CADT

Cambodia Academy of Digital Technology

Linear Regression

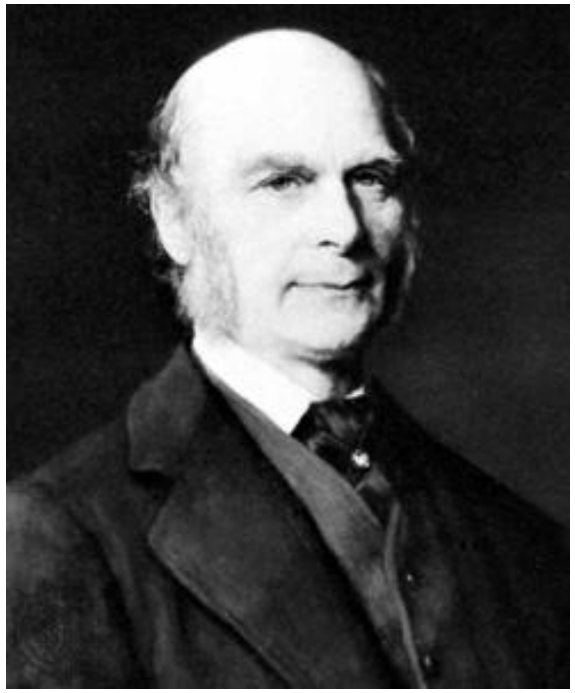
Lecturer: Soklay HENG

Academic Year: 2026-2027

Introduction

- **Regression (Analysis)** is a statistical method used in finance, investment, and other disciplines that attempts to determine the strength of relationship between one continuous dependent variable and a series of other changing variables known as independent variables.
- Regression helps investment and financial institution to value assets and understand the relationships between variables, such as commodity prices and the stocks of businesses dealing in those commodities.

Introduction

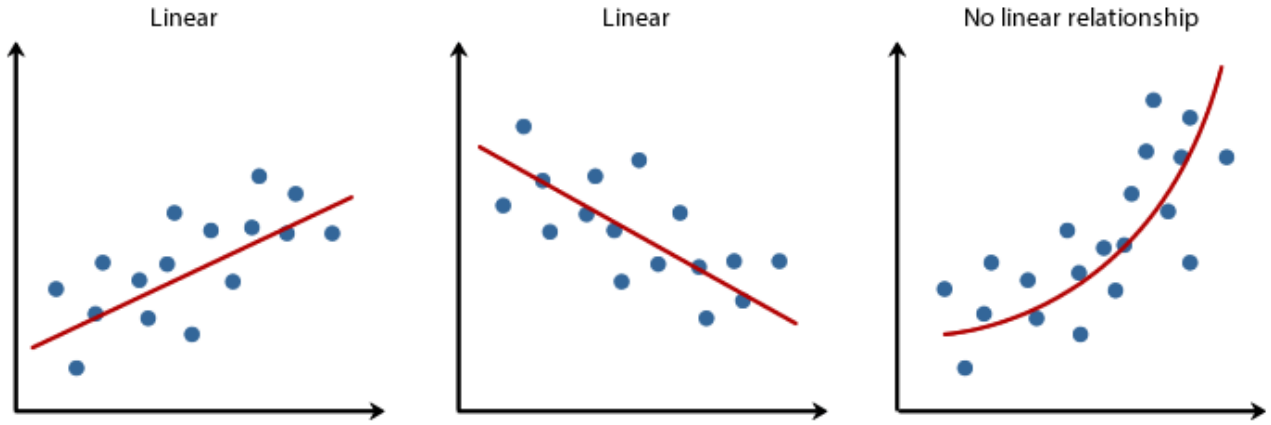


Francis Galton

- Regression was invented by the cousin of Charles Darwin, **Francis Galton**.
- Galton did his first regression in 1877 to estimate the size of pea seeds based on the size of their parents' seeds.
- Galton performed regression on a number of things, including the heights of humans.
- He noticed that if parents were above average in height, their children also tended to be above average but not as much as their parents.
- The heights of children were regressing toward a mean value.

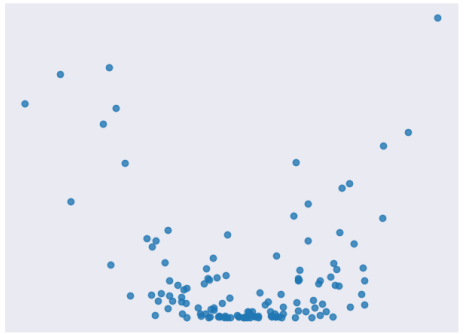
Introduction

- In regression, there are two types of data to be taken into account:
  - Linear Data
  - Non-Linear Data
- Choosing which learning algorithm to be applied, does depends on the distribution and the characteristic of the data.

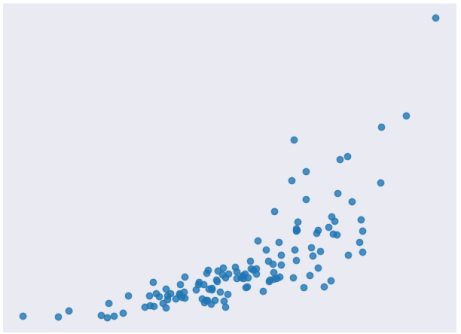


When to use linear regression

Linear regression is a useful technique but isn't always the right choice. Linear regression is a good choice when there is a linear relationship between your independent and dependent variables and you are trying to predict continuous values.



(1) Not a linear relationship at all.



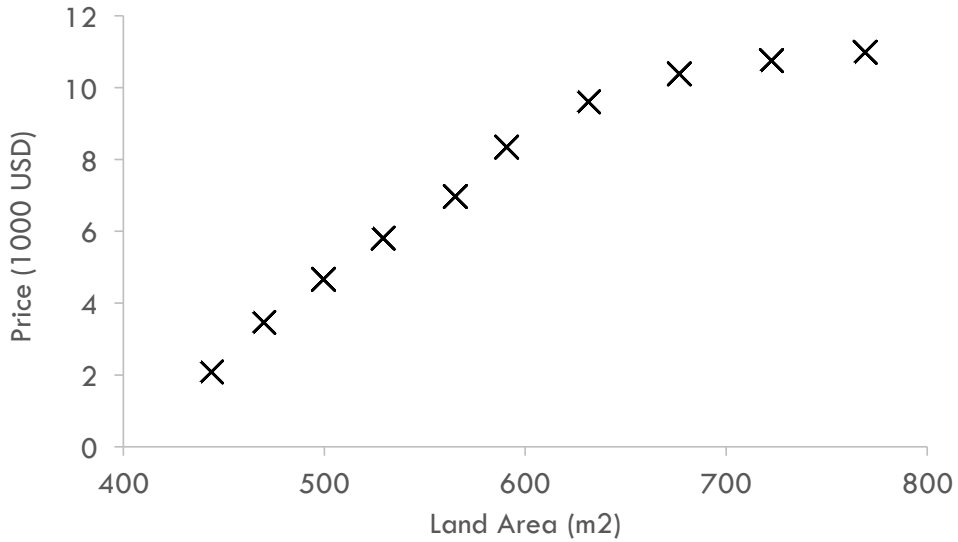
(2) Not linear, but can be transformed into a linear relationship.



(3) transformed data points of (2)

EXAMPLE

Land price prediction



Supervised Learning

Regression Problem:
predict continuous valued output

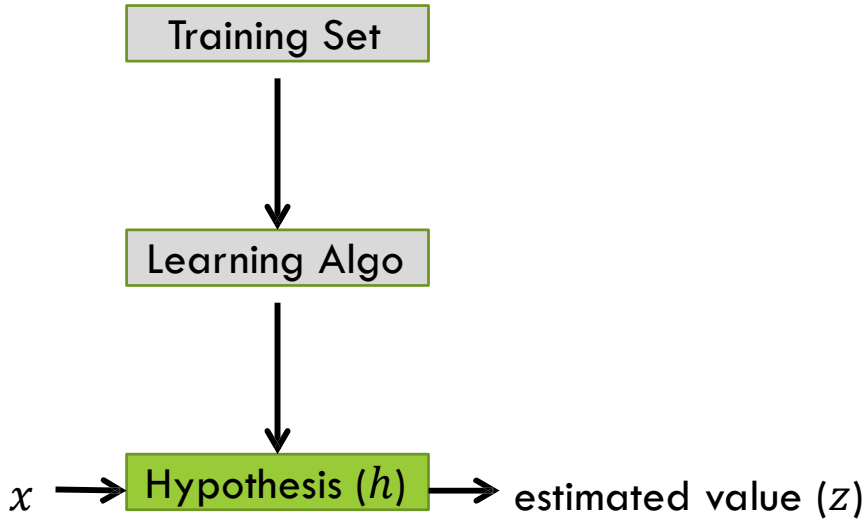
TRAINING SET

- Notations:
  - m = number of training examples
  - x = input variable / feature
  - y = output/target variable
  - (x, y) = one training example
  - $(x^{(i)}, y^{(i)})$ = the i^{th} training example

Example: training set of land price

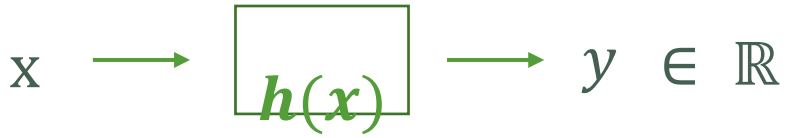
Size in m^2 (x)	Price in 1000 USD (y)
90	2.1
130	3.9
210	4.5
…	…

MODEL REPRESENTATION



Linear Regression

- **Linear Regression** is a simple regression algorithm which uses **linear model** to study and analyze the relationship between a scalar/continuous response and one or more explanatory variables. It performs well when the input and output variables have a linear relationship.
- The learning is done by fitting a **linear equation** to the observed data, which in turn will be then used to predict values for the new data/observation.
- The case of one independent variable is called **simple/univariate linear regression**. For more than one independent variable, the process is called **multi/multivariate linear regression**.

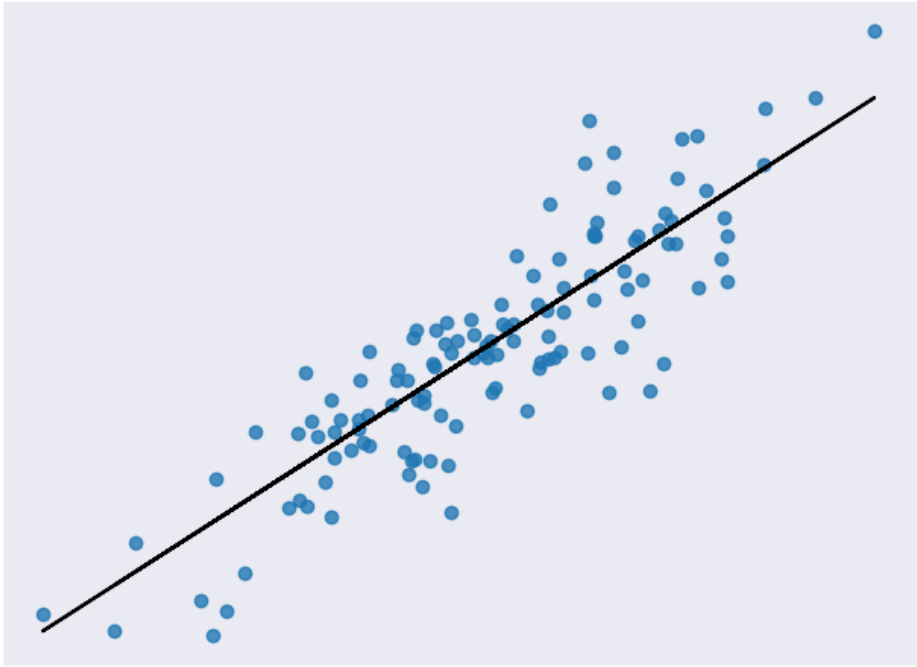


Example of a linear relationship

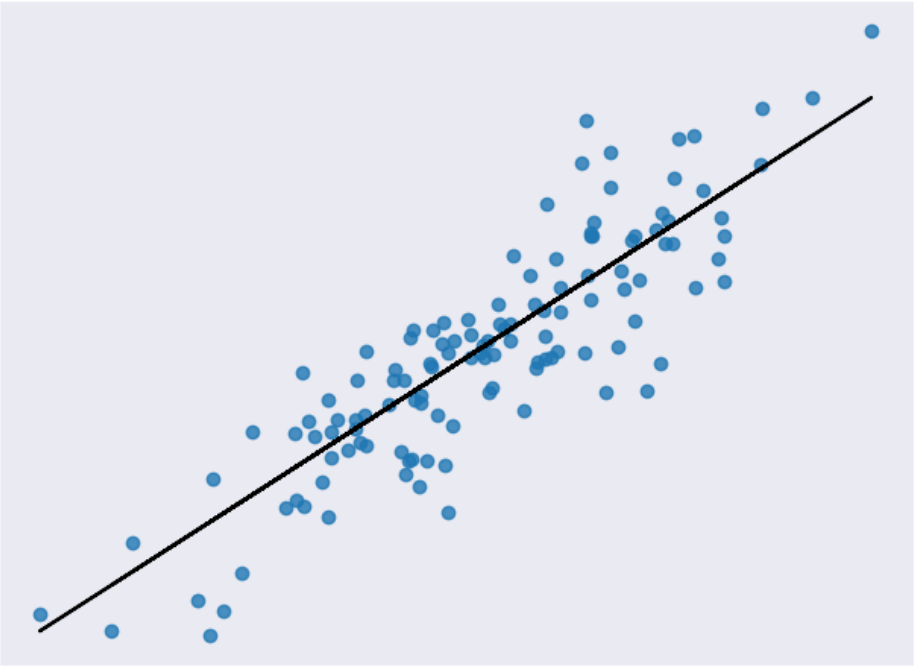


Problem statement

- Let's work on a simple linear regression that involves only a single input variable and a single output variable.



