

Regression

With Linear Regression

MODALG SoSe18

News & Motivation

META MARATHON ARTIFICIAL INTELLIGENCE

42 radical-explorative hours
NRW-Forum Düsseldorf
May 25 – 27, 2018

42 Stunden nonstop Talks, Performances, Filme, Konzerte, Ausstellung und Workshops zum **Thema Künstliche Intelligenz**: Der META Marathon ist ein neuartiges Technologie-Festival, das vom 25. bis 27. Mai 2018 im NRW-Forum Düsseldorf stattfindet. Die Teilnehmer gestalten das Festival selbst, wechseln die Rollen vom Experten zum Laien und experimentieren mit dem neuen Format – **inklusive Übernachtung vor Ort**. Festivaldirektor ist der Futurist und Unternehmer Christopher Peterka.

Advantages:

- Participants who can show a **three day participation** receive **three points** for this class
- Three days for free – **do it!!!**
- **Networking** and learn cool **AI-Stuff**
- Our Workshop: Reinforcement-Learning with DQN's
- Timeslot: Sa. 26.5., 11 am. to 15 pm.

Important:

- Send me your confirmation and your name **till 16.05.18, 18 pm.**
- **marcel.tiator@study.hs-duesseldorf.de**
- Use subject: „**META Marathon**“
- More Information: <https://www.nrw-forum.de/veranstaltungen/meta-marathon>




```
if char11.name[get]script src=(  
response?
```

```
(245,23,068,789,a48) [lock.command]
```

```
if #key_input
```

```
// script src= address [ #b dq ]  
status.command
```

```
logged:input.[true]
```

```
if ("true") add.string< status> (
```

```
logged:input.false function
```

```
if response?  
[lock.command]  
#key_input
```

```
if script src=[true]
```




Topics:

- **Supervised Learning & Classification**
- **Regression**
- **Linear Regression**
- **Gradient Descent**
- **Introduction Tensorflow**
- **Programming Exercise**

Machine Learning

**Supervised
Learning**

Classification

Regression

**Reinforcement
Learning**

**Unsupervised
Learning**

Clustering

$$y = f(x)$$

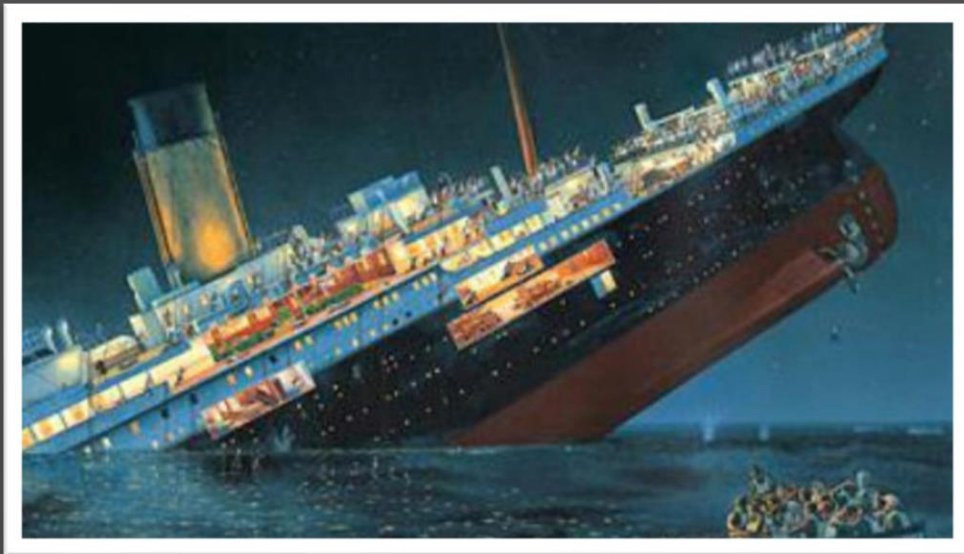
What is the aim of classification?

- **Differentiation**
- $y \in \{Partymaus, Spießer, Freak\}$



Tabular Data

$y: \text{Survived}$
 $y \in \{0,1\}$



data - Excel

Überprüfen Ansicht Add-Ins Team Was möchten Sie tun?

Daten abrufen und transformieren Abfragen und Verbindungen

Abfragen und Verbindungen

Sortieren und Filtern

MÖGLICHER DATENVERLUST

Einige Funktionen gehen möglicherweise verloren, wenn Sie diese Arbeitsmappe im CSV-Format (Trennzeichen getrennt) speichern. Um diese Funktionen zu erhalten, speichern Sie sie in einem Excel-Dateiformat.

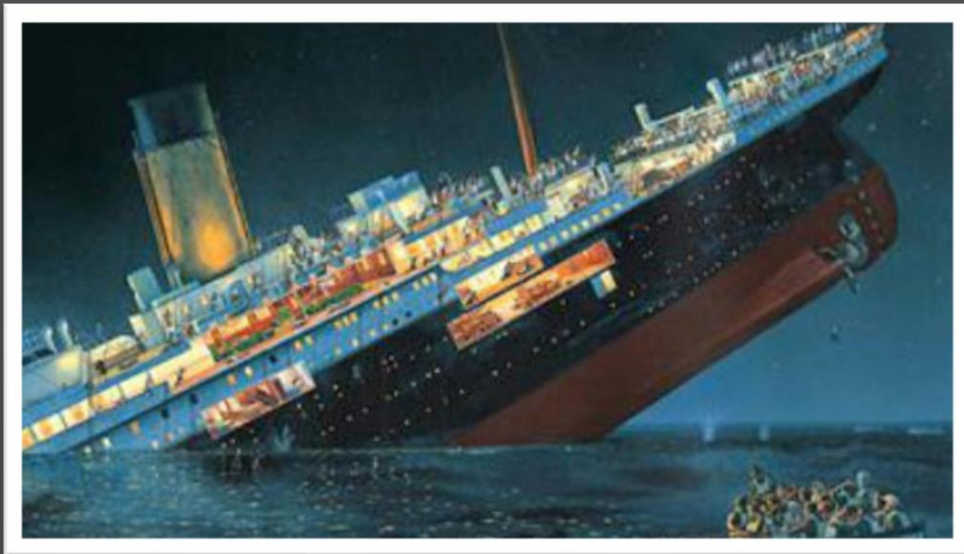
Nicht mehr anzeigen Speichern unter...

	A	B	C	D	E	F	G	H
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
2	1	0	3	Braund, Mr. C	male	22	1	0
3	2	1	1	Cumings, Mrs.	female	38	1	0
4	3	1	3	Heikkinen, Mif	female	26	0	0
5	4	1	1	Futrelle, Mrs.	female	35	1	0
6	5	0	3	Allen, Mr. W	male	35	0	0
7	6	0	3	Moran, Mr. J	male		0	0
8	7	0	1	McCarthy, M	male	54	0	0
9	8	0	3	Palsson, Mas	male	2	3	1
10	9	1	3	Johnson, Mrs	female	27	0	2
11	10	1	2	Nasser, Mrs.	female	14	1	0
12	11	1	3	Sandstrom, M	female	4	1	1
13	12	1	1	Bonnell, Miss	female	58	0	0
14	13	0	3	Saunderscock	male	20	0	0
15	14	0	3	Andersson, M	male	39	1	5
16	15	0	3	Vestrom, Mis	female	14	0	0
17	16	1	2	Hewlett, Mrs.	female	55	0	0
18	17	0	3	Rice, Master.	male	2	4	1
19	18	1	2	Williams, Mr.	male		0	0
20	19	0	3	Vander Plank	female	31	1	0
21	20	1	3	Masselmani,	female		0	0
22	21	0	2	Fynney, Mr. J	male	35	0	0
23	22	1	2	Beesley, Mr. I	male	34	0	0
24	23	1	3	McGowan, M	female	15	0	0
25	24	1	1	Sloper, Mr. W	male	28	0	0

Tabular Data

$x: class, sex, age$

e.g. $x = (1, 1, 45)^T$



data - Excel

Überprüfen Ansicht Add-Ins Team Was möchten Sie tun?

Daten abrufen und transformieren Abfragen und Verbindungen Alle aktualisieren Verknüpfungen bearbeiten Abfragen und Verbindungen Sortieren und Filtern Sortieren und Filtern Erweitern

MÖGLICHER DATENVERLUST Einige Funktionen gehen möglicherweise verloren, wenn Sie diese Arbeitsmappe im CSV-Format (Trennzeichen getrennt) speichern. Um diese Funktionen zu erhalten, speichern Sie sie in einem Excel-Dateiformat. Nicht mehr anzeigen Speichern unter...

	A	B	C	D	E	F	G	H
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch
2	1	0	3	Braund, Mr. male		22	1	0
3	2	1	1	Cumings, Mrs. female		38	1	0
4	3	1	3	Heikkinen, Mif. female		26	0	0
5	4	1	1	Futrelle, Mrs. female		35	1	0
6	5	0	3	Allen, Mr. Wil. male		35	0	0
7	6	0	3	Moran, Mr. J. male			0	0
8	7	0	1	McCarthy, M. male		54	0	0
9	8	0	3	Palsson, Mas. male		2	3	1
10	9	1	3	Johnson, Mrs. female		27	0	2
11	10	1	2	Nasser, Mrs. female		14	1	0
12	11	1	3	Sandstrom, M. female		4	1	1
13	12	1	1	Bonnell, Miss. female		58	0	0
14	13	0	3	Saunderscock, male		20	0	0
15	14	0	3	Andersson, M. male		39	1	5
16	15	0	3	Vestrom, Mis. female		14	0	0
17	16	1	2	Hewlett, Mrs. female		55	0	0
18	17	0	3	Rice, Master. male		2	4	1
19	18	1	2	Williams, Mr. male			0	0
20	19	0	3	Vander Plank. female		31	1	0
21	20	1	3	Masselmani, female			0	0
22	21	0	2	Fynney, Mr. J. male		35	0	0
23	22	1	2	Beesley, Mr. I. male		34	0	0
24	23	1	3	McGowan, M. female		15	0	0
25	24	1	1	Sloper, Mr. male		28	0	0

Discrete vs. Continuous

$$y \in N^n$$

Example:

$$y \in \{0,1,2,3,4\}$$
$$y \in \{(1,0,0),$$
$$(0,1,0),$$
$$(0,0,1)\}$$

$$y \in R^n$$

Example:

$$y \in \{0.1, -0.43, 3.44\}$$
$$y \in \{(-0.2, 0.3, -0.7),$$
$$(3.1, 5.0, -0.62),$$
$$(0.44, -0.4, 4.5)\}$$

Regression

y: DAX



y: DAX



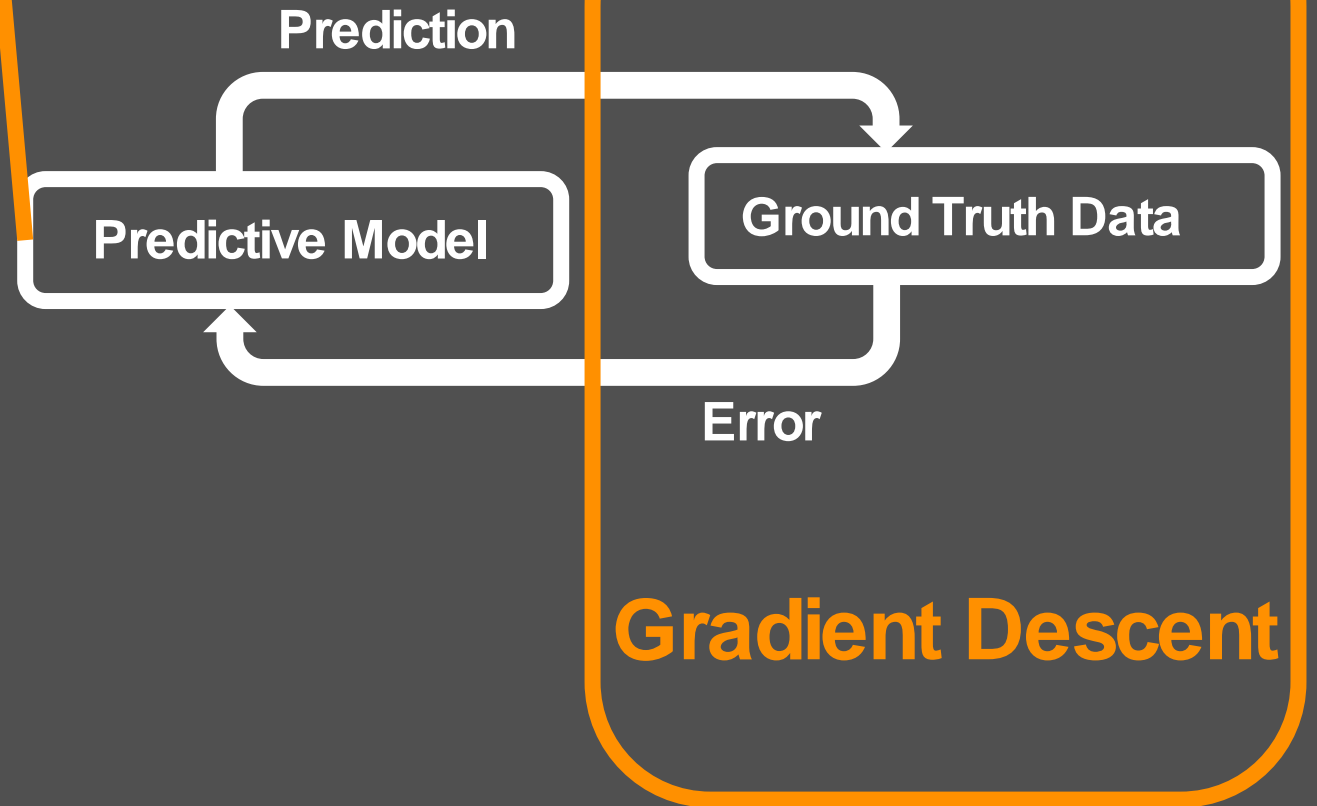
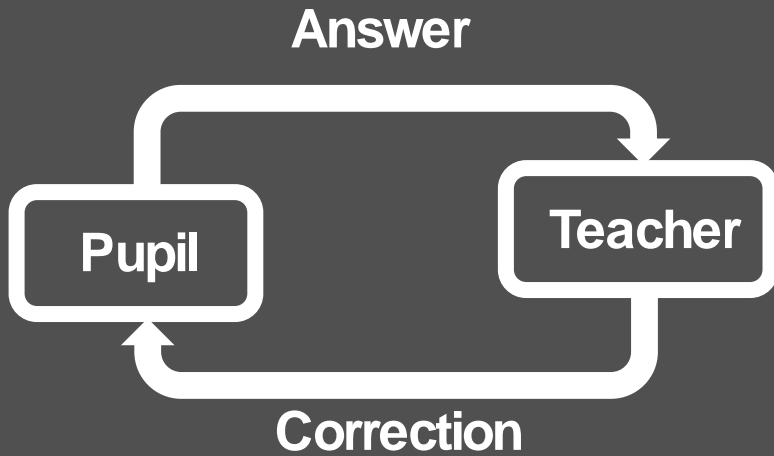
x: Dow Jones



Date	x: Dow Jones	y: DAX
08.05.18	24,300	12,940
09.05.18	24,400	12,910
10.05.18	24,600	12,975
11.05.18	24,800	13,020
12.05.18	24,600	???

