PRESENTATION TP2 ET TP3

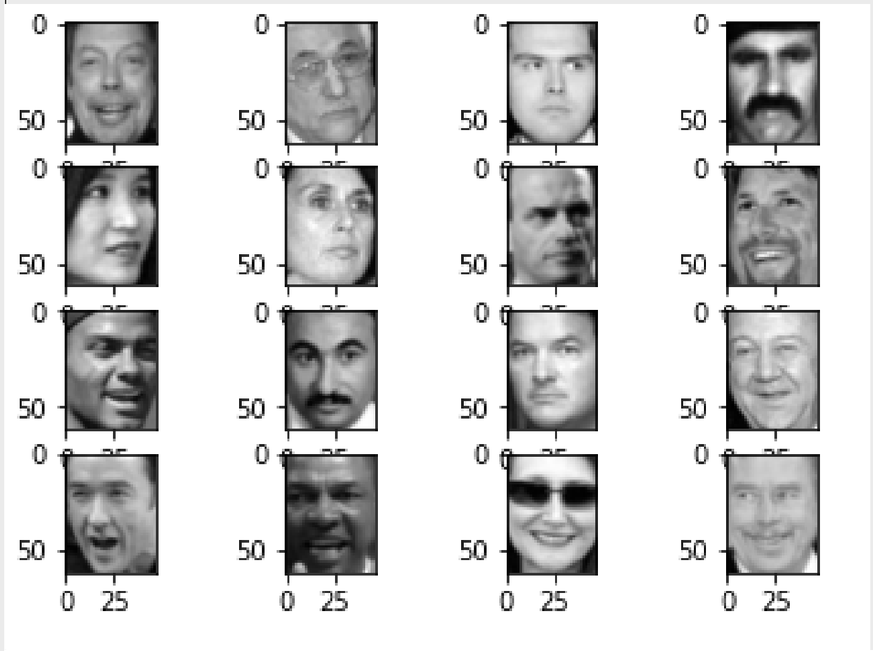
Réaliser par:
Belhassen Feryel
Ben Amor Insaf
Classe:
3DNIG2

