Machine Learning I

Mohamed Hussien

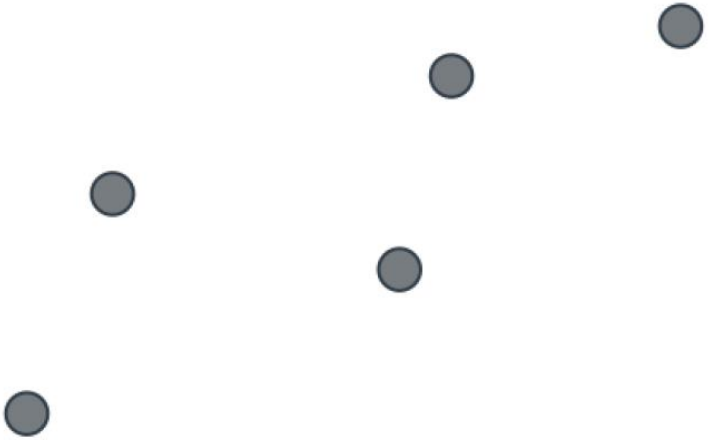
Linear Regression

Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

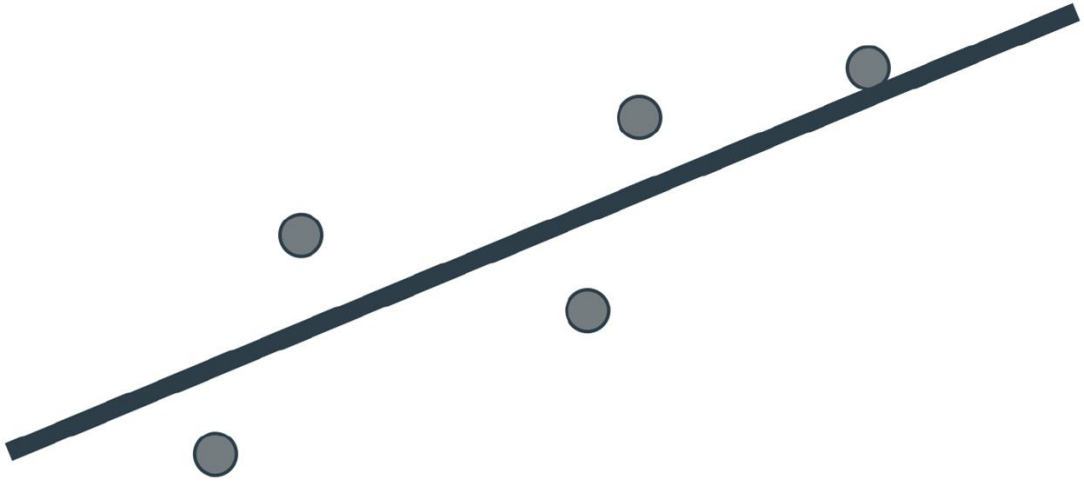
What is linear regression?

Can you design a linear road that pass by all these houses equally?!



What is linear regression?

Can you design a linear road that pass by all these houses equally?!



Housing prices problem

Number of rooms	Price
1	150
2	200
3	250
4	?
5	350
6	400
7	450

Housing prices problem

What are the features and labels here?

Is it classification or regression problem?

What is the price of the house of 4 rooms?

What is the price per extra room?(#Room **weight**)

What is the base price?(**Bias**)

What is equation that represent the price?

Price = **100** + **50** * (Number of rooms)

Number of rooms	Price
1	150
2	200
3	250
4	?
5	350
6	400
7	450

Lecture Overview

- **Problem Definition**
- **Bias and Weights**
- **How machines learn it?**
- **Multivariate Linear Regression**
- **Solve the problem graphically**
- **Solve the problem Mathematically**
- **Cost Function (Error)**
- **Gradient Descent**
- **Some training tricks**
- **SKlearn**
- **Normal Equation**

Weights and Bias

Price = 100 + 50 * (Number of rooms)

Weights and Bias

Price = 100 + 50 * (Number of rooms)

> WEIGHTS In the formula corresponding to the model, each feature gets multiplied by a corresponding factor. These factors are the weights. In the above formula the only feature is the number of rooms, and its corresponding weight is 50.

Weights and Bias

Price = 100 + 50 * (Number of rooms)

WEIGHTS In the formula corresponding to the model, each feature gets multiplied by a corresponding factor. These factors are the weights. In the above formula the only feature is the number of rooms, and its corresponding weight is 50.

BIAS As you can see, the formula corresponding to the model has a constant that is not attached to any of the features. This constant is called the bias. In this model, the bias is 100 and it corresponds to the base price of a house.

Weights and Bias

Price = 100 + 50 * (Number of rooms)

> WEIGHTS In the formula corresponding to the model, each feature gets multiplied by a corresponding factor. These factors are the weights. In the above formula the only feature is the number of rooms, and its corresponding weight is 50.

> BIAS As you can see, the formula corresponding to the model has a constant that is not attached to any of the features. This constant is called the bias. In this model, the bias is 100 and it corresponds to the base price of a house.

How machines learn this equation?

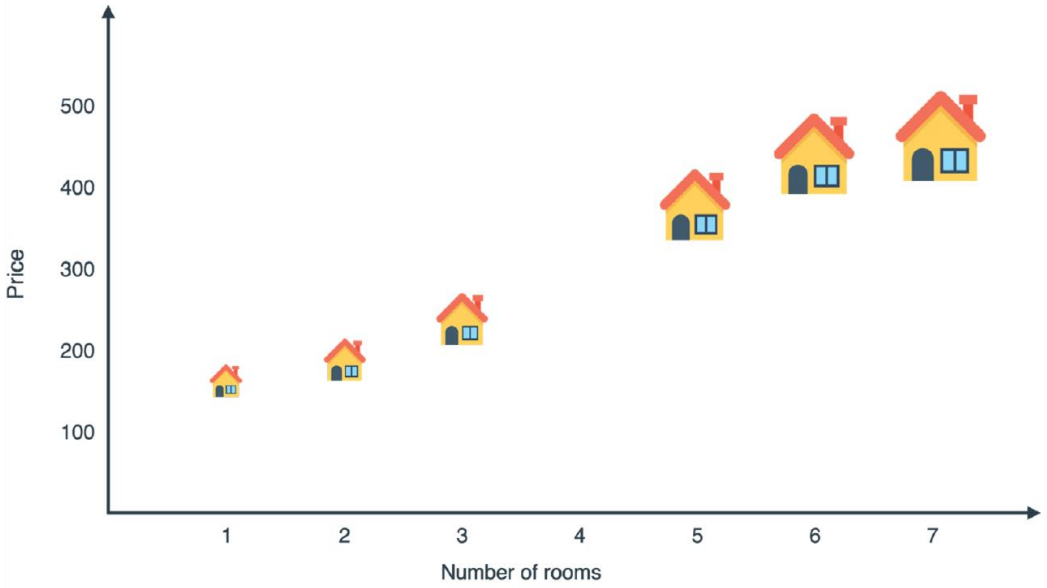
Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

How machines learn it? [**Remember**-Formulate-Predict]

Number of rooms	Price
1	155
2	197
3	244
4	?
5	356
6	407
7	448

How machines learn it? [**Remember**-Formulate-Predict]



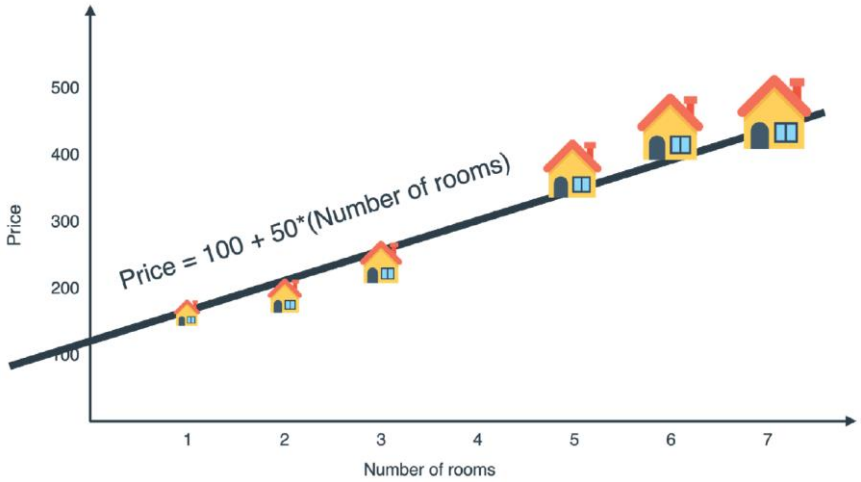
Number of rooms	Price
1	155
2	197
3	244
4	?
5	356
6	407
7	448

How machines learn it? [Remember-**Formulate**-Predict]

Price = 100 + 50 * (Number of rooms)

How machines learn it? [Remember-**Formulate**-Predict]

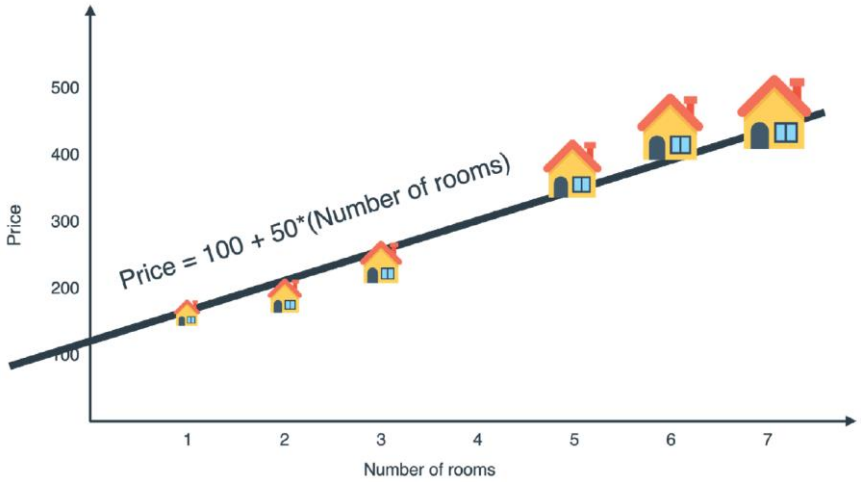
Price = 100 + 50 * (Number of rooms)



How machines learn it? [Remember-**Formulate**-Predict]

Price = 100 + 50 * (Number of rooms)

> Slope measures how steep the line is. It is calculated by dividing the rise over the run.

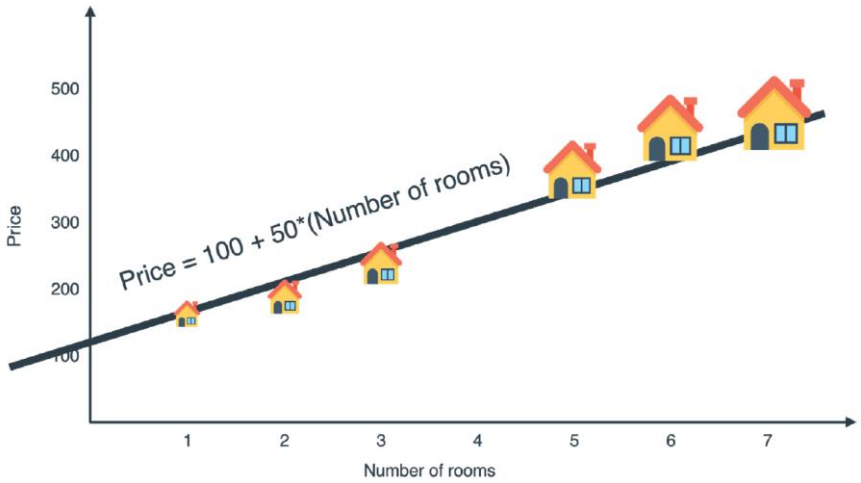


How machines learn it? [Remember-**Formulate**-Predict]

Price = 100 + 50 * (Number of rooms)

Slope measures how steep the line is. It is calculated by dividing the rise over the run.

Y-intercept is the height at which the line crosses the vertical (y) axis.



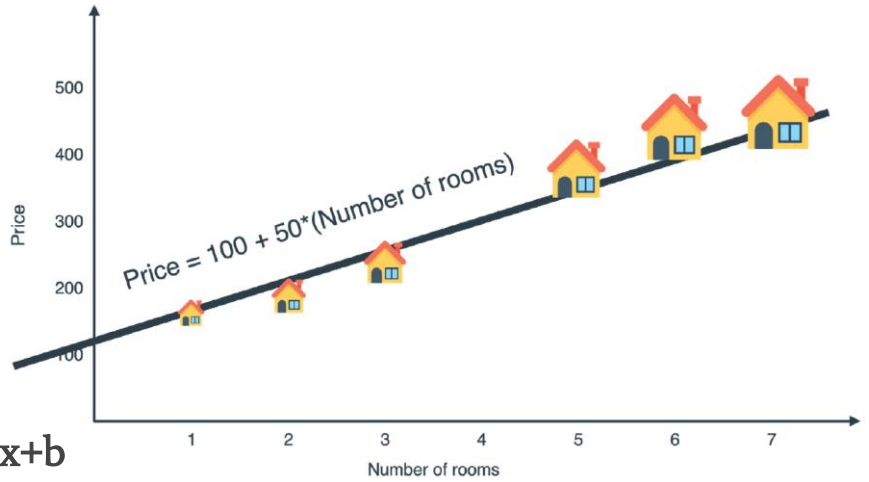
How machines learn it? [Remember-**Formulate**-Predict]

Price = 100 + 50 * (Number of rooms)

Slope measures how steep the line is. It is calculated by dividing the rise over the run.

Y-intercept is the height at which the line crosses the vertical (y) axis.

Linear Equation is the equation of a line. y=mx+b



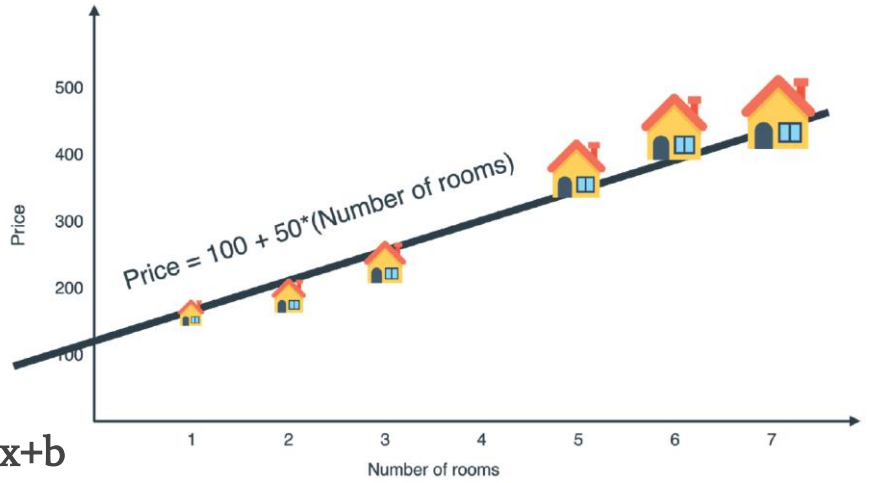
How machines learn it? [Remember-**Formulate**-Predict]

Price = 100 + 50 * (Number of rooms)

> Slope measures how steep the line is. It is calculated by dividing the rise over the run.

> Y-intercept is the height at which the line crosses the vertical (y) axis.

> **Linear Equation** is the equation of a line. y=mx+b



How many lines can solve the problem?

How machines learn it? [Remember-Formulate-**Predict**]

How machines learn it? [Remember-Formulate-**Predict**]

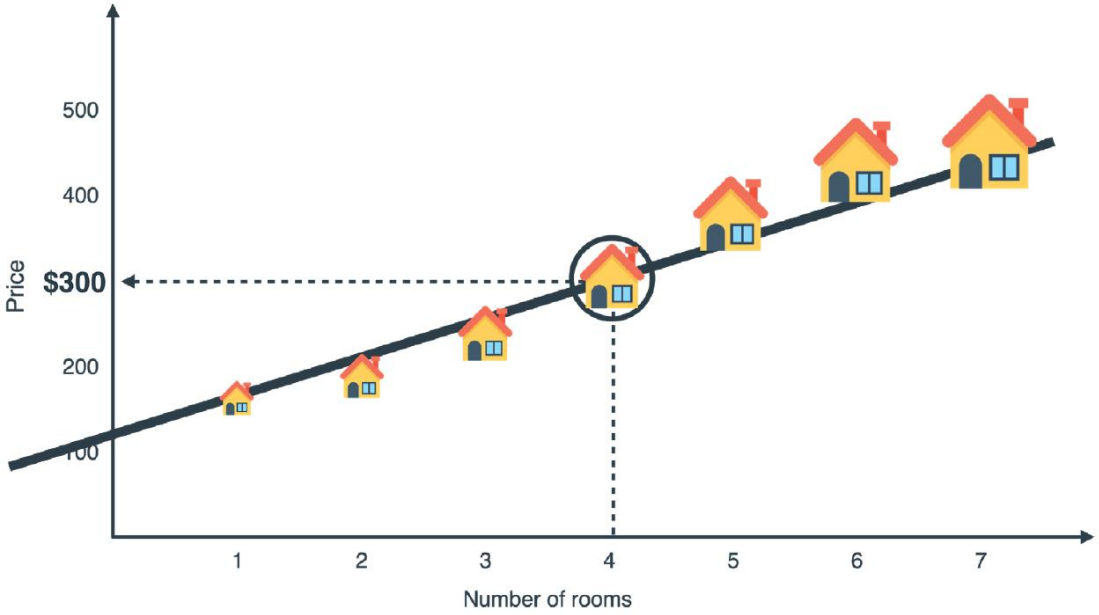
Price = 100 + 50 * (Number of rooms)

Price = 100 + 50 * (4)= 300

How machines learn it? [Remember-Formulate-**Predict**]

Price = 100 + 50 * (Number of rooms)

Price = 100 + 50 * (4)= 300



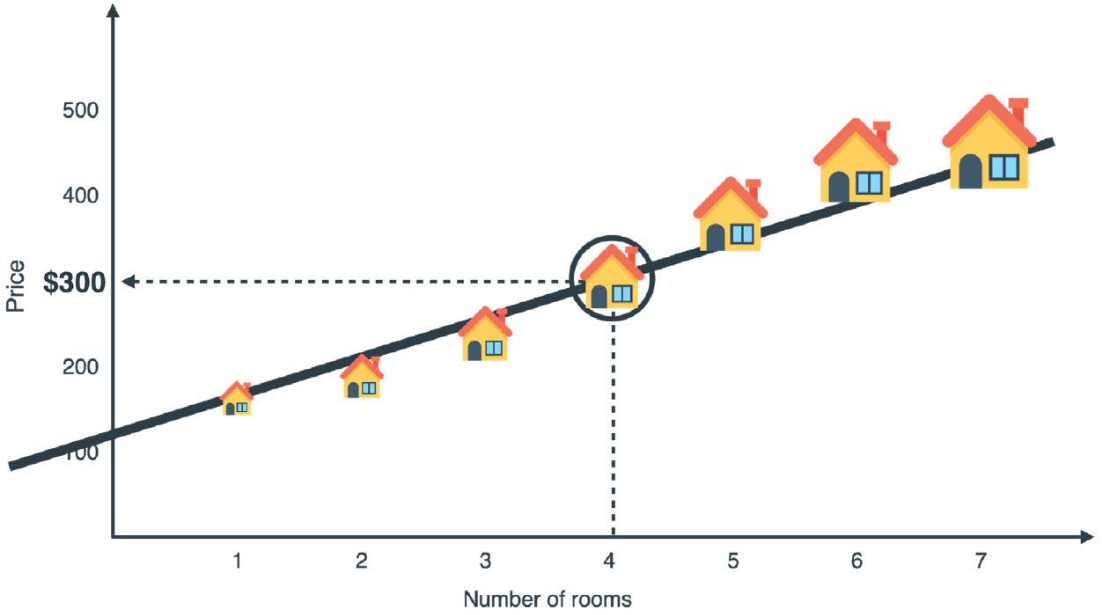
How machines learn it? [Remember-Formulate-**Predict**]

Price = 100 + 50 * (Number of rooms)

Price = 100 + 50 * (4)= 300

Some Questions:

- Can we have multiple features data?
- How does computer get this equation?



Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

Multivariate Linear Regression

Multivariate Linear Regression

Price = 30*(number of rooms) + 1.5*(size) + 10*(quality of the schools) - 2*(age of the house) + 50

What do you notice in this equation?

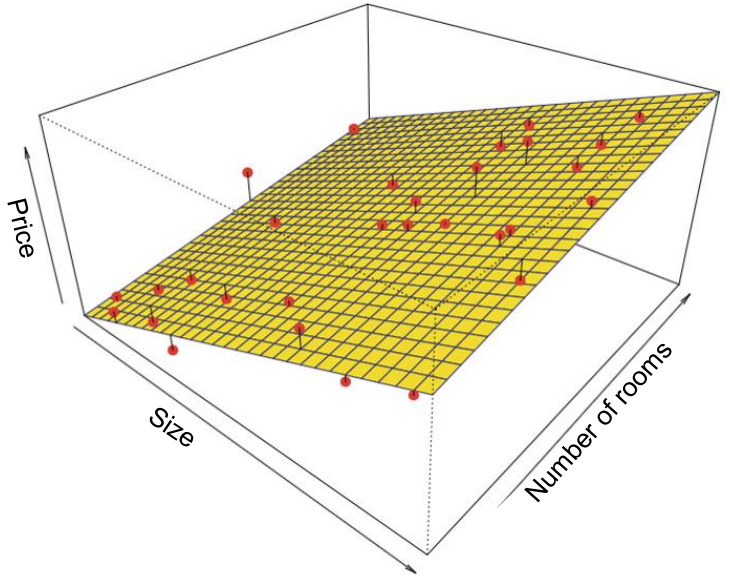
- 1 bias and multiple weights
- Different sign of weights
- Different weights value
- What is shape of the model? Still linear?

Multivariate Linear Regression

Price = 30*(number of rooms) + 1.5*(size) + 10*(quality of the schools) - 2*(age of the house) + 50

What do you notice in this equation?

- 1 bias and multiple weights
- Different sign of weights
- Different weights value
- What is shape of the model? Still linear?



Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

How machines formulate this equation?[Overview]

How machines formulate this equation?[Overview]

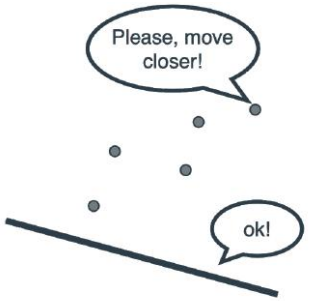
Inputs: A dataset of points.

Outputs: A linear regression model that fits that dataset.

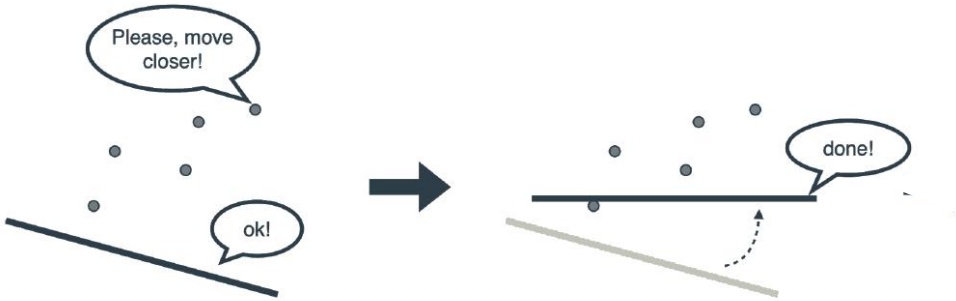
Procedure:

- Pick a model with random weights and a random bias.
- Repeat many times:
    - Pick a random data point.
    - Slightly adjust the weights(Slope) and bias(y-intercept) in order to improve the prediction for that particular data point.
- Return the model you've obtained.

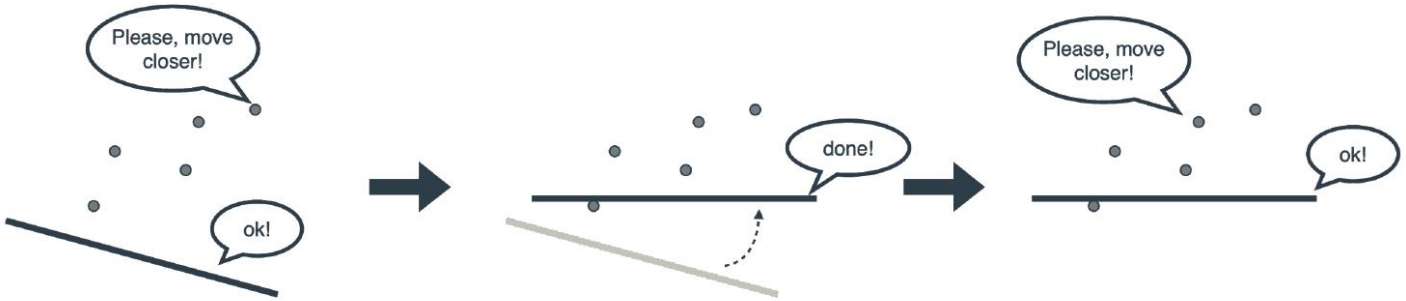
How machines formulate this equation?[Overview]



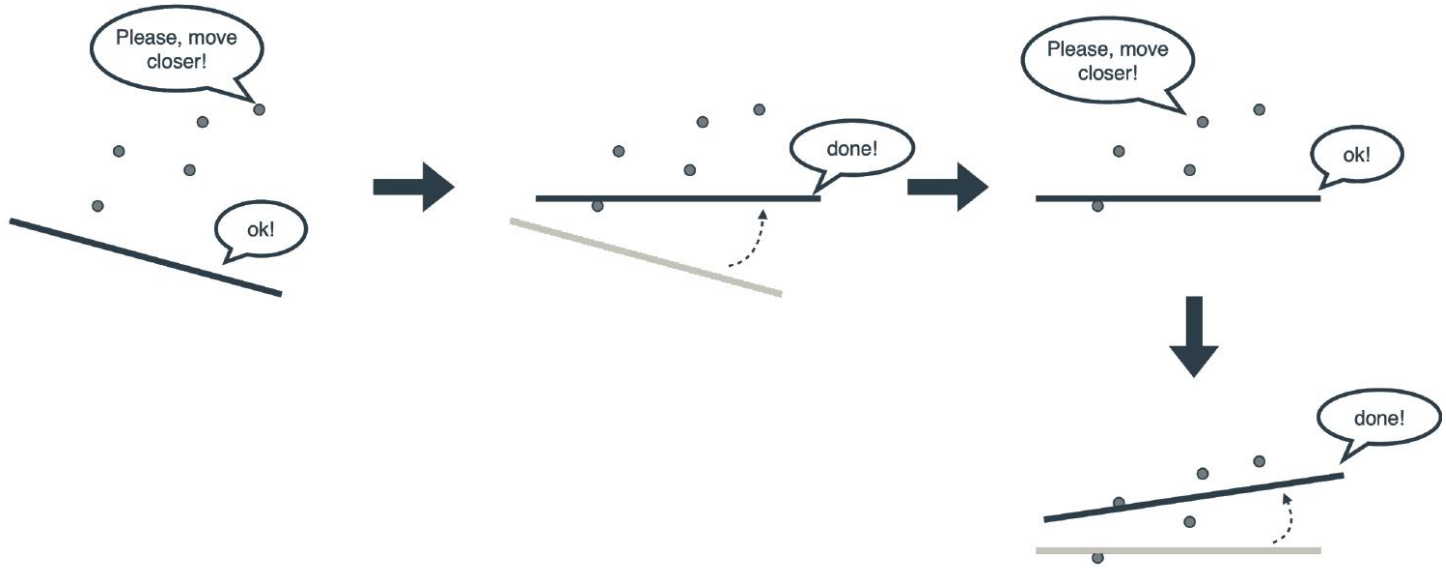
How machines formulate this equation?[Overview]



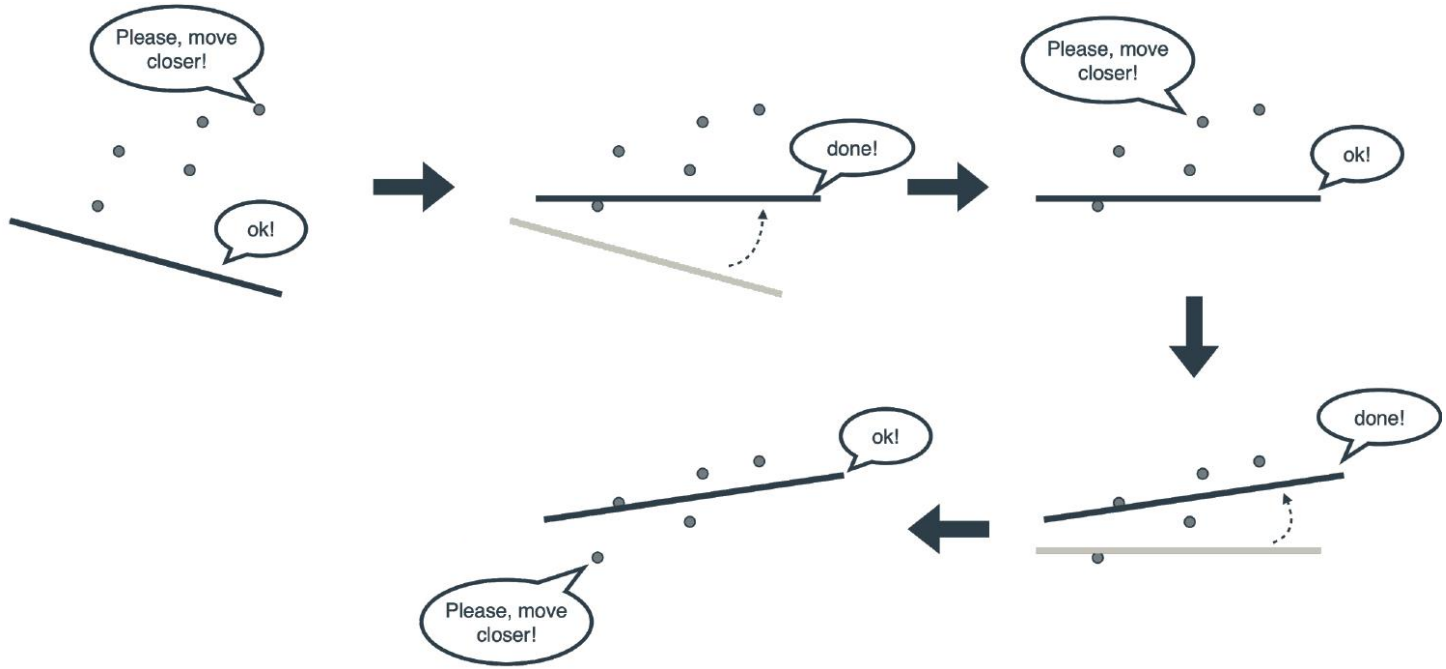
How machines formulate this equation?[Overview]



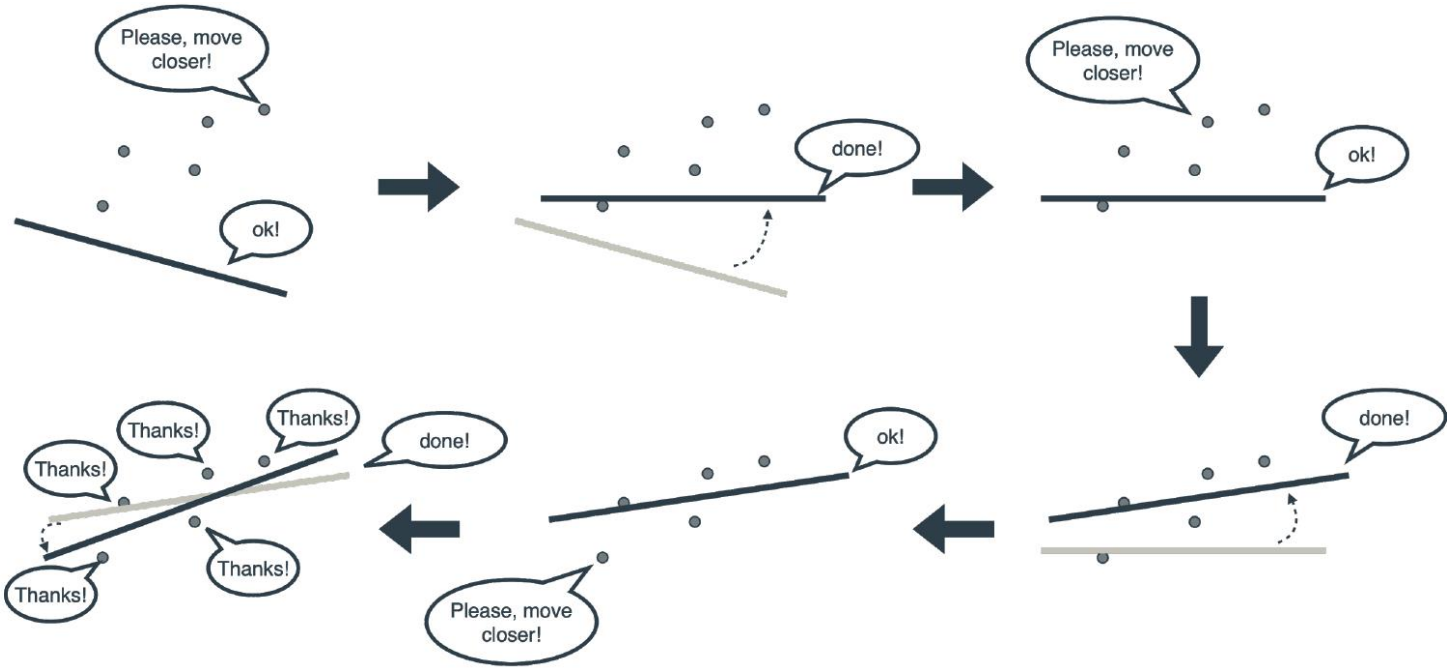
How machines formulate this equation?[Overview]



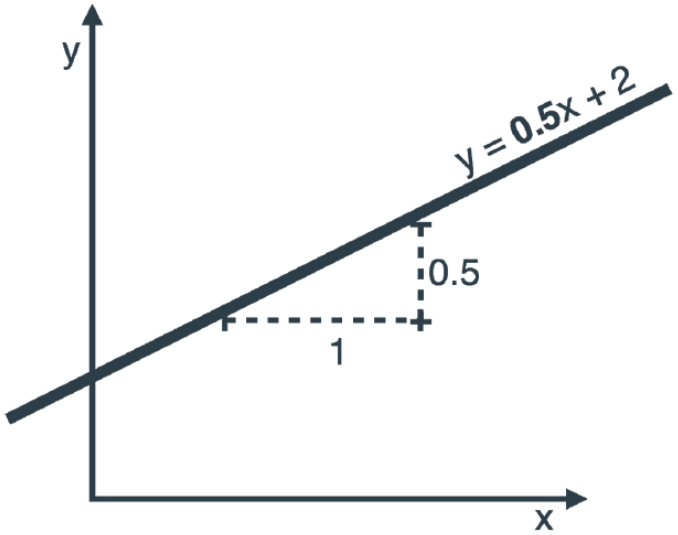
How machines formulate this equation?[Overview]



How machines formulate this equation?[Overview]

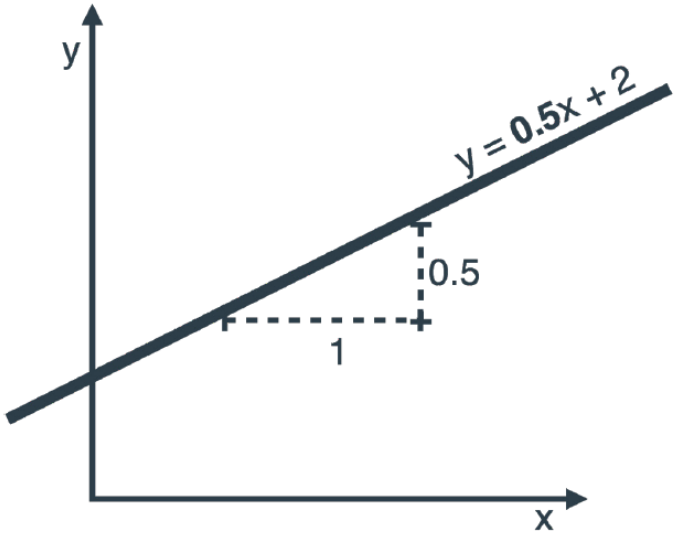


Slope and Y-intercept

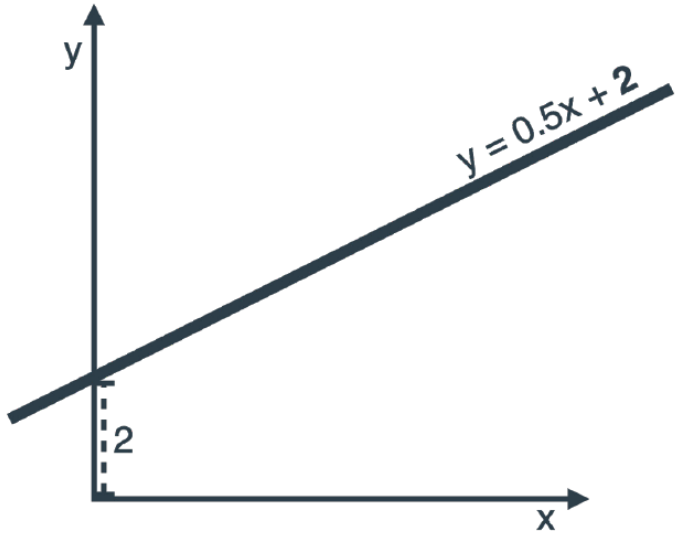


Slope = 0.5

Slope and Y-intercept

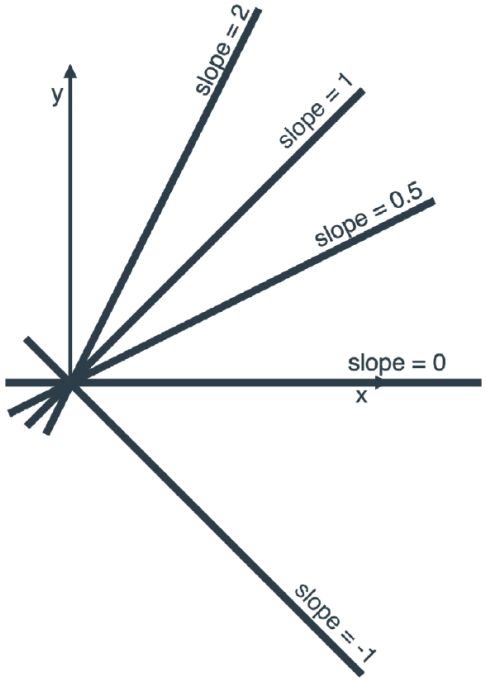


Slope = 0.5

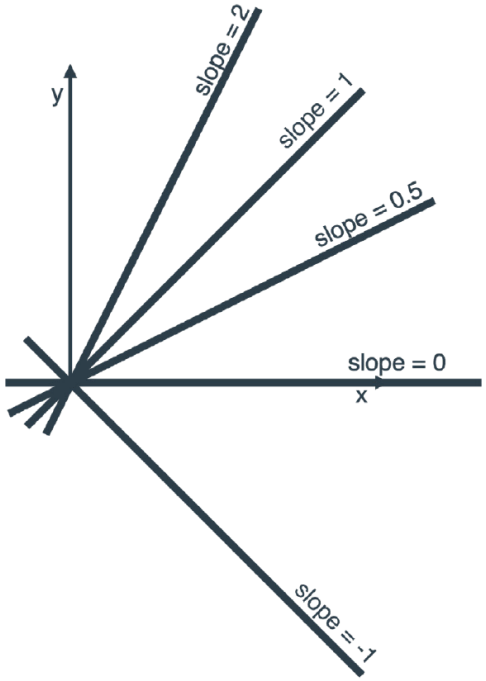


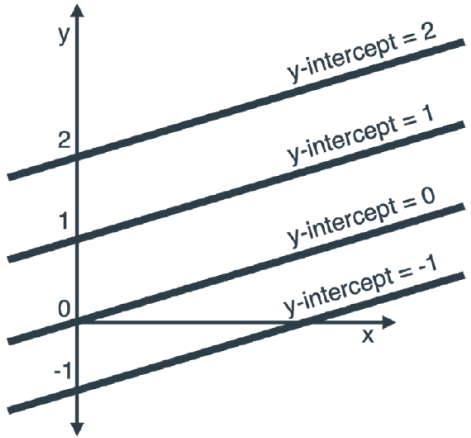
y-intercept = 2

Slope and Y-intercept

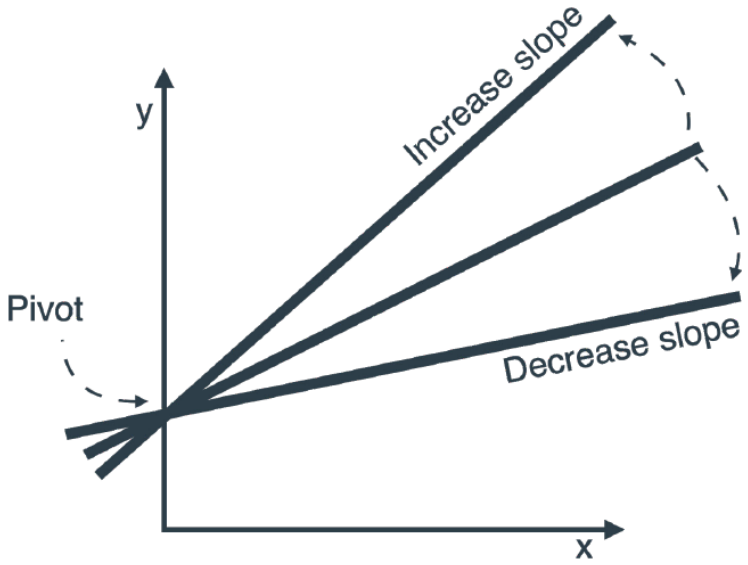


Slope and Y-intercept



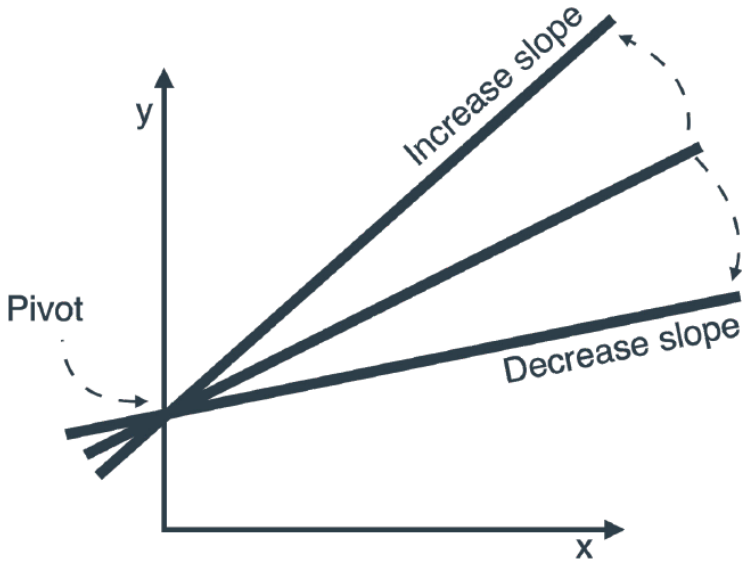


Slope and Y-intercept

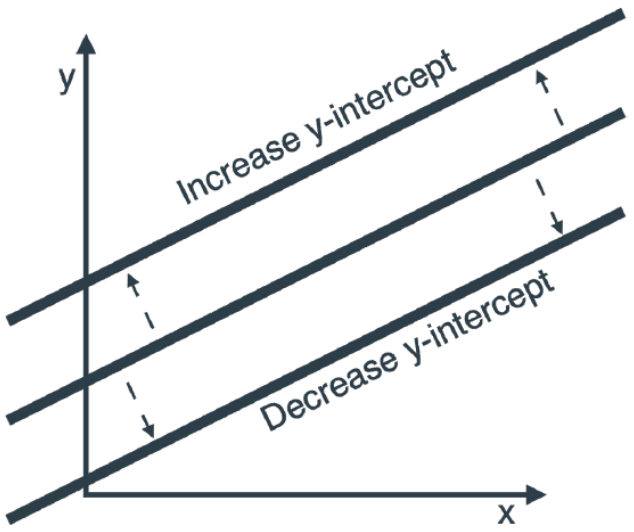


Rotate clockwise and counterclockwise

Slope and Y-intercept



Rotate clockwise and counterclockwise



Translate up and down

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

p: The price of a house in the dataset.

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

p: The price of a house in the dataset.

$\hat{p}$: The predicted price of a house.

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

p: The price of a house in the dataset.

$\hat{p}$: The predicted price of a house.

r: The number of rooms.

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

p: The price of a house in the dataset.

$\hat{p}$: The predicted price of a house.

r: The number of rooms.

m: The price per room.

Some variables of equation y=mx+b

$$\hat{p} = mr + b$$

p: The price of a house in the dataset.

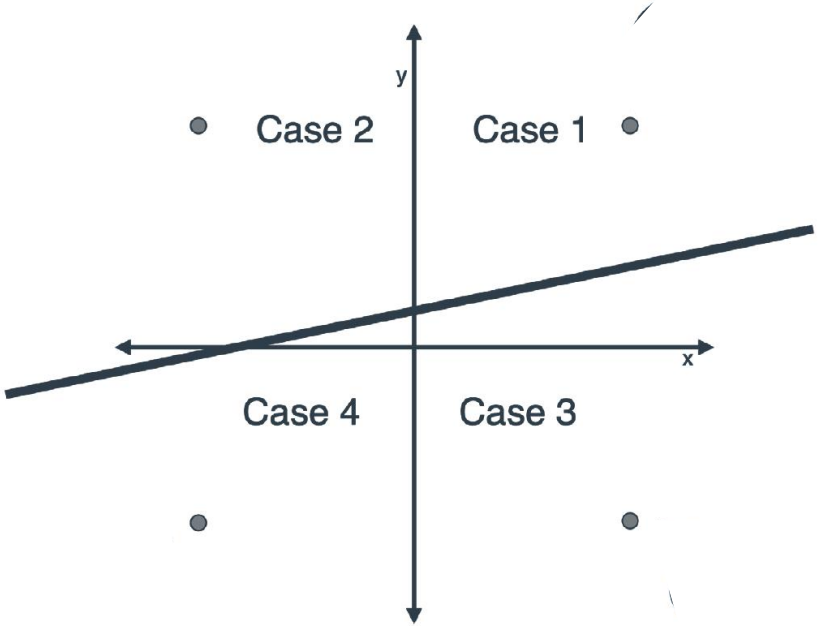
$\hat{p}$: The predicted price of a house.

r: The number of rooms.

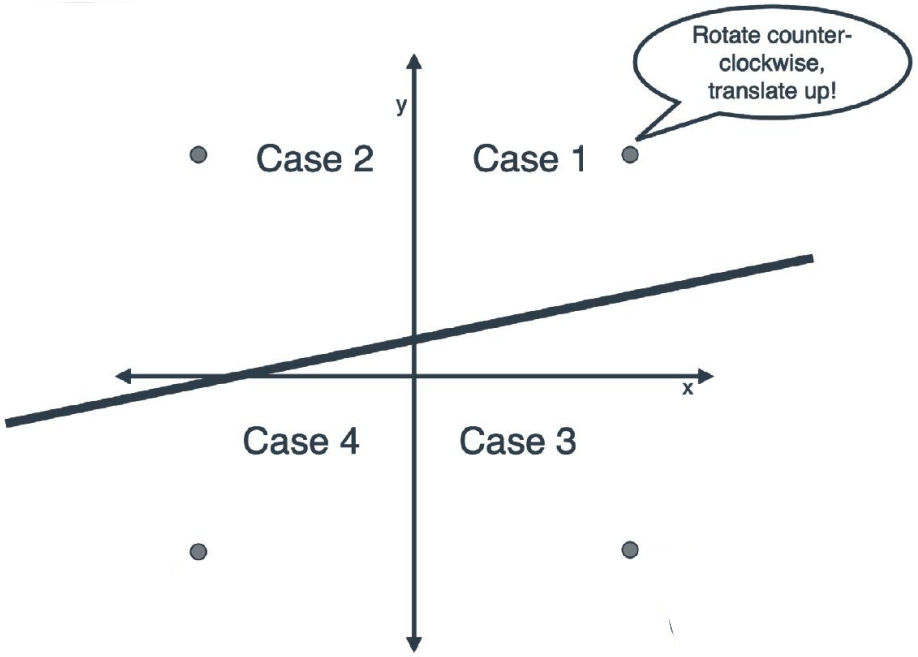
m: The price per room.

b: The base price for a house.

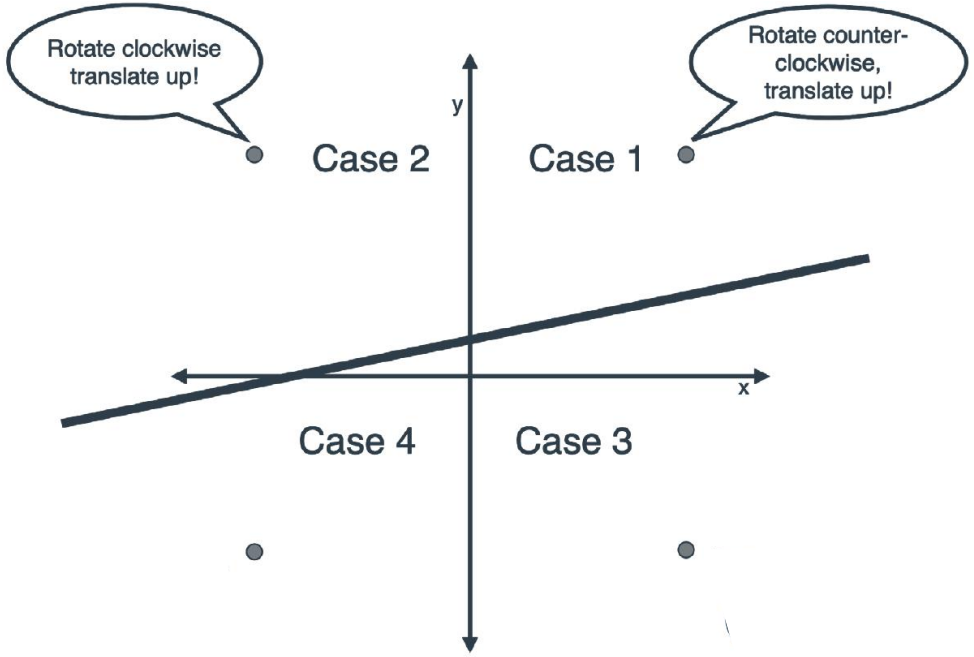
How machines formulate this equation?[The simple trick]



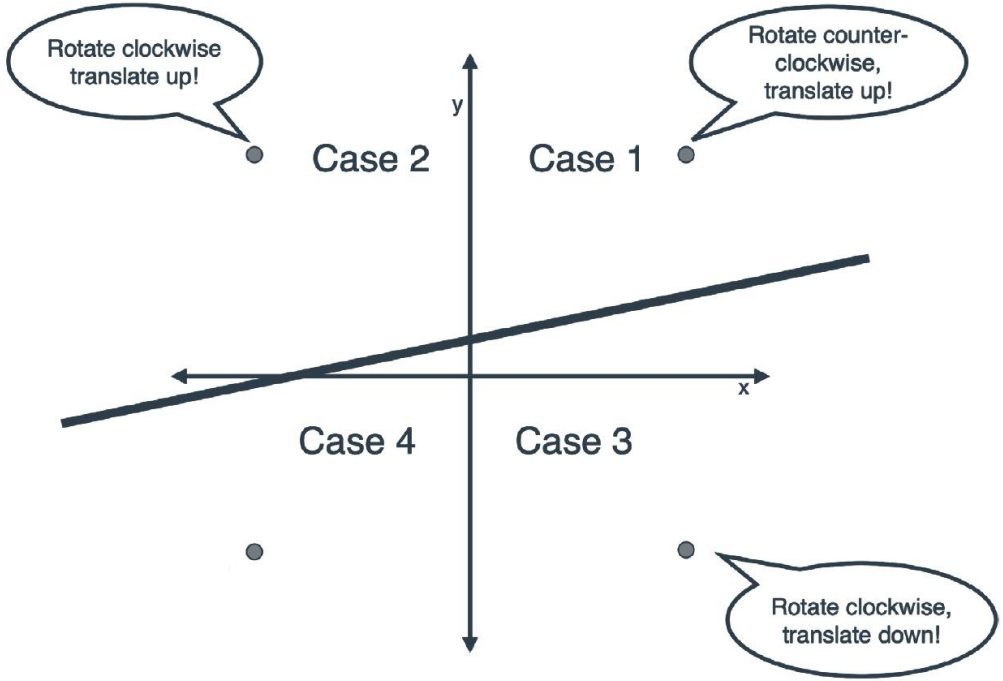
How machines formulate this equation?[The simple trick]



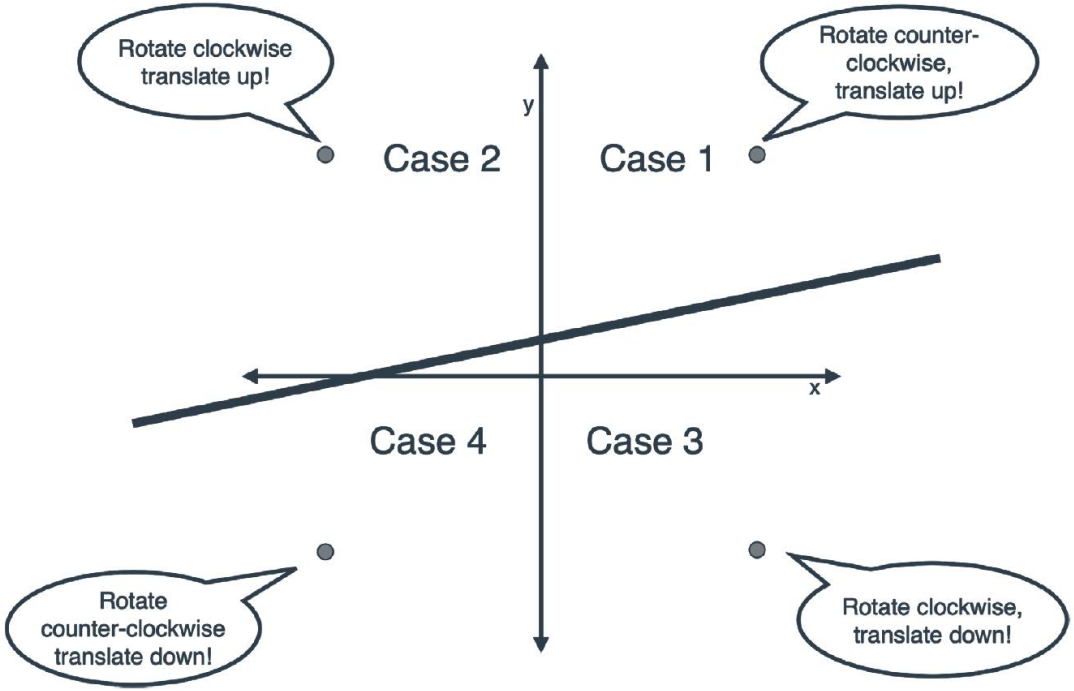
How machines formulate this equation?[The simple trick]



How machines formulate this equation?[The simple trick]



How machines formulate this equation?[The simple trick]



How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

Case 1: If the point is above the line and to the right of the y-axis, we rotate the line counterclockwise and translate it upwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

Case 1: If the point is above the line and to the right of the y-axis, we rotate the line counterclockwise and translate it upwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 2: If the point is above the line and to the left of the y-axis, we rotate the line clockwise and translate it upwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

Case 1: If the point is above the line and to the right of the y-axis, we rotate the line counterclockwise and translate it upwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 2: If the point is above the line and to the left of the y-axis, we rotate the line clockwise and translate it upwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 3: If the point is below the line and to the right of the y-axis, we rotate the line clockwise and translate it downwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Subtract η_2 to the y-intercept b. Obtain $b' - \eta_2$.

How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

Case 1: If the point is above the line and to the right of the y-axis, we rotate the line counterclockwise and translate it upwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 2: If the point is above the line and to the left of the y-axis, we rotate the line clockwise and translate it upwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 3: If the point is below the line and to the right of the y-axis, we rotate the line clockwise and translate it downwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Subtract η_2 to the y-intercept b. Obtain $b' - \eta_2$.

Case 4: If the point is below the line and to the left of the y-axis, we rotate the line counterclockwise and translate it downwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Subtract η_2 to the y-intercept b. Obtain $b' - \eta_2$.

Return: The line with equation $\hat{p} = m'r + b'$.

How machines formulate this equation?[The simple trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).

η1, η2 are learning rate

Output:

- A line with equation $\hat{p} = m'r + b$ that is closer to the point.

Procedure:

Pick two very small random numbers, call them η_1 and η_2 (the Greek letter 'eta').

Case 1: If the point is above the line and to the right of the y-axis, we rotate the line counterclockwise and translate it upwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 2: If the point is above the line and to the left of the y-axis, we rotate the line clockwise and translate it upwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Add η_2 to the y-intercept b. Obtain $b' + \eta_2$.

Case 3: If the point is below the line and to the right of the y-axis, we rotate the line clockwise and translate it downwards.

- Subtract η_1 to the slope m. Obtain $m' - \eta_1$.
- Subtract η_2 to the y-intercept b. Obtain $b' - \eta_2$.

Case 4: If the point is below the line and to the left of the y-axis, we rotate the line counterclockwise and translate it downwards.

- Add η_1 to the slope m. Obtain $m' + \eta_1$.
- Subtract η_2 to the y-intercept b. Obtain $b' - \eta_2$.

Return: The line with equation $\hat{p} = m'r + b'$.

What is learning rate?

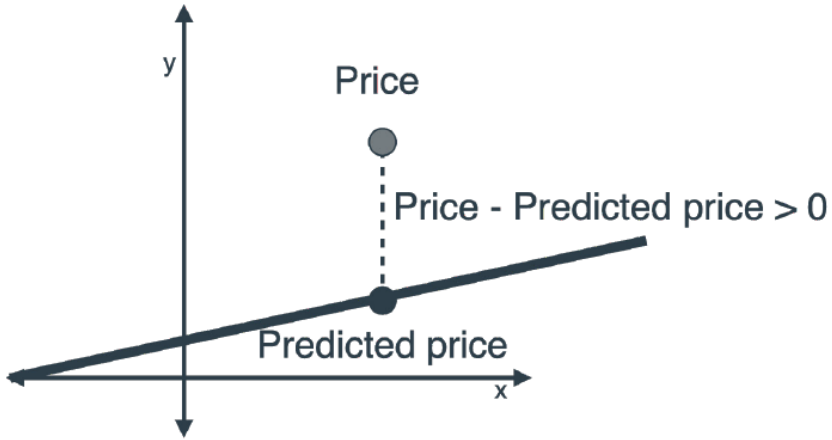
> A very small number that we pick before training our model. This number will help us make sure our model changes in very small amounts by training.

How machines formulate this equation?[The square trick]

Bias:

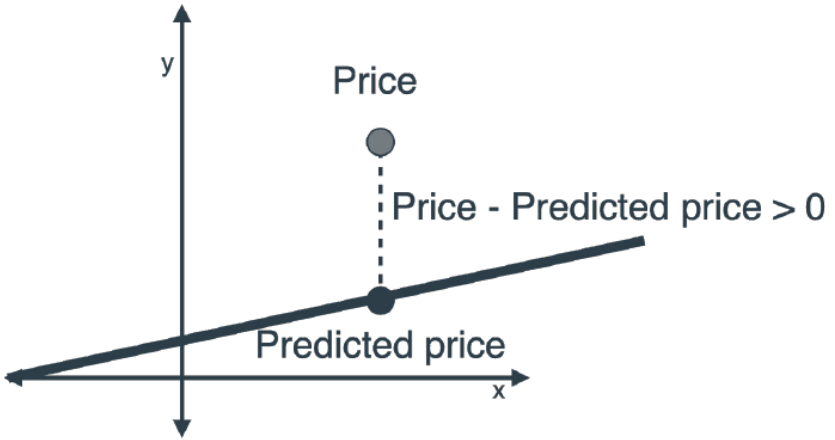
How machines formulate this equation?[The square trick]

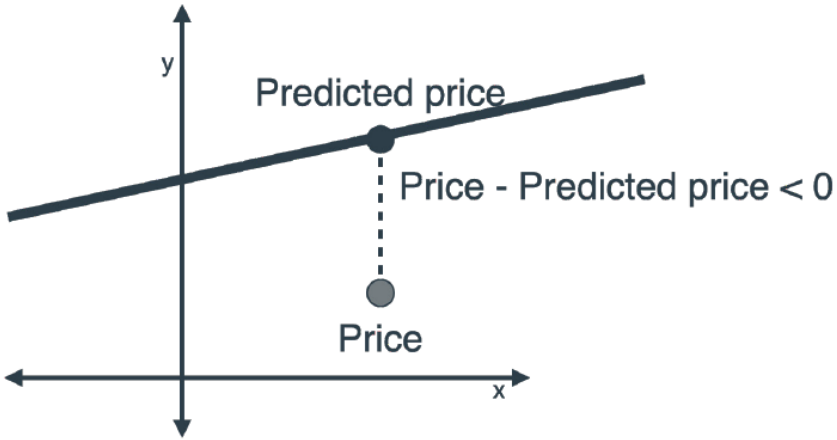
Bias:



How machines formulate this equation?[The square trick]

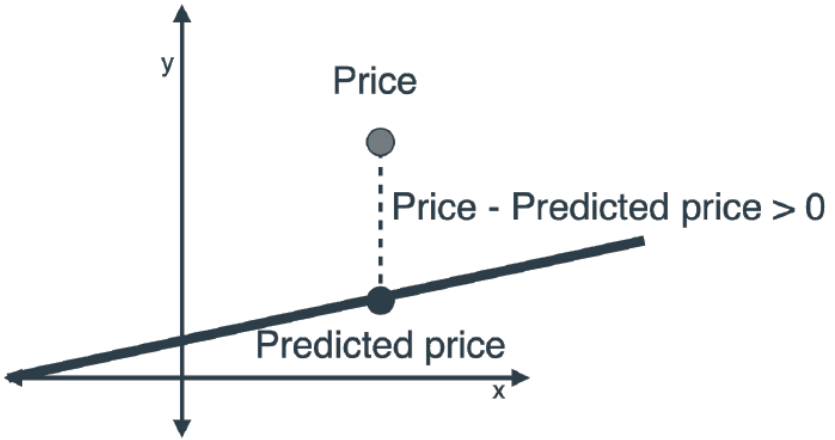
Bias:

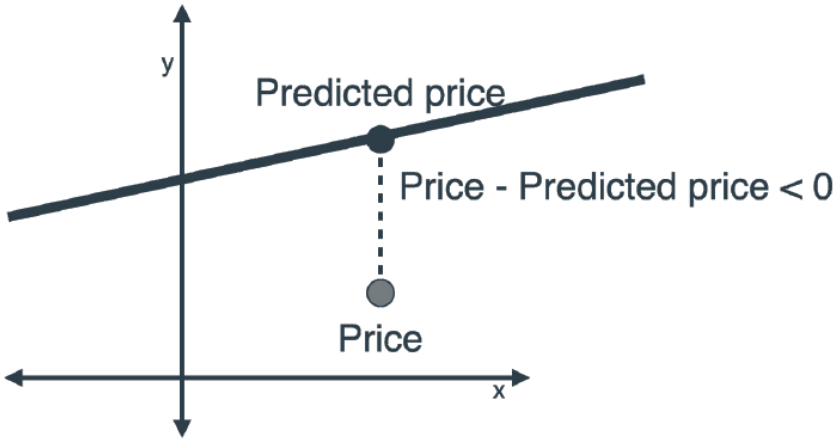




How machines formulate this equation?[The square trick]

Bias:

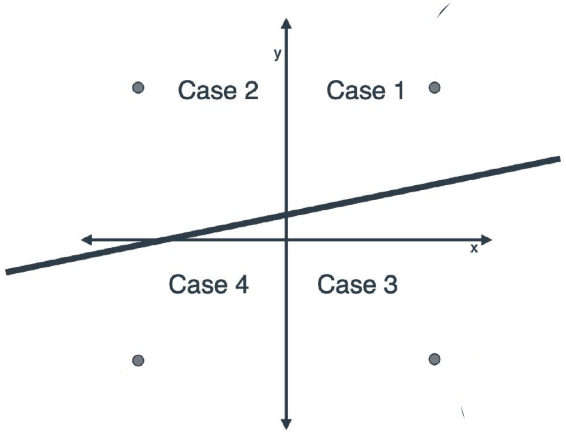




$$b' = b + \eta(p - \bar{p})$$

How machines formulate this equation?[The square trick]

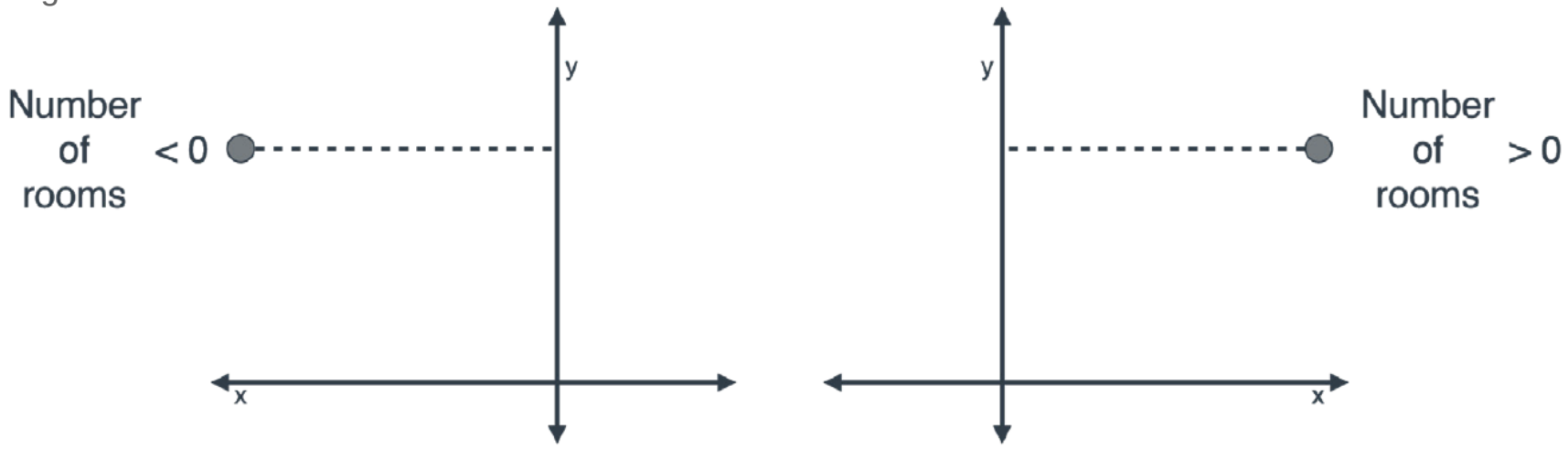
Weight:



$$m' = m + \eta \ (p - \bar{p})$$

How machines formulate this equation?[The square trick]

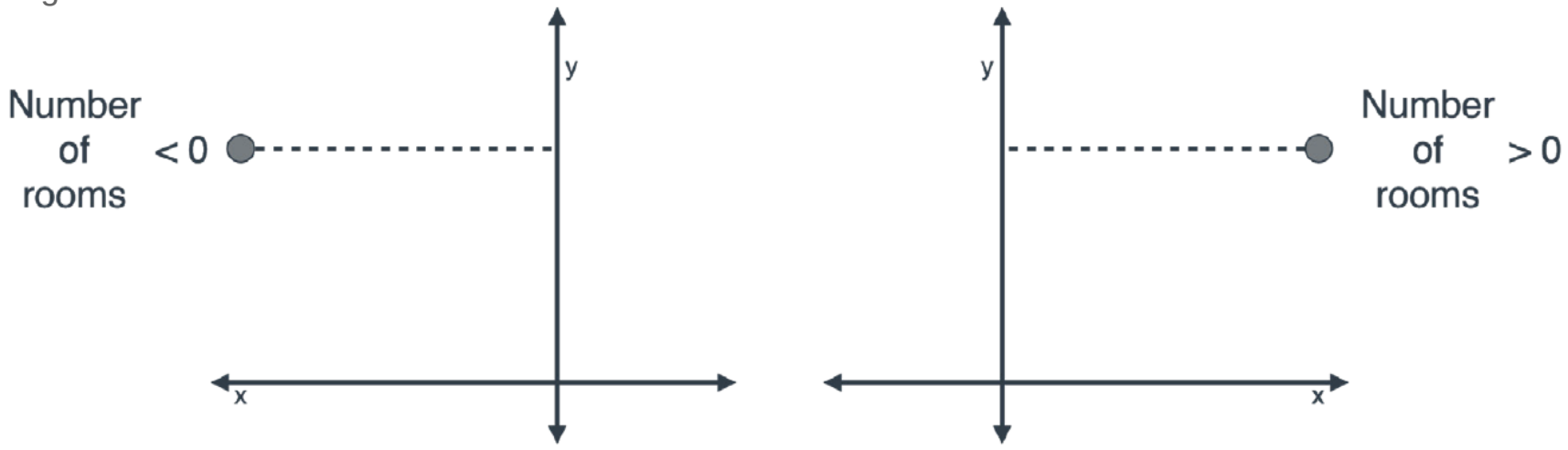
Weight:



$$m' = m + \eta \ (p - \bar{p})$$

How machines formulate this equation?[The square trick]

Weight:



$$m' = m + \eta r(p - \bar{p})$$

How machines formulate this equation?[The square trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).
- A small positive value η (the learning rate).

Output:

- A line with equation $\hat{p} = m'r + b'$. that is closer to the point.

Procedure:

- Add $\eta(p - \hat{p})$ to the y-intercept b. Obtain $b' = b + \eta(p - \hat{p})$ (this translates the line).
- Add $\eta r(p - \hat{p})$ to the slope m. Obtain $m' = m + \eta r(p - \hat{p})$ (this rotates the line).

Return: The line with equation $\hat{p} = m'r + b'$.

How machines formulate this equation?[The square trick]

Use colab to open this github notebook:

“s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb”

How machines formulate this equation?[The absolute trick]

Square Trick

$$b' = b + \eta(p - \overline{p})$$

$$m' = m + \eta r(p - \overline{p})$$

What if we want to
remove (P-P') part?

How machines formulate this equation?[The absolute trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).
- A small positive value η (the learning rate).

Output:

- A line with equation $\hat{p} = m'r + b'$. that is closer to the point.

Square Trick

$$b' = b + \eta(p - \overline{\overline{p}})$$

$$m' = m + \eta r(p - \overline{\overline{p}})$$

What if we want to remove (P-P') part?

How machines formulate this equation?[The absolute trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).
- A small positive value η (the learning rate).

Output:

- A line with equation $\hat{p} = m'r + b'$. that is closer to the point.

Procedure:

Case 1: If the point is above the line (i.e., if $p > \hat{p}$).

- Add η to the y-intercept b. Obtain $b' = b + \eta$ (this translates the line up).
- Add ηr to the slope m. Obtain $m' = m + \eta r$ (this rotates the line counterclockwise if the point is to the right of the y-axis, and clockwise if it is to the left of the y-axis).

Square Trick

$$b' = b + \eta(p - \overline{\overline{p}})$$

$$m' = m + \eta r(p - \overline{\overline{p}})$$

What if we want to remove (P-P') part?

How machines formulate this equation?[The absolute trick]

Inputs:

- A line with slope m, y-intercept b, and equation $\hat{p} = mr + b$.
- A point with coordinates (r, p).
- A small positive value η (the learning rate).

Output:

- A line with equation $\hat{p} = m'r + b'$. that is closer to the point.

Procedure:

Case 1: If the point is above the line (i.e., if $p > \hat{p}$).

- Add η to the y-intercept b. Obtain $b' = b + \eta$ (this translates the line up).
- Add ηr to the slope m. Obtain $m' = m + \eta r$ (this rotates the line counterclockwise if the point is to the right of the y-axis, and clockwise if it is to the left of the y-axis).

Case 2: If the point is below the line (i.e., if $p < \hat{p}$).

- Subtract η to the y-intercept b. Obtain $b' = b - \eta$ (this translates the line down).
- Subtract ηr to the slope m. Obtain $m' = m - \eta r$ (this rotates the line clockwise if the point is to the right of the y-axis, and counterclockwise if it is to the left of the y-axis).

Return: The line with equation $\hat{p} = m'r + b'$.

Here is the code for the absolute trick.

Square Trick

$$b' = b + \eta(p - \overline{\overline{p}})$$

$$m' = m + \eta r(p - \overline{\overline{p}})$$

What if we want to remove (P-P') part?

How machines formulate this equation?[The absolute trick]

Use colab to open this github notebook:

“s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb”

How machines formulate this equation?[The full algorithm]

Inputs:

- A dataset of houses with number of rooms and prices.

Outputs:

- Model weights: price per room and base price.

Procedure:

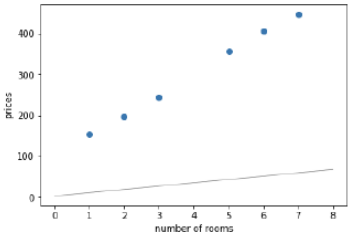
- Start with random values for the slope and y-intercept
- Repeat many times:
    - Pick a random data point
    - Update the slope and the y-intercept using the absolute or the square trick.

How machines formulate this equation?[The full algorithm]

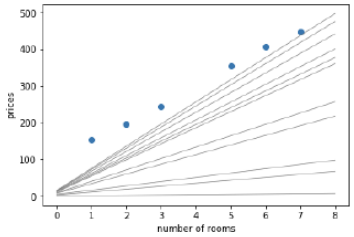
Use colab to open this github notebook:

“s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb”

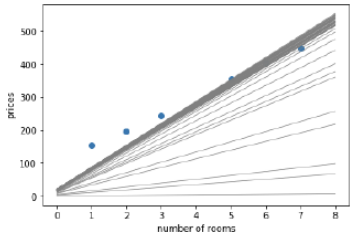
How machines formulate this equation?[The full algorithm]



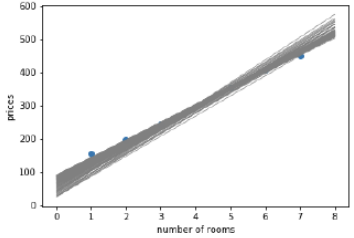
Starting point



Epochs 1-10



Epochs 1-50



Epochs 51-10,000

How machines formulate this equation?[Multivariate]

How machines formulate this equation?[Multivariate]

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

How machines formulate this equation?[Multivariate]

Size (feet^2) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

How machines formulate this equation?[Multivariate]

Size (feet^2) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

How machines formulate this equation?[Multivariate]

	Size (feet^2) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
	1416	3	2	40	232
	1534	3	2	30	315
	852	2	1	36	178

How machines formulate this equation?[Multivariate]

	Size (feet^2) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178

How machines formulate this equation?[Multivariate]

feature →

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178

How machines formulate this equation?[Multivariate]

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
feature →	X_1	X_2	X_3	X_4	Y
data sample → $X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

How machines formulate this equation?[Multivariate]

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
feature →	X_1	X_2	X_3	X_4	Y
data sample → $X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

X * W + b = Y'

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$\begin{bmatrix} 2104 & 5 & 1 & 45 \end{bmatrix}$$

X * W + b = Y'

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
	X_1	X_2	X_3	X_4	Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$\begin{bmatrix} 2104 & 5 & 1 & 45 \end{bmatrix} \quad X \quad * W \quad \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} \quad + b \quad = Y'$$

How machines formulate this equation?[Multivariate]

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
feature →	X_1	X_2	X_3	X_4	Y
data sample → $X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$\begin{bmatrix} 2104 & 5 & 1 & 45 \end{bmatrix} \quad * W \quad \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} \quad + b \quad = Y' \qquad \begin{bmatrix} y'_1 \end{bmatrix}$$

X

How machines formulate this equation?[Multivariate]

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
feature →	X_1	X_2	X_3	X_4	Y
data sample → $X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \end{bmatrix}$$

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
	X_1	X_2	X_3	X_4	Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features]

How machines formulate this equation?[Multivariate]

feature → X_1 | X_2 | X_3 | X_4 | Y

data sample →

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1]

How machines formulate this equation?[Multivariate]

feature

data sample

	Size (feet²)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
	X_1	X_2	X_3	X_4	Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

How machines formulate this equation?[Multivariate]

The general case will consist of a dataset of m points and n features.

- The data points are $(x^{(1)}, x^{(2)}, \ldots, x^{(m)})$. Each point is of the form $x^{(i)} = (x_1{}^{(i)}, x_2{}^{(i)}, \ldots, x_n{}^{(i)})$.
- The corresponding labels are $y_1, y_2, \ldots, y_m$.
- The weights of the model are $w_1, w_2, \ldots, w_n$.
- The bias of the model is b

$$X = \begin{bmatrix} — (x^{(1)})^T — \\ — (x^{(2)})^T — \\ \vdots \\ — (x^{(m)})^T — \end{bmatrix} \quad \vec{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(m)} \end{bmatrix}.$$

m*n m*1

How machines formulate this equation?[Multivariate]

Inputs:

- A model with equation $\hat{y} = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$.
- A point with coordinates (x, y).
- A small positive value η (the learning rate).

Output:

- A model with equation $\hat{y} = w_1' x_1 + w_2' x_2 + \cdots + w_n' x_n + b'$ that is closer to the point.

Procedure:

- Add $\eta(y - \hat{y})$ to the y-intercept b. Obtain $b' = b + \eta(y - \hat{y})$.
- For $i = 1, 2, \ldots, n$:
  - Add $\eta x(y - \hat{y})$ to the weight w_i. Obtain $w_i' = w_i + \eta r(y - \hat{y})$.

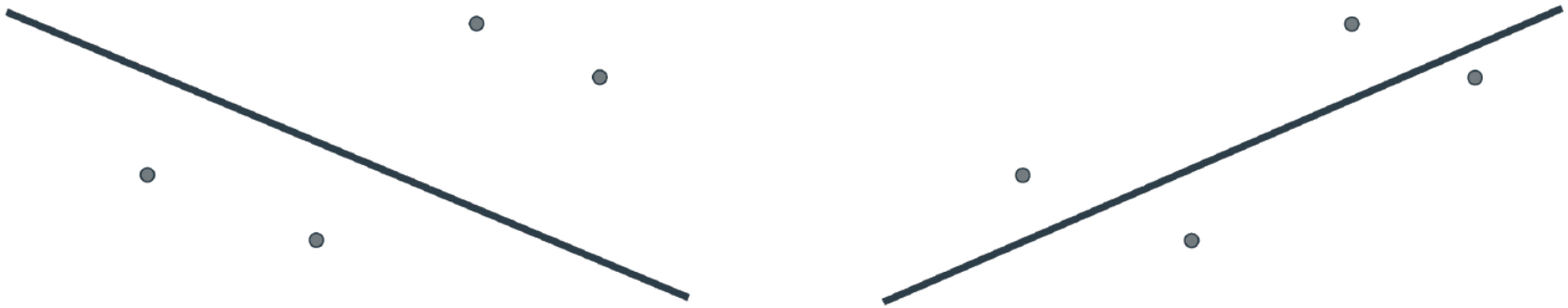
Return: The model with equation $\hat{y} = w_1' x_1 + w_2' x_2 + \cdots + w_n' x_n + b'$.

Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

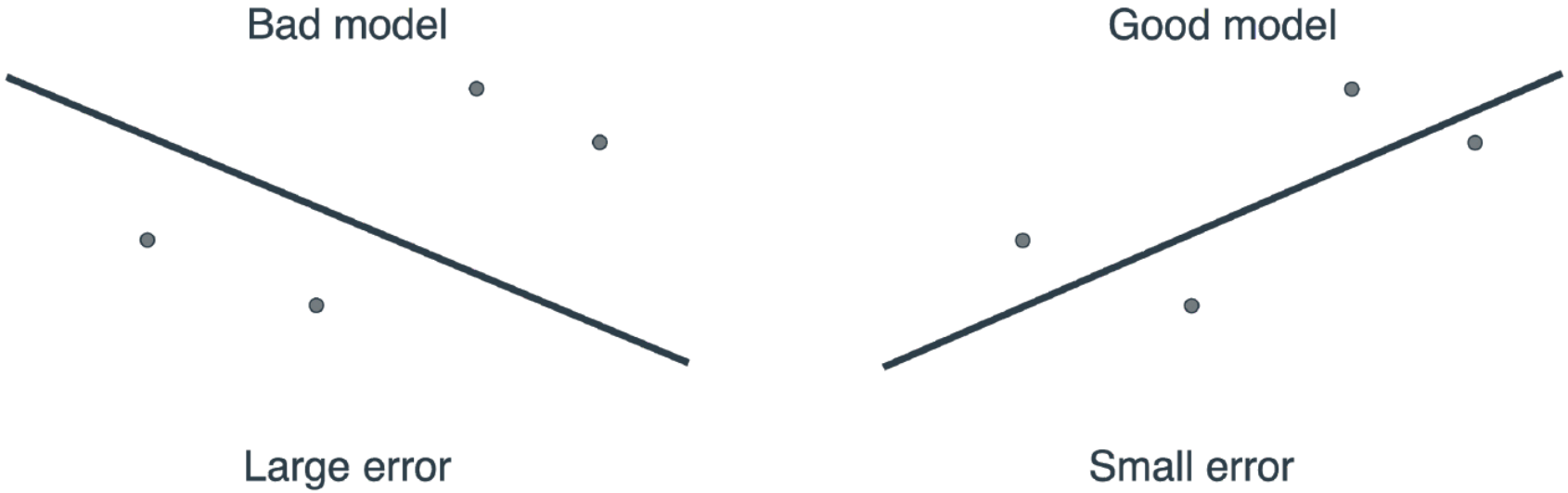
Is our model good or bad?

Which model is better and why?

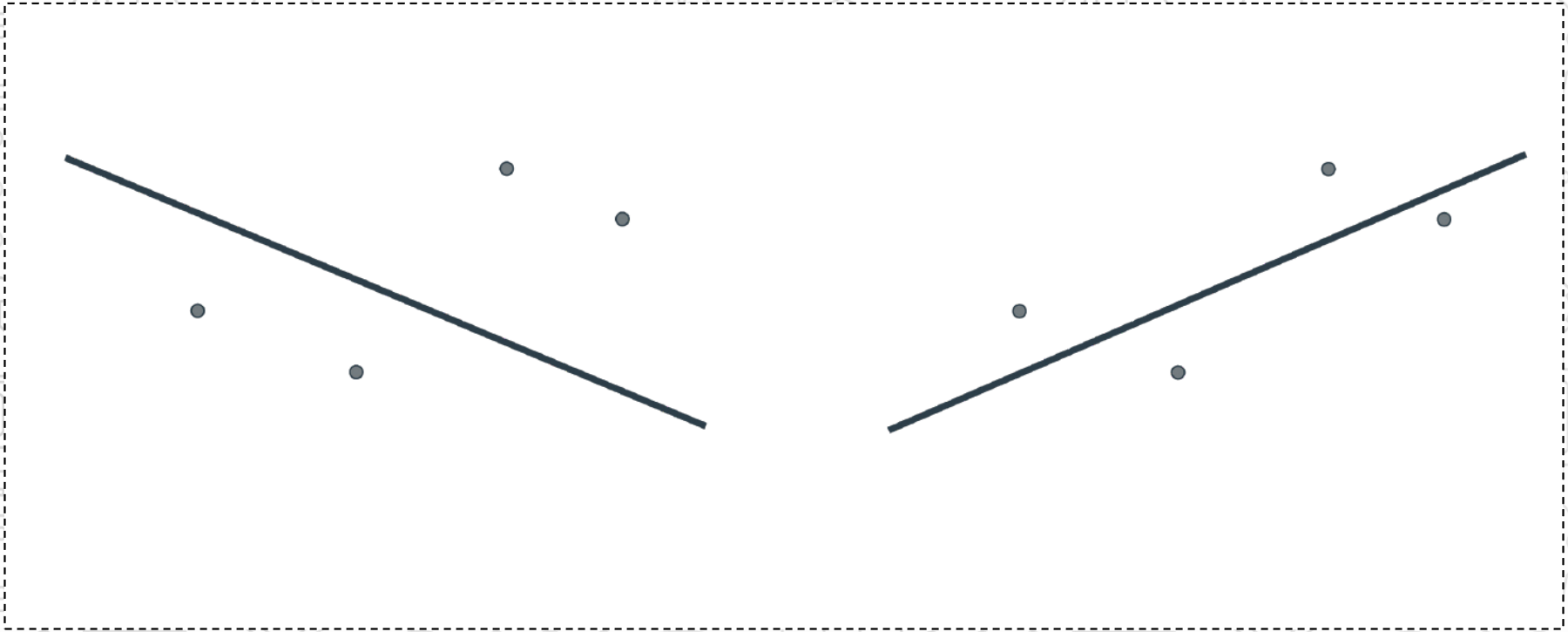


Is our model good or bad?

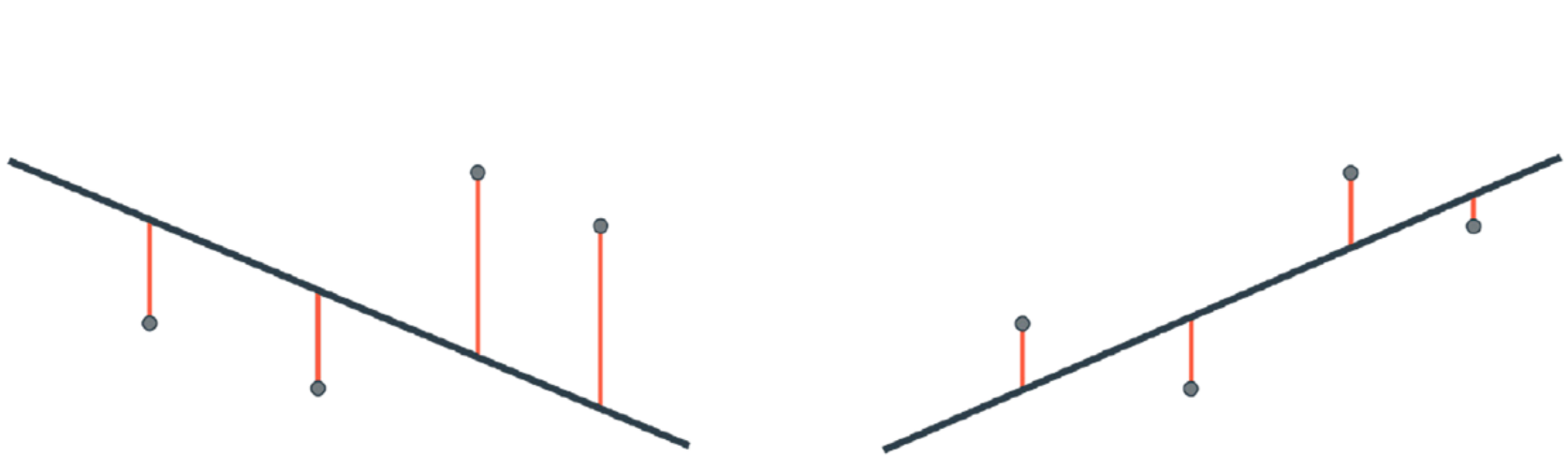
Which model is better and why?



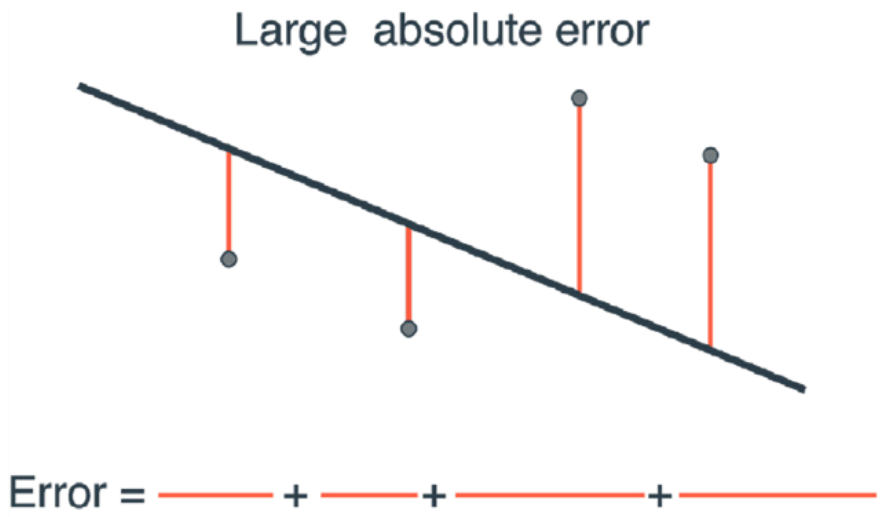
Absolute Error

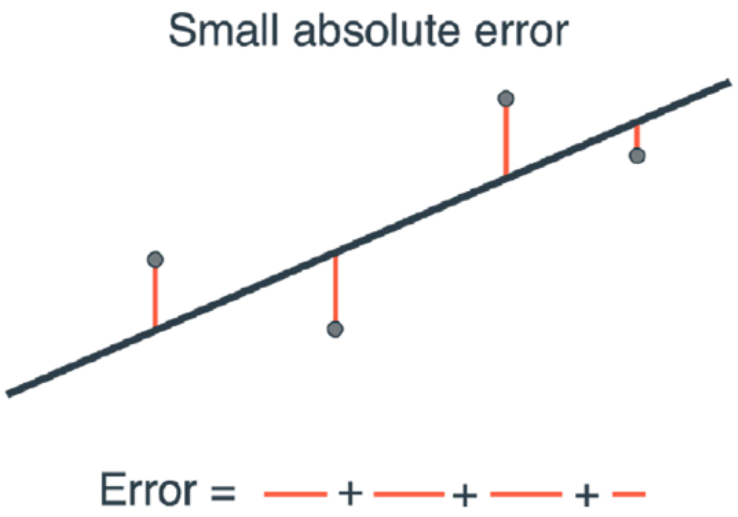


Absolute Error

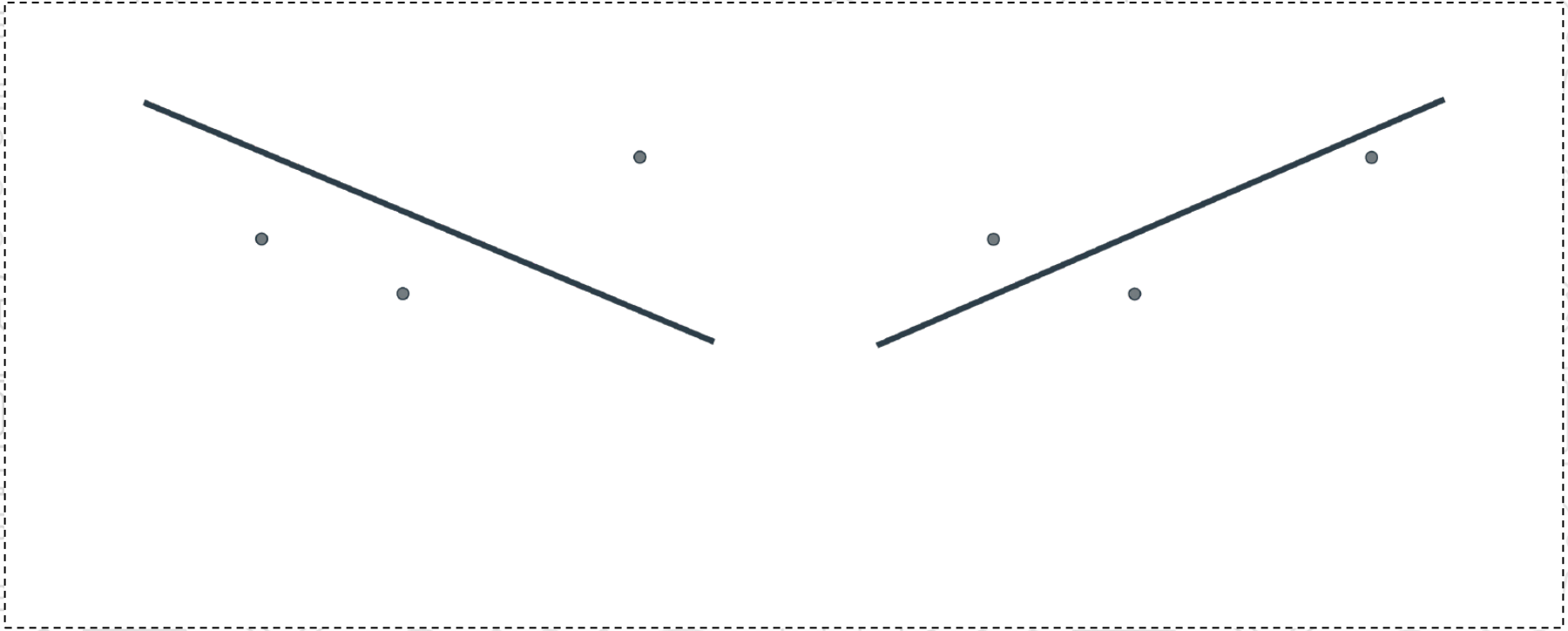


Absolute Error

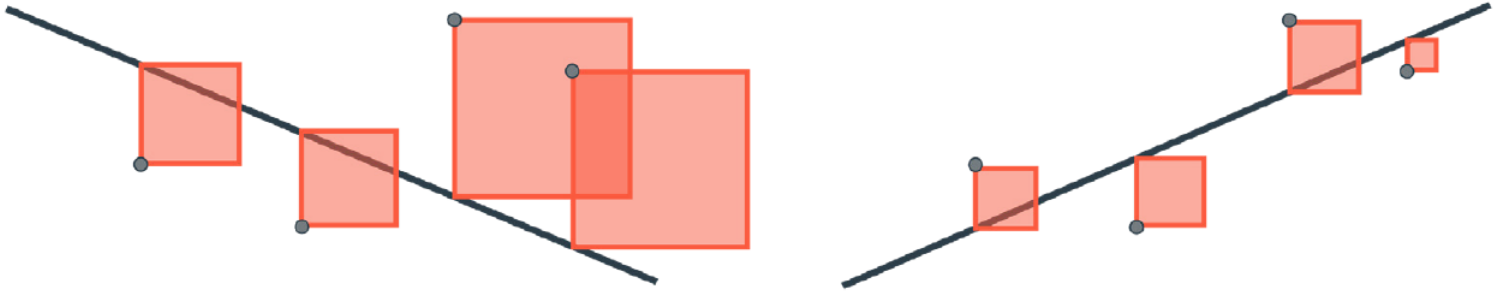




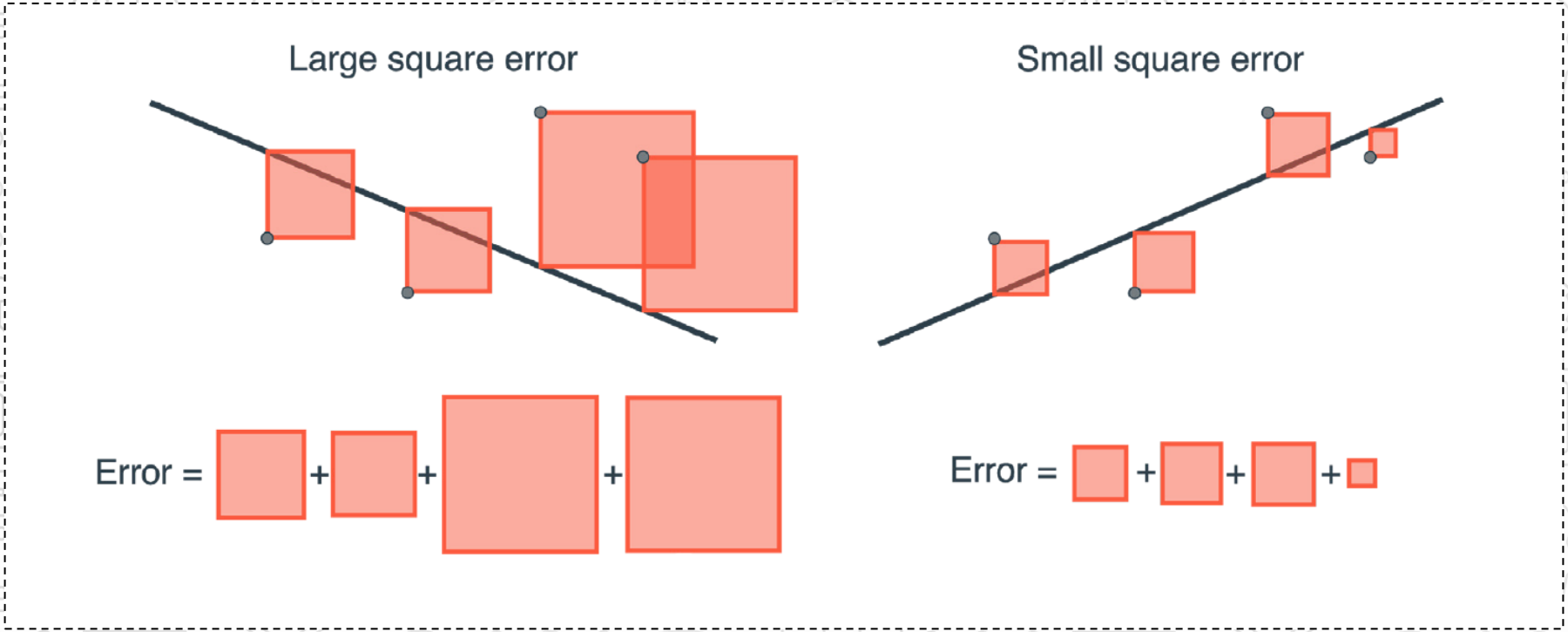
Square Error



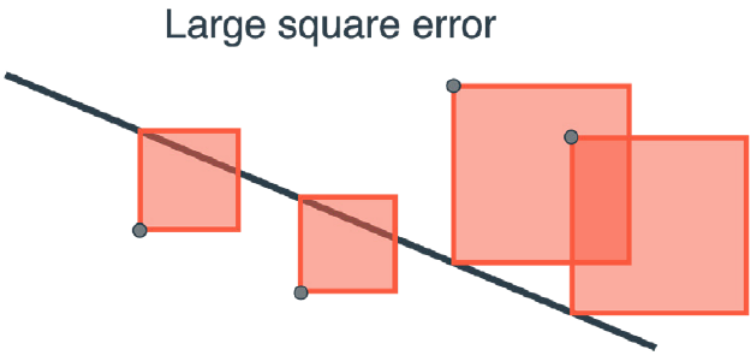
Square Error

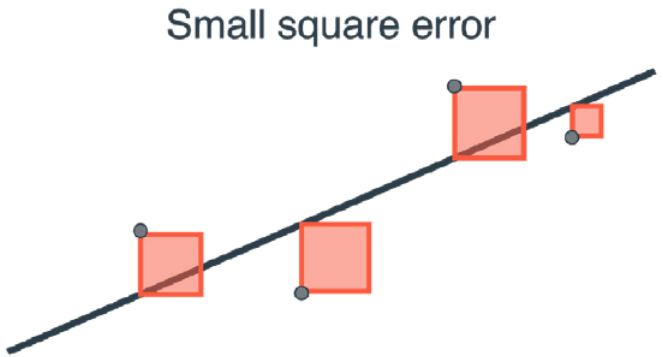


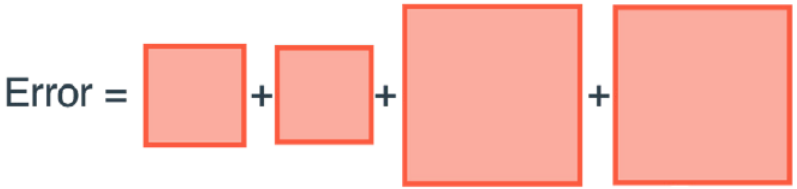
Square Error



Square Error







Which error function is better?