Today:

Linear Regression



Predictive Model:

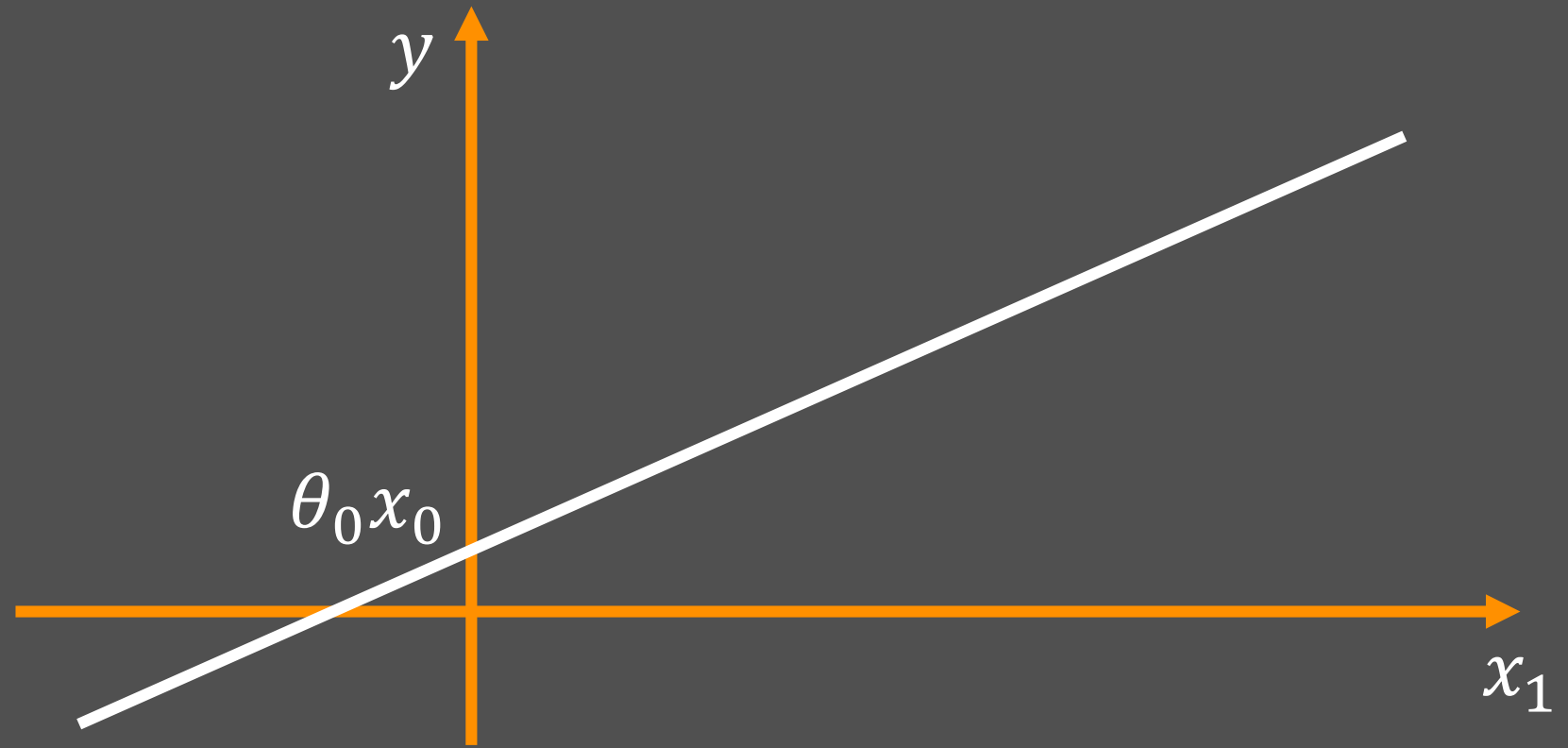
- $\hat{y} = \theta_1 x_1 + \theta_0 x_0$
- **Weights:** $\theta_0, \theta_1 \in R$
- $x_0 = 1$

Predictive Model:

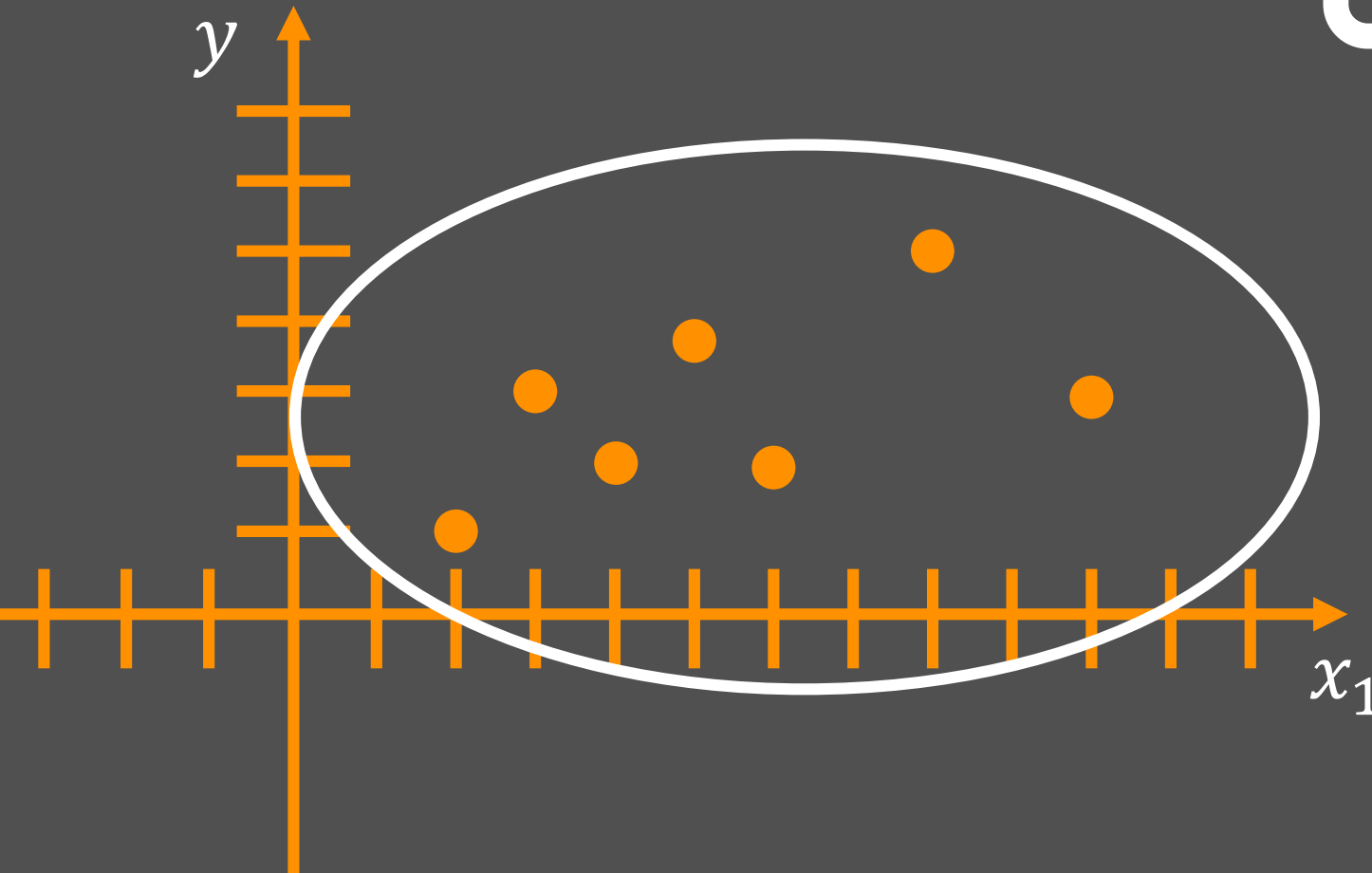
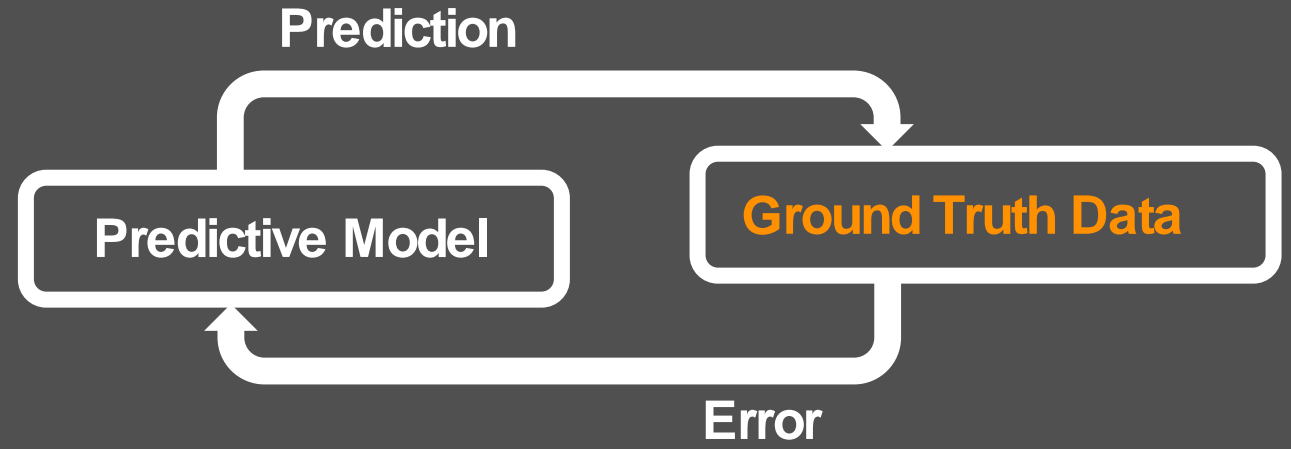
- $\hat{y} = mx + b$
- $m = \theta_1$
- $x = x_1$
- $b = \theta_0 x_0$

Slope of the line depends on weights θ_0, θ_1

$$\hat{y} = \theta_1 x_1 + \theta_0 x_0$$

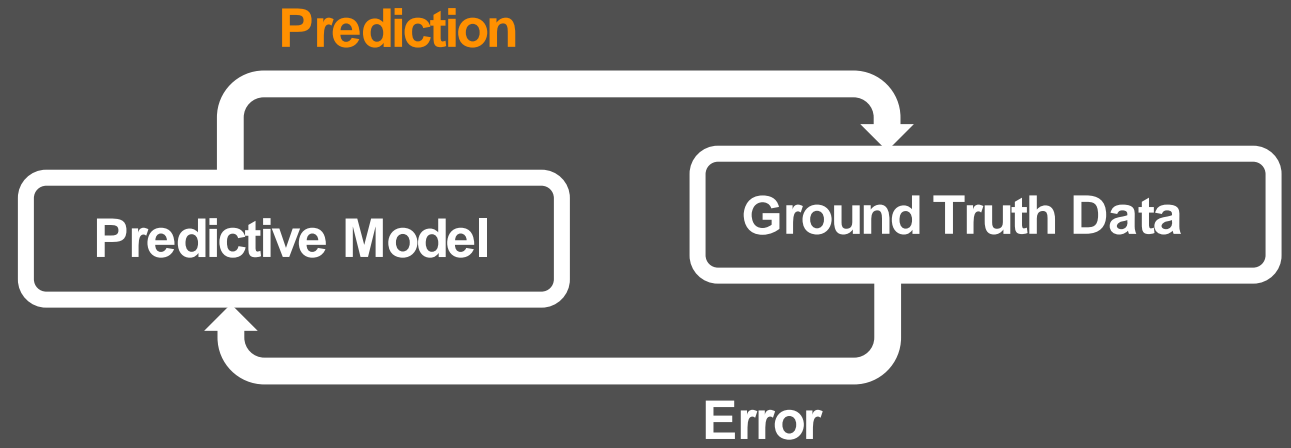
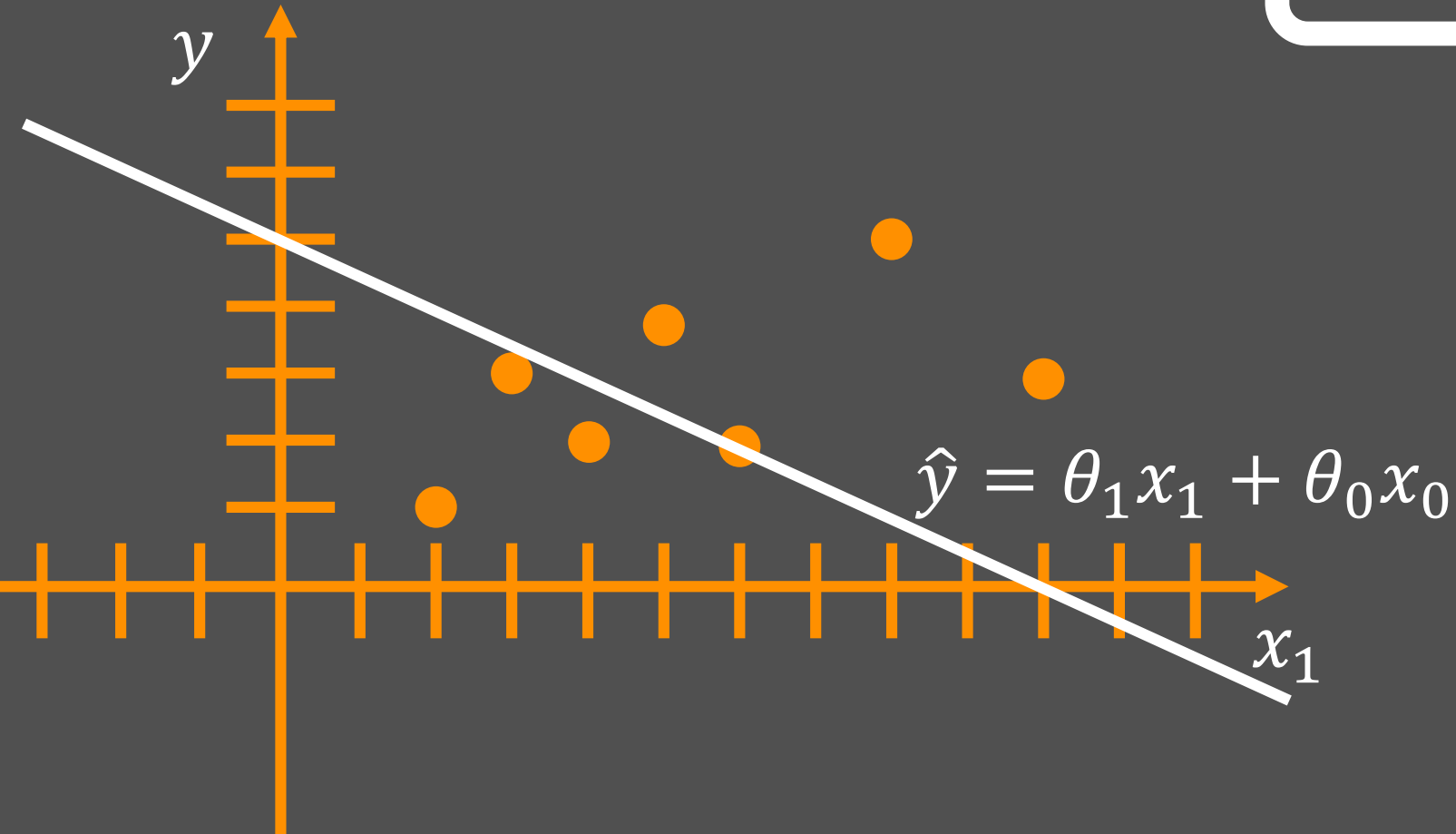


