DAT320: Compulsory assignment 0

Group 1, Joel, Vegard, Artush

2023-09-26

Load the necessary libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(moments)
library(Metrics)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following objects are masked from 'package:Metrics':
##
##     precision, recall
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(stats)
library(nortest)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(raster)
```

```
## Loading required package: sp

## The legacy packages maptools, rgdal, and rgeos, underpinning the sp package,
## which was just loaded, will retire in October 2023.
## Please refer to R-spatial evolution reports for details, especially
## https://r-spatial.org/r/2023/05/15/evolution4.html.
## It may be desirable to make the sf package available;
## package maintainers should consider adding sf to Suggests:.
## The sp package is now running under evolution status 2
##      (status 2 uses the sf package in place of rgdal)

##
## Attaching package: 'raster'

## The following object is masked from 'package:dplyr':
##
##     select
```

Exercise 1

Task a)

```
# Load the ChickWeight dataset
data("ChickWeight", package = "datasets")

# Convert the Diet variable to a factor
ChickWeight$Diet <- as.factor(ChickWeight$Diet)

# Rename the levels of the Diet factor
levels(ChickWeight$Diet) <- c("A", "B", "C", "D")

# Print a summary of the dataset
summary(ChickWeight)
```

```
##      weight           Time           Chick     Diet
##  Min.   : 35.0   Min.   : 0.00   13     : 12   A:220
##  1st Qu.: 63.0   1st Qu.: 4.00   9      : 12   B:120
##  Median :103.0   Median :10.00   20     : 12   C:120
##  Mean   :121.8   Mean   :10.72   10     : 12   D:118
##  3rd Qu.:163.8   3rd Qu.:16.00   17     : 12
##  Max.   :373.0   Max.   :21.00   19     : 12
##                                  (Other):506
```

Task b)

```
# Group and summarize the data

diet <- ChickWeight %>%
  group_by(Diet, Chick) %>%
  summarise(maxTime = max(Time)
            )
```

```
## `summarise()` has grouped output by 'Diet'. You can override using the
## `.groups` argument.
```

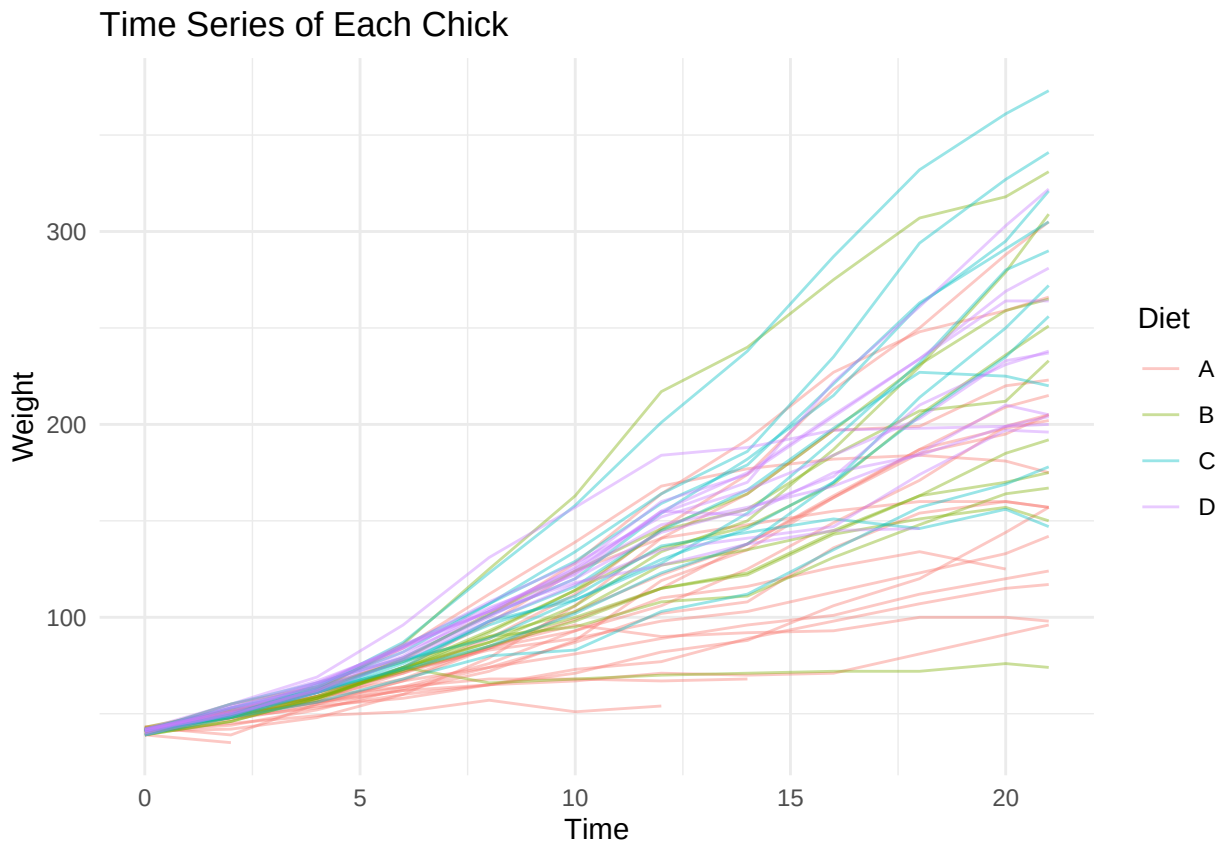
```
diet_summary <- diet %>%
  summarise(
    Number = n(),
    Min_Time = min(maxTime),
    Mean_Time = mean(maxTime),
    Max_Time = max(maxTime)
  ) %>%
  ungroup()

diet_summary
```

```
## # A tibble: 4 x 5
##   Diet  Number Min_Time Mean_Time Max_Time
##   <fct>  <int>    <dbl>     <dbl>    <dbl>
## 1 A         20        2      19.2       21
## 2 B         10       21      21         21
## 3 C         10       21      21         21
## 4 D         10       18      20.7       21
```

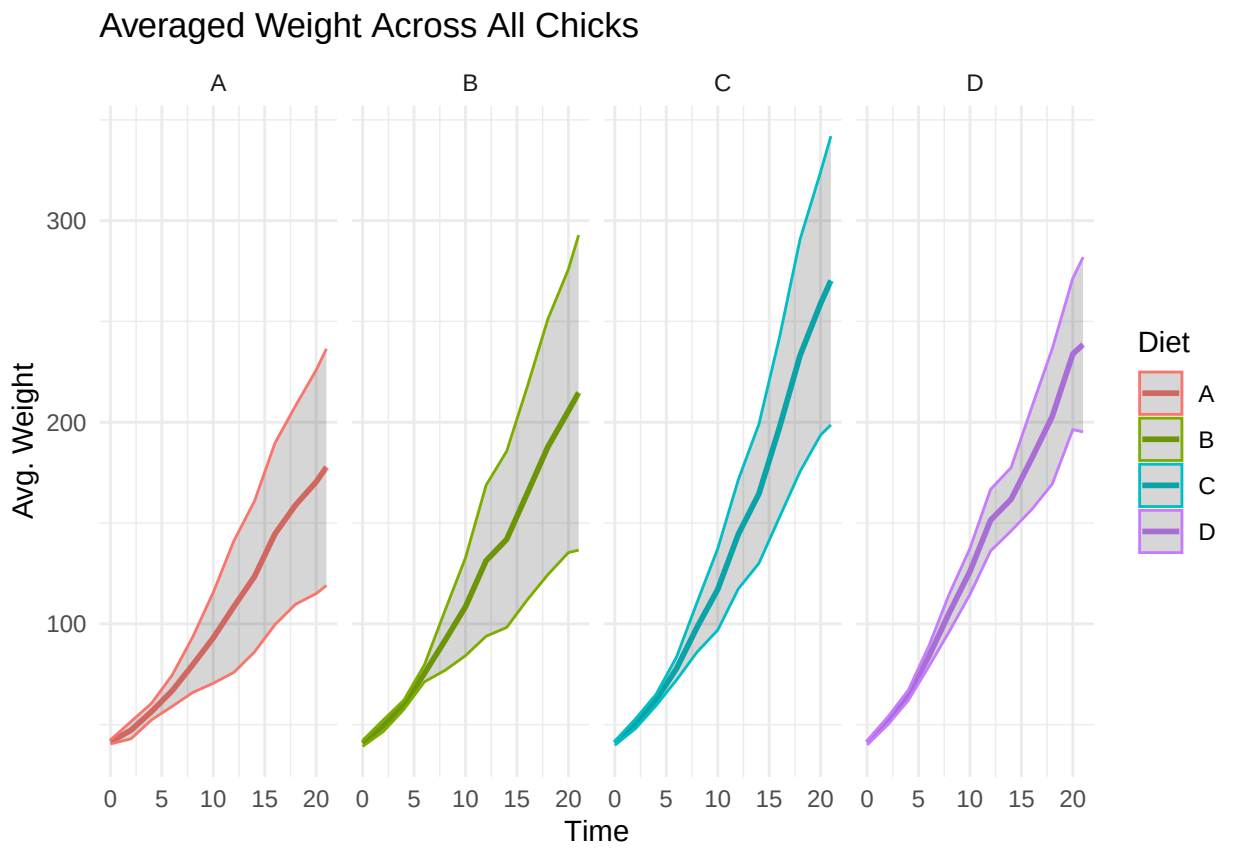
Task c)

```
# Load the ChickWeight dataset
data("ChickWeight", package = "datasets")
ChickWeight$Diet <- as.factor(ChickWeight$Diet)
levels(ChickWeight$Diet) <- c("A", "B", "C", "D")

# Compute summary statistics for each time point and Diet
summary_stats <- ChickWeight %>%
  group_by(Time, Diet) %>%
  summarise(
    mean_weight = mean(weight),
    sd_weight = sd(weight),
    n = n(),
    .groups = "drop" # Drop the grouping

  ) %>%
  mutate(
    se_weight = sd_weight / sqrt(n),
```

```
    lower = mean_weight - sd_weight,
    upper = mean_weight + sd_weight
  )

# Plot individual chick time series
ggplot(ChickWeight, aes(x = Time, y = weight, color = Diet, group = Chick)) +
  geom_line(alpha = 0.4) +
  ggtitle("Time Series of Each Chick") +
  xlab("Time") +
  ylab("Weight") +
  theme_minimal()
```




```
# Plot averaged weight across all chicks per time point
ggplot(summary_stats, aes(x = Time, y = mean_weight, color = Diet)) +
  geom_line(linewidth = 1) +
  geom_ribbon(aes(ymin = lower, ymax = upper), alpha = 0.2) +
  ggtitle("Averaged Weight Across All Chicks") +
  xlab("Time") +
  ylab("Avg. Weight") +
  theme_minimal() +
  facet_grid(. ~ Diet)
```



Task d)

Finally, interpret the results: Based on your analyses, what can we say about the influence of the different diets on the weight gain of chicks?

Based on our analysis we can see that diet C has the largest chick and average value, especially on the last day. Diet A Had the lowest weight gain in the 21 days. The diets C and B had the largest variation on weight between the chicks. For obtaining the largest chick in an 21 day span, we would recommend the diet C.

Exercise 2

Task a)

```
# Load the dataset
data <- read.csv("fish.csv", sep = ';')

# Get the names of the variables
variable_names <- colnames(data)

# Create a data frame to store the summary statistics
summary_stats <- data.frame(Variable = character(0),
                            Mean = numeric(0),
```

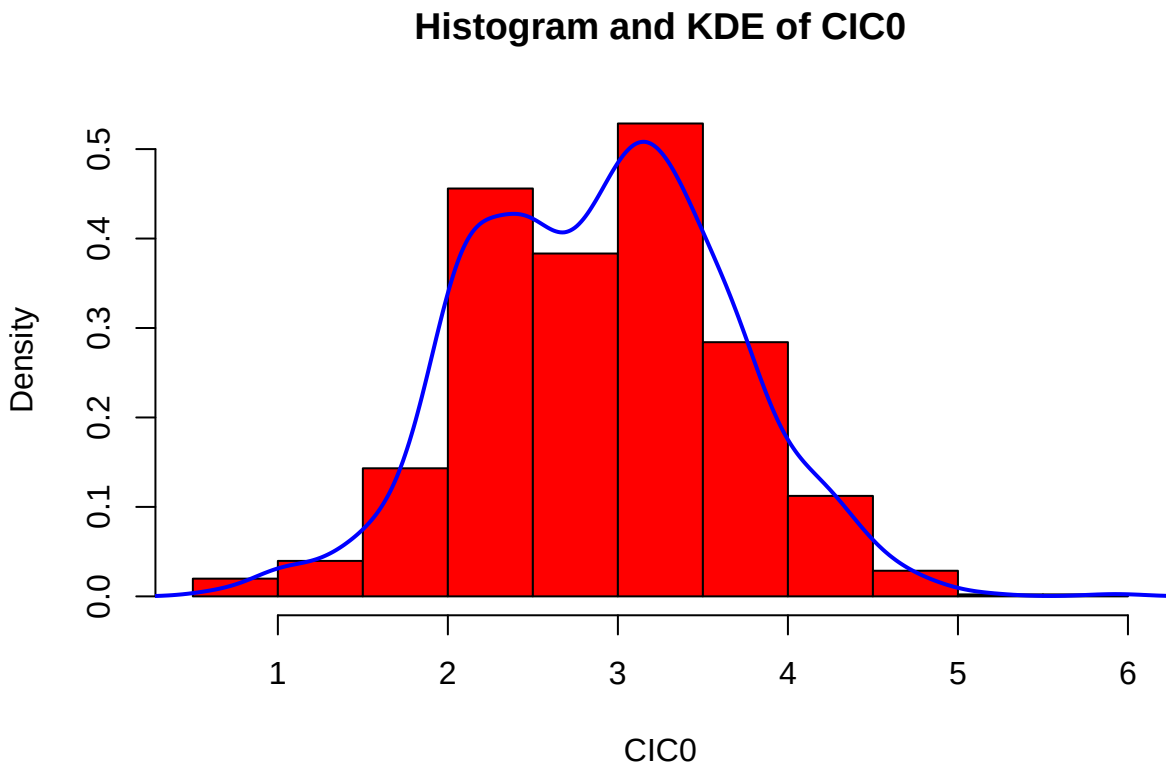
```
                            Variance = numeric(0),
                            Skewness = numeric(0),
                            Min = numeric(0),
                            Max = numeric(0))

# Loop through each variable and compute statistics and plot histograms
for (i in 1:length(variable_names)) {
  variable <- variable_names[i]

  # Compute statistics
  mean_val <- mean(data[[variable]])
  variance_val <- var(data[[variable]])
  skewness_val <- skewness(data[[variable]])
  min_val <- min(data[[variable]])
  max_val <- max(data[[variable]])

  # Append the results to the summary data frame
  summary_stats <- rbind(summary_stats,
                         data.frame(Variable = variable,
                                    Mean = mean_val,
                                    Variance = variance_val,
                                    Skewness = skewness_val,
                                    Min = min_val,
                                    Max = max_val))
}

