Assignment 1

Biomedical Data Science

Johnny Lee, s1687781

Problem 1 (25 points)

Files longegfr1.csv and longegfr2.csv (available on Learn) contain information regarding a longitudinal dataset containing records on 250 patients. For each subject, eGFR (estimated glomerular filtration rate, a measure of kidney function) was collected at irregularly spaced time points: variable "fu.years" contains the follow-up time (that is, the distance from baseline to the date when each eGFR measurement was taken, expressed in years).

Problem 1.a (4 points)

Convert the files to data tables and merge in an appropriate way into a single data table, then order the observations according to subject identifier and follow-up time.

Answer

```
v1 <- (read.csv("data_assignment1/1_longegfr1.csv"))
v2 <- (read.csv("data_assignment1/1_longegfr2.csv"))

#merging two dataset by id and follow-up years
data <- merge(v1, v2, by = c("id", "fu.years"),
              by.y = c("ID", "fu.years"), all = TRUE)
data <- setDT(data)

head(data, 10)
```

```
##     id fu.years sex baseline.age  egfr
##  1:  1   0.0000   0         65.5 76.48
##  2:  1   0.1533   0         65.5 47.36
##  3:  1   0.6899   0         65.5 94.87
##  4:  1   1.1882   0         65.5 52.12
##  5:  1   1.8398   0         65.5 91.91
##  6:  1   2.2806   0         65.5 76.52
##  7:  1   3.3895   0         65.5 46.79
##  8:  1   3.7563   0         65.5 35.56
##  9:  1   4.5229   0         65.5 28.41
## 10:  1   5.3607   0         65.5 20.85
```

By scrutinising the dataset, we realised that the columns `id` and `fu.years` are in common. As a result, we merge this two dataset by the two columns. Thus, we have a 4031 × 5 data table. Also `merge()` function does the ordering by itself thus we conclude this answer as above.

Problem 1.b (6 points)

Compute the average eGFR and length of follow-up for each patient, then tabulate the number of patients with average eGFR in the following ranges: (0, 15], (15, 30], (30, 60], (60,90], (90, max(eGFR)). Count and report the number of patients with missing average eGFR.

Answer

```
#initialising the vector to contain the average eGFR and length of follow-up
meanegfr <- lengthoffu <- c()
for(i in seq(1:length(unique(data[,id])))){
  #calculating the average eGFR by id
  patientmean <- mean(data[data$id==i]$egfr)
  #calculating the length of follow-up by id
  followup <- max(data[data$id==i]$fu.years)
  #storing into the intialised empty vectors
  meanegfr <- c(meanegfr, patientmean)
  lengthoffu <- c(lengthoffu, followup)
}

#tabulating the average eGFR and length of follow-up
mean.length <- data.frame(id = unique(data[,id]), meanegfr, lengthoffu)

head(mean.length, 10)
```

```
##    id  meanegfr lengthoffu
## 1   1  43.04333     6.4586
## 2   2  38.93294     2.0698
## 3   3  85.72000     6.5161
## 4   4  76.59308     5.2786
## 5   5        NA     6.3929
## 6   6  85.66435     6.2313
## 7   7  64.21758     5.8453
## 8   8  66.28333     1.5606
## 9   9  86.35750     5.8700
## 10 10 107.00429     5.1964
```

```
#tabulating the number of patients with average eGFR in the given ranges
table(cut(meanegfr, c(0,15,30,60,90, max(meanegfr, na.rm=TRUE))))
```

```
## 
##   (0,15]  (15,30]  (30,60]  (60,90] (90,148] 
##        1        9       83       82       36
```

```
#computing the numbers of patients with missing average eGFR
cat("number of patients with missing average eGFR:", sum(is.na(meanegfr)))
```

```
## number of patients with missing average eGFR: 39
```

Problem 1.c (6 points)

For patients with average eGFR in the (90,`max(eGFR)`) range, collect in a data table (or tibble) their identifier, sex, age at baseline, average eGFR, time of last eGFR reading and number of eGFR measurements taken.

Answer

```r
#storing the index that has the average eGFR greater than 90
idx <- c()
for (i in meanegfr){
  if(is.na(i) == FALSE){
    if(i > 90){idx <- c(idx, which(meanegfr == i))}
  }
}

#extracting those selected indexes from above
data90max <- data[id %in% idx]

data90max <- data90max %>%
  # Counting the number of eGFR measurements
  .[, no.eGFR := NROW(egfr), by = id] %>%
  # Computing the average eGFR by id
  .[, average.eGFR := mean(egfr, na.rm = TRUE), by = id] %>%
  group_by(id) %>%
  # Computing the time of last eGFR reading by id
  top_n(1, fu.years) %>%
  # removing egfr columns
  select(-egfr)

data90max <- as.data.table(data90max)
#setting orders given by the question
setcolorder(data90max, c("id", "sex", "baseline.age", "average.eGFR",
                         "fu.years", "no.eGFR"))

head(data90max, 15)
```

```
##      id sex baseline.age average.eGFR fu.years no.eGFR
##  1:  10   0         50.4    107.00429   5.1964       7
##  2:  14   0         65.1    116.09200   4.0986      10
##  3:  25   0         40.1     95.35625   4.2847       8
##  4:  31   0         74.8    113.59250   1.4675       8
##  5:  33   0         74.2    116.35000   1.6016       4
##  6:  45   1         24.9     91.25000   0.0000       1
##  7:  49   1         68.2    128.25800   6.1602       5
##  8:  52   1         56.3     93.31544   6.4805      57
##  9:  79   0         65.6     91.45057   5.2156      35
## 10:  80   0         67.7    106.09600   2.2834       5
## 11:  81   0         38.8    108.32000   5.7823       8
## 12:  92   1         41.2    101.33882   5.9713      17
## 13: 100   0         63.0    101.86769   6.5708      13
## 14: 102   0         38.7    105.96000   3.6934      10
## 15: 112   1         77.8     90.66500   5.0377       6
```

Problem 1.d (9 points)

For patients 3, 37, 162 and 223: * Plot the patient's eGFR measurements as a function of time. * Fit a linear regression model and add the regression line to the plot. * Report the 95% confidence interval for the regression coefficients of the fitted model. * Using a different colour, plot a second regression line computed after removing the extreme eGFR values (one each of the highest and the lowest value).

The plots should be appropriately labelled and the results should be accompanied by some explanation as you would communicate it to a colleague with a medical rather than statistical background.

Answer

```
par(mfrow = c(2,2), mar = c(1.5,1.5,1.5,1.5), oma = c(4,4,2.5,2.5))
for (i in c(3, 37, 162, 223)){
  #storing new data table that contains the selected id
  patientdata <- data[data$id == i]
  patientdata
  #fiting the time series model of eGFR
  fit1 <- lm(egfr ~ fu.years, data = patientdata)
  #95% confidence interval of the regression coefficients
  cat("95% confidence interval of fit1 and fit2 when id =", i, "\n")
  print(confint(fit1))
  #removing the extreme eGFR values
  newdata <- patientdata %>%
    #arranging by ascending order to find the minima and maxima
    arrange(egfr) %>%
    na.omit() %>%
    #removing the two extreme values
    slice(2:(n() - 1))
  #fiting the time series model of eGFR after removal of extremas
  fit2 <- lm(egfr ~ fu.years, data = newdata)
  #95% confidence interval of the regression coefficients
  print(confint(fit2))
  #scatter plot and the two fitted line or separate regressions
  #scatter plot
  plot(patientdata$fu.years, patientdata$egfr,
       main = paste("Time Series of eGFR, id =", i), cex = 0.5)
  #fitted lines of the regressions
  abline(fit1, col="red")
  abline(fit2, col="blue")
  legend("topright", legend = c("before removal", "after removal"),
         col = c("red", "blue"), lty = 1, cex = 0.6, bty = "n")
}
```

