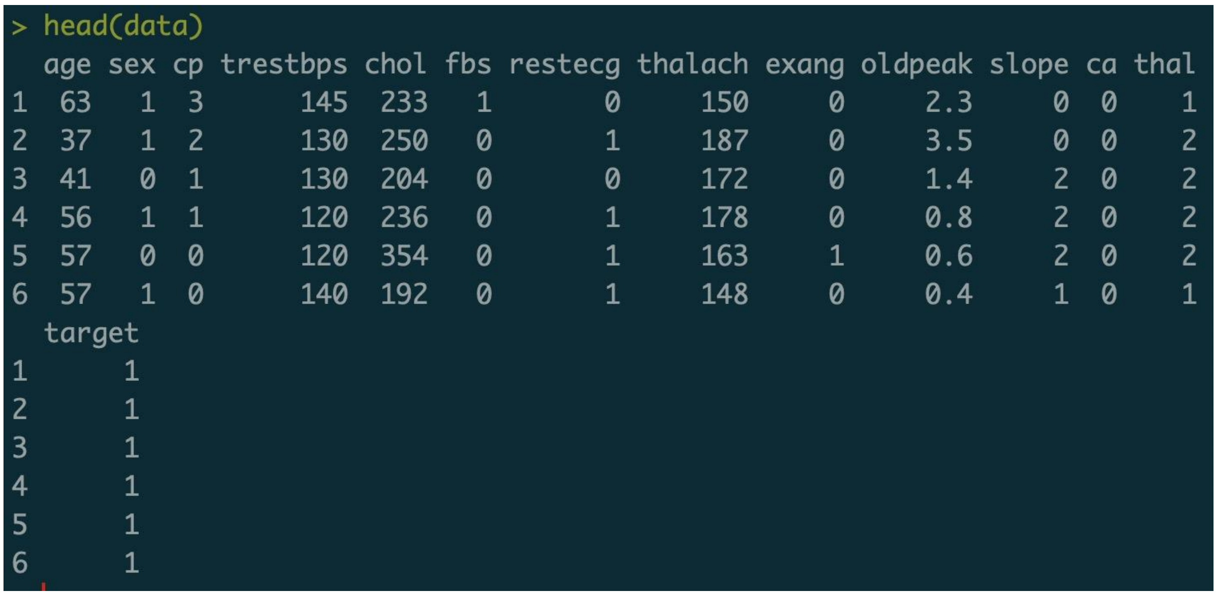
Subject: Big data
Student: Oleksandr Romanchenko
ID: 83459
Project: "Classification of heart diseases using different types of machine learning models with R programming language"

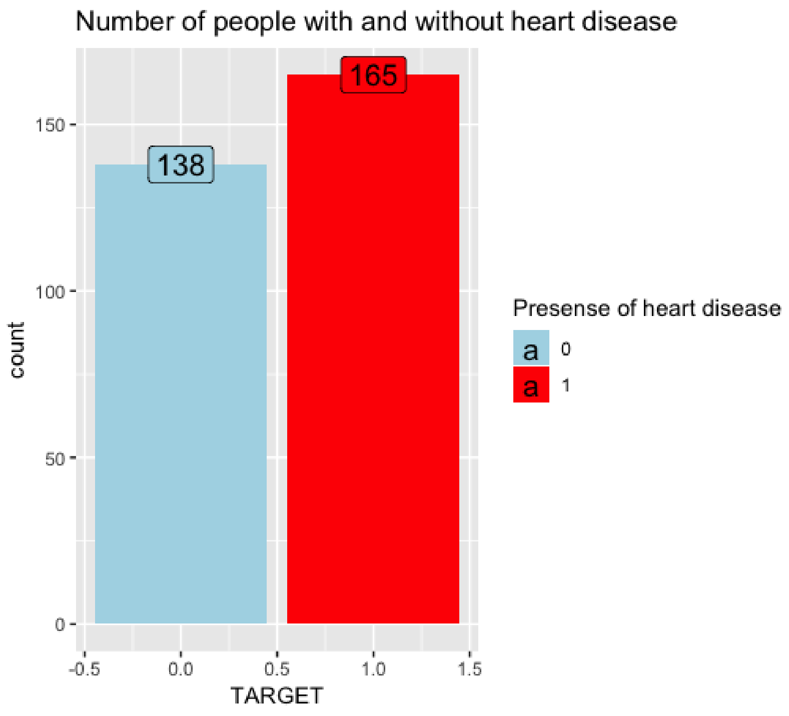
Data: this database contains 76 attributes, but all published experiments refer to using a subset of 14 of them. In particular, the Cleveland database is the only one that has been used by ML researchers to this date. The "goal" field (target) refers to the presence of heart disease in the patient. It is binary value (0 or 1). Data was taken from Kaggle.

Attribute Information:
> 1. **age**
> 2. **sex** (1 – male; 0 - female)
> 3. **cp** - chest pain type (4 values)
> 4. **trestbps** - resting blood pressure (in Hg on admission to the hospital)
> 5. **chol** - serum cholestoral in mg/dl
> 6. **fbs** - fasting blood sugar > 120 mg/dl (1 = true; 0 = false)
> 7. **restecg** - resting electrocardiographic results (values 0, 1, 2)
> 8. **thalach** - maximum heart rate achieved
> 9. **exang** - exercise induced angina (1 = yes; 0 = no)
> 10. **oldpeak** - ST depression induced by exercise relative to rest
> 11. **slope** - the slope of the peak exercise ST segment
> 12. **ca** - number of major vessels (0-3) colored by flourosopy
> 13. **thal** - 3 = normal; 6 = fixed defect; 7 = reversable defect
> 14. **target** – dependent variable (1 = heart disease; 0 = no heart disease)

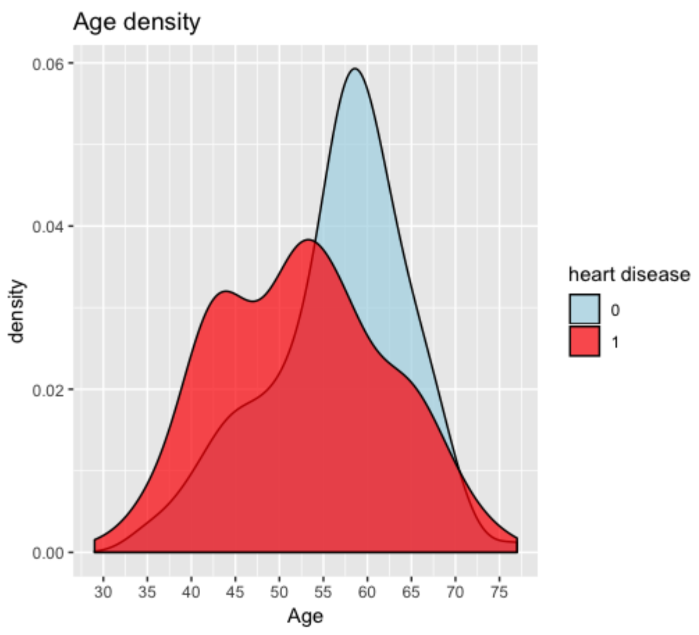


```
> head(data)
  age sex cp trestbps chol fbs restecg thalach exang oldpeak slope ca thal
1  63   1  3      145  233   1       0     150     0     2.3     0  0    1
2  37   1  2      130  250   0       1     187     0     3.5     0  0    2
3  41   0  1      130  204   0       0     172     0     1.4     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
```

Doing some preparations of data and visualizations of main aspects:

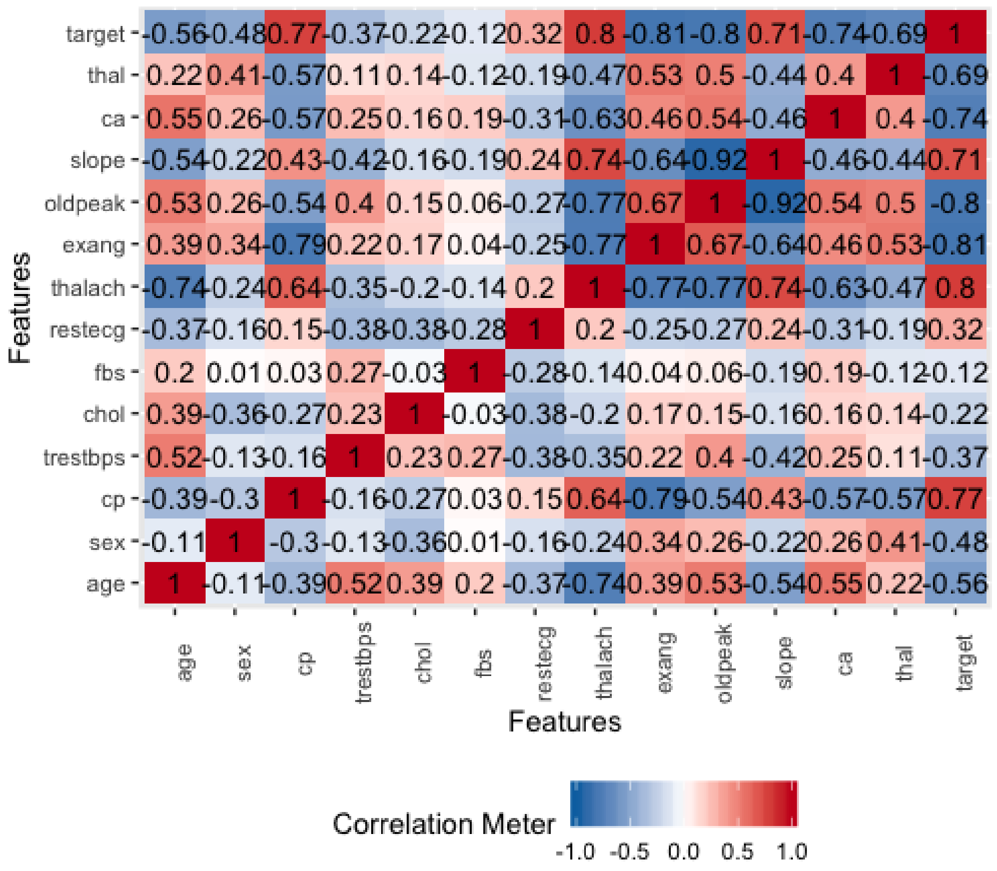


Number of observations in each category is almost equal, it is good for training the model.



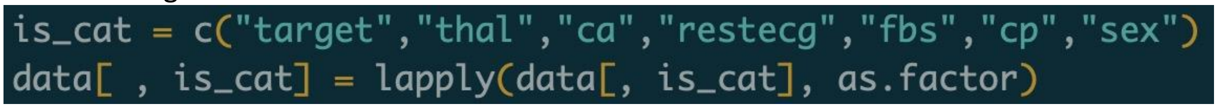
Interesting that according to this density plot the most people with heart disease are between 50 and 55 years old whereas those without illness are significantly concentrated near the age of 60. Is it because heart disease often leads to a fast death (in 2-3 years)? – It is possible. Unfortunately for now it’s difficult to conclude more because it requires more domain knowledge in this field.

Checking correlation plot



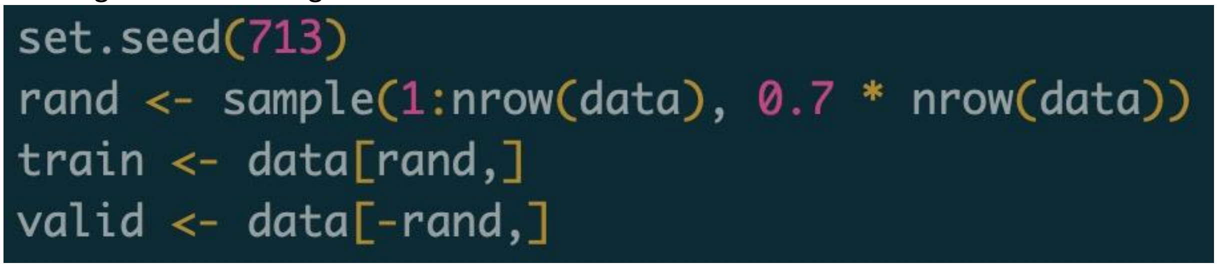
Considering correlation between variables to avoid collinearity and multicollinearity. Deleting variables: *slope*, *oldpeak*, *exang*, *thalach*.

