Jr Data Analyst

NGAIRA

2024-05-07

library(readxl)

##   
## Attaching package: 'readxl'

## The following object is masked from 'package:officer':  
##   
## read\_xlsx

Junior\_Data\_Analyst\_Data\_1\_ <- read\_excel("C:/Users/ADMIN/Downloads/Junior Data Analyst \_ Data (1).xlsx")  
View(Junior\_Data\_Analyst\_Data\_1\_)  
colnames(Junior\_Data\_Analyst\_Data\_1\_)<- tolower(colnames(Junior\_Data\_Analyst\_Data\_1\_))  
  
# Rename columns  
colnames(Junior\_Data\_Analyst\_Data\_1\_) <- c("hour", "date/time","solar","usage")

1. **Carry out checks on the data provided to confirm that the data is complete and fit for use. This should include**

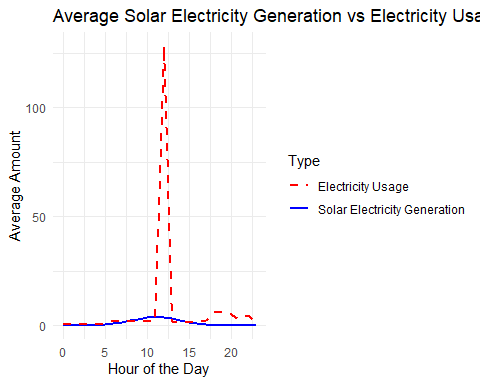
# Calculate average solar generation and electricity usage for each hour

Data\_Analyst <- aggregate(. ~ hour, data = Junior\_Data\_Analyst\_Data\_1\_, FUN = mean)  
  
Data\_Analyst$`date/time` <- as.POSIXct(Data\_Analyst$`date/time`)

# Create plot

ggplot(Data\_Analyst, aes(x = hour)) +  
 geom\_line(aes(y = solar, color = "Solar Electricity Generation"), size = 1) +  
 geom\_line(aes(y = usage, color = "Electricity Usage"), linetype = "dashed", size = 1) +  
 scale\_color\_manual(values = c("Solar Electricity Generation" = "blue", "Electricity Usage" = "red")) +  
 labs(title = "Average Solar Electricity Generation vs Electricity Usage by Hour",  
 x = "Hour of the Day",  
 y = "Average Amount",  
 color = "Type") +  
 theme\_minimal()

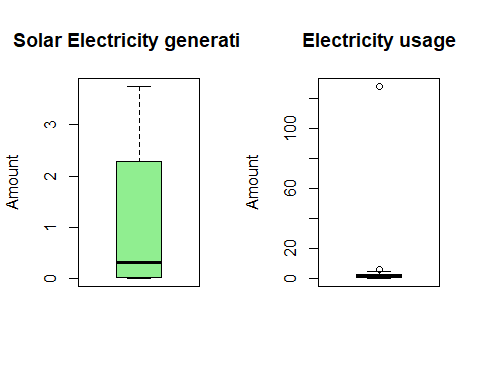
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.



# Create a boxplot to investigate outliers

# combine plots

par(mfrow = c(1, 2))   
boxplot(Data\_Analyst$solar, main= "Solar Electricity generation", ylab = "Amount", col = "lightgreen")  
boxplot(Data\_Analyst$usage, main= "Electricity usage", ylab = "Amount", col = "lightblue")



##PRESENCE OF AN OUTLIER # Define a function for Winsorization

winsorize <- function(x, threshold = 0.95) {  
 quantiles <- quantile(x, probs = c(threshold, 1 - threshold))  
 x[x > quantiles[2]] <- quantiles[2]  
 x[x < quantiles[1]] <- quantiles[1]  
 return(x)  
}

