

FINAL PROJECT REPORT

BIG DATA ANALYTICS RESEARCH PROJECT

AIT-INFS-580 ANALYTICS – BID DATA TO INFORMATION

FACULTY: AISHA SIKDER

SUBMITTED BY: AISWARYA SEN

GNUMBER: G01388802

TABLE OF CONTENTS

1. INTRODUCTION	6
1.1 DATASET SELECTED	6
1.2 RESEARCH QUESTIONS RELATED TO THE DATASET	7
1.3 THE POTENTIAL BENEFITS OF ANSWERING THE RESEARCH QUESTIONS	7
2. RELATED WORKS	8
2.1 RESEARCH PAPER: 1	8
2.2 RESEARCH PAPER: 2	10
2.3 RESEARCH PAPER: 3	12
3. DATASET ANALYSIS.....	14
3.1 IMPORT DATASET.....	14
3.2 DATA PREPROCESSING AND DATA CLEANING	16
3.2.1 CHECKING FOR NULL AND DATA UNIQUE VALUES IN ALL COLUMNS	16
3.2.2 DROPPING ROWS WITH NULL VALUES.....	19
3.2.3 CHECKING THE DATA TYPES.....	20
3.2.4 CHECKING DUPLICATES.....	21
3.2.5 GENERAL STATISTICS SUMMARY OF THE DATASET AND INTERPRETATION	21
3.2.6 TARGET ATTRIBUTE.....	22
3.3 NOIR ANALYSIS.....	22
3.4 DATA VISUALIZATION	24
3.4.1 TARGET DISTRIBUTION IN THE DATASET	24
3.4.2 GENDER ANALYSIS IN THE DATASET	25
3.4.3 AGE ANALYSIS IN THE DATASET	26
3.4.4 COORELATIN BASED FEATURE SELECTION IN THE DATASET	27
3.4.5 ANALYSIS BASED ON THE PHYSICAL ACTIVITY INDUCED ON THE PAITENTS.....	29
3.4.6 CHEST PAIN AND FASTING BLOOD SUGAR ANALYSIS.....	31
3.4.7 DISTRIBUTION AND HISTOGRAM PLOTS.....	33
3.4.8 COMPARING TWO ATTRIBUTES WITH RESPECT TO TARGET	35
3.4.9 RELATIONSHIP PLOT	37
3.5.0 SCATTER PLOT AND PAIR PLOT.....	38
3.5 PREPARING DATA FOR MODEL.....	40
3.5.1 DATA SCALING	40

3.5.2	DATA SPLIT AND TRAINING	40
3.5.3	LOGISTIC REGRESSION	41
3.5.4	K-NEAREST NEIGHBORS ALGORITHM.....	43
3.5.5	CLUSTERING	44
3.6	NLP ANALYSIS	45
4.	ANALYSIS OF RESULT	46
5.	CONCLUSION.....	47
5.1	SUMMARIZE FINDINGS OF ANALYSIS	47
5.2	LIMITATIONS	47
5.3	FUTURE SCOPE	47
5.4	CODE SNIPPETS.....	48
6.	REFERENCES	62

LIST OF TABLES

TABLE 1. 1 NOIR ANALYSIS	23
--------------------------------	----

LIST OF FIGURES

FIGURE 1. 1 IMPORT LIBRARIES	14
FIGURE 1. 2 IMPORT THE DATASET	14
FIGURE 1. 3 CREATE DATAFRAME	14
FIGURE 1. 4 DATAFRAME WITH HEADER LIST	15
FIGURE 1. 5 SHAPE OF THE DATASET	15
FIGURE 1. 6 DATA INFO	15
FIGURE 1. 7: CHECK FOR NULL VALUES	16
FIGURE 1. 8 UNIQUE VALUES IN THE DATASET	16
FIGURE 1. 9 UNIQUE VALUES : CA AND THAL	17
FIGURE 1. 10 SPEACIAL CHARACTER CONVERSION	17
FIGURE 1. 11 CHECK FOR NULL VALUES-2	18
FIGURE 1. 12 VISUALIZING THE MISSING VALUES	18
FIGURE 1. 13: DATAFRAME AFTER DROPPING THE NULL VALUES	19
FIGURE 1. 14 DATA SHAPE AFTER REMOVING NULL & MISSING VALUES	19
FIGURE 1. 15 DATATYPES	20
FIGURE 1. 16 DATATYPE AFTER CONVERSION	20
FIGURE 1. 17 CHECK FOR DUPLICATES	21
FIGURE 1. 18 TARGET ATTRIBUTE UPDATED	22
FIGURE 1. 19 ATTRIBUTE VALUE UPDATION	23
FIGURE 1. 20 GETTING COUNT VALUES	24
FIGURE 1. 21 TARGET DISTRIBUTION ANALYSIS	24
FIGURE 1. 22 GENDER ANALYSIS	25
FIGURE 1. 23 AGE DISTRIBUTION IN THE RANGE OF 10	26
FIGURE 1. 24 AGE ANALYSIS	27
FIGURE 1. 25 CORRELATION MATRIX	28
FIGURE 1. 26 EXERCISE INDUCED ANGINA ANALYSIS ON TARGET	29
FIGURE 1. 27 OLD PEAK ANALYSIS	30
FIGURE 1. 28 SLOPE ANALYSIS	30
FIGURE 1. 29 DATASET ANALYSIS : CHEST PAIN ANALYSIS	31
FIGURE 1. 30 FASTING BLOOD SUGAR ANALYSIS	32
FIGURE 1. 31 DISTRIBUTION PLOTS	33
FIGURE 1. 32 RESTING ECG ANALYSIS	34
FIGURE 1. 33 THALLIUM STRESS ANALYSIS	34
FIGURE 1. 34 CA ANALYSIS	35
FIGURE 1. 35 COMPARSION WITH TWO ATTRIBUTES AGAINST TARGET	36
FIGURE 1. 36 RELATIONSHIP PLOT	37
FIGURE 1. 37 SCATTER PLOT	38
FIGURE 1. 38 SNS PAIR PLOT	39

FIGURE 1. 39 DATA SCALING	40
FIGURE 1. 40 DATA SPLIT AND TRAIN	41
FIGURE 1. 41 CONFUSION MATRIX	42
FIGURE 1. 42 ACCURACY TEST	42
FIGURE 1. 43 CLASSIFICATION REPORT	43
FIGURE 1. 44 KNN	43
FIGURE 1. 45 KMEANS CLUSTERING	44
FIGURE 1. 46 KMEANS CLUSTERING	45
FIGURE 1. 47 IMPORTING LIBRARIES : NLP ANALYSIS	45
FIGURE 1. 48 NLP ANALYSIS : FREQUENCY COUNT	46

1. INTRODUCTION

Over the years, a lot of essential patient and healthcare data have been gathered globally. I am interested in the medical field and health science domain because of which I have opted for this dataset. Machine learning algorithms have lot of scope in this domain to be able to analyze and answer health care related questions. Detecting patterns using these datasets would give an upper hand to the health care practices to better provide services to the world, early detection of diseases and common health symptoms, aid in the diagnosis of illness based on already available data and so on. This aspect excites me as a software engineer where I can contribute to the betterment of health care by applying machine learning paradigms.

The heart is an extremely complex organ with a challenging role to play in keeping us alive. Heart disease accounts for nearly 630,000 deaths in the United States of America every year. It is so frequent that every 40 seconds at least one person will have heart attack and every minute at least one person dies out of heart failure conditions. A prompt diagnosis, prompt therapy, and ongoing monitoring are necessary for patients with heart disease. In the past, a variety of data mining techniques have been employed in the diagnosis and forecasting of heart illnesses to meet their needs. This dataset contains some medical information of patients which helps in determine the presence of heart disease in a person which could help an individual in getting medical diagnosis at the early stage.

1.1 DATASET SELECTED

❖ Heart Disease Databases

This file describes the contents of the heart-disease directory. The directory contains 4 databases concerning heart disease diagnosis. All attributes are numeric-valued. This database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. Experiments with the Cleveland database have concentrated on simply attempting to distinguish presence (values 1,2,3,4) from absence (value 0). The names and social security numbers of the patients were recently removed from the database, replaced with dummy values.

The "goal" field refers to the presence of heart disease in the patient. It is integer valued from 0 (no presence) to 4.

14 main attributes of the dataset used:

1. **age:** Age in years
2. **sex:** 1 = male; 0 = female
3. **cp:** Chest pain type
 - 1: Typical angina: chest pain related decrease blood supply to the heart
 - 2: Atypical angina: chest pain not related to heart
 - 3: Non-angina pain: typically, esophageal spasms (non-heart related)
 - 4: Asymptomatic: chest pain not showing signs of disease

4. **trestbps**: Resting blood pressure (in mm Hg on admission to the hospital)
anything above 130-140 is typically cause for concern
5. **chol**: serum cholesterol in mg/dl
6. **fbs**: Fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
7. **restecg**: Resting electrocardiographic result
Value 0: normal
Value 1: having ST-T wave
Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
8. **thalach**: Maximum heart rate achieved
9. **exang**: Exercise induced angina (1 = yes; 0 = no)
10. **old peak**: ST depression induced by exercise relative to rest
11. **slope**: The slope of the peak exercise ST segment
1: Up sloping: better heart rate with exercise (uncommon)
2: Flat sloping: minimal change (typical healthy heart)
3: Down sloping: signs of unhealthy heart
12. **ca**: number of major vessels (0-3) colored by fluoroscopy
13. **thal**: Thallium stress
3: normal
6: fixed defect
7: reversible defect
14. **target – num**: Diagnosis of heart disease (angiographic disease status)
Value 0: < 50% diameter narrowing
Value 1: > 50% diameter narrowing

1.2 RESEARCH QUESTIONS RELATED TO THE DATASET

1. How has gender and age affected the patterns of heart disease occurrence in patient for the past few years?
2. What are the most effective factors in determining the presence of heart diseases in a patient?
3. How much physical activity needs to be stimulated in the patients to detect the heart disease in patients?

1.3 THE POTENTIAL BENEFITS OF ANSWERING THE RESEARCH QUESTIONS

- This would help health care providers to formulate the probability of heart disease occurrence for a range of group and sex and thereby a routine checkup done for patients in this age group and sex could provide early detection of symptoms and thereby prevention.

- By identifying the most effective factors that contribute to the presence of heart disease will help health care providers to concentrate on those factors during the checkup and add these as part of quick diagnosis tests. This would also help categorize symptoms related to heart disease and other illnesses.
- This would help in formulating the detection procedure for heart disease for a specific age and sex. For different age group and sex, the level of exercise stimulus could be different and using the pattern we see in the dataset, the upper and lower bound of these inductions can be set accordingly which would help with early and accurate detections.

2. RELATED WORKS

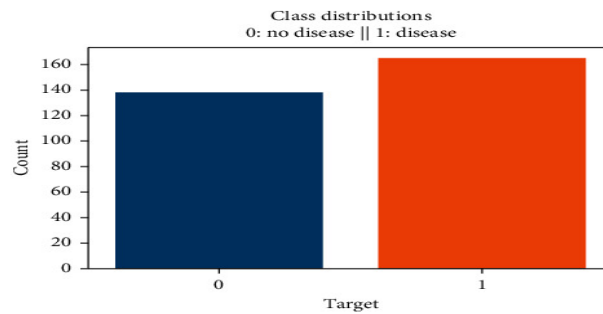
2.1 RESEARCH PAPER: 1

PREDICTION OF HEART DISEASE USING A COMBINATION OF MACHINE LEARNING AND DEEP LEARNING

Introduction:

In this paper different machine learning and deep learning technologies are used to compare the results and analysis of the UCI machine learning heart disease dataset. The dataset uses 14 main attributes to perform the analysis report. The correct prediction of heart disease can prevent life threats, and incorrect prediction can prove to be fatal at the same time. As time is passing, a lot of research data and patients records of hospitals are available. There are many open sources for accessing the patient's records and researches can be conducted so that various computer technologies could be used for doing the correct diagnosis of the patients and detect this disease to stop it from becoming fatal. Machine learning and Artificial Intelligence are playing a critical role in the medical field. A complete genomic data analysis can easily be done using machine learning models.

The classification is common in every study to predict if a patient has heart disease or not, and also one most common pattern which can be seen is that the dataset commonly used is of Cleveland. The results obtained achieved great accuracies like random forest with 89.2 percent accuracy; decision tree with 89.1 percent accuracy; ANN with 92.7 percent accuracy, 89 percent, and 89.7 percent accuracy; and SVM accuracy with 88 percent. Heart disease is very fatal and it should not be taken lightly. Heart disease happens more in males than females, which can be read further from Harvard Health Publishing. Researchers found that, throughout life, men were about twice as likely as women to have a heart attack. That higher risk persisted even after they accounted for traditional risk factors of heart disease, including high cholesterol, high blood pressure, diabetes, body mass index, and physical activity. For the preprocessing of data, two approaches were used. One without outliers and feature selection process and directly applying the data to the machine learning algorithms, and the results which were achieved were not promising. The distribution of the data plays an important role when the prediction or classification of a problem is to be done. This will help the model to find a pattern in the dataset that contributes to heart disease and which does not.



For checking the attribute values and determining the skewness of the data, many distribution plots are plotted so that some interpretation of the data can be seen. The distribution of age and sex, the distribution of chest pain and trestbps etc. are analyzed. This distribution helps us in analyzing the data and occurrence of heart disease.

For selecting the features and only choosing the important feature, the Lasso algorithm is used which is a part of embedded methods while performing feature selection. It shows better predictive accuracy than filter methods.

Machine learning proposed:

The proposed approach was applied to the dataset in which firstly the dataset was properly analyzed and then different machine learning algorithms consisting of linear model selection in which Logistic Regression was used. For focusing on neighbor selection technique K Neighbors Classifier was used, then tree-based technique like Decision Tree Classifier was used, and then a very popular and most popular technique of ensemble methods Random Forest Classifier was used in the report.

Deep learning proposed:

Two ways of deep learning approach was applied. One is using a sequential model and another is a functional deep learning approach. In this particular research, the first one is used. A sequential model with a fully connected dense layer is used, with the flatten and dropout layers to prevent the overfitting and the results are compared of the machine learning and deep learning and variations in the learning including computational time and accuracy can be analyzed.

For the evaluation process, confusion matrix, accuracy score, precision, recall, sensitivity, and F1 score are used. The conclusion which we found is that machine learning algorithms performed better in this analysis.

Conclusions:

The analysis of this research report helps in supporting my research questions regarding the dataset. A completed study of attributes is done. By identifying the most effective factors that contribute to the presence of heart disease will help health care providers to concentrate on those factors during the checkup and add these as part of quick diagnosis tests. This would also help categorize symptoms related

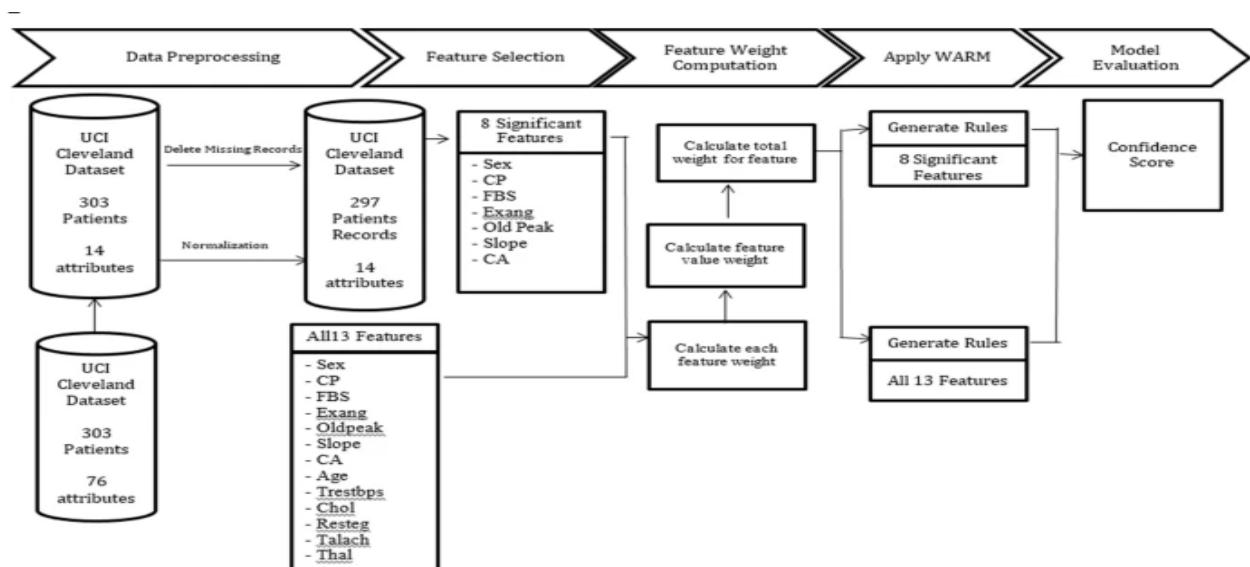
to heart disease and other illnesses. The analysis they have done will be very relatable to that of my research questions.

2.2 RESEARCH PAPER: 2

A NOVEL APPROACH FOR HEART DISEASE PREDICTION USING STRENGTH SCORES WITH SIGNIFICANT PREDICTORS

Introduction:

This paper is based on the diagnose cardiovascular disease based on the clinical tests and previous experience of diagnosing patients with similar symptoms. Patients who suffer from heart disease require quick diagnosis, early treatment and constant observations. To address their needs, many data mining approaches have been used in the past in diagnosing and predicting heart diseases. The study is aimed at predicting heart disease based on the scores of significant features using Weighted Associative Rule Mining. Weighted Association Rule Mining (WARM) is one of the data mining techniques used to discover the relationships between features and to determine mining rules that lead to certain predictions. The weight that is used in this mining technique provides users with a convenient way to indicate the importance of the features that contributes to heart disease and helps obtain more accurate rules. In many prediction models, different features have different importance. Hence, different weights are assigned to different features based on their predicting capabilities. The failure in determining the weight indicates the failure in determining the importance of the features.



Methodology

The retrieved Cleveland dataset went through a pre-processing phase. The significant features were retrieved from a total of 14 factors from the Cleveland dataset. Further, the weight of each significant feature was computed and assigned back to them accordingly. WARM was applied to the heart disease

dataset to generate rules. Finally, evaluation was performed to obtain the confidence score of the best rules generated using WARM based on significant features. Features were selected based on experiments conducted by Amin et al. since they had used the same dataset (UCI). They performed a set of experiments that dealt with 8100 combinations of features with 7 different classification models (K-NN, Decision Tree, Naïve Bayes, Logistic Regression, Neural Network and Vote) to identify significant features.

