Heart Disease Prediction

Peter Thai

25 April 2025

Introduction

This project applies supervised machine learning techniques to predict the presence of heart disease using the UCI Heart Disease dataset. We will explore the data, train models, and evaluate performance based on accuracy and AUC metrics.

Load Libraries and Data

```r
if (!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
if (!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if (!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: randomForest
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:dplyr':
##
##     combine
##
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
if (!require(pROC)) install.packages("pROC", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: pROC
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(tidyverse)
library(caret)
library(randomForest)
library(pROC)

heart <- read.csv("heart.csv")
heart$target <- as.factor(heart$target)
```

Exploratory Data Analysis

```
summary(heart)
```

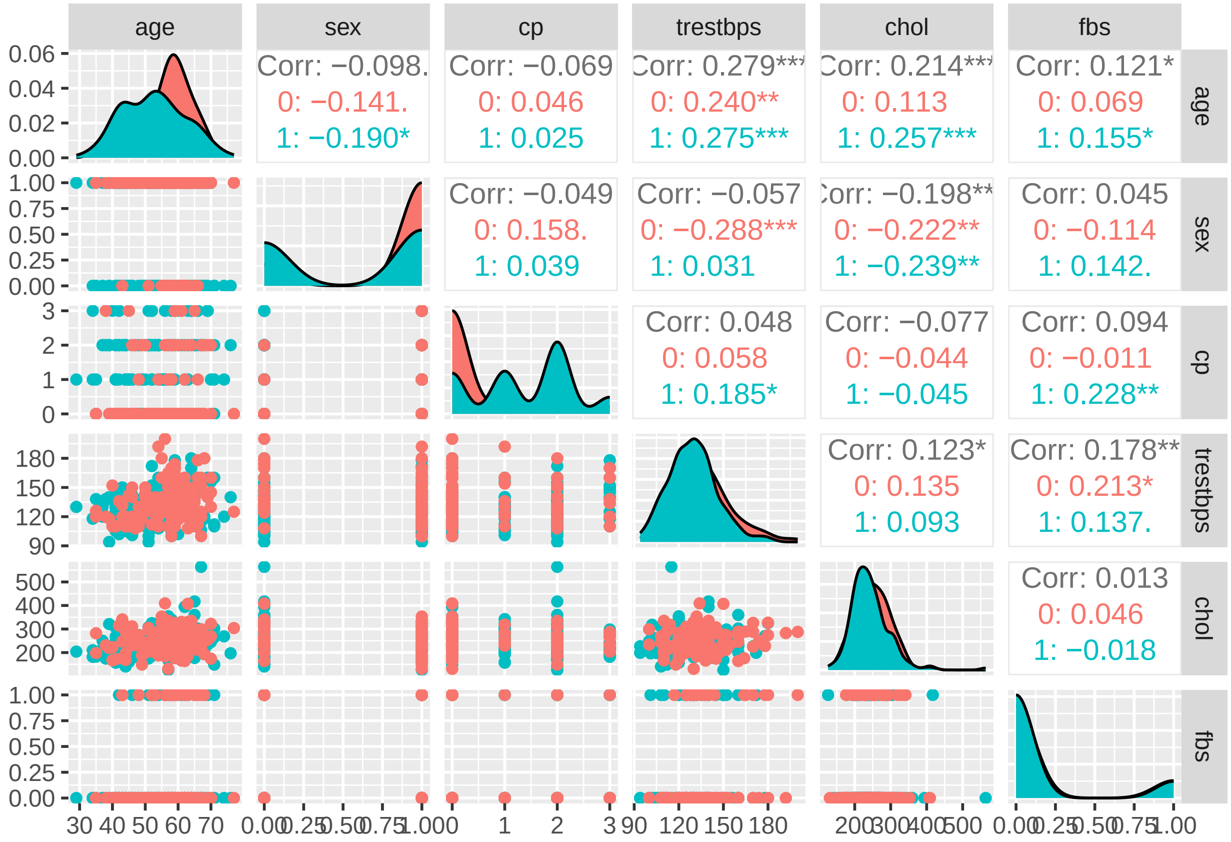
```
##       age             sex               cp           trestbps
##  Min.   :29.00   Min.   :0.0000   Min.   :0.000   Min.   : 94.0
##  1st Qu.:47.50   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:120.0
##  Median :55.00   Median :1.0000   Median :1.000   Median :130.0
##  Mean   :54.37   Mean   :0.6832   Mean   :0.967   Mean   :131.6
##  3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.000   3rd Qu.:140.0
##  Max.   :77.00   Max.   :1.0000   Max.   :3.000   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 :240.0   Median :0.0000   Median :1.0000   Median :153.0
##  Mean   :246.3   Mean   :0.1485   Mean   :0.5281   Mean   :149.6
##  3rd Qu.:274.5   3rd Qu.:0.0000   3rd Qu.:1.0000   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.   :0.000   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 :1.000   Median :0.0000
##  Mean   :0.3267   Mean   :1.04   Mean   :1.399   Mean   :0.7294
##  3rd Qu.:1.0000   3rd Qu.:1.60   3rd Qu.:2.000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :6.20   Max.   :2.000   Max.   :4.0000
##       thal       target
##  Min.   :0.000   0:138
##  1st Qu.:2.000   1:165
##  Median :2.000
##  Mean   :2.314
##  3rd Qu.:3.000
##  Max.   :3.000
```

```
str(heart)
```

```
## ’data.frame’:    303 obs. of  14 variables:
##  $ age     : int  63 37 41 56 57 57 56 44 52 57 ...
##  $ sex     : int  1 1 0 1 0 1 0 1 1 1 ...
##  $ cp      : int  3 2 1 1 0 0 1 1 2 2 ...
##  $ trestbps: int  145 130 130 120 120 140 140 120 172 150 ...
##  $ chol    : int  233 250 204 236 354 192 294 263 199 168 ...
##  $ fbs     : int  1 0 0 0 0 0 0 0 1 0 ...
##  $ restecg : int  0 1 0 1 1 1 0 1 1 1 ...
##  $ thalach : int  150 187 172 178 163 148 153 173 162 174 ...
##  $ exang   : int  0 0 0 0 1 0 0 0 0 0 ...
##  $ oldpeak : num  2.3 3.5 1.4 0.8 0.6 0.4 1.3 0 0.5 1.6 ...
##  $ slope   : int  0 0 2 2 2 1 1 2 2 2 ...
##  $ ca      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ thal    : int  1 2 2 2 2 1 2 3 3 2 ...
##  $ target  : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

```
heart %>% GGally::ggpairs(columns = 1:6, aes(color = target))
```

```
## Registered S3 method overwritten by ’GGally’:
##   method from
##   +.gg   ggplot2
```



Data Partitioning

```
set.seed(123)
train_index <- createDataPartition(heart$target, p = 0.8, list = FALSE)
train_data <- heart[train_index, ]
test_data <- heart[-train_index, ]
```

Logistic Regression Model

```
log_model <- glm(target ~ ., data = train_data, family = "binomial")
summary(log_model)
```

```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = train_data)
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.654796   2.801571   0.234 0.815199
## age            0.016291   0.026641   0.612 0.540861
## sex         -1.973555   0.546594  -3.611 0.000305 ***
## cp           0.796468   0.204425   3.896 9.77e-05 ***
## trestbps    -0.013829   0.011577  -1.195 0.232282
## chol        -0.004440   0.004294  -1.034 0.301066
## fbs          0.282036   0.612297   0.461 0.645071
## restecg      0.895128   0.408819   2.190 0.028557 *
## thalach      0.035307   0.011729   3.010 0.002609 **
## exang       -0.968394   0.466916  -2.074 0.038077 *
## oldpeak     -0.638153   0.247076  -2.583 0.009800 **
## slope        0.305689   0.415728   0.735 0.462151
## ca          -0.765629   0.211001  -3.629 0.000285 ***
## thal        -1.129838   0.340057  -3.322 0.000892 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 335.05  on 242  degrees of freedom
## Residual deviance: 161.07  on 229  degrees of freedom
## AIC: 189.07
##
## Number of Fisher Scoring iterations: 6
```

```
log_probs <- predict(log_model, test_data, type = "response")
log_preds <- ifelse(log_probs > 0.5, 1, 0) %>% as.factor()
confusionMatrix(log_preds, test_data$target)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 19  3
##          1  8 30
##
##                Accuracy : 0.8167
##                  95% CI : (0.6956, 0.9048)
##     No Information Rate : 0.55
##     P-Value [Acc > NIR] : 1.344e-05
##
##                   Kappa : 0.6233
##
##  Mcnemar's Test P-Value : 0.2278
##
##             Sensitivity : 0.7037
##             Specificity : 0.9091
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.7895
##              Prevalence : 0.4500
##          Detection Rate : 0.3167
##    Detection Prevalence : 0.3667
##       Balanced Accuracy : 0.8064
##
##        'Positive' Class : 0
##
```

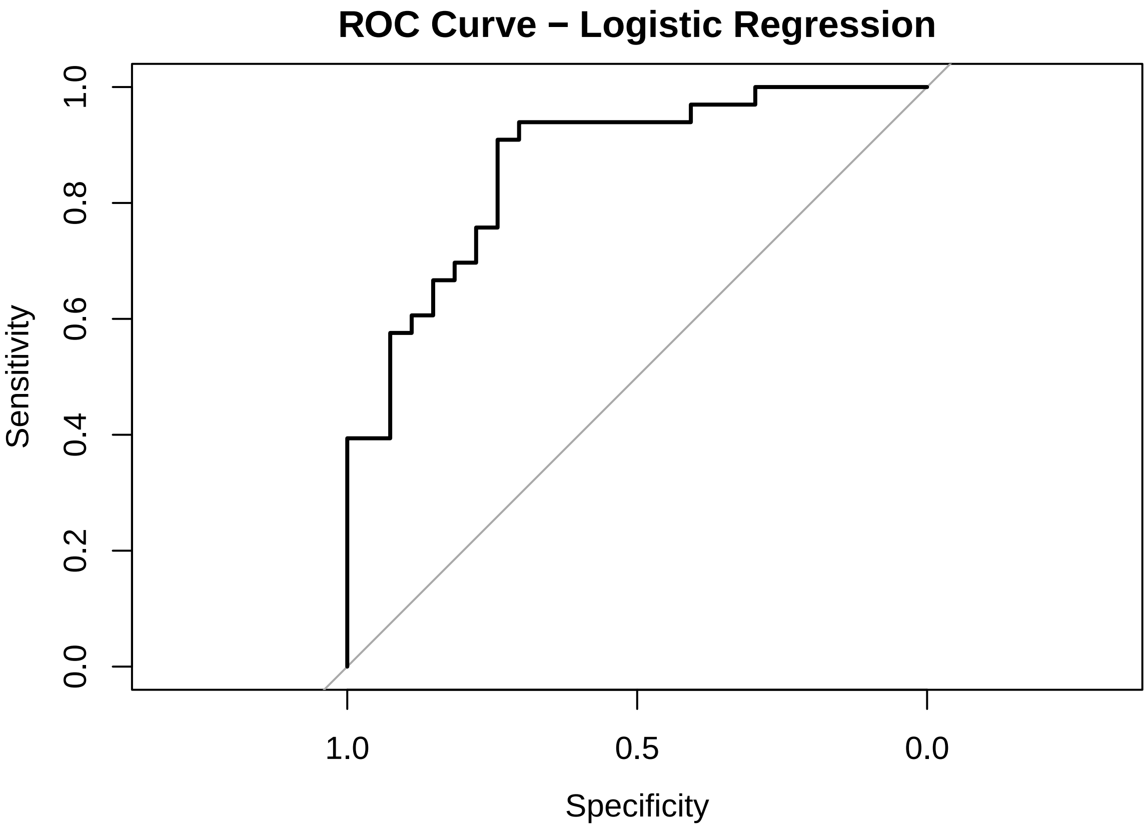
ROC Curve - Logistic Regression

```
log_roc <- roc(test_data$target, log_probs)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(log_roc, main = "ROC Curve - Logistic Regression")
```



```
auc(log_roc)
```

```
## Area under the curve: 0.8676
```

Random Forest Model

```
set.seed(123)
rf_model <- randomForest(target ~ ., data = train_data, ntree = 500, mtry = 4, importance = TRUE)
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = target ~ ., data = train_data, ntree = 500,      mtry = 4, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 19.34%
## Confusion matrix:
##    0   1 class.error
## 0 84  27   0.2432432
## 1 20 112   0.1515152
```

```
rf_preds <- predict(rf_model, test_data)
confusionMatrix(rf_preds, test_data$target)
```

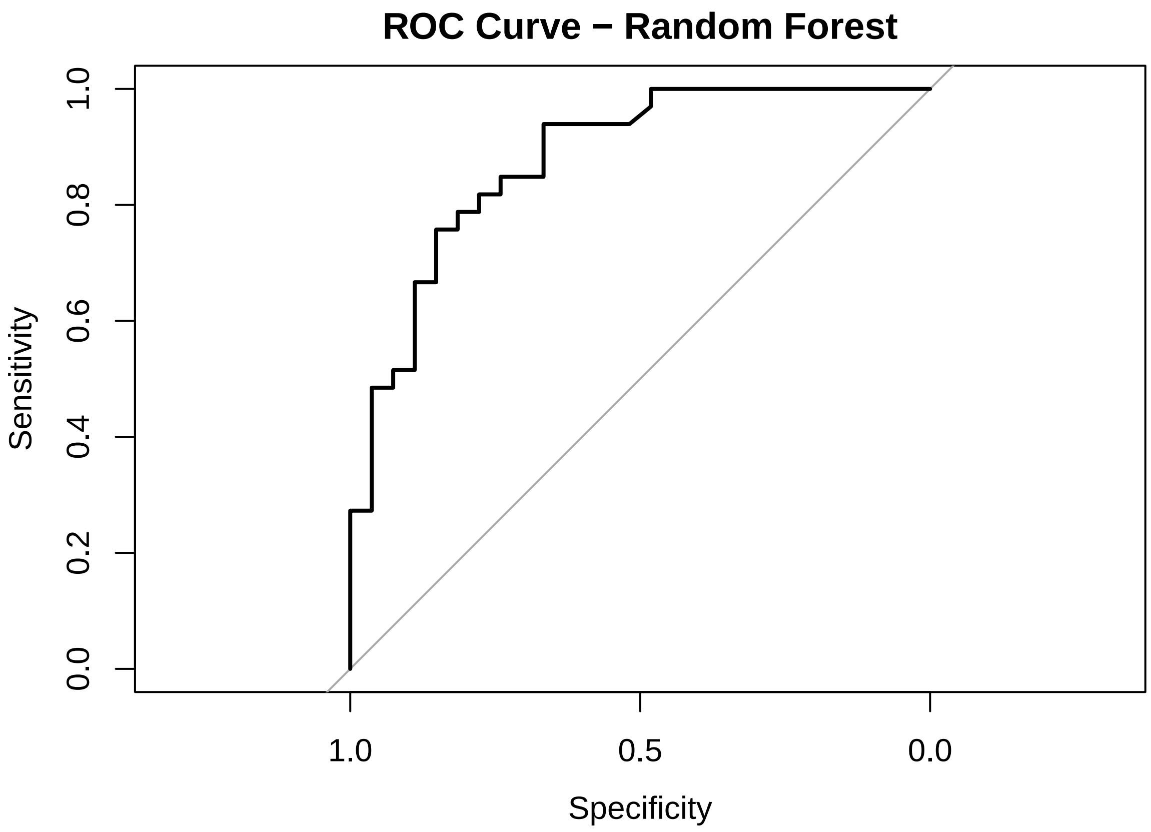
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 20  5
##          1  7 28
##
##                Accuracy : 0.8
##                  95% CI : (0.6767, 0.8922)
##     No Information Rate : 0.55
##     P-Value [Acc > NIR] : 4.67e-05
##
##                   Kappa : 0.5932
##
##  Mcnemar’s Test P-Value : 0.7728
##
##             Sensitivity : 0.7407
##             Specificity : 0.8485
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.8000
##              Prevalence : 0.4500
##          Detection Rate : 0.3333
##    Detection Prevalence : 0.4167
##       Balanced Accuracy : 0.7946
##
##        ’Positive’ Class : 0
##
```

