EXERCISE 13

From sklearn.datasets load digits dataset and do following

1. Classify digits (0 to 9) using KNN classifier. You can use different values for k neighbors and need to figure out a value of K that gives you a maximum score. You can manually try different values of K or use gridsearchcv
2. Plot confusion matrix
3. Plot classification report

Importing all modules & libraries

In [48]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.datasets import load_digits
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold, cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import confusion_matrix
```

In [49]:
```
data = load_digits()
```

In [50]:
```
dir(data)
```

Out[50]:
```
['DESCR', 'data', 'feature_names', 'frame', 'images', 'target', 'target_names']
```

Making data into Dataframe (Not required)

In [51]:
```
df = pd.DataFrame(data.data, columns=data.feature_names)
df.head()
```

Out[51]:

	pixel_0_0	pixel_0_1	pixel_0_2	pixel_0_3	pixel_0_4	pixel_0_5	pixel_0_6	pixel_0_7	pixel_1_0	pixel_1_1	...	pixel_6_6	pixel_
0	0.0	0.0	5.0	13.0	9.0	1.0	0.0	0.0	0.0	0.0	...	0.0	
1	0.0	0.0	0.0	12.0	13.0	5.0	0.0	0.0	0.0	0.0	...	0.0	
2	0.0	0.0	0.0	4.0	15.0	12.0	0.0	0.0	0.0	0.0	...	5.0	
3	0.0	0.0	7.0	15.0	13.0	1.0	0.0	0.0	0.0	8.0	...	9.0	
4	0.0	0.0	0.0	1.0	11.0	0.0	0.0	0.0	0.0	0.0	...	0.0	

5 rows × 64 columns

Splitting X and y (Not required)

In [52]:
```
X = data.data
y = data.target
```

Splitting Train Test dataset (Not required)

In [53]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Using GridSearchCV to find optimal K value

In [54]:
```
clf = GridSearchCV(KNeighborsClassifier(), {
    'n_neighbors' : [3, 5, 7]
}, cv=5, return_train_score=False)
```

In [55]:
```
clf.fit(data.data, data.target)
```

Out[55]:

GridSearchCV
estimator: KNeighborsClassifier
KNeighborsClassifier

Checking CV Result

In [56]:
```
clf.cv_results_
```

Out[56]:
```
{'mean_fit_time': array([0.00255718, 0.00166116, 0.0010591 ]),
 'std_fit_time': array([0.00247732, 0.00197151, 0.00211821]),
 'mean_score_time': array([0.00782285, 0.00232501, 0.01273012]),
 'std_score_time': array([0.00464995, 0.00465002, 0.00691768]),
 'param_n_neighbors': masked_array(data=[3, 5, 7],
              mask=[False, False, False],
        fill_value='?',
             dtype=object),
 'params': [{'n_neighbors': 3}, {'n_neighbors': 5}, {'n_neighbors': 7}],
 'split0_test_score': array([0.95555556, 0.94722222, 0.93611111]),
 'split1_test_score': array([0.95833333, 0.95555556, 0.96111111]),
 'split2_test_score': array([0.96657382, 0.96657382, 0.96935933]),
 'split3_test_score': array([0.98607242, 0.98050139, 0.98050139]),
 'split4_test_score': array([0.96657382, 0.9637883 , 0.95264624]),
 'mean_test_score': array([0.96662179, 0.96272826, 0.95994584]),
 'std_test_score': array([0.01067232, 0.01116854, 0.01505909]),
 'rank_test_score': array([1, 2, 3])}
```

Making CV result into DataFrame

In [57]:
```
newdf = pd.DataFrame(clf.cv_results_)
newdf
```

Out[57]:

	mean_fit_time	std_fit_time	mean_score_time	std_score_time	param_n_neighbors	params	split0_test_score	split1_test_s
0	0.002557	0.002477	0.007823	0.004650	3	{'n_neighbors': 3}	0.955556	0.95
1	0.001661	0.001972	0.002325	0.004650	5	{'n_neighbors': 5}	0.947222	0.95
2	0.001059	0.002118	0.012730	0.006918	7	{'n_neighbors': 7}	0.936111	0.96

Getting only useful columns that is n_neighbors, mean_test_score and rank_test_score

In [58]:
```
newdf[['param_n_neighbors','mean_test_score', 'rank_test_score']]
```

Out[58]:

	param_n_neighbors	mean_test_score	rank_test_score
0	3	0.966622	1
1	5	0.962728	2
2	7	0.959946	3

Getting the Best parameter for our KNN

In [59]:
```
clf.best_params_
```

Out[59]:
```
{'n_neighbors': 3}
```

Getting the Best score for our KNN

In [60]:
```
clf.best_score_
```

Out[60]:
```
0.966621788919839
```

PREDICTION

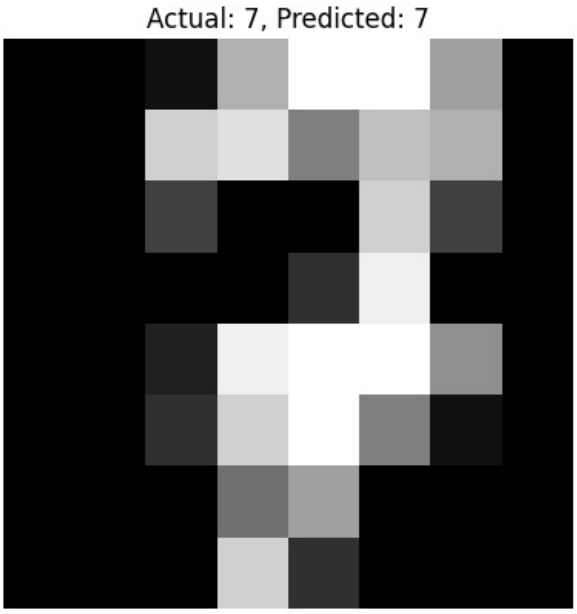
In [61]:
```
predict = clf.predict(X_test)
predict
```

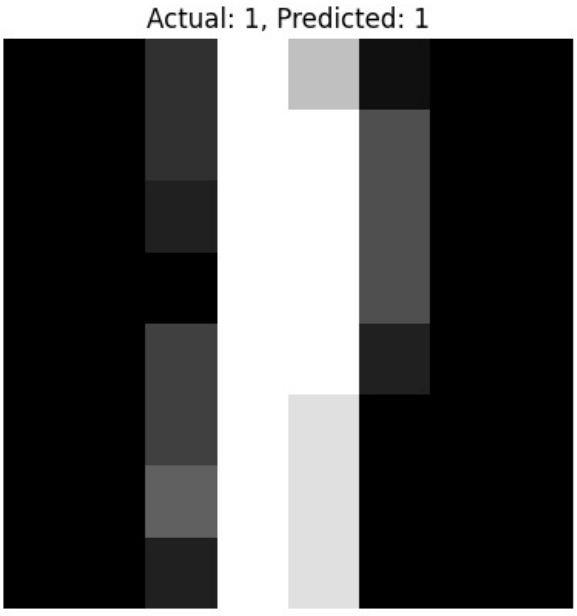
Out[61]:
```
array([7, 1, 4, 0, 1, 0, 6, 2, 5, 1, 1, 3, 4, 7, 6, 6, 1, 6, 2, 3, 0, 5,
       0, 2, 1, 8, 1, 5, 7, 2, 1, 4, 2, 7, 8, 0, 6, 3, 7, 2, 3, 5, 3, 9,
       2, 6, 3, 6, 9, 1, 2, 9, 6, 1, 3, 8, 8, 3, 3, 0, 5, 6, 9, 1, 6, 3,
       8, 0, 1, 4, 7, 7, 0, 4, 8, 7, 1, 9, 8, 2, 9, 4, 5, 6, 7, 4, 9, 0,
       0, 8, 2, 2, 4, 9, 9, 7, 9, 2, 8, 4, 1, 1, 0, 7, 0, 1, 7, 5, 3, 0,
       7, 3, 1, 3, 2, 5, 6, 5, 3, 2, 4, 4, 3, 6, 7, 4, 4, 6, 4, 8, 0, 7,
       8, 5, 3, 2, 5, 0, 1, 7, 2, 7, 0, 0, 4, 4, 2, 3, 5, 7, 8, 7, 7, 8,
       4, 7, 5, 5, 4, 3, 4, 0, 6, 2, 9, 5, 3, 9, 5, 5, 6, 8, 3, 4, 3, 6,
       0, 9, 1, 5, 5, 9, 5, 5, 8, 3, 9, 2, 1, 1, 1, 3, 4, 3, 1, 2, 4, 7,
       4, 7, 3, 6, 3, 3, 3, 9, 3, 0, 0, 4, 8, 0, 3, 5, 4, 1, 9, 3, 8, 0,
       3, 1, 4, 7, 2, 8, 3, 0, 8, 8, 6, 5, 5, 1, 9, 6, 3, 5, 6, 6, 5, 9,
       2, 6, 2, 0, 5, 1, 5, 0, 2, 0, 4, 5, 7, 5, 2, 4, 5, 4, 9, 9, 5, 1,
       3, 9, 9, 2, 6, 2, 6, 4, 9, 9, 1, 0, 7, 5, 8, 7, 0, 9, 5, 0, 3, 1,
       1, 3, 7, 9, 5, 2, 8, 1, 7, 1, 2, 2, 0, 2, 2, 5, 6, 2, 6, 4, 4, 3,
       5, 1, 4, 8, 3, 2, 3, 6, 6, 3, 7, 6, 2, 6, 0, 9, 9, 3, 0, 9, 4, 9,
       7, 2, 5, 2, 3, 1, 5, 3, 9, 5, 1, 6, 4, 5, 8, 3, 8, 4, 6, 4, 4, 6,
       9, 6, 8, 5, 8, 2, 1, 7])
```

Visualizing truth and predicted value

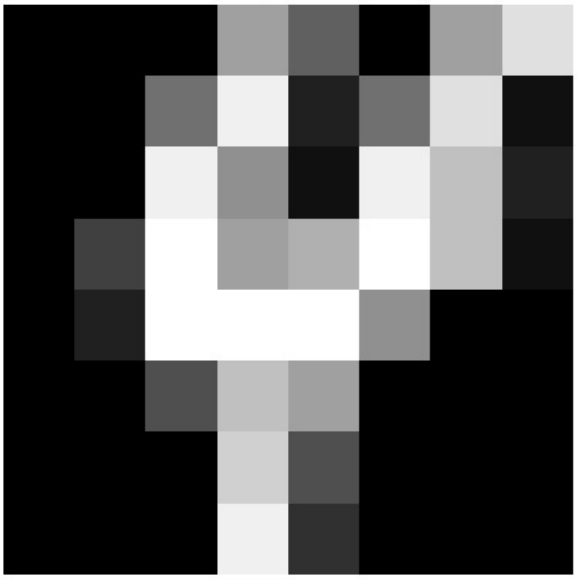
In [62]:
```
# Display actual and predicted values alongside the images
for i in range(len(predict)):
    plt.imshow(X_test[i].reshape(8, 8), cmap='gray')  # Reshape the flattened image to its original shape
    plt.title(f'Actual: {y_test[i]}, Predicted: {predict[i]}')
    plt.axis('off')  # Turn off axis labels for better visualization

    plt.show()
```

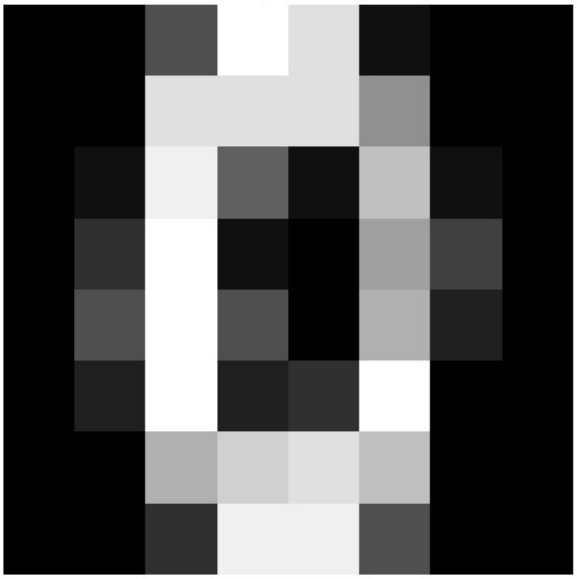




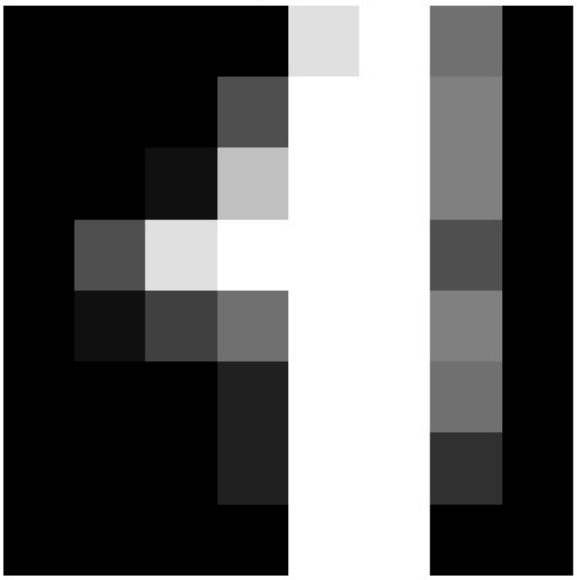
Actual: 4, Predicted: 4



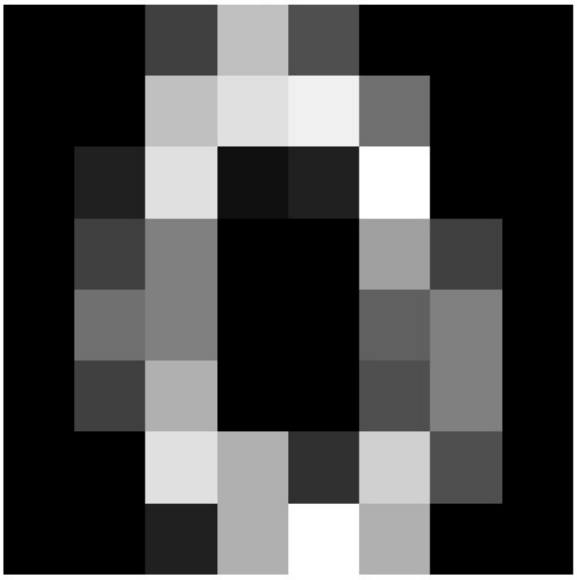
Actual: 0, Predicted: 0



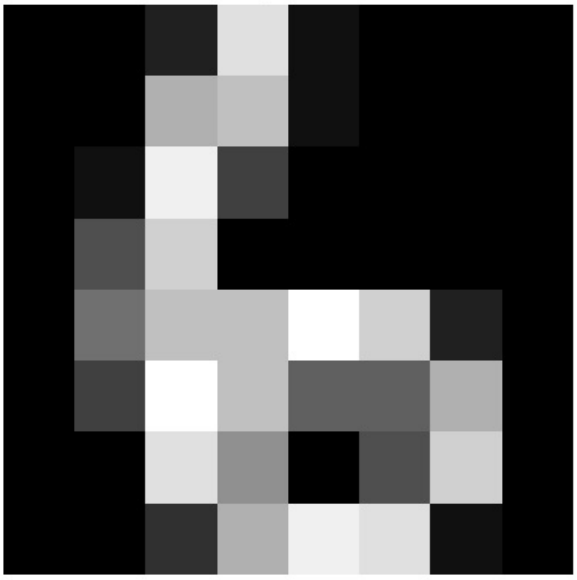
Actual: 1, Predicted: 1



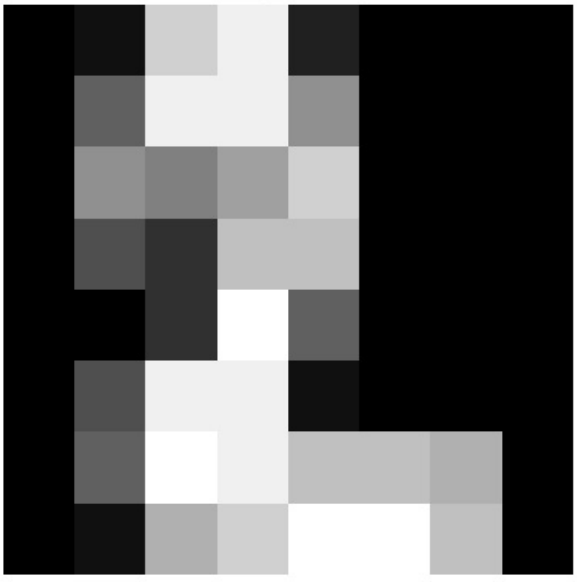
Actual: 0, Predicted: 0



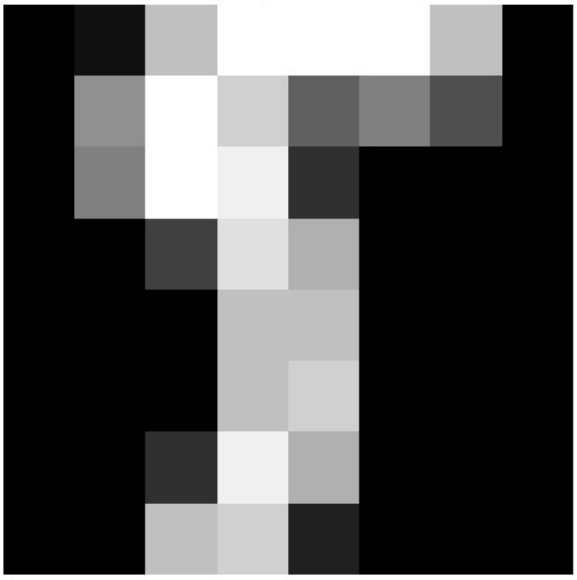
Actual: 6, Predicted: 6



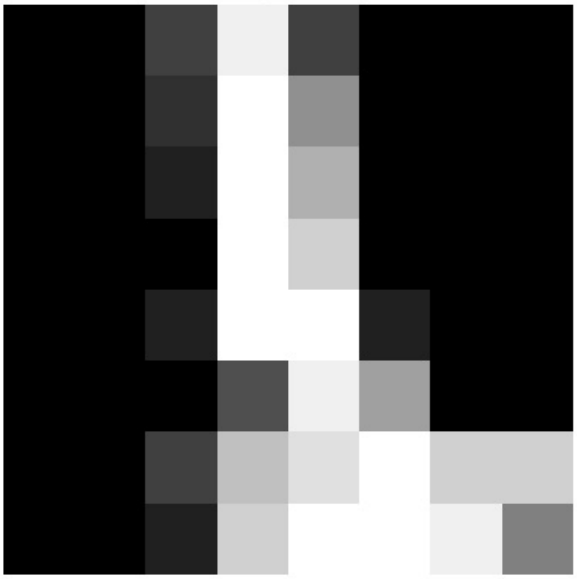
Actual: 2, Predicted: 2



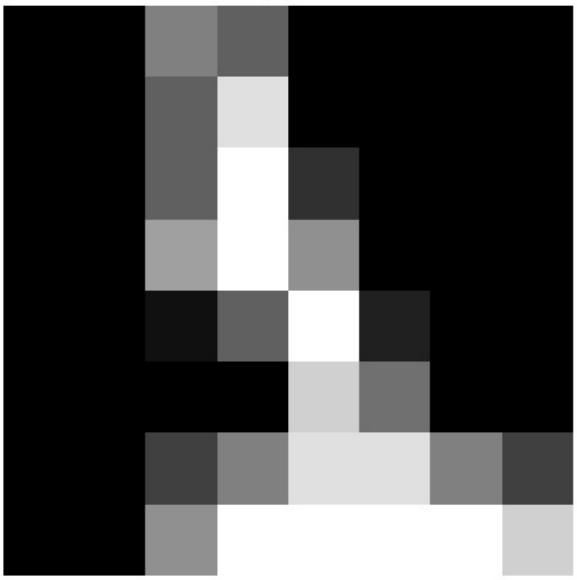
Actual: 5, Predicted: 5



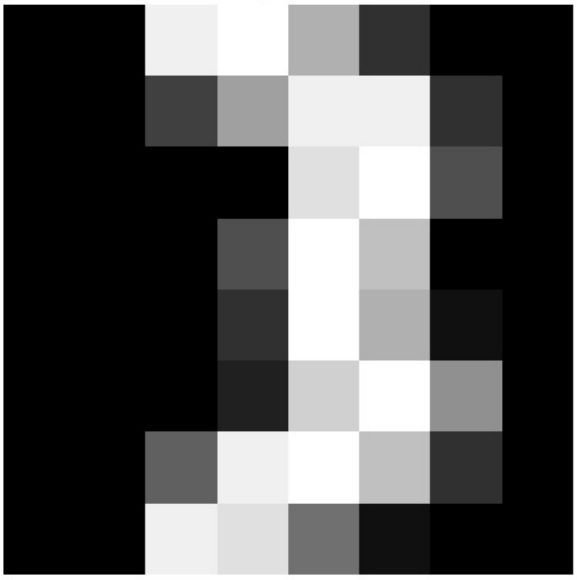
Actual: 1, Predicted: 1



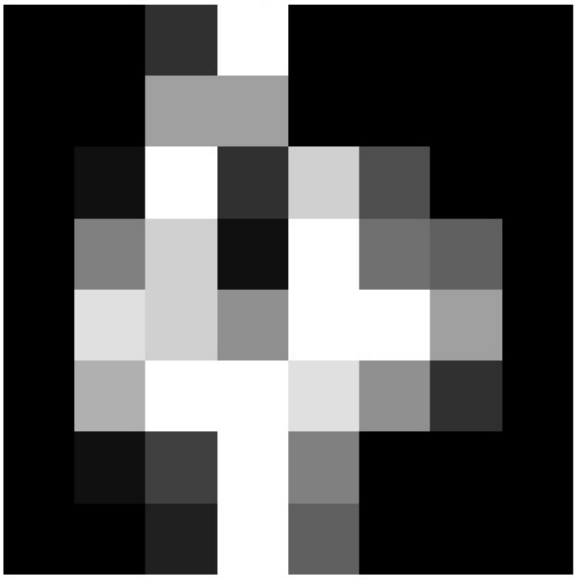
Actual: 1, Predicted: 1



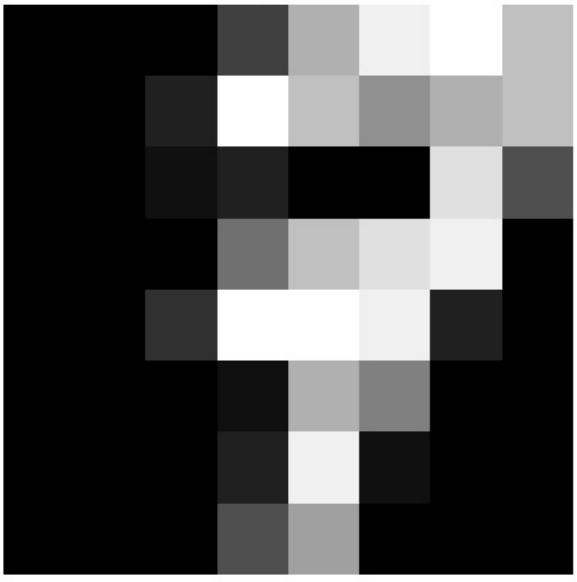
Actual: 3, Predicted: 3



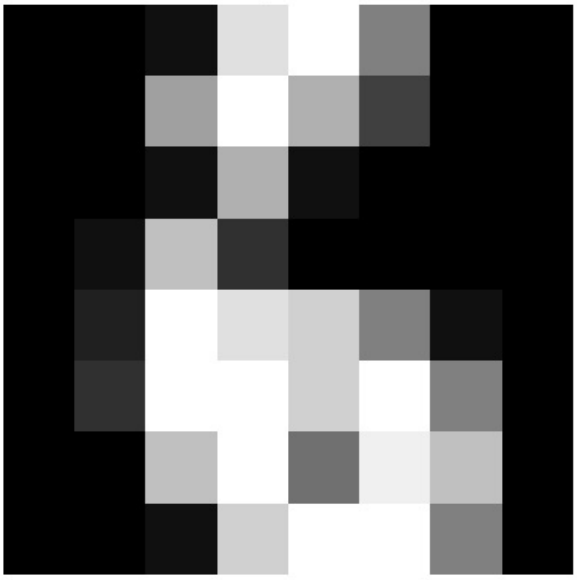
Actual: 4, Predicted: 4



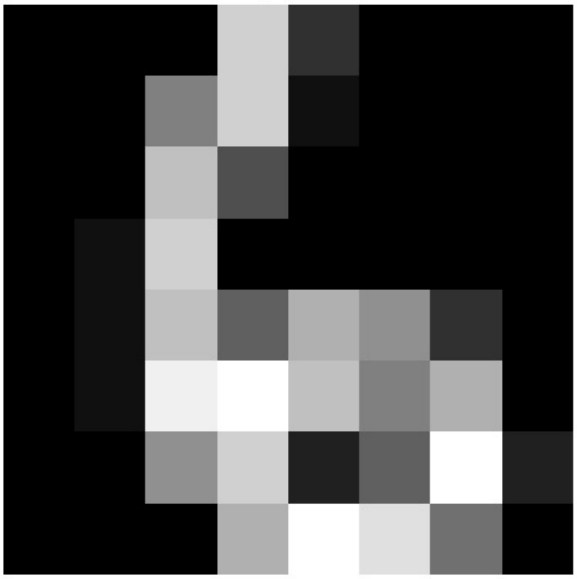
Actual: 7, Predicted: 7



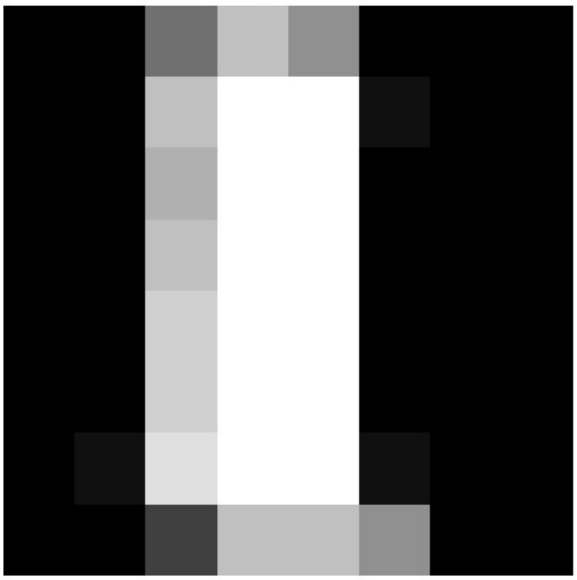
Actual: 6, Predicted: 6



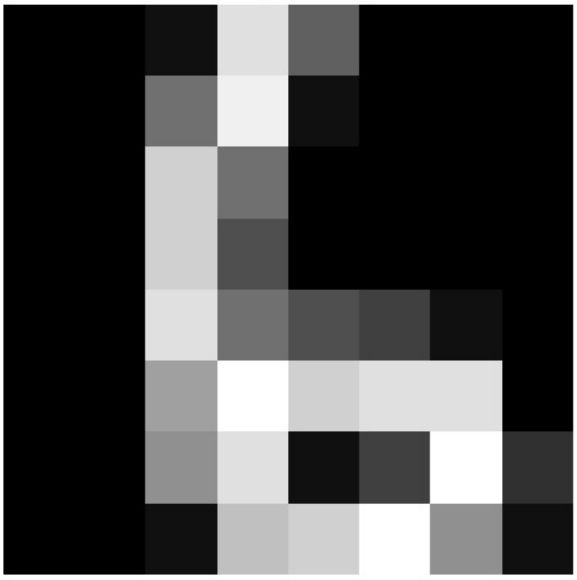
Actual: 6, Predicted: 6



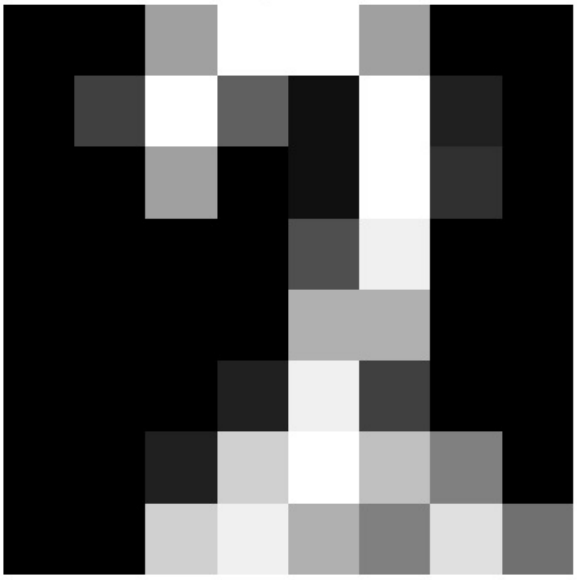
Actual: 1, Predicted: 1



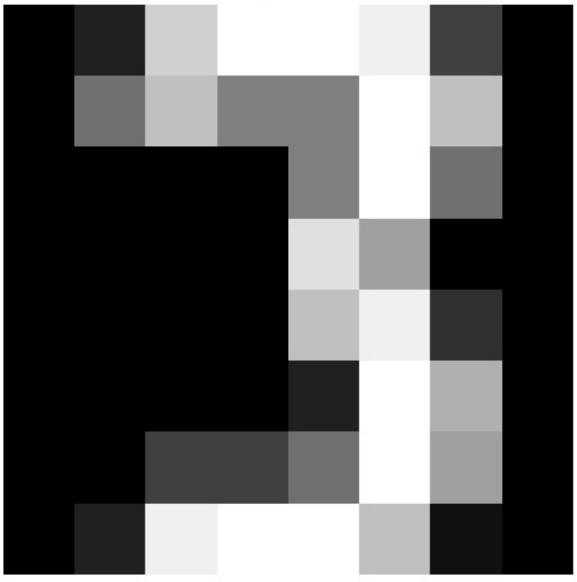
Actual: 6, Predicted: 6



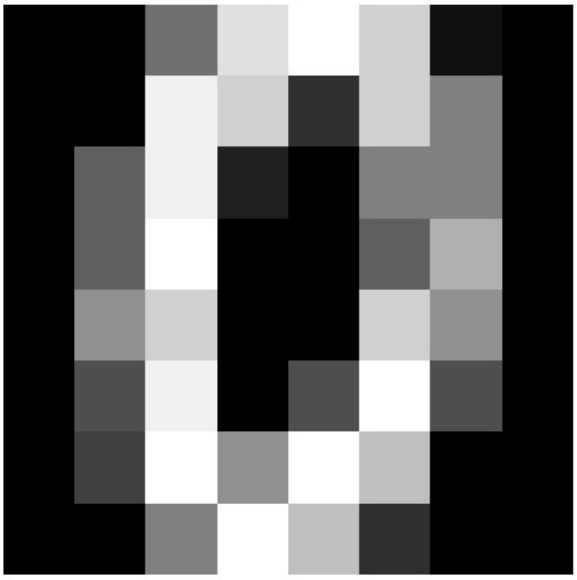
Actual: 2, Predicted: 2



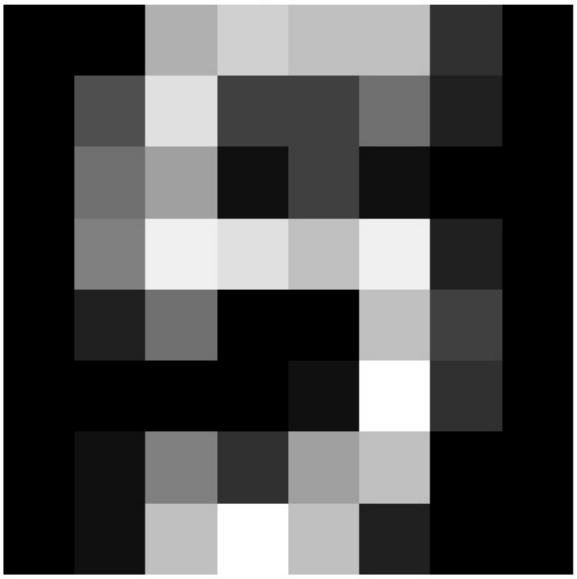
Actual: 3, Predicted: 3



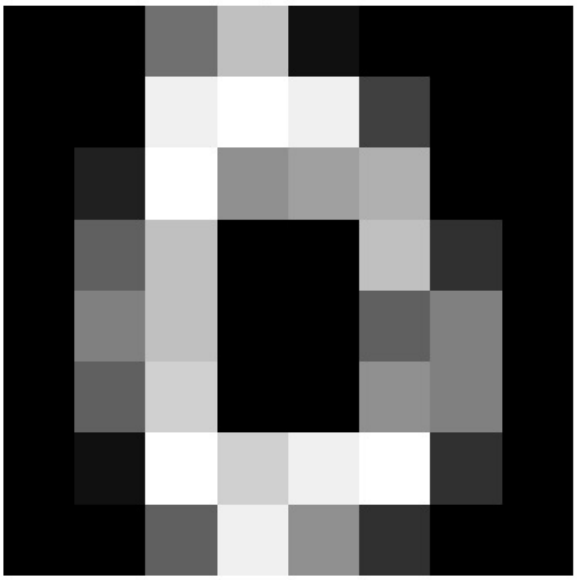
Actual: 0, Predicted: 0



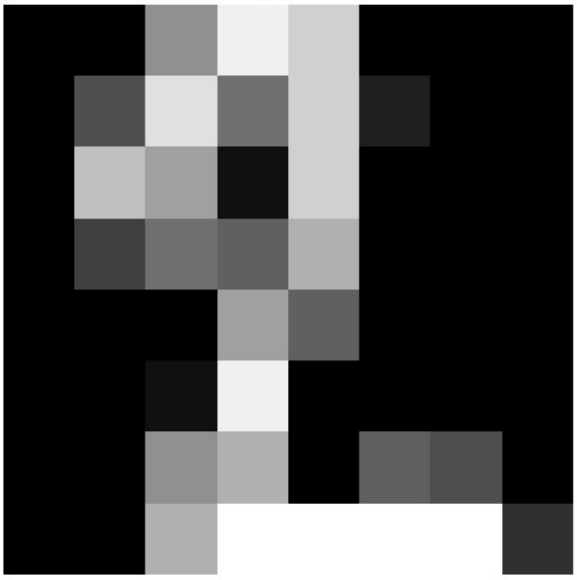
Actual: 5, Predicted: 5



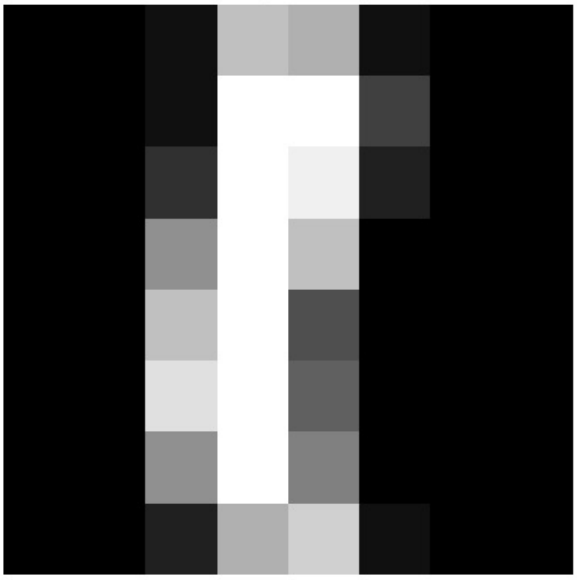
Actual: 0, Predicted: 0



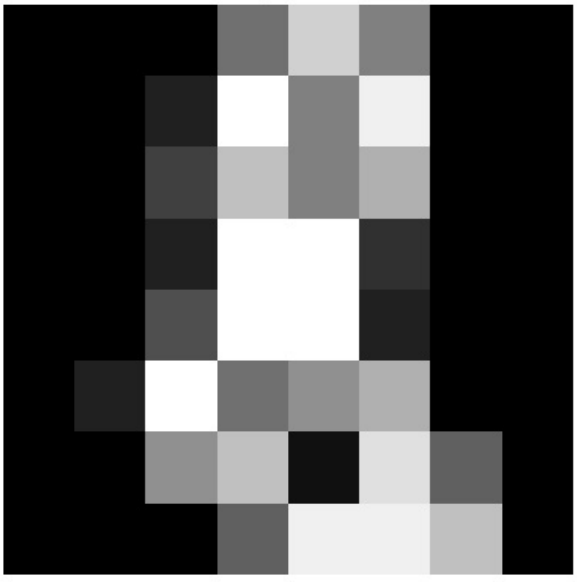
Actual: 2, Predicted: 2



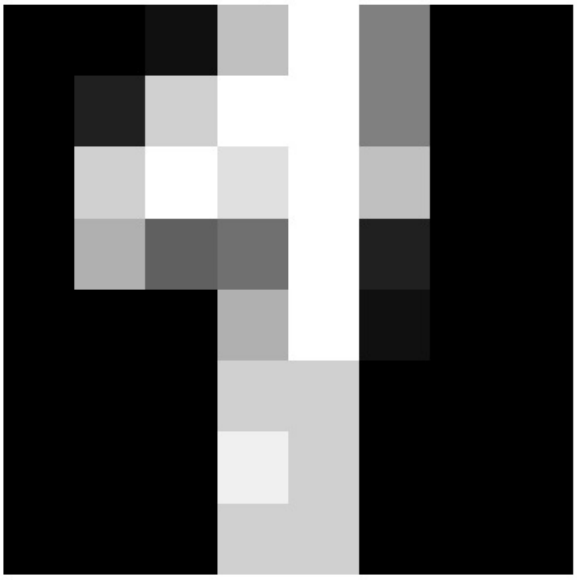
Actual: 1, Predicted: 1



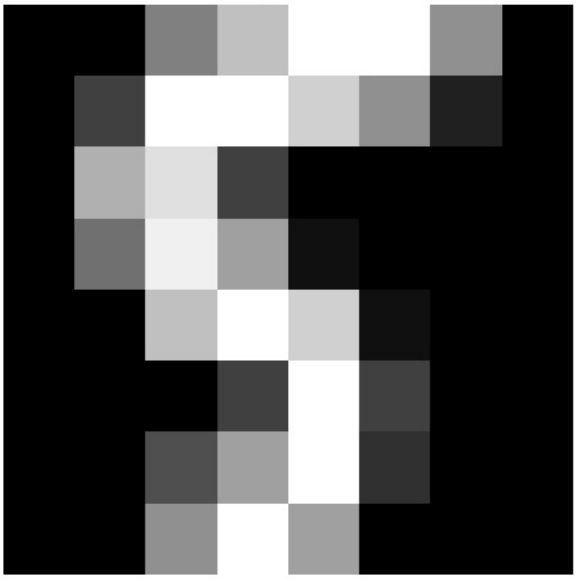
Actual: 8, Predicted: 8



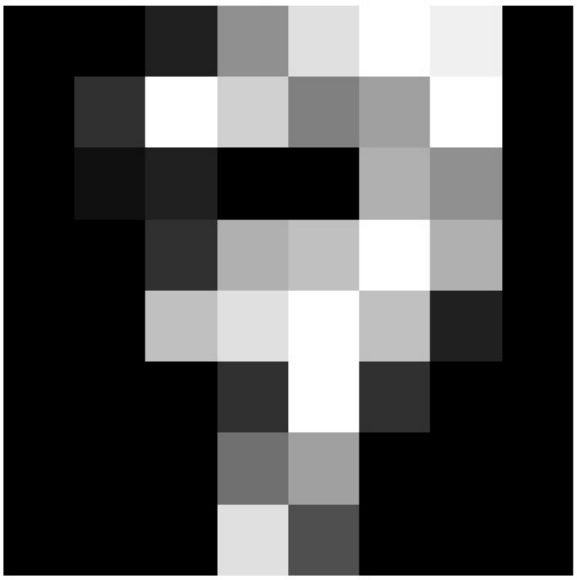
Actual: 1, Predicted: 1



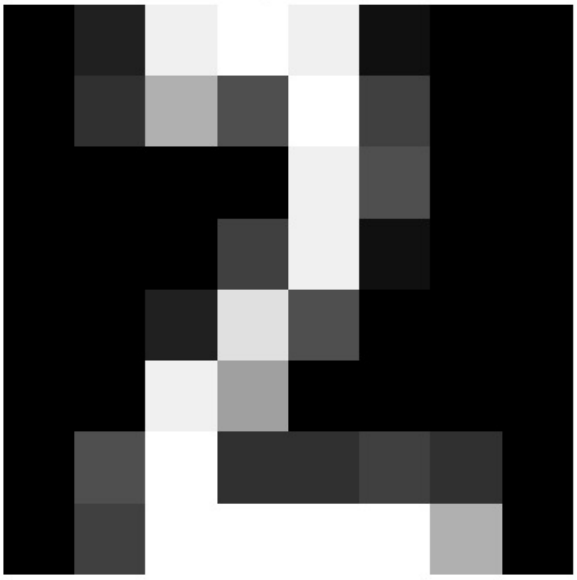
Actual: 5, Predicted: 5



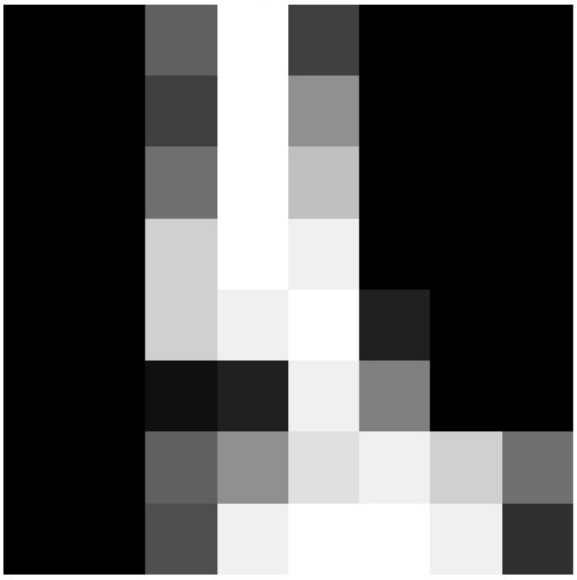
Actual: 7, Predicted: 7



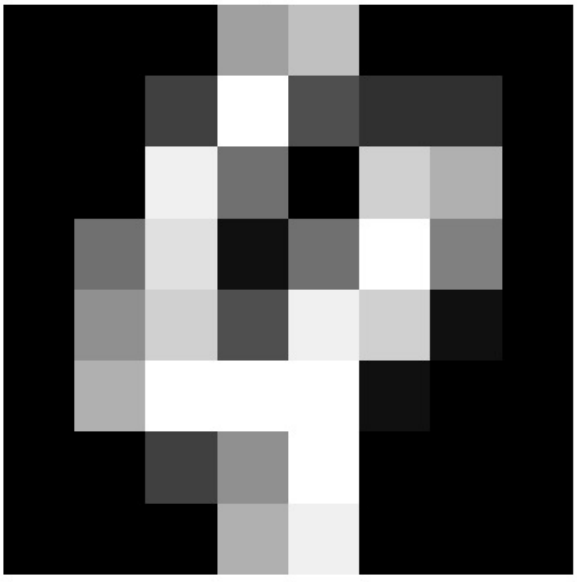
Actual: 2, Predicted: 2



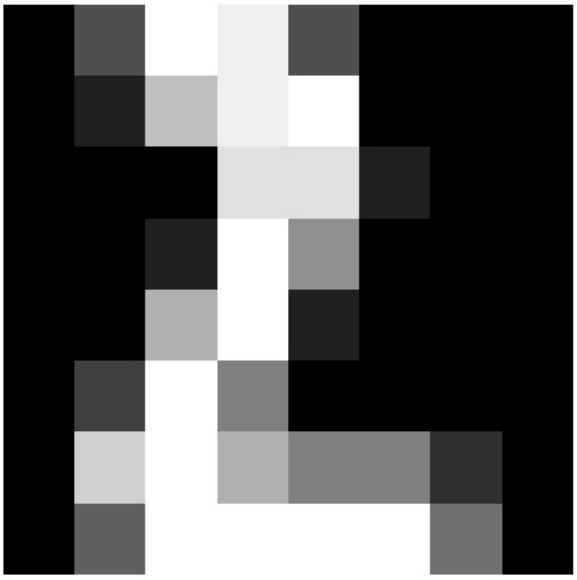
Actual: 1, Predicted: 1



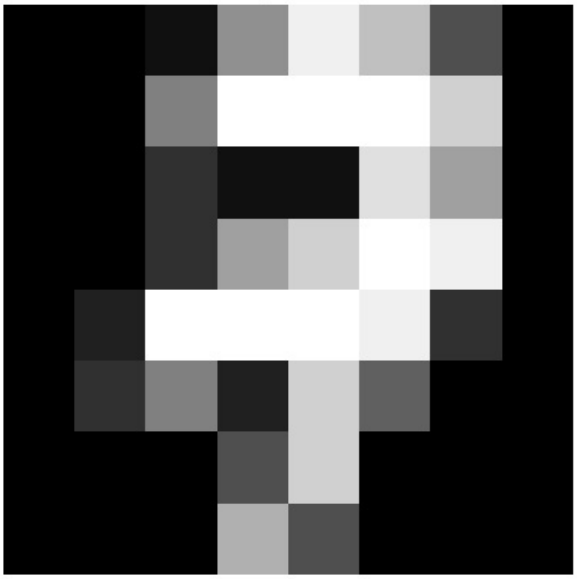
Actual: 4, Predicted: 4



