

#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 Politècnica de Catalunya
Technical University of Catalonia

Lab created by



Míriam Bellver

miriam.bellver@bsc.edu

PhD Candidate

Barcelona Supercomputing Center



Supervised Learning

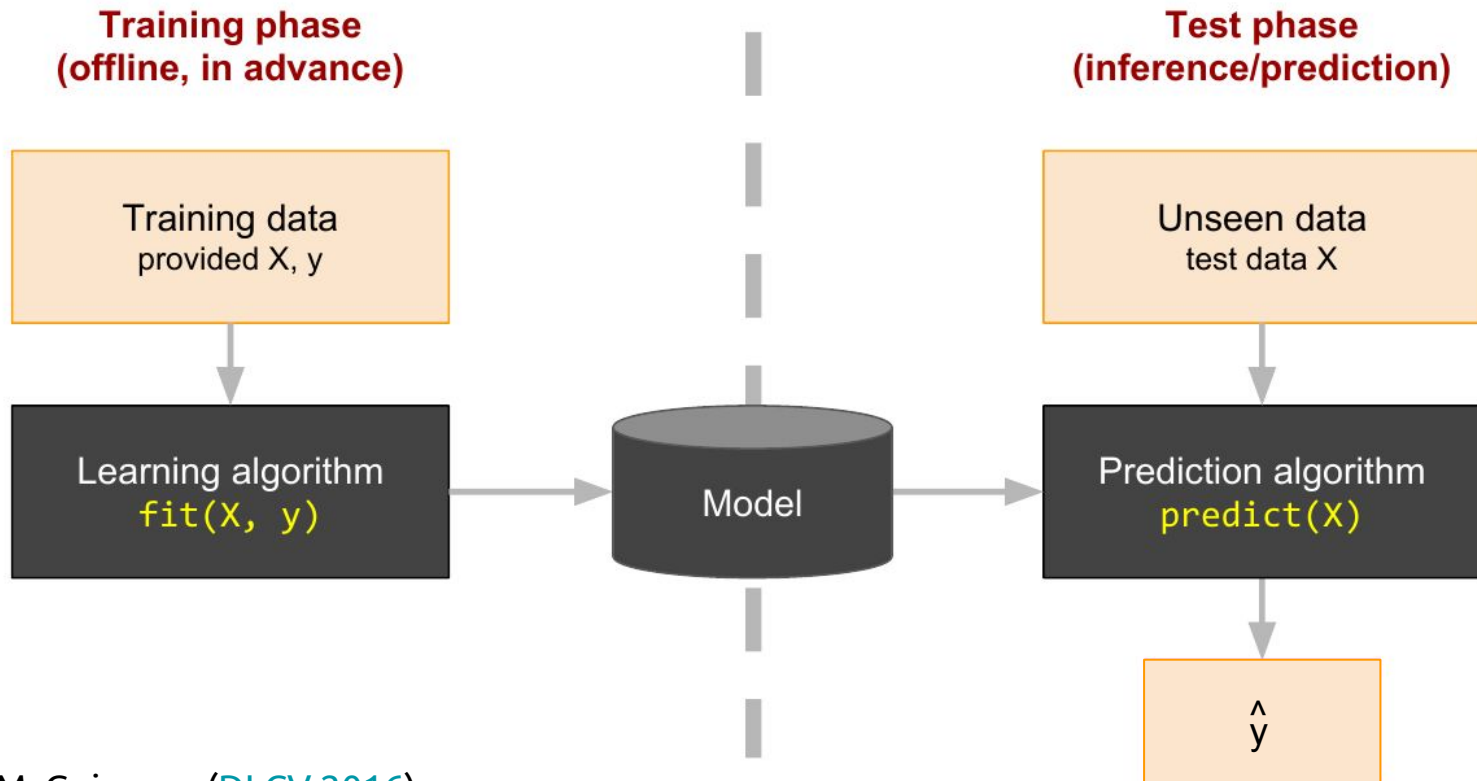
Fit a function: $\mathbf{y} = f(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^m$

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

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

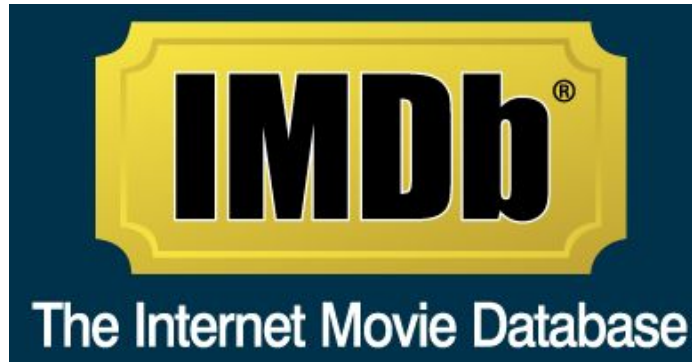


Supervised Learning

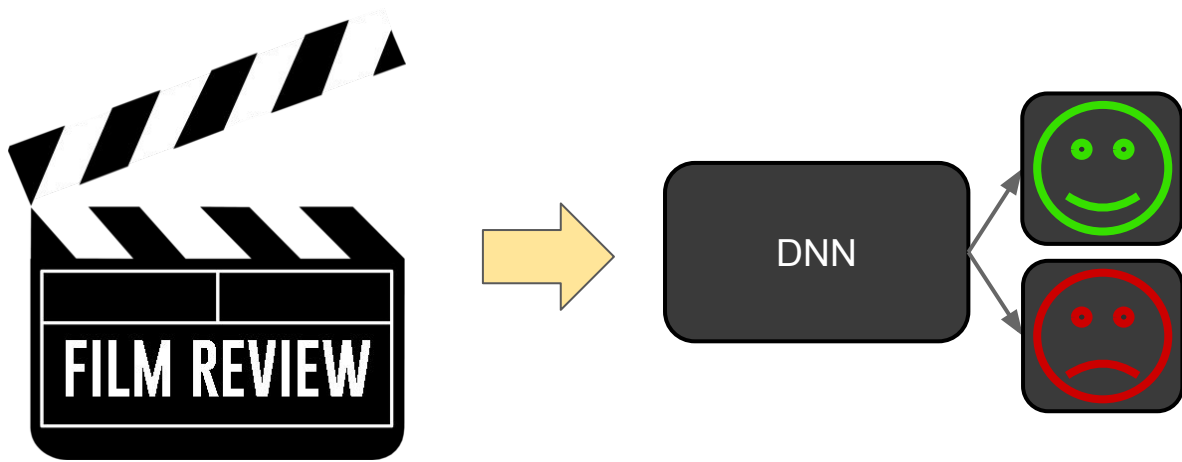


Dataset: IMDB Movies Datasets

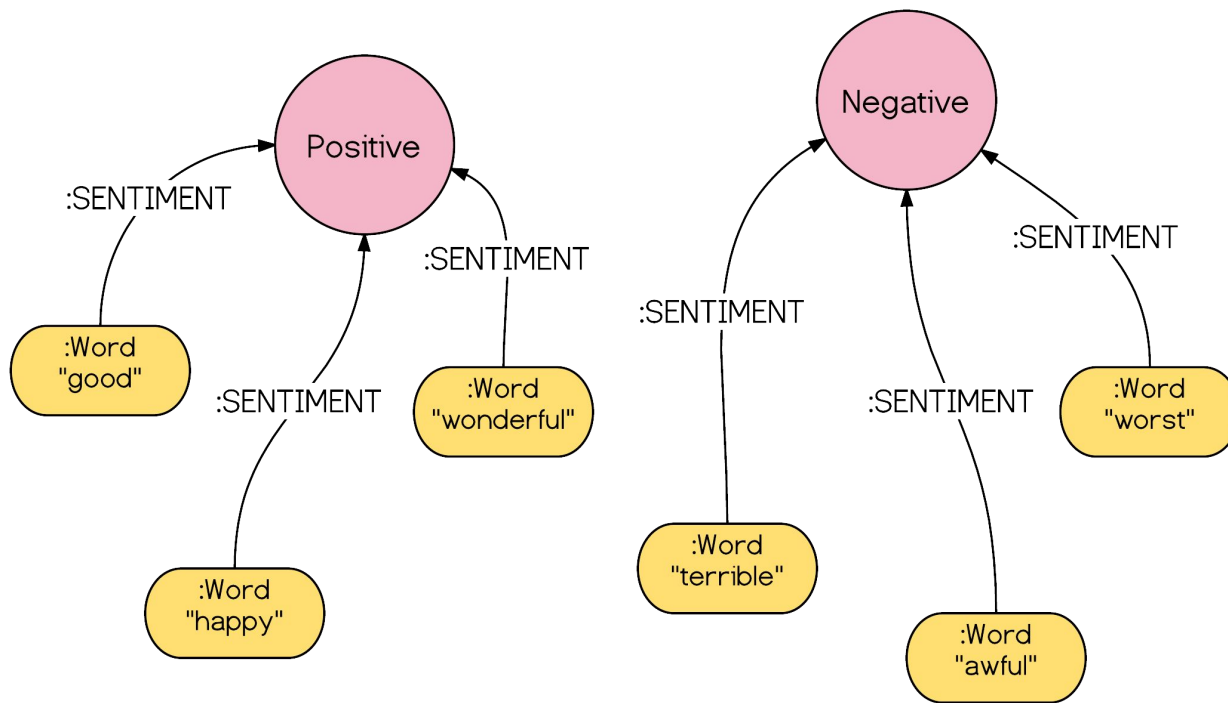
- 14,762 movies
- The purpose of the dataset was to produce movies recommendation systems
- The details that this dataset has are:
 - title, region, language, original title, start year, end year, directors.. lots of information regarding movies.



Task



Task



One hot encoding for Words

cat: $x^T = [1, 0, 0, \dots, 0]$

dog: $x^T = [0, 1, 0, \dots, 0]$

.

.

house: $x^T = [0, 0, 0, \dots, 0, 1, 0, \dots, 0]$

.

.

.

Bag of Words encoding for Reviews

the dog is on the table



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

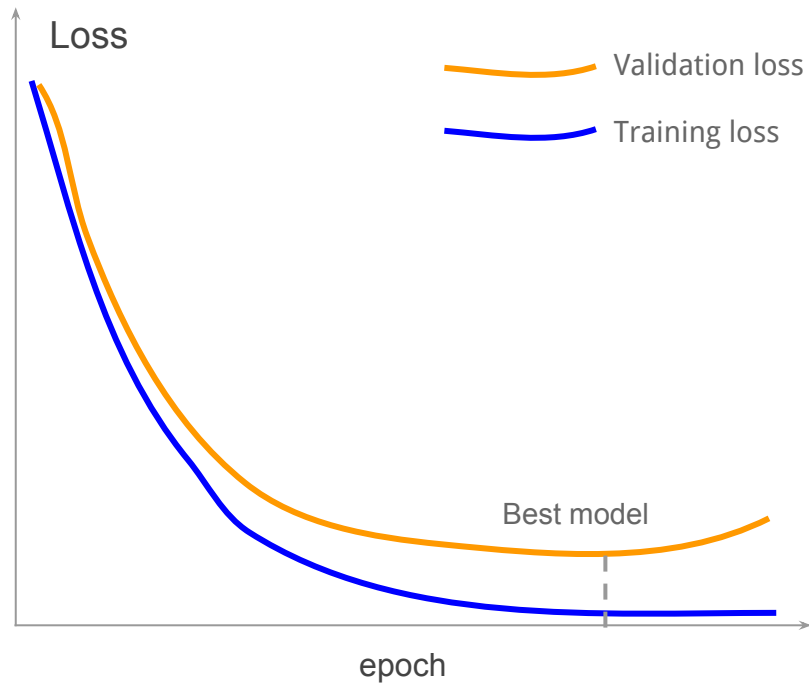
1. Try 0.1 for the learning rate. If this diverges, divide by 3. Repeat.
2. Try an existing network architecture and adapt it for your problem
3. Try overfit the data with a big model, then regularize

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



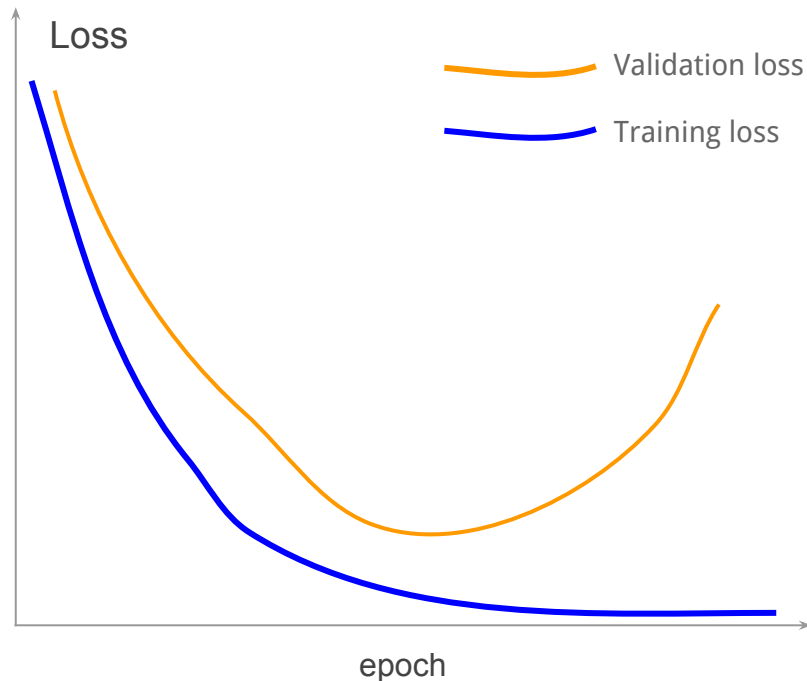
Overfitting (to the training data)

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



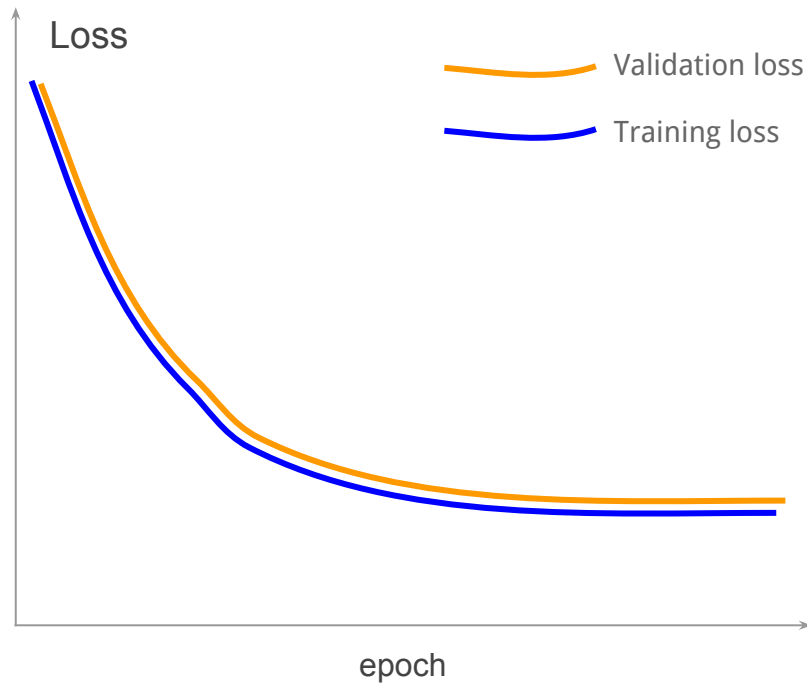
Underfitting (to the training data)

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

The screenshot displays the Google Colaboratory web interface. At the top, there's a header with the 'co' logo and the text 'Hello, Colaboratory'. Below this is a navigation bar with options like 'Archivo', 'Editar', 'Vista', 'Insertar', 'Entorno de ejecución', 'Herramientas', and 'Ayuda'. A secondary bar contains 'CÓDIGO', 'TEXTO', 'CELDA', and 'COPIAR EN DRIVE'. On the right side of the header, there are links for 'CONECTAR' and 'EDICIÓN', and a 'COMPARTIR' button. A left sidebar menu lists various topics: 'Welcome to Colaboratory!', 'Local runtime support', 'Python 3', 'TensorFlow execution', 'Visualization', 'Forms', 'Examples', and 'For more information:'. Below this menu is a 'SECCIÓN' button. The main content area features a 'Welcome to Colaboratory!' section with introductory text about the project and its purpose. This is followed by a 'Local runtime support' section. A 'Python 3' section is expanded, showing a list of bullet points about language choices and file storage. At the bottom, a code cell is shown with a Python script that prints a versioned greeting, which has been executed, resulting in the output 'Hello, Colaboratory from Python 3!'.

co Hello, Colaboratory

Archivo Editar Vista Insertar Entorno de ejecución Herramientas Ayuda

CÓDIGO TEXTO CELDA CELDA COPIAR EN DRIVE

CONECTAR EDICIÓN

COMPARTIR

Índice Fragmentos de código

Welcome to Colaboratory!

Local runtime support

Python 3

TensorFlow execution

Visualization

Forms

Examples

For more information:

SECCIÓN

Welcome to Colaboratory!

Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a Jupyter notebook environment that requires no setup to use and runs entirely in the cloud.

Colaboratory notebooks are stored in [Google Drive](#) and can be shared just as you would with Google Docs or Sheets. Colaboratory is free to use.

For more information, see our [FAQ](#).

Local runtime support

Colab also supports connecting to a Jupyter runtime on your local machine. For more information, see our [documentation](#).

Python 3

Colaboratory supports both Python2 and Python3 for code execution.

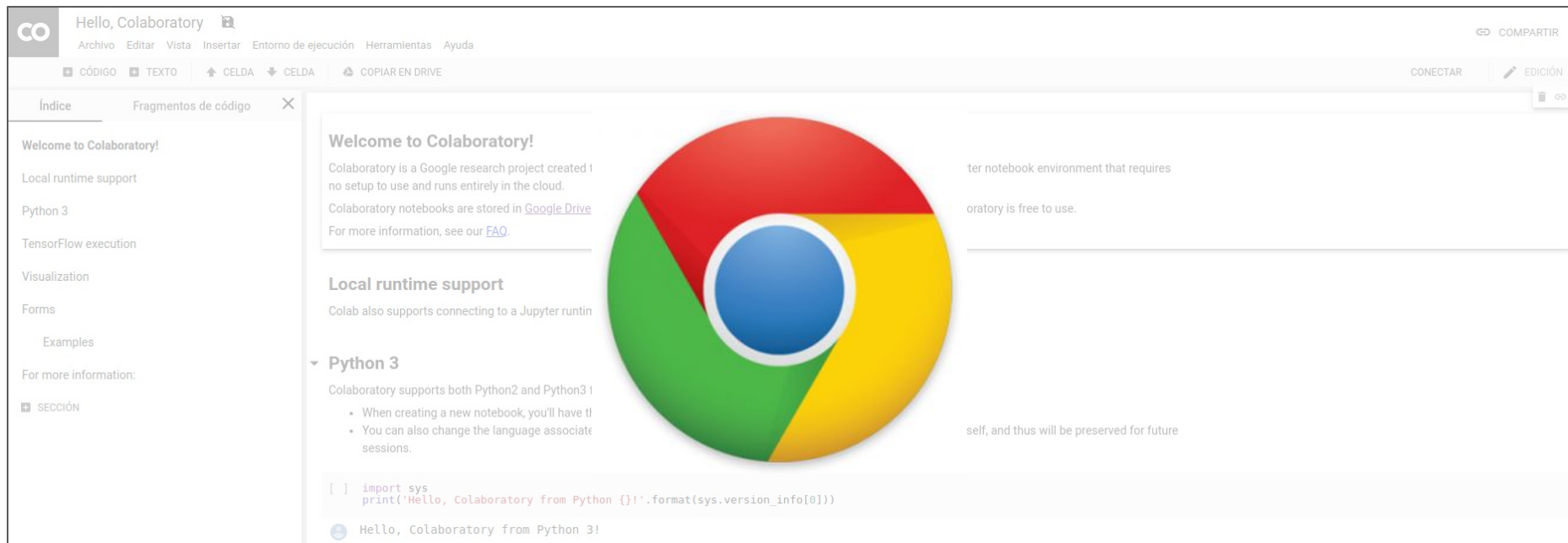
- When creating a new notebook, you'll have the choice between Python 2 and Python 3.
- You can also change the language associated with a notebook; this information will be written into the `.ipynb` file itself, and thus will be preserved for future sessions.

```
[ ] import sys
print('Hello, Colaboratory from Python {}'.format(sys.version_info[0]))
```

Hello, Colaboratory from Python 3!

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

Google Colab



The screenshot displays the Google Colaboratory web interface. At the top, the header includes the 'co' logo, the title 'Hello, Colaboratory', and a menu with options: Archivo, Editar, Vista, Insertar, Entorno de ejecución, Herramientas, and Ayuda. On the right, there are links for 'COMPARTIR' and 'CONECTAR', and a button for 'EDICIÓN'. Below the header, a toolbar shows icons for 'CÓDIGO', 'TEXTO', 'CELDA', and 'COPIAR EN DRIVE'. The left sidebar contains a table of contents with links to 'Índice', 'Fragmentos de código', 'Welcome to Colaboratory!', 'Local runtime support', 'Python 3', 'TensorFlow execution', 'Visualization', 'Forms', 'Examples', and 'For more information:'. The main content area features a large, colorful circular logo in the center. To the left of the logo, the 'Welcome to Colaboratory!' section explains that Colaboratory is a Google research project that runs entirely in the cloud and that notebooks are stored in Google Drive. Below this, the 'Local runtime support' section mentions that Colab also supports connecting to a Jupyter runtime. The 'Python 3' section states that Colaboratory supports both Python 2 and Python 3, and lists two bullet points: 'When creating a new notebook, you'll have to...' and 'You can also change the language associated with the notebook for future sessions.' To the right of the logo, there is a text box that says 'ter notebook environment that requires' and 'oratory is free to use.' At the bottom, a code cell is shown with the following code:

```
[ ] import sys
print('Hello, Colaboratory from Python {}'.format(sys.version_info[0]))
```

 Below the code, there is a button with a plus sign and the text 'Hello, Colaboratory from Python 3!'.

co Hello, Colaboratory

Archivo Editar Vista Insertar Entorno de ejecución Herramientas Ayuda

CÓDIGO TEXTO CELDA CELDA COPIAR EN DRIVE

COMPARTIR CONECTAR EDICIÓN

Índice Fragmentos de código

Welcome to Colaboratory!

Colaboratory is a Google research project created in 2015. It is a free, open-source, and no setup to use and runs entirely in the cloud. Colaboratory notebooks are stored in [Google Drive](#). For more information, see our [FAQ](#).

Local runtime support

Colab also supports connecting to a Jupyter runtime.

Python 3

Colaboratory supports both Python2 and Python3!

- When creating a new notebook, you'll have to...
- You can also change the language associated with the notebook for future sessions.

ter notebook environment that requires

oratory is free to use.

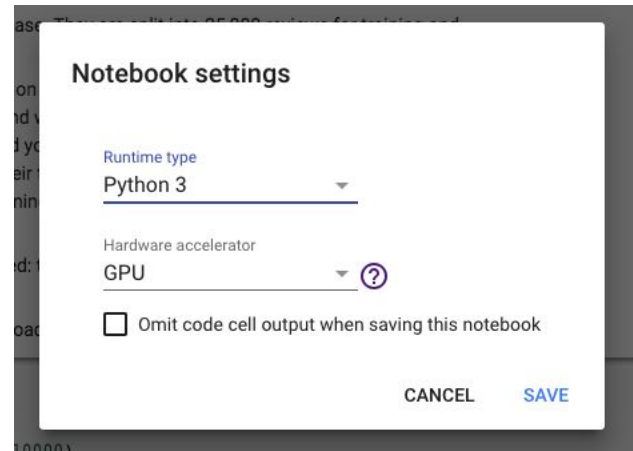
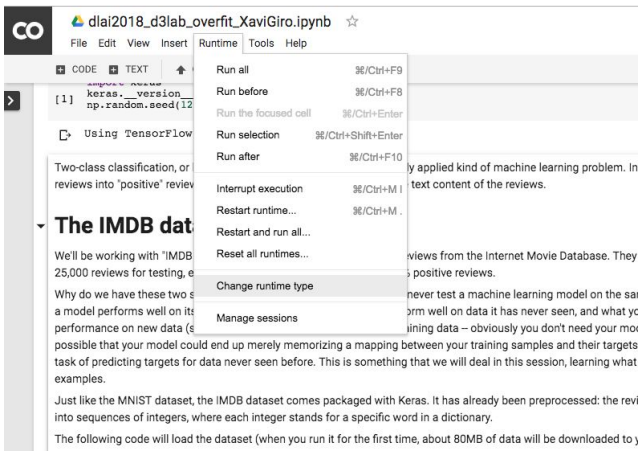
```
[ ] import sys
print('Hello, Colaboratory from Python {}'.format(sys.version_info[0]))
```

Hello, Colaboratory from Python 3!

<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

Undergradese

What undergrads ask vs. what they're REALLY asking

"Is it going to be an open book exam?"

Translation: "I don't have to actually memorize anything, do I?"

"Hmm, what do you mean by that?"

Translation: "What's the answer so we can all go home."

"Are you going to have office hours today?"

Translation: "Can I do my homework in your office?"

"Can i get an extension?"

Translation: "Can you re-arrange your life around mine?"

"Is this going to be on the test?"

Translation: "Tell us what's going to be on the test."

"Is grading going to be curved?"

Translation: "Can I do a mediocre job and still get an A?"

JORGE CHAM © 2008



WWW.PHDCOMICS.COM