Técnicas Avanzadas de Data Mining y Sistemas Inteligentes

Maestría en Informática
Escuela de Posgrado
Pontificia Universidad Católica del Perú

2018-2

Renato Hermoza - renato.hermoza@pucp.edu.pe

Machine Learning

Traditional Programming

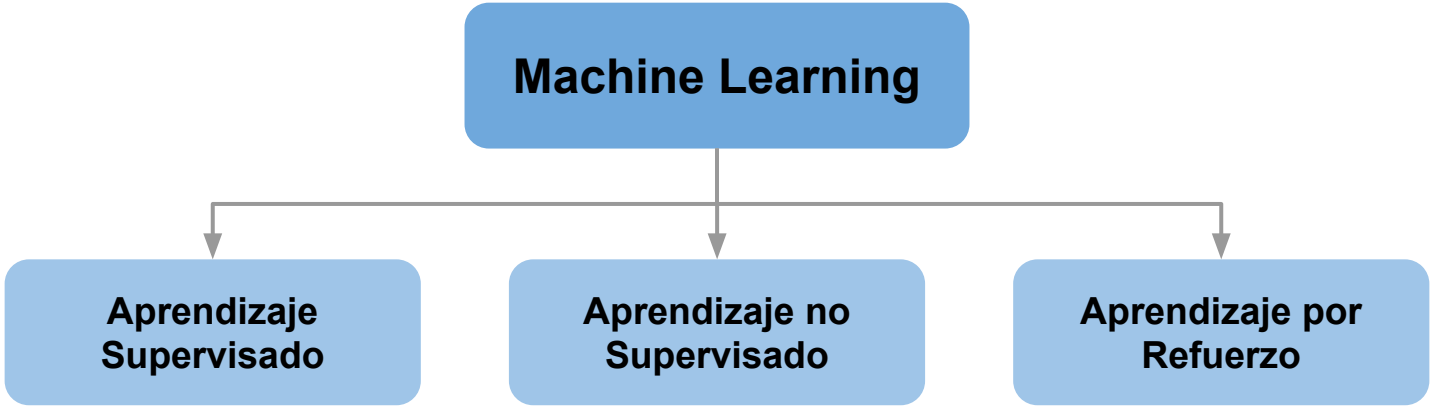
Traditional Programming

Machine Learning

Traditional Programming

Machine Learning

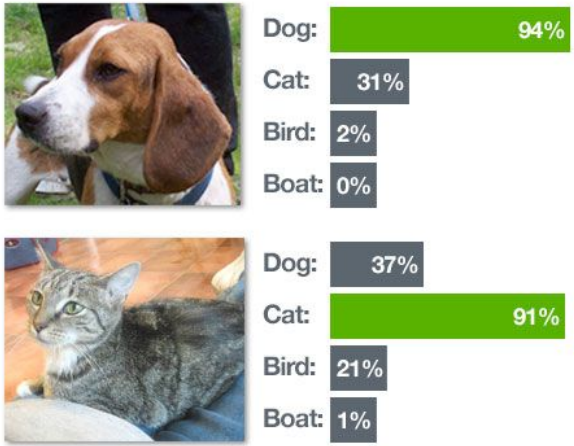
Los algoritmos de machine learning usan métodos computacionales para aprender directamente de la data, sin estar explícitamente programados.

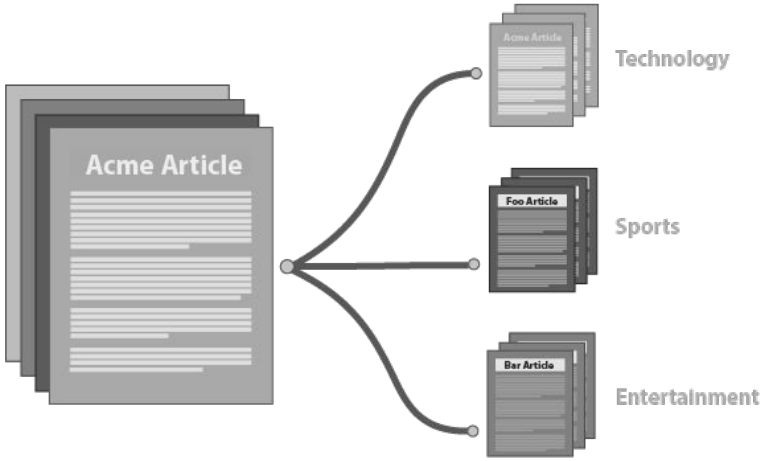

Machine Learning
Aprendizaje Supervisado
Aprendizaje no Supervisado
Aprendizaje por Refuerzo

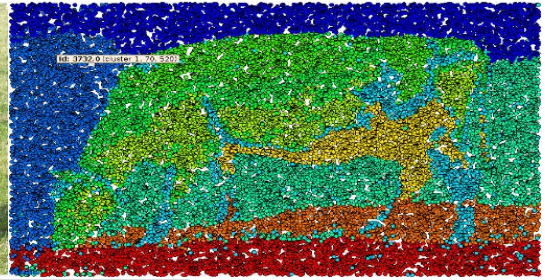
Machine

Aprendizaje Supervisado

Aprend
Super



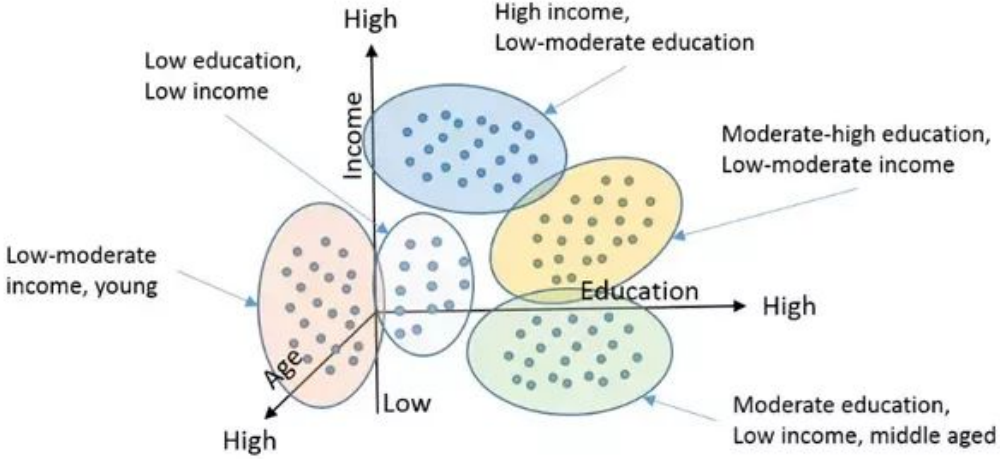


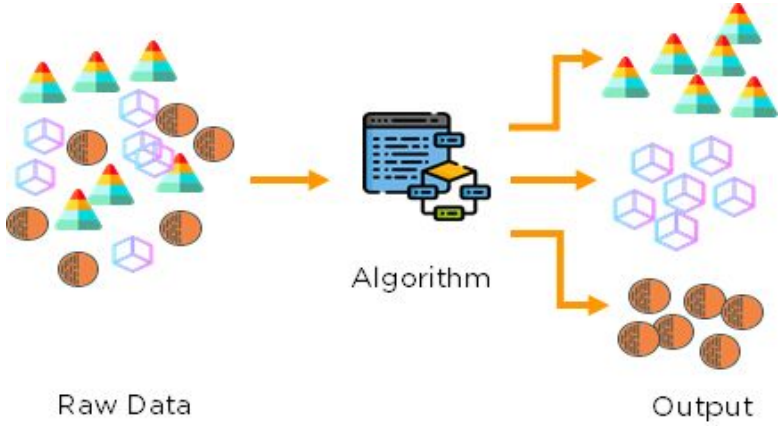


Aprendizaje Supervisado

Aprendizaje no Supervisado

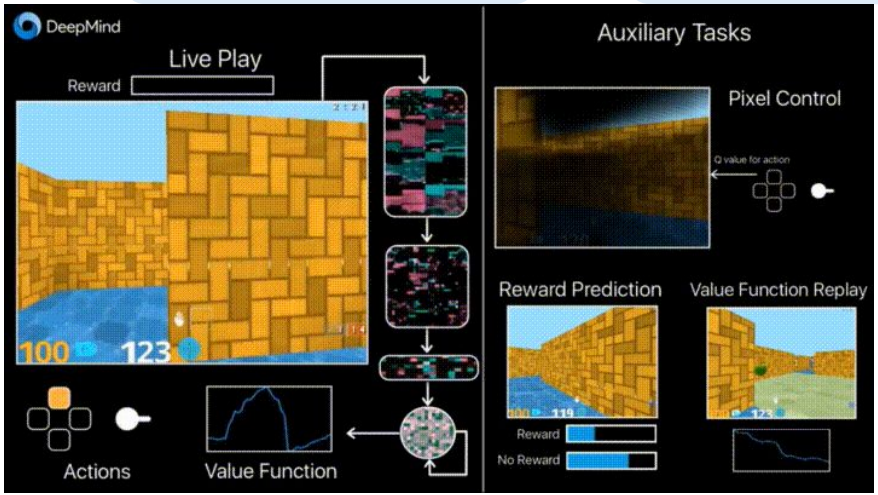
Aprendizaje por Refuerzo



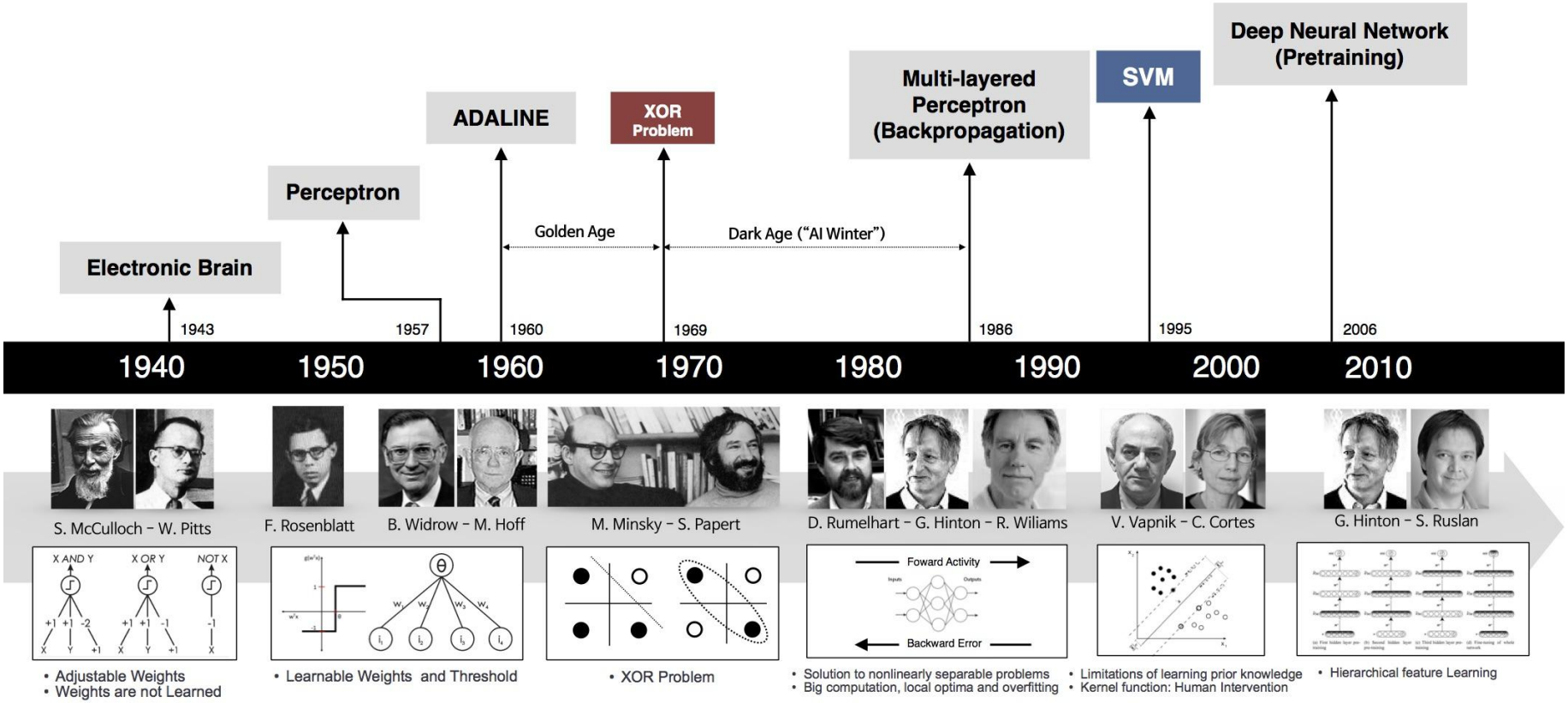


Machine Learning

- Aprendizaje Supervisado
- Aprendizaje no Supervisado
- **Aprendizaje por Refuerzo**



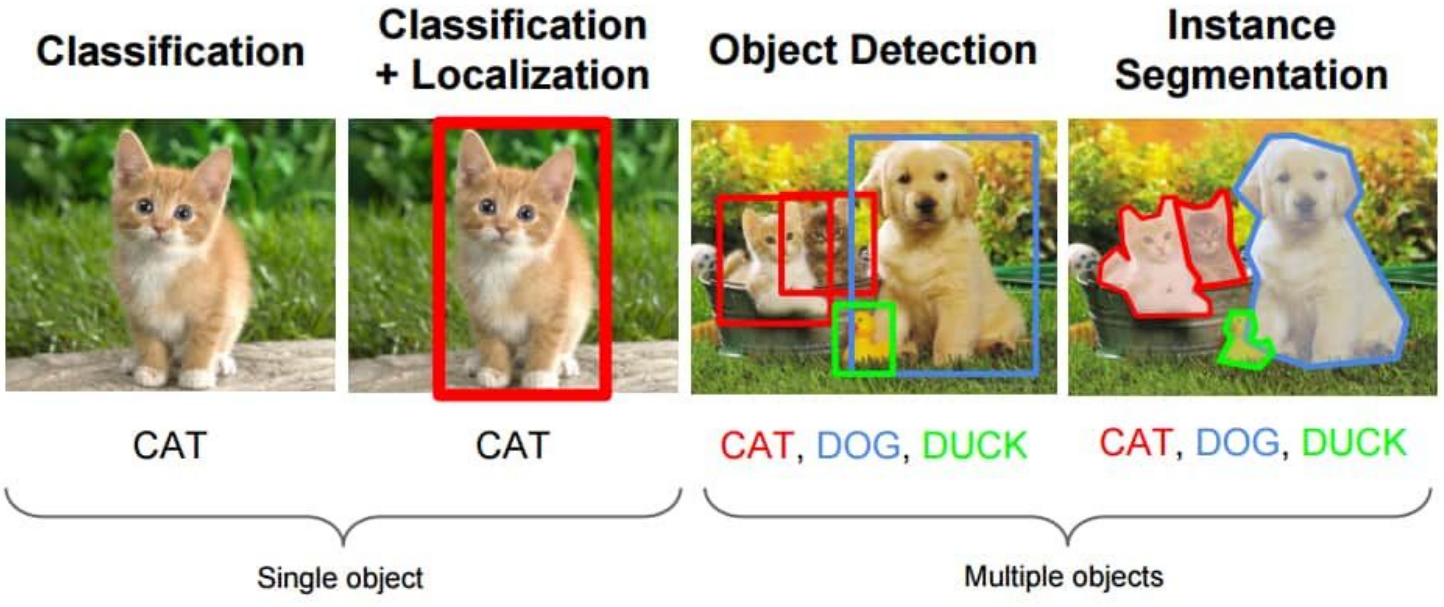
Deep Learning


Deep Neural Network (Pretraining)
SVM
Multi-layered Perceptron (Backpropagation)
XOR Problem
ADALINE
Perceptron
Electronic Brain
Golden Age
Dark Age ("AI Winter")
1943
1957
1960
1969
1986
1995
2006
1940
1950
1960
1970
1980
1990
2000
2010
S. McCulloch - W. Pitts
F. Rosenblatt
B. Widrow - M. Hoff
M. Minsky - S. Papert
D. Rumelhart - G. Hinton - R. Wiliams
V. Vapnik - C. Cortes
G. Hinton - S. Ruslan
X AND Y
X OR Y
NOT X
Foward Activity
Backward Error
• Adjustable Weights
• Weights are not Learned
• Learnable Weights and Threshold
• XOR Problem
• Solution to nonlinearly separable problems
• Big computation, local optima and overfitting
• Limitations of learning prior knowledge
• Kernel function: Human Intervention
• Hierarchical feature Learning

¿Qué problemas se están resolviendo con DL?