What is the best error (cost) function?

- Absolute Error
  |P'-P|
  Enough to measure the error
- Square Error
  $(P'-P)^2$
  Nicer derivative
- Mean Square Error
  $1/n\ (P'-P)^2$
  Enable comparing between models
- Root Mean Square error
  $root(1/n\ (P'-P)^2)$
  Unit has a mean (Dollar square has no meaning)
- Use colab to open this github notebook:
  "s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb"

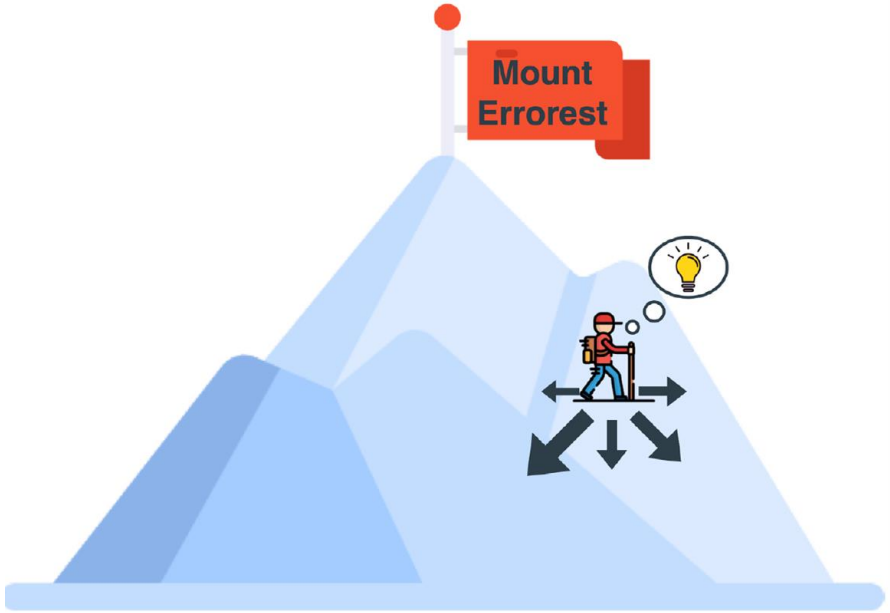
What is the best error (cost) function?

- Absolute Error
  |P'-P|
  Enough to measure the error
- Square Error
  $(P'-P)^2$
  Nicer derivative
- Mean Square Error
  $1/n\ (P'-P)^2$
  Enable comparing between models
- Root Mean Square error
  $root(1/n\ (P'-P)^2)$
  Unit has a mean (Dollar square has no meaning)
- Use colab to open this github notebook:
  "s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb"

Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

Gradient Descent

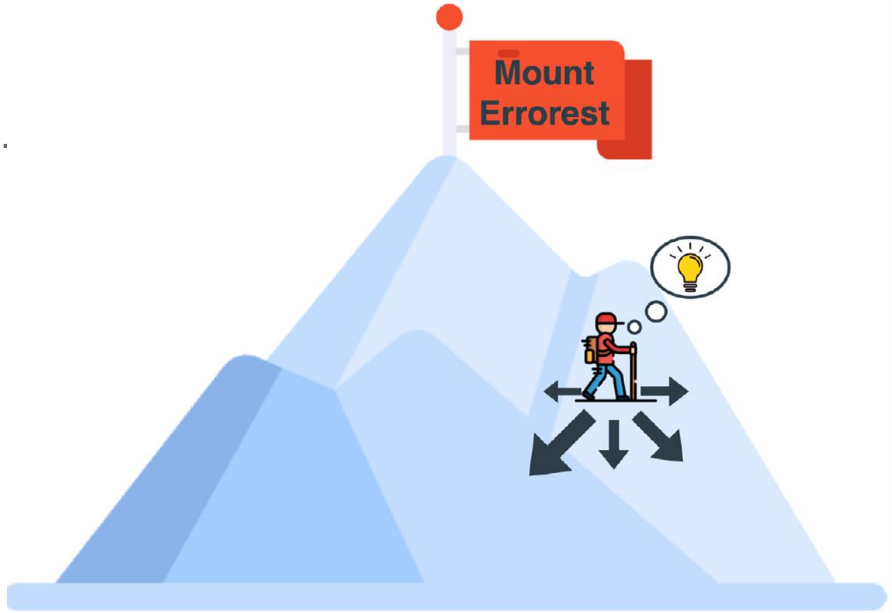


Gradient Descent

- Should we think how to update the weights and bias for every problem like we thought how to get?

  NO

- The story starts with the **error (cost) function**.
- The machine's goal is to **decrease the error**.
- Here comes the **gradient descent** magic.

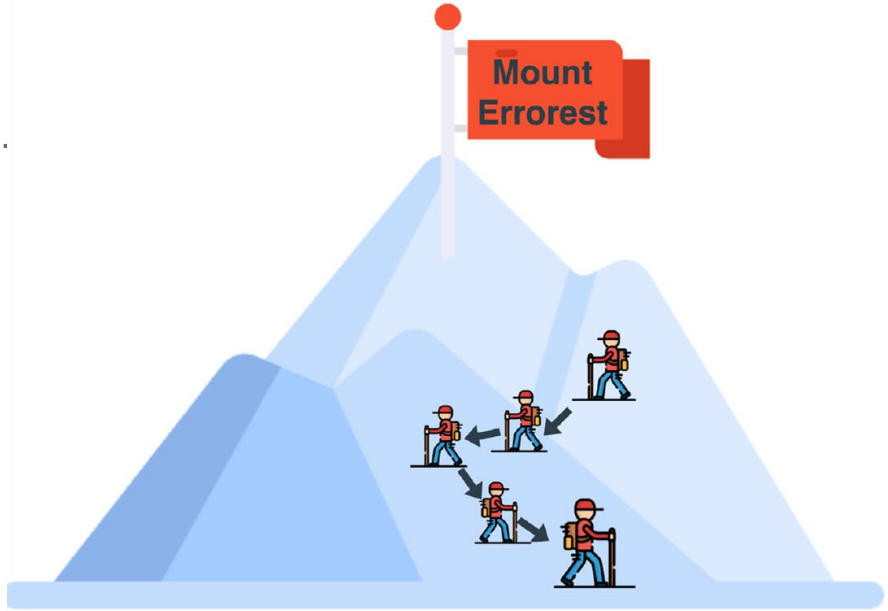


Gradient Descent

- Should we think how to update the weights and bias for every problem like we thought how to get?

  NO

- The story starts with the **error (cost) function**.
- The machine's goal is to **decrease the error**.
- Here comes the **gradient descent** magic.



Gradient Descent[Mathematics]

$$\hat{p} = mr + b$$

Gradient Descent[Mathematics]

$$\hat{p} = mr + b$$

MSE = 1/2n (P'-P)^2

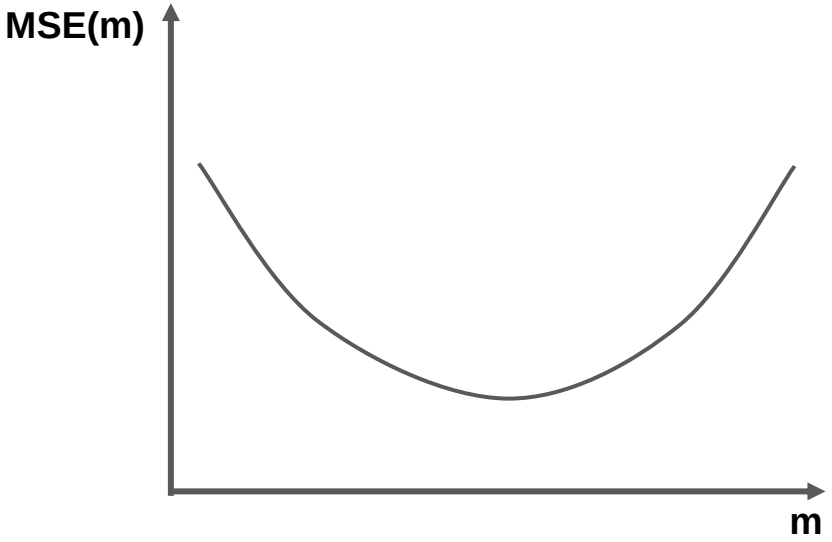
MSE = 1/2n ((mr+b)-P)^2

Gradient Descent[Mathematics]

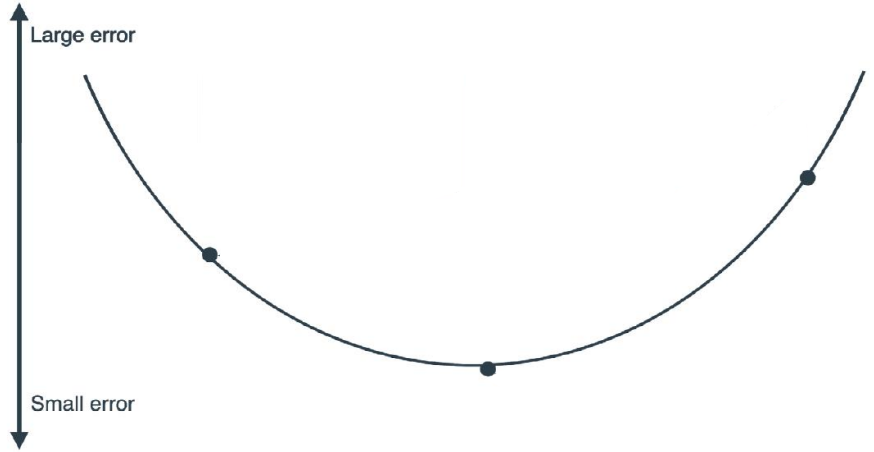
$$\hat{p} = mr + b$$

MSE = 1/2n (P'-P)^2

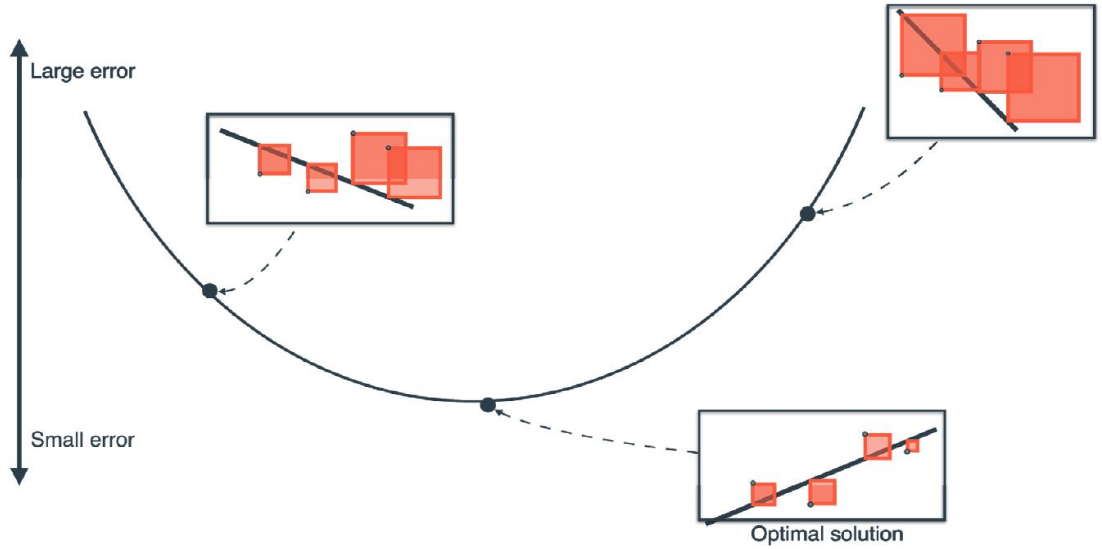
MSE = 1/2n ((mr+b)-P)^2



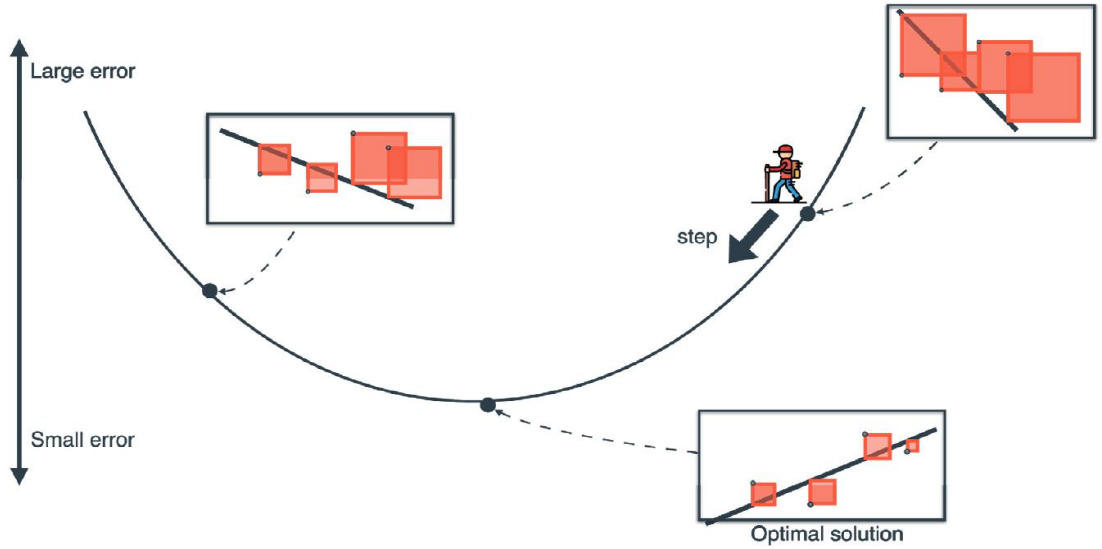
Gradient Descent[Mathematics]



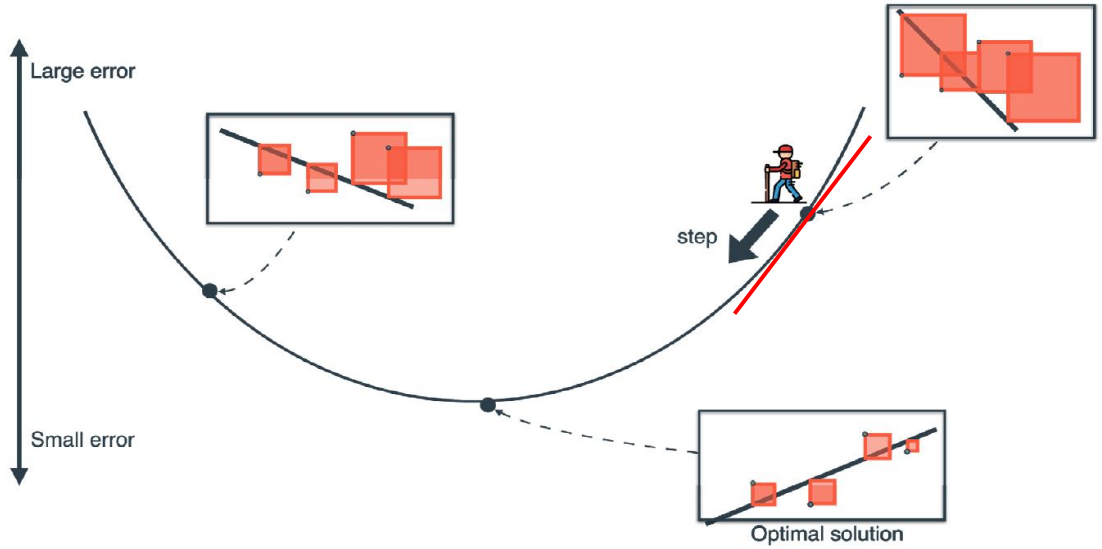
Gradient Descent[Mathematics]



Gradient Descent[Mathematics]



Gradient Descent[Mathematics]



Gradient Descent[Mathematics]

$m' = m - \eta$ (slope)

$m' = m - \eta \, \partial MSE/\partial m$

$MSE = 1/2n \, (P' - P)^2$

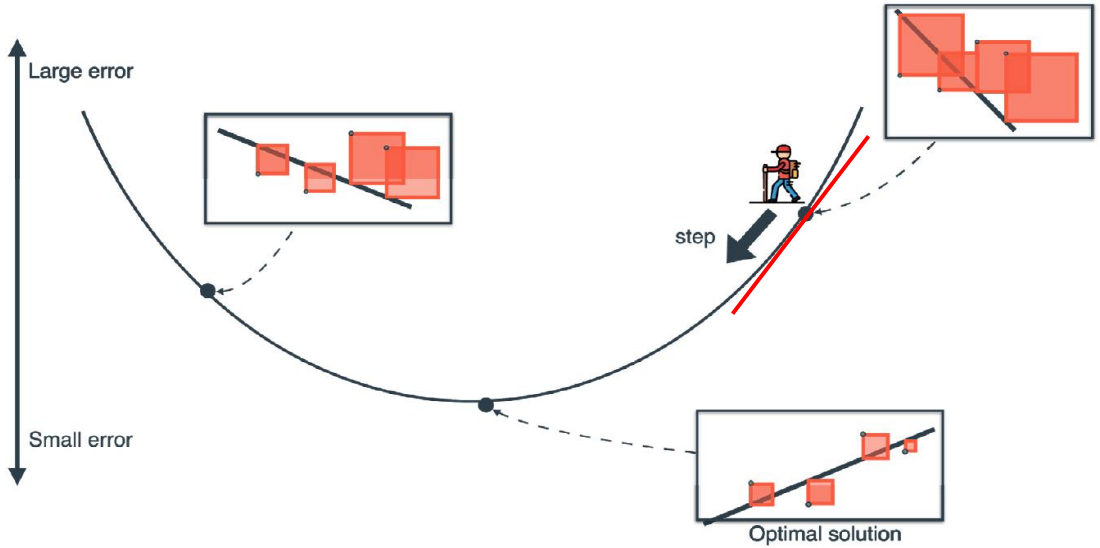
$MSE = 1/2n \, ((mr + b) - P)^2$

$\partial MSE/\partial m = 1/n * ((mr + b) - P)r$

$= 1/n * (P' - P)r$

$m' = m - \eta * 1/n * (P' - P)r$

$b' = b - \eta * 1/n * (P' - P)$



Gradient Descent[Mathematics]

m'=m - η (slope)

m'=m - η ∂MSE/∂m

MSE = 1/2n (P'-P)²

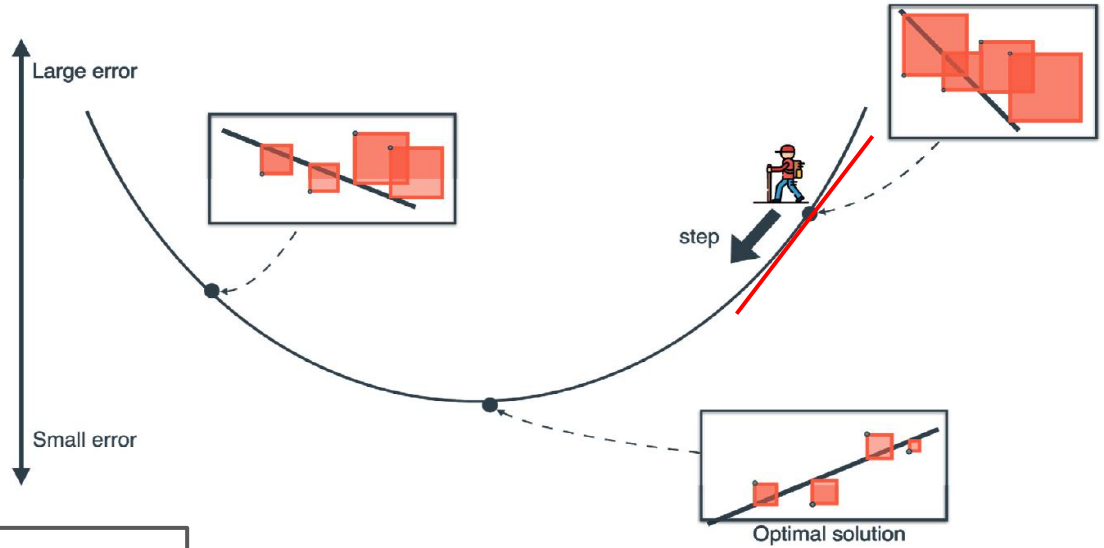
MSE = 1/2n ((mr+b)-P)²

∂MSE/∂m = 1/n * ((mr+b)-P)r

=1/n * (P'-P)r

m'=m - η * 1/n * (P'-P)r

b'=b - η * 1/n * (P'-P)



