Task 2

Statistical Analysis

Title: Analysis of Factors Influencing Concrete Compressive Strength

Dataset: Concrete Compressive Strength

2.2 Introduction

Concrete is a cornerstone material in civil engineering, valued for its strength, durability, and versatility. However, its compressive strength is influenced by a complex interplay of variables, including its age and the proportions of ingredients such as cement, blast furnace slag, fly ash, water, superplasticizer, coarse aggregate, and fine aggregate. Understanding these relationships is essential for optimizing concrete performance and ensuring safety in construction applications.

2.3 Objective

This report investigates the factors influencing concrete compressive strength through a comprehensive statistical analysis. Using regression techniques and hypothesis testing, the aim is to identify significant predictors and interactions, providing insights into the optimization of concrete mixtures.

2.4 Background Research and Literature Review

One of the widely used materials in construction is concrete, due to its strength, durability, and versatility. Its compressive strength, a critical measure of performance, depends on a complex interplay of factors including its age, composition, and curing conditions. The primary ingredients influencing strength are water and cement, aggregates (fine and coarse), and chemical admixtures such as superplasticizers and supplementary materials like blast furnace slag and fly ash.

Studies have extensively explored the prediction and optimization of concrete compressive strength. Yeh (1998) demonstrated that concrete strength is a nonlinear function of its ingredients and age. Using artificial neural networks, Yeh established a robust predictive framework, which highlighted the critical role of factors such as cement content and curing age. Similar studies have incorporated statistical regression models, design of experiments, and other advanced computational methods to improve predictions and reveal interactions among variables (Yeh, 2006).

2.5 Preparation and Exploration of Dataset

Source and Structure:

The dataset was developed and donated by Prof. I.-C. Yeh from Chung-Hua University, Taiwan. It includes variables measured in real-world laboratory experiments to determine the compressive strength of concrete under varying conditions.

Variable Breakdown:

Input Variables:

1. Cement (kg/m^3)
2. Blast Furnace Slag (kg/m^3)
3. Fly Ash (kg/m^3)
4. Water (kg/m^3)
5. Superplasticizer (kg/m^3)
6. Coarse Aggregate (kg/m^3)
7. Fine Aggregate (kg/m^3)
8. Age (days, range: 1–365)
9. Binary Variables:
    - Concrete Category (Coarse/Fine aggregate proportions)
    - Contains Fly Ash (TRUE/FALSE)

10. Output Variable:

- Concrete compressive strength (MPa).

Data Summary:

Observations: 1030

Missing Values: None

Attribute Types: Quantitative (8 predictors), Binary (2 predictors).

Citation: Yeh, I.-C. (1998). Modeling of strength of high-performance concrete using artificial neural networks. Cement and Concrete Research, 28(12), 1797–1808.

2.7 Data Preparation Steps

```
# ------------------------------------------------
# ----------------Load Libraries and Dataset
# Install necessary libraries
install.packages("readxl")
install.packages("ggplot2")
install.packages("dplyr")
install.packages("corrplot")
install.packages("stats")
install.packages("car")
install.packages("fitdistrplus")
install.packages("tidyverse")

install.packages("xgboost")
install.packages("randomForest")

# Load libraries
library(readxl)
library(ggplot2)
library(dplyr)
library(corrplot)
library(car)
library(fitdistrplus)
library(tidyverse)
library(tidyr)

library(randomForest)
library(xgboost)
# ------------------------------------------------
```

Figure 2.1: Installing and loading libraries for data analysis.

```
25  # Load the dataset
26  Concrete_data <- read_excel("concrete compressive strength.xlsx")
```

Figure 2.2: The dataset was imported using “read_excel” R-code above.

```r
# Manually rename columns
colnames(Concrete_data) <- c(
  "Cement",
  "Blast_Furnace_Slag",
  "Fly_Ash",
  "Water",
  "Superplasticizer",
  "Coarse_Aggregate",
  "Fine_Aggregate",
  "Age",
  "Concrete_Category",
  "Contains_Fly_Ash",
  "Compressive_Strength"
)
```

Figure 2.3: Variable names were rewritten for ease of use during analysis using the "**colnames**"

```r
#-----------------------------------------------
# Overview of the data
str(Concrete_data)
summary(Concrete_data)
names(Concrete_data)
head(Concrete_data)
tail(Concrete_data)
```

```
A tibble: 6 × 11
Cement Blast_Furnace_Slag Fly_Ash Water Superplasticizer Coarse_Aggregate Fine_Aggregate   Age Concrete_Category
  <dbl>              <dbl>   <dbl> <dbl>            <dbl>            <dbl>          <dbl> <dbl> <chr>
  540                  0       0   162              2.5             1040            676     28 Coarse
  540                  0       0   162              2.5             1055            676     28 Coarse
  332.               142.      0   228              0                932            594    270 Coarse
  332.               142.      0   228              0                932            594    365 Coarse
  199.               132.      0   192              0                978.           826.   360 Fine
  266                114       0   228              0                932            670     90 Coarse
i 2 more variables: Contains_Fly_Ash <lgl>, Compressive_Strength <dbl>
```

Figure 2.4: Structure Checking, and variables were inspected for structure, types, and summary statistics (Wickham & Grolemund, 2017).

```r
# Check if any duplicate rows exist
anyDuplicated(Concrete_data)
# Output: Index of the first duplicate row, or 0 if no duplicates
```

```
> anyDuplicated(Concrete_data)
[1] 78
```

Figure 2.5: With the code above, 78 duplicates were seen

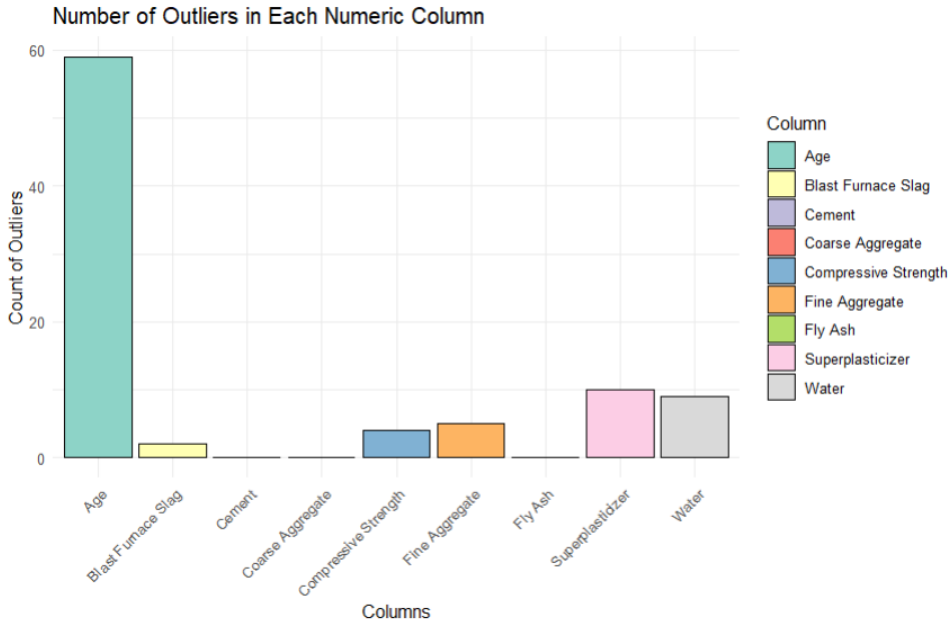
Outlier Detection: Visualisations and summary statistics were used to identify potential outliers in numerical variables, also these records are realistic and as such the outliers would only be addressed on variables that are extremely skewed and badly affect the outcome of the regression model (Wickham, 2016).

```r
# Function to detect outliers using IQR
detect_outliers <- function(column) {
  Q1 <- quantile(column, 0.25)
  Q3 <- quantile(column, 0.75)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  which(column < lower_bound | column > upper_bound)
}

# Apply outlier detection to numeric columns
numeric_columns <- Concrete_data[, sapply(Concrete_data, is.numeric)]
outliers <- lapply(numeric_columns, detect_outliers)

# Display outliers
outliers

# Count outliers in each numeric column
outlier_counts <- sapply(outliers, length)
outlier_data <- data.frame(Column = names(outlier_counts), Count = outlier_counts)
print(outlier_data)
```

