Assignment 1

Ted Ladas - s2124289

25-Feb-2021

Assignment 1

Biomedical Data Science

Due on Thursday 25th February 2020, 5:00pm

The assignment is marked out of 100 points, and will contribute to 20% of your final mark. Please knit this document in PDF format and submit using the gradescope link on Learn. If you can't knit to PDF directly, knit it to word and you should be able to either convert to PDF or print it and scan to PDF using a scanning app on your phone. If you have any code that doesn't run you won't be able to knit the document so comment it as you might still get some grades for partial code. Clear and reusable code will be rewarded so pay attention to indentation, choice of variable identifiers, comments, error checking, etc. 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 as and when required and any comments necessary within code chunks to make it easier to follow your code/reasoning.

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 (or tibble) and merge in an appropriate way into a single data table, then order the observations according to subject identifier and follow-up time.

```
is.nan.data.frame = function(x)
  do.call(cbind, lapply(x, is.nan))
```

```
# Enter code here.

# reading datasets and setting primary keys
longegfr1 = data.table(fread('./data/longegfr1.csv'), key=c('id', 'fu.years'))
longegfr2 = data.table(fread('./data/longegfr2.csv'), key=c('ID', 'fu.years'))
colnames(longegfr2) = c('id','fu.years','egfr') # change column names to make joining easier
pk = c('id','fu.years')
longegfr = merge(longegfr1,
                 longegfr2,
                 by=pk,
                 all=TRUE)[order(id,fu.years)] # FULL OUTER JOIN
longegfr %>% summary()
```

```
##        id           fu.years           sex          baseline.age
##  Min.   :  1.0   Min.   :0.0000   Min.   :0.0000   Min.   :18.3
##  1st Qu.: 58.0   1st Qu.:0.8597   1st Qu.:0.0000   1st Qu.:54.7
##  Median :123.0   Median :2.3682   Median :0.0000   Median :63.5
##  Mean   :118.9   Mean   :2.6598   Mean   :0.4297   Mean   :63.2
##  3rd Qu.:177.0   3rd Qu.:4.4353   3rd Qu.:1.0000   3rd Qu.:74.4
##  Max.   :250.0   Max.   :6.6283   Max.   :1.0000   Max.   :91.4
##
##       egfr
##  Min.   :  4.83
##  1st Qu.: 41.05
##  Median : 61.08
##  Mean   : 66.31
##  3rd Qu.: 86.44
##  Max.   :174.94
##  NA's   :212
```

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.

```
# Enter code here.
longegfr[,mean_egfr:=mean(egfr),
         by=id]
longegfr[,count_egfr:=length(fu.years),
         by=id]
```

```
longegfr[,fu_total:=max(fu.years)-min(fu.years),
         by=id]
summary(longegfr)
```

```
##        id           fu.years           sex          baseline.age
##  Min.   :  1.0   Min.   :0.0000   Min.   :0.0000   Min.   :18.3
##  1st Qu.: 58.0   1st Qu.:0.8597   1st Qu.:0.0000   1st Qu.:54.7
##  Median :123.0   Median :2.3682   Median :0.0000   Median :63.5
##  Mean   :118.9   Mean   :2.6598   Mean   :0.4297   Mean   :63.2
##  3rd Qu.:177.0   3rd Qu.:4.4353   3rd Qu.:1.0000   3rd Qu.:74.4
##  Max.   :250.0   Max.   :6.6283   Max.   :1.0000   Max.   :91.4
##
##       egfr          mean_egfr        count_egfr       fu_total
##  Min.   :  4.83   Min.   : 14.87   Min.   : 1.00   Min.   :0.000
##  1st Qu.: 41.05   1st Qu.: 44.12   1st Qu.:13.00   1st Qu.:3.817
##  Median : 61.08   Median : 59.28   Median :24.00   Median :5.287
##  Mean   : 66.31   Mean   : 60.53   Mean   :28.09   Mean   :4.680
##  3rd Qu.: 86.44   3rd Qu.: 76.59   3rd Qu.:39.00   3rd Qu.:6.097
##  Max.   :174.94   Max.   :147.69   Max.   :88.00   Max.   :6.628
##  NA's   :212      NA's   :758
```

```
longegfr
```

```
##        id fu.years sex baseline.age   egfr mean_egfr count_egfr fu_total
##    1:   1   0.0000   0         65.5  76.48  43.04333         15   6.4586
##    2:   1   0.1533   0         65.5  47.36  43.04333         15   6.4586
##    3:   1   0.6899   0         65.5  94.87  43.04333         15   6.4586
##    4:   1   1.1882   0         65.5  52.12  43.04333         15   6.4586
##    5:   1   1.8398   0         65.5  91.91  43.04333         15   6.4586
##   ---
## 4027: 249   1.9521   1         50.2  91.94  75.59571          7   2.6174
## 4028: 249   2.1246   1         50.2  69.51  75.59571          7   2.6174
## 4029: 249   2.5982   1         50.2  53.28  75.59571          7   2.6174
## 4030: 249   2.6174   1         50.2  66.78  75.59571          7   2.6174
## 4031: 250   0.0000   1         48.6 101.23 101.23000          1   0.0000
```

```
longegfr$bag_egfr = cut(longegfr$mean_egfr, c(0,15,30,60,90,Inf))

longegfr_nas = longegfr[,sum(is.na(mean_egfr)), by=id]
longegfr_nas[,missing_ind:=ifelse(V1>0,1,0)]
patients_egfr_na = longegfr_nas[,sum(missing_ind)]
cat('There are', patients_egfr_na, 'patients with missing values for \'mean egfr\'')
```

```
## There are 39 patients with missing values for 'mean egfr'
```

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.

```
# Enter code here.
egfr_90_plus = unique(longegfr[mean_egfr>90, .(id,sex,baseline.age,mean_egfr, count_egfr),])[order(mean
egfr_90_plus
```

```
##      id sex baseline.age mean_egfr count_egfr
##  1: 120   0          90.9  90.04000          2
##  2: 170   0          87.0  90.56000          2
##  3: 157   0          63.8  90.57308         13
##  4: 140   0          51.6  90.60929         28
##  5: 112   1          77.8  90.66500          6
##  6:  45   1          24.9  91.25000          1
##  7:  79   0          65.6  91.45057         35
##  8:  52   1          56.3  93.31544         57
##  9: 196   1          62.5  94.26000          2
## 10: 115   0          70.3  94.56900         10
## 11: 177   1          78.7  94.85769         26
## 12:  25   0          40.1  95.35625          8
## 13: 169   0          82.8  97.12400         10
## 14: 250   1          48.6 101.23000          1
## 15:  92   1          41.2 101.33882         17
## 16: 100   0          63.0 101.86769         13
## 17: 242   0          54.3 102.24000          4
## 18: 241   1          62.3 105.25200          5
## 19: 102   0          38.7 105.96000         10
## 20: 220   1          47.0 106.00857          7
## 21: 215   1          45.4 106.08278         54
## 22:  80   0          67.7 106.09600          5
## 23:  10   0          50.4 107.00429          7
## 24:  81   0          38.8 108.32000          8
## 25: 245   1          50.5 111.02900         10
## 26: 238   1          55.1 113.37833          6
## 27:  31   0          74.8 113.59250          8
## 28: 205   0          59.9 114.84833          6
## 29:  14   0          65.1 116.09200         10
## 30:  33   0          74.2 116.35000          4
## 31: 234   1          55.3 116.38250          4
## 32: 218   0          56.6 117.65750          4
## 33: 247   0          48.4 118.70667          9
## 34:  49   1          68.2 128.25800          5
## 35: 134   0          31.7 133.29500          2
## 36: 173   0          22.1 147.69000          1
##      id sex baseline.age mean_egfr count_egfr
```

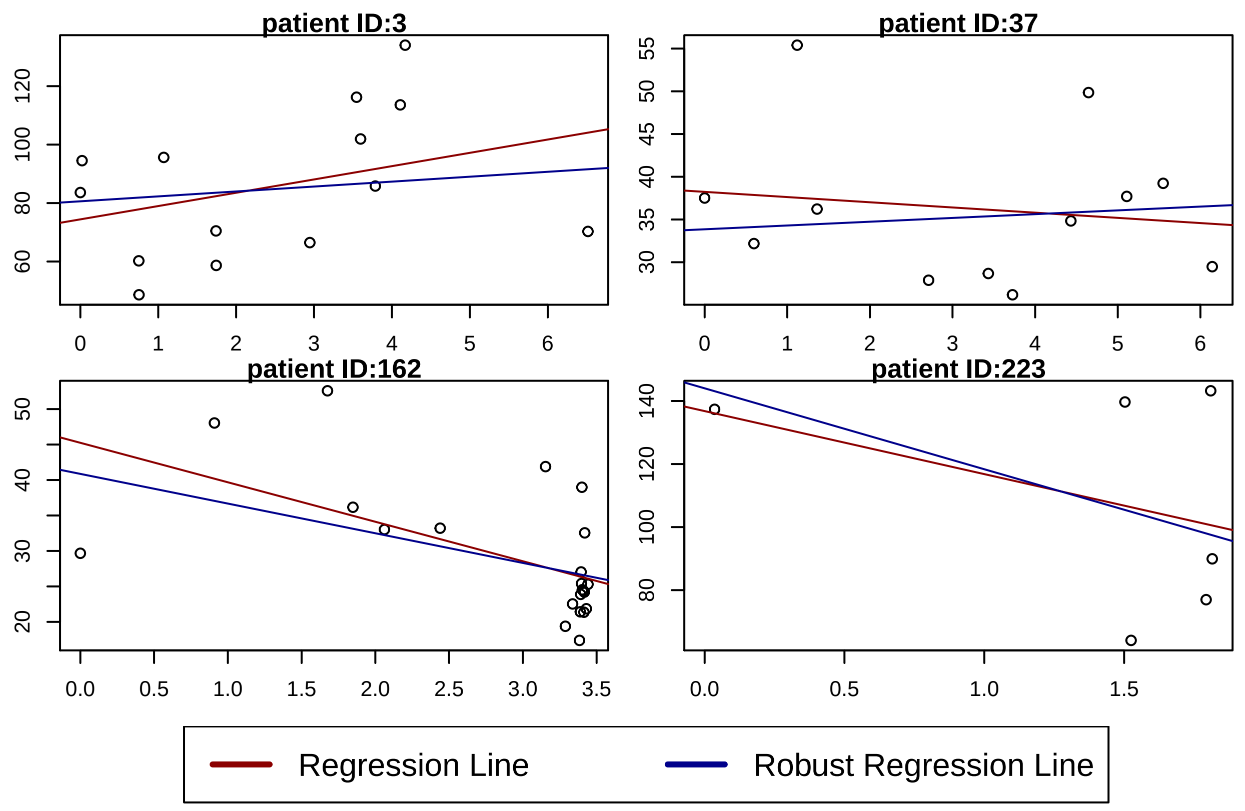
Problem 1.d (9 points)

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

```r
# Enter code here.
patient_id = c(3,37,162,223)
subset = longegfr[id %in% patient_id, .(id, fu.years, egfr)][order(id, fu.years)]
subset_1 = subset[id==patient_id[1]]
subset_2 = subset[id==patient_id[2]]
subset_3 = subset[id==patient_id[3]]
subset_4 = subset[id==patient_id[4]]
```

```
subset_1_rob = subset_1[!(egfr==max(egfr, na.rm=TRUE) | egfr==min(egfr, na.rm=TRUE))]
subset_2_rob = subset_2[!(egfr==max(egfr, na.rm=TRUE) | egfr==min(egfr, na.rm=TRUE))]
subset_3_rob = subset_3[!(egfr==max(egfr, na.rm=TRUE) | egfr==min(egfr, na.rm=TRUE))]
subset_4_rob = subset_4[!(egfr==max(egfr, na.rm=TRUE) | egfr==min(egfr, na.rm=TRUE))]

reg1 = lm(subset_1[,egfr] ~ subset_1[,fu.years])
reg2 = lm(subset_2[,egfr] ~ subset_2[,fu.years])
reg3 = lm(subset_3[,egfr] ~ subset_3[,fu.years])
reg4 = lm(subset_4[,egfr] ~ subset_4[,fu.years])
reg1_rob = lm(subset_1_rob[,egfr] ~ subset_1_rob[,fu.years])
reg2_rob = lm(subset_2_rob[,egfr] ~ subset_2_rob[,fu.years])
reg3_rob = lm(subset_3_rob[,egfr] ~ subset_3_rob[,fu.years])
reg4_rob = lm(subset_4_rob[,egfr] ~ subset_4_rob[,fu.years])

coef_1 = summary(reg1, conf.int=TRUE)$coefficients[2,1]
sd_1   = summary(reg1, conf.int=TRUE)$coefficients[2,2]
coef_2 = summary(reg2, conf.int=TRUE)$coefficients[2,1]
sd_2   = summary(reg2, conf.int=TRUE)$coefficients[2,2]
coef_3 = summary(reg3, conf.int=TRUE)$coefficients[2,1]
sd_3   = summary(reg3, conf.int=TRUE)$coefficients[2,2]
coef_4 = summary(reg4, conf.int=TRUE)$coefficients[2,1]
sd_4   = summary(reg4, conf.int=TRUE)$coefficients[2,2]

ci_1 = c(coef_1 - 2*sd_1 , coef_1 + 2*sd_1)
ci_2 = c(coef_2 - 2*sd_2 , coef_2 + 2*sd_2)
ci_3 = c(coef_3 - 2*sd_3 , coef_3 + 2*sd_3)
ci_4 = c(coef_4 - 2*sd_4 , coef_4 + 2*sd_4)

m = matrix(c(1,2,3,4,5,5),nrow = 3,ncol = 2,byrow = TRUE)

layout(mat = m,heights = c(0.4,0.4,0.2))

par(mar = c(2,2,1,1))
plot(subset_1[,fu.years], subset_1[,egfr], xlab='years', ylab='egfr', main='patient ID:3')
abline(reg1, col='red4')
abline(reg1_rob, col='blue4')
plot(subset_2[,fu.years], subset_2[,egfr], xlab='years', ylab='egfr', main='patient ID:37')
abline(reg2, col='red4')
abline(reg2_rob, col='blue4')
plot(subset_3[,fu.years], subset_3[,egfr], xlab='years', ylab='egfr', main='patient ID:162')
abline(reg3, col='red4')
abline(reg3_rob, col='blue4')
plot(subset_4[,fu.years], subset_4[,egfr], xlab='years', ylab='egfr', main='patient ID:223')
abline(reg4, col='red4')
abline(reg4_rob, col='blue4')
plot(1, type = "n", axes=FALSE, xlab="", ylab="")
plot_colors = c("blue","black", "green", "orange", "pink")
legend(x = "top",inset = 0,
        legend = c("Regression Line", "Robust Regression Line"),
        col=c('red4', 'blue4'), lwd=3, cex=1.5, horiz = TRUE)
```



```
cat('\npatient 3  , 95% confidence interval for slope:', round(ci_1,2))
```

```
##
## patient 3  , 95% confidence interval for slope: -2.52 11.62
```

```
cat('\npatient 37 , 95% confidence interval for slope:', round(ci_2,2))
```

```
##
## patient 37 , 95% confidence interval for slope: -3.29 2.07
```

```
cat('\npatient 162, 95% confidence interval for slope:', round(ci_3,2))
```

```
##
## patient 162, 95% confidence interval for slope: -9.11 -2.02
```

```
cat('\npatient 223, 95% confidence interval for slope:', round(ci_4,2))
```

```
##
## patient 223, 95% confidence interval for slope: -67.49 27.52
```

Write up: On the four diagrams, we can see the measurements of eGFR for patients with IDs, 3, 36 and 223 as a function of time. Also, we have estimated a linear relationship of those measurements plotted in Red colour. We can clearly see that for patients, 162 and 223, their eGFR index tends to fall over time. Our analysis goes a step further and also estimated a second line, the blue one, where we have excluded from each patients measurements their highest and lowest values. This allows us to understand a very important piece of information, that is, how sensitive our estimated linear line is to outliers For example, we can see on the bottom two diagrams that the relationship of eGFR as a function of time changes slightly when the outliers are removed, which means that we have confidence on our estimation. However, for patient 3, we can see a big change in the slope of the line. The red line indicates an increase of eGFR over time, while the blue one hints that the relationship might not be as dramatic as we think. Most interestingly, for patient 7, the relationship seems to change sign and from slightly postie be slightly negative. Those results indicate that we might need more observations for these patients to draw a safer conclusion.

patient 3 , 95% confidence interval for slope: -2.52 11.62 patient 37 , 95% confidence interval for slope: -3.29 2.07 patient 162, 95% confidence interval for slope: -9.11 -2.02 patient 223, 95% confidence interval for slope: -67.49 27.52

We have also calculated the 95% confidence intervals for the slope of the lines, which are presented above. This information is extremely important because it allows us to understand not only the slope of the curve, but also it's variability or 'how sure are we that the slope is what it is'. For example, we suspected from the diagrams that for patient 3 the slope could be zero, or non changing in time overall. This is confirmed from the 95% confidence intervals. Notice that in both cases, they include 0 in them. This means that in a significance level of 95% we **cannot** exclude zero as a possibility for the slope. In contrast, for patient 162, we are very confident in saying that the slope is negative over time. Lastly, a very interesting case is the case of patient 223. While from the diagram we can clearly see a negative linear relationship, our confidence interval suggests otherwise. This extreme big variance in the estimate for slope mainly comes from the fact that patient 223 has too few samples to base our analysis on. We need more measurements in order to be able to judge their situation better.

Problem 2 (25 points)

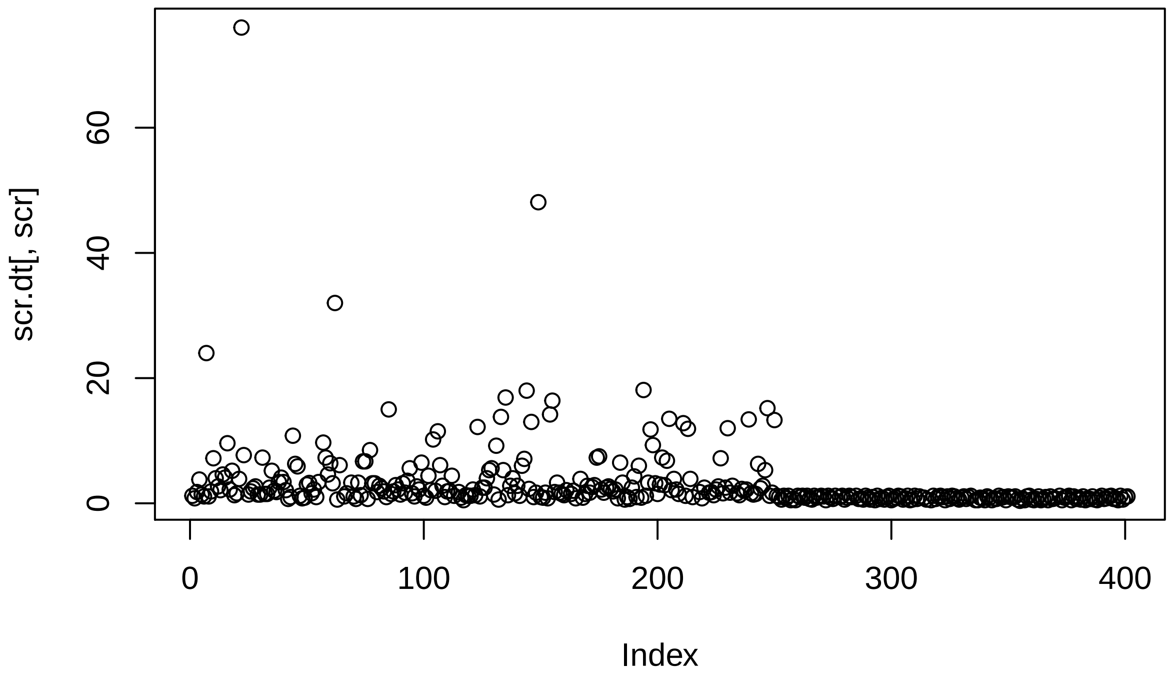
The MDRD4 and CKD-EPI equations are two different ways of estimating the glomerular filtration rate (eGFR) in adults: $MDRD4 = 175 \times Scr^{-1.154} \times Age^{-0.203}[\times 0.742 if female][\times 1.212 if black]$, and $CKD\_EPI = 141 \times \min(Scr/\kappa, 1)^{\alpha} \times \max(Scr/\kappa, 1)^{-1.209} \times 0.993^{Age}[\times 1.018 \;\; if \;\; female][\times 1.158 \;\; if \; black]$, (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

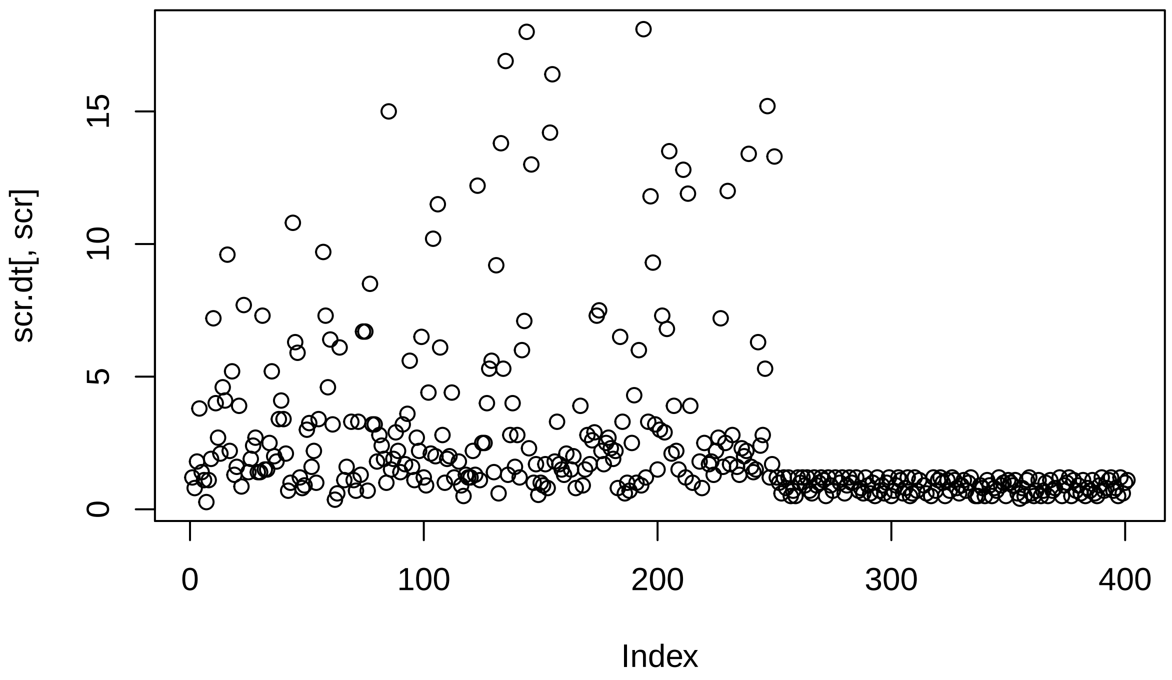
Problem 2.a (7 points)

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 micro-mol/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
# Enter code here.
scr.dt = data.table(fread('./data/scr.csv'))
plot(scr.dt[,scr])
```



```r
scr.dt[, scr:=ifelse(scr>19, scr/88.42, scr)]
plot(scr.dt[,scr])
```



```
IQR_scr = summary(scr.dt[,scr])[5] - summary(scr.dt[,scr])[2]
IQR_scr
```

```
## 3rd Qu.
##    1.85
```

Write up: We suspected that some of the measurements of serum creatinine might have been reported in the wrong units of measurements. This is clearly beeing shown in the plot above. We selected a cut off line on `scr=19`, so all measurements above that line where converted to micro-mol/L be dividing the observations with `88.42`. We are reporting the Inter Quantile range of the variable is reported and it is 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.

```
# Enter code here.
scr.dt[,scr_est1:=175*
         scr^(-1.154)*
         age^(-0.203)*
         (fifelse(scr.dt[,sex=='Female'],0.742,1, na=1))*
         (fifelse(scr.dt[,ethnic=='Black'],1.212,1,na=1))]

scr.dt[,min_scr_1:=pmin(scr/fifelse(scr.dt[,sex=='Female'],0.7,0.9,na=1),1, na.rm=TRUE)^(fifelse(scr.dt
scr.dt[,max_scr_1:=pmax(scr/fifelse(scr.dt[,sex=='Female'],0.7,0.9,na=1),1, na.rm=TRUE)^(-1.209)]
scr.dt[,scr_est2:=141*
         min_scr_1*
         max_scr_1*
         0.993^age*
         (fifelse(scr.dt[,sex=='Female'],1.018,1, na=1))*
         (fifelse(scr.dt[,ethnic=='Black'],1.158,1,na=1))]
mean_est_1 = round(mean(scr.dt$scr_est1, na.rm=TRUE),2)
sd_est_1   = round(sd  (scr.dt$scr_est1, na.rm=TRUE),2)
```

```
mean_est_2 = round(mean(scr.dt$scr_est2, na.rm=TRUE),2)
sd_est_2   = round(sd  (scr.dt$scr_est2, na.rm=TRUE),2)

cat('\nMean of eGFR according to MDRD4',mean_est_1)
```

```
##
## Mean of eGFR according to MDRD4 61.11
```

```
cat('\nStandard deviations of eGFR according to MDRD4',sd_est_1)
```

```
##
## Standard deviations of eGFR according to MDRD4 49.82
```

```
cat('\nMean of eGFR according to CKD-EPI',mean_est_2)
```

```
##
## Mean of eGFR according to CKD-EPI 61.77
```

```
cat('\nStandard deviations of eGFR according to CKD-EPI',sd_est_2)
```

```
##
## Standard deviations of eGFR according to CKD-EPI 42.44
```

```
cat('\nLinear correlation of the two estimates of eGFR',cor(scr.dt$scr_est1,scr.dt$scr_est2, method='pe
```

```
##
## Linear correlation of the two estimates of eGFR 0.9531475
```

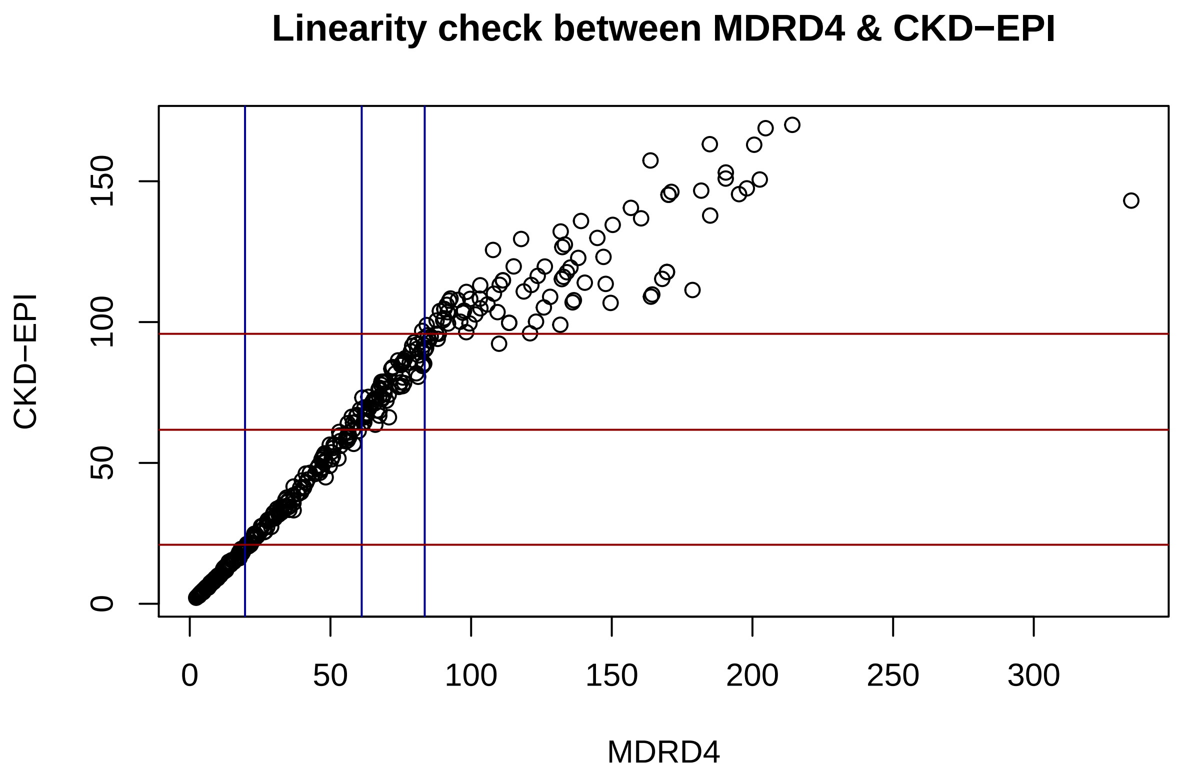
Report (rounded to the second decimal place) mean and standard deviation of the two eGFR vectors and their Pearson correlation coefficient. Also report the same quantities according to strata of MDRD4 eGFR: 0-60, 60-90 and > 90.

Write up: We can clearly see from the output of the code above, that the two estimates are very similar in their mean and deviation. The linear correlation estimate of 95.3% is also a very good indicator of that.

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 quartiles. Is the relationship between the two eGFR equations linear? Justify your answer.

```
# Enter code here.
plot(scr.dt$scr_est1,scr.dt$scr_est2, xlab='MDRD4', ylab='CKD-EPI', main='Linearity check between MDRD4
abline(v=mean_est_1, col='blue4')
abline(v=quantile(scr.dt$scr_est1, na.rm=TRUE)[2], col='blue4')
abline(v=quantile(scr.dt$scr_est1, na.rm=TRUE)[4], col='blue4')
abline(h=mean_est_2, col='red4')
abline(h=quantile(scr.dt$scr_est2, na.rm=TRUE)[2], col='red4')
abline(h=quantile(scr.dt$scr_est2, na.rm=TRUE)[4], col='red4')
```


Linearity check between MDRD4 & CKD−EPI
CKD−EPI
MDRD4
0
50
100
150
200
250
300

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".

```r
# Enter code here.
cohort = data.table(fread('./data/cohort.csv', stringsAsFactors = TRUE))
lab1 = data.table(fread('./data/lab1.csv'))
linker = data.table(fread('./data/linker.csv'))
tmp = merge(cohort, linker, by=c('id'), all=TRUE)
cohort.dt = merge(tmp, lab1, by.x=c('LABID', 'yob'), by.y=c('LABID', 'yob'), all=TRUE)
cohort.dt$albumin = ordered(cohort.dt$albumin,
                            levels = c('normo','micro','macro'))
cohort.dt
```

```
##        LABID  yob      id age  bp diabetes albumin urea creatinine glucose
##   1:   LID_1 1986 PID_285  33  80        0   normo 37.0    106.104     100
##   2:  LID_10 1980 PID_153  39  70        1   normo 20.0     70.736     121
##   3: LID_100 1951  PID_13  68  70        1   micro 72.0    185.682     208
##   4: LID_101 1965 PID_110  54  70        1    <NA> 50.1    167.998     233
##   5: LID_102 1953 PID_222  66  70        1   micro 30.0    150.314     248
##  ---
## 396:  LID_95 1962 PID_254  57  80        0   normo 17.0    106.104     119
## 397:  LID_96 1978 PID_297  41  70        0   normo 38.0     53.052     125
## 398:  LID_97 1964 PID_119  55  70        0   micro 25.0    106.104      99
## 399:  LID_98 1974 PID_236  45  70        0   micro 93.0    203.366     113
## 400:  LID_99 1963 PID_100  56 180        1   normo 24.0    106.104     298
```

Problem 3.b (10 points)

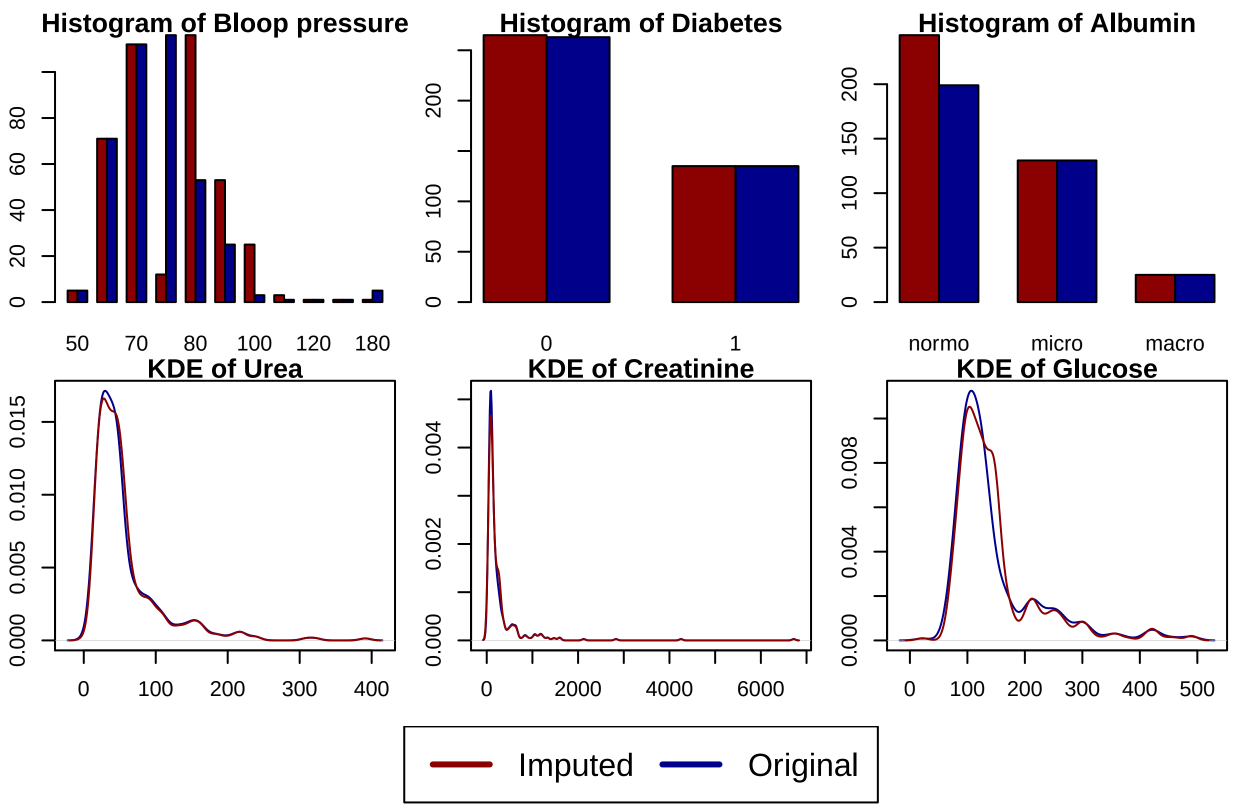
Create a copy of the dataset where you will impute all missing values. Update any missing age fields using the year of birth, for all other continuous variables write a function called impute.to.mean and impute to mean, impute any categorical variable to the mode. Compare the distributions of the imputed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

```r
# Enter code here.
cohort_complete = copy(cohort.dt)
cohort_complete[is.na(age), age:=abs(year(Sys.Date())-round(yob))]
cohort_complete
```

```
##        LABID  yob      id age  bp diabetes albumin urea creatinine glucose
##   1:   LID_1 1986 PID_285  33  80        0   normo 37.0    106.104     100
##   2:  LID_10 1980 PID_153  39  70        1   normo 20.0     70.736     121
##   3: LID_100 1951  PID_13  68  70        1   micro 72.0    185.682     208
##   4: LID_101 1965 PID_110  54  70        1    <NA> 50.1    167.998     233
##   5: LID_102 1953 PID_222  66  70        1   micro 30.0    150.314     248
##  ---
## 396:  LID_95 1962 PID_254  57  80        0   normo 17.0    106.104     119
## 397:  LID_96 1978 PID_297  41  70        0   normo 38.0     53.052     125
## 398:  LID_97 1964 PID_119  55  70        0   micro 25.0    106.104      99
## 399:  LID_98 1974 PID_236  45  70        0   micro 93.0    203.366     113
## 400:  LID_99 1963 PID_100  56 180        1   normo 24.0    106.104     298
```

```r
#' This is a function that takes as impute a column of a data.table and imputes
#' the NAs with its mean / mode if the vector is numeric or categorical respectively.
#' @param x A vector of numeric or categorical values for which the NAs will be imputed.
impute.to.mean = function(x) {
    if (is.numeric(x)){
      if (all(na.omit(x) %in% 0:1)){
        x[is.na(x)] = unique(x)[which.max(tabulate(match(x, unique(x))))]
      } else {x[is.na(x)] = mean(x, na.rm=TRUE)}
    } else if (is.factor(x)){x[is.na(x)] = unique(x)[which.max(tabulate(match(x, unique(x))))]}
    return(x)
  }
```

```r
numcols = cohort_complete %>% select(bp,diabetes,urea,creatinine,glucose, albumin) %>% colnames
cohort_complete %>% .[, (numcols) := lapply(.SD, impute.to.mean), .SDcols = numcols]
m = matrix(c(1,2,3,4,5,6,7,7,7),nrow = 3,ncol = 3,byrow = TRUE)
layout(mat = m,heights = c(0.4,0.4,0.2))
par(mar = c(2,2,1,1))
barplot(rbind(table(cohort_complete$bp), table(cohort.dt$bp)), beside=TRUE, col=c('red4','blue4'), main=
barplot(rbind(table(cohort_complete$diabetes), table(cohort.dt$diabetes)), beside=TRUE, col=c('red4','bl
barplot(rbind(table(cohort_complete$albumin), table(cohort.dt$albumin)), beside=TRUE, col=c('red4','blu
plot(density(cohort.dt$urea, na.rm=TRUE), col='blue4', main='KDE of Urea')
lines(density(cohort_complete$urea), col='red4')
plot(density(cohort.dt$creatinine, na.rm=TRUE), col='blue4', main='KDE of Creatinine')
lines(density(cohort_complete$creatinine), col='red4')
plot(density(cohort.dt$glucose, na.rm=TRUE), col='blue4', main='KDE of Glucose')
lines(density(cohort_complete$glucose), col='red4')
plot(1, type = "n", axes=FALSE, xlab="", ylab="")
legend(x = "top",inset = 0,
        legend = c("Imputed", "Original"),
        col=c('red4', 'blue4'), lwd=3, cex=1.5, horiz = TRUE)
```

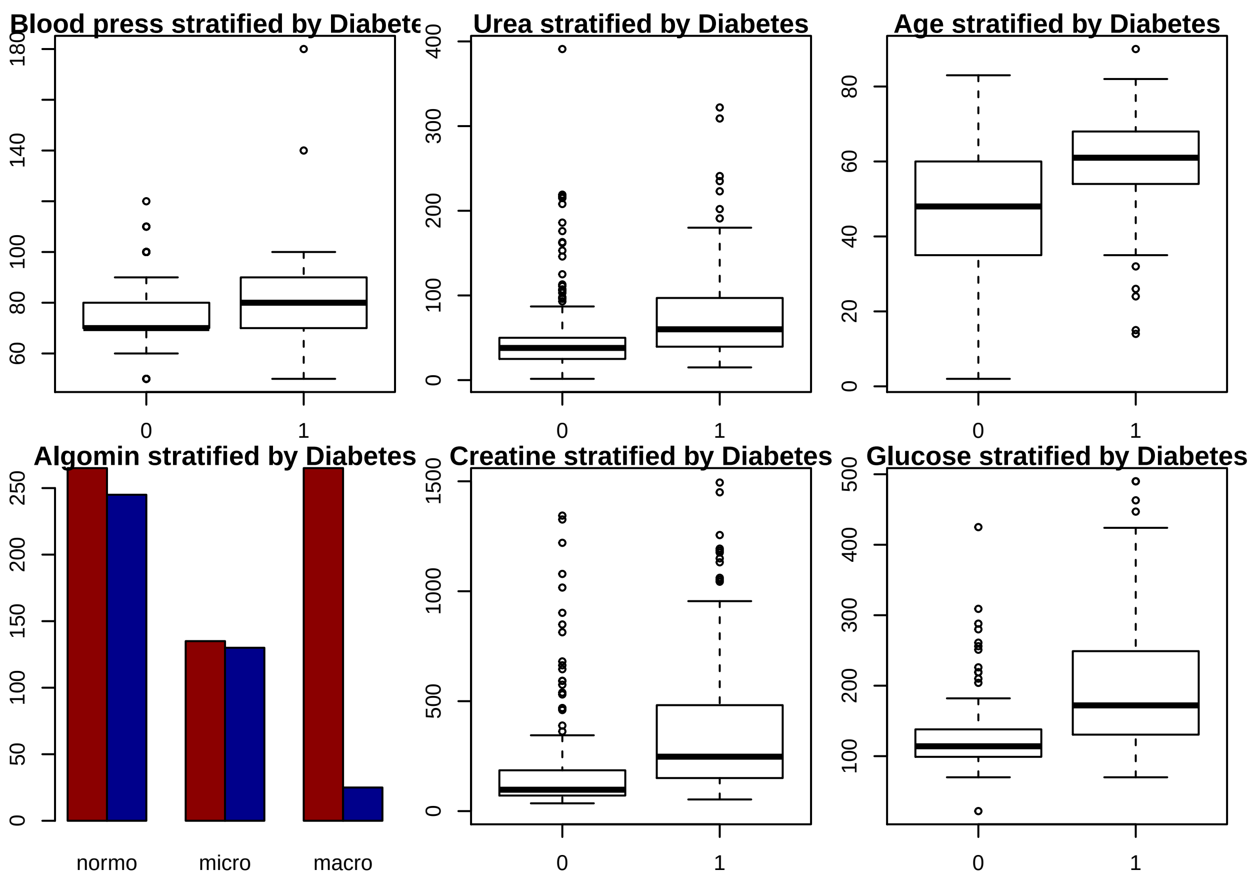


The diagrams above show the distribution of our variables with (red) and without (blue) the imputation step. This step is important because we need to understand whether we introduced some kind of bias to the dataset when imputing the values. The bias would be visible if the red lines / bars where vastly different from the blue ones. For example, if we focus on the Urea diagram, we can clearly see that the distribution after the imputation of the missing values follows very closely the original distribution. If wee shift our attention to the Glucose variable we can see that our imputation while still doing a reasonable imputation, the red line doesn't follow the blue one that closely. A very important variable, the variable of Diabetes, is our primary focus though, since this is our target variable (the one we are trying to estimate), and we wouldn't want to skew our data in anyway. We can clearly see from the diagram that the distributions before an after the imputation are almost identical.

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.

```
# Enter code here.
bp = cohort_complete$bp
urea = cohort_complete$urea
creatinine = cohort_complete$creatinine
glucose = cohort_complete$glucose
albumin = cohort_complete$albumin
diabetes = cohort_complete$diabetes

m = matrix(c(1,2,3,4,5,6),nrow = 2,ncol = 3,byrow = TRUE)
layout(mat = m,heights = c(0.4,0.4,0.2))
par(mar = c(2,2,1,1))
boxplot(bp ~ diabetes, data=cohort_complete, main="Blood press stratified by Diabetes")
boxplot(urea ~ diabetes, data=cohort_complete, main="Urea stratified by Diabetes")
boxplot(age ~ diabetes, data=cohort_complete, main="Age stratified by Diabetes")
barplot(rbind(table(cohort_complete$diabetes), table(cohort_complete$albumin)), beside=TRUE, col=c('red4
boxplot(creatinine ~ diabetes, data=cohort_complete, main="Creatine stratified by Diabetes", ylim=c(0,15
boxplot(glucose ~ diabetes, data=cohort_complete, main="Glucose stratified by Diabetes")
```



```
albumin_table = table(albumin, diabetes)
colnames(albumin_table) = c("No Diabetes", "Diabetes")
albumin_table
```

```
##        diabetes
## albumin No Diabetes Diabetes
##   normo         192       53
##   micro          61       69
##   macro          12       13
```

The above boxplots are plotted in order to help us decide which variables can potentially be important in building a model to predict and understand diabetes. We have stratified each variable by the diabetes levels (0: No diabetes present in subject, 1: Diabetes present in subject) and we are essentially looking to see what variables have the most different distributions between the strata. What we take from this diagram is that the Glucose variable plays a key role in predicting diabetes, which is what the literature sujest and we know to date. Secondly, we see that the Algomin variable plays a very important role, when it is in 'macro' levels, so this could also be a good predictor. With that in mind, we will quantify the relation be fitting and comparing many different models.

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.

```r
# Enter code here.
model1 = glm(diabetes ~ glucose + albumin, family='binomial')
summary(model1)
```

```
##
## Call:
## glm(formula = diabetes ~ glucose + albumin, family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0990  -0.7013  -0.5208   0.6802   2.2565
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.151621   0.416888  -7.560 4.03e-14 ***
## glucose      0.017834   0.002513   7.095 1.29e-12 ***
## albumin.L    0.534820   0.342585   1.561   0.1185
## albumin.Q   -0.448139   0.256400  -1.748   0.0805 .
## ---
## 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: 389.59  on 396  degrees of freedom
## AIC: 397.59
##
## Number of Fisher Scoring iterations: 5
```

The above results comes from a very simple linear model that uses the variables glucose and albumin to predict diabetes in a patient. The output of the model is a number from zero to one, which we can convert to binary (0 or 1) on some threshold. The most important numbers are the ones on the coefficient section. These numbers are the coefficient of each variable and our model can be summarized as:

$\text{diabetes} = -3.151 + 0.018 \times \text{glucose} + 0.535[\text{if albumin is normo}]$

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.

```
# Enter code here.
model2 = glm(diabetes ~ glucose + albumin + cohort_complete$age, family='binomial')
summary(model2)
```

```
##
## Call:
## glm(formula = diabetes ~ glucose + albumin + cohort_complete$age,
##     family = "binomial")
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7615  -0.7072  -0.4022   0.6420   2.5238
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -5.451612   0.681802  -7.996 1.29e-15 ***
## glucose              0.016281   0.002538   6.416 1.40e-10 ***
## albumin.L            0.642553   0.373277   1.721   0.0852 .
## albumin.Q           -0.247191   0.277030  -0.892   0.3722
## cohort_complete$age  0.047137   0.009543   4.940 7.83e-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: 359.43  on 395  degrees of freedom
## AIC: 369.43
##
## Number of Fisher Scoring iterations: 5
```

```
pval = pchisq(model1$deviance - model2$deviance, df=1, lower.tail=FALSE)
signif(pval, 2)
```

```
## [1] 4e-08
```

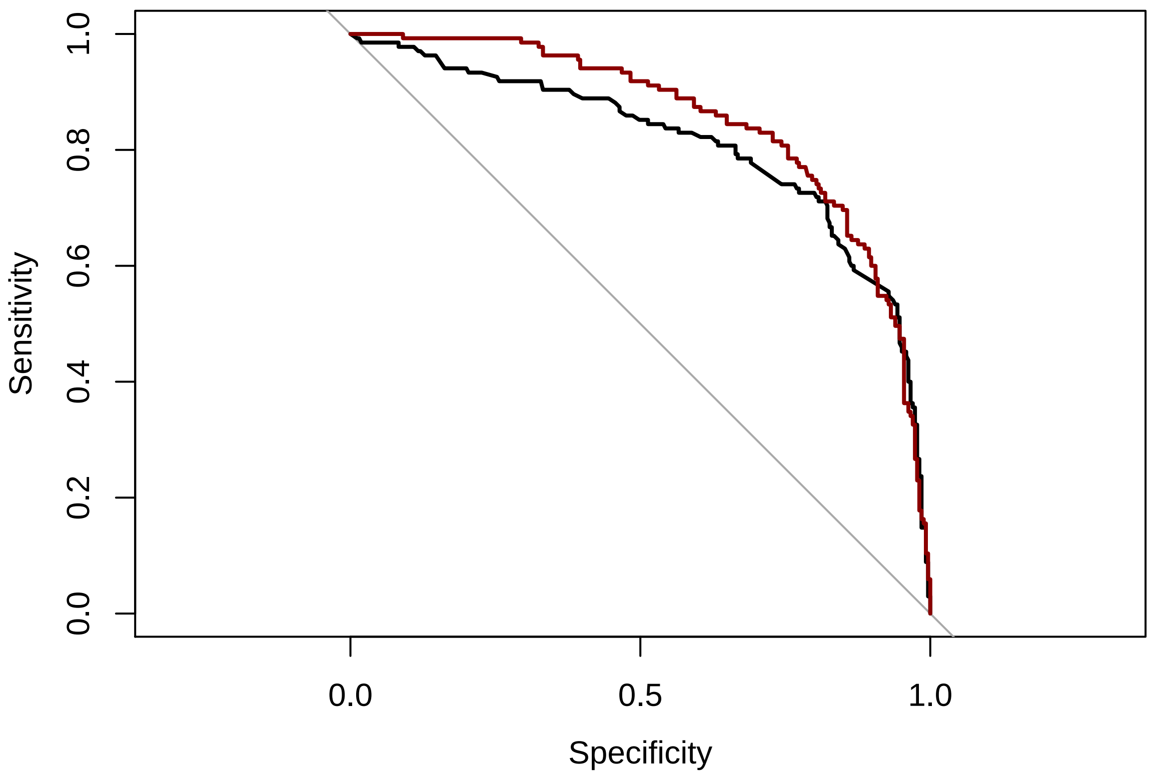
```
roc1 = roc(cohort_complete$diabetes, model1$fitted.values , plot=TRUE, xlim = c(0,1))
```

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

```
## Setting direction: controls < cases
```

```
roc2 = roc(cohort_complete$diabetes, model2$fitted.values , plot=TRUE, xlim = c(0,1), add=TRUE, col="re
```

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



```
roc1$auc
```

```
## Area under the curve: 0.8089
```

```
roc2$auc
```

```
## Area under the curve: 0.8522
```

Problem 4.b (10 points)

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

```
# Enter code here.
set.seed(1903)
k = 10
folds = createFolds(cohort_complete$diabetes, k=k)

pred.cv <- NULL
regr.cv <- NULL
auc.cv <- numeric(k)
for(f in 1:k) {
  train.idx <- setdiff(1:nrow(cohort_complete), folds[[f]])
  regr.cv[[f]] <- glm(diabetes ~ glucose + albumin + cohort_complete$age, family='binomial')
  test.idx <- folds[[f]]
  pred.cv[[f]] <- data.frame(obs = cohort_complete$diabetes[test.idx],
                             pred = predict(regr.cv[[f]], newdata = cohort_complete, type = "response"))
  auc.cv[f] <- roc(obs ~ pred, data = pred.cv[[f]])$auc
}

round(mean(auc.cv), 3)
```

```
## [1] 0.843
```