(A Very Brief) Introduction to Machine Learning

Pablo Mesejo

pmesejo@go.ugr.es

Universidad de Granada

Departamento de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE GRANADA

DaSCI

Instituto Andaluz de Investigación en **Data Science** and **Computational Intelligence**

References

- Murphy, K. P. (2022). *Probabilistic machine learning: an introduction*. MIT press.
- Abu-Mostafa, Y. S., Magdon-Ismail, M., & Lin, H. T. (2012). *Learning from data.* AMLBook. https://work.caltech.edu/library/index.html
- Hastie, T., Tibshirani, R. & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer.
- Bishop, C. M. (2006). *Pattern recognition and machine learning.* Springer.
- Szeliski (2022). Chapter 5.1 and 5.2

What is machine learning?

- Provide machines with the ability to automatically learn from data:

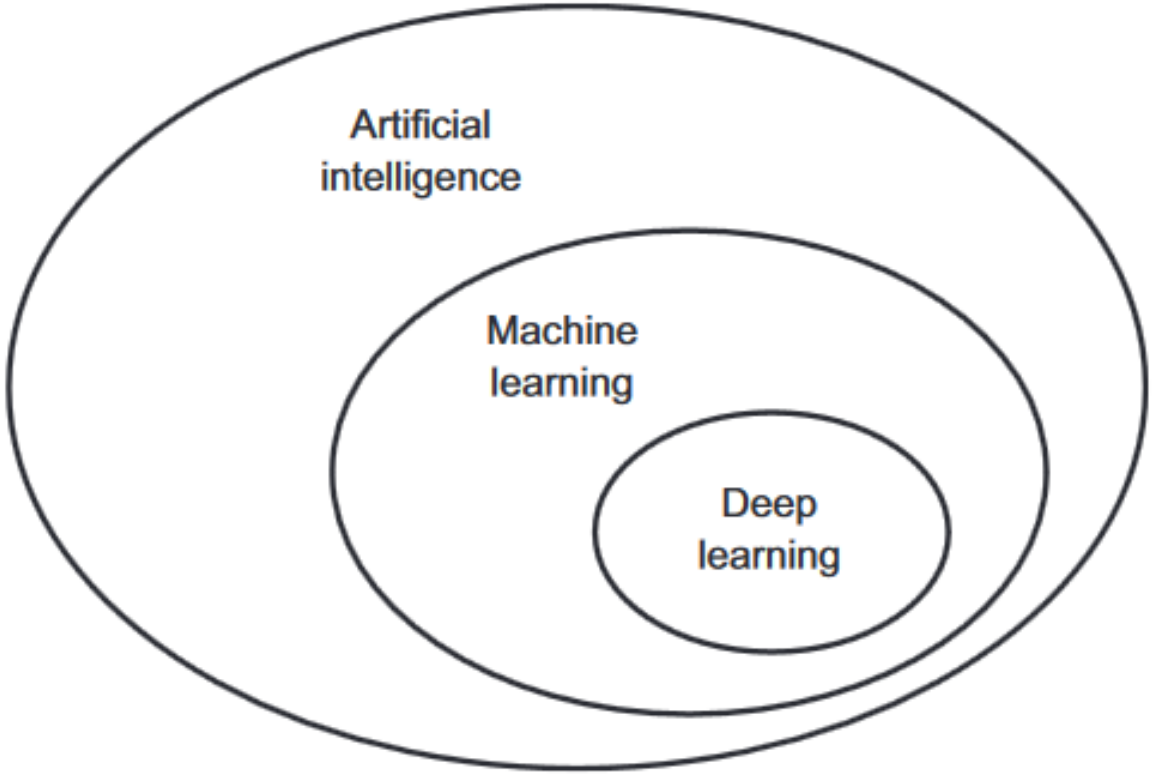
  data + learning algorithms = machine learning

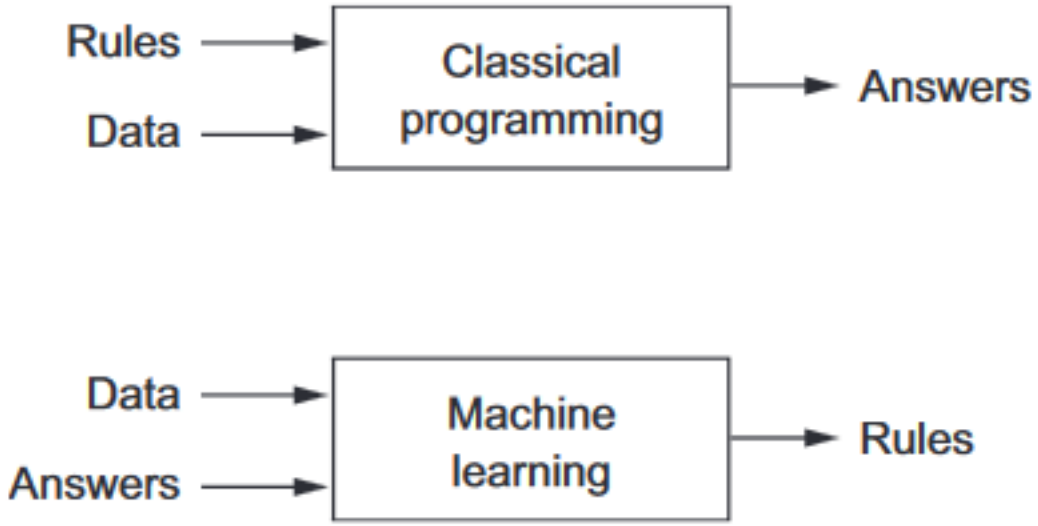
- "Use a set of observations to discover an underlying process" (Learning From Data, Abu-Mostafa et al., 2012)
  - There is a pattern
  - We can't describe it mathematically
  - We have data

Example: movie recommender

- Our tastes/preferences are not arbitrary, but rather follow certain patterns.
- We cannot mathematically define why we like what we like.
- We have examples of movies we liked.

What is machine learning?





Images taken from “Deep Learning with Python” (F. Chollet, 2018)

A bit of history

- McCulloch, W. S., & Pitts, W. (1943). *"A logical calculus of the ideas immanent in nervous activity"*. The bulletin of mathematical biophysics, 5, 115-133.
  - First neuron model: It only allowed for **binary inputs and outputs**, it only used the **threshold step activation function**, and it did **not incorporate weighting** for the different inputs.
- Samuel, Arthur L. (1959). *"Some Studies in Machine Learning Using the Game of Checkers"*. IBM Journal of Research and Development. 44: 206–226.
  - Introduction of the term "machine learning" (self-teaching computers)
- Rosenblatt, F. (1962). *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms*. Spartan books.
  - Discussed and extended the concept of **Perceptron** (initially developed by him in 1957-58)
- Other models, like Adaline:
  - Widrow, B. (1960). *An adaptive "ADALINE" neuron using chemical "memistors".* Tech.Report Nº 1553-2. Office of Naval Research.
  - Adaline has a different learning rule than Perceptron and employs a different activation function (linear).

A bit of history

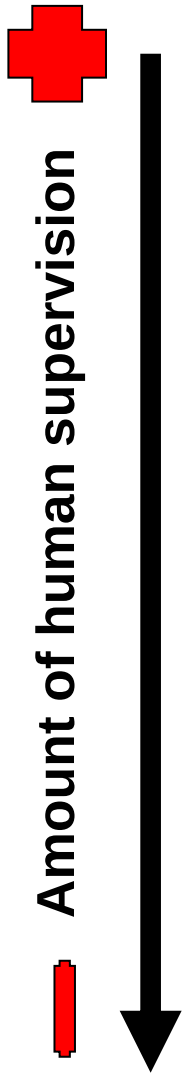
NEW NAVY DEVICE LEARNS BY DOING

Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

WASHINGTON, July 7 (UPI) —The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The New York Times, 1958, on Frank Rosenblatt's Perceptron, developed in 1957, and based on McCulloch&Pitts' ideas (1943) and Hebbian learning (1949)

Main paradigms of machine learning

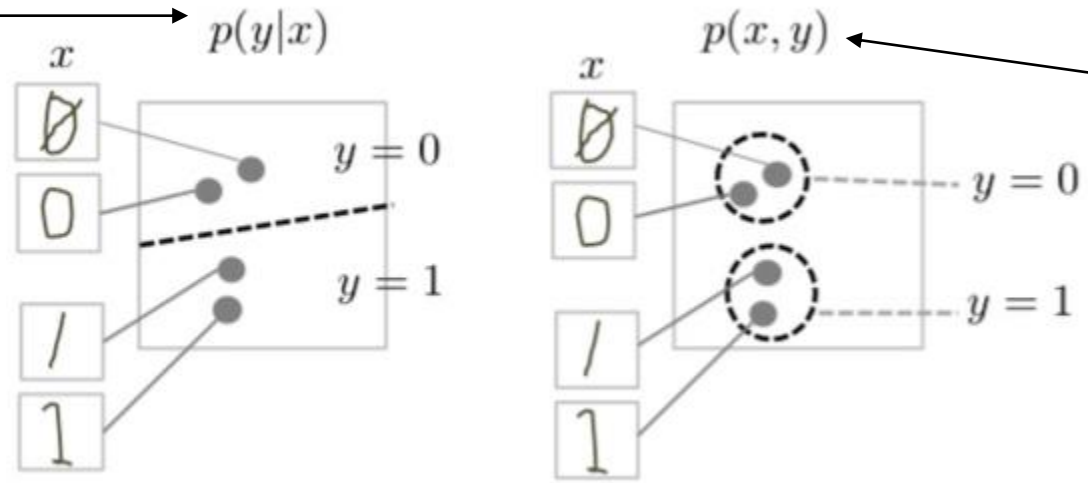


- **Supervised** learning
  - Weakly supervised
  - Semi-supervised / Few-shot
- **Reinforcement** learning
- **Unsupervised** learning
  - Self-supervised

Main paradigms of machine learning

- There are other categorizations:
  - **Discriminative** models discriminate (draw boundaries) between different kinds of data instances.
  - **Generative** models try to model how data is placed throughout the space and can generate new data instances.

Discriminative models capture the conditional probability p(Y | X)



Generative models capture the joint probability p(X, Y), or just p(X) if there are no labels

https://developers.google.com/machine-learning/gan/generative?hl=en

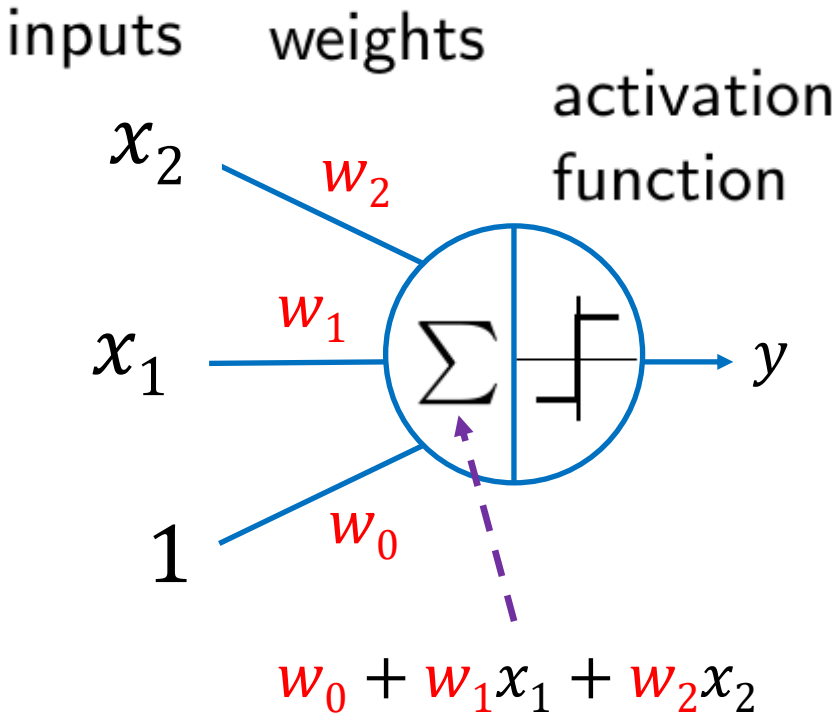
Interesting readings:
- Ng, A., & Jordan, M. (2001). On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, *14*.
- https://stackoverflow.com/questions/879432/what-is-the-difference-between-a-generative-and-a-discriminative-algorithm
- Breiman, L. (2001). Statistical modeling: The two cultures. *Statistical science*, *16*(3), 199-231.

But… what do these machines learn?

- Weights!
- Examples:
  - *Linear regression*

$$y = w_0 + w_1 x_1 + w_2 x_2$$

  - *Perceptron*

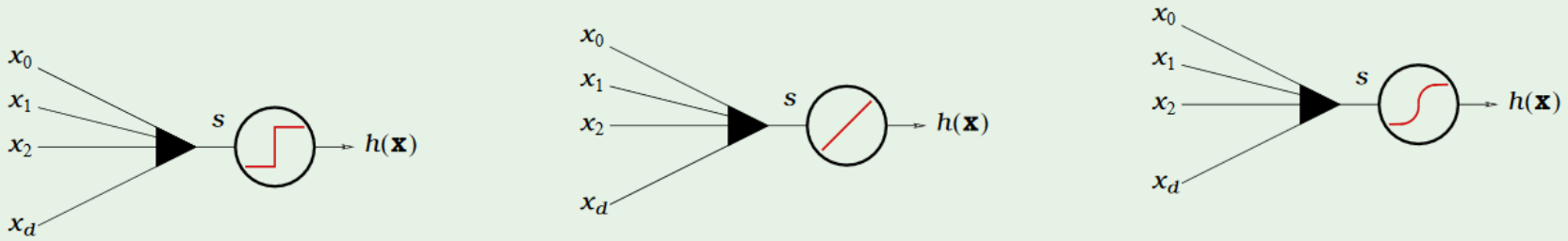


Note: machine learning models can be extremely different. For instance, in k-nearest neighbors (KNN) there is no training step (even if, given some data, it allows you to make predictions). The only "training" that happens for KNN, is memorizing the data (creating a local copy), so you can do a search and majority vote.

But… what do these machines learn?

$$s = \sum_{i=0}^{d} w_i x_i$$

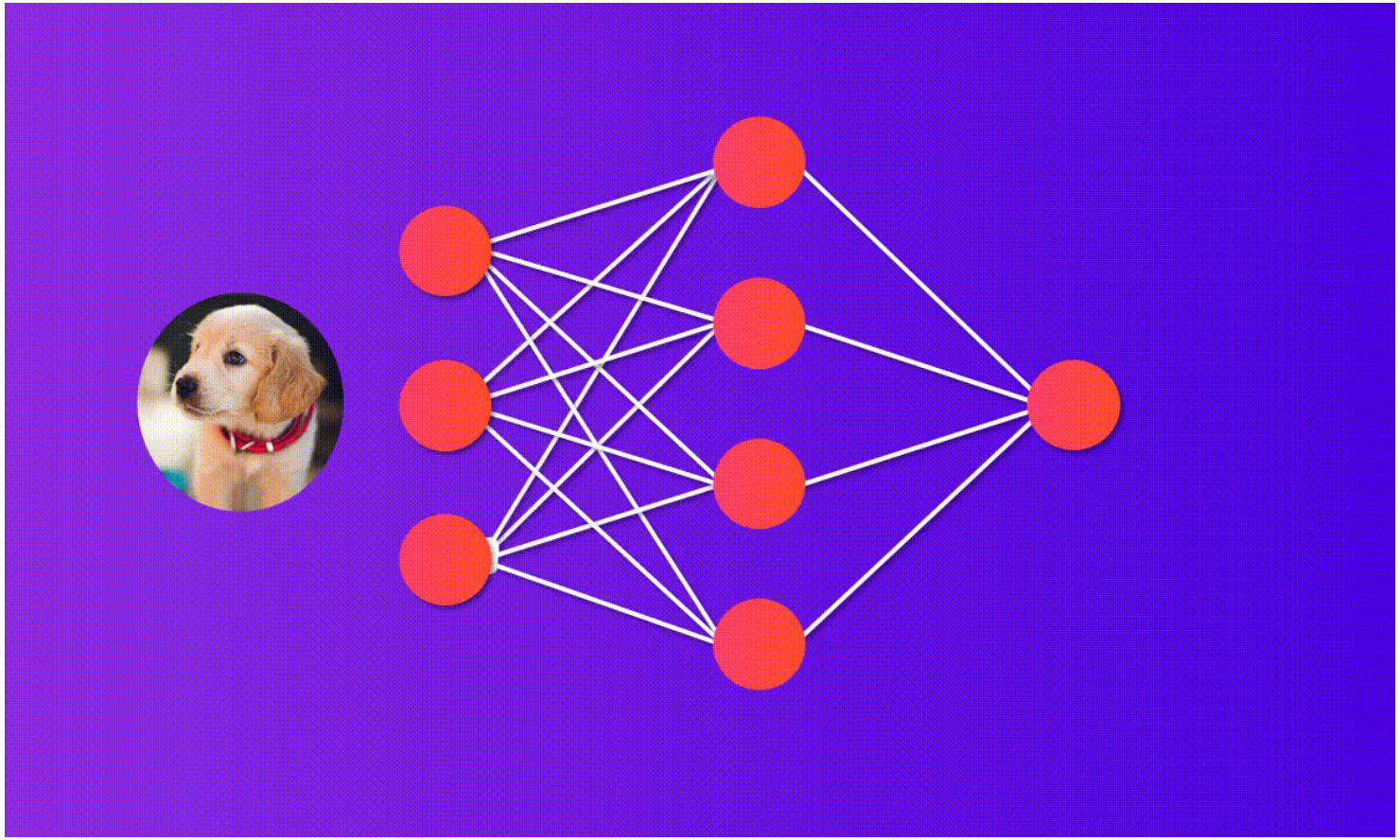
linear classification	linear regression	logistic regression
$h(\mathbf{x}) = \text{sign}(s)$	$h(\mathbf{x}) = s$	$h(\mathbf{x}) = \theta(s)$



http://work.caltech.edu/slides/slides09.pdf (slide 10)

How do they learn?

Showing them **lots of examples** and **progressively improving** as we go!!!!

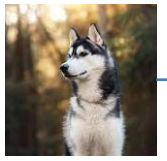


We modify the weights to minimize the output error

How do they learn?

Input example → **Expected Output**

 → Cat

 → Dog

 → Cat

 → Cat

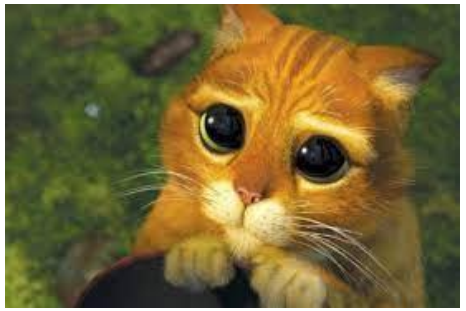
 → Dog

 → Cat

This type of learning is known as **Supervised Learning**

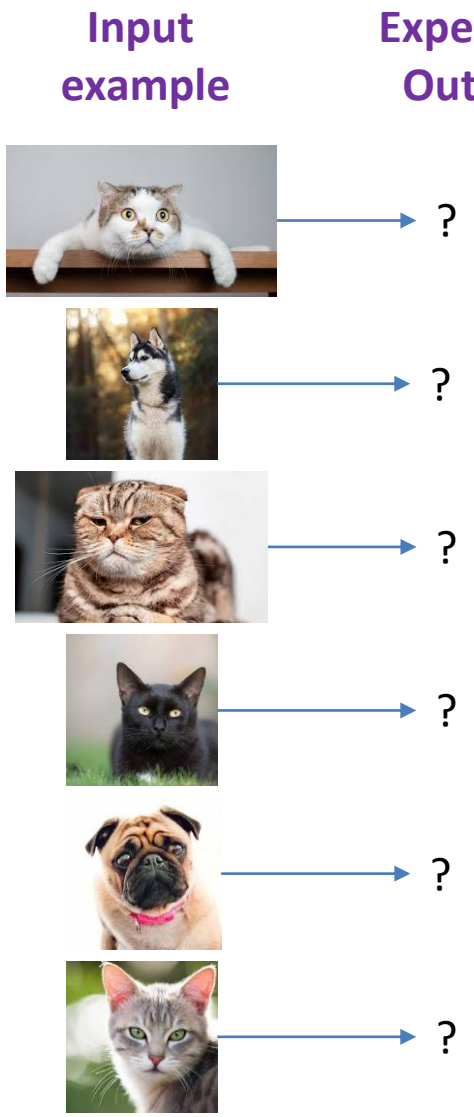
Once your machine is trained

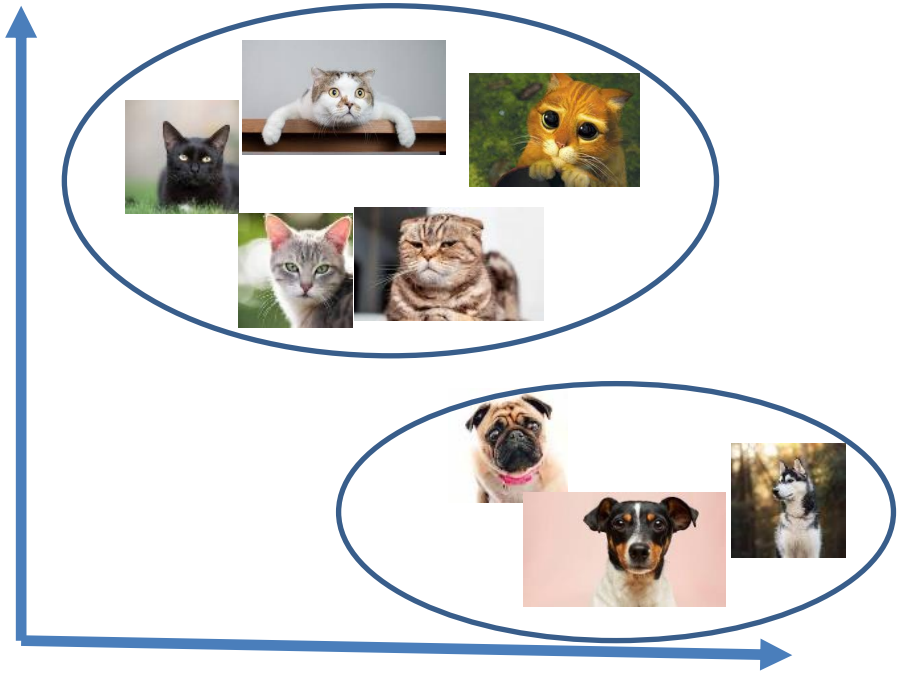
Cat or Dog?



How do they learn?

Unsupervised Learning

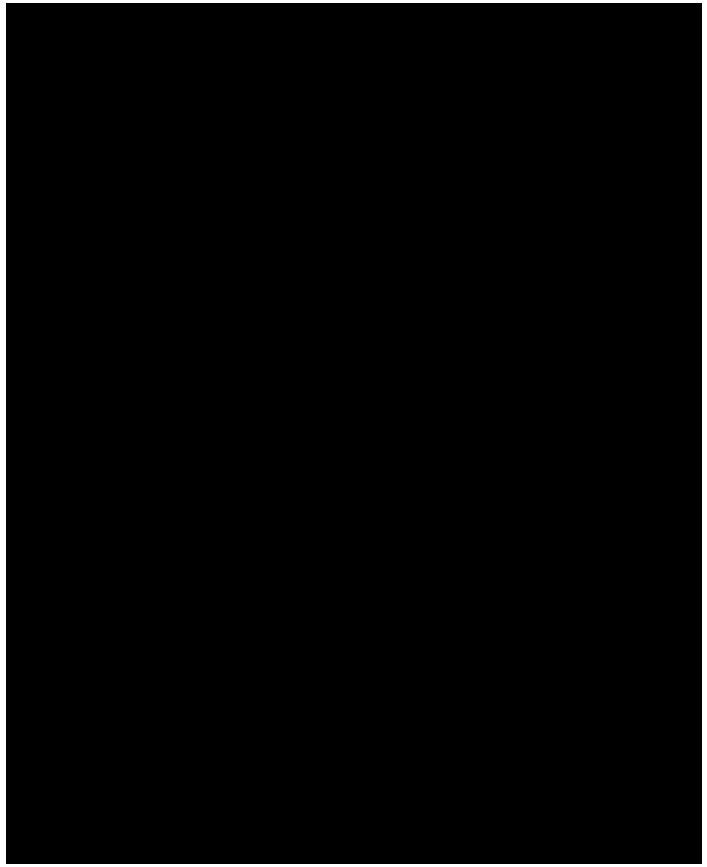




Similar patterns can be found among the data that allow them **to be grouped/clustered** (although we do not know what is a cat and what is a dog)

How do they learn?

Reinforcement Learning



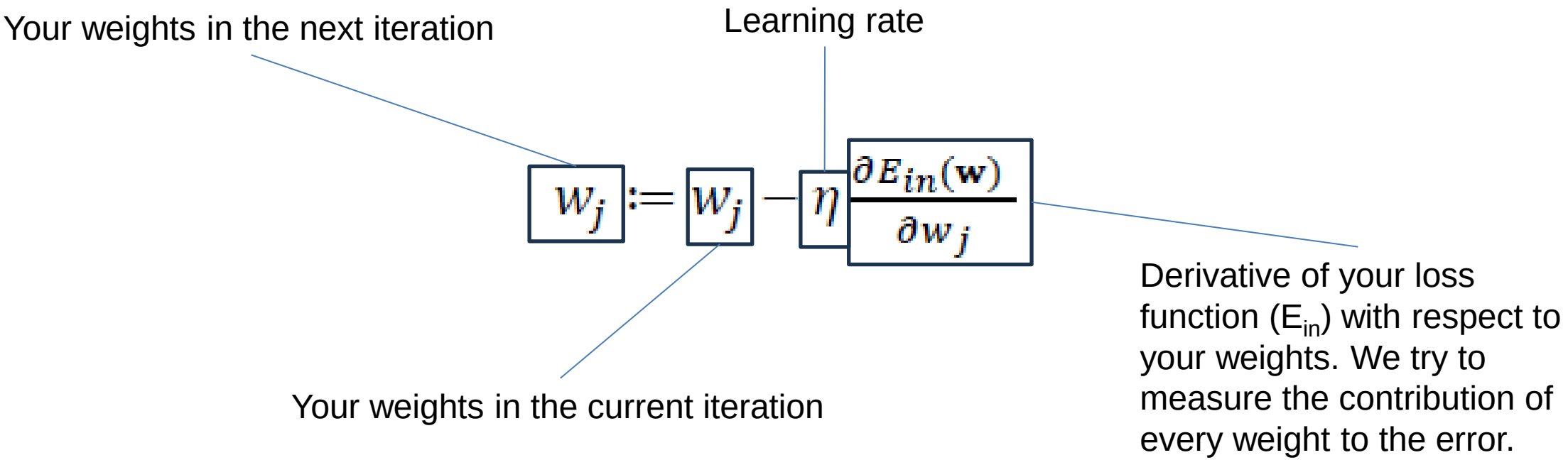
Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., & Riedmiller, M. (2013). Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

https://www.deepmind.com/publications/playing-atari-with-deep-reinforcement-learning

We have a **generic signal of whether we are doing it right** (reward) **or wrong** (punishment). The **machine learns a policy/strategy** of action

Ok, but… how do you learn those weights?

- Usually, employing some version of **gradient descent**

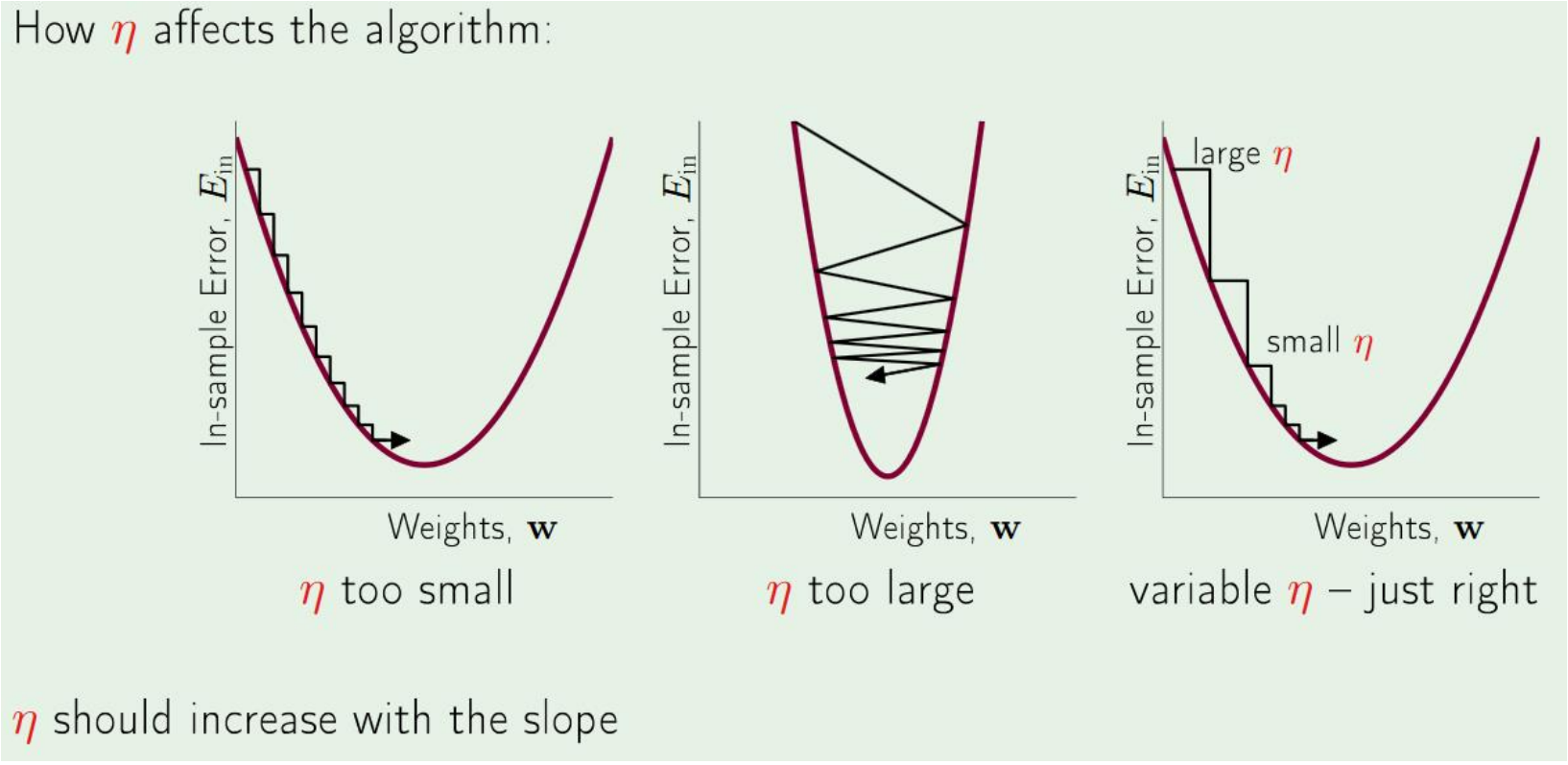


Training is posed as an optimization problem!!!

Ok, but… how do you learn those weights?

- Usually, employing some version of **gradient descent**

$$w_j := w_j - \eta \frac{\partial E_{in}(\mathbf{w})}{\partial w_j}$$



http://work.caltech.edu/slides/slides09.pdf
(slide 21)

Ok, but… how do you learn those weights?

$$w_j := w_j - \eta \frac{\partial E_n(\mathbf{w})}{\partial w_j}$$

Loss Function

"The function we want to minimize or maximize is called the objective function or criterion. When we are minimizing it, we may also call it the cost function, loss function, or error function."
("Deep Learning", Ian J. Goodfellow, Yoshua Bengio and Aaron Courville, MIT Press, 2016 (page 82))

Regression (Mean Squared Error)

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

(Binary) Classification (Cross-entropy loss)

$$E = -\frac{1}{N}\sum_{N}^{i=1}[y_i(log(\hat{y}_i)) + (1 - y_i)log(1 - \hat{y}_i)]$$

Note:
$E = 0$
if $y = 1$ and $\hat{y} = 1$. But as $\hat{y} \to 0$, $E \to \infty$
if $y = 0$ and $\hat{y} = 0$. But as $\hat{y} \to 1$, $E \to \infty$

Two of the most common ones!

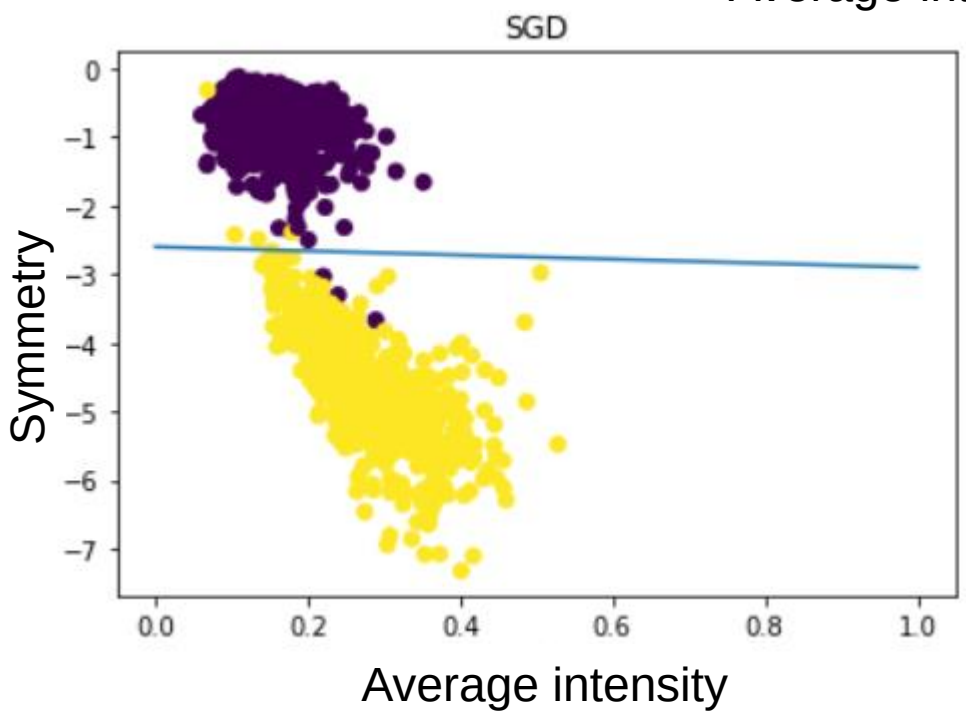
Regarding the bias

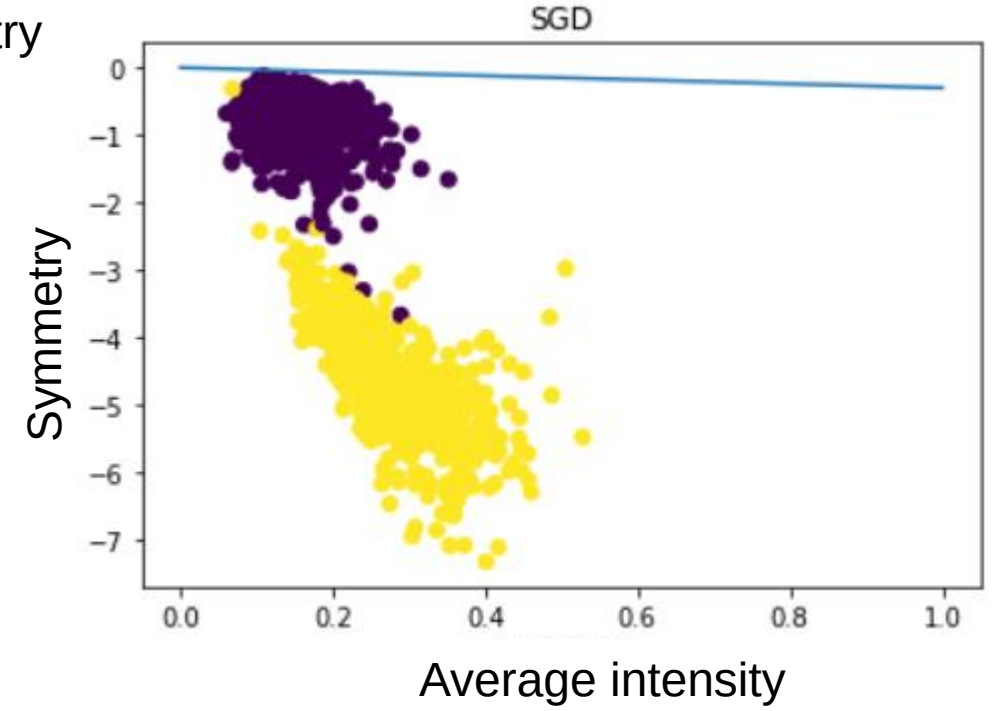
bias or intercept. Without it, our decision boundary is forced to pass through (0,0)

$$y = w_0 + w_1 x_1 + w_2 x_2$$

Average intensity

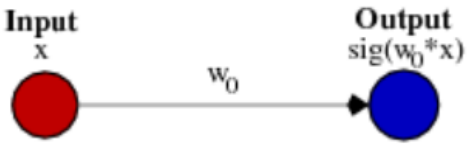
Symmetry



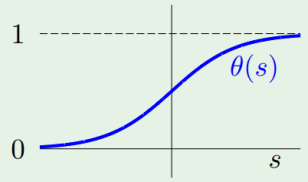


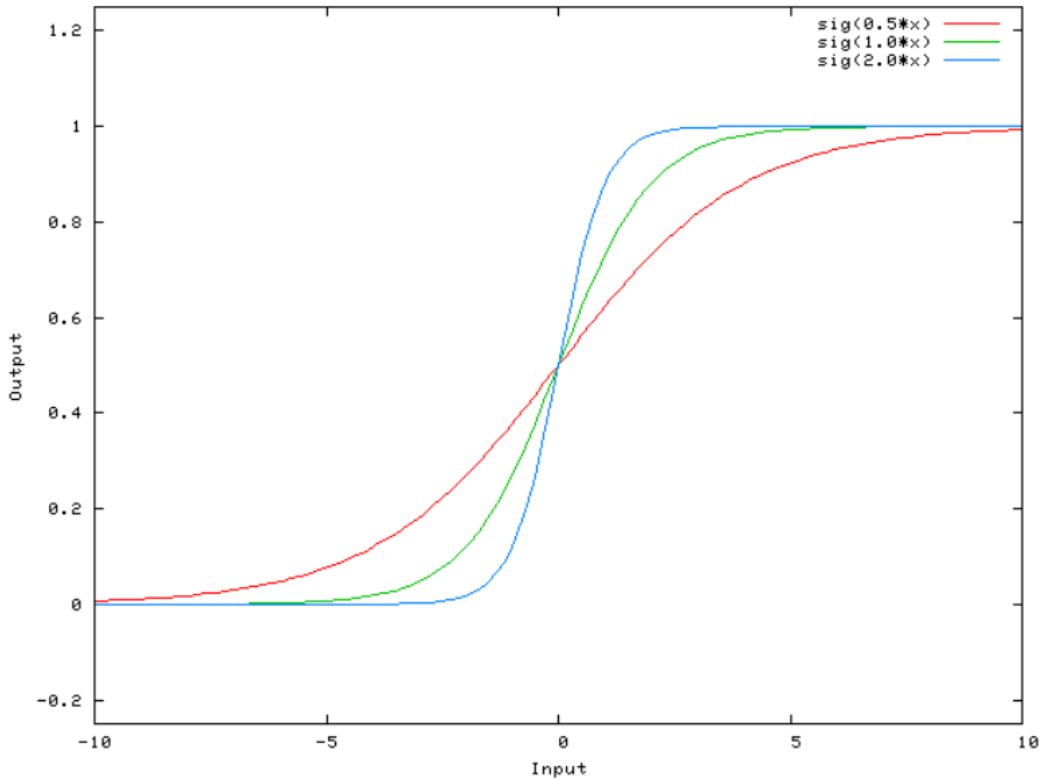
We lose flexibility in our model: we force our decision boundary to go through the origin

Regarding the bias



$$\theta(s) = \frac{e^s}{1+e^s}$$

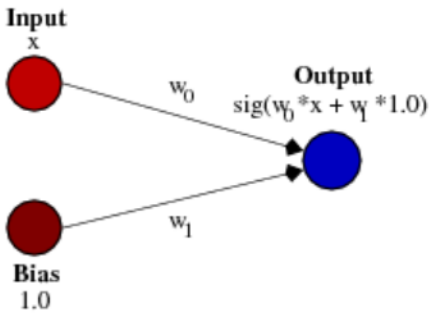




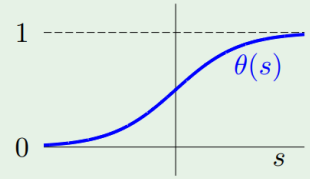
In this case, without bias, w_0 only modulates the slope of the sigmoid. How can we shift the curve? Without bias we cannot…

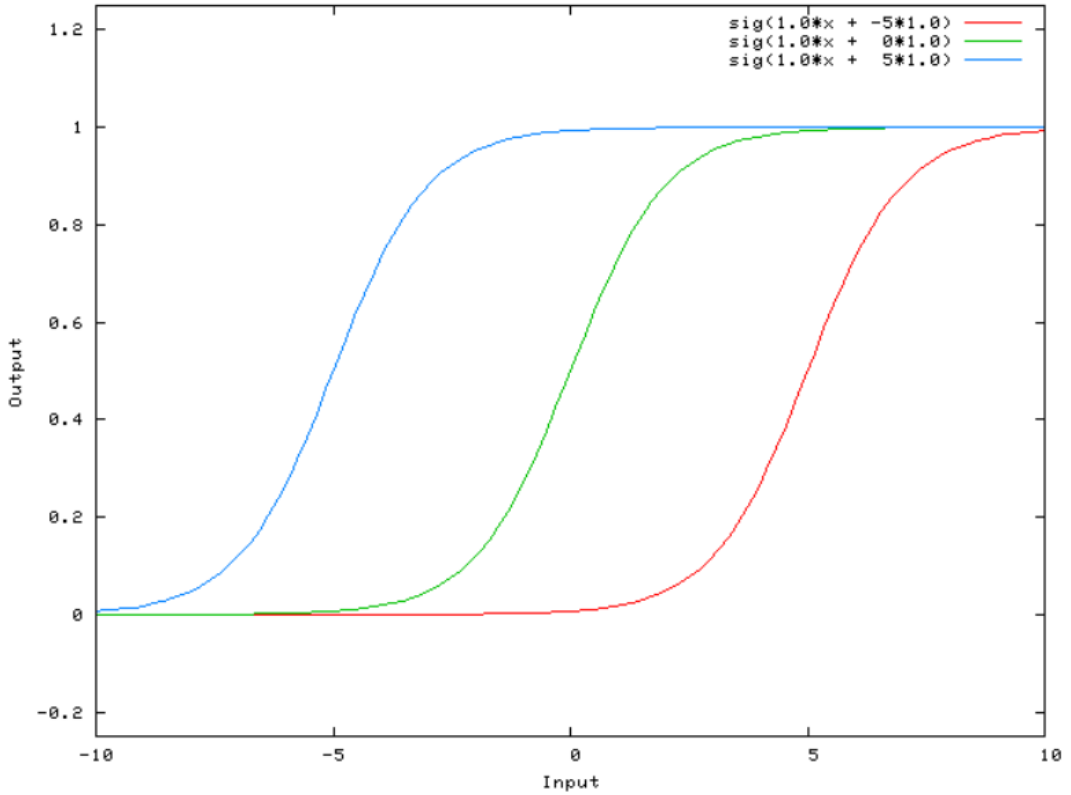
"What is the role of the bias in neural networks?“
(https://stackoverflow.com/questions/2480650/what-is-the-role-of-the-bias-in-neural-networks)

Regarding the bias



$$\theta(s) = \frac{e^s}{1+e^s}$$

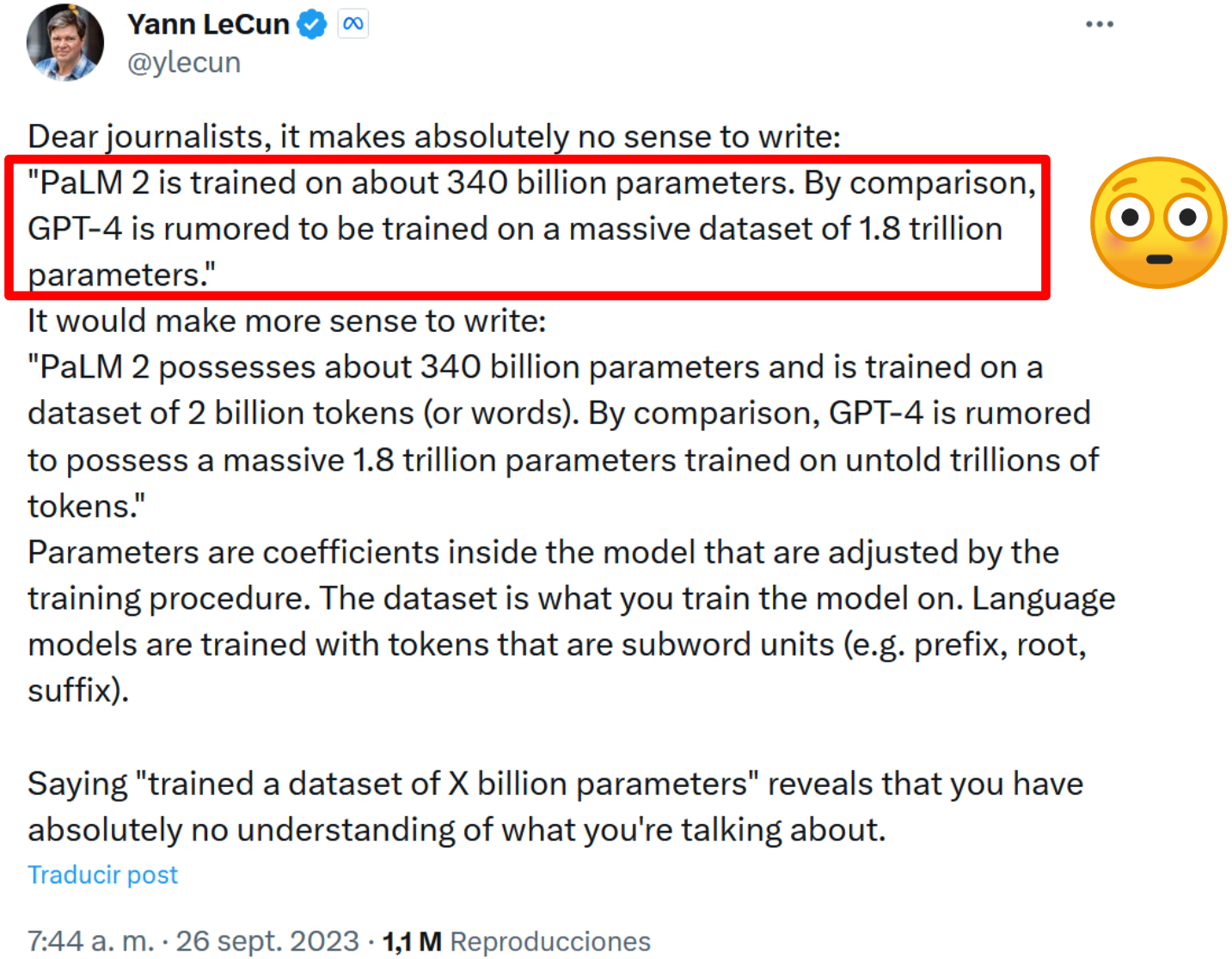




The bias provides richness and flexibility in terms of functions that can be learned.

"What is the role of the bias in neural networks?“
(https://stackoverflow.com/questions/2480650/what-is-the-role-of-the-bias-in-neural-networks)

Importance of understanding the essential concepts



Yann LeCun ∞
@ylecun

Dear journalists, it makes absolutely no sense to write:
"PaLM 2 is trained on about 340 billion parameters. By comparison, GPT-4 is rumored to be trained on a massive dataset of 1.8 trillion parameters."
It would make more sense to write:
"PaLM 2 possesses about 340 billion parameters and is trained on a dataset of 2 billion tokens (or words). By comparison, GPT-4 is rumored to possess a massive 1.8 trillion parameters trained on untold trillions of tokens."
Parameters are coefficients inside the model that are adjusted by the training procedure. The dataset is what you train the model on. Language models are trained with tokens that are subword units (e.g. prefix, root, suffix).

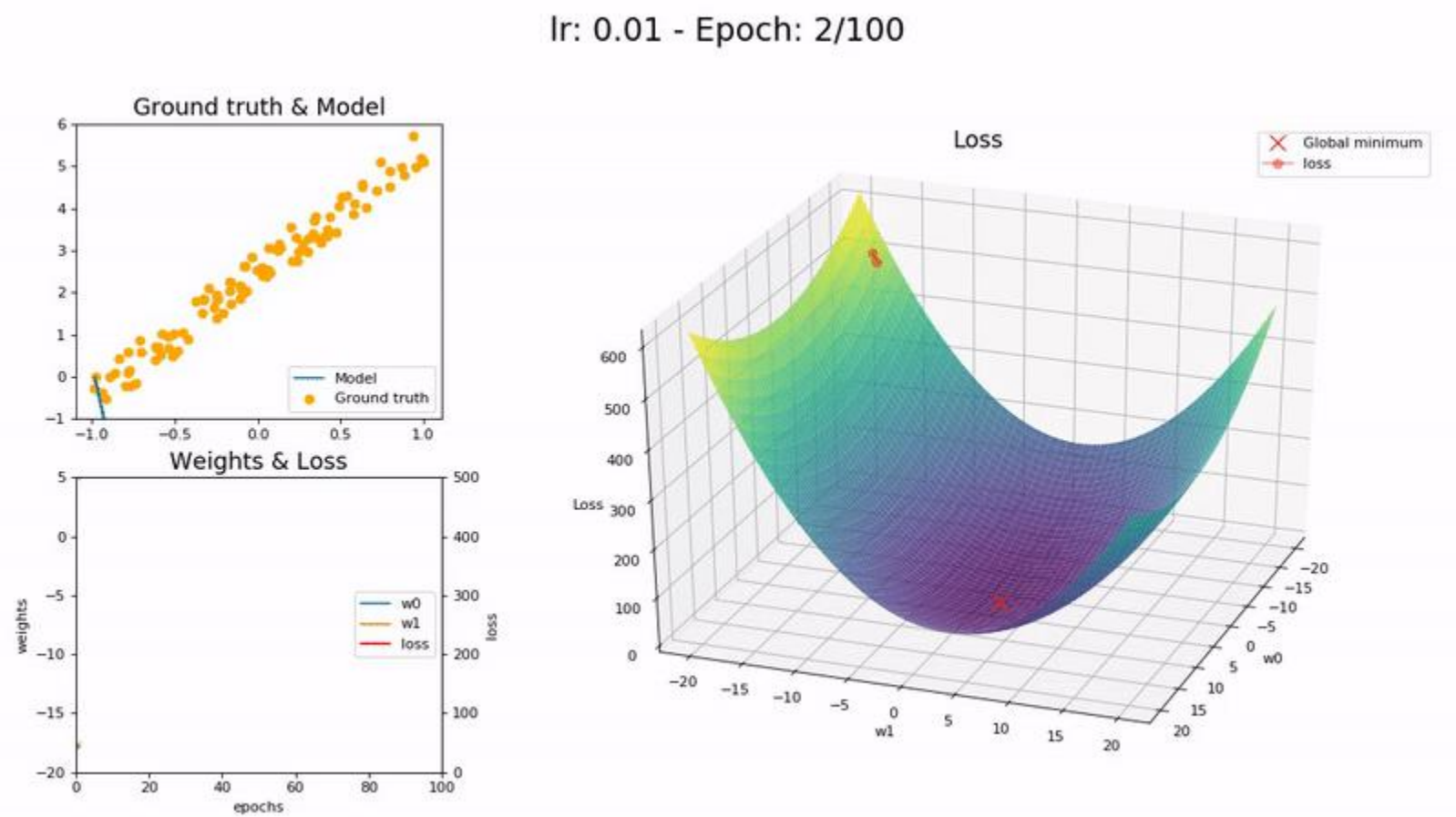
Saying "trained a dataset of X billion parameters" reveals that you have absolutely no understanding of what you're talking about.

Traducir post

7:44 a. m. · 26 sept. 2023 · **1,1 M** Reproducciones

Importance of learning rate

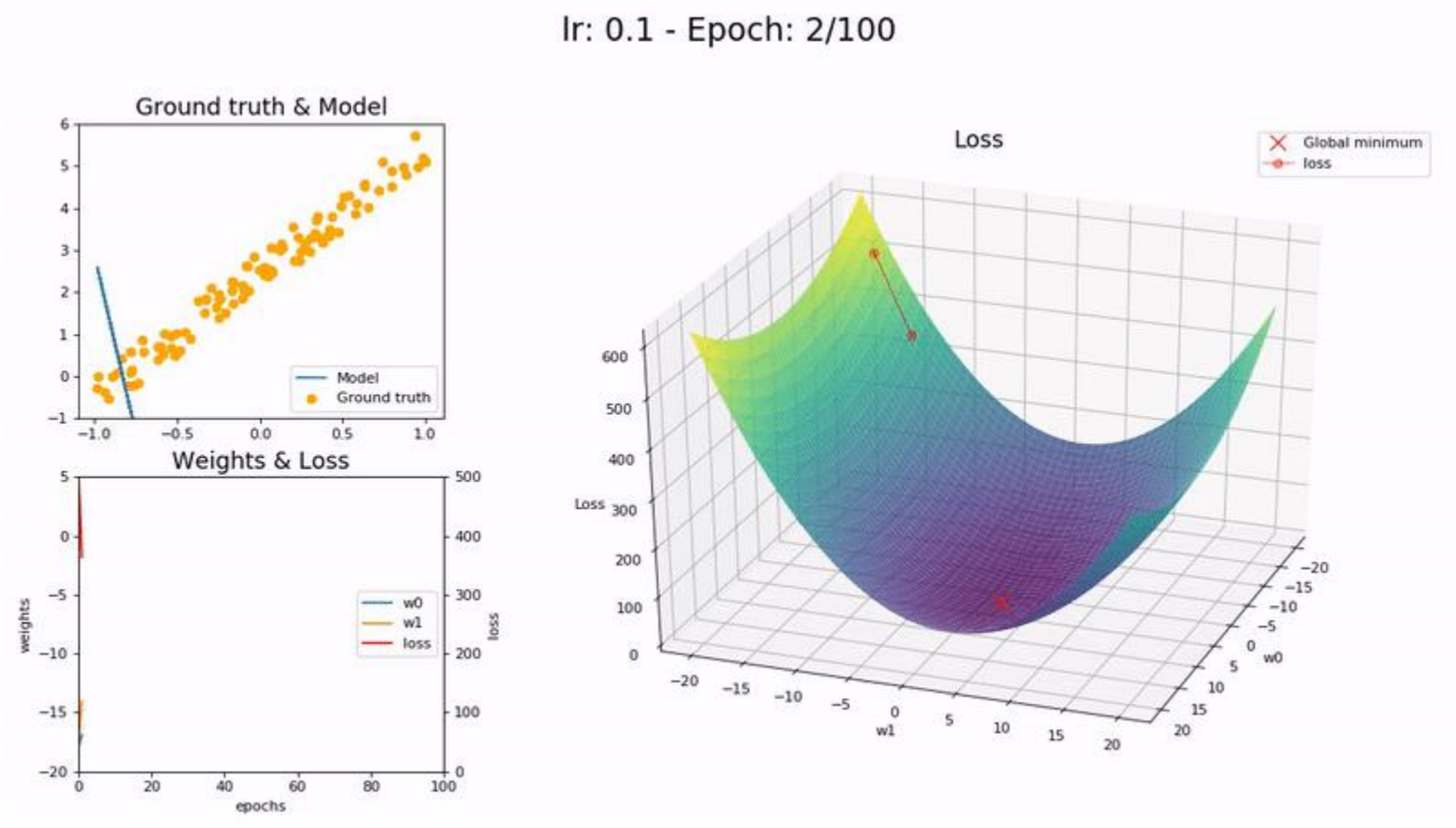
Learning rate too small: the optimization of the model progresses too slowly. In fact, it does not reach the optimum at the indicated epochs.



https://iconof.com/1cycle-learning-rate-policy/

Importance of learning rate

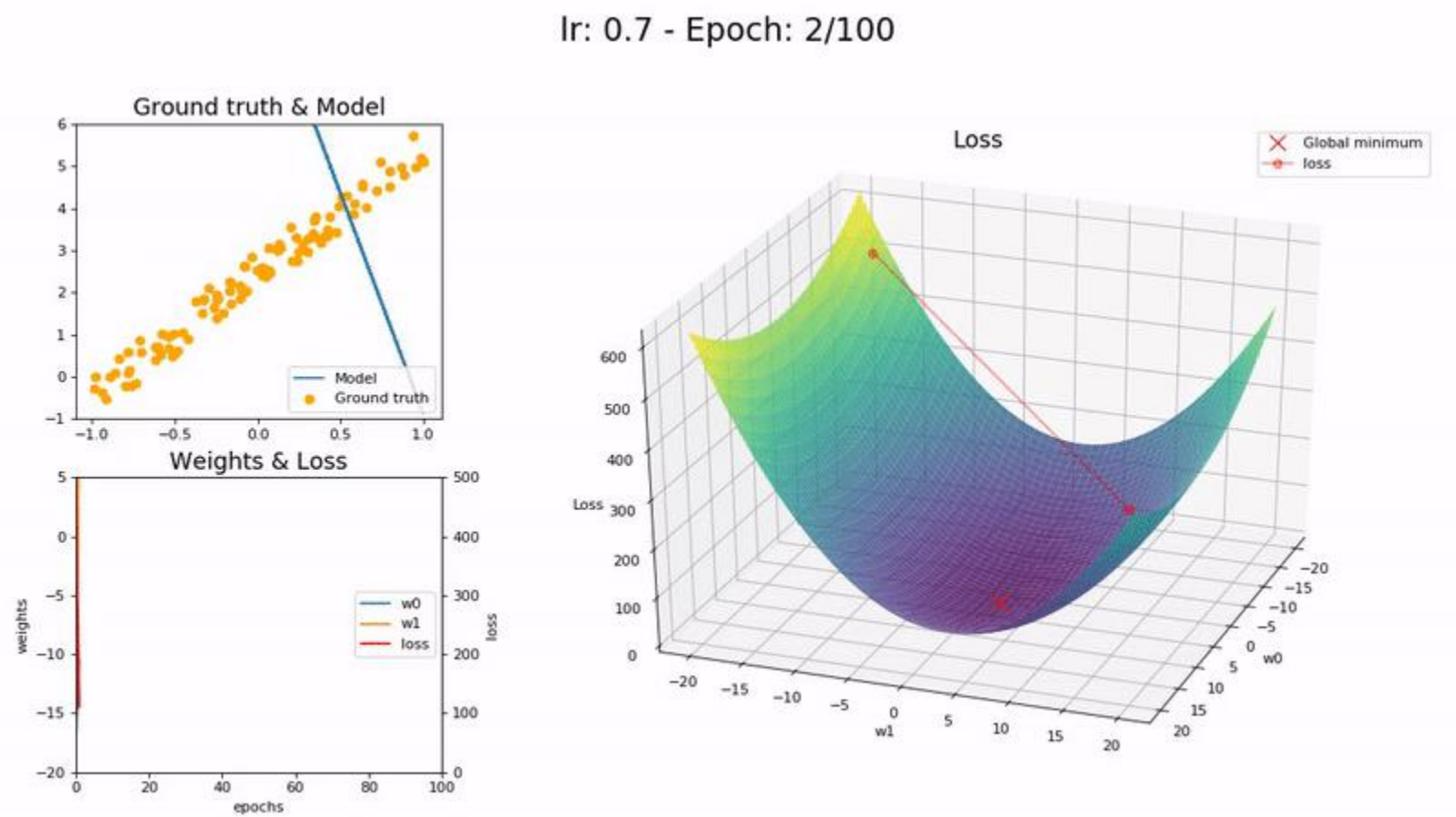
Adequate learning rate: the model converges at the right epochs.



https://iconof.com/1cycle-learning-rate-policy/

Importance of learning rate

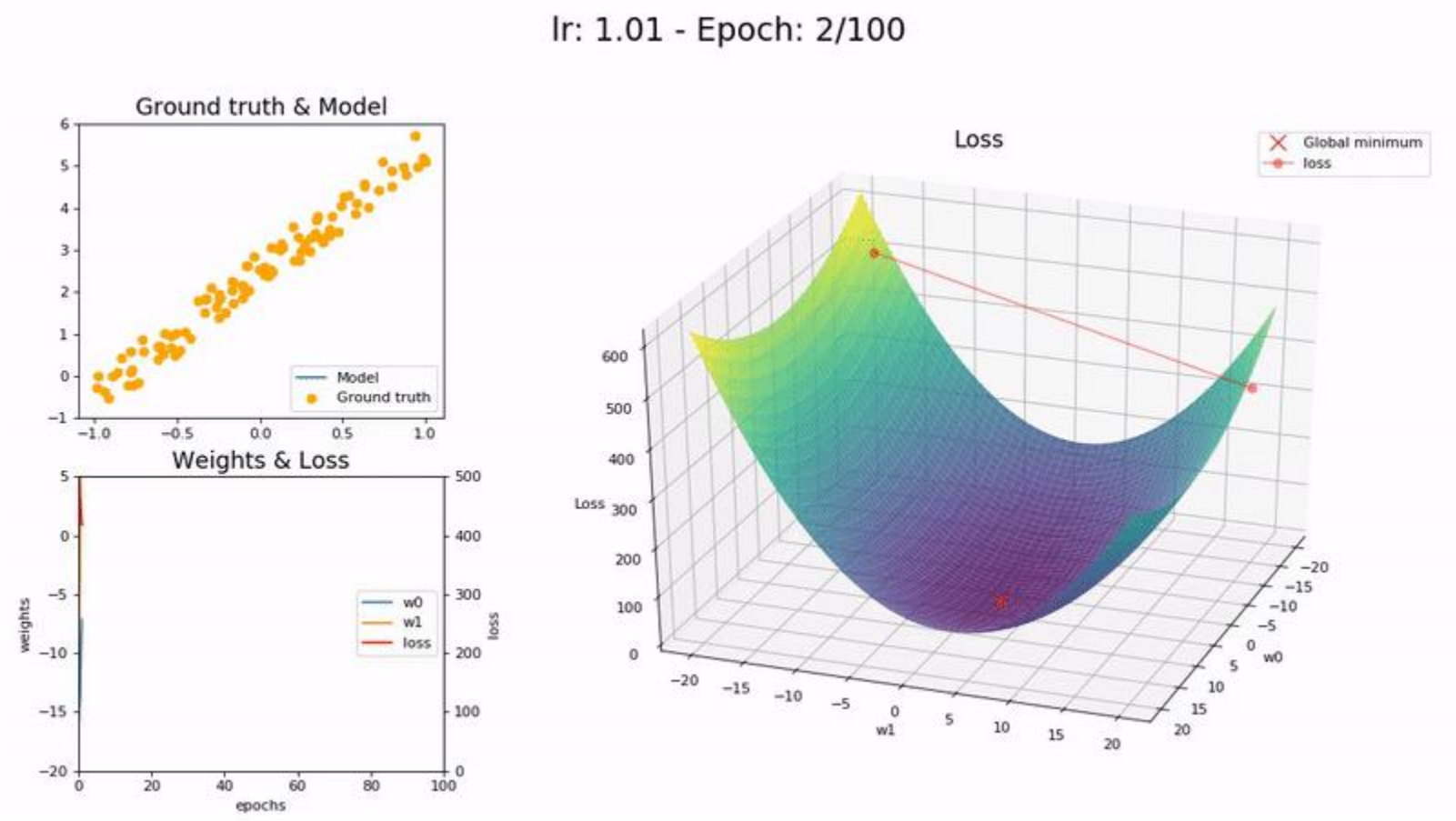
Optimal learning rate: the model reaches the optimum very quickly (less than 10 epochs).



https://iconof.com/1cycle-learning-rate-policy/

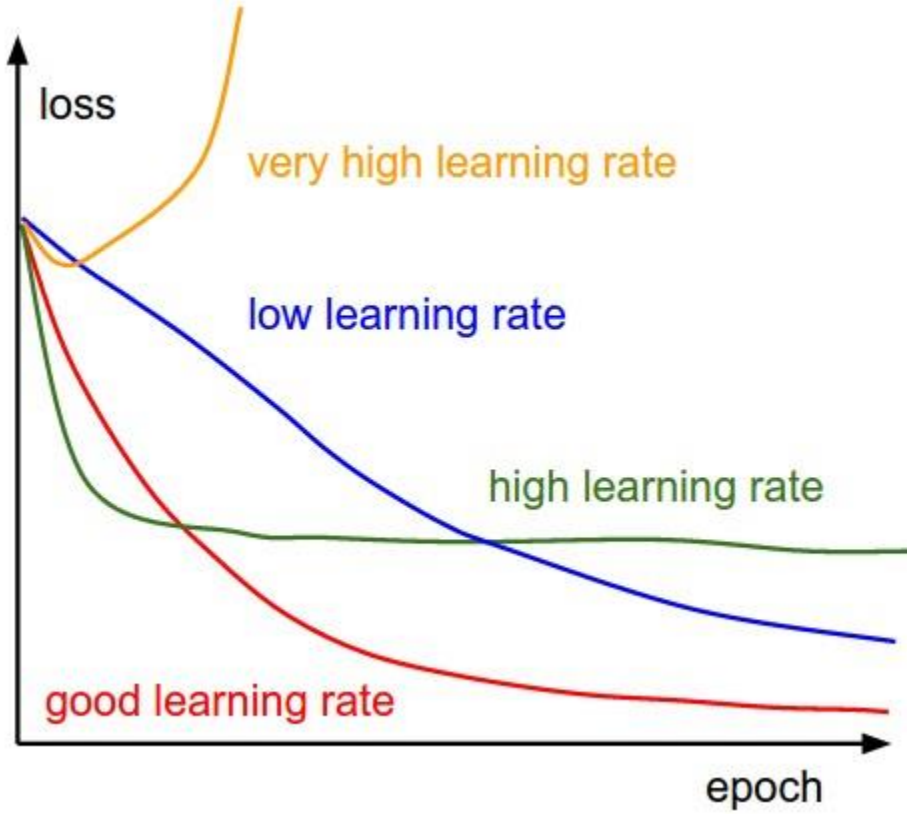
Importance of learning rate

Learning rate too high: the model diverges.



https://iconof.com/1cycle-learning-rate-policy/

Importance of learning rate



https://cs231n.github.io/neural-networks-3/

Interesting references about optimization:

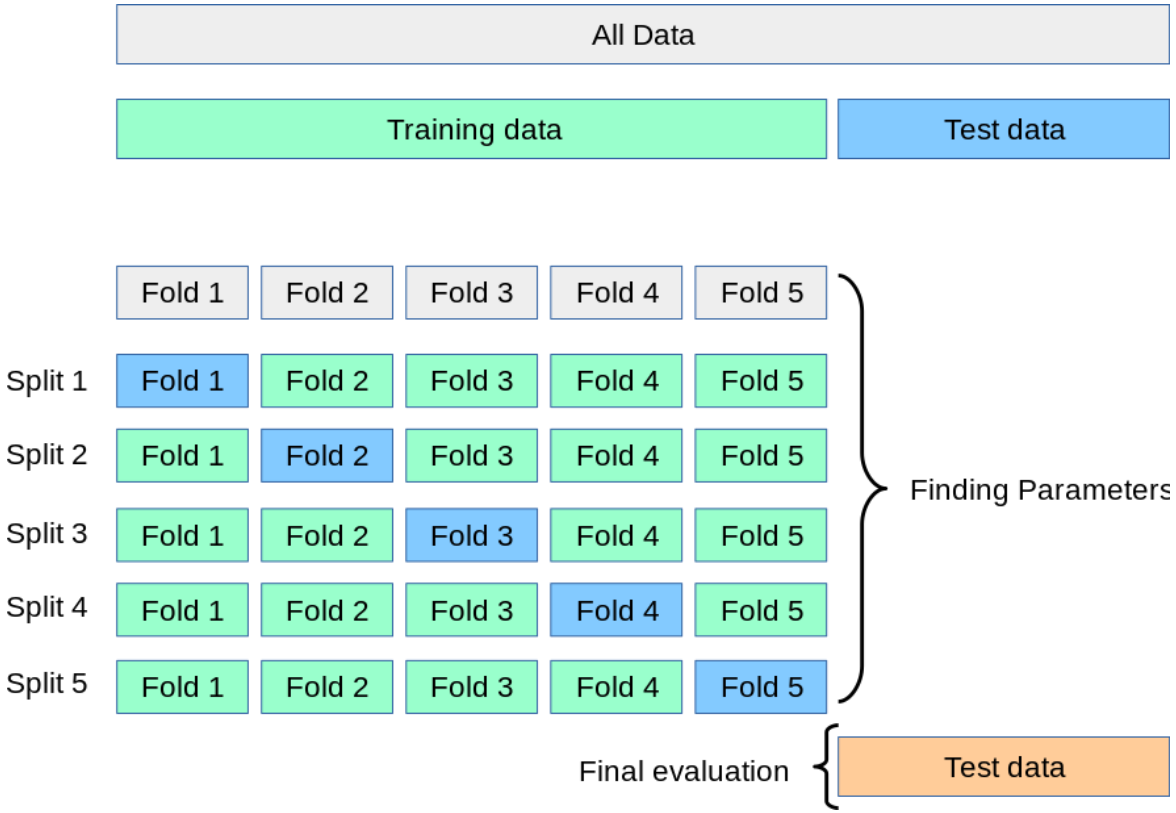
- https://www.ruder.io/optimizing-gradient-descent/
- https://d2l.ai/chapter_optimization/index.html

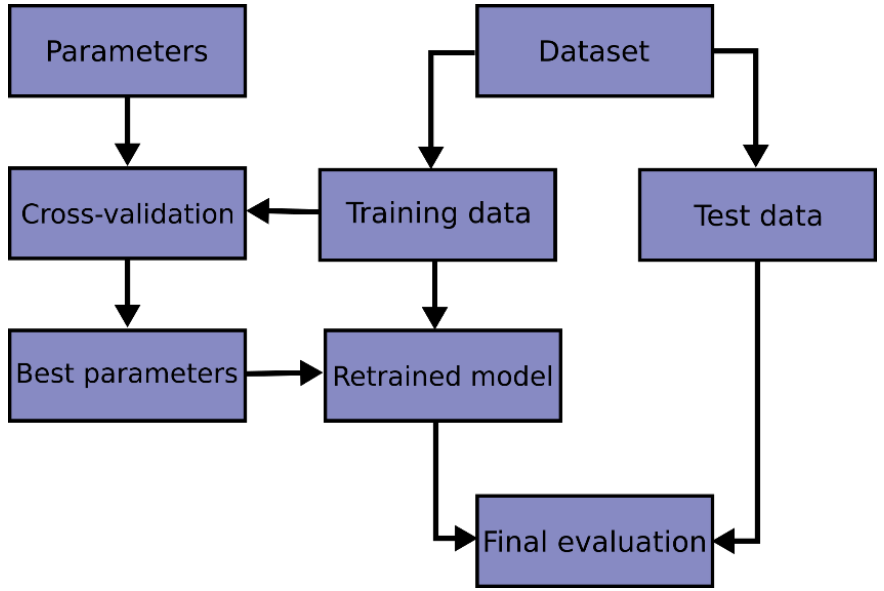
Experimental validation protocol

- Training and testing on the same data 

- Hold-out:
  - Single training-test partition
    → very dependent on this particular partition
- Cross-validation:
  - Multiple training-test partitions
- Leave-one-out (= most extreme case of cross-validation):
  - As many training-test partitions as training examples
    → you train with all your examples (but one) and test on the remaining one

Experimental validation protocol





https://scikit-learn.org/stable/modules/cross_validation.html

Interesting readings:
https://neptune.ai/blog/cross-validation-in-machine-learning-how-to-do-it-right
https://datascience.stackexchange.com/questions/108792/why-is-the-k-fold-cross-validation-needed

Experimental validation protocol

What is the most common scenario in Deep Learning? 🤔

i. If we have very little data → ***leave-one-out***

ii. If we have models whose training is very expensive → ***hold-out***

iii. If we consider that data are relatively scarce we could avoid the separation of a test set → use of **CV on all available data, approximation of E_{out} with E_{cv}**

iv. In the remaining cases, the most common scenario is to do what was shown before → separation of training and testing, use of **CV on the training set, approximation of E_{out} with E_{test}**

Experimental validation protocol

"We mostly have large datasets when it is not worth the trouble to do something like k-fold cross-validation. We just use a train/valid/test split. Cross-validation becomes useful when the dataset is tiny (like hundreds of examples), but then you can't typically learn a complex model.“

(Yoshua Bengio, 2016, https://qr.ae/pKyHsD)

Data preprocessing

Manipulations on the initial data to obtain the training (and test) set.

There are no universal rules. Each problem and variable/feature requires a different preprocessing. Here we provide some general references.

- Remove data without variability
  - They're not discriminant
- Remove extreme/atypical data (*outliers*).
  - Very careful! In case of doubt, better not to delete anything!
- What do we do with missing data?
  - *Missing data imputation*
- Dimensionality reduction (e.g. PCA)
  - The goal should be to reduce the number of variables (i.e., the complexity of your problem) while maintaining relevant information from the original data.
- Data transformations
  - Certain transformations in the data can allow a nonlinearly separable problem to become linearly separable. See, for instance, https://work.caltech.edu/slides/slides03.pdf (slides 19-23) or "Solving XOR with a single Perceptron"

Data preprocessing

- Feature/variable scaling
  - All variables are in a similar range of values
  - Goal:
    - Speed up optimization/training (see Andrew Ng's Normalizing Inputs (C2W1L09))
    - Some techniques are more affected by variables scale (KNN, K-Means, SVM,…) than others (DTs, RF,…)
  - Which coefficients should we use to scale the test data (those calculated in test or those calculated before in training)? 🤔
- Data encoding

a) If they are binary or numerical → you can leave them as they are (unless you have some compelling reason to encode/transform them in another way)
   - In general, does it make sense to normalize binary variables? 🤔

b) If they are categorical variables:
   - Nominal → one-hot encoding (red – 1 0 0, green – 0 1 0, blue – 0 0 1)
   - Ordinal → integer encoding (low – 1, medium – 2, high – 3) or one-hot encoding
   - Cyclical (days, months,…) → cosine/sine transformation

Note: some of these ideas lose meaning and usefulness when operating with Deep Learning (end-to-end learning)

Some terminology

- Batch: set of examples in which the training set is divided (to train with and adjust the weights).
  - Batch Gradient Descent. Batch Size = Training Set Size
  - Stochastic Gradient Descent. Batch Size = 1
  - Mini-Batch Gradient Descent. 1 < Batch Size < Training Set Size
- Epoch: each time the network, during training, sees the entire training set.
- Iteration: number of batches required to complete an epoch (number of weights update per epoch).

Example: a dataset with 1000 images, using mini-batches of 100 images
→ it will take 10 iterations to complete a single epoch.

Some terminology

- Loss function: it's used to optimize your model during training (you want to minimize it using an optimizer).
  - Regression: Mean Squared Error (MSE)
  - Binary Classification: Cross-entropy loss
  - Multiclass Classification: Categorical cross-entropy loss
- Evaluation Metric: it's used to evaluate the performance of your trained model.
  - Regression: MSE, Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), Coefficient of Determination (R^2),…
  - Classification: Confusion Matrix, Accuracy, Precision, Recall, F_1 score,…

Data snooping

– **Data Snooping**: it occurs when you (even if very subtly) use test data. We are interested in **solving a problem**, not in **overfitting a specific dataset**.

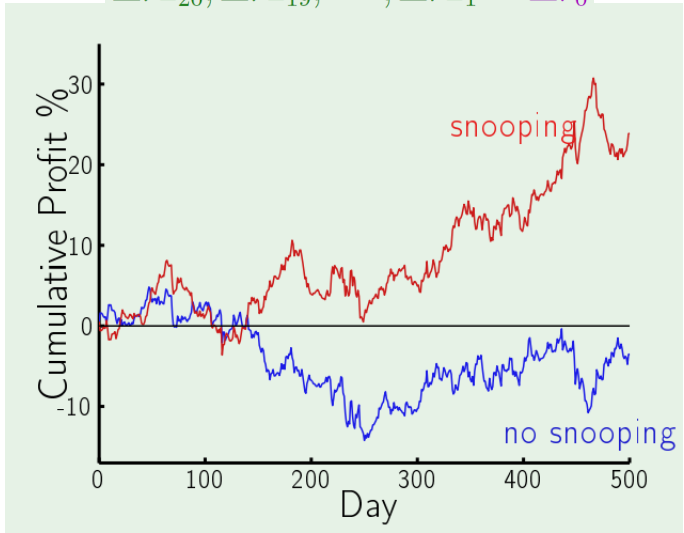
- Example 1:
  - Someone calculates the mean and standard deviation to standardize the data using all the examples (including, without realizing it, the test, which we should not know about at all).
    - » First you would have to divide the data, then scale the training data and, finally, scale the test data with the scaling factors used in training.
- Example 2:
  - Our data, when visualized, we have the impression that it is linearly separable and therefore we decided to use the Perceptron Learning Algorithm (PLA). Most likely, this premise is too strong, too attached to our data, and it is inadvisable to assume it.

Predict US Dollar versus British Pound

Normalize data, split randomly: $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}$

Train on $\mathcal{D}_{\text{train}}$ only, test g on $\mathcal{D}_{\text{test}}$

$\Delta r_{-20}, \Delta r_{-19}, \cdots, \Delta r_{-1} \rightarrow \Delta r_0$



https://home.work.caltech.edu/slides/slides17.pdf

Lecture 17 - Three Learning Principles

https://www.youtube.com/watch?v=EZBUDG12Nr0

Data snooping

- **Data Snooping**: it occurs when you (even if very subtly) use test data. We are interested in **solving a problem**, not in **overfitting a specific dataset**.

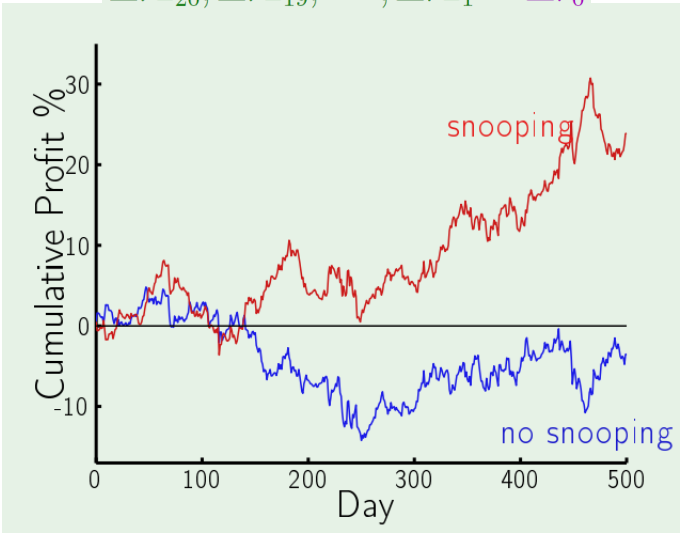
  - Example 3:
    - We have 3 candidate hypotheses (SVM, MLP, RL), and we decide which one is the best by looking at its performance with E_{test} → data snooping! We're making a methodological decision by checking our results on the test set… The test is only run on a single final hypothesis!

Predict US Dollar versus British Pound

Normalize data, split randomly: $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}$

Train on $\mathcal{D}_{\text{train}}$ only, test g on $\mathcal{D}_{\text{test}}$

$\Delta r_{-20}, \Delta r_{-19}, \cdots, \Delta r_{-1} \rightarrow \Delta r_0$



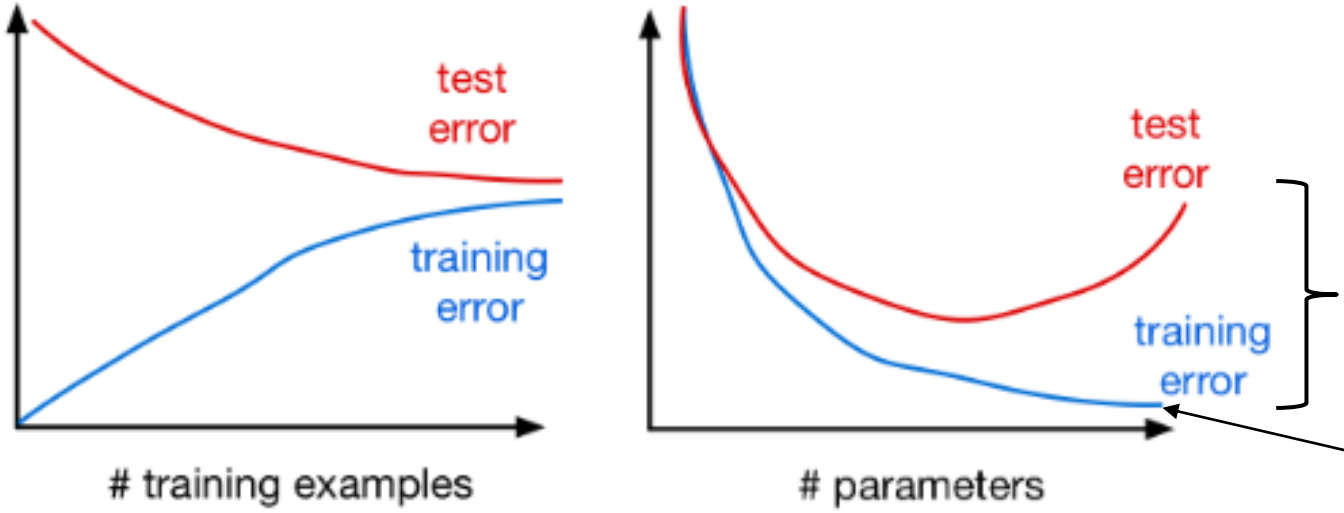
https://home.work.caltech.edu/slides/slides17.pdf

Lecture 17 - Three Learning Principles

https://www.youtube.com/watch?v=EZBUDG12Nr0

Optimization vs Generalization

- Even if training in machine learning is posed as an optimization problem, where we want to minimize a loss function, the most important thing is **generalization**
  - **Ability to adapt properly to new, previously unseen data**



https://www.cs.toronto.edu/~lczhang/321/notes/notes09.pdf (by Roger Grosse)

Ways to improve training

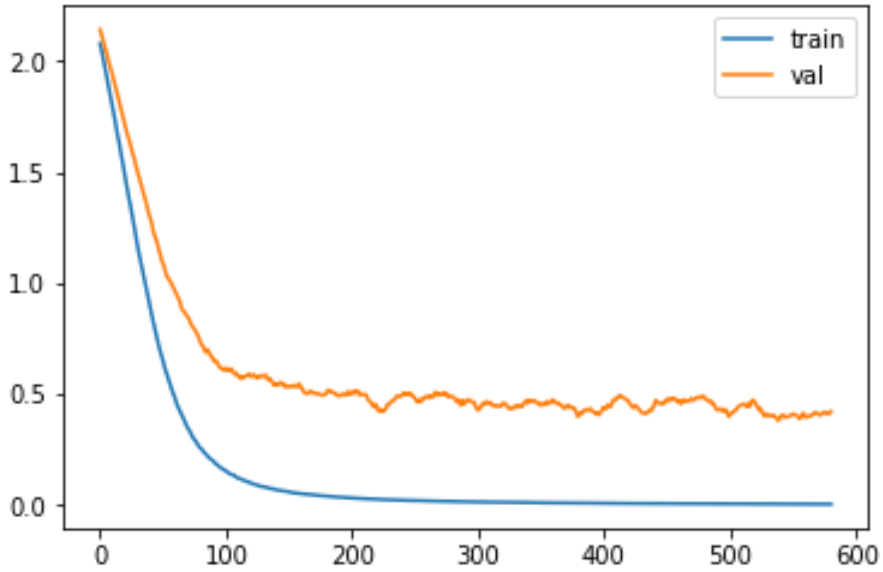
- Deep networks have many parameters → they need large amounts of data to train and have a high risk of overfitting

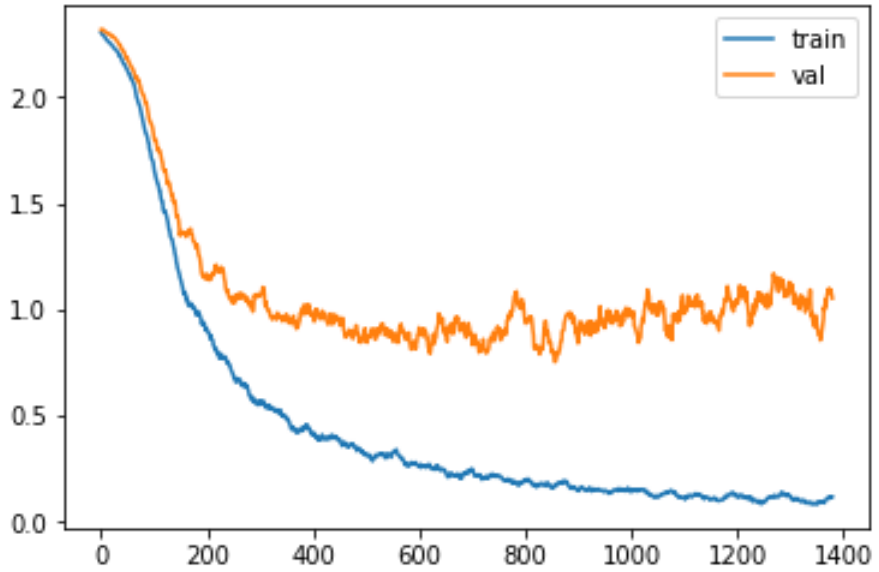
- There are different strategies to improve training (accelerate it and/or regularize it):
  - Early Stopping
  - Loss Penalties
  - Data augmentation
  - Batch normalization and other ways of normalization
  - Dropout
  - Use a different optimizer or select a better learning rate

Regularization
= strategy to reduce overfitting

Overfitting

- The model does not generalize properly
- The model is too complex / we have too little data





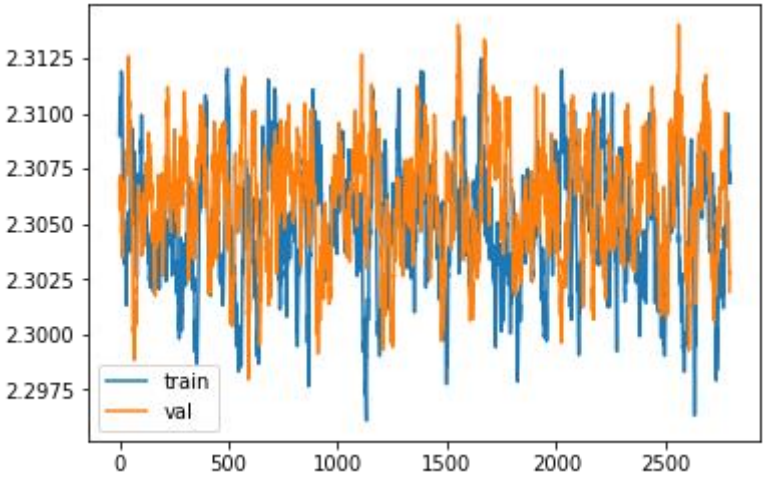
Look at the training curves because they give you a lot of information about what might be happening to you

Overfitting

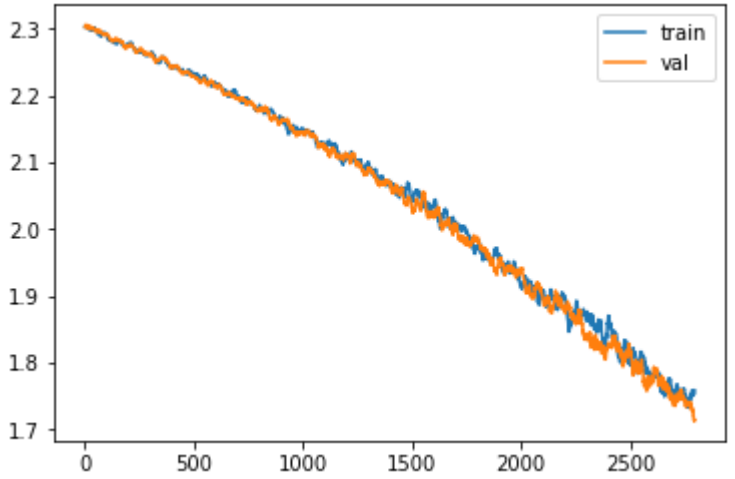
	More parameters/weights	Less parameters/weights
Pros	• Higher capacity model	• Simpler model and, usually, more interpretable
Cons	• More prone to overfit • You need more data to train with • Slower training	• Lower capacity model (less expressive in the number of functions it can approximate)

Tips:
- try overfit first, then try to close the gap between train and val.
- do not use neural network size as regularizer (i.e., usually higher depth (more parameters) is good). Use stronger regularization instead.

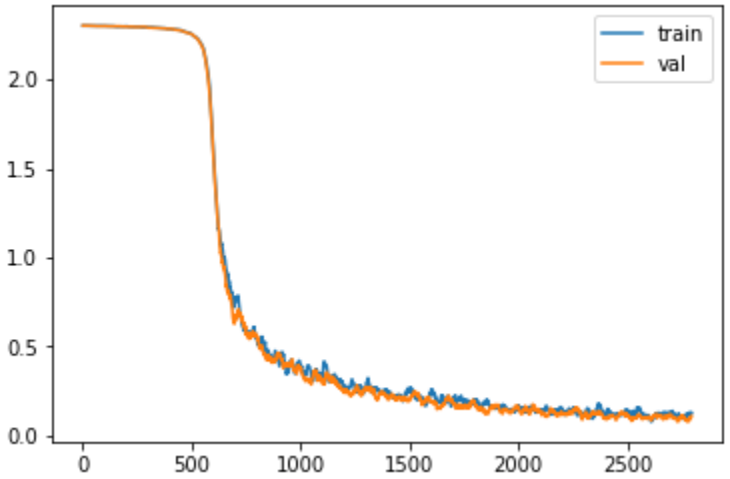
Looking at training curves



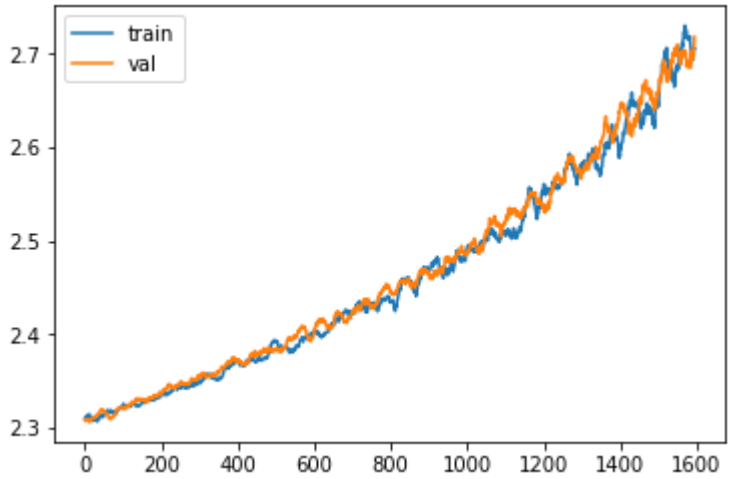
No learning: gradients not applied to weights



Not yet converged: needs more time



Slow start: improper initialization of weights



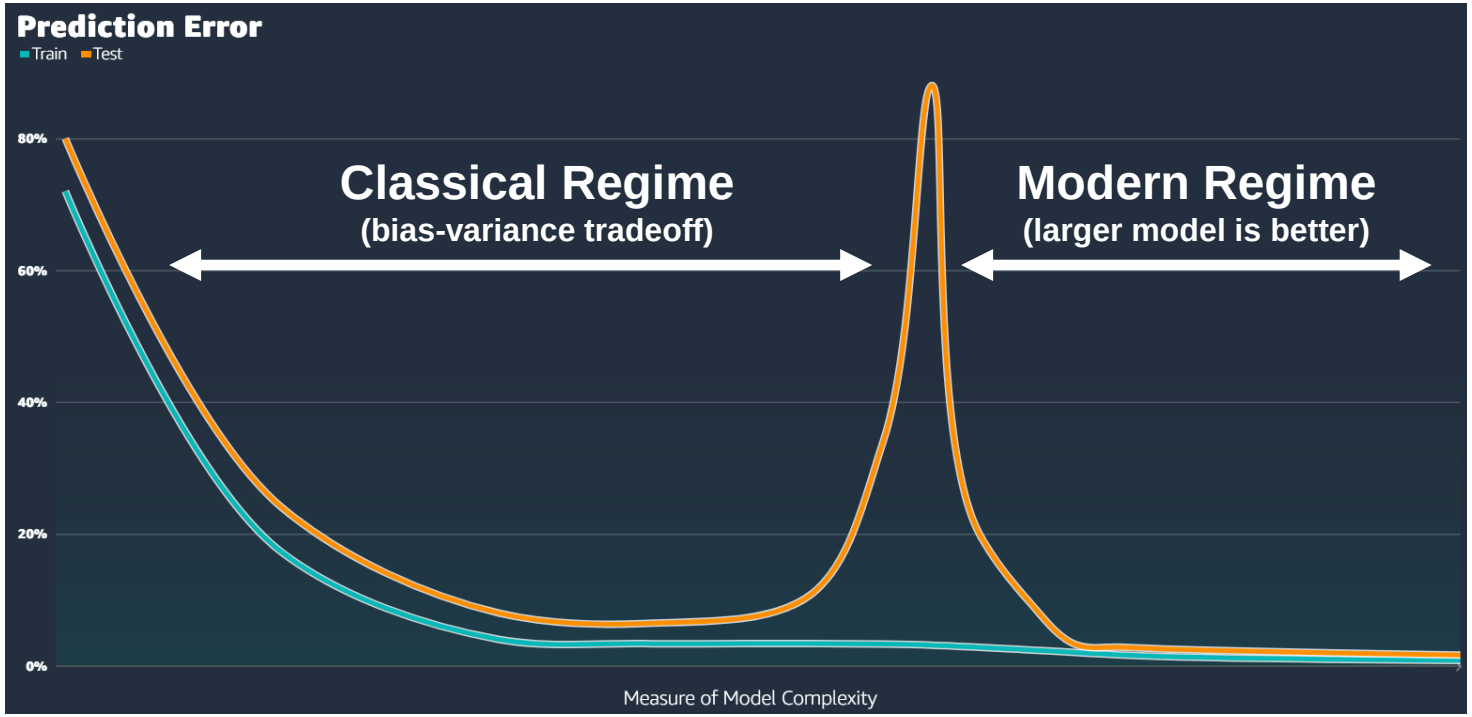
You are probably applying the negative of gradients

[Tips and tricks for tuning NNs (Fei & Nish)]

Looking at training curves

Deep Learning is nowadays a very active research field, and many things assumed to be true are now nuanced.

Deep double descent

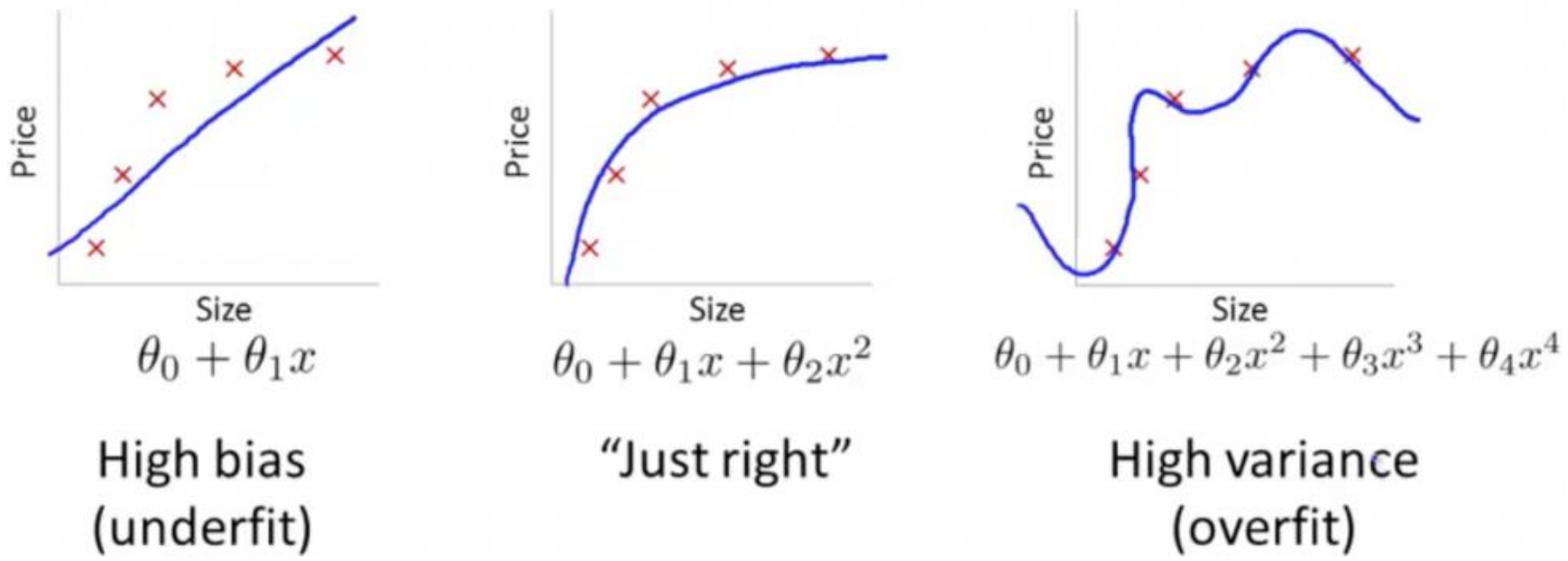


Recommended readings:
https://mlu-explain.github.io/double-descent/
https://openai.com/research/deep-double-descent

Schaeffer, Rylan, et al. "Double Descent Demystified: Identifying, Interpreting & Ablating the Sources of a Deep Learning Puzzle." *arXiv preprint arXiv:2303.14151* (2023).

Bias-variance tradeoff

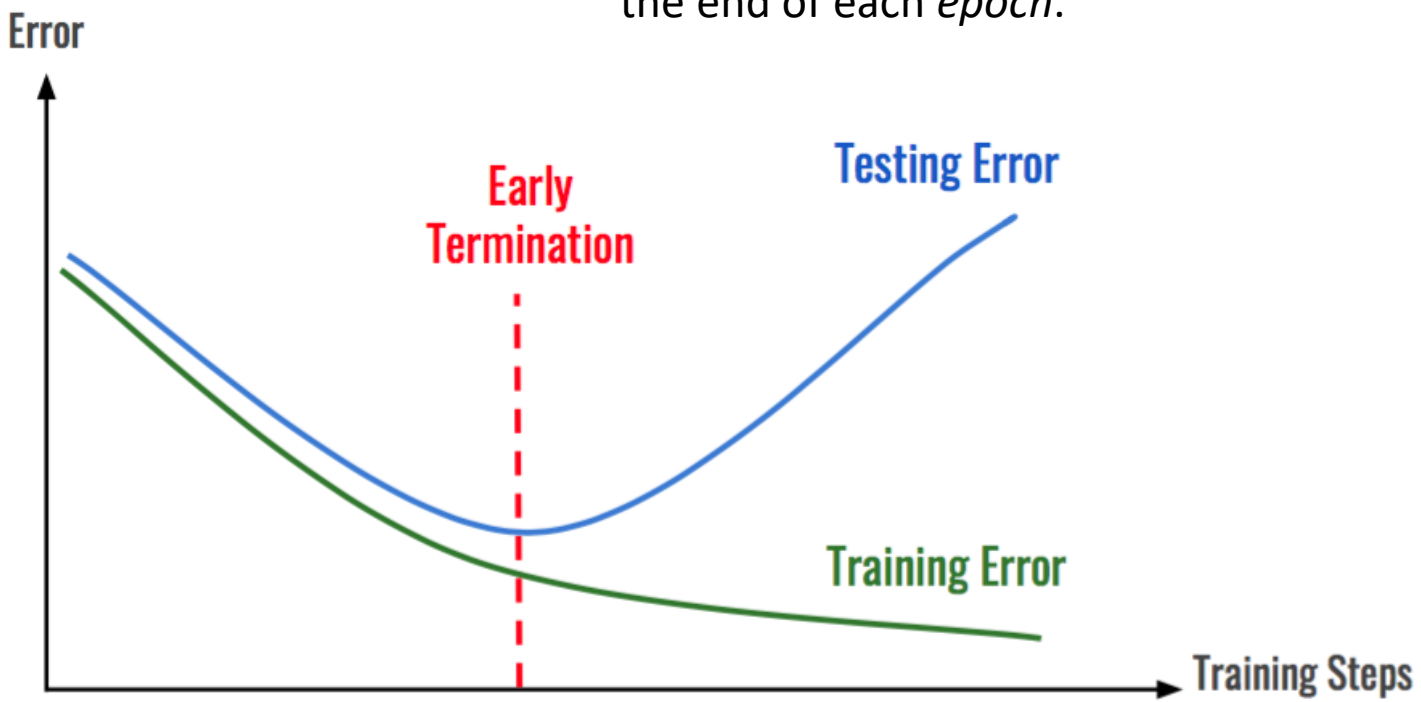
- **Bias:** error from erroneous assumptions in the learning algorithm.
  High bias can cause **underfitting**.
- **Variance:** error from sensitivity to small fluctuations in the training set.
  High variance may result in **overfitting**.



More information:
Andrew Ng: https://www.youtube.com/watch?v=SjQyLhQIXSM
Abu-Mostafa: https://work.caltech.edu/library/080.html

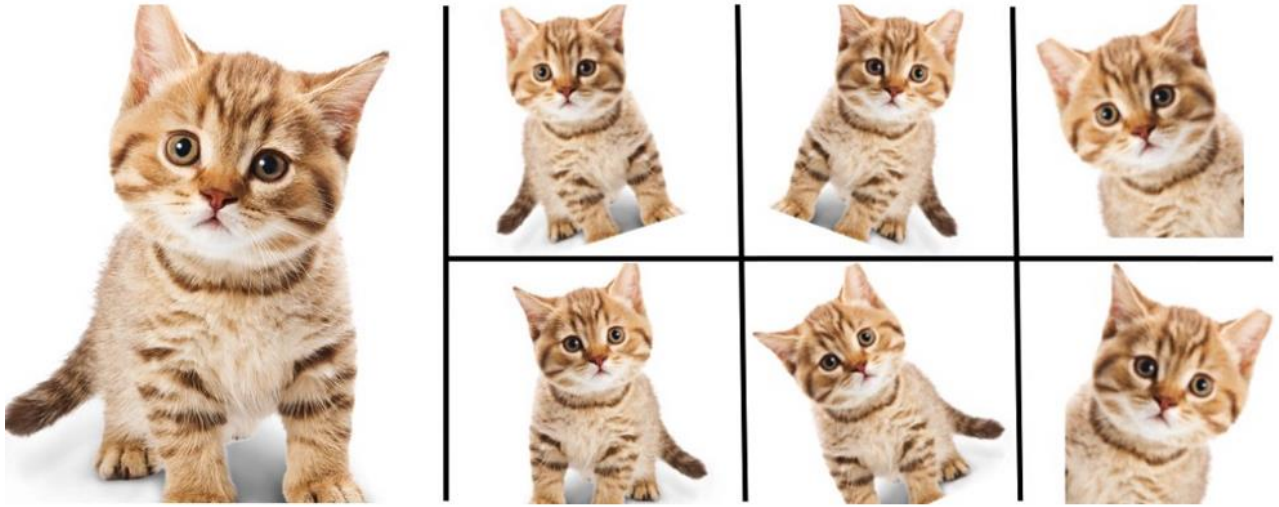
Early Stopping

Note: generally, the validation set is passed at the end of each *epoch*.



Data Augmentation

- We apply different transformations to the input data → we increase the training set
  - E.g. in images: rotations, contrast changes, noise insertion, *mirroring*,…
  - Depending on the problem, some transformations will be valid and others will not. E.g. if you want to classify digits and you rotate a 6 a lot, it could become a 9…



Loss Penalties

L1 Regularization (a.k.a. Lasso regularization)

$$\text{Cost} = \sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2 + \lambda \sum_{j=0}^{M} |W_j|$$

L2 Regularization (a.k.a. Tikhonov regularization or Ridge regularization or Weight decay [in some cases])

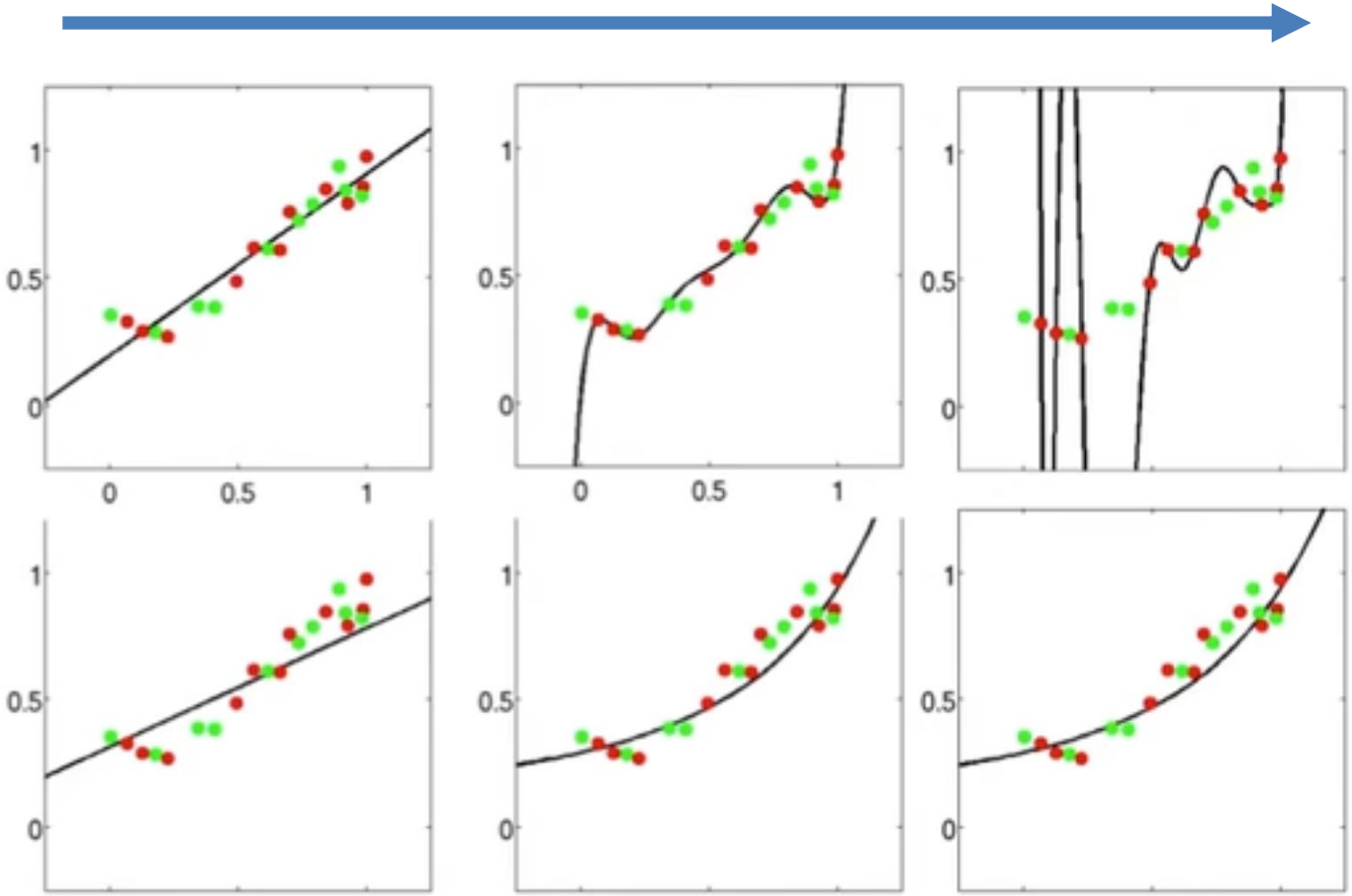
$$\text{Cost} = \underbrace{\sum_{i=0}^{N} (y_i - \sum_{j=0}^{M} x_{ij} W_j)^2}_{\text{Loss function}} + \underbrace{\lambda \sum_{j=0}^{M} W_j^2}_{\text{Regularization Term}}$$

We try to achieve a trade-off between having a low error and small weight/parameter values!!

Note: overfitting is associated with larger parameter/weight values.

Loss Penalties

Higher order polinomials



Source: Alexander Ihler

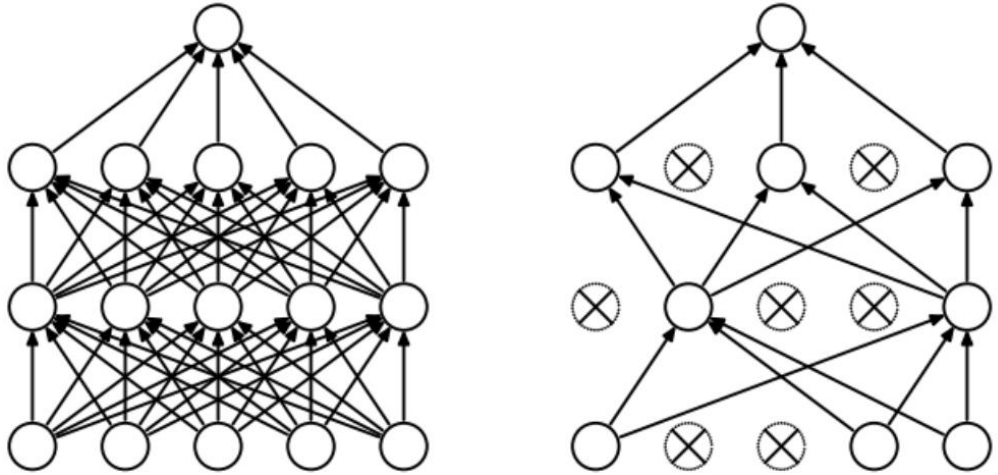
Normalization

- **Batch normalization** [Ioffe & Szegedy, 2015]
  - It is known that normalizing input accelerates training

  - The idea is to normalize the inputs to the hidden layers (usually just before applying the activ. func.)
  - We'll see more details when specifically talking about Deep Neural Networks.

Dropout

- **Dropout** [Hinton et al., 2012]
  - We randomly eliminate hidden units during training.
  - It forces the "generalization" of units, decoupling them from each other.
  - Generally, it is applied in fully-connected layers, and with different units in each iteration.



(a) Standard Neural Net

(b) After applying dropout.

Some machine learning models

Hypothesis Set	Learning Algorithm
Artificial Neural Networks (ANNs)	Backpropagation + Gradient Descent
Perceptron	Perceptron Learning Algorithm (PLA)
Linear Regression	Pseudoinverse
Logistic Regression	Stochastic Gradient Descent (SGD)
Support Vector Machines (SVMs)	Quadratic Programming
…	…

Hypothesis Set: set of candidate formulas to solve our problem ($f: X \rightarrow Y$).

Learning Algorithm: *it uses* the dataset to pick a formula g that approximates f. The algorithm chooses g from a set of candidate formulas under consideration (hypothesis set).

Learning Model: Hypothesis Set + Learning Algorithm

This is the terminology employed by Abu-Mostafa et al. (2012).
See **Lecture 1 – The Learning Problem**: https://work.caltech.edu/slides/slides01.pdf

Some machine learning models

There are **many different models** (see https://www.asimovinstitute.org/neural-network-zoo). Depending on your **problem** and **available data**, you will select your candidate models:

- **Tabular information**
  - Support Vector Machines (SVMs), Random Forest (RF), Multilayer Perceptron (MLP), logistic regression, linear regression,…
    - Binary classification, linearly separable problem: perceptron learning algorithm (PLA)
    - Binary classification, non-linearly separable problem: pockect algorithm
    - Binary classification, need of probabilistic interpretation: logistic regression
    - Multiclass classification, need of probabilistic interpretation: multinomial logistic regression
    - Regression: linear regression, polynomial regression
    - Binary classification (and regression), largest margin between two classes: SVM
    - Simple and interpretable models for classification and regression: Decision Trees (DTs)
    - Boost performance using an ensemble of weak learners: RF
    - …

This list is not intended to be exhaustive or definitive. These are just some ideas that can serve as a general reference on certain occasions.

Some machine learning models

There are **many different models** (see https://www.asimovinstitute.org/neural-network-zoo). Depending on your **problem** and **available data**, you will select your candidate models:

- **Images**
  - Convolutional Neural Networks (ConvNets)
- **Sequence tagging (audio processing, time series)**
  - Recurrent Neural Networks (RNNs), Transformers
- **Generation**
  - Generative Adversarial Networks (GANs), Difussion Models
- **Structured information in the form of a graph**
  - Graph Neural Networks (GNNs)
- **Dimensionality reduction or denoising**
  - Autoencoders

This list is not intended to be exhaustive or definitive. These are just some ideas that can serve as a general reference on certain occasions.

(A Very Brief) Introduction to Machine Learning

Pablo Mesejo

pmesejo@go.ugr.es

Universidad de Granada

Departamento de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE GRANADA

DaSCI

Instituto Andaluz de Investigación en **Data Science** and **Computational Intelligence**