questions

- How do we calculate how far our model is from the correct answer in regression problem?
- How do we update our model?

Answer

- We calculate the cost function.
- We use optimization algorithms like Gradient Descent.

Linear Regression

Cost Function J : Mean Squared Error (MSE)

Calculates the difference between the algorithm's output and the right answer.

Let : theta_0=a , theta_1=b

h(xi) = theta_0*xi + theta_1 , xi : the number of bedrooms of a house

yi : the house's price

m : number of training examples

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)^2$$

error

squared

mean

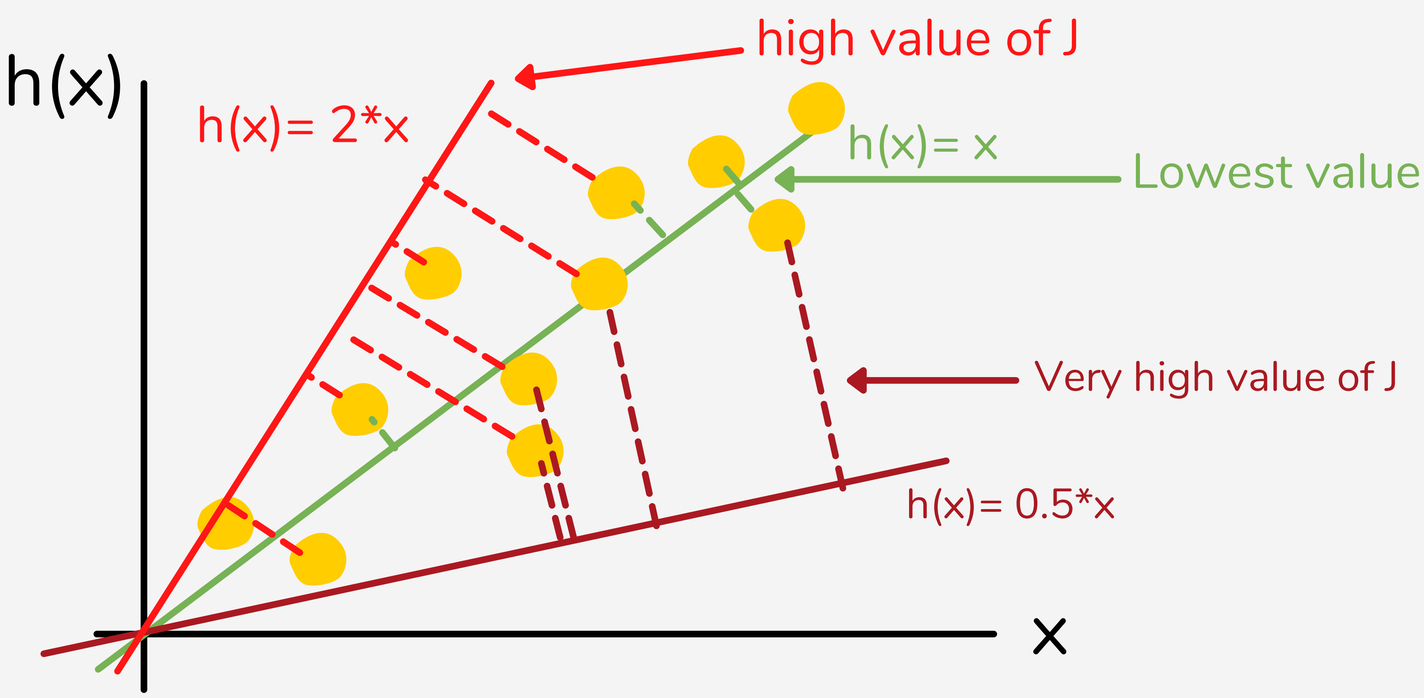
Linear Regression

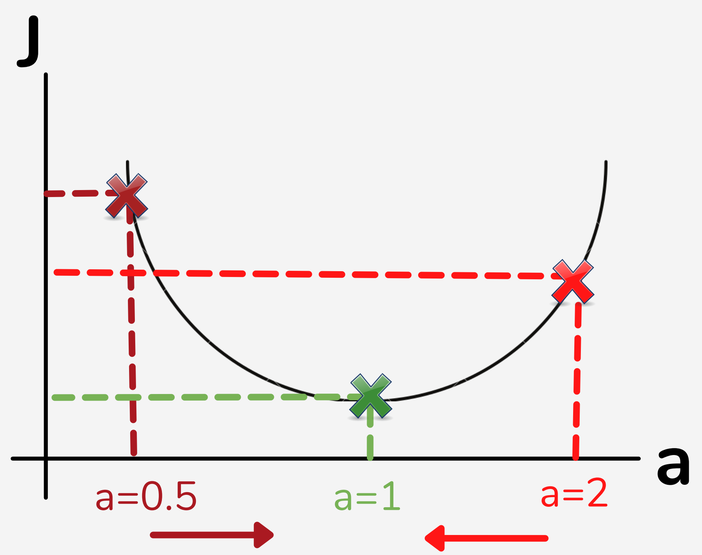
Optimization algorithm : Gradient Descent

