A Project Report
On

Predictive Analytics for Heart Disease Detection

Course: BUAN6340.001 – Programming for Data Science – F24
Submitted by: Ashlesha Sanjay Kadam
NET ID: AXK220237

Table of Contents

Executive Summary

In the quest to enhance early detection and intervention strategies for heart disease, our project has successfully leveraged machine learning techniques to develop a robust predictive model. Utilizing a comprehensive dataset from the UCI Machine Learning Repository, which aggregates data from varied clinical environments such as Cleveland, Hungary, Switzerland, and VA Long Beach, this initiative has marked a significant advancement in predictive health analytics.

The cornerstone of our project was the application of logistic regression, a method chosen for its proven efficacy in binary classification tasks. This model not only demonstrated a high degree of accuracy, achieving 85% on our test sets, but also showcased superior performance when compared against other sophisticated models, including RandomForest, XGBoost, SVM, and an enhanced version of XGBoost.

Our methodology encompassed extensive data preprocessing to address and rectify issues such as missing values and standardize feature scales across the dataset. Techniques such as median imputation were employed to handle missing entries effectively, ensuring the integrity and utility of our data. Additionally, categorical variables were meticulously one-hot encoded to better fit the logistic regression framework.

The exploratory phase of our analysis revealed critical insights, which were instrumental in refining our model and selecting impactful features. This detailed exploration aided in understanding the complex interactions and distributions of the variables involved, thereby informing our predictive modeling strategy.

Subsequent validation through a rigorous 10-fold cross-validation process affirmed the stability and consistency of our logistic regression model, outperforming other models tested in parallel. This validation confirmed the logistic regression approach's robustness and highlighted its potential as a reliable tool in clinical settings.

The success of this project underscores the potential of machine learning in revolutionizing diagnostic processes, particularly in the detection of heart disease. By continuing to refine our approach and integrate more comprehensive data, we can significantly improve patient outcomes and reduce heart disease-related mortality. This predictive model is a promising tool in the healthcare industry’s ongoing efforts to enhance diagnostic accuracy and patient care.

1.0. Introduction

Heart disease remains the foremost cause of mortality worldwide, representing a major public health challenge. The ability to predict heart disease early significantly enhances treatment outcomes and can substantially reduce the mortality rate. In this context, predictive analytics emerges as a powerful tool, providing insights that help identify individuals at high risk before the onset of more severe complications.

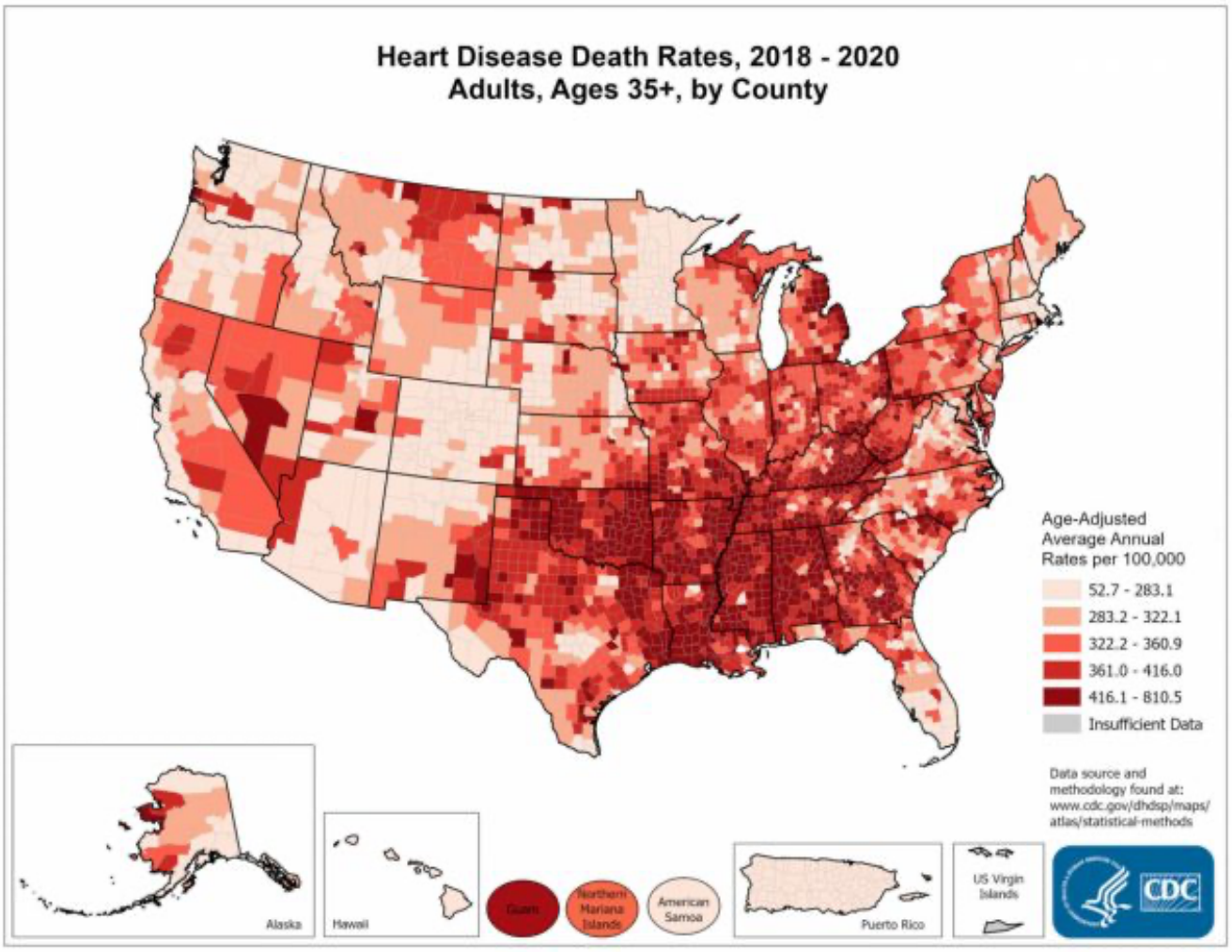


Figure 1. Heart Disease Death Rates among adults in the United States, 2018-2020

This map illustrates the heart disease death rates for adults aged 35 and older in the United States, spanning the years 2018 to 2020. The data, presented by county, shows varying death rates color-coded from light pink (lower rates) to dark red (higher rates), highlighting significant geographical disparities in heart disease mortality across the country. Data sources include the CDC, and the map employs age-adjusted rates to allow for fair comparisons between counties with different age demographics.

Prevalence and Mortality:
Heart disease is the leading cause of death globally, accounting for more than 17.9 million deaths annually, which represents 31% of all global deaths (Source: World Health Organization). Approximately 85% of these deaths are due to heart attacks and strokes.

Economic Impact:
The global economic impact of cardiovascular diseases is substantial, with an estimated cost of about $863 billion in 2010, projected to rise to $1,044 billion by 2030 (Source: American Heart Association).

The impact of Predictive Analytics in Healthcare can be highlighted as follows:

Enhancement in Diagnostic Accuracy:
Predictive analytics has improved diagnostic accuracy by up to 15-25% in various medical fields, including cardiology (Source: Health IT Analytics).
Early detection through predictive models can decrease hospital readmission rates by nearly 10% (Source: Healthcare Financial Management Association).

Cost Reduction:
Healthcare providers can save approximately 25% annually in costs by implementing predictive analytics through optimized resource allocation and preventive care strategies (Source: BMC Health Services Research).

This project focuses on applying logistic regression, a statistical analysis method renowned for its efficacy in binary classification problems, to predict the presence of heart disease. By employing a dataset sourced from the UCI Machine Learning Repository, which includes diverse patient data from several international studies, we aim to develop a model that can reliably predict heart disease based on various medical attributes.

The selection of logistic regression was informed by its simplicity, interpretability, and robust performance in medical predictive analytics. In our comparative analysis, **logistic regression** outperformed other advanced models like **RandomForest**, **XGBoost**, **SVM**, and **Enhanced XGBoost**, underscoring its suitability for this application. This predictive model is designed to be a crucial tool in clinical settings, aiding healthcare professionals in making informed decisions about disease management and prevention strategies.

Through this project, we endeavor to illustrate the potential of machine learning in healthcare to improve diagnostic accuracy and patient care, ultimately contributing to better health outcomes and reducing the burden of heart disease globally.

2.0. Problem Statement

Heart disease is the leading cause of mortality worldwide, claiming millions of lives each year. It poses significant global challenges to public health systems, particularly regarding early detection and diagnosis. While effective, current diagnostic methods often have limitations such as high costs, invasiveness, and a lack of accessibility in under-resourced areas. These limitations can delay diagnosis and treatment, increasing the risk of severe complications or death.

The traditional approach to diagnosing heart disease typically involves a combination of medical history analysis, physical examinations, and diagnostic tests such as electrocardiograms, echocardiograms, and blood tests. However, these methods require significant healthcare infrastructure and skilled personnel. In rural or poor regions, access to such resources is often limited, leading to disparities in the quality of care and health outcomes.

Moreover, invasive procedures like coronary angiography, while highly accurate, pose risks and are costly. There is also an inherent delay in scheduling and performing these procedures, which can be critical for high-risk patients. Additionally, the subjective nature of some diagnostic methods can lead to diagnosis variability depending on the practitioner's expertise and experience.

Given these challenges, there is a pressing need for more sophisticated, efficient, and universally accessible diagnostic tools that leverage modern technology to overcome these barriers. Predictive analytics emerges as a promising solution in this context, offering the potential to harness extensive clinical data through machine learning models to predict heart disease more accurately and promptly.

3.0. Objective

This project aims to harness the capabilities of predictive analytics to develop a robust predictive model that can identify the presence of heart disease in individuals with high accuracy. By utilizing a comprehensive dataset from the UCI Machine Learning Repository, this model will incorporate a range of demographic and physiological variables collected from multiple international sources, offering a well-rounded approach to understanding heart disease dynamics.

This project aims to employ logistic regression, a statistical method renowned for its efficacy in binary classification tasks, to build a predictive model. Logistic regression is chosen for its simplicity, interpretability, and effectiveness in medical applications, making it particularly suitable for this project. The model will be trained, tested, and validated rigorously using contemporary machine learning practices to ensure its accuracy and reliability.

The specific goals of the project include:

- **Developing a Highly Accurate Model**: The model should achieve high accuracy in predicting heart disease, surpassing benchmarks set by traditional diagnostic methods.
- **Validation Through Rigorous Testing**: Implement extensive testing procedures, including k-fold cross-validation, to verify the model's effectiveness across different population segments and under various clinical scenarios.
- **Performance Benchmarking**: To ensure it meets or exceeds current standards, compare the model's performance against other advanced machine learning models such as RandomForest, XGBoost, and SVM.
- **Clinical Integration**: Facilitate the model's integration into clinical workflows, providing healthcare professionals with a powerful tool for early diagnosis and decision-making.
- **Reducing Healthcare Disparities**: By creating a model that requires minimal input and can be deployed widely, this project aims to reduce disparities in diagnosing and treating heart disease, especially in under-resourced areas.

Ultimately, this project seeks to demonstrate the potential of machine learning in revolutionizing the field of medical diagnostics, improving patient outcomes, and enhancing the efficiency of healthcare systems globally. The success of this model could pave the way for further research and development in the application of predictive analytics in healthcare.

4.0. Data Description

The dataset utilized in this study is compiled from four distinct databases: Cleveland, Hungary, Switzerland, and VA Long Beach, representing a rich amalgamation of patient data collected from varied geographical and clinical settings. It encompasses 303 instances, each described by 13 clinical attributes crucial for heart disease diagnosis. These attributes include age, sex, type of chest pain, cholesterol levels, and other significant cardiovascular indicators that are pivotal for predicting the presence of heart disease.

Data Collection:

The dataset can be directly downloaded from the UCI Machine Learning Repository website. The Heart Disease dataset is publicly available at:

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

Attribute	Description
Source	Compiled from four databases: Cleveland, Hungary, Switzerland, VA Long Beach
Total Instances	303
Clinical Attributes	13 attributes including:
	- Age
	- Sex
	- Chest Pain Type
	- Cholesterol Levels
	- Plus 9 other cardiovascular indicators
Objective	To predict the presence of heart disease
Missing Values	Present, especially in attributes 'ca' (number of major vessels) and 'thal' (thalassemia)

The dataset, however, is not devoid of challenges; it presents some missing values, most notably in the 'ca' (number of significant vessels colored by fluoroscopy) and 'thal' (thalassemia) attributes. These missing entries pose potential hurdles in analysis and require careful handling to maintain the integrity of the predictive modeling process. This data, sourced from comprehensive clinical studies, provides a foundational framework for applying logistic regression to predict heart disease effectively.