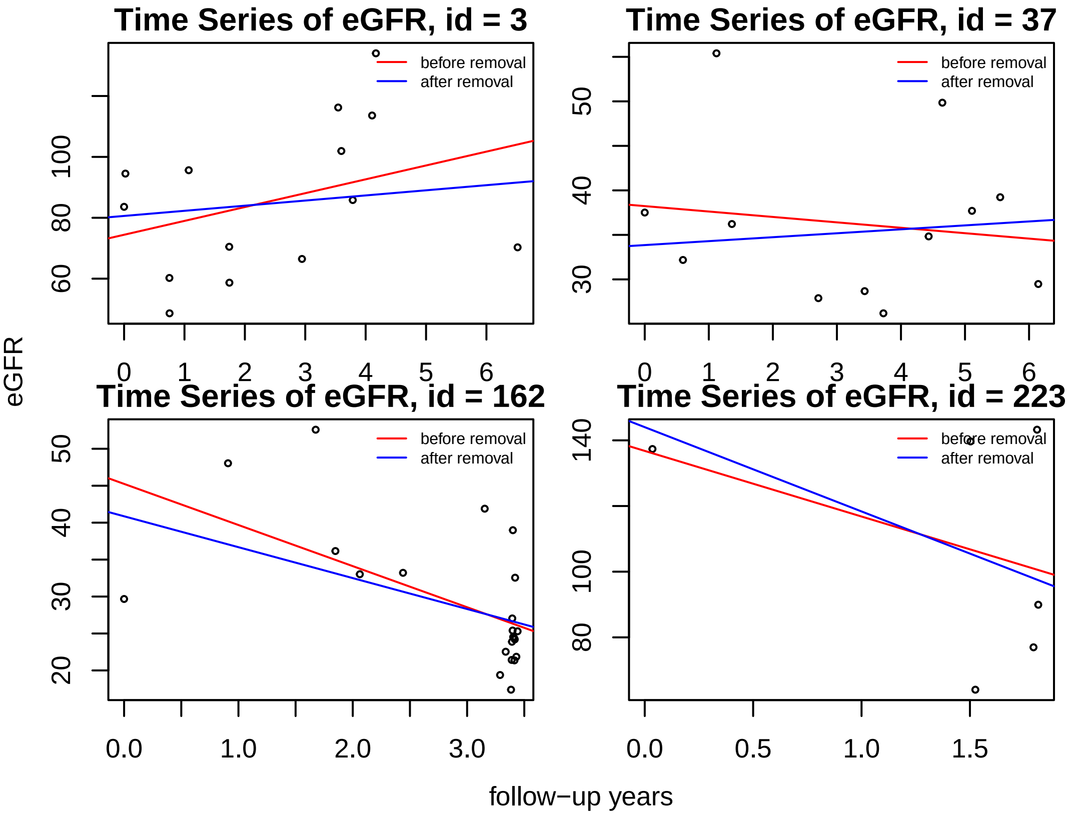
```
## 95% confidence interval of fit1 and fit2 when id = 3
##                  2.5 %   97.5 %
## (Intercept) 50.623768 98.21718
## fu.years    -3.151128 12.25612
##                  2.5 %     97.5 %
## (Intercept) 58.481407 102.720649
## fu.years    -5.441923   8.809287
```

```
## 95% confidence interval of fit1 and fit2 when id = 37
##                  2.5 %   97.5 %
## (Intercept) 26.911518 49.55334
## fu.years    -3.595705  2.37859
##                 2.5 %    97.5 %
## (Intercept) 24.189632 43.516722
## fu.years    -1.994624  2.879692
```

```
## 95% confidence interval of fit1 and fit2 when id = 162
##                 2.5 %    97.5 %
## (Intercept) 34.109333 56.382006
## fu.years    -9.257727 -1.872262
##                 2.5 %     97.5 %
## (Intercept) 30.565165 51.1595582
## fu.years    -7.562125 -0.8057698
```

```
## 95% confidence interval of fit1 and fit2 when id = 223
##                 2.5 %   97.5 %
## (Intercept)  34.71838 238.8642
## fu.years    -85.93757  45.9659
##                  2.5 %    97.5 %
## (Intercept)   17.14493 270.89855
## fu.years    -111.35297  60.00585
```

```
title(xlab = "follow-up years",
      ylab = "eGFR",
      outer = TRUE, line = 1)
```



eGFR stands for estimated Glomerular Filtration Rate which measures the functionality of patient's kidney and 60 or more is considered normal according to National Kidney Foundation. Also, the average measure of eGFR decreases with the decrease in age. Now we look at the plot. In the plot, patients have different number of measurements (data points) over different time range and this indicates that all patients are in different condition at the current measure. Thus we will describe them one by one.

First, we elaborate for the patient id, 3. The values of the eGFR of this patient suggest the healthy functionality of the kidney. The general trend of the graph is increasing as the follow-up year increases. This supports the fact that the patient is improving its kidney functionality. After removing the two extreme values, the gradient of the fitted line decreased and suggests that the change in eGFR through out the years is lower. With the noticeable trend, we can conclude that this indicates good kidney health of the patient.

Secondly, we elaborate for the patient id, 37. The patient seem to have a bad kidney funcationality or either considered old as the values of the plot suggested. The general trend of the graph is differs as the follow-up year increases before and after the removal of extreme. Before removing, we see the negative gradient whereas positive gradient for the fitted lines. Since removing the two extreme values is also acting as removing the outliers, we will continue with the analysis after removal. Then, we see that the kidney functionality of the patients improves gradually as the time passes and conclude that the patient is getting healthier than before.

Thirdly, we elaborate for the patient id, 162. This patient has less measured data compared to patient id 3 and 37. This shows that the patient has started to suffer from the kidney disease in more recent years and the age of the patient is either old or in bad state of kidney. The general trend of the graph is decreasing through out the years and it suggests that the kidney functionality of the patient is becoming worse. Moreover, after removing the extreme values, the general trend remained the same. We also see more values were collected in the recent years. This indicates that the patient can possibly be in a serious state undergoing intensive care with multiple measurement before medication.

Lastly, we elaborate for the patient id, 223. Similar to patient id 162, it has a decreasing trend with steep gradient but the age or the condition of the kidney seem to be relatively young and better respectively. Also, severity of the kidney condition is not in a serious stage as all the measurements are above 60 and the follow-up years are shorter than the rest of the other. Although the patient is having eGFR values higher than 60, the patient should be aware of its kidney condition and take medication to prevent further decrease in the kidney functionality. However, for this patient we are not solid towards this analysis as it contains a missing value. To be more accurate, we might have to collect more medical records throughout the years.

Problem 2 (25 points)

The MDRD4 and CKD-EPI equations are two different ways of estimating the glomerular filtration rate (eGFR) in adults:

$$\text{MDRD4} = 175 \times (\text{Scr})^{-1.154} \times \text{Age}^{-0.203}[\times 0.742 \text{ if female}][\times 1.212 \text{ if black}]$$

, and

$$\text{CKD-EPI} = 141 \times \min(\text{Scr}/\kappa, 1)^{\alpha} \times \max(\text{Scr}/\kappa, 1)^{-1.209} \times 0.993^{\text{Age}}[\times 1.018 \text{ if female}][\times 1.159 \text{ if black}]$$

, (1) where:
* SCR is serum creatinine (in mg/dL)
* κ is 0.7 for females and 0.9 for males
* α is -0.329 for females and -0.411 for males

Problem 2.a (7 points)

For the scr.csv dataset available on Learn, examine a summary of the distribution of serum creatinine and report the inter-quartile range. If you suspect that some serum creatinine values may have been reported in µmol/L convert them to mg/dL by dividing by 88.42. Justify your choice of values to convert and examine the distribution of serum creatinine following any changes you have made.

Answer

```r
scr <- setDT(read.csv("data_assignment1/2_scr.csv"))
summary(scr$scr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.400   0.900   1.300   3.072   2.800  76.000      18
```

```r
for (i in seq(1:nrow(scr))){
  #we first only consider the values that are not missing
  if (is.na(scr$scr[i])==FALSE){
    # we then consider for the values that are greater than 0.4 * 88.42
    if (scr$scr[i] > min(na.omit(scr$scr)*88.42)){
      # converting from µmol/L to mg/dL
      scr$scr[i] <- scr$scr[i]/88.42
    }
  }
}

summary(scr$scr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   0.400   0.900   1.200   2.752   2.800  32.000      18
```

Assuming that the missing values that were reported in µmol/L are in random, we can say that the distribution of the values that are needed to be converted and the distribution of the entire dataset should follow the same distribution. By National kidney Foundation, Cockroft-Gault Formula has conversion factor of 88.42. As we need to maintain the same 1st quartile, median and 3rd quartile to hold the same distribution, my choice of value is $0.4 * 88.42$. For those values in `scr` that are greater than our chosen standard are being convert. This can be checked using `summary()` function as shown above.

Problem 2.b (11 points)

Compute the eGFR according to the two equations. Report (rounded to the second decimal place) mean and standard deviation of the two eGFR vectors and their Pearson correlation coefficient. Also report the same quantities according to strata of MDRD4 eGFR: 0-60, 60-90 and > 90.

Answer

```
#computing MDRD4
#removing missing values
scr.mdrd <- scr %>% copy() %>% na.omit() %>%
  #equating into the equation
  .[, mdrd4:= 175 * scr^(-1.154) * age^(-0.203)] %>%
  #special case for sex = Female
  .[, mdrd4:= ifelse(sex == "Female", mdrd4 * 0.742, mdrd4)] %>%
  #special case for ethnic = Black
  .[, mdrd4:= ifelse(ethnic == "Black", mdrd4 * 1.212, mdrd4)]

head(scr.mdrd, 15)
```

```
##     age  scr    sex ethnic      mdrd4
##  1:  48  1.2 Female  Other  47.948478
##  2:   7  0.8   Male  Black 184.850199
##  3:  48  3.8 Female  Other  12.678846
##  4:  51  1.4   Male  Other  53.428078
##  5:  60  1.1   Male  Other  68.281993
##  6:  68 24.0   Male  Other   1.897907
##  7:  24  1.1   Male  Black  99.676006
##  8:  52  1.9 Female  Other  27.759499
##  9:  53  7.2   Male  Other   8.010292
## 10:  50  4.0 Female  Other  11.851513
## 11:  63  2.7   Male  Other  23.987121
## 12:  68  2.1 Female  Other  23.420792
## 13:  68  4.6   Male  Other  12.770744
## 14:  68  4.1   Male  Black  17.676195
## 15:  40  9.6   Male  Other   6.085256
```

```
#computing CKD_EPI
#removing missing values
scr.ckd <- scr %>% copy() %>% na.omit() %>%
  #computing the kappa for min and max
  .[, kappa := ifelse(sex=="Female", scr/0.7, scr/0.9)] %>%
  .[, minkappa := ifelse(kappa < 1, kappa, 1)] %>%
  .[, maxkappa := ifelse(kappa > 1, kappa, 1)] %>%
  #equating to the equation based on sex
  .[, ckd.epi := ifelse(sex=="Female",
    141 * (minkappa^(-0.329)) * (maxkappa^(-1.209)) *
      0.993^(age) * 1.018,
    141 * (minkappa^(-0.411)) * (maxkappa^(-1.209)) *
      0.993^(age))] %>%
  #special case for ethnic = Black
  .[, ckd.epi:= ifelse(ethnic == "Black", ckd.epi*1.159, ckd.epi)]

head(scr.ckd, 15)
```

```
##     age  scr    sex ethnic      kappa  minkappa  maxkappa    ckd.epi
##  1:  48  1.2 Female  Other  1.7142857 1.0000000  1.714286  53.397905
##  2:   7  0.8   Male  Black  0.8888889 0.8888889  1.000000 163.294281
##  3:  48  3.8 Female  Other  5.4285714 1.0000000  5.428571  13.252444
##  4:  51  1.4   Male  Other  1.5555556 1.0000000  1.555556  57.761862
##  5:  60  1.1   Male  Other  1.2222222 1.0000000  1.222222  72.578746
##  6:  68 24.0   Male  Other 26.6666667 1.0000000 26.666667   1.651094
##  7:  24  1.1   Male  Black  1.2222222 1.0000000  1.222222 108.322818
##  8:  52  1.9 Female  Other  2.7142857 1.0000000  2.714286  29.787787
##  9:  53  7.2   Male  Other  8.0000000 1.0000000  8.000000   7.864905
## 10:  50  4.0 Female  Other  5.7142857 1.0000000  5.714286  12.281808
## 11:  63  2.7   Male  Other  3.0000000 1.0000000  3.000000  23.998394
## 12:  68  2.1 Female  Other  3.0000000 1.0000000  3.000000  23.587189
## 13:  68  4.6   Male  Other  5.1111111 1.0000000  5.111111  12.166705
## 14:  68  4.1   Male  Black  4.5555556 1.0000000  4.555556  16.205967
## 15:  40  9.6   Male  Other 10.6666667 1.0000000 10.666667   6.085585
```

```r
#computing mean and standard deviation of MDRD4
cat("The mean of MDRD4 :",
    round(mean(scr.mdrd$mdrd4, na.rm = TRUE), 2),
    "| The standard deviation of MDRD4 :",
    round(sd(scr.mdrd$mdrd4, na.rm = TRUE),2),
    "\n")
```

```
## The mean of MDRD4 : 59.91 | The standard deviation of MDRD4 : 47.7
```

```r
#computing mean and standard deviation of CKD-EPI
cat("The mean of CKD-EPI :",
    round(mean(scr.ckd$ckd.epi, na.rm = TRUE),2),
    "| The standard deviation of CKD-EPI :",
    round(sd(scr.ckd$ckd.epi, na.rm = TRUE),2),
    "\n")
```

```
## The mean of CKD-EPI : 58.98 | The standard deviation of CKD-EPI : 41.99
```

```r
#computing the correlation between MDRD4 and CKD-EPI
cat("The Pearson correlation coefficient is " ,
    round(cor(scr.mdrd$mdrd4, scr.ckd$ckd.epi),2))
```

```
## The Pearson correlation coefficient is  0.97
```

