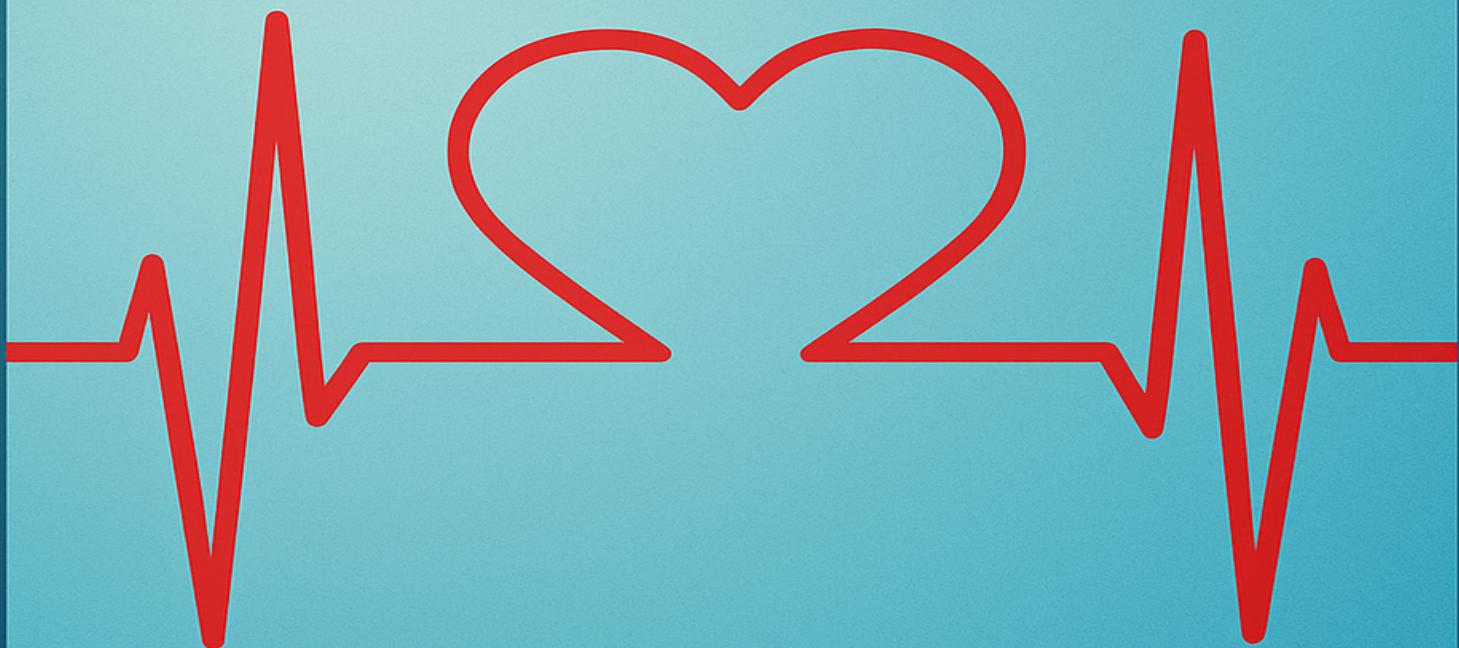


Forecasting Heart Disease Risks



Seif H. Kungulio

Forecasting Heart Disease Risks

Seif Kungulio

November 20, 2025

Contents

Business Understanding	4
Introduction	4
Problem statement	4
 Data Understanding	 5
Data description	5
Data dictionary	5
Initial observations	6
 Data Preparation	 7
Data loading	7
Data preprocessing	9
Helper functions	11
Exploratory data analysis	13
Class imbalance	13
Boxplots for numerical variables	13
Barplots for categorical variables	16
Histograms for Numerical Variables	21
Scatterplots for numerical variables	24
Pairwise correlation plots	27
Correlation matrix	28
Splitting the dataset	29
Scale numeric variables	29

Business Understanding

Introduction

Cardiovascular diseases remain one of the leading causes of death worldwide, imposing significant clinical and financial burdens on individuals and organizations alike. Early identification of people at risk of developing heart disease is therefore a priority not only in healthcare but also in sectors such as insurance and public health policy. For insurance companies in particular, understanding the likelihood that a policyholder will develop heart disease is crucial for assessing health risks, pricing premiums, and designing preventive wellness programs that promote healthier lifestyles among clients.

This project applies predictive analytics and machine learning techniques to the Heart Disease dataset from the UCI Machine Learning Repository. The dataset contains a diverse range of patient attributes—including demographic, physiological, and clinical indicators—that are known to influence cardiovascular health. By systematically analyzing these variables, the project aims to uncover patterns and risk factors associated with the onset of heart disease.

The broader business objective is to build a data-driven decision support system that can forecast an individual's probability of developing heart disease. Such a model enables insurers to perform risk stratification, design personalized premium plans, and support preventive health initiatives that lower long-term costs. Through this analytical framework, the project demonstrates how data science can transform raw medical data into actionable business intelligence, fostering both economic efficiency and social impact.

Problem statement

Problem statement

To develop models for an insurance company using the Heart Disease dataset from the UCI Machine Learning Repository. The goal is to predict the likelihood of a person developing heart disease, which would help the insurance company estimate health risks and adjust premiums accordingly.

Data Understanding

The dataset contains various features related to patients' health and demographic information. We will explore the dataset to understand its structure and relationships between variables.

Data description

The Heart Disease dataset from the UCI Machine Learning Repository contains 303 instances and 14 attributes. These attributes include both numerical and categorical variables related to patients' health metrics and demographic information. The target variable indicates the presence or absence of heart disease. These attributes are:

1. **age**: Age of the patient (numeric)
2. **sex**: Gender of the patient (1 = male, 0 = female)
3. **cp**: Chest pain type (categorical: 1-4)
4. **trestbps**: Resting blood pressure (numeric)
5. **chol**: Serum cholesterol (numeric)
6. **fbs**: Fasting blood sugar (1 = true, 0 = false)
7. **restecg**: Resting electrocardiographic results (categorical)
8. **thalach**: Maximum heart rate achieved (numeric)
9. **exang**: Exercise-induced angina (1 = yes, 0 = no)
10. **oldpeak**: ST depression induced by exercise (numeric)
11. **slope**: The slope of the peak exercise ST segment (categorical)
12. **ca**: Number of major vessels (0-3, numeric)
13. **thal**: Thalassemia (categorical: 1 = normal, 2 = fixed defect, 3 = reversible defect)
14. **target**: Heart disease (1 = disease, 0 = no disease)

Data dictionary

The dataset contains 14 key attributes that are either numerical or categorical.

Attribute	Type	Description	Constraints/ Rules
age	Numerical	The age of the patient in years	Range: 29-77 (based on dataset statistics)
sex	Categorical	The gender of the patient	Values: 1 = Male, 0 = Female
cp	Categorical	Type of chest pain experienced by the patient	Values: 1 = Typical angina, 2 = Atypical angina, 3 = Non-anginal pain, 4 = Asymptomatic
trestbps	Numerical	Resting blood pressure of the patient, measured in mmHg	Range: Typically, between 94 and 200 mmHg
chol	Numerical	Serum cholesterol level in mg/dl	Range: Typically, between 126 and 564 mg/dl
fbs	Categorical	Fasting blood sugar level > 120 mg/dl	Values: 1 = True, 0 = False
restecg	Categorical	Results of the patient's resting electrocardiogram	Values: 0 = Normal, 1 = ST-T wave abnormality, 2 = Probable or definite left ventricular hypertrophy

Attribute	Type	Description	Constraints/ Rules
thalach	Numerical	Maximum heart rate achieved during a stress test	Range: Typically, between 71 and 202 bpm
exang	Categorical	Whether the patient experiences exercise-induced angina	Values: 1 = Yes, 0 = No
oldpeak	Numerical	ST depression induced by exercise relative to rest (an ECG measure)	Range: 0.0 to 6.2 (higher values indicate more severe abnormalities)
slope	Categorical	Slope of the peak exercise ST segment	Values: 1 = Upsloping, 2 = Flat, 3 = Downsloping
ca	Numerical	Number of major vessels colored by fluoroscopy	Range: 0-3
thal	Categorical	Blood disorder variable related to thalassemia	Values: 3 = Normal, 6 = Fixed defect, 7 = Reversible defect
target	Categorical	Diagnosis of heart disease	Values: 0 = No heart disease, 1 = Presence of heart disease

Initial observations

- The dataset contains a mix of numerical and categorical variables.
- Some variables may require preprocessing, such as handling missing values and encoding categorical variables.
- Missing Values: Some fields like ca and thal may have missing values or unknown entries ('?').
- Data Types: Some categorical variables are encoded numerically and will need to be interpreted correctly during analysis.
- Class Imbalance: Preliminary checks suggest the dataset is relatively balanced between presence and absence of disease, but this will be verified.
- Outliers: Numerical fields such as chol (cholesterol) and trestbps (blood pressure) may have outliers that need to be detected and considered in analysis.

Data Preparation

Data loading

Load the dataset from the UCI website to memory

```
Heart.df <- read.csv(text = getURL(url), header = FALSE, na.strings = "?")
```

When I loaded the dataset using the `read.csv()` function, I made sure to convert the placeholder "?" values into proper `NA` entries. This was important because the original UCI file embeds missing values as question marks, which R would otherwise treat as literal characters. By handling this upfront, I ensured that all subsequent preprocessing steps—such as imputation, transformations, and statistical summaries—would correctly recognize missing data. This step was the foundation that allowed me to work with a clean and interpretable dataset from the very beginning.

Display dimensions of the dataset

```
dim(Heart.df)
```

```
## [1] 303 14
```

After loading the data, I checked the dimensions and confirmed that the dataset contained **303 records and 14 variables**. For me, this was a critical validation step because it reassured me that the dataset had been imported completely, without truncation or corruption. Knowing the exact dimensions also helped me plan the appropriate modeling strategy. With 303 observations, I recognized that I needed to be cautious about overfitting and choose models that perform well on modest-sized datasets.

Display the first six rows of the dataset

```
head(Heart.df)
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14
## 1 63  1  1 145 233  1  2 150  0 2.3   3   0   6   0
## 2 67  1  4 160 286  0  2 108  1 1.5   2   3   3   2
## 3 67  1  4 120 229  0  2 129  1 2.6   2   2   7   1
## 4 37  1  3 130 250  0  0 187  0 3.5   3   0   3   0
## 5 41  0  2 130 204  0  2 172  0 1.4   1   0   3   0
## 6 56  1  2 120 236  0  0 178  0 0.8   1   0   3   0
```

Inspecting the first few rows of the dataset gave me an initial sense of its structure in raw form. The variables were still labeled generically as `V1` through `V14`, and I could already see that categorical fields—such as `ca` and `thal`—contained "?" values. This preview confirmed that data quality issues were present and would require careful cleaning. It also highlighted that the dataset treated all variables as numeric values, even when some of them clearly represented categories. Seeing these patterns early helped me anticipate the transformations and encodings that would be needed later.

Rename the columns into a meaningful column names

```
colnames(Heart.df) <- c("age", "sex", "cp", "trestbps", "chol", "fbs", "restecg",
                         "thalach", "exang", "oldpeak", "slope", "ca", "thal", "target")
```

Renaming the default column headers to clinically meaningful names was an important step for both clarity and accuracy. By assigning labels such as `age`, `trestbps`, `chol`, and `thalach`, I transformed an ambiguous numeric matrix into a dataset with clearly defined medical variables. This made my analysis much more intuitive and significantly reduced the risk of errors during visualization or modeling. From this point forward, every plot, summary, and model became easier to interpret in a clinical and analytical context.