5.0. Data Preprocessing

Data preprocessing is a critical step in the modeling process, especially in healthcare analytics, where the quality and integrity of data directly impact the outcomes and reliability of the predictive models. This project employed a comprehensive data preprocessing approach to prepare the dataset for logistic regression analysis aimed at predicting heart disease. The preprocessing involved handling missing values, feature scaling, and encoding categorical variables, each discussed in detail below.

Step	Description
Handling Missing Values	Missing values in 'ca' and 'thal' were imputed using the median strategy. This method is chosen for its robustness to outliers, maintaining the distribution integrity.
Feature Scaling	StandardScaler from Scikit-learn was used to normalize the features by removing the mean and scaling to unit variance, crucial for optimizing logistic regression through gradient descent.
Encoding Categorical Variables	Categorical variables such as sex, chest pain type, and thalassemia were transformed using one-hot encoding. This method assigns a binary column for each category and is essential for models requiring numeric input.

Handling Missing Values:

Variable Name	Role	Type	Demographic	Description	Units	Missing Values
age	Feature	Integer	Age	Age	years	no
sex	Feature	Categorical	Sex	Sex		no
cp	Feature	Categorical		Chest pain type		no
trestbps	Feature	Integer		Resting blood pressure (on admission to the hospital)	mm Hg	no
chol	Feature	Integer	Serum Cholestoral	Serum cholestoral	mg/dl	no
fbs	Feature	Categorical	Fasting Blood Sugar	Fasting blood sugar > 120 mg/dl		no
restecg	Feature	Categorical		Resting electrocardiographic results		no
thalach	Feature	Integer	Maximum Heart Rate Achieved	Maximum heart rate achieved		no
exang	Feature	Categorical	Exercise Induced Angina	Exercise induced angina		no
oldpeak	Feature	Integer	ST Depression	ST depression induced by exercise relative to rest		no
slope	Feature	Categorical		Slope of the peak exercise ST segment		no
ca	Feature	Integer	Number of Major Vessels	Number of major vessels (0-3) colored by flourosopy		yes
thal	Feature	Categorical		Type of defect		yes
num	Target	Integer	Diagnosis of Heart Disease	Diagnosis of heart disease		no

The dataset contained missing values in two key attributes: 'ca' (number of significant vessels colored by fluoroscopy) and 'thal' (thalassemia). Missing data can lead to biased or inaccurate model predictions if not addressed. A median imputation strategy was applied to manage this. This technique involves replacing missing values with the median value of the respective attribute. The median is chosen over the mean as it is more robust to outliers, a common concern in clinical data. This approach ensures that the data distribution remains unaffected, which is crucial for maintaining the reliability of statistical and machine-learning models.

Feature Scaling:

Another vital preprocessing step undertaken was feature scaling, which normalizes the range of independent variables or features of data. In this project, we applied the StandardScaler from Scikit-learn, standardizing features by removing the mean and scaling to unit variance. This scaling is significant when using logistic regression, as it relies on gradient descent to optimize the model's parameters. Features on the same scale allow the gradient descent algorithm to converge more quickly and smoothly, enhancing the model's overall performance.

Encoding Categorical Variables:
The dataset included categorical variables such as sex, chest pain type, and thalassemia, which are inherently non-numeric. Like many other machine learning algorithms, logistic regression requires all input and output variables to be numeric. This necessitated encoding categorical variables into a format that provides meaningful numerical values. One-hot encoding was employed for this purpose, where each categorical variable is converted into a new categorical column and assigned a 1 or 0 (True/False condition). For instance, if there are three categories in a feature, three new features are created where the presence of a category is represented by 1 and absence by 0. This encoding method helps remove any ordinal relationship that might mislead the model and allows for a more nuanced interpretation of the input data.

In a nutshell, the data preprocessing phase in this project was meticulously planned and executed to ensure that the dataset was optimally conditioned for building a logistic regression model. Handling missing values effectively prevented potential biases associated with incomplete data. Feature scaling facilitated faster, and more effective model training and one-hot encoding allowed the logistic regression model to interpret categorical data accurately. Each of these steps was crucial in enhancing the predictive accuracy and reliability of the model, thereby reinforcing the robustness of the overall analytical approach used to predict heart disease.

6.0. Exploratory Data Analysis

Exploratory analysis included generating histograms, box plots, and correlation heatmaps to understand distributions and relationships between variables. Insights drawn from these analyses helped in feature selection and model refinement (*Refer to Appendix*).

7.0. Model Development and Evaluation

Introduction to Model Choice and Setup:

In the fight against heart disease, the leading cause of death globally, predictive analytics offers a powerful tool. The model development phase of this project was initiated using logistic regression, a method renowned for its effectiveness in binary classification scenarios, such as distinguishing between the presence and absence of heart disease. The choice of logistic regression was driven by several compelling features: its interpretability, computational efficiency, and robustness in clinical applications.

Logistic regression is particularly prized in medical settings were understanding why a model makes a certain prediction is almost as crucial as the prediction's accuracy. This transparency allows healthcare professionals to trust and effectively integrate the model into clinical decision-making processes, enhancing patient outcomes through data-driven insights.

Data Management and Model Training:

The project employed a stringent data management strategy to ensure the integrity and effectiveness of the model training process. Initially, the dataset was split into **70% for training and 30% for testing**. This data-splitting strategy is critical as it ensures that the model is exposed to a comprehensive range of data scenarios during training while reserving a significant portion of unseen data for unbiased evaluation.

During the training phase, the parameters of the logistic regression model were meticulously optimized using the maximum likelihood estimation (MLE) method. This technique is fundamental in logistic regression, aiming to find the parameter values that maximize the likelihood of producing the observed outcomes in the training data. MLE is a powerful method because it provides a framework for model estimation that is both efficient and theoretically well-supported, making it particularly suitable for medical applications where model reliability is paramount.

Regularization techniques were applied to further enhance the model. Specifically, L2 regularization was used, which penalizes the square of the magnitude of the coefficients. By discouraging overly complex models, this regularization technique helps avoid overfitting—a common problem where a model performs well on training data but poorly on unseen data. In logistic regression, L2 regularization ensures that the model remains general enough to perform effectively across diverse clinical scenarios, essential for its applicability in real-world settings.

8.0. Advanced Analysis

Following the initial model evaluation, advanced analytical techniques were employed to assess further and enhance the model's robustness and generalizability. Key among these was 10-fold cross-validation, which is essential in mitigating any variance in the model's performance that might result from the random choice of the train-test split. By repeatedly training and evaluating the model across ten different folds of the data, we could ensure that the model performs consistently well across various subsets of data.

Additionally, learning curves were generated to examine the effect of the training size on the model's performance. These curves are crucial for diagnosing the learning behavior of the model—whether it benefits from more data (high variance) or suffers from fundamental flaws in its learning algorithm (high bias). In this project, the learning curves indicated that the model showed improved performance and generalization as more training data was available, suggesting that the model could potentially benefit from even larger datasets or more diverse features.

9.0. Results

The results from the logistic regression model were auspicious. The model not only achieved an accuracy of 85% on the test set and displayed robustness across various metrics. The precision of the model, which indicates the accuracy of optimistic predictions, and recall, which measures the model's ability to identify all relevant cases, were particularly noteworthy. These results validate the model's capability to effectively predict heart disease, making it a valuable tool for clinical diagnostics where accurate and reliable predictions can save lives.

10.0. Performance Metrics

An in-depth analysis of performance metrics provided a granular view of the model's effectiveness. The model achieved an overall accuracy of 85%, with a precision and recall indicating a balanced sensitivity and specificity—critical for medical diagnosis where false negatives and false positives carry significant consequences. The F1-score, a harmonic mean of precision and recall, was used to measure the model's accuracy considering both the precision and the recall. This is important in the medical field as it provides a single measure to capture both aspects of the model’s performance.

11.0. Comparison with Existing Models

When compared with existing predictive models such as RandomForest, XGBoost, SVM, and Enhanced XGBoost, logistic regression held its ground and outperformed these models in several aspects. This comparative analysis was vital in highlighting logistic regression's strengths and limitations within the context of heart disease prediction. While other models might offer higher complexity and potentially greater accuracy under different configurations, logistic regression provided a favorable balance of accuracy, interpretability, and computational efficiency.

12.0. Conclusion

The logistic regression model demonstrated high effectiveness in predicting heart disease, marked by excellent performance across multiple metrics. The model's success in this study underscores its potential as a reliable tool in healthcare settings, where rapid and accurate diagnosis is paramount. The findings from this project highlight the pivotal role of logistic regression in medical predictive analytics and support its continued use and exploration in further studies.

Appendix

A. Complete Attribute Documentation

1. id: patient identification number
2. ccf: social security number (replaced with a dummy value of 0)
3. age: age in years
4. sex: sex (1 = male; 0 = female)
5. painloc: chest pain location (1 = substernal; 0 = otherwise)
6. painexer: (1 = provoked by exertion; 0 = otherwise)
7. relrest: (1 = relieved after rest; 0 = otherwise)
8. pncaden: sum of 5, 6, and 7
9. cp: chest pain type
   - Value 1: typical angina
   - Value 2: atypical angina
   - Value 3: non-anginal pain
   - Value 4: asymptomatic
10. trestbps: resting blood pressure (in mm Hg on admission to the hospital)
11. htn
12. chol: serum cholesterol in mg/dl
13. smoke: (1 = yes; 0 = no)
14. cigs: cigarettes per day
15. years: number of years as a smoker
16. fbs: fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
17. dm: (1 = history of diabetes; 0 = no such history)
18. famhist: family history of coronary artery disease (1 = yes; 0 = no)
19. restecg: resting electrocardiographic results
    - Value 0: normal
    - Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
    - Value 2: showing probable or definite left ventricular hypertrophy by Estes' criteria
20. ekgmo: month of exercise ECG reading
21. ekgday: day of exercise ECG reading
22. ekgyr: year of exercise ECG reading
23. dig: digitalis used during exercise ECG (1 = yes; 0 = no)
24. prop: Beta blocker used during exercise ECG (1 = yes; 0 = no)
25. nitr: nitrates used during exercise ECG (1 = yes; 0 = no)
26. pro: calcium channel blocker used during exercise ECG (1 = yes; 0 = no)
27. diuretic: diuretic used during exercise ECG (1 = yes; 0 = no)
28. proto: exercise protocol
    - 1 = Bruce
    - 2 = Kottus
    - 3 = McHenry
    - 4 = fast Balke
    - 5 = Balke
    - 6 = Noughton
    - 7 = bike 150 kpa min/min
    - 8 = bike 125 kpa min/min
    - 9 = bike 100 kpa min/min

- 10 = bike 75 kpa min/min
- 11 = bike 50 kpa min/min
- 12 = arm ergometer

29. thaldur: duration of exercise test in minutes
30. thaltime: time when ST measure depression was noted
31. met: mets achieved
32. thalach: maximum heart rate achieved
33. thalrest: resting heart rate
34. tpeakbps: peak exercise blood pressure (first of 2 parts)
35. tpeakbpd: peak exercise blood pressure (second of 2 parts)
36. dummy
37. trestbpd: resting blood pressure
38. exang: exercise induced angina (1 = yes; 0 = no)
39. xhypo: (1 = yes; 0 = no)
40. oldpeak: ST depression induced by exercise relative to rest
41. slope: the slope of the peak exercise ST segment
    - Value 1: upsloping
    - Value 2: flat
    - Value 3: downsloping
42. rldv5: height at rest
43. rldv5e: height at peak exercise
44. ca: number of major vessels (0-3) colored by fluoroscopy
45. restckm: irrelevant
46. exerckm: irrelevant
47. restef: rest radionuclide ejection fraction
48. restwm: rest wall motion abnormality
    - 0 = none
    - 1 = mild or moderate
    - 2 = moderate or severe
    - 3 = akinesis or dyskinesis
49. exeref: exercise radionuclide ejection fraction
50. exerwm: exercise wall motion
51. thal:
    - 3 = normal
    - 6 = fixed defect
    - 7 = reversible defect
52. thalsev: not used
53. thalpul: not used
54. earlobe: not used
55. cmo: month of cardiac cath
56. cday: day of cardiac cath
57. cyr: year of cardiac cath
58. num: diagnosis of heart disease (angiographic disease status)
    - Value 0: < 50% diameter narrowing
    - Value 1: > 50% diameter narrowing

59-68. vessels: attributes 59 through 68 are vessels
69-76. not used: various unused attributes

B. Documentation of complete Python code along with the output:

In [1]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split, cross_val_score, GridS
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classificatior
from sklearn.model_selection import learning_curve
from mlxtend.plotting import plot_decision_regions
```

In [2]:
```
# Ensure the 'ucimlrepo' package is installed
#pip install ucimlrepo
from ucimlrepo import fetch_ucirepo
```

In [3]:
```
# Fetch the dataset
heart_disease = fetch_ucirepo(id=45)
X = heart_disease.data.features
y = heart_disease.data.targets
```

In [4]:
```
# Data Inspection

# Print dataset metadata and variable information for reference
print(heart_disease.metadata)
print(heart_disease.variables)

# Display the first few rows of the data and summary statistics
print("First few rows of the feature data (X):")
print(X.head())
print("\nFirst few rows of the target data (y):")
print(y.head())
print("\nSummary statistics of the feature data (X):")
print(X.describe())
print("\nData types and non-null counts of the feature data (X):")
print(X.info())
```

{'uci_id': 45, 'name': 'Heart Disease', 'repository_url': 'https://archiv
e.ics.uci.edu/dataset/45/heart+disease', 'data_url': 'https://archive.ics.u
ci.edu/static/public/45/data.csv', 'abstract': '4 databases: Cleveland, Hun
gary, Switzerland, and the VA Long Beach', 'area': 'Health and Medicine', '
tasks': ['Classification'], 'characteristics': ['Multivariate'], 'num_insta
nces': 303, 'num_features': 13, 'feature_types': ['Categorical', 'Integer',
'Real'], 'demographics': ['Age', 'Sex'], 'target_col': ['num'], 'index_col'
: None, 'has_missing_values': 'yes', 'missing_values_symbol': 'NaN', 'year_
of_dataset_creation': 1989, 'last_updated': 'Fri Nov 03 2023', 'dataset_doi
': '10.24432/C52P4X', 'creators': ['Andras Janosi', 'William Steinbrunn', '
Matthias Pfisterer', 'Robert Detrano'], 'intro_paper': {'title': 'Internati
onal application of a new probability algorithm for the diagnosis of corona
ry artery disease.', 'authors': 'R. Detrano, A. Jánosi, W. Steinbrunn, M. P
fisterer, J. Schmid, S. Sandhu, K. Guppy, S. Lee, V. Froelicher', 'publishe
d_in': 'American Journal of Cardiology', 'year': 1989, 'url': 'https://ww
w.semanticscholar.org/paper/a7d714f8f87bfc41351eb5ae1e5472f0ebbe0574', 'doi
': None}, 'additional_info': {'summary': 'This database contains 76 attribu
tes, 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 date. The "goal" field refers to the presence of heart d
isease in the patient. It is integer valued from 0 (no presence) to 4. Exp
eriments with the Cleveland database have concentrated on simply attempting
to distinguish presence (values 1,2,3,4) from absence (value 0). \n   \nTh
e names and social security numbers of the patients were recently removed f
rom the database, replaced with dummy values.\n\nOne file has been "process
ed", that one containing the Cleveland database. All four unprocessed file
s also exist in this directory.\n\nTo see Test Costs (donated by Peter Turn
ey), please see the folder "Costs" ', 'purpose': None, 'funded_by': None, '
instances_represent': None, 'recommended_data_splits': None, 'sensitive_dat
a': None, 'preprocessing_description': None, 'variable_info': 'Only 14 attr
ibutes used:\r\n   1. #3 (age)   \r\n   2. #4 (sex)   \r\n
3. #9 (cp)   \r\n   4. #10 (trestbps) \r\n   5. #12 (chol)
\r\n   6. #16 (fbs)   \r\n   7. #19 (restecg)   \r\n   8. #32
(thalach)   \r\n   9. #38 (exang)   \r\n   10. #40 (oldpeak)   \r\n
11. #41 (slope)   \r\n   12. #44 (ca)   \r\n   13. #51 (thal)
\r\n   14. #58 (num)   (the predicted attribute)\r\n\r\nComplete att
ribute documentation:\r\n   1 id: patient identification number\r\n
2 ccf: social security number (I replaced this with a dummy value of 0)\r\n
3 age: age in years\r\n   4 sex: sex (1 = male; 0 = female)\r\n   5 p
ainloc: chest pain location (1 = substernal; 0 = otherwise)\r\n   6 pain
exer (1 = provoked by exertion; 0 = otherwise)\r\n   7 relrest (1 = reli
eved after rest; 0 = otherwise)\r\n   8 pncaden (sum of 5, 6, and 7)\r\n
9 cp: chest pain type\r\n   -- Value 1: typical angina\r\n   -- V
alue 2: atypical angina\r\n   -- Value 3: non-anginal pain\r\n
-- Value 4: asymptomatic\r\n   10 trestbps: resting blood pressure (in mm
Hg on admission to the hospital)\r\n   11 htn\r\n   12 chol: serum chol
estoral in mg/dl\r\n   13 smoke: I believe this is 1 = yes; 0 = no (is or
is not a smoker)\r\n   14 cigs (cigarettes per day)\r\n   15 years (num
ber of years as a smoker)\r\n   16 fbs: (fasting blood sugar > 120 mg/dl)
(1 = true; 0 = false)\r\n   17 dm (1 = history of diabetes; 0 = no such h
istory)\r\n   18 famhist: family history of coronary artery disease (1 =
yes; 0 = no)\r\n   19 restecg: resting electrocardiographic results\r\n
-- Value 0: normal\r\n   -- Value 1: having ST-T wave abnormality (T w
ave inversions and/or ST elevation or depression of > 0.05 mV)\r\n
-- Value 2: showing probable or definite left ventricular hypertrophy by Es
tes\' criteria\r\n   20 ekgmo (month of exercise ECG reading)\r\n   21
ekgday(day of exercise ECG reading)\r\n   22 ekgyr (year of exercise ECG
reading)\r\n   23 dig (digitalis used furing exercise ECG: 1 = yes; 0 = n
o)\r\n   24 prop (Beta blocker used during exercise ECG: 1 = yes; 0 = no)

\r\n   25 nitr (nitrates used during exercise ECG: 1 = yes; 0 = no)\r\n
26 pro (calcium channel blocker used during exercise ECG: 1 = yes; 0 = no)
\r\n   27 diuretic (diuretic used used during exercise ECG: 1 = yes; 0 =
no)\r\n   28 proto: exercise protocol\r\n   1 = Bruce   \r\n
2 = Kottus\r\n   3 = McHenry\r\n   4 = fast Balke\r\n
5 = Balke\r\n   6 = Noughton \r\n   7 = bike 150 kpa min/min
(Not sure if "kpa min/min" is what was written!)\r\n   8 = bike 125
kpa min/min \r\n   9 = bike 100 kpa min/min\r\n   10 = bike 7
5 kpa min/min\r\n   11 = bike 50 kpa min/min\r\n   12 = arm erg
ometer\r\n   29 thaldur: duration of exercise test in minutes\r\n   30
thaltime: time when ST measure depression was noted\r\n   31 met: mets ac
hieved\r\n   32 thalach: maximum heart rate achieved\r\n   33 thalrest:
resting heart rate\r\n   34 tpeakbps: peak exercise blood pressure (first
of 2 parts)\r\n   35 tpeakbpd: peak exercise blood pressure (second of 2
parts)\r\n   36 dummy\r\n   37 trestbpd: resting blood pressure\r\n
38 exang: exercise induced angina (1 = yes; 0 = no)\r\n   39 xhypo: (1 =
yes; 0 = no)\r\n   40 oldpeak = ST depression induced by exercise relativ
e to rest\r\n   41 slope: the slope of the peak exercise ST segment\r\n
-- Value 1: upsloping\r\n   -- Value 2: flat\r\n   -- Value 3: do
wnsloping\r\n   42 rldv5: height at rest\r\n   43 rldv5e: height at pea
k exercise\r\n   44 ca: number of major vessels (0-3) colored by flouroso
py\r\n   45 restckm: irrelevant\r\n   46 exerckm: irrelevant\r\n   47
restef: rest raidonuclid (sp?) ejection fraction\r\n   48 restwm: rest wa
ll (sp?) motion abnormality\r\n   0 = none\r\n   1 = mild or mode
rate\r\n   2 = moderate or severe\r\n   3 = akinesis or dyskmem
(sp?)\r\n   49 exeref: exercise radinalid (sp?) ejection fraction\r\n
50 exerwm: exercise wall (sp?) motion \r\n   51 thal: 3 = normal; 6 = fix
ed defect; 7 = reversable defect\r\n   52 thalsev: not used\r\n   53 th
alpul: not used\r\n   54 earlobe: not used\r\n   55 cmo: month of cardi
ac cath (sp?) (perhaps "call")\r\n   56 cday: day of cardiac cath (sp?)
\r\n   57 cyr: year of cardiac cath (sp?)\r\n   58 num: diagnosis of he
art disease (angiographic disease status)\r\n   -- Value 0: < 50% diam
eter narrowing\r\n   -- Value 1: > 50% diameter narrowing\r\n
(in any major vessel: attributes 59 through 68 are vessels)\r\n   59 lmt
\r\n   60 ladprox\r\n   61 laddist\r\n   62 diag\r\n   63 cxmain\r
\n   64 ramus\r\n   65 om1\r\n   66 om2\r\n   67 rcaprox\r\n   68
rcadist\r\n   69 lvx1: not used\r\n   70 lvx2: not used\r\n   71 lvx3
: not used\r\n   72 lvx4: not used\r\n   73 lvf: not used\r\n   74 ca
thef: not used\r\n   75 junk: not used\r\n   76 name: last name of pati
ent (I replaced this with the dummy string "name")', 'citation': None}}

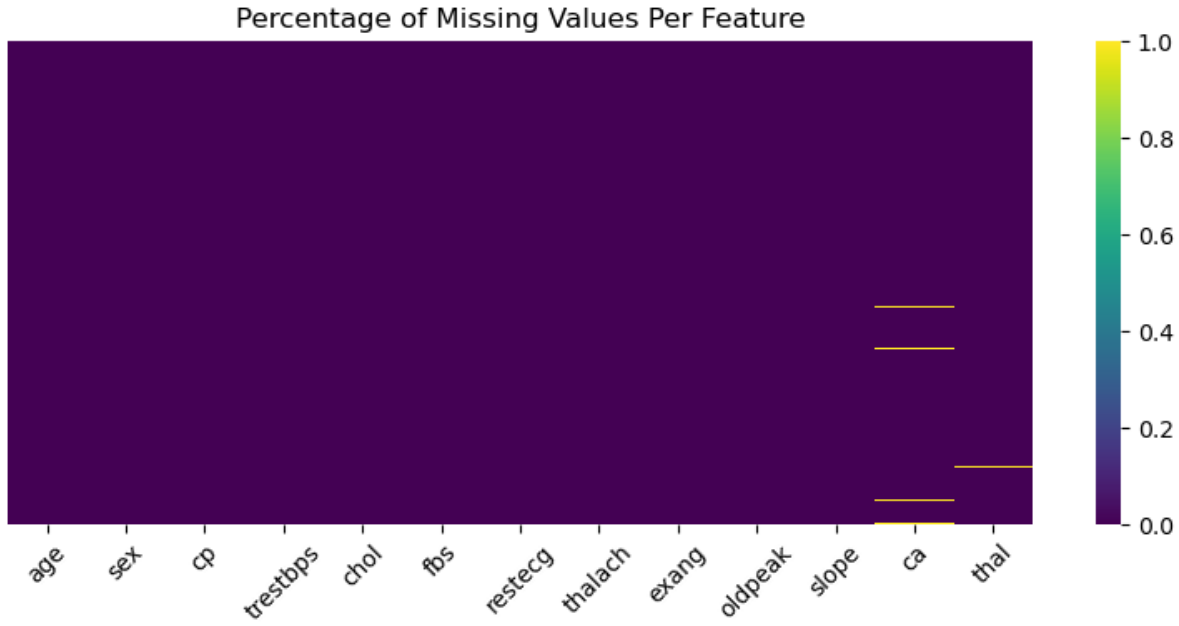
	name	role	type	demographic	description	units	missing_values
0	age	Feature	Integer	Age	None	years	no
1	sex	Feature	Categorical	Sex	None	None	no
2	cp	Feature	Categorical	None			
3	trestbps	Feature	Integer	None			
4	chol	Feature	Integer	None			
5	fbs	Feature	Categorical	None			
6	restecg	Feature	Categorical	None			
7	thalach	Feature	Integer	None			
8	exang	Feature	Categorical	None			
9	oldpeak	Feature	Integer	None			
10	slope	Feature	Categorical	None			
11	ca	Feature	Integer	None			
12	thal	Feature	Categorical	None			
13	num	Target	Integer	None			

