Week10

August 12, 2023

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from keras.datasets import mnist
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D
from keras.utils import np_utils
from keras import backend as K
from sklearn.metrics import confusion_matrix
import seaborn as sns
```

Display a confusion matrix on the test set classifications. Summarize your results.

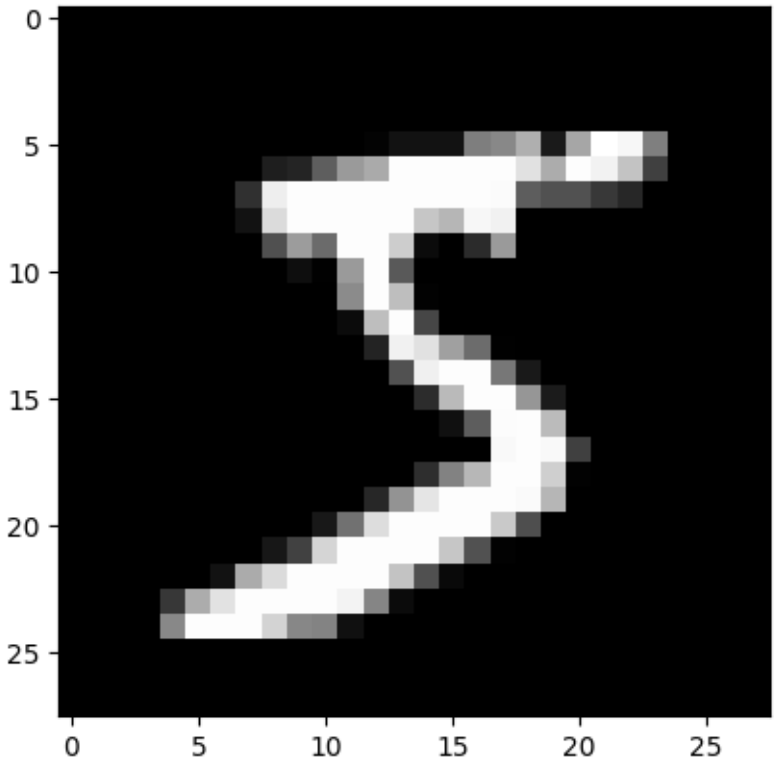
Load the MNIST data set

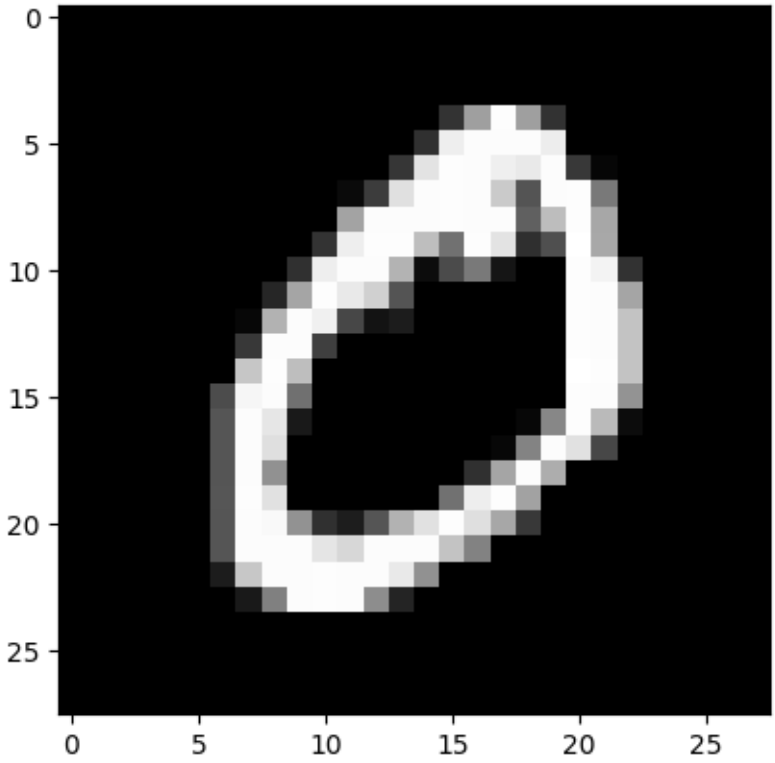
[3]:
```
(data_train, target_train), (data_test, target_test) = mnist.load_data()
```

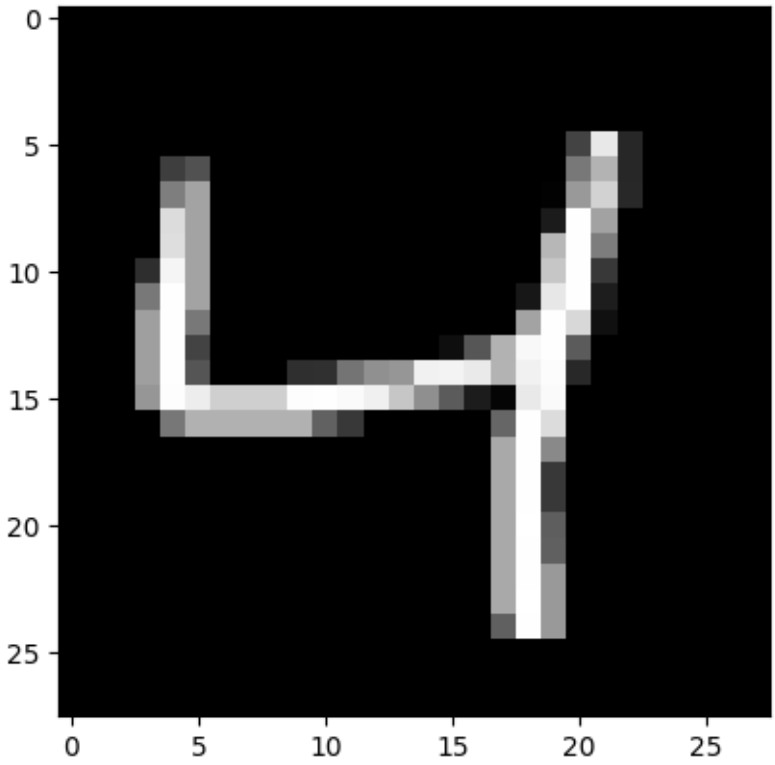
```