Display the structure of the dataframe

```
glimpse(Heart.df)
```

```
## Rows: 303
## Columns: 14
## $ age      <dbl> 63, 67, 67, 37, 41, 56, 62, 57, 63, 53, 57, 56, 56, 44, 52, 5~
## $ sex      <dbl> 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1~
## $ cp       <dbl> 1, 4, 4, 3, 2, 2, 4, 4, 4, 4, 2, 3, 2, 3, 3, 2, 4, 3, 2, 1~
## $ trestbps <dbl> 145, 160, 120, 130, 130, 120, 140, 120, 130, 140, 140, 140, 140, 1~
## $ chol     <dbl> 233, 286, 229, 250, 204, 236, 268, 354, 254, 203, 192, 294, 2~
## $ fbs      <dbl> 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0~
## $ restecg  <dbl> 2, 2, 2, 0, 2, 0, 2, 0, 2, 2, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 2~
## $ thalach  <dbl> 150, 108, 129, 187, 172, 178, 160, 163, 147, 155, 148, 153, 1~
## $ exang    <dbl> 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1~
## $ oldpeak  <dbl> 2.3, 1.5, 2.6, 3.5, 1.4, 0.8, 3.6, 0.6, 1.4, 3.1, 0.4, 1.3, 0~
## $ slope    <dbl> 3, 2, 2, 3, 1, 1, 3, 1, 2, 3, 2, 2, 2, 1, 1, 1, 3, 1, 1, 1, 2~
## $ ca       <dbl> 0, 3, 2, 0, 0, 0, 2, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ thal     <dbl> 6, 3, 7, 3, 3, 3, 3, 7, 7, 6, 3, 6, 7, 7, 3, 7, 3, 3, 3, 3~
## $ target   <int> 0, 2, 1, 0, 0, 0, 3, 0, 2, 1, 0, 0, 2, 0, 0, 1, 0, 0, 0, 0, 0~
```

Using `glimpse()`, I inspected the internal structure of the dataset and confirmed that all variables were loaded as numeric types. Although this is typical for UCI data, it reinforced the need to convert several columns—such as `sex`, `cp`, `exang`, `restecg`, and `thal`—into factors. Additionally, I noticed that the `target` variable contained values beyond just 0 and 1. This told me that the dataset initially encoded heart disease severity on a scale rather than a simple binary indicator. Recognizing this early helped me plan the binary conversion that would later make the dataset suitable for classification models.

Display the statistical summary of the dataframe

```
summary(Heart.df)
```

```
##      age          sex          cp        trestbps
##  Min.   :29.00   Min.   :0.0000   Min.   :1.000   Min.   : 94.0
##  1st Qu.:48.00  1st Qu.:0.0000  1st Qu.:3.000  1st Qu.:120.0
##  Median :56.00  Median :1.0000  Median :3.000  Median :130.0
##  Mean   :54.44  Mean   :0.6799  Mean   :3.158  Mean   :131.7
##  3rd Qu.:61.00  3rd Qu.:1.0000  3rd Qu.:4.000  3rd Qu.:140.0
##  Max.   :77.00  Max.   :1.0000  Max.   :4.000  Max.   :200.0
##
##      chol          fbs          restecg         thalach
##  Min.   :126.0   Min.   :0.0000   Min.   :0.0000   Min.   : 71.0
##  1st Qu.:211.0  1st Qu.:0.0000  1st Qu.:0.0000  1st Qu.:133.5
##  Median :241.0  Median :0.0000  Median :1.0000  Median :153.0
##  Mean   :246.7  Mean   :0.1485  Mean   :0.9901  Mean   :149.6
##  3rd Qu.:275.0  3rd Qu.:0.0000  3rd Qu.:2.0000  3rd Qu.:166.0
##  Max.   :564.0  Max.   :1.0000  Max.   :2.0000  Max.   :202.0
##
```

```

##      exang          oldpeak         slope          ca
##  Min.   :0.0000   Min.   :0.00   Min.   :1.000   Min.   :0.0000
##  1st Qu.:0.0000  1st Qu.:0.00   1st Qu.:1.000  1st Qu.:0.0000
##  Median :0.0000  Median :0.80   Median :2.000  Median :0.0000
##  Mean   :0.3267  Mean   :1.04   Mean   :1.601  Mean   :0.6722
##  3rd Qu.:1.0000  3rd Qu.:1.60   3rd Qu.:2.000  3rd Qu.:1.0000
##  Max.   :1.0000  Max.   :6.20   Max.   :3.000  Max.   :3.0000
##                NA's   :4
##
##      thal          target
##  Min.   :3.000   Min.   :0.0000
##  1st Qu.:3.000  1st Qu.:0.0000
##  Median :3.000  Median :0.0000
##  Mean   :4.734  Mean   :0.9373
##  3rd Qu.:7.000  3rd Qu.:2.0000
##  Max.   :7.000  Max.   :4.0000
##  NA's   :2

```

The summary statistics provided the first quantitative snapshot of the dataset. For instance, I observed that age ranged from 29 to 77, cholesterol included extremely high values reaching 564 mg/dl, and the **ca** and **thal** variables contained missing entries. The summaries also revealed skewness in variables like **oldpeak**, which aligns with the clinical behavior of ST depression measurements. Seeing these descriptive statistics helped me identify potential outliers and underscored the importance of filtering and standardizing certain variables. It also confirmed that the target variable included multiple values, further reinforcing the need for recoding before building predictive models.

Data preprocessing

We will preprocess the data by handling missing values, encoding categorical variables, and scaling numerical features.

Convert binary variables to (0, 1)

According to the data dictionary, the following attributes should be binary variables: **sex**, **fbs**, **exang**, and **target**. But, some shows to have values besides 0's and 1's. Let's convert binary variables to (0, 1)

```

Heart.df$target <- ifelse(Heart.df$target > 0, 1, 0)
Heart.df$sex <- ifelse(Heart.df$sex > 0, 1, 0)
Heart.df$fbs <- ifelse(Heart.df$fbs > 0, 1, 0)
Heart.df$exang <- ifelse(Heart.df$exang > 0, 1, 0)

```

Handle missing values

Handle missing values in **ca** and **thal** variables using mean/mode imputation.

```

Heart.df$ca[is.na(Heart.df$ca)] <- median(Heart.df$ca, na.rm = TRUE)
Heart.df$ca[Heart.df$ca == "?"] <- median(Heart.df$ca, na.rm = TRUE)
Heart.df$thal[is.na(Heart.df$thal)] <- median(Heart.df$thal, na.rm = TRUE)
Heart.df$thal[Heart.df$thal == "?"] <- median(Heart.df$thal, na.rm = TRUE)

```

Check for missing values if still exist

```
sapply(Heart.df, function(x) sum(is.na(x)))
```

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

Handle duplicate entries

Check for duplicate entries and print them if they exist.

```
dupes <- Heart.df[duplicated(Heart.df) | duplicated(Heart.df, fromLast = TRUE), ]
print(dupes)
```

```
## [1] age      sex      cp      trestbps chol      fbs      restecg      thalach
## [9] exang    oldpeak  slope    ca      thal      target
## <0 rows> (or 0-length row.names)
```

Convert categorical variables to factor.

Define a list of categorical columns with their levels and labels

```
categorical_columns <- list(
  sex = list(levels = c(0, 1),
             labels = c("Female", "Male")),
  cp = list(levels = c(1, 2, 3, 4),
            labels = c("Typical Angina", "Atypical Angina", "Non-Angina",
                      "Asymptomatic")),
  fbs = list(levels = c(0, 1),
             labels = c("False", "True")),
  restecg = list(levels = c(0, 1, 2),
                 labels = c("Normal", "Wave-abnormality", "Probable")),
  exang = list(levels = c(0, 1),
               labels = c("No", "Yes")),
  slope = list(levels = c(1, 2, 3),
               labels = c("Upsloping", "Flat", "Downsloping")),
  thal = list(levels = c(3, 6, 7),
              labels = c("Normal", "Fixed Defect", "Reversible")),
  target = list(levels = c(1, 0),
                labels = c("Yes", "No"))
)
```

Apply the factor transformation using a for-loop.

```
for (col in names(categorical_columns)) {
  Heart.df[[col]] <- factor(Heart.df[[col]],
                            levels = categorical_columns[[col]]$levels,
                            labels = categorical_columns[[col]]$labels)
}
```

Handle outliers in numerical variables

Apply multiple filters to identify and handle outliers in numerical variables.

```
Heart.df <- Heart.df[Heart.df$age > 40 &
                     Heart.df$trestbps < 170 &
                     Heart.df$chol < 340 &
                     Heart.df$chol > 150 &
                     Heart.df$thalach > 115 &
                     Heart.df$oldpeak < 2.4, ]
```

This section of the analysis performs a crucial data-cleaning step aimed at refining the quality of the dataset before modeling and visualization. The filtering operation applies a set of logical conditions to remove extreme or biologically implausible values from key continuous health indicators such as age, blood pressure, cholesterol, heart rate, and ST depression. By doing so, it ensures that the dataset reflects realistic patient characteristics and minimizes the influence of outliers that could distort statistical interpretation or predictive accuracy.

The first filter, `Heart.df$age > 40`, narrows the focus to patients over 40 years of age. This decision is grounded in clinical reasoning—heart disease is relatively uncommon in younger individuals, and including them could introduce noise rather than insight into cardiovascular risk patterns. The next condition, `Heart.df$trestbps < 170`, restricts resting blood pressure to physiologically typical values, removing excessively high readings that may result from measurement error or rare hypertensive crises.

Similarly, cholesterol values are filtered using two constraints: `Heart.df$chol < 340` and `Heart.df$chol > 150`. This dual boundary ensures that cholesterol readings fall within a realistic clinical range, excluding both unusually low and excessively high values. Extremely high cholesterol levels (above 340 mg/dl) could be outliers due to lab errors or rare genetic conditions, while very low levels (below 150 mg/dl) are equally atypical for this patient population.

The condition `Heart.df$thalach > 115` retains only those patients whose maximum heart rate achieved during exercise falls within a normal performance range. Extremely low thalach values often suggest incomplete stress tests or data entry errors, which could bias the interpretation of cardiovascular efficiency. Finally, `Heart.df$oldpeak < 2.4` removes extreme ST depression values. In clinical terms, oldpeak measures the degree of ST segment depression during exercise, and values beyond 2.4 are uncommon and may represent atypical cardiac events that do not align with general population patterns in the dataset.

Overall, these filters collectively enhance the integrity of the data and the reliability of subsequent analysis. By trimming implausible extremes, the dataset becomes more homogeneous, improving the clarity of boxplots, histograms, and scatterplots generated during exploratory data analysis. Moreover, this targeted filtering supports more stable and interpretable model outcomes by preventing a few extreme observations from disproportionately influencing trends or coefficients. The result is a dataset that better represents realistic health profiles, ultimately strengthening the credibility of insights drawn from the heart disease risk modeling process.

Helper functions

Function to create Box plots

```
HeartDiseaseBoxplot <- function(var1, var2) {
  ggplot(Heart.df, aes(x = .data[[var1]],
                        y = .data[[var2]],
                        fill = .data[[var1]])) +
    geom_boxplot() +
    labs(title = paste("Boxplot of", var2, "by", var1),
          x = var1, y = var2, fill = "Heart Disease")
}
```

The `HeartDiseaseBoxplot()` function is designed to visualize how a numerical health-related variable behaves across the different levels of a categorical grouping variable, typically the heart disease outcome. By placing the grouping variable on the x-axis and a continuous risk factor on the y-axis, the resulting boxplot highlights differences in median values, variability, and outliers for each group. This plot offers a fast and intuitive way to assess how a particular clinical measurement—such as age, blood pressure, cholesterol, maximum heart rate, or ST depression—differs between patients with and without diagnosed heart disease. It plays a critical role in early exploratory analysis, helping uncover potential risk factor patterns that may influence model development later.

Function to create Bar plots

```
HeartDiseaseBar <- function(var) {
  ggplot(Heart.df, aes(x = .data[[var]], fill = target)) +
    geom_bar(position = "dodge") +
    labs(title = paste("Distribution of Heart Disease by", var),
         x = var, fill = "Heart Disease")
}
```

The `HeartDiseaseBar()` function produces a categorical bar chart that compares the counts of heart disease outcomes within each category of a selected variable. This is especially useful for demographic or clinical classification variables such as sex, chest pain type, fasting blood sugar, resting ECG results, exercise-induced angina, slope, or thalassemia status. By breaking down the target variable across these groups, the bar chart reveals which categories show higher or lower observed heart disease prevalence. This insight is valuable in understanding population segments that may require additional attention, and it helps identify categorical predictors that could contribute significantly to a classification model.

Function to create Histograms

```
HeartDiseaseHist <- function(var1) {
  ggplot(Heart.df, aes(x = .data[[var1]], fill = target)) +
    geom_histogram(bins = 15) +
    labs(title = paste("Distribution of", var1),
         x = var1, fill = "Heart Disease")
}
```

The `HeartDiseaseHist()` function creates a histogram for a continuous health variable and overlays the distribution according to heart disease status. This visualization makes it easy to examine the overall shape, skewness, and concentration of values for a risk factor while simultaneously comparing the distribution between patients with and without heart disease. Whether analyzing age, resting blood pressure, cholesterol levels, maximum heart rate achieved, or ST depression, the histogram reveals whether certain ranges of values appear more frequently in either outcome group. This function enhances exploratory data analysis by showing how numerical features behave across the target variable and whether separation patterns exist that the model may later exploit.