Square Trick

$$b' = b + \eta(p - \bar{p})$$

$$m' = m + \eta r(p - \bar{p})$$

Gradient Descent[Mathematics]

m'=m - η (slope)

m'=m - η ∂MSE/∂m

MSE = 1/2n (P'-P)^2

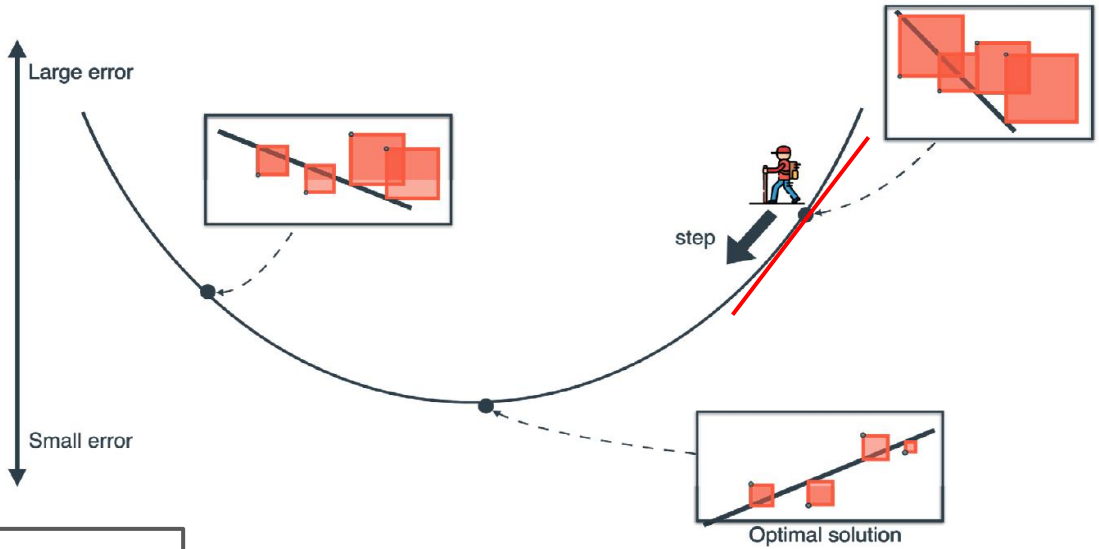
MSE = 1/2n ((mr+b)-P)^2

∂MSE/∂m = 1/n * ((mr+b)-P)r

=1/n * (P'-P)r

m'=m - η * 1/n * (P'-P)r

b'=b - η * 1/n * (P'-P)



Square Trick

$$b' = b + \eta(p - \bar{p})$$

$$m' = m + \eta r(p - \bar{p})$$

absolute error minimization vs absolute trick..

Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

Batch vs Stochastic vs Mini-Batch Gradient Descent

Batch vs Stochastic vs Mini-Batch Gradient Descent

m'=m - η * 1/n * (P'-P)r

When do we update the weights?

Batch Gradient Descent: Update weights after calculating the error over all the dataset.

Stochastic Gradient Descent: Update weights after calculating the error over only one point.

Mini-Batch Gradient Descent: Update weights after calculating the error over mini batch of points.

What is the difference between the 3 options?

Batch vs Stochastic vs Mini-Batch Gradient Descent

m'=m - η * 1/n * (P'-P)r
When do we update the weights?

Batch Gradient Descent: Update weights after calculating the error over all the dataset.
[Takes a lot of time]

Stochastic Gradient Descent: Update weights after calculating the error over only one point.

Mini-Batch Gradient Descent: Update weights after calculating the error over mini batch of points.

What is the difference between the 3 options?

Batch vs Stochastic vs Mini-Batch Gradient Descent

m'=m - η * 1/n * (P'-P)r

When do we update the weights?

Batch Gradient Descent: Update weights after calculating the error over all the dataset.
[Takes a lot of time]

Stochastic Gradient Descent: Update weights after calculating the error over only one point.
[Not stable]

Mini-Batch Gradient Descent: Update weights after calculating the error over mini batch of points.

What is the difference between the 3 options?

Batch vs Stochastic vs Mini-Batch Gradient Descent

m'=m - η * 1/n * (P'-P)r

When do we update the weights?

Batch Gradient Descent: Update weights after calculating the error over all the dataset.

[Takes a lot of time]

Stochastic Gradient Descent: Update weights after calculating the error over only one point.

[Not stable]

Mini-Batch Gradient Descent: Update weights after calculating the error over mini batch of points.

[balance between both]

What is the difference between the 3 options?

Learning rate(η or α)

Learning rate(η or α)

m’=m - η *** 1/n * (P’-P)r**
Machine learning models includes:
- Parameters: which is been trained for. (Weights and biases)
- Hyperparameters: which is parameters with a known value before training like learning rate- number of epochs.

Learning rate cases:

Low: small steps and
long training time

High: Not converging

Optimal: conversion
in short time

Learning rate(η or α)

m’=m - η *** 1/n * (P’-P)r**

Machine learning models includes:

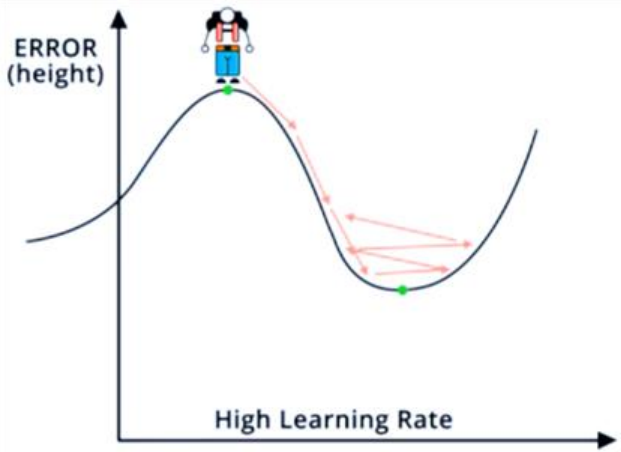
- Parameters: which is been trained for. (Weights and biases)
- Hyperparameters: which is parameters with a known value before training like learning rate- number of epochs.

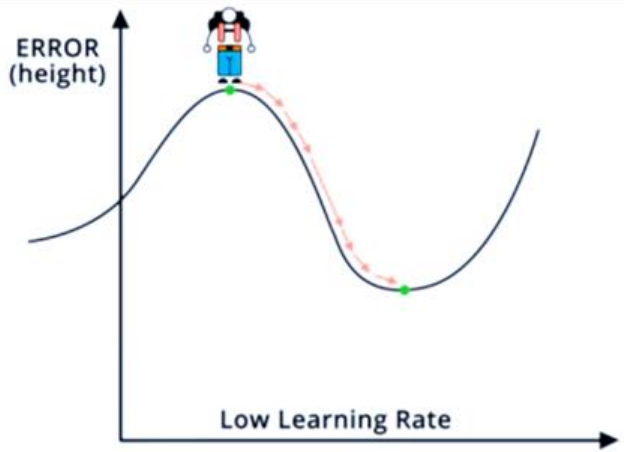
Learning rate cases:

Low: small steps and long training time

High: Not converging

Optimal: conversion in short time





Feature Scaling (Normalization)

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

Feature Scaling (Normalization)

m'=m - η *** 1/n * (P'-P)r**

- Notice that updating weights depends on **r (feature value)**
- This means in our example that all weights will update **very slow** comparing to the corresponding weight of (size)
- That will lead to **slow converging**
- To solve that we **normalize all the features to same scale**, for example -1 < X < 1
- **X = (X - mean) / range**

Size (feet^2)	Number of bedrooms	Number of floors	Age of home (years)	Price ($1000)
2104	5	1	45	460
1416	3	2	40	232
1534	3	2	30	315
852	2	1	36	178

Lecture Overview

- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

SKLearn

Use colab to open this github notebook:

“s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb”

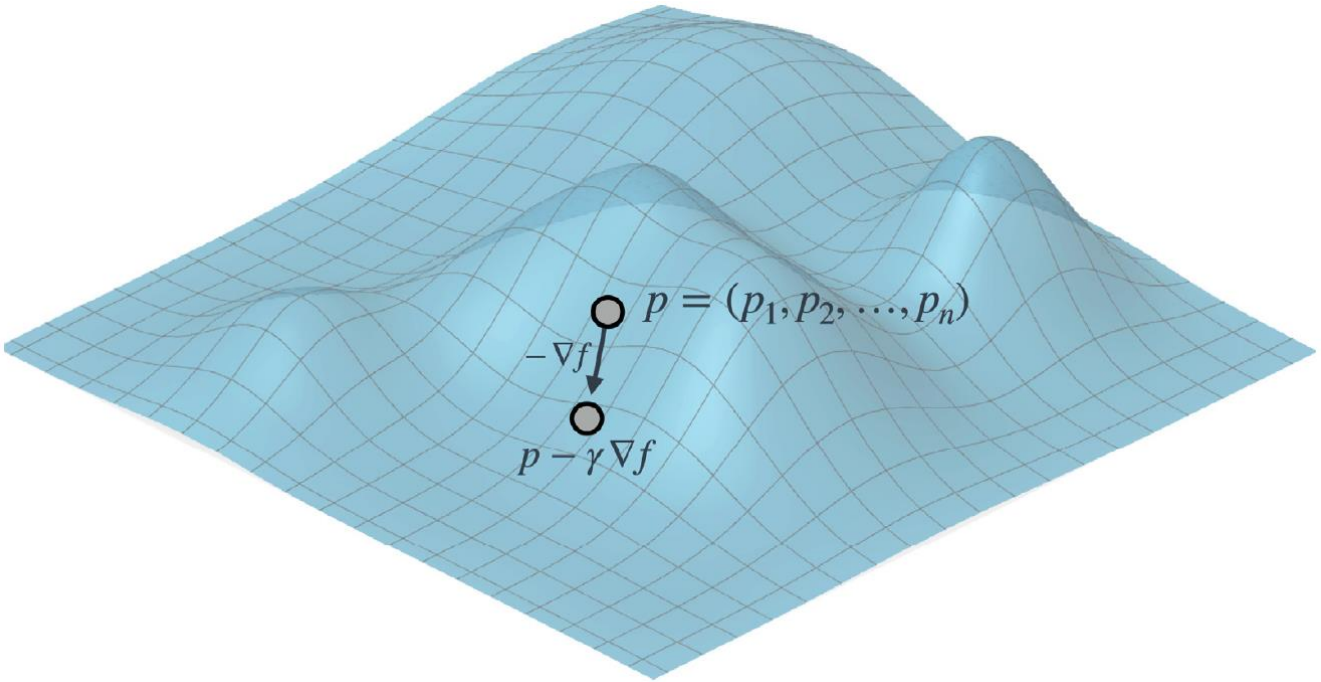
SKLearn

```
>>> import numpy as np
>>> from sklearn.linear_model import LinearRegression
>>> X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
>>> # y = 1 * x_0 + 2 * x_1 + 3
>>> y = np.dot(X, np.array([1, 2])) + 3
>>> reg = LinearRegression().fit(X, y)
>>> reg.score(X, y)
1.0
>>> reg.coef_
array([1., 2.])
>>> reg.intercept_
3.0...
>>> reg.predict(np.array([[3, 5]]))
array([16.])
```

Lecture Overview

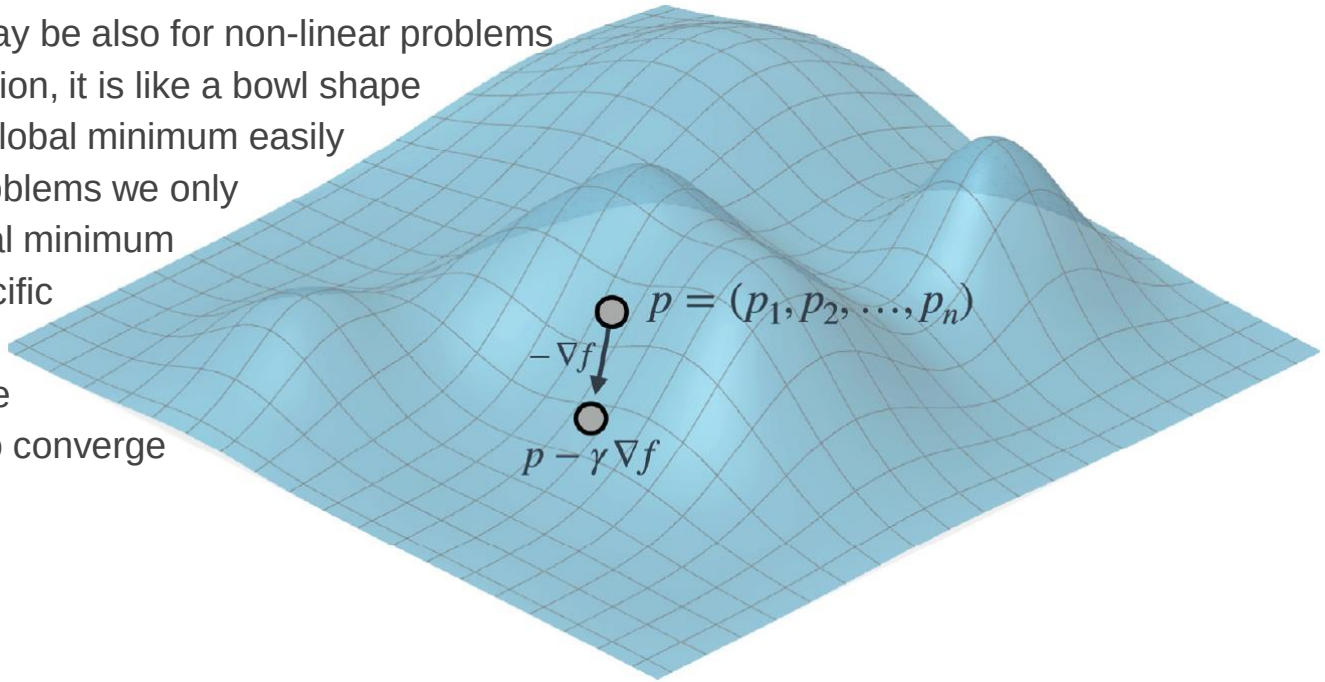
- Problem Definition
- Bias and Weights
- How machines learn it?
- Multivariate Linear Regression
- Solve the problem graphically
- Solve the problem Mathematically
- Cost Function (Error)
- Gradient Descent
- Some training tricks
- SKlearn
- Normal Equation

General Error (Cost) Function



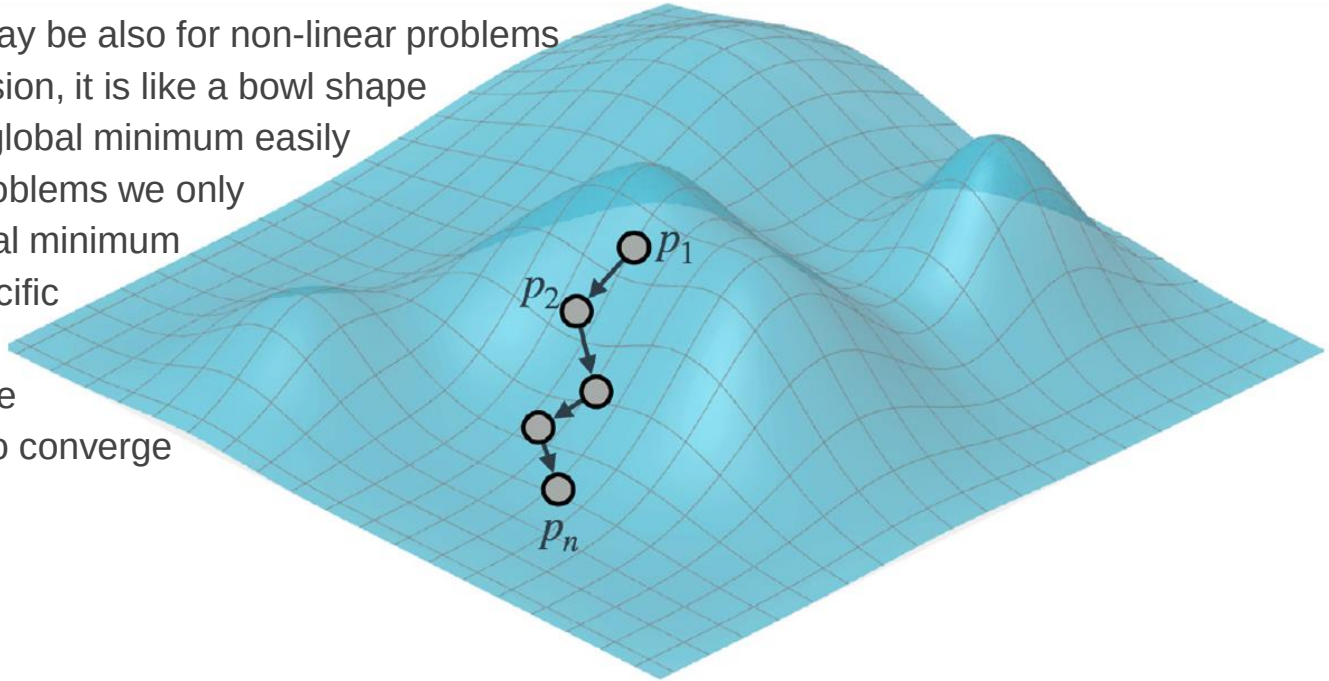
General Error (Cost) Function

- Cost function may be also for non-linear problems
- In linear regression, it is like a bowl shape and can reach global minimum easily
- In non-linear problems we only can reach a local minimum
- There is no specific one solution
- In both cases we use derivative to converge



General Error (Cost) Function

- Cost function may be also for non-linear problems
- In linear regression, it is like a bowl shape and can reach global minimum easily
- In non-linear problems we only can reach a local minimum
- There is no specific one solution
- In both cases we use derivative to converge