By comparing the two mean values, we can conclude that the mean values of MDRD4 and CKD-EPI are similar. The similarity is also observed in the standard deviation between the two equations. The Pearson correlation coefficient also suggest positive relationship between MDRD4 and CKD-EPI values with the value of 0.97.

```r
#computing the quantities according to strata of MDRD4
table(cut(scr.mdrd$mdrd4, c(0, 60, 90, max(scr.mdrd$mdrd4, na.rm = TRUE))))
```

```
##
##   (0,60]  (60,90] (90,214]
##      209       88       80
```

```r
#defining the index for each strata
idx1 <- which(cut(scr.mdrd$mdrd4,
                  c(0, 60, 90, max(scr.mdrd$mdrd4, na.rm = TRUE)))=="(0,60]")
idx2 <- which(cut(scr.mdrd$mdrd4,
                  c(0, 60, 90, max(scr.mdrd$mdrd4, na.rm = TRUE)))=="(60,90]")
```

```
idx3 <- which(cut(scr.mdrd$mdrd4,
                  c(0, 60, 90, max(scr.mdrd$mdrd4, na.rm = TRUE)))==
                paste0("(90,", floor(max(scr.mdrd$mdrd4, na.rm = TRUE)), "]"))
```

```
#computing the mean and standard deviation of each strata of MDRD4
cat("(0,60] strata of MDRD4\n", "mean:",
    round(mean(scr.mdrd$mdrd4[idx1], na.rm = TRUE), 2),
    "| standard deviation:", round(sd(scr.mdrd$mdrd4[idx1], na.rm = TRUE),2),
    "\n")
```

```
## (0,60] strata of MDRD4
##  mean: 25.9 | standard deviation: 17.3
```

```
cat("(60,90] strata of MDRD4\n", "mean:",
    round(mean(scr.mdrd$mdrd4[idx2], na.rm = TRUE), 2),
    "| standard deviation:", round(sd(scr.mdrd$mdrd4[idx2], na.rm = TRUE),2),
    "\n")
```

```
## (60,90] strata of MDRD4
##  mean: 73.41 | standard deviation: 8.4
```

```
cat(paste0("(90,", floor(max(scr.mdrd$mdrd4, na.rm = TRUE)),
           "] strata of MDRD4\n"),
    "mean:", round(mean(scr.mdrd$mdrd4[idx3], na.rm = TRUE), 2),
    "| standard deviation:", round(sd(scr.mdrd$mdrd4[idx3], na.rm = TRUE),2),
    "\n")
```

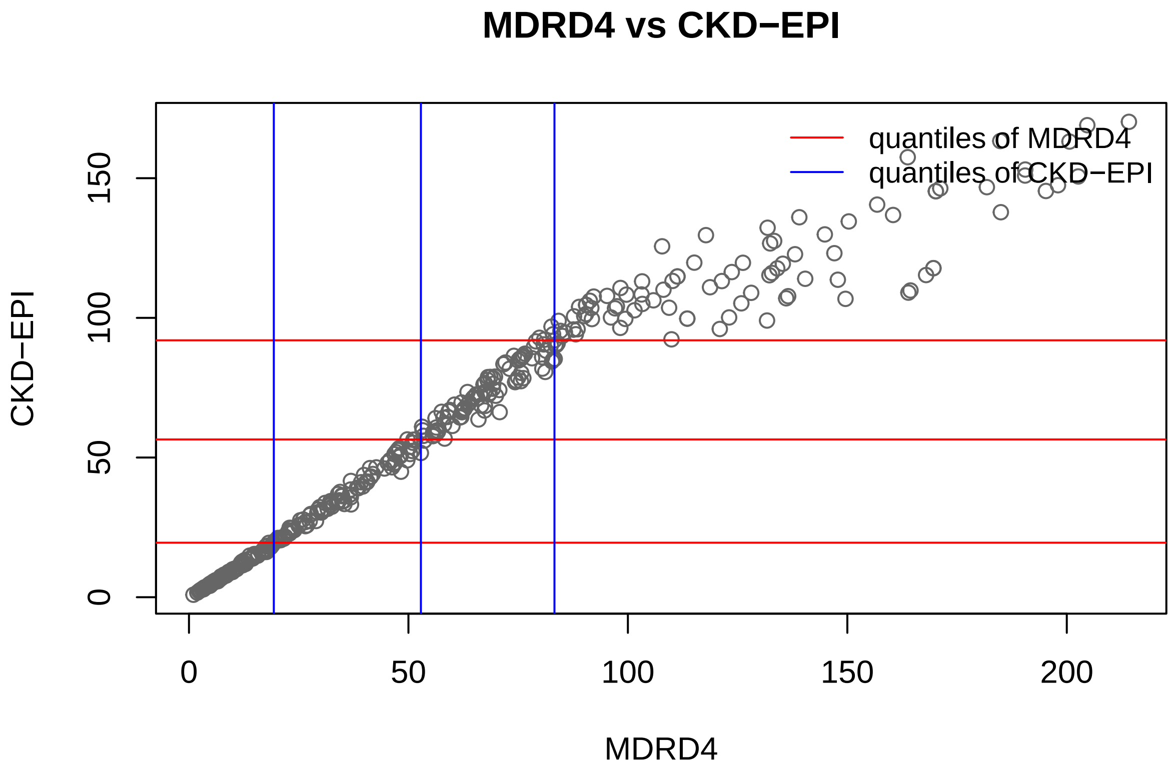
```
## (90,214] strata of MDRD4
##  mean: 133.91 | standard deviation: 34.04
```

Problem 2.c (7 points)

Produce a scatter plot of the two eGFR vectors, and add vertical and horizontal lines (i.e.) corresponding to median, first and third quantiles. Is the relationship between the two eGFR equations linear? Justify your answer.

Answer

```
#computing the quantiles for MDRD4
firstmdrd <- quantile(scr.mdrd$mdrd4)[2]
secondmdrd <- quantile(scr.mdrd$mdrd4)[3]
thirdmdrd <- quantile(scr.mdrd$mdrd4)[4]

#computing the quantiles for CKD-EPI
firstckd <- quantile(scr.ckd$ckd.epi)[2]
secondckd <- quantile(scr.ckd$ckd.epi)[3]
thirdckd <- quantile(scr.ckd$ckd.epi)[4]

#scatter plot of MDRD vs CKD-EPI
plot(scr.mdrd$mdrd4, scr.ckd$ckd.epi,
     main = "MDRD4 vs CKD-EPI", xlab="MDRD4", ylab = "CKD-EPI", col = "grey40")

#adding the quantiles of MDRD4
abline(h = firstckd, col = "red")
abline(h = secondckd, col = "red")
abline(h = thirdckd, col = "red")

#adding the quantiles of CKD-EPI
abline(v = firstmdrd, col = "blue")
abline(v = secondmdrd, col = "blue")
abline(v = thirdmdrd, col = "blue")

legend("topright", legend = c("quantiles of MDRD4", "quantiles of CKD-EPI"),
         col = c("red", "blue"), lty = 1, cex = 0.9, bty = "n")
```



By looking at the scatter plot above, we can conclude that there is a positive linear relationship between the eGFR values after the computation of it in previous part. However, we see that the positive relation is not followed after the values of 100. This is due to the chosen value when we are converting the values from µmol/L to mg/dL in *Q2.b* as some of the values are not converted. Although such values are observed, we believe that an outlier is possible to be observed in such a medical dataset. As a result, we conclude that there is a positive relationship between the two computed eGFR values.

