AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

MID-TERM PROJECT

Course: INTRODUCTION TO DATA SCIENCE

Sec: A

SUBMITTED TO

Faculty: Tohedul Islam

SUBMITTED BY

Group: 15

NAME	ID
1. Abdullah Muhammad Hamja	20 -43465-1
2. Sumaiya Ahmed Susmita	21-45266-2

Heart Disease Classification Dataset

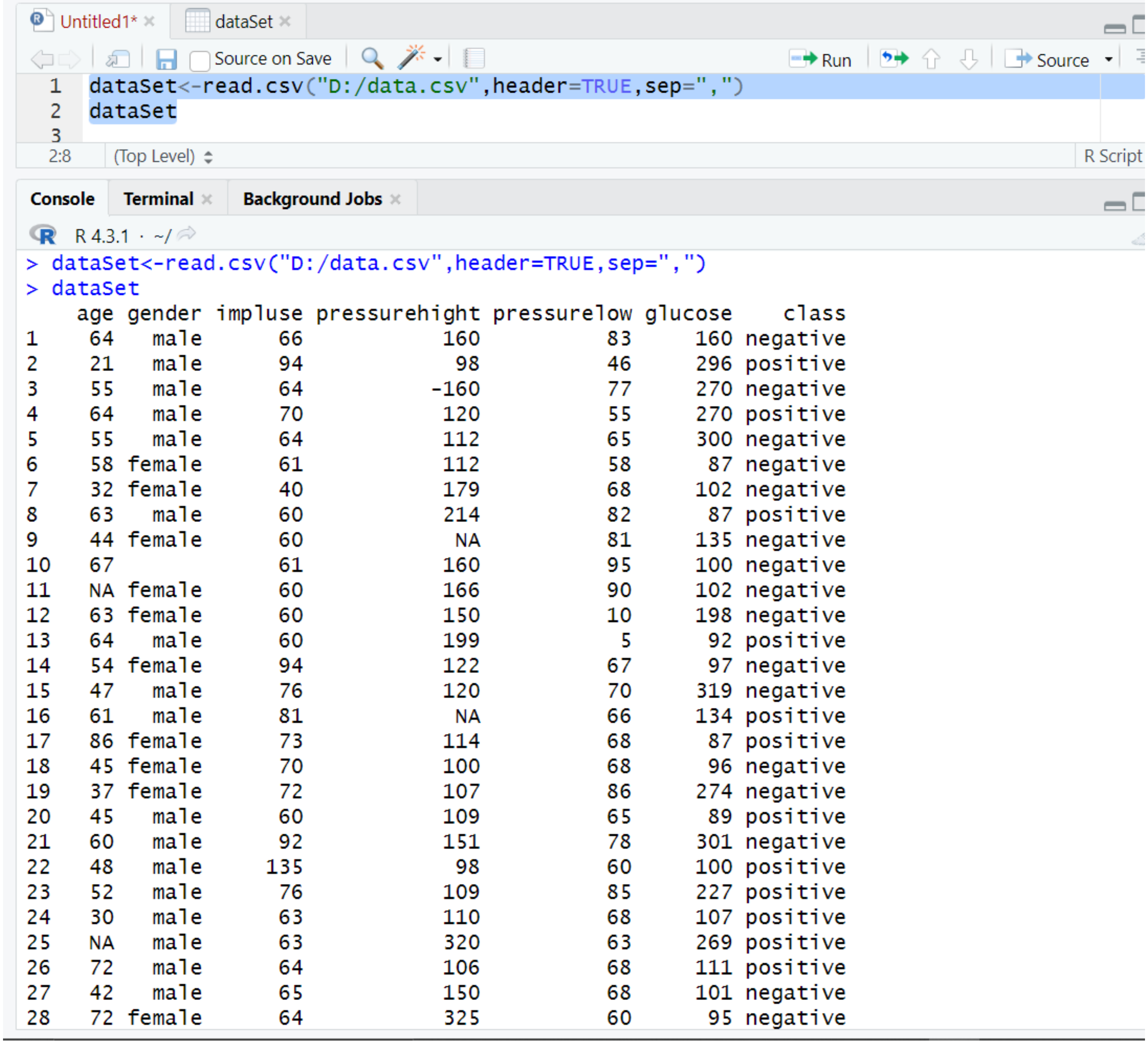
Cardiovascular diseases (CVDs) stand as a leading global cause of mortality, encompassing conditions like coronary heart disease, cerebrovascular disease, and more. With 17.9 million annual deaths attributed to CVDs, understanding and predicting factors contributing to heart attacks are crucial for public health. This dataset comprises 1319 samples, each characterized by eight input fields—age, gender, heart rate, systolic and diastolic blood pressure, blood sugar level, CK-MB, and Test-Troponin. The ninth field, 'class,' serves as the output, indicating the presence (positive) or absence (negative) of a heart attack. The dataset serves as a valuable resource for developing models to predict and understand the risk factors associated with cardiovascular events, contributing to advancements in preventive healthcare strategies.

Import CSV file in RStudio:

Code:

dataSet<-read.csv("D:/data.csv",header=TRUE,sep=",")

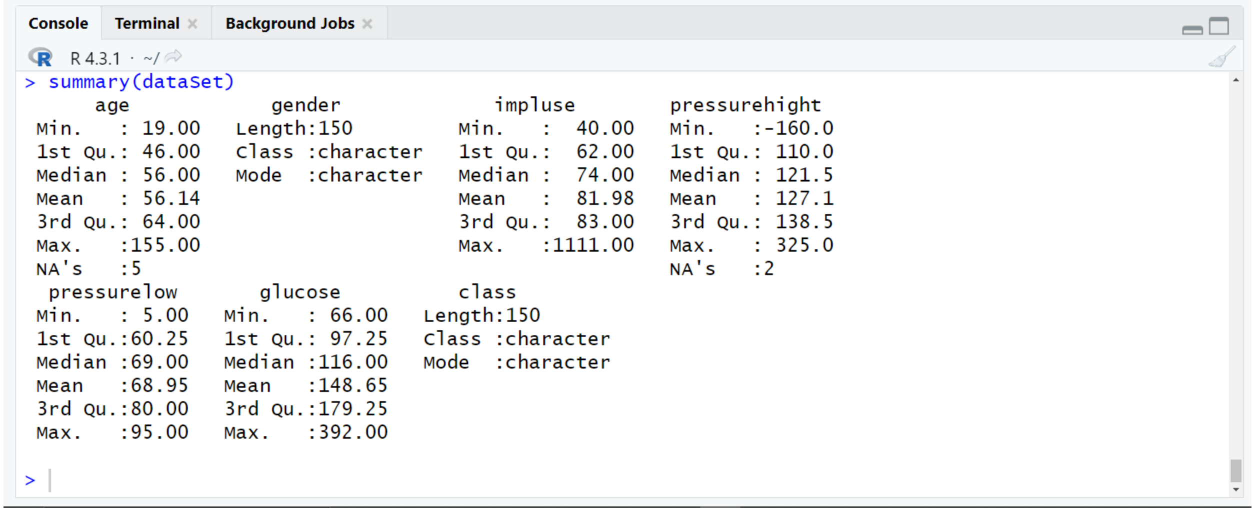
Dataset



```
dataSet<-read.csv("D:/data.csv",header=TRUE,sep=",")
dataSet
```

```
> dataSet<-read.csv("D:/data.csv",header=TRUE,sep=",")
> dataSet
   age gender impluse pressurehight pressurelow glucose    class
1   64   male      66           160          83     160 negative
2   21   male      94            98          46     296 positive
3   55   male      64          -160          77     270 negative
4   64   male      70           120          55     270 positive
5   55   male      64           112          65     300 negative
6   58 female      61           112          58      87 negative
7   32 female      40           179          68     102 negative
8   63   male      60           214          82      87 positive
9   44 female      60            NA          81     135 negative
10  67             61           160          95     100 negative
11  NA female      60           166          90     102 negative
12  63 female      60           150          10     198 negative
13  64   male      60           199           5      92 positive
14  54 female      94           122          67      97 negative
15  47   male      76           120          70     319 negative
16  61   male      81            NA          66     134 positive
17  86 female      73           114          68      87 positive
18  45 female      70           100          68      96 negative
19  37 female      72           107          86     274 negative
20  45   male      60           109          65      89 positive
21  60   male      92           151          78     301 negative
22  48   male     135            98          60     100 positive
23  52   male      76           109          85     227 positive
24  30   male      63           110          68     107 positive
25  NA   male      63           320          63     269 positive
26  72   male      64           106          68     111 positive
27  42   male      65           150          68     101 negative
28  72 female      64           325          60      95 negative
```

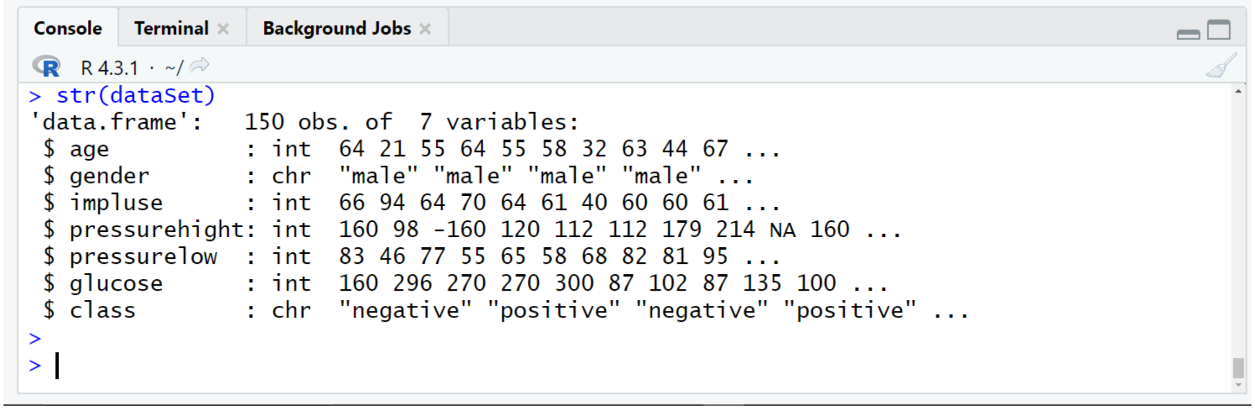
Summary of the dataset: It’s used to show a summary of the dataset.
Code: summary(dataSet)



```
> summary(dataSet)
      age             gender             impluse         pressurehight
 Min.   : 19.00   Length:150         Min.   :  40.00   Min.   :-160.0
 1st Qu.: 46.00   Class :character   1st Qu.:  62.00   1st Qu.: 110.0
 Median : 56.00   Mode  :character   Median :  74.00   Median : 121.5
 Mean   : 56.14                      Mean   :  81.98   Mean   : 127.1
 3rd Qu.: 64.00                      3rd Qu.:  83.00   3rd Qu.: 138.5
 Max.   :155.00                      Max.   :1111.00   Max.   : 325.0
 NA's   :5                                             NA's   :2
  pressurelow       glucose          class
 Min.   : 5.00   Min.   : 66.00   Length:150
 1st Qu.:60.25   1st Qu.: 97.25   Class :character
 Median :69.00   Median :116.00   Mode  :character
 Mean   :68.95   Mean   :148.65
 3rd Qu.:80.00   3rd Qu.:179.25
 Max.   :95.00   Max.   :392.00

> |
```

