

CSCI 111 Web Programming and Problem Solving  
Section 1

PART III Artificial Intelligence

Weeks [12 - 15]

Week-13-lecture: Machine Learning. Regression.

Instructor: Dr. Talgat Manglayev

# CONTENT

- The problem: Predict the price of a house
- The solution: Building a regression model for housing prices
- The linear regression algorithm
- How do we measure our results? The error function
- Polynomial regression
- Parameters and hyper parameters
- Underfitting and overfitting data
- Regularization

# The problem: Predict the price of a house

What would be the price for a house with 4 rooms?

Rooms	Price
1	10
2	15
3	20
4	?
5	30
6	35
7	40

# The problem: Predict the price of a house

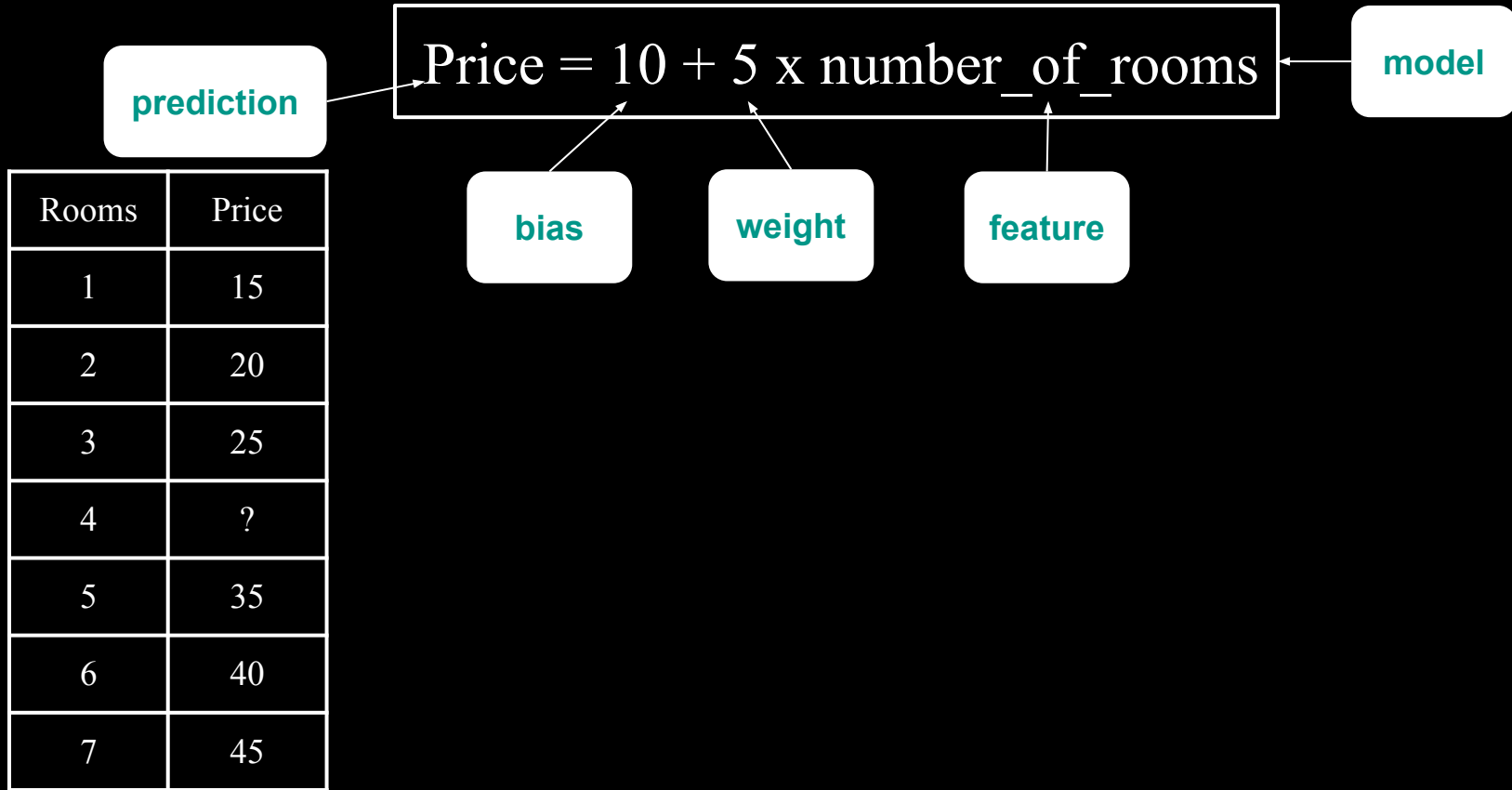
What would be the formula for price for a house?

$$\text{Price} = 5 + 5 \times \text{number\_of\_rooms}$$

Rooms	Price
1	10
2	15
3	20
4	?
5	30
6	35
7	40

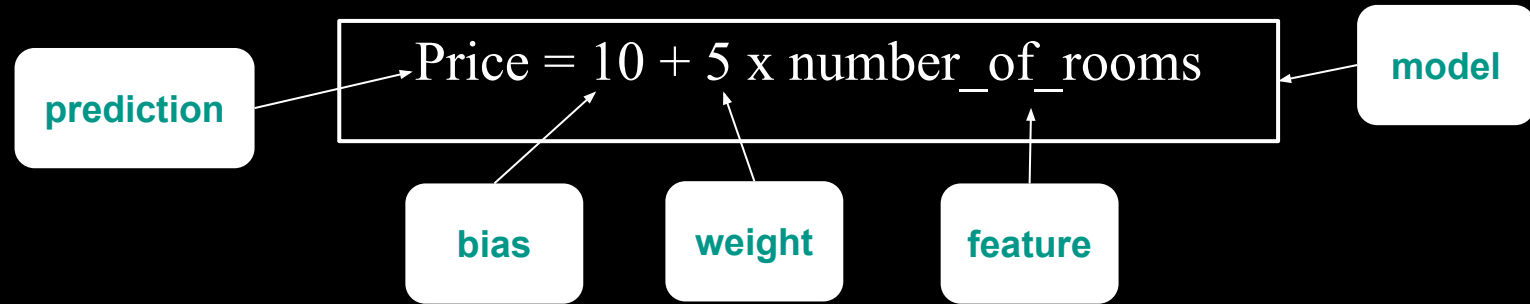
# The problem: Predict the price of a house

What would be the formula for price for a house?



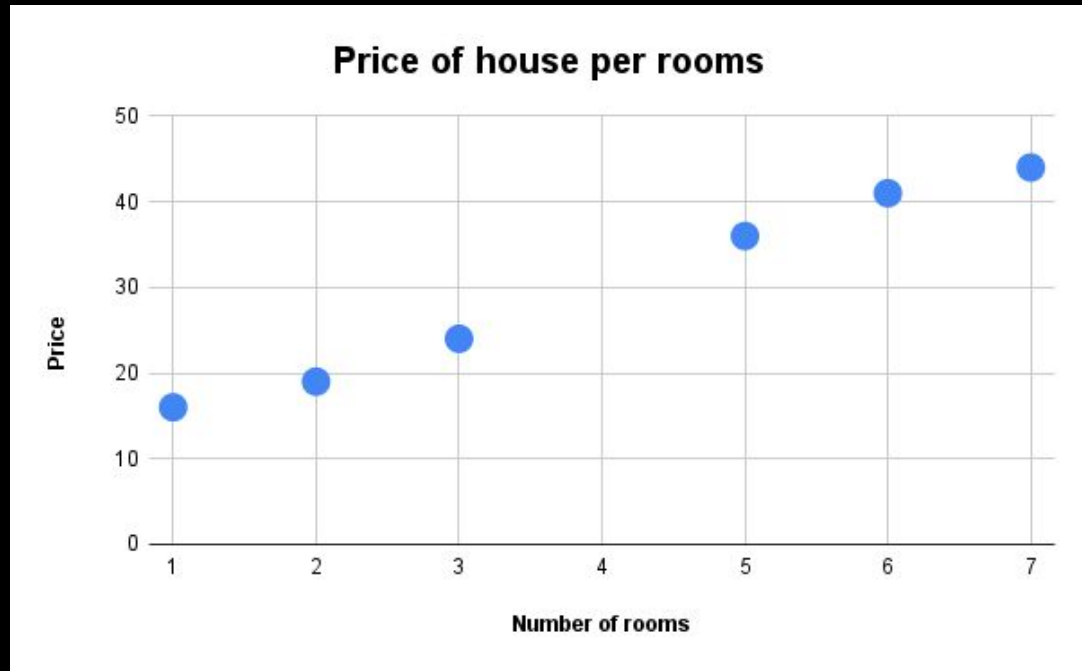
# The problem: Predict the price of a house

The model to predict price for a house



# The problem: Predict the price of a house

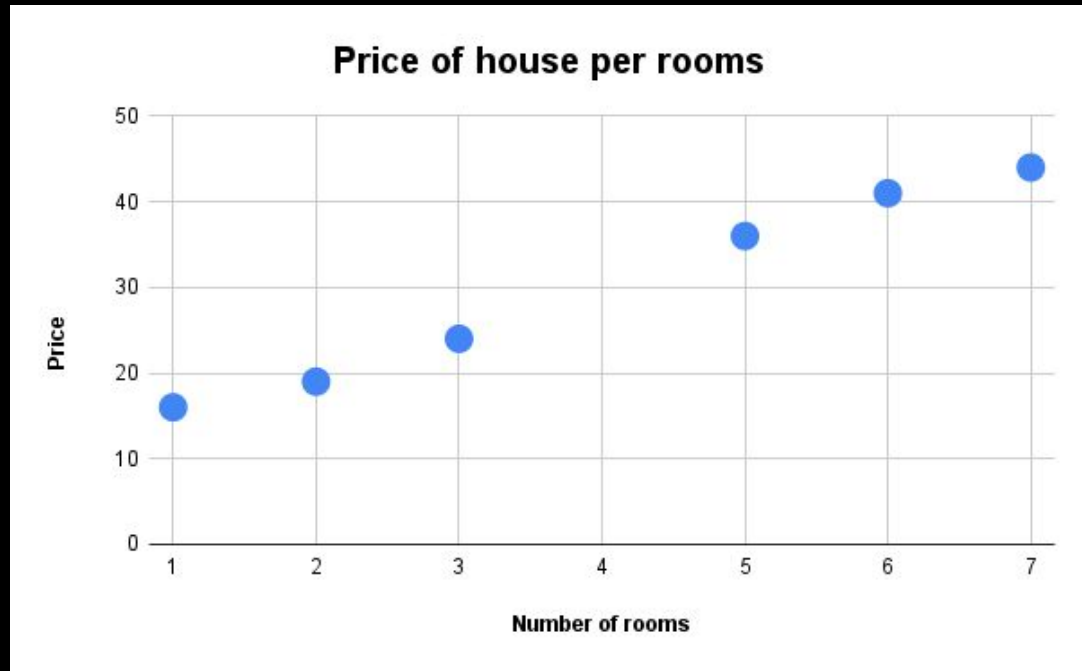
Rooms	Price
1	16
2	19
3	24
4	?
5	36
6	41
7	44



# The problem: Predict the price of a house

$$\text{Price} = 10 + 5 \times \text{number\_of\_rooms} + (\text{small\_error})$$

Rooms	Price
1	16
2	19
3	24
4	?
5	36
6	41
7	44





# The problem: Predict the price of a house

$$\text{Price} = 10 + 5 \times \text{number\_of\_rooms} + (\text{small\_error})$$

Rooms	Price
1	16
2	19
3	24
4	30
5	36
6	41
7	44

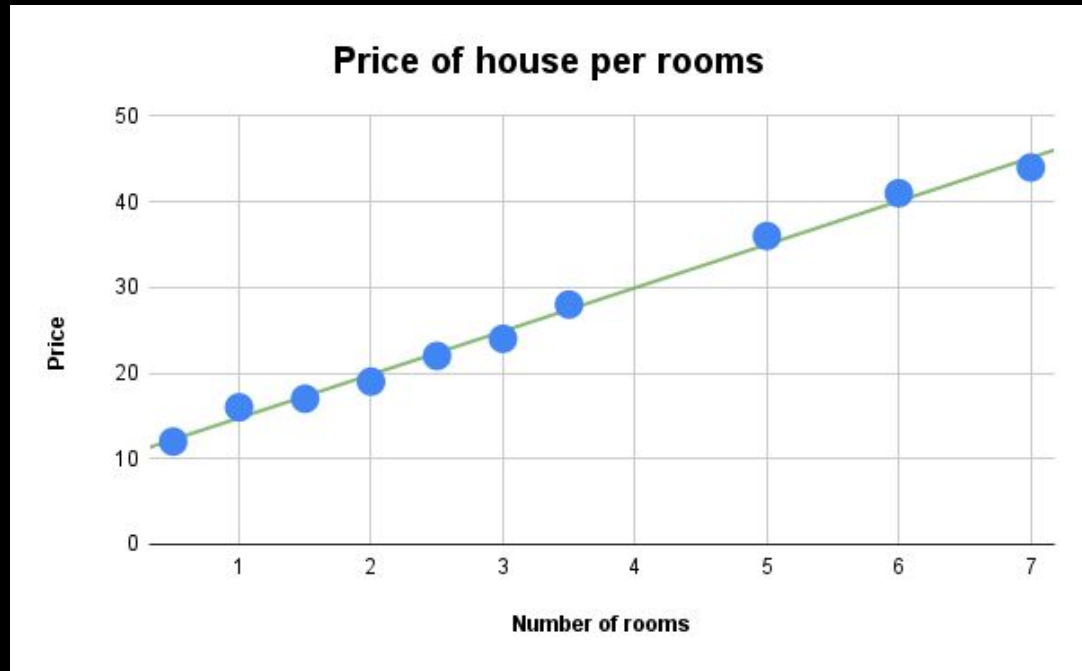


# The problem: Predict the price of a house

$$\text{Price} = 10 + 5 \times \text{number\_of\_rooms} + (\text{small\_error})$$

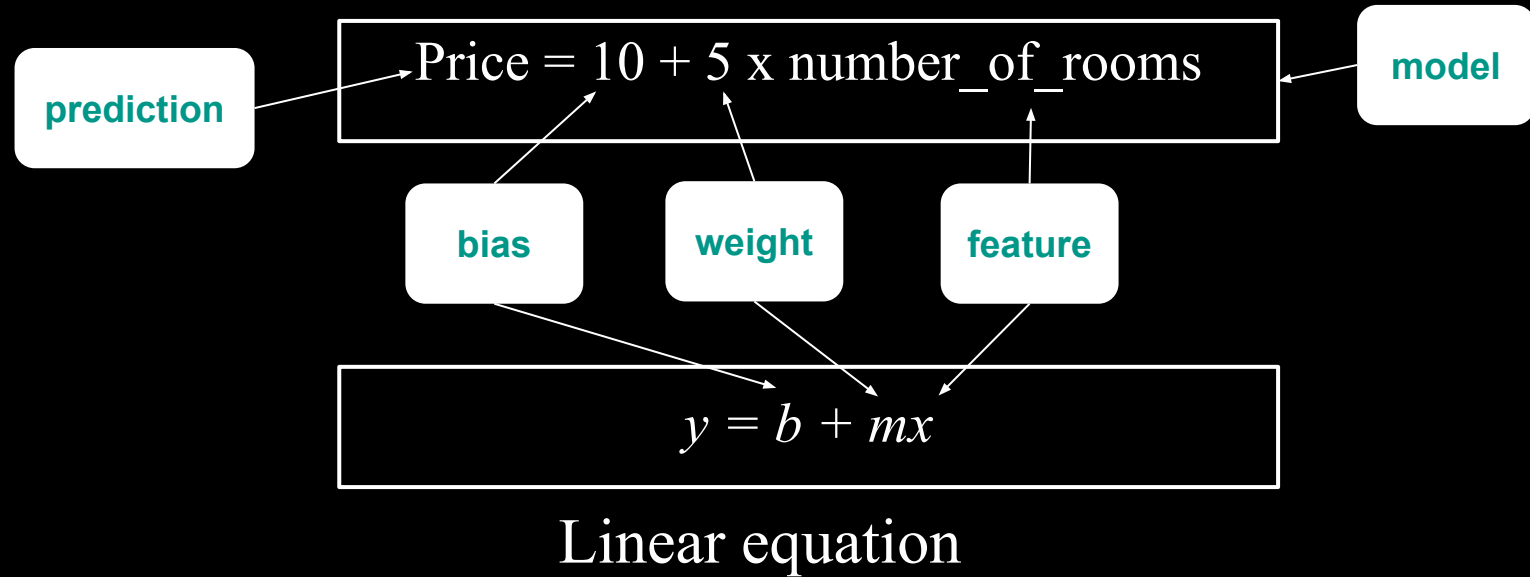
The goal of linear regression is to draw the straight line that passes as close as possible to these points.

Rooms	Price
1	16
2	19
3	24
4	30
5	36
6	41
7	44



# The problem: Predict the price of a house

The model to predict price for a house



# Multivariate linear regression

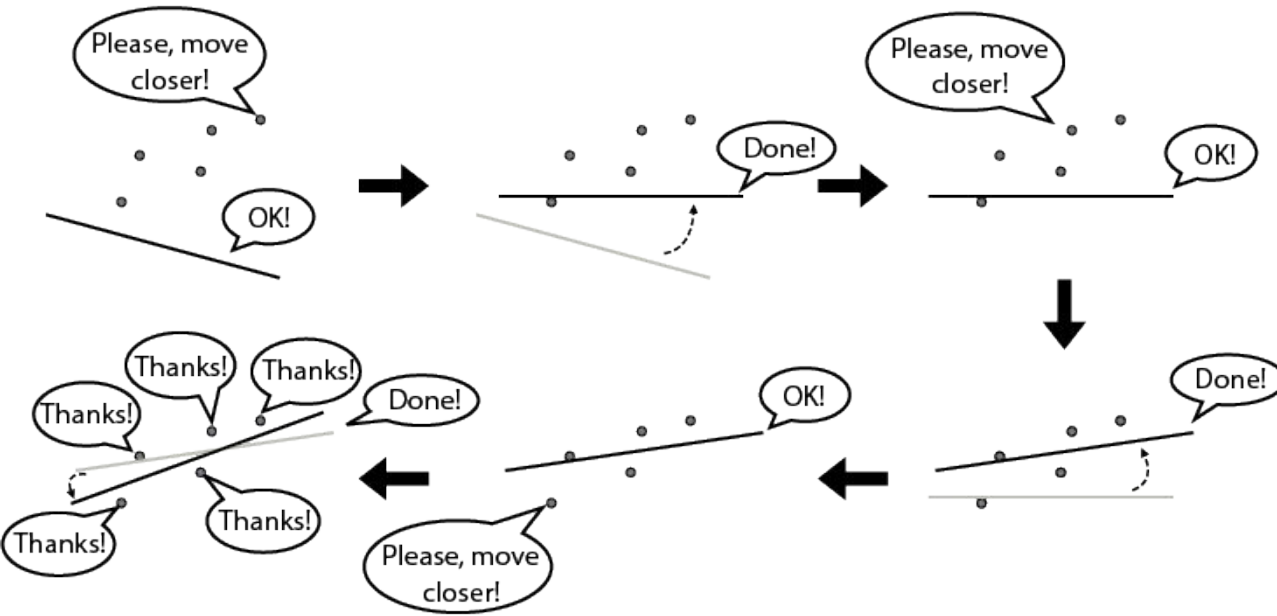
feature

$$\text{Price} = 5 \times \text{number\_of\_rooms} + 1.5 \times \text{size} - 2 \times \text{age} + 2 \times \text{number\_of\_schools} + 10$$

weight value -  
how important is  
the feature to  
determine the price

sign affects  
whether the price  
rises (+) or falls (-)

