

Heart disease prediction using SVM

Anish Singh Walia

11 march 2018

Prediction of heart disease

Aim of analysis

In the following document, I will be using SVM classification technique to predict heart disease (angiographic disease status). From a set of 14 variables, the most important to predict heart failure are whether or not there is a reversible defect in Thalassemia followed by whether or not there is an occurrence of asymptomatic chest pain.

Dataset:

The heart disease data are available at UCI The description of the database can be found [here](#).

let's read the dataset from the URL in R.

```
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```
require(pROC) #to plot the ROC curves
```

```
## Loading required package: pROC
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
```

```
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
```

```
##
```

```
##      cov, smooth, var
```

```
heartdf <- read.csv("https://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/processed.")
```

```
names(heartdf) <- c("age", "sex", "cp", "trestbps", "chol", "fbs", "restecg",  
                    "thalach", "exang", "oldpeak", "slope", "ca", "thal", "num")
```

```
attach(heartdf)
```

The variable we want to predict is num with Value 0: < 50% diameter narrowing and Value 1: > 50% diameter narrowing. We assume that every value with 0 means heart is okay, and 1,2,3,4 means heart disease.

From the possible values the variables can take, it is evident that the following need to be dumified because the distances in the values is random: cp,thal, restecg, slope

Let's get a quick idea of data

```
head(heartdf,3)
```

```
##   age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal  
## 1  63  1  1    145  233   1         2    150    0     2.3    3  0    6
```

```
## 2 67 1 4 160 286 0 2 108 1 1.5 2 3 3
## 3 67 1 4 120 229 0 2 129 1 2.6 2 2 7
## num
## 1 0
## 2 2
## 3 1
```

```
dim(heartdf) # dimensions of the dataset
```

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

Let's explore the data and find how many had heart attacks, women or men have of a particular age?

Let's first convert the dependent(class variable) Y – *num* to binary variable.

```
#converting the num variable to binary class variable
```

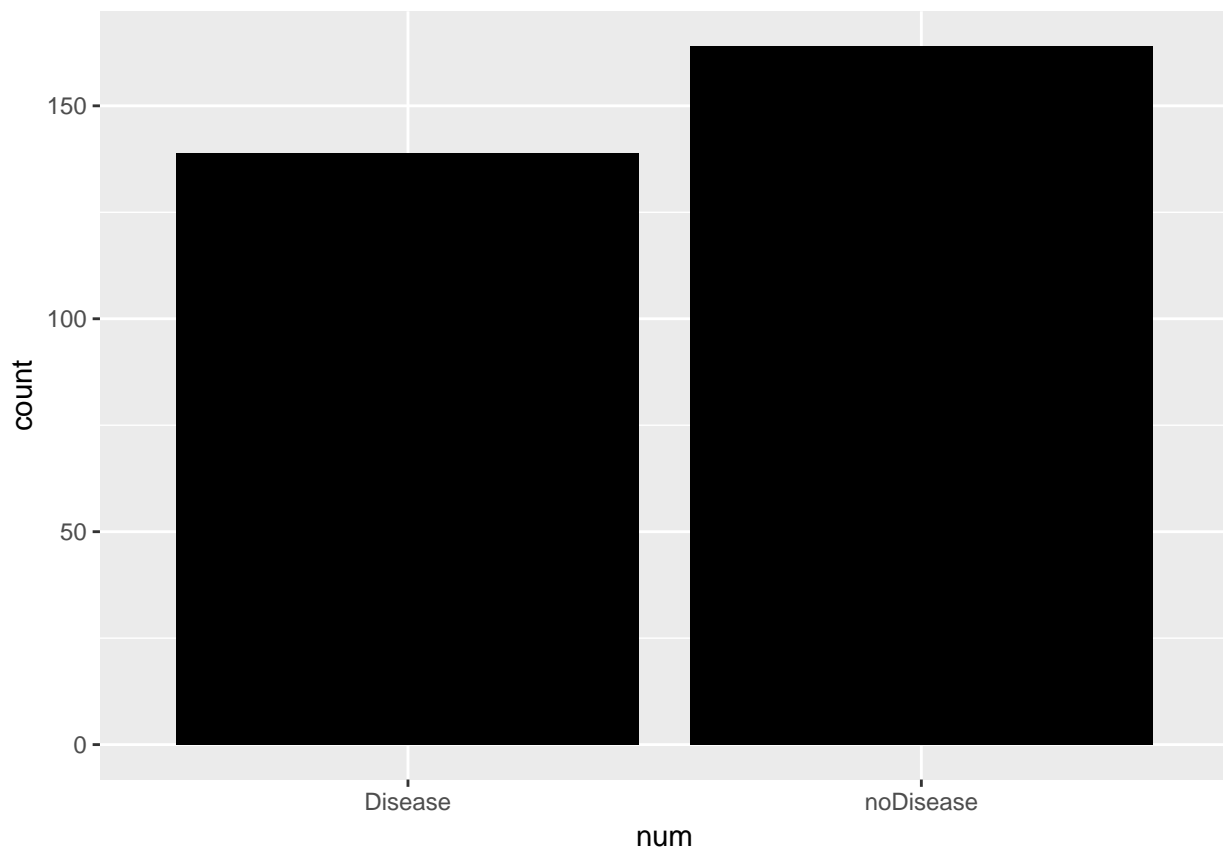
```
heartdf$num<-ifelse(heartdf$num > 0,"Disease","noDisease")
```

```
table(heartdf$num)
```

```
##
## Disease noDisease
##      139      164
```

```
#distribution of the target variable
```

```
ggplot(heartdf,aes(x = num)) +
  geom_bar(fill="black")
```



```
#converting to factor variable
heartdf$sex<-ifelse(heartdf$sex==0,"female","male")
```

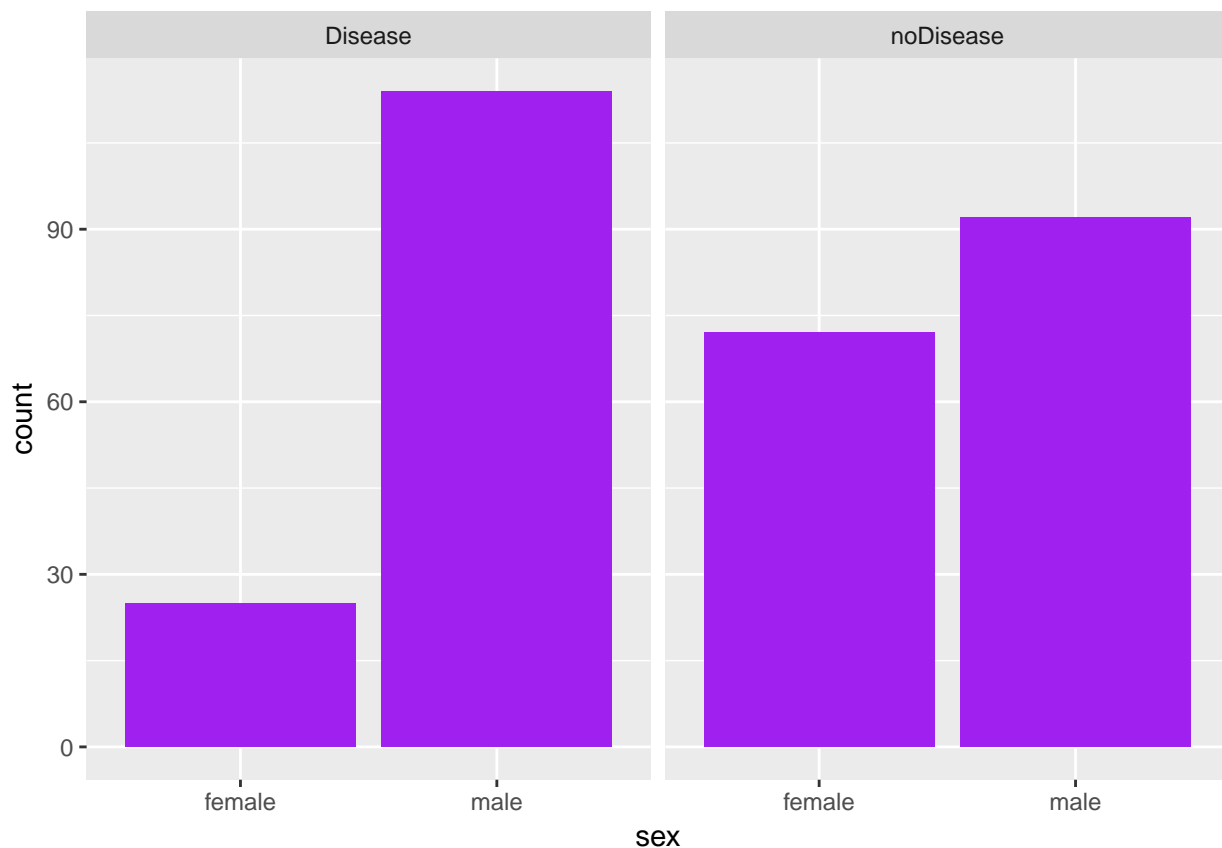
```
table(heartdf$sex)
```

```
##
## female    male
##      97     206
```

```
table(sex=heartdf$sex,disease=heartdf$num)
```

```
##           disease
## sex      Disease noDisease
## female      25         72
## male       114         92
```

```
ggplot(heartdf,aes(x=sex)) +
  geom_bar(fill="purple") +
  facet_wrap(~num)
```



```
#heart disease and age
#making a box plot to unerstand the statistical distribution
```