Problem 3 (31 points)

You have been provided with electronic health record data from a study cohort. Three CSV (Comma Separated Variable) files are provided on learn.

The first file is a cohort description file cohort.csv file with fields: * id = study identifier * yob = year of birth * age = age at measurement * bp = systolic blood pressure * albumin = last known albuminuric status (categorical) * diabetes = diabetes status

The second file lab1.csv is provided by a laboratory after measuring various biochemistry levels in the cohort blood samples. Notice that a separate lab identifier is used to anonymise results from the cohort. The year of birth is also provided as a check that the year of birth aligns between the two merged sets. * LABID = lab identifier * yob = year of birth * urea = blood urea * creatinine = serum creatinine * glucose = random blood glucose

To link the two data files together, a third linker file linker.csv is provided. The linker file includes a LABID identifier and the corresponding cohort id for each person in the cohort.

Problem 3.a (6 points)

Using all three files provided on learn, load and merge to create a single data table based dataset cohort.dt. This will be used in your analysis. Perform assertion checks to ensure that all identifiers in cohort.csv have been accounted for in the final table and that any validation fields are consistent between sets. After the checks are complete, drop the identifier that originated from lab dataset LABID. Ensure that a single yob field remains and rename it. Ensure that the albumin field is converted to a factor and the ordering of the factor is 1="normo",2="micro",3="macro".

Answer

```r
cohort <- read.csv("data_assignment1/3_cohort.csv")
lab1 <- read.csv("data_assignment1/3_lab1.csv")
linker <- read.csv("data_assignment1/3_linker.csv")

#merging lab1 with linker based on LABID
cohort.dt <- merge(linker, lab1, by = "LABID", all = TRUE)
#merging cohort and lab1 using linker based on id
cohort.dt <- setDT(merge(cohort, cohort.dt, by = c("id"),
                         sort = FALSE, all = TRUE))

#converting albumin field to a factor with ordering
# cohort.dt[, diabetes := factor(diabetes)]
cohort.dt[, albumin := factor(albumin, levels = c("normo", "micro", "macro"))]
levels(cohort.dt$albumin) <- c(1, 2, 3)

head(cohort.dt, 10)
```

```
##         id yob.x age  bp diabetes albumin   LABID yob.y urea creatinine glucose
##  1:  PID_1  1971  48  80        1       2 LID_307  1971   36    106.104     121
##  2:  PID_2  2012   7  50        0       3 LID_266  2012   18     70.736      NA
##  3:  PID_3  1957  62  80        1       2 LID_237  1957   53    159.156     423
##  4:  PID_4  1971  48  70        0       3 LID_154  1971   56    335.996     117
##  5:  PID_5  1968  51  80        0       2 LID_223  1968   26    123.788     106
##  6:  PID_6  1959  60  90        1       2  LID_22  1959   25     97.262      74
##  7:  PID_7  1951  68  70        0       1 LID_250  1951   54   2122.080     100
##  8:  PID_8  1995  24  NA        1       2 LID_236  1995   31     97.262     410
##  9:  PID_9  1967  52 100        1       2 LID_252  1967   60    167.998     138
## 10: PID_10  1966  53  90        1       2 LID_197  1966  107    636.624      70
```

```
                  #assertive check of the id field
assertcheck <- c(identical(cohort.dt$id, linker$id),
                  identical(cohort.dt$id, cohort$id),
                  #assertive check of the year of birth field
                  identical(cohort.dt$yob.x, cohort$yob),
                  identical(cohort.dt$yob.y, cohort$yob),
                  identical(cohort.dt$yob.x, lab1$yob),
                  identical(cohort.dt$yob.y, lab1$yob),
                  #assertive check of the LABID field
                  identical(cohort.dt$LABID, lab1$LABID),
                  identical(cohort.dt$LABID, linker$LABID))

cat("Out of 8 assertive checks, we have", sum(assertcheck), "passed")
```

```
## Out of 8 assertive checks, we have 8 passed
```

We notice that all the assertive checks passed with true values. Therefore we can conclude that the merging between the three dataset is completed.

```
cohort.dt <- cohort.dt %>%
  #ensuring only one year of birth field
  .[, !"yob.y"] %>%
  #removing LABID field
  .[, !"LABID"] %>%
  rename(yob = yob.x)

head(cohort.dt, 10)
```

```
##          id  yob age  bp diabetes albumin urea creatinine glucose
##  1:  PID_1 1971  48  80        1       2   36    106.104     121
##  2:  PID_2 2012   7  50        0       3   18     70.736      NA
##  3:  PID_3 1957  62  80        1       2   53    159.156     423
##  4:  PID_4 1971  48  70        0       3   56    335.996     117
##  5:  PID_5 1968  51  80        0       2   26    123.788     106
##  6:  PID_6 1959  60  90        1       2   25     97.262      74
##  7:  PID_7 1951  68  70        0       1   54   2122.080     100
##  8:  PID_8 1995  24  NA        1       2   31     97.262     410
##  9:  PID_9 1967  52 100        1       2   60    167.998     138
## 10: PID_10 1966  53  90        1       2  107    636.624      70
```

Problem 3.b (10 points)

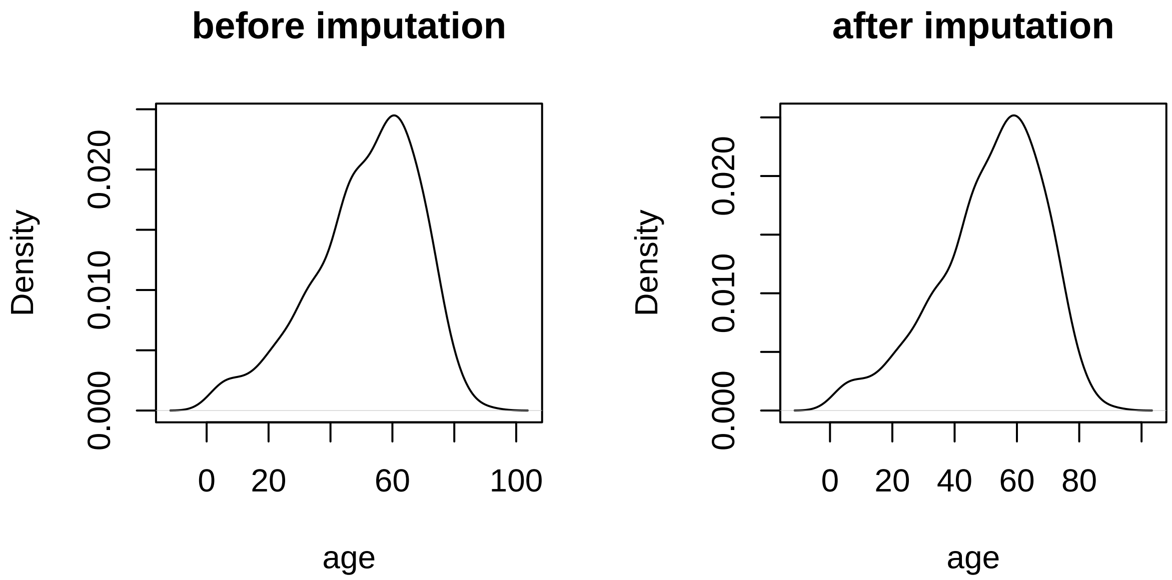
Create a copy of the dataset where you will impute all missing values. Update any missing age fields using the year of birth, for all other continuous variables write a function called impute.to.mean and impute to mean, impute any categorical variable to the mode. Compare the distributions of the imputed and non-imputed variables and decide which ones to keep for further analysis. Justify your answer.