- Visión computacional
  - Reconocimiento de imágenes
  - Segmentación de objetos
  - Reconstrucción de imágenes

Source: Fei-Fei Li, Andrej Karpathy & Justin Johnson (2016) cs231n, Lecture 8 - Slide 8, Spatial Localization and Detection (01/02/2016)



¿Qué problemas se están resolviendo con DL?

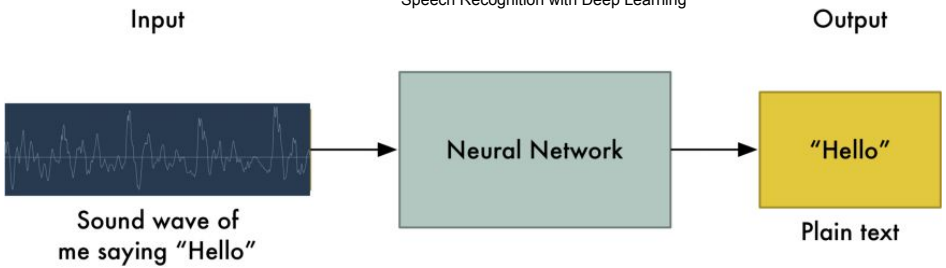
- Lenguaje natural (NLP)
  - Traducción
  - Parsing
  - Generación de conversaciones (chatbots)
  - Generación de resúmenes

Sorce: Data Science Lab. Available: http://datascience.jd.com/?page_id=251

¿Qué problemas se están resolviendo con DL?

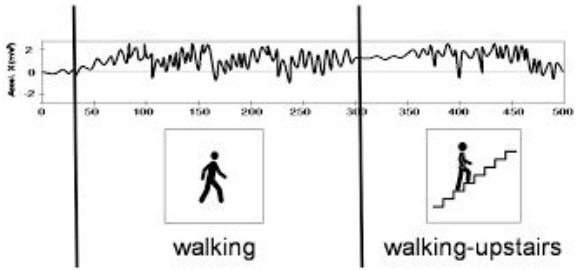
- Procesamiento de señales o datos en secuencia
  - Reconocimiento de acciones (sensores)
  - Reconocimiento de voz

Source: Adam Geitgey. Machine Learning is Fun Part 6: How to do Speech Recognition with Deep Learning



- Tareas combinadas
  - Generación de descripciones de imágenes (Image captioning)
  - Reconocimiento de videos

Source. Ran Xu, Priyanshu Agarwal, Suren Kumar, Venkat N. Krovi, and Jason J. Corso Combining Skeletal Pose with Local Motion for Human Activity Recognition



Source: Rubén San-Segundo, Juan M. Montero, José Moreno-Pimentel, José M. Pardo. HMM Adaptation for Improving a Human Activity Recognition System

Image Captioning

Describes without errors	Describes with minor errors	Somewhat related to the image	Unrelated to the image
 A person riding a motorcycle on a dirt road.	 Two dogs play in the grass.	 A skateboarder does a trick on a ramp.	 A dog is jumping to catch a frisbee.
 A group of young people playing a game of frisbee.	 Two hockey players are fighting over the puck.	 A little girl in a pink hat is blowing bubbles.	 A refrigerator filled with lots of food and drinks.
 A herd of elephants walking across a dry grass field.	 A close up of a cat laying on a couch.	 A red motorcycle parked on the side of the road.	 A yellow school bus parked in a parking lot.

[Vinyals et al., 2015]

A flower with thin pink petals resting around a cluster of white and yellow stamen.

This flower is white and yellow in colour, with petals that are multi coloured.

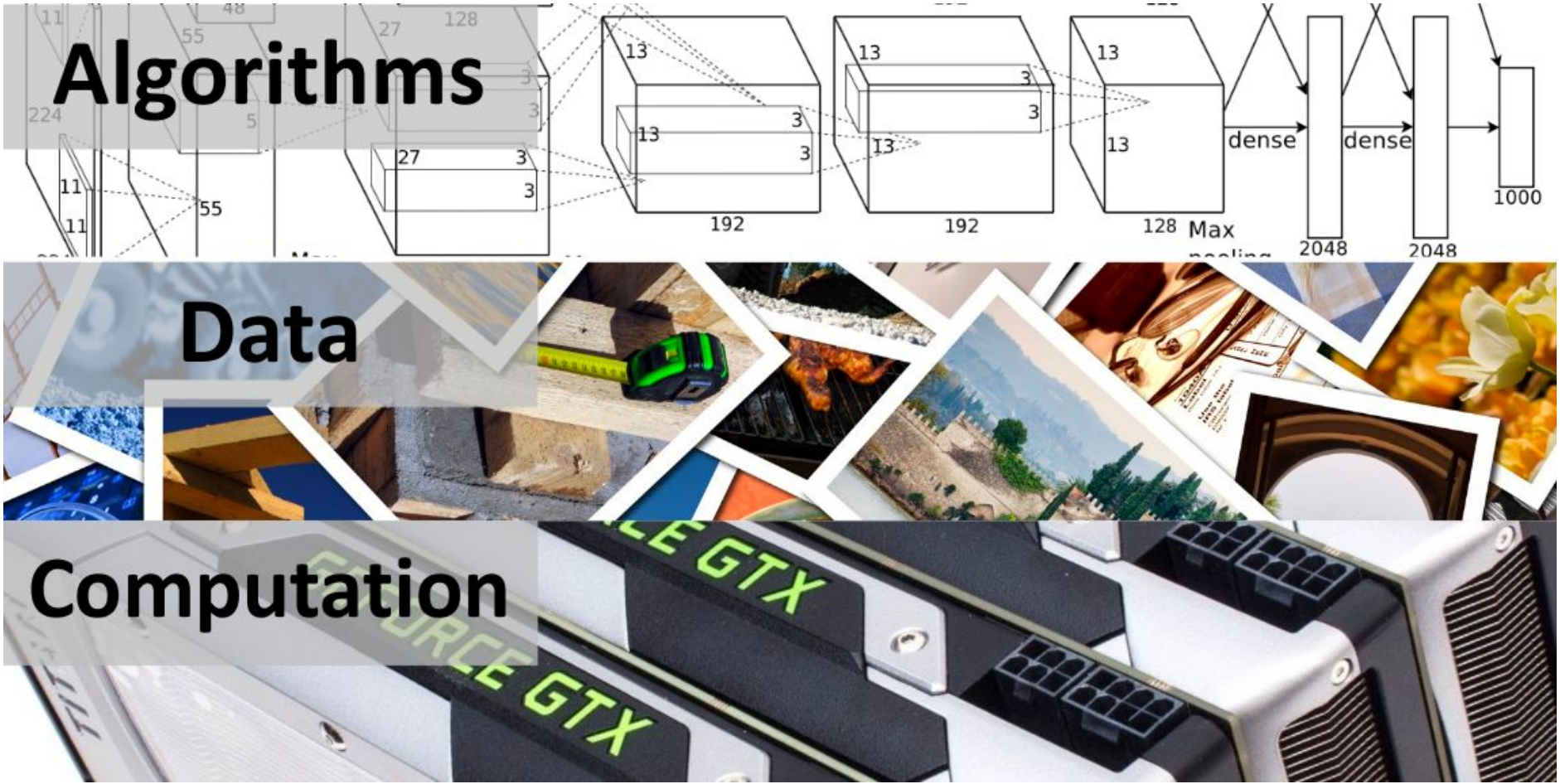
A fat round bird with a bright yellow breast and blue on top of the head.

The tan bird has a short beak, with stone black eyes.

Text to Image Synthesis

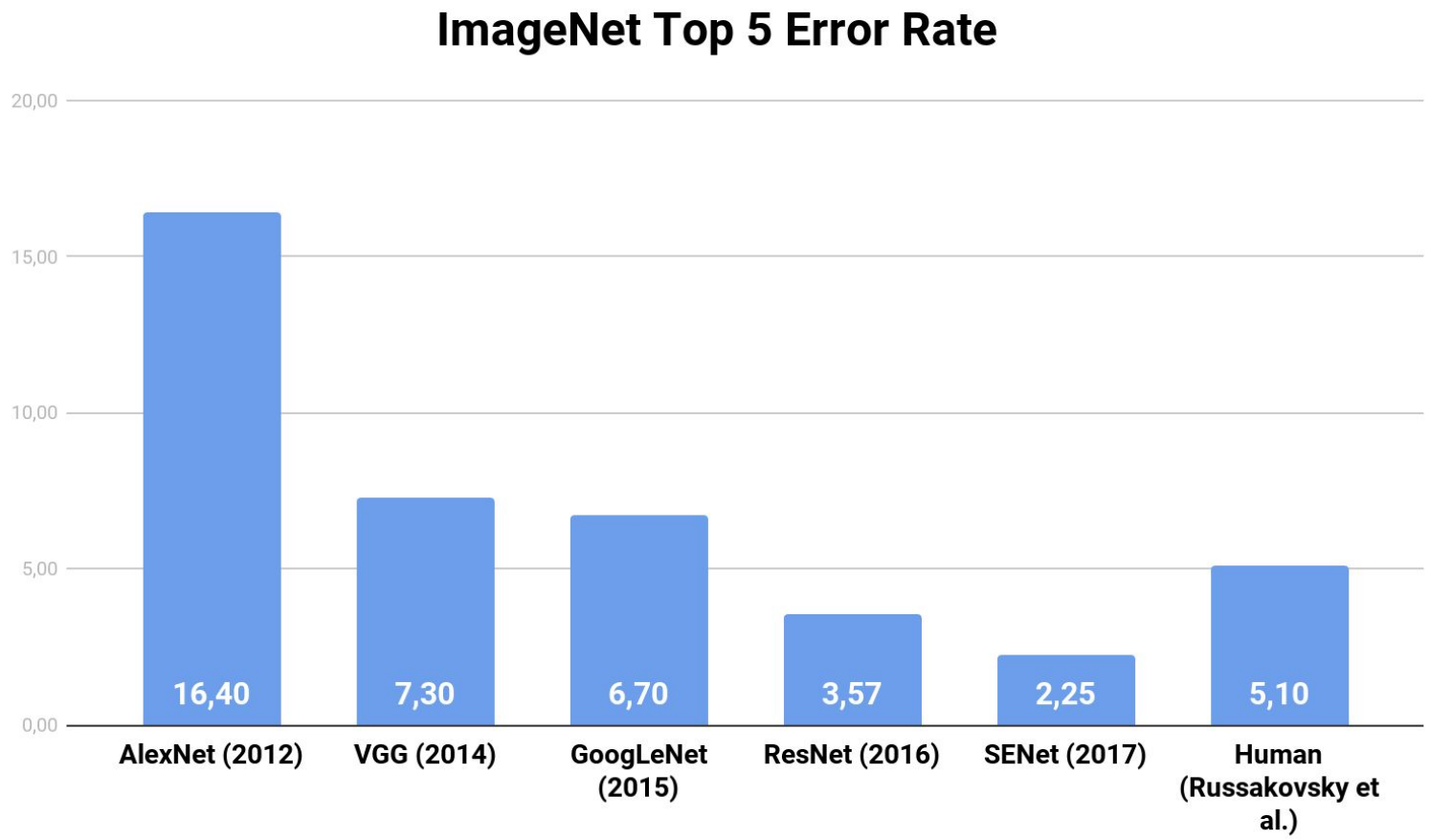
[Reed et al., 2016]

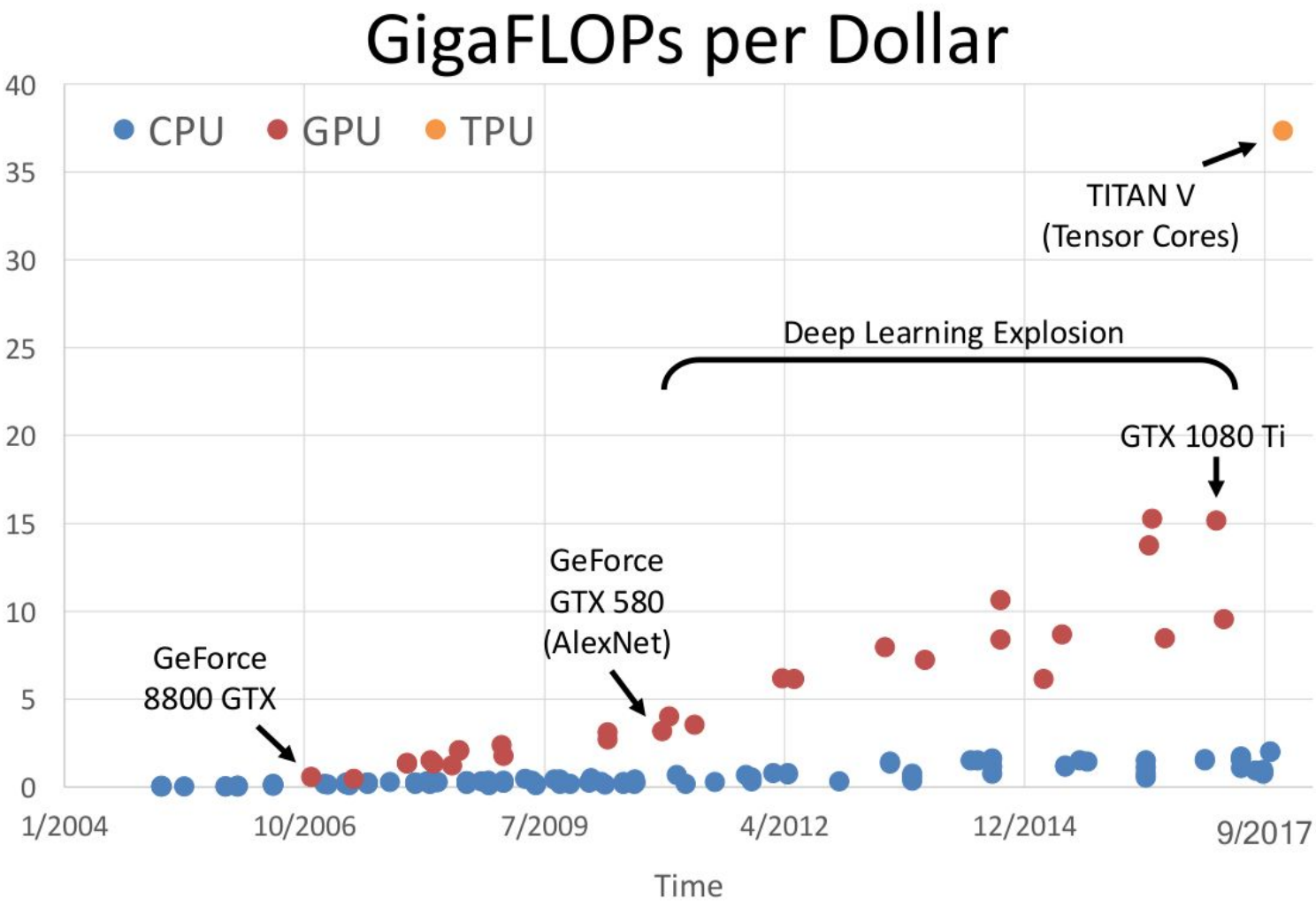
Deep Learning Ingredientes


Algorithms
Data
Computation
11
55
27
128
224
5
3
13
192
dense
dense
1000
2048
Max

IMAGENET

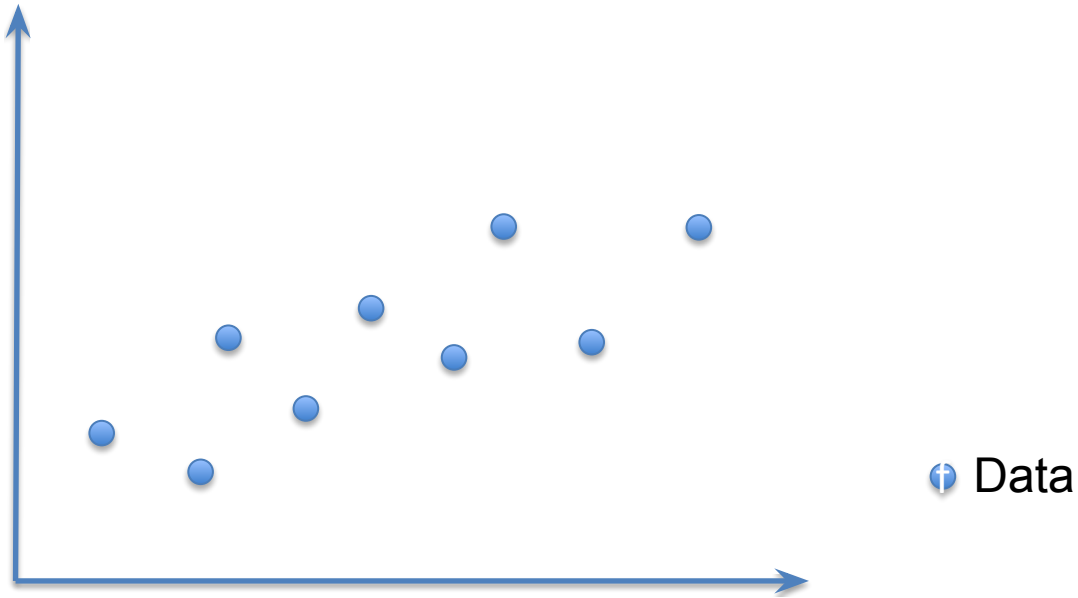
- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.


