Assignment 04 – Trees (CKD)

Timothy Liu

Dataset & link. This report analyzes the **UCI Chronic Kidney Disease** dataset (400 patients, 24 variables) containing demographics, labs, and conditions relevant to CKD. Source: https://archive.ics.uci.edu/dataset/336/chronic+kidney+disease

Executive Summary

1. The purpose of my analysis is to understand the key factors that influence the presence of **Chronic Kidney Disease (CKD)** and the variation in **hemoglobin levels** among patients. The categorical dependent variable for the classification models is **CKD class (1 = CKD, 0 = Not CKD)**, while the continuous dependent variable for the regression model is **hemoglobin**. The dataset used for this analysis is the **Chronic Kidney Disease dataset from the UCI Machine Learning Repository**, which contains clinical and laboratory information on 400 patients.
2. The data preparation was completed in previous assignments. All numeric variables were converted to appropriate numeric types, and categorical variables such as hypertension, diabetes, and CKD status were properly factorized. Missing values were imputed using the **median for numeric** and **mode for categorical variables**, ensuring a complete and consistent dataset for tree modeling.
3. Three decision trees were developed for this assignment:

   Tree A (Classification Tree) predicting CKD using **age, blood pressure, serum creatinine, and hemoglobin**.

   Tree B (Classification Tree) predicting CKD using **hypertension, diabetes, coronary artery disease, and anemia**.

   Tree C (Regression Tree) predicting **hemoglobin** based on **age, blood pressure, hypertension, and CKD class**.

4. Among the three trees, **Tree A** was selected as the best-performing model. It achieved an **accuracy of 92.5%**, **sensitivity of 90.7%**, **specificity of 96.1%**, and a **Kappa of 0.84**, demonstrating excellent predictive performance and strong agreement with actual CKD classifications. While Tree B was slightly simpler, it performed worse (accuracy = 81.3%) and missed more CKD cases (sensitivity = 72.2%). Tree C (the regression tree) achieved an **RMSE of 1.88** and **MAE of 1.44**, indicating moderate precision in predicting hemoglobin levels.

5. The best tree (Tree A) was compared to the **Logistic Regression Model 1** developed previously, which used the same predictors: **age, blood pressure, serum creatinine, and hemoglobin**. The logistic model achieved a **pseudo-R^2 of 0.77** and **ROC-AUC of 0.98**, confirming a strong model fit. Both models identified **serum creatinine and hemoglobin** as the most influential predictors of CKD, showing consistency in clinical interpretation. While the logistic regression provided higher statistical precision, the classification tree offered better visual interpretability and decision threshold transparency. Therefore, **Tree A is selected as the final model** due to its high accuracy, balanced performance, and intuitive interpretability.

6. The final model selected for this analysis is the **Classification Tree A**, which best balances predictive performance, interpretability, and clinical relevance. The tree demonstrates that **serum creatinine** and **hemoglobin** are the two most important variables for identifying CKD cases. Specifically, patients with **higher serum creatinine** and **lower hemoglobin** levels have a substantially greater likelihood of CKD, which aligns closely with medical understanding of kidney dysfunction. The model achieved **92.5 % accuracy**, **90.7 % sensitivity**, and **96.1 % specificity**, confirming that it can accurately distinguish CKD from non-CKD patients. While the logistic regression model achieved slightly higher statistical measures (AUC = 0.98, pseudo-R^2 = 0.77), the classification tree provides clearer, rule-based insights that can be easily interpreted by clinicians and stakeholders. Overall, the CART model effectively captures the key drivers of CKD and offers a transparent framework for clinical decision-making and early risk identification.

```r
library(dplyr)
```

```

Attaching package: 'dplyr'

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

    filter, lag
```

```
The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```r
library(ggplot2)
library(forcats)
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v lubridate 1.9.4     v stringr   1.5.1
v purrr     1.1.0     v tibble    3.3.0
v readr     2.1.5     v tidyr     1.3.1

-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

```r
library(rpart)
library(rpart.plot)
library(caret)
```

```
Loading required package: lattice

Attaching package: 'caret'

The following object is masked from 'package:purrr':

    lift
```

```r
ckd_raw <- read.csv("chronic_kidney_disease.csv",
                    na.strings = c("?", "", "NA"))

