**import** pandas **as** pd

**import** cv2

**import** numpy **as** np

**import** os

**from** random **import** shuffle

**from** tqdm **import** tqdm

**import** scipy

**import** skimage

**from** skimage.transform **import** resize

**from** sklearn.model\_selection **import** train\_test\_split

print(os**.**listdir("cancer/"))

['train', 'test', 'val', '.DS\_Store']

In [4]:

print(os**.**listdir("cancer/train"))

['Cancer', 'Normal']

In [5]:

TRAIN\_DIR **=** "cancer/train/"

TEST\_DIR **=** "cancer/test/"

In [6]:

*#Preprocessing*

**def** get\_label(Dir):

**for** nextdir **in** os**.**listdir(Dir):

**if** **not** nextdir**.**startswith('.'):

**if** nextdir **in** ['NORMAL']:

label **=** 0

**elif** nextdir **in** ['CANCER']:

label **=** 1

**else**:

label **=** 2

**return** nextdir, label

In [7]:

**def** preprocessing\_data(Dir):

X **=** []

y **=** []

**for** nextdir **in** os**.**listdir(Dir):

nextdir, label **=** get\_label(Dir)

temp **=** Dir **+** nextdir

**for** image\_filename **in** tqdm(os**.**listdir(temp)):

path **=** os**.**path**.**join(temp **+** '/' , image\_filename)

img **=** cv2**.**imread(path,cv2**.**IMREAD\_GRAYSCALE)

**if** img **is** **not** **None**:

img **=** skimage**.**transform**.**resize(img, (150, 150, 3))

img **=** np**.**asarray(img)

X**.**append(img)

y**.**append(label)

X **=** np**.**asarray(X)

y **=** np**.**asarray(y)

**return** X,y

In [7]:

*#X\_train, y\_train = preprocessing\_data(TRAIN\_DIR)*

In [8]:

**def** get\_data(Dir):

X **=** []

y **=** []

**for** nextDir **in** os**.**listdir(Dir):

**if** **not** nextDir**.**startswith('.'):

**if** nextDir **in** ['NORMAL']:

label **=** 0

**elif** nextDir **in** ['CANCER']:

label **=** 1

**else**:

label **=** 2

temp **=** Dir **+** nextDir

**for** file **in** tqdm(os**.**listdir(temp)):

img **=** cv2**.**imread(temp **+** '/' **+** file)

**if** img **is** **not** **None**:

img **=** skimage**.**transform**.**resize(img, (150, 150, 3))

*#img\_file = scipy.misc.imresize(arr=img\_file, size=(150, 150, 3))*

img **=** np**.**asarray(img)

X**.**append(img)

y**.**append(label)

X **=** np**.**asarray(X)

y **=** np**.**asarray(y)

**return** X,y

In [10]:

X\_train, y\_train **=** get\_data(TRAIN\_DIR)

100%|██████████| 2478/2478 [00:27<00:00, 23.64it/s]

100%|██████████| 2483/2483 [00:39<00:00, 63.38it/s]

In [11]:

X\_test , y\_test **=** get\_data(TEST\_DIR)

100%|██████████| 620/620 [00:17<00:00, 35.22it/s]

100%|██████████| 620/620 [00:15<00:00, 40.49it/s]

In [12]:

print(X\_train**.**shape,'\n',X\_test**.**shape)

(4961, 150, 150, 3)

(1240, 150, 150, 3)

In [13]:

print(y\_train**.**shape,'\n',y\_test**.**shape)

(4961,)

(1240,)

In [12]:

**from** keras.utils.np\_utils **import** to\_categorical

y\_train **=** to\_categorical(y\_train, 2)

y\_test **=** to\_categorical(y\_test, 2)

Using TensorFlow backend.

