DATA PREPARATION PHASE

```{r pressure, echo=FALSE}

# install.packages("arrow")

# install.packages("tidyverse)

library(arrow)

library(tidyverse)



# 1. Static House Data

house_data_path <-  "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/static_house_info.parquet"

static_house_data <-  read_parquet(house_data_path)

view(static_house_data)


#2. Energy Usage Data

energy_data_path <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/2023-houseData/.parquet"

energy_usage_data <- read_parquet(energy_data_path)

view(energy_usage_data)


energy_data_path1 <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/2023-houseData/65.parquet"

energy_usage_data1 <- read_parquet(energy_data_path1)

view(energy_usage_data1)


#3. Meta Data File

Meta_data <- read_csv("C:/Users/Soundarya Ravi/Downloads/data_dictionary.csv")

view(Meta_data)


#4. Weather Data
```

```
Weather_data <- read_csv("C:/Users/Soundarya Ravi/Downloads/G4500010.csv")
view(Weather_data)

# Unique Buildings
unique_building_id <- unique(static_house_data$bldg_id)
class(unique_building_id)
paste0(length(unique_building_id))

```

```{r}
energy_path <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/2023-houseData/"
list_of_dfs_final <- list()
# Define the iteration points
iteration_list <- c(1500, 3000, 4500, 5710)

for (iter in iteration_list) {
  list_of_dfs <- list()
  cat("-----------------------------------------------> New Iter : ", iter)

  # Filter parent list for each iteration - This is the list of Buildings that we will load
  building_id_filtered_list <- unique_building_id[seq_len(iter)]

  # All elements in building_id_filtered_list
  for (i in seq_along(building_id_filtered_list)) {
    elem <- building_id_filtered_list[i]
    path <- paste0(energy_path, as.character(elem), ".parquet")

    completition_status <- i * 100 / length(building_id_filtered_list)
```

```
  print(path)
  cat(" Completion Status: ", completition_status, "%", " Iteration: ", iter, " where i = ", i)

  # Filter for July and add Building Number
  df <- read_parquet(path)
  df <- subset(df, grepl("2018-07", time))
  df$bldg_id <- elem
  cat(" Datatype of df: ", class(df))

  # Add DF to List
  list_of_dfs[[i]] <- df

  # Break Loop at 10% completion
  # if (completition_status > iter) {break}
 }

 # Concat Dataframes
 pre_final_df <- do.call(rbind, list_of_dfs)

 # Add DF to Master List
 list_of_dfs_final[[iter]] <- pre_final_df
 # Size of DF - Check
}
# Combine all dataframes into one
final_df <- do.call(rbind, list_of_dfs_final)
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```{r}

library(tidyverse)

energy_merged_path <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/energy_final.csv")

```

```{r}

dim(energy_merged_path)


df <- head(energy_merged_path, 1000)

view(df)

```

```{r}


#Merging Up the Weather based on the county ID

print(unique(static_house_data$in.county))


list_county <- c("G4500910", "G4500730", "G4500710", "G4500790", "G4500450", "G4500150", "G4500350", "G4500190", "G4500830",

  "G4500510", "G4500070", "G4500670", "G4500750", "G4500290", "G4500490", "G4500130", "G4500630", "G4500870",

  "G4500550", "G4500010", "G4500430", "G4500890", "G4500850", "G4500770", "G4500030", "G4500590", "G4500610",

  "G4500250", "G4500530", "G4500210", "G4500410", "G4500570", "G4500690", "G4500310", "G4500090", "G4500470",
```

```
  "G4500050", "G4500330", "G4500650", "G4500230", "G4500270", "G4500370", "G4500110", "G4500170", "G4500390",
  "G4500810")
length(list_county)
```

```{r}
# Create an empty data frame to store the merged data
merged_df <- data.frame()

# List of county codes
list_county <- c("G4500910", "G4500730", "G4500710", "G4500790", "G4500450", "G4500150", "G4500350", "G4500190", "G4500830",
  "G4500510", "G4500070", "G4500670", "G4500750", "G4500290", "G4500490", "G4500130", "G4500630", "G4500870",
  "G4500550", "G4500010", "G4500430", "G4500890", "G4500850", "G4500770", "G4500030", "G4500590", "G4500610",
  "G4500250", "G4500530", "G4500210", "G4500410", "G4500570", "G4500690", "G4500310", "G4500090", "G4500470",
  "G4500050", "G4500330", "G4500650", "G4500230", "G4500270", "G4500370", "G4500110", "G4500170", "G4500390",
  "G4500810")

# Iterate through each county code
for (county in list_county) {
  url <- paste0('https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/weather/2023-weather-data/', county, '.csv')
```

```
  # Use tryCatch to handle errors
  tryCatch({
    df_county <- read.csv(url)
    merged_df <- rbind(merged_df, df_county)
  }, error = function(e) {
    cat("Error reading file for county", county, ":", conditionMessage(e), "\n")
  })
}



# Print the first few rows of the merged data frame
print(head(merged_df))
```

```{r}
# Create an empty data frame to store the merged data
merged_df_new<- data.frame()

# List of county codes
list_county <- c("G4500910", "G4500730", "G4500710", "G4500790", "G4500450", "G4500150",
"G4500350", "G4500190", "G4500830",
  "G4500510", "G4500070", "G4500670", "G4500750", "G4500290", "G4500490", "G4500130",
"G4500630", "G4500870",
  "G4500550", "G4500010", "G4500430", "G4500890", "G4500850", "G4500770", "G4500030",
"G4500590", "G4500610",
  "G4500250", "G4500530", "G4500210", "G4500410", "G4500570", "G4500690", "G4500310",
"G4500090", "G4500470",
  "G4500050", "G4500330", "G4500650", "G4500230", "G4500270", "G4500370", "G4500110",
"G4500170", "G4500390",
```

```
  "G4500810")

# Iterate through each county code
for (county in list_county) {
  url <- paste0('https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/weather/2023-weather-data/', county, '.csv')

  # Use tryCatch to handle errors
  tryCatch({
    df_county <- read.csv(url)
    # Add a new column 'county_id' with the current county code
    df_county$county_id <- county
    merged_df_new <- rbind(merged_df_new, df_county)
  }, error = function(e) {
    cat("Error reading file for county", county, ":", conditionMessage(e), "\n")
  })
}

# Print the first few rows of the merged data frame
print(head(merged_df_new))
```

```{r}
view(merged_df_new)
```

```

```{r}
energy_merged_path<- na.omit(energy_merged_path)
```

```{r}
# Assuming merged_df has a column named date_time

# Load the dplyr package
library(dplyr)

# Filter merged_df for the month of July
merged_df_new<- merged_df_new %>%
  filter(format(as.Date(date_time), "%Y-%m") == "2018-07")

# Print the first few rows of the filtered data frame
print(head(merged_df_new))
nrow(merged_df_new)

```

```{r}
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(final_df, file ="energy_final.csv",row.names = FALSE)
```
```

```{r}

weather_data_merged_with_county <- merged_df_new

```

```{r}
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(weather_data_merged_with_county, file ="weather_final_county.csv",row.names = FALSE)



```
```{r}
#view(static_house_data)
print(unique(static_house_data$in.ahs_region))
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")
```



```{r}
print(unique(static_house_data$in.hvac_system_is_faulted
))
```

```{r}

meta_data_path <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/data_dictionary.csv"

input_df_metadata <- read_csv(meta_data_path,show_col_types = FALSE)

meta_data_df=as.data.frame(input_df_metadata)

view(meta_data_df)

```

```{r}

view(weather_data_merged)

```

```{r}

str(weather_data_merged_with_county)

```

```{r}

#Grouping the time frame of Weather File

#str(weather_data_merged)

# Assuming 'weather_data_merged' contains data for multiple counties

# Assuming 'weather_data_merged' contains data for multiple counties

result_df <- weather_data_merged_with_county %>%

  mutate(hour = hour(strptime(date_time, format = "%Y-%m-%d %H:%M:%S")),

         time_split = case_when(

           hour %in% c(0, 1, 2, 3) ~ "Late Night",

           hour %in% c(4, 5, 6, 7, 8) ~ "Early Morning",
```

```
        hour %in% c(9, 10, 11, 12) ~ "Morning",
        hour %in% c(13, 14, 15) ~ "Noon",
        hour %in% c(16, 17, 18) ~ "Evening",
        hour %in% c(19, 20, 21, 22, 23) ~ "Night",
        TRUE ~ "Other"
      )) %>%
group_by(county_id, time_split) %>%
summarise(
  Dry_Bulb_Temperature_C = mean(Dry.Bulb.Temperature...C.),
  Relative_Humidity = mean(Relative.Humidity....),
  Wind_Speed_m_s = mean(Wind.Speed..m.s.),
  Wind_Direction_Deg = mean(Wind.Direction..Deg.),
  Global_Horizontal_Radiation_W_m2 = mean(Global.Horizontal.Radiation..W.m2.),
  Direct_Normal_Radiation_W_m2 = mean(Direct.Normal.Radiation..W.m2.),
  Diffuse_Horizontal_Radiation_W_m2 = mean(Diffuse.Horizontal.Radiation..W.m2.)
) %>%
ungroup() %>%
mutate(
  time_range = case_when(
    time_split == "Late Night" ~ "00:00:00 to 03:00:00",
    time_split == "Early Morning" ~ "04:00:00 to 08:00:00",  # Adjust as needed
    time_split == "Morning" ~ "09:00:00 to 12:00:00",        # Adjust as needed
    time_split == "Noon" ~ "13:00:00 to 15:00:00",           # Adjust as needed
    time_split == "Evening" ~ "16:00:00 to 18:00:00",        # Adjust as needed
    time_split == "Night" ~ "19:00:00 to 23:00:00",          # Adjust as needed
    TRUE ~ "Other"
  )
) %>%
arrange(county_id, time_range)
```

```
# Print the result
print(result_df)
```

```
```{r}

# Assuming 'result_df' is your existing dataframe

weather_july_timeframe <- result_df %>%
  select(county_id, time_range, time_split, everything())

# Print the new dataframe
print(weather_july_timeframe)
```

```
```{r}
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(weather_july_timeframe, file ="weather_tf_july_276.csv",row.names = FALSE)
```

```
```{r}

energy_merged_july <- read_csv("C:/Users/Soundarya Ravi/Downloads/energydataaa_idsProj.csv") #subhiksha

```

```{r}

#energy_combined_features <- read_csv("C:/Users/Soundarya Ravi/Downloads/combined_dependent_energy_data.csv") #subhiksha


head(energy_combined_features)

head(energy_merged_july)

str(energy_combined_features)

```

---

title: "Energy_new"

output: html_document

date: "2023-12-02"

```{r setup, include=FALSE}

knitr::opts_chunk$set(echo = TRUE)

```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.
```

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

```{r}

library(tidyverse)

energy_merged_path <- read_csv("")

```


```{r}

weather_combined <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/weather_tf_july_276.csv")

```



```{r}

#raw_energy_merged <- read_csv("C:/Users/Soundarya Ravi/Downloads/rawenergymerge.csv")

raw_energy_merged <- na.omit(raw_energy_merged)

nrow(raw_energy_merged)


```


```{r}




```


```{r}
```

raw_energy_merged$out.electricity.pv.energy_consumption[raw_energy_merged$out.electricity.pv.energy_consumption < 0] <- 0

```

```{r}
summary(raw_energy_merged)
```

```{r}
library(tidyverse)
#path <- "C:/Users/Soundarya Ravi/Desktop/Shiny/Raw Files/rawenergyfinal.csv"
data <- raw_energy_merged

#view(data111)
head(data,100)
colnames(data)

# Kitchen
# out.electricity.range_oven.energy_consumption
# out.electricity.dishwasher.energy_consumption
# out.electricity.refrigerator.energy_consumption
# out.electricity.freezer.energy_consumption
# out.natural_gas.range_oven.energy_consumption
# out.natural_gas.grill.energy_consumption
# out.propane.range_oven.energy_consumption
#
```
```

laundry

out.electricity.clothes_dryer.energy_consumption

out.natural_gas.clothes_dryer.energy_consumption

out.electricity.clothes_washer.energy_consumption

out.propane.clothes_dryer.energy_consumption

#

heating_cooling

out.electricity.heating_fans_pumps.energy_consumption

out.electricity.heating_hp_bkup.energy_consumption

out.electricity.heating.energy_consumption

out.electricity.cooling.energy_consumption

out.natural_gas.heating_hp_bkup.energy_consumption

out.natural_gas.heating.energy_consumption

out.propane.heating_hp_bkup.energy_consumption

out.propane.heating.energy_consumption

out.fuel_oil.heating_hp_bkup.energy_consumption

out.fuel_oil.heating.energy_consumption

out.natural_gas.fireplace.energy_consumption

out.electricity.cooling_fans_pumps.energy_consumption

#

water_heating

out.electricity.hot_water.energy_consumption

out.fuel_oil.hot_water.energy_consumption

out.natural_gas.hot_water.energy_consumption

out.propane.hot_water.energy_consumption

#

electrical_appliances

out.electricity.lighting_exterior.energy_consumption

out.electricity.lighting_garage.energy_consumption

```
# out.electricity.lighting_interior.energy_consumption
# out.electricity.plug_loads.energy_consumption
# out.electricity.mech_vent.energy_consumption
# out.natural_gas.lighting.energy_consumption
# out.electricity.ceiling_fan.energy_consumption
#
# outdoor_appliances
# out.electricity.hot_tub_heater.energy_consumption
# out.electricity.hot_tub_pump.energy_consumption
# out.electricity.pool_heater.energy_consumption
# out.electricity.pool_pump.energy_consumption
# out.natural_gas.hot_tub_heater.energy_consumption
# out.natural_gas.pool_heater.energy_consumption
# out.electricity.well_pump.energy_consumption
#
# renewable_energy
# out.electricity.pv.energy_consumption

# kitchen
data$out.kitchen.energy_consumption <- data$out.electricity.range_oven.energy_consumption +
  data$out.electricity.dishwasher.energy_consumption +
  data$out.electricity.refrigerator.energy_consumption +
  data$out.electricity.freezer.energy_consumption +
  data$out.natural_gas.range_oven.energy_consumption +
  data$out.natural_gas.grill.energy_consumption +
  data$out.propane.range_oven.energy_consumption

# laundry
data$out.laundry.energy_consumption <- data$out.electricity.clothes_dryer.energy_consumption +
```

```
  data$out.natural_gas.clothes_dryer.energy_consumption +
  data$out.electricity.clothes_washer.energy_consumption +
  data$out.propane.clothes_dryer.energy_consumption

# heating_cooling
data$out.heating_cooling.energy_consumption <-
data$out.electricity.heating_fans_pumps.energy_consumption +
  data$out.electricity.heating_hp_bkup.energy_consumption +
  data$out.electricity.heating.energy_consumption +
  data$out.electricity.cooling.energy_consumption +
  data$out.natural_gas.heating_hp_bkup.energy_consumption +
  data$out.natural_gas.heating.energy_consumption +
  data$out.propane.heating_hp_bkup.energy_consumption +
  data$out.propane.heating.energy_consumption +
  data$out.fuel_oil.heating_hp_bkup.energy_consumption +
  data$out.fuel_oil.heating.energy_consumption +
  data$out.natural_gas.fireplace.energy_consumption +
  data$out.electricity.cooling_fans_pumps.energy_consumption

# water_heating
data$out.water_heating.energy_consumption <- data$out.electricity.hot_water.energy_consumption +
  data$out.fuel_oil.hot_water.energy_consumption +
  data$out.natural_gas.hot_water.energy_consumption +
  data$out.propane.hot_water.energy_consumption

# electrical_appliances
data$out.electrical_appliances.energy_consumption <-
data$out.electricity.lighting_exterior.energy_consumption +
  data$out.electricity.lighting_garage.energy_consumption +
```

```
  data$out.electricity.lighting_interior.energy_consumption +
  data$out.electricity.plug_loads.energy_consumption +
  data$out.electricity.mech_vent.energy_consumption +
  data$out.natural_gas.lighting.energy_consumption +
  data$out.electricity.ceiling_fan.energy_consumption

# outdoor_appliances
data$out.outdoor_appliances.energy_consumption <-
data$out.electricity.hot_tub_heater.energy_consumption +
  data$out.electricity.hot_tub_pump.energy_consumption +
  data$out.electricity.pool_heater.energy_consumption +
  data$out.electricity.pool_pump.energy_consumption +
  data$out.natural_gas.hot_tub_heater.energy_consumption +
  data$out.natural_gas.pool_heater.energy_consumption +
  data$out.electricity.well_pump.energy_consumption

# renewable_energy
data$out.renewable_energy.energy_consumption <- data$out.electricity.pv.energy_consumption

#total
data$out.total.energy_consumption <- data$out.electricity.range_oven.energy_consumption +
  data$out.electricity.dishwasher.energy_consumption +
  data$out.electricity.refrigerator.energy_consumption +
  data$out.electricity.freezer.energy_consumption +
  data$out.natural_gas.range_oven.energy_consumption +
  data$out.natural_gas.grill.energy_consumption +
  data$out.propane.range_oven.energy_consumption +
  data$out.electricity.clothes_dryer.energy_consumption +
  data$out.natural_gas.clothes_dryer.energy_consumption +
```

```
data$out.electricity.clothes_washer.energy_consumption +
data$out.propane.clothes_dryer.energy_consumption +
data$out.electricity.heating_fans_pumps.energy_consumption +
data$out.electricity.heating_hp_bkup.energy_consumption +
data$out.electricity.heating.energy_consumption +
data$out.electricity.cooling.energy_consumption +
data$out.natural_gas.heating_hp_bkup.energy_consumption +
data$out.natural_gas.heating.energy_consumption +
data$out.propane.heating_hp_bkup.energy_consumption +
data$out.propane.heating.energy_consumption +
data$out.fuel_oil.heating_hp_bkup.energy_consumption +
data$out.fuel_oil.heating.energy_consumption +
data$out.natural_gas.fireplace.energy_consumption +
data$out.electricity.cooling_fans_pumps.energy_consumption +
data$out.electricity.hot_water.energy_consumption +
data$out.fuel_oil.hot_water.energy_consumption +
data$out.natural_gas.hot_water.energy_consumption +
data$out.propane.hot_water.energy_consumption +
data$out.electricity.lighting_exterior.energy_consumption +
data$out.electricity.lighting_garage.energy_consumption +
data$out.electricity.lighting_interior.energy_consumption +
data$out.electricity.plug_loads.energy_consumption +
data$out.electricity.mech_vent.energy_consumption +
data$out.natural_gas.lighting.energy_consumption +
data$out.electricity.ceiling_fan.energy_consumption +
data$out.electricity.hot_tub_heater.energy_consumption +
data$out.electricity.hot_tub_pump.energy_consumption +
data$out.electricity.pool_heater.energy_consumption +
data$out.electricity.pool_pump.energy_consumption +
```

```
  data$out.natural_gas.hot_tub_heater.energy_consumption +
  data$out.natural_gas.pool_heater.energy_consumption +
  data$out.electricity.well_pump.energy_consumption +
  data$out.electricity.pv.energy_consumption

colnames(data[,1:45])
colnames(data[,45:53])
data_dependent <- data[,44:53]



setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(data_dependent, file ="energy_merged_column.csv",row.names = FALSE)
```

```

```{r}
head(data)
```

```{r}

#nrow(data_dependent)
head(data_dependent)
```
```

```
str(data_dependent)

```

```{r}
library(dplyr)
library(tibble)

process_energy_data <- function(energy_data) {
  result_df <- energy_data %>%
    mutate(hour = hour(time),
           time_split = case_when(
             hour %in% c(0, 1, 2, 3) ~ "Late Night",
             hour %in% c(4, 5, 6, 7, 8) ~ "Early Morning",
             hour %in% c(9, 10, 11, 12) ~ "Morning",
             hour %in% c(13, 14, 15) ~ "Noon",
             hour %in% c(16, 17, 18) ~ "Evening",
             hour %in% c(19, 20, 21, 22, 23) ~ "Night",
             TRUE ~ "Other"
           )) %>%
    group_by(bldg_id, time_split) %>%
    summarise(
      out.kitchen_energy_consumption = mean(out.kitchen.energy_consumption),
      out.laundry_energy_consumption = mean(out.laundry.energy_consumption),
      out.heating_cooling_energy_consumption = mean(out.heating_cooling.energy_consumption),
      out.water_heating_energy_consumption = mean(out.water_heating.energy_consumption),
      out.electrical_appliances_energy_consumption =
mean(out.electrical_appliances.energy_consumption),
```

```
    out.outdoor_appliances_energy_consumption =
mean(out.outdoor_appliances.energy_consumption),
    out.renewable_energy_energy_consumption = mean(out.renewable_energy.energy_consumption),
    out.total_energy_consumption = mean(out.total.energy_consumption)
  ) %>%
  ungroup() %>%
  mutate(
    time_range = case_when(
      time_split == "Late Night" ~ "00:00:00 to 03:00:00",
      time_split == "Early Morning" ~ "04:00:00 to 08:00:00",
      time_split == "Morning" ~ "09:00:00 to 12:00:00",
      time_split == "Noon" ~ "13:00:00 to 15:00:00",
      time_split == "Evening" ~ "16:00:00 to 18:00:00",
      time_split == "Night" ~ "19:00:00 to 23:00:00",
      TRUE ~ "Other"
    )
  ) %>%
  arrange(bldg_id, time_range)

 return(result_df)
}

# Example usage:
# Assuming your energy dataframe is named energy_data
energy_result_df <- process_energy_data(data_dependent)

# Print the result
print(energy_result_df)
```

```

```{r}
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(energy_result_df, file ="energy_aggregated_final.csv",row.names = FALSE)
```

```{r}
library(arrow)
house_data_path <-  "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/static_house_info.parquet"
static_house_data <-  read_parquet(house_data_path)

Static_Columns_Filtered <-  c('bldg_id','in.county', 'in.county_and_puma','in.sqft',
'in.bedrooms','in.building_america_climate_zone', 'in.ceiling_fan', 'in.city', 'in.clothes_dryer',
'in.clothes_washer', 'in.cooking_range', 'in.cooling_setpoint',
'in.cooling_setpoint_offset_magnitude','in.dishwasher','in.federal_poverty_level',
'in.geometry_attic_type', 'in.geometry_floor_area', 'in.geometry_floor_area_bin',
'in.geometry_garage','in.heating_fuel', 'in.heating_setpoint','in.heating_setpoint_offset_magnitude',
'in.hot_water_fixtures', 'in.hvac_cooling_efficiency',
'in.hvac_cooling_partial_space_conditioning','in.hvac_cooling_type', 'in.hvac_has_ducts',
'in.hvac_has_zonal_electric_heating', 'in.hvac_heating_efficiency', 'in.hvac_heating_type',
'in.hvac_heating_type_and_fuel','in.income','in.income_recs_2015', 'in.income_recs_2020',
'in.infiltration', 'in.insulation_ceiling', 'in.insulation_floor', 'in.insulation_foundation_wall',
'in.insulation_rim_joist', 'in.insulation_roof', 'in.insulation_slab', 'in.insulation_wall', 'in.lighting',
'in.misc_extra_refrigerator', 'in.misc_freezer', 'in.misc_gas_fireplace','in.misc_gas_grill',
'in.misc_gas_lighting', 'in.misc_hot_tub_spa', 'in.misc_pool', 'in.misc_pool_heater', 'in.misc_pool_pump',
'in.misc_well_pump', 'in.natural_ventilation','in.occupants', 'in.orientation', 'in.plug_load_diversity',
'in.refrigerator', 'in.roof_material', 'in.tenure','in.usage_level', 'in.vacancy_status', 'in.vintage',
'in.vintage_acs','in.water_heater_efficiency', 'in.water_heater_fuel',
'in.weather_file_city','in.weather_file_latitude','in.weather_file_longitude',
```
```

'in.window_areas','in.windows', 'upgrade.insulation_roof', 'upgrade.water_heater_efficiency',
'upgrade.hvac_cooling_efficiency', 'upgrade.infiltration_reduction',
'upgrade.geometry_foundation_type','upgrade.clothes_dryer', 'upgrade.insulation_ceiling',
'upgrade.hvac_heating_type', 'upgrade.insulation_wall', 'upgrade.insulation_foundation_wall',

'upgrade.hvac_heating_efficiency', 'upgrade.cooking_range')

static_house_filtered <- static_house_data %>% select(all_of(Static_Columns_Filtered))

```

```{r}

setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(static_house_filtered, file ="statichousefirstcopy.csv",row.names = FALSE)

```

```{r}

unique_values <- sapply(static_house_filtered, unique)

print(unique_values)

copy_SHF <- static_house_filtered

```

```{r}

str(copy_SHF)

static_house_filtered <- copy_SHF

```
```

```{r}

# Coding it ordinally for the model to understand

# Ordinal coding for in.sqft

in_sqft_mapping <- c("885"=3, "1220"=4, "1690"=5, "2176"=6, "2663"=7, "3301"=8, "8194"=9, "328"=1, "633"=2)

static_house_filtered$in.sqft <- as.numeric(in_sqft_mapping[as.character(static_house_filtered$in.sqft)])



# Ordinal coding for in.bedrooms

in_bedrooms_mapping <- c("3"=3, "2"=2, "4"=4, "1"=1, "5"=5)

static_house_filtered$in.bedrooms <- as.numeric(in_bedrooms_mapping[as.character(static_house_filtered$in.bedrooms)])


# Ordinal coding for in.building_america_climate_zone

in_building_america_climate_zone_mapping <- c("Mixed-Humid"=1, "Hot-Humid"=2)

static_house_filtered$in.building_america_climate_zone <- as.numeric(in_building_america_climate_zone_mapping[static_house_filtered$in.building_america_climate_zone])


# Ordinal coding for in.ceiling_fan

in_ceiling_fan_mapping <- c("Standard Efficiency"=2, "None"=0, "Standard Efficiency, No usage"=1)

static_house_filtered$in.ceiling_fan <- as.numeric(in_ceiling_fan_mapping[static_house_filtered$in.ceiling_fan])


# # Ordinal coding for in.city (assuming the given order)

# in_city_mapping <- c(
```

```
#   "SC, Rock Hill"=1, "Not in a census Place"=2, "In another census Place"=3, "SC, Goose Creek"=4,
#   "SC, Mount Pleasant"=5, "SC, Sumter"=6, "SC, Charleston"=7, "SC, Hilton Head Island"=8,
# #   "SC, North Charleston"=9, "SC, Greenville"=10, "SC, Myrtle Beach"=11, "SC, Columbia"=12,
# #   "SC, Florence"=13, "SC, North Myrtle Beach"=14, "SC, Spartanburg"=15, "SC, Summerville"=16
# # )
# static_house_filtered$in.city <- as.numeric(in_city_mapping[static_house_filtered$in.city])

# Ordinal coding for in.clothes_dryer
in_clothes_dryer_mapping <- c(
  "Gas, 100% Usage"= 5, "Electric, 100% Usage"=2, "Electric, 80% Usage"=1,
  "Electric, 120% Usage"=3, "None"=0, "Propane, 100% Usage"=8,
  "Gas, 120% Usage"=6, "Propane, 80% Usage"=7, "Gas, 80% Usage"=4,
  "Propane, 120% Usage"=9
)
static_house_filtered$in.clothes_dryer <-
as.numeric(in_clothes_dryer_mapping[static_house_filtered$in.clothes_dryer])

# Ordinal coding for in.clothes_washer
in_clothes_washer_mapping <- c(
  "Standard, 100% Usage"=5, "EnergyStar, 100% Usage"=2, "Standard, 80% Usage"=4,
  "EnergyStar, 80% Usage"=1, "Standard, 120% Usage"=6, "EnergyStar, 120% Usage"=3,
  "None"=0
)
static_house_filtered$in.clothes_washer <-
as.numeric(in_clothes_washer_mapping[static_house_filtered$in.clothes_washer])

# Ordinal coding for in.cooking_range
in_cooking_range_mapping <- c(
  "Electric, 100% Usage"=2, "Gas, 80% Usage"=4, "Electric, 80% Usage"=1,
```

```
  "Gas, 120% Usage"=6, "Electric, 120% Usage"=3, "Gas, 100% Usage"=5,
  "Propane, 80% Usage"=7, "Propane, 100% Usage"=8, "Propane, 120% Usage"=9,
  "None"=0
)
static_house_filtered$in.cooking_range <-
as.numeric(in_cooking_range_mapping[static_house_filtered$in.cooking_range])


# Ordinal coding for in.cooling_setpoint (assuming the given order)
in_cooling_setpoint_mapping <- c(
  "72F"=7, "76F"=9, "70F"=6, "60F"=1, "78F"=10, "75F"=8, "68F"=5, "62F"=2, "65F"=3,
  "80F"=11, "67F"=4
)
static_house_filtered$in.cooling_setpoint <-
as.numeric(in_cooling_setpoint_mapping[static_house_filtered$in.cooling_setpoint])


# Ordinal coding for in.cooling_setpoint_offset_magnitude (assuming the given order)
in_cooling_setpoint_offset_magnitude_mapping <- c("0F"=0, "2F"=1, "5F"=2, "9F"=3)
static_house_filtered$in.cooling_setpoint_offset_magnitude <-
as.numeric(in_cooling_setpoint_offset_magnitude_mapping[static_house_filtered$in.cooling_setpoint_
offset_magnitude])


# Ordinal coding for in.dishwasher
in_dishwasher_mapping <- c(
  "None"=0, "290 Rated kWh, 100% Usage"=1, "318 Rated kWh, 80% Usage"=2,
  "318 Rated kWh, 120% Usage"=3, "290 Rated kWh, 80% Usage"=4,
  "290 Rated kWh, 120% Usage"=5, "318 Rated kWh, 100% Usage"=6
)
static_house_filtered$in.dishwasher <-
as.numeric(in_dishwasher_mapping[static_house_filtered$in.dishwasher])
```

```
# Ordinal coding for in.federal_poverty_level

in_federal_poverty_level_mapping <- c(

  "0-100%"=1, "150-200%"=2, "100-150%"=3, "400%+"=6, "200-300%"=4, "300-400%"=5

)

static_house_filtered$in.federal_poverty_level <-
as.numeric(in_federal_poverty_level_mapping[static_house_filtered$in.federal_poverty_level])


# Ordinal coding for in.geometry_attic_type

in_geometry_attic_type_mapping <- c("Vented Attic"=1, "Finished Attic or Cathedral Ceilings"=2)

static_house_filtered$in.geometry_attic_type <-
as.numeric(in_geometry_attic_type_mapping[static_house_filtered$in.geometry_attic_type])


# Ordinal coding for in.geometry_floor_area

in_geometry_floor_area_mapping <- c(

  "0-499"=0 ,"500-749"=1,"750-999"=2,"1000-1499"=3,"1500-1999"=4,"2000-2499"=5,"2500-
2999"=6,"3000-3999"=7,"4000+"=8

)

static_house_filtered$in.geometry_floor_area <-
as.numeric(in_geometry_floor_area_mapping[static_house_filtered$in.geometry_floor_area])


# Ordinal coding for in.geometry_floor_area_bin

in_geometry_floor_area_bin_mapping <- c("0-1499"=1, "1500-2499"=2, "2500-3999"=3, "4000+"=4)

static_house_filtered$in.geometry_floor_area_bin <-
as.numeric(in_geometry_floor_area_bin_mapping[static_house_filtered$in.geometry_floor_area_bin])


# Ordinal coding for in.geometry_garage

in_geometry_garage_mapping <- c("1 Car"=1, "None"=0, "2 Car"=2, "3 Car"=3)

static_house_filtered$in.geometry_garage <-
as.numeric(in_geometry_garage_mapping[static_house_filtered$in.geometry_garage])


# Ordinal coding for in.heating_fuel
```

```
in_heating_fuel_mapping <- c(
  "Natural Gas"=1, "Electricity"=2, "Propane"=3, "Other Fuel"=4,
  "Fuel Oil"=5, "None"=0
)
static_house_filtered$in.heating_fuel <-
as.numeric(in_heating_fuel_mapping[static_house_filtered$in.heating_fuel])


# Ordinal coding for in.heating_setpoint
in_heating_setpoint_mapping <- c(
  "70F"=6, "65F"=3, "68F"=5, "72F"=7, "75F"=8, "76F"=9,
  "78F"=10, "67F"=4, "55F"=0, "60F"=1, "80F"=11, "62F"=2
)
static_house_filtered$in.heating_setpoint <-
as.numeric(in_heating_setpoint_mapping[static_house_filtered$in.heating_setpoint])


# Ordinal coding for in.heating_setpoint_offset_magnitude
in_heating_setpoint_offset_magnitude_mapping <- c("0F"=0, "3F"=1, "12F"=3, "6F"=2)
static_house_filtered$in.heating_setpoint_offset_magnitude <-
as.numeric(in_heating_setpoint_offset_magnitude_mapping[static_house_filtered$in.heating_setpoint_
offset_magnitude])


# Ordinal coding for in.hot_water_fixtures
in_hot_water_fixtures_mapping <- c("100% Usage"=2, "50% Usage"=1, "200% Usage"=3)
static_house_filtered$in.hot_water_fixtures <-
as.numeric(in_hot_water_fixtures_mapping[static_house_filtered$in.hot_water_fixtures])


# Ordinal coding for in.hvac_cooling_efficiency
in_hvac_cooling_efficiency_mapping <- c(
  "AC, SEER 15"=1, "AC, SEER 13"=2, "None"=0, "AC, SEER 10"=4,
  "Heat Pump"=5, "Room AC, EER 10.7"=6, "Room AC, EER 8.5"=7,
```

```
  "AC, SEER 8"=8, "Room AC, EER 9.8"=9, "Room AC, EER 12.0"=10
)
static_house_filtered$in.hvac_cooling_efficiency <-
as.numeric(in_hvac_cooling_efficiency_mapping[static_house_filtered$in.hvac_cooling_efficiency])


# Ordinal coding for in.hvac_cooling_partial_space_conditioning
in_hvac_cooling_partial_space_conditioning_mapping <- c(
  "100% Conditioned"=5, "None"=0, "80% Conditioned"=4,
  "60% Conditioned"=3, "20% Conditioned"=1, "40% Conditioned"=2
)
static_house_filtered$in.hvac_cooling_partial_space_conditioning <-
as.numeric(in_hvac_cooling_partial_space_conditioning_mapping[static_house_filtered$in.hvac_cooling
_partial_space_conditioning])


# Ordinal coding for in.hvac_cooling_type
in_hvac_cooling_type_mapping <- c("Central AC"=1, "None"=0, "Heat Pump"=2, "Room AC"=3)
static_house_filtered$in.hvac_cooling_type <-
as.numeric(in_hvac_cooling_type_mapping[static_house_filtered$in.hvac_cooling_type])


# Ordinal coding for in.hvac_has_ducts
in_hvac_has_ducts_mapping <- c("Yes"=1, "No"=0)
static_house_filtered$in.hvac_has_ducts <-
as.numeric(in_hvac_has_ducts_mapping[static_house_filtered$in.hvac_has_ducts])


# Ordinal coding for in.hvac_has_zonal_electric_heating
in_hvac_has_zonal_electric_heating_mapping <- c("No"=0, "Yes"=1)
static_house_filtered$in.hvac_has_zonal_electric_heating <-
as.numeric(in_hvac_has_zonal_electric_heating_mapping[static_house_filtered$in.hvac_has_zonal_elec
tric_heating])


# Ordinal coding for in.hvac_heating_efficiency
```

```
# Ordinal coding for in_hvac_heating_efficiency




# Ordinal coding for in.hvac_heating_type

in_hvac_heating_type_mapping <- c("Ducted Heating"=1, "Non-Ducted Heating"=2, "Ducted Heat Pump"=3, "None"=0)

static_house_filtered$in.hvac_heating_type <- as.numeric(in_hvac_heating_type_mapping[static_house_filtered$in.hvac_heating_type])

# Ordinal coding for in.hvac_heating_type_and_fuel

in_hvac_heating_type_and_fuel_mapping <- c(

  "Natural Gas Fuel Furnace"=1, "Natural Gas Fuel Boiler"=2, "Electricity Electric Furnace"=3,

  "Electricity ASHP"=4, "Electricity Baseboard"=5, "Propane Fuel Furnace"=6,

  "None"=7, "Propane Fuel Boiler"=8, "Natural Gas Fuel Wall/Floor Furnace"=9,

  "Fuel Oil Fuel Furnace"=10, "Propane Fuel Wall/Floor Furnace"=11, "Electricity Electric Boiler"=12,

  "Fuel Oil Fuel Boiler"=13, "Electricity Electric Wall Furnace"=14, "Fuel Oil Fuel Wall/Floor Furnace"=15

)


static_house_filtered$in.hvac_heating_type_and_fuel <- as.numeric(in_hvac_heating_type_and_fuel_mapping[as.character(static_house_filtered$in.hvac_heating_type_and_fuel)])




# Ordinal coding for in.income

# Updated Ordinal coding for in.income using maximum value of the range

income_mapping <- c(

  "<10000"=9999, "10000-14999"=14999, "15000-19999"=19999, "20000-24999"=24999,

  "25000-29999"=29999, "30000-34999"=34999, "35000-39999"=39999, "40000-44999"=44999,

  "45000-49999"=49999, "50000-59999"=59999, "60000-69999"=69999, "70000-79999"=79999,

  "80000-99999"=99999, "100000-119999"=119999, "120000-139999"=139999,
```

```
  "140000-159999"=159999, "160000-179999"=179999, "180000-199999"=199999, "200000+"=200000
)

static_house_filtered$in.income <- as.numeric(income_mapping[static_house_filtered$in.income])

# Ordinal coding for in.insulation_ceiling
in_insulation_ceiling_mapping <- c(
  "R-30"=4, "R-13"=2, "R-38"=5, "R-19"=3, "R-7"=1, "Uninsulated"=7, "None"=0, "R-49"=6
)
static_house_filtered$in.insulation_ceiling <-
as.numeric(in_insulation_ceiling_mapping[static_house_filtered$in.insulation_ceiling])

# Ordinal coding for in.insulation_floor
in_insulation_floor_mapping <- c("None"=0, "Uninsulated"=1, "Ceiling R-13"=2, "Ceiling R-19"=3)
static_house_filtered$in.insulation_floor <-
as.numeric(in_insulation_floor_mapping[static_house_filtered$in.insulation_floor])

# Ordinal coding for in.insulation_foundation_wall
in_insulation_foundation_wall_mapping <- c(
  "None"=0, "Uninsulated"=1, "Wall R-10, Exterior"=3, "Wall R-15, Exterior"=4, "Wall R-5, Exterior"=2
)
static_house_filtered$in.insulation_foundation_wall <-
as.numeric(in_insulation_foundation_wall_mapping[static_house_filtered$in.insulation_foundation_wal
l])

# Ordinal coding for in.insulation_rim_joist
in_insulation_rim_joist_mapping <- c("None"=0, "Uninsulated"=1, "R-10, Exterior"=3, "R-15, Exterior"=4,
"R-5, Exterior"=2)
static_house_filtered$in.insulation_rim_joist <-
as.numeric(in_insulation_rim_joist_mapping[static_house_filtered$in.insulation_rim_joist])
```

```
# Ordinal coding for in.insulation_roof
in_insulation_roof_mapping <- c(
  "Unfinished, Uninsulated"=1, "Finished, R-30"=2, "Finished, R-38"=3,
  "Finished, R-19"=4, "Finished, R-13"=5, "Finished, R-49"=6, "Finished, R-7"=7, "Finished, Uninsulated"=8
)
static_house_filtered$in.insulation_roof <-
as.numeric(in_insulation_roof_mapping[static_house_filtered$in.insulation_roof])


# Ordinal coding for in.insulation_slab
in_insulation_slab_mapping <- c(
  "Uninsulated"=1, "None"=2, "2ft R10 Under, Horizontal"=3,
  "2ft R5 Under, Horizontal"=4, "2ft R5 Perimeter, Vertical"=5, "2ft R10 Perimeter, Vertical"=6
)
static_house_filtered$in.insulation_slab <-
as.numeric(in_insulation_slab_mapping[static_house_filtered$in.insulation_slab])


# # Ordinal coding for in.insulation_wall
# in_insulation_wall_mapping <- c(
#   "Wood Stud, Uninsulated"=1, "Wood Stud, R-11"=2, "Brick, 12-in, 3-wythe, R-11"=3,
#   "CMU, 6-in Hollow, R-7"=4, "Wood Stud, R-15"=5, "Wood Stud, R-7"=6,
#   "CMU, 6-in Hollow, R-11"=7, "CMU, 6-in Hollow, Uninsulated"=8,
#   "Brick, 12-in, 3-wythe, Uninsulated"=9, "Brick, 12-in, 3-wythe, R-7"=10,
#   "Wood Stud, R-19"=11, "Brick, 12-in, 3-wythe, R-15"=12, "CMU, 6-in Hollow, R-15"=13,
#   "Brick, 12-in, 3-wythe, R-19"=14, "CMU, 6-in Hollow, R-19"=15
# )
# static_house_filtered$in.insulation_wall <-
as.numeric(in_insulation_wall_mapping[static_house_filtered$in.insulation_wall])


# Ordinal coding for in.lighting
in_lighting_mapping <- c("100% Incandescent"=1, "100% LED"=2, "100% CFL"=3)
```

```
static_house_filtered$in.lighting <- as.numeric(in_lighting_mapping[static_house_filtered$in.lighting])


# Ordinal coding for in.misc_extra_refrigerator

in_misc_extra_refrigerator_mapping <- c("EF 17.6"=5, "None"=0, "EF 15.9"=4, "EF 19.9"=6, "EF 6.7"=1, "EF 10.5"=2, "EF 10.2"=3)

static_house_filtered$in.misc_extra_refrigerator <- as.numeric(in_misc_extra_refrigerator_mapping[static_house_filtered$in.misc_extra_refrigerator])


# Ordinal coding for in.misc_freezer

in_misc_freezer_mapping <- c("EF 12, National Average"=1, "None"=0)

static_house_filtered$in.misc_freezer <- as.numeric(in_misc_freezer_mapping[static_house_filtered$in.misc_freezer])


# Ordinal coding for in.misc_gas_fireplace

in_misc_gas_fireplace_mapping <- c("None"=0, "Gas Fireplace"=2)

static_house_filtered$in.misc_gas_fireplace <- as.numeric(in_misc_gas_fireplace_mapping[static_house_filtered$in.misc_gas_fireplace])


# Ordinal coding for in.misc_gas_grill

in_misc_gas_grill_mapping <- c("None"=0, "Gas Grill"=2)

static_house_filtered$in.misc_gas_grill <- as.numeric(in_misc_gas_grill_mapping[static_house_filtered$in.misc_gas_grill])


# Ordinal coding for in.misc_gas_lighting

in_misc_gas_lighting_mapping <- c("None"=0, "Gas Lighting"=2)

static_house_filtered$in.misc_gas_lighting <- as.numeric(in_misc_gas_lighting_mapping[static_house_filtered$in.misc_gas_lighting])


# Ordinal coding for in.misc_hot_tub_spa

in_misc_hot_tub_spa_mapping <- c("None"=0, "Gas"=2, "Electric"=1)
```

```
static_house_filtered$in.misc_hot_tub_spa <-
as.numeric(in_misc_hot_tub_spa_mapping[static_house_filtered$in.misc_hot_tub_spa])


# Ordinal coding for in.misc_pool

in_misc_pool_mapping <- c("None"=0, "Has Pool"=1)

static_house_filtered$in.misc_pool <-
as.numeric(in_misc_pool_mapping[static_house_filtered$in.misc_pool])


# Ordinal coding for in.misc_pool_heater

in_misc_pool_heater_mapping <- c("None"=0, "Electric"=1, "Solar"=3, "Gas"=2)

static_house_filtered$in.misc_pool_heater <-
as.numeric(in_misc_pool_heater_mapping[static_house_filtered$in.misc_pool_heater])


# Ordinal coding for in.misc_pool_pump

in_misc_pool_pump_mapping <- c("None"=0, "1.0 HP Pump"=1)

static_house_filtered$in.misc_pool_pump <-
as.numeric(in_misc_pool_pump_mapping[static_house_filtered$in.misc_pool_pump])


# Ordinal coding for in.misc_well_pump

in_misc_well_pump_mapping <- c("None"=0, "Typical Efficiency"=1)

static_house_filtered$in.misc_well_pump <-
as.numeric(in_misc_well_pump_mapping[static_house_filtered$in.misc_well_pump])


# Ordinal coding for in.natural_ventilation

in_natural_ventilation_mapping <- c("Cooling Season, 7 days/wk"=1)

static_house_filtered$in.natural_ventilation <-
as.numeric(in_natural_ventilation_mapping[static_house_filtered$in.natural_ventilation])


# Ordinal coding for in.occupants

in_occupants_mapping <- c("3"=3, "1"=1, "2"=2, "4"=4, "5"=5, "8"=8, "6"=6, "7"=7, "10+"=10, "9"=9)
```

```
static_house_filtered$in.occupants <-
as.numeric(in_occupants_mapping[static_house_filtered$in.occupants])


# Ordinal coding for in.orientation

in_orientation_mapping <- c("North"=1, "West"=2, "South"=3, "Northeast"=4, "Northwest"=5,
"Southeast"=6, "East"=7, "Southwest"=8)

static_house_filtered$in.orientation <-
as.numeric(in_orientation_mapping[static_house_filtered$in.orientation])


# Ordinal coding for in.plug_load_diversity

in_plug_load_diversity_mapping <- c("100%"=1, "50%"=0, "200%"=2)

static_house_filtered$in.plug_load_diversity <-
as.numeric(in_plug_load_diversity_mapping[static_house_filtered$in.plug_load_diversity])


# Ordinal coding for in.refrigerator

in_refrigerator_mapping <- c("EF 6.7, 100% Usage"=1, "EF 17.6, 100% Usage"=2, "EF 19.9, 100%
Usage"=3, "EF 10.5, 100% Usage"=4, "EF 15.9, 100% Usage"=5, "EF 10.2, 100% Usage"=6, "None"=0)

static_house_filtered$in.refrigerator <-
as.numeric(in_refrigerator_mapping[static_house_filtered$in.refrigerator])


# Ordinal coding for in.roof_material

in_roof_material_mapping <- c("Composition Shingles"=1, "Metal, Dark"=2, "Tile, Concrete"=3, "Wood
Shingles"=4, "Tile, Clay or Ceramic"=5, "Slate"=6, "Asphalt Shingles, Medium"=7)

static_house_filtered$in.roof_material <-
as.numeric(in_roof_material_mapping[static_house_filtered$in.roof_material])


# Ordinal coding for in.tenure

in_tenure_mapping <- c("Renter"=1, "Owner"=2)

static_house_filtered$in.tenure <- as.numeric(in_tenure_mapping[static_house_filtered$in.tenure])


# Ordinal coding for in.usage_level

in_usage_level_mapping <- c("Medium"=2, "Low"=1, "High"=3)
```

```
static_house_filtered$in.usage_level <-
as.numeric(in_usage_level_mapping[static_house_filtered$in.usage_level])


# Ordinal coding for in.vacancy_status

in_vacancy_status_mapping <- c("Occupied"=1, "Vacant"=2)

static_house_filtered$in.vacancy_status <-
as.numeric(in_vacancy_status_mapping[static_house_filtered$in.vacancy_status])


# Ordinal coding for in.vintage

in_vintage_mapping <- c("1950s"=1, "2000s"=2, "<1940"=3, "1980s"=4, "1990s"=5, "1970s"=6,
"1960s"=7, "2010s"=8, "1940s"=9)

static_house_filtered$in.vintage <- as.numeric(in_vintage_mapping[static_house_filtered$in.vintage])


# Ordinal coding for in.vintage_acs

in_vintage_acs_mapping <- c("1940-59"=1, "2000-09"=2, "<1940"=3, "1980-99"=4, "1960-79"=5,
"2010s"=6)

static_house_filtered$in.vintage_acs <-
as.numeric(in_vintage_acs_mapping[static_house_filtered$in.vintage_acs])


# Ordinal coding for in.water_heater_efficiency

in_water_heater_efficiency_mapping <- c("Natural Gas Standard"=1, "Electric Standard"=2, "Natural Gas
Premium"=3, "Propane Tankless"=4, "Propane Standard"=5, "Electric Premium"=6, "Other Fuel"=7,
"Propane Premium"=8, "Electric Tankless"=9, "Electric Heat Pump, 80 gal"=10, "Natural Gas
Tankless"=11, "Fuel Oil Standard"=12)

static_house_filtered$in.water_heater_efficiency <-
as.numeric(in_water_heater_efficiency_mapping[static_house_filtered$in.water_heater_efficiency])


# Ordinal coding for in.water_heater_fuel

in_water_heater_fuel_mapping <- c("Natural Gas"=1, "Electricity"=2, "Propane"=3, "Other Fuel"=4, "Fuel
Oil"=5)

static_house_filtered$in.water_heater_fuel <-
as.numeric(in_water_heater_fuel_mapping[static_house_filtered$in.water_heater_fuel])
```

```
# Ordinal coding for in.weather_file_city

in_weather_file_city_mapping <- c("Rock Hill York Co"=1, "Oconee Co Rgnl"=2, "Columbia Metro"=3, "Columbia Owens Apt"=4, "Greenville Greenvil"=5, "Charleston Muni"=6, "Myrtle Beach Civ"=7, "Anderson Rgnl"=8, "Florence Rgnl"=9, "Orangeburg Muni"=10, "Beaufort Mcas"=11, "Shaw Afb Sumter"=12, "Greenwood Co"=13, "Augusta Bush Field"=14, "Monroe Airport"=15, "Rutherfordton"=16, "Maxton"=17, "Daniel Field"=18)

static_house_filtered$in.weather_file_city <- as.numeric(in_weather_file_city_mapping[static_house_filtered$in.weather_file_city])


# Ordinal coding for in.weather_file_latitude

static_house_filtered$in.weather_file_latitude <- as.numeric(static_house_filtered$in.weather_file_latitude)


# Ordinal coding for in.weather_file_longitude

static_house_filtered$in.weather_file_longitude <- as.numeric(static_house_filtered$in.weather_file_longitude)


# Ordinal coding for in.window_areas

in_window_areas_mapping <- c("F12 B12 L12 R12"=1, "F18 B18 L18 R18"=2, "F9 B9 L9 R9"=3, "F15 B15 L15 R15"=4, "F30 B30 L30 R30"=5, "F6 B6 L6 R6"=6)

static_house_filtered$in.window_areas <- as.numeric(in_window_areas_mapping[static_house_filtered$in.window_areas])


# Ordinal coding for in.windows

in_windows_mapping <- c("Double, Low-E, Non-metal, Air, M-Gain"=1, "Single, Clear, Non-metal"=2, "Double, Clear, Metal, Air"=3, "Single, Clear, Non-metal, Exterior Clear Storm"=4, "Double, Clear, Non-metal, Air"=5, "Single, Clear, Metal"=6, "Double, Clear, Metal, Air, Exterior Clear Storm"=7, "Double, Clear, Non-metal, Air, Exterior Clear Storm"=8, "Triple, Low-E, Non-metal, Air, L-Gain"=9, "Single, Clear, Metal, Exterior Clear Storm"=10)

static_house_filtered$in.windows <- as.numeric(in_windows_mapping[static_house_filtered$in.windows])
```

```
# Ordinal coding for upgrade.hvac_cooling_efficiency

upgrade_hvac_cooling_efficiency_mapping <- c("Heat Pump"=1)

static_house_filtered$upgrade.hvac_cooling_efficiency <-
as.numeric(upgrade_hvac_cooling_efficiency_mapping[static_house_filtered$upgrade.hvac_cooling_effi
ciency])
```

```
# Convert to character if necessary

static_house_filtered$in.occupants <- as.character(static_house_filtered$in.occupants)
```

```
# Ordinal coding for in.occupants

in_occupants_mapping <- c("3"=1, "1"=2, "2"=3, "4"=4, "5"=5, "8"=6, "6"=7, "7"=8, "10+"=9, "9"=10)

static_house_filtered$in.occupants <-
as.numeric(in_occupants_mapping[static_house_filtered$in.occupants])
```

```
# Ordinal coding for upgrade.clothes_dryer

upgrade_clothes_dryer_mapping <- c("Electric, Premium, Heat Pump, Ventless, 100% Usage"=1,
"Electric, Premium, Heat Pump, Ventless, 80% Usage"=2, "Electric, Premium, Heat Pump, Ventless, 120%
Usage"=3 )

static_house_filtered$upgrade.clothes_dryer <-
as.numeric(upgrade_clothes_dryer_mapping[static_house_filtered$upgrade.clothes_dryer])
```

```
# Ordinal coding for upgrade.insulation_ceiling

upgrade_insulation_ceiling_mapping <- c("R-49"=1)

static_house_filtered$upgrade.insulation_ceiling <-
as.numeric(upgrade_insulation_ceiling_mapping[static_house_filtered$upgrade.insulation_ceiling])
```

```
# Ordinal coding for upgrade.hvac_heating_type

upgrade_hvac_heating_type_mapping <- c("Ducted Heat Pump"=1)

static_house_filtered$upgrade.hvac_heating_type <-
as.numeric(upgrade_hvac_heating_type_mapping[static_house_filtered$upgrade.hvac_heating_type])


# Ordinal coding for upgrade.insulation_wall

upgrade_insulation_wall_mapping <- c("Wood Stud, R-13"=1)

static_house_filtered$upgrade.insulation_wall <-
as.numeric(upgrade_insulation_wall_mapping[static_house_filtered$upgrade.insulation_wall])


# Ordinal coding for upgrade.insulation_foundation_wall

upgrade_insulation_foundation_wall_mapping <- c("Wall R-10, Interior"=2)

static_house_filtered$upgrade.insulation_foundation_wall <-
as.numeric(upgrade_insulation_foundation_wall_mapping[static_house_filtered$upgrade.insulation_fo
undation_wall])


# Ordinal coding for upgrade.hvac_heating_efficiency

upgrade_hvac_heating_efficiency_mapping <- c("MSHP, SEER 24, 13 HSPF"=1, "MSHP, SEER 29.3, 14
HSPF, Max Load"=2)

static_house_filtered$upgrade.hvac_heating_efficiency <-
as.numeric(upgrade_hvac_heating_efficiency_mapping[static_house_filtered$upgrade.hvac_heating_eff
iciency])


# Ordinal coding for upgrade.cooking_range

upgrade_cooking_range_mapping_2 <- c("Electric, Induction, 100% Usage"=1, "Electric, Induction, 80%
Usage"=2, "Electric, Induction, 120% Usage"=3)

static_house_filtered$upgrade.cooking_range <-
as.numeric(upgrade_cooking_range_mapping_2[static_house_filtered$upgrade.cooking_range])
```

```
# Ordinal coding for in.orientation

in_orientation_mapping <- c("North"=1, "West"=2, "South"=3, "Northeast"=4, "Northwest"=5,
"Southeast"=6, "East"=7, "Southwest"=8)

static_house_filtered$in.orientation <-
as.numeric(in_orientation_mapping[static_house_filtered$in.orientation])

# Ordinal coding for in.tenure

in_tenure_mapping <- c("Renter"=1, "Owner"=2)

static_house_filtered$in.tenure <- as.numeric(in_tenure_mapping[static_house_filtered$in.tenure])

# Ordinal coding for in.usage_level

in_usage_level_mapping <- c("Medium"=1, "Low"=2, "High"=3)

static_house_filtered$in.usage_level <-
as.numeric(in_usage_level_mapping[static_house_filtered$in.usage_level])

# Ordinal coding for in.vacancy_status

in_vacancy_status_mapping <- c("Occupied"=1, "Vacant"=2)

static_house_filtered$in.vacancy_status <-
as.numeric(in_vacancy_status_mapping[static_house_filtered$in.vacancy_status])

# Ordinal coding for in.vintage

in_vintage_mapping <- c("1950s"=1, "2000s"=2, "<1940"=3, "1980s"=4, "1990s"=5, "1970s"=6,
"1960s"=7, "2010s"=8, "1940s"=9)

static_house_filtered$in.vintage <- as.numeric(in_vintage_mapping[static_house_filtered$in.vintage])

# Ordinal coding for in.vintage_acs

in_vintage_acs_mapping <- c("1940-59"=1, "2000-09"=2, "<1940"=3, "1980-99"=4, "1960-79"=5,
"2010s"=6)
```

```
static_house_filtered$in.vintage_acs <-
as.numeric(in_vintage_acs_mapping[static_house_filtered$in.vintage_acs])


# Ordinal coding for in.water_heater_efficiency

in_water_heater_efficiency_mapping <- c("Natural Gas Standard"=1, "Electric Standard"=2, "Natural Gas Premium"=3, "Propane Tankless"=4, "Propane Standard"=5, "Electric Premium"=6, "Other Fuel"=7, "Propane Premium"=8, "Electric Tankless"=9, "Electric Heat Pump, 80 gal"=10, "Natural Gas Tankless"=11, "Fuel Oil Standard"=12)

static_house_filtered$in.water_heater_efficiency <-
as.numeric(in_water_heater_efficiency_mapping[static_house_filtered$in.water_heater_efficiency])


# Ordinal coding for in.water_heater_fuel

in_water_heater_fuel_mapping <- c("Natural Gas"=1, "Electricity"=2, "Propane"=3, "Other Fuel"=4, "Fuel Oil"=5)

static_house_filtered$in.water_heater_fuel <-
as.numeric(in_water_heater_fuel_mapping[static_house_filtered$in.water_heater_fuel])


# Ordinal coding for in.weather_file_city

in_weather_file_city_mapping <- c("Rock Hill York Co"=1, "Oconee Co Rgnl"=2, "Columbia Metro"=3, "Columbia Owens Apt"=4, "Greenville Greenvil"=5, "Charleston Muni"=6, "Myrtle Beach Civ"=7, "Anderson Rgnl"=8, "Florence Rgnl"=9, "Orangeburg Muni"=10, "Beaufort Mcas"=11, "Shaw Afb Sumter"=12, "Greenwood Co"=13, "Augusta Bush Field"=14, "Monroe Airport"=15, "Rutherfordton"=16, "Maxton"=17, "Daniel Field"=18)

static_house_filtered$in.weather_file_city <-
as.numeric(in_weather_file_city_mapping[static_house_filtered$in.weather_file_city])


# Ordinal coding for in.weather_file_latitude

static_house_filtered$in.weather_file_latitude <-
as.numeric(static_house_filtered$in.weather_file_latitude)


# Ordinal coding for in.weather_file_longitude

static_house_filtered$in.weather_file_longitude <-
as.numeric(static_house_filtered$in.weather_file_longitude)
```

```
# Ordinal coding for in.window_areas

in_window_areas_mapping <- c("F12 B12 L12 R12"=1, "F18 B18 L18 R18"=2, "F9 B9 L9 R9"=3, "F15 B15 L15 R15"=4, "F30 B30 L30 R30"=5, "F6 B6 L6 R6"=6)

static_house_filtered$in.window_areas <- as.numeric(in_window_areas_mapping[static_house_filtered$in.window_areas])



# Ordinal coding for upgrade.insulation_roof

upgrade_insulation_roof_mapping <- c("Finished, R-30"=1)

static_house_filtered$upgrade.insulation_roof <- as.numeric(upgrade_insulation_roof_mapping[static_house_filtered$upgrade.insulation_roof])


# Ordinal coding for upgrade.water_heater_efficiency

upgrade_water_heater_efficiency_mapping <- c("Electric Heat Pump, 50 gal, 3.45 UEF"=1, "Electric Heat Pump, 66 gal, 3.35 UEF"=2, "Electric Heat Pump, 80 gal, 3.45 UEF"=3)

static_house_filtered$upgrade.water_heater_efficiency <- as.numeric(upgrade_water_heater_efficiency_mapping[static_house_filtered$upgrade.water_heater_efficiency])


# Ordinal coding for upgrade.hvac_cooling_efficiency

upgrade_hvac_cooling_efficiency_mapping <- c("Heat Pump"=1)

static_house_filtered$upgrade.hvac_cooling_efficiency <- as.numeric(upgrade_hvac_cooling_efficiency_mapping[static_house_filtered$upgrade.hvac_cooling_efficiency])


# Ordinal coding for upgrade.infiltration_reduction

upgrade_infiltration_reduction_mapping <- c("30%"=1)

static_house_filtered$upgrade.infiltration_reduction <- as.numeric(upgrade_infiltration_reduction_mapping[static_house_filtered$upgrade.infiltration_reduction])


# Ordinal coding for upgrade.geometry_foundation_type
```

upgrade_geometry_foundation_type_mapping <- c("Unvented Crawlspace"=1)

static_house_filtered$upgrade.geometry_foundation_type <- as.numeric(upgrade_geometry_foundation_type_mapping[static_house_filtered$upgrade.geometry_foundation_type])

```

```{r}

static_house_ordinal <- static_house_filtered

static_house_ordinal_copy <- static_house_ordinal

static_house_ordinal_copy_2 <- static_house_ordinal

```

```{r}

setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(static_house_ordinal, file ="raw_uncleaned_static_ordinal",row.names=FALSE)

```

```{r}

#Remove NA Rows or Replace NA rows

handle_missing_values <- function(df) {

  for (col in names(df)) {

    if (any(is.na(df[[col]]))) {

      # Check if column has missing values

      if (is.numeric(df[[col]])) {

        # For numeric columns, replace missing values with mean

```
      df[[col]][is.na(df[[col]])] <- mean(df[[col]], na.rm = TRUE)
    } else {
      # For non-numeric columns, replace missing values with mode
      mode_val <- as.character(which.max(table(df[[col]])))
      df[[col]][is.na(df[[col]])] <- mode_val
    }
  }
 }
 return(df)
}

# Applying the function to your dataframe
static_house_ordinal_copy <- handle_missing_values(static_house_ordinal_copy)
```

```{r}
unique_values <- sapply(static_house_ordinal_copy, unique)
print(unique_values)
```

```{r}
fill_mode_non_numeric <- function(x) {
 if (is.character(x) || is.factor(x)) {
  mode_val <- as.character(which.max(table(x)))
  x[is.na(x)] <- mode_val
 }
 return(x)
}
```

Applying the function to your dataframe

static_house_ordinal_copy <- lapply(static_house_ordinal_copy, fill_mode_non_numeric)

```

```{r}

static_house_ordinal_copy <- static_house_ordinal_copy_2

```

```{r}

str(static_house_ordinal_copy)

```

```{r}

selected_columns <- static_house_ordinal_copy[, !(names(static_house_ordinal_copy) %in% c(

  "upgrade.insulation_wall",

  "upgrade.insulation_foundation_wall",

  "upgrade.hvac_heating_type",

  "upgrade.insulation_ceiling",

  "upgrade.hvac_cooling_efficiency",

  "upgrade.infiltration_reduction",

  "upgrade.geometry_foundation_type",

  "upgrade.insulation_roof"

))]

```

```{r}
new_dataframe <- data.frame(selected_columns)

static_house_ordinal_clear <- new_dataframe

```

```{r}
static_house_ordinal_clear$upgrade.cooking_range <-
ifelse(static_house_ordinal_clear$upgrade.cooking_range == "", 0,
static_house_ordinal_clear$upgrade.cooking_range)

static_house_ordinal_clear$upgrade.clothes_dryer <-
ifelse(static_house_ordinal_clear$upgrade.clothes_dryer == "", 0,
static_house_ordinal_clear$upgrade.clothes_dryer)

static_house_ordinal_clear$upgrade.water_heater_efficiency <-
ifelse(static_house_ordinal_clear$upgrade.water_heater_efficiency == "", 0,
static_house_ordinal_clear$upgrade.water_heater_efficiency)

```

```{r}

unique_values <- sapply(static_house_ordinal_clear, unique)

```
print(unique_values)
```

```{r}
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(static_house_ordinal_clear, file ="SHOrdinalFinalModelFIle.csv",row.names=FALSE)
```

```{r}
#Model_Merge_SH <- static_house_ordinal_clear
# Assuming Model_Merge_SH and energy_result_df are your data frames
# and both have a column named 'bldg_id'

# Merge the data frames on the 'bldg_id' column
SH_Energy_Model_Ordinal <- merge(Model_Merge_SH, energy_result_df, by='bldg_id')

# Print or view the merged data frame
print(SH_Energy_Model_Ordinal)

```

```{r}
# Assuming SH_Energy_Model_Ordinal and weather_combined are your data frames
# and they have columns 'in.county' and 'county_id' respectively

# Merge the data frames on the specified columns
```

Final_Merged_Ordinal_Model<- merge(SH_Energy_Model_Ordinal, weather_combined, by.x=c('in.county', 'time_split'), by.y=c('county_id', 'time_split'))

Print or view the merged data frame

print(Final_Merged_Ordinal_Model)

```

```{r}

# Assuming Final_Merged_Ordinal_Model is your merged data frame

# Sort the data frame based on 'bldg_id' and 'time_range.x'

Final_team2_Modelling_SEW<-
Final_Merged_Ordinal_Model[order(Final_Merged_Ordinal_Model$bldg_id, Final_Merged_Ordinal_Model$time_range.x), ]

# Print or view the sorted data frame

print(Final_team2_Modelling_SEW)

```

```{r}

setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")

write.csv(Final_team2_Modelling_SEW, file
="Team2_Final_SEW_Ordinal_Modelling1.csv",row.names=FALSE)

```

```{r}

# Assuming your dataframe is named Final_team2_Modelling_SEW

# Select numeric columns only

numeric_columns <- Final_team2_Modelling_SEW[sapply(Final_team2_Modelling_SEW, is.numeric)]

# Calculate correlation matrix

correlation_matrix <- cor(numeric_columns)

# Print the correlation matrix

print(correlation_matrix)

```

```{r}

# Install corrplot package if not installed

# install.packages("corrplot")

# Load the corrplot package

library(corrplot)

# Assuming your dataframe is named Final_team2_Modelling_SEW

```
# Select numeric columns only
numeric_columns <- Final_team2_Modelling_SEW[sapply(Final_team2_Modelling_SEW, is.numeric)]

# Remove rows with missing values
numeric_columns <- na.omit(numeric_columns)

# Calculate correlation matrix
correlation_matrix <- cor(numeric_columns)
# Set a threshold for correlation values
threshold <- 0.4

# Filter correlation matrix for values above the threshold
filtered_correlation_matrix <- correlation_matrix * (abs(correlation_matrix) > threshold)

Plot the correlation matrix using corrplot without hierarchical clustering
corrplot(
  filtered_correlation_matrix,
  method = "color",   # Color-coded plot
  type = "upper",   # Show only upper triangle to avoid redundancy
  tl.cex = 0.7,   # Text label size
  tl.col = "black",   # Text label color
  col = colorRampPalette(c("#FFFFFF", "#009688", "#004D40"))(100), # Green color palette
  addCoef.col = "black",   # Coefficient color
  number.cex = 0.6   # Correlation coefficient text size
)
```

```

```{r}
# Visualize a subset of the correlation matrix
sub_cor_matrix <- correlation_matrix[1:10, 1:10]  # Adjust the range as needed

# Plot the subset
corrplot(sub_cor_matrix, method = "color")

```


```{r}
# Save correlation matrix to a CSV file
setwd("C:/Users/Soundarya Ravi/Desktop/Shiny")
write.csv(correlation_matrix, file = "correlation_matrix1.csv", row.names=FALSE)

```


```{r}

# Assuming your dataframe is named Final_team2_Modelling_SEW

# Select specific columns for correlation
selected_columns <- Final_team2_Modelling_SEW[, c(
  "time_split_numeric",
  "in.sqft",
  "in.bedrooms",
  "in.clothes_dryer",
  "in.clothes_washer",

"in.cooling_setpoint",

"in.federal_poverty_level",

"in.geometry_floor_area",

"in.geometry_floor_area_bin",

"in.geometry_garage",

"in.heating_setpoint",

"in.hot_water_fixtures",

"in.income",

"in.misc_hot_tub_spa",

"in.misc_pool",

"in.misc_hot_tub_spa",

"in.misc_pool",

"in.misc_pool_heater",

"in.misc_pool_pump",

"in.occupants",

"in.plug_load_diversity",

"in.usage_level",

"upgrade.water_heater_efficiency",

"upgrade.clothes_dryer",

"upgrade.cooking_range",

"out.kitchen_energy_consumption",

"out.laundry_energy_consumption",

"out.heating_cooling_energy_consumption",

"out.water_heating_energy_consumption",

"out.electrical_appliances_energy_consumption",

"out.outdoor_appliances_energy_consumption",

"Dry_Bulb_Temperature_C",

"Wind_Speed_m_s",

"Wind_Direction_Deg",

```
  "Diffuse_Horizontal_Radiation_W_m2"
)]

# Impute missing values with the mean of each column
selected_columns <- apply(selected_columns, 2, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x))

# Calculate correlation matrix for the selected columns
correlation_matrix <- cor(selected_columns)

# Create heatmap
heatmap(correlation_matrix)
```

```{r}
# Assuming your dataframe is named Final_team2_Modelling_SEW

# Create a numeric mapping for time_split
time_split_mapping <- c(
  "Late Night" = 1,
  "Early Morning" = 2,
  "Morning" = 3,
  "Noon" = 4,
  "Evening" = 5,
  "Night" = 6
)

# Convert time_split to numeric using the mapping
```

```
Final_team2_Modelling_SEW$time_split_numeric <-
as.numeric(factor(Final_team2_Modelling_SEW$time_split, levels = names(time_split_mapping), labels
= time_split_mapping))


# Display the updated dataframe

head(Final_team2_Modelling_SEW)


```

```{r}

threshold <- 0.5


# Filter correlation matrix for values above the threshold

filtered_correlation_matrix <- correlation_matrix * (abs(correlation_matrix) > threshold)



# Plot the correlation matrix using corrplot without hierarchical clustering

corrplot(
  filtered_correlation_matrix,
  method = "color",           # Color-coded plot
  type = "upper",             # Show only upper triangle to avoid redundancy
  tl.cex = 0.7,               # Text label size
  tl.col = "black",           # Text label color
  col = colorRampPalette(c("#FFFFFF", "#009688", "#004D40"))(100),  # Green color palette
  addCoef.col = "black",       # Coefficient color
  number.cex = 0.6            # Correlation coefficient text size
)
```


```{r}
```

```
# Install network package if not installed
# install.packages("network")

# Load the network package
library(network)

# Create a network object
net <- as.network(filtered_correlation_matrix)

# Plot the network
plot(net, displaylabels = TRUE, boxed.labels = TRUE, label.cex = 0.7)
```

EXPLORATORY DATA ANALYSIS

R Notebook

Code

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Ctrl+Shift+Enter*.

Hide

```
install.packages("corrplot")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/corrplot_0.92.zip'
Content type 'application/zip' length 3844830 bytes (3.7 MB)
downloaded 3.7 MB
```

```
package ‘corrplot’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("tidyverse")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/tidyverse_2.0.0.zip'
Content type 'application/zip' length 430853 bytes (420 KB)
downloaded 420 KB
```

```
package ‘tidyverse’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("arrow")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/arrow_14.0.0.2.zip'
Content type 'application/zip' length 17774853 bytes (17.0 MB)
downloaded 17.0 MB
```

```
package ‘arrow’ successfully unpacked and MD5 sums checked
Warning in install.packages :
  cannot remove prior installation of package ‘arrow’
Warning in install.packages :
  problem copying C:\Users\hp\AppData\Local\R\win-library\4.3\00LOCK\arrow\libs\x64\arrow.dll
to C:\Users\hp\AppData\Local\R\win-library\4.3\arrow\libs\x64\arrow.dll: Permission denied
Warning in install.packages :
  restored ‘arrow’

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("maps")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/maps_3.4.1.1.zip'
Content type 'application/zip' length 3098701 bytes (3.0 MB)
downloaded 3.0 MB
```

```
package ‘maps’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("reshape")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/reshape_0.8.9.zip'
Content type 'application/zip' length 170260 bytes (166 KB)
downloaded 166 KB
```

```
package ‘reshape’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("reshape2")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/reshape2_1.4.4.zip'
Content type 'application/zip' length 455913 bytes (445 KB)
downloaded 445 KB
```

```
package ‘reshape2’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\Rtmps1sZLe\downloaded_packages
```

Hide

```
install.packages("ggplot2")
```

```
Error in install.packages : Updating loaded packages
```

Hide

```
install.packages("factoextra")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/factoextra_1.0.7.zip'
Content type 'application/zip' length 415631 bytes (405 KB)
```

```

Restarting R session...
```

Hide

```
library(tidyverse)
library(arrow)
data <- read_csv("C:/Users/hp/Desktop/IDS Project/Final_Merged_For_Shiny_Geo.csv")
```

```
Rows: 34260 Columns: 96── Column specification ────────────────────────────
Delimiter: ","
chr (70): in.county, time_split, in.county_and_puma...
dbl (26): bldg_id, time_split_numeric, in.sqft, in....
ℹ Use `spec()` to retrieve the full column specification for this data.
ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

Hide

```
install.packages("GGally")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
```

```
Warning in install.packages :
  package ‘GGally’ is in use and will not be installed
```

Hide

```
install.packages("reshape2")
```

```
Error in install.packages : Updating loaded packages
```

Hide

```
summary(data)
```

```
   bldg_id       time_split_numeric  in.county
Min.   :    65   Min.   :1.0        Length:34260
1st Qu.:137588   1st Qu.:2.0        Class :character
Median :277502   Median :3.5        Mode  :character
Mean   :276418   Mean   :3.5
3rd Qu.:411188   3rd Qu.:5.0
Max.   :549916   Max.   :6.0

 time_split        in.county_and_puma    in.sqft
Length:34260       Length:34260       Min.   : 328
Class :character   Class :character   1st Qu.:1220
Mode  :character   Mode  :character   Median :1690
                                      Mean   :2114
                                      3rd Qu.:2176
                                      Max.   :8194

 in.bedrooms   in.building_america_climate_zone
Min.   :1.00   Length:34260
1st Qu.:3.00   Class :character
Median :3.00   Mode  :character
Mean   :3.26
3rd Qu.:4.00
Max.   :5.00

in.ceiling_fan       in.city          in.clothes_dryer
Length:34260       Length:34260       Length:34260
Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character




in.clothes_washer  in.cooking_range   in.cooling_setpoint
Length:34260       Length:34260       Min.   :60
Class :character   Class :character   1st Qu.:70
Mode  :character   Mode  :character   Median :72
                                      Mean   :73
                                      3rd Qu.:76
                                      Max.   :80

in.cooling_setpoint_offset_magnitude in.dishwasher
Length:34260                         Length:34260
Class :character                     Class :character
Mode  :character                     Mode  :character




in.federal_poverty_level in.geometry_attic_type
Length:34260             Length:34260
Class :character         Class :character
Mode  :character         Mode  :character
```

```
in.geometry_floor_area in.geometry_floor_area_bin
Length:34260           Length:34260
Class :character       Class :character
Mode  :character       Mode  :character
```

```
in.geometry_garage in.heating_fuel    in.heating_setpoint
Length:34260       Length:34260       Min.   :55.0
Class :character   Class :character   1st Qu.:68.0
Mode  :character   Mode  :character   Median :70.0
                                      Mean   :68.9
                                      3rd Qu.:72.0
                                      Max.   :80.0
```

```
in.heating_setpoint_offset_magnitude in.hot_water_fixtures
Length:34260                         Length:34260
Class :character                     Class :character
Mode  :character                     Mode  :character
```

```
in.hvac_cooling_efficiency
Length:34260
Class :character
Mode  :character
```

```
in.hvac_cooling_partial_space_conditioning in.hvac_cooling_type
Length:34260                               Length:34260
Class :character                           Class :character
Mode  :character                           Mode  :character
```

```
in.hvac_has_ducts  in.hvac_has_zonal_electric_heating
Length:34260       Length:34260
Class :character   Class :character
Mode  :character   Mode  :character
```

```
in.hvac_heating_efficiency in.hvac_heating_type
Length:34260               Length:34260
Class :character           Class :character
Mode  :character           Mode  :character
```

```
in.hvac_heating_type_and_fuel   in.income
Length:34260                  Min.   : 14999
Class :character              1st Qu.: 39999
Mode  :character              Median : 69999
                              Mean   : 77849
                              3rd Qu.: 99999
                              Max.   :199999
                              NA's   :3996
in.income_recs_2015 in.income_recs_2020 in.infiltration
Length:34260        Length:34260        Length:34260
Class :character    Class :character    Class :character
Mode  :character    Mode  :character    Mode  :character
```

```
in.insulation_ceiling in.insulation_floor
Length:34260          Length:34260
Class :character      Class :character
Mode  :character      Mode  :character
```

```
in.insulation_foundation_wall in.insulation_rim_joist
Length:34260                  Length:34260
Class :character              Class :character
Mode  :character              Mode  :character
```

```
in.insulation_roof in.insulation_slab in.insulation_wall
Length:34260       Length:34260       Length:34260
Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character
```

```
in.lighting        in.misc_extra_refrigerator
Length:34260       Length:34260
Class :character   Class :character
Mode  :character   Mode  :character
```

```
in.misc_freezer    in.misc_gas_fireplace in.misc_gas_grill
Length:34260       Length:34260          Length:34260
Class :character   Class :character      Class :character
Mode  :character   Mode  :character      Mode  :character
```

```
in.misc_gas_lighting in.misc_hot_tub_spa in.misc_pool
Length:34260         Length:34260        Length:34260
Class :character     Class :character    Class :character
Mode  :character     Mode  :character    Mode  :character
```

```
in.misc_pool_heater in.misc_pool_pump  in.misc_well_pump
Length:34260        Length:34260       Length:34260
Class :character    Class :character   Class :character
Mode  :character    Mode  :character   Mode  :character
```

```
in.natural_ventilation in.occupants       in.orientation
Length:34260           Length:34260       Length:34260
Class :character       Class :character   Class :character
Mode  :character       Mode  :character   Mode  :character
```

```
in.plug_load_diversity in.refrigerator    in.roof_material
Length:34260           Length:34260       Length:34260
Class :character       Class :character   Class :character
Mode  :character       Mode  :character   Mode  :character
```

```
 in.tenure         in.usage_level     in.vacancy_status
Length:34260       Length:34260       Length:34260
Class :character   Class :character   Class :character
Mode  :character   Mode  :character   Mode  :character
```

```
 in.vintage        in.vintage_acs
Length:34260       Length:34260
Class :character   Class :character
Mode  :character   Mode  :character
```

```
in.water_heater_efficiency in.water_heater_fuel
Length:34260               Length:34260
Class :character           Class :character
Mode  :character           Mode  :character
```

```
in.weather_file_city in.weather_file_latitude
Length:34260         Min.   :32.5
Class :character     1st Qu.:33.5
Mode  :character     Median :34.0
                     Mean   :34.0
                     3rd Qu.:34.9
                     Max.   :35.4

in.weather_file_longitude in.window_areas     in.windows
Min.   :-82.9             Length:34260       Length:34260
1st Qu.:-82.2             Class :character   Class :character
Median :-81.0             Mode  :character   Mode  :character
Mean   :-81.0
3rd Qu.:-80.0
Max.   :-78.9

upgrade.insulation_roof upgrade.water_heater_efficiency
Length:34260            Length:34260
Class :character        Class :character
Mode  :character        Mode  :character




upgrade.clothes_dryer upgrade.hvac_heating_efficiency
Length:34260          Length:34260
Class :character      Class :character
Mode  :character      Mode  :character




upgrade.cooking_range out.kitchen_energy_consumption
Length:34260          Min.   :0.000
Class :character      1st Qu.:0.095
Mode  :character      Median :0.134
                      Mean   :0.147
                      3rd Qu.:0.184
                      Max.   :0.817

out.laundry_energy_consumption
Min.   :0.000
1st Qu.:0.000
Median :0.021
Mean   :0.033
3rd Qu.:0.050
Max.   :0.397

out.heating_cooling_energy_consumption
Min.   :0.00
1st Qu.:0.33
Median :0.50
Mean   :0.60
3rd Qu.:0.77
Max.   :8.18
```

```
out.water_heating_energy_consumption
Min.   :0.000
1st Qu.:0.025
Median :0.041
Mean   :0.052
3rd Qu.:0.066
Max.   :1.017

out.electrical_appliances_energy_consumption
Min.   :0.000
1st Qu.:0.241
Median :0.374
Mean   :0.410
3rd Qu.:0.559
Max.   :2.565

out.outdoor_appliances_energy_consumption
Min.   :0.00
1st Qu.:0.00
Median :0.00
Mean   :0.06
3rd Qu.:0.02
Max.   :4.50

out.renewable_energy_energy_consumption
Min.   :0
1st Qu.:0
Median :0
Mean   :0
3rd Qu.:0
Max.   :0

out.total_energy_consumption time_range.x
Min.   :0.06                 Length:34260
1st Qu.:0.83                 Class :character
Median :1.18                 Mode  :character
Mean   :1.31
3rd Qu.:1.63
Max.   :8.33

Dry_Bulb_Temperature_C Relative_Humidity Wind_Speed_m_s
Min.   :21.1           Min.   :51.4      Min.   :0.43
1st Qu.:24.5           1st Qu.:64.4      1st Qu.:1.60
Median :26.9           Median :72.5      Median :2.48
Mean   :26.9           Mean   :75.2      Mean   :2.41
3rd Qu.:29.4           3rd Qu.:87.3      3rd Qu.:2.96
Max.   :31.8           Max.   :96.1      Max.   :5.27

Wind_Direction_Deg Global_Horizontal_Radiation_W_m2
Min.   : 30.2      Min.   :  0
1st Qu.:114.6      1st Qu.: 28
Median :139.3      Median :210
Mean   :134.4      Mean   :291
3rd Qu.:156.8      3rd Qu.:565
Max.   :211.2      Max.   :788
```

```
 Direct_Normal_Radiation_W_m2 Diffuse_Horizontal_Radiation_W_m2
 Min.   :  0                  Min.   :  0
 1st Qu.: 52                  1st Qu.: 16
 Median :206                  Median : 91
 Mean   :233                  Mean   :121
 3rd Qu.:428                  3rd Qu.:218
 Max.   :545                  Max.   :324

 Next_year_Pred Change_in_energy
 Min.   :0.09   Min.   :-4.41
 1st Qu.:0.95   1st Qu.: 0.05
 Median :1.32   Median : 0.11
 Mean   :1.46   Mean   : 0.15
 3rd Qu.:1.80   3rd Qu.: 0.21
 Max.   :6.90   Max.   : 1.68
```

Hide

```
head(data)
```

bldg_id <dbl>	time_split_numeric <dbl>	in.county <chr>	time_split <chr>	in.county_and_puma <chr>	in.sqft <dbl>	in
65	1	G4500910	Late Night	G4500910, G45000502	885	
65	2	G4500910	Early Morning	G4500910, G45000502	885	
65	3	G4500910	Morning	G4500910, G45000502	885	
65	4	G4500910	Noon	G4500910, G45000502	885	
65	5	G4500910	Evening	G4500910, G45000502	885	
65	6	G4500910	Night	G4500910, G45000502	885	

6 rows | 1-7 of 96 columns

Hide

```
#The data was in 24hours timeline, we have divided the data into 5 parts. Namely:
#Early Morning, Morning, Noon, Night and Late Night.
#Further we will be analyzing the electricity consumption of each section.
```

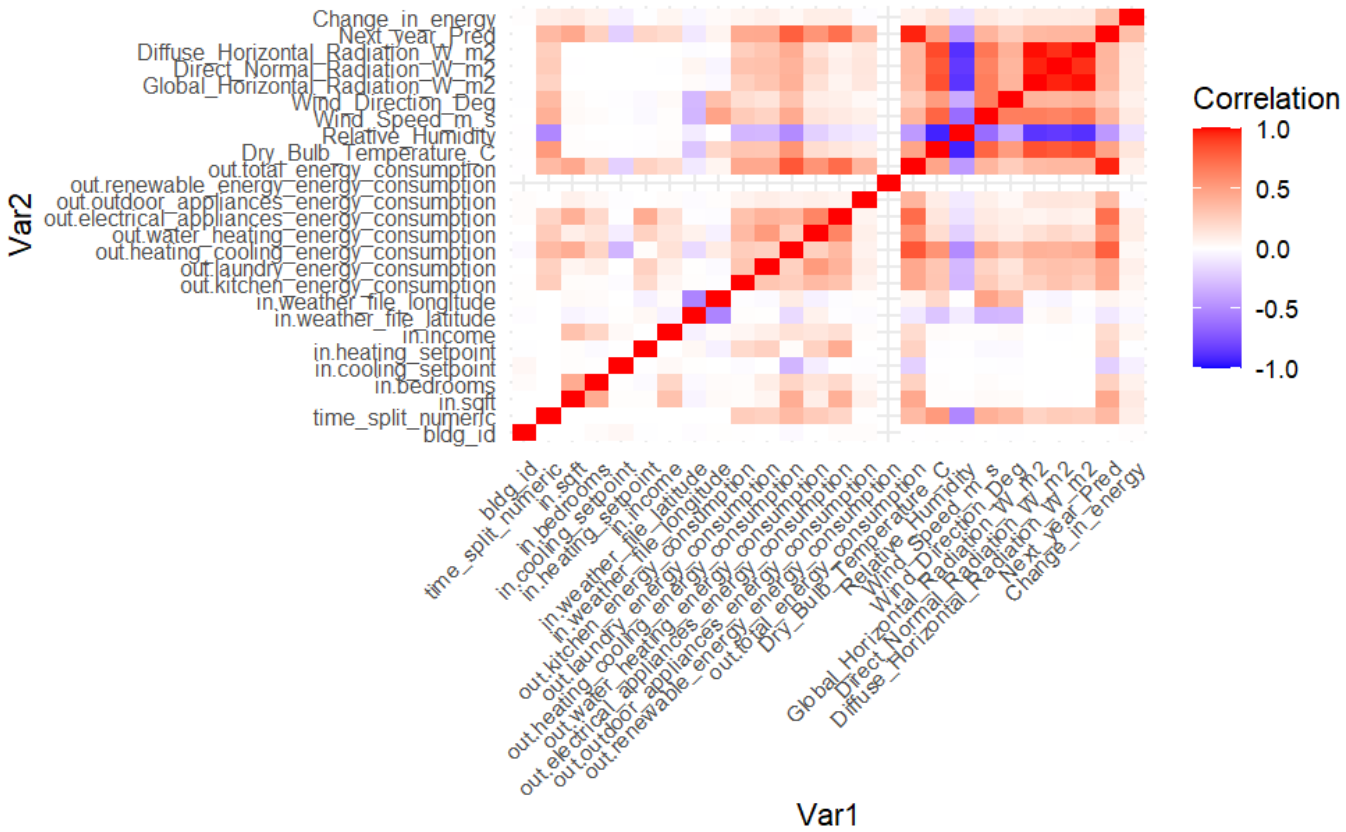
Hide

```
#First we output the correlation matrix of only the numeric values.
#Then we calculate the correlation matrix followed by the melt correlatoin matrix.
#The melted correlation matrix is created using the melt function,
#which transforms the correlation matrix into a long format.
#Then we display the heatmap plot.
#If we observer the image after saving it on our device we can see that the red zones are hig
hly correlated.
#We can nitpick the columns from the heat map but it is not the most efficient way of going a
bout when performing EDA.

library(reshape2)
library(ggplot2)
numeric_data <- data[sapply(data, is.numeric)]
cor_matrix <- cor(numeric_data, use = "complete.obs")
```

```
Warning: the standard deviation is zero
```

Hide

```
melted_cor_matrix <- melt(cor_matrix, na.rm = TRUE)
heatmap_plot <- ggplot(data = melted_cor_matrix,
                       aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name = "Correlation") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1,
                                   size = 8, hjust = 1),
        axis.text.y = element_text(size = 8))
print(heatmap_plot)
```



Hide

```
#Check dimensions of the melted correlation matrix.
#The melted correlation matrix has more rows because it represents
#each unique pair of variables and their correlations separately,
#whereas the correlation matrix is a square matrix
#showing all pairwise correlations between variables.
numeric_data_dimensions <- dim(numeric_data)
print("Dimensions of numeric_data:")
```

```
[1] "Dimensions of numeric_data:"
```

Hide

```
print(numeric_data_dimensions)
```

```
[1] 34260    26
```

Hide

```
cor_matrix_dimensions <- dim(cor_matrix)
melted_cor_matrix_dimensions <- dim(melted_cor_matrix)
print("Dimensions of Correlation Matrix:")
```

```
[1] "Dimensions of Correlation Matrix:"
```

Hide

```
print(cor_matrix_dimensions)
```

```
[1] 26 26
```

Hide

```
print("Dimensions of Melted Correlation Matrix:")
```

```
[1] "Dimensions of Melted Correlation Matrix:"
```

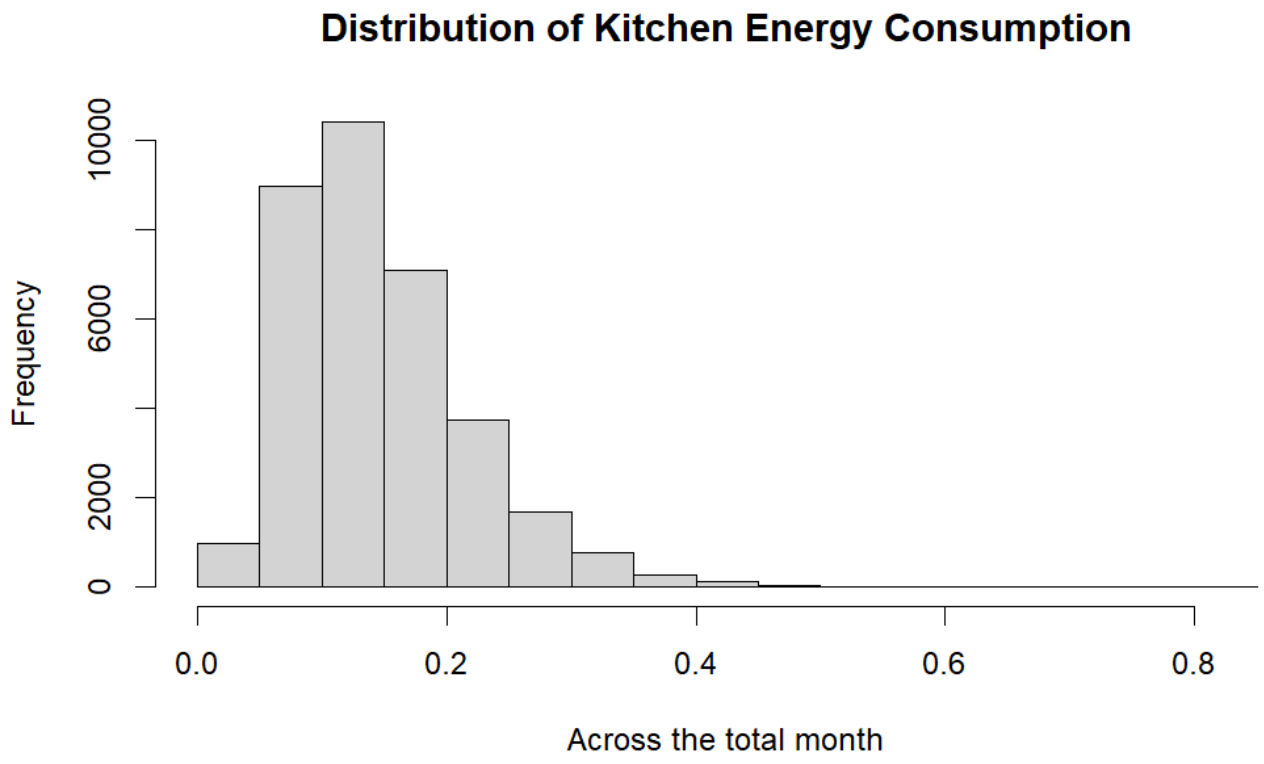
Hide

```
print(melted_cor_matrix_dimensions)
```

```
[1] 626   3
```

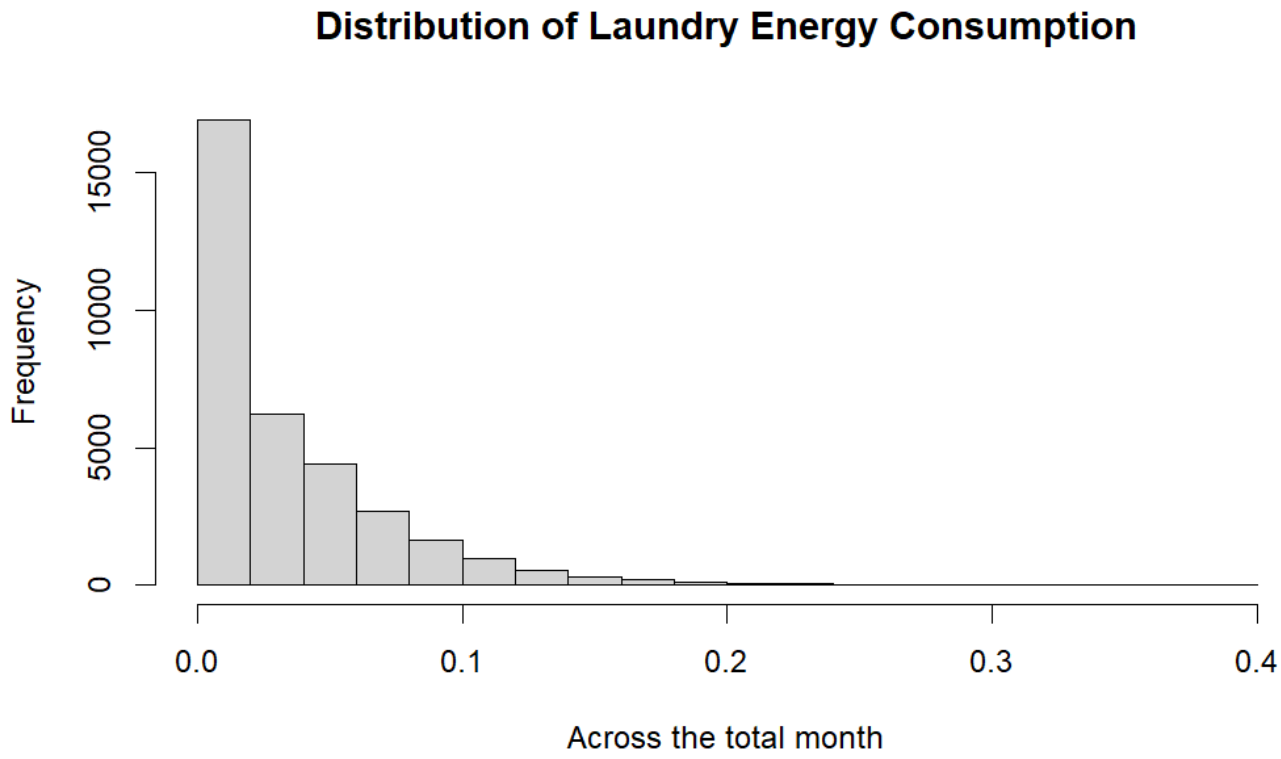
Hide

```
#Visualize the distribution of a few particular numeric variable like to get a better underst
anding of the column values.
#In this case we are looking at the #toal_energy_consumption.It is left skewed as most of the
phenomenons in the world.


hist(data$out.kitchen_energy_consumption, main="Distribution of Kitchen Energy Consumption",
xlab="Across the total month")
```



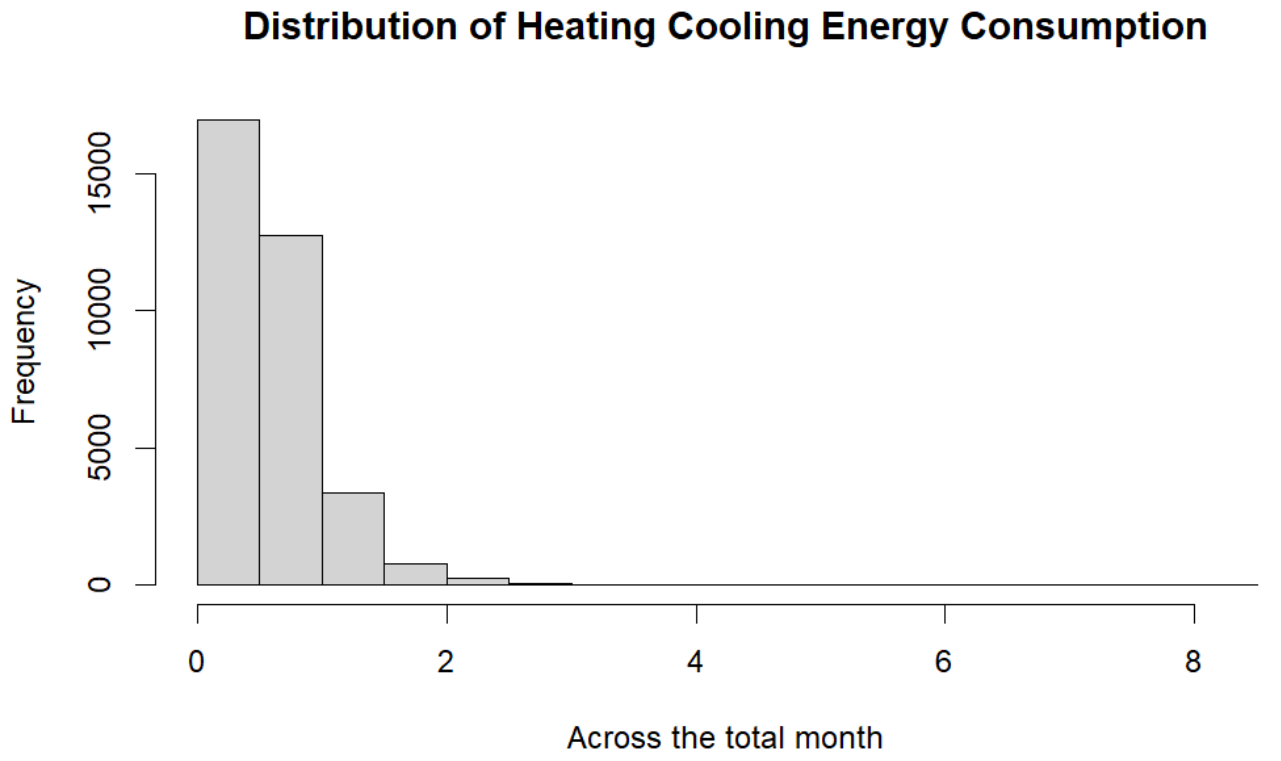
Hide

```
hist(data$out.laundry_energy_consumption, main="Distribution of Laundry Energy Consumption",
xlab="Across the total month")
```



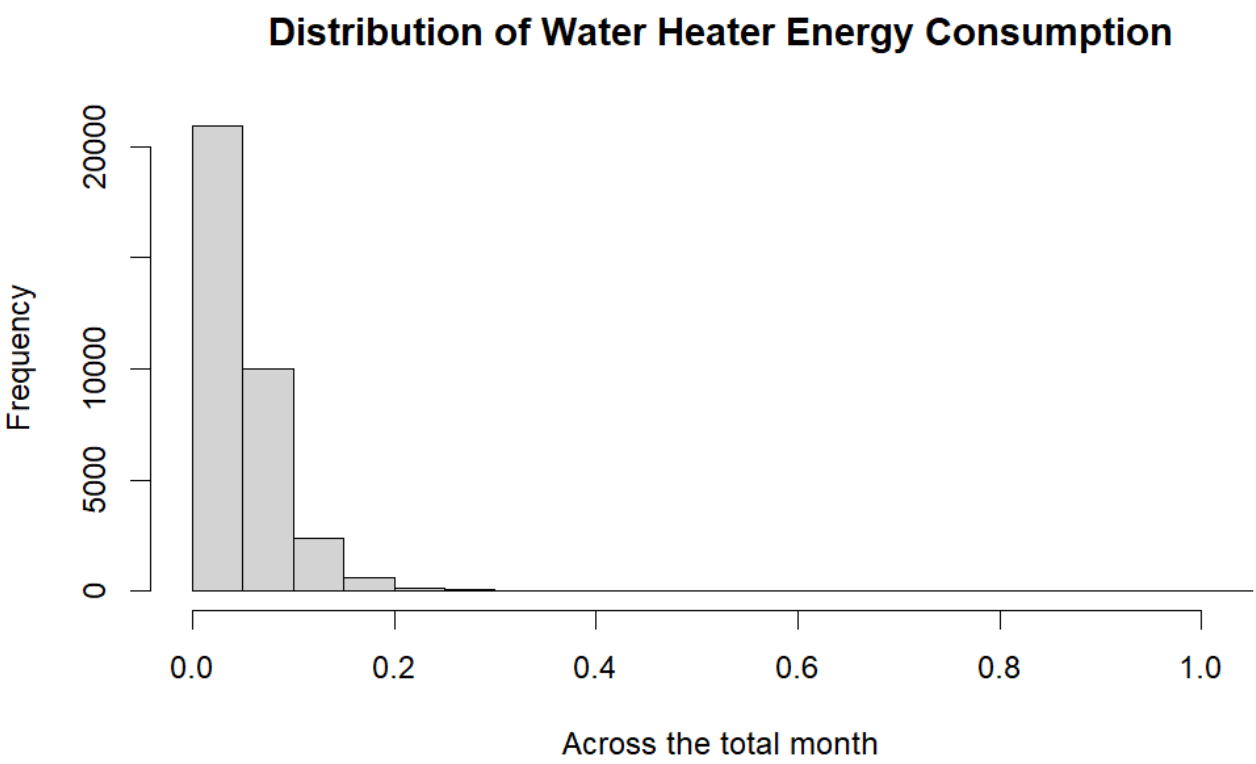
Hide

```
hist(data$out.heating_cooling_energy_consumption, main="Distribution of Heating Cooling Energ
y Consumption", xlab="Across the total month")
```



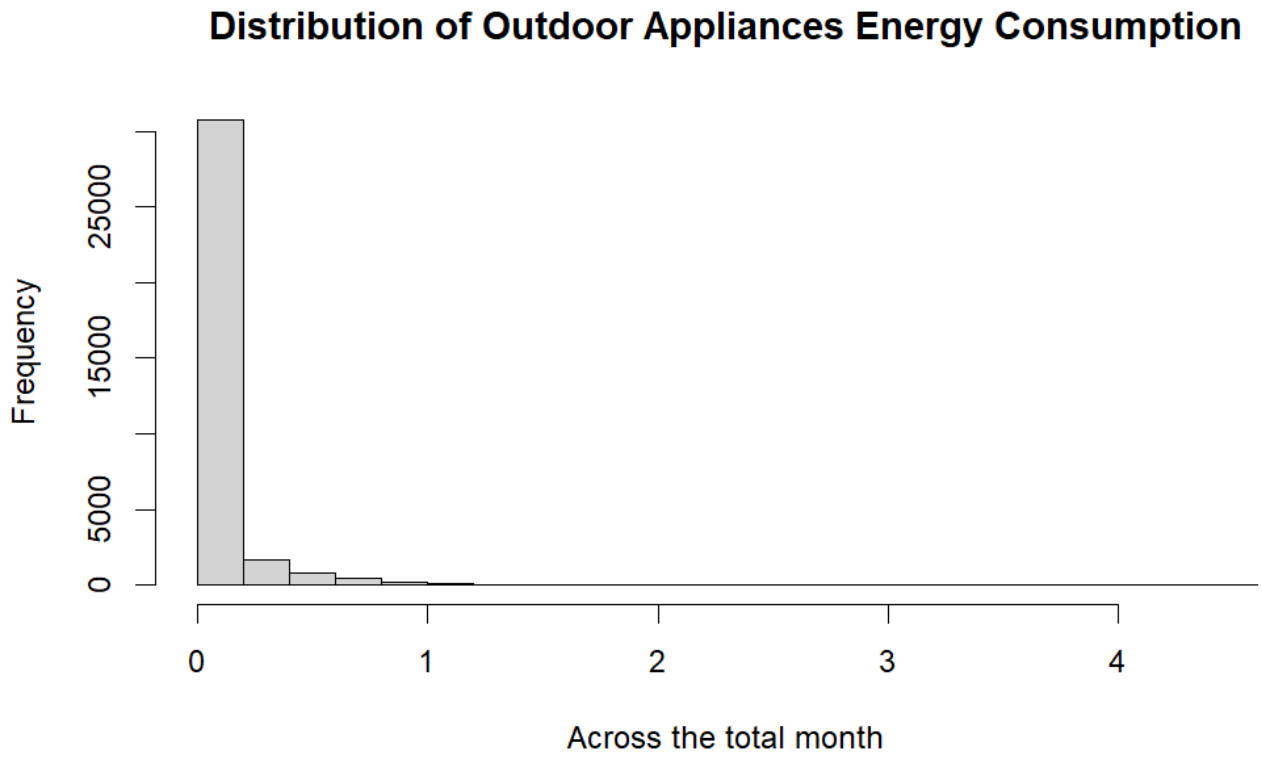
Hide

```
hist(data$out.water_heating_energy_consumption, main="Distribution of Water Heater Energy Con
sumption", xlab="Across the total month")
```



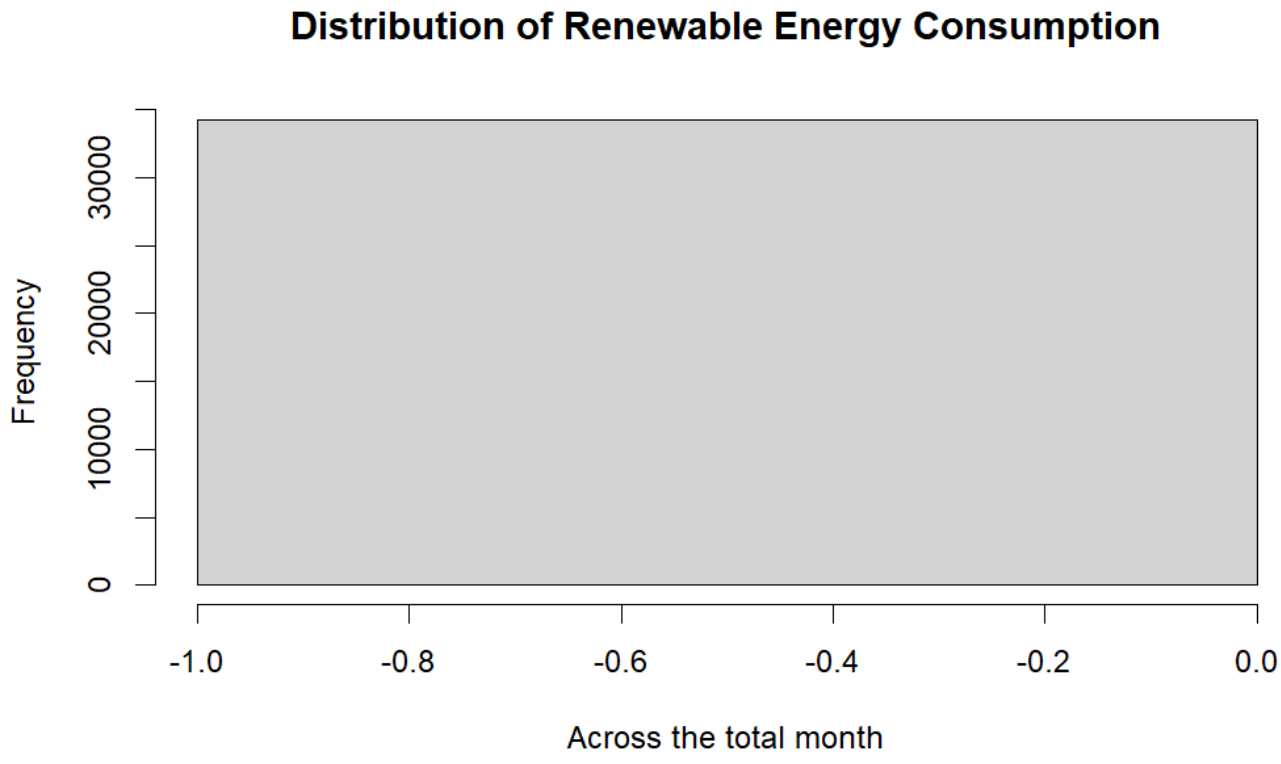
Hide

```
hist(data$out.outdoor_appliances_energy_consumption, main="Distribution of Outdoor Appliances
Energy Consumption", xlab="Across the total month")
```



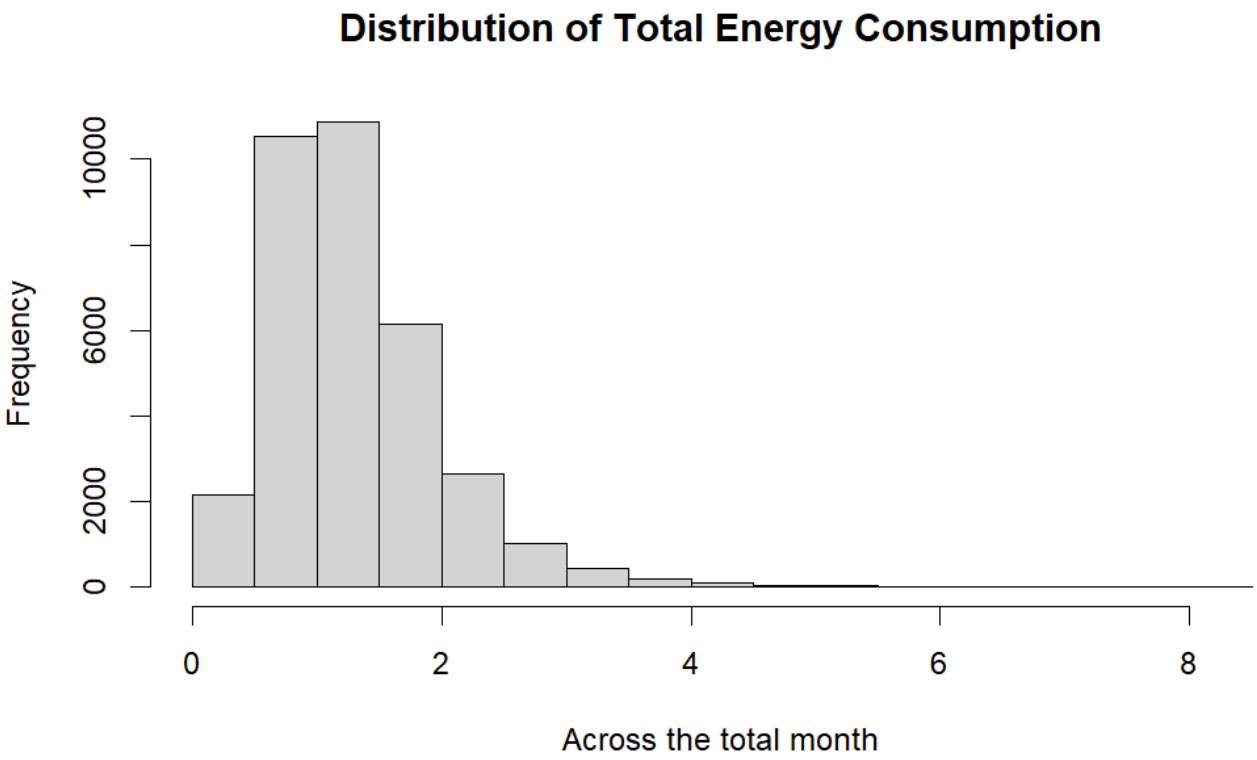
Hide

```
hist(data$out.renewable_energy_energy_consumption, main="Distribution of Renewable Energy Con
sumption", xlab="Across the total month")
```



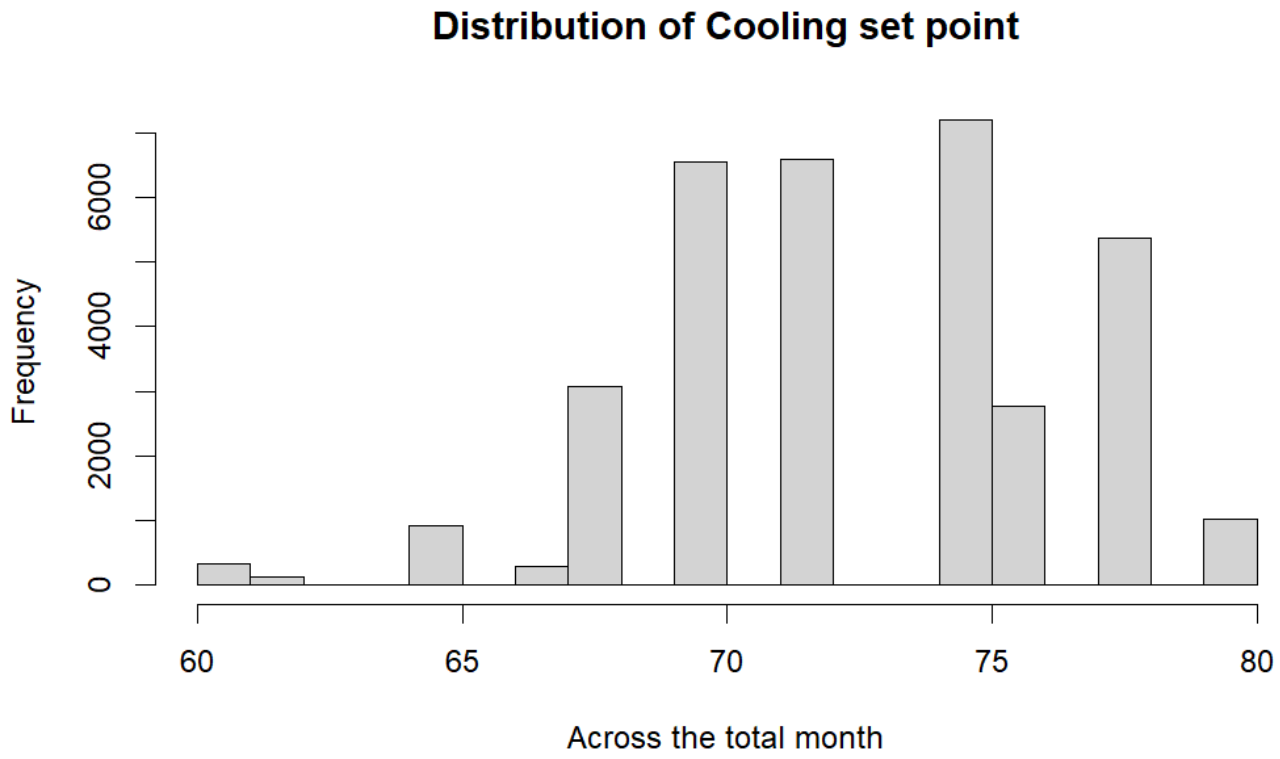
Hide

```
hist(data$out.total_energy_consumption, main="Distribution of Total Energy Consumption", xlab
="Across the total month")
```



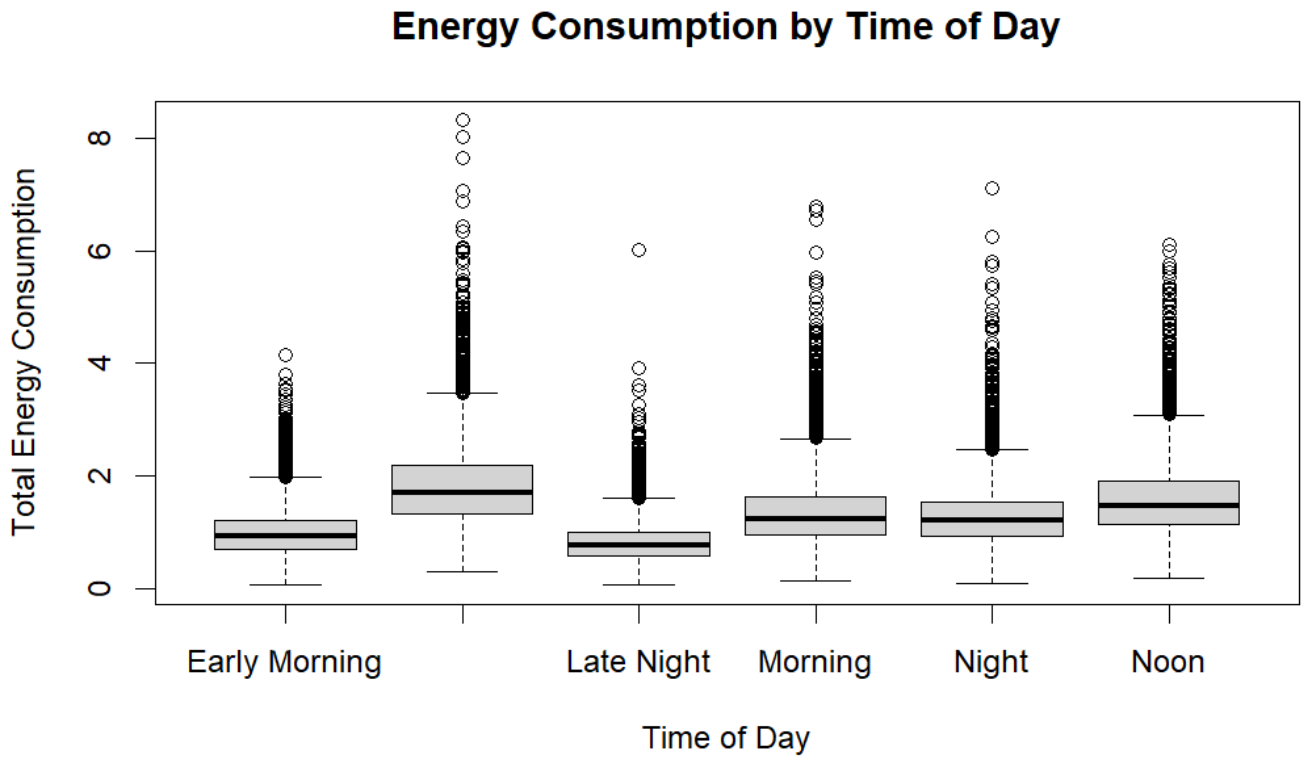
Hide

```
hist(data$in.cooling_setpoint, main="Distribution of Cooling set point", xlab="Across the tot
al month")
```



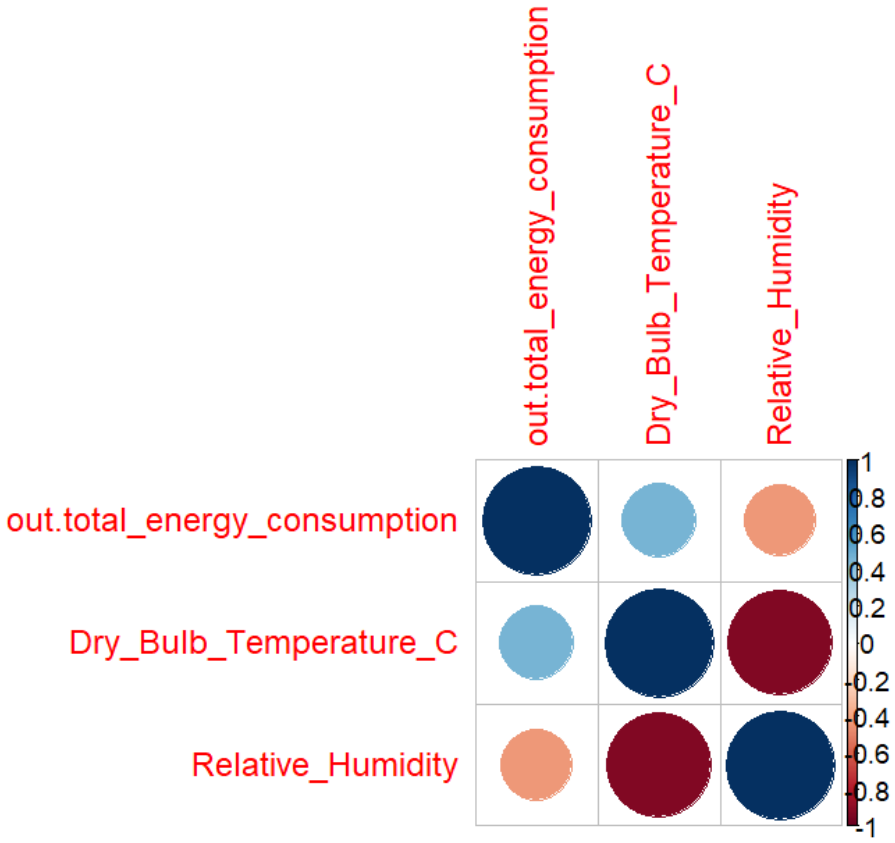
Hide

```
#Compare the energy consumption across different times of the day or #climate zones
#The boxplot you've provided visualizes the total energy consumption by time of day.
#The times of day are categorized as Early Morning, Late Night, Morning, Night, and Noon.
#From the boxplot, we can observe the following:
#The median energy consumption is highest in the Early Morning and Morning times.
#The distribution of energy consumption during the Early Morning and Night times has more out
liers on the higher side,
#indicating sporadic increases in energy consumption during these times.
#The Noon time has the lowest median energy consumption among the times of day displayed.
#The box for the Morning time is slightly skewed upwards, suggesting a higher variability wit
h more data points on the higher end of consumption.
#The spread of data in terms of IQR seems to be relatively similar across all times of day,
#except for Noon, which appears to have a slightly tighter IQR, indicating less variability i
n energy consumption during this time.


boxplot(data$out.total_energy_consumption ~ data$time_split, main="Energy Consumption by Time
of Day", xlab="Time of Day", ylab="Total Energy Consumption")
```



Hide

```
#We'll examine the relationship between energy consumption and weather variables such as temp
erature and humidity.
#Assuming your dataset has weather variables named 'temperature' and 'humidity'.
#Calculate the correlation matrix for the energy consumption and weather variables.
# Visualize the correlation matrix.
#There is a strong positive correlation between out.total_energy_consumption and Dry_Bulb_Tem
perature_C.
#This indicates that as the dry bulb temperature increases, the total energy consumption tend
s to increase as well, which is expected in climates
#where higher temperatures lead to increased use of air conditioning.
#There is a smaller, possibly negative correlation between out.total_energy_consumption and R
elative_Humidity.
#This could suggest that as relative humidity increases, the total energy consumption slightl
y decreases or doesn't change much. However, the correlation seems weak.
#The correlation between Dry_Bulb_Temperature_C and Relative_Humidity appears to be negative
and moderate,
#indicating that typically, as the temperature increases, the relative humidity tends to decr
ease, which is a common atmospheric behavior.

weather_correlation <- data %>%
  select(out.total_energy_consumption, Dry_Bulb_Temperature_C, Relative_Humidity) %>%
  cor(use = "complete.obs")
library(corrplot)
corrplot(weather_correlation, method = "circle")
```

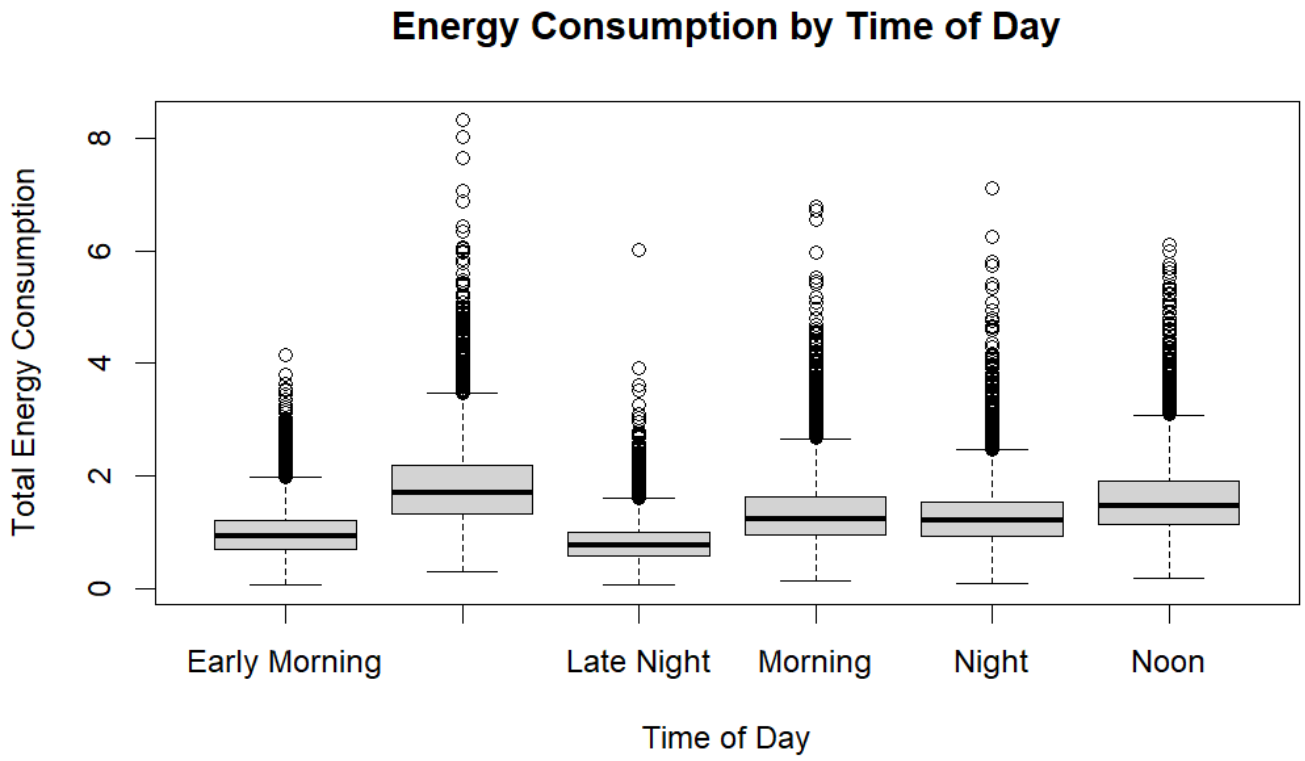


Hide

```
png("C:/Users/hp/Desktop/IDS Project/weather_correlation_plot.png", width = 800, height = 600)
```

Hide

```
#Compare the energy consumption across different times of the day or #climate zones

boxplot(data$out.total_energy_consumption ~ data$time_split, main="Energy Consumption by Time of Day", xlab="Time of Day", ylab="Total Energy Consumption")
```



Hide

```
#This helps us under that in the morning during the busy hours the
#energy consumption is more
```

Hide

```
#Geographical Analysis: If location data is available, plot the data points on a map.

library(ggplot2)
library(maps)
ggmap <- map_data("state")
ggplot() + geom_polygon(data = ggmap, aes(x = long, y = lat, group = group)) + geom_point(dat
a = data, aes(x = longitude, y = latitude), color = "red", size = 1)
```

```
Error in `geom_point()`:
! Problem while computing aesthetics.
ℹ Error occurred in the 2nd layer.
Caused by error:
! object 'longitude' not found
Backtrace:
  1. base (local) `<fn>`(x)
  2. ggplot2:::print.ggplot(x)
  4. ggplot2:::ggplot_build.ggplot(x)
  5. ggplot2:::by_layer(...)
 12. ggplot2 (local) f(l = layers[[i]], d = data[[i]])
 13. l$compute_aesthetics(d, plot)
 14. ggplot2 (local) compute_aesthetics(..., self = self)
 15. base::lapply(aesthetics, eval_tidy, data = data, env = env)
 16. rlang (local) FUN(X[[i]], ...)
```

Hide

```
install.packages("factoextra")
```

```
WARNING: Rtools is required to build R packages but is not currently installed. Please downlo
ad and install the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/hp/AppData/Local/R/win-library/4.3’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.3/factoextra_1.0.7.zip'
Content type 'application/zip' length 415631 bytes (405 KB)
downloaded 405 KB
```

```
package ‘factoextra’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
    C:\Users\hp\AppData\Local\Temp\RtmpiC3l3G\downloaded_packages
```

Hide

```
#Principal Component Analysis (PCA): Use PCA for dimensionality reduction, especially if the
dataset has many numeric variables.
library(factoextra)
```

```
Warning: package ‘factoextra’ was built under R version 4.3.2Loading required package: ggplot
2
Warning: package ‘ggplot2’ was built under R version 4.3.2Welcome! Want to learn more? See tw
o factoextra-related books at https://goo.gl/ve3WBa
```

Hide

```
library(caret)
```

```
Warning: package ‘caret’ was built under R version 4.3.2Loading required package: lattice
```

Hide

```
# Select numeric columns
num_data <- data[, sapply(data, is.numeric)]

# Identify columns with near zero or zero variance
nzv <- nearZeroVar(num_data)

# Exclude columns with near zero or zero variance
num_data_clean <- num_data[, -nzv]

# Perform PCA on the cleaned data
pca_result <- prcomp(num_data_clean, scale = TRUE)
```

```
Error in svd(x, nu = 0, nv = k) : infinite or missing values in 'x'
```

Hide

```
install.packages("GGally")
```

```
Error in install.packages : Updating loaded packages
```

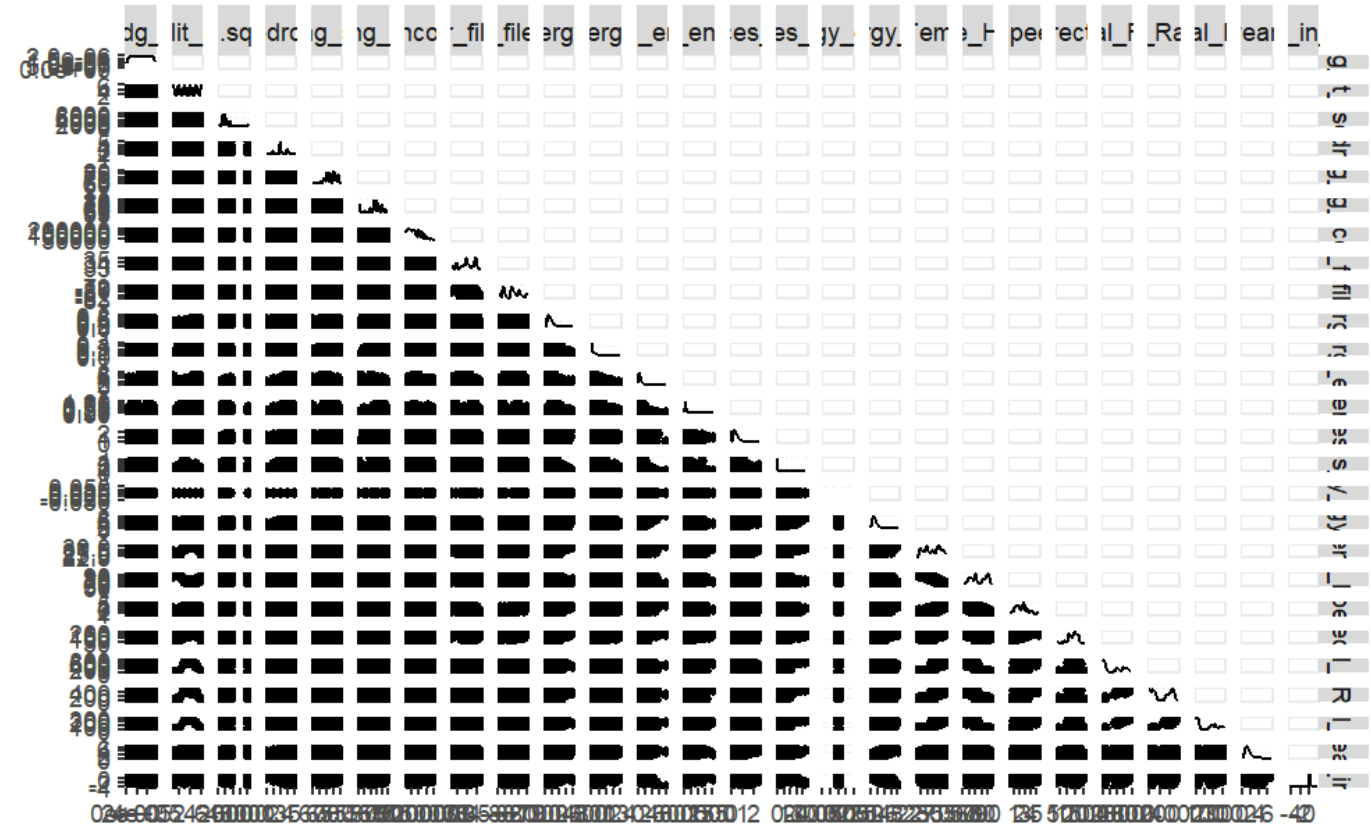
Hide

```
#Pairwise Plot: Visualize relationships between all pairs of numeric variables.
library(GGally)
```

```
Warning: package ‘GGally’ was built under R version 4.3.2Registered S3 method overwritten by
'GGally':
  method from
  +.gg   ggplot2
```

Hide

```
num_data <- data[, sapply(data, is.numeric)]
ggpairs(num_data)
```



Hide

```
NA
NA
NA
```

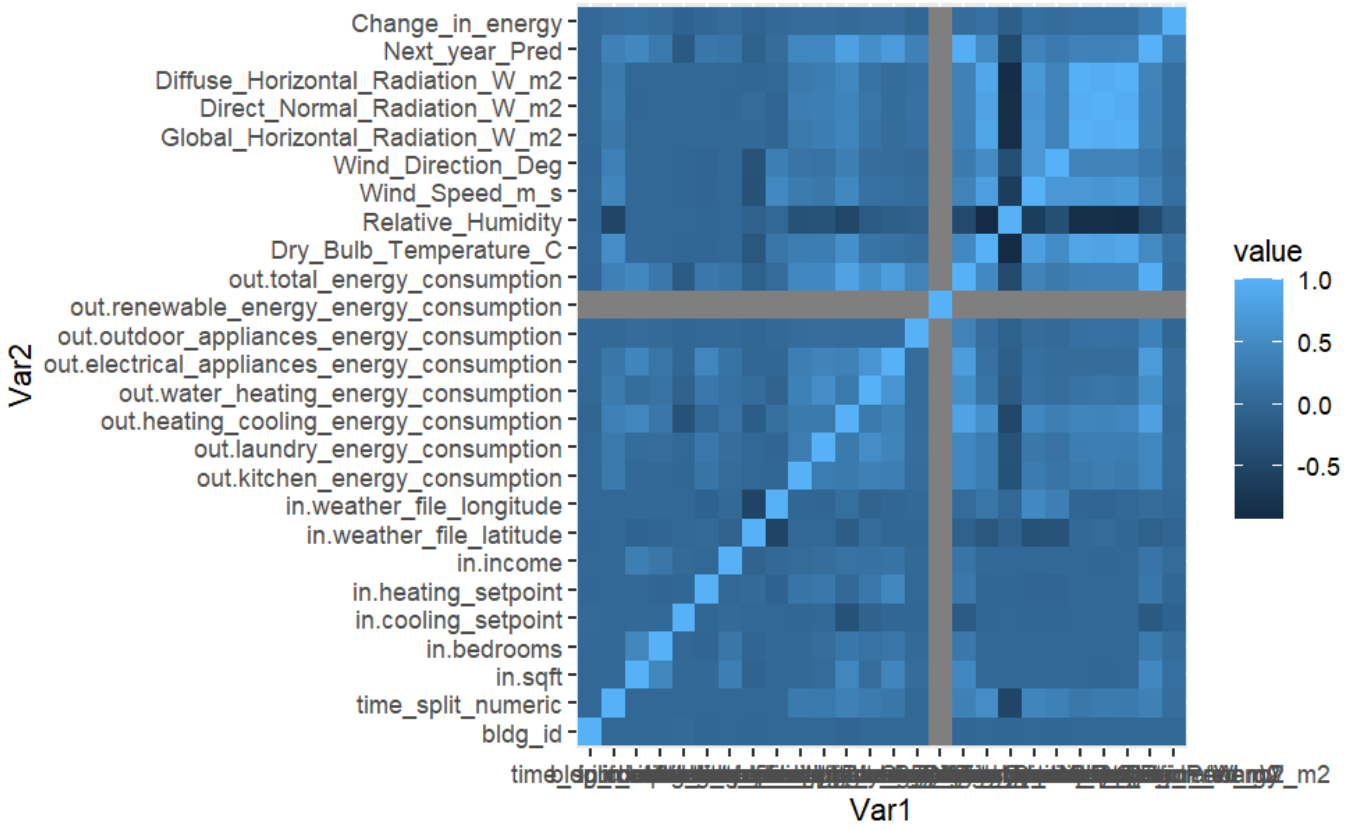
Hide

```
#Heatmap: Visualize the correlation matrix or any large matrix data.
library(ggplot2)
library(reshape2)
```

```
Warning: package 'reshape2' was built under R version 4.3.2
```

Hide

```
cor_melted <- melt(cor_matrix)
ggplot(cor_melted, aes(Var1, Var2, fill=value)) + geom_tile()
```



Hide

```
#Cluster Analysis: Perform cluster analysis to identify groups or patterns in the data.

library(cluster)
num_data <- data[, sapply(data, is.numeric)]
clusters <- kmeans(num_data, 3)
```

```
Error in do_one(nmeth) : NA/NaN/Inf in foreign function call (arg 1)
```

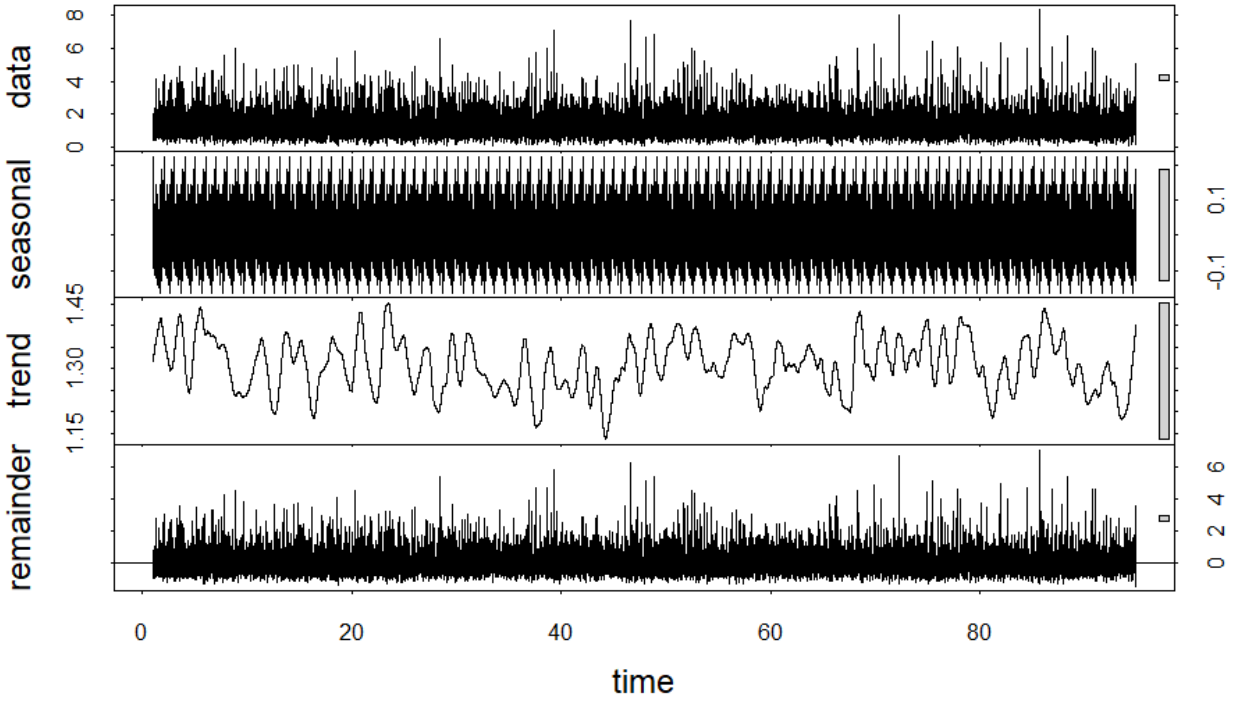
Hide

```
#Time Series Decomposition: If you have time series data, decompose it to analyze trends and
seasonality.

library(forecast)
```

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

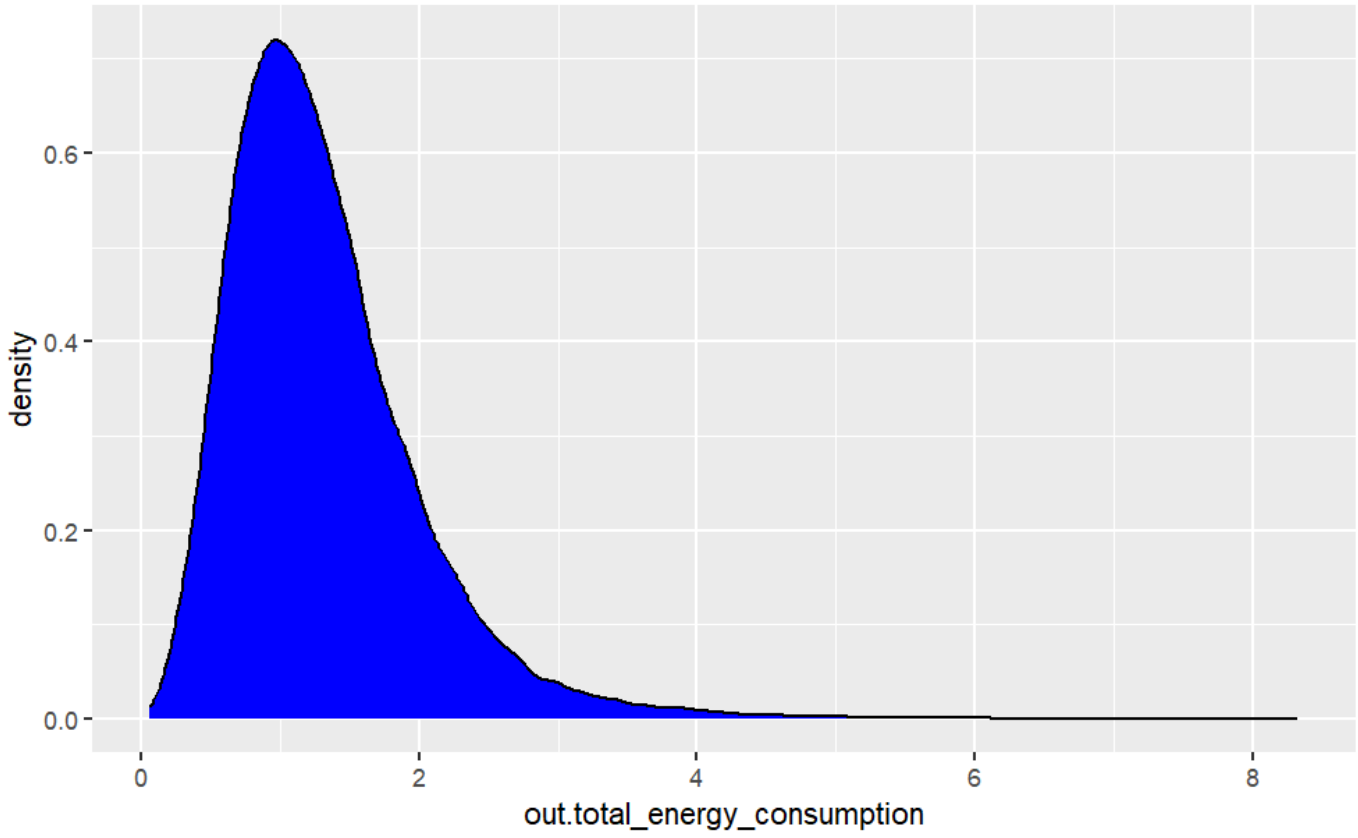
Hide

```
ts_data <- ts(data$out.total_energy_consumption, frequency=365)
decomposed_ts <- stl(ts_data, s.window="periodic")
plot(decomposed_ts)
```



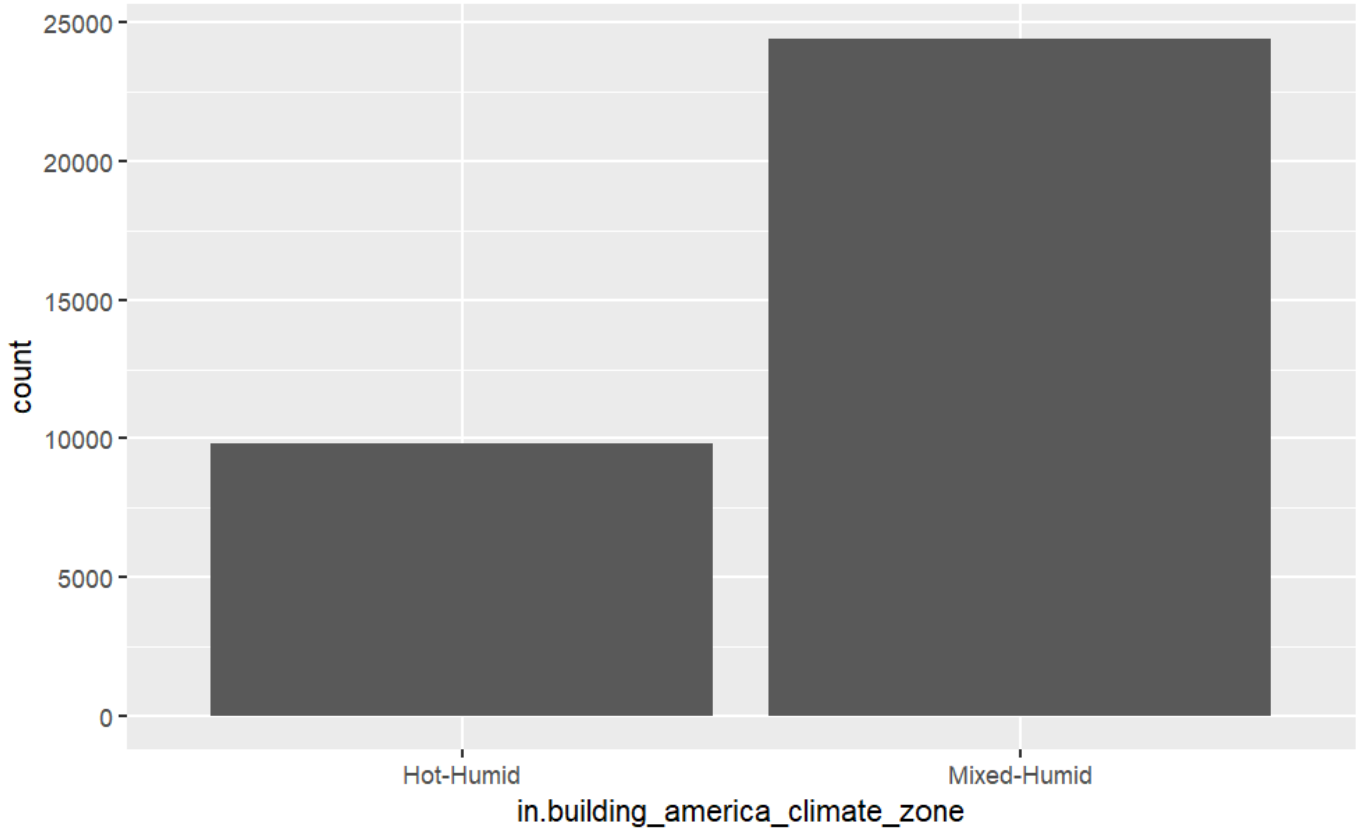
Hide

```
#Density Plot: To see the distribution of a variable.
library(ggplot2)
ggplot(data, aes(x=out.total_energy_consumption)) + geom_density(fill="blue")
```



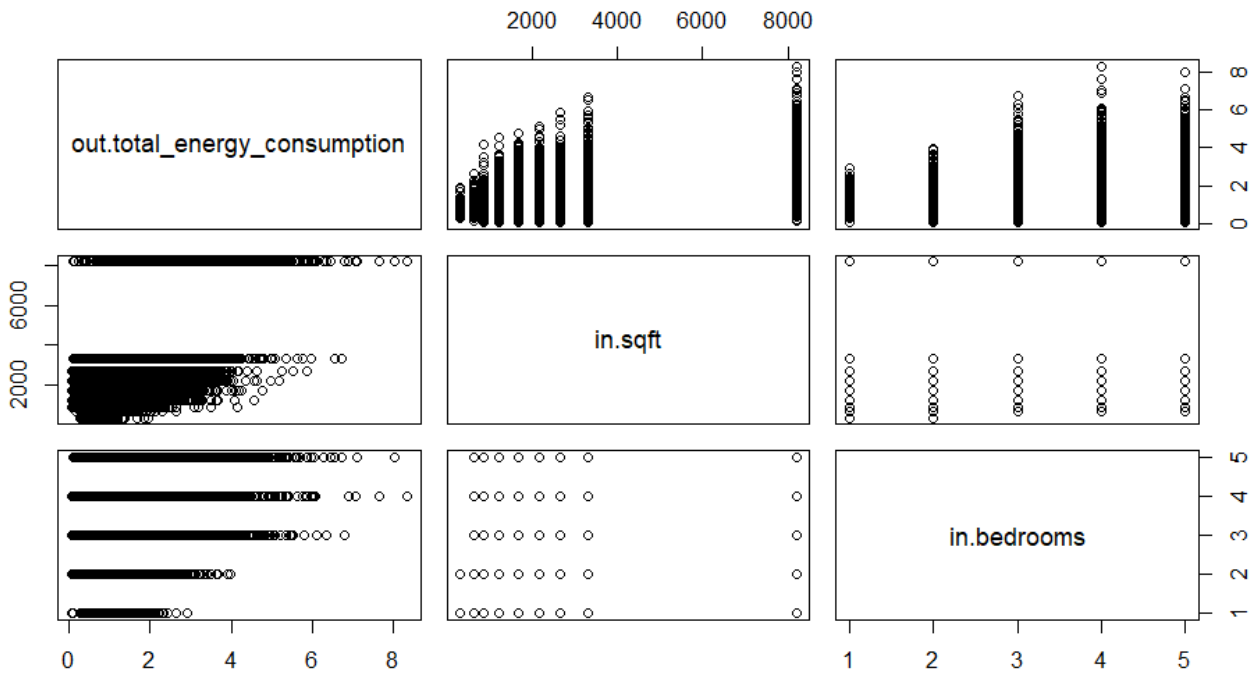
Hide

```
#Bar Plot for Categorical Data: Analyze the frequency of different categories in a categorica
l variable.
ggplot(data, aes(x=in.building_america_climate_zone)) + geom_bar()
```



Hide

```
#Multivariate Analysis: Analyze relationships between multiple variables.
pairs(~out.total_energy_consumption + in.sqft + in.bedrooms, data=data, main="Multivariate An
alysis")
```



Hide

```
NA
NA
```

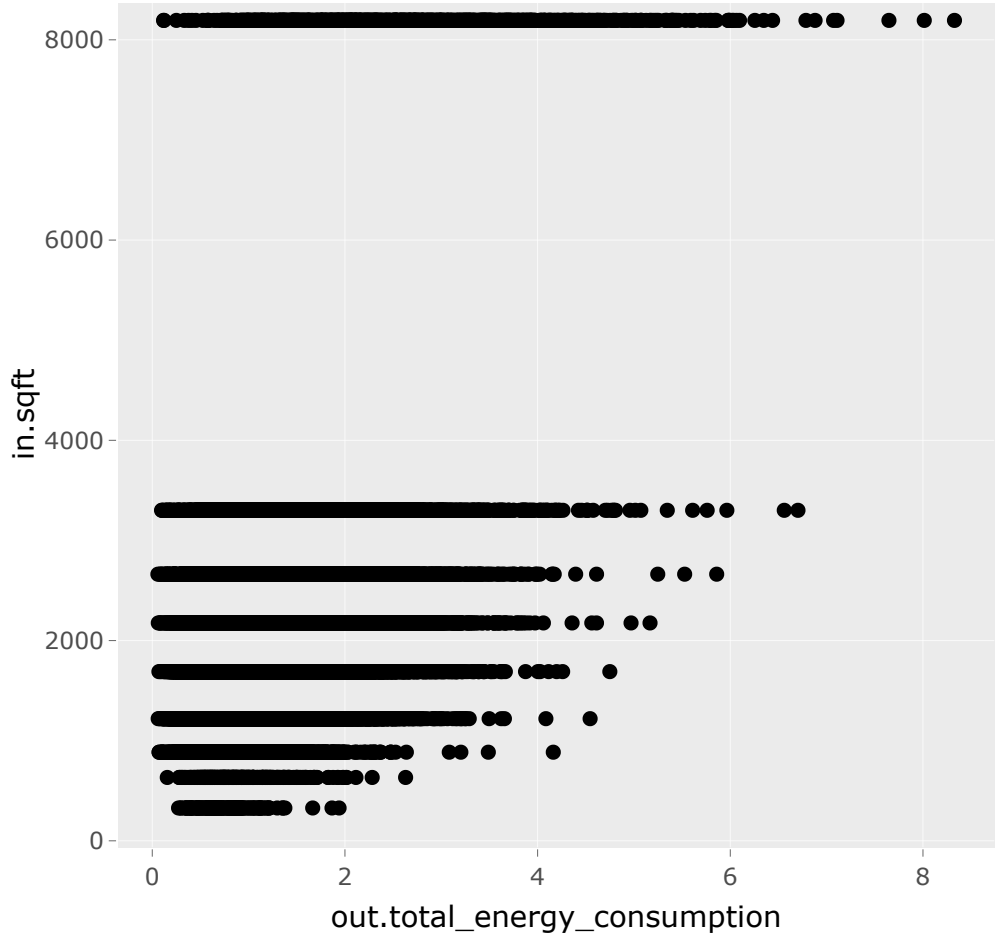
Hide

```
#Interactive Plots with Plotly: Create interactive plots for more detailed exploration.
library(plotly)
```

```
Attaching package: ‘plotly’

The following object is masked from ‘package:ggplot2’:

    last_plot

The following object is masked from ‘package:stats’:

    filter

The following object is masked from ‘package:graphics’:

    layout
```

Hide

```
p <- ggplot(data, aes(x=out.total_energy_consumption, y=in.sqft)) + geom_point()
ggplotly(p)
```



Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Ctrl+Alt+I*.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Ctrl+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

R Markdown

```
file <- '/Users/subhiksha/Downloads/abs_final_data.csv'
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ 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
```

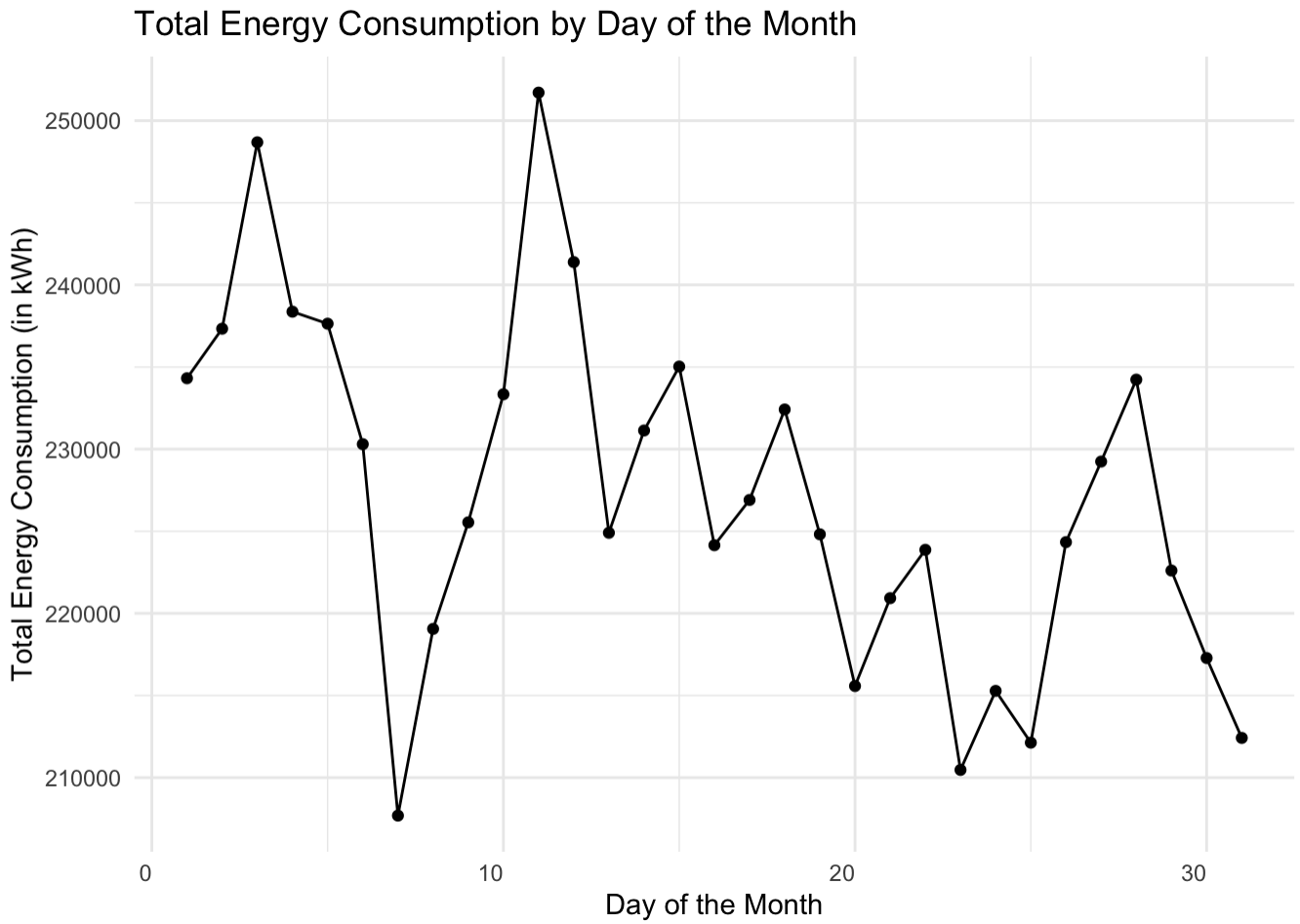
```
final_data <- read_csv(file)
```

```
## Rows: 4248240 Columns: 27
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl  (26): fireplace, grill, lighting.energy_model, pool_heater, pool_pump, ...
## dttm  (1): time
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

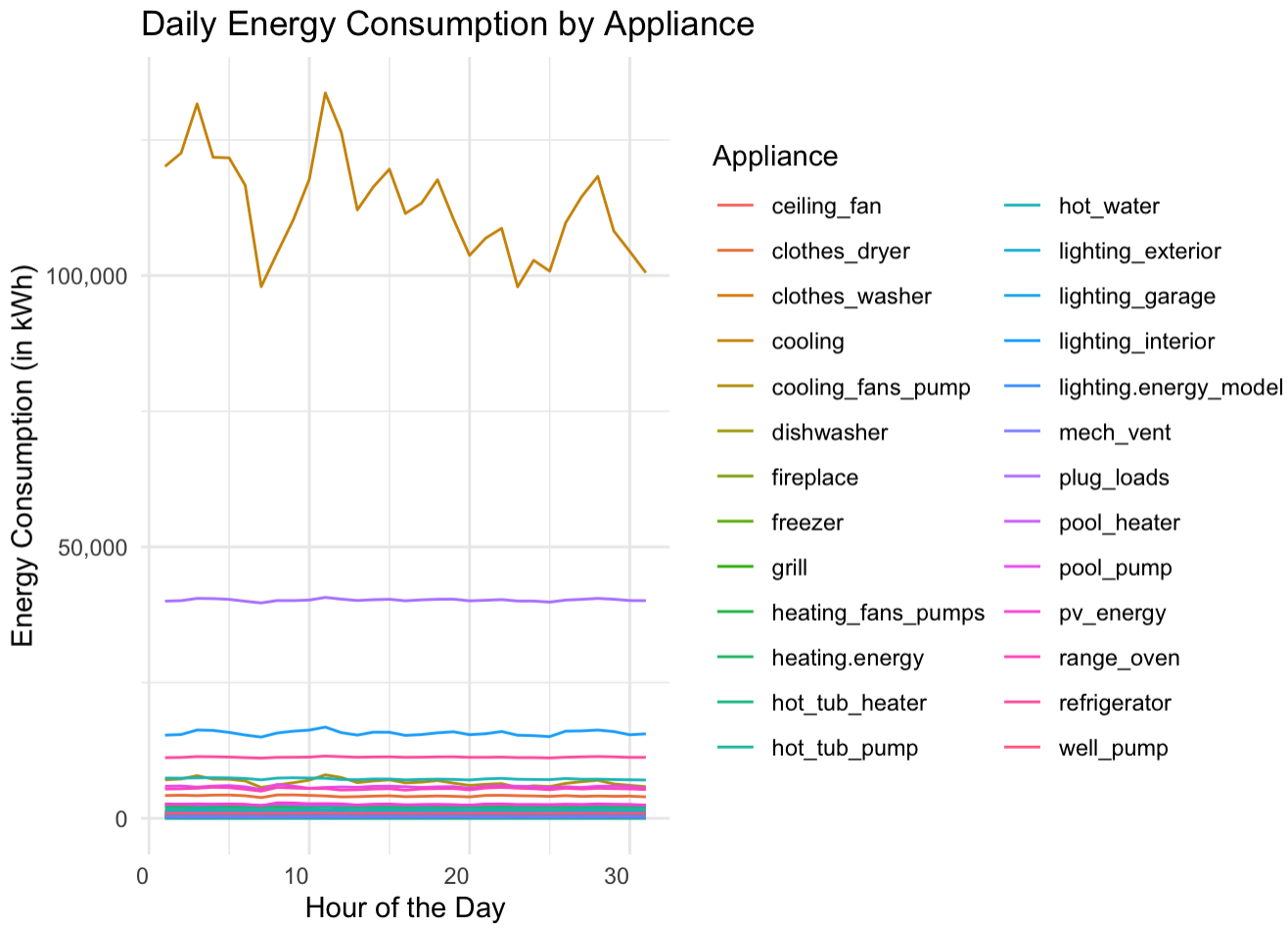
```
final_data$Day <- as.POSIXlt(final_data$time)$mday  # Extract day
final_data$Hour <- as.POSIXlt(final_data$time)$hour # Extract hour
final_data <- final_data[,-27]

daily_aggregated_data <- final_data %>%
  group_by(Day) %>%
  summarise_all(sum)
daily_aggregated_data <- daily_aggregated_data[,-28]

library(ggplot2)
library(tidyr)

day_long_data <- daily_aggregated_data %>%
  gather(key = "Appliance", value = "EnergyConsumption", -Day)

day_long_aggregated_data <- day_long_data %>%
  group_by(Appliance) %>%
  summarise(TotalEnergy = sum(EnergyConsumption)) %>%
  arrange(desc(TotalEnergy))

colnames(day_long_data)
```

```
## [1] "Day"               "Appliance"         "EnergyConsumption"
```

```
day_long_aggregated_data_sum <- day_long_data %>%
  group_by(Day) %>%
  summarise(TotalEnergy = sum(EnergyConsumption)) %>%
  arrange(Day)

  ggplot(day_long_aggregated_data_sum, aes(x = Day, y = TotalEnergy)) +
  geom_line() +
  geom_point()+
  labs(title = "Total Energy Consumption by Day of the Month",
       x = "Day of the Month",
       y = "Total Energy Consumption (in kWh)") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 0, hjust = 1))
```




```
ggplot(day_long_data, aes(x = Day, y = EnergyConsumption, color = Appliance)) +
geom_line() +
labs(title = "Daily Energy Consumption by Appliance",
     x = "Hour of the Day",
     y = "Energy Consumption (in kWh)") +
theme_minimal() +
scale_y_continuous(labels = scales::comma) +
theme(axis.text.x = element_text(angle = 0, hjust = 1))
```




```
energy_file <- '/Users/subhiksha/Documents/IDS/ids/Final_energy_data.csv'
energy_data <- read_csv(energy_file)
```

```
## New names:
## Rows: 4248240 Columns: 11
## ── Column specification
## ──────────────────────────────────────────────────────── Delimiter: "," dbl
## (10): ...1, bldg_id, out.kitchen.energy_consumption, out.laundry.energy... dttm
## (1): time
## ℹ Use `spec()` to retrieve the full column specification for this data. ℹ
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## • `` -> `...1`
```

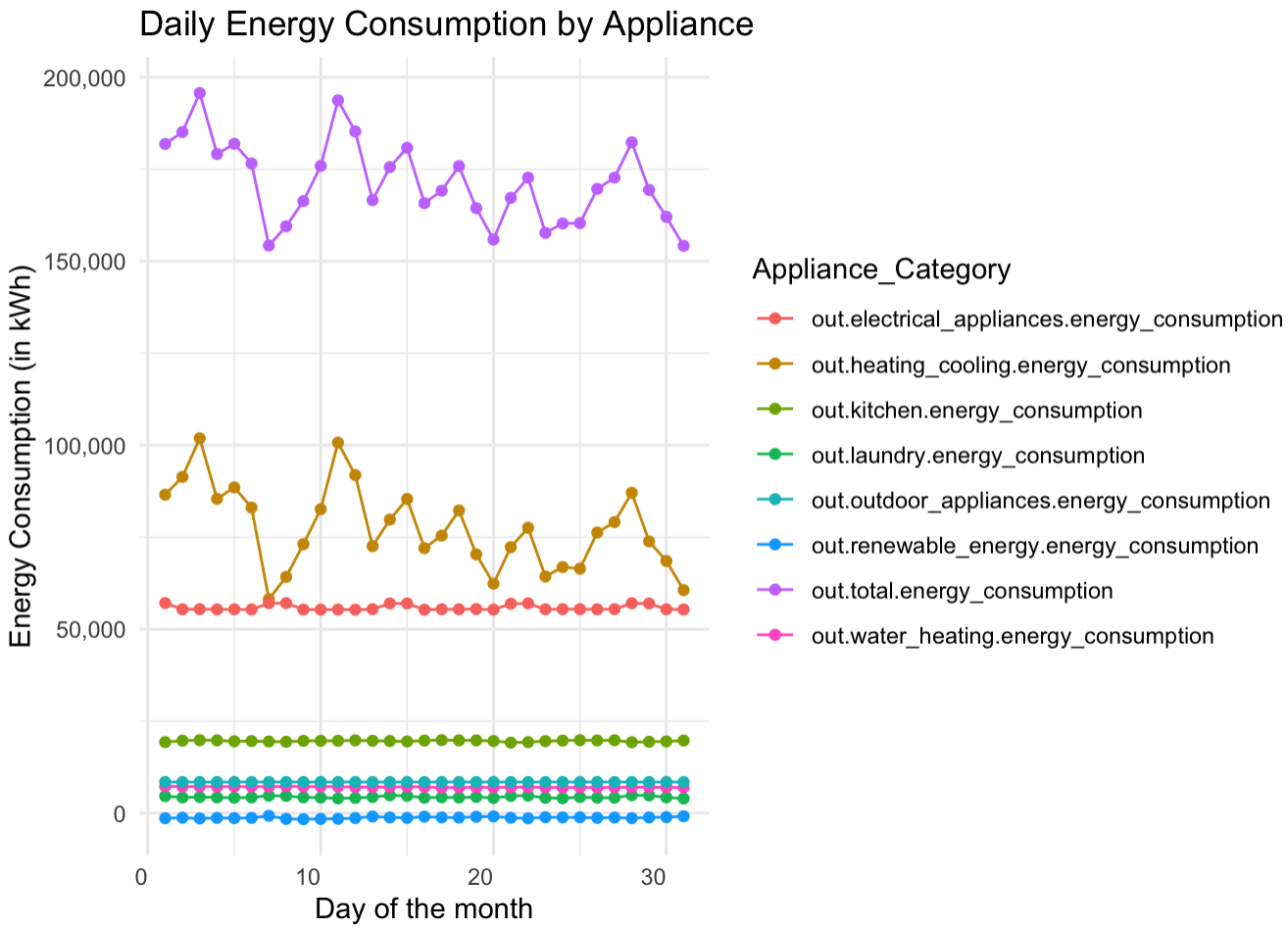
```
colnames(energy_data)
```

```
##  [1] "...1"
##  [2] "time"
##  [3] "bldg_id"
##  [4] "out.kitchen.energy_consumption"
##  [5] "out.laundry.energy_consumption"
##  [6] "out.heating_cooling.energy_consumption"
##  [7] "out.water_heating.energy_consumption"
##  [8] "out.electrical_appliances.energy_consumption"
##  [9] "out.outdoor_appliances.energy_consumption"
## [10] "out.renewable_energy.energy_consumption"
## [11] "out.total.energy_consumption"
```

```
energy_data<- energy_data[,-c(1)]
```

```
energy_data$Day <- as.POSIXlt(energy_data$time)$mday  # Extract day
energy_data$Hour <- as.POSIXlt(energy_data$time)$hour # Extract hour
energy_data2<- energy_data[,-1]
energy_data3<- energy_data2[,-1]
daily_aggregated_data1 <- energy_data3 %>%
  group_by(Day) %>%
  summarise_all(sum)

daily_aggregated_data1 <- daily_aggregated_data1[,-10]

day_long_data1 <- daily_aggregated_data1 %>%
  gather(key = "Appliance_Category", value = "EnergyConsumption", -Day)
```

```
ggplot(day_long_data1, aes(x = Day, y = EnergyConsumption, color = Appliance_Category))
+
  geom_line() +
  geom_point()+
  labs(title = "Daily Energy Consumption by Appliance",
       x = "Day of the month",
       y = "Energy Consumption (in kWh)") +
  theme_minimal() +
  scale_y_continuous(labels = scales::comma) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1))
```


Daily Energy Consumption by Appliance
Energy Consumption (in kWh)
200,000
150,000
100,000
50,000
0
0
10
20
30
Day of the month
Appliance_Category
out.electrical_appliances.energy_consumption
out.heating_cooling.energy_consumption
out.kitchen.energy_consumption
out.laundry.energy_consumption
out.outdoor_appliances.energy_consumption
out.renewable_energy.energy_consumption
out.total.energy_consumption
out.water_heating.energy_consumption

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats   1.0.0     ✔ readr     2.1.4
## ✔ ggplot2   3.4.4     ✔ stringr   1.5.0
## ✔ lubridate 1.9.3     ✔ tibble    3.2.1
## ✔ purrr     1.0.2     ✔ tidyr     1.3.0
```

```
## ── 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
```

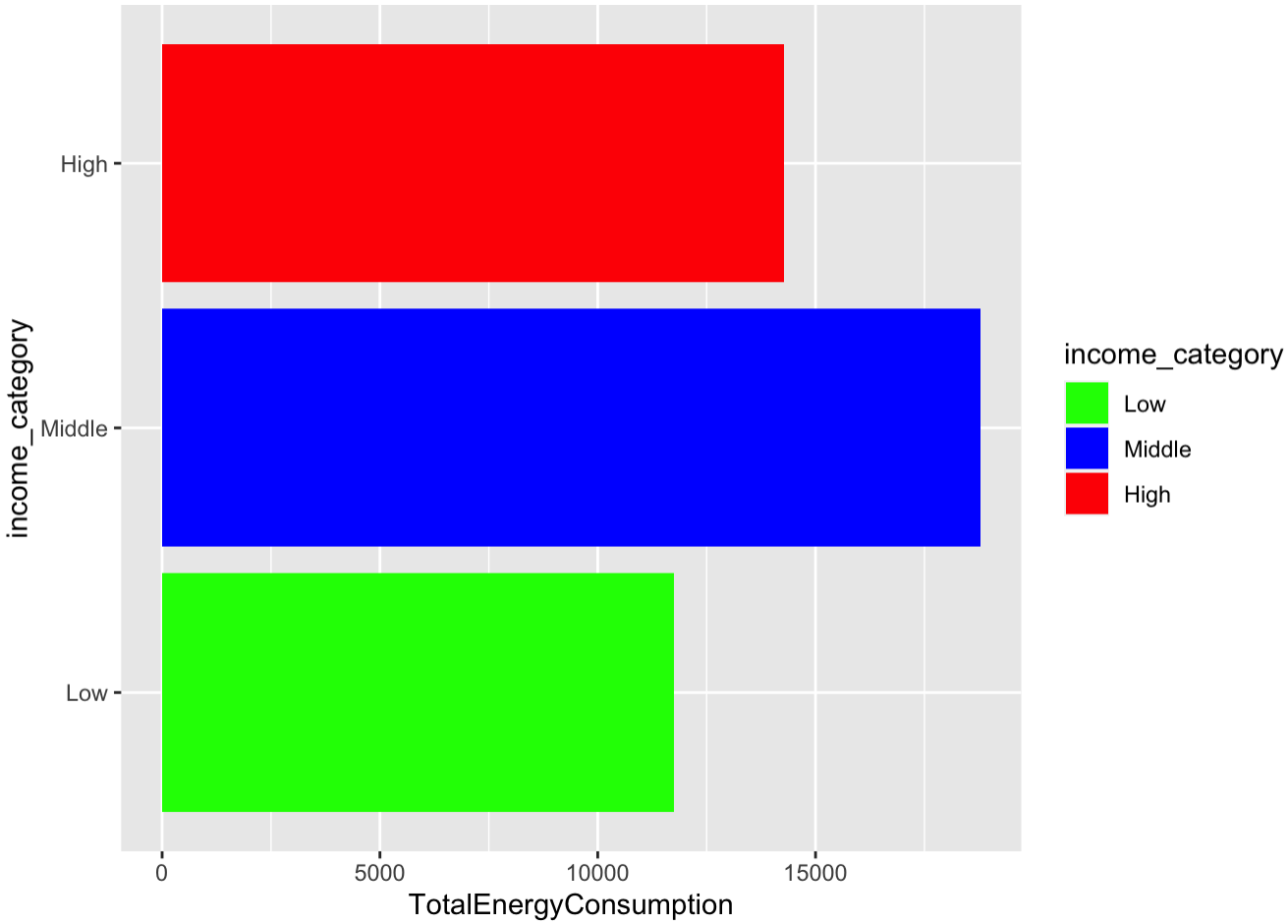
```
file_path <- '/Users/subhiksha/Downloads/Team2_Final_SEW_Ordinal_Modelling1.csv'
model <- read_csv(file_path)
```

```
## Rows: 34260 Columns: 94
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (11): in.county, time_split, in.county_and_puma, in.city, in.hvac_heatin...
## dbl (83): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
low_income_threshold <- 39999
middle_income_threshold <- 99999

# Categorize the 'in.income' into three groups
model <- model %>%
  mutate(income_category = case_when(
    in.income <= low_income_threshold ~ "Low",
    in.income > low_income_threshold & in.income <= middle_income_threshold ~ "Middle",
    in.income > middle_income_threshold ~ "High"
  ))

# Summing the total energy consumption for each income category
total_energy_consumption <- model %>%
  group_by(income_category) %>%
  summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))

# Print the result
print(total_energy_consumption)
```

```
## # A tibble: 3 × 2
##   income_category TotalEnergyConsumption
##   <chr>                            <dbl>
## 1 High                           14279.
## 2 Low                            11753.
## 3 Middle                         18785.
```

```
# Reorder the income categories
total_energy_consumption$income_category <- factor(total_energy_consumption$income_categ
ory, levels = c("Low", "Middle", "High"))

# Plot the bar graph with customized color and ordered y-axis
 ggplot(total_energy_consumption, aes(x = TotalEnergyConsumption, y = income_category,fi
ll=income_category)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("green", "blue", "red"))
```



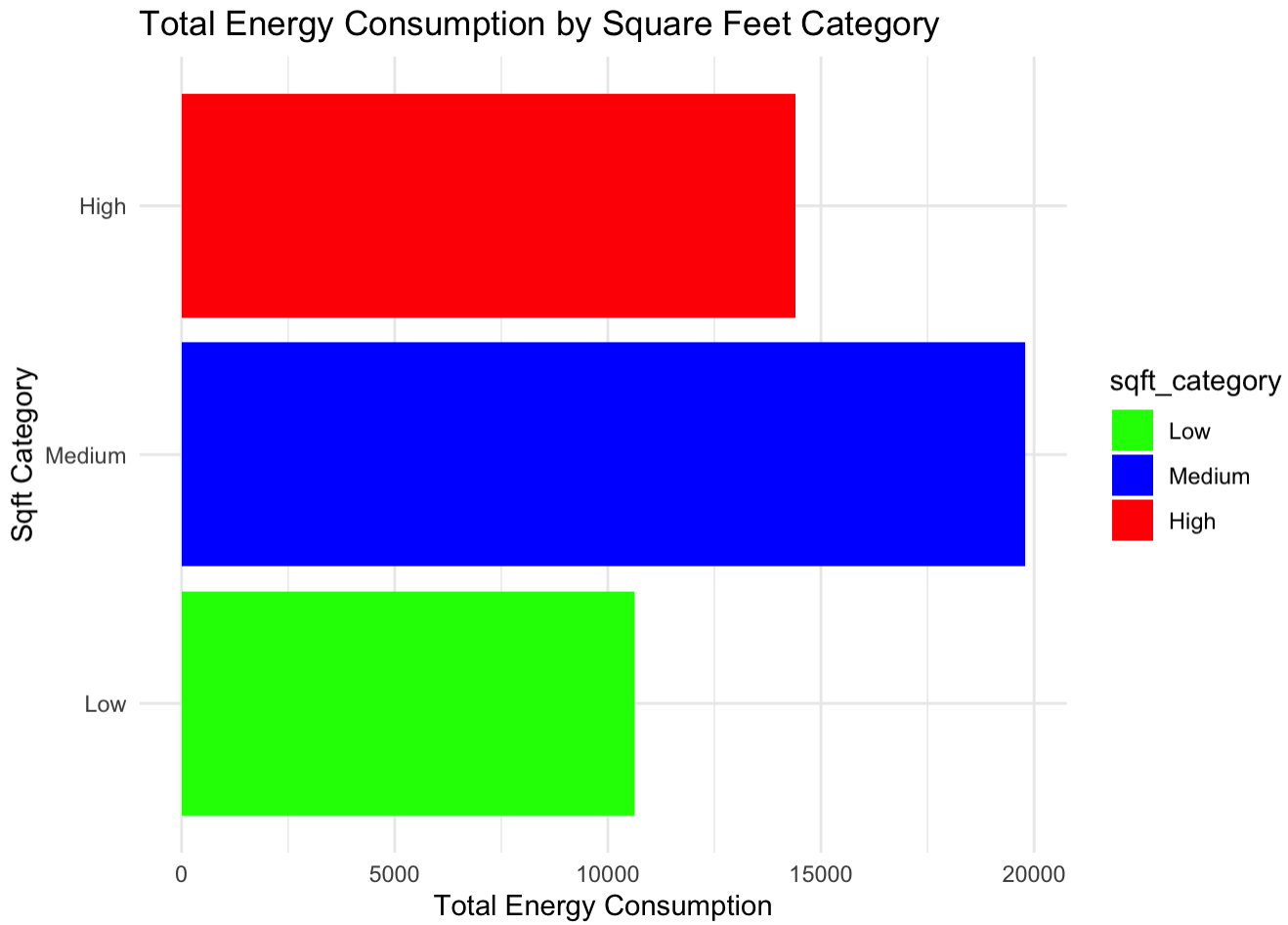

```
  labs(title = "Total Energy Consumption by Income Category",
         x = "Total Energy Consumption" ,
         y = "Income Category") +
  theme_minimal()
```

```
## NULL
```

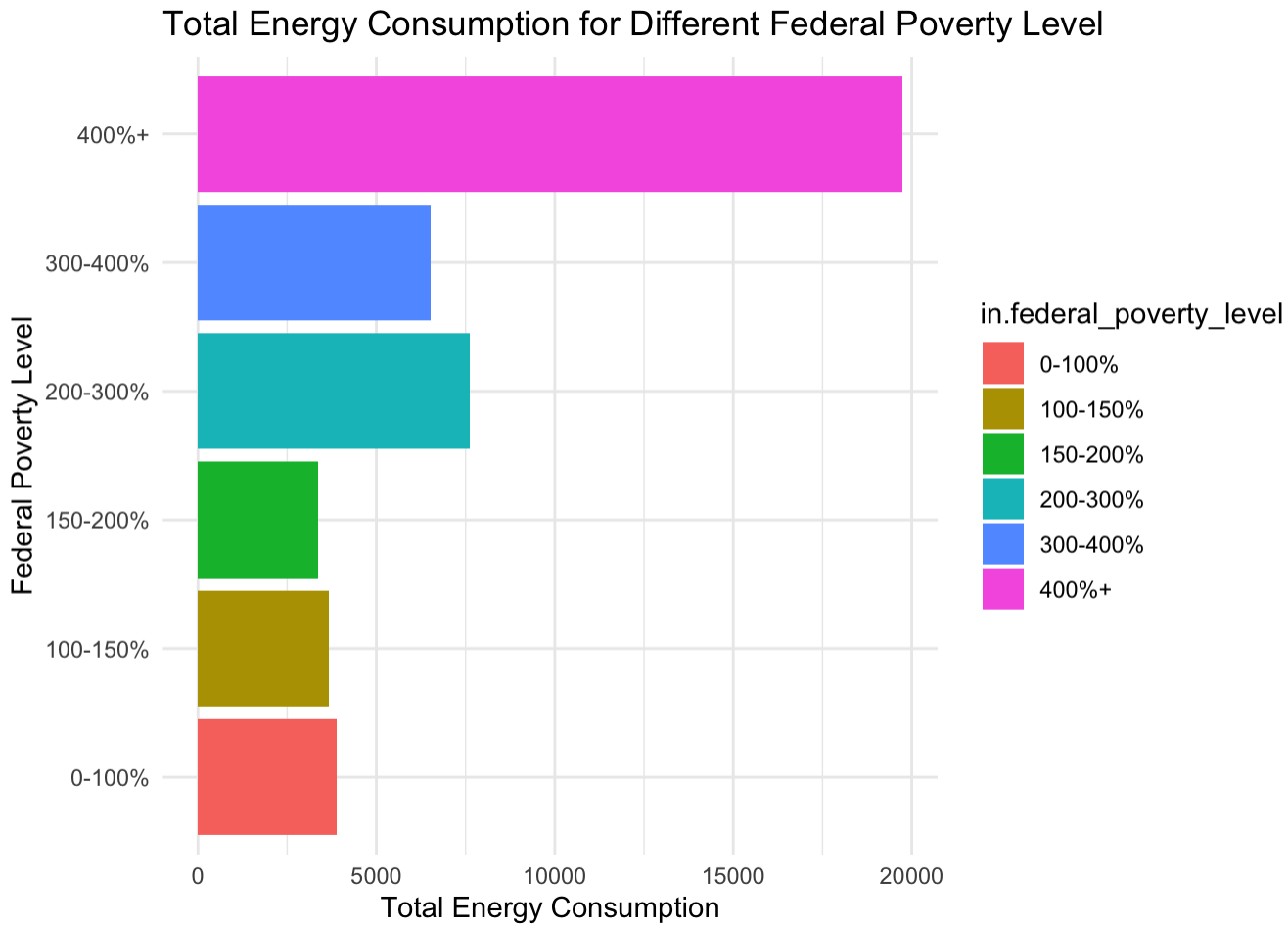
```
##############################################################################################
##########

## in.sqft column

file <- '/Users/subhiksha/Downloads/Final_Merged_For_Shiny_Geo.csv'
library(tidyverse)
peak_energy_file <- read_csv(file)
```

```
## Rows: 34260 Columns: 96
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## chr (70): in.county, time_split, in.county_and_puma, in.building_america_cli...
## dbl (26): bldg_id, time_split_numeric, in.sqft, in.bedrooms, in.cooling_setp...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

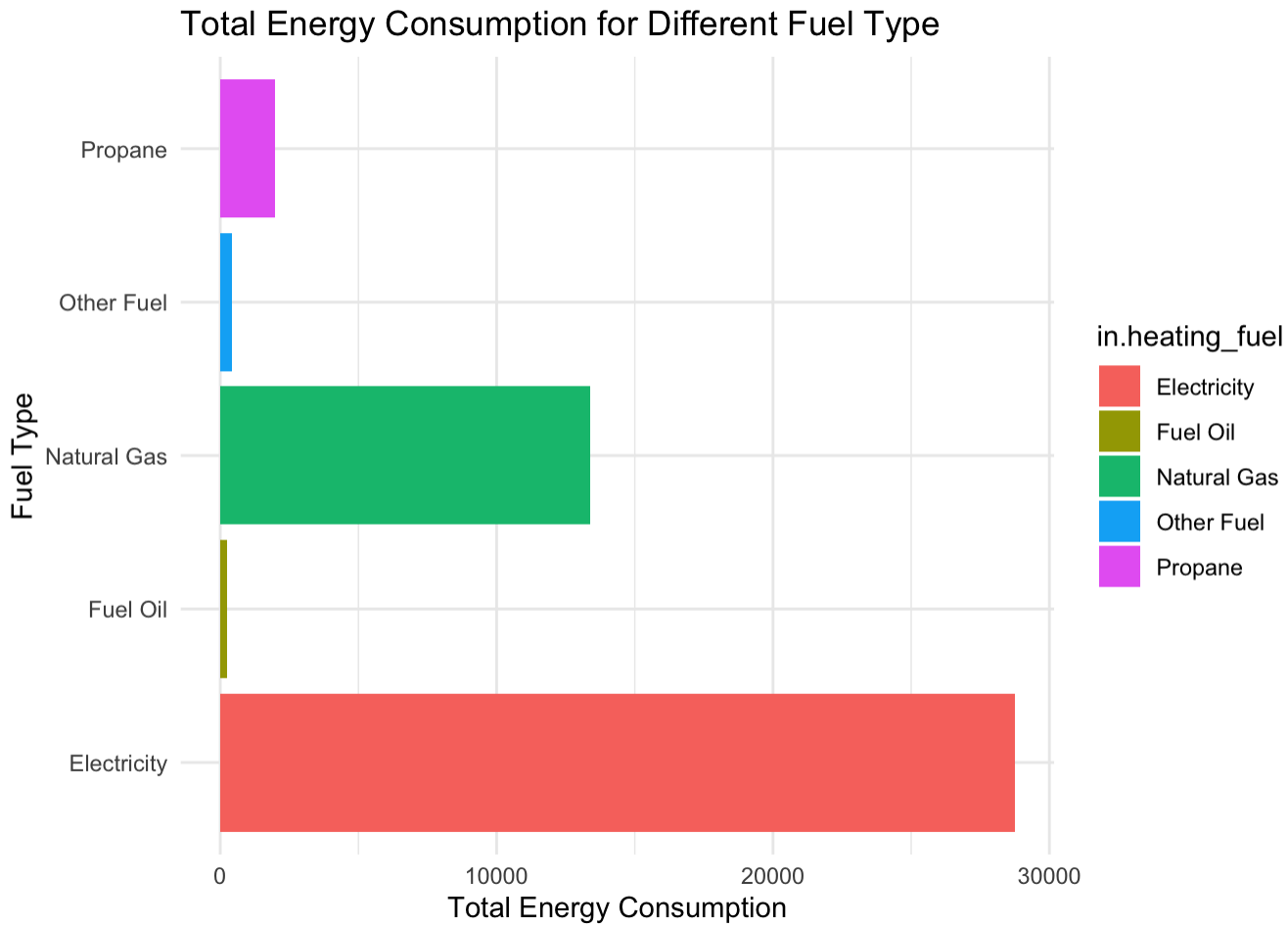
```
quantiles <- quantile(peak_energy_file$in.sqft, probs = c(0, 0.25, 0.75, 1), na.rm = TRU
E)

# Create the sqft_category column
peak_energy_file$sqft_category <- cut(peak_energy_file$in.sqft,
                          breaks = quantiles,
                          labels = c("Low", "Medium", "High"),
                          include.lowest = TRUE)

total_energy_by_sqft <- peak_energy_file %>%
  group_by(sqft_category) %>%
  summarize(TotalEnergy = sum(out.total_energy_consumption, na.rm = TRUE)) %>%
  ungroup()

  ggplot(total_energy_by_sqft, aes(x = TotalEnergy, y = sqft_category,fill = sqft_catego
ry)) +
  geom_bar(stat = "identity") +
  scale_fill_manual(values = c("green", "blue", "red")) +
  labs(title = "Total Energy Consumption by Square Feet Category",
  x = "Total Energy Consumption" ,
  y = "Sqft Category") +
  theme_minimal()
```




```
##########################################################################################
#################

#in.federal_poverty_level

total_energy_bypoverty <- peak_energy_file %>%
  group_by(in.federal_poverty_level) %>%
  summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))

  ggplot(total_energy_bypoverty, aes(x = TotalEnergyConsumption, y = in.federal_poverty_
level ,fill = in.federal_poverty_level)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Energy Consumption for Different Federal Poverty Level",
       x = "Total Energy Consumption" ,
       y = "Federal Poverty Level") +
  theme_minimal()
```




```
##############################################################################################
####################

##in.heating_fuel


total_energy_by_Heatingfuel <- peak_energy_file %>%
  group_by(in.heating_fuel) %>%
  summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))

total_energy_by_Heatingfuel <- total_energy_by_Heatingfuel[total_energy_by_Heatingfuel$i
n.heating_fuel != "None", ]

 ggplot(total_energy_by_Heatingfuel, aes(x = TotalEnergyConsumption, y = in.heating_fuel
,fill = in.heating_fuel)) +
  geom_bar(stat = "identity") +
  labs(title = "Total Energy Consumption for Different Fuel Type",
       x = "Total Energy Consumption" ,
       y = "Fuel Type") +
  theme_minimal()
```


Total Energy Consumption for Different Fuel Type
Propane
Other Fuel
Natural Gas
Fuel Oil
Electricity
Fuel Type
0
10000
20000
30000
Total Energy Consumption
in.heating_fuel
Electricity
Fuel Oil
Natural Gas
Other Fuel
Propane

```
##############################################################################################################
```

Untitled

2023-12-06

```
file <- '/Users/subhiksha/Downloads/Final_Merged_For_Shiny_Geo.csv'
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ 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
```

```
peak_energy_file <- read_csv(file)
```

```
## Rows: 34260 Columns: 96
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (70): in.county, time_split, in.county_and_puma, in.building_america_cli...
## dbl (26): bldg_id, time_split_numeric, in.sqft, in.bedrooms, in.cooling_setp...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
nrow(peak_energy_file)
```

```
## [1] 34260
```

```
colnames(peak_energy_file)
```

```
##  [1] "bldg_id"
##  [2] "time_split_numeric"
##  [3] "in.county"
##  [4] "time_split"
##  [5] "in.county_and_puma"
##  [6] "in.sqft"
##  [7] "in.bedrooms"
##  [8] "in.building_america_climate_zone"
##  [9] "in.ceiling_fan"
## [10] "in.city"
## [11] "in.clothes_dryer"
## [12] "in.clothes_washer"
## [13] "in.cooking_range"
## [14] "in.cooling_setpoint"
## [15] "in.cooling_setpoint_offset_magnitude"
## [16] "in.dishwasher"
## [17] "in.federal_poverty_level"
## [18] "in.geometry_attic_type"
## [19] "in.geometry_floor_area"
## [20] "in.geometry_floor_area_bin"
## [21] "in.geometry_garage"
## [22] "in.heating_fuel"
## [23] "in.heating_setpoint"
## [24] "in.heating_setpoint_offset_magnitude"
## [25] "in.hot_water_fixtures"
## [26] "in.hvac_cooling_efficiency"
## [27] "in.hvac_cooling_partial_space_conditioning"
## [28] "in.hvac_cooling_type"
## [29] "in.hvac_has_ducts"
## [30] "in.hvac_has_zonal_electric_heating"
## [31] "in.hvac_heating_efficiency"
## [32] "in.hvac_heating_type"
## [33] "in.hvac_heating_type_and_fuel"
## [34] "in.income"
## [35] "in.income_recs_2015"
## [36] "in.income_recs_2020"
## [37] "in.infiltration"
## [38] "in.insulation_ceiling"
## [39] "in.insulation_floor"
## [40] "in.insulation_foundation_wall"
## [41] "in.insulation_rim_joist"
## [42] "in.insulation_roof"
## [43] "in.insulation_slab"
## [44] "in.insulation_wall"
## [45] "in.lighting"
## [46] "in.misc_extra_refrigerator"
## [47] "in.misc_freezer"
## [48] "in.misc_gas_fireplace"
## [49] "in.misc_gas_grill"
## [50] "in.misc_gas_lighting"
## [51] "in.misc_hot_tub_spa"
## [52] "in.misc_pool"
```

```
## [53] "in.misc_pool_heater"
## [54] "in.misc_pool_pump"
## [55] "in.misc_well_pump"
## [56] "in.natural_ventilation"
## [57] "in.occupants"
## [58] "in.orientation"
## [59] "in.plug_load_diversity"
## [60] "in.refrigerator"
## [61] "in.roof_material"
## [62] "in.tenure"
## [63] "in.usage_level"
## [64] "in.vacancy_status"
## [65] "in.vintage"
## [66] "in.vintage_acs"
## [67] "in.water_heater_efficiency"
## [68] "in.water_heater_fuel"
## [69] "in.weather_file_city"
## [70] "in.weather_file_latitude"
## [71] "in.weather_file_longitude"
## [72] "in.window_areas"
## [73] "in.windows"
## [74] "upgrade.insulation_roof"
## [75] "upgrade.water_heater_efficiency"
## [76] "upgrade.clothes_dryer"
## [77] "upgrade.hvac_heating_efficiency"
## [78] "upgrade.cooking_range"
## [79] "out.kitchen_energy_consumption"
## [80] "out.laundry_energy_consumption"
## [81] "out.heating_cooling_energy_consumption"
## [82] "out.water_heating_energy_consumption"
## [83] "out.electrical_appliances_energy_consumption"
## [84] "out.outdoor_appliances_energy_consumption"
## [85] "out.renewable_energy_energy_consumption"
## [86] "out.total_energy_consumption"
## [87] "time_range.x"
## [88] "Dry_Bulb_Temperature_C"
## [89] "Relative_Humidity"
## [90] "Wind_Speed_m_s"
## [91] "Wind_Direction_Deg"
## [92] "Global_Horizontal_Radiation_W_m2"
## [93] "Direct_Normal_Radiation_W_m2"
## [94] "Diffuse_Horizontal_Radiation_W_m2"
## [95] "Next_year_Pred"
## [96] "Change_in_energy"
```

```
summary(peak_energy_file$out.total_energy_consumption)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.05835 0.83173 1.18313 1.30814 1.63109 8.32710
```

```
summary(peak_energy_file$Next_year_Pred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.08724 0.94739 1.32063 1.45717 1.80063 6.90360
```

```
#unique(peak_energy_file$in.county)
```

Including Plots

You can also embed plots, for example:

```
library(dplyr)
library(ggplot2)
library(readr)
# Data aggregation
grouped_data <- peak_energy_file %>%
  group_by(time_split, in.county) %>%
  summarize(Total_Energy_Consumption = sum(out.total_energy_consumption, na.rm = TRUE),
            Avg_Latitude = mean(in.weather_file_latitude, na.rm = TRUE),
            Avg_Longitude = mean(in.weather_file_longitude, na.rm = TRUE),
            City_Name = first(in.weather_file_city[!is.na(in.weather_file_city)])) %>%
  ungroup()
```

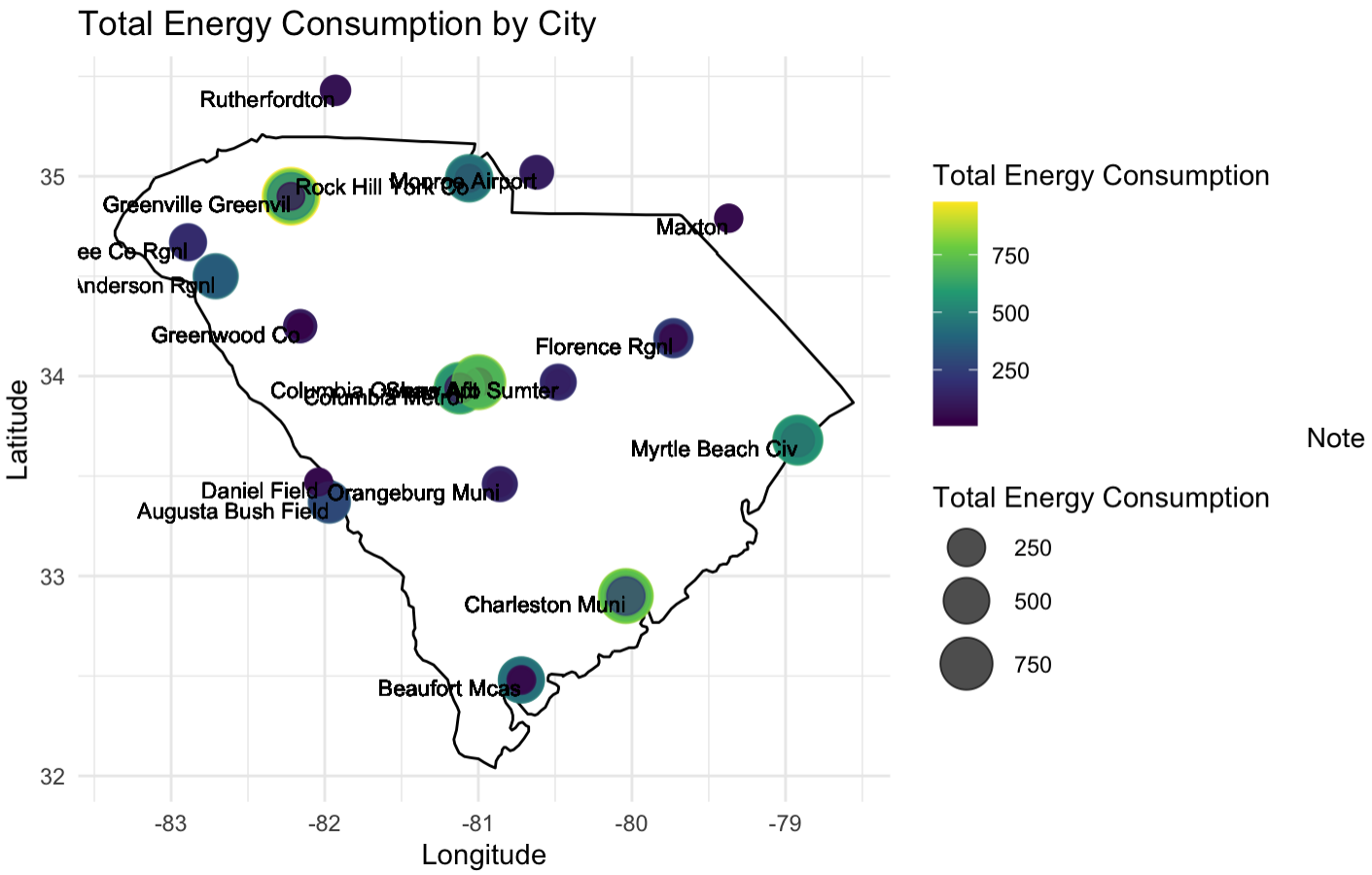
```
## `summarise()` has grouped output by 'time_split'. You can override using the
## `.groups` argument.
```

```
city_labels <- data.frame(label = grouped_data$City_Name,
                          lon = grouped_data$Avg_Longitude,
                          lat = grouped_data$Avg_Latitude)
```

```
sc_map <- map_data("state", region = "south carolina")
```

```
#install.packages("ggrepel")
library(ggrepel)
  ggplot() +
  geom_polygon(data = sc_map, aes(x = long, y = lat, group = group), fill = "white", col
or = "black") +
  geom_point(data = grouped_data, aes(x = Avg_Longitude, y = Avg_Latitude, size = Total_
Energy_Consumption, color = Total_Energy_Consumption), alpha = 0.7) +
  scale_size_continuous(range = c(1, 4))+
  geom_text(data = city_labels, aes(label = label, x = lon, y = lat), size = 3, hjust =
1, vjust = 1) +
  scale_color_viridis_c() +
  scale_size(range = c(3, 10)) +
  labs(title = "Total Energy Consumption by City",
       x = "Longitude",
       y = "Latitude",
       size = "Total Energy Consumption",
       color = "Total Energy Consumption") +
  theme_minimal() +
  coord_fixed(1.3)  # Hides the legend for size
```

```
## Scale for size is already present.
## Adding another scale for size, which will replace the existing scale.
```



that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Correlation

2023-12-03

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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
```

```
Final_team2_Modelling_SEW <- read_csv("C:/Users/Soundarya Ravi/Desktop/Team2_Final_SEW_Ordinal_Modelling1_cleaned.csv")
```

```
## Rows: 34260 Columns: 94
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (11): in.county, time_split, in.county_and_puma, in.city, in.hvac_heatin...
## dbl (83): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# Assuming your dataframe is named Final_team2_Modelling_SEW

# Select numeric columns only
numeric_columns <- Final_team2_Modelling_SEW[sapply(Final_team2_Modelling_SEW, is.numeric)]

# Calculate correlation matrix
correlation_matrix <- cor(numeric_columns)
```

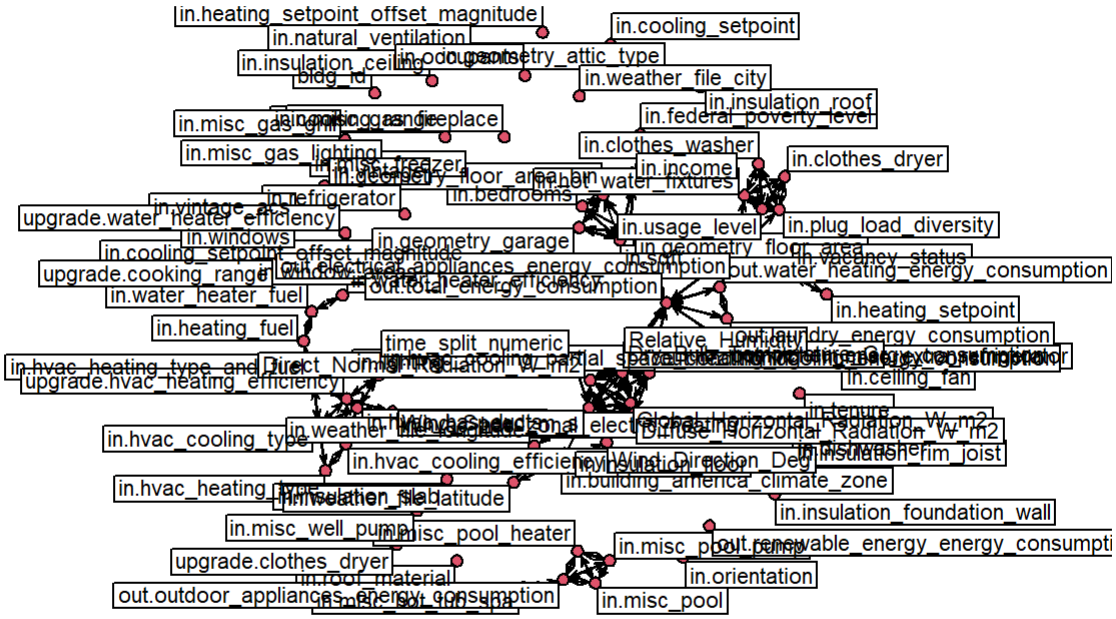
```
## Warning in cor(numeric_columns): the standard deviation is zero
```

```r
# Print the correlation matrix
#print(correlation_matrix)

threshold <- 0.4

# Filter correlation matrix for values above the threshold
filtered_correlation_matrix <- correlation_matrix * (abs(correlation_matrix) > threshold)
```

```r
# Install network package if not installed
# install.packages("network")

# Load the network package
library(network)
```

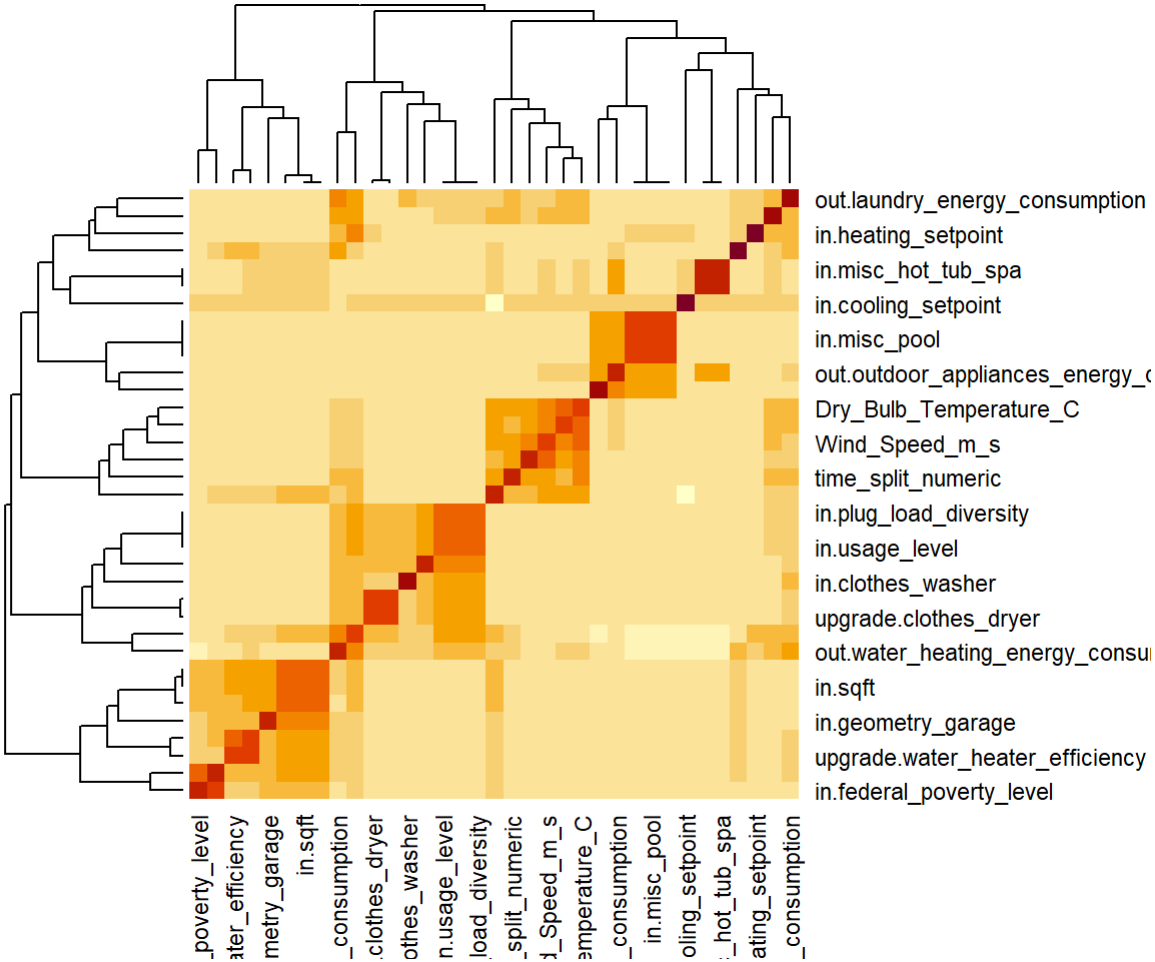
```
##
## 'network' 1.18.1 (2023-01-24), part of the Statnet Project
## * 'news(package="network")' for changes since last version
## * 'citation("network")' for citation information
## * 'https://statnet.org' for help, support, and other information
```

```r
# Create a network object
net <- as.network(filtered_correlation_matrix)

# Plot the network
plot(net, displaylabels = TRUE, boxed.labels = TRUE, label.cex = 0.7)
```

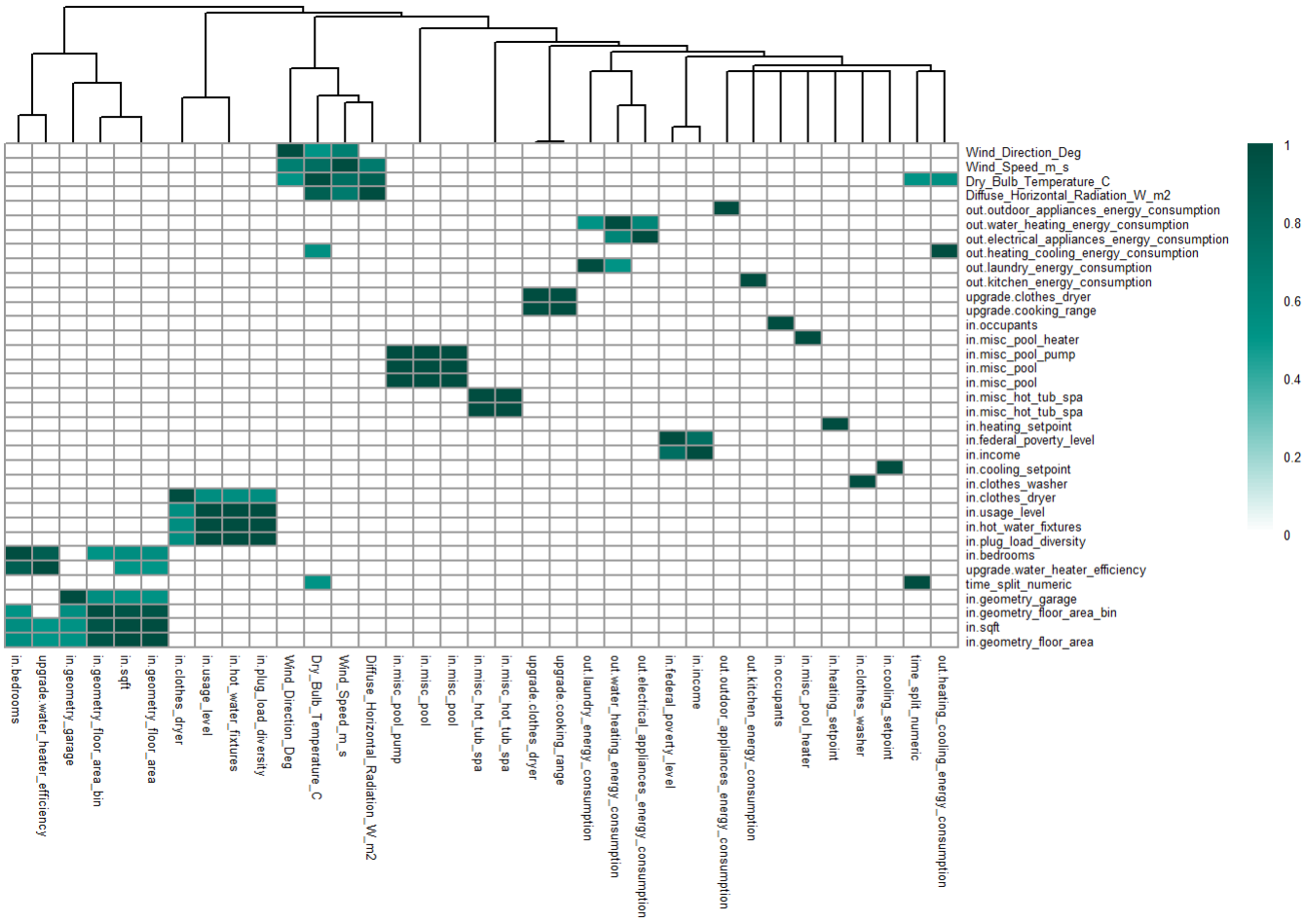

in.heating_setpoint_offset_magnitude
in.cooling_setpoint
in.natural_ventilation
in.insulation_ceiling
in.weather_file_city
bldg_id
in.insulation_roof
in.federal_poverty_level
in.misc_gas_fireplace
in.misc_gas_lighting
in.clothes_washer
in.clothes_dryer
in.income
in.bedrooms
in.plug_load_diversity
upgrade.water_heater_efficiency
in.windows
in.geometry_garage
in.geometry_floor_area
in.cooling_setpoint_offset_magnitude
out.water_heating_energy_consumption
upgrade.cooking_range
in.water_heater_fuel
in.heating_setpoint
in.heating_fuel
out.laundry_energy_consumption
time_split_numeric
upgrade.hvac_heating_efficiency
in.ceiling_fan
in.tenure
in.hvac_cooling_type
in.hvac_cooling_efficiency
in.insulation_floor
in.building_america_climate_zone
in.insulation_foundation_wall
in.misc_well_pump
in.misc_pool_heater
upgrade.clothes_dryer
in.orientation
in.misc_pool

```r
# Select specific columns for correlation
selected_columns <- Final_team2_Modelling_SEW[, c(
  "time_split_numeric",
  "in.sqft",
  "in.bedrooms",
  "in.clothes_dryer",
  "in.clothes_washer",
  "in.cooling_setpoint",
  "in.federal_poverty_level",
  "in.geometry_floor_area",
  "in.geometry_floor_area_bin",
  "in.geometry_garage",
  "in.heating_setpoint",
  "in.hot_water_fixtures",
  "in.income",
  "in.misc_hot_tub_spa",
  "in.misc_pool",
  "in.misc_hot_tub_spa",
  "in.misc_pool",
  "in.misc_pool_heater",
  "in.misc_pool_pump",
  "in.occupants",
  "in.plug_load_diversity",
  "in.usage_level",
  "upgrade.water_heater_efficiency",
  "upgrade.clothes_dryer",
  "upgrade.cooking_range",
  "out.kitchen_energy_consumption",
  "out.laundry_energy_consumption",
  "out.heating_cooling_energy_consumption",
  "out.water_heating_energy_consumption",
  "out.electrical_appliances_energy_consumption",
  "out.outdoor_appliances_energy_consumption",
  "Dry_Bulb_Temperature_C",
  "Wind_Speed_m_s",
  "Wind_Direction_Deg",
  "Diffuse_Horizontal_Radiation_W_m2"
)]
```

```r
# Impute missing values with the mean of each column
selected_columns <- apply(selected_columns, 2, function(x) ifelse(is.na(x), mean(x, na.rm = TRUE), x))

# Calculate correlation matrix for the selected columns
correlation_matrix <- cor(selected_columns)

# Create heatmap
heatmap(correlation_matrix)
```



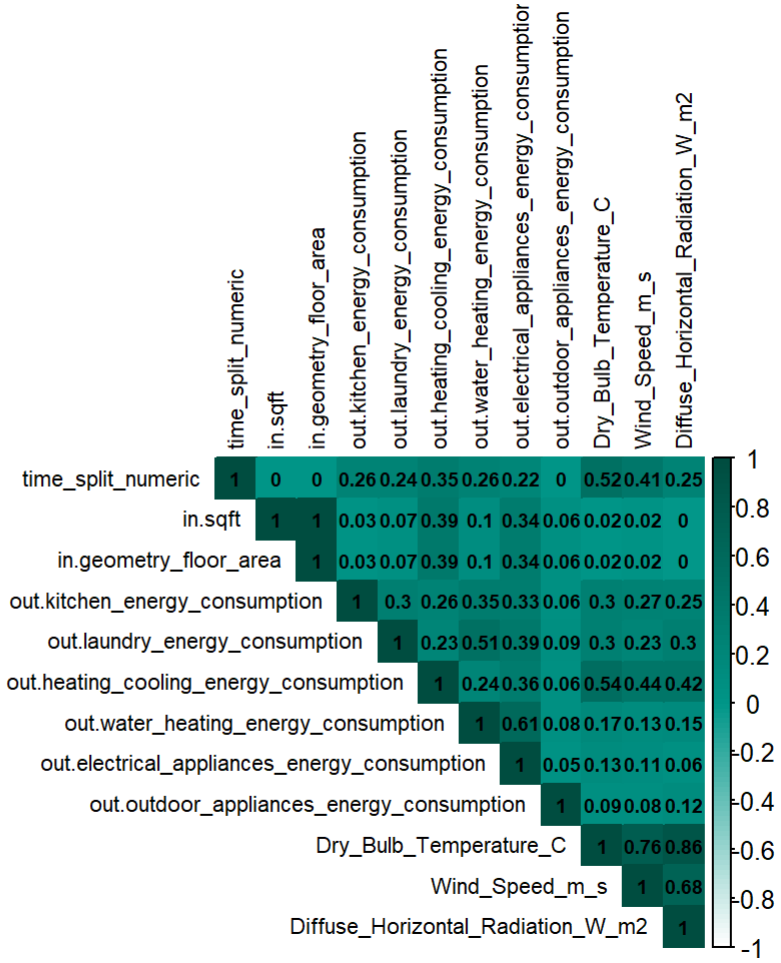
```r
# Install pheatmap package if not installed
# install.packages("pheatmap")

# Load the pheatmap package
library(pheatmap)

# Set the threshold
threshold <- 0.5

# Filter correlation matrix for values above the threshold
filtered_correlation_matrix <- correlation_matrix * (abs(correlation_matrix) > threshold)

# Perform hierarchical clustering on rows
row_order <- hclust(as.dist(1 - filtered_correlation_matrix))$order

# Set the size of the heatmap
heatmap_width <- 30 # Set the desired width
heatmap_height <- 100  # Set the desired height

# Create a dendrogram-based heatmap with custom row order and expanded size
pheatmap(
  filtered_correlation_matrix[row_order, ],
  cluster_rows = FALSE,  # Use custom row order
  cluster_cols = TRUE,
  color = colorRampPalette(c("#FFFFFF", "#009688", "#004D40"))(100),
  fontsize = 5,
  width = heatmap_width,
  height = heatmap_height
)
```


Wind_Direction_Deg
Wind_Speed_m_s
Dry_Bulb_Temperature_C
Diffuse_Horizontal_Radiation_W_m2
out.outdoor_appliances_energy_consumption
out.water_heating_energy_consumption
out.electrical_appliances_energy_consumption
out.heating_cooling_energy_consumption
out.laundry_energy_consumption
out.kitchen_energy_consumption
upgrade.clothes_dryer
upgrade.cooking_range
in.occupants
in.misc_pool_heater
in.misc_pool_pump
in.misc_pool
in.misc_pool
in.misc_hot_tub_spa
in.misc_hot_tub_spa
in.heating_setpoint
in.federal_poverty_level
in.income
in.cooling_setpoint
in.clothes_washer
in.clothes_dryer
in.usage_level
in.hot_water_fixtures
in.plug_load_diversity
in.bedrooms
upgrade.water_heater_efficiency
time_split_numeric
in.geometry_garage
in.geometry_floor_area_bin
in.sqft
in.geometry_floor_area
1
0.8
0.6
0.4
0.2
0
in.bedrooms
upgrade.water_heater_efficiency
in.geometry_garage
in.geometry_floor_area_bin
in.sqft
in.geometry_floor_area
in.clothes_dryer
in.usage_level
in.hot_water_fixtures
in.plug_load_diversity
Wind_Direction_Deg
Dry_Bulb_Temperature_C
Wind_Speed_m_s
Diffuse_Horizontal_Radiation_W_m2
in.misc_pool_pump
in.misc_pool
in.misc_pool
in.misc_hot_tub_spa
in.misc_hot_tub_spa
upgrade.clothes_dryer
upgrade.cooking_range
out.laundry_energy_consumption
out.water_heating_energy_consumption
out.electrical_appliances_energy_consumption
in.federal_poverty_level
in.income
out.outdoor_appliances_energy_consumption
out.kitchen_energy_consumption
in.occupants
in.misc_pool_heater
in.heating_setpoint
in.clothes_washer
in.cooling_setpoint
time_split_numeric
out.heating_cooling_energy_consumption

```r
# Assuming your dataframe is named Final_team2_Modelling_SEW

# Select the specified fields
selected_fields <- c(
  "time_split_numeric",
  "in.sqft",
  "in.geometry_floor_area",
  "out.kitchen_energy_consumption",
  "out.laundry_energy_consumption",
  "out.heating_cooling_energy_consumption",
  "out.water_heating_energy_consumption",
  "out.electrical_appliances_energy_consumption",
  "out.outdoor_appliances_energy_consumption",
  "Dry_Bulb_Temperature_C",
  "Wind_Speed_m_s",
  "Diffuse_Horizontal_Radiation_W_m2"
)

# Create a subset of the dataframe with selected fields
selected_dataframe <- Final_team2_Modelling_SEW[selected_fields]

# Calculate the correlation matrix
correlation_matrix <- cor(selected_dataframe)

# Plot the correlation matrix using corrplot
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
corrplot(
  correlation_matrix,
  method = "color",           # Color-coded plot
  type = "upper",             # Show only upper triangle to avoid redundancy
  tl.cex = 0.7,                # Text label size
  tl.col = "black",            # Text label color
  col = colorRampPalette(c("#FFFFFF", "#009688", "#004D40"))(100),  # Green color palette
  addCoef.col = "black",       # Coefficient color
  number.cex = 0.6            # Correlation coefficient text size
)
```



BUILDING THE MODELS

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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 becom
e errors
```

```
# Modeling_df_without_ordinal <- read_csv("C:/Users/Soundarya Ravi/Desktop/Merged_data.csv")

Model_new <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/final_modeling_df.csv")
```

```
## Rows: 34260 Columns: 73
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (73): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
Modeling_ordinality <- Model_new

# Assuming 'processed_data' is your data frame
Modeling_ordinality <- Modeling_ordinality[, colSums(is.na(Modeling_ordinality)) == 0]

# Now 'processed_data' contains only columns without any missing values
```

```
# setwd("C:/Users/Soundarya Ravi/Desktop/Shiny/")
#
# write.csv(Modeling_ordinality, file ="Ordinality_Final_Merge.csv",row.names=FALSE)
```

```
# Install and load necessary packages
# install.packages(c("dplyr", "caret"))

# Load packages
library(dplyr)
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
index <- createDataPartition(Modeling_ordinality$out.total_energy_consumption, p = 0.8, list = F
ALSE)
train_data <- Modeling_ordinality[index, ]
test_data <- Modeling_ordinality[-index, ]

# Function to handle character columns and build the linear regression model
build_lm_model <- function(data) {
  # Identify numeric and character columns
  numeric_cols <- sapply(data, is.numeric)
  char_cols <- sapply(data, is.character)

  # Convert character columns to factors
  data[, char_cols] <- lapply(data[, char_cols], as.factor)

  # Build linear regression model
  lm_model <- lm(out.total_energy_consumption ~ ., data = data[, numeric_cols | char_cols])

  return(lm_model)
}

# Build the linear regression model on the training set
lm_model <- build_lm_model(train_data)

# Make predictions on the test set
predictions <- predict(lm_model, newdata = test_data)
#print(predictions)
# Evaluate the model, e.g., calculate RMSE (Root Mean Squared Error) or other metrics
# ...

# View summary of the linear regression model
summary(lm_model)
```

```
##
## Call:
## lm(formula = out.total_energy_consumption ~ ., data = data[,
##     numeric_cols | char_cols])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.4894 -0.2244 -0.0340  0.1670  6.0305
##
## Coefficients: (6 not defined because of singularities)
##                                           Estimate Std. Error t value
## (Intercept)                             -2.696e+00  5.925e-01  -4.549
## bldg_id                                 -2.636e-08  1.482e-08  -1.779
## in.sqft                                  1.022e-01  4.625e-03  22.100
## in.bedrooms                              2.654e-02  3.385e-03   7.839
## in.building_america_climate_zone        -2.443e-02  1.224e-02  -1.997
## in.ceiling_fan                           4.219e-03  2.736e-03   1.542
## in.clothes_dryer                        -4.149e-03  2.378e-03  -1.745
## in.clothes_washer                        7.209e-03  1.581e-03   4.558
## in.cooking_range                         1.269e-03  1.360e-03   0.933
## in.cooling_setpoint                     -7.203e-02  1.200e-03 -60.030
## in.cooling_setpoint_offset_magnitude     1.262e-02  2.726e-03   4.628
## in.dishwasher                            4.971e-03  1.112e-03   4.472
## in.federal_poverty_level                -4.380e-02  2.176e-03 -20.131
## in.geometry_attic_type                  -2.107e-01  3.800e-02  -5.544
## in.geometry_floor_area                          NA         NA      NA
## in.geometry_floor_area_bin               1.540e-01  8.845e-03  17.412
## in.geometry_garage                      -3.212e-02  2.831e-03 -11.346
## in.heating_fuel                         -2.446e-03  6.400e-03  -0.382
## in.heating_setpoint                     -1.578e-02  1.278e-03 -12.347
## in.heating_setpoint_offset_magnitude     9.653e-03  2.281e-03   4.232
## in.hot_water_fixtures                    2.494e-01  4.421e-03  56.417
## in.hvac_cooling_efficiency               3.783e-03  1.947e-03   1.943
## in.hvac_cooling_partial_space_conditioning  3.896e-03  2.974e-03   1.310
## in.hvac_cooling_type                    -2.245e-02  7.287e-03  -3.081
## in.hvac_has_ducts                        1.496e-01  1.680e-02   8.902
## in.hvac_has_zonal_electric_heating       1.769e-02  1.201e-02   1.473
## in.hvac_heating_type                     1.039e-02  4.064e-03   2.556
## in.hvac_heating_type_and_fuel            3.201e-04  2.102e-03   0.152
## in.income                                1.473e-06  6.964e-08  21.144
## in.insulation_ceiling                   -2.497e-03  1.845e-03  -1.353
## in.insulation_floor                     -3.649e-02  3.669e-03  -9.945
## in.insulation_foundation_wall            1.055e-02  5.717e-03   1.845
## in.insulation_rim_joist                         NA         NA      NA
## in.insulation_roof                       2.534e-02  1.245e-02   2.034
## in.insulation_slab                      -2.607e-03  2.474e-03  -1.054
## in.lighting                             -8.671e-02  2.724e-03 -31.836
## in.misc_extra_refrigerator               1.475e-02  1.132e-03  13.033
## in.misc_freezer                          4.435e-02  4.818e-03   9.206
## in.misc_gas_fireplace                    7.350e-02  6.496e-03  11.316
## in.misc_gas_grill                        1.491e-02  6.872e-03   2.170
## in.misc_gas_lighting                     2.167e-02  9.821e-03   2.207
```

```
## in.misc_hot_tub_spa                           2.575e-01  6.780e-03  37.985
## in.misc_pool                                  2.447e-01  8.118e-03  30.137
## in.misc_pool_heater                           2.393e-01  9.727e-03  24.604
## in.misc_pool_pump                                    NA         NA      NA
## in.misc_well_pump                             4.274e-02  6.965e-03   6.137
## in.orientation                                5.344e-03  9.929e-04   5.382
## in.plug_load_diversity                               NA         NA      NA
## in.refrigerator                              -3.253e-03  1.932e-03  -1.684
## in.roof_material                              5.763e-04  1.105e-03   0.521
## in.tenure                                    -5.194e-02  6.769e-03  -7.672
## in.usage_level                                       NA         NA      NA
## in.vacancy_status                            -7.599e-01  1.099e-02 -69.163
## in.vintage                                    1.216e-03  1.472e-03   0.826
## in.vintage_acs                               -5.370e-03  2.227e-03  -2.411
## in.water_heater_efficiency                    7.018e-03  1.621e-03   4.330
## in.water_heater_fuel                         -1.295e-02  6.129e-03  -2.113
## in.weather_file_city                          5.858e-04  6.799e-04   0.862
## in.weather_file_latitude                     -1.383e-02  8.439e-03  -1.638
## in.weather_file_longitude                    -3.886e-02  3.334e-03 -11.657
## in.window_areas                              -1.516e-03  1.537e-03  -0.987
## in.windows                                    5.168e-03  1.137e-03   4.547
## upgrade.hvac_heating_efficiency                      NA         NA      NA
## Dry_Bulb_Temperature_C                        6.492e-02  5.900e-03  11.002
## Relative_Humidity                            -5.725e-03  1.266e-03  -4.521
## Wind_Speed_m_s                                8.150e-02  5.116e-03  15.930
## Wind_Direction_Deg                            2.184e-04  1.181e-04   1.849
## Diffuse_Horizontal_Radiation_W_m2            -4.933e-04  6.751e-05  -7.308
## time_split_numeric                            5.281e-02  2.224e-03  23.748
##                                              Pr(>|t|)
## (Intercept)                                  5.41e-06 ***
## bldg_id                                       0.07527 .
## in.sqft                                       < 2e-16 ***
## in.bedrooms                                  4.69e-15 ***
## in.building_america_climate_zone              0.04589 *
## in.ceiling_fan                                0.12310
## in.clothes_dryer                              0.08098 .
## in.clothes_washer                            5.18e-06 ***
## in.cooking_range                              0.35075
## in.cooling_setpoint                           < 2e-16 ***
## in.cooling_setpoint_offset_magnitude         3.71e-06 ***
## in.dishwasher                                7.79e-06 ***
## in.federal_poverty_level                      < 2e-16 ***
## in.geometry_attic_type                       2.98e-08 ***
## in.geometry_floor_area                             NA
## in.geometry_floor_area_bin                    < 2e-16 ***
## in.geometry_garage                            < 2e-16 ***
## in.heating_fuel                               0.70231
## in.heating_setpoint                           < 2e-16 ***
## in.heating_setpoint_offset_magnitude         2.33e-05 ***
## in.hot_water_fixtures                         < 2e-16 ***
## in.hvac_cooling_efficiency                    0.05203 .
## in.hvac_cooling_partial_space_conditioning    0.19026
```

```
## in.hvac_cooling_type                         0.00206 **
## in.hvac_has_ducts                            < 2e-16 ***
## in.hvac_has_zonal_electric_heating           0.14089
## in.hvac_heating_type                         0.01061 *
## in.hvac_heating_type_and_fuel                0.87896
## in.income                                    < 2e-16 ***
## in.insulation_ceiling                        0.17601
## in.insulation_floor                          < 2e-16 ***
## in.insulation_foundation_wall                0.06511 .
## in.insulation_rim_joist                           NA
## in.insulation_roof                           0.04192 *
## in.insulation_slab                           0.29192
## in.lighting                                  < 2e-16 ***
## in.misc_extra_refrigerator                   < 2e-16 ***
## in.misc_freezer                              < 2e-16 ***
## in.misc_gas_fireplace                        < 2e-16 ***
## in.misc_gas_grill                            0.03001 *
## in.misc_gas_lighting                         0.02733 *
## in.misc_hot_tub_spa                          < 2e-16 ***
## in.misc_pool                                 < 2e-16 ***
## in.misc_pool_heater                          < 2e-16 ***
## in.misc_pool_pump                                 NA
## in.misc_well_pump                           8.52e-10 ***
## in.orientation                              7.41e-08 ***
## in.plug_load_diversity                            NA
## in.refrigerator                              0.09224 .
## in.roof_material                             0.60209
## in.tenure                                   1.75e-14 ***
## in.usage_level                                    NA
## in.vacancy_status                            < 2e-16 ***
## in.vintage                                   0.40884
## in.vintage_acs                               0.01590 *
## in.water_heater_efficiency                  1.50e-05 ***
## in.water_heater_fuel                         0.03458 *
## in.weather_file_city                         0.38890
## in.weather_file_latitude                     0.10137
## in.weather_file_longitude                    < 2e-16 ***
## in.window_areas                              0.32378
## in.windows                                  5.46e-06 ***
## upgrade.hvac_heating_efficiency                   NA
## Dry_Bulb_Temperature_C                       < 2e-16 ***
## Relative_Humidity                           6.19e-06 ***
## Wind_Speed_m_s                               < 2e-16 ***
## Wind_Direction_Deg                           0.06441 .
## Diffuse_Horizontal_Radiation_W_m2           2.79e-13 ***
## time_split_numeric                           < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3857 on 27346 degrees of freedom
## Multiple R-squared:  0.6924, Adjusted R-squared:  0.6917
## F-statistic:   993 on 62 and 27346 DF,  p-value: < 2.2e-16
```

```r
#test_data$predictions <- predictions
mape <- mean(abs((test_data$out.total_energy_consumption - predictions) / test_data$out.total_en
ergy_consumption )) * 100

# Print the result
print(paste("MAPE:", mape))
```

```
## [1] "MAPE: 24.9013271848076"
```

```r
# # Install and load necessary packages
# # install.packages(c("e1071", "caret"))
 library(e1071)
 library(caret)
#
# # Set a seed for reproducibility
# set.seed(123)
#
# # Build SVM regression model
 svm_reg_model <- svm(out.total_energy_consumption ~ ., data = train_data, kernel = "radial")
#
# # Display the summary of the model
 summary(svm_reg_model)
```

```
##
## Call:
## svm(formula = out.total_energy_consumption ~ ., data = train_data,
##     kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.01470588
##     epsilon:  0.1
##
##
## Number of Support Vectors:  17158
```

```
#
 predictions <- predict(svm_reg_model, newdata = test_data)
# #print(predictions)
# # Evaluate the model, e.g., calculate RMSE (Root Mean Squared Error) or other metrics
# # ...
#
# # View summary of the linear regression model
# summary(svm_reg_model)
#test_data$predictions <- predictions
 mape <- mean(abs((test_data$out.total_energy_consumption - predictions) / test_data$out.total_e
nergy_consumption )) * 100
#Print the result
print(paste("MAPE:", mape))
```

```
## [1] "MAPE: 12.8670372041066"
```

Modeling_XGBoost

2023-12-04

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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
```

```
#Source<- "https://www.projectpro.io/recipes/apply-xgboost-r-for-regression"

Modeling_df <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/final_modeling_df.csv")
```

```
## Rows: 34260 Columns: 73
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (73): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#install.packages('xgboost')     # for fitting the xgboost model

#install.packages('caret')       # for general data preparation and model fitting

library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
#Clear NA before Modeling
Modeling_df <- Modeling_df[, colSums(is.na(Modeling_df)) == 0]
```

```
# Test and Train Data
set.seed(0) # Set seed for generating random data.

