STA2005S - Regression Assignment

true true

2024-10-17

Abstract

In this report, we explored the efficiency of 6 programming languages through the approximation of π. We found that efficiency of various programming languages can vary widely, with C and C++ being the most efficient programming languages. We also presented evidence for compiled languages having better performance than interpreted languages. Our results suggest that programmers can benefit from taking the efficiency of various programming languages into account, rather than simply opting for simplicity in the syntax of these languages .

0.1 Part One : Analysis

1 Section 1: Introduction

Air pollution, particularly high levels of particulate matter (PM), is a major environmental and public health issue in South Africa's urban centers. Exposure to elevated PM levels is linked to respiratory diseases and other serious health conditions. Understanding the factors influencing PM concentrations is crucial for developing policies that improve air quality and protect public health. This analysis seeks to identify the key drivers of air pollution in South Africa's cities, focusing on how various urban, environmental, and socioeconomic factors affect particulate matter levels.
Unknown Factors to Investigate:
Traffic Density: How do varying levels of vehicle traffic contribute to PM levels in different areas?
Industrial Activity: What is the impact of industrial activity near monitoring stations on air quality?
Temperature & Humidity: How do changes in weather conditions, like temperature and humidity, influence PM concentrations?
Wind Speed: How does wind speed affect the dispersion or accumulation of particulate matter in urban areas?
Day of the Week & Public Holidays: Do patterns of human activity on weekdays, weekends, and holidays significantly influence pollution levels?
Urban Greenery: How effective are green spaces in reducing air pollution in densely populated areas?

2 Objective

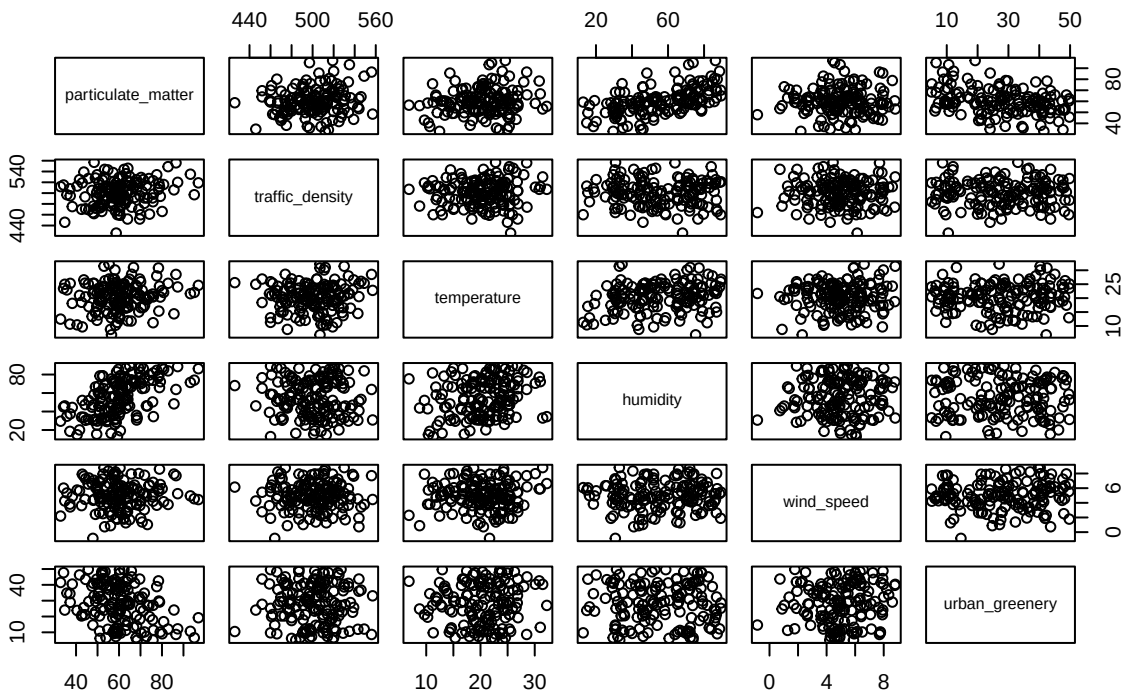
The goal of this analysis is to explore the relationships between PM levels and these explanatory variables. By identifying the most influential factors, we aim to inform urban planning and public health strategies that address air pollution and improve the quality of life in South African cities.