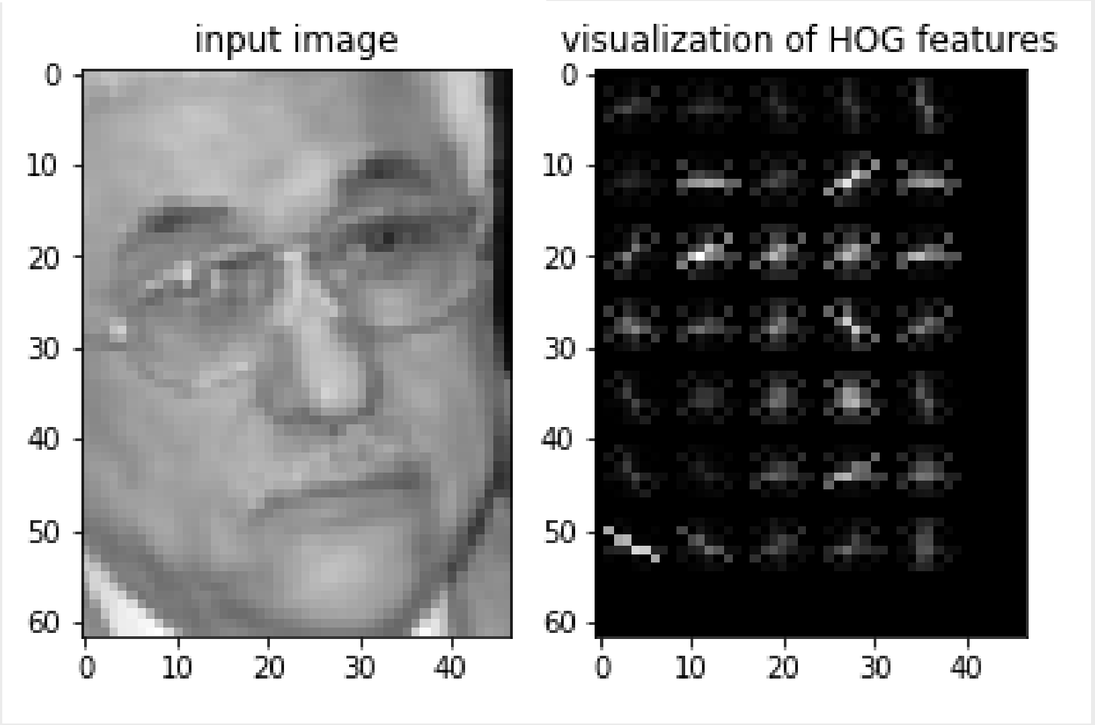
→

END-TO-END MACHINE LEARNING PROJECT
"FACE DETECTION APPLICATION"

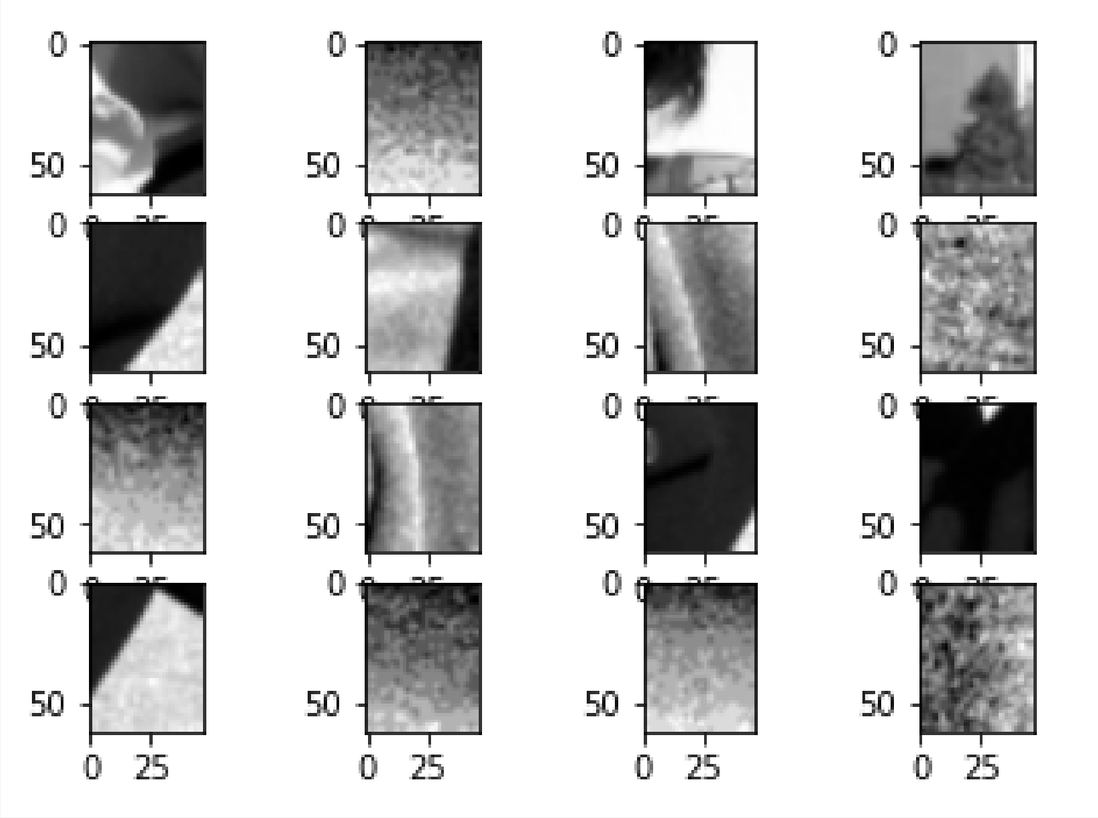
Feature Engineering

un processus qui consiste à transformer les données brutes en caractéristiques





```
from skimage import data, transform
import numpy as np
from sklearn.feature_extraction.image import PatchExtractor
imgs_to_use = ['camera', 'text', 'coins', 'moon', 'page', 'clock', 'immunohistochemistry', 'chelsea', 'coffee', 'hubble_deep_field']
images = [color.rgb2gray(getattr(data, name)()) for name in imgs_to_use]
def extract_patches(img, N, scale=1.0, patch_size=positive_patches[0].shape):
 extracted_patch_size = tuple((scale * np.array(patch_size)).astype(int))
 extractor = PatchExtractor(patch_size=extracted_patch_size,
 max_patches=N, random_state=0)
 patches = extractor.transform(img[np.newaxis])
 if scale != 1:
  patches = np.array([transform.resize(patch, patch_size)for patch in patches])
 return patches
negative_patches = np.vstack([extract_patches(im, 1000, scale)
 for im in images for scale in [0.5, 1.0, 2.0]])
```



Training model

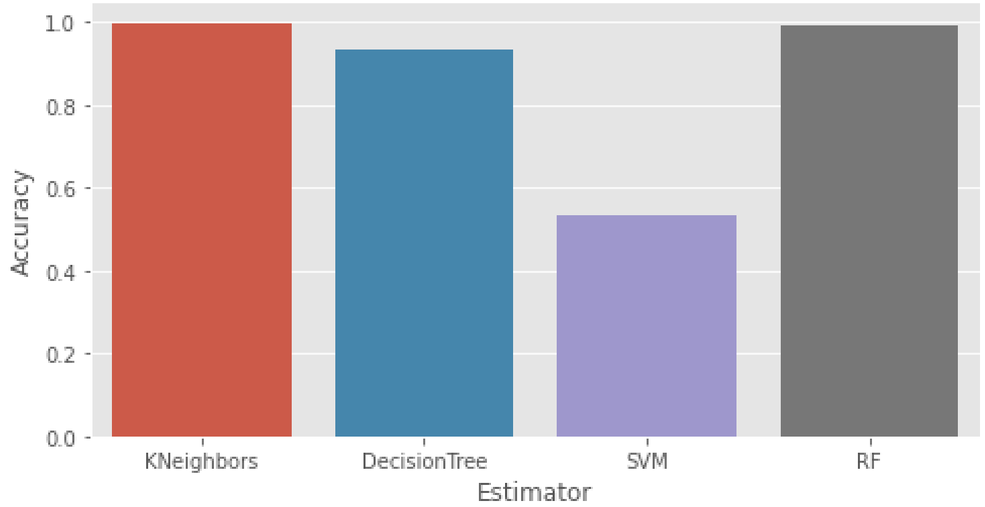
```
KNeighborsClassifier()
{'leaf_size': 25, 'n_neighbors': 5, 'p': 1, 'weights': 'uniform'}
DecisionTreeClassifier()