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-
datasets/mnist.npz
11490434/11490434 [==============================] - 0s 0us/step
```

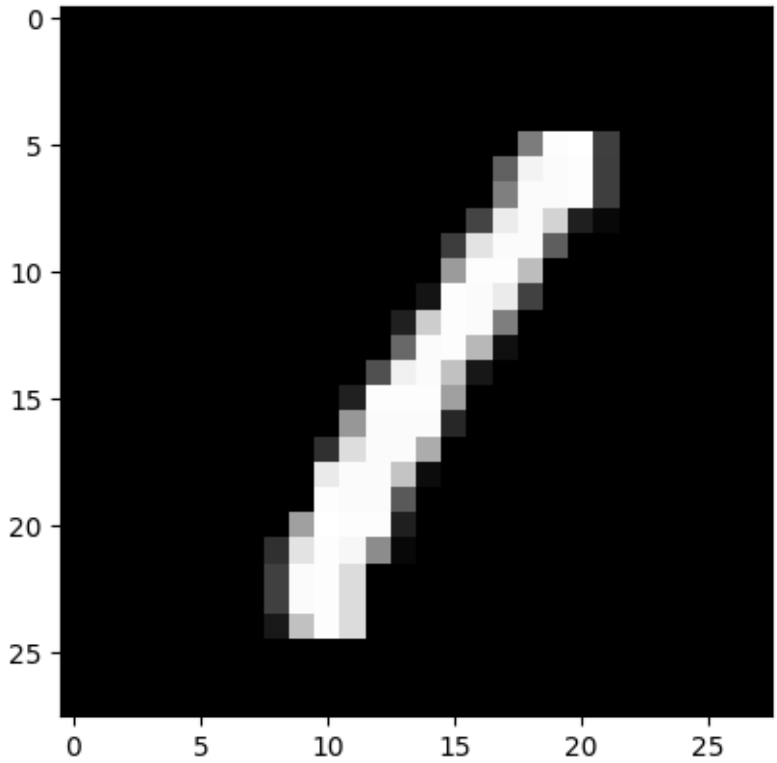
Display the first five images in the training data set (see section 8.1 in the Machine Learning with Python Cookbook).

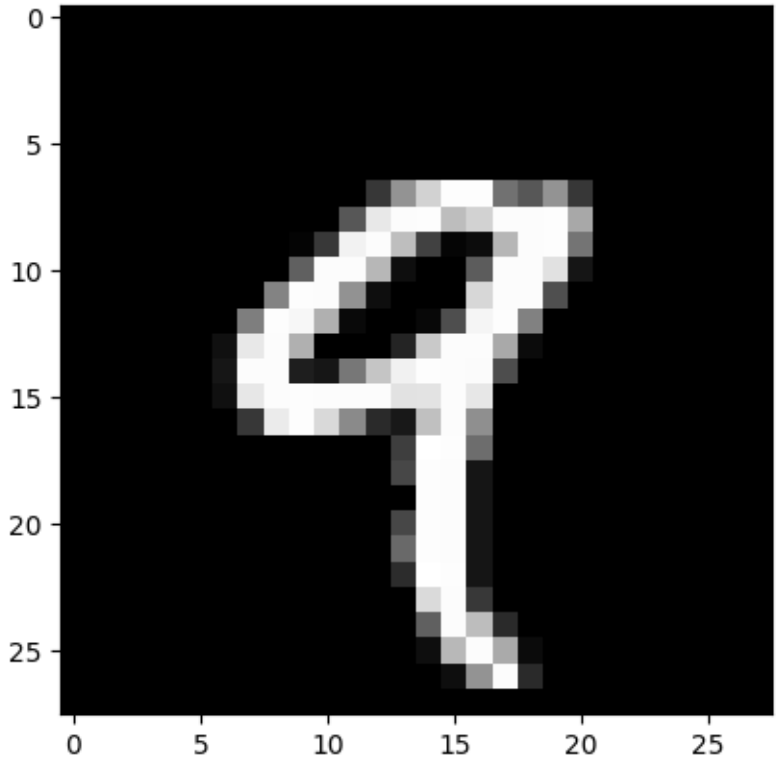
[6]:
```
for i in range(5):
    plt.imshow(data_train[i], cmap='gray')
    plt.show()
```


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25


0
5
10
15
20
25
0
5
10
15
20
25

Compare these to the first five training labels.

[8]:
```
for i in range(5):
    print("Label:", target_train[i])
```

```
Label: 5
Label: 0
Label: 4
Label: 1
Label: 9
```

Build and train a Keras CNN classifier on the MNIST training set.

[9]:
```
channels = 1
height = 28
width = 28
```

[27]:
```
# Reshape train and test data into features
x_train = data_train.reshape(data_train.shape[0], width, height, channels)
x_test = data_test.reshape(data_test.shape[0], width, height, channels)
# Rescale pixel intenstiy to between 0 and 1
x_train = x_train / 255
x_test = x_test / 255
```

[12]:
```
# On-hot encode target
y_train = np_utils.to_categorical(target_train)
y_test = np_utils.to_categorical(target_test)
number_of_classes = y_test.shape[1]
```

[29]:
```
# Start a nueral network
network = Sequential()
```

[30]:
```
# Build the Model

# Add convolutional layer with 64 filters, 5x5 window, and ReLu acrivation␣
 ↪function
network.add(Conv2D(filters=64,
                   kernel_size=(5, 5),
                   input_shape=(28, 28, 1),
                   activation='relu'))
# Add max pooling layer with a 2x2 window
network.add(MaxPool2D(pool_size=(2, 2)))
# Add dropout layer
network.add(Dropout(0.5))
# Add layer to flatten input
network.add(Flatten())
```

```
# Add fully connected layer of 128 units with ReLU activation function
network.add(Dense(128, activation='relu'))
# Add dropout layer
network.add(Dropout(0.5))
# Add fully connected layer with a softmax activation function
network.add(Dense(number_of_classes, activation='softmax'))
# Compile neural network
network.compile(loss="categorical_crossentropy",
                optimizer="rmsprop",
                metrics=["accuracy"])
```

[31]:
```
# Train neural network
network.fit(x_train,
            y_train,
            epochs=2,
            verbose=0,
            validation_batch_size=(x_test, y_test))
```

[31]:
```
<keras.callbacks.History at 0x7be379318fa0>
```

Report the test accuracy of your model.

[33]:
```
_, test_accuracy = network.evaluate(x_test, y_test)
print("Test Accuracy:", test_accuracy)
```

```
313/313 [==============================] - 3s 10ms/step - loss: 0.0487 -
accuracy: 0.9842
Test Accuracy: 0.9842000007629395
```

Display a confusion matrix on the test set classifications.

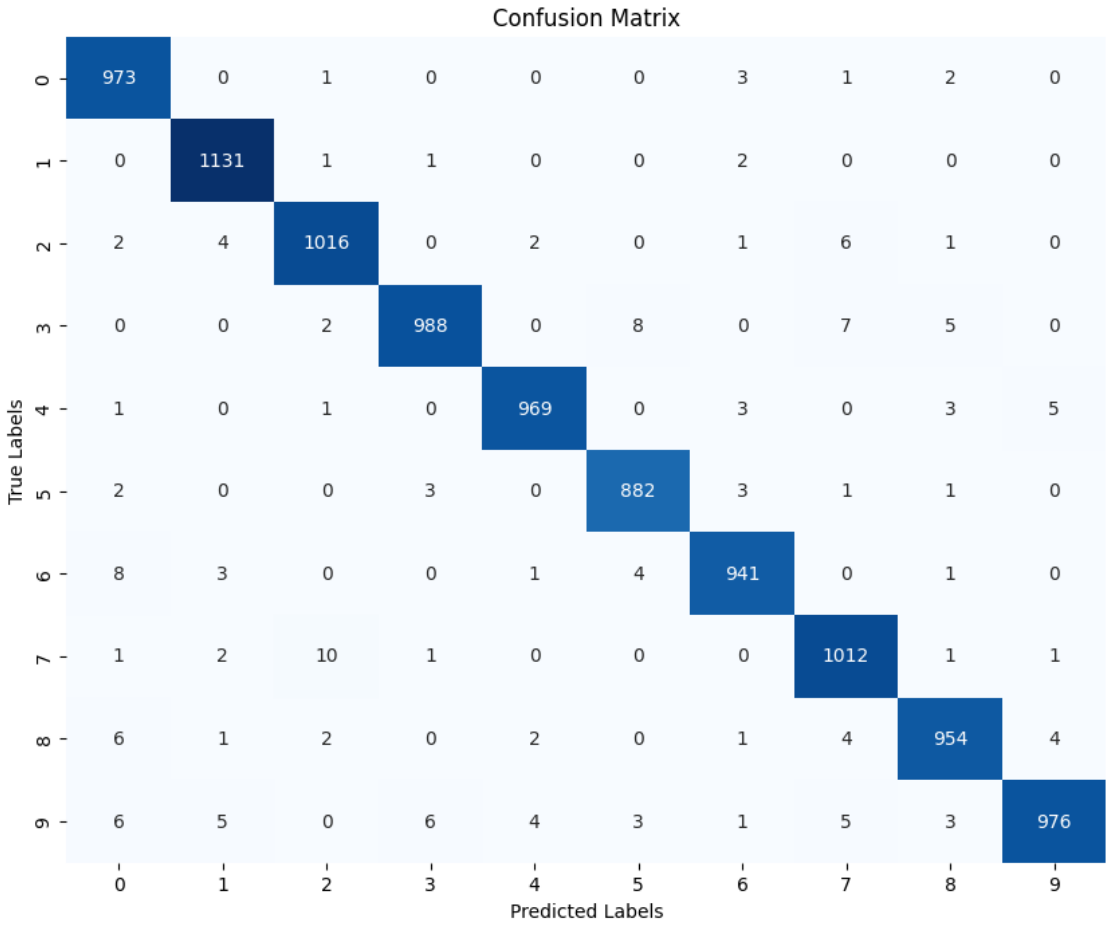
[ ]:
```
network.
```

[40]:
```
# Get the predicted labels for the test set
y_pred = network.predict(x_test).argmax(axis=1)

# Create the confusion matrix
cm = confusion_matrix(np.argmax(y_test, axis=1), y_pred)

# Plot the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', cbar=False)
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix')
plt.show()
```

```
313/313 [==============================] - 3s 11ms/step
```



The model had 98.4 % accuracy, meaning it was able to classify 98.4 % of the images in the test set. The confusion matrix shows high numbers along the diagnal which indicates correct predictions acrossed the labels.