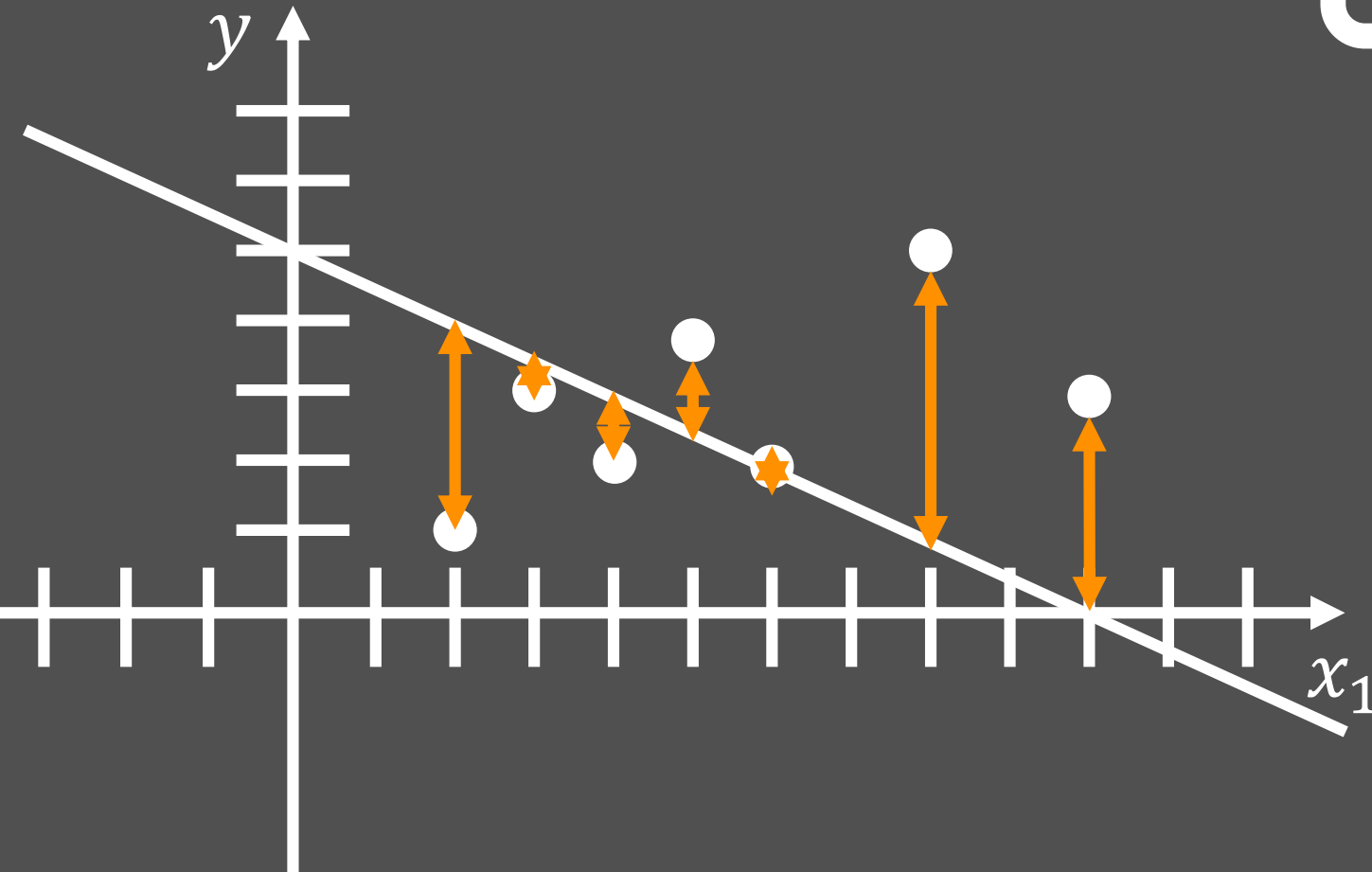
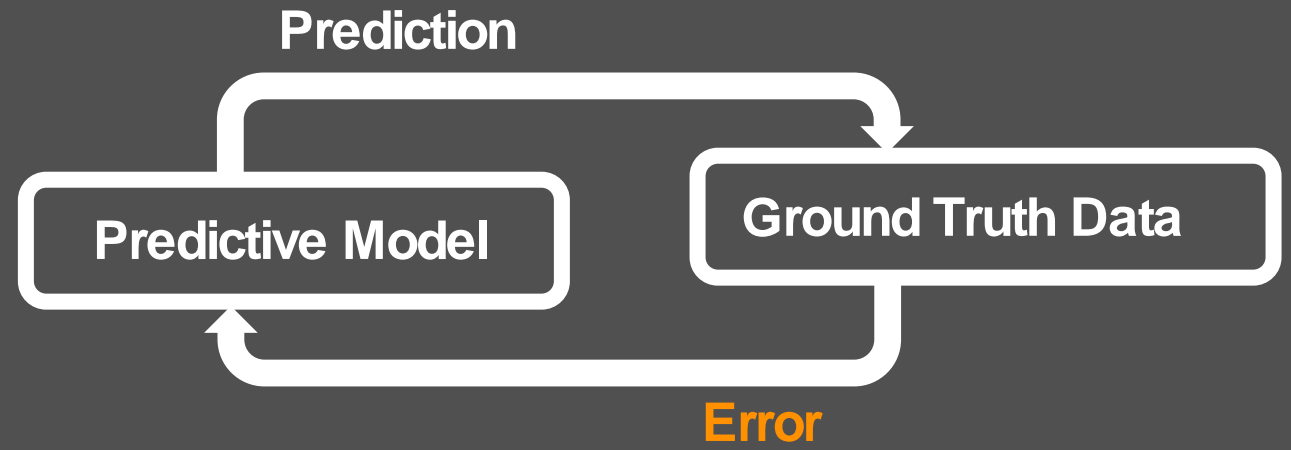
Find good weights!



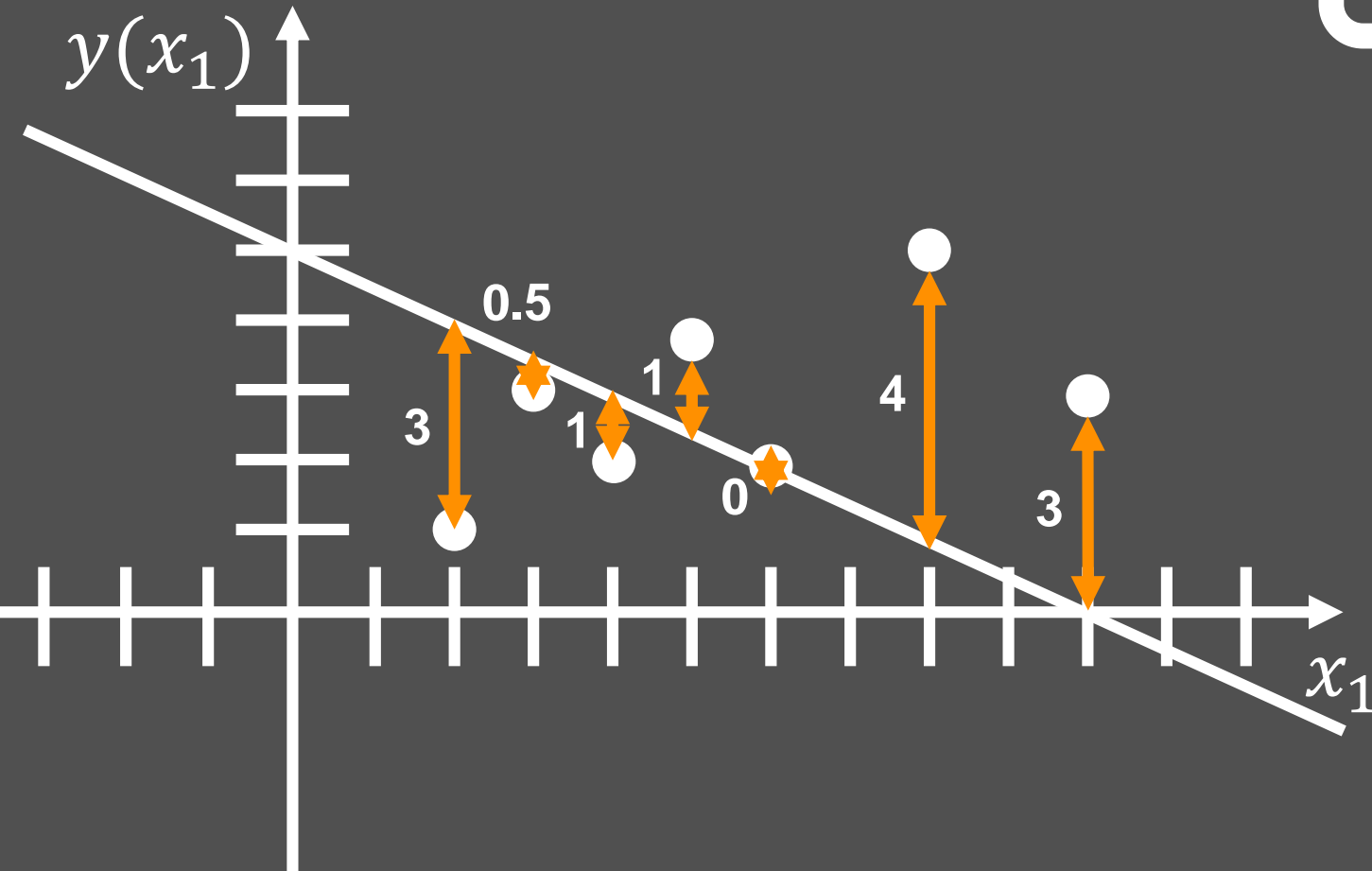
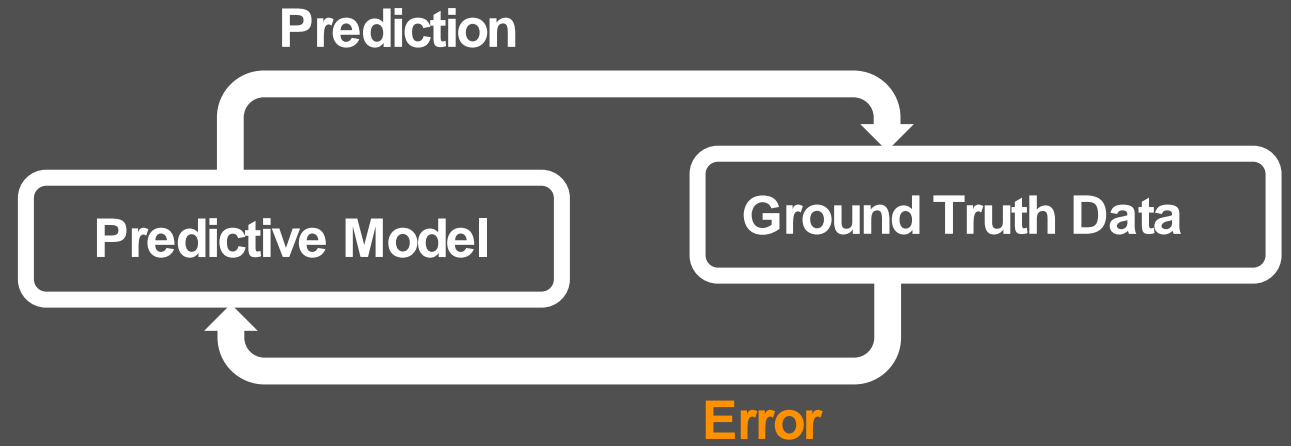
Ground Truth Data

Random Weights



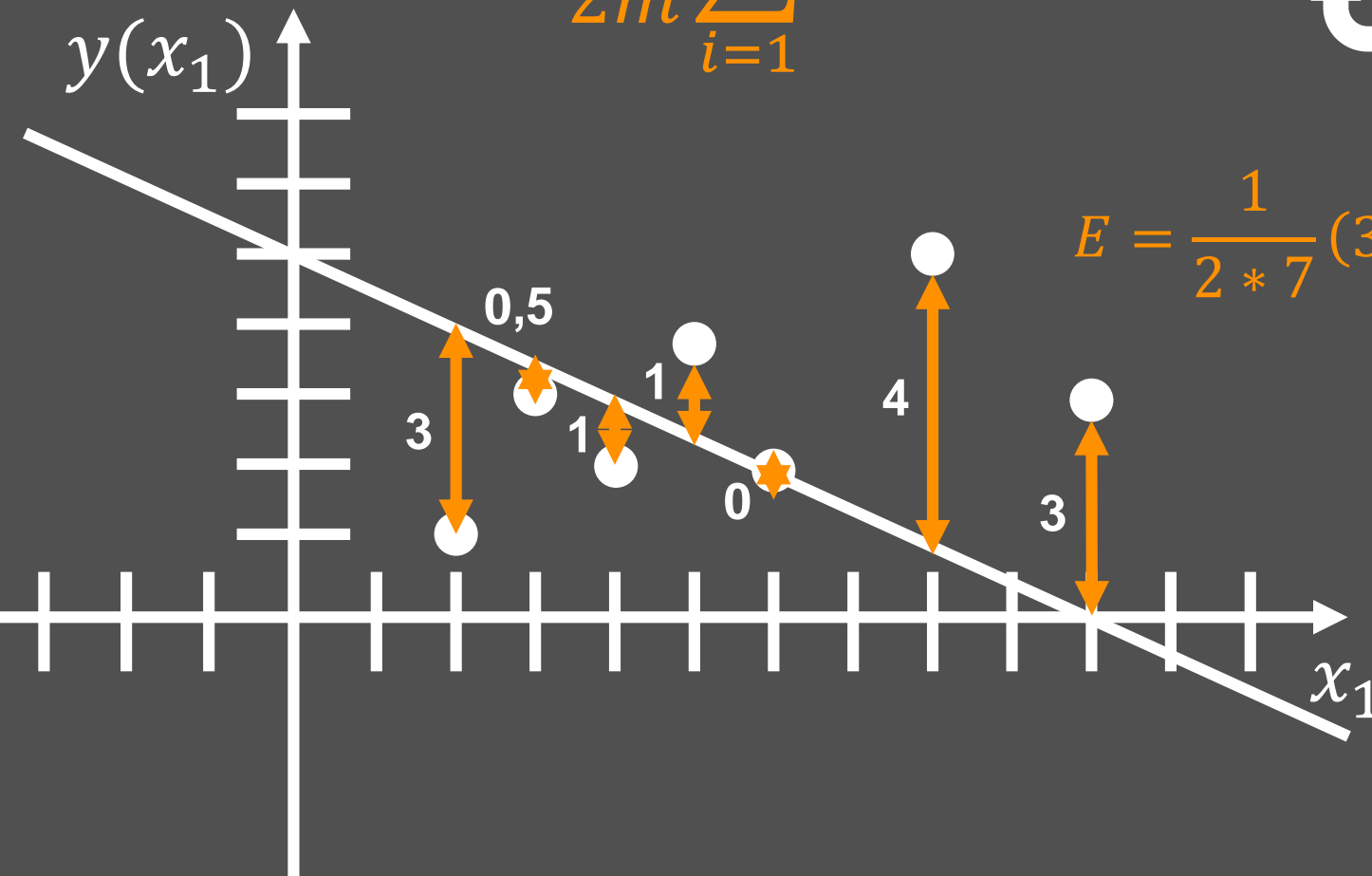
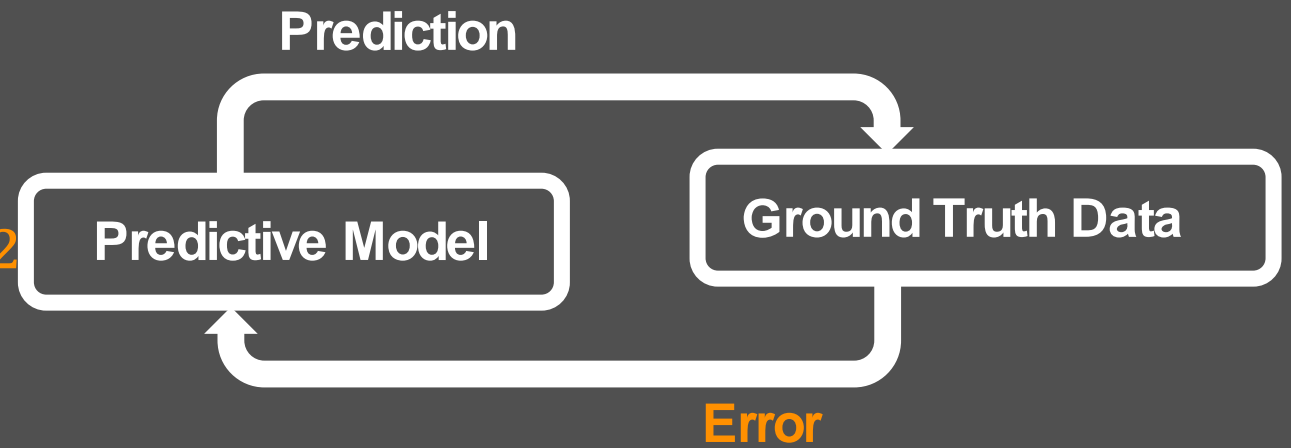


$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$$



$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$$

$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$$



$$E = \frac{1}{2 * 7} (3^2 + 0,5^2 + 1^2 + 1^2 + 0^2 + 4^2 + 3^2)$$

$$E = 2.859$$

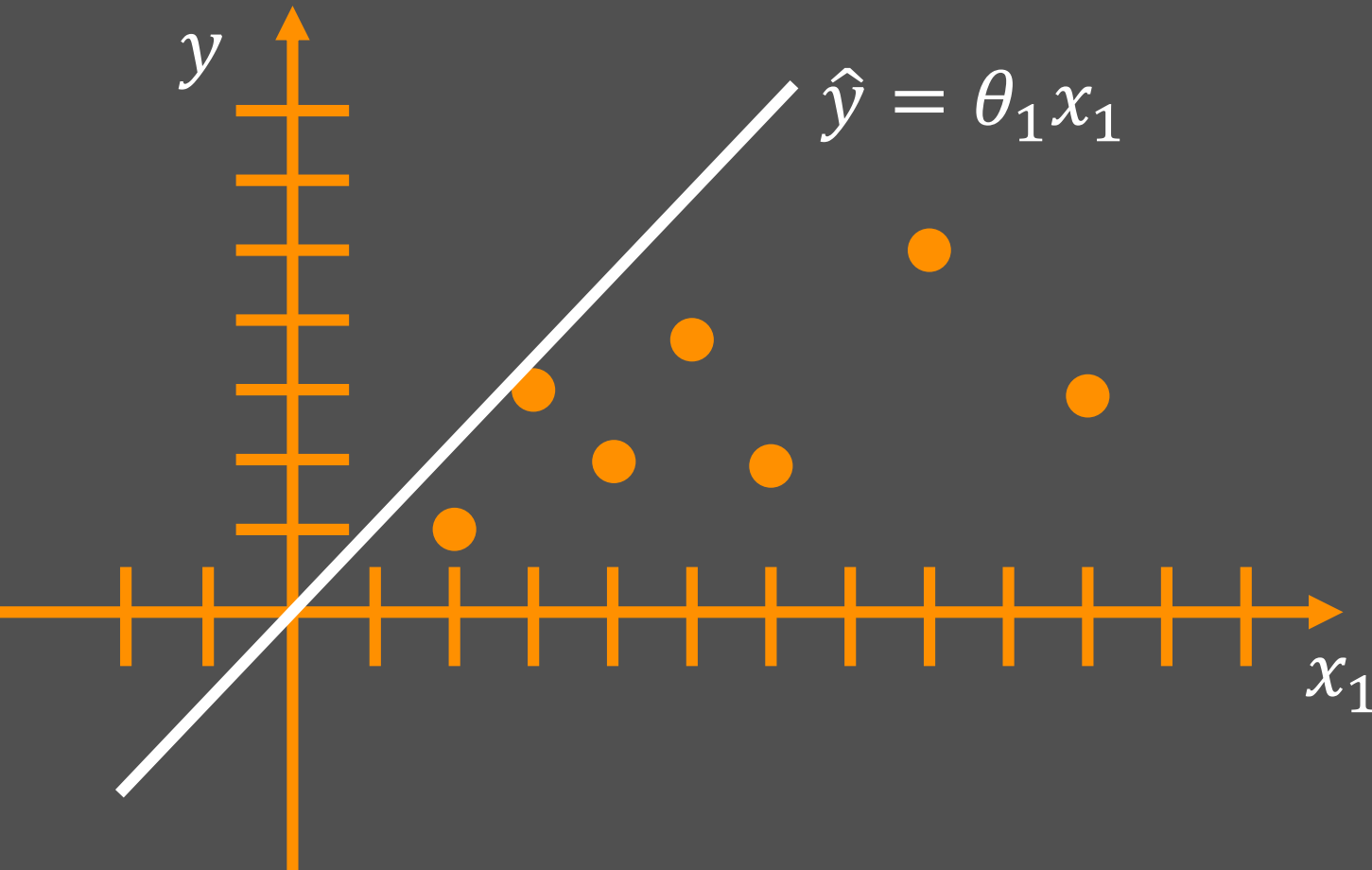
$$m = 7$$

Prediction would be as good as error of 2,589

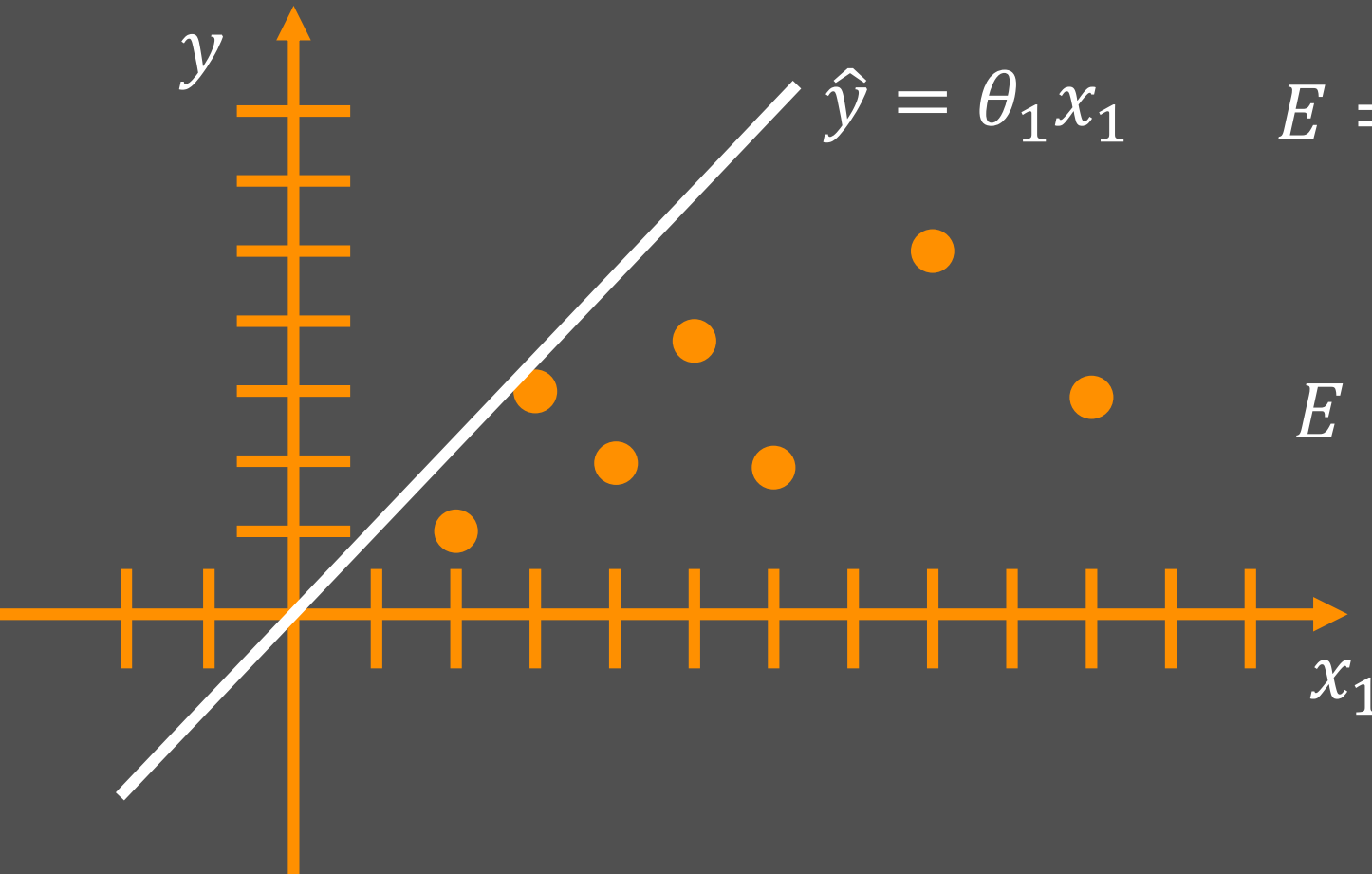
Gradient Descent:

- Tune weights: θ_0, θ_1
- Minimize error
- Formally: $\min_{\theta_0, \theta_1} E(\theta_0, \theta_1)$
- Prediction fits better to data
- Training Process

Let's facilitate the problem for visualization



Gradient Descent



$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$$

$$E = \frac{1}{2m} \sum_{i=1}^m ((\theta_1 x_{i,1}) - y_i)^2$$

An orange arrow points from the \hat{y}_i term in the first equation to the $(\theta_1 x_{i,1})$ term in the second equation.

Gradient Descent

$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^m ((\theta_1 x_{i,1}) - y_i)^2$$

Want: $\min_{\theta_1} E(\theta_1)$

Calculate: $\frac{\partial E(\theta_1)}{\partial \theta_1} = 2 * \frac{1}{2m} * \sum_{i=1}^m ((\theta_1 x_{i,1}) - y_i) * x_{i,1}$

$$\frac{\partial E(\theta_1)}{\partial \theta_1} = \frac{1}{m} \sum_{i=1}^m ((\theta_1 x_{i,1}) - y_i) * x_{i,1}$$

Tune weight: θ_1

$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_1)}{\partial \theta_1}$$

Algorithm:

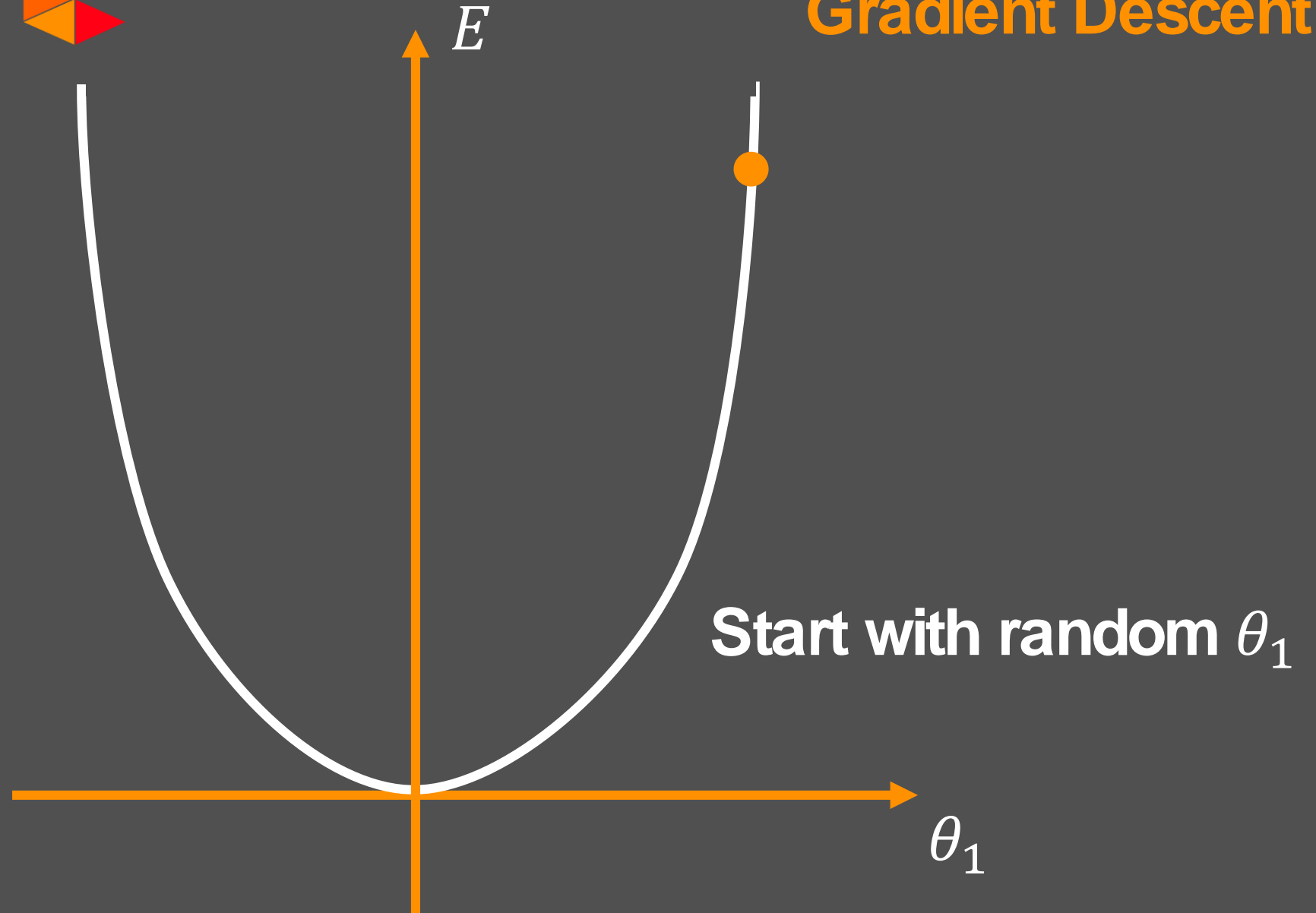
Start with random θ_1

repeat until convergence{

$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_1)}{\partial \theta_1}$$

}

Gradient Descent



Gradient Descent

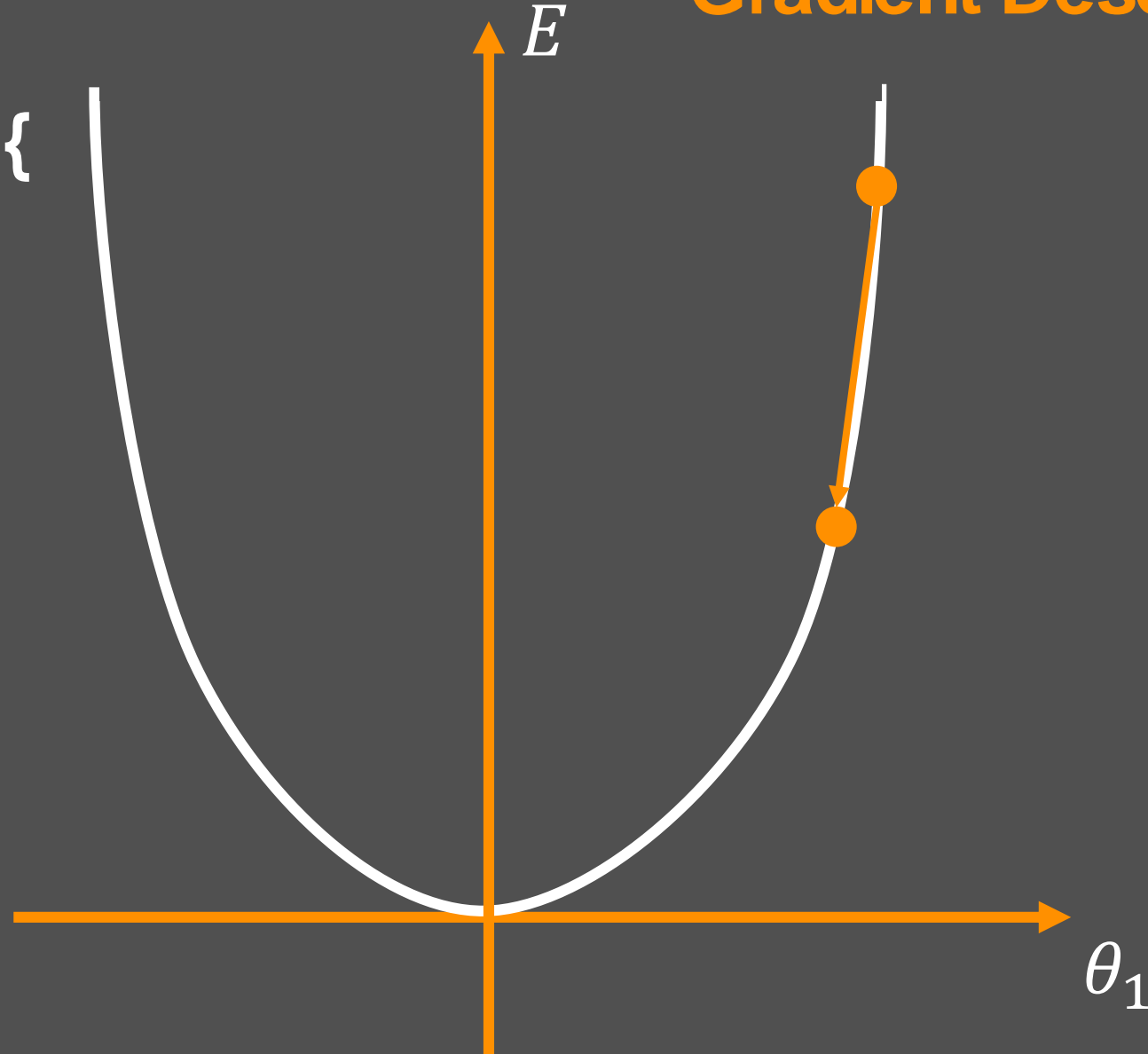
repeat until convergence{

$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_1)}{\partial \theta_1}$$

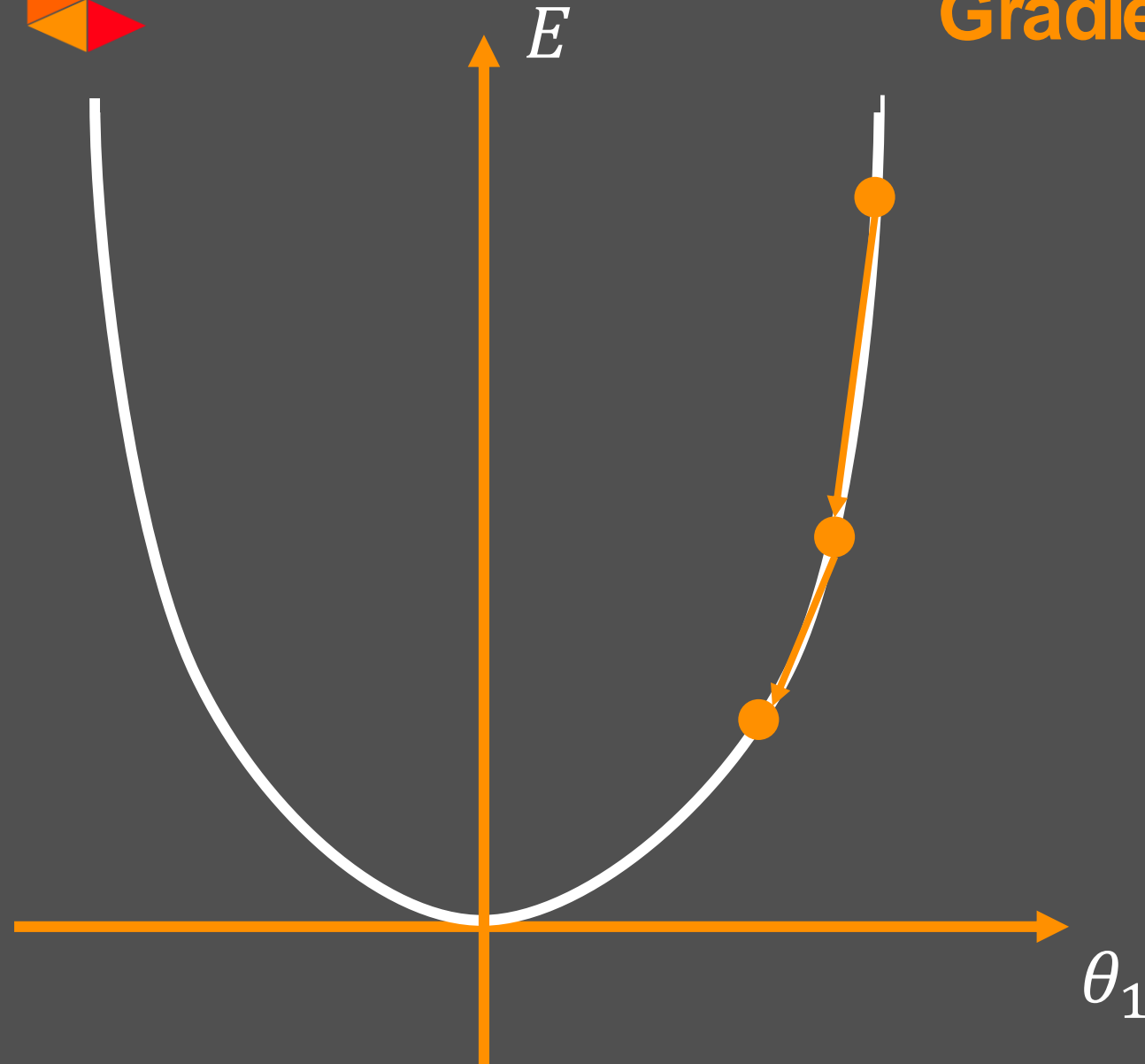
}

α : **Learning Rate**

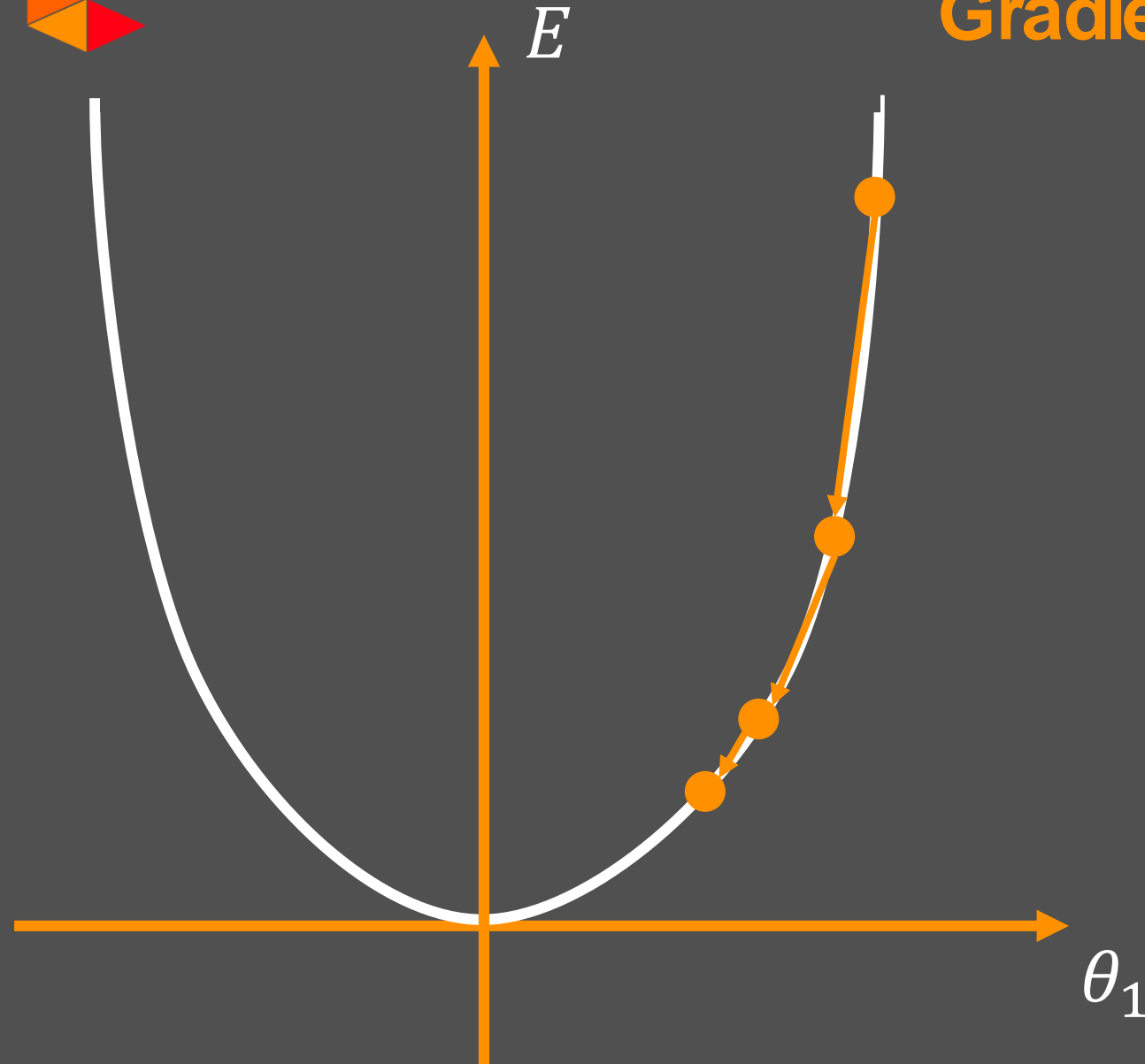
$\alpha = \text{const.}$



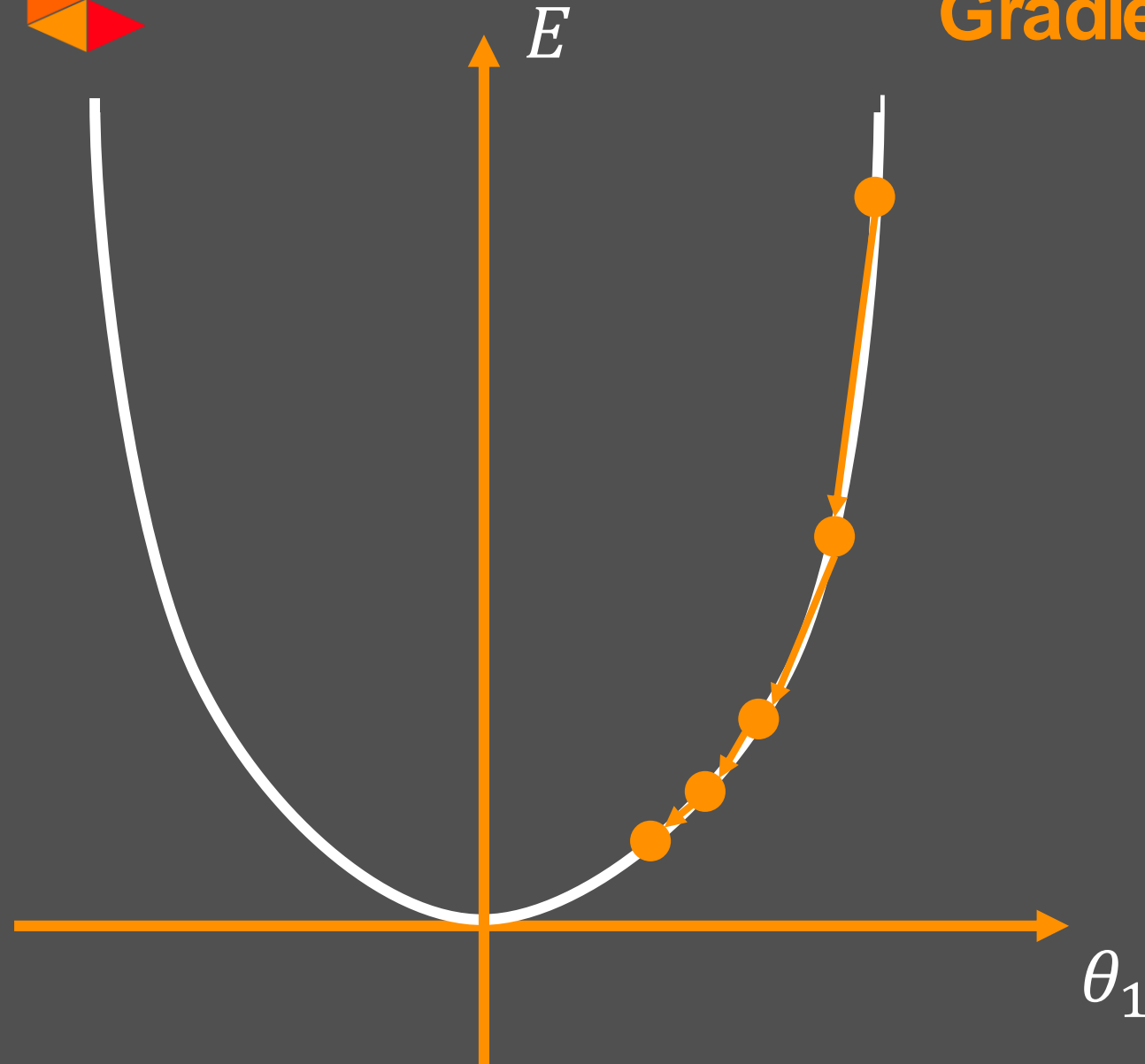
Gradient Descent



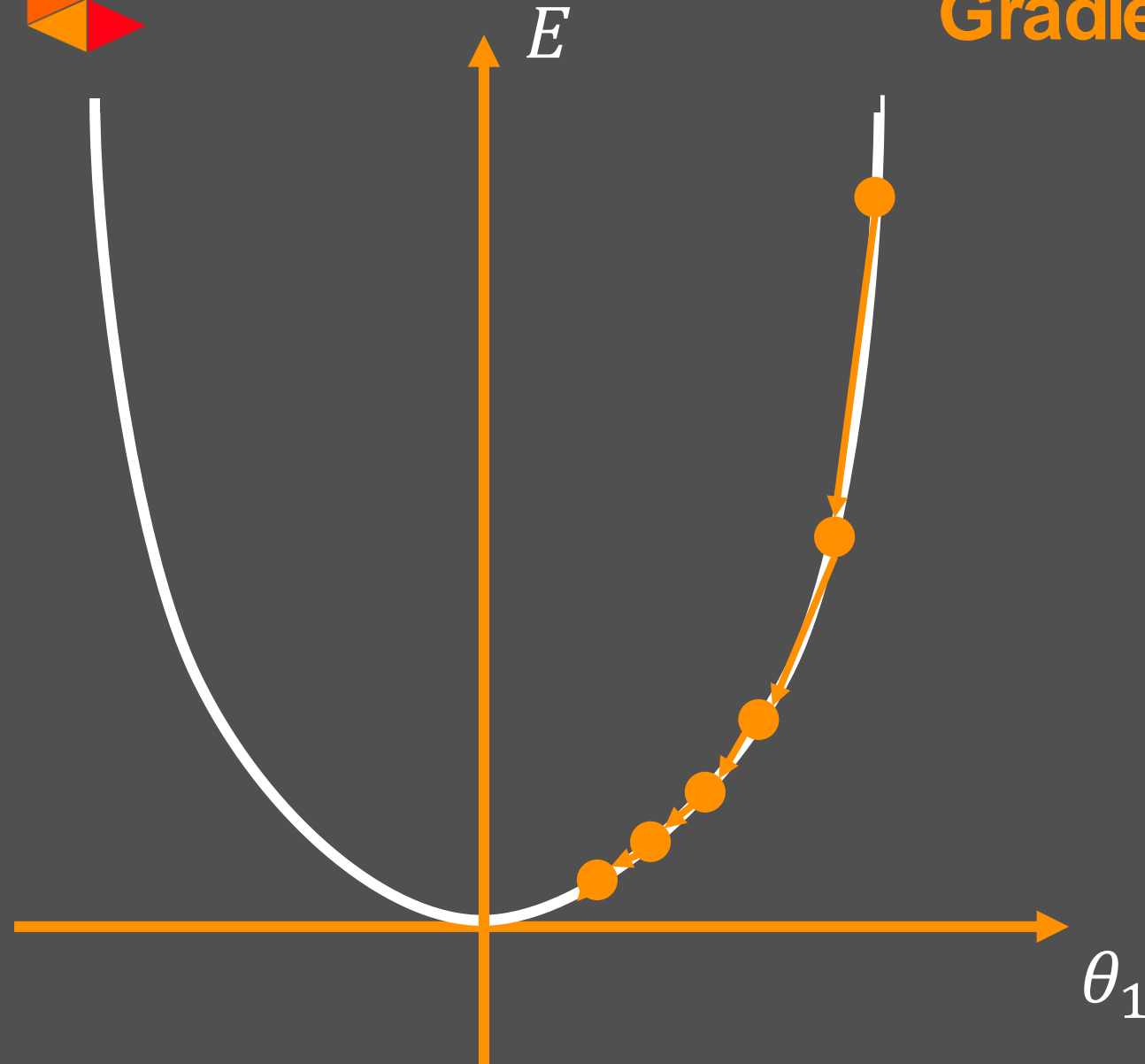
Gradient Descent



Gradient Descent

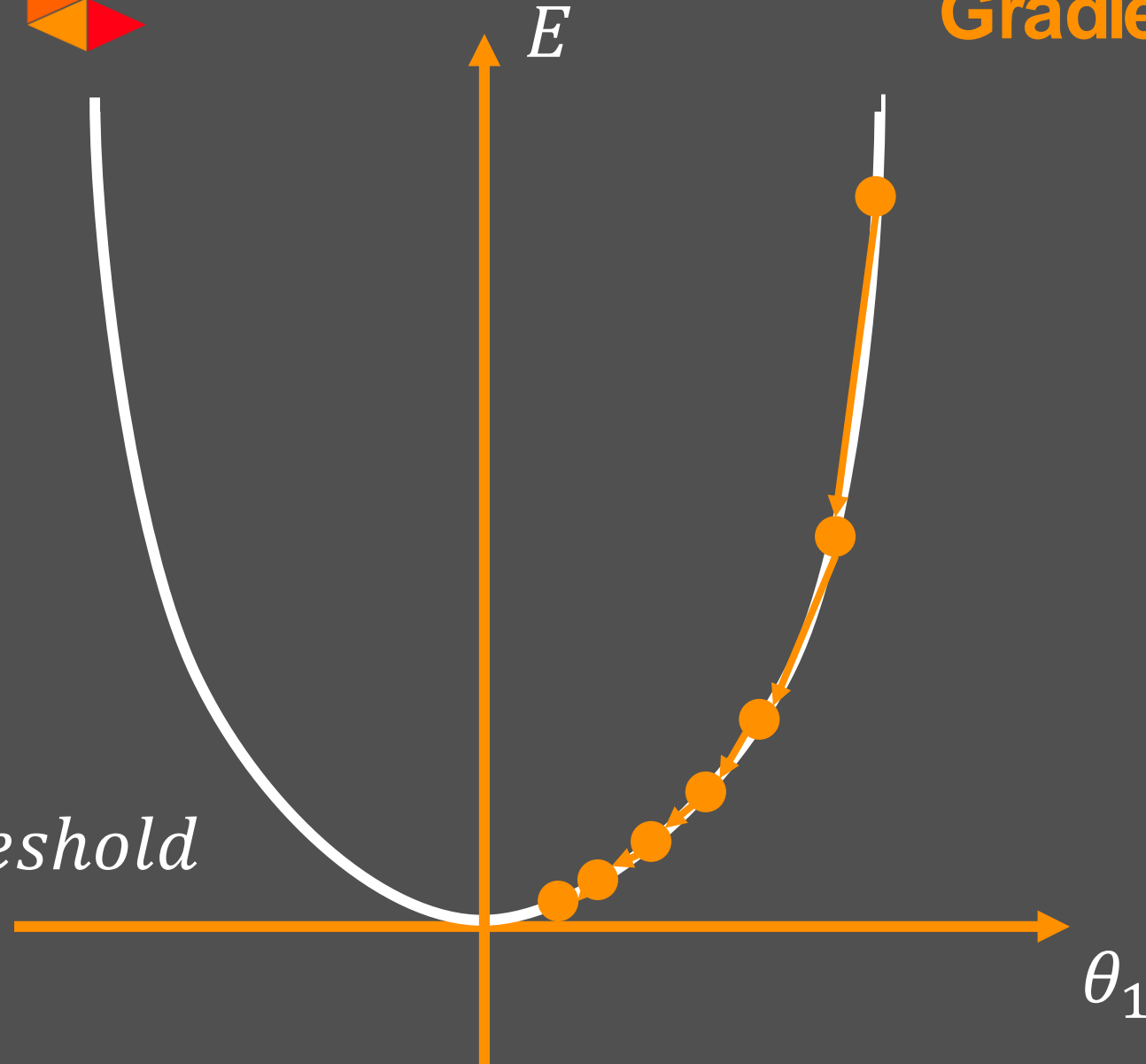


Gradient Descent

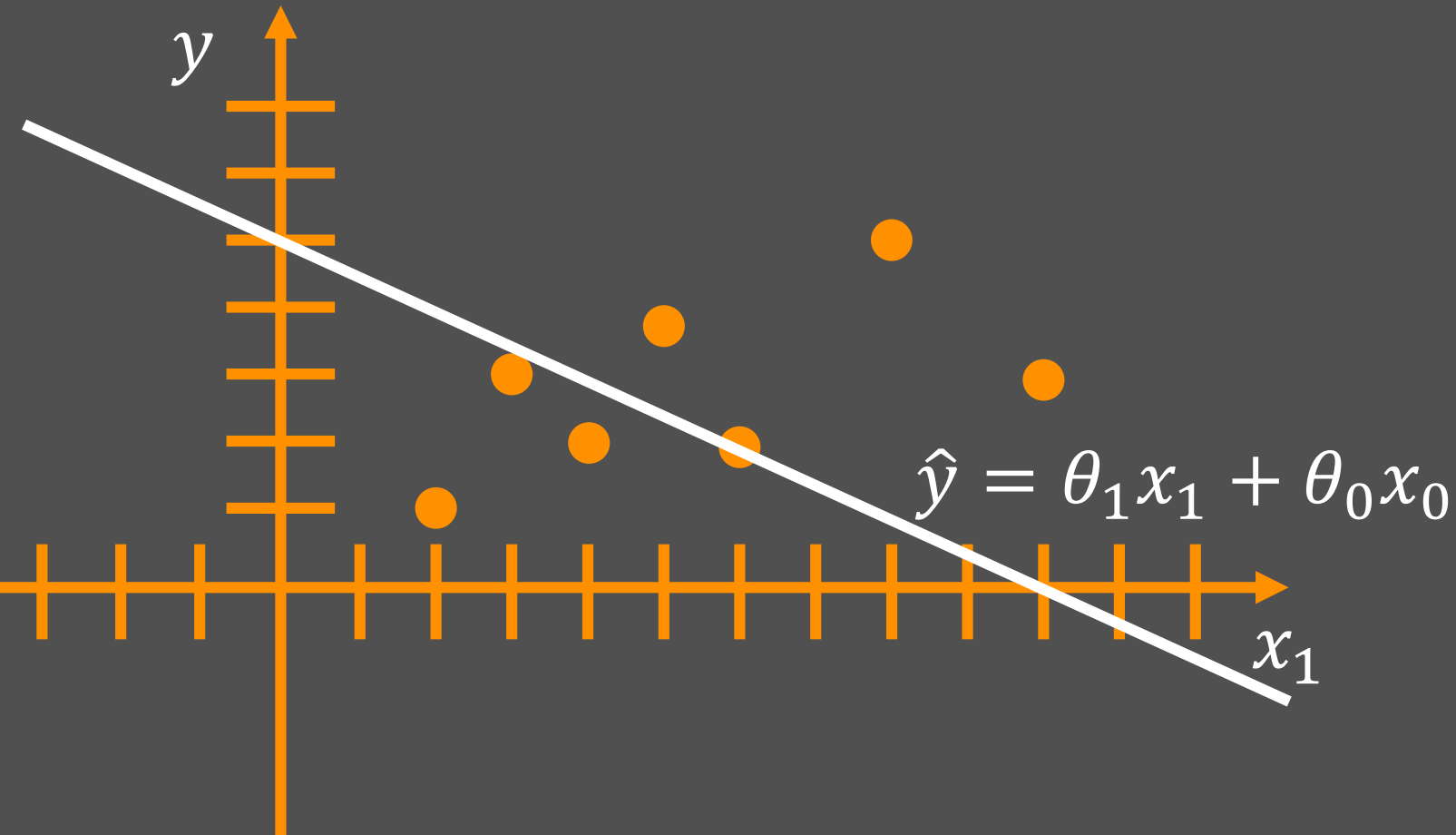


Gradient Descent

e.g.: $E < \text{threshold}$



Two weights:



$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^m ((\theta_1 x_{i,1} + \theta_0 x_{i,0}) - y_i)^2$$

