```
!pip install -q kaggle
```

```
from google.colab import files

files.upload()
```

Choose Files kaggle.json

- **kaggle.json**(application/json) - 66 bytes, last modified: 8/12/2023 - 100% done

```
Saving kaggle.json to kaggle.json
{'kaggle.json': b'{"username":"vpsjoewill","key":"8292c184bedc4481a6a82d11ab591c89"}'}
```

```
!mkdir ~/.kaggle
!cp kaggle.json ~/.kaggle/
```

```
!chmod 600 ~/.kaggle/kaggle.json
```

```
!kaggle datasets list
```

```
ref                                                                 title                                              size  lastUpda
------------------------------------------------------------------  -------------------------------------------------  -----  --------
nelgiriyewithana/global-youtube-statistics-2023                     Global YouTube Statistics 2023                      60KB  2023-07-
nelgiriyewithana/countries-of-the-world-2023                        Global Country Information Dataset 2023             23KB  2023-07-
brunoalarcon123/top-200-spotify-songs-dataset                       Top 200 Spotify Songs Dataset                       35MB  2023-08-
ishanshrivastava28/tata-online-retail-dataset                       TATA: Online Retail Dataset                         29MB  2023-08-
joebeachcapital/top-10000-most-popular-movies-from-imdb             Top 10000 Most Popular Movies from TMDB              2MB  2023-07-
guillemservera/precious-metals-data                                 Gold, Silver & Precious Metals Futures Daily Data  778KB  2023-08-
ivanbyone/population-and-gdp-africa                                Population and GDP (Africa)                         23KB  2023-08-
san2deep/flipkart-product-dataset                                   Flipkart Product Dataset                           652KB  2023-08-
anshtanwar/global-data-on-sustainable-energy                        Global Data on Sustainable Energy (2000-2020)      174KB  2023-08-
arnavsmayan/netflix-userbase-dataset                                Netflix Userbase Dataset                            25KB  2023-07-
nicolasgonzalezmunoz/world-bank-world-development-indicators        World Bank World Development Indicators              2MB  2023-07-
chanoncharuchinda/korean-drama-2015-23-actor-and-reviewmydramalist  Korean Drama from 2015-2023 with Actors & Reviews    8MB  2023-08-
kaggleprollc/infant-mortality-rate-india-data-collection            Infant Mortality Rate India - Data Collection        8KB  2023-08-
joebeachcapital/fast-food                                           Fast Food Nutrition                                 20KB  2023-08-
manavgupta92/from-data-entry-to-ceo-the-ai-job-threat-index         From Data Entry to CEO: The AI Job Threat Index    102KB  2023-08-
sjagkoo7/bmi-body-mass-index                                        BMI - Body Mass Index                                2KB  2023-07-
tforsyth/99bikes-sales-data                                         99Bikes Sales Data                                   2MB  2023-08-
yeoyunsianggeremie/most-popular-python-projects-on-github-2018-2023  Most Popular Python Projects on GitHub (2018-2023)  12MB  2023-08-
joebeachcapital/global-earth-temperatures                           Global Earth Temperatures                           33KB  2023-08-
juhibhojani/house-price                                             House Price                                          7MB  2023-08-
```

```
!kaggle datasets download -d paultimothymooney/chest-xray-pneumonia
```

```
Downloading chest-xray-pneumonia.zip to /content
100% 2.28G/2.29G [00:24<00:00, 122MB/s]
100% 2.29G/2.29G [00:25<00:00, 98.3MB/s]
```

```
!mkdir Dataset
# !unzip covid19-radiography-database.zip -d ~/Datas
# !unzip -q ./{tuberculosis-tb-chest-xray-dataset}.zip -d ~/Dataset
!unzip chest-xray-pneumonia.zip -d ~/Dataset
```

```
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person992_bacteria_2920.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person992_virus_1670.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person993_bacteria_2921.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person993_virus_1671.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person994_bacteria_2922.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person994_virus_1672.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person995_bacteria_2923.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person995_virus_1676.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person996_bacteria_2924.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person996_virus_1677.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person997_bacteria_2926.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person997_virus_1678.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person998_bacteria_2927.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person998_bacteria_2928.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person99_virus_183.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person9_bacteria_38.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person9_bacteria_39.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person9_bacteria_40.jpeg
