**PROJECT**

Project Explainable COVID-19 Pneumonia

**PROJECT CONTRIBUTORS**

Fernando Rios

Hassan Ouanir

Maha Faruque

Introduction:

In this project, the goals are: (1) to explore development of a machine learning algorithm to distinguish chest X-rays of individuals with respiratory illness testing positive for COVID-19 from other X-rays, (2) to promote discovery of patterns in such X-rays via machine learning interpretability algorithms.

Due to doctors are reluctant to accept black box algorithms such as your deep learning based method - as an AI engineer we need to listen to them and try to satisfy their needs, they are your customer after all. They tell you that your automated diagnostic system that processes the imaging they give you, must be *explainable*.

We need to follow theses tasks to achieve the goal:

They give you [the COVID X-ray / CT Imaging dataset](https://github.com/ieee8023/covid-chestxray-dataset) and:

1. First you find this [this implementation](https://github.com/aildnont/covid-cxr) of the method called Local Interpretable Model-Agnostic Explanations (i.e. LIME). You also read [this article](https://towardsdatascience.com/investigation-of-explainable-predictions-of-covid-19-infection-from-chest-x-rays-with-machine-cb370f46af1d) and you get your hands dirty and replicate the results in your colab notebook with GPU enabled kernel(40%).
2. A fellow AI engineer, tells you about another method called [SHAP](https://arxiv.org/abs/1705.07874) that stands for SHapley Additive exPlanations and she mentions that Shapley was a Nobel prize winner so it must be important. You then find out that [Google is using it and wrote a readable white paper](https://storage.googleapis.com/cloud-ai-whitepapers/AI%20Explainability%20Whitepaper.pdf) about it and your excitement grows. Your manager sees you on the corridor and mentions that your work is needed soon. You are keen to impress her and start writing your 3-5 page summary of the SHAP approach as can be applied to explaining deep learning classifiers such as the ResNet network used in (1). (40%)
3. After your presentation, your manager is clearly impressed with the depth of the SHAP approach and asks for some results for explaining the COVID-19 diagnoses via it. You notice that the extremely popular [SHAP Github repo](https://github.com/slundberg/shap) already has an example with VGG16 network applied to ImageNet. You think it wont be too difficult to plugin the model you trained in (1) and explain it. (20%)

Part I: Replicate repository

* Step 1.0 Cloning repository covid-cxr

# Mounting Google Drive

from google.colab import drive

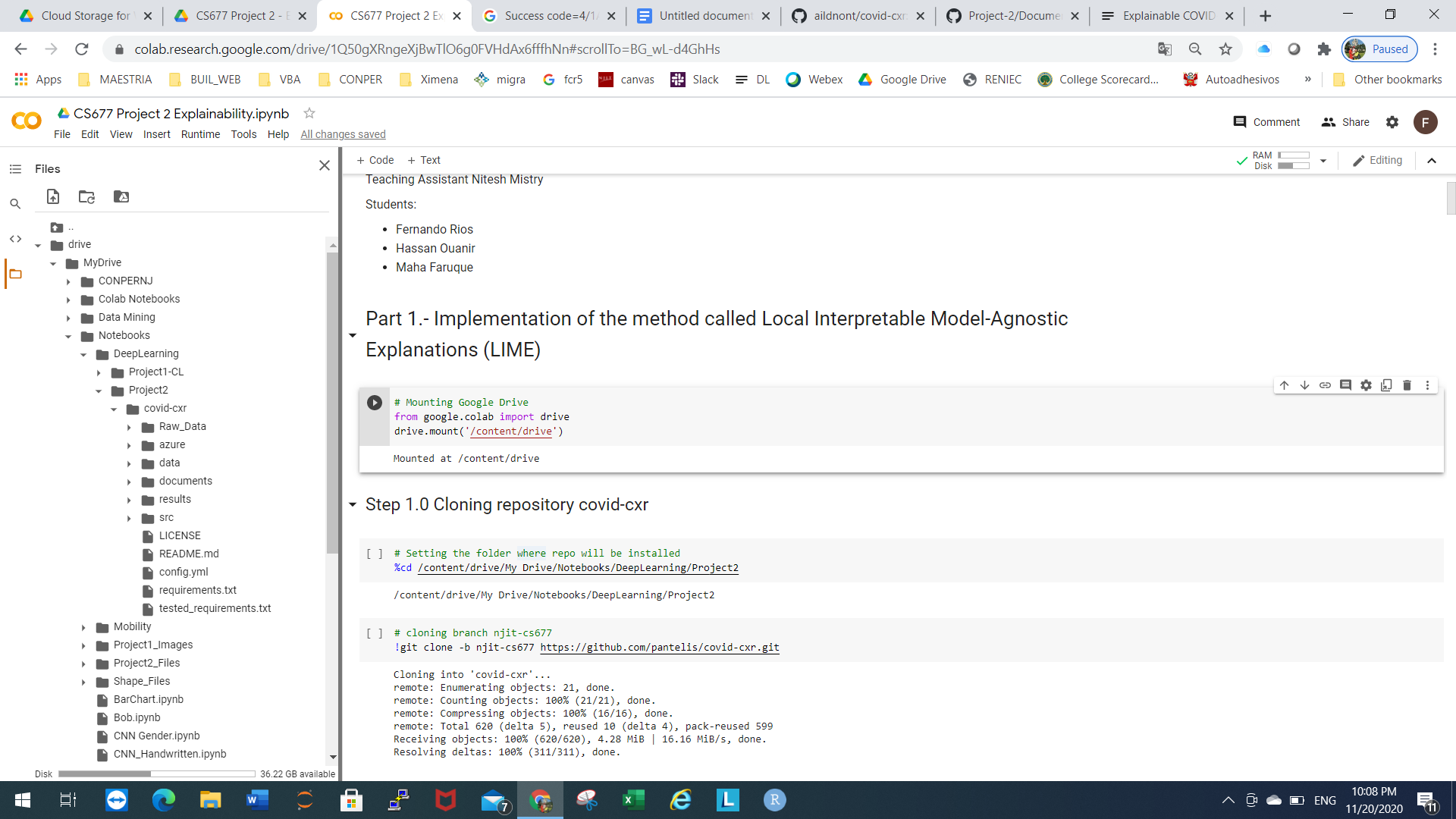
drive.mount('/content/drive')

