Final Project

```
# Load necessary libraries
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
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(lubridate)
library(ggplot2)
library(dplyr)
library(corrplot)
```

```
corrplot 0.92 loaded
```

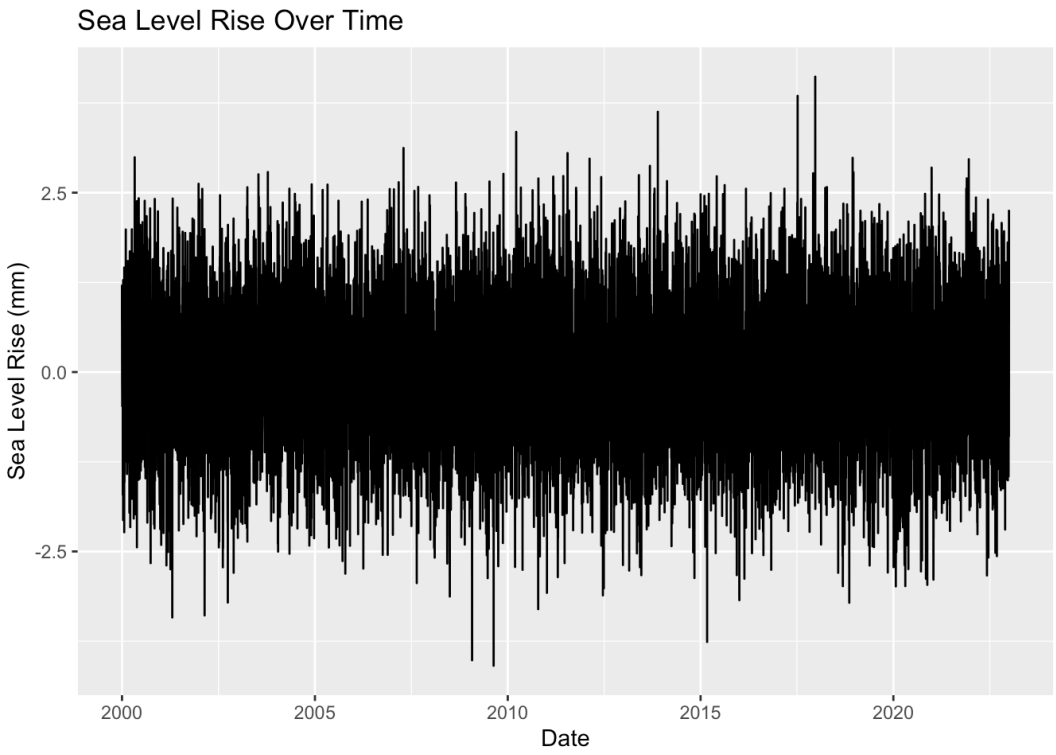
```
# Load the dataset
data <- read.csv("climate_change_data.csv")

# Convert Date to Date format
data$Date <- as.Date(data$Date, format="%Y-%m-%d")

# Handle missing values
data <- na.omit(data)

# Normalize data
data_scaled <- data %>%
  mutate(across(c(Temperature, CO2.Emissions, Sea.Level
head(data)
```

```
        Date         Location       Country Temperature
CO2.Emissions
1 2000-01-01  New Williamtown        Latvia   10.688986
403.1189
2 2000-01-01     North Rachel  South Africa   13.814430
```

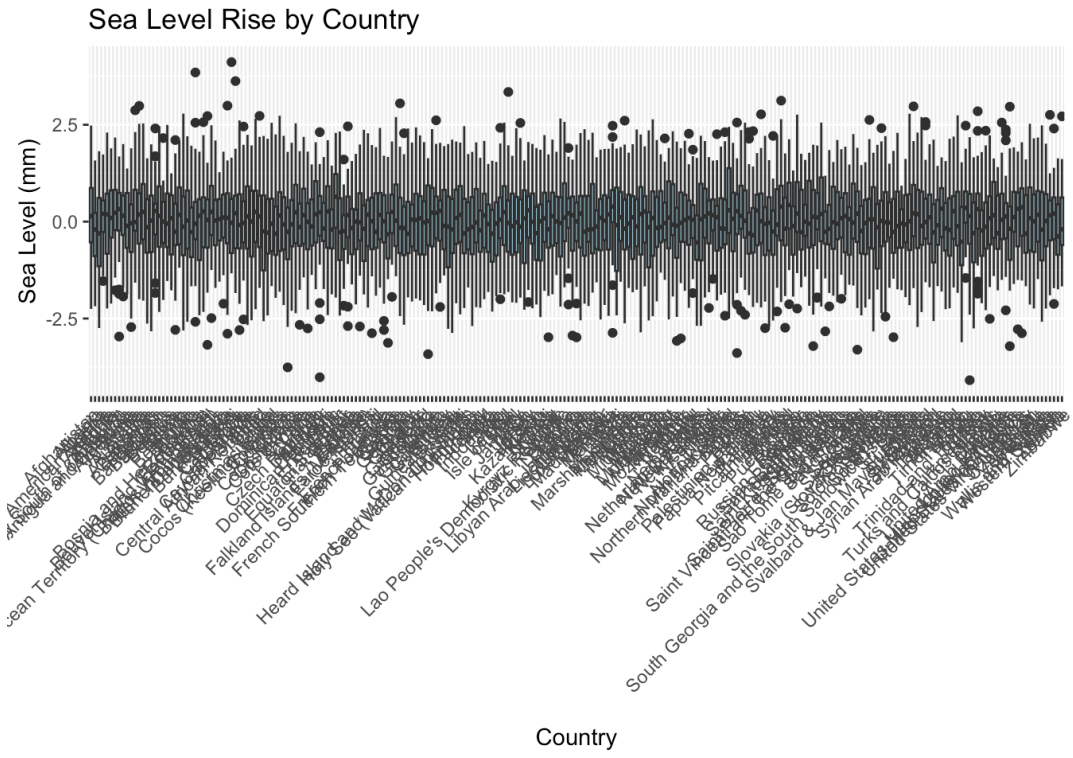
```
396.6635
3 2000-01-02 West Williamland French Guiana   27.323718
451.5532
4 2000-01-03      South David       Vietnam   12.309581
422.4050
5 2000-01-04   New Scottburgh       Moldova   13.210885
410.4730
6 2000-01-05     South Nathan  Saint Helena    6.229326
392.4733
  Sea.Level.Rise Precipitation Humidity Wind.Speed
1      0.7175060     13.835237 23.63126  18.492026
2      1.2057146     40.974084 43.98295  34.249300
3     -0.1607830     42.697931 96.65260  34.124261
4     -0.4759315      5.193341 47.46794   8.554563
5      1.1357566     78.695280 61.78967   8.001164
6      1.1222097     76.368331 48.97389  30.398908
```

```r
# Generate future data
future_data <- data.frame(
  Date = seq(from = max(data$Date), by = "month", lengt
  Temperature = c(20, 21, rep(mean(data$Temperature, na
  CO2.Emissions = c(400, 405, rep(mean(data$CO2.Emissio
  Precipitation = c(10, 12, rep(mean(data$Precipitation
  Humidity = c(60, 65, rep(mean(data$Humidity, na.rm =
  Wind.Speed = c(5, 6, rep(mean(data$Wind.Speed, na.rm
)

# Convert Date to Date format
future_data$Date <- as.Date(future_data$Date)
```

```r
# Plot sea level rise over time
ggplot(data, aes(x = Date, y = Sea.Level.Rise)) +
  geom_line() +
  labs(title = "Sea Level Rise Over Time", x = "Date",
```




```
ggplot(data, aes(x = Country, y = Sea.Level.Rise)) +
  geom_boxplot(fill = "lightblue") +
  labs(title = "Sea Level Rise by Country", x = "Countr
  theme(axis.text.x = element_text(angle = 45, hjust =
```



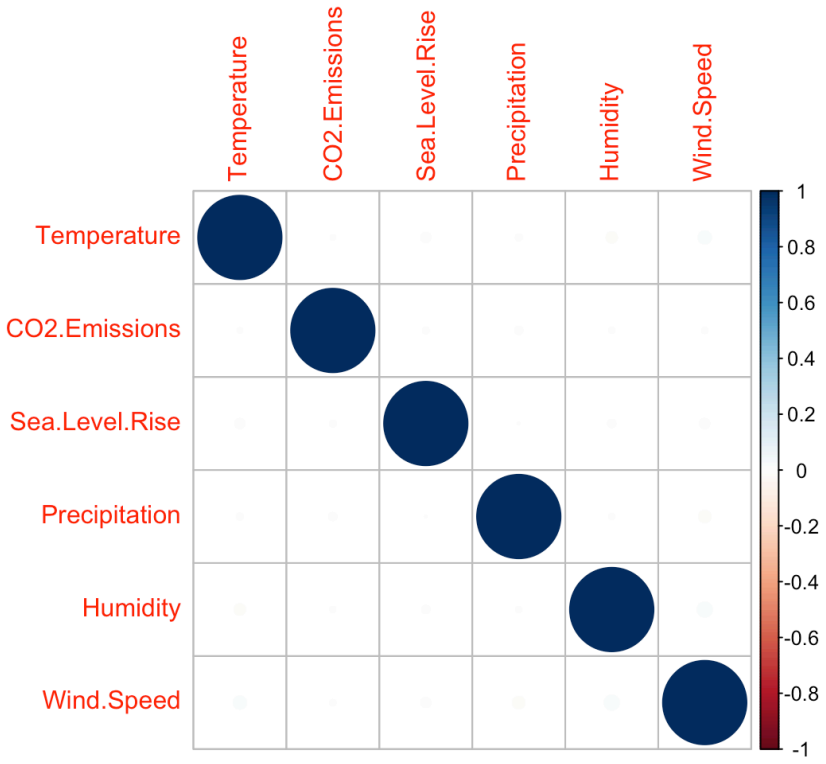
```
#To check whether co2 emissions can predict future temp
library(lmtest)
```

```
Loading required package: zoo


Attaching package: 'zoo'

The following objects are masked from 'package:base':

    as.Date, as.Date.numeric
```

```
# Convert data to time series
temperature_ts <- ts(data$Temperature, start = c(year(m
co2_ts <- ts(data$CO2.Emissions, start = c(year(min(dat

# Granger causality test
grangertest(temperature_ts ~ co2_ts, order = 2)
```

```
Granger causality test

Model 1: temperature_ts ~ Lags(temperature_ts, 1:2) +
Lags(co2_ts, 1:2)
Model 2: temperature_ts ~ Lags(temperature_ts, 1:2)
  Res.Df Df      F Pr(>F)
1   9993
2   9995 -2 0.0338 0.9668
```

```
# Correlation matrix
corr_matrix <- cor(data %>% select(Temperature, CO2.Emi
corrplot(corr_matrix, method = "circle")
```




```
# Loading necessary libraries
library(forecast)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```
library(prophet)
```

```
Loading required package: Rcpp

Loading required package: rlang


Attaching package: 'rlang'

The following objects are masked from 'package:purrr':

    %@%, flatten, flatten_chr, flatten_dbl, flatten_int,
flatten_lgl,
    flatten_raw, invoke, splice
```

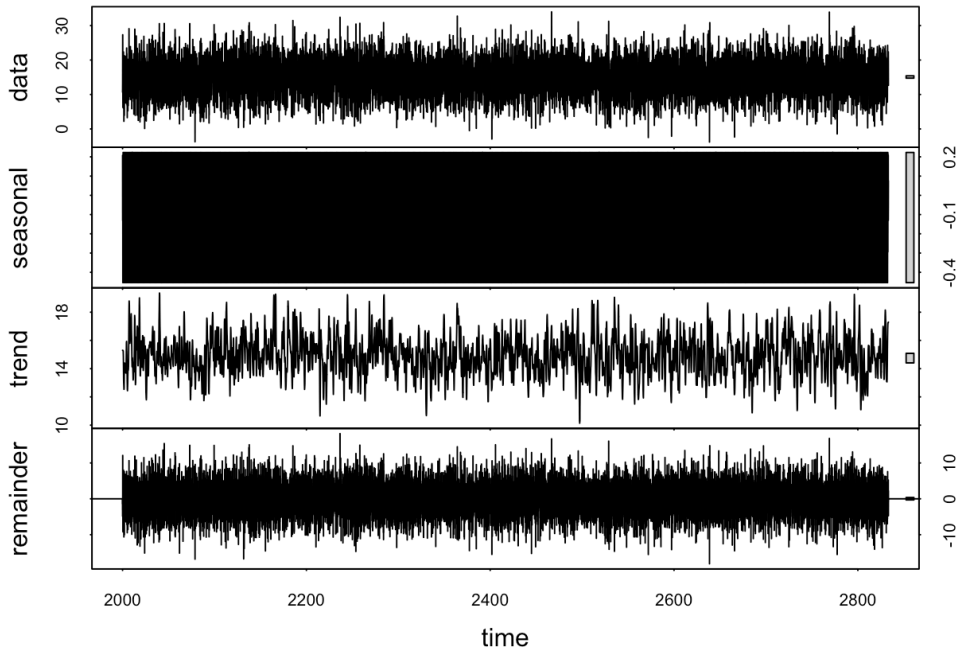
```
library(cluster)
library(factoextra)
```

```
Welcome! Want to learn more? See two factoextra-related books
at https://goo.gl/ve3WBa
```

```
library(reshape2)
```

```
Attaching package: 'reshape2'

The following object is masked from 'package:tidyr':

    smiths
```