Answer

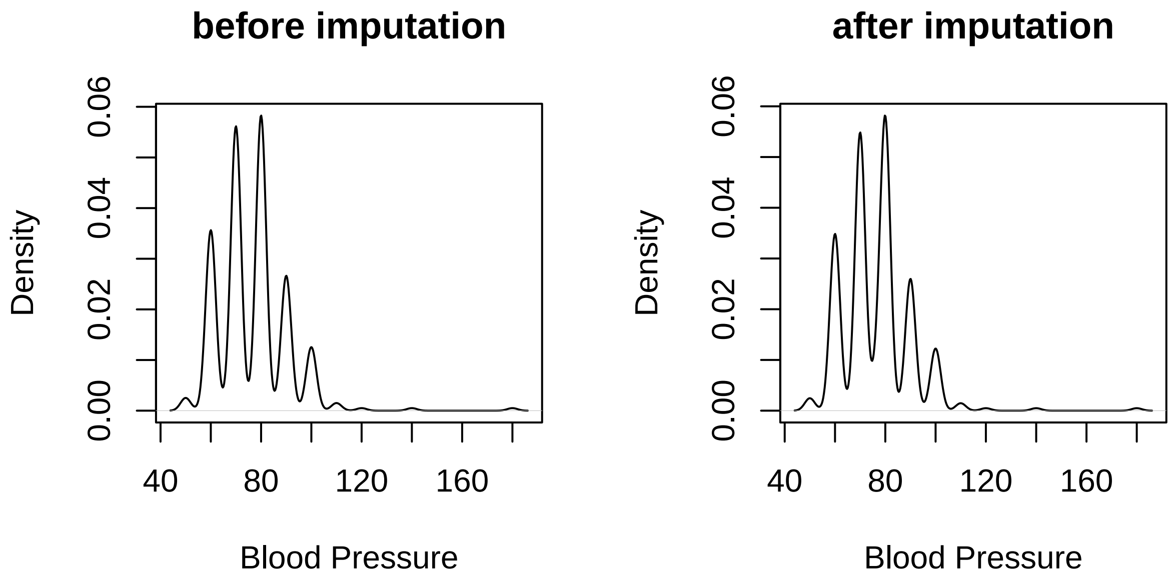
```
#imputation of the missing value for age
cohort.impute <- cohort.dt %>% copy() %>%
  .[, yob := floor(yob)] %>%
  .[, age := ifelse(is.na(age), 2022 - yob, age)]

#defining function for mean imputation
impute.to.mean <- function(x) {
  #check if numeric/integer columns
  if (is.numeric(x) || is.integer(x)){
    #find which values are missing
    na.idx <- is.na(x)
    #replace NAs with the mean values
    x[na.idx] <- mean(x, na.rm = TRUE)
    }#return the vector with imputed values
  return(x)
}
#performing mean imputation
numcols <- c("bp", "urea", "creatinine", "glucose")
# numcols <- cohort.impute %>% select_if(is.numeric) %>% colnames
cohort.impute <- cohort.impute %>%
  .[, (numcols) := lapply(.SD, impute.to.mean), .SDcols = numcols]

#defining a function to compute the mode
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}
#computing the mode of albumin
mode <- names(which.max(table(cohort.dt[, albumin])))

#imputation for categorical variables
cohort.impute <- cohort.impute %>%
  #imputation for diabetes
  .[, diabetes := ifelse(is.na(diabetes),
                       getmode(cohort.dt$diabetes),
                       diabetes)] %>%
  #imputation for albumin
  .[is.na(albumin), albumin := names(which.max(table(cohort.dt[, albumin])))]

head(cohort.impute, 55)
```

```
##         id  yob age        bp diabetes albumin      urea creatinine  glucose
##  1:  PID_1 1971  48  80.00000        1       2  36.00000   106.1040 121.0000
##  2:  PID_2 2012   7  50.00000        0       3  18.00000    70.7360 148.0365
##  3:  PID_3 1957  62  80.00000        1       2  53.00000   159.1560 423.0000
##  4:  PID_4 1971  48  70.00000        0       3  56.00000   335.9960 117.0000
##  5:  PID_5 1968  51  80.00000        0       2  26.00000   123.7880 106.0000
##  6:  PID_6 1959  60  90.00000        1       2  25.00000    97.2620  74.0000
##  7:  PID_7 1951  68  70.00000        0       1  54.00000  2122.0800 100.0000
##  8:  PID_8 1995  24  76.46907        1       2  31.00000    97.2620 410.0000
##  9:  PID_9 1967  52 100.00000        1       2  60.00000   167.9980 138.0000
## 10: PID_10 1966  53  90.00000        1       2 107.00000   636.6240  70.0000
## 11: PID_11 1969  50  60.00000        1       2  55.00000   353.6800 490.0000
## 12: PID_12 1956  63  70.00000        1       2  60.00000   238.7340 380.0000
## 13: PID_13 1951  68  70.00000        1       2  72.00000   185.6820 208.0000
## 14: PID_14 1951  68  70.00000        1       1  86.00000   406.7320  98.0000
## 15: PID_15 1951  68  80.00000        1       2  90.00000   362.5220 157.0000
## 16: PID_16 1979  40  80.00000        0       2 162.00000   848.8320  76.0000
## 17: PID_17 1972  47  70.00000        0       2  46.00000   194.5240  99.0000
## 18: PID_18 1972  47  80.00000        0       1  87.00000   459.7840 114.0000
## 19: PID_19 1959  60 100.00000        1       1  27.00000   114.9460 263.0000
## 20: PID_20 1957  62  60.00000        0       2  31.00000   141.4720 100.0000
## 21: PID_21 1958  61  80.00000        1       2 148.00000   344.8380 173.0000
## 22: PID_22 1959  60  90.00000        1       1 180.00000  6719.9200 148.0365
## 23: PID_23 1971  48  80.00000        0       3 163.00000   680.8340  95.0000
## 24: PID_24 1998  21  70.00000        0       1  57.42572   271.6664 148.0365
## 25: PID_25 1977  42 100.00000        0       3  50.00000   123.7880 148.0365
## 26: PID_26 1958  61  60.00000        1       1  75.00000   167.9980 108.0000
## 27: PID_27 1944  75  80.00000        1       1  45.00000   212.2080 156.0000
## 28: PID_28 1950  69  70.00000        1       2  87.00000   238.7340 264.0000
## 29: PID_29 1944  75  70.00000        1       2  31.00000   123.7880 123.0000
## 30: PID_30 1951  68  70.00000        0       2  28.00000   123.7880 148.0365
## 31: PID_31 1967  55  70.00000        1       1 155.00000   645.4660  93.0000
## 32: PID_32 1946  73  90.00000        0       2  33.00000   132.6300 107.0000
## 33: PID_33 1958  61  90.00000        1       2  39.00000   132.6300 159.0000
## 34: PID_34 1959  60 100.00000        0       2  55.00000   221.0500 140.0000
## 35: PID_35 1949  70  70.00000        1       2 153.00000   459.7840 171.0000
## 36: PID_36 1954  65  90.00000        1       2  39.00000   176.8400 270.0000
## 37: PID_37 1943  76  70.00000        0       2  29.00000   159.1560  92.0000
## 38: PID_38 1947  72  80.00000        1       1  65.00000   300.6280 137.0000
## 39: PID_39 1950  69  80.00000        0       2 103.00000   362.5220 148.0365
## 40: PID_40 1937  82  80.00000        1       2  70.00000   300.6280 140.0000
## 41: PID_41 1973  46  90.00000        0       2  80.00000   185.6820  99.0000
## 42: PID_42 1974  45  70.00000        0       1  20.00000    61.8940 148.0365
## 43: PID_43 1972  47 100.00000        0       1  29.00000    88.4200 204.0000
## 44: PID_44 1984  35  80.00000        1       2 202.00000   954.9360  79.0000
## 45: PID_45 1965  54  80.00000        1       2  77.00000   557.0460 207.0000
## 46: PID_46 1965  54  80.00000        1       2  89.00000   521.6780 208.0000
## 47: PID_47 1971  48  70.00000        1       1  24.00000   106.1040 124.0000
## 48: PID_48 2008  11  80.00000        0       2  17.00000    70.7360 148.0365
## 49: PID_49 1946  73  70.00000        1       1  32.00000    79.5780  70.0000
## 50: PID_50 1959  60  70.00000        1       2  72.00000   265.2600 144.0000
## 51: PID_51 1966  53  60.00000        1       1 114.00000   287.3650  91.0000
## 52: PID_52 1965  54 100.00000        1       2  66.00000   141.4720 162.0000
## 53: PID_53 1966  53  90.00000        0       1  38.00000   194.5240 148.0365
## 54: PID_54 1957  62  80.00000        1       1  24.00000    88.4200 246.0000
## 55: PID_55 1956  63  80.00000        0       2  57.42572   300.6280 148.0365
##         id  yob age        bp diabetes albumin      urea creatinine  glucose
```

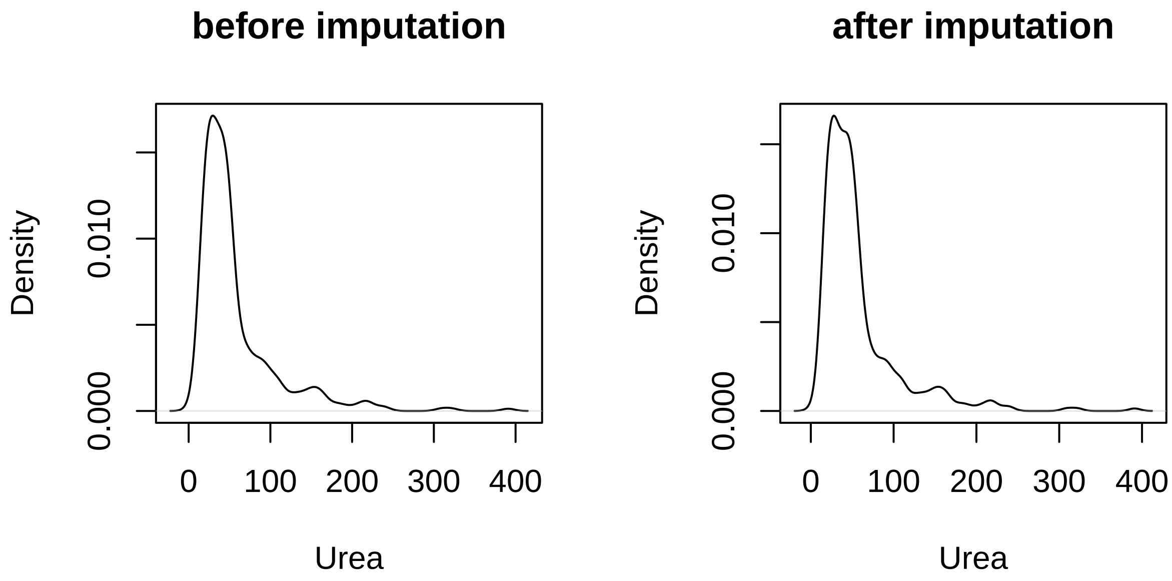
```
par(mfrow=c(1, 2))
plot(density(cohort.dt$age, na.rm = TRUE), main = "before imputation",
     xlab = "age")
plot(density(cohort.impute$age), main = "after imputation", xlab = "age")
```