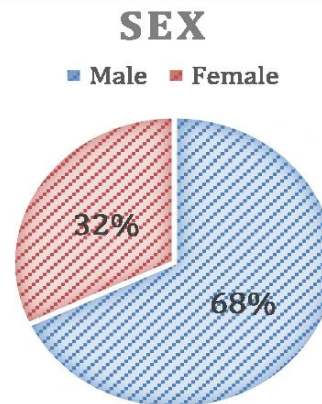
The fundamental of WARM states that different features in a dataset have different importance in predicting heart disease. The weight of each feature ranges from 0 to 1. Thus, a weight that is closer to 1 indicates a more significant feature. On the other hand, a weight that is closer to 0 is the least significant in heart disease prediction. Using the feature weight computation, they have done comparison between different attributes.

Male value is represented by 203 records and female by 94 records which gives a total of 297 records from the UCI dataset. To calculate the value of each feature weight, let A be the selected value and B be the rest of the features value,

$$W_{(\text{value}=A)} = \frac{A}{A \cup B}$$

Gender male value : $W_{(206)} = 203/297 = 0.68$

Gender female value : $W_{(97)} = 97/297 = 0.32$



Not all features in the heart disease dataset have the same level of significance in predicting the risk of heart disease. Thus, different weights based on their prediction capability are assigned. These values are then imported into Weka 3.8 to experiment with WARM using Apriori Algorithm. In the algorithmic process of Apriori, an item set X of length k is frequent if and only if every subset of X, having length k—1, is also frequent. This consideration results in a substantial reduction of search space and allows rule discovery in a computationally feasible time.

This phase generates rules based on the Apriori algorithm in Weighted Associative Rule Mining. Two sets of rules and confidence scores were generated for the followings:

(i) All features—this includes all the 13 features.

(ii) Selected significant features (8 features)

The rules were generated for all the features using the WARM. The highest confidence level achieved for predicting the risk of having heart disease is 96% and the number of features used to generate this rule is 3(CP, Slope and Thal).). The rule states that if the value of Chest Pain (CP) is asymptomatic, the slope is flat and the value of Thallium (Thal) is reversible, therefore, the patient has a very high tendency (confidence level = 96%) of having the risk of heart disease.

Conclusions:

The analysis done in this research report supports my research questions. This helps in understanding the significant features of the dataset that can be used for the prediction. They have extracted healthy rules from the dataset which helps in determining the physical activity needs to be stimulated in the patients to detect the heart disease in patients.

The extracted healthy rules and sick rules. This shows that the experiment with 8 significant features obtained the optimum confidence score of 100% for predicting healthy rules. The rules retrieved for this stated that if the sex is female, chest pain is non-angina and thallium heart scan is normal, this person is then predicted not to have heart disease. Assigning appropriate weight scores have proven to improve the performance of confidence level in the prediction. A set of significant features with different weights to represent the strength of each of the features was used in heart disease prediction. This was the first study that made use of significant features in executing WARM. This research has also contributed to listing the top rules in predicting heart disease based on the UCI dataset. This was the first research that benchmarked the healthy rules and sick rules with the highest confidence scores.

Future scope:

Future researches may look into predicting the risk levels of heart disease, as this will help medical practitioners and patients to gauge their heart disease severity. The algorithm used in this study for measuring weight can be further explored for use with other datasets to cater to other prediction models using the weighted approach. The machine learning techniques used in feature selection phase of this research is limited to the most popular techniques used in heart disease prediction research. Future researchers should look into exploring other machine learning techniques in selecting the significant features.

2.3 RESEARCH PAPER: 3

IMPROVING HEART DISEASE PREDICTION USING FEATURE SELECTION APPROACHES

Introduction:

This research paper mainly describes about the prediction of heart disease in medical field by using data science. As many researches done research related to that problem but the accuracy of prediction is still needed to be improved. So, this research focused on feature selection techniques and algorithms where

multiple heart disease datasets are used for experimentation analysis and to show the accuracy improvement. By using the Rapid miner as tool; Decision Tree, Logistic Regression, Logistic Regression SVM, Naïve Bayes and Random Forest; algorithms are used as feature selection techniques and improvement is shown in the results by showing the accuracy.

The proposed approach used to complete this research was started by downloading an open source UCI data set. After verifying the dataset, next step was preprocessing and data discretization in the form of Data cleaning, Data Transformation, Data Reduction, Binning and Select Attributes. After applying all these techniques on downloaded dataset, the main technique feature selection was applied. Later on, following algorithms were applied on the data i.e. Decision Tree, Logistic regression, Logistic regression SVM, Naïve Bayes and Random forest. After applying algorithms and techniques we could compare the results and discuss about conclusion. In this research paper, Rapid Miner was used. Rapid miner is a data analysis tool which already has instilled techniques to be implemented on the model and prepared data then could be convert data from csv file format to xlsx file format then import this data into the rapid miner in order to obtain the desired results. Applied ensemble process on data with sub process Minimum Redundancy Maximum Relevance Feature Selection (MRMR). The output after applying (MRMR) feature selection is in the form of weights. The next step is Data validation which is the main step of this experiment. Five techniques like decision tree, logistic regression, random forest, Naïve Bayes, logistic regression SVM are used as a sub process on the weighted data and results are obtained. The accuracy of Decision Tree is 82.22%, Logistic Regression 82.56%, Random Forest 84.17%, and Naïve Bayes 84.24% and Logistic Regression SVM is 84.85.

Conclusions:

This paper suggests Logistic Regression as a best feature selection technique for predicting heart disease. In future these techniques can also be applied on real time medical datasets and also can be used in the form of ensembles i.e. combinations of multiple techniques. This would result in increase of further accuracy and high performance. This feature selection technique could be taken from this research paper for my further analysis on the dataset.

Conclusion regarding all the three relevant research papers:

These three research papers help in confirming my research questions on the heart disease dataset. These papers have discussed on the effective factors that contribute to the presence of heart disease. By doing a study on heart disease dataset, helps in categorizing symptoms related to heart disease and other illness. In the research paper they have also extracted healthy rule and sick rule regarding the dataset which helps in determining the physical activity needs to be stimulated in the patients to detect the heart disease in patients.

3. DATASET ANALYSIS

Tools and methods used for the data analysis: Google Collab, Python and excel.

3.1 IMPORT DATASET

Before starting the study, we must import the dataset and required libraries. I've used Python and Google Collab for the entire analysis.

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
```

FIGURE 1. 1 IMPORT LIBRARIES

```
#Import the file
from google.colab import files
upload = files.upload()
```

Choose Files processed.cleveland.data

- processed.cleveland.data(n/a) - 18461 bytes, last modified: 10/4/2022 - 100% done

Saving processed.cleveland.data to processed.cleveland.data

FIGURE 1. 2 IMPORT THE DATASET

Create a dataframe and read the csv file.

```
#Create a dataframe
import pandas as pd
data = pd.read_csv('processed.cleveland.data', sep=",")
#Print the first 5 rows of the dataset
data.head()
```

	63.0	1.0	1.0.1	145.0	233.0	1.0.2	2.0	150.0	0.0	2.3	3.0	0.0.1	6.0	0
0	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	2.0	3.0	3.0	2
1	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	2.0	2.0	7.0	1
2	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	3.0	0.0	3.0	0
3	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	1.0	0.0	3.0	0
4	56.0	1.0	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	1.0	0.0	3.0	0

FIGURE 1. 3 CREATE DATAFRAME

The csv file did not contain a header list. Therefore, I have included the data frame with header list.

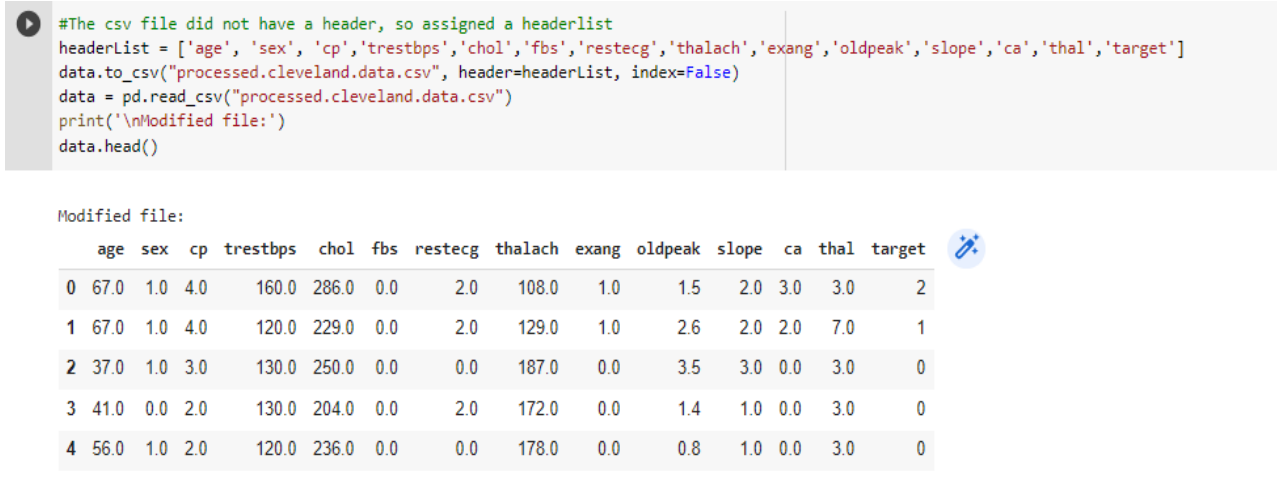


FIGURE 1. 4 DATAFRAME WITH HEADER LIST

Dataset information:

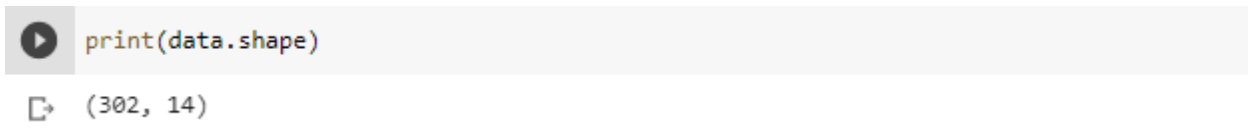


FIGURE 1. 5 SHAPE OF THE DATASET

In this dataset we have 302 rows and 14 attributes (columns).

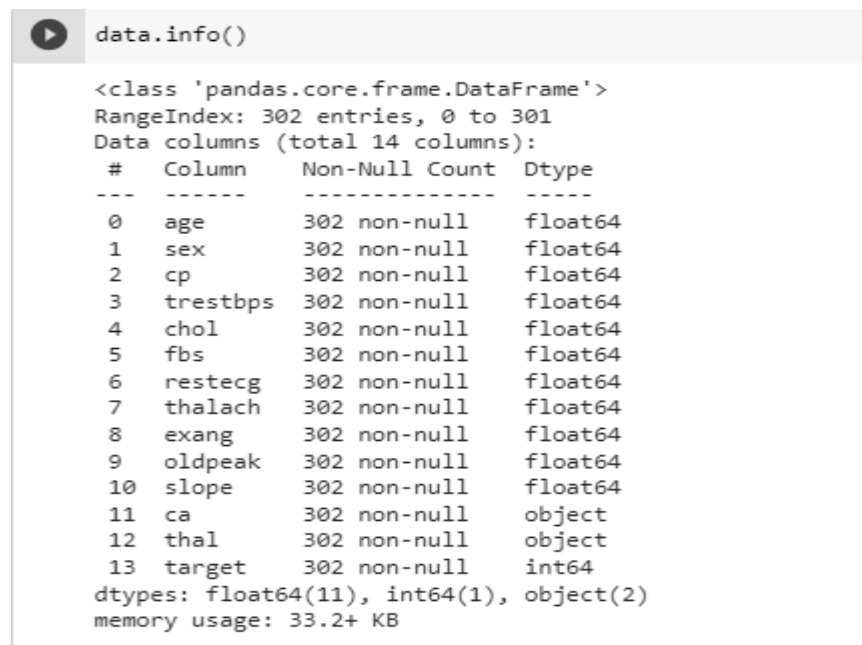


FIGURE 1. 6 DATA INFO

3.2 DATA PREPROCESSING AND DATA CLEANING

3.2.1 CHECKING FOR NULL AND DATA UNIQUE VALUES IN ALL COLUMNS

```
#checking for any null values
data.isnull().sum()
```

```
age      0
sex      0
cp       0
trestbps 0
chol     0
fbs      0
restecg  0
thalach  0
exang    0
oldpeak  0
slope    0
ca       0
thal     0
target   0
dtype: int64
```

FIGURE 1. 7: CHECK FOR NULL VALUES

We can infer from this that the dataset is free of null values. In order to confirm once more, we can examine the distinct values for each column to look for any data errors in the dataset.

```
for col in data.columns:
    print(col, data[col].unique())
```

```
age [67. 37. 41. 56. 62. 57. 63. 53. 44. 52. 48. 54. 49. 64. 58. 60. 50. 66.
 43. 40. 69. 59. 42. 55. 61. 65. 71. 51. 46. 45. 39. 68. 47. 34. 35. 29.
 70. 77. 38. 74. 76.]
sex [1. 0.]
cp [4. 3. 2. 1.]
trestbps [160. 120. 130. 140. 172. 150. 110. 132. 117. 135. 112. 105. 124. 125.
 142. 128. 145. 170. 155. 104. 180. 138. 108. 134. 122. 115. 118. 100.
 200. 94. 165. 102. 152. 101. 126. 174. 148. 178. 158. 192. 129. 144.
 123. 136. 146. 106. 156. 154. 114. 164.]
chol [286. 229. 250. 204. 236. 268. 354. 254. 203. 192. 294. 256. 263. 199.
 168. 239. 275. 266. 211. 283. 284. 224. 206. 219. 340. 226. 247. 167.
 230. 335. 234. 233. 177. 276. 353. 243. 225. 302. 212. 330. 175. 417.
 197. 198. 290. 253. 172. 273. 213. 305. 216. 304. 188. 282. 185. 232.
 326. 231. 269. 267. 248. 360. 258. 308. 245. 270. 208. 264. 321. 274.
 325. 235. 257. 164. 141. 252. 255. 201. 222. 260. 182. 303. 265. 309.
 307. 249. 186. 341. 183. 407. 217. 288. 220. 209. 227. 261. 174. 281.
 221. 205. 240. 289. 318. 298. 564. 246. 322. 299. 300. 293. 277. 214.
 207. 223. 160. 394. 184. 315. 409. 244. 195. 196. 126. 313. 259. 200.
 262. 215. 228. 193. 271. 210. 327. 149. 295. 306. 178. 237. 218. 242.
 319. 166. 180. 311. 278. 342. 169. 187. 157. 176. 241. 131.]
fbs [0. 1.]
restecg [2. 0. 1.]
thalach [108. 129. 187. 172. 178. 160. 163. 147. 155. 148. 153. 142. 173. 162.
 174. 168. 139. 171. 144. 132. 158. 114. 151. 161. 179. 120. 112. 137.
 157. 169. 165. 123. 128. 152. 140. 188. 109. 125. 131. 170. 113. 99.
 177. 141. 180. 111. 143. 182. 150. 156. 115. 149. 145. 146. 175. 186.
 185. 159. 130. 190. 136. 97. 127. 154. 133. 126. 202. 103. 166. 164.
 184. 124. 122. 96. 138. 88. 105. 194. 195. 106. 167. 95. 192. 117.
 121. 116. 71. 118. 181. 134. 90.]
exang [1. 0.]
oldpeak [1.5 2.6 3.5 1.4 0.8 3.6 0.6 3.1 0.4 1.3 0. 0.5 1.6 1. 1.2 0.2 1.8 3.2
 2.4 2. 2.5 2.2 2.8 3. 3.4 6.2 4. 5.6 2.9 0.1 2.1 1.9 4.2 0.9 1.1 3.8
 0.7 0.3 2.3 4.4]
slope [2. 3. 1.]
ca ['3.0' '2.0' '0.0' '1.0' '?']
thal ['3.0' '7.0' '6.0' '?']
target [2 1 0 3 4]
```

FIGURE 1. 8 UNIQUE VALUES IN THE DATASET

This makes it easier to comprehend that two columns named "ca" and "thal" contain "?" incorrect data. To provide better findings and forecasts, this should be eliminated from the data.

```
data['ca'].unique()

array(['3.0', '2.0', '0.0', '1.0', '?'], dtype=object)

data['thal'].unique()

array(['3.0', '7.0', '6.0', '?'], dtype=object)
```

FIGURE 1. 9 UNIQUE VALUES : CA AND THAL

I have changed the "?" to a null value in order to eliminate the character.

```
spec_chars = ["!", "\"", "#", "%", "&", "'", "(", ")",
              "*", "+", "-", ".", "/", ":", ";", "<",
              "=", ">", "?", "@", "[", "\\", "]", "^", "_",
              "`", "{", "|", "}", "~", "-"]
for char in spec_chars:
    data = data.replace(char, np.nan)
print(data)
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	
1	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	
2	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	
3	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	
4	56.0	1.0	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	
..	
297	45.0	1.0	1.0	110.0	264.0	0.0	0.0	132.0	0.0	1.2	
298	68.0	1.0	4.0	144.0	193.0	1.0	0.0	141.0	0.0	3.4	
299	57.0	1.0	4.0	130.0	131.0	0.0	0.0	115.0	1.0	1.2	
300	57.0	0.0	2.0	130.0	236.0	0.0	2.0	174.0	0.0	0.0	
301	38.0	1.0	3.0	138.0	175.0	0.0	0.0	173.0	0.0	0.0	

	slope	ca	thal	target
0	2.0	3.0	3.0	2
1	2.0	2.0	7.0	1
2	3.0	0.0	3.0	0
3	1.0	0.0	3.0	0
4	1.0	0.0	3.0	0
..
297	2.0	0.0	7.0	1
298	2.0	2.0	7.0	2
299	2.0	1.0	7.0	3
300	2.0	1.0	3.0	1
301	1.0	NaN	3.0	0

[302 rows x 14 columns]

FIGURE 1. 10 SPEACIAL CHARACTER CONVERSION

Now if we check for null values, we get the correct column with null values.

```
data.isnull().sum()
age      0
sex      0
cp       0
trestbps 0
chol     0
fbs      0
restecg  0
thalach  0
exang    0
oldpeak  0
slope    0
ca       4
thal     2
target   0
dtype: int64
```

FIGURE 1. 11 CHECK FOR NULL VALUES-2

I have used the Missingno library to visualize the values that are missing. The horizontal lines stand in for the missing values. This library offers an instructive method of viewing the missing values present in each column and determining whether there is any association between the missing values of various columns.

