Machine Learning Model to Predict the Presence of Heart Disease

By

R.A.D. Dinushika Gunasekera

Machine Learning Foundation Course

Data Science Academy, Dialog Axiata PLC

Contents

1 Introduction

Heart disease or cardiovascular disease describes a range of conditions that affect the heart. These types of diseases are the leading cause of death globally. An estimated 17.9 million people died from heart diseases in 2019, representing 32% of all global deaths. Of these deaths, 85% were due to heart attack and stroke. Out of the 17 million premature deaths (under the age of 70) due to noncommunicable diseases in 2019, 38% were caused by heart diseases. Most cardiovascular diseases can be prevented by addressing behavioural risk factors such as tobacco use, unhealthy diet and obesity, physical inactivity and harmful use of alcohol. It is important to detect cardiovascular disease as early as possible so that management with counselling and medicines can begin. [Source: World Health Organization].

Hence this is a major concern to address. But it has been noticed that it is difficult to identify heart diseases as there are multiple factors contributing such as high cholesterol, high blood pressure, diabetes, abnormal pulse rate and many more. Due to such complexities, people can adopt modern technology approaches like Machine Learning to predict the presence of the disease. These can assist the healthcare professionals in decision making.

In this report, I have used Machine Learning to predict whether a person is suffering from heart disease or not.

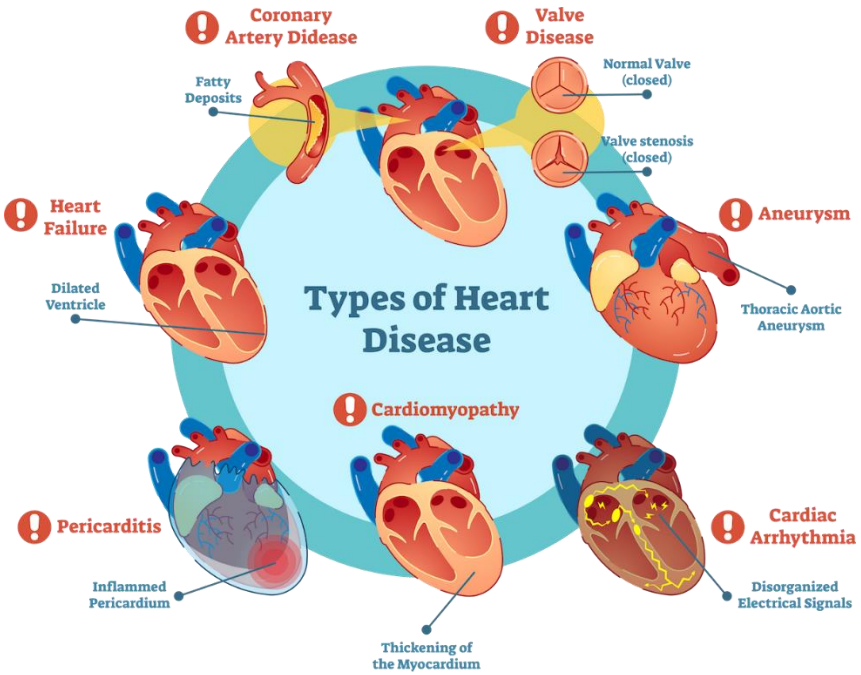


Figure 1: Types of Heart Diseases

2 Data

The Cleveland Heart Disease dataset was obtained from UCI Machine Learning data repository for this purpose.

Dataset: https://archive.ics.uci.edu/ml/datasets/Heart+Disease

Creators: Andras Janosi, M.D. (Hungarian Institute of Cardiology. Budapest), William Steinbrunn, M.D (University Hospital, Zurich, Switzerland), Matthias Pfisterer, M.D. (University Hospital, Basel, Switzerland), Robert Detrano, M.D., Ph.D. (V.A. Medical Center, Long Beach and Cleveland Clinic Foundation).

Data Set Characteristics:	Multivariate	**Number of Instances:**	302
Attribute Characteristics:	Categorical, Integer, Real	**Number of Attributes:**	75
Associated Tasks:	Classification	**Missing Values?**	Yes

Table 1: Dataset Information

The dataset consists of 302 individuals data. Even though it says there are 75 attributes, the processed.cleveland.data uses only 14 attributes. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0). Details of the 14 attributes are given below.