Transforming some columns to factor



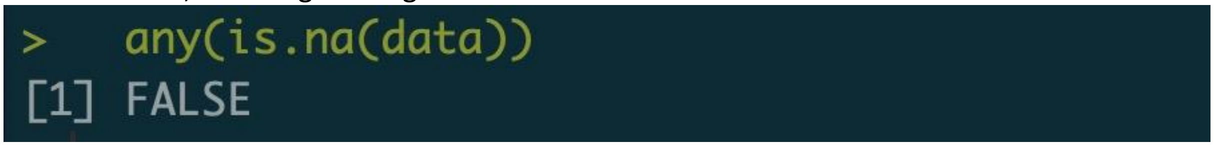
```
is_cat = c("target","thal","ca","restecg","fbs","cp","sex")
data[ , is_cat] = lapply(data[, is_cat], as.factor)
```

Dividing data on training and validation datasets



```
set.seed(713)
rand <- sample(1:nrow(data), 0.7 * nrow(data))
train <- data[rand,]
valid <- data[-rand,]
```

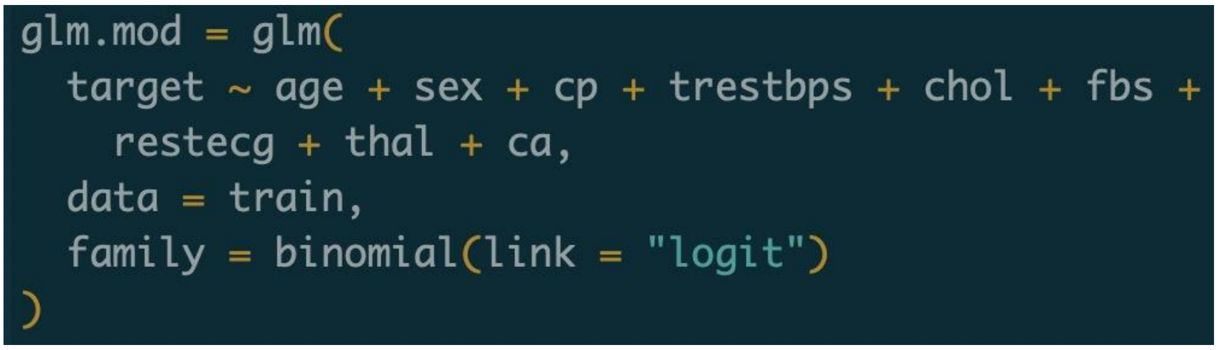
And of course, checking missing values



```
>   any(is.na(data))
[1] FALSE
```

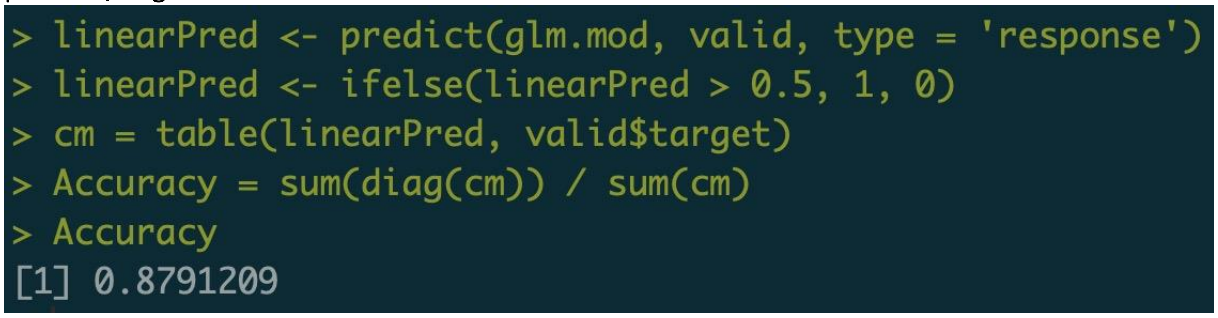
Finally, it’s time to start modelling part! I’m going to predict target variable (whether the person is going to have a heart disease or not).

Logistic regression



```
glm.mod = glm(
  target ~ age + sex + cp + trestbps + chol + fbs +
    restecg + thal + ca,
  data = train,
  family = binomial(link = "logit")
)
```

Then making prediction and assessing accuracy of the model simply dividing number of true positive/negative over number of observations in dataset.

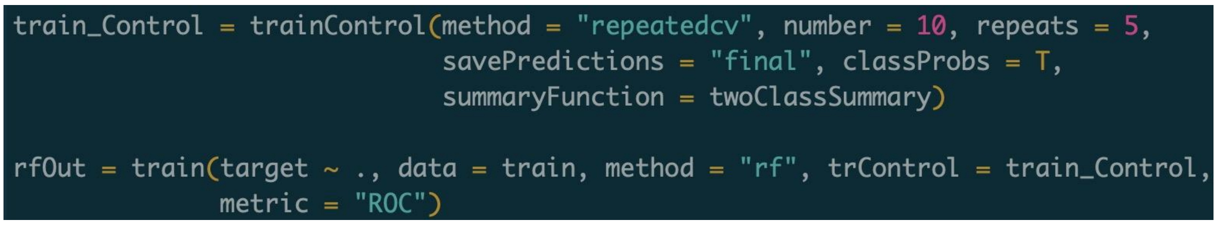


```
> linearPred <- predict(glm.mod, valid, type = 'response')
> linearPred <- ifelse(linearPred > 0.5, 1, 0)
> cm = table(linearPred, valid$target)
> Accuracy = sum(diag(cm)) / sum(cm)
> Accuracy
[1] 0.8791209
```

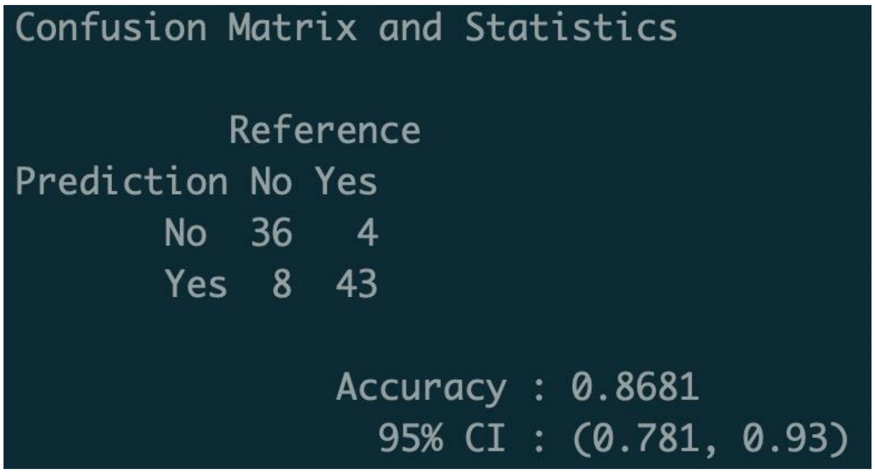
*cm – confusion matrix

Not that bad! Let’s try the next model.

Random forest



```
train_Control = trainControl(method = "repeatedcv", number = 10, repeats = 5,
                             savePredictions = "final", classProbs = T,
                             summaryFunction = twoClassSummary)

rfOut = train(target ~ ., data = train, method = "rf", trControl = train_Control,
              metric = "ROC")
```

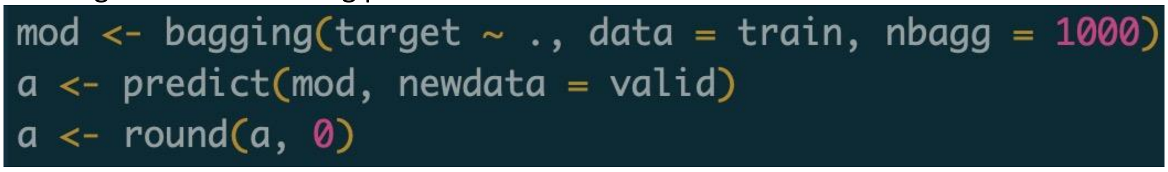


```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  36   4
       Yes  8  43

               Accuracy : 0.8681
                 95% CI : (0.781, 0.93)
```

Random forest gives a bit less accuracy than logistic regression: 0.8681
Let's see what other models can show.

