Import packages

In [135]:
```
import numpy as np
import random
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import tensorflow
import os
import glob
import time
from PIL import Image
from tqdm import tqdm
from sklearn.utils import shuffle

from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import InputLayer, Conv2D, BatchNormalizati
on, MaxPool2D, Dropout, Flatten, Dense
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint, R
educeLROnPlateau

from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import confusion_matrix, classification_report, acc
uracy_score
from sklearn.decomposition import PCA #Principal Component Analysis
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegressionCV, LogisticRegressio
n



from tensorflow.keras.applications.vgg16 import VGG16
import tensorflow.keras.applications.vgg16 as vgg16
from tensorflow.keras.applications.densenet import DenseNet121
import tensorflow.keras.applications.densenet as densenet
from tensorflow.keras.applications.xception import Xception
import tensorflow.keras.applications.xception as xception
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.optimizers import SGD, Adam
import matplotlib.pyplot as plt
import time
```

Load data

In [4]:
```
# reference: https://www.kaggle.com/gauravrajpal/leukemia-classification
-v1-2-xception-66-52

# load training data
train_0_all = glob.glob('cs156 final/training_data/fold_0/all/*.bmp')
train_0_hem = glob.glob('cs156 final/training_data/fold_0/hem/*.bmp')
train_1_all = glob.glob('cs156 final/training_data/fold_1/all/*.bmp')
train_1_hem = glob.glob('cs156 final/training_data/fold_1/hem/*.bmp')
train_2_all = glob.glob('cs156 final/training_data/fold_2/all/*.bmp')
train_2_hem = glob.glob('cs156 final/training_data/fold_2/hem/*.bmp')
```

In [5]:
```
# load labels for test data
valid_labels = pd.read_csv('cs156 final/validation_data/C-NMC_test_preli
m_phase_data_labels.csv')
valid_labels.head(10)
```

Out[5]:

	Patient_ID	new_names	labels
0	UID_57_29_1_all.bmp	1.bmp	1
1	UID_57_22_2_all.bmp	2.bmp	1
2	UID_57_31_3_all.bmp	3.bmp	1
3	UID_H49_35_1_hem.bmp	4.bmp	0
4	UID_58_6_13_all.bmp	5.bmp	1
5	UID_57_8_11_all.bmp	6.bmp	1
6	UID_H49_29_2_hem.bmp	7.bmp	0
7	UID_H30_6_2_hem.bmp	8.bmp	0
8	UID_58_2_1_all.bmp	9.bmp	1
9	UID_54_35_3_all.bmp	10.bmp	1

In [7]:
```
# stack all training data
train_all = np.hstack((np.array(train_0_all), np.array(train_1_all), np.
array(train_2_all)))
train_hem = np.hstack((np.array(train_0_hem), np.array(train_1_hem), np.
array(train_2_hem)))
```

In [8]:
```
# read and convert images to numpy array for healthy
train_hem_np = []
for i in tqdm(range(len(train_hem))):
    train_hem_np.append(np.array(Image.open(train_hem[i]).resize((100,10
0))))
```

```
100%|██████████| 3389/3389 [00:08<00:00, 419.36it/s]
```

In [9]:
```
# read and convert images to numpy array for leukemia
train_all_np = []
for i in tqdm(range(len(train_all))):
    train_all_np.append(np.array(Image.open(train_all[i]).resize((100,10
0))))
```

```
100%|██████████| 7272/7272 [00:18<00:00, 398.90it/s]
```

In [10]:
```
# stack healthy and leukemia for training
train_np = np.vstack((np.array(train_all_np), np.array(train_hem_np)))
train_label = np.vstack((np.ones((len(train_all_np),1)), np.zeros((len(t
rain_hem_np),1))))
train_np, train_label = shuffle(train_np, train_label, random_state=123)
```

In [77]:
```
# check shape of training set
train_np.shape
```

Out[77]:
```
(10661, 100, 100, 3)
```

In [87]:
```
# proportion of class 1 in training
train_label.sum() / len(train_label)
```

Out[87]:
```
0.68211237219773
```

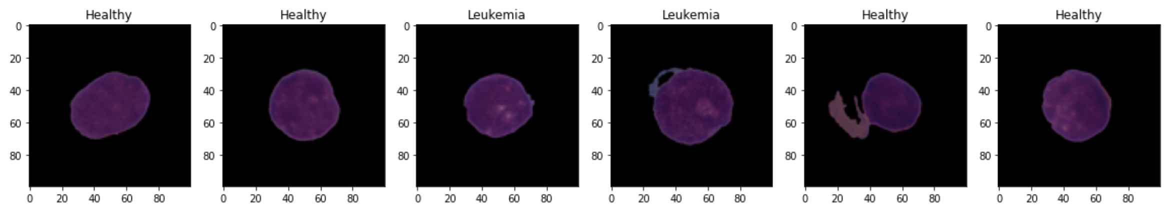
In [152]:
```
# number of class 1 and class 0 in training
print('leukemia', 10661*0.68211237219773)
print('healthy', 10661 - 10661*0.68211237219773)
```

```
leukemia 7272.0
healthy 3389.0
```

In [78]:
```
# sample randomly 3000 samples from each class in training to train
ind_bal = shuffle(np.hstack((np.random.choice(np.where(train_label == 1)
[0], 3000), np.random.choice(np.where(train_label == 0)[0], 3000))))
X_train = train_np[ind_bal]
y_train = train_label[ind_bal]
```

In [98]:
```
# check out the images for healthy and leukemia
fig, ax = plt.subplots(nrows = 1, ncols = 6, figsize = (20,20))
for i in tqdm(range(6)):
    ind = np.random.randint(len(X_train))
    ax[i].imshow(X_train[ind])
    if y_train[ind] == 1:
        ax[i].set_title('Leukemia')
    else:
        ax[i].set_title('Healthy')
```

```
100%|██████████| 6/6 [00:00<00:00, 426.85it/s]
```



In [43]:
```
# load the test set
test_np = []
test_label = []
for i in tqdm(range(valid_labels.shape[0])):
    tail = f'{i+1}.bmp'
    path = 'cs156 final/validation_data/C-NMC_test_prelim_phase_data/' + 
tail
    test_np.append(np.array(Image.open(path).resize((100,100))))
    test_label.append(valid_labels[valid_labels['new_names'] == tail]['l
abels'].iloc[0])
test_np = np.array(test_np)
test_label = np.array(test_label).reshape(-1,1)
```

```
100%|██████████| 1867/1867 [00:07<00:00, 256.28it/s]
```

In [79]:
```
# randomly sample 500 healthy and leukemia samples for test
ind_bal_test = shuffle(np.hstack((np.random.choice(np.where(test_label =
= 1)[0], 500), np.random.choice(np.where(test_label == 0)[0], 500))))
X_test = test_np[ind_bal_test]
y_test = test_label[ind_bal_test]
```

Build the models

Deep Learning

```
In [124]: # reference: https://www.kaggle.com/gauravrajpal/leukemia-classification
-v1-2-xception-66-52

# build the deep learning model with transfer learning
def individual_model(app, model_name, path_to_model, X_train, y_train, X
_test, y_test, min_delta = 0.00003, k = 0, epochs=20, batch_size=20, thr
eshold=0.5, drop_rate=0.3, class_weight=None):
    # augment the images
    X_train, X_val, y_train, y_val = train_test_split(X_train, y_train,
random_state=123, test_size=0.2)

    datagen  = ImageDataGenerator(horizontal_flip=True,
                                  vertical_flip=True,
                                  preprocessing_function=app.preproces
s_input)
    datagen.fit(X_train)

    # pre-process test data
    X_val = app.preprocess_input(X_val)
    X_test = app.preprocess_input(X_test)

    # get the base model
    base_model = model_name(include_top=False, input_shape=(100, 100, 3
), weights='imagenet')

    # get the output of the base model
    mod = base_model.output

    # add new layers on top
    mod = Flatten()(mod)
    mod = Dense(1024, activation='relu')(mod)
    mod = Dropout(drop_rate)(mod)
    mod = Dense(512, activation='relu')(mod)
    mod = Dropout(drop_rate)(mod)
    mod = Dense(256, activation='relu')(mod)
    mod = Dropout(drop_rate)(mod)

    # add output layers
    output = Dense(1, activation='sigmoid')(mod)

    # start time
    start = time.time()

    # make new model
    model = Model(inputs=base_model.inputs, outputs=output)

    # unfreeze k last layers of base model
    for layer in model.layers[:len(base_model.layers)-k]:
        layer.trainable = False
    for layer in model.layers[len(base_model.layers)-k:]:
        layer.trainable = True

    # compile the model
    model.compile(optimizer=SGD(lr=0.0001, momentum=0.9), loss='binary_c
rossentropy', metrics = ['accuracy'])

#     model.compile(optimizer=Adam(lr=0.0001), loss='binary_crossentrop
y', metrics = ['accuracy'])

    # path to log checkpoints
    filepath = f'./best_weights/{path_to_model}'
    # early stopping if no improvement after 10 epochs
    earlystopping = EarlyStopping(monitor = 'val_accuracy',
                                  mode = 'max' ,
                                  patience = 10,
                                  verbose = 1)

    # checkpoint for best performance
    checkpoint = ModelCheckpoint(filepath,
                                  monitor = 'val_accuracy',
                                  mode='max',
                                  save_best_only=True,
                                  verbose = 1)

    # reduce learning rate if no improvement
    learning_rate = ReduceLROnPlateau(monitor = 'val_accuracy',
                                      mode = 'max',
                                      patience = 5,
                                      factor = 0.3,
                                      min_delta = min_delta)

    # callbacks
    callbacks = [earlystopping, checkpoint, learning_rate]

    # train the model with validation and callbacks
    history = model.fit(datagen.flow(X_train, y_train), epochs=epochs,\
                        validation_data = (X_val, y_val),
                        batch_size = batch_size, class_weight = class_we
ight, callbacks=callbacks)

#     history = model.fit(X_train, y_train, epochs=epochs,\
#                         validation_data = (X_test, y_test),
#                         batch_size = batch_size, class_weight = class_
weight, callbacks=callbacks)

    print('training time', time.time() - start)

    # test the model
    print('loss and accuracy', model.evaluate(X_test, y_test))

    # get predicted labels based on threshold
    y_pred_prob_train = model.predict(X_train, batch_size=20, verbose=1)
    y_pred_prob = model.predict(X_test, batch_size=20, verbose=1)
    y_pred = [[1] if y_pred_prob[i][0] > threshold else [0] for i in ran
ge(len(y_test))]

    # detailed report
    print('detailed report')
    print(classification_report(y_test, y_pred))
    print('confusion matrix')
    print(confusion_matrix(y_test, y_pred))

    # plot accuracy of train and validation at each epoch
    plt.plot(history.history['accuracy'])
    plt.plot(history.history['val_accuracy'])
    plt.title('model accuracy')
    plt.ylabel('accuracy')
    plt.xlabel('epoch')
    plt.legend(['train', 'validation'], loc='upper left')
    plt.show()

    # return probability for each test sample
    return y_pred_prob_train, y_train, y_pred_prob
```