In [13]:

print(y\_train**.**shape,'\n',y\_test**.**shape)

(4961,)

(1240,)

In [14]:

Pimages **=** os**.**listdir(TRAIN\_DIR **+** "CANCER")

Nimages **=** os**.**listdir(TRAIN\_DIR **+** "NORMAL")

In [15]:

**import** matplotlib.pyplot **as** plt

**def** plotter(i):

imagep1 **=** cv2**.**imread(TRAIN\_DIR**+**"CANCER/"**+**Pimages[i])

imagep1 **=** skimage**.**transform**.**resize(imagep1, (150, 150, 3) , mode **=** 'reflect')

imagen1 **=** cv2**.**imread(TRAIN\_DIR**+**"NORMAL/"**+**Nimages[i])

imagen1 **=** skimage**.**transform**.**resize(imagen1, (150, 150, 3))

pair **=** np**.**concatenate((imagen1, imagep1), axis**=**1)

print("(Left) - No CANCER Vs (Right) - CANCER")

print("-----------------------------------------------------------------------------------------------------------------------------------")

plt**.**figure(figsize**=**(10,5))

plt**.**imshow(pair)

plt**.**show()

**for** i **in** range(0,5):

plotter(i)

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

In [16]:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn **import** metrics

**from** sklearn.metrics **import** accuracy\_score

*#function*

**def** train\_test\_rmse(x,y):

x **=** Iris\_data[x]

y **=** Iris\_data[y]

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(x, y, test\_size **=** 0.2,random\_state**=**123)

linreg **=** LinearRegression()

linreg**.**fit(X\_train, y\_train)

y\_pred **=** linreg**.**predict(X\_test)

print(accuracy\_score(y\_test, y\_pred)) *# or you can save it in variable and return it*

**return** np**.**sqrt(metrics**.**mean\_squared\_error(y\_test, y\_pred))

In [17]:

**import** seaborn **as** sns

count **=** y\_train**.**sum(axis **=** 0)

sns**.**countplot(x **=** count)

Out[17]:

****

In [18]:

**from** keras.callbacks **import** ReduceLROnPlateau , ModelCheckpoint

lr\_reduce **=** ReduceLROnPlateau(monitor**=**'val\_acc', factor**=**0.1, epsilon**=**0.0001, patience**=**1, verbose**=**1)

/home/neuzan/Programs/anaconda3/envs/DeepL/lib/python3.6/site-packages/keras/callbacks.py:1065: UserWarning: `epsilon` argument is deprecated and will be removed, use `min\_delta` instead.

warnings.warn('`epsilon` argument is deprecated and '

In [19]:

filepath**=**"weights.hdf5"

checkpoint **=** ModelCheckpoint(filepath, monitor**=**'val\_acc', verbose**=**1, save\_best\_only**=True**, mode**=**'max')

In [20]:

**from** keras.models **import** Sequential

**from** keras.layers **import** Dense , Activation

**from** keras.layers **import** Dropout

**from** keras.layers **import** Flatten

**from** keras.constraints **import** maxnorm

**from** keras.optimizers **import** SGD , RMSprop

**from** keras.layers **import** Conv2D , BatchNormalization

**from** keras.layers **import** MaxPooling2D

**from** keras.utils **import** np\_utils

**from** keras **import** backend **as** K

K**.**set\_image\_dim\_ordering('th')

**from** sklearn.model\_selection **import** GridSearchCV

**from** keras.wrappers.scikit\_learn **import** KerasClassifier

In [21]:

*#X\_train=X\_train.reshape(5216,3,150,150)*

*#X\_test=X\_test.reshape(624,3,150,150)*

In [22]:

model **=** Sequential()

model**.**add(Conv2D(16, (3, 3), activation**=**'relu', padding**=**"same", input\_shape**=**(150,150,3)))

model**.**add(Conv2D(16, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(Conv2D(32, (3, 3), activation**=**'relu', padding**=**"same"))

model**.**add(Conv2D(32, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(Conv2D(64, (3, 3), activation**=**'relu', padding**=**"same"))

model**.**add(Conv2D(64, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(MaxPooling2D(pool\_size**=**(2, 2)))

model**.**add(Flatten())

model**.**add(Dense(64, activation**=**'relu'))

model**.**add(Dropout(0.2))

model**.**add(Dense(2 , activation**=**'sigmoid'))

model**.**compile(loss**=**'binary\_crossentropy',

optimizer**=**RMSprop(lr**=**0.00005),

metrics**=**['accuracy'])

print(model**.**summary())

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d\_1 (Conv2D) (None, 16, 150, 3) 21616

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_2 (Conv2D) (None, 16, 150, 3) 2320

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_3 (Conv2D) (None, 32, 150, 3) 4640

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_4 (Conv2D) (None, 32, 150, 3) 9248