# Print the summary statistics
print(summary_stats)
```

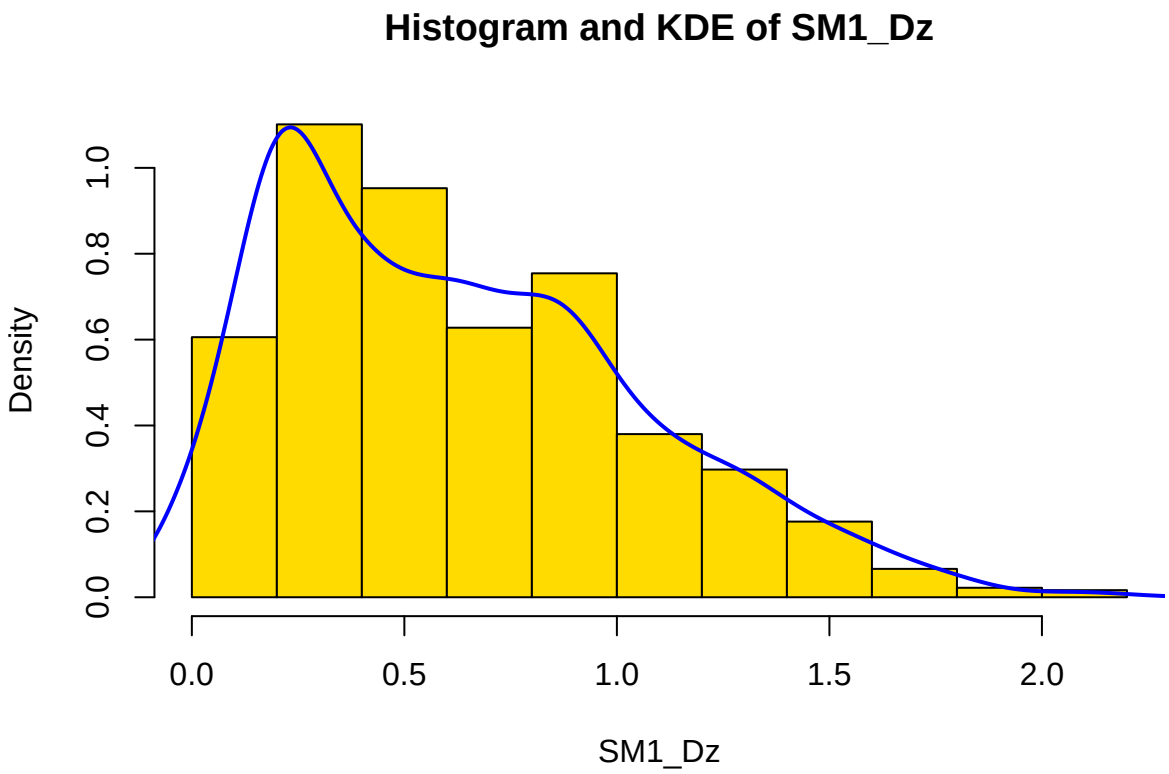
```
##   Variable      Mean  Variance    Skewness    Min   Max
## 1     CIC0 2.8981289 0.5716698  0.04538274  0.667 5.926
## 2   SM1_Dz 0.6284681 0.1835772  0.69394123  0.000 2.171
## 3   GATS1i 1.2935914 0.1554746  0.72191215  0.396 2.920
## 4    NdsCH 0.2290749 0.3664305  3.39519403  0.000 4.000
## 5    NdssC 0.4856828 0.7418014  2.23538970  0.000 6.000
## 6    MLOGP 2.1092852 2.0540072 -0.03513315 -2.884 6.515
## 7     LC50 4.0644306 2.1190580  0.25172172  0.053 9.612
```

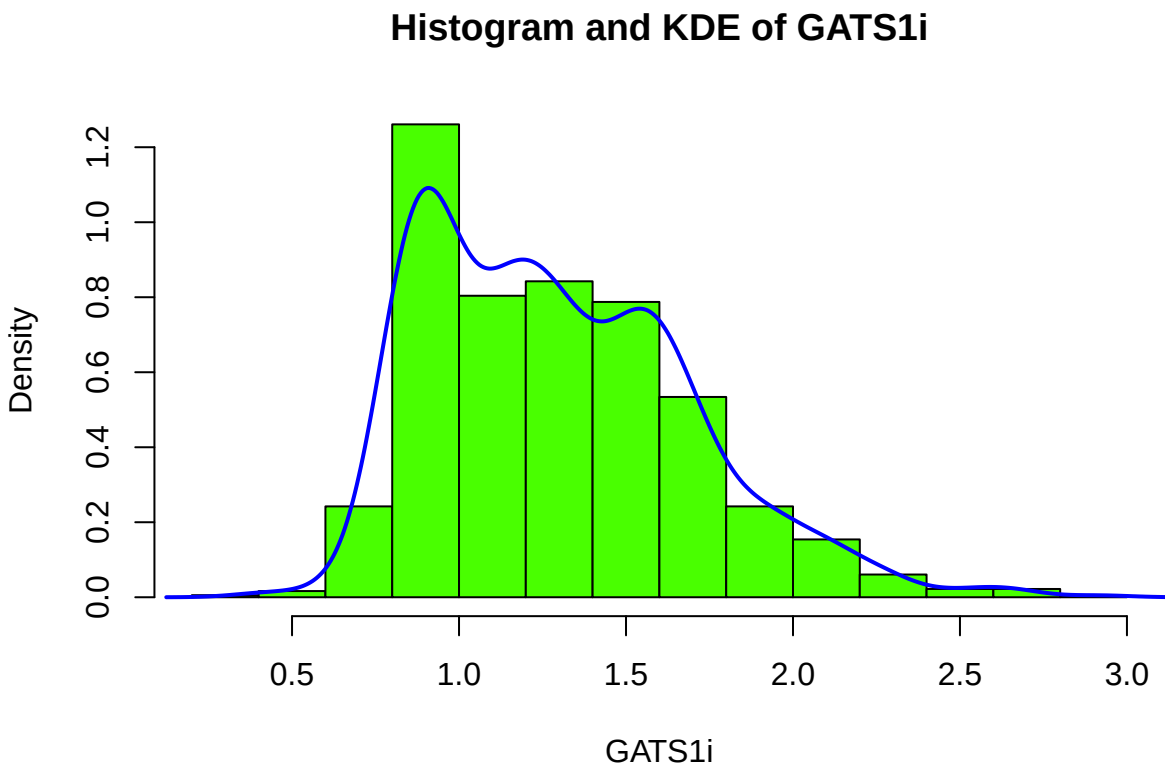
Task a (histogram)

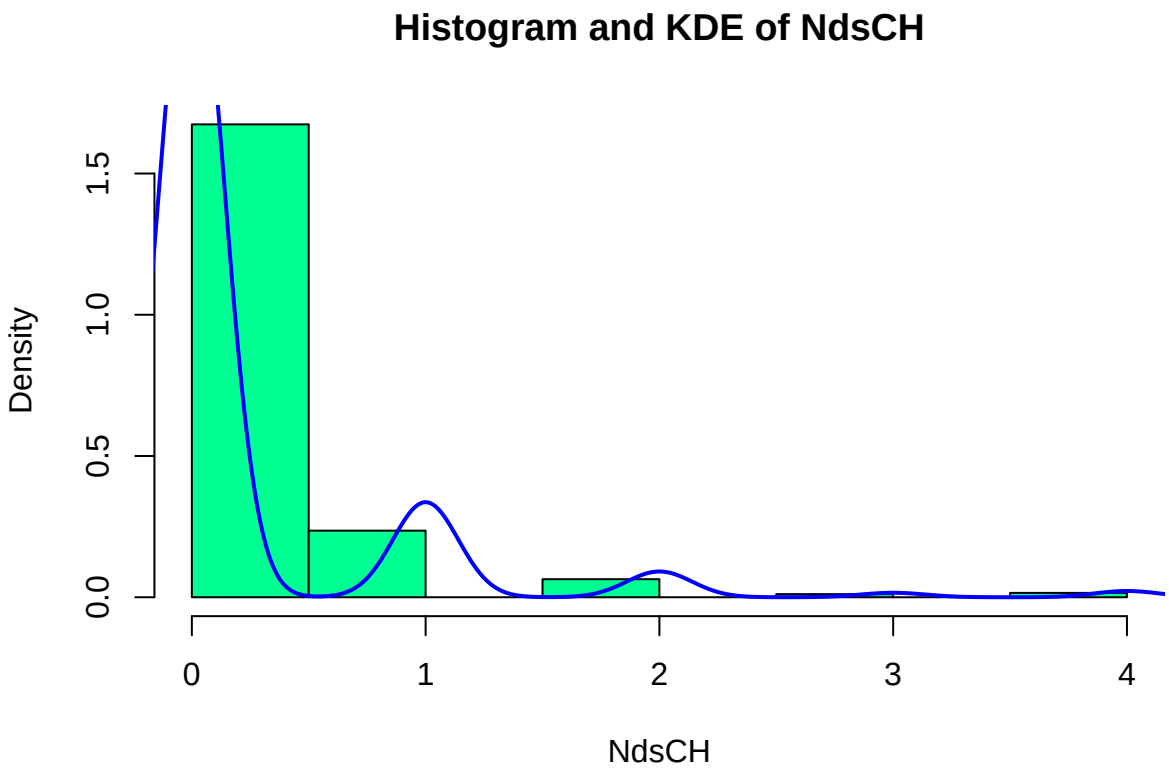
```
# Get the names of the variables
variable_names <- colnames(data)