In [140]:
```
# vgg16, 0 trainable layers in base model
y_pred_prob_train_vgg1, y_train_vgg1, y_pred_prob_vgg1 = individual_mode
l(vgg16, VGG16, 'vgg16', X_train, y_train, X_test, y_test, min_delta =
0.00003, k = 0, epochs=20, batch_size=20, threshold=0.5, drop_rate=0.3,
class_weight=None)
```

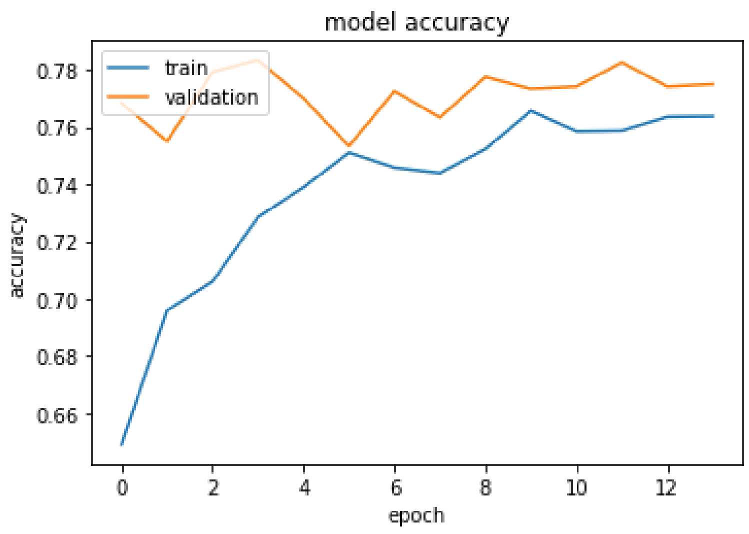
```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.9696 - acc
uracy: 0.6492
Epoch 00001: val_accuracy improved from -inf to 0.76833, saving model t
o ./best_weights/vgg16
INFO:tensorflow:Assets written to: ./best_weights/vgg16/assets
150/150 [==============================] - 31s 206ms/step - loss: 0.969
6 - accuracy: 0.6492 - val_loss: 0.4809 - val_accuracy: 0.7683
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.6531 - acc
uracy: 0.6960
Epoch 00002: val_accuracy did not improve from 0.76833
150/150 [==============================] - 27s 182ms/step - loss: 0.653
1 - accuracy: 0.6960 - val_loss: 0.4962 - val_accuracy: 0.7550
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.6094 - acc
uracy: 0.7060
Epoch 00003: val_accuracy improved from 0.76833 to 0.77917, saving mode
l to ./best_weights/vgg16
INFO:tensorflow:Assets written to: ./best_weights/vgg16/assets
150/150 [==============================] - 30s 200ms/step - loss: 0.609
4 - accuracy: 0.7060 - val_loss: 0.4817 - val_accuracy: 0.7792
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.5566 - acc
uracy: 0.7285
Epoch 00004: val_accuracy improved from 0.77917 to 0.78333, saving mode
l to ./best_weights/vgg16
INFO:tensorflow:Assets written to: ./best_weights/vgg16/assets
150/150 [==============================] - 31s 208ms/step - loss: 0.556
6 - accuracy: 0.7285 - val_loss: 0.4753 - val_accuracy: 0.7833
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.5389 - acc
uracy: 0.7390
Epoch 00005: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 188ms/step - loss: 0.538
9 - accuracy: 0.7390 - val_loss: 0.4707 - val_accuracy: 0.7700
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.5240 - acc
uracy: 0.7510
Epoch 00006: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 187ms/step - loss: 0.524
0 - accuracy: 0.7510 - val_loss: 0.5109 - val_accuracy: 0.7533
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.5192 - acc
uracy: 0.7458
Epoch 00007: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 187ms/step - loss: 0.519
2 - accuracy: 0.7458 - val_loss: 0.4702 - val_accuracy: 0.7725
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.5201 - acc
uracy: 0.7440
Epoch 00008: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 186ms/step - loss: 0.520
1 - accuracy: 0.7440 - val_loss: 0.4819 - val_accuracy: 0.7633
Epoch 9/20
150/150 [==============================] - ETA: 0s - loss: 0.5117 - acc
uracy: 0.7523
Epoch 00009: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 188ms/step - loss: 0.511
7 - accuracy: 0.7523 - val_loss: 0.4605 - val_accuracy: 0.7775
Epoch 10/20
150/150 [==============================] - ETA: 0s - loss: 0.4968 - acc
uracy: 0.7656
```

```
Epoch 00010: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 188ms/step - loss: 0.496
8 - accuracy: 0.7656 - val_loss: 0.4625 - val_accuracy: 0.7733
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.5009 - acc
uracy: 0.7585
Epoch 00011: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 189ms/step - loss: 0.500
9 - accuracy: 0.7585 - val_loss: 0.4611 - val_accuracy: 0.7742
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.4950 - acc
uracy: 0.7588
Epoch 00012: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 190ms/step - loss: 0.495
0 - accuracy: 0.7588 - val_loss: 0.4606 - val_accuracy: 0.7825
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.4935 - acc
uracy: 0.7635
Epoch 00013: val_accuracy did not improve from 0.78333
150/150 [==============================] - 29s 190ms/step - loss: 0.493
5 - accuracy: 0.7635 - val_loss: 0.4617 - val_accuracy: 0.7742
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4846 - acc
uracy: 0.7638
Epoch 00014: val_accuracy did not improve from 0.78333
150/150 [==============================] - 28s 190ms/step - loss: 0.484
6 - accuracy: 0.7638 - val_loss: 0.4595 - val_accuracy: 0.7750
Epoch 00014: early stopping
training time 405.6991868019104
32/32 [==============================] - 4s 139ms/step - loss: 0.7037 -
accuracy: 0.5840
loss and accuracy [0.7037352323532104, 0.5839999914169312]
240/240 [==============================] - 23s 97ms/step
50/50 [==============================] - 5s 94ms/step
detailed report
              precision    recall  f1-score   support

           0       0.68      0.32      0.43       500
           1       0.55      0.85      0.67       500

    accuracy                           0.58      1000
   macro avg       0.62      0.58      0.55      1000
weighted avg       0.62      0.58      0.55      1000

confusion matrix
[[159 341]
 [ 75 425]]
```


model accuracy
train
validation
accuracy
epoch
0.78
0.76
0.74
0.72
0.70
0.68
0.66
0
2
4
6
8
10
12

In [141]:
```
# vgg16, 7 trainable layers in base model
y_pred_prob_train_vgg2, y_train_vgg2, y_pred_prob_vgg2 = individual_mode
l(vgg16, VGG16, 'vgg16_2', X_train, y_train, X_test, y_test, min_delta =
0.00003, k = 7, epochs=20, batch_size=20, threshold=0.5, drop_rate=0.3,
class_weight=None)
```

```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.6288 - acc
uracy: 0.6927
Epoch 00001: val_accuracy improved from -inf to 0.75833, saving model t
o ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 76s 506ms/step - loss: 0.628
8 - accuracy: 0.6927 - val_loss: 0.4849 - val_accuracy: 0.7583
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.5093 - acc
uracy: 0.7544
Epoch 00002: val_accuracy improved from 0.75833 to 0.77833, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 73s 488ms/step - loss: 0.509
3 - accuracy: 0.7544 - val_loss: 0.4780 - val_accuracy: 0.7783
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.4854 - acc
uracy: 0.7671
Epoch 00003: val_accuracy did not improve from 0.77833
150/150 [==============================] - 70s 466ms/step - loss: 0.485
4 - accuracy: 0.7671 - val_loss: 0.4642 - val_accuracy: 0.7733
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.4692 - acc
uracy: 0.7779
Epoch 00004: val_accuracy did not improve from 0.77833
150/150 [==============================] - 70s 469ms/step - loss: 0.469
2 - accuracy: 0.7779 - val_loss: 0.4828 - val_accuracy: 0.7517
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.4604 - acc
uracy: 0.7821
Epoch 00005: val_accuracy improved from 0.77833 to 0.80333, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 73s 486ms/step - loss: 0.460
4 - accuracy: 0.7821 - val_loss: 0.4528 - val_accuracy: 0.8033
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.4554 - acc
uracy: 0.7833
Epoch 00006: val_accuracy improved from 0.80333 to 0.80500, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 72s 482ms/step - loss: 0.455
4 - accuracy: 0.7833 - val_loss: 0.4404 - val_accuracy: 0.8050
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.4525 - acc
uracy: 0.7844
Epoch 00007: val_accuracy did not improve from 0.80500
150/150 [==============================] - 70s 469ms/step - loss: 0.452
5 - accuracy: 0.7844 - val_loss: 0.4730 - val_accuracy: 0.7658
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.4400 - acc
uracy: 0.7917
Epoch 00008: val_accuracy did not improve from 0.80500
150/150 [==============================] - 70s 467ms/step - loss: 0.440
0 - accuracy: 0.7917 - val_loss: 0.4436 - val_accuracy: 0.7933
Epoch 9/20
150/150 [==============================] - ETA: 0s - loss: 0.4309 - acc
uracy: 0.7981
Epoch 00009: val_accuracy did not improve from 0.80500
150/150 [==============================] - 70s 468ms/step - loss: 0.430
9 - accuracy: 0.7981 - val_loss: 0.4555 - val_accuracy: 0.8042
Epoch 10/20
```

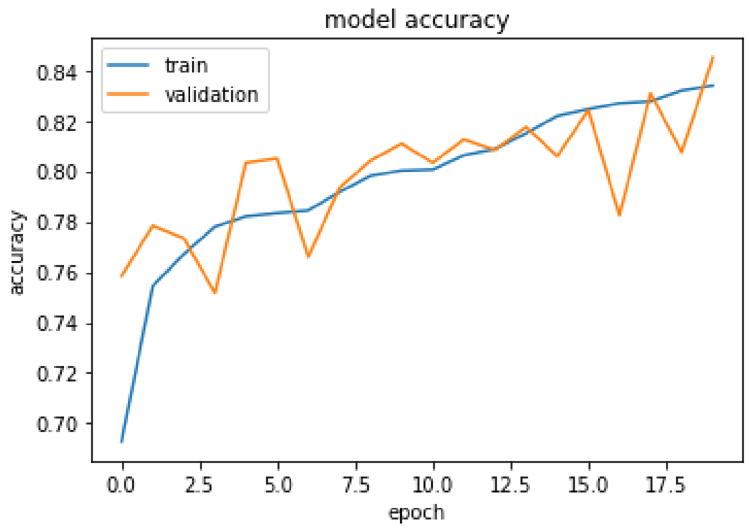
```
150/150 [==============================] - ETA: 0s - loss: 0.4231 - acc
uracy: 0.8002
Epoch 00010: val_accuracy improved from 0.80500 to 0.81083, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 73s 488ms/step - loss: 0.423
1 - accuracy: 0.8002 - val_loss: 0.4293 - val_accuracy: 0.8108
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.4259 - acc
uracy: 0.8006
Epoch 00011: val_accuracy did not improve from 0.81083
150/150 [==============================] - 71s 472ms/step - loss: 0.425
9 - accuracy: 0.8006 - val_loss: 0.4249 - val_accuracy: 0.8033
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.4150 - acc
uracy: 0.8062
Epoch 00012: val_accuracy improved from 0.81083 to 0.81250, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 73s 483ms/step - loss: 0.415
0 - accuracy: 0.8062 - val_loss: 0.4057 - val_accuracy: 0.8125
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.4106 - acc
uracy: 0.8087
Epoch 00013: val_accuracy did not improve from 0.81250
150/150 [==============================] - 70s 468ms/step - loss: 0.410
6 - accuracy: 0.8087 - val_loss: 0.4209 - val_accuracy: 0.8083
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4020 - acc
uracy: 0.8150
Epoch 00014: val_accuracy improved from 0.81250 to 0.81750, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 72s 480ms/step - loss: 0.402
0 - accuracy: 0.8150 - val_loss: 0.4165 - val_accuracy: 0.8175
Epoch 15/20
150/150 [==============================] - ETA: 0s - loss: 0.3872 - acc
uracy: 0.8219
Epoch 00015: val_accuracy did not improve from 0.81750
150/150 [==============================] - 71s 471ms/step - loss: 0.387
2 - accuracy: 0.8219 - val_loss: 0.4002 - val_accuracy: 0.8058
Epoch 16/20
150/150 [==============================] - ETA: 0s - loss: 0.3789 - acc
uracy: 0.8246
Epoch 00016: val_accuracy improved from 0.81750 to 0.82417, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 73s 488ms/step - loss: 0.378
9 - accuracy: 0.8246 - val_loss: 0.3825 - val_accuracy: 0.8242
Epoch 17/20
150/150 [==============================] - ETA: 0s - loss: 0.3774 - acc
uracy: 0.8269
Epoch 00017: val_accuracy did not improve from 0.82417
150/150 [==============================] - 74s 494ms/step - loss: 0.377
4 - accuracy: 0.8269 - val_loss: 0.4569 - val_accuracy: 0.7825
Epoch 18/20
150/150 [==============================] - ETA: 0s - loss: 0.3731 - acc
uracy: 0.8277
Epoch 00018: val_accuracy improved from 0.82417 to 0.83083, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 78s 522ms/step - loss: 0.373
1 - accuracy: 0.8277 - val_loss: 0.3858 - val_accuracy: 0.8308
```

```
Epoch 19/20
150/150 [==============================] - ETA: 0s - loss: 0.3641 - acc
uracy: 0.8319
Epoch 00019: val_accuracy did not improve from 0.83083
150/150 [==============================] - 73s 488ms/step - loss: 0.364
1 - accuracy: 0.8319 - val_loss: 0.4536 - val_accuracy: 0.8075
Epoch 20/20
150/150 [==============================] - ETA: 0s - loss: 0.3601 - acc
uracy: 0.8340
Epoch 00020: val_accuracy improved from 0.83083 to 0.84500, saving mode
l to ./best_weights/vgg16_2
INFO:tensorflow:Assets written to: ./best_weights/vgg16_2/assets
150/150 [==============================] - 76s 510ms/step - loss: 0.360
1 - accuracy: 0.8340 - val_loss: 0.3892 - val_accuracy: 0.8450
training time 1460.3803207874298
32/32 [==============================] - 4s 135ms/step - loss: 0.6762 -
accuracy: 0.6520
loss and accuracy [0.6762085556983948, 0.6520000100135803]
240/240 [==============================] - 23s 97ms/step
50/50 [==============================] - 5s 96ms/step
detailed report
              precision    recall  f1-score   support

           0       0.75      0.45      0.57       500
           1       0.61      0.85      0.71       500

    accuracy                           0.65      1000
   macro avg       0.68      0.65      0.64      1000
weighted avg       0.68      0.65      0.64      1000

confusion matrix
[[227 273]
 [ 75 425]]
```