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_5 (Conv2D) (None, 64, 150, 3) 18496

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_6 (Conv2D) (None, 64, 150, 3) 36928

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_1 (MaxPooling2 (None, 64, 75, 1) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten\_1 (Flatten) (None, 4800) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense) (None, 64) 307264

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout\_1 (Dropout) (None, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_2 (Dense) (None, 2) 130

=================================================================

Total params: 400,642

Trainable params: 400,642

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

In [23]:

batch\_size **=** 256

epochs **=** 10

In [24]:

history **=** model**.**fit(X\_train, y\_train, validation\_data **=** (X\_test , y\_test) ,callbacks**=**[lr\_reduce,checkpoint] ,

epochs**=**epochs)

Train on 5216 samples, validate on 624 samples

Epoch 1/10

5216/5216 [==============================] - 12s 2ms/step - loss: 0.5063 - acc: 0.7597 - val\_loss: 0.4808 - val\_acc: 0.7780

Epoch 00001: val\_acc improved from -inf to 0.77804, saving model to weights.hdf5

Epoch 2/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2925 - acc: 0.8792 - val\_loss: 0.6008 - val\_acc: 0.7252

Epoch 00002: ReduceLROnPlateau reducing learning rate to 4.999999873689376e-06.

Epoch 00002: val\_acc did not improve from 0.77804

Epoch 3/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2312 - acc: 0.9042 - val\_loss: 0.5019 - val\_acc: 0.7780

Epoch 00003: ReduceLROnPlateau reducing learning rate to 4.999999873689376e-07.

Epoch 00003: val\_acc did not improve from 0.77804

Epoch 4/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2249 - acc: 0.9077 - val\_loss: 0.4912 - val\_acc: 0.7821

Epoch 00004: val\_acc improved from 0.77804 to 0.78205, saving model to weights.hdf5

Epoch 5/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2243 - acc: 0.9097 - val\_loss: 0.4968 - val\_acc: 0.7796

Epoch 00005: ReduceLROnPlateau reducing learning rate to 4.999999987376214e-08.

Epoch 00005: val\_acc did not improve from 0.78205

Epoch 6/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2251 - acc: 0.9078 - val\_loss: 0.4975 - val\_acc: 0.7796

Epoch 00006: ReduceLROnPlateau reducing learning rate to 5.000000058430488e-09.

Epoch 00006: val\_acc did not improve from 0.78205

Epoch 7/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2224 - acc: 0.9094 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00007: ReduceLROnPlateau reducing learning rate to 4.999999969612646e-10.

Epoch 00007: val\_acc did not improve from 0.78205

Epoch 8/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2255 - acc: 0.9078 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00008: ReduceLROnPlateau reducing learning rate to 4.999999858590343e-11.

Epoch 00008: val\_acc did not improve from 0.78205

Epoch 9/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2243 - acc: 0.9085 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00009: ReduceLROnPlateau reducing learning rate to 4.999999719812465e-12.

Epoch 00009: val\_acc did not improve from 0.78205

Epoch 10/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2242 - acc: 0.9088 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00010: ReduceLROnPlateau reducing learning rate to 4.999999546340118e-13.

Epoch 00010: val\_acc did not improve from 0.78205

In [34]:

model**.**save('mymodel.h5')

In [27]:

**import** matplotlib.pyplot **as** plt

**from** keras.models **import** load\_model

plt**.**plot(history**.**history['acc'])

plt**.**plot(history**.**history['val\_acc'])

plt**.**title('model accuracy')

plt**.**ylabel('accuracy')

plt**.**xlabel('epoch')

plt**.**legend(['train', 'test'], loc**=**'upper left')

plt**.**show()

*# summarize history for loss*

plt**.**plot(history**.**history['loss'])

plt**.**plot(history**.**history['val\_loss'])

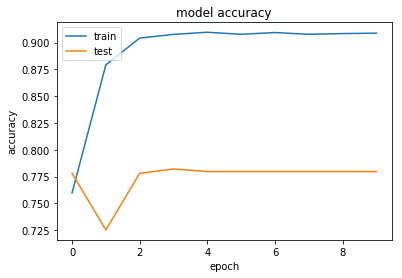
plt**.**title('model loss')