$$h(x) = a.x$$

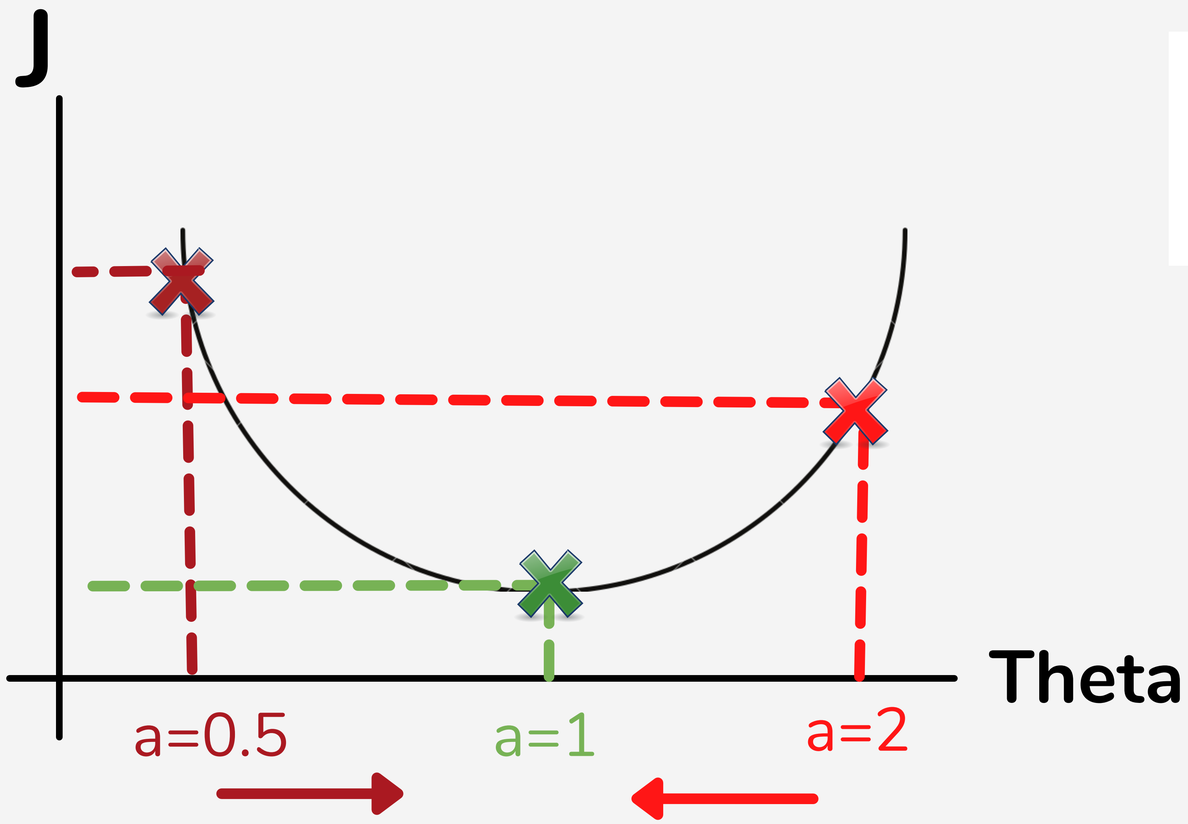
theta_0=a , b=0

x : the number of bedrooms of a house





Linear Regression



$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

alpha : learning rate between 0 and 1

theta_j : coefficient (a=theta_0 , b=theta_1...)

