Data Preparation for Titanic dataset

Name	ID	Section-B Group 11
Abir bokhtiar	22-47038-1	
Abdul hadi jebal	21-44912-2	

About the Dataset:

The Titanic dataset contains information on 891 passengers who were aboard the Titanic. It includes 10 columns detailing various attributes: “Gender” (gender of the passenger), “age” (age of each passenger), “sibsp” (number of siblings/spouses aboard), “parch” (number of parents/children aboard), `fare` (ticket fare), `embarked` (port of embarkation), “class” (passenger class), “who” (description of the passenger: man, woman, or child), “alone” (whether the passenger was alone), and “survived” (survival status, where 0 indicates no and 1 indicates yes). The dataset features a mix of numerical and categorical data, with some missing values in every columns except “survived” column.

Load Data:

Code:

```
install.packages("readxl")

library(readxl)
mainData <- read_excel("E:/desktop/Data Science/MID Project/Abir/Midterm_Project_Dataset_section(B).xlsx")
mainData
```

Output:

```
> mainData
# A tibble: 105 x 10
   Gender   age sibsp parch fare              embarked class  who   alone survived
   <chr>  <dbl> <dbl> <dbl> <chr>             <chr>    <chr>  <chr> <lgl>     <dbl>
 1 female    24     0     0 7.7957999999999998 S        Third  mannn TRUE          0
 2 female    17     0     0 8.6624999999999996 S        Third  man   TRUE          0
 3 male      21     0     0 7.75              Q        Third  woman TRUE          0
 4 male      35     0     0 7.6292            Q        Third  woman TRUE          0
 5 male      37     0     0 9.5875000000000004 S        Third  woman TRUE          0
 6 male      16     0     0 86.5              S        First  woman TRUE          1
 7 female    NA     1     0 108.9             C        First  mannn FALSE         0
 8 male      33     0     2 NA                S        Second woman FALSE         0
 9 female    40     0     0 26.55             S        First  man   TRUE          1
10 female    28     0     0 22.524999999999999 S        Third  man   TRUE          0
# ℹ 95 more rows
# ℹ Use `print(n = ...)` to see more rows
>
```

Description: Load Dataset to mainData

Data Structure:

Code:

```
numberOfColumn<-ncol(mainData)
numberOfRow<-nrow(mainData)
cat("number Of Columns: ", numberOfColumn,"\n")
cat("number Of Rows: ", numberOfRow,"\n")
str(mainData)
```

Output:

```
> numberOfColumn<-ncol(mainData)
> numberOfRow<-nrow(mainData)
> cat("number Of Columns: ", numberOfColumn,"\n")
number Of Columns:  10
> cat("number Of Rows: ", numberOfRow,"\n")
number Of Rows:  105
> str(mainData)
tibble [105 × 10] (S3: tbl_df/tbl/data.frame)
 $ Gender  : chr [1:105] "female" "female" "male" "male" ...
 $ age     : num [1:105] 24 17 21 35 37 16 NA 33 40 28 ...
 $ sibsp   : num [1:105] 0 0 0 0 0 0 1 0 0 0 ...
 $ parch   : num [1:105] 0 0 0 0 0 0 0 2 0 0 ...
 $ fare    : chr [1:105] "7.7957999999999998" "8.6624999999999996" "7.75" "7.6292" ...
 $ embarked: chr [1:105] "S" "S" "Q" "Q" ...
 $ class   : chr [1:105] "Third" "Third" "Third" "Third" ...
 $ who     : chr [1:105] "mannn" "man" "woman" "woman" ...
 $ alone   : logi [1:105] TRUE TRUE TRUE TRUE TRUE TRUE ...
 $ survived: num [1:105] 0 0 0 0 0 1 0 0 1 0 ...
>
```

Description: To see the summary of the structure of data set. All the attributes with their type and their instances are shown here. The total number of instances is 105 and columns are 10.

Remove unnecessary rows:

Code:

```
rows_to_discard <- apply(mainData, 1, function(row) {
  sum(!is.na(row)) == 1 && !is.na(row["survived"])
})

cleanData3 <- mainData

cleanData3 <- mainData[!rows_to_discard, ]
cleanData3
numberOfRowMain<-nrow(mainData)
numberOfRow<-nrow(cleanData3)
cat("number Of Rows: ", numberOfRowMain,"\n")
cat("number Of Rows: ", numberOfRow,"\n")
```

Output:

```
> rows_to_discard <- apply(mainData, 1, function(row) {
+   sum(!is.na(row)) == 1 && !is.na(row["survived"])
+ })
>
> cleanData3 <- mainData
>
> cleanData3 <- mainData[!rows_to_discard, ]
> cleanData3
# A tibble: 103 × 10
   Gender   age sibsp parch fare               embarked class  who   alone survived
   <chr>  <dbl> <dbl> <dbl> <chr>              <chr>    <chr>  <chr> <lgl>     <dbl>
 1 female    24     0     0 7.7957999999999998 S        Third  mannn TRUE          0
 2 female    17     0     0 8.6624999999999996 S        Third  man   TRUE          0
 3 male      21     0     0 7.75               Q        Third  woman TRUE          0
 4 male      35     0     0 7.6292             Q        Third  woman TRUE          0
 5 male      37     0     0 9.5875000000000004 S        Third  woman TRUE          0
 6 male      16     0     0 86.5               S        First  woman TRUE          1
 7 female    NA     1     0 108.9              C        First  mannn FALSE         0
 8 male      33     0     2 NA                 S        Second woman FALSE         0
 9 female    40     0     0 26.55              S        First  man   TRUE          1
10 female    28     0     0 22.524999999999999 S        Third  man   TRUE          0
# ℹ 93 more rows
# ℹ Use `print(n = ...)` to see more rows
> numberOfRowMain<-nrow(mainData)
> numberOfRow<-nrow(cleanData3)
> cat("number Of Rows: ", numberOfRowMain,"\n")
number Of Rows:  105
> cat("number Of Rows: ", numberOfRow,"\n")
number Of Rows:  103
>
```

Description: Last two rows were unnecessary, so we needed to remove them.
Number of instances: 105
Number of instances after removal: 103

Unique Categories:

Code:

unique(mainData$age)
unique(mainData$who)

Output:

```
> unique(mainData$age)
 [1]  24  17  21  35  37  16  NA  33  40  28  26  29  30  36  54  47  34  55  22  44
[21]  41  50  45  48  23   2  10  20  32   9  11  64  19   8  27  25  62  39  53 139
[41]  18  60 152 149
> unique(mainData$who)
[1] "mannn" "man"   "woman" "child" NA
>
```

Description: Using unique functions we can find unique attributes. Here in age category, there are: different numeric values as well as NA which is missing value. Further we will take care of them in missing values section.
In who attribute, there are unique values like: "mannn", "man", "woman" , “child” and NA. Here also missing value exists. Also here "mannn" is invalid value of who attribute.

Handle Invalid values:

Code:

```
realData <- mainData
realData$who <- ifelse(substr(tolower(realData$who), 1, 1) == "m", "man",
                ifelse(substr(tolower(realData$who), 1, 1) == "w", "woman",
                    ifelse(substr(tolower(realData$who), 1, 1) == "c", "child",
                        NA)))
unique(realData$who)
```

Output:

```
> realData <- mainData
> realData <- mainData
>
> realData$who <- ifelse(substr(tolower(realData$who), 1, 1) == "m", "man",
+                        ifelse(substr(tolower(realData$who), 1, 1) == "w", "woman",
+                               ifelse(substr(tolower(realData$who), 1, 1) == "c", "chi
ld",
+                                      NA)))
>
> unique(realData$who)
[1] "man"   "woman" "child" NA
>
```

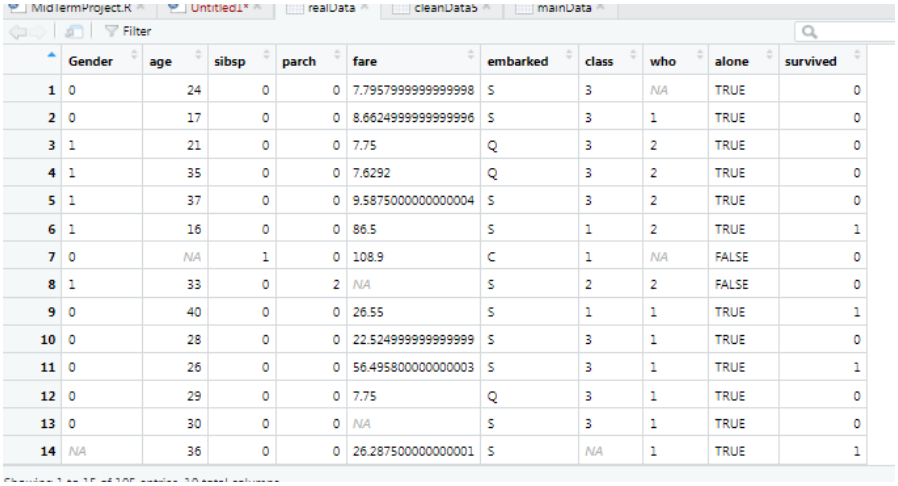
Description: Previously some invalid values were found on the who attribute. We assumed that if the starting character was m means man, w means woman, c means child and ignored further characters of each instance in who attribute. And after replacing invalid values, there's no extra invalid values found using unique() function.

Annotating Datasets:

Code:

```
realData$Gender <- factor(realData$Gender,
                  levels = c("male", "female"),
                  labels = c(1, 0))
realData$Gender

realData$class <- factor(realData$class,
                  levels = c("First", "Second", "Third"),
                  labels = c(1, 2, 3))
realData$class

realData$who <- factor(realData$who,
                levels = c("child", "man", "woman"),
                labels = c(0, 1, 2))
realData
```

Output:



	Gender	age	sibsp	parch	fare	embarked	class	who	alone	survived
1	0	24	0	0	7.7957999999999998	S	3	*NA*	TRUE	0
2	0	17	0	0	8.6624999999999996	S	3	1	TRUE	0
3	1	21	0	0	7.75	Q	3	2	TRUE	0
4	1	35	0	0	7.6292	Q	3	2	TRUE	0
5	1	37	0	0	9.5875000000000004	S	3	2	TRUE	0
6	1	16	0	0	86.5	S	1	2	TRUE	1
7	0	*NA*	1	0	108.9	C	1	*NA*	FALSE	0
8	1	33	0	2	*NA*	S	2	2	FALSE	0
9	0	40	0	0	26.55	S	1	1	TRUE	1
10	0	28	0	0	22.524999999999999	S	3	1	TRUE	0
11	0	26	0	0	56.495800000000003	S	3	1	TRUE	1
12	0	29	0	0	7.75	Q	3	1	TRUE	0
13	0	30	0	0	*NA*	S	3	1	TRUE	0
14	*NA*	36	0	0	26.287500000000001	S	*NA*	1	TRUE	1

Description: The value of these categorical attributes has been converted to numerical values using factor() function.
For Gender attribute: "male"→1, "female"→0
For class attribute: "first"→1, "second"→2, "third"→3
For who attribute: "child"→0, "man"→1, "woman"→2

Data Statistics:

Code:
summary(realData)
Output:

```
> summary(realData)
  Gender        age             sibsp            parch            fare
 1   :37   Min.   :  2.00   Min.   :0.0000   Min.   :0.0000   Length:105
 0   :61   1st Qu.: 23.50   1st Qu.:0.0000   1st Qu.:0.0000   Class :character
 NA's: 7   Median : 33.00   Median :0.0000   Median :0.0000   Mode  :character
           Mean   : 36.25   Mean   :0.3495   Mean   :0.3398
           3rd Qu.: 40.50   3rd Qu.:1.0000   3rd Qu.:0.0000
           Max.   :152.00   Max.   :4.0000   Max.   :4.0000
           NA's   :14       NA's   :2        NA's   :2
   embarked          class       who        alone           survived
 Length:105         1   :27   0   : 5   Mode :logical   Min.   :0.0000
 Class :character   2   :20   1   :61   FALSE:37        1st Qu.:0.0000
 Mode  :character   3   :52   2   :35   TRUE :66        Median :0.0000
                    NA's: 6   NA's: 4   NA's :2         Mean   :0.3619
                                                       3rd Qu.:1.0000
                                                       Max.   :1.0000

>
```

Description: Descriptive Statistics has been shown using summary() function. This summary will help on our following works.

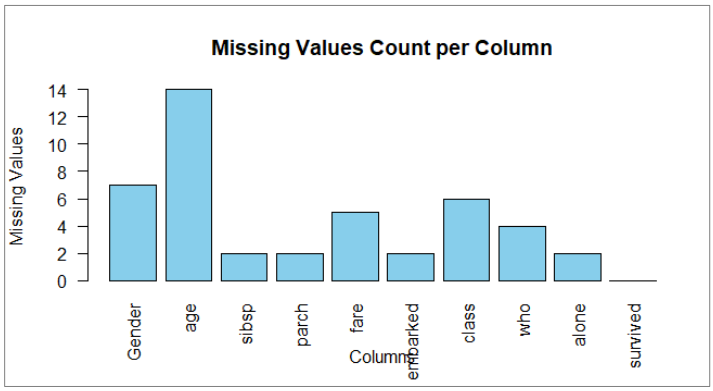
Find Missing Values:

Code:
na_counts <- colSums(is.na(realData))
print(na_counts)
Output:

```
> na_counts <- colSums(is.na(realData))
> print(na_counts)
  Gender      age    sibsp    parch     fare embarked    class      who
       7       14        2        2        5        2        6        4
   alone survived
       2        0
> |
```

Visualize:

```
barplot(na_counts,
    main = "Missing Values Count per Column",
    xlab = "Columns",
    ylab = "Missing Values",
    col = "skyblue",
    las = 2)
```



Description: Using colSums() and is.na() we found the number of missing values for each attribute.
The following missing values were found:
Gender: 7
age: 14
sibsp: 2
parch: 2
fare: 5
embarked: 2
class: 6
who: 4
alone: 2
survived: 0

Handle Missing Values:

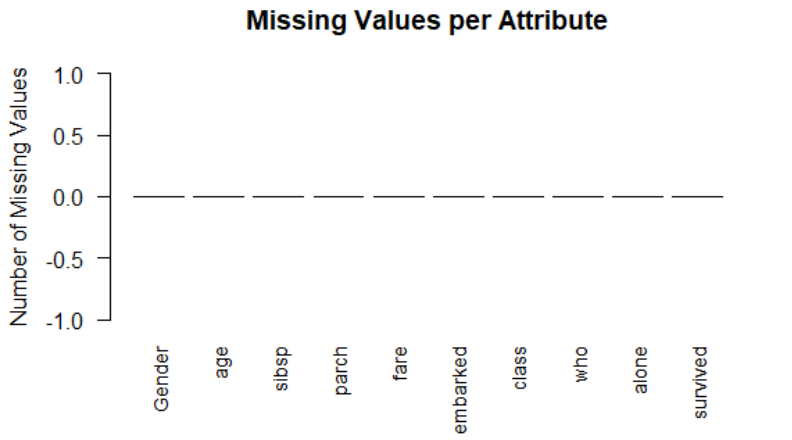
1. By Discard Instances:

Code:

```
cleanData1 <- na.omit(realData, cols = "")
na_counts <- colSums(is.na(cleanData1))
print(na_counts)
barplot(na_counts, names.arg = names(na_counts),
    ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
    main = "Missing Values per Attribute", las =2)
```

Output:

```
> cleanData1 <- na.omit(realData, cols = "")
> na_counts <- colSums(is.na(cleanData1))
> print(na_counts)
  Gender      age    sibsp    parch     fare embarked    class      who    alone
       0        0        0        0        0        0        0        0        0
survived
       0
> barplot(na_counts, names.arg = names(na_counts),
+         ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
+         main = "Missing Values per Attribute", las =2)
> |
```



Description: All the instances having null values have been removed using na.omit() function.

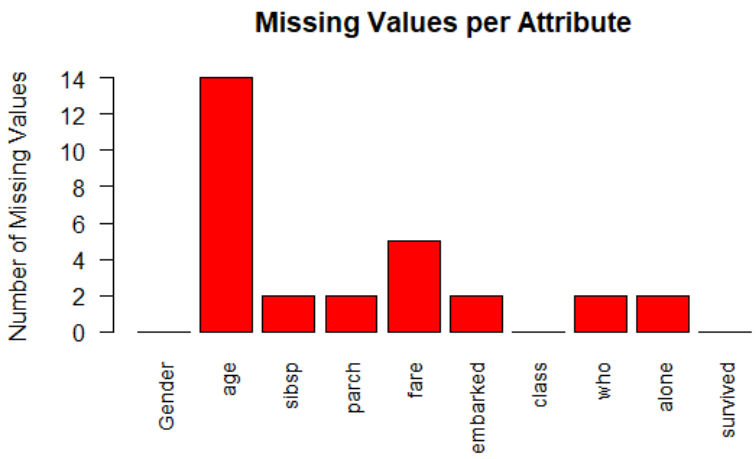
2. Replace by Most Frequent/Average Value

Code: (for categorical attribute)

```
cleanData2 <- mainData
mode_Gender <- names(sort(table(cleanData2$Gender), decreasing = TRUE))[1]
cleanData2$Gender[is.na(cleanData2$Gender)] <- mode_Gender

mode_class <- names(sort(table(cleanData2$class), decreasing = TRUE))[1]
cleanData2$class[is.na(cleanData2$class)] <- mode_class

cleanData2

na_counts <- colSums(is.na(cleanData2))
print(na_counts)
barplot(na_counts, names.arg = names(na_counts),
    ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
    main = "Missing Values per Attribute", las =2)
```

Output:

```
> cleanData2 <- mainData
> mode_Gender <- names(sort(table(cleanData2$Gender), decreasing = TRUE))[1]
> cleanData2$Gender[is.na(cleanData2$Gender)] <- mode_Gender
>
> mode_class <- names(sort(table(cleanData2$class), decreasing = TRUE))[1]
> cleanData2$class[is.na(cleanData2$class)] <- mode_class
>
> cleanData2
# A tibble: 105 × 10
   Gender   age sibsp parch fare               embarked class  who   alone survived
   <chr>  <dbl> <dbl> <dbl> <chr>              <chr>    <chr>  <chr> <lgl>     <dbl>
 1 female    24     0     0 7.7957999999999998 S        Third  mannn TRUE          0
 2 female    17     0     0 8.6624999999999996 S        Third  man   TRUE          0
 3 male      21     0     0 7.75               Q        Third  woman TRUE          0
 4 male      35     0     0 7.6292             Q        Third  woman TRUE          0
 5 male      37     0     0 9.5875000000000004 S        Third  woman TRUE          0
 6 male      16     0     0 86.5               S        First  woman TRUE          1
 7 female    NA     1     0 108.9              C        First  mannn FALSE         0
 8 male      33     0     2 NA                 S        Second woman FALSE         0
 9 female    40     0     0 26.55              S        First  man   TRUE          1
10 female    28     0     0 22.524999999999999 S        Third  man   TRUE          0
# ℹ 95 more rows
# ℹ Use `print(n = ...)` to see more rows
>
> na_counts <- colSums(is.na(cleanData2))
> print(na_counts)
  Gender      age    sibsp    parch     fare embarked    class      who    alone
       0       14        2        2        5        2        0        2        2
survived
       0
> barplot(na_counts, names.arg = names(na_counts),
+         ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
+         main = "Missing Values per Attribute", las =2)
> |
```



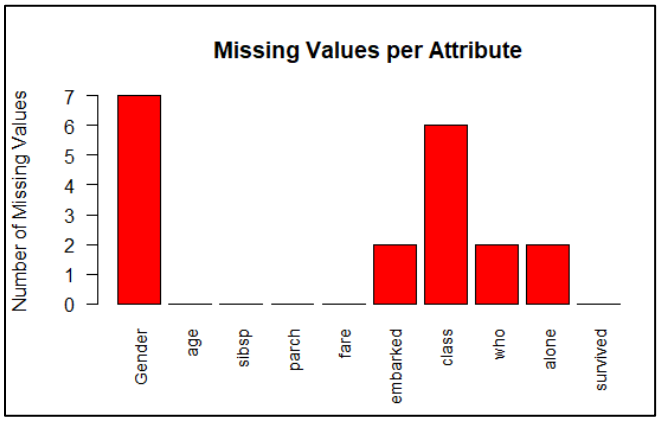
Description: After sorting the values of a column, the most frequent value has been found for the columns, replaced with missing values of that column.

Code: (for numerical attribute)

```
cleanData3 <- mainData
cleanData3$fare <- as.numeric(cleanData3$fare)

for(col_name in names(cleanData3)) {
 if(is.numeric(cleanData3[[col_name]])) {
  column_mean <- mean(cleanData3[[col_name]], na.rm = TRUE)

  cleanData3[[col_name]][is.na(cleanData3[[col_name]])] <- column_mean
  cleanData3[[col_name]] <- round(cleanData3[[col_name]], digits = 0)
 }
}
na_counts <- colSums(is.na(cleanData3))
print(na_counts)
barplot(na_counts, names.arg = names(na_counts),
     ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
     main = "Missing Values per Attribute", las =2)
```

Output:

```
> cleanData3 <- mainData
> cleanData3$fare <- as.numeric(cleanData3$fare)
Warning message:
NAs introduced by coercion
>
> for(col_name in names(cleanData3)) {
+   if(is.numeric(cleanData3[[col_name]])) {
+     column_mean <- mean(cleanData3[[col_name]], na.rm = TRUE)
+
+     cleanData3[[col_name]][is.na(cleanData3[[col_name]])] <- column_mean
+     cleanData3[[col_name]] <- round(cleanData3[[col_name]], digits = 0)
+   }
+ }
>
> na_counts <- colSums(is.na(cleanData3))
> print(na_counts)
  Gender      age    sibsp    parch     fare embarked    class      who    alone
       7        0        0        0        0        2        6        2        2
survived
       0
> barplot(na_counts, names.arg = names(na_counts),
+         ylab = "Number of Missing Values", col = "red",cex.names = 0.9,
+         main = "Missing Values per Attribute", las =2)
>
```



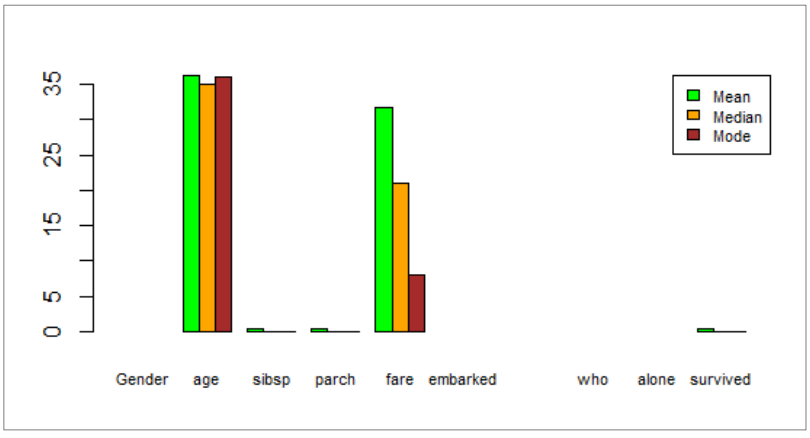
Description: After fare column was converted into numerical then for each column with numerical attributes, the estimated values for each row were founded using the average value. Then after rounding the estimated values, the missing values were replaced with them.

Mean-Median-Mode Graph:

Code:

```
getMode <- function(v) {
 tabulated <- table(v)
 mode_value <- names(sort(tabulated, decreasing = TRUE))[1]
 return(as.numeric(mode_value))
}

means <- sapply(cleanData2, function(x) if(is.numeric(x)) mean(x, na.rm = TRUE) else NA)
medians <- sapply(cleanData2, function(x) if(is.numeric(x)) median(x, na.rm = TRUE) else NA)
modes <- sapply(cleanData2, function(x) if(is.numeric(x) || is.factor(x) || is.character(x)) getMode(x)
else NA)

stat_values <- rbind(means, medians, modes)

row_names <- c("Mean", "Median", "Mode")
rownames(stat_values) <- row_names

barplot(stat_values, beside = TRUE,
    col = c("green", "orange", "brown"),
    legend.text = row_names, args.legend = list(x = "topright", cex = 0.7),
    cex.names = 0.7)
```

Output:



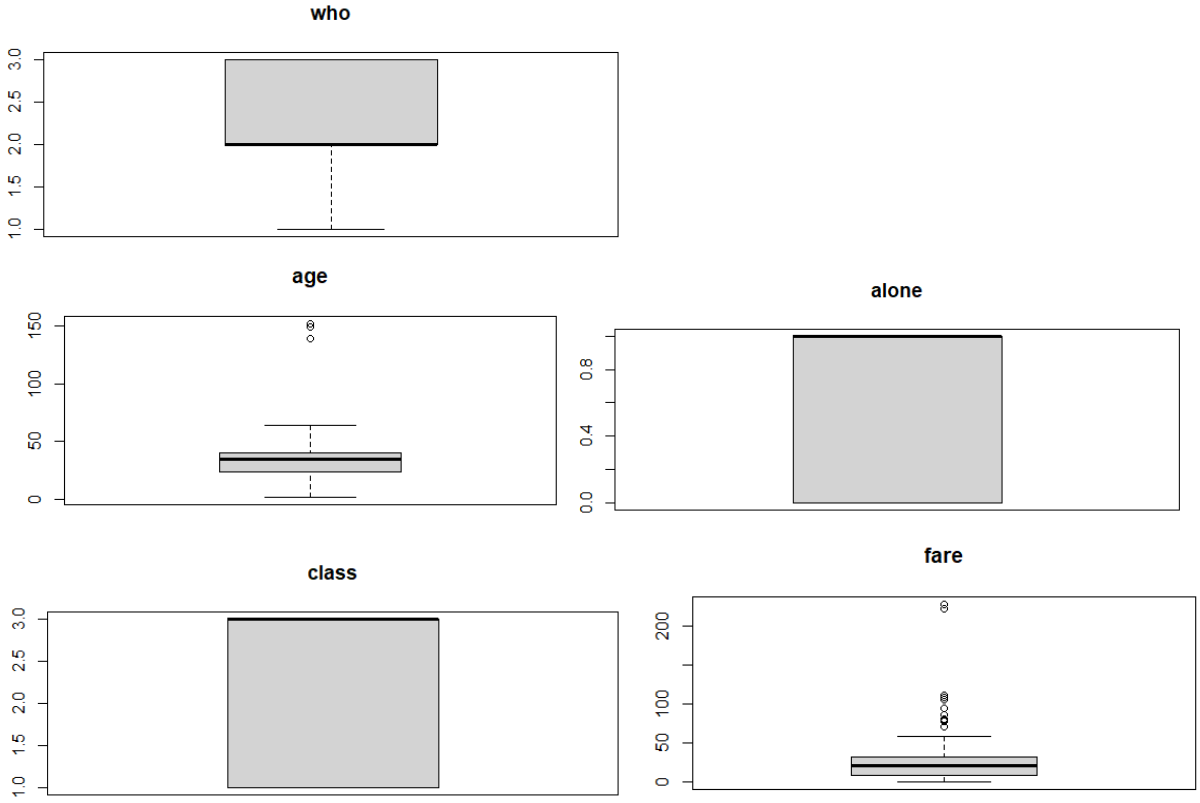
Description: Initially mean, mode and were found using mean(), median() and getMode() function used in sapply() function, then the values were combined in stat_values matrix using rbind(). A barplot has been drawn to visualize.

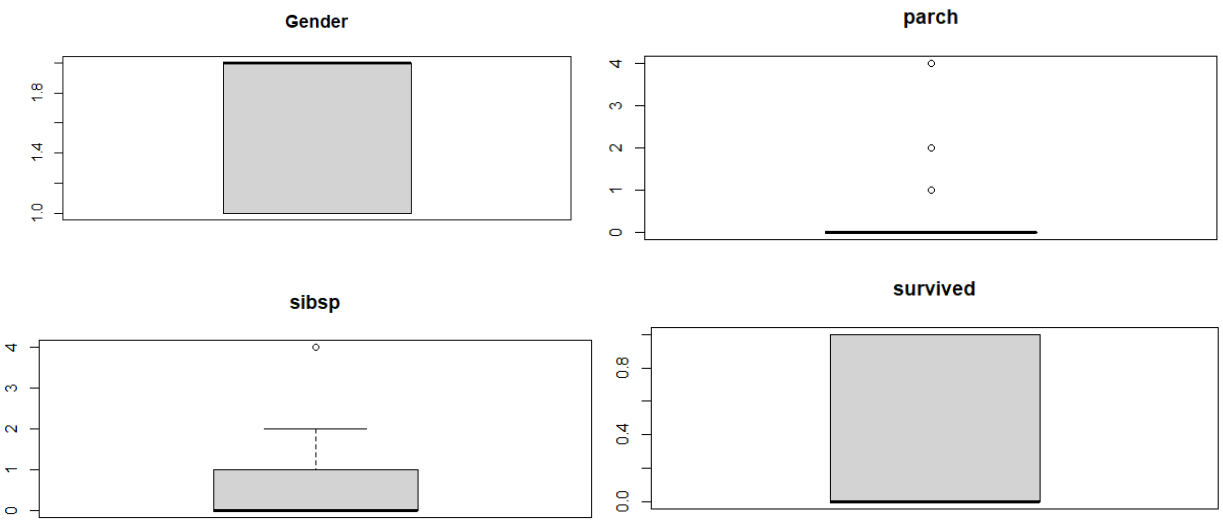
Find Outliers:

Code:

```
library(ggplot2)
boxplot(realData$Gender, main = "Gender")
boxplot(realData$age, main = "age" )
boxplot(realData$sibsp, main = "sibsp" )
boxplot(realData$parch, main = "parch" )
boxplot(realData$fare<-round(as.numeric(realData$fare)), main = "fare")
boxplot(realData$class, main = "class" )
boxplot(realData$who, main = "who")
boxplot(realData$alone, main = "alone" )
boxplot(realData$survived, main = "survived" )
```

Output:





Description: Using boxplot, we found outliers on “Gender" "age" "sibsp" "parch" "fare" "embarked" "class" "who" "alone" "survived"

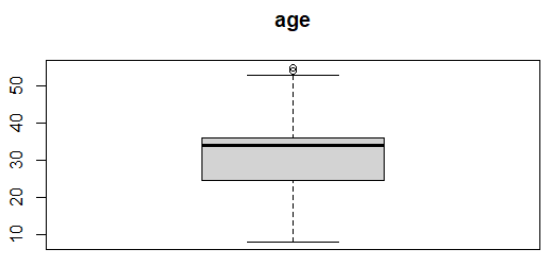
Removing Outliers:

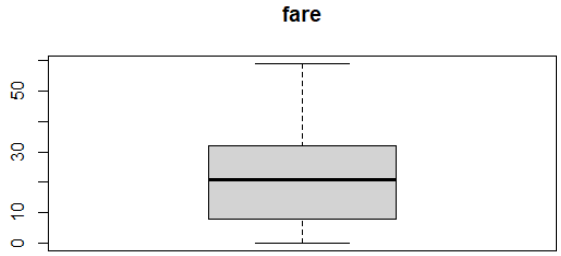
Code:

```
age_mean <- round(mean(cleanData2$age, na.rm = TRUE))
age_outliers <- boxplot.stats(cleanData2$age)$out
age_outliers
cleanData2$age[cleanData2$age %in% age_outliers] <- age_mean
boxplot(cleanData2$age, main = "age")
age_outliers <- boxplot.stats(cleanData2$age)$out
age_outliers

fare_mean <- round(mean(cleanData2$fare, na.rm = TRUE))
fare_outliers <- boxplot.stats(cleanData2$fare)$out
fare_outliers
cleanData2$fare[cleanData2$fare %in% fare_outliers] <- fare_mean
boxplot(cleanData2$fare, main = "fare")
fare_outliers <- boxplot.stats(cleanData2$fare)$out
fare_outliers
```

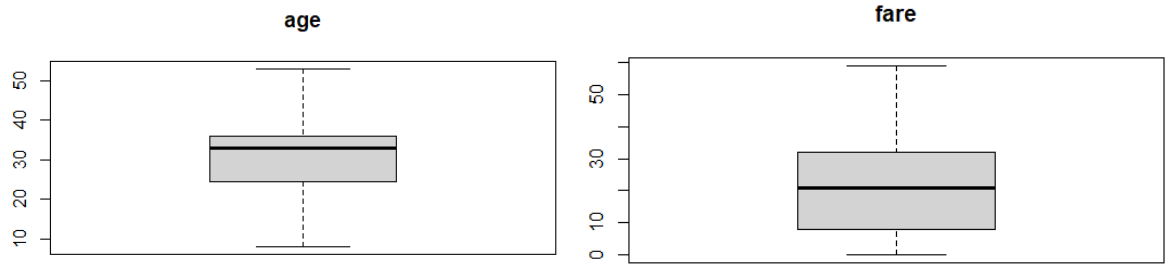
Output:

```
> age_mean <- round(mean(cleanData2$age, na.rm = TRUE))
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
[1] 139 152 149
> cleanData2$age[cleanData2$age %in% age_outliers] <- age_mean
> boxplot(cleanData2$age, main = "age")
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
[1]  2 64 62 62 60
> fare_mean <- round(mean(cleanData2$fare, na.rm = TRUE))
> fare_outliers <- boxplot.stats(cleanData2$fare)$out
> fare_outliers
 [1]  86 109  94 222 106  71 106 111 228  80 111  80  79  78
> cleanData2$fare[cleanData2$fare %in% fare_outliers] <- fare_mean
> boxplot(cleanData2$fare, main = "fare")
> fare_outliers <- boxplot.stats(cleanData2$fare)$out
> fare_outliers
numeric(0)
```





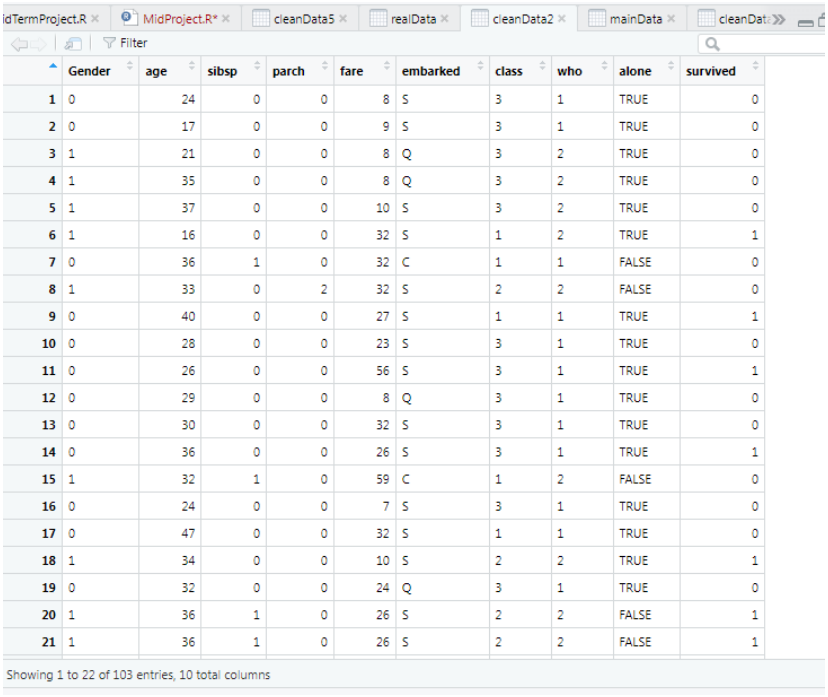
```
> age_mean <- round(mean(cleanData2$age, na.rm = TRUE))
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
[1]  2 64 62 62 60
> cleanData2$age[cleanData2$age %in% age_outliers] <- age_mean
> boxplot(cleanData2$age, main = "age")
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
[1] 54 55 54
> fare_mean <- round(mean(cleanData2$fare, na.rm = TRUE))
> fare_outliers <- boxplot.stats(cleanData2$fare)$out
> fare_outliers
numeric(0)
> cleanData2$fare[cleanData2$fare %in% fare_outliers] <- fare_mean
> boxplot(cleanData2$fare, main = "fare")
> fare_outliers <- boxplot.stats(cleanData2$fare)$out
> fare_outliers
numeric(0)
> age_mean <- round(mean(cleanData2$age, na.rm = TRUE))
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
[1] 54 55 54
> cleanData2$age[cleanData2$age %in% age_outliers] <- age_mean
> boxplot(cleanData2$age, main = "age")
> age_outliers <- boxplot.stats(cleanData2$age)$out
> age_outliers
numeric(0)
>
> fare_mean <- round(mean(cleanData2$fare, na.rm = TRUE))
> fare_outliers <- boxplot.stats(cleanData2$fare)$out
> fare_outliers
numeric(0)
```



Description: The outliers of age attribute were removed from age, but while plotting box plot, we found that some new outliers appeared. We removed the outliers again and boxplotted them. We can also ignore those outliers because the values are so near to our data. After removing outliers of fare using mean value, there no outlier remains.

Normalization:

realData table:



idTermProject.R | MidProject.R* | cleanData5 | realData | cleanData2 | mainData | cleanDat

Filter

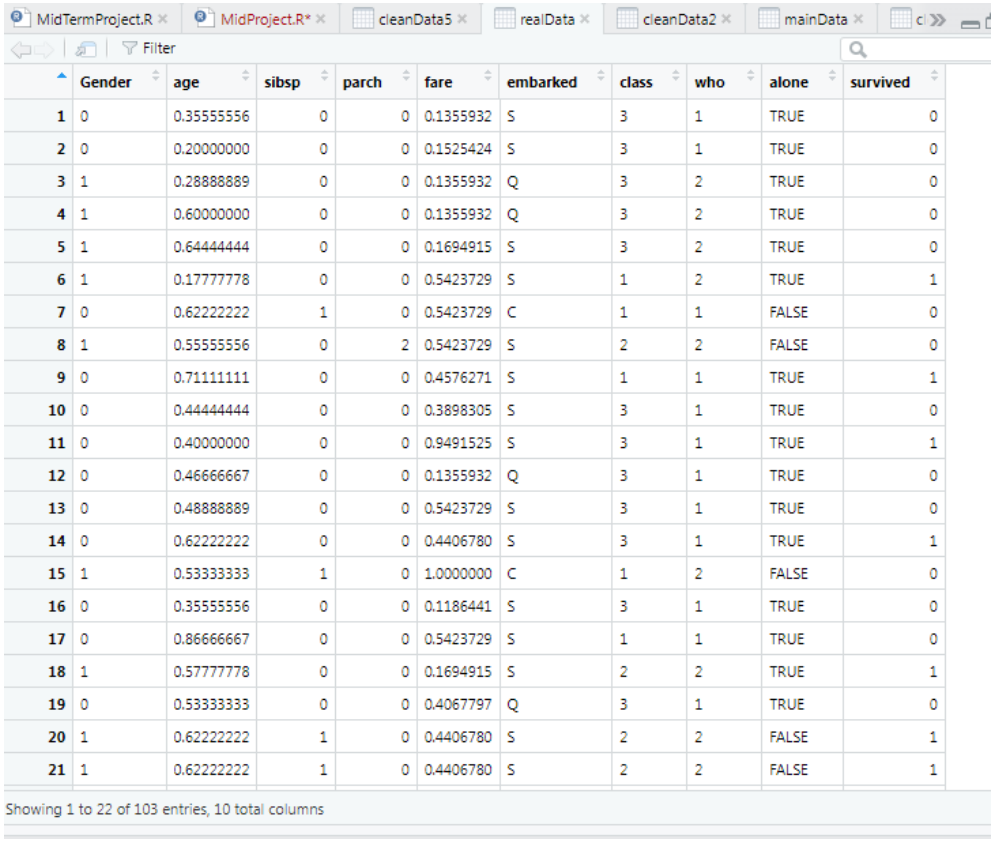
	Gender	age	sibsp	parch	fare	embarked	class	who	alone	survived
1	0	24	0	0	8	S	3	1	TRUE	0
2	0	17	0	0	9	S	3	1	TRUE	0
3	1	21	0	0	8	Q	3	2	TRUE	0
4	1	35	0	0	8	Q	3	2	TRUE	0
5	1	37	0	0	10	S	3	2	TRUE	0
6	1	16	0	0	32	S	1	2	TRUE	1
7	0	36	1	0	32	C	1	1	FALSE	0
8	1	33	0	2	32	S	2	2	FALSE	0
9	0	40	0	0	27	S	1	1	TRUE	1
10	0	28	0	0	23	S	3	1	TRUE	0
11	0	26	0	0	56	S	3	1	TRUE	1
12	0	29	0	0	8	Q	3	1	TRUE	0
13	0	30	0	0	32	S	3	1	TRUE	0
14	0	36	0	0	26	S	3	1	TRUE	1
15	1	32	1	0	59	C	1	2	FALSE	0
16	0	24	0	0	7	S	3	1	TRUE	0
17	0	47	0	0	32	S	1	1	TRUE	0
18	1	34	0	0	10	S	2	2	TRUE	1
19	0	32	0	0	24	Q	3	1	TRUE	0
20	1	36	1	0	26	S	2	2	FALSE	1
21	1	36	1	0	26	S	2	2	FALSE	1

Showing 1 to 22 of 103 entries, 10 total columns

Code:

```
realData<- cleanData2

min_age <- min(realData$age, na.rm = TRUE)
max_age <- max(realData$age, na.rm = TRUE)

realData$age <- (realData$age - min_age) / (max_age - min_age)

min_fare <- min(realData$fare, na.rm = TRUE)
max_fare <- max(realData$fare, na.rm = TRUE)

realData$fare <- (realData$fare - min_fare) / (max_fare - min_fare)
```

Output:



MidTermProject.R | MidProject.R* | cleanData5 | realData | cleanData2 | mainData

Filter

	Gender	age	sibsp	parch	fare	embarked	class	who	alone	survived
1	0	0.35555556	0	0	0.1355932	S	3	1	TRUE	0
2	0	0.20000000	0	0	0.1525424	S	3	1	TRUE	0
3	1	0.28888889	0	0	0.1355932	Q	3	2	TRUE	0
4	1	0.60000000	0	0	0.1355932	Q	3	2	TRUE	0
5	1	0.64444444	0	0	0.1694915	S	3	2	TRUE	0
6	1	0.17777778	0	0	0.5423729	S	1	2	TRUE	1
7	0	0.62222222	1	0	0.5423729	C	1	1	FALSE	0
8	1	0.55555556	0	2	0.5423729	S	2	2	FALSE	0
9	0	0.71111111	0	0	0.4576271	S	1	1	TRUE	1
10	0	0.44444444	0	0	0.3898305	S	3	1	TRUE	0
11	0	0.40000000	0	0	0.9491525	S	3	1	TRUE	1
12	0	0.46666667	0	0	0.1355932	Q	3	1	TRUE	0
13	0	0.48888889	0	0	0.5423729	S	3	1	TRUE	0
14	0	0.62222222	0	0	0.4406780	S	3	1	TRUE	1
15	1	0.53333333	1	0	1.0000000	C	1	2	FALSE	0
16	0	0.35555556	0	0	0.1186441	S	3	1	TRUE	0
17	0	0.86666667	0	0	0.5423729	S	1	1	TRUE	0
18	1	0.57777778	0	0	0.1694915	S	2	2	TRUE	1
19	0	0.53333333	0	0	0.4067797	Q	3	1	TRUE	0
20	1	0.62222222	1	0	0.4406780	S	2	2	FALSE	1
21	1	0.62222222	1	0	0.4406780	S	2	2	FALSE	1

Showing 1 to 22 of 103 entries, 10 total columns

Description: Normalized using Xnew=(X-Xmin)/(Xmax-Xmin)

Convert Numerical Attributes to Categorical:

Code:

```
outputData<- realData

outputData$Gender <- factor(outputData$Gender,
                            levels = c(0, 1),
                            labels = c("female", "male"))

outputData$who <- factor(outputData$who,
                         labels = c("child", "man", "woman"),
                         levels = c(0, 1, 2))
```

Output:

MidTermProject.R | MidProject.R* | outputData | cleanData5 | realData | cleanData2

Filter

	Gender	age	sibsp	parch	fare	embarked	class	who	alone	survived
1	female	0.35555556	0	0	0.1355932	S	3	man	TRUE	0
2	female	0.20000000	0	0	0.1525424	S	3	man	TRUE	0
3	male	0.28888889	0	0	0.1355932	Q	3	woman	TRUE	0
4	male	0.60000000	0	0	0.1355932	Q	3	woman	TRUE	0
5	male	0.64444444	0	0	0.1694915	S	3	woman	TRUE	0
6	male	0.17777778	0	0	0.5423729	S	1	woman	TRUE	1
7	female	0.62222222	1	0	0.5423729	C	1	man	FALSE	0
8	male	0.55555556	0	2	0.5423729	S	2	woman	FALSE	0
9	female	0.71111111	0	0	0.4576271	S	1	man	TRUE	1
10	female	0.44444444	0	0	0.3898305	S	3	man	TRUE	0
11	female	0.40000000	0	0	0.9491525	S	3	man	TRUE	1
12	female	0.46666667	0	0	0.1355932	Q	3	man	TRUE	0
13	female	0.48888889	0	0	0.5423729	S	3	man	TRUE	0

Showing 1 to 13 of 103 entries, 10 total columns

Description: Gender and who columns converted back to categorical attributes shown using factor() function.

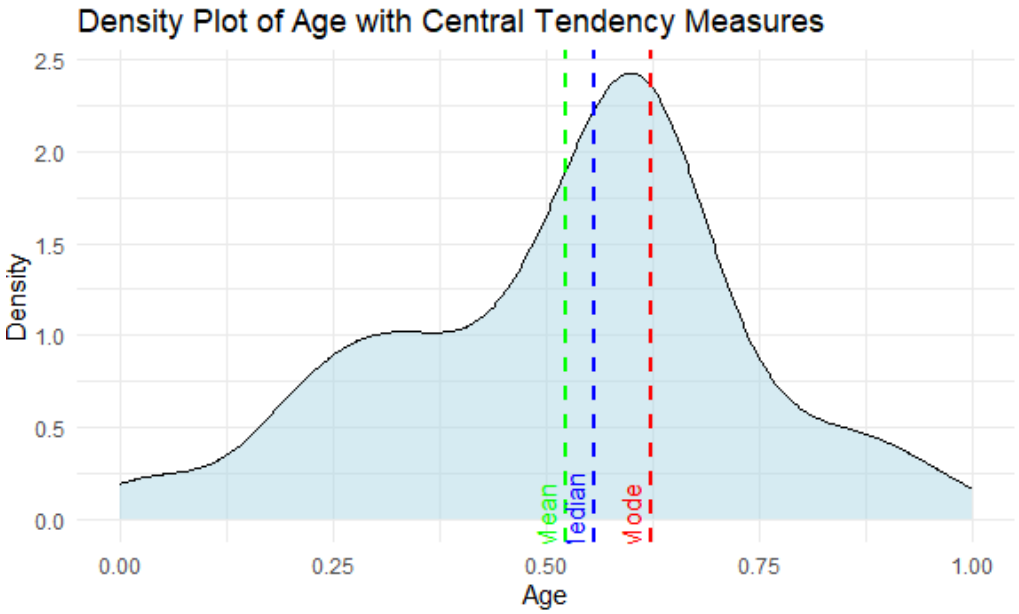
Measure of Central tendency:

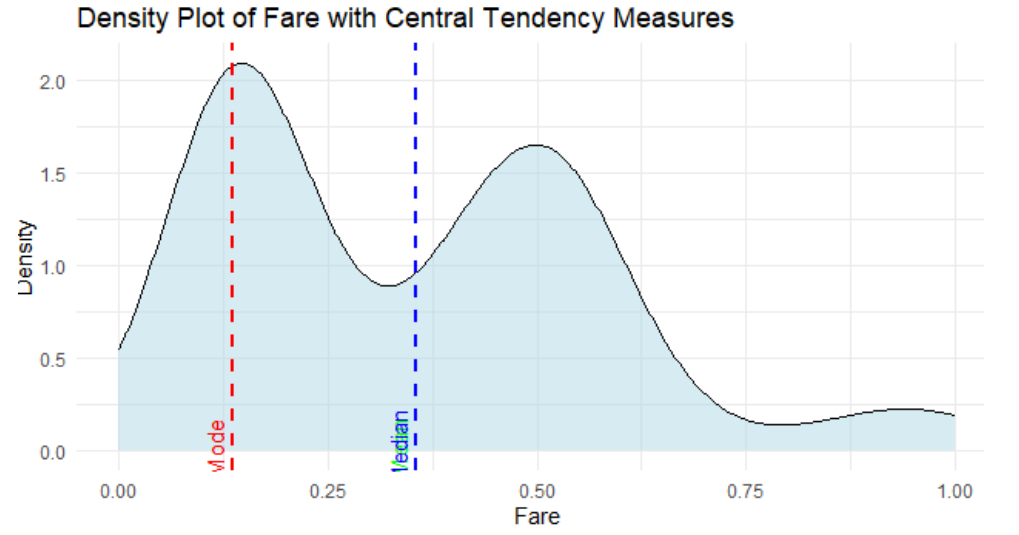
Code:

```
library(ggplot2)


mean_age <- mean(realData$age, na.rm = TRUE)

median_age <- median(realData$age, na.rm = TRUE)

mode_age <- as.numeric(names(sort(table(realData$age), decreasing = TRUE))[1])


plot_data <- data.frame(age = realData$age)
```

```
ggplot(plot_data, aes(x = age)) +
  geom_density(fill = "lightblue", alpha = 0.5) +
  geom_vline(xintercept = mean_age, col = "green", linetype = "dashed", size = 1) +  # Mean
  geom_vline(xintercept = median_age, col = "blue", linetype = "dashed", size = 1) + # Median
  geom_vline(xintercept = mode_age, col = "red", linetype = "dashed", size = 1) +    # Mode
  labs(title = "Density Plot of Age with Central Tendency Measures",
       x = "Age", y = "Density") +
  theme_minimal() +
  annotate("text", x = mean_age, y = 0.02, label = "Mean", col = "green", angle = 90, vjust = -0.5) +
  annotate("text", x = median_age, y = 0.02, label = "Median", col = "blue", angle = 90, vjust = -0.5) +
  annotate("text", x = mode_age, y = 0.02, label = "Mode", col = "red", angle = 90, vjust = -0.5)

mean_fare <- mean(realData$fare, na.rm = TRUE)
median_fare <- median(realData$fare, na.rm = TRUE)
mode_fare <- as.numeric(names(sort(table(realData$fare), decreasing = TRUE))[1])

plot_data <- data.frame(fare = realData$fare)
ggplot(plot_data, aes(x = fare)) +
  geom_density(fill = "lightblue", alpha = 0.5) +
  geom_vline(xintercept = mean_fare, col = "green", linetype = "dashed", size = 1) +  # Mean
  geom_vline(xintercept = median_fare, col = "blue", linetype = "dashed", size = 1) + # Median
  geom_vline(xintercept = mode_fare, col = "red", linetype = "dashed", size = 1) +    # Mode
  labs(title = "Density Plot of Fare with Central Tendency Measures",
       x = "Fare", y = "Density") +
  theme_minimal() +
  annotate("text", x = mean_fare, y = 0.02, label = "Mean", col = "green", angle = 90, vjust = -0.5) +
  annotate("text", x = median_fare, y = 0.02, label = "Median", col = "blue", angle = 90, vjust = -0.5) +
  annotate("text", x = mode_fare, y = 0.02, label = "Mode", col = "red", angle = 90, vjust = -0.5)
```

Output:





Description: Central tendency measured for age and fare column attribute.

Measure of spread:

Code:

```
iqr_age <- IQR(realData$age, na.rm = TRUE)

iqr_fare <- IQR(realData$fare, na.rm = TRUE)

cat("Interquartile Range (IQR) for Age: ", iqr_age, "\n")

cat("Interquartile Range (IQR) for Fare: ", iqr_fare, "\n")


sd_age <- sd(realData$age, na.rm = TRUE)

sd_fare <- sd(realData$fare, na.rm = TRUE)

cat("Standard Deviation for Age: ", sd_age, "\n")

cat("Standard Deviation for Fare: ", sd_fare, "\n")


var_age <- var(realData$age, na.rm = TRUE)

var_fare <- var(realData$fare, na.rm = TRUE)

cat("Variance for Age: ", var_age, "\n")

cat("Variance for Fare: ", var_fare, "\n")
```

Output:

```
> iqr_age <- IQR(realData$age, na.rm = TRUE)
> iqr_fare <- IQR(realData$fare, na.rm = TRUE)
> cat("Interquartile Range (IQR) for Age: ", iqr_age, "\n")
Interquartile Range (IQR) for Age:  0.2555556
> cat("Interquartile Range (IQR) for Fare: ", iqr_fare, "\n")
Interquartile Range (IQR) for Fare:  0.4067797
>
> sd_age <- sd(realData$age, na.rm = TRUE)
> sd_fare <- sd(realData$fare, na.rm = TRUE)
> cat("Standard Deviation for Age: ", sd_age, "\n")
Standard Deviation for Age:  0.2080641
> cat("Standard Deviation for Fare: ", sd_fare, "\n")
Standard Deviation for Fare:  0.2299984
>
> var_age <- var(realData$age, na.rm = TRUE)
> var_fare <- var(realData$fare, na.rm = TRUE)
> cat("Variance for Age: ", var_age, "\n")
Variance for Age:  0.04329068
> cat("Variance for Fare: ", var_fare, "\n")
Variance for Fare:  0.05289928
>
```

Description: Inter quartile range, Standard deviation and Variance for age and fare column.

Handling Imbalance data:

Code:

```
library(dplyr)
library(ROSE)

class_distribution <- table(realData$survived)
print(class_distribution)

if (class_distribution[1] > class_distribution[2]) {
  majority <- filter(realData, survived == 0)
  minority <- filter(realData, survived == 1)
} else {
  majority <- filter(realData, survived == 1)
  minority <- filter(realData, survived == 0)
}

set.seed(123)
oversampled_minority <- minority %>% sample_n(nrow(majority), replace = TRUE)
oversampled_data <- bind_rows(majority, oversampled_minority)

table(oversampled_data$survived)
```

Output:

```
> library(dplyr)
> library(ROSE)
>
> class_distribution <- table(realData$survived)
> print(class_distribution)

 0  1
65 38
>
> if (class_distribution[1] > class_distribution[2]) {
+   majority <- filter(realData, survived == 0)
+   minority <- filter(realData, survived == 1)
+ } else {
+   majority <- filter(realData, survived == 1)
+   minority <- filter(realData, survived == 0)
+ }
>
> set.seed(123)
> oversampled_minority <- minority %>% sample_n(nrow(majority), replace = TRUE)
>
> oversampled_data <- bind_rows(majority, oversampled_minority)
>
> table(oversampled_data$survived)

 0  1
65 65
>
```

Description: First checking if minority and majority exists and defined ‘0’ as majority class and ‘1’ as minority class. Then minority class ‘1’ is oversampled to balance with the majority class ‘0’