ROC Curve - Random Forest

```
rf_probs <- predict(rf_model, test_data, type = "prob")[,2]
rf_roc <- roc(test_data$target, rf_probs)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls < cases
```

```
plot(rf_roc, main = "ROC Curve - Random Forest")
```

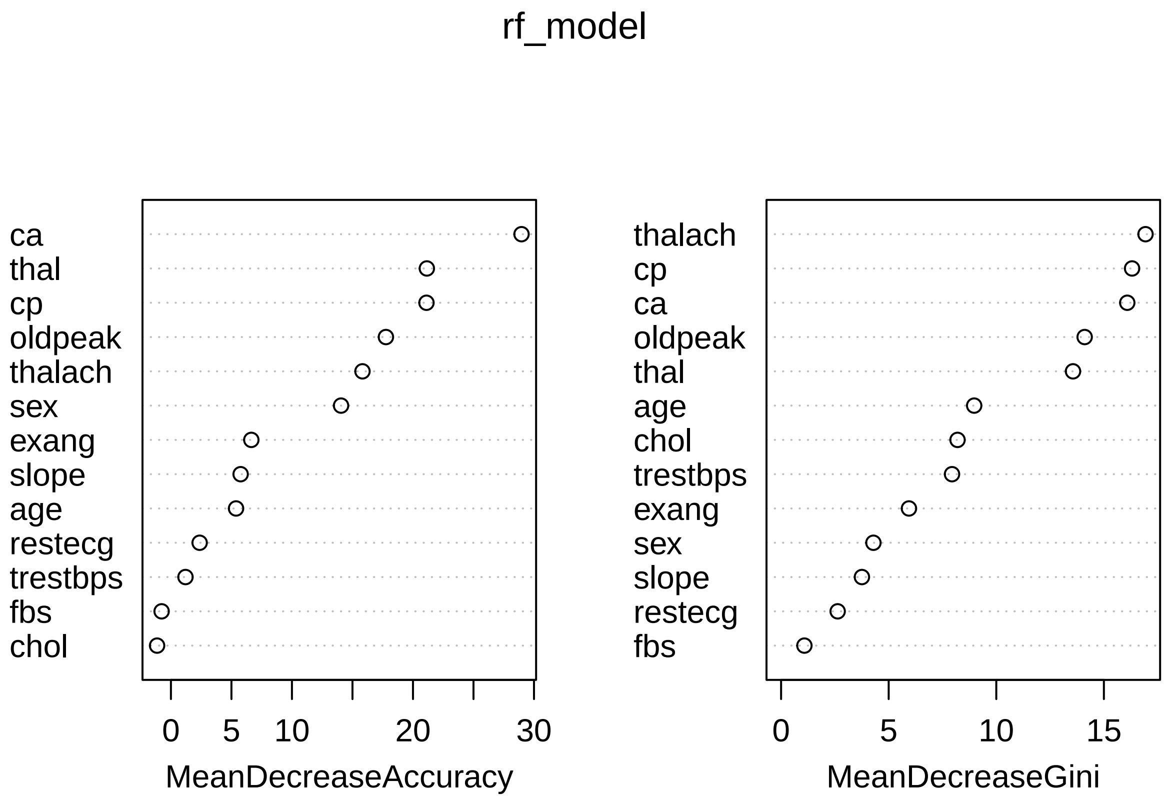


```
auc(rf_roc)
```

```
## Area under the curve: 0.8782
```

Feature Importance

```
varImpPlot(rf_model)
```



Conclusion

Both models performed well, with Random Forest showing slightly higher predictive power based on AUC. This demonstrates the value of ensemble methods for structured healthcare data.

References

- UCI Machine Learning Repository: Heart Disease Dataset
- caret, randomForest, and pROC R packages