In [1]:
```
from sklearn.datasets import load_digits
digits = load_digits()
```

In [2]:
```
print("Image Data Shape", digits.data.shape)
```

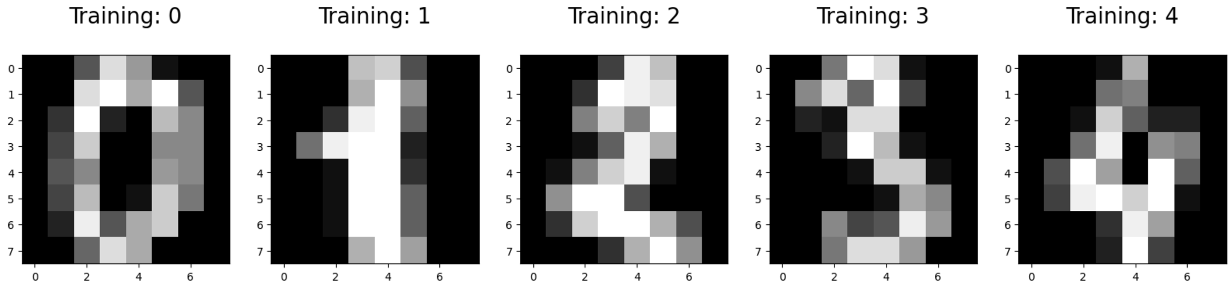
```
Image Data Shape (1797, 64)
```

In [3]:
```
print("Label Data Shape", digits.target.shape)
```

```
Label Data Shape (1797,)
```

In [4]:
```
import numpy as np
import matplotlib.pyplot as plt
```

In [7]:
```
plt.figure(figsize=(20,4))
for index, (image,label) in enumerate(zip(digits.data[0:5],digits.target[0:5])):
    plt.subplot(1,5,index + 1)
    plt.imshow(np.reshape(image,(8,8)),cmap=plt.cm.gray)
    plt.title('Training: %i\n' % label, fontsize = 20)
```



Training: 0 Training: 1 Training: 2 Training: 3 Training: 4

In [8]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(digits.data,
                                                    digits.target, test_size=0.25, random_state=0
```

In [9]:
```
from sklearn.linear_model import LogisticRegression
```

In [10]:
```
logisticRegr = LogisticRegression()
```

In [11]:
```
logisticRegr.fit(x_train,y_train)
```

```
C:\Users\kaush\anaconda3\Lib\site-packages\sklearn\linear_model\_logistic.py:460: ConvergenceWarnin
g: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html (https://scikit-learn.org/stable/mod
ules/preprocessing.html)
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-l
earn.org/stable/modules/linear_model.html#logistic-regression)
  n_iter_i = _check_optimize_result(
```

Out[11]:
```
▾ LogisticRegression
LogisticRegression()
```

In [12]:
```
logisticRegr.predict(x_test[0].reshape(1,-1))
```

Out[12]:
```
array([2])
```

In [13]:
```
logisticRegr.predict(x_test[0:10])
```

Out[13]:
```
array([2, 8, 2, 6, 6, 7, 1, 9, 8, 5])
```

In [15]:

```
predictions = logisticRegr.predict(x_test)
```

In [16]:

```
score = logisticRegr.score(x_test,y_test)
print(score)
```

```
0.9511111111111111
```

In [19]:

```
 import seaborn as sns
from sklearn import metrics
```

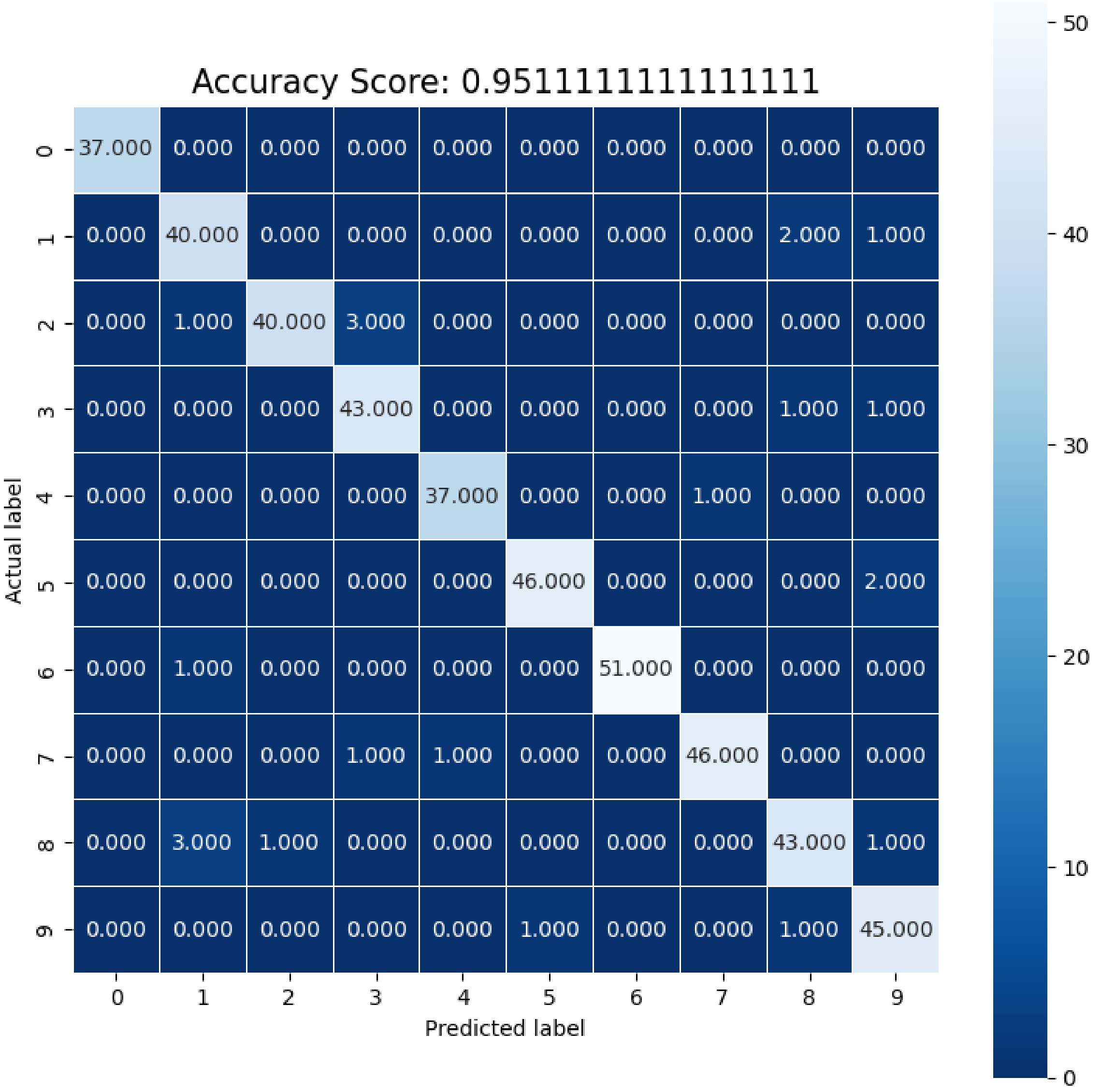
In [21]:

```
cm = metrics.confusion_matrix(y_test,predictions)
print(cm)
```

```
[[37  0  0  0  0  0  0  0  0  0]
 [ 0 40  0  0  0  0  0  0  2  1]
 [ 0  1 40  3  0  0  0  0  0  0]
 [ 0  0  0 43  0  0  0  0  1  1]
 [ 0  0  0  0 37  0  0  1  0  0]
 [ 0  0  0  0  0 46  0  0  0  2]
 [ 0  1  0  0  0  0 51  0  0  0]
 [ 0  0  0  1  1  0  0 46  0  0]
 [ 0  3  1  0  0  0  0  0 43  1]
 [ 0  0  0  0  0  1  0  0  1 45]]
```

In [26]:

```
plt.figure(figsize=(9,9))
sns.heatmap(cm, annot=True,fmt=".3f", linewidth=.5, square = True, cmap = 'Blues_r');
plt.ylabel('Actual label');
plt.xlabel('Predicted label');
all_sample_title = 'Accuracy Score: {0}'.format(score)
plt.title(all_sample_title, size = 15);
```



	0	1	2	3	4	5	6	7	8	9
0	37.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	40.000	0.000	0.000	0.000	0.000	0.000	0.000	2.000	1.000
2	0.000	1.000	40.000	3.000	0.000	0.000	0.000	0.000	0.000	0.000
3	0.000	0.000	0.000	43.000	0.000	0.000	0.000	0.000	1.000	1.000
4	0.000	0.000	0.000	0.000	37.000	0.000	0.000	1.000	0.000	0.000
5	0.000	0.000	0.000	0.000	0.000	46.000	0.000	0.000	0.000	2.000
6	0.000	1.000	0.000	0.000	0.000	0.000	51.000	0.000	0.000	0.000
7	0.000	0.000	0.000	1.000	1.000	0.000	0.000	46.000	0.000	0.000
8	0.000	3.000	1.000	0.000	0.000	0.000	0.000	0.000	43.000	1.000
9	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000	1.000	45.000

In [ ]: