```
df = readtable('Heart_Disease_Dataset.csv');
%We import the dataset in a table format
%The link of the dataset: https://www.kaggle.com/datasets/pritsheta/heart-
attack

%%
%We first delete the rows with 'thal'==0 because based on the description
%of the dataset there should be only values between 1-3.
%DATA PRE-PROCESSING - Deleting rows with 'thal'=0
delete_rows = df.thal==0;
df(delete_rows,:) = [];
%%
%We will first apply some exploratory data analysis methods to get a better
%understanding and insights of the structure of the dataset and the
%distribution of the variables.

%EXPLORATORY DATA ANALYSIS - Distribution of the 'output' variable
target_label_target = categorical(df.('target'));
figure;
histogram(target_label_target,'BarWidth',0.6,'EdgeColor','none','FaceColor'
,'blue');
xlabel('Class Labels');
ylabel('Total Observations');
title('Distribution of the class labels for the target variable');
%This histogram shows us the distribution of our target variable.
%We can see that the dataset is pretty balanced which is helpful for our
%predictions.

%%
% EXPLORATORY DATA ANALYSIS - Distribution of the 'age' variable for
%the variable 'output' =='1' and 'output'=='0' separately
target_label_age = df.('age');
figure;
subplot(1,2,1);
boxplot(target_label_age(target_label_target=='1'));
ylabel('Age');
title('Age for target=1');

subplot(1,2,2)
boxplot(target_label_age(target_label_target=='0'));
ylabel('Age');
title('Age for target=0');
%These boxplots show us the distribution of the ages for patients with and
%without a heart attack separately. We want to see if the age distribution
%of the patients with and without a heartattack is significantly different.
%It seems that the average age of people with and without a heart attack is
%quite similar but the age distribution of patients with heart attack is
%wider.
% Reference link for boxplot:
https://uk.mathworks.com/help/stats/boxplot.html
% Reference link for subplot function:
https://uk.mathworks.com/help/matlab/ref/subplot.html

%%
%EXPLORATORY DATA ANALYSIS - Number of males and females
target_label_sex = categorical(df.('sex'));
figure;
histogram(target_label_sex(target_label_target=='1'),'BarWidth',0.6,'EdgeCo
lor','none','FaceColor','blue');
```

```
hold on;
histogram(target_label_sex(target_label_target=='0'),'BarWidth',0.6,'EdgeCo
lor','none','FaceColor','red');
hold off;
xlabel('Class Label');
ylabel('Total Observations');
title('male=1,female=0');
legend('target=1','target=0');
xticks(categories(target_label_sex));
%This histogram shows us the number of male and female patients for
%patients with a heart attack separately. We want to see if the
%distribution of the sex differs between patients with and without the
%disease. Based on this spesific dataset, it seems that a lot of women
%have a high risk of getting a heart attack.

% Reference link for hold function:
https://uk.mathworks.com/help/matlab/ref/hold.html

%%
%EXPLORATORY DATA ANALYSIS - Distribution of the slope level
target_label_slope = categorical(df.('slope'));
figure;
histogram(target_label_slope(target_label_target=='1'),'BarWidth',0.6,'Edge
Color','none','FaceColor','blue');
hold on;
histogram(target_label_slope(target_label_target=='0'),'BarWidth',0.6,'Edge
Color','none','FaceColor','red');
hold off;
xlabel('Class Label');
ylabel('Total Observations');
title('Slope Level');
legend('target=1','target=0');
xticks(categories(target_label_slope));
%These boxplots compare the distribution of the slope for people with and
%without a heart attack. We want to see if the distribution of the slope is
%significantly different for people with and without the disease. Based on
% %the distribution of the values, it seems that that the higher the slope
% of the peak exercise is, the more chance there is for someone to get a
% heart attack.

% Reference link for histogram:
https://uk.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.hi
stogram.html
% Reference link for subplot function:
https://uk.mathworks.com/help/matlab/ref/subplot.html

%%
%EXPLORATORY DATA ANALYSIS - Distribution of max heart rate reached
target_label_thalach = df.('thalach');
figure;
subplot(1,2,1);
boxplot(target_label_thalach(target_label_target=='1'));
ylabel('Max Heart Rate Achieved');
title('thalach for target=1');

subplot(1,2,2);
boxplot(target_label_thalach(target_label_target=='0'));
ylabel('Max Heart Rate Achieved');
title('thalach for target=0');
```

```
%These boxplots compare the maximum heart rate reached for people with and
%without a heart attack. We want to see if the distribution is different
%for patients with and without the disease. There seem to be some
%differences based on the distribution of the values. The average maximum
%heart rate for people with a heart attack and the general distribution of
%the values is a lot higher for people with a heart attack. So we expect to
%see a significant positive correlation between the variables.

% Reference link for boxplot:
https://uk.mathworks.com/help/stats/boxplot.html
% Reference link for subplot function:
https://uk.mathworks.com/help/matlab/ref/subplot.html

%%
%EXPLORATORY DATA ANALYSIS - Chest pain levels
target_label_cp = categorical(df.('cp'));
figure;
subplot(2,1,1)
histogram(target_label_cp(target_label_target=='1'),'BarWidth',0.6,'EdgeCol
or','none','FaceColor','blue');
hold on;
histogram(target_label_cp(target_label_target=='0'),'BarWidth',0.6,'EdgeCol
or','none','FaceColor','red');
hold off;
xlabel('Class Label');
ylabel('Total Observations');
title('Chest Pain Level');
legend('target=1','target=0');
xticks(categories(target_label_cp));

target_label_restecg = categorical(df.('restecg'));
subplot(2,1,2)
histogram(target_label_restecg(target_label_target=='1'),'BarWidth',0.6,'Ed
geColor','none','FaceColor','blue');
hold on;
histogram(target_label_restecg(target_label_target=='0'),'BarWidth',0.6,'Ed
geColor','none','FaceColor','red');
hold off;
xlabel('Class Label');
ylabel('Total Observations');
title('Resting Electrocardiogram Results');
legend('target=1','target=0');
xticks(categories(target_label_restecg));
%These histograms compare the chest pain levels for people with and without
%a heartattack combined with the comparison of the resting
%electrocardiogram results. We want to see how these variables differ for
%people with and without a heartattack.
%It seems that people with a heart attack can suffer all levels of chest
%pains. As it was expected, people without a heart attack have mainly level
%0 chest pains. So we can see that the chest pain level has a major affect
%in people getting a heart attack.
%As for the resting electrocaridogram results, people without a heart
%attack have mainly a normal electrocardiogram, while most people with a
%heart attack have an ST-T wave abnormality.
%Resting electrocardiogram results:
%Value 0: normal
%Value 1: having ST-T wave abnormality (T wave inversions and/or ST
%elevation or depression of > 0.05 mV)
%Value 2: showing probable or definite left ventricular hypertrophy by
```

```
%Estes' criteria

%Reference link for histogram:
https://uk.mathworks.com/help/matlab/ref/matlab.graphics.chart.primitive.histogram.html

%%
%EXPLORATORY DATA ANALYSIS - Thallium results
target_label_thal = categorical(df.('thal'));
figure;
histogram(target_label_thal(target_label_target=='1'),'BarWidth',0.6,'EdgeColor','none','FaceColor','blue');
hold on;
histogram(target_label_thal(target_label_target=='0'),'BarWidth',0.6,'EdgeColor','none','FaceColor','red');
hold off;
xlabel('Class Label');
ylabel('Total Observations');
title('Thallium test results (thal)');
legend('target=1','target=0');
xticks(categories(target_label_thal));
%These histograms compare the thallium results (heart tissue can't absorb
%thallium both under stress and in rest)for people with and
%without a heartattack. We want to see how these variables differ for
%people with and without a heartattack.

%Thallium results:
%Value 1: Normal
%Value 2: Fixed defect (heart tissue can't absorb thallium both under
%stress and in rest)
%Value 3: reversible defect (heart tissue is unable to absorb thallium only
%under the exercise portion of the test)

%For the people with a higher chance of a heart attack, the thallium
%results show that they have a fixed defect thallium (value 1) but for
%people with a lower risk of heart attack we can see many instances with
%reversible defect.
%%
missing_values = ismissing(df);
total_missing_values = sum(missing_values);
missing_values_table = array2table(total_missing_values, 'VariableNames',
df.Properties.VariableNames);
%We calculate the total missing values for each variable in our dataset.
%%

mins = min(df);
maxs = max(df);
medians = median(df);
means = mean(df);
stds = std(df);

summary_statistics_before_dp = table(mins,maxs,medians,means,stds);
%We calculate a summary statistics table before removing the outliers and
%normalizing certain variables in our dataset.
%%
% EXPLORATORY DATA ANALYSIS - Showing the distribution and skewness of the
% variables with the outliers
%('thalach' , 'chol' , 'trestbps' , 'age')
figure;
```

```
subplot(2,2,1);
histogram(df.('thalach'));
title('thalach with outliers');

subplot(2,2,2);
histogram(df.('chol'));
title('chol with outliers');

subplot(2,2,3);
histogram(df.('trestbps'));
title('trestbps with outliers');

subplot(2,2,4);
histogram(df.('age'));
title('age with outliers');
%%
%DATA PREPROCESSING - Normalizing Variables
df.('thalach') = (df.('thalach') - median(df.('thalach')))/
mad(df.('thalach'));
df.('chol') = (df.('chol') - median(df.('chol')))/ mad(df.('chol'));
df.('trestbps') = (df.('trestbps') - median(df.('trestbps')))/
mad(df.('trestbps'));
df = normalize(df,'norm',Inf,'DataVariables','age');
df.('age') = (df.('age') - min(df.('age'))) / (max(df.('age')) -
min(df.('age')));
df.('thalach') = (df.('thalach') - min(df.('thalach'))) /
(max(df.('thalach')) - min(df.('thalach')));
df.('chol') = (df.('chol') - min(df.('chol'))) / (max(df.('chol')) -
min(df.('chol')));
df.('trestbps') = (df.('trestbps') - min(df.('trestbps'))) /
(max(df.('trestbps')) - min(df.('trestbps')));
%For the 'thalach' , 'chol' and 'trestbps' variables, we normalize the data
%using the min-max scaler to bring the values into interval [0,1]. We can
%see from the boxplots in the exploratory analysis that there are some
%outliers. Normalization to the interval [0,1] is not robust to outliers.
%Outliers can have an affect to the scaling of the data so we will also try
%normaziling the dataset without the outliers to see the differences and
%compare.

%Reference link for normalizing the data:
https://moodle4.city.ac.uk/mod/folder/view.php?id=382059

%%
% EXPLORATORY DATA ANALYSIS - Showing the distribution and skewness of the
% normalized variables with the outliers
%('thalach' , 'chol' , 'trestbps' , 'age')
figure;
subplot(2,2,1);
histogram(df.('thalach'));
title('normalized thalach with outliers');

subplot(2,2,2);
histogram(df.('chol'));
title('normalized chol with outliers');

subplot(2,2,3);
histogram(df.('trestbps'));
title('normalized trestbps with outliers');
```

```
subplot(2,2,4);
histogram(df.('age'));
title('normalized age with outliers');

%We can see that all of the variables are either
%to the right or left except the 'age' variable. This is because as we can
%see from the boxplots, the 'age' column does not contain any outliers that
%affect the skewness of the distribution.

%%
%DATA PREPROCESSING - Removing the outliers from the dataset
df_without_outliers = rmoutliers(df,'mean');
%We remove the outliers from the dataset to see the differences in the
%distribution of the data and the skewness.
%To remove the outliers we used the rmoutliers() function, which defines an
%outlier as an element that is more than three standard deviations away
%from the mean value.
%Reference link for removing outliers:
https://uk.mathworks.com/help/matlab/ref/rmoutliers.html#d126e1420378

%%
%EXPLORATORY DATA ANALYSIS -  Distribution of the 'output' variable after
%removing the outliers
target_label_target = categorical(df_without_outliers.('target'));
figure;
histogram(target_label_target,'BarWidth',0.6,'EdgeColor','none','FaceColor'
,'blue');
xlabel('Class Labels');
ylabel('Total Observations');
title('Distribution of the class labels for the target variable without the
outliers');
%This histogram shows us the distribution of our target variable.
%We can see that the dataset is still pretty balanced even though we
%removed the outliers.

%%
%DATA PREPROCESSING - Normalizing variables after removing the outliers
df_without_outliers.('age') = (df_without_outliers.('age') -
min(df_without_outliers.('age'))) / (max(df_without_outliers.('age')) -
min(df_without_outliers.('age')));
df_without_outliers.('thalach') = (df_without_outliers.('thalach') -
min(df_without_outliers.('thalach'))) /
(max(df_without_outliers.('thalach')) -
min(df_without_outliers.('thalach')));
df_without_outliers.('chol') = (df_without_outliers.('chol') -
min(df_without_outliers.('chol'))) / (max(df_without_outliers.('chol')) -
min(df_without_outliers.('chol')));
df_without_outliers.('trestbps') = (df_without_outliers.('trestbps') -
min(df_without_outliers.('trestbps'))) /
(max(df_without_outliers.('trestbps')) -
min(df_without_outliers.('trestbps')));

%We normalize the data without the outliers using the min-max scaler method
%to see the differences in the distribution and skewness of the variables.