Want: $\min_{\theta_0, \theta_1} E(\theta_0, \theta_1)$

$$\frac{\partial E(\theta_0, \theta_1)}{\partial \theta_1} = \frac{1}{m} \sum_{i=1}^m ((\theta_1 x_{i,1}) - y_i) * x_{i,1}$$

$$\frac{\partial E(\theta_0, \theta_1)}{\partial \theta_0} = \frac{1}{m} \sum_{i=1}^m ((\theta_0 x_{i,0}) - y_i) * x_{i,0}$$

$$= \frac{1}{m} \sum_{i=1}^m ((\theta_0 x_{i,0}) - y_i), x_{i,0} = 1$$

Gradient Descent

Start with random θ_0, θ_1

repeat until convergence{

$$\theta_0 := \theta_0 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_0}$$

$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_1}$$

}

Implementation note:

Start with random θ_0, θ_1

repeat until convergence{

$$tmp_0 := \theta_0 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_0}$$

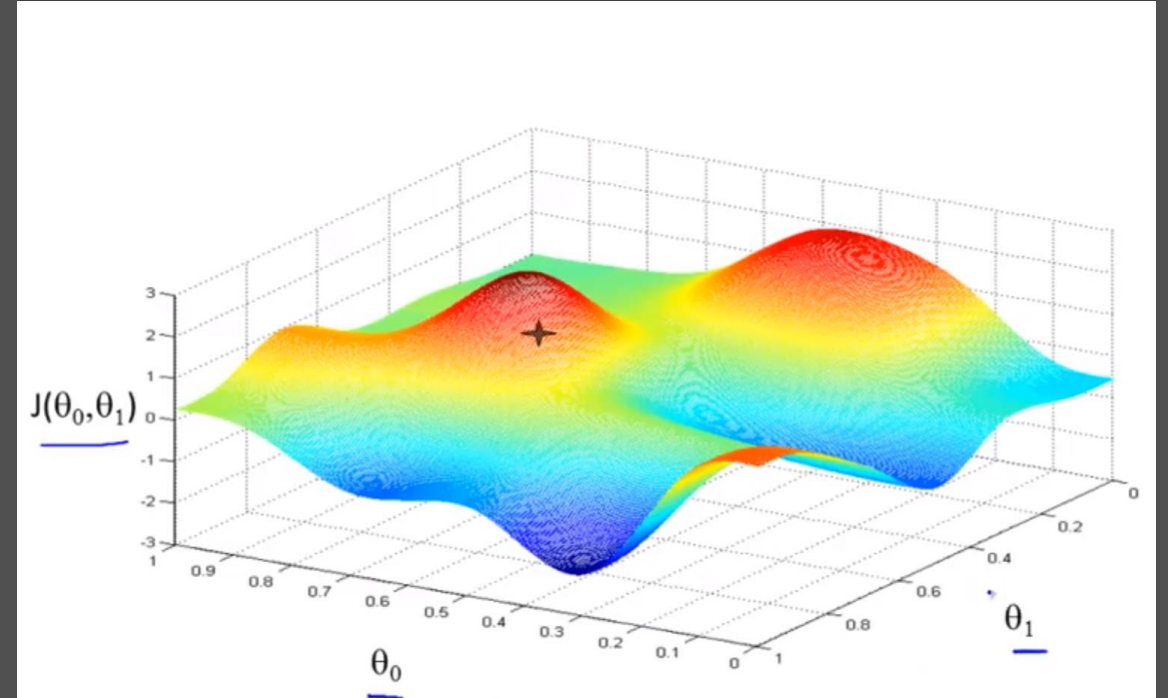
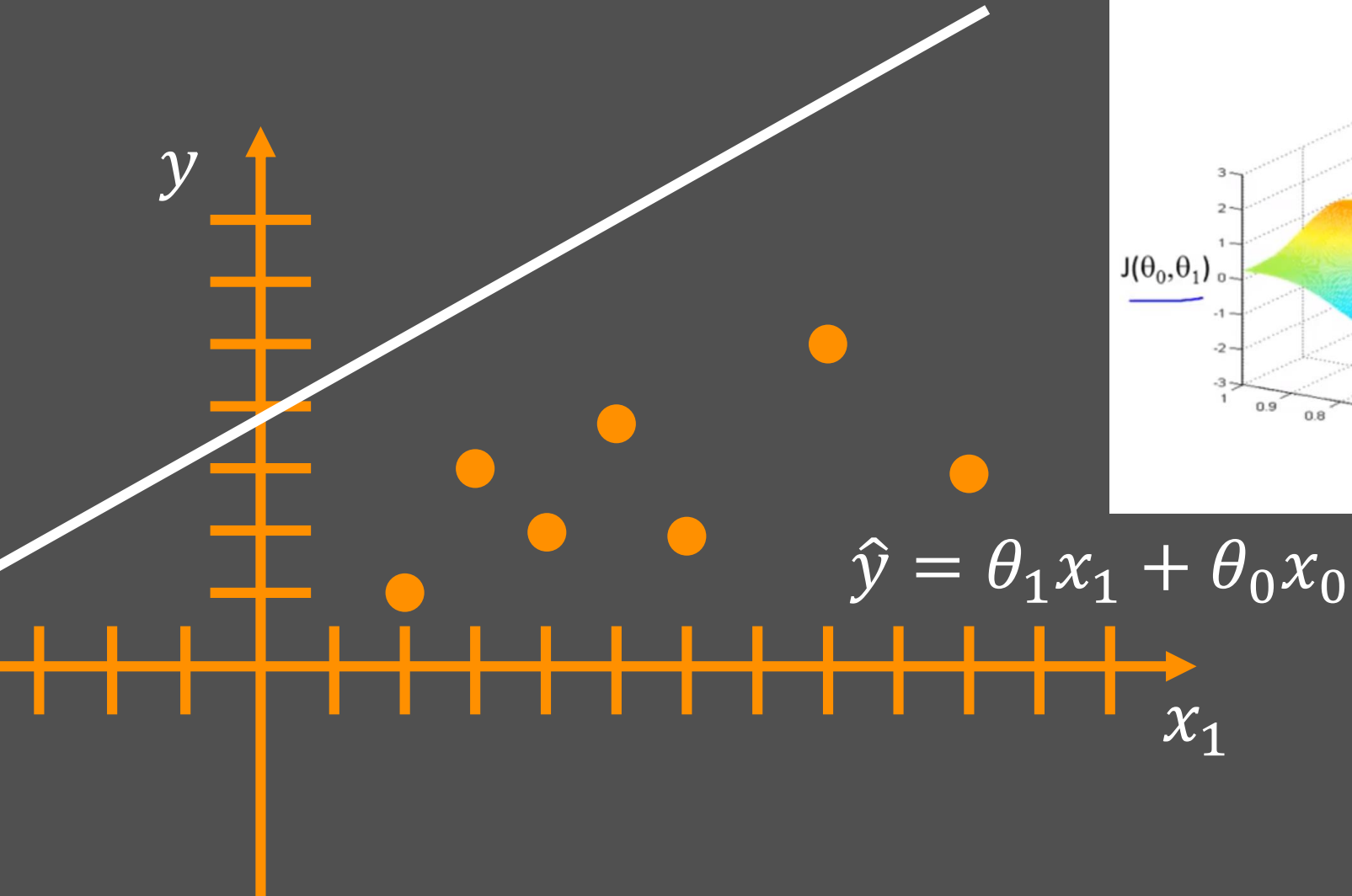
$$tmp_1 := \theta_1 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_1}$$

