

# Deeplearning from zero to hero.

Gianluca Carucci | @rucka | [gianluca.carucci.org](http://gianluca.carucci.org)

what I do

Software Engineer | Agile coach

@Hyperfair inc



COMMUNITY

communities

---

Ugidotnet | Scala Milano Meetup

# Neural Networks for Machine Learning

## derivatives of a logistic neuron

**About this course:** Learn about artificial neural networks and how they're being used for machine learning, as applied to speech and object recognition, image segmentation, modeling language and human motion, etc. We'll emphasize both the basic algorithms and the practical tricks needed to get

"good" results.

▼ More

**Created by:** University of Toronto



**Taught by:** Geoffrey Hinton, Professor  
Department of Computer Science

$$\frac{d}{dz} \left( \frac{e^{-z}}{(1+e^{-z})^2} \right) = \frac{1}{(1+e^{-z})^2} \left( \frac{e^{-z}}{(1+e^{-z})} \right)' =$$

$$\frac{d}{dz} \left( \frac{e^{-z}}{(1+e^{-z})^2} \right) = \frac{(1+e^{-z}) - 1}{(1+e^{-z})^2} = \frac{(1+e^{-z})}{(1+e^{-z})^2} =$$



not a good start

A black and white photograph of a woman with long, light-colored hair, seen from behind and slightly to the side. She is wearing a dark, sleeveless top and a wide-brimmed hat. Her head is bowed, and she appears to be crying or deeply distressed. The background is blurred, showing what might be foliage or a garden.

is there another way?





look mum, no code!

STARTED

PROGRAMMER'S GUIDE

TUTORIALS

PERFORMANCE

MOBILE

## How to Retrain Inception's Final Layer for New Categories

# transfer learning<sup>1</sup>

Modern object recognition models have millions of parameters and can take weeks to fully train.

Transfer learning is a technique that shortcuts a lot of this work by taking a fully-trained model for a set of categories like ImageNet, and retrains from the existing weights for new classes. In this example we'll be retraining the final layer from scratch, while leaving all the others untouched. For more information on the approach you can see this paper on Decaf.

Though it's not as good as a full training run, this is surprisingly effective for many applications, and can be run in as little as thirty minutes on a laptop, without requiring a GPU. This tutorial will show you how to run the example script on your own images, and will explain some of the options you have to help control the training process.

[How to Retrain Inception's Final Layer for New Categories](#)

[TensorFlow for Poets](#)

Note: A version of this tutorial is also available as a [codelab](#).

Before you start, you must install tensorflow.

Contents

Training on Flower

Bottlenecks

Training

Visualizing the Retraining with TensorBoard

Using the Retrained Model

Training on Your Own Categories

Creating a Set of Training Images

Training Steps

Distortions

Hyper-parameters

Training, Validation

Testing Sets

Other Model Architectures



# the simpsons detector

```
root@cc5a496bf40e:/code/simpson_retraining# ./retrain.sh
INFO:tensorflow:Looking for images in 'marge'
INFO:tensorflow:Looking for images in 'homer'
INFO:tensorflow:Looking for images in 'bart'
INFO:tensorflow:Looking for images in 'lisa'
...
INFO:tensorflow:2018-02-25 09:26:27.689515: Step 0: Train accuracy = 53.0%
INFO:tensorflow:2018-02-25 09:26:27.689750: Step 0: Cross entropy = 1.367082
INFO:tensorflow:2018-02-25 09:26:28.583996: Step 0: Validation accuracy = 33.0% (N=100)
...
INFO:tensorflow:2018-02-25 09:26:49.359180: Step 30: Train accuracy = 65.0%
INFO:tensorflow:2018-02-25 09:26:49.359404: Step 30: Cross entropy = 1.172318
INFO:tensorflow:2018-02-25 09:26:49.871993: Step 30: Validation accuracy = 69.0% (N=100)
...
INFO:tensorflow:2018-02-25 09:35:37.061001: Step 1000: Train accuracy = 95.0%
INFO:tensorflow:2018-02-25 09:35:37.061221: Step 1000: Cross entropy = 0.311776
INFO:tensorflow:2018-02-25 09:35:37.600067: Step 1000: Validation accuracy = 87.0% (N=100)
...
INFO:tensorflow:2018-02-25 09:44:03.139348: Step 2000: Train accuracy = 95.0%
INFO:tensorflow:2018-02-25 09:44:03.139593: Step 2000: Cross entropy = 0.286085
INFO:tensorflow:2018-02-25 09:44:03.538799: Step 2000: Validation accuracy = 83.0% (N=100)
...
INFO:tensorflow:2018-02-25 10:00:39.674607: Step 3999: Train accuracy = 97.0%
INFO:tensorflow:2018-02-25 10:00:39.674859: Step 3999: Cross entropy = 0.175444
INFO:tensorflow:2018-02-25 10:00:40.073648: Step 3999: Validation accuracy = 90.0% (N=100)
INFO:tensorflow:Final test accuracy = 90.4% (N=261)
```

```
root:/code/simpson_retraining# ./evaluate.sh /data/simpson/test_set/0.jpg^
bart 0.63594097
marge 0.3191907
lisa 0.034744244
homer 0.010124116
```



# common scenarios

6. Se dissermos que fomos trevas, mentimos, e não praticamos a verdade.  
7. Mas se andarmos na luz, como ele na luz está, não comulgamos uns com os outros, e o sinal de Jesus Cristo, seu Filho, nos purifica de todo pecado.  
8. Se dissermos que não temos pecado nenhum, enganamos-nos a nós mesmos, e não há verdade em nós.  
9. Se confessarmos os nossos pecados, ele é fiel e justo para nos perdoar os pecados, e nos purifica de toda iniquidade.  
10. Se dissermos que não pecamos, fazendo-lhe mentiras, e a sua palavra não está em nós.  
**2** Meus filhos, estas coisas vos escrevo para que não pecareis. Se, porém, alguém pecar, temos um Advogado para com o Pai, Jesus Cristo, o justo.  
2. Ele é a propiciação pelos nossos pecados, e não somente pelos nossos, mas também pelos de todo o mundo.  
3. E nisto sabemos que o conhecemos, se guardamos os seus mandamentos.  
4. Aquela que diz: Eu o conheço, e não guarda os seus mandamentos, é mentiroso, e nele não está a verdade.  
5. Mas qualquer que guarda a sua palavra, o amor de Deus nele tem-se verdadeiramente aperfeiçoado. E nisto conhecemos que estamos nele.  
6. Aquela que diz que está nele, também deve andar como de andou.  
7. Amados, não vos escrevo mandamento novo, mas um mandamento antigo, que desde o princípio testem. Este mandamento antigo é a palavra que ouvistes.  
8. Vendo-vos, souze as trevas vão passar, e o dia brilha a vós.  
10. Aquela que não tem traição, não tem traição.  
11. Mas se vos troparam, não vos desanimem.  
12. Pondo-vos a minha paz no coração.  
13. Pois o mundo vos combate desde o princípio.  
14. Eu vos dei a vitória sobre o mundo, porque vós escrivestes os princípios da vida, e a tua palavra é a verdade de Deus esta é a vitória.  
15. Não ameis o mundo, nem o amor do mundo.  
16. Pois tudo o que há no mundo pertence à concupiscência dos olhos e à carne, e ao orgulho do mundo.  
17. Ora, o mundo passa, e a sua concupiscência, mas que faz a vontade de Deus permanece para sempre.  
18. Filhos, está a ultima hora, e como o diabo anda de dia, os astros tem surgido, pervertindo o que é a ultima hora.

Models and examples built with TensorFlow

1,819 commits 12 branches 2 releases 304 contributors Apache-2.0

Branch: master	New pull request	Create new file	Upload files	Find file	Clone or download
a-dai Merge pull request #3414 from a-dai/master ...					Latest commit d903d5e 13 hours ago
official	Merge pull request #3400 from asimshankar/mnist-eager				6 days ago
research	Merge pull request #3404 from m-dalvi/mnist-eager				13 hours ago
samples	Fix one more Distributioned_Tensorflow				21 hours ago
tutorials	six.moves works with python2/3				14 days ago
.gitignore	Added PyCharm to .gitignore				6 months ago
.gitmodules	Move the research models into a research subfolder (#2430)				5 months ago
AUTHORS	Spatial Transformer model				2 years ago
CODEOWNERS	Add a-dai to CODEOWNERS for adversarial_text.				6 days ago
CONTRIBUTING.md	Fixing small typo				8 months ago
ISSUE_TEMPLATE.md	Update ISSUE_TEMPLATE.md				15 days ago
LICENSE	Update LICENSE				2 years ago
README.md	Update Readme to link to master research models (#3318)				17 days ago
WORKSPACE	Consolidate privacy/ and differential_privacy/.				a year ago
README.md					

## TensorFlow Models

This repository contains a number of different models implemented in TensorFlow:

The official models are a collection of example models that use TensorFlow's high-level APIs. They are intended to be well-maintained, tested, and kept up to date with the latest stable TensorFlow API. They should also be reasonably optimized for fast performance while still being easy to read. We especially recommend newer TensorFlow users to start here.

## TensorFlow Models<sup>2</sup>

The research models are a large collection of models implemented in TensorFlow by researchers. They are not officially supported or available in release branches; it is up to the individual researchers to maintain the models and/or provide support on issues and pull requests.

The samples folder contains code snippets and smaller models that demonstrate features of

- audioset: Models and supporting code for use with AudioSet.
- autoencoder: various autoencoders.
- brain\_coder: Program synthesis with reinforcement learning.
- cognitive\_mapping\_and\_planning: implementation of a spatial memory based mapping and planning architecture for visual navigation.
- compression: compressing and decompressing images using a pre-trained Residual GRU network.
- delf: deep local features for image matching and retrieval.
- differential\_privacy: privacy-preserving student models from multiple teachers.
- domain\_adaptation: domain separation networks.
- gan: generative adversarial networks.
- adversarial\_text: neural network for generating text.
- image\_caption: convolutional networks for computer vision.
- learning\_to\_remember\_rare\_events: a large-scale life-long memory module for use in deep learning.
- Ifads: sequential variational autoencoder for analyzing neuroscience data.
- lm\_1b: language modeling on the one billion word benchmark.
- namigner: recognize and generate names.
- neural\_gpu: highly parallel neural computer.
- neural\_programmer: neural network augmented with logic and mathematic operations.
- next\_frame\_prediction: probabilistic future frame synthesis via cross convolutional networks.
- object\_detection: localizing and identifying multiple objects in a single image.
- pcl\_rl: code for several reinforcement learning algorithms, including Path Consistency Learning.
- ptn: perspective transformer nets for 3D object reconstruction.
- qa\_kg: module networks for question answering on knowledge graphs.
- real\_nvp: density estimation using real-valued non-volume preserving (real NVP) transformations.
- rebar: low-variance, unbiased gradient estimates for discrete latent variable models.
- resnet: deep and wide residual networks.
- skip\_thoughts: recurrent neural network sentence-to-vector encoder.
- slim: image classification models in TF-Slim.
- street: identify the name of a street (in France) from an image using a Deep RNN.
- swivel: the Swivel algorithm for generating word embeddings.
- syntaxnet: neural models of natural language syntax.
- tcn: Self-supervised representation learning from multi-view video.

common solutions

ready to go models<sup>2</sup>

## Object Detection

- + Face Detection



• Previous

© Copyright 2013, Alexan

Built with Sphinx using a t

<sup>3</sup> - [OpenCV-Python](#)

- [Spacy](#)

- [gensim](#)

# common solutions

## ready to go libraries<sup>3</sup>

### Get things done

spaCy is designed to help you do real work — to build real products, or gather real insights. The library respects your time, and tries to avoid wasting it. It's easy to install, and its API is simple and productive. We like to think of spaCy as the Ruby on Rails of Natural Language Processing.