%Reference link for normalizing data:
https://moodle4.city.ac.uk/mod/folder/view.php?id=382059

%%
%EXPLORATORY DATA ANALYSIS - Showing the distribution and skewness of the
```

```
% normalized variables after removing the outliers
%('thalach' , 'chol' , 'trestbps' , 'age')
figure;
subplot(2,2,1);
histogram(df_without_outliers.('thalach'));
title('thalach normalized without outliers');

subplot(2,2,2);
histogram(df_without_outliers.('chol'));
title('chol normalized without outliers');

subplot(2,2,3);
histogram(df_without_outliers.('trestbps'));
title('trestbps normalized without outliers');

subplot(2,2,4);
histogram(df_without_outliers.('age'));
title('age normalized without outliers');
%We can see that the variables now follow a normal distribution after
%removing the outliers and applying normalization. So removing the outliers
%is beneficial to our analysis.

%%
%EXPLORATORY DATA ANALYSIS - Correlation Heatmap of the variables
%EXPLORATORY DATA ANALYSIS - Heatmap of the significance (p-value) of the
%without the outliers
selected_data = df_without_outliers(:,:);
selected_matrix = table2array(selected_data);
[correlation_coefficient , p_value] = corr(selected_matrix , 'Type',
'Spearman');
%We select the spearman correlation instead of the pearson because of the
%nature of our data. In the dataset we have both ordinal and interval data
%The independent variables and our dependent variable are not all the same
%type of data and in the same scale, so we cant assume that there is a
%linear relationship between them. So it is better to analyze the spearman
%correlation between the variables.
column_names = selected_data.Properties.VariableNames;

figure('Position', [100 100 800 600]);
heatmap(column_names,column_names,correlation_coefficient);
clim([-1,1]);
title('Correlation Heatmap of Variables without outliers');

figure('Position', [100 100 800 600]);
heatmap(column_names,column_names,p_value);
colors = [0 0 1; 1 0 0];
ax = gca; %This is used to set the properties in the current axes
colormap(ax,colors);
clim(ax,[0,0.1]); %We set the color scale limit
colorbar;
title('P-Value Heatmap of Variables without outliers');

%We create a correlation heatmap to check for the correlation between the
%variable we want to predict (dependent variable) and all of the
%independent variables. This will show us which variables affect more the
%values of our dependent variable.

%We also check for multicolienarity between our independent variables. We
%check to see if any independent variables are highly correlated so we can
```

```
%avoid to use both in our machine learning algorithms.

%We also created a heatmap with the significance testing of the variables.
%We visualize the p-values associated with the correlation coefficients to
% %see which correlations are statistically significant.

%Reference link for the corr() function:
https://uk.mathworks.com/help/stats/corr.html
%Reference link for heatmaps:
https://uk.mathworks.com/help/matlab/ref/heatmap.html
%Reference link for ax=gca function:
https://uk.mathworks.com/help/matlab/ref/gca.html
%Reference link for clim function:
https://uk.mathworks.com/help/matlab/ref/clim.html

%We can see that the correlations between the independent variables and
%between the independent variables and the target variable have not been
%affected by removing the outliers.

%After comparing the distribution of the data with and without the
%outliers, it's best for the outliers to be removed. There are four main
%reasons why i decided to remove the outliers:

%1. We want to eliminate the extreme values in the variables that may
%distort our model predictions.

%2. The distribution of the class labels for the target variable was not
%significantly affected by the removal of the outliers.

%3. The normalized variables without the outliers follow a more normal
%distribution whereas with the outliers are more skewed which can affect
%the model predictions.

%4. Only around 5% of the observations were removed after applying the
%outlier removal function. This is not considered a large amount of data.

%%
mins = min(df);
maxs = max(df);
medians = median(df);
means = mean(df);
stds = std(df);

summary_statistics = table(mins,maxs,medians,means,stds);
%We calculate a summary statistics table after performing all the necessary
%data preprocessing tasks.
%%
%CHOOSING THE INDEPENDENT VARIABLES THAT WILL OPTIMIZE OUR MODELS
Y = df_without_outliers{:,end}; %The dependent variable ('target')
X = df_without_outliers{:,{'cp','thalach', 'exang', 'ca','thal'}};
%We make sure the independent variables that we choose have a low,
statistically
%signficant correlation between them to avoid multicolienarity. We also
%want to choose the variables that are highly, statistically significant,
%correlated with our dependent variables. After looking at the correlation
%heatmap and the p-value heatmap and trying different combination, the
%independent variables that we feel optimize the predictions are the above.

%%
```

```
% SPLITTING THE DATASET TO TRAIN/TEST SET
rng('default'); %For reproducibility
cvp = cvpartition(size(df_without_outliers,1), 'Holdout' , 0.2);
%We partition the data using the holdout cross validation method with a
%80-20 (0.2) split for testing and training. This means that 80% of our
%data will be used to train our model and 20% will be used to test and
%validate our model predictions.
%Reference link for cvpartition function:
https://uk.mathworks.com/help/stats/cvpartition.html#d126e317540
X_train = X(training(cvp),:); %This is the X training set
Y_train = Y(training(cvp), :); %This is the Y training set
X_test = X(test(cvp), :); %This is the X test set
Y_test = Y(test(cvp), :); %This is the Y test set

combined_test_set = horzcat(X_test,Y_test); %We use the horzcat function so
%that the y_test is added at the end of the X_test
%Reference link for horzcat function:
https://uk.mathworks.com/help/matlab/ref/double.horzcat.html
headers = {'cp', 'thalach', 'exang', 'ca', 'thal', 'target'};
%We set the headers of the test set
combined_test_table = array2table(combined_test_set, 'VariableNames',
headers);
%We transform the array to a table.
writetable(combined_test_table, 'test_set.csv');
%We write the test set table to a csv file
%%
%MODEL PREDICTIONS - DECISION TREES (HOLDOUT CROSS VALIDATION) without
%hyperparameters
%We obtain the partitions of the data.
rng(1);
DTMdl = fitctree(X_train,Y_train);
%We fit a decision tree model to our training X data
%Reference link for  decision tree model (fitctree):
https://uk.mathworks.com/help/stats/fitctree.html
DTMdl_trainError = resubLoss(DTMdl);
%This metric is the total misclassification of our model predictions
predictedlabels_DT = predict(DTMdl , X_test);
%We predict the labels of our Y test set


%MODEL EVALUATIONS - DECISION TREES (HOLDOUT CROSS VALIDATION) without
%hyperparameters
%We will use the test set to evaluate our model based on the predictions it
%makes. We will use the X test to predict the Y labels. We want to see how
%similar the Y true values on our test set are similar to the model
%predictions.
confusion_matrix_DT = confusionmat(Y_test,predictedlabels_DT);
%This is a confusion matrix between the actual labels in the dependent
%variable and the labels that the decision tree model predicted. From a
%confusion matrix, we can calculate useful metrics that give an insight to
%how well the model actually predicted.

%Reference link for confusion matrix and chart:
https://uk.mathworks.com/help/stats/confusionmat.html?fbclid=IwAR1ybVZQjwP_
KYFCIfOg08QHPLhuYSxPE3Gm10q67UyOQosXLuboJJ_yFr0
TP_DT = confusion_matrix_DT(2,2);
 %True Positives: The model predicted correctly
%the positive label '1' (has the disease)
TN_DT = confusion_matrix_DT(1,1); %True Negatives: The model predicted
```

```
%correctly the negative label '0' (does not have the disease)
FP_DT = confusion_matrix_DT(1,2); %False Positives: The model predicted a
%positive label '1' when the actual label is negative '0'
FN_DT = confusion_matrix_DT(2,1); %False Negative: The model predicted a
%negative label '0' when the actual label is positive '1'

Accuracy_DT = (TP_DT+TN_DT) / (TP_DT+TN_DT+FP_DT+FN_DT);  %Accuracy is a
useful metric
%when the dataset is balanced.
Precision_DT = TP_DT/(TP_DT+FP_DT);
Recall_DT = TP_DT/(TP_DT+FN_DT);
F1_Score_DT = (2*TP_DT) / ((2*TP_DT)+FP_DT+FN_DT);
%Reference link for how to calculate the above metrics:
%https://moodle4.city.ac.uk/mod/folder/view.php?id=382059
[~,pp_score_DT] = predict(DTMdl,X_test); %The Predict function
%transforms the label predictions into the posterior probability scores.
pp_scores_positive_DT = pp_score_DT(:,2); %These are the posterior
%probability scores for the positive label '1' for each observation.
[X_Rate_DT,Y_Rate_DT,~,AUC_DT] = perfcurve(Y_test,pp_scores_positive_DT,1);
%In the above code we calculate the ROC graph and AUC using the perfcurve
%function. X_Rate is the False Positive and Y_Rate is the True Positive.

disp(AUC_DT);
%The close the AUC is to 1, the better.
%Reference link for ROC curve:
https://uk.mathworks.com/help/stats/perfcurve.html?fbclid=IwAR3yt-
8iUsEGtWlTPCUUjT3vRf3_W3hwLmSNB47gqQyN68yUCbKZ_61ifkU#bupy9b3-1
%In the section: Plot ROC Curve for Classification Tree.

