Contents

Apply Logistic Regression on Recidivism Data %%

```
clear all;
clc;
close all;
```

Import train and test data. %

```
train_data = readtable("Recidivismtrainset.csv");
test_data = readtable("Recidivismtestset.csv");
```

Split Predictor Variables and Response Variable in train % and test data. %

```
x_train = train_data(:,1:end-1);
y_train = train_data(:,end);
x_test = test_data(:,1:end-1);
y_test = test_data(:,end);
```

Fit train data to Generalized Linear Model, a into which category % Logistic Regression falls. %

```
rng(1);
tic
Mdl1 = fitglm(train_data,'Distribution','binomial','Link','logit');
toc

% Code reference: %
% Statistics and Machine Learning Toolbox™ User's Guide %
% Revision September 2021, R2021b, Chapter 12 %
```

```
Warning: Regression design matrix is rank deficient to within machine
precision.
Elapsed time is 0.493700 seconds.
```

Predict Response for train data using the Model. %

```
yfittrain = predict(Mdl1,x_train);
```

It is observed that the predicted values of the response variable % are numbers between 0 and 1 as these are probabilities. Whereas % the actual values in the response variable are 0 or 1. So the % predicted values are to be converted to binary classification of % 0 and 1. For this a threshold is to be taken such that the % predicted values greater than or equal to this threshold can be % considered as 1 and the predicted values less than it can be% considered as 0. %

```
% After numerous iterations in the range of 0.1 to 0.9, 0.52 was %
% found to be the best threshold. %

% Assign a variable to the considered threshold. %

P1 = 0.52;
```

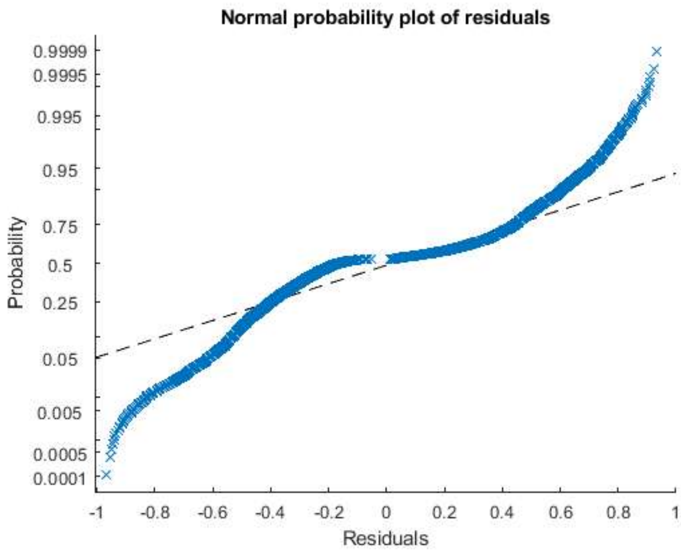
Convert the predicted values of response varibale of train set % into 0 and 1 unsing the assigned threshold. %

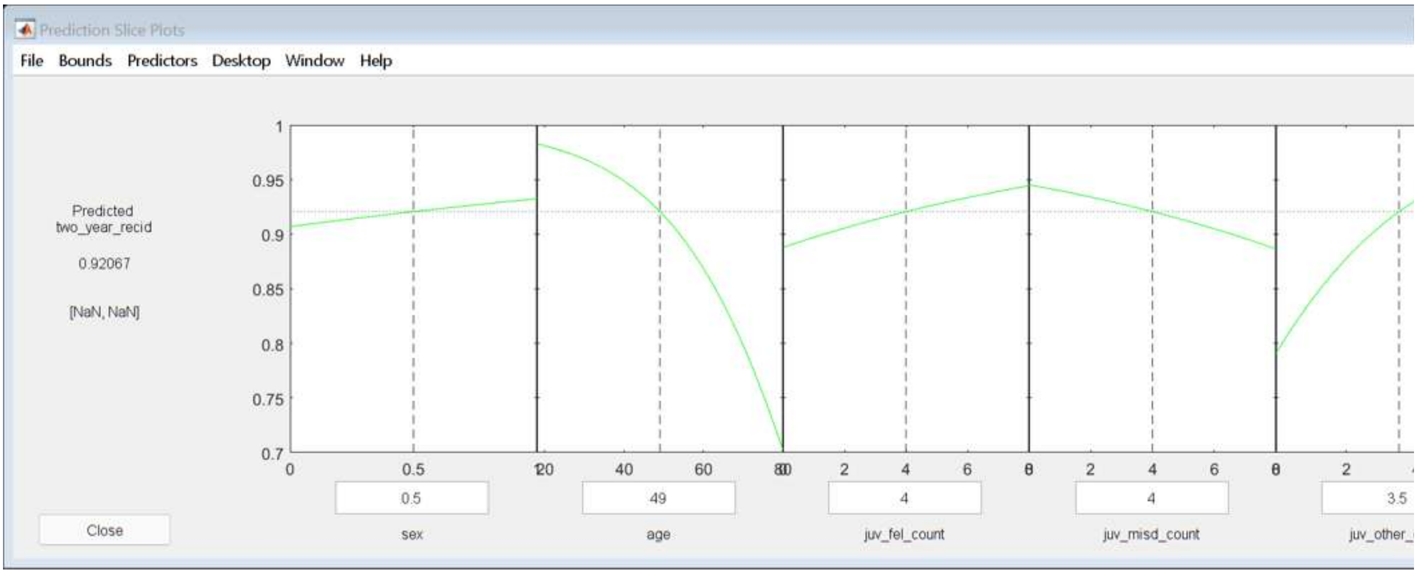
```
v1train = logical((predict(Mdl1,x_train)>= P1));

% Compare the predicted values of the response variables with %
% the original values and create a logical vector of the %
% comparision, such that if both prediction value and original %
% value are same then the instance is assigned as 1, else 0. %

v2train = (v1train == y_train.two_year_recid);

% Calculate accuracy of the Model for train data using the %
% comparision Vector. %

trainaccuracy = sum(v2train)/size(v1train,1);

% Calculate error of the Model for train data using the %
% comparision Vector. %

trainError = 1- trainaccuracy;
```

Examin Quality of fitted Model. %

```
% Plot Diagnostic Plots. %

figure
plotSlice(Mdl1)
plotDiagnostics(Mdl1)
```

```
% Diagnostic plots indicate that removing any variables from %
% the predictors may not improve the Model performance. %

% Plot Residual Plots. %

plotResiduals(Mdl1,'probability')

% The plot of normal probability of residuals indicates that %
% Model cannot be improved with normalization of data. %

% Code reference: %
% Statistics and Machine Learning Toolbox™ User's Guide %
% Revision September 2021, R2021b, Chapter 12 %
```





Predicted
two_year_recid

0.92067

[NaN, NaN]