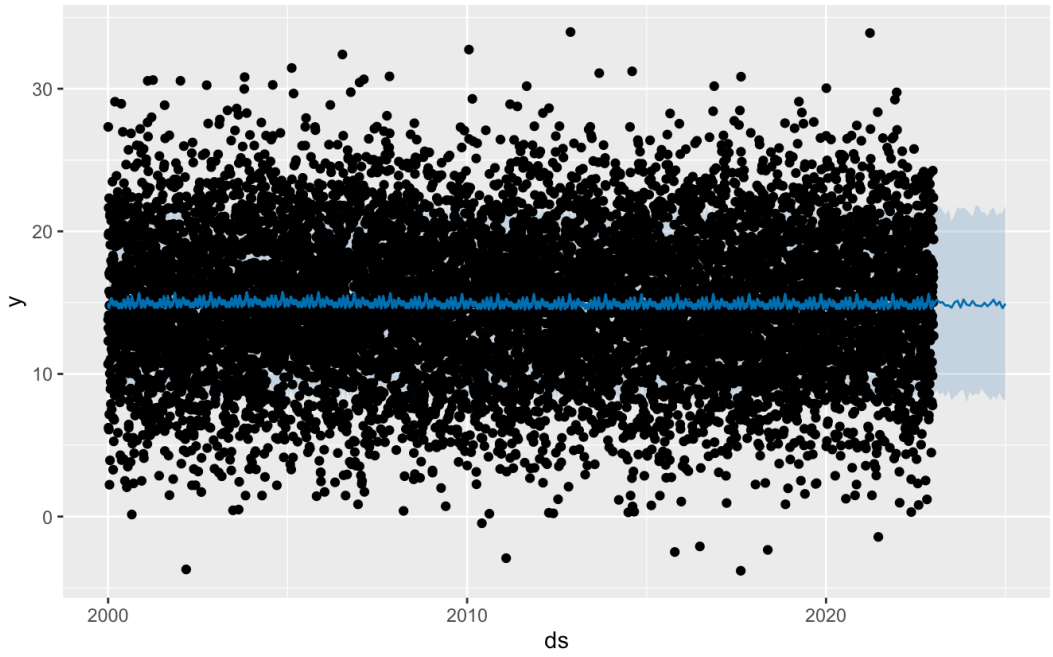
```
# Seasonal Decomposition of Time Series
temperature_ts <- ts(data$Temperature, start = c(year(m
decomp <- stl(temperature_ts, s.window = "periodic")
plot(decomp)
```



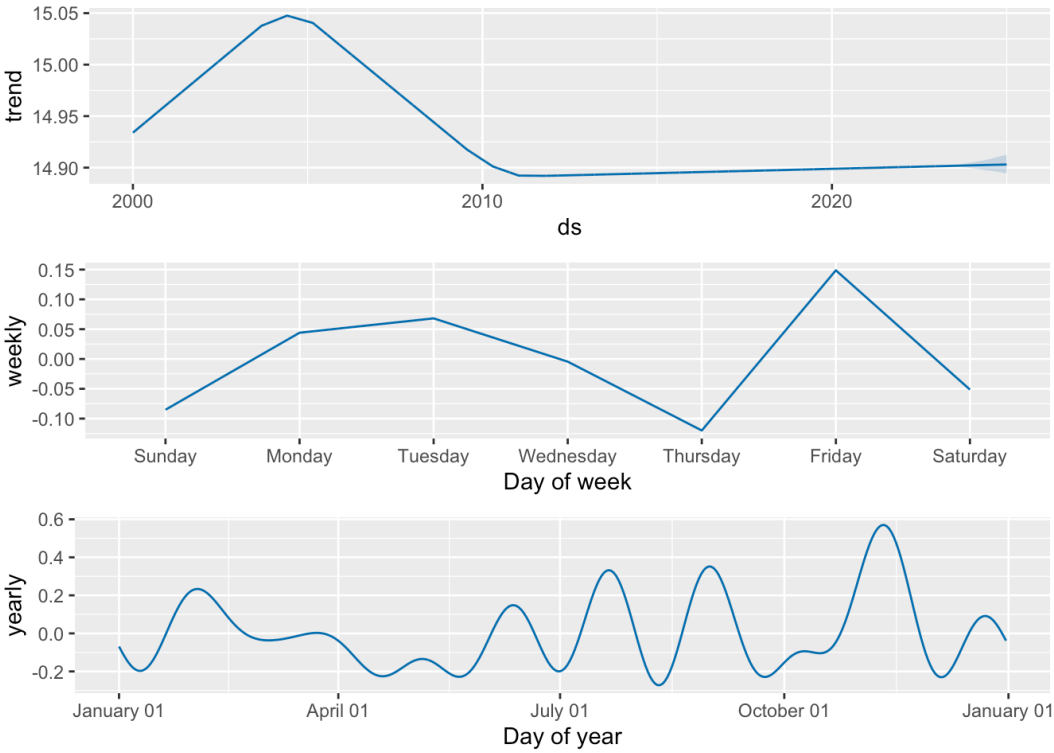

```
# Time Series Forecasting using Prophet
data_prophet <- data %>%
  select(Date, Temperature) %>%
  rename(ds = Date, y = Temperature)
m <- prophet(data_prophet)
```

```
Disabling daily seasonality. Run prophet with
daily.seasonality=TRUE to override this.
```

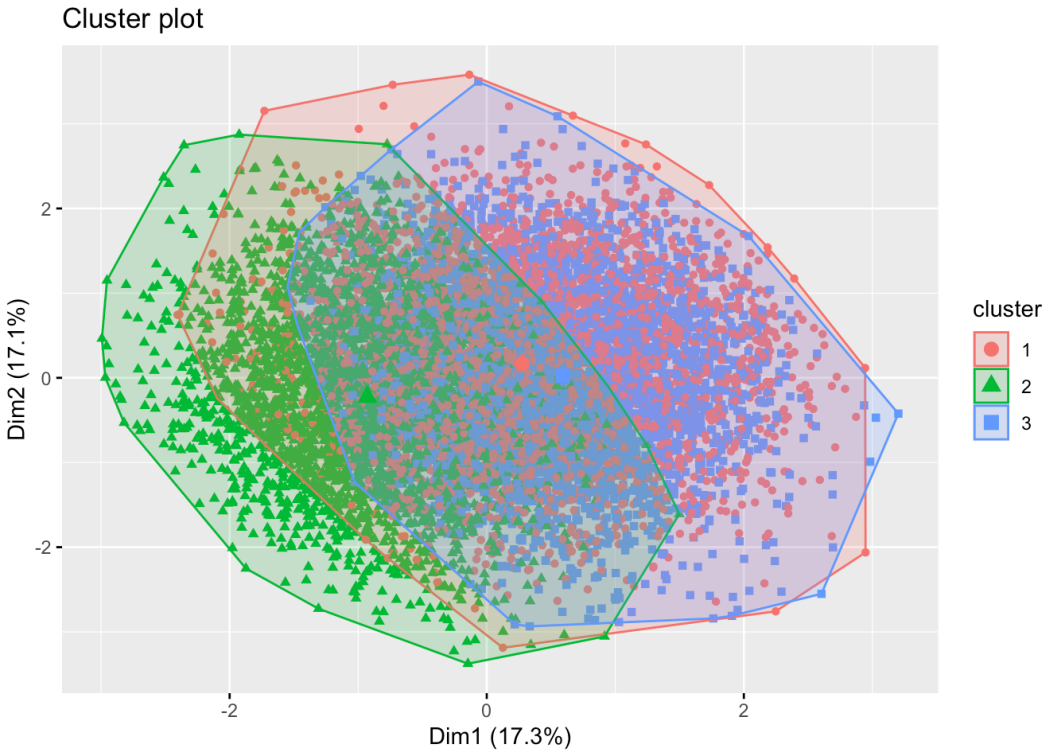
```
future <- make_future_dataframe(m, periods = 24, freq =
forecast <- predict(m, future)
plot(m, forecast)
```



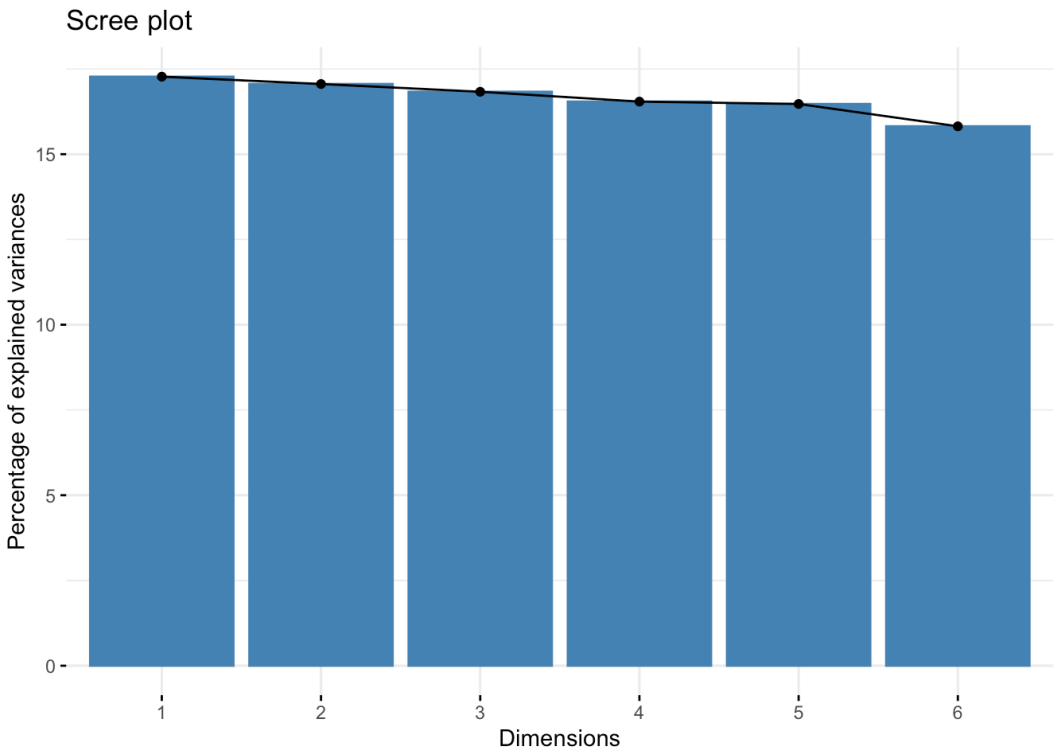

```
prophet_plot_components(m, forecast)
```



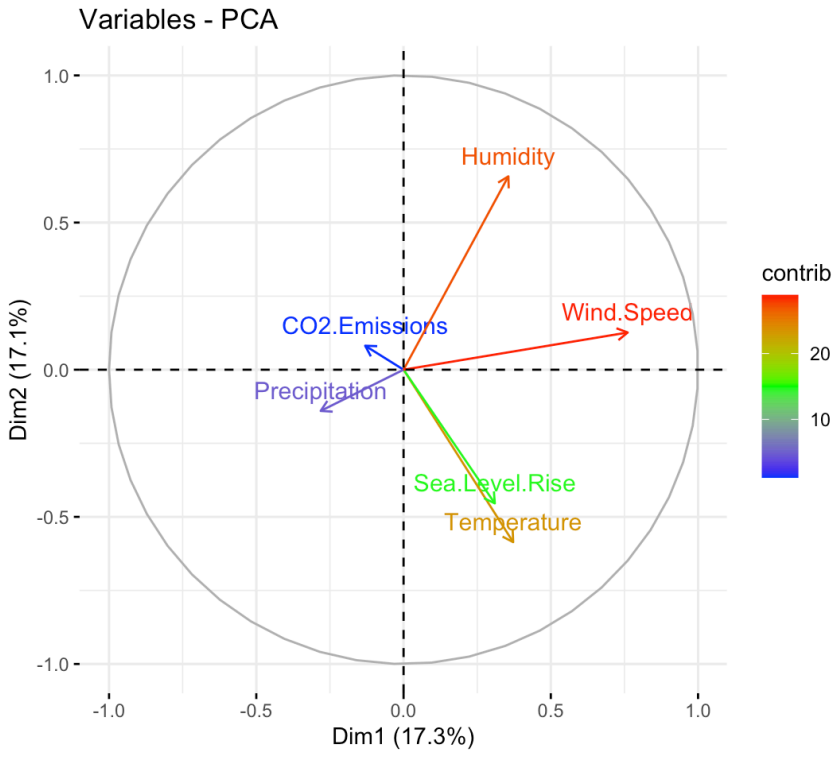
```
# Clustering Analysis
data_scaled <- data %>%
  select(Temperature, CO2.Emissions, Sea.Level.Rise, Pr
  scale()
set.seed(123)
clusters <- kmeans(data_scaled, centers = 3)
fviz_cluster(clusters, data = data_scaled, geom = "poin
```




```
# Principal Component Analysis (PCA)
pca <- prcomp(data_scaled, center = TRUE, scale. = TRUE
fviz_eig(pca)
```




```
fviz_pca_var(pca, col.var = "contrib", gradient.cols =
```

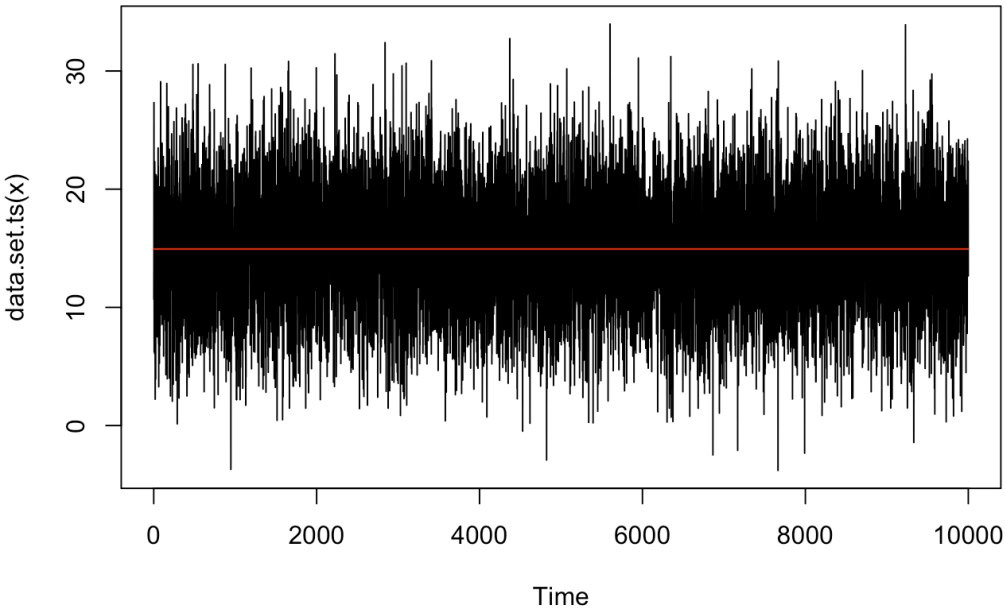



```
# Loading necessary libraries
library(zoo)
library(anomalize)
```

```
if (!requireNamespace("changepoint", quietly = TRUE)) {
  install.packages("changepoint")
}

library(changepoint)
```

```
Successfully loaded changepoint package version 2.2.4
 See NEWS for details of changes.
```

```
# Change Point Detection using PELT method
cpt_temp <- cpt.meanvar(data$Temperature, method = "PEL
plot(cpt_temp)
```




```
# Extracting change points
change_points <- cpts(cpt_temp)
change_points
```

```
integer(0)
```

```
# Anomaly Detection using IQR method
detect_anomalies <- function(x) {
  q25 <- quantile(x, 0.25)
  q75 <- quantile(x, 0.75)
  iqr <- q75 - q25
  lower_bound <- q25 - 1.5 * iqr
```

```
  upper_bound <- q75 + 1.5 * iqr
  which(x < lower_bound | x > upper_bound)
}
anomalies <- lapply(data %>% select(Temperature, CO2.Em
anomalies
```

```
$Temperature
 [1]   85  161  289  481  545  686  878  947 1196 1448 1514
1558 1582 1650 1655
[16] 1666 1866 1996 2225 2248 2526 2693 2840 2878 2941 3029
3048 3099 3409 3580
[31] 4090 4371 4413 4529 4619 4823 4870 4958 5071 5340 5341
5394 5451 5602 5950
[46] 6187 6302 6330 6348 6353 6374 6581 6684 6865 6944 7169
7330 7342 7492 7653
[61] 7664 7668 7991 8205 8369 8701 8938 9048 9229 9331 9532
9553 9589 9729 9819
[76] 9920

$CO2.Emissions
 [1]  209  229  231  362  417  443  620  635  650  856  904
937  957 1038 1388
[16] 1652 1690 1720 1750 1821 1889 2050 2199 2249 2257 2704
2711 2956 3105 3334
[31] 3402 3562 3652 3663 3915 3992 4005 4192 4294 4323 4561
4625 4765 4803 5038
[46] 5104 5442 5571 5589 5887 6026 6138 6209 6316 6351 6443
6610 6826 7017 7257
[61] 7533 7755 7758 7825 7924 7962 8037 8057 8133 8193 8198
8445 8500 8934 9047
[76] 9215 9323 9425 9637 9667 9743 9889

$Sea.Level.Rise
 [1]  145  548  568  933 1138 1194 1260 1540 1645 2519 2722
3175 3324 3696 3948
[16] 4124 4191 4243 4301 4432 4445 4517 4693 4792 4863 4914
4976 5024 5272 5403
[31] 5423 5433 5719 5828 5840 5856 5952 6042 6454 6597 6706
6932 6960 7017 7320
[46] 7617 7792 7816 8075 8127 8199 8238 8247 8634 8700 8725
8832 8866 9008 9064
[61] 9079 9128 9148 9525 9547 9752

$Precipitation
integer(0)

$Humidity
```

```
integer(0)

$Wind.Speed
integer(0)
```