%%
%MODEL PREDICTIONS - DECISION TREES (HOLDOUT CROSS VALIDATION) with
%hyperparameters
%We follow the exact same steps as before but now we add hyperparameters to
%our model to see how much our model improves. We set the
%'OptimizeHyperparameters' to 'auto' to find the best possible
%hyperparameter that will optimize our model predictions.
rng(1);%For reproducibility
DTMdl_HP = fitctree(X_train,Y_train,'OptimizeHyperparameters','auto');

%We save the final decision trees trained model with the best fitting
%hyperparameters to a .mat file
save('BestDTModel.mat','DTMdl_HP');
%We load the saved file
load('BestDTModel.mat');
%If we run our code, from the results we can see that the best
%hyperparameter to use in our decision tree model is MinLeafSize with an
%observed feasible point of 11.
optimalMinLeafSize = 11;
DTMdl_H_trainError_HP = resubLoss(DTMdl_HP);
predictedlabels_DT_HP = predict(DTMdl_HP , X_test);

%MODEL EVALUATIONS - DECISION TREES (HOLDOUT CROSS VALIDATION) with
%hyperparameters
confusion_matrix_DT_HP = confusionmat(Y_test,predictedlabels_DT_HP);

TP_DT_HP = confusion_matrix_DT_HP(2,2);
%True Positives: The model predicted correctly
%the positive label '1' (has the disease)
```

```
TN_DT_HP = confusion_matrix_DT_HP(1,1); %True Negatives: The model
predicted
%correctly the negative label '0' (does not have the disease)
FP_DT_HP = confusion_matrix_DT_HP(1,2); %False Positives: The model
predicted a
%positive label '1' when the actual label is negative '0'
FN_DT_HP = confusion_matrix_DT_HP(2,1); %False Negative: The model
predicted a
%negative label '0' when the actual label is positive '1'

Accuracy_DT_HP = (TP_DT_HP+TN_DT_HP) /
(TP_DT_HP+TN_DT_HP+FP_DT_HP+FN_DT_HP);  %Accuracy is a useful metric
%when the dataset is balanced.
Precision_DT_HP = TP_DT_HP/(TP_DT_HP+FP_DT_HP);
Recall_DT_HP = TP_DT_HP/(TP_DT_HP+FN_DT_HP);
F1_Score_DT_HP = (2*TP_DT_HP) / ((2*TP_DT_HP)+FP_DT_HP+FN_DT_HP);
%Reference link for how to calculate the above metrics:
%https://moodle4.city.ac.uk/mod/folder/view.php?id=382059
[~,pp_score_DT_HP] = predict(DTMdl_HP,X_test); %The Predict function
%transforms the label predictions into the posterior probability scores.
pp_scores_positive_DT_HP = pp_score_DT_HP(:,2); %These are the posterior
%probability scores for the positive label '1' for each observation.
[X_Rate_DT_HP,Y_Rate_DT_HP,~,AUC_DT_HP] =
perfcurve(Y_test,pp_scores_positive_DT_HP,1);
%In the above code we calculate the ROC graph and AUC using the perfcurve
%function. X_Rate is the False Positive and Y_Rate is the True Positive.

disp(AUC_DT_HP);
%The close the AUC is to 1, the better.
%Reference link for ROC curve:
https://uk.mathworks.com/help/stats/perfcurve.html?fbclid=IwAR3yt-
8iUsEGtWlTPCUUjT3vRf3_W3hwLmSNB47gqQyN68yUCbKZ_61ifkU#bupy9b3-1
%In the section: Plot ROC Curve for Classification Tree.
%Reference link for confusion matrix and chart:
https://uk.mathworks.com/help/stats/confusionmat.html?fbclid=IwAR1ybVZQjwP_
KYFCIfOg08QHPLhuYSxPE3Gm10q67UyOQosXLuboJJ_yFr0
%%
%MODEL PREDICTIONS - NAIVE BAYES (HOLDOUT CROSS VALIDATION) without any
%hyperparameters
rng(1);
NBMdl = fitcnb(X_train,Y_train);
%We fit a naive bayes model to our training X data
%Reference Link for Naive Bayes model (fitcnb):
https://uk.mathworks.com/help/stats/fitcnb.html
NBMdl_trainError = resubLoss(NBMdl);
%This metric is the total misclassification of our model predictions
predictedlabels_NB = predict(NBMdl , X_test);
%We predict the labels of our Y test set


%MODEL EVALUATIONS - NAIVE BAYES
NB_confusion_matrix = confusionmat(Y_test,predictedlabels_NB);
%This is a confusion matrix between the actual labels in the dependent
%variable and the labels that the decision tree model predicted. From a
%confusion matrix, we can calculate useful metrics that give an insight to
%how well the model actually predicted.
