Assignment 1

Biomedical Data Science (MATH11174), 22/23, Semester 2

Josephine Li

March 9, 2023

Due on Thursday, 9th of March 2023, 5:00pm

! Pay Attention

The assignment is marked out of 100 points, and will contribute to ***20%*** of your final mark. The aim of this assignment is to produce a precise report in biomedical studies with the help of statistics and machine learning. Please complete this assignment using **Quarto/Rmarkdown file and render/knit this document only in PDF format** and submit using the **gradescope link on Learn**. You can simply click render on the top left of Rstudio (`Ctrl+Shift+K`). If you cannot render/knit to PDF directly, open **Terminal** in your RStudio (`Alt+Shift+R`) and type `quarto tools install tinytex`, otherwise please follow this link. If you have any code that does not run you will not be able to render nor knit the document so comment it as you might still get some grades for partial code.
Clear and reusable code will be rewarded. Codes without proper indentation, choice of variable identifiers, **comments**, error checking, etc will be penalised. An initial code chunk is provided after each subquestion but **create as many chunks as you feel is necessary** to make a clear report. Add plain text explanations in between the chunks when required to make it easier to follow your code and reasoning. Ensure that all answers containing multiple values should be presented and formatted with `kable()` and `kable_styling()` or using Markdown syntax. All plots must be displayed with clear title, label and legend.

Problem 1 (25 points)

Files `longegfr1.csv` and `longegfr2.csv` (available on Assessment > Assignment 1) 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 table format and merge in an appropriate way into a single data table.
- Order the observations according to subject identifier and follow-up time.
- Print first 10 values of the new dataset using `head()`.

```r
#Answer in this chunk
# Convert 2 files to data table format
longegfr1 <- fread("data_assignment1/longegfr1.csv", stringsAsFactors = F)
longegfr2 <- fread("data_assignment1/longegfr2.csv", stringsAsFactors = F)

# Search the merging way
output <- data.frame(longegfr1.col.name =  names(longegfr1))
kable(output,"markdown")
```

longegfr1.col.name
id
sex
baseline.age
fu.years

```r
output <- data.frame(longegfr2.col.name =  names(longegfr2))
kable(output,"markdown")
```

longegfr2.col.name
ID
fu.years
egfr

We can see from the upper result, 2 data tables have different numbers of observations and different variables. But 2 subjects both have patients' `id/ID` and `fu.years` , which need to be further researched.

```r
# further exploration for ID and fu.years variables
output <- data.frame(
  dataset = c("in longegfr1 not in longegfr2",
              "in longegfr2 not in longegfr1"),
  id = c(length(longegfr1[!id %in% longegfr2$ID]$id),
         length(longegfr2[!ID %in% longegfr1$id]$ID)),
  fu.years = c(length(longegfr1[!fu.years %in%
                                  longegfr2$fu.years]$fu.years),
               length(longegfr2[!fu.years %in%
                                  longegfr1$fu.years]$fu.years))
)
kable(output,"markdown")
```

dataset	id	fu.years
in longegfr1 not in longegfr2	11	50
in longegfr2 not in longegfr1	0	0

There are 11 times that patients' `id` have records in subject 1 but not in subject 2, and 50 times that `fu.years` records are in subject 1 but not in subject 2. Therefore, I will merge 2 subjects by variables `ID` and `fu.years`. For those observations who do not have record in some variables, fill `NA`.

```r
# Merge 2 subjects
longegfr <- merge(longegfr1,longegfr2,by.x = c("id","fu.years"),
                  by.y = c("ID","fu.years"),all = T)
# order observations according to 2 certain variables
longegfr <- longegfr[order(id,fu.years)]
# print first 10 values
head(longegfr,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
```

Problem 1.b (6 points)

- Compute the average eGFR and length of follow-up for each patient.
- Print first 10 values of the new dataset using `head()`.
- 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.

```r
#Answer in this chunk
# Compute the average eGFR and length of follow-up for each patient.
eGFR_fu <- longegfr[,as.list(
  c(.(fu_length = max(fu.years)-min(fu.years)),
    .(egfr_mean = mean(egfr,na.rm = T)))),
  by=list(id,sex,baseline.age)]

# print first 10 values of the new dataset
head(eGFR_fu,10)
```

```
    id sex baseline.age fu_length egfr_mean
 1:  1   0         65.5    6.4586  43.04333
 2:  2   1         83.2    2.0698  38.93294
 3:  3   1         19.6    6.5161  85.72000
 4:  4   0         50.3    5.2786  76.59308
 5:  5   1         72.1    6.3929  13.90892
 6:  6   1         65.5    6.2313  85.66435
 7:  7   0         73.0    5.8453  64.21758
 8:  8   1         84.7    1.5606  66.28333
 9:  9   0         72.6    5.8700  86.35750
10: 10   0         50.4    5.1964 107.00429
```

```r
# Tabulate the number of patients with average eGFR in the following ranges
cut_points <- c(seq(0,30,15),seq(60,90,30),Inf)
Tab_eGFR <- cut(eGFR_fu$egfr_mean,breaks = cut_points, right = T)
table(Tab_eGFR)
```

```
Tab_eGFR
  (0,15]  (15,30]  (30,60]  (60,90] (90,Inf]
       2        9       84       86       66
```

```r
# Count and report the number of patients with missing average eGFR
sum(is.na(eGFR_fu$egfr_mean))
```

```
[1] 3
```

The number of patients with missing average eGFR is 3.

Problem 1.c (6 points)

- For patients with average eGFR in the $(90, \texttt{max(eGFR)})$ range, collect their identifier, sex, age at baseline, average eGFR, time of last eGFR reading and number of eGFR measurements taken in a data table.
- Print the summary of the new dataset.

```r
#Answer in this chunk
# collect id,sex,age,average eGFR,last.eGFR and number of eGFR measurement
# here I consider the last reading sGFR not be "NA"
# initialize a new data.frame
eGFR_high <- data.frame(matrix(ncol = 6, nrow = 0))
# add needed records
for(i in eGFR_fu[egfr_mean>90]$id){
  new_row <- c(i,
               eGFR_fu[id == i]$sex,
               eGFR_fu[id == i]$baseline.age,
               eGFR_fu[id == i]$egfr_mean,
               last(longegfr[id == i]$egfr,na_rm = T),
               table(longegfr[id == i]$id))
  eGFR_high <- rbind(eGFR_high,new_row)
}
# rename column names for the new data.frame
colnames(eGFR_high) <- c("id","sex","age","mean.eGFR","last.eGFR","measure.num")

# print the summary of the new dataset
summary(eGFR_high)
```

```
       id              sex              age          mean.eGFR
 Min.   : 10.00   Min.   :0.0000   Min.   :22.10   Min.   : 90.04
 1st Qu.: 86.25   1st Qu.:0.0000   1st Qu.:47.20   1st Qu.: 99.13
 Median :144.00   Median :0.0000   Median :55.20   Median :109.81
 Mean   :141.88   Mean   :0.3333   Mean   :55.27   Mean   :112.13
```

```
3rd Qu.:197.50   3rd Qu.:1.0000   3rd Qu.:63.80   3rd Qu.:123.20
Max.   :250.00   Max.   :1.0000   Max.   :90.90   Max.   :147.69
  last.eGFR        measure.num
Min.   : 50.31   Min.   : 1.00
1st Qu.: 87.32   1st Qu.: 6.00
Median :121.84   Median :10.00
Mean   :118.58   Mean   :14.58
3rd Qu.:150.12   3rd Qu.:18.75
Max.   :174.43   Max.   :57.00
```

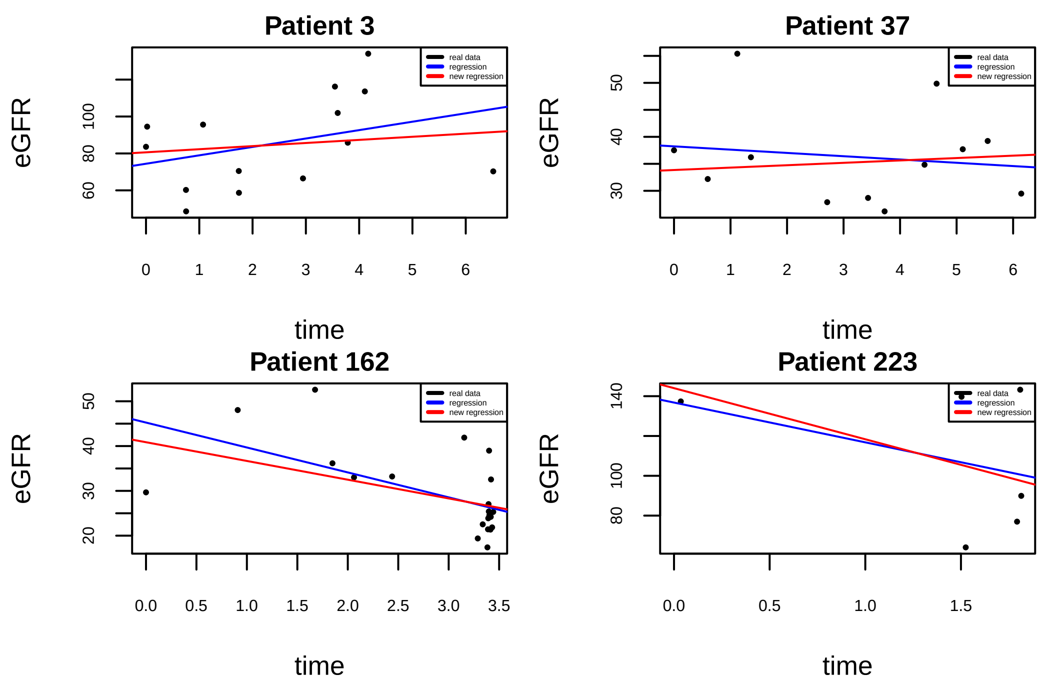
Problem 1.d (9 points)

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).

(All plots should be displayed in the same figure. 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 background with a very little statistical knowledge.)

```r
# define a plot function to plot eGFR based on time
plot.eGFR <- function(patient.id){
  # collect data we need
  patient <- longegfr[id==patient.id,c("fu.years","egfr")]
  # plot the original data
  par(mar = c(3.8,3.8,1.4,1))
  plot(patient$fu.years,patient$egfr,
       main=paste("Patient",patient.id),cex.main=1,
       xlab = "time",ylab = "eGFR",cex.lab=1,cex.axis = 0.6,
       pch=16,cex=0.5)
  # plot the regression line
  abline(lm(patient$egfr~patient$fu.years),col="blue")
  # plot the regression line without extreme value
  patient <- patient[-c(which.min(patient$egfr),which.max(patient$egfr)),]
  abline(lm(patient$egfr~patient$fu.years),col="red")
  # add legend
  legend("topright", legend = c("real data", "regression", "new regression"),
          col = c("black", "blue","red"), lwd = 2, cex = 0.3,bty = 1)
  }
# plot for certain patients
opar <- par(mfrow=c(2,2),mar=c(3.8,3.8,1.4,1))
for(i in c(3,37,162,223)){
  plot.eGFR(i)
}
```



```r
# calculate the 95% confidence intervals
for(i in c(3,37,162,223)){
  patient <- longegfr[id==i,c("fu.years","egfr")]
  # colnames(patient) <- c("time","eGFR")
  model <- lm(egfr~fu.years,patient)
  ci <- confint(model,level = 0.95)
  print(paste("Patient",i,":",sep = ""))
  print(ci)
}
```

```
[1] "Patient3:"
                 2.5 %   97.5 %
(Intercept) 50.623768 98.21718
```

```
fu.years    -3.151128 12.25612
[1] "Patient37:"
                2.5 %   97.5 %
(Intercept) 26.911518 49.55334
fu.years    -3.595705  2.37859
[1] "Patient162:"
                2.5 %    97.5 %
(Intercept) 34.109333 56.382006
fu.years    -9.257727 -1.872262
[1] "Patient223:"
                2.5 %   97.5 %
(Intercept)  34.71838 238.8642
fu.years    -85.93757  45.9659
```

The upper outputs is a table with two columns (**`2.5%`** and **`97.5%`**) and two rows (one for the intercept and one for the slope). The values in each rows represent the lower and upper bounds of the 95% confidence interval for that coefficient. For example, for patient 3, the 95% confidence intervals for slope is $(-3.15, 12.26]$, and the 95% confidence intervals for intercept is $(50.62, 98.22]$.

Problem 2 (25 points)

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

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

, and

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

, 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,

- 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.

```r
#Answer in this chunk
# load scr.csv file
scr.data <- fread("data_assignment1/scr.csv", stringsAsFactors = F)
# Examine a summary of the distribution of serum creatinine
summary(scr.data$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
# Report the inter-quartile range.
IQR(scr.data$scr,na.rm = T)
```

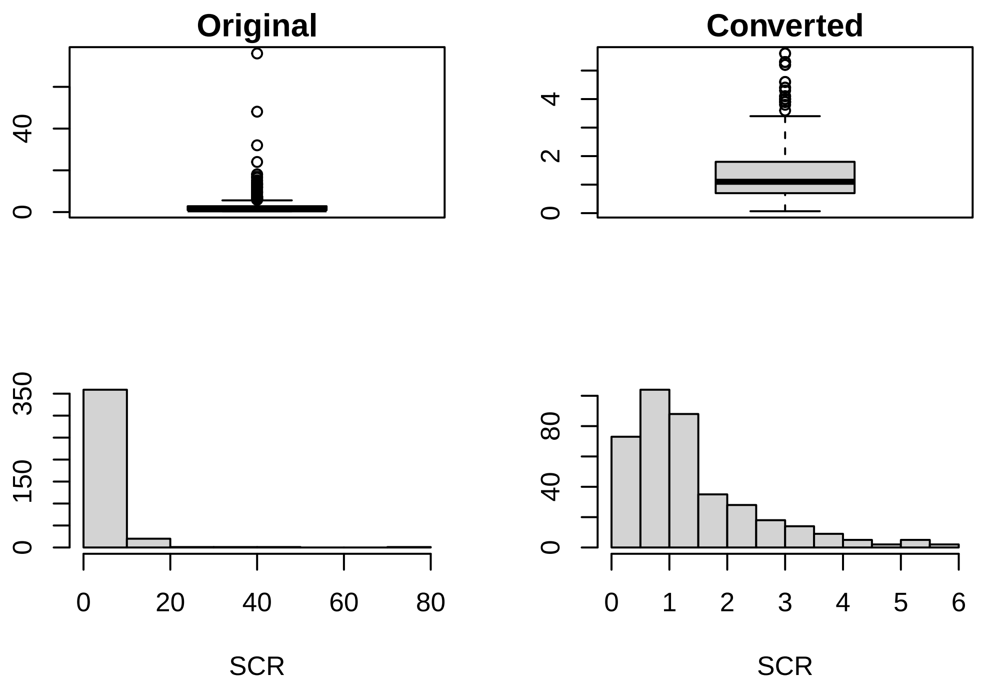
```
[1] 1.9
```

The inter-quartile range of serum creatinine, which is $Q_1 - Q_3$, is 1.9.

Generally, the normal range for serum creatinine in adults is: 0.6 to 1.1 mg/dL for males; 0.5 to 0.9 mg/dL for female. If records are extremely higher than these 2 ranges, we can suspect that they are recorded by unit µmol/L. The standard error of serum creatinine can vary depending on the population being studied, but a commonly used estimate is around 0.1-0.2 mg/dL, considered it as a Normal distribution, it is rarely to have records greater than $\mu + 3 \times \sigma^2$, which is $1.1 + 3 \times 0.2^2 = 1.22$. However, consider about people who are unhealthy may have uncommon values in this dataset, I need to choose a value bigger than 1.22 in this dataset.

In box plot, we will consider those values who are bigger than $Q_3 + 1.5 \times IQR$ or smaller than $Q_1 - 1.5 \times IQR$ as abnormal values. Therefore, I choose$Q_3 + 1.5 \times IQR$ to define values recorded by wrong unit.

```r
# covert abnormal value
scr.covert <- scr.data
bound <- quantile(scr.data$scr,0.75,na.rm = T)+1.5*IQR(scr.data$scr,na.rm = T)
scr.covert$scr <- ifelse(scr.covert$scr>=bound,scr.covert$scr/88.4,scr.covert$scr)

# examine the distribution of serum creatinine
opar <- par(mfrow=c(2,2),mar=c(3.8,3.8,1.4,1))
boxplot(scr.data$scr, main = "Original")
boxplot(scr.covert$scr, main = "Converted")
hist(scr.data$scr,xlab = "SCR", ylab = "",main = "")
hist(scr.covert$scr,xlab = "SCR" ,ylab = "",main = "")
```



We can see from the upper 4 graphs. The left are box plot and histogram plot of original data, the right are graphs of data after converted abnormal values. We can found that values which are extremely higher are converted to a normal range.

Problem 2.b (11 points)

- Compute the eGFR according to the two equations using the newly converted `SCR` values.
- Report (rounded to the second decimal place) mean and standard deviation of the two eGFR vectors and their Pearson correlation coefficient.
- Report the same quantities according to strata of MDRD4 eGFR: $(0-60)$, $(60-90)$ and (> 90).
- Print first 15 values for both datasets using `head()`.

```r
# ---- Compute eCFR according to 2 equations using ---- #
# initialize a data.frame to save results
Estimate.eGFR <- data.frame(matrix(ncol = 3, nrow = 0))
for(i in 1:nrow(scr.covert)){
  if(complete.cases(scr.covert[i, ])){
    # Equation1: MDRD4
    MDRD4 <- 175*(scr.covert$scr[i]^(-1.154))*
      (scr.covert$age[i]^(-0.203))*
```

```r
        ifelse(scr.covert$sex[i] == "Female",0.742,1)*
        ifelse(scr.covert$ethnic[i] =="Black",1.212,1)
      # Equation2: CKD-EPI
      k <- ifelse(scr.covert$sex[i]=="Female",0.7,0.9)
      alpha <- ifelse(scr.covert$sex[i]=="Female",-0.329,-0.411)
      CKD_EPI <- 141*min(scr.covert$scr[i]/k,1)^(alpha)*
        max(scr.covert$scr[i]/k,1)^(-1.209)*
        (0.993^(scr.covert$age[i]))*
        ifelse(scr.covert$sex[i] == "Female",1.018,1)*
        ifelse(scr.covert$ethnic[i] == "Black",1.159,1)
    }else{
      # for rows with Not Available reocord(s)
      MDRD4 <- NA
      CKD_EPI <- NA
    }
    new_row <- data.frame(i,MDRD4,CKD_EPI)
    Estimate.eGFR <- rbind(Estimate.eGFR,new_row)
  }
  # rename column names
  colnames(Estimate.eGFR) <- c("i","MDRD4","CKD_EPI")
```

```r
# ---- Report mean, standard value, Pearson correlation coefficient ---- #
# report mean, standard deviation and Pearson correlation coefficient
output <- data.frame(
  method = c("MRD4","CKD_EPI"),
  mean = c(round(mean(Estimate.eGFR$MDRD4,na.rm = T),2),
           round(mean(Estimate.eGFR$CKD_EPI,na.rm = T),2)),
  standard.value = c(round(sd(Estimate.eGFR$MDRD4,na.rm = T),2),
          round(sd(Estimate.eGFR$CKD_EPI,na.rm = T),2)),
  Pearson.correlation = c(round(cor(na.omit(Estimate.eGFR[,2]),
                                    na.omit(Estimate.eGFR[,3]),
                                    method = "pearson"),2),
                          round(cor(na.omit(Estimate.eGFR[,3]),
                                    na.omit(Estimate.eGFR[,2]),
                                    method = "pearson"),2))
)
kable(output, "markdown")
```

method	mean	standard.value	Pearson.correlation
MRD4	188.95	358.94	0.86

method	mean	standard.value	Pearson.correlation
CKD_EPI	85.84	64.26	0.86

```r
# ---- Report same quantities caccording to strata of MDRD4 eGFR---- #
output <- data.frame(matrix(ncol = 5, nrow = 0))
t <- 0
for(m in c(60,90,1000)){
  group <- subset(Estimate.eGFR,MDRD4<=m & MDRD4 > t)
  p.cor <- round(cor(na.omit(group$MDRD4),na.omit(group$CKD_EPI)),2)
  for(j in c("MDRD4","CKD_EPI")){
    new_row <- c(paste("(",t,",",i,")"),
                 j,
                 round(mean(group[,j]),2),
                 round(sd(group[,j]),2),
                 p.cor)

    output <- rbind(output,new_row)
  }
  t <- m
}
colnames(output) <- c("Group","Method","Mean",
                       "Standard.value","Pearson.cor")
kable(output, "markdown")
```

Group	Method	Mean	Standard.value	Pearson.cor
(0 , 401)	MDRD4	31.9	15.14	0.99
(0 , 401)	CKD_EPI	33.15	16.72	0.99
(60 , 401)	MDRD4	73.41	8.4	0.93
(60 , 401)	CKD_EPI	80.18	10.42	0.93
(90 , 401)	MDRD4	263.4	250.49	0.89
(90 , 401)	CKD_EPI	136.08	35.84	0.89

```r
# ---- Print first 15 values for both datasets---- #
head(Estimate.eGFR[2:3],15)
```

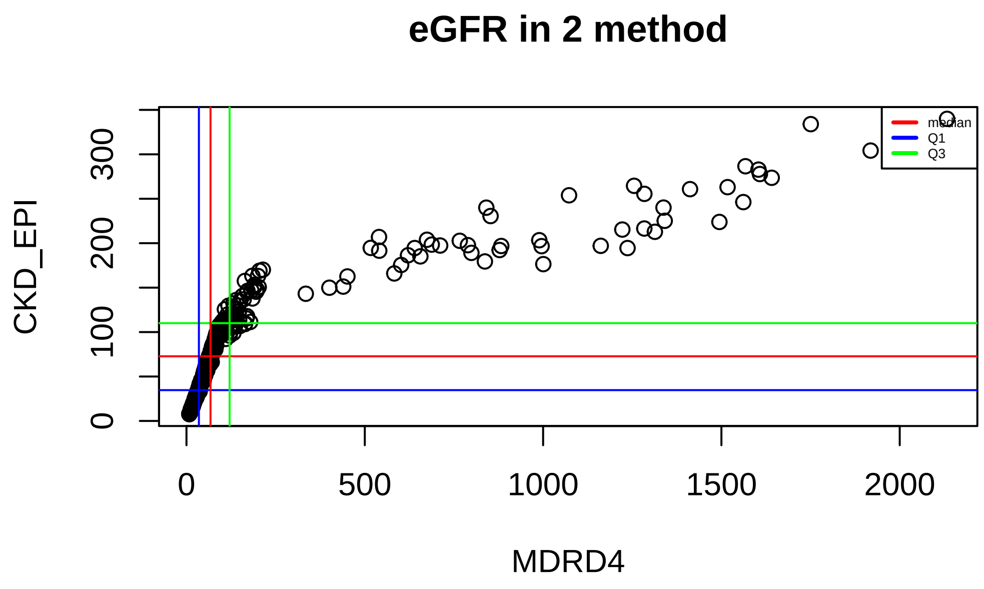
```
      MDRD4   CKD_EPI
1  47.94848  53.39791
2 184.85020 163.29428
```

```
3           NA        NA
4     12.67885  13.25244
5     53.42808  57.76186
6     68.28199  72.57875
7    334.56529 143.11552
8     99.67601 108.32282
9     27.75950  29.78779
10  1412.06361 260.82618
11    11.85151  12.28181
12    23.98712  23.99839
13    23.42079  23.58719
14    12.77074  12.16671
15    17.67620  16.20597
```

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.

```r
#Answer in this chunk
plot(Estimate.eGFR$MDRD4,Estimate.eGFR$CKD_EPI,type="p",
     ylab="CKD_EPI",xlab="MDRD4",
     main="eGFR in 2 method",col ="black")

# Add median and quantile lines for CKD_EPI
abline(h = median(na.omit(Estimate.eGFR$CKD_EPI)), col = "red")
abline(h = quantile(na.omit(Estimate.eGFR$CKD_EPI), 0.25), col = "blue")
abline(h = quantile(na.omit(Estimate.eGFR$CKD_EPI), 0.75), col = "green")

# Add median and quantile lines for MDRD4
abline(v = median(na.omit(Estimate.eGFR$MDRD4)),col = "red")
abline(v = quantile(na.omit(Estimate.eGFR$MDRD4), 0.25),col = "blue")
abline(v = quantile(na.omit(Estimate.eGFR$MDRD4), 0.75), col = "green")

# Add lengend
legend("topright", legend = c( "median", "Q1","Q3"),
         col = c("red","blue","green"), lwd = 2, cex = 0.4,bty = 1)
```



From the upper graph, we can find that the results for 2 methods to measure eGFR have strong linear relationship intuitively. And from problem 2.b, we can find that all Pearson correlation coefficients that we calculated are very close to 1. Therefore, the relationship between the two eGFR equations is linear.

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 `lab1.csv` dataset `LABID`.
- Ensure that a single `yob` field remains and rename it to `yob`.
- Ensure that the `albumin` field is converted to a factor and the ordering of the factor is `1="normo"`, `2="micro"`, `3="macro"`.
- Print first 10 values of the new dataset using `head()`.

```r
#Answer in this chunk
# load 3 files
cohort <- fread("data_assignment1/cohort.csv", stringsAsFactors = T)
lab1 <- fread("data_assignment1/lab1.csv", stringsAsFactors = T)
linker <- fread("data_assignment1/linker.csv", stringsAsFactors = F)

# merge 3 files to a single data table
cohort.dt <- merge(merge(lab1,linker,by.x = c("LABID"),
                   by.y = c("LABID"),all = T),
                   cohort ,by.x = c("id"),
                   by.y = c("id"),all = T)
```

```r
# check all identifiers in cohort.csv have been accounted for in the final table
check_ids <- setdiff(cohort$id,cohort.dt$id)
print(length(check_ids))
```

```
[1] 0
```

The length of `check_ids` is 0, which means there is no difference between cohort's id and cohort.dt's id, we can ensure that all identifiers in `cohort.csv` have been accounted for in the final table and that any validation fields are consistent between sets.

```r
# drop the identifier that originated from lab1.csv dataset LABID.
cohort.dt <- subset(cohort.dt, select = -c(LABID))
```

```r
# check if 2 yob are equal
for(i in 1:length(cohort.dt$id)){
  if(cohort.dt$yob.x[i] != cohort.dt$yob.y[i]){
    print(paste("notice",i))
  }
}
```

There are no outputs from the upper code block, which means each `yob` values in `cohort.csv` are equal to the corresponding value in `lab1.csv`. Therefore, we can delete one of them.

```r
# Ensure that a single yob field remains and rename it to yob
cohort.dt <- subset(cohort.dt, select = -c(yob.y))
setnames(cohort.dt, "yob.x", "yob")
```

```r
# Ensure that the albumin field is converted to a factor
# and the ordering of the factor is 1="normo", 2="micro", 3="macro".
cohort.dt$albumin <- factor(cohort.dt$albumin,
                            levels = c("normo","micro","macro"),labels = c(1,2,3))
```

```r
# Print first 10 values of the new dataset using head().
head(cohort.dt,10)
```

```
         id  yob urea creatinine glucose age  bp diabetes albumin
 1:   PID_1 1971   36    106.104     121  48  80        1       2
 2:  PID_10 1966  107    636.624      70  53  90        1       2
 3: PID_100 1963   24    106.104     298  56 180        1       1
 4: PID_101 1985   22     79.578     153  34  70        0       3
 5: PID_102 1948   80    389.048      88  71  90        0       2
 6: PID_103 2002   32    185.682      92  17  60        0       1
 7: PID_104 1943  217    901.884     226  76  70        0       2
 8: PID_105 1964   88    176.840     143  55  90        1    <NA>
 9: PID_106 1954   32   1016.830     115  65  80        0       1
10: PID_107 1969  118    539.362      89  50  90        1    <NA>
```

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.
- Perform mean imputation for all other continuous variables by writing a single function called `impute.to.mean()` and impute to mean, impute any categorical variable to the mode.
- Print first 15 values of the new dataset using `head()`.
- Compare each distribution of the imputed and non-imputed variables and decide which ones to keep for further analysis. Justify your answer.

```r
#Answer in this chunk
# Create a copy of the dataset where you will impute all missing values.
cohort.dt.imputed <- cohort.dt %>% copy()
# Update any missing age fields using the year of birth
# calculate a base year
for(i in 1:length(cohort.dt.imputed$id)){
  if(!(is.na(cohort.dt.imputed$yob[i]) &is.na(cohort.dt.imputed$age[i]))){
    base_year <- cohort.dt.imputed$yob[i] + cohort.dt.imputed$age[i]
    break
  }
}
# impute age
cohort.dt.imputed <- cohort.dt.imputed[,age:=ifelse(is.na(age),base_year-yob,age)]
```

```r
# writing a single function called impute.to.mean()
impute.to.mean <- function(dataset){
  mean <- mean(dataset,na.rm = T)
  dataset <- ifelse(is.na(dataset),mean,dataset)
  return(dataset)}

# writing a single function called impute.to.mode()
impute.to.mode <- function(dataset){
  mode <- as.numeric(names(table(dataset)[which.max(table(dataset))]))
  dataset <- ifelse(is.na(dataset),mode,dataset)
  dataset <- factor(dataset)
  return(dataset)}

# impute missing values (mean)
cohort.dt.imputed$urea <- impute.to.mean(cohort.dt.imputed$urea)
cohort.dt.imputed$creatinine <- impute.to.mean(cohort.dt.imputed$creatinine)
cohort.dt.imputed$bp <- impute.to.mean(cohort.dt.imputed$bp)
cohort.dt.imputed$glucose <- impute.to.mean(cohort.dt.imputed$glucose)

# impute missing values (mode)
cohort.dt.imputed$diabetes <- impute.to.mode(cohort.dt.imputed$diabetes)
cohort.dt.imputed$albumin <- impute.to.mode(cohort.dt.imputed$albumin)
```

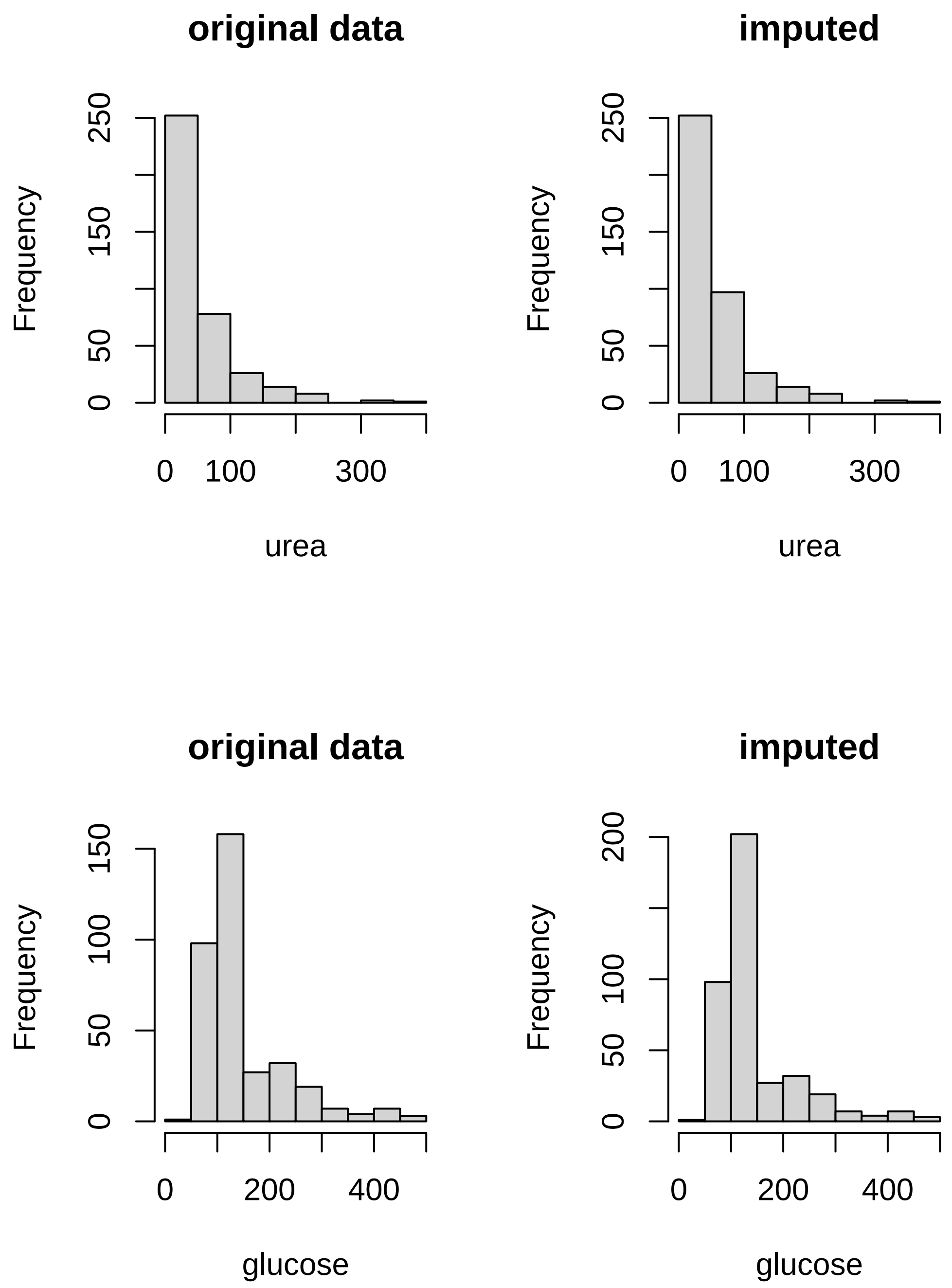
```r
# I used to write the function in this code block, however I find that "diabetes"
# is a 0-1 variable, and we cannot use a R function t distinguish it.
# Therefore I change my answer which is showed in the upper code block.

# impute.to.mean <- function(dataset){
#   for(i in 1:ncol(dataset)){
#     if(is.numeric(dataset[[i]])){
#       if(sum(is.na(dataset[[i]])>0)){
#         impute.mean <- mean(dataset[[i]],na.rm = T)
#         dataset[[i]] <- ifelse(is.na(dataset[[i]]),impute.mean,dataset[[i]])
#       }
#     }else{
```

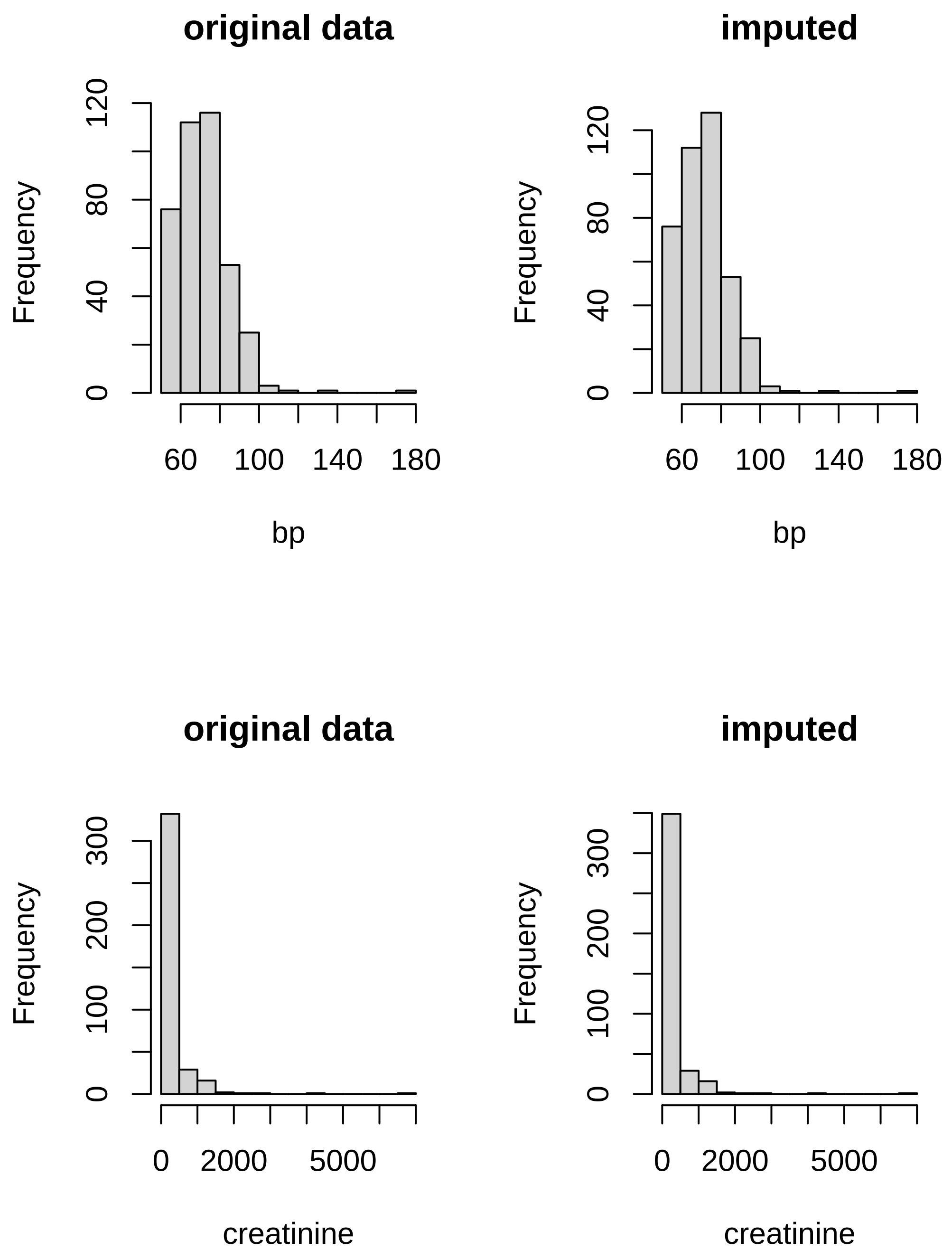
```
#       if(sum(is.na(dataset[[i]])>0)){
#         mode <- as.numeric(names(table(dataset)[which.max(table(dataset))]))
#         dataset[[i]] <- ifelse(is.na(dataset[[i]]),mode,dataset[[i]])
#         dataset[[i]] <- factor(dataset[[i]])
#       }
#     }
#   }
#   return(dataset)
# }
```

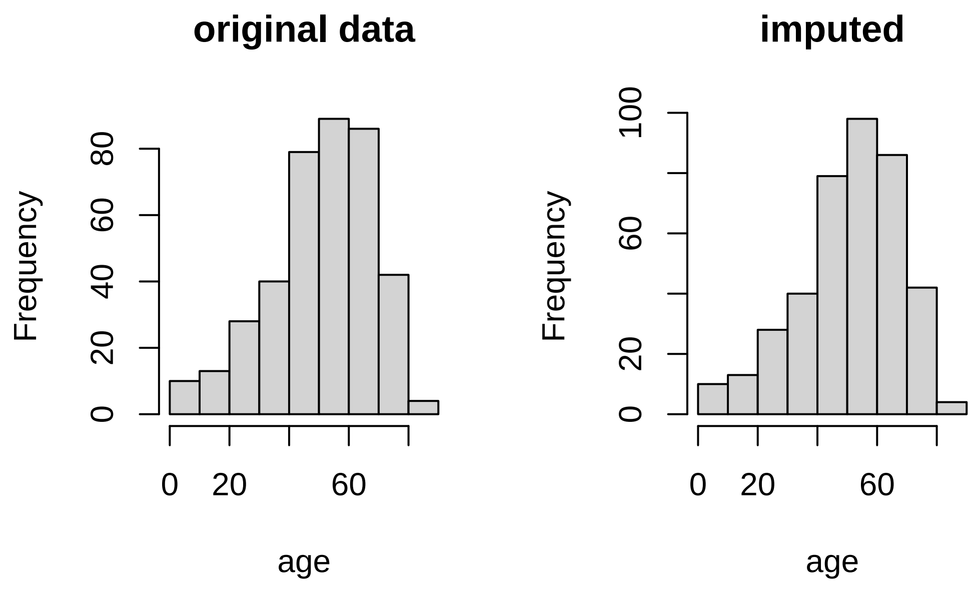
```
# Print first 15 values of the new dataset using head().
head(cohort.dt.imputed,15)
```

```
         id  yob  urea creatinine glucose age  bp diabetes albumin
 1:   PID_1 1971  36.0    106.104     121  48  80        1       2
 2:  PID_10 1966 107.0    636.624      70  53  90        1       2
 3: PID_100 1963  24.0    106.104     298  56 180        1       1
 4: PID_101 1985  22.0     79.578     153  34  70        0       3
 5: PID_102 1948  80.0    389.048      88  71  90        0       2
 6: PID_103 2002  32.0    185.682      92  17  60        0       1
 7: PID_104 1943 217.0    901.884     226  76  70        0       2
 8: PID_105 1964  88.0    176.840     143  55  90        1       1
 9: PID_106 1954  32.0   1016.830     115  65  80        0       1
10: PID_107 1969 118.0    539.362      89  50  90        1       1
11: PID_108 1964  53.0    247.576     297  55 100        1       2
12: PID_109 1974  15.0     88.420     107  45  80        0       1
13:  PID_11 1969  55.0    353.680     490  50  60        1       2
14: PID_110 1965  50.1    167.998     233  54  70        1       1
15: PID_111 1956  19.0    176.840     123  63  90        0       1
```

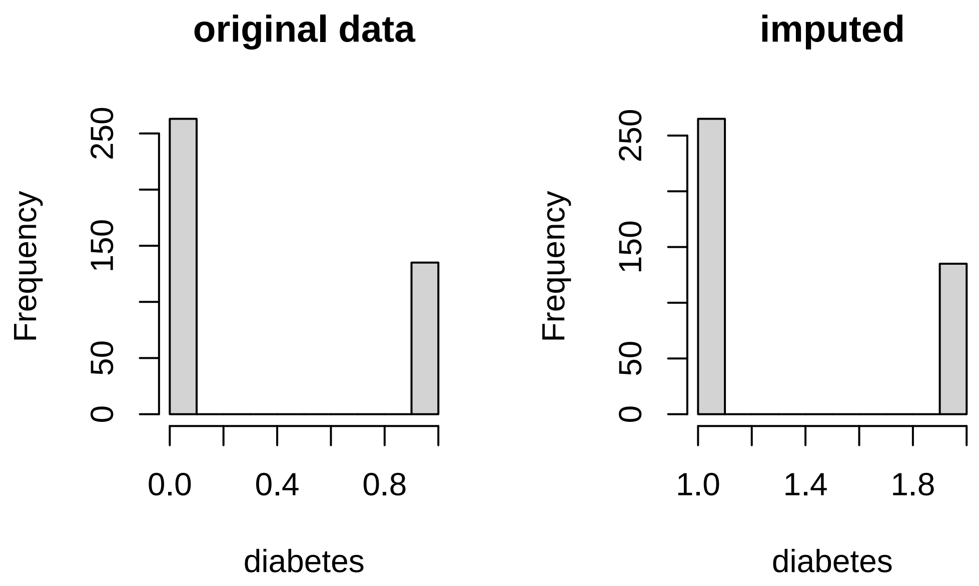
```
# Compare each distribution of the imputed and non-imputed variables
# and decide which ones to keep for further analysis.
for(i in c("urea","glucose","bp","creatinine","age")){
  par(mfrow = c(1,2))
  hist(cohort.dt[[i]], xlab = i, main = "original data")
  hist(cohort.dt.imputed[[i]], xlab = i, main = "imputed")
}
```

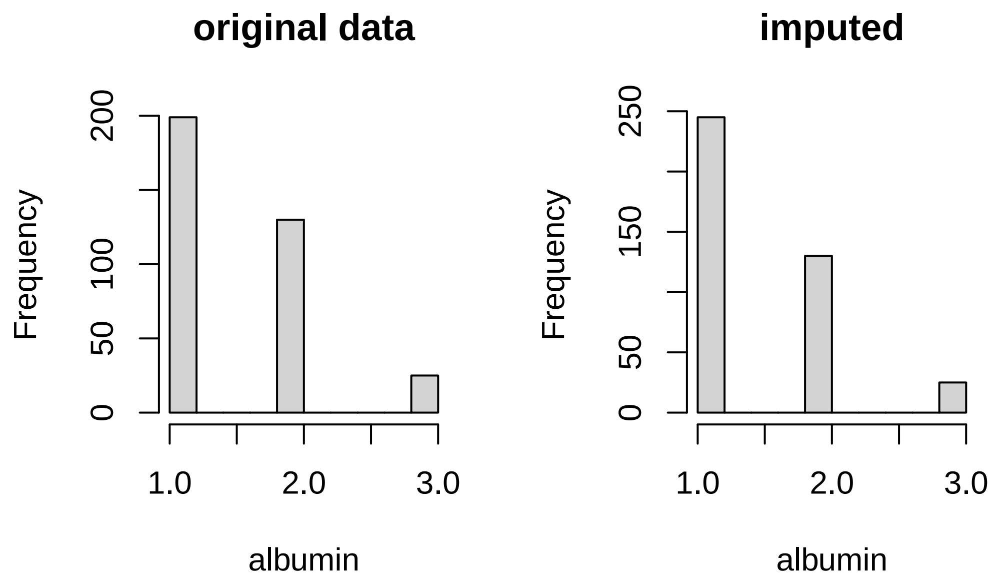

original data
imputed
Frequency
0
50
150
250
0
100
300
urea
original data
imputed
Frequency
0
50
100
150
200
0
200
400
glucose


original data
imputed
Frequency
0
40
80
120
60
100
140
180
bp
original data
imputed
Frequency
0
100
200
300
0
2000
5000
creatinine



```r
for(i in c("diabetes","albumin")){
  par(mfrow = c(1,2))
  hist(as.numeric(cohort.dt[[i]]), xlab = i, main = "original data")
  hist(as.numeric(cohort.dt.imputed[[i]]), xlab = i, main = "imputed")
}
```





We have 7 variables which are: `urea`,`glucose`,`bp`,`creatinine`,`age`, `diabetes`,`albumin`.(`age` can be transfered from `yob`, so we only keep one of them). As the imputation of `age` is from real recording data yob, therefore those imputed data are as same as real data, we can keep it.

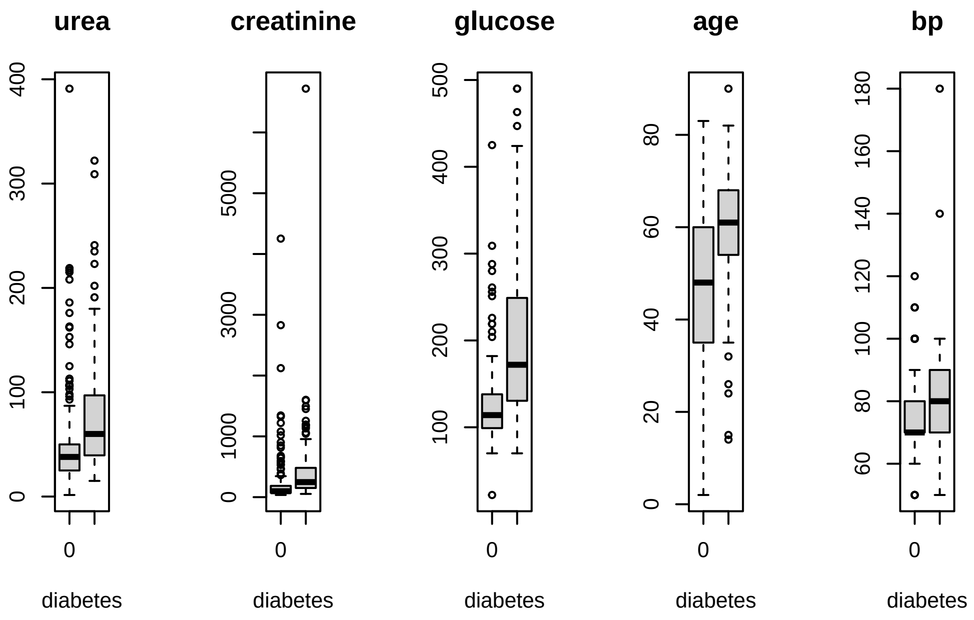
For other variables, we can find that only albumin has a clear difference when values equal to 1. The frequency when albumin= 1 is less than 200 before imputed but almost 250 after imputed. Therefore, we do not choose it.

In conclusion, we choose urea,glucose,bp,creatinine,age, diabetes for further analysis.

Problem 3.c (6 points)

- Plot a single figure containing boxplots of potential predictors for diabetes grouped by cases and controls. (Hint : par(mfrow=c(1,5))))
- Use these to decide which predictors to keep for future analysis.
- For any categorical variables create a table instead. Justify your answers.

```r
#Answer in this chunk
# Plot a single figure containing boxplots of potential predictors for diabetes
par(mfrow=c(1,5))
# boxplot(cohort.dt.imputed$yob ~ cohort.dt.imputed$diabetes)
for(k in colnames(cohort.dt.imputed[,-c("diabetes","yob","id","albumin")])){
  boxplot(cohort.dt.imputed[,get(k)] ~ cohort.dt.imputed$diabetes,
          main = k, xlab = "diabetes",ylab = "")
}
```



We can see from the upper 5 graphs that `urea`, `glucose` and `age` can be keep for future analysis.

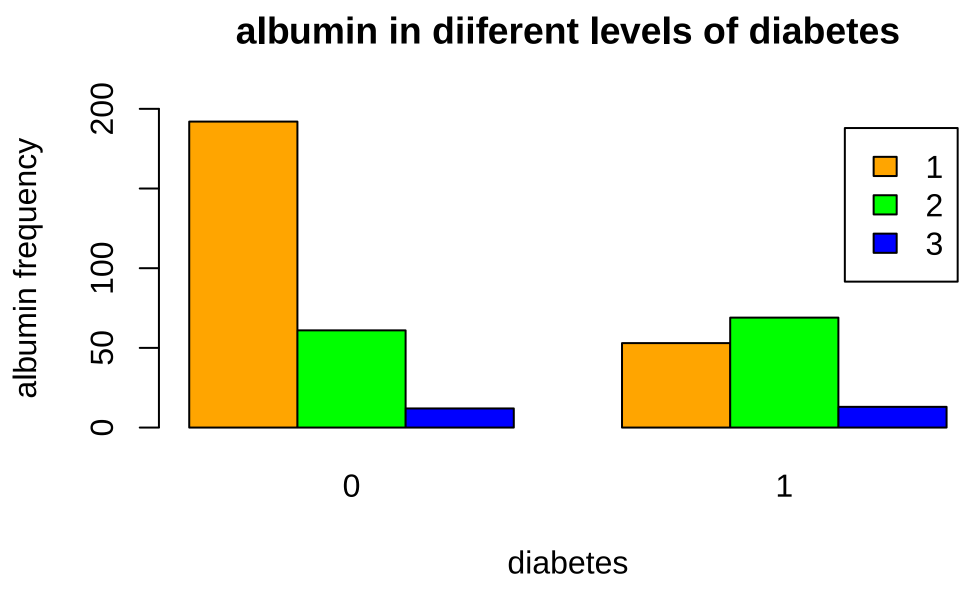
For these 3 variables, the distributions' shapes are similar when `diabetes = 0` and `diabetes = 1`. The median values are all about in the middle of Q1 and Q2. And the proportion of length between $Q1 - IQR \times 1.5$ and $Q2 + IQR \times 1.5$ when `diabetes = 0` and `diabetes = 1` are similar for these 3 variables.

However, for another variables which are `creatinine` and `bp` the box plots are much more different. For `creatinine`, data are much more concentrating when `diabetes = 0` comparing with when `diabetes = 1` .For `bp` , the median is much more close to Q1 for diabetes $= 0$ comparing with when `diabetes = 1`.

```r
# For any categorical variables create a table instead.
cate.var <- table(cohort.dt.imputed$albumin,cohort.dt.imputed$diabetes)
cate.var
```

```
      0   1
1   192  53
2    61  69
3    12  13
```

```r
barplot(cate.var, beside=TRUE, legend=TRUE,
        main = "albumin in diiferent levels of diabetes",
        ylim=c(0, 200),
        ylab = "albumin frequency",
        xlab = "diabetes",
        col=c("orange","green","blue"))
```



From the table's figure and the histogram plot, we can see that there no clear relations between `diabetes` and `albumin`. Therefore, we do not choose to keep it.

Problem 3.d (9 points)

- Use your findings from the previous exercise and 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 background with a very little statistical knowledge.

```r
#Answer in this chunk
# findings from the previous exercise: urea, glucose and age
# fit an appropriate model of diabetes with two predictors.
```

```r
regr.dia.au <-glm(diabetes ~ age+urea, data = cohort.dt.imputed,
                  family = binomial(link="logit"))
# Print a summary
summary(regr.dia.au)
```

```

Call:
glm(formula = diabetes ~ age + urea, family = binomial(link = "logit"),
    data = cohort.dt.imputed)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.9564  -0.8147  -0.4880   1.0432   2.6360

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -4.396010   0.550389  -7.987 1.38e-15 ***
age          0.054019   0.008996   6.005 1.92e-09 ***
urea         0.013131   0.002682   4.895 9.83e-07 ***
---
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: 425.42  on 397  degrees of freedom
AIC: 431.42

Number of Fisher Scoring iterations: 4
```

Explain the result to a colleague with a medical background and little statistical knowledge:

We are doing a work to fit an appropriate model with 2 predictors. First, I need to choose the predictors. From the previous exercise, we have chose `urea`, `glucose` and `age` to do further analysis. As `age` is accurate (imputed from recording `yob` values), I'll keep it and randomly choose another one, here I choose urea. Second, we fit this model in R by using glm() function to generalising linear models. The first parameter means we need to fit a model to predict "diabetes" based on "age" and "urea". And the data are come from dataset "cohort.dt.imputed". As "diabetes" is a 0-1 variables, therefore for "family" parameter, we choose "binomial", and the link function "logit" is the standard option for binomial family.

We can see the result from the output of summary() function. The first column called "Estimate" shows the regression coefficients that the model gives us. In this model, it is show as:$predict.diabetes = -4.39 + 0.054 \times age + 0.13 \times urea$. And the last column $Pr(> |z|)$ tells us if this factor is statistical significant($p < 0.05$). In our model, the 2 factors are both statistical significant.

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 background with a very little statistical knowledge.
- Print a summary and explain the results as you would communicate it to a colleague with a medical background with a very little statistical knowledge.

```r
#Answer in this chunk
# fit the model with 3 predictors
regr.dia.aug <-glm(diabetes ~ age+urea+glucose, data = cohort.dt.imputed,
                   family = binomial(link="logit"))
# compare models by a likelihood ratio test
pval <- pchisq(regr.dia.au$deviance - regr.dia.aug$deviance, df=1, lower.tail=FALSE)
# report p-value
signif(pval, 2)
```

```
[1] 7.1e-19
```

The p value is further less than 0.5, therefore, the additional term is significant. To get more support, we use AIC and BIC to compare 2 models.

```r
output <- data.frame(
  model = c("AIC with 2 predictors","AIC with 3 predictors"),
  AIC = c(regr.dia.au$aic,regr.dia.aug$aic),
  BIC = c(BIC(regr.dia.au),BIC(regr.dia.aug))
)
kable(output,"markdown")
```

model	AIC	BIC
AIC with 2 predictors	431.4155	443.3899
AIC with 3 predictors	354.6715	370.6374

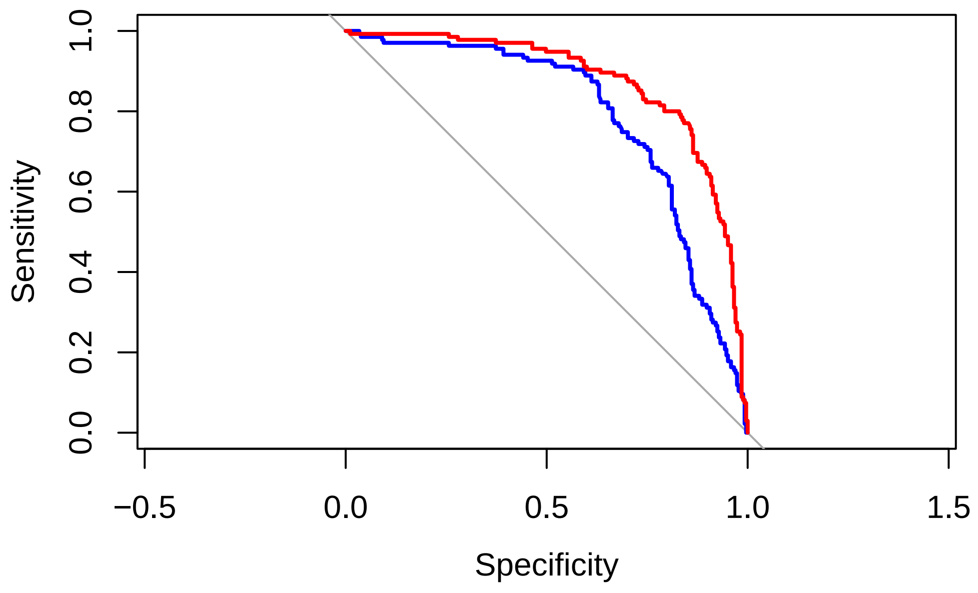
As the AIC and BIC for 3 predictors both lower than for 2 predictors, therefore, model with 3 predictors performs better.

```r
# Plot a ROC curve for both models and report the AUC
roc(cohort.dt.imputed$diabetes, regr.dia.au$fitted.values,
    plot = TRUE, xlim = c(0,1),col = "blue")
```

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

Setting direction: controls < cases


Call:
roc.default(response = cohort.dt.imputed$diabetes, predictor = regr.dia.au$fitted.values,

Data: regr.dia.au$fitted.values in 265 controls (cohort.dt.imputed$diabetes 0) < 135 cases (
Area under the curve: 0.7869
```

```r
roc(cohort.dt.imputed$diabetes, regr.dia.aug$fitted.values,
    plot = TRUE, xlim = c(0,1),add = TRUE, col = "red")
```

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



```
Call:
roc.default(response = cohort.dt.imputed$diabetes, predictor = regr.dia.aug$fitted.values,

Data: regr.dia.aug$fitted.values in 265 controls (cohort.dt.imputed$diabetes 0) < 135 cases
Area under the curve: 0.8745
```

Explain the results to a colleague with a medical background with a very little statistical knowledge:

AUC represents Area Under Curve, which is an important value to compare models or measure if the model fitting good or bad. It gives the overall probability of correctly ranking a randomly chosen case above a randomly chosen control, which means, with a bigger area under the curve, the model represented by this curve is better. Therefore, from the upper graph, the model with 3 predictors, which is represented by the red curve, is better than model with 2 predictors.

```r
# Print a summary
summary(regr.dia.aug)
```

```
Call:
glm(formula = diabetes ~ age + urea + glucose, family = binomial(link = "logit"),
    data = cohort.dt.imputed)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.0576  -0.6495  -0.3912   0.6083   2.9107

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -6.590163   0.707935  -9.309  < 2e-16 ***
age          0.047483   0.009922   4.786 1.70e-06 ***
urea         0.012717   0.002959   4.298 1.73e-05 ***
glucose      0.017042   0.002472   6.894 5.42e-12 ***
---
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: 346.67  on 396  degrees of freedom
AIC: 354.67
```

```
Number of Fisher Scoring iterations: 5
```

Explain the results to a colleague with a medical background with a very little statistical knowledge:

We are doing a work to fit an appropriate model with 2 predictors. First, from the previous exercise, we have chose **`urea`**, **`glucose`** and **`age`** to do further analysis. Second, we fit this model in R by using glm() function to generalising linear models. The first parameter means we need to fit a model to predict "diabetes" based on "age" and "urea". And the data are come from dataset "cohort.dt.imputed". As "diabetes" is a 0-1 variables, therefore for "family" parameter, we choose "binomial", and the link function "logit" is the standard option for binomial family.

We can see the result from the output of summary() function. The first column called "Estimate" shows the regression coefficients that the model gives us. In this model, it is show as:$predict.diabetes = -6.59 + 0.047 \times age + 0.13 \times urea + 0.17 \times glucose$. And the last column $Pr(> |z|)$ tells us if this factor is statistical significant($p < 0.05$). In our model, the 3 factors are both statistical significant.

Problem 4.b (10 points)

- Perform 10-folds cross validation for your chosen model based on the above answers.
- Report the mean cross-validated AUCs in 3 significant figures.

```r
#Answer in this chunk
# Perform 10-folds cross validation for chosen model
set.seed(1)
num.folds <- 10
folds <- createFolds(cohort.dt.imputed$diabetes, k = num.folds)
# for 2 predictors model,fit 10-fold cv models
regr.cv.au <- NULL
auc.cv.au <- NULL
for(f in 1:num.folds) {
  train.idx <- setdiff(1:nrow(cohort.dt.imputed), folds[[f]])
  regr.cv.au[[f]] <- glm(diabetes ~ age + urea,
                        data = cohort.dt.imputed,
                        subset = train.idx, family = "binomial")
  auc.cv.au[f] <- roc(cohort.dt.imputed[train.idx]$diabetes
                        ~regr.cv.au[[f]]$fitted.values)$auc
}
```

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

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```r
# for 3 predictors model,fit 10-fold cv models
regr.cv.aug <- NULL
auc.cv.aug <- NULL
for(f in 1:num.folds) {
  train.idx <- setdiff(1:nrow(cohort.dt.imputed), folds[[f]])
  regr.cv.aug[[f]] <- glm(diabetes ~ age + urea + glucose,
                          data = cohort.dt.imputed,
                          subset = train.idx, family = "binomial")
  auc.cv.aug[f] <- roc(cohort.dt.imputed[train.idx]$diabetes
                          ~regr.cv.aug[[f]]$fitted.values)$auc
}
```

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

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

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

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases

Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```r
# report the mean cross-validated AUCs in 3 significant figures.
output <- data.frame(
  Model = c("2 predictors","3 predictors"),
  Mean.cv.AUCs = c(
    round(mean(auc.cv.au),3),
    round(mean(auc.cv.aug),3)
  )
)
kable(output,"markdown")
```

Model	Mean.cv.AUCs
2 predictors	0.787
3 predictors	0.875