Figure 2.6: Above is the r-code I used to detect outliers in numeric columns

```r
# Plot outlier counts
ggplot(outlier_data, aes(x = Column, y = Count, fill = Column)) +
  geom_bar(stat = "identity", color = "black") +
  labs(title = "Number of Outliers in Each Numeric Column") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  scale_fill_brewer(palette = "Set3")
```

Figure 2.7: Plot above visualises outliers count below.



```r
# Replace outliers with the median value of the column
for (col_name in names(outliers)) {
  outlier_indices <- outliers[[col_name]]
  median_value <- median(Concrete_data[[col_name]], na.rm = TRUE)
  Concrete_data[outlier_indices, col_name] <- median_value
}
```

Figure 2.8: This steps above help replace outliers with median values of respective columns

```r
# Verify that outliers have been replaced and no further outliers exist
sapply(Concrete_data, function(x) sum(is.na(x)))
```

```
             Cement   Blast_Furnace_Slag             Fly_Ash                Water     Superplasticizer
                  0                    0                   0                    0                    0
   Coarse_Aggregate       Fine_Aggregate                 Age    Concrete_Category     Contains_Fly_Ash
                  0                    0                   0                    0                    0
Compressive_Strength
                  0
```

Figure 2.9: Managing Missing Values; All data was complete and prepared for additional analysis. No missing values were found (Wickham & Grolemund, 2017).

```r
# Check the class of the variables
class(Concrete_data$`Concrete Category`)
class(Concrete_data$`Contains Fly Ash`)
```

```
[1] "character"
> class(Concrete_data$`Contains_Fly_Ash`)
[1] "logical"
```

Figure 2.10: The Type Conversion; for appropriate analysis, categorical variables such as Contains Fly Ash and Concrete Category were transformed into factors (R Core Team, 2023).

```r
# Convert categorical variables to factors
Concrete_data$`Concrete_Category` <- as.factor(Concrete_data$`Concrete_Category`)
Concrete_data$`Contains_Fly_Ash` <- as.factor(Concrete_data$`Contains_Fly_Ash`)
```

Figure 2.11: The code above converts categorical variables to factor for efficient use on models.

```r
# Check if conversion was successful
is.factor(Concrete_data$`Concrete_Category`)
is.factor(Concrete_data$`Contains_Fly_Ash`)
```

```
> is.factor(Concrete_data$`Concrete_Category`)
[1] TRUE
> is.factor(Concrete_data$`Contains_Fly_Ash`)
[1] TRUE
```

Figure 2.12: From the outcome above they are now factors.

```r
# View contingency table for categorical variables
table(Concrete_data$`Concrete_Category`, Concrete_data$`Contains_Fly_Ash`)
```

```
        FALSE TRUE
Coarse    296  243
Fine      270  221
```

Figure 2.13: this outcome shows the correlation between these two categories.

```r
# ------------------------------------------------
# Exploratory Data Analysis (EDA)
# Boxplot for Compressive Strength by Concrete Category
ggplot(Concrete_data, aes(x = `Concrete_Category`, y = `Compressive_Strength`)) +
  geom_boxplot(fill = "orange") +
  labs(title = "Boxplot of Compressive Strength by Concrete Category",
       x = "Concrete Category", y = "Compressive Strength (MPa)")
```

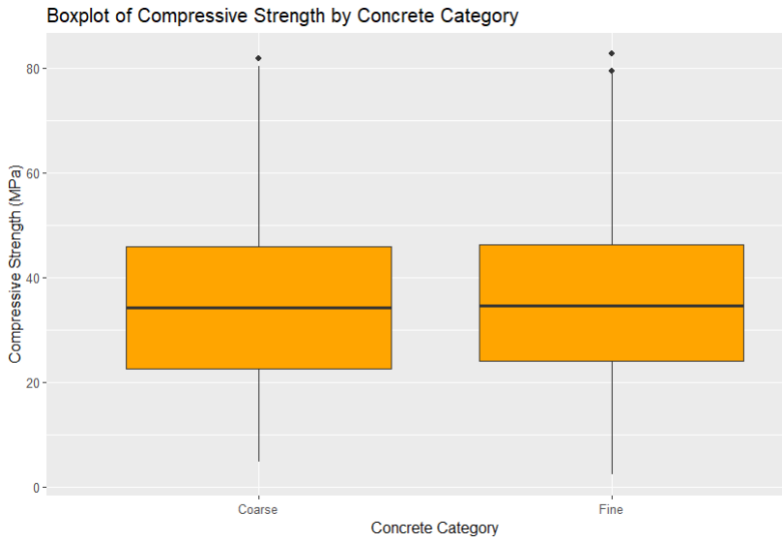


Figure 2.14: This R-code above helped explore the balance of coarse and fine aggregate.

```r
# Barplot for Contains Fly Ash
ggplot(Concrete_data, aes(x = `Contains_Fly_Ash`)) +
  geom_bar(fill = "blue") +
  labs(title = "Distribution of Concrete Categories", x = "Contains Fly Ash", y = "Frequency")
```

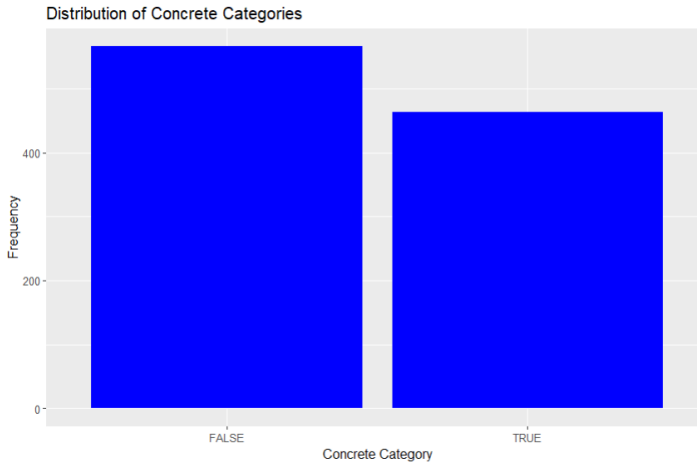


Figure 2.15: Ratio between the addition of Fly ash or non-addition of fly ash in mixture.

2.8 Correlation Analysis:

In this task correlation analysis examines the relationships between the predictors of concrete's compressive strength. Using Pearson and Spearman correlation coefficients, the analysis identifies linear and nonlinear dependencies, highlighting key factors with significant influence. This provides insights into variable interactions and guides model development for strength prediction.

As we have both numerical and categorical variable in the dataset, different variable types need separate correlation analysis

```r
# Explicitly using dplyr's select function
continuous_data <- dplyr::select(Concrete_data, -Concrete_Category, -Contains_Fly_Ash)
```

Figure 2.16: The R-code above is a new dataframe housing only numeric variables.

```r
# Plot histograms with density overlays
long_data <- continuous_data %>%
  pivot_longer(cols = everything(), names_to = "Variable", values_to = "Value")

ggplot(long_data, aes(x = Value)) +
  geom_histogram(aes(y = ..density..), bins = 10, fill = "blue", color = "black", alpha = 0.7) +
  geom_density(alpha = 0.2, fill = "red") +
  labs(title = "Distribution of Variables", x = "Value", y = "Density") +
  facet_wrap(~Variable, scales = "free") +
  theme_minimal()
```

Figure 2.17: The code above visualises distributions of all continuous variables using histograms and density plots.

```r
# Creating and plot correlation matrix
corr_matrix <- cor(continuous_data)
corrplot(corr_matrix, method = "number", type = "upper")
```

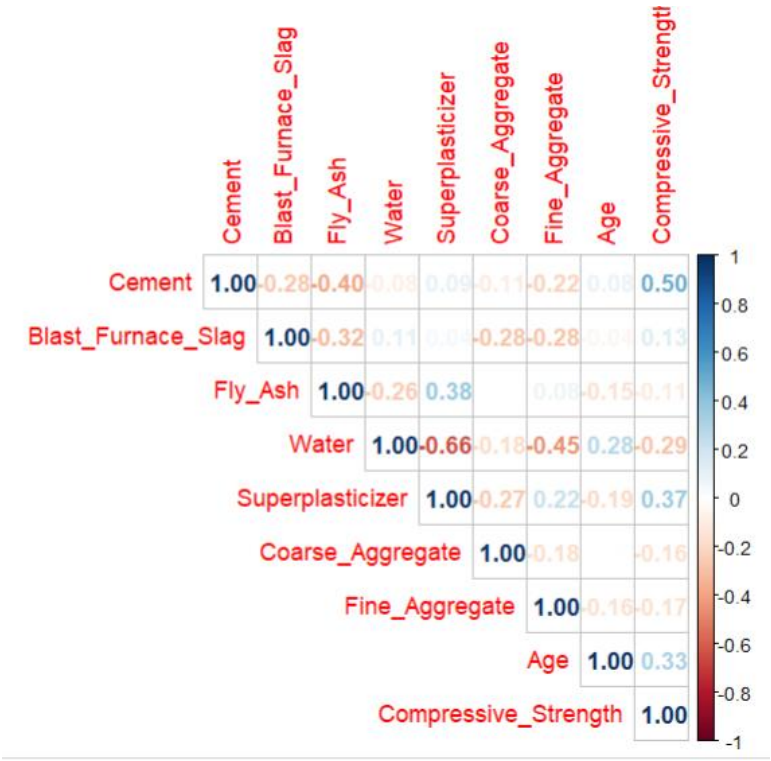


Figure 2.18: The code calculates the correlation matrix for my numerical variables and displays the correlation matrix

The correlation matrix provides valuable insights into the relationships between variables influencing concrete compressive strength. Notably, there is a strong positive correlation between Cement and Concrete Compressive Strength (**r = 0.50**). This relationship highlights that increasing cement content in the mix enhances the concrete's strength, which aligns with its role as the primary binding agent in concrete mixtures (Yeh, 1998). Cement contributes significantly to the mechanical properties of concrete, reinforcing its importance in mix design.

Another key finding is the negative correlation between Fly Ash and Water (**r = -0.29**). Fly ash, commonly used as a partial substitute for cement, reduces the water demand due to its finer particle size and pozzolanic reactions. This interaction improves workability while potentially enhancing strength, provided the mix is well-designed (Yeh, 1999).

The Superplasticizer shows a moderate positive correlation with Concrete Compressive Strength (**r = 0.37**). Superplasticizers improve concrete workability and reduce water content without compromising strength, making them an essential component of high-performance concrete (Yeh, 2006).

Finally, the correlation between Age and Concrete Compressive Strength is relatively low (r = 0.33), suggesting that while curing age influences strength, material proportions have a more dominant impact. These relationships underscore the intricate balance of materials in optimizing concrete performance (Yeh, 2003).

2.9 Regression Analysis:

Regression analysis helps understand how different factors, like cement, water, and age, affect concrete compressive strength. Using multiple linear regression, we can identify the most important variables, measure their impact, and improve our ability to predict strength accurately. This builds on earlier research by Yeh (1998).

2.9.1 Reason Simpler Linear Regression (SLR)

The specific effects of each variable on the compressive strength of concrete were ascertained using Simple Linear Regression (SLR). This helped to identify the most influential predictor which is “Cement”, providing a starting point for understanding how each material contributes to concrete performance (Yeh, 1998). By adding the most important variables one after the other until a certain stopping rule was reached, I decided to go forward incrementally. The “Stats package’s” “lm” function was employed.

```
model_1 <- lm(Compressive_Strength ~ Cement, data = continuous_data)
summary(model_1)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 13.442795   1.296925   10.37   <2e-16 ***
Cement       0.079580   0.004324   18.41   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.5 on 1028 degrees of freedom
Multiple R-squared:  0.2478,    Adjusted R-squared:  0.2471
F-statistic: 338.7 on 1 and 1028 DF,  p-value: < 2.2e-16
```

Figure 2.19: shows that **concrete compressive strength** = 13.442795 + 0.079580 x Cement

R^2 = 0.25 i.e. 25% of the total variability in the compressive strength of concrete can be predicted using this regression equation.

1. Linearity

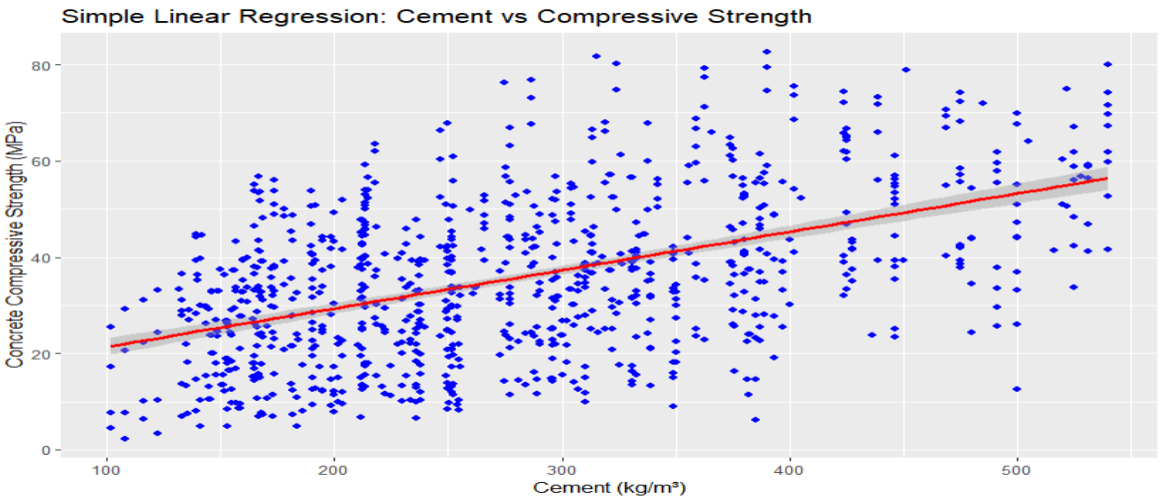
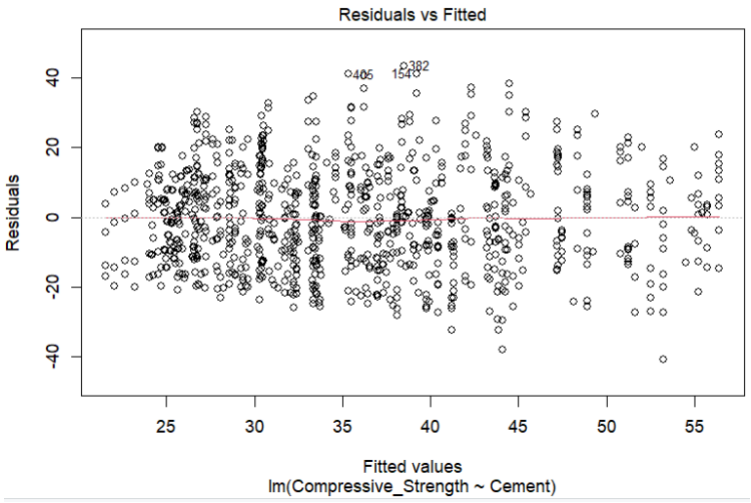


Figure 2.20: Concrete’s compressive strength and cement have a very linear connection.

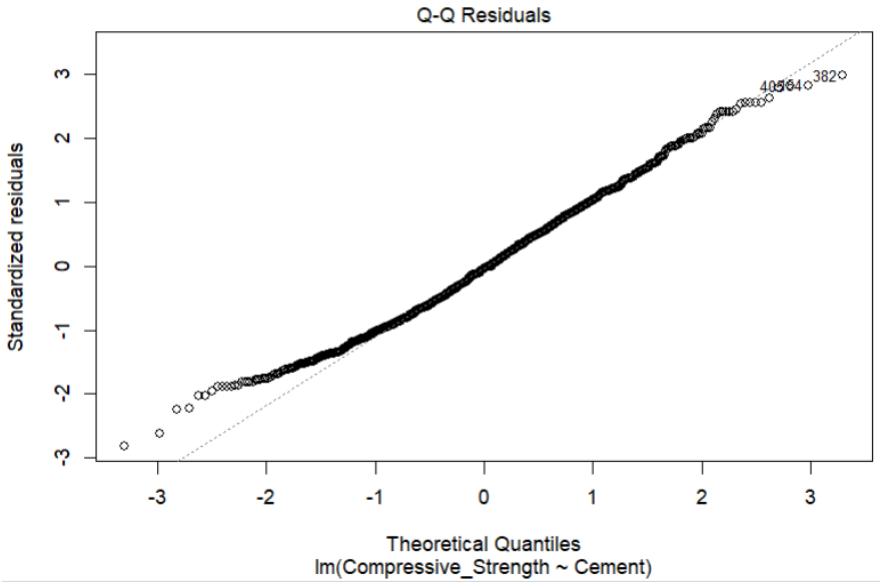
2. Residuals' Independence




```
plot(model_1, 1)
```

Figure 2.21: The correlation is approximately zero, the plot does not have a pattern where the red line is approximately horizontal at zero.

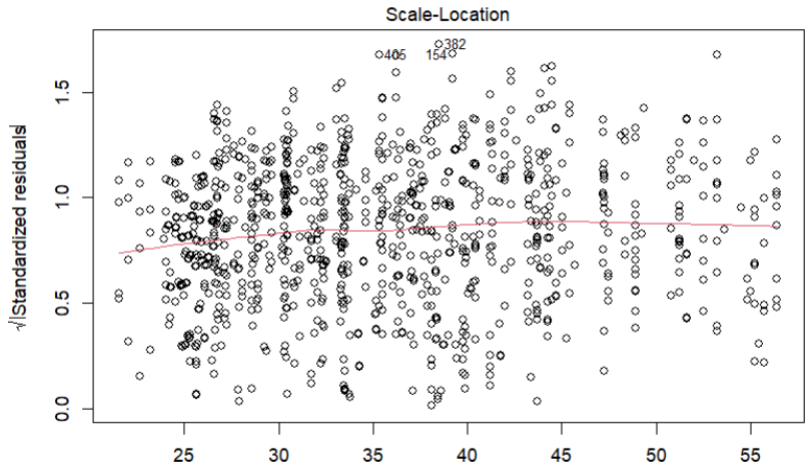
3. Normality of Residuals:




```
plot(model_1, 2)
```

Figure 2.22: The residuals are approximately normally distributed on the Quantum-Quantum Residuals.

4. Equal variances of the residuals (Homoscedasticity)




```
plot(model_1, 3)
```

Figure 2.23: Above is the Scale-Locations plot which checks the Homoscedasticity assumption. It says variance the variance of the residuals is constant.

2.9.2 Multi Linear Regression (MLR)

In this project, Multiple Linear Regression (MLR) was performed to analyze the combined effects of various ingredients on concrete compressive strength. MLR allows us to simultaneously evaluate the influence of variables such as Cement, Superplasticiser, Water, Fine Aggregate, Coarse Aggregate, Fly Ash, and Age. This approach provides a more realistic understanding of how these factors interact in concrete mixes.

The results showed that Cement, Superplasticizer, and Age were significant positive predictors of compressive strength, while Water had a negative effect. Cement, being the main binding agent, plays a critical role in enhancing strength, and superplasticizers improve the mix's workability and allow for a lower water-to-cement ratio, reducing porosity. Age reflects the curing period during which strength development occurs due to the hydration process. Conversely, excessive water content was linked to reduced strength, as it creates voids in the concrete structure, weakening the material (Yeh, 2006).

Multi Linear Regression One (1)

```
# ----------------------MLR equation--------
# ============================================
model_2 <- lm(Compressive_Strength ~
                Cement +
                Superplasticizer
              , data = continuous_data)

summary(model_2)
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     9.190242   1.250302    7.35 4.03e-13 ***
Cement          0.074794   0.004036   18.53  < 2e-16 ***
Superplasticizer 0.902460   0.070603   12.78  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 13.47 on 1027 degrees of freedom
Multiple R-squared:  0.3511,    Adjusted R-squared:  0.3498
F-statistic: 277.8 on 2 and 1027 DF,  p-value: < 2.2e-16
```

Figure 2.24: The two coefficients in the Pr(>|t|) column are significant at the 0.05 level, also the intercept is significant. Compared to the 0.25 value of the prior SLR model, the Adjusted R2 value of 0.35 is significantly higher. It is obvious that there is a positive association between the two variables.

Concrete compressive strength = 0.074794 x Cement + 0.902460 x Superplasticizer

Adjusted R (2) = 0.35 i.e. This Regression equation can predict 35%

Multi Linear Regression Two (2)

```
model_3 <- lm(Compressive_Strength ~
                Cement +
                Superplasticizer +
                Age
              , data = continuous_data)

summary(model_3)
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     5.097652   1.146613   4.446 9.71e-06 ***
Cement          0.068836   0.003628  18.976  < 2e-16 ***
Superplasticizer 1.111650   0.064459  17.246  < 2e-16 ***
Age             0.097900   0.006090  16.077  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 12.05 on 1026 degrees of freedom
Multiple R-squared:  0.4816,    Adjusted R-squared:  0.4801
F-statistic: 317.8 on 3 and 1026 DF,  p-value: < 2.2e-16
```

Figure 2.25: All three of the coefficients in the Pr(>|t|) column are significant at the 0.05 level, also the interception is significant. Compared to the 0.35 value of the prior MLR model, the Adjusted R2 value of 0.48 is significantly higher.

Concrete compressive strength = 0.068836 x Cement + 1.111650 x Superplasticizer + 0.097900 x Age

Adjusted R (2) = 0.48 i.e. This Regression equation can predict 48% of the entire variability in the Compressive strength.

Multi Linear Regression Three (3)

```
model_4 <- lm(Compressive_Strength ~
                Cement +
                Superplasticizer +
                Age + Blast_Furnace_Slag
              , data = Concrete_data)

summary(model_4)
```

```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)        -2.000793   1.207644  -1.657   0.0979 .
Cement              0.081162   0.003517  23.076   <2e-16 ***
Superplasticizer    1.059540   0.060166  17.610   <2e-16 ***
Age                 0.098492   0.005671  17.368   <2e-16 ***
Blast_Furnace_Slag  0.053168   0.004226  12.580   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.22 on 1025 degrees of freedom
Multiple R-squared:  0.551,     Adjusted R-squared:  0.5492
F-statistic: 314.4 on 4 and 1025 DF,  p-value: < 2.2e-16
```

Figure 2.26: All four of the coefficients in the Pr(>|t|) column are significant at the 0.05 level, but the intercept is not. Compared to the 0.48 value of the prior MLR model, the Adjusted R2 value of 0.55 is not significantly higher.

Multi Linear Regression Three (4)

```
model_5 <- lm(Compressive_Strength ~
                Cement +
                Superplasticizer +
                Age +
                Blast_Furnace_Slag +
                Water
              , data = Concrete_data)

summary(model_5)
```

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)        35.491872   4.304056   8.246 4.98e-16 ***
Cement              0.081291   0.003386  24.007  < 2e-16 ***
Superplasticizer    0.613852   0.076040   8.073 1.92e-15 ***
Age                 0.109262   0.005588  19.554  < 2e-16 ***
Blast_Furnace_Slag  0.060130   0.004141  14.520  < 2e-16 ***
Water              -0.197010   0.021775  -9.047  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.8 on 1024 degrees of freedom
Multiple R-squared:  0.5842,    Adjusted R-squared:  0.5822
F-statistic: 287.8 on 5 and 1024 DF,  p-value: < 2.2e-16
```

Figure 2.27: All five of the coefficients in the Pr(>|t|) column are significant at the 0.05 level, but the intercept is not. Compared to the 0.55 value of the prior MLR model, the Adjusted R2 value of 0.58 is not significantly higher. Take note of the water variable's negative sign. It is obvious that there is a negative association between the “**Water**” and “**Compressive strength**” variables if you look at the correlation matrix again.

Making sure the fitted (model_5) meets MLR assumptions

1. Linearity

Drawing a scatter plot matrix of the five (5) variables finding their indices and putting them in the “**c() vector**”, that have the best contribution the concrete compressive strength. The first on the right is the concrete compressive strength.

```r
# Scatterplot matrix with linear regression lines
pairs(
  Concrete_data[, c(11,1, 5, 8,2, 4)],
  lower.panel = NULL,
  upper.panel = function(x, y) {
    points(x, y, pch = 19, cex = 0.2)  # Add points
    abline(lm(y ~ x), col = "red", lwd = 2)  # Add thick regression line
  }
)
```

Figure 2.28: The R-code above select the variables by their number of arrangement in the dataset to show a matrix of linearity between "**IV's**" and the target variable.

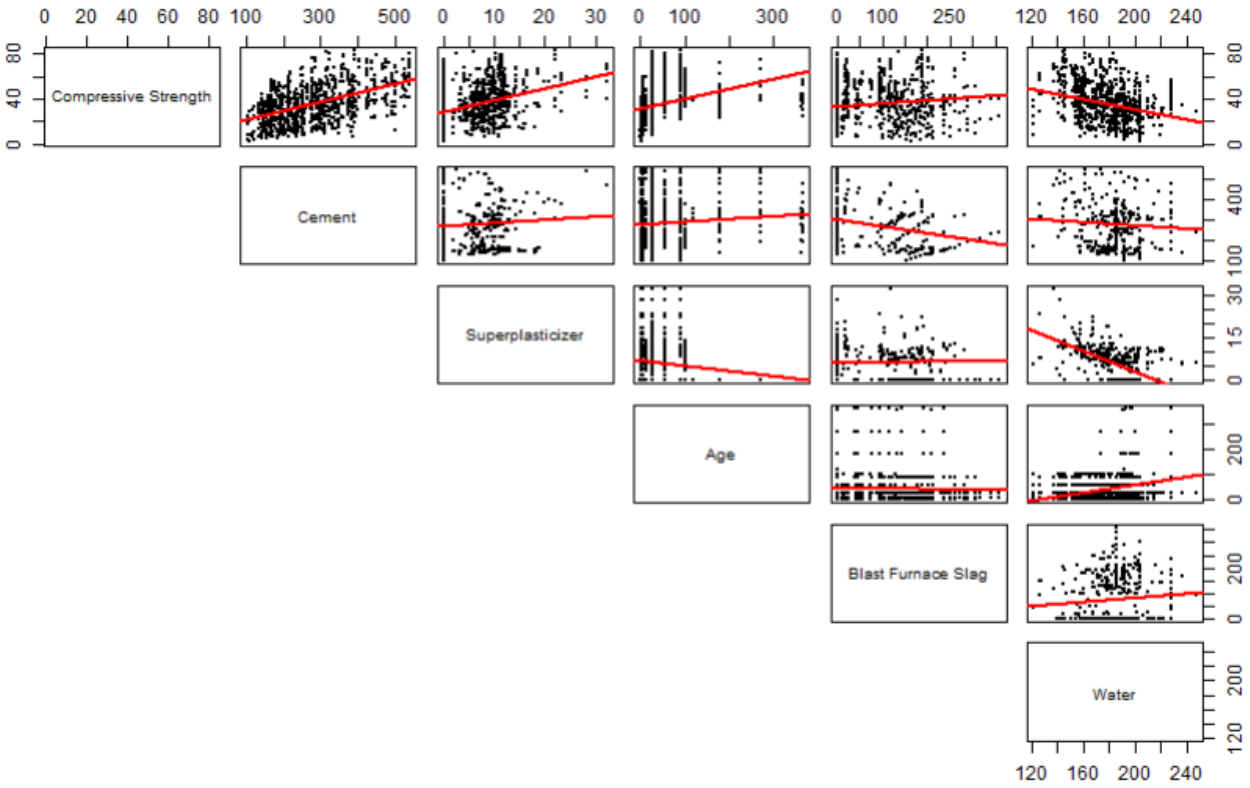


Figure 2.29: On the first row you'd find out all independent variables have an approximately linear relation with the concrete compressive strength.

2. Residuals Independence: model_5

```
plot(model_5, 1)
```

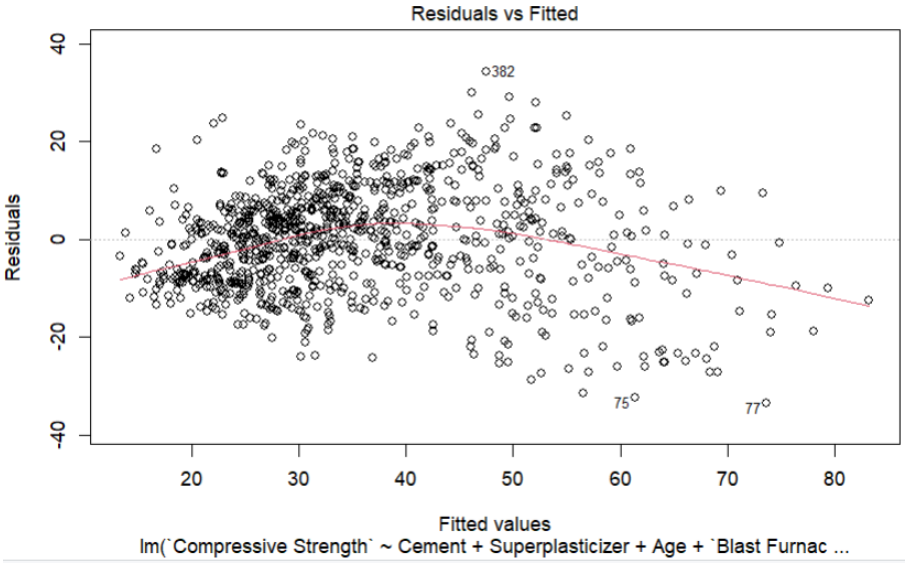


Figure 2.30: From the plot above, the correlation is not approximately "0", and at the tail end it is bent and has not met the assumption. It looks like there is a relationship between residual and fits.

3. Normality of Residuals: model_5

```
plot(model_5, 2)
```

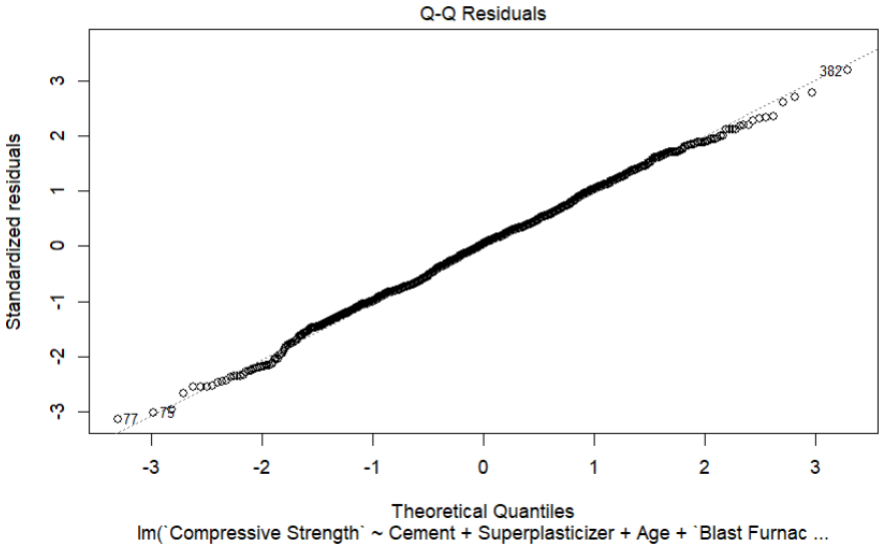


Figure 2.31: The residuals are roughly distributed regularly. The line is close to the observations.

4. Homoscedasticity (equal variances of the residuals): model_5

```
plot(model_5, 3)
```

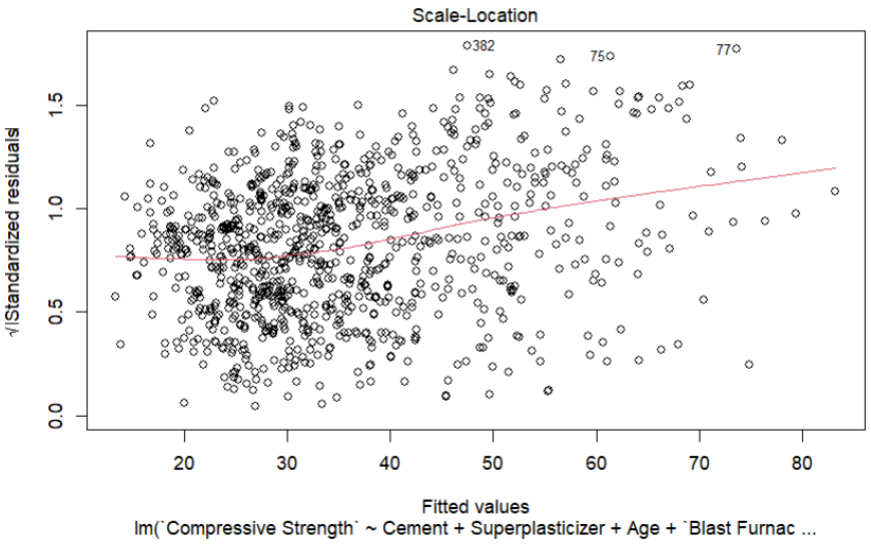


Figure 2.32: The average residuals on the scale-location plot are dispersed randomly with approximately equal variability around the red line, dropping around the value of "1.0". However, there is an upward elevation fall at the tail end of the red line due to inconsistency. This indicates that the residuals' variance is not entirely constant and has nothing to do with a fitted value.

2.9.3 Caring out Log of my fifth MLR model checking its ability to pass all assumptions

```
# Apply log transformation to columns and handle zero or negative values by replacing with 0.001
Concrete_data <- Concrete_data %>%
  mutate(
    log_Cement = log(ifelse(Cement > 0, Cement, 0.001)),
    log_Superplasticizer = log(ifelse(Superplasticizer > 0, Superplasticizer, 0.001)),
    log_Age = log(ifelse(Age > 0, Age, 0.001)),
    log_Blast_Furnace_Slag = log(ifelse(`Blast_Furnace_Slag` > 0, `Blast_Furnace_Slag`, 0.001)),
    log_Water = log(ifelse(Water > 0, Water, 0.001))
  )
```

Figure 2.33: The code above converts my fifth MLR model log scale, handling non-positives

```r
# Fit the MLR model with log-transformed variables
model_log <- lm(Compressive_Strength ~
                  log_Cement +
                  log_Superplasticizer +
                  log_Age +
                  log_Blast_Furnace_Slag +
                  log_Water,
                data = Concrete_data)

# Display the summary of the model
summary(model_log)
```

```
Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            49.08260   13.53489   3.626 0.000302 ***
log_Cement             23.96497    0.63792  37.567  < 2e-16 ***
log_Superplasticizer    0.72059    0.06582  10.949  < 2e-16 ***
log_Age                 8.36556    0.19969  41.893  < 2e-16 ***
log_Blast_Furnace_Slag  0.90371    0.04186  21.591  < 2e-16 ***
log_Water             -33.07530    2.43737 -13.570  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 7.528 on 1024 degrees of freedom
Multiple R-squared:  0.7979,    Adjusted R-squared:  0.7969
F-statistic: 808.6 on 5 and 1024 DF,  p-value: < 2.2e-16
```

Figure 2.34: The R-squared is 0.79, indicates model explains 80% variance, with significant predictor(p-value<0.05)

Residuals Independence: model_log

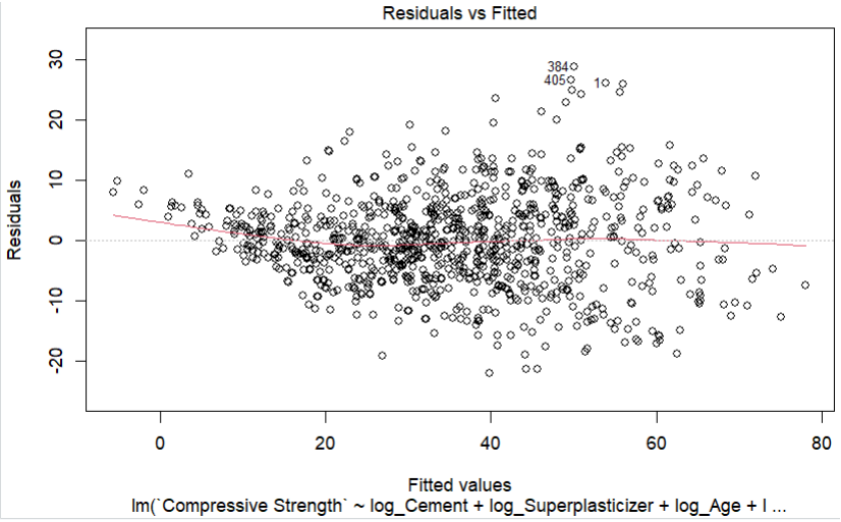


Figure 2.35: From the plot above, the correlation is approximately "0", although and at the tail end it is greatly met the assumption. It looks like there is a relationship between residual and fits.

Normality of Residuals: model_log

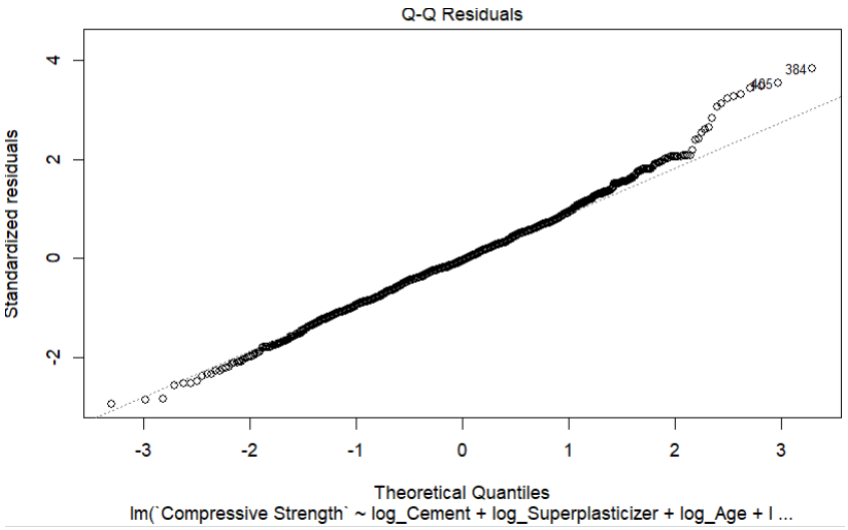


Figure 2.36: The residuals are roughly distributed regularly. The line is close to the observations.

Homoscedasticity (equal variances of the residuals): model_5

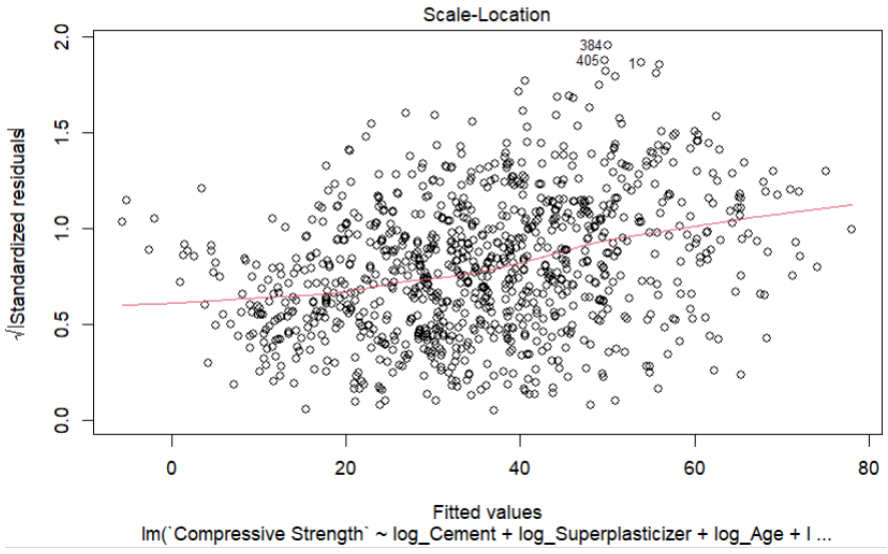


Figure 2.37: On the scale-Location plot, the average residuals are falling around the value of "1.0" they are randomly scattered around the red line with roughly equal variability but at the tail end

of the red line there is a slight fall out of consistency. This reveals that the variance of the residuals is not constant and are related to a fitted value.

Because it considered the combined impacts of several variables, the MLR analysis produced a more accurate estimate of comparism strength than the SLR. This emphasizes the importance of balancing material proportions in achieving optimal concrete performance.

```
# Check multicollinearity with VIF
vif(model_log)
```

```
> vif(model_log)
          log_Cement   log_Superplasticizer                log_Age log_Blast_Furnace_Slag
            1.072308              1.523959               1.027766               1.069246
           log_Water
            1.534515
```

Figure 2.38: The VIF values show low collinearity since they are below 5.

```
# Model evaluation
predictions <- predict(model_log, newdata = Concrete_data)
rmse <- sqrt(mean((predictions - Concrete_data$`Compressive_Strength`)^2))
rsq <- 1 - sum((predictions - Concrete_data$`Compressive_Strength`)^2) /
          sum((Concrete_data$`Compressive_Strength` - mean(Concrete_data$`Compressive_Strength`))^2)

cat("RMSE: ", rmse, "\n")
cat("R-squared: ", rsq, "\n")
```

```
> cat("RMSE: ", rmse, "\n")
RMSE:  7.506172
> cat("R-squared: ", rsq, "\n")
R-squared:  0.7979166
```

Figure 2.39: From the above we can see that by normalizing using Log of the continuous variables we have RMSE as **7.5** while its R-squared as **0.79** (80%)

2.10 Random Forest

```
# ------------------------------------------------------------
newConcrete_data <- dplyr::select(Concrete_data,
                                  Cement,
                                  Blast_Furnace_Slag,
                                  Fly_Ash,
                                  Water,
                                  Superplasticizer,
                                  Coarse_Aggregate,
                                  Fine_Aggregate,
                                  Age,
                                  Concrete_Category,
                                  Contains_Fly_Ash,
                                  Compressive_Strength)

# View the first few rows of the new dataset
head(newConcrete_data)
```

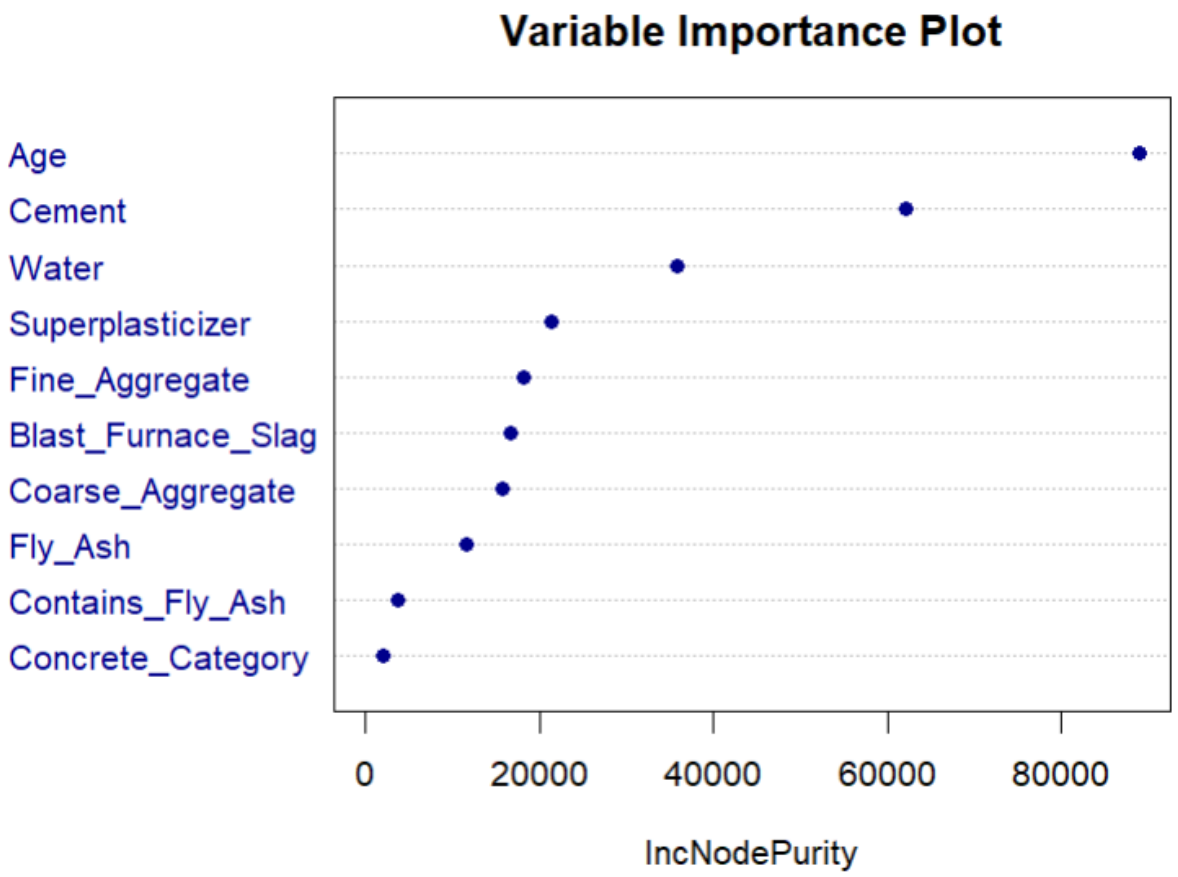
Figure 2.40: In other to carry out random forest non linear model I selected all my variables, as RF can handle categorical variables.

```
# Fit the random forest regression model using all predictors
rf_model <- randomForest(Compressive_Strength ~ ., data = newConcrete_data, ntree = 500)

# Display the model summary
print(rf_model)
```

```
Call:
 randomForest(formula = Compressive_Strength ~ ., data = newConcrete_data,      ntree = 500)
               Type of random forest: regression
                     Number of trees: 500
No. of variables tried at each split: 3

          Mean of squared residuals: 23.14517
                    % Var explained: 91.7
```

Figure 2.41: Above is the result of the Random Forest and we have 92% variance explained. This quality of result is as a result of Random forest non dependency on linearity of variables.

```r
# Plot the variable importance
varImpPlot(rf_model, main = "Variable Importance Plot", col = "darkblue", pch = 19, cex = 1.2)
```




```r
# Predict using the random forest model
rf_predictions <- predict(rf_model, newdata = newConcrete_data)

# Evaluate model performance
rf_rmse <- sqrt(mean((rf_predictions - newConcrete_data$Compressive_Strength)^2))
cat("Random Forest Model RMSE: ", rf_rmse, "\n")
```

```
Random Forest Model RMSE:  2.628103
```

Figure 2.42 Above is the plot of the variable importance of each variable based on the Node of Purity. It has an RMSE of 2.628103

2.11 XGBoost Model

```
# Handle categorical variables by converting them into factors or dummy variables
newConcrete_data$Concrete_Category <- as.factor(newConcrete_data$Concrete_Category)
newConcrete_data$Contains_Fly_Ash <- as.factor(newConcrete_data$Contains_Fly_Ash)

# Select predictor variables (all columns except 'Compressive_Strength')
X <- dplyr::select(newConcrete_data, -Compressive_Strength)

# Convert all columns to numeric (including factors) using model.matrix() for one-hot encoding of cate
X_matrix <- model.matrix(~ ., data = X)

# Convert target variable to numeric
y <- newConcrete_data$Compressive_Strength
y_matrix <- as.matrix(y)

# Fit the XGBoost model
xgb_model <- xgboost(data = X_matrix, label = y_matrix, objective = "reg:squarederror", nrounds = 500)

# Display the model summary
print(xgb_model)
```

Figure 2.43: Converting categorical variables, selecting predictors and fit XGBoost regression model.

```
# Predict using the XGBoost model
xgb_predictions <- predict(xgb_model, newdata = X_matrix)

# Calculate RMSE for XGBoost model
xgb_rmse <- sqrt(mean((xgb_predictions - y_matrix)^2))
cat("XGBoost Model RMSE: ", xgb_rmse, "\n")
```

```
##### xgb.Booster
raw: 1.6 Mb
call:
  xgb.train(params = params, data = dtrain, nrounds = nrounds,
    watchlist = watchlist, verbose = verbose, print_every_n = print_every_n,
    early_stopping_rounds = early_stopping_rounds, maximize = maximize,
    save_period = save_period, save_name = save_name, xgb_model = xgb_model,
    callbacks = callbacks, objective = "reg:squarederror")
params (as set within xgb.train):
  objective = "reg:squarederror", validate_parameters = "TRUE"
xgb.attributes:
  niter
callbacks:
  cb.print.evaluation(period = print_every_n)
  cb.evaluation.log()
# of features: 11
niter: 500
nfeatures : 11
evaluation_log:
  iter train_rmse
 <num>      <num>
     1 28.0662088
     2 20.2937453
   ---        ---
   499  0.9106941
   500  0.9106926
```

```
XGBoost Model RMSE:  0.9106925
```

Figure 2.44: The code above will predict XGBoost and calculate RMSE delaying model performance which results to 0.9106925.

```
# -------------------- Performance Comparison ------------------

# The model with the lower RMSE is the better performing model
if (rf_rmse < xgb_rmse) {
  cat("Random Forest performs better than XGBoost.\n")
} else if (xgb_rmse < rf_rmse) {
  cat("XGBoost performs better than Random Forest.\n")
} else {
  cat("Both Random Forest and XGBoost perform equally well.\n")
}
```

```
XGBoost performs better than Random Forest.
```

Figure 2.45: The XGBoost performs better than the Random Forest.

```
# Calculate R-squared for Random Forest model
rf_rsq <- 1 - sum((rf_predictions - y)^2) / sum((y - mean(y))^2)
cat("Random Forest R-squared: ", rf_rsq, "\n")

# Calculate R-squared for XGBoost model
xgb_rsq <- 1 - sum((xgb_predictions - y)^2) / sum((y - mean(y))^2)
cat("XGBoost R-squared: ", xgb_rsq, "\n")
```

```
> rf_rsq <- 1 - sum((rf_predictions - y)^2) / sum((y - mean(y))^2)
> cat("Random Forest R-squared: ", rf_rsq, "\n")
Random Forest R-squared:  0.975227
>
> # Calculate R-squared for XGBoost model
> xgb_rsq <- 1 - sum((xgb_predictions - y)^2) / sum((y - mean(y))^2
> cat("XGBoost R-squared: ", xgb_rsq, "\n")
XGBoost R-squared:  0.9970253
```

Figure 2.46: As seen above, where Random Forest has R-squared of **0.98%** XGBoost R-squared has **0.997%**

2.12 Hypothesis testing:

Hypothesis testing evaluates specific research questions about the predictors of concrete strength, such as effects in superplasticizer, water, and fly ash. By using statistics test like ANOVA, t-tests, analysis determines the significance of these factors, contributing to the understanding of strength variation (Yeh, 1998).

```
# Hypothesis 1: Does Superplasticizer significantly affect Compressive Strength?
model_superplasticizer <- lm(Compressive_Strength ~ Superplasticizer, data = continuous_data)
summary(model_superplasticizer)
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     29.48707    0.71360   41.32   <2e-16 ***
Superplasticizer  1.02979    0.08809   11.69   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 15.48 on 1028 degrees of freedom
Multiple R-squared:  0.1173,    Adjusted R-squared:  0.1165
F-statistic: 136.7 on 1 and 1028 DF,  p-value: < 2.2e-16
```

Figure 2.47: With positive coefficient and strong statistical evidence, superplasticizer significantly improved compressive strength

```
# Hypothesis 2: Does Water content reduce Compressive Strength?
cor_test_water <- cor.test(continuous_data$Water, continuous_data$Compressive_Strength)
cor_test_water
```

```
        Pearson's product-moment correlation

data:  continuous_data$Water and continuous_data$Compressive_Strength
t = -10.084, df = 1028, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.3546199 -0.2434138
sample estimates:
       cor
-0.3000359
```

Figure 2.49: With compressive strength of statistically significant p-value of (2e-16) water has a negative correlation with compressive strength.

```r
# Hypothesis 3: Does Fly Ash presence reduce Compressive Strength?
t_test_fly_ash <- t.test(Compressive_Strength ~ Contains_Fly_Ash, data = Concrete_data)
summary(t_test_fly_ash)
```

```
> summary(t_test_fly_ash)
            Length Class  Mode
statistic   1      -none- numeric
parameter   1      -none- numeric
p.value     1      -none- numeric
conf.int    2      -none- numeric
estimate    2      -none- numeric
null.value  1      -none- numeric
stderr      1      -none- numeric
alternative 1      -none- character
method      1      -none- character
data.name   1      -none- character
```

Figure 2.50: The p-value indicates significance in the t-test comparing fly ash to compressive strength.

```r
# Hypothesis 4: Is Age positively correlated with Compressive Strength?
age_corr <- cor.test(continuous_data$Age, continuous_data$Compressive_Strength)
age_corr
```

```
> age_corr

        Pearson's product-moment correlation

data:  continuous_data$Age and continuous_data$Compressive_Strength
t = 18.497, df = 1028, p-value < 2.2e-16
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 0.4524405 0.5441858
sample estimates:
      cor
0.4997134
```

Figure 2.51: The age positively corelates with Compressive strength(cor = 0.4997)

```r
# Hypothesis 5: Does Concrete Category significantly affect Compressive Strength?
anova_category <- aov(Compressive_Strength ~ Concrete_Category, data = Concrete_data)
summary(anova_category)
```

```
> summary(anova_category)
                   Df Sum Sq Mean Sq F value Pr(>F)
Concrete_Category    1    853   853.3   3.154  0.076 .
Residuals         1028 278074   270.5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 2.52: With (p-value = 0.076), ANOVA shows no significant difference in compression strength across the concrete categories.

```
# Two-Way ANOVA - Interaction Effect between Concrete Category and Fly Ash Presence
two_way_anova <- aov(Compressive_Strength ~ Concrete_Category * Contains_Fly_Ash, data = Concrete_data)
summary(two_way_anova)
```

```
> summary(two_way_anova)
                                  Df Sum Sq Mean Sq F value Pr(>F)
Concrete_Category                  1    853   853.3   3.176 0.0750 .
Contains_Fly_Ash                   1    813   813.4   3.028 0.0822 .
Concrete_Category:Contains_Fly_Ash  1   1642  1642.0   6.112 0.0136 *
Residuals                        1026 275619   268.6
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figure 2.53: Between Concrete categories and Contains Fly Ash, the (p-value = 0.0136) indicates combined effects influence on compressive strength.

```
# Conclusion

cat("Summary of Findings:\n")
cat("1. Cement and Superplasticizer are significant predictors of compressive strength.\n")
cat("2. Positive correlation between Age and compressive strength.\n")
cat("3. ANOVA shows significant differences in strength across concrete categories.\n")
cat("4. Water has a weak negative association with compressive strength.\n")

cat("\nConclusion:\n")
cat("This analysis provides insights on factors affecting concrete compressive strength, with Cement, Age, and
cat("Further investigation with non-linear models or interaction terms could yield additional insights.")
```

```
> # ---------------------------------------------
> # Conclusion
>
> cat("Summary of Findings:\n")
Summary of Findings:
> cat("1. Cement and Superplasticizer are significant predictors of compressive strength.\n")
1. Cement and Superplasticizer are significant predictors of compressive strength.
> cat("2. Positive correlation between Age and compressive strength.\n")
2. Positive correlation between Age and compressive strength.
> cat("3. ANOVA shows significant differences in strength across concrete categories.\n")
3. ANOVA shows significant differences in strength across concrete categories.
> cat("4. Water has a weak negative association with compressive strength.\n")
4. Water has a weak negative association with compressive strength.
>
> cat("\nConclusion:\n")

Conclusion:
> cat("This analysis provides insights on factors affecting concrete compressive strength, with Cement, Age, and Supe
rplasticizer identified as key variables.\n")
This analysis provides insights on factors affecting concrete compressive strength, with Cement, Age, and Superplasti
cizer identified as key variables.
> cat("Further investigation with non-linear models or interaction terms could yield additional insights.")
Further investigation with non-linear models or interaction terms could yield additional insights.
```

Figure 2.47: Conclusion

Conclusion

With an R-squared of 99.7%, the top-performing model **XGBoost** outperformed the others in terms of accuracy. It was perfect for forecasting the compressive strength of concrete because of its capacity to manage intricate linkages and nonlinear interaction between variables. The model indicated that age, cement, and superplasticiser were important predictors, and that too much water had a detrimental effect on strength. These results highlight how crucial it is to maximise material proportions to attain the intended performance. Engineers can improve durability and safety in building applications by using this knowledge to design exact concrete mixtures. The findings offer a strong basis for material balancing, guaranteeing better structural integrity and effective resource use.