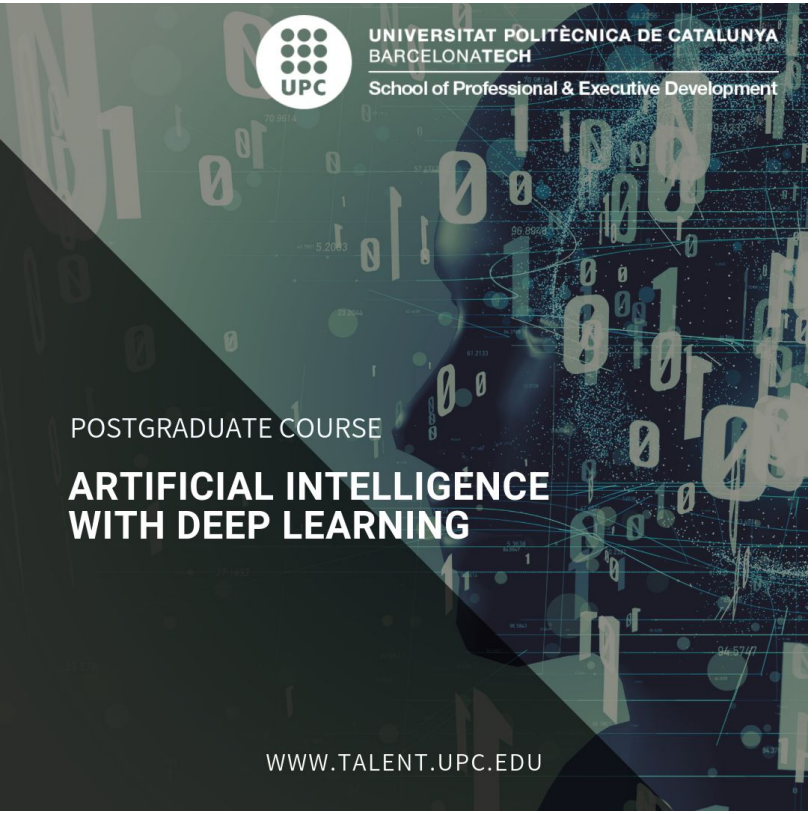


#DLUPC

Day 2 Lab 2

Underfitting & Overfitting

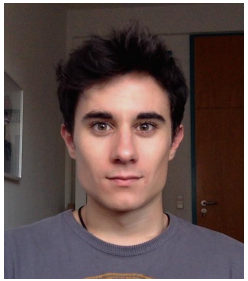
Xavier Giro-i-Nieto
xavier.giro@upc.edu

Associate Professor
Universitat Politècnica de Catalunya
Technical University of Catalonia

Daniel Fojo
dani.fojo@gmail.com

Research Engineer
Disney Research

Acknowledgements



Víctor Campos
victor.campos@bsc.es

PhD Candidate

Barcelona Supercomputing Center

Míriam Bellver
miriam.bellver@bsc.edu

PhD Candidate
Barcelona Supercomputing Center

Amaia Salvador
amaia.salvador@upc.edu

PhD Candidate
Universitat Politècnica de Catalunya

Santiago Pascual de la Puente
santi.pascual@upc.edu

PhD Candidate
Universitat Politecnica de Catalunya
Technical University of Catalonia

