Image Classification

- Import the necessry libraries
- Preprocessing on the datastet
  - Traing and test data generators

In [113]:
```
import os
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import tensorflow as tf

from sklearn.metrics import accuracy_score, classification_report, conf
from tensorflow import keras
from keras.applications.vgg16 import VGG16
from keras.layers import *
from keras.models import Model, Sequential
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from keras.utils import plot_model
```

In [114]:
```
tf.keras.preprocessing.image.load_img(r"C:\Users\younu\Desktop\My Py Sc
```

Out[114]:

In [115]:
```
df = pd.read_csv(r"C:\Users\younu\Desktop\My Py Scripts\Git Repos\11_Em
df.head()
```

Out[115]:

	image_names	emergency_or_not
0	0.jpg	1
1	1.jpg	1
2	2.jpg	1
3	3.jpg	1
4	4.jpg	1

In [116]:
```
df['emergency_or_not'].value_counts()
```
Out[116]:
```
emergency_or_not
0    1361
1     991
Name: count, dtype: int64
```

In [117]:
```
df.tail()
```
Out[117]:

	image_names	emergency_or_not
2347	2347.jpg	0
2348	2348.jpg	0
2349	2349.jpg	0
2350	2350.jpg	0
2351	2351.jpg	0

In [118]:
```
train_df = df.groupby('emergency_or_not', group_keys = False).apply(lam
test_df = df.drop(train_df.index)

train_df.head()
```
Out[118]:

	image_names	emergency_or_not
2126	2126.jpg	0
1648	1648.jpg	0
2021	2021.jpg	0
1911	1911.jpg	0
1790	1790.jpg	0

In [119]:
```
test_df.head()
```
Out[119]:

	image_names	emergency_or_not
9	9.jpg	1
11	11.jpg	1
19	19.jpg	1
23	23.jpg	1
28	28.jpg	1

In [120]:
```
print(train_df.shape, test_df.shape)
```
```
(1882, 2) (470, 2)
```

In [121]:
```
test_df['emergency_or_not'].value_counts()
```
Out[121]:
```
emergency_or_not
0    272
1    198
Name: count, dtype: int64
```

In [122]:
```python
test_df = df.groupby('emergency_or_not', group_keys = False).apply(lamb
test_df.head()
```

Out[122]:

	image_names	emergency_or_not
2126	2126.jpg	0
1648	1648.jpg	0
2021	2021.jpg	0
1911	1911.jpg	0
1790	1790.jpg	0

In [123]:
```python
train_df['emergency_or_not'] = train_df['emergency_or_not'].astype(str)
test_df['emergency_or_not'] = test_df['emergency_or_not'].astype(str)
```

In [124]:
```python
print(train_df.shape, test_df.shape)
```
```
(1882, 2) (470, 2)
```

In [125]:
```python
test_df['emergency_or_not'].value_counts()
```

Out[125]:
```
emergency_or_not
0    272
1    198
Name: count, dtype: int64
```

In [126]:
```python
training_data_generator = ImageDataGenerator(rotation_range = 30,
                                             width_shift_range = 0.2,
                                             height_shift_range = 0.2,
                                             shear_range = 0.2,
                                             zoom_range = 0.3,
                                             channel_shift_range = 0.2,
                                             fill_mode = 'nearest',
                                             horizontal_flip = True,
                                             vertical_flip = True,
                                             rescale = 1./255)

testing_data_generator = ImageDataGenerator(rescale = 1./255)
```

In [127]:
```python
folder_path = r"C:\Users\younu\Desktop\My Py Scripts\Git Repos\11_Emerg
```

In [128]:
```python
train_generator = training_data_generator.flow_from_dataframe(train_df,
                                                              x_col = 'i
                                                              target_siz
```
```
Found 1882 validated image filenames belonging to 2 classes.
```

In [129]:
```python
test_generator = testing_data_generator.flow_from_dataframe(test_df, di
                                                              x_col = 'i
                                                              target_siz
```
```
Found 470 validated image filenames belonging to 2 classes.
```

In [130]:
```
vgg16_model = VGG16(include_top = False, weights = 'imagenet', input_sh
vgg16_model.trainable = False # We give false to not change the actual
```

In [132]:
```
output = vgg16_model.layers[-1].output  # -1 to access the last layer (
flatten = Flatten()(output)
```