Problem statement



- Let's work on a simple linear regression that involves only a single input variable and a single output variable.
- Our goal is to find the line that best models the path of the data points called a line of best fit.

Problem statement

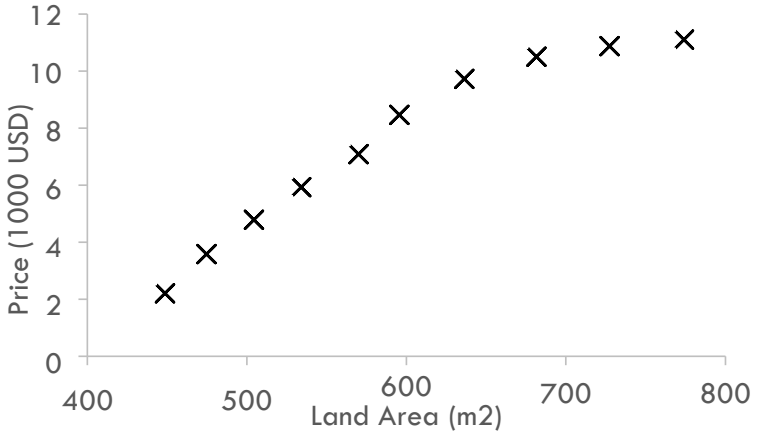
Mathematically, we are looking to construct:

$$y = a_1 * x + b \qquad (1)$$

- During inference, we compute y by using a_1, b and the input x.
- b, a_1 are called coefficients, or model parameters.
- b is called bias
- a_1 is called weight
- x is called input and y is called output. Sometime we write $\hat{y}$ to indicate "esimated value of y".

Changing b will move the line up or down on the plot and changing a_1 changes the slope of the line. Linear regression helps us pick appropriate values for b and a_1.

COST FUNCTION



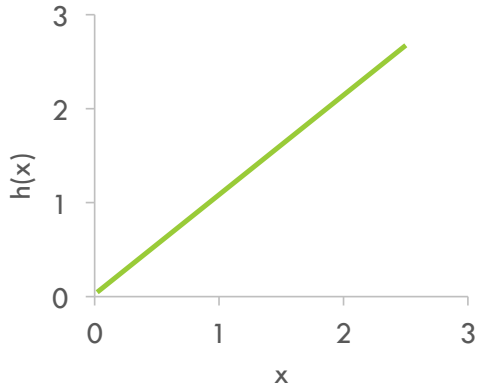
Size in m^2 (x)	Price in 1000 USD (y)
90	2.1
130	3.9
210	4.5
…	…

- Hypothesis: $h(x) = ax + b$ where a and b are called **parameters**

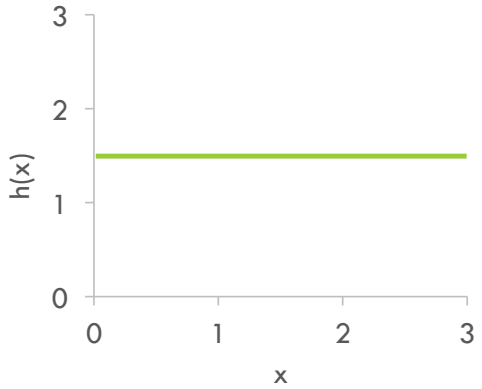
How to choose a and b?

COST FUNCTION (CONT.)

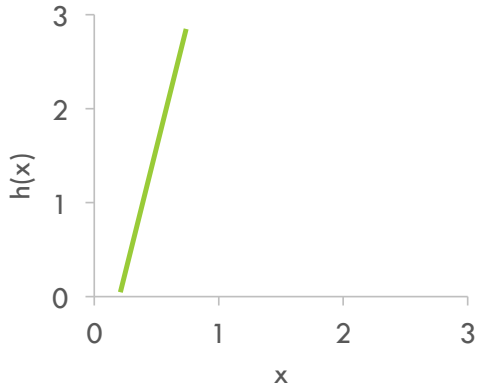
- Hypothesis: $h(x) = ax + b$



$a = 1$
$b = 0$

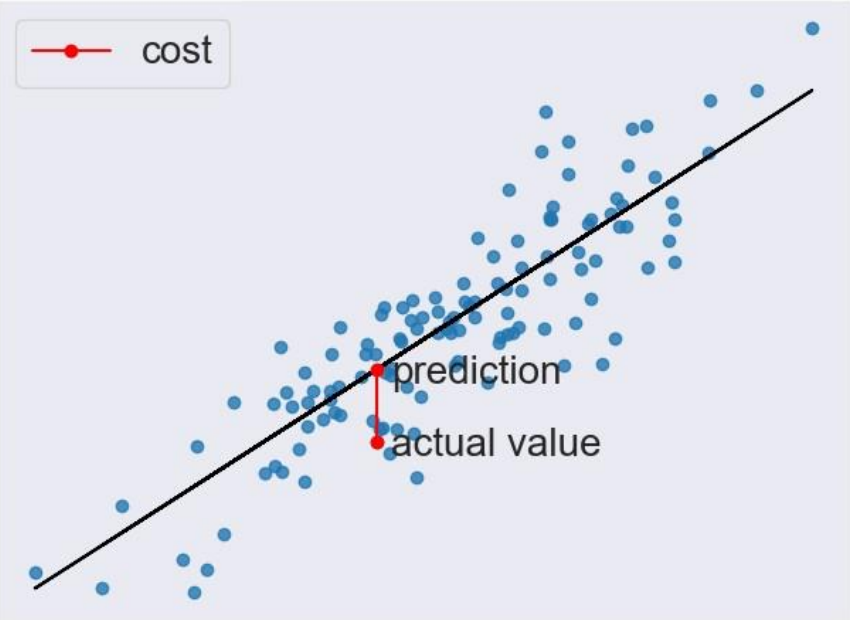


$a = 0$
$b = 1.5$



$a = 2.5$
$b = -0.5$

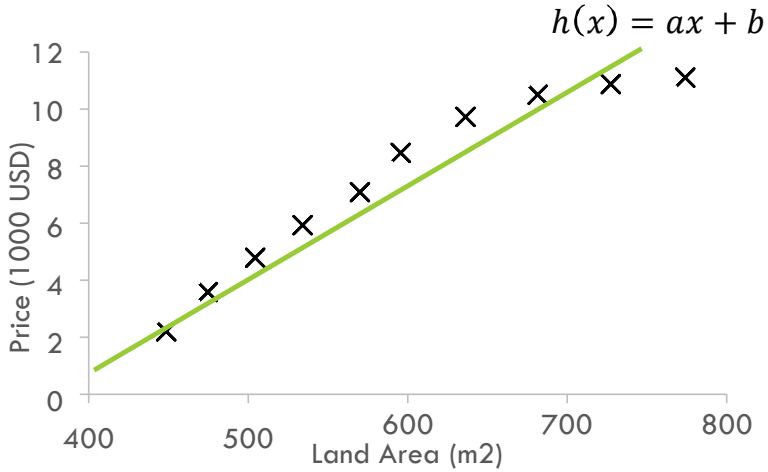
How to find those parameters?



Once we have a prediction, we need some way to tell if it's reasonable. A cost function helps us do this.

The cost function compares all the predictions against their actual values and provides us with a single number that allow us to score the prediction function.

COST FUNCTION (CONT.)



- Choose a and b so that $h(x)$ is close to y for the training examples (x, y)

For each $(x^{(i)}, y^{(i)})$:
Minimize $|h(x^{(i)}) - y^{(i)}|$

$\Rightarrow$ Minimize $\frac{1}{m} \sum_{i=1}^{m} |h(x^{(i)}) - y^{(i)}|$

Squared Error Function:

$$J(a, b) = \frac{1}{m} \sum_{1=1}^{m} (h(x^{(i)}) - y^{(i)})^2$$

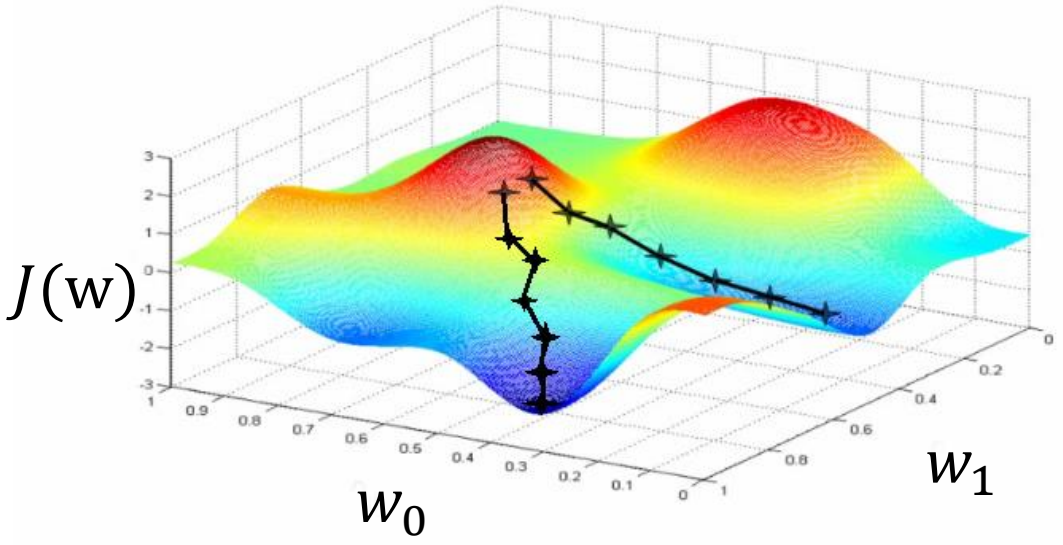
$\Rightarrow \min_{a,b} J(a, b)$

Gradient Descent

Gradient Descent is an optimization algorithm used to **minimize** the **cost function**. We first initialize the model parameters by random values and iteratively update their values till we reach the global minimum by computing derivative of the loss with respect to each parameter in every iteration and updating parameters as follows:

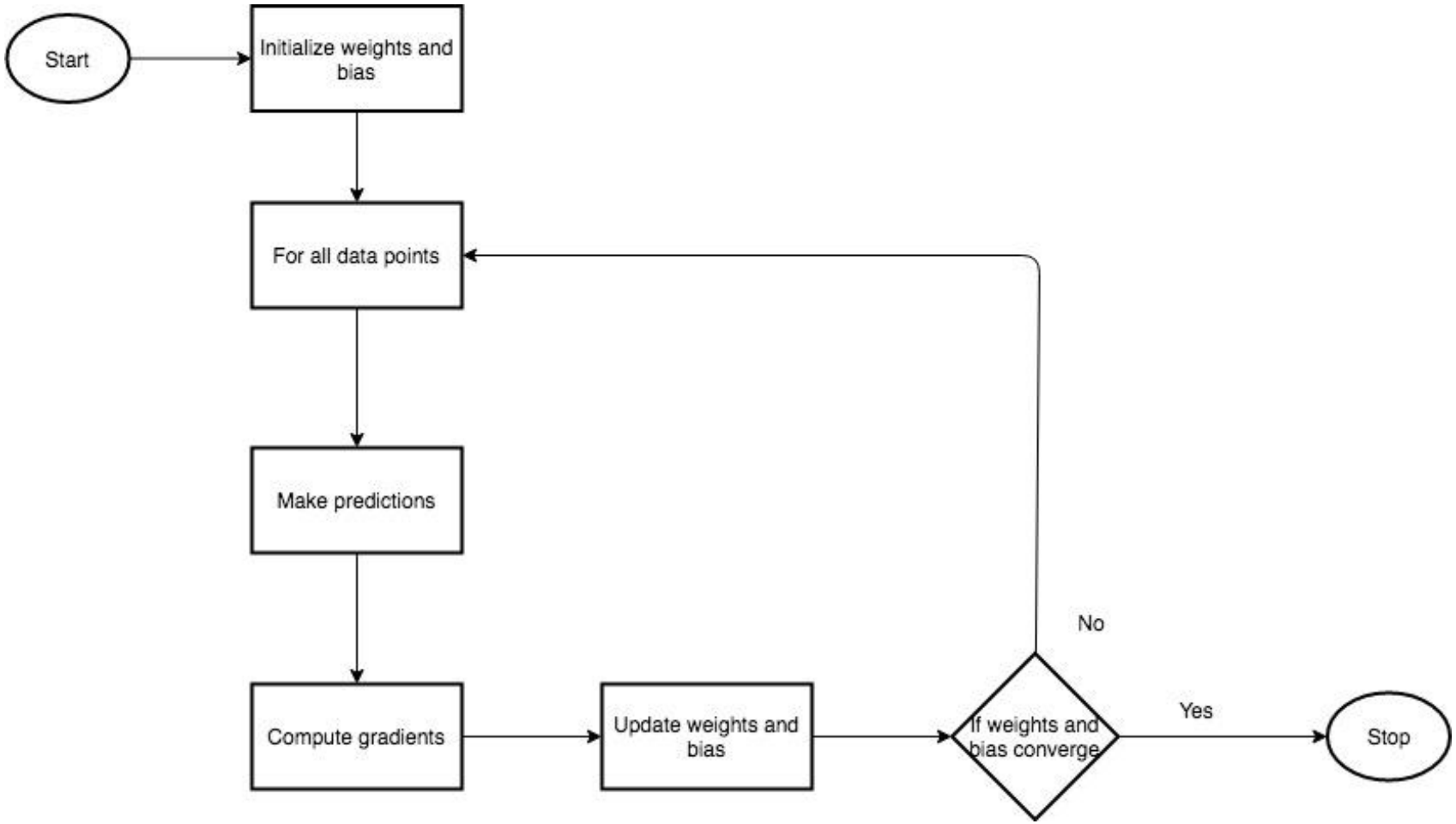
$$w_i = w_i - \eta \frac{\partial}{\partial w_i} J(\mathrm{w})$$