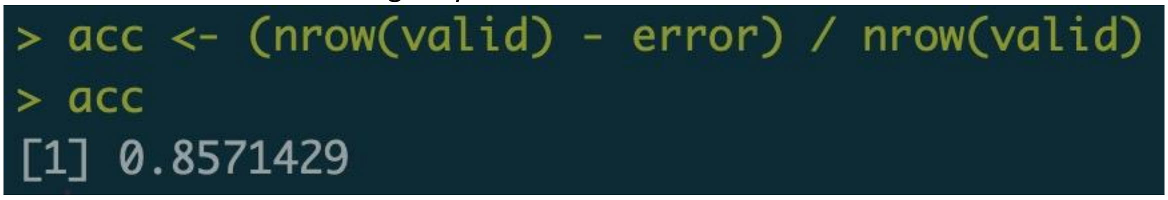
Bagging

Building model and making prediction



```
mod <- bagging(target ~ ., data = train, nbagg = 1000)
a <- predict(mod, newdata = valid)
a <- round(a, 0)
```

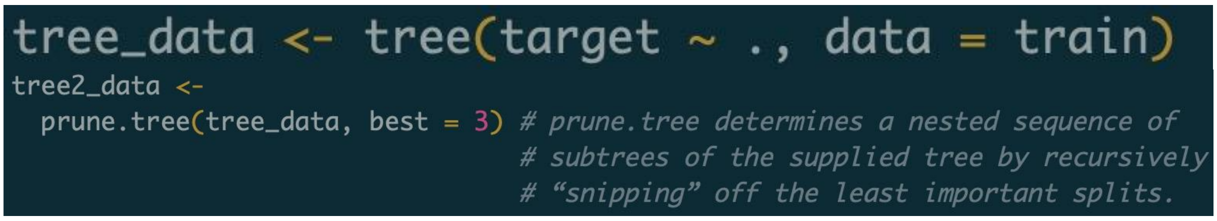
Calculating accuracy by subtracting number of incorrect predictions from number of rows of validation dataset and dividing it by number of rows.



```
> acc <- (nrow(valid) - error) / nrow(valid)
> acc
[1] 0.8571429
```

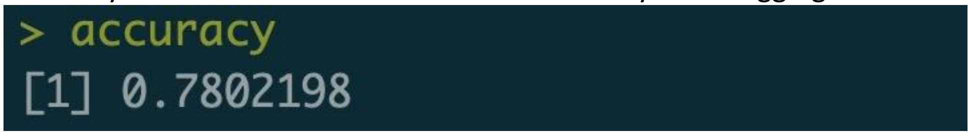
Almost 86%. The worst result so far but not significantly smaller than 2 previous models.

Decision tree



```
tree_data <- tree(target ~ ., data = train)
tree2_data <-
  prune.tree(tree_data, best = 3) # prune.tree determines a nested sequence of
                                  # subtrees of the supplied tree by recursively
                                  # “snipping” off the least important splits.
```

Accuracy and error were calculated the same way as in bagging model.

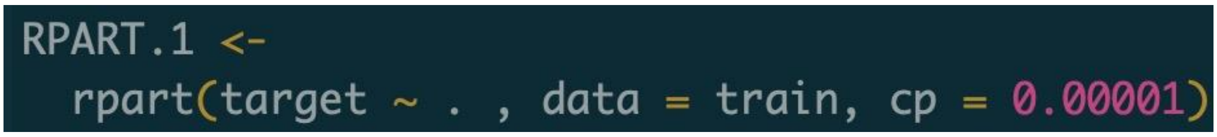


```
> accuracy
[1] 0.7802198
```

78%. Even few percent’s worse than bagging.