Lab created by

Míriam Bellver

miriam.bellver@bsc.edu

PhD Candidate

Barcelona Supercomputing Center

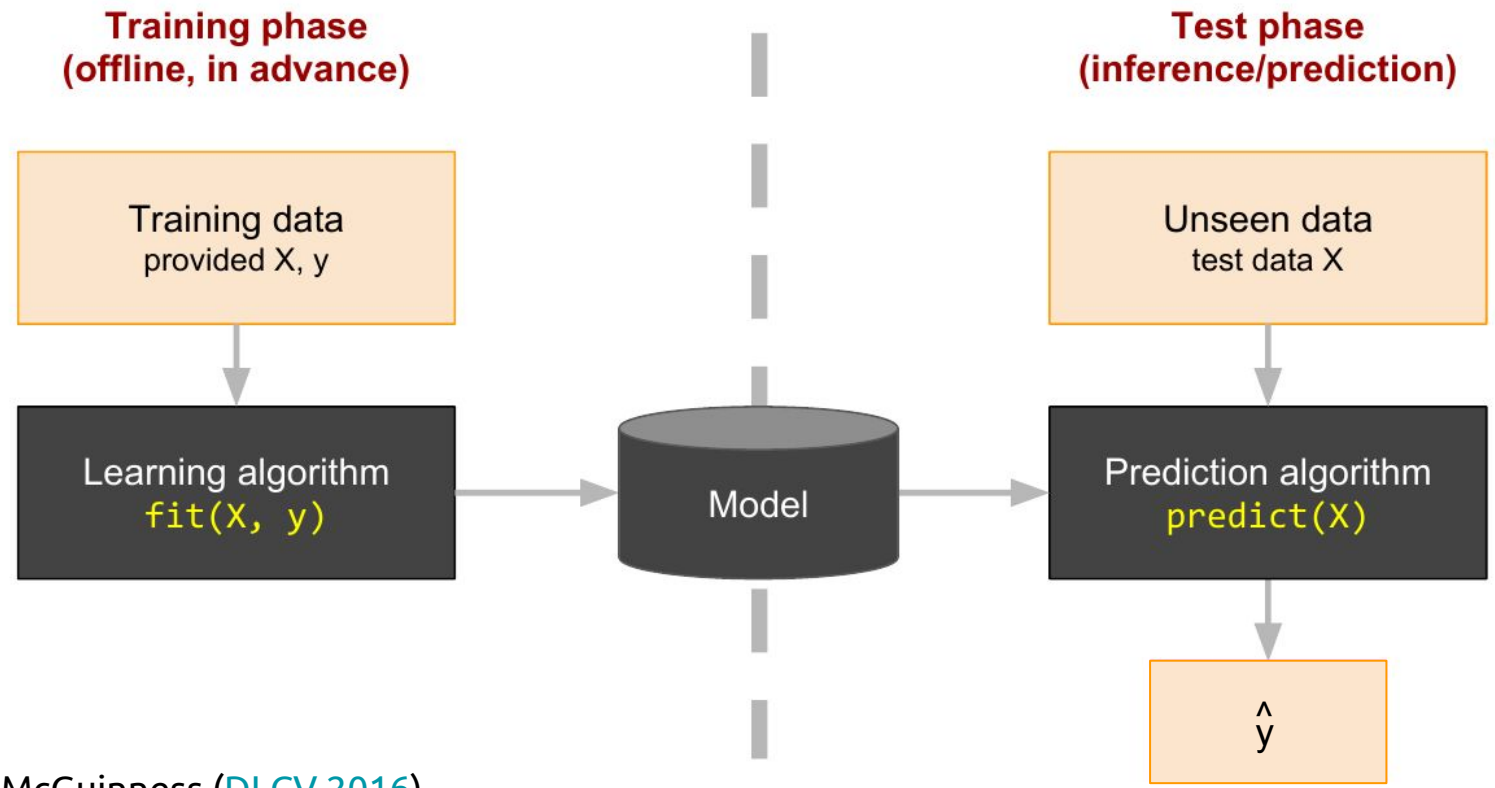
Supervised Learning

Fit a function: $\mathbf{y} = f(\mathbf{x}), \quad \mathbf{x} \in \mathbb{R}^m$

Given paired training examples $\{(\mathbf{x}_i, \mathbf{y}_i)\}$

Key point: **generalize well to unseen examples**

Supervised Learning

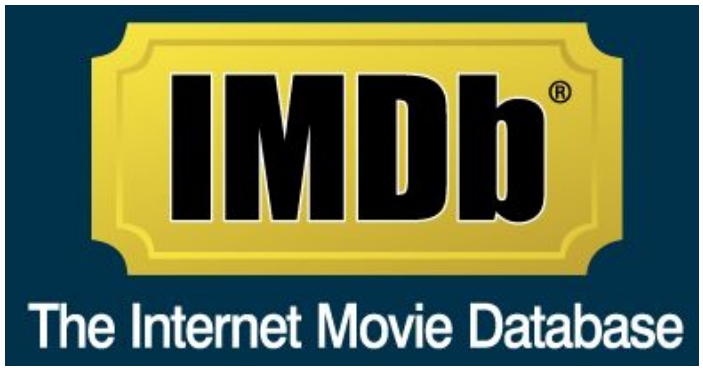


Slide: Kevin McGuinness (DLCV 2016)

