Module Code & Module Title

CU6051NI - Artificial Intelligence

Assessment Weightage & Type

75% Individual Coursework

Year and Semester

2023-24 Autumn

Student Name: Kshitiz Shrestha

London Met ID: 21049529

College ID: NP01CP4A210184

Assignment Due Date: January 17th, 2023

Assignment Submission Date: January 17th, 2023

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Abstract:

This report encapsulates a brief description of a machine learning project ‘Heart Disease Prediction System.’ that leverages three machine learning algorithms K-Nearest Neighbors, Logistic Regression, and Random Forest Classifier. This report consists of different sections which include introductions to relevant topics, problem domain of the project, background, solution part where the entire process of developing this project is delivered, and finally a conclusion section. This report makes it easy to understand the project and topics related to it in depth. This report also consists of the entire process of developing the model with screenshots and a result section where the evaluation scores of all three models are discussed in brief.

Table of Contents:

1. Introduction: 6
 1.1. AI concepts 6
  1.1.1. Artificial Intelligence: 6
  1.1.2. Machine learning: 7
 1.2. Problem domain: 9
2. Background: 11
 2.1. Research work: 11
  2.1.1. AI/ML in healthcare: 11
  2.1.2. AI/ML for heart disease prediction: 12
  2.1.3. Advantages of the topic/problem domain: 13
  2.1.4. Disadvantages of the topic/problem domain: 14
  2.1.5. Dataset used: 15
 2.2. Similar Projects Literature Review: 17
  2.2.1. Research Paper 1: 17
  2.2.2. Research Paper 2: 18
  2.2.3. Research Paper 3: 18
  2.2.4. Research Paper 4: 19
  2.2.5. Research Paper 5: 20
3. Solution: 21
 3.1. Algorithms used: 21
  3.1.1. Logistic Regression: 21
  3.1.2. K-nearest Neighbors: 23
  3.1.3. Random forest: 24
 3.3. Flowchart: 27
 3.4. Development Process: 29
 3.4.1. Tools/Toolkits used: 29
 3.4.2. Entire Process: 31
  3.4.2.1. Importing necessary libraries: 31
  3.4.2.2. Loading the data: 31
  3.4.2.3. Data Exploration: 32
  3.4.2.4. Data analysis and visualization: 32
  3.4.2.5. Data preparation: 51
  3.4.2.6. Model Training: 51
  3.4.2.7. Model Evaluation: 52
  3.4.2.8. Hyperparameter tuning: 55
  3.4.2.9. Visualization of evaluation metrics and comparison between models: 58

3.4.2.10. Making predictions: 64
3.4.3. Results: 67
4. Conclusion: 68
4.1. Analysis of the work done: 68
4.2. How the solution addresses real-world problems: 68
4.3. Limitations: 68
4.4. Future improvements: 68
5. Bibliography: 69

Table of Figures:

Figure 1: Artificial Intelligence Imagination Image 6
Figure 2: Machine learning process image(Source: https://mapendo.co/blog/training-data-the-milestone-of-machine-learning) Accessed: 19 December 2023 7
Figure 3: Global Heart Disease Prevalence Diagram (Source: https://www.bhf.org.uk/-/media/files/for-professionals/research/heart-statistics/bhf-cvd-statistics-global-factsheet.pdf?rev=f323972183254ca0a1043683a9707a01&hash=5AA21565EEE5D85691D37157B31E4AAA) Accessed: 19 December 2023 9
Figure 4: AI in Healthcare (Source: https://www.diagnosio.com/tag/fitness-trackers/) Accessed: 15 January 2023 11
Figure 5: Dataset values. 15
Figure 6: Dataset data types and value counts 16
Figure 7: Sigmoid Curve (Source: https://www.javatpoint.com/logistic-regression-in-machine-learning) Accessed: 19 December 2023 21
Figure 8: Logistic Function Formula (Source: https://medium.com/@rohit_batra/logistic-regression-algorithm-in-just-4-steps-16f34617c03c) 22
Figure 9: K-Nearest Neighbors Graph (Source: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning) Accessed: 19 December 2023 23
Figure 10: Euclidean Distance Formula (Source: https://medium.com/@luigi.fiori.lf0303/distance-metrics-and-k-nearest-neighbor-knn-1b840969c0f4#:~:text=The%20formula%20to%20calculate%20Euclidean,total%20is%20our%20Euclidean%20distance.) 23
Figure 11: Random Forest Classifier. 24
Figure 12: Example of random forest classifier. 25
Figure 13: Flowchart-1 27
Figure 14: Flowchart-2 28
Figure 15: Process screenshot: Importing libraries 31
Figure 16: Process screenshot: Loading data 31
Figure 17: Process screenshot: Data exploration 32
Figure 18: Process Screenshot: Data visualization 1 32
Figure 19: Process Screenshot: Data visualization 2 33
Figure 20: Process Screenshot: Data visualization 3 34
Figure 21: Process Screenshot: Data visualization 4 35

Figure 22: Process Screenshot: Data visualization 5 36
Figure 23: Process Screenshot: Data visualization 6 37
Figure 24: Process Screenshot: Data visualization 7 38
Figure 25: Process Screenshot: Data visualization 8 39
Figure 26: Process Screenshot: Data visualization 9 40
Figure 27: Process Screenshot: Data visualization 10 41
Figure 28: Process Screenshot: Data visualization 11 42
Figure 29: Process Screenshot: Data visualization 12 43
Figure 30: Process Screenshot: Data visualization 13 44
Figure 31: Process Screenshot: Data visualization 14 45
Figure 32: Process Screenshot: Data visualization 15 46
Figure 33: Process Screenshot: Data visualization 16 47
Figure 34: Process Screenshot: Data visualization 17 48
Figure 35: Process Screenshot: Data visualization 18 49
Figure 36: Process Screenshot: Data visualization 19 50
Figure 37: Process Screenshot: Data visualization 20 50
Figure 38: Data preparation 51
Figure 39: Model Training 51
Figure 40: Model Evaluation 52
Figure 41: Evaluation results 1 53
Figure 42: Evaluation results 2 54
Figure 43: KNN parameter tuning 55
Figure 44: Re-evaluation after parameter tuning: KNN 55
Figure 45: Logistic Regression parameter tuning 56
Figure 46: Re-evaluation after parameter tuning: logistic regression...................................56
Figure 47: Random Forest parameters tuning 57
Figure 48: Re-evaluation after parameters tuning: Random Forest.......................................57
Figure 49: Visualization of accuracy comparison of all models 58
Figure 50: Visualization of precision comparison of all models 59
Figure 51: Visualization of Recall comparison of all models 60
Figure 52: Confusion matrix: KNN 61
Figure 53: Confusion matrix: Logistic regression 62
Figure 54: Confusion matrix: Random forest 63
Figure 55: Model prediction 1 64
Figure 56: Model prediction 2 65
Figure 57: Model prediction 3 66

1. Introduction:

1.1. AI concepts

1.1.1. Artificial Intelligence:

Artificial Intelligence as its name suggests is an intelligence that is created artificially using computer systems. It refers to the ability of a digital computer or a computer-controlled robot to perform tasks commonly associated with intelligent beings. (Copeland, 2023) In other words, AI is the ability of computer technology to think, reason, and provide solutions. It is an advanced replica of human intelligence achieved through computer systems. According to IBM, AI in its simplest form is a field, that combines computer science and robust datasets, to enable problem-solving. In today's world, AI technologies are being implemented in various areas such as health diagnosis, driving automation, etc. and its application is rising drastically due to its problem-solving ability. In recent years, AI has become so advanced that some of it can perform tasks that exceed human intelligence itself. Some examples of AI include ChatGPT, Google Translate, etc. AI is a vast field that encompasses other various sub-fields like machine learning, deep learning, and natural language processing.

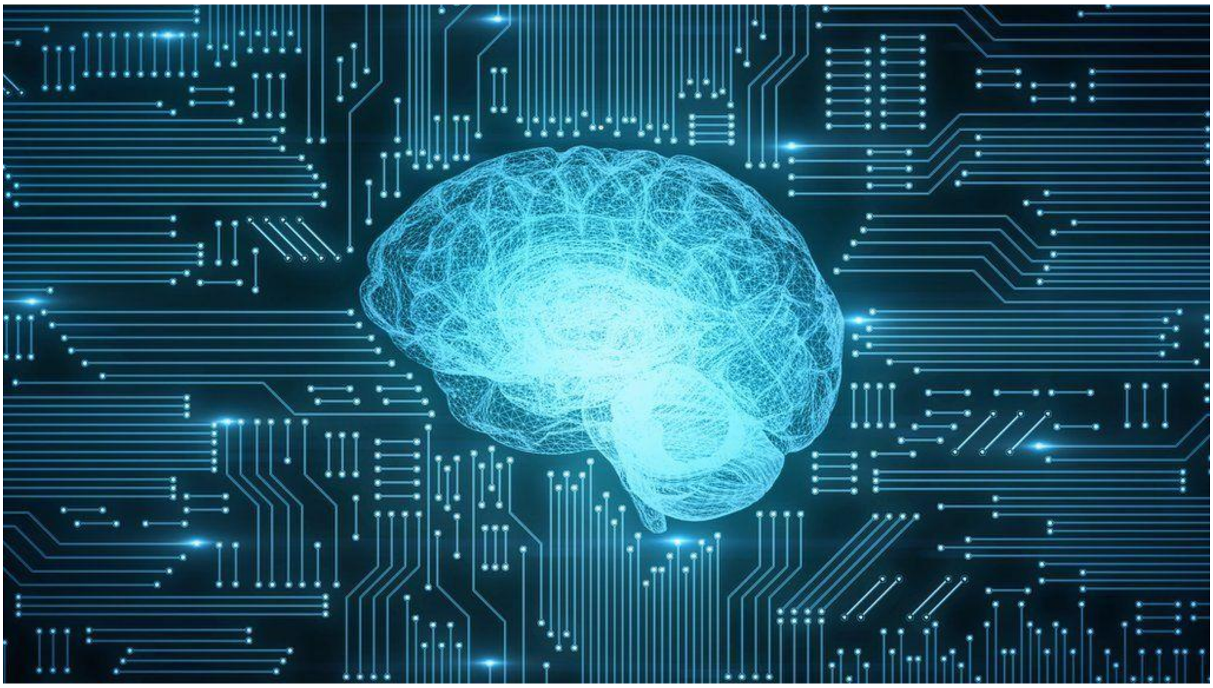


Figure 1: Artificial Intelligence Imagination Image

(Source: https://www.bbc.co.uk/newsround/49274918) Accessed: 19 December 2023

1.1.2. Machine learning:

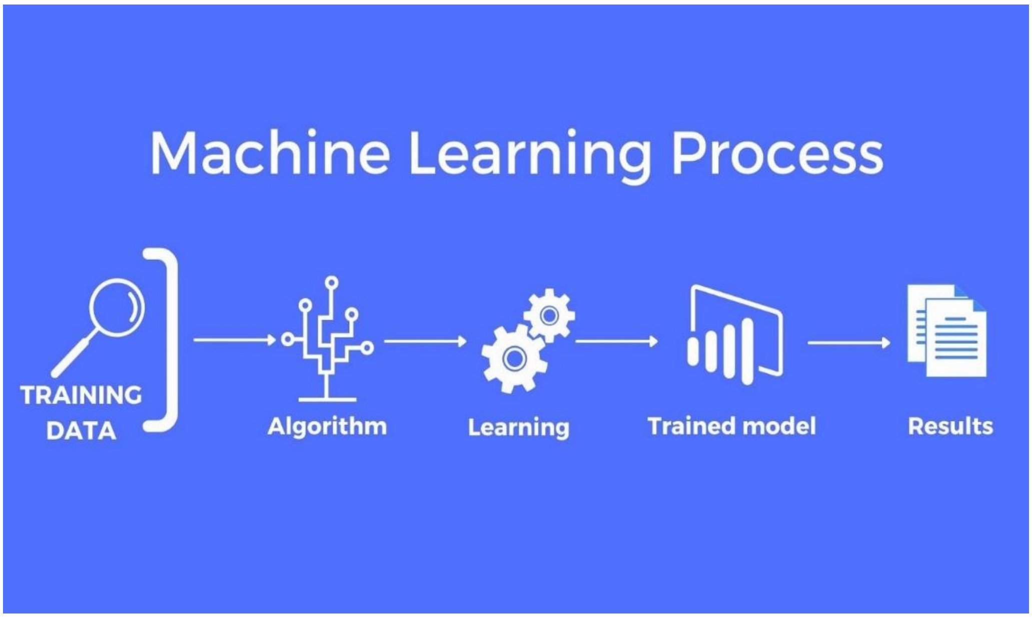


Figure 2: Machine learning process image(Source: https://mapendo.co/blog/training-data-the-milestone-of-machine-learning) Accessed: 19 December 2023

As mentioned above, machine learning is a sub-field of Artificial Intelligence. Machine learning is a branch of AI that focuses on data and algorithms to imitate the way that humans learn. (IBM, 2023) Machine learning is a sub-field of AI that uses datasets, statistical methods, and algorithms to make classifications, predictions, recommendations, or any other decisions. Typically, machine learning is used to uncover key insights from data or identify patterns. There are two main types of machine learning methods they are:

- Supervised: Supervised learning is one of the methods of machine learning that uses labeled datasets to train the algorithm and solve problems. They are mostly used for classification and prediction outcomes. Some examples of supervised learning are Naïve Bayes, logistic regression, etc. Some supervised learning applications are spam email detectors, spam SMS detectors, etc.
- Unsupervised: Unsupervised learning is another method of machine learning that uses algorithms to analyze and cluster unlabeled datasets and solve problems. They are mostly used to discover hidden patterns or data groupings. K-means clustering is one of the examples of this method. Some applications of this unsupervised learning are image classification, customer segmentation, etc.

Building a machine learning system consists of a series of steps by which a working model can predict or classify required predictions or classifications. The entire process of machine learning is usually divided into 7 stages: (Simplilearn, 2023)

- Data collection: Collecting data is the first step in machine learning, here a dataset is either created or sourced from online repositories. Collecting and creating an original dataset takes a lot of time, effort, and bounds to be a bit expensive, so sourcing data is the most viable option. Data collection in machine learning is one of the most crucial steps as the models learn from the data to make predictions, so the dataset should be picked considering various factors like authenticity, reliability, relevance, quality, and so on. The dataset should be sourced from a reliable source for quality data.
- Data preparation: The second step is usually data preparation. In this stage, the collected or sourced data is prepared before training the model with the dataset. This stage consists of steps like cleaning the data, visualizing the data for better understanding, splitting the data, and in some cases scaling or standardizing the data. This step also plays a crucial role as this stage handles data anomalies, missing values, and outliers, cleaning the data and making the data fit for training.
- Choosing a model: This stage consists of choosing a model for the system relevant to the dataset. This is another crucial step as a model relevant to the dataset will give more accurate predictions than a model that is relevant to the dataset.
- Training the model: This stage is the most necessary step in machine learning. In this stage the prepared dataset is passed through a model, basically training the model to recognize patterns so that predictions can be made.
- Evaluation: Another stage in machine learning is model evaluation, here the trained model is tested for its accuracy, precision, and other evaluation metrics so that we can get an overview of how good the model is at doing its job.
- Parameter tuning: This is another stage in machine learning where the chosen algorithm parameters are changed and passed through the model to determine the best set of parameters that gives the model the best accuracy.
- Making predictions: The last stage in machine leaning after model training, evaluation, and model tuning is to make predictions. This step is where the machine learning model predicts an output based on the input it its trained with.

1.2. Problem domain:

Heart disease also known as cardiovascular disease has become quite common among people. According to WHO, cardiovascular disease is responsible for almost 18 million deaths every year and most of these deaths occur prematurely among people under 70 years of age. 157 people are hospitalized due to a heart attack every day, equating to, on average, one person every nine minutes. (Heartfoundation, 2023)

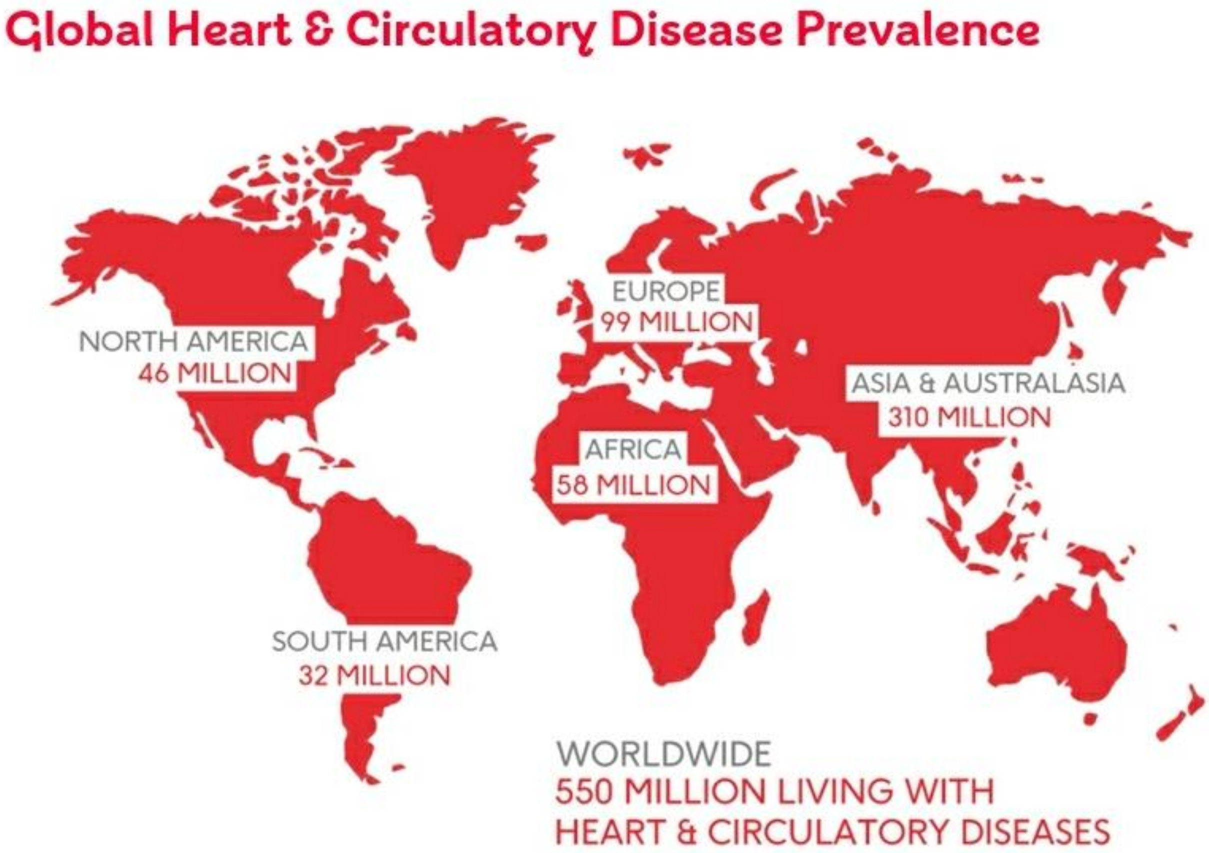

Global Heart & Circulatory Disease Prevalence
NORTH AMERICA 46 MILLION
EUROPE 99 MILLION
ASIA & AUSTRALASIA 310 MILLION
AFRICA 58 MILLION
SOUTH AMERICA 32 MILLION
WORLDWIDE
550 MILLION LIVING WITH HEART & CIRCULATORY DISEASES

Figure 3: Global Heart Disease Prevalence Diagram (Source: https://www.bhf.org.uk/-/media/files/for-professionals/research/heart-statistics/bhf-cvd-statistics-global-factsheet.pdf?rev=f323972183254ca0a1043683a9707a01&hash=5AA21565EEE5D85691D37157B31E4AAA) Accessed: 19 December 2023

Recently in Gujarat, India, more than 1000 people lost their lives due to cardiovascular disease in just six months of which 80% belonged to the age group of 11-25 years old. Witnessing this rise in heart attacks, State Education Minister Kuber Dindor expresses that nearly two lakh school teachers and college professors will be taught CPR. He also added, “The 108-ambulance service receives 173 cardiac emergency calls per day.”

(TheHindu, 2023) Cases like this are experienced all around the world, especially in recent years, as people nowadays are living a very unhealthy life, with no physical activity, junk foods, playing video games all day in a room, unhealthy sleep patterns, smoking, drinking, and so on. So, unless people are motivated to implement a healthy lifestyle or track their heart health, the rate of cardiovascular deaths around the world will continue at this rate or might even increase.

Anybody is at risk when it comes to heart disease. Four out of five people don’t know they have heart failure until their symptoms are severe enough to put them in the emergency room. (Beckerman, 2023) Heart disease is the leading cause of death for several populations. Almost half of the people in the United States are at risk of heart disease, and the numbers are rising. (Donovan, 2023) It is on the rise because of people’s lifestyles these days. Most of them exercise very little with lots of technological advancements, there is less to no physical movement which directly contributes to bad heart health. Similarly, people nowadays smoke a lot, especially teenagers and youths with the introduction of vapes and e-cigarettes. In 2019, close to 2.9 million children started using e-cigarettes, or more than 7,900 per day. This was an increase from more than 2.2 million in 2018 and close to 2.1 million in 2017. This also contributes to cardiovascular complications in many youths these days (Nay, 2023). As scary as it sounds, heart disease can be prevented with regular checkups diagnosis, and lifestyle changes. With early detection and lifestyle changes, heart diseases can be effectively handled or most importantly managed. Heart diseases are globally known as ‘silent killers’ as the symptoms might not manifest until the condition has already advanced. So, identifying risk factors early on is very crucial for preventing any later complications. Regular screening and blood tests help in identifying risk factors early. However, many people are unaware of the risk factors metrics or might know less about them or sometimes the clinical staff might interpret mistakes while deducing heart conditions of a patient, so if a machine learning system is developed that can accurately predict underlying heart conditions using different factors and metrics, it can assist in the healthcare sector for both patients and the healthcare workers or specialists.

2. Background:

2.1. Research work:

2.1.1. AI/ML in healthcare:

Figure 4: AI in Healthcare (Source: https://www.diagnosio.com/tag/fitness-trackers/) Accessed: 15 January 2023

With the advancements of technology, implementing Artificial Intelligence in various sectors has been easy. One of the sectors where trend of artificial intelligence is drastically rising is healthcare sector. AI is dramatically being implemented or at least is being tried in the healthcare sector. AI is already changing the patient experience, how clinicians practice medicine, and how the pharmaceutical industry operates. (Solulab, 2023) AI can also be implemented in developing healthcare robots in the future that can automate normal healthcare activities. Different branches of AI like Machine Learning, Neural Networks, Deep Learning, and such are rising in the healthcare sector for different purposes like Disease prediction, Biomedical data visualizations, improved diagnosis, personalized treatment options, and developed medications (Coursera, 2023) AI and branches can also be implemented for predicting disease outbreaks, maintaining smart health records, drug discovery and manufacturing, etc.

Artificial Intelligence is still in its early phase in the health sector, especially in the diagnostic sector of healthcare, but increasing health datasets for AI makes it possible now to detect some anomalies like cancer, heart disease, diabetes early on. A study in UK was conducted where dataset of mammograms was used to build a ML model, which resulted in 5.7% reduction in false positives and a 9.4% reduction in false negatives. Another study conducted in South Korea where a radiologist and an AI was compared in detecting early-stage breast cancers which resulted in AI outperformed the radiologist achieving a rate of 91% compared to the radiologists' 74%. Another study conducted using Convolutional Neural Networks, demonstrated accurate diagnoses of melanoma cases when compared to dermatologists. Furthermore, researchers have been implementing AI in the detection of diabetic retinopathy, analysis of EKG abnormalities, and prediction of cardiovascular disease risk factors. Deep learning algorithms have also been utilized to identify pneumonia from chest radiography, achieving high sensitivity and specificity compared to radiologists. Another study conducted on acute appendicitis using ML techniques specifically a random forest algorithm achieved high accuracy in diagnosing and predicting the need for surgery. Furthermore, ML algorithms and models are also being implemented in predicting genetics-related traits and potential risks. ML algorithms and models are now making it feasible to predict certain genetic traits ranging from simple eye color to medication allergies or reactions. AI and ML are also being implemented in other various sectors of healthcare like drug discovery, treatment assistance, drug monitoring and optimization, and so on. (BMC Medical Education, 2023)

2.1.2. AI/ML for heart disease prediction:

Disease diagnosis specifically cardiovascular disease detection or prediction is feasible these days because of the availability of high-quality datasets. There are many contributions made towards the sector of heart disease prediction models and many have already been successful. There are already numerous efforts made regarding heart disease predictions like the ASCVD risk estimator which is an online tool developed by the American College of Cardiology (ACC) and the American Heart Association (AHA) that takes various risk factors into account such as age, gender, race, cholesterol levels, blood pressure, diabetes status, and smoking status to assess the likelihood of developing a cardiovascular event. (Cleveland Clinic, 2023)

There are other various approaches in predicting heart disease using multiple machine learning algorithms that perform outstandingly well with high accuracy and precision. A study conducted by Shah et al and other authors in 2020 aimed to develop a cardiovascular disease prediction system using algorithms like naive Bayes, decision tree, random forest, and k-nearest neighbor (KKN) and dataset from the UCI repository with 303 instances resulted in KNN as the highest performing algorithm with 90.8% accuracy. In another study conducted by Drod et al. in 2022, the aim was to utilize machine learning in identifying cardiovascular complications in individuals with metabolic-associated fatty liver disease (MAFLD). ML methods, including a multiple logistic regression classifier, univariate feature ranking, and principal component analysis (PCA), were used in 191 MAFLD patient's dataset to identify individuals with the highest risk of cardiovascular disease This approach achieved great performance by accurately identifying 85.11% of high-risk patients and 79.17% of low-risk patients. According to the findings of the study it was determined that an ML method is useful for detecting MAFLD patients with widespread cardiovascular diseases based on simple patient criteria. (Bhatt, 2023) Similarly, there are lots of other approaches to predicting heart disease prediction with varying accuracies and precisions.

2.1.3. Advantages of the topic/problem domain:

1. Early Intervention and Prevention: This topic helps in contributing towards the Machine Learning approach in predicting heart diseases early on which might help in early intervention and prevention of later complications of patient's heart diseases in the healthcare sector.
2. Personalized Healthcare: Machine Learning models like these can help in personalizing patient's healthcare and risk assessments.
3. Cost Savings: With early detection of heart diseases, both patients and the healthcare facility can benefit from some amount of cost savings by avoiding advanced heart disease treatments.
4. Healthcare Decision-Making: With accurate early predictions with large volumes of data, these types of predictive models can assist both individuals and healthcare professionals in healthcare decision making specifically ones to manage or control heart diseases.

5. Research: This type of models helps in health-related research as it demonstrates relationships between various health factors and diseases and the research can contribute to further advancements in understanding the disease and developing advanced treatments or controlling measures.

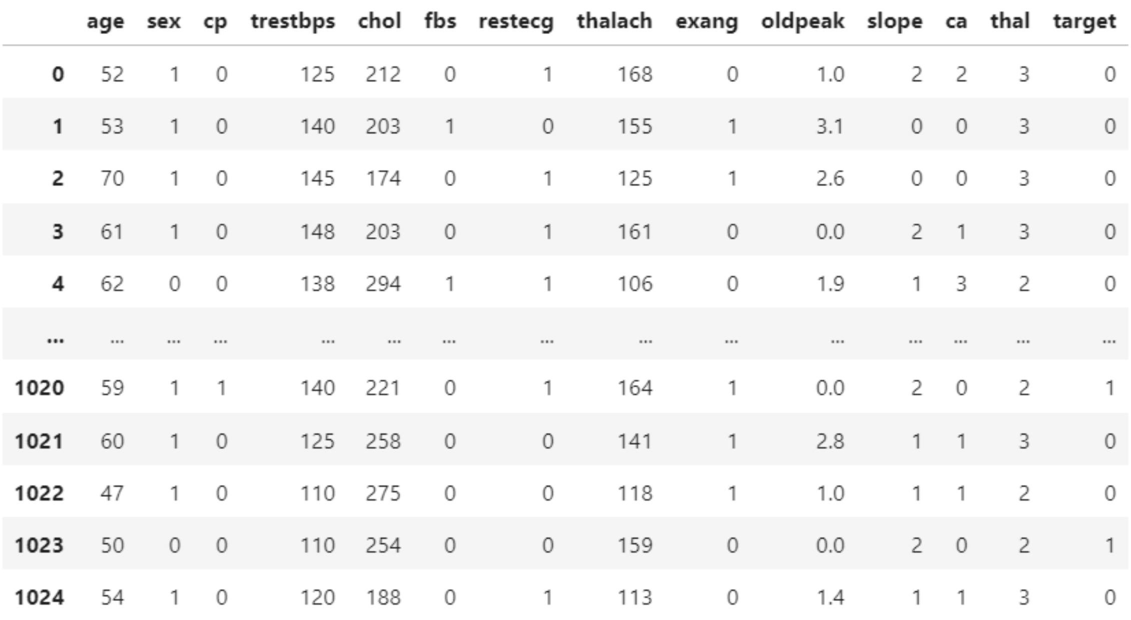
2.1.4. Disadvantages of the topic/problem domain:

1. Data Dependency: The performance and accuracy of the model heavily depends on the dataset used. If the dataset is incomplete or inaccurate, it might lead to less accurate predictions.
2. Patient Trust and Acceptance: These kinds of AI models are hard to get accepted specially when it comes to individuals' health, for actual implementation of these kinds of models in the real world, patient trust might be hard to gain in the beginning phase.
3. Model Validation Challenges: These kinds of models may face challenges such as overfitting and underfitting, which might affect their ability to generalize or adapt to new data. So, regular validation is necessary to ensure the performance of the model.
4. Legal and Liability Issues: Another challenge of these kinds of predictive models is that faulty, mistake or misleading predictions made by the model can lead to legal and liability issues for the healthcare provider especially when it comes to sensitive topics like health.
5. Security Risks: Security issues are another challenge when it comes to machine learning models. If proper cybersecurity measures are not applied, data breaches can happen which will lead to faulty predictions further resulting in patient mistrust or even legal issues.

2.1.5. Dataset used:

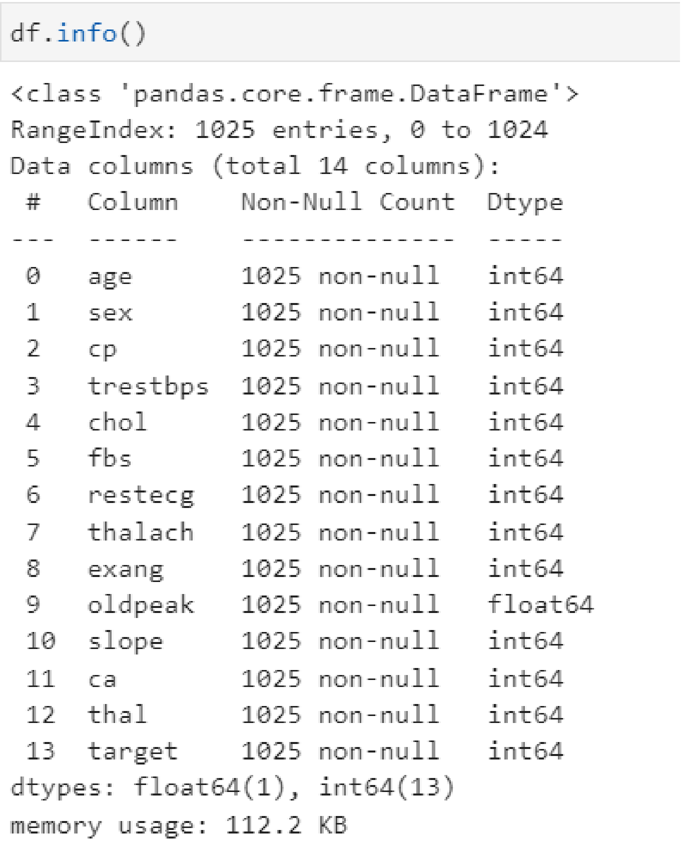
The dataset used in this project is from Kaggle.com, the dataset taken from the website dates from 1998. It consists of four databases: Cleveland, Hungary, Switzerland, and Long Beach V. The data in this study contains 14 columns and 1025 entries. The first variable is age with units in years (age). The second is sex with a value of 1 which means male and a value of 0 means female. The third variable is the variable type of cp which is chest pain. The fourth one is trestbps which indicates resting blood pressure. The fifth one is chol which indicates serum cholesterol in mg/dl. The sixth variable indicates fbs which is fasting blood sugar where 1 = true, 0 = false. The seventh variable indicates restecg which means resting electrocardiographic results. The eight variable indicates thalach which means maximum heart rate achieved. The ninth variable indicates exang which means exercise-induced angina where 1 = yes, 0 = no, and the tenth variable indicates oldpeak which means ST depression induced by exercise relative to rest. The eleventh variable indicates slope which means the slope of the peak training segment ST. The twelfth variable indicates ca which means number of major vessels (0-3) colored by flourosopy. The thirteenth variable indicates thal which has values 1 meaning normal, 2 meaning permanent disability, 3 meaning reversible defects. The last variable indicates target which means the prediction of heart disease made.

Dataset Link: https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset



	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

Figure 5: Dataset values.



```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

Figure 6: Dataset data types and value counts

2.2. Similar Projects Literature Review:

2.2.1. Research Paper 1:

Title: Diabetes Mellitus Prediction Using Supervised Machine Learning Techniques

Authors: Srishti Mahajan, Pradeepta Kumar Sarangi, Ashok Kumar Sahoo, Mukesh Rohra

Published Date: 08 June 2023

DOI: 10.1109/InCACCT57535.2023.10141734

Summary:

This report published by the above-mentioned authors discusses the implementation of machine learning methods for predicting diabetes. They have implemented methods such as logistic regression, and random forest classifier. From the report, we understand that Diabetes is a prevalent condition among most adult people, Diabetes is a condition characterized by abnormal levels of glucose levels in an individual. The study made by the above authors discusses the importance of early diagnosis of diabetes and how it can prevent further complications like blindness, kidney failure, and so on associated with diabetes. The research uses a dataset from Kaggle, named the early-stage diabetes risk prediction dataset. The authors have also mentioned previously achieved projects on a similar topic. They have explored various models like Naïve Bayes, SVM, KNN, Random Forest, and Decision tree with accuracies ranging from 71.37% to 98%. The authors contributed to this research by comparing multiple algorithms and their performance. The performance comparison is assisted by confusion matrices, ROC curves, and so on. The results involve random forest with 99.03% accuracy and logistic regression with an accuracy of 94.23%. Thus, the authors carefully highlighted the importance of early diagnosis of diabetes through these models and how it can have a life-saving impact on diabetic individuals.

2.2.2. Research Paper 2:

Title: Machine Learning-Based Heart Disease Prediction

Authors: Ambika Sekhar, Amrutha Babu, Jayalekshmi V.K., Adithya Udayan

Published Date: 30 March 2023

DOI: 10.1109/ICNGIS54955.2022.10079736

Summary:

This research published by the above-mentioned authors discusses the implementation of machine learning methods for predicting heart diseases using patient datasets. This study implements multiple algorithms such as KNN, Random Forest, Support Vector Machine, and Decision Trees to analyze ECG signals and analyze the risk of heart disease of an individual. The introduction part consists of the dangers of heart disease and the importance of early detection for preventing further complications. The literature review part consists of various research made by the authors on similar topics. The third section focuses on understanding dataset values, cleaning, and preparing data. The fourth section involves a detailed explanation of the algorithms used in the project. The fifth section consists of results that uncover KNN achieving the highest accuracy among the other algorithms used. Finally, the last section is the conclusion section which concludes the study. The project is quite well conducted by the authors as they also have made a web interface to easily perform heart disease prediction.

2.2.3. Research Paper 3:

Title: Heart Disease Prediction Using Logistic Regression Algorithm

Authors: Bhagyesh Randhawan, Ritesh Jagtap, Amruta Bhilawade, Durgesh Chaure

Published Date: 2022-04-25.

DOI Link: https://doi.org/10.22214/ijraset.2022.41860

Summary:

This study conducted by the above-mentioned authors consists of a heart disease prediction system using a Logistic Regression Algorithm. This study consists of an introduction to rising cases of incidences related to the cardiovascular system and the importance of detecting them early. The authors have used the UCI Heart Disease dataset for this project. The methodology section talks about processes like data retrieval, data preparation, prediction, and data validation. In the data analysis section, the authors talk about how different variables or metrics influence the cardiovascular performance of an individual. The study concludes that the algorithm used which is Logistic Regression demonstrates an accuracy of 85% in predicting heart diseases. This study is well conducted by the authors as every little detail has been explained very well. They have explained the importance of detecting heart diseases early, and they have also explained the algorithm itself. They have also conducted data validation using a confusion matrix and K-fold cross-validation.

2.2.4. Research Paper 4:

Title: Heart Disease Prediction Using Logistic Regression Algorithm

Authors: Chintan M. Bhatt, Parth Patel, Tarang Ghetia, Pier Luigi Mazzeo

Published Date: 6 February 2023

DOI Link: https://doi.org/10.3390/a16020088

Summary:

This study conducted by the above-mentioned authors aims to predict the probability of heart disease. The authors have made an approach using machine learning techniques like decision tree, random forest, multilayer perceptron, and XGBoost and a dataset of 70,000 patient records. According to the article, the authors used GridSearchCV for hyperparameter tuning and achieved the best parameters for best performing model which resulted in multilayer perceptron having the best performance out of all with the highest cross-validation accuracy of 87.28%, with recall, precision, F1 score, and AUC scores of 84.85, 88.70, 86.71, and 0.95, respectively. The reaming used classifiers also performed well demonstrating high accuracy with an AUC of above 0.9 for all models. Overall the study was done very well with lots of literature review in the background section and good model building with lots of data visualizations.

2.2.5. Research Paper 5:

Title: Implementation of a Heart Disease Risk Prediction Model Using Machine Learning

Authors: K. Karthick, S. K. Aruna, Ravi Samikannu, Ramya Kuppusammy, Yuvaraja Teekaraman, and Amrush Ramesh Thelkar

Published Date: 02 May 2022

DOI Link: https://doi.org/10.1155/2022/6517716

Summary:

This study conducted by the above-mentioned authors focuses on predicting heart disease using machine learning (ML) techniques on the UCI ML repository's Cleveland HD dataset. The authors have used various machine learning techniques like SVM with RBF kernel, Gaussian Naive Bayes, logistic regression, LightGBM, XGBoost, and random forest, to construct a predictive model. The authors have conducted research on similar projects that indicated algorithms achieving accuracies in the range of 75.58%–87.5% on average. The dataset that is used by the authors has 303 instances with 13 attributes. The authors have performed proper data visualizations which demonstrates the relationship between those attributes. The model is then trained and tested using the above-mentioned machine learning techniques and evaluation is performed at the end by the authors. The results section demonstrates that the random forest algorithm performs the best with an accuracy of 88.5% and a ROC of 0.92. This study showcases valuable insights into the application of machine learning in heart disease prediction systems and highlights the potential of random forest classification for heart disease prediction models.

3. Solution:

3.1. Algorithms used:

The proposed solution for the detection of heart disease involves the use of three algorithms they are:

3.1.1. Logistic Regression:

Logistic regression falls under the supervised learning methods of machine learning. It is a statistical method that accomplishes classification tasks by predicting the probability of an outcome. In simpler terms, this algorithm is a statistical model that predicts the likelihood of an event happening. This algorithm is mostly used in predictive models like heart disease predictions, email spam detection, etc. This algorithm uses a logistic function called the sigmoid function to map predictions. The sigmoid function is an S-shaped curve that converts any real value to a range between 0 and 1. (Kanade, 2022)

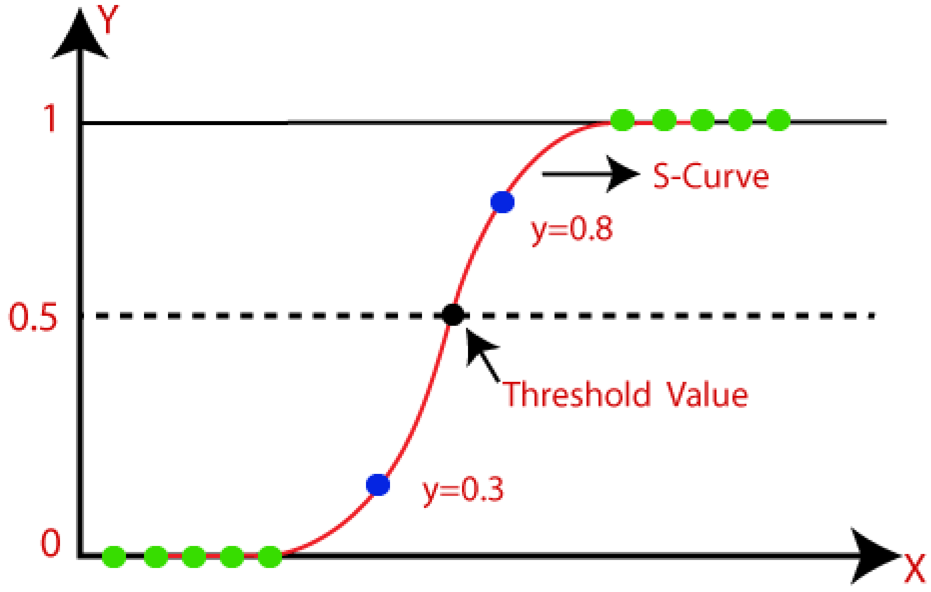


Figure 7: Sigmoid Curve (Source: https://www.javatpoint.com/logistic-regression-in-machine-learning) Accessed: 19 December 2023

The formula for logistic regression:

$$P = \frac{1}{1+e^{-(\beta_0+\beta_1 X_1+\beta_2 X_2+\beta_3 X_3+...+\beta_n X_n)}}$$

Figure 8: Logistic Function Formula (Source: https://medium.com/@rohit_batra/logistic-regression-algorithm-in-just-4-steps-16f34617c03c)

Accessed: 19 December 2023

where,

p = the probability of event happening,

e = base of the natural logarithm,

b_0, b_1, b_2…b_p = coefficients associated with each input feature.

x_1, x_2, x_3,…x_p =

x_1, x_2, x_3,…x_p = values of the input features.

The logistic regression algorithm learns the different values of coefficients in the process of training which then is used to predict the probability of an outcome. If the predicted probability of happening is greater than the threshold (0.5) the value becomes 1 which is predicted to occur, similarly, if the predicted probability is less than the threshold, the value becomes 0 which means that the probability is not predicted to occur.

Advantages of Logistic Regression Algorithm:

- This algorithm performs very well when the dataset is linearly separable.
- This algorithm is easy to implement and very efficient to train.

Disadvantages of Logistic Regression Algorithm:

- This algorithm creates linear boundaries, so, we won't be able to obtain better results when dealing with complex or non-linear data.

  This algorithm may lead to overfitting if the number of observations is lesser than the number of features. (geeksforgeeks, 2023)

3.1.2. K-nearest Neighbors:

K-nearest Neighbors is one of the simplest algorithms in machine learning and is very simple to implement and understand. The KNN algorithm relies on the idea that similar data points tend to have similar values. (Srivastav, 2023) As shown in the image below, the KNN algorithm simply compares a new data point to similar known data points, among these k neighbors, the algorithm counts the number of the data points in each category and assigns the new data point to that category for which the number of the neighbor is maximum. (Javatpoint, 2023)

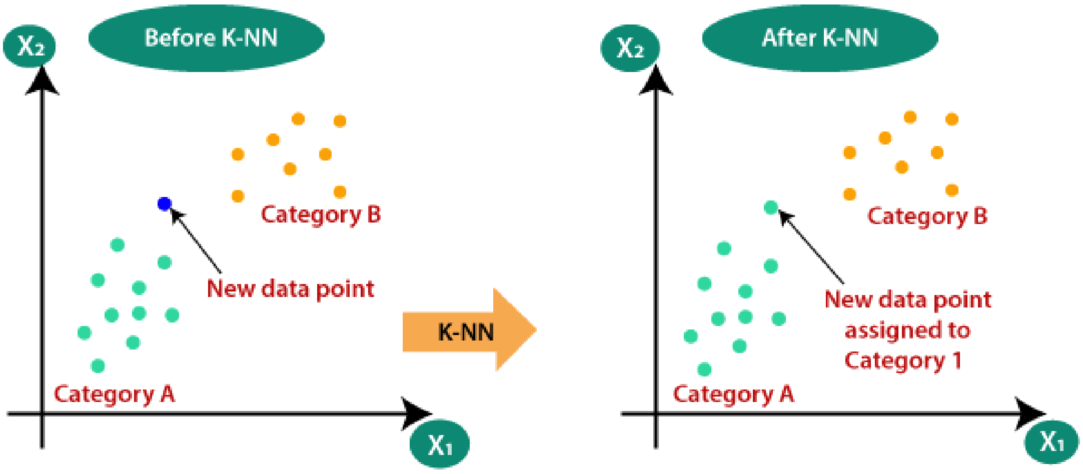


Figure 9: K-Nearest Neighbors Graph (Source: https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning) Accessed: 19 December 2023

Following is the Euclidean distance formula used in this algorithm:

$$d(x, y) = \sum_{i=1}^{n} |x_i - y_i|$$

Figure 10: Euclidean Distance Formula (Source: https://medium.com/@luigi.fiori.lf0303/distance-metrics-and-k-nearest-neighbor-knn-1b840969c0f4#:~:text=The%20formula%20to%20calculate%20Euclidean,total%20is%20our%20Euclidean%20distance.)

Accessed: 19 December 2023

Advantages of the K-nearest Neighbors Algorithm:

- This algorithm is very simple and easy to learn and implement.
- This algorithm has a single hyperparameter which is the value of K, which enables easy hyperparameter tuning.

Disadvantages of the K-nearest Neighbors Algorithm:

- This algorithm doesn't work well with large datasets.
- This algorithm is sensitive to outliers, missing or mislabeled values, as a single outlier, missing or mislabeled value can change the class boundaries. (MLNerds, 2019)

3.1.3. Random forest:

Random forest is a supervised machine learning algorithm that integrates multiple classifiers to solve a complex problem. (Turing, 2023) In simple terms, a random forest algorithm is a mixture of multiple decision trees it takes the average prediction from all the decision trees to make more accurate predictions as shown in the figure below. It is one of the most popular algorithms used for both regression and classification problems.

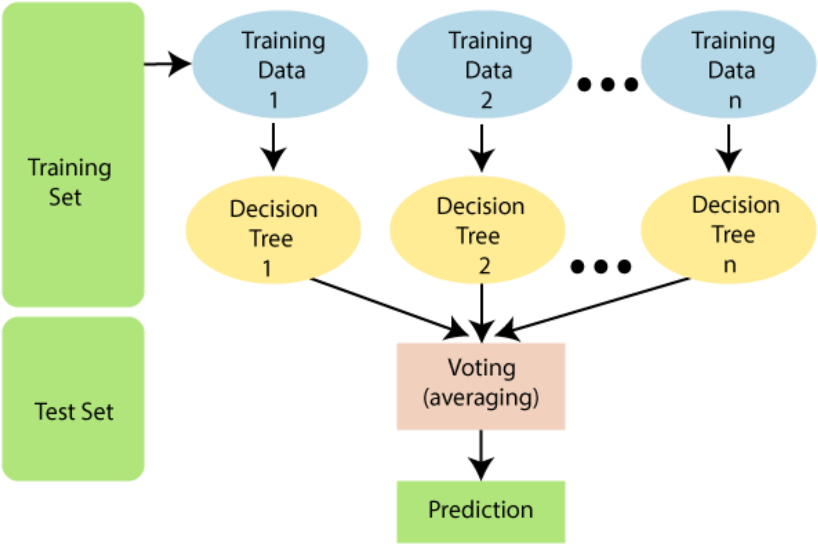


Figure 11: Random Forest Classifier.

Source: https://www.turing.com/kb/random-forest-algorithm#what-is-random-forest-algorithm?

Advantages of Random Forest Algorithm:

- The accuracy of this algorithm is generally high as it takes the average decision made by multiple decision trees.
- This algorithm can handle large datasets and is not influenced by outliers as well.

Disadvantages of Random Forest Algorithm:

- This algorithm is not easily interpretable as it doesn't provide proper visibility of coefficients.
- This algorithm can be intensive for large datasets. (Singh, 2020)

Here, there is a prediction system, that predicts whether the fruit is an apple or banana, as shown in the picture below, each multiple-decision tree makes predictions accordingly, but the average or the majority is then taken as the final prediction of the algorithm. In the picture, two decision trees have predicted that the unknown fruit might be an apple, so with the majority, the final prediction from the algorithm is apple.

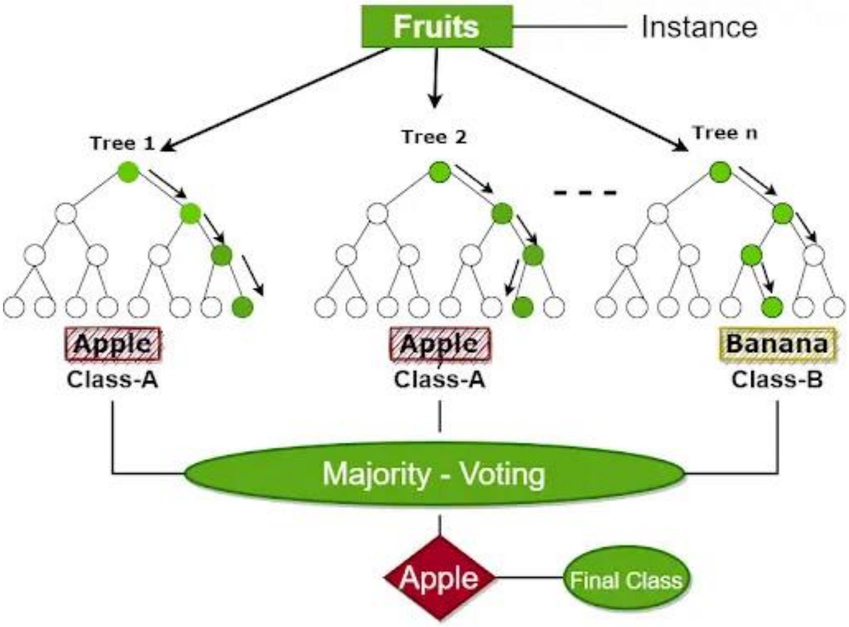


Figure 12: Example of random forest classifier.

Source: https://www.turing.com/kb/random-forest-algorithm#what-is-random-forest-algorithm?

3.2. Pseudocode:

```
START
    IMPORT OS
    IMPORT PANDAS AS PD
    IMPORT NUMPY AS NP
    IMPORT MATPLOTLIB.PYPLOT AS PLT
    IMPORT SEABORN AS SNS
    IMPORT TRAIN_TEST_SPLIT FROM SKLEARN.MODEL_SELECTION
    IMPORT STANDARDSCALER FROM SKLEARN.PREPROCESSING
    IMPORT KNEIGHBORSCLASSIFIER FROM SKLEARN.NEIGHBORS
    IMPORT LOGISTICREGRESSION FROM SKLEARN.LINEAR_MODEL
    IMPORT RANDOMFORESTCLASSIFIER FROM SKLEARN.ENSEMBLE
    IMPORT ACCURACY_SCORE, PRECISION_SCORE, RECALL_SCORE,
    CONFUSION_MATRIX FROM SKLEARN.METRICS
    IMPORT GRIDSEARCHCV FROM SKLEARN.MODEL_SELECTION
    IMPORT CROSS_VAL_SCORE FROM SKLEARN.MODEL_SELECTION

    IMPORT WARNINGS
    WARNINGS.FILTERWARNINGS('IGNORE')

    SET WORKING DIRECTORY

    LOAD THE DATASET

    EXPLORE THE DATASET

    ANALYZE AND VISUALIZE THE DATA

    SPLIT THE DATA INTO FEATURES (X) AND TARGET (Y), TRAIN-TEST SPLIT,
    SCALE THE DATA

    TRAIN MODELS ON THE DATASET
    EVALUATE MODELS ON TEST DATA
    PRINT EVALUATION METRICS

    TUNE THE HYPERPARAMETERS FOR DIFFERENT ACCURACIES
    RE-EVALUATE THE MODELS ON BEST HYPERPARAMETERS
    PRINT EVALUATION METRICS AFTER RE-EVALUATION

    VISUALIZE ACCURACY, PRECISION, RECALL, AND CONFUSION
    MATRICES
    PREDICT USING THE MODEL ON A NEW DATA
END
```

3.3. Flowchart:

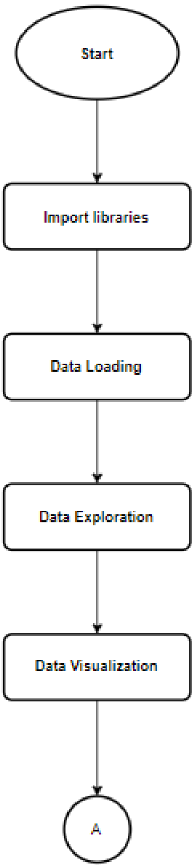


Figure 13: Flowchart-1

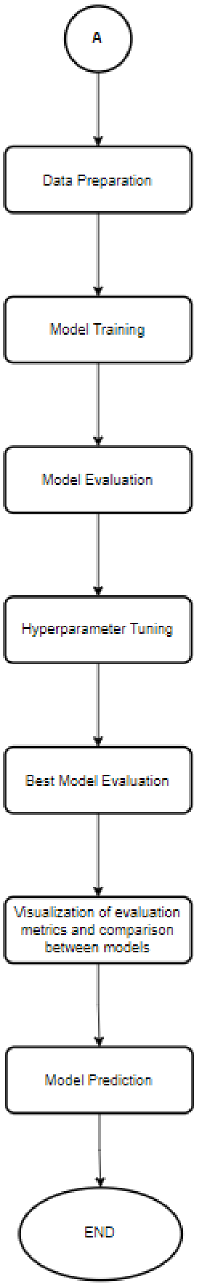


Figure 14: Flowchart-2

3.4. Development Process:

3.4.1. Tools/Toolkits used:

- **JupyterLab:**

  Jupyterlab is used in this project as a coding/editing environment. JupyterLab is a highly extensible, feature-rich notebook authoring application and editing environment. Jupyter Notebook is used in the JupyterLab for coding the system using python libraries. A computational notebook is a shareable document that combines computer code, plain language descriptions, data, rich visualizations like 3D models, charts, graphs and figures, and interactive controls. (JupyterLab, 2023)

- **Python:**

  Python is a versatile programming language that is commonly used in data science, machine learning, and so on. Here, python is the primary language for the implementation of my project heart disease prediction system.

- **Pandas:**

  Pandas is a python library that is mostly used for data manipulation. Here, in this project it is used for the same purpose which is loading, manipulating, and handling the dataset.

- **NumPy:**

  NumPy is also a library for python that is commonly used for performing scientific computing operations. It is often used in conjunction with Pandas. Here, in this project NumPy is used in multiple places for storing values and other different operations.

- **Matplotlib and Seaborn:**

  Matplotlib and Seaborn are also python libraries, these libraries are used for implementing graphical representation and visualizations. In this project, it is used for multiple visualizations of dataset, and evaluation metrics.

- **Scikit-learn:**

  Scikit-learn is a machine learning python library. In this project, various tools provided by this library has been used to build, train, and evaluate the model such as:

  sklearn.model_selection.train_test_split: This function from scikit-learn helps to split the dataset into train and test splits.

  sklearn.preprocessing.StandardScaler: This function helps to scale the dataset which is also known as standardizing the dataset which is crucial for some machine learning algorithms.

  sklearn.neighbors.KNeighborsClassifier: This function is used to implement KNN algorithm on our dataset.

  sklearn.linear_model.LogisticRegression: This function is used to implement Logistic Regression algorithm on our dataset.

  sklearn.ensemble.RandomForestClassifier: This function is used to implement Random Forest Algorithm on our dataset.

  sklearn.metrics.accuracy_score, precision_score, recall_score, confusion_matrix: These functions are used to calculate the accuracy, precision, recall and confusion matrix of all the models to evaluate the performance of each model.

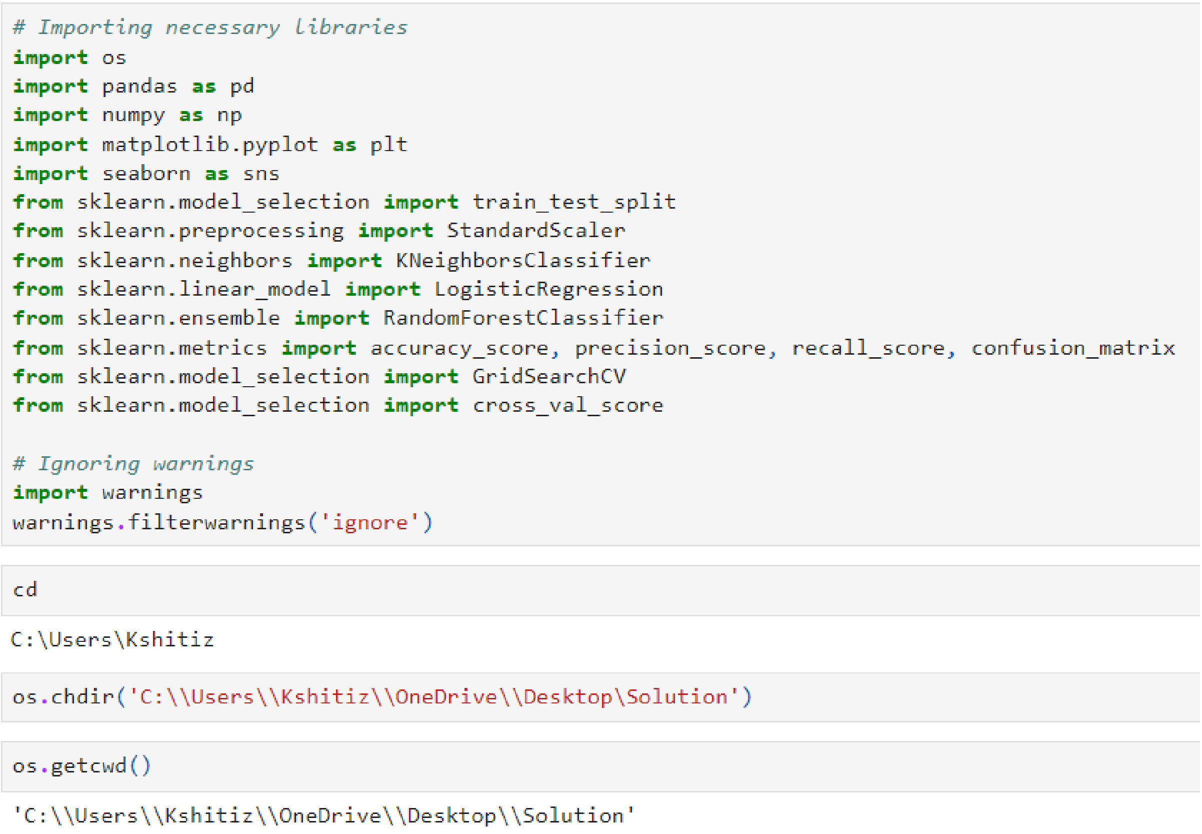
  sklearn.model_selection.GridSearchCV: This function is used for hyperparameter tuning for each models to achieve the best hyperparameters.

  sklearn.model_selection.cross_val_score: This function is used in our project to calculate the cross validation score which helps in obtaining a more robust performance estimate.

3.4.2. Entire Process:

3.4.2.1.Importing necessary libraries:

Firstly, all the necessary python libraries and scikit-learn functions are imported, and directory changed to the working directory where the dataset is available:



```
# Importing necessary libraries
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_val_score

# Ignoring warnings
import warnings
warnings.filterwarnings('ignore')
```

```
cd
```

```
C:\Users\Kshitiz
```

```
os.chdir('C:\\Users\\Kshitiz\\OneDrive\\Desktop\Solution')
```

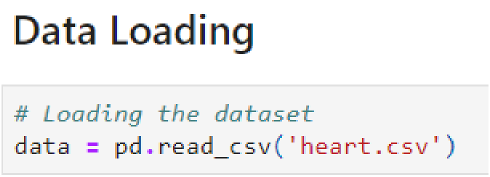
```
os.getcwd()
```

```
'C:\\Users\\Kshitiz\\OneDrive\\Desktop\\Solution'
```

Figure 15: Process screenshot: Importing libraries

3.4.2.2.Loading the data:

Second step is to load the data:



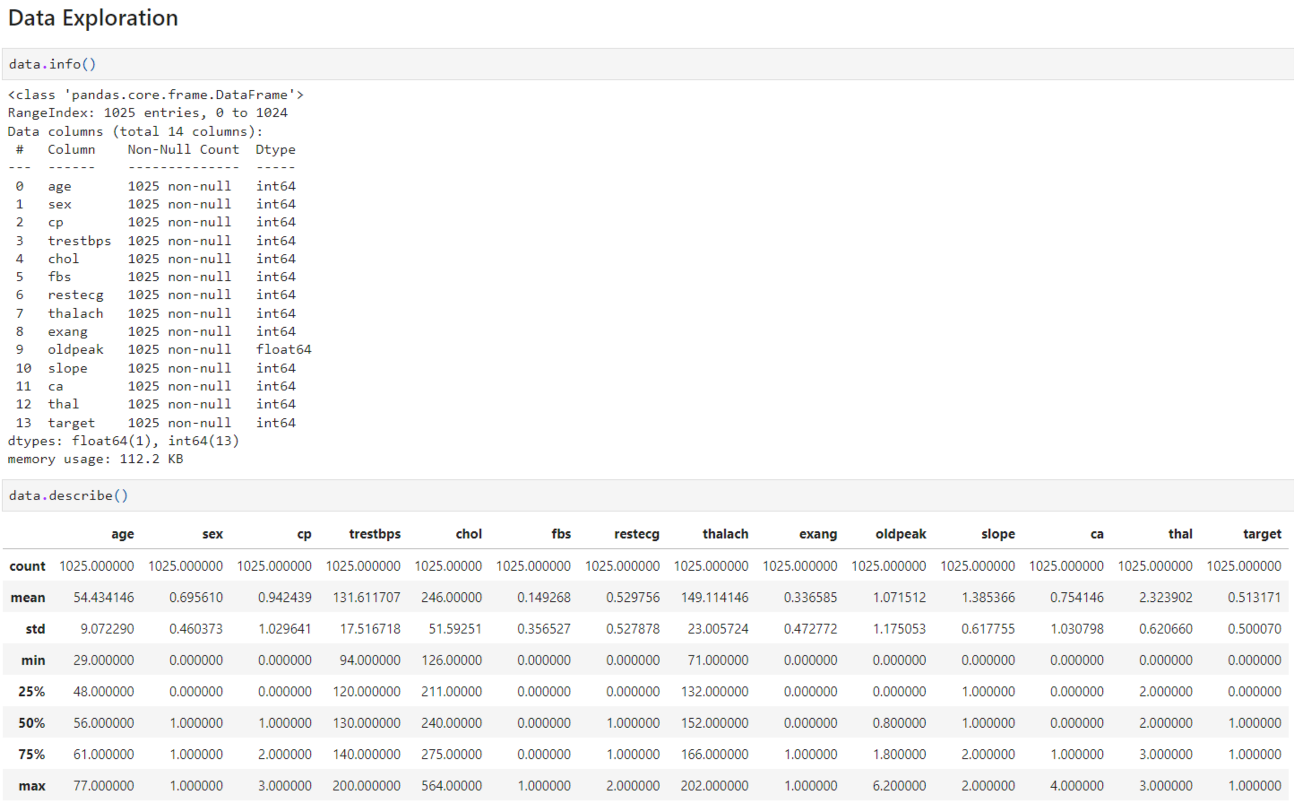
Data Loading

```
# Loading the dataset
data = pd.read_csv('heart.csv')
```

Figure 16: Process screenshot: Loading data

3.4.2.3.Data Exploration:

Third step is to explore the data to know about the attributes and values:



Data Exploration

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1025 non-null   int64  
 1   sex       1025 non-null   int64  
 2   cp        1025 non-null   int64  
 3   trestbps  1025 non-null   int64  
 4   chol      1025 non-null   int64  
 5   fbs       1025 non-null   int64  
 6   restecg   1025 non-null   int64  
 7   thalach   1025 non-null   int64  
 8   exang     1025 non-null   int64  
 9   oldpeak   1025 non-null   float64
 10  slope     1025 non-null   int64  
 11  ca        1025 non-null   int64  
 12  thal      1025 non-null   int64  
 13  target    1025 non-null   int64  
dtypes: float64(1), int64(13)
memory usage: 112.2 KB
```

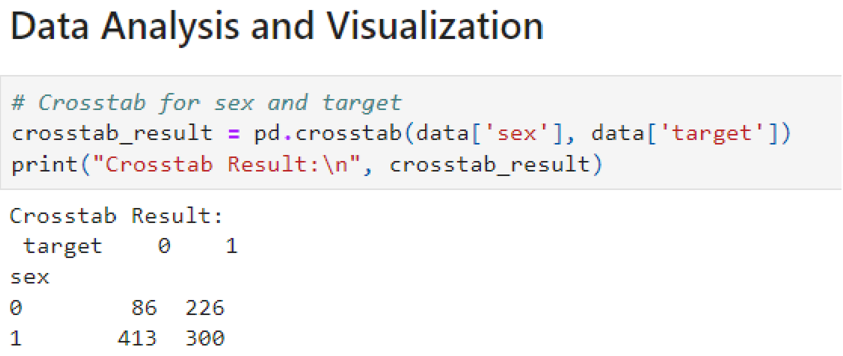
```
data.describe()
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
count	1025.000000	1025.000000	1025.000000	1025.000000	1025.00000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000	1025.000000
mean	54.434146	0.695610	0.942439	131.611707	246.00000	0.149268	0.529756	149.114146	0.336585	1.071512	1.385366	0.754146	2.323902	0.513171
std	9.072290	0.460373	1.029641	17.516718	51.59251	0.356527	0.527878	23.005724	0.472772	1.175053	0.617755	1.030798	0.620660	0.500070
min	29.000000	0.000000	0.000000	94.000000	126.00000	0.000000	0.000000	71.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	48.000000	0.000000	0.000000	120.000000	211.00000	0.000000	0.000000	132.000000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000
50%	56.000000	1.000000	1.000000	130.000000	240.00000	0.000000	1.000000	152.000000	0.000000	0.800000	1.000000	0.000000	2.000000	1.000000
75%	61.000000	1.000000	2.000000	140.000000	275.00000	0.000000	1.000000	166.000000	1.000000	1.800000	2.000000	1.000000	3.000000	1.000000
max	77.000000	1.000000	3.000000	200.000000	564.00000	1.000000	2.000000	202.000000	1.000000	6.200000	2.000000	4.000000	3.000000	1.000000

Figure 17: Process screenshot: Data exploration

3.4.2.4.Data analysis and visualization:

This is the fourth stage where data analysis and visualization is done to understand the data graphically, in this stage multiple visualizations are done which includes: crosstab, features distribution, features correlation matrix and so on, the visualizations done are shown below:

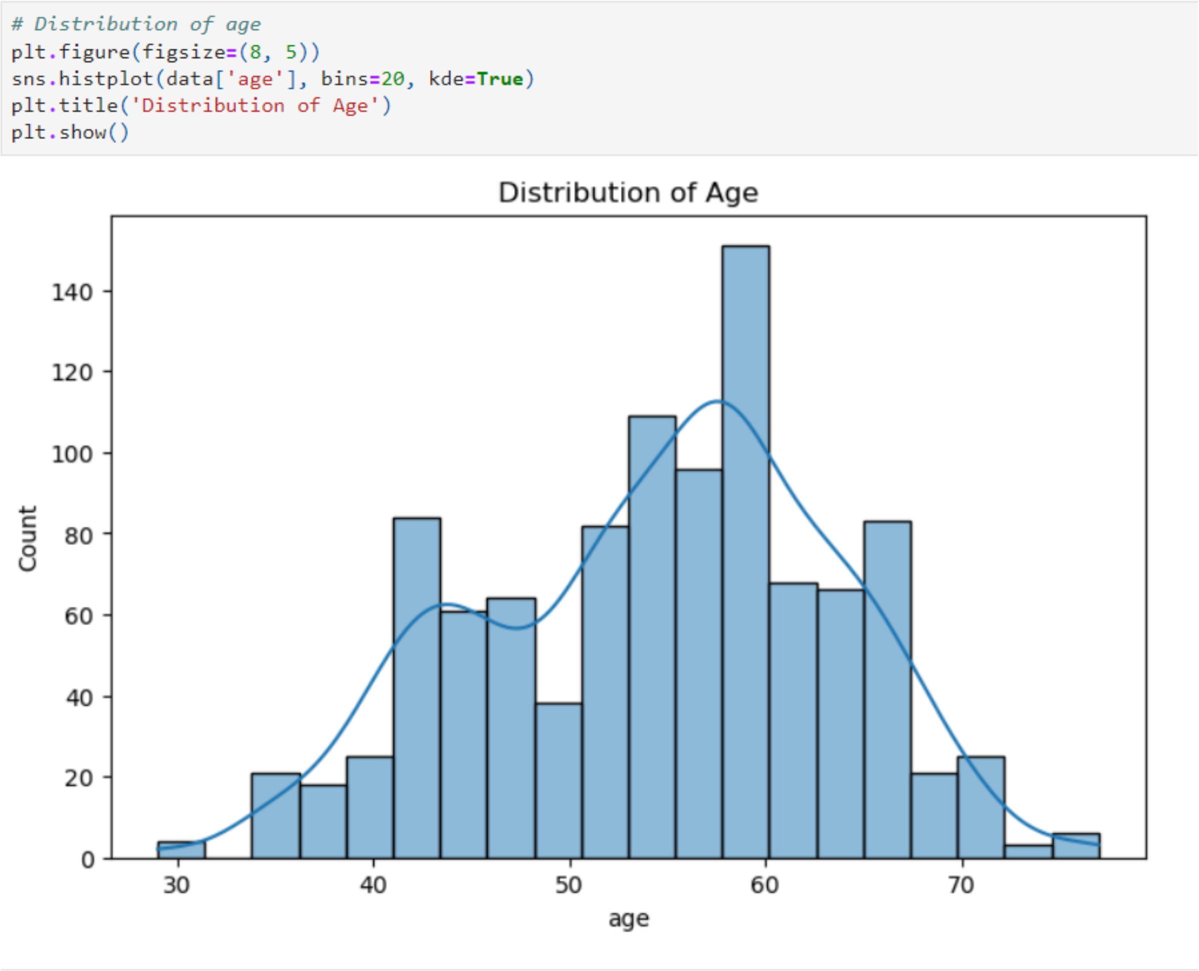


Data Analysis and Visualization

```
# Crosstab for sex and target
crosstab_result = pd.crosstab(data['sex'], data['target'])
print("Crosstab Result:\n", crosstab_result)
```

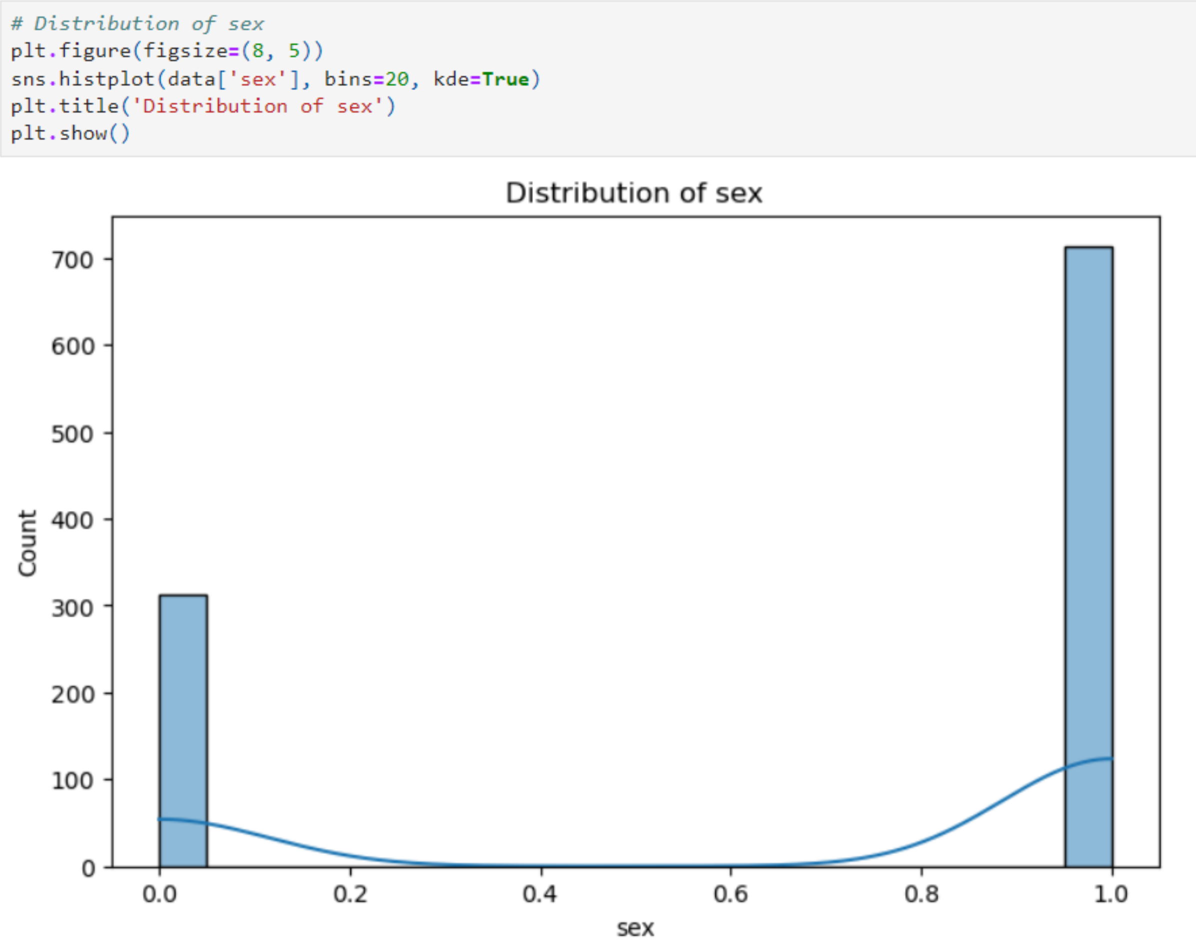
```
Crosstab Result:
 target    0    1
sex             
0        86  226
1       413  300
```

Figure 18: Process Screenshot: Data visualization 1



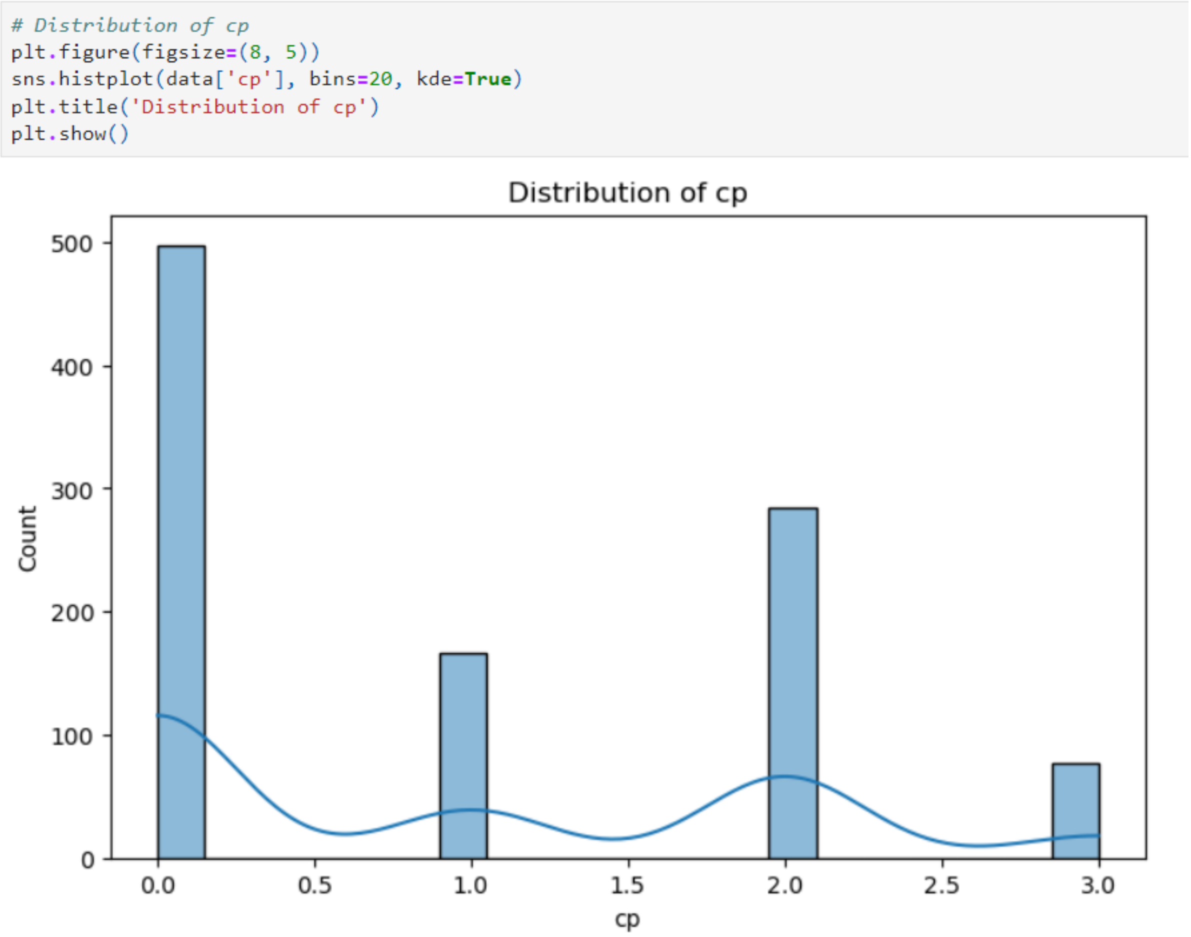
```
# Distribution of age
plt.figure(figsize=(8, 5))
sns.histplot(data['age'], bins=20, kde=True)
plt.title('Distribution of Age')
plt.show()
```

Figure 19: Process Screenshot: Data visualization 2



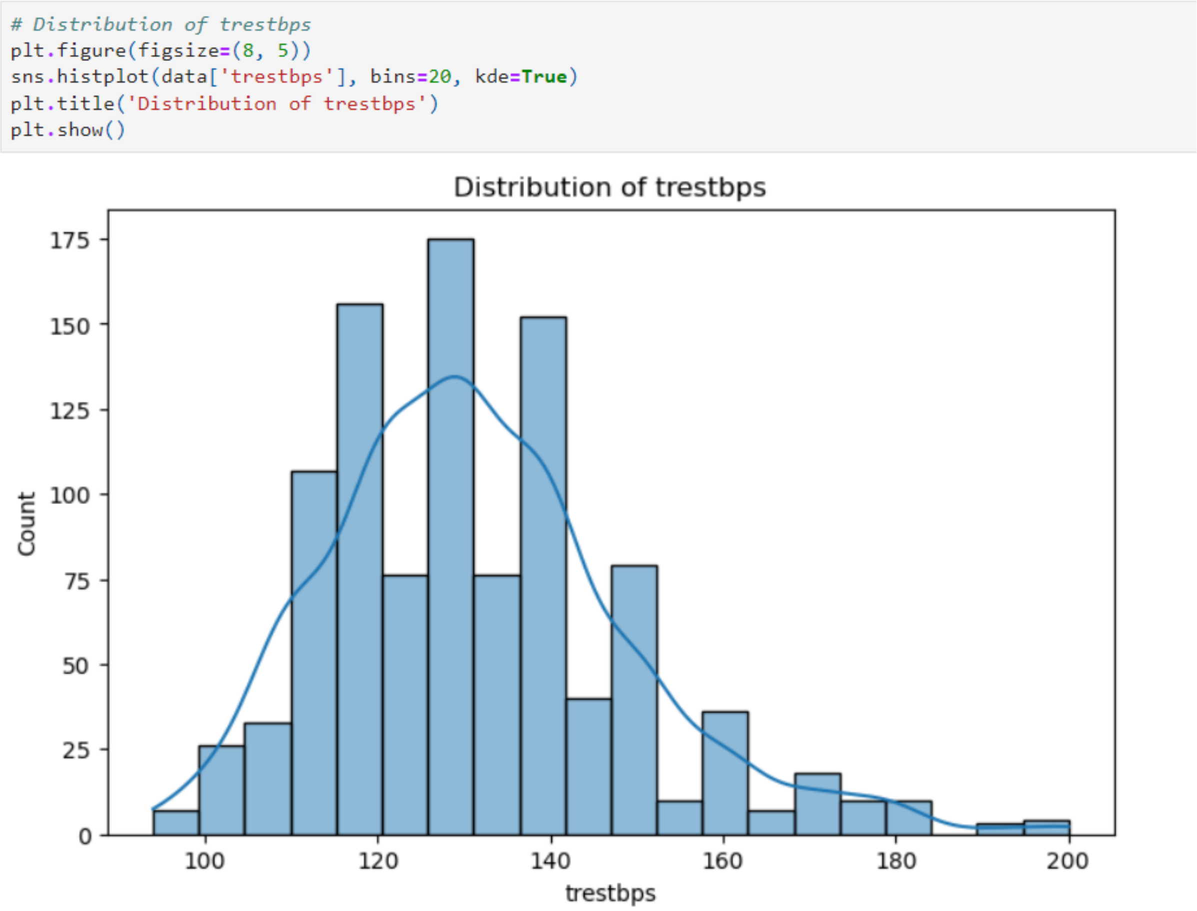
```
# Distribution of sex
plt.figure(figsize=(8, 5))
sns.histplot(data['sex'], bins=20, kde=True)
plt.title('Distribution of sex')
plt.show()
```

Figure 20: Process Screenshot: Data visualization 3



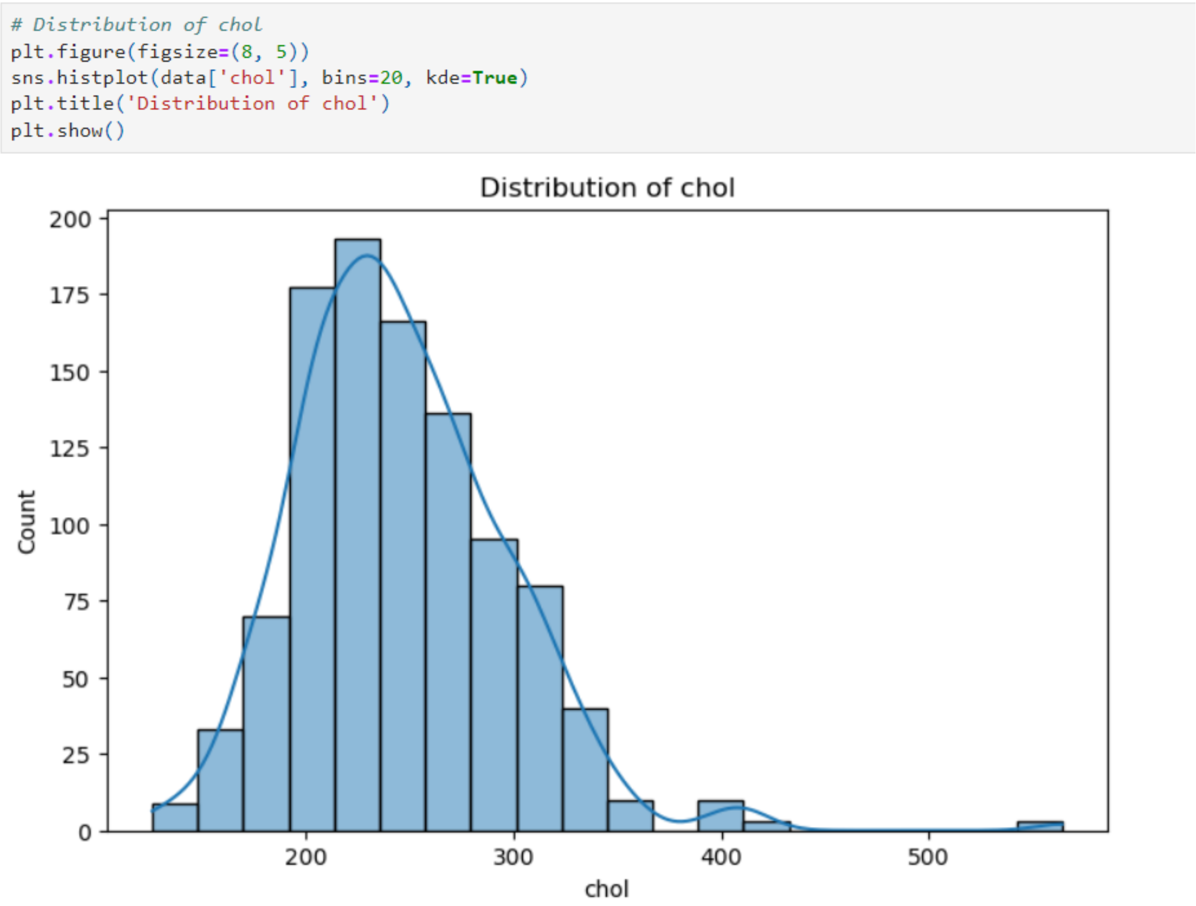
```
# Distribution of cp
plt.figure(figsize=(8, 5))
sns.histplot(data['cp'], bins=20, kde=True)
plt.title('Distribution of cp')
plt.show()
```

Figure 21: Process Screenshot: Data visualization 4



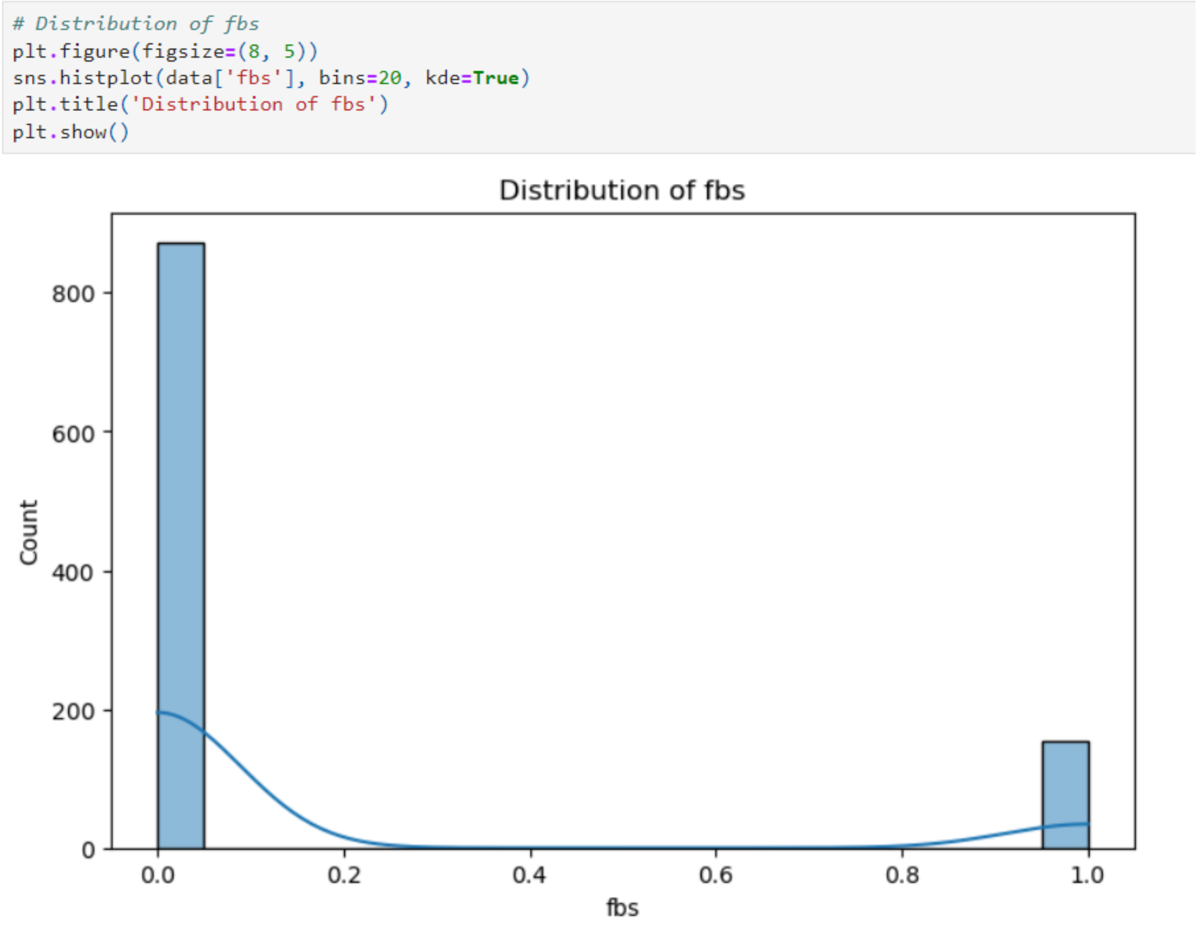
```
# Distribution of trestbps
plt.figure(figsize=(8, 5))
sns.histplot(data['trestbps'], bins=20, kde=True)
plt.title('Distribution of trestbps')
plt.show()
```

Figure 22: Process Screenshot: Data visualization 5



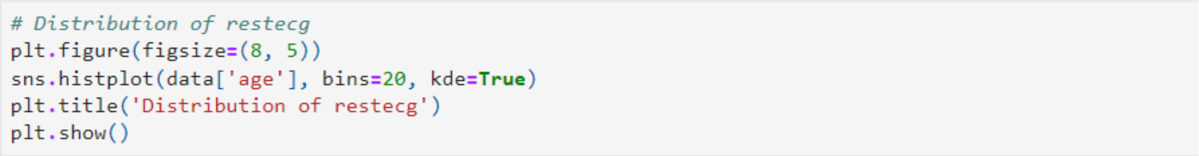
```
# Distribution of chol
plt.figure(figsize=(8, 5))
sns.histplot(data['chol'], bins=20, kde=True)
plt.title('Distribution of chol')
plt.show()
```

Figure 23: Process Screenshot: Data visualization 6



```
# Distribution of fbs
plt.figure(figsize=(8, 5))
sns.histplot(data['fbs'], bins=20, kde=True)
plt.title('Distribution of fbs')
plt.show()
```

Figure 24: Process Screenshot: Data visualization 7



```
# Distribution of restecg
plt.figure(figsize=(8, 5))
sns.histplot(data['age'], bins=20, kde=True)
plt.title('Distribution of restecg')
plt.show()
```

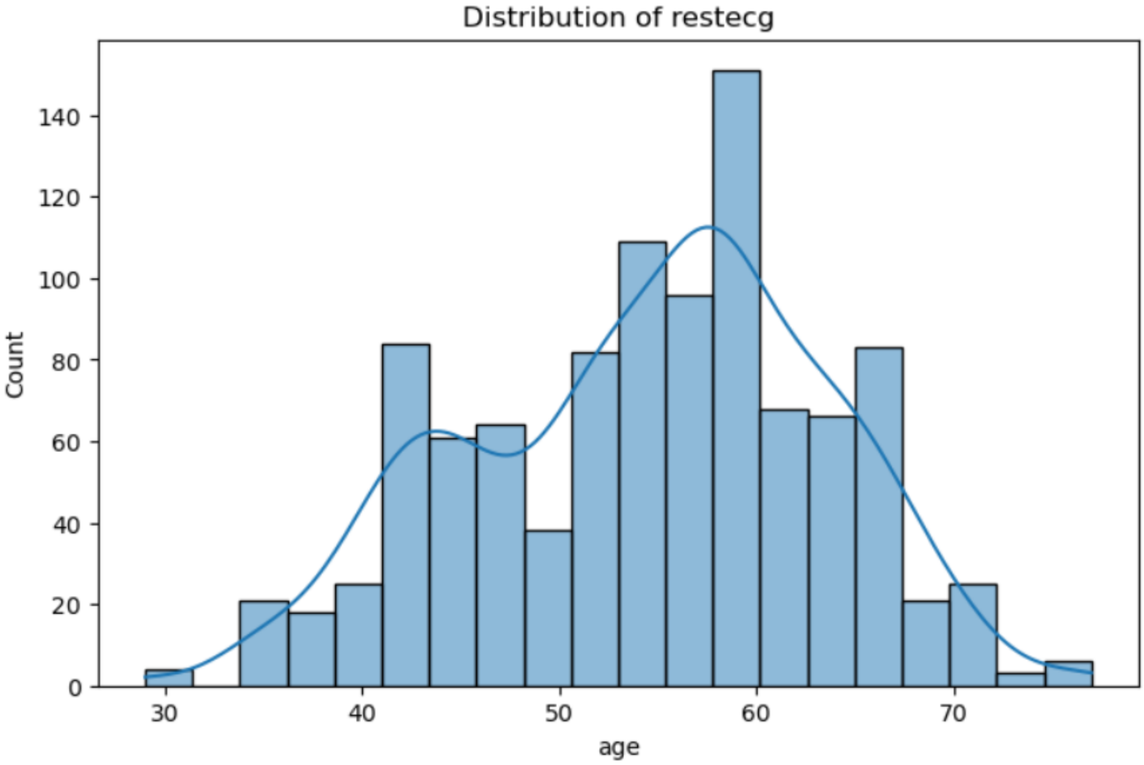
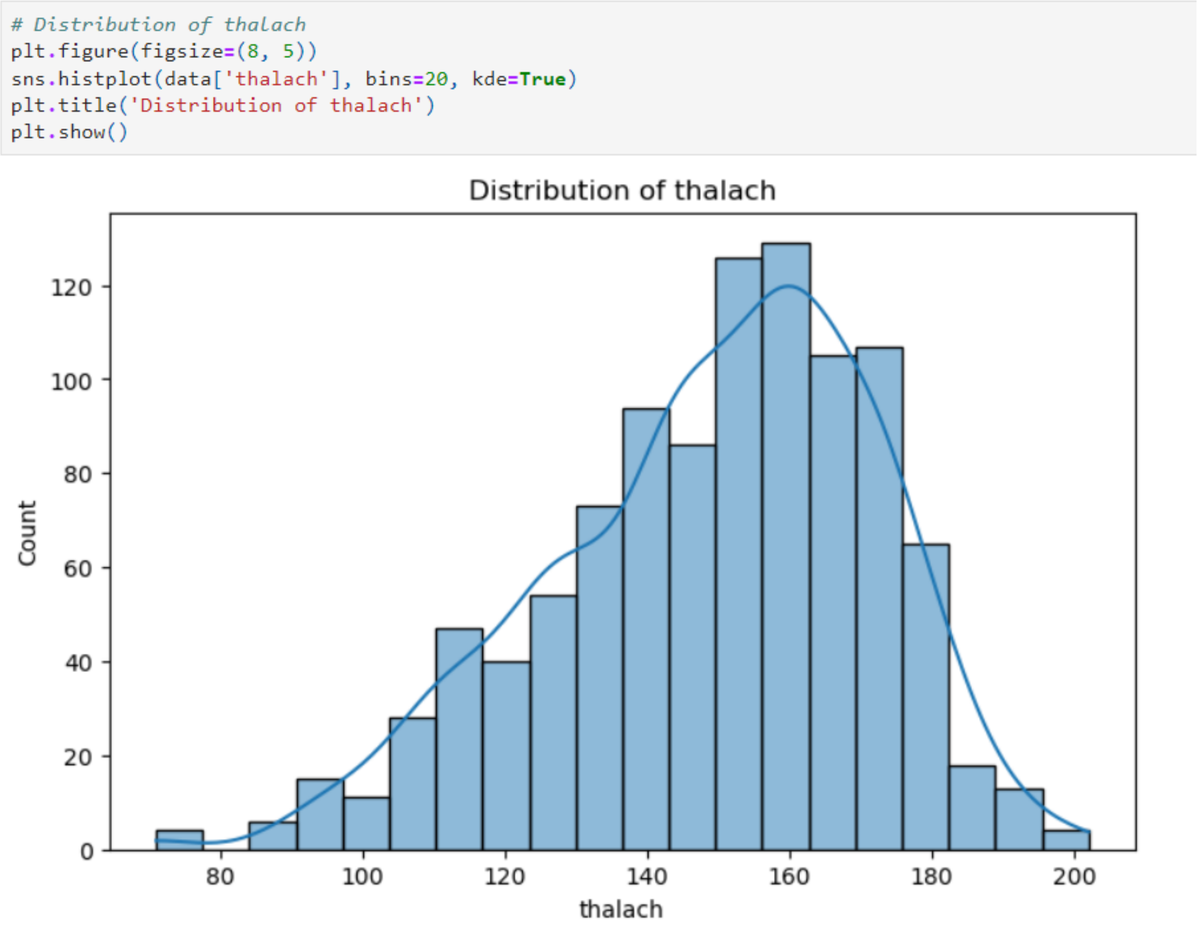
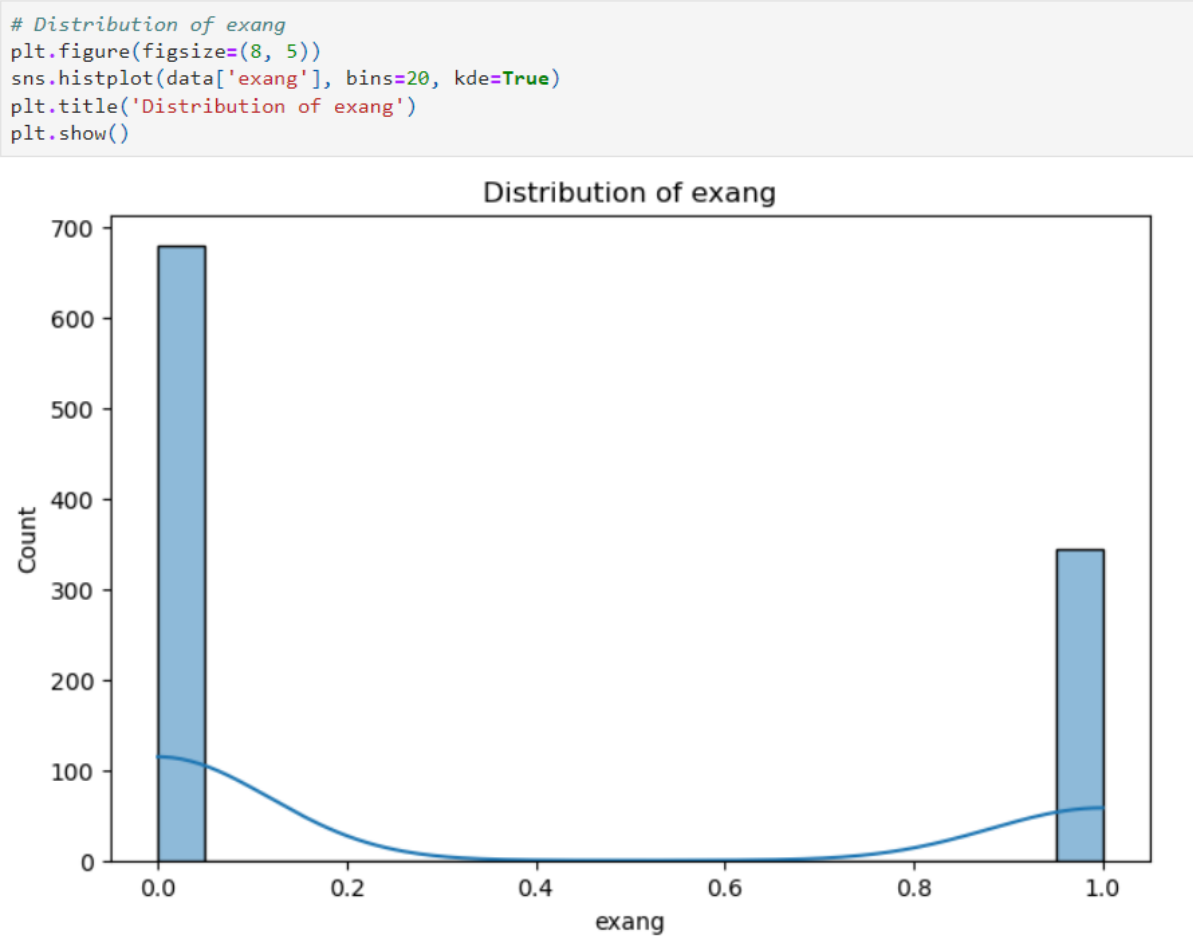


Figure 25: Process Screenshot: Data visualization 8



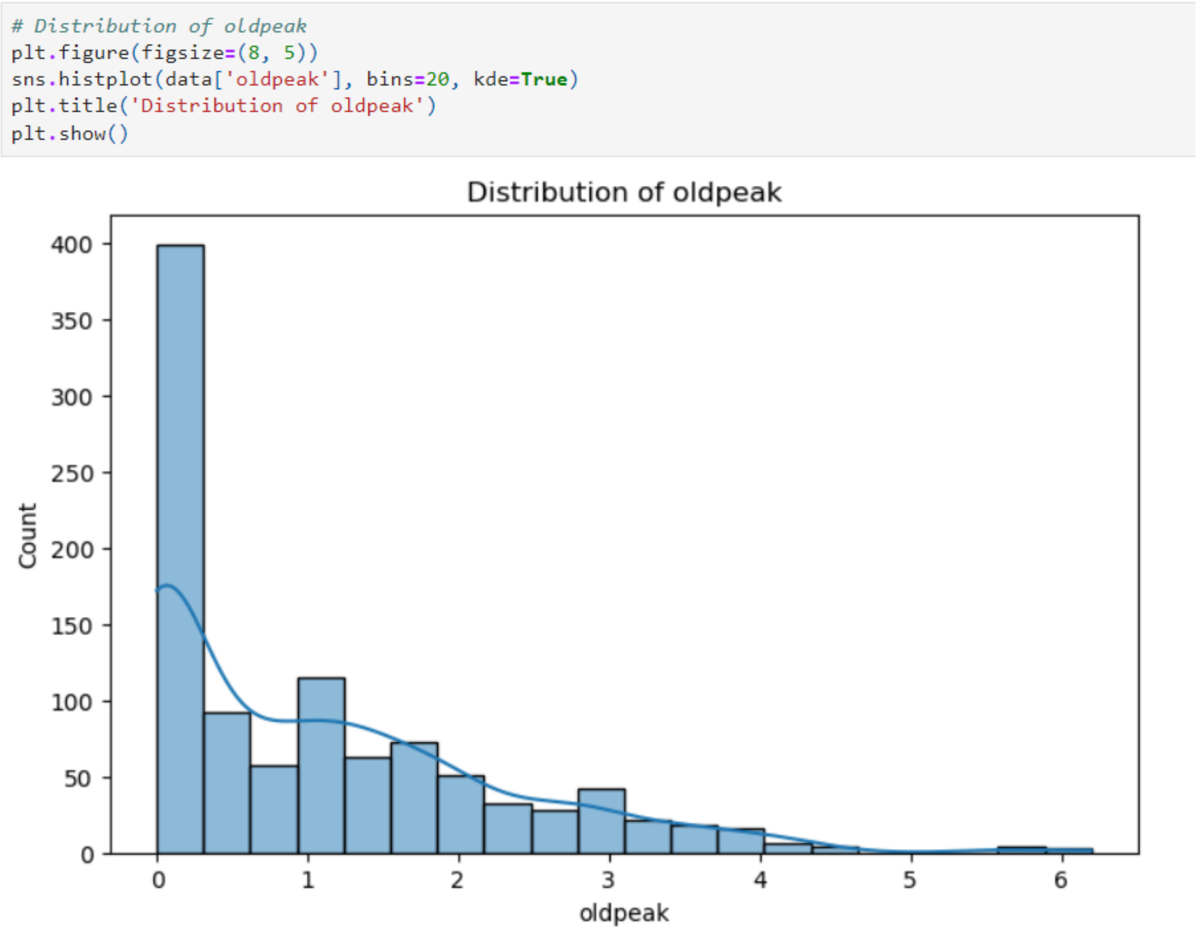
```
# Distribution of thalach
plt.figure(figsize=(8, 5))
sns.histplot(data['thalach'], bins=20, kde=True)
plt.title('Distribution of thalach')
plt.show()
```

Figure 26: Process Screenshot: Data visualization 9



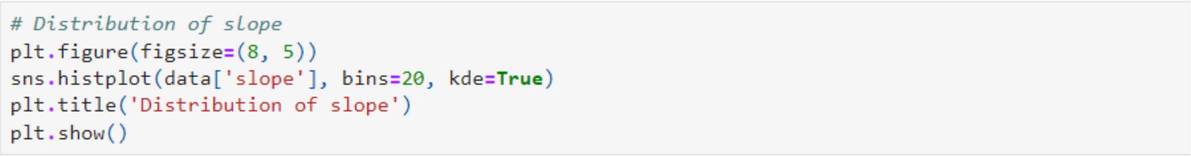
```
# Distribution of exang
plt.figure(figsize=(8, 5))
sns.histplot(data['exang'], bins=20, kde=True)
plt.title('Distribution of exang')
plt.show()
```

Figure 27: Process Screenshot: Data visualization 10



```
# Distribution of oldpeak
plt.figure(figsize=(8, 5))
sns.histplot(data['oldpeak'], bins=20, kde=True)
plt.title('Distribution of oldpeak')
plt.show()
```

Figure 28: Process Screenshot: Data visualization 11



```
# Distribution of slope
plt.figure(figsize=(8, 5))
sns.histplot(data['slope'], bins=20, kde=True)
plt.title('Distribution of slope')
plt.show()
```

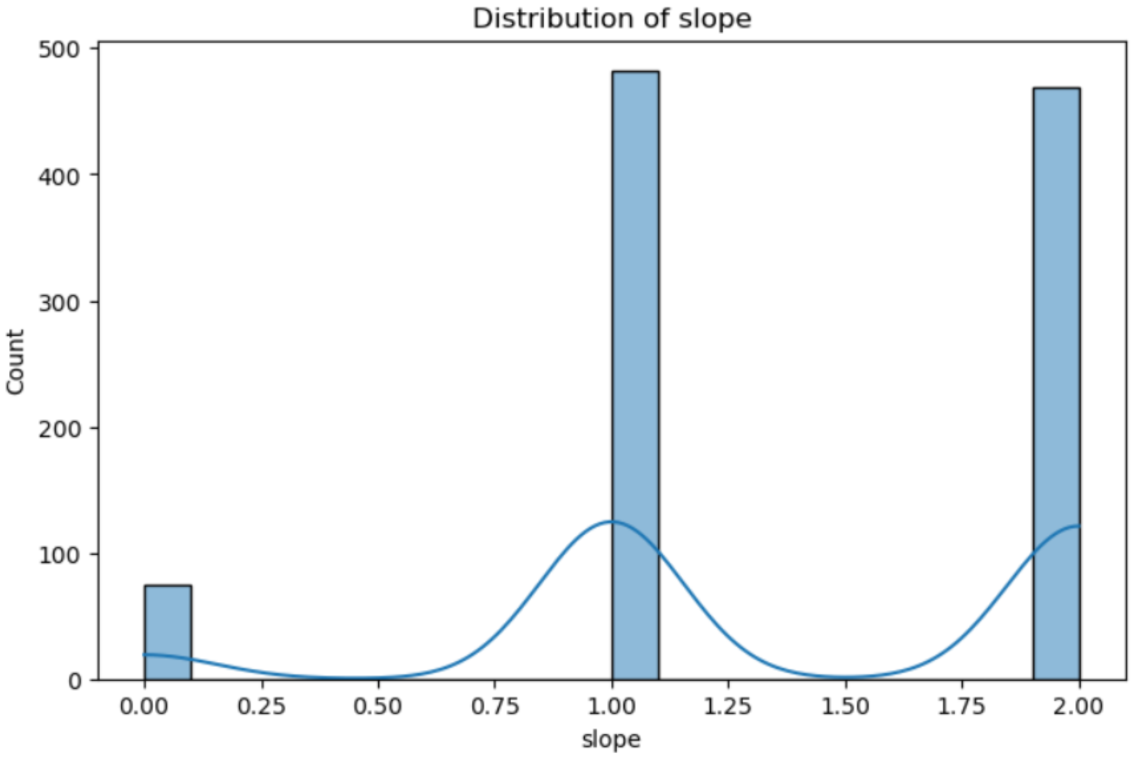
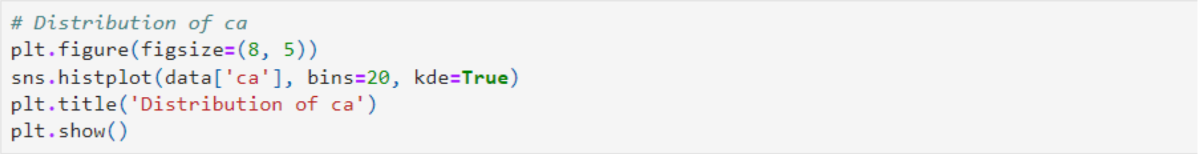


Figure 29: Process Screenshot: Data visualization 12