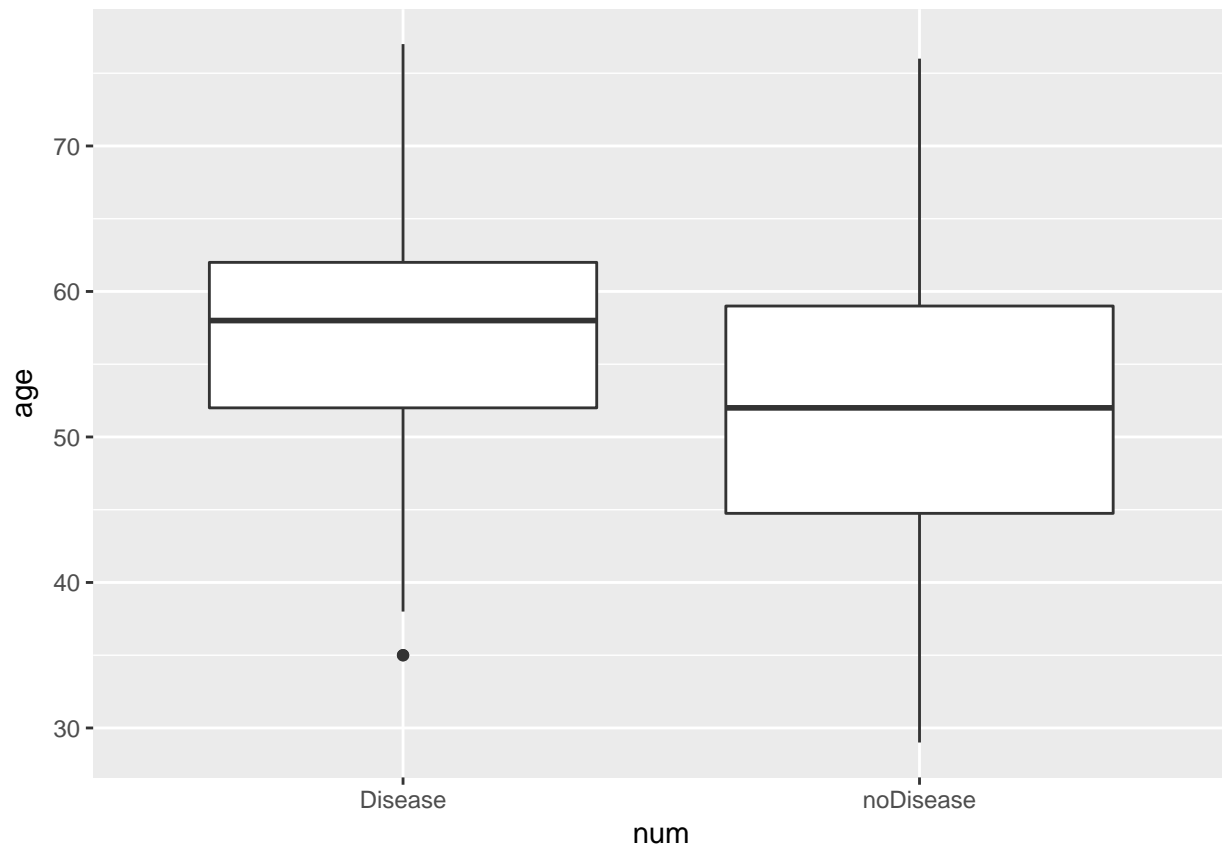
```
by(heartdf$age,heartdf$num,summary)
```

```
## heartdf$num: Disease
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  35.00  52.00   58.00  56.63  62.00   77.00
```

```
## -----
## heartdf$num: noDisease
##   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   29.00  44.75   52.00   52.59  59.00   76.00
```

So people who had heart disease for them the mean age is 56.6

```
ggplot(heartdf,aes(x = num,y = age)) +
  geom_boxplot()
```



Let's do some correlation analysis between some variables-

```
cor.test(age,chol) #very low correlation
```

```
##
## Pearson's product-moment correlation
##
## data: age and chol
## t = 3.707, df = 301, p-value = 0.0002496
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.09859353 0.31423005
## sample estimates:
##      cor
## 0.2089503
```

We can see that age and cholesterol levels have very low correlation.

```
#confusion matrix of chest pain and heart disease
table(cp,num)
```

```
##      num
## cp   0  1  2  3  4
##   1 16  5  1  0  1
##   2 41  6  1  2  0
##   3 68  9  4  4  1
##   4 39 35 30 29 11
```

```
#confusion matrix of exercise induced asthma and heart disease
table(exang,num)
```

```
##      num
## exang  0   1   2   3   4
##   0 141  30  14  12   7
##   1  23  25  22  23   6
```

We can notice from the table that people who had heart diseases had severe level of chest pain. Also people who had heart diseases had exercise induced asthma.