  inflating: /root/Dataset/chest_xray/train/PNEUMONIA/person9_bacteria_41.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1427-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1430-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1431-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1436-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1437-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1438-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1440-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/NORMAL/NORMAL2-IM-1442-0001.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1946_bacteria_4874.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1946_bacteria_4875.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1947_bacteria_4876.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1949_bacteria_4880.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1950_bacteria_4881.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1951_bacteria_4882.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1952_bacteria_4883.jpeg
  inflating: /root/Dataset/chest_xray/val/PNEUMONIA/person1954_bacteria_4886.jpeg
```

```
import os
for dirname, _, filenames in os.walk('/root/Dataset/chest_xray'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/root/Dataset/chest_xray/train/PNEUMONIA/person1150_bacteria_3095.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person992_bacteria_2919.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person370_virus_752.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person601_bacteria_2459.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person364_bacteria_1657.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person1366_virus_2349.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person416_bacteria_1840.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person1531_bacteria_4003.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person471_bacteria_2006.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person1233_virus_2090.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person980_bacteria_2906.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person1134_bacteria_3076.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person352_bacteria_1625.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person441_bacteria_1911.jpeg
/root/Dataset/chest_xray/train/PNEUMONIA/person836_virus_1473.jpeg
```

```
import matplotlib.pyplot as plt
import seaborn as sns
import keras
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Conv2D , MaxPool2D , Flatten , Dropout , BatchNormalization
from keras.preprocessing.image import ImageDataGenerator
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report,confusion_matrix
from keras.callbacks import ReduceLROnPlateau
import cv2
import os
import random
import shutil
```

```
labels = ['PNEUMONIA', 'NORMAL']
img_size = 150
def get_training_data(data_dir):
    data = []
    for label in labels:
        path = os.path.join(data_dir, label)
        class_num = labels.index(label)
        for img in os.listdir(path):
            try:
                img_arr = cv2.imread(os.path.join(path, img), cv2.IMREAD_GRAYSCALE)
                resized_arr = cv2.resize(img_arr, (img_size, img_size)) # Reshaping images to preferred size
                data.append([resized_arr, class_num])
            except Exception as e:
                print(e)
    return np.array(data)
```

```
train = get_training_data('/root/Dataset/chest_xray/train')
test = get_training_data('/root/Dataset/chest_xray/test')
val = get_training_data('/root/Dataset/chest_xray/val')
```

```
<ipython-input-12-b2613b36a4a4>:15: VisibleDeprecationWarning: Creating an ndarray from ragged nested sequences (which is a list-or-tupl
  return np.array(data)
```

```
x_train = []
y_train = []

x_val = []
y_val = []

x_test = []
y_test = []

for feature, label in train:
    x_train.append(feature)
    y_train.append(label)

for feature, label in test:
    x_test.append(feature)
    y_test.append(label)

for feature, label in val:
    x_val.append(feature)
    y_val.append(label)
```

```
x_train = np.array(x_train) / 255
x_val = np.array(x_val) / 255
x_test = np.array(x_test) / 255


x_train = x_train.reshape(-1, img_size, img_size, 1)
y_train = np.array(y_train)

x_val = x_val.reshape(-1, img_size, img_size, 1)
y_val = np.array(y_val)

x_test = x_test.reshape(-1, img_size, img_size, 1)
y_test = np.array(y_test)


datagen = ImageDataGenerator(
