```
% Implementing ANN model for classification

close all; clear all; clc;
```

reading data

```
df = readmatrix('scaled_df.csv');

%Selecting all 4424 rows and excluding the index numbers in the first row
x = df(2:4425,1:34);
y = df(2:4425,35:37);
```

splitting train and test set

```
len = length(y);
P = 0.8;  % percentage

rng(11) % to get the same shuffle every time
idx = randperm(len);  % generating random numbers for index

% selecting 80% of the data to train
x_train = x( idx(1:round(P*len)), :);
y_train = y( idx(1:round(P*len)), :);

% selecting the remaining 20% for test
x_test = x( idx(round(P*len)+1:end), :);
y_test = y( idx(round(P*len)+1:end), :);
```

creating ANN model and training it with train dataset

```
% assigning the number of neurons in hidden layer
hidden_size = 20;

% measuring the start time of training
start_time = tic;

% define the artificial neural network architecture
% patternnet() if a special variation of feedforwardnet() which is specifically
% used for classification and therefore by default uses Cross Entropy Loss
% function with sigmoid activation function in hidden layer and softmax
% function in output layer. (Note: These are the set of functions used in
% Python code as well.)



net = patternnet(hidden_size);

fprintf('\n Activation function in hidden layer: %s', net.layers{1}.transferFcn);
fprintf('\n Activation function in output layer: %s', net.layers{2}.transferFcn);

[net, tr] = train(net, x_train', y_train');

% time taken to train the model
```

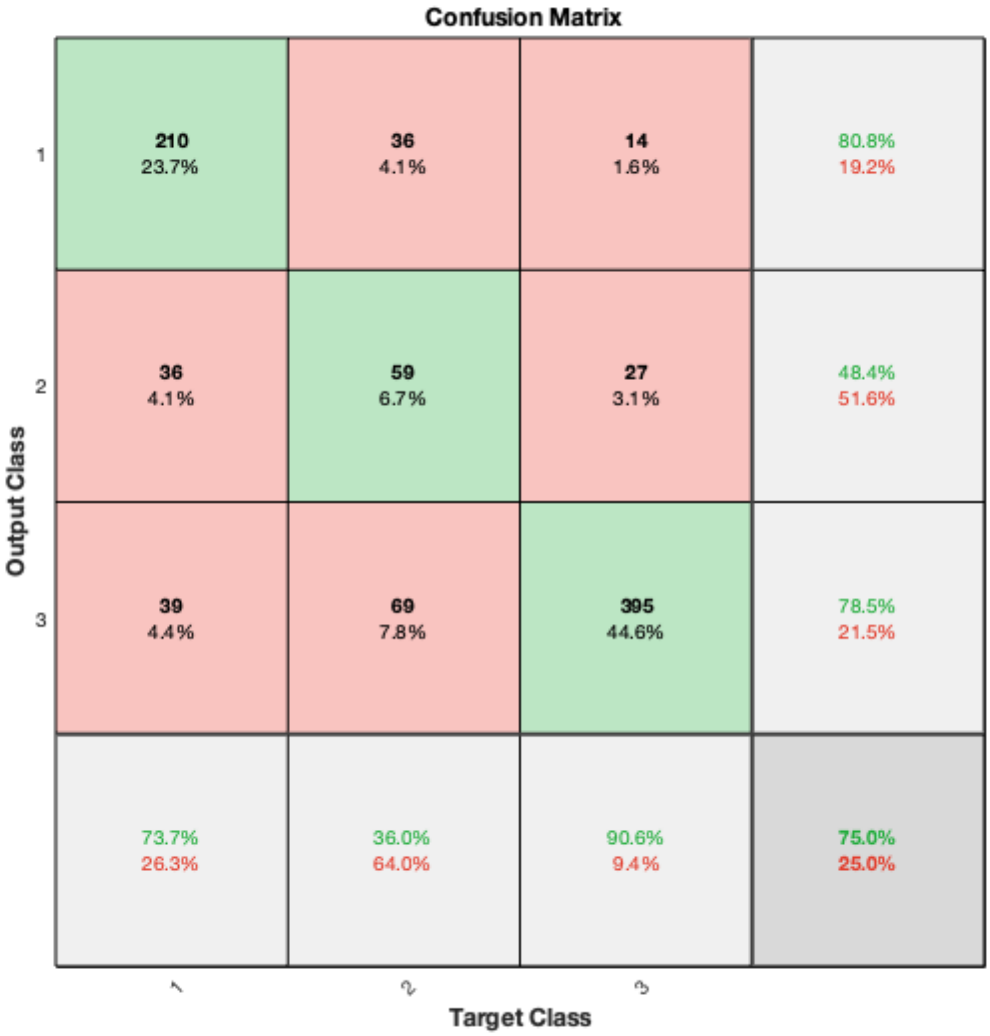
```
time_taken = toc(start_time);
fprintf('\n Time taken to train the model: %2.2f seconds', time_taken);

% view(net);
```

```
 Activation function in hidden layer: tansig
 Activation function in output layer: softmax
 Time taken to train the model: 4.47 seconds
```

testing the model with test dataset

```
y_pred = net(x_test');


[~,y_max] = max(y_test,[],2);

[~,pred_max] = max(y_pred',[],2);

% calculating the test accuracy
accuracy = (sum(y_max==pred_max) / size(y_max,1)) * 100;
fprintf('\n Testing Accuracy: %2.2f', accuracy);

loss = perform(net,y_test',y_pred);
fprintf('\n Testing Loss: %2.2f', loss);

% confusion matrix for test set
plotconfusion(y_test', y_pred);
```

```
 Testing Accuracy: 75.03
 Testing Loss: 0.20
```



Published with MATLAB® R2022b