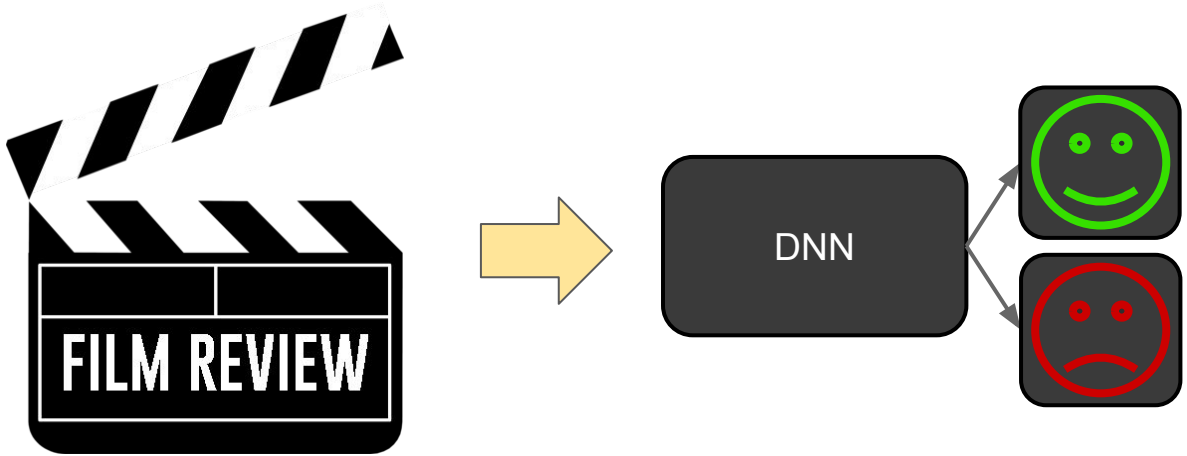
Data

Dataset: IMDB Movies Datasets

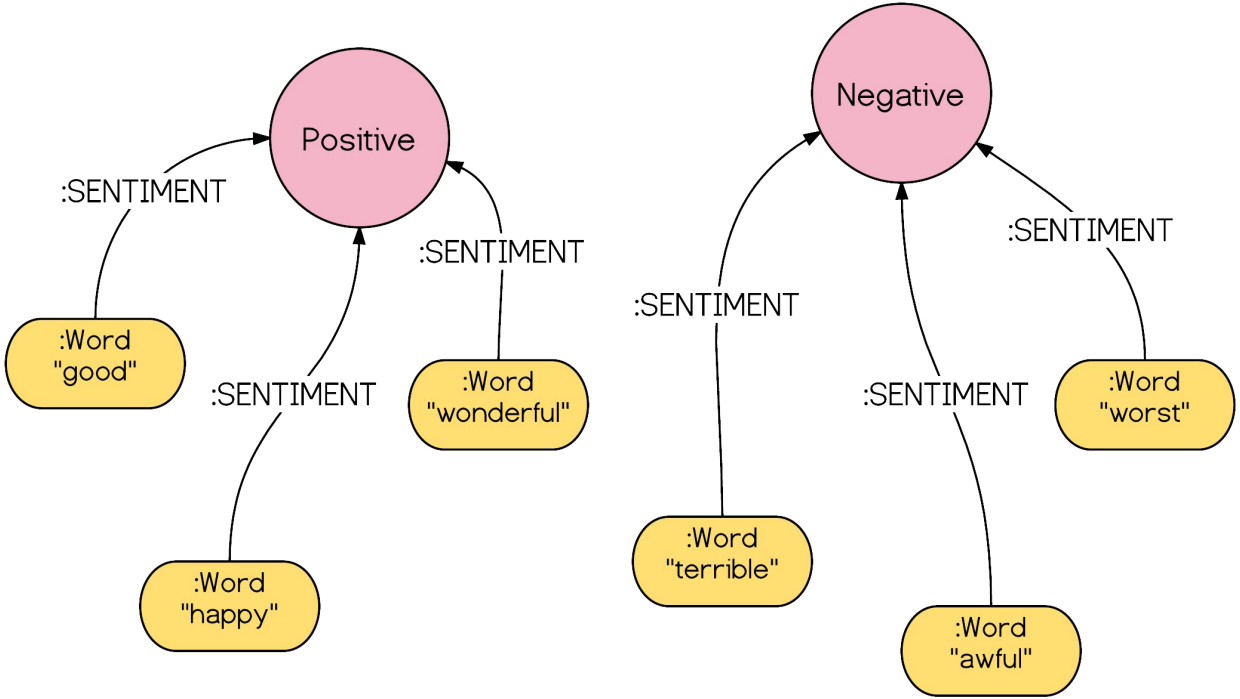
- 14,762 movies
- The purpose of the dataset was to produce movies recommendation systems
- The details that this dataset has are:
  - title, region, language, original title, start year, end year, directors.. lots of information regarding movies.



Task



Task


Positive
:SENTIMENT
:SENTIMENT
:SENTIMENT
:Word
"good"
:Word
"wonderful"
:Word
"happy"
Negative
:SENTIMENT
:SENTIMENT
:SENTIMENT
:Word
"terrible"
:Word
"awful"
:Word
"worst"

One hot encoding for Words

cat: x^T = [1,0,0, ..., 0]

dog: x^T = [0,1,0, ..., 0]

.

.

house: x^T = [0,0,0, …,0,1,0,...,0]

.

.

.

Bag of Words encoding for Reviews

Fig: gk_ (Medium 2017)

Practical tips for training deep nets

Choosing hyperparameters

Can already see we have lots of **hyperparameters** to choose:

1. Learning rate
2. Regularization constant
3. Number of epochs
4. Number of hidden layers
5. Nodes in each hidden layer
6. Weight initialization strategy
7. Loss function
8. Activation functions
9. ...

:(

Choosing these is a bit of an art.

Good news: in practice many configurations work well

There are some reasonable **heuristics**. E.g

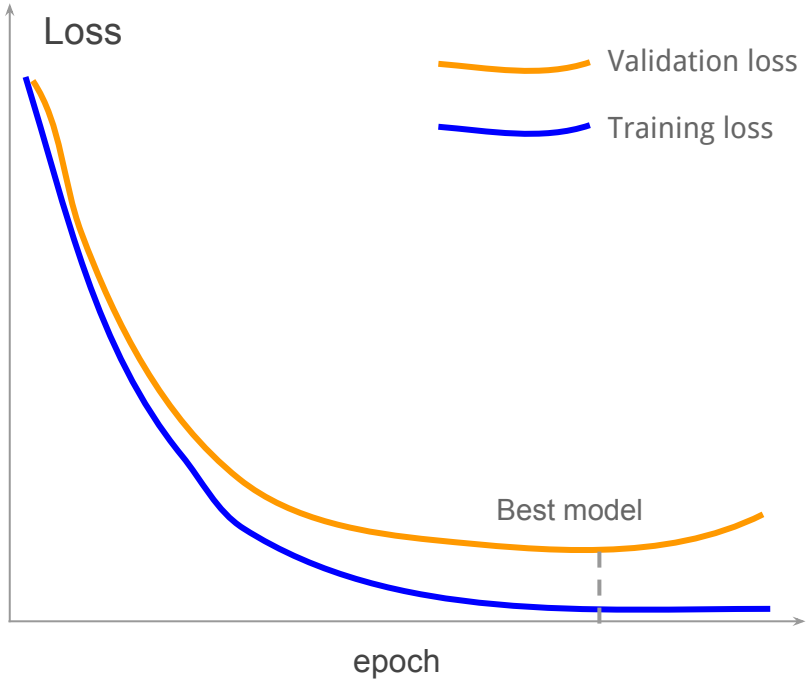
1. Try 0.1 for the learning rate. If this diverges, divide by 3. Repeat.
2. Try an existing network architecture and adapt it for your problem
3. Try overfit the data with a big model, then regularize

You can also do a **hyperparameter search** if you have enough compute:

- Randomized search tends to work well

Training and monitoring progress

1. Split data into train, validation, and test sets
   - Keep 5-30% of data for validation
2. Fit model parameters on train set using SGD
3. After each epoch:
   - **Test model on validation set** and compute loss
     - Also compute whatever other metrics you are interested in, e.g. top-5 accuracy
   - Save a snapshot of the model
4. Plot **learning curves** as training progresses
5. Stop when validation loss starts to increase
6. Use model with minimum validation loss



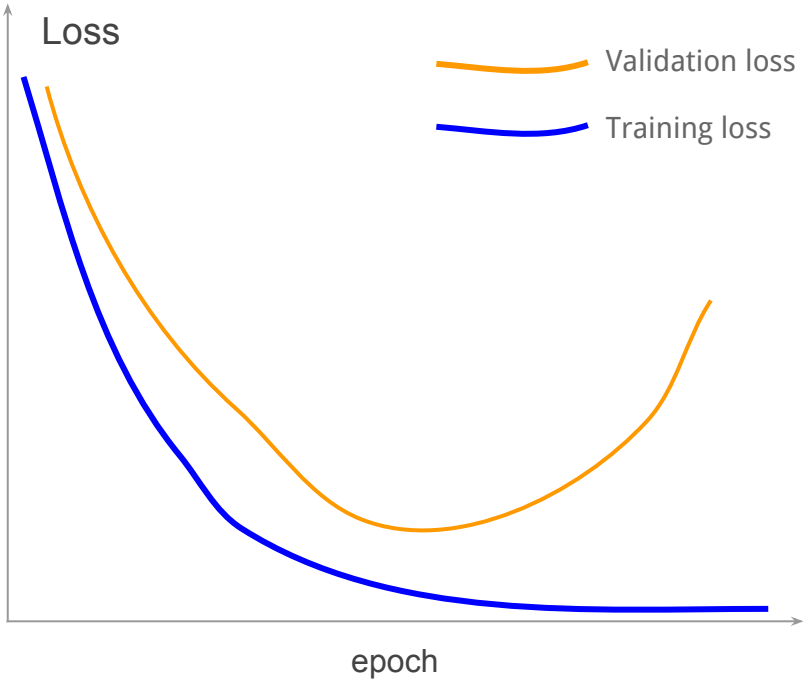
Overfitting (to the training data)

Symptoms:

- Validation loss decreases at first, then starts increasing
- Training loss continues to go down

Try:

- Find more training data
- Add stronger regularization
  - dropout, drop-connect, L^2
- Data augmentation (flips, rotations, noise)
- Reduce complexity of your model



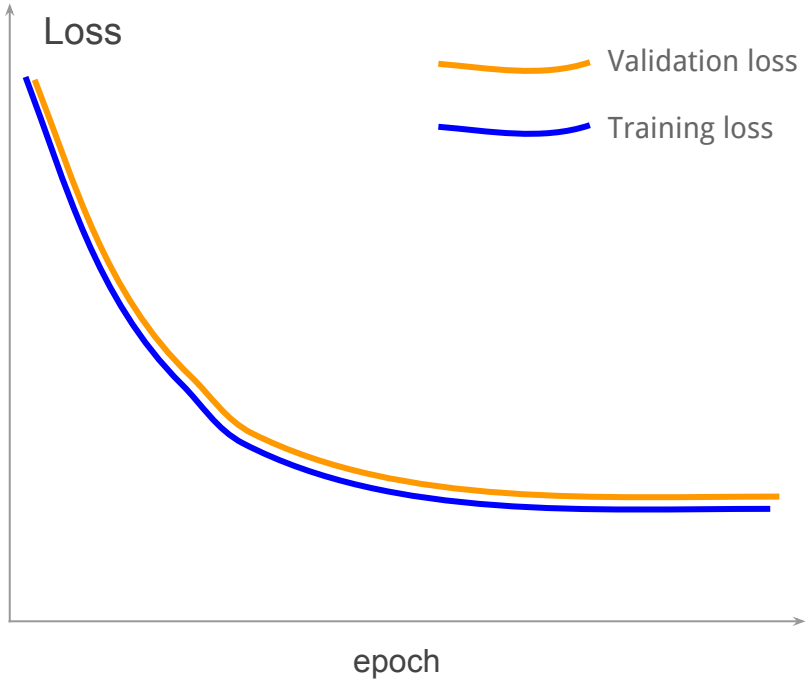
Underfitting (to the training data)