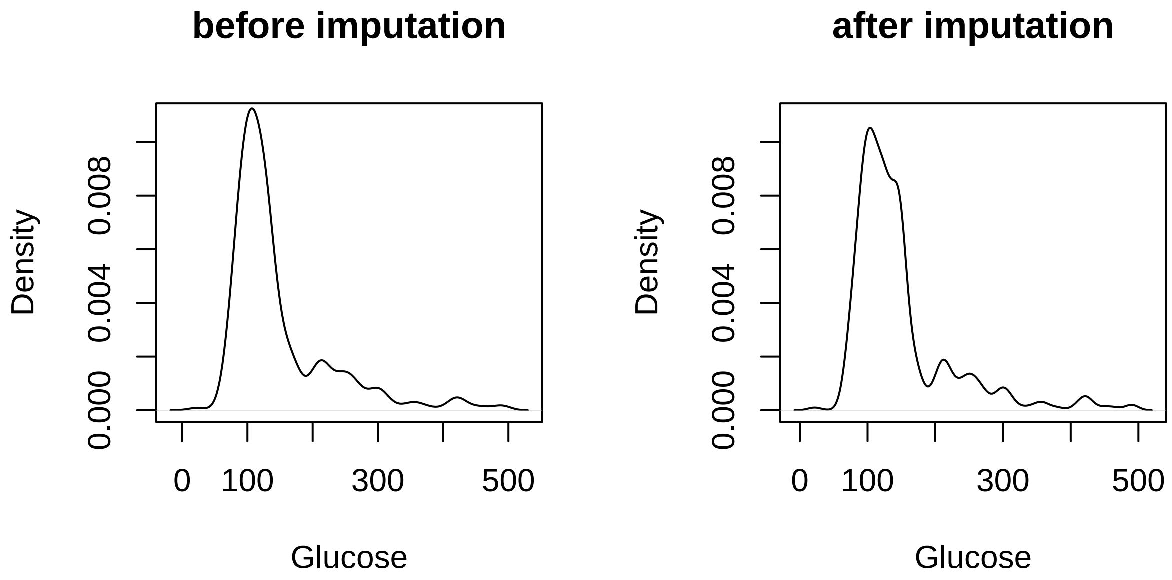
```
par(mfrow=c(1, 2))
plot(density(cohort.dt$bp, na.rm = TRUE), main = "before imputation",
     xlab = "Blood Pressure")
plot(density(cohort.impute$bp), main = "after imputation",
     xlab = "Blood Pressure")
```



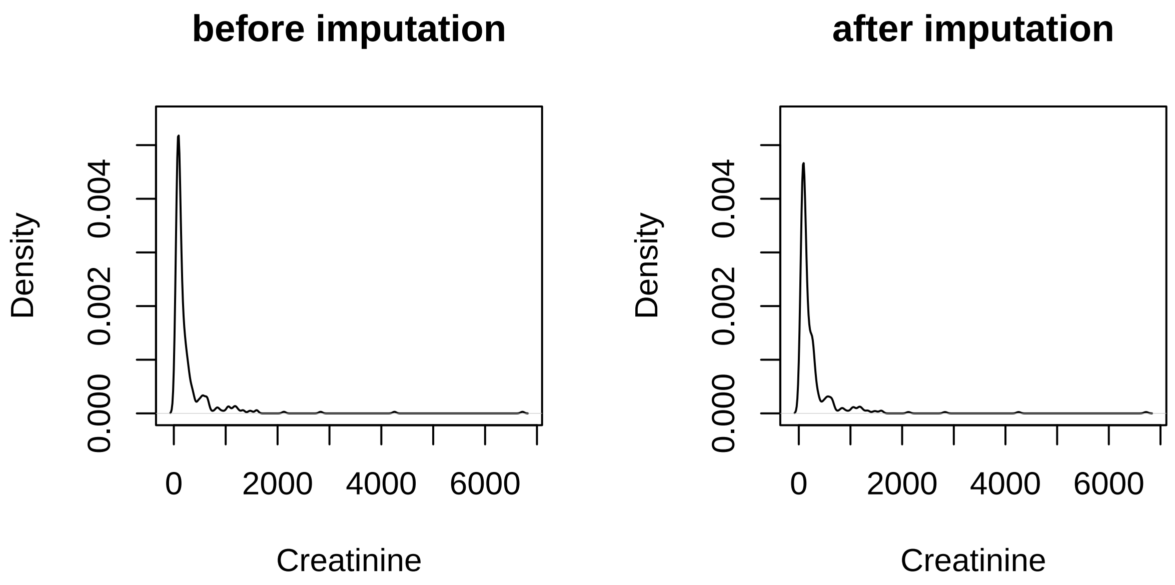
```r
par(mfrow=c(1, 2))
plot(density(cohort.dt$urea, na.rm = TRUE), main = "before imputation", xlab = "Urea")
plot(density(cohort.impute$urea), main = "after imputation", xlab = "Urea")
```



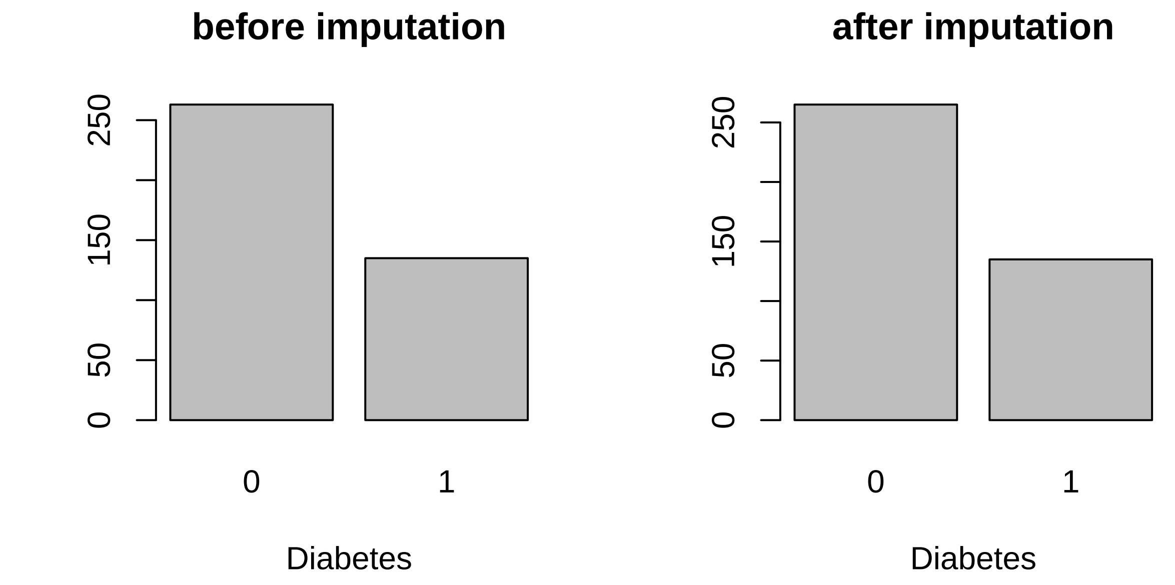
```r
par(mfrow=c(1, 2))
plot(density(cohort.dt$glucose, na.rm = TRUE), main = "before imputation",
     xlab = "Glucose", ylim=c(0,0.011))
plot(density(cohort.impute$glucose), main = "after imputation",
     xlab = "Glucose", ylim=c(0,0.011))
```



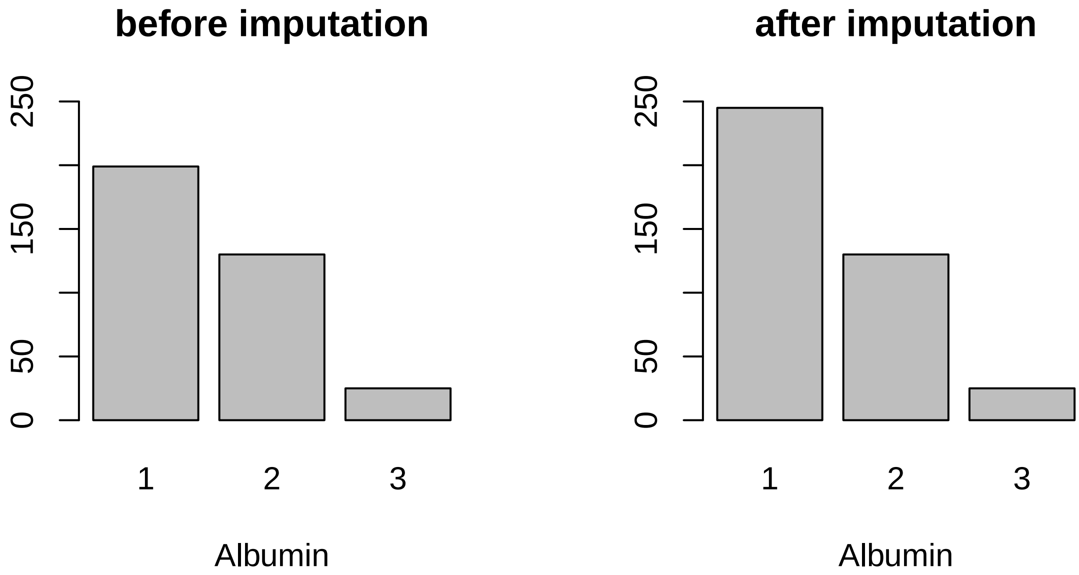
```
par(mfrow=c(1, 2))
plot(density(cohort.dt$creatinine, na.rm = TRUE), main = "before imputation",
     xlab = "Creatinine", ylim = c(0, 0.0055))
plot(density(cohort.impute$creatinine), main = "after imputation",
     xlab = "Creatinine", ylim = c(0, 0.0055))
```



```
par(mfrow=c(1, 2))
plot(factor(na.omit(cohort.dt$diabetes)), main = "before imputation", xlab = "Diabetes")
plot(factor(cohort.impute$diabetes), main = "after imputation", xlab = "Diabetes")
```



```
par(mfrow=c(1, 2))
plot(na.omit(cohort.dt$albumin), main = "before imputation",
     xlab = "Albumin", ylim = c(0, 250))
plot(cohort.impute$albumin, main = "after imputation",
     xlab = "Albumin", ylim = c(0, 250))
```