2.1 Attributes Information

1. ***'age'***: displays the age of the individual

2. ***'sex'***: displays the gender of the individual using the following format
   1 = male
   0 = female

3. ***'cp'***: displays the type of chest-pain experienced by the individual using the following format :
   1 = typical angina
   2 = atypical angina
   3 = non-anginal pain
   4 = asymptotic

4. ***'trestbps'***: displays the resting blood pressure value of an individual in mmHg (unit)

5. ***'chol'***: displays the serum cholesterol in mg/dl (unit)

6. ***'fbs'***: compares the fasting blood sugar value of an individual with 120mg/dl.
   If fasting blood sugar > 120mg/dl then : 1 (true)
   else : 0 (false)

7. ***'restecg'***: displays resting electrocardiographic results
   0 = normal
   1 = having ST-T wave abnormality
   2 = left ventricular hyperthrophy

8. ***'thalach'***: displays the max heart rate achieved by an individual

9. ***'exang'***: Exercise induced angina
   1 = yes
   0 = no

10. ***'oldpeak'***: ST depression induced by exercise relative to rest

11. ***'slope'***: The slope of the Peak exercise ST segment
    1 = upsloping
    2 = flat
    3 = downsloping

12. ***'ca'***: Number of major vessels (0–3) colored by flourosopy

13. ***'thal'***: displays the thalassemia
    3 = normal
    6 = fixed defect
    7 = reversible defect

14. ***'num'***: Diagnosis of heart disease. Displays whether the individual is suffering from heart disease or not:
    0 = absence
    1, 2, 3, 4 = present

2.2 Analyzing the Dataset

2.2.1 Data Types & Non-null Counts

To understand the data types and the non-null counts of the attributes: data.info()

Findings: No nulls. ID is integer. 'ca' & 'thal' are of object data type and remaining are float.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 302 entries, 0 to 301
Data columns (total 15 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   ID        302 non-null    int64
 1   age       302 non-null    float64
 2   sex       302 non-null    float64
 3   cp        302 non-null    float64
 4   trestbps  302 non-null    float64
 5   chol      302 non-null    float64
 6   fbs       302 non-null    float64
 7   restecg   302 non-null    float64
 8   thalach   302 non-null    float64
 9   exang     302 non-null    float64
 10  oldpeak   302 non-null    float64
 11  slope     302 non-null    float64
 12  ca        302 non-null    object
 13  thal      302 non-null    object
 14  num       302 non-null    int64
dtypes: float64(11), int64(2), object(2)
memory usage: 35.5+ KB
```

2.2.2 Identifying Missing Values

Missing values with NA: `data[data.isna().any(axis=1)]`

ID	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num	reslt

Findings: No missing values with NA

Missing values with '?': `data.isin(['?']).value_counts()`

```
ID     age    sex    cp     trestbps  chol   fbs    restecg  thalach  exang  oldpeak  slope  ca     thal   num    reslt
False  False  False  False  False     False  False  False    False    False  False    False  False  False  False  False    296
                                                                                        True   False  False  False      4
                                                                                        False  True   False  False      2
dtype: int64
```

Findings: 'ca' has 4 rows with '?' and 'thal' has 2 rows

Finding the exact row details – index numbers: `data[data.values == '?']`

	ID	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num	reslt
86	86	53.0	0.0	3.0	128.0	216.0	0.0	2.0	115.0	0.0	0.0	1.0	0.0	?	0	0
165	165	52.0	1.0	3.0	138.0	223.0	0.0	0.0	169.0	0.0	0.0	1.0	?	3.0	0	0
191	191	43.0	1.0	4.0	132.0	247.0	1.0	2.0	143.0	1.0	0.1	2.0	?	7.0	1	1
265	265	52.0	1.0	4.0	128.0	204.0	1.0	0.0	156.0	1.0	1.0	2.0	0.0	?	2	1
286	286	58.0	1.0	2.0	125.0	220.0	0.0	0.0	144.0	0.0	0.4	2.0	?	7.0	0	0
301	301	38.0	1.0	3.0	138.0	175.0	0.0	0.0	173.0	0.0	0.0	1.0	?	3.0	0	0

Findings: index 165, 191, 286, 301 contain '?' in 'ca' column and 86, 265 contain '?' in 'thal' column

2.2.3 Analyzing the Variable to be Predicted

- Analyzing the actual unique values of the data variable to be predicted, 'num':
  data['num'].unique()

```
array([2, 1, 0, 3, 4])
```

Findings: Five unique values as 0, 1, 2, 3, 4

- Introducing 'reslt' for easy referencing;
  num = 0; disease absent; num = 1, 2, 3, 4; disease present. Hence introducing reslt = 0 for disease absent and reslt = 1 for disease present: data['reslt'] = np.where(data['num']==0,0,1)
  Validating a data sample: data[['num', 'reslt']].sample(10)

	num	reslt
30	2	1
41	0	0
145	4	1
290	0	0
212	3	1
22	3	1
211	0	0
184	0	0
222	3	1
93	0	0

- To check whether balanced data counts are available: data['reslt'].value_counts()

