```
#Week 11: Brandon Mather DSC550-T301
```

```
import numpy as np
from keras.datasets import mnist
from matplotlib import pyplot as plt
from keras.models import Sequential
from keras.layers import Dense, Dropout, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D
from keras.utils import np_utils
from keras import backend as K
```

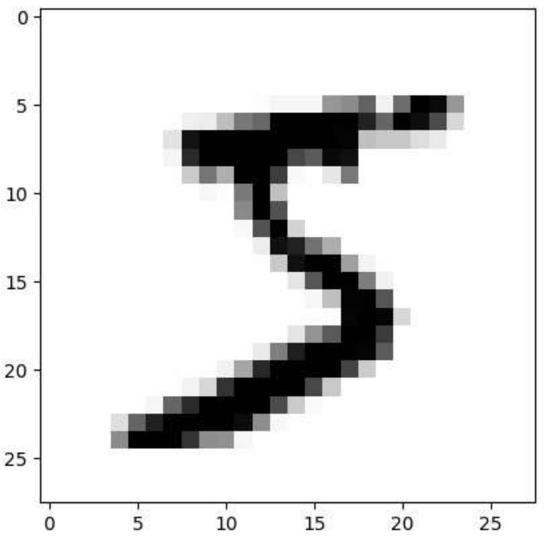
```
#1) Load the MNIST data set.
K.set_image_data_format("channels_first")
np.random.seed(0)
```

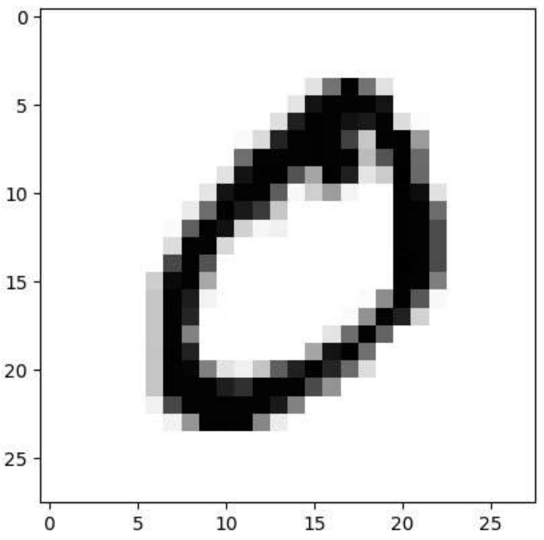
```
channels = 1
height = 28
width = 28
```

```
(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 [==============================] - 1s 0us/step
```

```
#2)Display the first five images in the training data set (see section 8.1 in the Machine Learning with Python Cookbook). Compare these to th
for i in range(5):
    img = data_train[i].reshape((28,28))
    plt.imshow(img, cmap="Greys")
    plt.show()
```





```
#3)Build and train a Keras CNN classifier on the MNIST training set.
data_train = data_train.reshape(data_train.shape[0], channels, height, width)


data_test = data_test.reshape(data_test.shape[0], channels, height, width)


features_train = data_train / 255
features_test = data_test / 255
```

```
target_train = np_utils.to_categorical(target_train)
target_test = np_utils.to_categorical(target_test)
number_of_classes = target_test.shape[1]


network = Sequential()


network.add(Conv2D(filters=64,
kernel_size=(5, 5),
input_shape=(channels, width, height),
activation='relu'))


network.add(MaxPooling2D(pool_size=(2, 2)))


network.add(Dropout(0.5))


network.add(Flatten())


network.add(Dense(128, activation="relu"))


network.add(Dropout(0.5))


network.add(Dense(number_of_classes, activation="softmax"))


network.compile(loss="categorical_crossentropy",
optimizer="rmsprop",
metrics=["accuracy"])


network.fit(features_train,
target_train,
epochs=2,
verbose=0,
batch_size=1000,
validation_data=(features_test, target_test))
```

```
<keras.callbacks.History at 0x7f7743d227d0>
```

```
#4)Report the test accuracy of your model.


score = network.evaluate(data_test, target_test, verbose=0)
score
```

```
[13.384376525878906, 0.9695000052452087]
```

```
#Accuracy is 96.9%


#5)Display a confusion matrix on the test set classifications.


from sklearn.metrics import confusion_matrix


predict = network.predict(data_test)
```

```
313/313 [==============================] - 1s 2ms/step
```

```
confusion_matrix(target_test, predict)


#6)Summarize your results.
```

The results of my model is an accuaracy of 96.9%