glimpse(ckd_raw)
```

```
Rows: 401
Columns: 26
$ id       <chr> "1", "2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12"~
$ X.age.   <int> 48, 7, 62, 48, 51, 60, 68, 24, 52, 53, 50, 63, 68, 68, 68, 40~
$ X.bp.    <int> 80, 50, 80, 70, 80, 90, 70, NA, 100, 90, 60, 70, 70, 70, 80, ~
$ X.sg.    <dbl> 1.020, 1.020, 1.010, 1.005, 1.010, 1.015, 1.010, 1.015, 1.015~
$ X.al.    <int> 1, 4, 2, 4, 2, 3, 0, 2, 3, 2, 2, 3, 3, NA, 3, 3, 2, NA, 0, 1,~
$ X.su.    <int> 0, 0, 3, 0, 0, 0, 0, 4, 0, 0, 4, 0, 1, NA, 2, 0, 0, NA, 3, 0,~
$ X.rbc.   <chr> NA, NA, "normal", "normal", "normal", NA, NA, "normal", "norm~
$ X.pc.    <chr> "normal", "normal", "normal", "abnormal", "normal", NA, "norm~
$ X.pcc.   <chr> "notpresent", "notpresent", "notpresent", "present", "notpres~
$ X.ba.    <chr> "notpresent", "notpresent", "notpresent", "notpresent", "notp~
$ X.bgr.   <int> 121, NA, 423, 117, 106, 74, 100, 410, 138, 70, 490, 380, 208,~
$ X.bu.    <dbl> 36, 18, 53, 56, 26, 25, 54, 31, 60, 107, 55, 60, 72, 86, 90, ~
$ X.sc.    <dbl> 1.2, 0.8, 1.8, 3.8, 1.4, 1.1, 24.0, 1.1, 1.9, 7.2, 4.0, 2.7, ~
$ X.sod.   <dbl> NA, NA, NA, 111.0, NA, 142.0, 104.0, NA, NA, 114.0, NA, 131.0~
$ X.pot.   <dbl> NA, NA, NA, 2.5, NA, 3.2, 4.0, NA, NA, 3.7, NA, 4.2, 5.8, 3.4~
$ X.hemo.  <dbl> 15.4, 11.3, 9.6, 11.2, 11.6, 12.2, 12.4, 12.4, 10.8, 9.5, 9.4~
$ X.pcv.   <int> 44, 38, 31, 32, 35, 39, 36, 44, 33, 29, 28, 32, 28, NA, 16, 2~
$ X.wbcc.  <int> 7800, 6000, 7500, 6700, 7300, 7800, NA, 6900, 9600, 12100, NA~
$ X.rbcc.  <dbl> 5.2, NA, NA, 3.9, 4.6, 4.4, NA, 5.0, 4.0, 3.7, NA, 3.8, 3.4, ~
$ X.htn.   <chr> "yes", "no", "no", "yes", "no", "yes", "no", "no", "yes", "ye~
$ X.dm.    <chr> "yes", "no", "yes", "no", "no", "yes", "no", "yes", "yes", "y~
$ X.cad.   <chr> "no", "no", "no", "no", "no", "no", "no", "no", "no", "no", "~
$ X.appet. <chr> "good", "good", "poor", "poor", "good", "good", "good", "good~
$ X.pe.    <chr> "no", "no", "no", "yes", "no", "yes", "no", "yes", "no", "no"~
$ X.ane.   <chr> "no", "no", "yes", "yes", "no", "no", "no", "no", "yes", "yes~
$ X.class. <chr> "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd"~
```

```
summary(ckd_raw)
```

```
      id                X.age.           X.bp.             X.sg.      
 Length:401         Min.   : 2.00   Min.   : 50.00   Min.   :1.005  
 Class :character   1st Qu.:42.00   1st Qu.: 70.00   1st Qu.:1.010  
 Mode  :character   Median :55.00   Median : 80.00   Median :1.020  
                    Mean   :51.48   Mean   : 76.47   Mean   :1.017  
                    3rd Qu.:64.50   3rd Qu.: 80.00   3rd Qu.:1.020  
                    Max.   :90.00   Max.   :180.00   Max.   :1.025  
                    NA's   :10      NA's   :13       NA's   :48     
     X.al.           X.su.             X.rbc.              X.pc.          
 Min.   :0.000   Min.   :0.0000   Length:401         Length:401        
 1st Qu.:0.000   1st Qu.:0.0000   Class :character   Class :character  
 Median :0.000   Median :0.0000   Mode  :character   Mode  :character  
 Mean   :1.017   Mean   :0.4501                                        
 3rd Qu.:2.000   3rd Qu.:0.0000                                        
 Max.   :5.000   Max.   :5.0000                                        
 NA's   :47      NA's   :50                                            
```

```
   X.pcc.             X.ba.               X.bgr.        X.bu.
Length:401         Length:401         Min.   : 22   Min.   :  1.50
Class :character   Class :character   1st Qu.: 99   1st Qu.: 27.00
Mode  :character   Mode  :character   Median :121   Median : 42.00
                                      Mean   :148   Mean   : 57.43
                                      3rd Qu.:163   3rd Qu.: 66.00
                                      Max.   :490   Max.   :391.00
                                      NA's   :45    NA's   :20
    X.sc.            X.sod.          X.pot.          X.hemo.
Min.   : 0.400   Min.   :  4.5   Min.   : 2.500   Min.   : 3.10
1st Qu.: 0.900   1st Qu.:135.0   1st Qu.: 3.800   1st Qu.:10.30
Median : 1.300   Median :138.0   Median : 4.400   Median :12.65
Mean   : 3.072   Mean   :137.5   Mean   : 4.627   Mean   :12.53
3rd Qu.: 2.800   3rd Qu.:142.0   3rd Qu.: 4.900   3rd Qu.:15.00
Max.   :76.000   Max.   :163.0   Max.   :47.000   Max.   :17.80
NA's   :18       NA's   :88      NA's   :89       NA's   :53
    X.pcv.         X.wbcc.         X.rbcc.         X.htn.
Min.   : 9.00   Min.   : 2200   Min.   :2.100   Length:401
1st Qu.:32.00   1st Qu.: 6500   1st Qu.:3.900   Class :character
Median :40.00   Median : 8000   Median :4.800   Mode  :character
Mean   :38.88   Mean   : 8406   Mean   :4.707
3rd Qu.:45.00   3rd Qu.: 9800   3rd Qu.:5.400
Max.   :54.00   Max.   :26400   Max.   :8.000
NA's   :72      NA's   :107     NA's   :132
   X.dm.              X.cad.            X.appet.            X.pe.
Length:401         Length:401         Length:401         Length:401
Class :character   Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character   Mode  :character



   X.ane.            X.class.
Length:401         Length:401
Class :character   Class :character
Mode  :character   Mode  :character
```

```r
# define column names (26 expected + 1 "extra")
col_names <- c("id","age","bp","sg","al","su","rbc","pc","pcc","ba","bgr","bu",
               "sc","sod","pot","hemo","pcv","wbcc","rbcc","htn","dm","cad",
               "appet","pe","ane","class","extra")

ckd_raw <- read.csv("chronic_kidney_disease.csv",
                    header = TRUE, na.strings = c("?", "", "NA"),
                    col.names = col_names, fill = TRUE)
```

