

Assignment 2

Course: GNG5300 Machine Learning for Scientists and Engineers  
Professor: Olubisi Runsewe  
Date: 18/03/2022  
By: Tareq Dawoudiah, 7136770

Contents

[Part A: EDA 2](#_Toc98535398)

[Step 1 2](#_Toc98535399)

[Step 2 6](#_Toc98535400)

[Step 3 7](#_Toc98535401)

[Part B: Feature Engineering 8](#_Toc98535402)

[Step 1 8](#_Toc98535403)

[Step 2 8](#_Toc98535404)

[Step 3 8](#_Toc98535405)

[Step 4 8](#_Toc98535406)

[Part C: Model Development I 8](#_Toc98535407)

[Step 1 8](#_Toc98535408)

[Step 2 10](#_Toc98535409)

[Part D: Model Development II 12](#_Toc98535410)

[Step 1 12](#_Toc98535411)

[Part E: Model Comparison 13](#_Toc98535412)

[Step 1 13](#_Toc98535413)

[Step 2 15](#_Toc98535414)

# Part A: EDA

## Step 1

* The excel sheet is imported into a data frame. Lines 8-9.
* Histograms are plotted and printed. Lines 61-69.

|  |
| --- |
| ---    ---    ---    ---    --- |

## Step 2

* Getting a list with all the ages for positive and negative cases. Lines 75-79.
* Plotting the required histogram. Lines 81-85.

|  |
| --- |
|  |

* Getting list of male/female ages for positive cases. Lines 87-94.
* Plotting the required box plot. Lines 96-104.

|  |
| --- |
|  |

The median age for positive male cases is 57.0

The median age for positive female cases is 58.0

The median age for positive female cases is higher than that of males

## Step 3

* Changing all string column to numerical values. Lines 11-57.
* Creating a new data frame with all the numerical values. Lines 110-122.
* Plotting the required heat map. Lines 124-127.

|  |
| --- |
|  |

# Part B: Feature Engineering

## Step 1

* Checking for duplicate and missing data and then removing them. Lines 137-142.

## Step 2

* Removing negative old peak and cholesterol values (also negative for cholesterol). Lines 148-156.

## Step 3

* Converting categorical data to numeric values. Lines 11-57.

## Step 4

* Using a standard scaler to scale the data. Lines 169-173.

# Part C: Model Development I

## Step 1

* Creating the KNN model and getting the model performance results. Lines 226-240.

|  |
| --- |
| The 5 fold cross validation score for KNeighborsClf is : [0.85271318 0.85271318 0.8125 0.875 0.90625 ]  KNeighborsClf: 0.86 accuracy with a standard deviation of 0.03  KNeighborsClf: Classification report:  precision recall f1-score support  0 0.85 0.89 0.87 123  1 0.91 0.88 0.89 153  accuracy 0.88 276  macro avg 0.88 0.88 0.88 276  weighted avg 0.88 0.88 0.88 276  243 correct predictions out of 276 for KNeighborsClf  The KNeighborsClf percentage of the correct predictions is: 0.8804347826086957  Confusion matrix for KNeighborsClf:  [[109 14]  [ 19 134]] |

* Creating the SVM model and getting the model performance results. Lines 246-260.

|  |
| --- |
| The 5 fold cross validation score for SvmClf is : [0.84496124 0.8372093 0.859375 0.859375 0.8828125 ]  SvmClf: 0.86 accuracy with a standard deviation of 0.02  SvmClf: Classification report:  precision recall f1-score support  0 0.85 0.85 0.85 123  1 0.88 0.88 0.88 153  accuracy 0.86 276  macro avg 0.86 0.86 0.86 276  weighted avg 0.86 0.86 0.86 276  238 correct predictions out of 276 for SvmClf  The SvmClf percentage of the correct predictions is: 0.8623188405797102  Confusion matrix for SvmClf:  [[104 19]   * [ 19 134]] |

* Creating the DT model and getting the model performance results. Lines 266-289.

|  |
| --- |
| Optimal depth of the decision tree: {'max\_depth': 5}  The 5 fold cross validation score for DecisionTreeClassifierClf is : [0.84496124 0.82945736 0.8515625 0.8359375 0.8671875 ]  DecisionTreeClassifierClf: 0.85 accuracy with a standard deviation of 0.01  DecisionTreeClassifierClf: Classification report:  precision recall f1-score support  0 0.81 0.83 0.82 123  1 0.86 0.84 0.85 153  accuracy 0.84 276  macro avg 0.83 0.84 0.84 276  weighted avg 0.84 0.84 0.84 276  231 correct predictions out of 276 for DecisionTreeClassifierClf  The DecisionTreeClassifierClf percentage of the correct predictions is: 0.8369565217391305  Confusion matrix for DecisionTreeClassifierClf:  [[102 21]  [ 24 129]] |

* Creating the XGboot model and getting the model performance results. Lines 295-309.

|  |
| --- |
| The 5 fold cross validation score for GradientBoostingClf is : [0.80620155 0.88372093 0.8515625 0.890625 0.8828125 ]  GradientBoostingClf: 0.86 accuracy with a standard deviation of 0.03  GradientBoostingClf: Classification report:  precision recall f1-score support  0 0.84 0.85 0.84 123  1 0.88 0.87 0.87 153  accuracy 0.86 276  macro avg 0.86 0.86 0.86 276  weighted avg 0.86 0.86 0.86 276  237 correct predictions out of 276 for GradientBoostingClf  The GradientBoostingClf percentage of the correct predictions is: 0.8586956521739131  Confusion matrix for GradientBoostingClf:  [[104 19]  [ 20 133]] |