ImageNet Top 5 Error Rate
20,00
15,00
10,00
5,00
0,00
16,40
7,30
6,70
3,57
2,25
5,10
AlexNet (2012)
VGG (2014)
GoogLeNet (2015)
ResNet (2016)
SENet (2017)
Human (Russakovsky et al.)

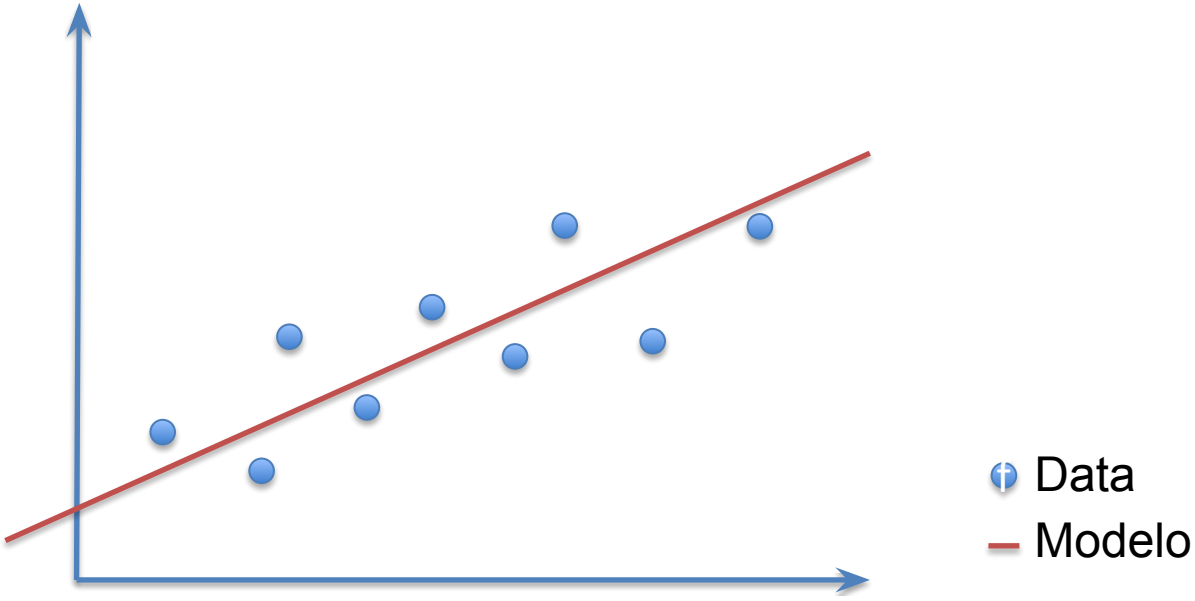

GigaFLOPs per Dollar
CPU
GPU
TPU
TITAN V
(Tensor Cores)
Deep Learning Explosion
GTX 1080 Ti
GeForce
GTX 580
(AlexNet)
GeForce
8800 GTX
40
35
30
25
20
15
10
5
0
1/2004
10/2006
7/2009
4/2012
12/2014
9/2017
Time

Regresión Lineal Simple

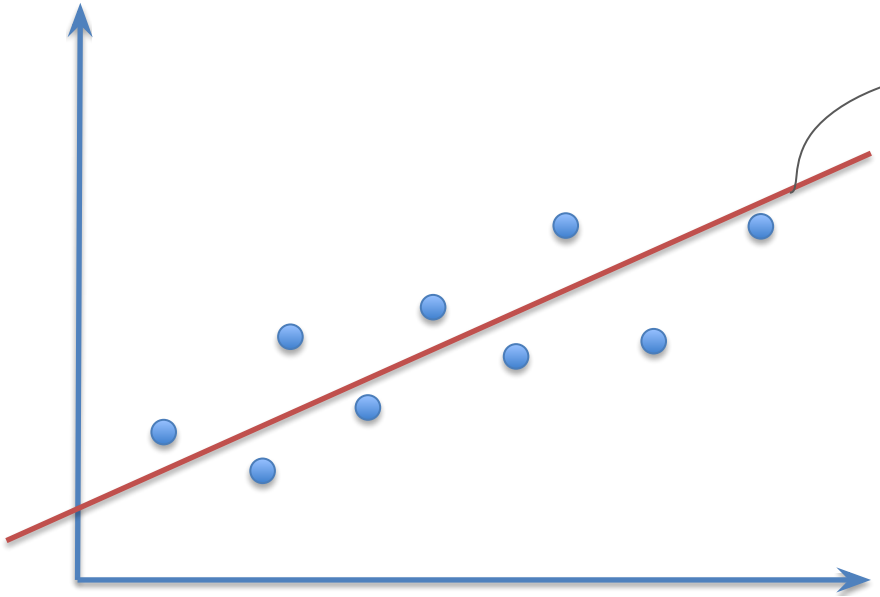
Regresión lineal simple



Regresión lineal simple



Regresión lineal simple



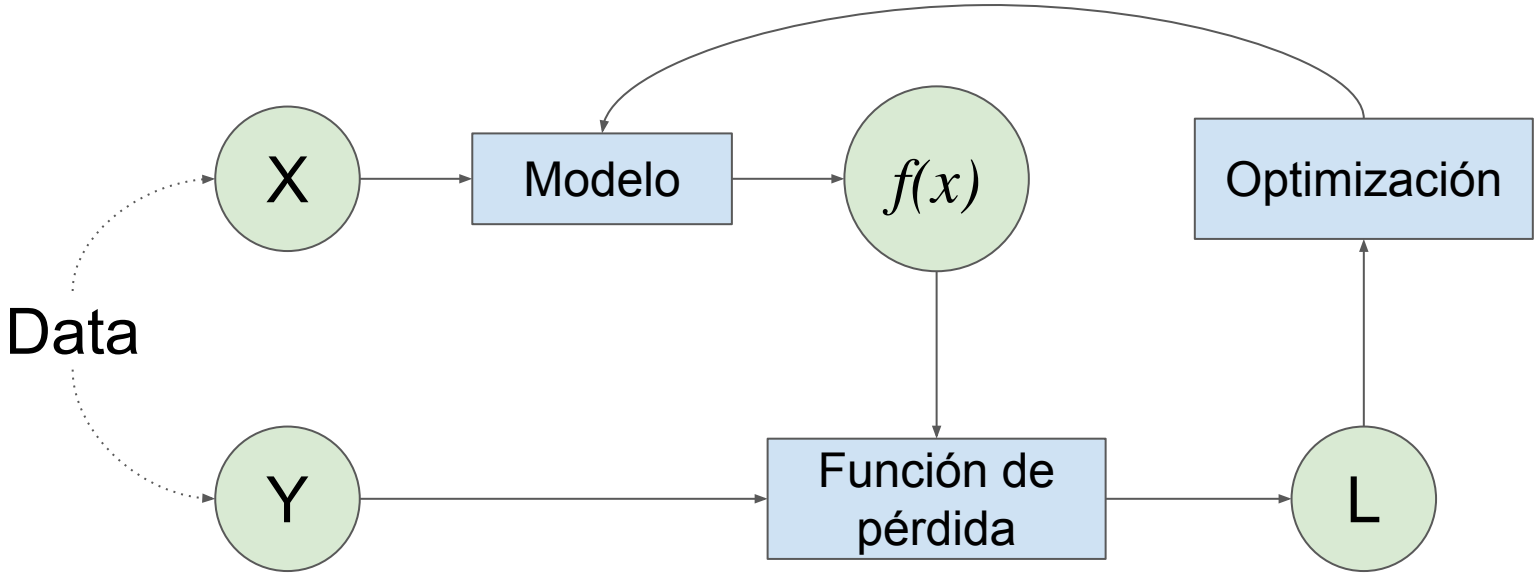
$$f(x) = ax + b$$

Donde:

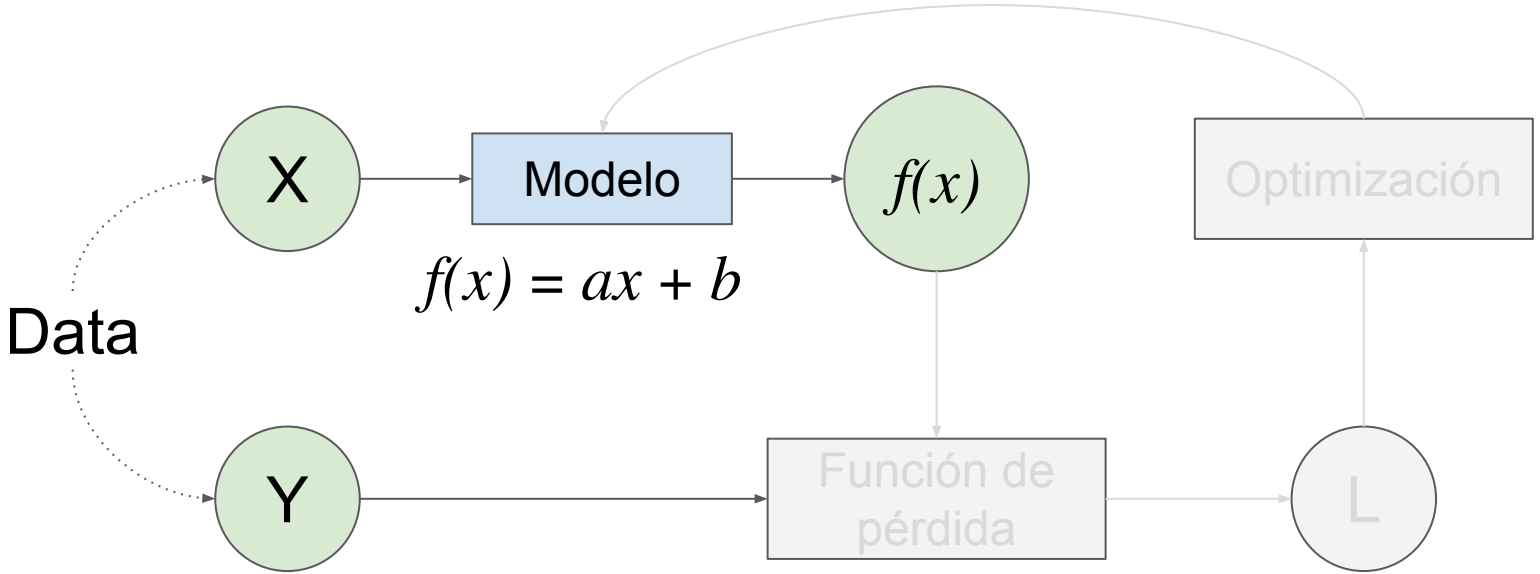
a = pendiente

b = término de intercepción (bias)

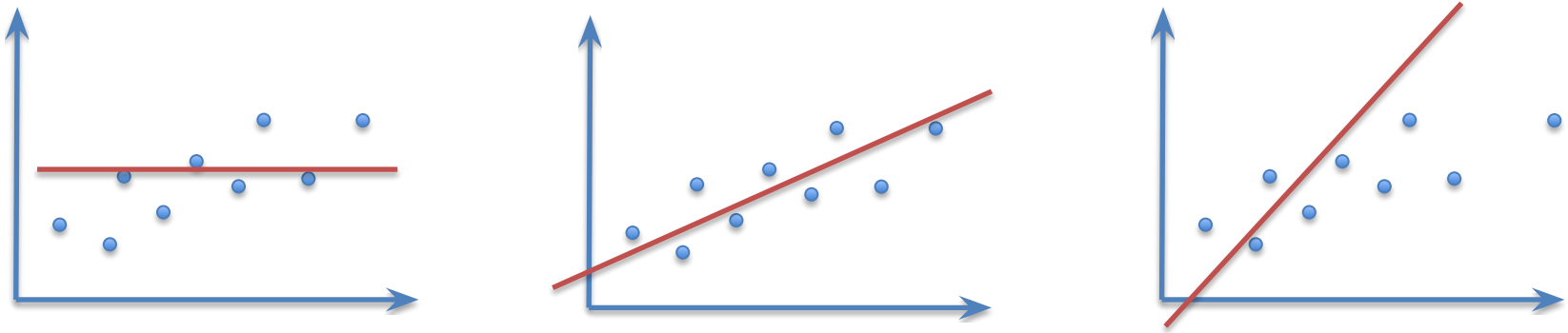
Regresión lineal simple



Regresión lineal simple

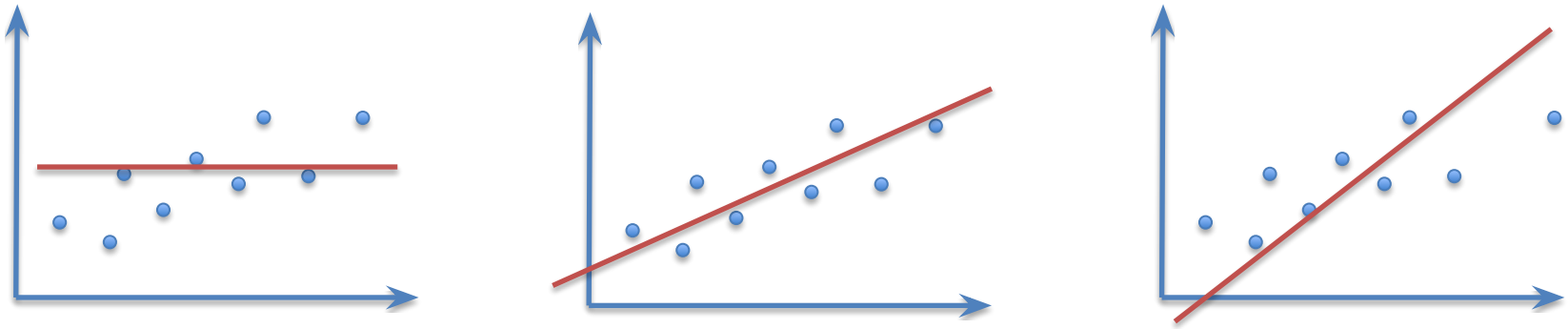


Función de pérdida



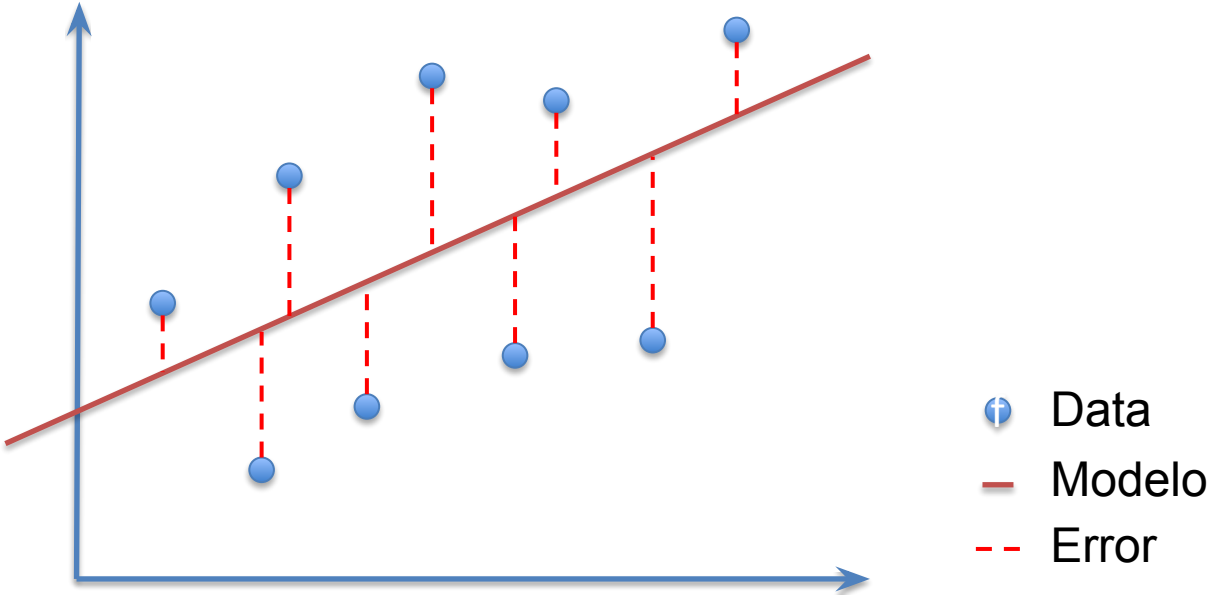
¿Cuál de los modelos describe mejor la data?