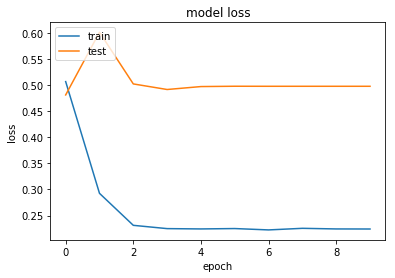
plt**.**ylabel('loss')

plt**.**xlabel('epoch')

plt**.**legend(['train', 'test'], loc**=**'upper left')

plt**.**show()

****

****

In [28]:

**from** sklearn.metrics **import** confusion\_matrix

pred **=** model**.**predict(X\_test)

pred **=** np**.**argmax(pred,axis **=** 1)

y\_true **=** np**.**argmax(y\_test,axis **=** 1)

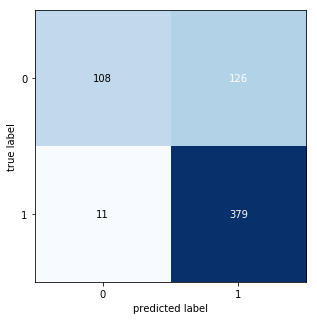
In [29]:

CM **=** confusion\_matrix(y\_true, pred)

**from** mlxtend.plotting **import** plot\_confusion\_matrix

fig, ax **=** plot\_confusion\_matrix(conf\_mat**=**CM , figsize**=**(5, 5))

plt**.**show()

****

In [16]:

*#PRECISION = (TP/(TP+FP))*

379**/**(379**+**126)

Out[16]:

0.7504950495049505

In [17]:

*#RECALL = (TP/(TP+FN))*

379 **/** (379 **+** 11)

Out[17]:

0.9717948717948718

In [18]:

*#ACCURACY = (TP+TN)/(TP+TN+FP+FN)*

(379**+**108)**/**(379**+**108**+**126**+**11)

Out[18]:

0.780448717948718

In [ ]:

**import** cv2

**import** numpy **as** np

**import** os

**from** random **import** shuffle

**from** tqdm **import** tqdm

**import** scipy

**import** skimage

**from** skimage.transform **import** resize

**from** sklearn.model\_selection **import** train\_test\_split

print(os**.**listdir("cancer/"))

['train', 'test', 'val', '.DS\_Store']

In [4]:

print(os**.**listdir("cancer/train"))

['Cancer', 'Normal']

In [5]:

TRAIN\_DIR **=** "cancer/train/"

TEST\_DIR **=** "cancer/test/"

In [6]:

*#Preprocessing*

**def** get\_label(Dir):

**for** nextdir **in** os**.**listdir(Dir):

**if** **not** nextdir**.**startswith('.'):

**if** nextdir **in** ['NORMAL']:

label **=** 0

**elif** nextdir **in** ['CANCER']:

label **=** 1

**else**:

label **=** 2

**return** nextdir, label

In [7]:

**def** preprocessing\_data(Dir):

X **=** []

y **=** []

**for** nextdir **in** os**.**listdir(Dir):

nextdir, label **=** get\_label(Dir)

temp **=** Dir **+** nextdir

**for** image\_filename **in** tqdm(os**.**listdir(temp)):

path **=** os**.**path**.**join(temp **+** '/' , image\_filename)

img **=** cv2**.**imread(path,cv2**.**IMREAD\_GRAYSCALE)

**if** img **is** **not** **None**:

img **=** skimage**.**transform**.**resize(img, (150, 150, 3))

img **=** np**.**asarray(img)

X**.**append(img)

y**.**append(label)

X **=** np**.**asarray(X)

y **=** np**.**asarray(y)

**return** X,y

In [7]:

*#X\_train, y\_train = preprocessing\_data(TRAIN\_DIR)*

In [8]:

**def** get\_data(Dir):

X **=** []

y **=** []

**for** nextDir **in** os**.**listdir(Dir):

**if** **not** nextDir**.**startswith('.'):

**if** nextDir **in** ['NORMAL']:

label **=** 0

**elif** nextDir **in** ['CANCER']:

label **=** 1

**else**:

label **=** 2

temp **=** Dir **+** nextDir

**for** file **in** tqdm(os**.**listdir(temp)):

img **=** cv2**.**imread(temp **+** '/' **+** file)

