UCI Heart Disease Prediction

Lucas Mendicino

11/21/2021

Load Packages

```
#EDA
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.1.0     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(DataExplorer)
library(gtsummary)
library(ggsci)
theme_gtsummary_journal(journal = "jama")
```

```
## Setting theme `JAMA`
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
# Penalized Regression
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-3
```

```
# Random Forest
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## 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
```

```
# Support Vector Machine
library(e1071)
# PCA
library(PCAmixdata)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
library(kernlab)
```

```
##
## Attaching package: 'kernlab'
```

```
## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

```
library(mltools)
```

```
##
## Attaching package: 'mltools'
```

```
## The following object is masked from 'package:e1071':
##
##     skewness
```

```
## The following object is masked from 'package:tidyr':
##
##     replace_na
```

```
library(varhandle)
library(data.table)
```

```
##
## Attaching package: 'data.table'
```

```
## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

In this project, I will be using the Heart Disease dataset from the UC Irvine Machine Learning Repository. The dataset contains 14 features.

Outline

1. Prepare the Data
2. Exploratory Data Analysis
3. Modeling

- Clustering (Principle Component Analysis, Hierarchical Clustering Analysis, K-Means)
- Logistic Regression
- Random Forest
- SVM
-

Prepare the Data

Let's load the data

```
# Read datasets Cleveland_hd.csv into hd_data
hd <- read.csv("~/Desktop/UCI-Heart-Disease-Prediction/heart.csv")
```

Let's look at the structure and a few rows

```
str(hd)
```

```
## '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  : int  1 1 1 1 1 1 1 1 1 1 ...
```

```
head(hd)
```

```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal
## 1  63   1  3      145  233   1       0     150     0     2.3     0  0    1
## 2  37   1  2      130  250   0       1     187     0     3.5     0  0    2
## 3  41   0  1      130  204   0       0     172     0     1.4     2  0    2
## 4  56   1  1      120  236   0       1     178     0     0.8     2  0    2
## 5  57   0  0      120  354   0       1     163     1     0.6     2  0    2
## 6  57   1  0      140  192   0       1     148     0     0.4     1  0    1
##   target
```

```
## 1       1
## 2       1
## 3       1
## 4       1
## 5       1
## 6       1
```

The variables include:

- age: age in years
- sex: (1 = male; 0 = female)
- cp: chest pain type (typical angina, atypical angina, non-anginal pain, or asymptomatic angina)
- trestbps: resting blood pressure (in mm Hg on admission to the hospital)
- chol: serum cholestoral in mg/dl
- fbs: Fasting blood sugar (< 120 mg/dl or > 120 mg/dl) (1 = true; 0 = false)
- restecg: resting electrocardiographic results (normal, ST-T wave abnormality, or left ventricular hypertrophy)
- thalach: Max. heart rate achieved during thalium stress test
- exang: Exercise induced angina (1 = yes; 0 = no)
- oldpeak: ST depression induced by exercise relative to rest
- slope: Slope of peak exercise ST segment (0 = upsloping, 1 = flat, or 2 = downsloping)
- ca: number of major vessels (0-3) colored by flourosopy 4 = NA
- thal: Thalium stress test result 3 = normal; 6 = fixed defect; 7 = reversable defect 0 = NA
- target: Heart disease status 1 or 0 (0 = heart disease 1 = asymptomatic)

It is important to note that for the target variable, 0 is assigned to heart disease and 1 is asymptomatic. Let's recode this.

```
hd_new <- hd %>% mutate(target = recode(target, "0" = "1", "1" = "0"))
```

Let's tidy the dataset a bit more.

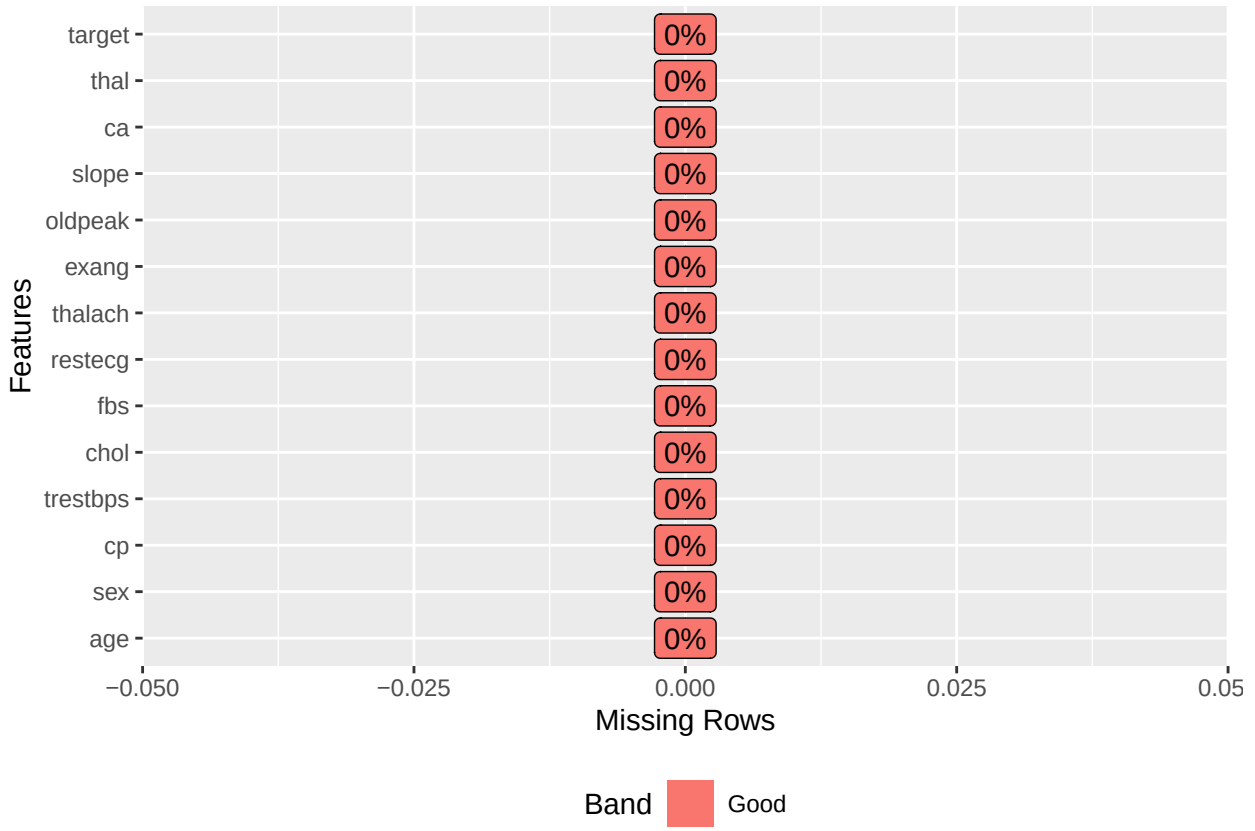
```
hd2 <- hd_new %>%
  filter(
    thal != 0 & ca != 4 # remove values correspondind to NA in original dataset
  ) %>%
# Recode the categorical variables as factors using the dplyr library.
  mutate(
    sex = case_when(
      sex == 0 ~ "female",
      sex == 1 ~ "male"
           ),
    fbs = case_when(
      fbs == 0 ~ "<=120",
      fbs == 1 ~ ">120"
            ),
    exang = case_when(
      exang == 0 ~ "no",
      exang == 1 ~ "yes"
            ),
    cp = case_when(
      cp == 3 ~ "typical angina",
      cp == 1 ~ "atypical angina",
      cp == 2 ~ "non-anginal pain",
      cp == 0 ~ "asymptomatic angina"
          ),
    restecg = case_when(
```

```
      restecg == 0 ~ "hypertrophy",
      restecg == 1 ~ "normal",
      restecg == 2 ~ "wave abnormality"
              ),
    target = case_when(
      target == 0 ~ "asymptomatic",
      target == 1 ~ "heart-disease"
              ),
    slope = case_when(
      slope == 2 ~ "upsloping",
      slope == 1 ~ "flat",
      slope == 0 ~ "downsloping"
    ),
    thal = case_when(
      thal == 1 ~ "fixed defect",
      thal == 2 ~ "normal",
      thal == 3 ~ "reversable defect"
    ),
    sex = as.factor(sex),
    fbs = as.factor(fbs),
    exang = as.factor(exang),
    cp = as.factor(cp),
    slope = as.factor(slope),
    ca = as.factor(ca),
    thal = as.factor(thal),
    restecg = as.factor(restecg),
    target = as.factor(target)
  )

hd_EDA <- hd2
hd_model <- hd2
```

Exploratory Data Analysis

Let's check out missing values

```
plot_missing(hd_EDA)
```



The dataset is complete with no missing values.

Let's get summary statistics:

```
hd_EDA %>% tbl_summary(statistic = list(all_continuous() ~ "{mean} ({sd})",
                       all_categorical() ~ "{n} ({p}%)"),
                     digits = all_continuous() ~ 2)
```

Characteristic	**N = 296**
age, Mean (SD)	54.52 (9.06)
sex, n (%)	
female	95 (32%)
male	201 (68%)
cp, n (%)	
asymptomatic angina	141 (48%)
atypical angina	49 (17%)
non-anginal pain	83 (28%)
typical angina	23 (7.8%)
trestbps, Mean (SD)	131.60 (17.73)
chol, Mean (SD)	247.16 (51.98)
fbs, n (%)	
<=120	253 (85%)
>120	43 (15%)
restecg, n (%)	
hypertrophy	145 (49%)
normal	147 (50%)
wave abnormality	4 (1.4%)
thalach, Mean (SD)	149.56 (22.97)
exang, n (%)	97 (33%)

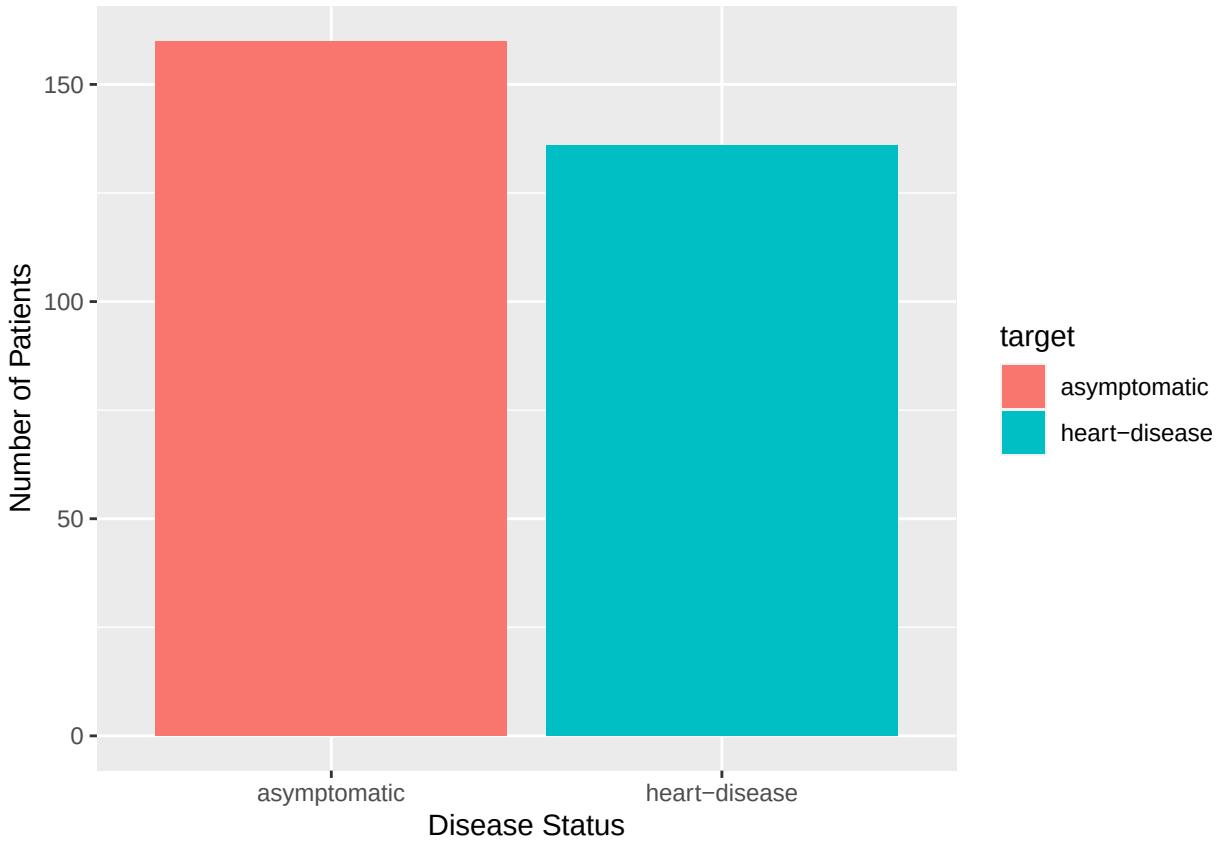
Characteristic	N = 296
oldpeak, Mean (SD)	1.06 (1.17)
slope, n (%)	
downsloping	21 (7.1%)
flat	137 (46%)
upsloping	138 (47%)
ca, n (%)	
0	173 (58%)
1	65 (22%)
2	38 (13%)
3	20 (6.8%)
thal, n (%)	
fixed defect	18 (6.1%)
normal	163 (55%)
reversable defect	115 (39%)
target, n (%)	
asymptomatic	160 (54%)
heart-disease	136 (46%)

Target: Whether patient has heart disease or not

```
hd_EDA %>% group_by(target) %>% count()
```

```
## # A tibble: 2 x 2
## # Groups:   target [2]
##   target            n
##   <fct>         <int>
## 1 asymptomatic    160
## 2 heart-disease   136
```

```
ggplot(hd_EDA, aes(target, fill = target)) + geom_bar() + xlab("Disease Status") + ylab("Number of Pati
```



Age: age in years

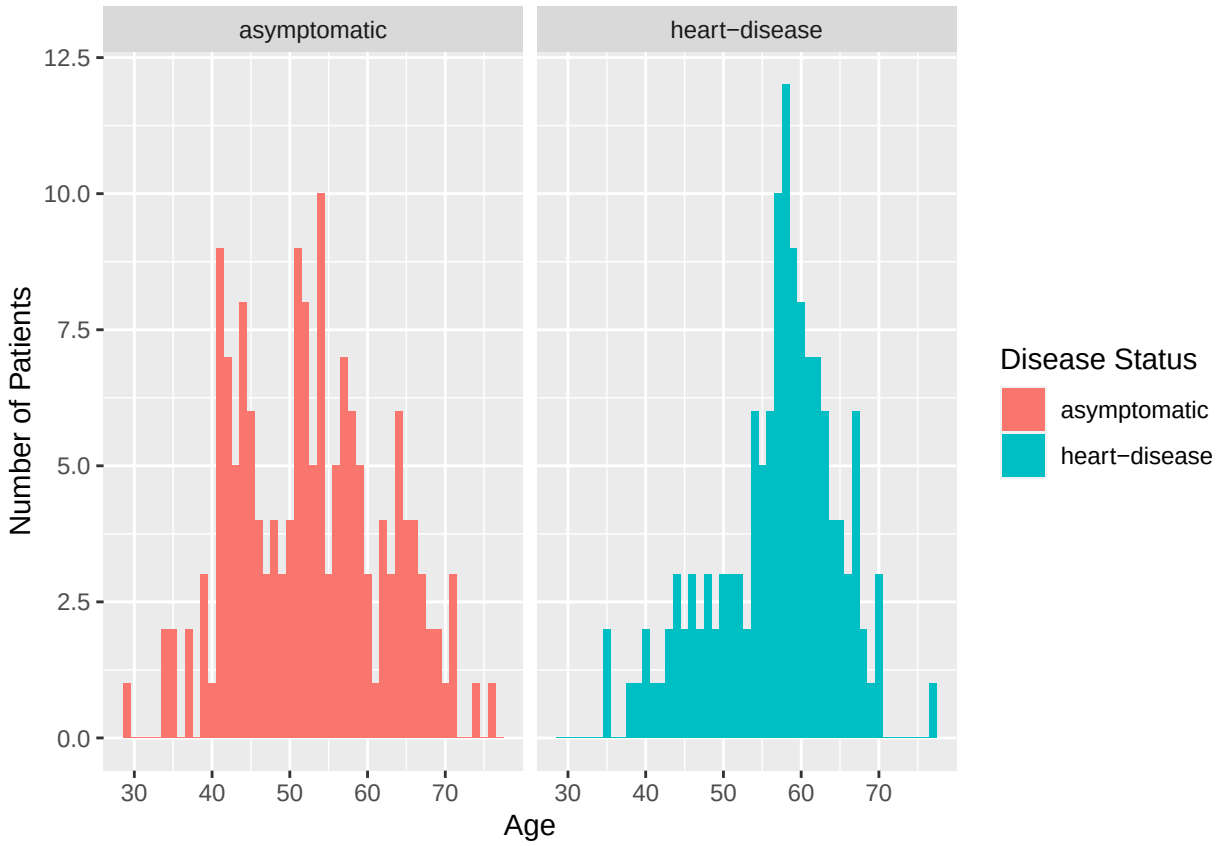
```
hd_EDA %>% summarise(Mean_Age = mean(age), Median_Age = median(age), SD_age = sd(age))
```

```
##   Mean_Age Median_Age   SD_age
## 1 54.52365         56 9.059471
```

```
hd_EDA %>% group_by(target) %>% summarise(Mean_Age = mean(age), Median_Age = median(age), SD_age = sd(ag
```

```
## # A tibble: 2 x 4
##   target        Mean_Age Median_Age SD_age
##   <fct>            <dbl>      <dbl>  <dbl>
## 1 asymptomatic      52.6         52   9.55
## 2 heart-disease     56.7         58   7.92
```

```
ggplot(hd_EDA, aes(age, fill = target)) + geom_histogram(binwidth = 1) + labs(fill = "Disease Status", x
```

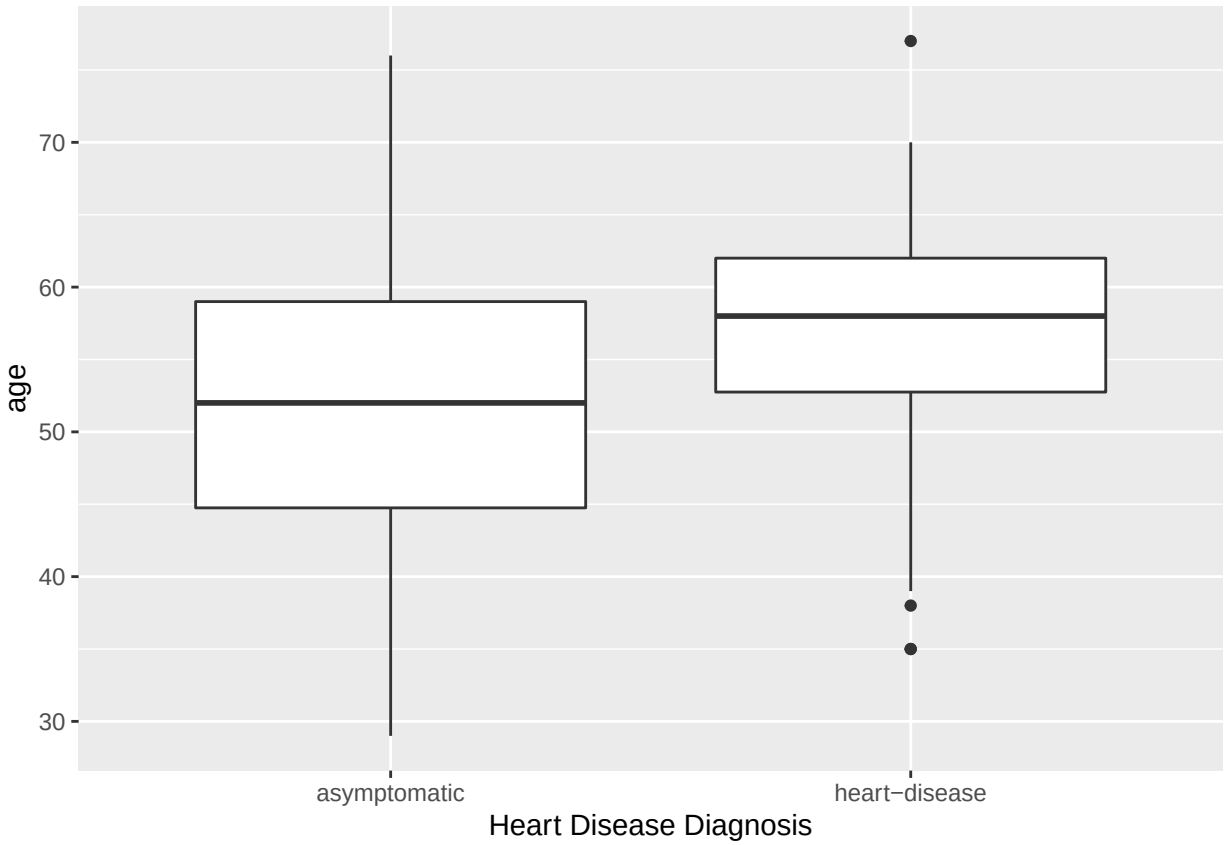


Does age have an effect? Let's use a t-test

```
hd_age <- t.test(hd_EDA$age ~ hd_EDA$target)
print(hd_age)
```

```
##
##  Welch Two Sample t-test
##
## data:  hd_EDA$age by hd_EDA$target
## t = -4.0279, df = 293.84, p-value = 7.17e-05
## alternative hypothesis: true difference in means between group asymptomatic and group heart-disease i
## 95 percent confidence interval:
##  -6.090695 -2.092393
## sample estimates:
##  mean in group asymptomatic mean in group heart-disease
##                    52.64375                    56.73529
```

```
ggplot(data = hd_EDA, aes(x = target, y = age)) + geom_boxplot() + xlab("Heart Disease Diagnosis")
```



Sex: Patient Sex

```
hd_EDA %>% group_by(sex) %>% count()
```

```
## # A tibble: 2 x 2
## # Groups:   sex [2]
##   sex        n
##   <fct>  <int>
## 1 female    95
## 2 male     201
```

```
hd_EDA %>% group_by(target, sex) %>% count()
```

```
## # A tibble: 4 x 3
## # Groups:   target, sex [4]
##   target        sex        n
##   <fct>         <fct>  <int>
## 1 asymptomatic  female    71
## 2 asymptomatic  male      89
## 3 heart-disease female    24
## 4 heart-disease male     112
```

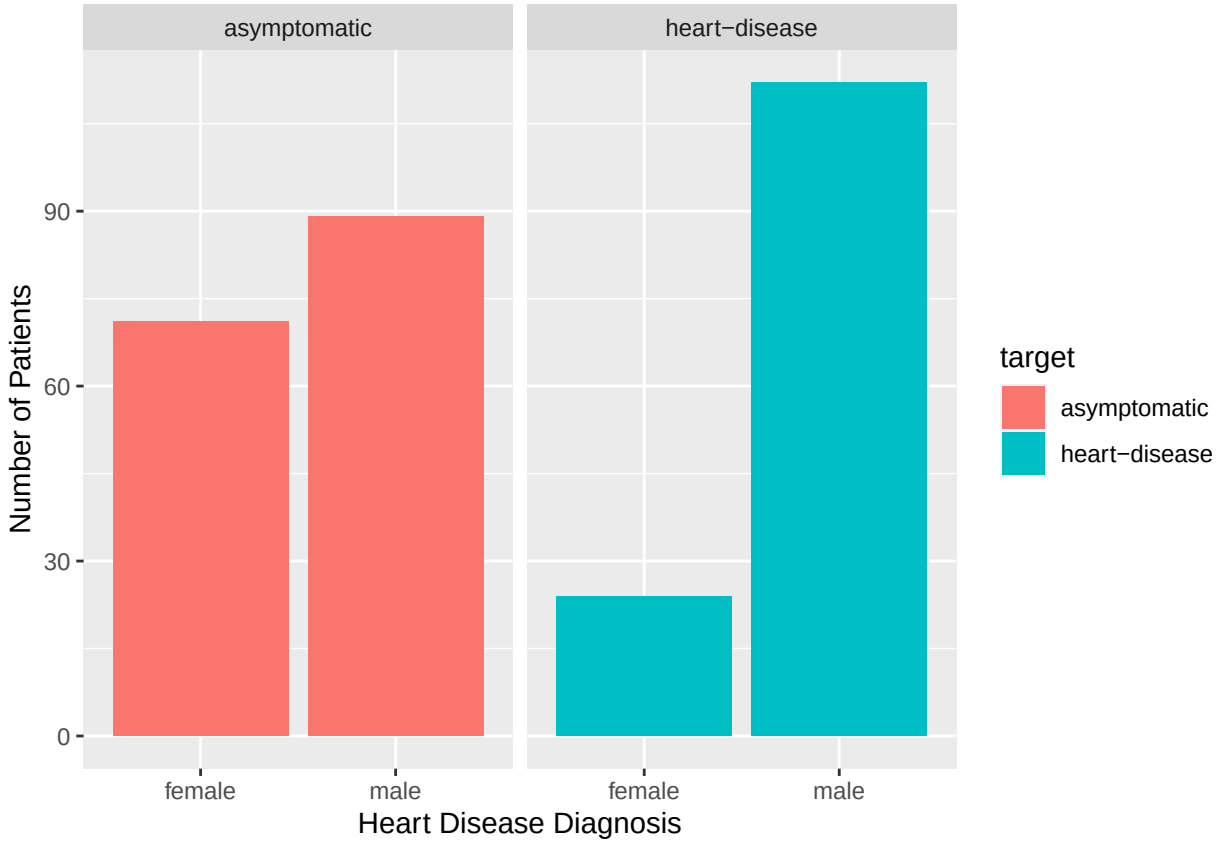
Does sex have an effect? Let's use a chi-squared test

```
hd_sex <- chisq.test(hd_EDA$sex, hd_EDA$target)
print(hd_sex)
```

```
##
```

```
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  hd_EDA$sex and hd_EDA$target
## X-squared = 22.886, df = 1, p-value = 1.719e-06
```

```
ggplot(data = hd_EDA, aes(x = sex, fill = target)) + geom_bar() + ylab("Number of Patients") + xlab("Hea
```



cp: Chest Pain Type

- 0: asymptomatic
- 1: atypical angina
- 2: pain without relation to angina
- 3: typical angina

```
hd_EDA %>% group_by(cp) %>% count()
```

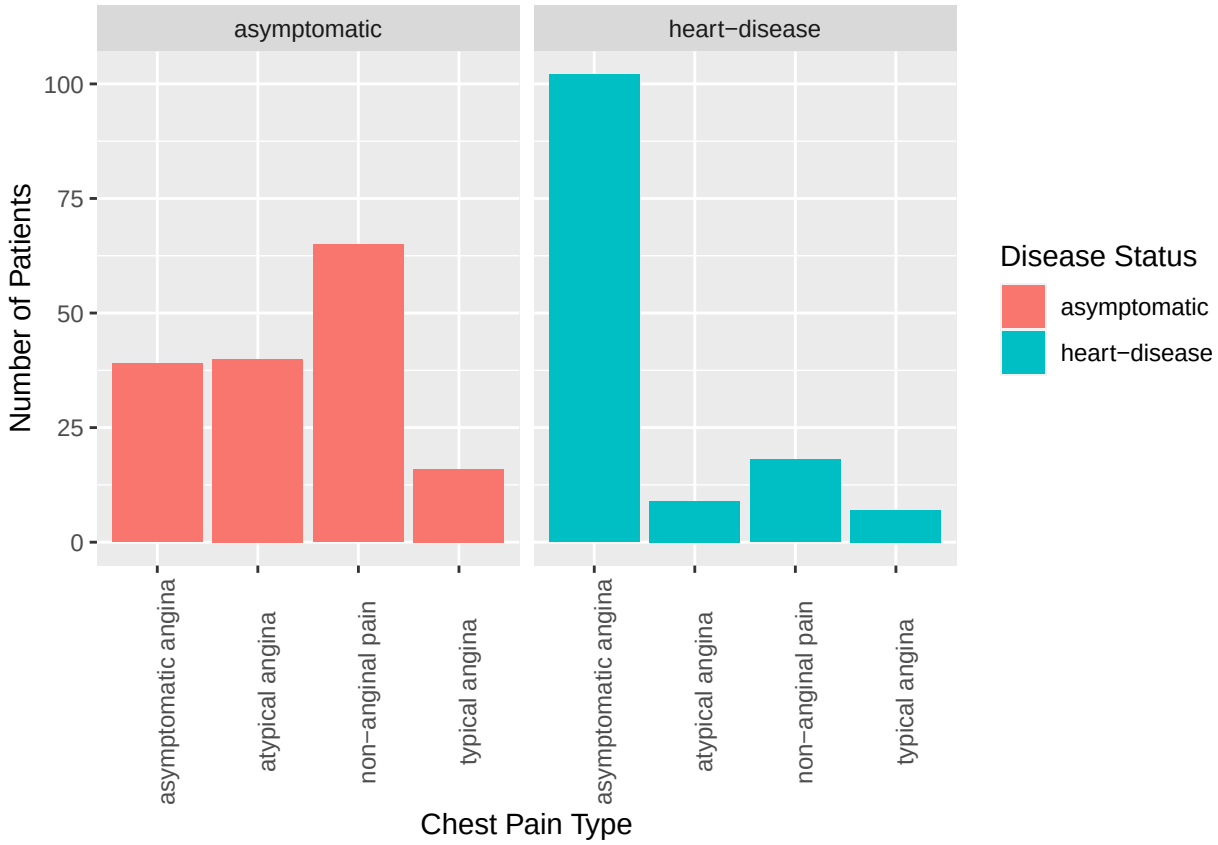
```
## # A tibble: 4 x 2
## # Groups:   cp [4]
##   cp                     n
##   <fct>              <int>
## 1 asymptomatic angina   141
## 2 atypical angina        49
## 3 non-anginal pain       83
## 4 typical angina         23
```

```
hd_EDA %>% group_by(target, cp) %>% count()
```

```
## # A tibble: 8 x 3
## # Groups:   target, cp [8]
```

```
##   target        cp                    n
##   <fct>         <fct>               <int>
## 1 asymptomatic  asymptomatic angina    39
## 2 asymptomatic  atypical angina        40
## 3 asymptomatic  non-anginal pain       65
## 4 asymptomatic  typical angina         16
## 5 heart-disease asymptomatic angina   102
## 6 heart-disease atypical angina         9
## 7 heart-disease non-anginal pain       18
## 8 heart-disease typical angina          7
```

```
ggplot(hd_EDA, aes(cp, fill = target)) + geom_bar() + labs(fill = "Disease Status", x = "Chest Pain Type
  facet_grid(~target)+ scale_color_jama() + theme(axis.text.x = element_text(angle = 90))
```




```
hd_cp <- chisq.test(hd_EDA$cp, hd_EDA$target)
print(hd_cp)
```

```
##
##  Pearson's Chi-squared test
##
## data:  hd_EDA$cp and hd_EDA$target
## X-squared = 76.454, df = 3, p-value < 2.2e-16
```

```
print(hd_cp$residuals)
```

```
##                      hd_EDA$target
## hd_EDA$cp             asymptomatic heart-disease
##   asymptomatic angina    -4.262933      4.623800
##   atypical angina         2.625767     -2.848044
```

```
##   non-anginal pain        3.006086     -3.260558
##   typical angina          1.011799     -1.097450
```

trestbps: Resting Blood Pressure in mmHg at admittance

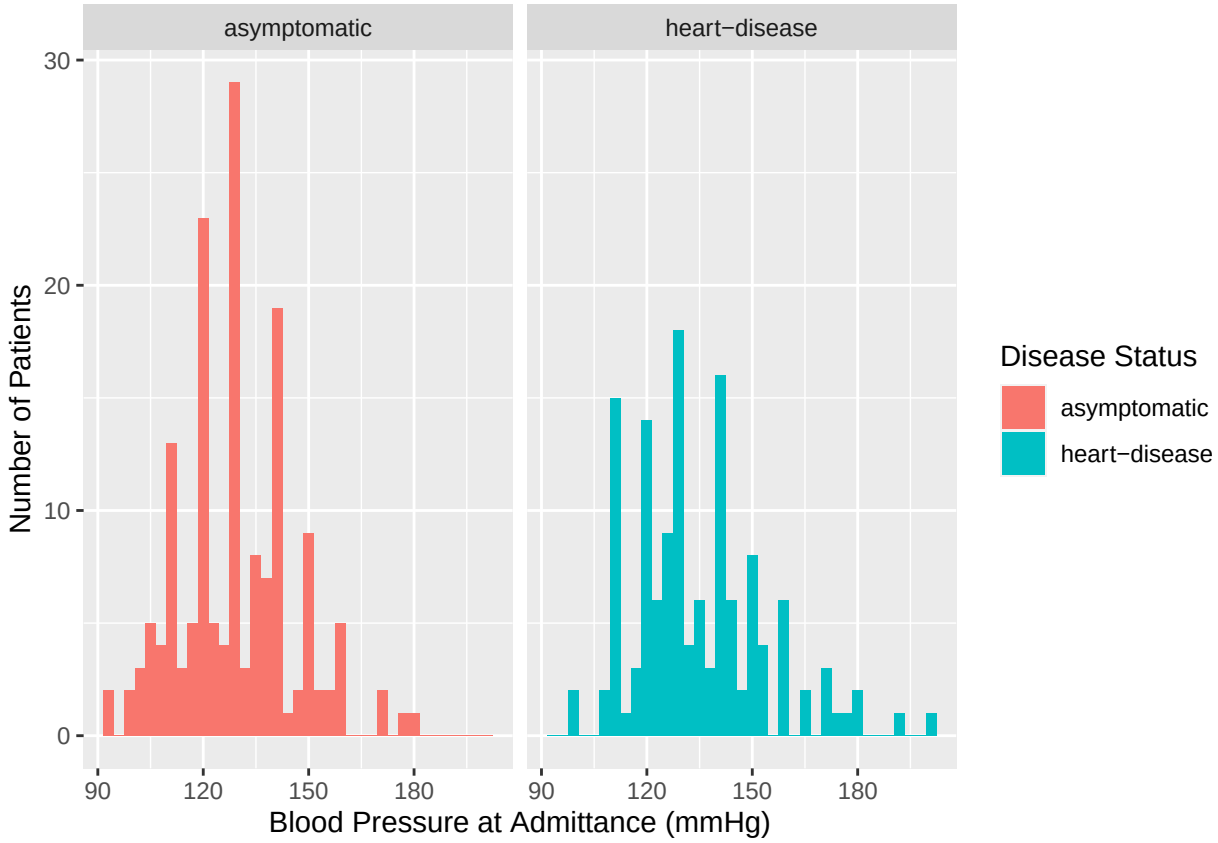
```
hd_EDA %>% summarise(Mean_bps = mean(trestbps), Median_bps = median(trestbps), SD_bps = sd(trestbps))
```

```
##   Mean_bps Median_bps   SD_bps
## 1 131.6047        130 17.72662
```

```
hd_EDA %>% group_by(target) %>% summarise(Mean_bps = mean(trestbps), Median_bps = median(trestbps), SD_
```

```
## # A tibble: 2 x 4
##   target        Mean_bps Median_bps SD_bps
##   <fct>            <dbl>      <dbl>  <dbl>
## 1 asymptomatic      129.        130   16.4
## 2 heart-disease     134.        130   18.9
```

```
ggplot(hd_EDA, aes(trestbps, fill = target)) + geom_histogram(binwidth = 3) +
  labs(fill = "Disease Status", x = "Blood Pressure at Admittance (mmHg)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```




```
hd_bps <- t.test(hd_EDA$trestbps ~ hd_EDA$target)
print(hd_bps)
```

```
##
##  Welch Two Sample t-test
##
## data:  hd_EDA$trestbps by hd_EDA$target
```

```
## t = -2.5529, df = 269.49, p-value = 0.01123
## alternative hypothesis: true difference in means between group asymptomatic and group heart-disease i
## 95 percent confidence interval:
##  -9.366470 -1.210001
## sample estimates:
##  mean in group asymptomatic mean in group heart-disease
##                    129.1750                    134.4632
```

chol: Cholesterol level in mg/dl

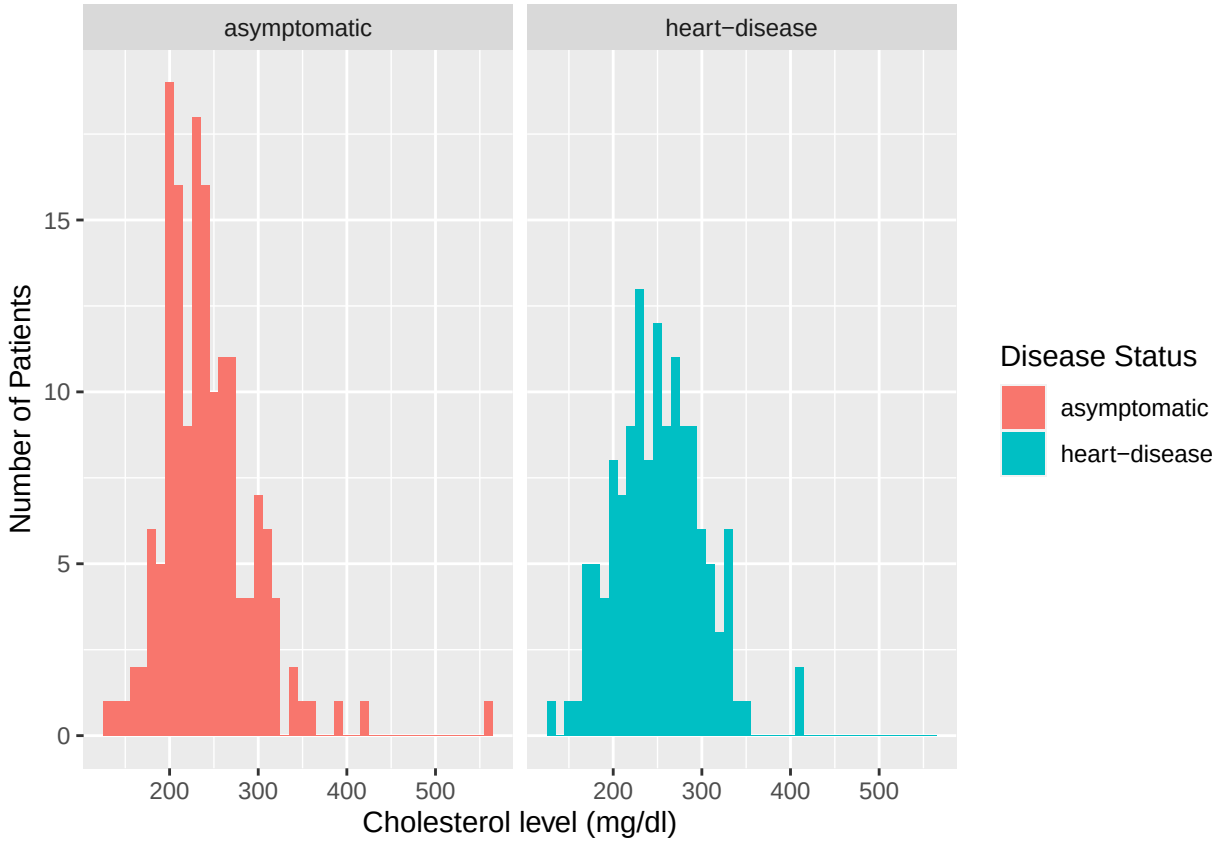
```
hd_EDA %>% summarise(Mean_chol = mean(chol), Median_chol = median(chol), SD_chol = sd(chol))
```

```
##   Mean_chol Median_chol  SD_chol
## 1  247.1554       242.5 51.97701
```

```
hd_EDA %>% group_by(target) %>% summarise(Mean_chol = mean(chol), Median_chol = median(chol), SD_chol =
```

```
## # A tibble: 2 x 4
##   target        Mean_chol Median_chol SD_chol
##   <fct>             <dbl>       <dbl>   <dbl>
## 1 asymptomatic       243.        236.    53.8
## 2 heart-disease      251.        251     49.7
```

```
ggplot(hd_EDA, aes(chol, fill = target)) + geom_histogram(binwidth = 10) +
  labs(fill = "Disease Status", x = "Cholesterol level (mg/dl)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```



```
hd_chol <- t.test(hd_EDA$chol ~ hd_EDA$target)
print(hd_chol)
```

```
##
##  Welch Two Sample t-test
##
## data:  hd_EDA$chol by hd_EDA$target
## t = -1.3248, df = 291.95, p-value = 0.1863
## alternative hypothesis: true difference in means between group asymptomatic and group heart-disease i
## 95 percent confidence interval:
##  -19.809169   3.870198
## sample estimates:
##  mean in group asymptomatic mean in group heart-disease
##                    243.4938                    251.4632
```

fbs: Whether Blood Sugar Level is greater than 120 mg/dl

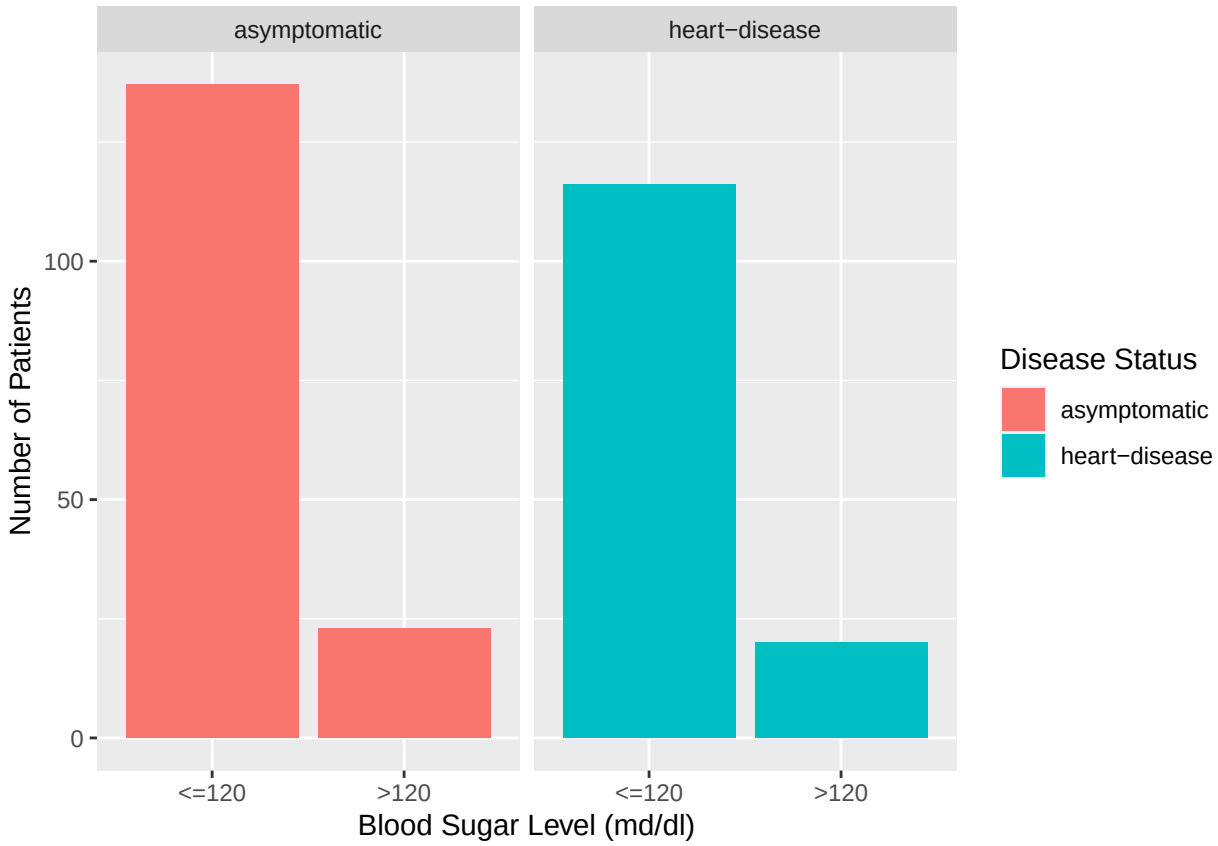
```
hd_EDA %>% group_by(fbs) %>% count()
```

```
## # A tibble: 2 x 2
## # Groups:   fbs [2]
##   fbs       n
##   <fct> <int>
## 1 <=120   253
## 2 >120     43
```

```
hd_EDA %>% group_by(target, fbs) %>% count()
```

```
## # A tibble: 4 x 3
## # Groups:   target, fbs [4]
##   target        fbs       n
##   <fct>         <fct> <int>
## 1 asymptomatic  <=120   137
## 2 asymptomatic  >120     23
## 3 heart-disease <=120   116
## 4 heart-disease >120     20
```

```
ggplot(hd_EDA, aes(fbs, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Blood Sugar Level (md/dl)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```




```
hd_fbs <- chisq.test(hd_EDA$fbs, hd_EDA$target)
print(hd_fbs)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  hd_EDA$fbs and hd_EDA$target
## X-squared = 1.4767e-30, df = 1, p-value = 1
```

restecg: Resting Electrocardiogram

- 0: probable left ventricular hypertrophy
- 1: normal
- 2: abnormalities in the T wave or ST segment

```
hd_EDA %>% group_by(restecg) %>% count()
```

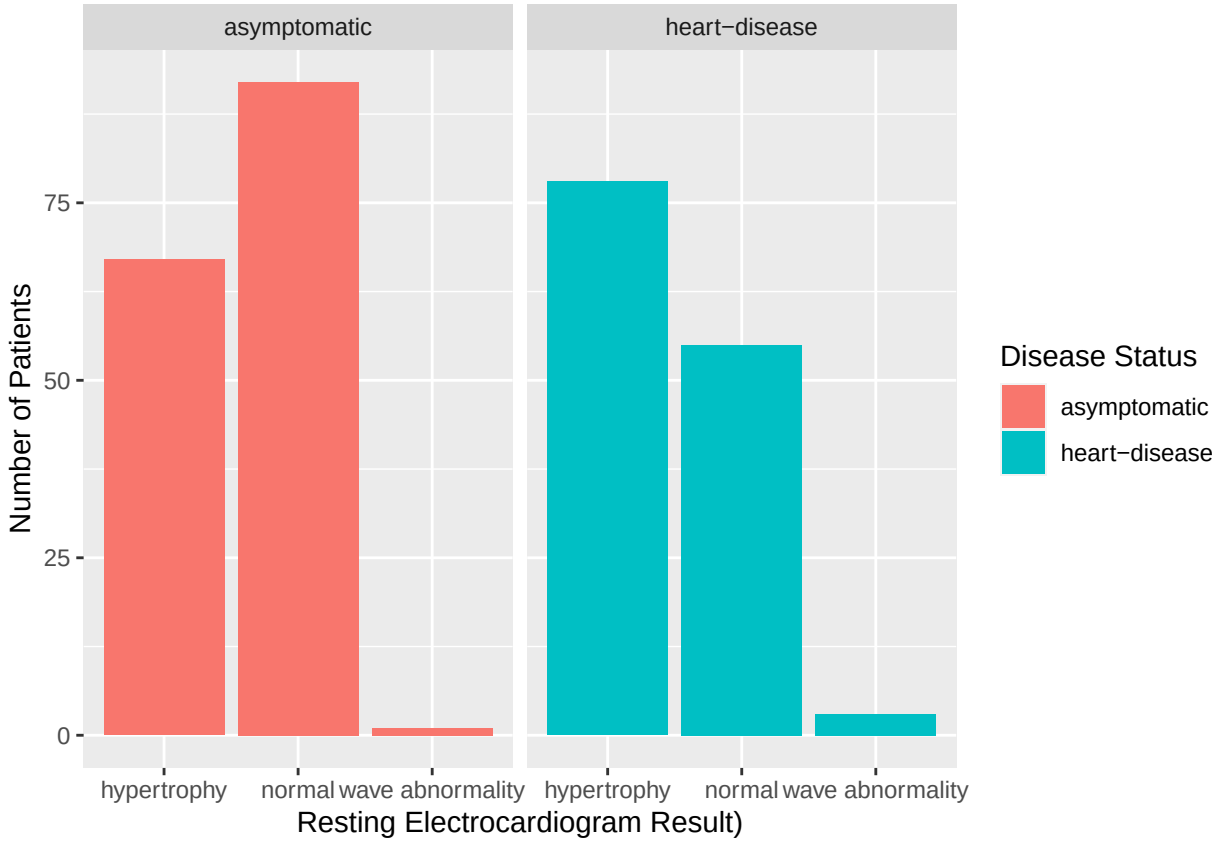
```
## # A tibble: 3 x 2
## # Groups:   restecg [3]
##   restecg              n
##   <fct>            <int>
## 1 hypertrophy        145
## 2 normal             147
## 3 wave abnormality     4
```

```
hd_EDA %>% group_by(target, restecg) %>% count()
```

```
## # A tibble: 6 x 3
## # Groups:   target, restecg [6]
```

```
##   target        restecg              n
##   <fct>         <fct>            <int>
## 1 asymptomatic  hypertrophy          67
## 2 asymptomatic  normal               92
## 3 asymptomatic  wave abnormality      1
## 4 heart-disease hypertrophy          78
## 5 heart-disease normal               55
## 6 heart-disease wave abnormality      3
```

```
ggplot(hd_EDA, aes(restecg, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Resting Electrocardiogram Result)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```



use different test

```
hd_ecg <- chisq.test(hd_EDA$restecg, hd_EDA$target)
```

```
## Warning in chisq.test(hd_EDA$restecg, hd_EDA$target): Chi-squared approximation
## may be incorrect
```

```
print(hd_ecg)
```

```
##
##  Pearson's Chi-squared test
##
## data:  hd_EDA$restecg and hd_EDA$target
## X-squared = 9.2624, df = 2, p-value = 0.009743
```

```
print(hd_ecg$residuals)
```

```
##                    hd_EDA$target
## hd_EDA$restecg      asymptomatic heart-disease
##   hypertrophy         -1.2852341     1.3940321
##   normal               1.4068359    -1.5259278
##   wave abnormality    -0.7903557     0.8572611
```

thalach: Maximum Heart Rate during Stress Test

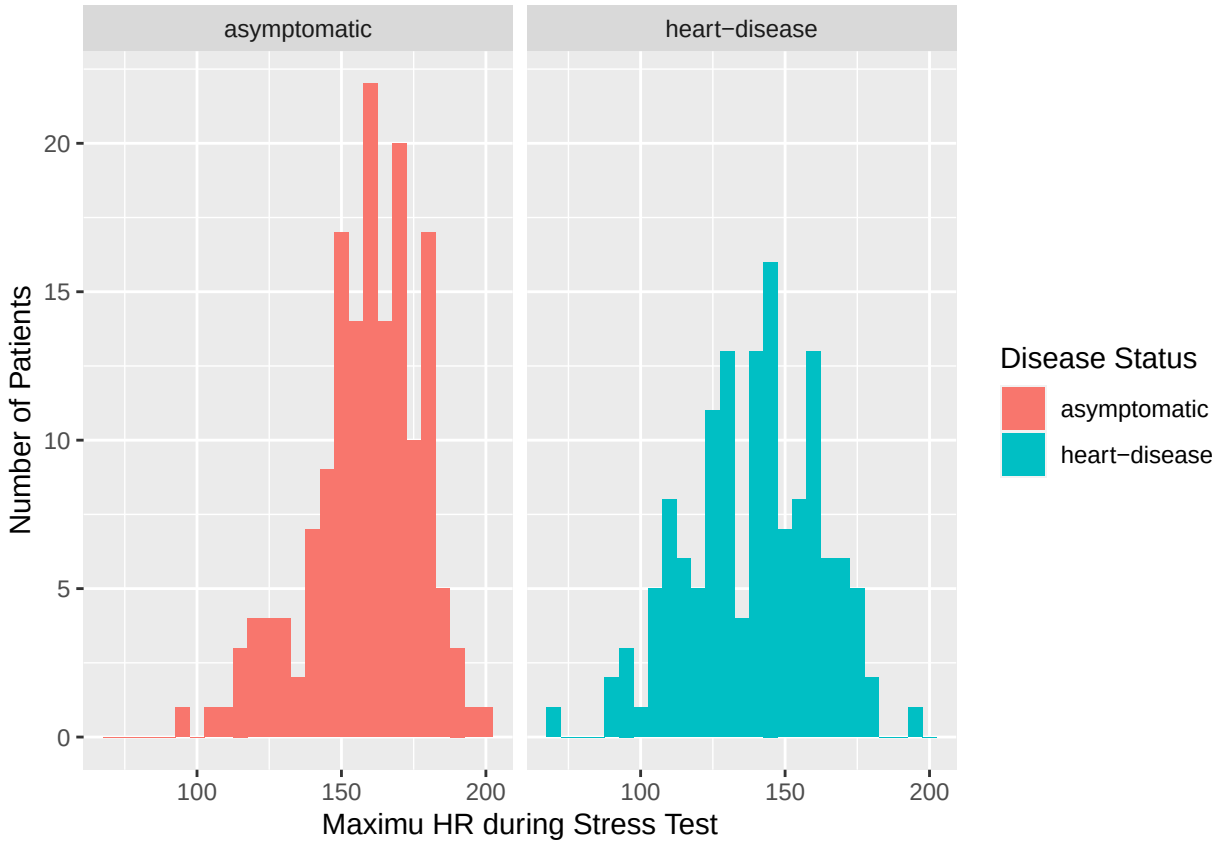
```
hd_EDA %>% summarise(Mean_thalach = mean(thalach), Median_thalach = median(thalach), SD_thalach = sd(tha
```

```
##   Mean_thalach Median_thalach SD_thalach
## 1     149.5608          152.5   22.97079
```

```
hd_EDA %>% group_by(target) %>% summarise(Mean_thalach = mean(thalach), Median_thalach = median(thalach
```

```
## # A tibble: 2 x 4
##   target        Mean_thalach Median_thalach SD_thalach
##   <fct>                <dbl>          <dbl>      <dbl>
## 1 asymptomatic          159.           161        19.0
## 2 heart-disease         139.           142.       22.7
```

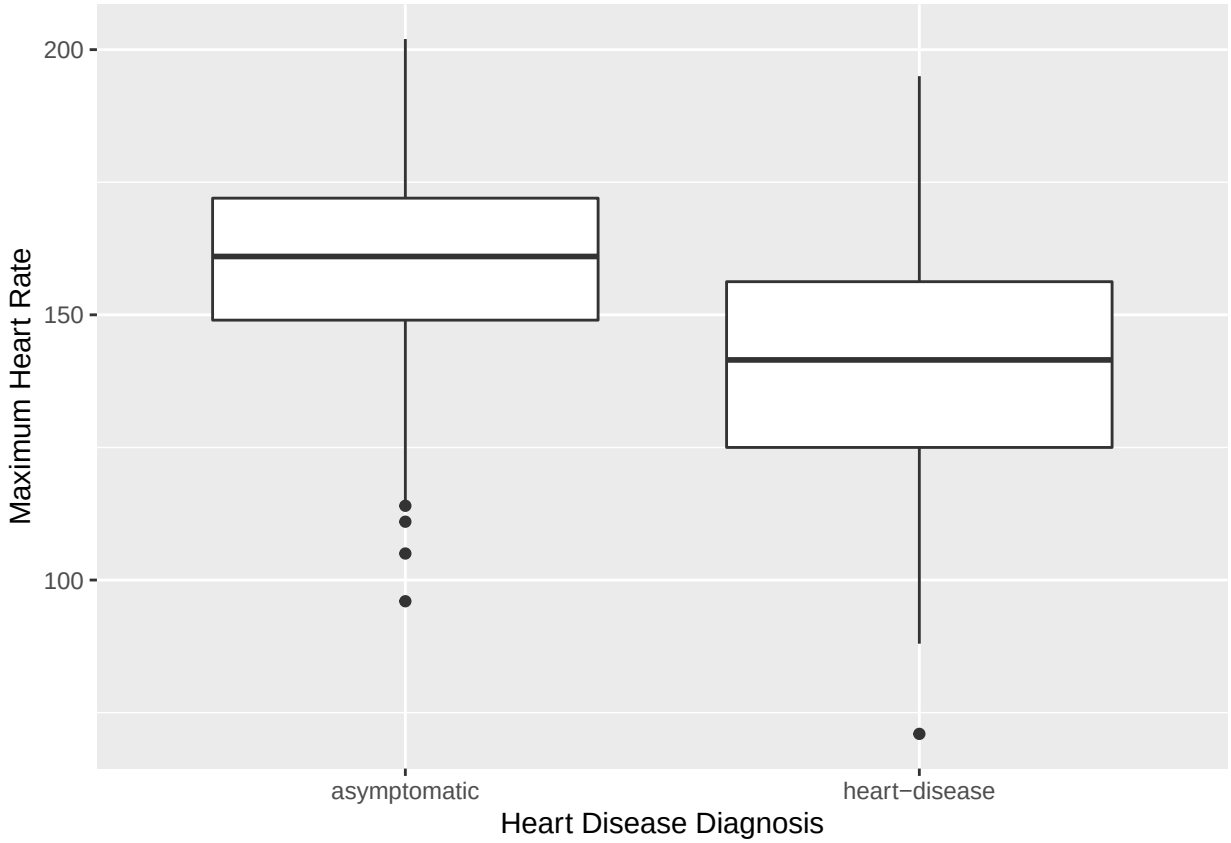
```
ggplot(hd_EDA, aes(thalach, fill = target)) + geom_histogram(binwidth = 5) +
  labs(fill = "Disease Status", x = "Maximu HR during Stress Test", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```



```
hd_thalach <- t.test(hd_EDA$thalach ~ hd_EDA$target)
print(hd_thalach)
```

```
##
##  Welch Two Sample t-test
##
## data:  hd_EDA$thalach by hd_EDA$target
## t = 7.9747, df = 264.36, p-value = 4.628e-14
## alternative hypothesis: true difference in means between group asymptomatic and group heart-disease i
## 95 percent confidence interval:
##  14.78535 24.48009
## sample estimates:
##  mean in group asymptomatic mean in group heart-disease
##                    158.5813                    138.9485
```

```
ggplot(data = hd_EDA, aes(x = target, y = thalach)) + geom_boxplot() +
  xlab("Heart Disease Diagnosis") + ylab("Maximum Heart Rate") +
  scale_color_jama()
```



exang: Whether Patient had Angina During Exercise

```
hd_EDA %>% group_by(exang) %>% count()
```

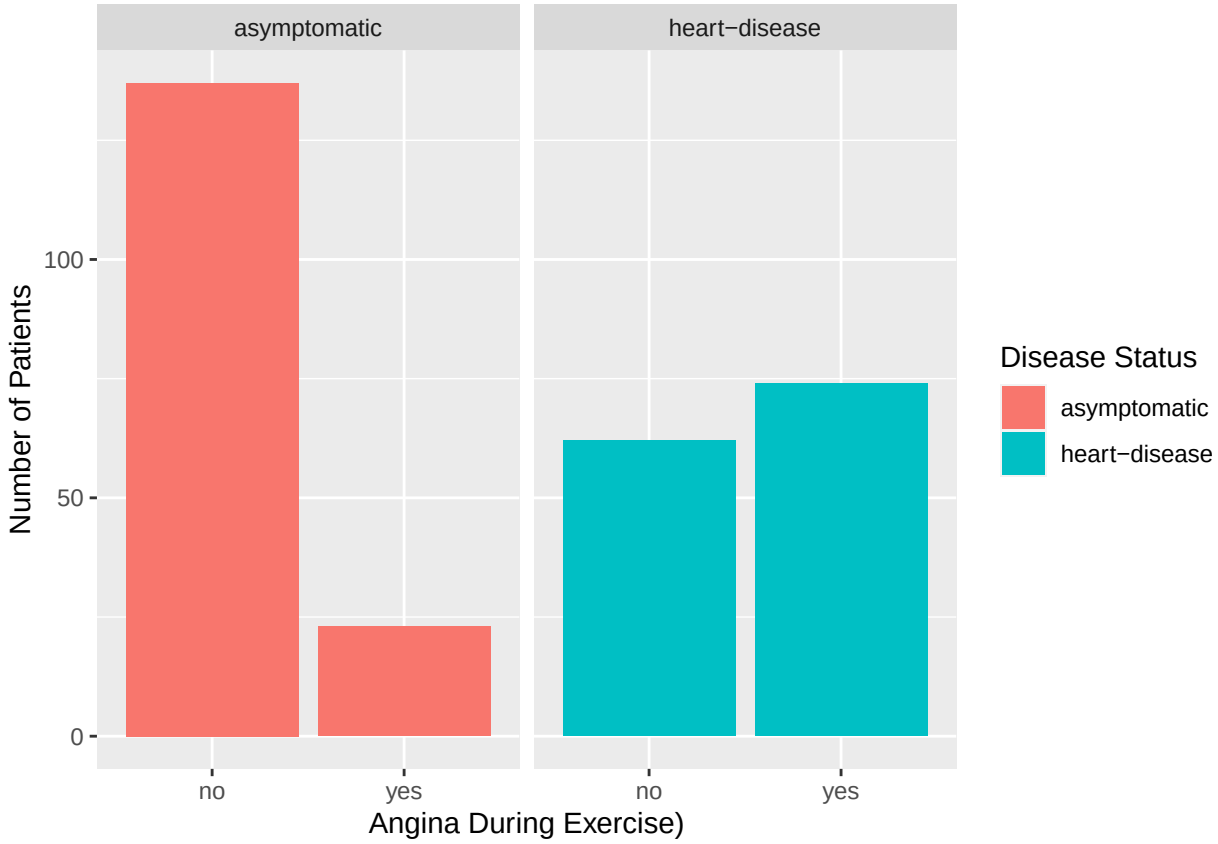
```
## # A tibble: 2 x 2
## # Groups:   exang [2]
##   exang     n
##   <fct> <int>
```

```
## 1 no      199
## 2 yes      97
```

```
hd_EDA %>% group_by(target, exang) %>% count()
```

```
## # A tibble: 4 x 3
## # Groups:   target, exang [4]
##   target        exang     n
##   <fct>         <fct> <int>
## 1 asymptomatic  no      137
## 2 asymptomatic  yes      23
## 3 heart-disease no       62
## 4 heart-disease yes      74
```

```
ggplot(hd_EDA, aes(exang, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Angina During Exercise)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```




```
hd_exang <- chisq.test(hd_EDA$exang, hd_EDA$target)
print(hd_exang)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  hd_EDA$exang and hd_EDA$target
## X-squared = 51.685, df = 1, p-value = 6.517e-13
```

oldpeak: Decrease of the ST segment during Exercise according to the same one on rest

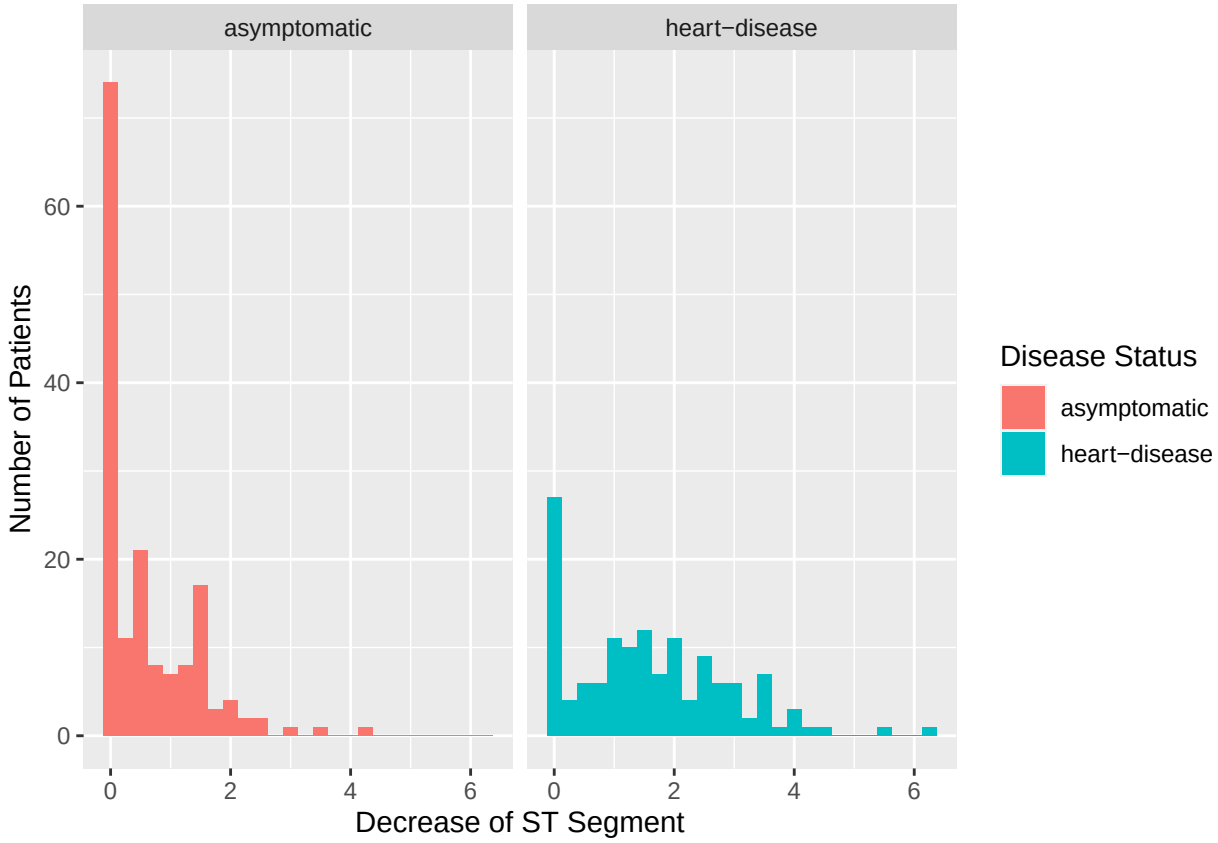
```
hd_EDA %>% summarise(Mean_oldpeak = mean(oldpeak), Median_oldpeak = median(oldpeak), SD_oldpeak = sd(old
```

```
##   Mean_oldpeak Median_oldpeak SD_oldpeak
## 1     1.059122            0.8   1.166474
```

```
hd_EDA %>% group_by(target) %>% summarise(Mean_oldpeak = mean(oldpeak), Median_oldpeak = median(oldpeak
```

```
## # A tibble: 2 x 4
##   target        Mean_oldpeak Median_oldpeak SD_oldpeak
##   <fct>                <dbl>          <dbl>      <dbl>
## 1 asymptomatic         0.599            0.2      0.787
## 2 heart-disease        1.60             1.4      1.30
```

```
ggplot(hd_EDA, aes(oldpeak, fill = target)) + geom_histogram(binwidth = 0.25) +
  labs(fill = "Disease Status", x = "Decrease of ST Segment", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```




```
hd_oldpeak <- t.test(hd_EDA$oldpeak ~ hd_EDA$target)
print(hd_oldpeak)
```

```
##
##  Welch Two Sample t-test
##
## data:  hd_EDA$oldpeak by hd_EDA$target
## t = -7.8363, df = 214.28, p-value = 2.139e-13
## alternative hypothesis: true difference in means between group asymptomatic and group heart-disease
## 95 percent confidence interval:
```

```
##  -1.254018 -0.749953
## sample estimates:
##  mean in group asymptomatic mean in group heart-disease
##                    0.598750                    1.600735
```

slope: Slope of the ST segment during the most demanding part of exercise

- 0: descending
- 1: flat
- 2: ascending

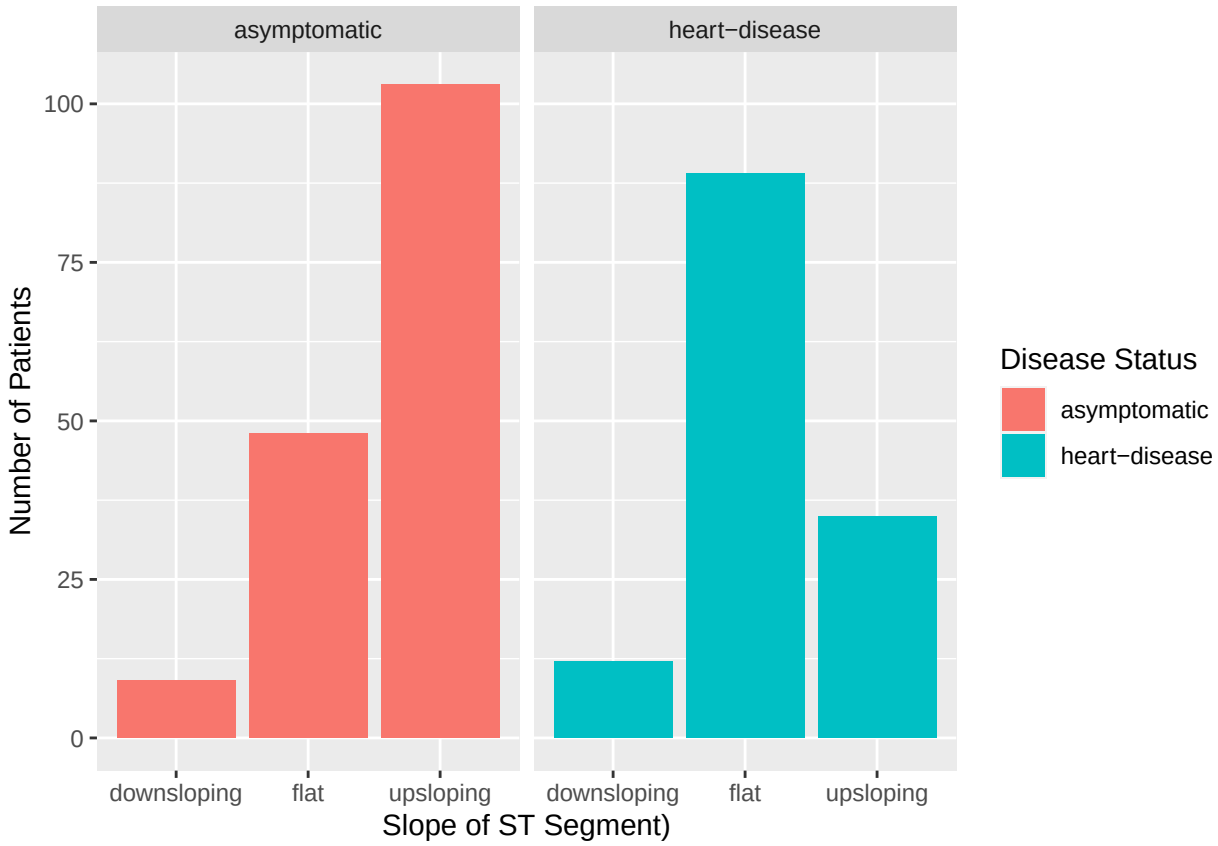
```
hd_EDA %>% group_by(slope) %>% count()
```

```
## # A tibble: 3 x 2
## # Groups:   slope [3]
##   slope           n
##   <fct>       <int>
## 1 downsloping    21
## 2 flat          137
## 3 upsloping     138
```

```
hd_EDA %>% group_by(target, slope) %>% count()
```

```
## # A tibble: 6 x 3
## # Groups:   target, slope [6]
##   target        slope           n
##   <fct>         <fct>       <int>
## 1 asymptomatic  downsloping     9
## 2 asymptomatic  flat           48
## 3 asymptomatic  upsloping     103
## 4 heart-disease downsloping    12
## 5 heart-disease flat           89
## 6 heart-disease upsloping      35
```

```
ggplot(hd_EDA, aes(slope, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Slope of ST Segment)", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```




```
hd_exang <- chisq.test(hd_EDA$exang, hd_EDA$target)
print(hd_exang)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  hd_EDA$exang and hd_EDA$target
## X-squared = 51.685, df = 1, p-value = 6.517e-13
```

thal: Results of the blood flow observed via radioactive dye

- 0: NULL (dropped from the dataset previously)
- 1: fixed defect (no blood flow in some part of the heart)
- 2: normal blood flow
- 3: reversible defect (a blood flow is observed but it is not normal)

```
hd_EDA %>% group_by(thal) %>% count()
```

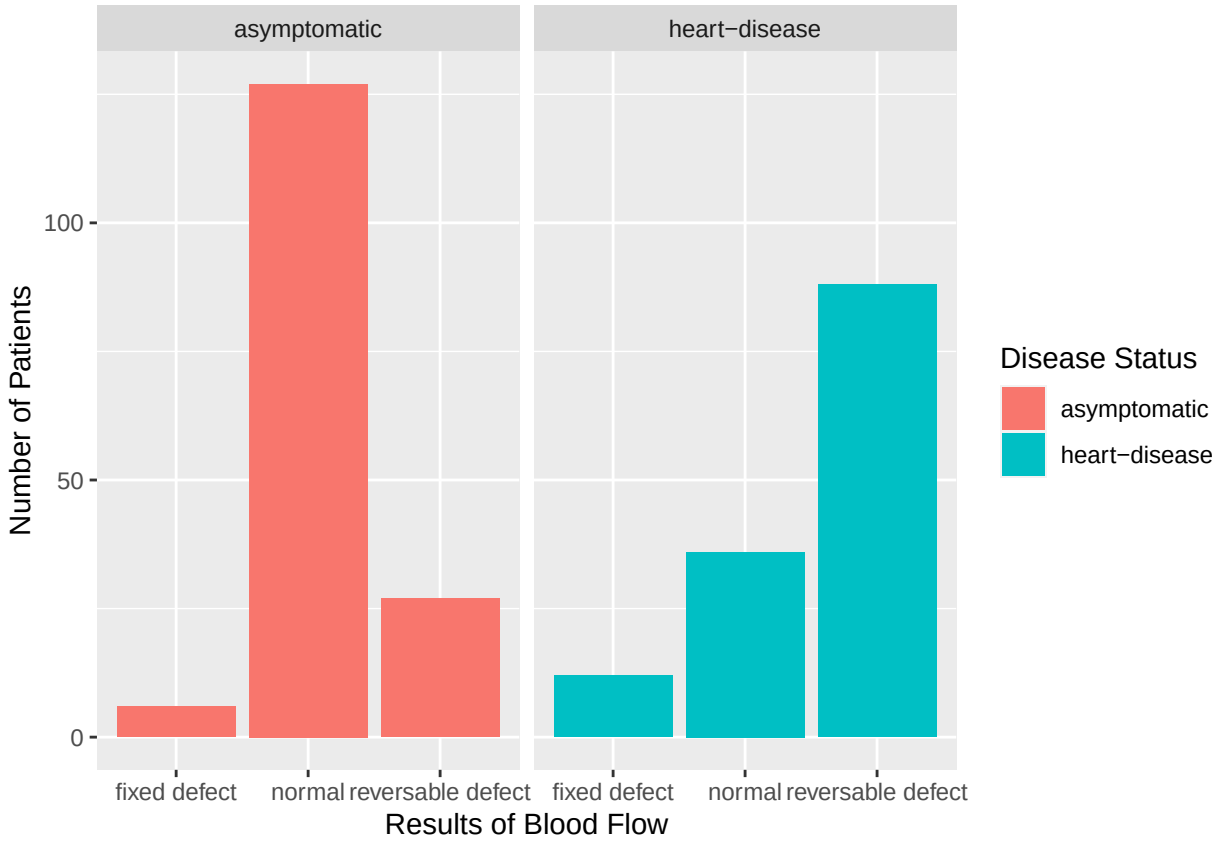
```
## # A tibble: 3 x 2
## # Groups:   thal [3]
##   thal                 n
##   <fct>            <int>
## 1 fixed defect        18
## 2 normal             163
## 3 reversable defect  115
```

```
hd_EDA %>% group_by(target, thal) %>% count()
```

```
## # A tibble: 6 x 3
```

```
## # Groups:   target, thal [6]
##   target        thal                 n
##   <fct>         <fct>            <int>
## 1 asymptomatic  fixed defect         6
## 2 asymptomatic  normal             127
## 3 asymptomatic  reversable defect   27
## 4 heart-disease fixed defect        12
## 5 heart-disease normal              36
## 6 heart-disease reversable defect   88
```

```
ggplot(hd_EDA, aes(thal, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Results of Blood Flow", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```



need different test

```
hd_exang <- chisq.test(hd_EDA$exang, hd_EDA$target)
print(hd_exang)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  hd_EDA$exang and hd_EDA$target
## X-squared = 51.685, df = 1, p-value = 6.517e-13
```

ca: Number of main blood vessels coloured by radioactive dye

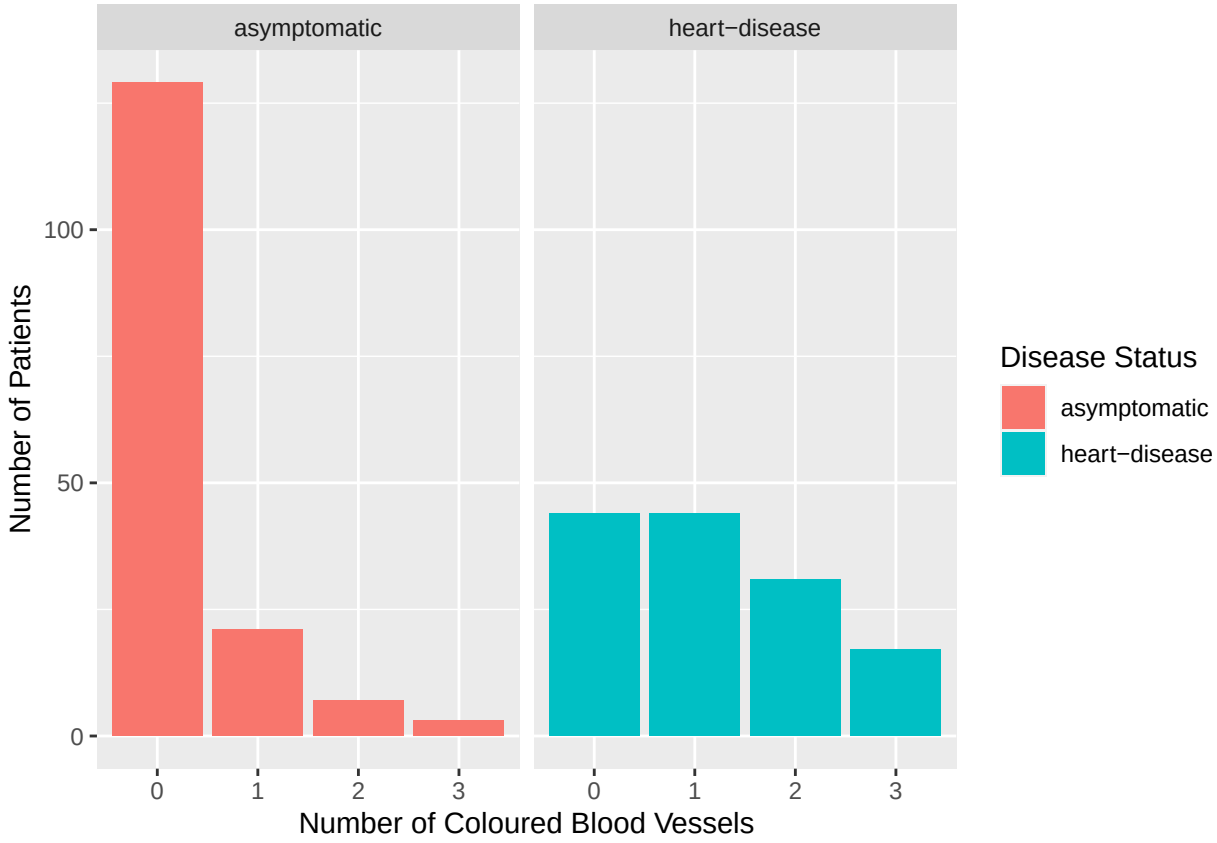
```
hd_EDA %>% group_by(ca) %>% count()
```

```
## # A tibble: 4 x 2
## # Groups:   ca [4]
##   ca        n
##   <fct> <int>
## 1 0       173
## 2 1        65
## 3 2        38
## 4 3        20
```

```
hd_EDA %>% group_by(target, ca) %>% count()
```

```
## # A tibble: 8 x 3
## # Groups:   target, ca [8]
##   target        ca        n
##   <fct>         <fct> <int>
## 1 asymptomatic  0       129
## 2 asymptomatic  1        21
## 3 asymptomatic  2         7
## 4 asymptomatic  3         3
## 5 heart-disease 0        44
## 6 heart-disease 1        44
## 7 heart-disease 2        31
## 8 heart-disease 3        17
```

```
ggplot(hd_EDA, aes(ca, fill = target)) + geom_bar() +
  labs(fill = "Disease Status", x = "Number of Coloured Blood Vessels", y = "Number of Patients") +
  facet_grid(~target)+ scale_color_jama()
```



need different test

```
hd_ca <- chisq.test(hd_EDA$ca, hd_EDA$target)
print(hd_ca)
```

```
##
##  Pearson's Chi-squared test
##
## data:  hd_EDA$ca and hd_EDA$target
## X-squared = 73.396, df = 3, p-value = 7.996e-16
```

Wilcoxon rank sum test; Pearson's Chi-squared test; Fisher's exact test

```
hd_EDA %>% filter(thal != 0 & ca != 4) %>%
  tbl_summary(by = target) %>% add_p()
```

Characteristic	**asymptomatic**, N = 160	**heart-disease**, N = 136	**p-value**
age, Median (IQR)	52 (45 – 59)	58 (53 – 62)	<0.001
sex, n (%)			<0.001
female	71 (44)	24 (18)	
male	89 (56)	112 (82)	
cp, n (%)			<0.001
asymptomatic angina	39 (24)	102 (75)	
atypical angina	40 (25)	9 (6.6)	
non-anginal pain	65 (41)	18 (13)	
typical angina	16 (10)	7 (5.1)	
trestbps, Median (IQR)	130 (120 – 140)	130 (120 – 145)	0.029
chol, Median (IQR)	236 (209 – 268)	251 (218 – 283)	0.056
fbs, n (%)			0.94
<=120	137 (86)	116 (85)	
>120	23 (14)	20 (15)	
restecg, n (%)			0.006
hypertrophy	67 (42)	78 (57)	
normal	92 (57)	55 (40)	
wave abnormality	1 (0.6)	3 (2.2)	
thalach, Median (IQR)	161 (149 – 172)	142 (125 – 156)	<0.001
exang, n (%)	23 (14)	74 (54)	<0.001
oldpeak, Median (IQR)	0.20 (0.00 – 1.10)	1.40 (0.60 – 2.52)	<0.001
slope, n (%)			<0.001
downsloping	9 (5.6)	12 (8.8)	
flat	48 (30)	89 (65)	
upsloping	103 (64)	35 (26)	
ca, n (%)			<0.001
0	129 (81)	44 (32)	
1	21 (13)	44 (32)	
2	7 (4.4)	31 (23)	
3	3 (1.9)	17 (12)	
thal, n (%)			<0.001
fixed defect	6 (3.8)	12 (8.8)	
normal	127 (79)	36 (26)	
reversable defect	27 (17)	88 (65)	

age: Chi-Squared - heart disease older than asymptomatic sex: Chi-Squared - More males with heart disease cp:

Machine Learning

Training and Test sets

```
hd_model <- na.omit(hd_model)

set.seed(44)
train.samples <- hd_model$target %>%
  createDataPartition(p = 0.7, list = FALSE)

train_data <- hd_model[train.samples, ]
test_data <- hd_model[-train.samples, ]
```

Logistic Regression

full model

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

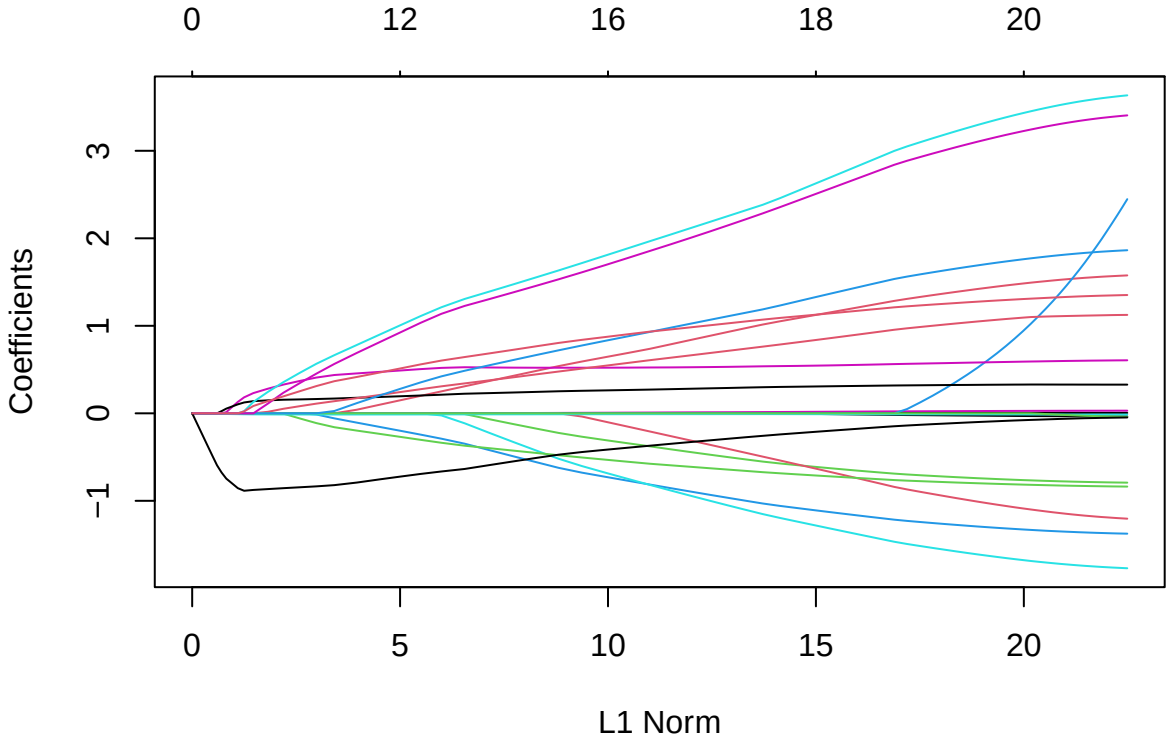
```
##
## Call:
## glm(formula = target ~ ., family = "binomial", data = train_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6582  -0.4668  -0.1629   0.2986   2.7583
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -3.544e+00  3.727e+00  -0.951 0.341611
## age                      -3.759e-02  3.038e-02  -1.237 0.216091
## sexmale                   1.603e+00  6.561e-01   2.443 0.014558 *
## cpatypical angina        -8.002e-01  6.875e-01  -1.164 0.244464
## cpnon-anginal pain       -1.389e+00  6.240e-01  -2.225 0.026068 *
## cptypical angina         -1.797e+00  7.824e-01  -2.296 0.021658 *
## trestbps                  3.081e-02  1.463e-02   2.106 0.035174 *
## chol                      4.492e-03  4.589e-03   0.979 0.327625
## fbs>120                  -1.237e+00  7.713e-01  -1.604 0.108615
## restecgnormal            -8.433e-01  4.908e-01  -1.718 0.085780 .
## restecgwave abnormality   1.251e+01  1.455e+03   0.009 0.993140
## thalach                  -1.426e-02  1.500e-02  -0.951 0.341730
## exangyes                  6.107e-01  5.141e-01   1.188 0.234898
## oldpeak                   3.269e-01  2.961e-01   1.104 0.269562
## slopeflat                 1.131e+00  1.172e+00   0.965 0.334414
## slopeupsloping           -5.607e-02  1.287e+00  -0.044 0.965246
## ca1                       1.893e+00  6.337e-01   2.987 0.002815 **
## ca2                       3.691e+00  9.493e-01   3.888 0.000101 ***
## ca3                       3.456e+00  1.302e+00   2.653 0.007972 **
## thalnormal               -3.736e-02  9.609e-01  -0.039 0.968984
## thalreversable defect     1.366e+00  9.056e-01   1.508 0.131562
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 287.12  on 207  degrees of freedom
## Residual deviance: 131.62  on 187  degrees of freedom
## AIC: 173.62
##
## Number of Fisher Scoring iterations: 14
```

LASSO

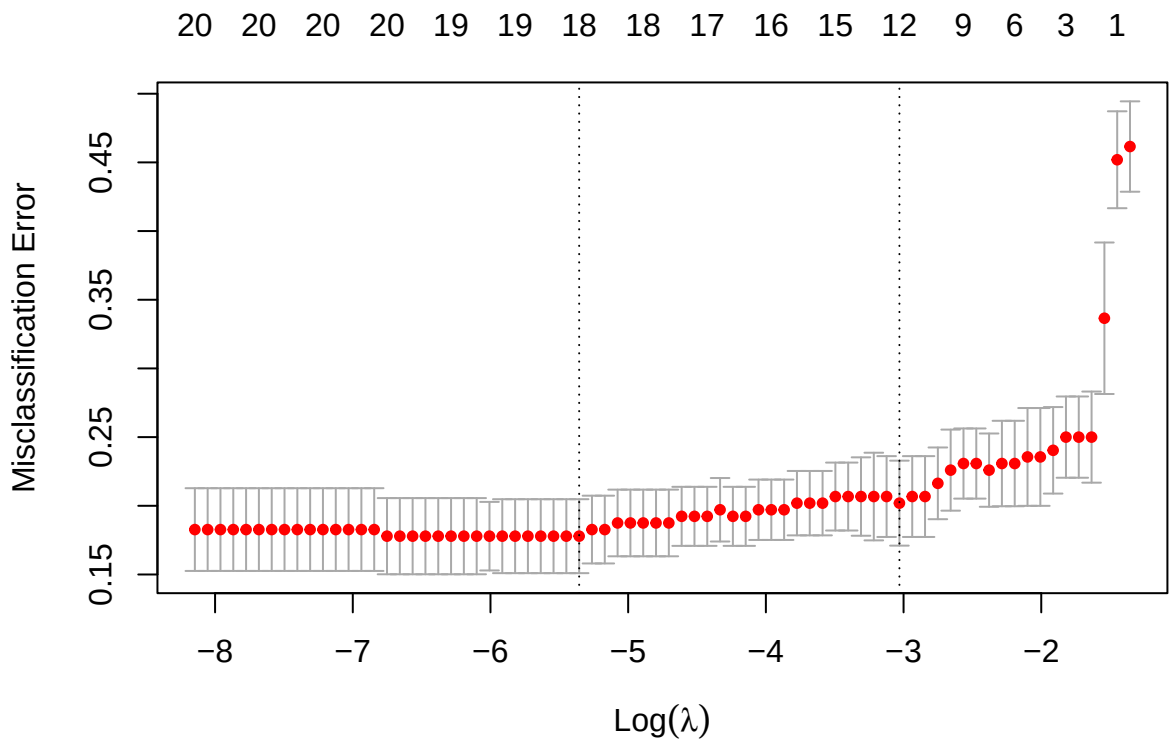
```
x <- model.matrix(target ~ ., train_data)[,-1]
y <- ifelse(train_data$target == "heart-disease", 1 , 0)
```

```
log_lasso <- glmnet(x, y, family = "binomial", alpha = 1, lambda = NULL)
plot(log_lasso)
```



Let's use misclassification error in 10-fold cross validation

```
cv_lasso <- cv.glmnet(x, y, family = "binomial", alpha = 1, lambda = NULL, type.measure = "class")
plot(cv_lasso)
```




```
cv_lasso$lambda.min
```

```
## [1] 0.004720117
```

```
coef(cv_lasso, cv_lasso$lambda.min)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                                      s1
## (Intercept)                -3.042598601
## age                        -0.016628301
## sexmale                     1.260908911
## cpatypical angina          -0.678625131
## cpnon-anginal pain         -1.200243655
## cptypical angina           -1.440651588
## trestbps                    0.021423072
## chol                        0.002596477
## fbs>120                    -0.809511812
## restecgnormal              -0.755352474
## restecgwave abnormality     .
## thalach                    -0.011030396
## exangyes                    0.559450883
## oldpeak                     0.316499162
## slopeflat                   0.937434569
## slopeupsloping              .
## ca1                         1.505147359
## ca2                         2.943787546
## ca3                         2.790949925
## thalnormal                 -0.156838670
## thalreversable defect       1.198596506
```

```
cv_lasso$lambda.1se
```

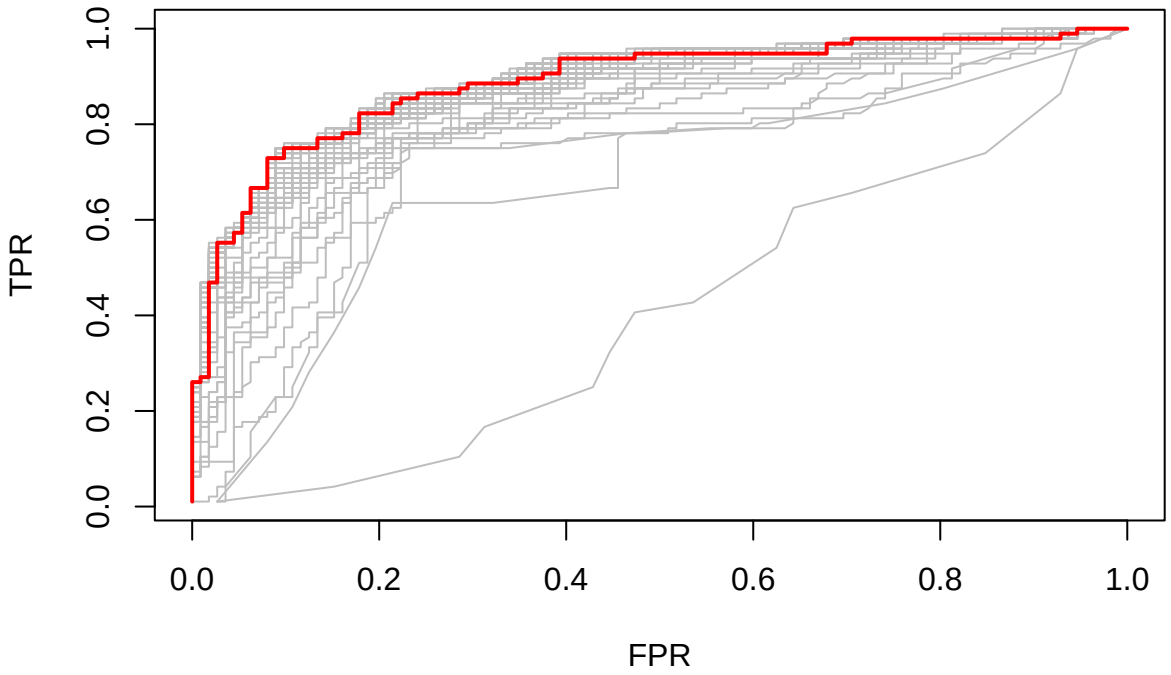
```
## [1] 0.04831186
```

```
coef(cv_lasso, cv_lasso$lambda.1se)
```

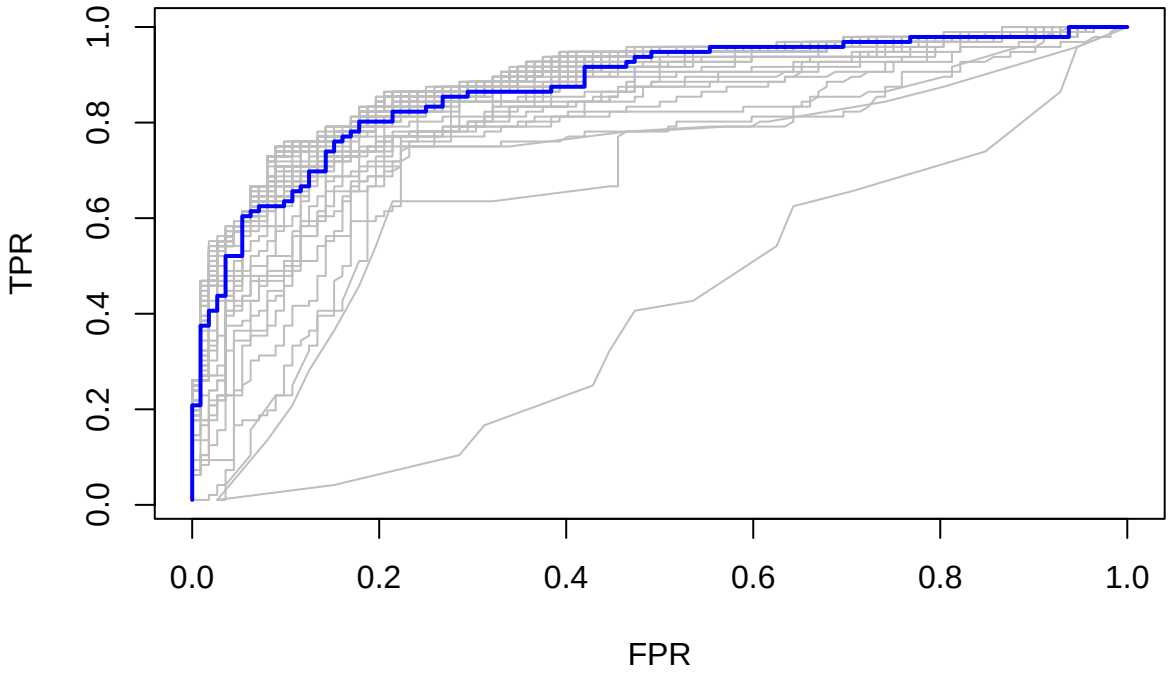
```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                                    s1
## (Intercept)              1.30849402
## age                      .
## sexmale                  0.14907345
## cpatypical angina        .
## cpnon-anginal pain      -0.19945296
## cptypical angina         .
## trestbps                 .
## chol                     .
## fbs>120                  .
## restecgnormal           -0.26903318
## restecgwave abnormality  .
## thalach                 -0.01303082
## exangyes                 0.48929242
## oldpeak                  0.19499207
## slopeflat                0.24290962
## slopeupsloping           .
## ca1                      0.27970612
## ca2                      1.00585401
## ca3                      0.92418748
## thalnormal              -0.72307459
## thalreversable defect    0.51261582
```

```
cv_lasso_auc <- cv.glmnet(x, y, family = "binomial", alpha = 1, lambda = NULL, type.measure = "auc", ke

rocs_l <- roc.glmnet(cv_lasso_auc$fit.preval, newy = y)
```

```
best_l_min <- cv_lasso_auc$index["min",]

plot(rocs_l[[best_l_min]], type = "l")
invisible(sapply(rocs_l, lines, col="grey"))
lines(rocs_l[[best_l_min]], lwd = 2,col = "red")
```




```
best_l_1se <- cv_lasso_auc$index["1se",]
plot(rocs_l[[best_l_1se]], type = "l")
invisible(sapply(rocs_l, lines, col="grey"))
lines(rocs_l[[best_l_1se]], lwd = 2,col = "blue")
```



LASSO Logistic Regression models

```
lasso_model_min <- glmnet(x, y, alpha = 1, family = "binomial",
                          lambda = cv_lasso$lambda.min)

lasso_model_1se <- glmnet(x, y, alpha = 1, family = "binomial",
                          lambda = cv_lasso$lambda.1se)
```

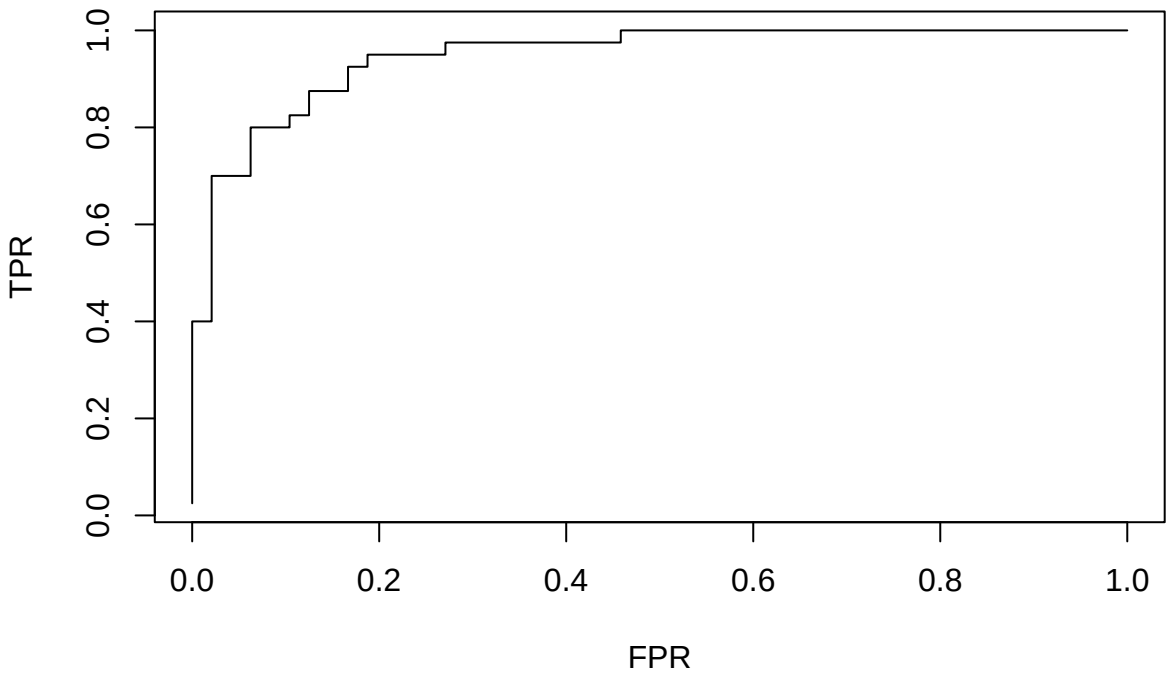
Let's look at the performance of the two penalized models on the test data

```
test_x <- model.matrix(target ~., test_data)[,-1]
test_y <- ifelse(test_data$target == "heart-disease", 1 , 0)


a_min <- assess.glmnet(lasso_model_min, newx = test_x, newy = test_y, s = "lambda.min")
a_1se <- assess.glmnet(lasso_model_1se, newx = test_x, newy = test_y, s = "lambda.1se")

do.call(rbind, Map(data.frame, Lasso_min = a_min, Lasso_1se = a_1se))
```

```
##          Lasso_min Lasso_1se
## deviance 0.6193404 0.8157155
## class    0.1363636 0.1363636
## auc      0.9473958 0.9239583
## mse      0.1916308 0.2451411
## mae      0.4354615 0.6136725
```

Let's closer at the LASSO model with the minimum lambda value

```
roc_lasso_min_t <- roc.glmnet(lasso_model_min, newx = test_x, newy = test_y)
plot(roc_lasso_min_t, type = "l")
```



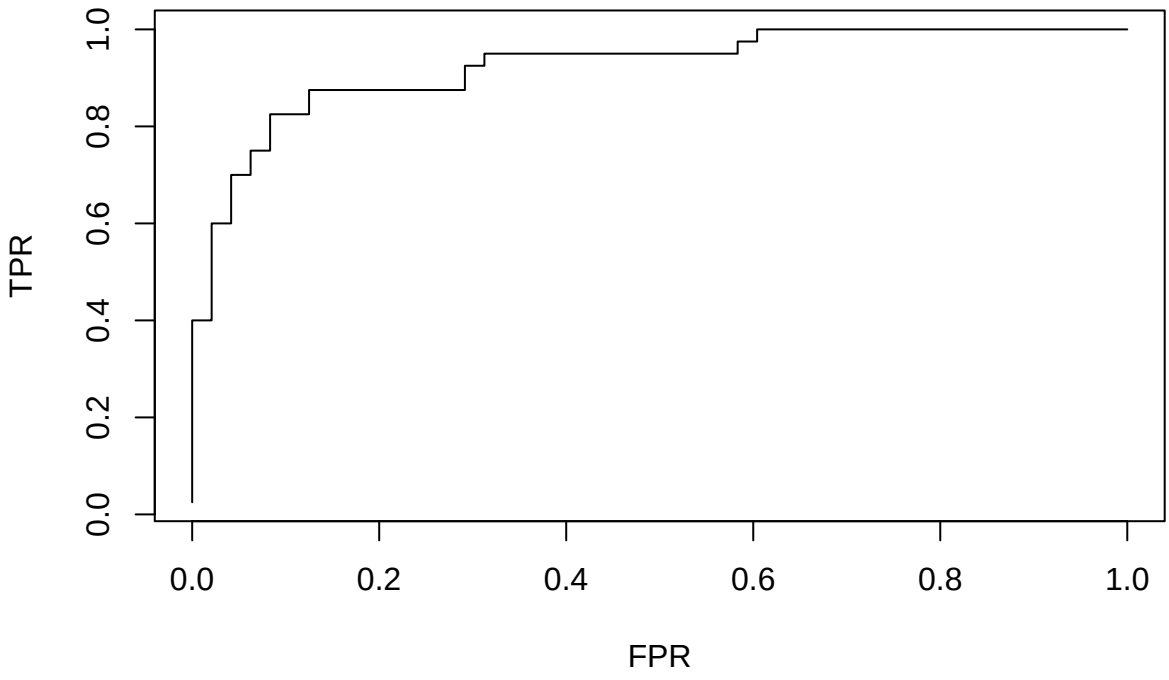

```
min_pred <- predict(lasso_model_min, newx = test_x)

min_pred <- as.factor(ifelse(min_pred > 0.5, 1, 0))
cnf_min <- confusionMatrix(min_pred, as.factor(test_y), positive = "1")
cnf_min
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 45 12
##          1  3 28
```

```
##
##                Accuracy : 0.8295
##                  95% CI : (0.7345, 0.9013)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 1.846e-08
##
##                   Kappa : 0.6497
##
##  Mcnemar's Test P-Value : 0.03887
##
##             Sensitivity : 0.7000
##             Specificity : 0.9375
##          Pos Pred Value : 0.9032
##          Neg Pred Value : 0.7895
##              Prevalence : 0.4545
##          Detection Rate : 0.3182
##    Detection Prevalence : 0.3523
##       Balanced Accuracy : 0.8187
##
##        'Positive' Class : 1
##
```

Let's closer at the LASSO model with the 1 SE lambda value

```
roc_lasso_1se_t <- roc.glmnet(lasso_model_1se, newx = test_x, newy = test_y)
plot(roc_lasso_1se_t, type = "l")
```



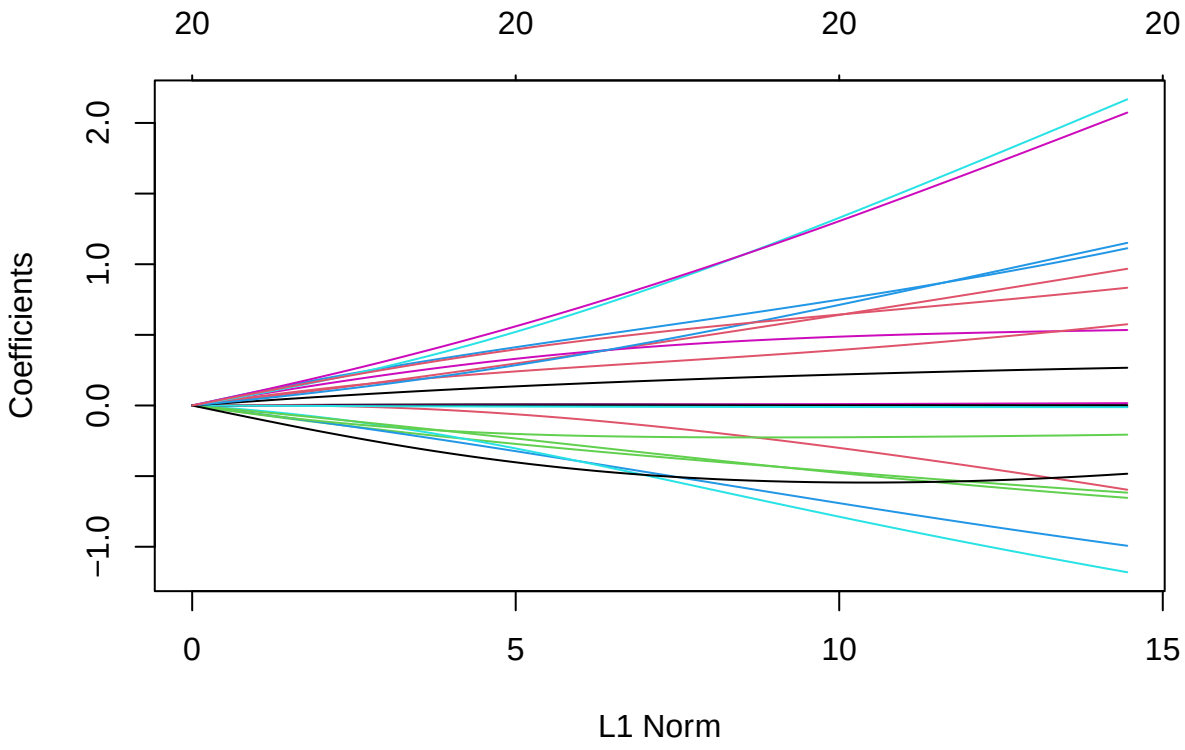

```
onese_pred <- predict(lasso_model_1se, newx = test_x)

onese_pred <- as.factor(ifelse(onese_pred > 0.5, 1, 0))
cnf_1se <- confusionMatrix(onese_pred, as.factor(test_y), positive = "1")
cnf_1se
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction  0  1
##          0 47 16
##          1  1 24
##
##                Accuracy : 0.8068
##                  95% CI : (0.7088, 0.8832)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 2.609e-07
##
##                   Kappa : 0.5978
##
##  Mcnemar's Test P-Value : 0.000685
##
##             Sensitivity : 0.6000
##             Specificity : 0.9792
##          Pos Pred Value : 0.9600
##          Neg Pred Value : 0.7460
##              Prevalence : 0.4545
##          Detection Rate : 0.2727
##    Detection Prevalence : 0.2841
##       Balanced Accuracy : 0.7896
##
##        'Positive' Class : 1
##
```

Ridge

```
log_ridge <- glmnet(x, y, family = "binomial", alpha = 0, lambda = NULL)
plot(log_ridge)
```



ridge model

```
cv_ridge <- cv.glmnet(x, y, family = "binomial", alpha = 0, lambda = NULL, type.measure = "class")
```

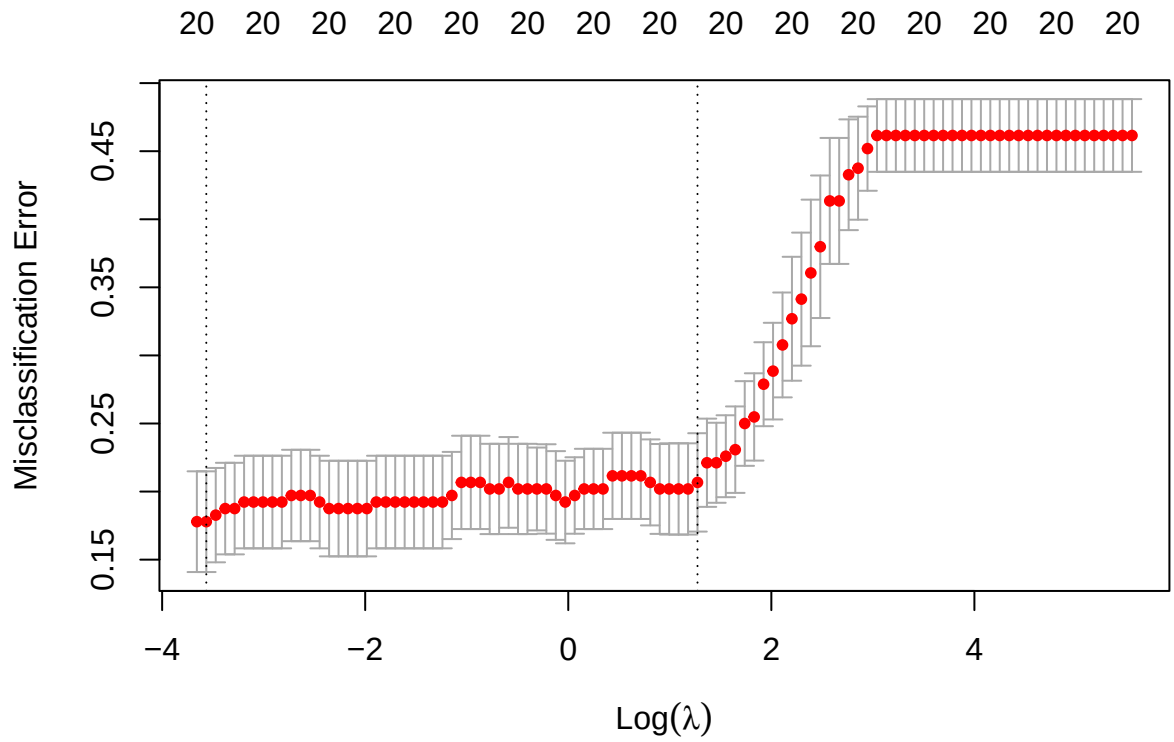
```
coef(cv_ridge, cv_ridge$lambda.min)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                                      s1
## (Intercept)                -1.773723470
## age                        -0.007503914
## sexmale                     0.941404175
## cpatypical angina          -0.606450672
## cpnon-anginal pain         -0.971440122
## cptypical angina           -1.151873834
## trestbps                    0.015942659
## chol                        0.002474698
## fbs>120                    -0.571141159
## restecgnormal              -0.641935748
## restecgwave abnormality     1.080034447
## thalach                    -0.011777615
## exangyes                    0.531698602
## oldpeak                     0.263981390
## slopeflat                   0.558292072
## slopeupsloping             -0.208685477
## ca1                         1.116632225
## ca2                         2.100175353
## ca3                         2.011696463
## thalnormal                 -0.493061142
## thalreversable defect       0.817233243
```

```
coef(cv_ridge, cv_ridge$lambda.1se)
```

```
## 21 x 1 sparse Matrix of class "dgCMatrix"
##                                       s1
## (Intercept)                -0.2541574145
## age                         0.0028807478
## sexmale                     0.0651556505
## cpatypical angina          -0.0758153217
## cpnon-anginal pain         -0.0705491712
## cptypical angina           -0.0447470490
## trestbps                    0.0012143175
## chol                        0.0001930193
## fbs>120                     0.0016634763
## restecgnormal              -0.0528778820
## restecgwave abnormality     0.1173169876
## thalach                    -0.0022213968
## exangyes                    0.0923178649
## oldpeak                     0.0382770545
## slopeflat                   0.0749791921
## slopeupsloping             -0.0721462018
## ca1                         0.0560224356
## ca2                         0.1003043198
## ca3                         0.1205867597
## thalnormal                 -0.1144064508
## thalreversable defect       0.1079244491
```

```
plot(cv_ridge)
```



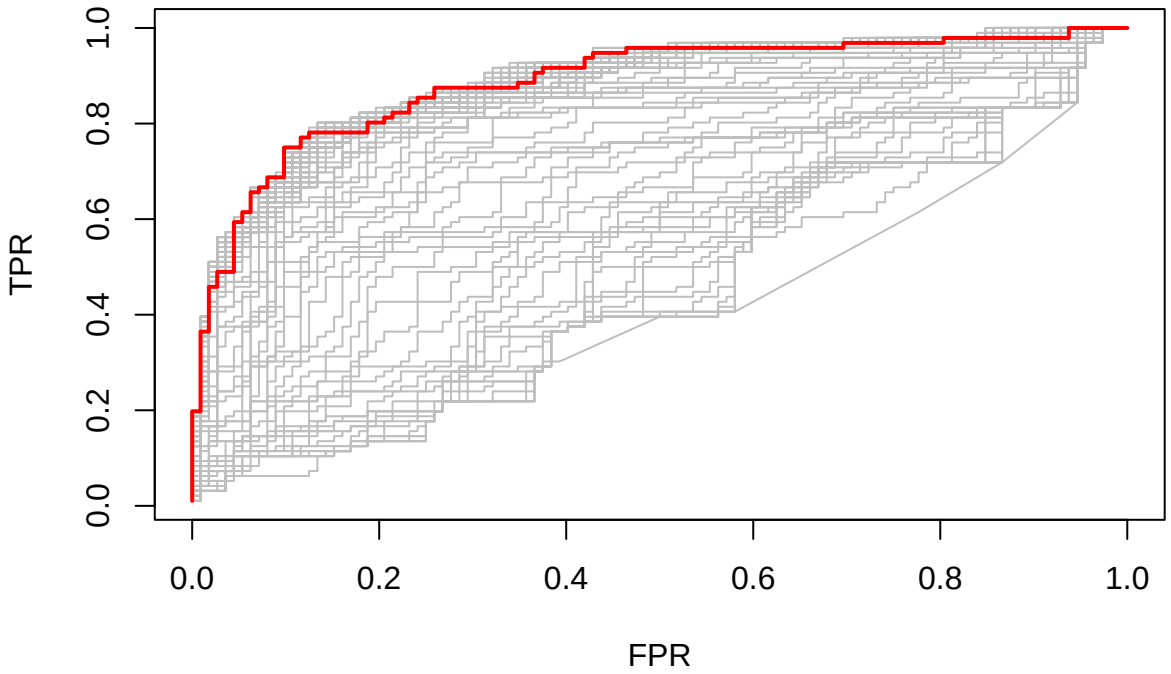

```
ridge_model_min <- glmnet(x, y, alpha = 0, family = "binomial",
                          lambda = cv_ridge$lambda.min)

ridge_model_1se <- glmnet(x, y, alpha = 0, family = "binomial",
                          lambda = cv_ridge$lambda.1se)
```

```
test_x <- model.matrix(target ~., test_data)[,-1]
test_y <- ifelse(test_data$target == "heart-disease", 1 , 0)


ar_min <- assess.glmnet(ridge_model_min, newx = test_x, newy = test_y, s = "lambda.min")
ar_1se <- assess.glmnet(ridge_model_1se, newx = test_x, newy = test_y, s = "lambda.1se")

do.call(rbind, Map(data.frame, Ridge_min = ar_min, Ridge_1se = ar_1se))
```

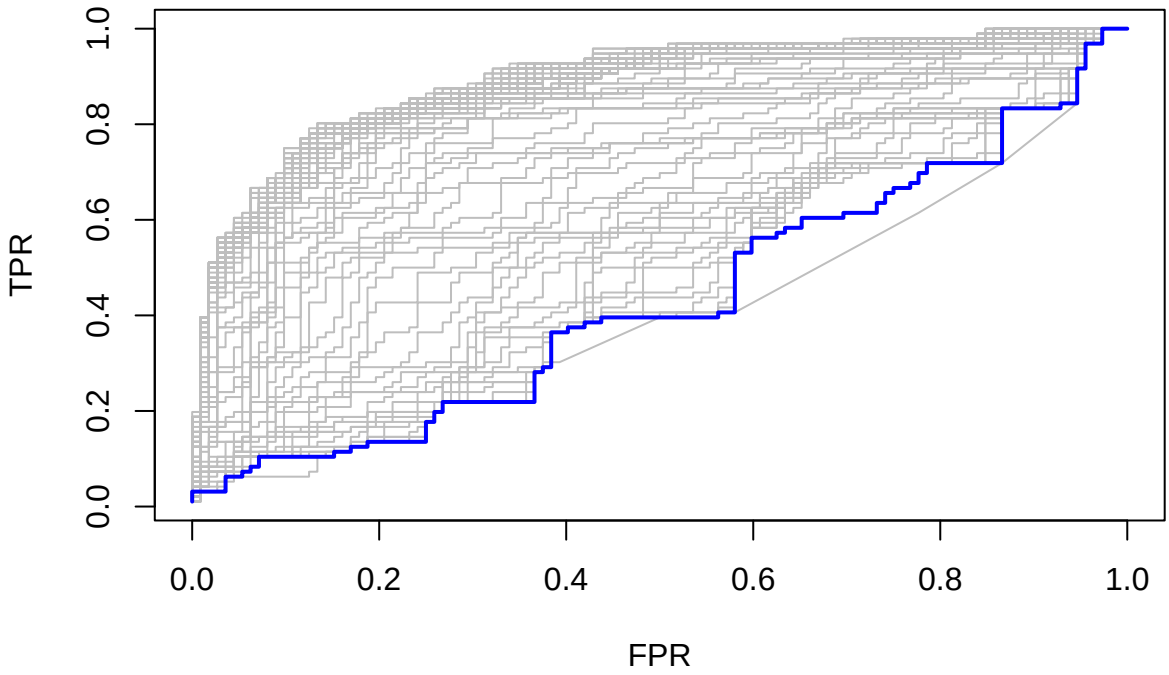
```
##           Ridge_min Ridge_1se
## deviance 0.6258539 1.1930837
## class    0.1250000 0.1818182
## auc      0.9484375 0.9276042
## mse      0.1896013 0.4046045
## mae      0.4601911 0.8929649
```

```
cv_ridge_auc_r <- cv.glmnet(x, y, family = "binomial", alpha = 0, lambda = NULL, type.measure = "auc", l

rocs_r <- roc.glmnet(cv_ridge_auc_r$fit.preval, newy = y)
```

```
best_r_min <- cv_ridge_auc_r$index["min",]
plot(rocs_r[[best_r_min]], type = "l")
invisible(sapply(rocs_r, lines, col="grey"))
lines(rocs_r[[best_r_min]], lwd = 2,col = "red")
```




```
best_r_1se <- cv_ridge_auc_r$index["1se",]
plot(rocs_r[[best_r_1se]], type = "l")
invisible(sapply(rocs_r, lines, col="grey"))
lines(rocs_r[[best_r_1se]], lwd = 2,col = "blue")
```




```
min_pred_r <- predict(ridge_model_min, newx = test_x)

min_pred_r <- as.factor(ifelse(min_pred_r > 0.5, 1, 0))
cnf_min_r <- confusionMatrix(min_pred_r, as.factor(test_y), positive = "1")
cnf_min_r
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 46 10
##          1  2 30
##
##                Accuracy : 0.8636
##                  95% CI : (0.7739, 0.9275)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 1.927e-10
##
##                   Kappa : 0.7203
##
##  Mcnemar's Test P-Value : 0.04331
##
##             Sensitivity : 0.7500
##             Specificity : 0.9583
##          Pos Pred Value : 0.9375
##          Neg Pred Value : 0.8214
##              Prevalence : 0.4545
##          Detection Rate : 0.3409
##    Detection Prevalence : 0.3636
##       Balanced Accuracy : 0.8542
##
##        'Positive' Class : 1
##
```

