**Code with comments:**

import numpy as np

from keras.models import Sequential

from keras.layers import Dense

import matplotlib.pyplot as plt

#get the data from the provided csv

features\_and\_targets = np.loadtxt('features\_and\_targets.csv',delimiter=',')

np.random.shuffle(features\_and\_targets)

#split our data

#assign features to X and what is predicted to Y

x = features\_and\_targets[:,0:5]

y = features\_and\_targets[:,5:7]

print(y)

print(y[:,1])

#set Random Seed for reproducible results

np.random.seed(7)

#reset variables

model = None

accuracy = None

loss = None

#train the model over 150 epochs

epochs = 150

#using Sequential Model Type

model = Sequential()

#add model layers to NN Using requred values from BB relu sigmoid, SSR, adam, accuracy metrics

model.add(Dense(4, input\_dim = 5, activation = 'relu'))

model.add(Dense(3, activation = 'relu'))

model.add(Dense(2, activation = 'sigmoid'))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam', metrics = ['accuracy'])

epoch\_hist = model.fit(x, y, epochs = epochs, verbose = 0)

#evaluate will create a list of loss and accuracy of the model to a list we will use later

scores = model.evaluate(x,y)

#print the accuracy

print('\n%s: %.2f%%' % (model.metrics\_names[1], scores[1]\*100))

#create predicted values from the model we just trained

predicted\_targets = model.predict(x)

print(predicted\_targets)

#plot the Model Accuracy and loss of the

accuracy = epoch\_hist.history['accuracy']

loss = epoch\_hist.history['loss']

plt.plot(loss,label='Loss')

plt.plot(accuracy, label='Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy and Loss')

plt.legend()

plt.title("Accuracy and Loss vs Epochs Run")

plt.show()

#plot a scatter chart of the Predicted targets vs the Observed Targets to see how well our NN is performing

plt.scatter(predicted\_targets[:,0],predicted\_targets[:,1], label = "Predicted Targets")

plt.scatter(y[:,0],y[:,1], label = "Observed Targets")

plt.legend()

plt.title('Observed vs Predicted Targets for 150 Epochs')

plt.show()

#### NEXT WE DO 500 ####

#reset variables

model = None

accuracy = None

loss = None

#epochs = None

#assign epochs

epochs = 500

model = Sequential()

model.add(Dense(4, input\_dim = 5, activation = 'relu'))

model.add(Dense(3, activation = 'relu'))

model.add(Dense(2, activation = 'sigmoid'))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam', metrics = ['accuracy'])

epoch\_hist = model.fit(x, y, epochs = epochs, verbose = 0)

score = model.evaluate(x,y)

print((model.metrics\_names[1], score[1]\*100))

predicted\_targets = model.predict(x)

accuracy = epoch\_hist.history['accuracy']

loss = epoch\_hist.history['loss']

plt.plot(loss,label='Loss')

plt.plot(accuracy, label='Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy and Loss')

plt.legend()

plt.title("Accuracy and Loss vs Epochs Run")

plt.show()

plt.scatter(predicted\_targets[:,0],predicted\_targets[:,1], label = "Predicted Targets")

plt.scatter(y[:,0],y[:,1], label = "Observed Targets")

plt.legend()

plt.title('Observed vs Predicted Targets for 500 Epochs')

plt.show()

#### NOW WE DO 1500 ####

#reset variables

model = None

accuracy = None

loss = None

epochs = None

#assign epochs

epochs = 1500

model = Sequential()

model.add(Dense(4, input\_dim = 5, activation = 'relu'))

model.add(Dense(3, activation = 'relu'))

model.add(Dense(2, activation = 'sigmoid'))

model.compile(loss = 'mean\_squared\_error', optimizer = 'adam', metrics = ['accuracy'])

epoch\_hist = model.fit(x, y, epochs = epochs, verbose = 0)

score = model.evaluate(x,y)

print((model.metrics\_names[1], score[1]\*100))

predicted\_targets = model.predict(x)

accuracy = epoch\_hist.history['accuracy']

loss = epoch\_hist.history['loss']

plt.plot(loss,label='Loss')

plt.plot(accuracy, label='Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy and Loss')

plt.legend()

plt.title("Accuracy and Loss vs Epochs Run")

plt.show()

plt.scatter(predicted\_targets[:,0],predicted\_targets[:,1], label = "Predicted Targets")

plt.scatter(y[:,0],y[:,1], label = "Observed Targets")

plt.legend()

plt.title('Observed vs Predicted Targets for 1500 Epochs')

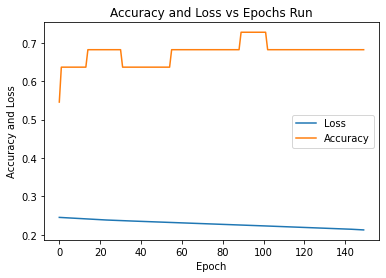
plt.show()

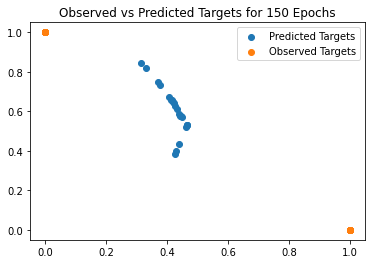
**Results:**

[[0. 1.] [1. 0.] [0. 1.] [0. 1.] [0. 1.] [1. 0.] [0. 1.] [1. 0.] [1. 0.] [1. 0.] [0. 1.] [0. 1.] [0. 1.] [1. 0.] [0. 1.] [0. 1.] [1. 0.] [0. 1.] [0. 1.] [0. 1.] [0. 1.] [1. 0.]] [1. 0. 1. 1. 1. 0. 1. 0. 0. 0. 1. 1. 1. 0. 1. 1. 0. 1. 1. 1. 1. 0.]

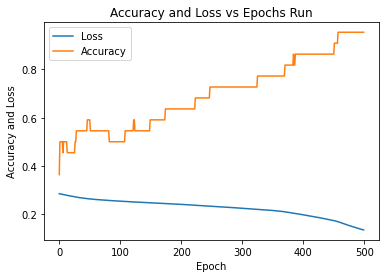
1/1 [==============================] - 1s 630ms/step - loss: 0.2123 - accuracy: 0.6818 accuracy: 68.18%

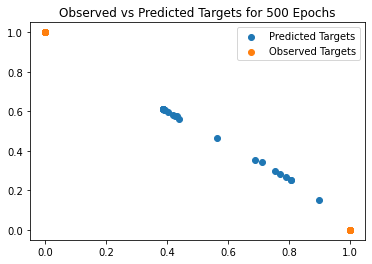
[[0.4670303 0.53315353] [0.42535496 0.38213634] [0.43995544 0.43430507] [0.3776762 0.735404 ] [0.4670303 0.53315353] [0.4382263 0.5844684 ] [0.4190717 0.64651453] [0.40719283 0.67313117] [0.4632273 0.5193043 ] [0.4670303 0.53315353] [0.42714852 0.6279978 ] [0.44504714 0.5778347 ] [0.4214842 0.6410168 ] [0.41520435 0.6552652 ] [0.4134372 0.65923715] [0.44801855 0.57153636] [0.429806 0.39785767] [0.33051962 0.8208277 ] [0.36993068 0.7506969 ] [0.43347514 0.61328745] [0.31421873 0.8458823 ] [0.4411884 0.57819676]]





1/1 [==============================] - 1s 599ms/step - loss: 0.1335 - accuracy: 0.9545 ('accuracy', 95.45454382896423)





1/1 [==============================] - 1s 562ms/step - loss: 0.0254 - accuracy: 1.0000 ('accuracy', 100.0)