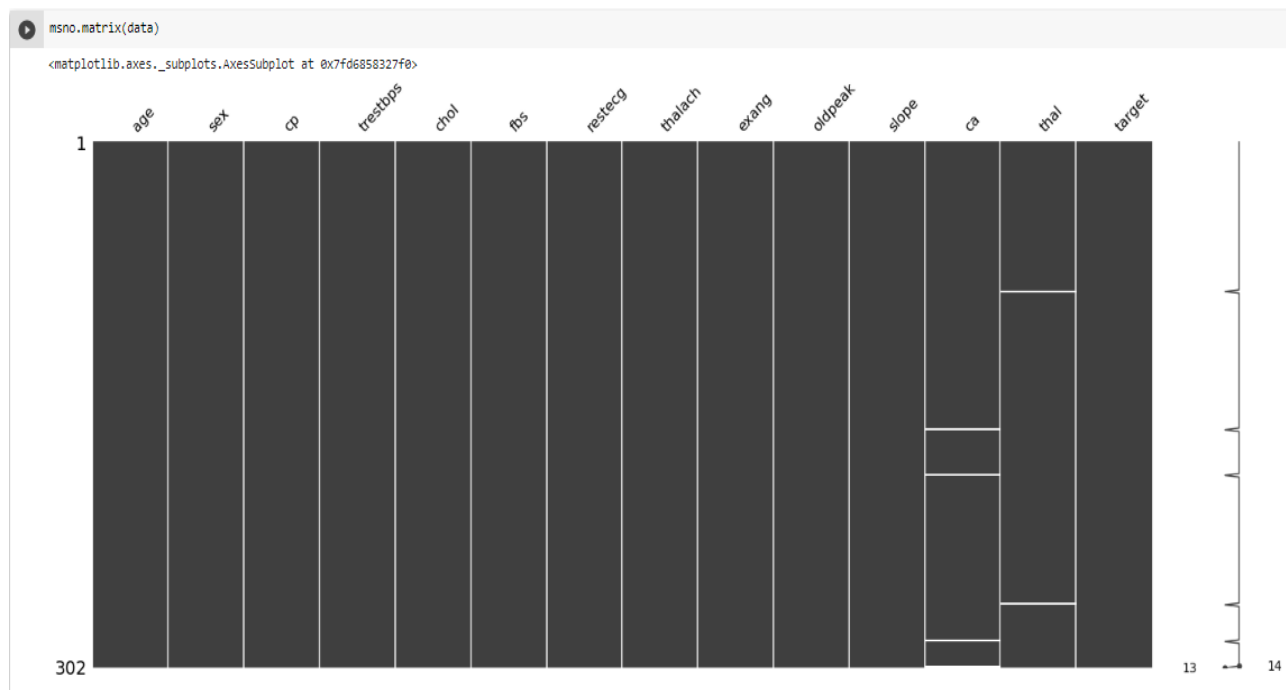


FIGURE 1. 12 VISUALIZING THE MISSING VALUES

3.2.2 DROPPING ROWS WITH NULL VALUES

```

▶ print('DataFrame after dropping the rows having missing values:')
my_data = data.dropna(axis=0)
print(my_data)

```

DataFrame after dropping the rows having missing values:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	\
0	67.0	1.0	4.0	160.0	286.0	0.0	2.0	108.0	1.0	1.5	
1	67.0	1.0	4.0	120.0	229.0	0.0	2.0	129.0	1.0	2.6	
2	37.0	1.0	3.0	130.0	250.0	0.0	0.0	187.0	0.0	3.5	
3	41.0	0.0	2.0	130.0	204.0	0.0	2.0	172.0	0.0	1.4	
4	56.0	1.0	2.0	120.0	236.0	0.0	0.0	178.0	0.0	0.8	
..	
296	57.0	0.0	4.0	140.0	241.0	0.0	0.0	123.0	1.0	0.2	
297	45.0	1.0	1.0	110.0	264.0	0.0	0.0	132.0	0.0	1.2	
298	68.0	1.0	4.0	144.0	193.0	1.0	0.0	141.0	0.0	3.4	
299	57.0	1.0	4.0	130.0	131.0	0.0	0.0	115.0	1.0	1.2	
300	57.0	0.0	2.0	130.0	236.0	0.0	2.0	174.0	0.0	0.0	

	slope	ca	thal	target
0	2.0	3.0	3.0	2
1	2.0	2.0	7.0	1
2	3.0	0.0	3.0	0
3	1.0	0.0	3.0	0
4	1.0	0.0	3.0	0
..
296	2.0	0.0	7.0	1
297	2.0	0.0	7.0	1
298	2.0	2.0	7.0	2
299	2.0	1.0	7.0	3
300	2.0	1.0	3.0	1

[296 rows x 14 columns]

FIGURE 1. 13: DATAFRAME AFTER DROPPING THE NULL VALUES

All of the missing values from the dataset were successfully deleted. The dataset currently contains 296 rows and 14 attributes (columns).

```

▶ print(my_data.shape)

```

(296, 14)

FIGURE 1. 14 DATA SHAPE AFTER REMOVING NULL & MISSING VALUES

3.2.3 CHECKING THE DATA TYPES

The data types in the dataset are validated. Most of the columns are given as float datatypes. Only the "Old Peak" column in the dataset is in float data type, while the rest are in integer data type.

```
data.dtypes
age      float64
sex      float64
cp       float64
trestbps float64
chol     float64
fbs      float64
restecg  float64
thalach  float64
exang    float64
oldpeak  float64
slope    float64
ca       object
thal     object
target   int64
dtype: object
```

FIGURE 1. 15 DATATYPES

Age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, slope, ca, and thal data types have been changed to integer data types.

```
my_data[['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'slope']] = my_data[['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'slope']].astype(int)

[79] my_data[['ca', 'thal']] = my_data[['ca', 'thal']].astype(float).astype(int)
```

```
my_data.dtypes
age      int64
sex      int64
cp       int64
trestbps int64
chol     int64
fbs      int64
restecg  int64
thalach  int64
exang    int64
oldpeak  float64
slope    int64
ca       int64
thal     int64
target   int64
dtype: object
```

FIGURE 1. 16 DATATYPE AFTER CONVERSION

3.2.4 CHECKING DUPLICATES

```
#checking for any data duplicates
duplicates = my_data.duplicated().sum()
if duplicates:
    print("The Duplilated rows available in the dataset")
else:
    print("There are no duplicates in the dataset")

There are no duplicates in the dataset
```

FIGURE 1. 17 CHECK FOR DUPLICATES

There were no duplicates available in the dataset.

3.2.5 GENERAL STATISTICS SUMMARY OF THE DATASET AND INTERPRETATION

[86] my_data.describe()

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000	296.000000
mean	54.513514	0.675676	3.165541	131.648649	247.398649	0.141892	0.993243	149.597973	0.327703	1.051351	1.597973	0.679054	4.726351	0.949324
std	9.051631	0.468915	0.958262	17.775956	52.078915	0.349530	0.994879	22.980401	0.470171	1.165841	0.613848	0.939726	1.940500	1.235410
min	29.000000	0.000000	1.000000	94.000000	126.000000	0.000000	0.000000	71.000000	0.000000	0.000000	1.000000	0.000000	3.000000	0.000000
25%	48.000000	0.000000	3.000000	120.000000	211.000000	0.000000	0.000000	133.000000	0.000000	0.000000	1.000000	0.000000	3.000000	0.000000
50%	56.000000	1.000000	3.000000	130.000000	243.000000	0.000000	1.000000	153.000000	0.000000	0.800000	2.000000	0.000000	3.000000	0.000000
75%	61.000000	1.000000	4.000000	140.000000	276.250000	0.000000	2.000000	166.000000	1.000000	1.600000	2.000000	1.000000	7.000000	2.000000
max	77.000000	1.000000	4.000000	200.000000	564.000000	1.000000	2.000000	202.000000	1.000000	6.200000	3.000000	3.000000	7.000000	4.000000

- This helps in understanding the general statistical study of the dataset. The describe () method returns description of the data in the Data Frame. If the Data Frame contains numerical data, the description contains this information for each column:

- ❖ count - The number of not-empty values
- ❖ mean - The average (mean) value
- ❖ std - The standard deviation.
- ❖ min - the minimum value
- ❖ 25% - The 25% percentile*
- ❖ 50% - The 50% percentile*
- ❖ 75% - The 75% percentile*
- ❖ max - the maximum value

3.2.6 TARGET ATTRIBUTE

The target attribute, num, is specified as follows in the dataset:

Num: diagnosis of heart disease (Angiographic disease status)

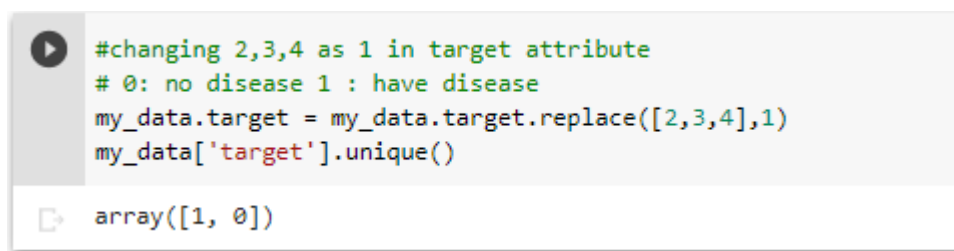
-- Value 0: < 50% diameter narrowing

-- Value 1: > 50% diameter narrowing

In the given dataset we have 0,1,2,3,4 values. For the better understanding and data visualization I have replaced the values 2,3,4 as 1.

1: defines people with heart disease

0: defines people without heart disease.



```
#changing 2,3,4 as 1 in target attribute
# 0: no disease 1 : have disease
my_data.target = my_data.target.replace([2,3,4],1)
my_data['target'].unique()

array([1, 0])
```

FIGURE 1. 18 TARGET ATTRIBUTE UPDATED

3.3 NOIR ANALYSIS

Nominal and Ordinal Datatypes are considered as categorical datatype.

NOMINAL: A set of things that may be recognized by their names or classifications and have distinctive values, or binomial features. Order is irrelevant.

ORDINAL: Items that can be ordered, such as government departments or military ranks, but whose differences cannot be quantified. There is no mathematical unit of measurement, although they can have order.

Interval and Ratio datatypes are considered as continuous datatype.

INTERVAL: Items with order, a mathematical unit of measurement, and a quantifiable distance between them but no absolute zero.

RATIO: Items with order, a mathematical unit of measurement, and a quantifiable distance between them with absolute zero.

Categorical datatype: sex, fbs, exang, target, cp, restecg, slope, ca, thal

Continuous datatype: age, trestbps, chol, thalach, old peak

SL NO:	ATTRIBUTE	DESCRIPTION	DATA TYPE
1	Age	Patient Age in years	Ratio
2	Sex	Gender of the Patients(Male : 1 Female:0)	Nominal
3	Cp	Chest pain experienced by the patients can be categorized into 1,2,3 and 4.	Ordinal
4	trestbps	Level of blood pressure at rest mode	Interval
5	chol	Serum Cholesterol	Ratio
6	fbs	Blood sugar level on fasting; True : 1 and False : 0	Nominal
7	restecg	Resting electrocardiographic result	Ordinal
8	thalach	Maximum heart rate achieved	Interval
9	Exang	Exercise induced angina (1 = yes; 0 = no)	Nominal
10	old peak	ST depression induced by exercise relative to rest	Ratio
11	Slope	ST segment measured in the forms of slope during peak exercise. (1,2,3)	Ordinal
12	Ca	number of major vessels (0-3) colored by fluoroscopy	Ordinal
13	Thal	Thallium stress which can be categorized into 3,6,7	Ordinal
14	Target	Diagnosis of heart disease which can predict if the patient has disease or not. 1: Have disease 0: No disease	Nominal

TABLE 1. 1 NOIR ANALYSIS

I changed the attribute values for easier understanding prior to beginning the categorical attribute plotting.

```

my_data['sex'] = my_data.sex.replace({1: "Male", 0: "Female"})
my_data['cp'] = my_data.cp.replace({1: "typical_angina", 2: "atypical_angina", 3:"non-anginal pain", 4: "asymtomatic"})
my_data['fbs'] = my_data.fbs.replace({1: "True", 0: "False"})
my_data['restecg'] = my_data.restecg.replace({0: "normal", 1: "ST-T wave", 2:"Left ventricular hypertrophy"})
my_data['exang'] = my_data.exang.replace({1: "Yes", 0: "No"})
my_data['slope'] = my_data.slope.replace({1: "upsloping", 2: "flat", 3:"downsloping"})
my_data['thal'] = my_data.thal.replace({6: "fixed_defect", 7: "reversible_defect", 3:"normal"})
my_data['target'] = my_data.target.replace({1: "Have Disease", 0: "No disease"})

```



```

my_data.head()

```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	67	Male	asymtomatic	160	286	False	Left ventricular hypertrophy	108	Yes	1.5	downsloping	3	normal	Have Disease
1	67	Male	asymtomatic	120	229	False	Left ventricular hypertrophy	129	Yes	2.6	downsloping	2	reversible_defect	Have Disease
2	37	Male	non-anginal pain	130	250	False	normal	187	No	3.5	downsloping	0	normal	No disease
3	41	Female	atypical_angina	130	204	False	Left ventricular hypertrophy	172	No	1.4	flat	0	normal	No disease
4	56	Male	atypical_angina	120	236	False	normal	178	No	0.8	flat	0	normal	No disease

FIGURE 1. 19 ATTRIBUTE VALUE UPDATION

3.4 DATA VISUALIZATION

The dataset explores the three research questions.

3.4.1 TARGET DISTRIBUTION IN THE DATASET

```
my_data['target'].value_counts()

No disease      159
Have Disease    137
Name: target, dtype: int64
```

FIGURE 1. 20 GETTING COUNT VALUES

```
fig, ax = plt.subplots(figsize=(10,8))
Target_analysis = my_data.target.value_counts()
Target_analysis.plot.bar(title="Diagnosis of Heart disease", ylabel='Count',
                        color = ['black', 'grey'], ax=ax)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.09, i.get_height()-50, \
            str(round((i.get_height()/total)*100, 2))+'%', fontsize=14,
            color='white', weight = 'bold')

plt.tight_layout()
```

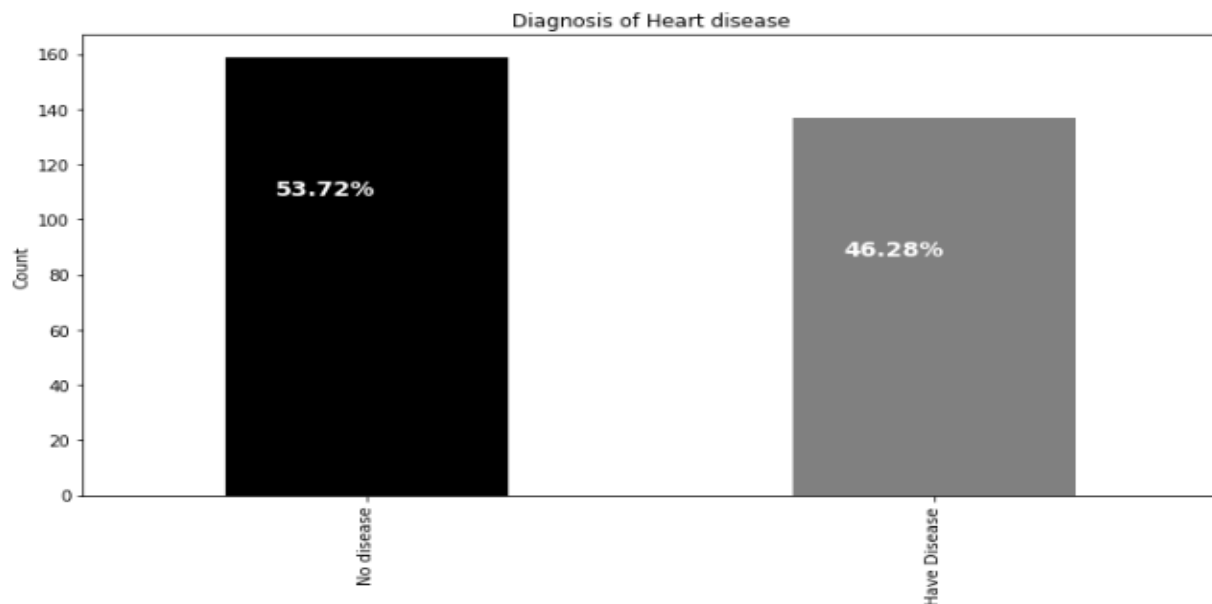


FIGURE 1. 21 TARGET DISTRIBUTION ANALYSIS

The dataset is balanced with 53.72% individuals without heart disease and 46.28% with heart disease.

RESEARCH QUESTION: How has gender and age affected the patterns of heart disease occurrence in patient for the past few years?

3.4.2 GENDER ANALYSIS IN THE DATASET

```
[▶] fig, ax = plt.subplots(figsize=(8, 7))
p = sns.countplot(x="sex", data = my_data, hue='target', palette='BuPu', ax = ax)
plt.xlabel('Gender', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("GENDER ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.05, i.get_height()-15,
            str(round((i.get_height()/total)*100, 2))+'%', fontsize=14,
            color='white', weight = 'bold')
plt.tight_layout()
```

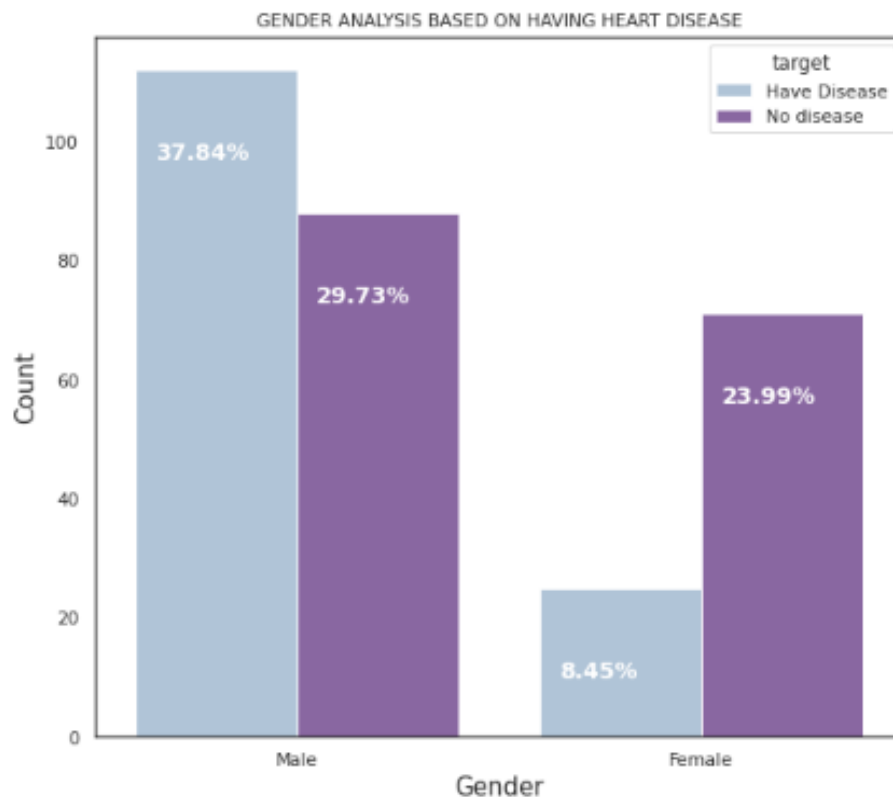


FIGURE 1. 22 GENDER ANALYSIS

We can see from the dataset that there are more male disease patients than female ones.