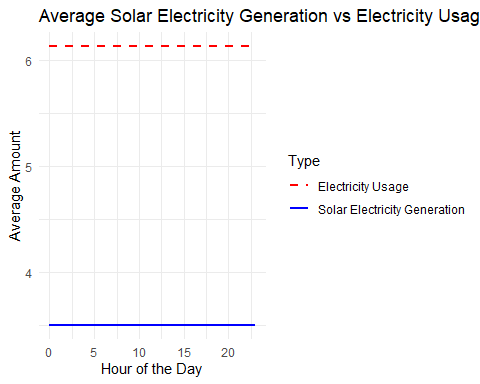
#APPLY WINSORIZE # Apply Winsorization to solar generation and electricity usage columns

Data\_Analyst$solar <- winsorize(Data\_Analyst$solar)  
Data\_Analyst$usage <- winsorize(Data\_Analyst$usage)

# Recalculate average data after Winsorization

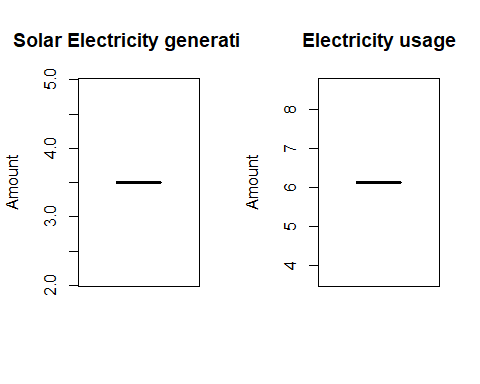
Data\_Analyst<- aggregate(. ~ hour, data = Data\_Analyst, FUN = mean)  
view(Data\_Analyst)

# Create plot  
ggplot(Data\_Analyst, aes(x = hour)) +  
 geom\_line(aes(y = solar, color = "Solar Electricity Generation"), size = 1) +  
 geom\_line(aes(y = usage, color = "Electricity Usage"), linetype = "dashed", size = 1) +  
 scale\_color\_manual(values = c("Solar Electricity Generation" = "blue", "Electricity Usage" = "red")) +  
 labs(title = "Average Solar Electricity Generation vs Electricity Usage by Hour",  
 x = "Hour of the Day",  
 y = "Average Amount",  
 color = "Type") +  
 theme\_minimal()



#CONFIRM PRESENCE OF OUTLIERS # Create a boxplot to investigate outliers # combine plots

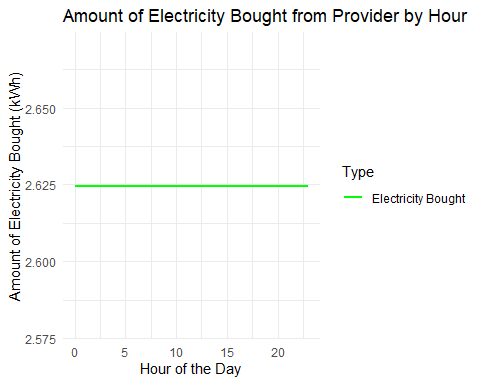
par(mfrow = c(1, 2))   
boxplot(Data\_Analyst$solar, main= "Solar Electricity generation", ylab = "Amount", col = "lightgreen")  
boxplot(Data\_Analyst$usage, main= "Electricity usage", ylab = "Amount", col = "lightblue")



1. **Calculate, for each hour in 2020, the amount of electricity that needed to be bought from the electricity provider (measured in kWh and subject to a minimum of zero).**

# Calculate the amount of electricity needed to be bought for each hour

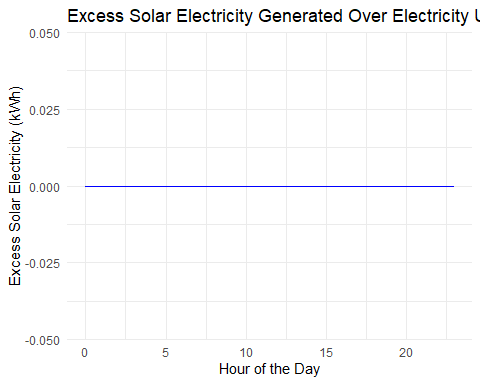
Data\_Analyst$buy\_from\_provider <- pmax(0, Data\_Analyst$usage - Data\_Analyst$solar)  
  