sex: 0.5

age: 49

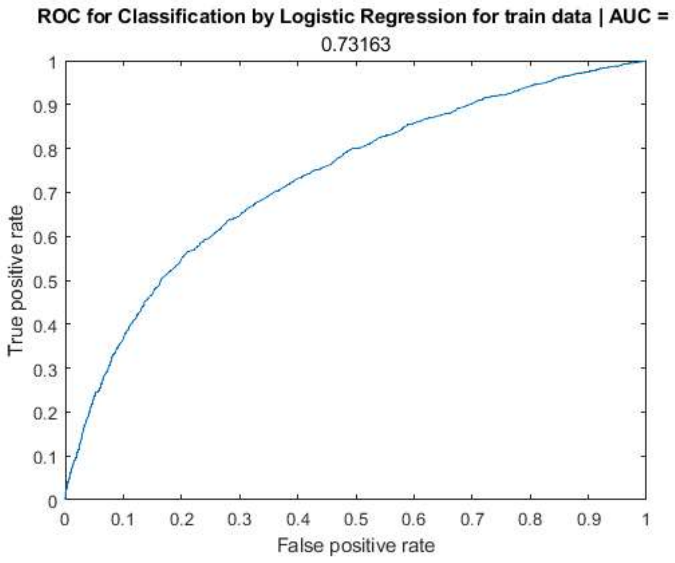
juv_fel_count: 4

juv_misd_count: 4

juv_other_: 3.5

Evaluation metrics for Model Performance. %

```
% Check AUC of the Model for training data. %

scoretrain = Mdl1.Fitted.Probability;
[Xtr,Ytr,Ttr,AUCtr] = perfcurve(y_train.two_year_recid,scoretrain,'1');

% Plot ROC of the Model for training data. %

plot(Xtr,Ytr)
xlabel('False positive rate')
ylabel('True positive rate')
title('ROC for Classification by Logistic Regression for train data | AUC = ',AUCtr)

% Code reference: %
% https://uk.mathworks.com/help/stats/perfcurve.html?s_tid=doc_ta#bunsogv-AUC
%
```



With an AUC of 0.7316 the Model appears to perform well on the % training data. So, the response can be predicted for test data. %

```
% Predict Response for test data using the Model. %

rng(1);
tic
yfittest = predict(Mdl1,x_test);
toc
```

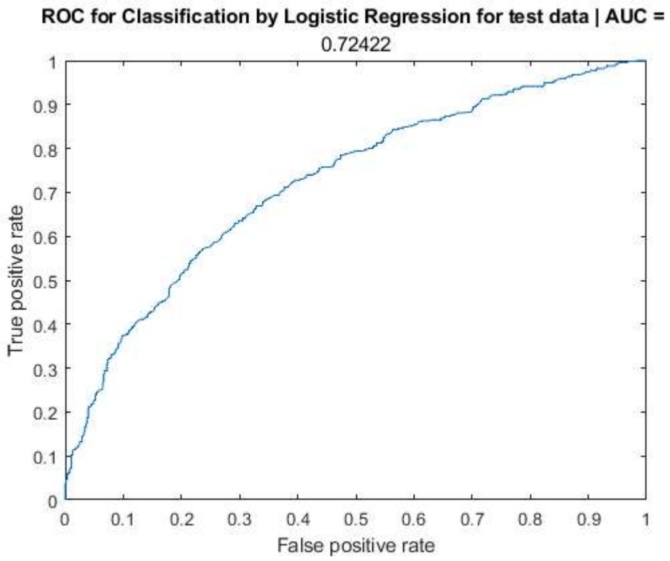
```
Elapsed time is 0.023258 seconds.
```

Evaluate Model performance for test data. %

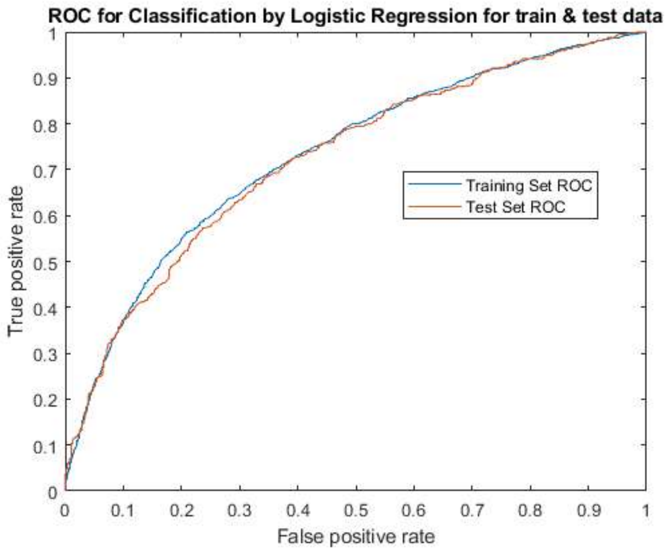
```
% Convert predicted values of response variables for test data %
% into 0 and 1. %

v1test = logical((predict(Mdl1,x_test)>= P1));

% Comparision with original values of response variable from %
% test data. %

v2test = (v1test == y_test.two_year_recid);

% Calculate accuracy of the Model for test data using the %
% comparision Vector. %

testaccuracy = sum(v2test)/size(v1test,1);

% Calculate error of the Model for test data using the %
% comparision Vector. %

testError = 1- testaccuracy;
```