$$\theta_0 := tmp_0$$

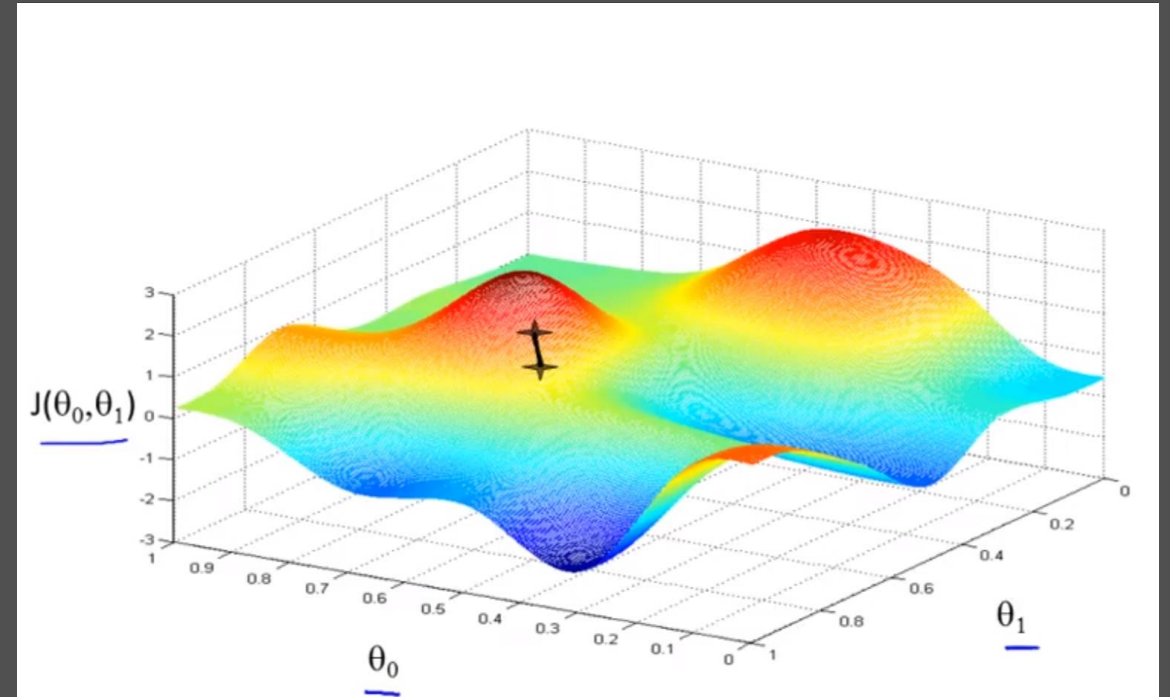
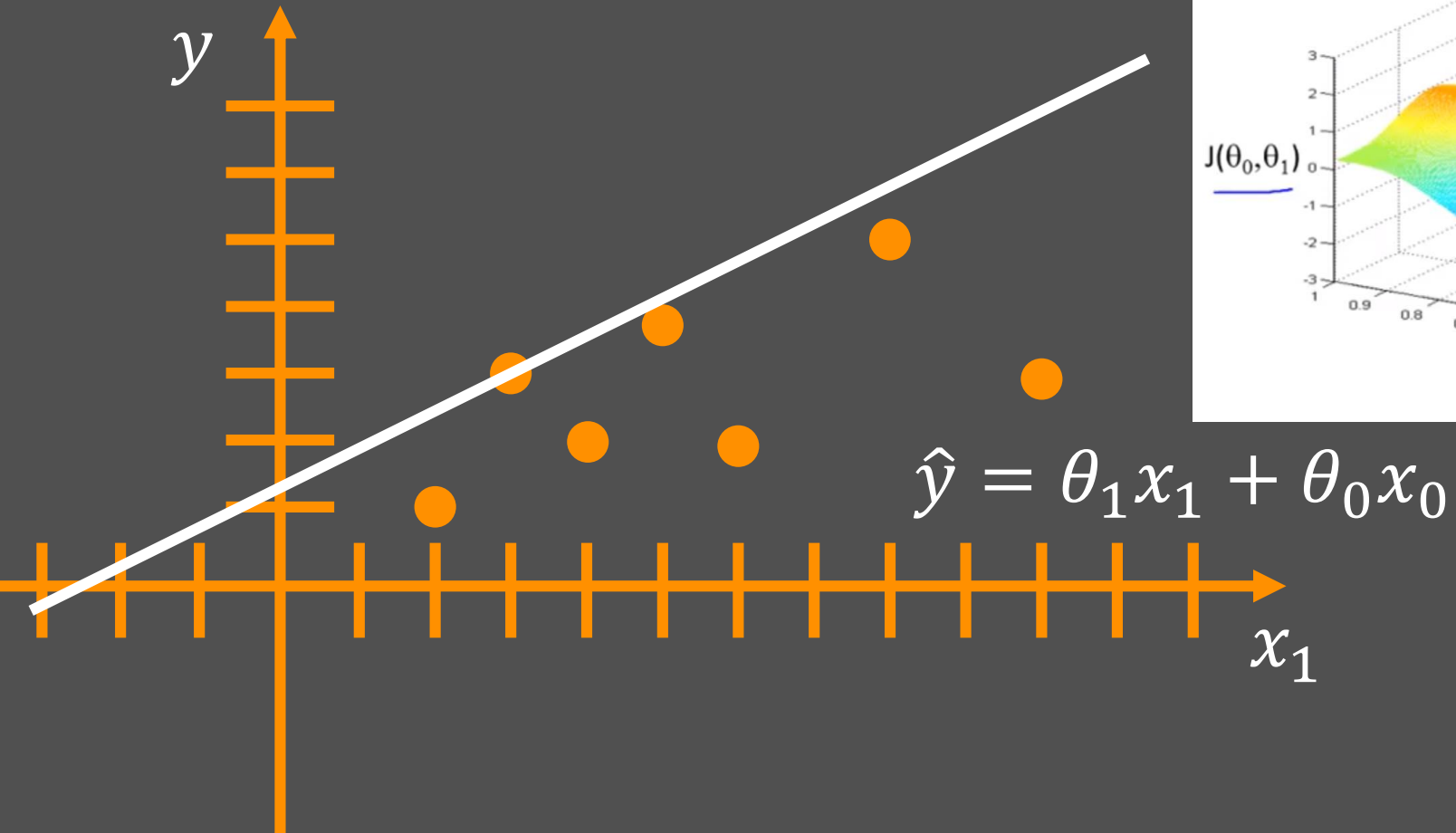
$$\theta_1 := tmp_1$$

}

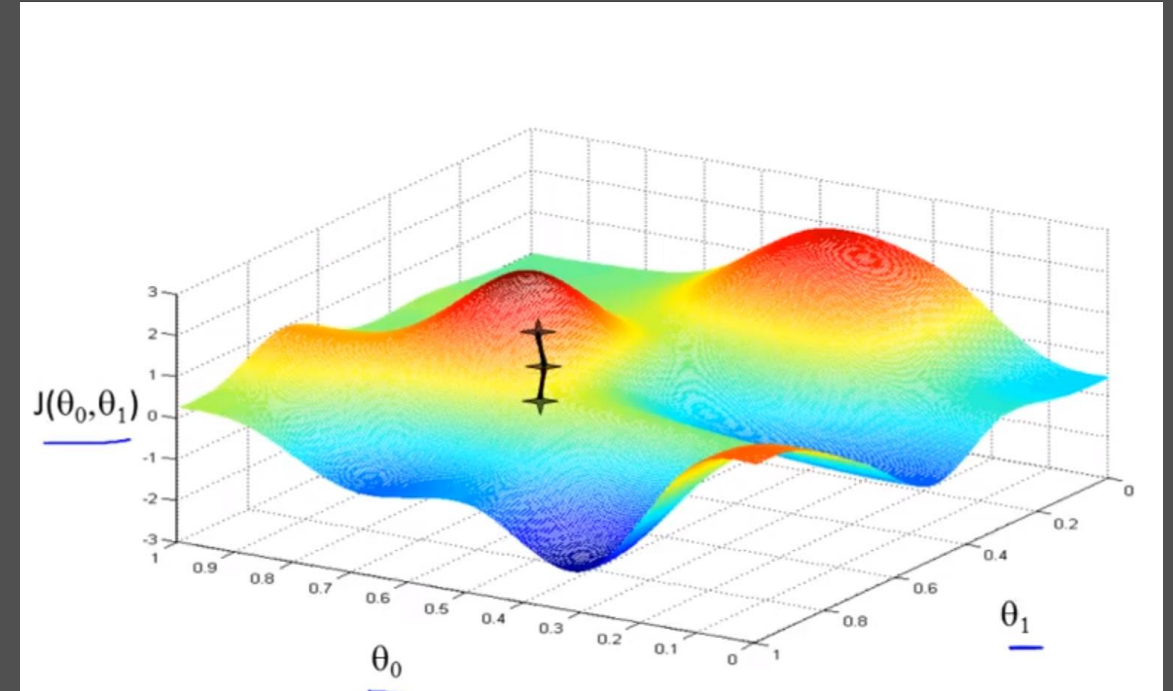
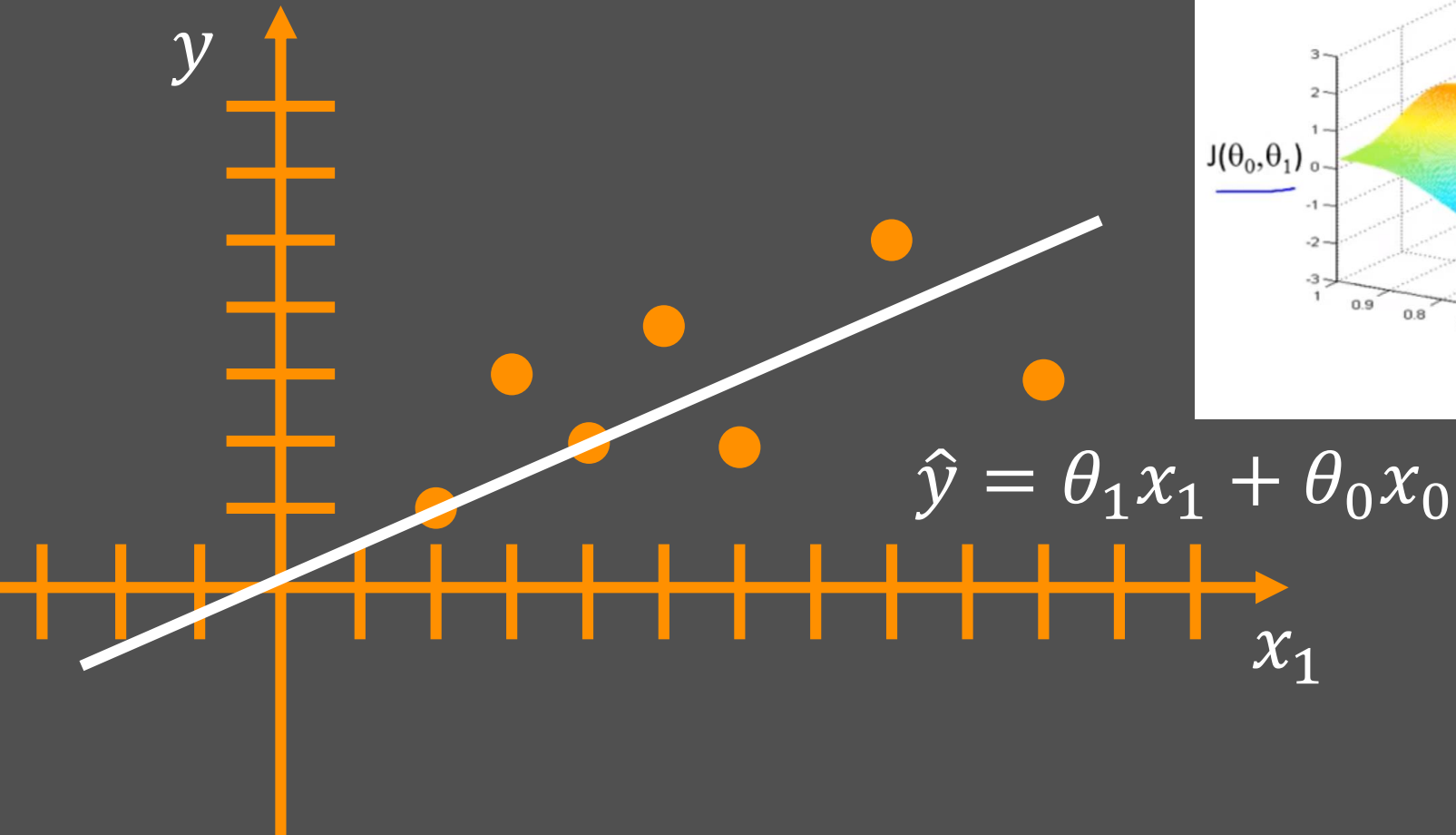
Gradient Descent