# Setting the folder where repo will be installed

%cd /content/drive/My Drive/Notebooks/DeepLearning/Project2

# cloning branch njit-cs677

!git clone -b njit-cs677 https://github.com/pantelis/covid-cxr.git



* Step 2.0 Installing requirements

!pip install -r tested\_requirements.txt

* Step 3.0 Creating folder Raw\_Data to contain all of your raw data. Set the RAW\_DATA field in the PATHS

section of config.yml to the address of this new folder.

# Creating Raw Data Folder

!mkdir Raw\_Data

* Step 4.0 Cloning the covid-chestxray-dataset repository inside of the RAW\_DATA folder.

!git clone https://github.com/ieee8023/covid-chestxray-dataset

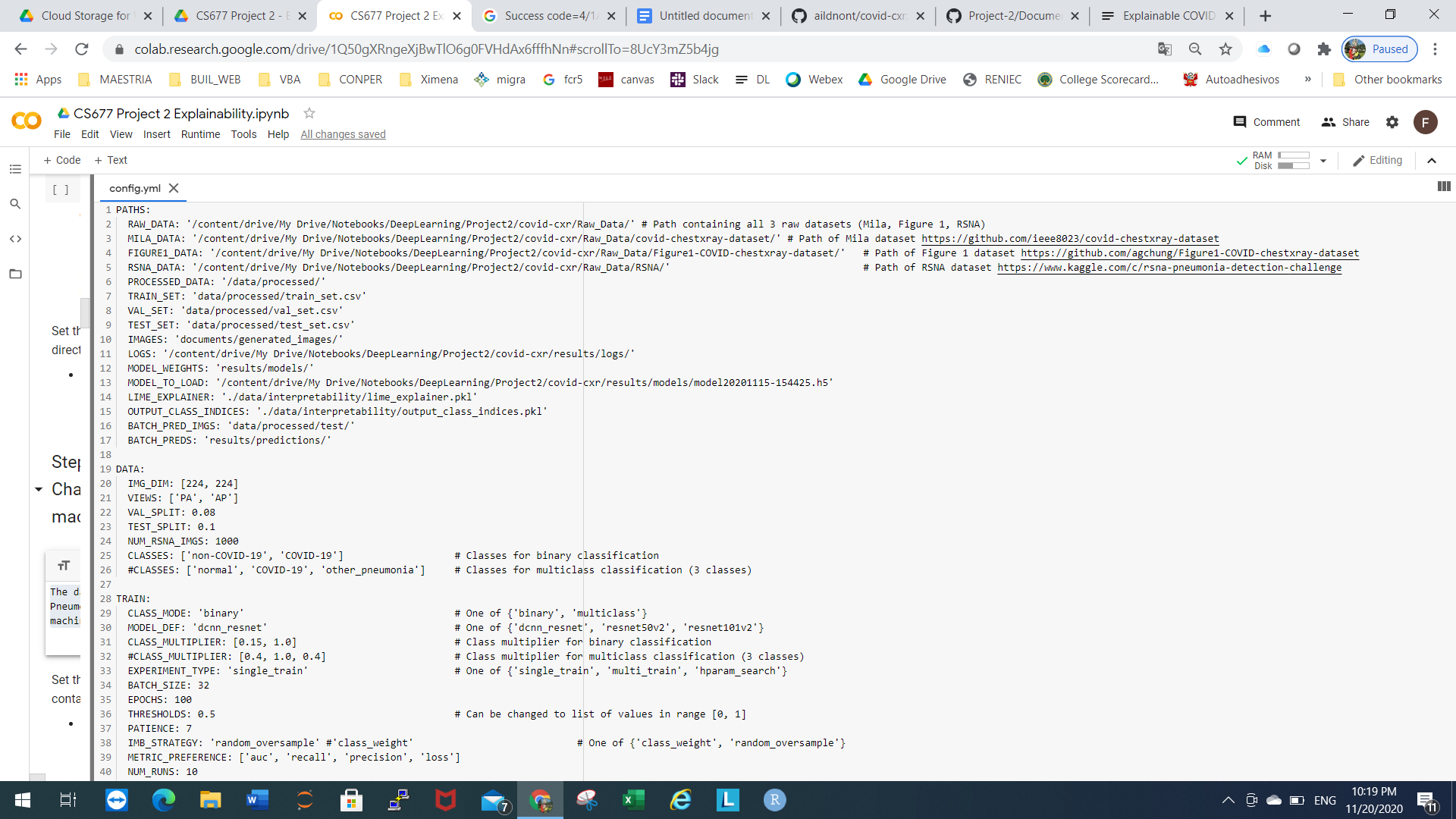
* Step 5.0 Cloning the Figure1-COVID-chestxray-dataset repository inside of RAW\_DATA folder.

