STAT202 Assignment 3: Multiple Linear Regression II

David Ewing

Due on 7th August 2024

#Introduction

This document contains the analysis for Assignment 3. It explores the aquatic\_toxicity dataset using multiple linear regression to understand the relationships between various predictors and the response variable LC50.

# Step 0: setup

loading libraries:

set.seed(82171165) #set seed   
  
knitr::opts\_chunk$set(  
 echo = TRUE, # Show all code by default  
 message = TRUE, # Include package messages  
 warning = TRUE # Include warnings if they occur  
)  
  
  
  
library(conflicted)  
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.3 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2

library(readxl)

## Warning: package 'readxl' was built under R version 4.4.2

library(readr)   
library(performance)

## Warning: package 'performance' was built under R version 4.4.2

library(GGally)

## Warning: package 'GGally' was built under R version 4.4.2

## Registered S3 method overwritten by 'GGally':  
## method from   
## +.gg ggplot2

library(flextable)

## Warning: package 'flextable' was built under R version 4.4.2

library(broom)  
library(skimr)

## Warning: package 'skimr' was built under R version 4.4.2

library(data.table)  
library(lmtest)

## Warning: package 'lmtest' was built under R version 4.4.2

## Loading required package: zoo  
##   
## Attaching package: 'zoo'  
##   
## The following objects are masked from 'package:data.table':  
##   
## yearmon, yearqtr  
##   
## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(leaps)

## Warning: package 'leaps' was built under R version 4.4.2

conflict\_prefer("filter", "dplyr"); conflict\_prefer("select", "dplyr")

## [conflicted] Will prefer dplyr::filter over any other package.  
## [conflicted] Will prefer dplyr::select over any other package.

# Step 2: Read aquatic\_toxicity.xlsx

# Load the data  
toxic <- read\_excel("../data/aquatic\_toxicity.xlsx")

# Step 3: Select a random sample of 500 rows

set.seed(82171165)  
  
toxic\_rows <- nrow(toxic)  
toxic\_sample <- toxic |> slice\_sample(n = 500) # without replacement  
my\_toxic <- toxic\_sample  
  
# Summarise missing values   
# View the first few rows of the sample  
  
skim\_toxic <- skim(my\_toxic) |>  
 select(skim\_variable, n\_missing)  
skim\_toxic

## # A tibble: 9 × 2  
## skim\_variable n\_missing  
## <chr> <int>  
## 1 tpsa 0  
## 2 saacc 0  
## 3 h\_050 0  
## 4 mlogp 0  
## 5 rdchi 0  
## 6 gats1p 0  
## 7 nn 0  
## 8 c\_040 0  
## 9 lc50 0

head(toxic\_sample)

## # A tibble: 6 × 9  
## tpsa saacc h\_050 mlogp rdchi gats1p nn c\_040 lc50  
## <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  
## 1 0 0 0 2.85 1.99 0.938 0 0 3.53  
## 2 93.2 118. 1 -0.147 2.36 1.64 2 0 5.9   
## 3 45.8 50.7 0 2.24 2.12 0.942 1 0 4.01  
## 4 64.2 92.5 0 3.61 4.24 1.18 4 2 4.84  
## 5 133. 75.2 0 0.906 3.03 0.941 3 1 7.86  
## 6 35.2 43.9 2 1.25 2.05 1.08 1 0 5.64

# Step 4: Estimate the correlations between all variables

# Calculate the correlation matrix  
cors <- cor(my\_toxic, use = "complete.obs") # Ensure complete observations  
cors\_3d <- round(cors, 3) # Round to 3 decimal places  
  
# Convert the correlation matrix into a data frame and add row identifiers  
cor\_df <- as.data.frame(cors\_3d)  
cor\_df <- tibble::rownames\_to\_column(cor\_df, var = " ") # Add var column  
  
cor\_table <- flextable(cor\_df) %>%  
 set\_caption("Correlation Matrix of Variables in my\_toxic (Rounded to 3 Decimal Places)") %>%  
 bg(part = "header", bg = "#D3D3D3") %>% # Add grey header  
 bg(j = 1, bg = "#D3D3D3", part = "all") %>% # Add grey column  
 theme\_box() %>%  
 autofit() %>% # adjust column widths  
 align(j = 1, align = "left", part = "all") # Align the first column (row identifiers) to the left  
#cors

cor\_table

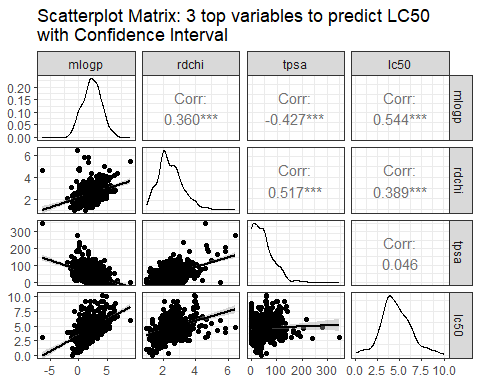
Correlation Matrix of Variables in my\_toxic (Rounded to 3 Decimal Places)

|  | **tpsa** | **saacc** | **h\_050** | **mlogp** | **rdchi** | **gats1p** | **nn** | **c\_040** | **lc50** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| tpsa | 1.000 | 0.847 | 0.622 | -0.427 | 0.517 | 0.172 | 0.595 | 0.445 | 0.046 |
| saacc | 0.847 | 1.000 | 0.743 | -0.370 | 0.566 | 0.213 | 0.491 | 0.492 | -0.107 |
| h\_050 | 0.622 | 0.743 | 1.000 | -0.465 | 0.262 | 0.056 | 0.453 | 0.189 | -0.212 |
| mlogp | -0.427 | -0.370 | -0.465 | 1.000 | 0.360 | -0.372 | -0.267 | -0.105 | 0.544 |
| rdchi | 0.517 | 0.566 | 0.262 | 0.360 | 1.000 | 0.048 | 0.340 | 0.418 | 0.389 |
| gats1p | 0.172 | 0.213 | 0.056 | -0.372 | 0.048 | 1.000 | 0.074 | 0.141 | -0.310 |
| nn | 0.595 | 0.491 | 0.453 | -0.267 | 0.340 | 0.074 | 1.000 | 0.315 | -0.073 |
| c\_040 | 0.445 | 0.492 | 0.189 | -0.105 | 0.418 | 0.141 | 0.315 | 1.000 | 0.018 |
| lc50 | 0.046 | -0.107 | -0.212 | 0.544 | 0.389 | -0.310 | -0.073 | 0.018 | 1.000 |

# Step 4+: Top variables to predict LC50

# 3 variables with the highest correlation with LC50  
lc50\_sort <- sort(cors["lc50", ], decreasing = TRUE)  
lc50\_sort <- lc50\_sort[lc50\_sort != 1]  
top\_3 <- names(lc50\_sort[1:3])  
  
lc50\_matrix <- my\_toxic[, c(top\_3, "lc50")] |>  
 ggpairs(  
 lower = list(continuous = wrap("smooth", method = "lm", se = TRUE)),  
 title = "Scatterplot Matrix: 3 top variables to predict LC50\nwith Confidence Interval"  
 ) +  
 theme\_bw()

lc50\_matrix



A positive correlation between **mlogp** and **LC50**; with weaker positive relationships for **rdchi** and **LC50**; and an insiginficatn association between **tpsa** and **LC50**. Each relationship appear generally linear but variability, particularly in **rdchi**, could affect predictability. THere has been no attempt to remove outliers.

# Step 5: multiple linear regression model to predict LC50

lc50\_arg <- as.formula(paste("lc50 ~ ", paste(top\_3, collapse = " + ")))  
m1 <- lm(lc50\_arg, data = my\_toxic)  
summary(m1)

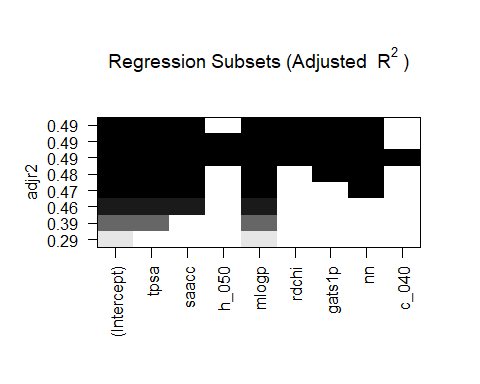
##   
## Call:  
## lm(formula = lc50\_arg, data = my\_toxic)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.8829 -0.8113 -0.1803 0.6748 5.0817   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 2.680400 0.190861 14.044 < 2e-16 \*\*\*  
## mlogp 0.741600 0.056631 13.095 < 2e-16 \*\*\*  
## rdchi -0.217494 0.125996 -1.726 0.0849 .   
## tpsa 0.015881 0.002363 6.720 4.99e-11 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.29 on 496 degrees of freedom  
## Multiple R-squared: 0.3938, Adjusted R-squared: 0.3901   
## F-statistic: 107.4 on 3 and 496 DF, p-value: < 2.2e-16

tidy(m1)

## # A tibble: 4 × 5  
## term estimate std.error statistic p.value  
## <chr> <dbl> <dbl> <dbl> <dbl>  
## 1 (Intercept) 2.68 0.191 14.0 5.85e-38  
## 2 mlogp 0.742 0.0566 13.1 7.31e-34  
## 3 rdchi -0.217 0.126 -1.73 8.49e- 2  
## 4 tpsa 0.0159 0.00236 6.72 4.99e-11

# Step 6: Fit All Regression Subsets using adjusted R-squared

all\_models <- regsubsets(lc50 ~ ., data = my\_toxic)  
plot(all\_models, scale = "adjr2",   
 main = expression("Regression Subsets (Adjusted " ~ R^2 ~ ")"))



Both mlogp and tpsa appear here and in the coorelation matrix as good and perhaps reliable predictors for LC50, however, saacc and nn are selected the regression subsets relative to adjusted r-squared. This suggest that while they may not be strong standalone predictors, they may reduce the variablility when added to combinations with other variables like mlogp.

# Step 7: Fit the Best Model and Compare

# Fit the best model identified in Step 6  
m2\_arg1 <- lc50 ~ saacc + nn + mlogp # Based on the best subset from Step 6  
m2 <- lm(m2\_arg1, data = my\_toxic)  
  
# Summary of the best model  
summary(m2)

##   
## Call:  
## lm(formula = m2\_arg1, data = my\_toxic)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -2.6368 -0.8883 -0.2242 0.6039 5.3188   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 3.105245 0.143825 21.590 <2e-16 \*\*\*  
## saacc 0.002382 0.001168 2.038 0.042 \*   
## nn 0.046922 0.051358 0.914 0.361   
## mlogp 0.570540 0.039272 14.528 <2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.38 on 496 degrees of freedom  
## Multiple R-squared: 0.3069, Adjusted R-squared: 0.3027   
## F-statistic: 73.22 on 3 and 496 DF, p-value: < 2.2e-16

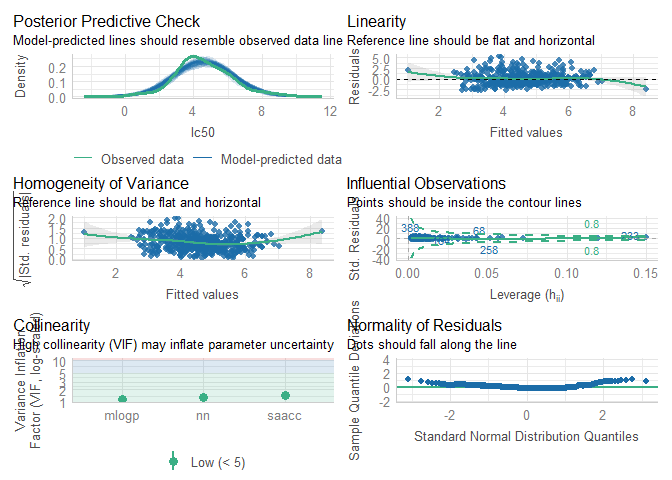
Comparison of Regression Models: Step 5 vs Step 7

| **Metric** | **m1** | **m2** |
| --- | --- | --- |
| Adjusted R² | 0.3901 | 0.3027 |
| Residual SE | 0.3027 | 1.3796 |

While m2 includes saacc and nn, which may reduce multicollinearity, it does not outperform m1 in terms of predictive power or error minimisation. m1 has a higher adjusted r-square indicating that it explains more vaariance of LC50 than m2. The residule SE for m1 is significantly lower than m2 meaning that m1 provides a more precise model of LC50 than m2.

# Step8: Diagnostic Plots for Model Residuals and to Check Assumptions

check\_model(m2)



m2\_residuals <- residuals(m2)

Diagnostic Metrics and Assumption Validation

| **Metric** | **Assumption\_Met** |
| --- | --- |
| Linearity | No, deviations from linearity in the Residuals vs Fitted plot. |
| Homoscedasticity | No, increasing variance of residuals indicates heteroscedasticity. |
| Normality of Residuals | No, Q-Q plot shows deviations from normality at the tails. |
| Influential Observations | No, some influential points (e.g., 388, 258) are identified. |
| Collinearity | Yes, VIF values are below 5, indicating low multicollinearity.. |
| Posterior Predictive Check | Yes, reasonable alignment. |

The posterior predictive check shows reasonable alignment between the model-predicted LC50 and observed data. The Residuals vs Fitted Values plot indicates curvature, suggesting potential non-linearity in the predictor-response relationships. The Scale-Location plot shows increasing residual variance as fitted values increase, discrediting the assumption of homoscedasticity. Influential observations, such as points 388 and 258, are flagged as close to the Cook’s distance lines and may disproportionately influence the model. The Variance Inflation Factor (VIF) values for the predictors (mlogp, nn, saacc) are all below 5, indicating low multicollinearity and stability. The Q-Q plot shows deviations from the normal at the tails, suggesting that residuals are not perfectly normally distributed.

# Step 9: Predict LC50 with Confidence and Prediction Intervals

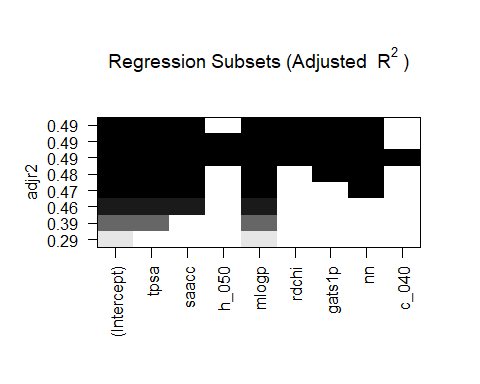
# Step 9: Predict LC50 with Confidence and Prediction Intervals  
  
# Create tibble of specified characteristics  
new\_chemicals <- tibble(  
 tpsa = c(69.97, 3.24),  
 saacc = c(97.43, 3.12),  
 h\_050 = c(0, 0),  
 mlogp = c(3.12, 9.15),  
 rdchi = c(3.72, 5.49),  
 gats1p = c(1.26, 1.56),  
 nn = c(0, 1),  
 c\_040 = c(2, 0)  
)  
  
# Generate predictions with confidence intervals  
confidence\_predictions <- predict(  
 m1,   
 newdata = new\_chemicals,   
 interval = "confidence",   
 level = 0.95  
)  
  
# Generate predictions with prediction intervals  
prediction\_predictions <- predict(  
 m1,   
 newdata = new\_chemicals,   
 interval = "prediction",   
 level = 0.95  
)  
  
# Combine the results into a single tibble  
predicted\_chemicals <- new\_chemicals %>%  
 mutate(  
 Predicted\_LC50 = confidence\_predictions[, "fit"],  
 Lwr\_Confidence = confidence\_predictions[, "lwr"],  
 Upr\_Confidence = confidence\_predictions[, "upr"],  
 Lwr\_Prediction = prediction\_predictions[, "lwr"],  
 Upr\_Prediction = prediction\_predictions[, "upr"]  
 )  
  
# Create a table summarising predictions for display  
simple\_table <- tibble(  
 Chemical = c("Chemical 1", "Chemical 2"),  
 `Predicted LC50` = round(confidence\_predictions[, "fit"], 3),  
 `Lwr Confidence` = round(confidence\_predictions[, "lwr"], 3),  
 `Upr Confidence` = round(confidence\_predictions[, "upr"], 3),  
 `Lwr Prediction` = round(prediction\_predictions[, "lwr"], 3),  
 `Upr Prediction` = round(prediction\_predictions[, "upr"], 3)  
)  
  
# Display the table using flextable  
library(flextable)  
  
flextable(simple\_table) %>%  
 set\_caption("Predicted LC50 with 95% Confidence and Prediction Intervals") %>%  
 bg(part = "header", bg = "#D3D3D3") %>%  
 theme\_box() %>%  
 align(j = 1, align = "left", part = "all") %>%  
 align(j = 2:6, align = "center", part = "all") %>%  
 autofit()

Predicted LC50 with 95% Confidence and Prediction Intervals

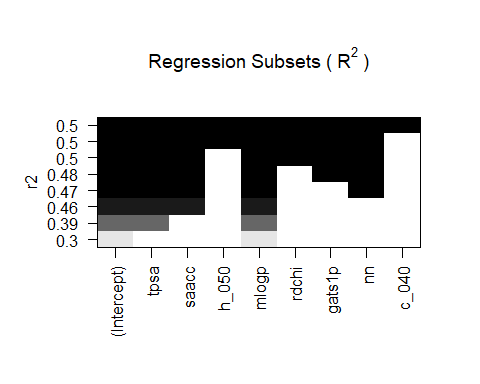
| **Chemical** | **Predicted LC50** | **Lwr Confidence** | **Upr Confidence** | **Lwr Prediction** | **Upr Prediction** |
| --- | --- | --- | --- | --- | --- |
| Chemical 1 | 5.296 | 5.078 | 5.515 | 2.752 | 7.841 |
| Chemical 2 | 8.323 | 7.739 | 8.908 | 5.722 | 10.925 |

# Step 10: Compare Regression Subsets with Different Scales

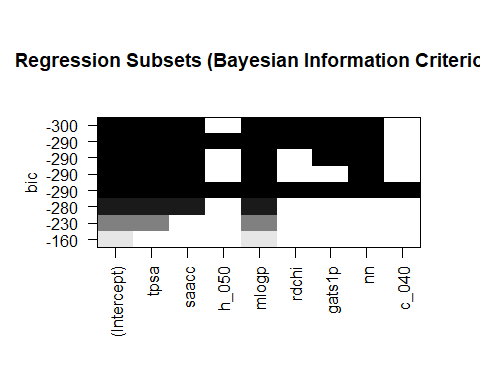
# Step 10: Compare Regression Subsets with Different Scales  
  
# Plot using adjusted R²  
plot(all\_models, scale = "adjr2",   
 main = expression("Regression Subsets (Adjusted " ~ R^2 ~ ")"))



# Plot using R²  
plot(all\_models, scale = "r2",   
 main = expression("Regression Subsets (" ~ R^2 ~ ")"))



# Plot using BIC  
plot(all\_models, scale = "bic",   
 main = "Regression Subsets (Bayesian Information Criterion)")



# Create a summary table of selected models  
model\_summary <- tibble(  
 Criterion = c("Adjusted R²", "R²", "BIC"),  
 `Selected Model` = c(  
 "lc50 ~ mlogp + nn + saacc", # Adjusted R² model  
 "lc50 ~ mlogp + nn + saacc + tpsa + rdchi", # R² model  
 "lc50 ~ mlogp + saacc" # BIC model  
 ),  
 Explanation = c(  
 "Balances fit and simplicity.",  
 "Maximises variance explained\nbut may overfit.",  
 "Prioritises simplicity and\navoids overfitting."  
 )  
)  
  
# Use flextable to format and display the table  
library(flextable)  
  
flextable(model\_summary) %>%  
 set\_caption("Comparison of Models Selected by Adjusted R², R², and BIC") %>%  
 bg(part = "header", bg = "#D3D3D3") %>%  
 theme\_box() %>%  
 align(j = 1:3, align = "left", part = "all") %>%  
 autofit()

Comparison of Models Selected by Adjusted R², R², and BIC

| **Criterion** | **Selected Model** | **Explanation** |
| --- | --- | --- |
| Adjusted R² | lc50 ~ mlogp + nn + saacc | Balances fit and simplicity. |
| R² | lc50 ~ mlogp + nn + saacc + tpsa + rdchi | Maximises variance explained but may overfit. |
| BIC | lc50 ~ mlogp + saacc | Prioritises simplicity and avoids overfitting. |

Adjusted r-square selects a balanced model with moderate predictors, optimising fit while penalising complexity. r-square prioritises explaining variance, often favouring larger models and risk overfitting. BIC heavily penalises model complexity, selecting simpler models.

# Step 11: Reproduce Regression Coefficients for m2 using Matrix Algebra

X <- model.matrix(m2)  
  
y <- my\_toxic %>% pull(lc50) # Extract lc50 column as a vector  
  
  
coefficients\_m2 <- solve(t(X) %\*% X) %\*% t(X) %\*% y # regression coefficients   
lm\_coefficients <- coef(m2)  
  
# Combine the results into a tibble for comparison  
comparison\_df <- tibble(  
 Predictor = names(lm\_coefficients),  
 `Matrix Algebra Coefficients` = as.vector(coefficients\_m2),  
 `lm() Coefficients` = lm\_coefficients  
)  
  
comparison\_table <- flextable(comparison\_df) %>%  
 set\_caption("Comparison of Regression Coefficients: Matrix Algebra vs lm()") %>%  
 bg(part = "header", bg = "#D3D3D3") %>%  
 theme\_box() %>%  
 align(j = 1, align = "left", part = "all") %>% # Left-align Predictor column  
 align(j = 2:3, align = "center", part = "all") %>% # Center-align coefficient columns  
 border\_inner\_v(part = "all") %>%  
 border\_inner\_h(part = "all") %>%  
 border\_outer(part = "all") %>%  
 autofit()  
  
# Display the table  
comparison\_table

Comparison of Regression Coefficients: Matrix Algebra vs lm()

| **Predictor** | **Matrix Algebra Coefficients** | **lm() Coefficients** |
| --- | --- | --- |
| (Intercept) | 3.105244701 | 3.105244701 |
| saacc | 0.002381938 | 0.002381938 |
| nn | 0.046921703 | 0.046921703 |
| mlogp | 0.570539551 | 0.570539551 |