In [142]:
```
# xception, 0 trainable layers in base model
y_pred_prob_train_xception1, y_train_xception1, y_pred_prob_xception1 =
individual_model(xception, Xception, 'xception', X_train, y_train, X_tes
t, y_test, min_delta = 0.00003, k = 0, epochs=20, batch_size=20, thresho
ld=0.5, drop_rate=0.3, class_weight=None)
```

```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.6578 - acc
uracy: 0.6098
Epoch 00001: val_accuracy improved from -inf to 0.72417, saving model t
o ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 43s 286ms/step - loss: 0.657
8 - accuracy: 0.6098 - val_loss: 0.5801 - val_accuracy: 0.7242
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.5830 - acc
uracy: 0.6992
Epoch 00002: val_accuracy improved from 0.72417 to 0.73417, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 40s 264ms/step - loss: 0.583
0 - accuracy: 0.6992 - val_loss: 0.5387 - val_accuracy: 0.7342
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.5577 - acc
uracy: 0.7254
Epoch 00003: val_accuracy improved from 0.73417 to 0.74333, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 41s 271ms/step - loss: 0.557
7 - accuracy: 0.7254 - val_loss: 0.5145 - val_accuracy: 0.7433
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.5462 - acc
uracy: 0.7271
Epoch 00004: val_accuracy improved from 0.74333 to 0.75000, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 41s 273ms/step - loss: 0.546
2 - accuracy: 0.7271 - val_loss: 0.5033 - val_accuracy: 0.7500
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.5236 - acc
uracy: 0.7435
Epoch 00005: val_accuracy improved from 0.75000 to 0.75250, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 41s 275ms/step - loss: 0.523
6 - accuracy: 0.7435 - val_loss: 0.4948 - val_accuracy: 0.7525
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.5195 - acc
uracy: 0.7465
Epoch 00006: val_accuracy improved from 0.75250 to 0.75667, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 40s 265ms/step - loss: 0.519
5 - accuracy: 0.7465 - val_loss: 0.4898 - val_accuracy: 0.7567
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.5121 - acc
uracy: 0.7556
Epoch 00007: val_accuracy improved from 0.75667 to 0.76083, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 36s 240ms/step - loss: 0.512
1 - accuracy: 0.7556 - val_loss: 0.4829 - val_accuracy: 0.7608
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.5036 - acc
uracy: 0.7556
Epoch 00008: val_accuracy improved from 0.76083 to 0.76250, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 39s 259ms/step - loss: 0.503
```

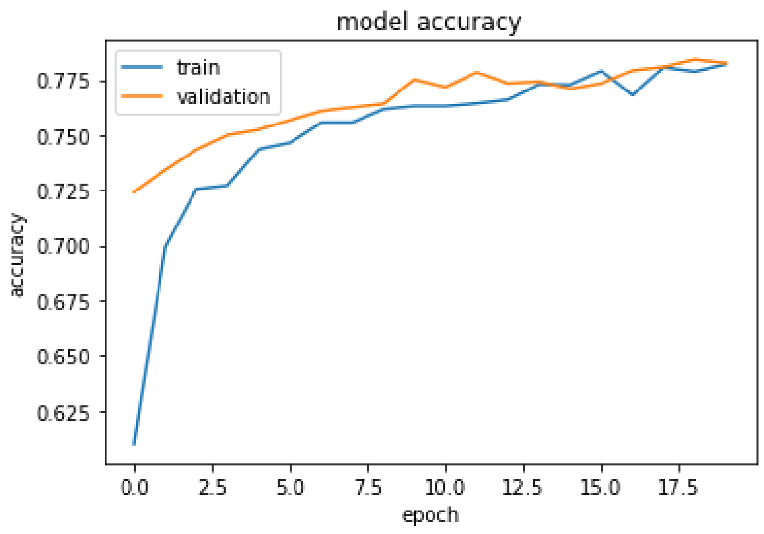
```
6 - accuracy: 0.7556 - val_loss: 0.4797 - val_accuracy: 0.7625
Epoch 9/20
150/150 [==============================] - ETA: 0s - loss: 0.4944 - acc
uracy: 0.7617
Epoch 00009: val_accuracy improved from 0.76250 to 0.76417, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 37s 247ms/step - loss: 0.494
4 - accuracy: 0.7617 - val_loss: 0.4778 - val_accuracy: 0.7642
Epoch 10/20
150/150 [==============================] - ETA: 0s - loss: 0.4918 - acc
uracy: 0.7631
Epoch 00010: val_accuracy improved from 0.76417 to 0.77500, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 37s 249ms/step - loss: 0.491
8 - accuracy: 0.7631 - val_loss: 0.4725 - val_accuracy: 0.7750
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.4929 - acc
uracy: 0.7631
Epoch 00011: val_accuracy did not improve from 0.77500
150/150 [==============================] - 21s 142ms/step - loss: 0.492
9 - accuracy: 0.7631 - val_loss: 0.4727 - val_accuracy: 0.7717
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.4909 - acc
uracy: 0.7644
Epoch 00012: val_accuracy improved from 0.77500 to 0.77833, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 37s 248ms/step - loss: 0.490
9 - accuracy: 0.7644 - val_loss: 0.4698 - val_accuracy: 0.7783
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.4820 - acc
uracy: 0.7660
Epoch 00013: val_accuracy did not improve from 0.77833
150/150 [==============================] - 21s 140ms/step - loss: 0.482
0 - accuracy: 0.7660 - val_loss: 0.4672 - val_accuracy: 0.7733
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4843 - acc
uracy: 0.7729
Epoch 00014: val_accuracy did not improve from 0.77833
150/150 [==============================] - 21s 141ms/step - loss: 0.484
3 - accuracy: 0.7729 - val_loss: 0.4655 - val_accuracy: 0.7742
Epoch 15/20
150/150 [==============================] - ETA: 0s - loss: 0.4824 - acc
uracy: 0.7727
Epoch 00015: val_accuracy did not improve from 0.77833
150/150 [==============================] - 21s 140ms/step - loss: 0.482
4 - accuracy: 0.7727 - val_loss: 0.4638 - val_accuracy: 0.7708
Epoch 16/20
150/150 [==============================] - ETA: 0s - loss: 0.4781 - acc
uracy: 0.7790
Epoch 00016: val_accuracy did not improve from 0.77833
150/150 [==============================] - 21s 140ms/step - loss: 0.478
1 - accuracy: 0.7790 - val_loss: 0.4616 - val_accuracy: 0.7733
Epoch 17/20
150/150 [==============================] - ETA: 0s - loss: 0.4821 - acc
uracy: 0.7681
Epoch 00017: val_accuracy improved from 0.77833 to 0.77917, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 35s 235ms/step - loss: 0.482
1 - accuracy: 0.7681 - val_loss: 0.4609 - val_accuracy: 0.7792
```

```
Epoch 18/20
150/150 [==============================] - ETA: 0s - loss: 0.4760 - acc
uracy: 0.7806
Epoch 00018: val_accuracy improved from 0.77917 to 0.78083, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 37s 250ms/step - loss: 0.476
0 - accuracy: 0.7806 - val_loss: 0.4631 - val_accuracy: 0.7808
Epoch 19/20
150/150 [==============================] - ETA: 0s - loss: 0.4737 - acc
uracy: 0.7788
Epoch 00019: val_accuracy improved from 0.78083 to 0.78417, saving mode
l to ./best_weights/xception
INFO:tensorflow:Assets written to: ./best_weights/xception/assets
150/150 [==============================] - 37s 244ms/step - loss: 0.473
7 - accuracy: 0.7788 - val_loss: 0.4573 - val_accuracy: 0.7842
Epoch 20/20
150/150 [==============================] - ETA: 0s - loss: 0.4674 - acc
uracy: 0.7821
Epoch 00020: val_accuracy did not improve from 0.78417
150/150 [==============================] - 21s 139ms/step - loss: 0.467
4 - accuracy: 0.7821 - val_loss: 0.4563 - val_accuracy: 0.7825
training time 673.9549684524536
32/32 [==============================] - 3s 85ms/step - loss: 0.7652 -
accuracy: 0.5860
loss and accuracy [0.7651541829109192, 0.5860000252723694]
240/240 [==============================] - 16s 66ms/step
50/50 [==============================] - 3s 64ms/step
detailed report
              precision    recall  f1-score   support

           0       0.65      0.37      0.47       500
           1       0.56      0.80      0.66       500

    accuracy                           0.59      1000
   macro avg       0.61      0.59      0.57      1000
weighted avg       0.61      0.59      0.57      1000

confusion matrix
[[186 314]
 [100 400]]
```



In [143]:
```
# xception, 7 trainable layers in base model
y_pred_prob_train_xception2, y_train_xception2, y_pred_prob_xception2 = 
individual_model(xception, Xception, 'xception_2', X_train, y_train, X_t
est, y_test, min_delta = 0.00003, k = 7, epochs=20, batch_size=20, thres
hold=0.5, drop_rate=0.3, class_weight=None)
```

```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.6861 - acc
uracy: 0.5610
Epoch 00001: val_accuracy improved from -inf to 0.58833, saving model t
o ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 45s 301ms/step - loss: 0.686
1 - accuracy: 0.5610 - val_loss: 0.6622 - val_accuracy: 0.5883
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.6517 - acc
uracy: 0.6540
Epoch 00002: val_accuracy improved from 0.58833 to 0.73500, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 43s 287ms/step - loss: 0.651
7 - accuracy: 0.6540 - val_loss: 0.6008 - val_accuracy: 0.7350
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.6225 - acc
uracy: 0.7008
Epoch 00003: val_accuracy improved from 0.73500 to 0.74500, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 46s 306ms/step - loss: 0.622
5 - accuracy: 0.7008 - val_loss: 0.5778 - val_accuracy: 0.7450
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.6016 - acc
uracy: 0.7210
Epoch 00004: val_accuracy improved from 0.74500 to 0.74917, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 44s 294ms/step - loss: 0.601
6 - accuracy: 0.7210 - val_loss: 0.5569 - val_accuracy: 0.7492
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.5724 - acc
uracy: 0.7317
Epoch 00005: val_accuracy did not improve from 0.74917
150/150 [==============================] - 28s 187ms/step - loss: 0.572
4 - accuracy: 0.7317 - val_loss: 0.5391 - val_accuracy: 0.7492
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.5541 - acc
uracy: 0.7406
Epoch 00006: val_accuracy improved from 0.74917 to 0.75167, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 45s 297ms/step - loss: 0.554
1 - accuracy: 0.7406 - val_loss: 0.5218 - val_accuracy: 0.7517
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.5382 - acc
uracy: 0.7481
Epoch 00007: val_accuracy improved from 0.75167 to 0.76167, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 44s 292ms/step - loss: 0.538
2 - accuracy: 0.7481 - val_loss: 0.5087 - val_accuracy: 0.7617
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.5325 - acc
uracy: 0.7519
Epoch 00008: val_accuracy improved from 0.76167 to 0.76250, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 42s 283ms/step - loss: 0.532
5 - accuracy: 0.7519 - val_loss: 0.5003 - val_accuracy: 0.7625
Epoch 9/20
```