## Step 2

* Using the VotingClassifier to achieve the majority voting approach, first get the results for soft voting. Lines 315-317, 323-332.

|  |
| --- |
| The 5 fold cross validation score for MajorityVotingSoftClf is : [0.86821705 0.86821705 0.859375 0.875 0.890625 ]  MajorityVotingSoftClf: 0.87 accuracy with a standard deviation of 0.01  MajorityVotingSoftClf: Classification report:  precision recall f1-score support  0 0.83 0.85 0.84 123  1 0.88 0.86 0.87 153  accuracy 0.86 276  macro avg 0.86 0.86 0.86 276  weighted avg 0.86 0.86 0.86 276  237 correct predictions out of 276 for MajorityVotingSoftClfPredicted  The MajorityVotingSoftClfPredicted percentage of the correct predictions is: 0.8586956521739131  Confusion matrix for MajorityVotingSoftClf:  [[105 18]   * [ 21 132]] |

* Using the VotingClassifier to achieve the majority voting approach, then get the results for hard voting. Lines 319-321, 334-343.

|  |
| --- |
| The 5 fold cross validation score for MajorityVotingHardClf is : [0.85271318 0.86046512 0.8515625 0.890625 0.8828125 ]  MajorityVotingHardClf: 0.87 accuracy with a standard deviation of 0.02  MajorityVotingHardClf: Classification report:  precision recall f1-score support  0 0.83 0.88 0.85 123  1 0.90 0.86 0.88 153  accuracy 0.87 276  macro avg 0.86 0.87 0.87 276  weighted avg 0.87 0.87 0.87 276  239 correct predictions out of 276 for MajorityVotingHardClf  The MajorityVotingHardClf percentage of the correct predictions is: 0.8659420289855072  Confusion matrix for MajorityVotingHardClf:  [[108 15]  [ 22 131]] |

* We noticed that the hard voting method gave us slightly better results than the soft method.

# Part D: Model Development II

## Step 1

* Creating a deep neural network model using Keras with 3 dense layers. Two inner layers using ReLU, outer layer using Sigmoid. We then get the model performance results. Lines 356-360, 383-397.

|  |
| --- |
| Accuracy of kerasModelReLU: 93.46  kerasModelReLU: Classification report:  precision recall f1-score support  0 0.83 0.90 0.86 123  1 0.92 0.85 0.88 153  accuracy 0.87 276  macro avg 0.87 0.88 0.87 276  weighted avg 0.88 0.87 0.87 276  241 correct predictions out of 276 for kerasModelReLU  The kerasModelReLU percentage of the correct predictions is: 0.8731884057971014  Confusion matrix for kerasModelReLU:  [[111 12]  [ 23 130]] |

* By changing the activation function, we created two new models and assessed them:
  + Two inner layers use Tanh, outer layer using Sigmoid. Lines 362-366, 399-413.

|  |
| --- |
| Accuracy of kerasModelTanh: 93.46  kerasModelTanh: Classification report:  precision recall f1-score support  0 0.80 0.85 0.82 123  1 0.88 0.82 0.85 153  accuracy 0.84 276  macro avg 0.84 0.84 0.84 276  weighted avg 0.84 0.84 0.84 276  231 correct predictions out of 276 for kerasModelTanh  The kerasModelTanh percentage of the correct predictions is: 0.8369565217391305  Confusion matrix for kerasModelTanh:  [[105 18]  [ 27 126]] |

* + All three layers using Sigmoid. Lines 368-372, 415-428.

|  |
| --- |
| Accuracy of kerasModelSigmoid: 87.54  kerasModelSigmoid: Classification report:  precision recall f1-score support  0 0.85 0.85 0.85 123  1 0.88 0.88 0.88 153  accuracy 0.86 276  macro avg 0.86 0.86 0.86 276  weighted avg 0.86 0.86 0.86 276  238 correct predictions out of 276 for kerasModelSigmoid  The kerasModelSigmoid percentage of the correct predictions is: 0.8623188405797102  Confusion matrix for kerasModelSigmoid:  [[104 19]  [ 19 134]] |

* The change of layer activation functions did change our results; however, our first model still had the better results.

# Part E: Model Comparison

## Step 1

* Getting precision, recall, accuracy and F-measure for all algorithms and then saving them in lists for plotting. Lines 454-470.
* Plotting bar graphs to display how algorithms compare for precision, recall, accuracy and F-measure.

|  |
| --- |
| ---    --- |

## Step 2

* Printing the models that performed best and worst according to each criterion.

|  |
| --- |
| KNeighborsClf has the highest accuracy  DecisionTreeClassifierClf has the lowest accuracy  KNeighborsClf has the highest precision  DecisionTreeClassifierClf has the lowest precision  KNeighborsClf has the highest recall  DecisionTreeClassifierClf has the lowest recall  KNeighborsClf has the highest F1  DecisionTreeClassifierClf has the lowest F1 |

* Between our ensemble and deep neural network classifier, we saw better results for the neural network classifier, however, between all the models the k-nearest neighbors’ algorithm is the champion.