2.1 Section 2 : Data Exploration

density plot

pairwsie plots

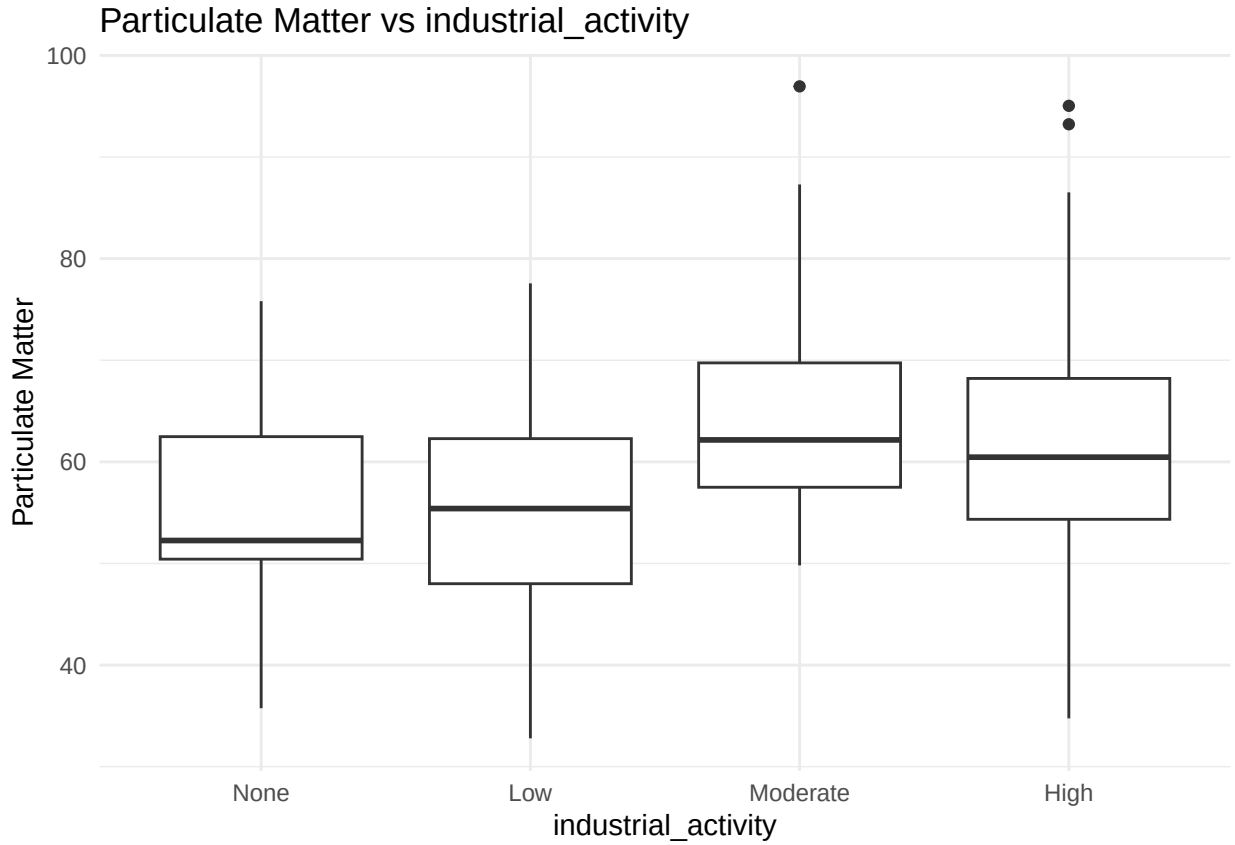
```
continuous_vars <- data_tidy_air_quality[, sapply(data_tidy_air_quality, is.numeric)]
pairs(continuous_vars, main = "Pairwise Scatterplots of Continuous Variables")
```

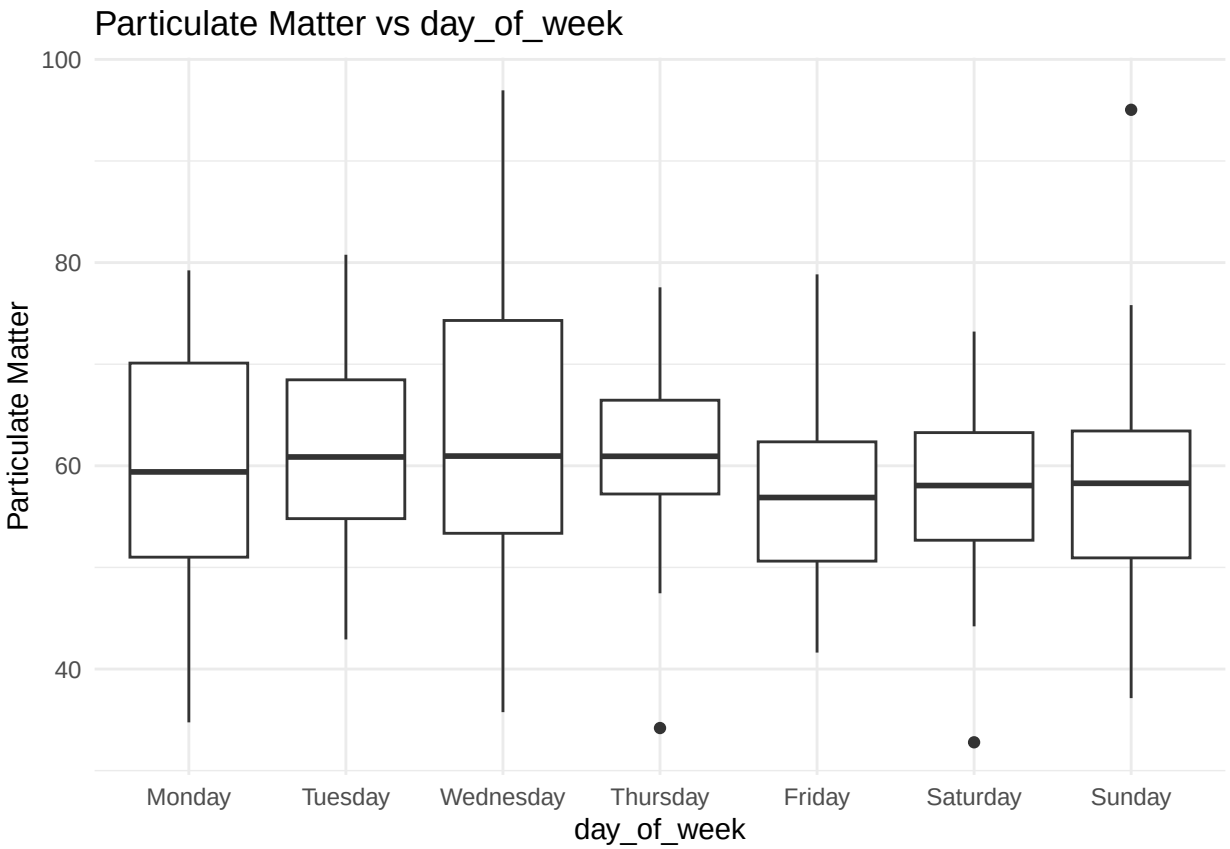


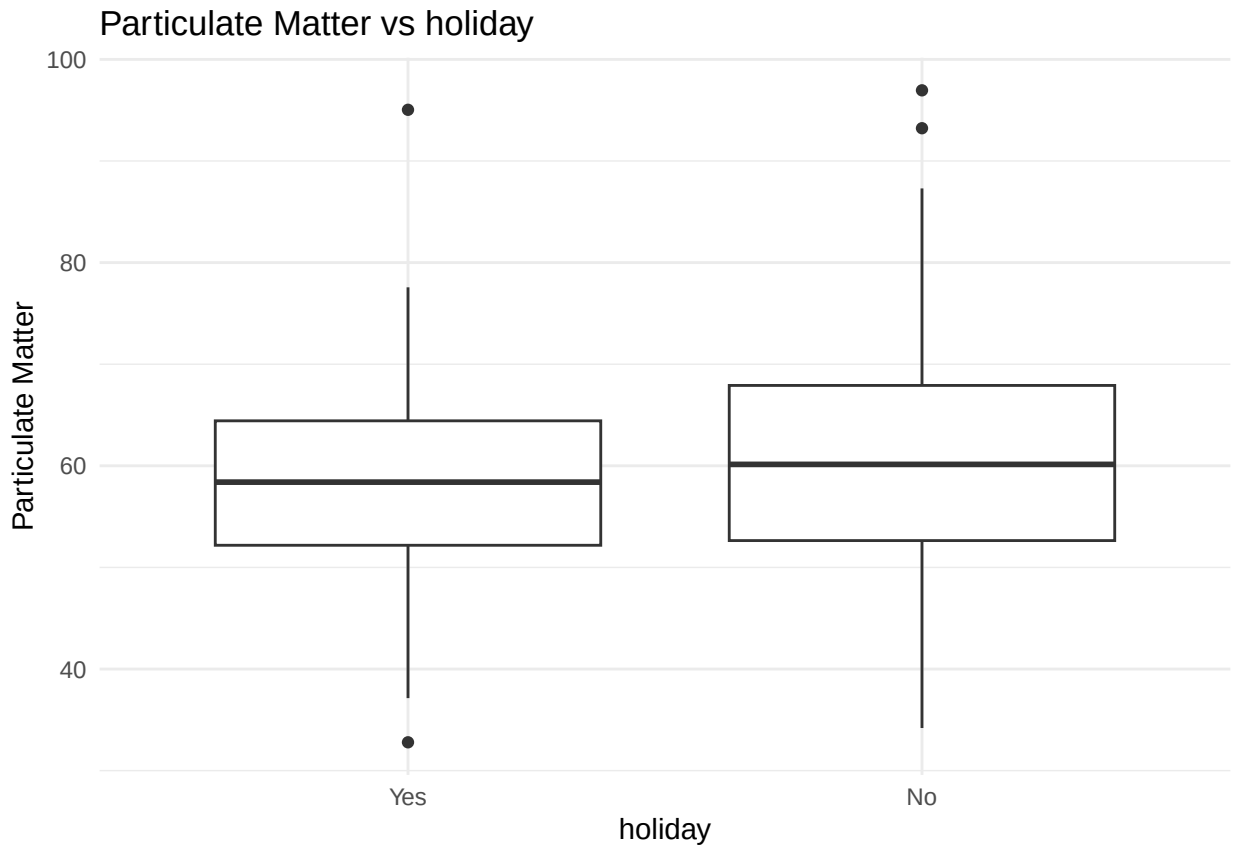
categorial variable plots

```
data_tidy_air_quality$industrial_activity <- factor(data_tidy_air_quality$industrial_activity,
                                    levels = c("None","Low", "Moderate", "High"))  # Adjust the levels a

data_tidy_air_quality$day_of_week <- factor(data_tidy_air_quality$day_of_week,
                              levels = c("Monday", "Tuesday", "Wednesday",
                                         "Thursday", "Friday", "Saturday", "Sunday"))

data_tidy_air_quality$holiday <- factor(data_tidy_air_quality$holiday,
                              levels = c("Yes", "No"))

categorical_vars <- names(data_tidy_air_quality)[sapply(data_tidy_air_quality, is.factor)]


for (var in categorical_vars) {
  plt<- ggplot(data_tidy_air_quality, aes_string(x = var, y = "particulate_matter")) +
    geom_boxplot() +
    labs(title = paste("Particulate Matter vs", var),
         x = var,
         y = "Particulate Matter") +
    theme_minimal()


  print(plt)  # Print the plot
}
```

```
## Warning: ‘aes_string()‘ was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with ‘aes()‘.
## i See also ‘vignette("ggplot2-in-packages")‘ for more information.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```




Particulate Matter vs day_of_week
Particulate Matter
100
80
60
40
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
day_of_week



tabular representation of relationship between categorial variables

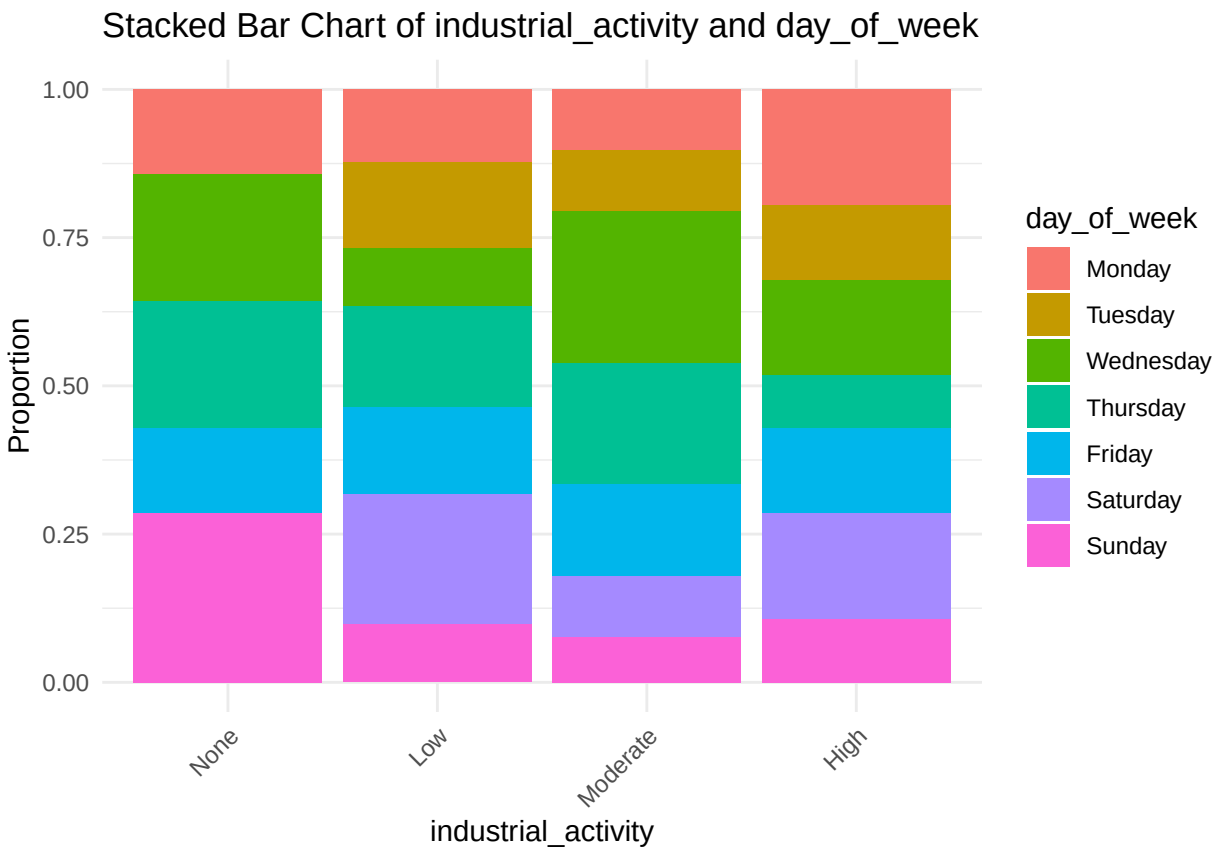
```
for (i in 1:(length(categorical_vars)-1)) {
  for (j in (i+1):length(categorical_vars)) {
    cat("Contingency Table for", categorical_vars[i], "and", categorical_vars[j], "\n")
    print(table(data_tidy_air_quality[[categorical_vars[i]]], data_tidy_air_quality[[categorical_vars[j]
    cat("\n")
  }
}
```

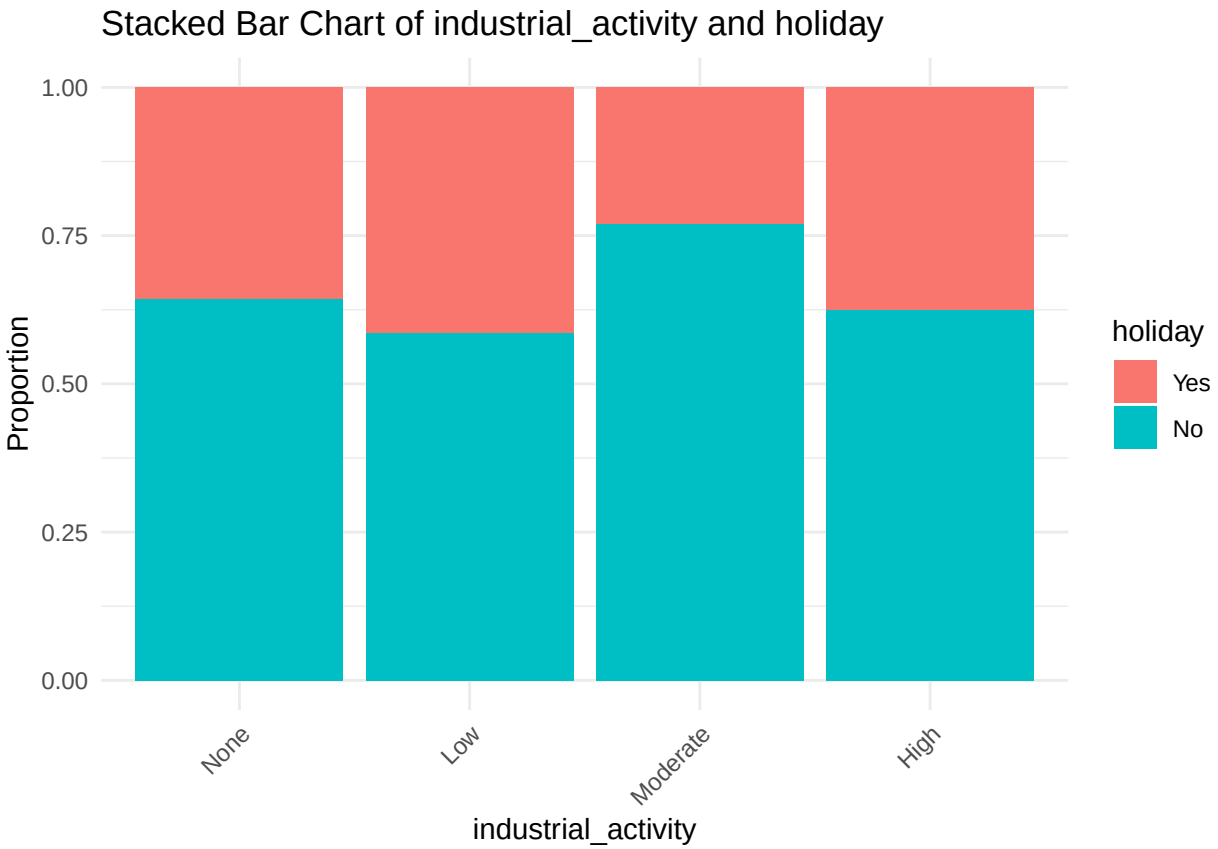
```
## Contingency Table for industrial_activity and day_of_week
##
##            Monday Tuesday Wednesday Thursday Friday Saturday Sunday
##   None          2       0         3        3      2        0      4
##   Low           5       6         4        7      6        9      4
##   Moderate      4       4        10        8      6        4      3
##   High         11       7         9        5      8       10      6
##
## Contingency Table for industrial_activity and holiday
##
##            Yes No
##   None       5  9
##   Low       17 24
##   Moderate   9 30
##   High      21 35
##
## Contingency Table for day_of_week and holiday
```

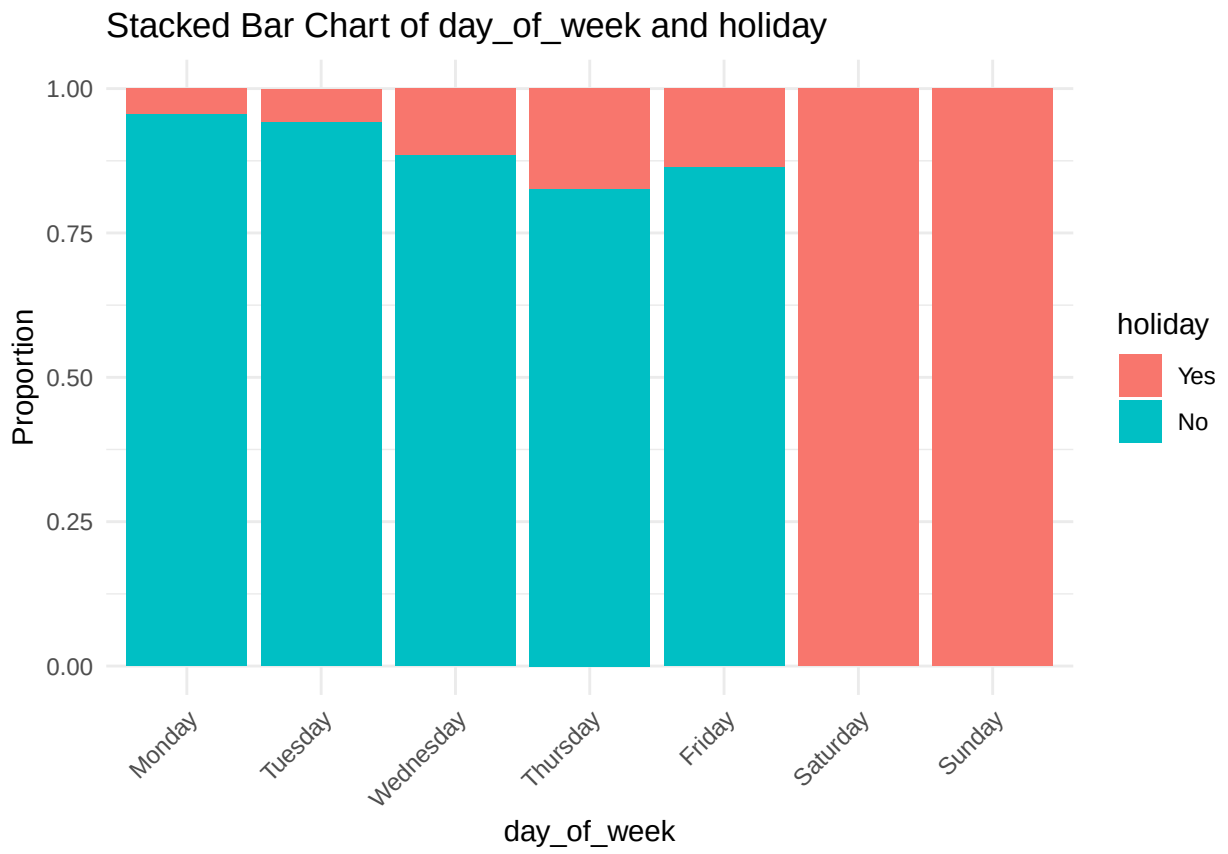
```
##
##             Yes No
##   Monday      1 21
##   Tuesday     1 16
##   Wednesday   3 23
##   Thursday    4 19
##   Friday      3 19
##   Saturday   23  0
##   Sunday     17  0
```

visual representation of relationship between categorial variables

```
for (i in 1:(length(categorical_vars) - 1)) {
  for (j in (i + 1):length(categorical_vars)) {
    # Create the plot
    p <- ggplot(data_tidy_air_quality, aes_string(x = categorical_vars[i], fill = categorical_vars[j])) +
      geom_bar(position = "fill") +  # Use "fill" to make it a stacked bar chart (proportions)
      labs(title = paste("Stacked Bar Chart of", categorical_vars[i], "and", categorical_vars[j]),
           x = categorical_vars[i],
           y = "Proportion") +
      theme_minimal() +
      theme(axis.text.x = element_text(angle = 45, hjust = 1))

    # Print the plot
    print(p)
  }
}
```


Stacked Bar Chart of industrial_activity and day_of_week
1.00
0.75
0.50
0.25
0.00
Proportion
None
Low
Moderate
High
industrial_activity
day_of_week
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday


Stacked Bar Chart of industrial_activity and holiday
1.00
0.75
0.50
0.25
0.00
Proportion
None
Low
Moderate
High
industrial_activity
holiday
Yes
No



comments
distribution characterisitcs
The distribution of particulate matter levels is generally right-skewed, indicating that a small number of observations have significantly high levels of particulate matter while most observations are clustered at lower levels. The presence of outliers suggests variations in local conditions affecting air quality.
Observed Relationships
1. Traffic Density: A positive correlation exists between particulate matter levels and traffic density, suggesting that areas with higher vehicle traffic tend to experience elevated levels of particulate matter.
2. Urban Greenery: A negative trend is observed, where higher urban greenery correlates with lower particulate matter, indicating that vegetation may help mitigate air pollution.
3. Temperature and Wind Speed: No strong relationship was identified between particulate matter and temperature. However, there is a slight negative correlation with wind speed, indicating that higher wind speeds may help disperse particulate matter.

Potential Collinearity
Some potential collinearity is observed among the explanatory variables, particularly between traffic density and urban greenery. High traffic areas often have less vegetation, leading to a relationship that may confound the analysis. Additionally, temperature and wind speed may also exhibit collinearity, as changes in one could affect the other.

3 Section 3

simple linear regression

```r
X <- cbind(1,data_tidy_air_quality$traffic_density)

Y <-data_tidy_air_quality$particulate_matter
bhat <- solve(t(X) %*% X) %*% t(X) %*% Y

Cmat <- solve(t(X) %*% X)

k <- ncol(X)
rss <- t(Y - X %*% bhat) %*% (Y - X %*% bhat)
# Calculate s2 = RSS/(n-k)
s2 <- as.numeric((rss)/148)
s2
```

```
## [1] 143.5745
```

```r
c_ii <- diag(Cmat)

std.error <- sqrt(s2 * c_ii)
std.error
```

```
## [1] 20.37801682  0.04065266
```

```r
mod1<-lm(data_tidy_air_quality$particulate_matter ~ data_tidy_air_quality$traffic_density, data = data_

summary(mod1)
```

```
##
## Call:
## lm(formula = data_tidy_air_quality$particulate_matter ~ data_tidy_air_quality$traffic_density,
##     data = data_tidy_air_quality)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -28.332  -7.561  -1.050   6.110  35.243
##
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                         18.11537   20.37802   0.889   0.3755
## data_tidy_air_quality$traffic_density  0.08400    0.04065   2.066   0.0406 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.98 on 148 degrees of freedom
## Multiple R-squared:  0.02804,    Adjusted R-squared:  0.02147
## F-statistic: 4.269 on 1 and 148 DF,  p-value: 0.04055
```

hypthesis test

```r
# Summary of ANOVA results
summary(aov(particulate_matter ~ industrial_activity, data = data_tidy_air_quality))
```

```
##                       Df Sum Sq Mean Sq F value Pr(>F)
## industrial_activity    3   2182   727.3   5.396 0.0015 **
## Residuals            146  19680   134.8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Calculate F-statistic and p-value manually
group_means <- tapply(data_tidy_air_quality$particulate_matter, data_tidy_air_quality$industrial_activi
overall_mean <- mean(data_tidy_air_quality$particulate_matter)

# Calculate SST
SST <- sum((data_tidy_air_quality$particulate_matter - overall_mean)^2)

# Calculate SSB
n <- table(data_tidy_air_quality$industrial_activity)
SStreatment <- sum(n * (group_means - overall_mean)^2)

# Calculate SSW
group_means_vector <- unlist(tapply(data_tidy_air_quality$particulate_matter, data_tidy_air_quality$indu
SSerror <- sum((data_tidy_air_quality$particulate_matter - group_means_vector)^2)

# Calculate degrees of freedom
k <- length(unique(data_tidy_air_quality$industrial_activity))
N <- nrow(data)
DFtreatment <- k - 1
DFerror <- 150 - k

# Calculate Mean Squares
MStreatment <- SStreatment / DFtreatment
MSerror <- SSerror / DFerror


# Calculate F-statistic
F_statistic <- MStreatment/MSerror

# Output F-statistic
F_statistic
```

```
## [1] 5.395959
```

```r
# Calculate p-value
p_value <- pf(F_statistic, DFtreatment, DFerror, lower.tail = FALSE)
p_value
```

```
## [1] 0.001502236
```

4 Question 4

Table 1: Confidence Interval for each Coefficients

	2.5 %	Estimate	97.5 %
Intercept			
(Intercept)	-21.0568	13.7937	48.6442
Traffic Density			
traffic_density	0.0155	0.0799	0.1444
Industrial Activity			
industrial_activityLow	-3.1721	2.6589	8.4900
industrial_activityModerate	0.6047	6.4545	12.3043
industrial_activityHigh	-0.2503	5.3652	10.9806
Natural Factors			
temperature	-1.1521	-0.2815	0.5891
humidity	-0.1111	0.1926	0.4962
wind_speed	-0.8040	0.0193	0.8426
temperature:humidity	-0.0088	0.0061	0.0209
Day of Week			
day_of_weekTuesday	-5.9877	0.0133	6.0142
day_of_weekWednesday	-5.3501	0.1565	5.6630
day_of_weekThursday	-5.5367	0.1662	5.8690
day_of_weekFriday	-8.0602	-2.4221	3.2161
day_of_weekSaturday	-12.3605	-4.4832	3.3940
day_of_weekSunday	-10.2167	-2.0885	6.0396
Holiday			
holidayNo	-6.7151	-0.9961	4.7228
Urban Greenery			
urban_greenery	-0.4142	-0.2954	-0.1766

4.1 Hypothesis Testing

We'd like to perform hypothesis tests on the following variables: Temperature, Humidity, Industrial Levels, and Day of Week.

We'll start by examining whether Temperature has an effect on the concentration of Particulate Matter. We'll use the following We use the following set of hypothesis:

$$H_0 : \beta_{temp} = \beta_{hum:temp} = 0$$
$$H_A : \beta_{temp} \neq 0 \text{ and } \beta_{hum:temp} \neq 0$$

This can be done by comparing the restricted and un restricted model:

$$Y_R = \beta_0 + \beta_{traffic} X +$$

```
model_unrestricted <- lm(particulate_matter ~ . +
                         temperature:humidity,
                         data=data_tidy_air_quality)
model_restricted <- update(model_unrestricted, .~.
                           - temperature
                           - temperature:humidity)
anova(model_restricted, model_restricted)
```

Using the anova function in R, we compare the two models with F test. The F test yields a P value 0.6815, suggesting that temperature doesn't have a significant effect on the concentration of particular matter.

We now test for the effect of humidity. Repeating the same procedure, we obtain a P value < 0.00001. Suggesting that it's likely that humidity has an effect on the concentration of particulate matters.

$$H_0 : \beta_{hum} = \beta_{hum:temp} = 0$$
$$H_A : \beta_{hum} \neq 0 \text{ and } \beta_{hum:temp} \neq 0$$

```
model_unrestricted <- lm(particulate_matter ~ . +
                         temperature:humidity,
                         data=data_tidy_air_quality)
model_restricted <- update(model_unrestricted, .~.
                           - humidity
                           - temperature:humidity)
anova(model_restricted, model_unrestricted)
```

4.1.1 Categorical Variables

For the day of week, we take Monday as the reference category and test for the following set of hypothesis, using the same procedure.

$H_0 : \beta_{Tuesday} = \beta_{Wednesday} = \beta_{Thursday} = \beta_{Friday} = \beta_{Saturday} = \beta_{Sunday} = 0$
$H_A : \beta_{Tuesday} \neq 0$ and $\beta_{Wednesday} \neq 0$ and $\beta_{Thursday} \neq 0$ and $\beta_{Friday} \neq 0$ and $\beta_{Saturday} \neq 0$ and $\beta_{Sunday} \neq 0$

```
data_tidy_air_quality$day_of_week <-
  relevel(factor(data_tidy_air_quality$day_of_week), ref="Monday")
model_unrestricted <- lm(particulate_matter ~ .,
                         data=data_tidy_air_quality)
model_restricted <- update(model_unrestricted, .~.
                           - day_of_week)
anova(model_restricted, model_unrestricted)
```

The P value is 0.7735, indicating that there is no evidence that supports rejecting the null hypothesis.

We do the same for industrial activity, taking No Activity as the reference category to test for the set of hypothesis:

$$H_0 : \beta_{Low} = \beta_{Moderate} = \beta_{High} = 0$$
$$H_A : \beta_{Low} \neq 0 \text{ and } \beta_{Moderate} \neq 0 \text{ and } \beta_{High} \neq 0$$

```
data_tidy_air_quality$industrial_activity <- relevel(factor(data_tidy_air_quality$industrial_activity),
model_unrestricted <- lm(particulate_matter ~ ., data_tidy_air_quality)
model_restricted <- lm(particulate_matter ~ . - industrial_activity, data_tidy_air_quality)
anova(model_unrestricted, model_restricted)
```

4.2 Intepretation

From the summary output and the F test for Natural Factors, we get that only Moderate Industrial Activity (p-value: 0.03), Traffic Density (p-value: 0.0155), Urban Greenery (p-value: < 0.0001), and Humidity (p-value: < 0.0001) have a significant effect (p value less than or equal to 0.05) on the concentration of particulate matters.

The average increase in concentration of particulate matter by Moderate Industrial Activity is estimated to be 6.4545 $\frac{\mu g}{m^3}$(CI95%: 0.6047, 12.3043), indicating a significant positive association between the two factors. The average increase in concentration of particulate matter caused by Traffic Density, that is one extra vehicle per hour, is estimated to be 0.0799 $\frac{\mu g}{m^3}$ (CI95%: 0.0155, 0.1444), indicating a positive association. The average decrease in concentration of particulate matter caused by a unit in the percentage of area covered by Urban Greenery is estimated to be 0.2954 $\frac{\mu g}{m^3}$ (CI95%:-0.4142, -0.1766), indicating a negative association between the two. The effect of humidity can interact with temperature, and is thus not independent of other factors. Using the anova model, we can only deduce that there is likely an effect, but we can't be certain about the confidence interval.

5 part 2

scenario A

```
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 lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x dplyr::group_rows() masks kableExtra::group_rows()
## x dplyr::lag()        masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
# Set the seed for reproducibility
set.seed(123)

# Number of simulations
n_simulations <- 1000

temperature <- data_tidy_air_quality$temperature

# Initialize a vector to store whether the null hypothesis was rejected in each simulation
reject_null <- numeric(n_simulations)


# Variance of the uniform distribution needs to be 100, so we calculated b = 17.32
a <- -17.32
b <- 17.32

# Run simulations
for (i in 1:n_simulations) {
```

```r
  # Generate error term e ~ Uniform(a, b)
  e <- runif(length(temperature), min = a, max = b)

  # Generate Y = 30 + e (since b1 = 0)
  Y <- 30 + e


  # Fit the linear model Y = b0 + b1 * temperature
  model <- lm(Y ~ temperature)

  # Perform hypothesis test on b1 (null hypothesis: b1 = 0) and extract from lm
  p_value <- summary(model)$coefficients[2, 4]
  p_value

  # Record if the null hypothesis is rejected (a = 0.05)
  reject_null[i] <- ifelse(p_value < 0.05, 1, 0)

  }

# Calculate Type I error rate (proportion of times the null was incorrectly rejected)
type_1_error_rate <-(1000-sum(reject_null))/1000

# Output the Type I error rate

type_1_error_rate
```

```
## [1] 0.957
```

scenario b

```r
set.seed(123)  # For reproducibility

# Load data (assuming temperature values are stored in 'temperature' variable)
temperature <- data_tidy_air_quality$temperature

n <- length(temperature)  # Number of observations in temperature data


reject_null <- numeric(n_simulations)  # To store whether the null hypothesis is rejected

for (i in 1:n_simulations) {
  # Simulate heteroscedastic errors with mean variance 100 and variance of the variances = 50
  # Generate different variances for each observation, ensuring mean is 100 and variance is 50
  variances <- rnorm(n, mean = 100, sd = sqrt(50))  # Variance of each error term
  e <- rnorm(n, mean = 0, sd = sqrt(abs(variances)))  # Simulated errors with heteroscedasticity

  # Generate Y values under null hypothesis (beta_1 = 0)
  Y <- 30 + e

  # Fit the model Y ~ temperature
  model <- lm(Y ~ temperature)
```

```r
  # Perform hypothesis test for beta_1 (test if beta_1 = 0)
  p_value <- summary(model)$coefficients[2, 4]  # Extract p-value for temperature coefficient

  # Record whether the null hypothesis is rejected (p-value < 0.05)
  reject_null[i] <- as.numeric(p_value < 0.05)
}

# Calculate the Type I error rate (proportion of rejected null hypotheses)
type_1_error_rate <- (1000-sum(reject_null))/1000

# Print the Type I error rate
type_1_error_rate
```

```
## [1] 0.953
```

scenario C

```r
set.seed(123)  # For reproducibility

# Load data (assuming temperature values are stored in 'temperature' variable)
temperature <- data_tidy_air_quality$temperature

n <- length(temperature)  # Number of observations in temperature data
n_sim <- 1000  # Number of simulations

# Function to generate autocorrelated errors
generate_ar1_errors <- function(n, rho, var_epsilon) {
  sigma_u_squared <- var_epsilon * (1 - rho^2)
  u <- rnorm(n, mean = 0, sd = sqrt(sigma_u_squared))
  epsilon <- numeric(n)
  epsilon[1] <- u[1]
  for (i in 2:n) {
    epsilon[i] <- rho * epsilon[i - 1] + u[i]
  }
  epsilon
}

rho <- 0.3  # Autocorrelation coefficient
var_epsilon <- 100  # Desired variance of errors

reject_null <- numeric(n_sim)  # To store whether the null hypothesis is rejected

for (i in 1:n_sim) {
  # Generate autocorrelated errors with Corr(epsilon_i, epsilon_(i+1)) = 0.3
  e <- generate_ar1_errors(n, rho, var_epsilon)

  # Generate Y values under null hypothesis (beta_1 = 0)
  Y <- 30 + e

  # Fit the model Y ~ temperature
  model <- lm(Y ~ temperature)

  # Perform hypothesis test for beta_1 (test if beta_1 = 0)
```

```r
  p_value <- summary(model)$coefficients[2, 4]  # Extract p-value for temperature coefficient

  # Record whether the null hypothesis is rejected (p-value < 0.05)
  reject_null[i] <- as.numeric(p_value < 0.05)
}

# Calculate the Type I error rate (proportion of rejected null hypotheses)
type_1_error_rate <- (1000-sum(reject_null))/1000

# Print the Type I error rate
type_1_error_rate
```

```
## [1] 0.947
```