Symptoms:

- Training loss decreases at first but then stops
- Training loss still high
- Training loss tracks validation loss

Try:

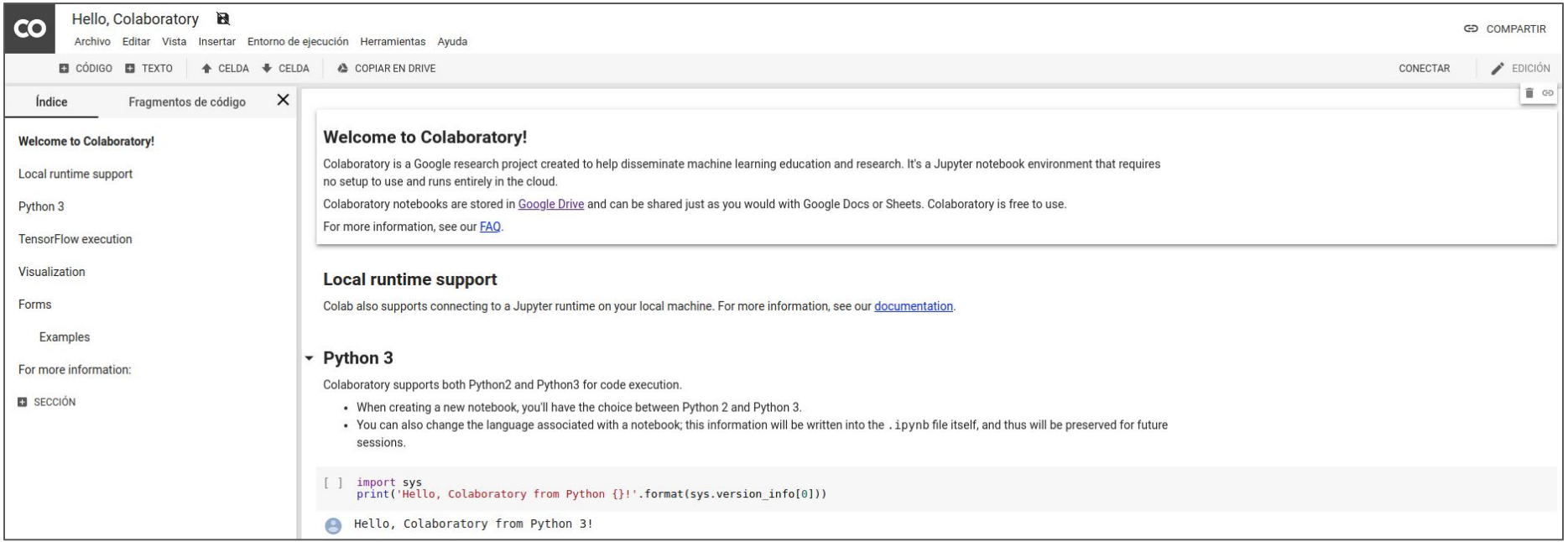
- Increase model capacity
  - Add more layers, increase layer size
- Use more suitable network architecture
  - E.g. multi-scale architecture
- Decrease regularization strength