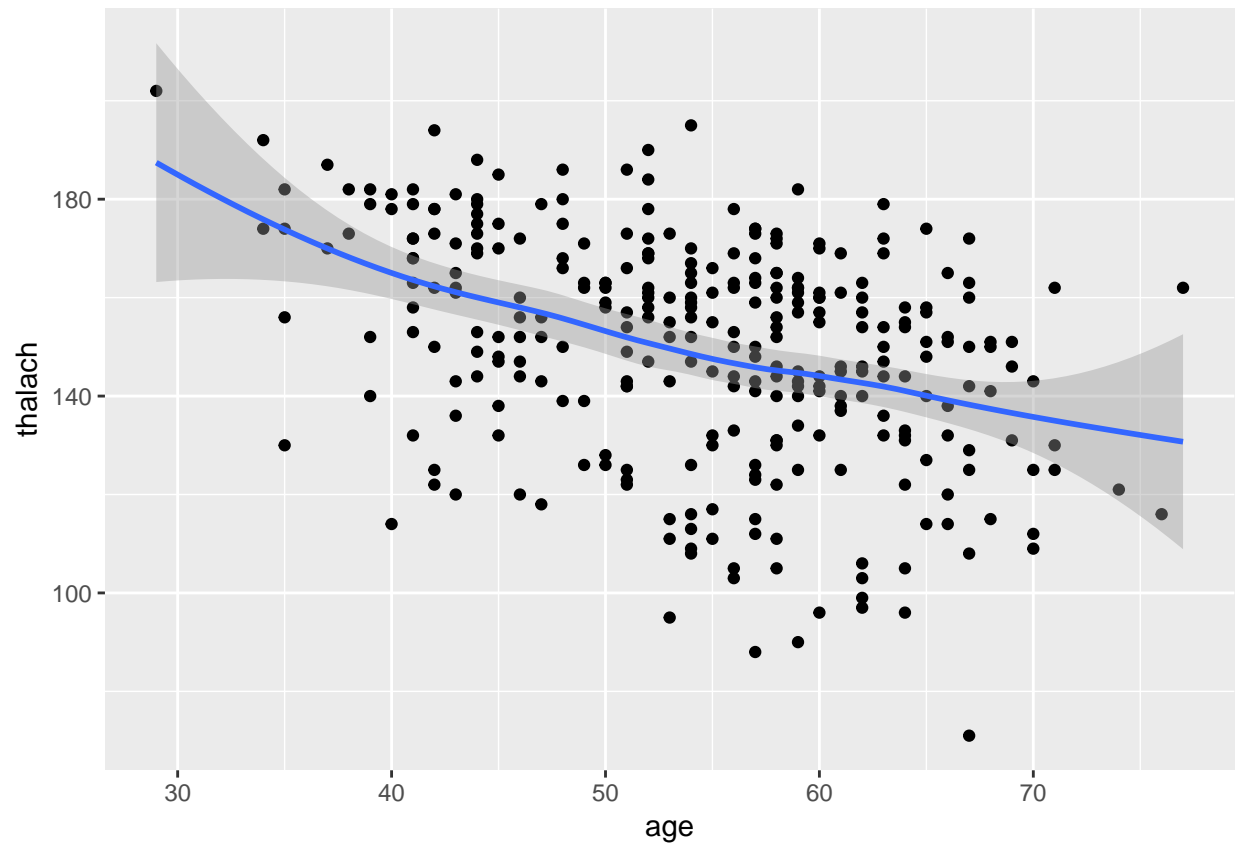
Correlation between age and maximum heart rate achieved-

```
cor.test(age,thalach)
```

```
##
## Pearson's product-moment correlation
##
## data: age and thalach
## t = -7.4329, df = 301, p-value = 1.109e-12
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
## -0.4849644 -0.2941816
## sample estimates:
##      cor
## -0.3938058
```

```
ggplot(heartdf,aes(x = age,y = thalach )) +
  geom_point() +
  geom_smooth()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



We can notice that as age increase maximum heart rate achieved decreases, as the correlation is negative.

predictive Modelling

let's now predict who is likely to have a heart disease and who is not?

Separating training and testing data

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.4.3
```

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

```
set.seed(20)
```

```
inTrainRows <- createDataPartition(heartdf$num, p=0.7, list=FALSE)
```

```
trainData <- heartdf[inTrainRows,]
```

```
testData <- heartdf[-inTrainRows,]
```

```
nrow(trainData)/(nrow(testData)+nrow(trainData)) #checking whether really 70% -> OK
```

```
## [1] 0.7029703
```

Building a SVM classifier

Now SVM classifier tends to generate hyperplanes which separate the classes with maximum margins i.e in simpler terms it aims to generate maximum marginal hyperplane.

So amongst a set of competing hypothesis $h_i(x)$ we want to choose the one which maximizes the margin between both the classes on either side of the separating hyperplane, i.e our main goal is to separate the classes on either sides of hyperplane with **maximum margin** using the *support vectors*(which help us define the maximum margins).

So a linear SVM classifier will generate a simple linear hyperplane for linearly separable data.

```
# for this to work add names to all levels (numbers not allowed)
feature.names=names(heartdf)
```

```
for (f in feature.names) {
  if (class(heartdf[[f]])=="factor") {
    levels <- unique(c(heartdf[[f]]))
    heartdf[[f]] <- factor(heartdf[[f]],
                          labels=make.names(levels))
  }
}
```

```
#converting to factor variable with 2 levels
heartdf$num<-as.factor(heartdf$num)
levels(heartdf$num) <- c("Notdisease", "Disease")
```

```
table(heartdf$num)
```

```
##
## Notdisease      Disease
##           139         164
```

```
set.seed(10)
```

```
inTrainRows <- createDataPartition(heartdf$num,p=0.7,list=FALSE)
trainData2 <- heartdf[inTrainRows,]
testData2 <- heartdf[-inTrainRows,]
```

```
#cross validation
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           ## Estimate class probabilities
                           classProbs = TRUE,
                           ## Evaluate performance using
                           ## the following function
                           summaryFunction = twoClassSummary)
```

```
svmModel <- train(num ~ ., data = na.omit(trainData2),
                  method = "svmRadial",
```

```

trControl = fitControl,
preProcess = c("center", "scale"),
tuneLength = 8,
metric = "ROC")

svmModel

## Support Vector Machines with Radial Basis Function Kernel
##
## 209 samples
## 13 predictor
## 2 classes: 'Notdisease', 'Disease'
##
## Pre-processing: centered (13), scaled (13)
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 188, 189, 189, 187, 189, 188, ...
## Resampling results across tuning parameters:
##
## C      ROC      Sens      Spec
## 0.25  0.8903140  0.7963333  0.8625758
## 0.50  0.8907567  0.7860000  0.8634091
## 1.00  0.8888721  0.7857778  0.8625000
## 2.00  0.8912214  0.7941111  0.8680303
## 4.00  0.8887837  0.7942222  0.8740152
## 8.00  0.8792062  0.7970000  0.8594697
## 16.00 0.8593729  0.7680000  0.8406818
## 32.00 0.8329242  0.7380000  0.7964394
##
## Tuning parameter 'sigma' was held constant at a value of 0.05051126
## ROC was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.05051126 and C = 2.

#prediction on test data-class labels
svmPrediction <- predict(svmModel, testData2)

#probability of no heart disease-finding probabilities value
svmPredictionprob <- predict(svmModel, testData2, type='prob')[2]

#generating a confusion matrix
ConfMatrixPrediction <- confusionMatrix(svmPrediction, na.omit(testData2)$num)

ConfMatrixPrediction$table

##           Reference
## Prediction Notdisease Disease
## Notdisease      32      7
## Disease         9     40

```

In the confusion matrix the diagonals represent the correctly classified examples, whereas the offdiagonals are incorrectly classifier examples.

Let's find the ROC curver and the AUC value to better understand the accuracy and performance-

ROC curve is the plot of True positive rate vs the false positive rate.


```

#ROC and AUC value
AUC<- roc(na.omit(testData2)$num,as.numeric(as.matrix((svmPredictionprob))))$auc

Accuracy<- ConfMatrixPrediction$overall['Accuracy']

svmPerformance<-cbind(AUC,Accuracy)

svmPerformance

```

```

##              AUC  Accuracy
## Accuracy 0.8993254 0.8181818

```

Hence we get an **AUC** value of 0.911 and overall **prediction** accuracy of 0.89