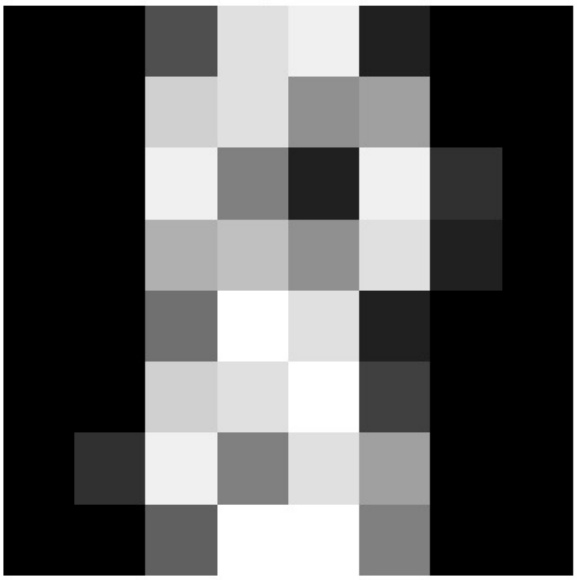
Actual: 2, Predicted: 2



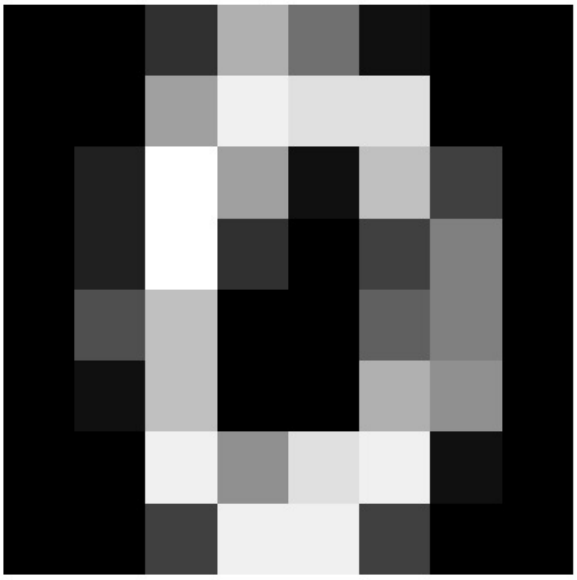
Actual: 7, Predicted: 7



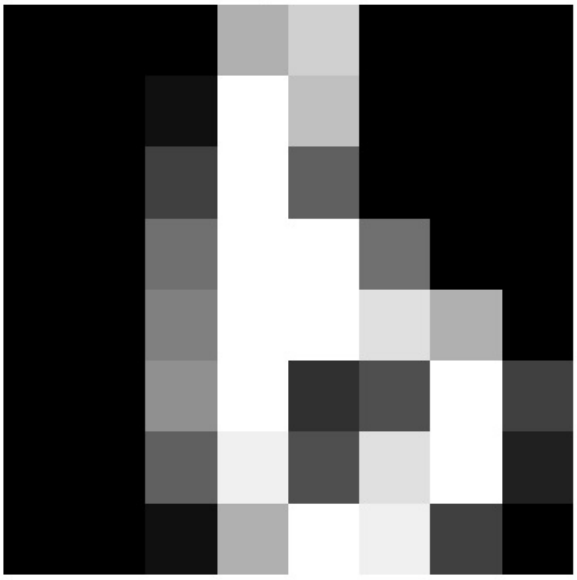
Actual: 8, Predicted: 8



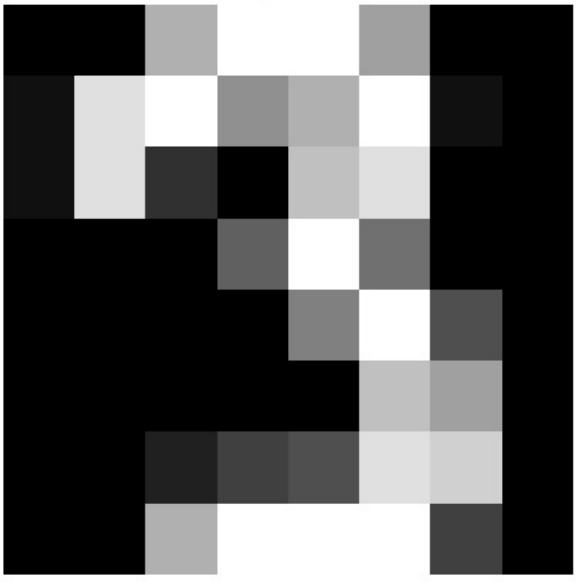
Actual: 0, Predicted: 0



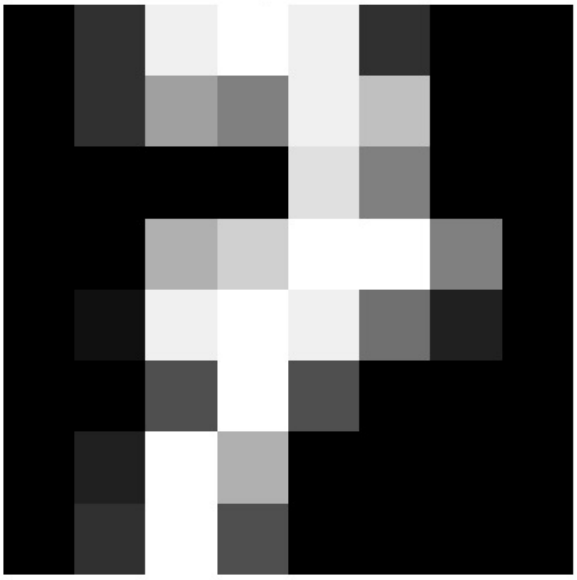
Actual: 6, Predicted: 6



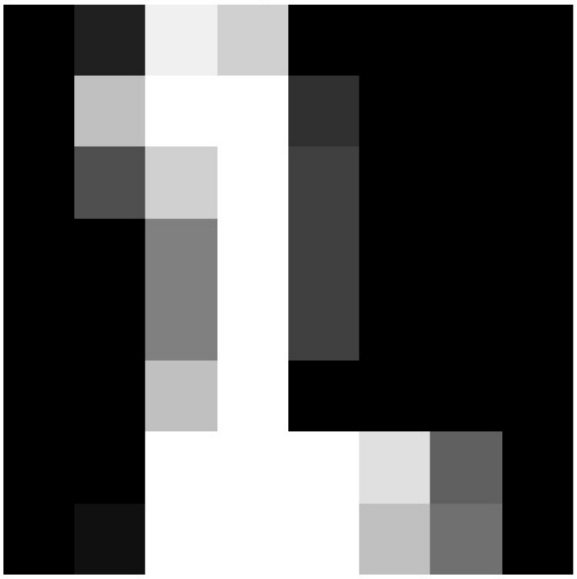
Actual: 3, Predicted: 3



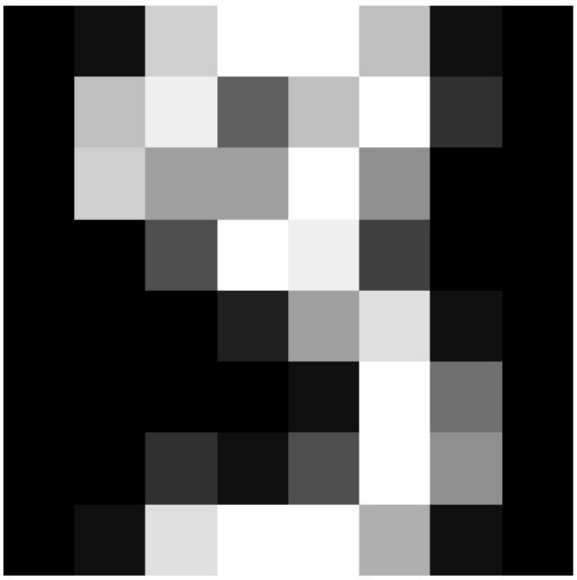
Actual: 7, Predicted: 7



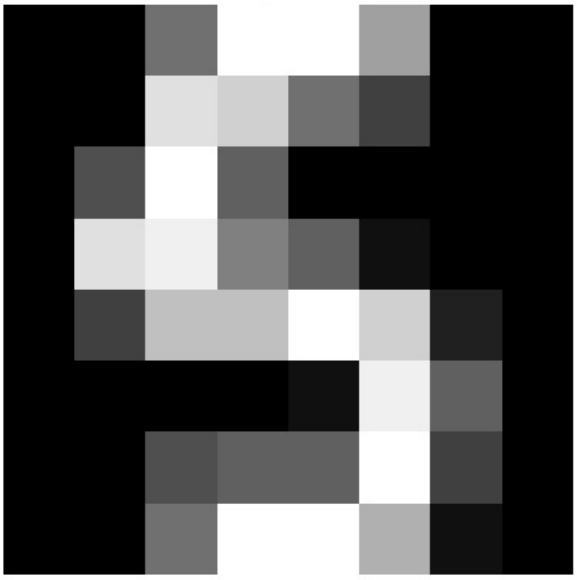
Actual: 2, Predicted: 2



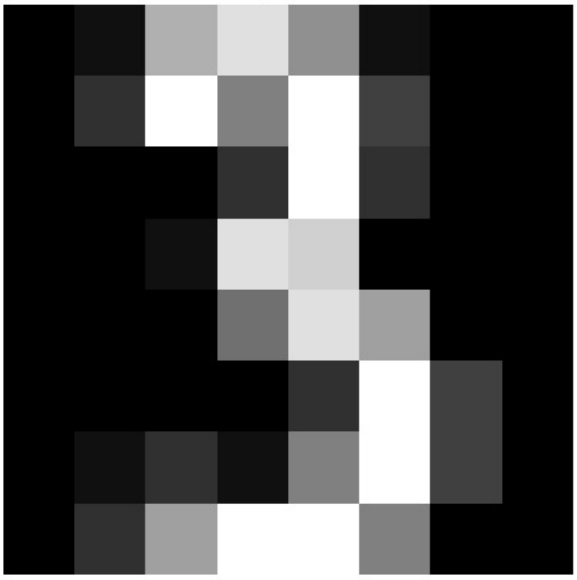
Actual: 3, Predicted: 3



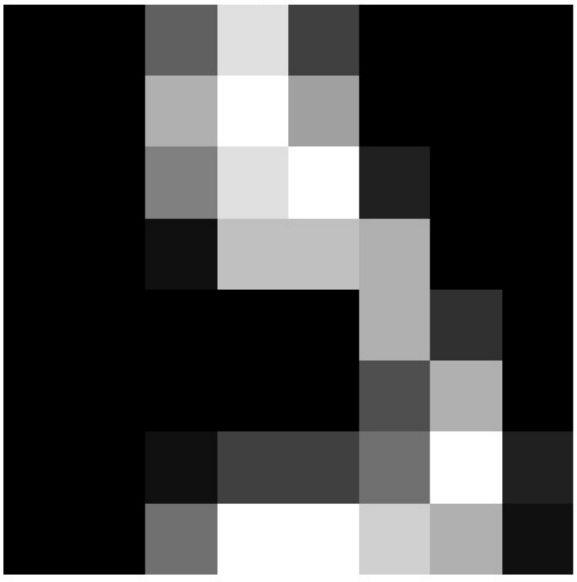
Actual: 5, Predicted: 5



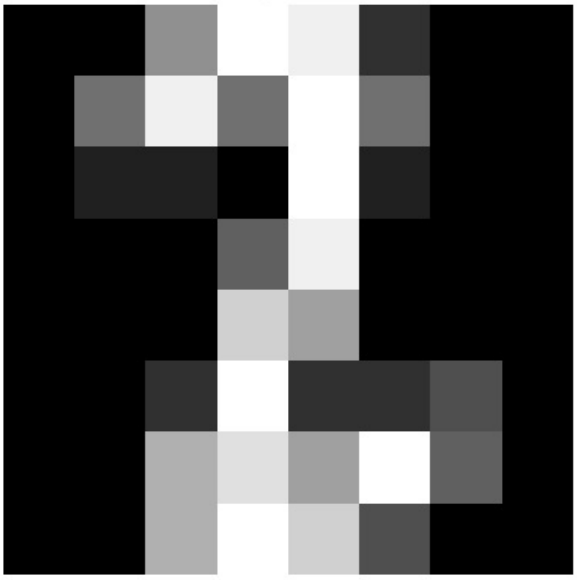
Actual: 3, Predicted: 3



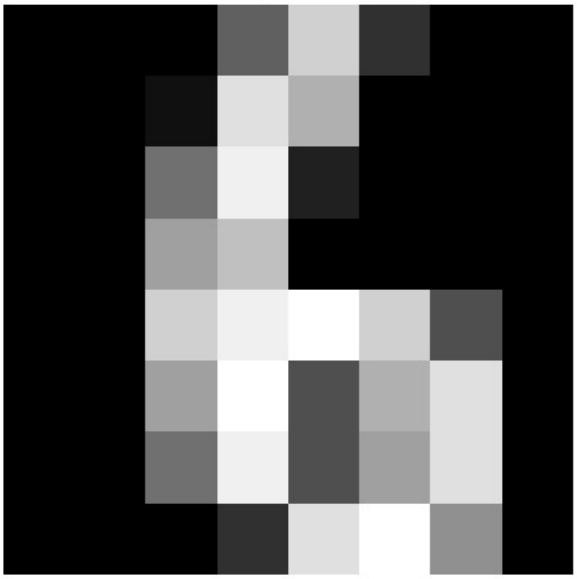
Actual: 9, Predicted: 9



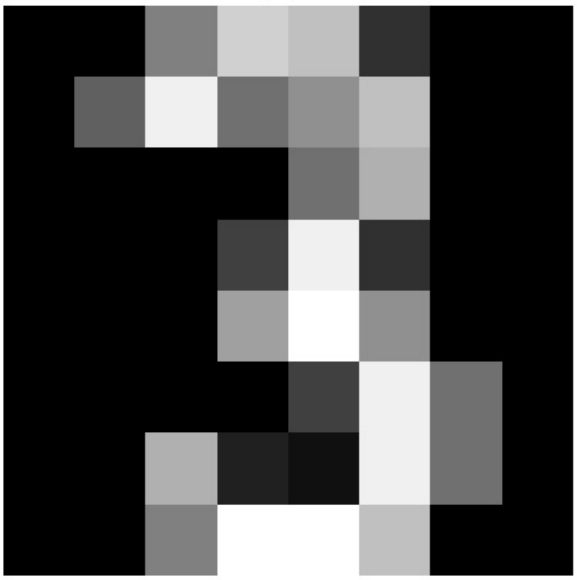
Actual: 2, Predicted: 2



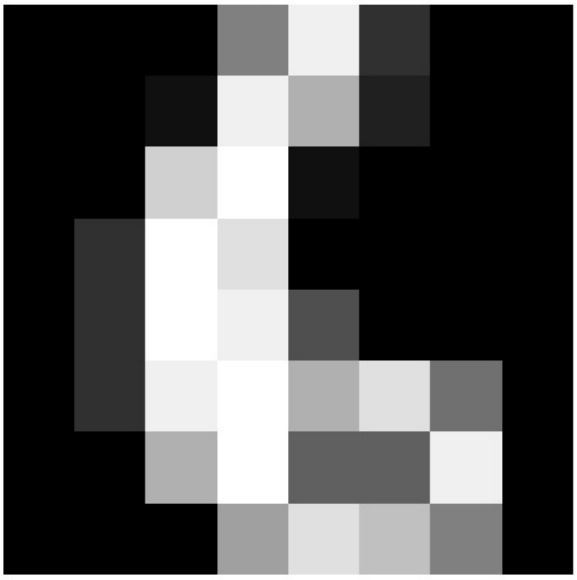
Actual: 6, Predicted: 6



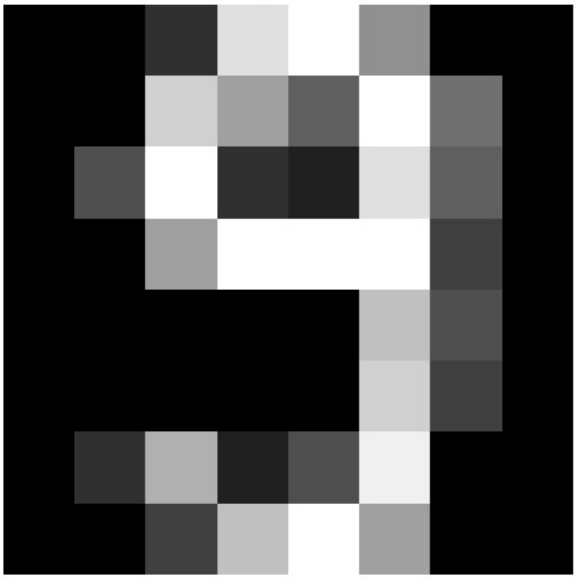
Actual: 3, Predicted: 3



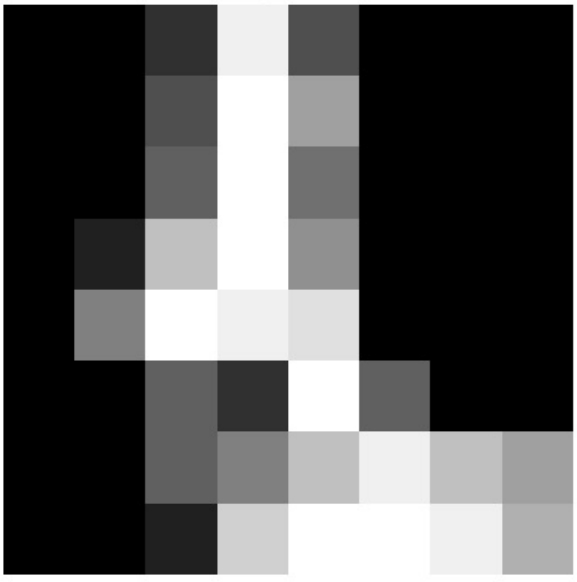
Actual: 6, Predicted: 6



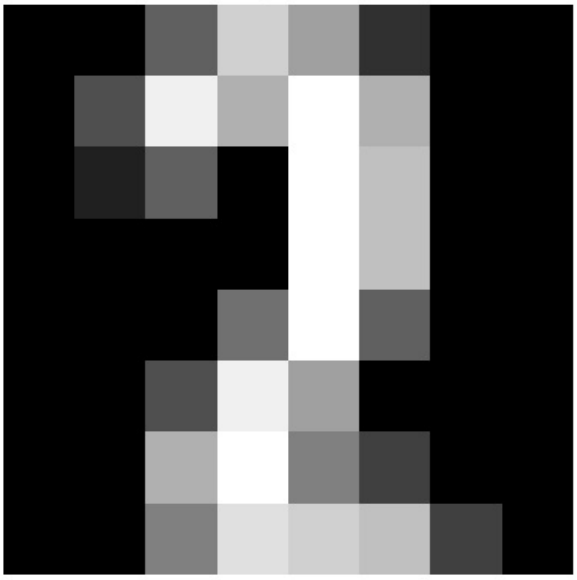
Actual: 9, Predicted: 9



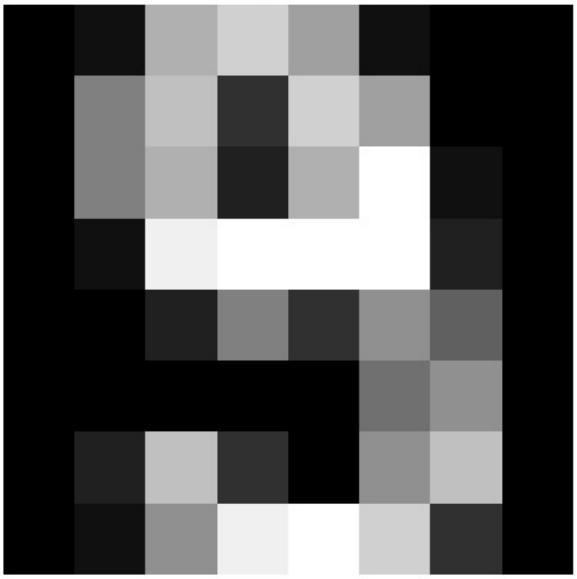
Actual: 1, Predicted: 1



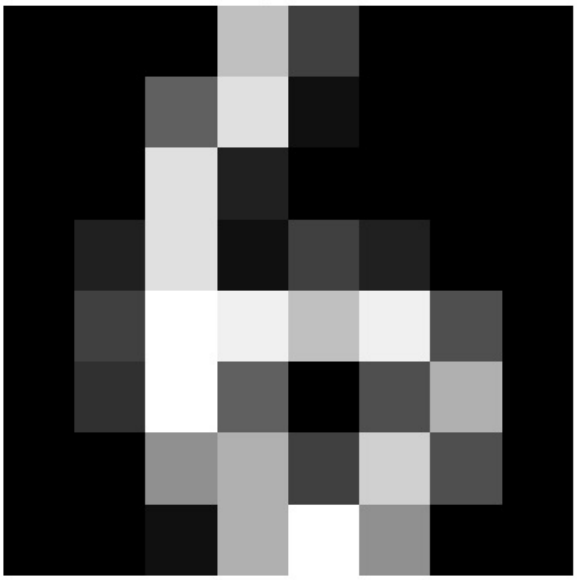
Actual: 2, Predicted: 2



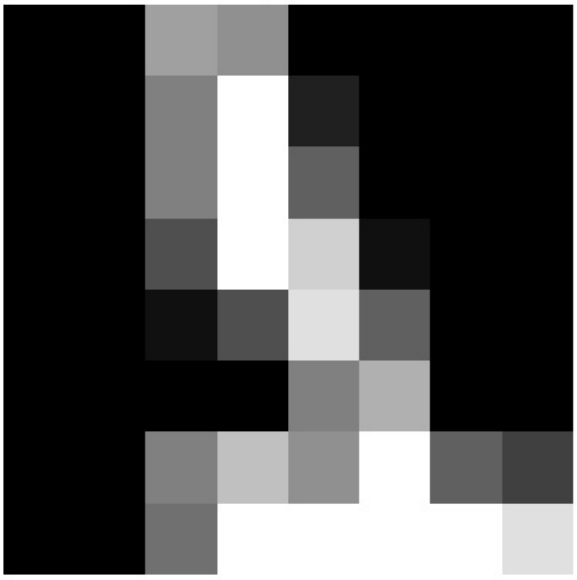
Actual: 9, Predicted: 9



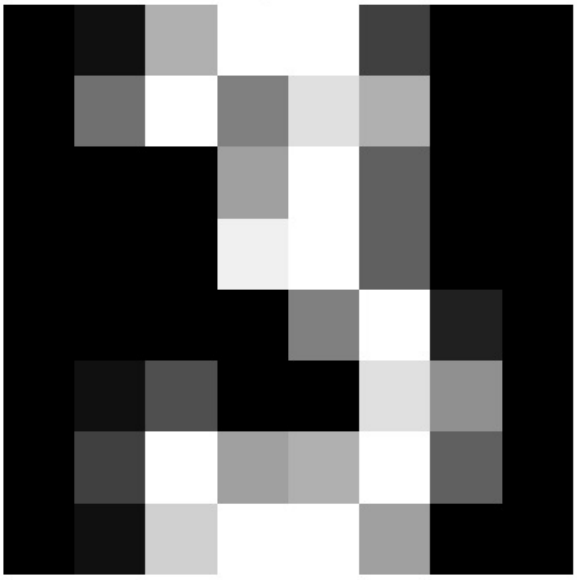
Actual: 6, Predicted: 6



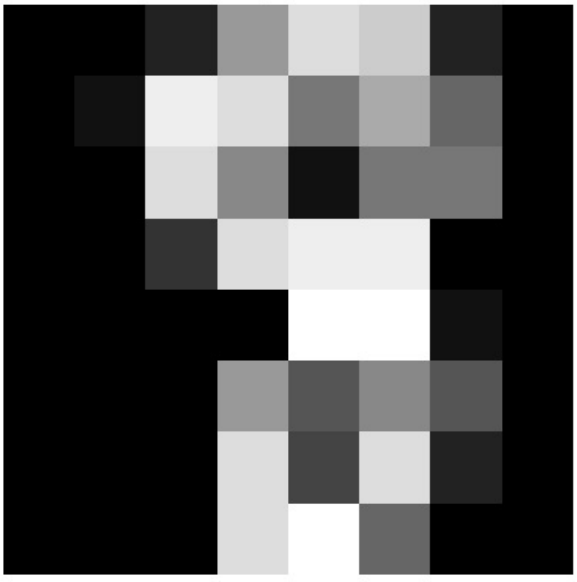
Actual: 1, Predicted: 1



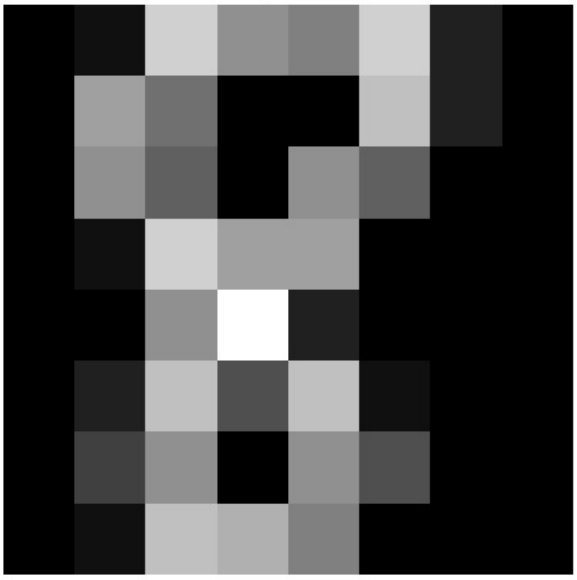
Actual: 3, Predicted: 3



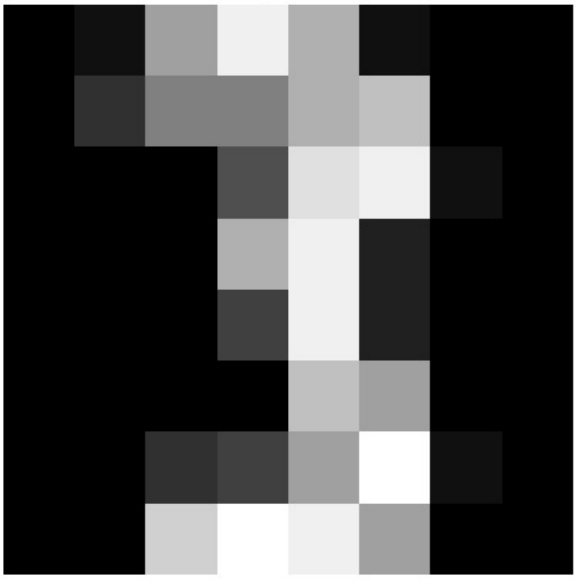
Actual: 8, Predicted: 8



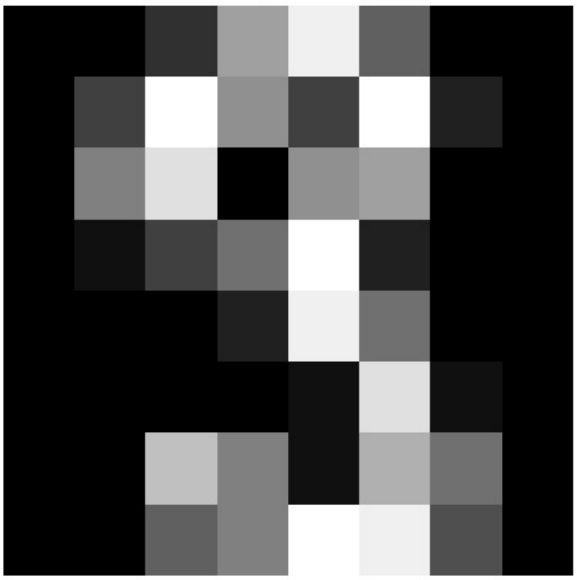
Actual: 8, Predicted: 8



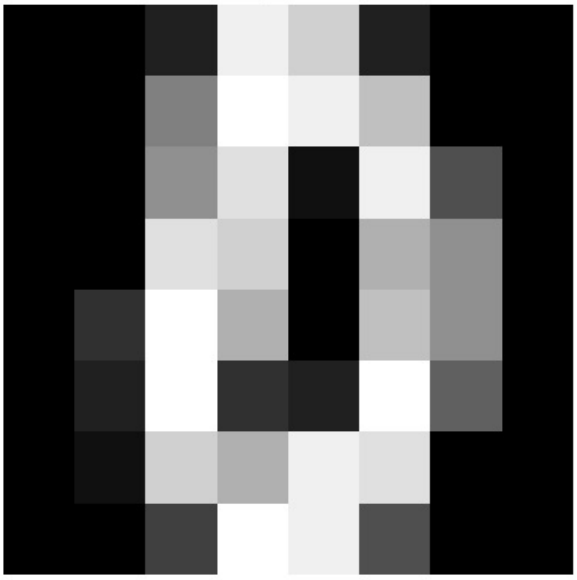
Actual: 3, Predicted: 3



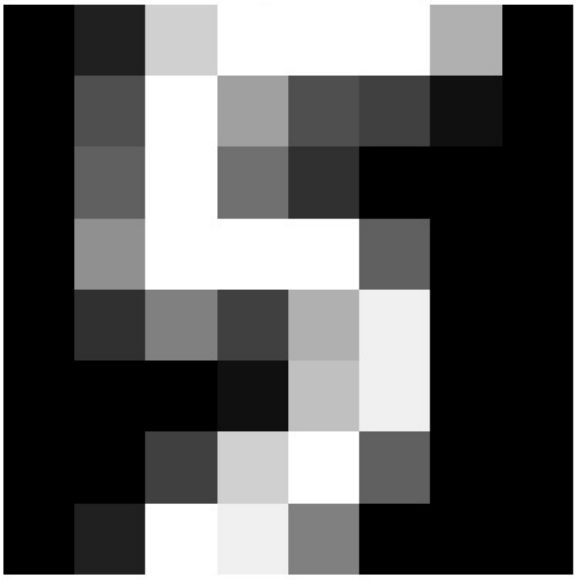
Actual: 3, Predicted: 3



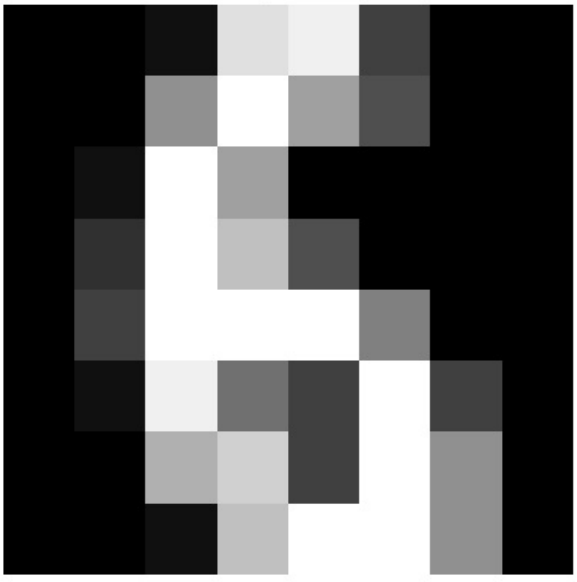
Actual: 0, Predicted: 0



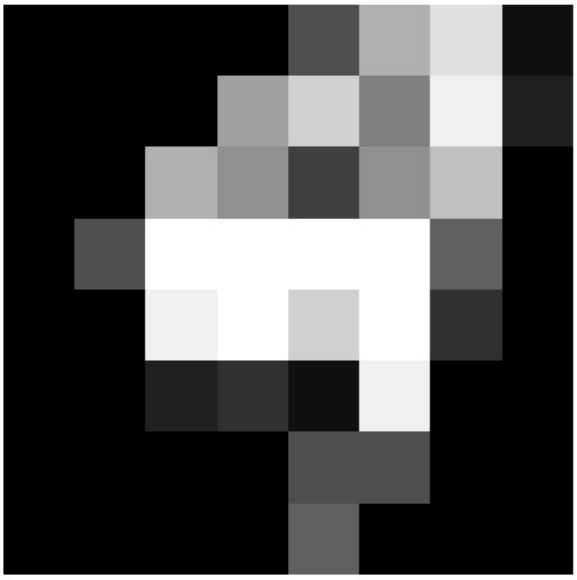
Actual: 5, Predicted: 5



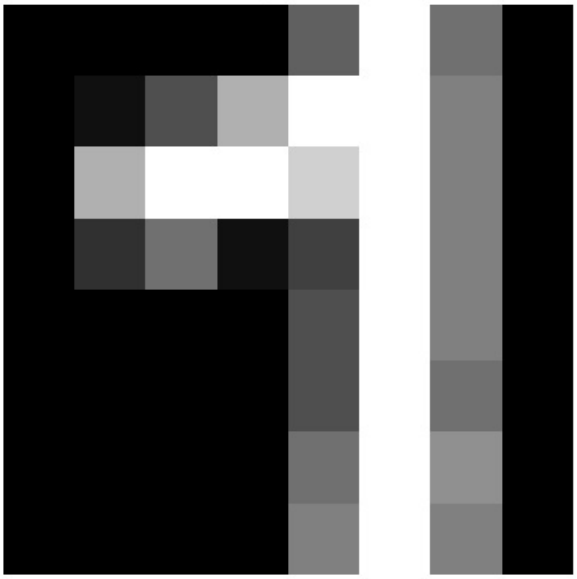
Actual: 6, Predicted: 6



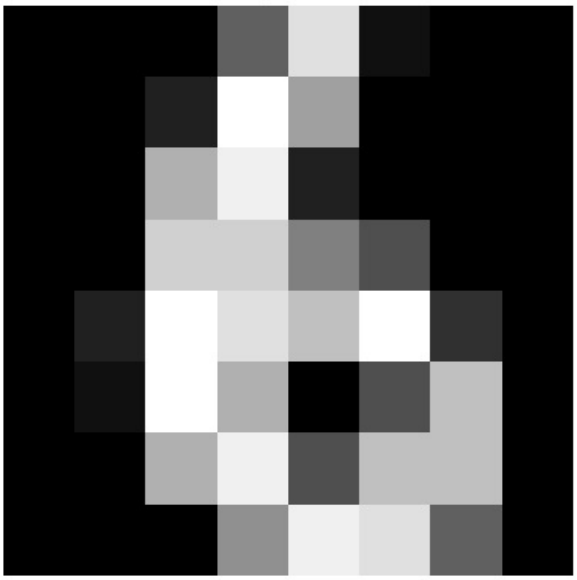
Actual: 9, Predicted: 9



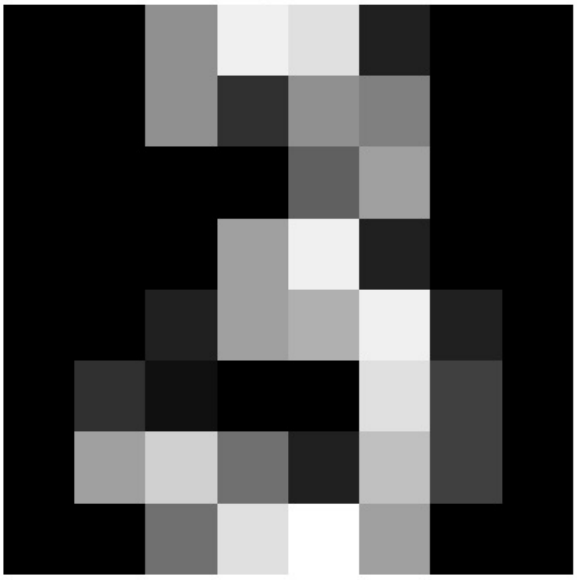
Actual: 1, Predicted: 1



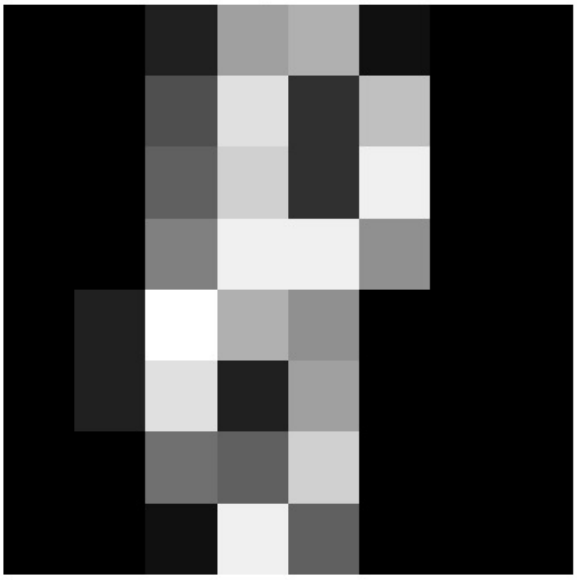
Actual: 6, Predicted: 6



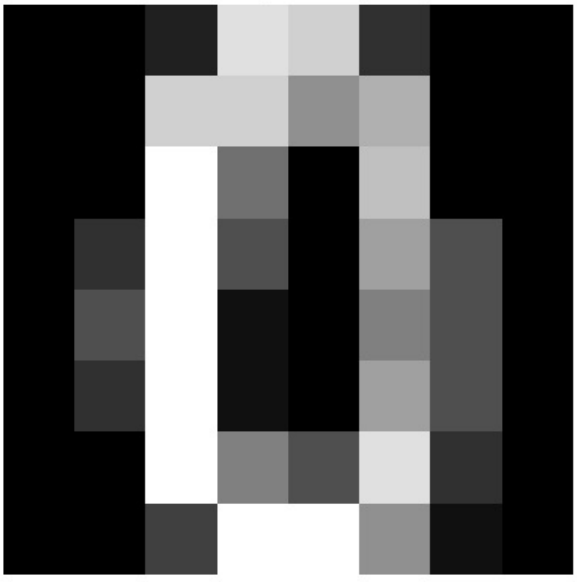
Actual: 3, Predicted: 3



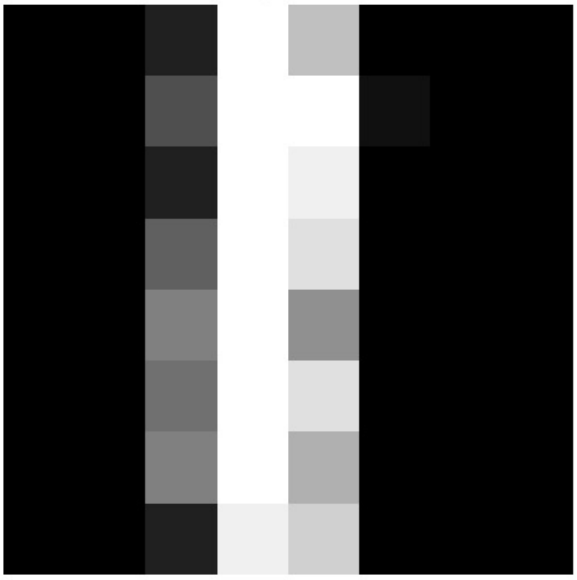
Actual: 8, Predicted: 8



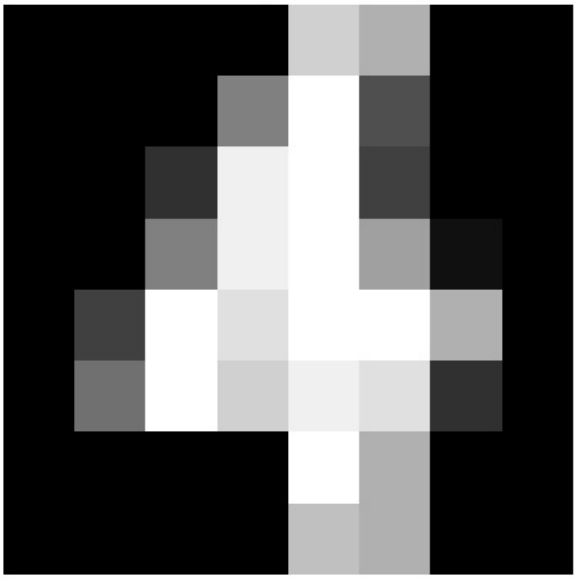
Actual: 0, Predicted: 0



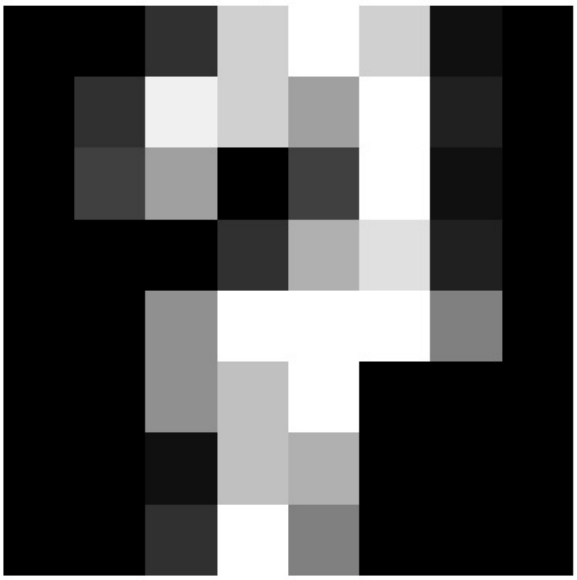
Actual: 1, Predicted: 1



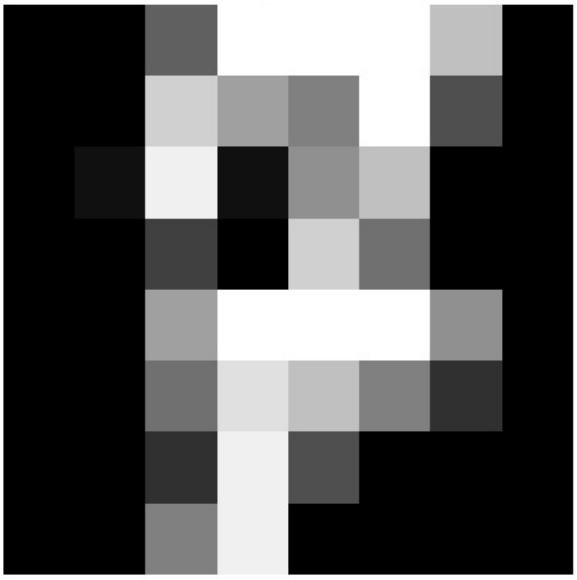
Actual: 4, Predicted: 4



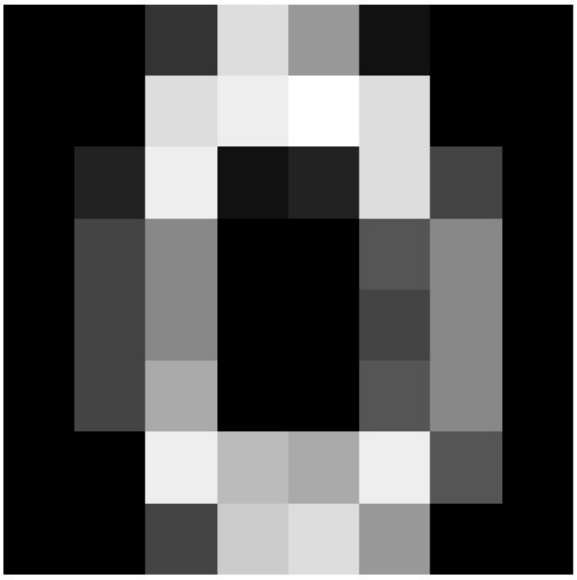
Actual: 7, Predicted: 7



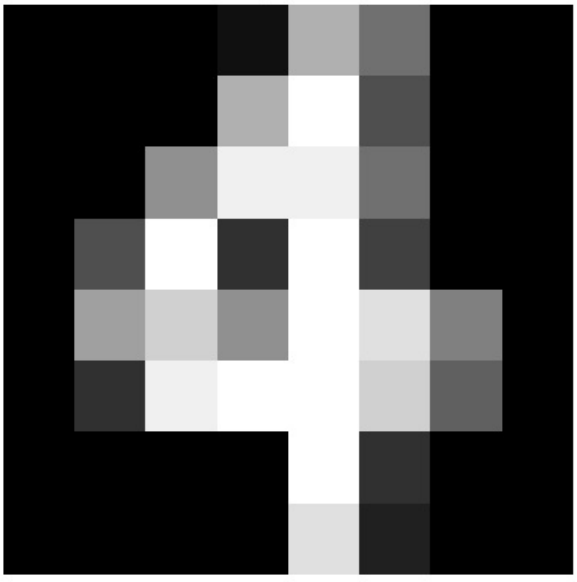
Actual: 7, Predicted: 7



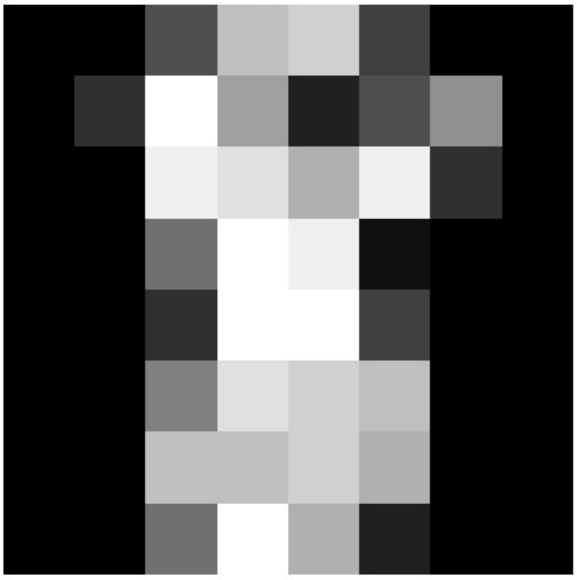
Actual: 0, Predicted: 0



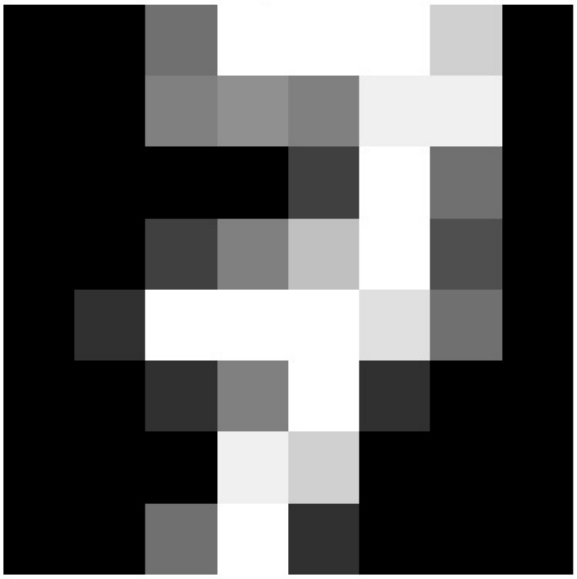
Actual: 4, Predicted: 4



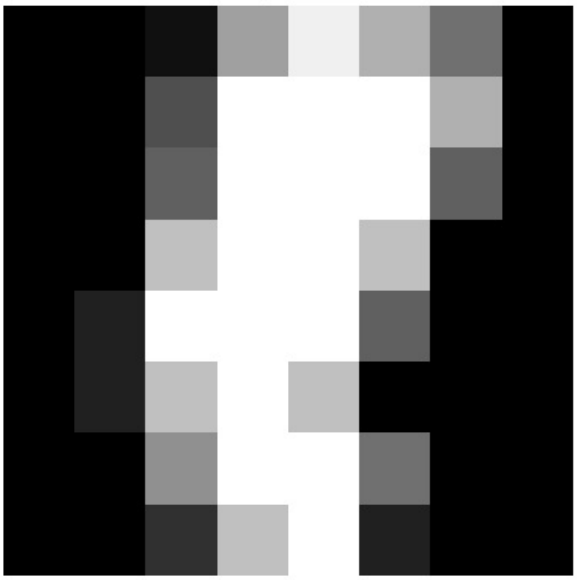
Actual: 8, Predicted: 8



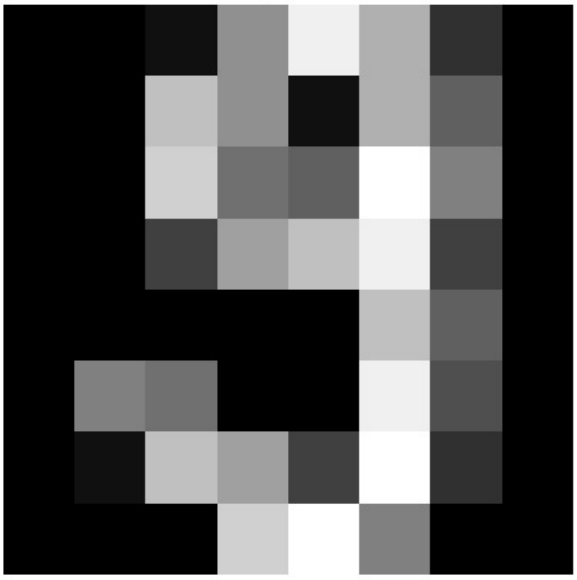
Actual: 7, Predicted: 7



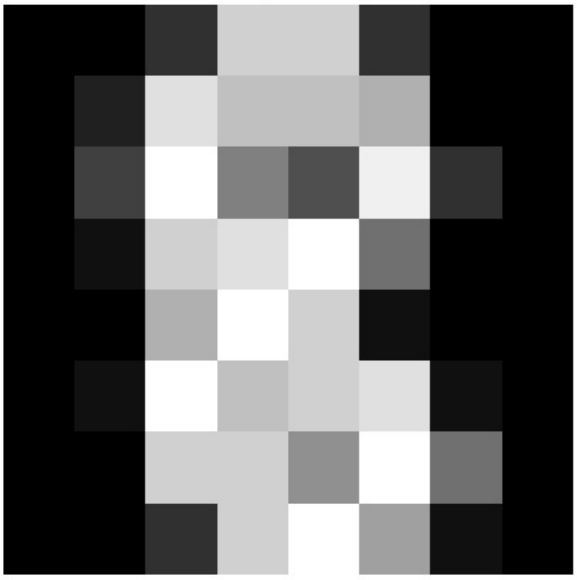
Actual: 1, Predicted: 1



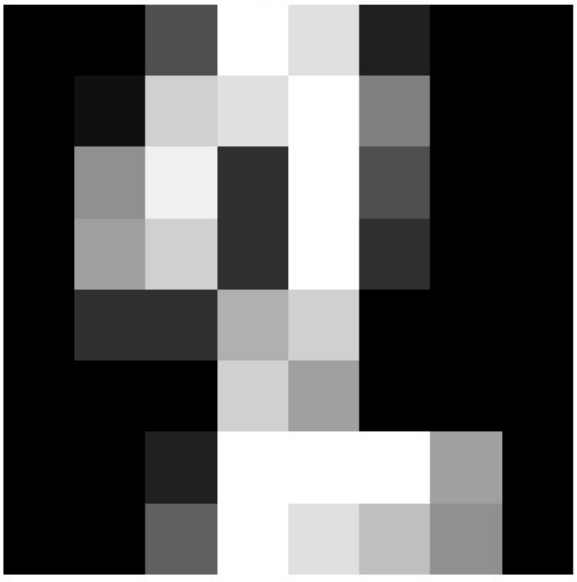
Actual: 9, Predicted: 9



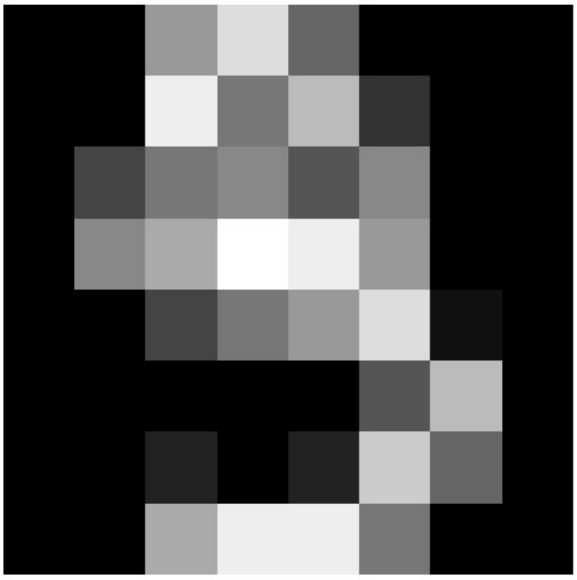
Actual: 8, Predicted: 8



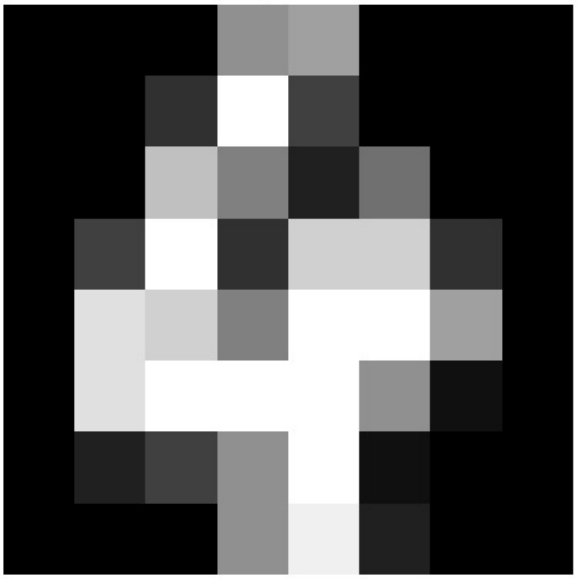
Actual: 2, Predicted: 2



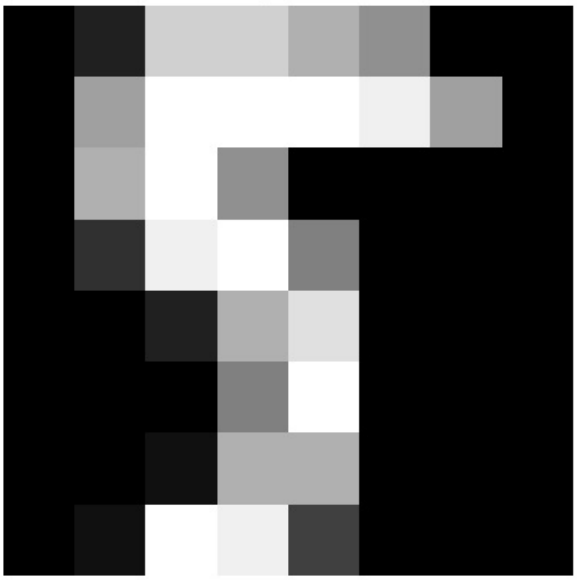
Actual: 9, Predicted: 9



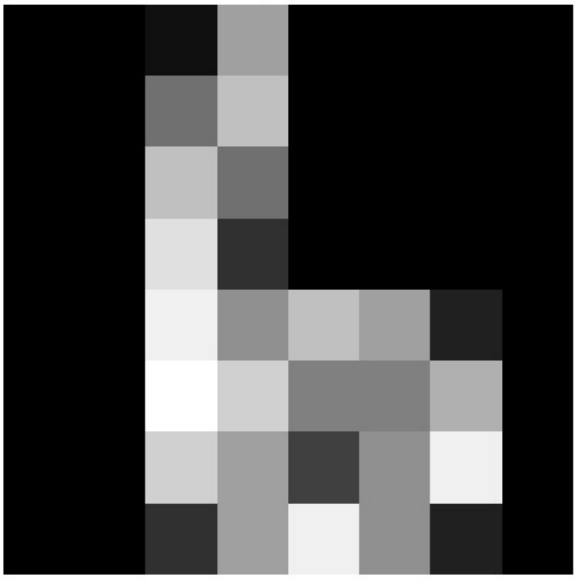
Actual: 4, Predicted: 4



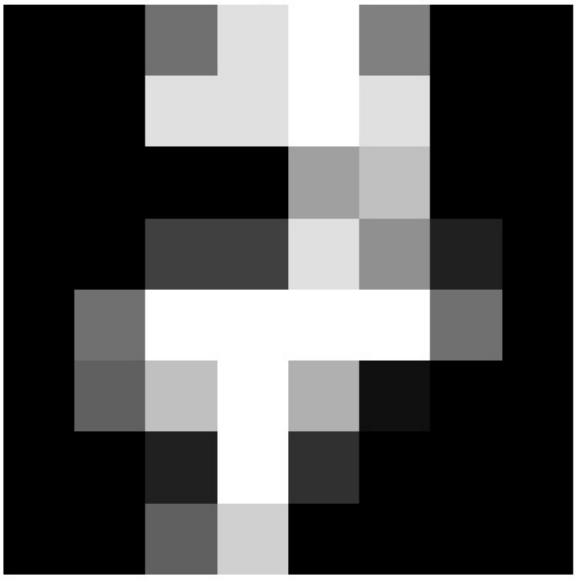
Actual: 5, Predicted: 5



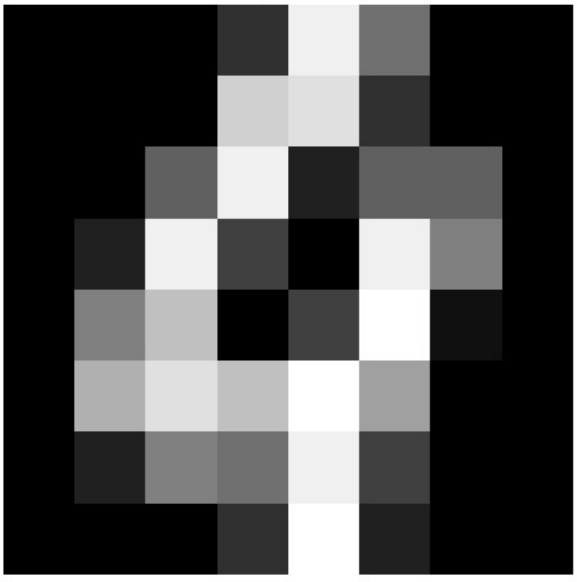
Actual: 6, Predicted: 6



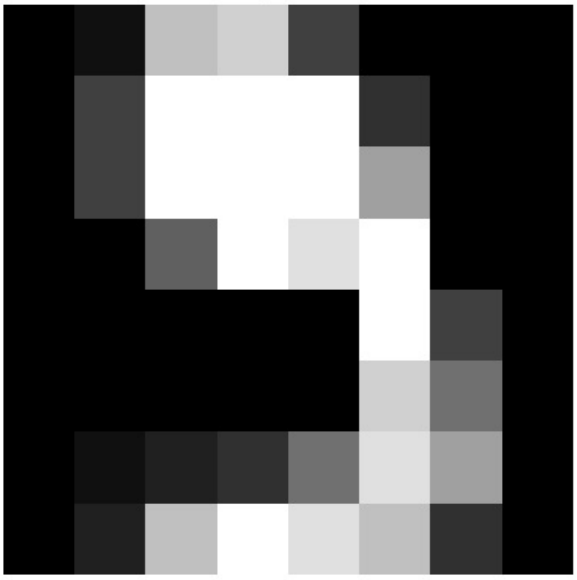
Actual: 7, Predicted: 7



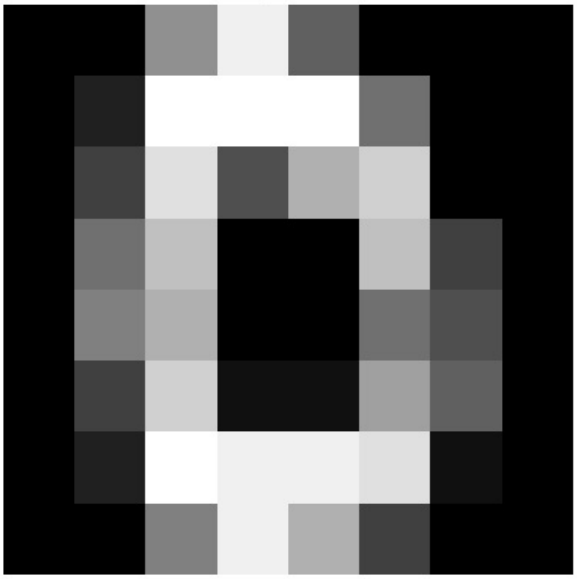
Actual: 4, Predicted: 4



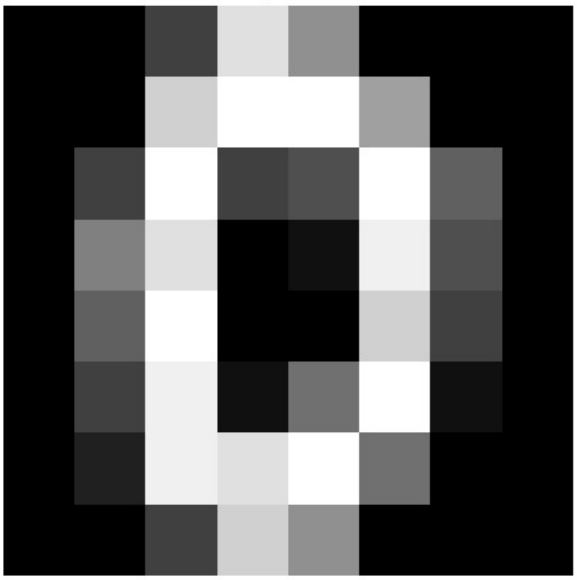
Actual: 9, Predicted: 9



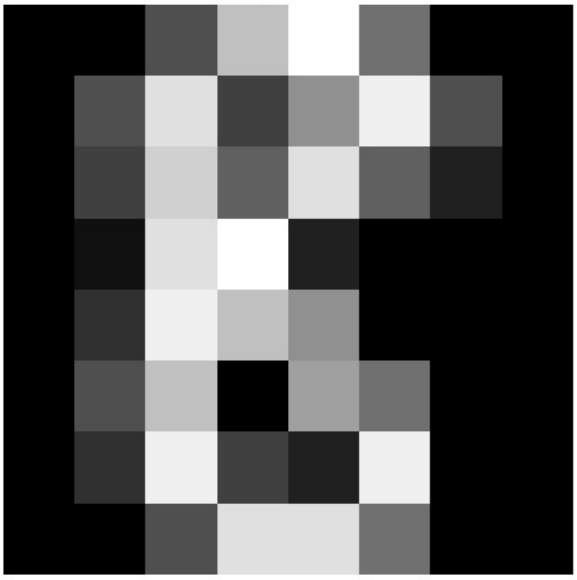
Actual: 0, Predicted: 0



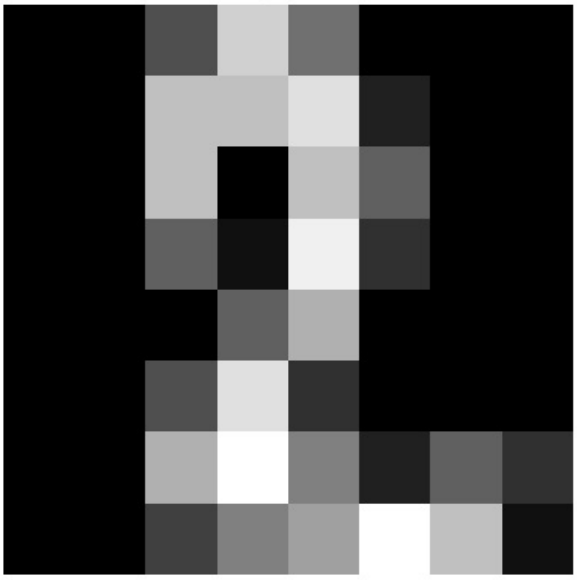
Actual: 0, Predicted: 0



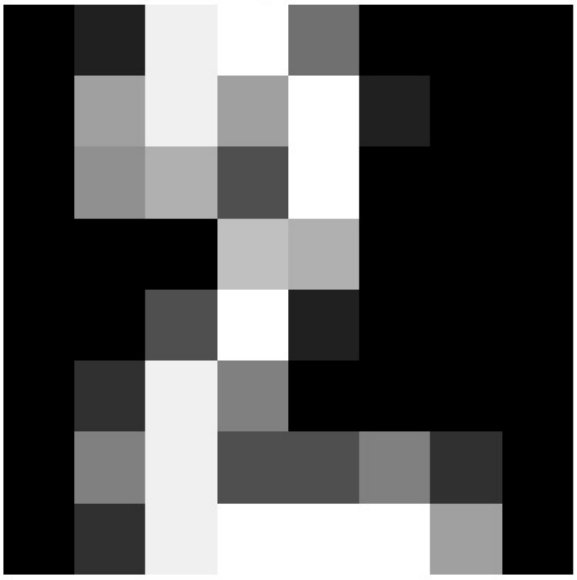
Actual: 8, Predicted: 8



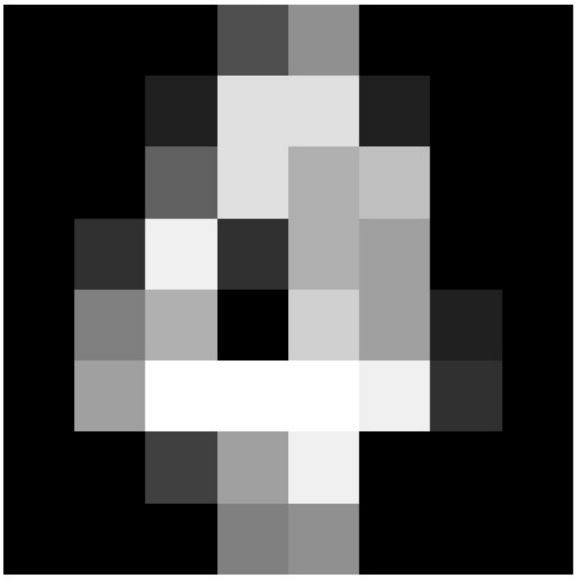
Actual: 2, Predicted: 2



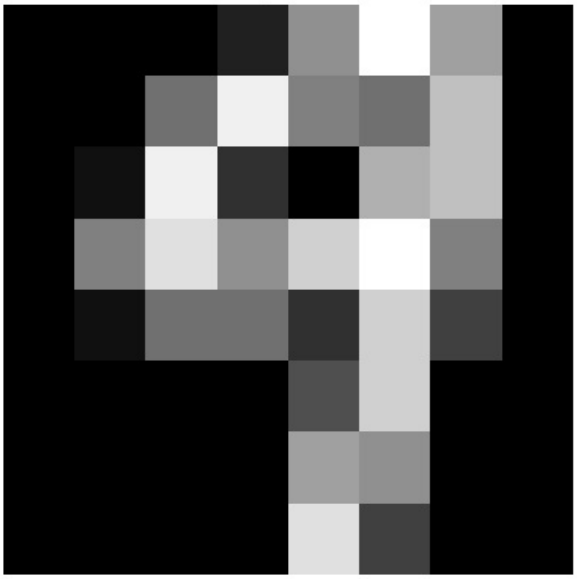
Actual: 2, Predicted: 2



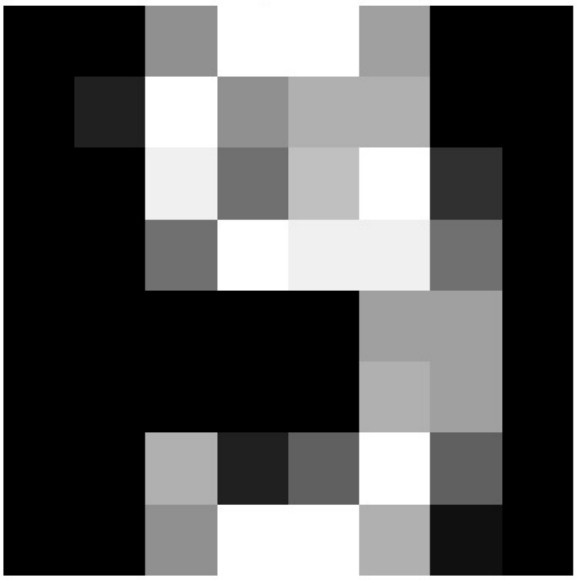
Actual: 4, Predicted: 4



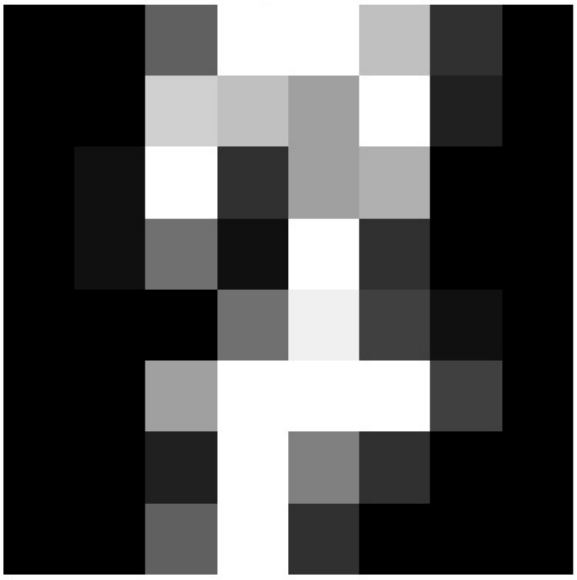
Actual: 9, Predicted: 9



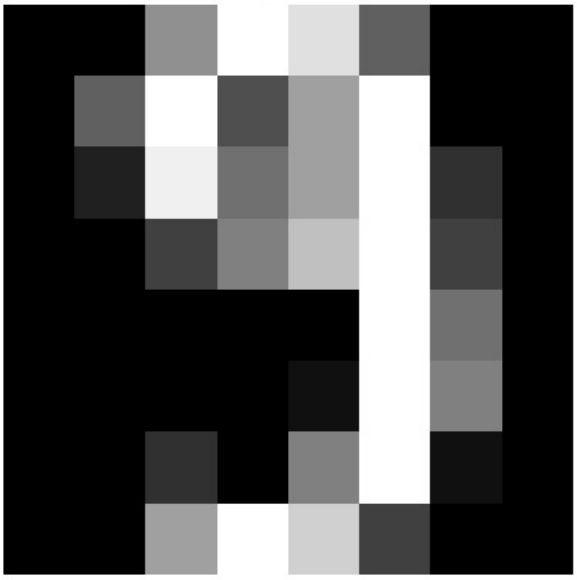
Actual: 9, Predicted: 9



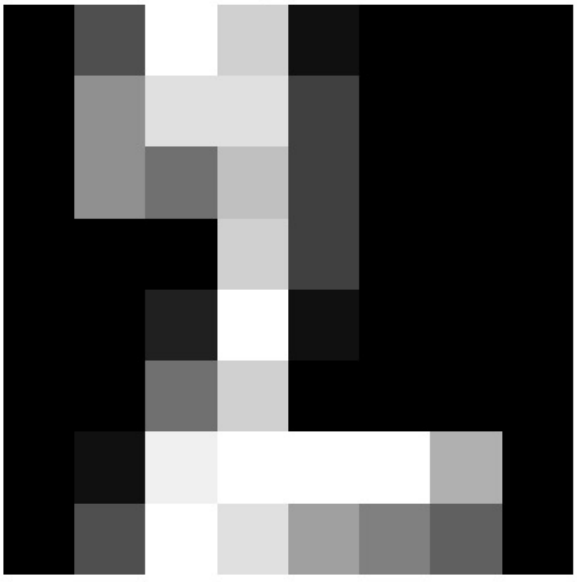
Actual: 7, Predicted: 7



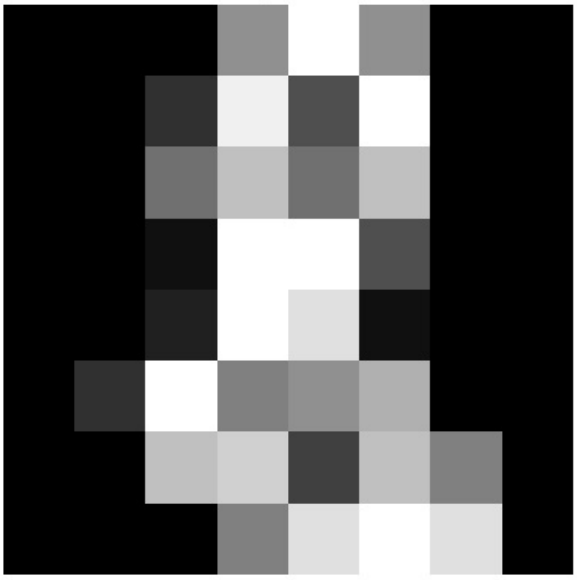
Actual: 9, Predicted: 9



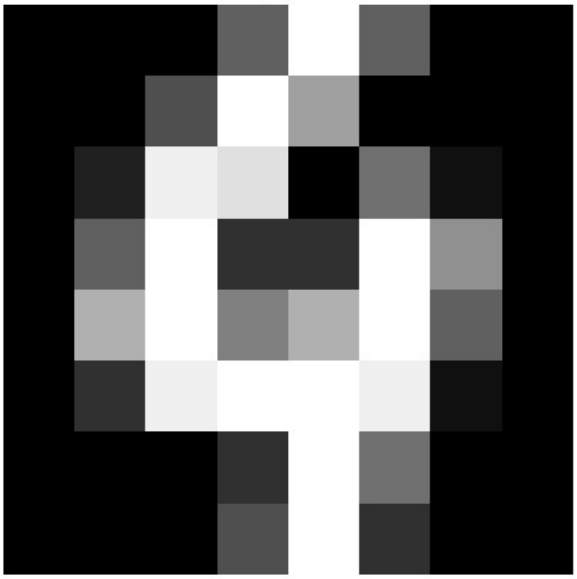
Actual: 2, Predicted: 2



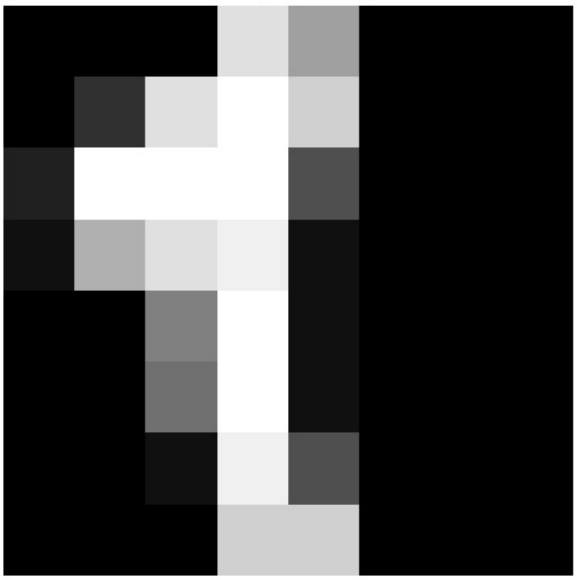
Actual: 8, Predicted: 8



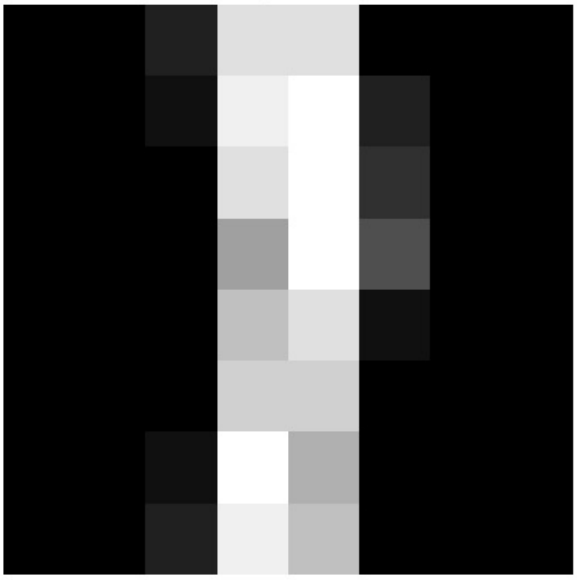
Actual: 4, Predicted: 4



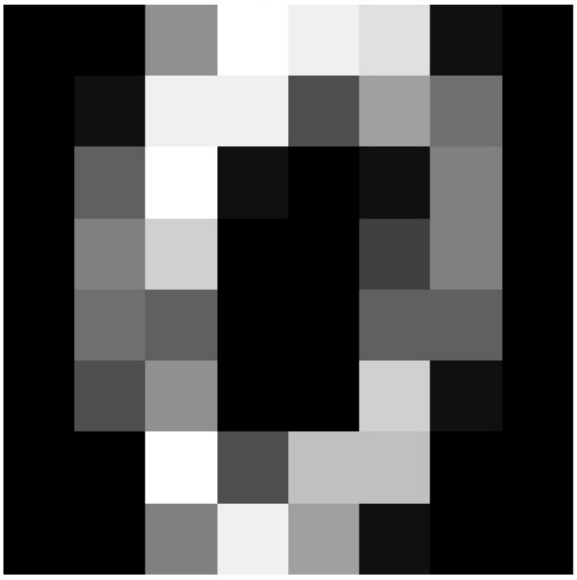
Actual: 1, Predicted: 1



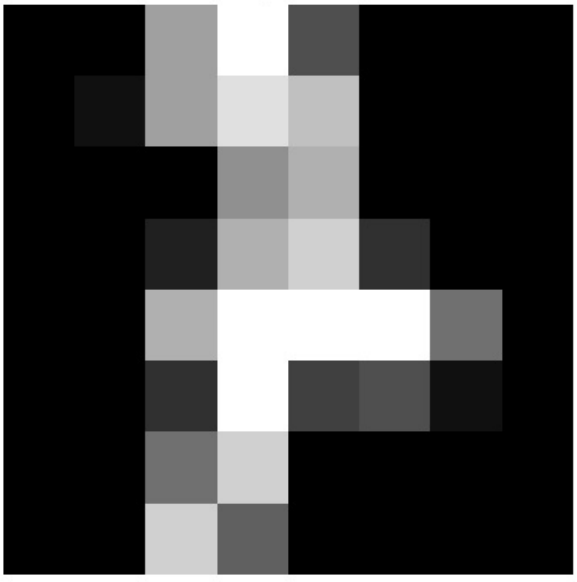
Actual: 1, Predicted: 1



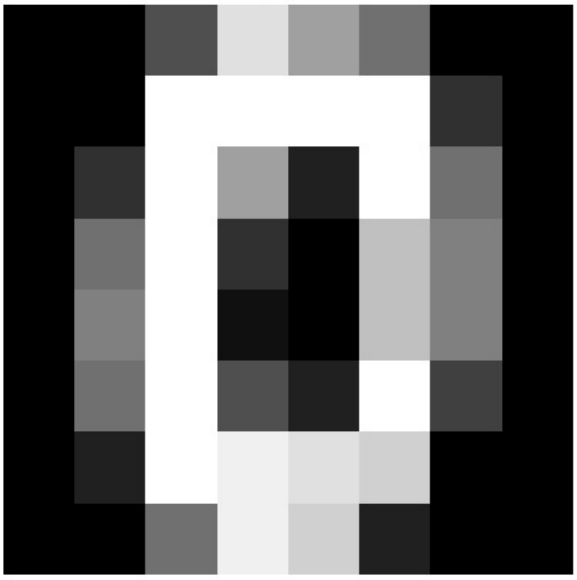
Actual: 0, Predicted: 0



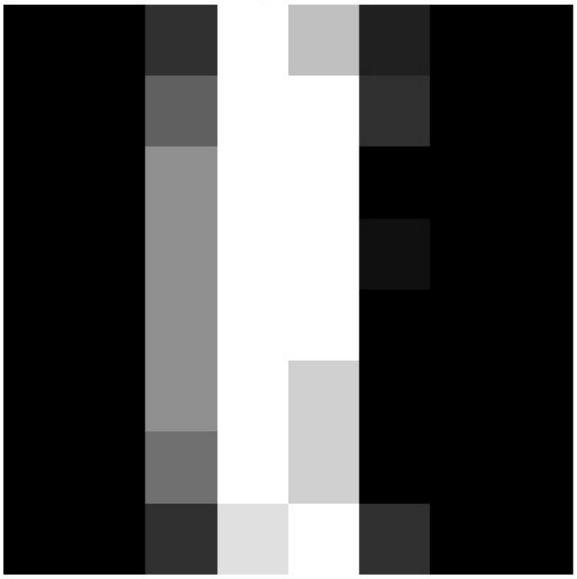
Actual: 7, Predicted: 7



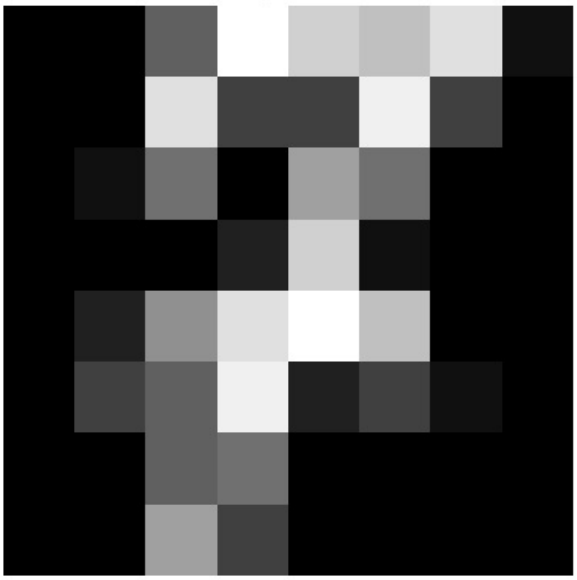
Actual: 0, Predicted: 0



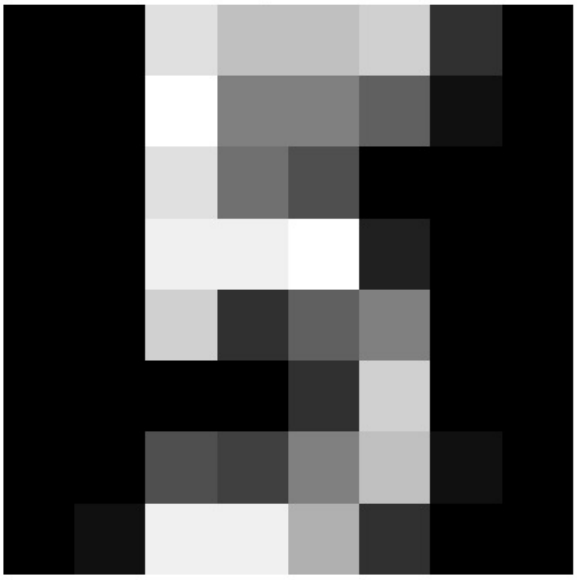
Actual: 1, Predicted: 1



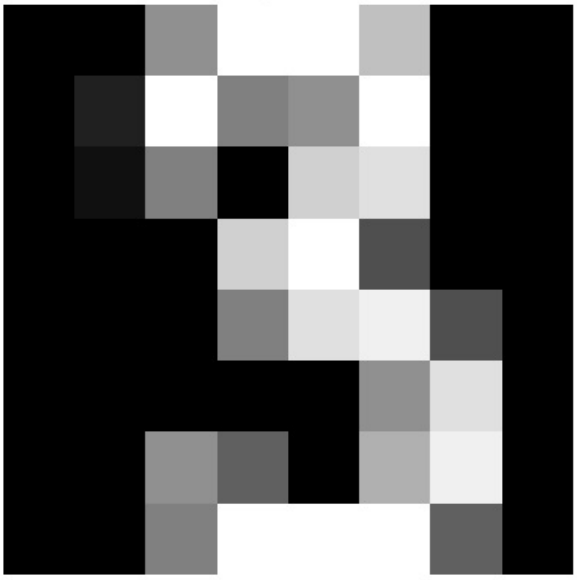
Actual: 7, Predicted: 7



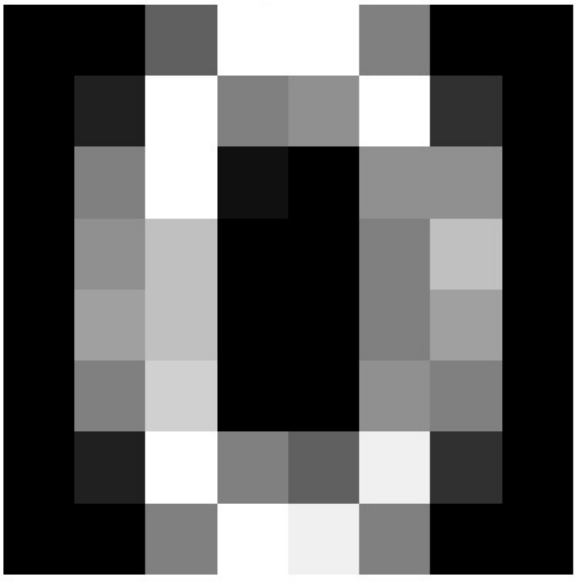
Actual: 5, Predicted: 5



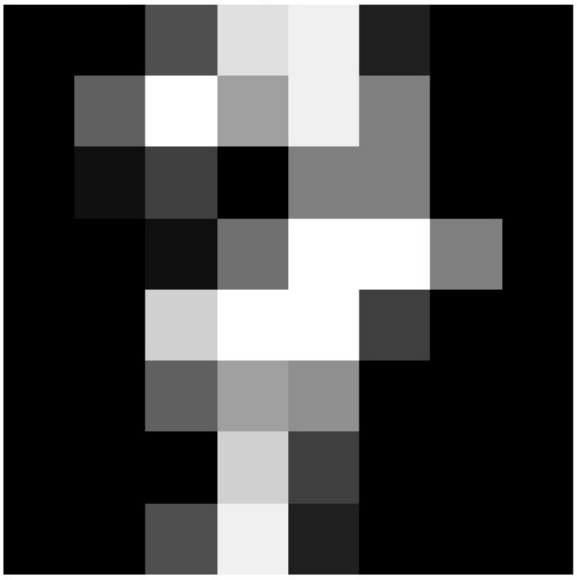
Actual: 3, Predicted: 3



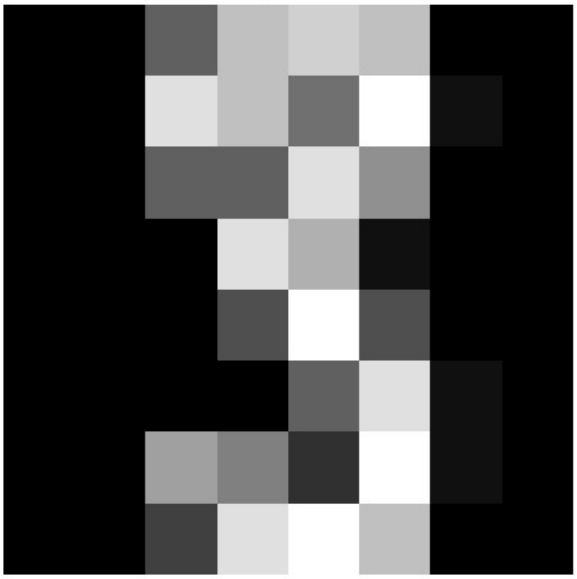
Actual: 0, Predicted: 0



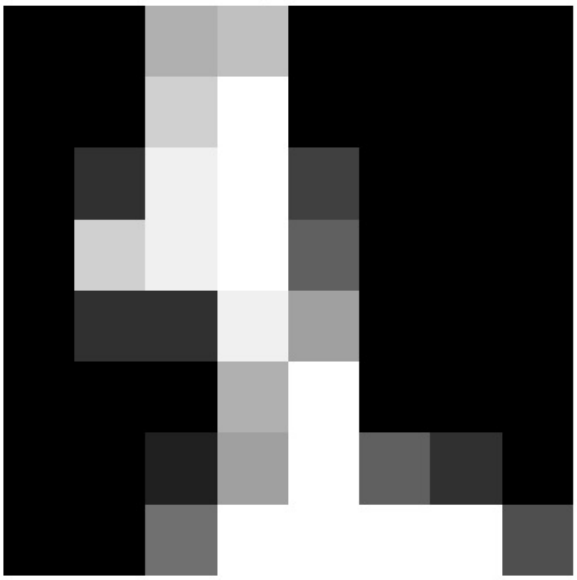
Actual: 7, Predicted: 7



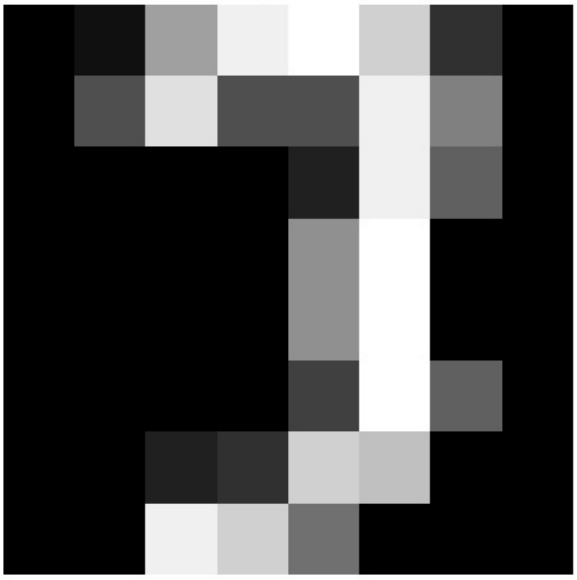
Actual: 3, Predicted: 3



Actual: 1, Predicted: 1

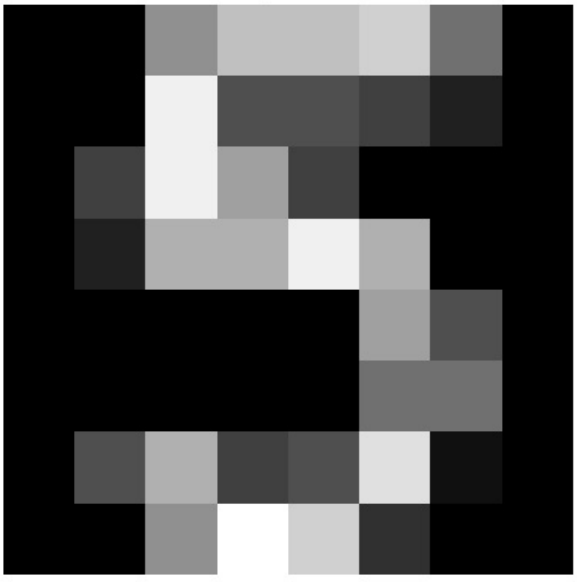


Actual: 3, Predicted: 3

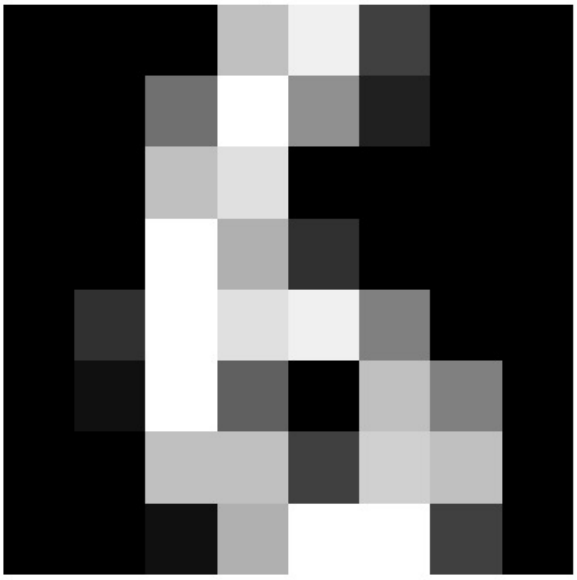


Actual: 2, Predicted: 2

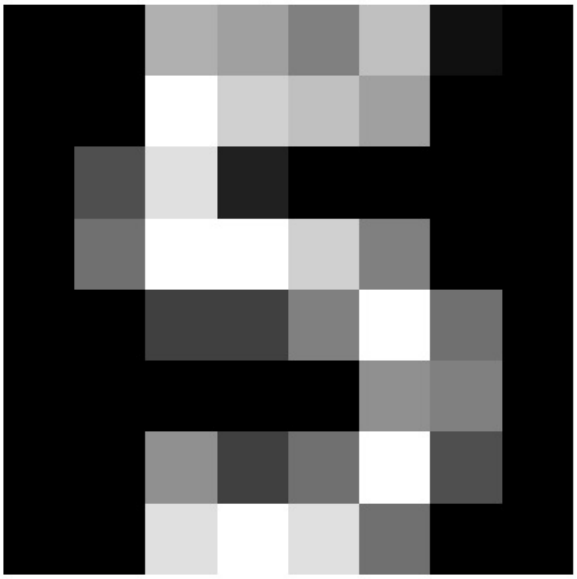
Actual: 5, Predicted: 5



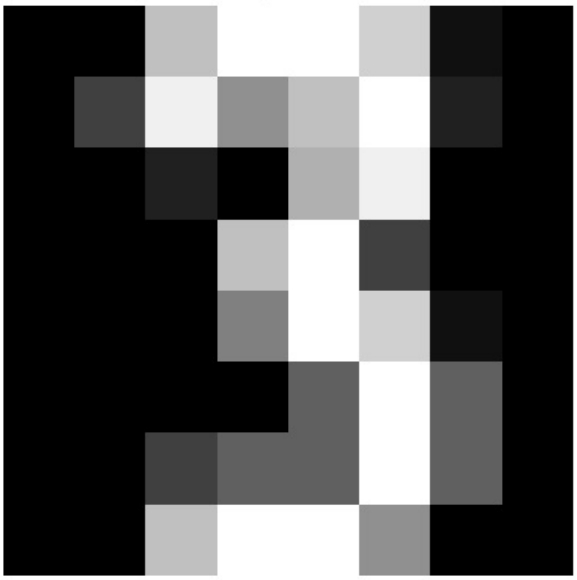
Actual: 6, Predicted: 6



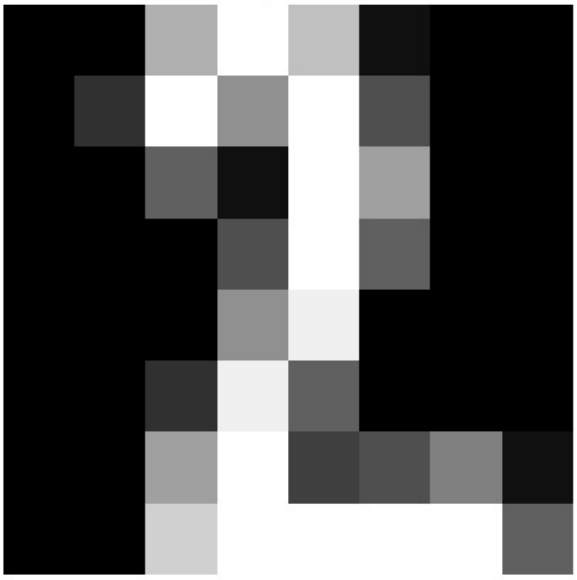
Actual: 5, Predicted: 5



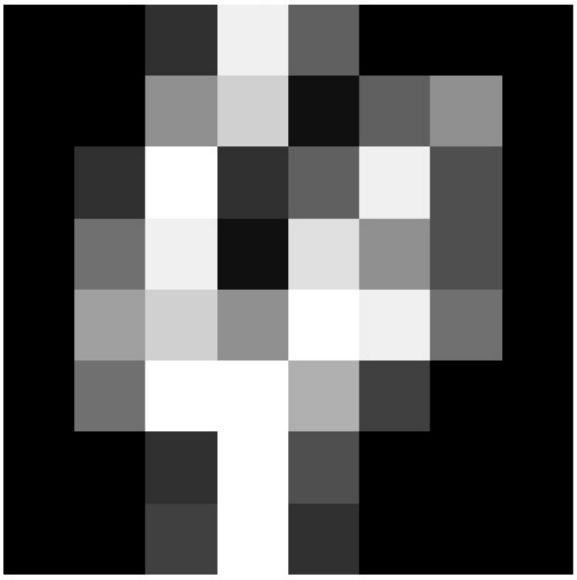
Actual: 3, Predicted: 3



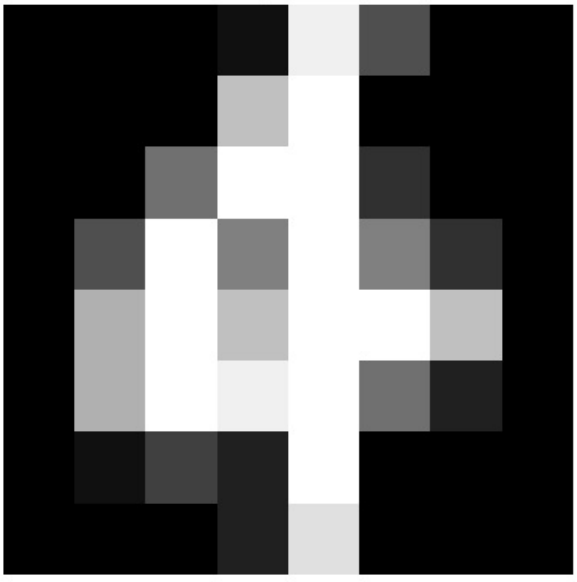
Actual: 2, Predicted: 2



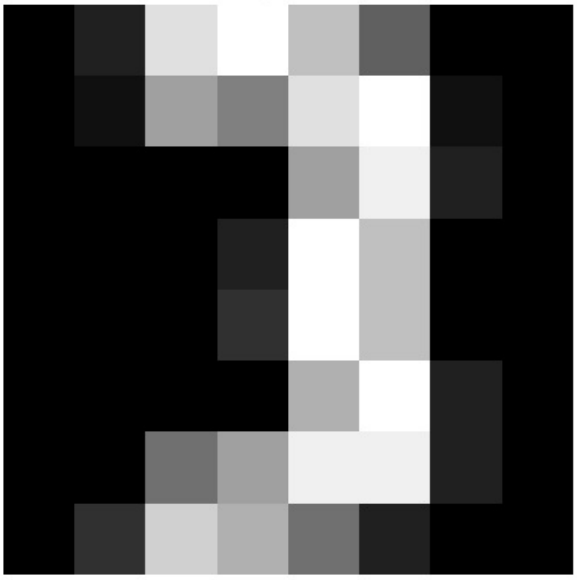
Actual: 4, Predicted: 4



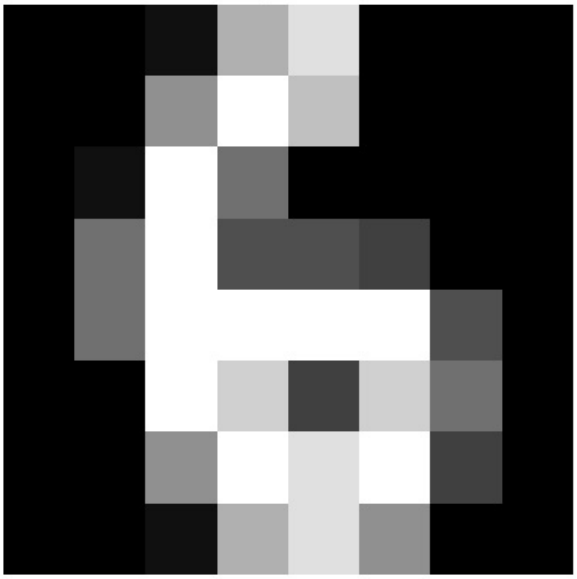
Actual: 4, Predicted: 4



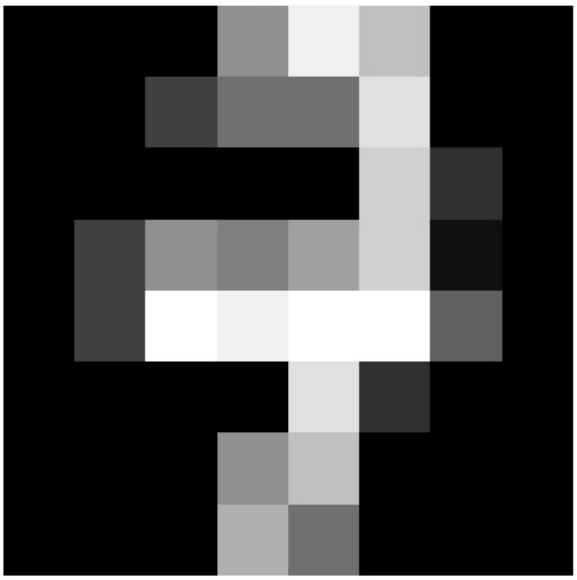
Actual: 3, Predicted: 3



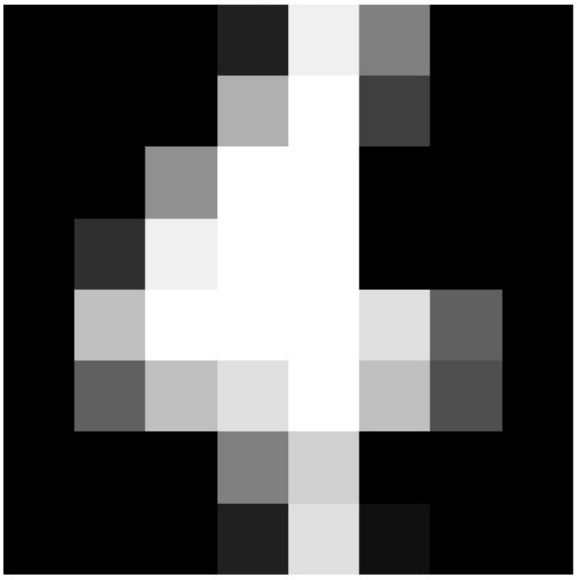
Actual: 6, Predicted: 6



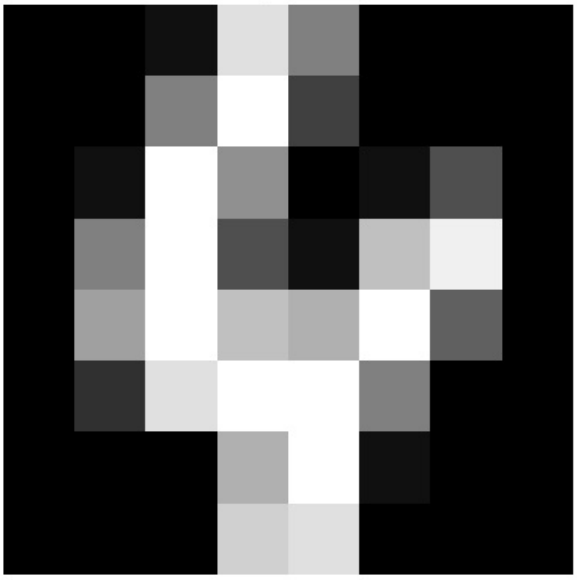
Actual: 7, Predicted: 7



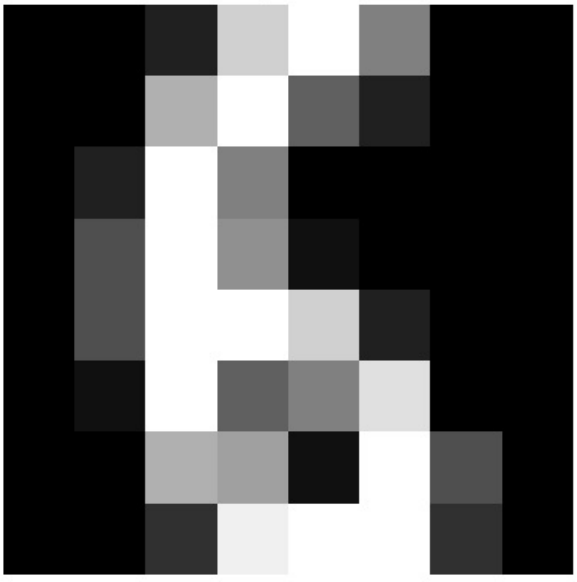
Actual: 4, Predicted: 4



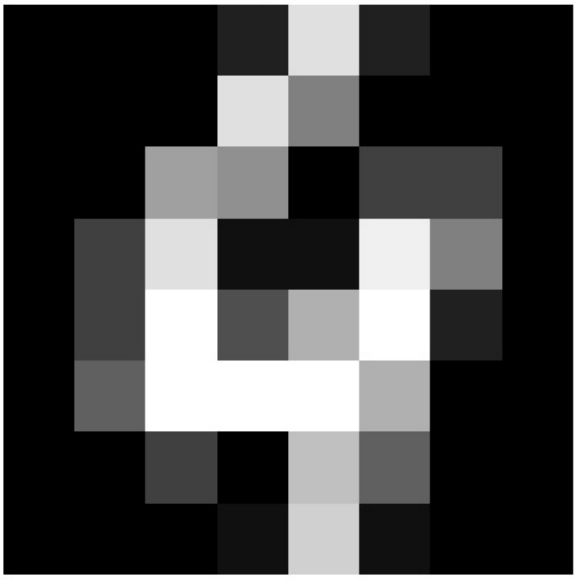
Actual: 4, Predicted: 4



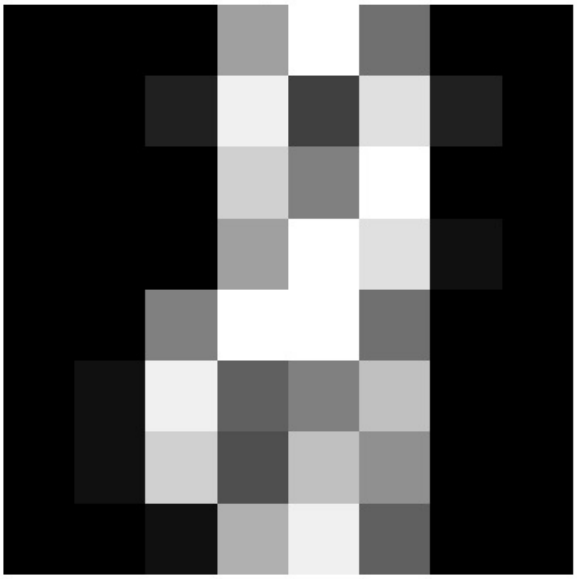
Actual: 6, Predicted: 6



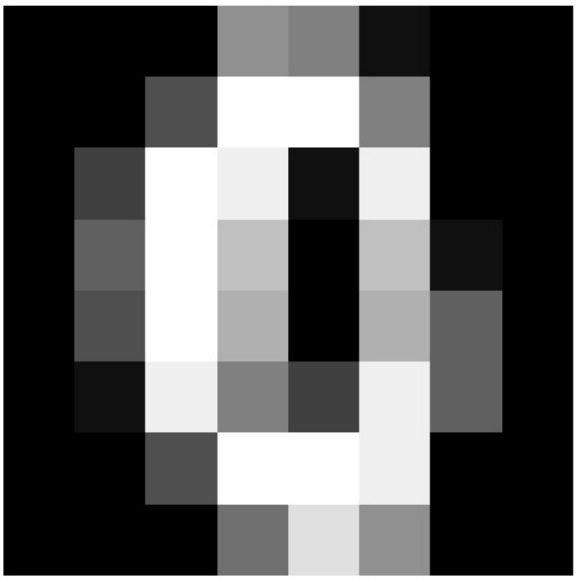
Actual: 4, Predicted: 4



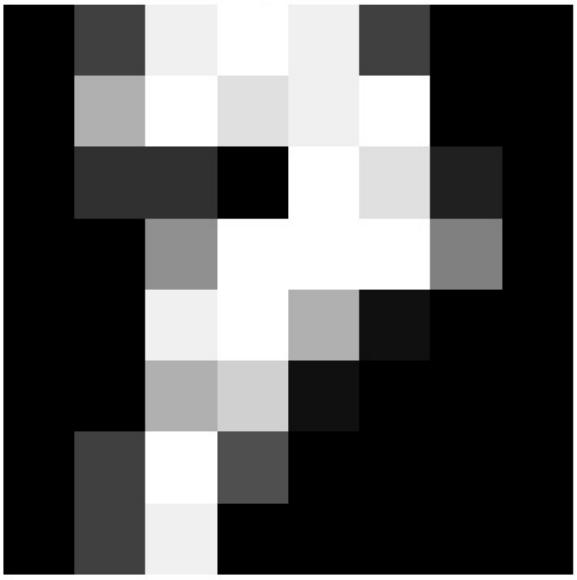
Actual: 8, Predicted: 8



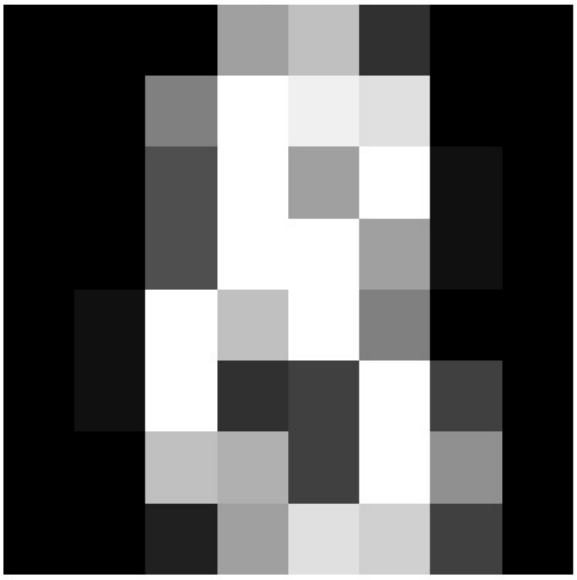
Actual: 0, Predicted: 0



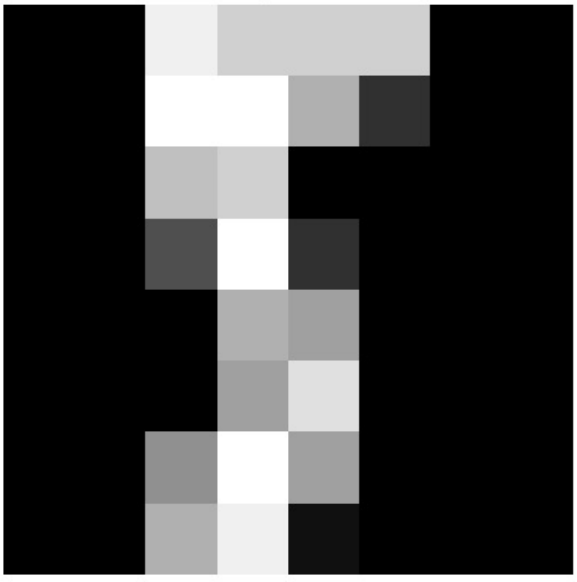
Actual: 7, Predicted: 7



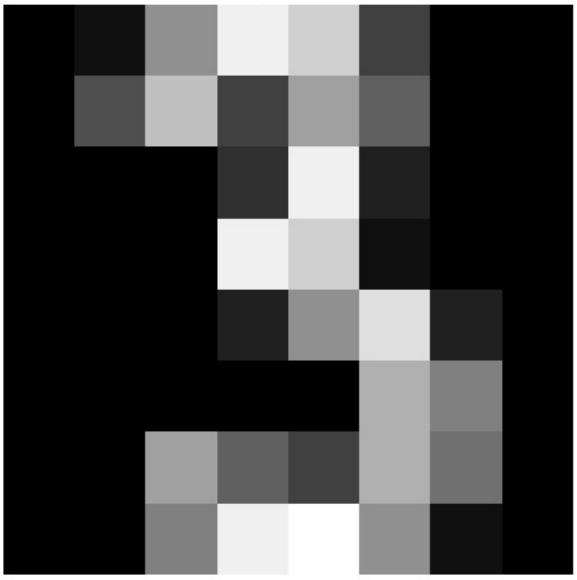
Actual: 8, Predicted: 8



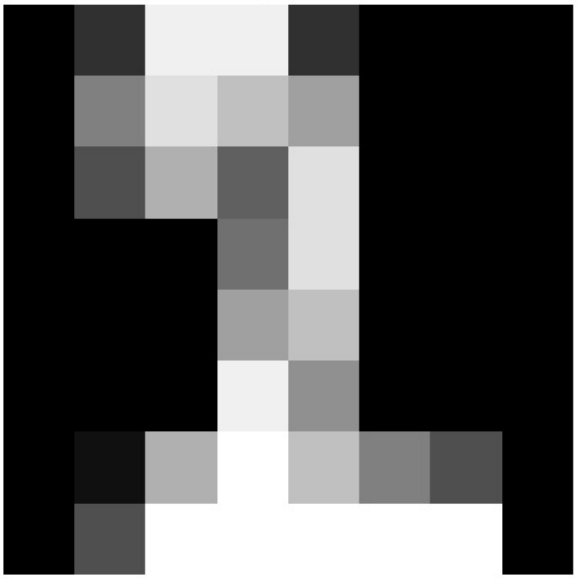
Actual: 5, Predicted: 5



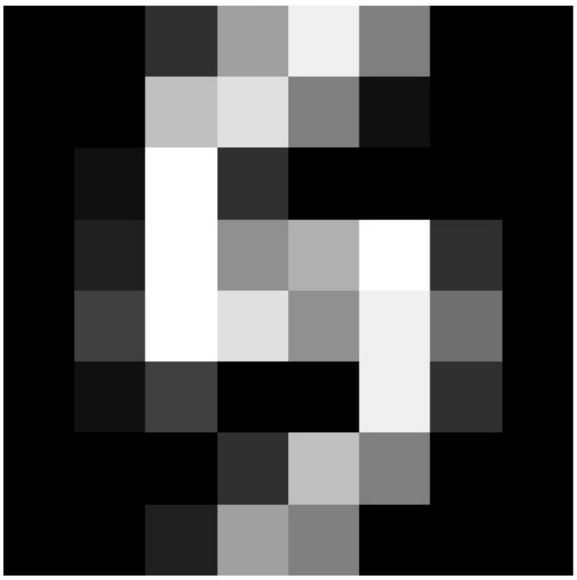
Actual: 3, Predicted: 3



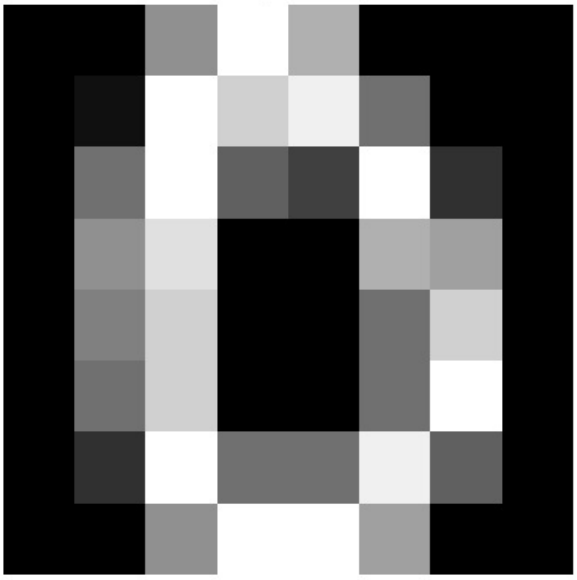
Actual: 2, Predicted: 2



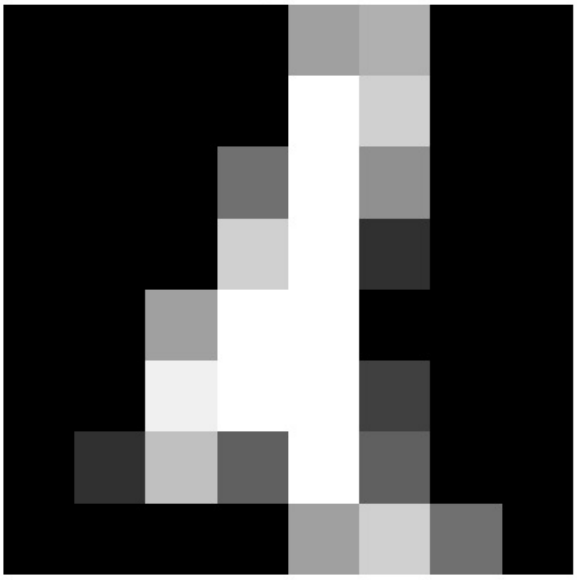
Actual: 5, Predicted: 5



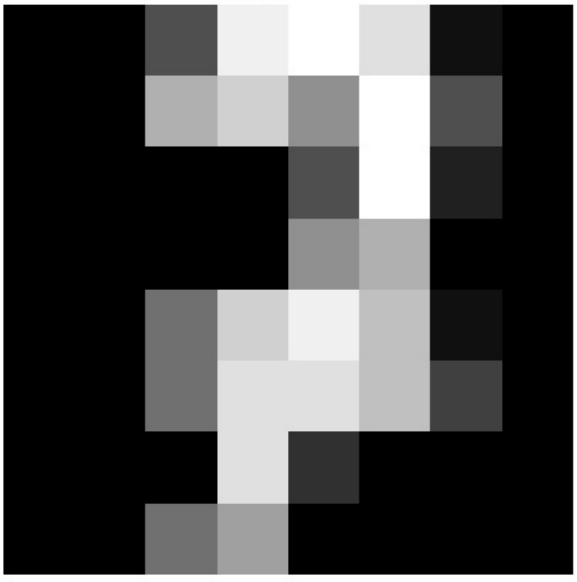
Actual: 0, Predicted: 0



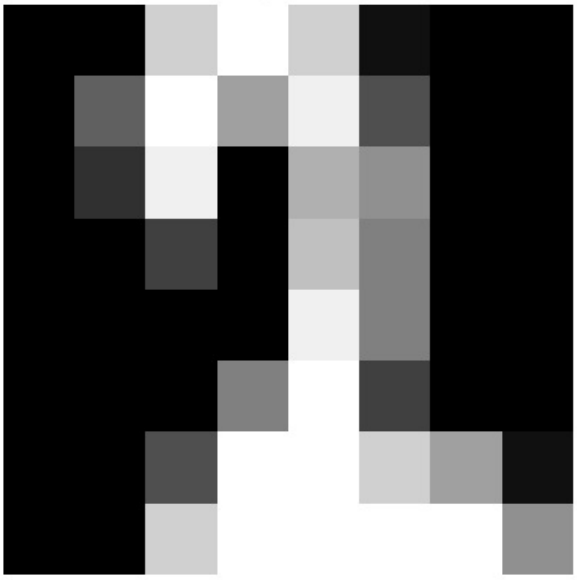
Actual: 1, Predicted: 1



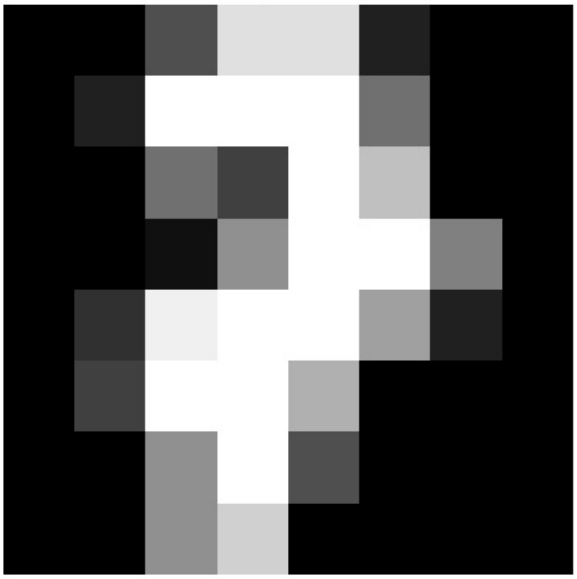
Actual: 7, Predicted: 7



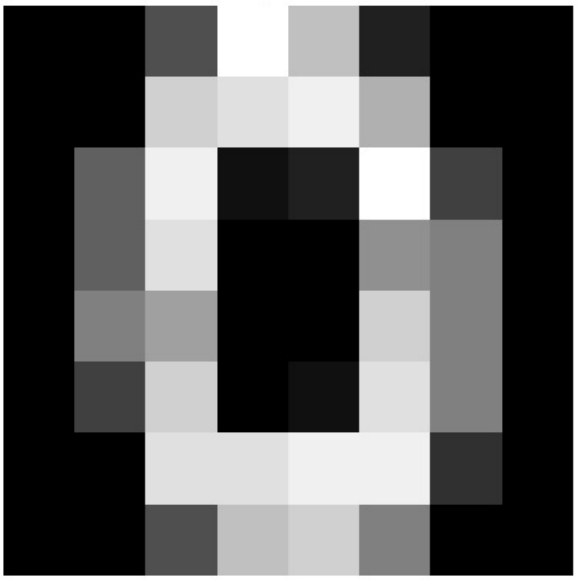
Actual: 2, Predicted: 2



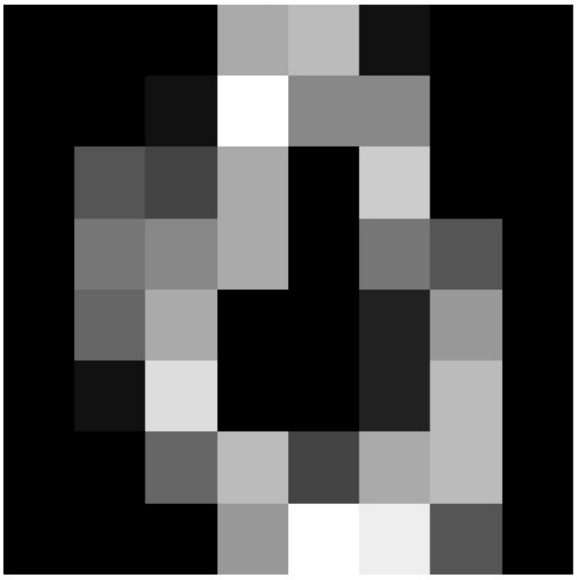
Actual: 7, Predicted: 7



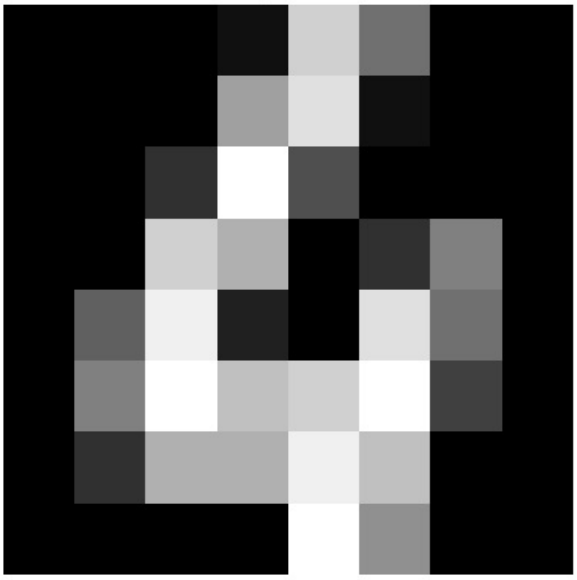
Actual: 0, Predicted: 0



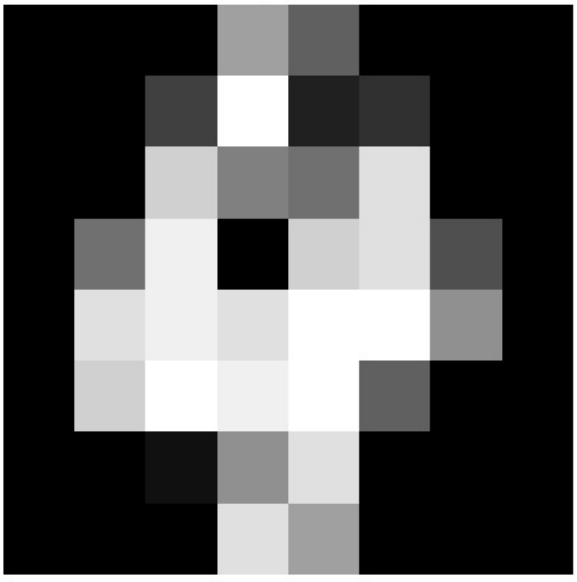
Actual: 0, Predicted: 0



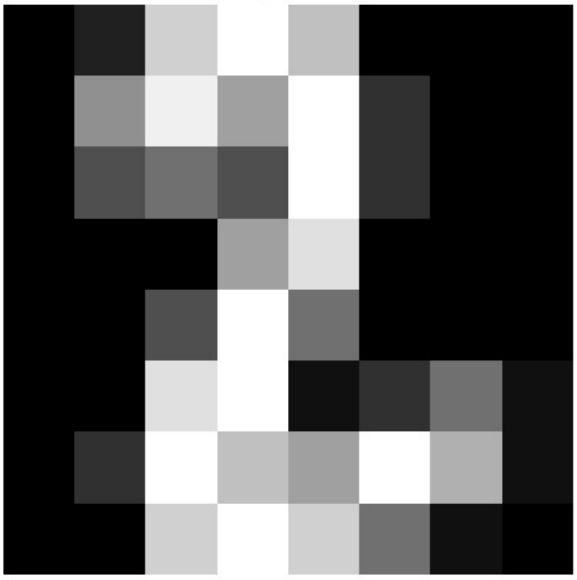
Actual: 4, Predicted: 4



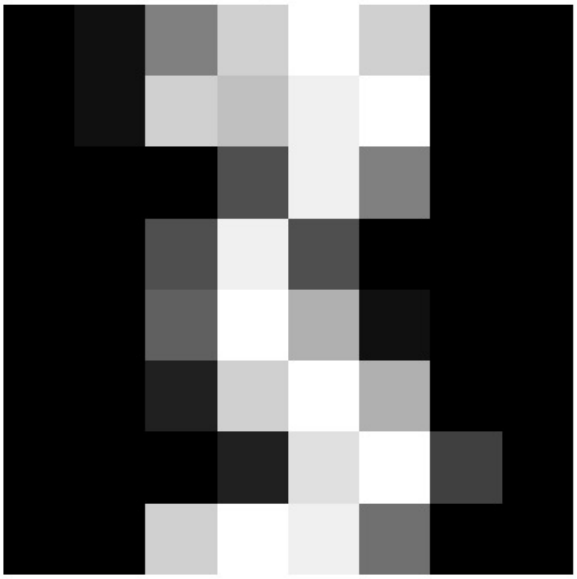
Actual: 4, Predicted: 4



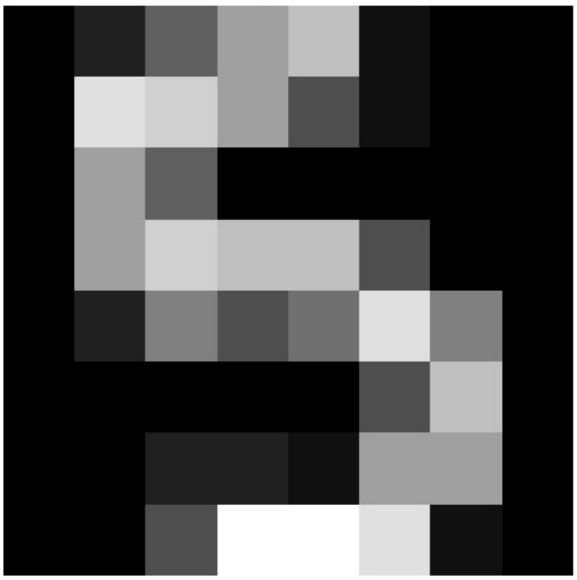
Actual: 2, Predicted: 2



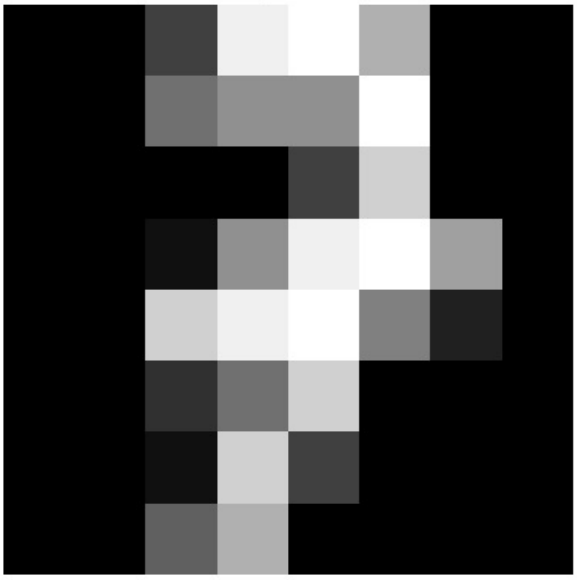
Actual: 3, Predicted: 3



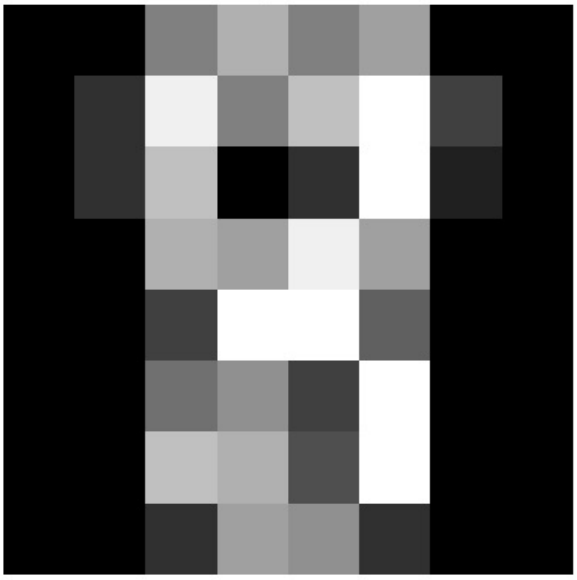
Actual: 5, Predicted: 5



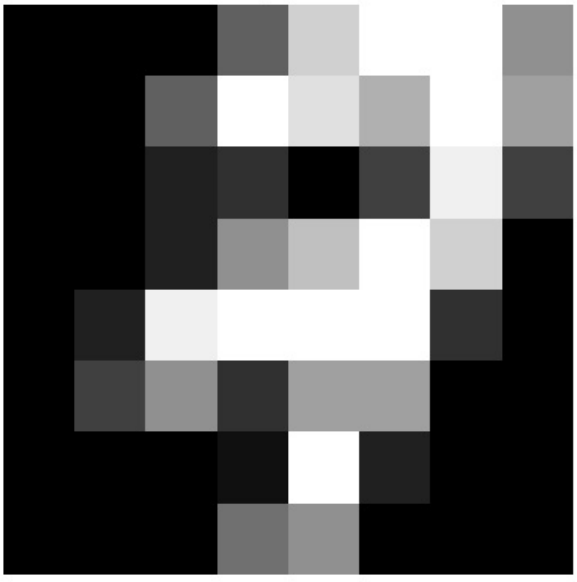
Actual: 7, Predicted: 7



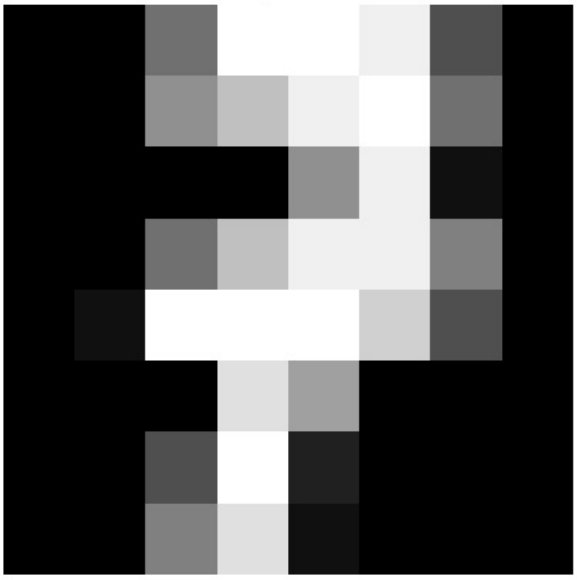
Actual: 8, Predicted: 8



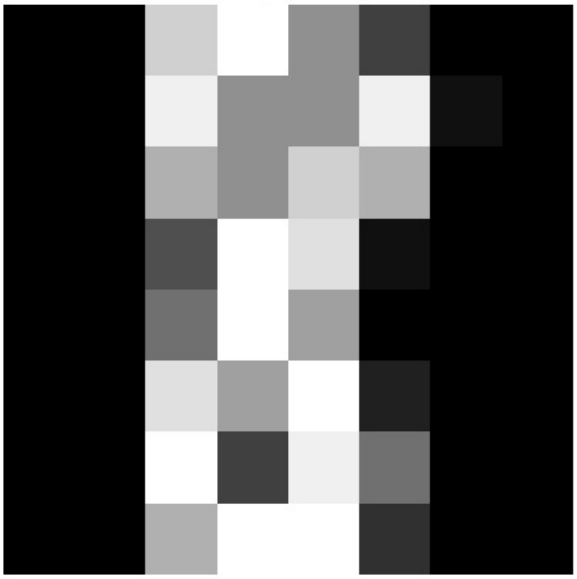
Actual: 7, Predicted: 7



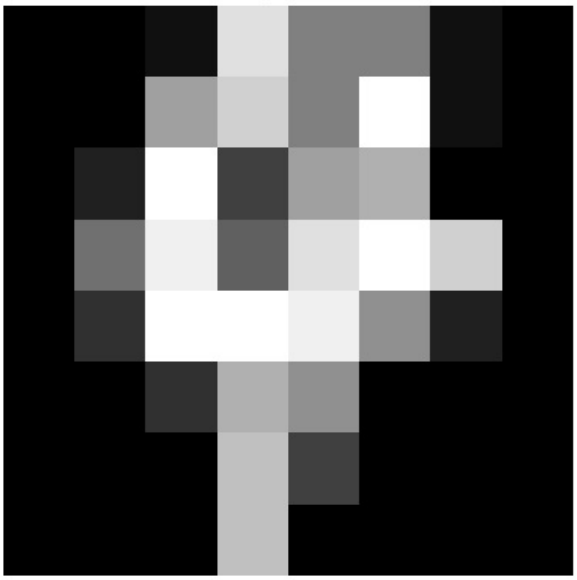
Actual: 7, Predicted: 7



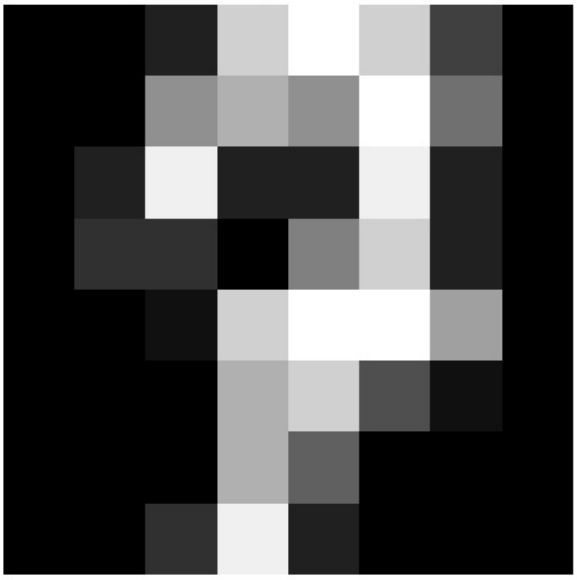
Actual: 8, Predicted: 8



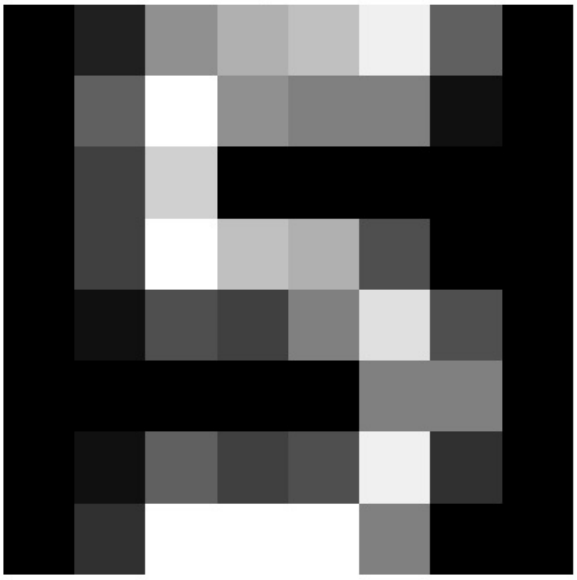
Actual: 4, Predicted: 4



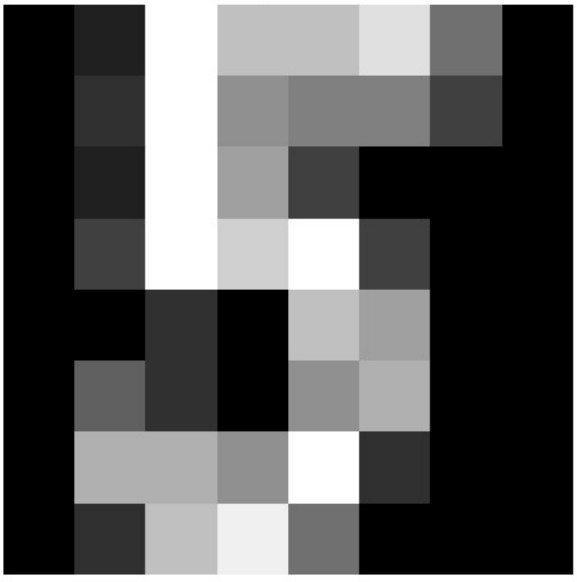
Actual: 7, Predicted: 7



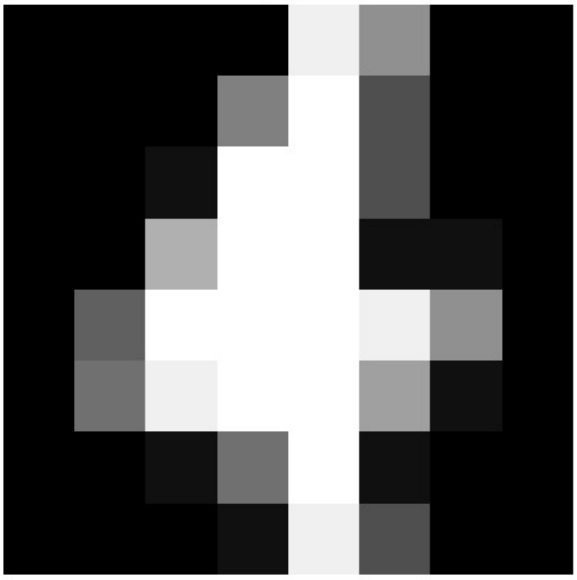
Actual: 5, Predicted: 5



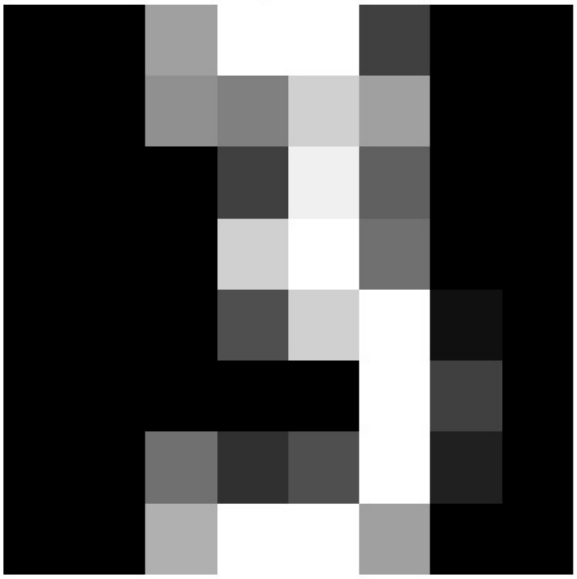
Actual: 5, Predicted: 5



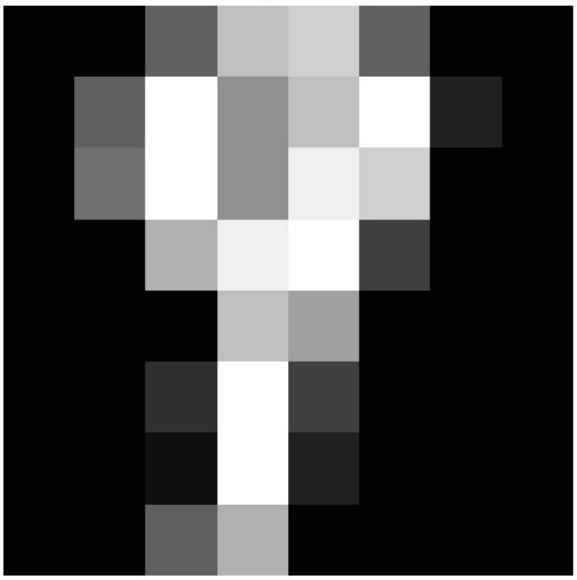
Actual: 4, Predicted: 4



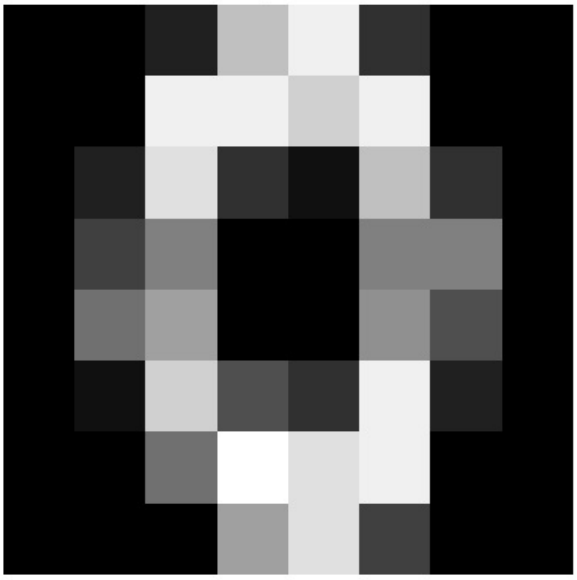
Actual: 3, Predicted: 3



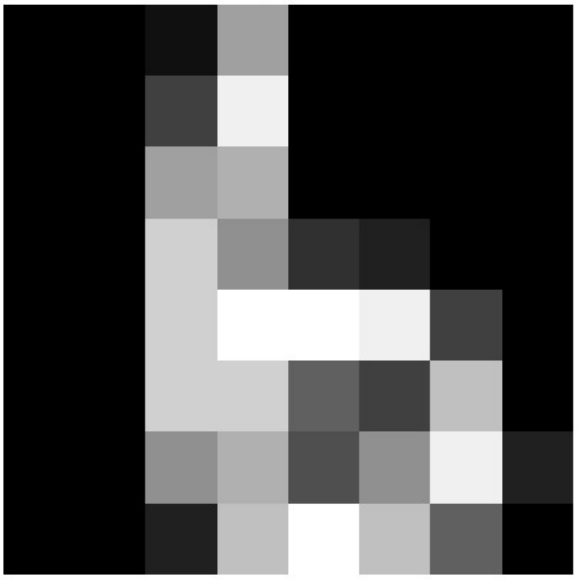
Actual: 9, Predicted: 4



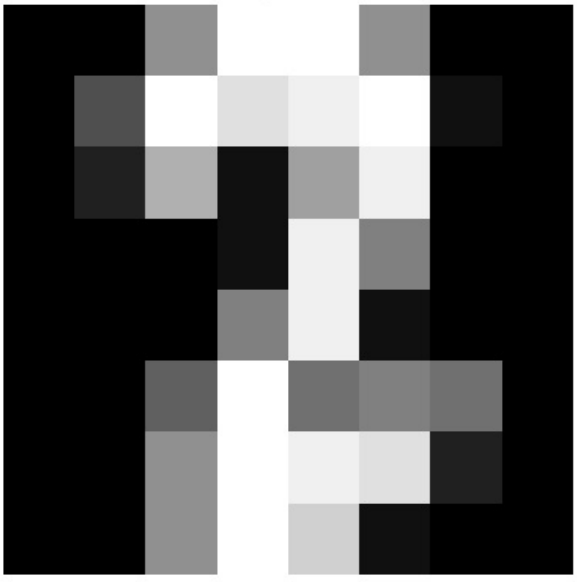
Actual: 0, Predicted: 0



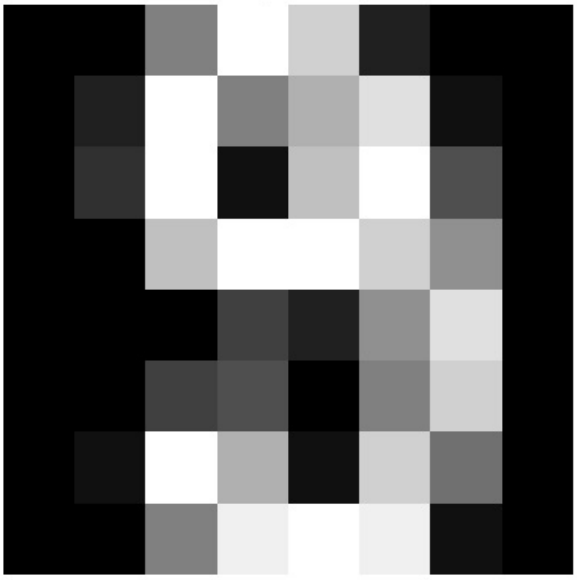
Actual: 6, Predicted: 6



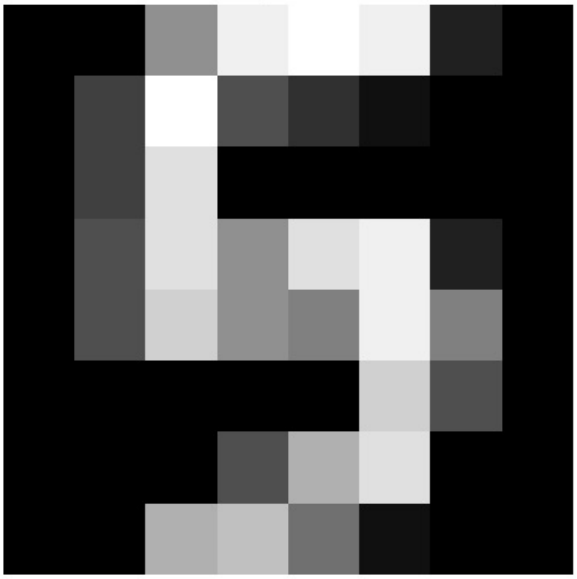
Actual: 2, Predicted: 2



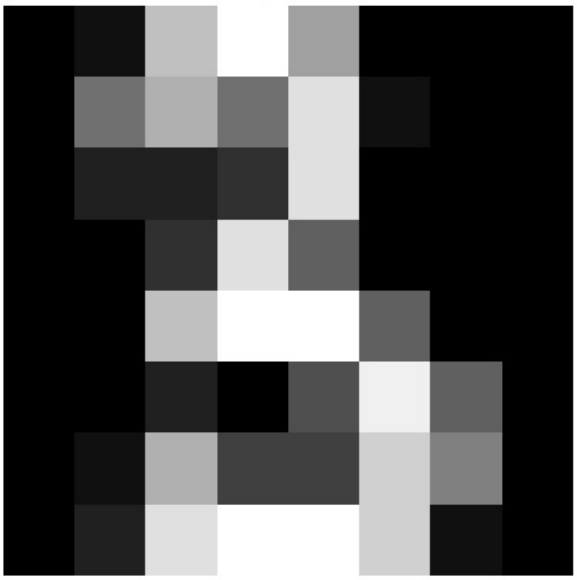
Actual: 9, Predicted: 9



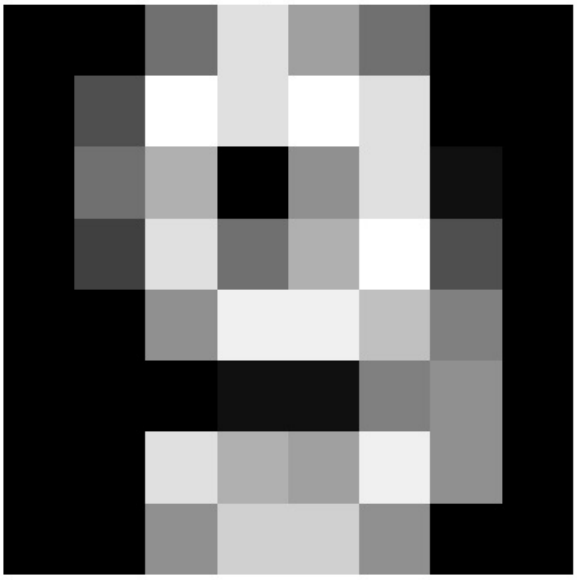
Actual: 5, Predicted: 5



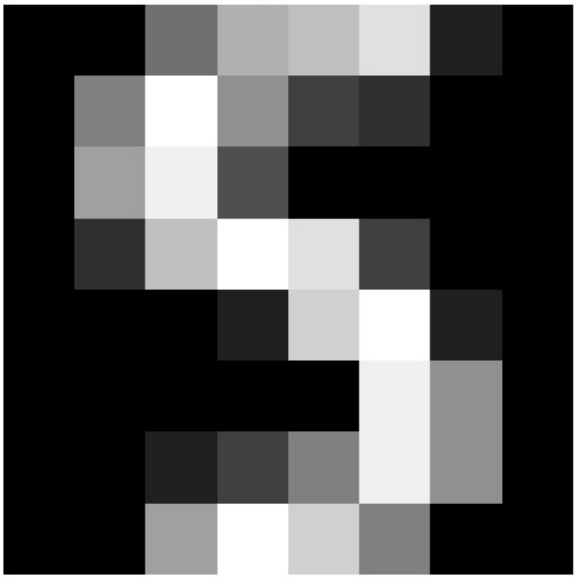
Actual: 3, Predicted: 3



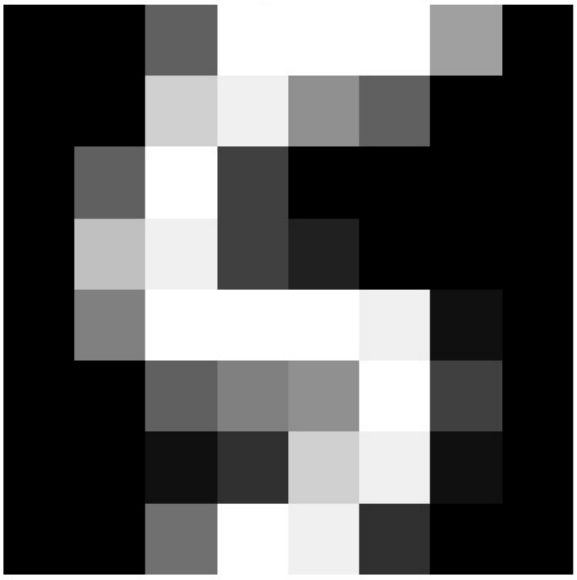
Actual: 9, Predicted: 9



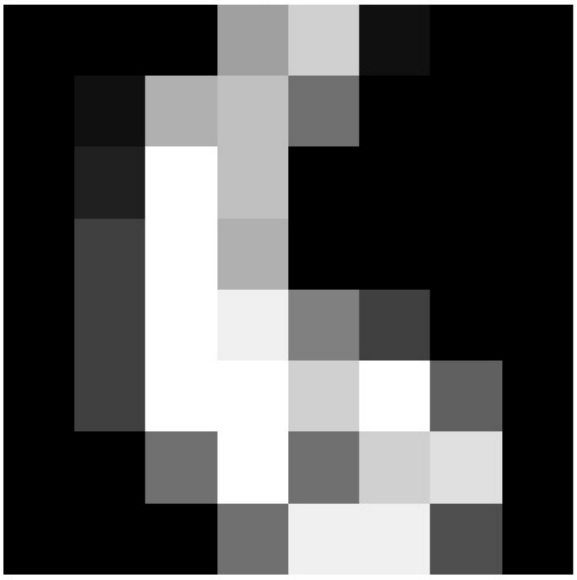
Actual: 5, Predicted: 5



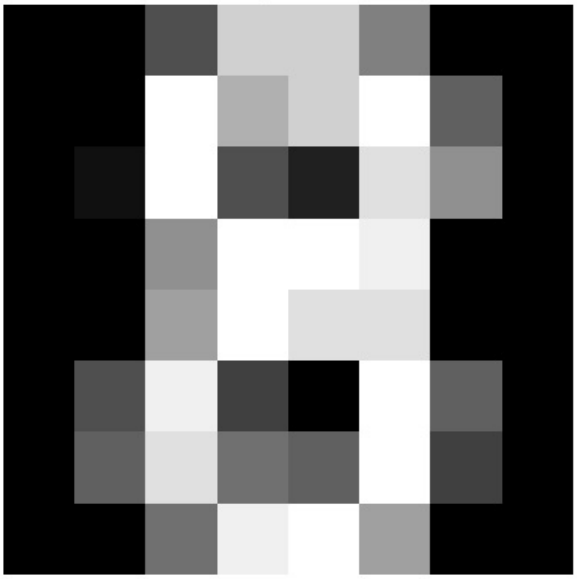
Actual: 5, Predicted: 5



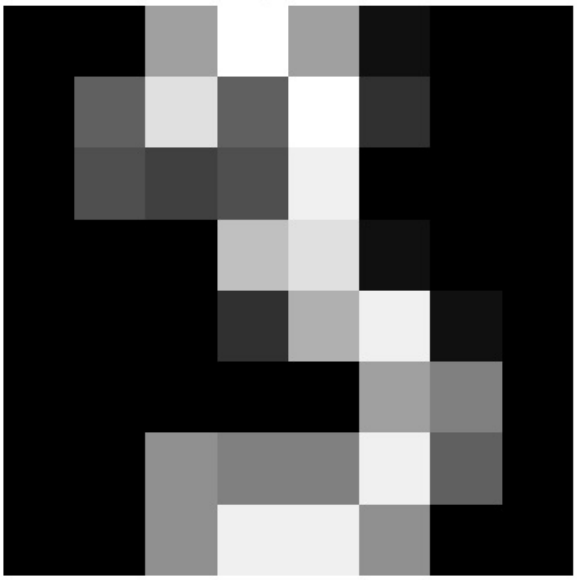
Actual: 6, Predicted: 6



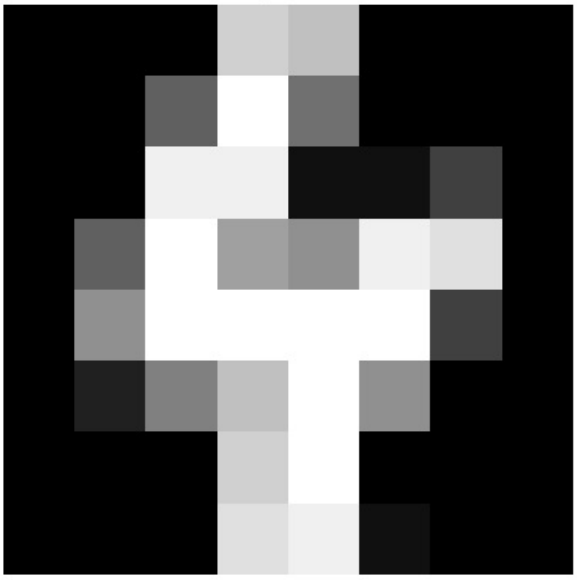
Actual: 8, Predicted: 8



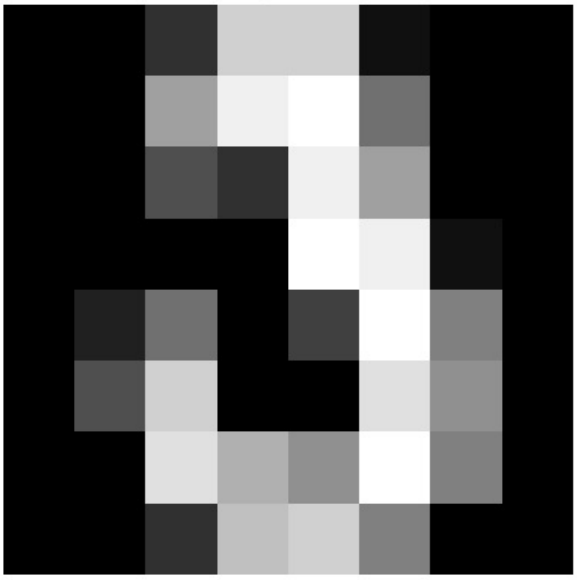
Actual: 3, Predicted: 3



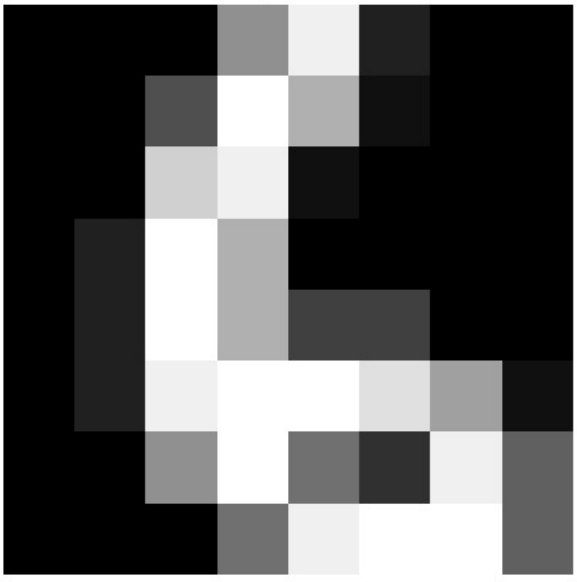
Actual: 4, Predicted: 4



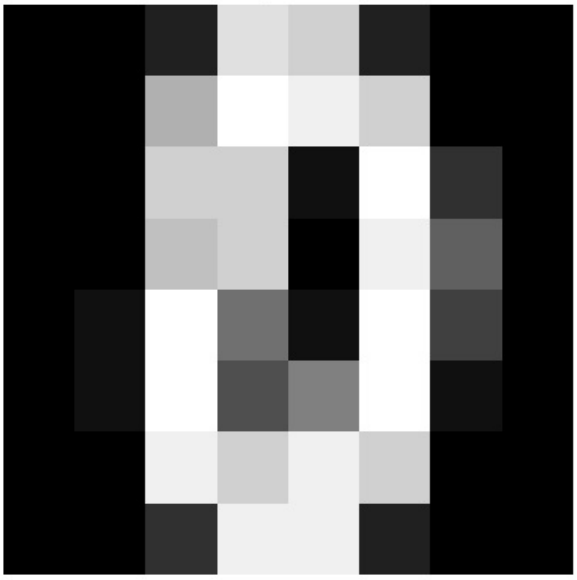
Actual: 3, Predicted: 3



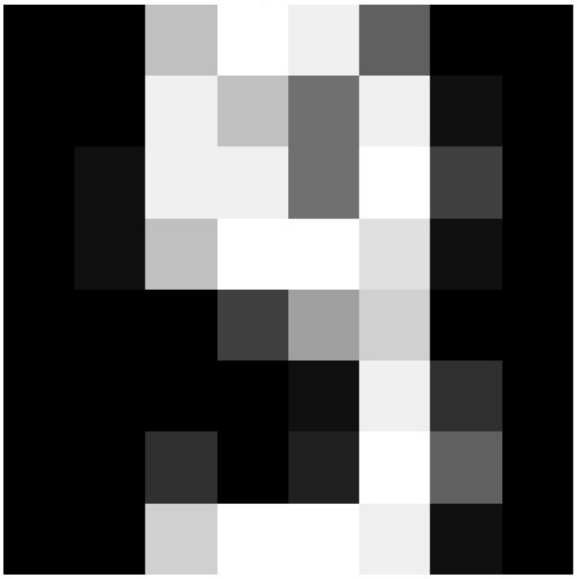
Actual: 6, Predicted: 6



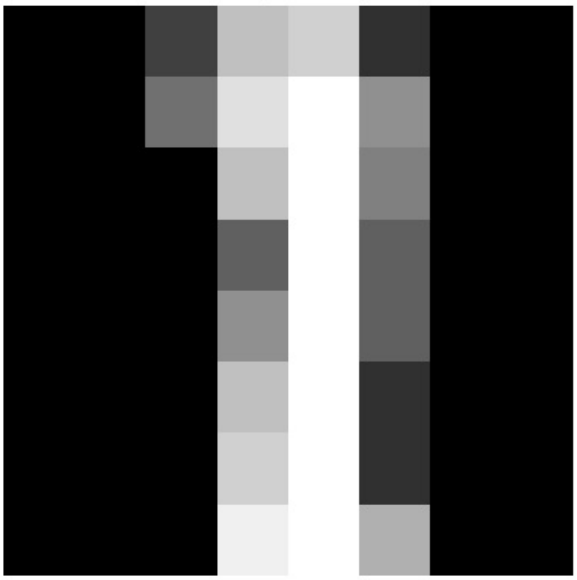
Actual: 0, Predicted: 0



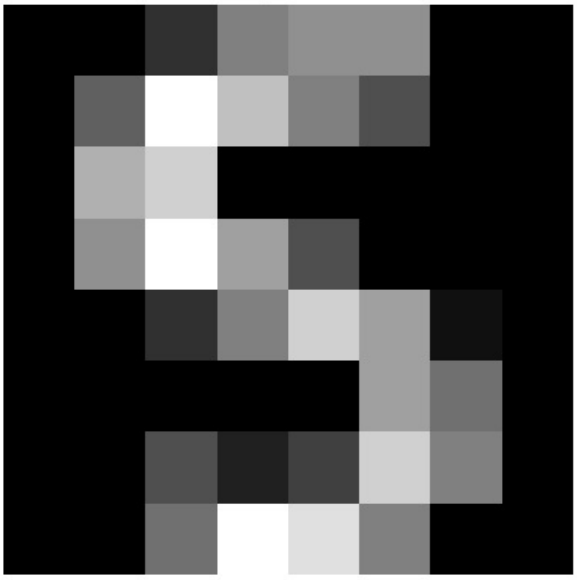
Actual: 9, Predicted: 9



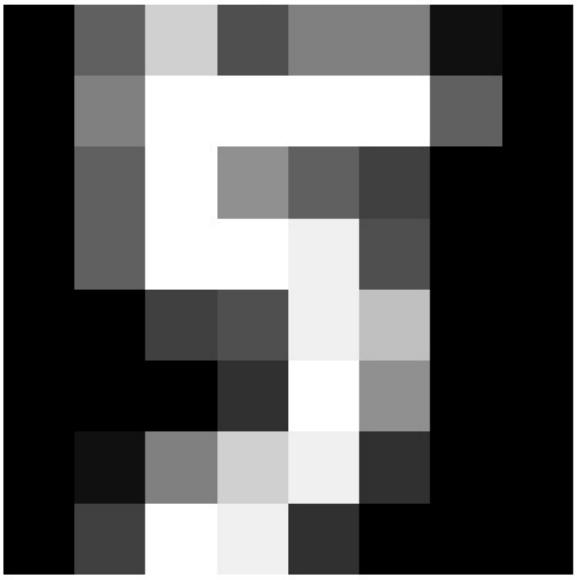
Actual: 1, Predicted: 1



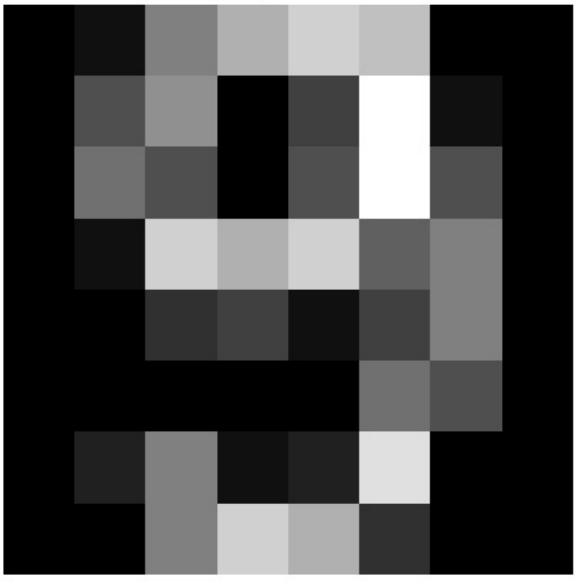
Actual: 5, Predicted: 5



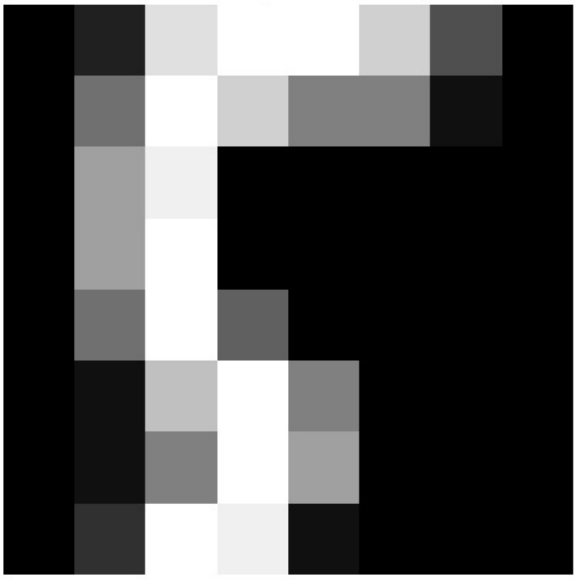
Actual: 5, Predicted: 5



Actual: 9, Predicted: 9

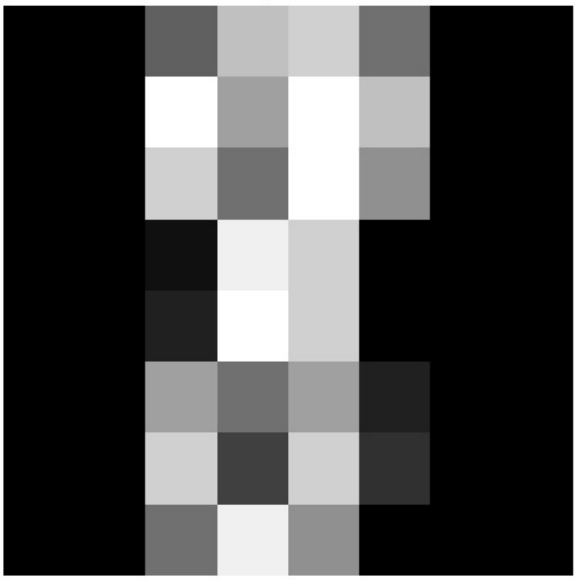


Actual: 5, Predicted: 5

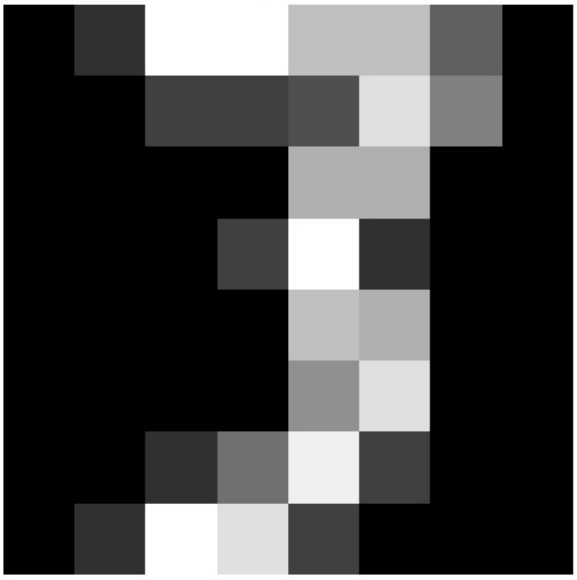


Actual: 5, Predicted: 5

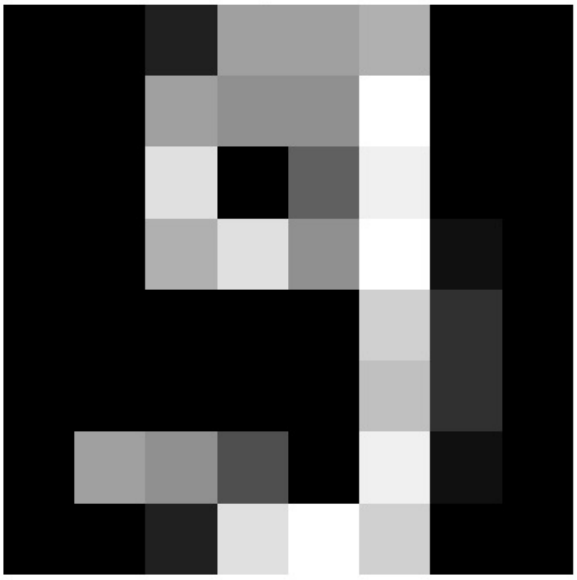
Actual: 8, Predicted: 8



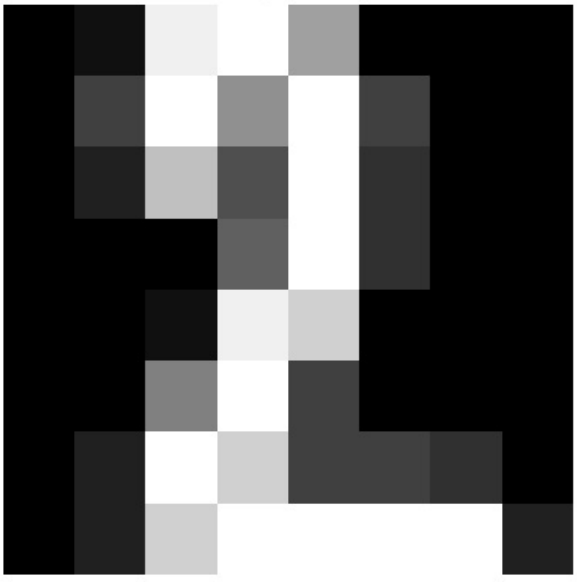
Actual: 3, Predicted: 3



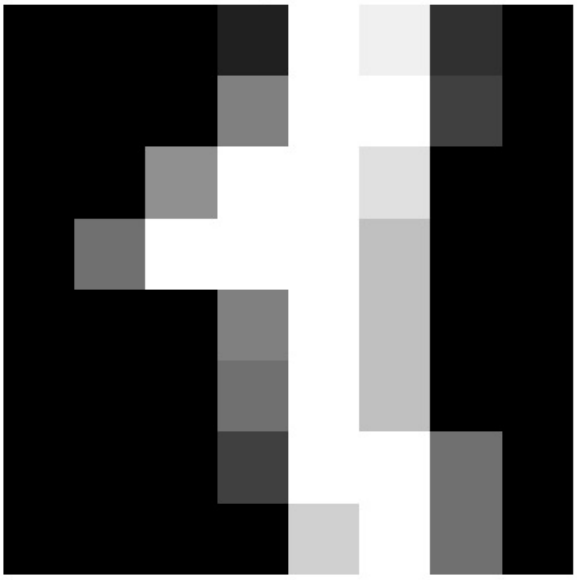
Actual: 9, Predicted: 9



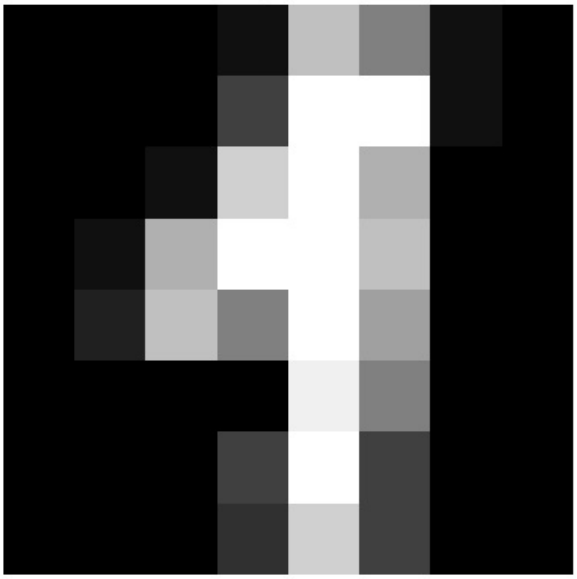
Actual: 2, Predicted: 2



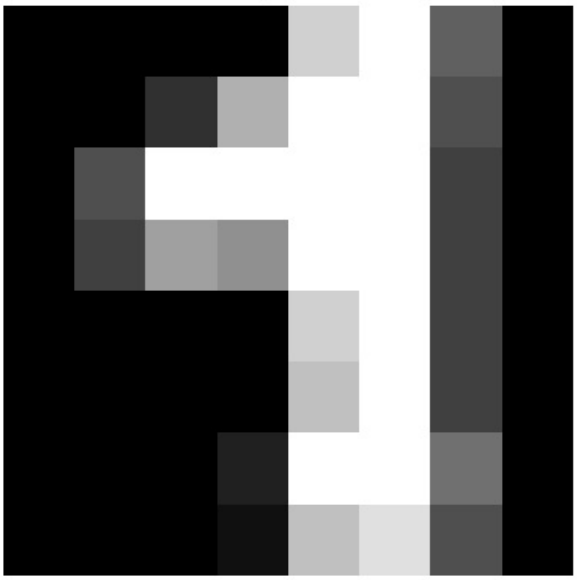
Actual: 1, Predicted: 1



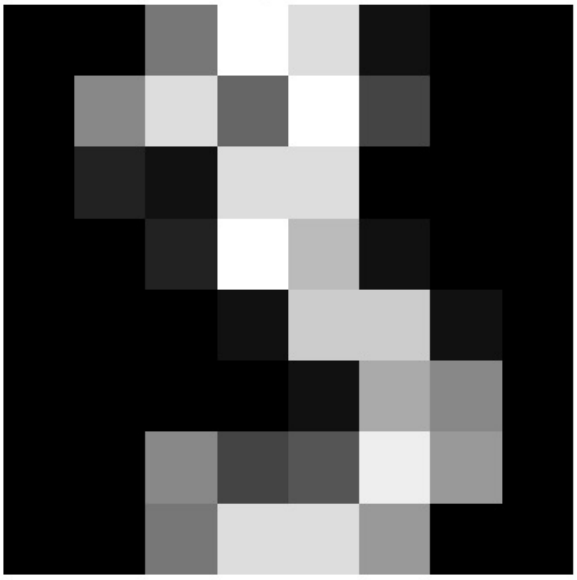
Actual: 1, Predicted: 1



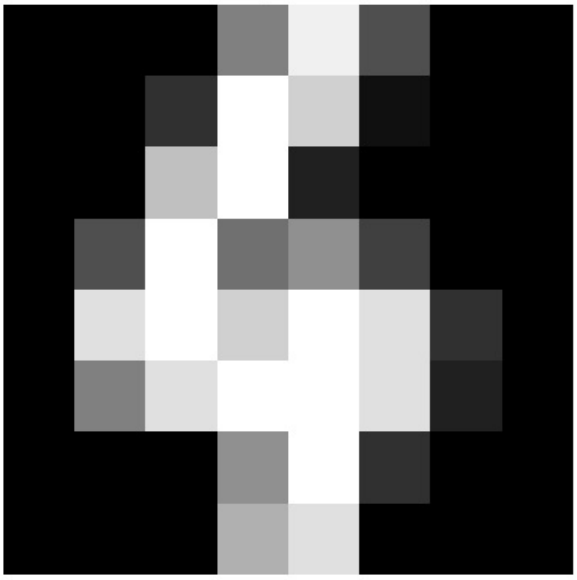
Actual: 1, Predicted: 1



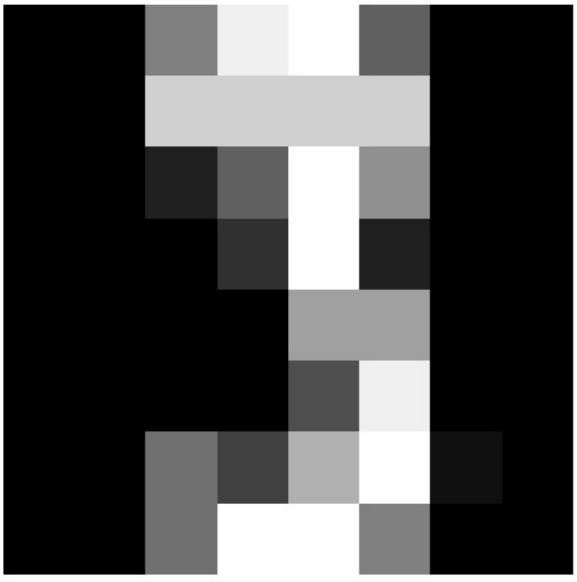
Actual: 3, Predicted: 3



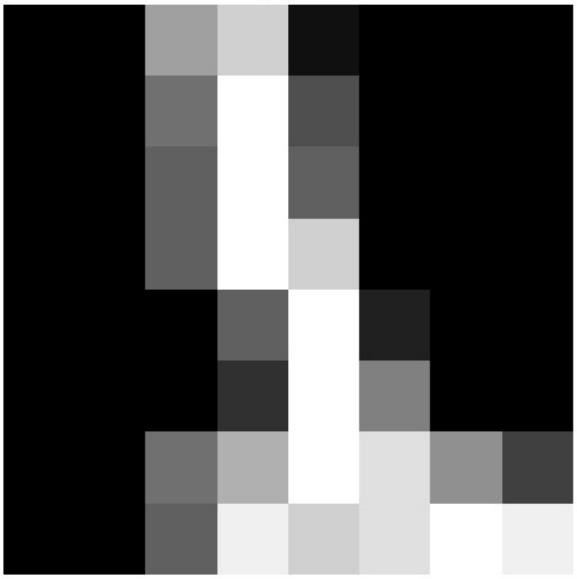
Actual: 4, Predicted: 4



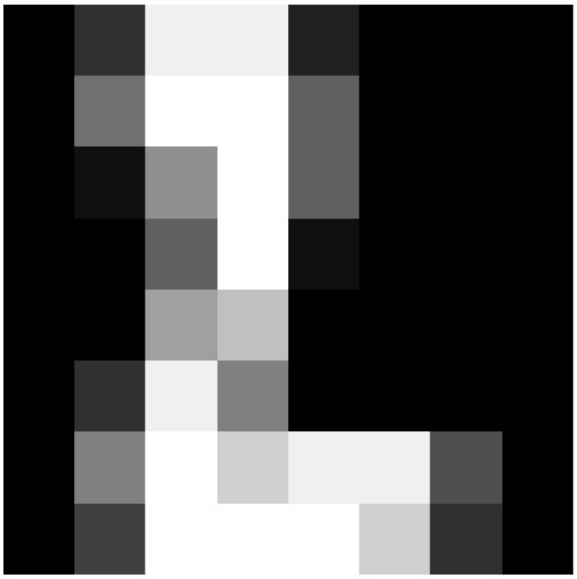
Actual: 3, Predicted: 3



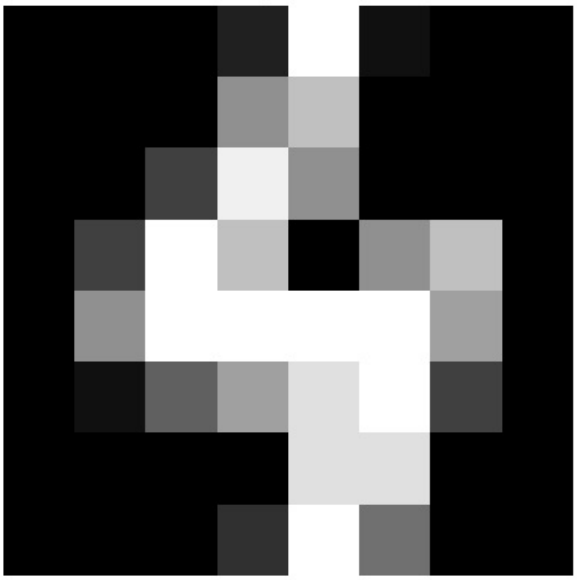
Actual: 1, Predicted: 1



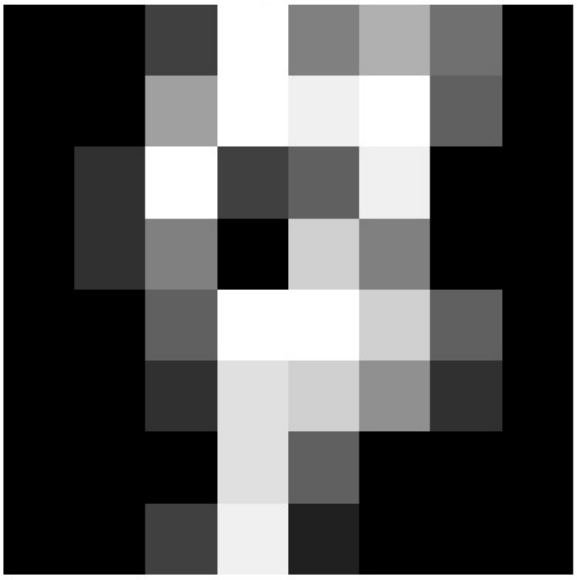
Actual: 2, Predicted: 2



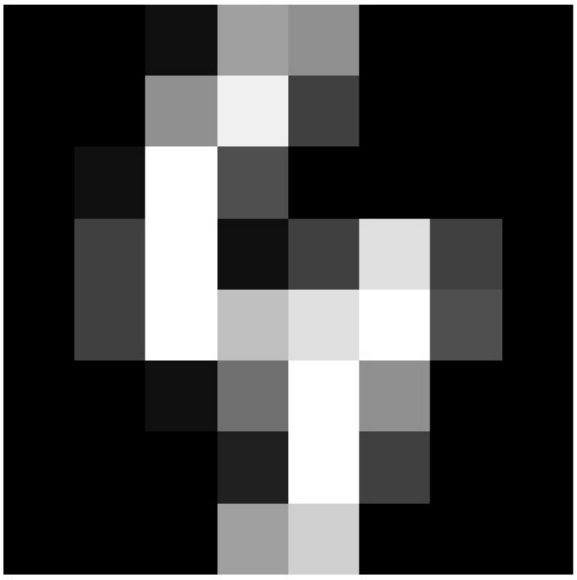
Actual: 4, Predicted: 4



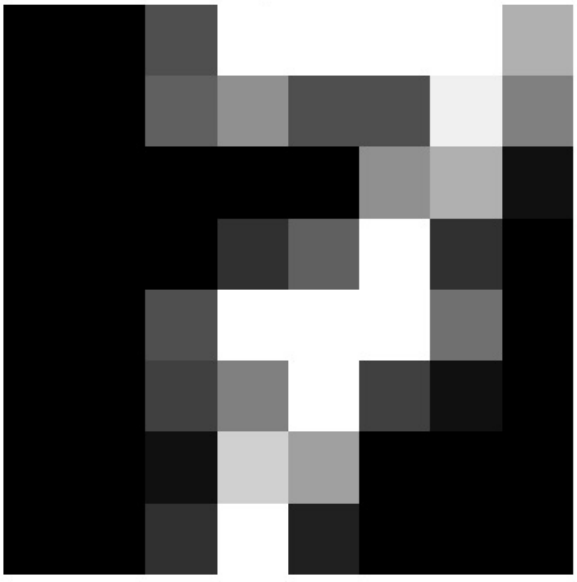
Actual: 7, Predicted: 7



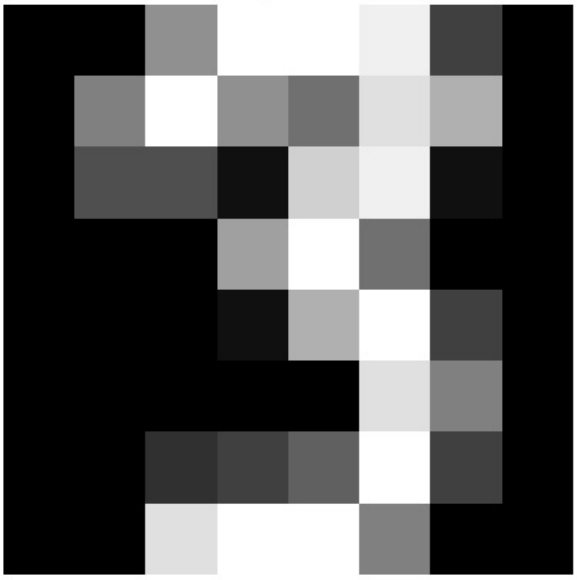
Actual: 4, Predicted: 4



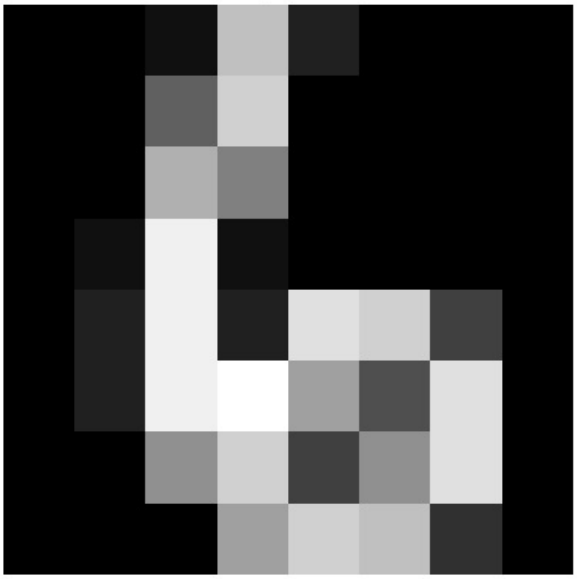
Actual: 7, Predicted: 7



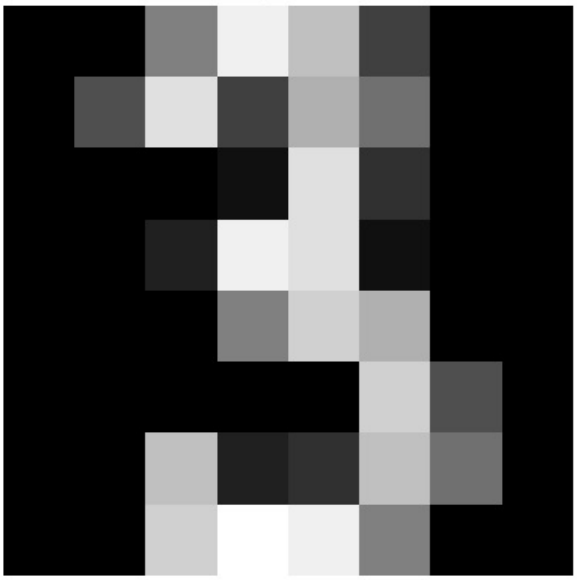
Actual: 3, Predicted: 3



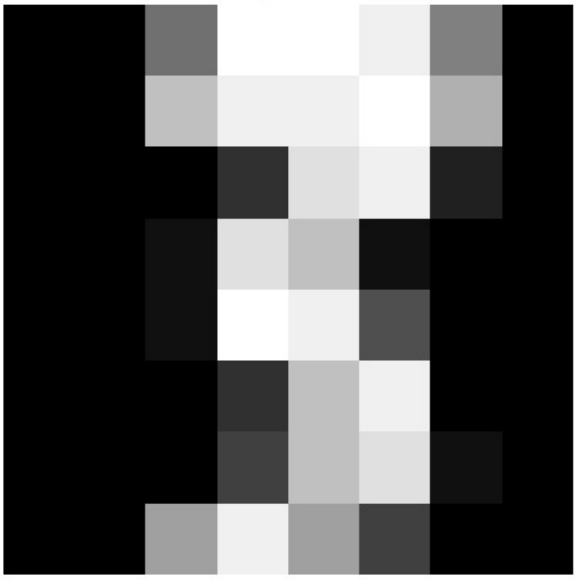
Actual: 6, Predicted: 6



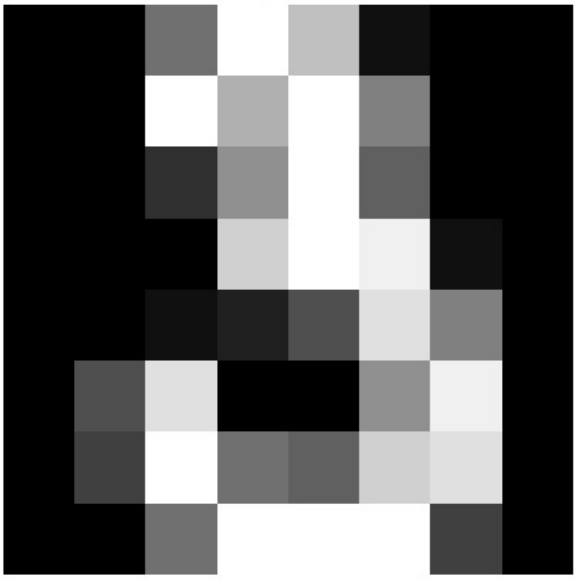
Actual: 3, Predicted: 3



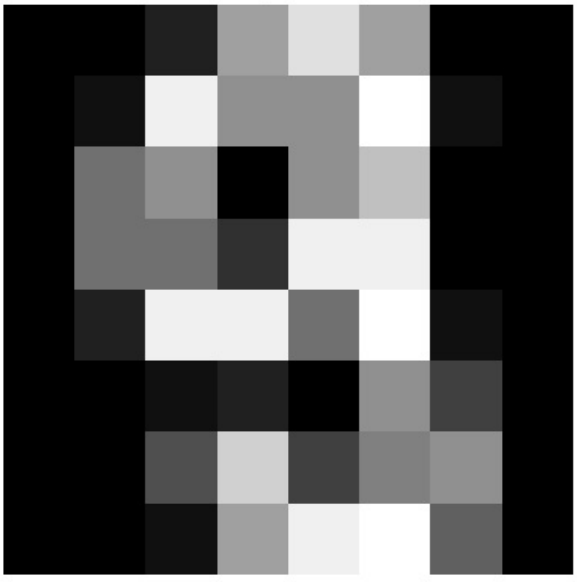
Actual: 3, Predicted: 3



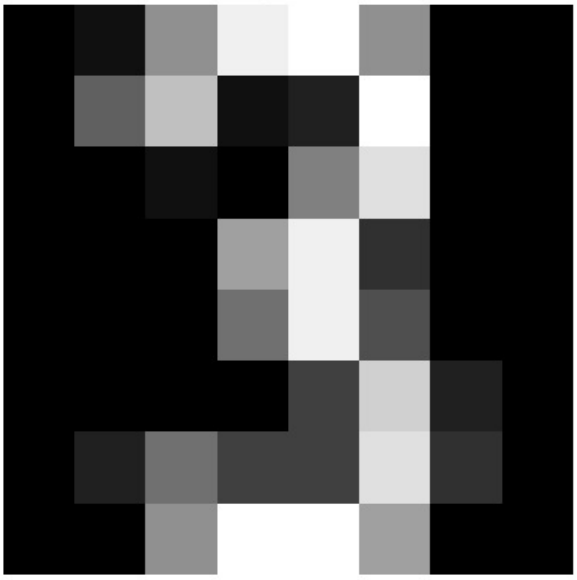
Actual: 3, Predicted: 3



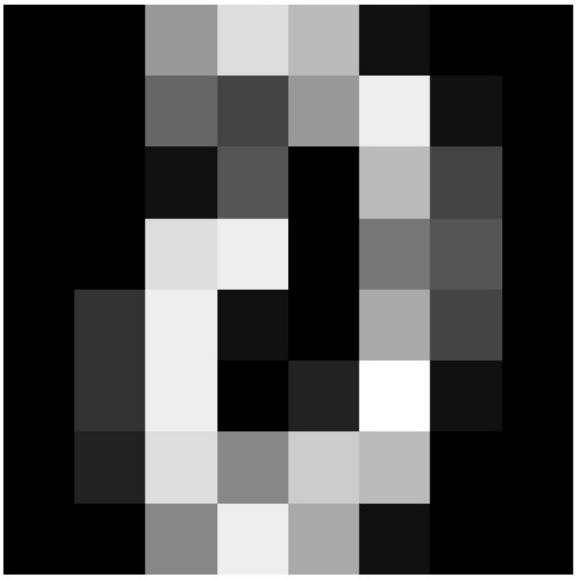
Actual: 9, Predicted: 9



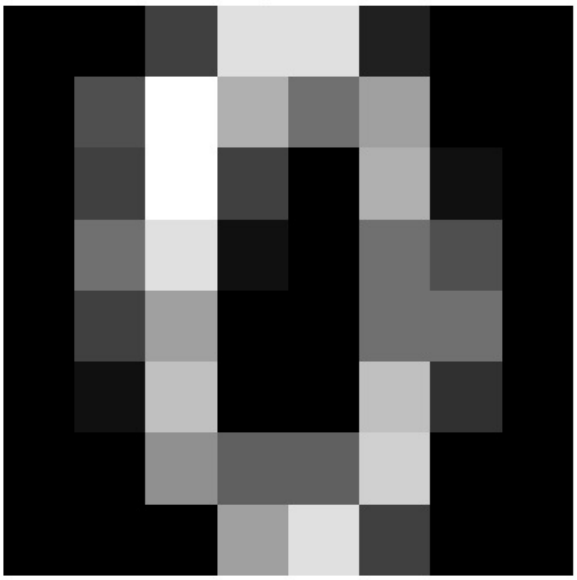
Actual: 3, Predicted: 3



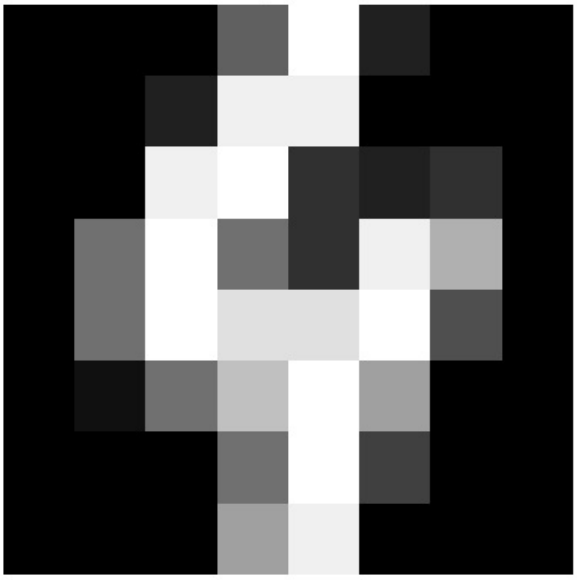
Actual: 0, Predicted: 0



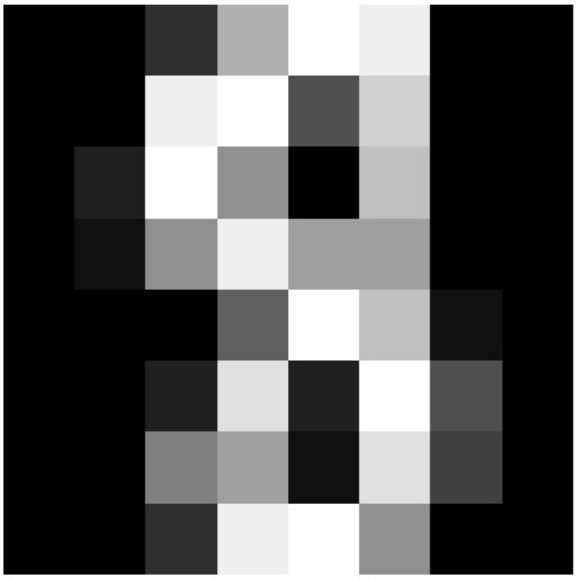
Actual: 0, Predicted: 0



Actual: 4, Predicted: 4

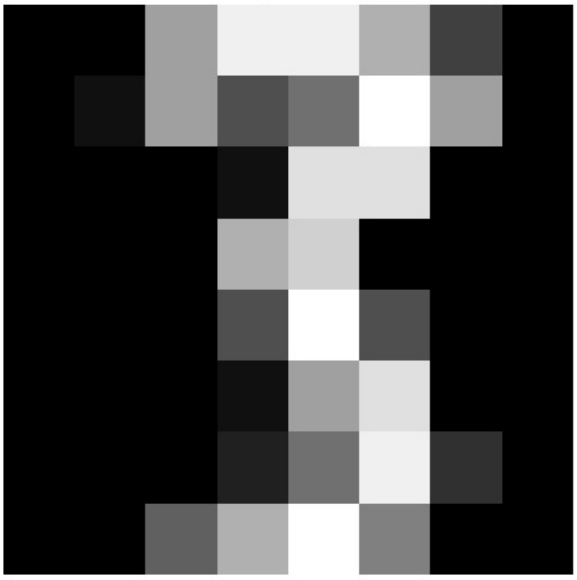


Actual: 8, Predicted: 8

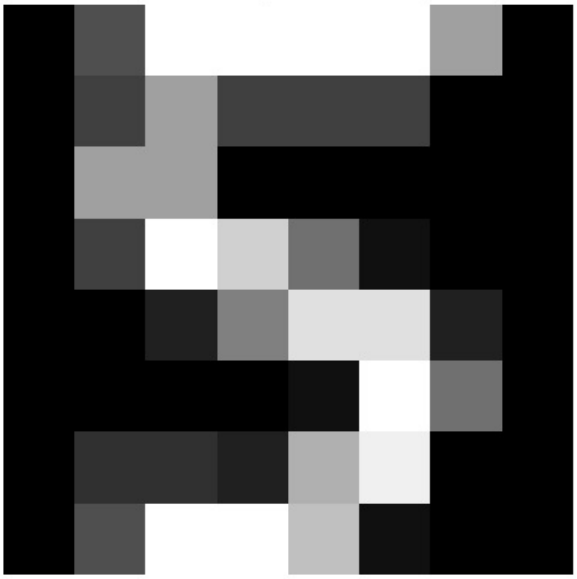


Actual: 0, Predicted: 0

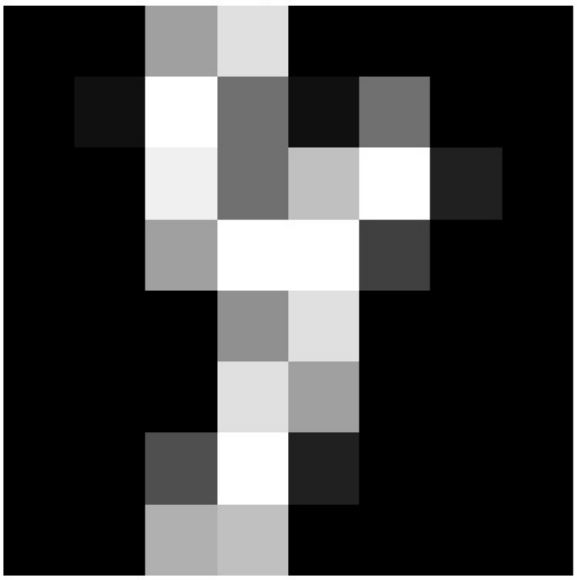
Actual: 3, Predicted: 3



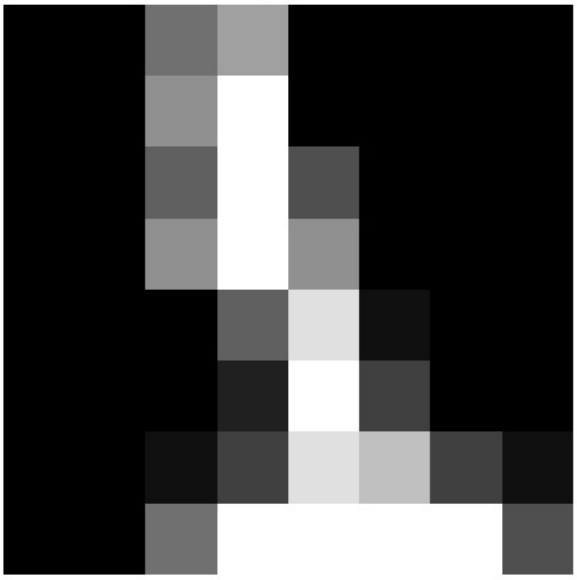
Actual: 5, Predicted: 5



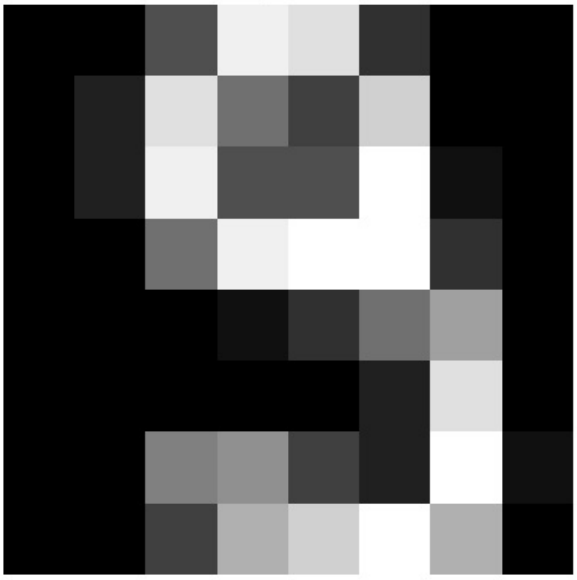
Actual: 4, Predicted: 4



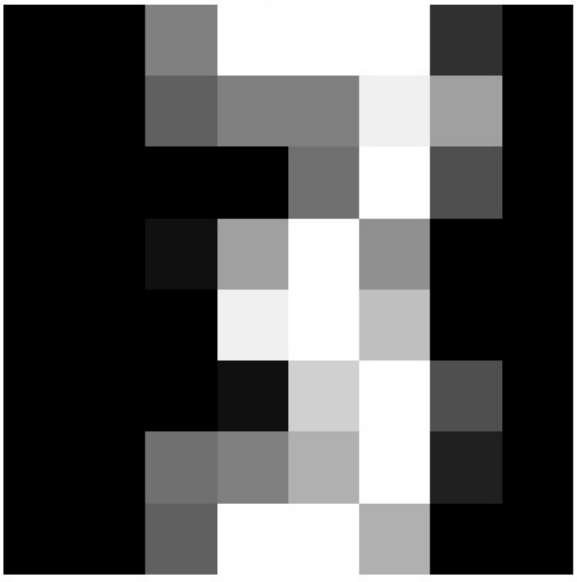
Actual: 1, Predicted: 1



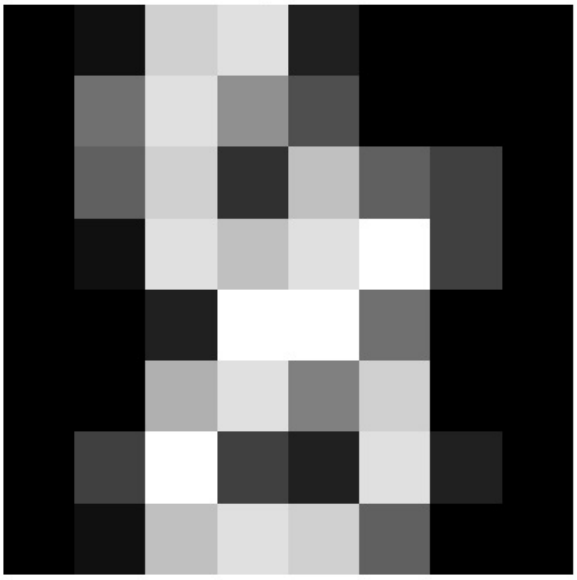
Actual: 9, Predicted: 9



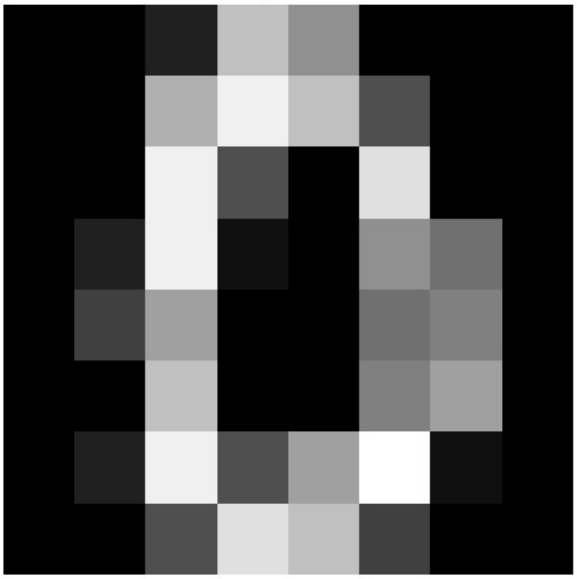
Actual: 3, Predicted: 3



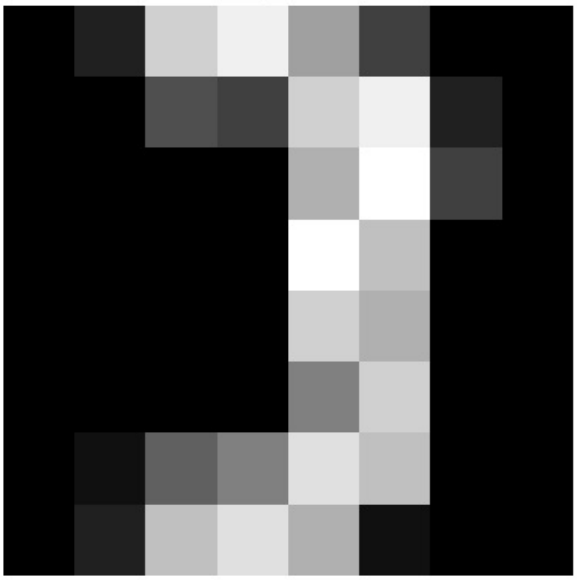
Actual: 8, Predicted: 8



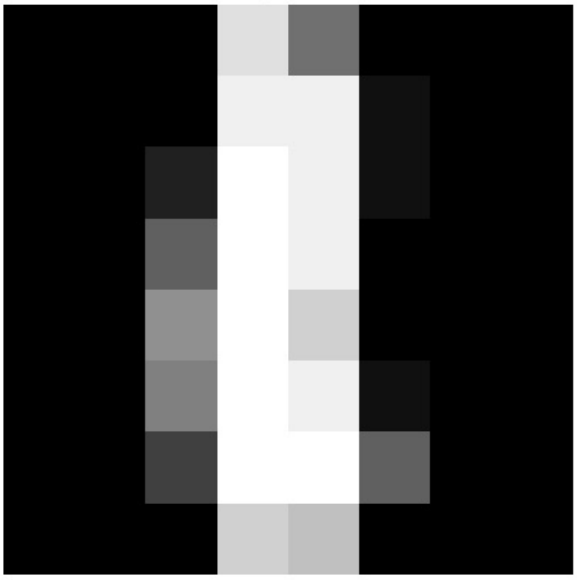
Actual: 0, Predicted: 0



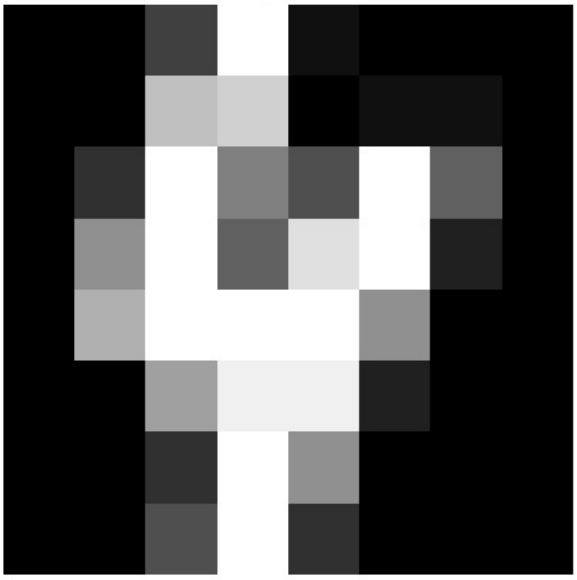
Actual: 3, Predicted: 3



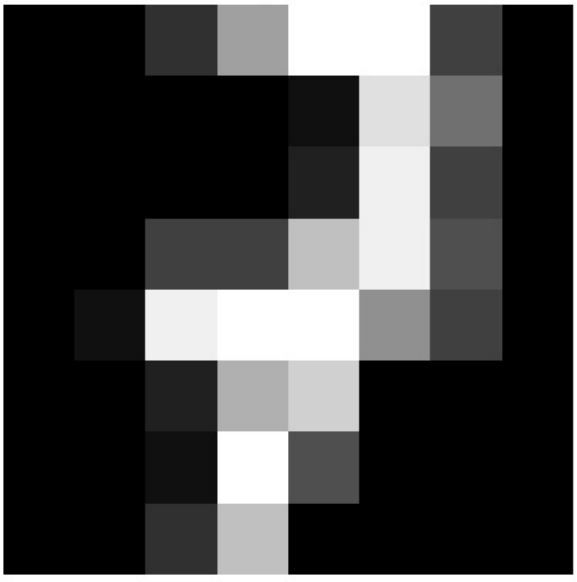
Actual: 1, Predicted: 1



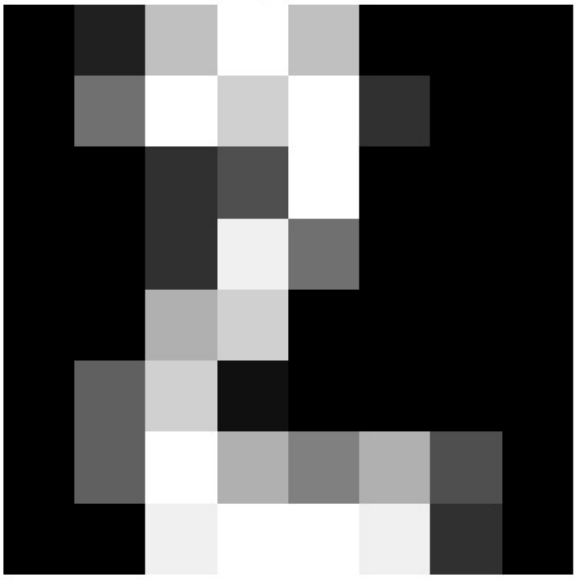
Actual: 4, Predicted: 4



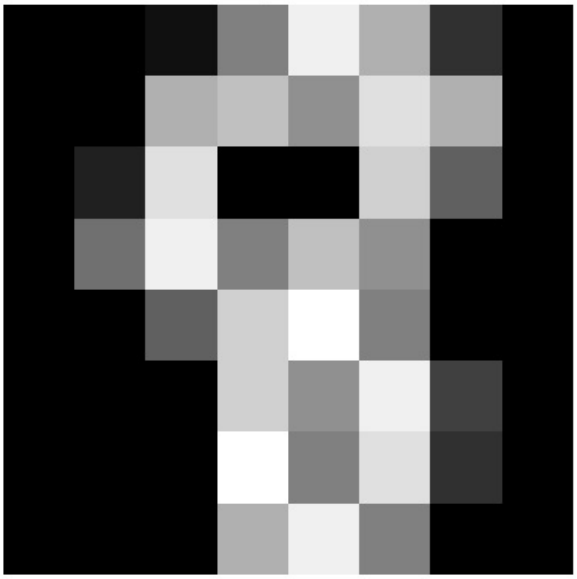
Actual: 7, Predicted: 7



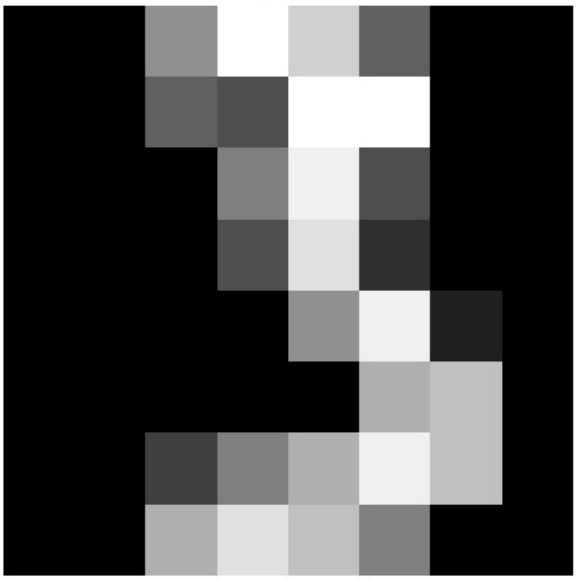
Actual: 2, Predicted: 2



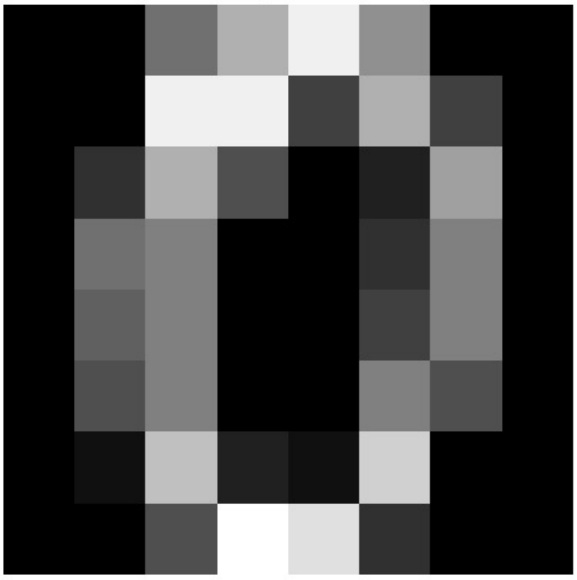
Actual: 8, Predicted: 8



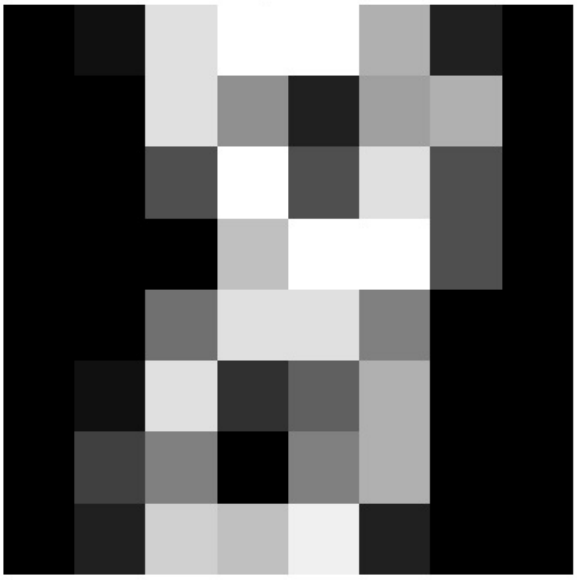
Actual: 3, Predicted: 3



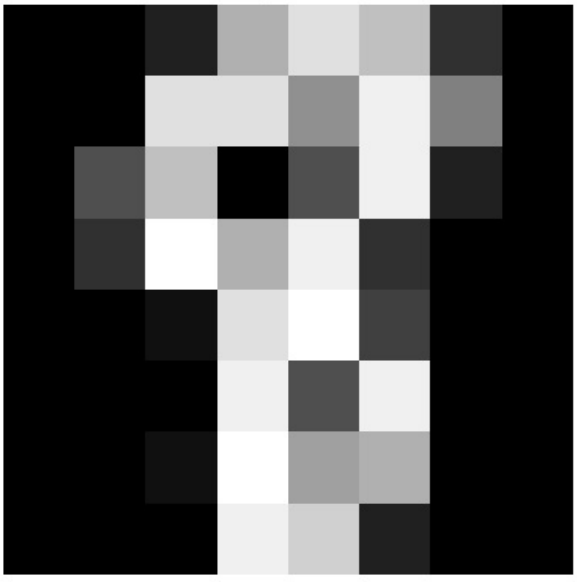
Actual: 0, Predicted: 0



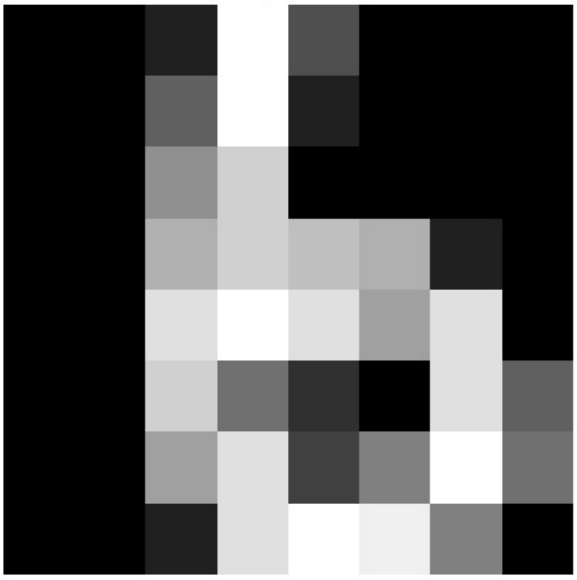
Actual: 8, Predicted: 8



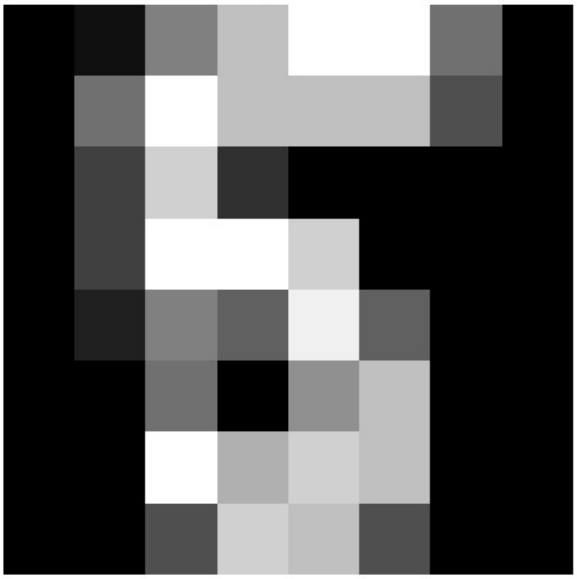
Actual: 8, Predicted: 8



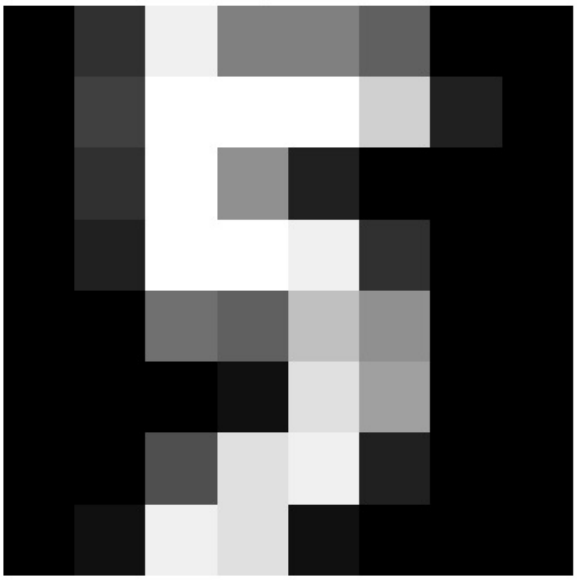
Actual: 6, Predicted: 6



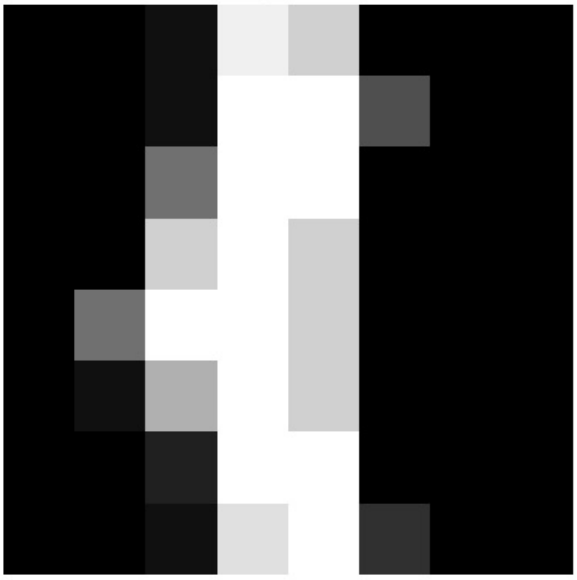
Actual: 5, Predicted: 5



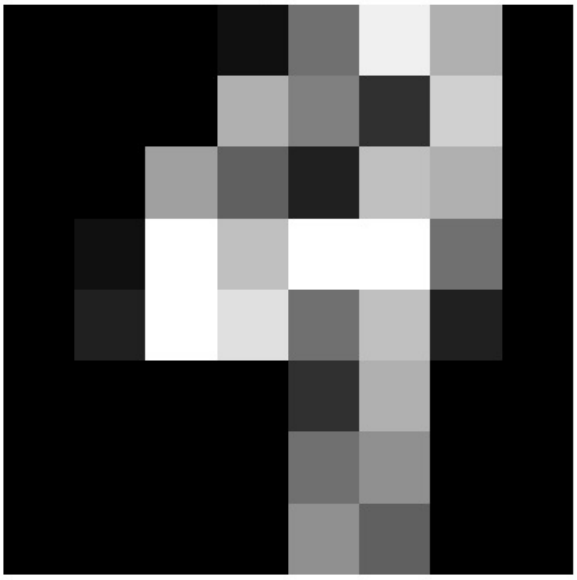
Actual: 5, Predicted: 5



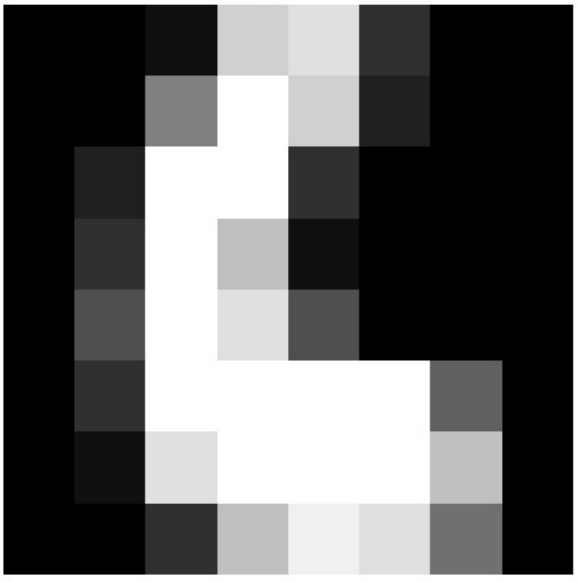
Actual: 1, Predicted: 1



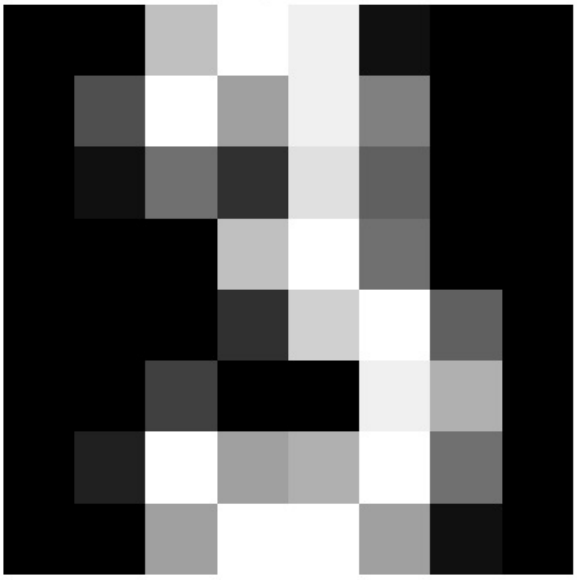
Actual: 9, Predicted: 9



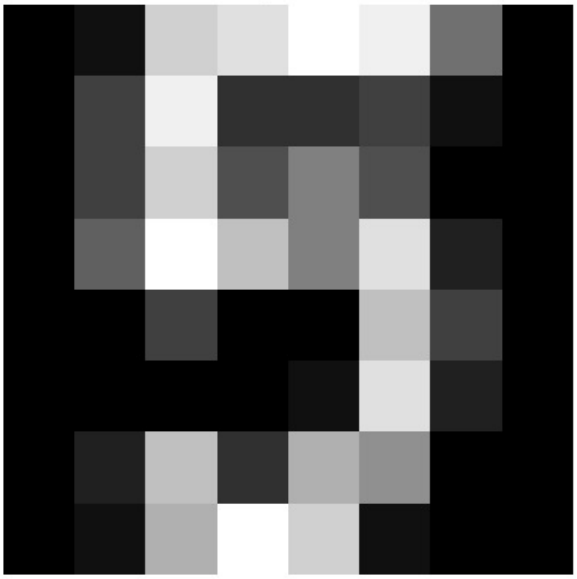
Actual: 6, Predicted: 6



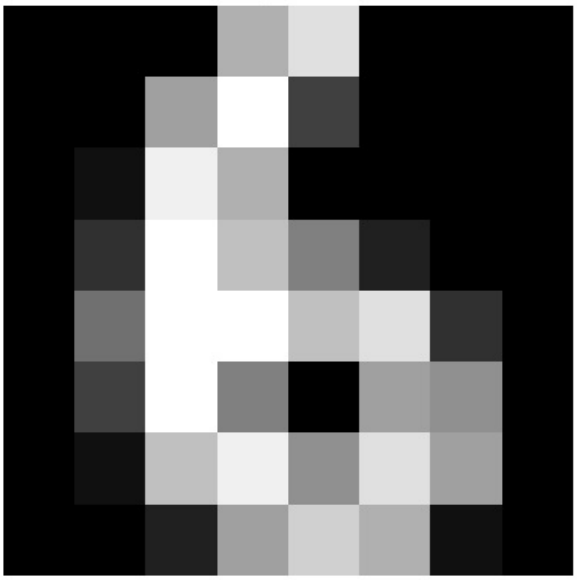
Actual: 3, Predicted: 3



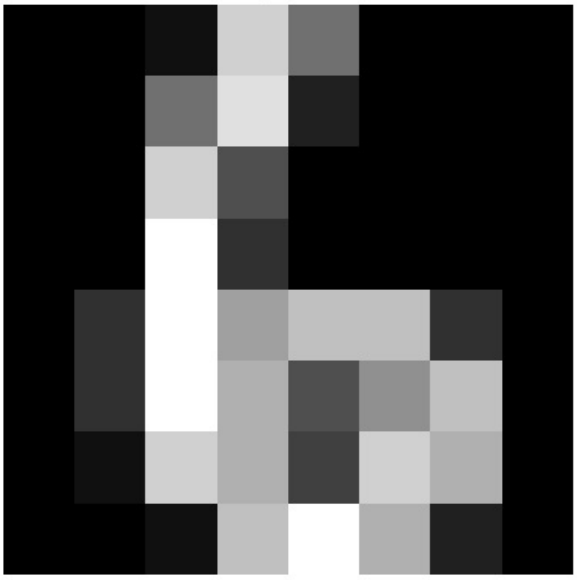
Actual: 5, Predicted: 5



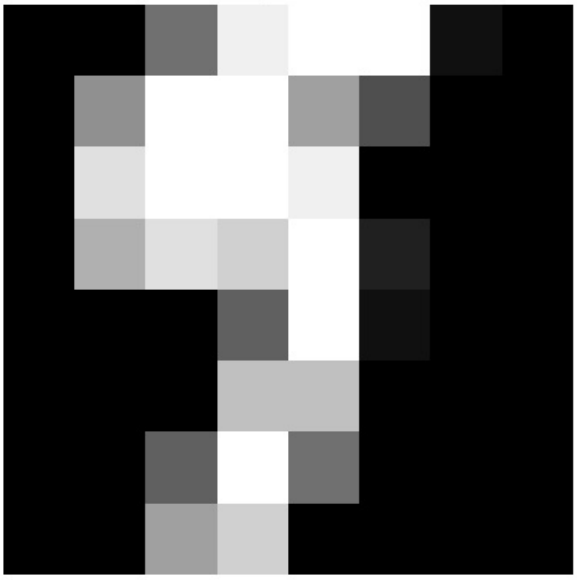
Actual: 6, Predicted: 6



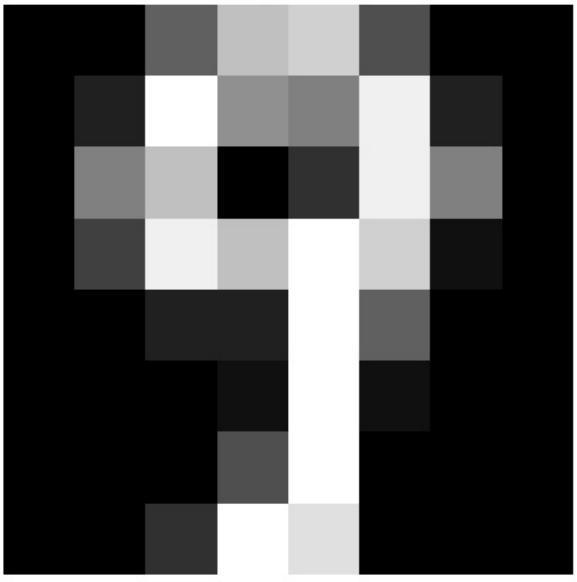
Actual: 6, Predicted: 6



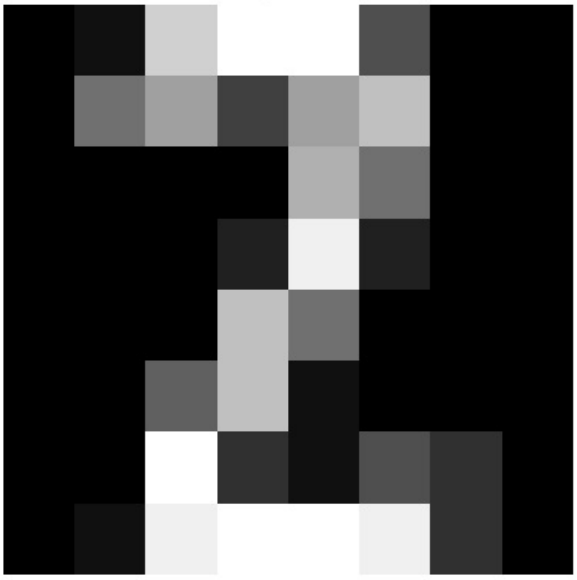
Actual: 5, Predicted: 5



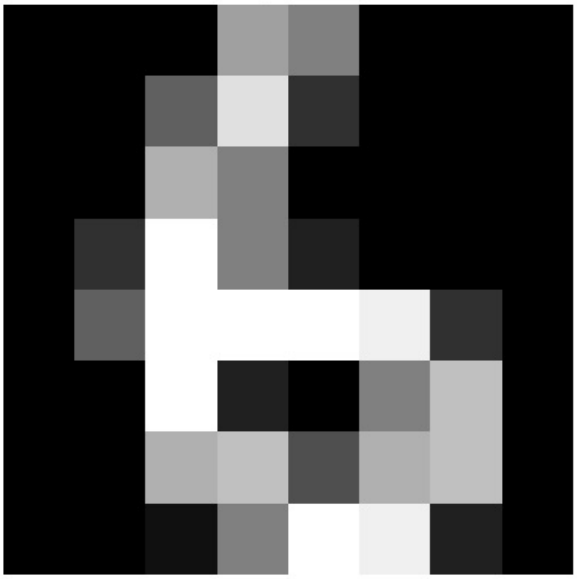
Actual: 9, Predicted: 9



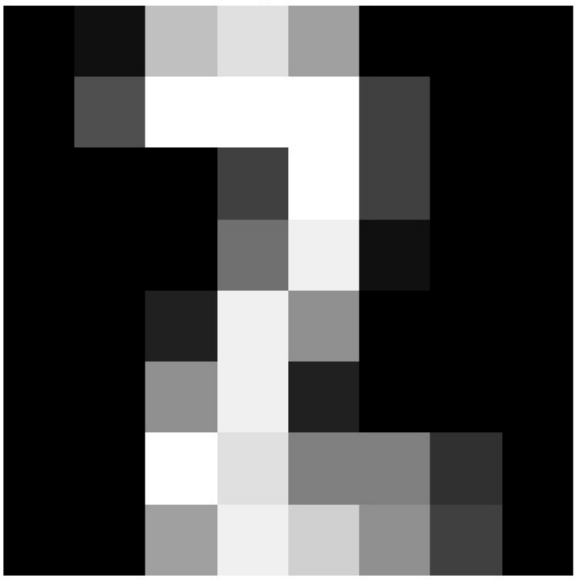
Actual: 2, Predicted: 2



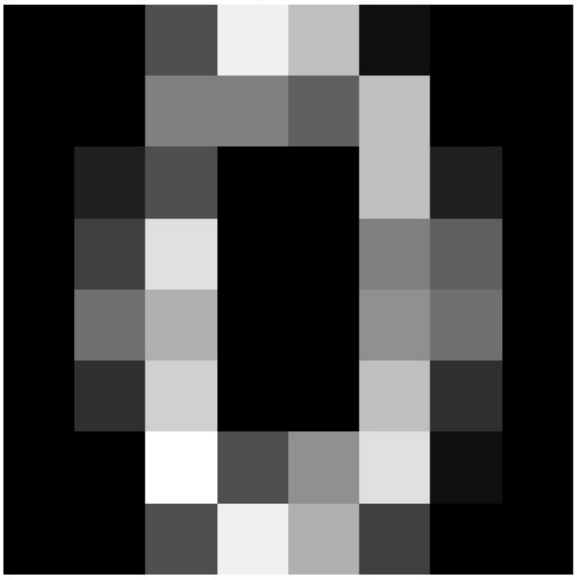
Actual: 6, Predicted: 6



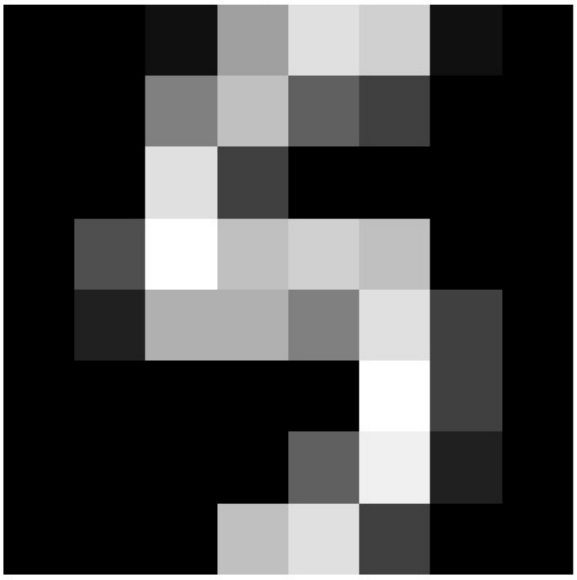
Actual: 2, Predicted: 2



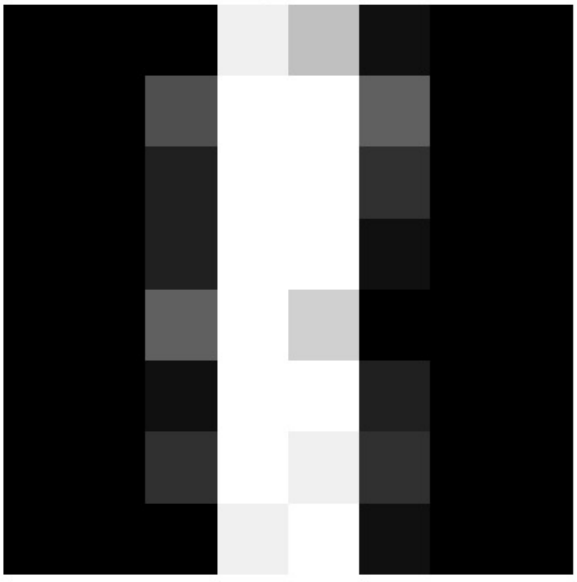
Actual: 0, Predicted: 0



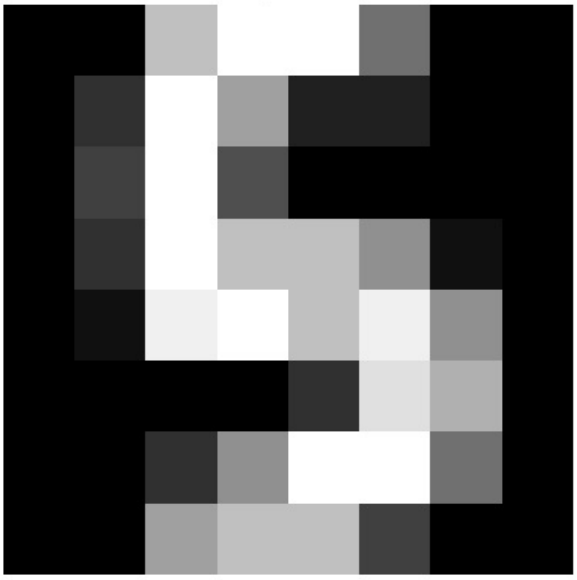
Actual: 5, Predicted: 5



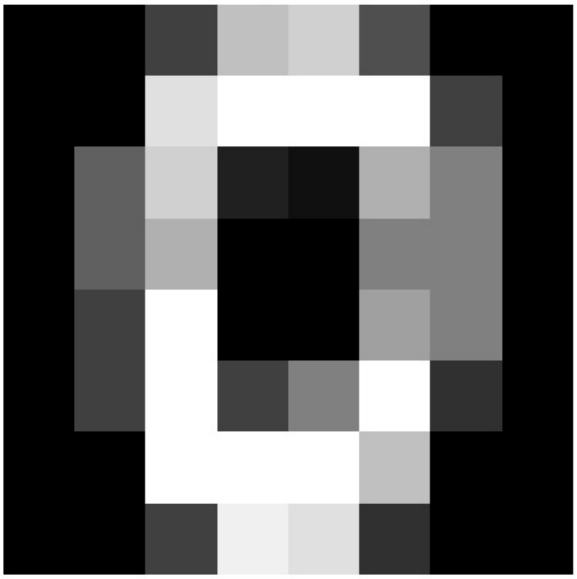
Actual: 1, Predicted: 1



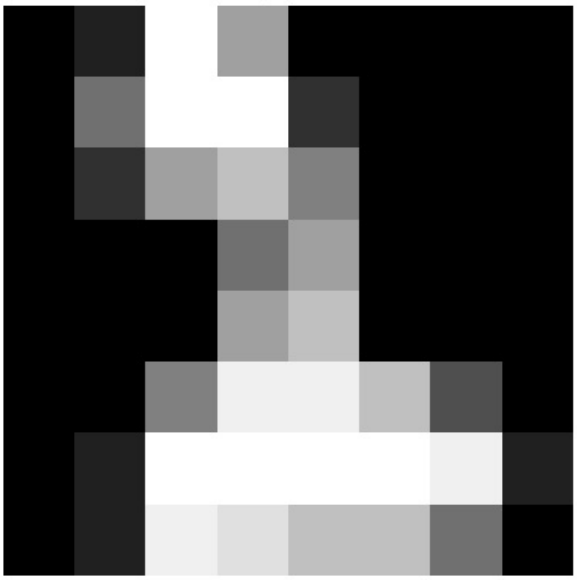
Actual: 5, Predicted: 5



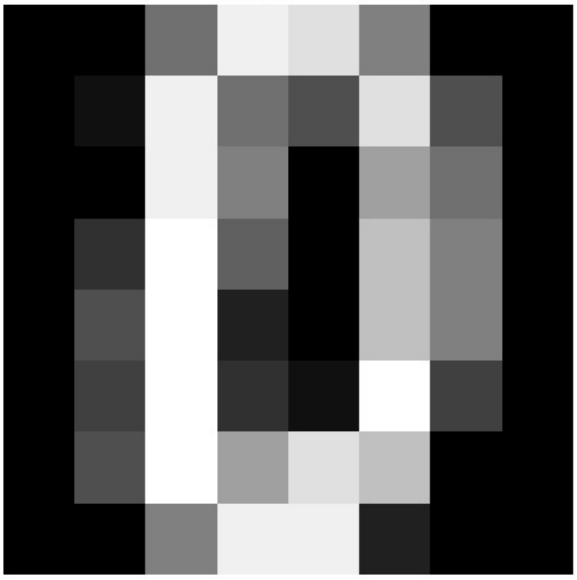
Actual: 0, Predicted: 0



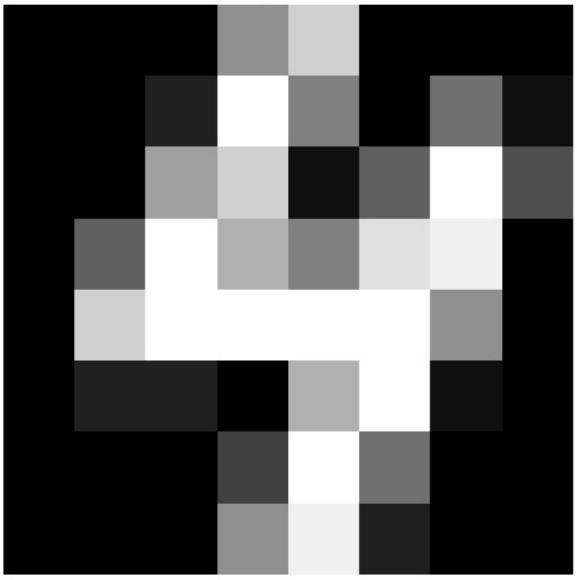
Actual: 2, Predicted: 2



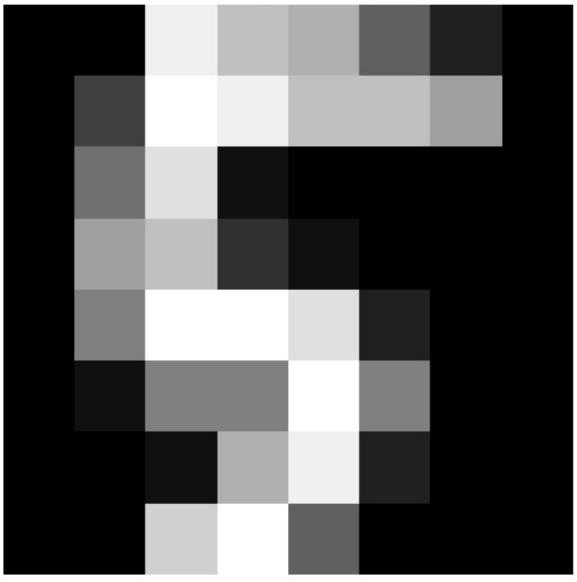
Actual: 0, Predicted: 0



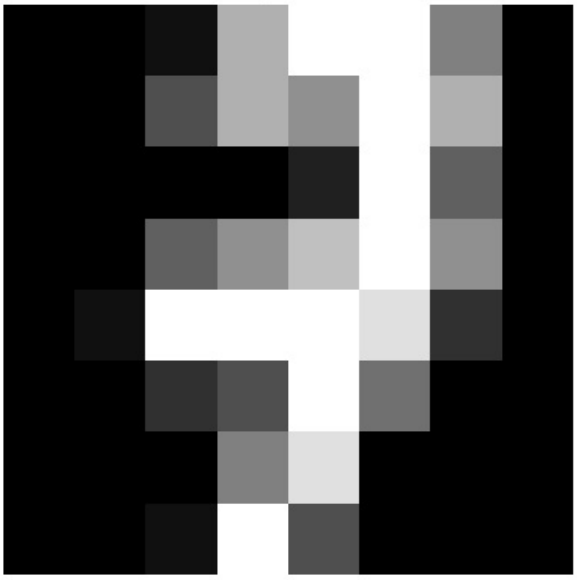
Actual: 4, Predicted: 4



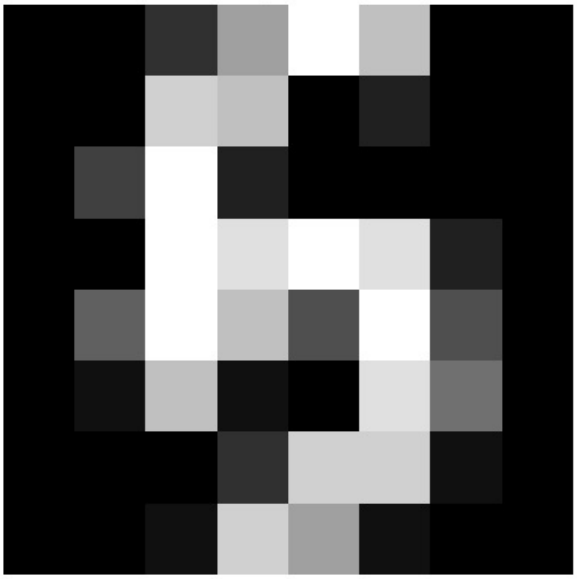
Actual: 5, Predicted: 5



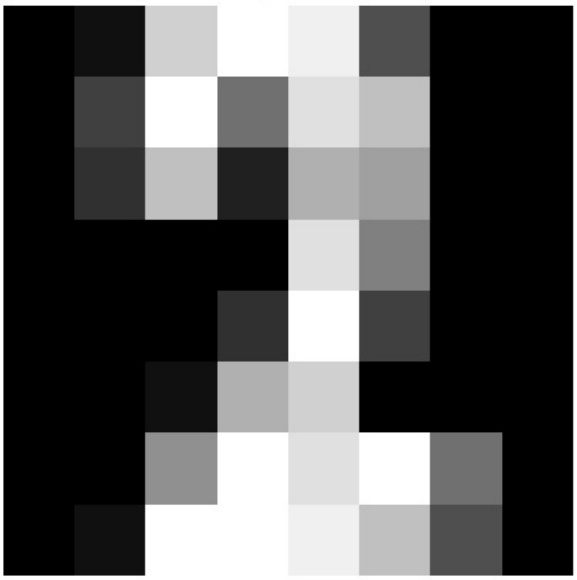
Actual: 7, Predicted: 7



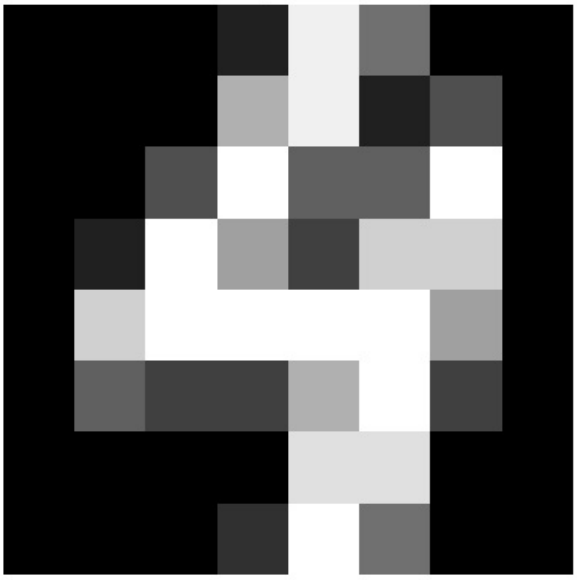
Actual: 5, Predicted: 5



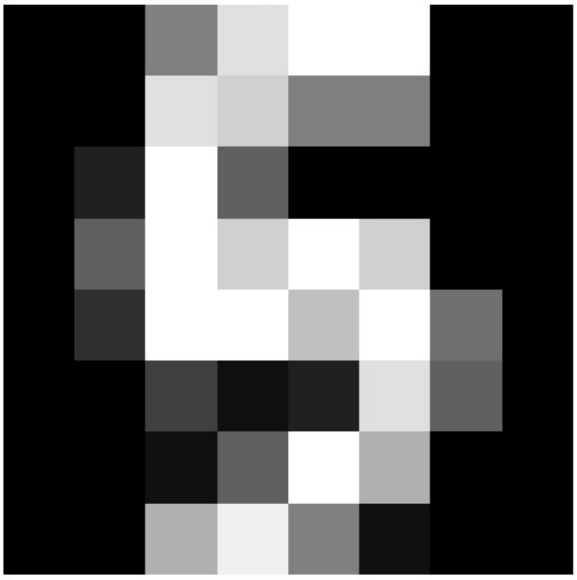
Actual: 2, Predicted: 2



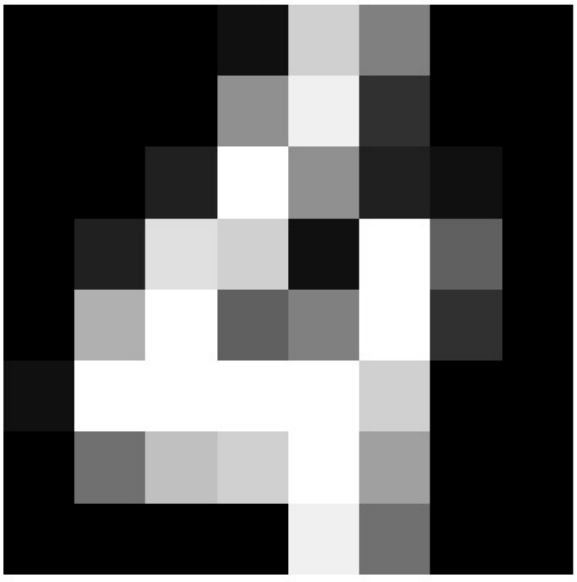
Actual: 4, Predicted: 4



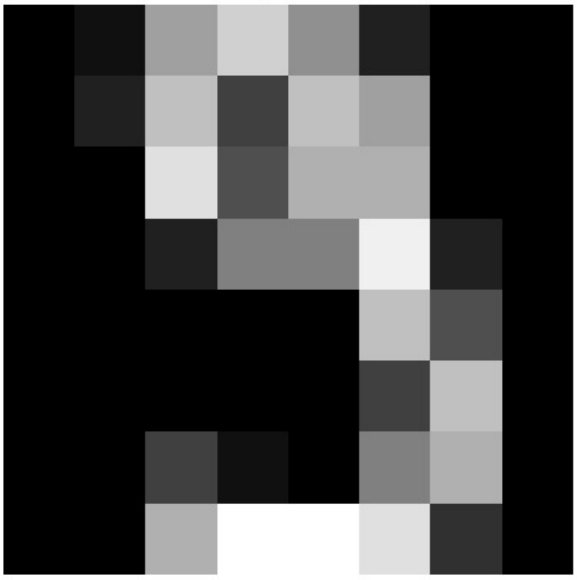
Actual: 5, Predicted: 5



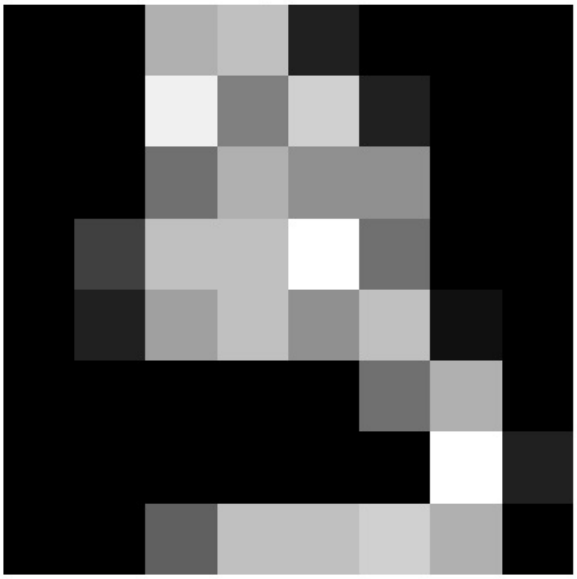
Actual: 4, Predicted: 4



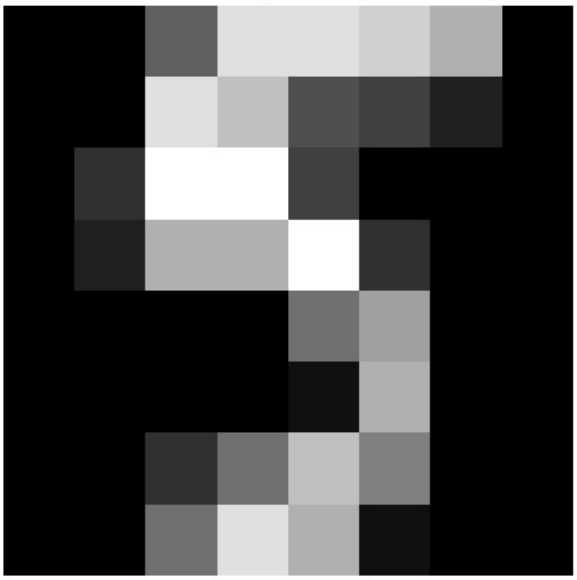
Actual: 9, Predicted: 9



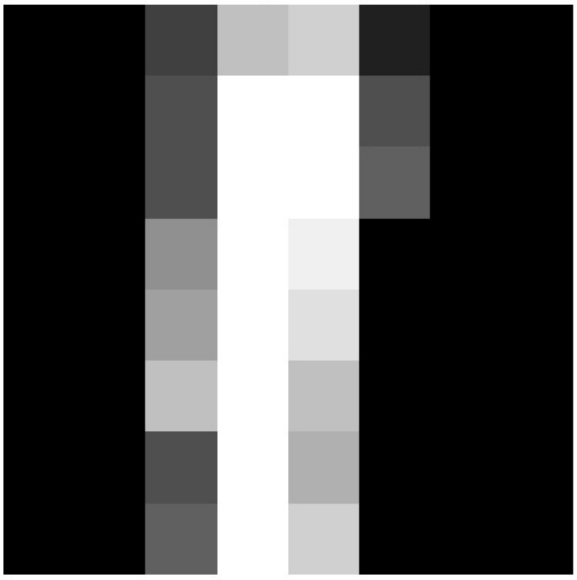
Actual: 9, Predicted: 9



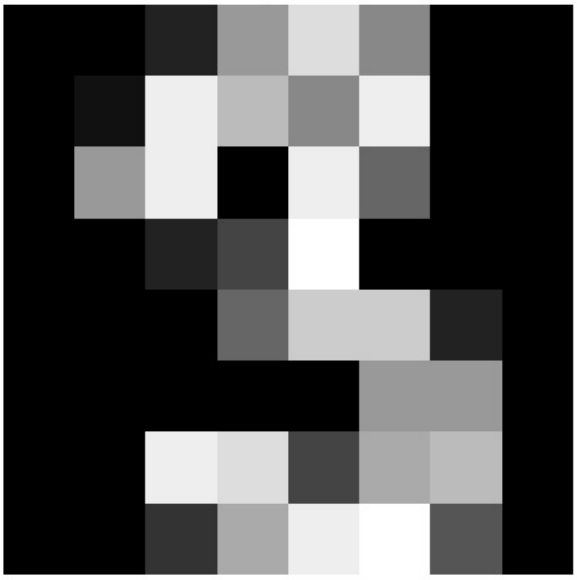
Actual: 5, Predicted: 5



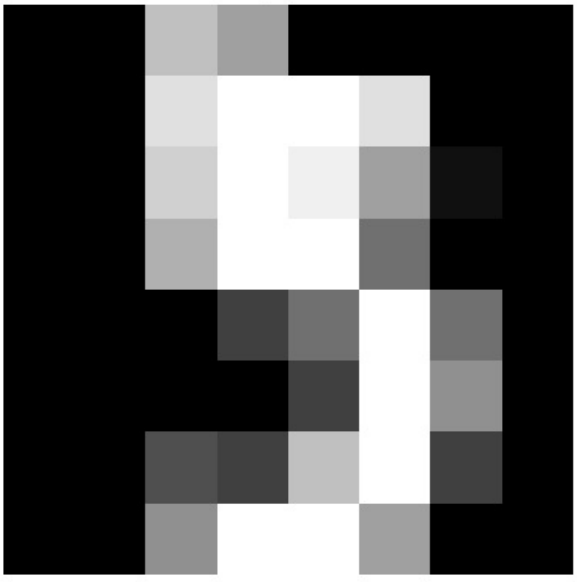
Actual: 1, Predicted: 1



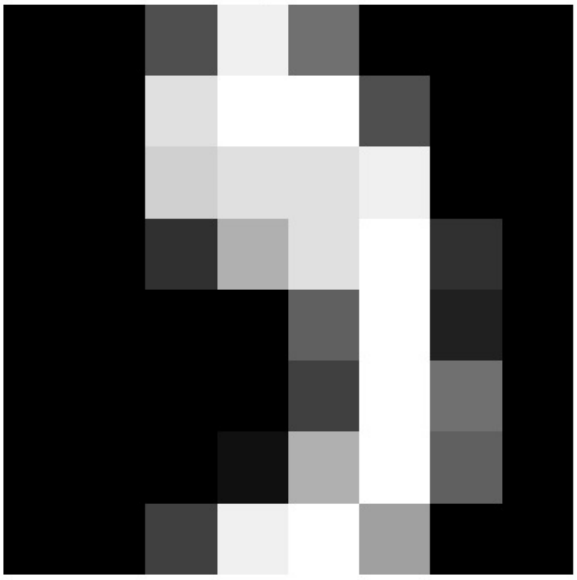
Actual: 3, Predicted: 3



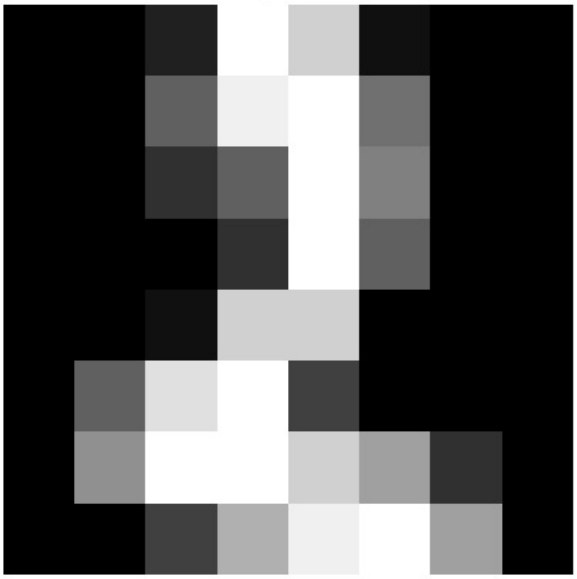
Actual: 5, Predicted: 9



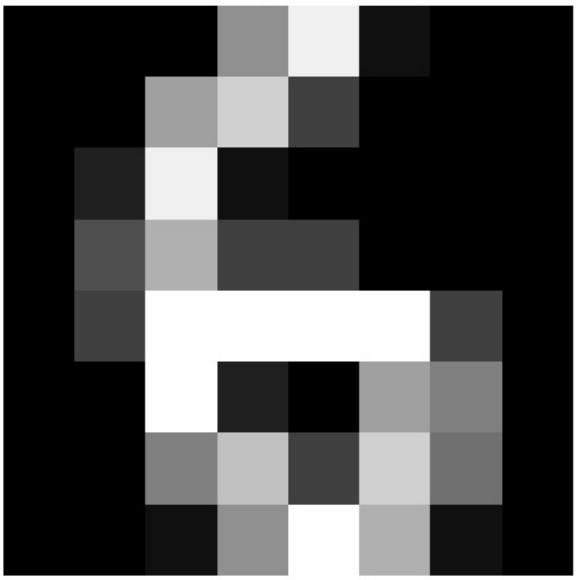
Actual: 9, Predicted: 9



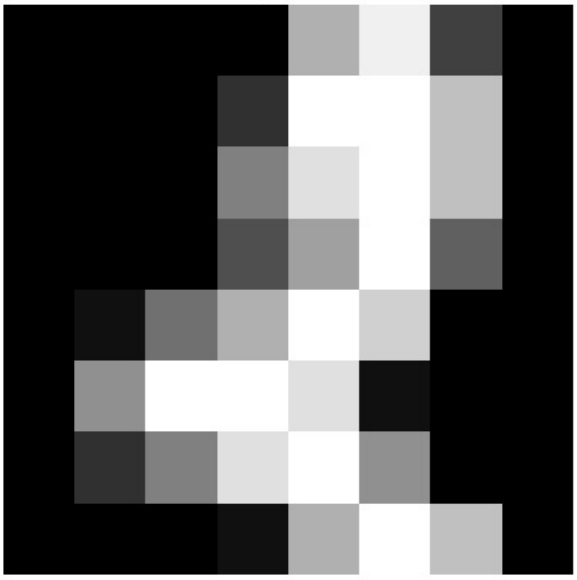
Actual: 2, Predicted: 2



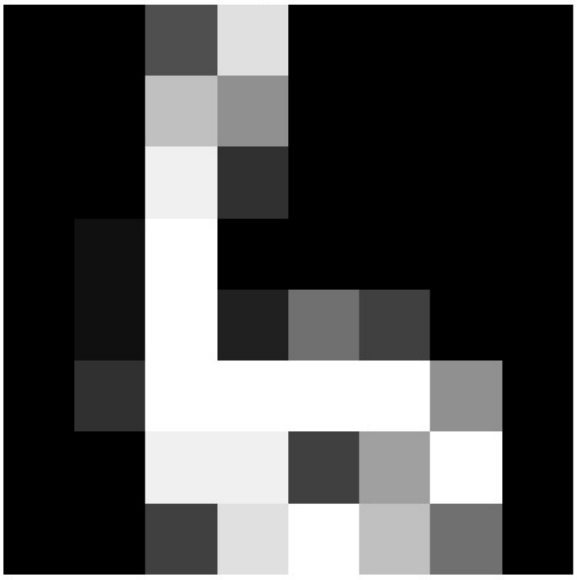
Actual: 6, Predicted: 6



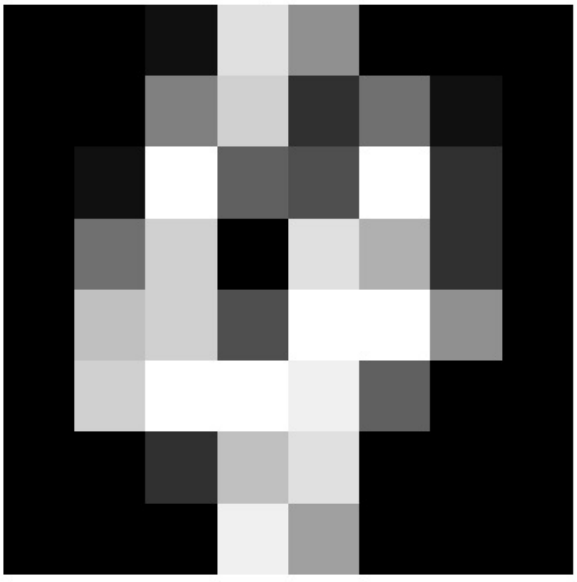
Actual: 2, Predicted: 2



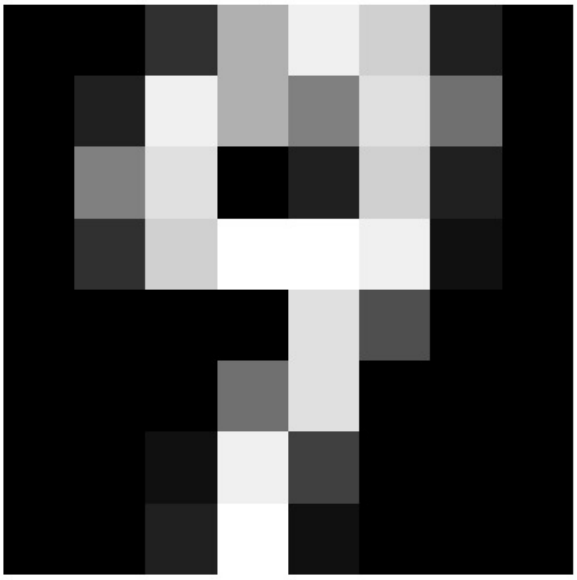
Actual: 6, Predicted: 6



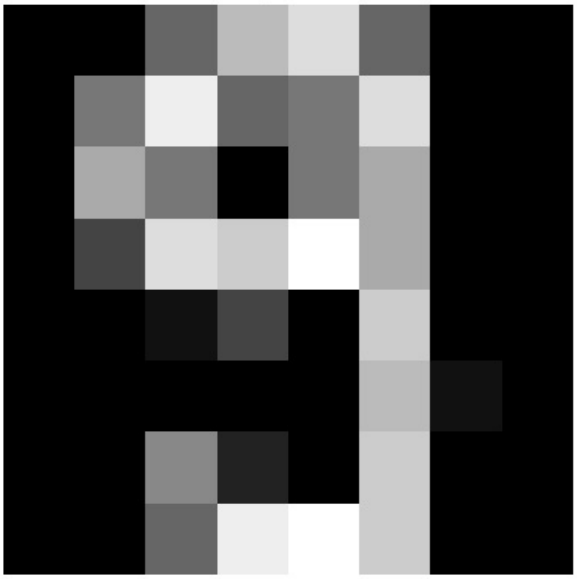
Actual: 4, Predicted: 4



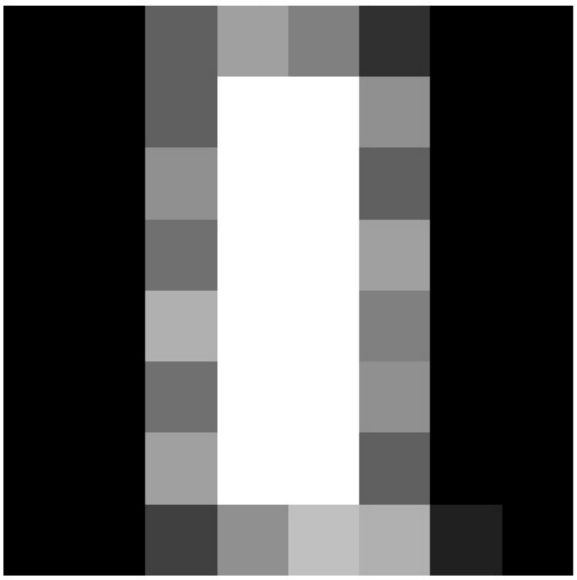
Actual: 9, Predicted: 9



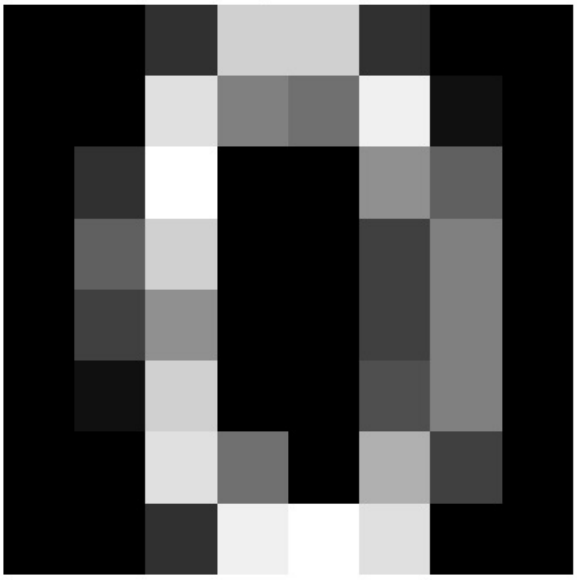
Actual: 9, Predicted: 9



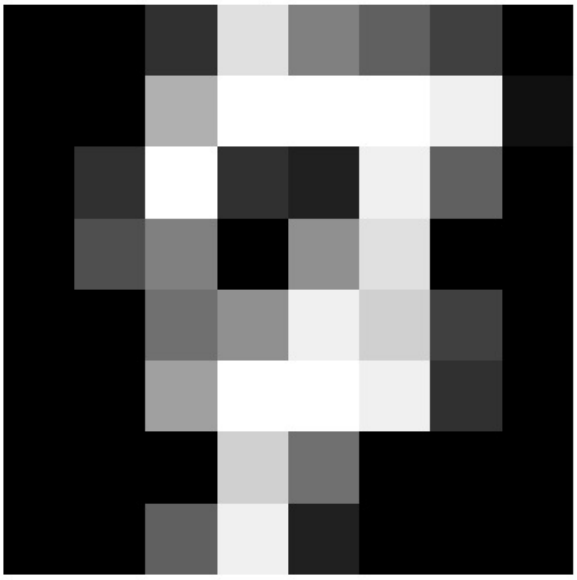
Actual: 1, Predicted: 1



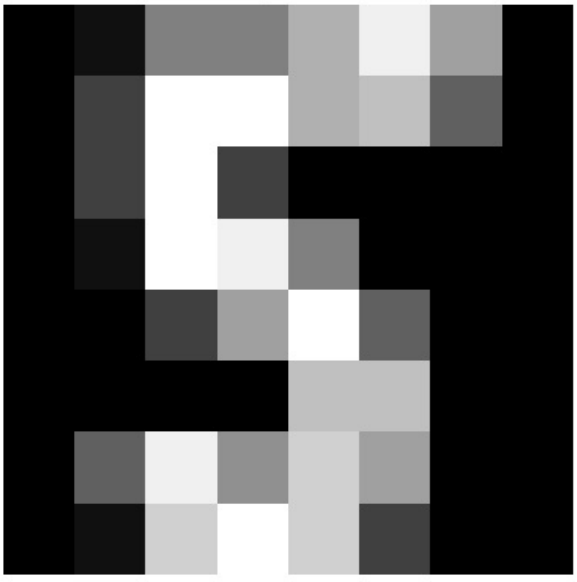
Actual: 0, Predicted: 0



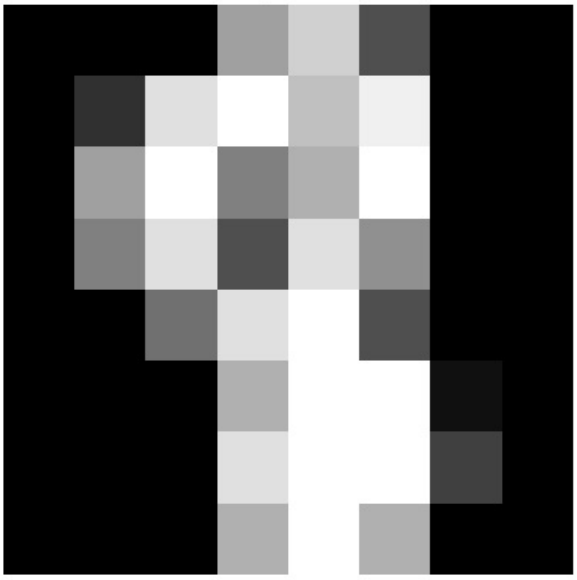
Actual: 7, Predicted: 7



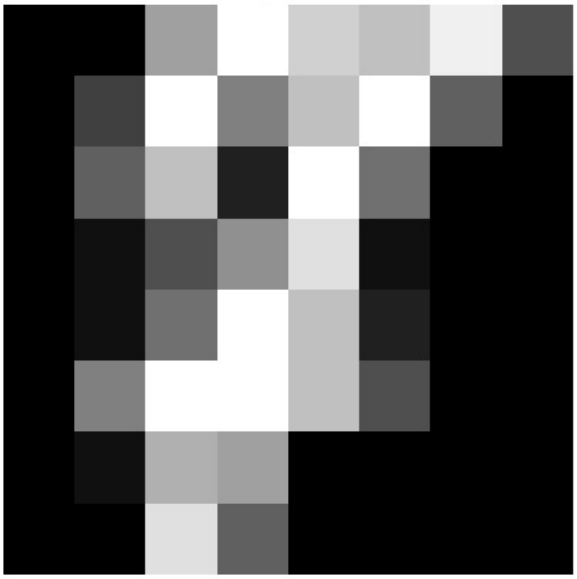
Actual: 5, Predicted: 5



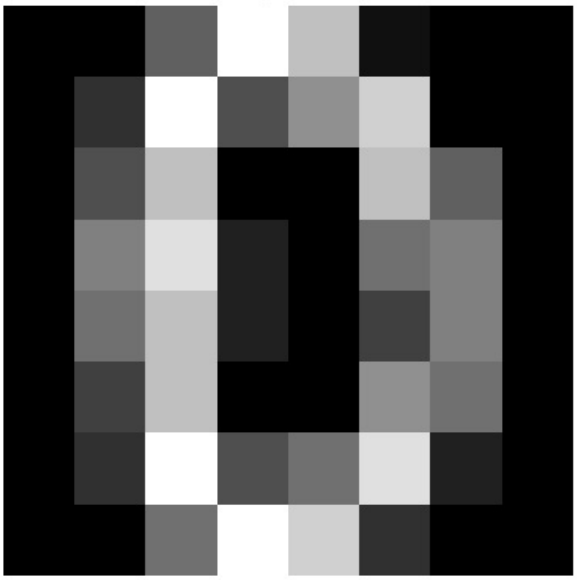
Actual: 8, Predicted: 8



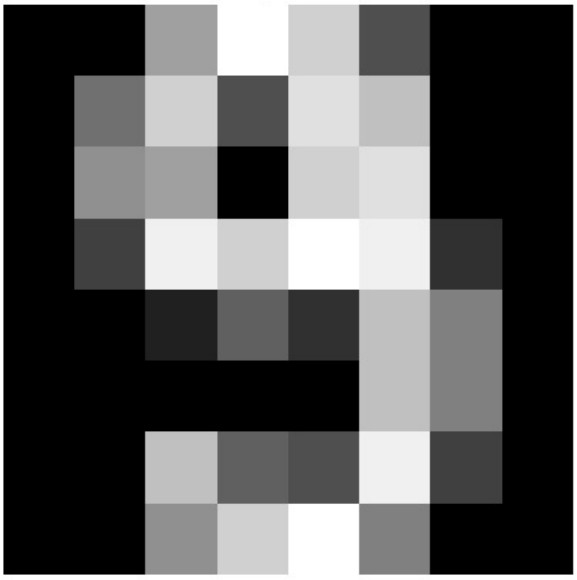
Actual: 7, Predicted: 7



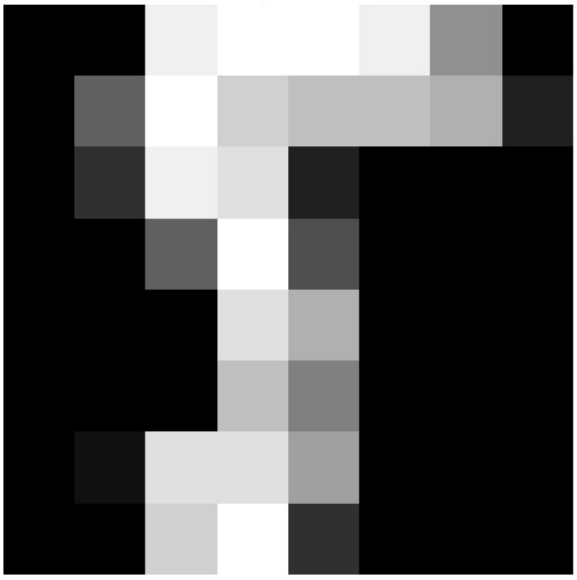
Actual: 0, Predicted: 0



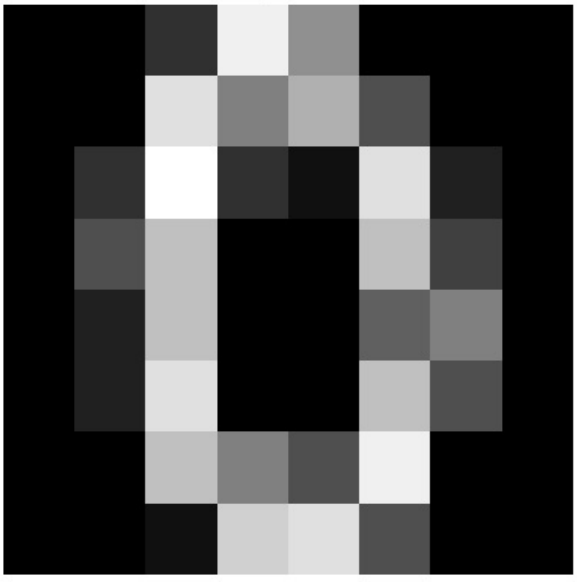
Actual: 9, Predicted: 9



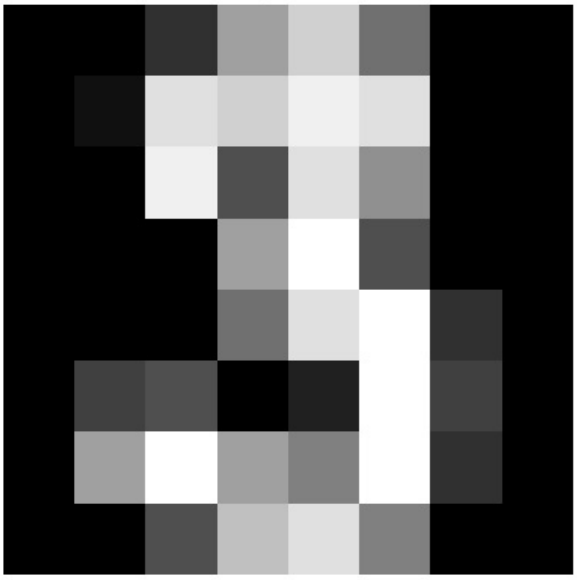
Actual: 5, Predicted: 5



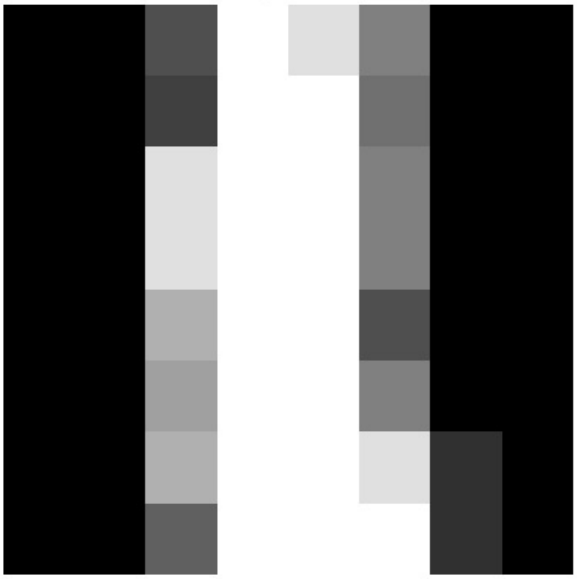
Actual: 0, Predicted: 0



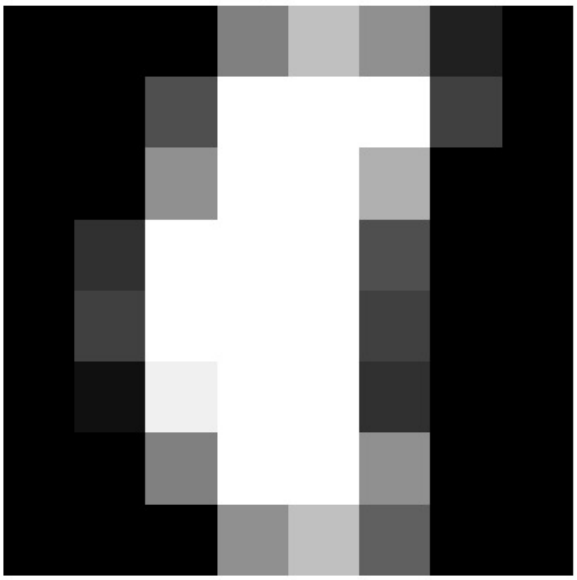
Actual: 3, Predicted: 3



Actual: 1, Predicted: 1

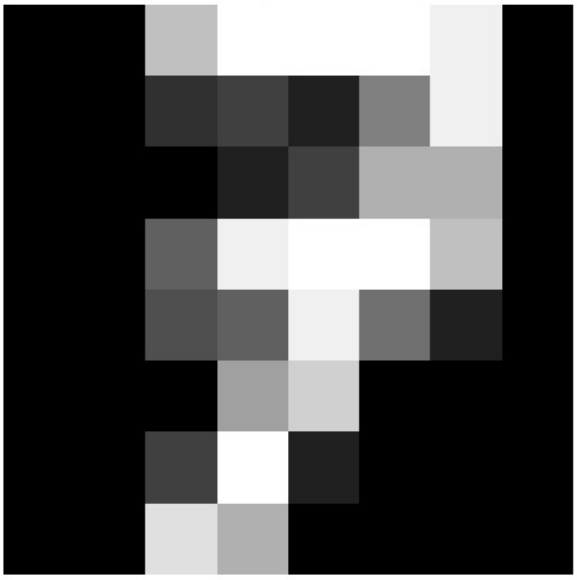


Actual: 1, Predicted: 1

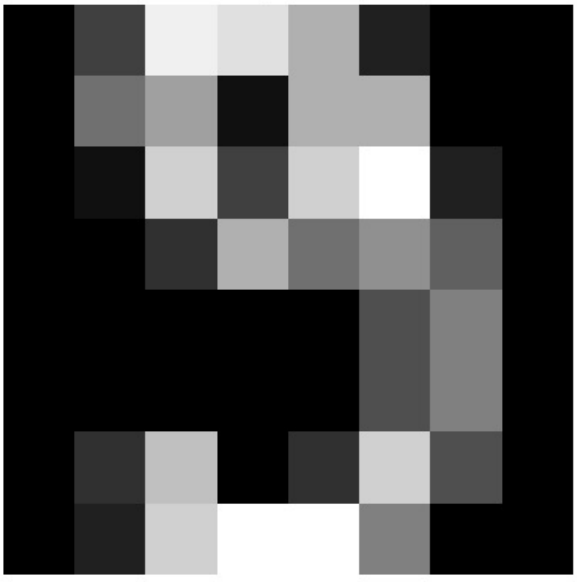


Actual: 3, Predicted: 3

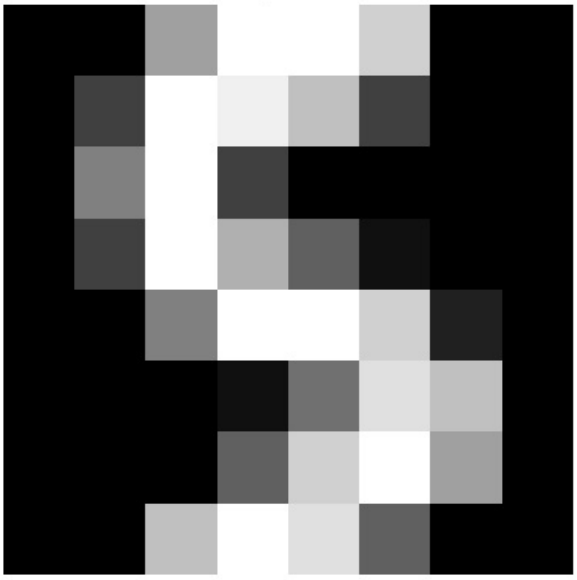
Actual: 7, Predicted: 7



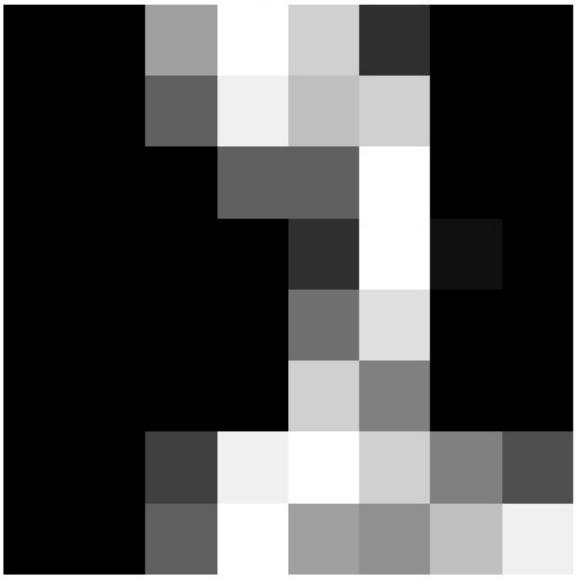
Actual: 9, Predicted: 9



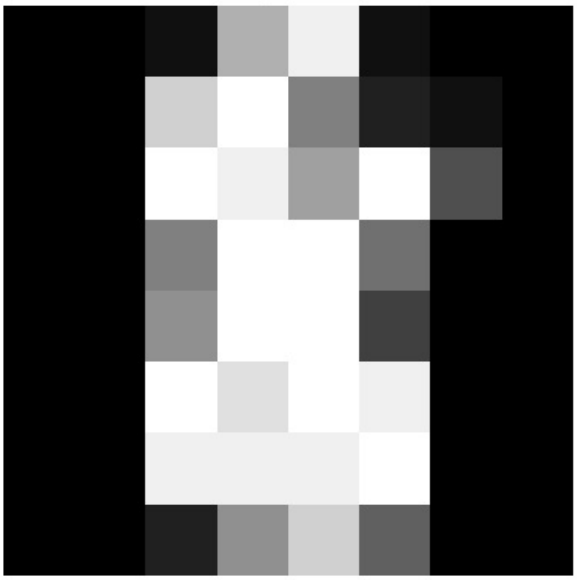
Actual: 5, Predicted: 5



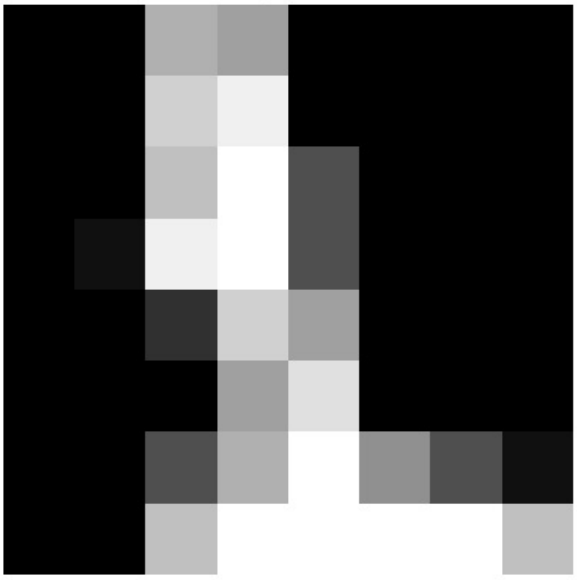
Actual: 2, Predicted: 2



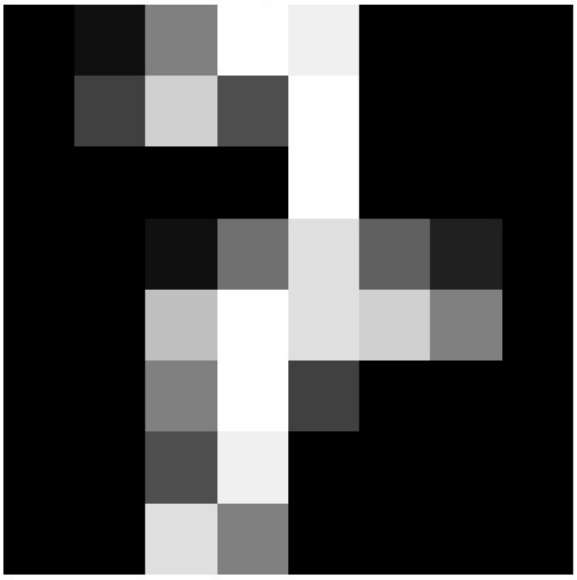
Actual: 8, Predicted: 8



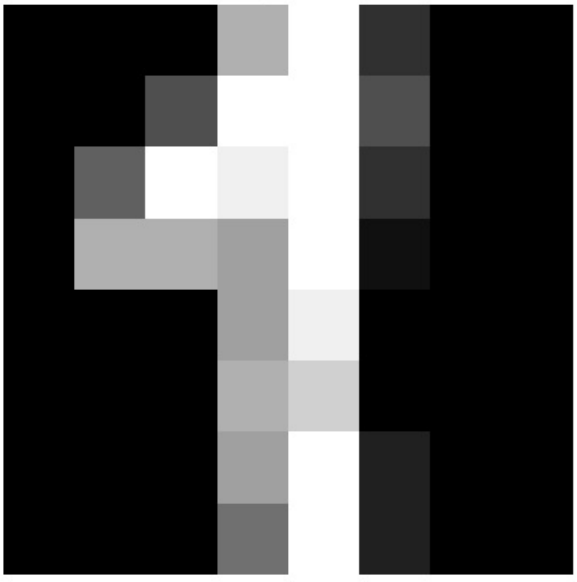
Actual: 1, Predicted: 1



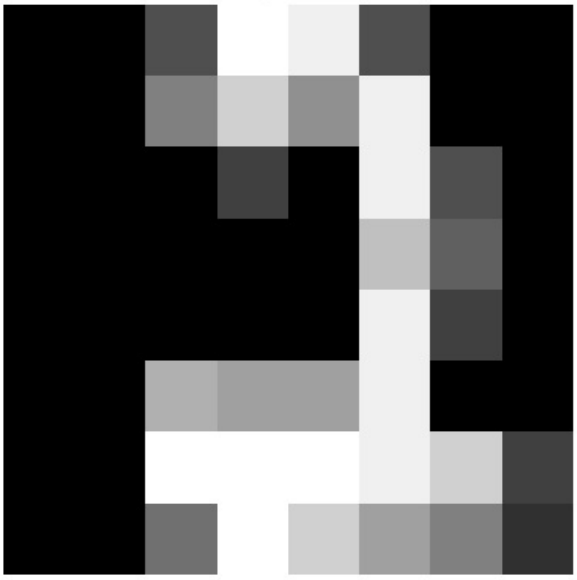
Actual: 7, Predicted: 7



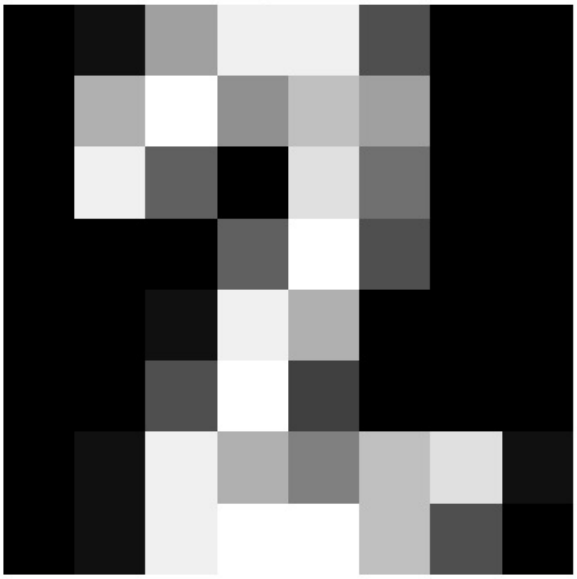
Actual: 1, Predicted: 1



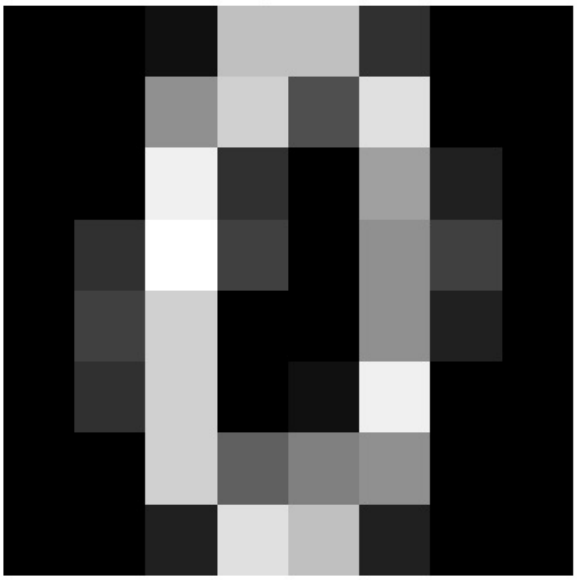
Actual: 2, Predicted: 2



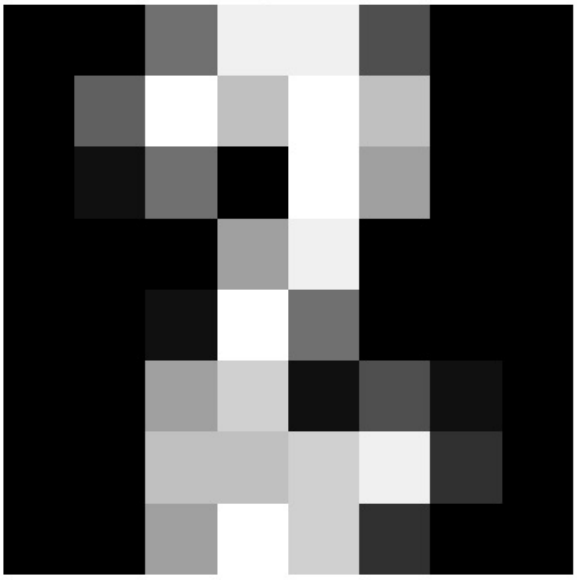
Actual: 2, Predicted: 2



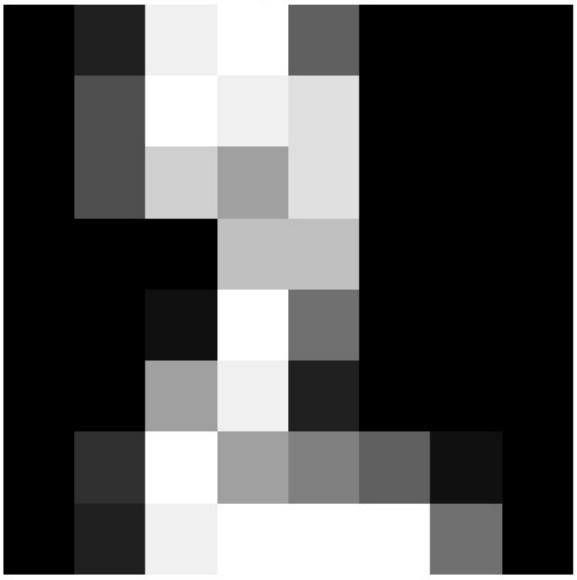
Actual: 0, Predicted: 0



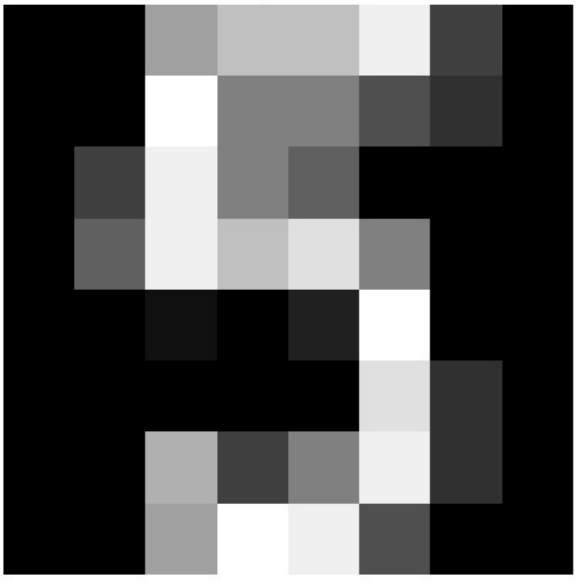
Actual: 2, Predicted: 2



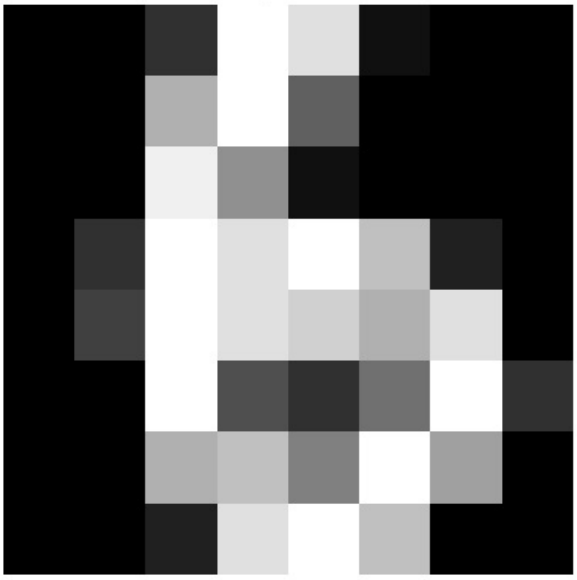
Actual: 2, Predicted: 2



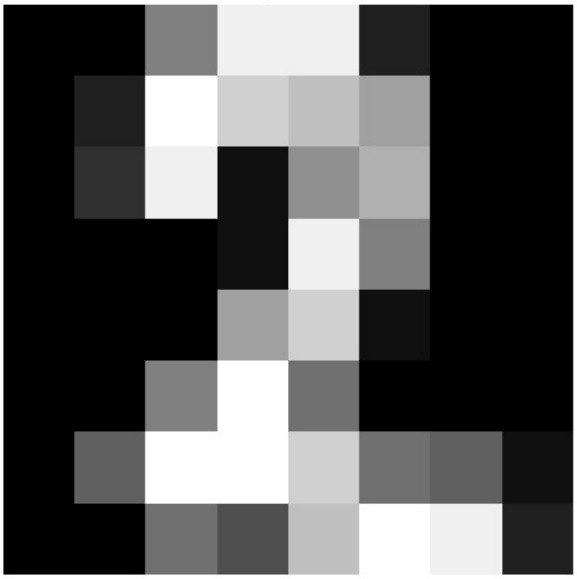
Actual: 5, Predicted: 5



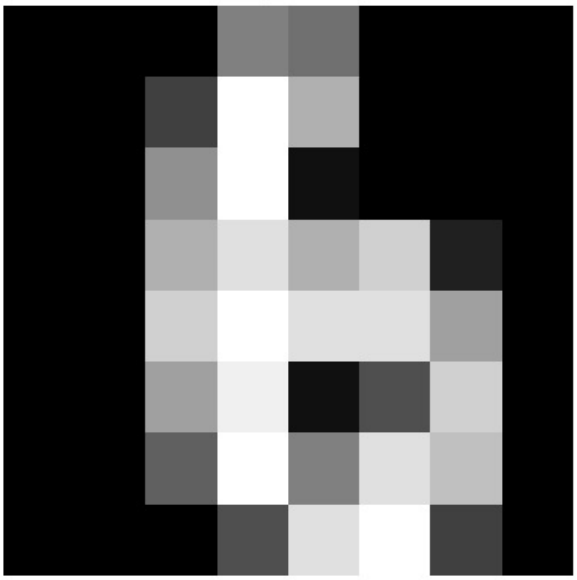
Actual: 6, Predicted: 6



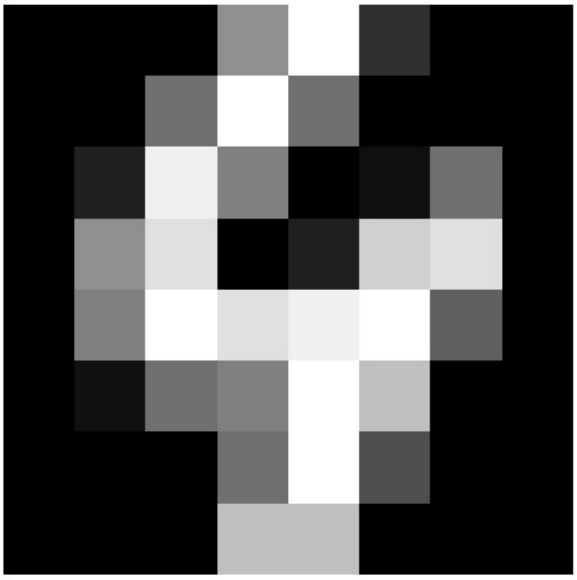
Actual: 2, Predicted: 2



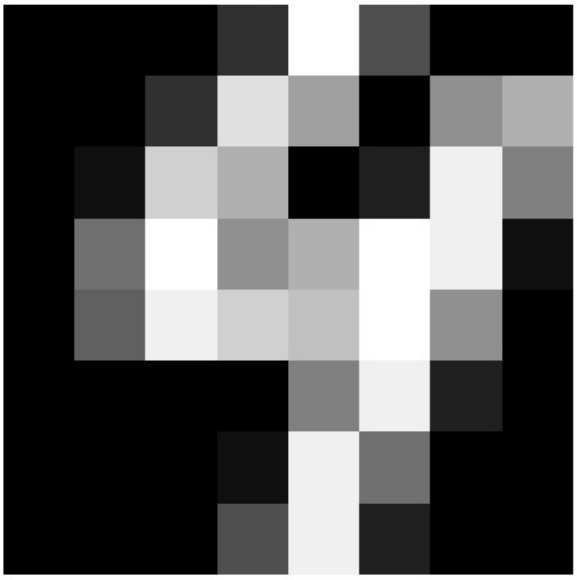
Actual: 6, Predicted: 6



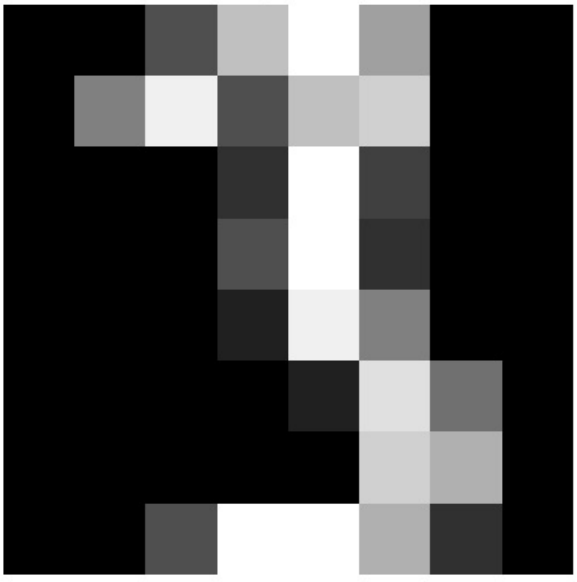
Actual: 4, Predicted: 4



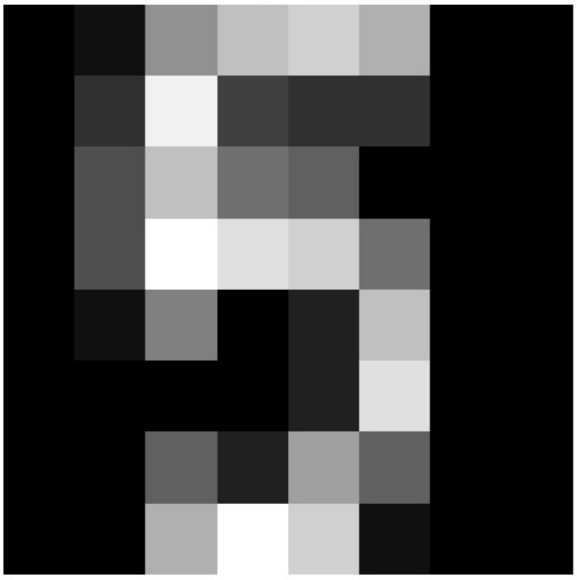
Actual: 4, Predicted: 4



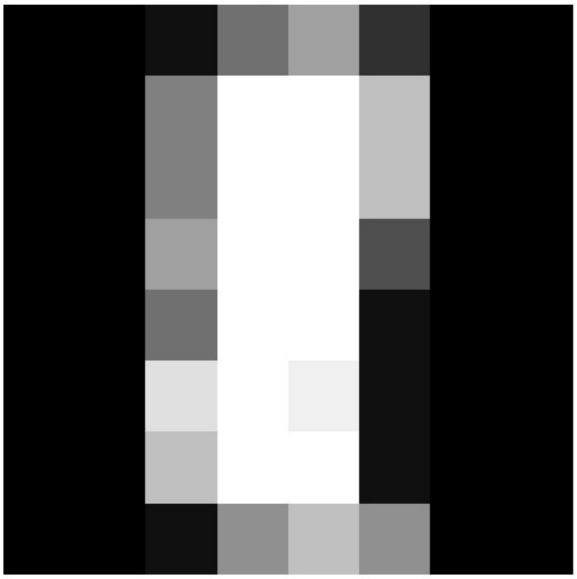
Actual: 3, Predicted: 3



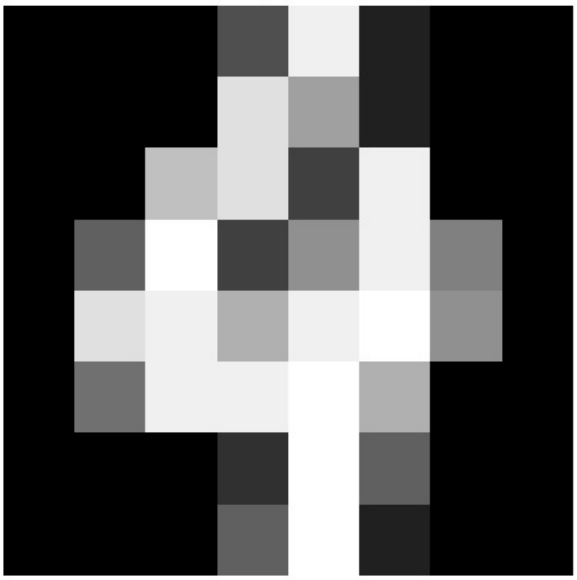
Actual: 5, Predicted: 5



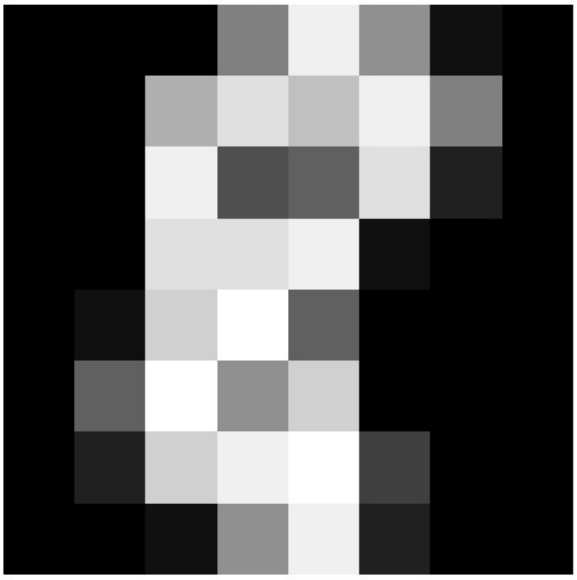
Actual: 1, Predicted: 1



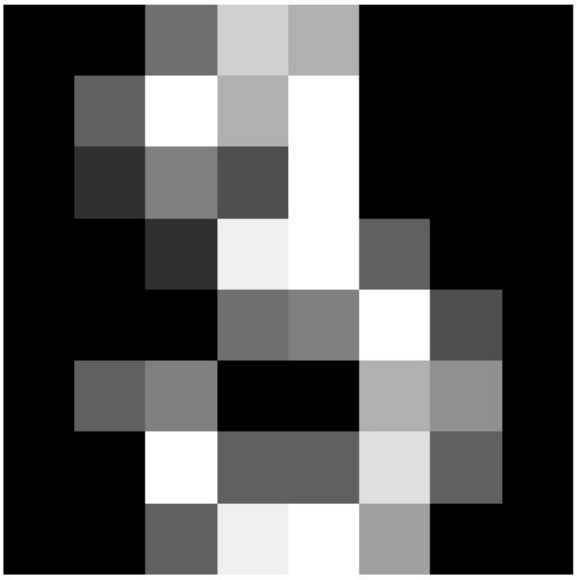
Actual: 4, Predicted: 4



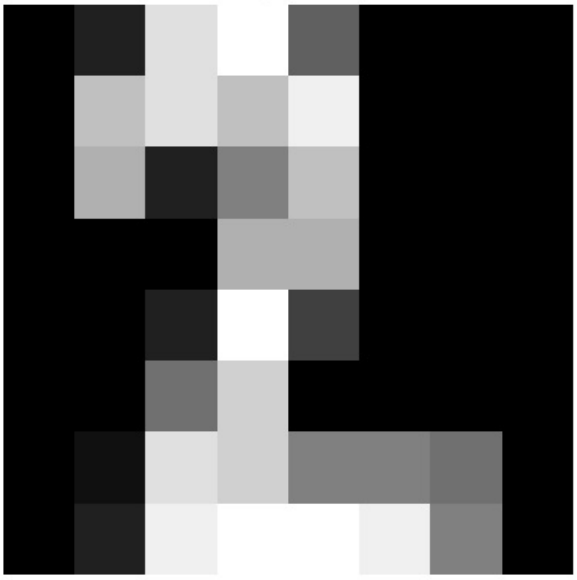
Actual: 8, Predicted: 8



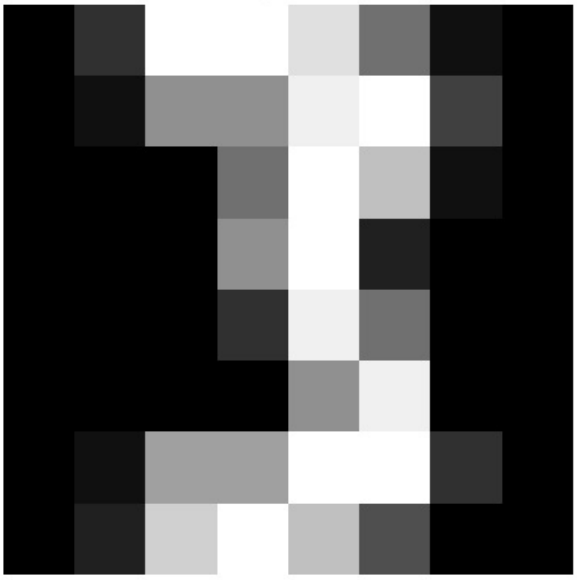
Actual: 3, Predicted: 3



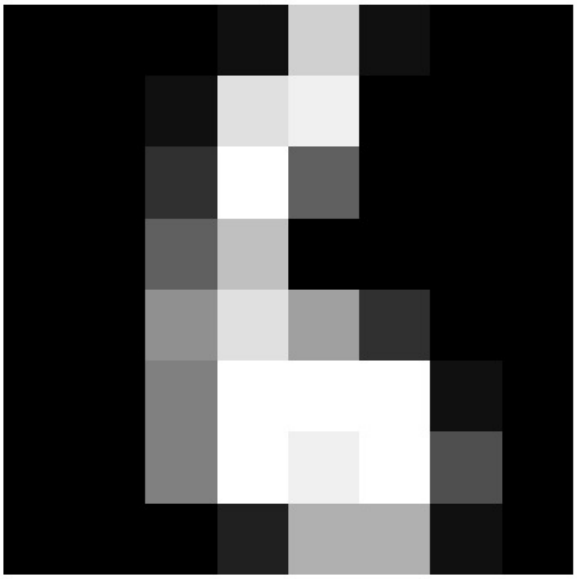
Actual: 2, Predicted: 2



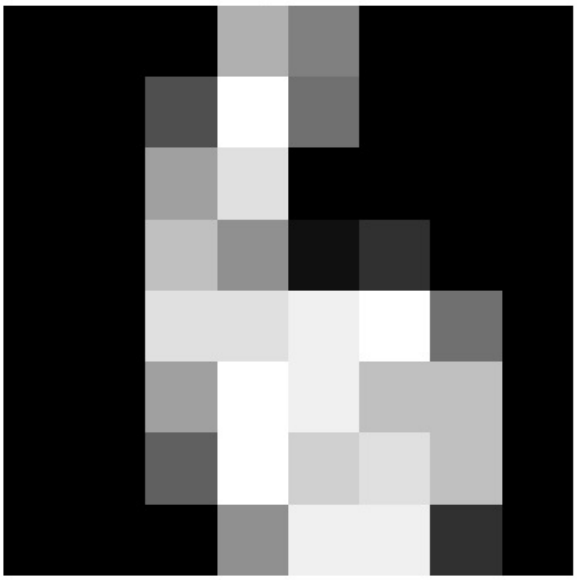
Actual: 3, Predicted: 3



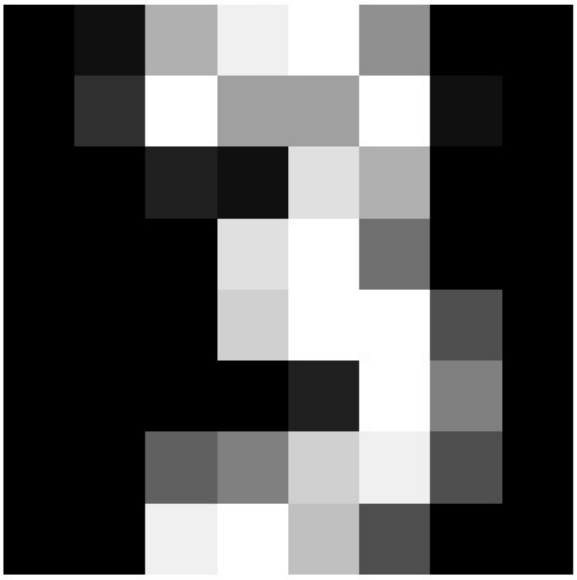
Actual: 6, Predicted: 6



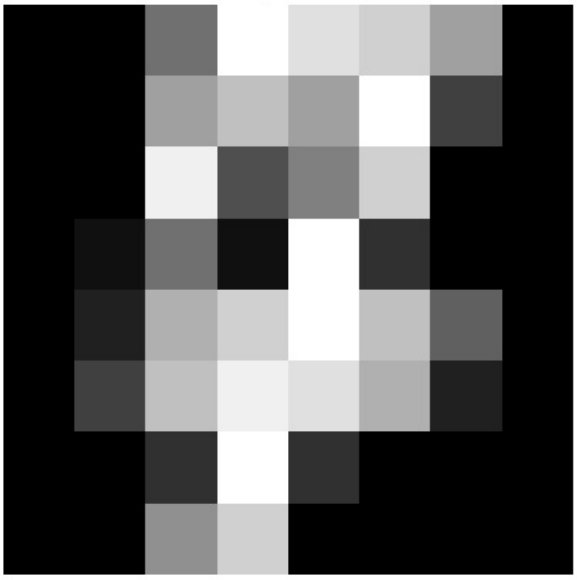
Actual: 6, Predicted: 6



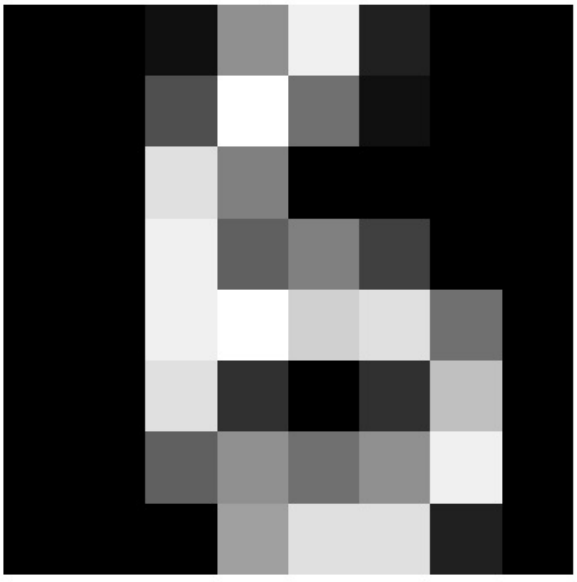
Actual: 3, Predicted: 3



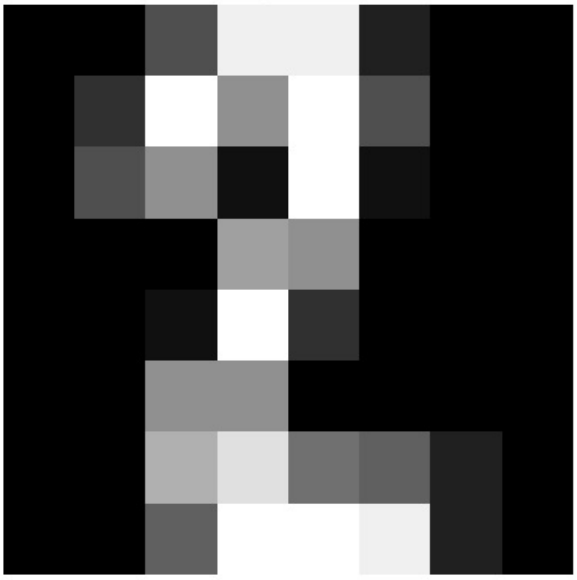
Actual: 7, Predicted: 7



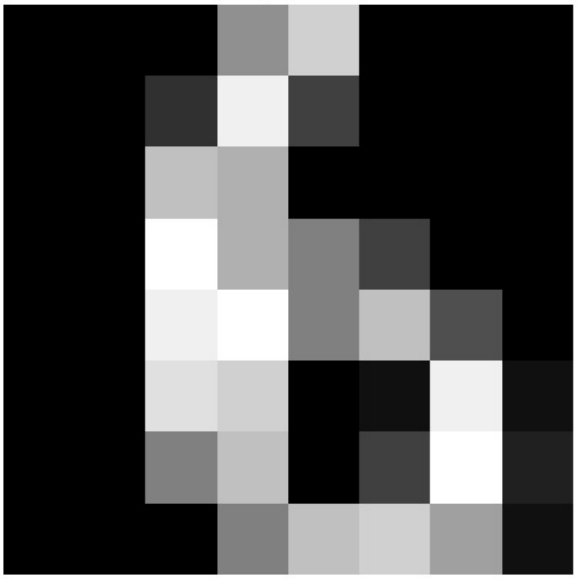
Actual: 6, Predicted: 6



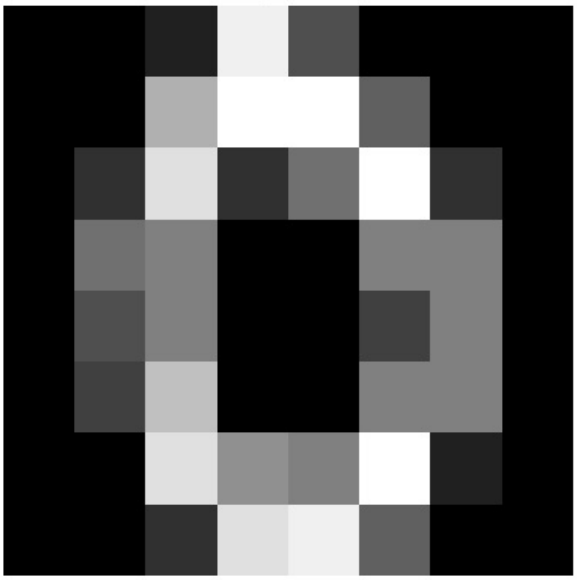
Actual: 2, Predicted: 2



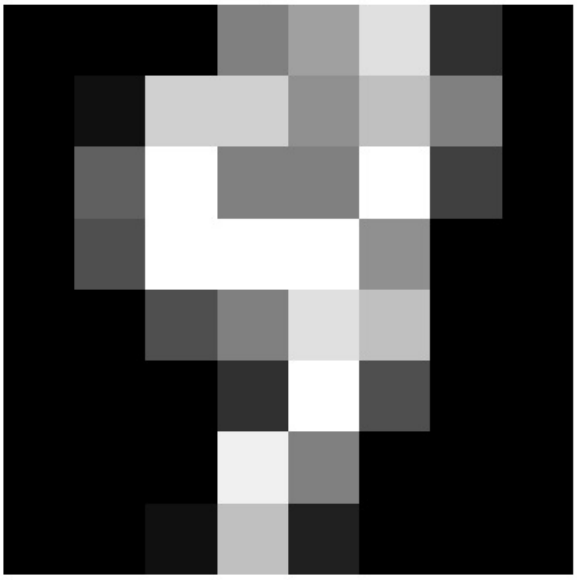
Actual: 6, Predicted: 6



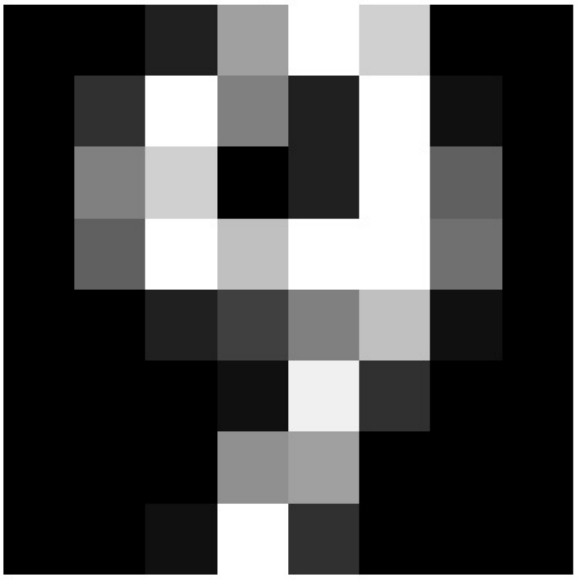
Actual: 0, Predicted: 0



Actual: 9, Predicted: 9

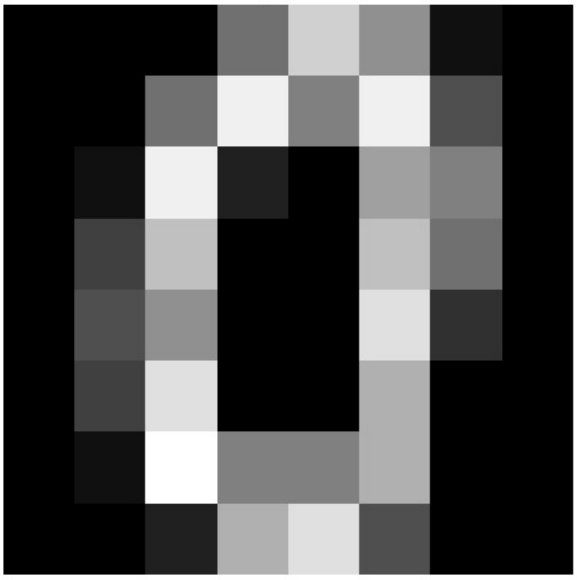


Actual: 9, Predicted: 9

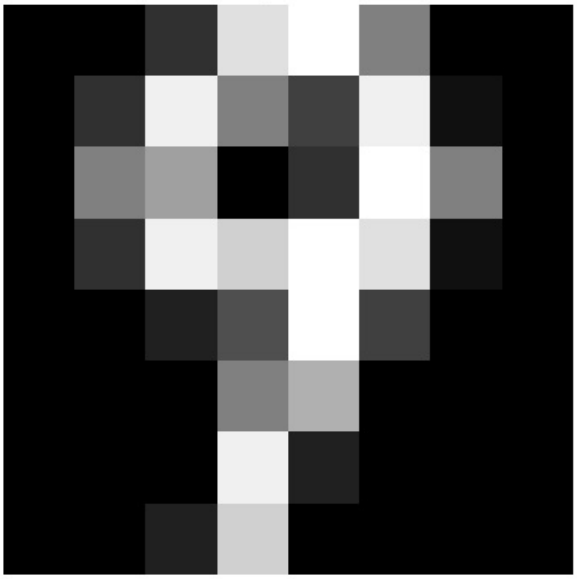


Actual: 3, Predicted: 3

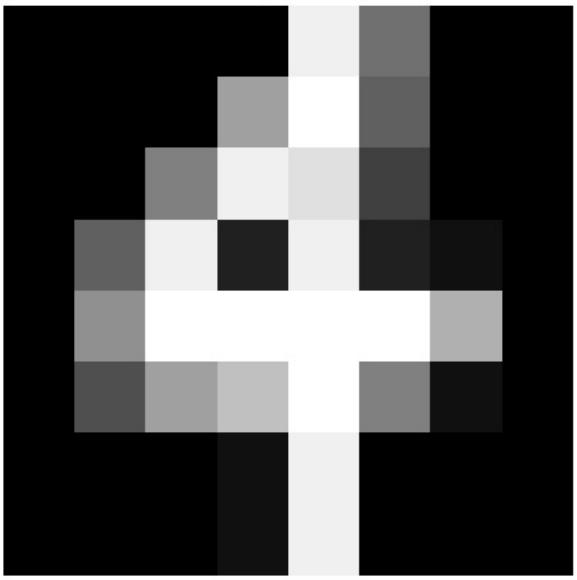
Actual: 0, Predicted: 0



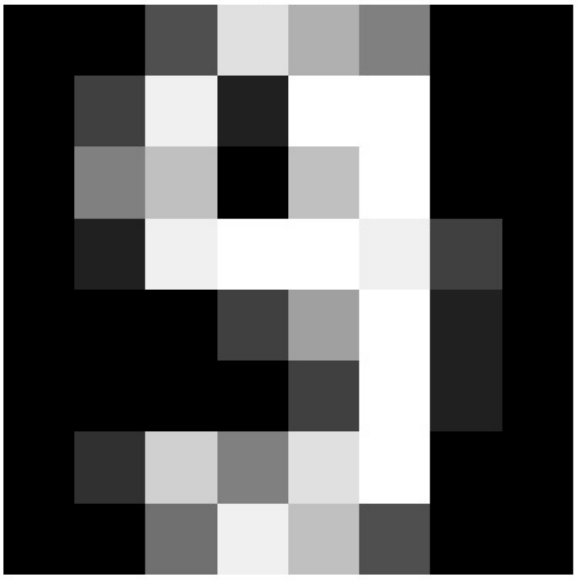
Actual: 9, Predicted: 9



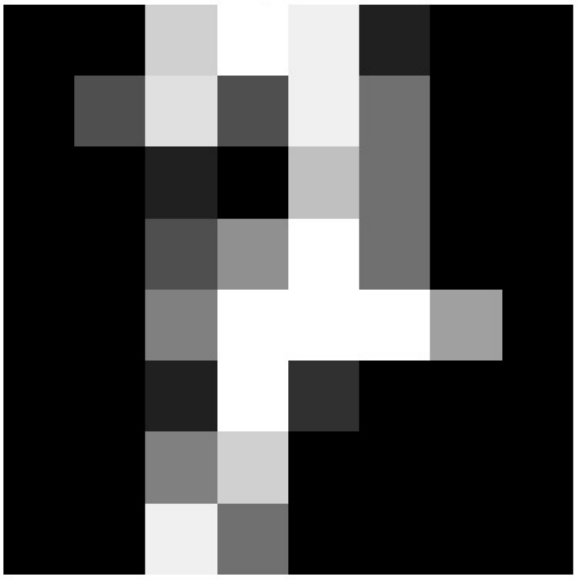
Actual: 4, Predicted: 4



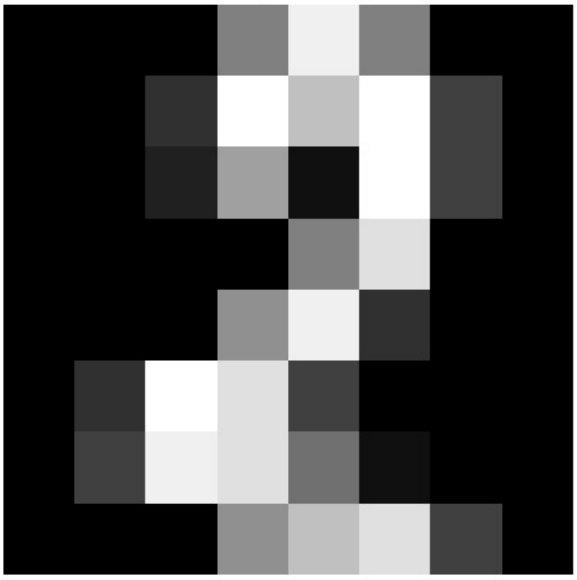
Actual: 9, Predicted: 9



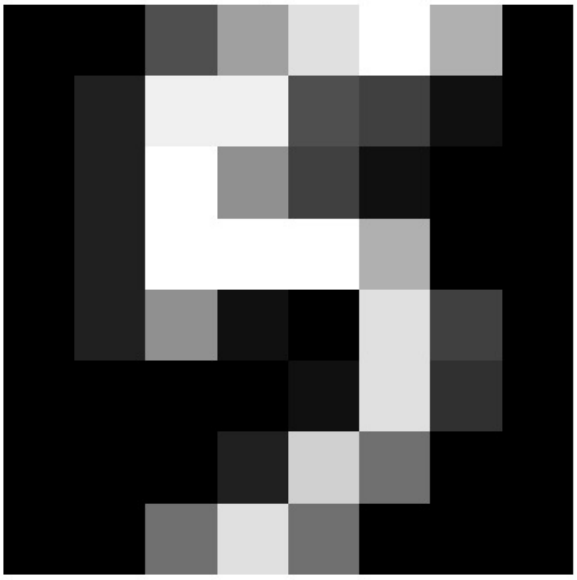
Actual: 7, Predicted: 7



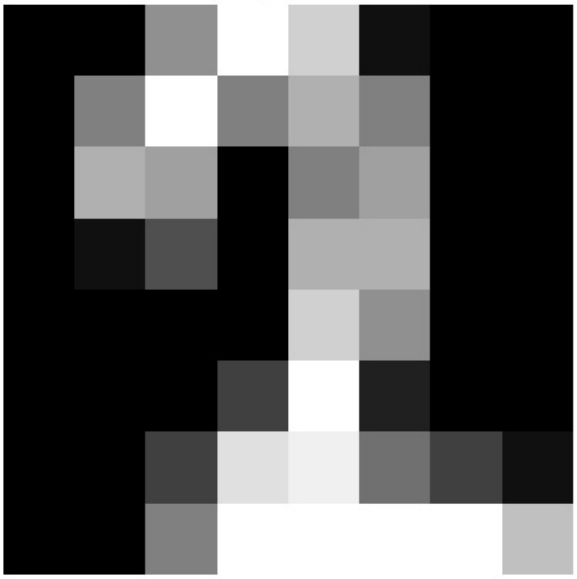
Actual: 2, Predicted: 2



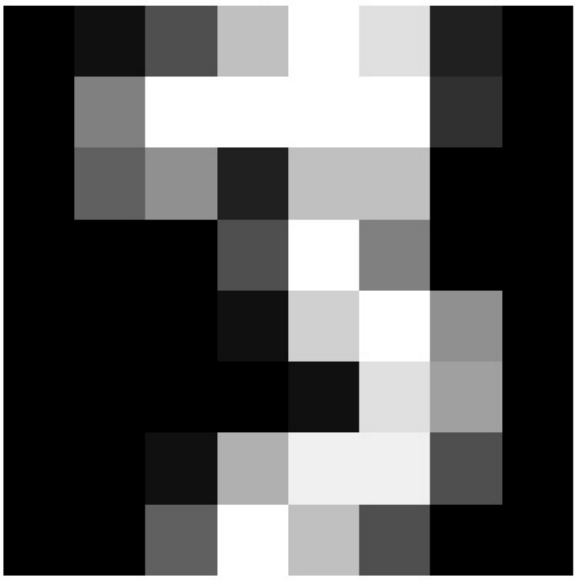
Actual: 5, Predicted: 5



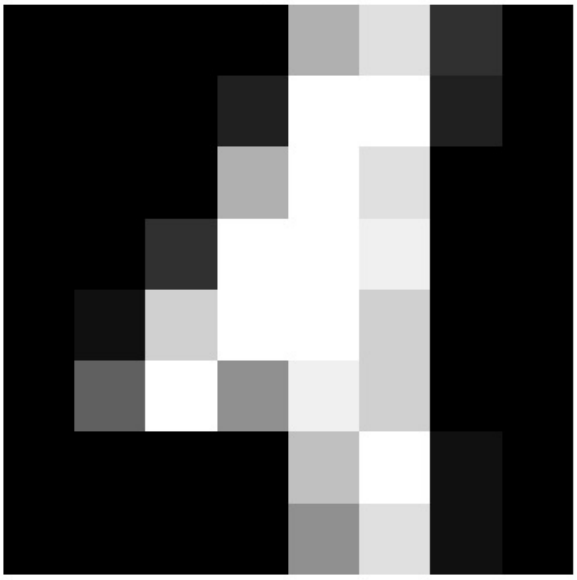
Actual: 2, Predicted: 2



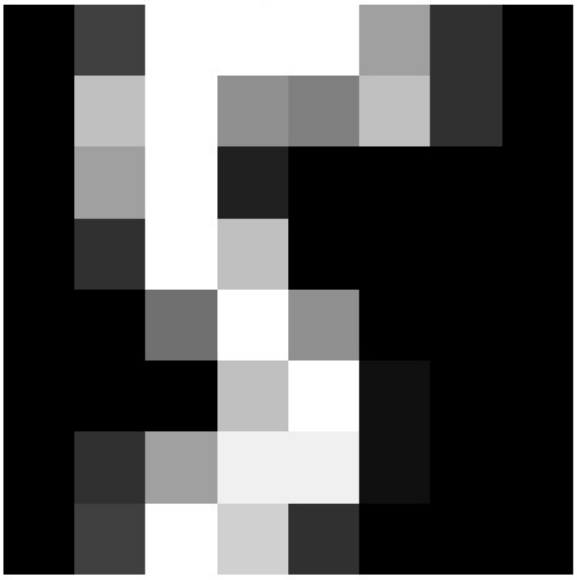
Actual: 3, Predicted: 3



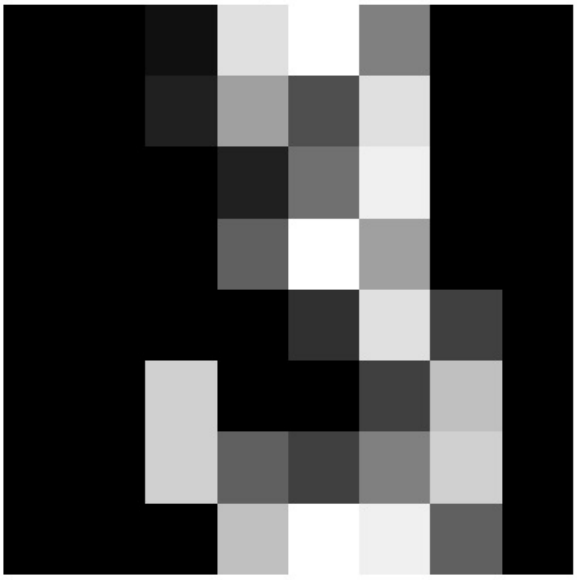
Actual: 1, Predicted: 1



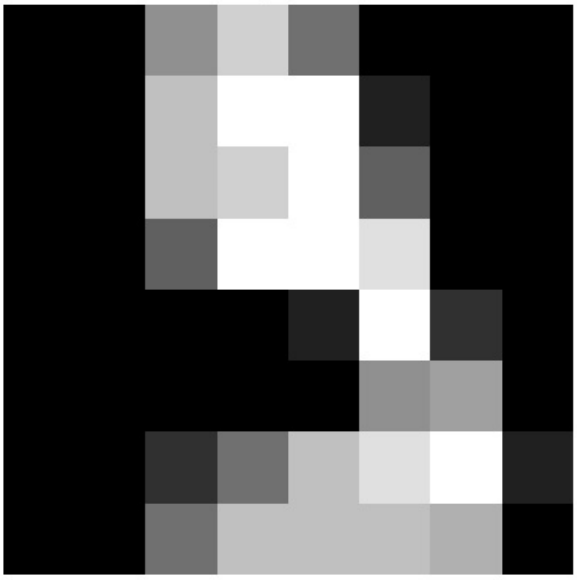
Actual: 5, Predicted: 5



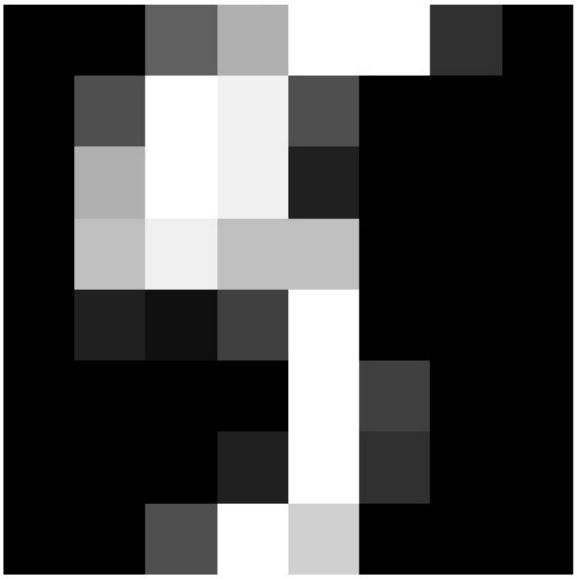
Actual: 3, Predicted: 3



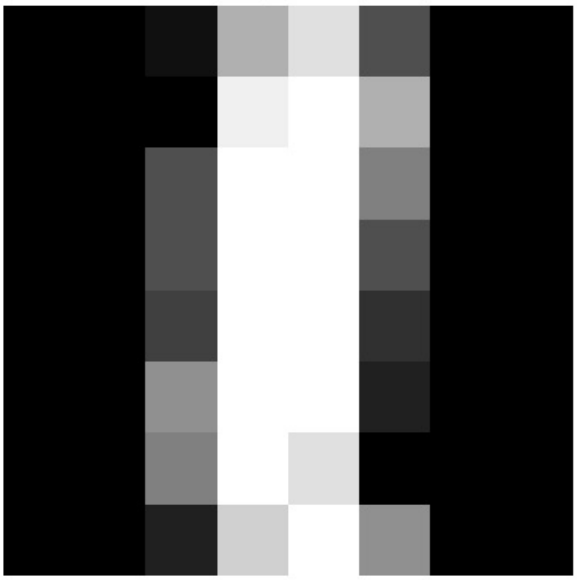
Actual: 9, Predicted: 9



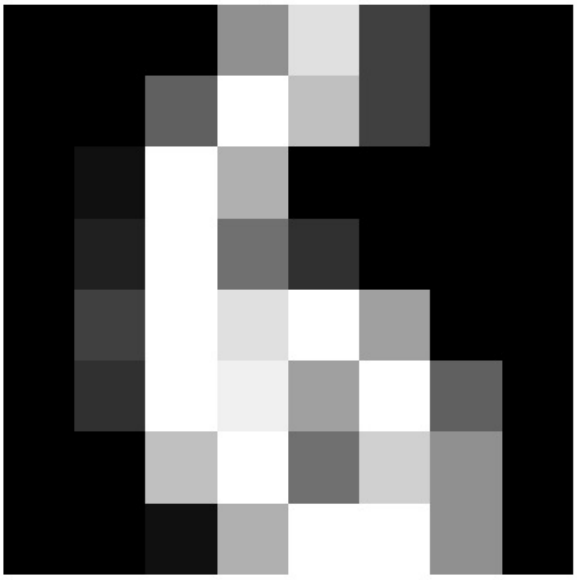
Actual: 5, Predicted: 5



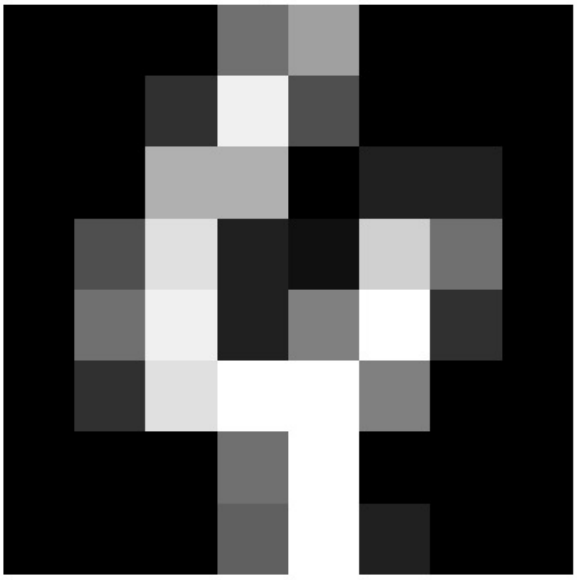
Actual: 1, Predicted: 1



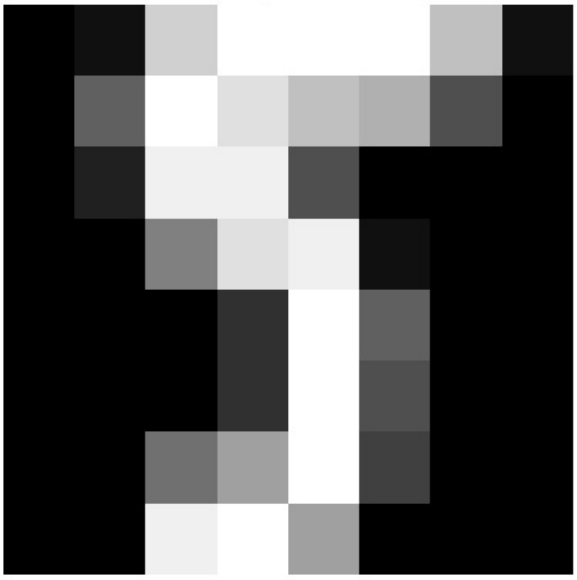
Actual: 6, Predicted: 6



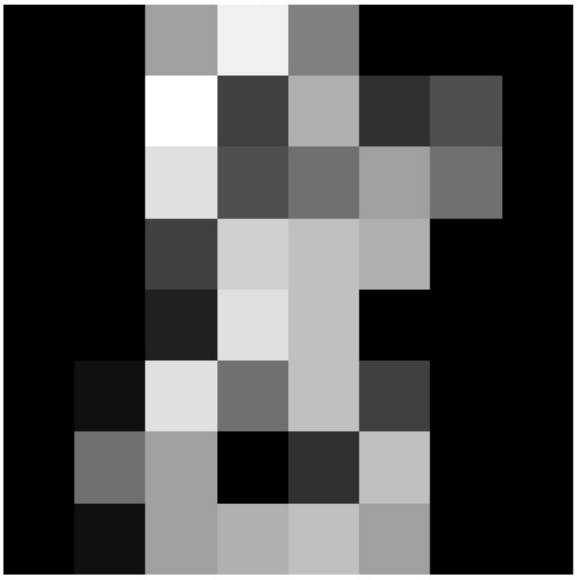
Actual: 4, Predicted: 4



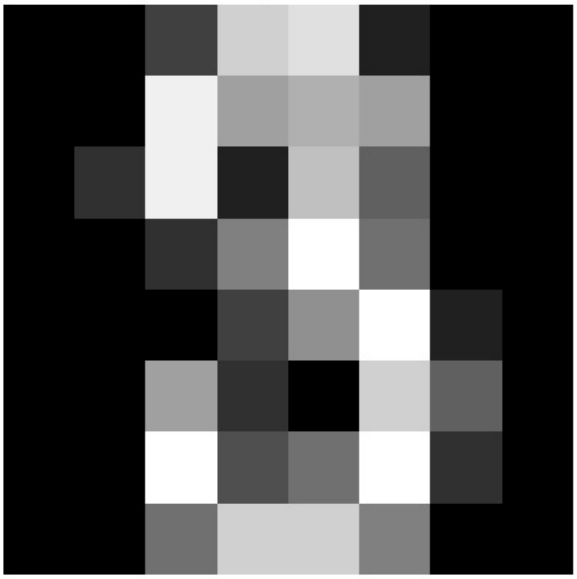
Actual: 5, Predicted: 5



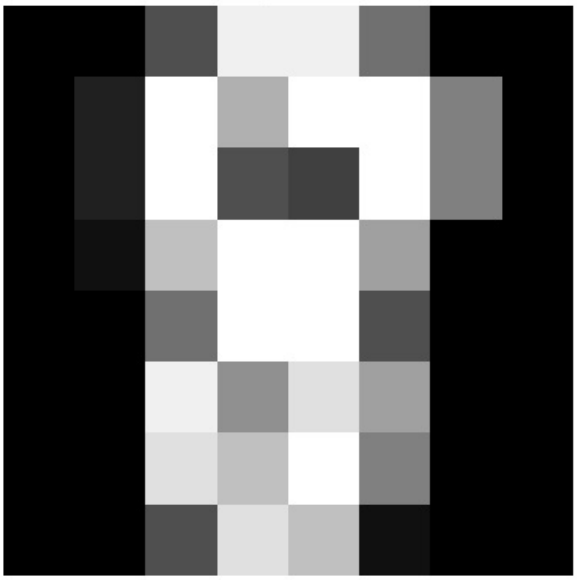
Actual: 8, Predicted: 8



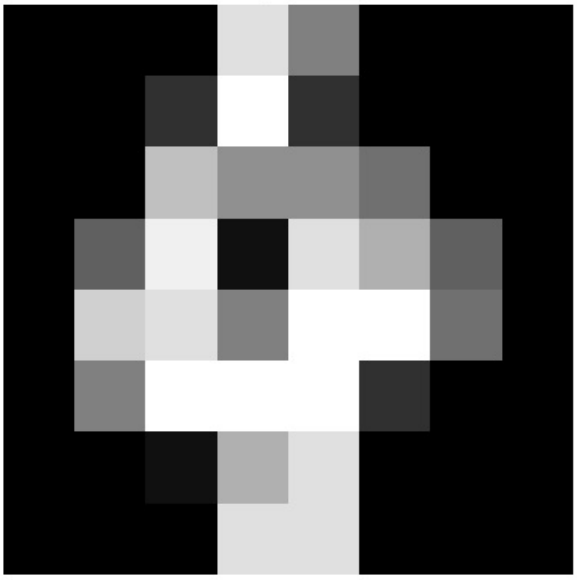
Actual: 3, Predicted: 3



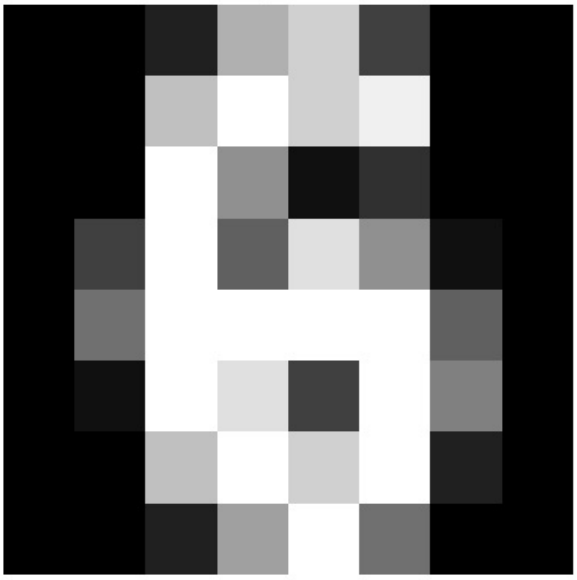
Actual: 8, Predicted: 8



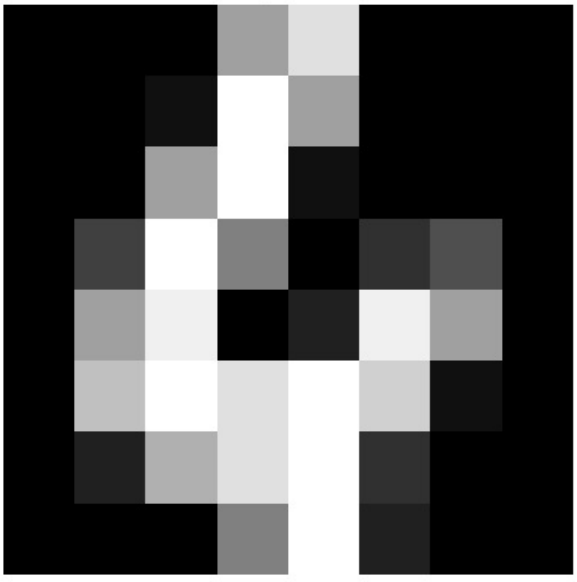
Actual: 4, Predicted: 4



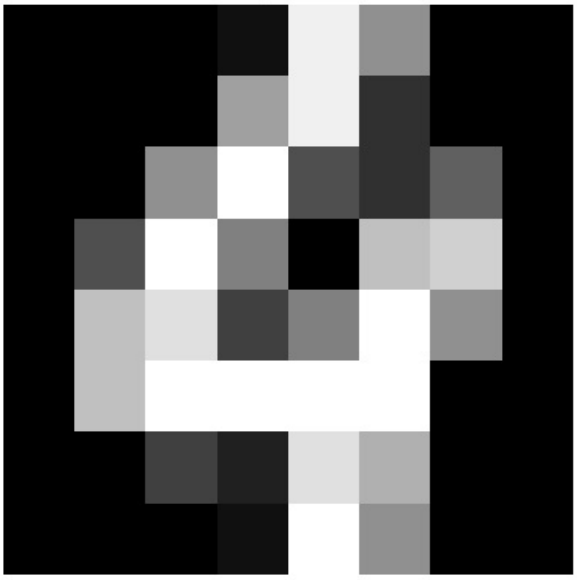
Actual: 6, Predicted: 6



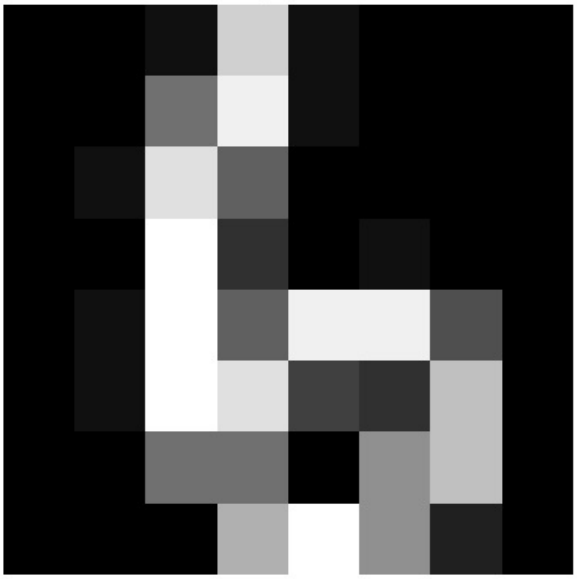
Actual: 4, Predicted: 4



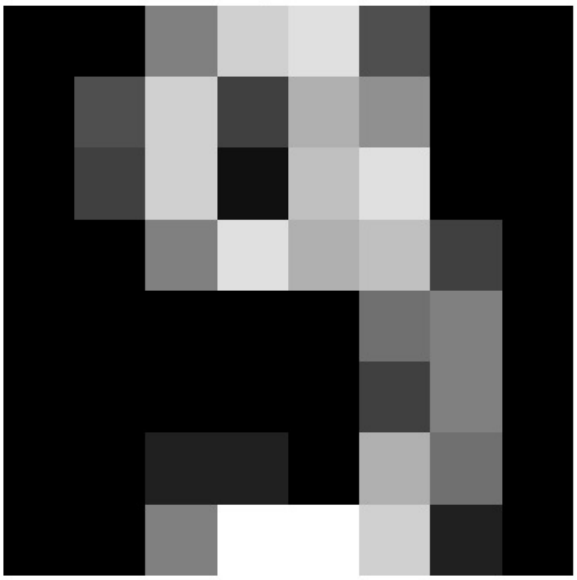
Actual: 4, Predicted: 4



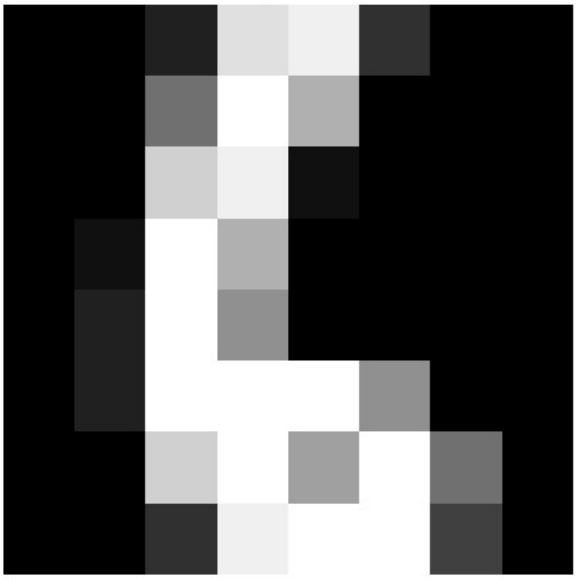
Actual: 6, Predicted: 6



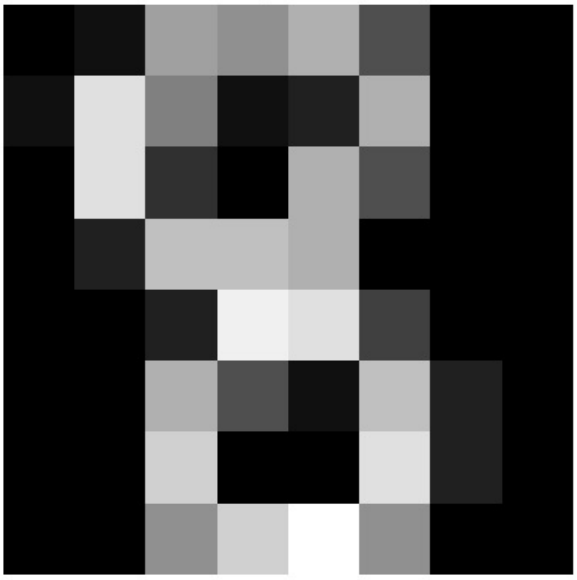
Actual: 9, Predicted: 9



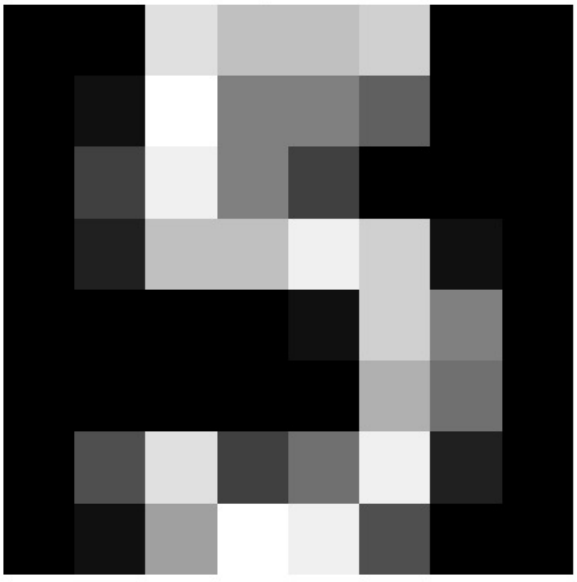
Actual: 6, Predicted: 6



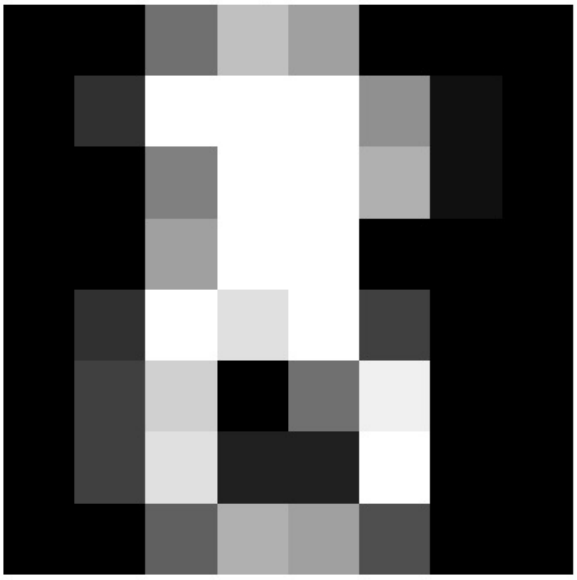
Actual: 8, Predicted: 8



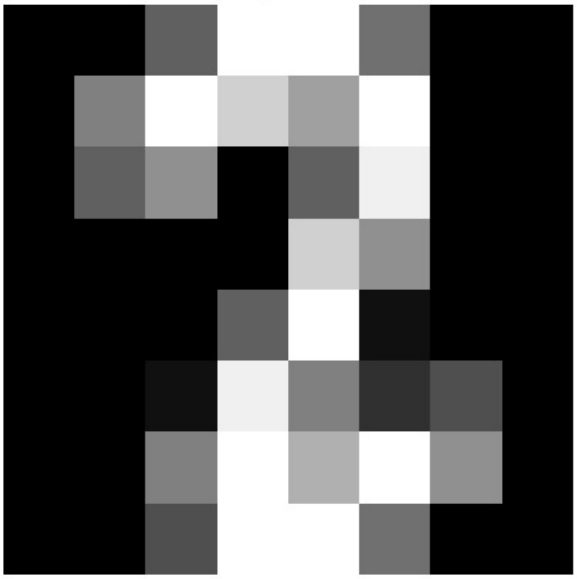
Actual: 5, Predicted: 5



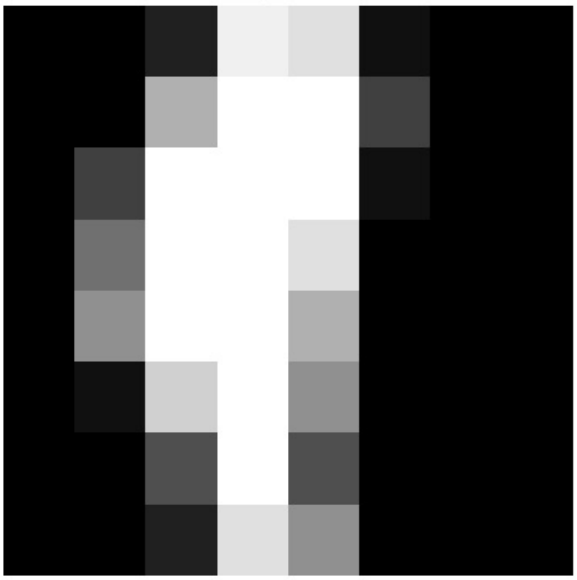
Actual: 8, Predicted: 8

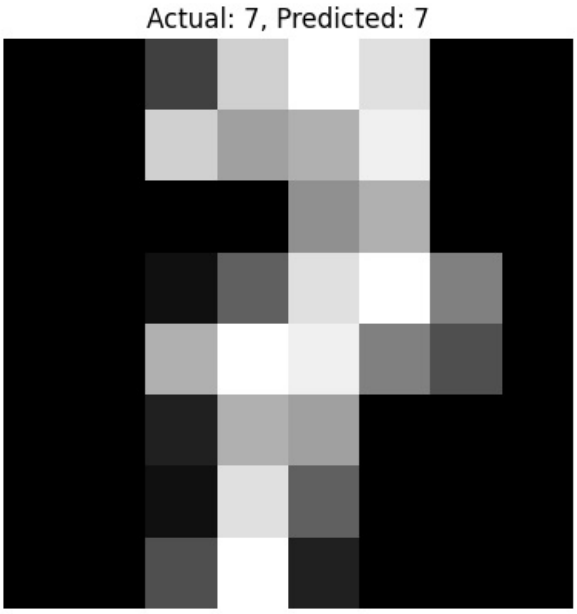


Actual: 2, Predicted: 2



Actual: 1, Predicted: 1

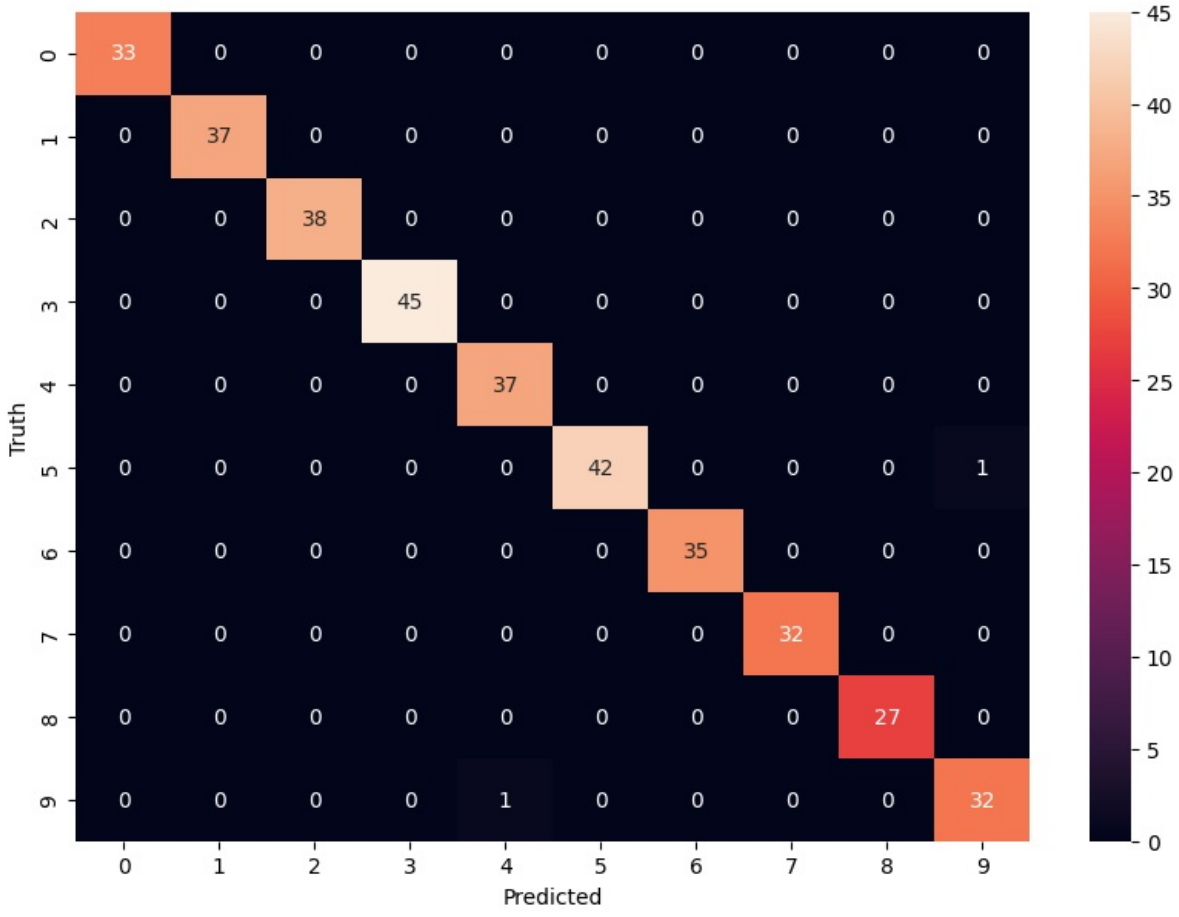




Confusion Matrix

In [64]:
```
cm = confusion_matrix(y_test, predict)
plt.figure(figsize=(10, 7))
sns.heatmap(cm, annot=True)
plt.xlabel("Predicted")
plt.ylabel("Truth")
```

Out[64]: Text(95.72222222222221, 0.5, 'Truth')



Classification Report

In [66]:
```
from sklearn.metrics import classification_report
```

```
print(classification_report(y_test, predict))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        33
           1       1.00      1.00      1.00        37
           2       1.00      1.00      1.00        38
           3       1.00      1.00      1.00        45
           4       0.97      1.00      0.99        37
           5       1.00      0.98      0.99        43
           6       1.00      1.00      1.00        35
           7       1.00      1.00      1.00        32
           8       1.00      1.00      1.00        27
           9       0.97      0.97      0.97        33

    accuracy                           0.99       360
   macro avg       0.99      0.99      0.99       360
weighted avg       0.99      0.99      0.99       360
```