```

```
%Reference link for confusion matrix and chart:
https://uk.mathworks.com/help/stats/confusionmat.html?fbclid=IwAR1ybVZQjwP_
KYFCIfOg08QHPLhuYSxPE3Gm10q67UyOQosXLuboJJ_yFr0
TP_NB = NB_confusion_matrix(2,2);
 %True Positives: The model predicted correctly
%the positive label '1' (has the disease)
TN_NB = NB_confusion_matrix(1,1); %True Negatives: The model predicted
%correctly the negative label '0' (does not have the disease)
FP_NB = NB_confusion_matrix(1,2); %False Positives: The model predicted a
%positive label '1' when the actual label is negative '0'
FN_NB = NB_confusion_matrix(2,1); %False Negative: The model predicted a
%negative label '0' when the actual label is positive '1'

Accuracy_NB = (TP_NB+TN_NB) / (TP_NB+TN_NB+FP_NB+FN_NB);  %Accuracy is a
useful metric
%when the dataset is balanced.
Precision_NB = TP_NB/(TP_NB+FP_NB);
Recall_NB = TP_NB/(TP_NB+FN_NB);
F1_Score_NB = (2*TP_NB) / ((2*TP_NB)+FP_NB+FN_NB);
%Reference link for how to calculate the above metrics:
%https://moodle4.city.ac.uk/mod/folder/view.php?id=382059
[~,pp_score_NB] = predict(NBMdl,X_test); %The Predict function
%transforms the label predictions into the posterior probability scores.
pp_scores_positive_NB = pp_score_NB(:,2); %These are the posterior
%probability scores for the positive label '1' for each observation.
[X_Rate_NB,Y_Rate_NB,~,AUC_NB] = perfcurve(Y_test,pp_scores_positive_NB,1);
%In the above code we calculate the ROC graph and AUC using the perfcurve
%function. X_Rate is the False Positive and Y_Rate is the True Positive.

disp(AUC_NB);
%The close the AUC is to 1, the better.
%Reference link for ROC curve:
https://uk.mathworks.com/help/stats/perfcurve.html?fbclid=IwAR3yt-
8iUsEGtWlTPCUUjT3vRf3_W3hwLmSNB47gqQyN68yUCbKZ_61ifkU#bupy9b3-1
%In the section: Plot ROC Curve for Classification Tree.

%%
%MODEL PREDICTIONS - NAIVE BAYES (HOLDOUT CROSS VALIDATION) with
%hyperparameters
%We follow the exact same steps as before but we also add hyperparameters
%to our naive bayes model to see how much it improves
rng(1);
%Just like for decision trees, we use the 'OptimizeHyperparameters' and
%'auto' to find automatically the best possible hyperparameter that
%optimizes the performance of the model.
NBMdl_HP = fitcnb(X_train,Y_train,'OptimizeHyperparameters','auto');

%We save the trained naive bayes model with the best fitting
%hyperparameters to a .mat file
save('BestNBModel.mat','NBMdl_HP');
%We load the saved file
load('BestNBModel.mat');

NBMdl_H_trainError_HP = resubLoss(NBMdl_HP);
predictedlabels_NB_HP = predict(NBMdl_HP , X_test);

%MODEL EVALUATIONS - NAIVE BAYES (WITH HYPERPARAMETERS)
NB_confusion_matrix_HP = confusionmat(Y_test,predictedlabels_NB_HP);
TP_NB_HP = NB_confusion_matrix_HP(2,2);
```

```
TN_NB_HP = NB_confusion_matrix_HP(1,1);
FP_NB_HP = NB_confusion_matrix_HP(1,2);
FN_NB_HP = NB_confusion_matrix_HP(2,1);

Accuracy_NB_HP = (TP_NB_HP+TN_NB_HP) /
(TP_NB_HP+TN_NB_HP+FP_NB_HP+FN_NB_HP);
Precision_NB_HP = TP_NB_HP/(TP_NB_HP+FP_NB_HP);
Recall_NB_HP = TP_NB_HP/(TP_NB_HP+FN_NB_HP);
F1_Score_NB_HP = (2*TP_NB_HP) / ((2*TP_NB_HP)+FP_NB_HP+FN_NB_HP);
[~,pp_score_NB_HP] = predict(NBMdl_HP,X_test);
pp_scores_positive_NB_HP = pp_score_NB_HP(:,2);
[X_Rate_NB_HP,Y_Rate_NB_HP,~,AUC_NB_HP] =
perfcurve(Y_test,pp_scores_positive_NB_HP,1);


disp(AUC_NB_HP);

%%
%COMPARING AND EVALUATING THE MODELS OF DECISION TREES AND NAIVE BAYES WITH
%AND WITHOUT HYPERPARAMETERS (Holdout)
figure;
subplot(2,2,1);
confusion_chart_DT = confusionchart(Y_test,predictedlabels_DT);
title('DT without hyperparameters (Holdout)');
subplot(2,2,2);
confusion_chart_NB = confusionchart(Y_test,predictedlabels_NB);
title('NB without hyperparameters (Holdout)');
subplot(2,2,3);
confusion_chart_DT_HP = confusionchart(Y_test,predictedlabels_DT_HP);
title('DT with hyperparameters (Holdout)');
subplot(2,2,4);
confusion_chart_NB_HP = confusionchart(Y_test,predictedlabels_NB_HP);
title('NB with hyperparameters (Holdout)');
%has been transformed to a chart.

%%
%COMPARING AND EVALUATING THE MODELS OF DECISION TREES AND NAIVE BAYES
%(Holdout Cross Validation)
%We compare the metrics of decision trees and naive bayes
ml_algorithms = {'Decision Trees' , 'Naive Bayes'};
models_accuracy = [Accuracy_DT Accuracy_NB];
models_precision = [Precision_DT Precision_NB];
models_recall = [Recall_DT Recall_NB];
models_f1_score = [F1_Score_DT F1_Score_NB];
models_AUC_score = [AUC_DT AUC_NB];

models_accuracy_HP = [Accuracy_DT_HP Accuracy_NB_HP];
models_precision_HP = [Precision_DT_HP Precision_NB_HP];
models_recall_HP = [Recall_DT_HP Recall_NB_HP];
models_f1_score_HP = [F1_Score_DT_HP F1_Score_NB_HP];
models_AUC_score_HP = [AUC_DT_HP AUC_NB_HP];
figure;
barWidth = 0.3;

subplot(2,1,1);
barColors = ['r'; 'b'];  % Red for Naive Bayes, Blue for Decision Trees
```

```
bar(1:5, [models_accuracy(1), models_precision(1),
models_recall(1),models_f1_score(1),models_AUC_score(1)], barWidth,
'FaceColor', barColors(1,:), 'DisplayName', 'Decision Trees');
hold on;
bar(1.3:5.3, [models_accuracy(2),
models_precision(2),models_recall(2),models_f1_score(2),models_AUC_score(2)
], barWidth, 'FaceColor', barColors(2,:), 'DisplayName', 'Naive Bayes');
xlabel('Metrics');
ylabel('Values of Metrics');
title('Performance Metrics Comparison between DT and NB without
hyperparameters');
xticks(1.15:5.15);
xticklabels({'Accuracy', 'Precision','Recall','F1 Score', 'AUC Score'});
legend('Location', 'Best');
hold off;

subplot(2,1,2);
barColors = ['r'; 'b'];  % Red for Naive Bayes, Blue for Decision Trees

bar(1:5, [models_accuracy_HP(1), models_precision_HP(1),
models_recall_HP(1),models_f1_score_HP(1),models_AUC_score_HP(1)],
barWidth, 'FaceColor', barColors(1,:), 'DisplayName', 'Decision Trees');
hold on;
bar(1.3:5.3, [models_accuracy_HP(2),
models_precision_HP(2),models_recall_HP(2),models_f1_score_HP(2),models_AUC
_score_HP(2)], barWidth, 'FaceColor', barColors(2,:), 'DisplayName', 'Naive
Bayes');
xlabel('Metrics');
ylabel('Values of Metrics');
title('Performance Metrics Comparison between DT and NB with
hyperparameters');
xticks(1.15:5.15);
xticklabels({'Accuracy', 'Precision','Recall','F1 Score', 'AUC Score'});
legend('Location', 'Best');
hold off;

%%
%COMPARING AND EVALUATING THE MODELS OF DECISION TREES AND NAIVE BAYES
%We plot and compare the true positive and false positive rates (ROC) of
%decision trees and naive bayes (Holdout cross validation)
figure;
plot(X_Rate_DT,Y_Rate_DT,'b');
hold on;
plot(X_Rate_NB,Y_Rate_NB,'r');
plot(X_Rate_DT_HP,Y_Rate_DT_HP,'y');
plot(X_Rate_NB_HP,Y_Rate_NB_HP,'g');
xlabel('False positive rate')
ylabel('True positive rate')
title('ROC Curve for DT and NB (Holdout)');
legend('ROC for DT without hyperparameters','ROC for NB without
hyperparameters', 'ROC for DT with hyperparameters', 'ROC for NB with
hyperparameters');
%The AUC Score and ROC curve does not seem to change a lot for both the the
%decision trees and naive bayes models after adding hyperparameters. The
%ROC curve is closer to 1 in the decision trees model.

%Reference link for ROC curve:
https://uk.mathworks.com/help/stats/perfcurve.html?fbclid=IwAR3yt-
8iUsEGtWlTPCUUjT3vRf3_W3hwLmSNB47gqQyN68yUCbKZ_61ifkU#bupy9b3-1
```

```
%In the section: Plot ROC Curve for Classification Tree
```