3.4.3 AGE ANALYSIS IN THE DATASET

```

min_age = min(my_data['age'])
max_age = max(my_data['age'])
mean_of_age = my_data.age.mean()
fig, ax = plt.subplots(figsize=(10, 7))
sns.barplot(x = my_data.age.value_counts()[:10].index, y = my_data.age.value_counts()[:10].values, palette = 'BuPu', ax = ax)
plt.xlabel('Age')
plt.ylabel('Age distribution')
print("The minimum age in the dataset is :", min_age)
print("The maximum age in the dataset is :", max_age)
print("The mean of the age in the dataset is :", mean_of_age)

```

```

The minimum age in the dataset is : 29
The maximum age in the dataset is : 77
The mean of the age in the dataset is : 54.513513513513516

```

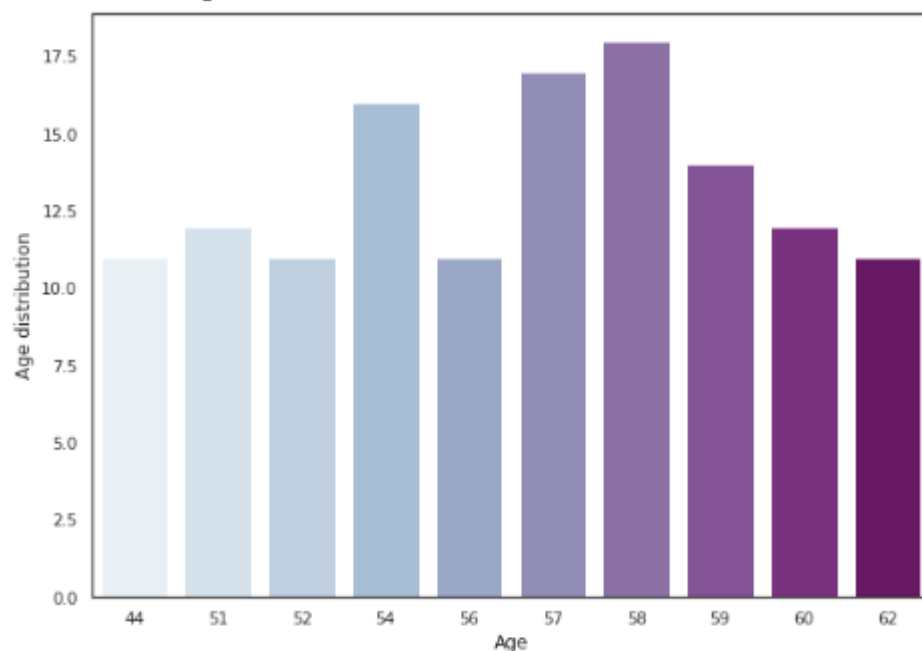


FIGURE 1. 23 AGE DISTRIBUTION IN THE RANGE OF 10

The majority of the patients are in their fifties and sixties. The mean age is about 54 years; the youngest is 29 and the oldest is 77.

Age Analysis based on the target:

```

[189] fig, ax = plt.subplots(figsize=(19, 7))
      p = sns.countplot(x="age", data = my_data, hue='target', palette='icefire', ax = ax)
      plt.xlabel('Age', fontsize = 15)
      plt.ylabel('Count', fontsize = 15)
      plt.title("AGE ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)

```

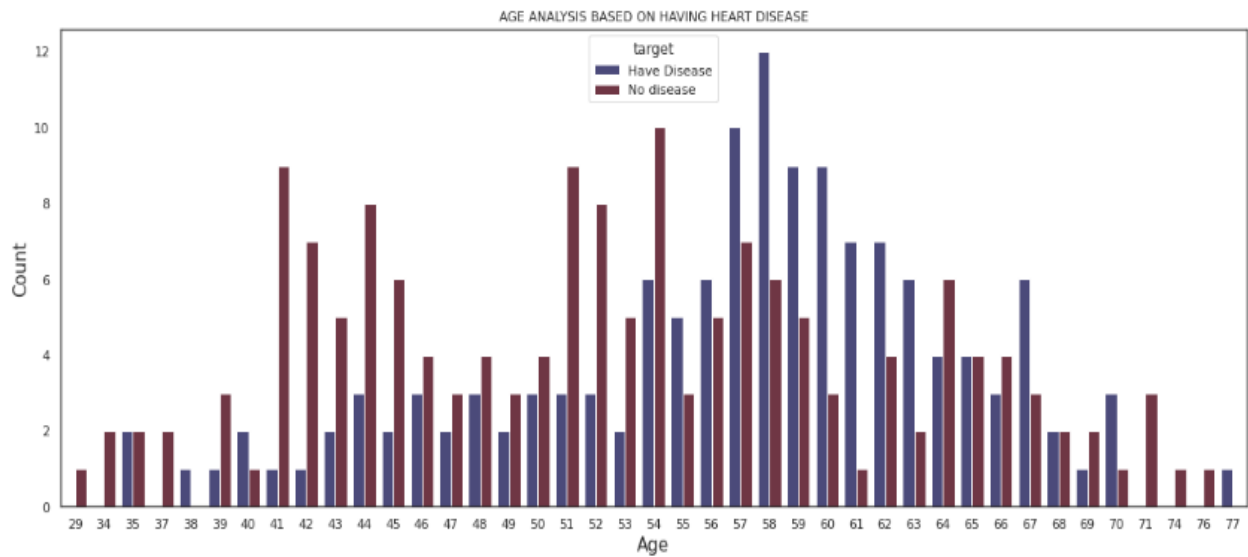


FIGURE 1. 24 AGE ANALYSIS

It is obvious from this graph that the individuals in their fifties and sixties are heart disease sufferers. People who are 58 years old or older in this dataset have heart disease.

3.4.4 COORELATIN BASED FEATURE SELECTION IN THE DATASET

RESEARCH QUESTION: What are the most effective factors in determining the presence of heart diseases in a patient?

my_data.corr()

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.000000	-0.094802	0.118743	0.288805	0.203846	0.125821	0.147193	-0.395204	0.098919	0.194405	0.153807	0.365356	0.124787	0.230677
sex	-0.094802	1.000000	0.014272	-0.068212	-0.197629	0.033539	0.031618	-0.060586	0.145444	0.104357	0.028328	0.093769	0.382707	0.281261
cp	0.118743	0.014272	1.000000	-0.031599	0.070606	-0.040004	0.072291	-0.342089	0.375759	0.213564	0.171151	0.232361	0.276014	0.405980
trestbps	0.288805	-0.068212	-0.031599	1.000000	0.132380	0.176636	0.147075	-0.049199	0.068578	0.189078	0.116556	0.099967	0.136750	0.156210
chol	0.203846	-0.197629	0.070606	0.132380	1.000000	0.015132	0.166298	-0.000058	0.058744	0.039676	-0.007164	0.115387	0.011481	0.079546
fbs	0.125821	0.033539	-0.040004	0.176636	0.015132	1.000000	0.061255	-0.008067	0.004878	-0.000472	0.029783	0.159755	0.057441	0.010889
restecg	0.147193	0.031618	0.072291	0.147075	0.166298	0.061255	1.000000	-0.072474	0.084466	0.110482	0.128753	0.131828	0.016598	0.170041
thalach	-0.395204	-0.060586	-0.342089	-0.049199	-0.000058	-0.008067	-0.072474	1.000000	-0.384642	-0.348376	-0.392858	-0.268921	-0.275070	-0.424377
exang	0.098919	0.145444	0.375759	0.068578	0.058744	0.004878	0.084466	-0.384642	1.000000	0.292629	0.258355	0.146783	0.328979	0.420130
oldpeak	0.194405	0.104357	0.213564	0.189078	0.039676	-0.000472	0.110482	-0.348376	0.292629	1.000000	0.576984	0.297897	0.343520	0.428842
slope	0.153807	0.028328	0.171151	0.116556	-0.007164	0.029783	0.128753	-0.392858	0.258355	0.576984	1.000000	0.116398	0.277282	0.343609
ca	0.365356	0.093769	0.232361	0.099967	0.115387	0.159755	0.131828	-0.268921	0.146783	0.297897	0.116398	1.000000	0.258398	0.462007
thal	0.124787	0.382707	0.276014	0.136750	0.011481	0.057441	0.016598	-0.275070	0.328979	0.343520	0.277282	0.258398	1.000000	0.529841
target	0.230677	0.281261	0.405980	0.156210	0.079546	0.010889	0.170041	-0.424377	0.420130	0.428842	0.343609	0.462007	0.529841	1.000000

corr() is used to find the pairwise correlation of all columns in the Pandas data frame in Python. The function helped me understand the correlation between target and other columns in the dataset.

- -1: Indicates a perfectly negative linear correlation between two variables.
- 0: Denotes that there is no linear correlation between two variables.
- 1: A perfectly positive linear correlation exists between two variables.

```
26] sns.set(style="white")
plt.rcParams['figure.figsize'] = (15, 10)
sns.heatmap(my_data.corr(), annot = True, linewidths=.5, cmap="BuPu")
plt.title('CORRELATION BETWEEN VARIABLES', fontsize = 15)
```

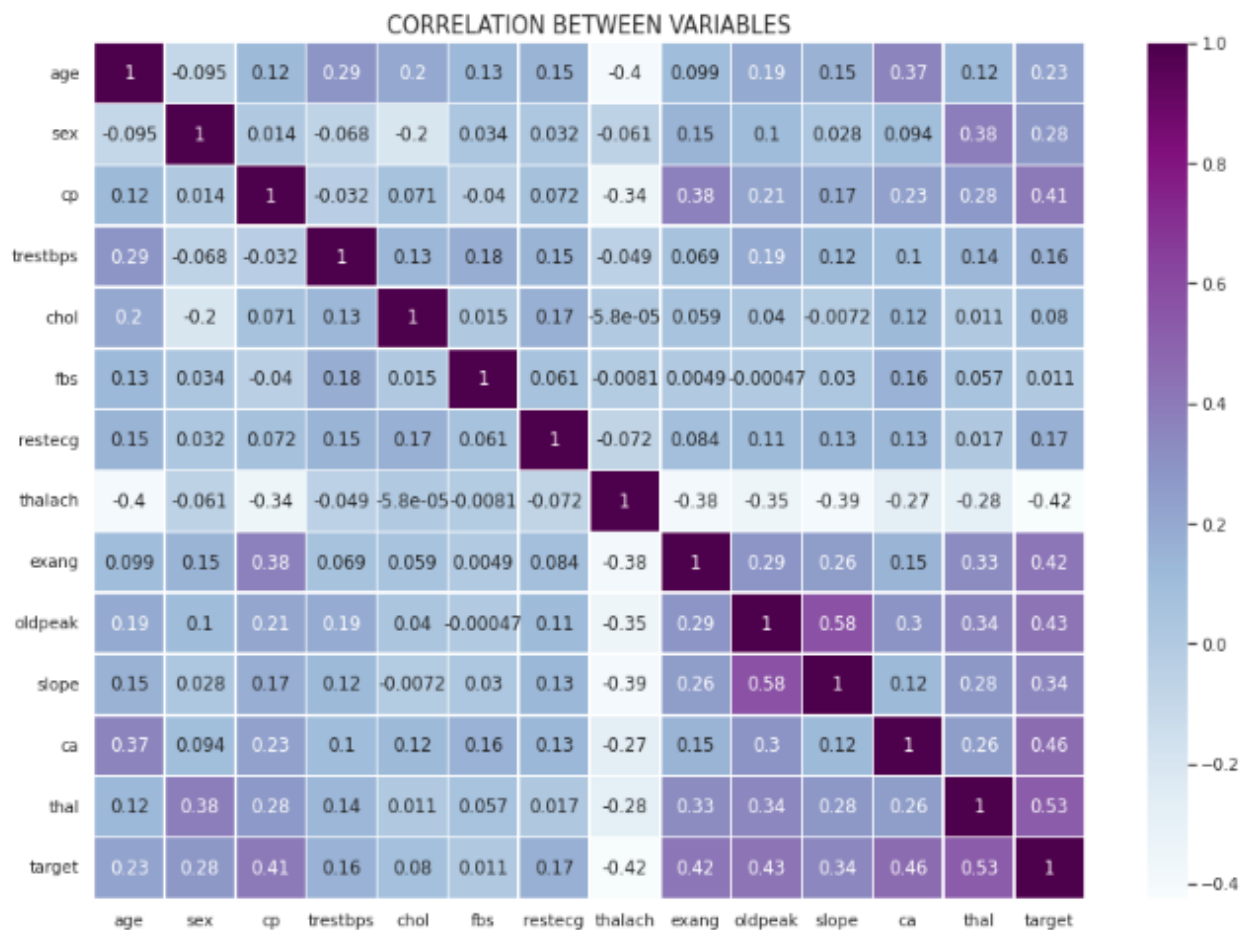


FIGURE 1. 25 CORRELATION MATRIX

GRAPH INTERPRETATION:

- It is high positively correlated at 1.0 (top) and negatively correlated at -0.4 (bottom)
- cp, exang, old peak, ca, thal, slope shows good positive correlation with target
- thalach shows a good negative correlation with target
- age, sex, trestbps, chol, fbs, restecg has low correlation with the target

By doing so, it is possible to identify crucial elements that may contribute to heart disease.

3.4.5 ANALYSIS BASED ON THE PHYSICAL ACTIVITY INDUCED ON THE PAITENTS

RESEARCH QUESTION: How much physical activity needs to be stimulated in the patients to detect the heart disease in patients?

EXANG ANALYSIS:

```
fig, ax = plt.subplots(figsize=(8, 6))
p = sns.countplot(x="exang", data = my_data, hue='target', palette='icefire', ax = ax)
plt.xlabel('exang', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("EXERCISE INDUCED ANGINA ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)
```

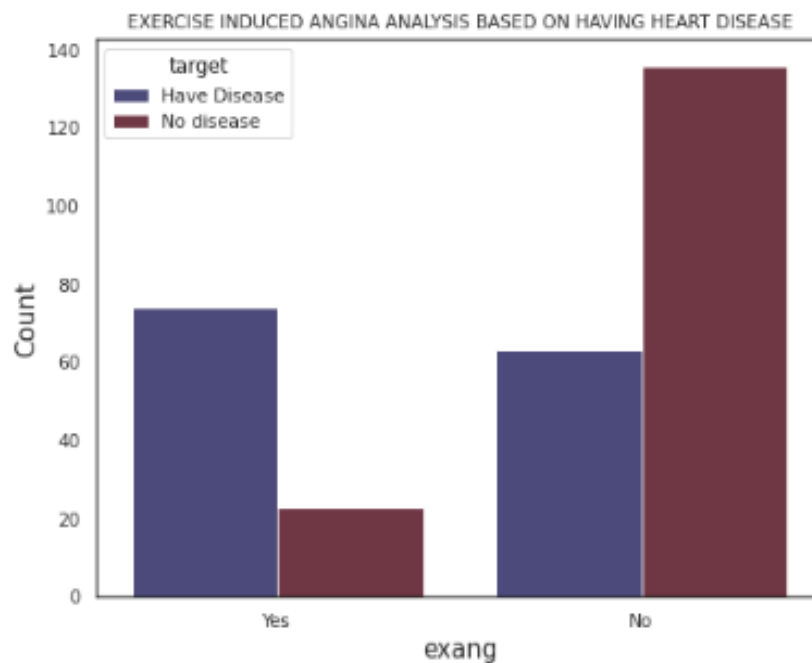


FIGURE 1. 26 EXERCISE INDUCED ANGINA ANALYSIS ON TARGET

Exercise induced angina: This attribute indicates the angina pain after doing heavy exercise.

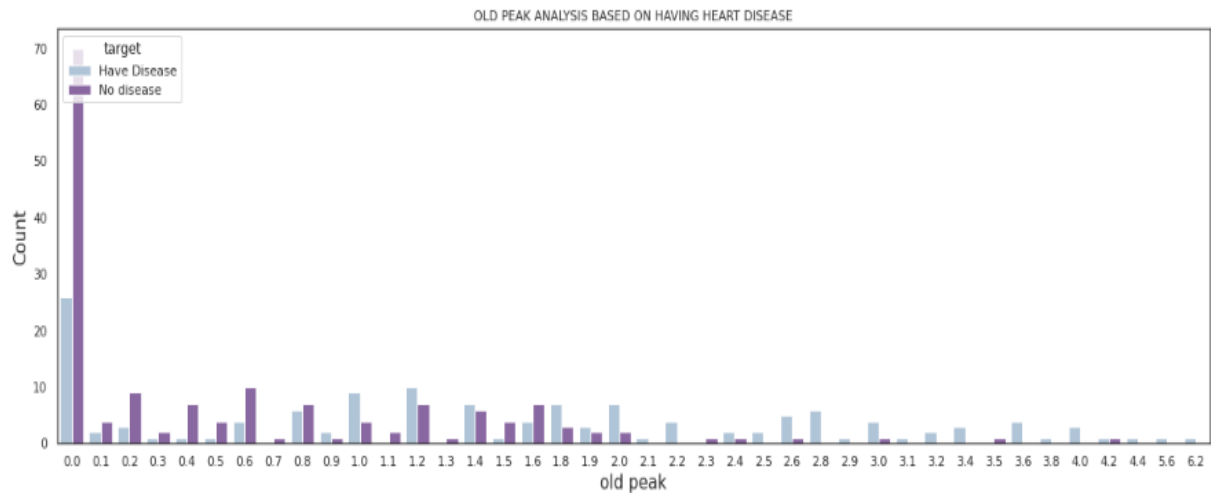
YES: Indicates pain

NO: Indicates no pain

It is obvious from this graph that those with heart disease are the ones who experience discomfort after engaging in strenuous exercise.