In [133]:

```python
dense1 = Dense(512, activation = 'relu')(flatten)
dense2 = Dense(512, activation = 'relu')(dense1)
output1 = Dense(1, activation = 'sigmoid', name = 'emergency_or_not')(d

model = Model(inputs = vgg16_model.input, outputs = output1)
model.summary()
```

```
Model: "functional_1"

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩
│ input_layer_1 (InputLayer)      │ (None, 200, 200, 3)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block1_conv1 (Conv2D)           │ (None, 200, 200, 64)   │         1,792 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block1_conv2 (Conv2D)           │ (None, 200, 200, 64)   │        36,928 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block1_pool (MaxPooling2D)      │ (None, 100, 100, 64)   │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block2_conv1 (Conv2D)           │ (None, 100, 100, 128)  │        73,856 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block2_conv2 (Conv2D)           │ (None, 100, 100, 128)  │       147,584 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block2_pool (MaxPooling2D)      │ (None, 50, 50, 128)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block3_conv1 (Conv2D)           │ (None, 50, 50, 256)    │       295,168 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block3_conv2 (Conv2D)           │ (None, 50, 50, 256)    │       590,080 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block3_conv3 (Conv2D)           │ (None, 50, 50, 256)    │       590,080 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block3_pool (MaxPooling2D)      │ (None, 25, 25, 256)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block4_conv1 (Conv2D)           │ (None, 25, 25, 512)    │     1,180,160 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block4_conv2 (Conv2D)           │ (None, 25, 25, 512)    │     2,359,808 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block4_conv3 (Conv2D)           │ (None, 25, 25, 512)    │     2,359,808 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block4_pool (MaxPooling2D)      │ (None, 12, 12, 512)    │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block5_conv1 (Conv2D)           │ (None, 12, 12, 512)    │     2,359,808 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block5_conv2 (Conv2D)           │ (None, 12, 12, 512)    │     2,359,808 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block5_conv3 (Conv2D)           │ (None, 12, 12, 512)    │     2,359,808 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ block5_pool (MaxPooling2D)      │ (None, 6, 6, 512)      │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ flatten_2 (Flatten)             │ (None, 18432)          │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_4 (Dense)                 │ (None, 512)            │     9,437,696 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense_5 (Dense)                 │ (None, 512)            │       262,656 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ emergency_or_not (Dense)        │ (None, 1)              │           513 │
└─────────────────────────────────┴────────────────────────┴───────────────┘

 Total params: 24,415,553 (93.14 MB)
```

```
Trainable params: 9,700,865 (37.01 MB)

Non-trainable params: 14,714,688 (56.13 MB)
```

In [134]:
```
model.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics
```

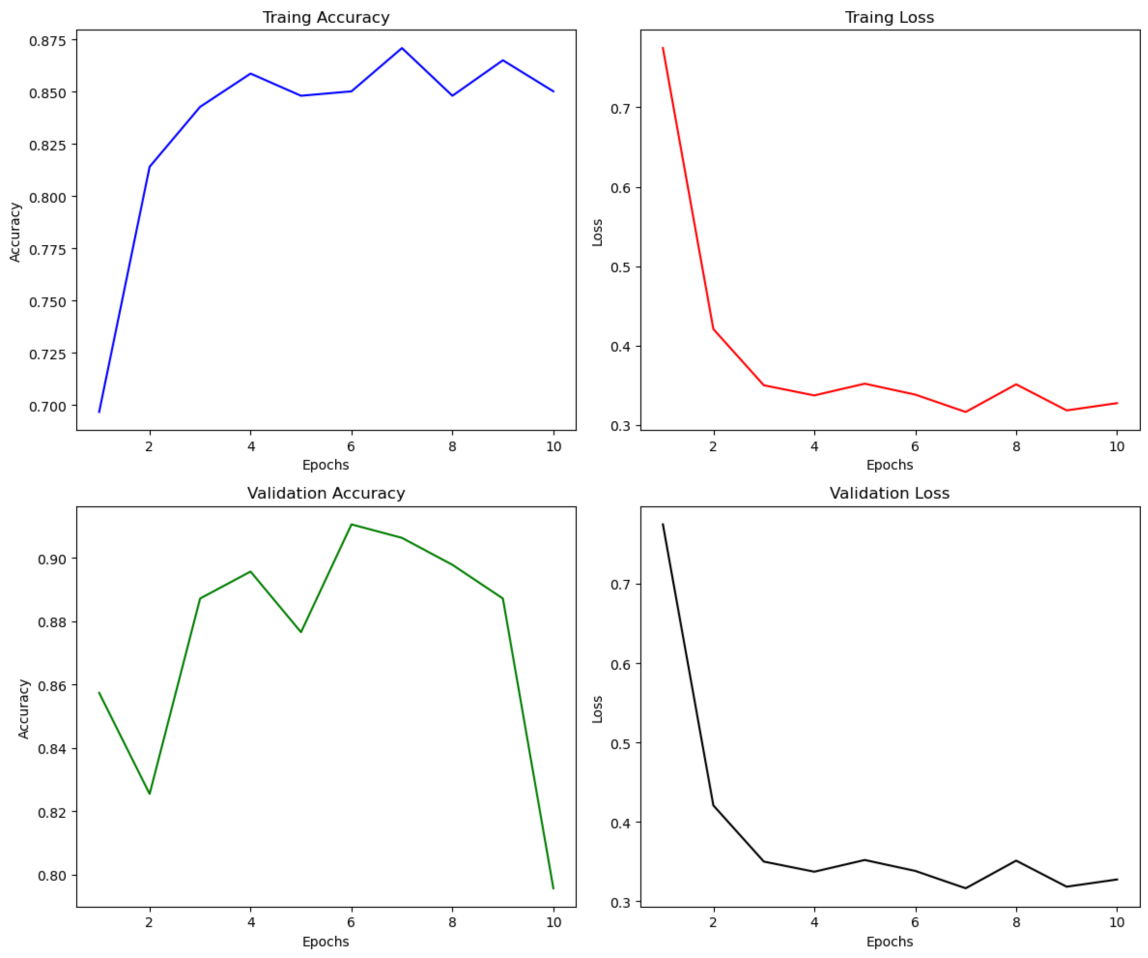
In [135]:
```
history = model.fit(train_generator, batch_size = 32, epochs = 10, vali
```

```
Epoch 1/10

C:\Users\younu\anaconda3\Lib\site-packages\keras\src\trainers\data_adapter
s\py_dataset_adapter.py:121: UserWarning: Your `PyDataset` class should ca
ll `super().__init__(**kwargs)` in its constructor. `**kwargs` can include
`workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these argu
ments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()

59/59 ━━━━━━━━━━━━━━━━━━━━ 337s 6s/step - accuracy: 0.6189 - loss: 1.2496
- val_accuracy: 0.8574 - val_loss: 0.3211
Epoch 2/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 483s 8s/step - accuracy: 0.8109 - loss: 0.4330
- val_accuracy: 0.8255 - val_loss: 0.3859
Epoch 3/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 424s 7s/step - accuracy: 0.8478 - loss: 0.3490
- val_accuracy: 0.8872 - val_loss: 0.2832
Epoch 4/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 407s 7s/step - accuracy: 0.8543 - loss: 0.3476
- val_accuracy: 0.8957 - val_loss: 0.2506
Epoch 5/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 407s 7s/step - accuracy: 0.8610 - loss: 0.3285
- val_accuracy: 0.8766 - val_loss: 0.2838
Epoch 6/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 392s 7s/step - accuracy: 0.8596 - loss: 0.3147
- val_accuracy: 0.9106 - val_loss: 0.2271
Epoch 7/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 411s 7s/step - accuracy: 0.8845 - loss: 0.2983
- val_accuracy: 0.9064 - val_loss: 0.2154
Epoch 8/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 444s 7s/step - accuracy: 0.8594 - loss: 0.3289
- val_accuracy: 0.8979 - val_loss: 0.2255
Epoch 9/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 407s 7s/step - accuracy: 0.8542 - loss: 0.3200
- val_accuracy: 0.8872 - val_loss: 0.2738
Epoch 10/10
59/59 ━━━━━━━━━━━━━━━━━━━━ 381s 6s/step - accuracy: 0.8543 - loss: 0.3266
- val_accuracy: 0.7957 - val_loss: 0.4908
```

In [136]:
```
history.history.keys()
```

Out[136]:
```
dict_keys(['accuracy', 'loss', 'val_accuracy', 'val_loss'])
```

In [137]:
```python
x = np.linspace(1, 10, 10)

fig, axs = plt.subplots(2, 2, figsize = (12, 10))

axs[0, 0].plot(x, history.history['accuracy'], 'b')
axs[0, 0].set_title('Traing Accuracy')
axs[0, 0].set_xlabel('Epochs')
axs[0, 0].set_ylabel('Accuracy')

axs[0, 1].plot(x, history.history['loss'], 'r')
axs[0, 1].set_title('Traing Loss')
axs[0, 1].set_xlabel('Epochs')
axs[0, 1].set_ylabel('Loss')

axs[1, 0].plot(x, history.history['val_accuracy'], 'g')
axs[1, 0].set_title('Validation Accuracy')
axs[1, 0].set_xlabel('Epochs')
axs[1, 0].set_ylabel('Accuracy')

axs[1, 1].plot(x, history.history['loss'], 'k')
axs[1, 1].set_title('Validation Loss')
axs[1, 1].set_xlabel('Epochs')
axs[1, 1].set_ylabel('Loss')

plt.tight_layout()
plt.show()
```



In [138]:
```python
test_generator.class_indices
```

Out[138]:
```
{'0': 0, '1': 1}
```

In [139]:
```python
# Checking the predicton probability of each class in test dataset

predictions = model.predict(test_generator, verbose = 1)
predictions
```

```
15/15 ━━━━━━━━━━━━━━━━━━━━ 74s 5s/step
```

Out[139]:
```
array([[0.99331623],
       [0.98817325],
       [0.08325341],
       [0.07799871],
       [0.9643081 ],
       [0.9911224 ],
       [0.6962938 ],
       [0.580725  ],
       [0.03578774],
       [0.9861002 ],
       [0.9969041 ],
       [0.99850017],
       [0.9985583 ],
       [0.9384076 ],
       [0.50707495],
       [0.8069996 ],
       [0.0292578 ],
       [0.99905896]
```

In [140]:
```python
# Converting the predictions into labels

predicted_classes = np.argmax(predictions, axis = 1)
predicted_classes
```

Out[140]:
```
array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0], dtype=int64)
```

In [141]:
```python
true_classes = test_generator.labels
```

In [142]:
```python
accuracy_score(true_classes, predicted_classes)
```

Out[142]:
```
0.5787234042553191
```

In [143]:
```
report = classification_report(true_classes, predicted_classes, target_
print(report)
```

```
              precision    recall  f1-score   support

           0       0.58      1.00      0.73       272
           1       0.00      0.00      0.00       198

    accuracy                           0.58       470
   macro avg       0.29      0.50      0.37       470
weighted avg       0.33      0.58      0.42       470


C:\Users\younu\anaconda3\Lib\site-packages\sklearn\metrics\_classificatio
n.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\younu\anaconda3\Lib\site-packages\sklearn\metrics\_classificatio
n.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\younu\anaconda3\Lib\site-packages\sklearn\metrics\_classificatio
n.py:1344: UndefinedMetricWarning: Precision and F-score are ill-defined a
nd being set to 0.0 in labels with no predicted samples. Use `zero_divisio
n` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

In [144]:
```
test_generator.class_indices
```

Out[144]:
```
{'0': 0, '1': 1}
```

In [145]:

```
confu_matrix = confusion_matrix(true_classes, predictions)
ns.heatmap(confu_matrix, annot=True, fmt='d', cmap='Blues', xticklabels
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[145], line 1
----> 1 confu_matrix = confusion_matrix(true_classes, predictions)
      2 ns.heatmap(confu_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=test_generator.class_indices.keys(), yticklabels=test_generator.class_indices.keys())

File ~\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:317, in confusion_matrix(y_true, y_pred, labels, sample_weight, normalize)
    232 def confusion_matrix(
    233     y_true, y_pred, *, labels=None, sample_weight=None, normalize=None
    234 ):
    235     """Compute confusion matrix to evaluate the accuracy of a classification.
    236 
    237     By definition a confusion matrix :math:`C` is such that :math:`C_{i, j}`
   (...)
    315     (0, 2, 1, 1)
    316     """
--> 317     y_type, y_true, y_pred = _check_targets(y_true, y_pred)
    318     if y_type not in ("binary", "multiclass"):
    319         raise ValueError("%s is not supported" % y_type)

File ~\anaconda3\Lib\site-packages\sklearn\metrics\_classification.py:95, in _check_targets(y_true, y_pred)
     92     y_type = {"multiclass"}
     94 if len(y_type) > 1:
---> 95     raise ValueError(
     96         "Classification metrics can't handle a mix of {0} and {1} targets".format(
     97             type_true, type_pred
     98         )
     99     )
    101 # We can't have more than one value on y_type => The set is no more needed
    102 y_type = y_type.pop()

ValueError: Classification metrics can't handle a mix of binary and continuous targets
```

In [ ]:

```

```