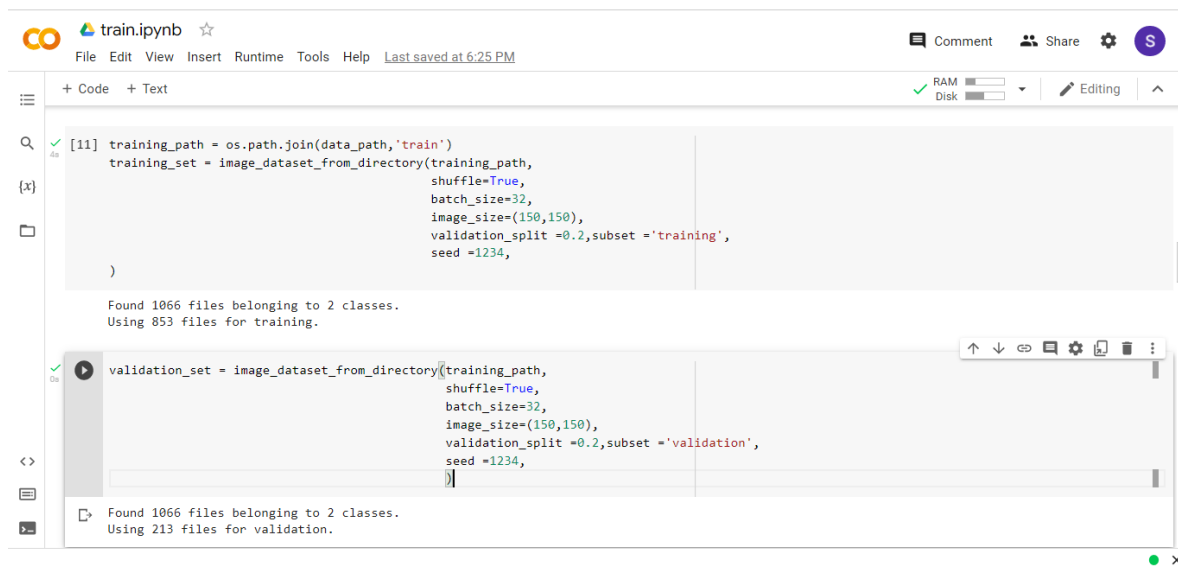


## Proyecto integrador: Machine Learning.

1. GitHub: [https://github.com/SaraCastril1/Pl\\_MLProject.git](https://github.com/SaraCastril1/Pl_MLProject.git)
2. Nuevo modelo → EfficientNet B1 →NOTA: Los cambios del nuevo modelo se encuentran en la rama EfficientNetB1\_model del repositorio de GitHub



The screenshot shows a Jupyter Notebook interface with the following content:

- Top Bar:** Includes the 'train.ipynb' title, a star icon, and navigation links: File, Edit, View, Insert, Runtime, Tools, Help. A status bar on the right shows 'RAM' and 'Disk' usage, and a 'Comment' button.
- Code Cell 1:** Contains the following Python code:

```
[11] training_path = os.path.join(data_path, 'train')
      training_set = image_dataset_from_directory(training_path,
                                                  shuffle=True,
                                                  batch_size=32,
                                                  image_size=(150,150),
                                                  validation_split =0.2,subset = 'training',
                                                  seed =1234,
                                                  )
```

Below the code, the output is displayed: "Found 1066 files belonging to 2 classes. Using 853 files for training."
- Code Cell 2:** Contains the following Python code:

```
validation_set = image_dataset_from_directory(training_path,
                                              shuffle=True,
                                              batch_size=32,
                                              image_size=(150,150),
                                              validation_split =0.2,subset = 'validation',
                                              seed =1234,
                                              )
```

Below the code, the output is displayed: "Found 1066 files belonging to 2 classes. Using 213 files for validation."

✓ [13] training\_set.class\_names

['cat', 'dog']

✓ 1m

```
class_names = training_set.class_names
plt.figure(figsize=(10,10))
for images, labels in training_set.take(1):
    for i in range(9):
        ax = plt.subplot(3,3, i + 1)
        plt.imshow(images[i].numpy().astype("uint8"))
        plt.title(class_names[labels[i]])
        plt.axis("off")
```



✓ 4s [15] base\_model = keras.applications.EfficientNetB1(  
 weights='imagenet',  
 input\_shape = (150,150,3),  
 include\_top=False,  
 )  
base\_model.trainable =False

Downloading data from [https://storage.googleapis.com/keras-applications/efficientnetb1\\_notop.h5](https://storage.googleapis.com/keras-applications/efficientnetb1_notop.h5)  
27025408/27018416 [=====] - 0s 0us/step  
27033600/27018416 [=====] - 0s 0us/step

✓

```
inputs = keras.Input(shape = (150,150,3))
x = tf.keras.applications.efficientnet.preprocess_input(inputs)
x = base_model(x, training=False)
x = keras.layers.GlobalAveragePooling2D()(x)
x = keras.layers.Dropout(0.2)(x)
outputs = keras.layers.Dense(1)(x)
model = keras.Model(inputs,outputs)
```