Today's objectives

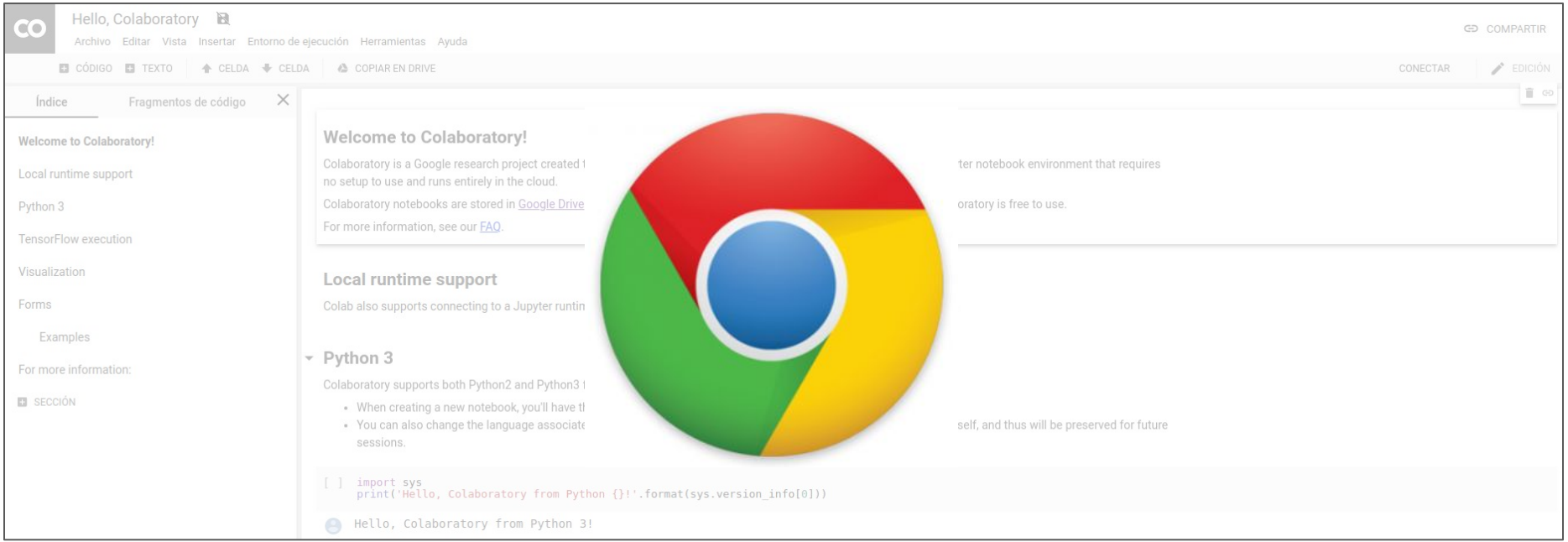
- Underfit a neural network.
- Overfit a neural network.
- Tune a deep neural network by:
  - Modifying the capacity of the model
  - Weight regularization
  - Dropout

Google Colab



https://colab.research.google.com/

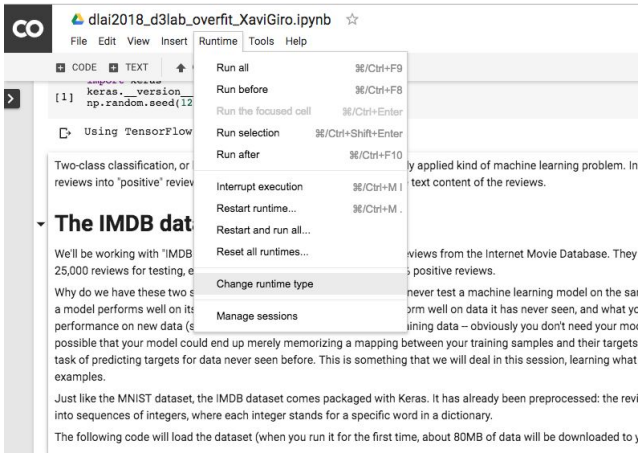
Google Colab

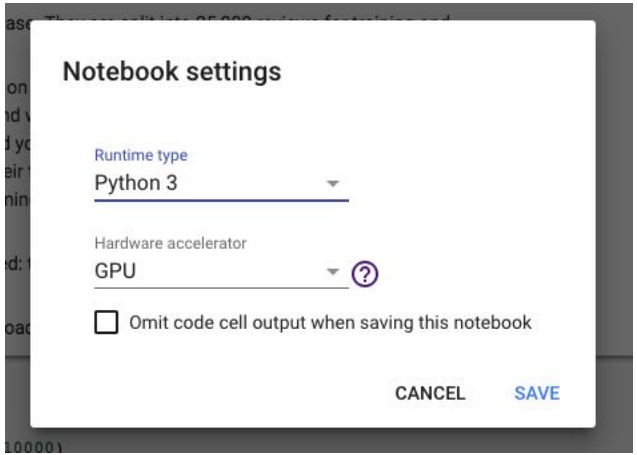


https://colab.research.google.com/

Google Colab

1. Download the notebook of this lab session
2. Login to a Google account: yours or aidlupc2019@gmail.com (talentcenter)
3. From there, open it with Colab
4. Change runtime type to work with GPU! Your trainings will be much faster :)





Final Questions