{'criterion': 'gini', 'max_depth': None, 'max_features': None, 'random_state': 42, 'splitter': 'best'}
SVC()
{'C': 0.1, 'gamma': 1, 'kernel': 'rbf'}
RandomForestClassifier()
{'bootstrap': True, 'criterion': 'gini', 'max_depth': 85, 'max_features': 'sqrt', 'n_estimators': 60, 'random_state': 42}
```

```
print(best_model)
```

```
GridSearchCV(cv=5, estimator=KNeighborsClassifier(),
             param_grid={'leaf_size': [25], 'n_neighbors': [5], 'p': [1],
                         'weights': ['uniform']})
```

Best model

	Estimator	Accuracy
0	KNeighbors	0.997922
1	DecisionTree	0.936154
2	SVM	0.533245
3	RF	0.989611




```
ypred=best_model.predict(X_test)
evaluate_preds(y_test,ypred)
```

```
{'accuracy': 1.0, 'precision': 1.0, 'recall': 1.0, 'f1': 1.0}
```

Classifying Iris Flowers

Toy model to play to classify iris flowers into (setosa, versicolor, virginica) based on their sepal/petal and length/width.

Plant Features

Sepal characteristics

Sepal lenght (cm)

0.50

0.50 8.00

Sepal width (cm)

0.50

0.50 4.40

Pepal characteristics

Petal lenght (cm)

0.50

0.50 7.00

Petal width (cm)

0.50

0.10 2.50

Predict type of Iris

Face detection

Try model to detect a face.

Upload Image

Drag and drop file here

Limit 200MB per file • JPG, PNG, JPEG

Browse files

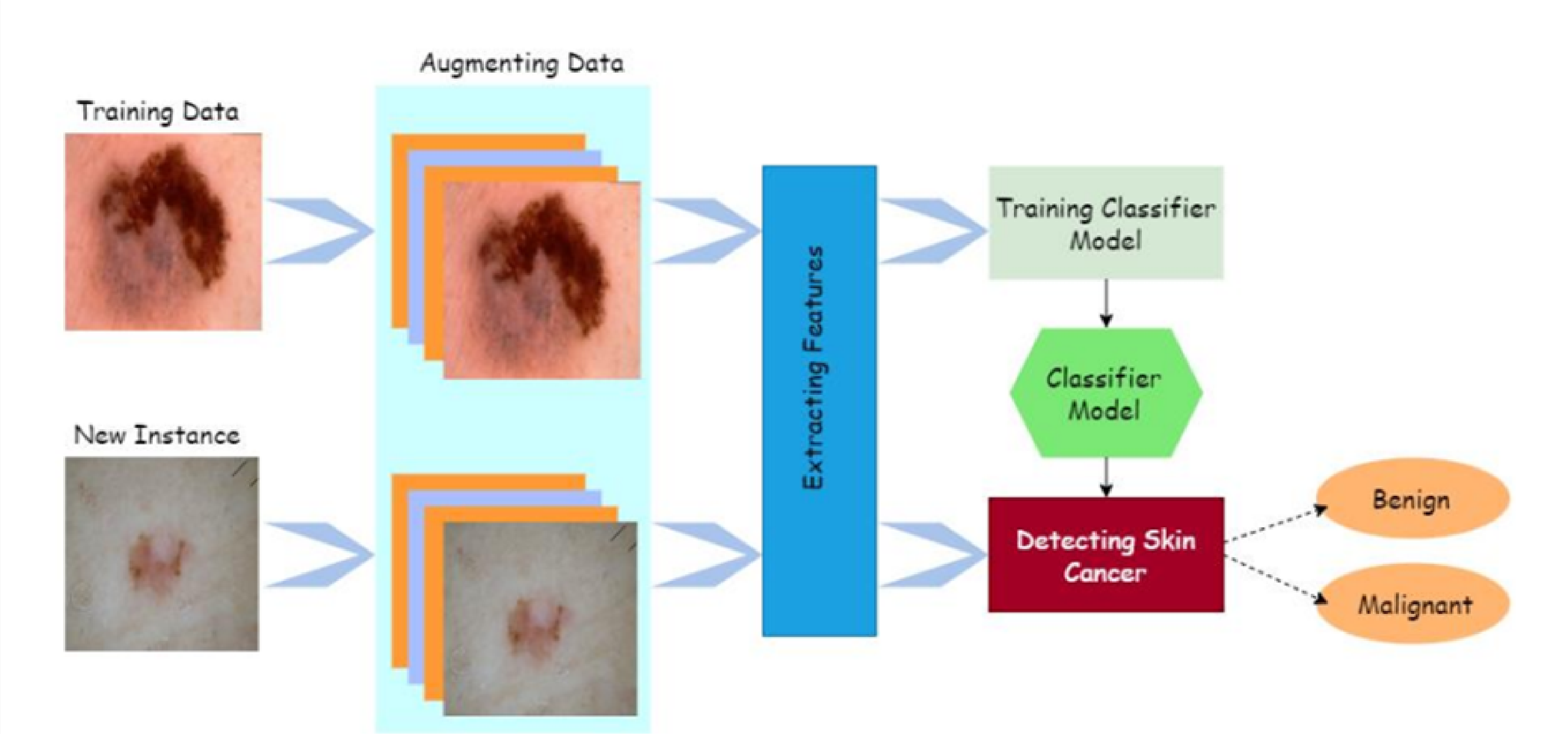
pres

suiv

Process

Skin Cancer Detection using Deep Learning

Processus de détection du cancer



Data Preprocessing:

```
train_datagen = tf.keras.preprocessing.image.ImageDataGenerator(
      rescale = 1./255,
      rotation_range=40,
      horizontal_flip=True,
      width_shift_range=0.3,
      height_shift_range=0.3,
      shear_range=0.3,
      zoom_range=0.3,
      fill_mode='nearest',
      validation_split=0.5)

train_generator = train_datagen.flow_from_directory(
    train_dir,
    subset="training",
    shuffle=True,
    seed=42,
    color_mode="rgb",
    class_mode="binary",
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE)

validation_generator = train_datagen.flow_from_directory(
    train_dir,
    shuffle=False,
    seed=42,
    color_mode="rgb",
    class_mode="binary",
    subset="validation",
    target_size=IMAGE_SIZE,
    batch_size=BATCH_SIZE)
```

```
Found 20 images belonging to 2 classes.
Found 20 images belonging to 2 classes.
```

Les transformations doivent être adaptées à l'ensemble de données d'entraînement, puis appliquées aux ensembles de données d'entraînement/validation/test.

Modeling

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer (KerasLayer)    (None, 1280)              5919312

 flatten_1 (Flatten)         (None, 1280)              0

 dense_2 (Dense)             (None, 512)               655872

 dropout_1 (Dropout)         (None, 512)               0

 dense_3 (Dense)             (None, 2)                 1026

=================================================================
Total params: 6,576,210
Trainable params: 656,898
Non-trainable params: 5,919,312
```

efficientNet Summary

Training Model