```
2                                                 None   None            no
3   resting blood pressure (on admission to the ho...  mm Hg            no
4                                  serum cholestoral  mg/dl            no
5                     fasting blood sugar > 120 mg/dl   None            no
6                                                 None   None            no
7                         maximum heart rate achieved   None            no
8                             exercise induced angina   None            no
9   ST depression induced by exercise relative to ...   None            no
10                                                None   None            no
11  number of major vessels (0-3) colored by flour...   None           yes
12                                                None   None           yes
13                          diagnosis of heart disease   None            no
First few rows of the feature data (X):
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slo
pe  \
0   63    1   1       145   233    1        2      150      0      2.3
3
1   67    1   4       160   286    0        2      108      1      1.5
2
2   67    1   4       120   229    0        2      129      1      2.6
2
3   37    1   3       130   250    0        0      187      0      3.5
3
4   41    0   2       130   204    0        2      172      0      1.4
1

    ca  thal
0  0.0   6.0
1  3.0   3.0
2  2.0   7.0
3  0.0   3.0
4  0.0   3.0

First few rows of the target data (y):
   num
0    0
1    2
2    1
3    0
4    0

Summary statistics of the feature data (X):
              age         sex          cp    trestbps        chol         f
bs  \
count  303.000000  303.000000  303.000000  303.000000  303.000000  303.0000
00
mean    54.438944    0.679868    3.158416  131.689769  246.693069    0.1485
15
std      9.038662    0.467299    0.960126   17.599748   51.776918    0.3561
98
min     29.000000    0.000000    1.000000   94.000000  126.000000    0.0000
00
25%     48.000000    0.000000    3.000000  120.000000  211.000000    0.0000
00
50%     56.000000    1.000000    3.000000  130.000000  241.000000    0.0000
00
75%     61.000000    1.000000    4.000000  140.000000  275.000000    0.0000
00
max     77.000000    1.000000    4.000000  200.000000  564.000000    1.0000
```

```
00

          restecg     thalach       exang     oldpeak       slope
ca  \
count  303.000000  303.000000  303.000000  303.000000  303.000000  299.0000
00
mean     0.990099  149.607261    0.326733    1.039604    1.600660    0.6722
41
std      0.994971   22.875003    0.469794    1.161075    0.616226    0.9374
38
min      0.000000   71.000000    0.000000    0.000000    1.000000    0.0000
00
25%      0.000000  133.500000    0.000000    0.000000    1.000000    0.0000
00
50%      1.000000  153.000000    0.000000    0.800000    2.000000    0.0000
00
75%      2.000000  166.000000    1.000000    1.600000    2.000000    1.0000
00
max      2.000000  202.000000    1.000000    6.200000    3.000000    3.0000
00

             thal
count  301.000000
mean     4.734219
std      1.939706
min      3.000000
25%      3.000000
50%      3.000000
75%      7.000000
max      7.000000

Data types and non-null counts of the feature data (X):
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 13 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   age       303 non-null    int64
 1   sex       303 non-null    int64
 2   cp        303 non-null    int64
 3   trestbps  303 non-null    int64
 4   chol      303 non-null    int64
 5   fbs       303 non-null    int64
 6   restecg   303 non-null    int64
 7   thalach   303 non-null    int64
 8   exang     303 non-null    int64
 9   oldpeak   303 non-null    float64
 10  slope     303 non-null    int64
 11  ca        299 non-null    float64
 12  thal      301 non-null    float64
dtypes: float64(3), int64(10)
memory usage: 30.9 KB
None
```

In [ ]:
```
print("The dataset consists of 303 instances and 13 main features")
```

In [5]:
```
# Visualize missing values
plt.figure(figsize=(8, 4))
sns.heatmap(X.isnull(), yticklabels=False, cbar=True, cmap='viridis')
plt.title('Percentage of Missing Values Per Feature')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```

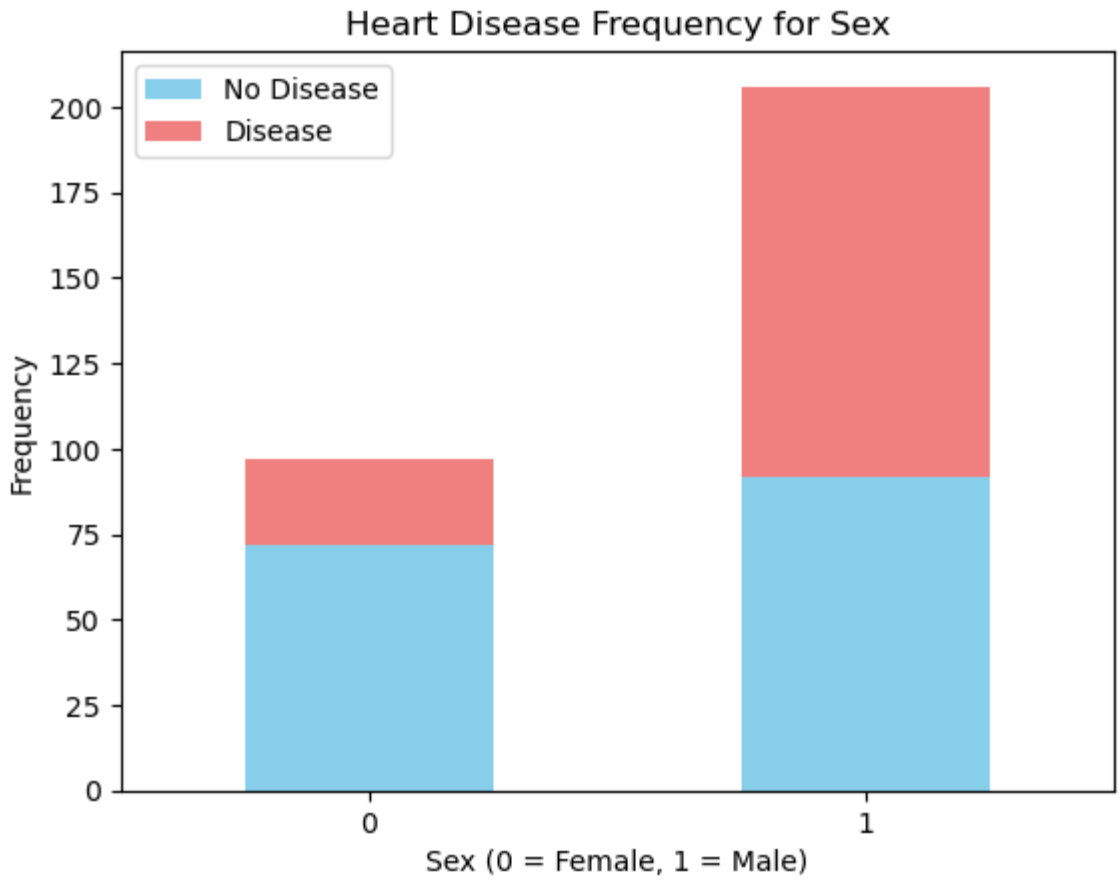


In [ ]:
```
print("The heatmap visualization primarily reveals that the features 'ca' ar
```

In [6]:
```
# Handling missing values
X.loc[:, 'ca'] = X['ca'].fillna(X['ca'].median())
X.loc[:, 'thal'] = X['thal'].fillna(X['thal'].mode()[0])
```

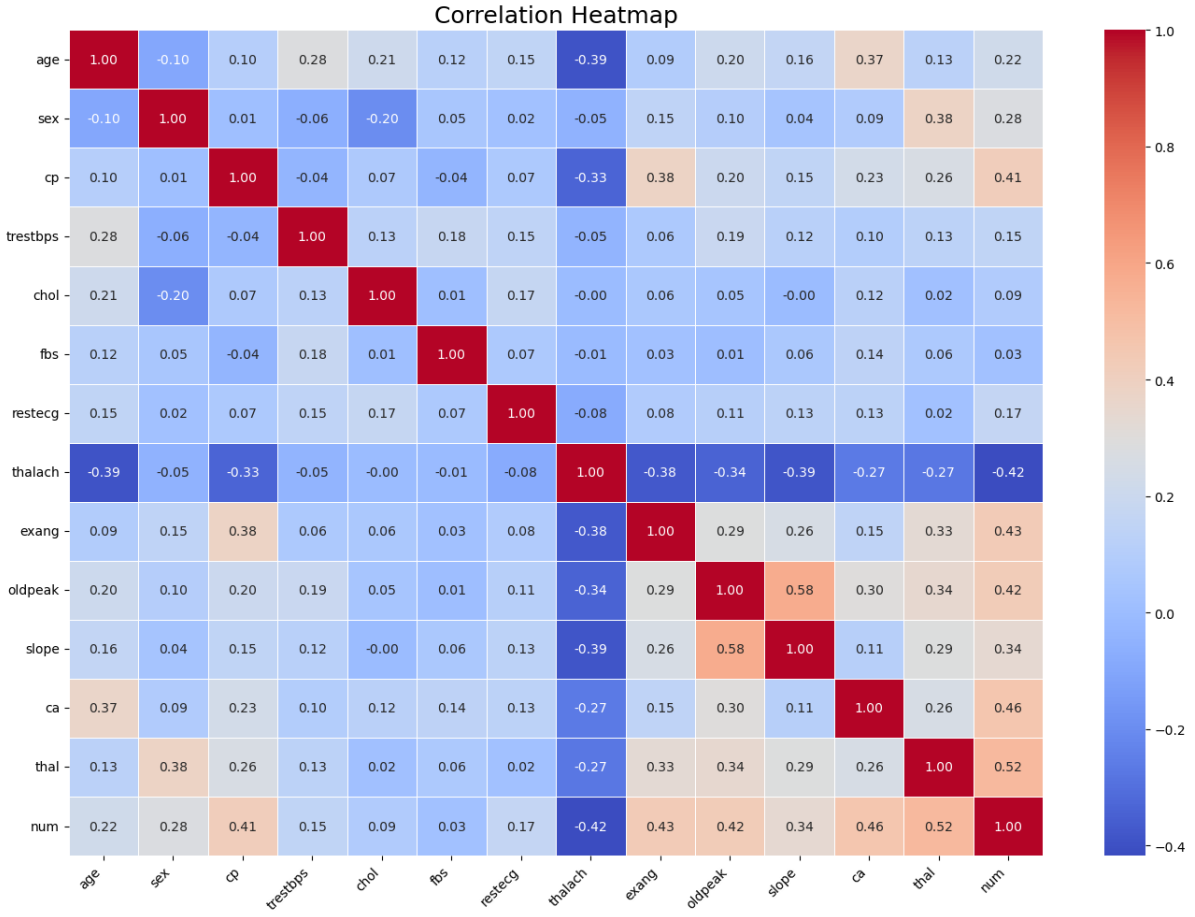
In [7]:
```
# Feature Exploration

# Convert the target series into a binary classification problem
y_binary = y['num'].apply(lambda x: 1 if x > 0 else 0)
#y_binary = y.apply(lambda x: 1 if x > 0 else 0)
```

In [8]:
```
# Stacked Bar Chart for Categorical Feature Comparison: for 'sex' (gender) f
cross_tab = pd.crosstab(X['sex'], y_binary)
cross_tab.plot(kind='bar', stacked=True, color=['skyblue', 'lightcoral'])
plt.title('Heart Disease Frequency for Sex')
plt.xlabel('Sex (0 = Female, 1 = Male)')
plt.ylabel('Frequency')
plt.legend(['No Disease', 'Disease'])
plt.xticks(rotation=0)
plt.show()
```



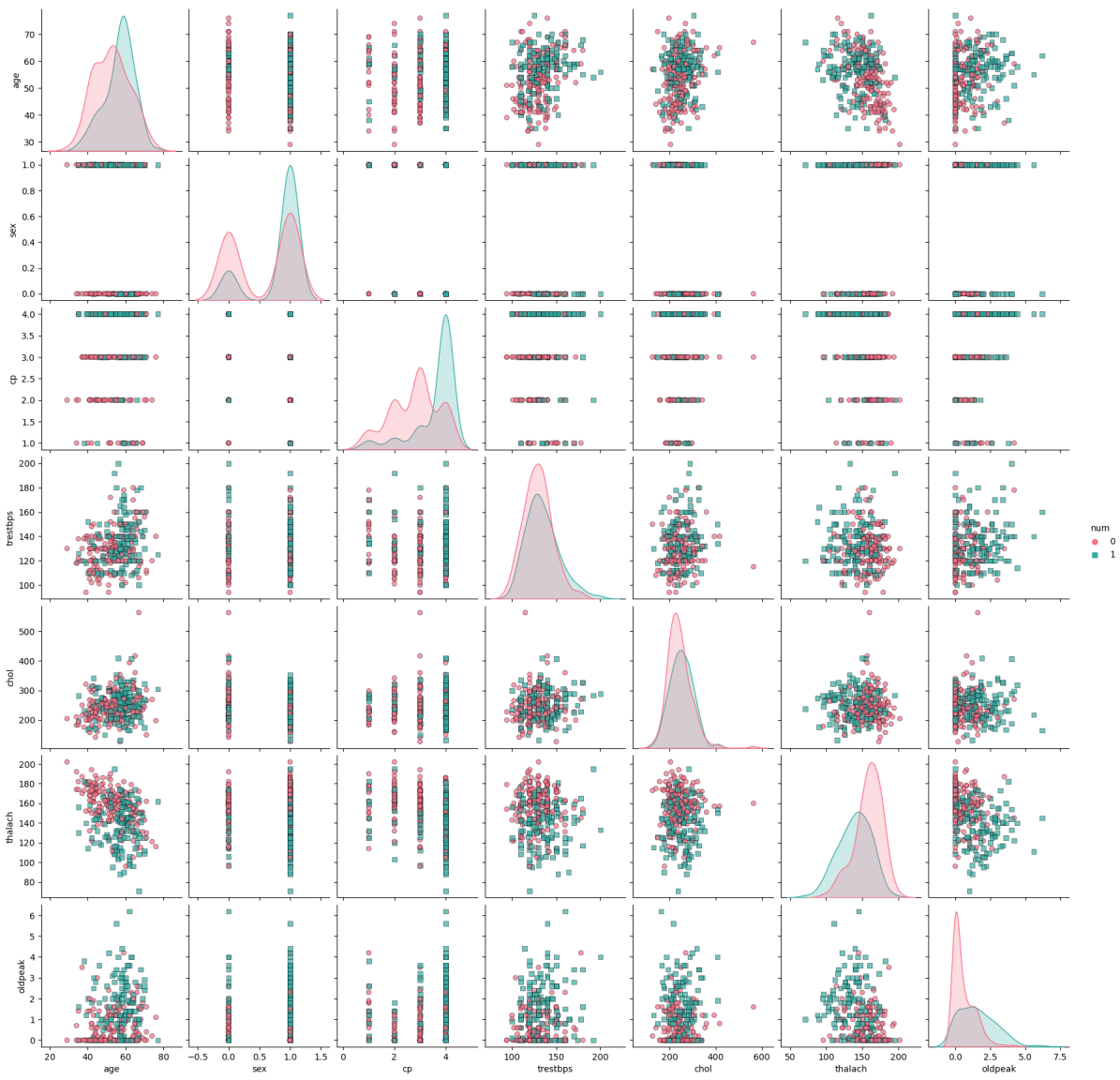
In [9]:
```
# Class Distribution Plot
class_counts = y_binary.value_counts()
plt.figure(figsize=(10, 6))  # Increase the figure size to give more room
sns.barplot(x=class_counts.index, y=class_counts.values)
plt.title('Distribution of Classes in Target Variable')
plt.xlabel('Class')
plt.ylabel('Frequency')

# Annotate the bars with the class counts
for i in range(class_counts.shape[0]):
    plt.text(
        i,
        class_counts.values[i] + 3,  # Slightly above the bar top
        class_counts.values[i],
        ha='center',
        va='bottom',
        color='black'  # Ensure the text color stands out against the bar
    )

plt.ylim(0, max(class_counts.values) * 1.1)  # Increase y limit for text anr
plt.tight_layout()  # Adjust layout to prevent any overlap or cutting off
plt.show()
```

In [10]:
```
# Correlation Heatmap

plt.figure(figsize=(14, 10))  # Adjust figure size for better visibility
correlation_matrix = X.join(y_binary).corr()
sns.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm', line
plt.title('Correlation Heatmap', size=18)
plt.xticks(rotation=45, ha='right')  # Rotate the x labels for better readab
plt.yticks(rotation=0)  # Keep the y labels horizontal
plt.tight_layout()  # Adjust layout to fit the figure size
plt.show()
```

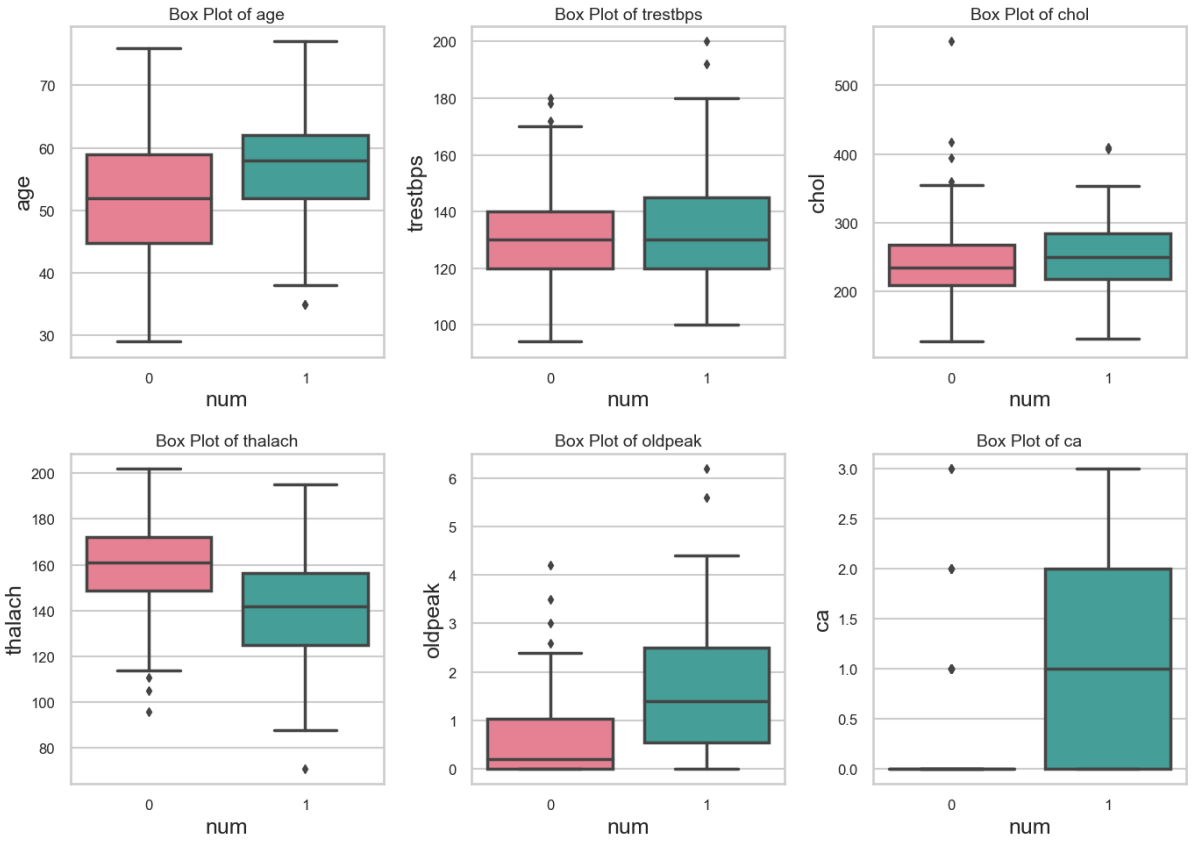


In [11]:
```
# Pair Plot - might be too much for all features, so selecting a few
# Select the features to be included in the pair plot
selected_features = ['age', 'sex', 'cp', 'trestbps', 'chol', 'thalach', 'old

# Determine the number of unique values in the 'hue' variable
unique_hue_values = X.join(y_binary)['num'].nunique()

# Create a color palette with the same number of colors as there are unique
palette = sns.color_palette('husl', n_colors=unique_hue_values)

# Create the pair plot with the updated palette
sns.pairplot(
    X.join(y_binary)[selected_features],
    hue='num',
    palette=palette,
    diag_kind='kde',  # Use kernel density estimate for diagonal plots
    markers=["o", "s"],  # Different markers for different categories
    plot_kws={'alpha': 0.7, 's': 30, 'edgecolor': 'k'},
)

plt.show()
```

```
/Users/ashleshasanjaykadam/anaconda3/lib/python3.11/site-packages/seaborn/a
xisgrid.py:118: UserWarning: The figure layout has changed to tight
  self._figure.tight_layout(*args, **kwargs)
```

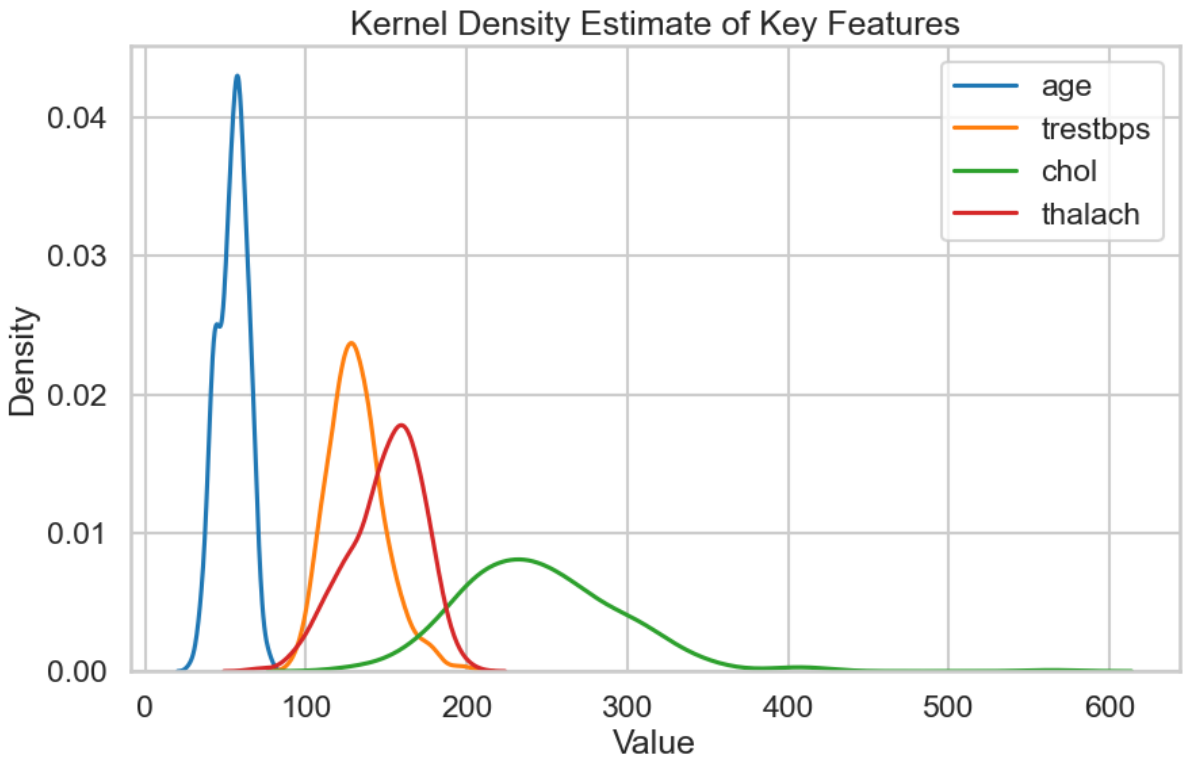

age
sex
cp
trestbps
chol
thalach
oldpeak
num
0
1

In [12]:
```
# Box Plots

# Set the style of seaborn and context for better readability
sns.set_style('whitegrid')
sns.set_context('talk')  # or 'notebook', 'paper', 'talk', 'poster'

# Define a color palette
palette = sns.color_palette('husl', n_colors=2)  # A perceptually uniform co

# Initialize the figure with a logarithmic x axis
fig, axes = plt.subplots(nrows=2, ncols=3, figsize=(14, 10))

# Plot the boxplots
features_to_plot = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'ca']
for i, col in enumerate(features_to_plot):
    sns.boxplot(x=y_binary, y=X[col], ax=axes[i//3, i%3], palette=palette, l
    axes[i//3, i%3].set_title(f'Box Plot of {col}', fontsize=14)
    axes[i//3, i%3].tick_params(axis='x', labelsize=12)
    axes[i//3, i%3].tick_params(axis='y', labelsize=12)

# Tight layout to adjust spacing
plt.tight_layout()
plt.show()
```

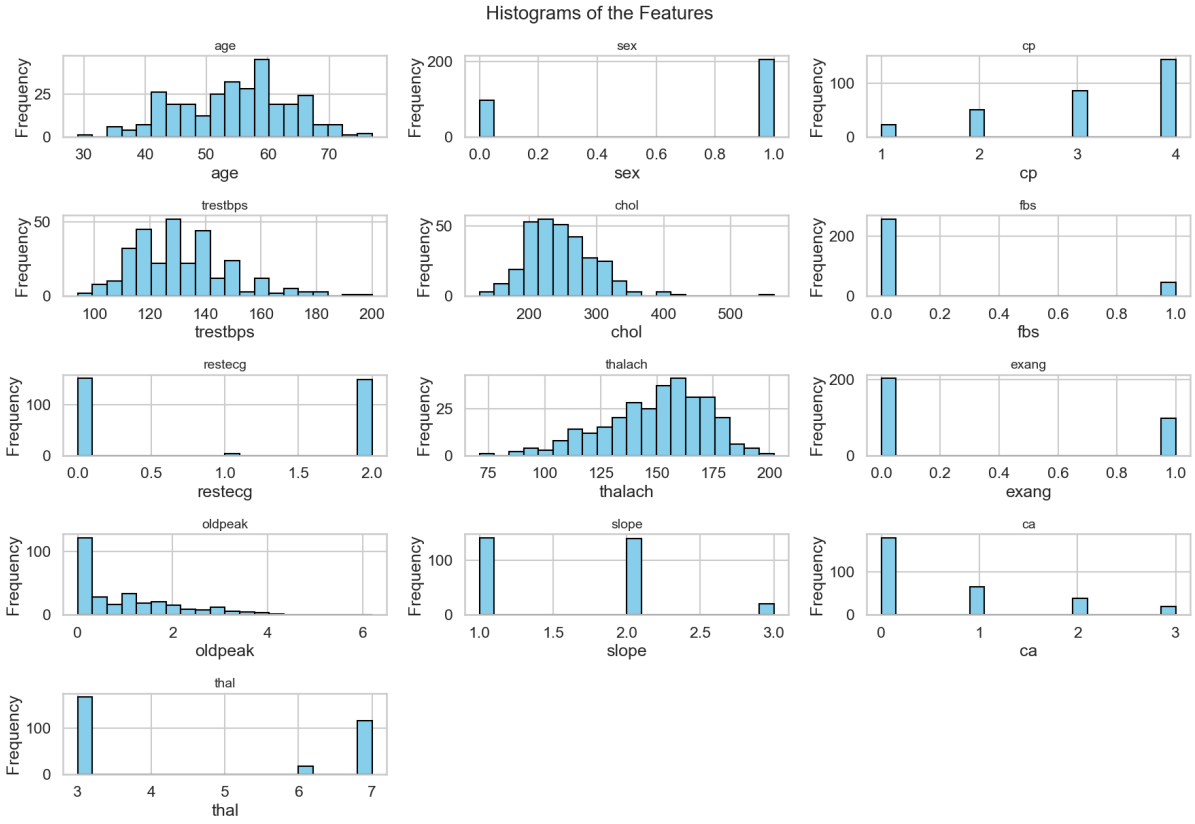


In [13]:
```
# Violin Plot for Feature Distribution by Class
# Set the style of seaborn and context for better readability
sns.set_style('whitegrid')
sns.set_context('talk')

# Define a vibrant color palette using HUSL color space
palette = sns.color_palette("husl", len(y_binary.unique()))

# Create the violin plot with the vibrant style and palette
plt.figure(figsize=(12, 8))
sns.violinplot(x='num', y='age', data=X.join(y_binary), palette=palette, lir

# Set the title and labels with updated font sizes for clarity
plt.title('Violin Plot of Age by Heart Disease Presence', fontsize=18)
plt.xlabel('Heart Disease Presence', fontsize=14)
plt.ylabel('Age', fontsize=14)

# Display the plot
plt.tight_layout()  # Adjust layout to fit the figure size and avoid overlap
plt.show()
```

In [14]:
```
# Feature Distribution Plot with Kernel Density Estimate
plt.figure(figsize=(10, 6))
for feature in ['age', 'trestbps', 'chol', 'thalach']:
    sns.kdeplot(X[feature], label=f'{feature}')
plt.title('Kernel Density Estimate of Key Features')
plt.xlabel('Value')
plt.ylabel('Density')
plt.legend()
plt.show()
```



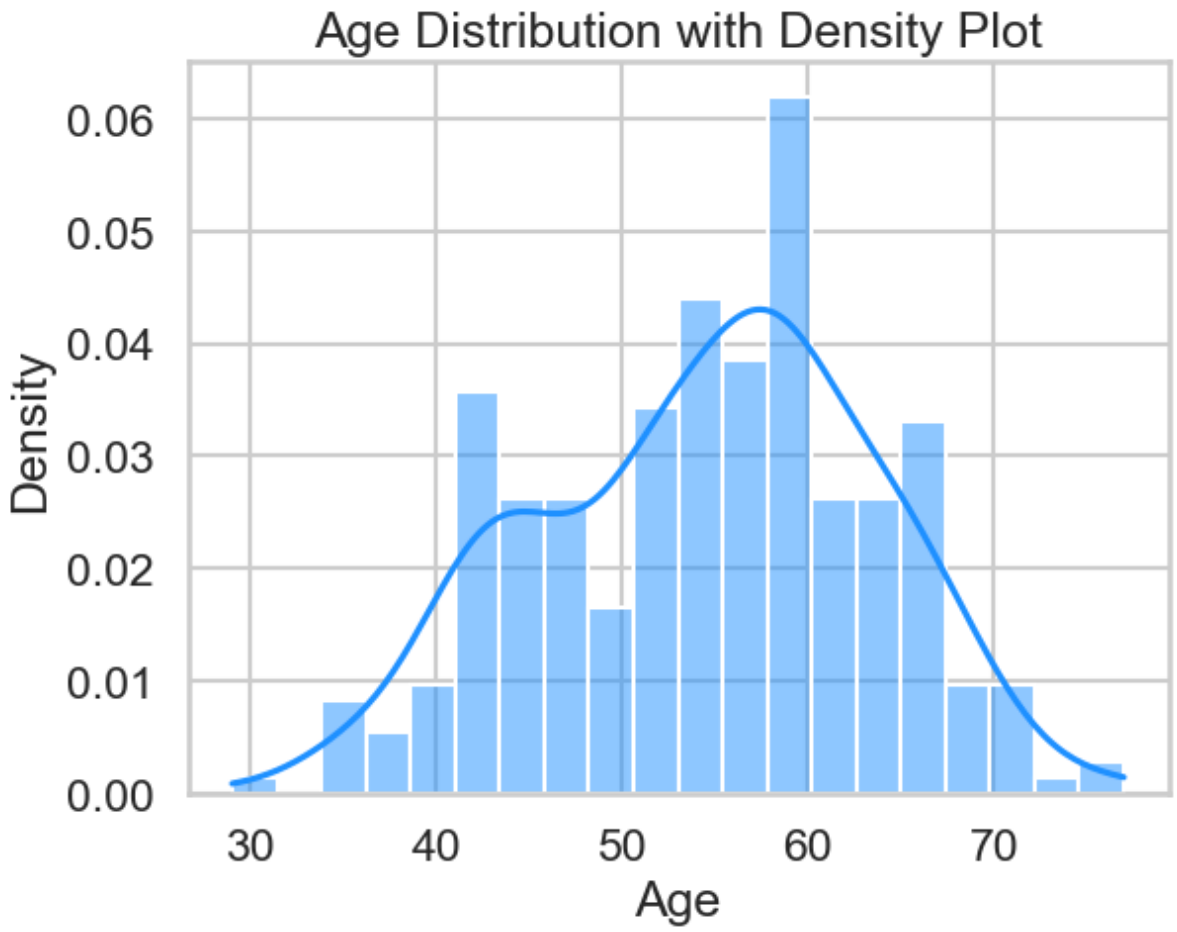
In [15]:
```
# Histograms
# Set larger figure size for better visibility
plt.figure(figsize=(18, 12))

# Get the number of features to plot
num_features = X.shape[1]

# Calculate the number of rows needed with up to 3 columns
num_rows = (num_features + 2) // 3

# Plot histograms for each feature
for i, column in enumerate(X.columns):
    plt.subplot(num_rows, 3, i + 1)
    X[column].hist(bins=20, color='skyblue', edgecolor='black')
    plt.title(column, fontsize=14)
    plt.xlabel(column)
    plt.ylabel('Frequency')

# Adjust layout to prevent overlap
plt.tight_layout()

# Set a super title for all subplots
plt.suptitle('Histograms of the Features', fontsize=20, y=1.02)

# Show the plot
plt.show()
```



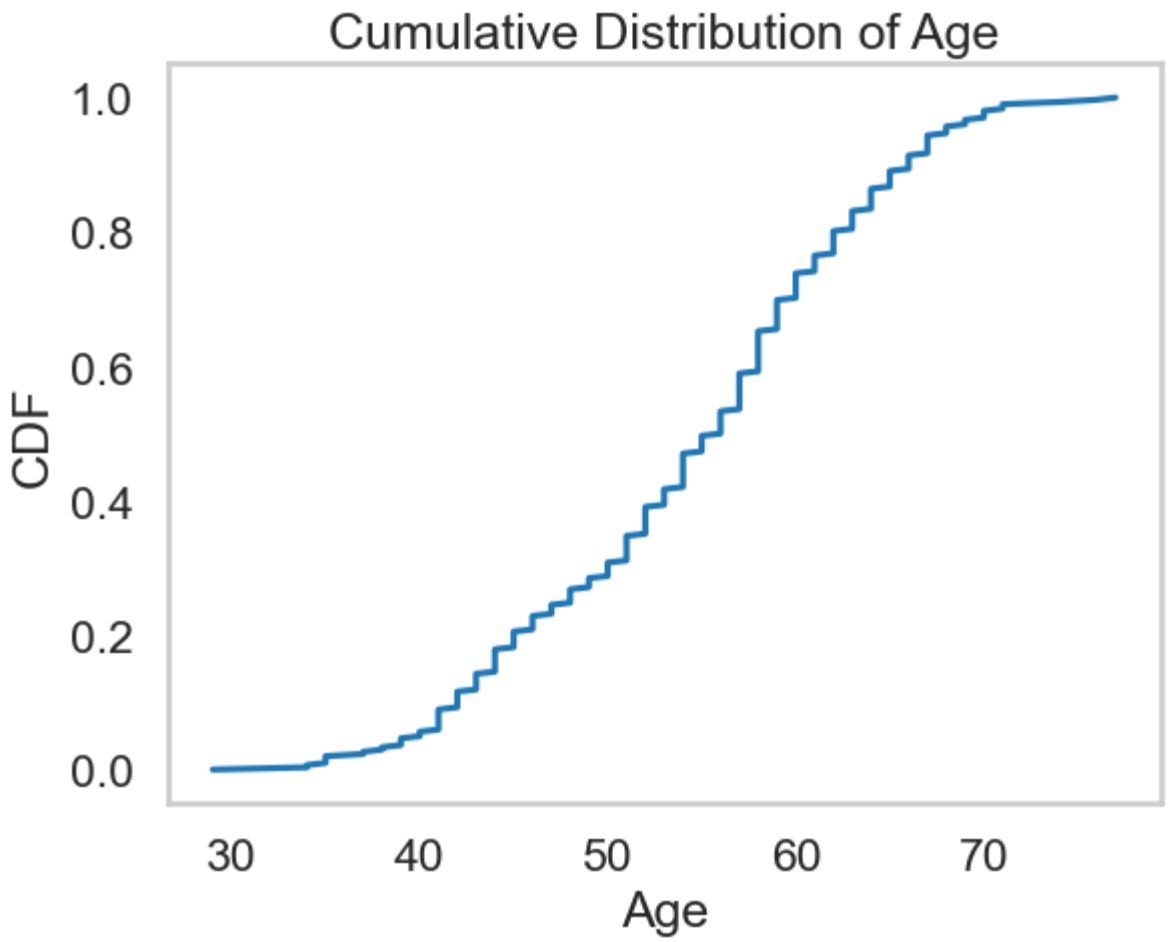
In [16]:
```python
# Feature Engineering


# Identify categorical variables and explore their unique values
categorical_columns = [col for col in X.columns if X[col].dtype == 'object']
print("Categorical columns and their unique values:")
for col in categorical_columns:
    print(f"{col} unique values: {X[col].unique()}")

# One-hot encode the categorical variables
X_encoded = pd.get_dummies(X, columns=categorical_columns, drop_first=True)
print("\nData after encoding categorical features:")
print(X_encoded.head())

# Feature Scaling
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_encoded)
X_scaled = pd.DataFrame(X_scaled, columns=X_encoded.columns)
print("\nData after scaling:")
print(X_scaled.head())
```

```
Categorical columns and their unique values:

Data after encoding categorical features:
   age  sex  cp  trestbps  chol  fbs  restecg  thalach  exang  oldpeak  slo
pe  \
0   63    1   1       145   233    1        2      150      0      2.3
3
1   67    1   4       160   286    0        2      108      1      1.5
2
2   67    1   4       120   229    0        2      129      1      2.6
2
3   37    1   3       130   250    0        0      187      0      3.5
3
4   41    0   2       130   204    0        2      172      0      1.4
1

    ca  thal
0  0.0   6.0
1  3.0   3.0
2  2.0   7.0
3  0.0   3.0
4  0.0   3.0

Data after scaling:
        age       sex        cp  trestbps      chol       fbs   restecg  \
0  0.948726  0.686202 -2.251775  0.757525 -0.264900  2.394438  1.016684
1  1.392002  0.686202  0.877985  1.611220  0.760415 -0.417635  1.016684
2  1.392002  0.686202  0.877985 -0.665300 -0.342283 -0.417635  1.016684
3 -1.932564  0.686202 -0.165268 -0.096170  0.063974 -0.417635 -0.996749
4 -1.489288 -1.457296 -1.208521 -0.096170 -0.825922 -0.417635  1.016684

    thalach     exang   oldpeak     slope        ca      thal
0  0.017197 -0.696631  1.087338  2.274579 -0.711131  0.660004
1 -1.821905  1.435481  0.397182  0.649113  2.504881 -0.890238
2 -0.902354  1.435481  1.346147  0.649113  1.432877  1.176752
3  1.637359 -0.696631  2.122573  2.274579 -0.711131 -0.890238
4  0.980537 -0.696631  0.310912 -0.976352 -0.711131 -0.890238
```

In [17]:
```
# Density Plot Overlaid with Histogram: for 'age' feature
sns.histplot(X['age'], kde=True, color='dodgerblue', bins=20, stat='density'
plt.title('Age Distribution with Density Plot')
plt.xlabel('Age')
plt.ylabel('Density')
plt.show()
```



In [18]:
```
# Cumulative Distribution Function (CDF): for 'age' feature
age_sorted = np.sort(X['age'])
p = np.arange(len(age_sorted)) / (len(age_sorted) - 1)
plt.plot(age_sorted, p)
plt.title('Cumulative Distribution of Age')
plt.xlabel('Age')
plt.ylabel('CDF')
plt.grid()
plt.show()
```



In [19]:
```
# Splitting the data

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_binary, test
print(f"Training set size: {X_train.shape[0]} samples")
print(f"Testing set size: {X_test.shape[0]} samples")
```

```
Training set size: 212 samples
Testing set size: 91 samples
```

In [20]:
```
# Building and training the model

model = LogisticRegression()

# Train the model
model.fit(X_train, y_train)

# Predict on the test set
y_pred = model.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy of the logistic regression model: {accuracy:.2f}")
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred))
print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```

```
Accuracy of the logistic regression model: 0.85
Confusion Matrix:
[[39  9]
 [ 5 38]]

Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.81      0.85        48
           1       0.81      0.88      0.84        43

    accuracy                           0.85        91
   macro avg       0.85      0.85      0.85        91
weighted avg       0.85      0.85      0.85        91
```

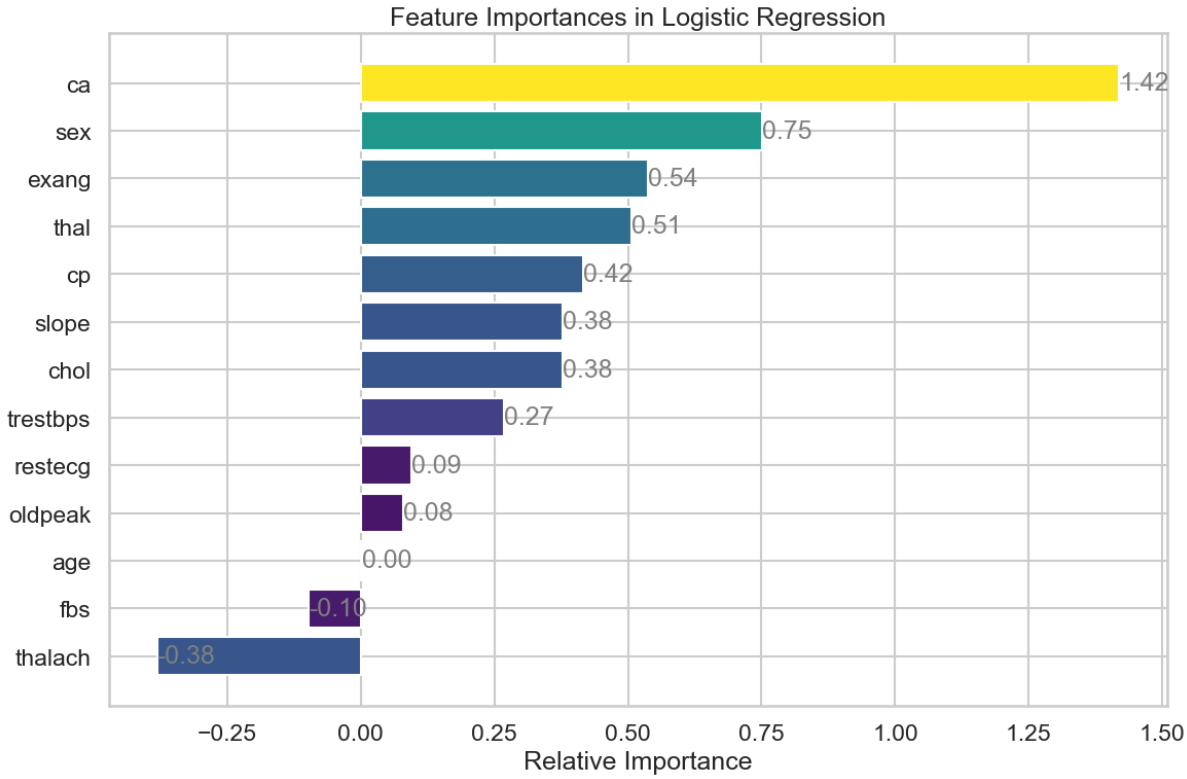
In [21]:
```
# Cross-Validation

scores = cross_val_score(model, X_scaled, y_binary, cv=10)  # 10-fold cross-
print("Cross-validation scores:", scores)
print("Average cross-validation score: {:.2f}".format(scores.mean()))
```

```
Cross-validation scores: [0.87096774 0.80645161 0.80645161 0.96666667 0.8
0.66666667
 0.86666667 0.83333333 0.7        0.86666667]
Average cross-validation score: 0.82
```

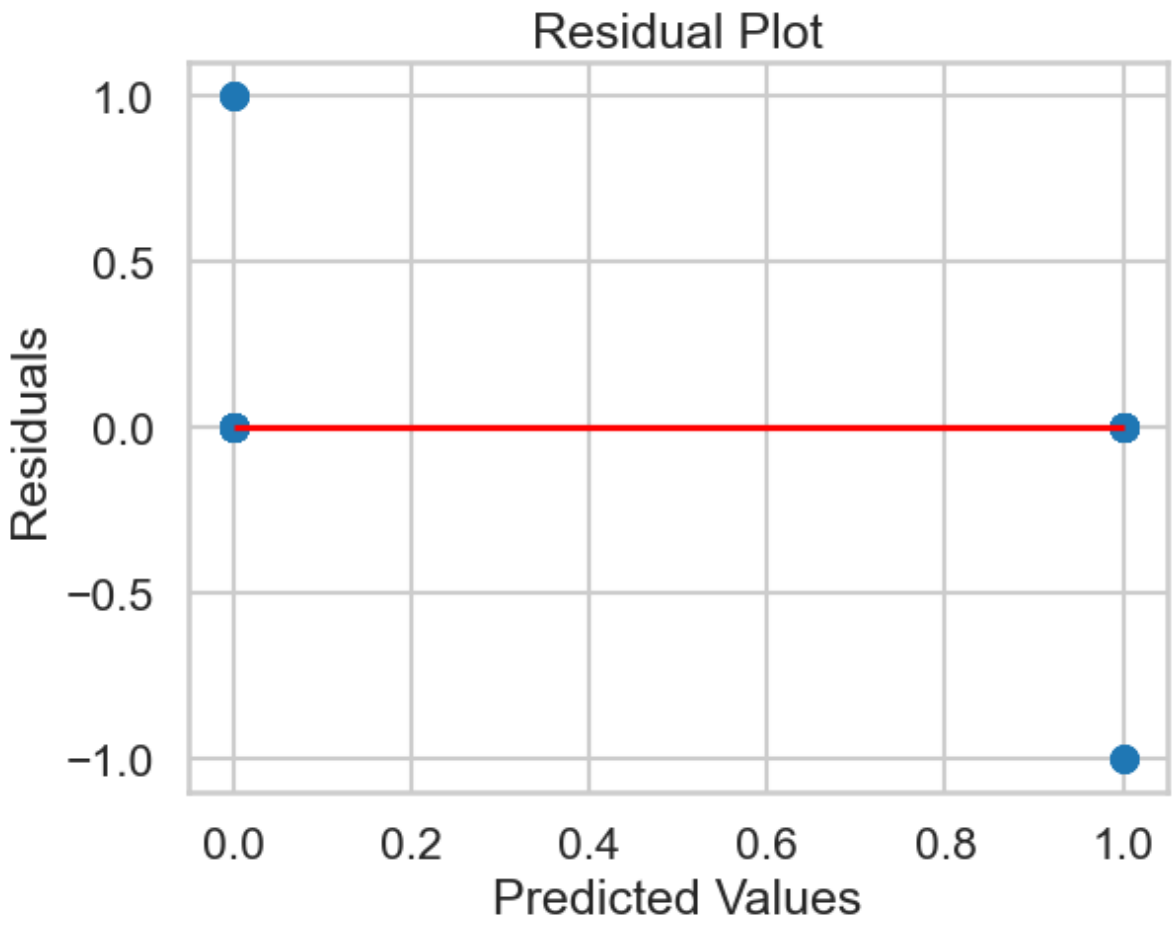
In [22]:
```
# Re-train and Evaluate the Model with Optimal Parameters

optimal_model = LogisticRegression(C=0.01, solver='lbfgs')
optimal_model.fit(X_train, y_train)
y_pred_optimal = optimal_model.predict(X_test)

# Evaluate the optimized model
accuracy_optimal = accuracy_score(y_test, y_pred_optimal)
print(f"Optimized model accuracy: {accuracy_optimal:.2f}")
print("Confusion Matrix:")
print(confusion_matrix(y_test, y_pred_optimal))
print("\nClassification Report:")
print(classification_report(y_test, y_pred_optimal))
```

```
Optimized model accuracy: 0.86
Confusion Matrix:
[[43  5]
 [ 8 35]]

Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.90      0.87        48
           1       0.88      0.81      0.84        43

    accuracy                           0.86        91
   macro avg       0.86      0.85      0.86        91
weighted avg       0.86      0.86      0.86        91
```

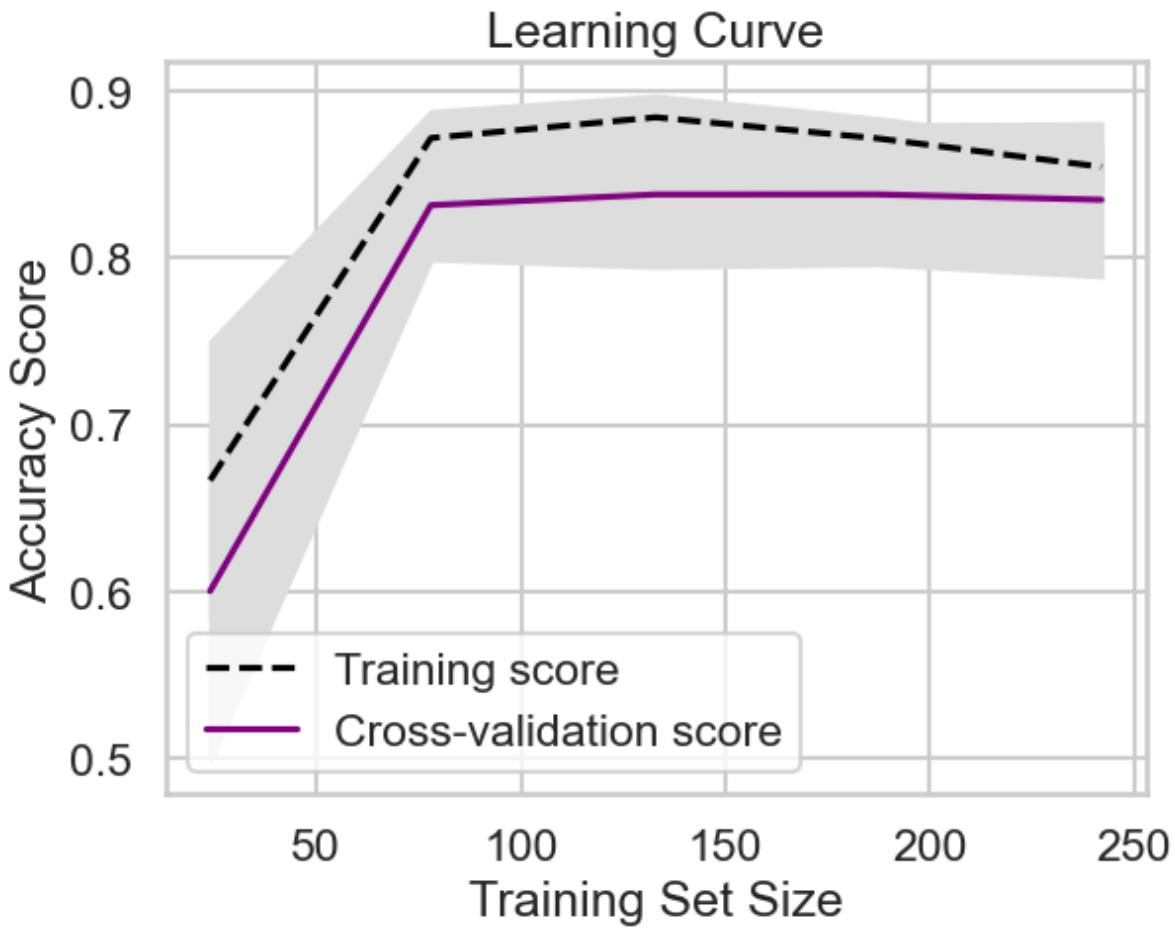
In [23]:
```
# Visualizing Feature Importance
importance = model.coef_[0]
features = X_encoded.columns
indices = np.argsort(importance)

# Choose a color palette
colors = plt.cm.viridis(np.abs(importance[indices]) / np.max(np.abs(importar

plt.figure(figsize=(12, 8))
plt.barh(range(len(indices)), importance[indices], color=colors)
plt.yticks(range(len(indices)), [features[i] for i in indices])
plt.xlabel('Relative Importance')
plt.title('Feature Importances in Logistic Regression')

# Annotate the bars with the coefficient values
for i in range(len(indices)):
    plt.text(importance[indices][i], i, f"{importance[indices][i]:.2f}", col

plt.tight_layout()
plt.show()
```



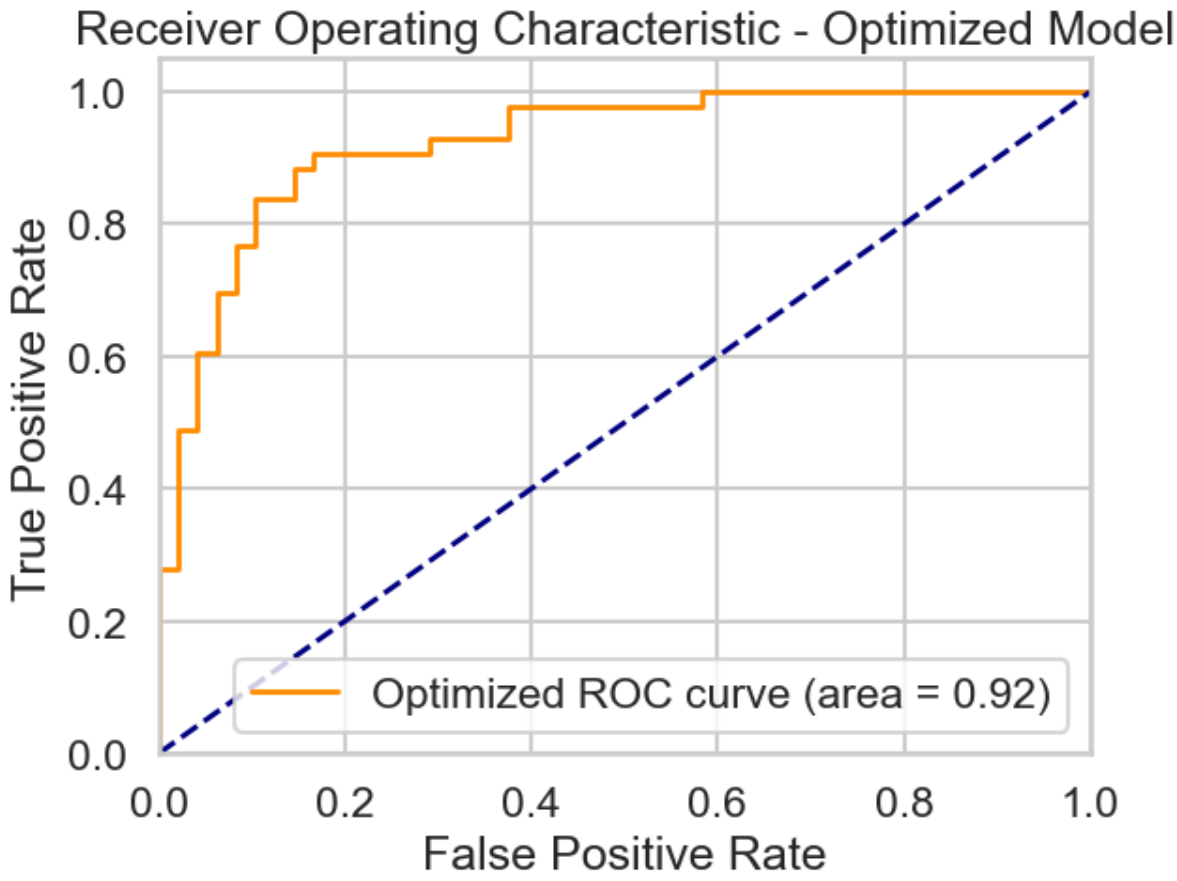
In [24]:
```
# Residual Plot
residuals = y_test - y_pred_optimal

plt.figure()
plt.scatter(y_pred_optimal, residuals)
plt.hlines(y=0, xmin=y_pred_optimal.min(), xmax=y_pred_optimal.max(), colors
plt.title('Residual Plot')
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.show()
```



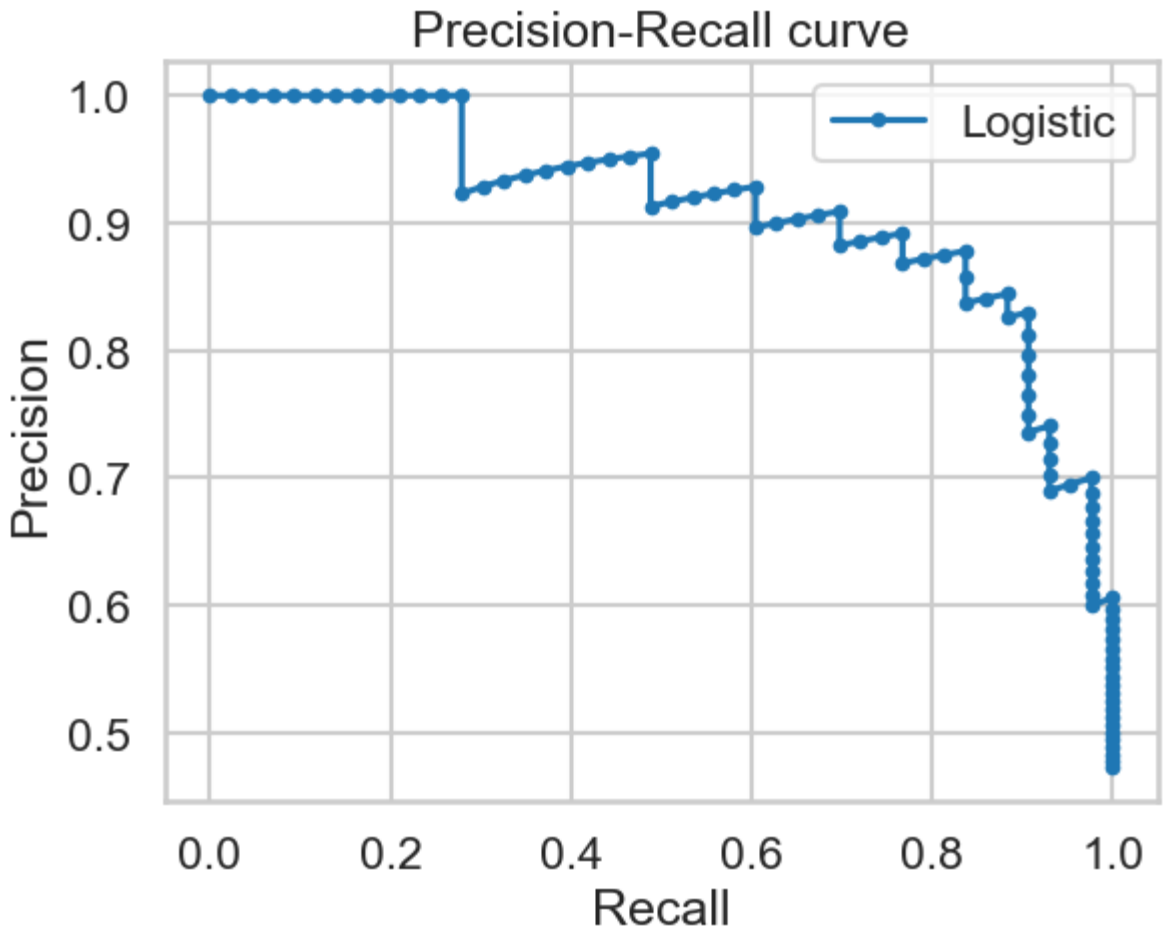
In [25]:
```
train_sizes, train_scores, test_scores = learning_curve(optimal_model, X_sca

# Calculate mean and standard deviation for training set scores
train_mean = np.mean(train_scores, axis=1)
train_std = np.std(train_scores, axis=1)

# Calculate mean and standard deviation for test set scores
test_mean = np.mean(test_scores, axis=1)
test_std = np.std(test_scores, axis=1)

plt.figure()
plt.plot(train_sizes, train_mean, '--', color="black", label="Training score
plt.plot(train_sizes, test_mean, color="purple", label="Cross-validation sco

# Draw bands
plt.fill_between(train_sizes, train_mean - train_std, train_mean + train_std
plt.fill_between(train_sizes, test_mean - test_std, test_mean + test_std, co

plt.title('Learning Curve')
plt.xlabel('Training Set Size')
plt.ylabel('Accuracy Score')
plt.legend(loc="best")
plt.show()
```



In [26]:
```
# ROC Curve and AUC

# Compute probabilities for the AUC score
y_scores_optimal = optimal_model.decision_function(X_test)
fpr_optimal, tpr_optimal, _ = roc_curve(y_test, y_scores_optimal)
roc_auc_optimal = auc(fpr_optimal, tpr_optimal)

# Plot ROC curve for the optimized model
plt.figure()
plt.plot(fpr_optimal, tpr_optimal, color='darkorange', lw=2, label='Optimize
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic - Optimized Model')
plt.legend(loc="lower right")
plt.show()
```



In [27]:
```
# Precision-Recall Curve
from sklearn.metrics import precision_recall_curve
precision, recall, _ = precision_recall_curve(y_test, y_scores_optimal)

plt.figure()
plt.plot(recall, precision, marker='.', label='Logistic')
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title('Precision-Recall curve')
plt.legend()
plt.show()
```



In [28]:
```
# Evaluating Model Sensitivity to Data Partitioning

# Cross-validation with StratifiedKFold
skf = StratifiedKFold(n_splits=5)
scores = cross_val_score(model, X_scaled, y_binary, cv=skf, scoring='accurac
print("Cross-validation scores:", scores)
print("Average score across different partitions:", scores.mean())
```

```
Cross-validation scores: [0.85245902 0.8852459  0.78688525 0.81666667 0.8
]
Average score across different partitions: 0.8282513661202184
```

In [ ]: