MLP Model Testing

This script is used for testing the MLP trained on the Training data in the 'NC_MultLayerPerceptron_Methodology.m' on the unseen Testing data. The script follows these steps:

1. Loads Testing data

2. Obtain best parameters from 'MLP_Results' in training and grid search

3. The best MLP network, 'bestnet', saved in 'NC_MultiLayerPerceptron_Methodology.m' is loaded and predictions are made from the test set predictors.

4. The ROC curve is created based on the comparison between predictions from test data and actual test data outputs.

5. The confusion matrix is created and visualised and accuracy, precision, sensitivity and specificity caluclated.

1. Load Data

```
%Load the testing data

load('Testing.mat');
x_test = Testing(:, 1:42)';
t_test = Testing(:, 43)';
```

2. Get Best Parameters

```
%From the 'MLP_Results' table saved at the end of
%'NC_MultiLayerPerceptron_Methodology', retrive the paraemeters that
%resulted in best average 10-fold accuracy and print these.

%load MLP_Results Table
load('MLP_Results.mat')

%find row with best accuracy, and the index
[best_acc, best_acc_idx] = max(MLP_Results.Avg_Acc);

%get row using the index
best_parameters = MLP_Results{best_acc_idx,:};
best_struct = [best_parameters(1), best_parameters(1), best_parameters(1)];
best_momentum = best_parameters(4);
best_learningrate = best_parameters(5);

%print the optimum MLP hyperparameters
fprintf('Best Parameters: \n')
```

```
Best Parameters:
```

```
fprintf('Structure: %s')
```

```
Structure:
```

```
best_struct
```

```
best_struct = 1×3
    50    50    50
```

```
fprintf('Momentum: %2f \n',best_momentum)
```

```
Momentum: 0.500000
```

```
fprintf('Learning Rate: %2f \n\n',best_learningrate)
```

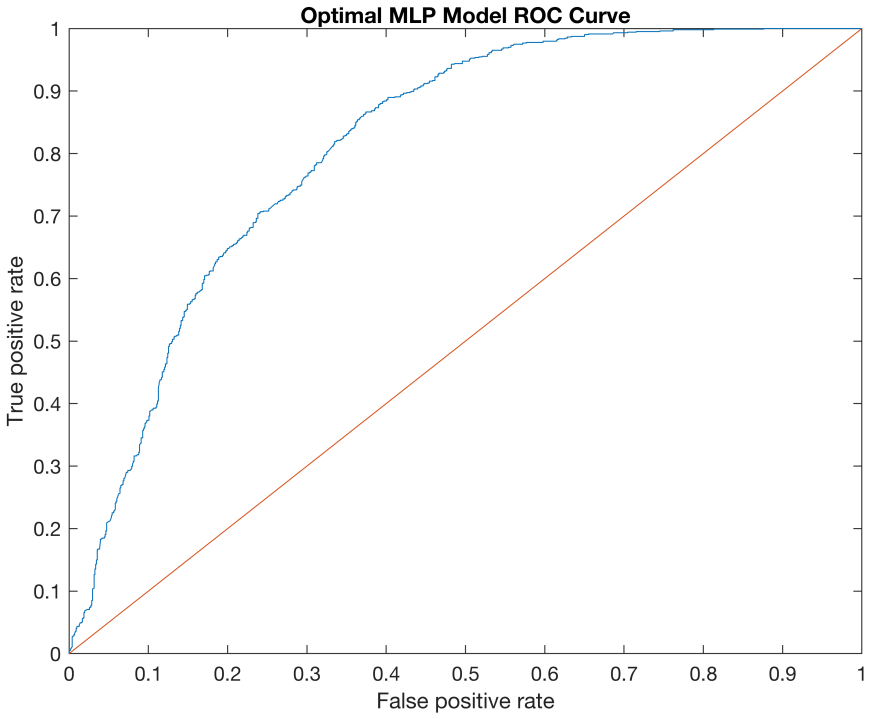
```
Learning Rate: 0.900000
```

3. Load Best MLP Model

```
%The best neural network which was modelled using these parameters was saved
%at the end of 'NC_MultiLayerPerceptron_Methodology'. Load this network and
%make predictions on the unseen test data.

%load the neural network made from optimal hyperparameters
load('best_net.mat');

%make predictions using test data
pred_test = best_net(x_test);
```

4. ROC Curve

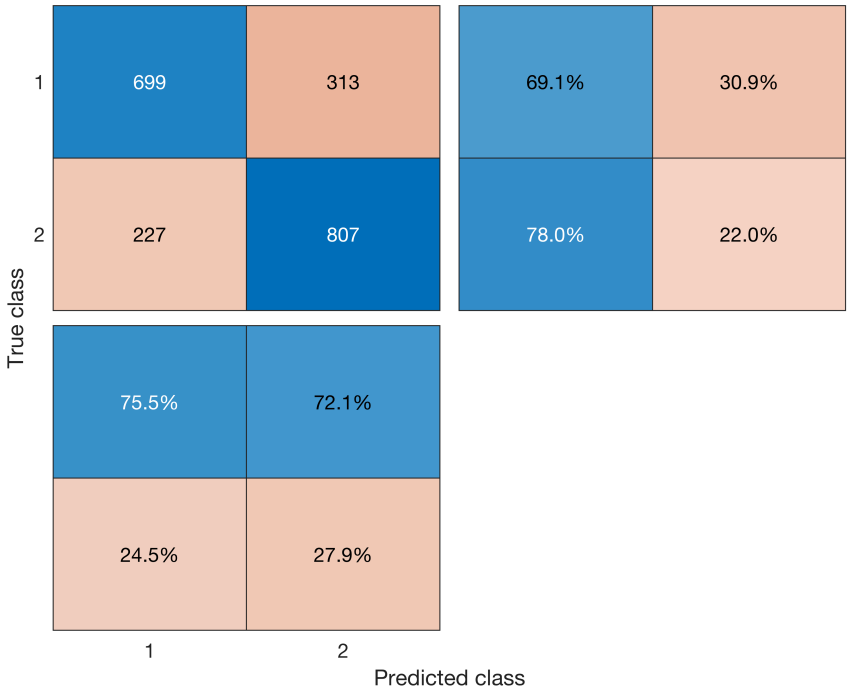
```
%Plot the ROC curve to assess model performance of false positive rate and
%true positive rate

%ROC curve
f2 = figure;
[TP,FP,T,AUC] = perfcurve(t_test , pred_test, '1');
plot(TP,FP, [0 1],[0 1])
xlabel('False positive rate')
ylabel('True positive rate')
title('Optimal MLP Model ROC Curve')
```



3. Confusion Matrix

A confusion matrix can be polotted to get info on TP, TN, FP and FN. From these, improtant metrics are derived to assess model performance.

```
f1 = figure;
%create confusion matrix
C = confusionmat(t_test, round(pred_test));
%plot chart from confusion matrix
D = confusionchart(C,'RowSummary','row-normalized','ColumnSummary','column-normalized');
```




```
%create model generalisation statistics
accuracy = (C(1,1) + C(2,2)) / (C(1,1) + C(1, 2) + +C(2, 1) + C(2,2));
precision = C(1,1) / (C(1,1) + C(1,2));
sensitivity = (C(1,1)) / (C(1, 1)+C(2, 1));
specificity = (C(2,2)) / (C(2, 2)+C(1, 2));
f1_score = 2 * ((precision * sensitivity)/(precision + sensitivity));

%print model generalisation statistics
fprintf('\nAccuracy: %2f \n',accuracy) %proportion of correct predictions
```

```
Accuracy: 0.736070
```

```
fprintf('Precision: %2f \n',precision) %proportion of positive results that were correctly classified
```

```
Precision: 0.690711
```

```
fprintf('Sensitivity: %2f \n',sensitivity) %amount of customers leaving that were correctly idenified
```

```
Sensitivity: 0.754860
```

```
fprintf('Specificity: %2f \n',specificity) %amount of customers staying that were correctly identified
```

```
Specificity: 0.720536
```

```
fprintf('F1: %2f \n',f1_score)
```

```
F1: 0.721362
```

```
fprintf('AUC: %2f \n',AUC)
```

```
AUC: 0.813113
```