In [ ]:

```
#Create your own dataset and apply Logistic Regression on the created dataset. Print the predicted score and confusion matrix.
```

In [1]:

```
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
```

In [2]:

```
x = np.arange(10).reshape(-1, 1)
print(x)
```

```
[[0]
 [1]
 [2]
 [3]
 [4]
 [5]
 [6]
 [7]
 [8]
 [9]]
```

In [3]:

```
y = np.array([0, 0, 0, 0, 1, 1, 1, 1, 1, 1])
```

In [4]:

```
model = LogisticRegression(solver='liblinear', random_state=0)
```

In [5]:

```
model.fit(x, y)
```

Out[5]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=0, solver='liblinear', tol=0.0001, verbose=0,
                   warm_start=False)
```

In [6]:

```
model.predict_proba(x)
```

Out[6]:

```
array([[0.74002157, 0.25997843],
       [0.62975524, 0.37024476],
       [0.5040632 , 0.4959368 ],
       [0.37785549, 0.62214451],
       [0.26628093, 0.73371907],
       [0.17821501, 0.82178499],
       [0.11472079, 0.88527921],
       [0.07186982, 0.92813018],
       [0.04422513, 0.95577487],
       [0.02690569, 0.97309431]])
```

In [7]:

```
print(model.predict(x))
```

```
[0 0 0 1 1 1 1 1 1 1]
```

In [8]:

```
model.score(x, y)
```

Out[8]:

```
0.9
```

In [9]:

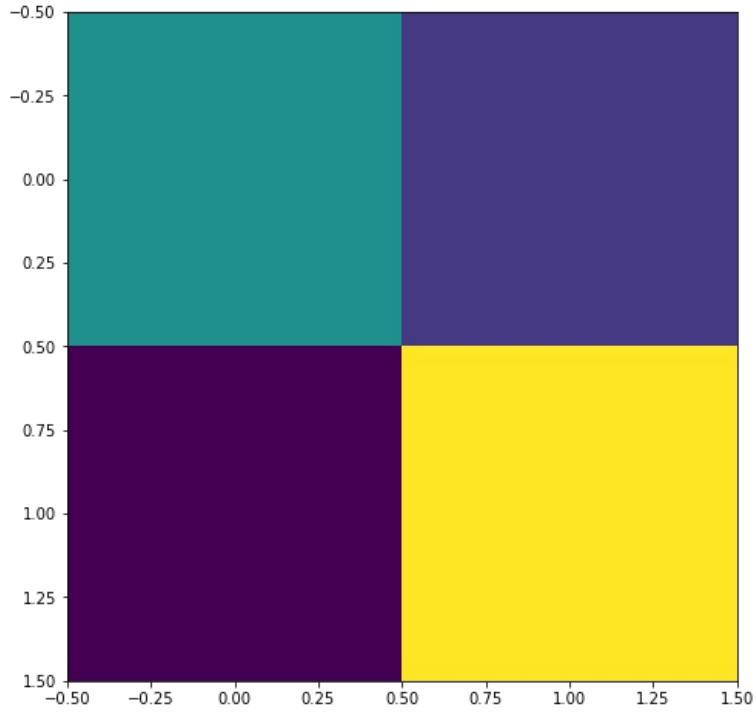
```
cm = confusion_matrix(y, model.predict(x))
```

In [10]:

```
fig, ax = plt.subplots(figsize=(8, 8))
ax.imshow(cm)
```

Out[10]:

```
<matplotlib.image.AxesImage at 0x2b35e0b9608>
```



In [11]:

```
ax.grid(False)
ax.xaxis.set(ticks=(0, 1), ticklabels=('Predicted 0s', 'Predicted 1s'))
ax.yaxis.set(ticks=(0, 1), ticklabels=('Actual 0s', 'Actual 1s'))
ax.set_ylim(1.5, -0.5)
```

Out[11]:

```
(1.5, -0.5)
```

In [12]:

```
for i in range(2):
    for j in range(2):
        ax.text(j, i, cm[i, j], ha='center', va='center', color='red')
```

In [13]:

```
plt.show()
```

In [14]:

```
print(classification_report(y, model.predict(x)))
```

```
              precision    recall  f1-score   support

           0       1.00      0.75      0.86         4
           1       0.86      1.00      0.92         6

    accuracy                           0.90        10
   macro avg       0.93      0.88      0.89        10
weighted avg       0.91      0.90      0.90        10
```