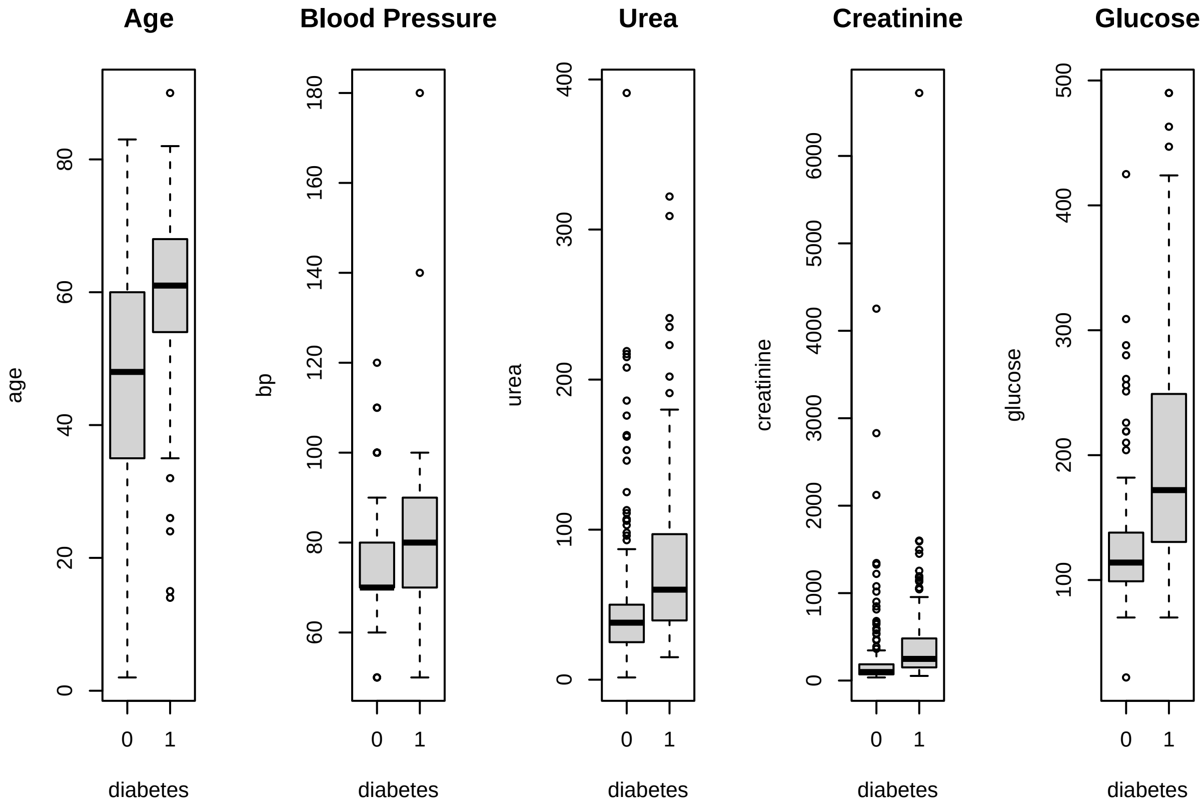
Let us look at the plots above, the plots are generally having the identical shape among each others. However, we can observe some slight differences between the plots. There are several reasons for different variables. For the continuous variables, as we perform the mean imputation, we can maintain the same mean within the variable but the standard error decreases. Therefore, there could be a change in the shape of the density plots. For the categorical variables, especially for the albumin, the mode of albumin is *normo* which consist of approximately 200 values. However, if we perform the imputation by adding the mode value into the missing value, this will increase the number of *normo* factor as the number of missing value is high compared to the entries of diabetes. Although there are such noticeable differences, we can still conclude that the variables after the imputation follows the similar distribution to the original dataset.

Problem 3.c (6 points)

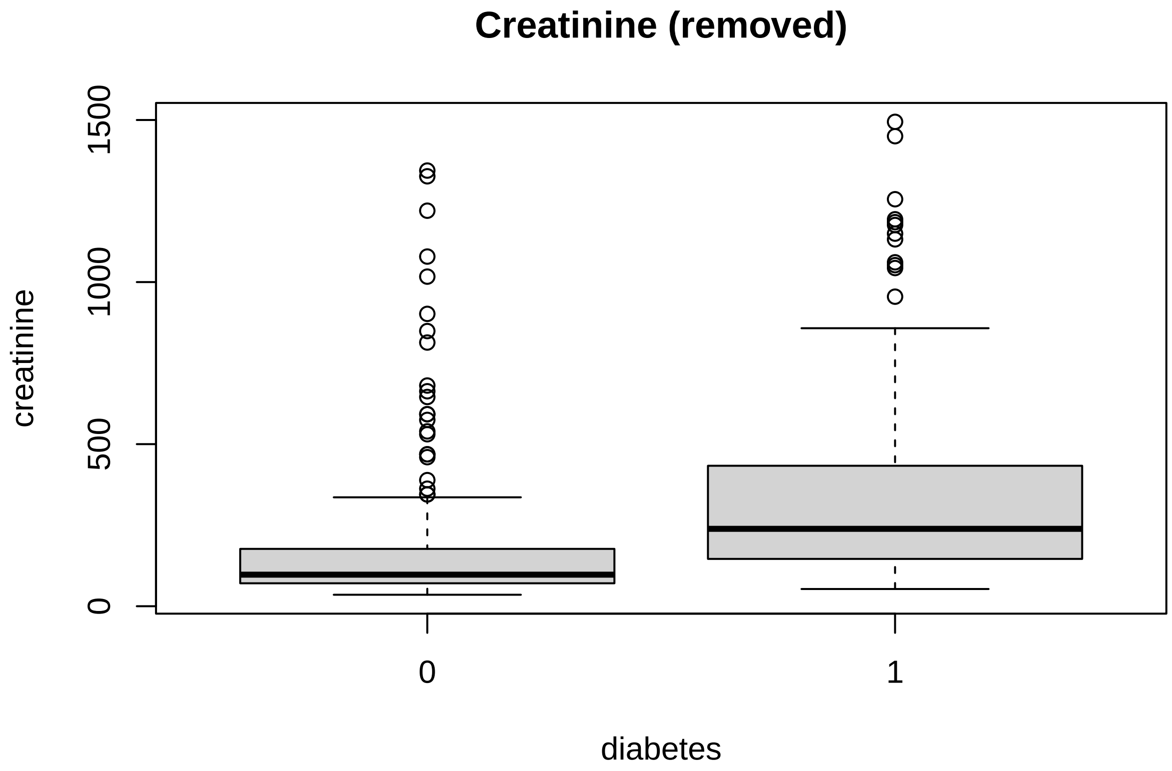
Plot boxplots of potential predictors for diabetes grouped by cases and controls and use these to decide which predictors to keep for future analysis. For any categorical variables create a table instead. Justify your answers.

Answer

```
#computing the boxplot of diabetes against each continuous variables
par(mfrow=c(1, 5))
boxplot(age ~ diabetes, data = cohort.impute, main = "Age")
boxplot(bp ~ diabetes, data = cohort.impute, main = "Blood Pressure")
boxplot(urea ~ diabetes, data = cohort.impute, main = "Urea")
boxplot(creatinine ~ diabetes, data = cohort.impute, main = "Creatinine")
boxplot(glucose ~ diabetes, data = cohort.impute, main = "Glucose")
```



```
#removing the outlier in creatinine
par(mfrow=c(1, 1))
boxplot(creatinine ~ diabetes,
        data = cohort.impute[cohort.impute$creatinine<1500],
        main = "Creatinine (removed)")
```



```
#computing the table of diabetes against albumin
data.frame(control = table(cohort.impute$albumin[cohort.impute$diabetes==0]),
           case = table(cohort.impute$albumin[cohort.impute$diabetes==1]))
```

```
##   control.Var1 control.Freq case.Var1 case.Freq
## 1            1          192         1        53
## 2            2           61         2        69
## 3            3           12         3        13
```

By looking at the box plot, we need to choose variables that show clear dividend between the classes of diabetes. By that, we can see that `age`, `urea`, `creatinine` and `glucose` is matching the criteria. To scrutinise further, we removed the obvious outlier in `creatinine`, and plotted the boxplot. Generally, a ratio of albumin(mcg/L) to creatinine (mg/L) of less than 30 is considered normal, *normo* by Mayo Clinic. The ratio of 30-300 indicates microalbuminuria, *micro* and higher than 300 indicates macroalbuminuria, *macro*. Looking at the table, we can see that there is clear difference in the *normo* factor. However, the others factors are having similar numbers in both classes of *micro* and *macro* regardless of the case of `diabetes`. Since we know `urea` and `glucose` have direct impact towards diabetes we choose `urea` and `glucose` as the variables.

Problem 3.d (9 points)

Use your findings from the previous exercise fit an appropriate model of diabetes with two predictors. Print a summary and explain the results as you would communicate it to a colleague with a medical rather than statistical background.

Answer

```
#fitting logistic regression with
#response variable : diabetes
#explanatory variables : glucose and urea
fit1 <- glm(diabetes ~ glucose + urea, family = "binomial", data = cohort.impute)
summary(fit1)
```

```
##
## Call:
## glm(formula = diabetes ~ glucose + urea, family = "binomial",
##     data = cohort.impute)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0400  -0.6765  -0.5045   0.5714   2.2936
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -4.282899   0.421948 -10.150  < 2e-16 ***
## glucose      0.018582   0.002474   7.511 5.88e-14 ***
## urea         0.014097   0.002966   4.753 2.01e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 511.49  on 399  degrees of freedom
## Residual deviance: 374.32  on 397  degrees of freedom
## AIC: 380.32
##
## Number of Fisher Scoring iterations: 5
```

```
#testing the goodness of fit by deriving p-value
signif(pchisq(fit1$null.deviance-fit1$deviance, df = 2, lower.tail = FALSE), 3)
```

```
## [1] 1.64e-30
```

Looking at the summary of the fit, we now have the model

$$\log(\mathbb{P}(diabetes|glucose, urea)) = -4.28 + 0.0185 \times glucose + 0.0141 \times urea$$

We can also see that the p-value of each variable is less than 0.05. This indicates that the variables are significant. Looking at the p-value of the goodness of fit of the model, it has the value of $1.64e - 30 < 0.05$. Therefore, we conclude that model is a good fit to the data.