```
# Distribution of ca
plt.figure(figsize=(8, 5))
sns.histplot(data['ca'], bins=20, kde=True)
plt.title('Distribution of ca')
plt.show()
```

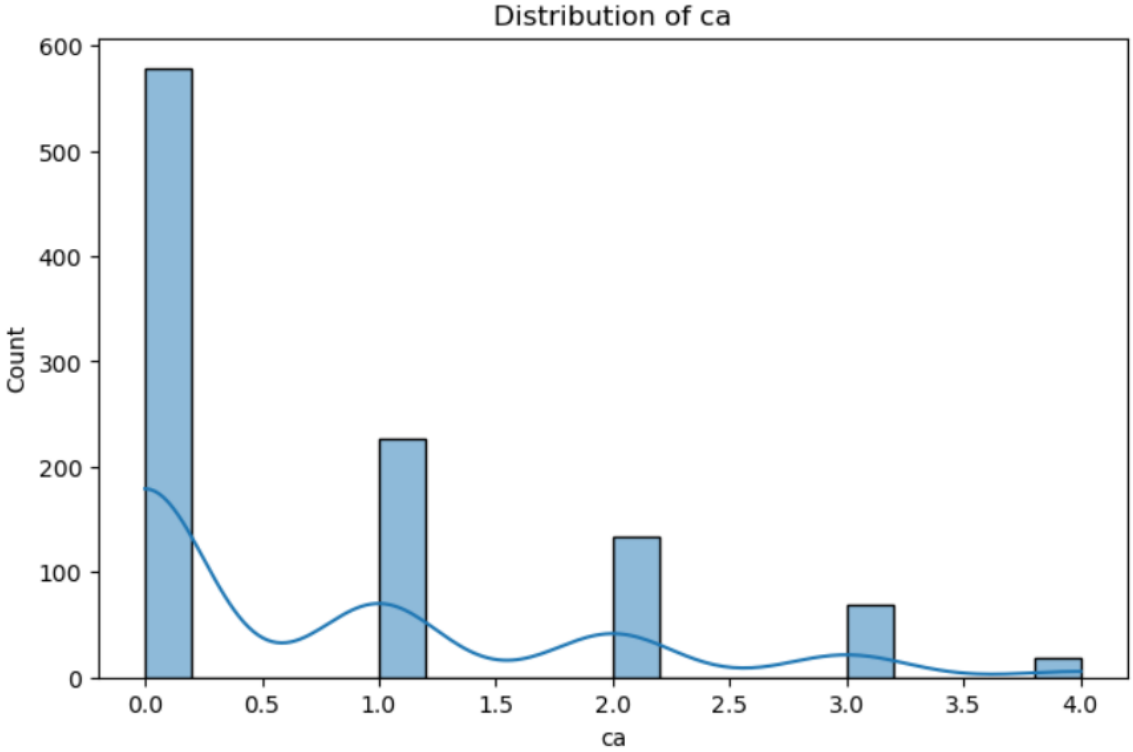
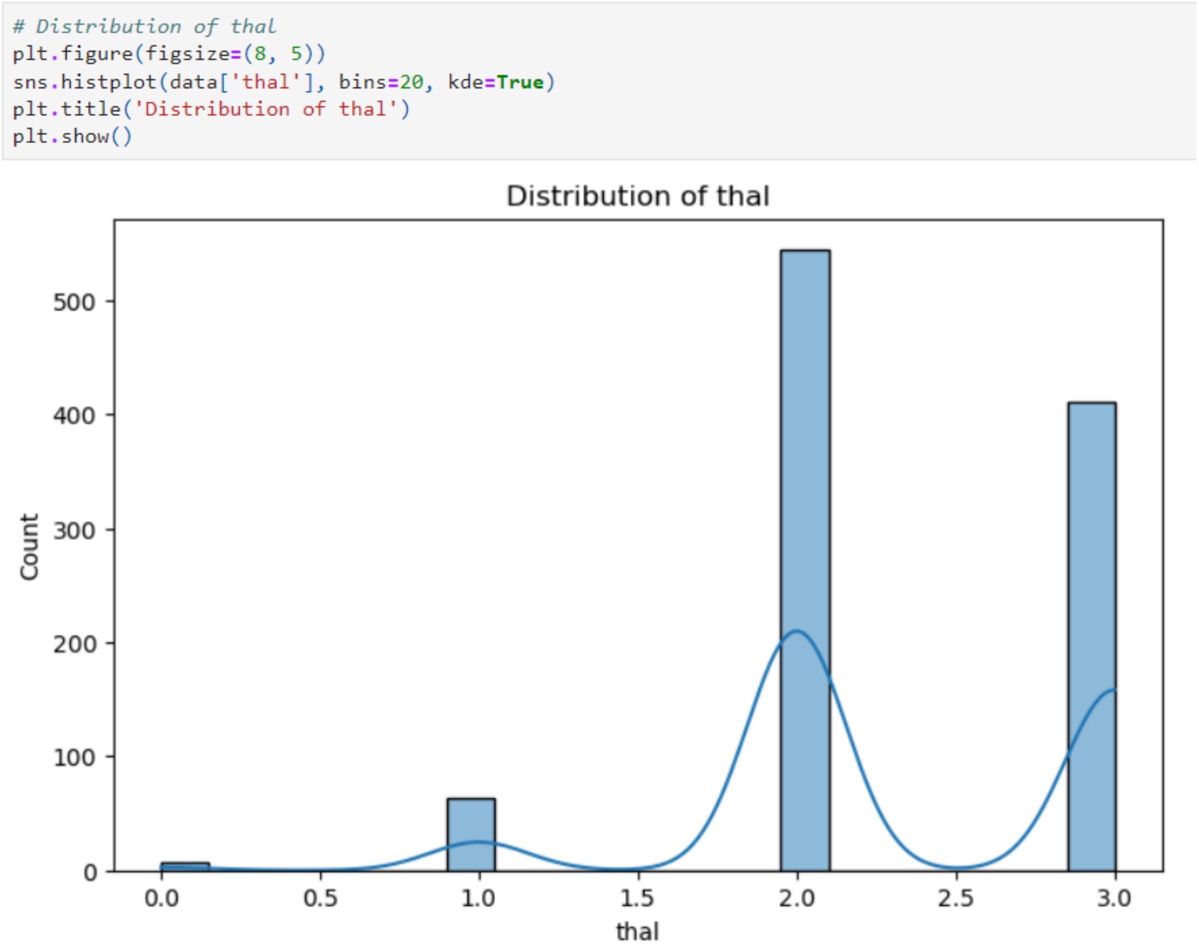
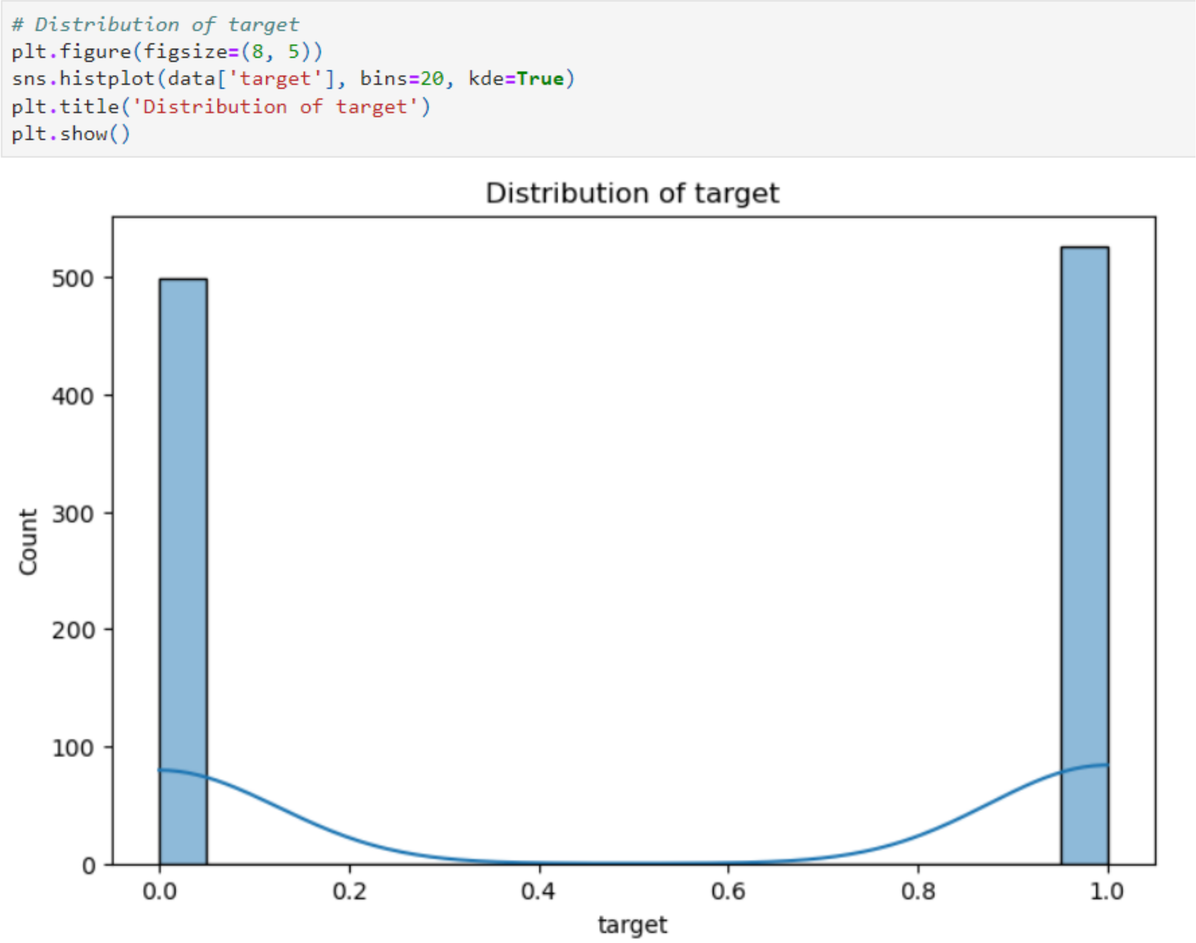


Figure 30: Process Screenshot: Data visualization 13



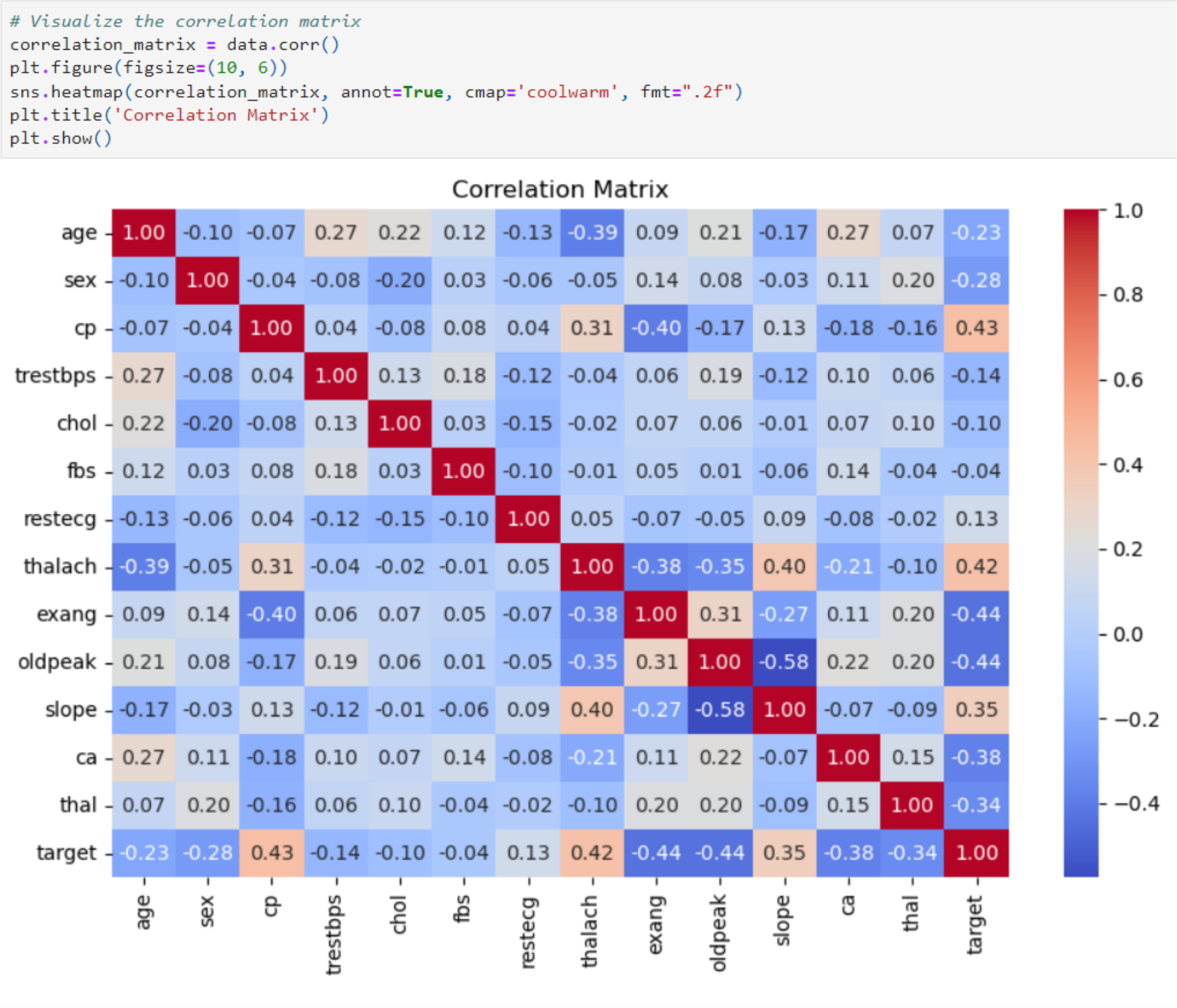
```
# Distribution of thal
plt.figure(figsize=(8, 5))
sns.histplot(data['thal'], bins=20, kde=True)
plt.title('Distribution of thal')
plt.show()
```

Figure 31: Process Screenshot: Data visualization 14



```
# Distribution of target
plt.figure(figsize=(8, 5))
sns.histplot(data['target'], bins=20, kde=True)
plt.title('Distribution of target')
plt.show()
```

Figure 32: Process Screenshot: Data visualization 15

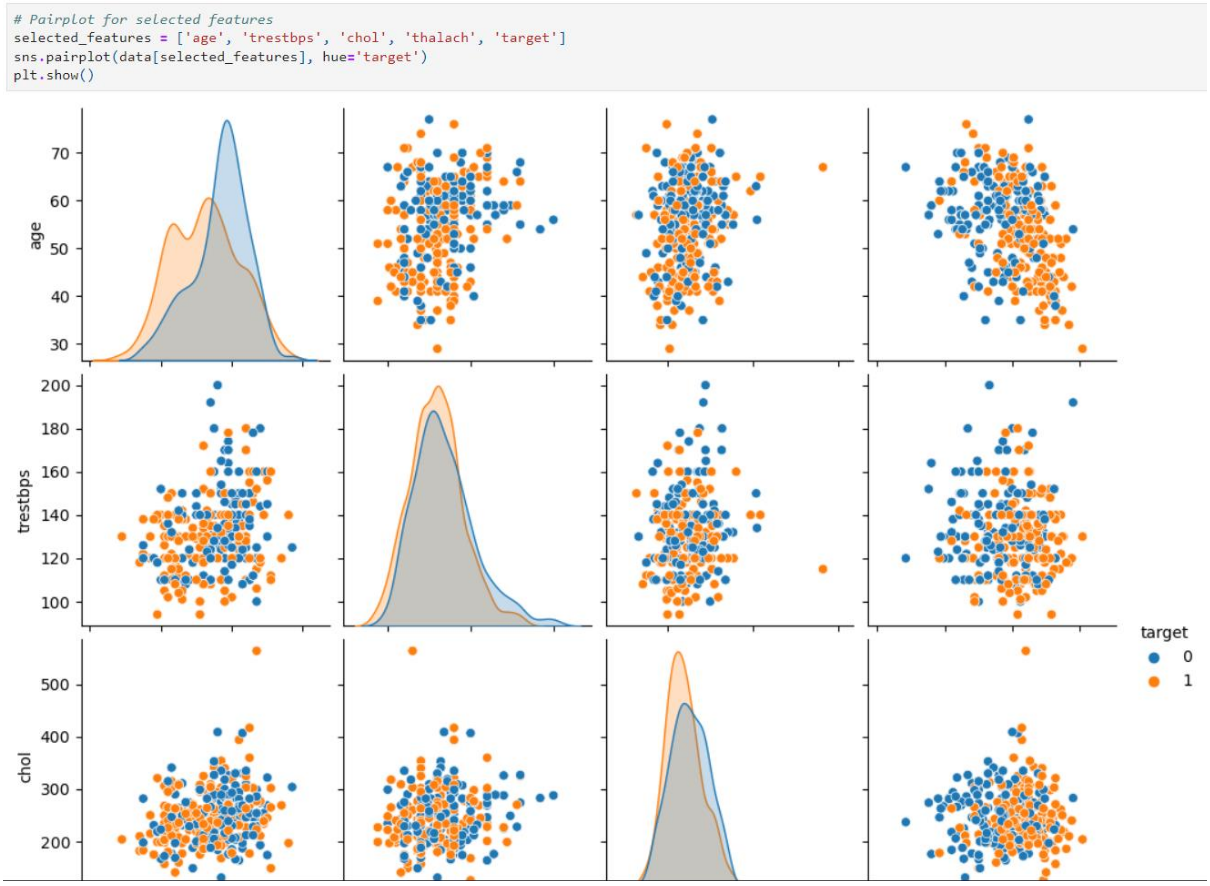


```
# Visualize the correlation matrix
correlation_matrix = data.corr()
plt.figure(figsize=(10, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```

Correlation Matrix

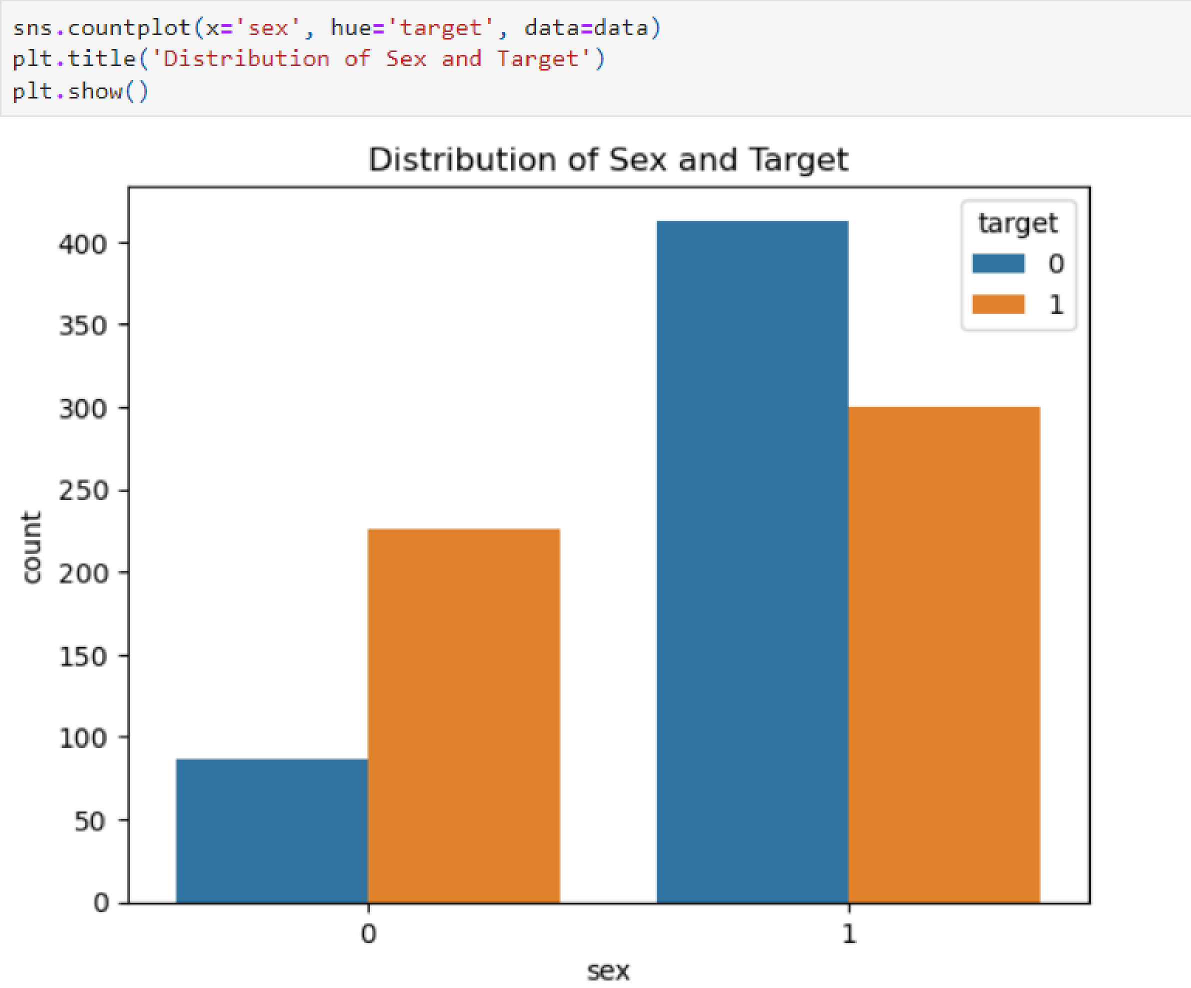
	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
age	1.00	-0.10	-0.07	0.27	0.22	0.12	-0.13	-0.39	0.09	0.21	-0.17	0.27	0.07	-0.23
sex	-0.10	1.00	-0.04	-0.08	-0.20	0.03	-0.06	-0.05	0.14	0.08	-0.03	0.11	0.20	-0.28
cp	-0.07	-0.04	1.00	0.04	-0.08	0.08	0.04	0.31	-0.40	-0.17	0.13	-0.18	-0.16	0.43
trestbps	0.27	-0.08	0.04	1.00	0.13	0.18	-0.12	-0.04	0.06	0.19	-0.12	0.10	0.06	-0.14
chol	0.22	-0.20	-0.08	0.13	1.00	0.03	-0.15	-0.02	0.07	0.06	-0.01	0.07	0.10	-0.10
fbs	0.12	0.03	0.08	0.18	0.03	1.00	-0.10	-0.01	0.05	0.01	-0.06	0.14	-0.04	-0.04
restecg	-0.13	-0.06	0.04	-0.12	-0.15	-0.10	1.00	0.05	-0.07	-0.05	0.09	-0.08	-0.02	0.13
thalach	-0.39	-0.05	0.31	-0.04	-0.02	-0.01	0.05	1.00	-0.38	-0.35	0.40	-0.21	-0.10	0.42
exang	0.09	0.14	-0.40	0.06	0.07	0.05	-0.07	-0.38	1.00	0.31	-0.27	0.11	0.20	-0.44
oldpeak	0.21	0.08	-0.17	0.19	0.06	0.01	-0.05	-0.35	0.31	1.00	-0.58	0.22	0.20	-0.44
slope	-0.17	-0.03	0.13	-0.12	-0.01	-0.06	0.09	0.40	-0.27	-0.58	1.00	-0.07	-0.09	0.35
ca	0.27	0.11	-0.18	0.10	0.07	0.14	-0.08	-0.21	0.11	0.22	-0.07	1.00	0.15	-0.38
thal	0.07	0.20	-0.16	0.06	0.10	-0.04	-0.02	-0.10	0.20	0.20	-0.09	0.15	1.00	-0.34
target	-0.23	-0.28	0.43	-0.14	-0.10	-0.04	0.13	0.42	-0.44	-0.44	0.35	-0.38	-0.34	1.00

Figure 33: Process Screenshot: Data visualization 16



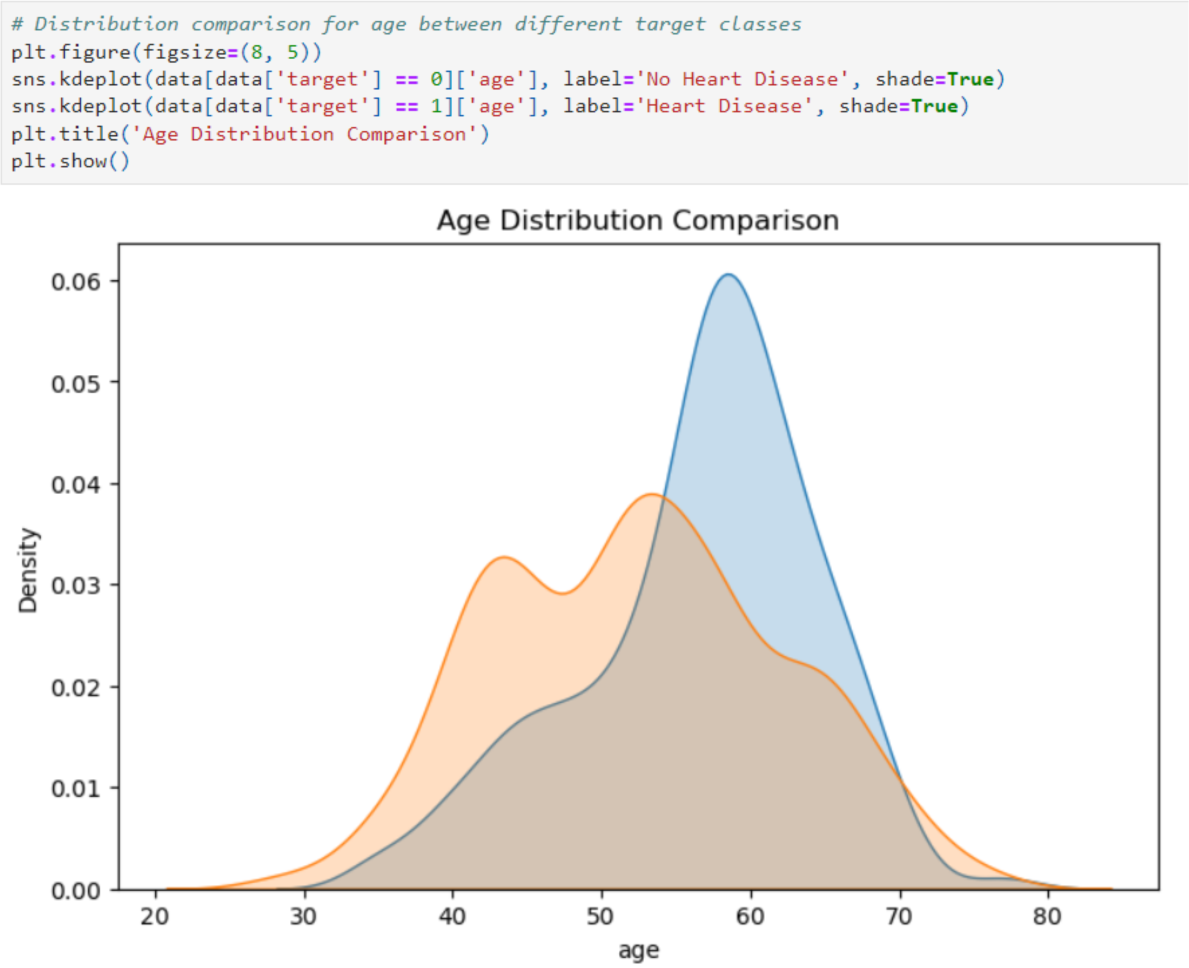
```
# Pairplot for selected features
selected_features = ['age', 'trestbps', 'chol', 'thalach', 'target']
sns.pairplot(data[selected_features], hue='target')
plt.show()
```

Figure 34: Process Screenshot: Data visualization 17



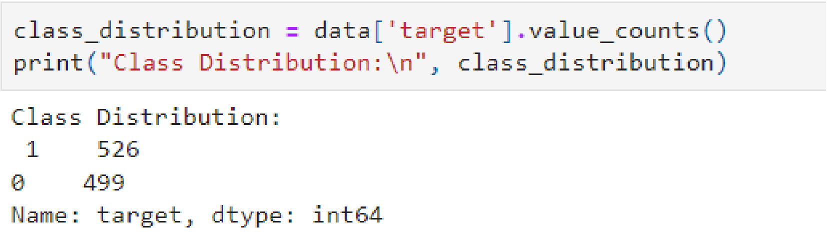
```
sns.countplot(x='sex', hue='target', data=data)
plt.title('Distribution of Sex and Target')
plt.show()
```

Figure 35: Process Screenshot: Data visualization 18



```
# Distribution comparison for age between different target classes
plt.figure(figsize=(8, 5))
sns.kdeplot(data[data['target'] == 0]['age'], label='No Heart Disease', shade=True)
sns.kdeplot(data[data['target'] == 1]['age'], label='Heart Disease', shade=True)
plt.title('Age Distribution Comparison')
plt.show()
```

Figure 36: Process Screenshot: Data visualization 19



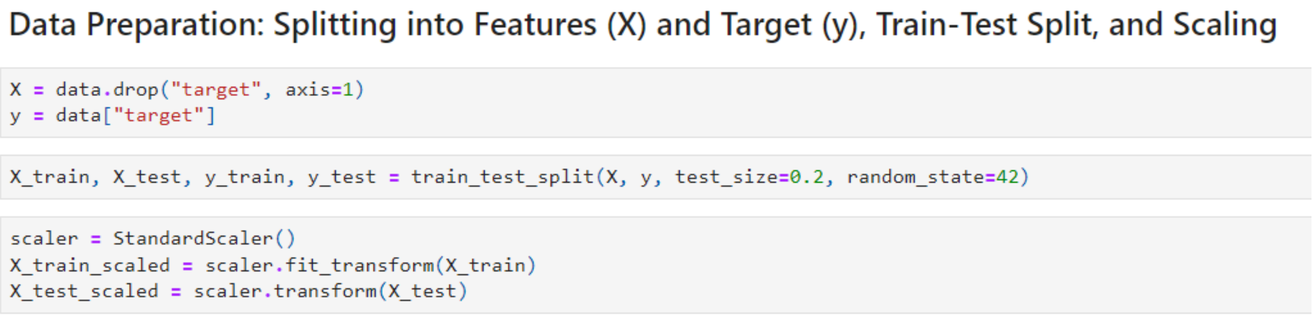
```
class_distribution = data['target'].value_counts()
print("Class Distribution:\n", class_distribution)
```

```
Class Distribution:
 1    526
0    499
Name: target, dtype: int64
```

Figure 37: Process Screenshot: Data visualization 20

3.4.2.5.Data preparation:

The next stage after visualization is data preparation. In this stage, the data is split into X(features) and y(target) and train and test splits where 80% is used for train split and 20% is used for test split. The dataset is then scaled using standard scaling.