Gradient Descent

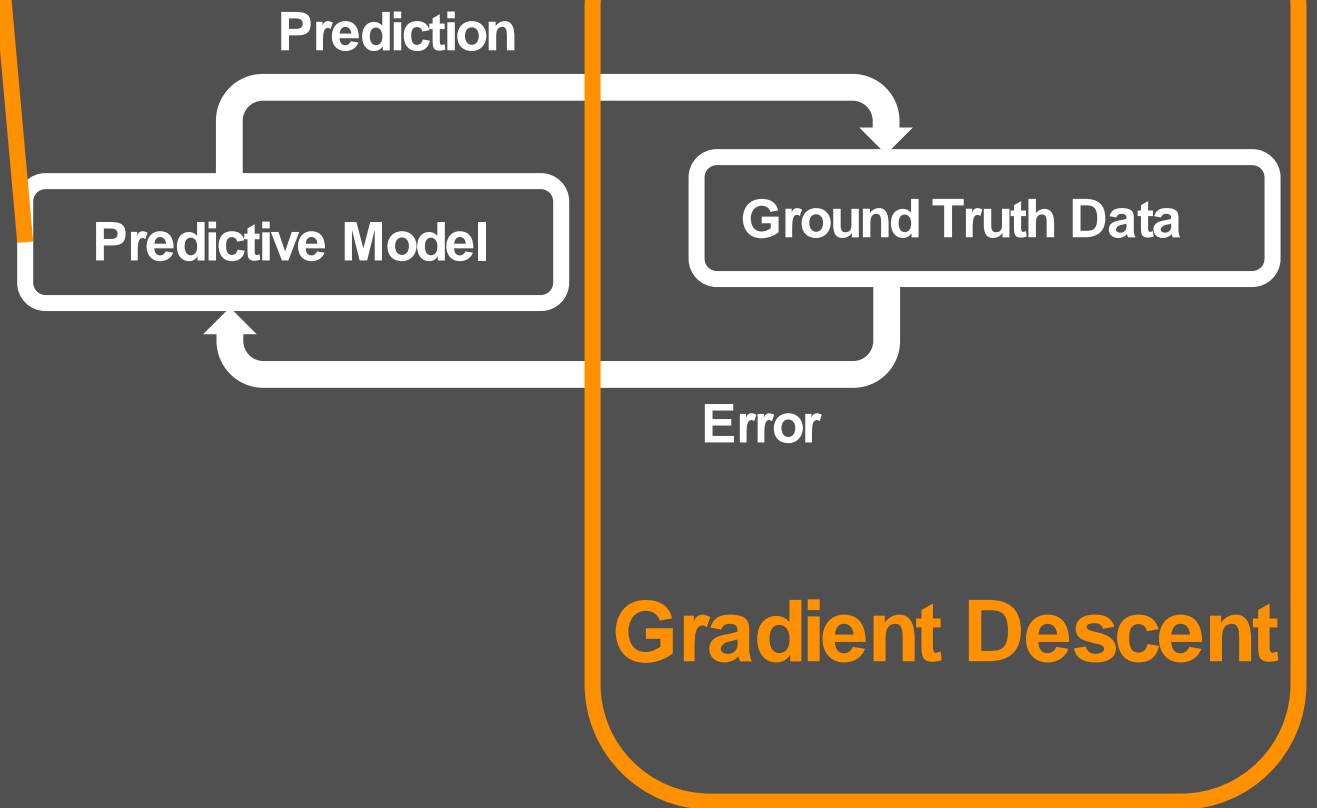
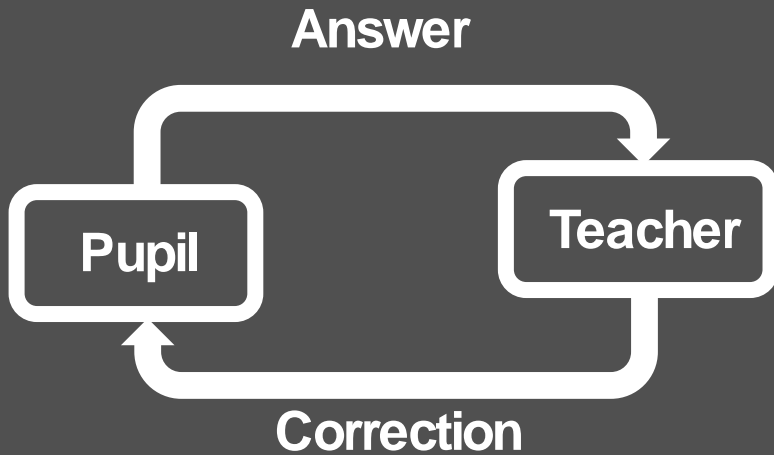


Gradient Descent



You learned:

Linear Regression



Questions?



TensorFlow

About:

- **open source software library for high performance numerical computation**
- **variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices**
- **used across many other scientific domains**



UBER



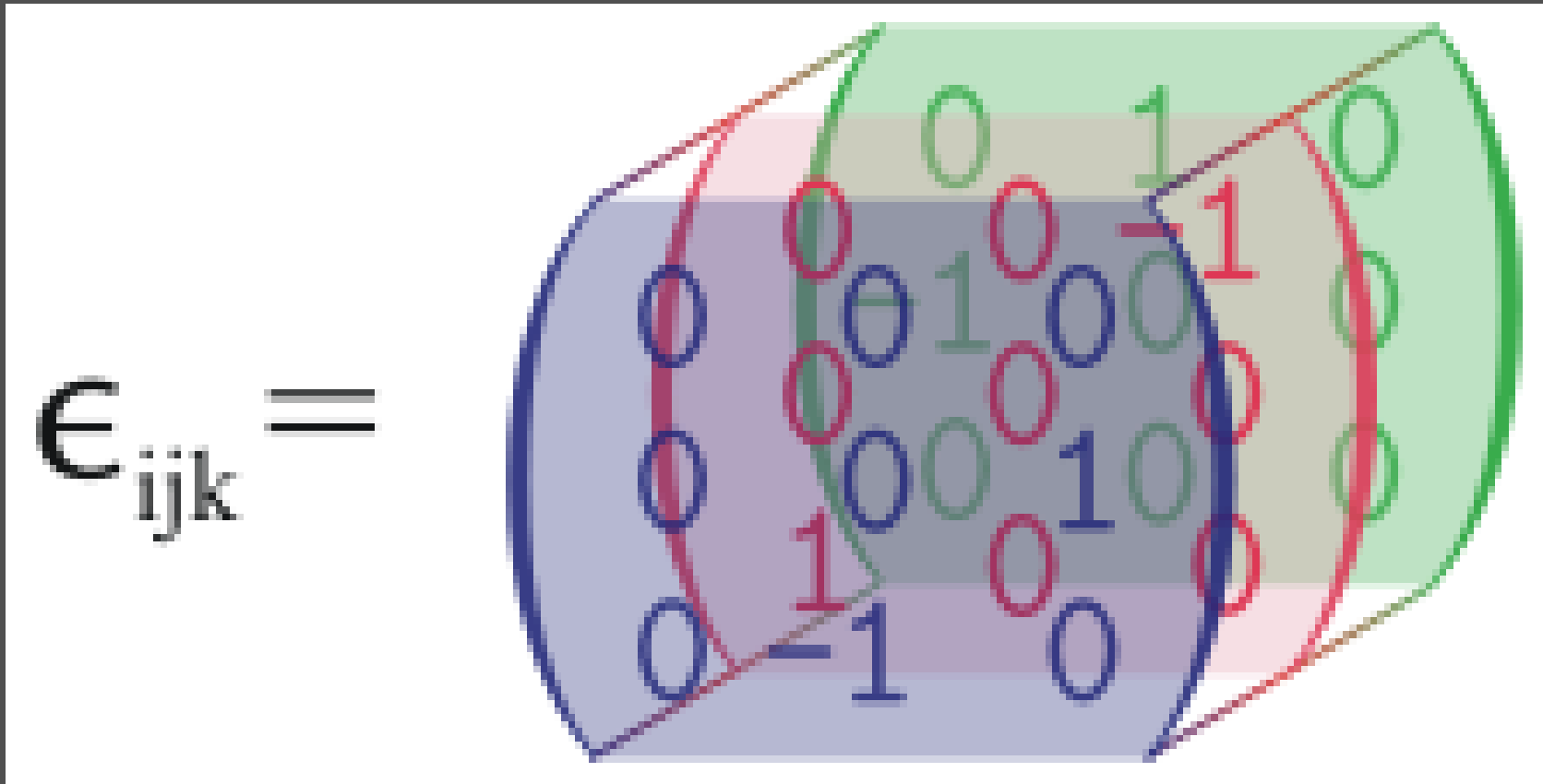
kakao



ARM

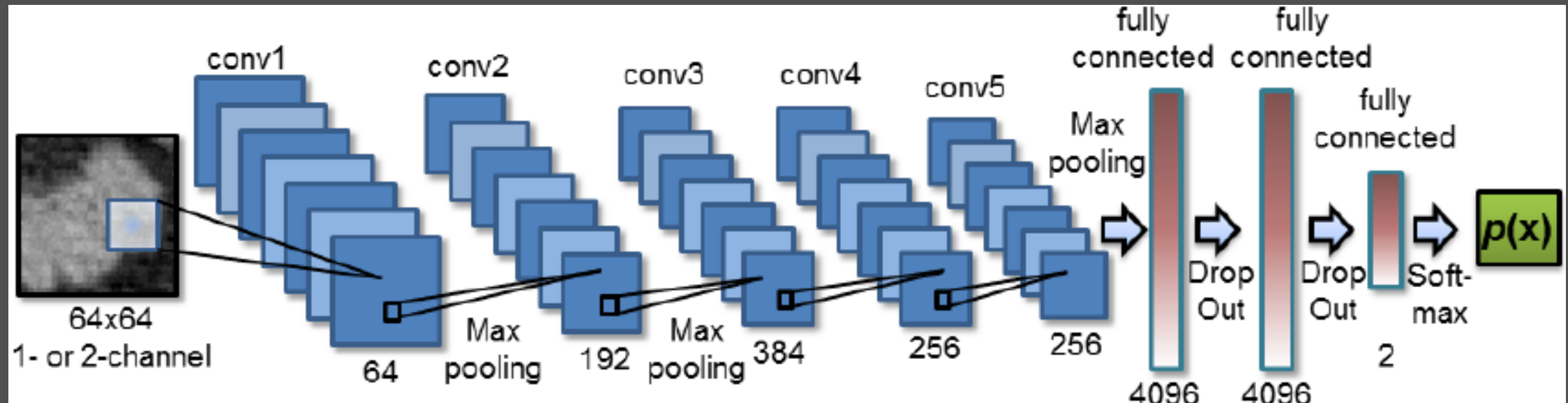


What is a Tensor? (compare with array of array of array ...)



Why is this useful?

- Try to describe multiple images in one mathematical structure

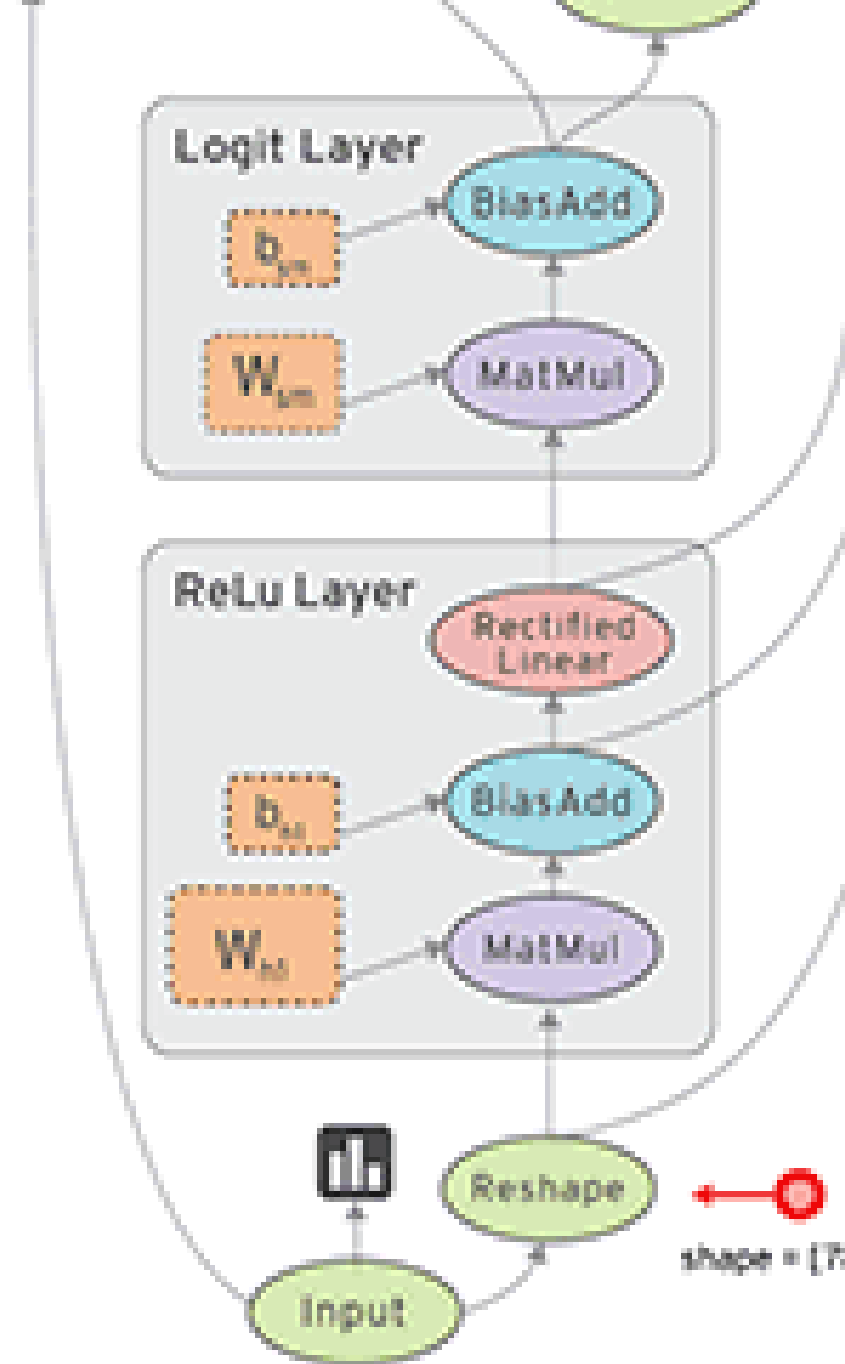


Basic elements:

- **tf.Variable:** remember θ
- **tf.constant**
- **tf.placeholder:** variable which can be used in a graph
- **tf.flags:** Global parameter, e.g. α : learning rate

Graph:

- Blueprint of computations
- Chain of functions



tf.placeholder:

- **Useful for input data**

tf.Variable:

- **Values are manipulated while training**

Basic operations:

- **`tf.multiply(x, y)`**: $x * y$ element-wise
- **`tf.reduce_sum(x)`**: Computes the sum of elements across dimensions of a tensor
- **`tf.losses.mean_squared_error(y, \hat{y})`**: $\frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$
- **`tf.add(x, y)`**: $x + y$
- **`tf.scalar_mul(α, x)`**: $\alpha * x$

Basic operations:

- **`tf.transpose(x)`**: x^T
- **`tf.subtract(x, y)`**: $x - y$
- **`tf.reshape(X, shape)`**: resizes X to new shape

Task today:

- Implement linear regression with gradient descent
- Execute „**lr_train_gd.py**“
- Implement functions in „**lr_model_gd.py**“
- Some helper functions „**split_dataset.py**“

„**lr_train_gd.py**“:

- Load training data
- Call train function of model
- Load test data
- Test model

„lr_model_gd.py“:

- **inference: prediction** $\hat{y} = \theta_1 x_1 + \theta_0 x_0$
- **loss: computation of** $E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$
- **gradient_descent: Computation of new θ**
- **train: One training step**

Hints:

- n : **#Columns = #Features**
- $\hat{y} = \theta_n x_n + \theta_{n-1} x_{n-1} + \dots + \theta_1 x_1 + \theta_0 x_0$
- $\hat{y} = \theta x$
- Use **print!** For instance: print shape of tensor (tf.shape)

<https://github.com/mati3230/modalg181>