# The linear regression algorithm



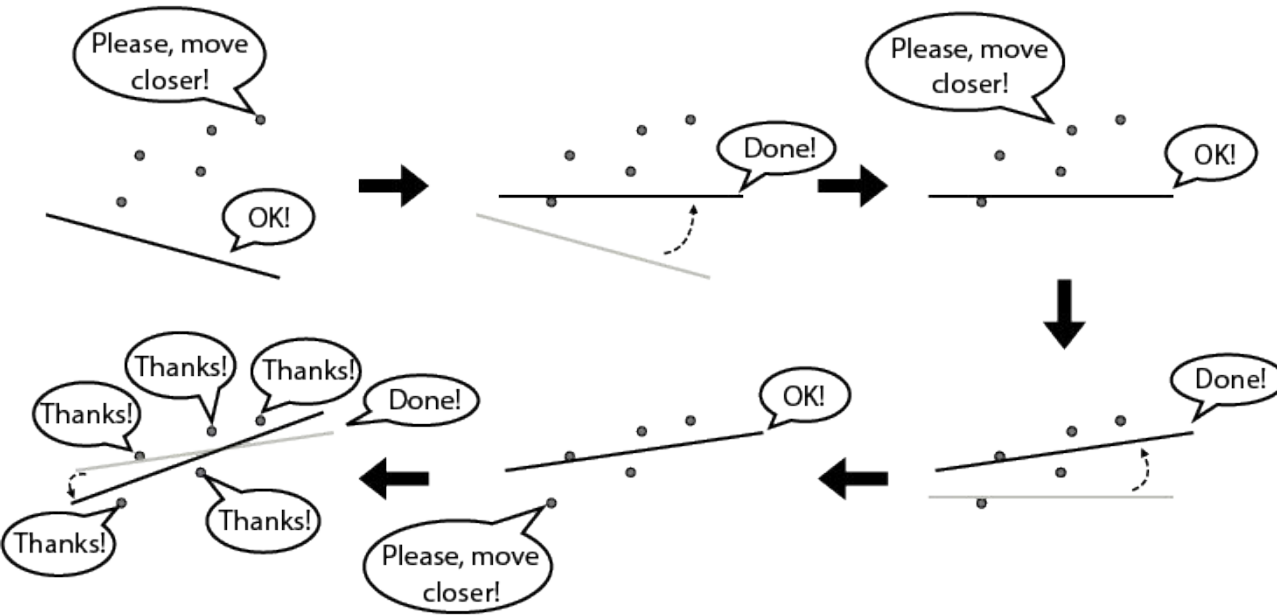
Pick a random line.

**Repeat many times:**

- Pick a random data point.
- Move the line a little closer to the point.

**Return** the line you've obtained.

# The linear regression algorithm



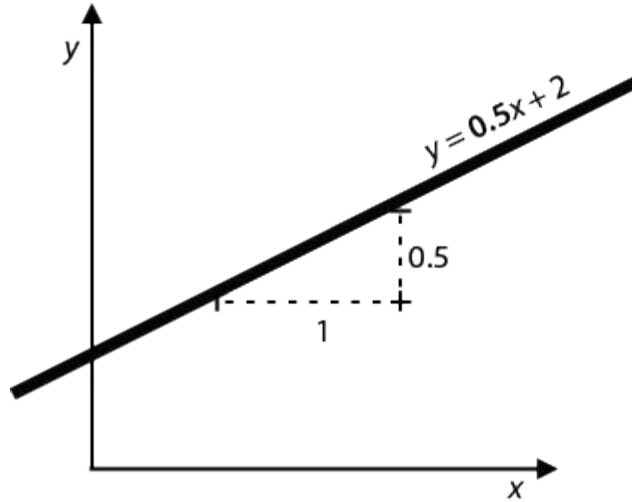
Pick a model with random weights and a random bias.

## Repeat many times:

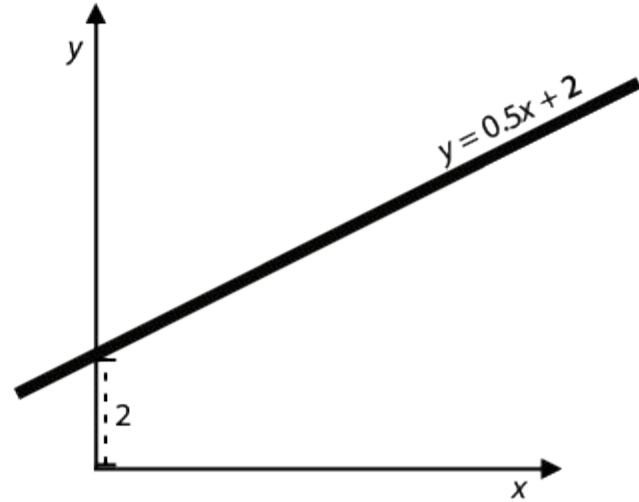
- Pick a random data point.
- Slightly adjust the weights and bias to improve the prediction for that particular data point.

**Return** the model you've obtained.

# The linear regression algorithm

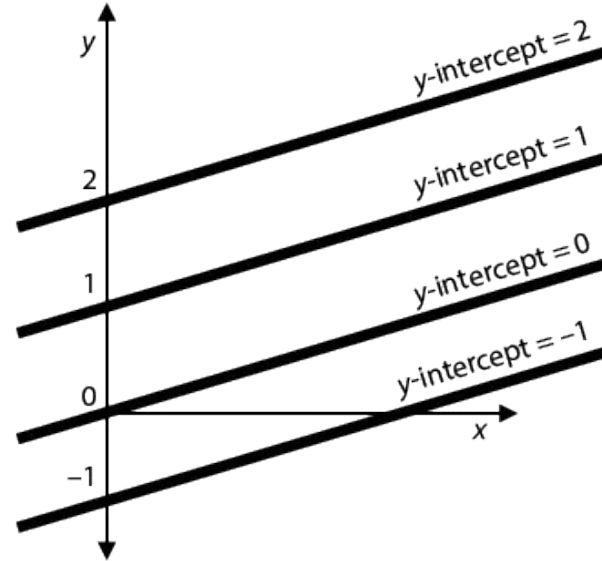
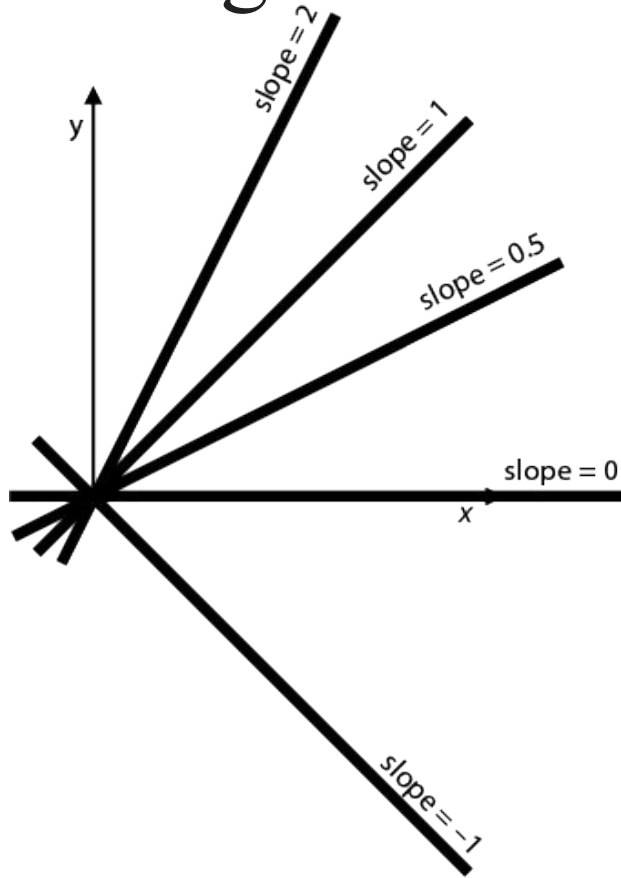


Slope = 0.5



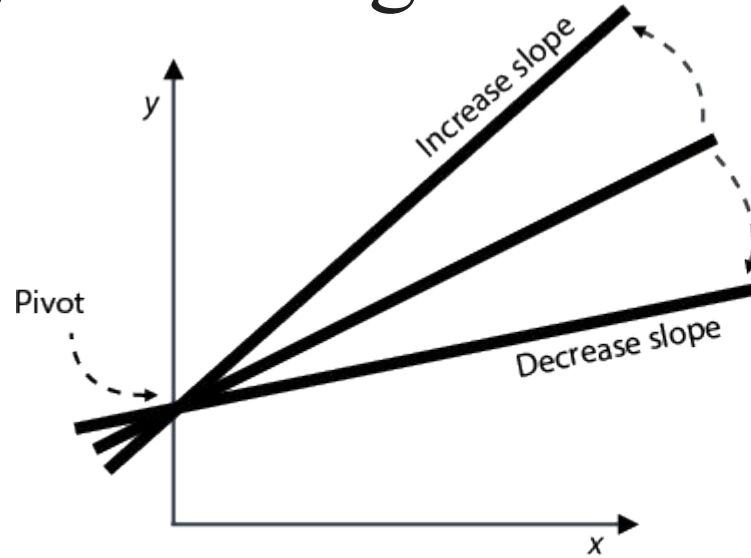
y-intercept = 2

# The linear regression algorithm





# The linear regression algorithm

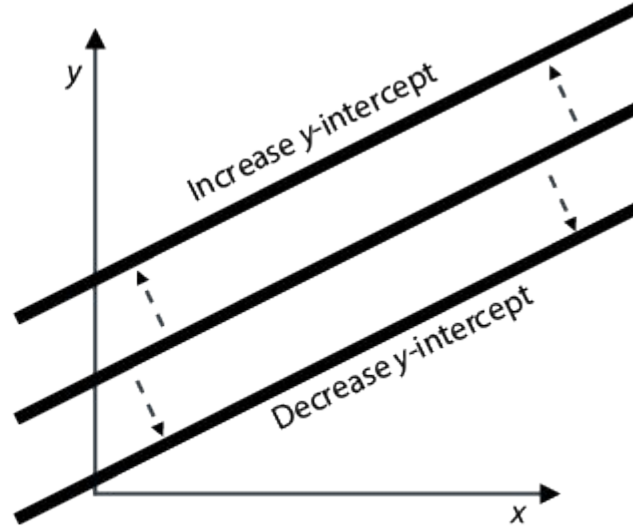


**Rotate clockwise and  
counterclockwise**

If we increase the slope of a line, the line will rotate counterclockwise.

If we decrease the slope of a line, the line will rotate clockwise.

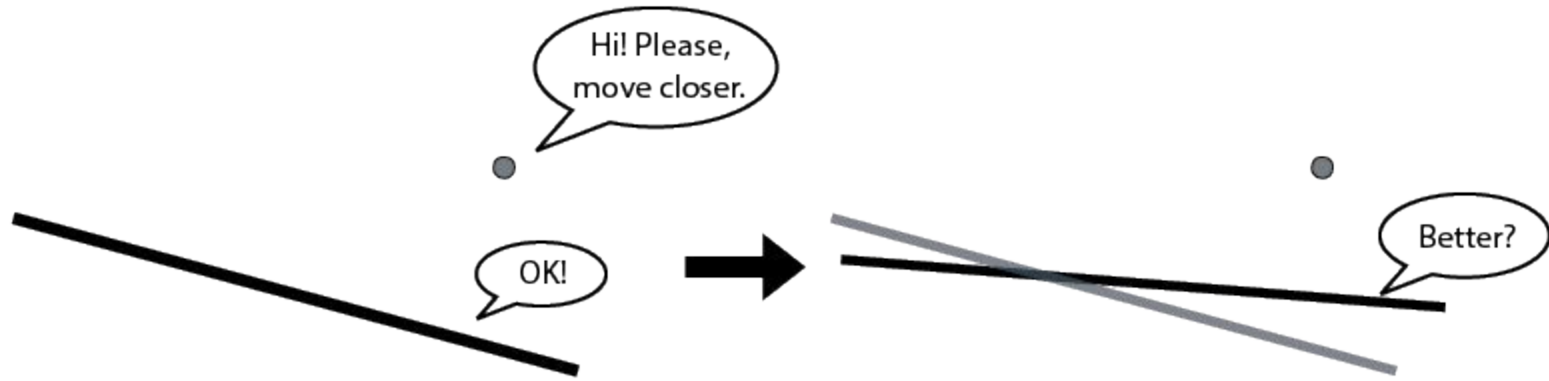
# The linear regression algorithm



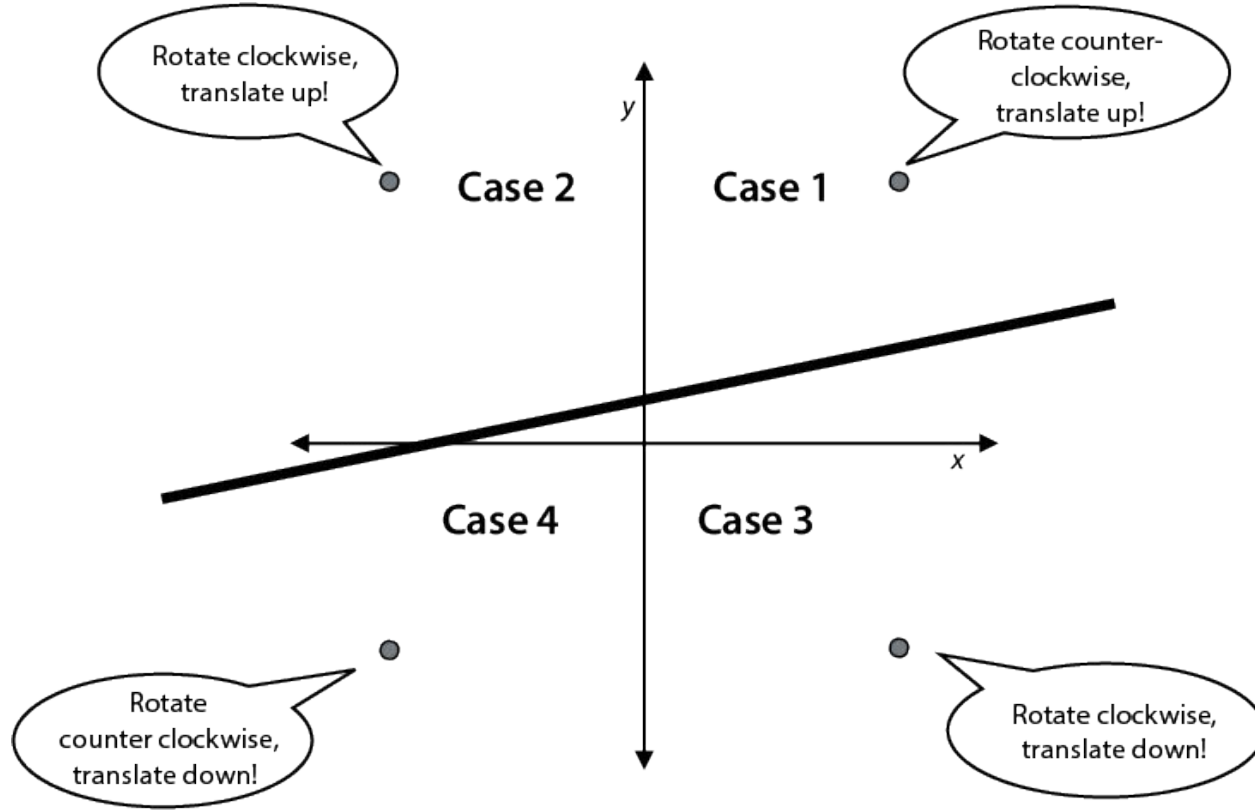
**Translate up and down**

If we increase the y-intercept of a line, the line is translated upward.  
If we decrease the y-intercept of a line, the line is translated downward.

# The linear regression algorithm



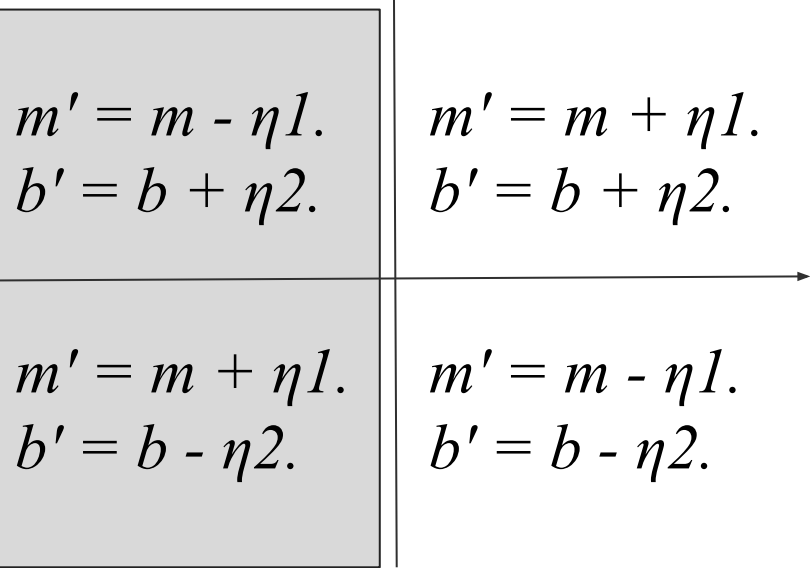
# The linear regression algorithm



# The linear regression. Simple Trick

**input:**  $y = mx + b$ ,  $(x, y)$

Pick two very small random numbers:  $\eta 1$  and  $\eta 2$


$$\begin{aligned} m' &= m - \eta 1. \\ b' &= b + \eta 2. \end{aligned}$$

$$\begin{aligned} m' &= m + \eta 1. \\ b' &= b + \eta 2. \end{aligned}$$

$$\begin{aligned} m' &= m + \eta 1. \\ b' &= b - \eta 2. \end{aligned}$$

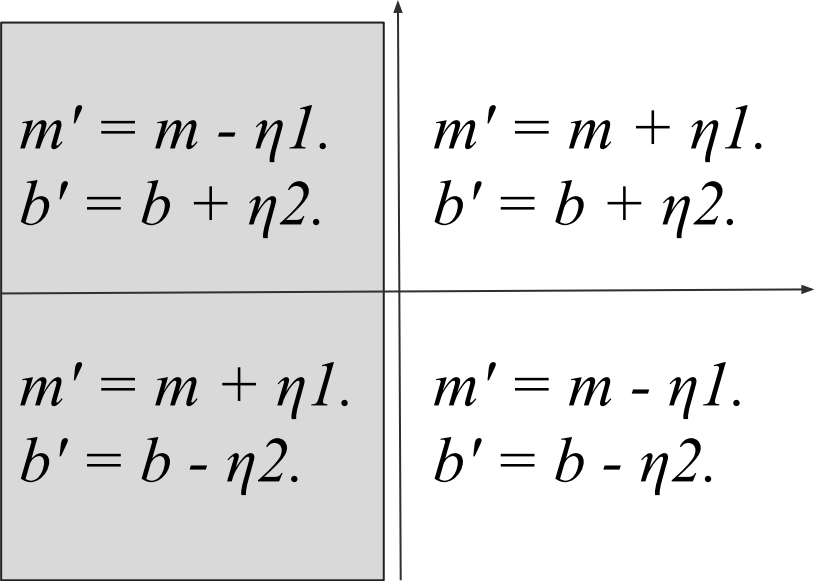
$$\begin{aligned} m' &= m - \eta 1. \\ b' &= b - \eta 2. \end{aligned}$$

**output:**  $y' = m'x + b'$  closer to  $(x, y)$

# The linear regression. Simple Trick

**input:**  $y = mx + b$ ,  $(x, y)$

Pick two very small random numbers:  $\eta 1$  and  $\eta 2$


$$\begin{aligned} m' &= m - \eta 1. \\ b' &= b + \eta 2. \end{aligned}$$

$$\begin{aligned} m' &= m + \eta 1. \\ b' &= b + \eta 2. \end{aligned}$$

$$\begin{aligned} m' &= m + \eta 1. \\ b' &= b - \eta 2. \end{aligned}$$

$$\begin{aligned} m' &= m - \eta 1. \\ b' &= b - \eta 2. \end{aligned}$$

- If the model gave us a price for the house that is lower than the actual price, add a small random amount to the price per room and to the base price of the house.
- If the model gave us a price for the house that is higher than the actual price, subtract a small random amount from the price per room and the base price of the house.

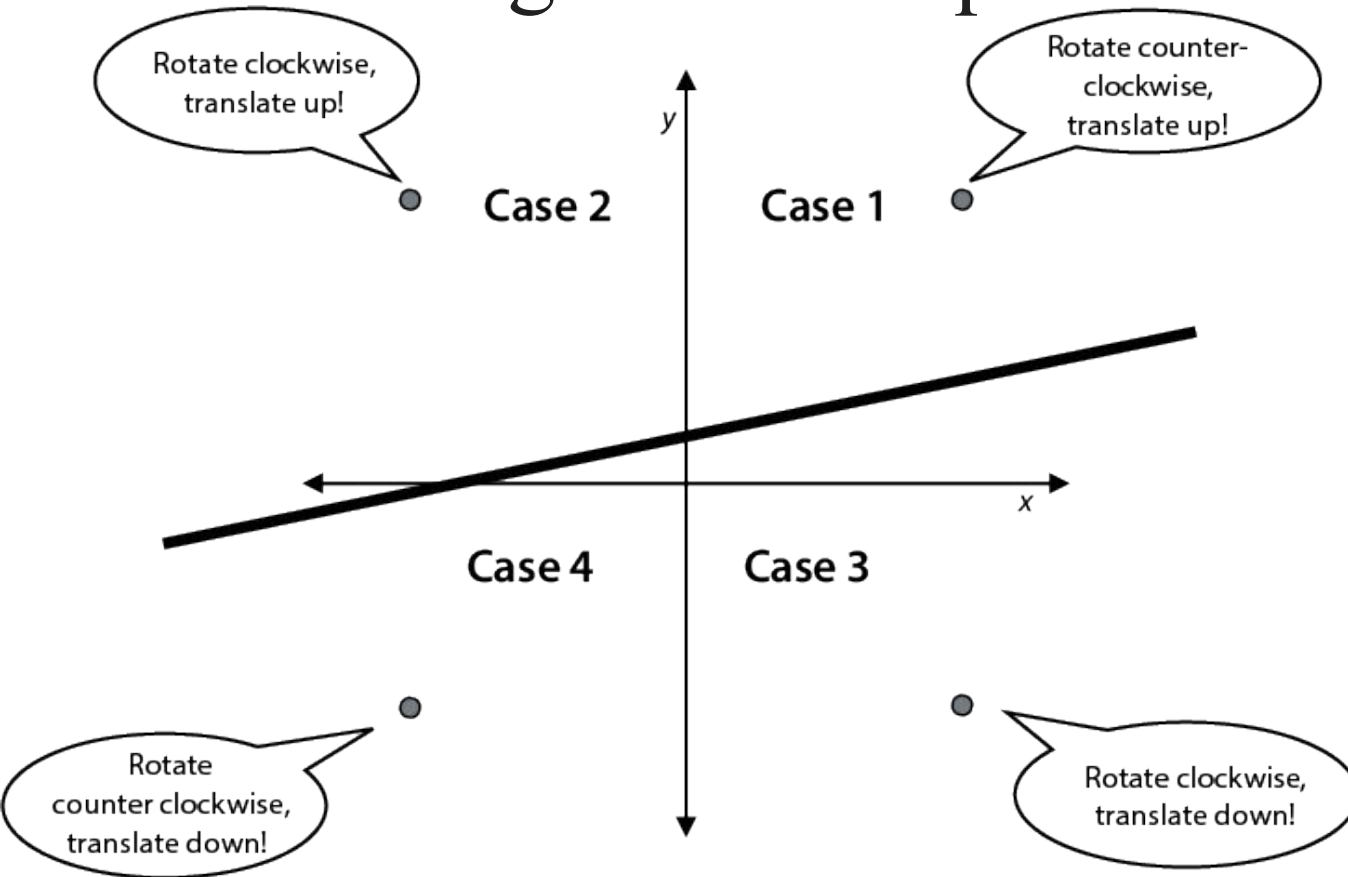
**output:**  $y' = m'x + b'$  closer to  $(x, y)$

# The linear regression. Square trick

**Move the line closer to point:**

**find values with the correct signs (+ or −) to add to the slope and the  $y$ -intercept.**

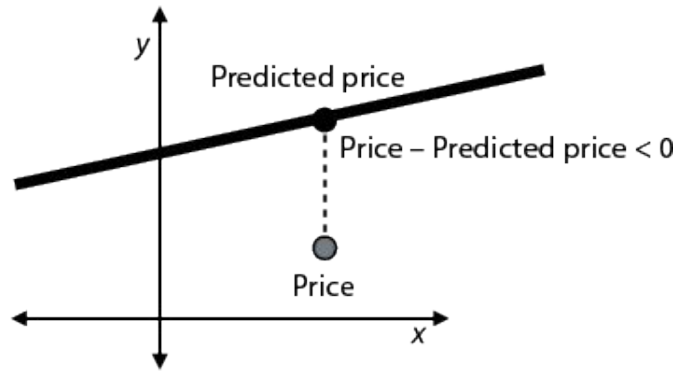
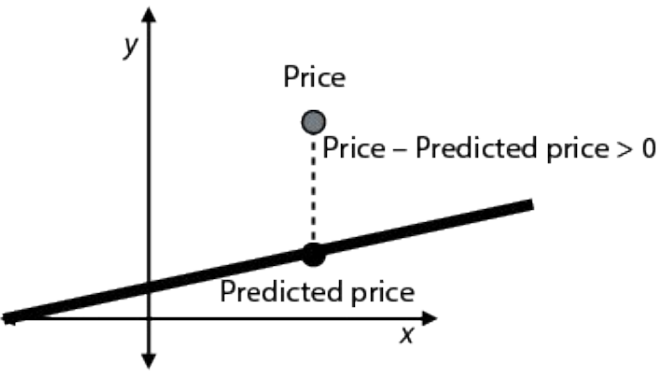
# The linear regression. Square trick



**Observation 1:** In the simple trick, when the point is above the line, we add a small amount to the  $y$ -intercept. When it is below the line, we subtract a small amount.



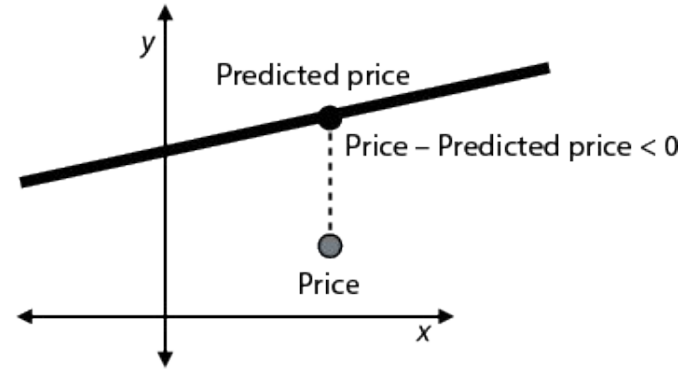
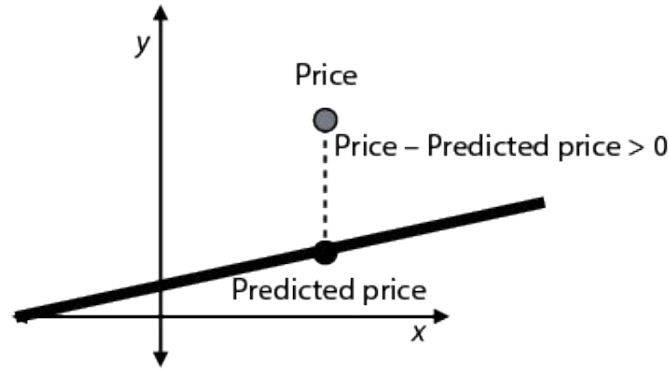
# The linear regression. Square trick



**Observation 2:** If a point is above the line, the value  $y - y'$  (the difference between the price and the predicted price) is positive. If it is below the line, this value is negative.

# The linear regression. Square trick

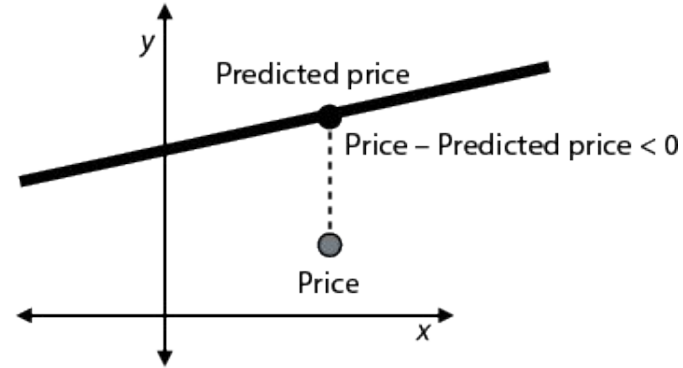
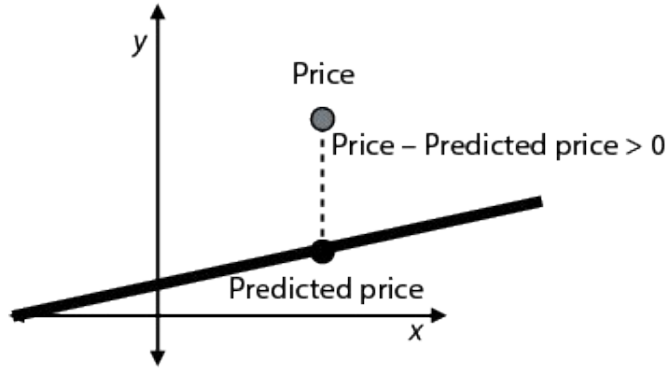
if we add the difference  $y - y'$  to the y-intercept, the line will always move toward the point



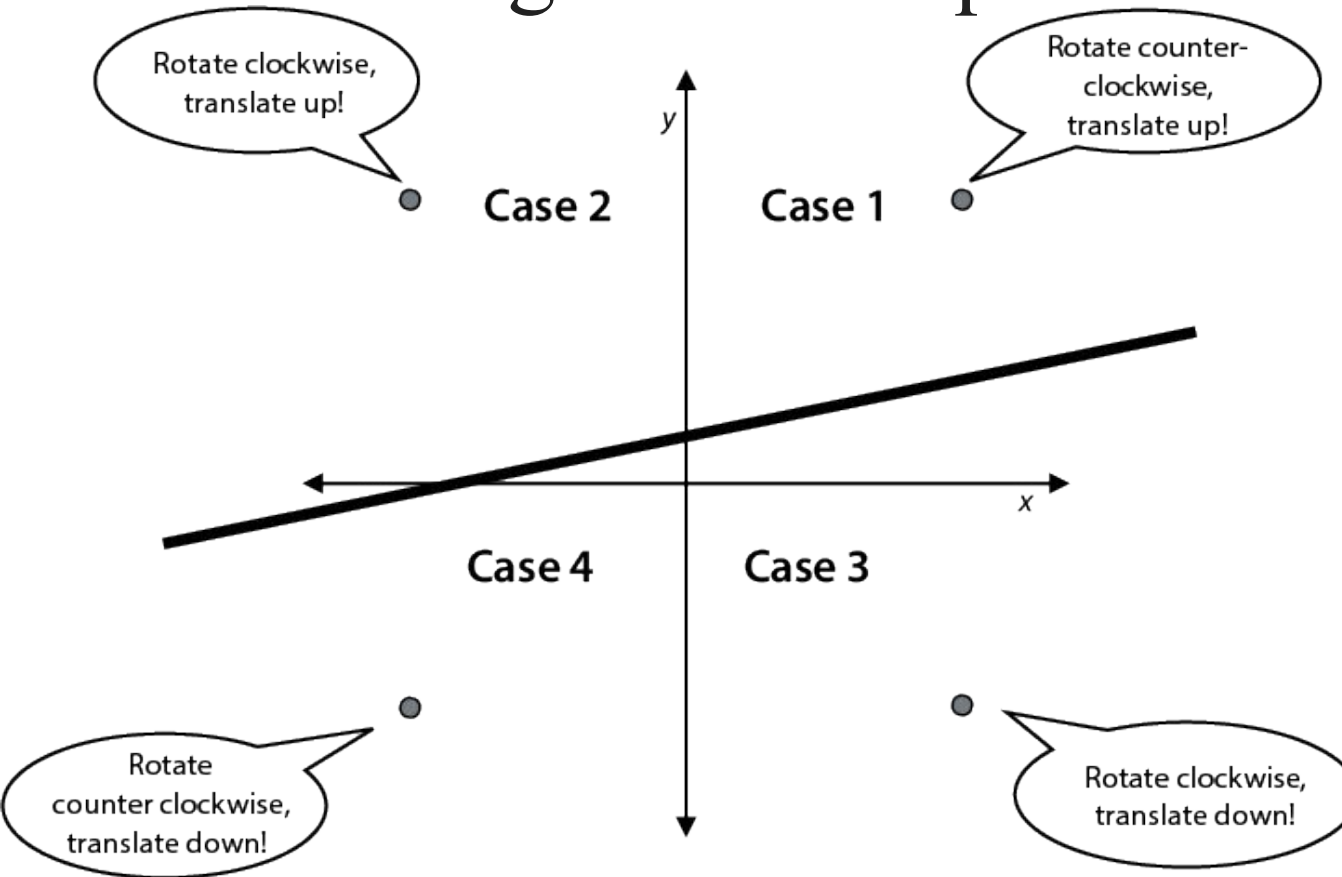
# The linear regression. Square trick

if we add the difference  $y - y'$  to the y-intercept, the line will always move toward the point

**learning rate** - A very small number  $\eta$ , which is set before training model to keep the changes in very small amounts by training. The value  $\eta \cdot (y - y')$  is added to the y-intercept to move the line in the direction of the point.

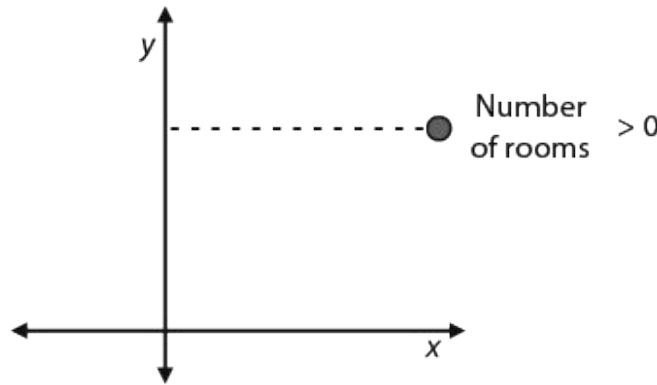
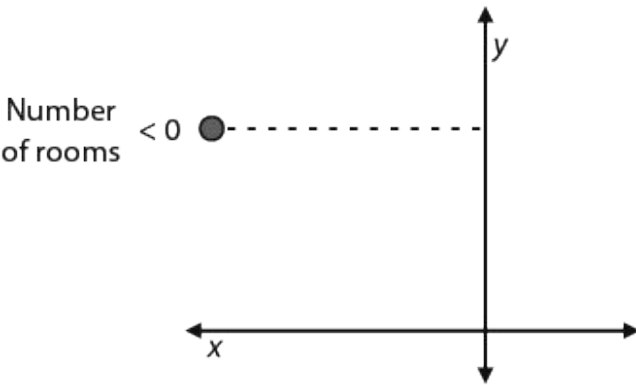


# The linear regression. Square trick



**Observation 3:** In the simple trick, when the point is in scenario 1 or 4 (above the line and to the right of the vertical axis, or below the line and to the left of the vertical axis), we rotate the line counterclockwise. Otherwise (scenario 2 or 3), we rotate it clockwise.

# The linear regression. Square trick

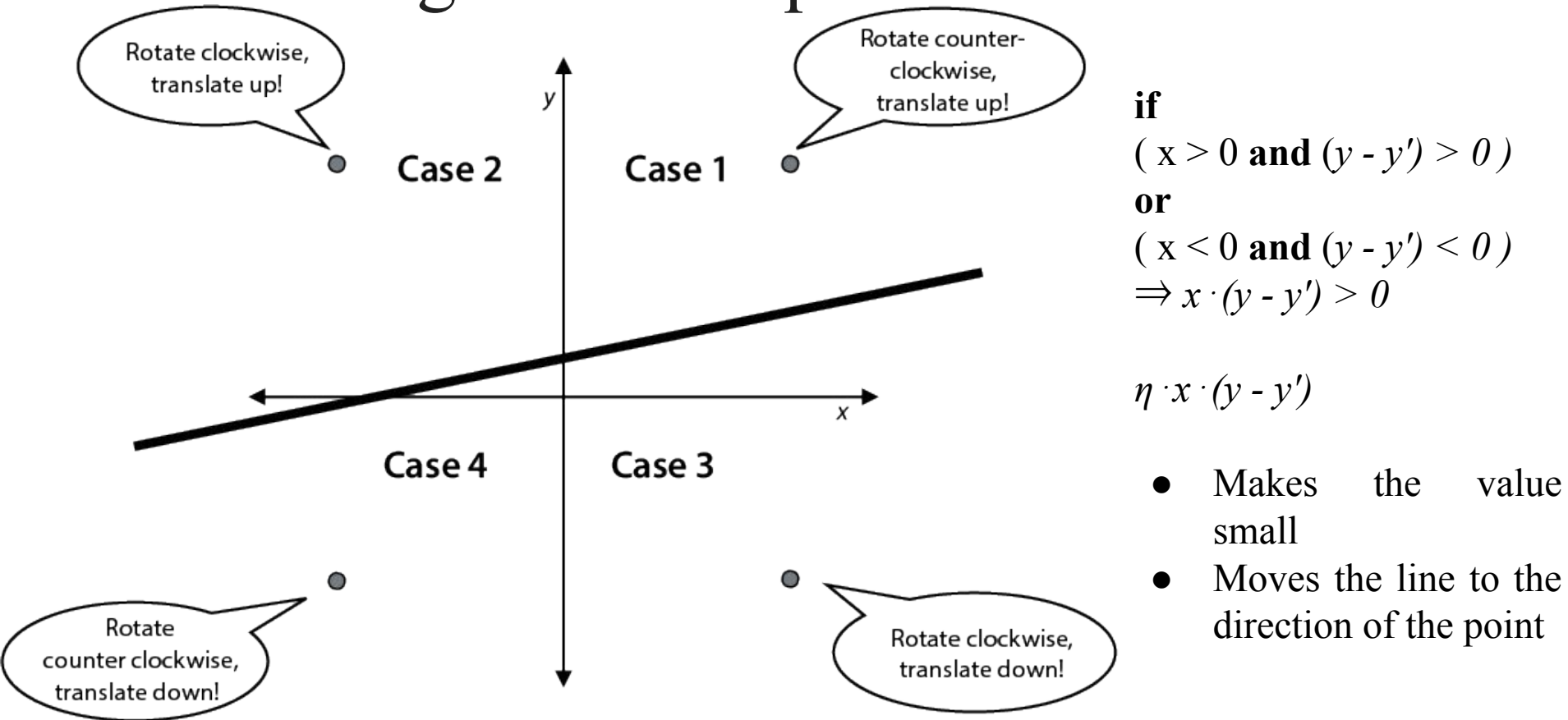


## Observation 4:

If a point  $(x, y)$  is to the right of the vertical axis, then  $x$  is positive.

If the point is to the left of the vertical axis, then  $x$  is negative.

# The linear regression. Square trick



# The linear regression. Square trick

input:

$$y = mx + b, (x, y), \eta$$

steps:

$$m' = m + \eta \cdot x \cdot (y - y') \text{ (rotate).}$$

$$b' = b + \eta \cdot (y - y') \text{ (translate).}$$

output:

price per room and base price:  $m', b'$

# The linear regression

Repeat the square trick many times to move the line closer to the points.

**Input:** a dataset of houses with number of rooms and prices

**Procedure:**

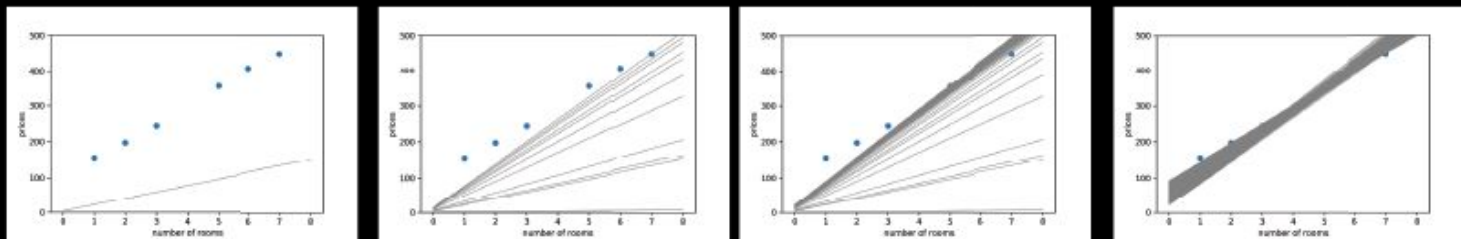
- Random values for  $m$  and  $b$ .

- Repeat many times:

  - Pick a random data point.

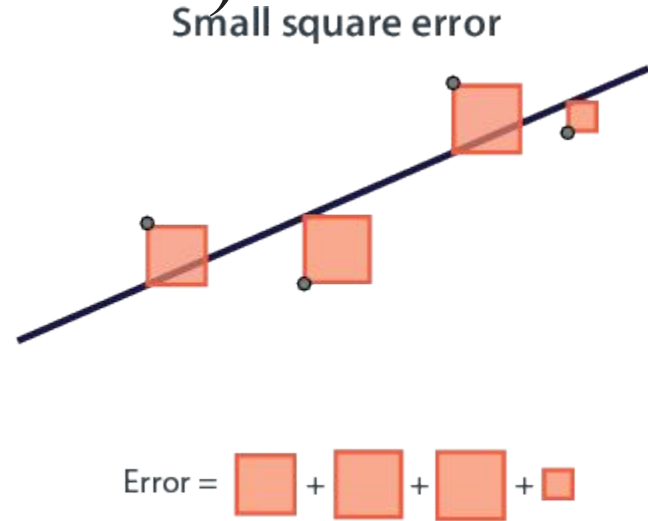
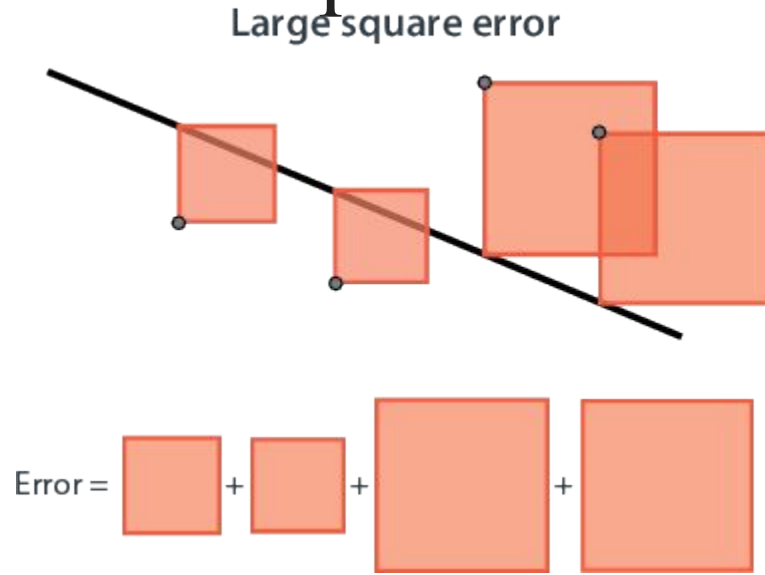
  - Update the slope and the y-intercept using the square trick.

**Output:**  $y' = m'x + b'$





# Root mean square error. (RMSE)

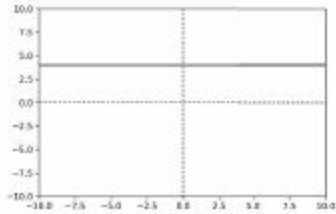


The model makes an error of around RMSE for any prediction we make.

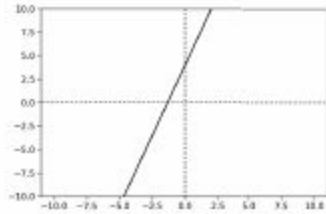
# Polynomial regression

If graph plotted from data is not a line, but a curve.

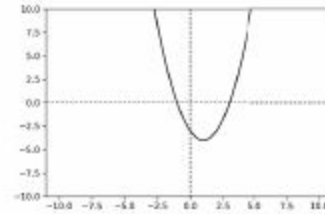
Function for nonlinear data - **polynomial**.



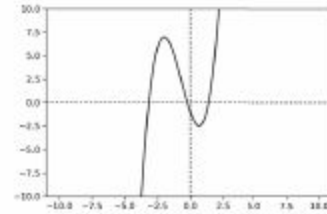
Degree 0  
 $y = 4$



Degree 1  
 $y = 3x + 4$

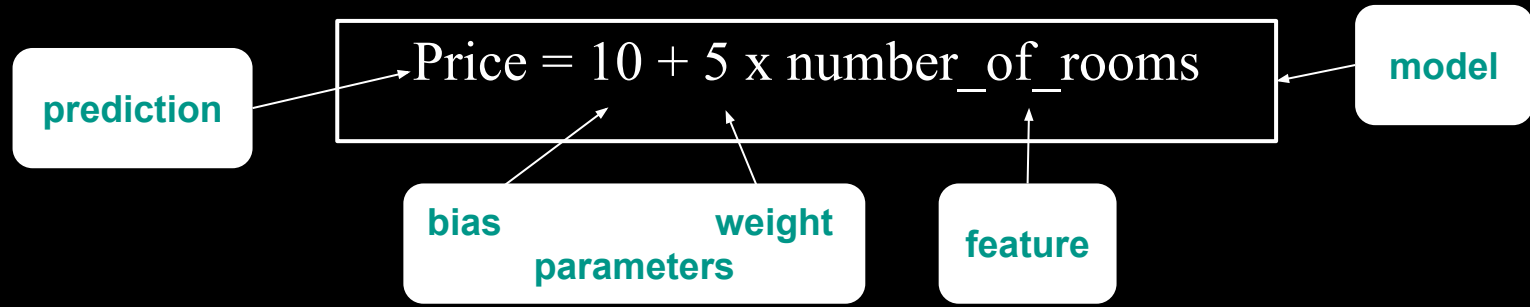


Degree 2  
 $y = x^2 - 2x - 3$



Degree 3  
 $y = x^3 + 2x^2 - 4x - 1$

# Parameters



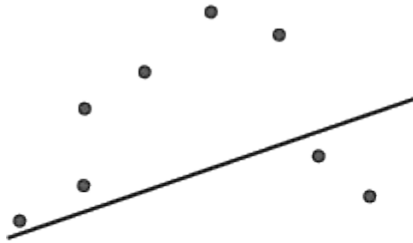
Quantity that the model creates/modifies during the training - parameter.

Quantity that you set before the training process - hyperparameter.

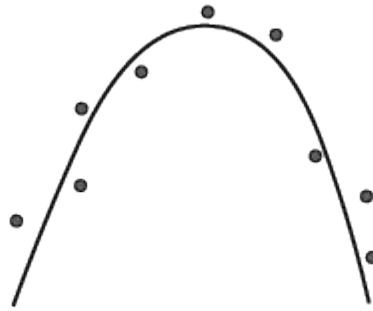
hyper parameters

```
function linear_regression(features, labels, learning_rate = 0.01, epochs = 100)
{ ... }
```

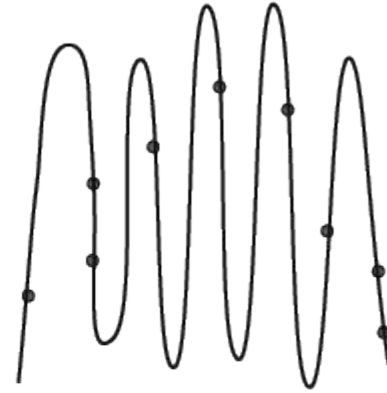
# Underfitting and overfitting data



**Model 1**  
Polynomial of degree 1  
(a line)



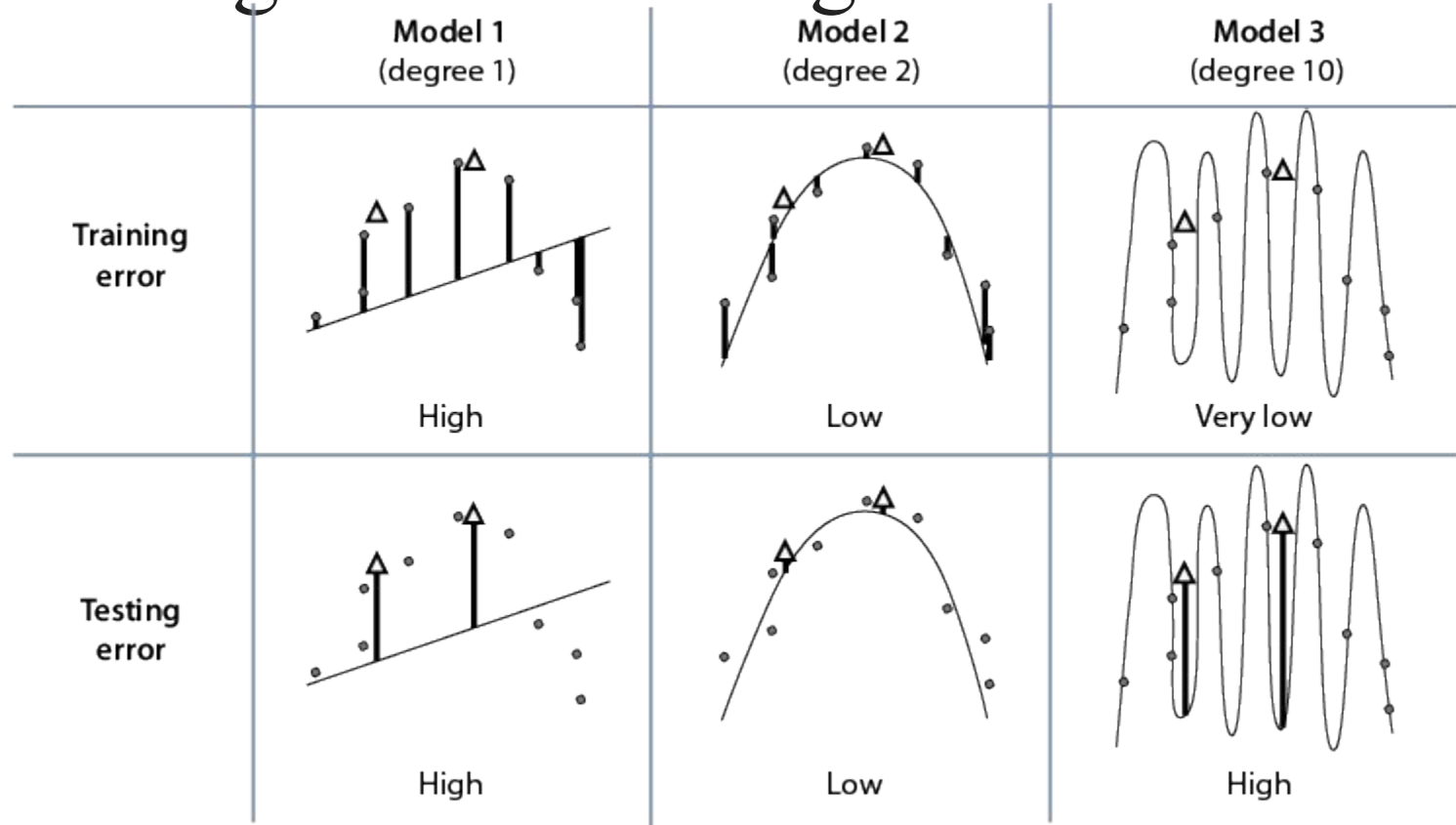
**Model 2**  
Polynomial of degree 2  
(a parabola)



**Model 3**  
Polynomial of degree 10

simple models tend to underfit and complex models tend to overfit

# Underfitting and overfitting data



# Training, Validation, Testing

80 % - training set: for training all models

10 % - validation set: for making decisions on which model to use

10 % - testing set: for checking how well the model did

# Regularization example 1



Problem: Broken roof



Roofer 1

Solution: Bandage  
(Underfitting)



Roofer 2

Solution: Shingles  
(Correct)



Roofer 3

Solution: Titanium  
(Overfitting)

**LEAK:**

1000 mL

1 mL

0 mL

**COST:**

1 \$

100\$

100 000\$

**Regularization:**

**LEAK+COST:**

1001

101

100000

# Regularization example 2 (recommend movie)

$$\hat{y} = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + w_6x_6 + w_7x_7 + w_8x_8 + w_9x_9 + w_{10}x_{10} + b$$

$x_i$  - the amount of time the user watched the movie for  $i = \overrightarrow{1,10}$

$\hat{y}$  - the amount of time the model predicts that the user will watch movie 11

$w_i$  - is the weight associated to movie  $i$

$b$  - bias

**Model 1:**  $\hat{y} = 2x_3 + 1.4x_7 - 0.5x_9 + 4$

**Model 2:**  $\hat{y} = 22x_1 - 103x_2 - 14x_3 + 109x_4 - 93x_5 + 203x_6 + 87x_7 - 55x_8 + 378x_9 - 25x_{10} + 8$



# Regularization Complexity

L1 and L2 - the model performance and complexity

L1 norm - sum absolute coefficient; L2 norm - square coefficient.

Model 1:  $\hat{y} = 2x_3 + 1.4x_7 - 0.5x_9 + 4$

Model 2:  $\hat{y} = 22x_1 - 103x_2 - 14x_3 + 109x_4 - 93x_5 + 203x_6 + 87x_7 - 55x_8 + 378x_9 - 25x_{10} + 8$

L1 norm:

Model 1:  $|2| + |1.4| + |-0.5| = 3.9$

Model 2:  $|22| + |-103| + |-14| + |109| + |-93| + |203| + |87| + |-55| + |378| + |-25| = 1089$

L2 norm:

Model 1:  $2^2 + 1.4^2 + (-0.5)^2 = 6.21$

Model 2:  $22^2 + (-103)^2 + (-14)^2 + 109^2 + (-93)^2 + 203^2 + 87^2 + (-55)^2 + 378^2 + (-25)^2 = 227,131$

# Regularization

$\lambda$  - regularization parameter (10, 1, 0.1, 0.01)

Error = Regression error (RMSE) +  $\lambda$  \* Regularization term (L1 or L2)

L1 - Turns some of the coefficients into zero. If we have too many features and we'd like to get rid of most of them.

L2 - Shrinks all the coefficients but rarely turns them into zero. If we have only few features and believe they are all relevant, then L2 regularization is what we need

# Regularization

Method 1:  $\text{if}(\text{weight} > 0) \text{ weight} = \text{weight} - \lambda,$   
 $\text{else if}(\text{weight} < 0) \text{ weight} = \text{weight} + \lambda$

Method 2:  $\text{if } \lambda = 0.01, \text{ weight} \times (1 - \lambda) \approx \text{weight} \times 1.$

$$\hat{y} = 2x_3 + 1.4x_7 - 0.5x_9 + 4, \lambda = 0.01$$

Method 1: 1.99, 1.39, 0.06

Method 2, 1.98, 1.386, and -0.495

# Self study

Serrano, L., 2021. Grokking machine learning. Simon and Schuster.,  
Chapter 3, Chapter 4

Video lectures: [chapter 1, 2](#), [chapter 3](#);

Serrano, L., 2021. Grokking machine learning. Simon and Schuster.,  
Chapter 4

# Summary

- Regression is an important part of machine learning. It consists of training an algorithm with labeled data and using it to make predictions on future (unlabeled) data.
- Labeled data is data that comes with labels, which in the regression case, are numbers. For example, the numbers could be prices of houses.
- In a dataset, the features are the properties that we use to predict the label. For example, if we want to predict housing prices, the features are anything that describes the house and which could determine the price, such as size, number of rooms, school quality, crime rate, age of the house, and distance to the highway.
- The linear regression method for predicting consists in assigning a weight to each of the features and adding the corresponding weights multiplied by the features, plus a bias.
- Graphically, we can see the linear regression algorithm as trying to pass a line as close as possible to a set of points.
- The way the linear regression algorithm works is by starting with a random line and then slowly moving it closer to each of the points that is misclassified, to attempt to classify them correctly.

# Summary

Two problems that come up quite often are underfitting and overfitting.

Underfitting - very simple model to fit dataset. Overfitting - an overly complex model to fit dataset.

An effective way to tell overfitting and underfitting apart is by using a testing dataset.

Split the data into two sets to test a model: a training set and a testing set. The training set - train the model, and the testing set - evaluate the model.

Never use testing data for training or making decisions in models.

The validation set is another portion of dataset used to make decisions about the hyperparameters in model.

A model that underfits will perform poorly in the training set and in the validation set. A model that overfits will perform well in the training set but poorly in the validation set. A good model will perform well on both the training and the validation sets.

The model complexity graph is used to determine the correct complexity of a model, so that it doesn't underfit or overfit.

Regularization is a very important technique to reduce overfitting in machine learning models. It consists of adding a measure of complexity (regularization term) to the error function during the training process.

The L1 and L2 norms are the two most common measures of complexity used in regularization.

Using the L1 norm leads to L1 regularization, or lasso regression. Using the L2 norm leads to L2 regularization, or ridge regression.

L1 regularization is recommended when our dataset has numerous features, and we want to turn many of them into zero. L2 regularization is recommended when our dataset has few features, and we want to make them small but not zero.