OLD PEAK ANALYSIS:

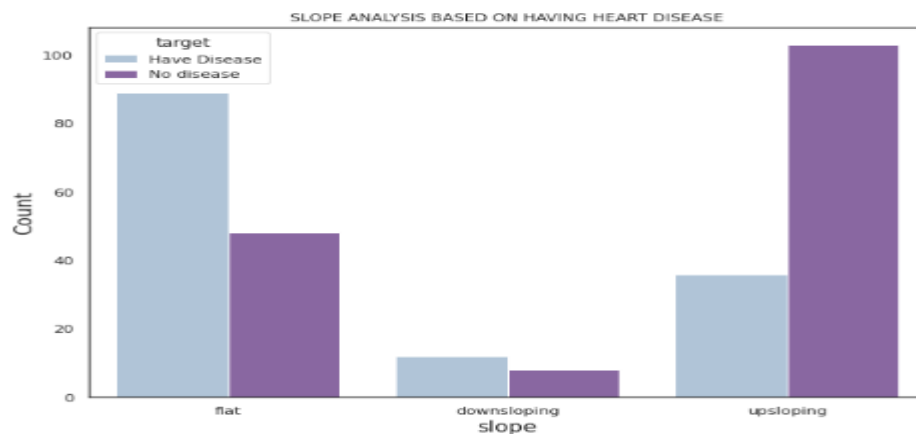
```
fig, ax = plt.subplots(figsize=(20, 6))
p = sns.countplot(x="oldpeak", data = my_data, hue='target', palette='BuPu', ax = ax)
plt.xlabel('old peak', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("OLD PEAK ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)
```

**FIGURE 1. 27 OLD PEAK ANALYSIS**

Old peak: ST depression induced by exercise relative to rest looks at stress of heart during exercise. A heart that isn't healthy will stress more. From this graph, we can see the old peak = 1.2 has the more chance of having disease. These characteristics assist us in identifying individuals who have heart disease by requiring them to engage in physical activity.

SLOPE ANALYSIS:

```
fig, ax = plt.subplots(figsize=(10, 7))
p = sns.countplot(x="slope", data = my_data, hue='target', palette='BuPu', ax = ax)
plt.xlabel('slope', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("SLOPE ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)
```

**FIGURE 1. 28 SLOPE ANALYSIS**

We can deduct from the graph that the healthy individuals will have an upward sloping. (A higher heart rate following activity)

3.4.6 CHEST PAIN AND FASTING BLOOD SUGAR ANALYSIS

```
fig, ax = plt.subplots(figsize=(10, 7))
p = sns.countplot(x="cp", data = my_data, hue='target', palette='icefire', ax = ax)
plt.xlabel('Chest pain', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("CHEST PAIN ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 14)
```

Cp: Chest pain type

- 1: Typical angina: chest pain related decrease blood supply to the heart
- 2: Atypical angina: chest pain not related to heart
- 3: Non-angina pain: typically, esophageal spasms (non-heart related)
- 4: Asymptomatic: chest pain not showing signs of disease (Silent attack)

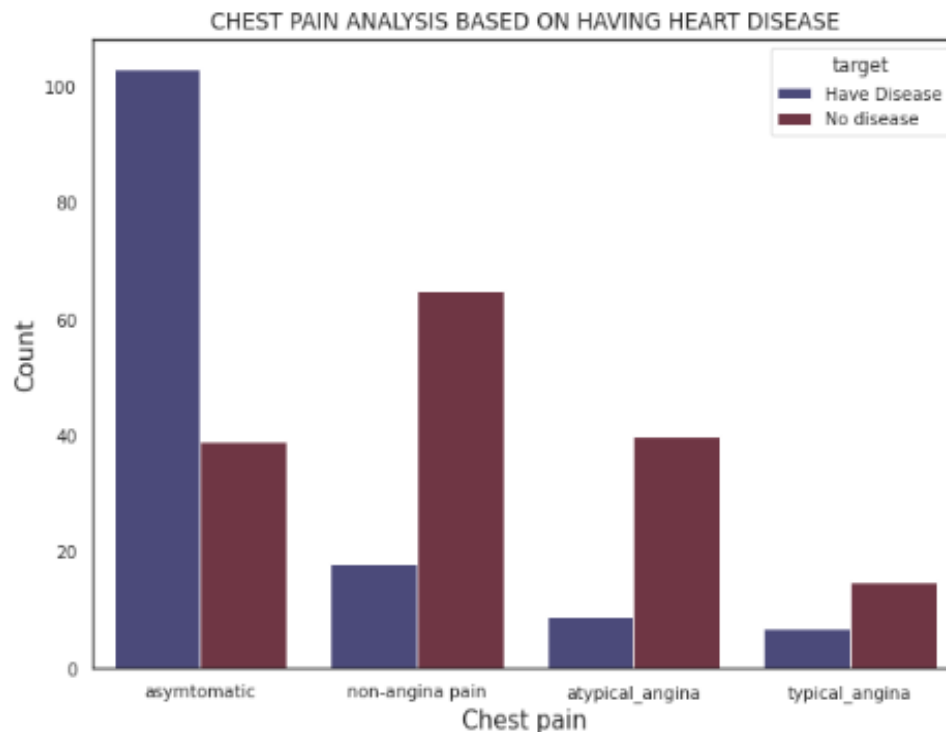


FIGURE 1. 29 DATASET ANALYSIS : CHEST PAIN ANALYSIS

This graph shows that there are more people in the sample who have asymptomatic chest discomfort. A temporary change in myocardial perfusion without chest pain or the typical angina symptoms is known as silent (asymptomatic) myocardial ischemia (SMI).

```
fig, ax = plt.subplots(figsize=(10, 7))
p = sns.countplot(x="fbs", data = my_data, hue='target', palette='icefire', ax = ax)
plt.xlabel('Fasting blood sugar', fontsize = 15)
plt.ylabel('Count', fontsize = 15)
plt.title("FASTING BLOOD SUGAR DISTRIBUTION ACCORDING TO TARGET VARIABLE", fontsize = 15)
plt.text(1,90,'True: 1')
plt.text(1,85,'False: 0')

totals = []
for i in ax.patches:
    totals.append(i.get_height())
total = sum(totals)
for i in ax.patches:
    ax.text(i.get_x()+.05, i.get_height()-15,
            str(round((i.get_height()/total)*100, 2))+ '%', fontsize=14,
            color='white', weight = 'bold')
plt.tight_layout()
```

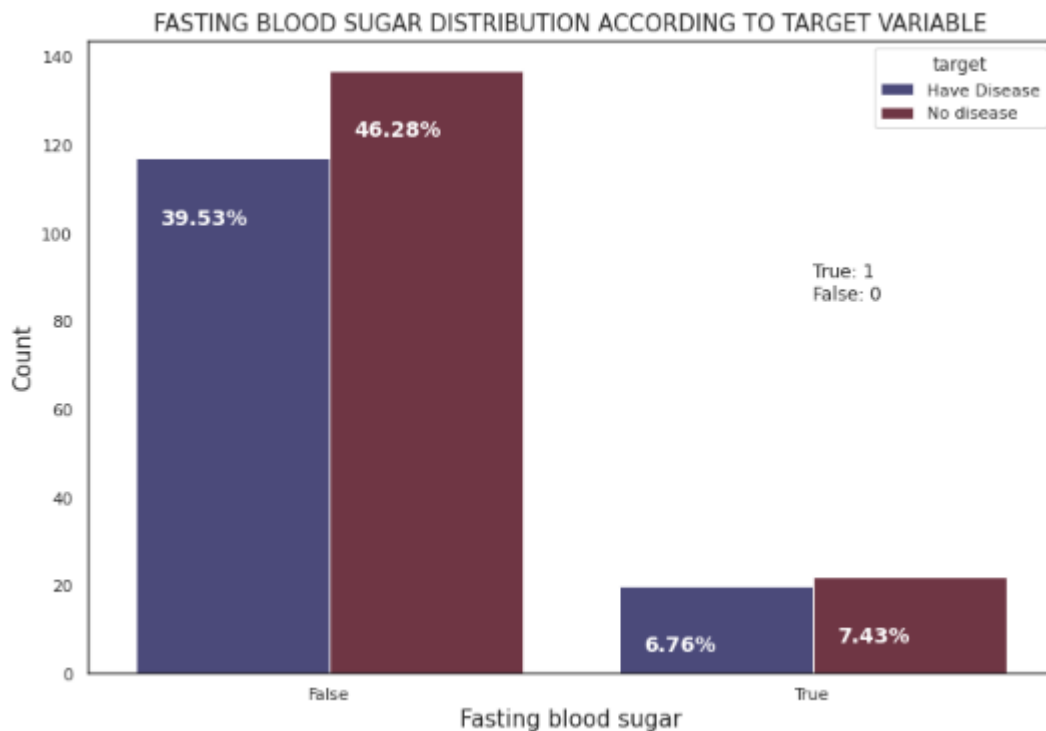


FIGURE 1. 30 FASTING BLOOD SUGAR ANALYSIS

FBS: Here, we observe that the number for class true, is lower compared to class false. However, if we look closely, there are lower number of heart disease patient without diabetes.

This provide an indication that fbs might not be a strong feature differentiating between heart disease and non-disease patient.

3.4.7 DISTRIBUTION AND HISTOGRAM PLOTS

```
my_data.hist(edgecolor = "black", color = 'indigo', figsize=(20,13))
```

By contrasting the empirical distribution of the data with the theoretical values anticipated from a certain distribution, distribution plots provide a visual assessment of the distribution of sample data.

- normal distribution: age, trestbps
- left skewed: old peak, chol
- right skewed: thalach

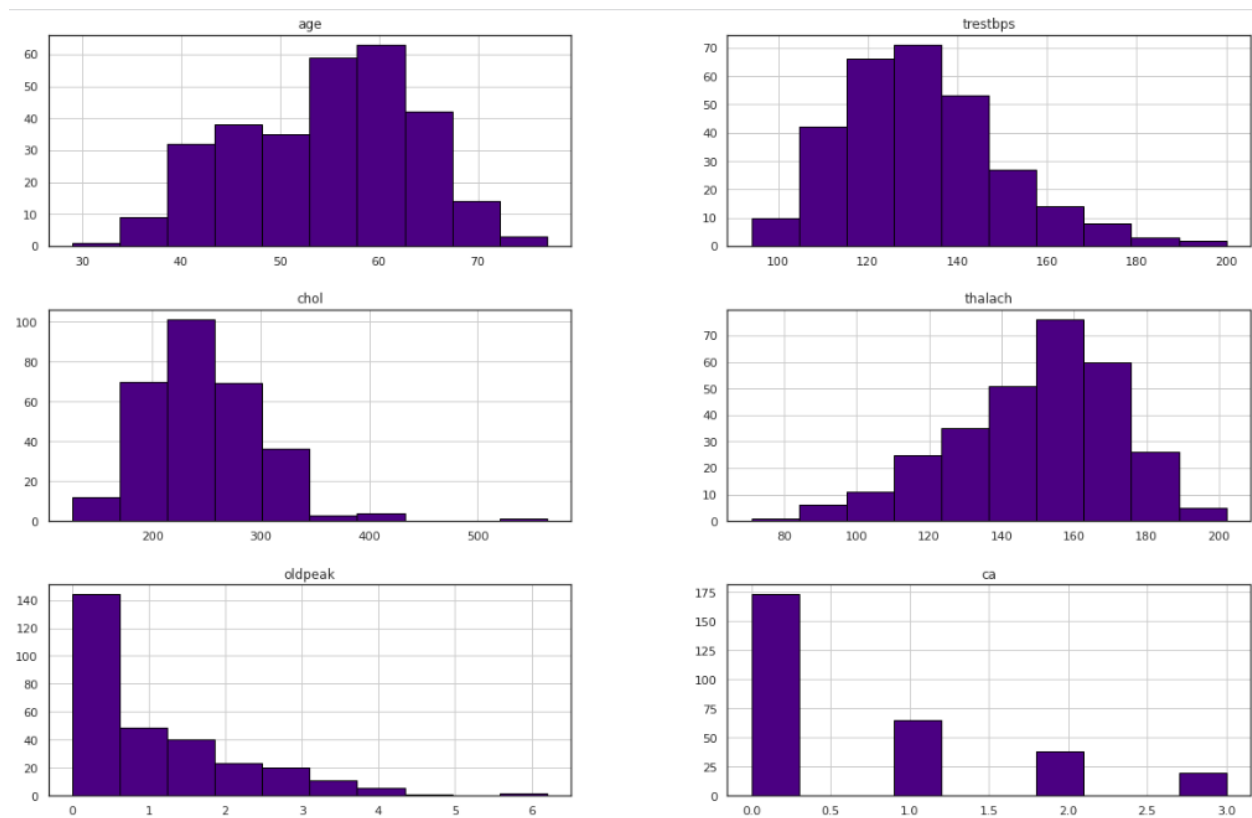


FIGURE 1. 31 DISTRIBUTION PLOTS

RESTING ECG WITH RESPECT TO TARGET:

```
plt.figure(figsize=(10, 6))
sns.histplot(data = my_data, x = 'restecg', kde = True, hue = 'target', palette='icefire')
plt.xlabel("Resting ECG", fontsize=20)
plt.ylabel("Count", fontsize=20)
plt.title("Resting ECG WITH RESPECT TO TARGET", fontsize=20)
plt.show()
```

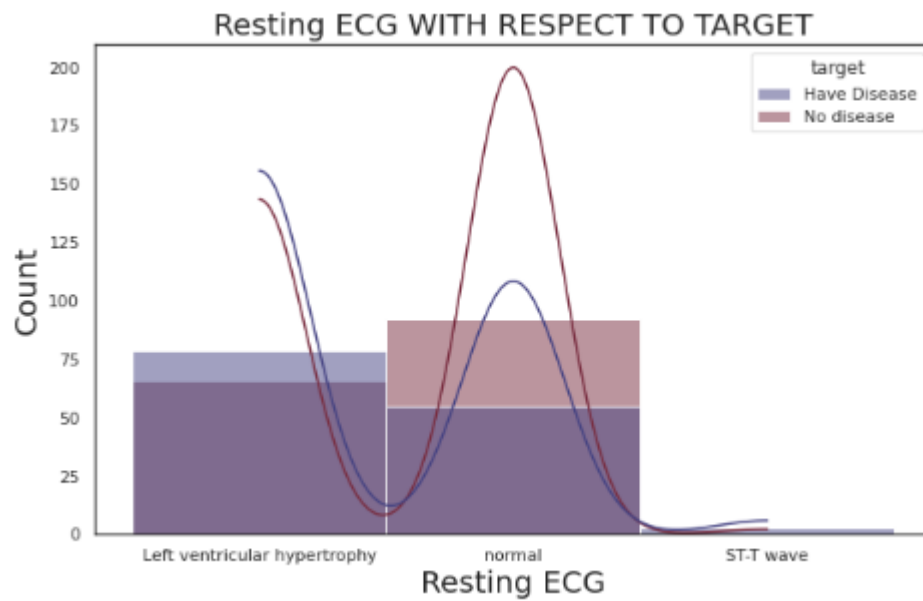


FIGURE 1.32 RESTING ECG ANALYSIS

People with normal ECG are healthy people compared to that of diseased individuals.

THALLIUM STRESS WITH RESPECT TO TARGET:

```
plt.figure(figsize=(10, 6))
sns.histplot(data = my_data, x = 'thal', kde = True, hue = 'target', palette='icefire')
plt.xlabel("Thallium stress", fontsize=20)
plt.ylabel("Count", fontsize=20)
plt.title("THALLIUM STRESS WUTH RESPECT TO TARGET", fontsize=20)
plt.show()
```

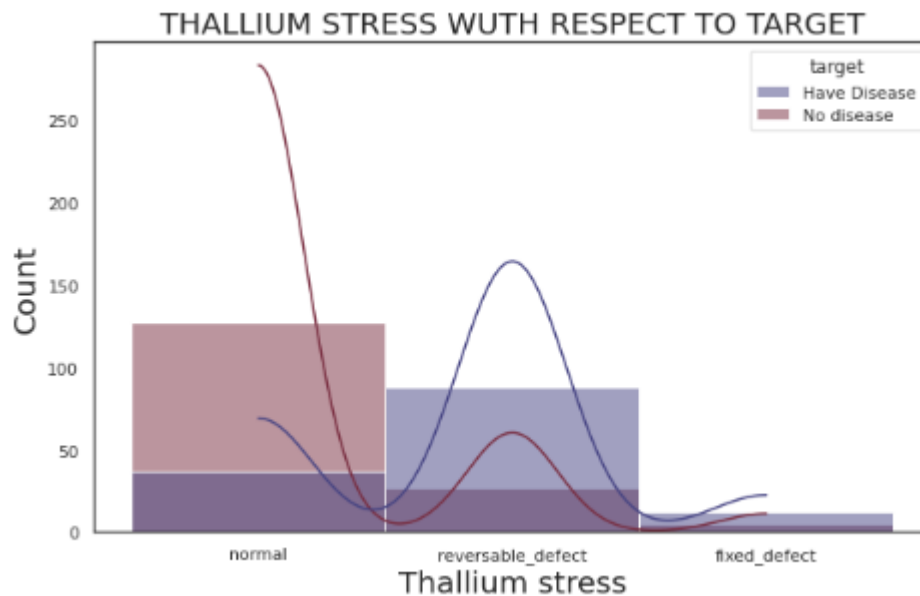


FIGURE 1.33 THALLIUM STRESS ANALYSIS

People who have a reversible defect are more likely to have heart disease.

