# *Exploratory Data Analysis and Prediction on Heart Attack*

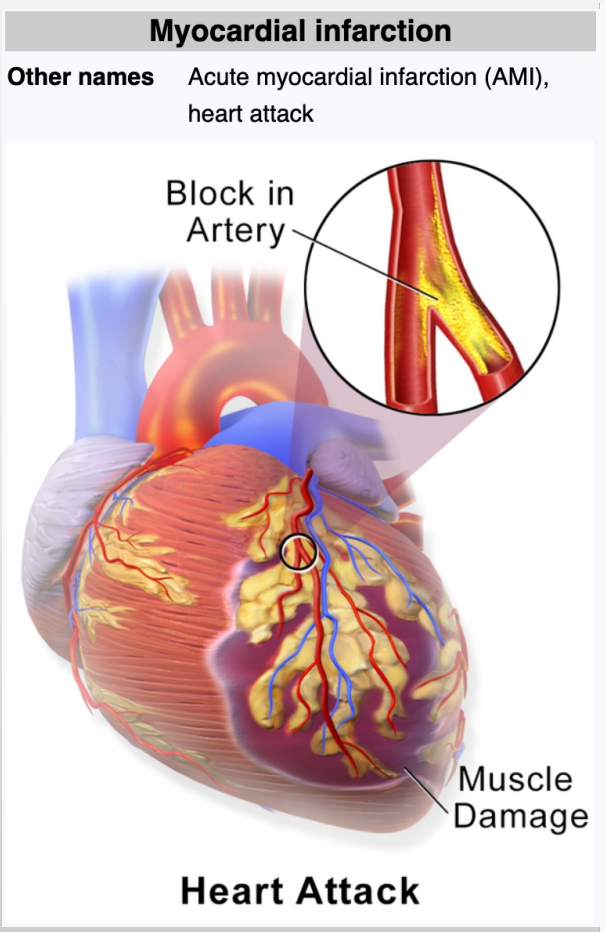
**AMS6001 Group Project (Group 2)**


FUNG Kam Man / 馮錦文
LIU Tong / 劉通
TIAN Donghao / 田冬皓
XIAO Xin / 肖鑫
HE Xiaoxiao / 何瀟瀟


**Nov 23, 2023**

# Introduction

## ***Heart Attack*** *-- Definition from the Wikipedia*




**Myocardial infarction**

| | |
|---|---|
| **Other names** | Acute myocardial infarction (AMI), heart attack |

A myocardial infarction occurs when an atherosclerotic plaque slowly builds up in the inner lining of a coronary artery and then suddenly ruptures, causing catastrophic thrombus formation, totally occluding the artery and preventing blood flow downstream to the heart muscle.

| | |
|---|---|
| **Specialty** | Cardiology, emergency medicine |
| **Symptoms** | Chest pain, shortness of breath, nausea/vomiting, dizziness or lightheadedness, cold sweat, feeling tired; arm, neck, back, jaw, or stomach pain,[1][2] decreased level or total loss of consciousness |
| **Complications** | Heart failure, irregular heartbeat, cardiogenic shock, coma, cardiac arrest[3][4] |
| **Causes** | Usually coronary artery disease[3] |
| **Risk factors** | High blood pressure, smoking, diabetes, lack of exercise, obesity, high blood cholesterol[5][6] |
| **Diagnostic method** | Electrocardiograms (ECGs), blood tests, coronary angiography[7] |
| **Treatment** | Percutaneous coronary intervention, thrombolysis[8] |
| **Medication** | Aspirin, nitroglycerin, heparin[8][9] |
| **Prognosis** | STEMI 10% risk of death (developed world)[8] |
| **Frequency** | 15.9 million (2015)[10] |

https://en.wikipedia.org/wiki/Myocardial_infarction

## *Goal of Study:*

*- To explore the dataset through data analysis to identify attributes strongly associated with heart attacks*

*- To develop a predictive model using data mining techniques to accurately forecast the occurrence of heart attacks*

# Data Preprocessing

- ***Heart.csv*** **: 303 objects (302 distinct), 14 attributes**

**Numerical Data (5):** ⟶ **Scaling**

- **age [int.]** : Age of the patient
- **trtbps [cont.]** : resting blood pressure (in mmHg)
- **chol [cont.]** : cholestoral in mg/dl fetched via BMI sensor
- **thalachh [int.]** : maximum heart rate achieved
- **oldpeak [cont.] :** ST depression induced by exercise relative to rest

**Categorical Data (9):** ⟶ **Factorization**

- **sex [cat.]** : Sex of the patient (1 = male; 0 = female)
- **exng [cat.]** : exercise induced angina (1 = yes; 0 = no)
- **ca [cat.]**: number of major vessels (0-3)
- **cp [cat.]** : Chest Pain type
- **fbs [cat.]** : fasting blood sugar (> 120 mg/dl) (1 = true; 0 = false)
- **restecg [cat.]** : resting electrocardiographic results
- **slp [cat.] :** The slope of the peak exercise ST segment
- **thall [cat.]** : Thal rate
- **output [cat.]** : target (0= less chance of heart attack; 1= more chance of heart attack)



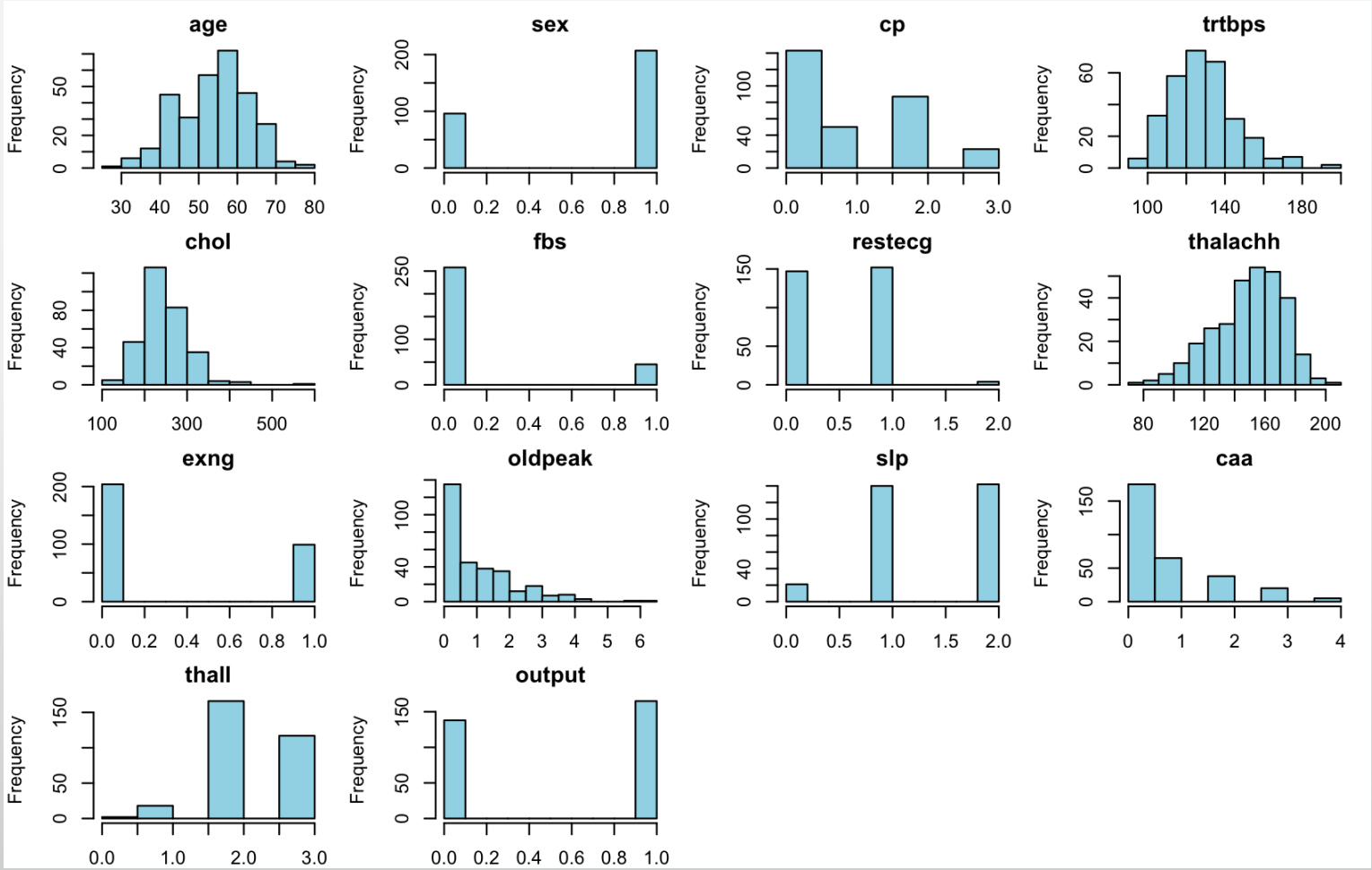
Source: https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset/data

# Exploratory Data Analysis (1) Summary Statistics

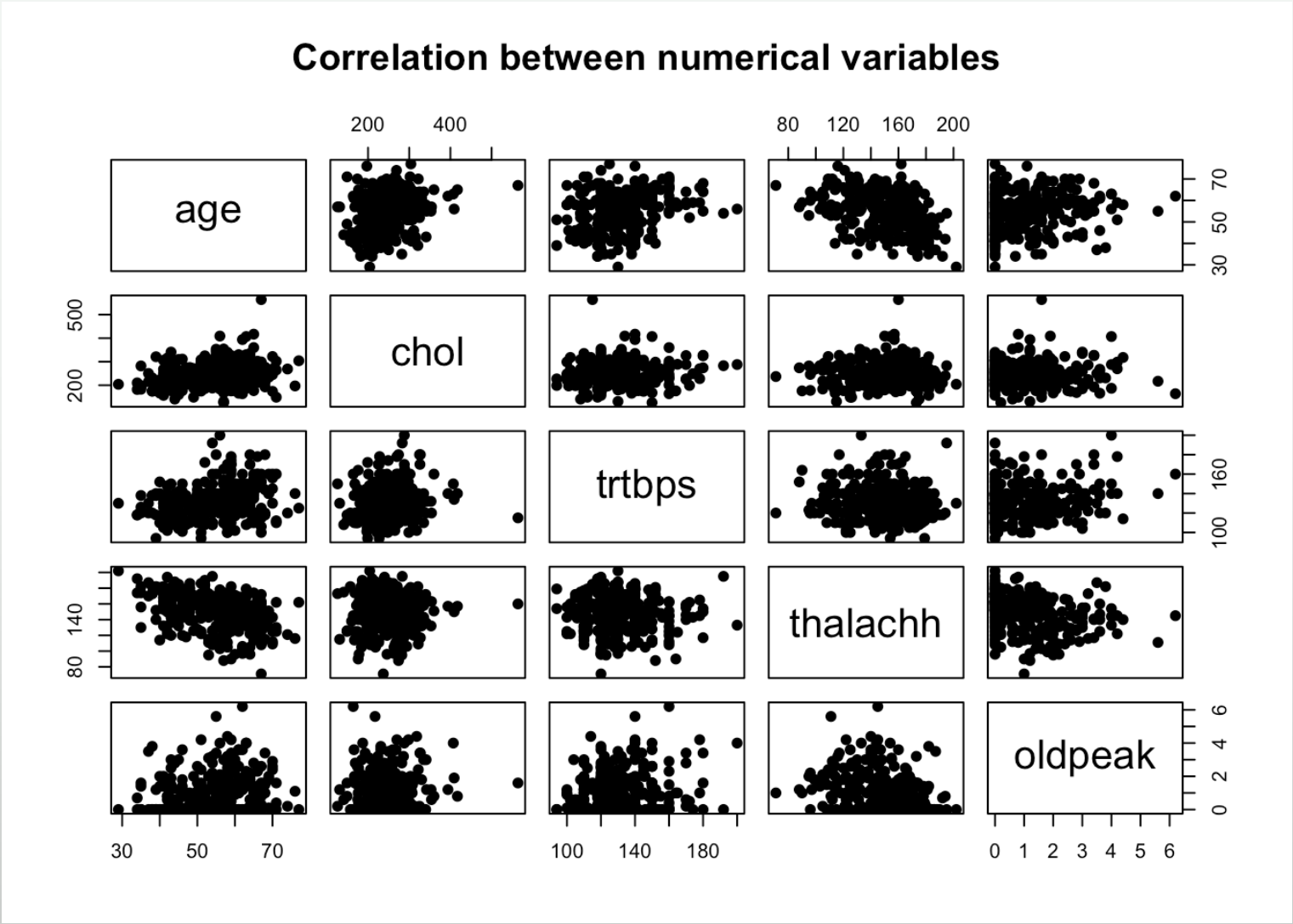
```
> summary(clean.data)
      age              sex               cp             trtbps           chol      
 Min.   :29.00   Min.   :0.0000   Min.   :0.0000   Min.   : 94.0   Min.   :126.0  
 1st Qu.:48.00   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:120.0   1st Qu.:211.0  
 Median :55.50   Median :1.0000   Median :1.0000   Median :130.0   Median :240.5  
 Mean   :54.42   Mean   :0.6821   Mean   :0.9636   Mean   :131.6   Mean   :246.5  
 3rd Qu.:61.00   3rd Qu.:1.0000   3rd Qu.:2.0000   3rd Qu.:140.0   3rd Qu.:274.8  
 Max.   :77.00   Max.   :1.0000   Max.   :3.0000   Max.   :200.0   Max.   :564.0  
      fbs             restecg          thalachh          exng          oldpeak     
 Min.   :0.000   Min.   :0.0000   Min.   : 71.0   Min.   :0.0000   Min.   :0.000  
 1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:133.2   1st Qu.:0.0000   1st Qu.:0.000  
 Median :0.000   Median :1.0000   Median :152.5   Median :0.0000   Median :0.800  
 Mean   :0.149   Mean   :0.5265   Mean   :149.6   Mean   :0.3278   Mean   :1.043  
 3rd Qu.:0.000   3rd Qu.:1.0000   3rd Qu.:166.0   3rd Qu.:1.0000   3rd Qu.:1.600  
 Max.   :1.000   Max.   :2.0000   Max.   :202.0   Max.   :1.0000   Max.   :6.200  
      slp              caa             thall           output     
 Min.   :0.000   Min.   :0.0000   Min.   :0.000   Min.   :0.000  
 1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:0.000  
 Median :1.000   Median :0.0000   Median :2.000   Median :1.000  
 Mean   :1.397   Mean   :0.7185   Mean   :2.315   Mean   :0.543  
 3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:3.000   3rd Qu.:1.000  
 Max.   :2.000   Max.   :4.0000   Max.   :3.000   Max.   :1.000  
```

# Exploratory Data Analysis (2) Univariate Analysis

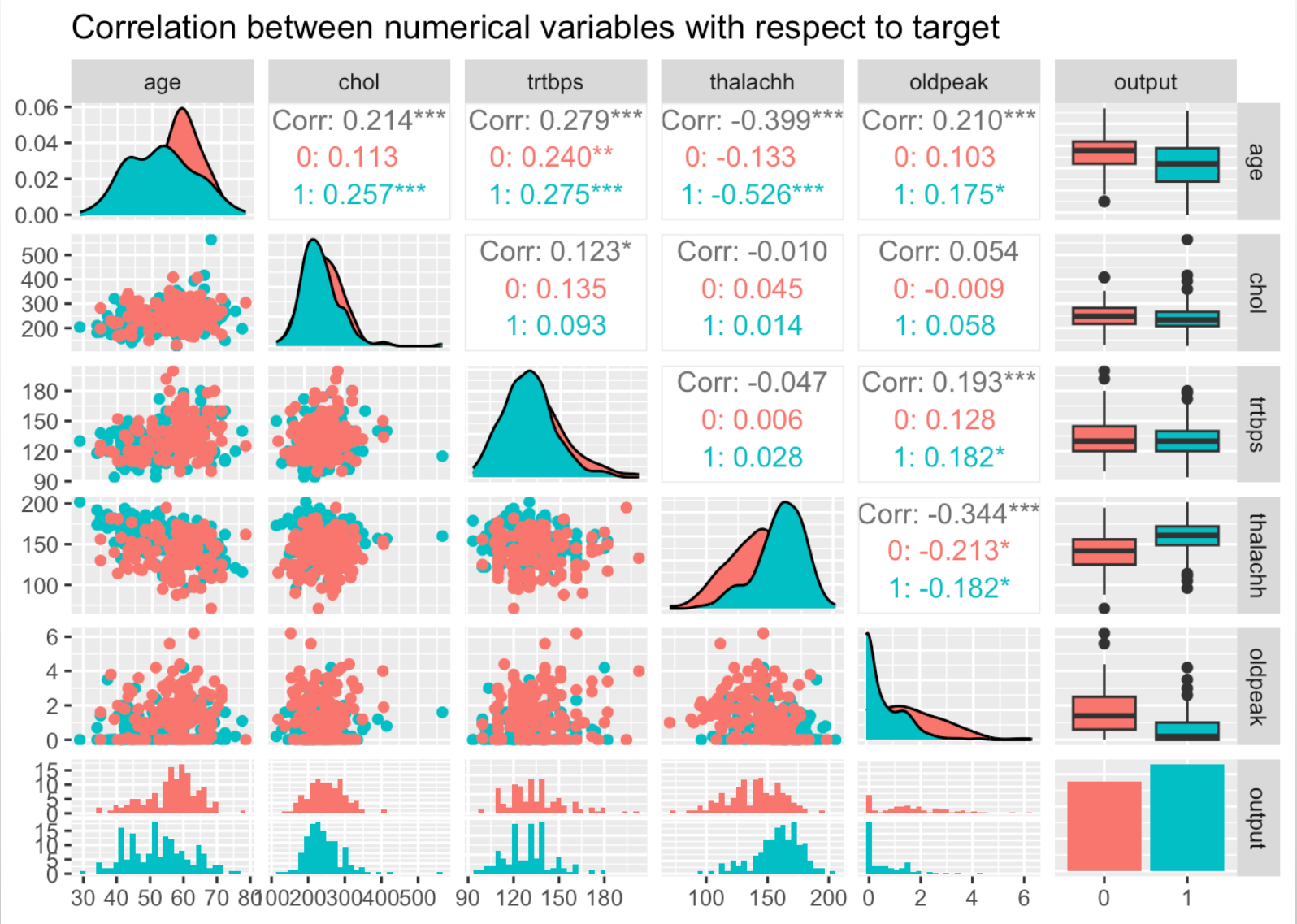
## Distribution of All Variables



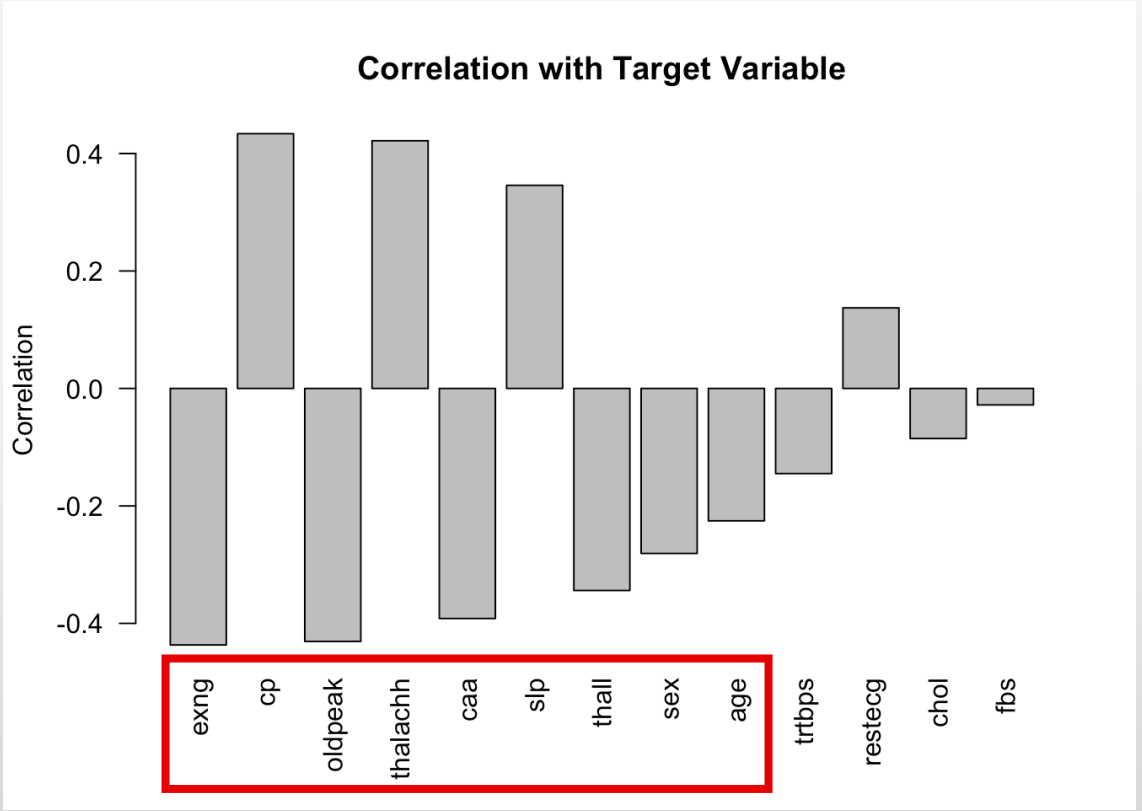
# Exploratory Data Analysis (3) Bivariate Analysis of Numeric Variables



# Exploratory Data Analysis (3) Bivariate Analysis of Numeric Variables



# Exploratory Data Analysis (4) Correlation with Target Variable




Correlation from **9 variables** that exceeds threshold (0.2):

```
## [1] "age"      "sex"      "cp"       "thalachh" "exng"     "oldpeak"  "slp"
## [8] "caa"      "thall"
```

# Exploratory Data Analysis (5) Subset selection (Backward/BIC)

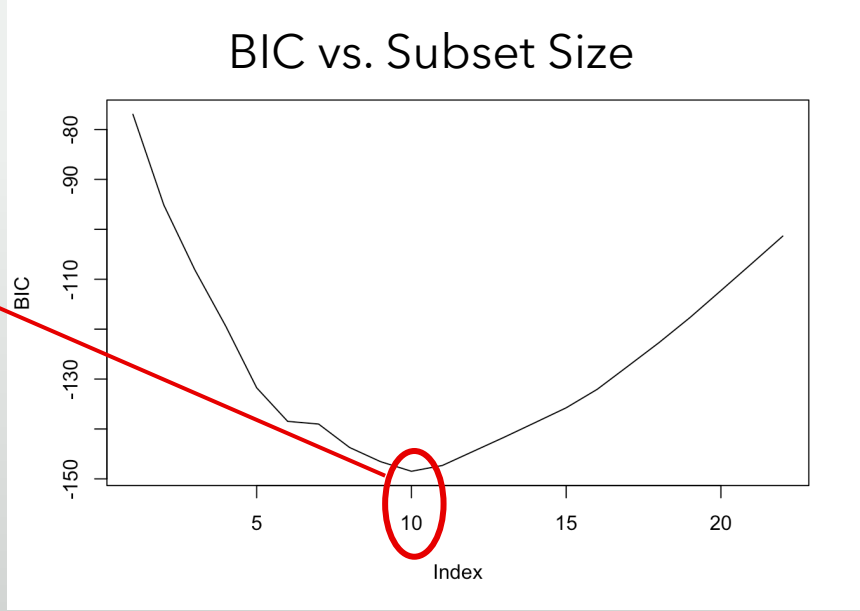
## Backward selection to find the best subset

```
result.all <-regsubsets(output ~ ., data = train_data_k, method = "backward", nvm
ax=22)
summary(result.all)
```

```
## Subset selection object
## Call: regsubsets.formula(output ~ ., data = train_data_k, method = "backward",
##     nvmax = 22)
## 22 Variables  (and intercept)
## Selection Algorithm: backward
##           age sex1 cp1 cp2 cp3 trtbps chol fbs1 restecg1 restecg2 thalachh exng1 oldpeak slp1 slp2 caa1 caa2 caa3 caa4 thall1 thall2 thall3
## 1  ( 1 )  " "  " "  " "  " "  " "  " "  " "  " "  " "     " "      " "     " "  " "     " "  " "  " "  " "  " "  " "  " "    "*"    " "
## 2  ( 1 )  " "  " "  " "  " "  " "  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  " "  " "  " "  " "  " "  " "    "*"    " "
## 3  ( 1 )  " "  " "  " "  " "  " "  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  " "  "*"  " "  " "  " "  " "    "*"    " "
## 4  ( 1 )  " "  " "  " "  " "  " "  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  " "  " "  " "  " "    "*"    " "
## 5  ( 1 )  " "  " "  " "  " "  " "  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 6  ( 1 )  " "  " "  " "  " "  "*"  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 7  ( 1 )  " "  " "  " "  "*"  "*"  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 8  ( 1 )  " "  " "  " "  "*"  "*"  "*"  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 9  ( 1 )  " "  " "  "*"  "*"  "*"  "*"  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 10  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  " "     " "      " "     " "  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 11  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  " "     " "      " "     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 12  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  " "     " "      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  " "    "*"    " "
## 13  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  " "     " "      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 14  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  " "     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 15  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  " "  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 16  ( 1 ) " "  " "  "*"  "*"  "*"  "*"  "*"  " "  " "  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 17  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  " "  " "  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 18  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 19  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    " "
## 20  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  " "  " "  "*"    "*"    "*"
## 21  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"     "*"      "*"     "*"  " "     " "  "*"  "*"  "*"  "*"  "*"  "*"  "*"    "*"    "*"
## 22  ( 1 ) "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"     "*"      "*"     "*"  "*"     "*"  "*"  "*"  "*"  "*"  "*"  "*"  "*"    "*"    "*"
```

## BIC plot to find the proper size of subset

```
plot(summary(result.all)$bic, type="l", ylab="BIC")
```




**Subset of 10 includes factorized variables, which only belong to 6 distinct variables.**
**Subset of 6 variables selected:** (sex, cp, trtbps, slp, caa, thall)

# Exploratory Data Analysis (6) Summary

- Correlation between numerical variables are not strong;
- There is no clear distinction among the numerical variables with respect to the target;
- Correlation between predictors and target variable suggests **a subset of 9** variables:
  - **(age + sex + exng + caa + cp + thalachh + slp + oldpeak + thall)**
- Backward subset selection suggests that **a subset of 6** could be proper:
  - **(sex + cp + trtbps + slp + caa + thall)**
- We will also try to build models with all predictors and see their effect.

# Modeling Techniques (1) Methodology

| Algorithms | # Predictors | Other Details | Cross-Validation |
|---|---|---|---|
| KNN | All, 9, 6 | method = "knn"<br>k (#nearest neighbors) = 1:16; preProcess = "scale" | 10 folds |
| Decision Tree | All, 9, 6 | method = "rpart" | 10 folds |
| SVM | All, 9, 6 | method = "svmLinear" | 10 folds |
| Naïve Bayes | All, reduced set | method = "nb"<br>fL = 1:5, usekernal = c(TRUE,FALSE), adjust = 1:3 | 10 folds |
| Logistic Regression | All, 9, 6 | family = "binominal" | - |

# Modeling Techniques (2) Model Training

KNN

## (1) Model with all predictors (KNN)

```
# Set trainControl
ctrl <- trainControl(method = "cv", number = 10) # 10-fold Cross-validation

# Cross-validation using train() function
knn_model <- train_data_k %>% train(output ~ .,  # Using all predictors
                    data = .,
                    preProcess = "scale",
                    method = "knn",
                    trControl = ctrl,
                    tuneGrid=data.frame(k = 1:16),
                    tuneLength = 10)

knn_model
```

```
## k-Nearest Neighbors
##
## 257 samples
##  13 predictor
##   2 classes: '0', '1'
##
## Pre-processing: scaled (22)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 232, 231, 231, 231, 231, ...
## Resampling results across tuning parameters:
##
##    k   Accuracy   Kappa
##     1  0.7473846  0.4928546
##     2  0.7515385  0.4971629
##     3  0.7784615  0.5544117
##     4  0.7629231  0.5230654
##     5  0.7940000  0.5867561
##     6  0.7980000  0.5917257
##     7  0.8215385  0.6399477
##     8  0.8369231  0.6699859
##     9  0.8369231  0.6722639
##    10  0.8330769  0.6640361
##    11  0.8253846  0.6492807
##    12  0.8252308  0.6488483
##    13  0.8252308  0.6492229
##    14  0.8329231  0.6646882
##    15  0.8406154  0.6796062
##    16  0.8367692  0.6706394
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 15.
```

| k | Accuracy | Kappa |
|---|---|---|
| 15 | 0.84062 | 0.67961 |

# Modeling Techniques (2) Model Training

KNN

## (2) Model with 9 predictors (KNN9)

"age", "sex", "exng", "caa","cp","thalachh","slp", "oldpeak","thall"

```
#set trainControl
ctrl <- trainControl(method = "cv", number = 10) # 10-fold Cross-validation

# Cross-validation using train() function
knn_model9 <- train_data_k %>% train(output ~ age + sex + exng + caa + cp + thalach
h + slp + oldpeak + thall,        # Using selected 9 variables
                    data = .,
                    preProcess = "scale",
                    method = "knn",
                    trControl = ctrl,
                    tuneGrid=data.frame(k = 1:16),
                    tuneLength = 10)

knn_model9
```

| k | Accuracy | Kappa |
|---|---|---|
| 15 | 0.84831 | 0.69373 |

```
## k-Nearest Neighbors
##
## 257 samples
##   9 predictor
##   2 classes: '0', '1'
##
## Pre-processing: scaled (17)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 232, 231, 231, 232, 231, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.7741538  0.5442621
##    2  0.7621538  0.5222910
##    3  0.8055385  0.6055594
##    4  0.7969231  0.5903918
##    5  0.8169231  0.6303136
##    6  0.8250769  0.6465956
##    7  0.8323077  0.6619749
##    8  0.8401538  0.6769634
##    9  0.8364615  0.6697731
##   10  0.8366154  0.6699944
##   11  0.8286154  0.6546815
##   12  0.8404615  0.6787726
##   13  0.8403077  0.6776244
##   14  0.8444615  0.6857555
##   15  0.8483077  0.6937292
##   16  0.8403077  0.6776502
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 15.
```

# Modeling Techniques (2) Model Training

KNN

## (3) Model with 6 predictors (KNN6)

"sex", "cp", "thalachh","exng", "caa", "thall"

```
#set trainControl
ctrl <- trainControl(method = "cv", number = 10) # 10-fold Cross-validation

# Cross-validation using train() function
knn_model6 <- train_data_k %>% train(output ~ sex+cp+oldpeak+slp+caa+thall,
# Using selected 6 variables
                    data = .,
                    preProcess = "scale",
                    method = "knn",
                    trControl = ctrl,
                    tuneGrid=data.frame(k = 1:16),
                    tuneLength = 10)

knn_model6
```

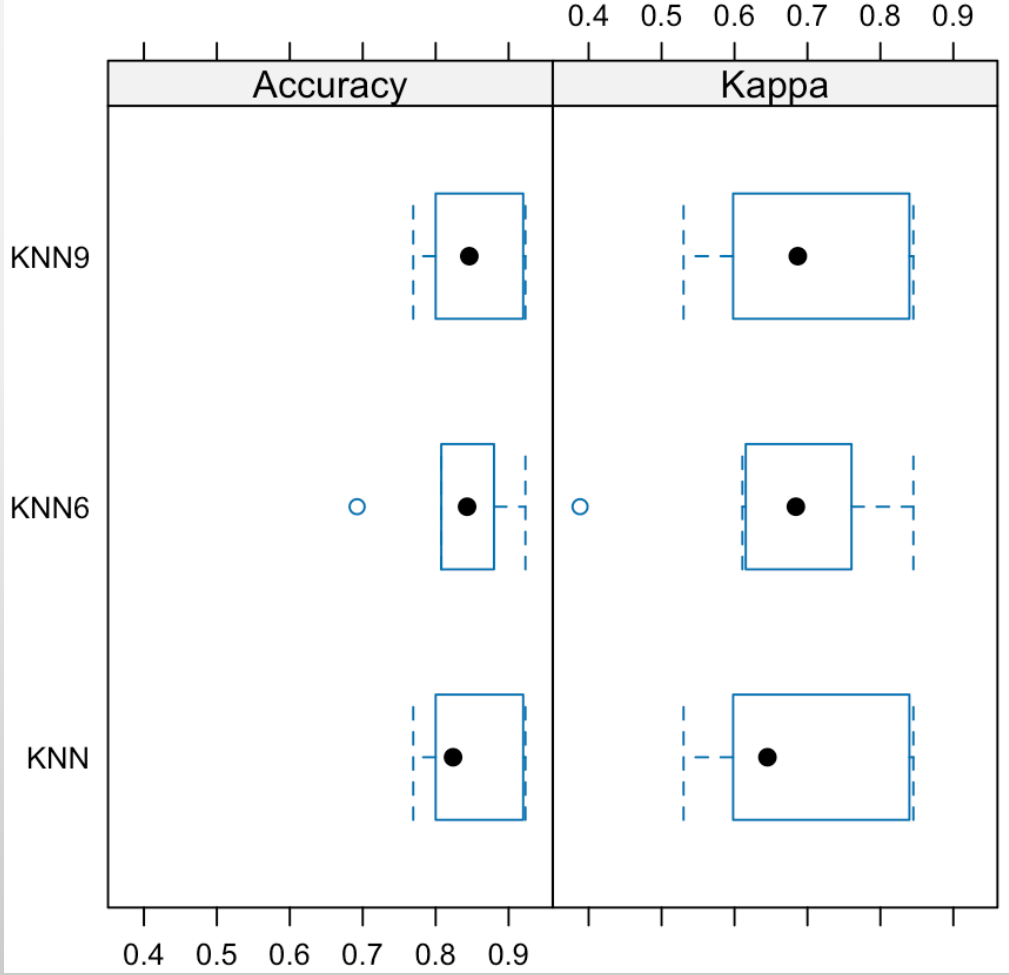
| k | Accuracy | Kappa |
|---|---|---|
| 16 | 0.84062 | 0.68014 |

```
## k-Nearest Neighbors
##
## 257 samples
##   6 predictor
##   2 classes: '0', '1'
##
## Pre-processing: scaled (14)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 231, 231, 231, 232, 232, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    1  0.7670769  0.5314654
##    2  0.7820000  0.5614195
##    3  0.8058462  0.6083055
##    4  0.7936923  0.5825088
##    5  0.8130769  0.6237295
##    6  0.8175385  0.6314283
##    7  0.8170769  0.6317802
##    8  0.8133846  0.6256417
##    9  0.8210769  0.6410296
##   10  0.8130769  0.6242338
##   11  0.8247692  0.6479192
##   12  0.8366154  0.6719848
##   13  0.8249231  0.6487688
##   14  0.8289231  0.6561203
##   15  0.8367692  0.6721742
##   16  0.8406154  0.6801429
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 16.
```

# Modeling Techniques (3) Model Evaluation

**Comparison of three KNN models:**

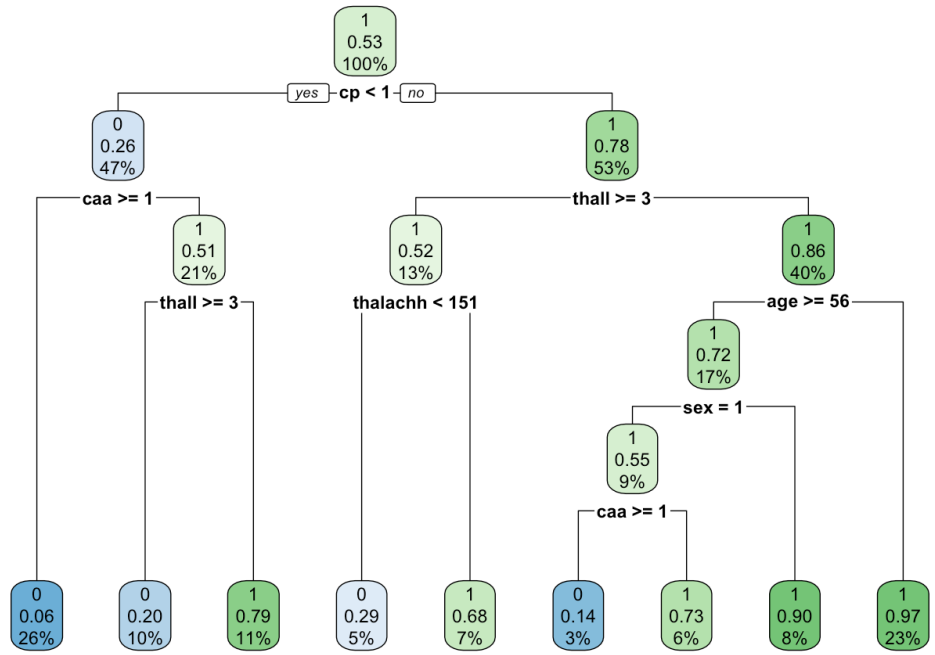
| Model | K | Accuracy | Kappa |
|---|---|---|---|
| KNN | 15 | 0.84062 | 0.67961 |
| KNN9 | 15 | 0.84831 | 0.69373 |
| KNN6 | 16 | 0.84062 | 0.68014 |



# Modeling Techniques (2) Model Training

**Decision Tree**

## (1) Model with all predictors (DT)




**Accuracy**
0.84108

**Kappa**
0.67880

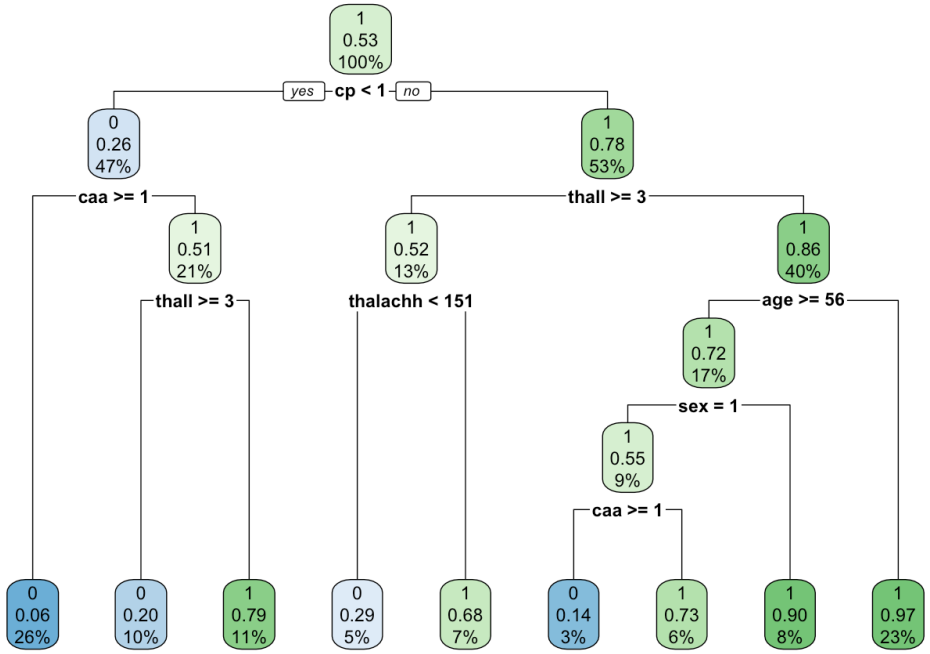
```
## CART
##
## 257 samples
##  13 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 232, 231, 232, 231, 231, 231, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.8410769  0.6788024
##   0.05462963  0.8173846  0.6285750
##   0.10925926  0.7592308  0.5137451
##   0.16388889  0.7592308  0.5137451
##   0.21851852  0.7592308  0.5137451
##   0.27314815  0.7592308  0.5137451
##   0.32777778  0.7592308  0.5137451
##   0.38240741  0.7592308  0.5137451
##   0.43703704  0.7592308  0.5137451
##   0.49166667  0.6343077  0.2289262
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

# Modeling Techniques (2) Model Training

**Decision Tree**

## (2) Model with 9 predictors (DT9)

```
"age", "sex", "exng", "caa","cp","thalachh","slp", "oldpeak","thall"
```




```
## CART
##
## 257 samples
##   9 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 232, 231, 232, 231, 231, ...
## Resampling results across tuning parameters:
##
##   cp           Accuracy   Kappa
##   0.00000000   0.8364615  0.6704850
##   0.05462963   0.8246154  0.6445699
##   0.10925926   0.7704615  0.5383165
##   0.16388889   0.7704615  0.5383165
##   0.21851852   0.7704615  0.5383165
##   0.27314815   0.7704615  0.5383165
##   0.32777778   0.7704615  0.5383165
##   0.38240741   0.7704615  0.5383165
##   0.43703704   0.7704615  0.5383165
##   0.49166667   0.6143077  0.1896499
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

**Accuracy**
0.83646

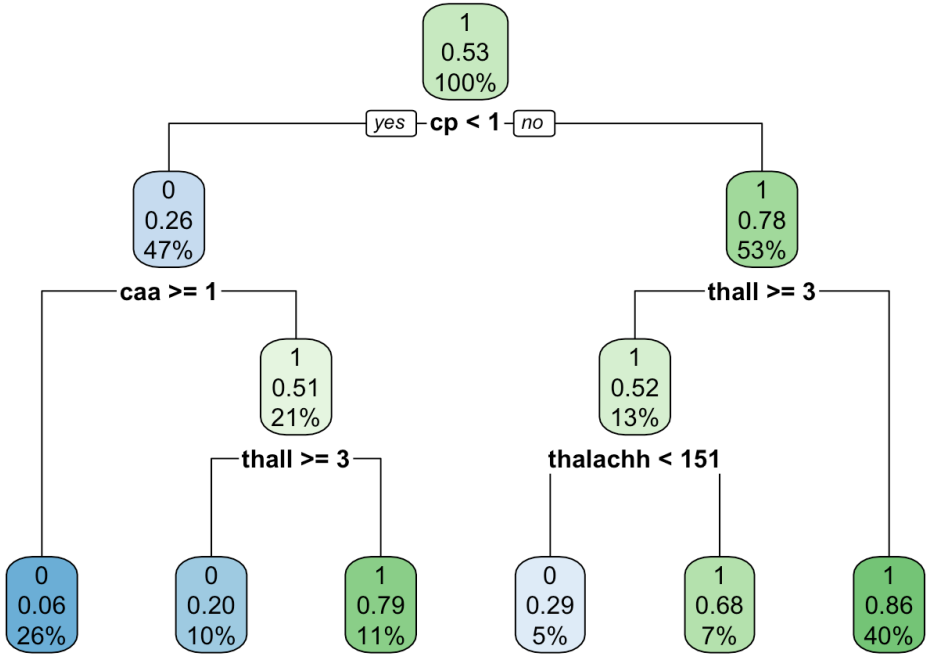
**Kappa**
0.67049

# Modeling Techniques (2) Model Training

**Decision Tree**

## (3) Model with 6 predictors (suggested by subset selection) (DT6)

`"sex", "cp", "thalachh","exng", "caa", "thall"`




```
## CART
##
## 257 samples
##   6 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 232, 231, 232, 231, 231, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.8483077  0.6952327
##   0.05462963  0.8290769  0.6505581
##   0.10925926  0.7630769  0.5243090
##   0.16388889  0.7630769  0.5243090
##   0.21851852  0.7630769  0.5243090
##   0.27314815  0.7630769  0.5243090
##   0.32777778  0.7630769  0.5243090
##   0.38240741  0.7630769  0.5243090
##   0.43703704  0.7630769  0.5243090
##   0.49166667  0.6493846  0.2577948
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

**Accuracy**
0.84831

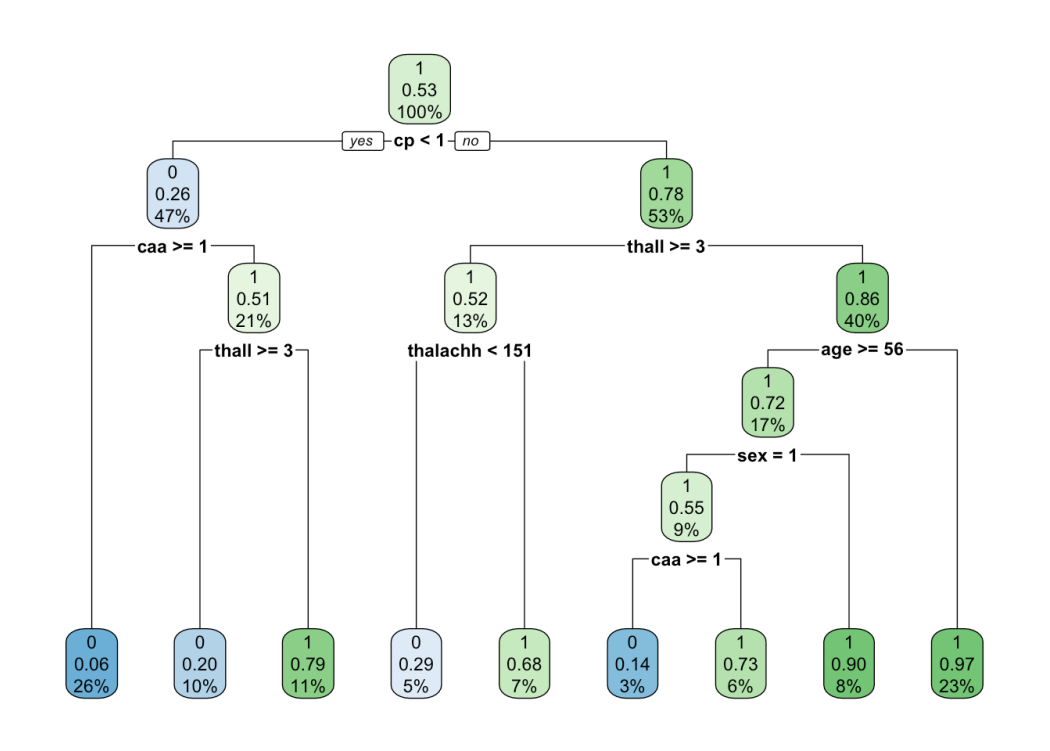
**Kappa**
0.69523

# Modeling Techniques (2) Model Training

**Decision Tree**

## (4) Model with 6 predictors (chosen from first tree model) (DT6_2)

```
"age","sex", "cp", "thalachh", "caa", "thall",
```




**Accuracy**
0.84108

**Kappa**
0.67880

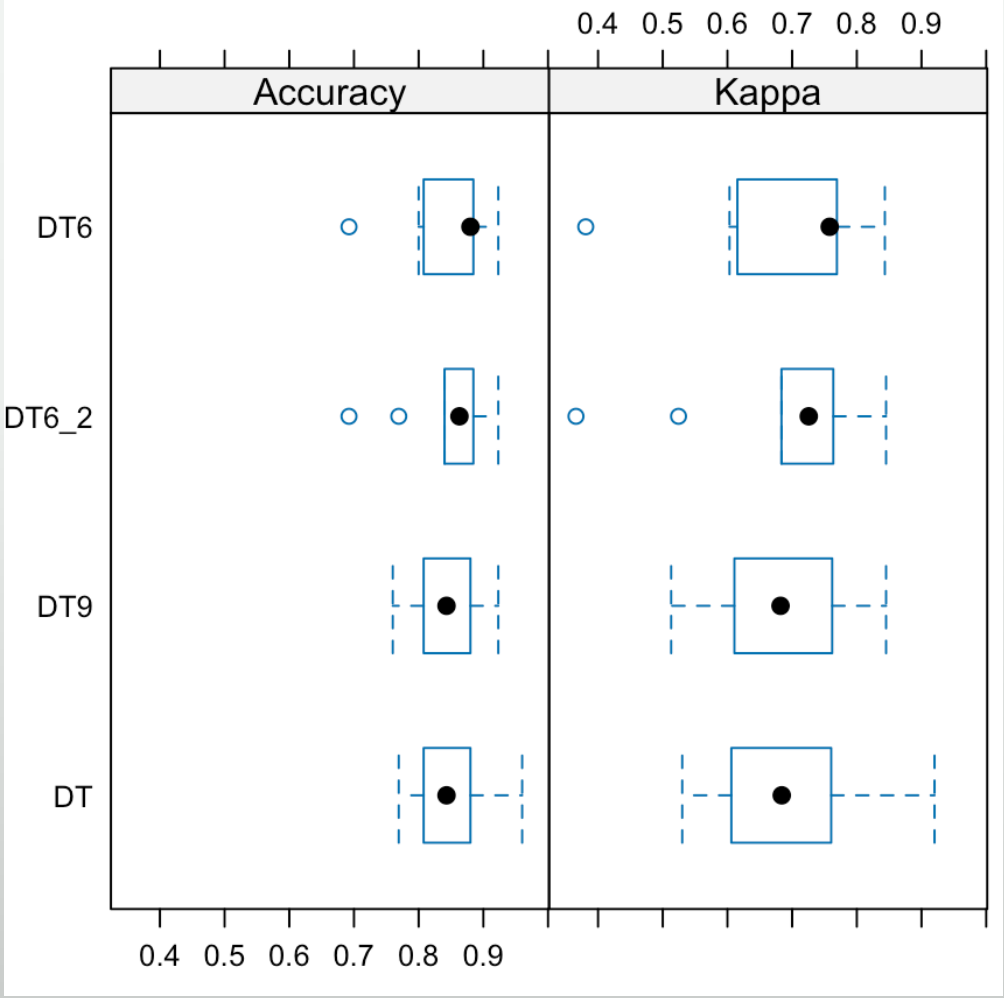
```
## CART
##
## 257 samples
##  13 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 232, 231, 232, 231, 231, 231, ...
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.00000000  0.8410769  0.6788024
##   0.05462963  0.8173846  0.6285750
##   0.10925926  0.7592308  0.5137451
##   0.16388889  0.7592308  0.5137451
##   0.21851852  0.7592308  0.5137451
##   0.27314815  0.7592308  0.5137451
##   0.32777778  0.7592308  0.5137451
##   0.38240741  0.7592308  0.5137451
##   0.43703704  0.7592308  0.5137451
##   0.49166667  0.6343077  0.2289262
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.
```

# Modeling Techniques (3) Model Evaluation

**Decision Tree**

**Comparison of four Decision Tree models:**

| Model | Accuracy | Kappa |
|---|---|---|
| DT | 0.84108 | 0.67880 |
| DT9 | 0.83646 | 0.67049 |
| DT6 | 0.84831 | 0.69523 |
| DT6_2 | 0.84108 | 0.67880 |



# Modeling Techniques (2) Model Training

SVM

## (1) Model with all predictors (SVM)

```
train_index <- createFolds(train_data_k_s$output, k = 10)
svmFit <- train_data_k_s %>% train(output ~., # using all predictors
method = "svmLinear",
data = .,
tuneLength = 10,
trControl = trainControl(method = "cv", indexOut = train_index))
svmFit
```

```
## Support Vector Machines with Linear Kernel
##
## 257 samples
##  13 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 231, 231, 231, 232, 232, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8673846  0.732203
##
## Tuning parameter 'C' was held constant at a value of 1
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 87
##
## Objective Function Value : -75.2803
## Training error : 0.124514
```

| Accuracy | Kappa |
|---|---|
| 0.86738 | 0.73220 |

# Modeling Techniques (2) Model Training

SVM

## (2) Model with nine predictors (SVM9)

`"age", "sex", "exng", "caa","cp","thalachh","slp", "oldpeak","thall"`

```
svmFit9 <- train_data_k_s %>% train(output ~ age + sex + exng + caa + cp + thalachh
+ slp + oldpeak + thall, # using six selected predictors
method = "svmLinear",
data = .,
tuneLength = 10,
trControl = trainControl(method = "cv", indexOut = train_index))
svmFit9
```

```
## Support Vector Machines with Linear Kernel
##
## 257 samples
##   9 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 231, 231, 231, 232, 231, 232, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8749231  0.7475304
##
## Tuning parameter 'C' was held constant at a value of 1
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 91
##
## Objective Function Value : -81.774
## Training error : 0.124514
```

| **Accuracy** | **Kappa** |
|---|---|
| 0.87492 | 0.74753 |

# Modeling Techniques (2) Model Training

## (3) Model with six predictors (SVM6)

`"sex", "cp", "thalachh","exng", "caa", "thall"`

```
svmFit6 <- train_data_k_s %>% train(output ~ sex + cp + trtbps + slp + caa + thall,
# using six selected predictors
method = "svmLinear",
data = .,
tuneLength = 10,
trControl = trainControl(method = "cv", indexOut = train_index))
svmFit6
```

```
## Support Vector Machines with Linear Kernel
##
## 257 samples
##   6 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 232, 231, 231, 231, 231, 232, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8481538  0.6947419
##
## Tuning parameter 'C' was held constant at a value of 1
```

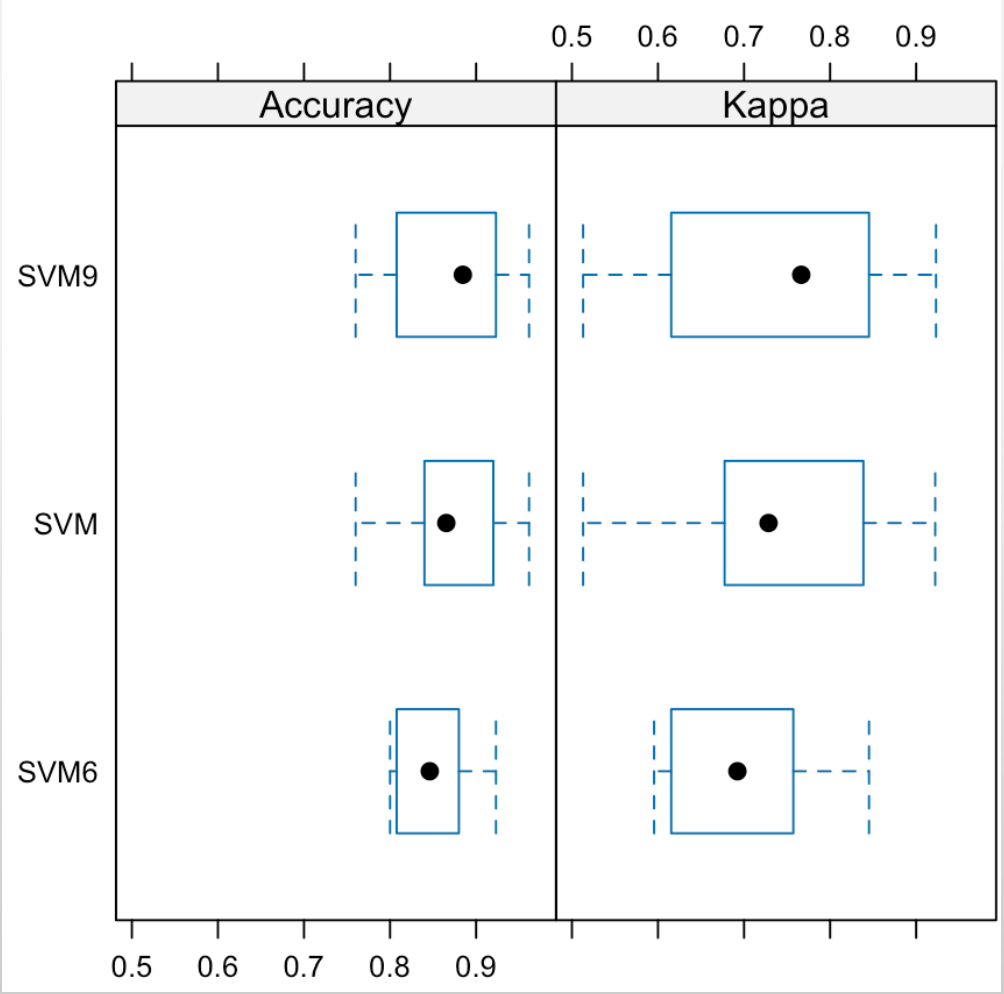
```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Linear (vanilla) kernel function.
##
## Number of Support Vectors : 91
##
## Objective Function Value : -81.774
## Training error : 0.124514
```

| **Accuracy** | **Kappa** |
|---|---|
| 0.84815 | 0.69474 |

# Modeling Techniques (3) Model Evaluation

**Comparison of three SVM models:**

| Model | Accuracy | Kappa |
|---|---|---|
| SVM | 0.86738 | 0.73220 |
| SVM9 | 0.87492 | 0.74753 |
| SVM6 | 0.84815 | 0.69474 |



# Modeling Techniques (2) Model Training

## Naïve Bayes

### (1) Model with all predictors [excluding ('restecg', 'caa', 'thall') ]

```
suppressMessages({ suppressWarnings({
NBayesFit <- train_data_k_n %>% train(output ~ .,
method = "nb",
data = .,
trControl = trainControl(
    method = "cv", # used for configuring resampling method: in this case cross validation
    number = 10,
    index = createFolds(train_data_k_n$output, k = 10),
    #summaryFunction = twoClassSummary,
    classProbs = TRUE,
    verboseIter = TRUE,
    savePredictions = TRUE
  ),
tuneLength = 10,
tuneGrid = expand.grid(
    fL = 1:5,
    usekernel = c(TRUE, FALSE),
    adjust = 1:3
  )
)
})})
```

```
## Naive Bayes
##
## 257 samples
##  10 predictor
##   2 classes: 'NH', 'H'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 25, 25, 26, 26, 26, 25, ...
## Resampling results across tuning parameters:
##
##   fL  usekernel  adjust  Accuracy   Kappa
##   1   FALSE      1       0.7532468  0.5041238
##   1   FALSE      2       0.7532468  0.5041238
##   1   FALSE      3       0.7532468  0.5041238
##   1    TRUE      1       0.7103411  0.4193265
##   1    TRUE      2       0.7267652  0.4510198
##   1    TRUE      3       0.7025601  0.3958767
##   2   FALSE      1       0.7532468  0.5041238
##   2   FALSE      2       0.7532468  0.5041238
##   2   FALSE      3       0.7532468  0.5041238
##   2    TRUE      1       0.7103411  0.4193265
##   2    TRUE      2       0.7267652  0.4510198
##   2    TRUE      3       0.7025601  0.3958767
##   3   FALSE      1       0.7532468  0.5041238
##   3   FALSE      2       0.7532468  0.5041238
##   3   FALSE      3       0.7532468  0.5041238
##   3    TRUE      1       0.7103411  0.4193265
##   3    TRUE      2       0.7267652  0.4510198
##   3    TRUE      3       0.7025601  0.3958767
##   4   FALSE      1       0.7532468  0.5041238
##   4   FALSE      2       0.7532468  0.5041238
##   4   FALSE      3       0.7532468  0.5041238
##   4    TRUE      1       0.7103411  0.4193265
##   4    TRUE      2       0.7267652  0.4510198
##   4    TRUE      3       0.7025601  0.3958767
##   5   FALSE      1       0.7532468  0.5041238
##   5   FALSE      2       0.7532468  0.5041238
##   5   FALSE      3       0.7532468  0.5041238
##   5    TRUE      1       0.7103411  0.4193265
##   5    TRUE      2       0.7267652  0.4510198
##   5    TRUE      3       0.7025601  0.3958767
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 1, usekernel = FALSE and adjust
##  = 1.
```

**(Notes on dropping nominal variables with rare levels:**
We have tried fitting into all feature in initial attempt of Naive Bayes Classifier. However, the training set does not contain observations for all values in some nominal variables. They are **'restecg'**, **'caa'** and **'thall'**. And we got error messages, indicating that each of the variables have a very rare level (or more), in such case, the three variables are dropped.)

# Modeling Techniques (2) Model Training

## Naïve Bayes

### (2) Model with 6 predictors [excluding ('restecg', 'caa', 'thall')] 'sex','cp','trtbps','slp'

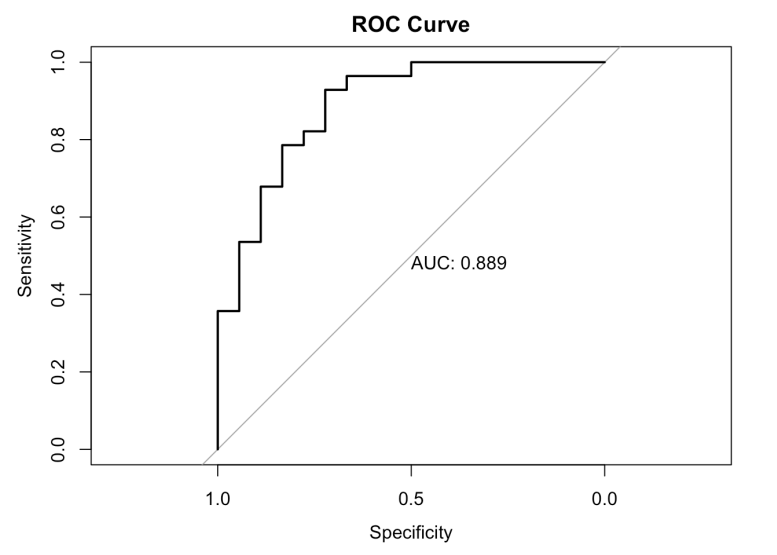
```
suppressMessages({ suppressWarnings({
NBayesFit_r <- train_data_k_n_reduced %>% train(output ~ .,
method = "nb",
data = .,
trControl = trainControl(
    method = "cv", # used for configuring resampling method: in this case cross validation
    number = 10,
    index = createFolds(train_data_k_n_reduced$output, k = 10),
    #summaryFunction = twoClassSummary,
    classProbs = TRUE,
    verboseIter = TRUE,
    savePredictions = TRUE
  ),
tuneLength = 10,
tuneGrid = expand.grid(
    fL = 1:5,
    usekernel = c(TRUE, FALSE),
    adjust = 1:3
  )
)
})})
```

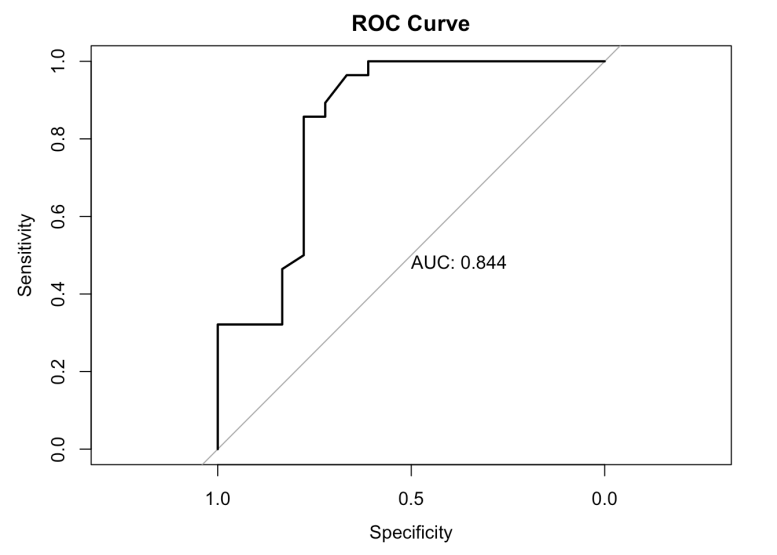
```
## Naive Bayes
##
## 257 samples
##   4 predictor
##   2 classes: 'NH', 'H'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 26, 25, 25, 26, 26, 26, ...
## Resampling results across tuning parameters:
##
##   fL  usekernel  adjust  Accuracy   Kappa
##   1   FALSE      1       0.7284483  0.4453552
##   1   FALSE      2       0.7284483  0.4453552
##   1   FALSE      3       0.7284483  0.4453552
##   1    TRUE      1       0.6610800  0.3325641
##   1    TRUE      2       0.6783307  0.3629579
##   1    TRUE      3       0.6419391  0.2813974
##   2   FALSE      1       0.7284483  0.4453552
##   2   FALSE      2       0.7284483  0.4453552
##   2   FALSE      3       0.7284483  0.4453552
##   2    TRUE      1       0.6610800  0.3325641
##   2    TRUE      2       0.6783307  0.3629579
##   2    TRUE      3       0.6419391  0.2813974
##   3   FALSE      1       0.7284483  0.4453552
##   3   FALSE      2       0.7284483  0.4453552
##   3   FALSE      3       0.7284483  0.4453552
##   3    TRUE      1       0.6610800  0.3325641
##   3    TRUE      2       0.6783307  0.3629579
##   3    TRUE      3       0.6419391  0.2813974
##   4   FALSE      1       0.7284483  0.4453552
##   4   FALSE      2       0.7284483  0.4453552
##   4   FALSE      3       0.7284483  0.4453552
##   4    TRUE      1       0.6610800  0.3325641
##   4    TRUE      2       0.6783307  0.3629579
##   4    TRUE      3       0.6419391  0.2813974
##   5   FALSE      1       0.7284483  0.4453552
##   5   FALSE      2       0.7284483  0.4453552
##   5   FALSE      3       0.7284483  0.4453552
##   5    TRUE      1       0.6610800  0.3325641
##   5    TRUE      2       0.6783307  0.3629579
##   5    TRUE      3       0.6419391  0.2813974
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were fL = 1, usekernel = FALSE and adjust
##  = 1.
```

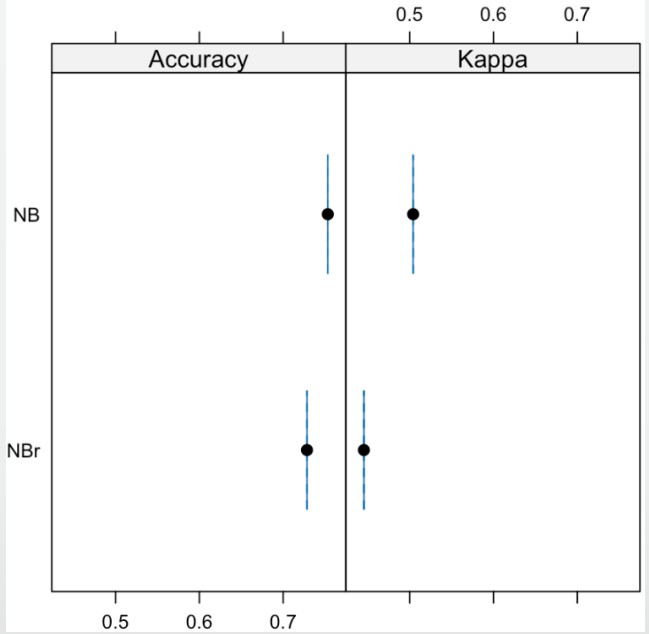
**(Notes on dropping nominal variables with rare levels:**
We have tried fitting into all feature in initial attempt of Naive Bayes Classifier. However, the training set does not contain observations for all values in some nominal variables. They are **'restecg'**, **'caa'** and **'thall'**. And we got error messages, indicating that each of the variables have a very rare level (or more), in such case, the three variables are dropped.)

# Modeling Techniques (3) Model Evaluation

## Naïve Bayes




| Model | Accuracy | Kappa |
|---|---|---|
| NB | 0.75325 | 0.50412 |
| NBr | 0.72845 | 0.44536 |

# Modeling Techniques (2) Model Training

## Logistic Regression

### (1) Model with all predictors (LR)

```
#create model
model_LG <- glm(output ~ ., data = train_data_k, family = "binomial")
#evaluate model
predicted_output <- predict(model_LG, newdata = test_data_k, type = "response")
predicted_labels <- ifelse(predicted_output > 0.5, 1, 0)
accuracy <- mean(predicted_labels == test_data_k$output)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.869565217391304"
```

### (2) Model with nine predictors (LR9)

```
#create model
model_LG9 <- glm(output ~ age + sex + exng + caa + cp +
                 thalachh + slp + oldpeak + thall, data = train_data_k, family = "binomial")
#evaluate model
predicted_output_9 <- predict(model_LG9, newdata = test_data_k, type = "response")
predicted_labels_9 <- ifelse(predicted_output_9 > 0.5, 1, 0)
accuracy_9 <- mean(predicted_labels_9 == test_data_k$output)
print(paste("Accuracy:", accuracy_9))
```

```
## [1] "Accuracy: 0.891304347826087"
```

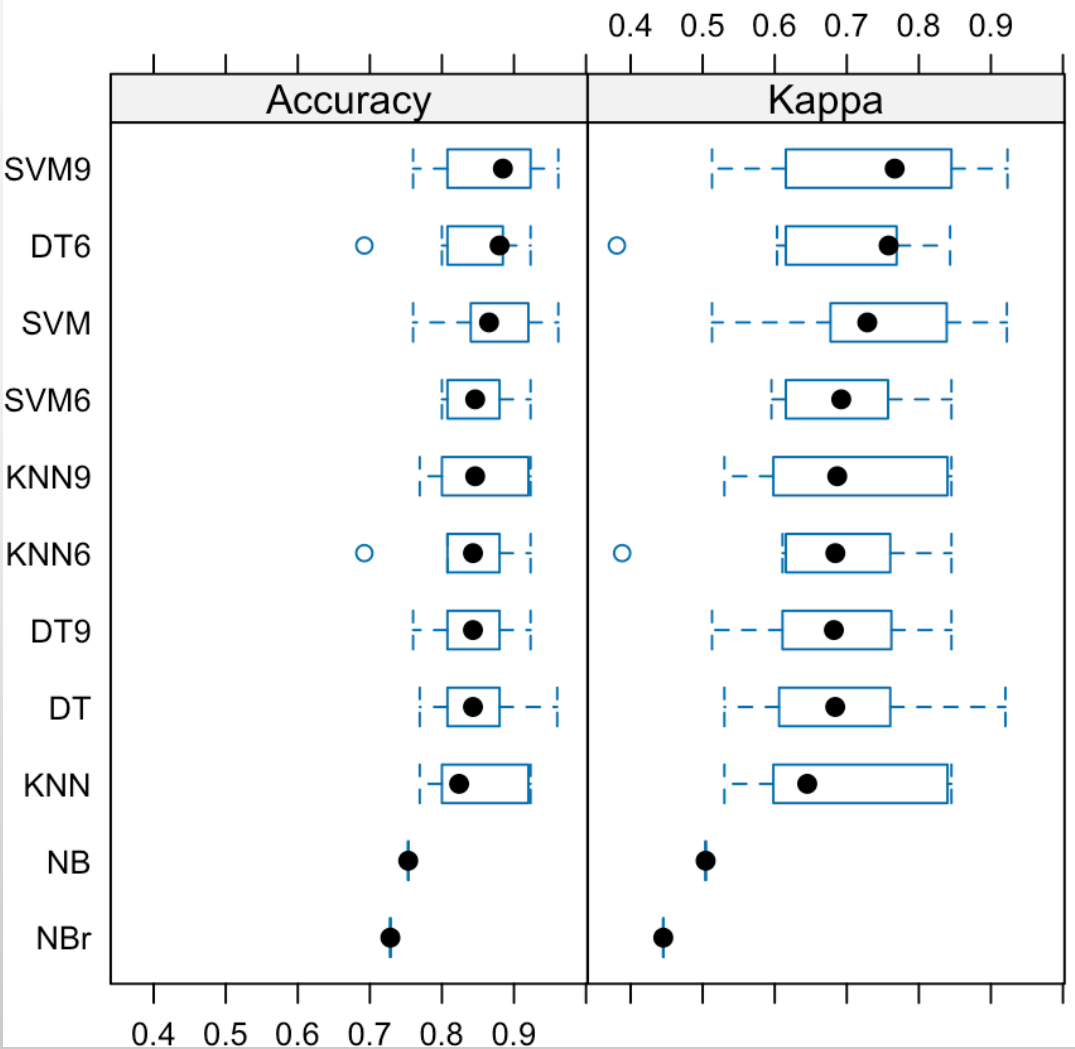
### (3) Model with six predictors (LR6)

```
model_bw <-glm(output ~  sex + cp + trtbps + slp + caa + thall, data = train_data_k, family = "binomial")
predicted_bw <- predict(model_bw, newdata = test_data_k, type = "response")
predictedbw_labels <- ifelse(predicted_bw > 0.5, 1, 0)
accuracy_bw <- mean(predictedbw_labels == test_data_k$output)
print(paste("Accuracy:", accuracy_bw))
```

```
## [1] "Accuracy: 0.891304347826087"
```

(Note: ANOVA can be added to further evaluate the regression models.)

# Model Comparison




- Among the classifiers, **SVM using 9 selected features** work best;
- On the other hand, **Logistics Regression using 6 selected features** also work well.
- We put them both to Test Data to decide which is our *FINAL SELECTED MODEL*.

# Final Model Validation

## (1) SVM using 9 selected features

```
predicted_svm9 <- predict(svmFit9, newdata = test_data_k_s)
accuracy_svm9 <- mean(predicted_svm9 == test_data_k_s$output)
print(paste("Accuracy:", accuracy_svm9))
```

```
## [1] "Accuracy: 0.869565217391304"
```

## (2) Logistics Regression with 6 selected features

```
predicted_bw <- predict(model_bw, newdata = test_data_k)
predictedbw_labels <- ifelse(predicted_bw > 0.5, 1, 0)
accuracy_bw <- mean(predictedbw_labels == test_data_k$output)
print(paste("Accuracy:", accuracy_bw))
```

```
## [1] "Accuracy: 0.891304347826087"
```

# Conclusion

- Among 13 variables, we selected 2 subsets for model building based on:
  - (1) Correlation with target variable (9 selected)
  - (2) Backward subset selection (6 selected)

- We tried both classification and regression methods on selected predictors to build prediction models, validated each model with cross-validation, and compared them by accuracy and kappa values. Methods including:
  - Classification: KNN, Decision Tree, SVM, Naïve Bayes
  - Regression: Logistic regression

- We shortlisted two best candidates: (1) SVM with 9 selected features, and (2) Logistics Regression with 6 selected features.

- Finally, we tested the two finalists on the test data and found **Logistics Regression with 6 selected features** gives the best prediction.

# Thank you

Q & A

# Appendix I: Dataset description

- ***Heart.csv* : 303 objects, 14 attributes**
  - **age [int.]** : Age of the patient
  - **sex [cat.]** : Sex of the patient (1 = male; 0 = female)
  - **exng [cat.]** : exercise induced angina (1 = yes; 0 = no)
  - **caa [cat.]**: number of major vessels (0-3)
  - **cp [cat.]** : Chest Pain type
    - 1 - typical angina
    - 2 - atypical angina
    - 3 - non-anginal pain
    - 4 - asymptomatic
  - **trtbps [cont.]** : resting blood pressure (in mmHg)
  - **chol [cont.]** : cholestoral in mg/dl fetched via BMI sensor
  - **fbs [cat.]** : fasting blood sugar (> 120 mg/dl) (1 = true; 0 = false)
  - **restecg [cat.]** : resting electrocardiographic results
    - 0 - normal
    - 1 - having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV)
    - 2 - showing probable or definite left ventricular hypertrophy by Estes' criteria
  - **thalachh [int.]** : maximum heart rate achieved
  - **slp [cat.]** : The slope of the peak exercise ST segment (1 = upsloping; 2 = flat; 3 = downsloping)
  - **oldpeak [cont.] :** ST depression induced by exercise relative to rest
  - **thall [cat.]** : Thal rate
  - **output [cat.]** : target (0= less chance of heart attack; 1= more chance of heart attack)



Source: https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset/data