Normal Equation

feature

data sample

	Size (feet²) X_1	Number of bedrooms X_2	Number of floors X_3	Age of home (years) X_4	Price ($1000) Y
$X^{(1)}$	2104	5	1	45	460
$X^{(2)}$	1416	3	2	40	232
$X^{(3)}$	1534	3	2	30	315
$X^{(4)}$	852	2	1	36	178
Weights	W_1	W_2	W_3	W_4	

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} b \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} b \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples*(#features+1)]

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} b \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples*(#features+1)] [(#features+1)*1]

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} b \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples*(#features+1)] [(#features+1)*1] [#samples*1]

Normal Equation

$$X \begin{bmatrix} 2104 & 5 & 1 & 45 \\ 1416 & 3 & 2 & 40 \\ 1534 & 3 & 2 & 30 \\ 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} + b = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples* #features] [#features*1] [#samples*1]

$$X \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix} * W \begin{bmatrix} b \\ W_1 \\ W_2 \\ W_3 \\ W_4 \end{bmatrix} = Y' \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \\ y'_4 \end{bmatrix}$$

[#samples*(#features+1)] [(#features+1)*1] [#samples*1]

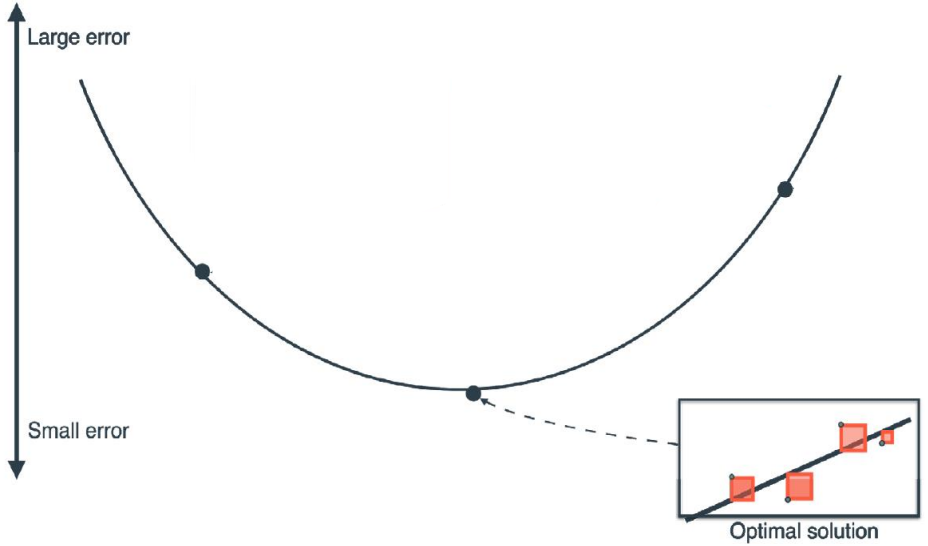
$$X * W = Y'$$

Normal Equation

X * W = Y’

Normal Equation

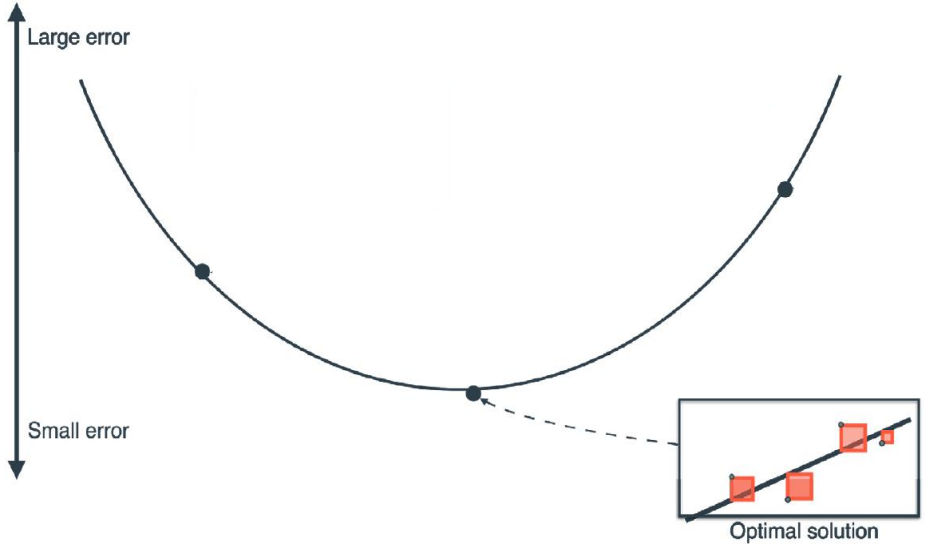
X * W = Y’



Normal Equation

X * W = Y’

Goal is to get the optimal W

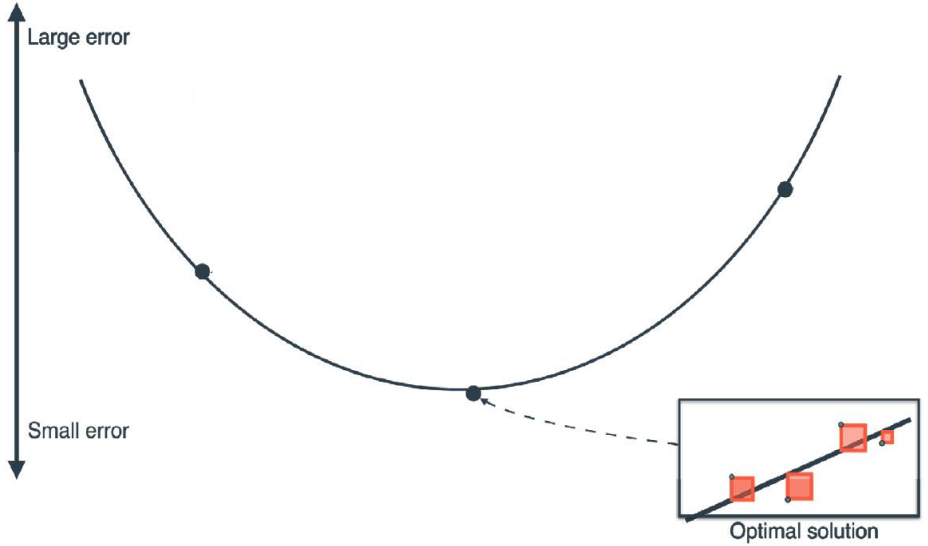


Normal Equation

X * W = Y'

Goal is to get the optimal W

Optimal case: Y'=Y



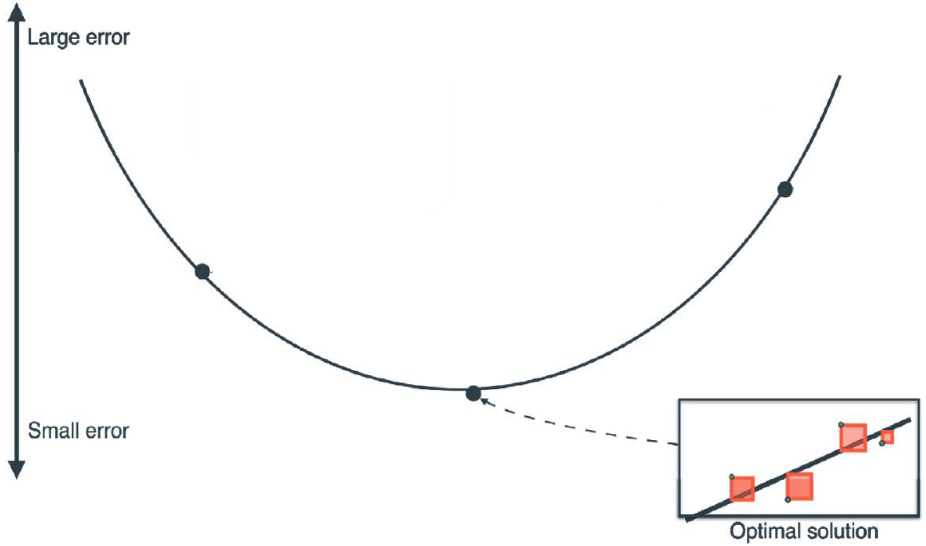
Normal Equation

X * W = Y'

Goal is to get the optimal W

Optimal case: Y'=Y

X * W = Y



Normal Equation

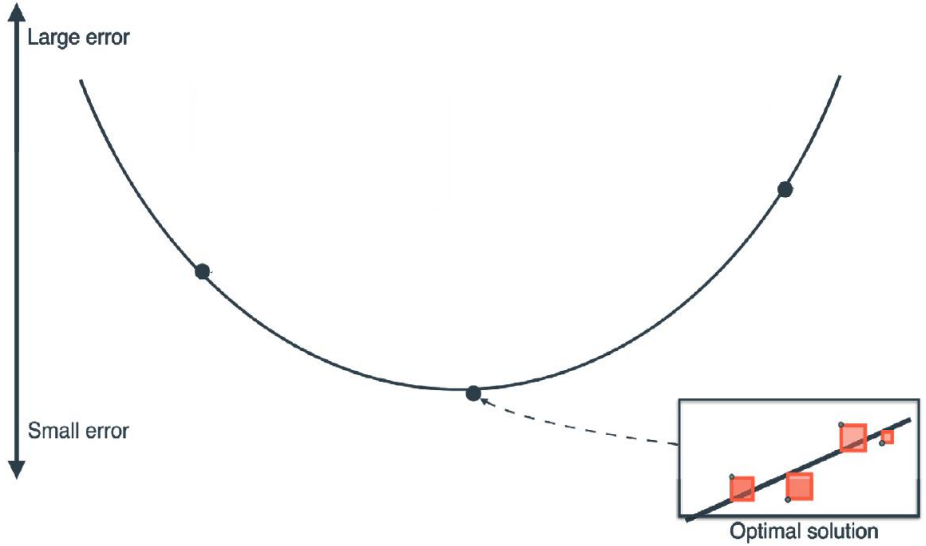
$X * W = Y'$

Goal is to get the optimal W

Optimal case: Y'=Y

$X * W = Y$

$W = X^{-1} * Y$



Normal Equation

X * W = Y'

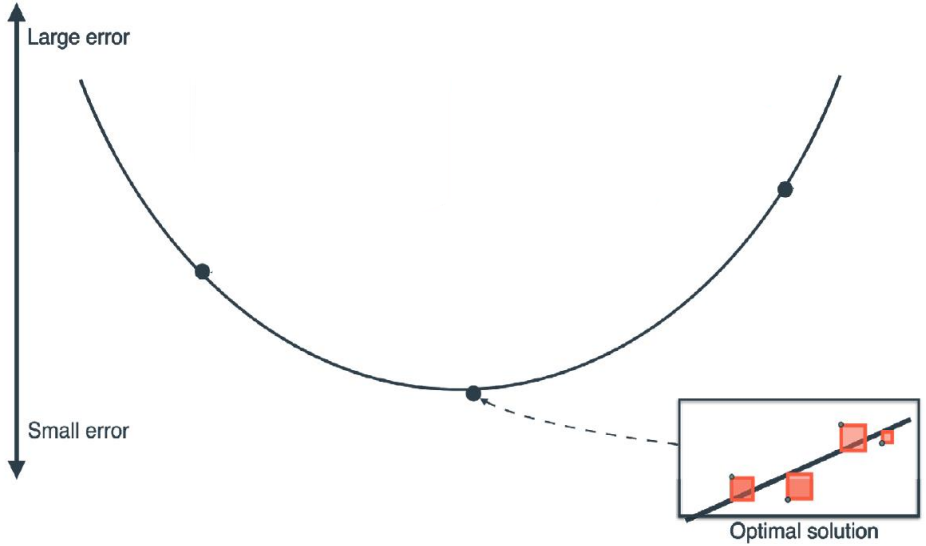
Goal is to get the optimal W

Optimal case: Y'=Y

X * W = Y

~~**W = X-1 * Y**~~

If X is not square, X has no inverse



Normal Equation

$X * W = Y'$

Goal is to get the optimal W

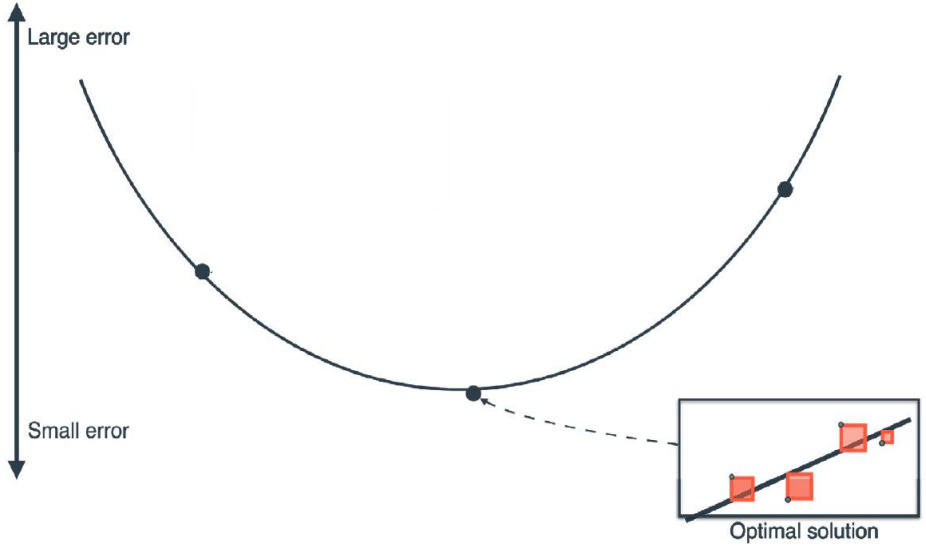
Optimal case: Y'=Y

$X * W = Y$

~~$W = X^{-1} * Y$~~

If X is not square, X has no inverse

$X^T * X * W = X^T * Y$



Normal Equation

$X * W = Y'$

Goal is to get the optimal W

Optimal case: Y'=Y

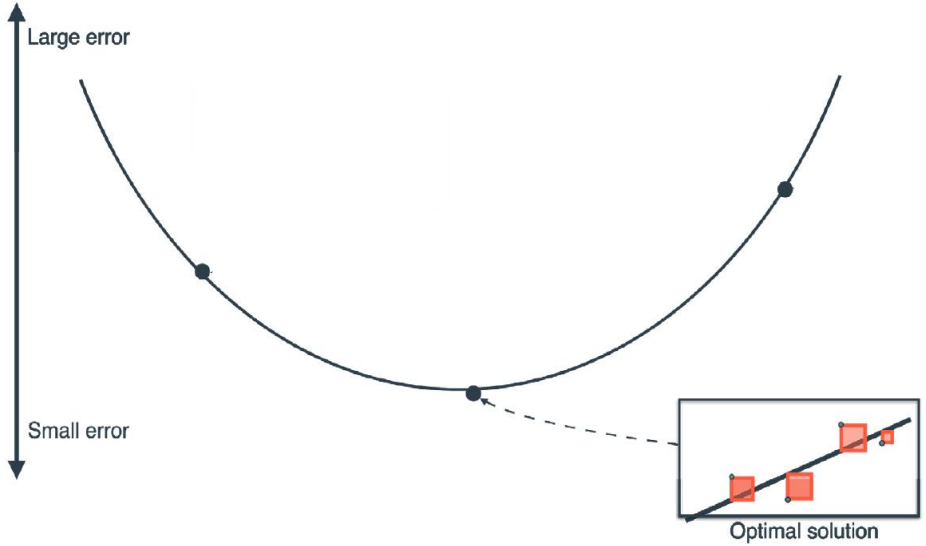
$X * W = Y$

~~$W = X^{-1} * Y$~~

If X is not square, X has no inverse

$X^T * X * W = X^T * Y$

$W = (X^T * X)^{-1} * X^T * Y$



Normal Equation

$$X * W = Y'$$

Goal is to get the optimal W

Optimal case: Y'=Y

$$X * W = Y$$

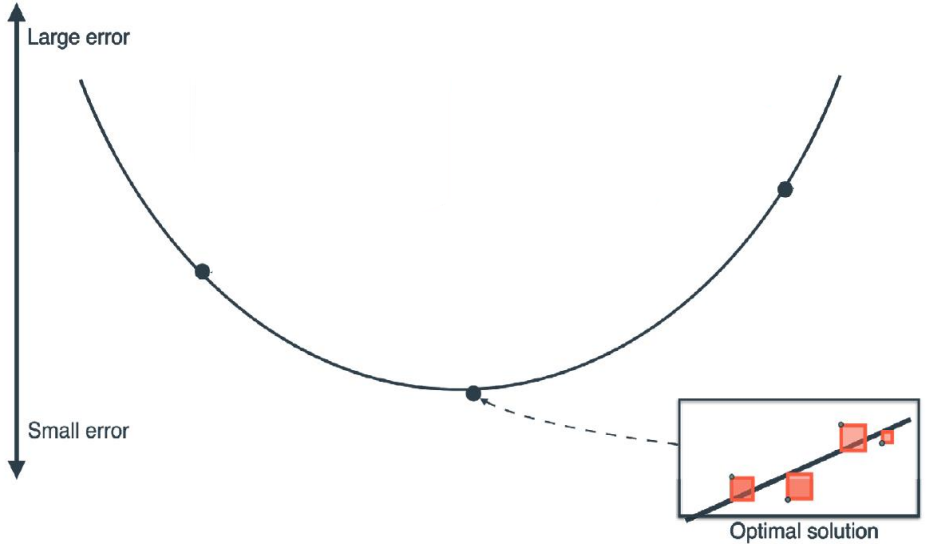
$$W = X^{-1} * Y$$

If X is not square, X has no inverse

$$X^T * X * W = X^T * Y$$

$$W = (X^T * X)^{-1} * X^T * Y$$

Pseudo-inverse



Normal Equation

$X * W = Y'$

Goal is to get the optimal W

Optimal case: Y'=Y

$X * W = Y$

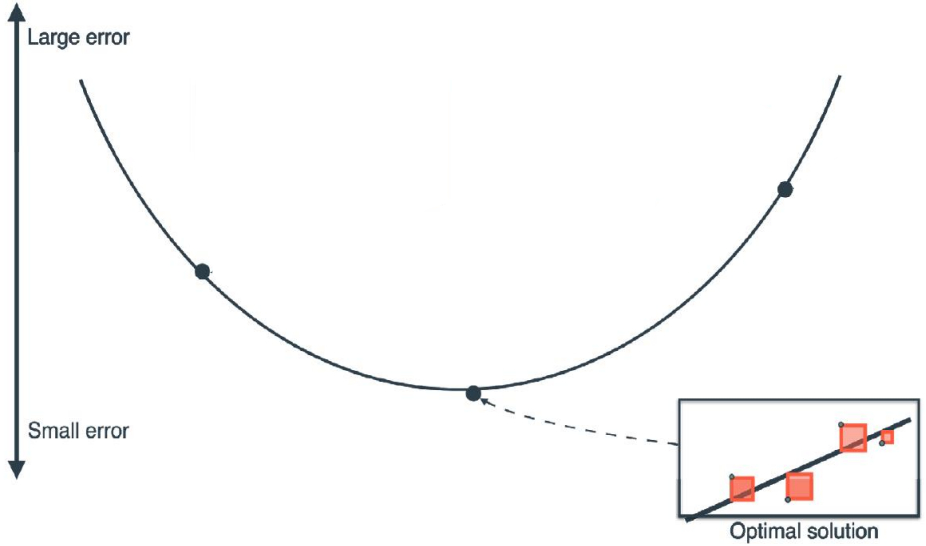
$W = X^{-1} * Y$

If X is not square, X has no inverse

$X^T * X * W = X^T * Y$

$W = (X^T * X)^{-1} * X^T * Y$

Pseudo-inverse



Why not always use normal equation instead of gradient descent?

Normal Equation

Use colab to open this github notebook:

“s7s/machine_learning_1/linear_regression/Coding_linear_regression.ipynb”

Important Concepts

- Weights vs Bias and mapping them to slope and y-intercept
- How Multivariate Linear Regression looks like?
- Model evaluation
- Gradient descent
- Gradient descent types
- Parameters vs hyperparameters
- Learning rate
- Features scaling
- Normal equation

Feedback