**if** img **is** **not** **None**:

img **=** skimage**.**transform**.**resize(img, (150, 150, 3))

*#img\_file = scipy.misc.imresize(arr=img\_file, size=(150, 150, 3))*

img **=** np**.**asarray(img)

X**.**append(img)

y**.**append(label)

X **=** np**.**asarray(X)

y **=** np**.**asarray(y)

**return** X,y

In [10]:

X\_train, y\_train **=** get\_data(TRAIN\_DIR)

100%|██████████| 2478/2478 [00:27<00:00, 23.64it/s]

100%|██████████| 2483/2483 [00:39<00:00, 63.38it/s]

In [11]:

X\_test , y\_test **=** get\_data(TEST\_DIR)

100%|██████████| 620/620 [00:17<00:00, 35.22it/s]

100%|██████████| 620/620 [00:15<00:00, 40.49it/s]

In [12]:

print(X\_train**.**shape,'\n',X\_test**.**shape)

(4961, 150, 150, 3)

(1240, 150, 150, 3)

In [13]:

print(y\_train**.**shape,'\n',y\_test**.**shape)

(4961,)

(1240,)

In [12]:

**from** keras.utils.np\_utils **import** to\_categorical

y\_train **=** to\_categorical(y\_train, 2)

y\_test **=** to\_categorical(y\_test, 2)

Using TensorFlow backend.

In [13]:

print(y\_train**.**shape,'\n',y\_test**.**shape)

(4961,)

(1240,)

In [14]:

Pimages **=** os**.**listdir(TRAIN\_DIR **+** "CANCER")

Nimages **=** os**.**listdir(TRAIN\_DIR **+** "NORMAL")

In [15]:

**import** matplotlib.pyplot **as** plt

**def** plotter(i):

imagep1 **=** cv2**.**imread(TRAIN\_DIR**+**"CANCER/"**+**Pimages[i])

imagep1 **=** skimage**.**transform**.**resize(imagep1, (150, 150, 3) , mode **=** 'reflect')

imagen1 **=** cv2**.**imread(TRAIN\_DIR**+**"NORMAL/"**+**Nimages[i])

imagen1 **=** skimage**.**transform**.**resize(imagen1, (150, 150, 3))

pair **=** np**.**concatenate((imagen1, imagep1), axis**=**1)

print("(Left) - No CANCER Vs (Right) - CANCER")

print("-----------------------------------------------------------------------------------------------------------------------------------")

plt**.**figure(figsize**=**(10,5))

plt**.**imshow(pair)

plt**.**show()

**for** i **in** range(0,5):

plotter(i)

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

(Left) - No CANCER Vs (Right) - CANCER

-----------------------------------------------------------------------------------------------------------------------------------

In [16]:

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn **import** metrics

**from** sklearn.metrics **import** accuracy\_score

*#function*

**def** train\_test\_rmse(x,y):

x **=** Iris\_data[x]

y **=** Iris\_data[y]

X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(x, y, test\_size **=** 0.2,random\_state**=**123)

linreg **=** LinearRegression()

linreg**.**fit(X\_train, y\_train)

y\_pred **=** linreg**.**predict(X\_test)

print(accuracy\_score(y\_test, y\_pred)) *# or you can save it in variable and return it*

**return** np**.**sqrt(metrics**.**mean\_squared\_error(y\_test, y\_pred))

In [17]:

**import** seaborn **as** sns

count **=** y\_train**.**sum(axis **=** 0)

sns**.**countplot(x **=** count)

Out[17]:



In [18]:

**from** keras.callbacks **import** ReduceLROnPlateau , ModelCheckpoint

lr\_reduce **=** ReduceLROnPlateau(monitor**=**'val\_acc', factor**=**0.1, epsilon**=**0.0001, patience**=**1, verbose**=**1)

/home/neuzan/Programs/anaconda3/envs/DeepL/lib/python3.6/site-packages/keras/callbacks.py:1065: UserWarning: `epsilon` argument is deprecated and will be removed, use `min\_delta` instead.

warnings.warn('`epsilon` argument is deprecated and '

In [19]:

filepath**=**"weights.hdf5"

checkpoint **=** ModelCheckpoint(filepath, monitor**=**'val\_acc', verbose**=**1, save\_best\_only**=True**, mode**=**'max')