!git clone https://github.com/agchung/Figure1-COVID-chestxray-dataset

* Step 6.0 Download and unzip the RSNA Pneumonia Detection Challenge dataset from Kaggle somewhere on your local machine.

The data was downloaded and unzip RSNA Pneumonia dataset from Kaggle to local machine, then uploaded to gdrive manually.

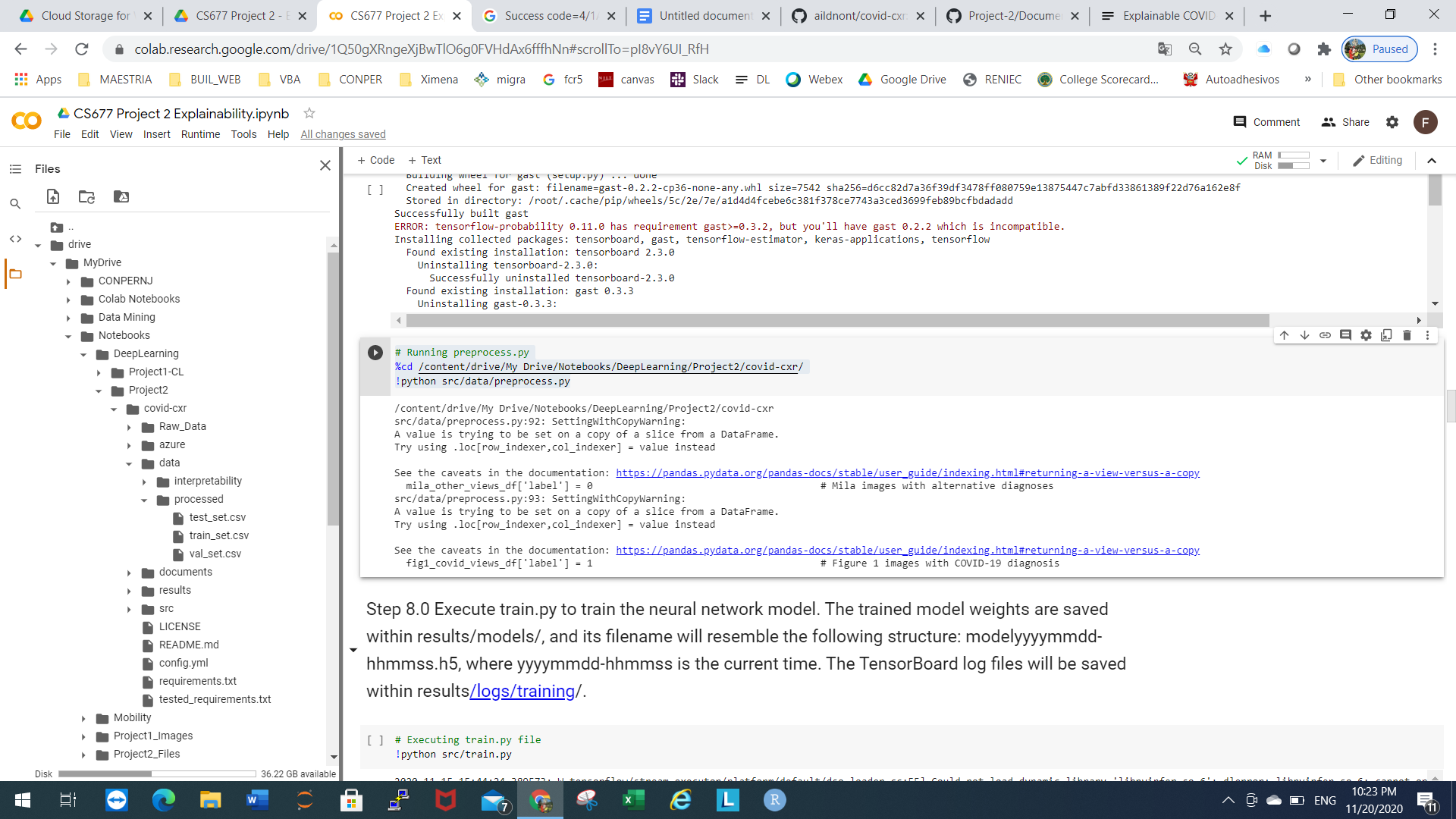
After this step the file config.yml looks like this:



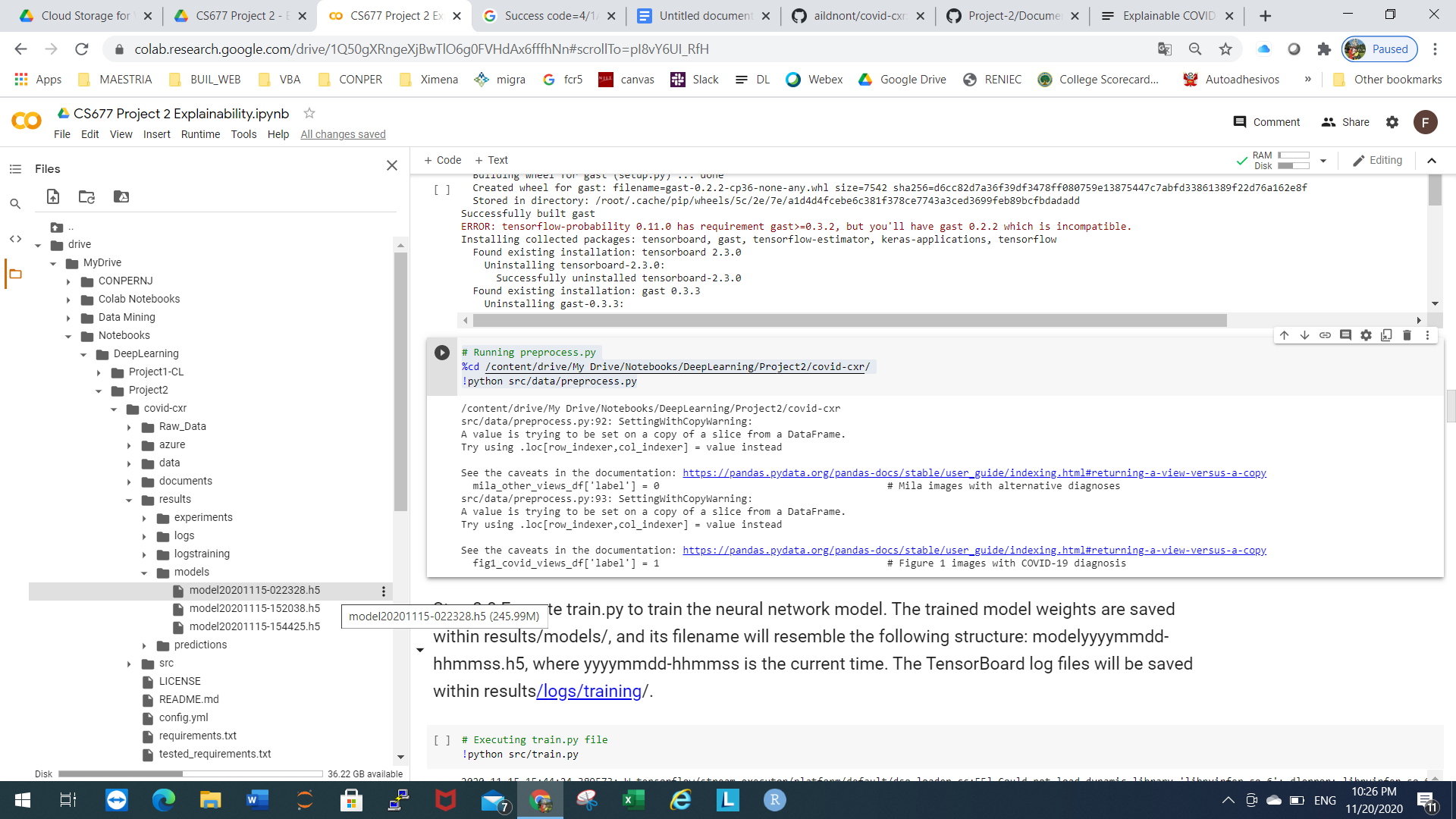
* Step 7.0 Execute preprocess.py to create Pandas DataFrames of filenames and labels. Preprocessed DataFrames and corresponding images of the dataset will be saved within data/processed/.

%cd /content/drive/My Drive/Notebooks/DeepLearning/Project2/covid-cxr/

!python src/data/preprocess.py

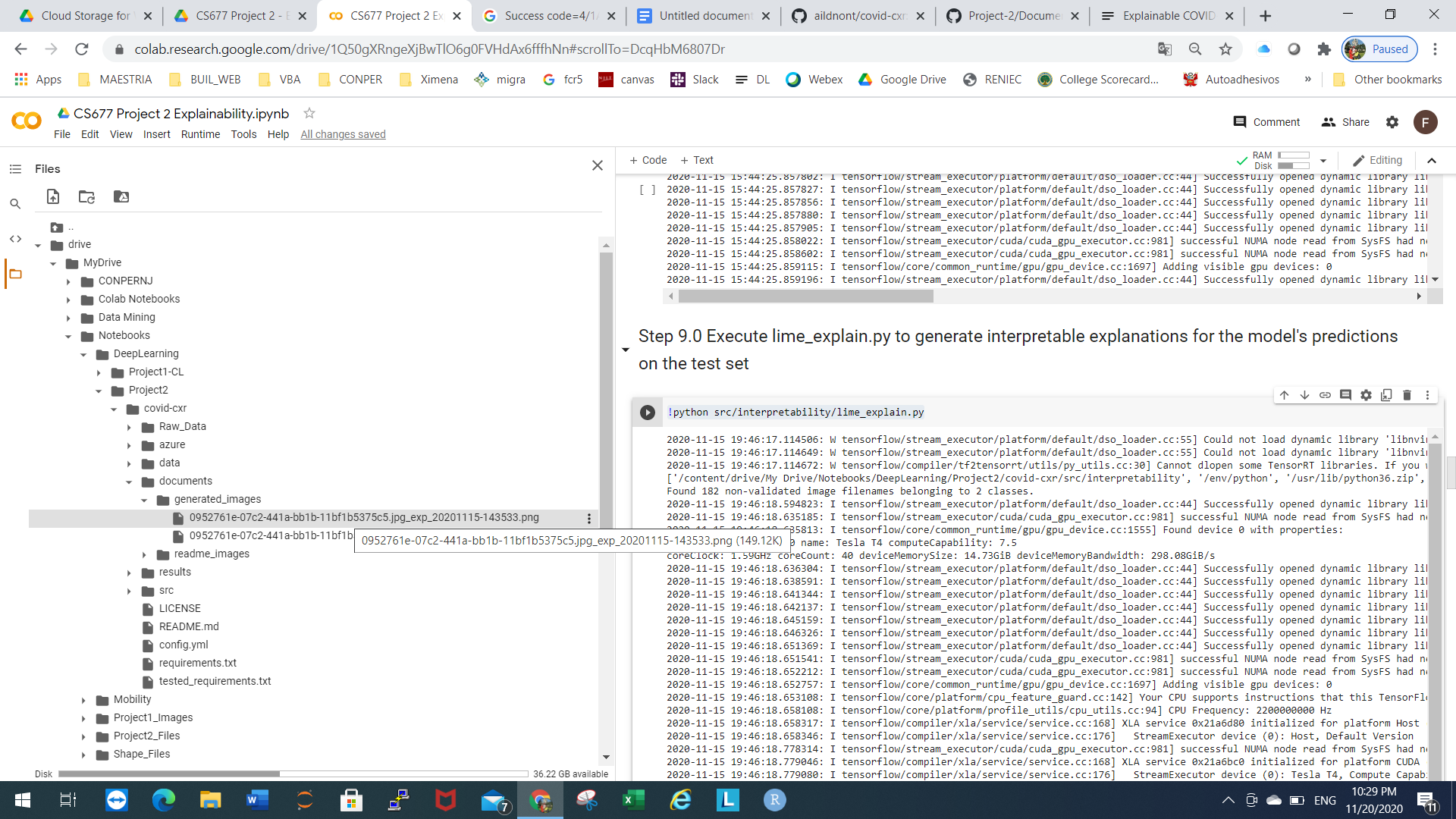


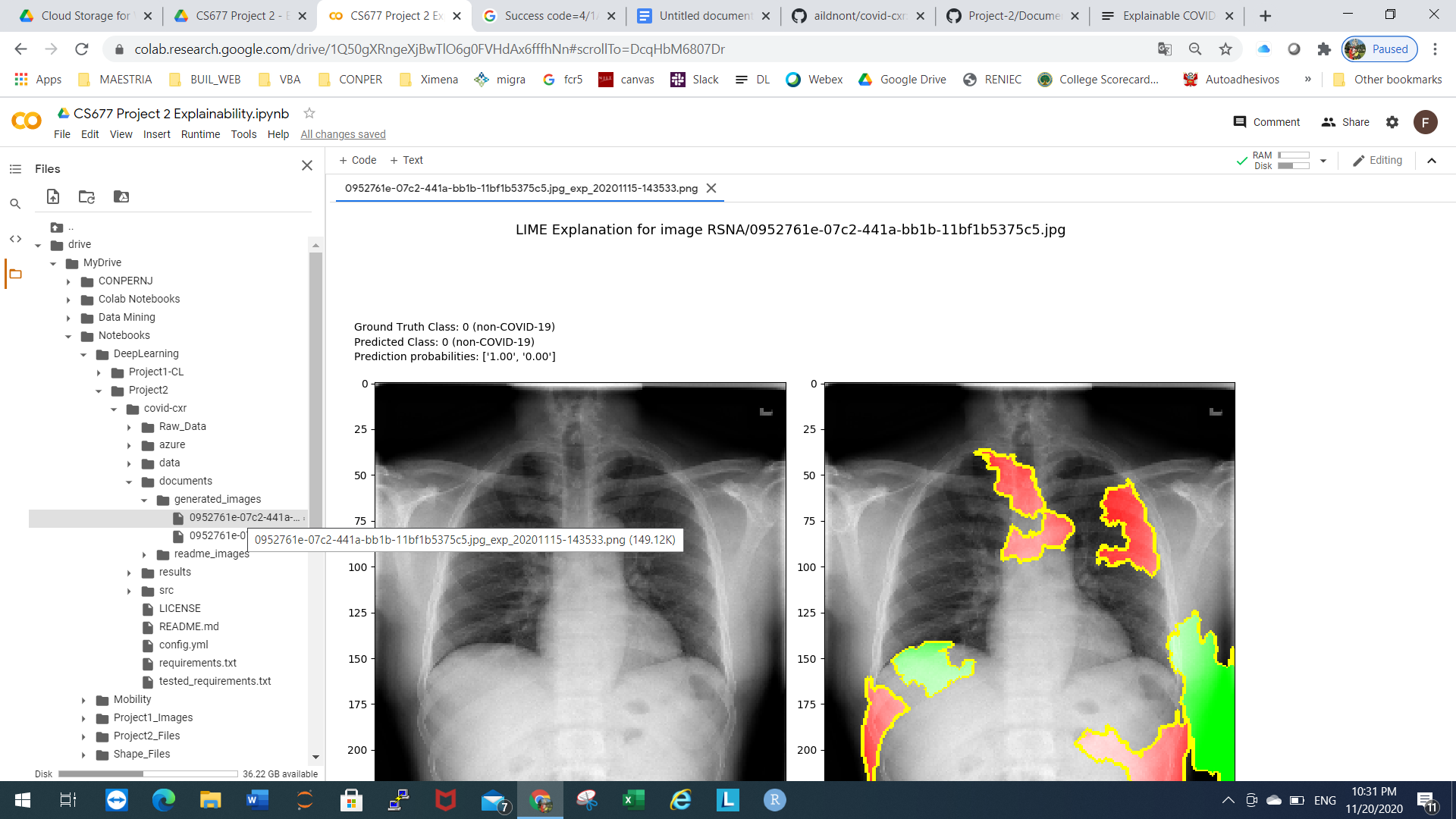
* Step 8.0 Execute train.py to train the neural network model. The trained model weights are saved within results/models/, and its filename will resemble the following structure: modelyyyymmdd-hhmmss.h5, where yyyymmdd-hhmmss is the current time. The TensorBoard log files will be saved within results/logs/training/.