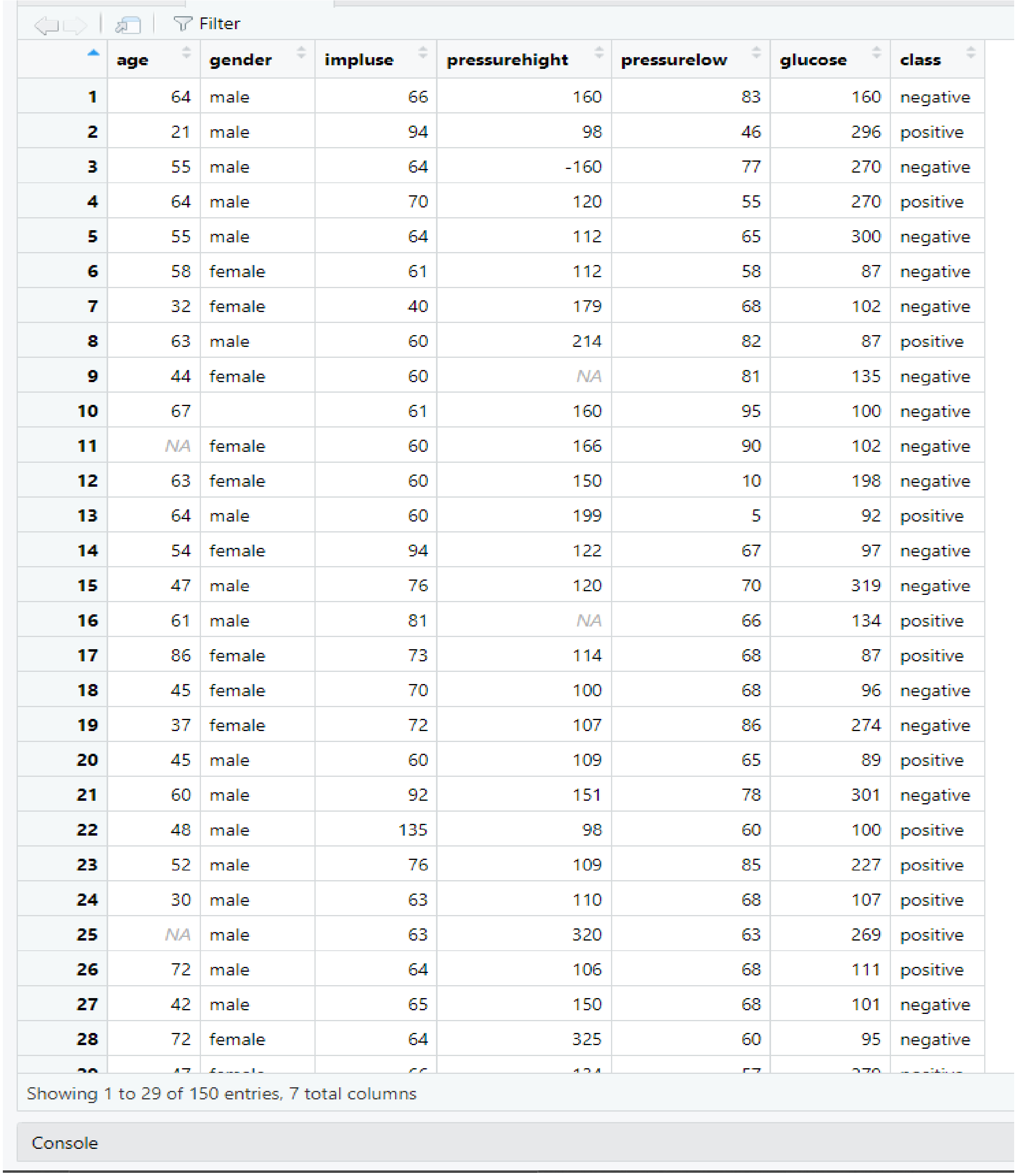
Structure of the dataset: we can use str function to display the whole structure of the dataset.

Code: str(dataSet)



```
> str(dataSet)
'data.frame':	150 obs. of  7 variables:
 $ age          : int  64 21 55 64 55 58 32 63 44 67 ...
 $ gender       : chr  "male" "male" "male" "male" ...
 $ impluse      : int  66 94 64 70 64 61 40 60 60 61 ...
 $ pressurehight: int  160 98 -160 120 112 112 179 214 NA 160 ...
 $ pressurelow  : int  83 46 77 55 65 58 68 82 81 95 ...
 $ glucose      : int  160 296 270 270 300 87 102 87 135 100 ...
 $ class        : chr  "negative" "positive" "negative" "positive" ...
>
> |
```

Displaying Dataset: Ctrl + left-click (mouse)



	age	gender	impluse	pressurehight	pressurelow	glucose	class
1	64	male	66	160	83	160	negative
2	21	male	94	98	46	296	positive
3	55	male	64	-160	77	270	negative
4	64	male	70	120	55	270	positive
5	55	male	64	112	65	300	negative
6	58	female	61	112	58	87	negative
7	32	female	40	179	68	102	negative
8	63	male	60	214	82	87	positive
9	44	female	60	NA	81	135	negative
10	67		61	160	95	100	negative
11	NA	female	60	166	90	102	negative
12	63	female	60	150	10	198	negative
13	64	male	60	199	5	92	positive
14	54	female	94	122	67	97	negative
15	47	male	76	120	70	319	negative
16	61	male	81	NA	66	134	positive
17	86	female	73	114	68	87	positive
18	45	female	70	100	68	96	negative
19	37	female	72	107	86	274	negative
20	45	male	60	109	65	89	positive
21	60	male	92	151	78	301	negative
22	48	male	135	98	60	100	positive
23	52	male	76	109	85	227	positive
24	30	male	63	110	68	107	positive
25	NA	male	63	320	63	269	positive
26	72	male	64	106	68	111	positive
27	42	male	65	150	68	101	negative
28	72	female	64	325	60	95	negative
29	47	female	66	134	57	270	positive

Showing 1 to 29 of 150 entries, 7 total columns

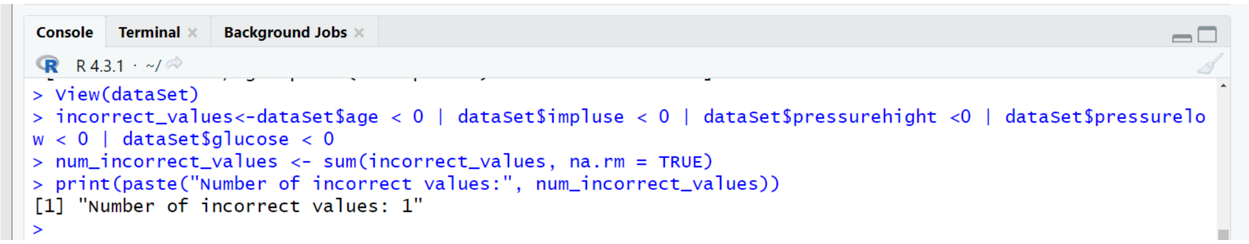
Data Quality

1. **Data Correctness**: Checks if there’s any incorrect value.
- Then summing up the occurrences of incorrect values.
- Printing the number of incorrect values.

Code:
incorrect_values<-dataSet$age < 0 | dataSet$impluse < 0 | dataSet$pressurehight <0 | dataSet$pressurelow < 0 | dataSet$glucose < 0

num_incorrect_values <- sum(incorrect_values, na.rm = TRUE)

print(paste("Number of incorrect values:", num_incorrect_values))

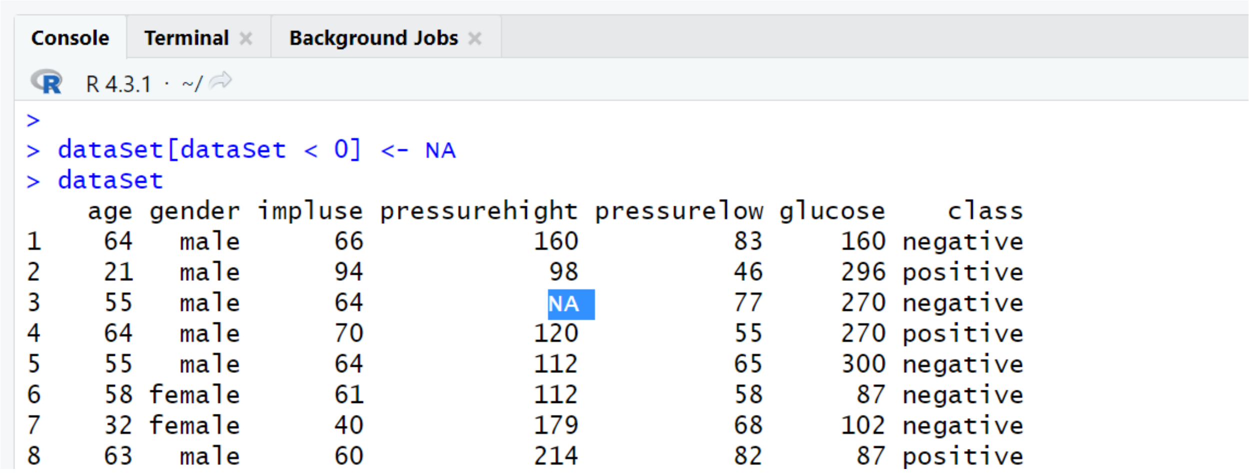


```
> View(dataSet)
> incorrect_values<-dataSet$age < 0 | dataSet$impluse < 0 | dataSet$pressurehight <0 | dataSet$pressurelo
w < 0 | dataSet$glucose < 0
> num_incorrect_values <- sum(incorrect_values, na.rm = TRUE)
> print(paste("Number of incorrect values:", num_incorrect_values))
[1] "Number of incorrect values: 1"
>
```