```r
onese_pred_r <- predict(ridge_model_1se, newx = test_x)

onese_pred_r <- as.factor(ifelse(onese_pred_r > 0.5, 1, 0))
cnf_1se_r <- confusionMatrix(onese_pred_r, as.factor(test_y), positive = "1")
```

```
## Warning in confusionMatrix.default(onese_pred_r, as.factor(test_y), positive =
## "1"): Levels are not in the same order for reference and data. Refactoring data
## to match.
```

```r
cnf_1se_r
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 48 40
##          1  0  0
##
##                Accuracy : 0.5455
##                  95% CI : (0.4358, 0.652)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 0.5439
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 6.984e-10
##
##             Sensitivity : 0.0000
##             Specificity : 1.0000
##          Pos Pred Value :    NaN
##          Neg Pred Value : 0.5455
##              Prevalence : 0.4545
##          Detection Rate : 0.0000
##    Detection Prevalence : 0.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 1
##
```

```r
do.call(rbind, Map(data.frame, Lasso_min = cnf_min$byClass, Lasso_1se = cnf_1se$byClass, Ridge_min = cn
```

```
##                     Lasso_min Lasso_1se Ridge_min Ridge_1se
## Sensitivity         0.7000000 0.6000000 0.7500000 0.0000000
## Specificity         0.9375000 0.9791667 0.9583333 1.0000000
## Pos Pred Value      0.9032258 0.9600000 0.9375000       NaN
## Neg Pred Value      0.7894737 0.7460317 0.8214286 0.5454545
## Precision           0.9032258 0.9600000 0.9375000        NA
## Recall              0.7000000 0.6000000 0.7500000 0.0000000
## F1                  0.7887324 0.7384615 0.8333333        NA
## Prevalence          0.4545455 0.4545455 0.4545455 0.4545455
## Detection Rate      0.3181818 0.2727273 0.3409091 0.0000000
## Detection Prevalence 0.3522727 0.2840909 0.3636364 0.0000000
## Balanced Accuracy   0.8187500 0.7895833 0.8541667 0.5000000
```

Random Forest

```
train_x_rf <- train_data %>% select(-target)
train_y_rf <- train_data %>% select(target) %>% mutate(target = as.factor(target))

test_x_rf <- test_data %>% select(-target)
test_y_rf <- test_data %>% select(target) %>% mutate(target = as.factor(target))
```

```
rf <- randomForest(target ~ ., data = train_data, importance = TRUE)
rf
```

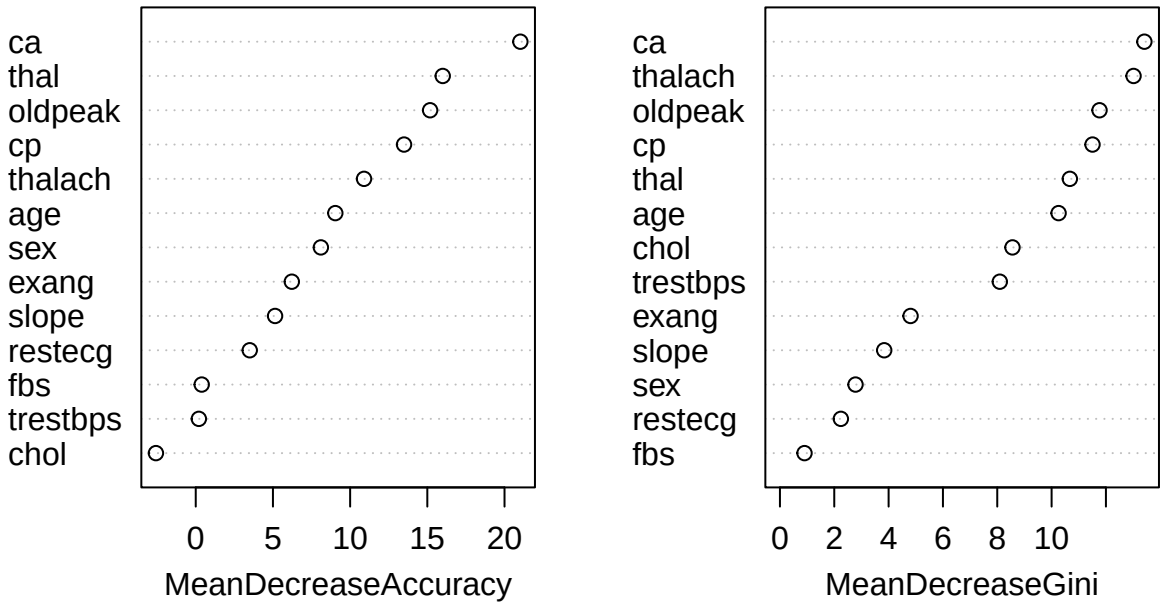
```
##
## Call:
##  randomForest(formula = target ~ ., data = train_data, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 18.75%
## Confusion matrix:
##               asymptomatic heart-disease class.error
## asymptomatic            96            16   0.1428571
## heart-disease           23            73   0.2395833
```

```
importance(rf)
```

```
##          asymptomatic heart-disease MeanDecreaseAccuracy MeanDecreaseGini
## age         8.4382139      4.188419            9.0394499       10.2574634
## sex         7.2871264      3.388281            8.1037737        2.7826813
## cp          7.8694953     12.235814           13.4916579       11.5060365
## trestbps    2.3748528     -2.107120            0.2005787        8.0934816
## chol       -0.8900332     -2.648165           -2.5802241        8.5597141
## fbs         2.6124259     -2.031281            0.3861614        0.9045091
## restecg     2.3962699      2.358433            3.4975729        2.2427662
## thalach    10.3933974      4.615066           10.9000297       13.0164611
## exang       2.6689493      6.094123            6.2222162        4.8078831
## oldpeak    10.9097950      9.928201           15.1799604       11.7638986
## slope       1.4673787      5.278861            5.1372768        3.8383419
## ca         17.7515644     13.940413           21.0265948       13.4154733
## thal       12.4366989     11.964788           15.9980899       10.6717290
```

```
varImpPlot(rf)
```

rf



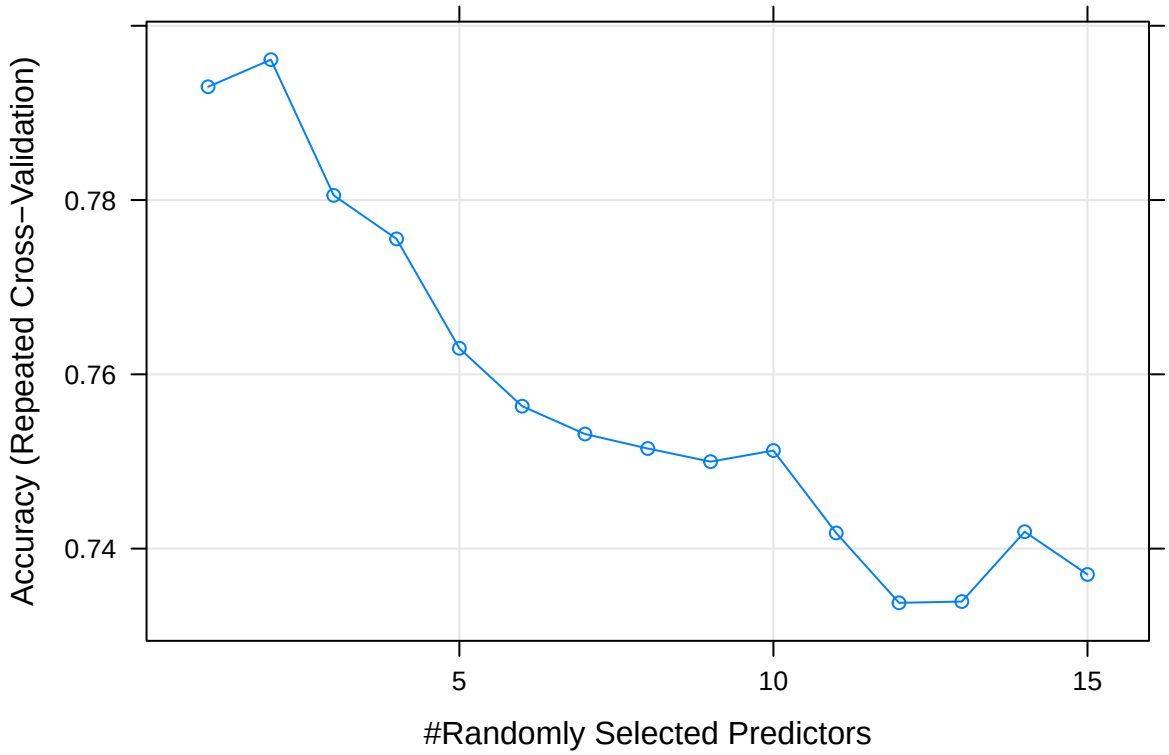
model tuning

```
control_rf <- trainControl(method = "repeatedcv", number = 10, repeats = 3, search = "grid")

tune_grid <- expand.grid(mtry = (1:15))

rf_grid_search <- train(target ~ .,
                       data = train_data,
                       method = "rf",
                       metric = "Accuracy",
                       trControl = control_rf,
                       tuneGrid = tune_grid)

rf_grid_search
```

```
## Random Forest
##
## 208 samples
##  13 predictor
##   2 classes: 'asymptomatic', 'heart-disease'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 187, 188, 187, 188, 187, 186, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##    1    0.7930014  0.5773407
##    2    0.7961111  0.5865549
```

```
##     3    0.7805339  0.5551043
##     4    0.7755411  0.5467043
##     5    0.7629870  0.5223165
##     6    0.7563420  0.5081888
##     7    0.7531530  0.5018553
##     8    0.7514863  0.4983790
##     9    0.7499784  0.4955033
##    10    0.7512554  0.4988605
##    11    0.7417965  0.4792026
##    12    0.7337807  0.4636136
##    13    0.7339250  0.4638571
##    14    0.7419408  0.4798957
##    15    0.7370346  0.4700346
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
rf_best <- rf_grid_search
plot(rf_best)
```




```
cnf_rf <- confusionMatrix(predict(rf_best, test_x_rf), test_y_rf[,1], positive = "heart-disease")
```

```
ds_rf <- defaultSummary(data.frame(obs = test_y_rf[,1],
                                   pred = predict(rf_best, test_x_rf)))

ds_rf
```

```
##  Accuracy     Kappa
## 0.8295455 0.6540881
```

Support Vector Machine

```
hd_model_svm <- na.omit(hd_new)

missing_ca_indeces <- which(hd_model_svm$ca %in% 4)
missing_thal_indeces <-which(hd_model_svm$thal %in% 0)
restecg_indices <-which(hd_model_svm$restecg %in% 2)
missing_values_indeces <- c(missing_ca_indeces, missing_thal_indeces, restecg_indices)
hd_model_svm <- hd_model_svm[-missing_values_indeces, ]

hd_model_svm <- hd_model_svm %>% mutate(target = as.integer(target))
set.seed(44)
train.samples_svm <- hd_model_svm$target %>%
  createDataPartition(p = 0.7, list = FALSE)

train_data_svm <- hd_model_svm[train.samples_svm, ]
test_data_svm <- hd_model_svm[-train.samples_svm, ]
```

Let's scale the numeric variables and 1-hot encode the categorical ones

```
train_data_svm <- train_data_svm %>% mutate(cp = as.factor(cp), restecg = as.factor(restecg),
                                            exang = as.factor(exang), slope = as.factor(slope),
                                            ca = as.factor(ca), thal = as.factor(thal))

test_data_svm <- test_data_svm %>% mutate(cp = as.factor(cp), restecg = as.factor(restecg),
                                            exang = as.factor(exang), slope = as.factor(slope),
                                            ca = as.factor(ca), thal = as.factor(thal))


train_data_svm[, c(1,4,5,8,10)] = scale(train_data_svm[, c(1,4,5,8,10)])
test_data_svm[, c(1,4,5,8,10)] = scale(test_data_svm[, c(1,4,5,8,10)])

dummy_train <- dummyVars(" ~ .", data = train_data_svm)
dummy_test <- dummyVars(" ~ .", data = test_data_svm)

train_data_svm <- data.frame(predict(dummy_train, newdata = train_data_svm))
test_data_svm <- data.frame(predict(dummy_test, newdata = test_data_svm))

train_data_svm$target <- as.factor(train_data_svm$target)
test_data_svm$target <- as.factor(test_data_svm$target)
```

```
train_control_svm <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

svm_lin <- train(target ~ ., data = train_data_svm, method = "svmLinear",
                 trControl = train_control_svm, tuneLength = 10)

svm_lin
```

```
## Support Vector Machines with Linear Kernel
##
## 205 samples
##  25 predictor
##   2 classes: '0', '1'
##
## No pre-processing
```

```
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 185, 184, 185, 184, 185, 185, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8598846  0.7155427
##
## Tuning parameter 'C' was held constant at a value of 1
```

```
pred_svm <- predict(svm_lin, train_data_svm)
confusionMatrix(table(pred_svm, train_data_svm$target), positive = "1")
```

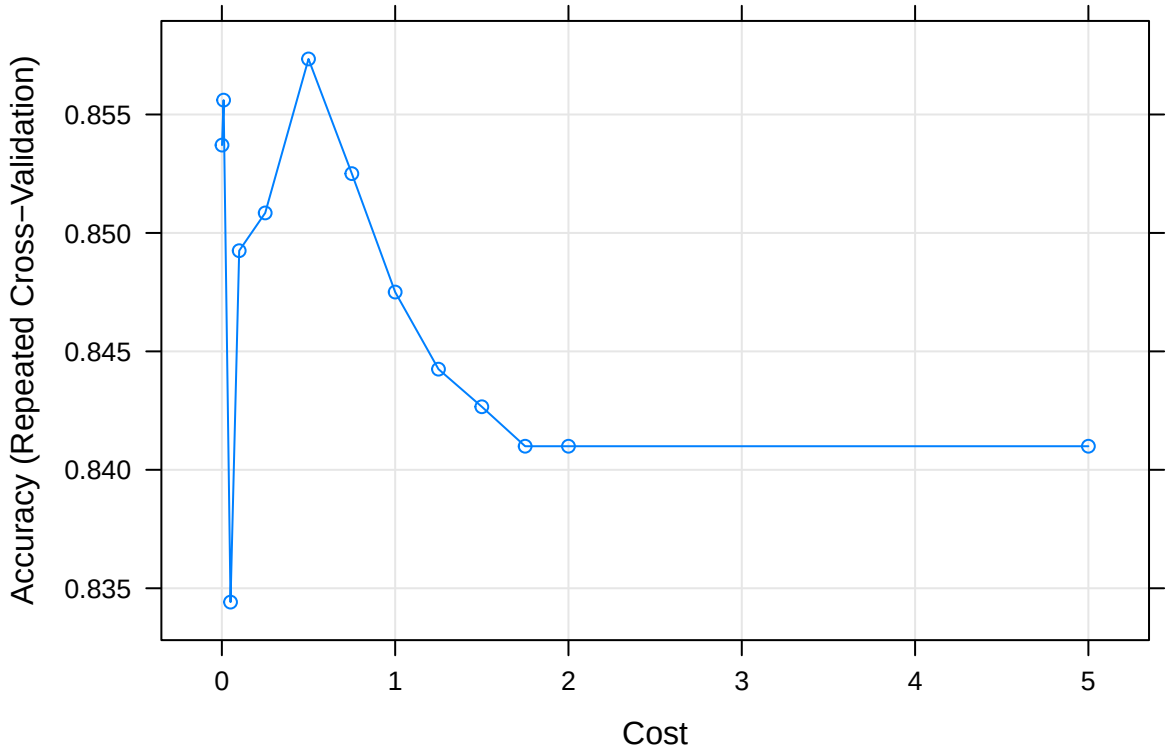
```
## Confusion Matrix and Statistics
##
##
## pred_svm   0   1
##        0 101  12
##        1  10  82
##
##                Accuracy : 0.8927
##                  95% CI : (0.842, 0.9315)
##     No Information Rate : 0.5415
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.7835
##
##  Mcnemar's Test P-Value : 0.8312
##
##             Sensitivity : 0.8723
##             Specificity : 0.9099
##          Pos Pred Value : 0.8913
##          Neg Pred Value : 0.8938
##              Prevalence : 0.4585
##          Detection Rate : 0.4000
##    Detection Prevalence : 0.4488
##       Balanced Accuracy : 0.8911
##
##        'Positive' Class : 1
##
```

```
grid_lin <- expand.grid(C = c(0.001, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2,5))

svm_lin_grid<- train(target ~., data = train_data_svm, method = "svmLinear", trControl = train_control_s
                     tuneGrid = grid_lin, tuneLength = 10)
svm_lin_grid
```

```
## Support Vector Machines with Linear Kernel
##
## 205 samples
##  25 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 185, 185, 185, 184, 183, 185, ...
```

```
## Resampling results across tuning parameters:
##
##   C      Accuracy   Kappa
##   0.001  0.8537085  0.6994294
##   0.010  0.8556061  0.7067418
##   0.050  0.8344156  0.6639182
##   0.100  0.8492496  0.6938446
##   0.250  0.8508442  0.6975144
##   0.500  0.8573449  0.7103666
##   0.750  0.8525036  0.7007673
##   1.000  0.8475036  0.6904208
##   1.250  0.8442496  0.6841509
##   1.500  0.8426623  0.6810132
##   1.750  0.8409957  0.6777278
##   2.000  0.8409957  0.6777278
##   5.000  0.8409957  0.6780735
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 0.5.
```

```
plot(svm_lin_grid)
```




```
pred_svm_lin_grid <- predict(svm_lin_grid, newdata = test_data_svm)

cnf_svm_lin <- confusionMatrix(pred_svm_lin_grid, test_data_svm$target, positive = "1")
cnf_svm_lin
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
```

```
##          0 39 12
##          1  9 27
##
##                Accuracy : 0.7586
##                  95% CI : (0.655, 0.844)
##     No Information Rate : 0.5517
##     P-Value [Acc > NIR] : 5.225e-05
##
##                   Kappa : 0.5085
##
##  Mcnemar's Test P-Value : 0.6625
##
##             Sensitivity : 0.6923
##             Specificity : 0.8125
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.7647
##              Prevalence : 0.4483
##          Detection Rate : 0.3103
##    Detection Prevalence : 0.4138
##       Balanced Accuracy : 0.7524
##
##        'Positive' Class : 1
##
```

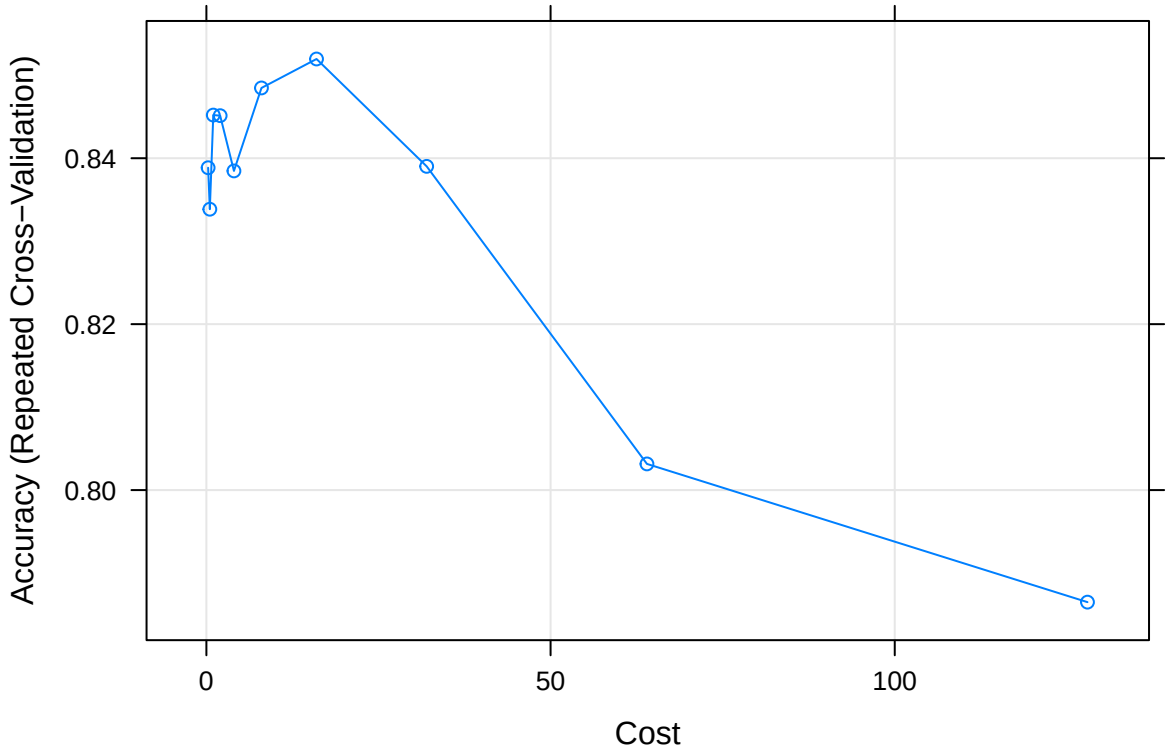
Radial

```
svm_rad <- train(target ~ ., data = train_data_svm, method = "svmRadial", scale = FALSE,
                 trControl = train_control_svm, tuneLength = 10)

svm_rad
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 205 samples
##  25 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 185, 184, 185, 184, 184, 184, ...
## Resampling results across tuning parameters:
##
##   C       Accuracy   Kappa
##     0.25  0.8388528  0.6718226
##     0.50  0.8338528  0.6625122
##     1.00  0.8452020  0.6853632
##     2.00  0.8451299  0.6855812
##     4.00  0.8384704  0.6722077
##     8.00  0.8484704  0.6923283
##    16.00  0.8519625  0.7004888
##    32.00  0.8390188  0.6755005
##    64.00  0.8031530  0.6022024
##   128.00  0.7864935  0.5691614
##
## Tuning parameter 'sigma' was held constant at a value of 0.02562191
```

```
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.02562191 and C = 16.
```

```
plot(svm_rad)
```




```
pred_rad <- predict(svm_rad, train_data_svm)

confusionMatrix(table(pred_rad, train_data_svm$target))
```

```
## Confusion Matrix and Statistics
##
##
## pred_rad   0   1
##        0 108   7
##        1   3  87
##
##                Accuracy : 0.9512
##                  95% CI : (0.9121, 0.9764)
##     No Information Rate : 0.5415
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9014
##
##  Mcnemar's Test P-Value : 0.3428
##
##             Sensitivity : 0.9730
##             Specificity : 0.9255
##          Pos Pred Value : 0.9391
##          Neg Pred Value : 0.9667
##              Prevalence : 0.5415
##          Detection Rate : 0.5268
```

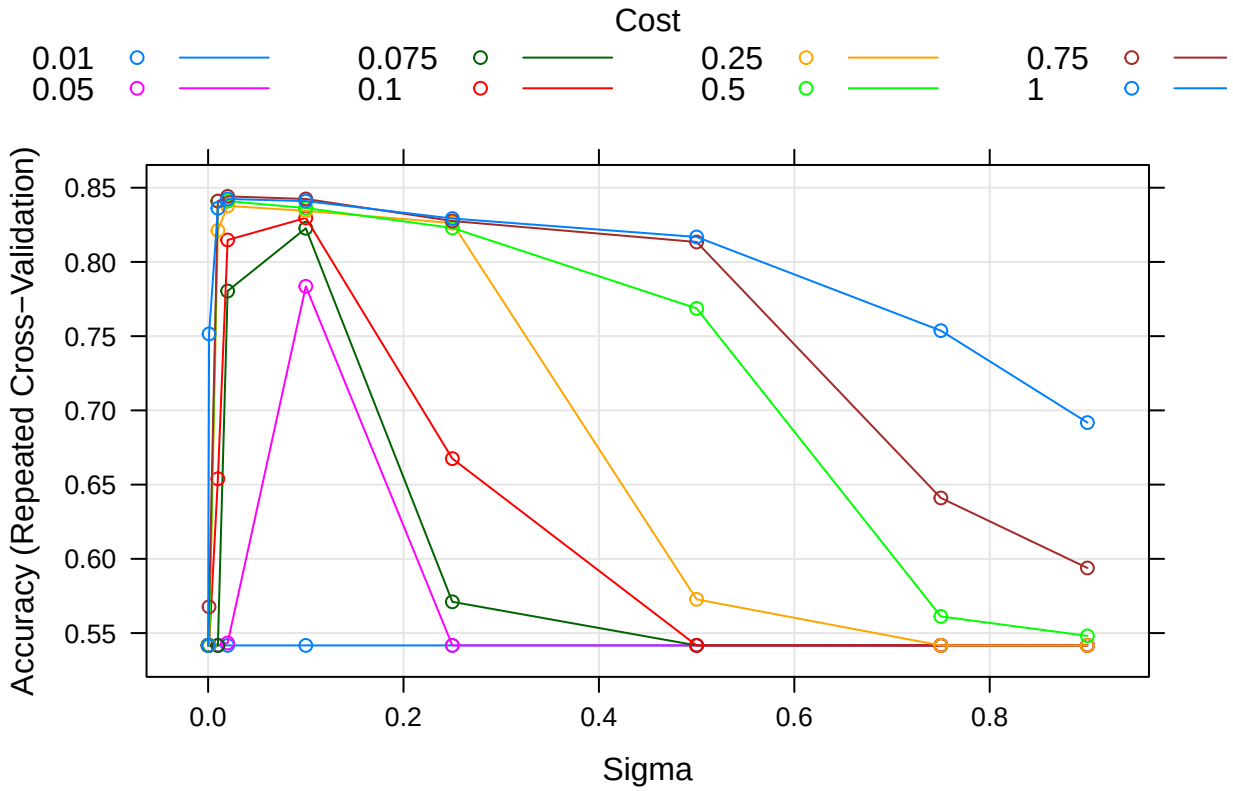
```
##    Detection Prevalence : 0.5610
##       Balanced Accuracy : 0.9493
##
##        'Positive' Class : 0
##
```

```
grid_rad <- expand.grid(sigma = c(0, 0.001, 0.01, 0.02, 0.1, 0.25, 0.5, 0.75,0.9),
 C = c(0.01, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1))

svm_rad_grid<- train(target ~., data = train_data_svm, method = "svmRadial", scale = FALSE, trControl =
                     tuneGrid = grid_rad, tuneLength = 10)
svm_rad_grid
```

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 205 samples
##  25 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 185, 185, 184, 184, 185, 185, ...
## Resampling results across tuning parameters:
##
##   sigma  C      Accuracy   Kappa
##   0.000  0.010  0.5416811  0.000000000
##   0.000  0.050  0.5416811  0.000000000
##   0.000  0.075  0.5416811  0.000000000
##   0.000  0.100  0.5416811  0.000000000
##   0.000  0.250  0.5416811  0.000000000
##   0.000  0.500  0.5416811  0.000000000
##   0.000  0.750  0.5416811  0.000000000
##   0.000  1.000  0.5416811  0.000000000
##   0.001  0.010  0.5416811  0.000000000
##   0.001  0.050  0.5416811  0.000000000
##   0.001  0.075  0.5416811  0.000000000
##   0.001  0.100  0.5416811  0.000000000
##   0.001  0.250  0.5416811  0.000000000
##   0.001  0.500  0.5416811  0.000000000
##   0.001  0.750  0.5677128  0.059918158
##   0.001  1.000  0.7515007  0.478119597
##   0.010  0.010  0.5416811  0.000000000
##   0.010  0.050  0.5416811  0.000000000
##   0.010  0.075  0.5416811  0.000000000
##   0.010  0.100  0.6539250  0.260034436
##   0.010  0.250  0.8212193  0.633967868
##   0.010  0.500  0.8409885  0.674777888
##   0.010  0.750  0.8409885  0.675047288
##   0.010  1.000  0.8361400  0.665896856
##   0.020  0.010  0.5416811  0.000000000
##   0.020  0.050  0.5433478  0.004029304
##   0.020  0.075  0.7804185  0.541867301
##   0.020  0.100  0.8147835  0.621026391
##   0.020  0.250  0.8377345  0.668910254
##   0.020  0.500  0.8409885  0.675744472
```

```
##   0.020  0.750  0.8441631  0.683172576
##   0.020  1.000  0.8424170  0.679879128
##   0.100  0.010  0.5416811  0.000000000
##   0.100  0.050  0.7835931  0.548160839
##   0.100  0.075  0.8226551  0.636875076
##   0.100  0.100  0.8296320  0.652803604
##   0.100  0.250  0.8344012  0.663936991
##   0.100  0.500  0.8362987  0.667358769
##   0.100  0.750  0.8424170  0.679800358
##   0.100  1.000  0.8409885  0.676759606
##   0.250  0.010  0.5416811  0.000000000
##   0.250  0.050  0.5416811  0.000000000
##   0.250  0.075  0.5710534  0.070033130
##   0.250  0.100  0.6675036  0.292617851
##   0.250  0.250  0.8261400  0.648490461
##   0.250  0.500  0.8228860  0.642296862
##   0.250  0.750  0.8275830  0.651589431
##   0.250  1.000  0.8292496  0.654882408
##   0.500  0.010  0.5416811  0.000000000
##   0.500  0.050  0.5416811  0.000000000
##   0.500  0.075  0.5416811  0.000000000
##   0.500  0.100  0.5416811  0.000000000
##   0.500  0.250  0.5727201  0.073974840
##   0.500  0.500  0.7687229  0.520712240
##   0.500  0.750  0.8134271  0.625074285
##   0.500  1.000  0.8168398  0.631663484
##   0.750  0.010  0.5416811  0.000000000
##   0.750  0.050  0.5416811  0.000000000
##   0.750  0.075  0.5416811  0.000000000
##   0.750  0.100  0.5416811  0.000000000
##   0.750  0.250  0.5416811  0.000000000
##   0.750  0.500  0.5612121  0.046877293
##   0.750  0.750  0.6410029  0.231997887
##   0.750  1.000  0.7537229  0.492280834
##   0.900  0.010  0.5416811  0.000000000
##   0.900  0.050  0.5416811  0.000000000
##   0.900  0.075  0.5416811  0.000000000
##   0.900  0.100  0.5416811  0.000000000
##   0.900  0.250  0.5416811  0.000000000
##   0.900  0.500  0.5481169  0.015137725
##   0.900  0.750  0.5938384  0.122390922
##   0.900  1.000  0.6917965  0.349278553
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.02 and C = 0.75.
```

```
plot(svm_rad_grid)
```




```
pred_svm_rad_grid <- predict(svm_rad_grid, newdata = test_data_svm)

cnf_svm_rad <- confusionMatrix(pred_svm_rad_grid, test_data_svm$target, positive = "1")
cnf_svm_rad
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 41 12
##          1  7 27
##
##                Accuracy : 0.7816
##                  95% CI : (0.6802, 0.8631)
##     No Information Rate : 0.5517
##     P-Value [Acc > NIR] : 6.819e-06
##
##                   Kappa : 0.5531
##
##  Mcnemar's Test P-Value : 0.3588
##
##             Sensitivity : 0.6923
##             Specificity : 0.8542
##          Pos Pred Value : 0.7941
##          Neg Pred Value : 0.7736
##              Prevalence : 0.4483
##          Detection Rate : 0.3103
##    Detection Prevalence : 0.3908
##       Balanced Accuracy : 0.7732
```

```
##
##        'Positive' Class : 1
##
```

PCA then SVM

```
hd_model1 <- hd_model %>% mutate(cp = as.factor(cp), restecg = as.factor(restecg),
                                                 exang = as.factor(exang), slope = as.factor(slope),
                                                 ca = as.factor(ca), thal = as.factor(thal), fbs = as.factor
                                                 sex = as.factor(sex))

hd_model2 <- hd_model1 %>% select(-c(target))
```

Using PCA of Mixed Data

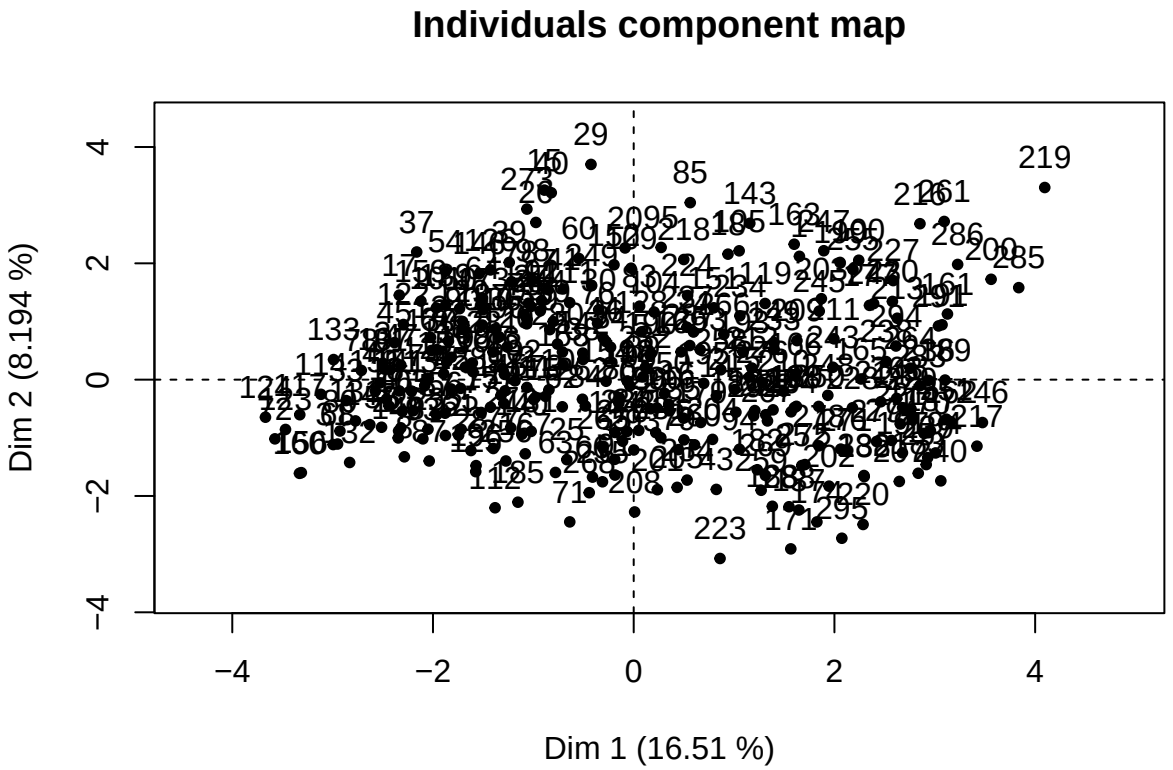
```
X.quanti <- splitmix(hd_model2)$X.quanti
```

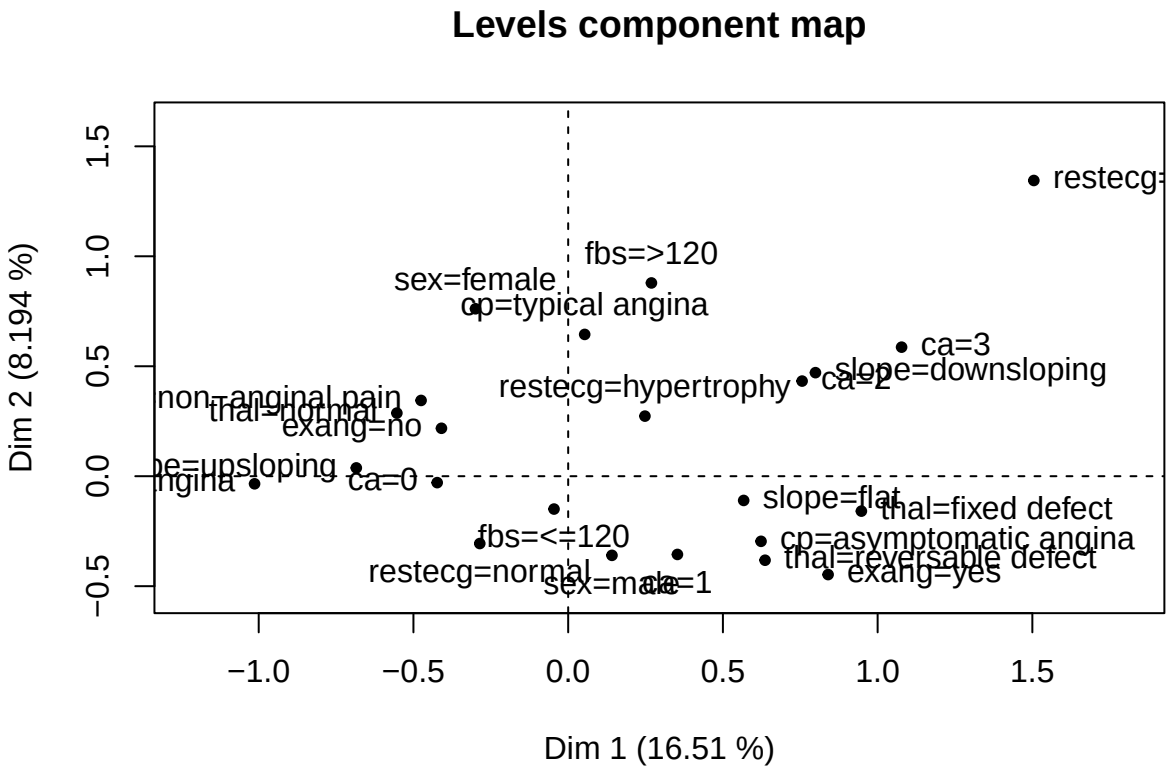
```
## Warning in splitmix(hd_model2): Columns of class integer are considered as
## quantitative
```

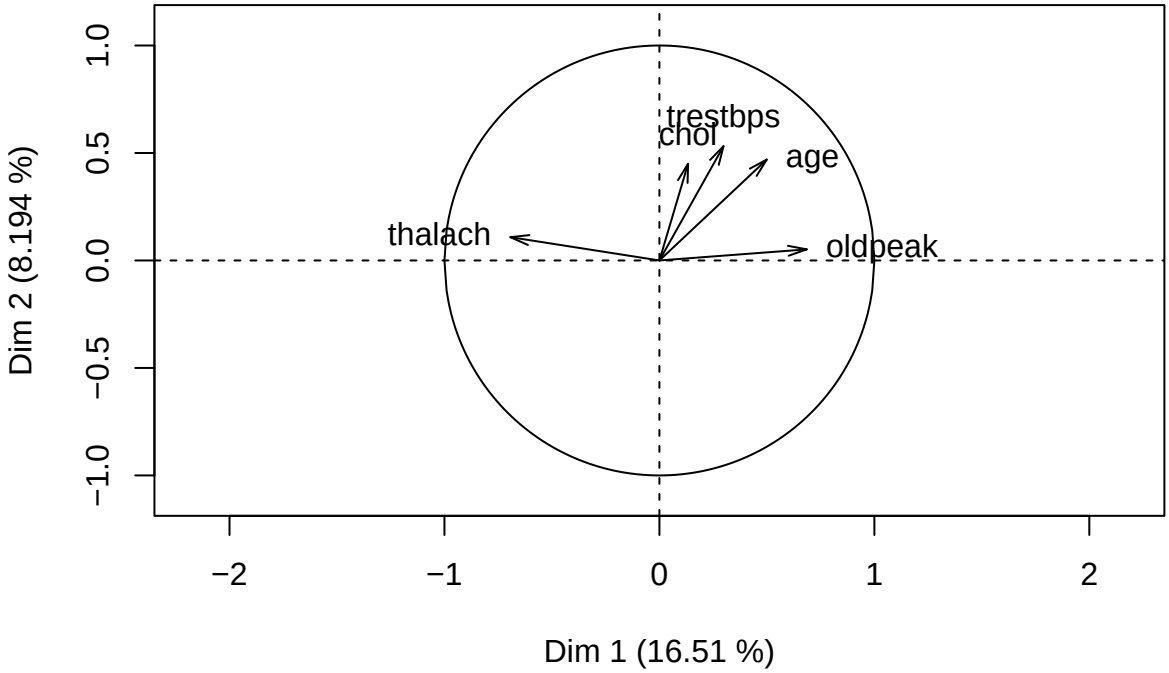
```
X.quali <- splitmix(hd_model2)$X.quali
```

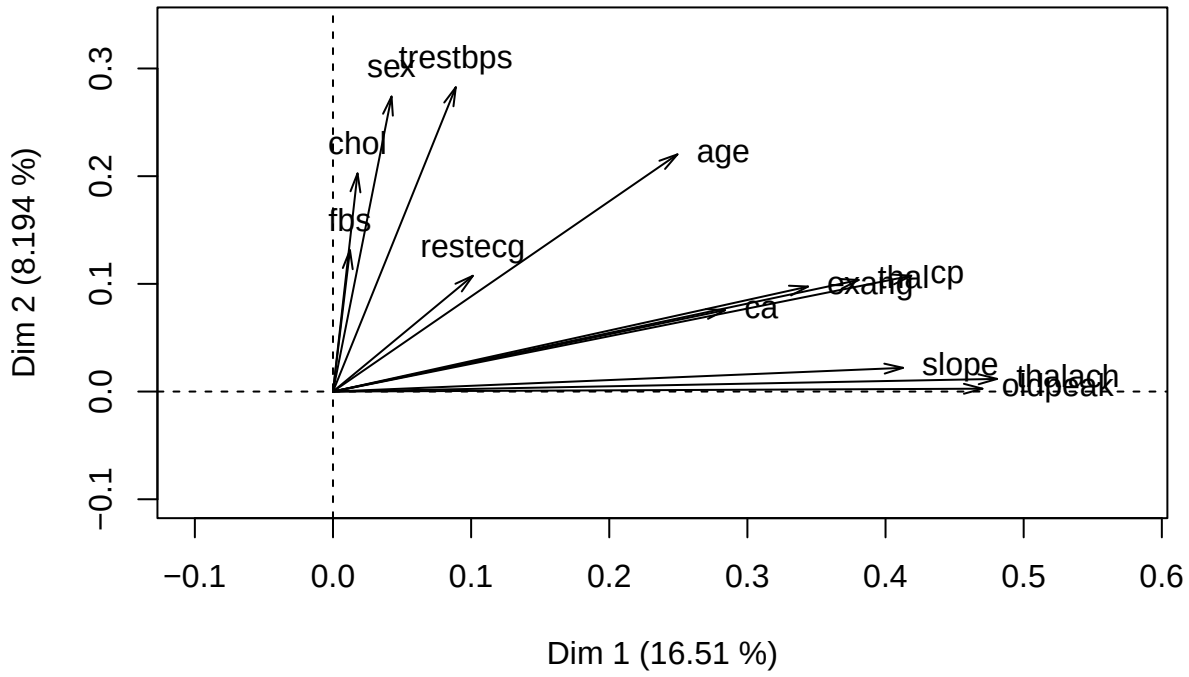
```
## Warning in splitmix(hd_model2): Columns of class integer are considered as
## quantitative
```

```
pca <- PCAmix(X.quanti, X.quali, ndim = 6, rename.level = TRUE)
```




Levels component map
Dim 2 (8.194 %)
Dim 1 (16.51 %)
restecg=
fbs=>120
sex=female
cp=typical angina
ca=3
slope=downsloping
ca=2
restecg=hypertrophy
non−anginal pain
thal=normal
exang=no
ca=0
slope=flat
thal=fixed defect
fbs=<=120
cp=asymptomatic angina
thal=reversable defect
exang=yes
restecg=normal
sex=male
ca=1


Correlation circle
Dim 2 (8.194 %)
Dim 1 (16.51 %)
trestbps
chol
age
thalach
oldpeak



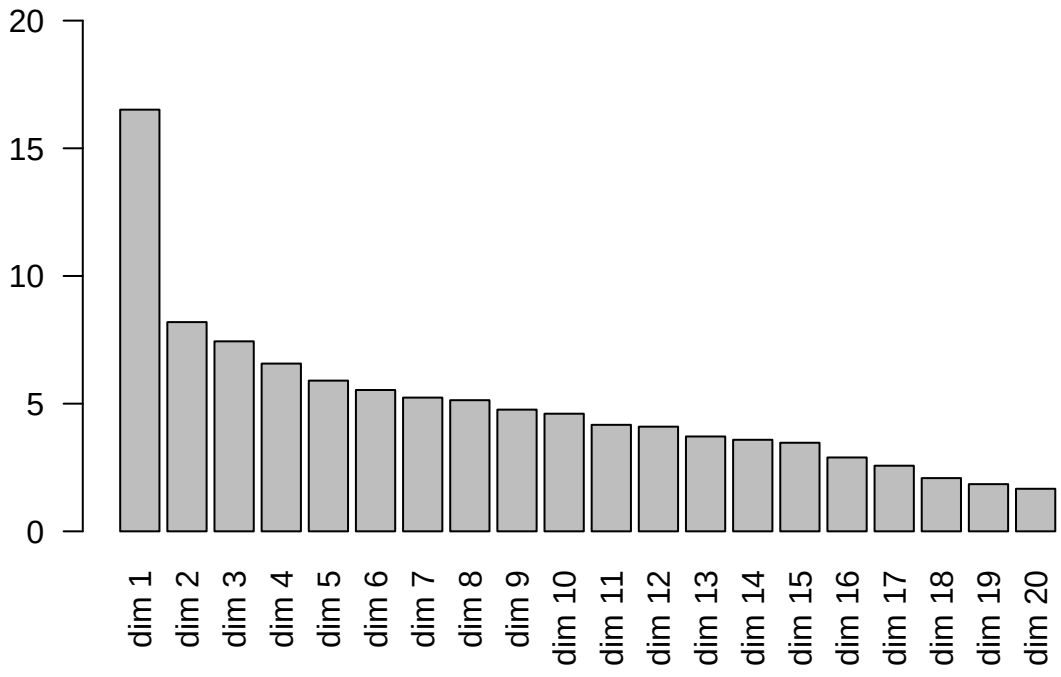

```
summary(pca)
```

```
##
## Call:
## PCAmix(X.quanti = X.quanti, X.quali = X.quali, ndim = 6, rename.level = TRUE)
##
## Method = Factor Analysis of mixed data (FAmix)
##
## Data:
##    number of observations:  296
##    number of  variables:  13
##         number of numerical variables:  5
##         number of categorical variables:  8
##
## Squared loadings :
##          dim 1 dim 2 dim 3 dim 4 dim 5 dim 6
## age       0.25  0.22  0.05  0.03  0.03  0.03
## trestbps  0.09  0.28  0.06  0.02  0.01  0.00
## chol      0.02  0.20  0.15  0.02  0.07  0.09
## thalach   0.48  0.01  0.06  0.02  0.04  0.03
## oldpeak   0.47  0.00  0.07  0.09  0.02  0.02
## sex       0.04  0.27  0.18  0.11  0.04  0.02
## cp        0.42  0.11  0.24  0.05  0.34  0.11
## fbs       0.01  0.13  0.10  0.11  0.14  0.00
## restecg   0.10  0.11  0.01  0.37  0.01  0.03
## exang     0.34  0.10  0.02  0.00  0.04  0.00
## slope     0.41  0.02  0.32  0.19  0.00  0.06
## ca        0.28  0.08  0.12  0.22  0.39  0.23
## thal      0.38  0.10  0.10  0.08  0.06  0.47
```

```
pca$eig
```

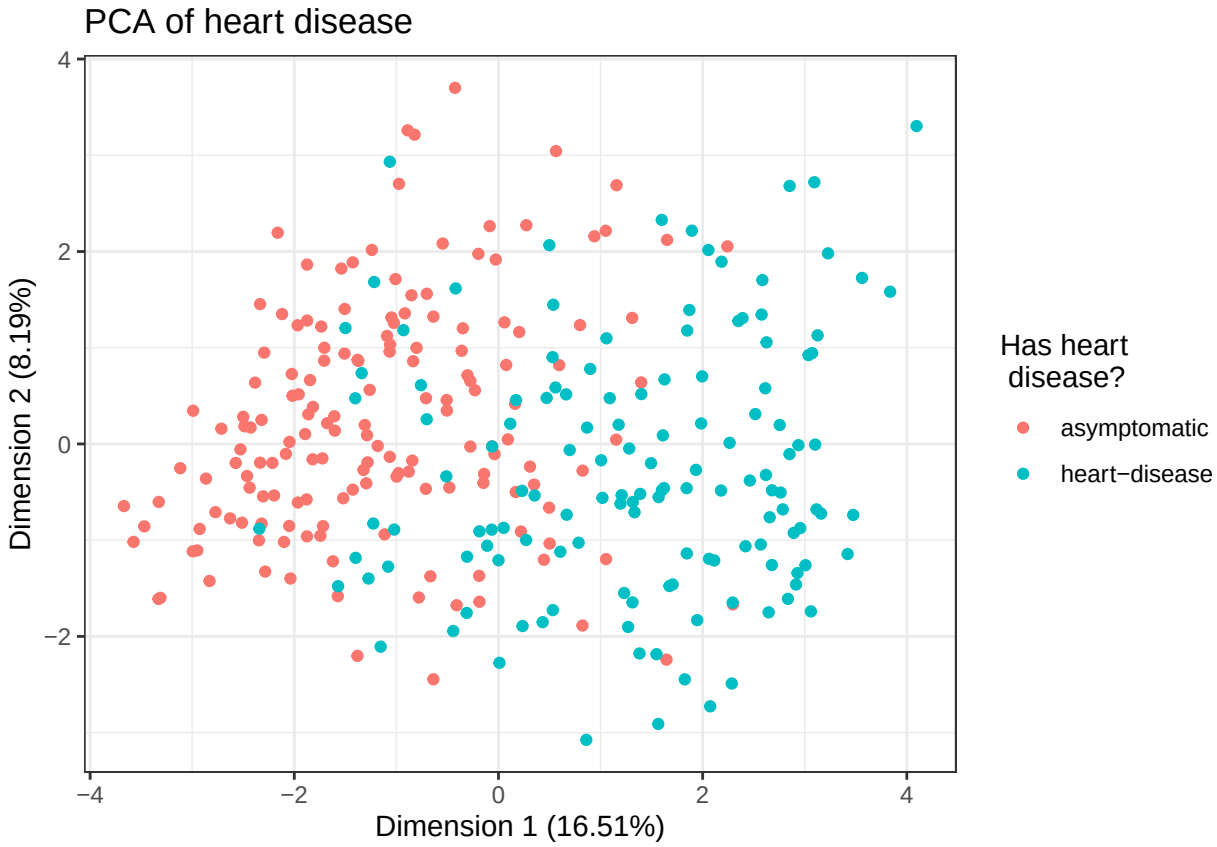
```
##         Eigenvalue Proportion Cumulative
## dim 1    3.3026702  16.513351   16.51335
## dim 2    1.6387739   8.193870   24.70722
## dim 3    1.4880233   7.440117   32.14734
## dim 4    1.3138621   6.569310   38.71665
## dim 5    1.1803770   5.901885   44.61853
## dim 6    1.1070230   5.535115   50.15365
## dim 7    1.0474271   5.237135   55.39078
## dim 8    1.0271847   5.135924   60.52671
## dim 9    0.9531850   4.765925   65.29263
## dim 10   0.9212945   4.606472   69.89910
## dim 11   0.8342686   4.171343   74.07045
## dim 12   0.8197531   4.098765   78.16921
## dim 13   0.7429420   3.714710   81.88392
## dim 14   0.7170539   3.585269   85.46919
## dim 15   0.6937680   3.468840   88.93803
## dim 16   0.5786214   2.893107   91.83114
## dim 17   0.5140594   2.570297   94.40144
## dim 18   0.4168099   2.084050   96.48549
## dim 19   0.3696068   1.848034   98.33352
## dim 20   0.3332960   1.666480  100.00000
```

```
barplot(pca$eig[,2], ylim = c(0,20), las = 2)
```

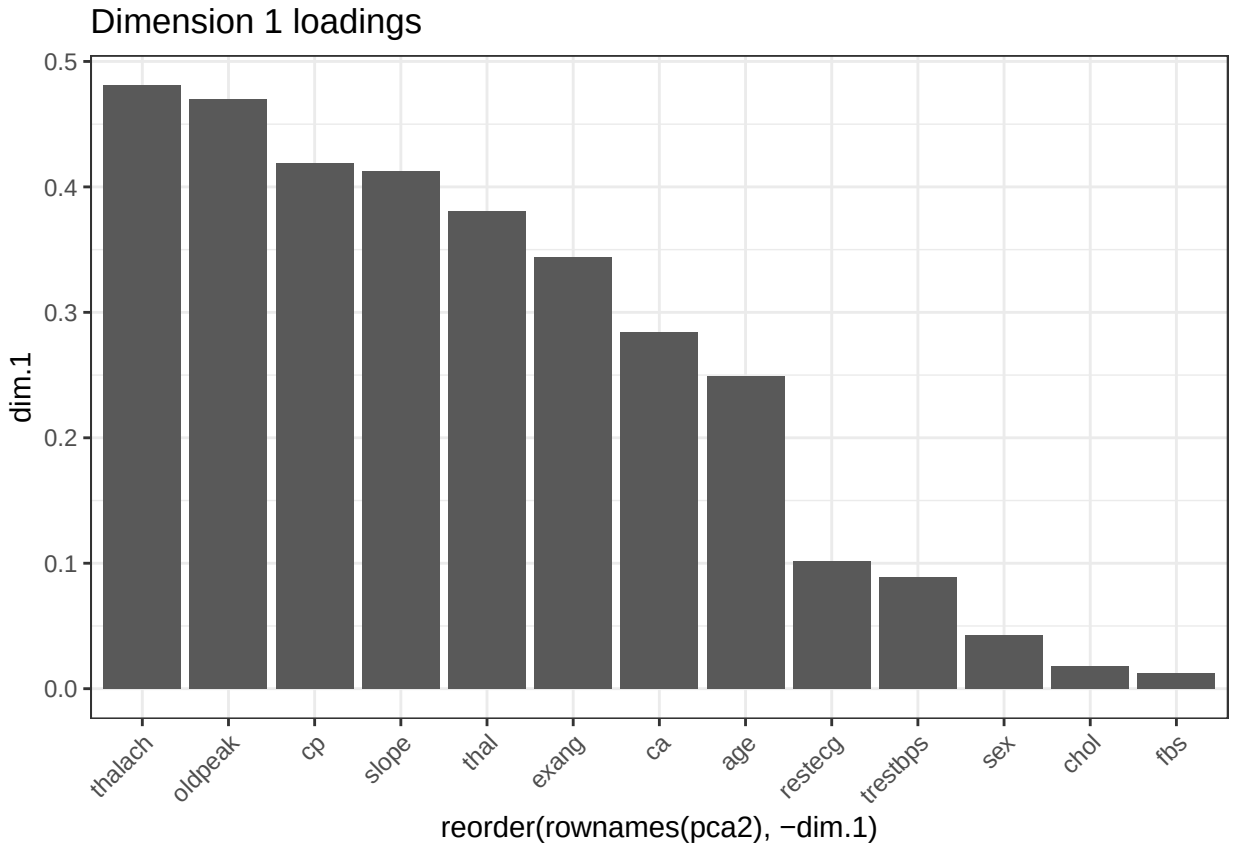



```
pca1 = data.frame(pca$ind$coord, hd_model1$target)
ggplot(pca1, aes(x = dim.1, y = dim.2, color = hd_model1.target)) +
  geom_point() +
  theme_bw() +
  guides(color = guide_legend(title = "Has heart \n disease?")) +
  ggtitle("PCA of heart disease") +
  xlab(paste("Dimension 1", paste0("(",
```

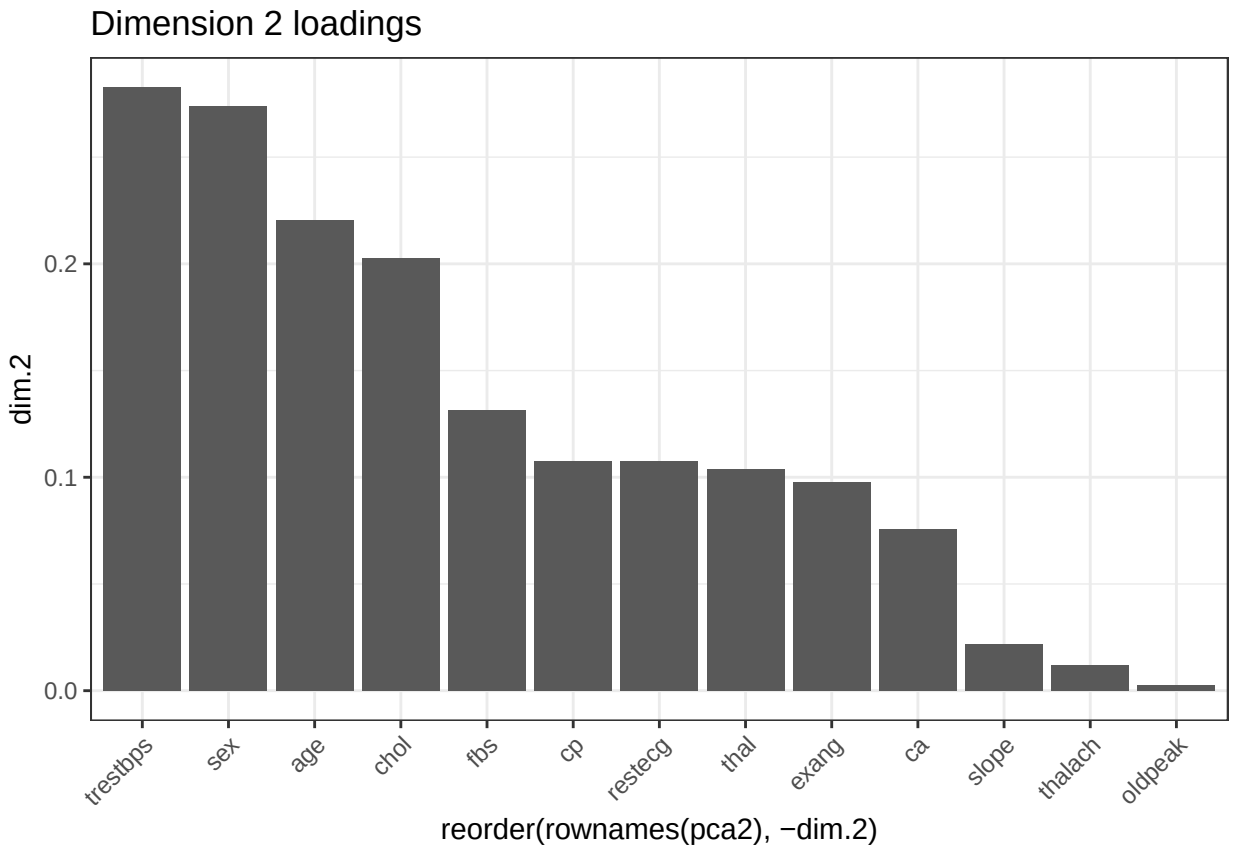
```
                                    round(pca$eig[1, 2], 2),
                                    "%", ")"))) +
  ylab(paste("Dimension 2", paste0("(",
                                    round(pca$eig[2, 2], 2),
                                    "%", ")")))
```




```
pca2 = data.frame(pca$sqload)
# Dimension 1
ggplot(pca2, aes(x = reorder(rownames(pca2), -dim.1), y = dim.1)) +
  geom_bar(stat = "identity") +
  theme_bw() + ggtitle("Dimension 1 loadings") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Dimension 2
ggplot(pca2, aes(x = reorder(rownames(pca2), -dim.2), y = dim.2)) +
  geom_bar(stat = "identity") +
  theme_bw() + ggtitle("Dimension 2 loadings") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
pcamix_data <- pca$scores[,1:6]

pcamix_data = cbind(hd_model1$target, pcamix_data)
colnames(pcamix_data) = c("target", "p1", "p2", "p3", "p4", "p5", "p6")

pcamix_data <- as.data.frame(pcamix_data)
pcamix_data <- pcamix_data %>% mutate(target = as.factor(target))

head(pcamix_data)
```

```
##   target         p1         p2         p3         p4         p5         p6
## 1      1  1.6512174  2.1204092  5.3769591  0.3571336 -1.2014289 -1.8107384
## 2      1 -1.0643937 -0.1341665  2.9356360  2.5229597  0.3106426  2.0675681
## 3      1 -2.5002721  0.2820591  0.2836409  0.6414651  1.2400444 -0.2691766
## 4      1 -2.4371924 -0.4546367  0.3247944 -0.1114306  0.5401093 -0.3380955
## 5      1 -0.7098405  0.4750501 -1.7601440  0.3058101  1.6391242  0.3027115
## 6      1  0.5012556 -1.0355505  1.1839160  0.7239259 -0.9244491 -2.8000498
```

```
pca_svm_model <- pcamix_data

set.seed(4)
train.samples_pca_svm <- pca_svm_model$target %>%
  createDataPartition(p = 0.7, list = FALSE)

train_data_pca_svm <- pca_svm_model[train.samples_pca_svm, ]
test_data_pca_svm <- pca_svm_model[-train.samples_pca_svm, ]
```

```r
pca_svm_lin <- train(target ~ ., data = train_data_pca_svm, method = "svmLinear",
                 trControl = train_control_svm, tuneLength = 10)

pca_svm_lin
```

```
## Support Vector Machines with Linear Kernel
##
## 208 samples
##   6 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 187, 188, 187, 187, 186, 188, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8378427  0.6738128
##
## Tuning parameter 'C' was held constant at a value of 1
```

```r
pred_pca_svm <- predict(pca_svm_lin, train_data_pca_svm)

confusionMatrix(table(pred_pca_svm, train_data_pca_svm$target))
```

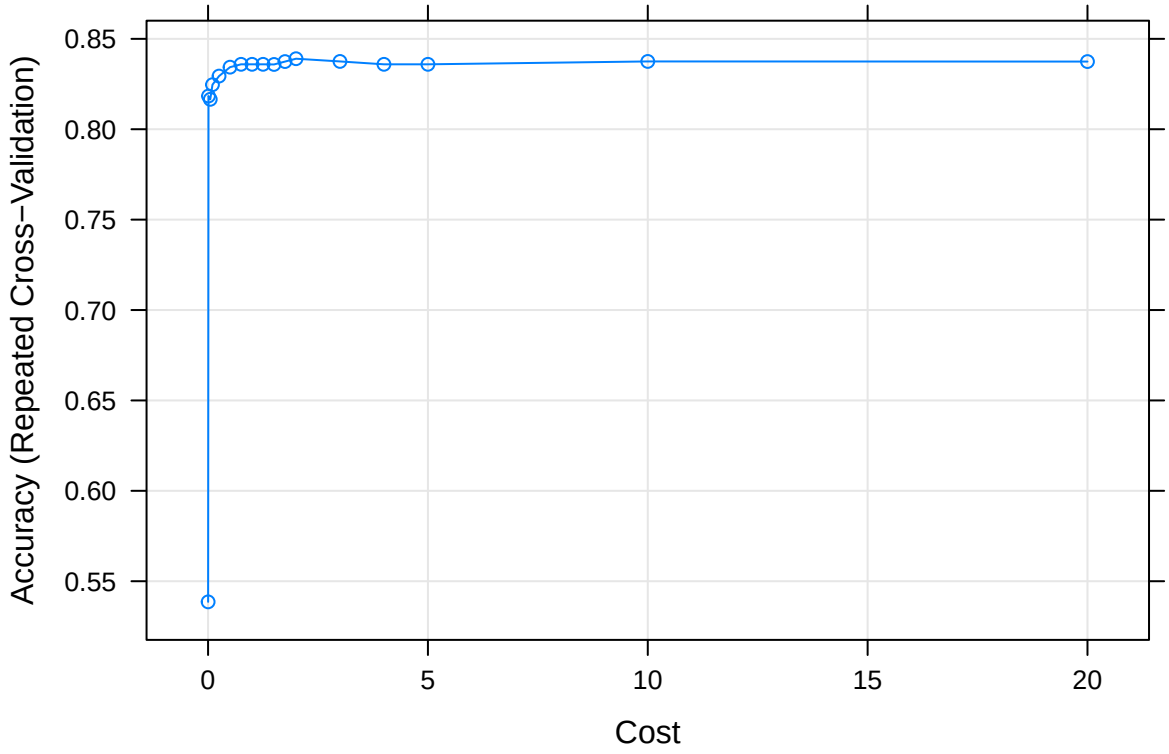
```
## Confusion Matrix and Statistics
##
##
## pred_pca_svm  1  2
##            1 95 15
##            2 17 81
##
##                Accuracy : 0.8462
##                  95% CI : (0.7898, 0.8923)
##     No Information Rate : 0.5385
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6909
##
##  Mcnemar's Test P-Value : 0.8597
##
##             Sensitivity : 0.8482
##             Specificity : 0.8438
##          Pos Pred Value : 0.8636
##          Neg Pred Value : 0.8265
##              Prevalence : 0.5385
##          Detection Rate : 0.4567
##    Detection Prevalence : 0.5288
##       Balanced Accuracy : 0.8460
##
##        'Positive' Class : 1
##
```

```r
grid_lin <- expand.grid(C = c(0.001, 0.01, 0.05, 0.1, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2,3,4,5,10,2
```

```
pca_svm_lin_grid<- train(target ~., data = train_data_pca_svm, method = "svmLinear", trControl = train_
                    tuneGrid = grid_lin, tuneLength = 10)
pca_svm_lin_grid
```

```
## Support Vector Machines with Linear Kernel
##
## 208 samples
##   6 predictor
##   2 classes: '1', '2'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 188, 187, 187, 187, 188, 187, ...
## Resampling results across tuning parameters:
##
##   C       Accuracy   Kappa
##    0.001  0.5385859  0.0000000
##    0.010  0.8183622  0.6338906
##    0.050  0.8165440  0.6315714
##    0.100  0.8246392  0.6479197
##    0.250  0.8294012  0.6570544
##    0.500  0.8343218  0.6681103
##    0.750  0.8359091  0.6711291
##    1.000  0.8359091  0.6711291
##    1.250  0.8359091  0.6711291
##    1.500  0.8358369  0.6712429
##    1.750  0.8374242  0.6744102
##    2.000  0.8390115  0.6777793
##    3.000  0.8374964  0.6746467
##    4.000  0.8359091  0.6713922
##    5.000  0.8359091  0.6713922
##   10.000  0.8374964  0.6746177
##   20.000  0.8374242  0.6745248
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 2.
```

```
plot(pca_svm_lin_grid)
```




```
best_pca_svm_lin <- pca_svm_lin_grid
```

```
pred_pca_svm_lin_best <- predict(best_pca_svm_lin, newdata = test_data_pca_svm)

cnf_pca_svm_lin <- confusionMatrix(pred_pca_svm_lin_best, test_data_pca_svm$target)
cnf_pca_svm_lin
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 42 11
##          2  6 29
##
##                Accuracy : 0.8068
##                  95% CI : (0.7088, 0.8832)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 2.609e-07
##
##                   Kappa : 0.6063
##
##  Mcnemar's Test P-Value : 0.332
##
##             Sensitivity : 0.8750
##             Specificity : 0.7250
##          Pos Pred Value : 0.7925
##          Neg Pred Value : 0.8286
##              Prevalence : 0.5455
##          Detection Rate : 0.4773
##    Detection Prevalence : 0.6023
```

```
##       Balanced Accuracy : 0.8000
##
##        'Positive' Class : 1
##
```

```
tune_pca_svm_rad <- tune(svm, target ~ ., data = train_data_pca_svm, kernel = "radial",
                         ranges = list(gamma = c(0, 0.0001, 0.001,0.01, 0.1, 0.3, 0.5, 0.75, 1), cost =

summary(tune_pca_svm_rad)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma cost
##   0.01  100
##
## - best performance: 0.155
##
## - Detailed performance results:
##     gamma cost     error dispersion
## 1  0.0000    5 0.4626190 0.14664317
## 2  0.0001    5 0.4626190 0.14664317
## 3  0.0010    5 0.1642857 0.08993153
## 4  0.0100    5 0.1595238 0.07759941
## 5  0.1000    5 0.1690476 0.09797650
## 6  0.3000    5 0.2169048 0.08058155
## 7  0.5000    5 0.2507143 0.08325431
## 8  0.7500    5 0.2559524 0.08933596
## 9  1.0000    5 0.2604762 0.07585242
## 10 0.0000   10 0.4626190 0.14664317
## 11 0.0001   10 0.4626190 0.14664317
## 12 0.0010   10 0.1833333 0.08232573
## 13 0.0100   10 0.1595238 0.08078102
## 14 0.1000   10 0.1642857 0.08993153
## 15 0.3000   10 0.2552381 0.07949830
## 16 0.5000   10 0.2457143 0.08417788
## 17 0.7500   10 0.2604762 0.07245471
## 18 1.0000   10 0.2652381 0.07578596
## 19 0.0000   50 0.4626190 0.14664317
## 20 0.0001   50 0.1690476 0.08708487
## 21 0.0010   50 0.1738095 0.08384199
## 22 0.0100   50 0.1647619 0.10022524
## 23 0.1000   50 0.2269048 0.09869423
## 24 0.3000   50 0.2697619 0.08367692
## 25 0.5000   50 0.2650000 0.10041517
## 26 0.7500   50 0.2554762 0.07424814
## 27 1.0000   50 0.2652381 0.07578596
## 28 0.0000  100 0.4626190 0.14664317
## 29 0.0001  100 0.1833333 0.08232573
## 30 0.0010  100 0.1642857 0.07462006
## 31 0.0100  100 0.1550000 0.09563706
## 32 0.1000  100 0.2697619 0.09007185
```

```
## 33 0.3000  100 0.2745238 0.07993486
## 34 0.5000  100 0.2650000 0.10041517
## 35 0.7500  100 0.2554762 0.07424814
## 36 1.0000  100 0.2652381 0.07578596
```

```
best_pca_svm_rad <- tune_pca_svm_rad$best.model
```

```
pred_pca_svm_rad_tune <- predict(best_pca_svm_rad, newdata = test_data_pca_svm)

cnf_pca_svm_rad <- confusionMatrix(pred_pca_svm_rad_tune, test_data_pca_svm$target)
cnf_pca_svm_rad
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2
##          1 40 11
##          2  8 29
##
##                Accuracy : 0.7841
##                  95% CI : (0.6835, 0.8647)
##     No Information Rate : 0.5455
##     P-Value [Acc > NIR] : 2.788e-06
##
##                   Kappa : 0.5618
##
##  Mcnemar's Test P-Value : 0.6464
##
##             Sensitivity : 0.8333
##             Specificity : 0.7250
##          Pos Pred Value : 0.7843
##          Neg Pred Value : 0.7838
##              Prevalence : 0.5455
##          Detection Rate : 0.4545
##    Detection Prevalence : 0.5795
##       Balanced Accuracy : 0.7792
##
##        'Positive' Class : 1
##
```

Compare Models

```
do.call(rbind, Map(data.frame, Lasso_min = cnf_min$byClass,
                   Lasso_1se = cnf_1se$byClass,
                   Ridge_min = cnf_min_r$byClass,
                   Ridge_1se = cnf_1se_r$byClass,
                   RandomForest = cnf_rf$byClass,
                   LinearSVM = cnf_svm_lin$byClass,
                   RadialSVM = cnf_svm_rad$byClass,
                   PCASVMLinear = cnf_pca_svm_lin$byClass,
                   PCASVMRadial = cnf_pca_svm_rad$byClass))
```

```
##                     Lasso_min Lasso_1se Ridge_min Ridge_1se RandomForest
## Sensitivity         0.7000000 0.6000000 0.7500000 0.0000000    0.7750000
## Specificity         0.9375000 0.9791667 0.9583333 1.0000000    0.8750000
```

```
## Pos Pred Value       0.9032258 0.9600000 0.9375000       NaN    0.8378378
## Neg Pred Value       0.7894737 0.7460317 0.8214286 0.5454545    0.8235294
## Precision            0.9032258 0.9600000 0.9375000        NA    0.8378378
## Recall               0.7000000 0.6000000 0.7500000 0.0000000    0.7750000
## F1                   0.7887324 0.7384615 0.8333333        NA    0.8051948
## Prevalence           0.4545455 0.4545455 0.4545455 0.4545455    0.4545455
## Detection Rate       0.3181818 0.2727273 0.3409091 0.0000000    0.3522727
## Detection Prevalence 0.3522727 0.2840909 0.3636364 0.0000000    0.4204545
## Balanced Accuracy    0.8187500 0.7895833 0.8541667 0.5000000    0.8250000
##                      LinearSVM RadialSVM PCASVMLinear PCASVMRadial
## Sensitivity          0.6923077 0.6923077    0.8750000    0.8333333
## Specificity          0.8125000 0.8541667    0.7250000    0.7250000
## Pos Pred Value       0.7500000 0.7941176    0.7924528    0.7843137
## Neg Pred Value       0.7647059 0.7735849    0.8285714    0.7837838
## Precision            0.7500000 0.7941176    0.7924528    0.7843137
## Recall               0.6923077 0.6923077    0.8750000    0.8333333
## F1                   0.7200000 0.7397260    0.8316832    0.8080808
## Prevalence           0.4482759 0.4482759    0.5454545    0.5454545
## Detection Rate       0.3103448 0.3103448    0.4772727    0.4545455
## Detection Prevalence 0.4137931 0.3908046    0.6022727    0.5795455
## Balanced Accuracy    0.7524038 0.7732372    0.8000000    0.7791667
```