Data Preparation: Splitting into Features (X) and Target (y), Train-Test Split, and Scaling

```
X = data.drop("target", axis=1)
y = data["target"]
```

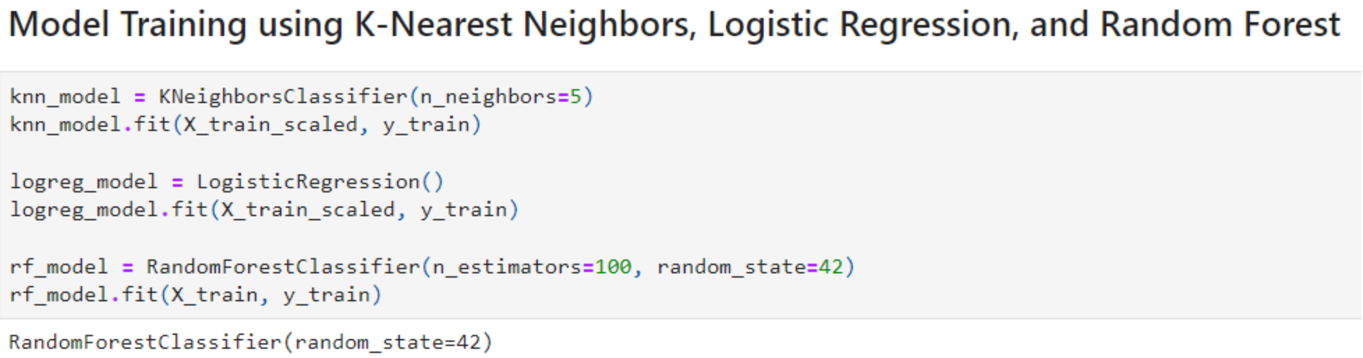
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

Figure 38: Data preparation

3.4.2.6.Model Training:

The next stage is training the model, here the model is trained using the prepared dataset. In our case, we train three models using the prepared dataset as we are using three different algorithms.



Model Training using K-Nearest Neighbors, Logistic Regression, and Random Forest

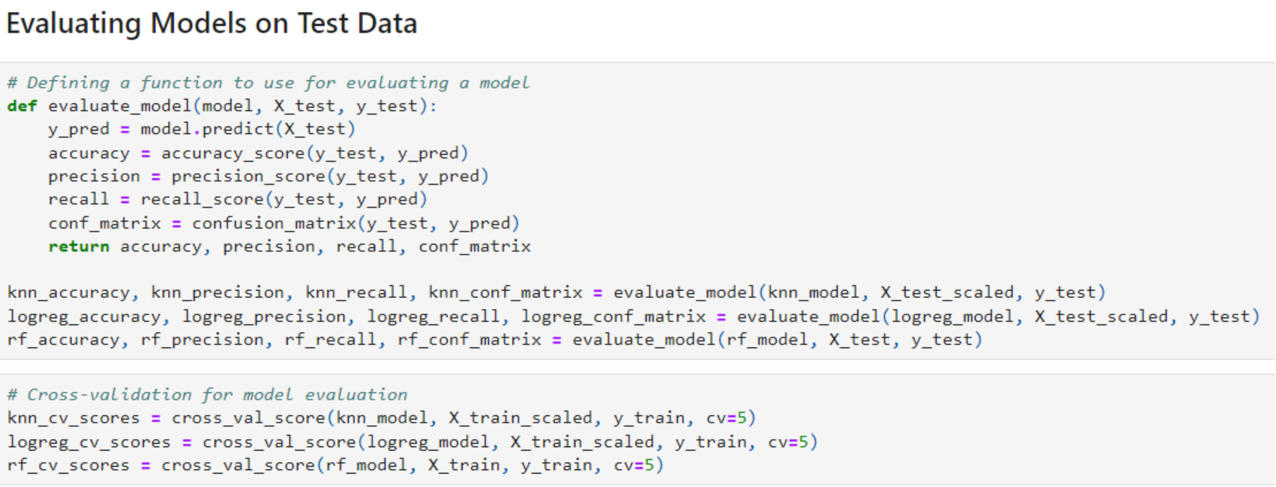
```
knn_model = KNeighborsClassifier(n_neighbors=5)
knn_model.fit(X_train_scaled, y_train)

logreg_model = LogisticRegression()
logreg_model.fit(X_train_scaled, y_train)

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
```

```
RandomForestClassifier(random_state=42)
```

Figure 39: Model Training

3.4.2.7.Model Evaluation:

In this stage, the trained model is evaluated, and the evaluated metrics are printed which consists of accuracy, precision, recall, confusion matrix and cross validation score.

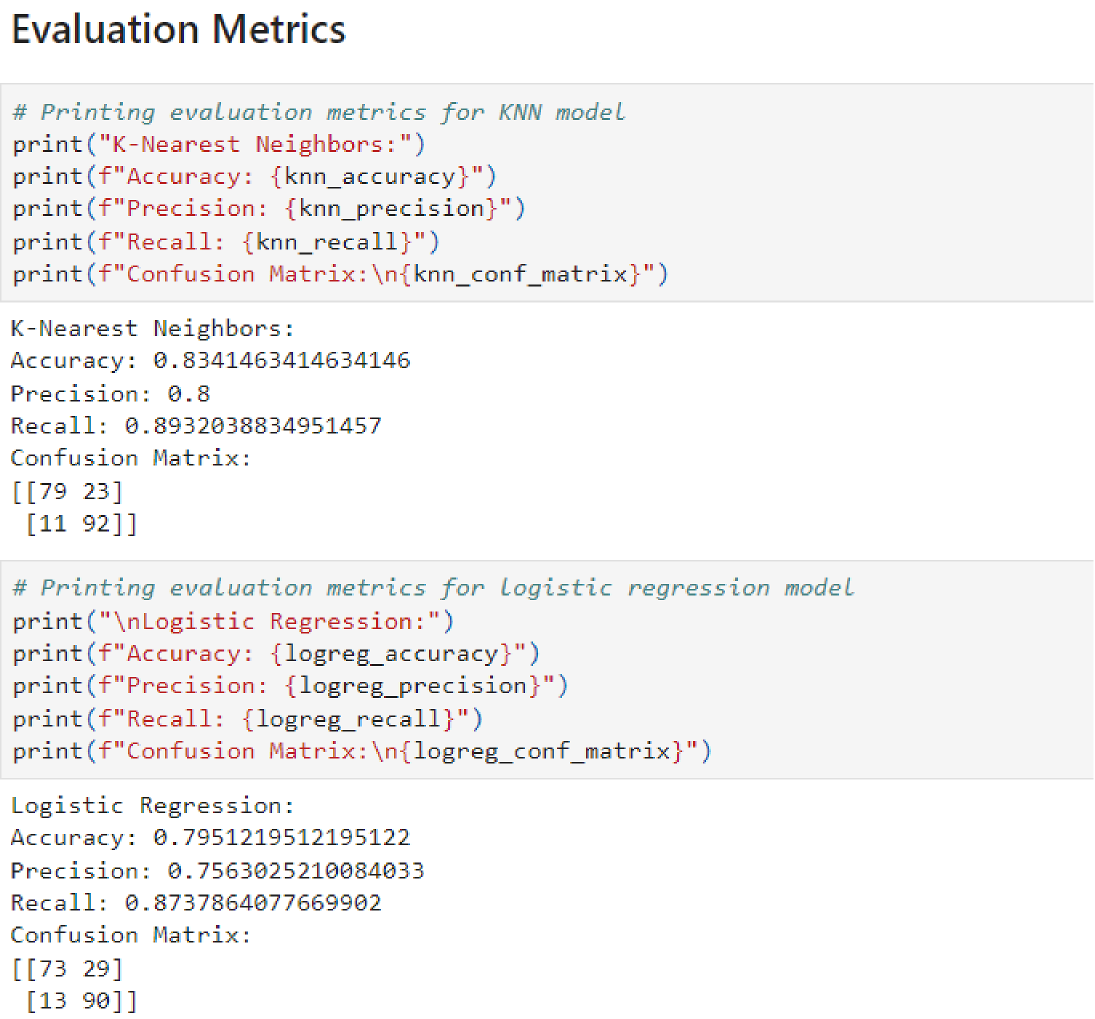


Evaluating Models on Test Data

```
# Defining a function to use for evaluating a model
def evaluate_model(model, X_test, y_test):
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    conf_matrix = confusion_matrix(y_test, y_pred)
    return accuracy, precision, recall, conf_matrix

knn_accuracy, knn_precision, knn_recall, knn_conf_matrix = evaluate_model(knn_model, X_test_scaled, y_test)
logreg_accuracy, logreg_precision, logreg_recall, logreg_conf_matrix = evaluate_model(logreg_model, X_test_scaled, y_test)
rf_accuracy, rf_precision, rf_recall, rf_conf_matrix = evaluate_model(rf_model, X_test, y_test)
```

```
# Cross-validation for model evaluation
knn_cv_scores = cross_val_score(knn_model, X_train_scaled, y_train, cv=5)
logreg_cv_scores = cross_val_score(logreg_model, X_train_scaled, y_train, cv=5)
rf_cv_scores = cross_val_score(rf_model, X_train, y_train, cv=5)
```

Figure 40: Model Evaluation



Evaluation Metrics

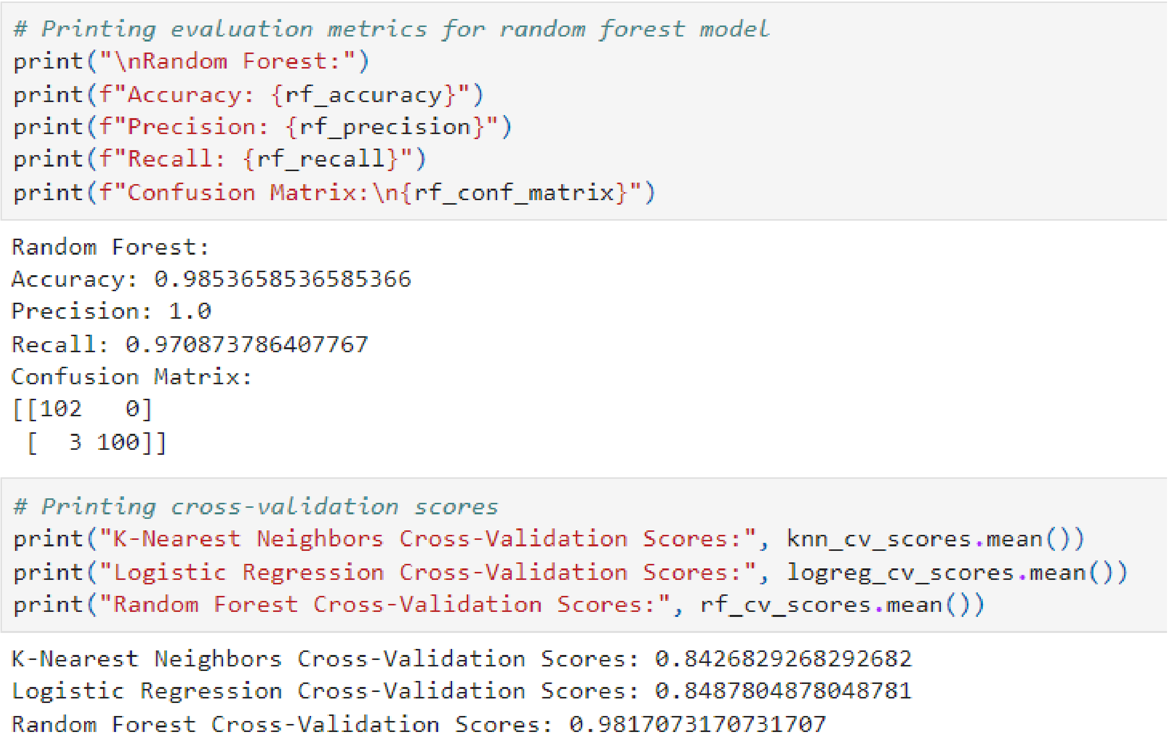
```
# Printing evaluation metrics for KNN model
print("K-Nearest Neighbors:")
print(f"Accuracy: {knn_accuracy}")
print(f"Precision: {knn_precision}")
print(f"Recall: {knn_recall}")
print(f"Confusion Matrix:\n{knn_conf_matrix}")
```

```
K-Nearest Neighbors:
Accuracy: 0.8341463414634146
Precision: 0.8
Recall: 0.8932038834951457
Confusion Matrix:
[[79 23]
 [11 92]]
```

```
# Printing evaluation metrics for logistic regression model
print("\nLogistic Regression:")
print(f"Accuracy: {logreg_accuracy}")
print(f"Precision: {logreg_precision}")
print(f"Recall: {logreg_recall}")
print(f"Confusion Matrix:\n{logreg_conf_matrix}")
```

```
Logistic Regression:
Accuracy: 0.7951219512195122
Precision: 0.7563025210084033
Recall: 0.8737864077669902
Confusion Matrix:
[[73 29]
 [13 90]]
```

Figure 41: Evaluation results 1



```
# Printing evaluation metrics for random forest model
print("\nRandom Forest:")
print(f"Accuracy: {rf_accuracy}")
print(f"Precision: {rf_precision}")
print(f"Recall: {rf_recall}")
print(f"Confusion Matrix:\n{rf_conf_matrix}")
```

```
Random Forest:
Accuracy: 0.9853658536585366
Precision: 1.0
Recall: 0.970873786407767
Confusion Matrix:
[[102   0]
 [  3 100]]
```

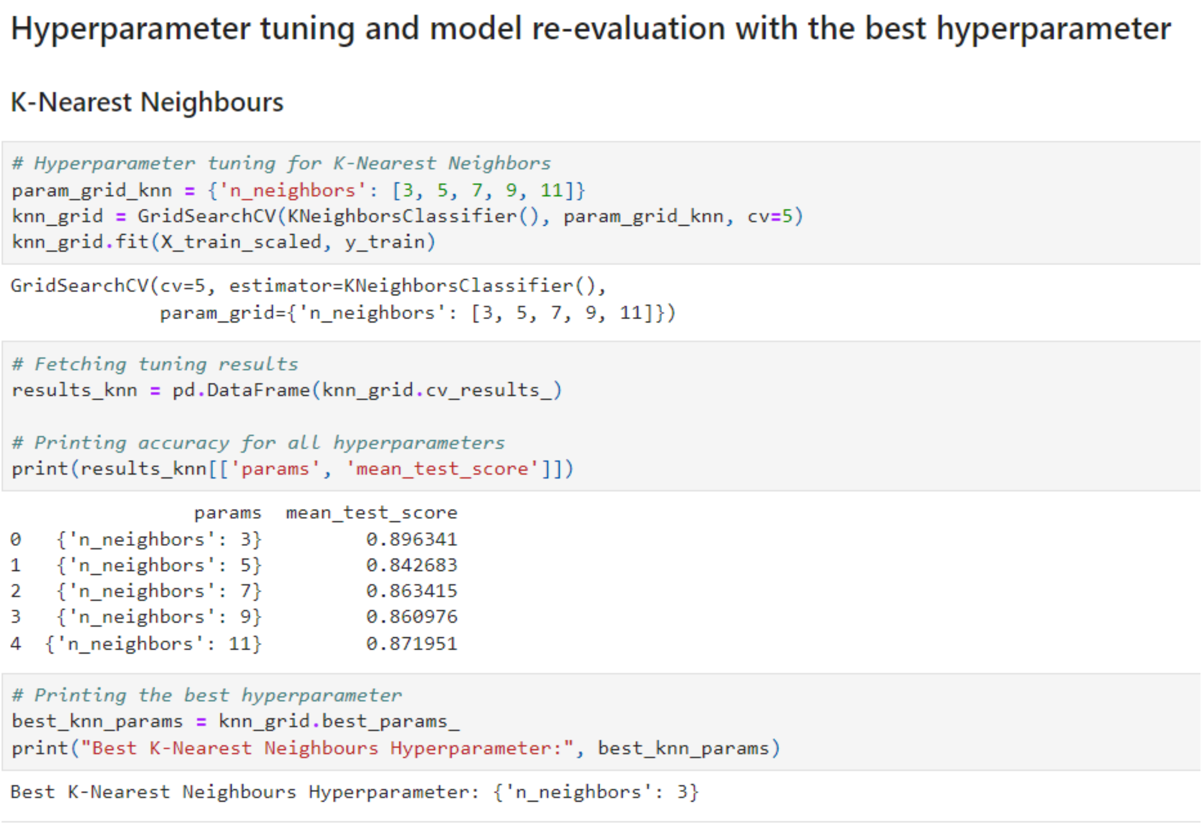
```
# Printing cross-validation scores
print("K-Nearest Neighbors Cross-Validation Scores:", knn_cv_scores.mean())
print("Logistic Regression Cross-Validation Scores:", logreg_cv_scores.mean())
print("Random Forest Cross-Validation Scores:", rf_cv_scores.mean())
```

```
K-Nearest Neighbors Cross-Validation Scores: 0.8426829268292682
Logistic Regression Cross-Validation Scores: 0.8487804878048781
Random Forest Cross-Validation Scores: 0.9817073170731707
```

Figure 42: Evaluation results 2

3.4.2.8.Hyperparameter tuning:

The next stage is hyperparameter tuning, here the models parameters are tuned to achieve the best performing model with the best set of parameters.



Hyperparameter tuning and model re-evaluation with the best hyperparameter

K-Nearest Neighbours

```
# Hyperparameter tuning for K-Nearest Neighbors
param_grid_knn = {'n_neighbors': [3, 5, 7, 9, 11]}
knn_grid = GridSearchCV(KNeighborsClassifier(), param_grid_knn, cv=5)
knn_grid.fit(X_train_scaled, y_train)
```

```
GridSearchCV(cv=5, estimator=KNeighborsClassifier(),
             param_grid={'n_neighbors': [3, 5, 7, 9, 11]})
```

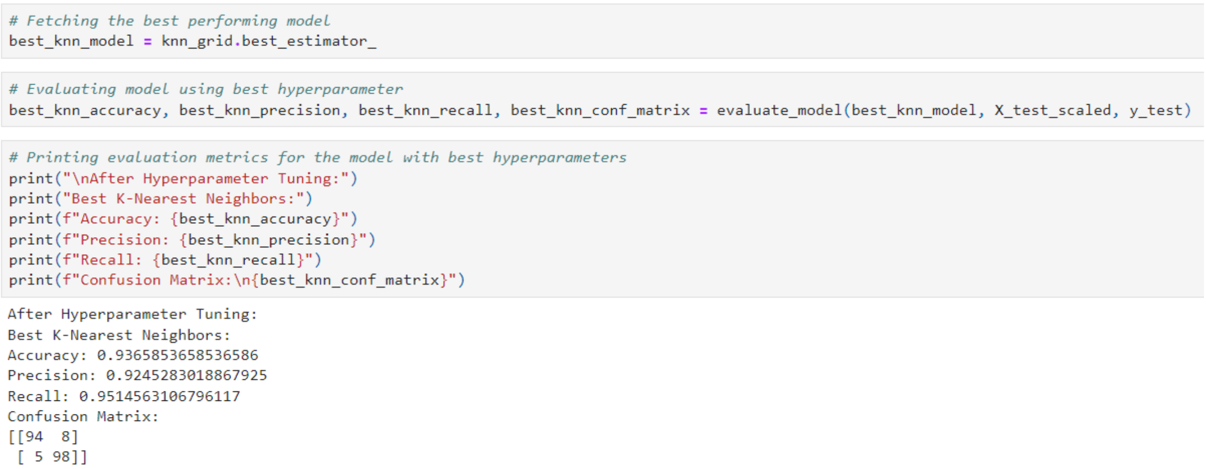
```
# Fetching tuning results
results_knn = pd.DataFrame(knn_grid.cv_results_)

# Printing accuracy for all hyperparameters
print(results_knn[['params', 'mean_test_score']])
```

```
                params  mean_test_score
0   {'n_neighbors': 3}         0.896341
1   {'n_neighbors': 5}         0.842683
2   {'n_neighbors': 7}         0.863415
3   {'n_neighbors': 9}         0.860976
4  {'n_neighbors': 11}         0.871951
```

```
# Printing the best hyperparameter
best_knn_params = knn_grid.best_params_
print("Best K-Nearest Neighbours Hyperparameter:", best_knn_params)
```

```
Best K-Nearest Neighbours Hyperparameter: {'n_neighbors': 3}
```

Figure 43: KNN parameter tuning



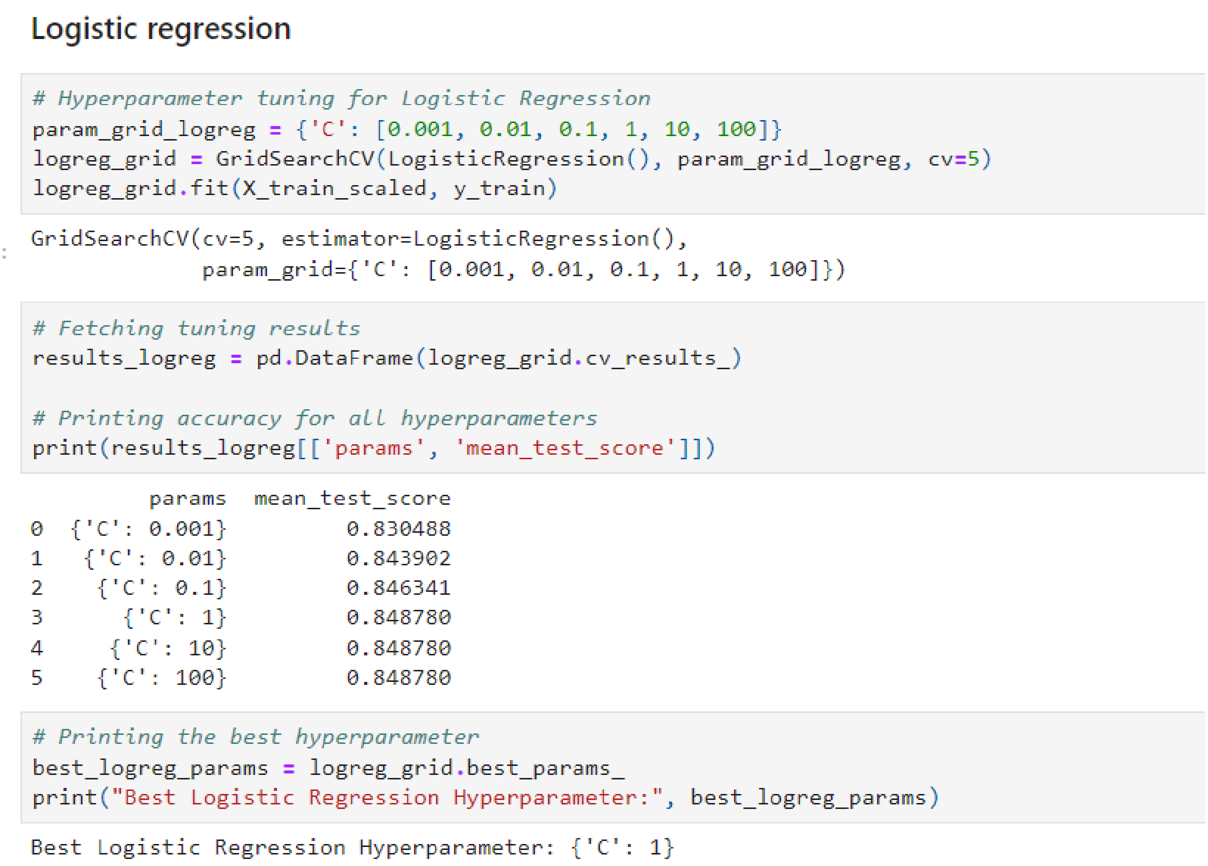
```
# Fetching the best performing model
best_knn_model = knn_grid.best_estimator_
```

```
# Evaluating model using best hyperparameter
best_knn_accuracy, best_knn_precision, best_knn_recall, best_knn_conf_matrix = evaluate_model(best_knn_model, X_test_scaled, y_test)
```

```
# Printing evaluation metrics for the model with best hyperparameters
print("\nAfter Hyperparameter Tuning:")
print("Best K-Nearest Neighbors:")
print(f"Accuracy: {best_knn_accuracy}")
print(f"Precision: {best_knn_precision}")
print(f"Recall: {best_knn_recall}")
print(f"Confusion Matrix:\n{best_knn_conf_matrix}")
```

```
After Hyperparameter Tuning:
Best K-Nearest Neighbors:
Accuracy: 0.9365853658536586
Precision: 0.9245283018867925
Recall: 0.9514563106796117
Confusion Matrix:
[[94  8]
 [ 5 98]]
```

Figure 44: Re-evaluation after parameter tuning: KNN



Logistic regression

```
# Hyperparameter tuning for Logistic Regression
param_grid_logreg = {'C': [0.001, 0.01, 0.1, 1, 10, 100]}
logreg_grid = GridSearchCV(LogisticRegression(), param_grid_logreg, cv=5)
logreg_grid.fit(X_train_scaled, y_train)
```

```
GridSearchCV(cv=5, estimator=LogisticRegression(),
             param_grid={'C': [0.001, 0.01, 0.1, 1, 10, 100]})
```

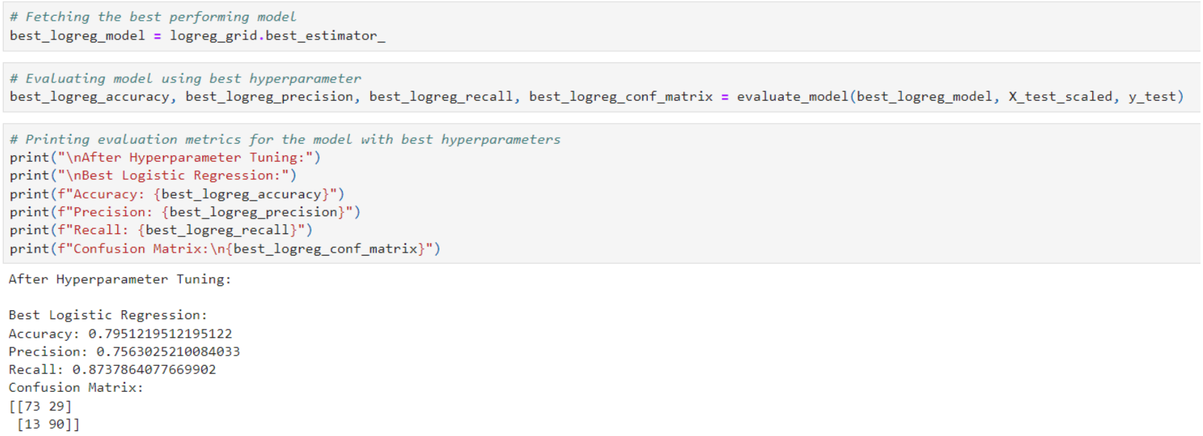
```
# Fetching tuning results
results_logreg = pd.DataFrame(logreg_grid.cv_results_)

# Printing accuracy for all hyperparameters
print(results_logreg[['params', 'mean_test_score']])
```

```
         params  mean_test_score
0  {'C': 0.001}         0.830488
1   {'C': 0.01}         0.843902
2    {'C': 0.1}         0.846341
3      {'C': 1}         0.848780
4     {'C': 10}         0.848780
5    {'C': 100}         0.848780
```

```
# Printing the best hyperparameter
best_logreg_params = logreg_grid.best_params_
print("Best Logistic Regression Hyperparameter:", best_logreg_params)
```

```
Best Logistic Regression Hyperparameter: {'C': 1}
```

Figure 45: Logistic Regression parameter tuning



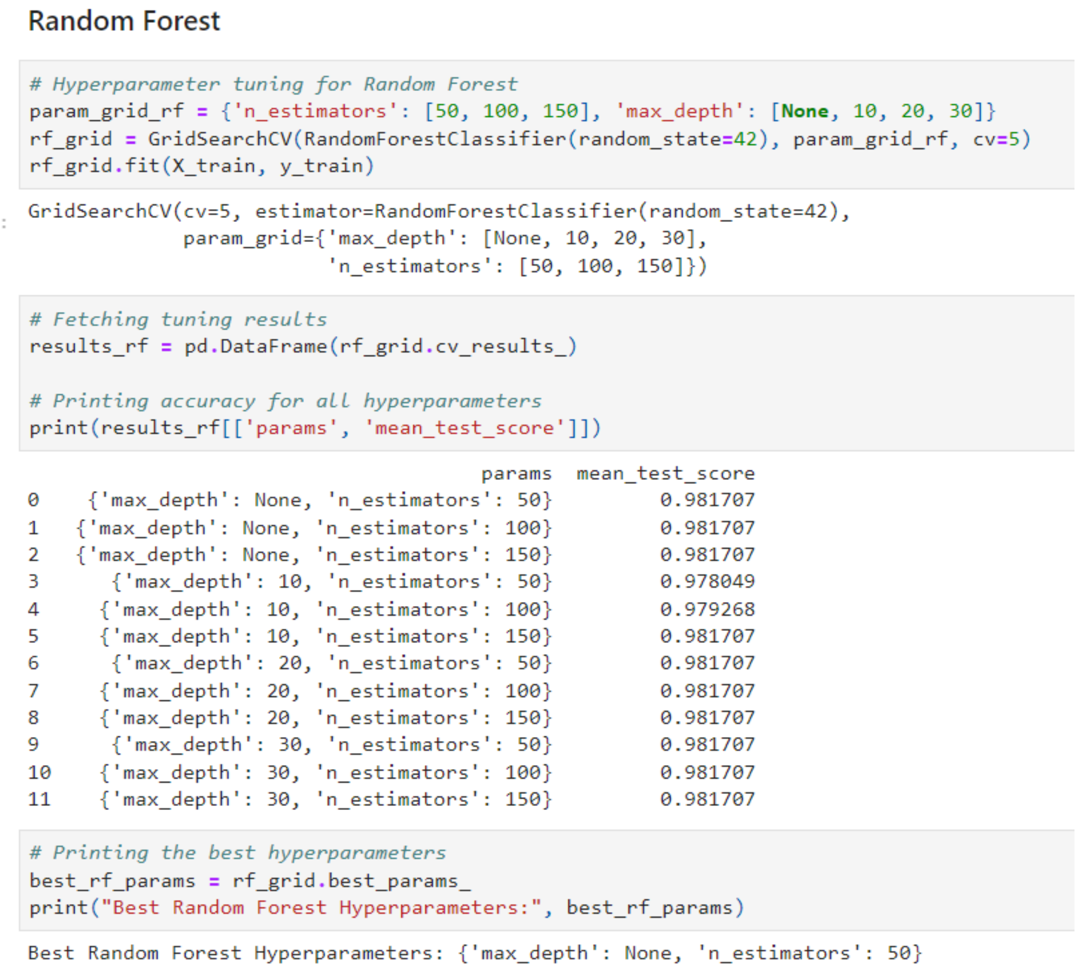
```
# Fetching the best performing model
best_logreg_model = logreg_grid.best_estimator_
```

```
# Evaluating model using best hyperparameter
best_logreg_accuracy, best_logreg_precision, best_logreg_recall, best_logreg_conf_matrix = evaluate_model(best_logreg_model, X_test_scaled, y_test)
```

```
# Printing evaluation metrics for the model with best hyperparameters
print("\nAfter Hyperparameter Tuning:")
print("\nBest Logistic Regression:")
print(f"Accuracy: {best_logreg_accuracy}")
print(f"Precision: {best_logreg_precision}")
print(f"Recall: {best_logreg_recall}")
print(f"Confusion Matrix:\n{best_logreg_conf_matrix}")
```

```
After Hyperparameter Tuning:

Best Logistic Regression:
Accuracy: 0.7951219512195122
Precision: 0.7563025210084033
Recall: 0.8737864077669902
Confusion Matrix:
[[73 29]
 [13 90]]
```

Figure 46: Re-evaluation after parameter tuning: logistic regression



Random Forest

```
# Hyperparameter tuning for Random Forest
param_grid_rf = {'n_estimators': [50, 100, 150], 'max_depth': [None, 10, 20, 30]}
rf_grid = GridSearchCV(RandomForestClassifier(random_state=42), param_grid_rf, cv=5)
rf_grid.fit(X_train, y_train)
```

```
GridSearchCV(cv=5, estimator=RandomForestClassifier(random_state=42),
             param_grid={'max_depth': [None, 10, 20, 30],
                         'n_estimators': [50, 100, 150]})
```

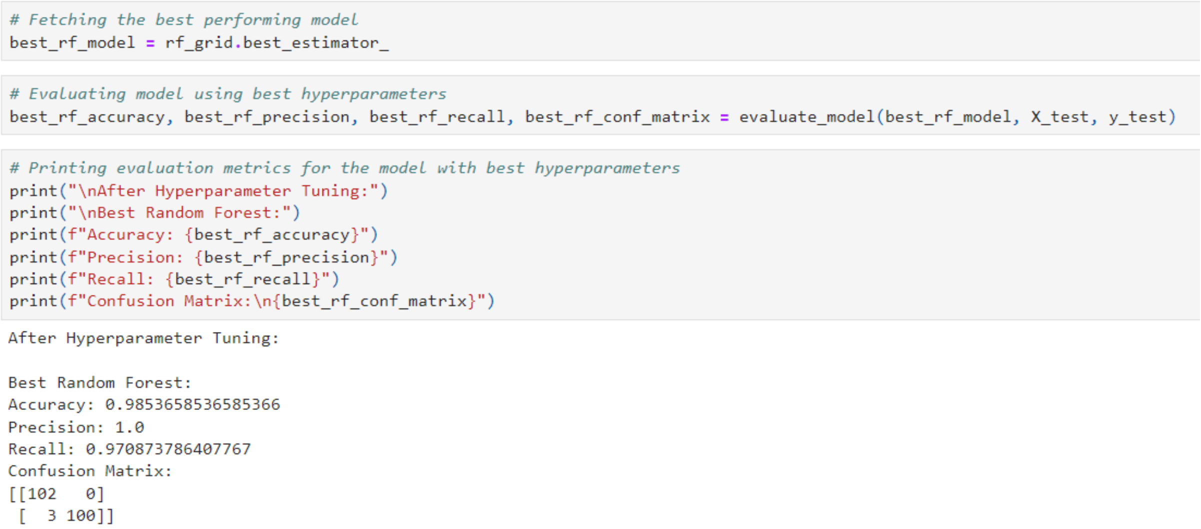
```
# Fetching tuning results
results_rf = pd.DataFrame(rf_grid.cv_results_)

# Printing accuracy for all hyperparameters
print(results_rf[['params', 'mean_test_score']])
```

```
                                      params  mean_test_score
0    {'max_depth': None, 'n_estimators': 50}         0.981707
1   {'max_depth': None, 'n_estimators': 100}         0.981707
2   {'max_depth': None, 'n_estimators': 150}         0.981707
3      {'max_depth': 10, 'n_estimators': 50}         0.978049
4     {'max_depth': 10, 'n_estimators': 100}         0.979268
5     {'max_depth': 10, 'n_estimators': 150}         0.981707
6      {'max_depth': 20, 'n_estimators': 50}         0.981707
7     {'max_depth': 20, 'n_estimators': 100}         0.981707
8     {'max_depth': 20, 'n_estimators': 150}         0.981707
9      {'max_depth': 30, 'n_estimators': 50}         0.981707
10    {'max_depth': 30, 'n_estimators': 100}         0.981707
11    {'max_depth': 30, 'n_estimators': 150}         0.981707
```

```
# Printing the best hyperparameters
best_rf_params = rf_grid.best_params_
print("Best Random Forest Hyperparameters:", best_rf_params)
```

```
Best Random Forest Hyperparameters: {'max_depth': None, 'n_estimators': 50}
```

Figure 47: Random Forest parameters tuning



```
# Fetching the best performing model
best_rf_model = rf_grid.best_estimator_
```

```
# Evaluating model using best hyperparameters
best_rf_accuracy, best_rf_precision, best_rf_recall, best_rf_conf_matrix = evaluate_model(best_rf_model, X_test, y_test)
```

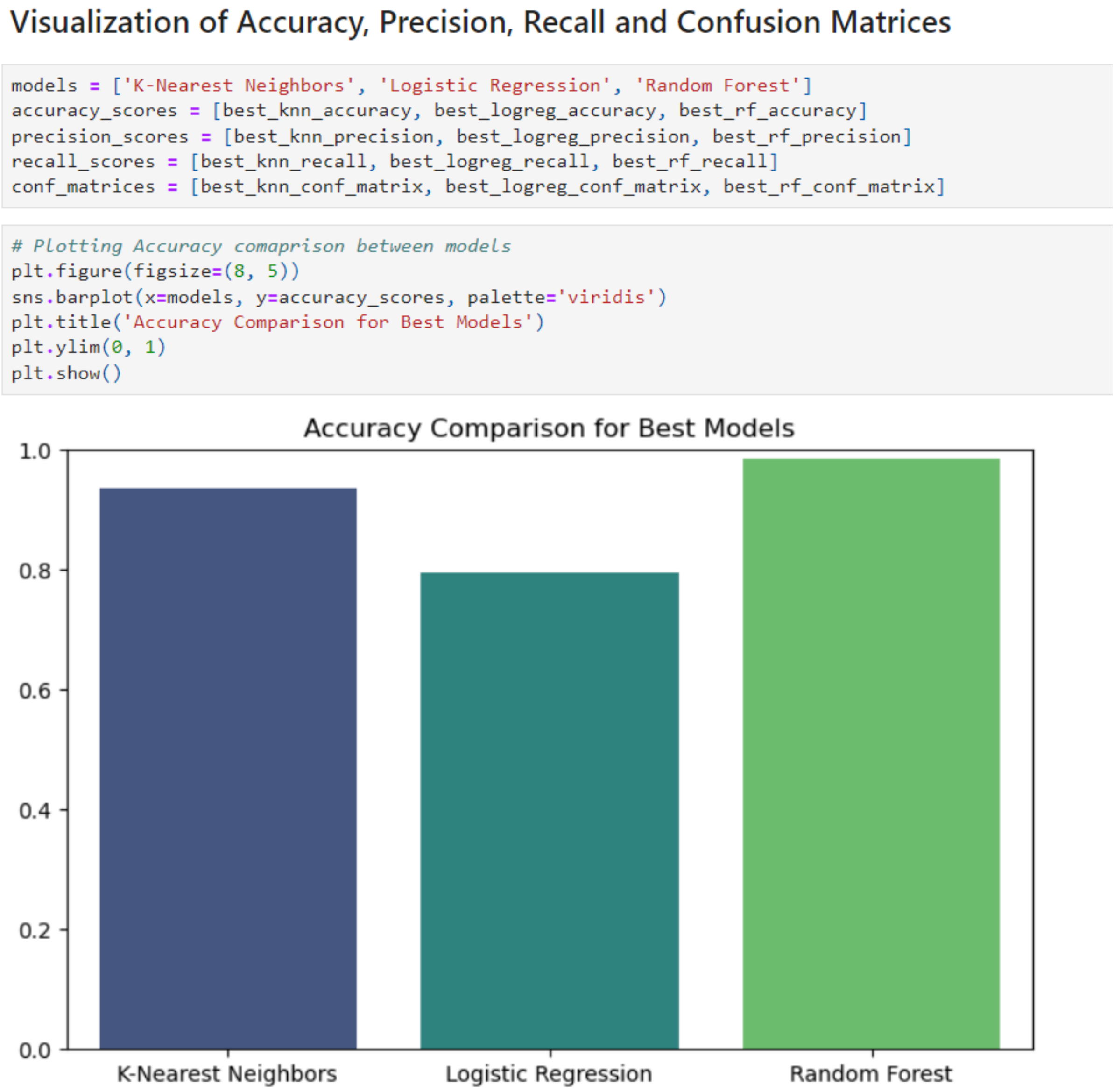
```
# Printing evaluation metrics for the model with best hyperparameters
print("\nAfter Hyperparameter Tuning:")
print("\nBest Random Forest:")
print(f"Accuracy: {best_rf_accuracy}")
print(f"Precision: {best_rf_precision}")
print(f"Recall: {best_rf_recall}")
print(f"Confusion Matrix:\n{best_rf_conf_matrix}")
```

```
After Hyperparameter Tuning:

Best Random Forest:
Accuracy: 0.9853658536585366
Precision: 1.0
Recall: 0.970873786407767
Confusion Matrix:
[[102   0]
 [  3 100]]
```

Figure 48: Re-evaluation after parameters tuning: Random Forest

3.4.2.9. Visualization of evaluation metrics and comparison between models:

The next stage is to visualize the evaluation metrics and compare between the model's using graphs and plots.

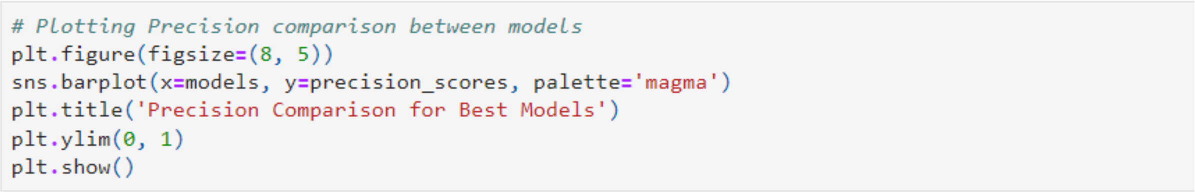


Visualization of Accuracy, Precision, Recall and Confusion Matrices

```
models = ['K-Nearest Neighbors', 'Logistic Regression', 'Random Forest']
accuracy_scores = [best_knn_accuracy, best_logreg_accuracy, best_rf_accuracy]
precision_scores = [best_knn_precision, best_logreg_precision, best_rf_precision]
recall_scores = [best_knn_recall, best_logreg_recall, best_rf_recall]
conf_matrices = [best_knn_conf_matrix, best_logreg_conf_matrix, best_rf_conf_matrix]
```

```
# Plotting Accuracy comaprison between models
plt.figure(figsize=(8, 5))
sns.barplot(x=models, y=accuracy_scores, palette='viridis')
plt.title('Accuracy Comparison for Best Models')
plt.ylim(0, 1)
plt.show()
```

Figure 49: Visualization of accuracy comparison of all models



```
# Plotting Precision comparison between models
plt.figure(figsize=(8, 5))
sns.barplot(x=models, y=precision_scores, palette='magma')
plt.title('Precision Comparison for Best Models')
plt.ylim(0, 1)
plt.show()
```

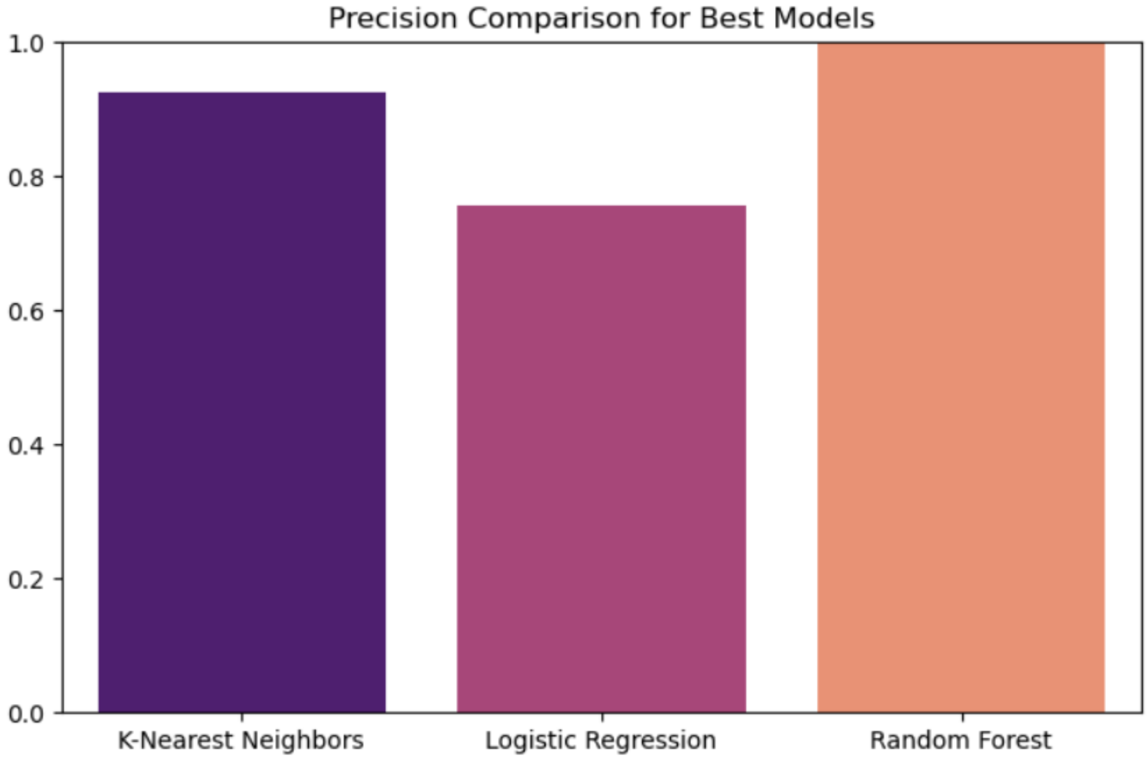
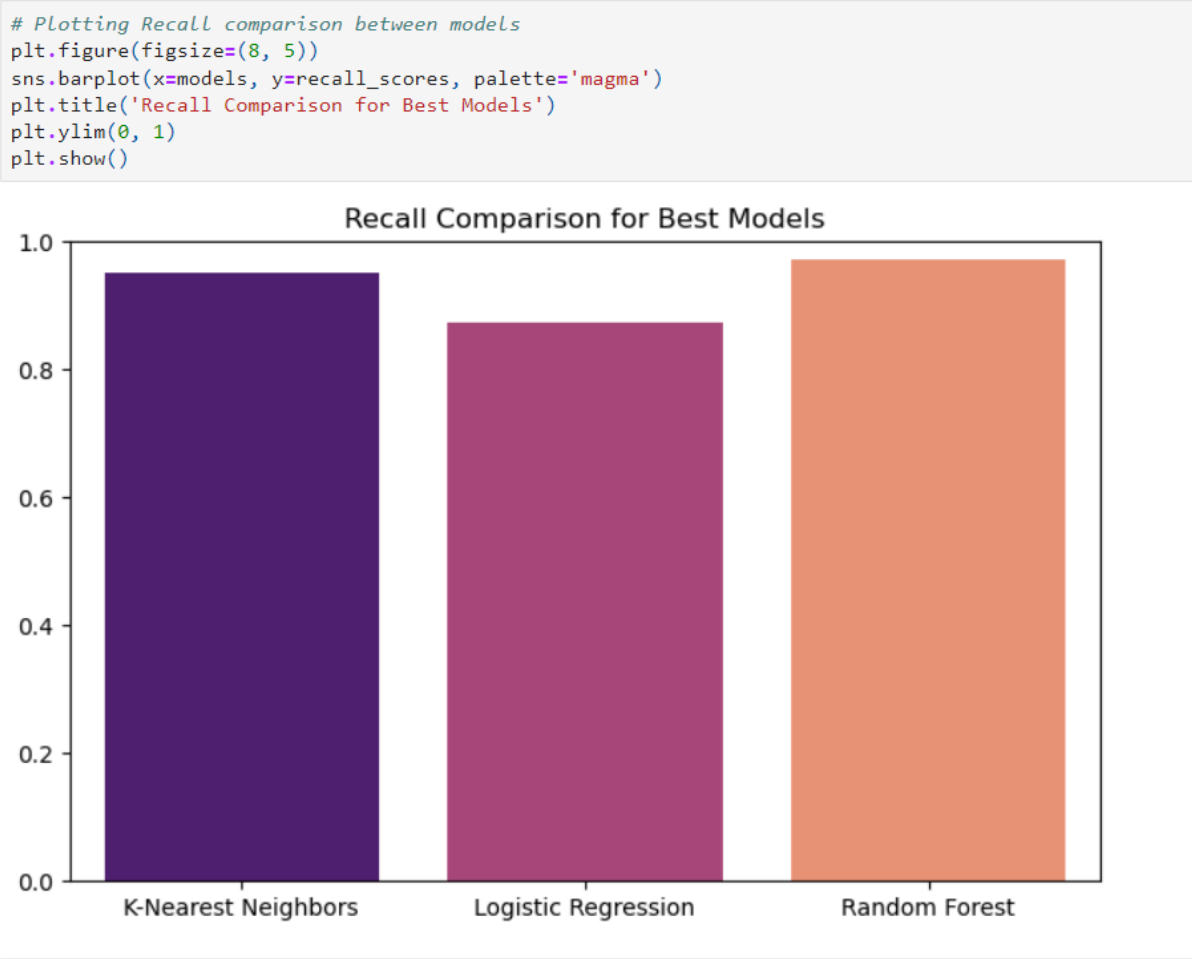
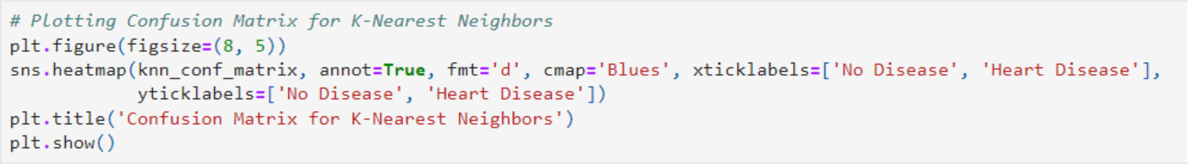


Figure 50: Visualization of precision comparison of all models



```
# Plotting Recall comparison between models
plt.figure(figsize=(8, 5))
sns.barplot(x=models, y=recall_scores, palette='magma')
plt.title('Recall Comparison for Best Models')
plt.ylim(0, 1)
plt.show()
```

Figure 51: Visualization of Recall comparison of all models



```
# Plotting Confusion Matrix for K-Nearest Neighbors
plt.figure(figsize=(8, 5))
sns.heatmap(knn_conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['No Disease', 'Heart Disease'],
            yticklabels=['No Disease', 'Heart Disease'])
plt.title('Confusion Matrix for K-Nearest Neighbors')
plt.show()
```

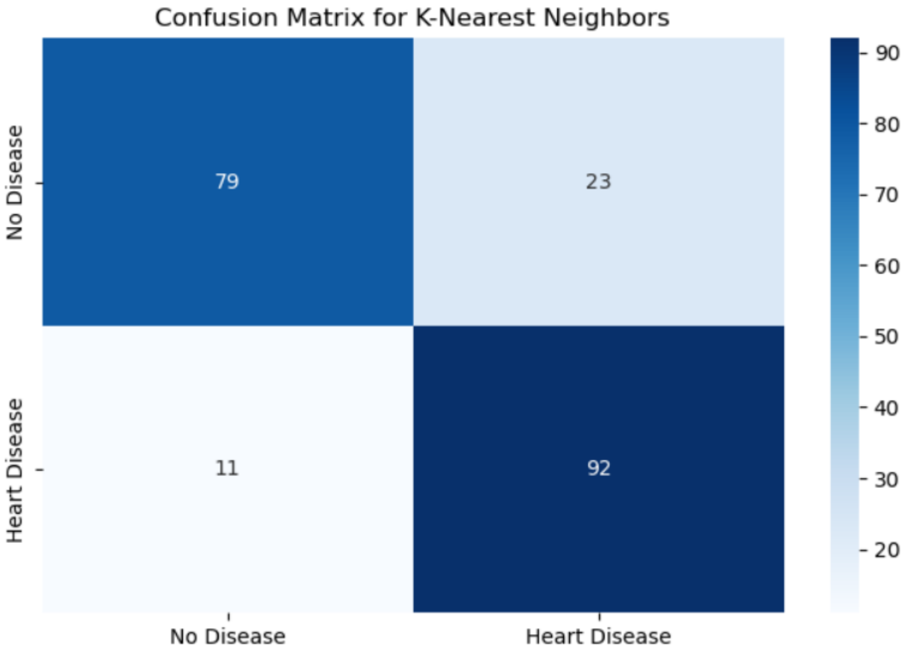
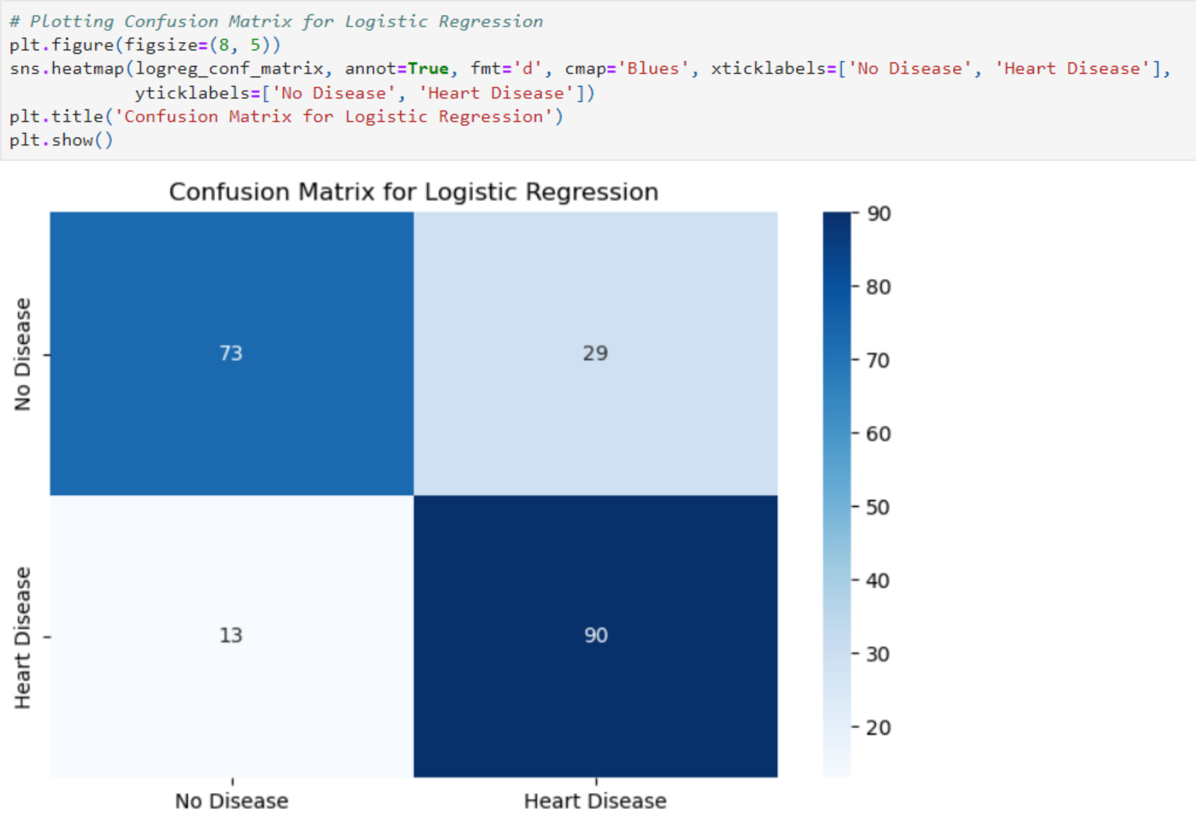
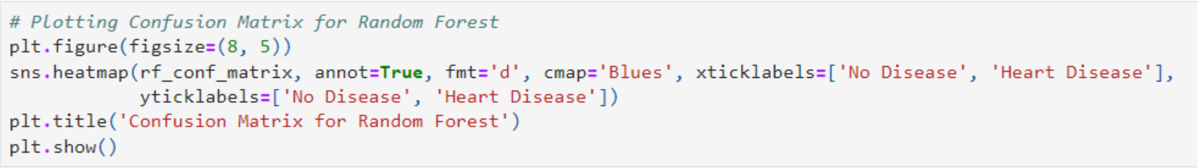


Figure 52: Confusion matrix: KNN



```
# Plotting Confusion Matrix for Logistic Regression
plt.figure(figsize=(8, 5))
sns.heatmap(logreg_conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['No Disease', 'Heart Disease'],
            yticklabels=['No Disease', 'Heart Disease'])
plt.title('Confusion Matrix for Logistic Regression')
plt.show()
```

Figure 53: Confusion matrix: Logistic regression



```
# Plotting Confusion Matrix for Random Forest
plt.figure(figsize=(8, 5))
sns.heatmap(rf_conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=['No Disease', 'Heart Disease'],
            yticklabels=['No Disease', 'Heart Disease'])
plt.title('Confusion Matrix for Random Forest')
plt.show()
```

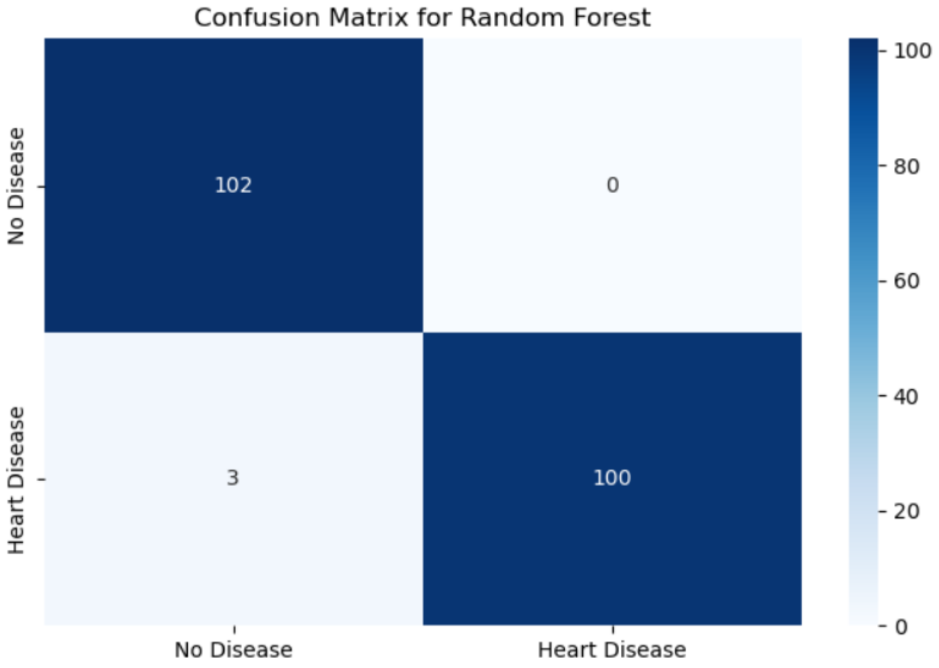
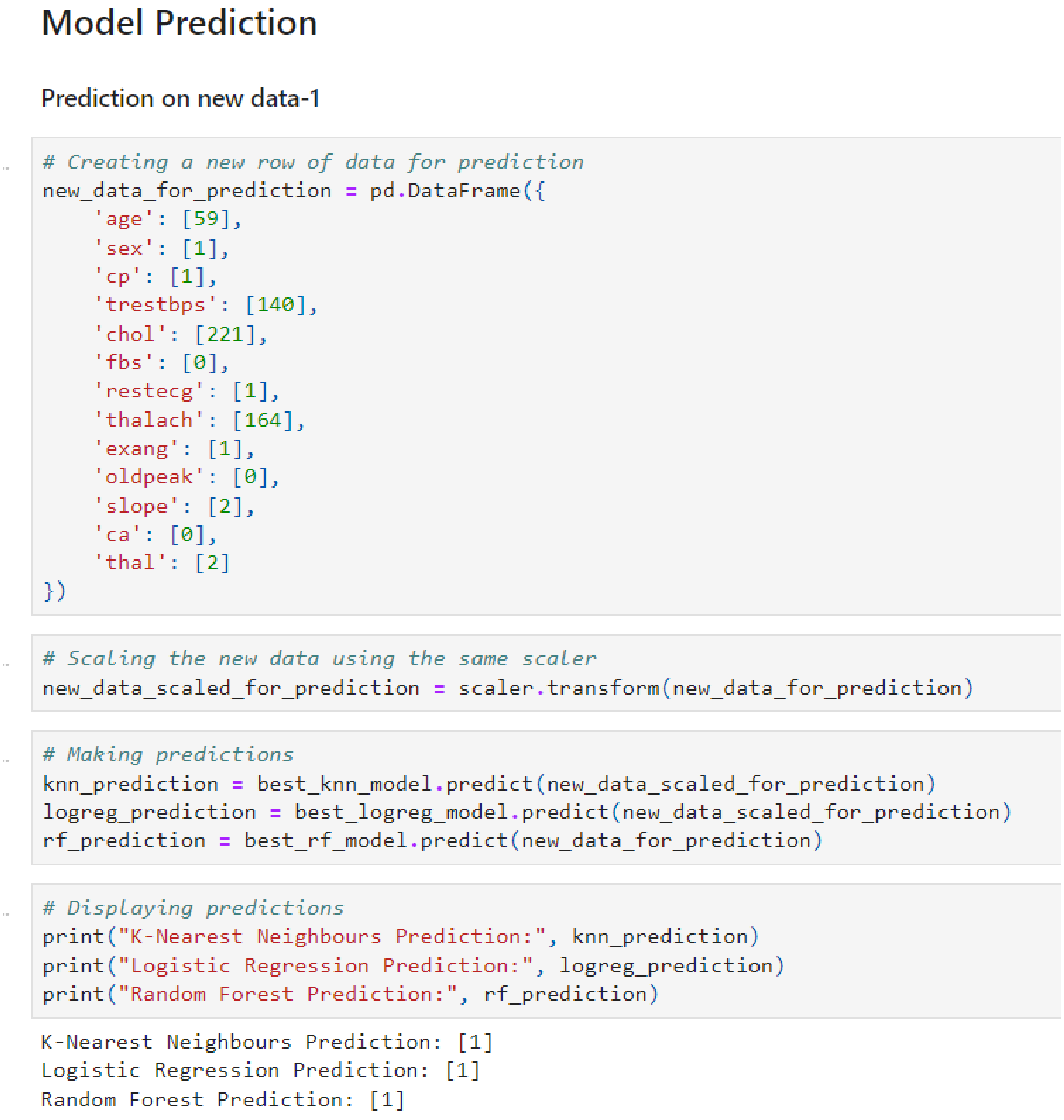


Figure 54: Confusion matrix: Random forest

3.4.2.10. Making predictions:

The final stage of this entire process of building a working model that predicts heart disease is to make predictions on a completely new data.



Model Prediction

Prediction on new data-1

```
# Creating a new row of data for prediction
new_data_for_prediction = pd.DataFrame({
    'age': [59],
    'sex': [1],
    'cp': [1],
    'trestbps': [140],
    'chol': [221],
    'fbs': [0],
    'restecg': [1],
    'thalach': [164],
    'exang': [1],
    'oldpeak': [0],
    'slope': [2],
    'ca': [0],
    'thal': [2]
})
```

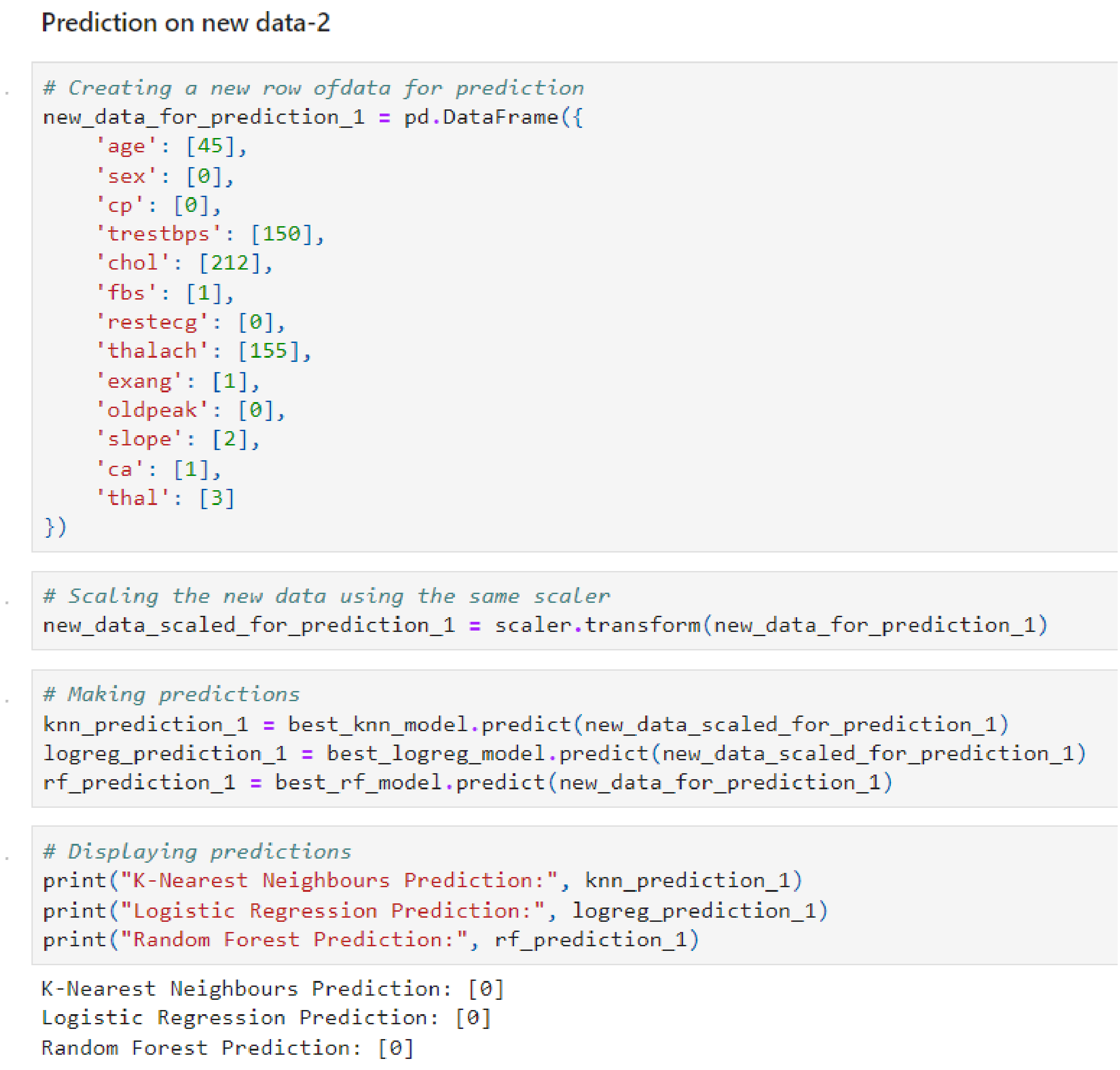
```
# Scaling the new data using the same scaler
new_data_scaled_for_prediction = scaler.transform(new_data_for_prediction)
```

```
# Making predictions
knn_prediction = best_knn_model.predict(new_data_scaled_for_prediction)
logreg_prediction = best_logreg_model.predict(new_data_scaled_for_prediction)
rf_prediction = best_rf_model.predict(new_data_for_prediction)
```

```
# Displaying predictions
print("K-Nearest Neighbours Prediction:", knn_prediction)
print("Logistic Regression Prediction:", logreg_prediction)
print("Random Forest Prediction:", rf_prediction)
```

```
K-Nearest Neighbours Prediction: [1]
Logistic Regression Prediction: [1]
Random Forest Prediction: [1]
```

Figure 55: Model prediction 1



Prediction on new data-2

```
# Creating a new row ofdata for prediction
new_data_for_prediction_1 = pd.DataFrame({
    'age': [45],
    'sex': [0],
    'cp': [0],
    'trestbps': [150],
    'chol': [212],
    'fbs': [1],
    'restecg': [0],
    'thalach': [155],
    'exang': [1],
    'oldpeak': [0],
    'slope': [2],
    'ca': [1],
    'thal': [3]
})
```

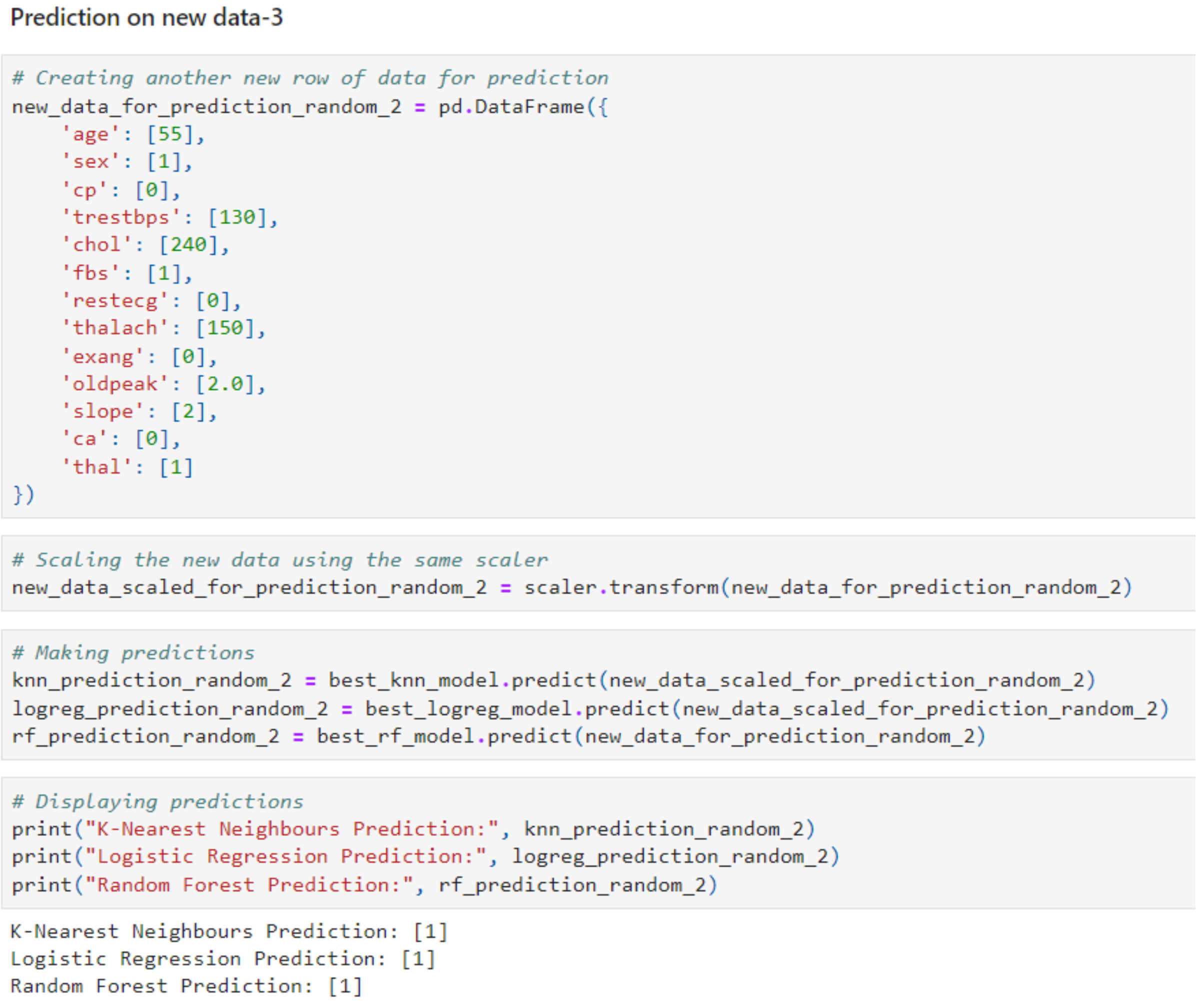
```
# Scaling the new data using the same scaler
new_data_scaled_for_prediction_1 = scaler.transform(new_data_for_prediction_1)
```

```
# Making predictions
knn_prediction_1 = best_knn_model.predict(new_data_scaled_for_prediction_1)
logreg_prediction_1 = best_logreg_model.predict(new_data_scaled_for_prediction_1)
rf_prediction_1 = best_rf_model.predict(new_data_for_prediction_1)
```

```
# Displaying predictions
print("K-Nearest Neighbours Prediction:", knn_prediction_1)
print("Logistic Regression Prediction:", logreg_prediction_1)
print("Random Forest Prediction:", rf_prediction_1)
```

```
K-Nearest Neighbours Prediction: [0]
Logistic Regression Prediction: [0]
Random Forest Prediction: [0]
```

Figure 56: Model prediction 2



Prediction on new data-3

```
# Creating another new row of data for prediction
new_data_for_prediction_random_2 = pd.DataFrame({
    'age': [55],
    'sex': [1],
    'cp': [0],
    'trestbps': [130],
    'chol': [240],
    'fbs': [1],
    'restecg': [0],
    'thalach': [150],
    'exang': [0],
    'oldpeak': [2.0],
    'slope': [2],
    'ca': [0],
    'thal': [1]
})
```

```
# Scaling the new data using the same scaler
new_data_scaled_for_prediction_random_2 = scaler.transform(new_data_for_prediction_random_2)
```

```
# Making predictions
knn_prediction_random_2 = best_knn_model.predict(new_data_scaled_for_prediction_random_2)
logreg_prediction_random_2 = best_logreg_model.predict(new_data_scaled_for_prediction_random_2)
rf_prediction_random_2 = best_rf_model.predict(new_data_for_prediction_random_2)
```

```
# Displaying predictions
print("K-Nearest Neighbours Prediction:", knn_prediction_random_2)
print("Logistic Regression Prediction:", logreg_prediction_random_2)
print("Random Forest Prediction:", rf_prediction_random_2)
```

```
K-Nearest Neighbours Prediction: [1]
Logistic Regression Prediction: [1]
Random Forest Prediction: [1]
```

Figure 57: Model prediction 3

3.4.3. Results:

In this project the model evaluation is done two times, one before hyperparameter tuning and one after the hyperparameter tuning. In the first evaluation, the KNN model achieved an accuracy of 0.83, precision of 0.8 and recall of 0.89, the logistic regression model achieved an accuracy of 0.8, precision of 0.75 and recall of 0.87. Similarly, the random forest achieved an accuracy of 0.98, a precision of 1.00, and a recall of 0.97 which conludes that the random forest model is the best performing model out of the three models. After the hyperparameter tuning, another evaluation was done with the best hyperparameters, where the KNN model achieved an accuracy of 0.93, precision of 0.92 and recall of 0.95 which is a drastic improvement compared to the evaluation before hyperparameter tuning, , the logistic regression model on the other hand achieved an accuracy of 0.8, precision of 0.75 and recall of 0.87 which is the same as before. In the similar way, the random forest achieved an accuracy of 0.98, a precision of 1.00, and a recall of 0.97 which is also unchanged as before. Thus, out of all the three models, random forest classification model turned out to be the best performing model with an accuracy of 0.98.

4. Conclusion:

4.1. Analysis of the work done:

In this, project, we have successfully created a working model that can predict heart disease in an individual using certain health metrics. The model was developed particularly using three algorithms KNN, Logistic Regression, and Random Forest Classifier. We have also created flowcharts, pseudocode for better understanding of the system and program. The process was also explained in detail to help others understand the project and the process behind achieving this project.

4.2. How the solution addresses real-world problems:

As mentioned earlier, the number of cardiovascular incidences has been on the rise due to multiple factors. So, many people have been facing untimely demise due to unexpected heart-related complications which potentially could have been managed if detected early. With this project, early identification of heart-related complications is possible which can further help people to detect and control cardiovascular problems to prevent any further complications. Similarly, traditional diagnostic methods are falling short of providing timely and accurate heart health predictions, which can also be assisted by this project.

4.3. Limitations:

Although the model has a very high accuracy, it still has lots of limitations. This model if utilized in the real world, the people would have absolutely no faith in a machine predicting their health. Developing trust from the people is great limitation of this model, especially because people are more doubtful when it comes to the matter of health. Similarly, this model has other limitations, such as security limitations, as the model is completely dependent upon the dataset, so if the dataset is corrupted or changed by anyone, the predictions are drastically hampered.

4.4. Future improvements:

The model achieved a great success in terms of the performance achieving an accuracy of 0.98. Although it achieved a great performance score, it still is incomplete. So, in the future many improvements can be brought to thus working model. One of them maybe creating a interface for easy use, so that anyone with little to less knowledge can operate this predictive model.

5. Bibliography:

Beckerman, J. (2023, January 23). Retrieved from https://www.webmd.com/heart-disease/heart-failure/early-diagnosis-heart-failure

Bhatt, C. M. (2023, February 6). *MDPI*. Retrieved from https://www.mdpi.com/1999-4893/16/2/88#:~:text=Many%20factors%2C%20such%20as%20diabetes,role%20in%20the%20medical%20field.

BMC Medical Education. (2023, September 22). Retrieved from https://bmcmededuc.biomedcentral.com/articles/10.1186/s12909-023-04698-z

Cleveland Clinic. (2023). Retrieved from https://my.clevelandclinic.org/health/articles/17085-heart-risk-factor-calculators

Copeland, B. (2023, December 18). Retrieved from https://www.britannica.com/technology/artificial-intelligence/Reasoning

Coursera. (2023, November 29). Retrieved from https://www.coursera.org/articles/machine-learning-in-health-care

Donovan, R. (2023, July 7). Retrieved from https://www.healthline.com/health/heart-disease#Who-gets-heart-disease?

geeksforgeeks. (2023). Retrieved from https://www.geeksforgeeks.org/advantages-and-disadvantages-of-logistic-regression/

Heartfoundation. (2023). Retrieved from https://www.heartfoundation.org.au/bundles/for-professionals/key-statistics-heart-attack

IBM. (2023). Retrieved from https://www.ibm.com/topics/machine-learning

Javatpoint. (2023). Retrieved from https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning

JupyterLab. (2023). Retrieved from https://jupyterlab.readthedocs.io/en/stable/

Kanade, V. (2022, April 18). Retrieved from https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-logistic-regression/

MLNerds. (2019). Retrieved from https://machinelearninginterview.com/topics/machine-learning/how-does-knn-algorithm-work-what-are-the-advantages-and-disadvantages-of-knn/

Nay, J. (2023, October 30). Retrieved from https://jeremybney.medium.com/why-smoking-rates-are-rising-again-9de865c58107

Simplilearn. (2023, August 21). Retrieved from https://www.simplilearn.com/tutorials/machine-learning-tutorial/machine-learning-steps#:~:text=Machine%20learning%20is%20the%20process,that%20data%20to%20learn%20more.

Singh, J. (2020, December 18). Retrieved from Medium: https://medium.datadriveninvestor.com/random-forest-pros-and-cons-c1c42fb64f04

Solulab. (2023). Retrieved from https://www.solulab.com/future-of-ai-in-healthcare/

Srivastav, T. (2023, October 20). Retrieved from https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/#h-what-is-knn-k-nearest-neighbor-algorithm

Thakur, M. (2023). Retrieved from https://www.educba.com/advantages-and-disadvantages-of-machine-learning/

TheHindu. (2023, December 02). Retrieved from https://www.thehindu.com/news/national/other-states/over-1000-died-of-heart-attack-in-six-months-gujarat-minister/article67597167.ece

Turing. (2023). Retrieved from https://www.turing.com/kb/random-forest-algorithm#what-is-random-forest-algorithm?