Linear Regression

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

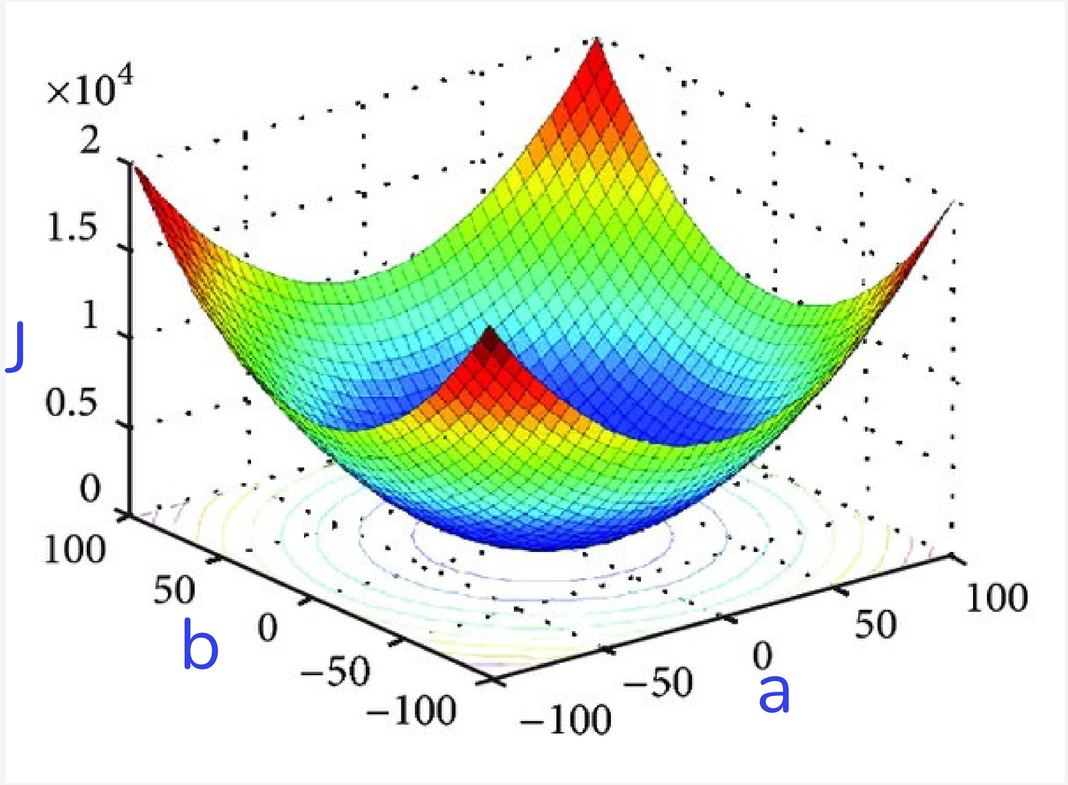
=

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_j^{(i)}$$

$$\begin{aligned}
\frac{\partial}{\partial \theta_j} J(\theta) &= \frac{\partial}{\partial \theta_j} \frac{1}{2} (h_\theta(x) - y)^2 \\
&= 2 \cdot \frac{1}{2} (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} (h_\theta(x) - y) \\
&= (h_\theta(x) - y) \cdot \frac{\partial}{\partial \theta_j} \left(\sum_{i=0}^{n} \theta_i x_i - y \right) \\
&= (h_\theta(x) - y) \, x_j
\end{aligned}$$

Linear Regression

If we don't assume that b=0, we'll have a cost function J depending on 2 variables : a (theta_0) and b (theta_1). Thus, the 2D graph becomes a 3D graph.



Linear Regression

NB : We first assumed we had one feature (number of bedrooms). In reality, we're going to have a lot of features x1,x2...xn. The Cost Function J will depend on all of them + the b (bias). Thus, our function will look like :

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \cdots + \theta_n x_n$$

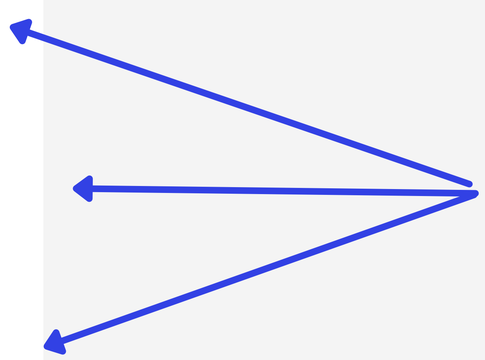
The coefficient's update using gradient descent will be :

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_0^{(i)}$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_1^{(i)}$$

$$\theta_2 := \theta_2 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)}) \cdot x_2^{(i)}$$

$$\dots$$



Update every coefficient using gradient descent

Linear Regression

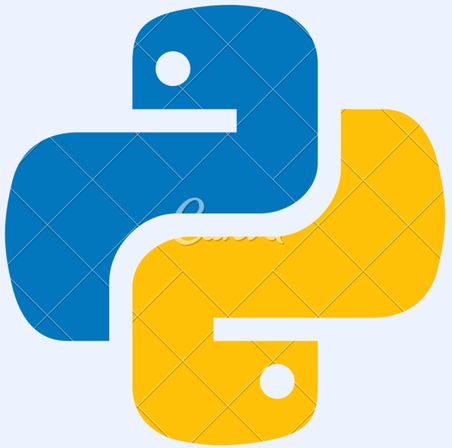
CODE

```
]  from sklearn.linear_model import LinearRegression
   from sklearn.metrics import mean_squared_error

   lr = LinearRegression()
   lr.fit(X_train,y_train)
   predict = lr.predict(X_test)
   test_score = mean_squared_error(y_test,predict_test)
   print(train_score)
```

Train the model (finds the function that suits the data the most)

The MSE that we talked about : using it to evaluate our prediction



LET'S PRACTICE!