✓ 2m [17] model.compile(optimizer='adam', loss =  
 tf.keras.losses.BinaryCrossentropy(from\_logits = True), metrics =  
 keras.metrics.BinaryAccuracy())  
model.fit(training\_set, epochs =20, validation\_data= validation\_set)

Epoch 1/20  
27/27 [=====] - 48s 1s/step - loss: 0.3371 - binary\_accuracy: 0.9191 - val\_loss: 0.1592 - val\_binary\_accuracy: 0.9155  
Epoch 2/20  
27/27 [=====] - 3s 94ms/step - loss: 0.0998 - binary\_accuracy: 0.9496 - val\_loss: 0.1143 - val\_binary\_accuracy: 0.9296  
Epoch 3/20  
27/27 [=====] - 3s 97ms/step - loss: 0.0696 - binary\_accuracy: 0.9637 - val\_loss: 0.0895 - val\_binary\_accuracy: 0.9390  
Epoch 4/20  
27/27 [=====] - 3s 96ms/step - loss: 0.0527 - binary\_accuracy: 0.9766 - val\_loss: 0.0747 - val\_binary\_accuracy: 0.9484  
Epoch 5/20  
27/27 [=====] - 3s 96ms/step - loss: 0.0455 - binary\_accuracy: 0.9812 - val\_loss: 0.0664 - val\_binary\_accuracy: 0.9531  
Epoch 6/20  
27/27 [=====] - 3s 96ms/step - loss: 0.0381 - binary\_accuracy: 0.9848 - val\_loss: 0.0616 - val\_binary\_accuracy: 0.9531  
Epoch 7/20  
27/27 [=====] - 4s 118ms/step - loss: 0.0325 - binary\_accuracy: 0.9871 - val\_loss: 0.0563 - val\_binary\_accuracy: 0.9531  
Epoch 8/20  
27/27 [=====] - 3s 96ms/step - loss: 0.0286 - binary\_accuracy: 0.9883 - val\_loss: 0.0538 - val\_binary\_accuracy: 0.9531

```

Epoch 6/20
27/27 [=====] - 3s 96ms/step - loss: 0.0381 - binary_accuracy: 0.9848 - val_loss: 0.0616 - val_binary_accuracy: 0.9531
Epoch 7/20
27/27 [=====] - 4s 118ms/step - loss: 0.0325 - binary_accuracy: 0.9871 - val_loss: 0.0563 - val_binary_accuracy: 0.9531
Epoch 8/20
27/27 [=====] - 3s 96ms/step - loss: 0.0286 - binary_accuracy: 0.9883 - val_loss: 0.0538 - val_binary_accuracy: 0.9531
Epoch 9/20
27/27 [=====] - 3s 96ms/step - loss: 0.0236 - binary_accuracy: 0.9918 - val_loss: 0.0521 - val_binary_accuracy: 0.9531
Epoch 10/20
27/27 [=====] - 3s 96ms/step - loss: 0.0231 - binary_accuracy: 0.9930 - val_loss: 0.0491 - val_binary_accuracy: 0.9577
Epoch 11/20
27/27 [=====] - 3s 96ms/step - loss: 0.0183 - binary_accuracy: 0.9965 - val_loss: 0.0479 - val_binary_accuracy: 0.9577
Epoch 12/20
27/27 [=====] - 3s 95ms/step - loss: 0.0180 - binary_accuracy: 0.9953 - val_loss: 0.0464 - val_binary_accuracy: 0.9577
Epoch 13/20
27/27 [=====] - 3s 97ms/step - loss: 0.0165 - binary_accuracy: 0.9965 - val_loss: 0.0452 - val_binary_accuracy: 0.9577
Epoch 14/20
27/27 [=====] - 3s 98ms/step - loss: 0.0139 - binary_accuracy: 0.9977 - val_loss: 0.0451 - val_binary_accuracy: 0.9577
Epoch 15/20
27/27 [=====] - 3s 95ms/step - loss: 0.0131 - binary_accuracy: 0.9988 - val_loss: 0.0442 - val_binary_accuracy: 0.9577
Epoch 16/20
27/27 [=====] - 3s 95ms/step - loss: 0.0122 - binary_accuracy: 0.9988 - val_loss: 0.0436 - val_binary_accuracy: 0.9577
Epoch 17/20
27/27 [=====] - 3s 94ms/step - loss: 0.0113 - binary_accuracy: 0.9988 - val_loss: 0.0422 - val_binary_accuracy: 0.9577
Epoch 18/20
27/27 [=====] - 3s 95ms/step - loss: 0.0113 - binary_accuracy: 0.9977 - val_loss: 0.0409 - val_binary_accuracy: 0.9577
Epoch 19/20

```

```

json_config = model.to_json()
with open('model_config_EfficientNetB1.json','w') as json_file:
    json_file.write(json_config)

model.save_weights('pets_EfficientNetB1_transferlearning.h5')

```

### 3. App:

PL\_MLProject - Google Drive x train.ipynb - Colaboratory x EfficientNet B0 to B7 x PetClassifierApp x +

← → ↺ ⓘ 127.0.0.1:8000

## Welcome to the Pet Classifier App

Choose File No file chosen

Upload



dog prob0.9698523283004761, cat prob0.030147671699523926