Function to create Scatter plots

```
HeartDiseaseScatter <- function(point1, point2){
  ggplot(Heart.df, aes(x = .data[[point1]],
                        y = .data[[point2]],
                        color = target)) +
    geom_point(size = 2) +
    geom_smooth(method = "lm", se = FALSE, color = "blue", formula = y ~ x) +
    labs(title = paste("Scatterplot of", point1, "by", point2),
         x = point1, y = point2, color = "Heart Disease")
}
```

The `HeartDiseaseScatter()` function generates a scatterplot showing the relationship between two numeric variables, with each point colored by heart disease status. By plotting one clinical indicator on the x-axis and another on the y-axis, this visualization helps reveal correlations, clusters, and potential separation between outcome groups. The function also overlays a linear trend line for each group, giving a quick visual summary of the direction and strength of the relationship. This makes the scatterplot especially useful for exploring associations such as age versus cholesterol, heart rate versus ST depression, or blood pressure versus heart rate. Ultimately, this function supports deeper understanding of how pairs of predictors interact and how those interactions relate to the presence of heart disease.

Exploratory data analysis

Class imbalance

```
Heart.df %>% count(target) %>% mutate(pct = n/sum(n))
```

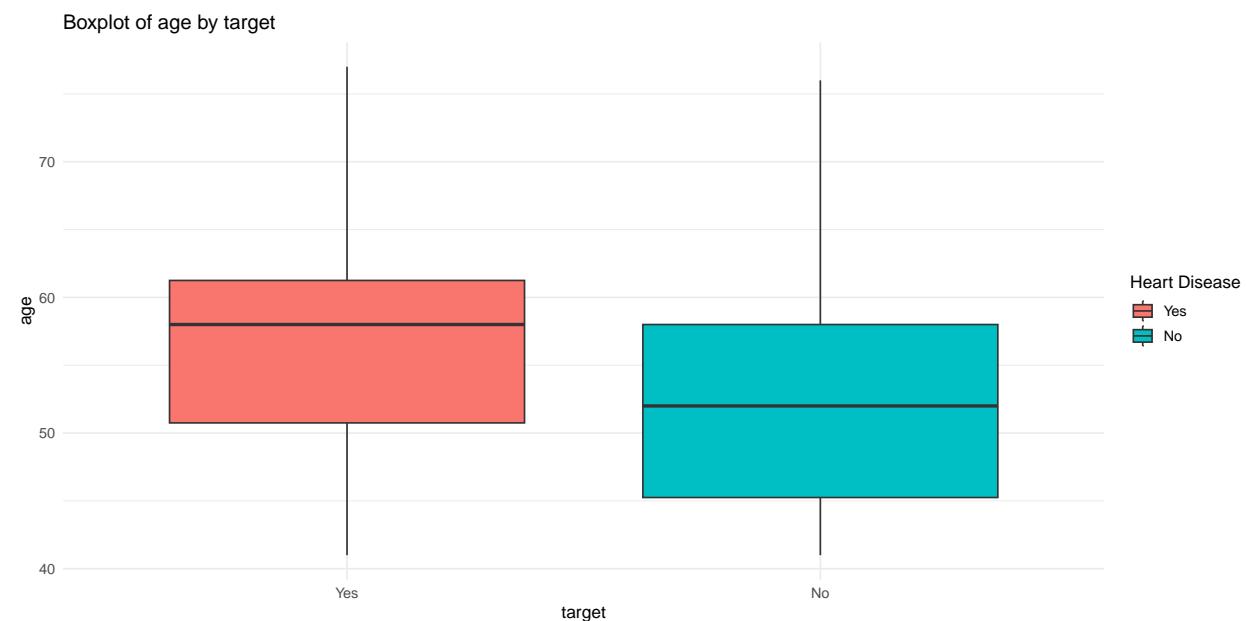
```
##   target     n      pct
## 1 Yes    68 0.3434343
## 2 No    130 0.6565657
```

Boxplots for numerical variables

I used boxplots to visually examine the distribution of key continuous health indicators — such as age, resting blood pressure (trestbps), cholesterol (chol), maximum heart rate (thalach), and ST depression (oldpeak) — across the binary target variable (Heart Disease: Yes / No). Boxplots were chosen because they efficiently highlight differences in central tendency (median), variability (IQR), and the presence of potential outliers between patients with and without heart disease.

Boxplot of Age by Heart Disease

```
HeartDiseaseBoxplot("target", "age")
```

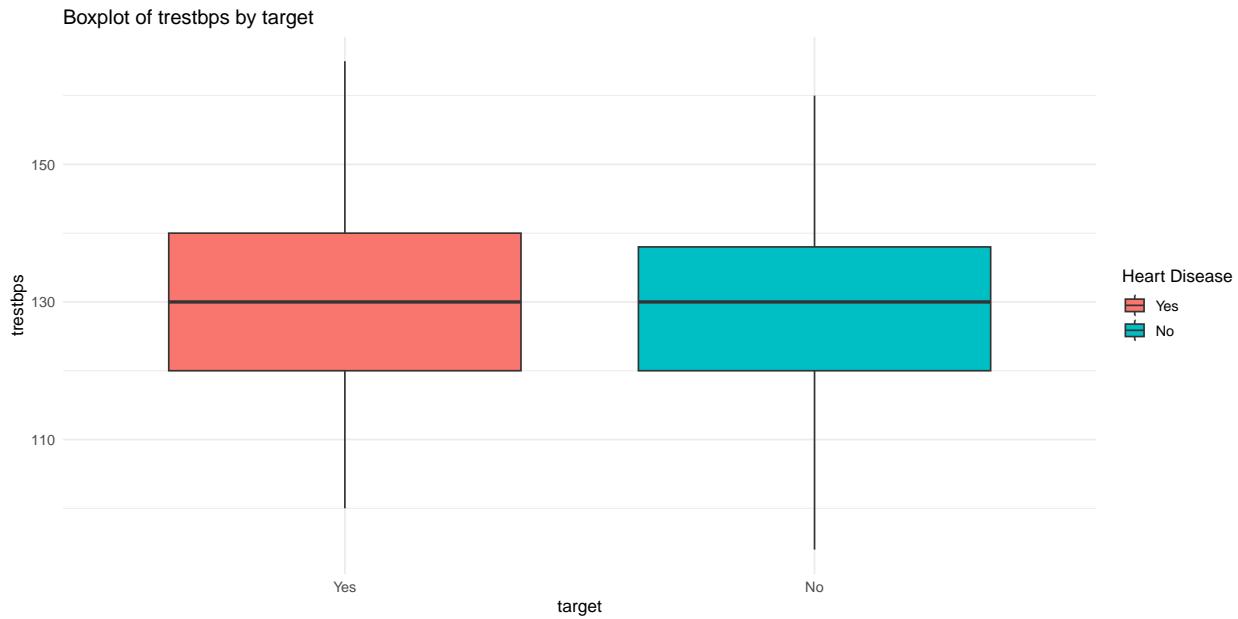


The age boxplot shows that both groups are relatively similar in age distribution, but individuals with

heart disease tend to be slightly older on average. The median age for the “Yes” group appears marginally higher, suggesting that age remains a contributing factor even after filtering out younger patients below 40. However, the overlap in interquartile ranges indicates that age alone does not offer strong separation between the two groups.

Boxplot of Resting Blood Pressure (trestbps) by Heart Disease

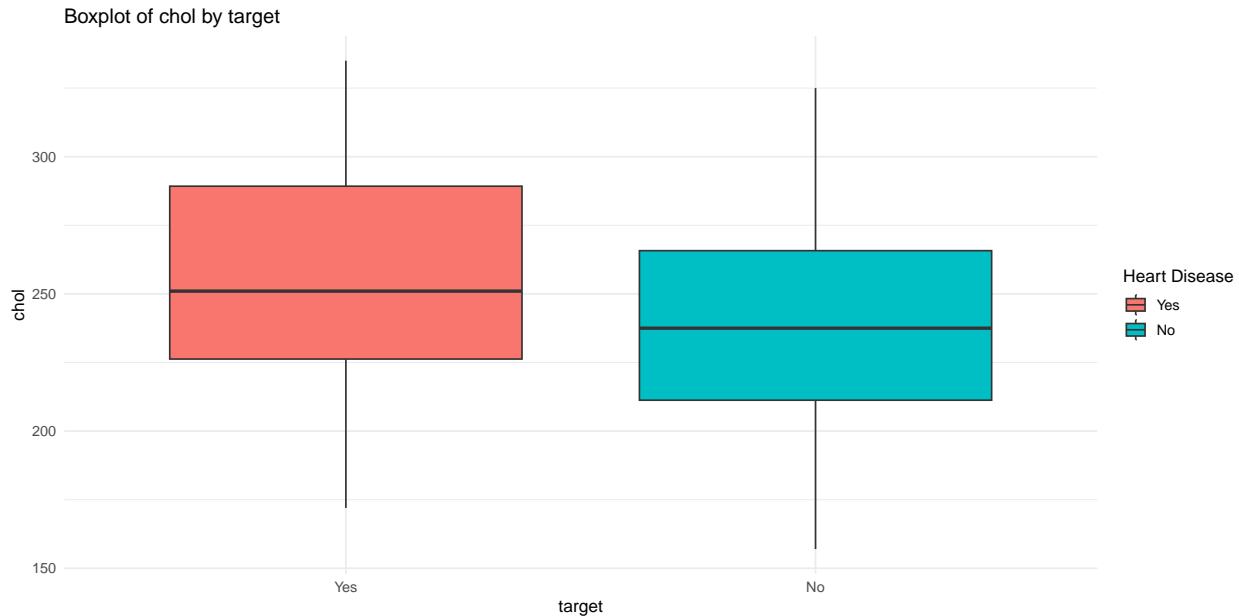
```
HeartDiseaseBoxplot("target", "trestbps")
```



The trestbps boxplot indicates that patients with heart disease have resting blood pressure levels that are generally comparable to those without heart disease. The medians of both groups are close, and their spreads overlap significantly. This suggests that resting blood pressure—after extreme values were filtered—does not strongly differentiate between the two groups.

Boxplot of Cholesterol (chol) by Heart Disease

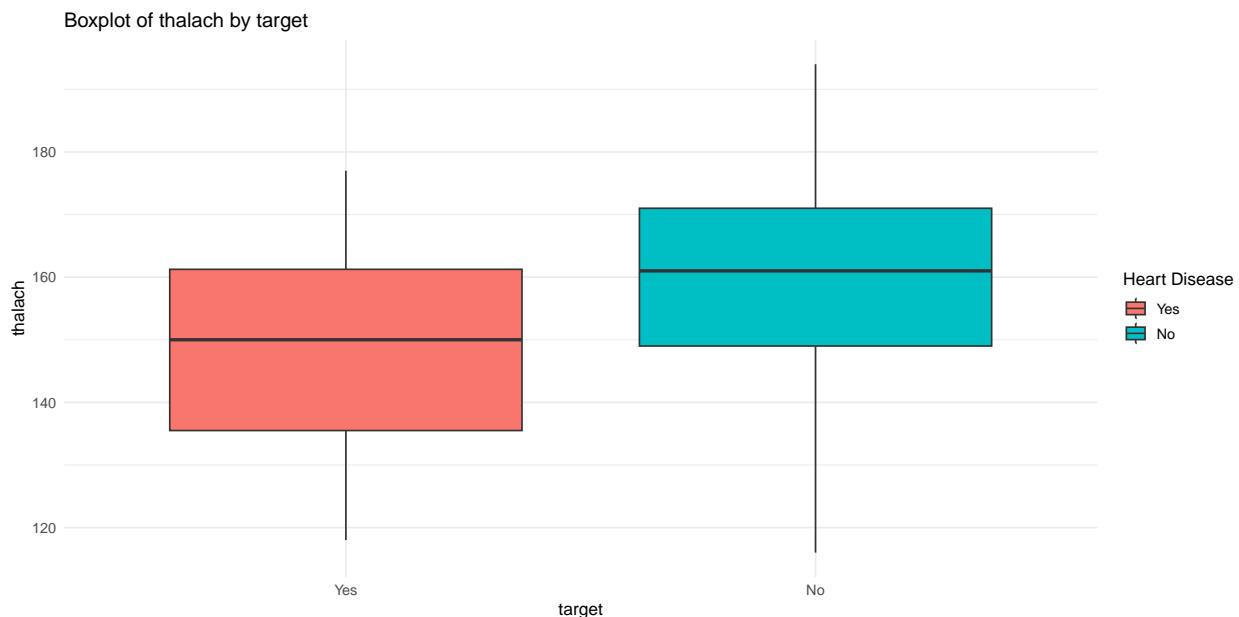
```
HeartDiseaseBoxplot("target", "chol")
```



The cholesterol boxplot shows slightly higher cholesterol values among the heart-disease group, with a marginally higher median level. Although there is considerable overlap in distributions, the “Yes” group tends to include more values toward the upper range. This pattern is consistent with cholesterol being a known cardiovascular risk factor, but not a deterministic one within this dataset.

Boxplot of Maximum Heart Rate Achieved (thalach) by Heart Disease

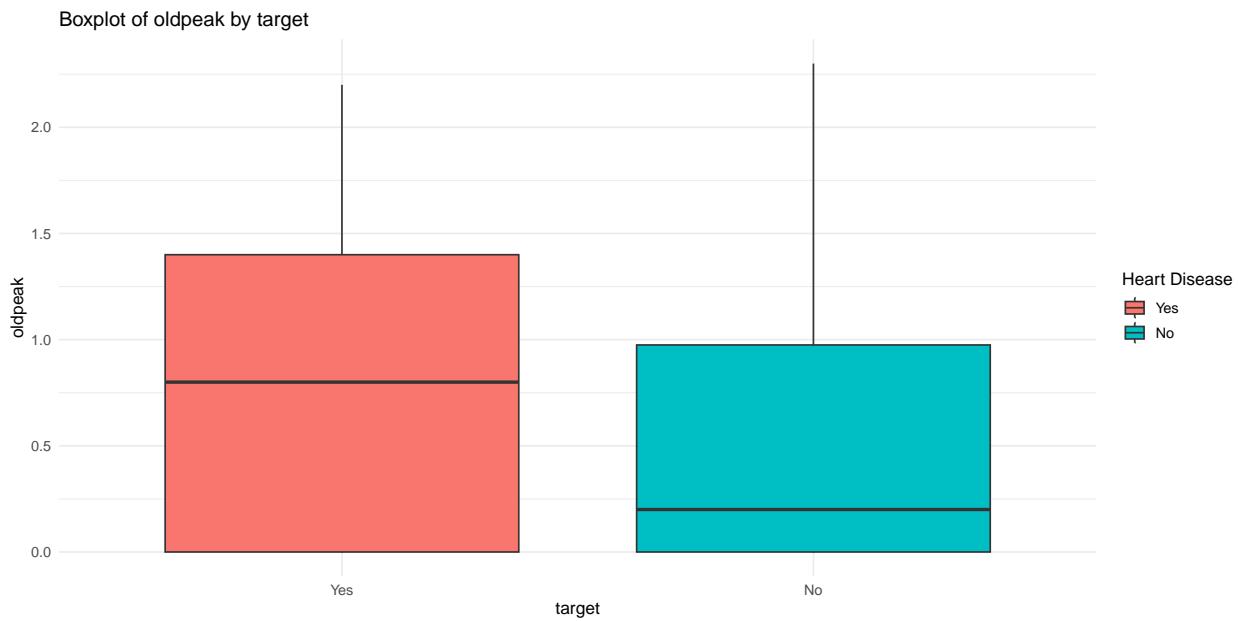
```
HeartDiseaseBoxplot("target", "thalach")
```



The thalach boxplot reveals a clearer distinction than the previous variables: patients with heart disease tend to have lower maximum heart rates achieved during exercise. The median thalach for the “Yes” group is visibly lower, and their distribution is shifted downward relative to the “No” group. This aligns with clinical expectations—reduced cardiac performance often manifests as lower achievable heart rates during stress tests.

Boxplot of ST Depression (oldpeak) by Heart Disease

```
HeartDiseaseBoxplot("target", "oldpeak")
```



This boxplot provides one of the strongest visual separations. Patients with heart disease generally exhibit higher oldpeak values, indicating greater ST depression during exercise. The median for the “Yes” group is noticeably higher, and the upper range of ST depression appears more pronounced. Because ST depression is directly linked to abnormal myocardial response, this variable stands out as a meaningful predictor.

Overall boxplots observations:

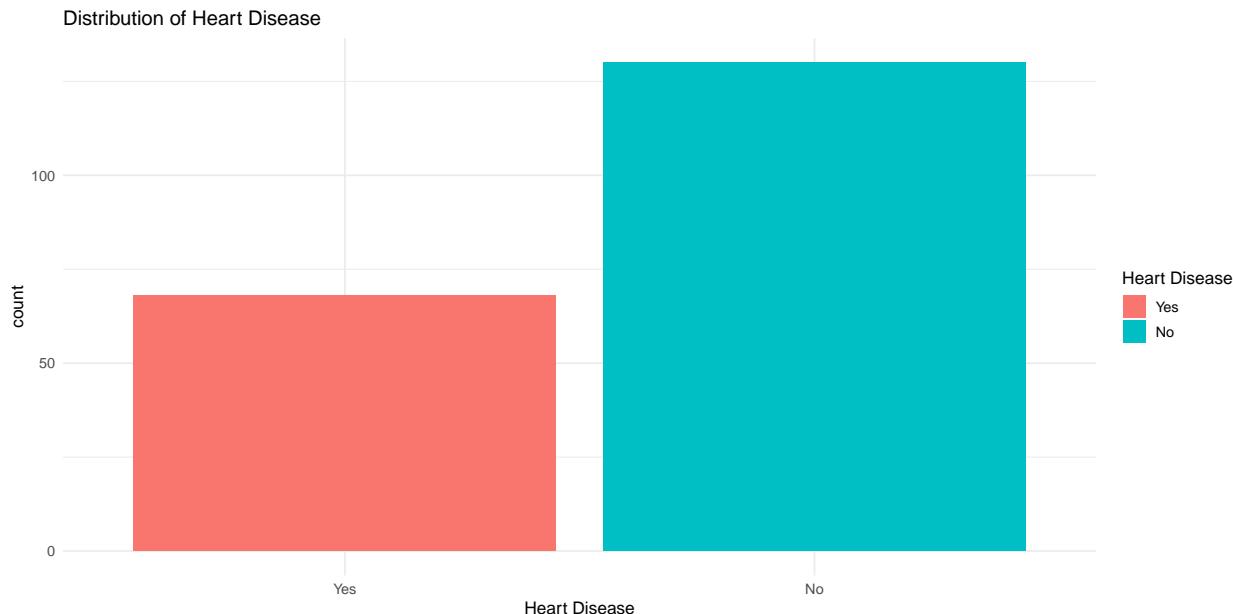
Across all numerical variables, thalach and oldpeak show the most distinct separation between heart-disease and non-heart-disease groups. Age and cholesterol exhibit moderate differentiation, while resting blood pressure shows minimal separation.

These insights support the modeling phase by identifying which numeric predictors carry the strongest signal for classification.

Barplots for categorical variables

Heart disease distribution

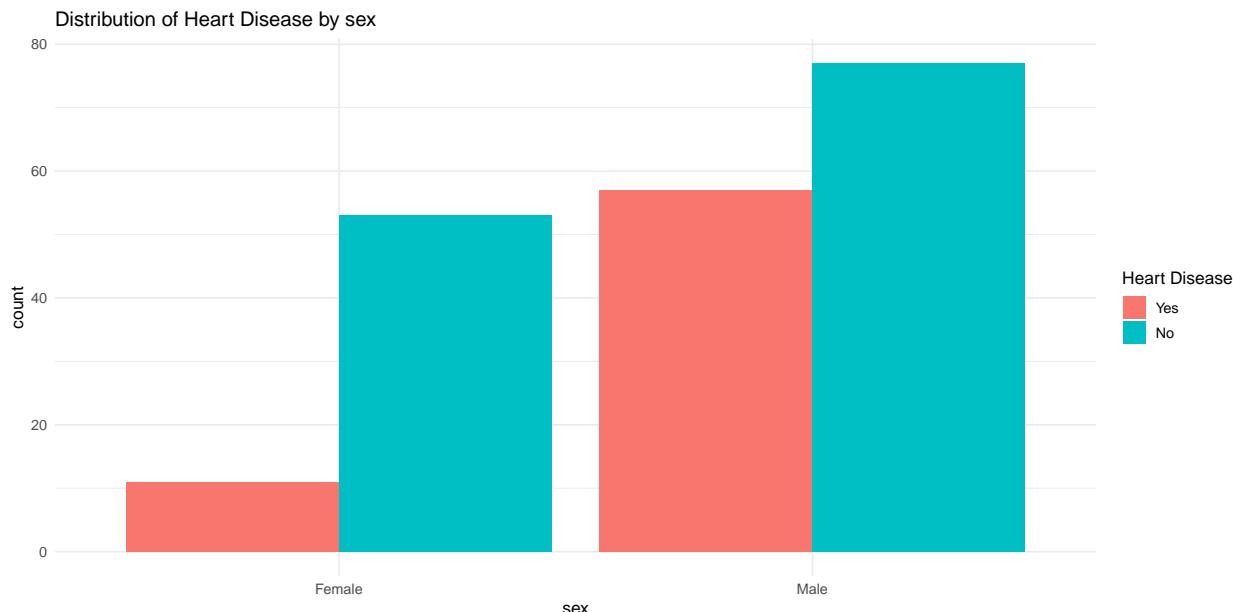
```
ggplot(Heart.df, aes(x=target, fill=target))+
  geom_bar() +
  ggtitle("Distribution of Heart Disease") +
  labs(x = "Heart Disease", fill = "Heart Disease")
```



The barplot of the target variable shows that the dataset contains more patients without heart disease than those diagnosed with the condition. While the groups are not perfectly balanced, the distribution is still sufficient to support reliable classification modeling. The presence of a substantial number of patients in both groups ensures that patterns distinguishing diseased from non-diseased individuals can be learned without the need for aggressive resampling techniques.

Visualize distribution of sex

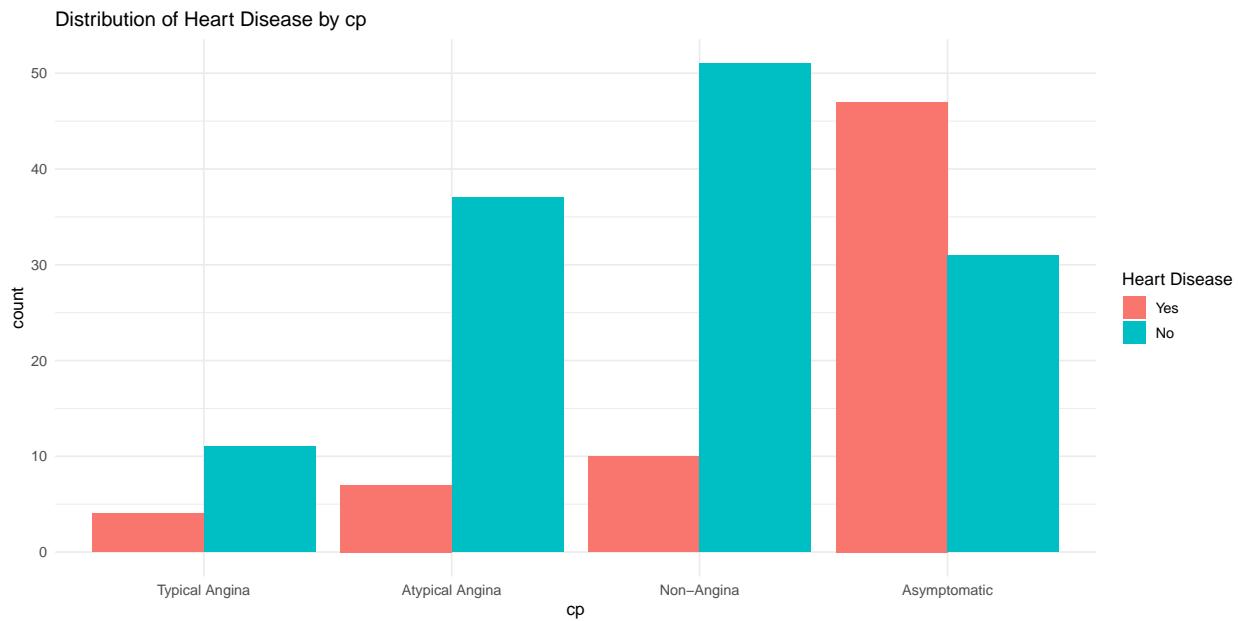
```
HeartDiseaseBar("sex")
```



The distribution of heart disease across sex reveals a noticeable difference between males and females. Although both groups include patients with and without heart disease, the proportion of heart disease cases is higher among males. This observation aligns with established clinical evidence showing that men are at a higher risk for developing cardiovascular conditions. The barplot therefore suggests that sex may hold moderate predictive value in distinguishing risk levels.

Visualize distribution of cp

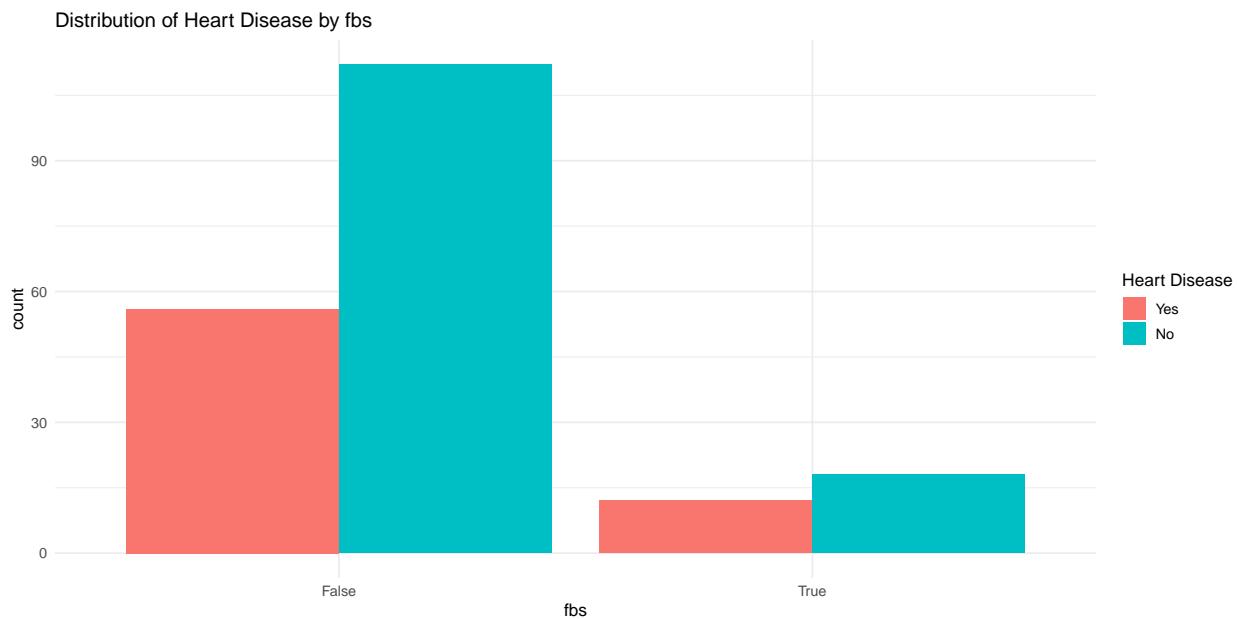
```
HeartDiseaseBar("cp")
```



Chest pain type stands out as one of the most informative categorical variables in the dataset. The barplot shows that patients with asymptomatic chest pain (classified as type 4) have a markedly higher prevalence of heart disease compared to the other chest pain categories. In contrast, typical and atypical angina types show fewer heart disease cases, and non-anginal discomfort falls somewhere in between. This strong separation highlights chest pain type as a critical clinical marker and a powerful predictor for modeling.

Visualize distribution of fbs

```
HeartDiseaseBar("fbs")
```

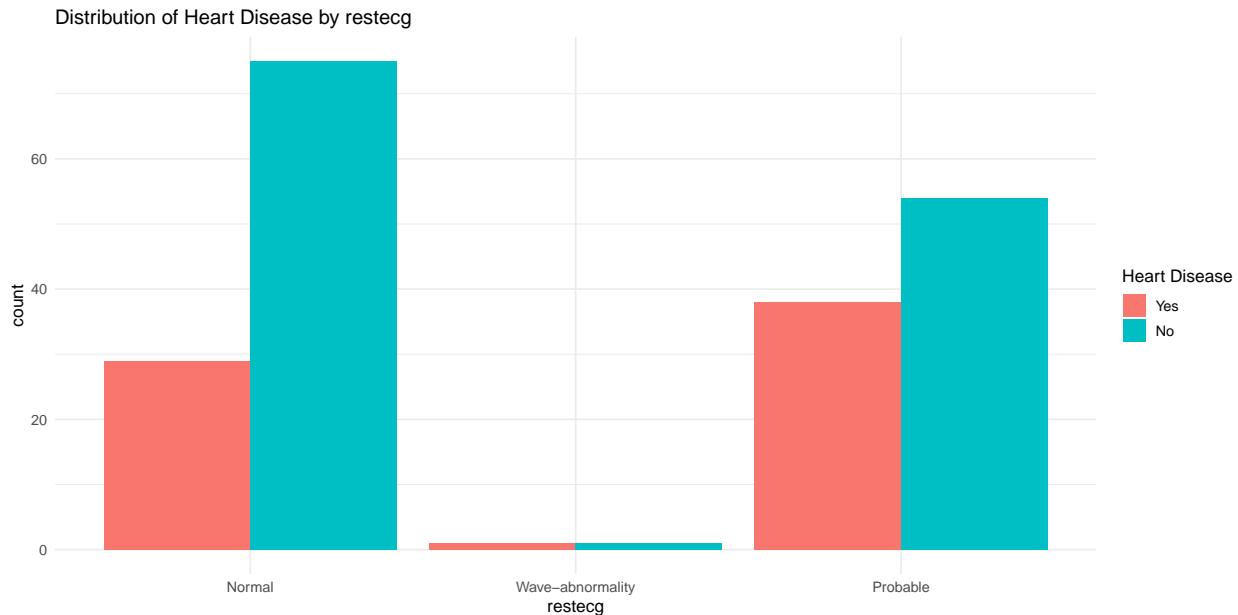


The comparison between patients with elevated fasting blood sugar and those within normal ranges reveals

minimal separation in heart disease outcomes. Both categories show similar proportions of “Yes” and “No” cases, suggesting that fasting blood sugar greater than 120 mg/dl does not substantially influence heart disease prevalence in this particular dataset. As a result, fbs appears to contribute only weak predictive value to the classification task.

Visualize distribution of restecg

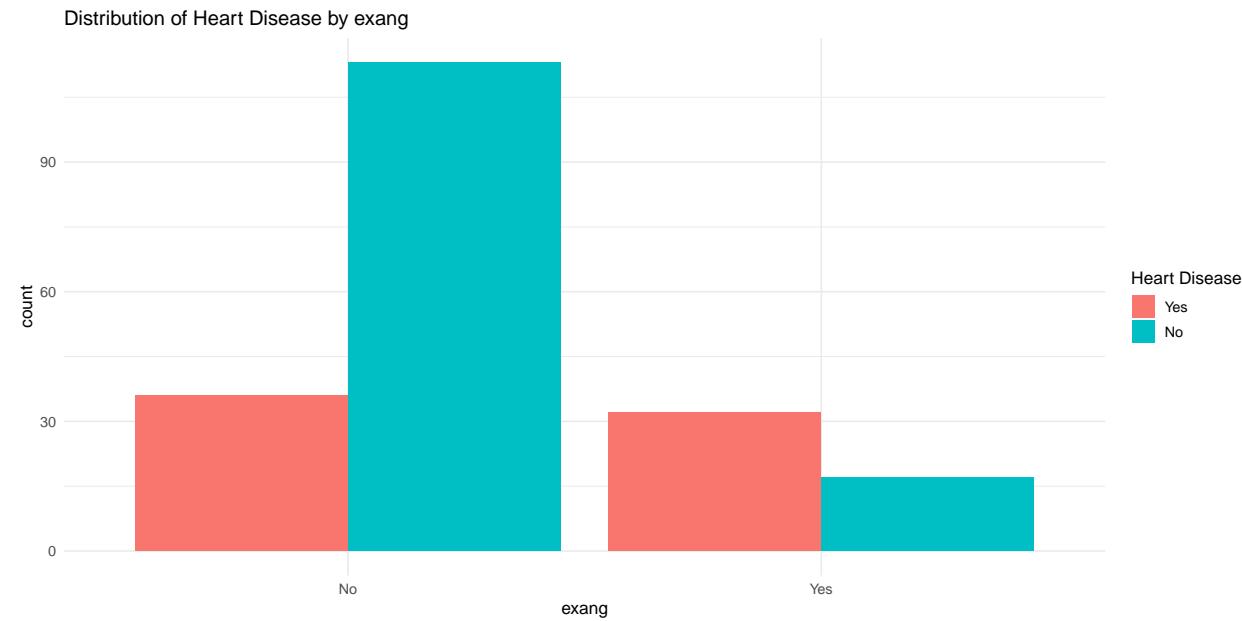
```
HeartDiseaseBar("restecg")
```



The barplot for resting ECG categories shows relatively similar distributions of heart disease across normal, wave-abnormality, and probable left ventricular hypertrophy groups. Although there are minor variations, these differences are not pronounced enough to indicate strong discriminative power. This suggests that resting ECG outcomes may provide some contextual clinical information but do not significantly differentiate between patients with and without heart disease in this dataset.

Visualize distribution of exang

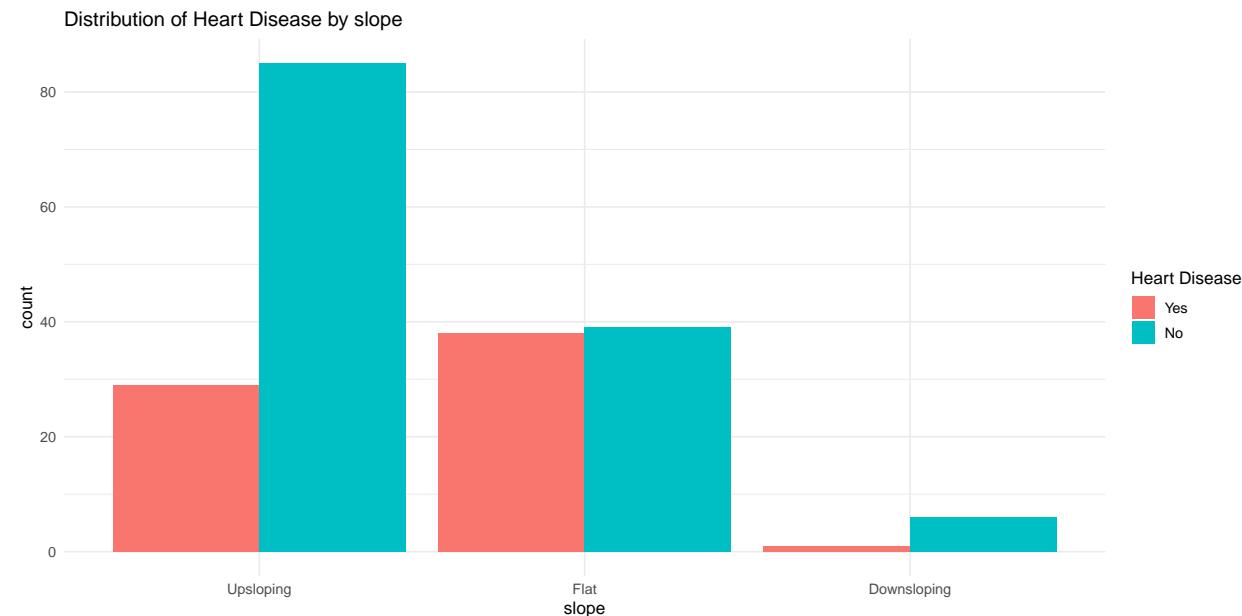
```
HeartDiseaseBar("exang")
```



Exercise-induced angina demonstrates a clear relationship with heart disease. The barplot shows that patients who experience angina during physical exertion are more likely to be diagnosed with heart disease compared to those who do not exhibit this symptom. This pattern is clinically meaningful, as exercise-induced chest pain often reflects underlying myocardial ischemia. Consequently, exang emerges as a strong and reliable predictor in identifying high-risk individuals.

