| # first step: create a Sequential object, as a sequence of layers. B/C NN is a sequence of layers. t1 = t.time() grid\_search\_nn1 = pd.DataFrame(columns = ['hd', 'nodes', 'activation', 'dropout', 'batch\_size', 'auc\_train', 'auc\_test1', 'auc\_test2'])   neurons = [4, 6] activations = ['relu', 'tanh'] dropout = [.5, 0] batch\_sizes = [100, 10000]  row = 0 **for** i **in** neurons:  **for** a **in** activations:  **for** d **in** dropout:  **for** s **in** batch\_sizes:    grid\_search\_nn1.loc[row, 'nodes'] = i  grid\_search\_nn1.loc[row, 'activation'] = a  grid\_search\_nn1.loc[row, 'dropout'] = d  grid\_search\_nn1.loc[row, 'batch\_size'] = s   classifier = Sequential()   # add the first hidden layer  classifier.add(Dense(units=i,kernel\_initializer='glorot\_uniform',  activation = a))   classifier.add(Dropout(d))   # add the second hidden layer  classifier.add(Dense(units=i,kernel\_initializer='glorot\_uniform',  activation = a))   classifier.add(Dropout(d))  # add the output layer  classifier.add(Dense(units=1,kernel\_initializer='glorot\_uniform',  activation = 'sigmoid'))   # add additional parameters  classifier.compile(optimizer='adam',loss='binary\_crossentropy',metrics=['accuracy', 'FalseNegatives'])   # train the model  classifier.fit(xtrain\_n\_df,ytrain,batch\_size=s,epochs=20)   grid\_search\_nn1.loc[row, 'auc\_train'] = roc\_auc\_score(ytrain, classifier.predict(xtrain\_n\_df))  grid\_search\_nn1.loc[row, 'auc\_test1'] = roc\_auc\_score(ytest1, classifier.predict(xtest1\_n\_df))  grid\_search\_nn1.loc[row, 'auc\_test2'] = roc\_auc\_score(ytest2, classifier.predict(xtest2\_n\_df))  print("Best Parameter Iteration", row, "AUC Train", roc\_auc\_score(ytrain, classifier.predict(xtrain\_n\_df)))   row += 1  grid\_search\_nn1['hd'] = 2 t2 = t.time() print(t2 - t1) |
| --- |