Función de pérdida

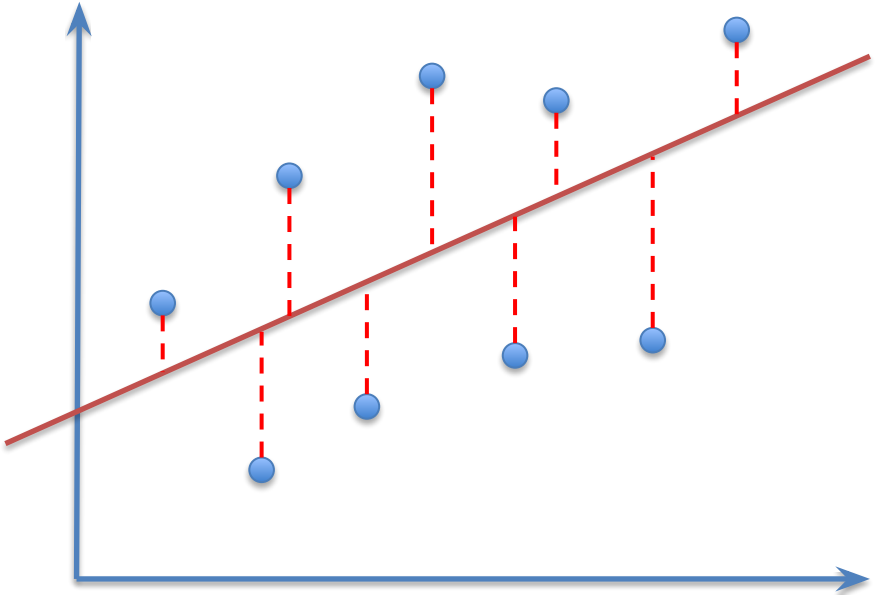


¿Cuál de los modelos describe mejor la data?
Evaluamos cada uno con una **función de pérdida** y escogemos el que tenga la menor pérdida.

Función de pérdida: MSE (Mean squared error)



Función de pérdida: MSE (Mean squared error)



Mean squared error:

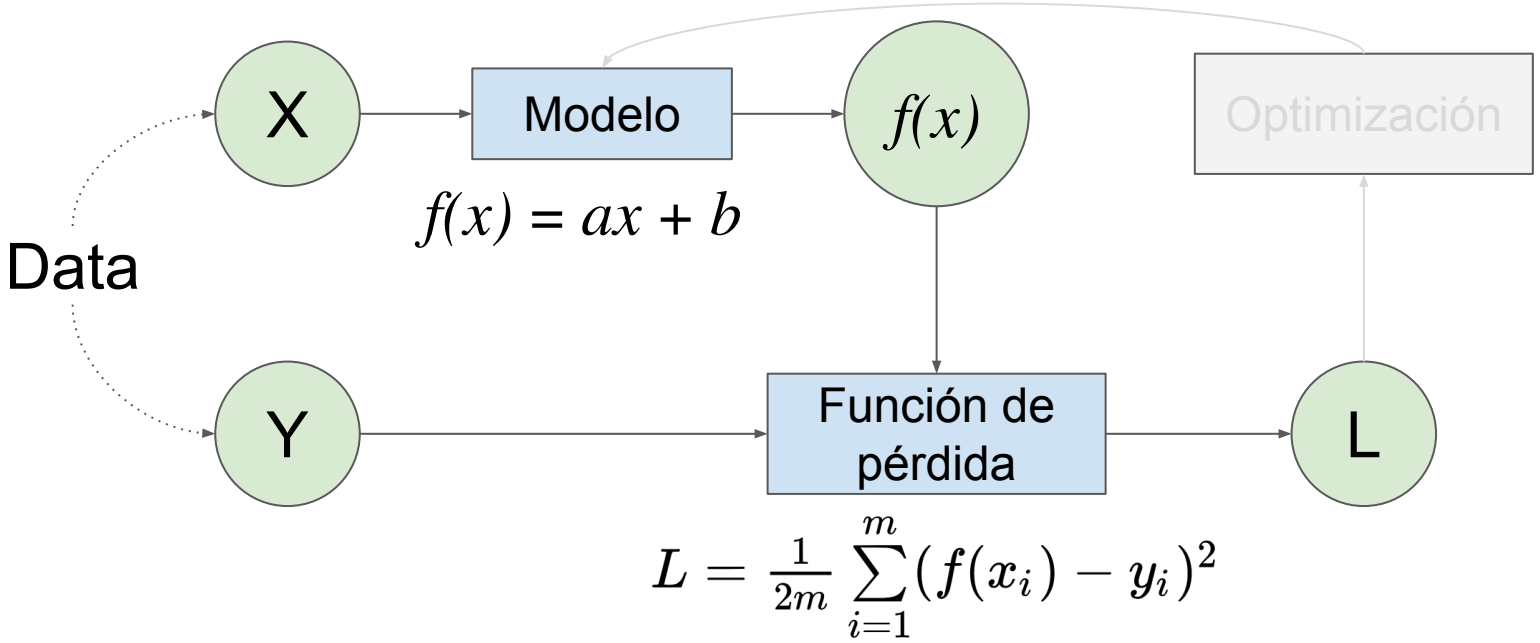
$$L = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

Data x

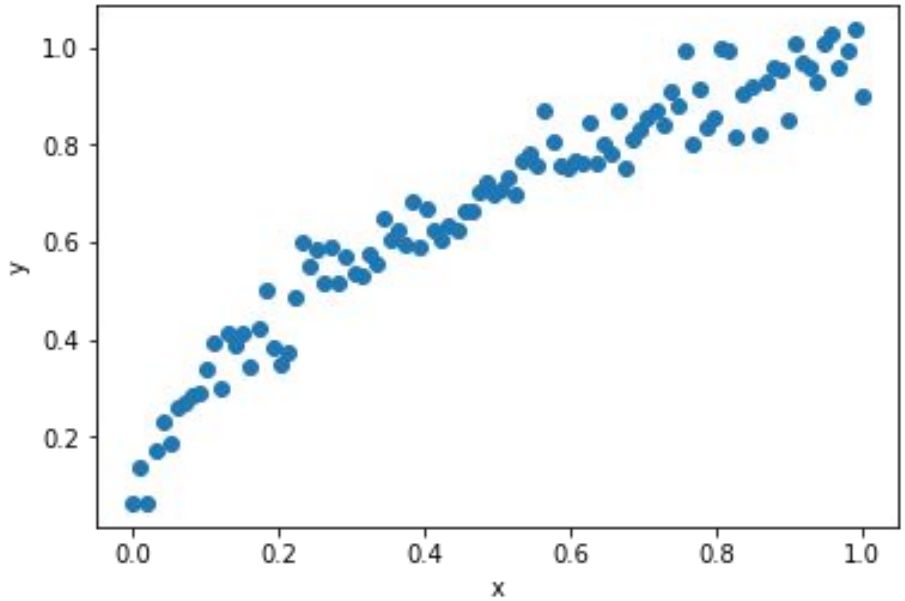
Modelo f(x)

Error f(x) - y

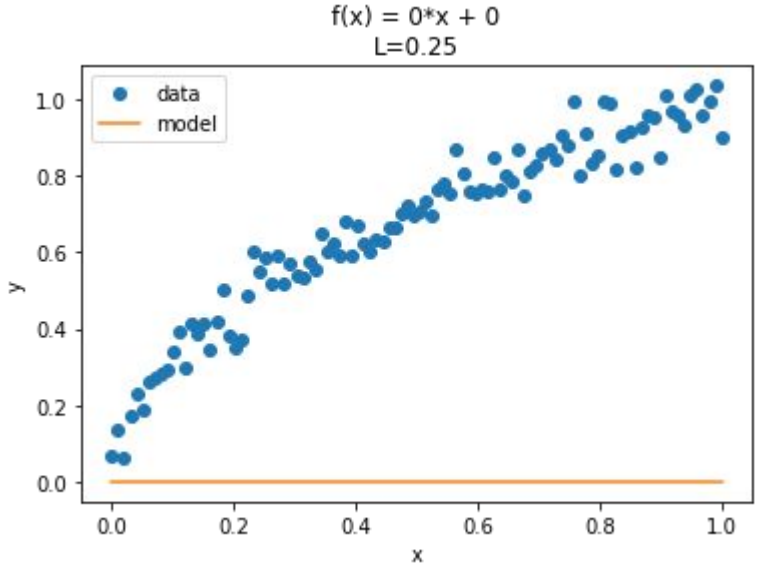
Función de pérdida: MSE (Mean squared error)



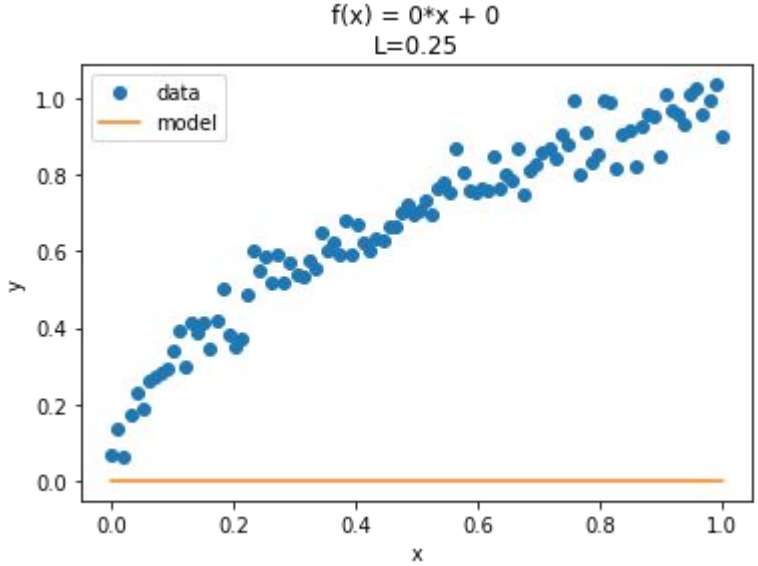
Optimización: Gradient descent

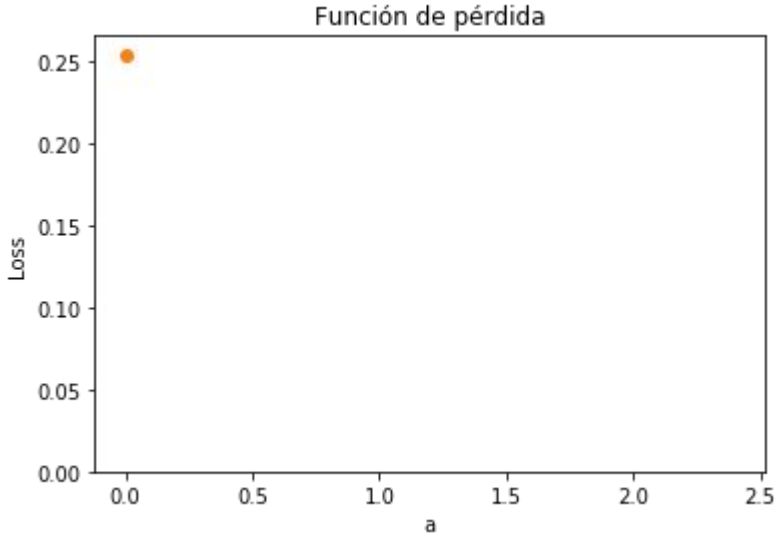


Optimización: Gradient descent

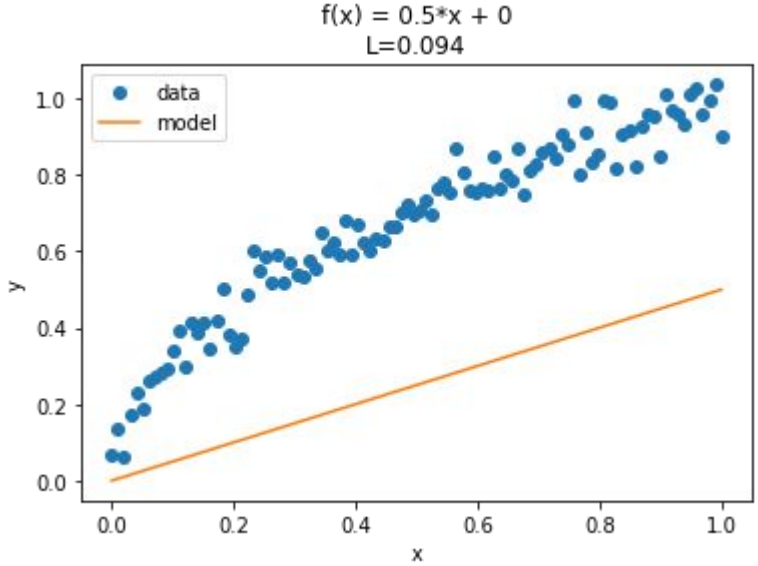


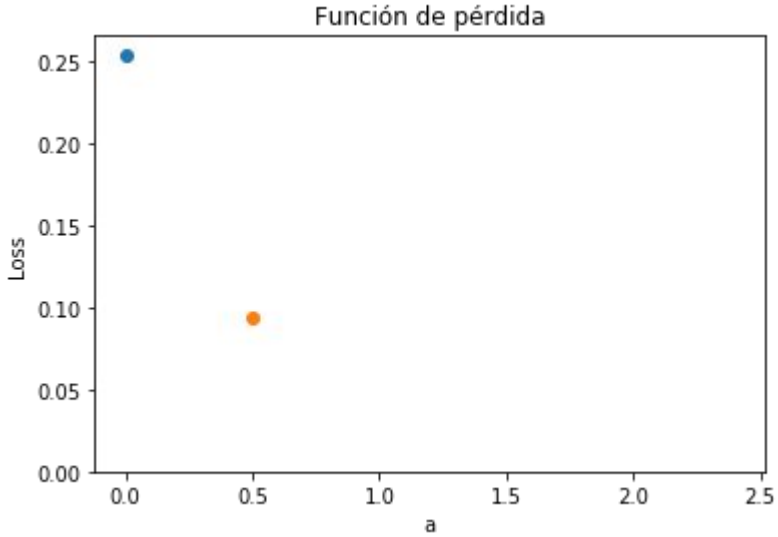
Optimización: Gradient descent



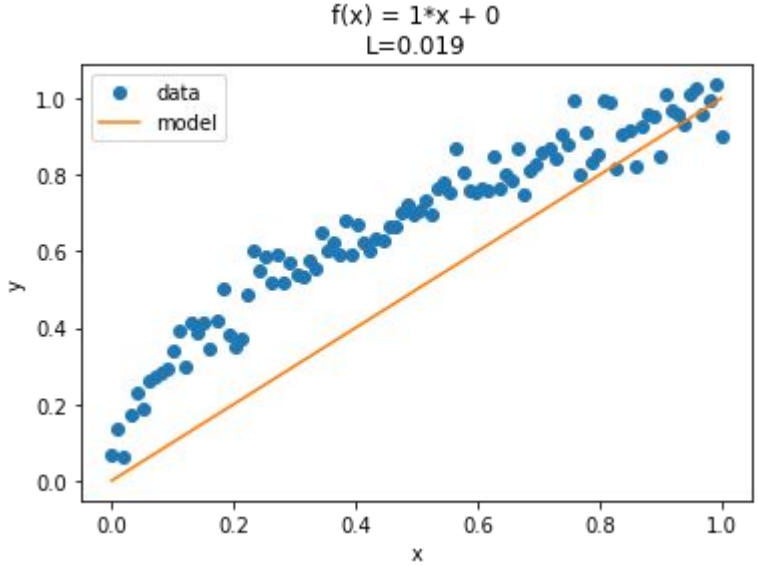


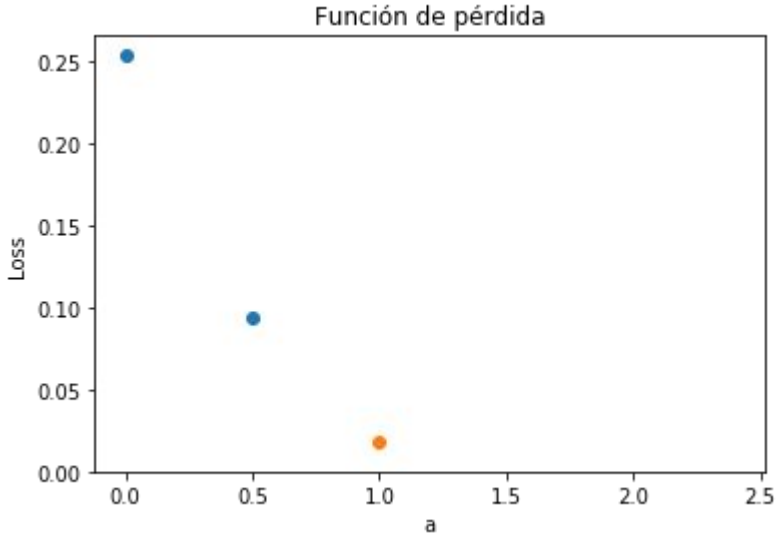
Optimización: Gradient descent



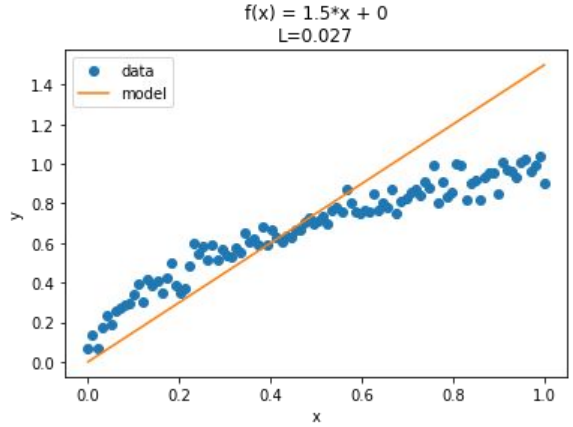


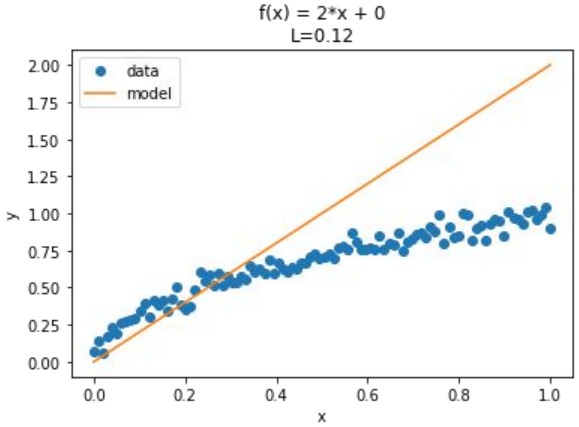
Optimización: Gradient descent

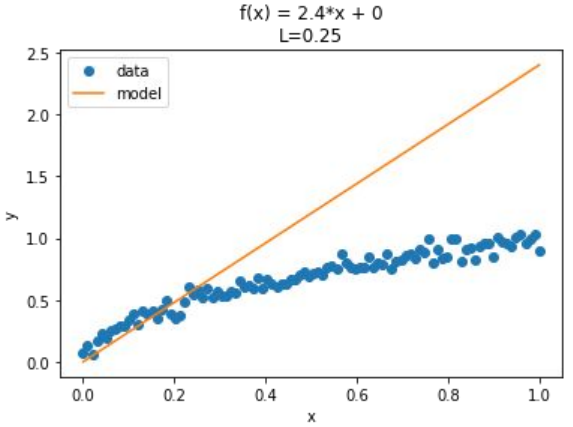


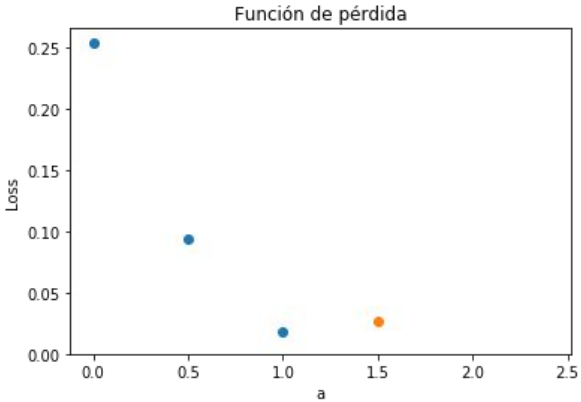


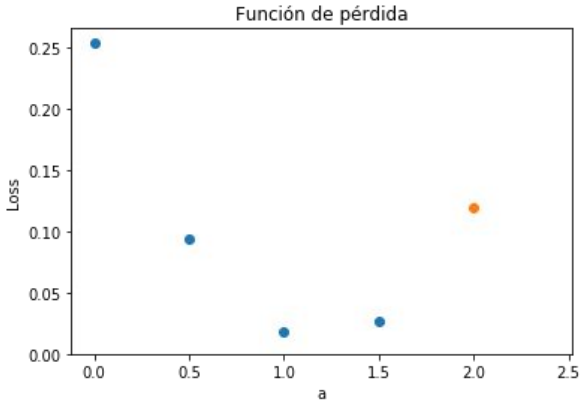
Optimización: Gradient descent

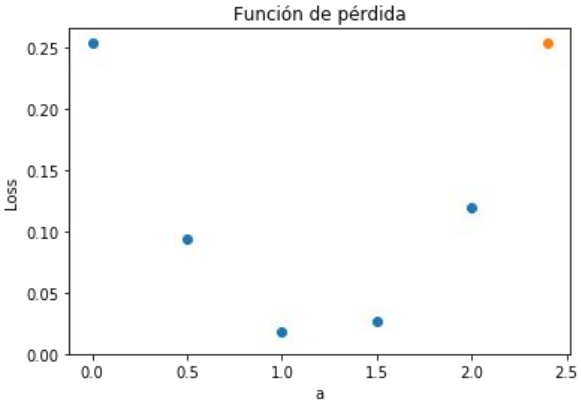




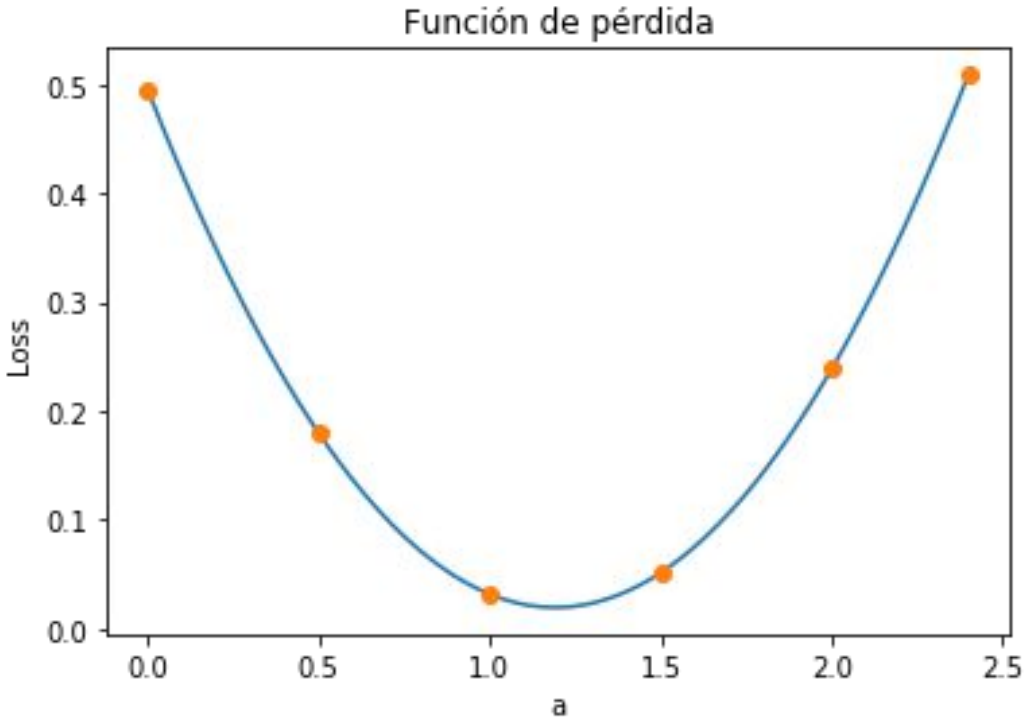




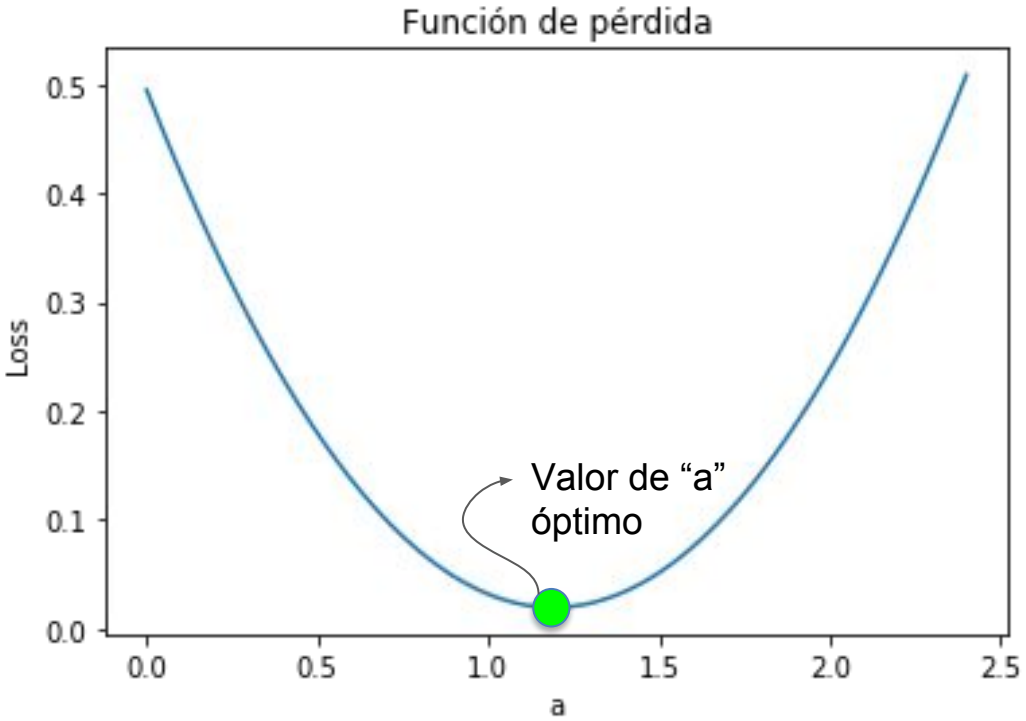




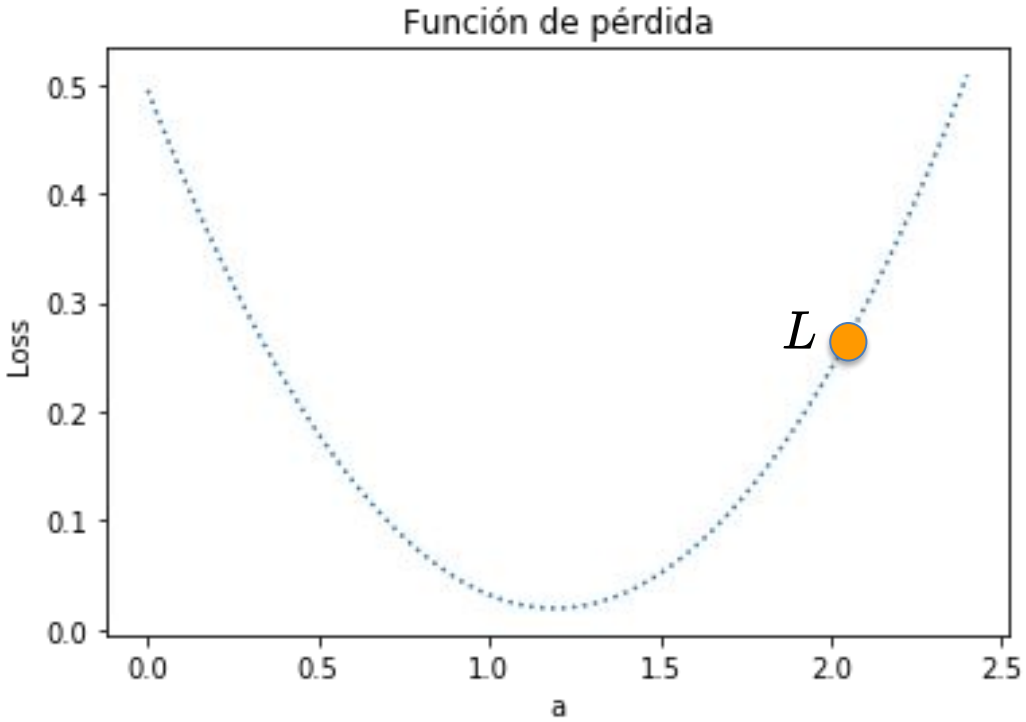
Optimización: Gradient descent



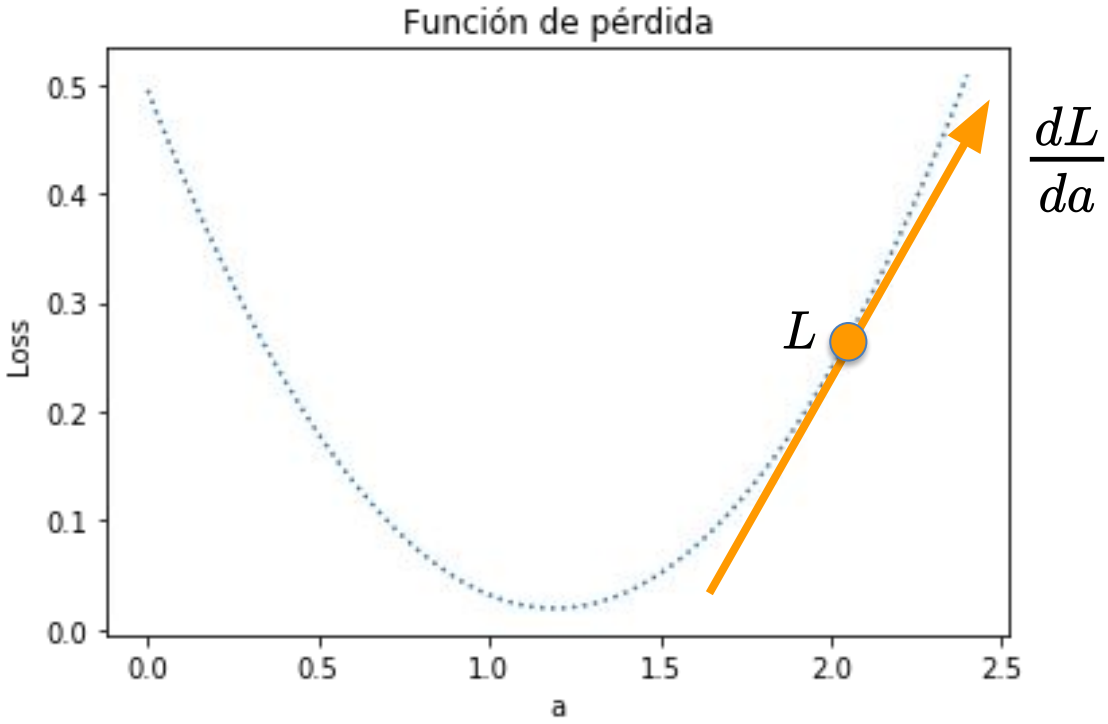
Optimización: Gradient descent



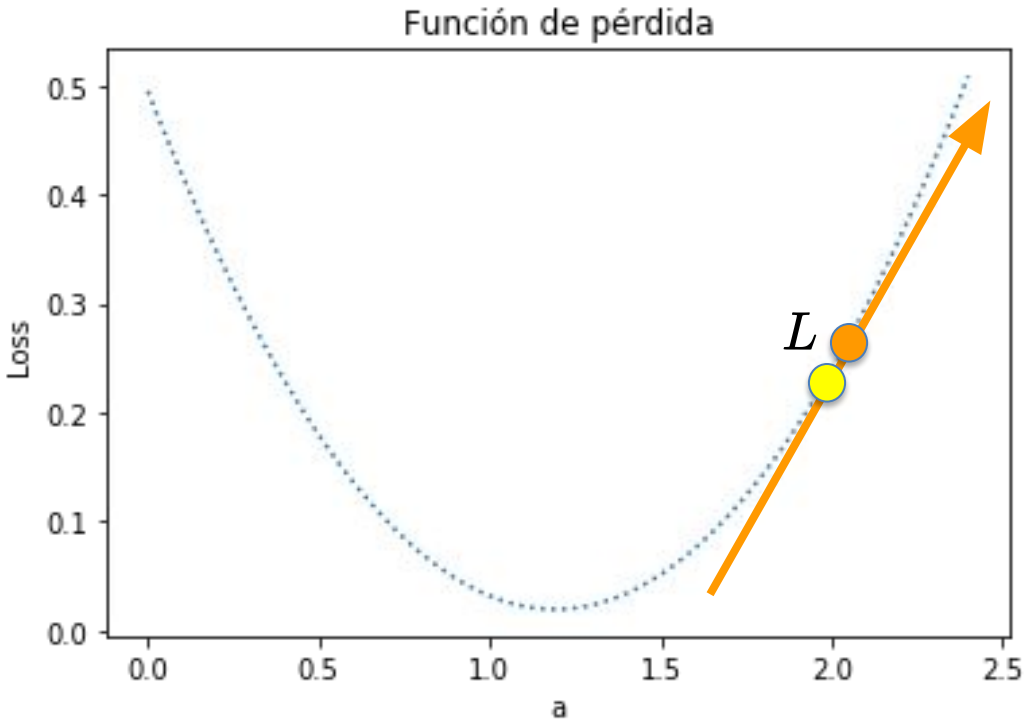
Optimización: Gradient descent



Optimización: Gradient descent



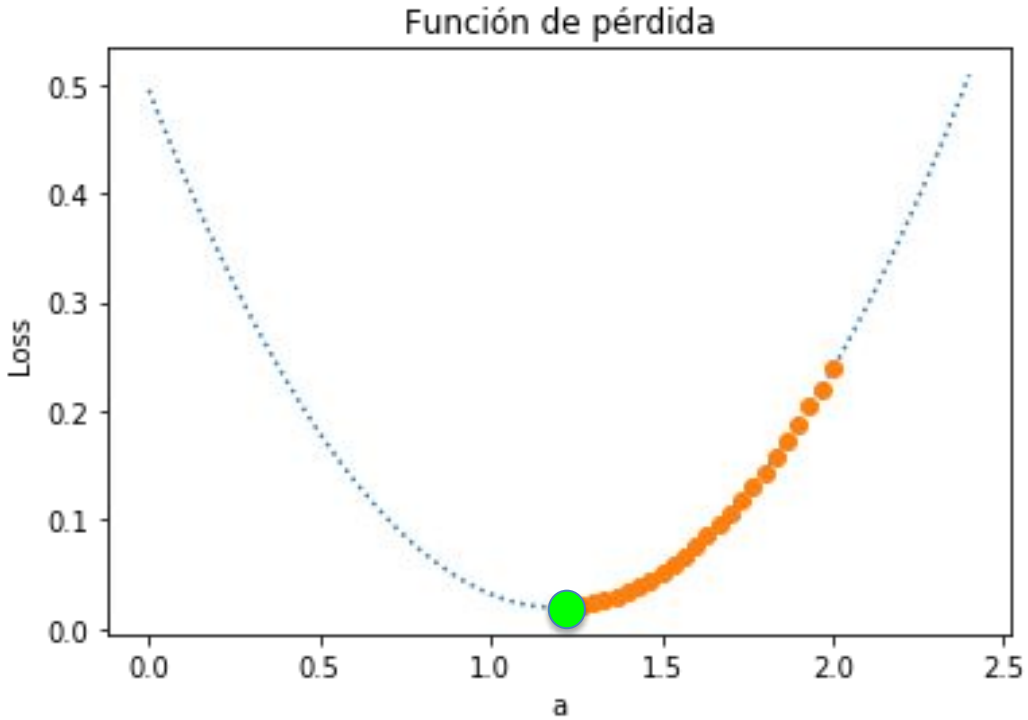
Optimización: Gradient descent



$$\frac{dL}{da}$$

$$new\,a = a - lr * (\frac{dL}{da})$$

Optimización: Gradient descent

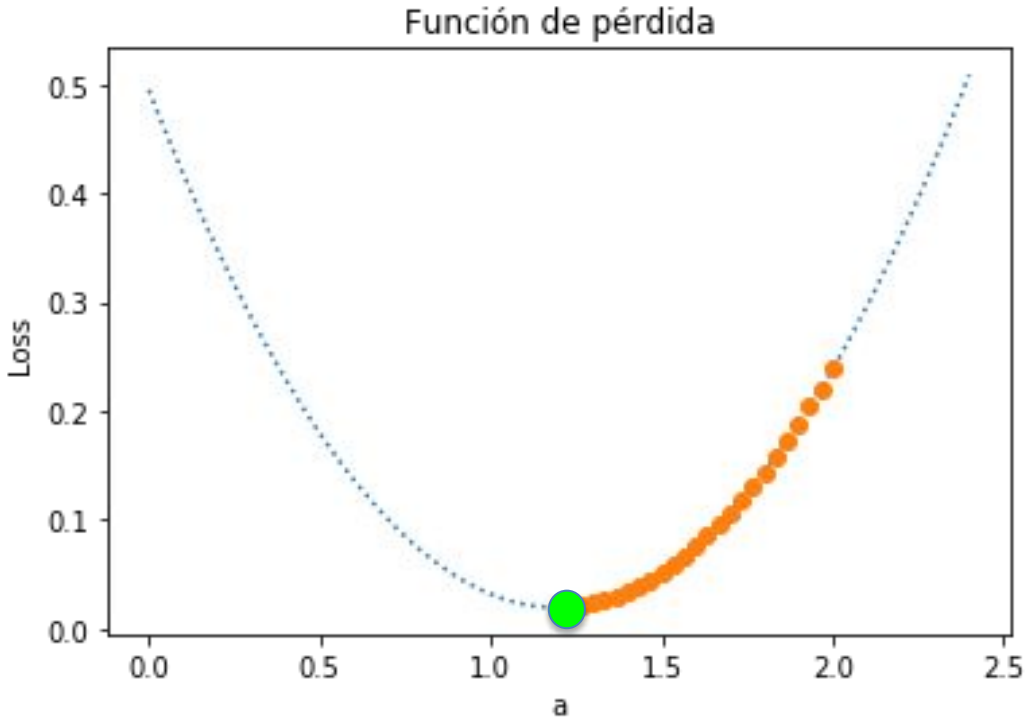


$$\frac{dL}{da}$$

$$new\,a = a - lr * (\frac{dL}{da})$$

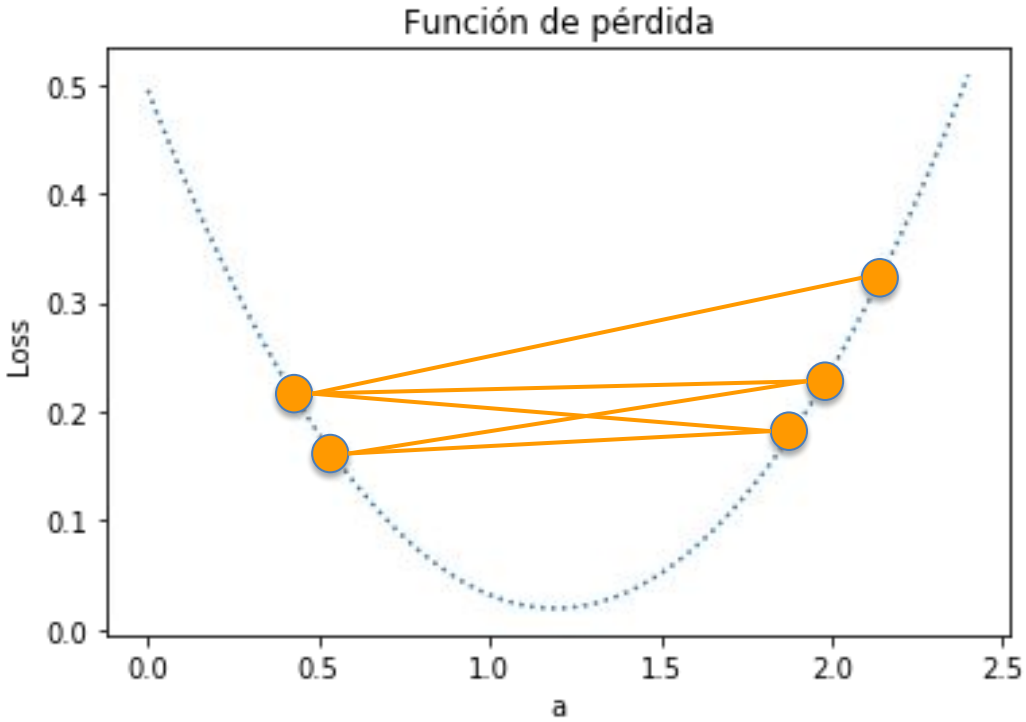
Se itera la optimización hasta converger.

Learning rate: lr



Con un lr bajo, el proceso de optimización puede ser largo.

Learning rate: lr



Pero con un lr muy alto, llegar al punto óptimo se puede volver imposible.

Optimización: Gradient descent

$$f(x_i) = ax_i + b$$

$$L = \frac{1}{2m} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

Optimización: Gradient descent

$$f(x_i) = ax_i + b$$

$$L = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$

$$\frac{dL}{da} = \frac{d(\frac{1}{2m}\sum_{i=1}^{m}(f(x_i)-y_i)^2)}{da} = \frac{1}{2m}\sum_{i=1}^{m}(2(f(x_i) - y_i)(\frac{d(ax_i+b)}{da})) = \sum_{i=1}^{m}(f(x_i) - y_i)(x_i)$$

Optimización: Gradient descent

$$f(x_i) = ax_i + b$$

$$L = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$

$$\frac{dL}{da} = \frac{d(\frac{1}{2m}\sum_{i=1}^{m}(f(x_i)-y_i)^2)}{da} = \frac{1}{2m}\sum_{i=1}^{m}(2(f(x_i) - y_i)(\frac{d(ax_i+b)}{da})) = \sum_{i=1}^{m}(f(x_i) - y_i)(x_i)$$

$$\frac{dL}{db} = \frac{d(\frac{1}{2m}\sum_{i=1}^{m}(f(x_i)-y_i)^2)}{db} = \frac{1}{2m}\sum_{i=1}^{m}(2(f(x_i) - y_i)(\frac{d(ax_i+b)}{db})) = \sum_{i=1}^{m}(f(x_i) - y_i)$$

Optimización: Gradient descent

$$f(x_i) = ax_i + b$$

$$L = \frac{1}{2m}\sum_{i=1}^{m}(f(x_i) - y_i)^2$$

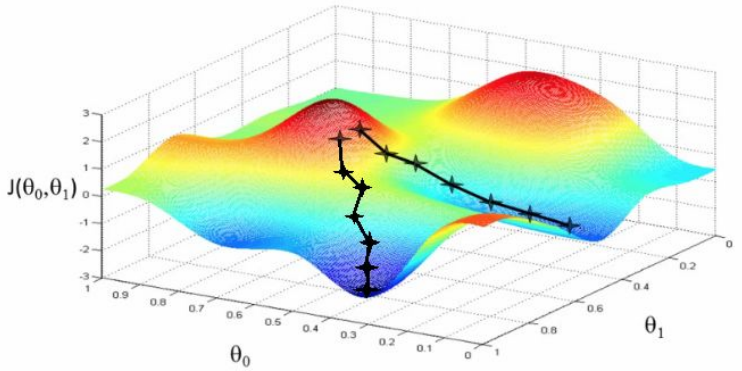
$$\frac{dL}{da} = \sum_{i=1}^{m}(f(x_i) - y_i)(x_i)$$

$$\frac{dL}{db} = \sum_{i=1}^{m}(f(x_i) - y_i)$$

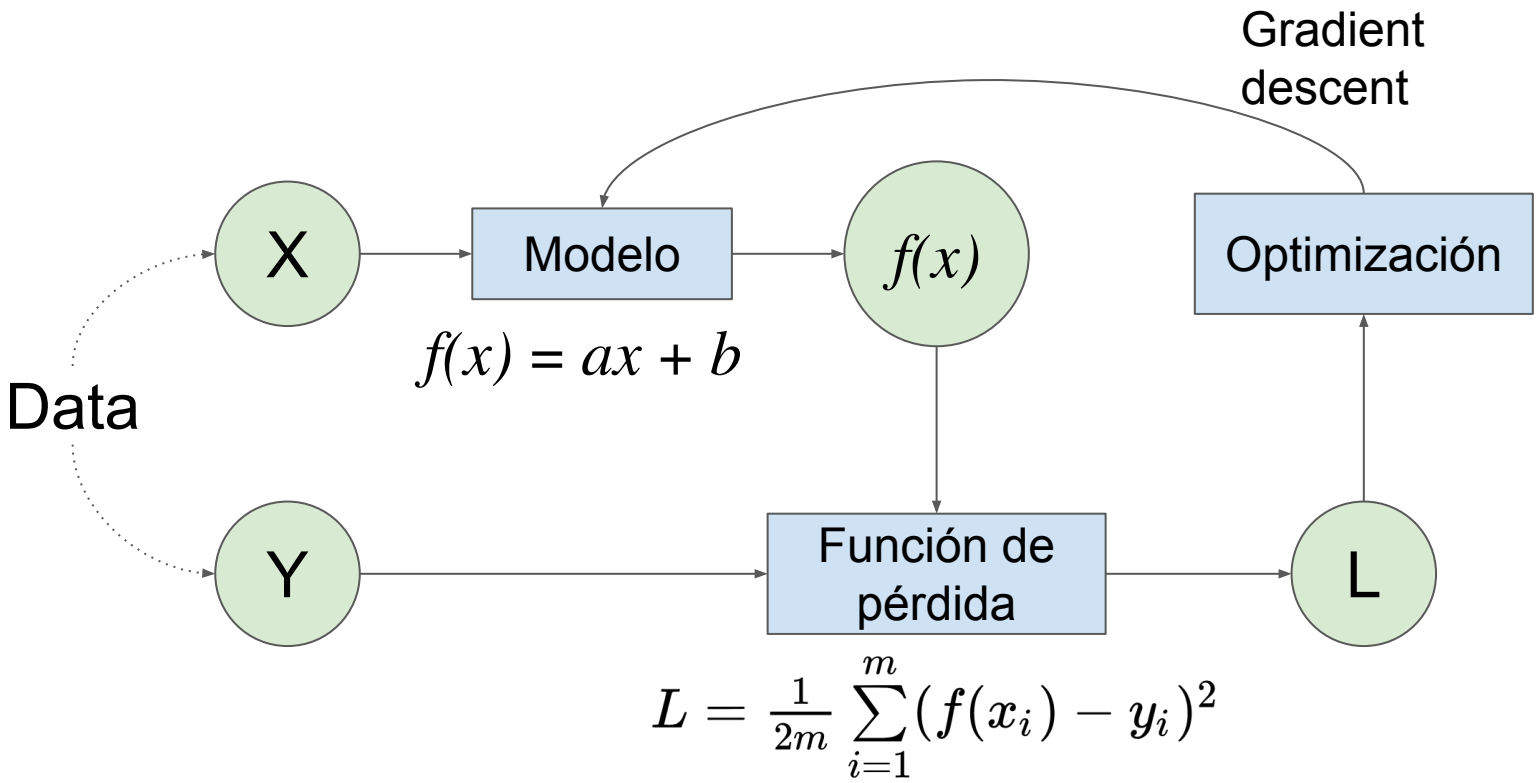
$$iterar:$$

$$new\, a = a - lr * (\frac{dL}{da})$$

$$new\, b = b - lr * (\frac{dL}{db})$$



Optimización: Gradient descent



Código

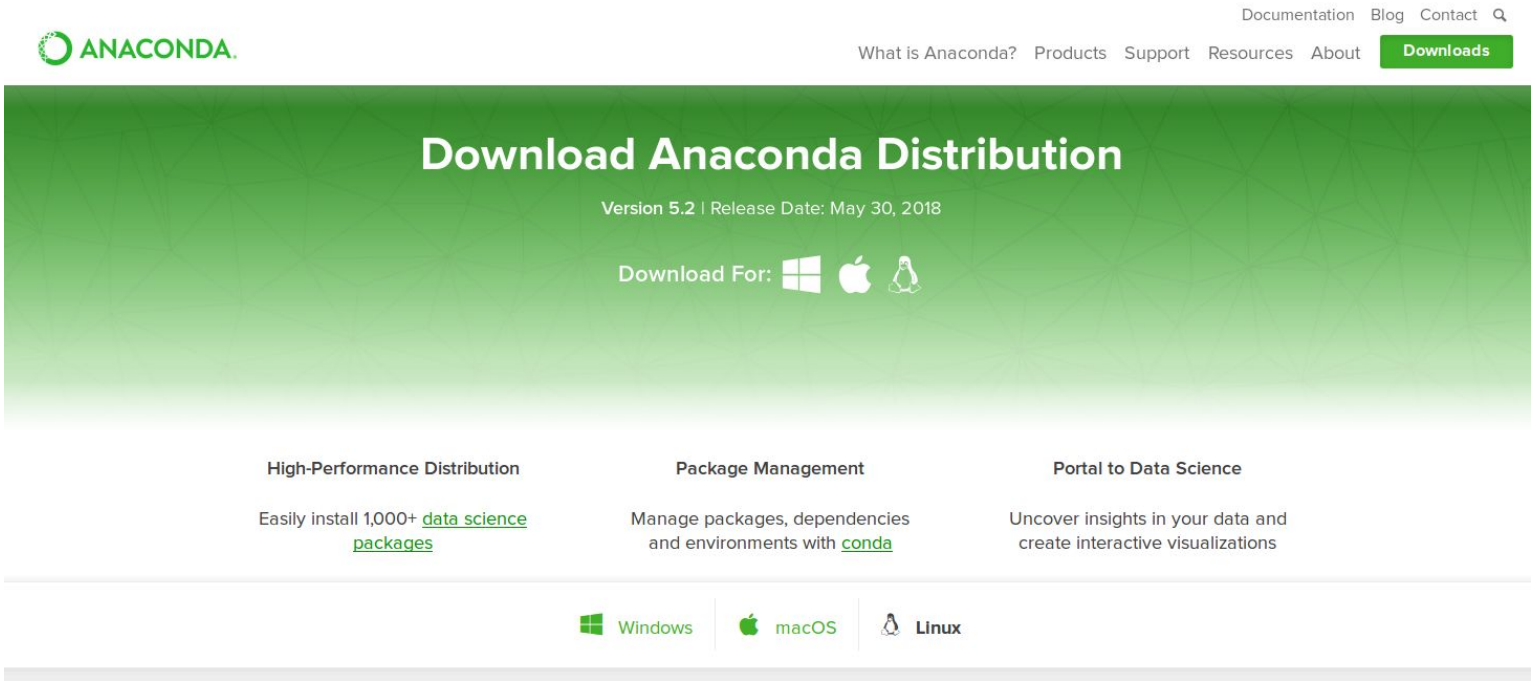
Herramientas

Ambiente de trabajo

Librerías

Anaconda (package manager)

https://www.anaconda.com/download



Anaconda (package manager)

Listar librerías instaladas:
```
> conda list
```

Instalar librería:
```
> conda install scikit-learn
```

Desinstalar:
```
> conda uninstall scikit-learn
```

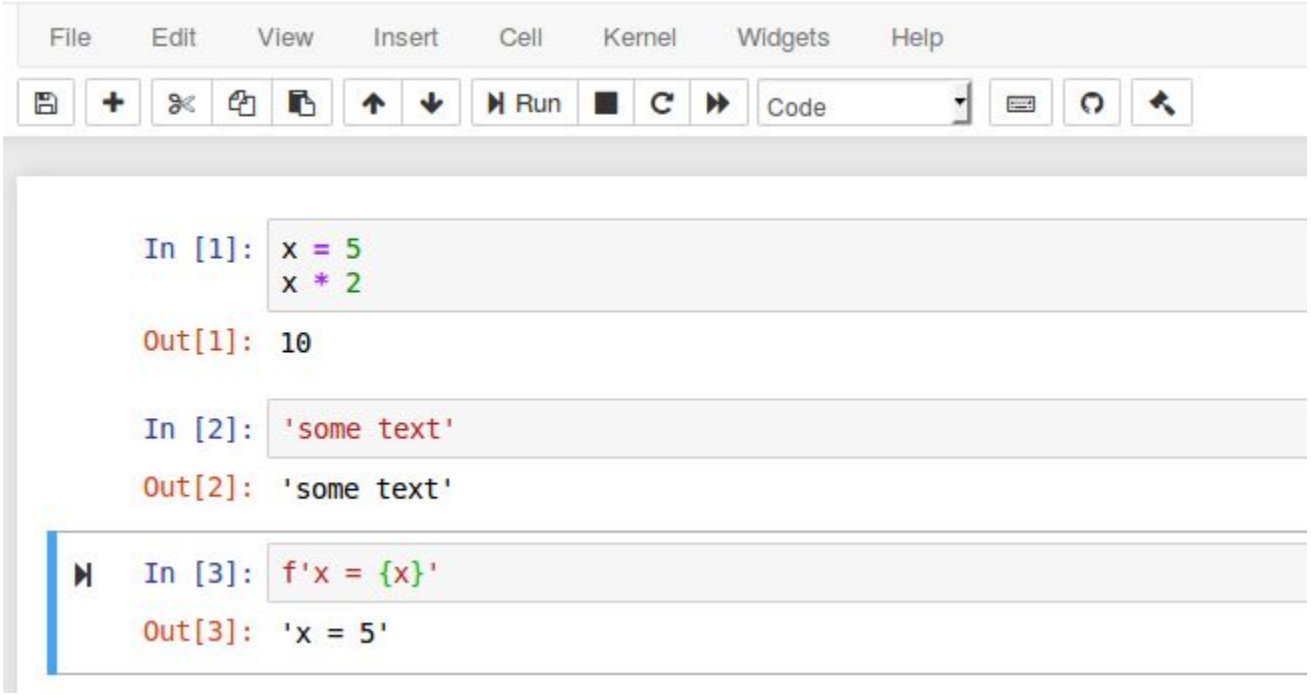
Actualizar librerías:
```
> conda update --all
```

Ambiente de trabajo

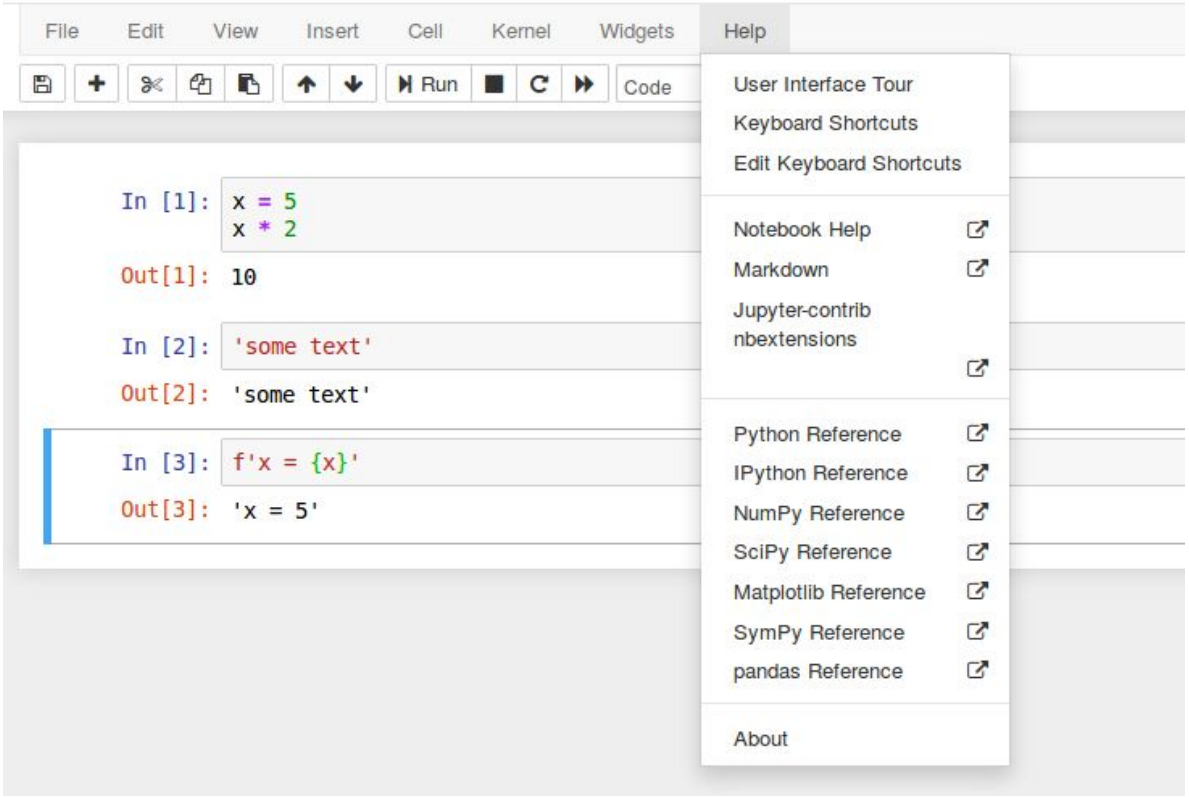
Iniciar jupyper:

```
> jupyter-notebook
```

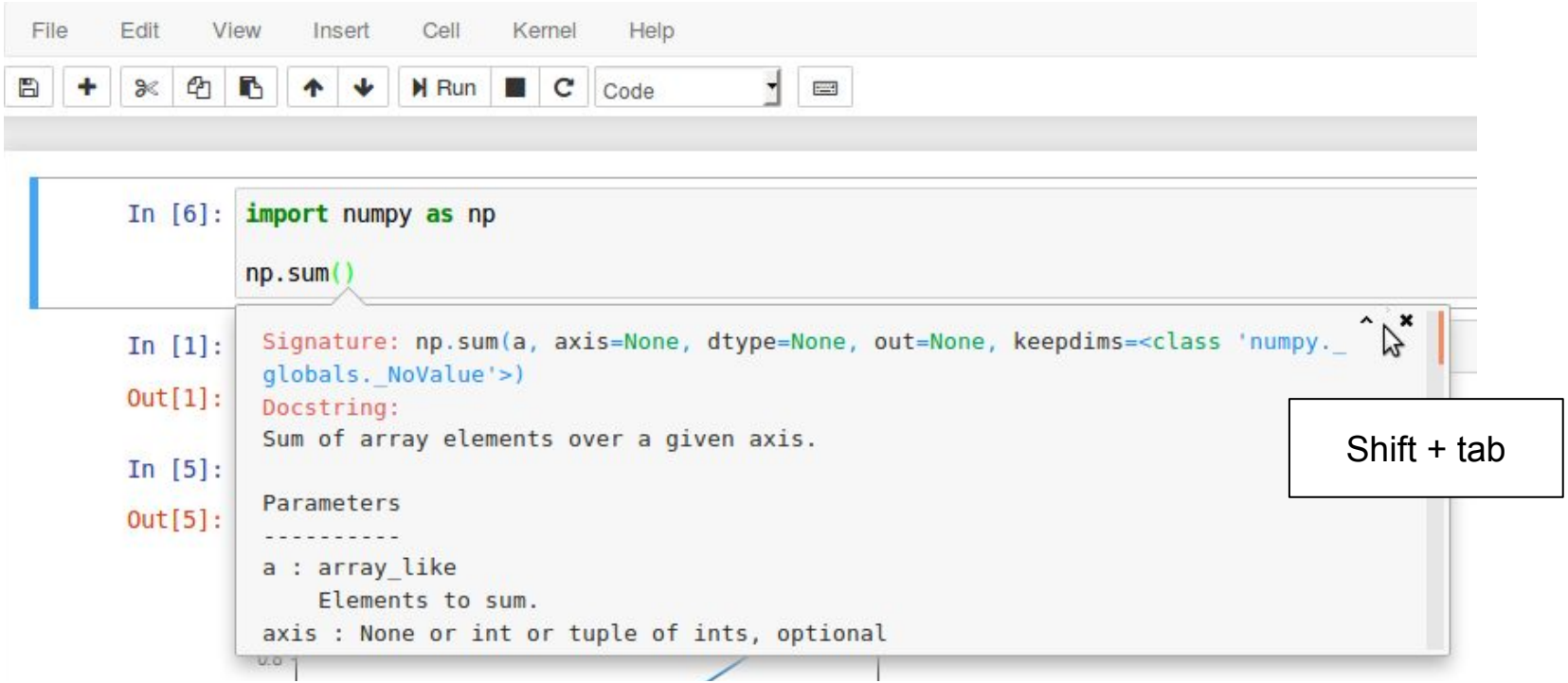
Jupyter notebook



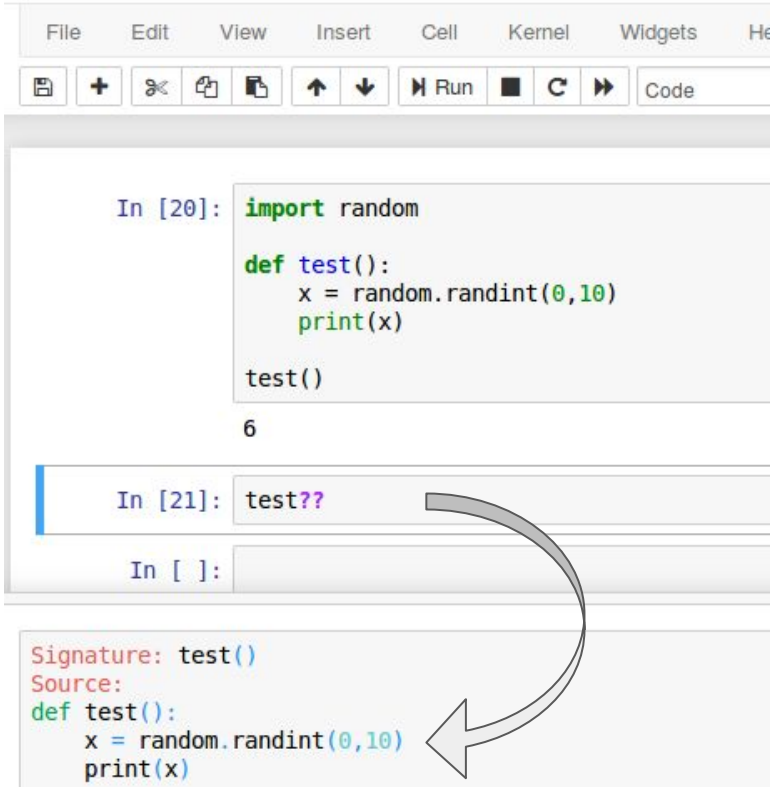
Jupyter notebook



Ver documentación



Ver código fuente

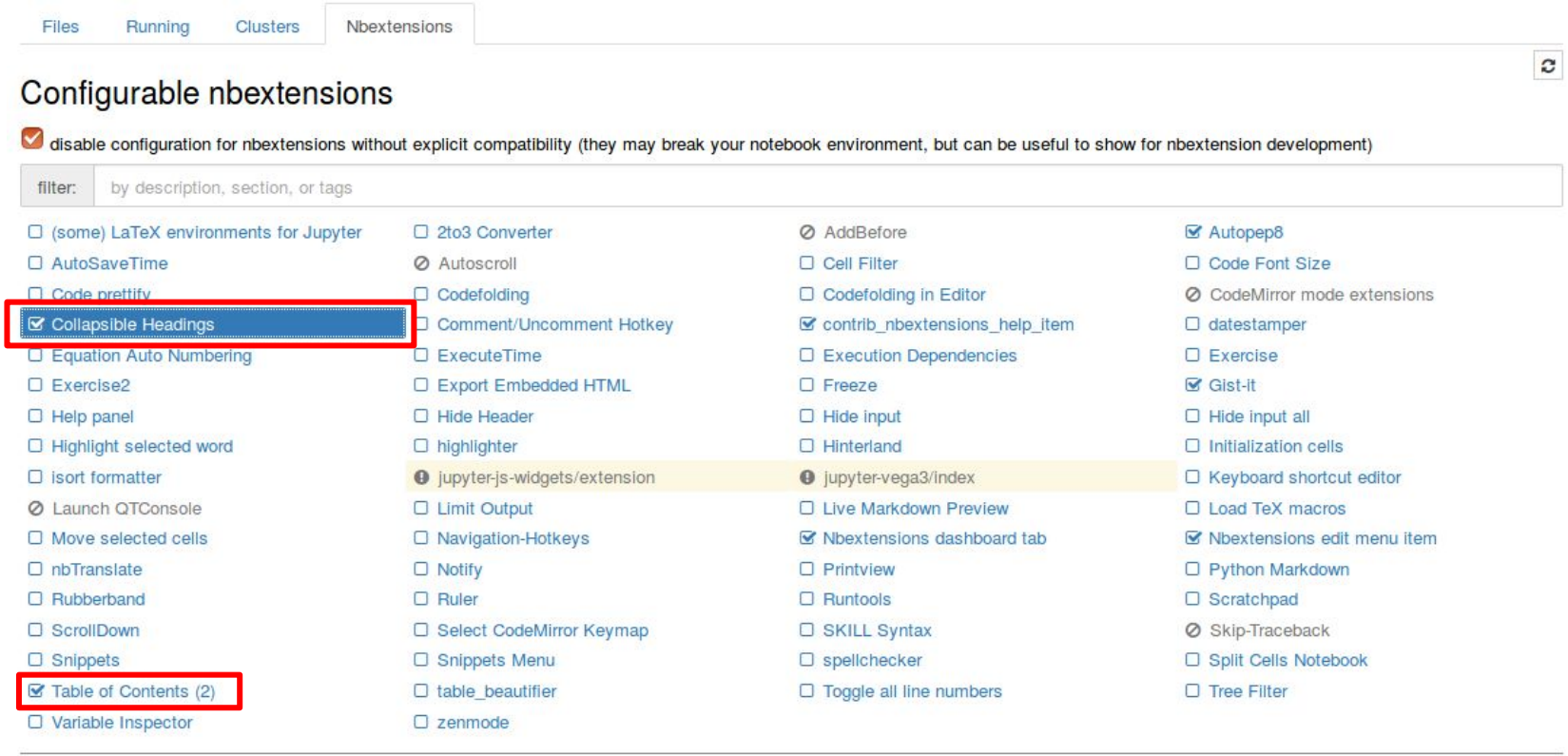


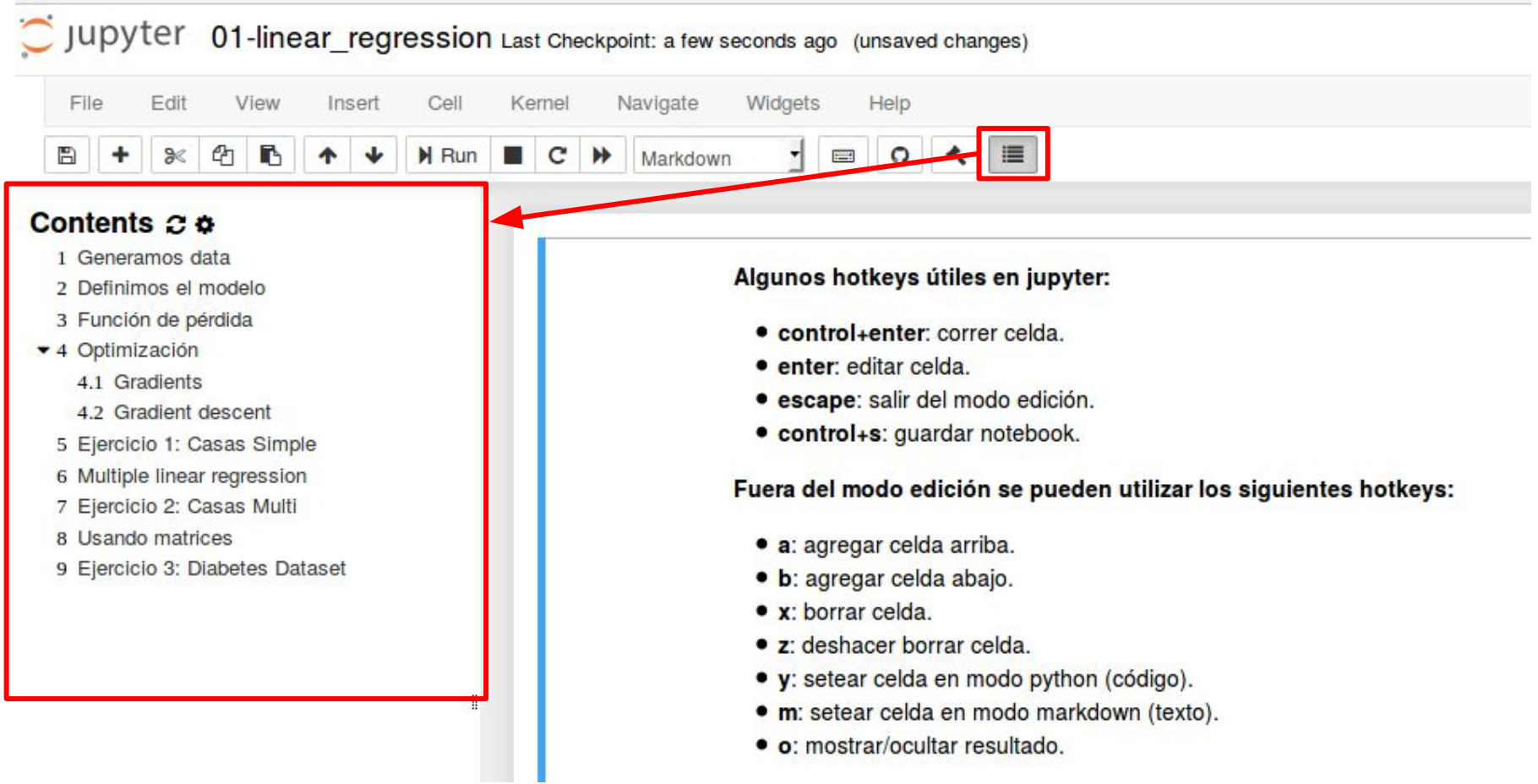
Notebook

Jupyter extensions

```
> conda install -c conda-forge jupyter_contrib_nbextensions
```

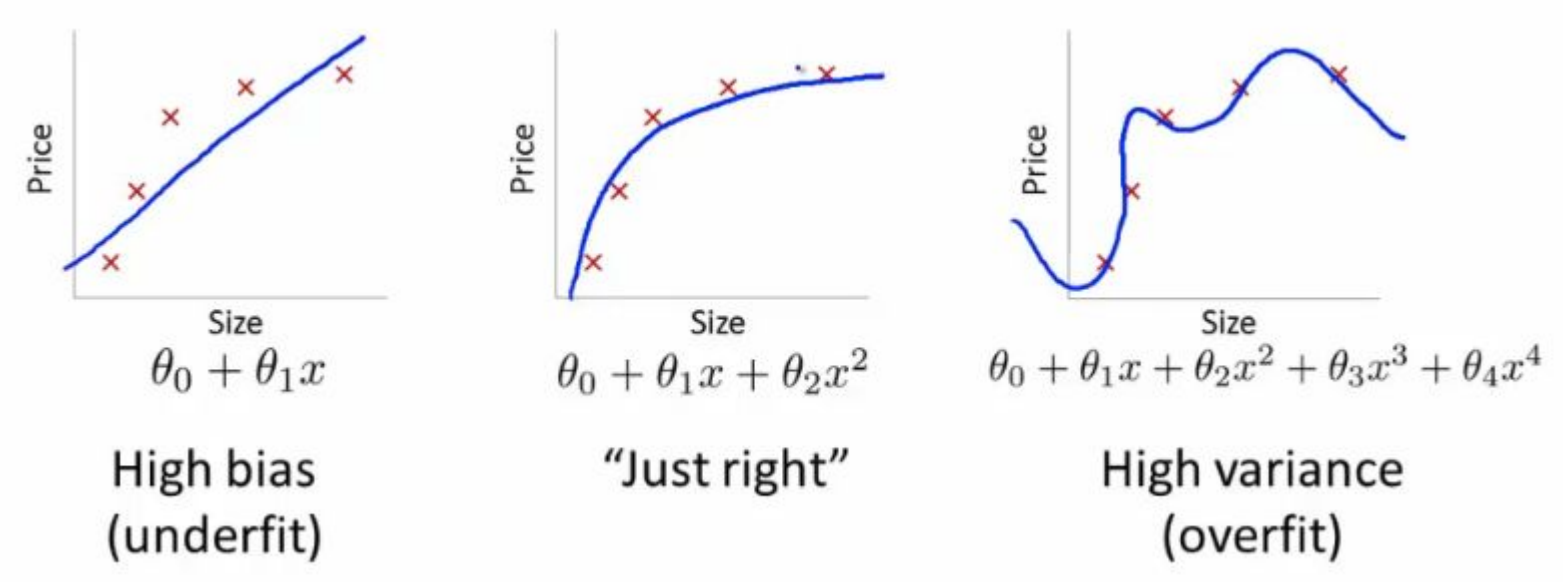
Jupyter extensions



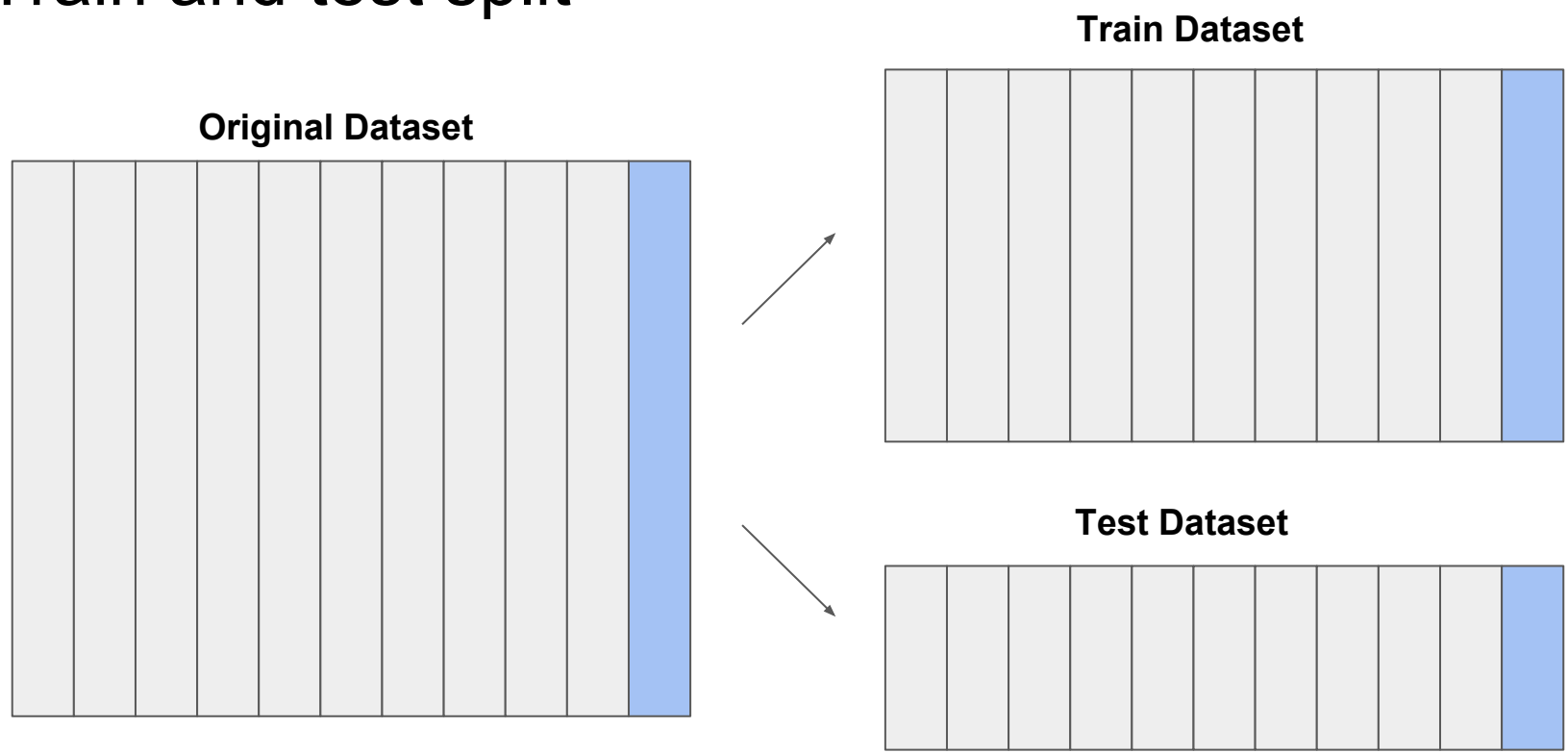

jupyter 01-linear_regression Last Checkpoint: a few seconds ago (unsaved changes)
File Edit View Insert Cell Kernel Navigate Widgets Help
Run
Markdown
Contents
1 Generamos data
2 Definimos el modelo
3 Función de pérdida
4 Optimización
4.1 Gradients
4.2 Gradient descent
5 Ejercicio 1: Casas Simple
6 Multiple linear regression
7 Ejercicio 2: Casas Multi
8 Usando matrices
9 Ejercicio 3: Diabetes Dataset
Algunos hotkeys útiles en jupyter:
control+enter: correr celda.
enter: editar celda.
escape: salir del modo edición.
control+s: guardar notebook.
Fuera del modo edición se pueden utilizar los siguientes hotkeys:
a: agregar celda arriba.
b: agregar celda abajo.
x: borrar celda.
z: deshacer borrar celda.
y: setear celda en modo python (código).
m: setear celda en modo markdown (texto).
o: mostrar/ocultar resultado.

Validación de modelos

Overfitting



Train and test split



Even better: Train, validation and test split