Where η is the learning rate/step



Gradient Descent Learning Process

Gradient Descent - Algorithms



Batch Gradient Descent Flowchart

Gradient Descent

- Given n examples $\{(\mathrm{x}_1, y_1), (\mathrm{x}_2, y_2), \ldots, (\mathrm{x}_n, y_n)\}$ such that $\mathrm{x} \in \mathbb{R}$, the gradients with respect to w_0 and w_1 are calculated as follows:

$$\frac{\partial}{\partial w_0} J(\mathrm{w}) = \frac{1}{n} \sum_{i=1}^{n} (h_{\mathrm{w}}(\mathrm{x}_i) - y_i)$$

$$\frac{\partial}{\partial w_1} J(\mathrm{w}) = \frac{1}{n} \sum_{i=1}^{n} (h_{\mathrm{w}}(\mathrm{x}_i) - y_i)\, x_i$$

$$\Longrightarrow \frac{\partial}{\partial w_i} J(\mathrm{w}) = \frac{1}{n} \sum_{i=1}^{n} (h_{\mathrm{w}}(\mathrm{x}_i) - y_i)\, x_i, \quad x_0 = 1$$

$$J(\mathrm{w}) = \frac{1}{n} \sum_{i=1}^{n} (h_{\mathrm{w}}(\mathrm{x}_i) - y_i)^2$$

$$h_{\mathrm{w}}(\mathrm{x}) = w_0 + w_1 x_1$$

Gradient Descent

Initialize w_0 and w_1 with random values

For all data points:

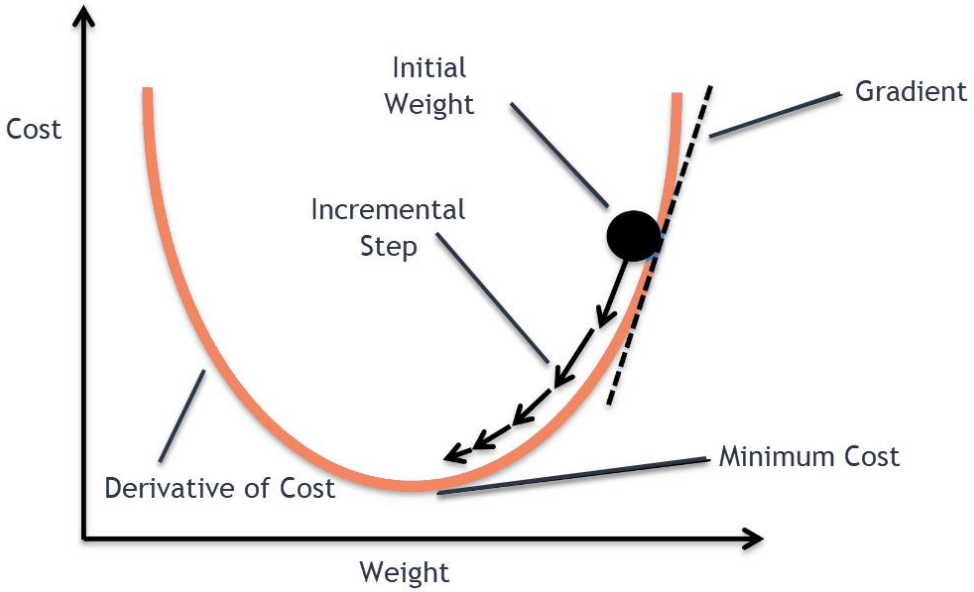
make predictions – compute $h_{\mathrm{w}}(\mathrm{x}_i)$

For each weight w_i:

compute the gradient and update new weight

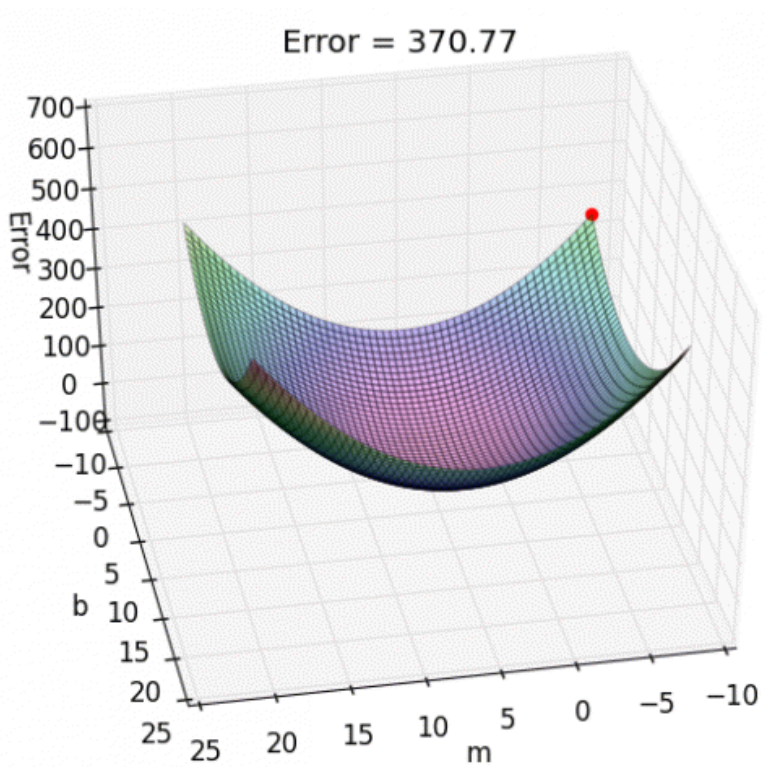
$$w_{\mathrm{i}} = w_{\mathrm{i}} - \frac{\eta}{n}\sum_{i=1}^{n}(h_{\mathrm{w}}(\mathrm{x}_i) - y_i)x_i$$

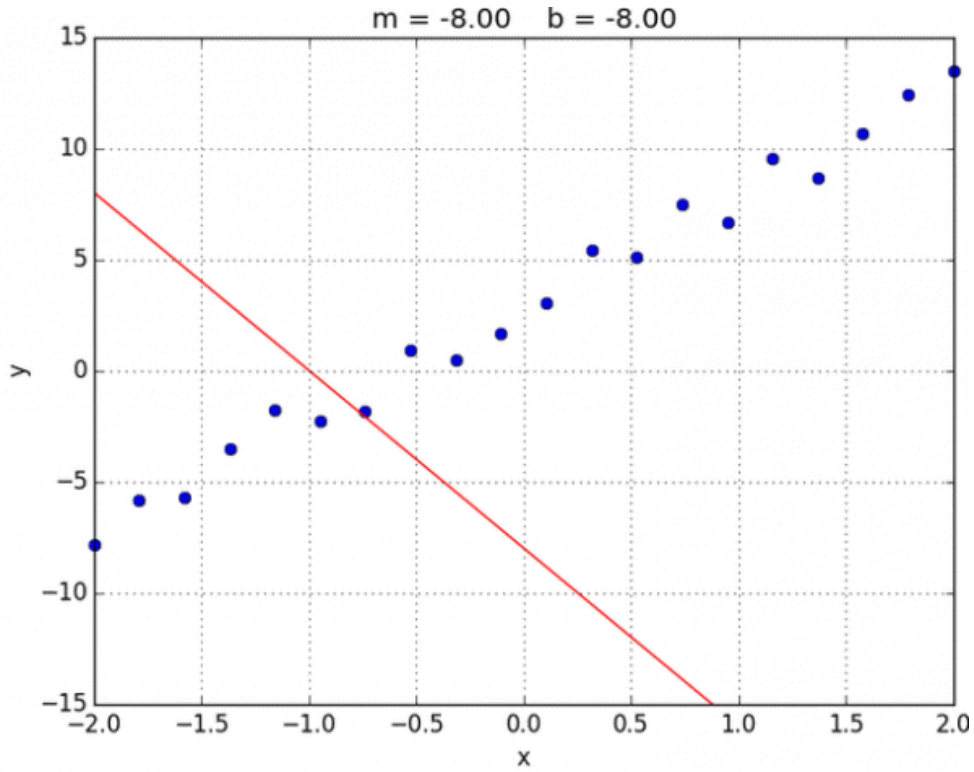
Until weights converges



Learning Process

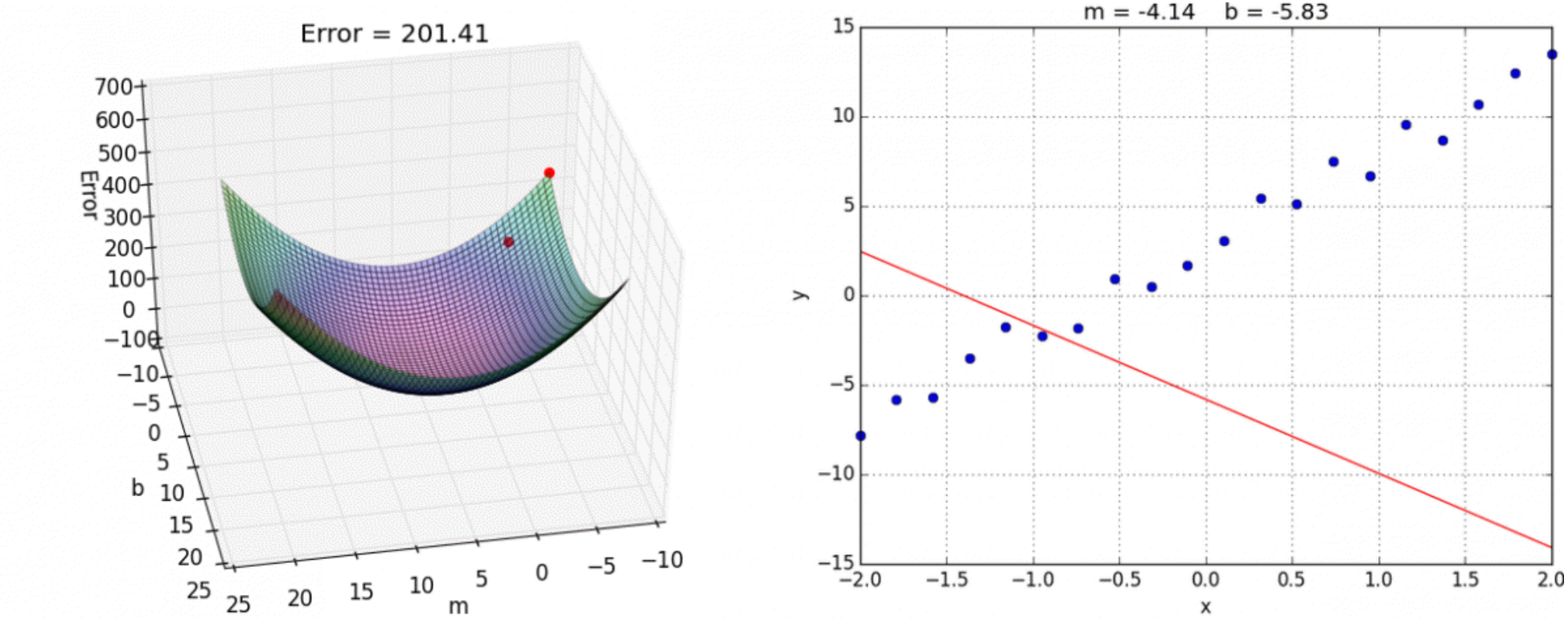
Gradient Descent





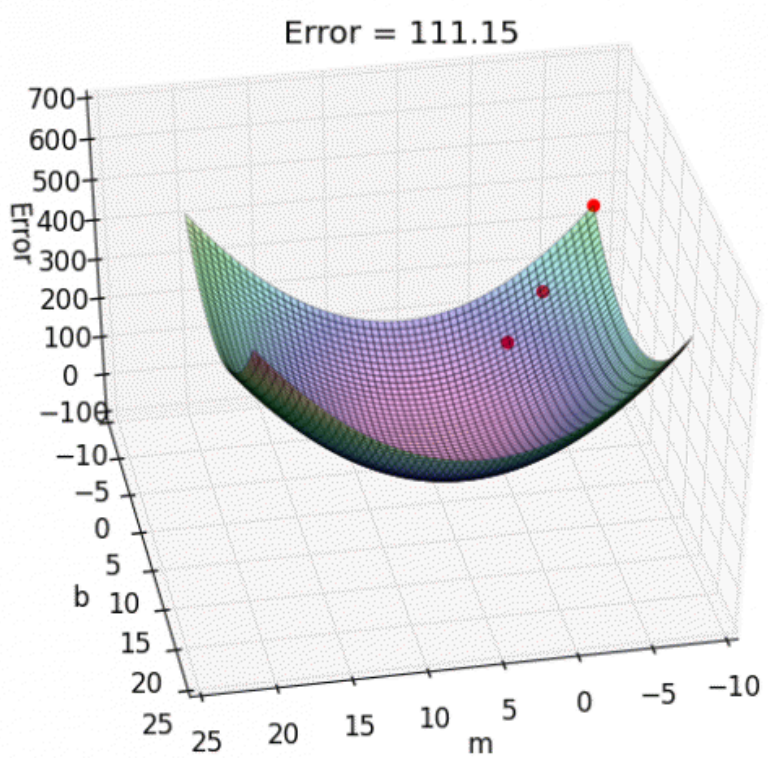
$$m = w_1,\ b\ = w_0$$

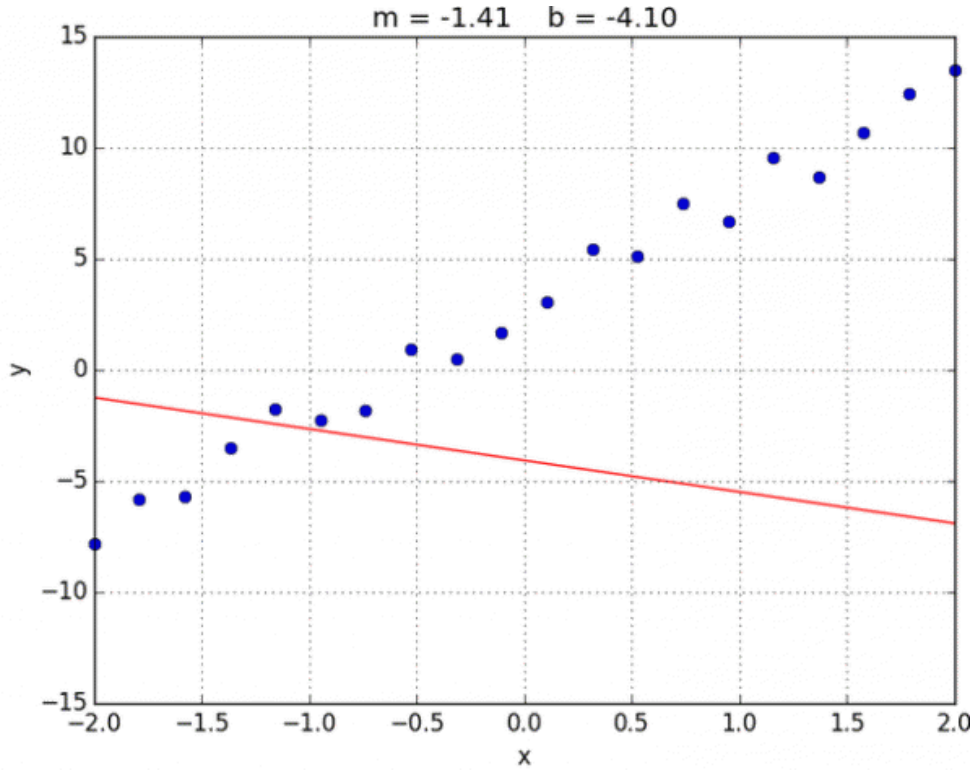
Gradient Descent



$$m = w_1, b = w_0$$

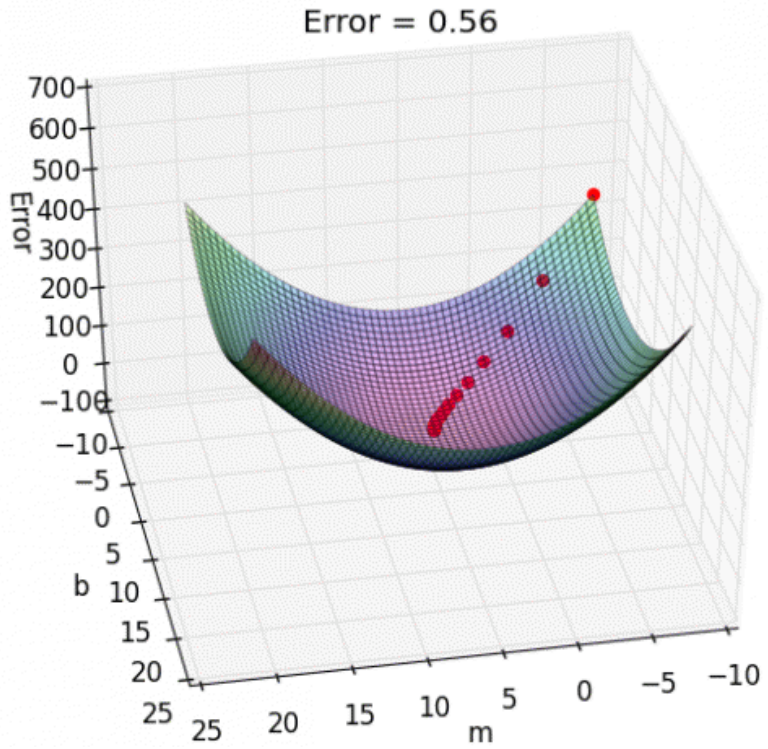
Gradient Descent

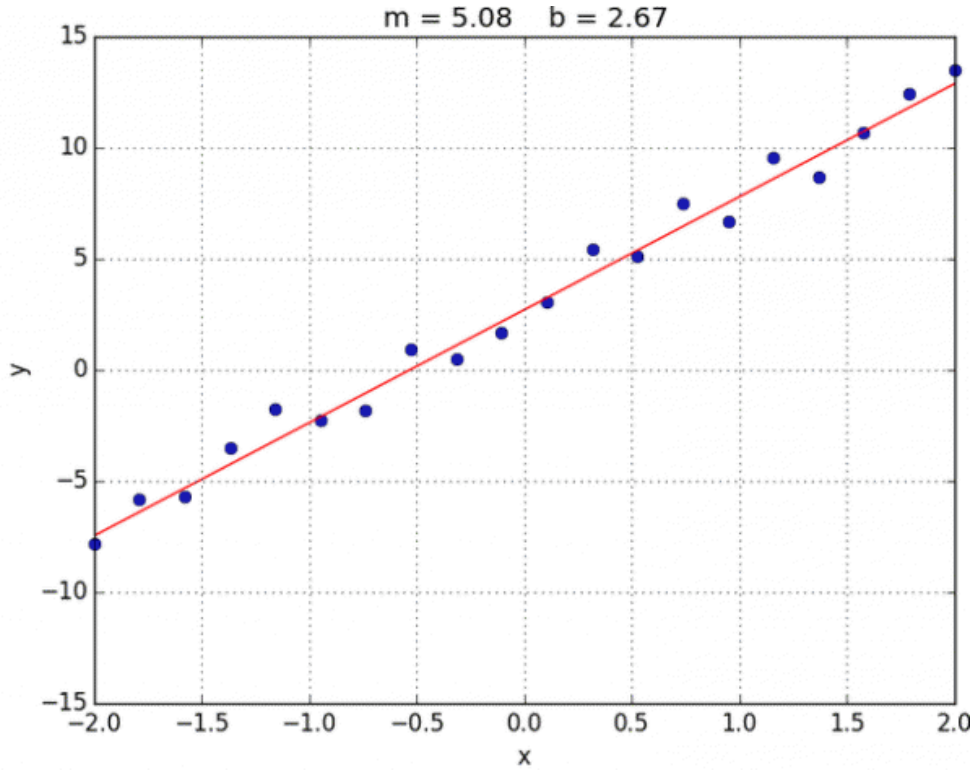




$m = w_1, b = w_0$

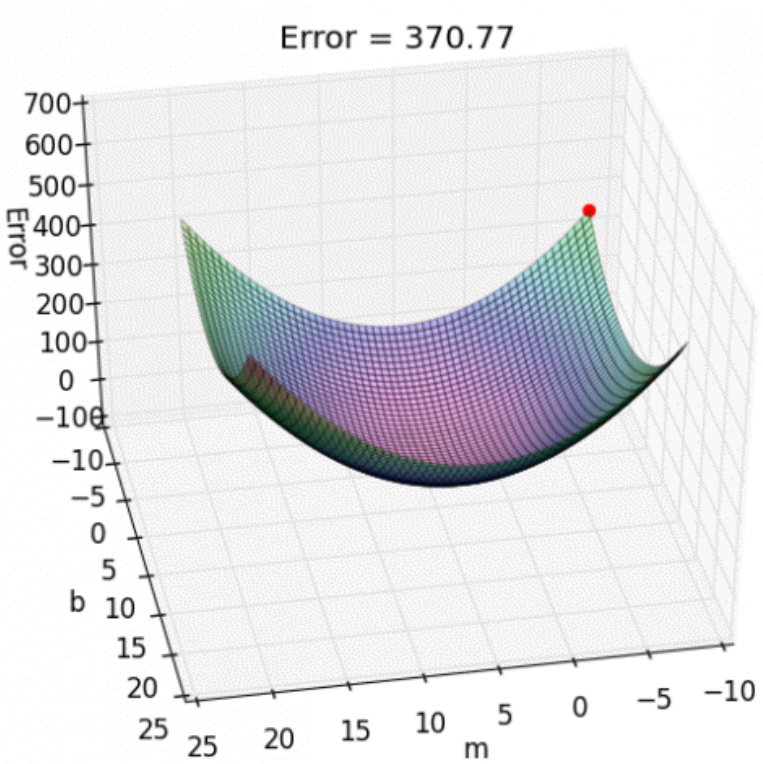
Gradient Descent

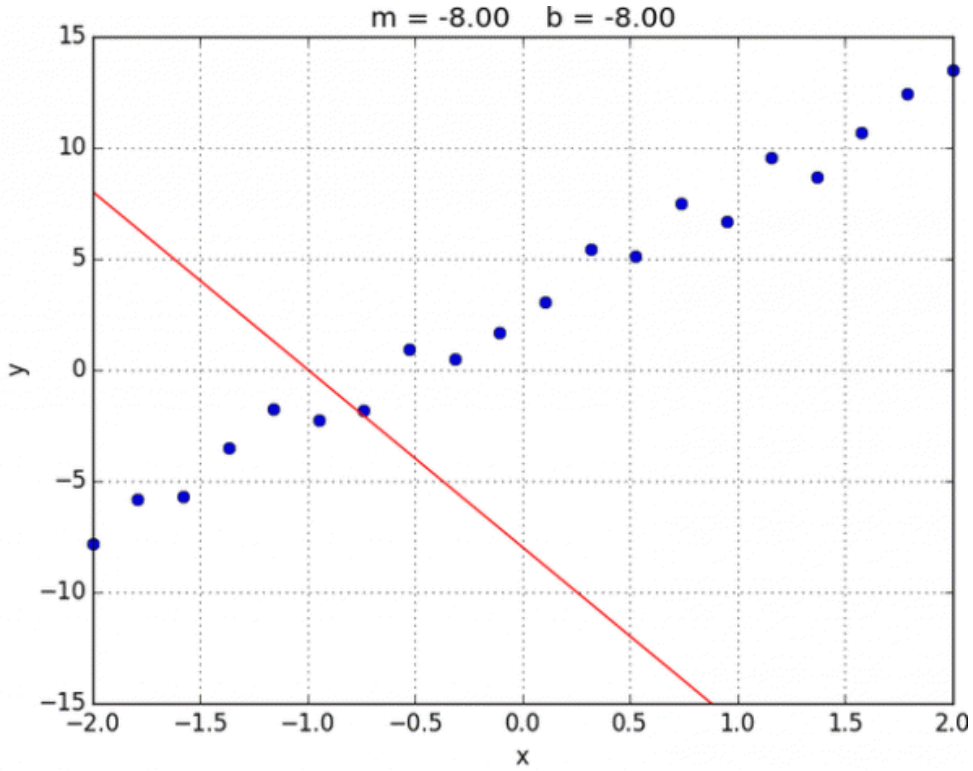




$m = w_1, b = w_0$

Gradient Descent

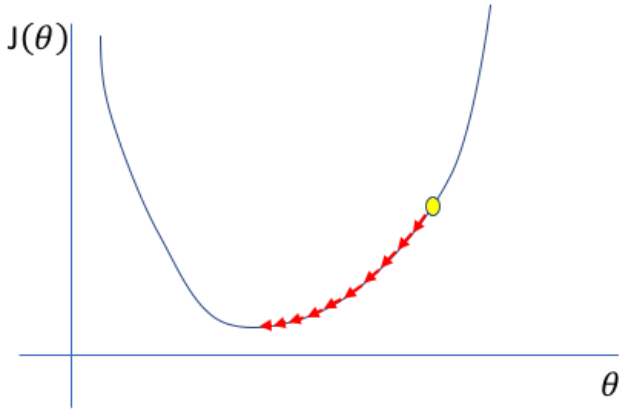




$$m = w_1, b = w_0$$

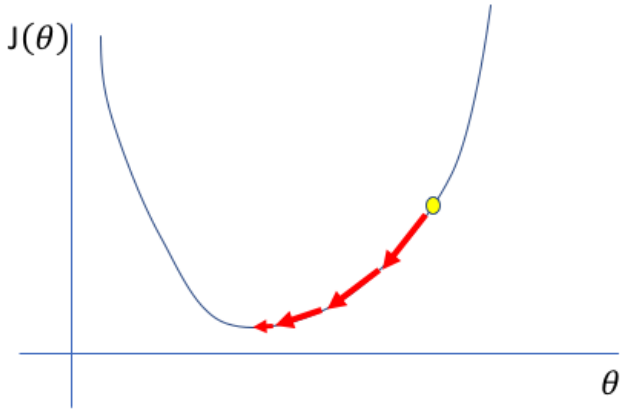
Gradient Descent - Effect of Learning Rate

Too low



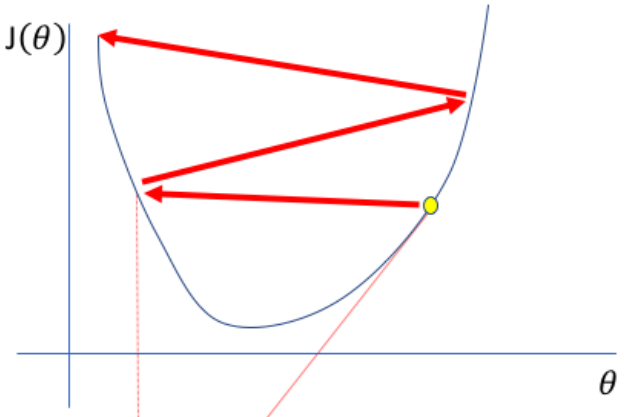
A small learning rate requires many updates before reaching the minimum point

Just right



The optimal learning rate swiftly reaches the minimum point

Too high



Too large of a learning rate causes drastic updates which lead to divergent behaviors