Diabetes is a disease that occurs when blood glucose is too high. In addition, one of the symptoms of kidney failure is diabetes as urea that builds up in the blood can cause diabetes. Also, kidney failure result in low albumin and low creatinine as well. Thus, it is easy to correlate the symptoms but those who has low albumin and creatinine may not also have diabetes as shown in the boxplot. Therefore, `glucose` and `urea` are sufficient factors to represent the dataset as evident above.

Problem 4 (19 points)

Problem 4.a. (9 points)

Add a third predictor to the final model from problem 3, perform a likelihood ratio test to compare both models and report the p-value for the test. Is there any support for the additional term? Plot a ROC curve for both models and report the AUC, explain the results as you would communicate it to a colleague with a medical rather than statistical background.

Answer

```
#fitting logistic regression with
#response variable : diabetes
#explanatory variables : glucose, urea and age
fit2 <- glm(diabetes ~ glucose + urea + age, family = "binomial", data = cohort.impute)
summary(fit2)
```

```
##
## Call:
## glm(formula = diabetes ~ glucose + urea + age, family = "binomial",
##     data = cohort.impute)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0518  -0.6480  -0.3891   0.6096   2.9163
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -6.609287   0.710864  -9.298  < 2e-16 ***
## glucose      0.017021   0.002468   6.895 5.38e-12 ***
## urea         0.012671   0.002955   4.289 1.80e-05 ***
## age          0.047823   0.009973   4.795 1.62e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 511.49  on 399  degrees of freedom
## Residual deviance: 346.46  on 396  degrees of freedom
## AIC: 354.46
##
## Number of Fisher Scoring iterations: 5
```

```
#testing the goodness of fit by deriving p-value
signif(pchisq(fit2$null.deviance - fit2$deviance, df=2, lower.tail=FALSE), 3)
```

```
## [1] 1.46e-36
```

```
#likelihood ratio test between two fits
pchisq(fit1$deviance - fit2$deviance, df = 1, lower.tail = FALSE)
```

```
## [1] 1.300336e-07
```

```
#computing the predicted values for both fits
diabetes.pred1 <- predict(fit1)
diabetes.pred2 <- predict(fit2)
```

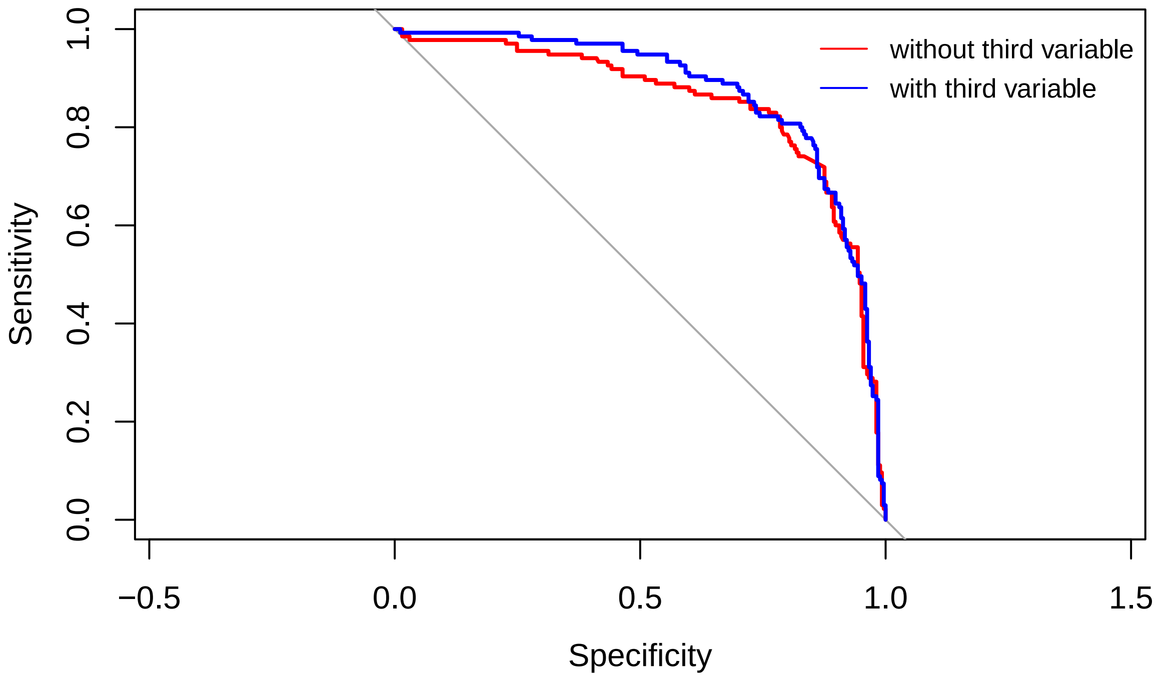
```r
#computing the ROC curve of both fits
roc(cohort.impute$diabetes, diabetes.pred1, plot = TRUE,
    xlim = c(0,1), col = "red")
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases

##
## Call:
## roc.default(response = cohort.impute$diabetes, predictor = diabetes.pred1,     plot = TRUE, xlim = c
##
## Data: diabetes.pred1 in 265 controls (cohort.impute$diabetes 0) < 135 cases (cohort.impute$diabetes 
## Area under the curve: 0.849
```

```r
roc(cohort.impute$diabetes, diabetes.pred2, plot = TRUE,
    add = TRUE, col = "blue")
```

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

##
## Call:
## roc.default(response = cohort.impute$diabetes, predictor = diabetes.pred2,     plot = TRUE, add = TRU
##
## Data: diabetes.pred2 in 265 controls (cohort.impute$diabetes 0) < 135 cases (cohort.impute$diabetes 
## Area under the curve: 0.8744
```

```r
legend("topright", legend = c("without third variable", "with third variable"),
         col = c("red", "blue"), lty = 1, cex = 0.8, bty = "n")
```



We included additional factor `age` into the model. For the diabetes patient, we see that those who are older will have more exposure towards having the diabetes. This is further evident in the boxplot, where the mean value of `age` who have diabetes is higher compared to the those who do not have diabetes in the boxplot.

Looking at the summary of the fit, we now have the model

$$\log(\mathbb{P}(diabetes|glucose, urea, age)) = -6.61 + 0.017 \times glucose + 0.0126 \times urea + 0.0478 \times age$$

The model here suggest that any increase in `glucose`, `urea` and `age` will lead to higher probability of having `diabetes`. This is also true in the field of biology and medical studies.

We can also see that the p-value of each variable is less than 0.05. This indicates that the variables are significant. Looking at the p-value of the goodness of fit of the model, it has the value of $1.46e - 36 < 0.05$. Therefore, we conclude that the model is a good fit to the data.

To select a better model, we performed several tasks to evaluate the goodness. First, likelihood ratio test was done. We compared the deviance of the two model and obtained a p-value of $1.3e - 7$. This suggests that, we reject the null hypothesis where the model with third predictor is a better model.

Now we look at the ROC curve with AUG values above, we can see that the model with third variable, age has a higher value of AUC value, 0.874 than 0.849. Overall, both statistically and biologically, the model with third feature is a preferred model than the model without third feature. In other words, we can conclude that with three variables, `glucose`, `urea` and `age` we can establish a model that best represents the diabetes classes. To further predict whether a patient has diabetes, this model is suitable to employ for future medical records.

Problem 4.b (10 points)

Perform 10-folds cross-validation for your chosen model and report the mean cross-validated AUCs.

Answer

```r
invisible({capture.output({
  #setting seed
  set.seed(1)

  #defining function to perform cross validation
  glm.cv <- function(formula, data, folds) {
    #initialising list of list to store regression of each fold
    regr.cv <- NULL
    for (f in 1:length(folds)) {
      #computing logistic regression on the training set
      regr.cv[[f]] <- glm(formula, data = data[-folds[[f]], ],
                          family = "binomial")
    }
    #returning the regression outputs
    return(regr.cv)
  }
  #initialising number of folds
  num.folds <- 10
  folds <- createFolds(cohort.impute$diabetes, k = num.folds)
  #storing the output of cross validation
  cv.m <- glm.cv(diabetes ~ glucose + urea + age, cohort.impute, folds)
  #initialsing list of list to store prediced values of each fold
  pred.cv <- NULL
  #initalising list to store auc valude of each fold
  auc.cv <- numeric(num.folds)
  for(f in 1:num.folds) {
    test.idx <- folds[[f]]
    #computing the predicted values
    pred.cv[[f]] <- data.frame(obs = cohort.impute$diabetes[test.idx],
                               pred = predict(cv.m[[f]],
                                              newdata = cohort.impute,
                                              type = "response")[test.idx])
    #computing the auc value of fold
    auc.cv[f] <- roc(obs ~ pred, data = pred.cv[[f]])$auc
  }
})})
```

```r
#computing the mean of AUC of the 10-folds cross validation
round(mean(auc.cv),3)
```

```
## [1] 0.869
```

After performing 10-fold cross validation, we obtained a mean AUC value of 0.869 and this seems reasonable compared to the AUC value we calculated in Q4 part a.