```
0    163
1    139
Name: reslt, dtype: int64
```

Findings: '0' contains 54% and '1' contains 46% before pre-processing. This is balance dataset before the pre-processing.

2.2.4 Obtaining Histograms

To identify the value spread of each attribute, Histogram was used: data.hist(bins=50, figsize=(20, 20))

Findings: It was noticed that chol, fbs, oldpeak, ca are skewed

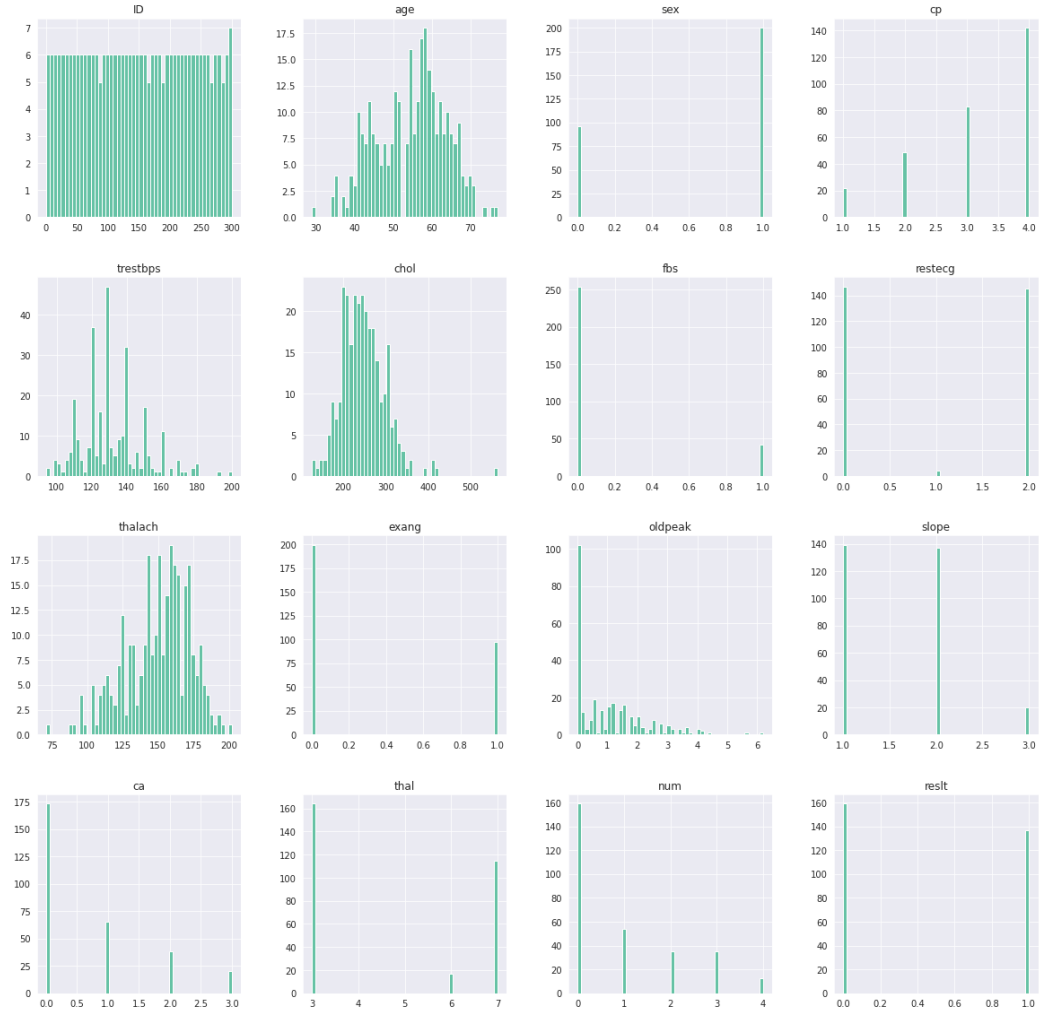


Figure 2: Histogram of the Attributes

2.2.5 Data Visualization

Data visualization was done using Seaborn.

- Gender vs Result

Figure 3: Gender vs Result

Findings: According to the dataset, males are more susceptible to heart diseases than females

- Chest Pain vs Result

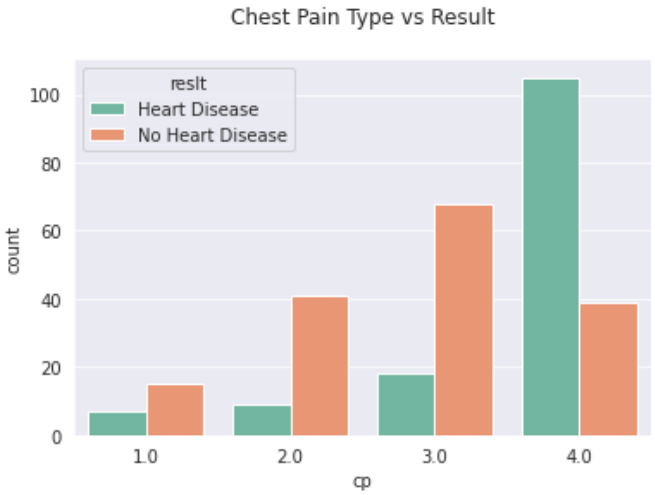


Figure 4: Chest Pain vs Result

Findings: From the four types of chest pains mentioned above, most of the heart disease patients are found to have asymptomatic chest pain.

- Age of Heart Diseased Patients

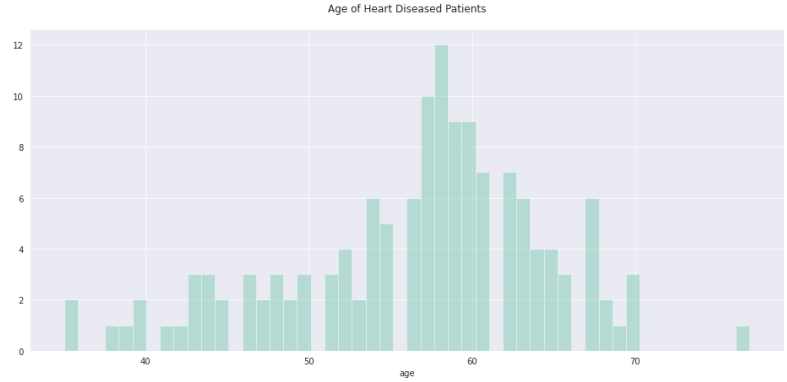


Figure 5: Age of Heart Diseased Patients

- Cholesterol Level of Heart Diseased Patients

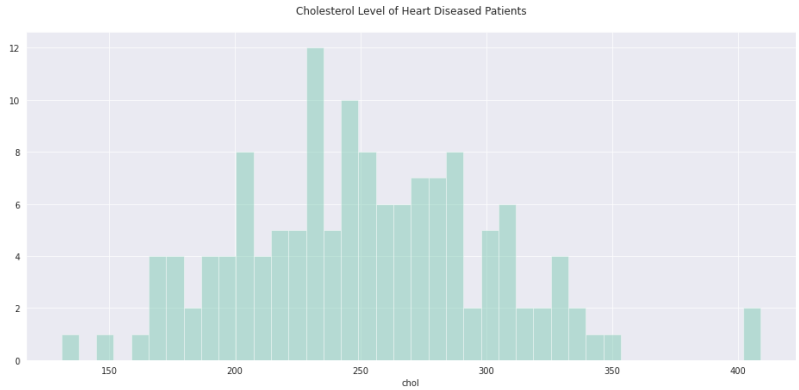


Figure 6: Cholesterol Level of Heart Diseased Patients

3 Methodology

3.1 ML Approach Selection

The problem is classification in nature where the objective is to classify whether a given patient is suffering from heart disease or not based on the input data. Leveraging the previous dataset available in UCI data repository, supervised learning algorithm is used for the model development.