Visualize distribution of slope

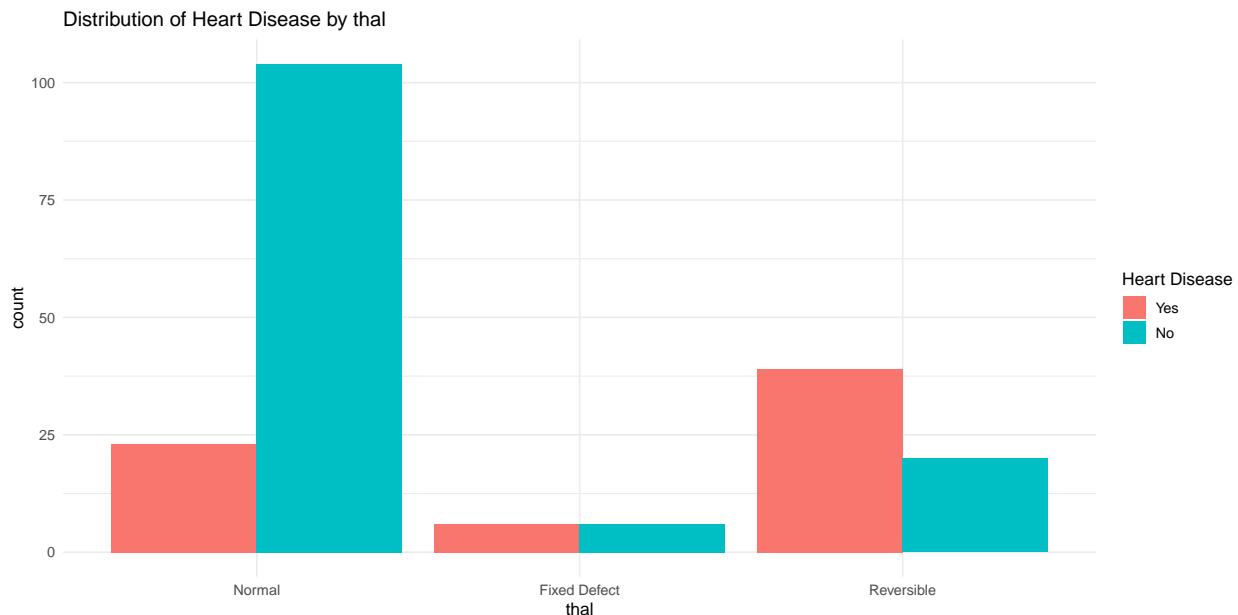
```
HeartDiseaseBar("slope")
```



The slope variable also shows substantial variation in heart disease prevalence across its categories. Patients with a flat slope ($\text{slope} = 2$) exhibit the highest concentration of heart disease cases, whereas the upsloping and downsloping categories show a predominance of non-diseased individuals. This distinct separation suggests that the ST segment response during exercise is a valuable clinical indicator and an important feature for predictive modeling.

Visualize distribution of thal

```
HeartDiseaseBar("thal")
```

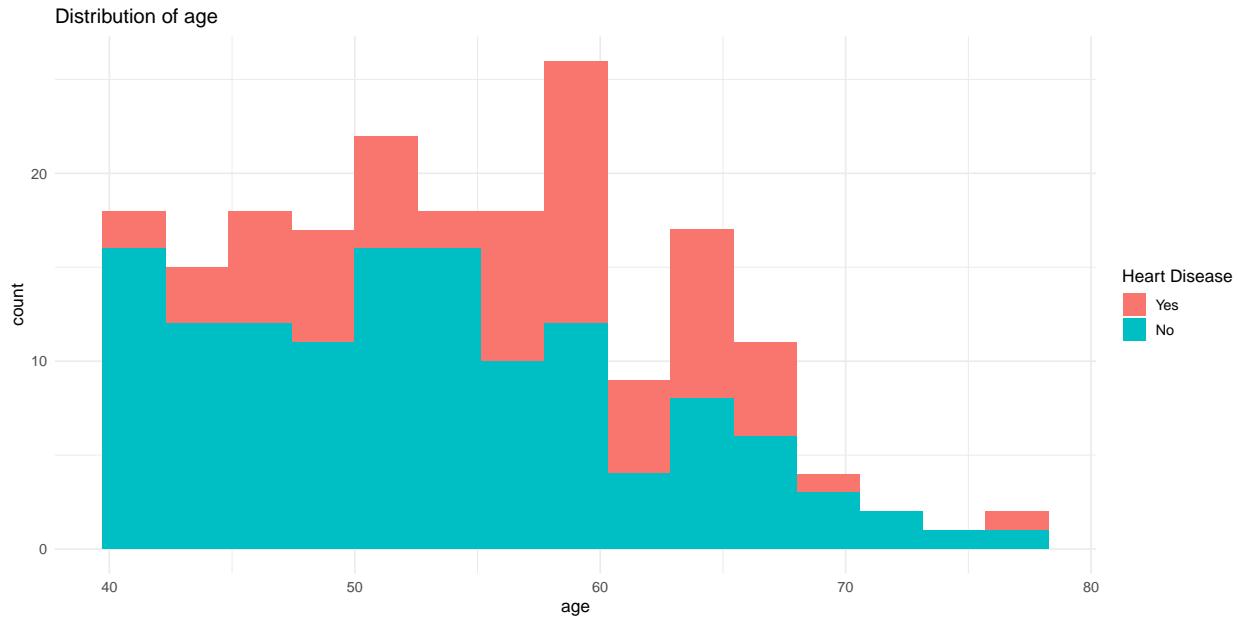


The thal variable is among the strongest categorical predictors of heart disease in the dataset. The barplot highlights that individuals with a reversible defect ($\text{thal} = 7$) have the highest count of heart disease cases, followed by those with fixed defects. Meanwhile, patients with normal thalassemia patterns show mostly non-diseased outcomes. This sharp contrast underscores the clinical relevance of thalassemia patterns and reinforces this variable's importance in risk prediction models.

Histograms for Numerical Variables

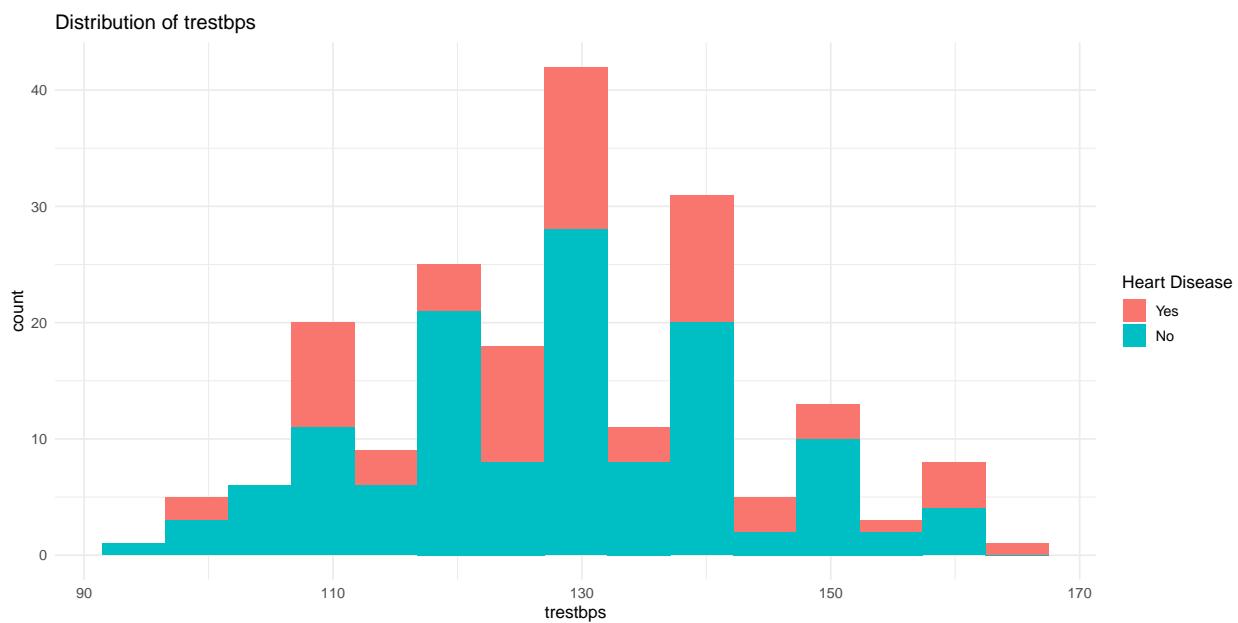
Histogram of age

```
HeartDiseaseHist("age")
```



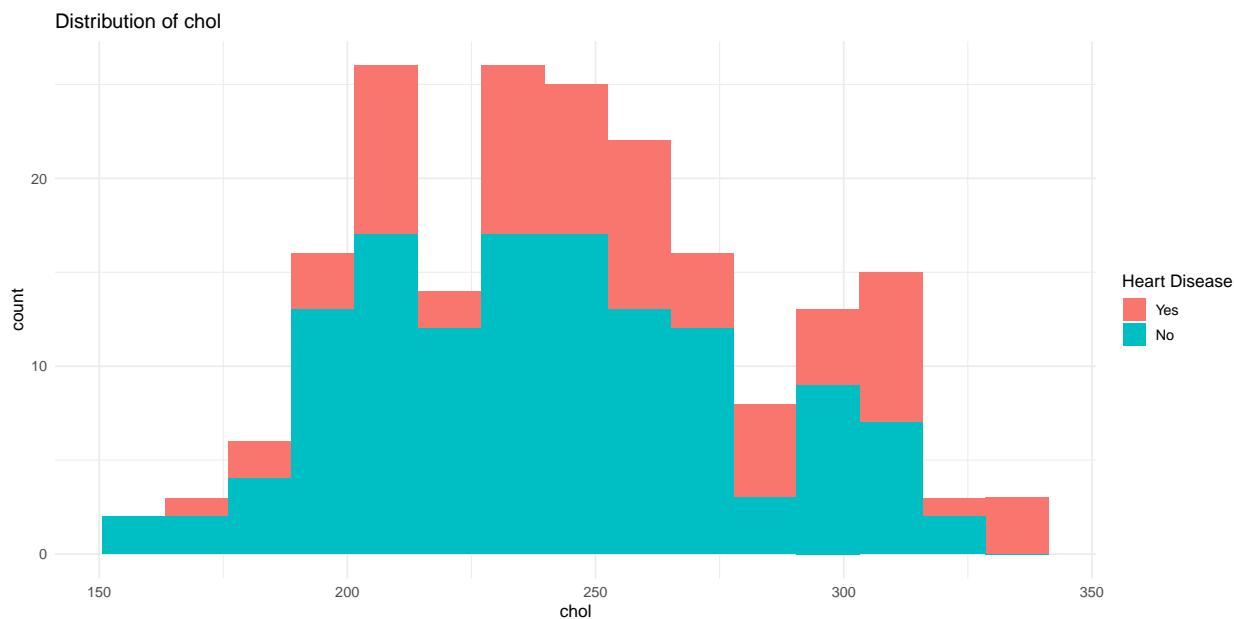
Histogram of trestbps

```
HeartDiseaseHist("trestbps")
```



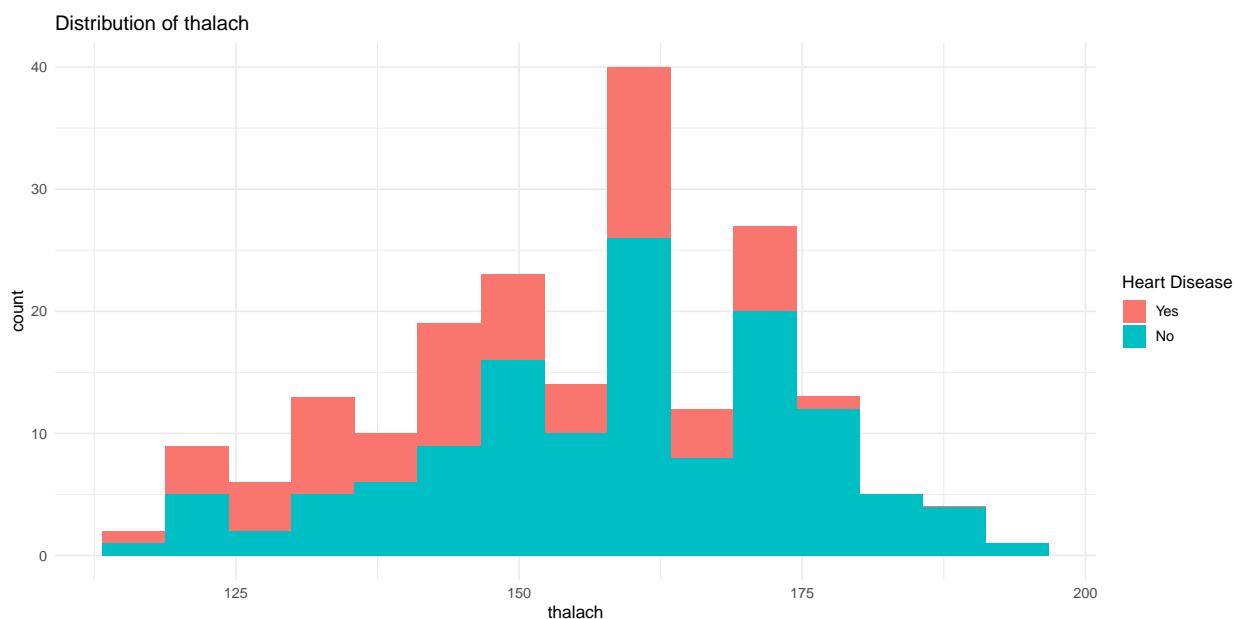
Histogram of chol

```
HeartDiseaseHist("chol")
```



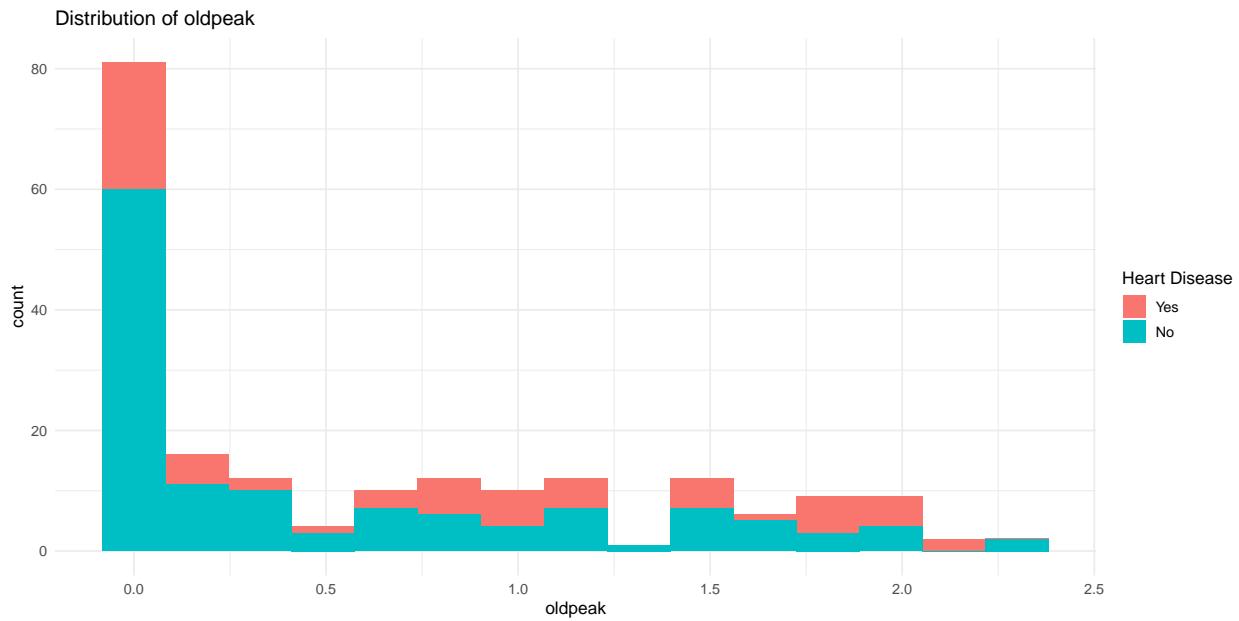
Histogram of thalach

```
HeartDiseaseHist("thalach")
```



Histogram of oldpeak

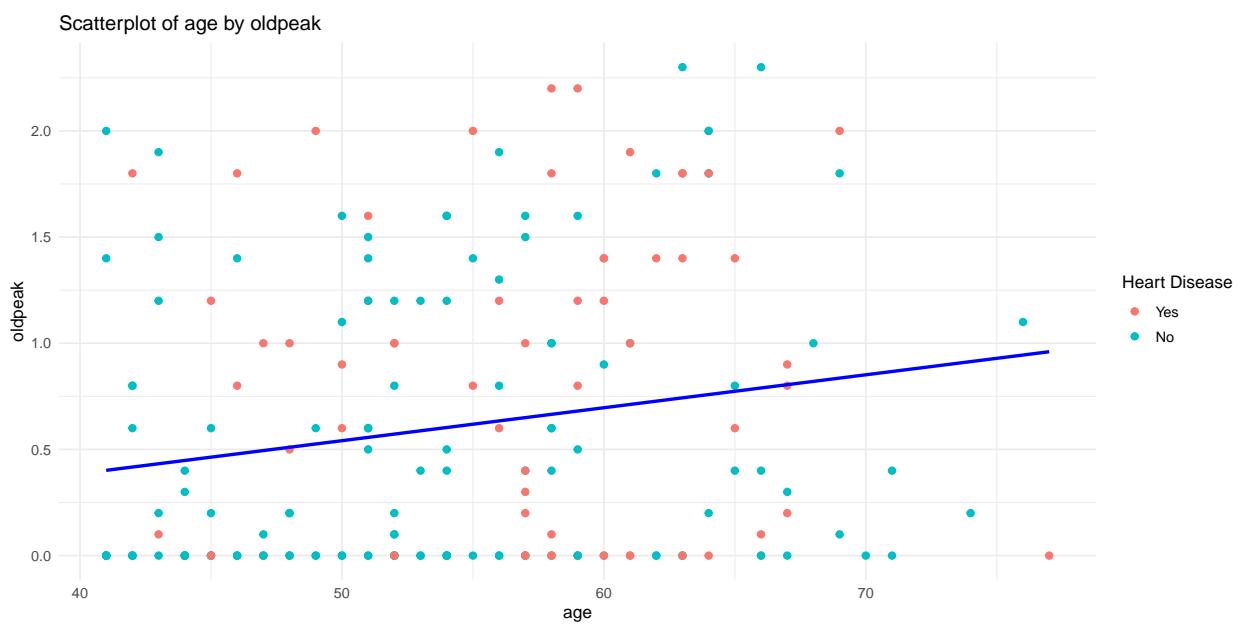
```
HeartDiseaseHist("oldpeak")
```



Scatterplots for numerical variables

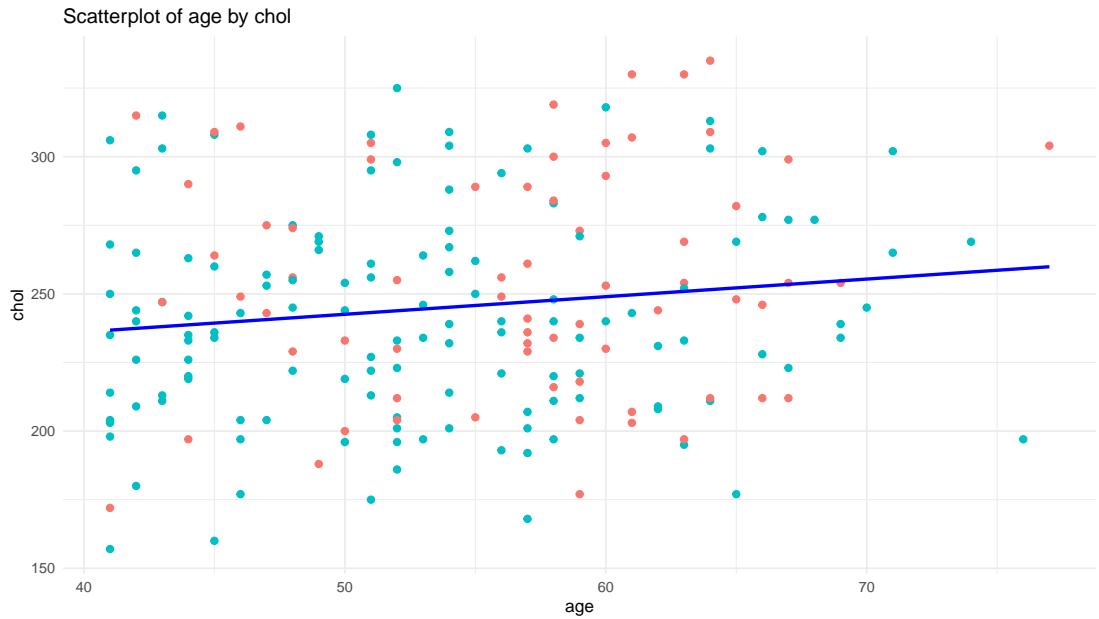
Scatterplot of age vs oldpeak

```
HeartDiseaseScatter("age", "oldpeak")
```



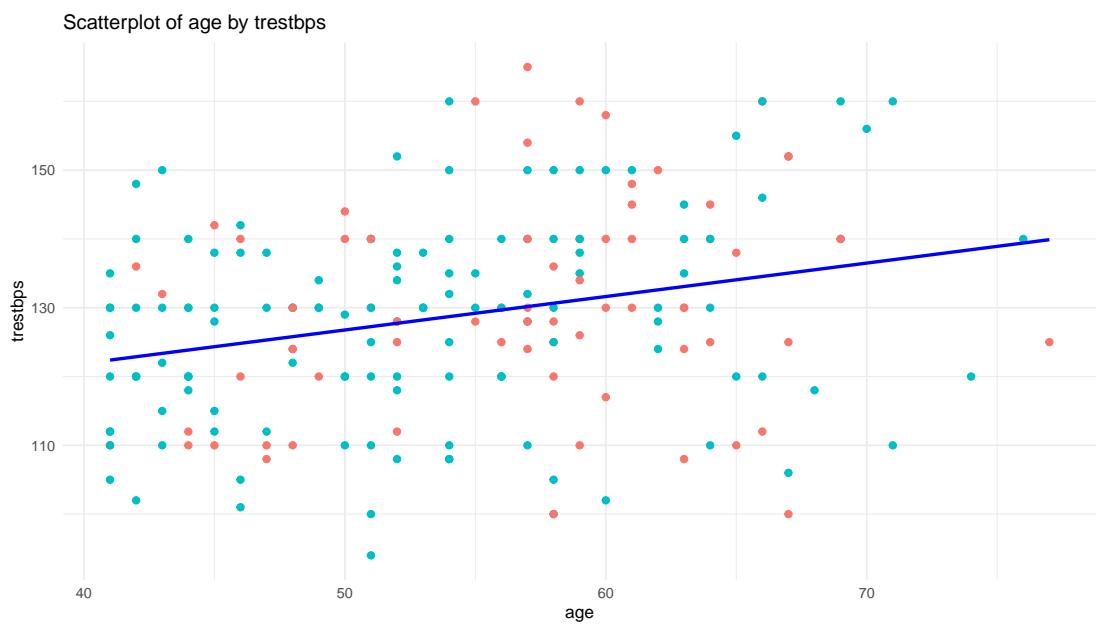
Scatterplot of age vs chol

```
HeartDiseaseScatter("age", "chol")
```



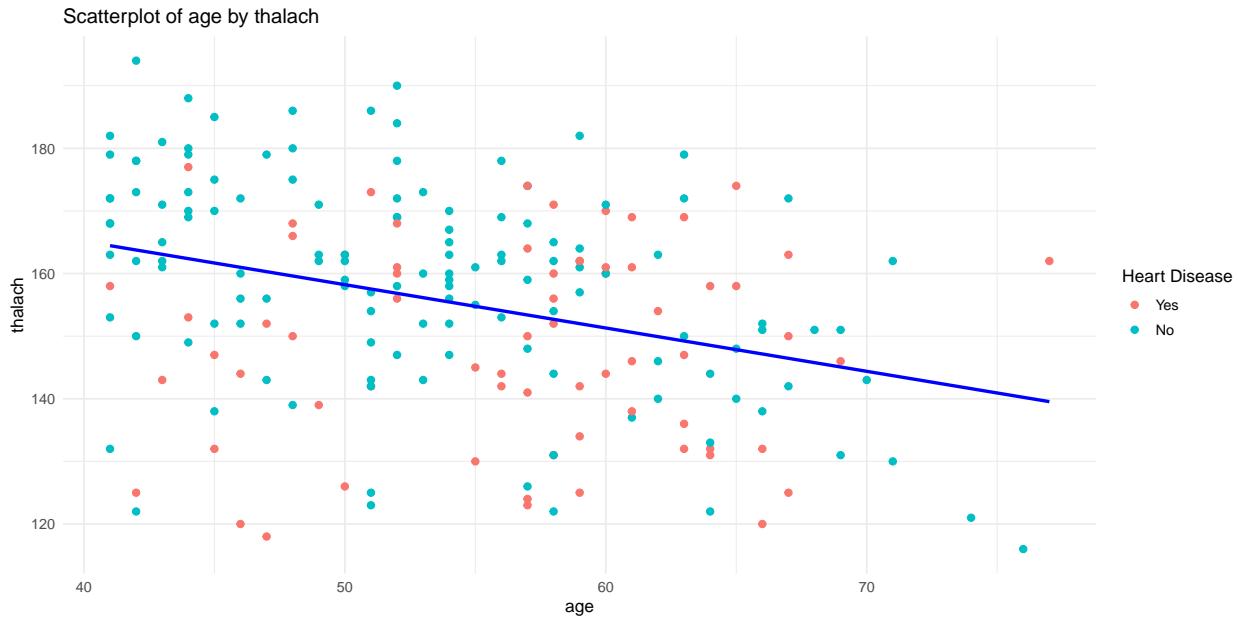
Scatterplot of age vs trestbps

```
HeartDiseaseScatter("age", "trestbps")
```



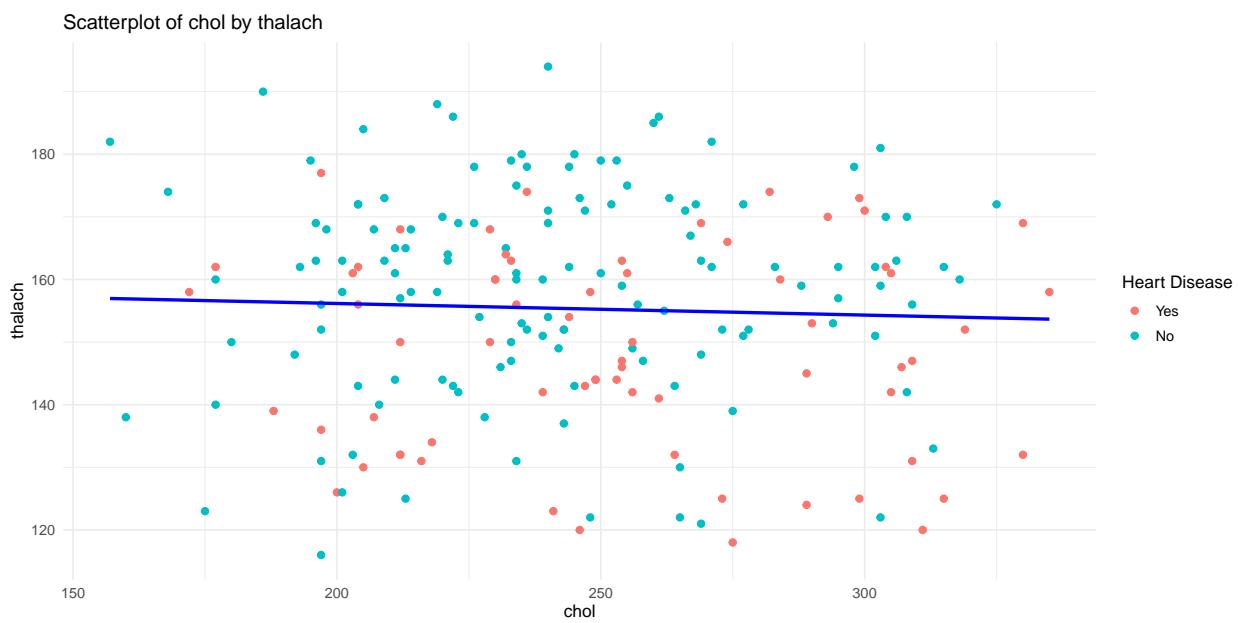
Scatterplot of age vs thalach

```
HeartDiseaseScatter("age", "thalach")
```



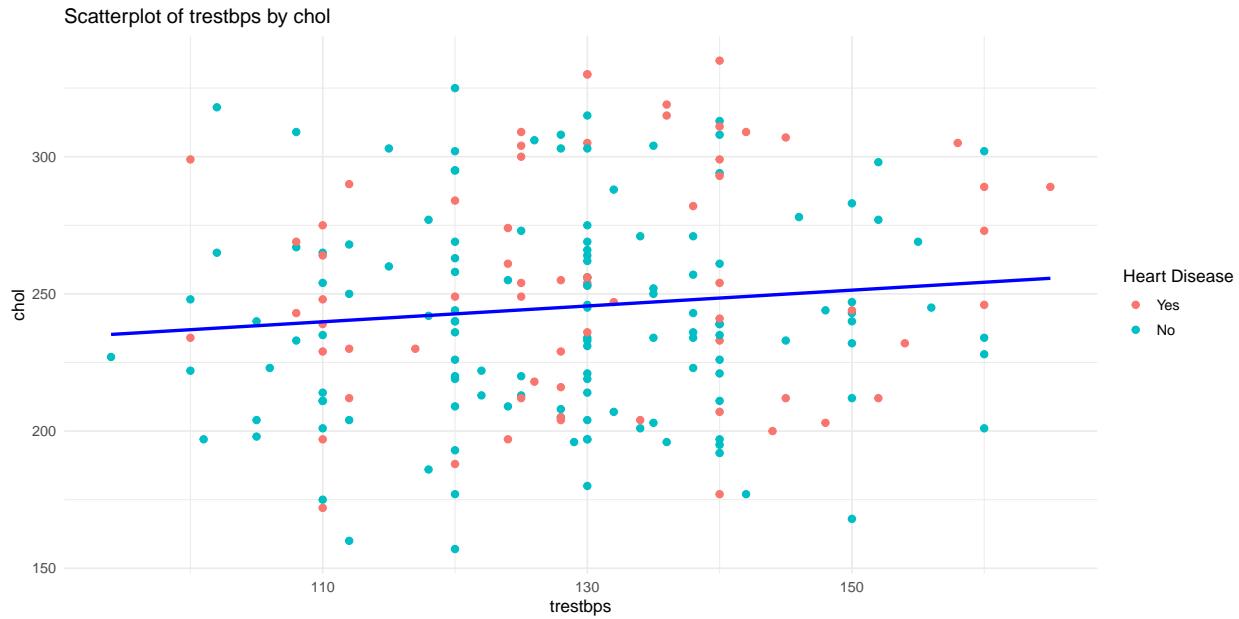
Scatterplot of chol vs thalach

```
HeartDiseaseScatter("chol", "thalach")
```



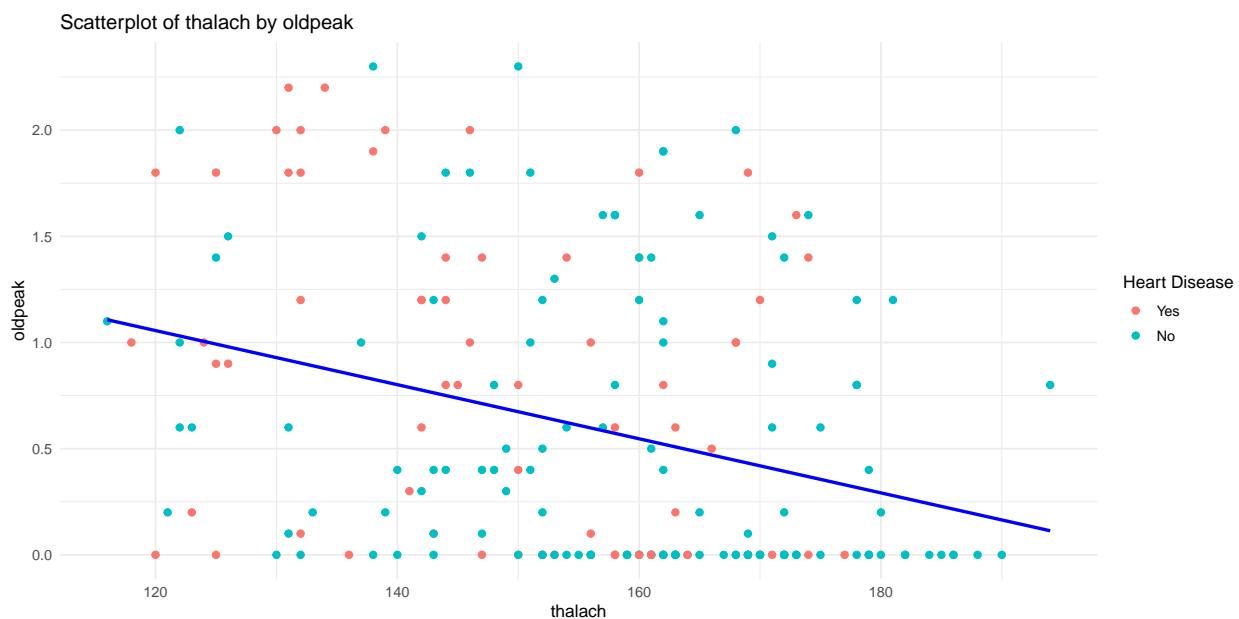
Scatterplot of trestbps vs chol

```
HeartDiseaseScatter("trestbps", "chol")
```



Scatterplot of thalach vs oldpeak

```
HeartDiseaseScatter("thalach", "oldpeak")
```



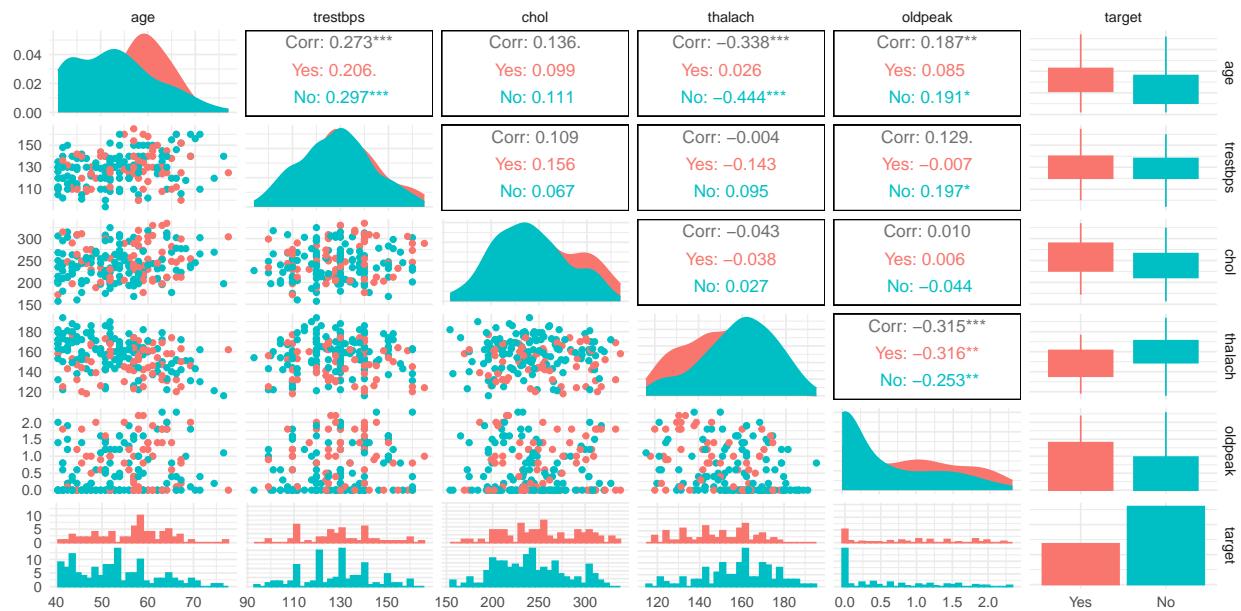
Pairwise correlation plots

Pairwise correlation plot for numerical variables

```
ggpairs(Heart.df[, c("age", "trestbps", "chol", "thalach", "oldpeak", "target")],  
       aes(color = target, fill = target))
```

```
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```

```
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value `binwidth`.
```



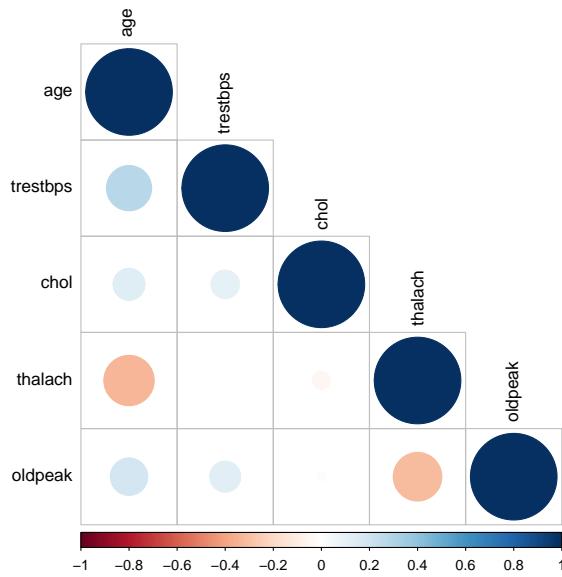
Correlation matrix

Correlation matrix for numerical variables

```
# Selecting only continuous variables
continuous_vars <- c("age", "trestbps", "chol", "thalach", "oldpeak")
continuous_data <- Heart.df %>% select(all_of(continuous_vars))

# Calculating correlation matrix
correlation_matrix <- cor(continuous_data)

# Plotting the correlation matrix
corrplot(correlation_matrix, method = "circle",
          type = "lower", tl.col = "black")
```



Splitting the dataset

Split the dataset into training set and testing set. I will use 70% of the data for training and 30% for testing.

```
# Set random seed for reproducibility
set.seed(123)

# Split dataset into training (70%) and testing (30%)
indx <- createDataPartition(Heart.df$target, p = 0.7, list = FALSE)
train_data <- Heart.df[indx, ]
test_data <- Heart.df[-indx, ]
```

Scale numeric variables

For models like SVM and KNN, we need to scale the numeric features. Scaling numerical features (excluding the target and categorical variables)

```
numeric_columns <- c("age", "trestbps", "chol", "thalach", "oldpeak")
train_data[numeric_columns] <- scale(train_data[numeric_columns])
test_data[numeric_columns] <- scale(test_data[numeric_columns])
```