# Define a color palette with different colors for each variable
color_palette <- rainbow(length(variable_names))

# Create a function for plotting histograms with KDE
plot_hist_kde <- function(variable) {
  hist_data <- data[[variable]]
  hist_title <- paste("Histogram and KDE of", variable)
```

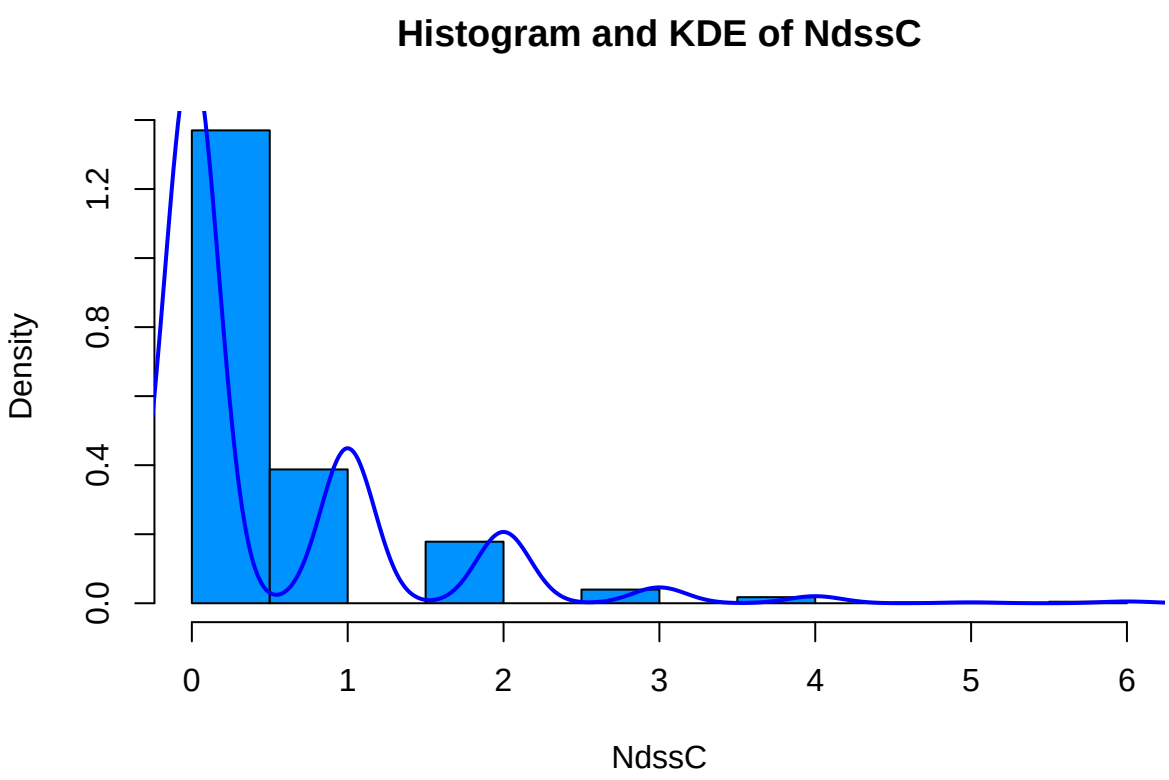
```r
  hist(hist_data, prob = TRUE, xlab = variable, main = hist_title, col = color_palette[i])
  lines(density(hist_data), lwd = 2, col = "blue")  # Adjust line color as needed
}

# Loop through each variable and create plots
for (i in 1:length(variable_names)) {
  variable <- variable_names[i]
  plot_hist_kde(variable)
}
```

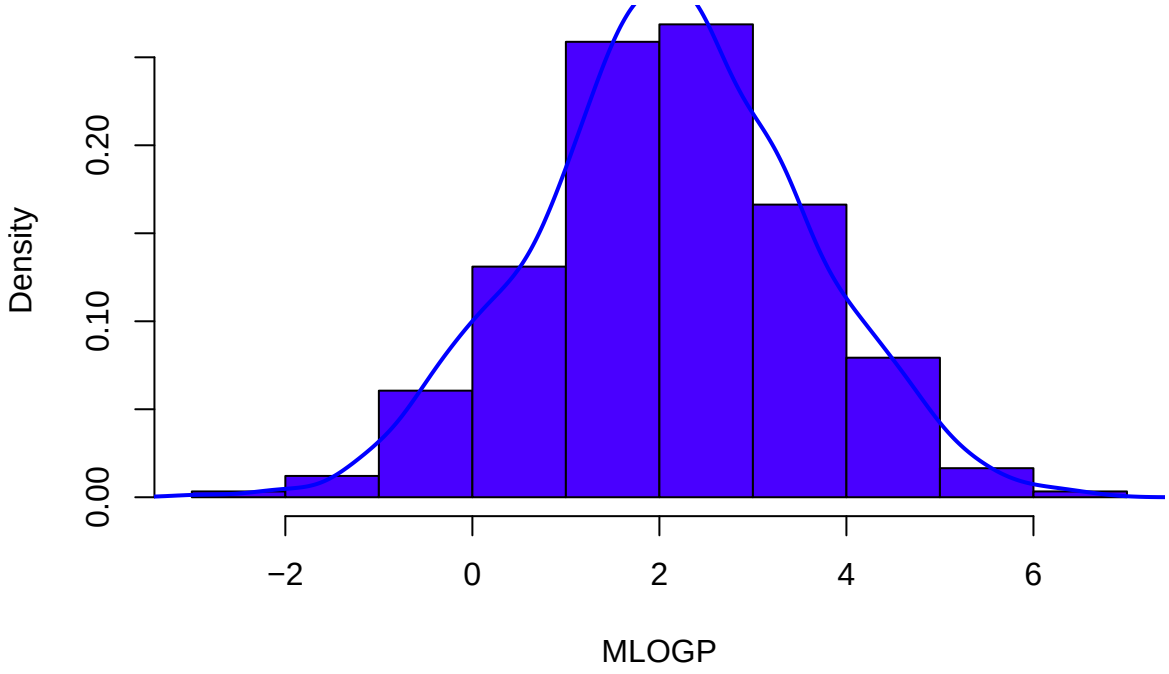


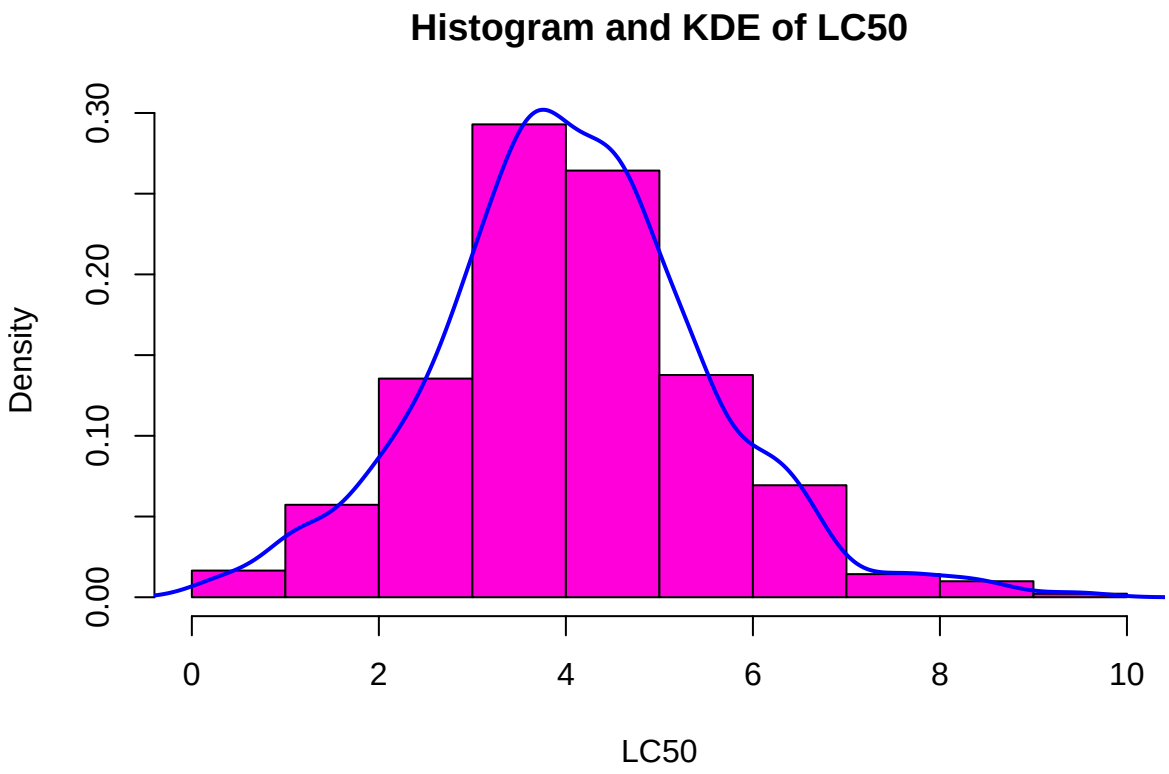

Histogram and KDE of SM1_Dz
Density
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
2.0
SM1_Dz


Histogram and KDE of GATS1i
Density
0.0
0.2
0.4
0.6
0.8
1.0
1.2
0.5
1.0
1.5
2.0
2.5
3.0
GATS1i


Histogram and KDE of NdsCH
Density
0.0
0.5
1.0
1.5
0
1
2
3
4
NdsCH


Histogram and KDE of NdssC
Density
0.0
0.4
0.8
1.2
0
1
2
3
4
5
6
NdssC


Histogram and KDE of MLOGP
Density
0.00
0.10
0.20
−2
0
2
4
6
MLOGP



Task a (shapiro test)

```
# Define a function to perform Shapiro-Wilk test on a variable
shapiro_test <- function(variable) {
  shapiro_result <- shapiro.test(variable)
  return(shapiro_result)
}

# Apply the Shapiro-Wilk test to each variable in the data
results <- lapply(data, shapiro_test)

# Print the results
for (i in 1:length(results)) {
  variable_name <- names(results)[i]
  p_value <- results[[i]]$p.value
  if (p_value <= 0.05) {
    cat(variable_name, "is not normally distributed (Shapiro-Wilk p-value:", p_value, ")\n")
  } else {
    cat(variable_name, "appears to be normally distributed (Shapiro-Wilk p-value:", p_value, ")\n")
  }
}
```

```
## CIC0 is not normally distributed (Shapiro-Wilk p-value: 0.0171922 )
## SM1_Dz is not normally distributed (Shapiro-Wilk p-value: 4.598798e-18 )
```

```
## GATS1i is not normally distributed (Shapiro-Wilk p-value: 4.966534e-16 )
## NdsCH is not normally distributed (Shapiro-Wilk p-value: 1.084519e-46 )
## NdssC is not normally distributed (Shapiro-Wilk p-value: 1.00963e-40 )
## MLOGP appears to be normally distributed (Shapiro-Wilk p-value: 0.7361579 )
## LC50 is not normally distributed (Shapiro-Wilk p-value: 4.586284e-05 )
```

Which family of probability distributions could be used to model each of the given variables?

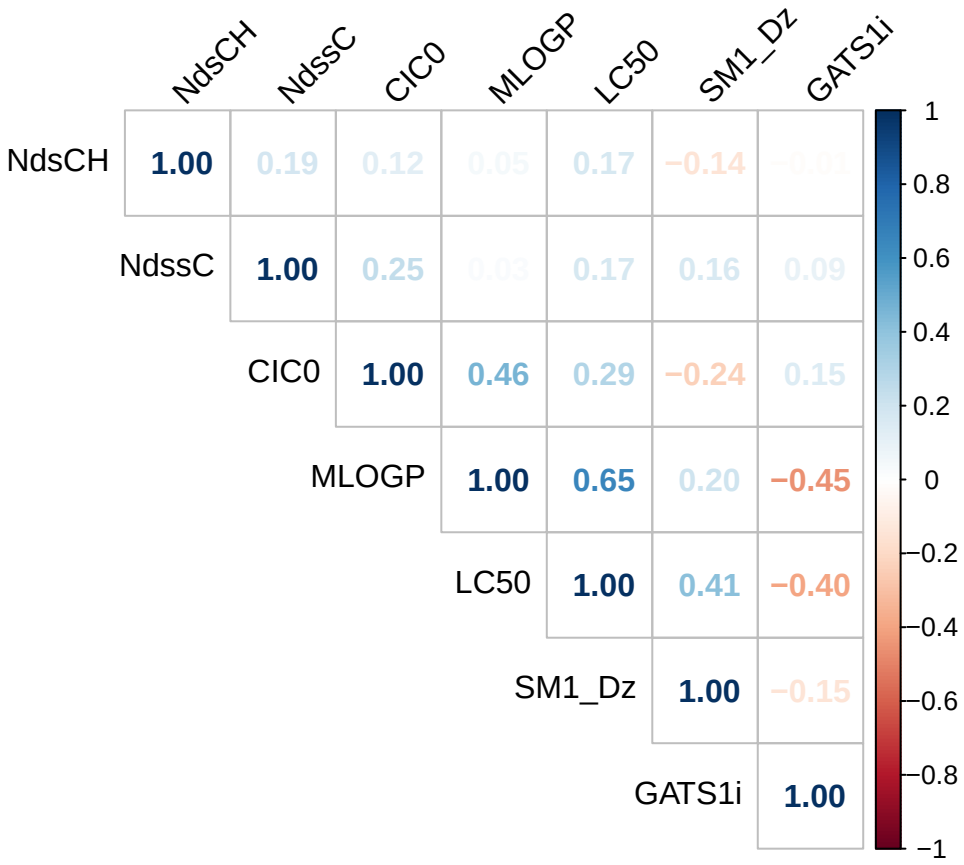
Based on visual inspections of the histograms and KDE, we can conclude that only MGLOP is normally distributed. We can see that CIC0 and LC50 follows closely a normal distribution, while SM1_Dz and GATS1i looks like log normal distribution. NdsCH and NdssC are skewed and binomial. We can also confirm this answer from the shapiro test.

Task b)

```
# Making a correlation matrix of the fisher data and printing
res = cor(data)
print(res)
```

```
##              CIC0     SM1_Dz      GATS1i       NdsCH      NdssC       MLOGP
## CIC0    1.0000000 -0.2353605  0.14762196  0.12134073 0.24663904  0.46386714
## SM1_Dz -0.2353605  1.0000000 -0.14596719 -0.14140138 0.16317892  0.20066284
## GATS1i  0.1476220 -0.1459672  1.00000000 -0.01065656 0.09240977 -0.45073916
## NdsCH   0.1213407 -0.1414014 -0.01065656  1.00000000 0.18816360  0.04861995
## NdssC   0.2466390  0.1631789  0.09240977  0.18816360 1.00000000  0.02849947
## MLOGP   0.4638671  0.2006628 -0.45073916  0.04861995 0.02849947  1.00000000
## LC50    0.2918543  0.4108932 -0.39796469  0.17200377 0.17238970  0.65166403
##              LC50
## CIC0    0.2918543
## SM1_Dz  0.4108932
## GATS1i -0.3979647
## NdsCH   0.1720038
## NdssC   0.1723897
## MLOGP   0.6516640
## LC50    1.0000000
```

```
# Heatmap of the correlation matrix
corrplot(res, type = "upper", order = "AOE", method = 'number',
         tl.col = "black", tl.srt = 45)
```



Explain what you see (you may disregard absolute correlations below the level of 0.4)! Which consequences between input variables may correlations have in a predictive model?

The first thing we notice is the high correlation between MLOGP and LC50, this molecular property has a direct effect on the toxicity level. Strongly interrelated predictor variables may result in collinearity problems, which can significantly elevate model variance, particularly within regression analysis.

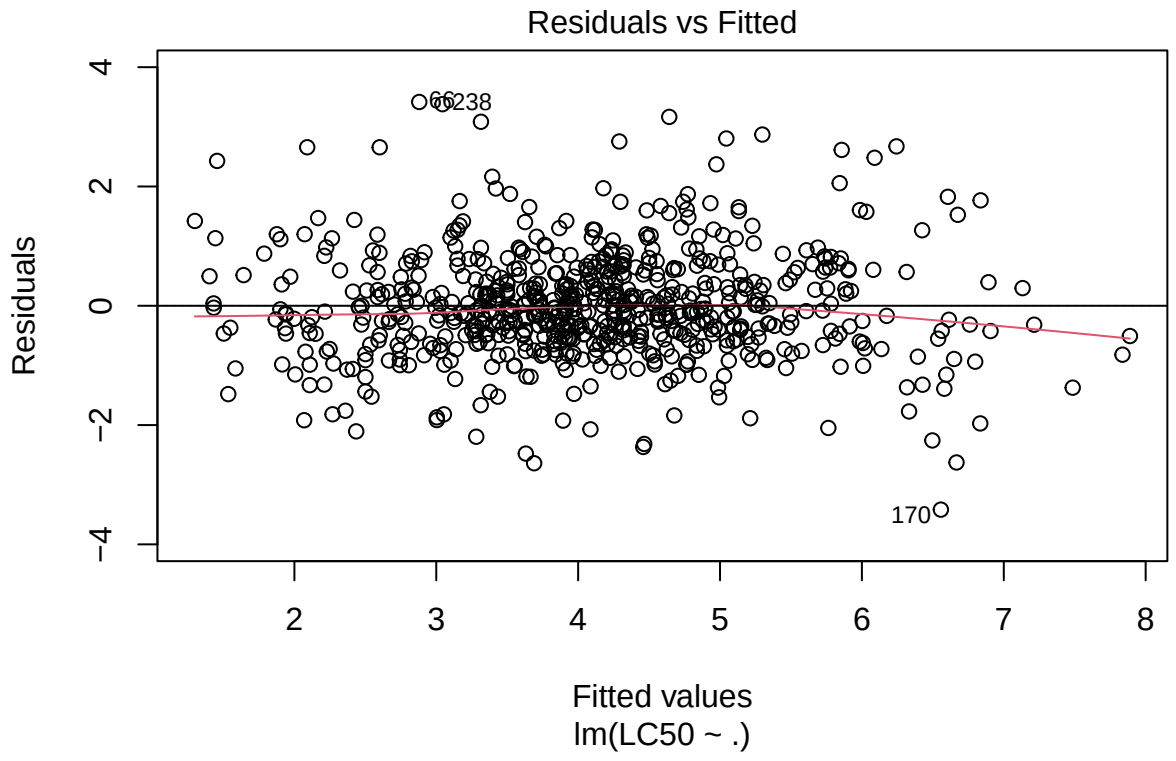
Task c)

```
#make this example reproducible
set.seed(1)

#use 75% of dataset as training set and 25% as test set
sample <- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.75,0.25))
train  <- data[sample, ]
test   <- data[!sample, ]

# Linear regression model
model = lm(formula=LC50 ~ ., data=train)

summary(model)
```

```
##
## Call:
## lm(formula = LC50 ~ ., data = train)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4176 -0.5049 -0.0737  0.5070  3.4169
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.99744    0.20271   9.854  < 2e-16 ***
## CIC0         0.45176    0.06806   6.638 6.54e-11 ***
## SM1_Dz       1.36131    0.09648  14.110  < 2e-16 ***
## GATS1i      -0.79262    0.11406  -6.949 8.66e-12 ***
## NdsCH        0.43957    0.06073   7.238 1.24e-12 ***
## NdssC        0.03284    0.04483   0.732    0.464
## MLOGP        0.38714    0.03760  10.297  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.898 on 677 degrees of freedom
## Multiple R-squared:  0.6226, Adjusted R-squared:  0.6193
## F-statistic: 186.2 on 6 and 677 DF,  p-value: < 2.2e-16
```

```r
#produce residual vs. fitted plot
plot(model,1)

#add a horizontal line at 0
abline(0,0)
```



```r
# predicting the model
preds = predict(model, test)

# RMSE (root mean squared error)
cat('RMSE score: ', RMSE(preds, test$LC50), "\n")
```

```
## RMSE score:  1.097393
```

```r
# MAE (mean absolute error)
cat('MAE score: ', mae(test$LC50, preds), "\n")
```

```
## MAE score:  0.7645595
```

```r
# coefficient of determination (R2 score)
cat('R2 score: ', summary(model)$r.squared, "\n")
```

```
## R2 score:  0.6226239
```

```r
# value of the likelihood with the "classical" sigma hat
log = sum(log(dnorm(x = data$LC50, mean = preds, sd = summary(model)$sigma)))

# AIC and BIC score

aic = AIC(model)
bic = BIC(model)

cat('log-likelihood: ', log, "\n")
```

```
## log-likelihood:  -2605.976
```

```r
cat('AIC: ', aic , "\n")
```

```
## AIC:  1802.925
```

```r
cat('BIC: ', bic, "\n")
```

```
## BIC:  1839.148
```

Which parameters are relevant for the model (i.e., have a parameter estimate, which is significantly unequal to 0)?

We can look on the summary to understand which parameter is relevant to the model. The parameters SM1_Dz and GATS1i are the most relevant, while NDssC is the least relevant with a value of 0.03.

Task d)

```r
#make this example reproducible
set.seed(1)

#use 75% of dataset as training set and 25% as test set
sample <- sample(c(TRUE, FALSE), nrow(data), replace=TRUE, prob=c(0.75,0.25))
train  <- data[sample, ]
test   <- data[!sample, ]

# Linear regression model
model = lm(LC50 ~ CIC0 + SM1_Dz + GATS1i + NdsCH + MLOGP, data=train)

summary(model)
```

```
##
## Call:
## lm(formula = LC50 ~ CIC0 + SM1_Dz + GATS1i + NdsCH + MLOGP, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4230 -0.5078 -0.0795  0.5017  3.4020
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.96344    0.19726   9.954  < 2e-16 ***
## CIC0         0.46740    0.06460   7.235 1.26e-12 ***
## SM1_Dz       1.38558    0.09058  15.297  < 2e-16 ***
## GATS1i      -0.79374    0.11401  -6.962 7.95e-12 ***
## NdsCH        0.44983    0.05908   7.614 8.92e-14 ***
## MLOGP        0.38196    0.03691  10.347  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8977 on 678 degrees of freedom
## Multiple R-squared:  0.6223, Adjusted R-squared:  0.6195
## F-statistic: 223.4 on 5 and 678 DF,  p-value: < 2.2e-16
```

```r
# predicting the model
preds = predict(model, test)

# RMSE (root mean squared error)
cat('RMSE score: ', RMSE(preds, test$LC50), "\n" )
```

```
## RMSE score:  1.100054
```

```r
# MAE (mean absolute error)
cat('MAE score: ', mae(test$LC50, preds), "\n" )
```

```
## MAE score:  0.7656347
```

```r
# coefficient of determination (R2 score)
cat('R2 score: ', summary(model)$r.squared, "\n" )
```

```
## R2 score:  0.6223248
```

```r
# value of the likelihood with the "classical" sigma hat
log = sum(log(dnorm(x = data$LC50, mean = preds, sd = summary(model)$sigma)))

# AIC and BIC score

aic = AIC(model)
bic = BIC(model)

cat('log-likelihood: ', log, "\n" )
```

```
## log-likelihood:  -2607.839
```

```r
cat('AIC: ', aic, "\n" )
```

```
## AIC:  1801.467
```

```r
cat('BIC: ', bic, "\n" )
```

```
## BIC:  1833.162
```

Compare the model summaries, evaluation metrics on the train and test set, as well as the log-likelihood, AIC and BIC values. Which model should be used?

The Values of the evaluation metrics (RMSE, MAE, R2) are very slightly better for the model whiteout the parameter NdssC.

Based on the log-likelihood, AIC and BIC, we can conclude that the model whiteout NdssC has best trade-off between goodness-of-fit and complexity, based on a lower overall value of the 3 variables.

In conclusion we would recommend that the model in task D should rather be used. The performance of the model is slightly better when excluding the parameter NdssC.

Exercise 3

Task a)

```r
Pic_1 = stack("3a.jpg")
```

```
## Warning: [rast] unknown extent
```

```r
Pic_2 = stack("3a_2.jpg")
```

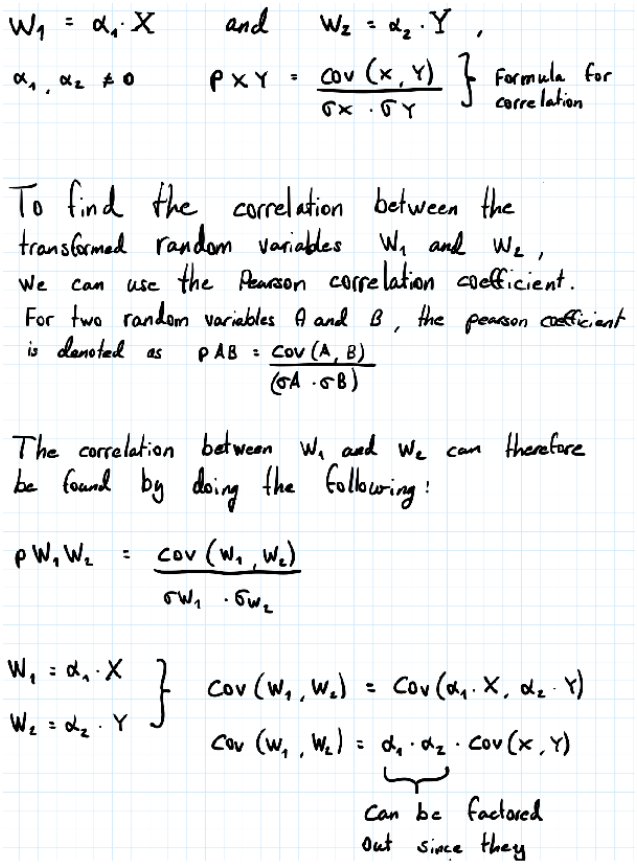
```
## Warning: [rast] unknown extent
```

```
plotRGB(Pic_1)
```

```
## Warning: [rast] unknown extent
```

```
## Warning: [rast] unknown extent
```

```
## Warning: [rast] unknown extent
```

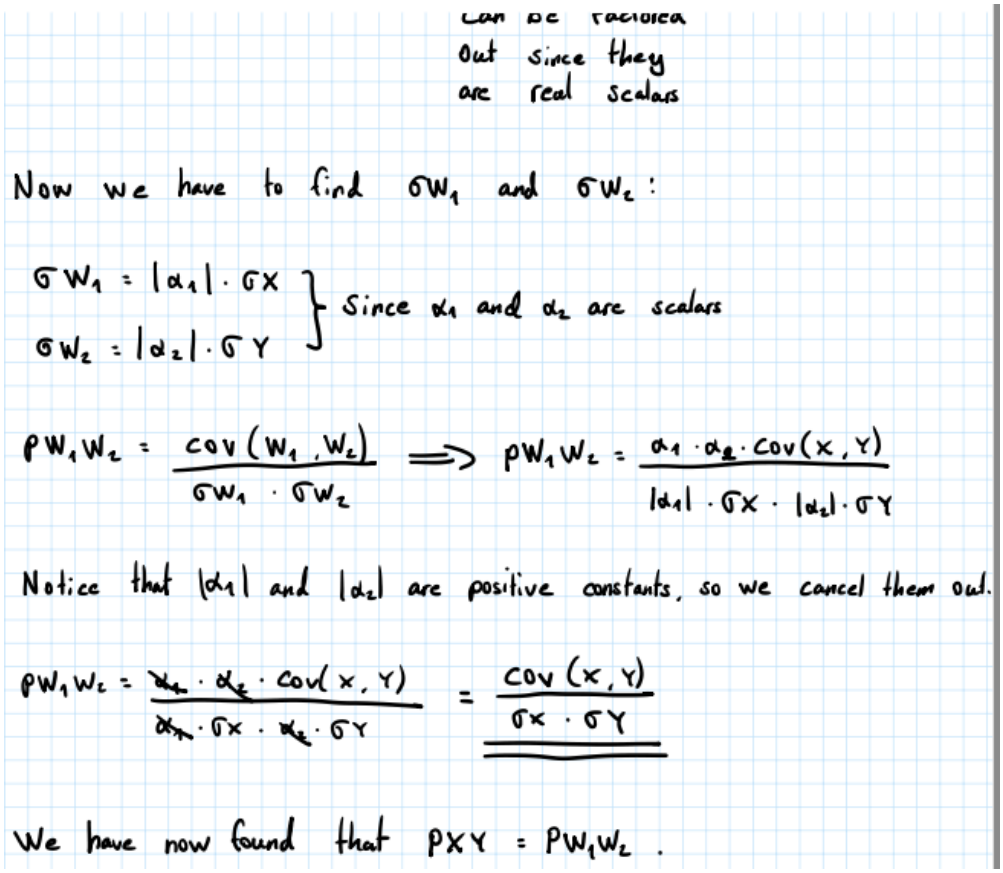



```
plotRGB(Pic_2)
```

```
## Warning: [rast] unknown extent
```

```
## Warning: [rast] unknown extent
```

```
## Warning: [rast] unknown extent
```



Task b)

Do research on the concept and explain in which scenarios it is necessary to use it instead of the default correlation coefficient ρ_{XY} . Give an example.

Partial correlation is particularly useful in scenarios where you want to examine the relationship between two variables while controlling for the influence of one or more other variables. It can be more appropriate than the Pearson correlation at several scenarios. One example is in biological and environmental studies, where one may want to see if there is a correlation between blood pressure (X) and amount of food eaten (Y) while controlling for weight (Z). Another example can be seen in the code in task d, where Z is found to be approximately 0.

Task c)

Does the same property as in task (a) hold for the partial correlation concept, as well, i.e. for the partial correlation of ρW1W2|Z ? Justify

In task (a), we showed that for the simple Pearson correlation (ρ), if you perform a linear transformation on two random variables X and Y (W1 = α_1 * X and W2 = α_2 * Y), the correlation between the transformed variables ρ(W1, W2) is the same as the correlation between the original variables ρ(X, Y), regardless of the values of α_1 and α_2 . This is because the linear transformation does not change the relationship between the variables, only scales it.

However, the partial correlation concept is different. It measures the relationship between two variables (W1 and W2) while controlling for a third variable (Z). The formula for ρ(W1, W2|Z) involves the correlations between X, Y, and Z, which can change the result significantly.

The property of ρ(W1, W2|Z) being equal to ρ(X, Y|Z) would only hold if the linear transformation (W1 = α_1 * X and W2 = α_2 * Y) does not change the relationship between X and Y while accounting for the influence of Z. In other words, for this property to hold, it must be the case that the linear transformation does not affect the conditional relationship between X and Y given Z.

In practice, whether this property holds or not depends on the specific values of α_1, α_2, and the underlying relationships between X, Y, and Z. If the linear transformation alters the conditional relationship between X and Y given Z, then ρ(W1, W2|Z) will not be the same as ρ(X, Y|Z). So, it is not a general property of partial correlation, and the result depends on the particulars of the problem at hand.

Task d)

```r
# Create a data frame with the provided values
data <- data.frame(X = c(1,1,1,1,0,0,0,0,1,1,1,1),
                   Z = c(3,1,3,1,1,-1,1,-1,3,1,3,1),
                   Y = c(5,5,9,1,1,1,5,-3,5,5,9,1))

# Calculate pairwise correlations
cor_XY <- cor(data$X, data$Y)  # Pearson correlation between X and Y
cor_XZ <- cor(data$X, data$Z)  # Pearson correlation between X and Z
cor_YZ <- cor(data$Y, data$Z)  # Pearson correlation between Y and Z

# Calculate partial correlation between X and Y, controlling for Z
partial_corr_XY_Z <- (cor_XY - cor_XZ * cor_YZ) / sqrt((1 - cor_XZ^2) * (1 - cor_YZ^2))

# Print the results
cat(paste0("Pearson correlation between X and Y: ", cor_XY), sep = "\n")
```

```
## Pearson correlation between X and Y: 0.554700196225229
```

```r
cat(paste0("Pearson correlation between X and Y: ", cor_XZ), sep = "\n")
```

```
## Pearson correlation between X and Y: 0.685994340570035
```

```r
cat(paste0("Pearson correlation between X and Y: ", cor_YZ), sep = "\n")
```

```
## Pearson correlation between X and Y: 0.80860754006264
```

```r
cat(paste0("Pearson correlation between X and Y: ", partial_corr_XY_Z), sep = "\n")
```

```
## Pearson correlation between X and Y: 5.18691165120874e-16
```

Exercise 4

Task a)

Step 1: The likelihood function for the model:

$$L(\beta; \sigma^2; X) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{\epsilon_i^2}{2\sigma^2}\right)$$

Step 2: The natural log of the likelihood function is caluclated:

$$\log L(\beta; \sigma^2; X) = \sum_{i=1}^{n} \left(-\log(\sigma\sqrt{2\pi}) - \frac{\epsilon_i^2}{2\sigma^2}\right)$$

Step 3: The equation for the i-th epsilon term is substituted into the equation:

$$\log L(\beta; \sigma^2; X) = -n\log(\sigma\sqrt{2\pi}) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - X\beta)^2$$

Step 4: The squared term is expanded:

$$\log L(\beta; \sigma^2; X) = -n\log(\sigma\sqrt{2\pi}) - \frac{1}{2\sigma^2}\left(\sum_{i=1}^{n} y_i^2 - 2\sum_{i=1}^{n} y_i(X\beta) + \sum_{i=1}^{n}(X\beta)^2\right)$$

Step 5: The second and third term in the parantheses is rewritten to transposed form:

$$\log L(\beta; \sigma^2; X) = -n\log(\sigma\sqrt{2\pi}) - \frac{1}{2\sigma^2}\left(\sum_{i=1}^{n} y_i^2 - 2(X\beta)^T y + \beta^T X^T X\beta\right)$$

$$\log L(\beta; \sigma^2; X) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)$$

Task b)

Assumption 1: Independence of Errors ϵ_i:

This assumption asserts that the errors ϵ_i for different data points are independent, meaning that the error in predicting one data point is not related to the error in predicting another data point. This assumption affects the first step in the calculations, as it allows us to write the likelihood as a product of individual likelihoods for each data point.

Assumption 2: Mean 0 of All ϵ_i:

This assumption states that the errors ϵ_i have a mean of zero, meaning they are centered around zero and do not systematically overestimate or underestimate the model predictions. This assumption would affect the first calculation step, as we would need to subtract the mean from the ϵ_i term. We've left the mean term out of our calculations since it's zero.

Assumption 3: Equal Variances σ^2 of all ϵ_i:

This assumption states that all errors ϵ_i have the same constant variance σ^2, meaning the variability of errors is consistent across all data points. This is a given assumption for linear regression, and this means that we can generalize the sigma term across all data points. So this assumption affects all calculation steps with sigma terms.

Task c)

First we start by taking the derivative of the log-likelihood function from task a):

$$\log L(\beta;\sigma^2;X) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2\sigma^2}(y - X\beta)^T(y - X\beta)$$

$$\frac{\partial \log L(\beta;\sigma^2;X)}{\partial \beta} = 0 - 0 - \frac{-2X^T(y - X\beta)}{2\sigma^2}$$

$$\frac{\partial \log L(\beta;\sigma^2;X)}{\partial \beta} = \frac{X^T(y - X\beta)}{\sigma^2}$$

We then set the derivative equal to zero:

$$\frac{X^T(y - X\hat{\beta})}{\sigma^2} = 0$$

$$X^T(y - X\hat{\beta}) = 0$$

$$X^Ty - X^TX\hat{\beta} = 0$$

$$X^TX\hat{\beta} = X^Ty$$

$$\hat{\beta} = (X^TX)^{-1}X^Ty$$

We can observe that our answer is similar to the Ordinary Least Squares estimate.

Task d)

First we calculate the derivative of the log-likelihood with respect to sigma and set it to zero:

$$\log L(\beta;\sigma^2;X) = -\frac{n}{2}\log(2\pi) - n\log(\sigma) - \frac{1}{2\sigma^2}\text{RSS}$$

$$\frac{\partial \log L(\beta;x)}{\partial \sigma} = -\frac{n}{\sigma} + \frac{1}{\sigma^3}\text{RSS}$$

$$\sigma^2 = \frac{\text{RSS}}{n}$$

Now that we have the expression for the MLE, we substitute it back into the log-likelihood formula:

$$\log L(\beta;\sigma^2;X) = -\frac{n}{2}\log(2\pi) - \frac{n}{2}\log\left(\frac{\text{RSS}}{n}\right) - \frac{n}{2}$$

$$\log L(\beta;\sigma^2;X) = -\frac{n}{2}\left(\log(2\pi) + \log\left(\frac{\text{RSS}}{n}\right) + 1\right)$$

Finally, we plug the log-likelihood expression into the BIC formula:

$$\text{BIC} = -2\log L(\hat{\beta};x) + k\log(n)$$

$$\text{BIC} = \left(n\left(\log(2\pi) + \log\left(\frac{\text{RSS}}{n}\right) + 1\right)\right) + dim(\beta)\log(n)$$