3.2 Computing Environment

Google Colab cloud computing environment is used to build the ML model.

3.3 Libraries Used

Below are the libraries use for ML algorithm development

- NumPy – This library is used to perform mathematical operations on multidimensional arrays and matrix data structures
- Pandas – This library is used for data analysis
- Matplotlib – Comprehensive library for creating static, animated, and interactive visualizations in Python
- Seaborn – Library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures
- Scikit-learn – Library used for machine learning. Includes ML models and other necessary functions and tools

3.4 Data Pre-processing

3.4.1 Analyzing the Dataset

Dataset was analyzed to understand the dataset and to figure out their behaviours. Process in Chapter #2 Data 2.2.1 – 2.2.5 was followed to analyze the dataset.

3.4.2 Data Visualization

Data visualization was used for better understanding of the dataset and to derive subset of decisions based on the story telling of data. Process in Chapter #2 Data 2.2.5 was followed for data visualization.

3.4.3 Treating Missing Values

Missing data values in Chapter #2 Data 2.2.2 were treated by removing them from the dataset. This method was used since both the values of 'reslt' were nearly equal and quantity of the dataset was adequate to build the model: `data.drop(labels=[86, 165, 191, 265, 286, 301], axis=0, inplace=True)`

The 'reslt' count was checked after dropping the missing values rows: `data['reslt'].value_counts()`

```
0    159
1    137
Name: reslt, dtype: int64
```

3.4.4 Data Transformation

Dataset was analyzed to process categorical data to numerical data transformation. Dataset did not contain such data for transformation.

Next, 'ca' and 'thal' data groups were converted to 'object' to 'integer' due to the removal of '?' from dataset: data['ca'] = pd.to_numeric(data['ca']) data['thal'] = pd.to_numeric(data['thal'])

```
ID            int64
age         float64
sex         float64
cp          float64
trestbps    float64
chol        float64
fbs         float64
restecg     float64
thalach     float64
exang       float64
oldpeak     float64
slope       float64
ca          float64
thal        float64
num           int64
reslt         int64
dtype: object
```

3.4.5 Finding Outliers

Next outliers were found in the dataset using boxplot method.

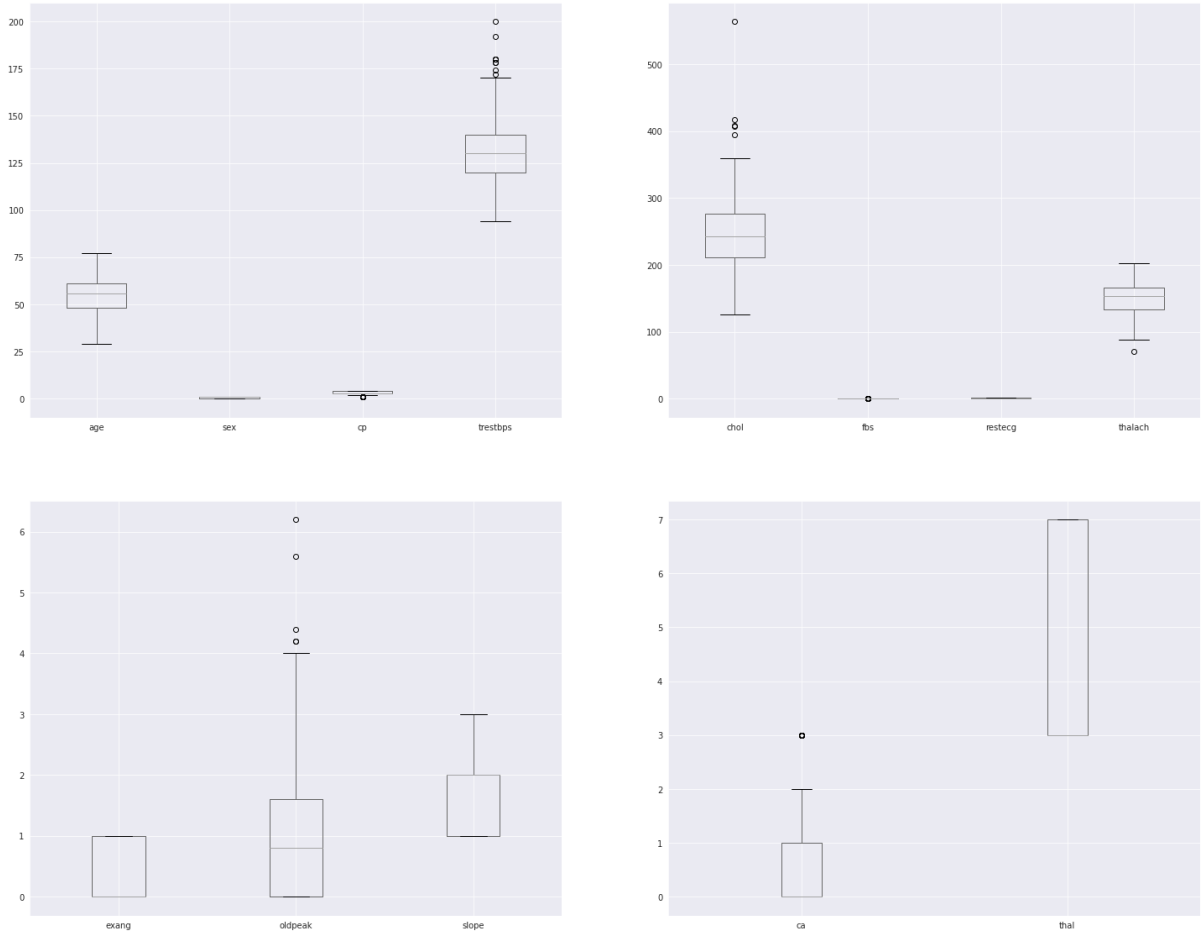


Figure 7: Boxplot for Outliers

Findings: Outliers are available in cp, trestbps, chol, fbs, thalach, oldpeak and ca.

Next, the exact rows of outliers were found using below for each variable above;

```
# 1st quartile
q1 = np.quantile(data['cp'], 0.25)

# 3rd quartile
q3 = np.quantile(data['cp'], 0.75)

iqr = q3-q1

# upper and lower whiskers
upper_bound = q3+(1.5*iqr)
lower_bound = q1-(1.5*iqr)

outliers = data['cp'][(data['cp'] < lower_bound) | (data['cp']> upper_bound)]
```

3.4.6 Treating Outliers

It was identified that there are total of 59 rows with outliers in all the variables. Since the dataset was adequate for the ML model building, it was decided to treat the outliers by removing the rows.

The 'reslt' count was checked after dropping the outliers

