```
"""
Created on Mon Apr 15 19:43:04 2019
Updated on Wed Jan 29 10:18:09 2020
@author: created by Sowmya Myneni and updated by Dijiang Huang
"""
"""
Updated on November 22, 2022
@author:Anirudh Kumar
Added preprocessing, splitting of testing and training datasets,
and selection of Scenario.
"""
########################################
# Part 1 - Datasets Pre-Processing
########################################

# To load a dataset file in Python, you can use Pandas. Import
pandas using the line below
import pandas as pd
# Import numpy to perform operations on the dataset
import numpy as np

# Controls which scenario we want to run.
# Accepts a, b or c as input
ScenarioA = ['Training-a1-a3', 'Testing-a2-a4']
ScenarioB = ['Training-a1-a2', 'Testing-a1']
ScenarioC = ['Training-a1-a2', 'Testing-a1-a2-a3']

while 1:
   Scenario = input ('Please enter the scenario you wish to run -
either a, b or c:')

   if Scenario.lower() == 'a':
       TrainingData = ScenarioA[0]
       TestingData  = ScenarioA[1]
       break
   elif Scenario.lower() == 'b':
       TrainingData = ScenarioB[0]
       TestingData  = ScenarioB[1]
       break
   elif Scenario.lower() == 'c':
       TrainingData = ScenarioC[0]
       TestingData  = ScenarioC[1]
       break

# Batch Size
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/

```
BatchSize=10
# Epohe Size
NumEpoch=10


# Import the Dataset specified in the TrainingData variable.
# It will be automatically set by choosing the appropriate
scenario at execution
#   time.
# If the dataset file has header, then keep header=0 otherwise use
header=none
# reference: https://www.shanelynn.ie/select-pandas-dataframe-
rows-and-columns-using-iloc-loc-and-ix/

import data_preprocessor as dp
X_train, y_train = dp.get_processed_data(TrainingData+'.csv',
'./categoryMappings/', classType ='binary')
X_test,  y_test  = dp.get_processed_data(TestingData+'.csv',
'./categoryMappings/', classType ='binary')

# The next section from fnn_sample.py is not required, as data is
already preprocessed

# Encoding categorical data (convert letters/words in numbers)
# Reference: https://medium.com/@contactsunny/label-encoder-vs-
one-hot-encoder-in-machine-learning-3fc273365621
# The following code work without warning in Python 3.6 or older.
Newer versions suggest to use ColumnTransformer
'''
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
le = LabelEncoder()
X[:, 1] = le.fit_transform(X[:, 1])
X[:, 2] = le.fit_transform(X[:, 2])
X[:, 3] = le.fit_transform(X[:, 3])
onehotencoder = OneHotEncoder(categorical_features = [1, 2, 3])
X = onehotencoder.fit_transform(X).toarray()
'''
# The following code work Python 3.7 or newer
#from sklearn.preprocessing import OneHotEncoder
#from sklearn.compose import ColumnTransformer
#ct = ColumnTransformer(
#    [('one_hot_encoder', OneHotEncoder(), [1,2,3])],    # The
column numbers to be transformed ([1, 2, 3] represents three
columns to be transferred)
#    remainder='passthrough'                          # Leave the
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/

```
rest of the columns untouched
#)
#X = np.array(ct.fit_transform(X), dtype=np.float)

# Splitting the dataset into the Training set and Test set (75%
of data are used for training)
# reference: https://scikit-
learn.org/stable/modules/generated/sklearn.model_selection.train_
test_split.html
#from sklearn.model_selection import train_test_split
#X_train, X_test, y_train, y_test = train_test_split(X, b,
test_size = 1.0,train_size = 1.0, random_state = 0)


# Perform feature scaling. For ANN you can use StandardScaler,
for RNNs recommended is
# MinMaxScaler.
# referece: https://scikit-
learn.org/stable/modules/generated/sklearn.preprocessing.Standard
Scaler.html
# https://scikit-learn.org/stable/modules/preprocessing.html
#from sklearn.preprocessing import StandardScaler
#sc = StandardScaler()
#X_train = sc.fit_transform(X_train)  # Scaling to the range [0,1]
#X_test = sc.fit_transform(X_test)


########################################
# Part 2: Building FNN
########################################

# Importing the Keras libraries and packages
#import keras
from keras.models import Sequential
from keras.layers import Dense

# Initialising the ANN
# Reference: https://machinelearningmastery.com/tutorial-first-
neural-network-python-keras/
classifier = Sequential()

# Adding the input layer and the first hidden layer, 6 nodes,
input_dim specifies the number of variables
# rectified linear unit activation function relu, reference:
https://machinelearningmastery.com/rectified-linear-activation-
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/

```
function-for-deep-learning-neural-networks/
classifier.add(Dense(units = 6, kernel_initializer = 'uniform',
activation = 'relu', input_dim = len(X_train[0])))

# Adding the second hidden layer, 6 nodes
classifier.add(Dense(units = 6, kernel_initializer = 'uniform',
activation = 'relu'))

# Adding the output layer, 1 node,
# sigmoid on the output layer is to ensure the network output is
between 0 and 1
classifier.add(Dense(units = 1, kernel_initializer = 'uniform',
activation = 'sigmoid'))

# Compiling the ANN,
# Gradient descent algorithm , Reference:
https://machinelearningmastery.com/adam-optimization-algorithm-
for-deep-learning/
# This loss is for a binary classification problems and is defined
in Keras as , Reference: https://machinelearningmastery.com/how-
to-choose-loss-functions-when-training-deep-learning-neural-
networks/
classifier.compile(optimizer = 'adam', loss =
'binary_crossentropy', metrics = ['accuracy'])

# Fitting the ANN to the Training set
# Train the model so that it learns a good (or good enough)
mapping of rows of input data to the output classification.
# add verbose=0 to turn off the progress report during the
training
# To run the whole training dataset as one Batch, assign batch
size: BatchSize=X_train.shape[0]
classifierHistory = classifier.fit(X_train, y_train, batch_size =
BatchSize, epochs = NumEpoch)

# evaluate the keras model for the provided model and dataset
loss, accuracy = classifier.evaluate(X_train, y_train)
print('Print the loss and the accuracy of the model on the
dataset')
print('Loss [0,1]: %.4f' % (loss), 'Accuracy [0,1]: %.4f' %
(accuracy))

########################################
# Part 3 - Making predictions and evaluating the model
########################################
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/

```
# Predicting the Test set results
y_pred = classifier.predict(X_test)
y_pred = (y_pred > 0.9)   # y_pred is 0 if less than 0.9 or equal
to 0.9, y_pred is 1 if it is greater than 0.9
# summarize the first 5 cases
#for i in range(5):
#    print('%s => %d (expected %d)' % (X_test[i].tolist(),
y_pred[i], y_test[i]))

# Making the Confusion Matrix
# [TN, FP ]
# [FN, TP ]
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print('Print the Confusion Matrix:')
print('[ TN, FP ]')
print('[ FN, TP ]=')
print(cm)

########################################
# Part 4 - Visualizing
########################################

# Import matplot lib libraries for plotting the figures.
import matplotlib.pyplot as plt

# You can plot the accuracy
print('Plot the accuracy')
# Keras 2.2.4 recognizes 'acc' and 2.3.1 recognizes 'accuracy'
# use the command python -c 'import keras;
print(keras.__version__)' on MAC or Linux to check Keras' version
plt.plot(classifierHistory.history['accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train'], loc='upper left')
plt.savefig('accuracy_sample.png')
plt.show()

# You can plot history for loss
print('Plot the loss')
plt.plot(classifierHistory.history['loss'])
plt.title('model loss')
plt.ylabel('loss')
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/

```
plt.xlabel('epoch')
plt.legend(['train'], loc='upper left')
plt.savefig('loss_sample.png')
plt.show()
```

This study source was downloaded by 100000869502194 from CourseHero.com on 11-30-2025 19:23:14 GMT -06:00

https://www.coursehero.com/file/210839344/fnn-samplepy/