        featurewise_center=False,  # set input mean to 0 over the dataset
        samplewise_center=False,  # set each sample mean to 0
        featurewise_std_normalization=False,  # divide inputs by std of the dataset
        samplewise_std_normalization=False,  # divide each input by its std
        zca_whitening=False,  # apply ZCA whitening
        rotation_range = 30,  # randomly rotate images in the range (degrees, 0 to 180)
        zoom_range = 0.2, # Randomly zoom image
        width_shift_range=0.1,  # randomly shift images horizontally (fraction of total width)
        height_shift_range=0.1,  # randomly shift images vertically (fraction of total height)
        horizontal_flip = True,  # randomly flip images
        vertical_flip=False)  # randomly flip images


datagen.fit(x_train)


model = Sequential()
model.add(Conv2D(32 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu' , input_shape = (150,150,1)))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.1))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(64 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(128 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Conv2D(256 , (3,3) , strides = 1 , padding = 'same' , activation = 'relu'))
model.add(Dropout(0.2))
model.add(BatchNormalization())
model.add(MaxPool2D((2,2) , strides = 2 , padding = 'same'))
model.add(Flatten())
model.add(Dense(units = 128 , activation = 'relu'))
model.add(Dropout(0.2))
model.add(Dense(units = 1 , activation = 'sigmoid'))
model.compile(optimizer = "rmsprop" , loss = 'binary_crossentropy' , metrics = ['accuracy'])
model.summary()
```

```
 batch_normalization (BatchN  (None, 150, 150, 32)     128
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 75, 75, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 75, 75, 64)        18496

 dropout (Dropout)           (None, 75, 75, 64)        0

 batch_normalization_1 (Batc  (None, 75, 75, 64)       256
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 38, 38, 64)       0
 2D)
 max_pooling2d_2 (MaxPooling  (None, 19, 19, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 19, 19, 128)       73856

 dropout_1 (Dropout)         (None, 19, 19, 128)       0

 batch_normalization_3 (Batc  (None, 19, 19, 128)      512
 hNormalization)

 max_pooling2d_3 (MaxPooling  (None, 10, 10, 128)      0
 2D)

 conv2d_4 (Conv2D)           (None, 10, 10, 256)       295168

 dropout_2 (Dropout)         (None, 10, 10, 256)       0

 batch_normalization_4 (Batc  (None, 10, 10, 256)      1024
 hNormalization)

 max_pooling2d_4 (MaxPooling  (None, 5, 5, 256)        0
 2D)

 flatten (Flatten)           (None, 6400)              0

 dense (Dense)               (None, 128)               819328

 dropout_3 (Dropout)         (None, 128)               0

 dense_1 (Dense)             (None, 1)                 129

=================================================================
Total params: 1,246,401
Trainable params: 1,245,313
Non-trainable params: 1,088
_________________________________________________________________
```

```
learning_rate_reduction = ReduceLROnPlateau(monitor='val_accuracy', patience = 2, verbose=1,factor=0.3, min_lr=0.000001)
```

```
history = model.fit(datagen.flow(x_train,y_train, batch_size = 32) ,epochs = 12 , validation_data = datagen.flow(x_val, y_val) ,callbacks = [
```

```
Epoch 1/12
163/163 [==============================] - 461s 3s/step - loss: 0.5710 - accuracy: 0.8436 - val_loss: 25.8157 - val_accuracy: 0.5000 - l
Epoch 2/12
163/163 [==============================] - 383s 2s/step - loss: 0.2737 - accuracy: 0.8961 - val_loss: 34.6026 - val_accuracy: 0.5000 - l
Epoch 3/12
163/163 [==============================] - ETA: 0s - loss: 0.2331 - accuracy: 0.9135
Epoch 3: ReduceLROnPlateau reducing learning rate to 0.0003000000142492354.
163/163 [==============================] - 389s 2s/step - loss: 0.2331 - accuracy: 0.9135 - val_loss: 44.5101 - val_accuracy: 0.5000 - l
Epoch 4/12
163/163 [==============================] - 389s 2s/step - loss: 0.1540 - accuracy: 0.9444 - val_loss: 0.5789 - val_accuracy: 0.6875 - lr
Epoch 5/12
163/163 [==============================] - 387s 2s/step - loss: 0.1506 - accuracy: 0.9477 - val_loss: 1.0535 - val_accuracy: 0.5000 - lr
Epoch 6/12
163/163 [==============================] - ETA: 0s - loss: 0.1254 - accuracy: 0.9553
Epoch 6: ReduceLROnPlateau reducing learning rate to 9.000000427477062e-05.
163/163 [==============================] - 386s 2s/step - loss: 0.1254 - accuracy: 0.9553 - val_loss: 30.3708 - val_accuracy: 0.5000 - l
Epoch 7/12
163/163 [==============================] - 385s 2s/step - loss: 0.1118 - accuracy: 0.9620 - val_loss: 0.4959 - val_accuracy: 0.6250 - lr
Epoch 8/12
163/163 [==============================] - ETA: 0s - loss: 0.1174 - accuracy: 0.9615
Epoch 8: ReduceLROnPlateau reducing learning rate to 2.700000040931627e-05.