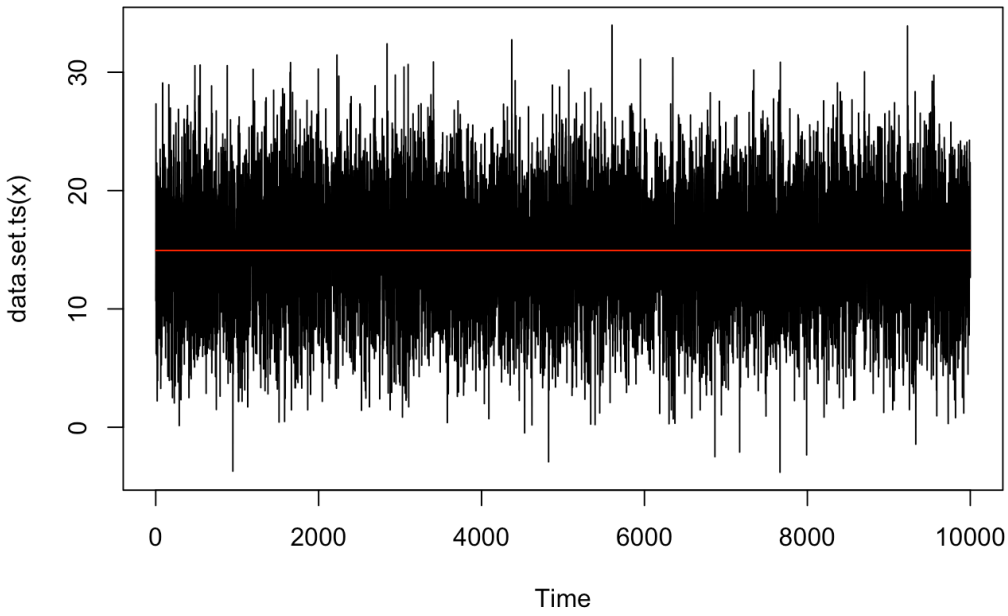
```
# Loading necessary libraries
library(zoo)
library(anomalize)
library(changepoint)

# Anomaly Detection using IQR method
detect_anomalies <- function(x) {
  q25 <- quantile(x, 0.25)
  q75 <- quantile(x, 0.75)
  iqr <- q75 - q25
  lower_bound <- q25 - 1.5 * iqr
  upper_bound <- q75 + 1.5 * iqr
  which(x < lower_bound | x > upper_bound)
}
anomalies <- lapply(data %>% select(Temperature, CO2.Em
anomalies
```

```
$Temperature
 [1]   85  161  289  481  545  686  878  947 1196 1448 1514
1558 1582 1650 1655
[16] 1666 1866 1996 2225 2248 2526 2693 2840 2878 2941 3029
3048 3099 3409 3580
[31] 4090 4371 4413 4529 4619 4823 4870 4958 5071 5340 5341
5394 5451 5602 5950
[46] 6187 6302 6330 6348 6353 6374 6581 6684 6865 6944 7169
7330 7342 7492 7653
[61] 7664 7668 7991 8205 8369 8701 8938 9048 9229 9331 9532
9553 9589 9729 9819
[76] 9920

$CO2.Emissions
 [1]  209  229  231  362  417  443  620  635  650  856  904
937  957 1038 1388
[16] 1652 1690 1720 1750 1821 1889 2050 2199 2249 2257 2704
2711 2956 3105 3334
[31] 3402 3562 3652 3663 3915 3992 4005 4192 4294 4323 4561
4625 4765 4803 5038
[46] 5104 5442 5571 5589 5887 6026 6138 6209 6316 6351 6443
6610 6826 7017 7257
[61] 7533 7755 7758 7825 7924 7962 8037 8057 8133 8193 8198
8445 8500 8934 9047
[76] 9215 9323 9425 9637 9667 9743 9889
```

```
$Sea.Level.Rise
 [1]  145  548  568  933 1138 1194 1260 1540 1645 2519 2722
3175 3324 3696 3948
[16] 4124 4191 4243 4301 4432 4445 4517 4693 4792 4863 4914
4976 5024 5272 5403
[31] 5423 5433 5719 5828 5840 5856 5952 6042 6454 6597 6706
6932 6960 7017 7320
[46] 7617 7792 7816 8075 8127 8199 8238 8247 8634 8700 8725
8832 8866 9008 9064
[61] 9079 9128 9148 9525 9547 9752

$Precipitation
integer(0)

$Humidity
integer(0)

$Wind.Speed
integer(0)
```

```r
# Rolling Window Analysis
# Compute rolling statistics
rolling_mean <- rollapply(data$Temperature, width = 12,
rolling_sd <- rollapply(data$Temperature, width = 12, F

# Add rolling statistics to the original data
data <- data %>%
  mutate(rolling_mean_temp = rolling_mean,
         rolling_sd_temp = rolling_sd)

# Cross-Correlation Analysis
ccf_temp_co2 <- ccf(data$Temperature, data$CO2.Emission
ccf_temp_precip <- ccf(data$Temperature, data$Precipita

# Time Series Decomposition and Recomposition
temperature_ts <- ts(data$Temperature, start = c(year(m
decompose_ts <- decompose(temperature_ts, type = "multi
trend <- decompose_ts$trend
seasonal <- decompose_ts$seasonal
random <- decompose_ts$random

# Using decomposed components for predictive modeling
trend_model <- lm(trend ~ time(trend))
seasonal_model <- lm(seasonal ~ time(seasonal))
random_model <- lm(random ~ time(random))
```

```
trend_pred <- predict(trend_model, newdata = data.frame
seasonal_pred <- predict(seasonal_model, newdata = data
random_pred <- predict(random_model, newdata = data.fra

recomposed_ts <- trend_pred * seasonal_pred * random_pr

# Change Point Detection using the changepoint package
cpt_temp <- cpt.meanvar(data$Temperature, method = "PEL
plot(cpt_temp)
```

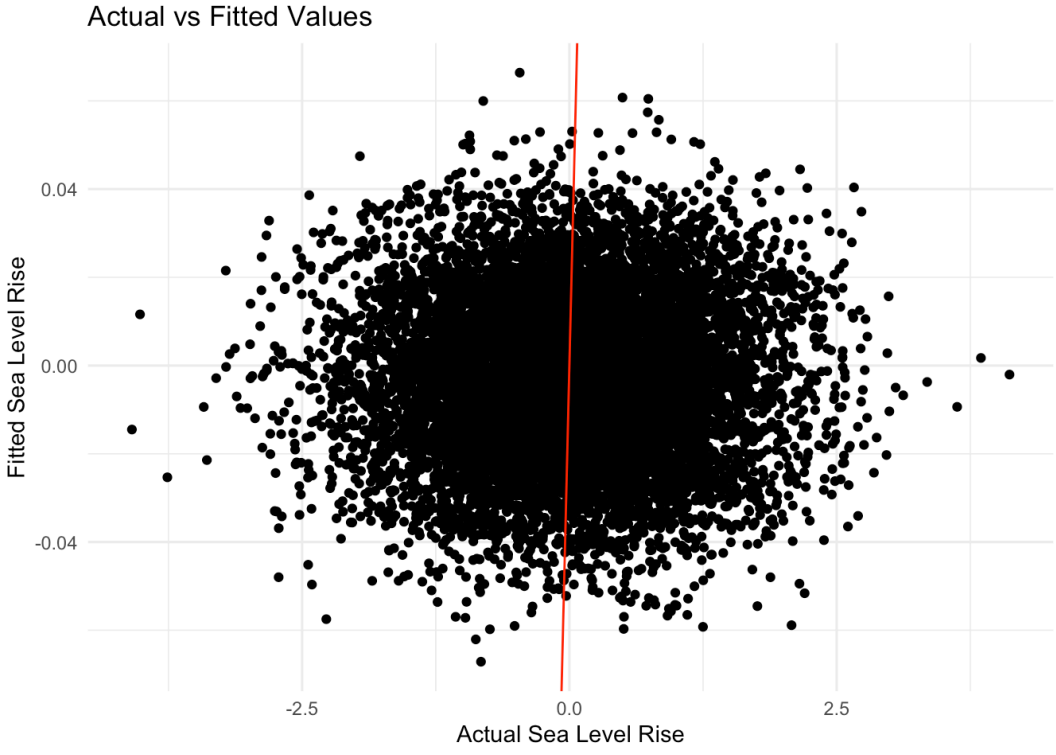



```
# Extract change points
change_points <- cpts(cpt_temp)
```

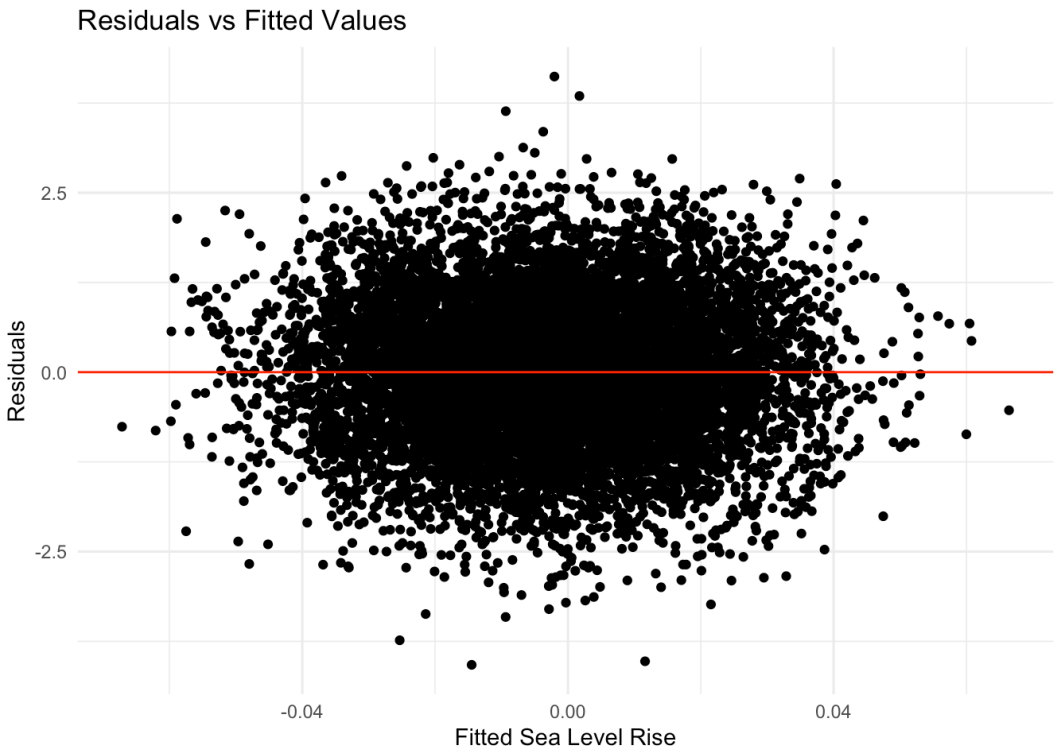
```
# Load necessary libraries
library(tidyverse)

# Prepare the data
data_lr <- data %>%
    select(Temperature, CO2.Emissions, Precipitation, H

# Train the linear regression model
lr_model <- lm(Sea.Level.Rise ~ ., data = data_lr)

# Display the model summary
summary(lr_model)
```

```
Call:
lm(formula = Sea.Level.Rise ~ ., data = data_lr)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0777 -0.6690  0.0057  0.6794  4.1186

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)   -5.548e-03  9.086e-02  -0.061    0.951
Temperature    2.221e-03  1.972e-03   1.127    0.260
CO2.Emissions -9.379e-05  1.995e-04  -0.470    0.638
Precipitation -3.072e-06  3.436e-04  -0.009    0.993
Humidity      -2.668e-04  3.429e-04  -0.778    0.437
Wind.Speed     8.049e-04  6.859e-04   1.173    0.241

Residual standard error: 0.9914 on 9994 degrees of freedom
Multiple R-squared:  0.0003515, Adjusted R-squared:  -0.0001486
F-statistic: 0.7029 on 5 and 9994 DF,  p-value: 0.6212
```

```r
# Plotting the fitted values vs actual values
ggplot(data_lr, aes(x = Sea.Level.Rise, y = predict(lr_
  geom_point() +
  geom_abline(slope = 1, intercept = 0, color = "red")
  labs(title = "Actual vs Fitted Values", x = "Actual S
  theme_minimal()
```



```
# Residuals plot to check for homoscedasticity
ggplot(data_lr, aes(x = predict(lr_model), y = residual
  geom_point() +
  geom_hline(yintercept = 0, color = "red") +
  labs(title = "Residuals vs Fitted Values", x = "Fitte
  theme_minimal()
```



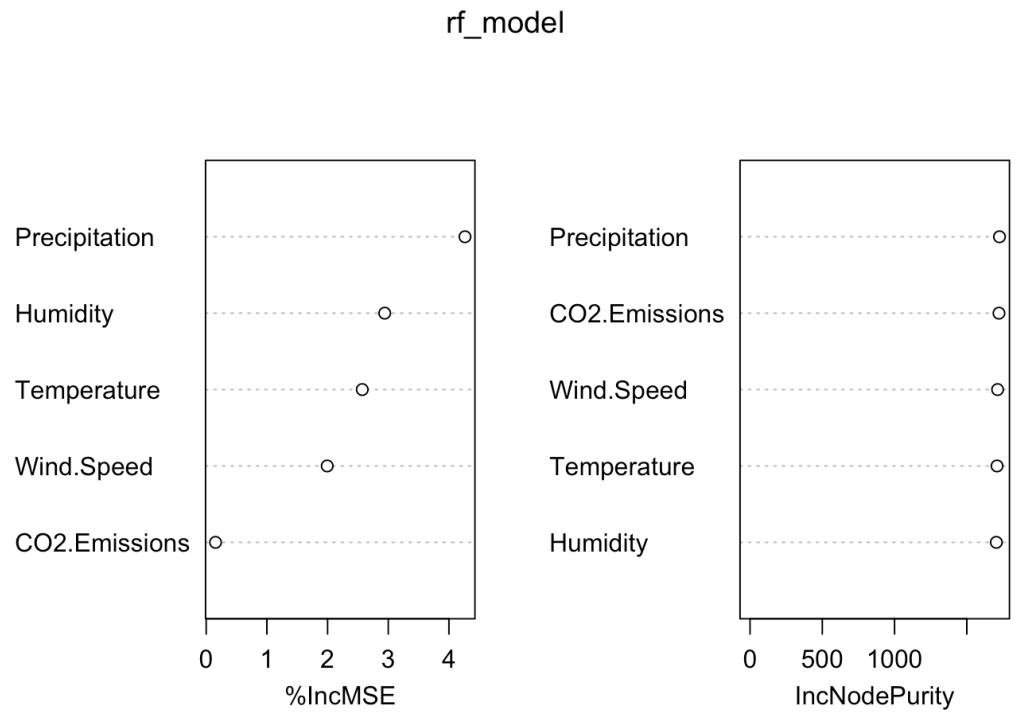

```
# Load necessary libraries
library(randomForest)
```

```
randomForest 4.7-1.1

Type rfNews() to see new features/changes/bug fixes.


Attaching package: 'randomForest'

The following object is masked from 'package:dplyr':

    combine

The following object is masked from 'package:ggplot2':

    margin
```

```
# Prepare the data
data_rf <- data %>% select(Temperature, CO2.Emissions,

# Train a Random Forest model
rf_model <- randomForest(Sea.Level.Rise ~ ., data = dat

# Display feature importance
importance(rf_model)
```

```
               %IncMSE IncNodePurity
Temperature   2.5714803      1709.264
CO2.Emissions 0.1536604      1722.724
Precipitation 4.2639154      1726.195
Humidity      2.9414851      1704.476
Wind.Speed    1.9936355      1713.355
```

```
varImpPlot(rf_model)
```




```
# Load necessary libraries
library(tidyverse)
library(Metrics)
```

```
Attaching package: 'Metrics'
```

```
The following object is masked from 'package:rlang':

    ll

The following object is masked from 'package:forecast':

    accuracy
```

```
# Prepare the data
data_lr <- data %>%
    select(Temperature, CO2.Emissions, Precipitation, H

# Train the linear regression model
lr_model <- lm(Sea.Level.Rise ~ ., data = data_lr)

# Make predictions
predictions <- predict(lr_model, newdata = data_lr)

# Print lengths to check for consistency
print(length(predictions))
```

```
[1] 10000
```

```
print(length(data$Sea.Level.Rise))
```

```
[1] 10000
```

```
# Calculate RMSE
rmse_value <- rmse(data$Sea.Level.Rise, predictions)

# Calculate MAE
mae_value <- mae(data$Sea.Level.Rise, predictions)

# Calculate R-squared manually
ss_total <- sum((data$Sea.Level.Rise - mean(data$Sea.Le
ss_residual <- sum((data$Sea.Level.Rise - predictions)^
r_squared <- 1 - (ss_residual / ss_total)

# Print the results
cat("RMSE:", rmse_value, "\n")
```

```
RMSE: 0.9911248
```

```
cat("MAE:", mae_value, "\n")
```

```
MAE: 0.7922148
```

```
        cat("R-squared:", r_squared, "\n")
```

```
R-squared: 0.0003515424
```

```
        # Linear regression
        model <- lm(Sea.Level.Rise ~ Temperature + CO2.Emission
        summary(model)
```

```
Call:
lm(formula = Sea.Level.Rise ~ Temperature + CO2.Emissions +
Precipitation +
    Humidity + Wind.Speed, data = data)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0777 -0.6690  0.0057  0.6794  4.1186

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)   -5.548e-03  9.086e-02  -0.061    0.951
Temperature    2.221e-03  1.972e-03   1.127    0.260
CO2.Emissions -9.379e-05  1.995e-04  -0.470    0.638
Precipitation -3.072e-06  3.436e-04  -0.009    0.993
Humidity      -2.668e-04  3.429e-04  -0.778    0.437
Wind.Speed     8.049e-04  6.859e-04   1.173    0.241

Residual standard error: 0.9914 on 9994 degrees of freedom
Multiple R-squared:  0.0003515, Adjusted R-squared:  -0.0001486
F-statistic: 0.7029 on 5 and 9994 DF,  p-value: 0.6212
```

```
        # Predictions
        future_pred_data <- data.frame(
          Temperature = c(20, 21),
          CO2.Emissions = c(400, 405),
          Precipitation = c(10, 12),
          Humidity = c(60, 65),
          Wind.Speed = c(5, 6)
        )
        predictions <- predict(model, newdata = future_data)
```

```
        # Load necessary library
        library(rpart)
```

```
# Fit a Decision Tree model
tree_model <- rpart(Sea.Level.Rise ~ ., data = data)

# Predictions on the test data
predictions_tree <- predict(tree_model, data)

# Calculate RMSE, MAE, and R-squared for the decision t
rmse_tree <- rmse(data$Sea.Level.Rise, predictions_tree
mae_tree <- mae(data$Sea.Level.Rise, predictions_tree)
r_squared_tree <- 1 - sum((data$Sea.Level.Rise - predic

# Print the results
cat("Decision Tree Regression Model\n")
```

```
Decision Tree Regression Model
```

```
cat("RMSE:", rmse_tree, "\n")
```

```
RMSE: 0.4957836
```

```
cat("MAE:", mae_tree, "\n")
```

```
MAE: 0.3186194
```

```
cat("R-squared:", r_squared_tree, "\n")
```

```
R-squared: 0.7498648
```

```
# Load necessary libraries
library(shiny)
library(dplyr)
library(tidyverse)
library(lubridate)
library(leaflet)

data <- tibble(
  Date = seq(as.Date("2000-01-01"), as.Date("2023-12-31
  Location = sample(c("Urban", "Rural"), length(seq(as.
  Country = sample(c("USA", "Canada", "Brazil", "Austra
  Temperature = runif(length(seq(as.Date("2000-01-01"),
  CO2_Emissions = runif(length(seq(as.Date("2000-01-01"
  Sea_Level_Rise = runif(length(seq(as.Date("2000-01-01
  Humidity = runif(length(seq(as.Date("2000-01-01"), as
)
```

```
# Define UI for the application
ui <- fluidPage(
  titlePanel("Environmental Dashboard"),
  sidebarLayout(
    sidebarPanel(
      dateRangeInput("dateRange", "Select Date Range",
      selectInput("country", "Select Country", choices
      selectInput("location", "Select Location", choice
      selectInput("measurement", "Select Measurement",
      downloadButton("downloadData", "Download Data")
    ),
    mainPanel(
      plotOutput("mainPlot"),
      tableOutput("summaryTable"),
      leafletOutput("map")
    )
  )
)

# Define server logic
server <- function(input, output) {

  filtered_data <- reactive({
    data %>%
      filter(Date >= input$dateRange[1], Date <= input$
             Country == input$country, Location == inpu
  })

  output$mainPlot <- renderPlot({
    ggplot(filtered_data(), aes(x = Date, y = !!sym(inp
      geom_line() +
      labs(title = paste(input$measurement, "Over Time"
      theme_minimal()
  })

  output$summaryTable <- renderTable({
    filtered_data() %>%
      summarise(
        Mean = mean(!!sym(input$measurement), na.rm = T
        Median = median(!!sym(input$measurement), na.rm
        SD = sd(!!sym(input$measurement), na.rm = TRUE)
      )
  })

  output$map <- renderLeaflet({
    leaflet() %>%
      addTiles() %>%
      addMarkers(lng = runif(5, -180, 180), lat = runif
```

```
  })

  output$downloadData <- downloadHandler(
    filename = function() {
      paste("filtered_data_", Sys.Date(), ".csv", sep =
    },
    content = function(file) {
      write.csv(filtered_data(), file, row.names = FALS
    }
  )
}

# Run the application
shinyApp(ui = ui, server = server)
```

Shiny applications not supported in static R Markdown documents