Packages need to install in case for the Exercises

1. Scikit
2. Numpy
3. Pandas
4. Keras
5. Tensorflow
6. Time

In the code a seed value has been used to make sure that the results are reproducible.

To run the code we need to place the optdigit.tes and opdigit.tra in the same folder to run it.

And then through terminal we can run the code using command

Python ‘filename’

Filenames are: NeuralNetwork.py and ConvolutionalNeuralNetwork.py

The below code snippet gives an idea that could be changed to get the desired result

|  |
| --- |
| #Adding the input layer and the first hidden layer # classifier.add(Dense(activation="relu", input\_dim=64, units=52, kernel\_initializer="uniform")) classifier.add(Dense(activation="tanh", input\_dim=64, units=32, kernel\_initializer="uniform"))  # Adding the second hidden layer # classifier.add(Dense(activation="relu", units=52, kernel\_initializer="uniform")) # classifier.add(Dense(activation="tanh", units=50, kernel\_initializer="uniform"))  # Adding the third hidden layer # classifier.add(Dense(activation="relu", units=100, kernel\_initializer="uniform")) # classifier.add(Dense(activation="tanh", units=100, kernel\_initializer="uniform"))  # Adding the output layer classifier.add(Dense(activation="softmax", units=10, kernel\_initializer="uniform"))  # compiling the ANN from keras import optimizers sgd = optimizers.SGD(lr=0.01, decay=1e-6, momentum=0.90) # classifier.compile(optimizer='sgd',loss='mean\_squared\_error', metrics=['accuracy']) classifier.compile(optimizer='sgd',loss='categorical\_crossentropy', metrics=['accuracy'])  # Fitting the ANN to the training data classifier.fit(X\_train, y\_train, batch\_size = 5, epochs = 50) |

Different lines are added in this which can be uncommented to get the desired output along with their parameter value in case of the neural network.

For convolutional neural network different code has been provided.

Parameters to be changed are shown in the below code snippet

|  |
| --- |
| # Initializing the CNN classifier = Sequential()  # Step-1 Convolution classifier.add(Convolution2D(32, (3, 3), input\_shape=(8, 8, 1), activation= 'relu'))  # Step-2 Pooling classifier.add(MaxPooling2D(pool\_size=(2, 2)))  # Using Dropout to reduce overfitting classifier.add(Dropout(0.25))  # Step-3 Flattening classifier.add(Flatten())  # Add fully connection layer classifier.add(Dense(activation="relu", units=128))  # Another dropout to test the scenario. classifier.add(Dropout(0.45))  # Output layer classifier.add(Dense(activation="softmax", units=10))  # Compiling the classifier # compiling the ANN from keras import optimizers sgd = optimizers.SGD(lr=0.01, decay=1e-6, momentum=0.90) classifier.compile(optimizer='sgd', loss='categorical\_crossentropy', metrics=['accuracy'])  # fitting the CNN classifier.fit(X\_train, y\_train, batch\_size = 10, epochs = 30) |

The required parameters can be changed to get the expected result as shown in the TEST REPORT.