```
Epoch 1/10
1/1 [==============================] - 2s 2s/step - loss: 0.0081 - accuracy: 1.0000 - val_loss: 0.3400 - val_accuracy: 0.9375
Epoch 2/10
1/1 [==============================] - 1s 523ms/step - loss: 0.0129 - accuracy: 1.0000 - val_loss: 0.2649 - val_accuracy: 0.9375
Epoch 3/10
1/1 [==============================] - 1s 583ms/step - loss: 0.0989 - accuracy: 0.9375 - val_loss: 0.3272 - val_accuracy: 0.8125
Epoch 4/10
1/1 [==============================] - 1s 569ms/step - loss: 0.0281 - accuracy: 1.0000 - val_loss: 0.2121 - val_accuracy: 0.8750
Epoch 5/10
1/1 [==============================] - 1s 508ms/step - loss: 0.0075 - accuracy: 1.0000 - val_loss: 0.3709 - val_accuracy: 0.8750
Epoch 6/10
1/1 [==============================] - 1s 528ms/step - loss: 0.0038 - accuracy: 1.0000 - val_loss: 0.4117 - val_accuracy: 0.8750
Epoch 7/10
1/1 [==============================] - 0s 496ms/step - loss: 0.0249 - accuracy: 1.0000 - val_loss: 0.3499 - val_accuracy: 0.8750
Epoch 8/10
1/1 [==============================] - 1s 539ms/step - loss: 0.0206 - accuracy: 1.0000 - val_loss: 0.3988 - val_accuracy: 0.9375
Epoch 9/10
1/1 [==============================] - 1s 508ms/step - loss: 0.0146 - accuracy: 1.0000 - val_loss: 0.4396 - val_accuracy: 0.8750
Epoch 10/10
1/1 [==============================] - 1s 562ms/step - loss: 0.0364 - accuracy: 1.0000 - val_loss: 0.4307 - val_accuracy: 0.8750
```

- Validez chaque étape en formant le modèle avec l'ensemble de données de validation.
- Nous atteignons une précision de 81 % après 15 époques mais un réglage fin peut augmenter cette précision.

Data Augmentation

L'augmentation des données dans l'analyse des données sont des techniques utilisées pour augmenter la quantité de données en ajoutant des copies légèrement modifiées de données déjà existantes ou de données synthétiques nouvellement créées à partir de données existantes.

Evaluate model

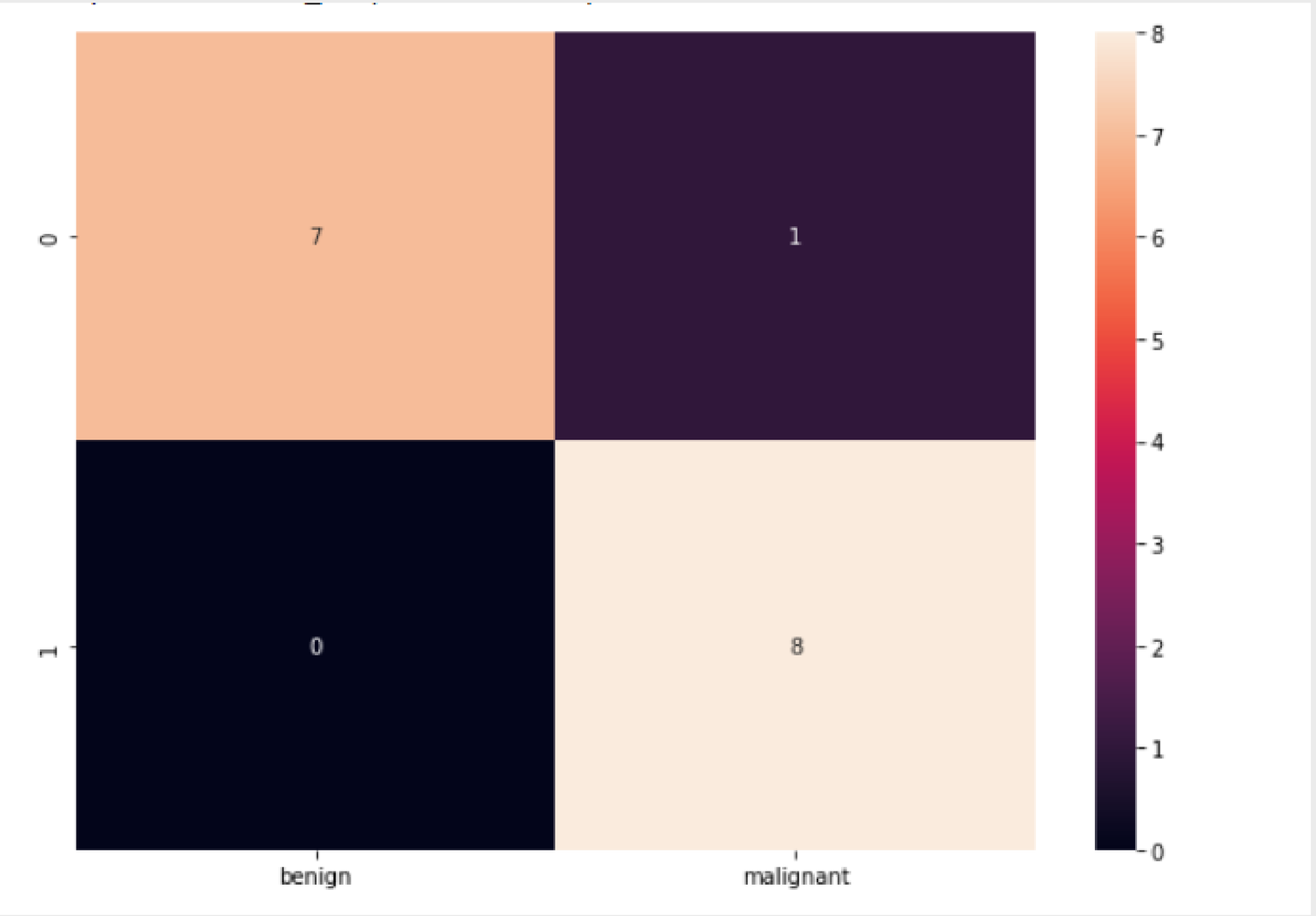
```
1/1 [==============================] - 0s 239ms/step
Classification Report
              precision    recall  f1-score   support

      benign       1.00      0.88      0.93         8
   malignant       0.89      1.00      0.94         8

    accuracy                           0.94        16
   macro avg       0.94      0.94      0.94        16
weighted avg       0.94      0.94      0.94        16

1/1 [==============================] - 0s 274ms/step - loss: 0.3638 - accuracy: 0.9375
[0.3638246953487396, 0.9375]
```

Confusion Matrix



Merci pour votre attention