Olive Oil Analysis

Nasrin Sultana Nipa

2024-02-05

To install the tidyverse package, I have an error related to CRAN mirror, to solve that issue I set a CRAN mirror before installing the package.

Project Overview:

Data loading, combining, correlation plotting and statistical analysis (mean, median and fdifference of mean and median for each column.

Dataset Courtesy:

"Massart, D. L., Vandeginste, B. G. M., Buydens, L. M. C., de Jong, S., Lewi, P. J., Smeyers-Verbeke, J. (1998) Handbook of Chemometrics and Qualimetrics: Part B. Elsevier. Tables 35.1 and 35.4."

Installation:

To use the 'tidyverse' library, installing the package in RStudio using the command 'install.packages()'.

```
install.packages("tidyverse")
```

```
## Installing package into 'C:/Users/nasrin/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'tidyverse' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\nasrin\AppData\Local\Temp\Rtmp48FmIW\downloaded_packages
```

After installation, to include the library in script, we need to use library() function.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

Loading Datasets:

We have two data files to proceed:

i) olive_oil_sensory.csv //contains the sensory panel variables

ii) olive_oil_chemical.csv //contains the chemical panel variables

This dataset contains scores on 6 attributes from a sensory panel and measurements of 5 physico-chemical quality parameters on 16 olive oil samples. The first five oils are Greek, the next five are Italian and the last six are Spanish.

To load each of the CSV files into tables with the read_csv() function from tidyverse can be used.

```
# Load the first CSV file (olive_oil_sensory.csv)
sensory_data <- read_csv("C:\\Users\\nasrin\\Desktop\\spring 2024\\DS\\hw3\\olive_oil_sensory.csv")
```

```
## Rows: 16 Columns: 7
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): region
## dbl (6): s_yellow, s_green, s_brown, s_glossy, s_transp, s_syrup
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# Load the second CSV file (olive_oil_chemical.csv)
chemical_data <- read_csv("C:\\Users\\nasrin\\Desktop\\spring 2024\\DS\\hw3\\datasetchemical\\olive_oil
```

```
## Rows: 16 Columns: 6
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): region
## dbl (5): c_Acidity, c_Peroxide, c_K232, c_K270, c_DK
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Using spec() to retrieve the full column specification for datasets.

```
spec(sensory_data)
```

```
## cols(
##   region = col_character(),
##   s_yellow = col_double(),
##   s_green = col_double(),
##   s_brown = col_double(),
##   s_glossy = col_double(),
##   s_transp = col_double(),
##   s_syrup = col_double()
## )
```

```
spec(chemical_data)
```

```
## cols(
##   region = col_character(),
##   c_Acidity = col_double(),
##   c_Peroxide = col_double(),
##   c_K232 = col_double(),
##   c_K270 = col_double(),
##   c_DK = col_double()
## )
```

Two see the whole dataframe of two datasets, we can do the following:

```
print(sensory_data)
```

```
## # A tibble: 16 x 7
##    region s_yellow s_green s_brown s_glossy s_transp s_syrup
##    <chr>     <dbl>   <dbl>   <dbl>    <dbl>    <dbl>   <dbl>
##  1 G1         21.4    73.4    10.1     79.7     75.2    50.3
##  2 G2         23.4    66.3     9.8     77.8     68.7    51.7
##  3 G3         32.7    53.5     8.7     82.3     83.2    45.4
##  4 G4         30.2    58.3    12.2     81.1     77.1    47.8
##  5 G5         51.8    32.5     8       72.4     65.3    46.5
##  6 I1         40.7    42.9    20.1     67.7     63.5    52.2
##  7 I2         53.8    30.4    11.5     77.8     77.3    45.2
##  8 I3         26.4    66.5    14.2     78.7     74.6    51.8
##  9 I4         65.7    12.1    10.3     81.6     79.6    48.3
## 10 I5         45      31.9    28.4     75.7     72.9    52.8
## 11 S1         70.9    12.2    10.8     87.7     88.1    44.5
## 12 S2         73.5     9.7     8.3     89.9     89.7    42.3
## 13 S3         68.1    12      10.8     78.4     75.1    46.4
## 14 S4         67.6    13.9    11.9     84.6     83.8    48.5
## 15 S5         71.4    10.6    10.8     88.1     88.5    46.7
## 16 S6         71.4    10      11.4     89.5     88.5    47.2
```

```r
print(chemical_data)
```

```
## # A tibble: 16 x 6
##    region c_Acidity c_Peroxide c_K232 c_K270   c_DK
##    <chr>      <dbl>      <dbl>  <dbl>  <dbl>  <dbl>
##  1 G1          0.73      12.7    1.9   0.139  0.003
##  2 G2          0.19      12.3    1.68  0.116 -0.004
##  3 G3          0.26      10.3    1.63  0.116 -0.005
##  4 G4          0.67      13.7    1.70  0.168 -0.002
##  5 G5          0.52      11.2    1.54  0.119 -0.001
##  6 I1          0.26      18.7    2.12  0.142  0.001
##  7 I2          0.24      15.3    1.89  0.116  0
##  8 I3          0.3       18.5    1.91  0.125  0.001
##  9 I4          0.35      15.6    1.82  0.104  0
## 10 I5          0.19      19.4    2.22  0.158 -0.003
## 11 S1          0.15      10.5    1.52  0.116 -0.004
## 12 S2          0.16       8.14   1.53  0.106 -0.002
## 13 S3          0.27      12.5    1.56  0.093 -0.002
## 14 S4          0.16      11      1.57  0.094 -0.003
## 15 S5          0.24      10.8    1.33  0.085 -0.003
## 16 S6          0.3       11.4    1.42  0.093 -0.004
```

Combining Tables: To combine the two tables into a single table using left join with all 11 columns and 16 rows.

As, 'region' is the common column in two tables, so we can do the left join on that column and can create the new tables named "oil". After that we can omit that column. If we try to omit the 'region' column before or during left join operation, then an error will occur because the left join is done by using 'region' column.

```r
# Combine the two tables using left_join
oil <- left_join(sensory_data, chemical_data, by = "region")

# Omit the 'region' column from the combined table
oil <- select(oil, -region)

# Show the combined table without the 'region' column
print(oil)
```

```
## # A tibble: 16 x 11
##    s_yellow s_green s_brown s_glossy s_transp s_syrup c_Acidity c_Peroxide
##       <dbl>   <dbl>   <dbl>    <dbl>    <dbl>   <dbl>     <dbl>      <dbl>
##  1     21.4    73.4    10.1     79.7     75.2    50.3      0.73       12.7
##  2     23.4    66.3     9.8     77.8     68.7    51.7      0.19       12.3
##  3     32.7    53.5     8.7     82.3     83.2    45.4      0.26       10.3
##  4     30.2    58.3    12.2     81.1     77.1    47.8      0.67       13.7
##  5     51.8    32.5     8       72.4     65.3    46.5      0.52       11.2
##  6     40.7    42.9    20.1     67.7     63.5    52.2      0.26       18.7
##  7     53.8    30.4    11.5     77.8     77.3    45.2      0.24       15.3
```

```
##  8     26.4    66.5    14.2     78.7     74.6    51.8      0.3       18.5
##  9     65.7    12.1    10.3     81.6     79.6    48.3      0.35      15.6
## 10     45      31.9    28.4     75.7     72.9    52.8      0.19      19.4
## 11     70.9    12.2    10.8     87.7     88.1    44.5      0.15      10.5
## 12     73.5     9.7     8.3     89.9     89.7    42.3      0.16       8.14
## 13     68.1    12      10.8     78.4     75.1    46.4      0.27      12.5
## 14     67.6    13.9    11.9     84.6     83.8    48.5      0.16      11
## 15     71.4    10.6    10.8     88.1     88.5    46.7      0.24      10.8
## 16     71.4    10      11.4     89.5     88.5    47.2      0.3       11.4
## # i 3 more variables: c_K232 <dbl>, c_K270 <dbl>, c_DK <dbl>
```

Creating Correlation Plot: This is the process of how we can create a correlation plot for all numeric variables in the combined tibble "oil" while excluding the categorical variable "region".

This code installs and loads the corrplot package first, extracts the numeric variables excluding 'region', computes the correlation matrix using the cor() function, and then creates a correlation plot using corrplot with red and blue colors for correlation strength.

```r
# Install the corrplot package if not already installed
install.packages("corrplot")
```

```
## Installing package into 'C:/Users/nasrin/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'corrplot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\nasrin\AppData\Local\Temp\Rtmp48FmIW\downloaded_packages
```

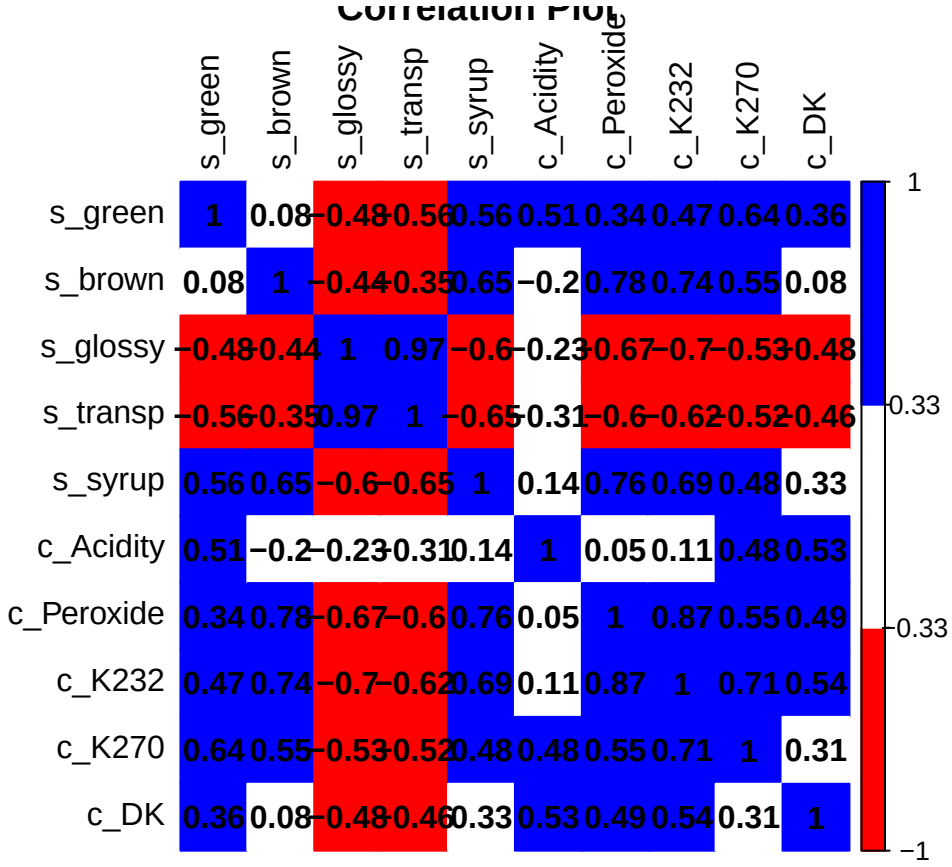
```r
# Load the corrplot package
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```r
# Extract numeric variables excluding 'region'
oil_vars <- colnames(oil)[-1]
numeric_data <- oil[oil_vars]

# Compute the correlation matrix
cor_matrix <- cor(numeric_data)

# Create a correlation plot using corrplot
corrplot(cor_matrix, method = "color", col = c("red", "white", "blue"), addCoef.col = "black", tl.col =
```



Reproducing Metrics:

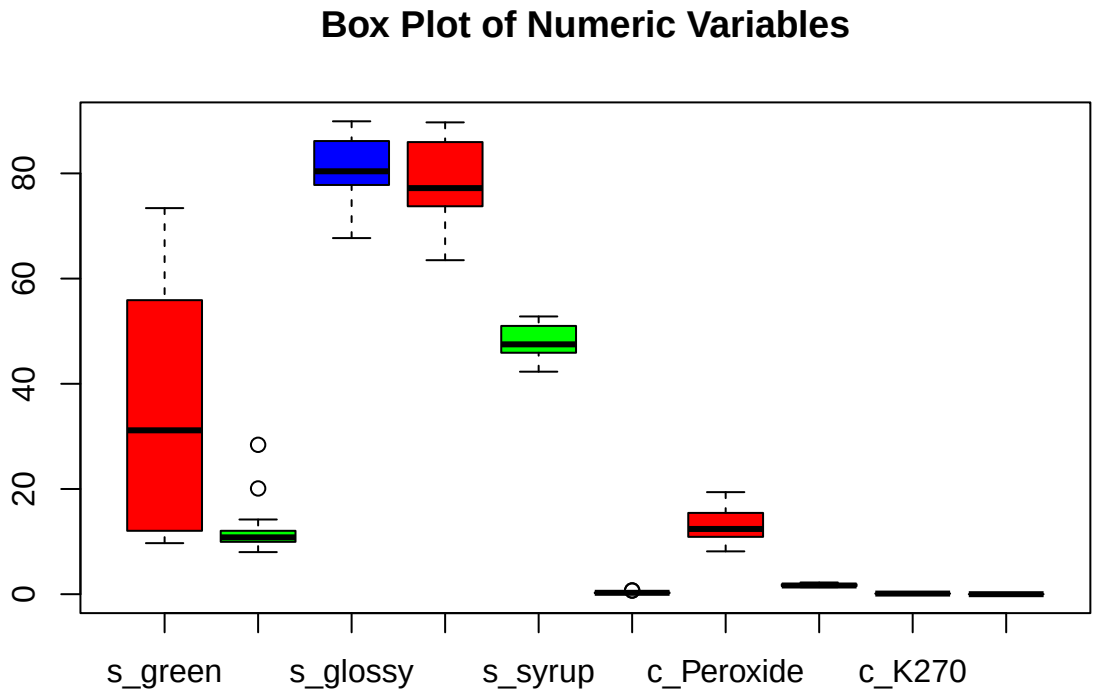
This code uses the summary() function to display summary statistics for each variable in the tibble "oil." Additionally, it creates a box plot for the numeric variables using boxplot(), with different colors for better visualization.

```
# Summary statistics
summary(oil)
```

```
##     s_yellow        s_green         s_brown         s_glossy
##  Min.   :21.40   Min.   : 9.70   Min.   : 8.00   Min.   :67.70
##  1st Qu.:32.08   1st Qu.:12.07   1st Qu.:10.03   1st Qu.:77.80
##  Median :52.80   Median :31.15   Median :10.80   Median :80.40
##  Mean   :50.88   Mean   :33.51   Mean   :12.33   Mean   :80.81
##  3rd Qu.:68.80   3rd Qu.:54.70   3rd Qu.:11.97   3rd Qu.:85.38
##  Max.   :73.50   Max.   :73.40   Max.   :28.40   Max.   :89.90
##     s_transp        s_syrup        c_Acidity        c_Peroxide
##  Min.   :63.50   Min.   :42.30   Min.   :0.1500   Min.   : 8.14
##  1st Qu.:74.17   1st Qu.:46.15   1st Qu.:0.1900   1st Qu.:10.95
##  Median :77.20   Median :47.50   Median :0.2600   Median :12.40
##  Mean   :78.19   Mean   :47.98   Mean   :0.3119   Mean   :13.25
##  3rd Qu.:84.88   3rd Qu.:50.65   3rd Qu.:0.3125   3rd Qu.:15.38
##  Max.   :89.70   Max.   :52.80   Max.   :0.7300   Max.   :19.40
##      c_K232          c_K270           c_DK
```

```
##  Min.   :1.331   Min.   :0.0850   Min.   :-0.00500
##  1st Qu.:1.536   1st Qu.:0.1015   1st Qu.:-0.00325
##  Median :1.653   Median :0.1160   Median :-0.00200
##  Mean   :1.708   Mean   :0.1181   Mean   :-0.00175
##  3rd Qu.:1.893   3rd Qu.:0.1285   3rd Qu.: 0.00000
##  Max.   :2.222   Max.   :0.1680   Max.   : 0.00300
```

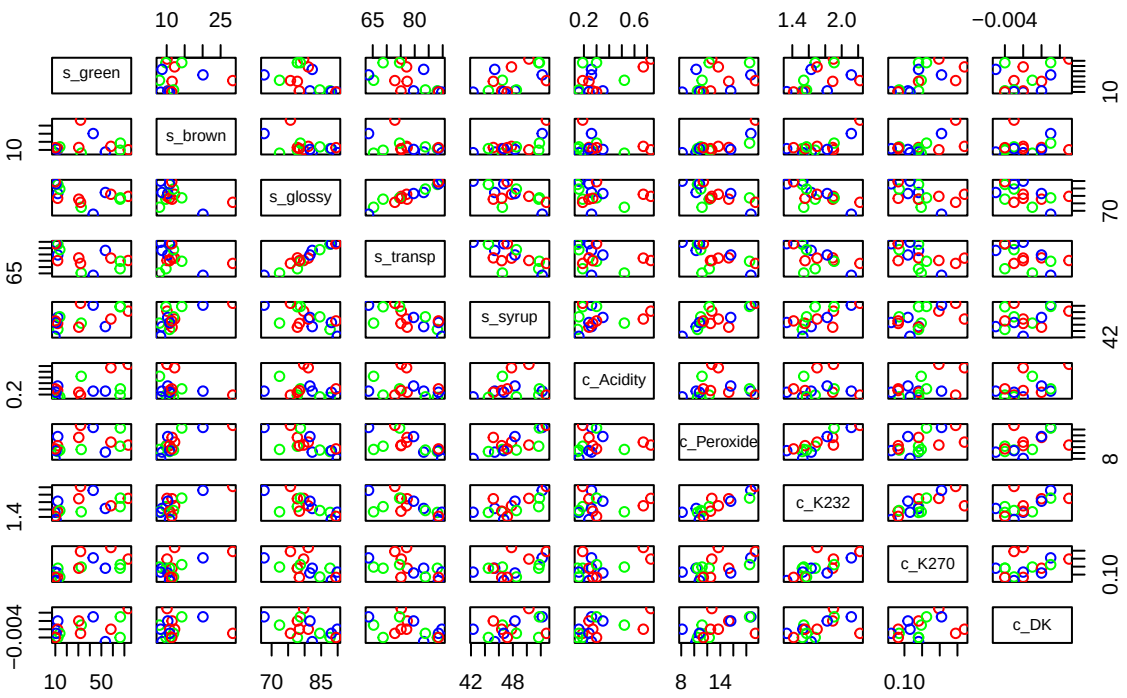
```
# Box plot for numeric variables
boxplot(oil[, oil_vars], col = c("red", "green", "blue"), main = "Box Plot of Numeric Variables", names
```



Pair Plotting: We can create a pair plot for the numeric variables in the combined tibble "oil" using the pairs() function.

This code uses the pairs() function to create a scatterplot matrix (pair plot) for the numeric variables in the tibble "oil".

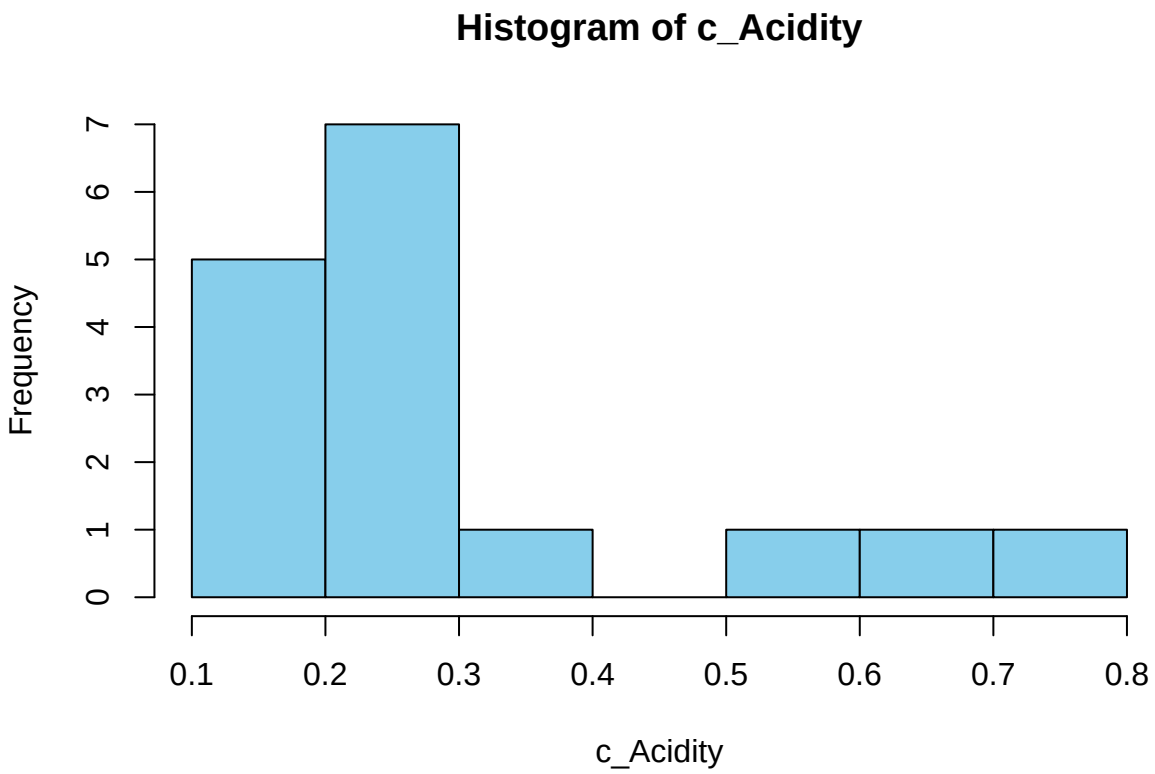
```
# Create a pair plot
pairs(oil[oil_vars], main = "Pair Plot of Numeric Variables", col = c("red", "green", "blue"))
```



Histogram:

Creating a histogram for the 'c_Acidity' column in the combined tibble "oil".

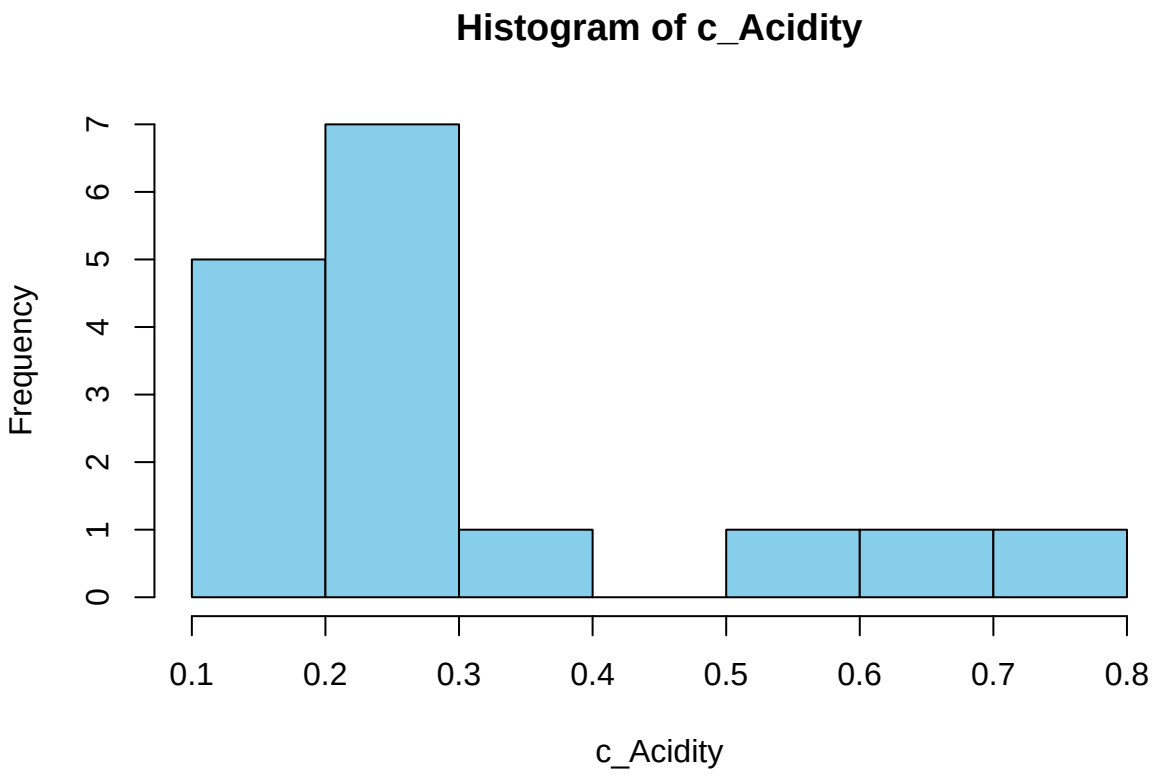
```
# Create a histogram for the 'c_Acidity' column
hist(oil$c_Acidity, col = "skyblue", main = "Histogram of c_Acidity", xlab = "c_Acidity")
```



The following code uses the hist() function to produce a histogram of the values in the 'c_Acidity' column, with a sky-blue color for better visualization.

This code uses as.numeric() to cast the 'c_Acidity' column to numeric before creating the histogram.

```
# Create a histogram for the 'c_Acidity' column after converting to numeric
hist(as.numeric(oil$c_Acidity), col = "skyblue", main = "Histogram of c_Acidity", xlab = "c_Acidity")
```



Compute the following:

Compute the mean value of each numeric column. (this will produce a vector of 11 values).

Compute the median values of each column.

Compute the difference between the mean and median for each column.

```r
# Compute the mean value of each numeric column
mean_values <- colMeans(oil)

# Compute the median values of each column
median_values <- apply(oil, 2, median)

# Compute the difference between the mean and median for each column
diff_mean_median <- mean_values - median_values

# Combine the results into a data frame
result_df <- data.frame(
  Variable = names(mean_values),
  Mean = mean_values,
  Median = median_values,
  Difference_Mean_Median = diff_mean_median
)
```

```r
# Print the result
print(result_df)
```

```
##              Variable       Mean  Median Difference_Mean_Median
## s_yellow     s_yellow 50.8750000 52.8000            -1.92500000
## s_green       s_green 33.5125000 31.1500             2.36250000
## s_brown       s_brown 12.3312500 10.8000             1.53125000
## s_glossy     s_glossy 80.8125000 80.4000             0.41250000
## s_transp     s_transp 78.1937500 77.2000             0.99375000
## s_syrup       s_syrup 47.9750000 47.5000             0.47500000
## c_Acidity   c_Acidity  0.3118750  0.2600             0.05187500
## c_Peroxide c_Peroxide 13.2525000 12.4000             0.85250000
## c_K232         c_K232  1.7082500  1.6535             0.05475000
## c_K270         c_K270  0.1181438  0.1160             0.00214375
## c_DK             c_DK -0.0017500 -0.0020             0.00025000
```