CA WITH RESPECT TO TARGET:

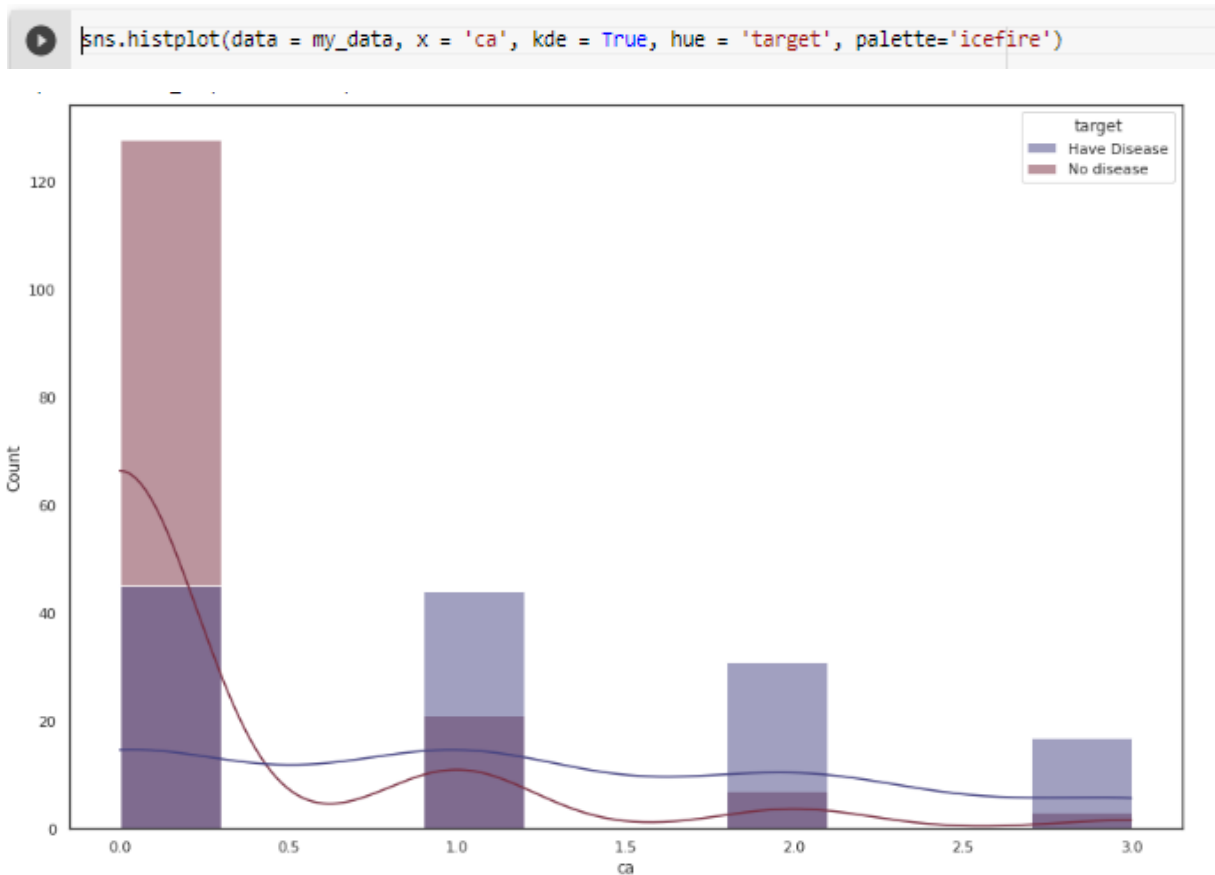


FIGURE 1. 34 CA ANALYSIS

ca: number of major vessels (0-3) colored by fluoroscopy —> When the value 0 there is no heart disease patients.

3.4.8 COMPARING TWO ATTRIBUTES WITH RESPECT TO TARGET

```
plt.figure(figsize=(9,8))
p = sns.lineplot(data = my_data, x = 'age', y = 'trestbps', err_style="bars", hue = 'sex', palette='icefire', ci = 0, legend='full')
plt.xlabel('Age', fontsize = 15)
plt.ylabel('Resting blood pressure', fontsize = 10)
plt.title("Age vs Resting BP ", fontsize = 10)
plt.show(p)

plt.figure(figsize=(9,8))
r = sns.lineplot(data = my_data, x = 'age', y = 'chol', err_style="bars", hue = 'sex', palette='icefire', ci = 0, legend='full')
plt.xlabel('Age', fontsize = 15)
plt.ylabel('cholesterol', fontsize = 10)
plt.title("Age vs cholesterol ", fontsize = 10)
plt.show(r)
```

GRAPH INTERPRETATION:

AGE VS RESTING BP: Resting blood pressure rises with age, and this dataset shows that resting blood pressure rises dramatically in women.

AGE VS CHOLESTROL: As people get older, their cholesterol levels rise. Women are more likely to be affected than men.

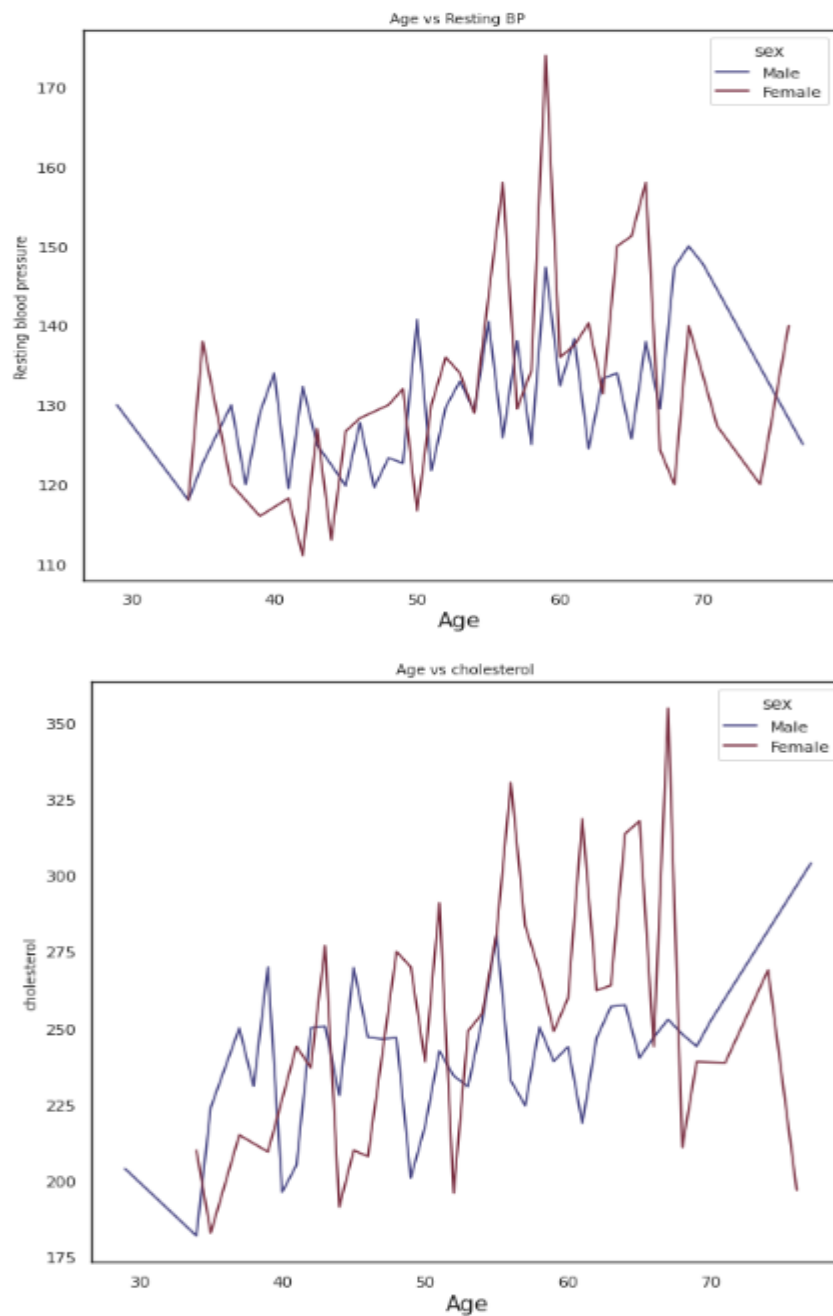


FIGURE 1. 35 COMPARSION WITH TWO ATTRIBUTES AGAINST TARGET

3.4.9 RELATIONSHIP PLOT

```
plt.figure(figsize=(10, 4))
sns.relplot(data = my_data, x='trestbps', y='thalach', hue='target', size='chol', sizes=(20, 300), alpha=.5, palette="icefire", height=6)
plt.xlabel("Resting BP", fontsize = 15)
plt.ylabel("Max Pulse", fontsize = 15)
plt.title("Max Pulse vs Resting BP", fontsize = 20)
```

- Relationship plot demonstrates the variation of max pulse (thalach) with respect to resting bp. size of circle: cholesterol
- This graph shows how the maximum pulse varies with resting blood pressure and what the cholesterol value is for each data point.
- Even with lower resting blood pressure, the maximum pulse is very high, and the majority of people in this group have heart disease.
- People can have heart disease even when their resting blood pressure is low and their maximum pulse is high.

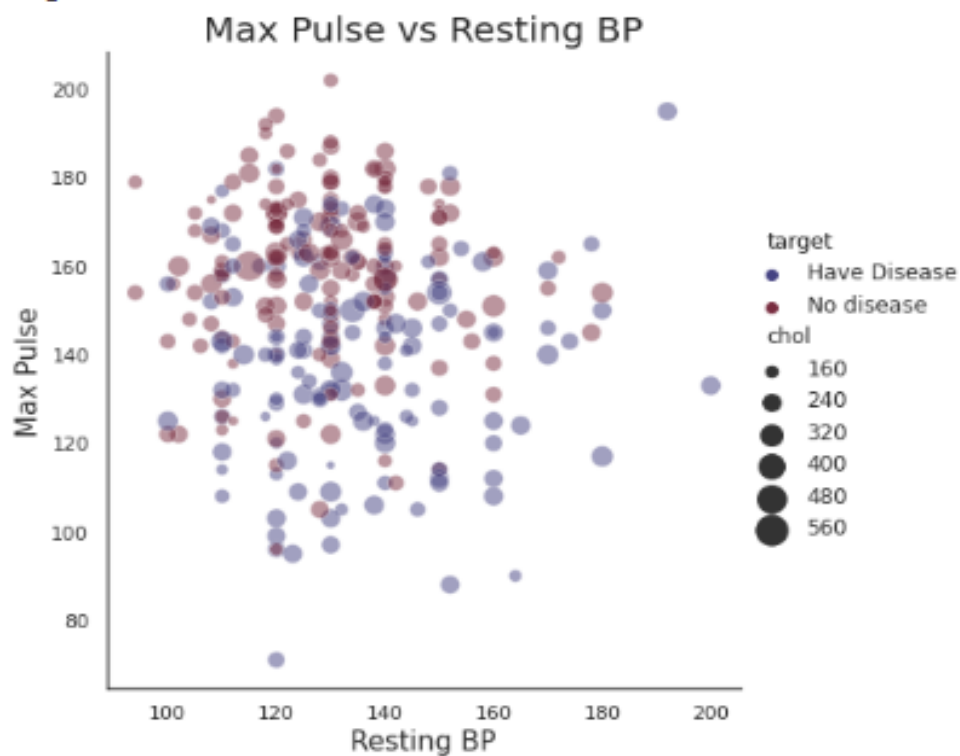


FIGURE 1. 36 RELATIONSHIP PLOT

3.5.0 SCATTER PLOT AND PAIR PLOT

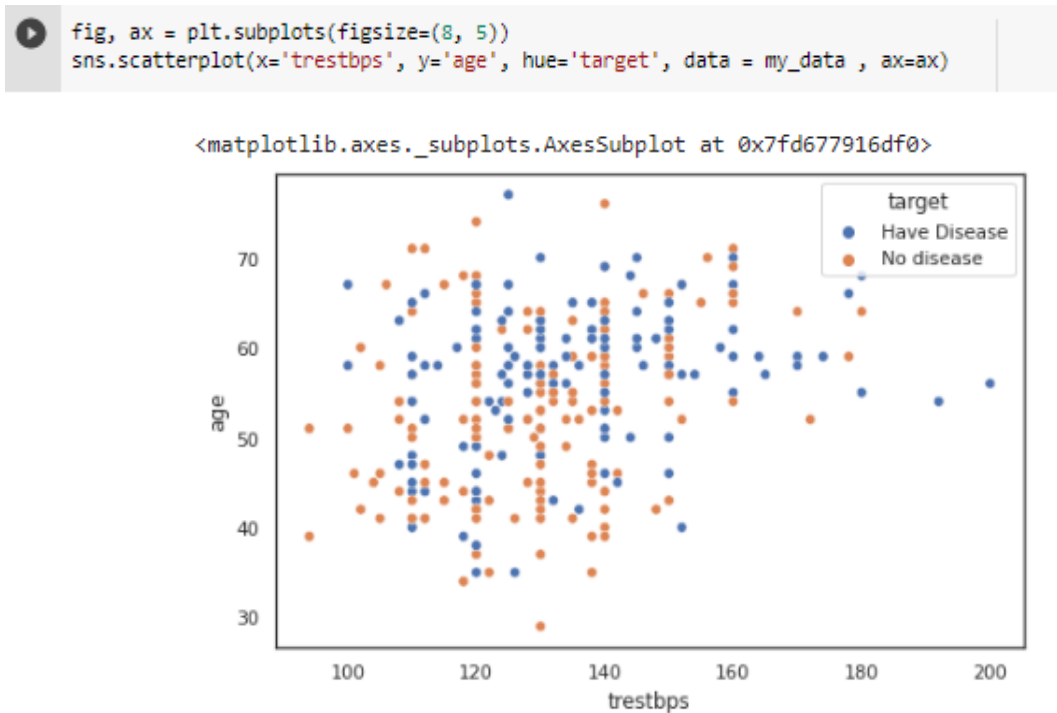


FIGURE 1. 37 SCATTER PLOT

This graph shows that people between the ages of 50 and 60 who have high resting blood pressure are more likely to have heart disease.

PAIR PLOT:

- To Plot pairwise relationships in a dataset
- Sns pair plot to visualize the distribution.
- oldpeak having a linear separation relation between disease and non-disease.
- thalach having a mild separation relation between disease and non-disease.
- Other features don't form any clear separation

```
sns.pairplot(my_data, hue = 'target')
```

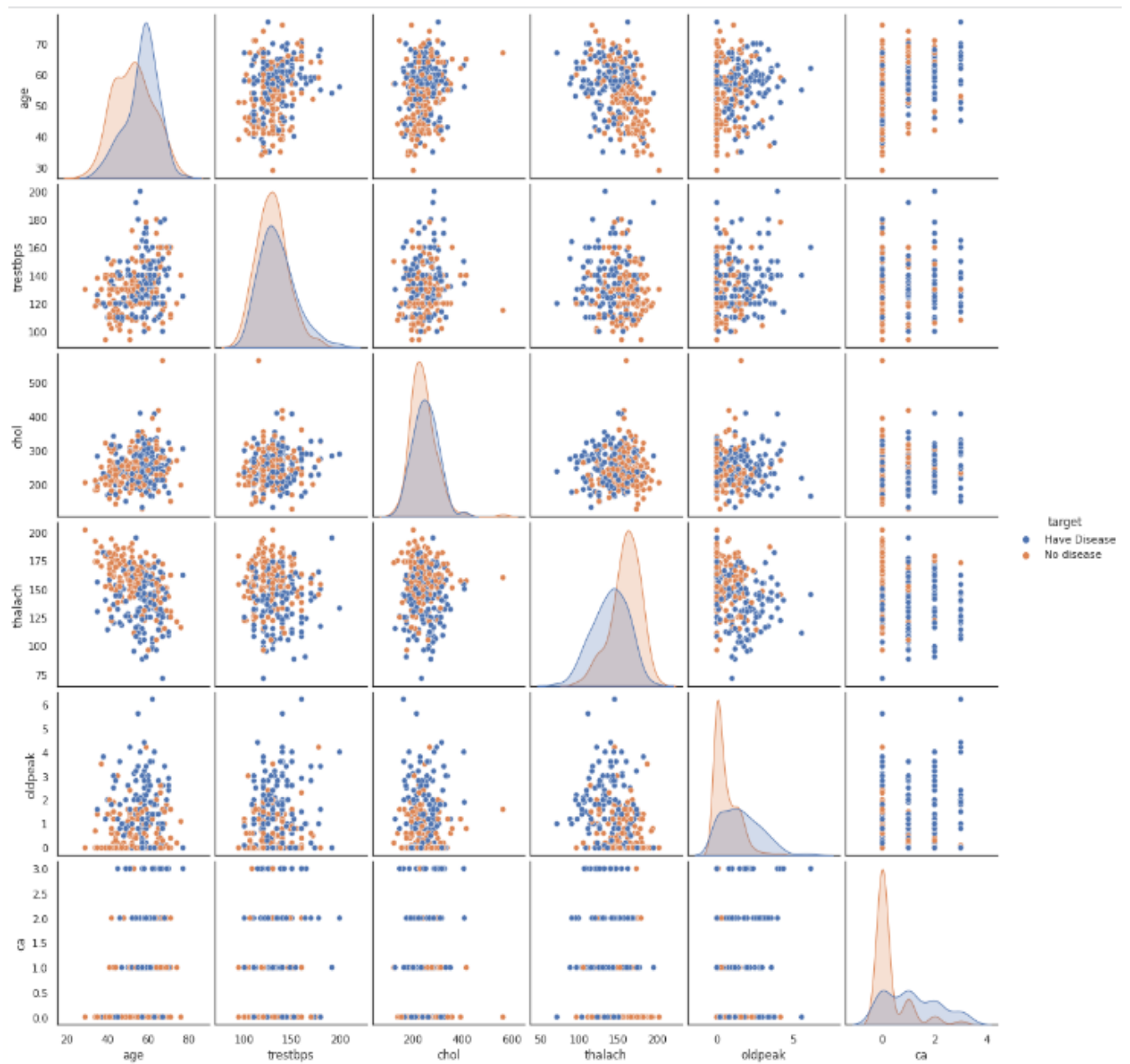


FIGURE 1. 38 SNS PAIR PLOT

3.5 PREPARING DATA FOR MODEL

3.5.1 DATA SCALING

Scaling helps to transform the so that it fits within a specific scale, like 0-100 or 0-1. This helps the algorithm to learn the values easily.