# Recalculate average data after adding buy\_from\_provider column  
Data\_Analyst <- aggregate(. ~ hour, data = Data\_Analyst, FUN = sum)  
  
# Create plot  
ggplot(Data\_Analyst, aes(x = hour)) +  
 geom\_line(aes(y = buy\_from\_provider, color = "Electricity Bought"), size = 1) +  
 scale\_color\_manual(values = c("Electricity Bought" = "green")) +  
 labs(title = "Amount of Electricity Bought from Provider by Hour",  
 x = "Hour of the Day",  
 y = "Amount of Electricity Bought (kWh)",  
 color = "Type") +  
 theme\_minimal()



1. **Calculate, for each hour in 2020, the excess solar electricity generated over electricity used (measured in kWh and subject to a minimum of zero).**

# Calculate the excess solar electricity generated over electricity used for each hour in 2020

Data\_Analyst$excess\_solar <- pmax(0, Data\_Analyst$solar - Data\_Analyst$usage)  
  
# Summarize excess solar electricity generated over electricity used for each hour in 2020  
excess\_solar\_2020 <- aggregate(excess\_solar ~ hour, data = Data\_Analyst, FUN = sum)  
  
# Create plot  
ggplot(excess\_solar\_2020, aes(x = hour, y = excess\_solar)) +  
 geom\_line(color = "blue") +  
 labs(title = "Excess Solar Electricity Generated Over Electricity Used in 2020",  
 x = "Hour of the Day",  
 y = "Excess Solar Electricity (kWh)") +  
 theme\_minimal()



1. **Model the cumulative battery charge level (measured in kWh) for each hour over 2020, assuming a battery had already been installed.**

**The battery charge level should:**

* + 1. **begin at zero at 1 January 2020 00:00.**
    2. **allow for the increase or decrease in charge level depending on the hourly results of parts (ii) and (iii).**
    3. **be subject to the cap on the maximum battery charge level.**

Model the cumulative battery charge level (measured in kWh) for each hour over 2020, assuming a battery had already been installed.

We iterate through each row of the data frame data, representing each hour.

For each hour, we calculate the net energy available for charging the battery by subtracting the electricity usage from the solar generation.

We then calculate the change in the battery charge level based on the net energy and the battery’s capacity, ensuring that the charge level does not exceed the capacity.

We update the cumulative charge level accordingly.

We store the hourly charge levels in a vector and create a data frame with timestamps and charge levels.

Finally, we plot the cumulative battery charge level over time

# Define battery parameters

battery\_capacity <- 1000 # kWh (example capacity)  
  
# Convert 'timestamp' to a POSIXct object if it's not already  
Data\_Analyst$`date/time` <- as.POSIXct(Data\_Analyst$`date/time`)

# Initialize cumulative charge level  
cumulative\_charge <- 0  
  
# Initialize vector to store hourly charge levels  
hourly\_charge\_levels <- numeric()

# Iterate through each hour of data  
for (i in 1:nrow(Data\_Analyst)) {  
 # Calculate net energy (solar generation - electricity usage)  
 net\_energy <- Data\_Analyst$solar[i] - Data\_Analyst$usage[i]  
}

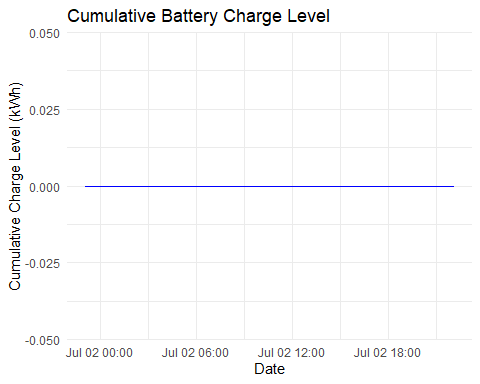
# Calculate charge level change based on net energy and battery capacity

charge\_change <- min(battery\_capacity - cumulative\_charge, max(-cumulative\_charge, net\_energy))  
   
 # Update cumulative charge level  
 cumulative\_charge <- cumulative\_charge + charge\_change  
   
 # Store hourly charge level  
 hourly\_charge\_levels <- c(hourly\_charge\_levels, cumulative\_charge)

# Create data frame with hourly charge levels

charge\_data <- data.frame(timestamp = Data\_Analyst$`date/time`, charge\_level = hourly\_charge\_levels)

# Plot cumulative battery charge level  
ggplot(charge\_data, aes(x = timestamp, y = charge\_level)) +  
 geom\_line(color = "blue") +  
 labs(title = "Cumulative Battery Charge Level",  
 x = "Date",  
 y = "Cumulative Charge Level (kWh)") +  
 theme\_minimal()



1. **Calculate the amount of electricity for each hour in 2020 that would have been bought from the electricity provider (measured in kWh and subject to a minimum of zero), assuming a battery had already been installed.**

To calculate the amount of electricity that would have been bought from the electricity provider for each hour, assuming a battery had already been installed, we need to consider the scenario where the battery is not able to meet the electricity demand entirely. In this case, any shortfall in meeting the demand with solar energy and the battery’s stored energy would need to be supplemented by purchasing electricity from the provider. Here’s how you can calculate it in R:

# Assuming ‘data’ contains columns ‘timestamp’, ‘solar’, ‘usage’, and ‘charge\_level’

# Define battery parameters  
battery\_capacity <- 1000 # kWh (example capacity)  
  
# Initialize vector to store amount of electricity bought from provider for each hour  
electricity\_bought <- numeric()  
  
# Iterate through each hour of data  
for (i in 1:nrow(Data\_Analyst)) {  
 # Calculate net energy (solar generation - electricity usage)  
 net\_energy <- Data\_Analyst$solar[i] - Data\_Analyst$usage[i]  
}

# Calculate available energy from battery

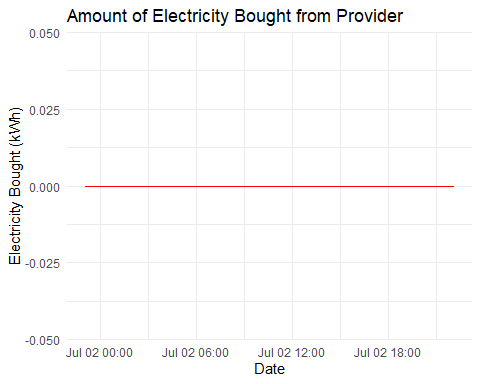
battery\_energy <- min(battery\_capacity, Data\_Analyst$charge\_level[i])

# Calculate total available energy (solar + battery)  
 total\_energy <- Data\_Analyst$solar[i] + battery\_energy

# Calculate amount of electricity bought from provider (if shortfall)  
 buy\_from\_provider <- max(0, Data\_Analyst$usage[i] - total\_energy)  
   
 # Store amount of electricity bought from provider  
 electricity\_bought <- c(electricity\_bought, buy\_from\_provider)

# Create data frame with amount of electricity bought from provider for each hour  
bought\_data <- data.frame(timestamp = Data\_Analyst$`date/time`, electricity\_bought = electricity\_bought)

# Plot amount of electricity bought from provider  
ggplot(bought\_data, aes(x = timestamp, y = electricity\_bought)) +  
 geom\_line(color = "red") +  
 labs(title = "Amount of Electricity Bought from Provider",  
 x = "Date",  
 y = "Electricity Bought (kWh)") +  
 theme\_minimal()



We iterate through each row of the data frame data, representing each hour.

For each hour, we calculate the total available energy, which includes energy generated by solar and stored energy in the battery.

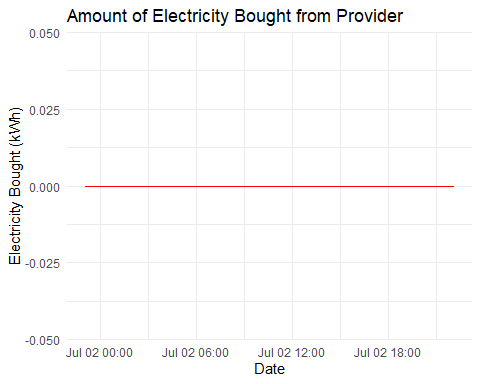
We calculate the amount of electricity that needs to be bought from the provider to meet the electricity demand not covered by solar and battery energy.

We store the amount of electricity bought from the provider in a vector and create a data frame with timestamps and electricity bought values.

Finally, we plot the amount of electricity bought from the provider over time.

1. To calculate the amount of electricity that would have been bought from the electricity provider for each hour, assuming a battery had already been installed, we need to consider the scenario where the battery is not able to meet the electricity demand entirely. In this case, any shortfall in meeting the demand with solar energy and the battery’s stored energy would need to be supplemented by purchasing electricity from the provider.

# Define battery parameters  
battery\_capacity <- 1000 # kWh (example capacity)  
# Initialize vector to store amount of electricity bought from provider for each hour  
electricity\_bought <- numeric()  
  
# Iterate through each hour of data  
for (i in 1:nrow(Data\_Analyst)) {  
 # Calculate net energy (solar generation - electricity usage)  
 net\_energy <- Data\_Analyst$solar[i] - Data\_Analyst$usage[i]  
   
 # Calculate available energy from battery  
 battery\_energy <- min(battery\_capacity, Data\_Analyst$charge\_level[i])  
   
 # Calculate total available energy (solar + battery)  
 total\_energy <- Data\_Analyst$solar[i] + battery\_energy  
   
 # Calculate amount of electricity bought from provider (if shortfall)  
 buy\_from\_provider <- max(0, Data\_Analyst$usage[i] - total\_energy)  
   
 # Store amount of electricity bought from provider  
 electricity\_bought <- c(electricity\_bought, buy\_from\_provider)  
}  
  
# Create data frame with amount of electricity bought from provider for each hour  
bought\_data <- data.frame(timestamp = Data\_Analyst$`date/time`, electricity\_bought = electricity\_bought)  
  
# Plot amount of electricity bought from provider  
ggplot(bought\_data, aes(x = timestamp, y = electricity\_bought)) +  
 geom\_line(color = "red") +  
 labs(title = "Amount of Electricity Bought from Provider",  
 x = "Date",  
 y = "Electricity Bought (kWh)") +  
 theme\_minimal()



1. **Calculate the saving over 2020 (in dollars ($), using 1 January 2022 electricity prices and ignoring discounting) from installing a battery compared to using the existing solar panels alone.**

To calculate the savings over 2020 from installing a battery compared to using existing solar panels alone, we need to consider the difference in electricity costs between purchasing electricity from the provider and using the battery to store excess solar energy.

Obtain electricity prices for 1 January 2022.

Calculate the total amount of electricity bought from the provider with and without the battery.

Multiply the difference in electricity bought by the price difference to get the savings.

# Define electricity prices  
electricity\_price\_2022 <- 0.12 # $/kWh (example price for 1 January 2022)  
  
# Convert 'timestamp' to a POSIXct object if it's not already  
Data\_Analyst$timestamp <- as.POSIXct(Data\_Analyst$`date/time`)  
  
# Initialize vector to store amount of electricity bought from provider with and without the battery  
electricity\_bought\_without\_battery <- numeric()  
electricity\_bought\_with\_battery <- numeric()  
  
# Iterate through each hour of data  
for (i in 1:nrow(Data\_Analyst)) {  
 # Calculate net energy (solar generation - electricity usage)  
 net\_energy <- Data\_Analyst$solar[i] - Data\_Analyst$usage[i]  
   
 # Calculate available energy from battery  
 battery\_energy <- min(battery\_capacity, Data\_Analyst$charge\_level[i])  
   
 # Calculate total available energy (solar + battery)  
 total\_energy <- Data\_Analyst$solar[i] + battery\_energy  
   
 # Calculate amount of electricity bought from provider without the battery  
 buy\_from\_provider\_without\_battery <- max(0, Data\_Analyst$usage[i] - Data\_Analyst$solar[i])  
   
 # Calculate amount of electricity bought from provider with the battery  
 buy\_from\_provider\_with\_battery <- max(0, Data\_Analyst$usage[i] - total\_energy)  
   
 # Store amounts of electricity bought from provider  
 electricity\_bought\_without\_battery <- c(electricity\_bought\_without\_battery, buy\_from\_provider\_without\_battery)  
 electricity\_bought\_with\_battery <- c(electricity\_bought\_with\_battery, buy\_from\_provider\_with\_battery)  
}  
  
# Calculate total electricity cost without the battery  
total\_cost\_without\_battery <- sum(electricity\_bought\_without\_battery) \* electricity\_price\_2022  
  
# Calculate total electricity cost with the battery  
total\_cost\_with\_battery <- sum(electricity\_bought\_with\_battery) \* electricity\_price\_2022  
  
# Calculate savings  
savings <- total\_cost\_without\_battery - total\_cost\_with\_battery  
  
# Print savings  
cat("Savings over 2020 from installing a battery compared to using existing solar panels alone: $", savings, "\n")

## Savings over 2020 from installing a battery compared to using existing solar panels alone: $ 7.558969

1. **Tabulate the data appropriately and then produce a chart to illustrate, on a monthly basis for the calendar year and measured in kWh, the:**
   * 1. **monthly solar generation.**
     2. **monthly electricity usage.**
     3. **monthly electricity purchased from the electricity provider (no battery).**
     4. **monthly electricity purchased from the electricity provider (with battery).**

To tabulate the data appropriately and then produce a chart to illustrate the monthly solar generation, electricity usage, electricity purchased from the electricity provider (without battery), and electricity purchased from the electricity provider (with battery) on a monthly basis for the calendar year, #Aggregate the data by month to calculate monthly totals. #Create a data frame or data table to store the monthly totals. #Produce a chart to visualize the monthly data.

# Convert 'timestamp' to a POSIXct object if it's not already  
Data\_Analyst$timestamp <- as.POSIXct(Data\_Analyst$`date/time`)  
  
# Extract month and year from timestamp  
Data\_Analyst$month <- format(Data\_Analyst$timestamp, "%Y-%m")  
  
# Aggregate data by month to calculate monthly totals  
monthly\_totals <- aggregate(cbind(solar, usage, buy\_from\_provider\_without\_battery, buy\_from\_provider\_with\_battery) ~ month, data = Data\_Analyst, FUN = sum)  
  
# Plot monthly data  
ggplot(monthly\_totals, aes(x = month)) +  
 geom\_line(aes(y = solar, color = "Solar Generation"), size = 1) +  
 geom\_line(aes(y = usage, color = "Electricity Usage"), size = 1) +  
 geom\_line(aes(y = buy\_from\_provider\_without\_battery, color = "Electricity Purchased (No Battery)"), size = 1, linetype = "dashed") +  
 geom\_line(aes(y = buy\_from\_provider\_with\_battery, color = "Electricity Purchased (With Battery)"), size = 1, linetype = "dashed") +  
 labs(title = "Monthly Electricity Data",  
 x = "Month",  
 y = "Monthly Amount (kWh)",  
 color = "Type") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))

## `geom\_line()`: Each group consists of only one observation.  
## ℹ Do you need to adjust the group aesthetic?  
## `geom\_line()`: Each group consists of only one observation.  
## ℹ Do you need to adjust the group aesthetic?  
## `geom\_line()`: Each group consists of only one observation.  
## ℹ Do you need to adjust the group aesthetic?  
## `geom\_line()`: Each group consists of only one observation.  
## ℹ Do you need to adjust the group aesthetic?