Rpart

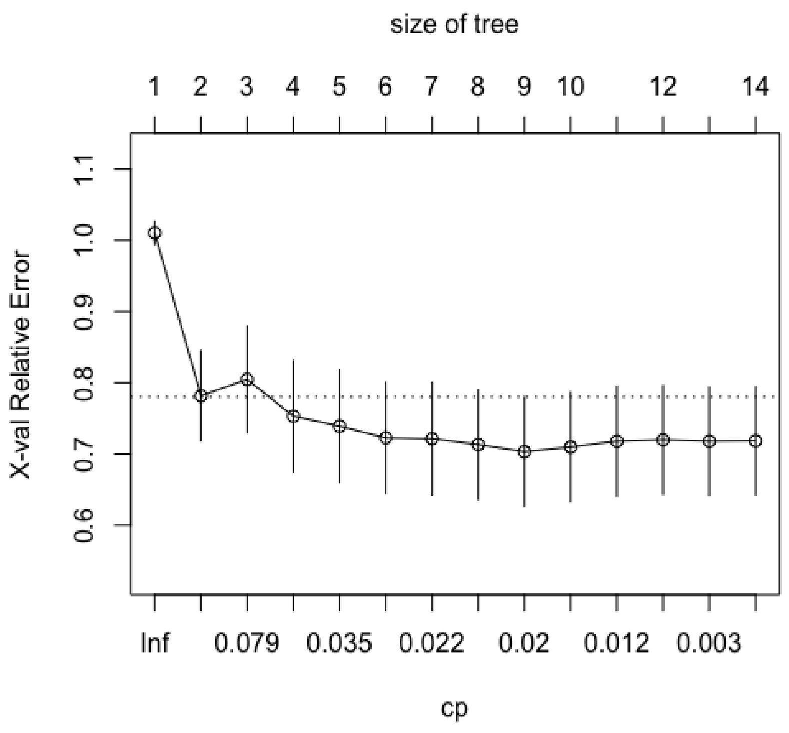
Just one more type of decision tree which allows to create some fancy plots for illustrating the logic of dividing variables at some thresholds inside the tree.



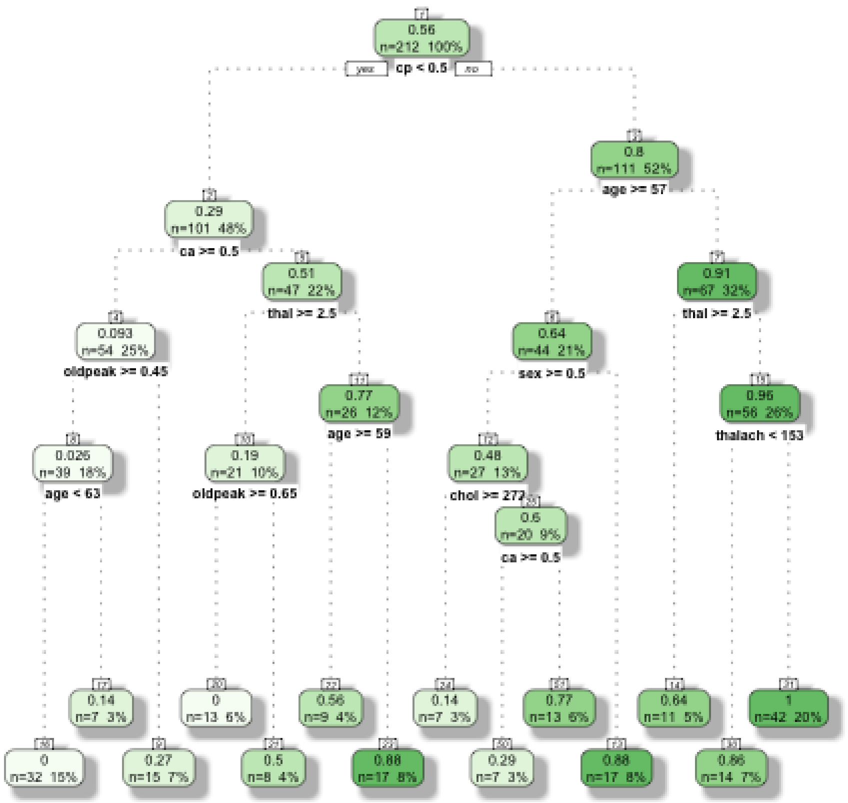
```
RPART.1 <-
  rpart(target ~ . , data = train, cp = 0.00001)
```