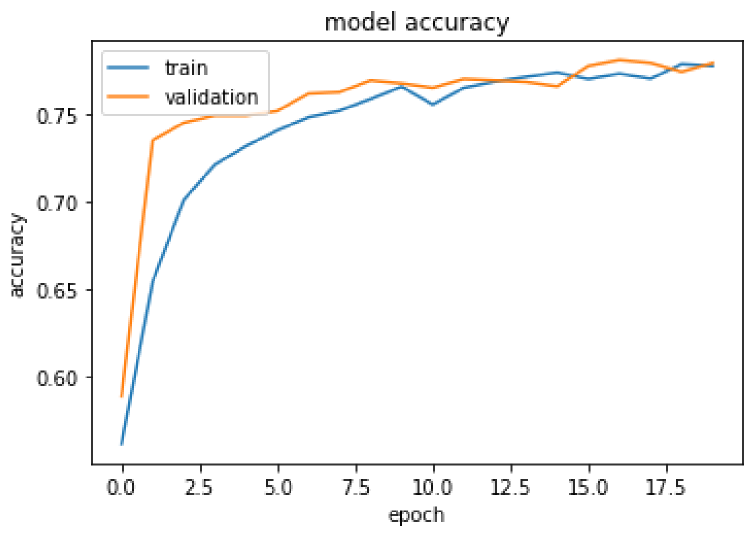
```
150/150 [==============================] - ETA: 0s - loss: 0.5174 - acc
uracy: 0.7585
Epoch 00009: val_accuracy improved from 0.76250 to 0.76917, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 44s 292ms/step - loss: 0.517
4 - accuracy: 0.7585 - val_loss: 0.4944 - val_accuracy: 0.7692
Epoch 10/20
150/150 [==============================] - ETA: 0s - loss: 0.5137 - acc
uracy: 0.7656
Epoch 00010: val_accuracy did not improve from 0.76917
150/150 [==============================] - 27s 180ms/step - loss: 0.513
7 - accuracy: 0.7656 - val_loss: 0.4882 - val_accuracy: 0.7675
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.5057 - acc
uracy: 0.7554
Epoch 00011: val_accuracy did not improve from 0.76917
150/150 [==============================] - 27s 181ms/step - loss: 0.505
7 - accuracy: 0.7554 - val_loss: 0.4835 - val_accuracy: 0.7650
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.5062 - acc
uracy: 0.7650
Epoch 00012: val_accuracy improved from 0.76917 to 0.77000, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 43s 287ms/step - loss: 0.506
2 - accuracy: 0.7650 - val_loss: 0.4786 - val_accuracy: 0.7700
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.5009 - acc
uracy: 0.7685
Epoch 00013: val_accuracy did not improve from 0.77000
150/150 [==============================] - 27s 181ms/step - loss: 0.500
9 - accuracy: 0.7685 - val_loss: 0.4760 - val_accuracy: 0.7692
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4943 - acc
uracy: 0.7713
Epoch 00014: val_accuracy did not improve from 0.77000
150/150 [==============================] - 27s 181ms/step - loss: 0.494
3 - accuracy: 0.7713 - val_loss: 0.4723 - val_accuracy: 0.7683
Epoch 15/20
150/150 [==============================] - ETA: 0s - loss: 0.4891 - acc
uracy: 0.7738
Epoch 00015: val_accuracy did not improve from 0.77000
150/150 [==============================] - 27s 180ms/step - loss: 0.489
1 - accuracy: 0.7738 - val_loss: 0.4694 - val_accuracy: 0.7658
Epoch 16/20
150/150 [==============================] - ETA: 0s - loss: 0.4938 - acc
uracy: 0.7700
Epoch 00016: val_accuracy improved from 0.77000 to 0.77750, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 44s 293ms/step - loss: 0.493
8 - accuracy: 0.7700 - val_loss: 0.4672 - val_accuracy: 0.7775
Epoch 17/20
150/150 [==============================] - ETA: 0s - loss: 0.4859 - acc
uracy: 0.7731
Epoch 00017: val_accuracy improved from 0.77750 to 0.78083, saving mode
l to ./best_weights/xception_2
INFO:tensorflow:Assets written to: ./best_weights/xception_2/assets
150/150 [==============================] - 43s 287ms/step - loss: 0.485
9 - accuracy: 0.7731 - val_loss: 0.4671 - val_accuracy: 0.7808
Epoch 18/20
150/150 [==============================] - ETA: 0s - loss: 0.4814 - acc
```

```
uracy: 0.7702
Epoch 00018: val_accuracy did not improve from 0.78083
150/150 [==============================] - 27s 179ms/step - loss: 0.481
4 - accuracy: 0.7702 - val_loss: 0.4610 - val_accuracy: 0.7792
Epoch 19/20
150/150 [==============================] - ETA: 0s - loss: 0.4750 - acc
uracy: 0.7785
Epoch 00019: val_accuracy did not improve from 0.78083
150/150 [==============================] - 28s 184ms/step - loss: 0.475
0 - accuracy: 0.7785 - val_loss: 0.4587 - val_accuracy: 0.7742
Epoch 20/20
150/150 [==============================] - ETA: 0s - loss: 0.4761 - acc
uracy: 0.7775
Epoch 00020: val_accuracy did not improve from 0.78083
150/150 [==============================] - 27s 180ms/step - loss: 0.476
1 - accuracy: 0.7775 - val_loss: 0.4577 - val_accuracy: 0.7792
training time 734.8332531452179
32/32 [==============================] - 3s 83ms/step - loss: 0.7581 -
accuracy: 0.5850
loss and accuracy [0.7581257820129395, 0.5849999785423279]
240/240 [==============================] - 15s 61ms/step
50/50 [==============================] - 3s 61ms/step
detailed report
              precision    recall  f1-score   support

           0       0.65      0.36      0.47       500
           1       0.56      0.81      0.66       500

    accuracy                           0.58      1000
   macro avg       0.61      0.58      0.56      1000
weighted avg       0.61      0.58      0.56      1000

confusion matrix
[[181 319]
 [ 96 404]]
```



In [144]:
```
# densenet, 0 trainable layers in base model
y_pred_prob_train_densenet1, y_train_densenet1, y_pred_prob_densenet1 = 
individual_model(densenet, DenseNet121, 'densenet', X_train, y_train, X_
test, y_test, min_delta = 0.00003, k = 0, epochs=20, batch_size=20, thre
shold=0.5, drop_rate=0.3, class_weight=None)
```

```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.6944 - acc
uracy: 0.6317
Epoch 00001: val_accuracy improved from -inf to 0.72833, saving model t
o ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 69s 460ms/step - loss: 0.694
4 - accuracy: 0.6317 - val_loss: 0.5319 - val_accuracy: 0.7283
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.5845 - acc
uracy: 0.6994
Epoch 00002: val_accuracy improved from 0.72833 to 0.73833, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 67s 446ms/step - loss: 0.584
5 - accuracy: 0.6994 - val_loss: 0.5151 - val_accuracy: 0.7383
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.5568 - acc
uracy: 0.7217
Epoch 00003: val_accuracy improved from 0.73833 to 0.74083, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 68s 454ms/step - loss: 0.556
8 - accuracy: 0.7217 - val_loss: 0.5046 - val_accuracy: 0.7408
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.5512 - acc
uracy: 0.7221
Epoch 00004: val_accuracy improved from 0.74083 to 0.74917, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 71s 476ms/step - loss: 0.551
2 - accuracy: 0.7221 - val_loss: 0.4984 - val_accuracy: 0.7492
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.5281 - acc
uracy: 0.7467
Epoch 00005: val_accuracy improved from 0.74917 to 0.75083, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 66s 441ms/step - loss: 0.528
1 - accuracy: 0.7467 - val_loss: 0.4961 - val_accuracy: 0.7508
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.5301 - acc
uracy: 0.7458
Epoch 00006: val_accuracy improved from 0.75083 to 0.75583, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 72s 477ms/step - loss: 0.530
1 - accuracy: 0.7458 - val_loss: 0.4875 - val_accuracy: 0.7558
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.5126 - acc
uracy: 0.7535
Epoch 00007: val_accuracy improved from 0.75583 to 0.75750, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 66s 438ms/step - loss: 0.512
6 - accuracy: 0.7535 - val_loss: 0.4857 - val_accuracy: 0.7575
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.5168 - acc
uracy: 0.7460
Epoch 00008: val_accuracy improved from 0.75750 to 0.76167, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 70s 469ms/step - loss: 0.516
```

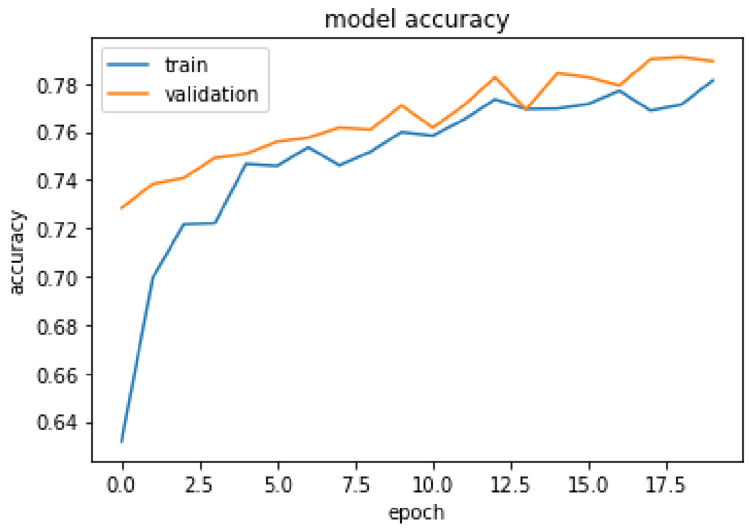
```
8 - accuracy: 0.7460 - val_loss: 0.4799 - val_accuracy: 0.7617
Epoch 9/20
150/150 [==============================] - ETA: 0s - loss: 0.5079 - acc
uracy: 0.7517
Epoch 00009: val_accuracy did not improve from 0.76167
150/150 [==============================] - 23s 154ms/step - loss: 0.507
9 - accuracy: 0.7517 - val_loss: 0.4792 - val_accuracy: 0.7608
Epoch 10/20
150/150 [==============================] - ETA: 0s - loss: 0.4993 - acc
uracy: 0.7598
Epoch 00010: val_accuracy improved from 0.76167 to 0.77083, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 66s 443ms/step - loss: 0.499
3 - accuracy: 0.7598 - val_loss: 0.4744 - val_accuracy: 0.7708
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.4940 - acc
uracy: 0.7583
Epoch 00011: val_accuracy did not improve from 0.77083
150/150 [==============================] - 23s 152ms/step - loss: 0.494
0 - accuracy: 0.7583 - val_loss: 0.4687 - val_accuracy: 0.7617
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.4913 - acc
uracy: 0.7650
Epoch 00012: val_accuracy did not improve from 0.77083
150/150 [==============================] - 23s 152ms/step - loss: 0.491
3 - accuracy: 0.7650 - val_loss: 0.4678 - val_accuracy: 0.7708
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.4826 - acc
uracy: 0.7733
Epoch 00013: val_accuracy improved from 0.77083 to 0.78250, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 71s 470ms/step - loss: 0.482
6 - accuracy: 0.7733 - val_loss: 0.4643 - val_accuracy: 0.7825
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4810 - acc
uracy: 0.7694
Epoch 00014: val_accuracy did not improve from 0.78250
150/150 [==============================] - 23s 154ms/step - loss: 0.481
0 - accuracy: 0.7694 - val_loss: 0.4625 - val_accuracy: 0.7692
Epoch 15/20
150/150 [==============================] - ETA: 0s - loss: 0.4786 - acc
uracy: 0.7696
Epoch 00015: val_accuracy improved from 0.78250 to 0.78417, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 71s 471ms/step - loss: 0.478
6 - accuracy: 0.7696 - val_loss: 0.4616 - val_accuracy: 0.7842
Epoch 16/20
150/150 [==============================] - ETA: 0s - loss: 0.4764 - acc
uracy: 0.7715
Epoch 00016: val_accuracy did not improve from 0.78417
150/150 [==============================] - 23s 154ms/step - loss: 0.476
4 - accuracy: 0.7715 - val_loss: 0.4625 - val_accuracy: 0.7825
Epoch 17/20
150/150 [==============================] - ETA: 0s - loss: 0.4706 - acc
uracy: 0.7769
Epoch 00017: val_accuracy did not improve from 0.78417
150/150 [==============================] - 23s 152ms/step - loss: 0.470
6 - accuracy: 0.7769 - val_loss: 0.4610 - val_accuracy: 0.7792
Epoch 18/20
150/150 [==============================] - ETA: 0s - loss: 0.4769 - acc
```

```
uracy: 0.7688
Epoch 00018: val_accuracy improved from 0.78417 to 0.79000, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 69s 462ms/step - loss: 0.476
9 - accuracy: 0.7688 - val_loss: 0.4542 - val_accuracy: 0.7900
Epoch 19/20
150/150 [==============================] - ETA: 0s - loss: 0.4628 - acc
uracy: 0.7713
Epoch 00019: val_accuracy improved from 0.79000 to 0.79083, saving mode
l to ./best_weights/densenet
INFO:tensorflow:Assets written to: ./best_weights/densenet/assets
150/150 [==============================] - 69s 462ms/step - loss: 0.462
8 - accuracy: 0.7713 - val_loss: 0.4560 - val_accuracy: 0.7908
Epoch 20/20
150/150 [==============================] - ETA: 0s - loss: 0.4636 - acc
uracy: 0.7812
Epoch 00020: val_accuracy did not improve from 0.79083
150/150 [==============================] - 23s 152ms/step - loss: 0.463
6 - accuracy: 0.7812 - val_loss: 0.4515 - val_accuracy: 0.7892
training time 1065.6567225456238
32/32 [==============================] - 3s 104ms/step - loss: 0.7049 -
accuracy: 0.6060
loss and accuracy [0.7048950791358948, 0.6060000061988831]
240/240 [==============================] - 21s 88ms/step
50/50 [==============================] - 4s 86ms/step
detailed report
              precision    recall  f1-score   support

           0       0.72      0.34      0.47       500
           1       0.57      0.87      0.69       500

    accuracy                           0.61      1000
   macro avg       0.65      0.61      0.58      1000
weighted avg       0.65      0.61      0.58      1000

confusion matrix
[[172 328]
 [ 66 434]]
```



In [145]:
```
# densenet, 7 trainable layers in base model
y_pred_prob_train_densenet2, y_train_densenet2, y_pred_prob_densenet2 =
individual_model(densenet, DenseNet121, 'densenet_2', X_train, y_train,
X_test, y_test, min_delta = 0.00003, k = 7, epochs=20, batch_size=20, th
reshold=0.5, drop_rate=0.3, class_weight=None)
```

```
Epoch 1/20
150/150 [==============================] - ETA: 0s - loss: 0.6708 - acc
uracy: 0.6521
Epoch 00001: val_accuracy improved from -inf to 0.73917, saving model t
o ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 74s 492ms/step - loss: 0.670
8 - accuracy: 0.6521 - val_loss: 0.5260 - val_accuracy: 0.7392
Epoch 2/20
150/150 [==============================] - ETA: 0s - loss: 0.5965 - acc
uracy: 0.7067
Epoch 00002: val_accuracy improved from 0.73917 to 0.75167, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 70s 464ms/step - loss: 0.596
5 - accuracy: 0.7067 - val_loss: 0.4930 - val_accuracy: 0.7517
Epoch 3/20
150/150 [==============================] - ETA: 0s - loss: 0.5604 - acc
uracy: 0.7231
Epoch 00003: val_accuracy improved from 0.75167 to 0.76500, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 69s 459ms/step - loss: 0.560
4 - accuracy: 0.7231 - val_loss: 0.4852 - val_accuracy: 0.7650
Epoch 4/20
150/150 [==============================] - ETA: 0s - loss: 0.5370 - acc
uracy: 0.7385
Epoch 00004: val_accuracy improved from 0.76500 to 0.76583, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 74s 490ms/step - loss: 0.537
0 - accuracy: 0.7385 - val_loss: 0.4721 - val_accuracy: 0.7658
Epoch 5/20
150/150 [==============================] - ETA: 0s - loss: 0.5234 - acc
uracy: 0.7450
Epoch 00005: val_accuracy improved from 0.76583 to 0.76750, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 74s 490ms/step - loss: 0.523
4 - accuracy: 0.7450 - val_loss: 0.4826 - val_accuracy: 0.7675
Epoch 6/20
150/150 [==============================] - ETA: 0s - loss: 0.5095 - acc
uracy: 0.7569
Epoch 00006: val_accuracy improved from 0.76750 to 0.77667, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 69s 462ms/step - loss: 0.509
5 - accuracy: 0.7569 - val_loss: 0.4632 - val_accuracy: 0.7767
Epoch 7/20
150/150 [==============================] - ETA: 0s - loss: 0.5052 - acc
uracy: 0.7490
Epoch 00007: val_accuracy improved from 0.77667 to 0.78250, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 70s 466ms/step - loss: 0.505
2 - accuracy: 0.7490 - val_loss: 0.4574 - val_accuracy: 0.7825
Epoch 8/20
150/150 [==============================] - ETA: 0s - loss: 0.4956 - acc
uracy: 0.7531
Epoch 00008: val_accuracy improved from 0.78250 to 0.79000, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 70s 464ms/step - loss: 0.495
```

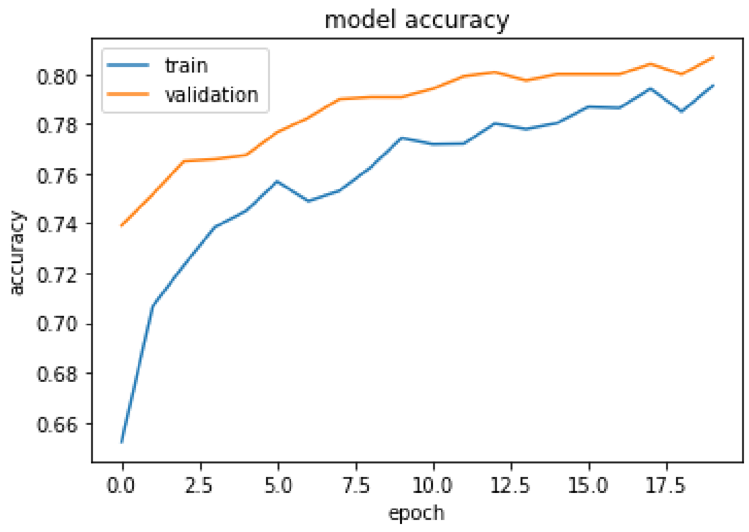
```
6 - accuracy: 0.7531 - val_loss: 0.4541 - val_accuracy: 0.7900
Epoch 9/20
150/150 [==============================] - ETA: 0s - loss: 0.4925 - acc
uracy: 0.7625
Epoch 00009: val_accuracy improved from 0.79000 to 0.79083, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 74s 490ms/step - loss: 0.492
5 - accuracy: 0.7625 - val_loss: 0.4550 - val_accuracy: 0.7908
Epoch 10/20
150/150 [==============================] - ETA: 0s - loss: 0.4768 - acc
uracy: 0.7744
Epoch 00010: val_accuracy did not improve from 0.79083
150/150 [==============================] - 26s 170ms/step - loss: 0.476
8 - accuracy: 0.7744 - val_loss: 0.4569 - val_accuracy: 0.7908
Epoch 11/20
150/150 [==============================] - ETA: 0s - loss: 0.4810 - acc
uracy: 0.7719
Epoch 00011: val_accuracy improved from 0.79083 to 0.79417, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 73s 490ms/step - loss: 0.481
0 - accuracy: 0.7719 - val_loss: 0.4482 - val_accuracy: 0.7942
Epoch 12/20
150/150 [==============================] - ETA: 0s - loss: 0.4746 - acc
uracy: 0.7721
Epoch 00012: val_accuracy improved from 0.79417 to 0.79917, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 69s 458ms/step - loss: 0.474
6 - accuracy: 0.7721 - val_loss: 0.4422 - val_accuracy: 0.7992
Epoch 13/20
150/150 [==============================] - ETA: 0s - loss: 0.4661 - acc
uracy: 0.7802
Epoch 00013: val_accuracy improved from 0.79917 to 0.80083, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 73s 485ms/step - loss: 0.466
1 - accuracy: 0.7802 - val_loss: 0.4410 - val_accuracy: 0.8008
Epoch 14/20
150/150 [==============================] - ETA: 0s - loss: 0.4588 - acc
uracy: 0.7779
Epoch 00014: val_accuracy did not improve from 0.80083
150/150 [==============================] - 26s 172ms/step - loss: 0.458
8 - accuracy: 0.7779 - val_loss: 0.4375 - val_accuracy: 0.7975
Epoch 15/20
150/150 [==============================] - ETA: 0s - loss: 0.4563 - acc
uracy: 0.7804
Epoch 00015: val_accuracy did not improve from 0.80083
150/150 [==============================] - 26s 175ms/step - loss: 0.456
3 - accuracy: 0.7804 - val_loss: 0.4355 - val_accuracy: 0.8000
Epoch 16/20
150/150 [==============================] - ETA: 0s - loss: 0.4535 - acc
uracy: 0.7869
Epoch 00016: val_accuracy did not improve from 0.80083
150/150 [==============================] - 26s 174ms/step - loss: 0.453
5 - accuracy: 0.7869 - val_loss: 0.4357 - val_accuracy: 0.8000
Epoch 17/20
150/150 [==============================] - ETA: 0s - loss: 0.4480 - acc
uracy: 0.7865
Epoch 00017: val_accuracy did not improve from 0.80083
150/150 [==============================] - 26s 171ms/step - loss: 0.448
0 - accuracy: 0.7865 - val_loss: 0.4331 - val_accuracy: 0.8000
```

```
Epoch 18/20
150/150 [==============================] - ETA: 0s - loss: 0.4477 - acc
uracy: 0.7942
Epoch 00018: val_accuracy improved from 0.80083 to 0.80417, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 70s 464ms/step - loss: 0.447
7 - accuracy: 0.7942 - val_loss: 0.4298 - val_accuracy: 0.8042
Epoch 19/20
150/150 [==============================] - ETA: 0s - loss: 0.4447 - acc
uracy: 0.7850
Epoch 00019: val_accuracy did not improve from 0.80417
150/150 [==============================] - 26s 173ms/step - loss: 0.444
7 - accuracy: 0.7850 - val_loss: 0.4294 - val_accuracy: 0.8000
Epoch 20/20
150/150 [==============================] - ETA: 0s - loss: 0.4457 - acc
uracy: 0.7954
Epoch 00020: val_accuracy improved from 0.80417 to 0.80667, saving mode
l to ./best_weights/densenet_2
INFO:tensorflow:Assets written to: ./best_weights/densenet_2/assets
150/150 [==============================] - 71s 475ms/step - loss: 0.445
7 - accuracy: 0.7954 - val_loss: 0.4256 - val_accuracy: 0.8067
training time 1162.5787949562073
32/32 [==============================] - 3s 107ms/step - loss: 0.6867 -
accuracy: 0.6000
loss and accuracy [0.6867461204528809, 0.6000000238418579]
240/240 [==============================] - 21s 89ms/step
50/50 [==============================] - 4s 90ms/step
detailed report
              precision    recall  f1-score   support

           0       0.66      0.41      0.50       500
           1       0.57      0.79      0.66       500

    accuracy                           0.60      1000
   macro avg       0.62      0.60      0.58      1000
weighted avg       0.62      0.60      0.58      1000

confusion matrix
[[203 297]
 [103 397]]
```



Traditional ML

In [170]:
```
# reference: https://colab.research.google.com/drive/1OPKdqvyUyFQ9gCOto2
dGqc5z9AdtoVIN

# transform the data with pca
def pca(X_train, X_test, n):
    pca = PCA(n_components=n)
    X_train_pca = pca.fit_transform(X_train)
    X_test_pca = pca.transform(X_test)
    return X_train_pca, X_test_pca

# transform the data with lda
def lda(X_train, X_test, y_train):
    lda = LinearDiscriminantAnalysis(solver='eigen', shrinkage=0.5)
    X_train_lda = lda.fit_transform(X_train, y_train)
    X_test_lda = lda.transform(X_test)
    return X_train_lda, X_test_lda

# logistic regression
def logistic_regression(X_train, X_test, y_train, y_test, threshold=0.5
):
    start = time.time()
    # build the model
    lr = LogisticRegressionCV(solver='liblinear')
    # fit the model
    lr.fit(X_train, y_train)
    print('running time: ', time.time()-start)
    # predict with train and test data
    y_train_pred = lr.predict(X_train)
    y_test_pred = np.where(lr.predict_proba(X_test)[:,1] > threshold, 1,
0)
    # print the results
    print('accuracy for train: ', accuracy_score(y_train, y_train_pred))
    print('accuracy for test: ', accuracy_score(y_test, y_test_pred))
    print('confusion matrix: ', confusion_matrix(y_test, y_test_pred))
    return lr.predict_proba(X_train), lr.predict_proba(X_test)

def rfc(X_train, X_test, y_train, y_test):
    # cross validation for random forest classifier
    param_grid = {
        'n_estimators': [200, 500],
        'max_features': ['auto', 'sqrt', 'log2'],
        'max_depth' : [4,5,6,7,8],
        'criterion' :['gini', 'entropy']
    }

    rfc_ = RandomForestClassifier(random_state=123)
    grc = GridSearchCV(rfc_, param_grid=param_grid, cv=7)
    grc.fit(X_train_lda, y_train)
    params = grc.best_params_

    # build random forest classifier
    rfc = RandomForestClassifier(random_state=123, max_features=params[
'max_features'],\
                                n_estimators= params['n_estimators'], m
ax_depth=params['max_depth'], criterion=params['criterion'])
    # fit the model
    rfc.fit(X_train, y_train)
    # predict on train and test
    y_train_pred = rfc.predict(X_train)
    y_test_pred = rfc.predict(X_test)

    print('accuracy for train: ', accuracy_score(y_train, y_train_pred))
    print('accuracy for test: ', accuracy_score(y_test, y_test_pred))
    return rfc.predict_proba(X_train), rfc.predict_proba(X_test)
```

In [110]:
```
# flatten data
X_train_flatten = np.array([X_train[i].flatten() for i in range(len(X_tr
ain))])
X_test_flatten = np.array([X_test[i].flatten() for i in range(len(X_test
))])
```

In [112]:
```
# transform data with pca and lda
X_train_pca, X_test_pca = pca(X_train_flatten, X_test_flatten, 50)
X_train_lda, X_test_lda = lda(X_train_flatten, X_test_flatten, y_train.r
avel())
```

In [172]:
```
# linear regression on pca-transformed data
y_pred_prob_pca_lr_train,  y_pred_prob_pca_lr_test = logistic_regression
(X_train_pca, X_test_pca, y_train.ravel(), y_test.ravel(), threshold =
0.5)
```

```
running time:  10.490171194076538
accuracy for train:  0.7741666666666667
accuracy for test:  0.598
confusion matrix:  [[198 302]
 [100 400]]
```

In [151]:
```
# linear regression on lda-transformed data
y_pred_prob_lda_lr_train,  y_pred_prob_lda_lr_test = logistic_regression
(X_train_lda, X_test_lda, y_train.ravel(), y_test.ravel(), , threshold =
0.5)
```

```
running time:  0.20975112915039062
accuracy for train:  0.897
accuracy for test:  0.59
confusion matrix:  [[237 263]
 [147 353]]
```

In [138]:
```
# random forest classifier on pca-transformed data
y_pred_prob_pca_rfc_train,  y_pred_prob_pca_rfc_test = rfc(X_train_pca,
X_test_pca, y_train.ravel(), y_test.ravel())
```

```
accuracy for train:  0.8736666666666667
accuracy for test:  0.588
```

Combined outputs

In [159]:
```
# combine the training predicted probabilities of logistic regression an
d random forest classifier
train_probs = np.hstack((y_pred_prob_pca_lr_train[:, 1].reshape(-1,1), y
_pred_prob_lda_lr_train[:, 1].reshape(-1,1), y_pred_prob_pca_rfc_train
[:, 1].reshape(-1,1)))
train_probs
```

Out[159]:
```
array([[0.01843482, 0.00233519, 0.05363039],
       [0.04834813, 0.01931042, 0.07878626],
       [0.76216741, 0.99451645, 0.61355708],
       ...,
       [0.80390993, 0.88128137, 0.77041916],
       [0.71174946, 0.76396806, 0.84654317],
       [0.83952279, 0.25583959, 0.67885789]])
```

In [160]:
```
# combine the test predicted probabilities of logistic regression and ra
ndom forest classifier
test_probs = np.hstack((y_pred_prob_pca_lr_test[:, 1].reshape(-1,1), y_p
red_prob_lda_lr_test[:, 1].reshape(-1,1), y_pred_prob_pca_rfc_test[:, 1]
.reshape(-1,1)))
test_probs
```

Out[160]:
```
array([[0.45777932, 0.08109445, 0.69083502],
       [0.81604671, 0.6485476 , 0.76819172],
       [0.87916245, 0.97777132, 0.73426752],
       ...,
       [0.49003854, 0.9545559 , 0.57672728],
       [0.57983259, 0.03566045, 0.70365834],
       [0.75967672, 0.90817719, 0.73855011]])
```

```
In [188]: # test the combined model with different classification thresholds
          for j in np.linspace(0.5, 0.9, 10):
              print('threshold: ', j)
              logistic_regression(train_probs, test_probs, y_train.ravel(), y_test
          .ravel(), threshold = j)
              print('-----------------------------')
```

```
threshold:  0.5
running time:  0.293597936630249
accuracy for train:  0.9568333333333333
accuracy for test:  0.566
confusion matrix:  [[175 325]
 [109 391]]
----------------------------
threshold:  0.5444444444444444
running time:  0.3421592712402344
accuracy for train:  0.9568333333333333
accuracy for test:  0.566
confusion matrix:  [[184 316]
 [118 382]]
----------------------------
threshold:  0.5888888888888889
running time:  0.3535337448120117
accuracy for train:  0.9568333333333333
accuracy for test:  0.564
confusion matrix:  [[191 309]
 [127 373]]
----------------------------
threshold:  0.6333333333333333
running time:  0.3530902862548828
accuracy for train:  0.9568333333333333
accuracy for test:  0.572
confusion matrix:  [[211 289]
 [139 361]]
----------------------------
threshold:  0.6777777777777778
running time:  0.352189302444458
accuracy for train:  0.9568333333333333
accuracy for test:  0.573
confusion matrix:  [[217 283]
 [144 356]]
----------------------------
threshold:  0.7222222222222222
running time:  0.353330135345459
accuracy for train:  0.9568333333333333
accuracy for test:  0.576
confusion matrix:  [[231 269]
 [155 345]]
----------------------------
threshold:  0.7666666666666666
running time:  0.35189151763916016
accuracy for train:  0.9568333333333333
accuracy for test:  0.572
confusion matrix:  [[237 263]
 [165 335]]
----------------------------
threshold:  0.8111111111111111
running time:  0.35523319244384766
accuracy for train:  0.9568333333333333
accuracy for test:  0.574
confusion matrix:  [[259 241]
 [185 315]]
----------------------------
threshold:  0.8555555555555556
running time:  0.3651437759399414
accuracy for train:  0.9568333333333333
accuracy for test:  0.577
confusion matrix:  [[277 223]
 [200 300]]
----------------------------
```

```
threshold:  0.9
running time:  0.36369895935058594
accuracy for train:  0.9568333333333333
accuracy for test:  0.576
confusion matrix:  [[301 199]
 [225 275]]
----------------------------
```

In [168]:

```
# combine the training predicted probabilities of 3 Deep Learning models
train_probs_2 = np.hstack((y_pred_prob_train_vgg2, y_pred_prob_train_xce
ption2, y_pred_prob_train_densenet2))
# combine the training predicted probabilities of 3 Deep Learning models
test_probs_2 = np.hstack((y_pred_prob_vgg2, y_pred_prob_xception2, y_pre
d_prob_densenet2))
```

In [189]:
```
# test the combined model with different classification thresholds
for j in np.linspace(0.5, 0.9, 10):
    print('threshold: ', j)
    logistic_regression(train_probs_2, test_probs_2, y_train_densenet2.r
avel(), y_test.ravel(), threshold = j)
    print('-----------------------------')
```

```
threshold:  0.5
running time:  0.21611738204956055
accuracy for train:  0.7970833333333334
accuracy for test:  0.662
confusion matrix:  [[249 251]
 [ 87 413]]
----------------------------
threshold:  0.5444444444444444
running time:  0.27676916122436523
accuracy for train:  0.7970833333333334
accuracy for test:  0.677
confusion matrix:  [[269 231]
 [ 92 408]]
----------------------------
threshold:  0.5888888888888889
running time:  0.2768261432647705
accuracy for train:  0.7970833333333334
accuracy for test:  0.69
confusion matrix:  [[286 214]
 [ 96 404]]
----------------------------
threshold:  0.6333333333333333
running time:  0.2769310474395752
accuracy for train:  0.7970833333333334
accuracy for test:  0.695
confusion matrix:  [[299 201]
 [104 396]]
----------------------------
threshold:  0.6777777777777778
running time:  0.27771425247192383
accuracy for train:  0.7970833333333334
accuracy for test:  0.704
confusion matrix:  [[321 179]
 [117 383]]
----------------------------
threshold:  0.7222222222222222
running time:  0.2767524719238281
accuracy for train:  0.7970833333333334
accuracy for test:  0.718
confusion matrix:  [[338 162]
 [120 380]]
----------------------------
threshold:  0.7666666666666666
running time:  0.2769191265106201
accuracy for train:  0.7970833333333334
accuracy for test:  0.725
confusion matrix:  [[354 146]
 [129 371]]
----------------------------
threshold:  0.8111111111111111
running time:  0.2769310474395752
accuracy for train:  0.7970833333333334
accuracy for test:  0.729
confusion matrix:  [[376 124]
 [147 353]]
----------------------------
threshold:  0.8555555555555556
running time:  0.27667689323425293
accuracy for train:  0.7970833333333334
accuracy for test:  0.722
confusion matrix:  [[394 106]
 [172 328]]
----------------------------
```

```
threshold:  0.9
running time:  0.2767066955566406
accuracy for train:  0.7970833333333334
accuracy for test:  0.712
confusion matrix:  [[429  71]
 [217 283]]
----------------------------
```