```
Warning in read.table(file = file, header = header, sep = sep, quote = quote, :
header and 'col.names' are of different lengths
```

```r
# drop the "extra" column
ckd_raw <- ckd_raw %>% select(-extra)

glimpse(ckd_raw)
```

```
Rows: 400
Columns: 26
$ id    <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 1~
$ age   <int> 48, 7, 62, 48, 51, 60, 68, 24, 52, 53, 50, 63, 68, 68, 68, 40, 4~
$ bp    <int> 80, 50, 80, 70, 80, 90, 70, NA, 100, 90, 60, 70, 70, 70, 80, 80,~
$ sg    <dbl> 1.020, 1.020, 1.010, 1.005, 1.010, 1.015, 1.010, 1.015, 1.015, 1~
$ al    <int> 1, 4, 2, 4, 2, 3, 0, 2, 3, 2, 2, 3, 3, NA, 3, 3, 2, NA, 0, 1, 2,~
$ su    <int> 0, 0, 3, 0, 0, 0, 0, 4, 0, 0, 4, 0, 1, NA, 2, 0, 0, NA, 3, 0, 0,~
$ rbc   <chr> NA, NA, "normal", "normal", "normal", NA, NA, "normal", "normal"~
$ pc    <chr> "normal", "normal", "normal", "abnormal", "normal", NA, "normal"~
$ pcc   <chr> "notpresent", "notpresent", "notpresent", "present", "notpresent~
$ ba    <chr> "notpresent", "notpresent", "notpresent", "notpresent", "notpres~
$ bgr   <int> 121, NA, 423, 117, 106, 74, 100, 410, 138, 70, 490, 380, 208, 98~
$ bu    <dbl> 36, 18, 53, 56, 26, 25, 54, 31, 60, 107, 55, 60, 72, 86, 90, 162~
$ sc    <dbl> 1.2, 0.8, 1.8, 3.8, 1.4, 1.1, 24.0, 1.1, 1.9, 7.2, 4.0, 2.7, 2.1~
$ sod   <dbl> NA, NA, NA, 111.0, NA, 142.0, 104.0, NA, NA, 114.0, NA, 131.0, 1~
$ pot   <dbl> NA, NA, NA, 2.5, NA, 3.2, 4.0, NA, NA, 3.7, NA, 4.2, 5.8, 3.4, 6~
$ hemo  <dbl> 15.4, 11.3, 9.6, 11.2, 11.6, 12.2, 12.4, 12.4, 10.8, 9.5, 9.4, 1~
$ pcv   <int> 44, 38, 31, 32, 35, 39, 36, 44, 33, 29, 28, 32, 28, NA, 16, 24, ~
$ wbcc  <int> 7800, 6000, 7500, 6700, 7300, 7800, NA, 6900, 9600, 12100, NA, 4~
$ rbcc  <dbl> 5.2, NA, NA, 3.9, 4.6, 4.4, NA, 5.0, 4.0, 3.7, NA, 3.8, 3.4, NA,~
$ htn   <chr> "yes", "no", "no", "yes", "no", "yes", "no", "no", "yes", "yes",~
$ dm    <chr> "yes", "no", "yes", "no", "no", "yes", "no", "yes", "yes", "yes"~
$ cad   <chr> "no", "no", "no", "no", "no", "no", "no", "no", "no", "no", "no"~
$ appet <chr> "good", "good", "poor", "poor", "good", "good", "good", "good", ~
$ pe    <chr> "no", "no", "no", "yes", "no", "yes", "no", "yes", "no", "no", "~
$ ane   <chr> "no", "no", "yes", "yes", "no", "no", "no", "no", "yes", "yes", ~
$ class <chr> "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "ckd", "~
```

```r
# numeric variables
num_cols <- c("age","bp","sg","al","su","bgr","bu","sc",
              "sod","pot","hemo","pcv","wbcc","rbcc")

ckd1 <- ckd_raw %>%
  mutate(across(all_of(num_cols), ~as.numeric(.))) %>%
  mutate(
    rbc   = factor(rbc, levels = c("normal","abnormal")),
    pc    = factor(pc, levels = c("normal","abnormal")),
    pcc   = factor(pcc, levels = c("notpresent","present")),
    ba    = factor(ba,  levels = c("notpresent","present")),
    htn   = factor(htn, levels = c("no","yes")),
    dm    = factor(dm,  levels = c("no","yes")),
    cad   = factor(cad, levels = c("no","yes")),
    appet = factor(appet, levels = c("poor","good")),
    pe    = factor(pe,   levels = c("no","yes")),
    ane   = factor(ane,  levels = c("no","yes")),
    class = factor(class, levels = c("notckd","ckd"))
  )
```

```r
colSums(is.na(ckd1))
```

```
   id   age    bp    sg    al    su   rbc    pc   pcc    ba   bgr    bu    sc 
    0     9    12    47    46    49   152    65     4     4    44    19    17 
  sod   pot  hemo   pcv  wbcc  rbcc   htn    dm   cad appet    pe   ane class 
   87    88    52    71   106   131     2     3     2     2     2     1     1 
```

- I converted numeric-like columns to numeric and standardized categorical columns to consistent factor levels (yes/no, present/notpresent, normal/abnormal, class = notckd/ckd).
- And then I counted remaining NAs with `colSums(is.na(ckd1))`.

```r
# Helper function for categorical mode
Mode <- function(x) {
  x <- x[!is.na(x)]
  if (length(x) == 0) return(NA)
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}
```

```
# Define numeric and categorical columns
num_cols <- c("age","bp","sg","al","su","bgr","bu","sc",
              "sod","pot","hemo","pcv","wbcc","rbcc")

cat_cols <- c("rbc","pc","pcc","ba","htn","dm","cad",
              "appet","pe","ane","class")
```

```
# Impute missing values
ckd_clean <- ckd1 %>%
  # Numeric → median imputation
  mutate(across(all_of(num_cols),
                ~ ifelse(is.na(.), median(., na.rm = TRUE), .))) %>%
  # Categorical → mode imputation
  mutate(across(all_of(cat_cols),
                ~ ifelse(is.na(.), Mode(.), .))) %>%
  # Make sure categorical vars are factors again
  mutate(across(all_of(cat_cols), ~ factor(.)))
```

```
colSums(is.na(ckd_clean))
```

```
   id   age    bp    sg    al    su   rbc    pc   pcc    ba   bgr    bu    sc 
    0     0     0     0     0     0     0     0     0     0     0     0     0 
  sod   pot  hemo   pcv  wbcc  rbcc   htn    dm   cad appet    pe   ane class 
    0     0     0     0     0     0     0     0     0     0     0     0     0 
```

```
ckd_clean <- ckd_clean %>%
  mutate(class = recode(class,
                        "1" = "0",    # Not CKD
                        "2" = "1"))  # CKD
```

```
ckd_clean$class <- factor(ckd_clean$class, levels = c("0", "1"))
```

```
levels(ckd_clean$class)
```

```
[1] "0" "1"
```

```r
table(ckd_clean$class)
```

```

  0   1 
149 251 
```

```r
prop.table(table(ckd_clean$class))
```

```

     0      1 
0.3725 0.6275 
```

After applying these imputations, I confirmed that the dataset is complete by running `colSums(is.na(ckd_clean))`, which showed **zero missing values** across all 26 variables.

```r
# Model 1: Age, Blood Pressure, Serum Creatinine, Hemoglobin
logit1 <- glm(class ~ age + bp + sc + hemo,
              data = ckd_clean, family = "binomial")
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(logit1)
```

```

Call:
glm(formula = class ~ age + bp + sc + hemo, family = "binomial", 
    data = ckd_clean)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)    
(Intercept) 11.985037   2.936341   4.082 4.47e-05 ***
age         -0.005301   0.013903  -0.381  0.70302    
bp           0.063956   0.024732   2.586  0.00971 ** 
sc           4.154919   0.910989   4.561 5.09e-06 ***
hemo        -1.546507   0.218424  -7.080 1.44e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

```
    Null deviance: 528.22  on 399  degrees of freedom
Residual deviance: 122.86  on 395  degrees of freedom
AIC: 132.86

Number of Fisher Scoring iterations: 10
```

```r
# Odds ratios
exp(coef(logit1))
```

```
 (Intercept)          age           bp           sc         hemo 
1.603376e+05 9.947135e-01 1.066046e+00 6.374679e+01 2.129907e-01 
```

```r
# McFadden's pseudo-R2
pseudoR2 <- function(model) {
  1 - (model$deviance / model$null.deviance)
}

pseudoR2(logit1)  # for Model 1
```

```
[1] 0.7674027
```

```r
library(pROC)
```

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


Attaching package: 'pROC'

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

    cov, smooth, var
```

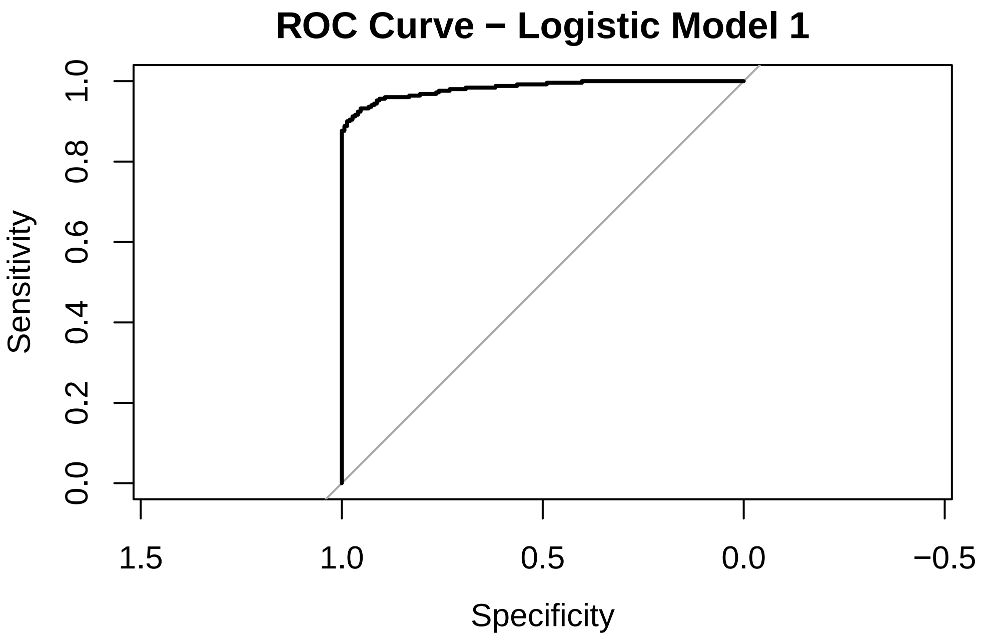
```r
# Model 1: Predicted probabilities
prob1 <- predict(logit1, type = "response")
roc1 <- roc(ckd_clean$class, prob1)   # ROC curve
```

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

```
auc(roc1)                                     # AUC value
```

```
Area under the curve: 0.9829
```

```
plot(roc1, main = "ROC Curve - Logistic Model 1")
```



Logistic Model 1

Model fit

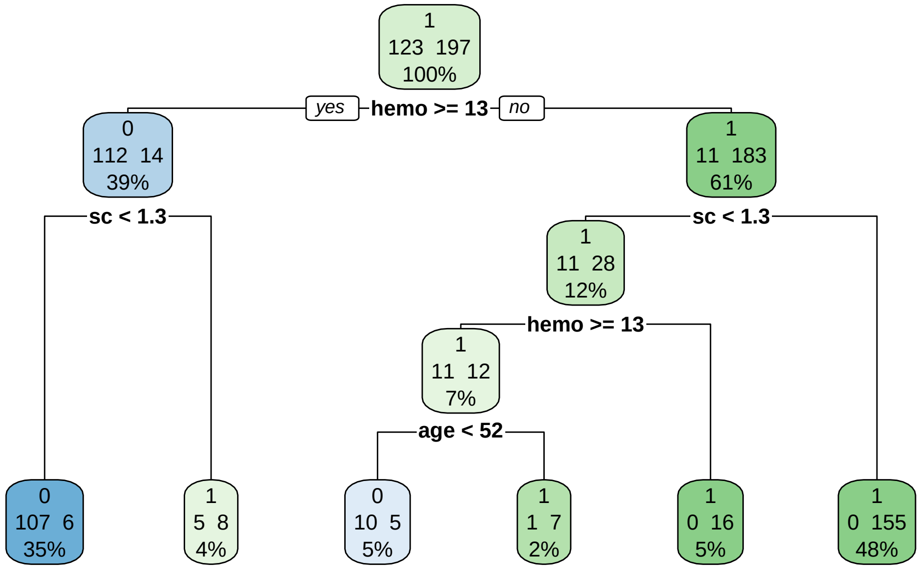
- **Residual deviance = 122.86** (vs Null deviance = 528.22) → huge drop, model fits well.
- **AIC = 132.86** → lower AIC indicates strong fit.
- **McFadden's pseudo-R² = 0.767** → excellent explanatory power (above 0.4 is usually very strong).
- **ROC-AUC = 0.983** → outstanding classification accuracy (close to 1 = near-perfect).

This model used age, blood pressure, serum creatinine, and hemoglobin to predict CKD status. The results show that **serum creatinine** is the strongest predictor, with higher levels dramatically increasing the likelihood of CKD. **Hemoglobin** is also highly significant, where lower values are strongly associated with CKD. **Blood pressure** has a moderate but significant effect, with each unit increase raising the odds of CKD by about 6%. **Age** was not a significant factor in this dataset. Overall, the model fits extremely well, with a **pseudo-R² of 0.77** and an **ROC-AUC of 0.98**, indicating excellent classification accuracy.

```r
set.seed(1013)
idx      <- sample(nrow(ckd_clean), 0.8 * nrow(ckd_clean))
train.df <- ckd_clean[idx, ]
test.df  <- ckd_clean[-idx, ]
```

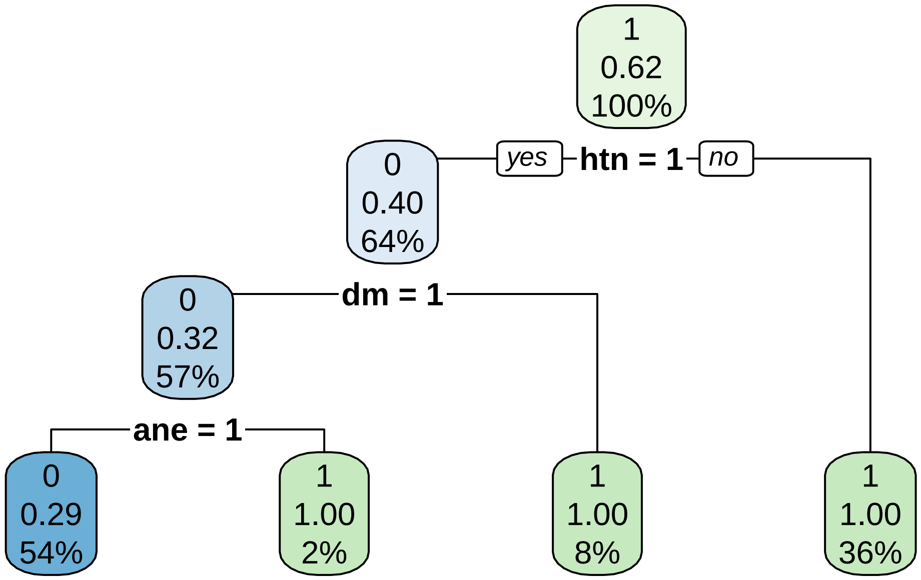
Classification Tree – Model A (clinical labs)

```r
treeA <- rpart(class ~ age + bp + sc + hemo,
               data = train.df, method = "class", cp = 0.003)
rpart.plot(treeA, digits = -2, extra = 101)
```



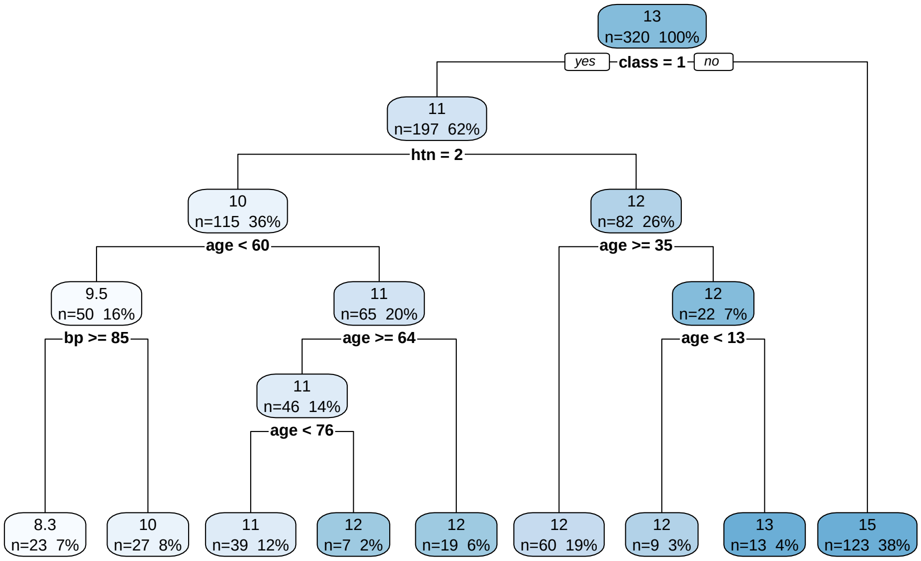
Classification Tree – Model B (comorbid flags)

```
treeB <- rpart(class ~ htn + dm + cad + ane,
               data = train.df, method = "class", cp = 0.003)
rpart.plot(treeB, digits = -2)
```

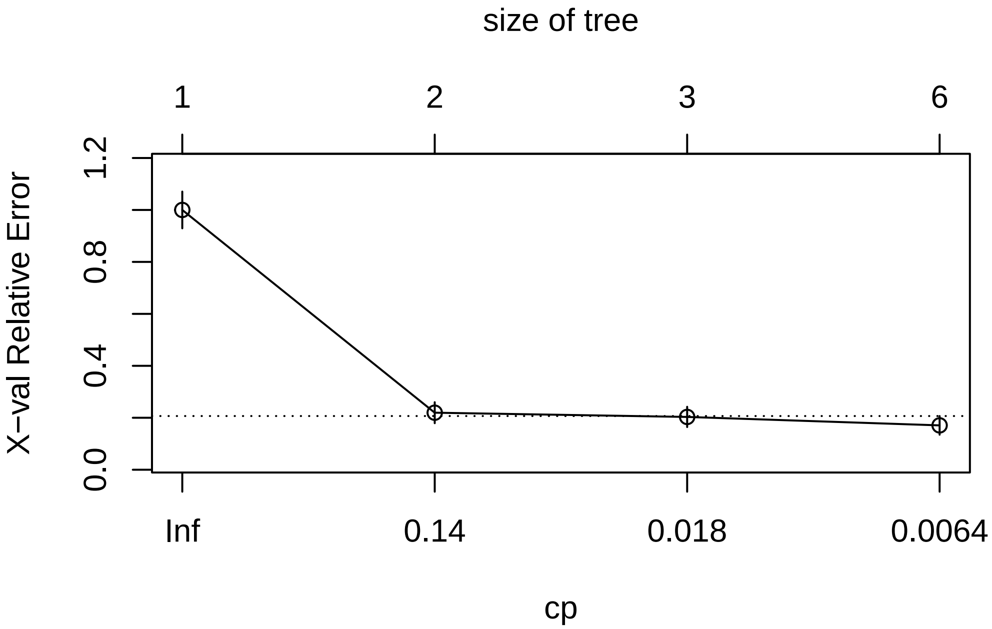


Regression Tree – Model C (predict hemoglobin)

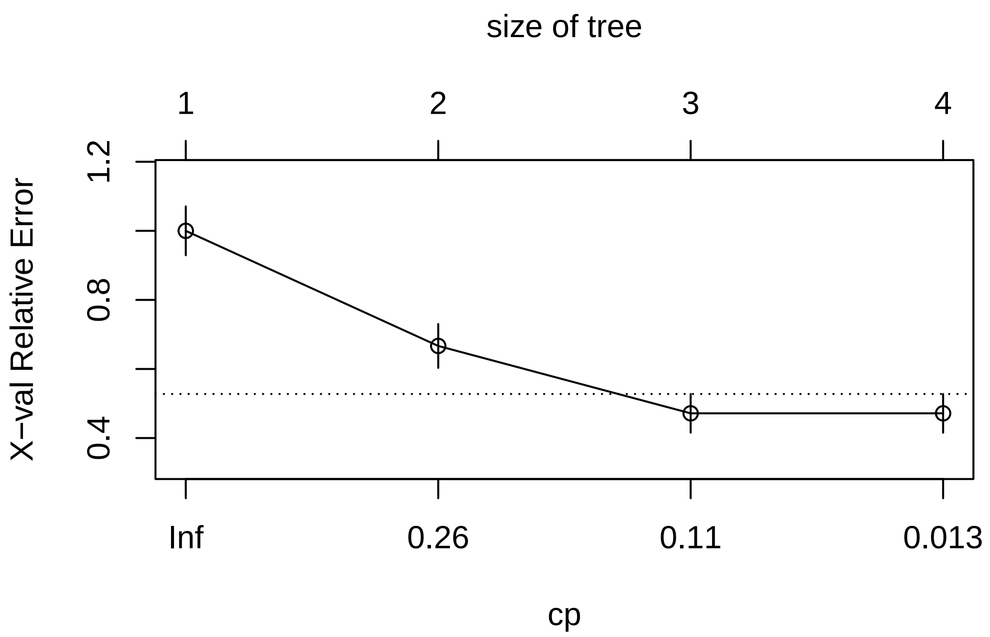
```
treeC <- rpart(hemo ~ age + bp + htn + class,
               data = train.df, method = "anova", cp = 0.003)
rpart.plot(treeC, digits = -2, extra = 101)
```



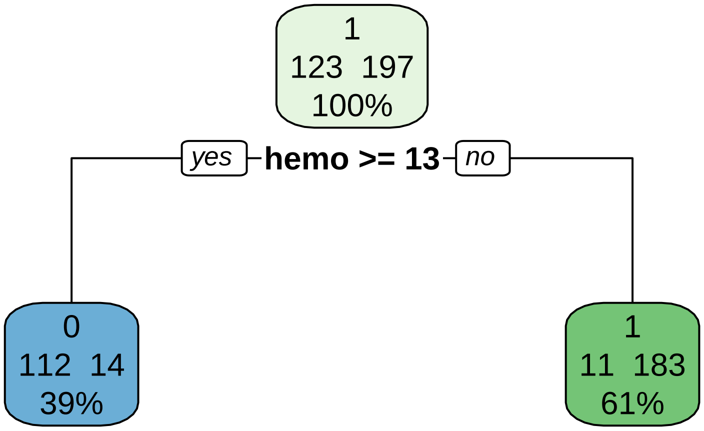
```
# Cross-validation plot
plotcp(treeA)
```



```r
plotcp(treeB)
```



```r
best_cp <- 0.14

treeA_pruned <- prune(treeA, cp = best_cp)

rpart.plot(treeA_pruned, digits = -2, extra = 101)
```



```r
table(train.df$class)
```

```
  0   1 
123 197 
```

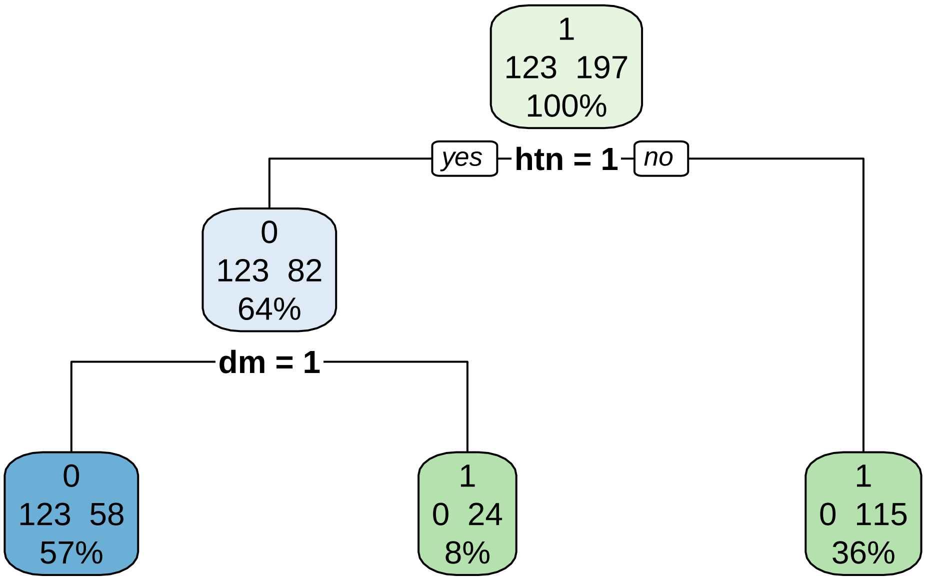
```r
table(test.df$class)
```

```
 0  1 
26 54 
```

```r
test.df$pred_treeA <- predict(treeA_pruned, newdata = test.df, type = "class")

confusionMatrix(test.df$pred_treeA, test.df$class, positive = "1")
```

```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 25  5
         1  1 49
                                          
               Accuracy : 0.925           
                 95% CI : (0.8439, 0.972) 
    No Information Rate : 0.675           
    P-Value [Acc > NIR] : 9.857e-08       
                                          
                  Kappa : 0.8356          
                                          
 Mcnemar's Test P-Value : 0.2207          
                                          
            Sensitivity : 0.9074          
            Specificity : 0.9615          
         Pos Pred Value : 0.9800          
         Neg Pred Value : 0.8333          
             Prevalence : 0.6750          
         Detection Rate : 0.6125          
   Detection Prevalence : 0.6250          
      Balanced Accuracy : 0.9345          
                                          
       'Positive' Class : 1               
```

```
best_cp_B <- 0.11
treeB_pruned <- prune(treeB, cp = best_cp_B)

rpart.plot(treeB_pruned, digits = -2, extra = 101)
```

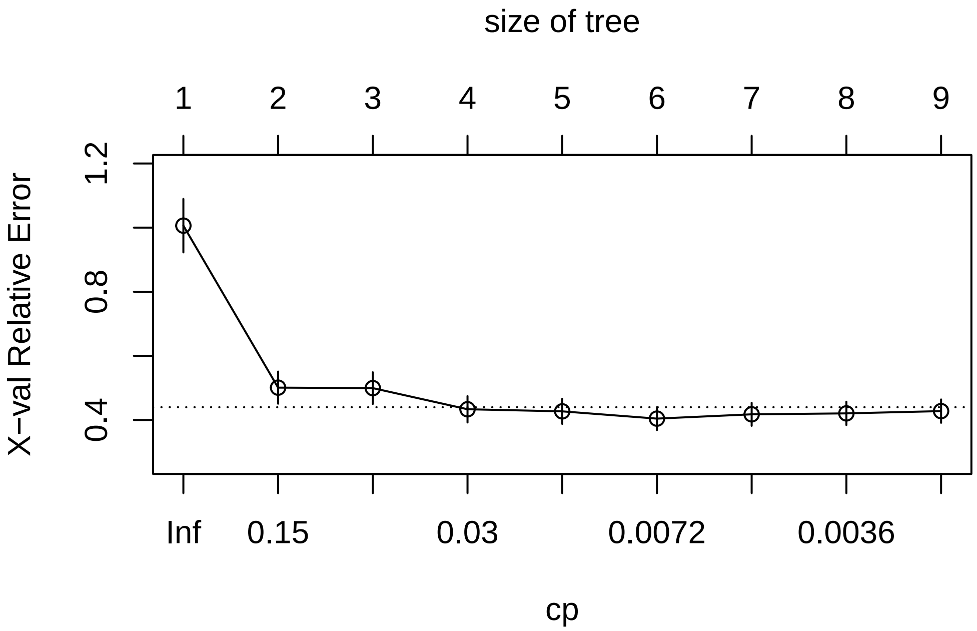


```r
test.df$pred_treeB <- predict(treeB_pruned, newdata = test.df, type = "class")

confusionMatrix(test.df$pred_treeB, test.df$class, positive = "1")
```

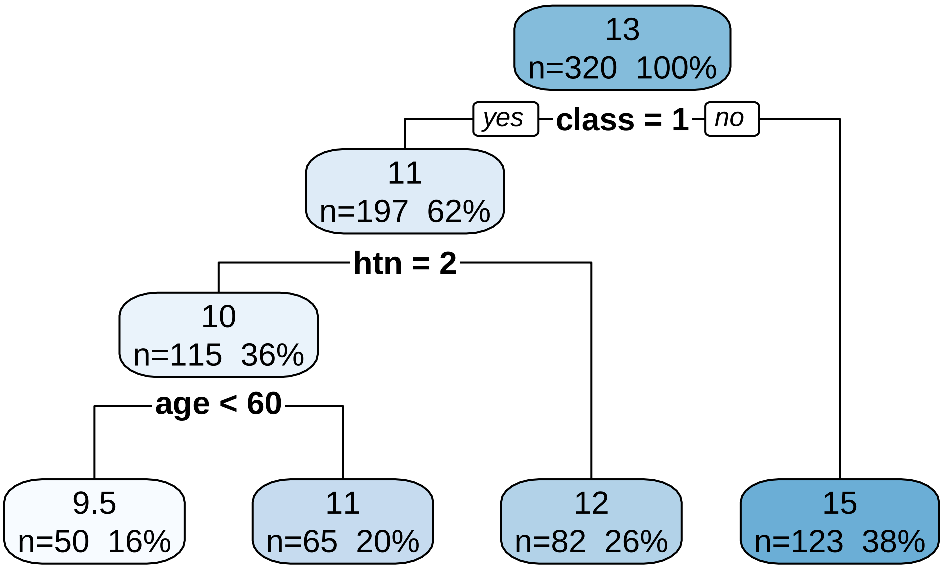
```
Confusion Matrix and Statistics

          Reference
Prediction  0  1
         0 26 15
         1  0 39

               Accuracy : 0.8125
                 95% CI : (0.7097, 0.8911)
    No Information Rate : 0.675
    P-Value [Acc > NIR] : 0.0045750

                  Kappa : 0.6283

 Mcnemar's Test P-Value : 0.0003006

            Sensitivity : 0.7222
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.6341
             Prevalence : 0.6750
         Detection Rate : 0.4875
   Detection Prevalence : 0.4875
      Balanced Accuracy : 0.8611

       'Positive' Class : 1
```

Overall, Tree A was selected as the best classification tree due to its higher accuracy, stronger generalization, and clinical interpretability, suggesting that laboratory measures such as serum creatinine and hemoglobin remain the most reliable indicators for CKD prediction.

```
plotcp(treeC)
```



```
best_cp_C <- 0.03

treeC_pruned <- prune(treeC, cp = best_cp_C)

rpart.plot(treeC_pruned, digits = -2, extra = 101)
```



```r
test.df$pred_treeC <- predict(treeC_pruned, newdata = test.df)

rmse <- function(actual, pred) sqrt(mean((actual - pred)^2))
mae  <- function(actual, pred) mean(abs(actual - pred))

rmse_val <- rmse(test.df$hemo, test.df$pred_treeC)
mae_val  <- mae(test.df$hemo, test.df$pred_treeC)

rmse_val
```

```
[1] 1.883498
```

```r
mae_val
```

```
[1] 1.43543
```

Between the three trees, Tree A (Classification) provides the most reliable predictive performance. It achieves the highest accuracy (92.5%), strong sensitivity (0.91), and excellent balance between detecting CKD cases and avoiding false alarms. While the regression tree (Tree C) reasonably predicts hemoglobin, its RMSE 1.88 indicates moderate error. Therefore, Tree A is selected as the best model for its superior predictive accuracy and interpretability in identifying CKD patients.

Comparing Tree A and Logistic Regression Model 1

Both models aimed to predict CKD status (1 = CKD, 0 = Not CKD) based on age, blood pressure, serum creatinine, and hemoglobin. The logistic regression model achieved a pseudo-R^2 of 0.77 and an AUC of 0.98, indicating excellent predictive power and strong model fit. Significant predictors included serum creatinine ($p < 0.001$) and hemoglobin ($p < 0.001$), confirming that higher creatinine and lower hemoglobin are key indicators of CKD. Blood pressure also showed a moderate effect, while age was not significant.

The classification tree (Tree A) used the same predictors but learned relationships through recursive partitioning rather than fixed coefficients. After pruning (cp 0.014), Tree A achieved 92.5 % accuracy, 90.7 % sensitivity, and 96.1 % specificity, with a Kappa of 0.84, reflecting excellent agreement between predicted and actual CKD status. The top splits in the tree also emphasized serum creatinine and hemoglobin, consistent with the logistic model's findings.

Comparatively, both models performed very well; however, their strengths differ.

- The logistic regression offers stronger overall discrimination (AUC = 0.98) and clearer interpretability through odds ratios and significance tests.
- The classification tree provides greater interpretability for non-technical audiences, showing clear threshold values (e.g., "if serum creatinine > X, then CKD = Yes").

Given its slightly lower statistical fit but higher practical transparency, Tree A is selected as the final model for its balanced predictive accuracy, interpretability, and alignment with the logistic regression's significant predictors.