```
0    136
1    101
Name: reslt, dtype: int64
```

Findings: Value count 0 is ~ 57% of the total value and value count 1 is ~ 43%. Hence this can be considered as a nearly balanced dataset.

Above finding eliminates the requirement of undersampling or oversampling to even out the dataset.

3.5 Feature Selection

Next the correlation was checked between the attributes in order to select the X variables and y variables.

3.5.1 Using Correlation Matrix

	ID	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	num	reslt
ID	1.000000	0.024505	-0.141722	-0.022154	-0.037225	-0.126234	-0.019977	-0.137862	-0.125767	0.013159	-0.159858	-0.037093	0.041617	-0.059394	-0.015907	-0.013738
age	0.024505	1.000000	-0.066848	0.167488	0.248770	0.142795	0.095498	0.122351	-0.403764	0.124884	0.197961	0.116631	0.397739	0.170413	0.219499	0.225197
sex	-0.141722	-0.066848	1.000000	0.138318	-0.019589	-0.134393	0.056344	0.018014	-0.067441	0.178988	0.183390	0.054296	0.089082	0.426454	0.293363	0.319765
cp	-0.022154	0.167488	0.138318	1.000000	0.048353	0.005085	-0.066027	0.144170	-0.368598	0.440916	0.337110	0.241462	0.262179	0.387359	0.442199	0.479012
trestbps	-0.037225	0.248770	-0.019589	0.048353	1.000000	0.100424	0.072665	0.136285	0.003743	0.025440	0.174139	0.037248	0.026240	0.092861	0.108419	0.106982
chol	-0.126234	0.142795	-0.134393	0.005085	0.100424	1.000000	-0.049124	0.138371	0.006748	0.062782	-0.009462	-0.070484	0.117486	-0.017437	0.027338	0.077700
fbs	-0.019977	0.095498	0.056344	-0.066027	0.072665	-0.049124	1.000000	0.036200	0.000505	0.038178	0.027997	0.045610	0.125958	0.040339	0.027984	0.013258
restecg	-0.137862	0.122351	0.018014	0.144170	0.136285	0.138371	0.036200	1.000000	-0.109260	0.125631	0.134770	0.149558	0.092997	0.006617	0.210689	0.192260
thalach	-0.125767	-0.403764	-0.067441	-0.368598	0.003743	0.006748	0.000505	-0.109260	1.000000	-0.453545	-0.381029	-0.391297	-0.245391	-0.364128	-0.441414	-0.412157
exang	0.013159	0.124884	0.178988	0.440916	0.025440	0.062782	0.038178	0.125631	-0.453545	1.000000	0.347751	0.313952	0.211836	0.374746	0.417204	0.449927
oldpeak	-0.159858	0.197961	0.183390	0.337110	0.174139	-0.009462	0.027997	0.134770	-0.381029	0.347751	1.000000	0.559423	0.284831	0.337250	0.513239	0.464561
slope	-0.037093	0.116631	0.054296	0.241462	0.037248	-0.070484	0.045610	0.149558	-0.391297	0.313952	0.559423	1.000000	0.092590	0.242195	0.370622	0.341223
ca	0.041617	0.397739	0.089082	0.262179	0.026240	0.117486	0.125958	0.092997	-0.245391	0.211836	0.284831	0.092590	1.000000	0.284466	0.508390	0.474808
thal	-0.059394	0.170413	0.426454	0.387359	0.092861	-0.017437	0.040339	0.006617	-0.364128	0.374746	0.337250	0.242195	0.284466	1.000000	0.551249	0.594139
num	-0.015907	0.219499	0.293363	0.442199	0.108419	0.027338	0.027984	0.210689	-0.441414	0.417204	0.513239	0.370622	0.508390	0.551249	1.000000	0.836159
reslt	-0.013738	0.225197	0.319765	0.479012	0.106982	0.077700	0.013258	0.192260	-0.412157	0.449927	0.464561	0.341223	0.474808	0.594139	0.836159	1.000000

Figure 8: Correlation Matrix

Findings: It was identified that 'thalach' is negatively correlated to 'reslt' and rest of the variables are positively correlated. Out of all the features, 'thal', 'ca', 'oldpeak', 'exang', 'thalach, 'cp' are highly correlated to 'reslt'.