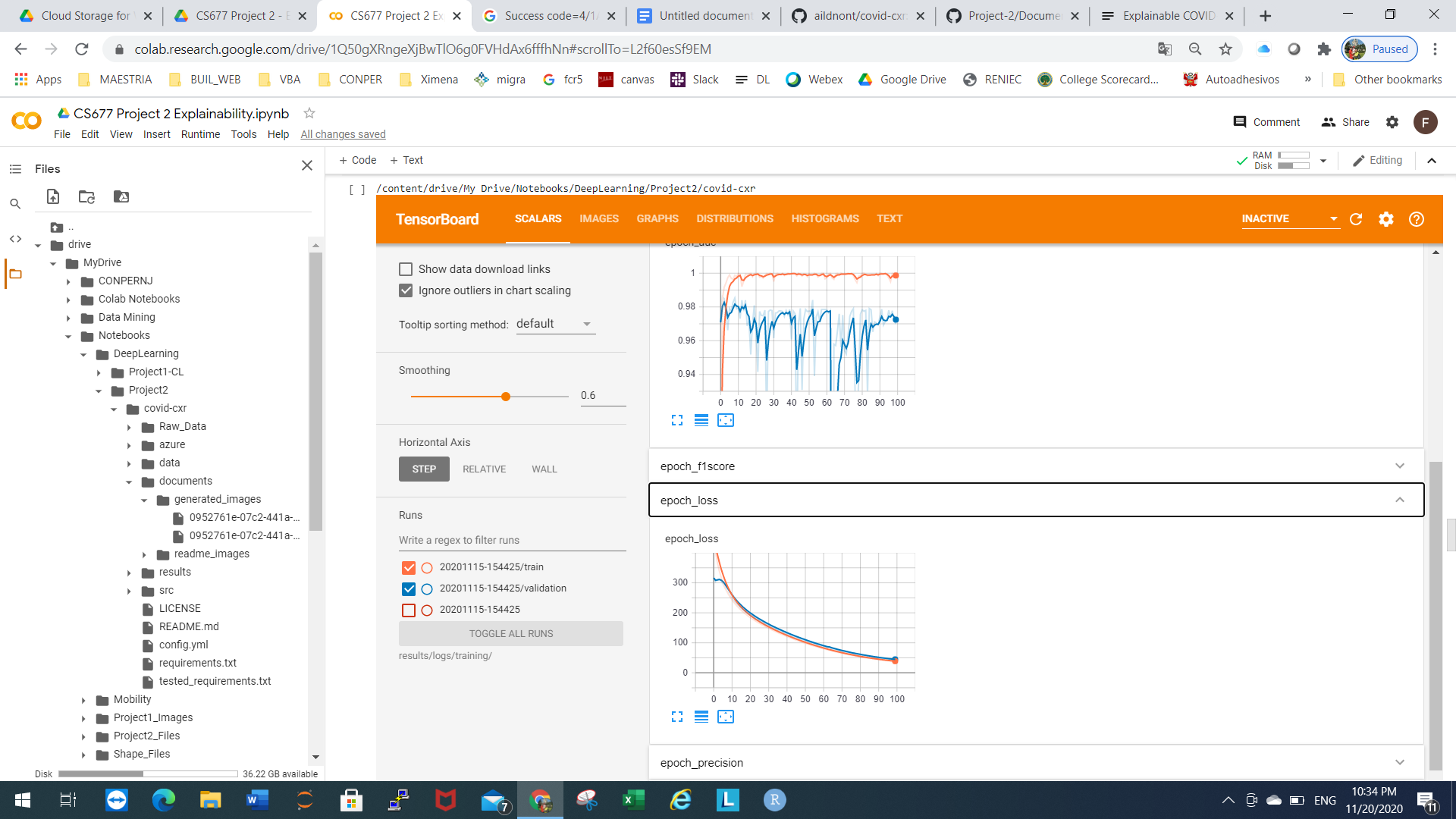
* Step 9.0 Execute lime\_explain.py to generate interpretable explanations for the model's predictions on the test set.

!python src/interpretability/lime\_explain.py

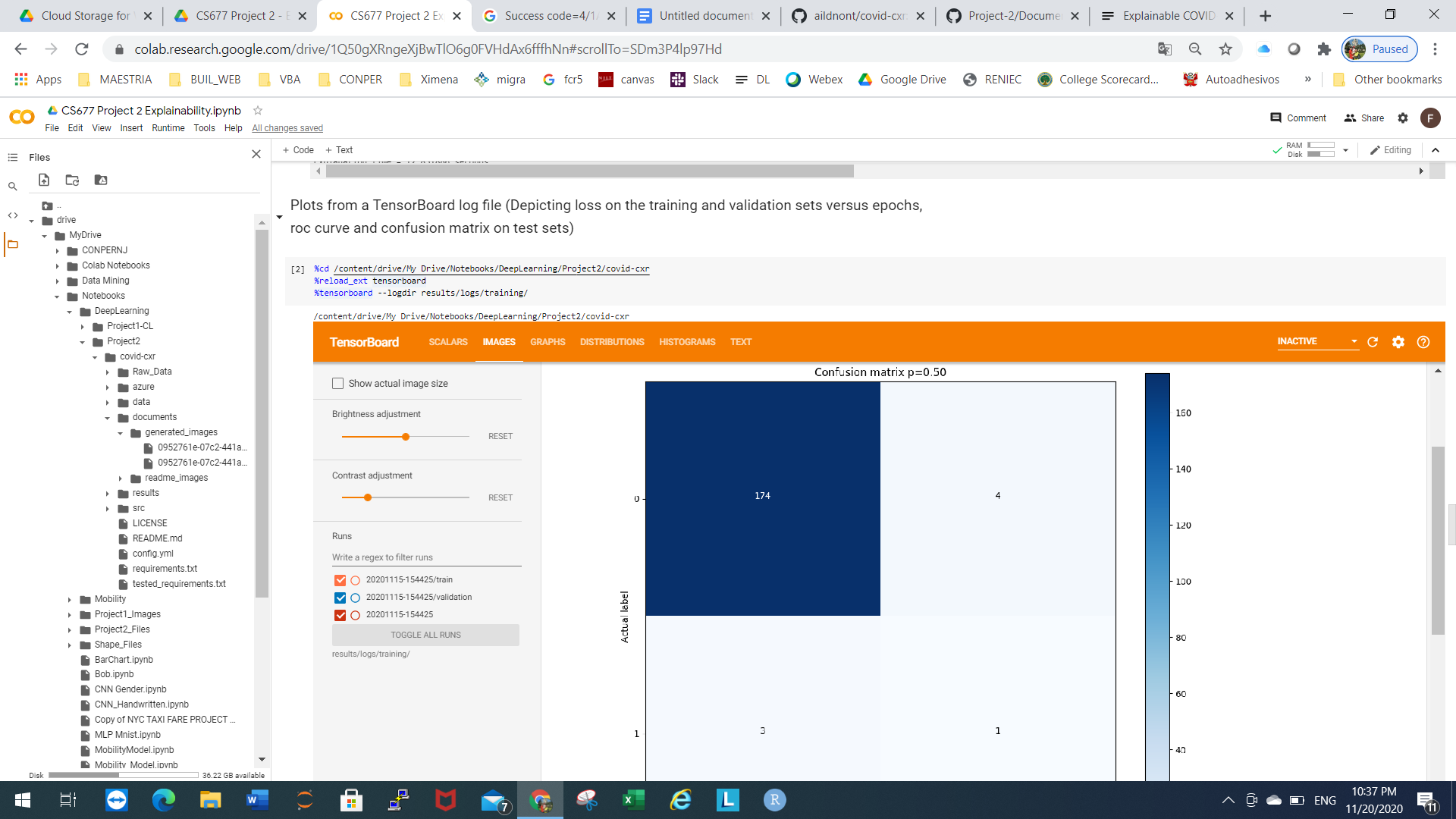


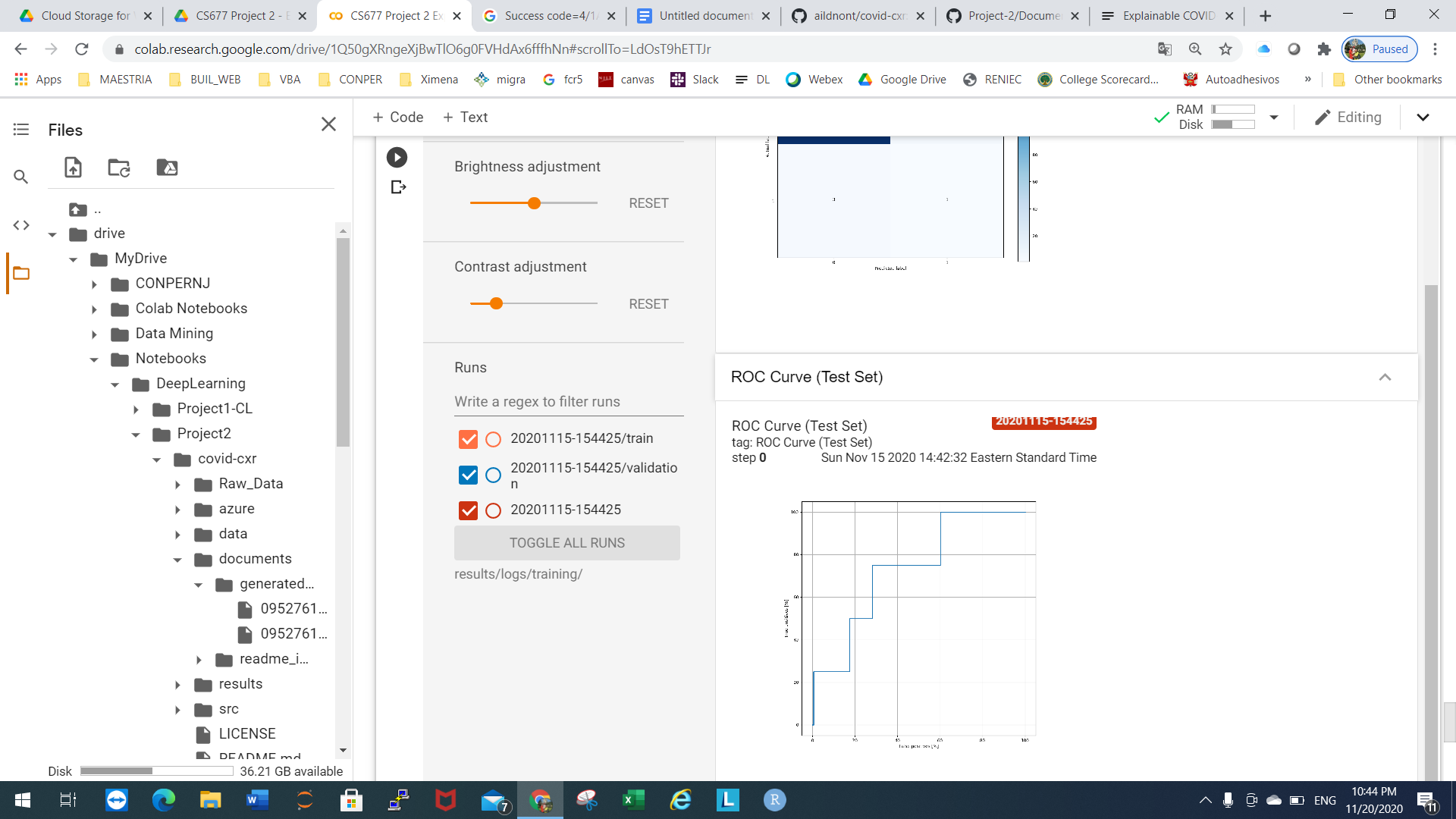


Plots from a TensorBoard log file (Depicting loss on the training and validation sets versus epochs, roc curve and confusion matrix on test sets)



Confusion matrix





Part II.- Summary of the SHAP approach

See summary in “Part 2 - SHAP Approach Summary .docx”

Part III.- Explaining the COVID-19 diagnoses via SHAP Explanation (See complete code in notebook)

%cd /content/drive/My Drive/Notebooks/DeepLearning/Project2/covid-cxr

cfg = yaml.full\_load(open("config.yml", 'r'))

input\_shape = cfg['DATA']['IMG\_DIM'] + [3]

thresholds = 1.0 / len(cfg['DATA']['CLASSES'])

covid\_class\_idx = 0

n\_classes = 2

output\_bias = [0.0, 0.0]

num\_gpus = 1

metrics = None

model = dcnn\_resnet(cfg['NN']['DCNN\_BINARY'], input\_shape, metrics, 2, output\_bias=output\_bias, gpus=num\_gpus)

# Calling model saved previously (see part 1)

model.load\_weights('results/models/model20201115-154425.h5')

from keras.preprocessing import image

img\_width, img\_height = 224, 224

img = image.load\_img('Raw\_Data/covid-chestxray-dataset/images/000001-2.jpg', target\_size=(img\_width, img\_height))

to\_explain = image.img\_to\_array(img).reshape(1, img\_width, img\_height, 3)

X = to\_explain.copy().reshape(1, img\_width, img\_height, 3)

class\_names = { '0': ['negative', 'no-covid'], '1': ['positive', 'covid'] }

def map2layer(x, layer):

feed\_dict = dict(zip([model.layers[0].input], [preprocess\_input(x.copy())]))

return tf.compat.v1.keras.backend.get\_session().run(model.layers[layer].input, feed\_dict)

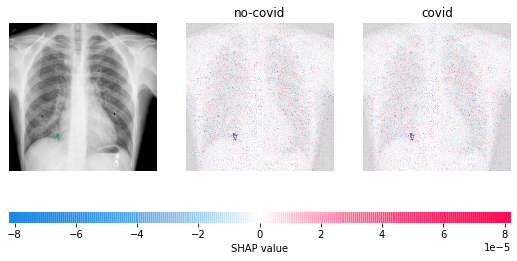
e = shap.GradientExplainer((model.layers[7].input, model.layers[-1].output), map2layer(preprocess\_input(X.copy()), 7))

shap\_values,indexes = e.shap\_values(map2layer(to\_explain, 7), ranked\_outputs=2)

index\_names = np.vectorize(lambda x: class\_names[str(x)][1])(indexes)

# plot the explanations

shap.image\_plot(shap\_values, to\_explain, index\_names)



**Explain with local smoothing**