low cp (complexity parameter) for more sophisticated model

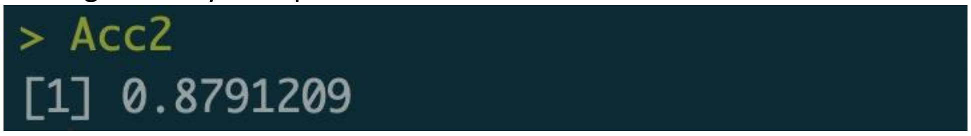
Change of error with decrease of complexity parameter and increase in size of the tree



Plot of variables inside the tree

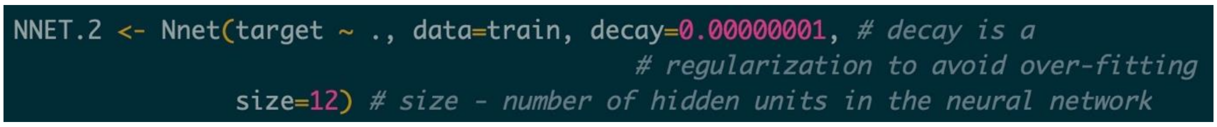


Finding accuracy as in previous models



```
> Acc2
[1] 0.8791209
```

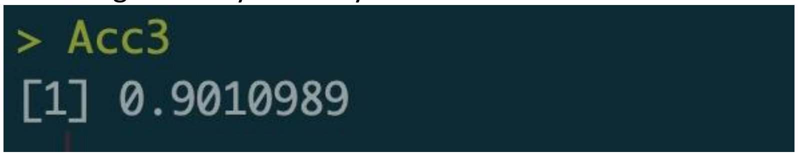
Neural Network



```
NNET.2 <- Nnet(target ~ ., data=train, decay=0.00000001, # decay is a
                                              # regularization to avoid over-fitting
               size=12) # size - number of hidden units in the neural network
```

Setting low decay parameter and making 12 neurons in the hidden layer to make more precise model

Assessing accuracy as always



```
> Acc3
[1] 0.9010989
```

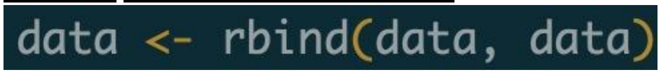
Comparison table of accuracy of all models

Model	Accuracy
Logistic Regression	0.8791
Random Forest	0.8681
Bagging	0.8571
Decision Tree	0.7802
Rpart	0.8791
Neural Network	0.9011

Conclusion: the worst model is Decision Tree with only 78% of correctly predicted heart diseases. A bit better are Bagging (0.8571) and Random Forest (0.8681). The 2nd place is hold by Logistic Regression and Rpart which both showed the accuracy of 0.8791. The best model is a Neural Network with 90% of correct predictions.

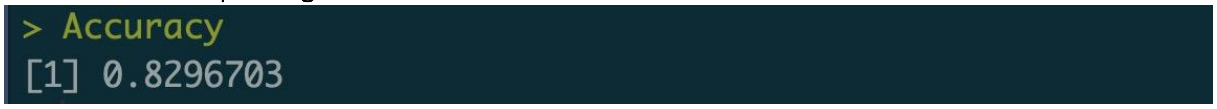
Remarks

- in seek of improvements through the feature engineering I've tried to double the data by itself to give model more observations in order to make it more accurate without changing pattern

  ```
  data <- rbind(data, data)
  ```

  but results were poor. All models decreased their accuracy on 4-5% on the average, here is the example of glm model:

  ```
  > Accuracy
  [1] 0.8296703
  ```

  before doubling the data it was 0.87.