In [20]:

**from** keras.models **import** Sequential

**from** keras.layers **import** Dense , Activation

**from** keras.layers **import** Dropout

**from** keras.layers **import** Flatten

**from** keras.constraints **import** maxnorm

**from** keras.optimizers **import** SGD , RMSprop

**from** keras.layers **import** Conv2D , BatchNormalization

**from** keras.layers **import** MaxPooling2D

**from** keras.utils **import** np\_utils

**from** keras **import** backend **as** K

K**.**set\_image\_dim\_ordering('th')

**from** sklearn.model\_selection **import** GridSearchCV

**from** keras.wrappers.scikit\_learn **import** KerasClassifier

In [21]:

*#X\_train=X\_train.reshape(5216,3,150,150)*

*#X\_test=X\_test.reshape(624,3,150,150)*

In [22]:

model **=** Sequential()

model**.**add(Conv2D(16, (3, 3), activation**=**'relu', padding**=**"same", input\_shape**=**(150,150,3)))

model**.**add(Conv2D(16, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(Conv2D(32, (3, 3), activation**=**'relu', padding**=**"same"))

model**.**add(Conv2D(32, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(Conv2D(64, (3, 3), activation**=**'relu', padding**=**"same"))

model**.**add(Conv2D(64, (3, 3), padding**=**"same", activation**=**'relu'))

model**.**add(MaxPooling2D(pool\_size**=**(2, 2)))

model**.**add(Flatten())

model**.**add(Dense(64, activation**=**'relu'))

model**.**add(Dropout(0.2))

model**.**add(Dense(2 , activation**=**'sigmoid'))

model**.**compile(loss**=**'binary\_crossentropy',

optimizer**=**RMSprop(lr**=**0.00005),

metrics**=**['accuracy'])

print(model**.**summary())

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d\_1 (Conv2D) (None, 16, 150, 3) 21616

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_2 (Conv2D) (None, 16, 150, 3) 2320

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_3 (Conv2D) (None, 32, 150, 3) 4640

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_4 (Conv2D) (None, 32, 150, 3) 9248

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_5 (Conv2D) (None, 64, 150, 3) 18496

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

conv2d\_6 (Conv2D) (None, 64, 150, 3) 36928

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

max\_pooling2d\_1 (MaxPooling2 (None, 64, 75, 1) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten\_1 (Flatten) (None, 4800) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense) (None, 64) 307264

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout\_1 (Dropout) (None, 64) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_2 (Dense) (None, 2) 130

=================================================================

Total params: 400,642

Trainable params: 400,642

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

In [23]:

batch\_size **=** 256

epochs **=** 10

In [24]:

history **=** model**.**fit(X\_train, y\_train, validation\_data **=** (X\_test , y\_test) ,callbacks**=**[lr\_reduce,checkpoint] ,

epochs**=**epochs)

Train on 5216 samples, validate on 624 samples

Epoch 1/10

5216/5216 [==============================] - 12s 2ms/step - loss: 0.5063 - acc: 0.7597 - val\_loss: 0.4808 - val\_acc: 0.7780

Epoch 00001: val\_acc improved from -inf to 0.77804, saving model to weights.hdf5

Epoch 2/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2925 - acc: 0.8792 - val\_loss: 0.6008 - val\_acc: 0.7252

Epoch 00002: ReduceLROnPlateau reducing learning rate to 4.999999873689376e-06.

Epoch 00002: val\_acc did not improve from 0.77804

Epoch 3/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2312 - acc: 0.9042 - val\_loss: 0.5019 - val\_acc: 0.7780

Epoch 00003: ReduceLROnPlateau reducing learning rate to 4.999999873689376e-07.

Epoch 00003: val\_acc did not improve from 0.77804

Epoch 4/10

5216/5216 [==============================] - 6s 1ms/step - loss: 0.2249 - acc: 0.9077 - val\_loss: 0.4912 - val\_acc: 0.7821

Epoch 00004: val\_acc improved from 0.77804 to 0.78205, saving model to weights.hdf5

Epoch 5/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2243 - acc: 0.9097 - val\_loss: 0.4968 - val\_acc: 0.7796

Epoch 00005: ReduceLROnPlateau reducing learning rate to 4.999999987376214e-08.

Epoch 00005: val\_acc did not improve from 0.78205

Epoch 6/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2251 - acc: 0.9078 - val\_loss: 0.4975 - val\_acc: 0.7796

Epoch 00006: ReduceLROnPlateau reducing learning rate to 5.000000058430488e-09.

Epoch 00006: val\_acc did not improve from 0.78205

Epoch 7/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2224 - acc: 0.9094 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00007: ReduceLROnPlateau reducing learning rate to 4.999999969612646e-10.

Epoch 00007: val\_acc did not improve from 0.78205

Epoch 8/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2255 - acc: 0.9078 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00008: ReduceLROnPlateau reducing learning rate to 4.999999858590343e-11.

Epoch 00008: val\_acc did not improve from 0.78205

Epoch 9/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2243 - acc: 0.9085 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00009: ReduceLROnPlateau reducing learning rate to 4.999999719812465e-12.

Epoch 00009: val\_acc did not improve from 0.78205

Epoch 10/10

5216/5216 [==============================] - 7s 1ms/step - loss: 0.2242 - acc: 0.9088 - val\_loss: 0.4974 - val\_acc: 0.7796

Epoch 00010: ReduceLROnPlateau reducing learning rate to 4.999999546340118e-13.

Epoch 00010: val\_acc did not improve from 0.78205

In [34]:

model**.**save('mymodel.h5')

In [27]:

**import** matplotlib.pyplot **as** plt

**from** keras.models **import** load\_model

plt**.**plot(history**.**history['acc'])

plt**.**plot(history**.**history['val\_acc'])

plt**.**title('model accuracy')

plt**.**ylabel('accuracy')

plt**.**xlabel('epoch')

plt**.**legend(['train', 'test'], loc**=**'upper left')

plt**.**show()

*# summarize history for loss*

plt**.**plot(history**.**history['loss'])

plt**.**plot(history**.**history['val\_loss'])

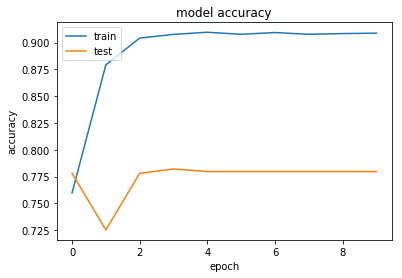
plt**.**title('model loss')

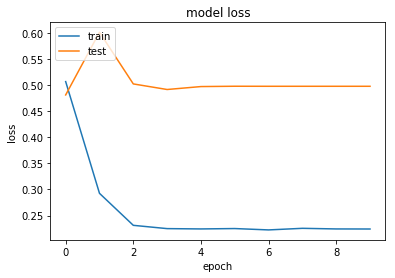
plt**.**ylabel('loss')

plt**.**xlabel('epoch')

plt**.**legend(['train', 'test'], loc**=**'upper left')

plt**.**show()





In [28]:

**from** sklearn.metrics **import** confusion\_matrix

pred **=** model**.**predict(X\_test)

pred **=** np**.**argmax(pred,axis **=** 1)

y\_true **=** np**.**argmax(y\_test,axis **=** 1)

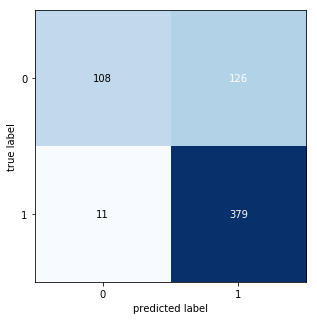
In [29]:

CM **=** confusion\_matrix(y\_true, pred)

**from** mlxtend.plotting **import** plot\_confusion\_matrix

fig, ax **=** plot\_confusion\_matrix(conf\_mat**=**CM , figsize**=**(5, 5))

plt**.**show()



In [16]:

*#PRECISION = (TP/(TP+FP))*

379**/**(379**+**126)

Out[16]:

0.7504950495049505

In [17]:

*#RECALL = (TP/(TP+FN))*

379 **/** (379 **+** 11)

Out[17]:

0.9717948717948718

In [18]:

*#ACCURACY = (TP+TN)/(TP+TN+FP+FN)*

(379**+**108)**/**(379**+**108**+**126**+**11)

Out[18]:

0.780448717948718

In [ ]: