**Week 3 - Task 1: Automating Repetitive Processes**

**User-Generated Prompt for AI (Example)**

*"I have a dataset containing air quality measurements with columns for Date, PM2.5, PM10, CO, NO2, SO2, and O3. I need a Python script that automates data transformation tasks, including:*\*

* **Filtering out records where PM2.5 is above 50.**
* **Creating a new column indicating whether air quality is 'Good', 'Moderate', or 'Unhealthy' based on PM2.5 levels.**
* **Applying Min-Max normalization to the PM10 column.**
* **Saving the transformed dataset as a new CSV file.** Can you generate an efficient Python script for this task?"\*

**AI-Generated Python Code for Task 1**

import pandas as pd

from sklearn.preprocessing import MinMaxScaler

def transform\_air\_quality\_data(file\_path):

"""

Automates repetitive air quality data transformations.

"""

try:

# Load dataset

df = pd.read\_csv(file\_path)

# Filter out records where PM2.5 is above 50

df = df[df['PM2.5'] <= 50]

# Categorizing PM2.5 levels

def categorize\_pm25(value):

if value < 12:

return "Good"

elif value < 35:

return "Moderate"

else:

return "Unhealthy"

df['Air\_Quality\_Category'] = df['PM2.5'].apply(categorize\_pm25)

# Normalizing PM10 levels using Min-Max Scaling

scaler = MinMaxScaler()

df[['PM10']] = scaler.fit\_transform(df[['PM10']])

# Save transformed dataset

transformed\_file\_path = "transformed\_" + file\_path

df.to\_csv(transformed\_file\_path, index=False)

print(f"Transformed dataset saved as {transformed\_file\_path}")

return df

except Exception as e:

print("Error processing the dataset:", e)

return None

# Example usage

file\_path = "Air\_Quality\_Dataset\_example.csv"

processed\_data = transform\_air\_quality\_data(file\_path)

**Week 3 - Task 1: Automating Repetitive Processes (R Version)**

**User-Generated Prompt for AI (Example)**

*"I have a dataset containing air quality measurements with columns for Date, PM2.5, PM10, CO, NO2, SO2, and O3. I need an R script that automates data transformation tasks, including:*\*

* **Filtering out records where PM2.5 is above 50.**
* **Creating a new column indicating whether air quality is 'Good', 'Moderate', or 'Unhealthy' based on PM2.5 levels.**
* **Applying Min-Max normalization to the PM10 column.**
* **Saving the transformed dataset as a new CSV file.** Can you generate an efficient R script for this task?"\*

**AI-Generated R Code for Task 1**

library(dplyr)

transform\_air\_quality\_data <- function(file\_path) {

"""

Automates repetitive air quality data transformations in R.

"""

tryCatch({

# Load dataset

df <- read.csv(file\_path, stringsAsFactors = FALSE)

# Filter out records where PM2.5 is above 50

df <- df %>% filter(PM2.5 <= 50)

# Categorizing PM2.5 levels

df$Air\_Quality\_Category <- case\_when(

df$PM2.5 < 12 ~ "Good",

df$PM2.5 >= 12 & df$PM2.5 < 35 ~ "Moderate",

TRUE ~ "Unhealthy"

)

# Normalizing PM10 levels using Min-Max Scaling

df$PM10 <- (df$PM10 - min(df$PM10, na.rm = TRUE)) / (max(df$PM10, na.rm = TRUE) - min(df$PM10, na.rm = TRUE))

# Save transformed dataset

transformed\_file\_path <- paste0("transformed\_", file\_path)

write.csv(df, transformed\_file\_path, row.names = FALSE)

print(paste("Transformed dataset saved as", transformed\_file\_path))

return(df)

}, error = function(e) {

print(paste("Error processing the dataset:", e))

return(NULL)

})

}

# Example usage

file\_path <- "Air\_Quality\_Dataset\_example.csv"

processed\_data <- transform\_air\_quality\_data(file\_path)