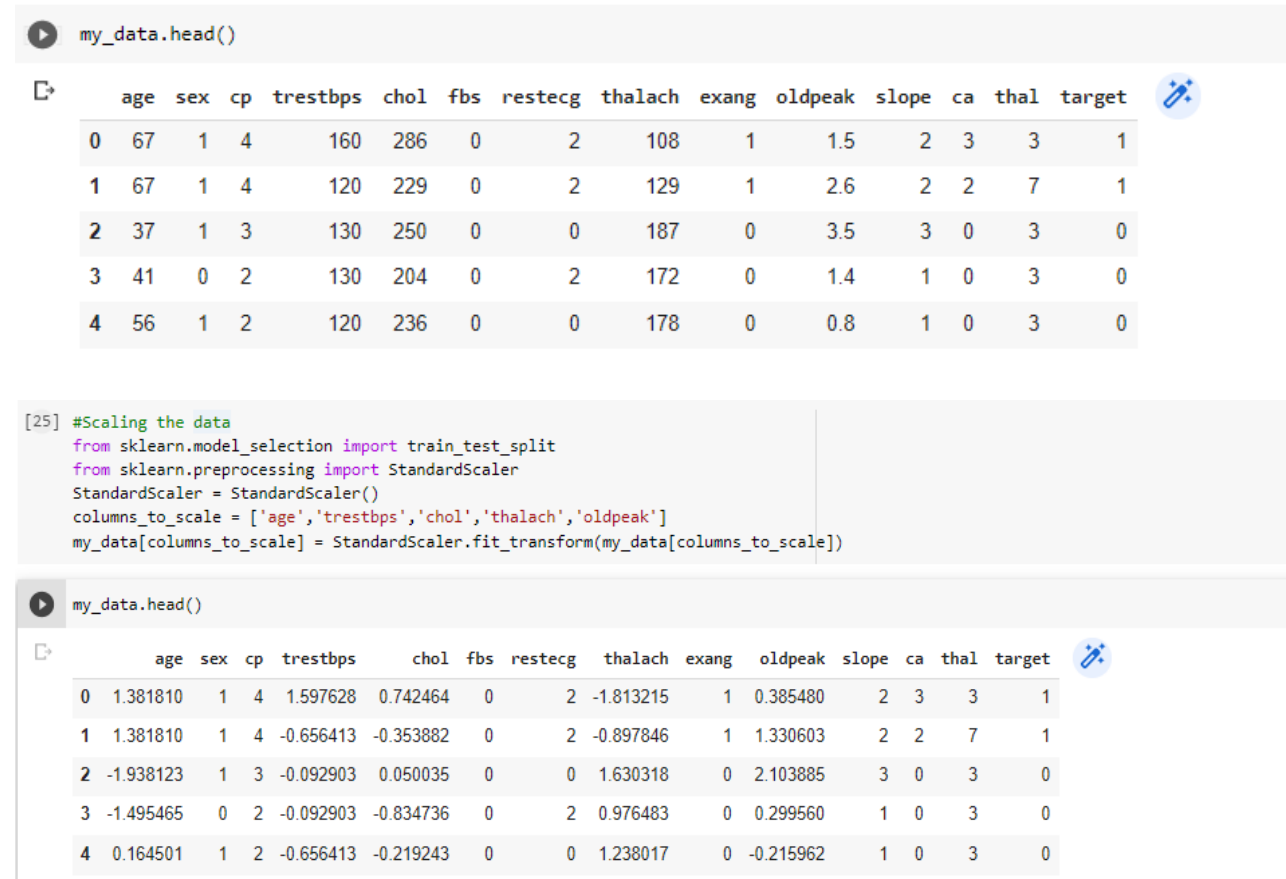


FIGURE 1. 39 DATA SCALING

3.5.2 DATA SPLIT AND TRAINING

- To train the data, I have split it into test set and training set.
- The algorithm will be trained to predict the target attribute.
- X will be training set, so we need to drop target attribute and keep all other attributes
- Y need to have target
- Preferred size is given for the test data.


```
[27] X= my_data.drop(['target'], axis=1)
      y= my_data['target']

[28] #To split data
      X_train, X_test,y_train, y_test = train_test_split(X,y,test_size=0.4,random_state=40)

▶ #We need to check the size of the training and testing dataset
  print('X_train :', X_train.size)
  print('X_test :',X_test.size)
  print('y_train :', y_train.size)
  print('y_test :', y_test.size)

X_train : 2301
X_test : 1547
y_train : 177
y_test : 119
```

FIGURE 1. 40 DATA SPLIT AND TRAIN

3.5.3 LOGISTIC REGRESSION

First I have built a logistic regression model. Since logistic regression identifies an equation that forecasts a result for a binary variable, Y, from one or more response variables, X, it is statistically similar to linear regression. Contrary to linear regression, the model does not strictly require continuous data, hence the response variables can be categorical or continuous.

```
[32] from sklearn.linear_model import LogisticRegression
      lr=LogisticRegression()

      Logistic_model = lr.fit(X_train,y_train)
      prediction = Logistic_model.predict(X_test)
```

I have created an object named logistic regression and assigned it to same name. Fitted the model into X_train and y_train.

- To predict on X_test: 1547

Confusion matrix:

To check accuracy, I have predicted it on y_test: 119 using a confusion matrix.

- True positive: 58 values are predicted correctly by the algorithm
- False positive: 11 values are incorrectly identified by the algorithm
- False negative: 6 values are incorrectly rejecting for certain class
- True negative: 44 certain cases are correctly rejected for certain class



FIGURE 1. 41 CONFUSION MATRIX

```
TruePositive = cm[0][0]
TrueNegative = cm[1][1]
FalseNegative = cm[1][0]
FalsePositive = cm[0][1]
print('Testing Accuracy:', (TruePositive+TrueNegative)/(TruePositive+TrueNegative+FalseNegative+FalsePositive))
```

Testing Accuracy: 0.8571428571428571

```
[37] from sklearn.metrics import accuracy_score
accuracy_score(y_test,prediction)
```

0.8571428571428571

FIGURE 1. 42 ACCURACY TEST

The accuracy calculated here is approx. 85%.

From sklearn we can import accuracy score and test on `y_test` we can make the predictions. We get both the value as same

```
from sklearn.metrics import classification_report
print(classification_report(y_test, prediction))
```

	precision	recall	f1-score	support
0	0.91	0.84	0.87	69
1	0.80	0.88	0.84	50
accuracy			0.86	119
macro avg	0.85	0.86	0.86	119
weighted avg	0.86	0.86	0.86	119

FIGURE 1. 43 CLASSIFICATION REPORT

From sklearn I have imported classification report to check on this:

In the Classification report,

- Accuracy for 0 (No disease) precision is 91% correctly predicted.
- Accuracy for 1 (Have disease) precision is 80% correctly predicted.

3.5.4 K-NEAREST NEIGHBORS ALGORITHM

The k-nearest neighbor's algorithm, sometimes referred to as KNN or k-NN, is a non-parametric, supervised learning classifier that relies on closeness to produce classifications or predictions about the grouping of a single data point.

```
from sklearn.neighbors import KNeighborsClassifier

KNN = KNeighborsClassifier()
model_2 = KNN.fit(X_train, y_train)
prediction_2 = model_2.predict(X_test)
cm_2 = confusion_matrix(y_test, prediction_2)
print(cm_2)
```

```
[[53 16]
 [ 5 45]]
```

```
print('Logistic Regression :', accuracy_score(y_test, prediction))
print('KNN :', accuracy_score(y_test, prediction_2))
```

```
Logistic Regression : 0.8571428571428571
KNN : 0.8235294117647058
```

FIGURE 1. 44 KNN

The best accuracy is obtained for logistic regression compared with KNN.

3.5.5 CLUSTERING

The K-means clustering algorithm groups data into K clusters, based on each point's distance from the center of the cluster, which is called the centroid.

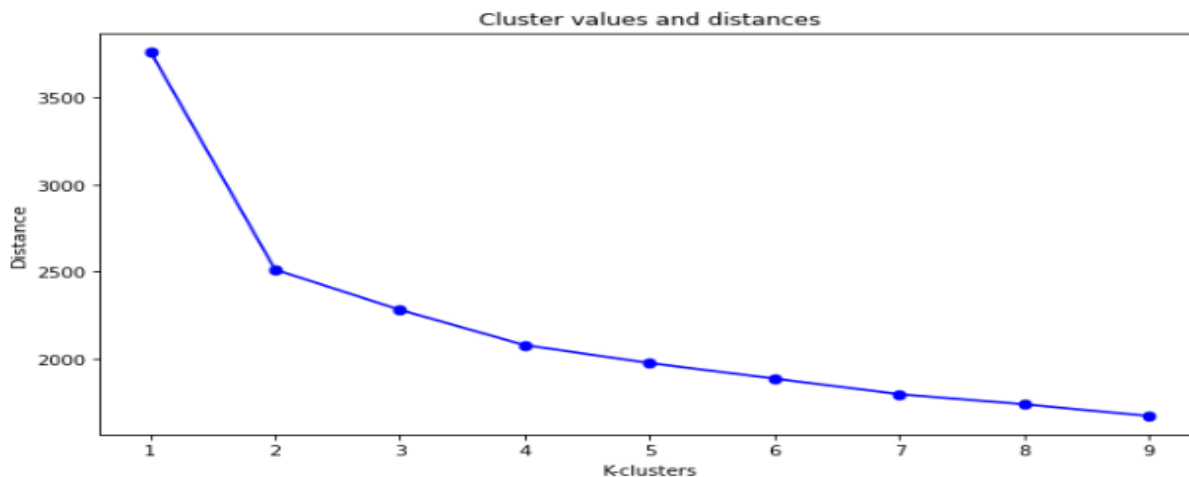
```

▶ from sklearn.cluster import KMeans
  from sklearn import preprocessing
  distances = []

  K = range(1,10)
  for k in K:
      ClusterInfo = kmeanmodel = KMeans(n_clusters=k).fit(my_data)
      distances.append(ClusterInfo.inertia_)

  plt.figure(figsize=(10, 5), dpi=80)
  plt.plot(K,distances, 'bo-')
  plt.xlabel('K-clusters')
  plt.ylabel('Distance')
  plt.title('Cluster values and distances')
  plt.show()

```



```

[43] kmeans = KMeans(n_clusters=3).fit(my_data)
      centroids = kmeans.cluster_centers_
      print(centroids)

```

```

[[ 0.80282638  0.43478261  3.02898551  0.15945157  0.27053057  0.20289855
  1.39130435 -0.39372909  0.30434783 -0.06279954  1.65217391  0.84057971
  3.         0.44927536]
 [ 0.14507154  0.8778626   3.48091603  0.16433296  0.00862997  0.16793893
  1.00763359 -0.33617941  0.50381679  0.39466244  1.79389313  0.96183206
  6.8778626   0.76335878]
 [-0.77499366  0.57291667  2.83333333 -0.33885184 -0.20622016  0.0625
  0.6875       0.7417376   0.10416667 -0.49341262  1.29166667  0.17708333
  3.03125      0.0625    ]]

```

FIGURE 1. 45 KMEANS CLUSTERING

Using a scatter plot, the clusters for the data points using “Chol” for your x-axis and “thalach” for your y-axis are plotted along with centroids. The cluster centroids help us to determine the three different clusters obtained.

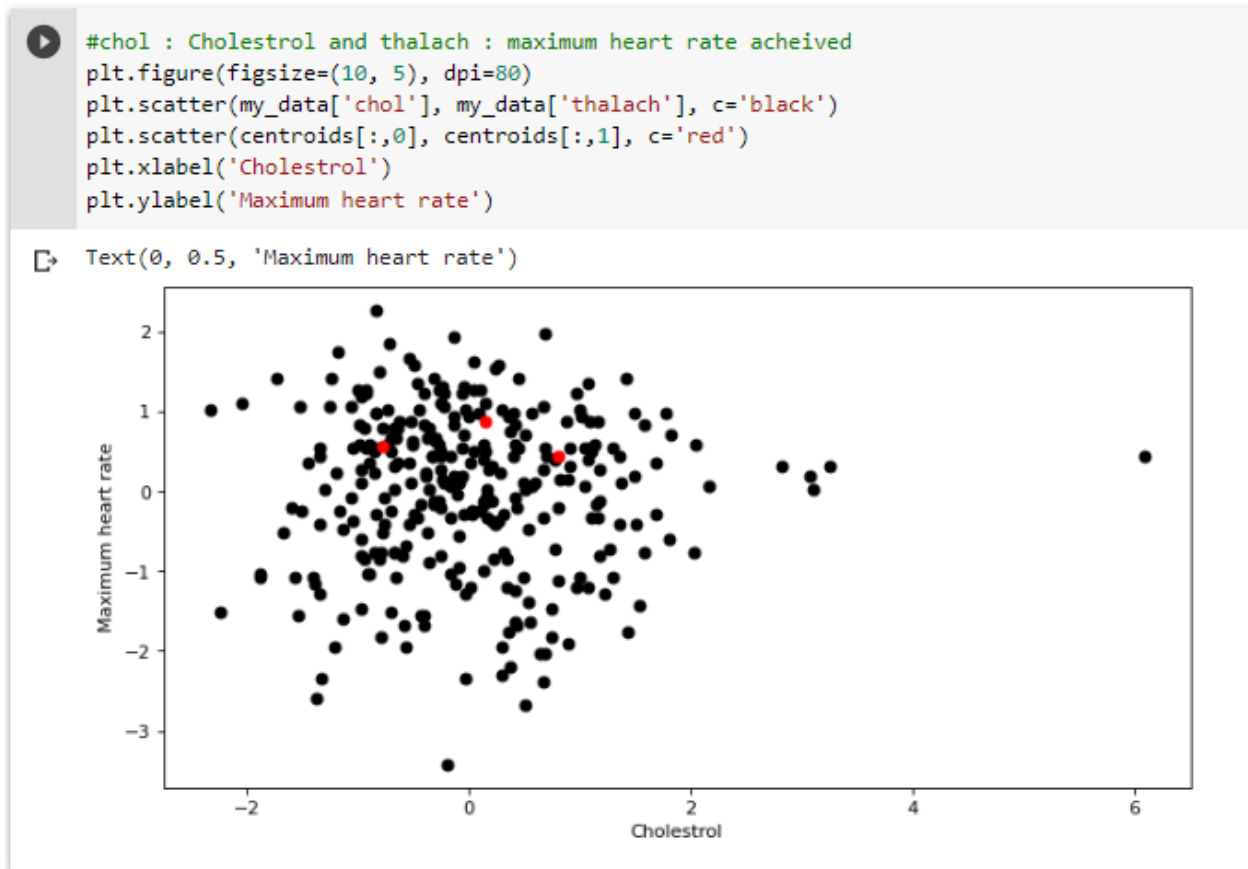


FIGURE 1. 46 KMEANS CLUSTERING

3.6 NLP ANALYSIS

For the NLP analysis, I have only considered the three attributes which had text values. (sex, target, cp)

Imported the necessary libraries.

```
[76] # import needed libraries
# needed for directory access
import os
import nltk
textfile = data_for_NLP_analysis.to_csv(index = False)
print(textfile)

from nltk.tokenize import RegexpTokenizer
tokenizer = RegexpTokenizer(r'\w+')
tokens = tokenizer.tokenize(textfile.lower())
print(tokens)

['sex', 'cp', 'target', 'male', 'asymtomatic', 'disease', 'male', 'asymtomatic', 'disease', 'male', 'non', 'anginal', 'pain', 'no_disease', 'female', 'atypical_angina', 'no_disease']
```

FIGURE 1. 47 IMPORTING LIBRARIES : NLP ANALYSIS

```
[81] word_frequency_allwords = nltk.FreqDist(tokens)
print(word_frequency_allwords.tabulate(10))
```

```

      male    no_disease  asymptomatic    disease    female    non    anginal    pain atypical_angina  typical_angina
None      200         159         142         137         96      83      83         83         49         22

```

```
[82] word_frequency_allwords['female']
```

```
96
```

```
[83] word_frequency_allwords['male']
```

```
200
```

```

plt.figure(figsize=(8, 5), dpi=80)
print(word_frequency_allwords.tabulate(10))
word_frequency_allwords.plot(10)

```

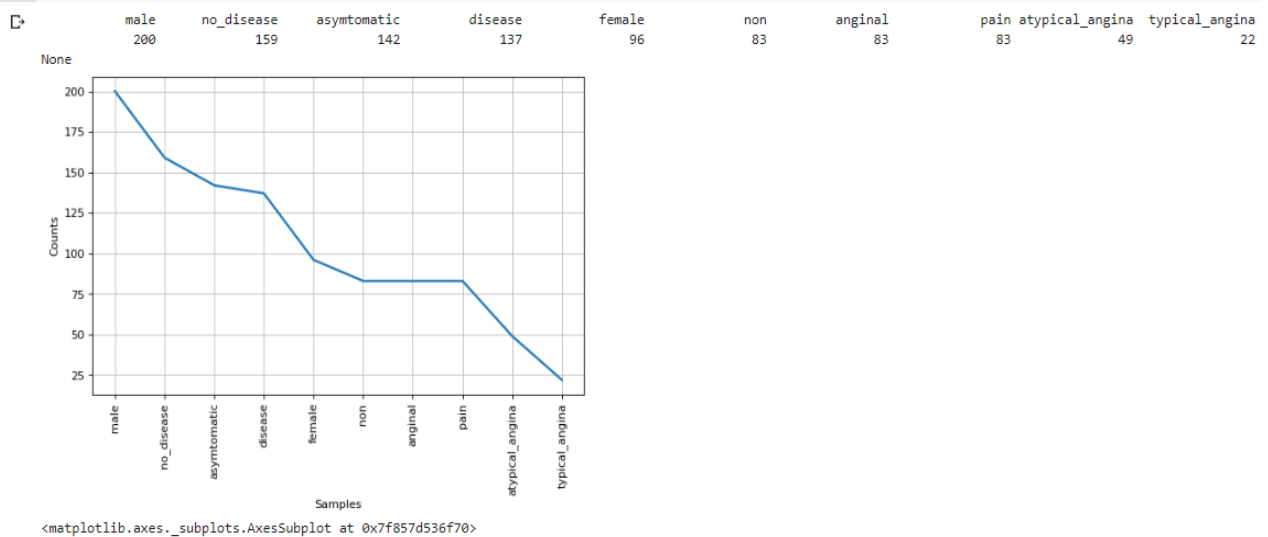


FIGURE 1. 48 NLP ANALYSIS : FREQUENCY COUNT

This graph aids in determining the frequency of words. We can also see that the dataset has a higher proportion of males than females. The dataset consists of fewer individuals with heart disease.

4. ANALYSIS OF RESULT

- To prevent inaccurate data in the dataset, data preprocessing is done in the initial phase.
- Analysis was carried out based on the research questions.
- The key attributes of the dataset in connection to the target were identified by a correlation feature-based selection.
- The logistic regression model accurately predicted the target in compared to the KNN.
- K-means clustering was plotted based on each point's distance from the center of the cluster.
- The frequency counts of the words were subjected to NLP analysis, and a graph was created.

5. CONCLUSION

5.1 SUMMARIZE FINDINGS OF ANALYSIS

The findings that have been noticed in the dataset are as follows:

- Males outnumber females in number.
- In this dataset, there are more individuals without heart disease than those who do.
- The data types are two: continuous (age, trestbps, chol, thalach, old peak) and categorical (sex, fbs, exang, target, cp, restecg, slope, ca, thal).
- The majority of the patients are in their fifties and sixties. The mean age is about 54 years; the youngest is 29 and the oldest is 77.
- According to the correlation matrix, the target is positively correlated with cp, exang, old peak, ca, thal, and slope. The target and Thalach have a significant inverse relationship. The target and age, sex, trestbps, chol, fbs, and restecg have little in common.
- The dataset makes it clear that people with heart disease are the ones who feel pain following arduous activity.
- There are more individuals with asymptomatic chest pain in the dataset.
- Age-related increases in resting blood pressure are seen in this sample, and these increases are particularly pronounced in women.
- People's cholesterol levels increase as they age. Women are more likely than men to be impacted.
- KNN has an around 82% accuracy compared to a roughly 85% accuracy for logistic regression.