163/163 [==============================] - 384s 2s/step - loss: 0.1174 - accuracy: 0.9615 - val_loss: 1.3976 - val_accuracy: 0.6250 - lr
Epoch 9/12
163/163 [==============================] - 384s 2s/step - loss: 0.1102 - accuracy: 0.9653 - val_loss: 0.3685 - val_accuracy: 0.6875 - lr
Epoch 10/12
163/163 [==============================] - ETA: 0s - loss: 0.1034 - accuracy: 0.9657
Epoch 10: ReduceLROnPlateau reducing learning rate to 8.100000013655517e-06.
163/163 [==============================] - 385s 2s/step - loss: 0.1034 - accuracy: 0.9657 - val_loss: 0.7956 - val_accuracy: 0.6875 - lr
Epoch 11/12
163/163 [==============================] - 385s 2s/step - loss: 0.1020 - accuracy: 0.9618 - val_loss: 0.4525 - val_accuracy: 0.8125 - lr
Epoch 12/12
163/163 [==============================] - 383s 2s/step - loss: 0.0990 - accuracy: 0.9666 - val_loss: 1.1798 - val_accuracy: 0.6250 - lr
```

```
print("Loss of the model is - " , model.evaluate(x_test,y_test)[0])
print("Accuracy of the model is - " , model.evaluate(x_test,y_test)[1]*100 , "%")
```

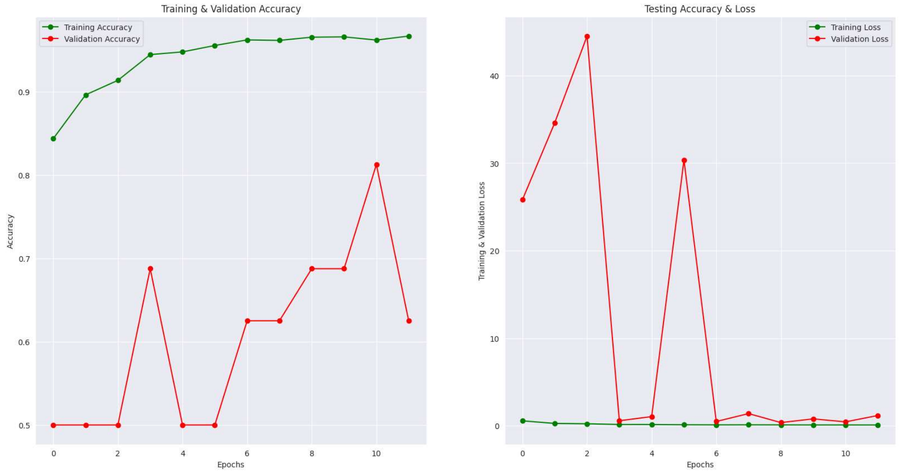
```
20/20 [==============================] - 11s 510ms/step - loss: 0.2720 - accuracy: 0.9071
Loss of the model is -  0.27203789353370667
```

```
     20/20 [==============================] - 11s 529ms/step - loss: 0.2720 - accuracy: 0.9071
     Accuracy of the model is -  90.70512652397156 %
```

```
epochs = [i for i in range(12)]
fig , ax = plt.subplots(1,2)
train_acc = history.history['accuracy']
train_loss = history.history['loss']
val_acc = history.history['val_accuracy']
val_loss = history.history['val_loss']
fig.set_size_inches(20,10)

ax[0].plot(epochs , train_acc , 'go-' , label = 'Training Accuracy')
ax[0].plot(epochs , val_acc , 'ro-' , label = 'Validation Accuracy')
ax[0].set_title('Training & Validation Accuracy')
ax[0].legend()
ax[0].set_xlabel("Epochs")
ax[0].set_ylabel("Accuracy")

ax[1].plot(epochs , train_loss , 'g-o' , label = 'Training Loss')
ax[1].plot(epochs , val_loss , 'r-o' , label = 'Validation Loss')
ax[1].set_title('Testing Accuracy & Loss')
ax[1].legend()
ax[1].set_xlabel("Epochs")
ax[1].set_ylabel("Training & Validation Loss")
plt.show()
```



```
predictions = (model.predict(x_test) > 0.5).astype("int32")
predictions = predictions.reshape(1,-1)[0]
predictions[:15]
```

```
     20/20 [==============================] - 11s 537ms/step
     array([0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0], dtype=int32)
```

```
print(classification_report(y_test, predictions, target_names = ['Pneumonia (Class 0)','Normal (Class 1)']))
```

```
                        precision    recall  f1-score   support

     Pneumonia (Class 0)       0.94      0.91      0.92       390
        Normal (Class 1)       0.86      0.90      0.88       234

                accuracy                           0.91       624
               macro avg       0.90      0.91      0.90       624
            weighted avg       0.91      0.91      0.91       624
```

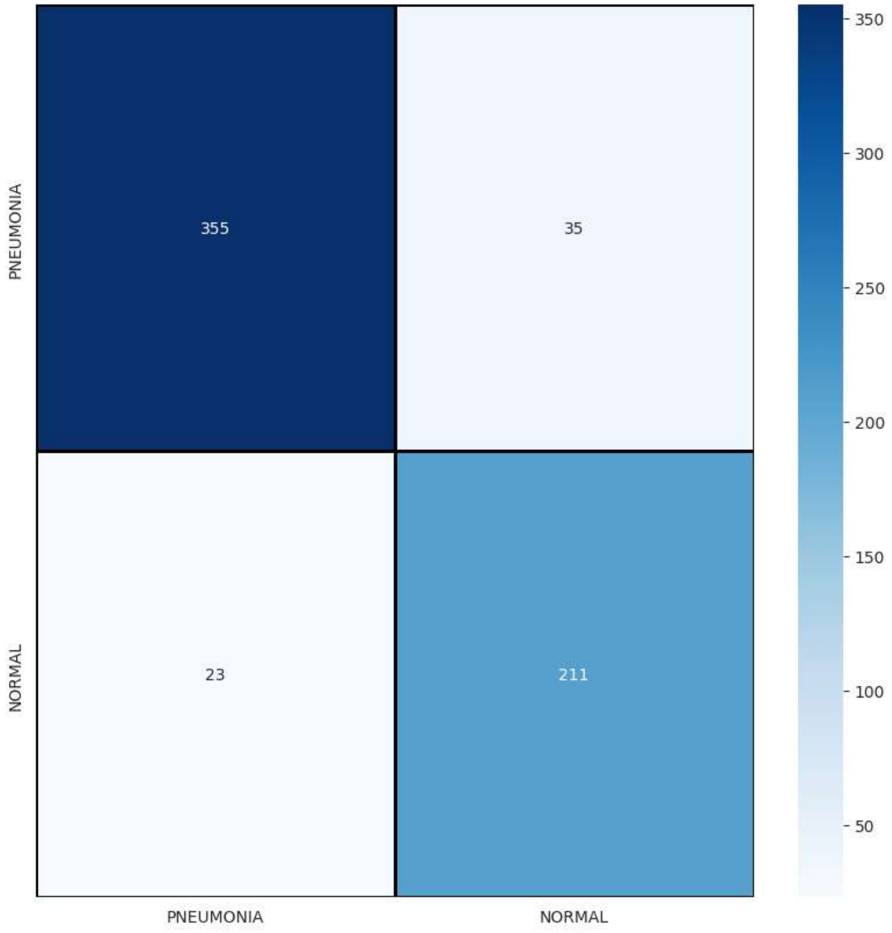
```
cm = confusion_matrix(y_test,predictions)
cm
```

```
     array([[355,  35],
            [ 23, 211]])
```

```
import pandas as pd
cm = pd.DataFrame(cm , index = ['0','1'] , columns = ['0','1'])
```

```
plt.figure(figsize = (10,10))
sns.heatmap(cm,cmap= "Blues", linecolor = 'black' , linewidth = 1 , annot = True, fmt='',xticklabels = labels,yticklabels = labels)
```

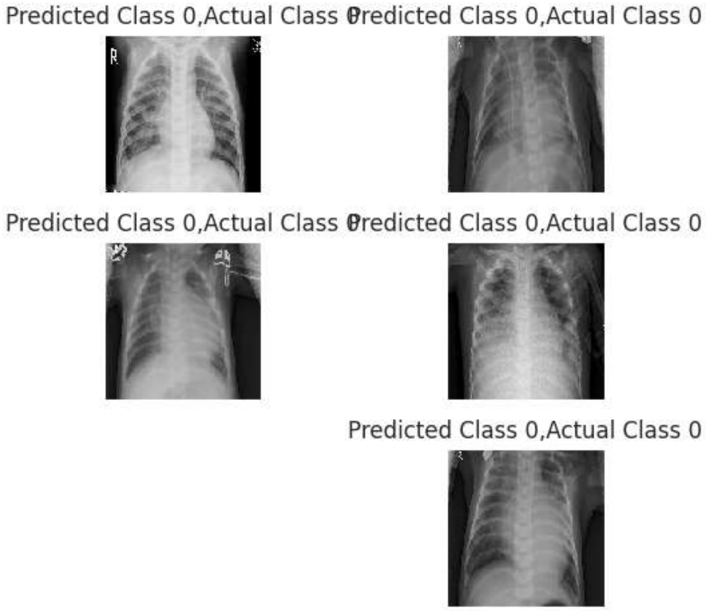
```
     <Axes: >
```



```
correct = np.nonzero(predictions == y_test)[0]
incorrect = np.nonzero(predictions != y_test)[0]
```

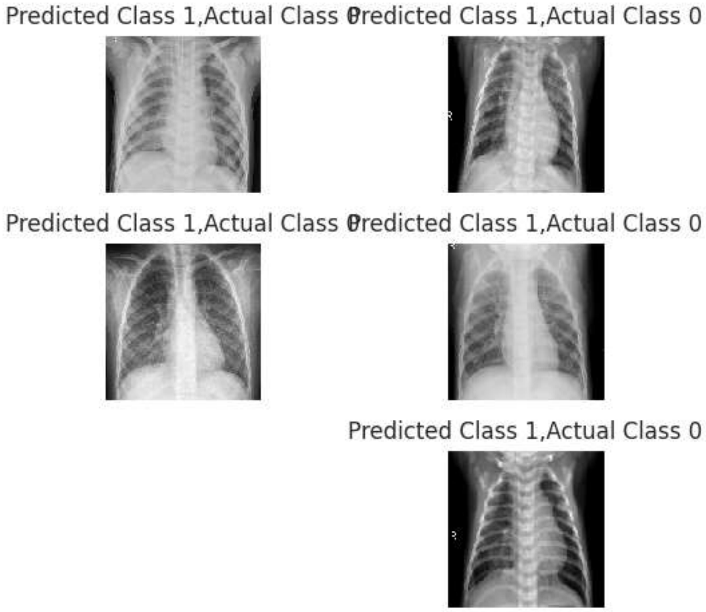
```
i = 0
for c in correct[:6]:
    plt.subplot(3,2,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(x_test[c].reshape(150,150), cmap="gray", interpolation='none')
    plt.title("Predicted Class {},Actual Class {}".format(predictions[c], y_test[c]))
    plt.tight_layout()
    i += 1
```

```
<ipython-input-57-3b0e8ec19e68>:3: MatplotlibDeprecationWarning: Auto-removal of overlap
  plt.subplot(3,2,i+1)
```


Predicted Class 0,Actual Class 0
Predicted Class 0,Actual Class 0
Predicted Class 0,Actual Class 0
Predicted Class 0,Actual Class 0
Predicted Class 0,Actual Class 0

```
i = 0
for c in incorrect[:6]:
    plt.subplot(3,2,i+1)
    plt.xticks([])
    plt.yticks([])
    plt.imshow(x_test[c].reshape(150,150), cmap="gray", interpolation='none')
    plt.title("Predicted Class {},Actual Class {}".format(predictions[c], y_test[c]))
    plt.tight_layout()
    i += 1
```

```
<ipython-input-58-d863d2b73908>:3: MatplotlibDeprecationWarning: Auto-removal of overlap
  plt.subplot(3,2,i+1)
```


Predicted Class 1,Actual Class 0
Predicted Class 1,Actual Class 0
Predicted Class 1,Actual Class 0
Predicted Class 1,Actual Class 0
Predicted Class 1,Actual Class 0

✓ 1s completed at 1:03 PM ● ×