3.5.2 Using Heatmap

Heatmap was generated using seaborn: sns.heatmap(correlation_matrix, annot=True)

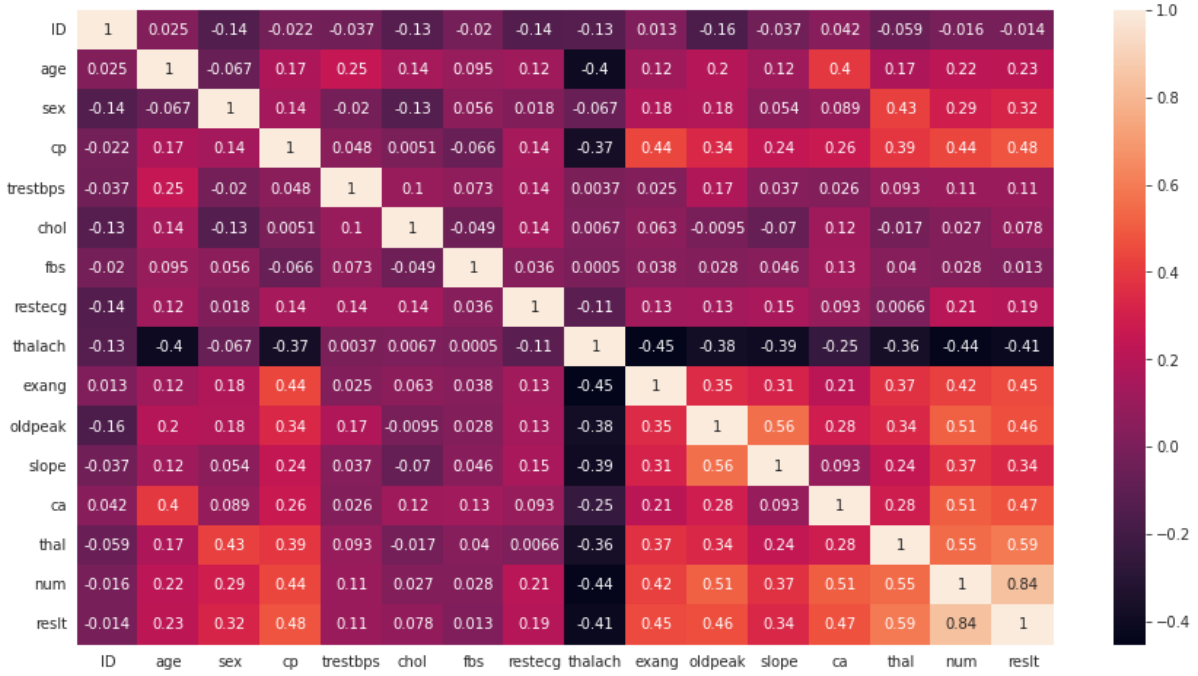
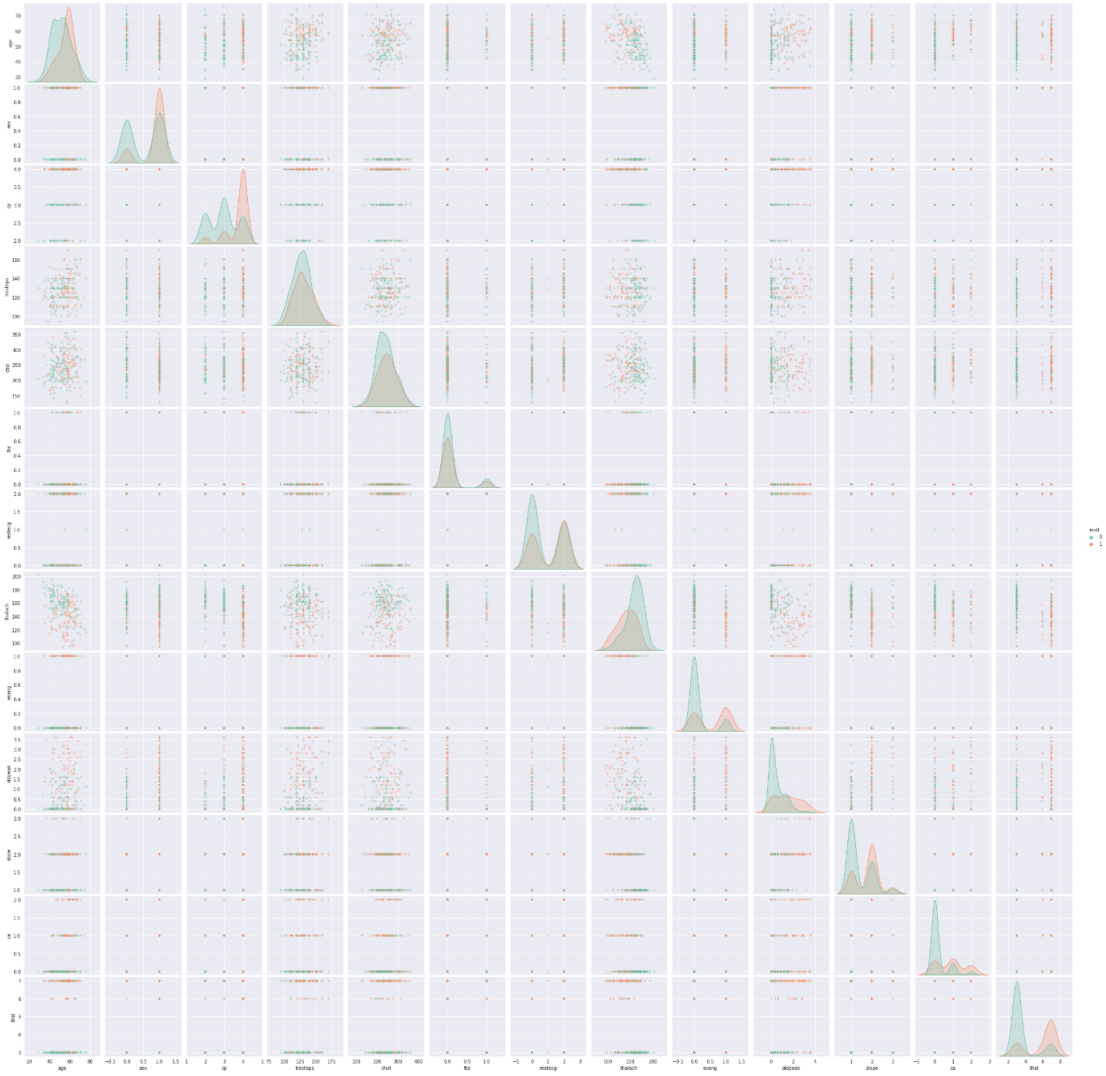


Figure 9: Heatmap

3.5.3 Using Pairplot

Pairplots were generated and relationships were analyzed between each pairs of variables against 'reslt' 0 and 1.



Based on the findings above below only 'thal' has correlation above 0.5. Further in the original dataset there had been 76 attributes from which above 14 attributes had been selected based on the correlation. Hence below features were selected for ML model.

- X_variables = ['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'ca', 'thal']
- y_variable = 'reslt'

3.6 Test Train Split

The dataset was split into 02 sets, one for training and one for testing. Training dataset contains 70% of the total dataset and remaining 30% is allocated for testing: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

3.7 Model Building

As there is a plan to test several classification models and compare the evaluation results, for easy usage purpose, a function was created to train the models.

- Input to the function: model, model_name, X_train, y_train, X_test, y_test. To train the data in the model, model.fit(X_train, y_train) was used
- Result ('reslt') was predicted using y_pred = model.predict(X_test) where 'y_pred' is the predicted value of 'reslt' using X_train dataset
- To evaluation and scoring purposes, accuracy, precision, recall, f1 score and the area under the ROC curve were used from Scikitlearn metrics scoring functions
- Output of the function: model name, model details from scikit learn, accuracy, precision, recall, f1 score, area under the roc curve, actual test dataset & predicted data set

3.8 Model Evaluation

A list was created called 'models' and model training function was called for each classification model in the 'model' list.

	model_name	model	accuracy	precision	recall	f1_score	roc_auc	y act	y pred
0	LGR	LogisticRegression(n_jobs=3, random_state=0)	0.902778	0.914286	0.902778	0.902759	0.925154	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, ...	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...
1	KNC	KNeighborsClassifier()	0.652778	0.677419	0.652778	0.651095	0.743441	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...	[0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, ...
2	RF	(DecisionTreeClassifier(criterion='entropy', m...	0.819444	0.870968	0.819444	0.818569	0.926312	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, ...
3	DT-G	DecisionTreeClassifier(random_state=0)	0.805556	0.805556	0.805556	0.805556	0.805556	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...	[1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...
4	DT-En	DecisionTreeClassifier(criterion='entropy', ra...	0.805556	0.823529	0.805556	0.805405	0.805556	[0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...	[1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, ...

3.9 Parameter Tuning

For the reference purposes highest scored model (Logistic Regression) from the above list was chosen and grid search was done for the hyper parameter tuning.

```
Tuned Model Parameters: {'C': 0.5}
Best model score: 0.8003589743589743
```

```
Confusion Matrix:
 y_pred   0   1  All
y_act
0       32   4   36
1        4  32   36
All     36  36   72


accuracy_lgr = 0.8888888888888888
Precision_lgr = 0.8888888888888888
Recall_lgr = [0.88888889 0.88888889]
F1 score_lgr = [0.88888889 0.88888889]
classification_report:
              precision    recall  f1-score   support

           0       0.89      0.89      0.89        36
           1       0.89      0.89      0.89        36

    accuracy                           0.89        72
   macro avg       0.89      0.89      0.89        72
weighted avg       0.89      0.89      0.89        72
```

3.10 Saving the Model

Pickle was use to store the best scoring model we received.

```
import pickle

save_file = 'model_LGR.pickle'
pickle.dump(model, open(save_file, 'wb'))
```

4 Results

Result of the above model evaluation is given below.

	model_name	model	accuracy	precision	recall	f1_score	roc_auc
0	LGR	LogisticRegression(n_jobs=3, random_state=0)	0.902778	0.914286	0.902778	0.902759	0.925154
1	KNC	KNeighborsClassifier()	0.652778	0.677419	0.652778	0.651095	0.743441
2	RF	(DecisionTreeClassifier(criterion='entropy', m...	0.819444	0.870968	0.819444	0.818569	0.926312
3	DT-G	DecisionTreeClassifier(random_state=0)	0.805556	0.805556	0.805556	0.805556	0.805556
4	DT-En	DecisionTreeClassifier(criterion='entropy', ra...	0.805556	0.823529	0.805556	0.805405	0.805556

Each model was compared against accuracy, precision, recall, f1_score and roc_auc.

5 Conclusion

- More focus was given to below matrices for model evaluation;
  - roc_auc: area under the roc curve is the measure of the ability of a classifier to distinguish between classes. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes hence better the model performance
  - f1_score: since it is the weighted average of Precision and Recall which take both false positives and false negatives into account.

$$f1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

- Considering above matrices performance Logistic Regression was selected as the best model for heart disease prediction as both f1 score and roc_auc are over 0.9 and higher in value compared to other models. Further, the rest of the evaluation matrices (accuracy, precision and recall) are too similar to above

6 Discussion

- Due to the missing value and outlier treatments, around 65 rows of data had to be removed which is around ~ 22% of dataset. A larger dataset would have contributed for further fine-tunning of the model
- Available dataset as 'Cleveland data' in the UCI repository is already a processed set of data. Out of 76 attributes, they have already chosen 14 attributes which are high in correlation. Hence variable removal was not required based on the correlation matric performance.