In [2]:

```
import warnings
warnings.filterwarnings('ignore')
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.metrics import accuracy_score
import os
import cv2
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn import preprocessing
from sklearn import utils
```

In [3]:

```
path=os.listdir('C:/Users/admin/Desktop/archive (9)/data')
classes={'no':0,'yes':1}
```

In [4]:

```
X=[]
Y=[]
for cls in classes:
    pth='C:/Users/admin/Desktop/archive (9)/data/'+cls
    for j in os.listdir(pth):
        img=cv2.imread(pth+'/'+j,0)
        img=cv2.resize(img,(200,200))
        X.append(img)
        Y.append(classes[cls])
```

In [5]:

```
np.unique(Y)
```

Out[5]:

```
array([0, 1])
```

In [6]:

```
X=np.array(X)
Y=np.array(Y)
```

In [7]:

```
pd.Series(Y).value_counts()
```

Out[7]:

```
0    232
1    212
dtype: int64
```

In [8]:

```
X.shape
```

Out[8]:

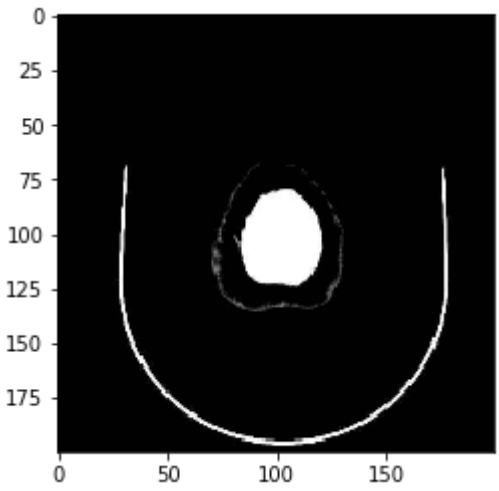
```
(444, 200, 200)
```

In [9]:

```
plt.imshow(X[0],cmap='gray')
```

Out[9]:

```
<matplotlib.image.AxesImage at 0x1c2e0aa1948>
```



In [10]:

```
X_updated=X.reshape(len(X),-1)
X_updated.shape
```

Out[10]:

```
(444, 40000)
```

In [11]:

```
xtrain, xtest, ytrain,ytest=train_test_split(X_updated,Y,random_state=10,test_size=0.20
)
```

In [12]:

```
xtrain.shape,xtest.shape
```

Out[12]:

```
((355, 40000), (89, 40000))
```

In [13]:

```
xtrain=xtrain/255
ytrain=ytrain/255
```

In [14]:

```
from sklearn.decomposition import PCA
```

In [15]:

```
print(xtrain.shape,xtest.shape)
```

```
(355, 40000) (89, 40000)
```

In [16]:

```
lab_enc = preprocessing.LabelEncoder()
encoded = lab_enc.fit_transform(ytrain)
```

In [23]:

```
lg = LogisticRegression(C=0.0012689610031679222,
                        solver='liblinear')
lg.fit(xtrain,encoded)
```

Out[23]:

```
LogisticRegression(C=0.0012689610031679222, class_weight=None, dual=False,
                   fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                   max_iter=100, multi_class='warn', n_jobs=None, penalty
='l2',
                   random_state=None, solver='liblinear', tol=0.0001, verb
ose=0,
                   warm_start=False)
```

In [24]:

```
from sklearn.model_selection import cross_val_score
cv_acc = cross_val_score(lg,
                         X_updated,
                         Y,
                         cv=5,
                         scoring="accuracy")

pred=lg.predict(xtest)
np.where(ytest!=pred)
```

Out[24]:

```
(array([], dtype=int64),)
```

In [26]:

```
dec={0:'no hemorrhage',1:'yes hemorrhage'}
```

In [27]:

```
plt.figure(figsize=(12,8))
```

Out[27]:

```
<Figure size 864x576 with 0 Axes>
```

```
<Figure size 864x576 with 0 Axes>
```

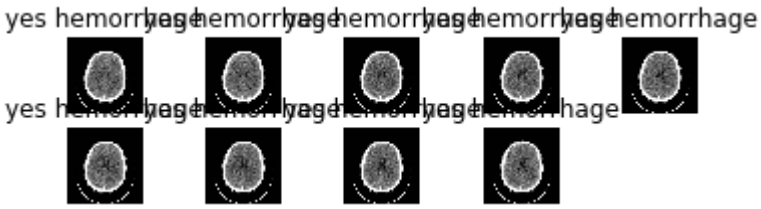
In [28]:

```
p=os.listdir('C:/Users/admin/Desktop/archive (9)/data/')
c=1
```

In [29]:

```
p=os.listdir('C:/Users/admin/Desktop/archive (9)/data/')
c=1
for i in os.listdir('C:/Users/admin/Desktop/archive (9)/data/yes')[:9]:
    plt.subplot(5,5,c)
    img=cv2.imread('C:/Users/admin/Desktop/archive (9)/data/yes/'+i,0)
    img1=cv2.resize(img,(200,200))
    img1=img1.reshape(1,-1)/255
    p=lg.predict(img1)
    plt.title(dec[p[0]])
    plt.imshow(img,cmap='gray')
    plt.axis('off')
    print(i)
    c+=1
```

```
1_0_100.jpg
1_0_101.jpg
1_0_102.jpg
1_0_103.jpg
1_0_104.jpg
1_0_105.jpg
1_0_106.jpg
1_0_107.jpg
1_0_108.jpg
```



In [51]:

```
import joblib
model_name="Hemorrhage_Detection.sav"
joblib.dump(lg, model_name)
```

Out[51]:

```
['Hemorrhage_Detection.sav']
```

In [ ]: