Chronic Kidney Disease Project

DASC 3213 - Statistical Learning

Avie Timby

May 02, 2024

```
knitr::opts_chunk$set(echo = TRUE)
library(ISLR2)
```

```
## Warning: package 'ISLR2' was built under R version 4.4.0
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.4.0
```

```
## Warning: package 'ggplot2' was built under R version 4.4.0
```

```
## Warning: package 'tibble' was built under R version 4.4.0
```

```
## Warning: package 'tidyr' was built under R version 4.4.0
```

```
## Warning: package 'readr' was built under R version 4.4.0
```

```
## Warning: package 'purrr' was built under R version 4.4.0
```

```
## Warning: package 'dplyr' was built under R version 4.4.0
```

```
## Warning: package 'stringr' was built under R version 4.4.0
```

```
## Warning: package 'forcats' was built under R version 4.4.0
```

```
## Warning: package 'lubridate' was built under R version 4.4.0
```

```
library(MASS)
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.4.0
```

```r
library(class)
library(dplyr)
library(tinytex)
```

```
## Warning: package 'tinytex' was built under R version 4.4.0
```

```r
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.2.3
```

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.4.0
```

Analysis of Previous Work

For this part of the project you will need to read the paper *Risk Factor Prediction of Chronic Kidney Disease based on Machine Learning Algorithms* by Islam et al (2020) and answer the following questions which are intended to help you understand and critique the paper.

Provide a short summary of the paper. Make sure to address the following questions in your response. - What is the statistical research question the paper tries to address? - Is this a supervised or unsupervised learning problem? - What models are used to investigate the research question? Which model was reported to have the best performance? - Are all of these models appropriate in the context of the problem? If not, which models have been used incorrectly? - How was model selection performed in the paper? What about model validation? - Did you find the paper to be reproducible in its current state?

Paper Summary

The research hopes to predict the risk factors most associated with CKD (Chronic Kidney Disease) using 6 different statistical algorithms to find the 'best' classification outcomes in order to predict who may be at risk of CKD. This problem is a supervised learning problem as the main goal is classification of data inputs The paper uses the models/algorithms: Naive Bayes, Random Forest, Simple Logistic Regression, Decision Stump, Linear Regression, and Simple Linear Regression to analyze the data and investigate the research question. Random forest was reported to have the best performance of the models, with 98.8858% accuracy.These models are used mostly appropriately for their tasks, although decision stump on its own is not the best predictor of significant features. There wasn't much model selection or validation performed in this paper, Bayes, Random Forest and simple logistic regression were just used to evaluate the accuracy of the other models predictions, and the one with the best accuracy was selected. The paper is somewhat reproducible in its current state, as we know which models and algorithms were used as well as the metric used for discerning a models validity, however we do not know exactly what was done to the data. We only know that the data was smoothed for missing values.

Reviewing and Cleaning Data

The next step of the project it to investigate the data to check if it needs to be modified or cleaned prior to fitting the models.

In order to better understand the data answer the following questions. You will need to view the data in R to answer some of the questions.

- What are the dimensions of the data set? How many covariates were measured on each experimental unit?
- Are there any missing values in the data set? How did Islam et al. (2020) report to handle any missing data?
- What types of covariates (Continuous, Discrete, Ordinal, Nominal) are reported to be in the data by Islam et al. (2020)?
- Review the list of covariates and their data types on the UC Irvine ML repository (https://archive.ics.uci.edu/dataset/857/risk+factor+prediction+of+chronic+kidney+disease)
- What are the types of covariates listed in the actual data set in R when you first load it? Does this properly align with the data types reported in Isalm et al. (2020) and on the UC Irvine ML repository?
- Reformat the data in R so that it is appropriate for further analysis.

```
ckd_dataset_v2 <- read_csv(
  "C:/Users/aviet/Documents/DASC3213/data/ckd-dataset-v2.csv",
  show_col_types = FALSE)
```

```
dim(ckd_dataset_v2)
```

```
## [1] 202  29
```

```
print("Count of total missing values  ")
```

```
## [1] "Count of total missing values  "
```

```
sum(is.na(ckd_dataset_v2))
```

```
## [1] 27
```

```
print("Count of total missing values by column  ")
```

```
## [1] "Count of total missing values by column  "
```

```
colSums(is.na(ckd_dataset_v2))
```

```
## bp (Diastolic)       bp limit             sg             al          class
##               1              1              1              1              1
##             rbc             su             pc            pcc             ba
##               1              1              1              1              1
##             bgr             bu            sod             sc            pot
##               1              1              1              1              1
##            hemo            pcv           rbcc           wbcc            htn
##               1              1              1              1              1
##              dm            cad          appet             pe            ane
##               1              1              1              1              1
##             grf          stage       affected            age
##               1              1              0              0
```

```
print("Count of total missing values by row  ")
```

```
## [1] "Count of total missing values by row  "
```

```
rowSums(is.na(ckd_dataset_v2))
```

```
##   [1]  0 27  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##  [26]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##  [51]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
##  [76]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## [101]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## [126]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## [151]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## [176]  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0
## [201]  0  0
```

The dimensions of the dataframe is 202x29, with 29 covariates. There are 27 missing pieces of data, and all 27 come from the same row or experimental unit. Islam et al. (2020) handled the missing data by filling replacing it with the mean value from the column. The covariates were originally reported to be nominal and converted via encoding. One of the covariates was originally categorical.

When the data is first loaded into R, the only data type available is discrete which doesn't match with either the Islam report or the UC Irving report.

```
# Remove row with missing data, its almost the entire row so it doesnt really contribute
ckd_dataset <- ckd_dataset_v2[-c(1:2),]
ckd_dataset <- ckd_dataset[-c(180),]
ckd_dataset
```

```
## # A tibble: 199 x 29
##    `bp (Diastolic)` `bp limit` sg      al    class rbc   su    pc    pcc   ba
##    <chr>            <chr>      <chr>   <chr> <chr> <chr> <chr> <chr> <chr> <chr>
```

```
##  1 0                0          1.019 ~ 1 - 1 ckd   0     < 0   0     0     0
##  2 0                0          1.009 ~ < 0   ckd   0     < 0   0     0     0
##  3 0                0          1.009 ~   4   ckd   1     < 0   1     0     1
##  4 1                1          1.009 ~ 3 - 3 ckd   0     < 0   0     0     0
##  5 0                0          1.015 ~ < 0   ckd   0     < 0   0     0     0
##  6 1                1            1.023 < 0   notc~ 0     < 0   0     0     0
##  7 0                0          1.019 ~ 3 - 3 ckd   0     < 0   0     0     0
##  8 0                0          1.019 ~ < 0   ckd   0     < 0   0     0     0
##  9 0                0            1.023 < 0   notc~ 0     < 0   0     0     0
## 10 1                2          1.009 ~   4   ckd   0     < 0   1     1     1
## # i 189 more rows
## # i 19 more variables: bgr <chr>, bu <chr>, sod <chr>, sc <chr>, pot <chr>,
## #   hemo <chr>, pcv <chr>, rbcc <chr>, wbcc <chr>, htn <chr>, dm <chr>,
## #   cad <chr>, appet <chr>, pe <chr>, ane <chr>, grf <chr>, stage <chr>,
## #   affected <chr>, age <chr>
```

```r
#Change names to make it easier to call/work with
names(ckd_dataset)[names(ckd_dataset) == "bp (Diastolic)"] <- "bp_diastolic"
names(ckd_dataset)[names(ckd_dataset) == "bp limit"] <- "bp_limit"


print(names(ckd_dataset))
```

```
##  [1] "bp_diastolic" "bp_limit"     "sg"           "al"           "class"
##  [6] "rbc"          "su"           "pc"           "pcc"          "ba"
## [11] "bgr"          "bu"           "sod"          "sc"           "pot"
## [16] "hemo"         "pcv"          "rbcc"         "wbcc"         "htn"
## [21] "dm"           "cad"          "appet"        "pe"           "ane"
## [26] "grf"          "stage"        "affected"     "age"
```

```r
ckd_dataset$sg[ckd_dataset$sg == "< 1.007"] <- "0 - 1.011"
ckd_dataset$sg[ckd_dataset$sg == "1.009 - 1.011"] <- "0 - 1.011"
ckd_dataset$sc[ckd_dataset$sc != "< 3.65"] <- "3.65+"
ckd_dataset$su[ckd_dataset$su != "< 0"] <- "0+"
ckd_dataset$rbcc[ckd_dataset$rbcc == "  7.41"] <- "> 6.23"
ckd_dataset$rbcc[ckd_dataset$rbcc == "6.23 - 6.82"] <- "> 6.23"
ckd_dataset$rbcc[ckd_dataset$rbcc == "< 2.69"] <- "< 3.28"
ckd_dataset$rbcc[ckd_dataset$rbcc == "2.69 - 3.28"] <- "< 3.28"
ckd_dataset$bgr[ckd_dataset$bgr == "112 - 154"] <- "< 154"
ckd_dataset$bgr[ckd_dataset$bgr == "< 112"] <- "< 154"
ckd_dataset$bgr[ckd_dataset$bgr != "< 154"] <- "154+"
ckd_dataset$al[ckd_dataset$al != "< 0"] <- "> 0"
ckd_dataset$bu[ckd_dataset$bu != "< 48.1" & ckd_dataset$bu != "48.1 - 86.2"] <- "> 86.2"
ckd_dataset$sod[ckd_dataset$sod == "128 - 133"] <- "< 133"
ckd_dataset$sod[ckd_dataset$sod == "< 118"] <- "< 133"
ckd_dataset$sod[ckd_dataset$sod == "118 - 123"] <- "< 133"
ckd_dataset$sod[ckd_dataset$sod == "123 - 128"] <- "< 133"
```

```
ckd_dataset$sod[ckd_dataset$sod == "143 - 148"] <- "143+"
ckd_dataset$sod[ckd_dataset$sod == "148 - 153"] <- "143+"
ckd_dataset$sod[ckd_dataset$sod == "  158"] <- "143+"
ckd_dataset$hemo[ckd_dataset$hemo == "< 6.1"] <- "< 11.3"
ckd_dataset$hemo[ckd_dataset$hemo == "10 - 11.3"] <- "< 11.3"
ckd_dataset$hemo[ckd_dataset$hemo == "6.1 - 7.4"] <- "< 11.3"
ckd_dataset$hemo[ckd_dataset$hemo == "7.4 - 8.7"] <- "< 11.3"
ckd_dataset$hemo[ckd_dataset$hemo == "8.7 - 10"] <- "< 11.3"
ckd_dataset$hemo[ckd_dataset$hemo == "15.2 - 16.5"] <- "15.2+"
ckd_dataset$hemo[ckd_dataset$hemo == "  16.5"] <- "15.2+"
ckd_dataset$pcv[ckd_dataset$pcv == "< 17.9"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "17.9 - 21.8"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "21.8 - 25.7"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "25.7 - 29.6"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "29.6 - 33.5"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "33.5 - 37.4"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "17.9 - 21.8"] <- "< 37.4"
ckd_dataset$pcv[ckd_dataset$pcv == "41.3 - 45.2"] <- "41.3 - 49.1"
ckd_dataset$pcv[ckd_dataset$pcv == "45.2 - 49.1"] <- "41.3 - 49.1"
ckd_dataset$wbcc[ckd_dataset$wbcc == "12120 - 14500"] <- "12120+"
ckd_dataset$wbcc[ckd_dataset$wbcc == "  24020"] <- "12120+"
ckd_dataset$wbcc[ckd_dataset$wbcc == "= 24020"] <- "12120+"
ckd_dataset$wbcc[ckd_dataset$wbcc == "14500 - 16880"] <- "12120+"
ckd_dataset$wbcc[ckd_dataset$wbcc == "16880 - 19260"] <- "12120+"
ckd_dataset$wbcc[ckd_dataset$wbcc == "19260 - 21640"] <- "12120+"
ckd_dataset$grf[ckd_dataset$grf == "  227.944"] <- "177.612+"
ckd_dataset$grf[ckd_dataset$grf == "177.612 - 202.778"] <- "177.612+"
ckd_dataset$grf[ckd_dataset$grf == "202.778 - 227.944"] <- "177.612+"
```

```
dataset <- ckd_dataset %>%
  mutate(across(c( 'bp_limit','bp_diastolic', 'rbc', 'pc', 'pcc', 'ba','htn', 'dm',

                   'cad','appet', 'pe','ane', 'affected'), as.integer))%>%
  mutate(across(c('sg', 'al', 'su', 'bgr', 'bu', 'sod','sc',
                  'pot', 'hemo', 'pcv', 'rbcc','wbcc','grf','stage', 'age', 'class'), as.factor

summary(dataset)
```

```
##   bp_diastolic       bp_limit                  sg       al         class
##  Min.   :0.0000   Min.   :0.0000    1.023      :41   < 0:115   ckd   :127
##  1st Qu.:0.0000   1st Qu.:0.0000   0 - 1.011    :48   > 0: 84   notckd: 72
##  Median :1.0000   Median :1.0000   1.015 - 1.017:36
```

```
##  Mean   :0.5377   Mean   :0.7487   1.019 - 1.021:74
##  3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :2.0000
##
##       rbc           su            pc              pcc               ba
##  Min.   :0.0000   < 0:169   Min.   :0.0000   Min.   :0.0000   Min.   :0.00000
##  1st Qu.:0.0000   0+ : 30   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.00000
##  Median :0.0000             Median :0.0000   Median :0.0000   Median :0.00000
##  Mean   :0.1256             Mean   :0.2261   Mean   :0.1357   Mean   :0.05528
##  3rd Qu.:0.0000             3rd Qu.:0.0000   3rd Qu.:0.0000   3rd Qu.:0.00000
##  Max.   :1.0000             Max.   :1.0000   Max.   :1.0000   Max.   :1.00000
##
##     bgr                bu             sod          sc                 pot
##  < 154:148   < 48.1     :108   < 133    :27   < 3.65:159   < 7.31       :196
##  154+ : 51   > 86.2     : 38   133 - 138:91   3.65+ : 40     42.59      :  1
##              48.1 - 86.2: 53   138 - 143:49                38.18 - 42.59:  1
##                                143+     :32                7.31 - 11.72 :  1
##
##
##
##           hemo             pcv              rbcc              wbcc
##  < 11.3     :71   < 37.4     :79   < 3.28     :11   < 4980      :10
##  11.3 - 12.6:48     49.1     :19   > 6.23     :10   12120+      :16
##  12.6 - 13.9:19   37.4 - 41.3:55   3.28 - 3.87:21   4980 - 7360 :47
##  13.9 - 15.2:26   41.3 - 49.1:46   3.87 - 4.46:21   7360 - 9740 :97
##  15.2+      :35                   4.46 - 5.05:95   9740 - 12120:29
##                                   5.05 - 5.64:23
##                                   5.64 - 6.23:18
##       htn              dm              cad             appet
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000   Min.   :0.000
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.000
##  Median :0.000   Median :0.0000   Median :0.0000   Median :0.000
##  Mean   :0.392   Mean   :0.3467   Mean   :0.1106   Mean   :0.201
##  3rd Qu.:1.000   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.:0.000
##  Max.   :1.000   Max.   :1.0000   Max.   :1.0000   Max.   :1.000
##
##        pe              ane                        grf      stage
##  Min.   :0.0000   Min.   :0.0000   < 26.6175         :68   s1:54
##  1st Qu.:0.0000   1st Qu.:0.0000   26.6175 - 51.7832 :38   s2:35
##  Median :0.0000   Median :0.0000   51.7832 - 76.949  :28   s3:31
##  Mean   :0.1759   Mean   :0.1608   76.949 - 102.115  :17   s4:45
##  3rd Qu.:0.0000   3rd Qu.:0.0000   102.115 - 127.281 :15   s5:34
##  Max.   :1.0000   Max.   :1.0000   177.612+          :13
##                                    (Other)           :20
##     affected          age
##  Min.   :0.0000   59 - 66:48
##  1st Qu.:0.0000   51 - 59:33
##  Median :1.0000   66 - 74:33
```

```
##  Mean   :0.6382   43 - 51:31
##  3rd Qu.:1.0000   27 - 35:14
##  Max.   :1.0000   35 - 43:12
##                   (Other):28
```

Reconstructing the Models for Chronic Kidney Disease

In this section you will reconstruct the appropriate models used in Islam et al. (2020) with the aim of improving them using techniques we have learned in class.

Logistic Regression

- Construct a Logistic regression model for the research question of Islam et al. (2020)

```
library(glmnet)
```

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

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

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

```
## Loaded glmnet 4.1-8
```

```
log_reg <- glm(class ~. -pot - affected,data = dataset, family='binomial' )
log_reg
```

```
##
## Call:  glm(formula = class ~ . - pot - affected, family = "binomial",
##     data = dataset)
##
## Coefficients:
##          (Intercept)          bp_diastolic              bp_limit
##              -20.2501               29.1220              -25.1674
##          sg0 - 1.011       sg1.015 - 1.017       sg1.019 - 1.021
##              -36.9403              -22.1332               -3.8708
##                 al> 0                   rbc                  su0+
##              -11.7873               16.8065               -3.1922
##                    pc                   pcc                    ba
##                4.7276               16.5412              20.6456
##               bgr154+              bu> 86.2         bu48.1 - 86.2
```

```
##              -5.5104             -14.5855             -13.4243
##         sod133 - 138          sod138 - 143               sod143+
##             -10.3562             -12.5788               1.5505
##              sc3.65+       hemo11.3 - 12.6       hemo12.6 - 13.9
##             -44.8180             -10.6226              16.8069
##      hemo13.9 - 15.2            hemo15.2+            pcv  49.1
##              13.3892              20.0850              19.3612
##       pcv37.4 - 41.3       pcv41.3 - 49.1           rbcc> 6.23
##              24.4556              19.8083              -6.4331
##      rbcc3.28 - 3.87       rbcc3.87 - 4.46       rbcc4.46 - 5.05
##              -8.2644             -10.1964             -16.8601
##      rbcc5.05 - 5.64       rbcc5.64 - 6.23           wbcc12120+
##             -16.9109             -12.8342             -16.8655
##      wbcc4980 - 7360       wbcc7360 - 9740      wbcc9740 - 12120
##              -5.1394              -3.0657              -1.5650
##                  htn                   dm                  cad
##              -8.2528             -16.9826              -1.0689
##                appet                   pe                  ane
##              -6.8465              -0.2138               3.3077
## grf102.115 - 127.281  grf127.281 - 152.446  grf152.446 - 177.612
##              29.9462              37.2963              33.2783
##          grf177.612+  grf26.6175 - 51.7832   grf51.7832 - 76.949
##              29.2485              -7.7286              -6.6033
##  grf76.949 - 102.115              stages2              stages3
##              20.0539              35.9934              35.8698
##              stages4              stages5              age  74
##              36.5812              79.0508              20.8724
##           age12 - 20           age20 - 27           age27 - 35
##               2.9680              17.9262              13.8566
##           age35 - 43           age43 - 51           age51 - 59
##              10.9604               8.0547              12.0173
##           age59 - 66           age66 - 74
##               9.6285              18.2971
##
## Degrees of Freedom: 198 Total (i.e. Null);  137 Residual
## Null Deviance:       260.5
## Residual Deviance: 3.36e-09  AIC: 124
```

- Perform forward model selection, what model does this method select?

```
stepAIC(log_reg, direction = 'forward', trace = FALSE)
```

```
##
## Call:  glm(formula = class ~ (bp_diastolic + bp_limit + sg + al + rbc +
##     su + pc + pcc + ba + bgr + bu + sod + sc + pot + hemo + pcv +
##     rbcc + wbcc + htn + dm + cad + appet + pe + ane + grf + stage +
##     affected + age) - pot - affected, family = "binomial", data = dataset)
```

```
##
## Coefficients:
##          (Intercept)          bp_diastolic              bp_limit
##              -20.2501               29.1220              -25.1674
##           sg0 - 1.011       sg1.015 - 1.017       sg1.019 - 1.021
##              -36.9403              -22.1332               -3.8708
##                 al> 0                   rbc                  su0+
##              -11.7873               16.8065               -3.1922
##                     pc                   pcc                    ba
##                4.7276               16.5412               20.6456
##               bgr154+              bu> 86.2         bu48.1 - 86.2
##               -5.5104              -14.5855              -13.4243
##          sod133 - 138          sod138 - 143               sod143+
##              -10.3562              -12.5788                1.5505
##               sc3.65+       hemo11.3 - 12.6       hemo12.6 - 13.9
##              -44.8180              -10.6226               16.8069
##      hemo13.9 - 15.2              hemo15.2+             pcv  49.1
##               13.3892               20.0850               19.3612
##       pcv37.4 - 41.3        pcv41.3 - 49.1            rbcc> 6.23
##               24.4556               19.8083               -6.4331
##      rbcc3.28 - 3.87       rbcc3.87 - 4.46       rbcc4.46 - 5.05
##              -8.2644              -10.1964              -16.8601
##      rbcc5.05 - 5.64       rbcc5.64 - 6.23           wbcc12120+
##              -16.9109              -12.8342              -16.8655
##      wbcc4980 - 7360       wbcc7360 - 9740      wbcc9740 - 12120
##              -5.1394               -3.0657               -1.5650
##                  htn                    dm                   cad
##              -8.2528              -16.9826               -1.0689
##                appet                    pe                   ane
##              -6.8465               -0.2138                3.3077
## grf102.115 - 127.281  grf127.281 - 152.446  grf152.446 - 177.612
##              29.9462               37.2963               33.2783
##          grf177.612+  grf26.6175 - 51.7832   grf51.7832 - 76.949
##              29.2485               -7.7286               -6.6033
##  grf76.949 - 102.115               stages2               stages3
##              20.0539               35.9934               35.8698
##              stages4               stages5               age  74
##              36.5812               79.0508               20.8724
##           age12 - 20            age20 - 27            age27 - 35
##               2.9680               17.9262               13.8566
##           age35 - 43            age43 - 51            age51 - 59
##              10.9604                8.0547               12.0173
##           age59 - 66            age66 - 74
##               9.6285               18.2971
##
## Degrees of Freedom: 198 Total (i.e. Null);  137 Residual
## Null Deviance:       260.5
## Residual Deviance: 3.36e-09  AIC: 124
```

The model selects: class ~ (bp_diastolic + bp_limit + sg + al + rbc + su + pc + pcc + ba + bgr + bu + sod + sc + pot + hemo + pcv + rbcc + wbcc + htn + dm + cad + appet + pe + ane + grf + stage + affected + age Which is all the factors except for pot and affected

- Perform backward model selection, what model does this method select?

```
stepAIC(log_reg, direction = 'backward', trace = FALSE)
```

```
##
## Call:  glm(formula = class ~ bp_diastolic + bp_limit + al + su + hemo +
##     pcv + appet, family = "binomial", data = dataset)
##
## Coefficients:
##     (Intercept)     bp_diastolic         bp_limit            al> 0
##         -104.24           165.43          -124.20          -164.90
##            su0+  hemo11.3 - 12.6  hemo12.6 - 13.9  hemo13.9 - 15.2
##          -42.16            41.11            83.09            83.52
##       hemo15.2+        pcv  49.1   pcv37.4 - 41.3   pcv41.3 - 49.1
##           86.69            44.62            42.03            83.23
##           appet
##          -43.67
##
## Degrees of Freedom: 198 Total (i.e. Null);  186 Residual
## Null Deviance:       260.5
## Residual Deviance: 3.245e-08     AIC: 26
```

Backward selection selects the model: class ~ bp_diastolic + bp_limit + al + su + hemo + pcv + appet

- What is are the training and test errors for 5-fold CV for one of the models selected above? How does the classification rate from your model compare to the rate for the logistic regression from Islam et al. (2020).

```
set.seed(0216)

n <- nrow(dataset)

start <- c(1, 41, 81, 121, 161)
end <- c(40, 80, 120, 160, n)
acc <- numeric(5)
train_error <- numeric(5)
test_error <- numeric(5)
data_fold <- sample(1:n)

for (k in 1:5){
  test_index <- data_fold[data_fold[start[k]:end[k]]]
```

```
  test <- dataset[test_index, ]
  train <- dataset[-test_index,]

  log_reg <- glm(class ~ bp_diastolic + bp_limit + al + su + hemo + pcv + appet, data = train,

  train_pred_probs <- predict.glm(log_reg, newdata = train, type = 'response')
  test_pred_probs <- predict.glm(log_reg, newdata = test, type = 'response')

  train_preds <- ifelse(train_pred_probs < 0.5, 'ckd', 'notckd')
  test_preds <- ifelse(test_pred_probs < 0.5, 'ckd', 'notckd')

  train_table <- table(train_preds, train$class)
  test_table <- table(test_preds, test$class)

  train_acc <- sum(diag(train_table))/ sum(train_table)
  test_acc <- sum(diag(test_table)) / sum(test_table)

  acc[k] <- test_acc
  train_error[k] <- 1 - train_acc
  test_error[k] <- 1 - test_acc

}

print('Accuracy')
```

```
## [1] "Accuracy"
```

```
mean(acc)
```

```
## [1] 0.9747436
```

```
print('Train Error')
```

```
## [1] "Train Error"
```

```
mean(train_error)
```

```
## [1] 0
```

```
print('Test Error')
```

```
## [1] "Test Error"
```

```r
mean(test_error)
```

```
## [1] 0.02525641
```

My logistic regression has a classification rate of 97.47%, the original report has a rate of 94.77%

- Construct an appropriate confidence interval for your model.

```r
confint(log_reg, level = .95)
```

```
## Waiting for profiling to be done...
```

```
##                       2.5 %   97.5 %
## (Intercept)     -3193.789 2325.777
## bp_diastolic    -2862.380 2286.269
## bp_limit        -3155.186 3635.922
## al> 0           -2374.022 2077.881
## su0+                   NA 8757.488
## hemo11.3 - 12.6 -2266.675 1994.191
## hemo12.6 - 13.9 -2865.241 3373.090
## hemo13.9 - 15.2 -2069.964 1906.771
## hemo15.2+       -2560.377 2571.169
## pcv  49.1       -4090.118 3423.855
## pcv37.4 - 41.3  -2394.272 2696.413
## pcv41.3 - 49.1  -2584.428 2533.590
## appet           -2859.050 2671.465
```

LDA

- Construct model for the research question of Islam et al. (2020)

```r
lda <- lda(class ~. -pot - affected, data=dataset)

lda
```

```
## Call:
## lda(class ~ . - pot - affected, data = dataset)
##
## Prior probabilities of groups:
##      ckd   notckd
## 0.638191 0.361809
##
## Group means:
##        bp_diastolic  bp_limit sg0 - 1.011 sg1.015 - 1.017 sg1.019 - 1.021
```

```
## ckd       0.5748031 0.9133858   0.3779528       0.2834646       0.2913386
## notckd    0.4722222 0.4583333   0.0000000       0.0000000       0.5138889
##            al> 0       rbc      su0+        pc       pcc          ba   bgr154+
## ckd    0.6614173 0.1968504 0.2362205 0.3543307 0.2125984 0.08661417 0.4015748
## notckd 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000 0.00000000 0.0000000
##         bu> 86.2 bu48.1 - 86.2 sod133 - 138 sod138 - 143    sod143+   sc3.65+
## ckd    0.2992126     0.3464567    0.5433071    0.2204724 0.02362205 0.3149606
## notckd 0.0000000     0.1250000    0.3055556    0.2916667 0.40277778 0.0000000
##        hemo11.3 - 12.6 hemo12.6 - 13.9 hemo13.9 - 15.2 hemo15.2+ pcv  49.1
## ckd         0.34645669      0.05511811      0.03937008 0.0000000 0.0000000
## notckd      0.05555556      0.16666667      0.29166667 0.4861111 0.2638889
##        pcv37.4 - 41.3 pcv41.3 - 49.1  rbcc> 6.23 rbcc3.28 - 3.87
## ckd         0.3307087     0.04724409 0.007874016       0.1653543
## notckd      0.1805556     0.55555556 0.125000000       0.0000000
##        rbcc3.87 - 4.46 rbcc4.46 - 5.05 rbcc5.05 - 5.64 rbcc5.64 - 6.23
## ckd         0.15748031       0.5590551      0.01574803     0.007874016
## notckd      0.01388889       0.3333333      0.29166667     0.236111111
##        wbcc12120+ wbcc4980 - 7360 wbcc7360 - 9740 wbcc9740 - 12120       htn
## ckd     0.1259843       0.1653543       0.5196850        0.1417323 0.6141732
## notckd  0.0000000       0.3611111       0.4305556        0.1527778 0.0000000
##                dm       cad     appet        pe       ane grf102.115 - 127.281
## ckd    0.5433071 0.1732283 0.3149606 0.2755906 0.2519685           0.02362205
## notckd 0.0000000 0.0000000 0.0000000 0.0000000 0.0000000           0.16666667
##        grf127.281 - 152.446 grf152.446 - 177.612 grf177.612+
## ckd              0.02362205                0.000  0.02362205
## notckd           0.11111111                0.125  0.13888889
##        grf26.6175 - 51.7832 grf51.7832 - 76.949 grf76.949 - 102.115    stages2
## ckd              0.28346457          0.09448819          0.03149606 0.09448819
## notckd           0.02777778          0.22222222          0.18055556 0.31944444
##          stages3    stages4   stages5    age  74 age12 - 20 age20 - 27
## ckd    0.2440945 0.32283465 0.2677165 0.05511811 0.02362205 0.02362205
## notckd 0.0000000 0.05555556 0.0000000 0.04166667 0.01388889 0.09722222
##        age27 - 35 age35 - 43 age43 - 51 age51 - 59 age59 - 66 age66 - 74
## ckd    0.02362205 0.03937008  0.1417323  0.1574803  0.2992126 0.20472441
## notckd 0.15277778 0.09722222  0.1805556  0.1805556  0.1388889 0.09722222
##
## Coefficients of linear discriminants:
##                               LD1
## bp_diastolic          0.378722160
## bp_limit             -0.575406343
## sg0 - 1.011          -1.704368257
## sg1.015 - 1.017      -1.512180521
## sg1.019 - 1.021      -0.471197227
## al> 0                -0.666380411
## rbc                   0.269147502
## su0+                 -0.046629720
## pc                    0.274284787
## pcc                   0.366157934
```

```
## ba                    0.488824925
## bgr154+              -0.021378114
## bu> 86.2              0.027187211
## bu48.1 - 86.2         0.076998137
## sod133 - 138         -0.143037706
## sod138 - 143         -0.164697919
## sod143+               0.740082827
## sc3.65+              -0.266148689
## hemo11.3 - 12.6      -0.116305832
## hemo12.6 - 13.9       1.372226268
## hemo13.9 - 15.2       1.788565420
## hemo15.2+             2.658981562
## pcv  49.1             1.302998438
## pcv37.4 - 41.3        0.136373661
## pcv41.3 - 49.1        1.160975444
## rbcc> 6.23            0.324553665
## rbcc3.28 - 3.87      -0.310105930
## rbcc3.87 - 4.46      -0.159100518
## rbcc4.46 - 5.05      -0.680452610
## rbcc5.05 - 5.64      -0.113603609
## rbcc5.64 - 6.23      -0.239958494
## wbcc12120+            0.180778116
## wbcc4980 - 7360       0.012588702
## wbcc7360 - 9740      -0.077350188
## wbcc9740 - 12120     -0.116452663
## htn                  -0.534736735
## dm                   -0.331332311
## cad                   0.168745164
## appet                -0.210621280
## pe                   -0.223692550
## ane                  -0.076019487
## grf102.115 - 127.281 -0.044195888
## grf127.281 - 152.446  0.501670383
## grf152.446 - 177.612  0.755745904
## grf177.612+           0.538903982
## grf26.6175 - 51.7832  0.001957458
## grf51.7832 - 76.949   0.361443805
## grf76.949 - 102.115   0.555548475
## stages2              -0.950063995
## stages3              -1.476112497
## stages4              -1.200191850
## stages5              -1.143526856
## age  74               3.219647168
## age12 - 20            0.798829231
## age20 - 27            3.293597877
## age27 - 35            2.756010602
## age35 - 43            2.640865444
## age43 - 51            2.917393892
```

```
## age51 - 59            3.164649839
## age59 - 66            2.855249969
## age66 - 74            2.730978150
```

- What is are the training and test errors for 5-fold CV for this model? How does the classification rate from your model compare to the rate for the Naive Bayes from Islam et al. (2020).

```
set.seed(0216)

n <- nrow(dataset)

start <- c(1, 41, 81, 121, 161)
end <- c(40, 80, 120, 160, n)
acc <- numeric(5)
train_error <- numeric(5)
test_error <- numeric(5)
data_fold <- sample(1:n)

for (k in 1:5){
  test_index <- data_fold[data_fold[start[k]:end[k]]]
  test <- dataset[test_index, ]
  train <- dataset[-test_index,]

  #Used model selected by backward selection for log_reg, AICstep doesnt work with LDA
  lda <- lda(class ~ bp_diastolic + bp_limit + al + su + hemo + pcv + appet, data = train)

  train_pred <- predict(lda, newdata = train)$class
  test_pred <- predict(lda, newdata = test)$class


  train_table <- table(train_pred, train$class)
  test_table <- table(test_pred, test$class)

  train_acc <- sum(diag(train_table))/ sum(train_table)
  test_acc <- sum(diag(test_table)) / sum(test_table)

  acc[k] <- test_acc
  train_error[k] <- 1 - train_acc
  test_error[k] <- 1 - test_acc

}

print('Accuracy')
```

```
## [1] "Accuracy"
```

```r
mean(acc)
```

```
## [1] 0.9446154
```

```r
print('Train Error')
```

```
## [1] "Train Error"
```

```r
mean(train_error)
```

```
## [1] 0.04021226
```

```r
print('Test Error')
```

```
## [1] "Test Error"
```

```r
mean(test_error)
```

```
## [1] 0.05538462
```

As the original report did not test with LDA, my LDA accuracy was 94.46%, and the naive bayes was 93.91%.

Naive Bayes

- Construct model for the research question of Islam et al. (2020)

```r
bayes <- naiveBayes(class ~. -pot - affected, data = dataset)

bayes
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##      ckd   notckd
## 0.638191 0.361809
##
```

```
## Conditional probabilities:
##         bp_diastolic
## Y             [,1]      [,2]
##   ckd    0.5748031 0.4963307
##   notckd 0.4722222 0.5027312
##
##         bp_limit
## Y             [,1]      [,2]
##   ckd    0.9133858 0.8910812
##   notckd 0.4583333 0.5017575
##
##         sg
## Y            1.023  0 - 1.011 1.015 - 1.017 1.019 - 1.021
##   ckd    0.04724409 0.37795276    0.28346457    0.29133858
##   notckd 0.48611111 0.00000000    0.00000000    0.51388889
##
##         al
## Y              < 0       > 0
##   ckd    0.3385827 0.6614173
##   notckd 1.0000000 0.0000000
##
##         rbc
## Y             [,1]     [,2]
##   ckd    0.1968504 0.399193
##   notckd 0.0000000 0.000000
##
##         su
## Y              < 0        0+
##   ckd    0.7637795 0.2362205
##   notckd 1.0000000 0.0000000
##
##         pc
## Y             [,1]      [,2]
##   ckd    0.3543307 0.4802043
##   notckd 0.0000000 0.0000000
##
##         pcc
## Y             [,1]      [,2]
##   ckd    0.2125984 0.4107662
##   notckd 0.0000000 0.0000000
##
##         ba
## Y              [,1]      [,2]
##   ckd    0.08661417 0.2823828
##   notckd 0.00000000 0.0000000
##
##         bgr
## Y           < 154      154+
```

```
##   ckd    0.5984252 0.4015748
##   notckd 1.0000000 0.0000000
##
##         bu
## Y           < 48.1    > 86.2 48.1 - 86.2
##   ckd    0.3543307 0.2992126   0.3464567
##   notckd 0.8750000 0.0000000   0.1250000
##
##         sod
## Y             < 133  133 - 138  138 - 143       143+
##   ckd    0.21259843 0.54330709 0.22047244 0.02362205
##   notckd 0.00000000 0.30555556 0.29166667 0.40277778
##
##         sc
## Y           < 3.65     3.65+
##   ckd    0.6850394 0.3149606
##   notckd 1.0000000 0.0000000
##
##         hemo
## Y            < 11.3 11.3 - 12.6 12.6 - 13.9 13.9 - 15.2      15.2+
##   ckd    0.55905512  0.34645669  0.05511811  0.03937008 0.00000000
##   notckd 0.00000000  0.05555556  0.16666667  0.29166667 0.48611111
##
##         pcv
## Y            < 37.4       49.1 37.4 - 41.3 41.3 - 49.1
##   ckd    0.62204724 0.00000000  0.33070866  0.04724409
##   notckd 0.00000000 0.26388889  0.18055556  0.55555556
##
##         rbcc
## Y             < 3.28      > 6.23 3.28 - 3.87 3.87 - 4.46 4.46 - 5.05
##   ckd    0.086614173 0.007874016 0.165354331 0.157480315 0.559055118
##   notckd 0.000000000 0.125000000 0.000000000 0.013888889 0.333333333
##         rbcc
## Y        5.05 - 5.64 5.64 - 6.23
##   ckd    0.015748031 0.007874016
##   notckd 0.291666667 0.236111111
##
##         wbcc
## Y            < 4980     12120+ 4980 - 7360 7360 - 9740 9740 - 12120
##   ckd    0.04724409 0.12598425  0.16535433  0.51968504   0.14173228
##   notckd 0.05555556 0.00000000  0.36111111  0.43055556   0.15277778
##
##         htn
## Y             [,1]      [,2]
##   ckd    0.6141732 0.4887179
##   notckd 0.0000000 0.0000000
##
##         dm
```

```
## Y             [,1]      [,2]
##   ckd    0.5433071 0.5000937
##   notckd 0.0000000 0.0000000
##
##         cad
## Y             [,1]      [,2]
##   ckd    0.1732283 0.3799434
##   notckd 0.0000000 0.0000000
##
##         appet
## Y             [,1]      [,2]
##   ckd    0.3149606 0.4663398
##   notckd 0.0000000 0.0000000
##
##         pe
## Y             [,1]      [,2]
##   ckd    0.2755906 0.4485809
##   notckd 0.0000000 0.0000000
##
##         ane
## Y             [,1]      [,2]
##   ckd    0.2519685 0.4358627
##   notckd 0.0000000 0.0000000
##
##         grf
## Y         < 26.6175 102.115 - 127.281 127.281 - 152.446 152.446 - 177.612
##   ckd    0.51968504        0.02362205        0.02362205        0.00000000
##   notckd 0.02777778        0.16666667        0.11111111        0.12500000
##         grf
## Y          177.612+ 26.6175 - 51.7832 51.7832 - 76.949 76.949 - 102.115
##   ckd    0.02362205        0.28346457       0.09448819       0.03149606
##   notckd 0.13888889        0.02777778       0.22222222       0.18055556
##
##         stage
## Y                 s1         s2         s3         s4         s5
##   ckd    0.07086614 0.09448819 0.24409449 0.32283465 0.26771654
##   notckd 0.62500000 0.31944444 0.00000000 0.05555556 0.00000000
##
##         age
## Y              < 12         74    12 - 20    20 - 27    27 - 35    35 - 43
##   ckd    0.03149606 0.05511811 0.02362205 0.02362205 0.02362205 0.03937008
##   notckd 0.00000000 0.04166667 0.01388889 0.09722222 0.15277778 0.09722222
##         age
## Y           43 - 51    51 - 59    59 - 66    66 - 74
##   ckd    0.14173228 0.15748031 0.29921260 0.20472441
##   notckd 0.18055556 0.18055556 0.13888889 0.09722222
```

- What is are the training and test errors for 5-fold CV for this model? How does the classifi-

cation rate from your model compare to the rate for the Naive Bayes from Islam et al. (2020).

```
set.seed(0216)

n <- nrow(dataset)

start <- c(1, 41, 81, 121, 161)
end <- c(40, 80, 120, 160, n)
acc <- numeric(5)
train_error <- numeric(5)
test_error <- numeric(5)
data_fold <- sample(1:n)

for (k in 1:5){
  test_index <- data_fold[data_fold[start[k]:end[k]]]
  test <- dataset[test_index, ]
  train <- dataset[-test_index,]

  # Used the model selected by backward selection for logistic regression bc AICstep isnt mean
  bayes <- naiveBayes(class ~ bp_diastolic + bp_limit + al + su + hemo + pcv + appet, data = tr

  train_pred <- predict(bayes, newdata = train)
  test_pred <- predict(bayes, newdata = test)

  train_table <- table(train_pred, train$class)
  test_table <- table(test_pred, test$class)

  train_acc <- sum(diag(train_table))/ sum(train_table)
  test_acc <- sum(diag(test_table)) / sum(test_table)

  acc[k] <- test_acc
  train_error[k] <- 1 - train_acc
  test_error[k] <- 1 - test_acc

}

print('Accuracy')
```

```
## [1] "Accuracy"
```

```
mean(acc)
```

```
## [1] 0.9296154
```

```r
print('Train Error')
```

```
## [1] "Train Error"
```

```r
mean(train_error)
```

```
## [1] 0.06408019
```

```r
print('Test Error')
```

```
## [1] "Test Error"
```

```r
mean(test_error)
```

```
## [1] 0.07038462
```

The accuracy of my Naive Bayes model is 92.96% and the original report had a rate of 93.91%.

Decision Tree Methods

- Construct a decision tree for the research question of Islam et al. (2020). Use the Gini index as the training-loss.

```r
set.seed(0216)
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.4.0
```

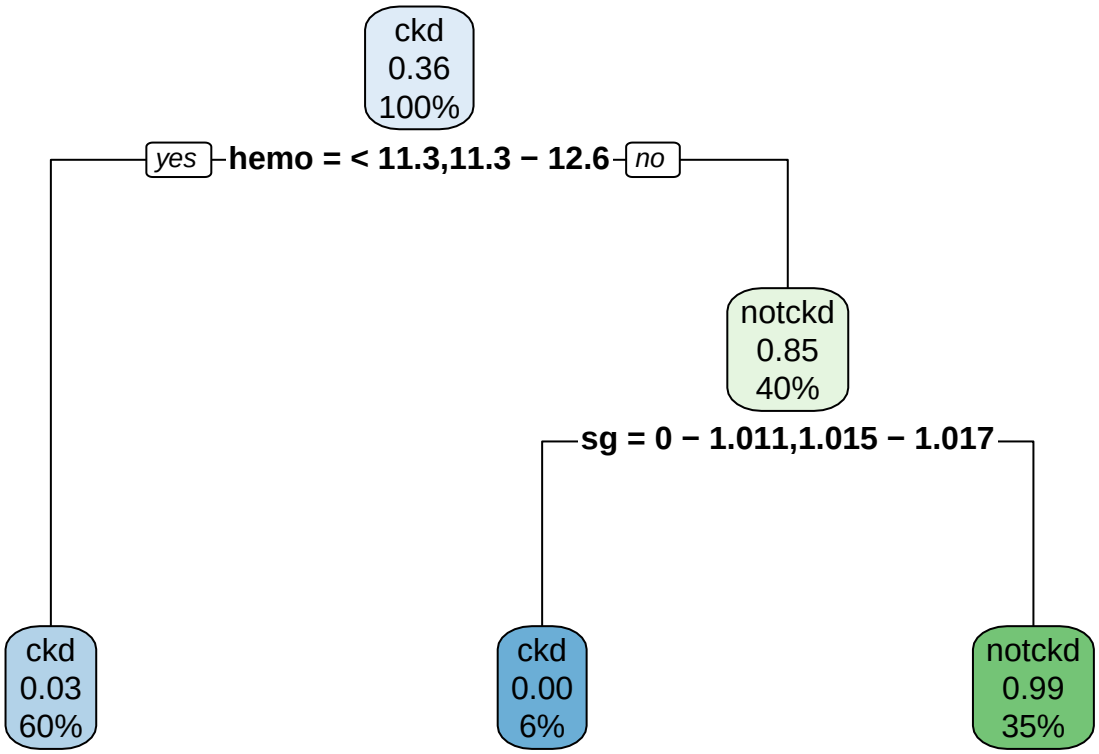
```r
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.2.3
```

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

```
## Warning: package 'rpart.plot' was built under R version 4.2.3
```

```r
dec_tree = rpart(class ~ . - pot - affected, data = dataset, method = 'class', parms = list(sp
rpart.plot(dec_tree)
```




```
cp.min <- dec_tree$cptable[which.min(dec_tree$cptable[,"xerror"]),"CP"]
```

- Use CV to choose the optimal pruning for your decision-tree model, what model does this method select?

```
library(partykit)
```

```
## Warning: package 'partykit' was built under R version 4.2.3
```

```
## Loading required package: grid
```

```
## Loading required package: libcoin
```

```
## Warning: package 'libcoin' was built under R version 4.2.3
```

```
## Loading required package: mvtnorm
```

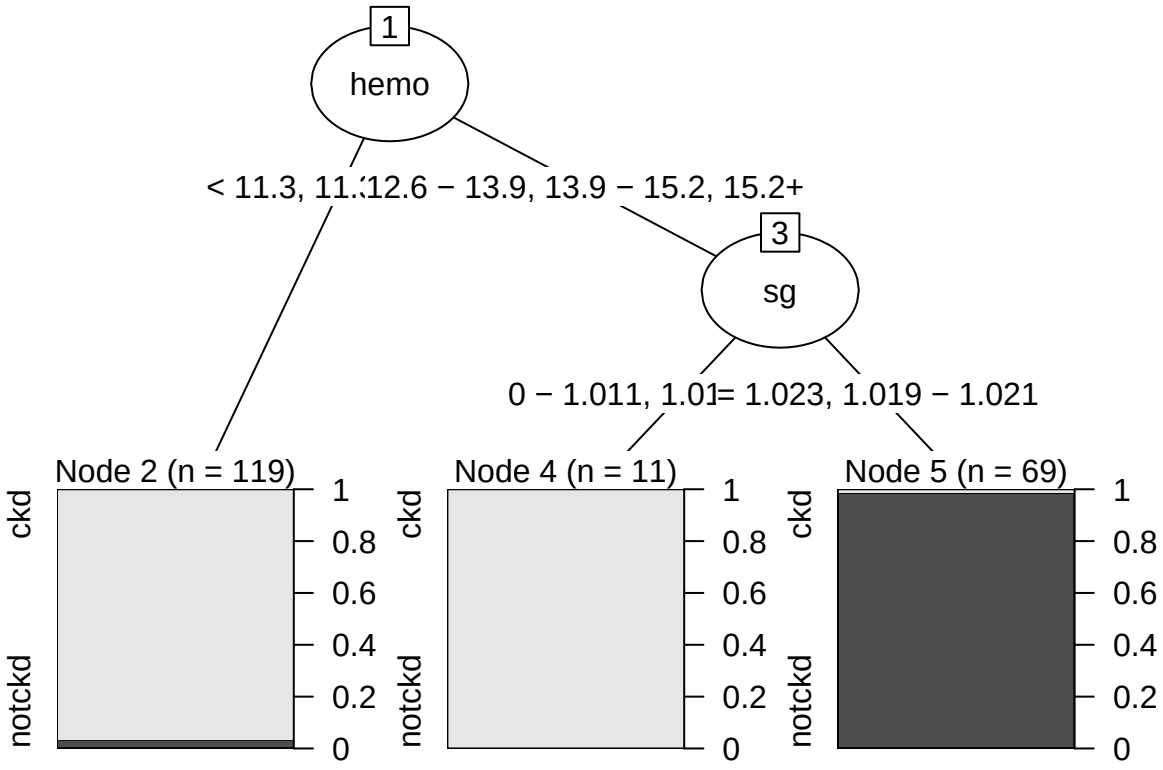
```
## Warning: package 'mvtnorm' was built under R version 4.4.0
```

```
pruning_tree <- prune(dec_tree, cp = cp.min)

#pruning_tree

#plot(pruning_tree)

dec_tree_party <- as.party(pruning_tree)
plot(dec_tree_party)
```



The model selected is: class ~ hemo + pcv + stage + grf + rbcc + sg as these are the most important variables designated

- What is are the training and test errors for 5-fold CV for this model? How does the classification rate from your model compare to the rate for the tree-based classifier from Islam et al. (2020).

```
set.seed(0216)

n <- nrow(dataset)

start <- c(1, 41, 81, 121, 161)
end <- c(40, 80, 120, 160, n)
```

```r
acc <- numeric(5)
train_error <- numeric(5)
test_error <- numeric(5)
data_fold <- sample(1:n)

for (k in 1:5){
  test_index <- data_fold[data_fold[start[k]:end[k]]]
  test <- dataset[test_index, ]
  train <- dataset[-test_index,]

  pruning_tree <- prune(dec_tree, cp = cp.min)

  train_pred <- predict(pruning_tree, newdata = train, type = 'class')
  test_pred <- predict(pruning_tree, newdata = test, type = 'class')

  train_table <- table(train_pred, train$class)
  test_table <- table(test_pred, test$class)

  train_acc <- sum(diag(train_table))/ sum(train_table)
  test_acc <- sum(diag(test_table)) / sum(test_table)

  acc[k] <- test_acc
  train_error[k] <- 1 - train_acc
  test_error[k] <- 1 - test_acc

}

print('Accuracy')
```

```
## [1] "Accuracy"
```

```r
mean(acc)
```

```
## [1] 0.9747436
```

```r
print('Train Error')
```

```
## [1] "Train Error"
```

```r
mean(train_error)
```

```
## [1] 0.02513365
```

```r
print('Test Error')
```

```
## [1] "Test Error"
```

```r
mean(test_error)
```

```
## [1] 0.02525641
```

The classification rate of mine is 97.47% and the rate of the tree based classifier in the original report was 98.89%.

```r
go <- randomForest(class ~.-pot-affected, dataset)
importance(go)
```

```
##              MeanDecreaseGini
## bp_diastolic       0.21249210
## bp_limit           1.75379959
## sg                 9.50081429
## al                 5.42057208
## rbc                0.28540630
## su                 0.51369789
## pc                 0.43950206
## pcc                0.05594763
## ba                 0.03955162
## bgr                1.47737685
## bu                 1.33603047
## sod                1.98725452
## sc                 0.16727308
## hemo              19.54071630
## pcv               13.39097005
## rbcc               6.50639681
## wbcc               0.42538128
## htn                4.05212388
## dm                 2.21093919
## cad                0.01757722
## appet              0.62569543
## pe                 0.51343157
## ane                0.12015236
## grf                8.08963246
## stage             10.40009703
## age                1.93634924
```

sg + al + hemo + pcv + grf + stage + rbcc

- Repeat the above procedure using bagging? What is the training error for this model?

```r
library(randomForest)
set.seed(0216)

n <- nrow(dataset)

start <- c(1, 41, 81, 121, 161)
end <- c(40, 80, 120, 160, n)
acc <- numeric(5)
train_error <- numeric(5)
test_error <- numeric(5)
data_fold <- sample(1:n)

for (k in 1:5){
  test_index <- data_fold[data_fold[start[k]:end[k]]]
  test <- dataset[test_index, ]
  train <- dataset[-test_index,]
  # Selected model from most important variables identified by randomForest
  forest <- randomForest(class ~ sg + al + rbcc + hemo + pcv + grf + stage, train)

  train_pred <- predict(forest, newdata = train)
  test_pred <- predict(forest, newdata = test)

  train_table <- table(train_pred, train$class)
  test_table <- table(test_pred, test$class)

  train_acc <- sum(diag(train_table))/ sum(train_table)
  test_acc <- sum(diag(test_table)) / sum(test_table)

  acc[k] <- test_acc
  train_error[k] <- 1 - train_acc
  test_error[k] <- 1 - test_acc

}

print('Accuracy')
```

```
## [1] "Accuracy"
```

```r
mean(acc)
```

```
## [1] 0.9644872
```

```r
print('Train Error')
```

```
## [1] "Train Error"
```

```r
mean(train_error)
```

```
## [1] 0
```

```r
print('Test Error')
```

```
## [1] "Test Error"
```

```r
mean(test_error)
```

```
## [1] 0.03551282
```

- Repeat the above procedure using boosting? What is the training error for this model?

I cannot run a boosting model without breaking RStudio

```r
library(gbm)
```

```
## Warning: package 'gbm' was built under R version 4.4.0
```

```
## Loaded gbm 2.1.9
```

```
## This version of gbm is no longer under development. Consider transitioning to gbm3, https:/
```

```r
library(plyr)
```

```
## Warning: package 'plyr' was built under R version 4.2.3
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

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

```
# Requires factor to be {0,1}. When I turn class into {0,1}, it causes a termnial error in R
# When I run gbm, it breaks my markdown and all variables become characters without way of cha

#boosted <- gbm(class ~. -pot - affected, data = dataset, interaction.depth = 5)
#summary(boosted)
```

- Which tree-based model preforms the best?

The pruned tree perfomrd the best, with roughly 97% accuracy.

Summary of Findings

Provide a high level non-technical overview of the project.

- Discuss the original research question and any issues with the original findings.
- Summarize your findings for the models for chronic kidney disease.
- Are you able to produce similar results to the original paper? Are you able to improve upon the previously existing results?
- Which model would you recommend to experts if they were interested in the research question?

The orginal research question looked to find the best model to determine variables that are predictors of CKD. The original findings didnt really utilize model selection or validation sets. I was able to produce very similar results to the original study, although mine did not improve upon their results. If I were to recommend a model to experts, I would recommend logistic regression or a simple pruned classification tree as they resulted in a similar, high accuracy point.