(**Invalid Value**)

Then, Replacing all negative/ incorrect values with ‘NA’
Code:
dataSet[dataSet < 0] <- NA

dataSet

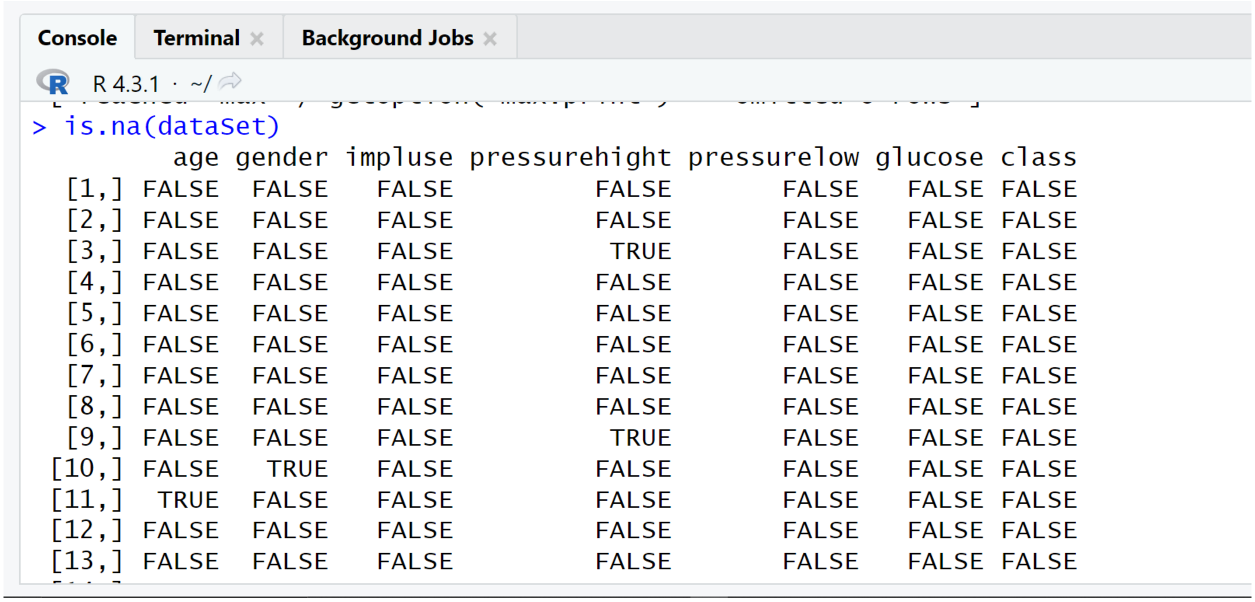


```
>
> dataSet[dataSet < 0] <- NA
> dataSet
    age gender impluse pressurehight pressurelow glucose    class
1    64   male      66           160          83     160 negative
2    21   male      94            98          46     296 positive
3    55   male      64            NA          77     270 negative
4    64   male      70           120          55     270 positive
5    55   male      64           112          65     300 negative
6    58 female      61           112          58      87 negative
7    32 female      40           179          68     102 negative
8    63   male      60           214          82      87 positive
```

2. **Data Completeness**: Checks missing values in dataset.
Code:

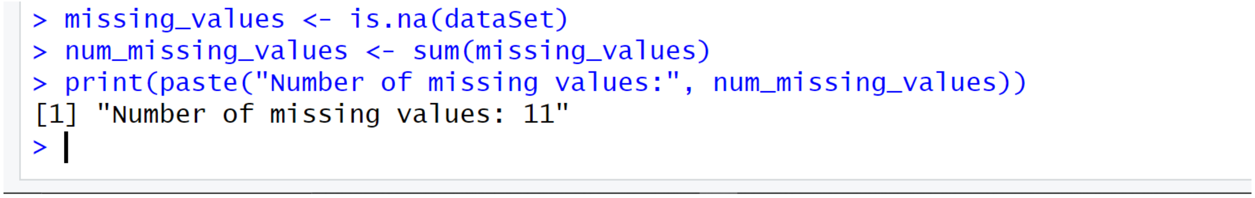
dataSet

is.na(dataSet)



```
> is.na(dataSet)
         age gender impluse pressurehight pressurelow glucose class
  [1,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [2,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [3,] FALSE  FALSE   FALSE          TRUE       FALSE   FALSE FALSE
  [4,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [5,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [6,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [7,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [8,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
  [9,] FALSE  FALSE   FALSE          TRUE       FALSE   FALSE FALSE
 [10,] FALSE   TRUE   FALSE         FALSE       FALSE   FALSE FALSE
 [11,]  TRUE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
 [12,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
 [13,] FALSE  FALSE   FALSE         FALSE       FALSE   FALSE FALSE
```

Then,
Checking and Summing the total number of missing values.



```
> missing_values <- is.na(dataSet)
> num_missing_values <- sum(missing_values)
> print(paste("Number of missing values:", num_missing_values))
[1] "Number of missing values: 11"
> 
```

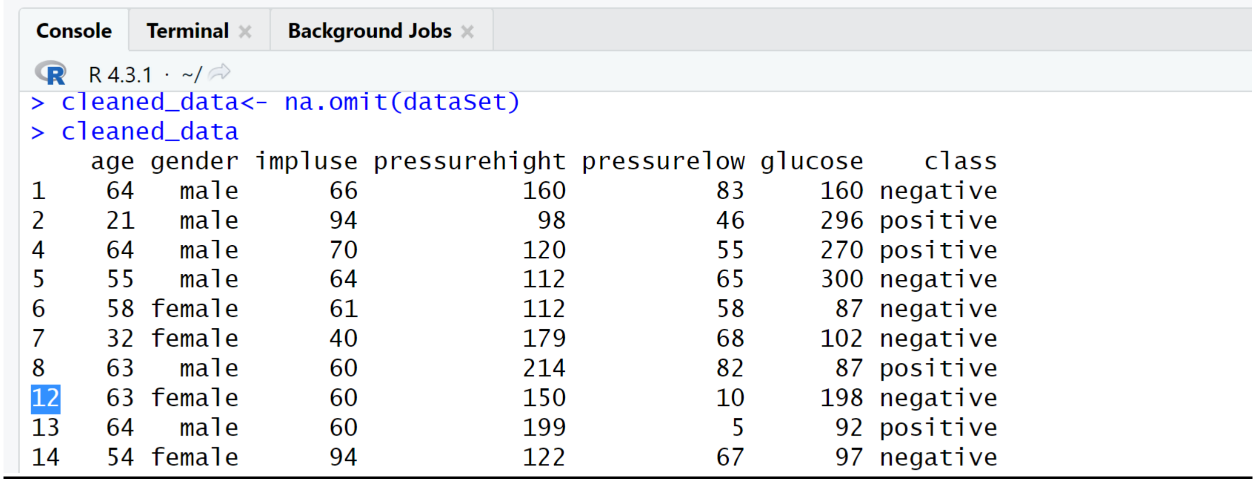
Missing Value

Discard Instances: Remove row with any missing values.

Code:

cleaned_data<- na.omit(dataSet)

cleaned_data



```
> cleaned_data<- na.omit(dataSet)
> cleaned_data
   age gender impluse pressurehight pressurelow glucose    class
1   64   male      66           160          83     160 negative
2   21   male      94            98          46     296 positive
4   64   male      70           120          55     270 positive
5   55   male      64           112          65     300 negative
6   58 female      61           112          58      87 negative
7   32 female      40           179          68     102 negative
8   63   male      60           214          82      87 positive
12  63 female      60           150          10     198 negative
13  64   male      60           199           5      92 positive
14  54 female      94           122          67      97 negative
```

Replace by Most Frequent/Average Value: This method generally refers to imputing missing values in a dataset with either 'the most frequent value (mode) for categorical data' or 'the average value (mean) for numerical data'.

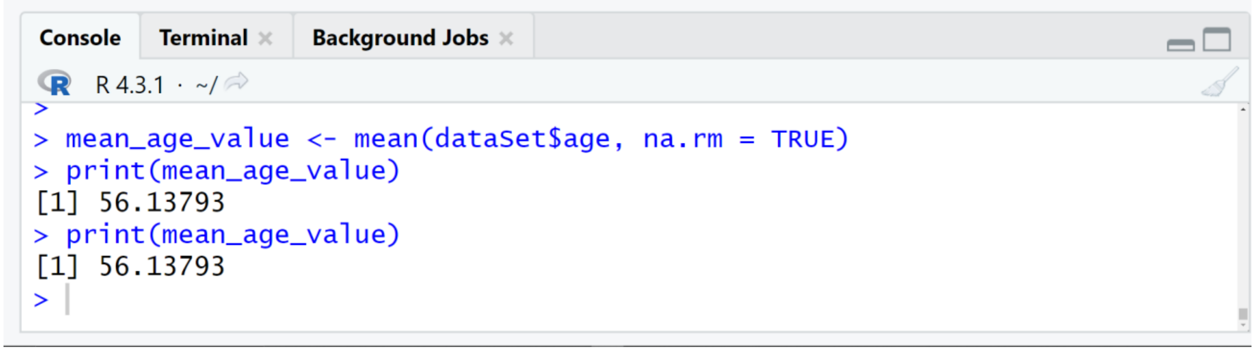
So, we have findout mean values for every column. (age, gender, impulse, pressurehight, pressurelow, glucose, class)

1. MEAN VALUE OF AGE: This code is used to calculate the mean value of the age.

Code:

mean_age_value <- mean(dataSet$age, na.rm = TRUE)

print(mean_age_value)



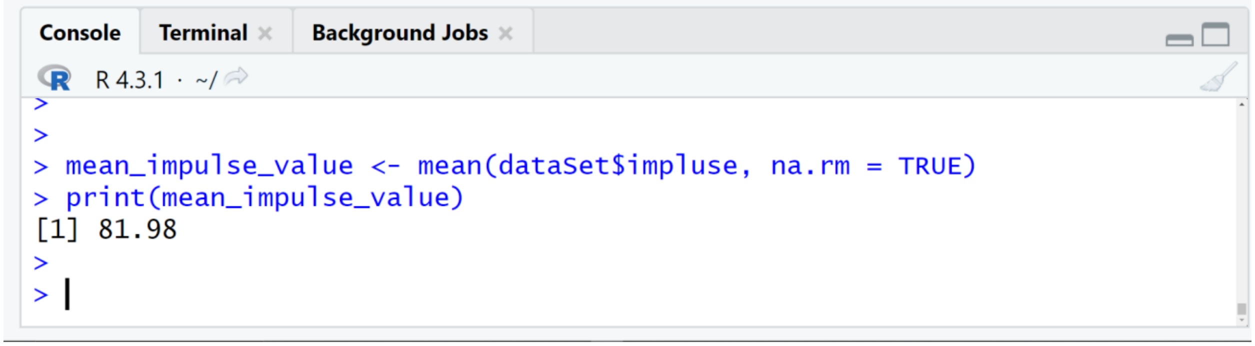
```
>
> mean_age_value <- mean(dataSet$age, na.rm = TRUE)
> print(mean_age_value)
[1] 56.13793
> print(mean_age_value)
[1] 56.13793
> 
```

2. MEAN VALUE OF IMPULSE: This code is used to calculate the mean value of the impulse.

Code:

mean_impulse_value <- mean(dataSet$impluse, na.rm = TRUE)

print(mean_impulse_value)



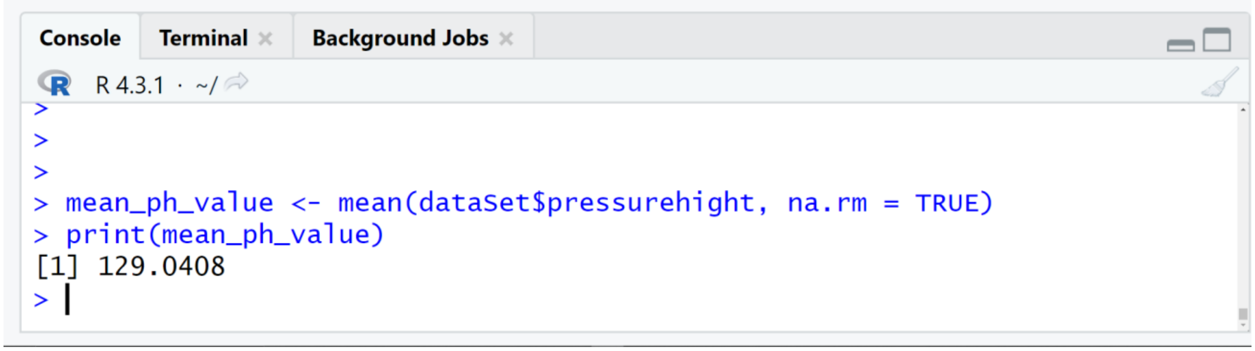
```
>
>
> mean_impulse_value <- mean(dataSet$impluse, na.rm = TRUE)
> print(mean_impulse_value)
[1] 81.98
>
> 
```

3. MEAN VALUE OF PRESSURE HIGHT: This code is used to calculate the mean value of the Pressureheight.

Code:

mean_ph_value <- mean(dataSet$pressurehight, na.rm = TRUE)

print(mean_ph_value)



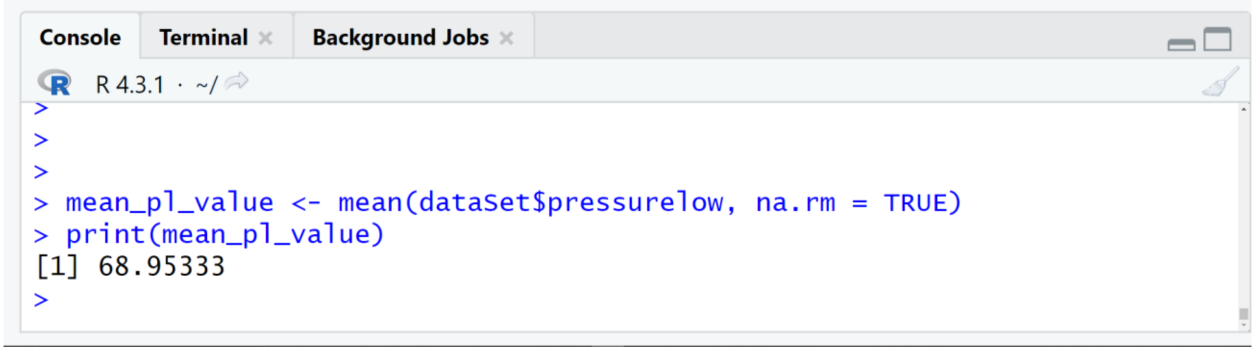
```
>
>
>
> mean_ph_value <- mean(dataSet$pressurehight, na.rm = TRUE)
> print(mean_ph_value)
[1] 129.0408
> 
```

4. **MEAN VALUE OF PRESSURE LOW:** This code is used to calculate the mean value of the pressurelow.

Code:

mean_pl_value <- mean(dataSet$pressurelow, na.rm = TRUE)

print(mean_pl_value)

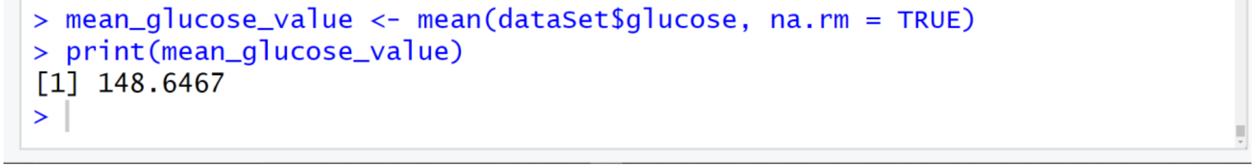


```
>
>
>
> mean_pl_value <- mean(dataSet$pressurelow, na.rm = TRUE)
> print(mean_pl_value)
[1] 68.95333
>
```

5. MEAN VALUE OF GLUCOSE: This code is used to calculate the mean value of the glucose.
Code:

mean_glucose_value <- mean(dataSet$glucose, na.rm = TRUE)

print(mean_glucose_value)

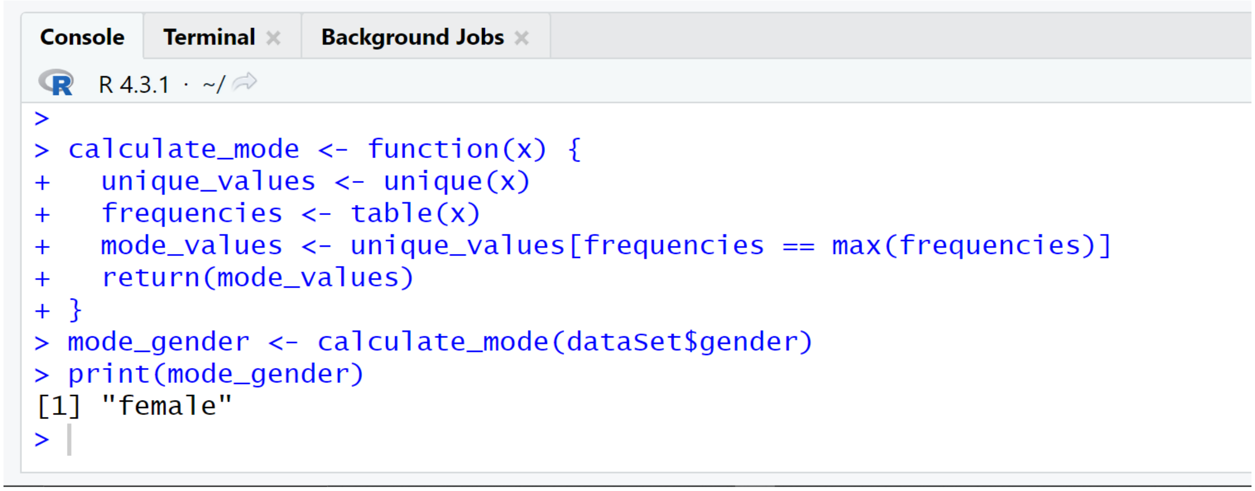


```
> mean_glucose_value <- mean(dataSet$glucose, na.rm = TRUE)
> print(mean_glucose_value)
[1] 148.6467
>
```

6. MODE VALUE OF GENDER: (Categorical) This code is used to calculate the mode value of Gender column.

Code:

```
calculate_mode <- function(x) {

  unique_values <- unique(x)

  frequencies <- table(x)

  mode_values <- unique_values[frequencies == max(frequencies)]

  return(mode_values) }

mode_gender <- calculate_mode(dataSet$gender)

print(mode_gender)
```

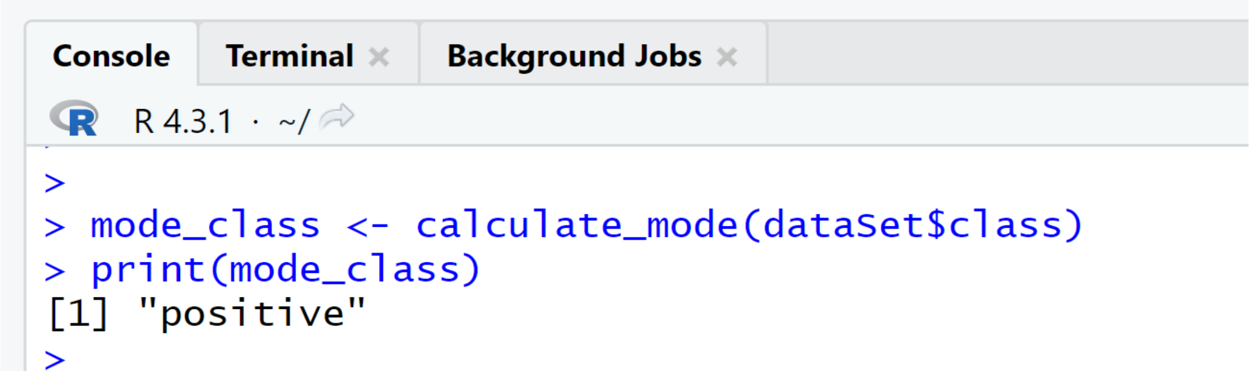


```
>
> calculate_mode <- function(x) {
+   unique_values <- unique(x)
+   frequencies <- table(x)
+   mode_values <- unique_values[frequencies == max(frequencies)]
+   return(mode_values)
+ }
> mode_gender <- calculate_mode(dataSet$gender)
> print(mode_gender)
[1] "female"
> 
```

7. MEAN VALUE OF CLASS: (Categorical) This code is used to calculate the mode value of the Class.

Code:

```
mode_class <- calculate_mode(dataSet$class)

print(mode_class)
```



```
>
> mode_class <- calculate_mode(dataSet$class)
> print(mode_class)
[1] "positive"
>
```

Fixing Missing values

Replacing the missing values in every column with the average mean values (numerical) & mode values (categorical).

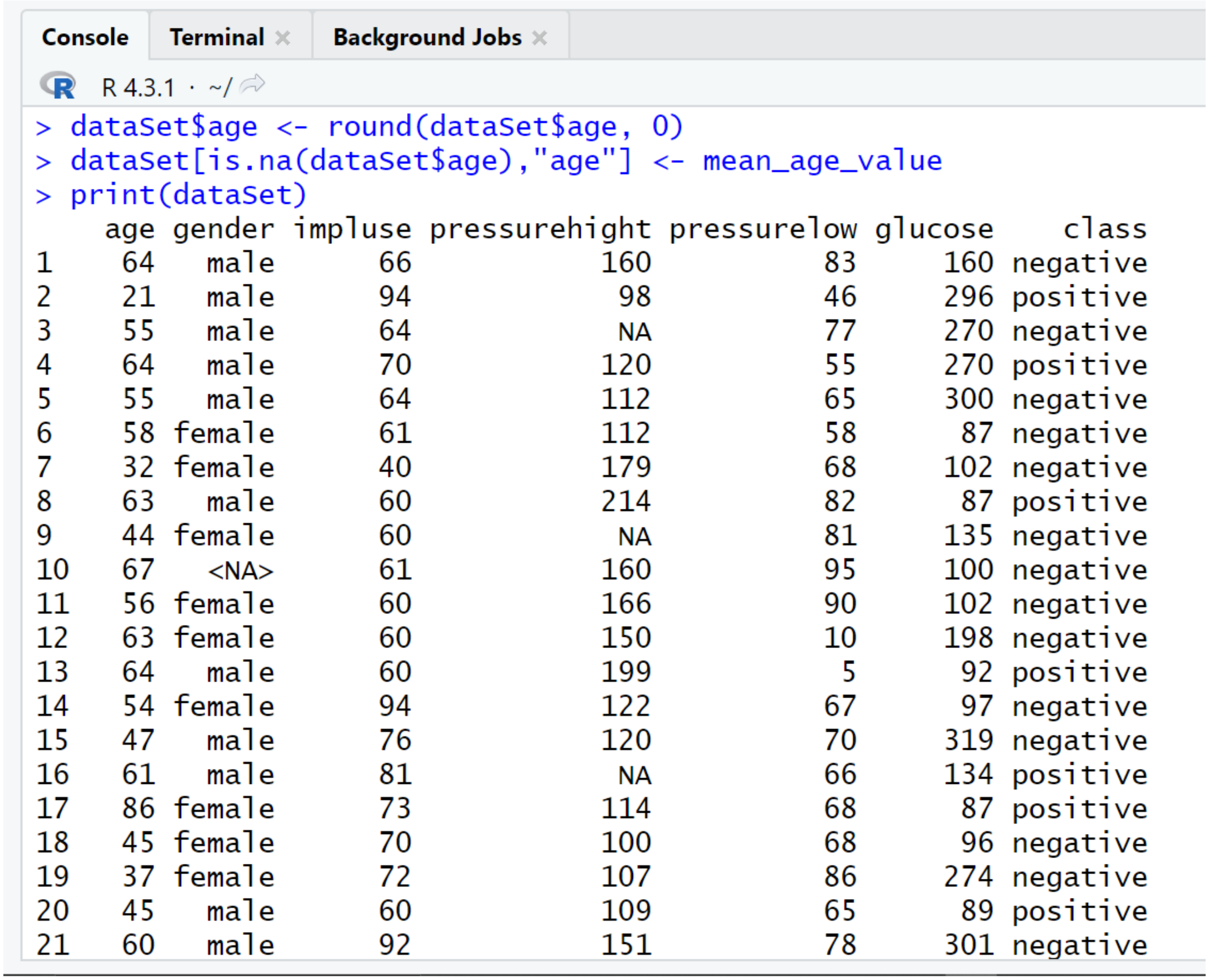
1. Fixing Age Column: There is no missing values in age column, but with the process and code we can fix all the column.

Code:

dataSet$age <- round(dataSet$age, 0)

dataSet[is.na(dataSet$age),"age"] <- mean_age_value

print(dataSet)



```
> dataSet$age <- round(dataSet$age, 0)
> dataSet[is.na(dataSet$age),"age"] <- mean_age_value
> print(dataSet)
   age gender impluse pressurehight pressurelow glucose    class
1   64   male      66           160          83     160 negative
2   21   male      94            98          46     296 positive
3   55   male      64            NA          77     270 negative
4   64   male      70           120          55     270 positive
5   55   male      64           112          65     300 negative
6   58 female      61           112          58      87 negative
7   32 female      40           179          68     102 negative
8   63   male      60           214          82      87 positive
9   44 female      60            NA          81     135 negative
10  67   <NA>      61           160          95     100 negative
11  56 female      60           166          90     102 negative
12  63 female      60           150          10     198 negative
13  64   male      60           199           5      92 positive
14  54 female      94           122          67      97 negative
15  47   male      76           120          70     319 negative
16  61   male      81            NA          66     134 positive
17  86 female      73           114          68      87 positive
18  45 female      70           100          68      96 negative
19  37 female      72           107          86     274 negative
20  45   male      60           109          65      89 positive
21  60   male      92           151          78     301 negative
```

2. Fixing Impulse Column:

Code:

```
dataSet$impluse <- round(dataSet$impluse, 2)

dataSet[is.na(dataSet$impluse),"impluse"] <- mean_impluse_value

print(dataSet)
```

3. Fixing Pressure Height Column:

Code:

```
dataSet$pressurehight <- round(dataSet$pressurehight, 2)

dataSet[is.na(dataSet$pressurehight),"pressurehight"] <- mean_ph_value

print(dataSet)
```

4. Fixing Pressure Low Column:

Code:

```
dataSet$pressurelow <- round(dataSet$pressurelow, 1)

dataSet[is.na(dataSet$pressurelow),"pressurelow"] <- mean_pl_value

print(dataSet)
```

5. Fixing GLUCOSE Column:

Code:

```
dataSet$glucose <- round(dataSet$glucose, 1)

dataSet[is.na(dataSet$glucose),"glucose"] <- mean_glucose_value

print(dataSet)
```

6. Fixing Gender Column(mode):

Code:

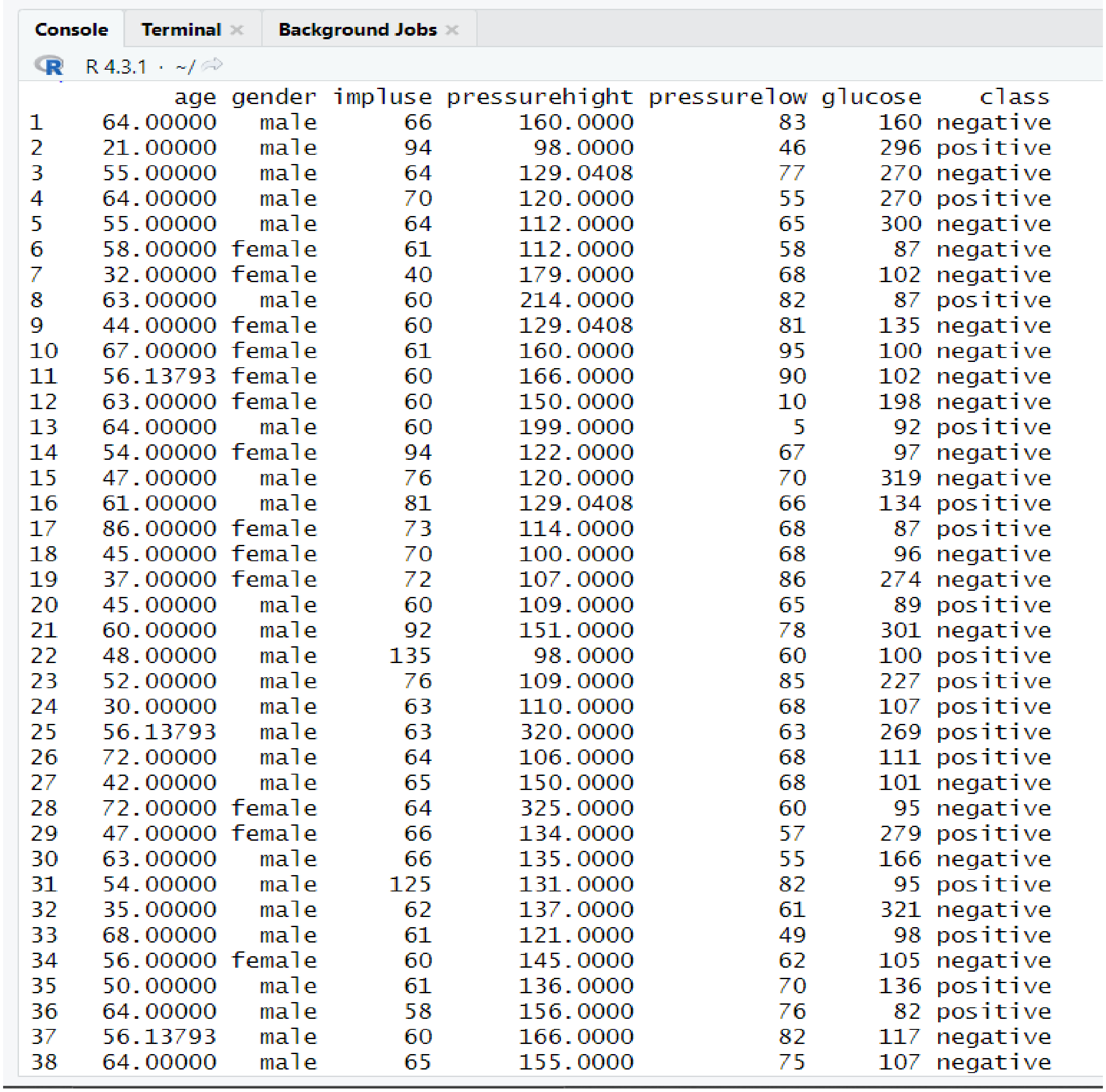
```
dataSet[is.na(dataSet$gender),"gender"] <- mode_gender

print(dataSet)
```

7. Fixing Class Column(mode):

Code:

```
dataSet[is.na(dataSet$class),"class"] <- mode_class

print(dataSet)
```

Here, The Fixed Dataset



```
        age gender impluse pressurehight pressurelow glucose    class
1  64.00000   male      66      160.0000          83     160 negative
2  21.00000   male      94       98.0000          46     296 positive
3  55.00000   male      64      129.0408          77     270 negative
4  64.00000   male      70      120.0000          55     270 positive
5  55.00000   male      64      112.0000          65     300 negative
6  58.00000 female      61      112.0000          58      87 negative
7  32.00000 female      40      179.0000          68     102 negative
8  63.00000   male      60      214.0000          82      87 positive
9  44.00000 female      60      129.0408          81     135 negative
10 67.00000 female      61      160.0000          95     100 negative
11 56.13793 female      60      166.0000          90     102 negative
12 63.00000 female      60      150.0000          10     198 negative
13 64.00000   male      60      199.0000           5      92 positive
14 54.00000 female      94      122.0000          67      97 negative
15 47.00000   male      76      120.0000          70     319 negative
16 61.00000   male      81      129.0408          66     134 positive
17 86.00000 female      73      114.0000          68      87 positive
18 45.00000 female      70      100.0000          68      96 negative
19 37.00000 female      72      107.0000          86     274 negative
20 45.00000   male      60      109.0000          65      89 positive
21 60.00000   male      92      151.0000          78     301 negative
22 48.00000   male     135       98.0000          60     100 positive
23 52.00000   male      76      109.0000          85     227 positive
24 30.00000   male      63      110.0000          68     107 positive
25 56.13793   male      63      320.0000          63     269 positive
26 72.00000   male      64      106.0000          68     111 positive
27 42.00000   male      65      150.0000          68     101 negative
28 72.00000 female      64      325.0000          60      95 negative
29 47.00000 female      66      134.0000          57     279 positive
30 63.00000   male      66      135.0000          55     166 negative
31 54.00000   male     125      131.0000          82      95 positive
32 35.00000   male      62      137.0000          61     321 negative
33 68.00000   male      61      121.0000          49      98 positive
34 56.00000 female      60      145.0000          62     105 negative
35 50.00000   male      61      136.0000          70     136 positive
36 64.00000   male      58      156.0000          76      82 positive
37 56.13793   male      60      166.0000          82     117 negative
38 64.00000   male      65      155.0000          75     107 negative
```

The dataset shows there’s no missing values

Outliers

Seting the layout to 2 rows and 2 columns, so that its easy to show the plots together.
Code:

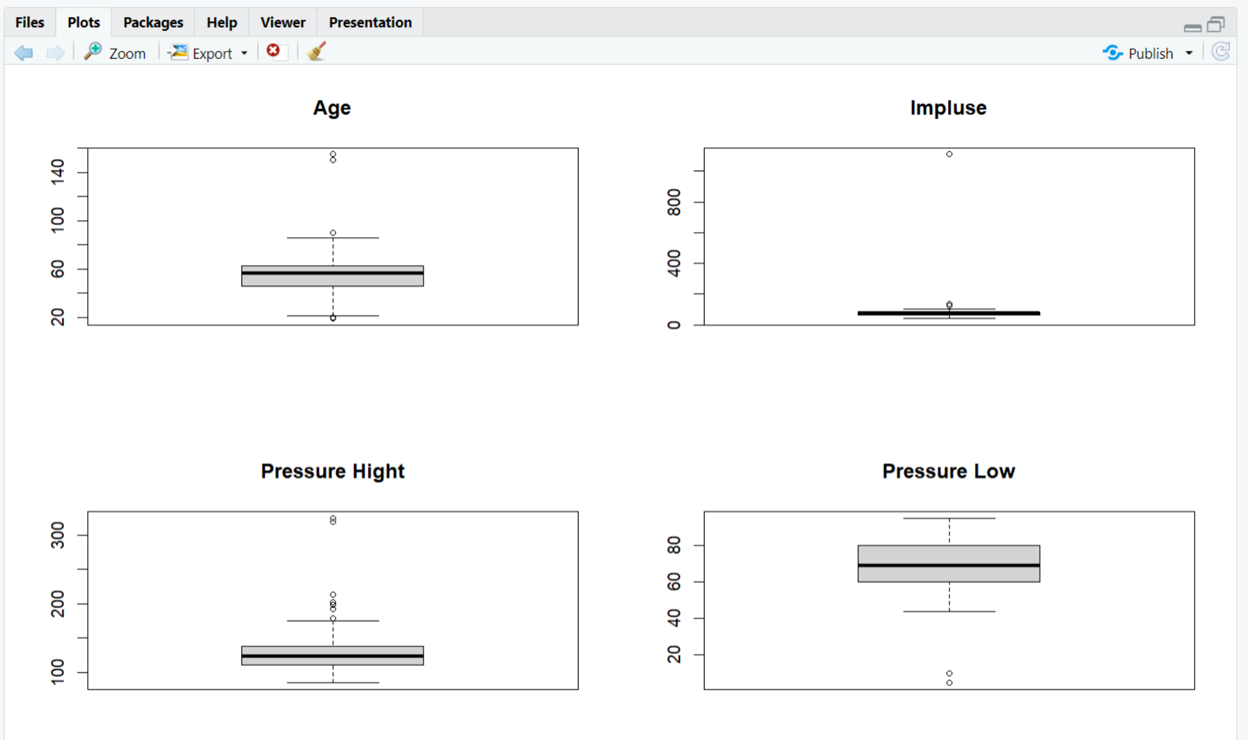
```
par(mfrow = c(2, 2))
```

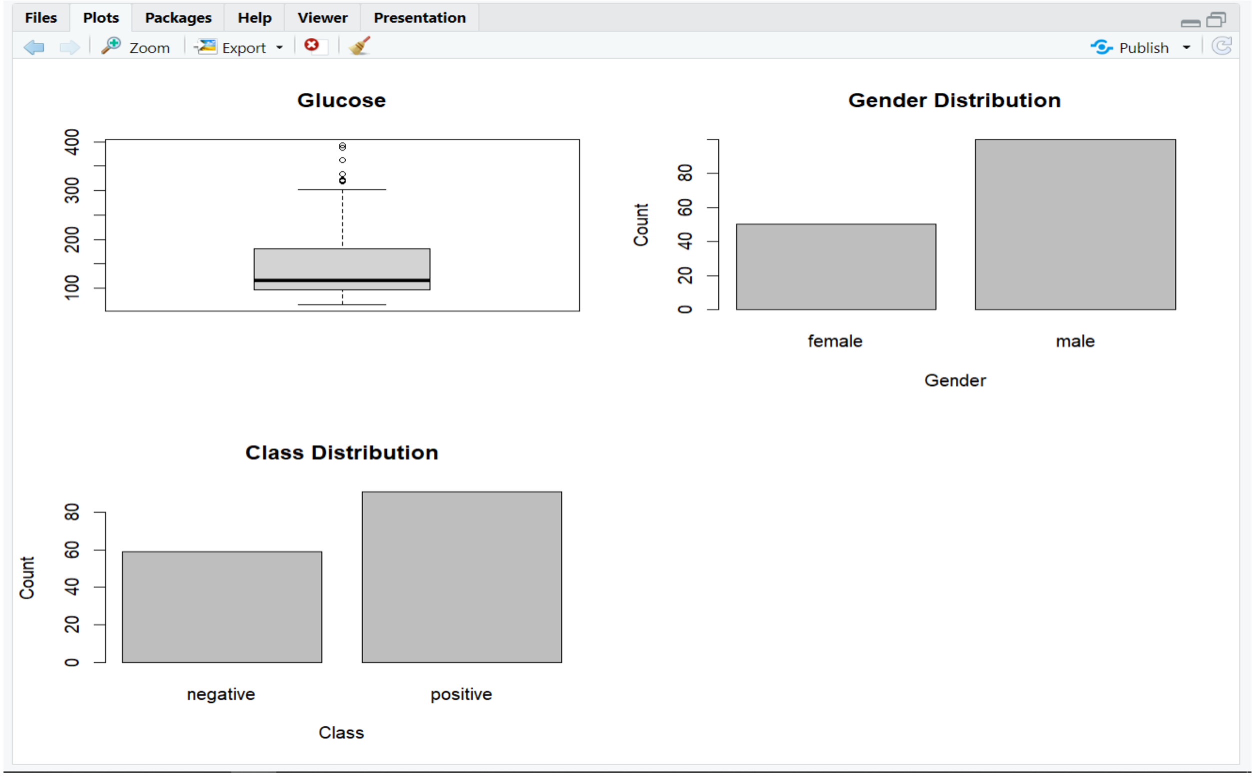
Now Box plot for (Age, Impluse, Pressureheight, Pressurelow, Glucose)
Code:

```
boxplot(dataSet$age, main = "Age")

boxplot(dataSet$impluse, main = "Impluse")

boxplot(dataSet$pressurehight, main = "Pressure Hight")

boxplot(dataSet$pressurelow, main = "Pressure Low")

boxplot(dataSet$glucose, main = "Glucose")
```

Bar plot for Categorical:

```
barplot(table(dataSet$gender), main = "Gender Distribution", xlab = "Gender", ylab = "Count")

barplot(table(dataSet$class), main = "Class Distribution", xlab = "Class", ylab = "Count")
```





Box Plot Info:

Age: It is shown that most of the values are in the range of 20 to 80 and the noisy values are above 140.

Impulse: It can be seen that most of the values are in the range of 40 to 135 and the outlier is 1111.

Pressure high: It is shown that most of the values are in the range of 85 to 215 and the noisy values are above 300.

Pressure low: It is shown that most of the values are in the range of 40 to 95 and the noisy values are below 15.

Glucose: It is shown that most of the values are in the range of 60 to 300 and the noisy values are above 300.

Bar Plot Info:

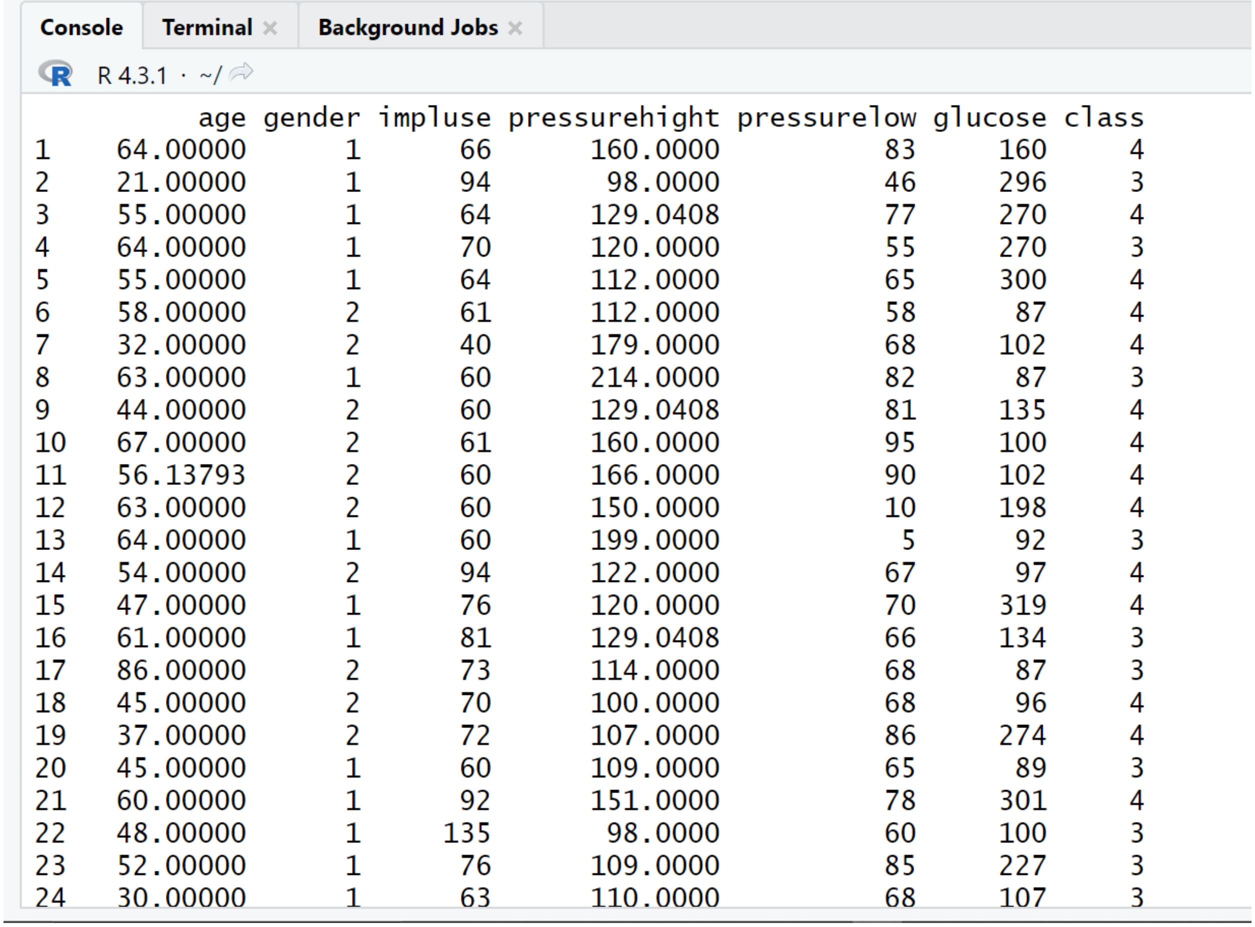
Gender: It is shown that the count of female is 50 and male count is around 100.

Class: It is shown that the count of negative is 60 and positive count is around 100.

<u>Data Type and Conversion</u>

Converting Categorical data in numerical data. [gender(1,2) & class(3,4)]
Code:

```
dataSet$gender<-factor(dataSet$gender,

                      levels = c("male","female"),

                      labels = c(1,2))

dataSet

dataSet$class<-factor(dataSet$class,

                     levels = c("positive","negative"),

                     labels = c(3,4))

dataSet
```



```
        age gender impluse pressurehight pressurelow glucose class
1  64.00000      1      66      160.0000          83     160     4
2  21.00000      1      94       98.0000          46     296     3
3  55.00000      1      64      129.0408          77     270     4
4  64.00000      1      70      120.0000          55     270     3
5  55.00000      1      64      112.0000          65     300     4
6  58.00000      2      61      112.0000          58      87     4
7  32.00000      2      40      179.0000          68     102     4
8  63.00000      1      60      214.0000          82      87     3
9  44.00000      2      60      129.0408          81     135     4
10 67.00000      2      61      160.0000          95     100     4
11 56.13793      2      60      166.0000          90     102     4
12 63.00000      2      60      150.0000          10     198     4
13 64.00000      1      60      199.0000           5      92     3
14 54.00000      2      94      122.0000          67      97     4
15 47.00000      1      76      120.0000          70     319     4
16 61.00000      1      81      129.0408          66     134     3
17 86.00000      2      73      114.0000          68      87     3
18 45.00000      2      70      100.0000          68      96     4
19 37.00000      2      72      107.0000          86     274     4
20 45.00000      1      60      109.0000          65      89     3
21 60.00000      1      92      151.0000          78     301     4
22 48.00000      1     135       98.0000          60     100     3
23 52.00000      1      76      109.0000          85     227     3
24 30.00000      1      63      110.0000          68     107     3
```

Data Exploration

Measure of Spread: (Range)

1.AGE RANGE

Code:

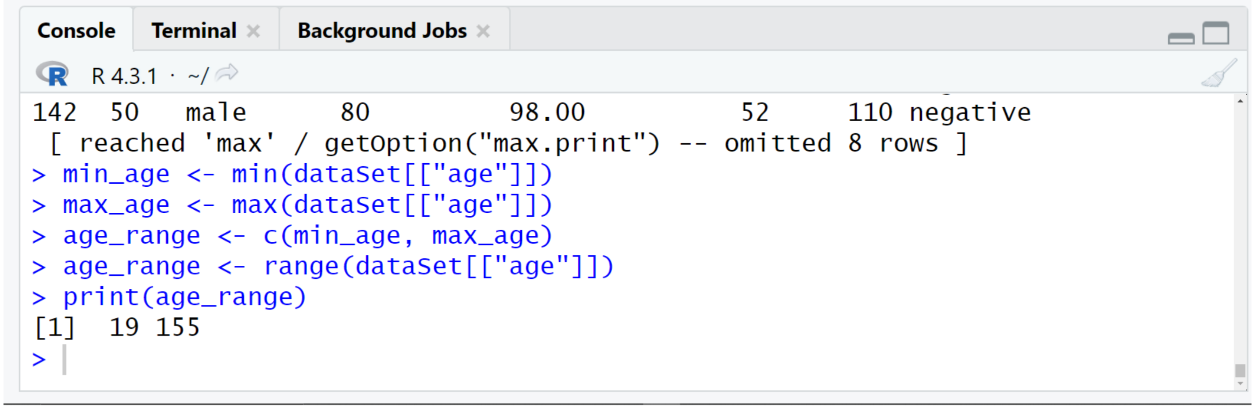
min_age <- min(dataSet[["age"]])

max_age <- max(dataSet[["age"]])

age_range <- c(min_age, max_age)

age_range <- range(dataSet[["age"]])

print(age_range)



```
142   50    male       80          98.00          52      110 negative
 [ reached 'max' / getOption("max.print") -- omitted 8 rows ]
> min_age <- min(dataSet[["age"]])
> max_age <- max(dataSet[["age"]])
> age_range <- c(min_age, max_age)
> age_range <- range(dataSet[["age"]])
> print(age_range)
[1]  19 155
> 
```

This code is used to find the range of age.

2.IMPLUSE RANGE

Code:

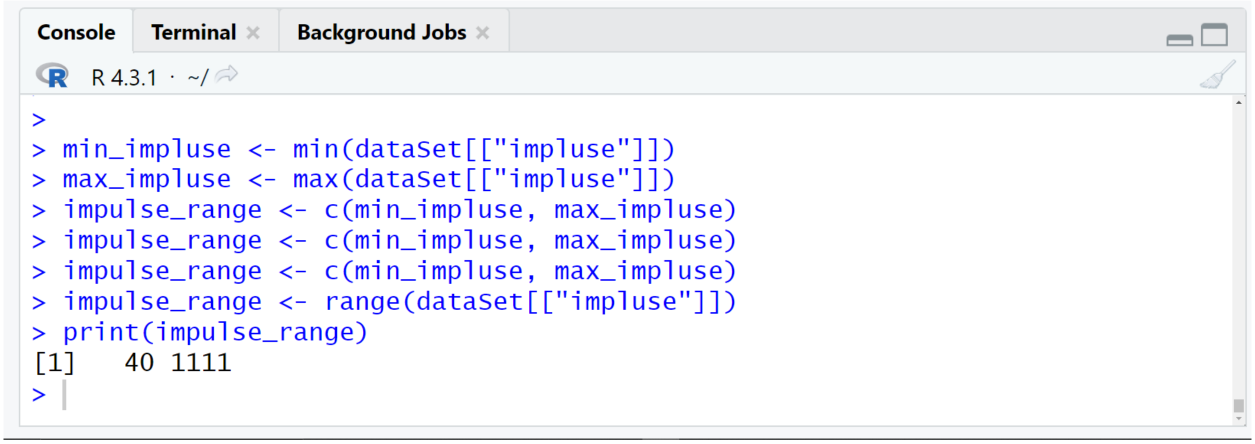
min_impluse <- min(dataSet[["impluse"]])

max_impluse <- max(dataSet[["impluse"]])

impulse_range <- c(min_impluse, max_impluse)

impulse_range <- range(dataSet[["impluse"]])

print(impulse_range)



```
>
> min_impluse <- min(dataSet[["impluse"]])
> max_impluse <- max(dataSet[["impluse"]])
> impulse_range <- c(min_impluse, max_impluse)
> impulse_range <- c(min_impluse, max_impluse)
> impulse_range <- c(min_impluse, max_impluse)
> impulse_range <- range(dataSet[["impluse"]])
> print(impulse_range)
[1]   40 1111
>
```

This code is used to find the range of impulse

Variance & Standard Deviation: (Age)

Variance Code:

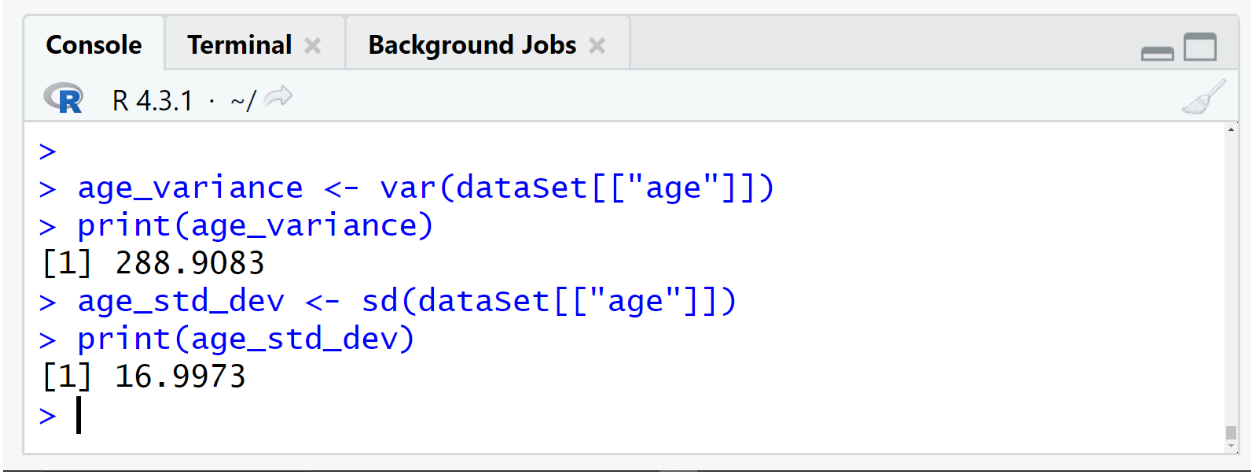
age_variance <- var(dataSet[["age"]])

print(age_variance)

Standard Deviation Code:

age_std_dev <- sd(dataSet[["age"]])

print(age_std_dev)



```
>
> age_variance <- var(dataSet[["age"]])
> print(age_variance)
[1] 288.9083
> age_std_dev <- sd(dataSet[["age"]])
> print(age_std_dev)
[1] 16.9973
>
```

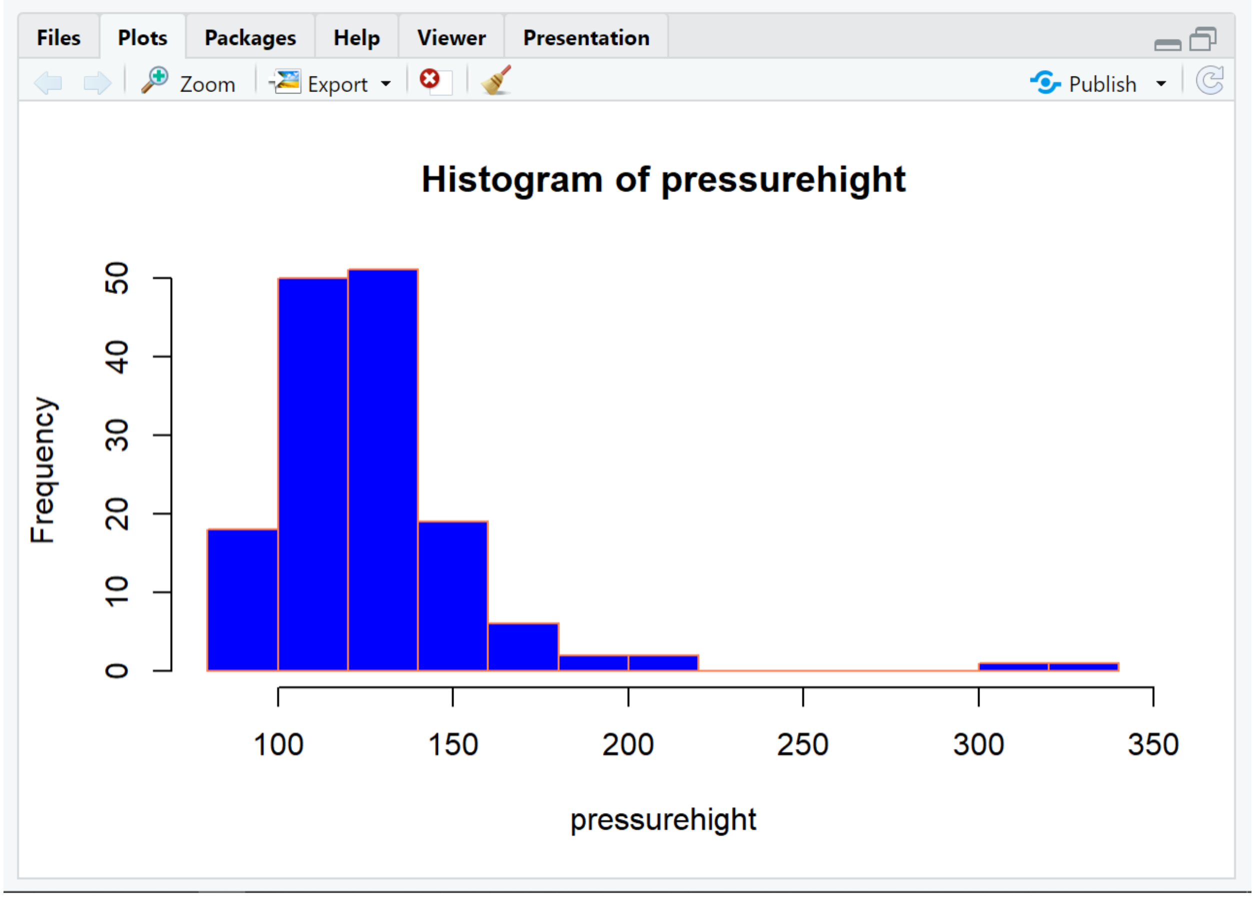
This code is used to find the variance & standard deviation of Age

Univariate Visualization

Histogram: (Pressurehight**)**

Code:

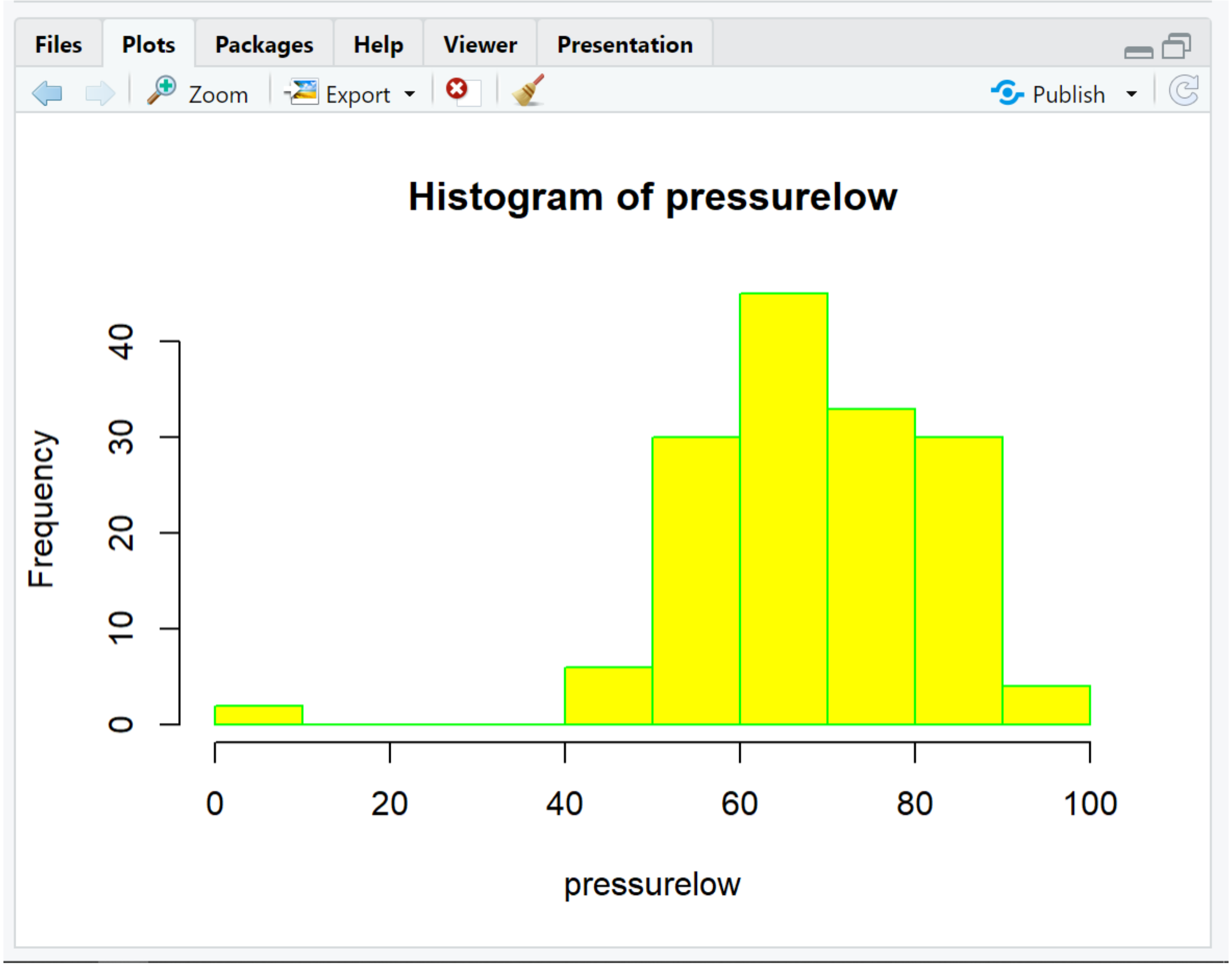
```
hist(dataSet[["pressurehight"]],

    main= paste("Histogram of", "pressurehight"),

    xlab = "pressurehight",

    ylab = "Frequency",

    col = "blue",

    border = "coral")
```



For, Pressurelow

Code:

```
hist(dataSet[["pressurelow"]],

    main= paste("Histogram of", "pressurelow"),

    xlab = "pressurelow",

    ylab = "Frequency",

    col = "yellow",

    border = "green")
```



Bar Plot: (already done in Outliers)