In [1]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report,confusion_matrix
from sklearn import metrics
import numpy as np
import pandas as pd
import os
import cv2
```

In [2]:
```
path = os.listdir('brain_tumor/train') # Path of the dataset
classes = {'glioma_tumor':0, 'meningioma_tumor':1, 'no_tumor':2, 'pituitary_tumor':3} # Class labels
```

In [3]:
```
X = []
Y = []
for cls in classes:
    pth = 'brain_tumor/train/' + cls
    for j in os.listdir(pth):
        img = cv2.imread(pth+'/'+j, 0) # 0 means gray color
        img = cv2.resize(img, (200,200)) # resize the image
        X.append(img)
        Y.append(classes[cls])
```

In [4]:
```
np.unique(Y)
```

Out[4]:
```
array([0, 1, 2, 3])
```

In [5]:
```
X = np.array(X)
Y = np.array(Y)
```

In [6]:
```
pd.Series(Y).value_counts() # Show each class label instances
```

Out[6]:
```
3    827
0    826
1    822
2    395
dtype: int64
```

In [7]:
```
X.shape # Show the total number of images and the Dimensions
```

Out[7]:
```
(2870, 200, 200)
```

In [8]:
```
X_updated = X.reshape(len(X), -1)
X_updated.shape
```

Out[8]:
```
(2870, 40000)
```

In [9]:
```
xtrain, xtest, ytrain, ytest = train_test_split(X_updated, Y, random_state=10, test_size =.20)
```

In [10]:
```
xtrain.shape, xtest.shape
```

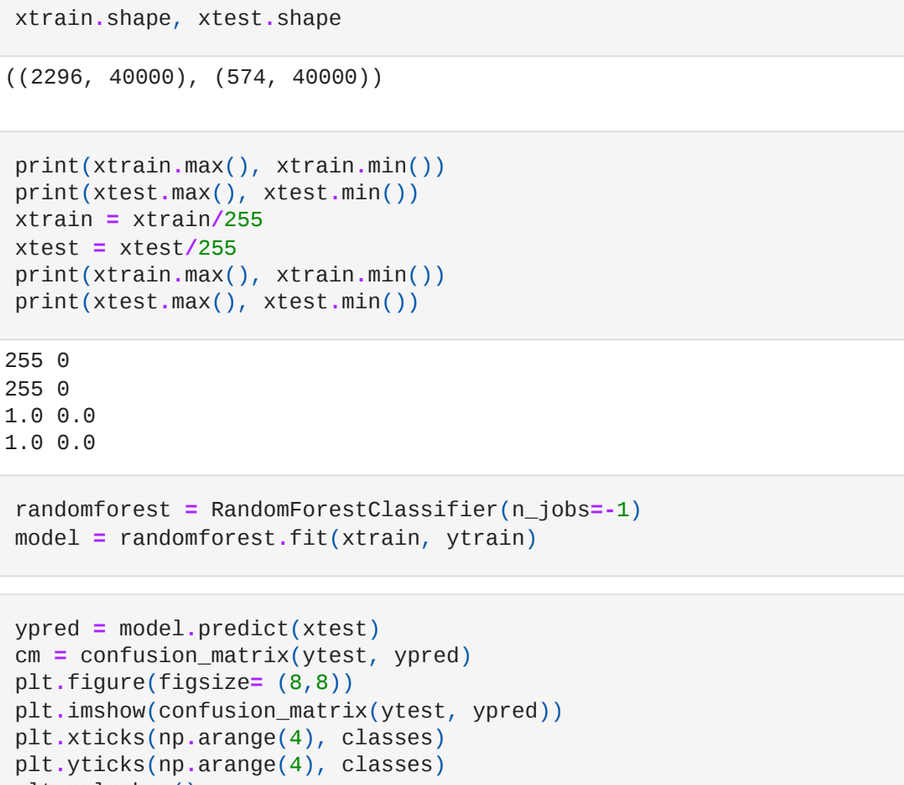
Out[10]:
```
((2296, 40000), (574, 40000))
```

In [11]:
```
print(xtrain.max(), xtrain.min())
print(xtest.max(), xtest.min())
xtrain = xtrain/255
xtest = xtest/255
print(xtrain.max(), xtrain.min())
print(xtest.max(), xtest.min())
```

```
255 0
255 0
1.0 0.0
1.0 0.0
```

In [13]:
```
randomforest = RandomForestClassifier(n_jobs=-1)
model = randomforest.fit(xtrain, ytrain)
```

In [14]:
```
ypred = model.predict(xtest)
cm = confusion_matrix(ytest, ypred)
plt.figure(figsize= (8,8))
plt.imshow(confusion_matrix(ytest, ypred))
plt.xticks(np.arange(4), classes)
plt.yticks(np.arange(4), classes)
plt.colorbar()
plt.show()
print(cm)
```




```
[[125  32   0   3]
 [  1 142  17   9]
 [  1   5  63   0]
 [  1   4   0 171]]
```

In [15]:
```
report = classification_report(ytest, ypred)
print(report)
```

```
              precision    recall  f1-score   support

           0       0.98      0.78      0.87       160
           1       0.78      0.84      0.81       169
           2       0.79      0.91      0.85        69
           3       0.93      0.97      0.95       176

    accuracy                           0.87       574
   macro avg       0.87      0.88      0.87       574
weighted avg       0.88      0.87      0.87       574
```

In [ ]:
```

```