Karanjot Singh

In [1]:

```
from sklearn.datasets import load_digits
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import metrics
from warnings import simplefilter
```

```
C:\Users\Karan Singh\Anaconda3\lib\importlib\_bootstrap.py:219: RuntimeWar
ning: numpy.ufunc size changed, may indicate binary incompatibility. Expec
ted 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
C:\Users\Karan Singh\Anaconda3\lib\importlib\_bootstrap.py:219: RuntimeWar
ning: numpy.ufunc size changed, may indicate binary incompatibility. Expec
ted 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
```

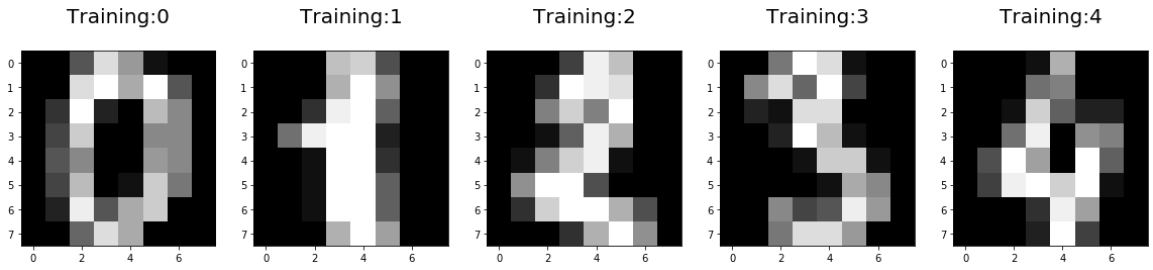
In [2]:

```
digits = load_digits()
print("Image data shape",digits.data.shape)
print("Label data shape",digits.target.shape)
```

```
Image data shape (1797, 64)
Label data shape (1797,)
```

In [5]:

```
plt.figure(figsize=(20,40))
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)
```



In [6]:

```
x_train,x_test,y_train,y_test = train_test_split(digits.data,digits.target,test_size=0.
23,random_state = 2)
print(x_train.shape,x_test.shape,y_train.shape,y_test.shape)
```

```
(1383, 64) (414, 64) (1383,) (414,)
```

In [7]:

```
logist = LogisticRegression()
logist.fit(x_train,y_train)
```

```
C:\Users\Karan Singh\Anaconda3\lib\site-packages\sklearn\linear_model\logi
stic.py:432: FutureWarning: Default solver will be changed to 'lbfgs' in
0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\Karan Singh\Anaconda3\lib\site-packages\sklearn\linear_model\logi
stic.py:469: FutureWarning: Default multi_class will be changed to 'auto'
in 0.22. Specify the multi_class option to silence this warning.
  "this warning.", FutureWarning)
```

Out[7]:

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

In [8]:

```
pred = logist.predict(x_test)

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

```
0.9420289855072463
```

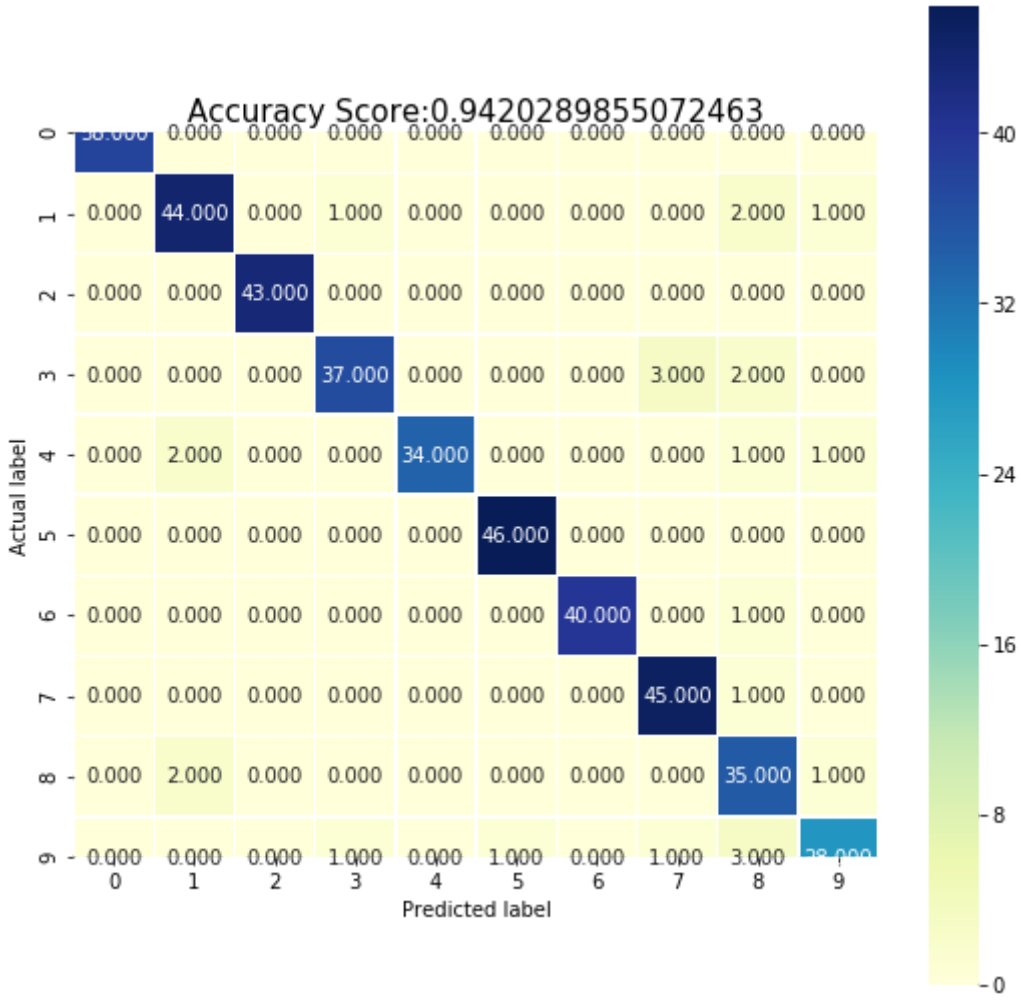
In [9]:

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

```
[[38  0  0  0  0  0  0  0  0  0]
 [ 0 44  0  1  0  0  0  0  2  1]
 [ 0  0 43  0  0  0  0  0  0  0]
 [ 0  0  0 37  0  0  0  3  2  0]
 [ 0  2  0  0 34  0  0  0  1  1]
 [ 0  0  0  0  0 46  0  0  0  0]
 [ 0  0  0  0  0  0 40  0  1  0]
 [ 0  0  0  0  0  0  0 45  1  0]
 [ 0  2  0  0  0  0  0  0 35  1]
 [ 0  0  0  1  0  1  0  1  3 28]]
```

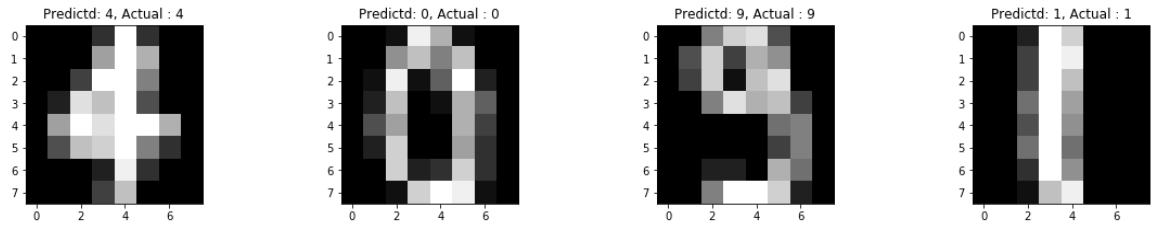
Out[9]:

```
Text(0.5, 1, 'Accuracy Score:0.9420289855072463')
```



In [13]:

```
index = 0
misclassifiedindex = []
for predict,actual in zip(pred,y_test):
    if predict == actual:
        misclassifiedindex.append(index)
        index+=1
plt.figure(figsize=(20,3))
for plotIndex, wrong in enumerate(misclassifiedindex[0:4]):
    plt.subplot(1,4,plotIndex +1)
    plt.imshow(np.reshape(x_test[wrong],(8,8)),cmap=plt.cm.gray)
    plt.title("Predictd: {}, Actual : {}".format(pred[wrong],y_test[wrong], fontsize=20
))
```



In [ ]:

```

```