#### similarities

- + Merklet file on disk.  
`corpus.mer`)

- + Working with 200 dimensions.  
`sp(200)`)

- + Count space and index (L  
`ity[1st][another\_corpus]`)

- + Unknown documents

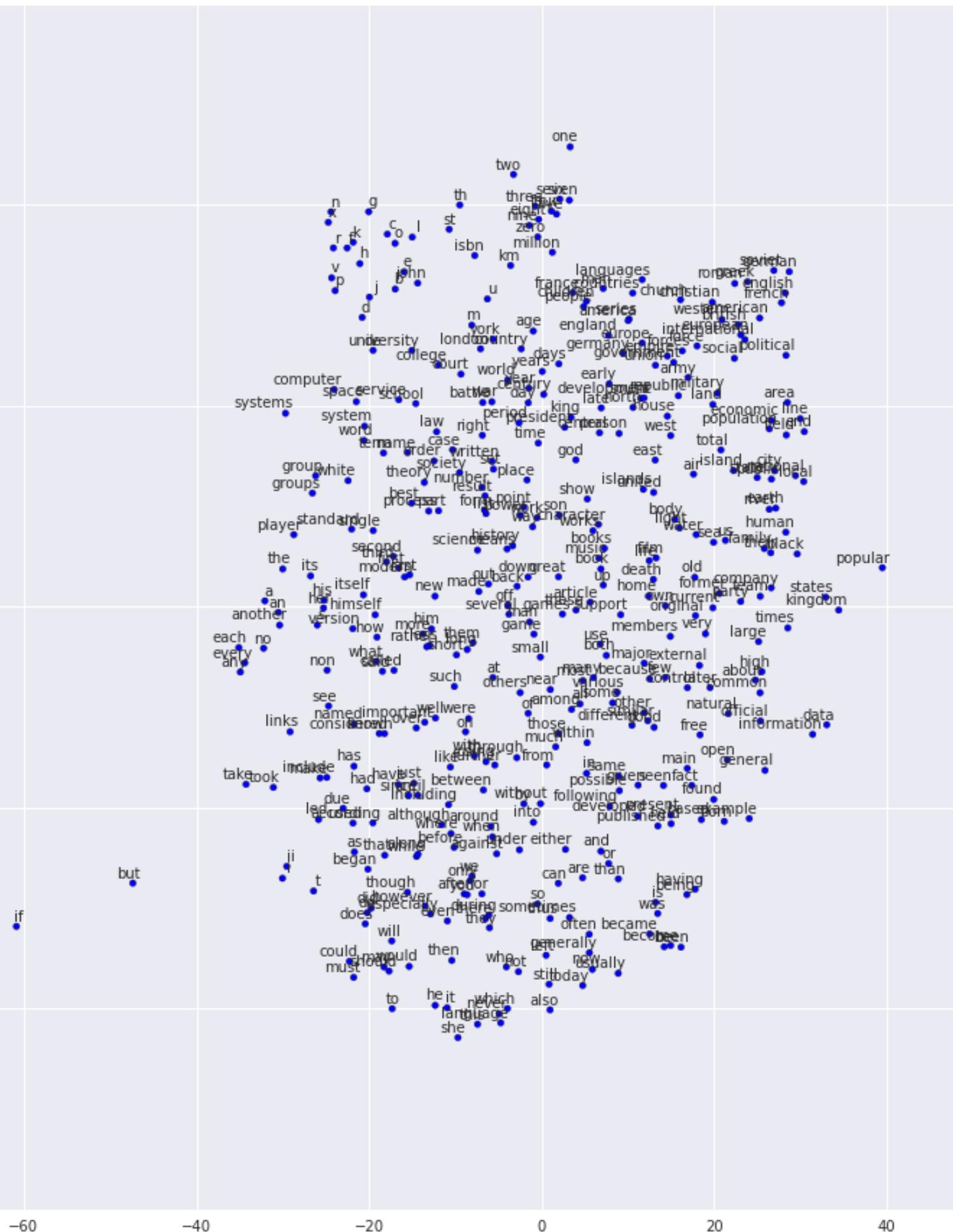
## Gensim is

- ✓ Available statistical

- ✓ Analyze plain-t

- ✓ Retrieve semant

# Word Algebra<sup>4</sup>



```
from gensim.models.KeyedVectors import load_word2vec_format

model = load_word2vec_format(
    'GoogleNews-vectors-negative300.bin',
    limit=2000000, binary=True)

#calculate: (Rome - Italy) + France = ?
model.similar_by_vector(
    model['Rome'] - model['Italy'] + model['France'],
    topn=1)
#[('Paris', 0.7167540192604065)]

#calculate: (doctor - man) + woman = ?
model.most_similar(
    model['doctor'] - model['man'] + model['woman'],
    topn=1)
#[('nurse', 0.7127888798713684)]

#calculate: (gone - go) + eat = ?
model.most_similar(
    positive=['gone', 'eat'], negative=['go'],
    topn=1)
#[('eaten', 0.7462186217308044)]
```

<sup>4</sup> Credits: The dark side of deep learning - Simone Scardapane - Codemotion 2017

## Datasets

1. MNIST Handwritten digits
  2. Google House Numbers from street view
  3. CIFAR-10 and CIFAR-100
  4. IMAGENET
  5. Tiny Images 80 Million tiny images<sup>6</sup>.
  6. Flickr Data 100 Million Yahoo dataset
  7. Berkeley Segmentation Dataset 500
  8. UC Irvine Machine Learning Repository
  9. Flickr 8k
  10. Flickr 30k
  11. Microsoft COCO
  12. VQA
- 
- 13.<sup>5</sup> - [Awesome deep learning#datasets \(Github\)](#)
  14. - [List of datasets for machine learning research \(Wikipedia\)](#)
  15. - [Deep Learning datasets](#)
- AVIRIR Pathfinder
16. Air Freight - The Air Freight data set is a ray-traced image sequence along with ground truth segmentation based on
- common solutions**
- ready to go dataset<sup>5</sup>**

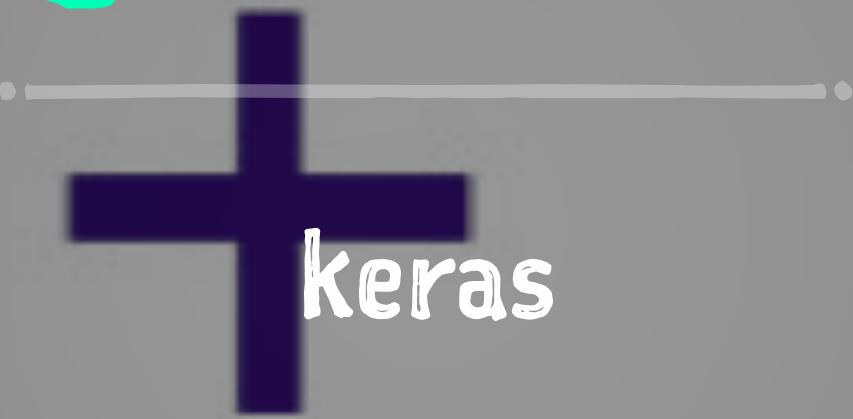
# RECIPES

## what else?

RECIPE  
Pork Chalupas

from Grams  
cooking time 6-8 hours  
2-3 # pork roast  
(I remove fat after roast begins to be  
tender or you can do it before cooked)  
1 lb. pinto beans  
clove garlic  
white pw.  
chopped  
over many  
preheat oven to Crock pot or stove top  
1 cube beef bouillon  
or canned broth  
Jalapeno (opt.)

# ingredients



keras

tensorflow

TensorFlow

# deep learning in 5 steps



**STAR  
WARS**

# 1. define the problem

regression

classification

# 2. prepare your data

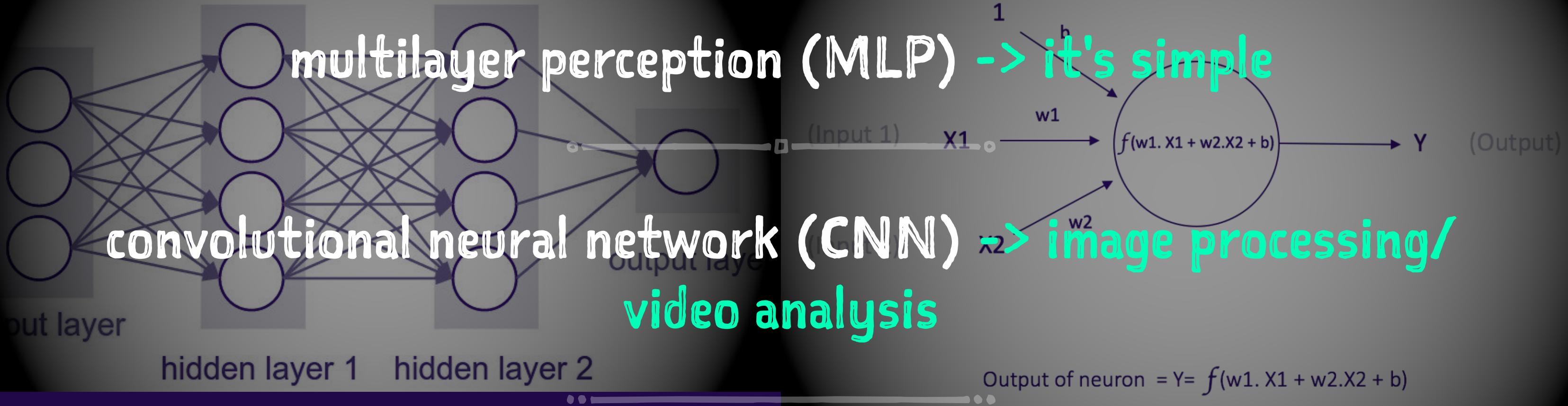
---

choose/create dataset

clean data

normalize data

# 3. define the model



# Neural Networks

Color Guided Matrix Multiplication for a Binary Classification Task  
with  $N = 4$

Input Layer

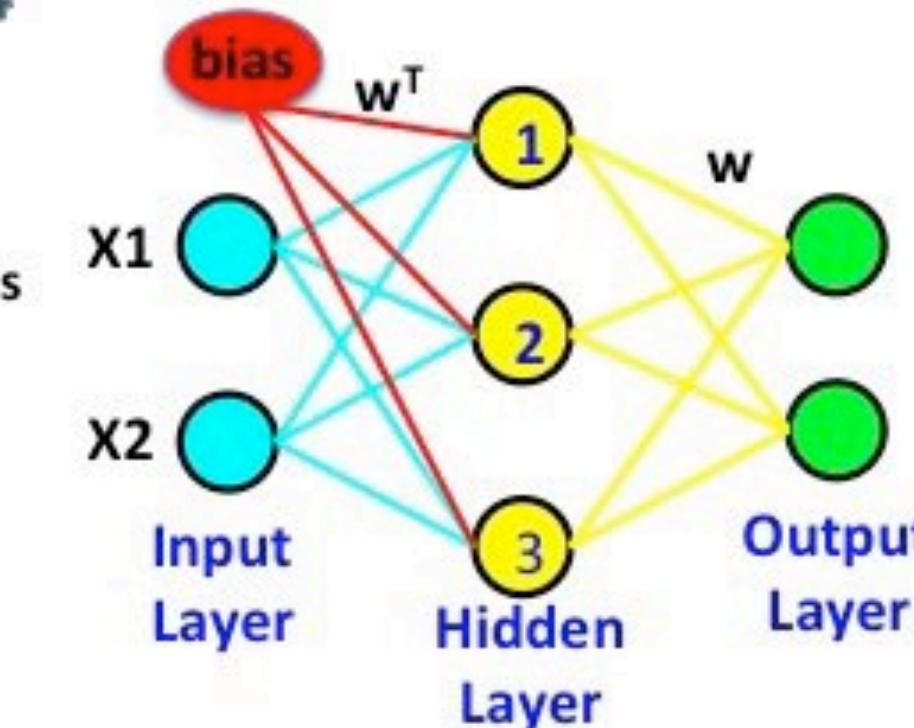
bias X1 X2

$$\begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad 4 \times 3$$

Weights  $w^T$   
(transposed)

$$\begin{bmatrix} .5 & .5 & .5 \\ .5 & .5 & .5 \\ .5 & .5 & .5 \end{bmatrix}$$

Go to  
Hidden Nodes  
 $3 \times 3$



Hidden  
Layer

Bias

Node 1

Node 2

Node 3

$$= \begin{bmatrix} 1 & 1 & 1 \\ .5 & .5 & .5 \\ .5 & .5 & .5 \\ 1 & 1 & 1 \end{bmatrix} \quad 4 \times 3$$

Sigmoid  
Function

$$\frac{1}{1 + e^{-(wx+b)}}$$

Weights

$$\begin{bmatrix} .2 & .1 \\ .4 & .1 \\ .4 & .1 \end{bmatrix} \quad 3 \times 2$$

Output  
Layer

$$\begin{bmatrix} 1 & .3 \\ .5 & .15 \\ .5 & .15 \\ 1 & .3 \end{bmatrix} \quad 4 \times 2$$

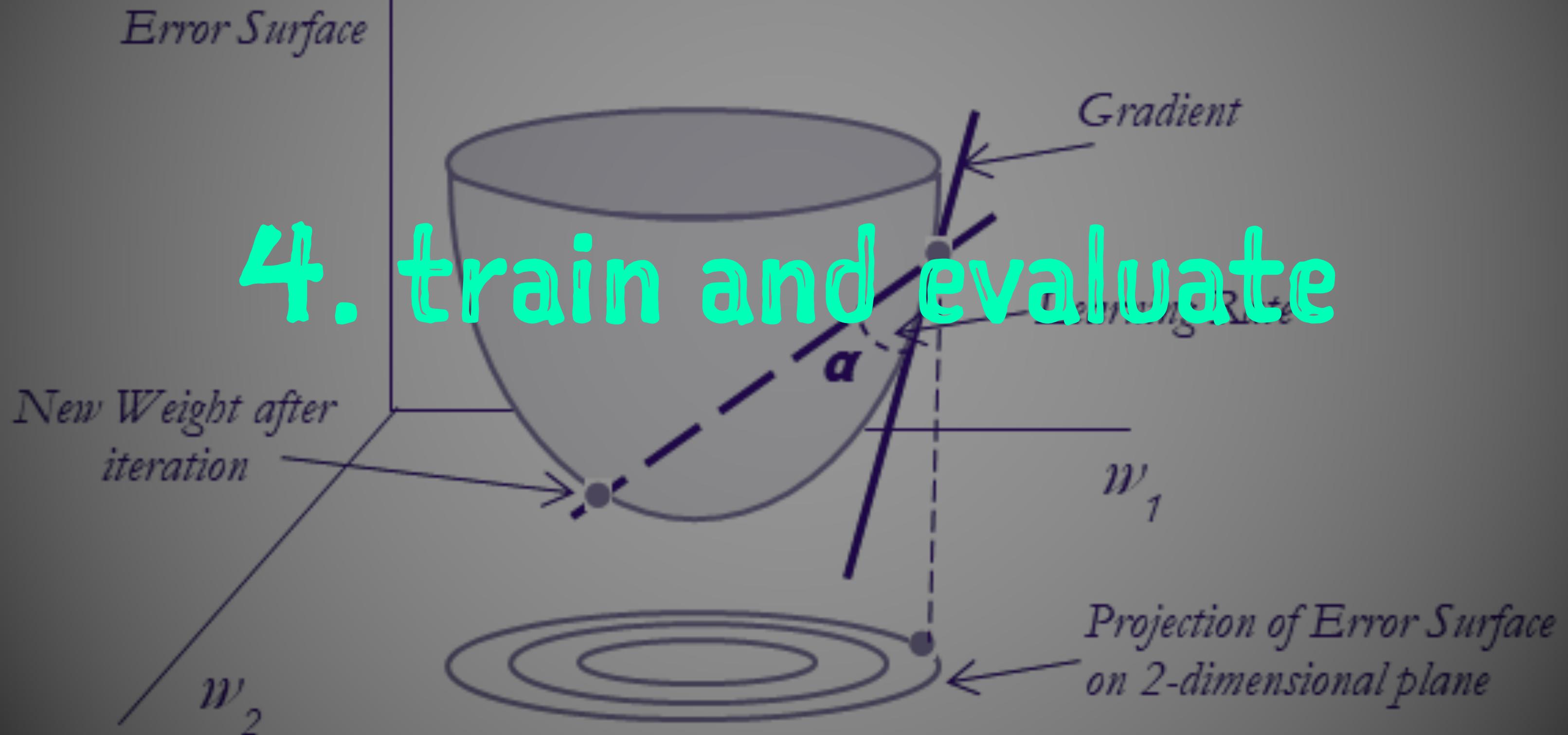
Sigmoid  
Function

$$\frac{1}{1 + e^{-(wx+b)}}$$

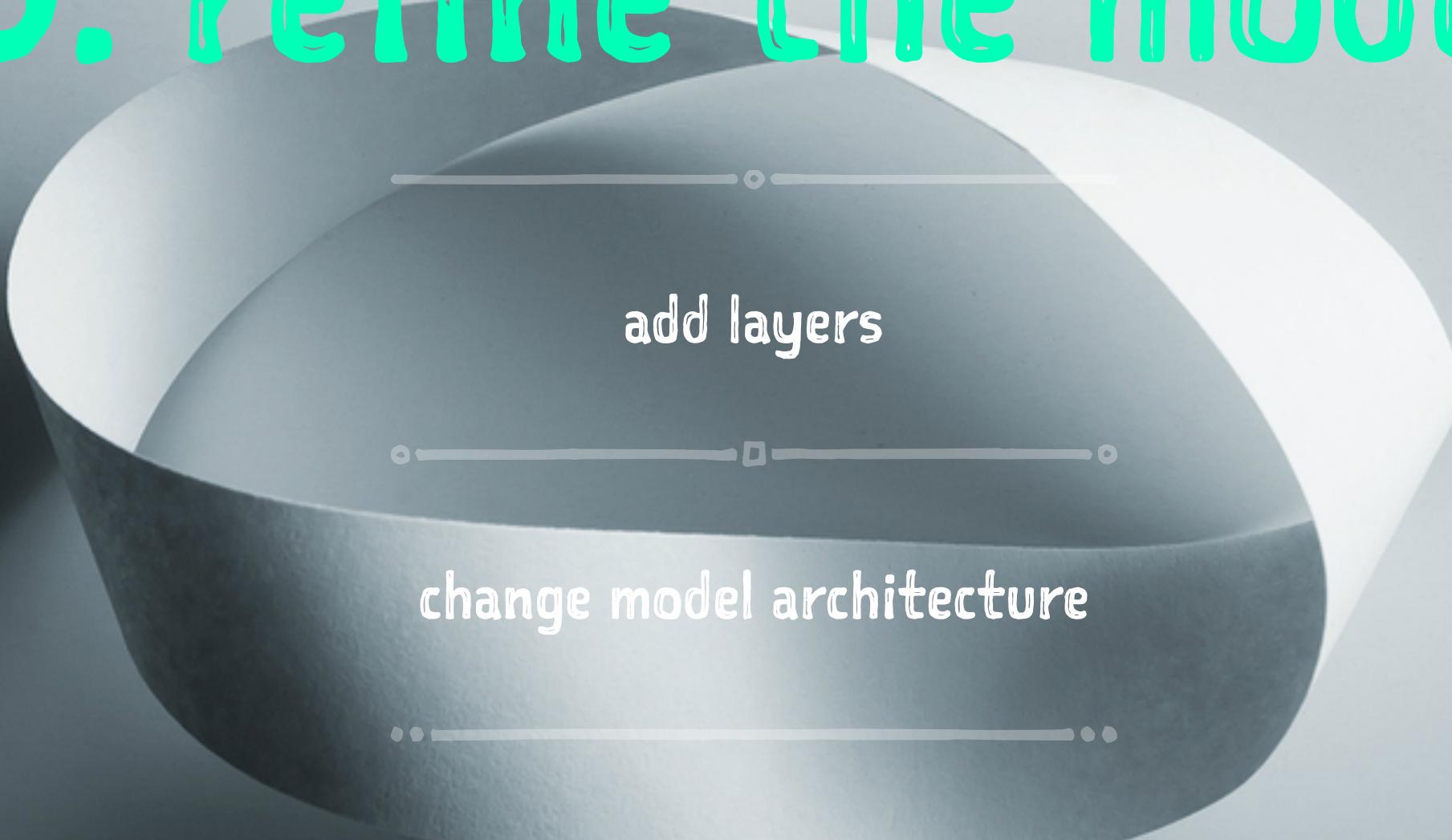
Output

$$\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$$

# *Gradient Descent*



# 5. refine the model



add layers

change model architecture

enhance your data



tip 1:

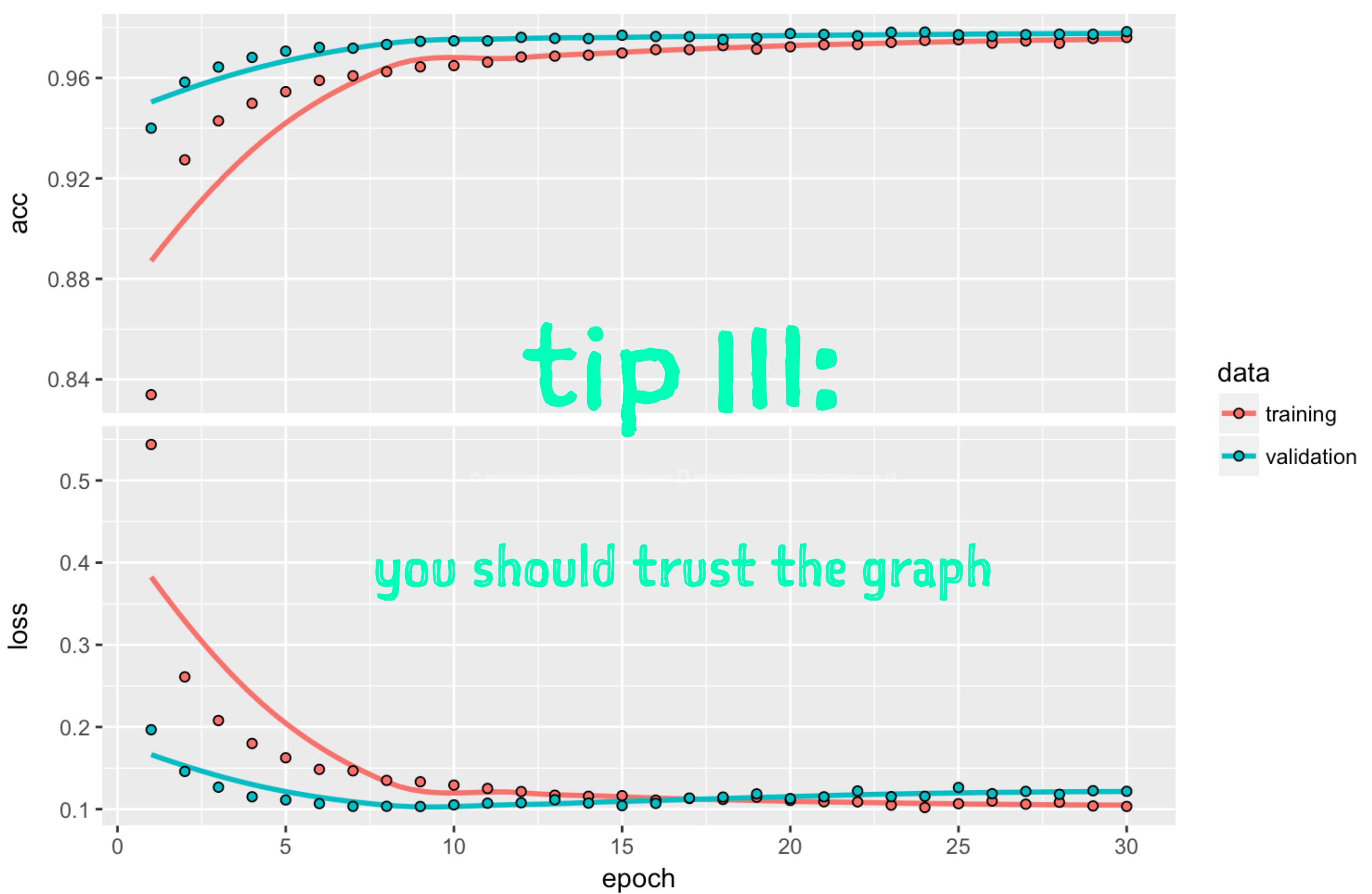
---

your dataset is the key

please, start with the simplest model possible  
please, start with the simplest mo

tip!!







tip IV:

---

patient

AFC

TUNING

REO — OFF

NO — ON

FM STEREO

M-AM TUNER

TUNER INPUT

FM 108 104 MHz

0 10 20 30 40 50

AM 530 600 700 800 kHz

1000 1200 1400 1600

104 108 MHz

1000 1200 1400 1600 kHz

# bonus track: hyperparameters tuning<sup>6</sup>

scikit learn GridSearchCV, RandomizedSearchCV

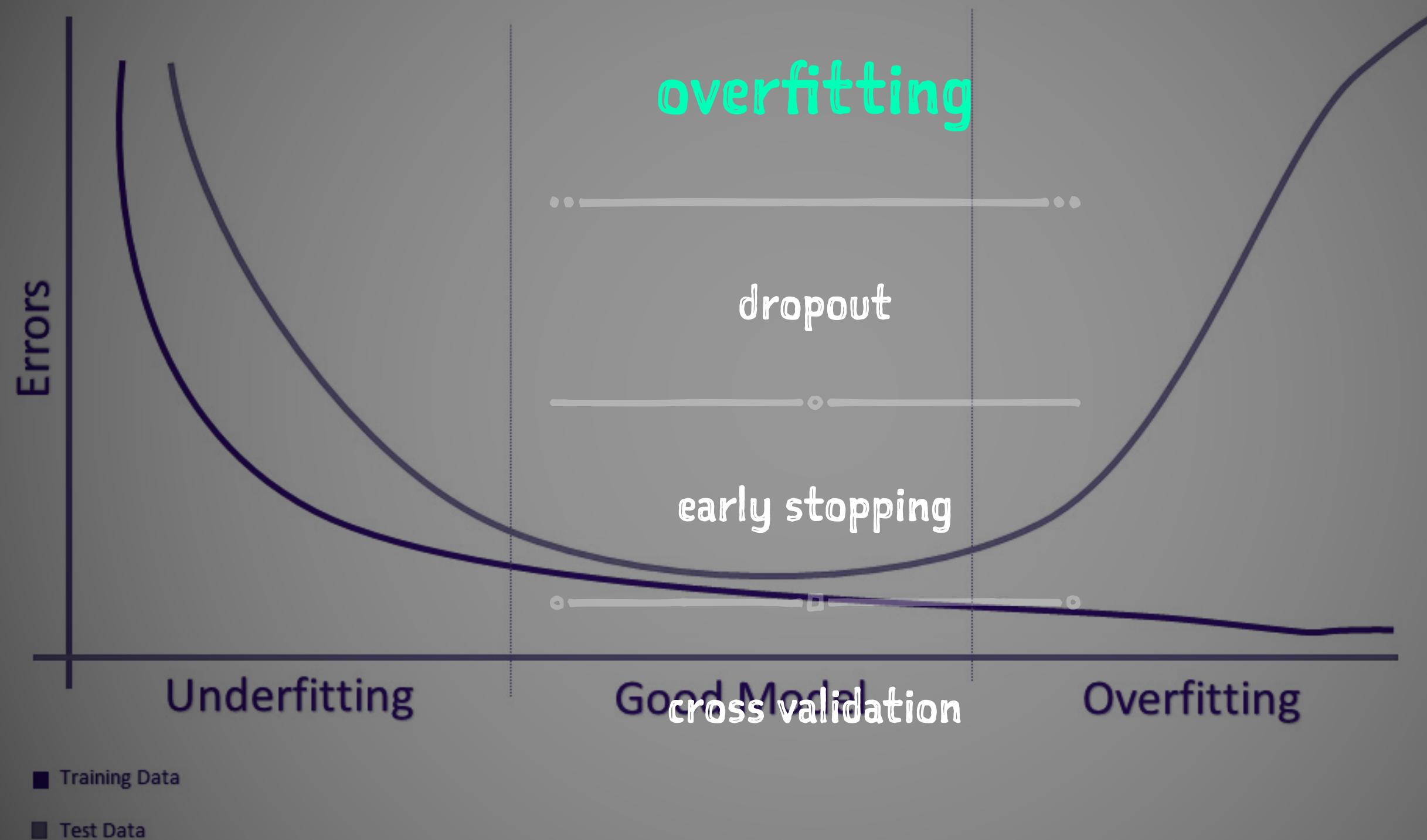
hyperopt

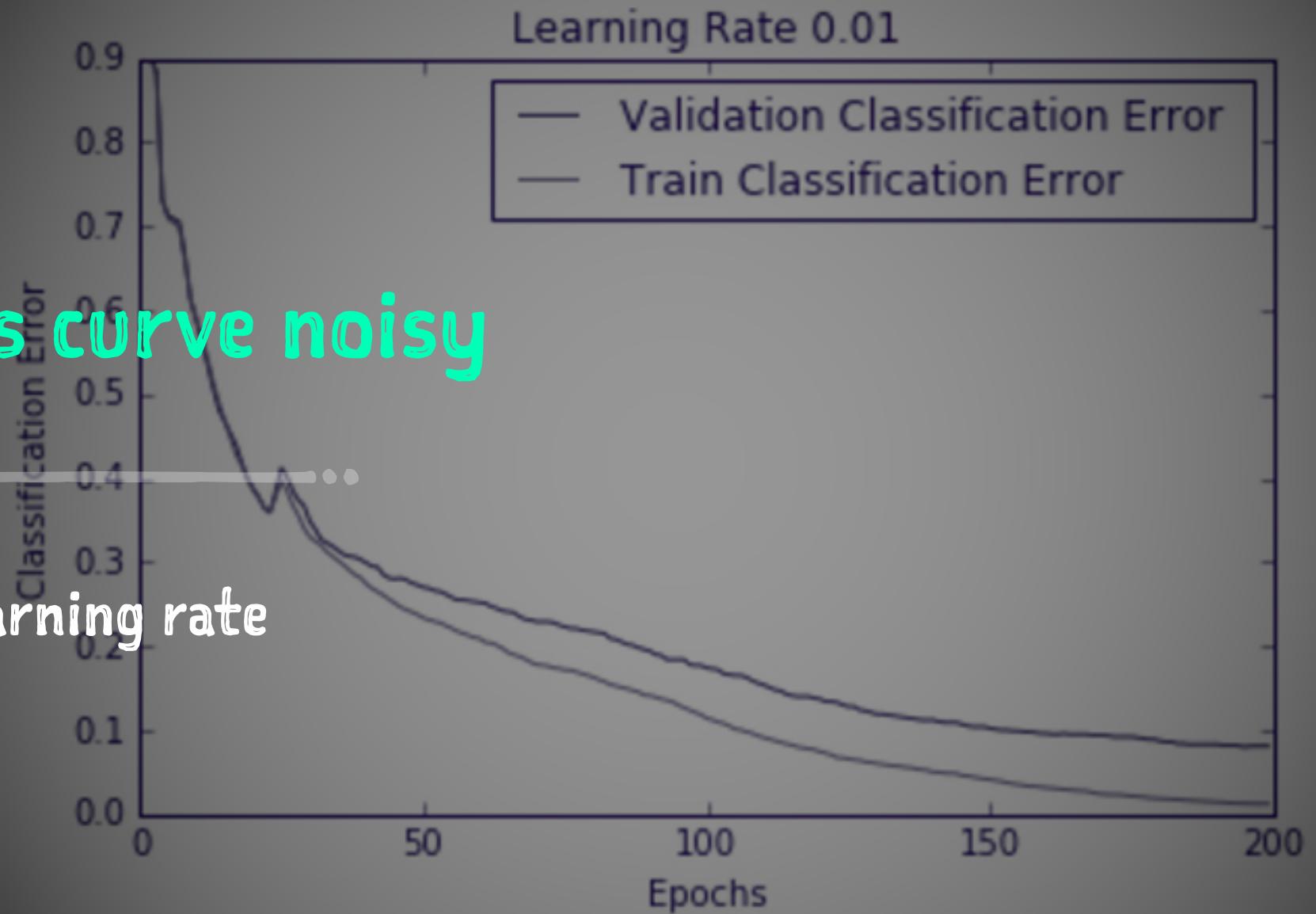
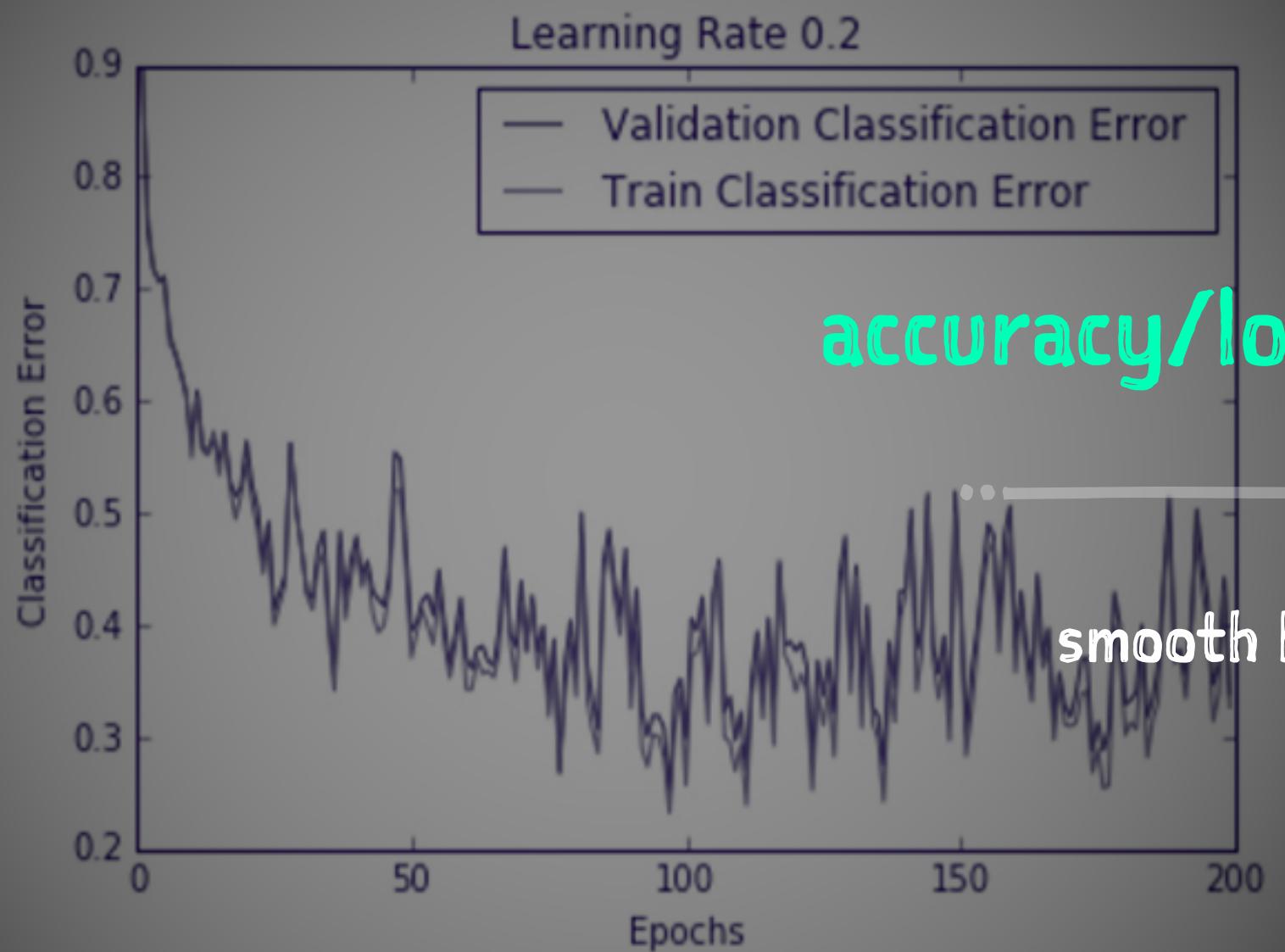
<sup>6</sup> - scikit learn Exhaustive&Random Grid Search

- Hyperopt

A black and white photograph showing a close-up of a wrench and a nut on a light-colored wooden surface. The wrench is positioned diagonally across the frame, with its handle pointing towards the top right and its jaws towards the bottom left. A single nut lies next to the wrench. The lighting creates strong highlights and shadows on the metallic surfaces of the tools and the grain of the wood.

troubleshooting





accuracy/loss curve noisy

smooth learning rate

# Resources #1

- [Google Machine Learning Crash Course](#)
- [Learn TensorFlow and deep learning, without a Ph.D. by Martin Görner](#)
- [How to Use Metrics for Deep Learning with Keras in Python](#)
- [Neural Networks for Machine Learning course from Coursera](#)

# Resources #2

- Machine learning finance
- Keras
- Tensorflow
- rucka/deeplearning docker image

It is a capital mistake  
to theorize before one has  
data

Sherlock Holmes

A photograph of a dark, wooded area. In the foreground, the textured bark of a large tree trunk is visible, showing signs of age and wear. A path or clearing leads into the background where more trees stand in a dense forest.

questions?

# thank you!



---

slide & code

---

<https://github.com/rucka/pycon9>

---

[https://github.com/rucka/deeplearning\\_docker](https://github.com/rucka/deeplearning_docker)