Check AUC of the Model for test data. %

```
[Xte,Yte,Tte,AUCte] = perfcurve(y_test.two_year_recid,yfittest,'1');

% Plot ROC of the Model for test data. %

plot(Xte,Yte)
xlabel('False positive rate')
ylabel('True positive rate')
title('ROC for Classification by Logistic Regression for test data | AUC = ',AUCte)
```



```
plot(Xtr,Ytr)
hold on
plot(Xte,Yte)
legend('Training Set ROC', 'Test Set ROC',Location='best')
xlabel('False positive rate')
ylabel('True positive rate')
title('ROC for Classification by Logistic Regression for train & test data')
hold off
```

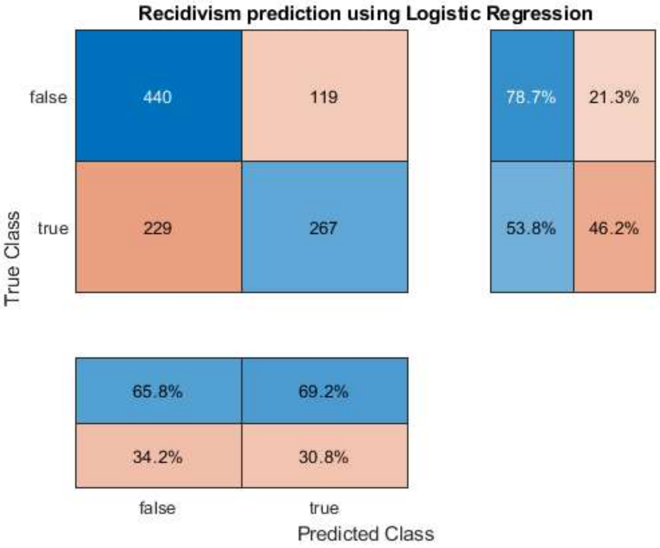


Metrics for Model performance for test data. %

```
% Convert table of response variable for test data into a %
% logical array, to use further. %

y_test_ar = table2array(y_test);
y_test_lg = logical(y_test_ar);

% Plot and assign Confusion Matrix for test data. %

conchart = confusionchart(y_test_lg,v1test);
conchart.Title = 'Recidivism prediction using Logistic Regression'
conchart.RowSummary = 'row-normalized'
conchart.ColumnSummary = 'column-normalized'

% Assing variable to Confusion Matrix for test data. %

confmat = confusionmat(y_test_lg,v1test);
confmat;

% Assigning variables to Components of Confusion Matrix %
% viz., True Negative, True Positive, False Negative, %
% and False Poistive. %

TN = confmat(1,1);
TP = confmat(2,2);
FN = confmat(2,1);
FP = confmat(1,2);

% Calculating other Model Evaluation Metrics for test data. %
Sensitivity = (TP/(TP + FN));
```

```
Specificity = (TN/(TN + FP));
Precision = (TP/(TP + FP));

% Formula Reference: %
% https://en.wikipedia.org/wiki/Confusion_matrix %
% Code Reference: %
% https://uk.mathworks.com/help/stats/confusionchart.html?s_tid=doc_ta %
```

```
conchart = 

  ConfusionMatrixChart (Recidivism prediction using Logi…) with properties:

    NormalizedValues: [2×2 double]
         ClassLabels: [2×1 logical]

  Use GET to show all properties


conchart = 

  ConfusionMatrixChart (Recidivism prediction using Logi…) with properties:

    NormalizedValues: [2×2 double]
         ClassLabels: [2×1 logical]

  Use GET to show all properties


conchart = 

  ConfusionMatrixChart (Recidivism prediction using Logi…) with properties:

    NormalizedValues: [2×2 double]
         ClassLabels: [2×1 logical]

  Use GET to show all properties
```



END %%

Published with MATLAB® R2021a