5.2 LIMITATIONS

Due to the dataset's limited size and higher proportion of healthy individuals than heart disease sufferers, I was unable to perform additional analysis on the persons with heart disease.

5.3 FUTURE SCOPE

The dataset size can be expanded. The evaluation findings can be improved once more by using machine learning and numerous other optimization approaches. The data can be normalized in a variety of other ways, and the results can be compared. Additionally, there may be additional ways to incorporate heart-disease-trained ML and DL models with specific multimedia for the benefit of patients and physicians.

5.4 CODE SNIPPETS

Import all Libraries

```
import os

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import missingno as msno
```

#Import the file

```
from google.colab import files

upload = files.upload()
```

#Create a dataframe

```
import pandas as pd

data = pd.read_csv('processed.cleveland.data', sep=",")

#Print the first 5 rows of the dataset

data.head()
```

#The csv file did not have a header, so assigned a headerlist

```
headerList = ['age', 'sex',
'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'oldpeak', 'slope', 'c
a', 'thal', 'target']

data.to_csv("processed.cleveland.data.csv", header=headerList, index=False)

data = pd.read_csv("processed.cleveland.data.csv")

print('\nModified file:')

data.head()
```

```
print(data.shape)
```

```
data.info()
```

```
data.dtypes
```



```

#checking for any null values

data.isnull().sum()

#CHECKING FOR UNIQUE VALUES

for col in data.columns:
    print(col, data[col].unique())

spec_chars = ["!", "'", "#", "%", "&", "!", "(", ")", "*", "+", ",", "-",
               ".", "/", ":", ";", "<", "=", ">", "?", "@", "[", "\\", "]", "^", "_", "`", "{", "|", "}", "~",
               "-", "-"]

for char in spec_chars:
    data = data.replace(char, np.nan)

print(data)

msno.matrix(data)

print('DataFrame after dropping the rows having missing values:')

my_data = data.dropna(axis=0)

print(my_data)

my_data[['age',
          'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach', 'exang', 'slope']] =
my_data[['age', 'sex', 'cp', 'trestbps', 'chol', 'fbs', 'restecg', 'thalach',
          'exang', 'slope']].astype(int)

my_data[['ca', 'thal']] = my_data[['ca', 'thal']].astype(float).astype(int)

#checking for any data duplicates

duplicates = my_data.duplicated().sum()

if duplicates:
    print("The Dupliterated rows available in the dataset")
else:
    print("There are no duplicates in the dataset")

```

```

my_data.describe()

#changing 2,3,4 as 1 in target column
# 0: no disease 1 : have disease
my_data.target = my_data.target.replace([2,3,4],1)
my_data['target'].unique()

my_data.corr()
sns.set(style="white")
plt.rcParams['figure.figsize'] = (15, 10)
sns.heatmap(my_data.corr(), annot = True, linewidths=.5, cmap="gist_gray_r")
plt.title('CORRELATION BETWEEN VARIABLES', fontsize = 15)

sns.pairplot(my_data, hue = 'target')

#FOR BETTER VISUALIZATION
my_data['sex'] = my_data.sex.replace({1: "Male", 0: "Female"})
my_data['cp'] = my_data.cp.replace({1: "typical_angina", 2:
"atypical_angina", 3:"non-angina pain", 4: "asymtomatic"})
my_data['fbs'] = my_data.fbs.replace({1: "True", 0: "False"})
my_data['restecg'] = my_data.restecg.replace({0: "normal", 1: "ST-T wave",
2:"Left ventricular hypertrophy"})
my_data['exang'] = my_data.exang.replace({1: "Yes", 0: "No"})
my_data['slope'] = my_data.slope.replace({1: "upsloping", 2: "flat",
3:"downsloping"})
my_data['thal'] = my_data.thal.replace({6: "fixed_defect", 7:
"reversible_defect", 3:"normal"})
my_data['target'] = my_data.target.replace({1: "Have Disease", 0: "No
disease"})

my_data['target'].value_counts()

```

```

fig, ax = plt.subplots(figsize=(10,6))

y = my_data['target'].value_counts().plot.pie(x= "Heart Disease", y = "No of
Patients", labels = ["Have Disease", "No Disease"],startangle = 60, colors =
["indigo","grey"] , ax=ax, explode = [0.1, 0], shadow = True)

ax.set(title = "Diagnosis of Heart disease")

plt.show()

```

```

fig, ax = plt.subplots(figsize=(8,6))

#my_data['sex'] = my_data.sex.replace({1: "Male", 0: "Female"})

Target_analysis = my_data.target.value_counts()

Target_analysis.plot.bar(title="Diagnosis of Heart disease", ylabel='Count',
color = ['black','grey'], edgecolor = ['grey','black'], ax=ax)

totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:
    ax.text(i.get_x()+.09, i.get_height()-50,
str(round((i.get_height()/total)*100, 2))+'%', fontsize=14, color='white',
weight = 'bold')

plt.tight_layout()

```

```

fig, ax = plt.subplots(figsize=(8, 7))

p = sns.countplot(x="sex", data = my_data, hue='target', palette='BuPu', ax =
ax)

plt.xlabel('Gender', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("GENDER ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)

totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:

```

```

    ax.text(i.get_x()+.05, i.get_height()-15,
str(round((i.get_height()/total)*100, 2))+'%', fontsize=14, color='white',
weight = 'bold')

plt.tight_layout()

gender = pd.crosstab(my_data['sex'], my_data['target'])

print(gender)

gender.plot(kind = 'bar', stacked = True, color = ['indigo', 'grey'],
figsize=(8,8), grid = False)

plt.text(0.2,150,'Male : 1')

plt.text(0.2,140,'Female : 0')

fig, ax = plt.subplots(figsize=(19, 7))

p = sns.countplot(x="age", data = my_data, hue='target', palette='icefire',
ax = ax)

plt.xlabel('Age', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("AGE ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)

min_age = min(my_data['age'])
max_age = max(my_data['age'])
mean_of_age = my_data.age.mean()

fig, ax = plt.subplots(figsize=(10, 7))

sns.barplot(x = my_data.age.value_counts()[10].index,y =
my_data.age.value_counts()[10].values,palette = 'BuPu', ax =ax)

plt.xlabel('Age')

plt.ylabel('Age distribution')

print("The minimum age in the dataset is :", min_age)

print("The maximum age in the dataset is :", max_age)

print("The mean of the age in the dataset is :", mean_of_age)

fig, ax = plt.subplots(figsize=(8, 6))

```

```
p = sns.countplot(x="exang", data = my_data, hue='target', palette='icefire',
ax = ax)

plt.xlabel('exang', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("EXERCISE INDUCED ANGINA ANALYSIS BASED ON HAVING HEART DISEASE",
fontsize = 11)
```

```
fig, ax = plt.subplots(figsize=(10, 7))

p = sns.countplot(x="slope", data = my_data, hue='target', palette='BuPu', ax
= ax)

plt.xlabel('slope', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("SLOPE ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)
```

```
fig, ax = plt.subplots(figsize=(20, 6))

p = sns.countplot(x="oldpeak", data = my_data, hue='target', palette='BuPu',
ax = ax)

plt.xlabel('old peak', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("OLD PEAK ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 11)
```

```
fig, ax = plt.subplots(figsize=(10, 7))

p = sns.countplot(x="cp", data = my_data, hue='target', palette='icefire', ax
= ax)

plt.xlabel('Chest pain', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("CHEST PAIN ANALYSIS BASED ON HAVING HEART DISEASE", fontsize = 14)
```

```
my_data.hist(edgecolor = "black", color = 'indigo', figsize=(20,13))
```

```
plt.figure(figsize=(9,8))

p = sns.lineplot(data=my_data, x='age', y='trestbps', err_style="bars",
hue='sex', palette='icefire', legend='full')

plt.xlabel('Age', fontsize = 15)
```

```

plt.ylabel('Resting blood pressure', fontsize = 10)
plt.title("Age vs Resting BP ", fontsize = 10)
plt.show(p)

plt.figure(figsize=(9,8))

r = sns.lineplot(data=my_data, x='age', y='chol', err_style="bars",
hue='sex', palette='icefire',legend='full')

plt.xlabel('Age', fontsize = 15)
plt.ylabel('cholesterol', fontsize = 10)
plt.title("Age vs cholesterol ", fontsize = 10)
plt.show(r)

plt.figure(figsize=(10, 4))

sns.relplot(data = my_data, x='trestbps', y='thalach', hue='target',
size='chol', sizes=(20, 300), alpha=.5, palette="icefire",height=6)

plt.xlabel("Resting BP", fontsize = 15)
plt.ylabel("Max Pulse", fontsize = 15)
plt.title("Max Pulse vs Resting BP", fontsize = 20)

fig, ax = plt.subplots(figsize=(8, 5))
sns.scatterplot(x='trestbps', y='age', hue='target', data = my_data , ax=ax)

sns.pairplot(my_data, hue = 'target')

plt.figure(figsize=(10, 6))

sns.histplot(data = my_data, x = 'restecg', kde = True, hue = 'target',
palette='icefire')

plt.xlabel("Resting ECG", fontsize=20)
plt.ylabel("Count", fontsize=20)
plt.title("Resting ECG WITH RESPECT TO TARGET", fontsize=20)
plt.show()

```

```

plt.figure(figsize=(10, 6))

sns.histplot(data = my_data, x = 'thal', kde = True, hue = 'target',
palette='icefire')

plt.xlabel("Thallium stress", fontsize=20)

plt.ylabel("Count", fontsize=20)

plt.title("THALLIUM STRESS WUTH RESPECT TO TARGET", fontsize=20)

plt.show()


sns.histplot(data = my_data, x = 'ca', kde = True, hue = 'target',
palette='icefire')


fig, ax = plt.subplots(figsize=(10, 7))

p = sns.countplot(x="fbs", data = my_data, hue='target', palette='icefire',
ax = ax)

plt.xlabel('Fasting blood sugar', fontsize = 15)

plt.ylabel('Count', fontsize = 15)

plt.title("FASTING BLOOD SUGAR DISTRIBUTION ACCORDING TO TARGET VARIABLE",
fontsize = 15)

plt.text(1,90,'True: 1')

plt.text(1,85,'False: 0')


totals = []

for i in ax.patches:
    totals.append(i.get_height())

total = sum(totals)

for i in ax.patches:
    ax.text(i.get_x()+.05, i.get_height()-15,
str(round((i.get_height()/total)*100, 2))+'%', fontsize=14, color='white',
weight = 'bold')

plt.tight_layout()


#DATA MODEL:

#Scaling the data

from sklearn.model_selection import train_test_split

```

```

from sklearn.preprocessing import StandardScaler

StandardScaler = StandardScaler()

columns_to_scale = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak']

my_data[columns_to_scale] =
StandardScaler.fit_transform(my_data[columns_to_scale])

X= my_data.drop(['target'], axis=1)
y= my_data['target']

#To split data
X_train, X_test,y_train, y_test =
train_test_split(X,y,test_size=0.4,random_state=40)

#We need to check the size of the training and testing dataset
print('X_train :', X_train.size)
print('X_test :',X_test.size)
print('y_train :', y_train.size)
print('y_test :', y_test.size)

#LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()

#LOGISTIC REGRESSION
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder

lr=LogisticRegression()

# Create a LabelEncoder object
le = LabelEncoder()

```



```

# Iterate through columns of X_train and X_test and transform non-numerical
columns

for column in X_train.columns:

    if X_train[column].dtype == type(object): # Check if the column is of
object type (likely string)

        # Fit the encoder to the column in the training data and transform it
X_train[column] = le.fit_transform(X_train[column])

        # Transform the column in the testing data using the fitted encoder
X_test[column] = le.transform(X_test[column])

Logistic_model = lr.fit(X_train,y_train)
prediction = Logistic_model.predict(X_test)

from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,prediction)
print(cm)
fig, ax = plt.subplots(figsize=(8, 5))
sns.heatmap(cm, annot=True,cmap='Blues', ax = ax)

TruePositive = cm[0][0]
TrueNegative = cm[1][1]
FalseNegative = cm[1][0]
FalsePositive =cm[0][1]

print('Testing
Accuracy:', (TruePositive+TrueNegative)/(TruePositive+TrueNegative+FalseNegati
ve+FalsePositive))

from sklearn.metrics import accuracy_score
accuracy_score(y_test,prediction)

from sklearn.metrics import classification_report
print(classification_report(y_test, prediction))

```

```

#KNN

from sklearn.neighbors import KNeighborsClassifier

KNN = KNeighborsClassifier()
model_2 = KNN.fit(X_train, y_train)
prediction_2 = model_2.predict(X_test)
cm_2 = confusion_matrix(y_test, prediction_2)
print(cm_2)

print('Logistic Regression :', accuracy_score(y_test, prediction))
print('KNN :', accuracy_score(y_test, prediction_2))

#CLUSTERING

from sklearn.cluster import KMeans
from sklearn import preprocessing
import pandas as pd # Import pandas for data manipulation

distances = []

# Assuming 'my_data' is your DataFrame
# Identify categorical columns
categorical_cols = my_data.select_dtypes(include=['object']).columns

# Apply one-hot encoding to categorical features
my_data_encoded = pd.get_dummies(my_data, columns=categorical_cols)

K = range(1,10)
for k in K:
    ClusterInfo = kmeanmodel = KMeans(n_clusters=k).fit(my_data_encoded) # Use
the encoded data for clustering
    distances.append(ClusterInfo.inertia_)

```

```

plt.figure(figsize=(10, 5), dpi=80)
plt.plot(K,distances, 'bo-')
plt.xlabel('K-clusters')
plt.ylabel('Distance')
plt.title('Cluster values and distances')
plt.show()

# Use the encoded data for fitting KMeans
kmeans = KMeans(n_clusters=3).fit(my_data_encoded)
centroids = kmeans.cluster_centers_
print(centroids)

#chol : Cholestrol and thalach : maximum heart rate acheived
plt.figure(figsize=(8, 5), dpi=80)
plt.scatter(my_data['chol'], my_data['thalach'], c='black')
plt.scatter(centroids[:,0], centroids[:,1], c='red')
plt.xlabel('cholesterol')
plt.ylabel('Maximum heart rate')

#NLP ANALYSIS
data_for_NLP_analysis = pd.DataFrame(my_data)
print(data_for_NLP_analysis)

data_for_NLP_analysis['target'] = data_for_NLP_analysis.target.replace({1:
"Disease", 0: "No_disease"})

data_for_NLP_analysis['sex'] = data_for_NLP_analysis.sex.replace({1: "Male",
0: "Female"})

data_for_NLP_analysis['cp'] = data_for_NLP_analysis.cp.replace({1:
"typical_angina",2: "atypical_angina", 3:"non-anginal pain", 4:
"asymtomatic"})

data_for_NLP_analysis['exang'] = data_for_NLP_analysis.exang.replace({1:
"Yes", 0: "No"})

data_for_NLP_analysis['fbs'] = data_for_NLP_analysis.fbs.replace({1: "True",
0: "False"})

```

```

data_for_NLP_analysis['slope'] = data_for_NLP_analysis.slope.replace({1:
"upsloping", 2: "flat",3:"downsloping"})

data_for_NLP_analysis['thal'] = data_for_NLP_analysis.thal.replace({6:
"fixed_defect", 7: "reversable_defect", 3:"normal"})

data_for_NLP_analysis =
data_for_NLP_analysis.drop(['age', 'restecg', 'thalach', 'oldpeak', 'ca' ,
'trestbps', 'chol', 'exang', 'fbs', 'slope', 'thal' ], axis=1)


# import needed libraries
# needed for directory access

import os
import nltk


textfile = data_for_NLP_analysis.to_csv(index=False)
print(textfile)


# import needed libraries
# needed for directory access

import os
import nltk


# Download the Punkt sentence tokenizer
nltk.download('punkt')


textfile = data_for_NLP_analysis.to_csv(index=False)
print(textfile)


# Assuming 'text' is the column in your DataFrame containing the text data
tokens = nltk.word_tokenize(' '.join(data_for_NLP_analysis['cp'])) # Tokenize
the text data

word_frequency_allwords = nltk.FreqDist(tokens)
print(word_frequency_allwords.tabulate(10))

```

```
word_frequency_allwords['female']  
word_frequency_allwords['male']  
plt.figure(figsize=(8, 5), dpi=80)  
print(word_frequency_allwords.tabulate(10))  
word_frequency_allwords.plot(10)
```

6. REFERENCES

❖ Provide a proper citation for the source of the dataset

Hungarian Institute of Cardiology. Budapest: Andras Janosi, M.D., University Hospital, Zurich, Switzerland: William Steinbrunn, M.D, University Hospital, Basel, Switzerland: Matthias Pfisterer, M.D, & V.A. Medical Center, Long Beach and Cleveland Clinic Foundation: Robert Detrano, M.D, Ph.D. (n.d.). *Heart Disease Data set* [Dataset]. UCI Machine Learning Repository. <https://archive.ics.uci.edu/ml/datasets/Heart+Disease>

"Heart Disease Data Set." *UCI Machine Learning Repository: Heart Disease Data Set*, <https://archive.ics.uci.edu/ml/datasets/Heart+Disease>.

❖ Relevant research papers

Prediction of Heart Disease Using a Combination of Machine Learning and Deep Learning

Bharti R, Khamparia A, Shabaz M, Dhiman G, Pande S, Singh P. Prediction of Heart Disease Using a Combination of Machine Learning and Deep Learning. *Comput Intell Neurosci*. 2021 Jul 1;2021:8387680. doi: 10.1155/2021/8387680. PMID: 34306056; PMCID: PMC8266441.

A novel approach for heart disease prediction using strength scores with significant predictors

Yazdani, A., Varathan, K.D., Chiam, Y.K. *et al*. A novel approach for heart disease prediction using strength scores with significant predictors. *BMC Med Inform Decis Mak* 21, 194 (2021). <https://doi.org/10.1186/s12911-021-01527-5>

Improving Heart Disease Prediction Using Feature Selection Approaches

S. Bashir, Z. S. Khan, F. Hassan Khan, A. Anjum and K. Bashir, "Improving Heart Disease Prediction Using Feature Selection Approaches," 2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST), 2019, pp. 619-623, doi: 10.1109/IBCAST.2019.8667106.

❖ Files:

All the presentation documents uploaded on blackboard
 580F20-Research-Report-Outline.docx
 FAQ for Citations
 Project Assignments 1, 2, 3
 CIG-Research-AIT580.docx
 Research-Report-Outline.docx
 Guide-to-Science-Writing.docx