

Introduction to Data Science

Midterm Project

Name	ID	Section
Amit Podder	20-42273-1	C
Writhik Banik Barshan	20-43124-1	C

Topic: Heart Attack Analysis & Prediction Dataset

Dataset Description:

The dataset is about Heart Attack Analysis & Prediction. Datasets like these are typically used for data analysis and machine learning projects to understand the factors that may contribute to heart attack analysis and also predict the likelihood of a heart attack based on certain variables. The dataset contains information about age, sex, chest pain type, resting BP, cholesterol, fasting BS, resting ECG, max HR, exercise angina, old peak, st slope, and heart disease of the patients. In this dataset, we can see four types of chest pain:

ASY : Asymptomatic

TA : Typical Angina

ATA : Atypical Angina

NAP : Non-Anginal Pain

If the fasting blood sugar level is less than 120mg/dl, then the value will be 1 and if the fasting blood sugar level is greater than 120mg/dl, then the value will be 0. Here, 1= true and 0 = false.

It seems the dataset has some missing values as indicated by blanks in certain cells. This data might be used for heart attack analysis and prediction in future heart attack data analysis.

Import the dataset as CSV and print the dataset:

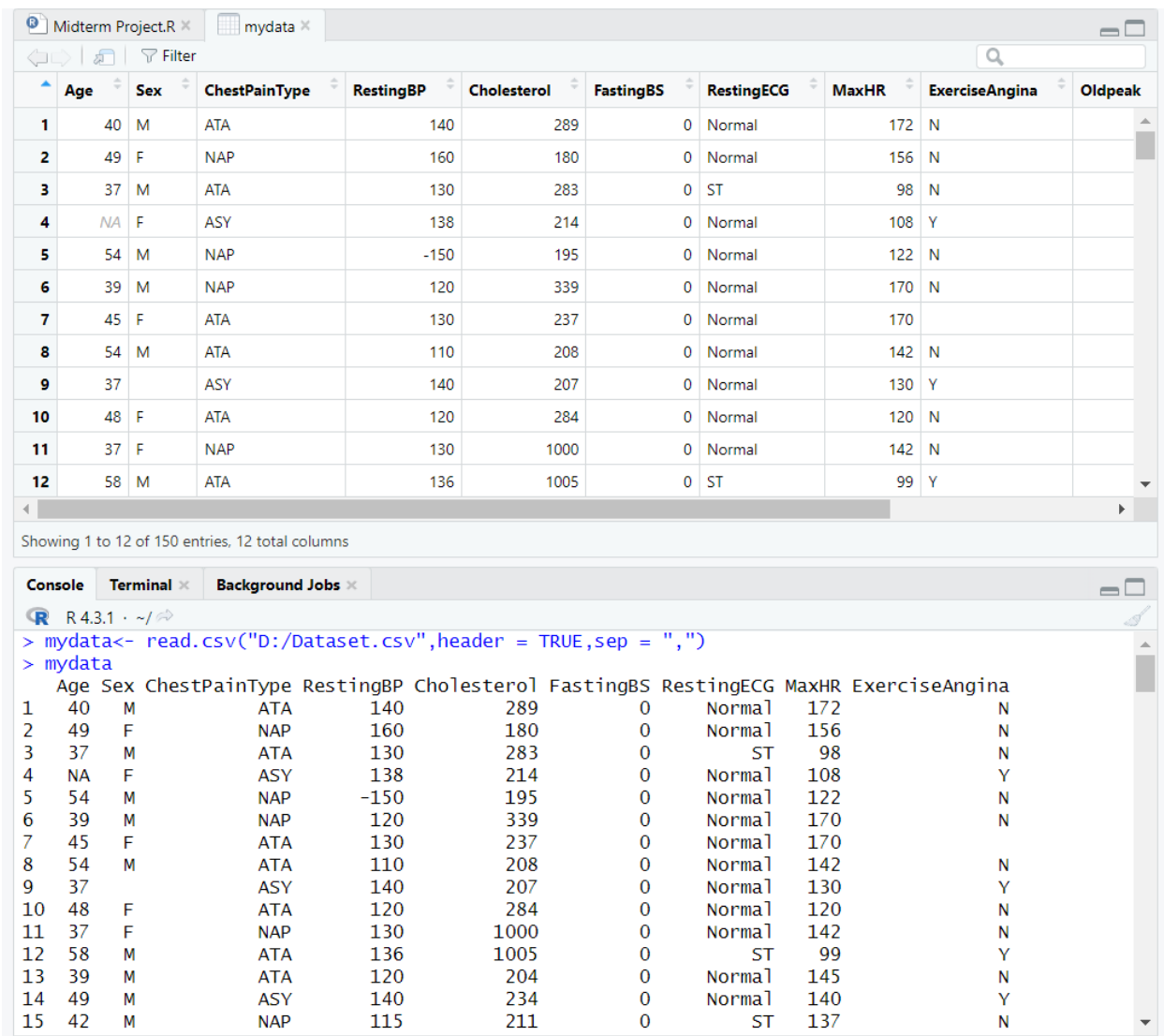
Code:

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

```
mydata
```

```
1 mydata<- read.csv("D:/Dataset.csv",header = TRUE,sep = ",")
2 mydata
```

Output:



	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak
1	40	M	ATA	140	289	0	Normal	172	N	
2	49	F	NAP	160	180	0	Normal	156	N	
3	37	M	ATA	130	283	0	ST	98	N	
4	NA	F	ASY	138	214	0	Normal	108	Y	
5	54	M	NAP	-150	195	0	Normal	122	N	
6	39	M	NAP	120	339	0	Normal	170	N	
7	45	F	ATA	130	237	0	Normal	170		
8	54	M	ATA	110	208	0	Normal	142	N	
9	37		ASY	140	207	0	Normal	130	Y	
10	48	F	ATA	120	284	0	Normal	120	N	
11	37	F	NAP	130	1000	0	Normal	142	N	
12	58	M	ATA	136	1005	0	ST	99	Y	

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina
1	40	M	ATA	140	289	0	Normal	172	N
2	49	F	NAP	160	180	0	Normal	156	N
3	37	M	ATA	130	283	0	ST	98	N
4	NA	F	ASY	138	214	0	Normal	108	Y
5	54	M	NAP	-150	195	0	Normal	122	N
6	39	M	NAP	120	339	0	Normal	170	N
7	45	F	ATA	130	237	0	Normal	170	
8	54	M	ATA	110	208	0	Normal	142	N
9	37		ASY	140	207	0	Normal	130	Y
10	48	F	ATA	120	284	0	Normal	120	N
11	37	F	NAP	130	1000	0	Normal	142	N
12	58	M	ATA	136	1005	0	ST	99	Y
13	39	M	ATA	120	204	0	Normal	145	N
14	49	M	ASY	140	234	0	Normal	140	Y
15	42	M	NAP	115	211	0	ST	137	N

Description:

Here we have imported the code of the dataset as a csv file. We can also see the output of the dataset imported in RStudio.

To see the column name of the dataset:

Code:

```
names(mydata)
```

```
4 names(mydata)
```

Output:

```
> names(mydata)
[1] "Age"          "Sex"          "ChestPainType" "RestingBP"    "Cholesterol"
[6] "FastingBS"    "RestingECG"   "MaxHR"         "ExerciseAngina" "Oldpeak"
[11] "ST_Slope"     "HeartDisease"
> |
```

Description:

In this code, we can see the column name of the dataset. Here, with the help of the code `names()`, we can see all the attribute names present in the dataset.

Finding the Missing(Null) values:

Code:

```
colSums(is.na(mydata))
```

```
6 colSums(is.na(mydata))
```

Output:

```
> colSums(is.na(mydata))
      Age      Sex ChestPainType      RestingBP      Cholesterol      FastingBS
      3         0         0         0         0         0
RestingECG      MaxHR ExerciseAngina      Oldpeak      ST_Slope      HeartDisease
      0         0         0         0         0         0
> |
```

Description:

In this code, we can see all the null values of the dataset. Here, with the help of the code `colSums(is.na())`, we can check missing values in each column.

Finding the specific row number of Missing(Null) Values and remove it from the dataset(for the “Age” attribute):

Code:

```
which(is.na(mydata$Age))
```

```
mydata <- mydata[-4,]
```

```
which(is.na(mydata$Age))
```

```
mydata <- mydata[-23,]
```

```
which(is.na(mydata$Age))
```

```
mydata <- mydata[-31,]
```

```
which(is.na(mydata$Age))
```

```
8  which(is.na(mydata$Age))
9  mydata <- mydata[-4,]
10 which(is.na(mydata$Age))
11 mydata <- mydata[-23,]
12 which(is.na(mydata$Age))
13 mydata <- mydata[-31,]
14 which(is.na(mydata$Age))
```

Output:

```
> which(is.na(mydata$Age))
[1]  4 24 33
> mydata <- mydata[-4,]
> which(is.na(mydata$Age))
[1] 23 32
> mydata <- mydata[-23,]
> which(is.na(mydata$Age))
[1] 31
> mydata <- mydata[-31,]
> which(is.na(mydata$Age))
integer(0)
> |
```

Description:

At first, we found the row numbers where “Age” has missing values. Here, with the help of the code `which(is.na(mydata$Age))`, we found the row numbers where “Age” has null values. After that, with the help of the code `mydata <- mydata[]`, we removed the row as it has null values in it.

Finding the Mean and Median value for the “Age” attribute from the dataset:

Code:

```
mean(mydata$Age)
```

```
median(mydata$Age)
```

```
16 mean(mydata$Age)
17 median(mydata$Age)
```

Output:

```
> mean(mydata$Age)
[1] 49.80952
> median(mydata$Age)
[1] 49
> |
```

Description:

After removing the null values from the “Age” attribute, we have calculated the Mean and Median values for the “Age” attribute. At first, with the help of the code `mean(mydata$Age)`, we calculated the Mean value for the “Age” attribute. After that, with the help of the code `median(mydata$Age)`, we calculated the Median value for the “Age” attribute.

Finding the Variance and Standard Deviation value for the “Age” attribute from the dataset:

Code:

```
var(mydata$Age)
```

```
sd(mydata$Age)
```

```
47 var(mydata$Age)
48 sd(mydata$Age)
```

Output:

```
> var(mydata$Age)
[1] 263.5388
> sd(mydata$Age)
[1] 16.23388
> |
```

Description:

After removing the null values from the “Age” attribute, we have calculated the Variance and Standard Deviation values for the “Age” attribute. At first, with the help of the code `var(mydata$Age)`, we calculated the Variance for the “Age” attribute. After that, with the help of the code `sd(mydata$Age)`, we calculated the Standard Deviation value for the “Age” attribute.

Annotating Datasets:

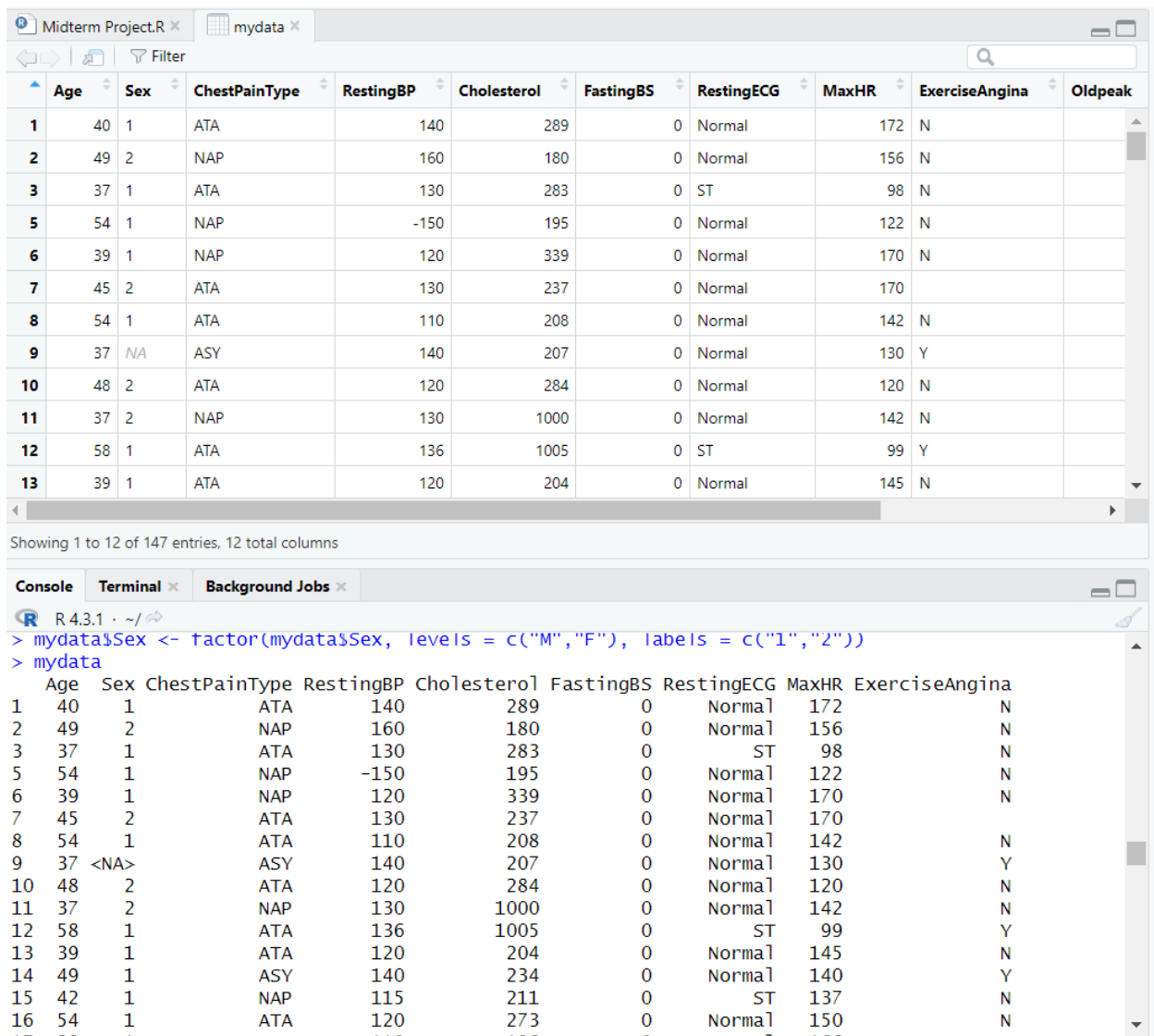
Code:

```
mydata$Sex <- factor(mydata$Sex, levels = c("M","F"), labels = c("1","2"))
```

mydata

```
19 mydata$Sex <- factor(mydata$Sex, levels = c("M","F"), labels = c("1","2"))
20 mydata
```

Output:



The screenshot displays the RStudio interface. The top pane shows a data table with 12 columns: Age, Sex, ChestPainType, RestingBP, Cholesterol, FastingBS, RestingECG, MaxHR, ExerciseAngina, and Oldpeak. The bottom pane shows the R console with the command `mydata$Sex <- factor(mydata$Sex, levels = c("M","F"), labels = c("1","2"))` and the output of the `mydata` object, which is a data frame with 16 rows and 10 columns.

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina	Oldpeak
1	40	1	ATA	140	289	0	Normal	172	N	
2	49	2	NAP	160	180	0	Normal	156	N	
3	37	1	ATA	130	283	0	ST	98	N	
5	54	1	NAP	-150	195	0	Normal	122	N	
6	39	1	NAP	120	339	0	Normal	170	N	
7	45	2	ATA	130	237	0	Normal	170		
8	54	1	ATA	110	208	0	Normal	142	N	
9	37	NA	ASY	140	207	0	Normal	130	Y	
10	48	2	ATA	120	284	0	Normal	120	N	
11	37	2	NAP	130	1000	0	Normal	142	N	
12	58	1	ATA	136	1005	0	ST	99	Y	
13	39	1	ATA	120	204	0	Normal	145	N	

	Age	Sex	ChestPainType	RestingBP	Cholesterol	FastingBS	RestingECG	MaxHR	ExerciseAngina
1	40	1	ATA	140	289	0	Normal	172	N
2	49	2	NAP	160	180	0	Normal	156	N
3	37	1	ATA	130	283	0	ST	98	N
5	54	1	NAP	-150	195	0	Normal	122	N
6	39	1	NAP	120	339	0	Normal	170	N
7	45	2	ATA	130	237	0	Normal	170	
8	54	1	ATA	110	208	0	Normal	142	N
9	37	<NA>	ASY	140	207	0	Normal	130	Y
10	48	2	ATA	120	284	0	Normal	120	N
11	37	2	NAP	130	1000	0	Normal	142	N
12	58	1	ATA	136	1005	0	ST	99	Y
13	39	1	ATA	120	204	0	Normal	145	N
14	49	1	ASY	140	234	0	Normal	140	Y
15	42	1	NAP	115	211	0	ST	137	N
16	54	1	ATA	120	273	0	Normal	150	N

Description:

Here, the “Sex” column is converted into numeric(1 and 2) where ‘1’ represents ‘M’ and ‘2’ represents ‘F’. With the help of the code `mydata$Sex <- factor(mydata$Sex, levels = c("M","F"), labels = c("1","2"))`, we were able to successfully converted ‘M’ and ‘F’ into numeric ‘1’ and ‘2’.

Finding the specific row number of Missing(Null) Values and remove it from the dataset(for the “Sex” attribute):

Code:

```
which(is.na(mydata$Sex))
```

```
mydata <- mydata[-8,]
```

```
which(is.na(mydata$Sex))
```

```
mydata <- mydata[-23,]
```

```
which(is.na(mydata$Sex))
```

```
mydata <- mydata[-35,]
```

```
which(is.na(mydata$Sex))
```

```
22  which(is.na(mydata$Sex))
23  mydata <- mydata[-8,]
24  which(is.na(mydata$Sex))
25  mydata <- mydata[-23,]
26  which(is.na(mydata$Sex))
27  mydata <- mydata[-35,]
28  which(is.na(mydata$Sex))
```

Output:

```
> which(is.na(mydata$Sex))
[1]  8 24 37
> mydata <- mydata[-8,]
> which(is.na(mydata$Sex))
[1] 23 36
> mydata <- mydata[-23,]
> which(is.na(mydata$Sex))
[1] 35
> mydata <- mydata[-35,]
> which(is.na(mydata$Sex))
integer(0)
> |
```


Description:

At first, we found the row numbers where “Sex” has missing values. Here, with the help of the code `which(is.na(mydata$Sex))`, we found the row numbers where “Sex” has null values. After that, with the help of the code `mydata <- mydata[,,]`, we removed the row as it has null values in it.

Summary of the structure of the dataset:

Code:

```
str(mydata)
```

```
30 str(mydata)
```

Output:

```
> str(mydata)
'data.frame': 144 obs. of 12 variables:
 $ Age      : int  40 49 37 54 39 45 54 48 37 58 ...
 $ Sex      : Factor w/ 2 levels "1","2": 1 2 1 1 1 2 1 2 2 1 ...
 $ ChestPainType : chr  "ATA" "NAP" "ATA" "NAP" ...
 $ RestingBP   : int  140 160 130 -150 120 130 110 120 130 136 ...
 $ Cholesterol : int  289 180 283 195 339 237 208 284 1000 1005 ...
 $ FastingBS   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ RestingECG  : chr  "Normal" "Normal" "ST" "Normal" ...
 $ MaxHR       : int  172 156 98 122 170 170 142 120 142 99 ...
 $ ExerciseAngina: chr  "N" "N" "N" "N" ...
 $ Oldpeak     : num  0 1 0 0 0 0 0 0 2 ...
 $ ST_Slope    : chr  "Up" "Flat" "Up" "Up" ...
 $ HeartDisease : int  0 1 0 0 0 0 0 0 1 ...
> |
```

Description:

The structure of the dataset is displayed with the help of the code `str()`.

Descriptive Statistics:

Code:

```
summary(mydata)
```

```
32 summary(mydata)
```

Output:

```
> summary(mydata)
   Age      Sex ChestPainType   RestingBP   Cholesterol
Min.   : 32.00  1:102  Length:144   Min.   : -150.0  Min.   : 85.0
1st Qu.: 42.75  2: 42  Class :character 1st Qu.: 120.0  1st Qu.: 203.5
Median : 49.50      Mode :character Median : 130.0  Median : 241.0
Mean   : 50.01                      Mean   : 128.9  Mean   : 259.5
3rd Qu.: 54.00                      3rd Qu.: 140.0  3rd Qu.: 277.5
Max.   :172.00                      Max.   : 190.0  Max.   :1005.0

   FastingBS   RestingECG   MaxHR   ExerciseAngina   oldpeak
Min.   :0.00000  Length:144   Min.   : 82.0  Length:144   Min.   :0.0000
1st Qu.:0.00000  Class :character 1st Qu.:124.0  Class :character 1st Qu.:0.0000
Median :0.00000  Mode  :character Median :140.0  Mode  :character Median :0.0000
Mean   :0.09028                      Mean   :140.2                      Mean   :0.5556
3rd Qu.:0.00000                      3rd Qu.:156.5                      3rd Qu.:1.0000
Max.   :1.00000                      Max.   :190.0                      Max.   :4.0000

   ST_slope   HeartDisease
Length:144   Min.   :0.0000
Class :character 1st Qu.:0.0000
Mode  :character Median :0.0000
                      Mean   :0.3681
                      3rd Qu.:1.0000
                      Max.   :1.0000

> |
```

Description:

Here, we are using this code to see the descriptive statistics. To see the descriptive statistics, we use the `summary()` function. We can also see the min, max, mean, and median values of the dataset.

Summary in Standard Deviation of numeric columns in the dataset:

Code:

```
install.packages("dplyr")
```

```
library("dplyr")
```

```
mydata%>%summarise_if(is.numeric,sd)
```

```
34 install.packages("dplyr")
35 library("dplyr")
36 mydata%>%summarise_if(is.numeric,sd)
```

Output:

```
> mydata%>%summarise_if(is.numeric,sd)
  Age RestingBP Cholesterol FastingBS MaxHR Oldpeak HeartDisease
1 16.32558  28.64309    116.9542 0.2875796 23.8912 0.8752844    0.4839599
> |
```

Description:

Here, we are using this code to see the standard deviation of numeric columns in the dataset. The standard deviation of numeric columns in the dataset is calculated using the dplyr package.

Standard Deviation of the values stored in a CSV file:

Code:

```
s <- mydata$Age
```

```
sd(s)
```

```
a <- mydata$Cholesterol
```

```
sd(a)
```

```
38 s <- mydata$Age
39 sd(s)
40 a <- mydata$Cholesterol
41 sd(a)
```

Output:

```
> s <- mydata$Age
> sd(s)
[1] 16.32558
> a <- mydata$Cholesterol
> sd(a)
[1] 116.9542
> |
```

Description:

Here, we are using this code to directly calculate the standard deviation of a specific numeric column in the dataset. Using the code “s <- mydata\$ sd(s)”, the standard deviation of the “Age” and “Cholesterol” columns is directly calculated.

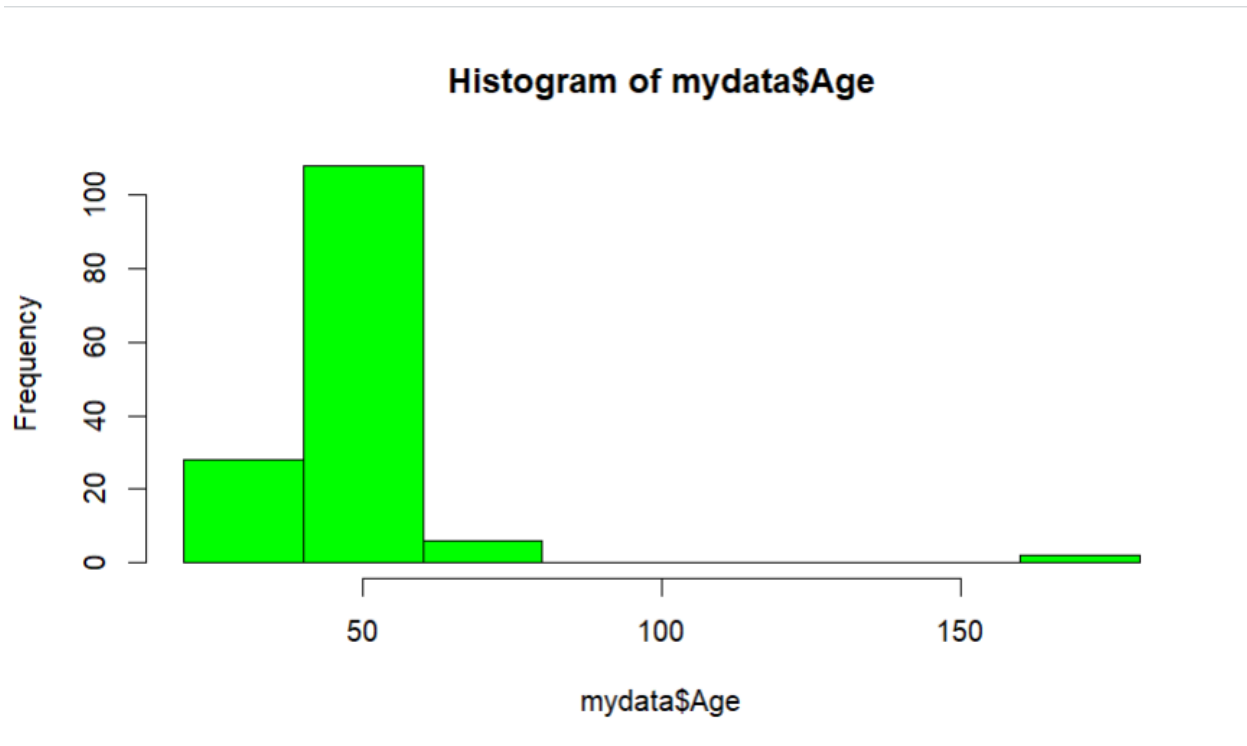
Histogram:

Code:

```
hist(mydata$Age, col = "Green")
```

```
43 hist(mydata$Age, col = "Green")
```

Output:



Description:

Here, we are using this code to make a histogram. Using the code “hist(mydata\$Age, col = "Green")”, the histogram for the “Age” columns is plotted.

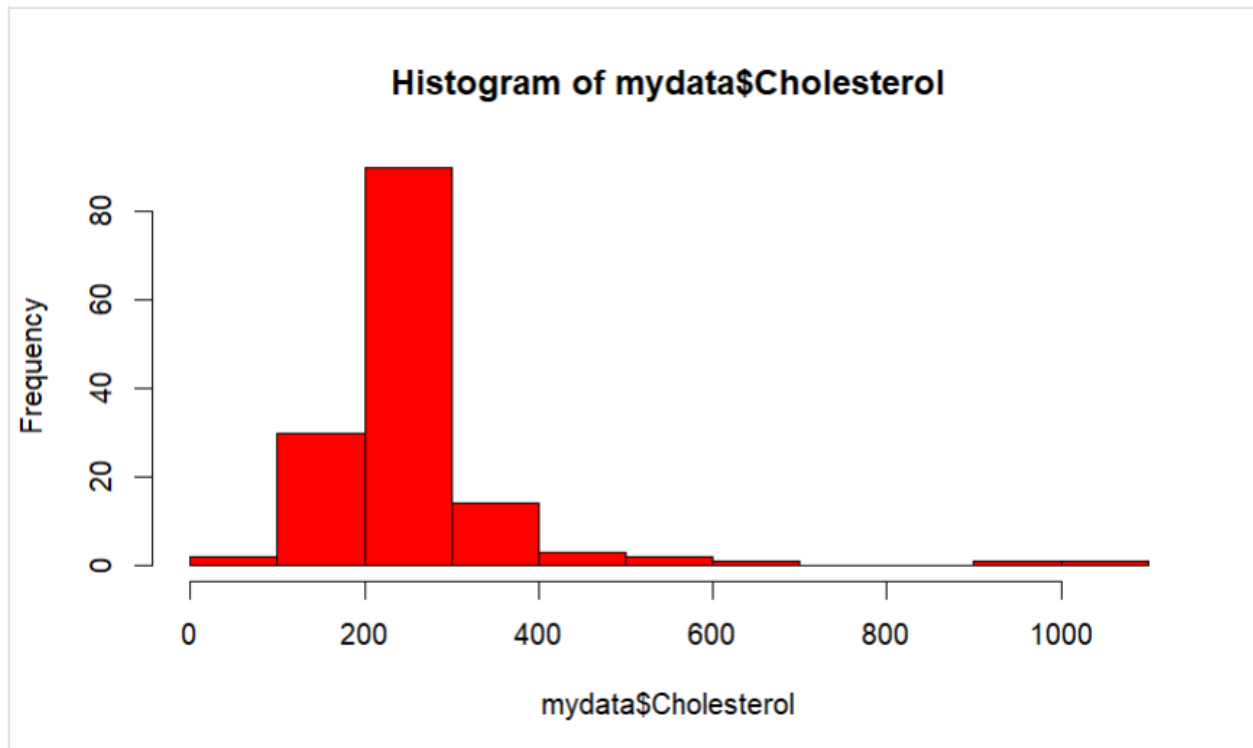
Histogram:

Code:

```
hist(mydata$Cholesterol, col = "Red")
```

```
45 hist(mydata$Cholesterol, col = "Red")
```

Output:



Description:

Here, we are using this code to make a histogram. Using the code “hist(mydata\$Cholesterol, col = "Red")”, the histogram for the “Cholesterol” columns is plotted.