# CreateDataPartition() function from the caret package to split the original dataset into a training and testing set
# Split data into training (80%) and testing set (20%)
parts <- createDataPartition(Modeling_df$out.total_energy_consumption, p = 0.8, list = FALSE)
train <- Modeling_df[parts,]
test <-  Modeling_df[-parts,]

# Define predictor and response variables in the training set
train_x <- data.matrix(train[, -which(names(train) == "out.total_energy_consumption")])
train_y <- train[["out.total_energy_consumption"]]

# Define predictor and response variables in the testing set
test_x <- data.matrix(test[, -which(names(train) == "out.total_energy_consumption")])
test_y <- test[["out.total_energy_consumption"]]

print(c("Length of train_y:", length(train_y)))
```

```
## [1] "Length of train_y:" "27409"
```

```
print(c("Number of rows in train_x:", nrow(train_x)))
```

```
## [1] "Number of rows in train_x:" "27409"
```

```
# Check if lengths match before creating xgb.DMatrix
stopifnot(length(train_y) == nrow(train_x))

# Continue with the rest of your code...
#define final training and testing sets
xgb_train = xgb.DMatrix(data = train_x, label = train_y)
xgb_test = xgb.DMatrix(data = test_x, label = test_y)
```

```
#defining a watchlist
watchlist = list(train=xgb_train, test=xgb_test)

#fit XGBoost model and display training and testing data at each iteartion
model = xgb.train(data = xgb_train, max.depth = 3, watchlist=watchlist, nrounds = 100)
```

```
## [1]  train-rmse:0.833092 test-rmse:0.842933
## [2]  train-rmse:0.680978 test-rmse:0.691427
## [3]  train-rmse:0.586000 test-rmse:0.596543
## [4]  train-rmse:0.518287 test-rmse:0.530318
## [5]  train-rmse:0.477151 test-rmse:0.489412
## [6]  train-rmse:0.447918 test-rmse:0.458376
## [7]  train-rmse:0.425127 test-rmse:0.435131
## [8]  train-rmse:0.407650 test-rmse:0.416784
## [9]  train-rmse:0.392178 test-rmse:0.401232
## [10] train-rmse:0.379628 test-rmse:0.388338
## [11] train-rmse:0.369379 test-rmse:0.377432
## [12] train-rmse:0.360975 test-rmse:0.369086
## [13] train-rmse:0.353762 test-rmse:0.362518
## [14] train-rmse:0.347192 test-rmse:0.355965
## [15] train-rmse:0.340942 test-rmse:0.349648
## [16] train-rmse:0.335566 test-rmse:0.343911
## [17] train-rmse:0.330976 test-rmse:0.339372
## [18] train-rmse:0.327137 test-rmse:0.335789
## [19] train-rmse:0.322582 test-rmse:0.331109
## [20] train-rmse:0.319670 test-rmse:0.327832
## [21] train-rmse:0.317091 test-rmse:0.325705
## [22] train-rmse:0.313950 test-rmse:0.323611
## [23] train-rmse:0.309055 test-rmse:0.318640
## [24] train-rmse:0.306845 test-rmse:0.316983
## [25] train-rmse:0.304193 test-rmse:0.314759
## [26] train-rmse:0.301700 test-rmse:0.312069
## [27] train-rmse:0.299897 test-rmse:0.310662
## [28] train-rmse:0.298331 test-rmse:0.309268
## [29] train-rmse:0.296533 test-rmse:0.307407
## [30] train-rmse:0.293200 test-rmse:0.304569
## [31] train-rmse:0.289907 test-rmse:0.301507
## [32] train-rmse:0.288075 test-rmse:0.299842
## [33] train-rmse:0.286951 test-rmse:0.298710
## [34] train-rmse:0.285763 test-rmse:0.297489
## [35] train-rmse:0.284018 test-rmse:0.295518
## [36] train-rmse:0.282335 test-rmse:0.294119
## [37] train-rmse:0.281168 test-rmse:0.293203
## [38] train-rmse:0.279014 test-rmse:0.290957
## [39] train-rmse:0.278217 test-rmse:0.290225
```

```
## [40] train-rmse:0.277397 test-rmse:0.289531
## [41] train-rmse:0.274468 test-rmse:0.286811
## [42] train-rmse:0.273445 test-rmse:0.285857
## [43] train-rmse:0.272454 test-rmse:0.284376
## [44] train-rmse:0.271554 test-rmse:0.283870
## [45] train-rmse:0.270862 test-rmse:0.283310
## [46] train-rmse:0.270149 test-rmse:0.282812
## [47] train-rmse:0.269439 test-rmse:0.282276
## [48] train-rmse:0.268156 test-rmse:0.281158
## [49] train-rmse:0.267257 test-rmse:0.280304
## [50] train-rmse:0.266526 test-rmse:0.279945
## [51] train-rmse:0.265881 test-rmse:0.279587
## [52] train-rmse:0.265222 test-rmse:0.278789
## [53] train-rmse:0.264693 test-rmse:0.278403
## [54] train-rmse:0.264010 test-rmse:0.277836
## [55] train-rmse:0.262741 test-rmse:0.276695
## [56] train-rmse:0.262250 test-rmse:0.276278
## [57] train-rmse:0.261673 test-rmse:0.275714
## [58] train-rmse:0.261167 test-rmse:0.275380
## [59] train-rmse:0.260322 test-rmse:0.274680
## [60] train-rmse:0.259840 test-rmse:0.274297
## [61] train-rmse:0.259049 test-rmse:0.273700
## [62] train-rmse:0.258032 test-rmse:0.272812
## [63] train-rmse:0.257819 test-rmse:0.272464
## [64] train-rmse:0.257292 test-rmse:0.272120
## [65] train-rmse:0.256900 test-rmse:0.271779
## [66] train-rmse:0.256443 test-rmse:0.271550
## [67] train-rmse:0.254974 test-rmse:0.270241
## [68] train-rmse:0.254566 test-rmse:0.270064
## [69] train-rmse:0.254139 test-rmse:0.269642
## [70] train-rmse:0.253650 test-rmse:0.269195
## [71] train-rmse:0.253061 test-rmse:0.268766
## [72] train-rmse:0.252678 test-rmse:0.268408
## [73] train-rmse:0.251480 test-rmse:0.266984
## [74] train-rmse:0.251229 test-rmse:0.266711
## [75] train-rmse:0.250596 test-rmse:0.266132
## [76] train-rmse:0.250227 test-rmse:0.265872
## [77] train-rmse:0.249578 test-rmse:0.265742
## [78] train-rmse:0.249349 test-rmse:0.265561
```

```
## [79] train-rmse:0.248832 test-rmse:0.265305
## [80] train-rmse:0.248425 test-rmse:0.265024
## [81] train-rmse:0.248045 test-rmse:0.264714
## [82] train-rmse:0.247659 test-rmse:0.264269
## [83] train-rmse:0.247200 test-rmse:0.264104
## [84] train-rmse:0.246898 test-rmse:0.264055
## [85] train-rmse:0.246571 test-rmse:0.263863
## [86] train-rmse:0.246460 test-rmse:0.263738
## [87] train-rmse:0.245856 test-rmse:0.263249
## [88] train-rmse:0.245477 test-rmse:0.262999
## [89] train-rmse:0.245288 test-rmse:0.262790
## [90] train-rmse:0.245048 test-rmse:0.262488
## [91] train-rmse:0.244465 test-rmse:0.261917
## [92] train-rmse:0.244248 test-rmse:0.261813
## [93] train-rmse:0.243802 test-rmse:0.261522
## [94] train-rmse:0.243557 test-rmse:0.261342
## [95] train-rmse:0.243350 test-rmse:0.261143
## [96] train-rmse:0.243002 test-rmse:0.260893
## [97] train-rmse:0.242623 test-rmse:0.260372
## [98] train-rmse:0.242452 test-rmse:0.260229
## [99] train-rmse:0.242108 test-rmse:0.260090
## [100]    train-rmse:0.241917 test-rmse:0.259881
```

```
#define final model
model_xgboost = xgboost(data = xgb_train, max.depth = 3, nrounds = 86, verbose = 0)

summary(model_xgboost)
```

```
##                Length Class              Mode
## handle              1 xgb.Booster.handle externalptr
## raw            104154 -none-             raw
## niter               1 -none-             numeric
## evaluation_log      2 data.table         list
## call               14 -none-             call
## params              2 -none-             list
## callbacks           1 -none-             list
## feature_names      68 -none-             character
## nfeatures           1 -none-             numeric
```

```
#use model to make predictions on test data
pred_y = predict(model_xgboost, xgb_test)
```

```
# Assuming pred_y is your predicted values

# Calculate Mean Squared Error (MSE)
mse <- mean((test_y - pred_y)^2)
cat('Mean Squared Error (MSE): ', round(mse, 3), '\n')
```

```
## Mean Squared Error (MSE):  0.07
```

```
# Calculate Root Mean Squared Error (RMSE) using caret package
rmse <- caret::RMSE(test_y, pred_y)
cat('Root Mean Squared Error (RMSE): ', round(rmse, 3), '\n')
```

```
## Root Mean Squared Error (RMSE):  0.264
```

```
# Calculate R-squared
y_test_mean <- mean(test_y)
tss <- sum((test_y - y_test_mean)^2)
rss <- sum((test_y - pred_y)^2)  # Using predicted values to calculate residuals
rsq <- 1 - (rss/tss)
cat('The R-squared of the test data is ', round(rsq, 3), '\n')
```

```
## The R-squared of the test data is  0.857
```

```
predictions_xgb <- predict(model_xgboost, newdata = xgb_test)

mape <- mean(abs((test$out.total_energy_consumption - predictions_xgb) / test$out.total_energy_consumption )) * 100

# Print the result
print(paste("MAPE:", mape))
```

```
## [1] "MAPE: 15.2211081956591"
```

PREDICTIVE MODEL FOR THE NEXT YEAR WITH WEATHER INCREASE BY 5

Future Energy Prediction

2023-12-04

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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 becom
e errors
```

```
# Modeling_df_without_ordinal <- read_csv("C:/Users/Soundarya Ravi/Desktop/Merged_data.csv")

ModelingDf_for_d <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/final_modeling_df.csv")
```

```
## Rows: 34260 Columns: 73
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## dbl (73): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
New_Dataset_Weather_5 <- ModelingDf_for_d
```

```
New_Dataset_Weather_5$Dry_Bulb_Temperature_C <- New_Dataset_Weather_5$Dry_Bulb_Temperature_C + 5
```

```
#install.packages('xgboost')     # for fitting the xgboost model

#install.packages('caret')       # for general data preparation and model fitting

library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
#Clear NA before Modeling
New_Dataset_Weather_5 <- New_Dataset_Weather_5[, colSums(is.na(New_Dataset_Weather_5)) == 0]
```

```
# Test and Train Data
set.seed(0) # Set seed for generating random data.

# CreateDataPartition() function from the caret package to split the original dataset into a tra
ining and testing set
# Split data into training (80%) and testing set (20%)
parts <- createDataPartition(New_Dataset_Weather_5$out.total_energy_consumption, p = 0.8, list =
FALSE)
train <- New_Dataset_Weather_5[parts,]
test <-  New_Dataset_Weather_5[-parts,]

# Define predictor and response variables in the training set
train_x <- data.matrix(train[, -which(names(train) == "out.total_energy_consumption")])
train_y <- train[["out.total_energy_consumption"]]

# Define predictor and response variables in the testing set
test_x <- data.matrix(test[, -which(names(train) == "out.total_energy_consumption")])
test_y <- test[["out.total_energy_consumption"]]

print(c("Length of train_y:", length(train_y)))
```

```
## [1] "Length of train_y:" "27409"
```

```
print(c("Number of rows in train_x:", nrow(train_x)))
```

```
## [1] "Number of rows in train_x:" "27409"
```

```
# Check if lengths match before creating xgb.DMatrix
stopifnot(length(train_y) == nrow(train_x))

# Continue with the rest of your code...
#define final training and testing sets
xgb_train = xgb.DMatrix(data = train_x, label = train_y)
xgb_test = xgb.DMatrix(data = test_x, label = test_y)
```

```
#defining a watchlist
watchlist = list(train=xgb_train, test=xgb_test)

#fit XGBoost model and display training and testing data at each iteartion
model = xgb.train(data = xgb_train, max.depth = 3, watchlist=watchlist, nrounds = 100)
```

```
## [1]  train-rmse:0.833092 test-rmse:0.842933
## [2]  train-rmse:0.680978 test-rmse:0.691427
## [3]  train-rmse:0.586000 test-rmse:0.596543
## [4]  train-rmse:0.518287 test-rmse:0.530318
## [5]  train-rmse:0.477151 test-rmse:0.489412
## [6]  train-rmse:0.447918 test-rmse:0.458376
## [7]  train-rmse:0.425127 test-rmse:0.435131
## [8]  train-rmse:0.407650 test-rmse:0.416784
## [9]  train-rmse:0.392178 test-rmse:0.401232
## [10] train-rmse:0.379628 test-rmse:0.388338
## [11] train-rmse:0.369379 test-rmse:0.377432
## [12] train-rmse:0.360975 test-rmse:0.369086
## [13] train-rmse:0.353762 test-rmse:0.362518
## [14] train-rmse:0.347192 test-rmse:0.355965
## [15] train-rmse:0.340942 test-rmse:0.349648
## [16] train-rmse:0.335566 test-rmse:0.343911
## [17] train-rmse:0.330976 test-rmse:0.339372
## [18] train-rmse:0.327137 test-rmse:0.335789
## [19] train-rmse:0.322582 test-rmse:0.331109
## [20] train-rmse:0.319670 test-rmse:0.327832
## [21] train-rmse:0.317091 test-rmse:0.325705
## [22] train-rmse:0.313950 test-rmse:0.323611
## [23] train-rmse:0.309055 test-rmse:0.318640
## [24] train-rmse:0.306845 test-rmse:0.316983
## [25] train-rmse:0.304193 test-rmse:0.314759
## [26] train-rmse:0.301700 test-rmse:0.312069
## [27] train-rmse:0.299897 test-rmse:0.310662
## [28] train-rmse:0.298331 test-rmse:0.309268
## [29] train-rmse:0.296533 test-rmse:0.307407
## [30] train-rmse:0.293200 test-rmse:0.304569
## [31] train-rmse:0.289907 test-rmse:0.301507
## [32] train-rmse:0.288075 test-rmse:0.299842
## [33] train-rmse:0.286951 test-rmse:0.298710
## [34] train-rmse:0.285763 test-rmse:0.297489
## [35] train-rmse:0.284018 test-rmse:0.295518
## [36] train-rmse:0.282335 test-rmse:0.294119
## [37] train-rmse:0.281168 test-rmse:0.293203
## [38] train-rmse:0.279014 test-rmse:0.290957
## [39] train-rmse:0.278217 test-rmse:0.290225
## [40] train-rmse:0.277397 test-rmse:0.289531
## [41] train-rmse:0.274468 test-rmse:0.286811
## [42] train-rmse:0.273445 test-rmse:0.285857
## [43] train-rmse:0.272454 test-rmse:0.284376
## [44] train-rmse:0.271554 test-rmse:0.283870
## [45] train-rmse:0.270862 test-rmse:0.283310
## [46] train-rmse:0.270149 test-rmse:0.282812
## [47] train-rmse:0.269439 test-rmse:0.282276
## [48] train-rmse:0.268156 test-rmse:0.281158
## [49] train-rmse:0.267257 test-rmse:0.280304
## [50] train-rmse:0.266526 test-rmse:0.279945
## [51] train-rmse:0.265881 test-rmse:0.279587
## [52] train-rmse:0.265222 test-rmse:0.278789
```

```
## [53] train-rmse:0.264693 test-rmse:0.278403
## [54] train-rmse:0.264010 test-rmse:0.277836
## [55] train-rmse:0.262741 test-rmse:0.276695
## [56] train-rmse:0.262250 test-rmse:0.276278
## [57] train-rmse:0.261673 test-rmse:0.275714
## [58] train-rmse:0.261167 test-rmse:0.275380
## [59] train-rmse:0.260322 test-rmse:0.274680
## [60] train-rmse:0.259840 test-rmse:0.274297
## [61] train-rmse:0.259049 test-rmse:0.273700
## [62] train-rmse:0.258032 test-rmse:0.272812
## [63] train-rmse:0.257819 test-rmse:0.272464
## [64] train-rmse:0.257292 test-rmse:0.272120
## [65] train-rmse:0.256900 test-rmse:0.271779
## [66] train-rmse:0.256443 test-rmse:0.271550
## [67] train-rmse:0.254974 test-rmse:0.270241
## [68] train-rmse:0.254566 test-rmse:0.270064
## [69] train-rmse:0.254139 test-rmse:0.269642
## [70] train-rmse:0.253650 test-rmse:0.269195
## [71] train-rmse:0.253061 test-rmse:0.268766
## [72] train-rmse:0.252678 test-rmse:0.268408
## [73] train-rmse:0.251480 test-rmse:0.266984
## [74] train-rmse:0.251229 test-rmse:0.266711
## [75] train-rmse:0.250596 test-rmse:0.266132
## [76] train-rmse:0.250227 test-rmse:0.265872
## [77] train-rmse:0.249578 test-rmse:0.265742
## [78] train-rmse:0.249349 test-rmse:0.265561
## [79] train-rmse:0.248832 test-rmse:0.265305
## [80] train-rmse:0.248425 test-rmse:0.265024
## [81] train-rmse:0.248045 test-rmse:0.264714
## [82] train-rmse:0.247659 test-rmse:0.264269
## [83] train-rmse:0.247200 test-rmse:0.264104
## [84] train-rmse:0.246898 test-rmse:0.264055
## [85] train-rmse:0.246571 test-rmse:0.263863
## [86] train-rmse:0.246460 test-rmse:0.263738
## [87] train-rmse:0.245856 test-rmse:0.263249
## [88] train-rmse:0.245477 test-rmse:0.262999
## [89] train-rmse:0.245288 test-rmse:0.262790
## [90] train-rmse:0.245048 test-rmse:0.262488
## [91] train-rmse:0.244465 test-rmse:0.261917
## [92] train-rmse:0.244248 test-rmse:0.261813
## [93] train-rmse:0.243802 test-rmse:0.261522
## [94] train-rmse:0.243557 test-rmse:0.261342
## [95] train-rmse:0.243350 test-rmse:0.261143
## [96] train-rmse:0.243002 test-rmse:0.260893
## [97] train-rmse:0.242623 test-rmse:0.260372
## [98] train-rmse:0.242452 test-rmse:0.260229
## [99] train-rmse:0.242108 test-rmse:0.260090
## [100]    train-rmse:0.241917 test-rmse:0.259881
```

```
#define final model
model_xgboost = xgboost(data = xgb_train, max.depth = 3, nrounds = 86, verbose = 0)

summary(model_xgboost)
```

```
##                Length Class              Mode
## handle              1 xgb.Booster.handle externalptr
## raw            104154 -none-             raw
## niter               1 -none-             numeric
## evaluation_log      2 data.table         list
## call               14 -none-             call
## params              2 -none-             list
## callbacks           1 -none-             list
## feature_names      68 -none-             character
## nfeatures           1 -none-             numeric
```

```
#use model to make predictions on test data
pred_y = predict(model_xgboost, xgb_test)
```

```
# Assuming pred_y is your predicted values

# Calculate Mean Squared Error (MSE)
mse <- mean((test_y - pred_y)^2)
cat('Mean Squared Error (MSE): ', round(mse, 3), '\n')
```

```
## Mean Squared Error (MSE):  0.07
```

```
# Calculate Root Mean Squared Error (RMSE) using caret package
rmse <- caret::RMSE(test_y, pred_y)
cat('Root Mean Squared Error (RMSE): ', round(rmse, 3), '\n')
```

```
## Root Mean Squared Error (RMSE):  0.264
```

```
# Calculate R-squared
y_test_mean <- mean(test_y)
tss <- sum((test_y - y_test_mean)^2)
rss <- sum((test_y - pred_y)^2)  # Using predicted values to calculate residuals
rsq <- 1 - (rss/tss)
cat('The R-squared of the test data is ', round(rsq, 3), '\n')
```

```
## The R-squared of the test data is  0.857
```

```
predictions_xgb <- predict(model_xgboost, newdata = xgb_test)

mape <- mean(abs((test$out.total_energy_consumption - predictions_xgb) / test$out.total_energy_c
onsumption )) * 100

# Print the result
print(paste("MAPE:", mape))
```

```
## [1] "MAPE: 15.2211081956591"
```

Add Predicted Values to the test and train

```
#use model to make predictions on test data
pred_y <- predict(model_xgboost, xgb_test)

pred_x <- predict(model_xgboost,xgb_train)
```

```
train$Next_year_Pred <- pred_x
test$Next_year_Pred <- pred_y
```

```
# Combine train and test dataframes
New_Weather_Increase_Prediction <- rbind(train, test)

# Check the structure of the new dataframe
str(New_Weather_Increase_Prediction)
```

```
## tibble [34,260 × 70] (S3: tbl_df/tbl/data.frame)
##  $ bldg_id                                : num [1:34260] 65 65 65 121 121 121 121 121 121
500 ...
##  $ in.sqft                                : num [1:34260] 3 3 3 4 4 4 4 4 4 4 ...
##  $ in.bedrooms                            : num [1:34260] 3 3 3 2 2 2 2 2 2 3 ...
##  $ in.building_america_climate_zone       : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.ceiling_fan                         : num [1:34260] 2 2 2 0 0 0 0 0 0 2 ...
##  $ in.clothes_dryer                       : num [1:34260] 5 5 5 2 2 2 2 2 2 1 ...
##  $ in.clothes_washer                      : num [1:34260] 5 5 5 2 2 2 2 2 2 4 ...
##  $ in.cooking_range                       : num [1:34260] 2 2 2 2 2 2 2 2 2 4 ...
##  $ in.cooling_setpoint                    : num [1:34260] 7 7 7 9 9 9 9 9 9 6 ...
##  $ in.cooling_setpoint_offset_magnitude   : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.dishwasher                          : num [1:34260] 0 0 0 1 1 1 1 1 1 0 ...
##  $ in.federal_poverty_level               : num [1:34260] 1 1 1 2 2 2 2 2 2 3 ...
##  $ in.geometry_attic_type                 : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.geometry_floor_area                 : num [1:34260] 2 2 2 3 3 3 3 3 3 3 ...
##  $ in.geometry_floor_area_bin             : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.geometry_garage                     : num [1:34260] 1 1 1 0 0 0 0 0 0 1 ...
##  $ in.heating_fuel                        : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.heating_setpoint                    : num [1:34260] 6 6 6 3 3 3 3 3 3 6 ...
##  $ in.heating_setpoint_offset_magnitude   : num [1:34260] 0 0 0 1 1 1 1 1 1 0 ...
##  $ in.hot_water_fixtures                  : num [1:34260] 2 2 2 2 2 2 2 2 2 1 ...
##  $ in.hvac_cooling_efficiency             : num [1:34260] 1 1 1 2 2 2 2 2 2 2 ...
##  $ in.hvac_cooling_partial_space_conditioning: num [1:34260] 5 5 5 5 5 5 5 5 5 5 ...
##  $ in.hvac_cooling_type                   : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.hvac_has_ducts                      : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.hvac_has_zonal_electric_heating     : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.hvac_heating_type                   : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.hvac_heating_type_and_fuel          : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.income                              : num [1:34260] 14999 14999 14999 19999 19999
...
##  $ in.insulation_ceiling                  : num [1:34260] 4 4 4 2 2 2 2 2 2 4 ...
##  $ in.insulation_floor                    : num [1:34260] 0 0 0 1 1 1 1 1 1 0 ...
##  $ in.insulation_foundation_wall          : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.insulation_rim_joist                : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.insulation_roof                     : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.insulation_slab                     : num [1:34260] 1 1 1 2 2 2 2 2 2 3 ...
##  $ in.lighting                            : num [1:34260] 1 1 1 2 2 2 2 2 2 2 ...
##  $ in.misc_extra_refrigerator             : num [1:34260] 5 5 5 5 5 5 5 5 5 0 ...
##  $ in.misc_freezer                        : num [1:34260] 1 1 1 0 0 0 0 0 0 0 ...
##  $ in.misc_gas_fireplace                  : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_gas_grill                      : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_gas_lighting                   : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_hot_tub_spa                    : num [1:34260] 0 0 0 0 0 0 0 0 0 2 ...
##  $ in.misc_pool                           : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_pool_heater                    : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_pool_pump                      : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.misc_well_pump                      : num [1:34260] 0 0 0 0 0 0 0 0 0 0 ...
##  $ in.orientation                         : num [1:34260] 1 1 1 2 2 2 2 2 2 2 ...
##  $ in.plug_load_diversity                 : num [1:34260] 1 1 1 1 1 1 1 1 1 0 ...
##  $ in.refrigerator                        : num [1:34260] 1 1 1 2 2 2 2 2 2 3 ...
##  $ in.roof_material                       : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
```

```
##  $ in.tenure                                : num [1:34260] 1 1 1 2 2 2 2 2 2 2 ...
##  $ in.usage_level                           : num [1:34260] 2 2 2 2 2 2 2 2 2 1 ...
##  $ in.vacancy_status                        : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.vintage                               : num [1:34260] 1 1 1 1 1 1 1 1 1 2 ...
##  $ in.vintage_acs                           : num [1:34260] 1 1 1 1 1 1 1 1 1 2 ...
##  $ in.water_heater_efficiency               : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.water_heater_fuel                     : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ in.weather_file_city                     : num [1:34260] 1 1 1 2 2 2 2 2 2 3 ...
##  $ in.weather_file_latitude                 : num [1:34260] 35 35 35 34.7 34.7 ...
##  $ in.weather_file_longitude                : num [1:34260] -81.1 -81.1 -81.1 -82.9 -82.9
...
##  $ in.window_areas                          : num [1:34260] 1 1 1 2 2 2 2 2 2 2 ...
##  $ in.windows                               : num [1:34260] 1 1 1 2 2 2 2 2 2 1 ...
##  $ upgrade.hvac_heating_efficiency          : num [1:34260] 1 1 1 1 1 1 1 1 1 1 ...
##  $ out.total_energy_consumption             : num [1:34260] 1.265 1.803 1.293 0.385 0.492
...
##  $ Dry_Bulb_Temperature_C                   : num [1:34260] 32.6 34.8 29.9 27.7 27.4 ...
##  $ Relative_Humidity                        : num [1:34260] 70.1 60.3 82.1 89.7 89.7 ...
##  $ Wind_Speed_m_s                           : num [1:34260] 2.15 2.44 1.08 1.17 1.43 ...
##  $ Wind_Direction_Deg                       : num [1:34260] 120.9 144 99.8 120 112.1 ...
##  $ Diffuse_Horizontal_Radiation_W_m2        : num [1:34260] 207.6 164 18.1 0 25.1 ...
##  $ time_split_numeric                       : num [1:34260] 3 5 6 1 2 3 4 5 6 1 ...
##  $ Next_year_Pred                           : num [1:34260] 1.342 1.824 1.259 0.531 0.62
```

```
# Assuming New_Weather_Increase_Prediction is your dataframe
New_Weather_Increase_Prediction <- New_Weather_Increase_Prediction[order(New_Weather_Increase_Pr
ediction$bldg_id, New_Weather_Increase_Prediction$time_split_numeric), ]

# Check the sorted dataframe
head(New_Weather_Increase_Prediction)
```

```
## # A tibble: 6 × 70
##   bldg_id in.sqft in.bedrooms in.building_america_climate_zone in.ceiling_fan
##     <dbl>   <dbl>       <dbl>                            <dbl>          <dbl>
## 1      65       3           3                                1              2
## 2      65       3           3                                1              2
## 3      65       3           3                                1              2
## 4      65       3           3                                1              2
## 5      65       3           3                                1              2
## 6      65       3           3                                1              2
## # ℹ 65 more variables: in.clothes_dryer <dbl>, in.clothes_washer <dbl>,
## #   in.cooking_range <dbl>, in.cooling_setpoint <dbl>,
## #   in.cooling_setpoint_offset_magnitude <dbl>, in.dishwasher <dbl>,
## #   in.federal_poverty_level <dbl>, in.geometry_attic_type <dbl>,
## #   in.geometry_floor_area <dbl>, in.geometry_floor_area_bin <dbl>,
## #   in.geometry_garage <dbl>, in.heating_fuel <dbl>, in.heating_setpoint <dbl>,
## #   in.heating_setpoint_offset_magnitude <dbl>, in.hot_water_fixtures <dbl>, …
```

```
#Difference In Energy Between Both Years
New_Weather_Increase_Prediction$Change_in_energy <- New_Weather_Increase_Prediction$Next_year_Pr
ed - New_Weather_Increase_Prediction$out.total_energy_consumption
```

```
# setwd("C:/Users/Soundarya Ravi/Desktop/Shiny")
# write.csv(New_Weather_Increase_Prediction,"Predicted_final_weatherplus5.csv",row.names=FALSE)
```

```
Weather_Increase_Predcited_Final <- read_csv("C:/Users/Soundarya Ravi/Desktop/Shiny/Predicted_fi
nal_weatherplus5.csv")
```

```
## Rows: 34260 Columns: 71
## ── Column specification ────────────────────────────────────────────────────
## Delimiter: ","
## dbl (71): bldg_id, in.sqft, in.bedrooms, in.building_america_climate_zone, i...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#Percentage Calculation For Increase In Temperature For Overall Data
# Percentage = (Total Change / Current Year ) * 100

Total_Change = sum(Weather_Increase_Predcited_Final$Change_in_energy)
Current_Year = sum(Weather_Increase_Predcited_Final$out.total_energy_consumption)
Percentage = (Total_Change/Current_Year)*100

Total_Change
```

```
## [1] 5105.762
```

```
Current_Year
```

```
## [1] 44816.9
```

```
# Example usage with cat
message <- "Total Energy Percentage Increase after Increasing the temperature by 5 is:"
variable_value <- Percentage # Replace this with your actual variable

cat(message, variable_value, "\n")
```

```
## Total Energy Percentage Increase after Increasing the temperature by 5 is: 11.39249
```

PEAK ENERGY DEMAND IN FUTURE BY TIME, GEOGRAPHY AND ATTRIBUTES

```
file <- '/Users/subhiksha/Downloads/Final_Merged_For_Shiny_Geo.csv'
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ dplyr     1.1.3     ✔ readr     2.1.4
## ✔ forcats   1.0.0     ✔ stringr   1.5.0
## ✔ ggplot2   3.4.4     ✔ tibble    3.2.1
## ✔ lubridate 1.9.3     ✔ tidyr     1.3.0
## ✔ 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
```

```
peak_energy_file <- read_csv(file)
```

```
## Rows: 34260 Columns: 96
## ── Column specification ────────────────────────────────────────────────────────
## Delimiter: ","
## chr (70): in.county, time_split, in.county_and_puma, in.building_america_cli...
## dbl (26): bldg_id, time_split_numeric, in.sqft, in.bedrooms, in.cooling_setp...
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
table(peak_energy_file$in.occupants)
```

```
##
##     1   10+     2     3     4     5     6     7     8     9
##  7668    48 13134  5610  4566  2016   690   348   138    42
```

```
nrow(peak_energy_file)
```

```
## [1] 34260
```

```
#colnames(peak_energy_file)
summary(peak_energy_file$out.total_energy_consumption)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.05835 0.83173 1.18313 1.30814 1.63109 8.32710
```

```
summary(peak_energy_file$Next_year_Pred)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## 0.08724 0.94739 1.32063 1.45717 1.80063 6.90360
```

```
#unique(peak_energy_file$in.county)

# Load necessary libraries
library(dplyr)
library(ggplot2)
library(readr)

# Data aggregation
grouped_data <- peak_energy_file %>%
  group_by(time_split, in.county) %>%
  summarize(Total_Energy_Demand = sum(Next_year_Pred, na.rm = TRUE),
            Avg_Latitude = mean(in.weather_file_latitude, na.rm = TRUE),
            Avg_Longitude = mean(in.weather_file_longitude, na.rm = TRUE),
            City_Name = first(in.weather_file_city[!is.na(in.weather_file_city)])) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'time_split'. You can override using the
## `.groups` argument.
```

```
grouped_data1 <- grouped_data[grouped_data$City_Name == 'Anderson Rgnl',]
class(grouped_data$Total_Energy_Demand)
```

```
## [1] "numeric"
```

```
#grouped_data$Total_Energy_Demand <round(grouped_data$Total_Energy_Demand)
min(grouped_data$Total_Energy_Demand)
```

```
## [1] 8.707579
```

```
max(grouped_data$Total_Energy_Demand)
```

```
## [1] 1081.414
```

```
summary(grouped_data$Total_Energy_Demand)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##    8.708   38.883   83.537  180.879  230.213 1081.414
```

```
unique(grouped_data$City_Name)
```

```
##  [1] "Greenwood Co"        "Augusta Bush Field"  "Orangeburg Muni"
##  [4] "Anderson Rgnl"       "Beaufort Mcas"       "Charleston Muni"
##  [7] "Rutherfordton"       "Rock Hill York Co"   "Monroe Airport"
## [10] "Shaw Afb Sumter"     "Florence Rgnl"       "Maxton"
## [13] "Daniel Field"        "Columbia Owens Apt"  "Myrtle Beach Civ"
## [16] "Greenville Greenvil" "Columbia Metro"      "Oconee Co Rgnl"
```

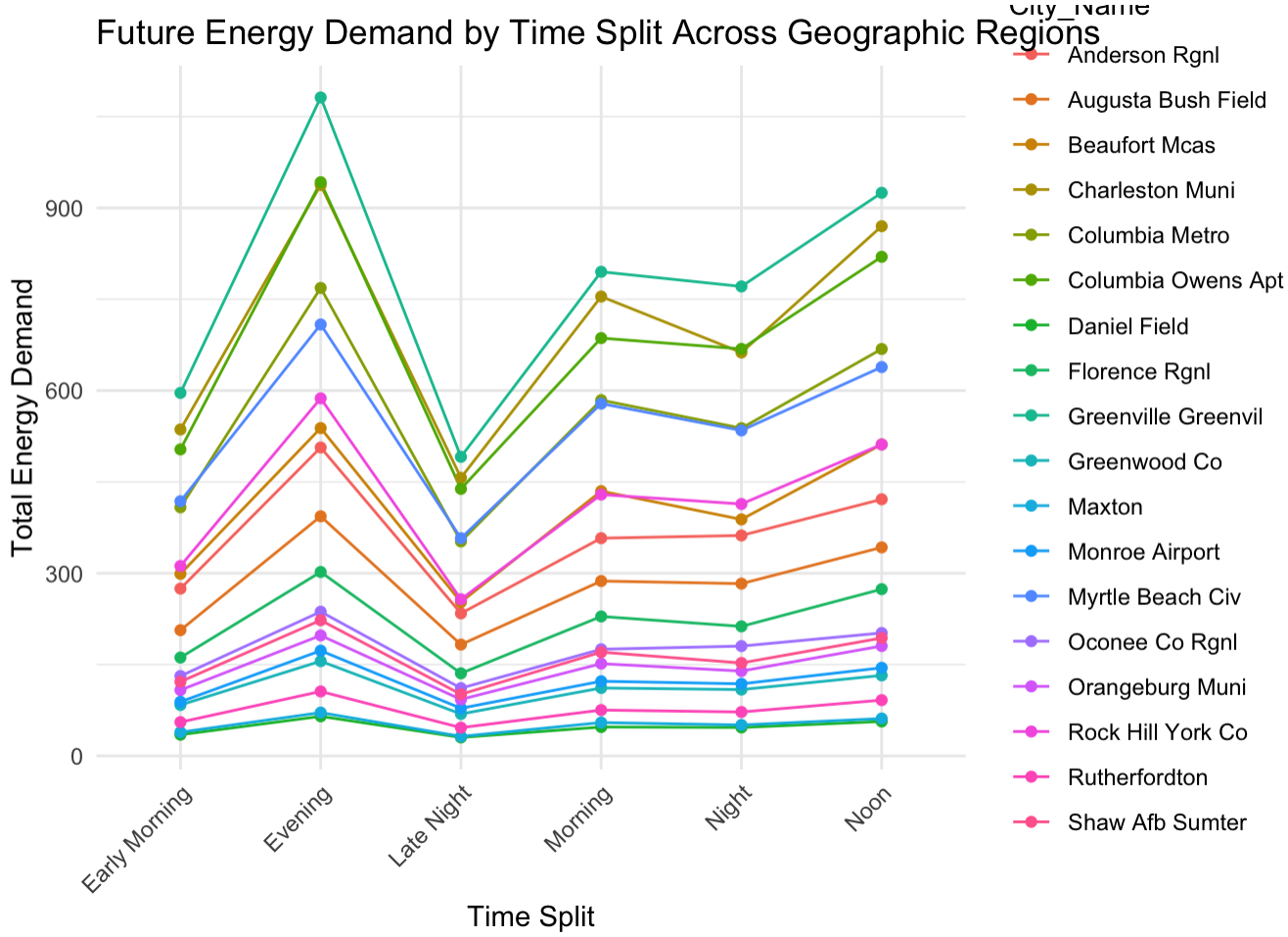
```
region_time_summaries <- grouped_data %>%
  group_by(City_Name, time_split) %>%
  summarise(Total_Energy_Demand = max(Total_Energy_Demand))
```

```
## `summarise()` has grouped output by 'City_Name'. You can override using the
## `.groups` argument.
```

```
# Checking the result
print(region_time_summaries)
```

```
## # A tibble: 108 × 3
## # Groups:   City_Name [18]
##    City_Name          time_split    Total_Energy_Demand
##    <chr>              <chr>                       <dbl>
##  1 Anderson Rgnl      Early Morning                275.
##  2 Anderson Rgnl      Evening                      506.
##  3 Anderson Rgnl      Late Night                   234.
##  4 Anderson Rgnl      Morning                      358.
##  5 Anderson Rgnl      Night                        362.
##  6 Anderson Rgnl      Noon                         421.
##  7 Augusta Bush Field Early Morning                206.
##  8 Augusta Bush Field Evening                      393.
##  9 Augusta Bush Field Late Night                   183.
## 10 Augusta Bush Field Morning                      287.
## # ℹ 98 more rows
```

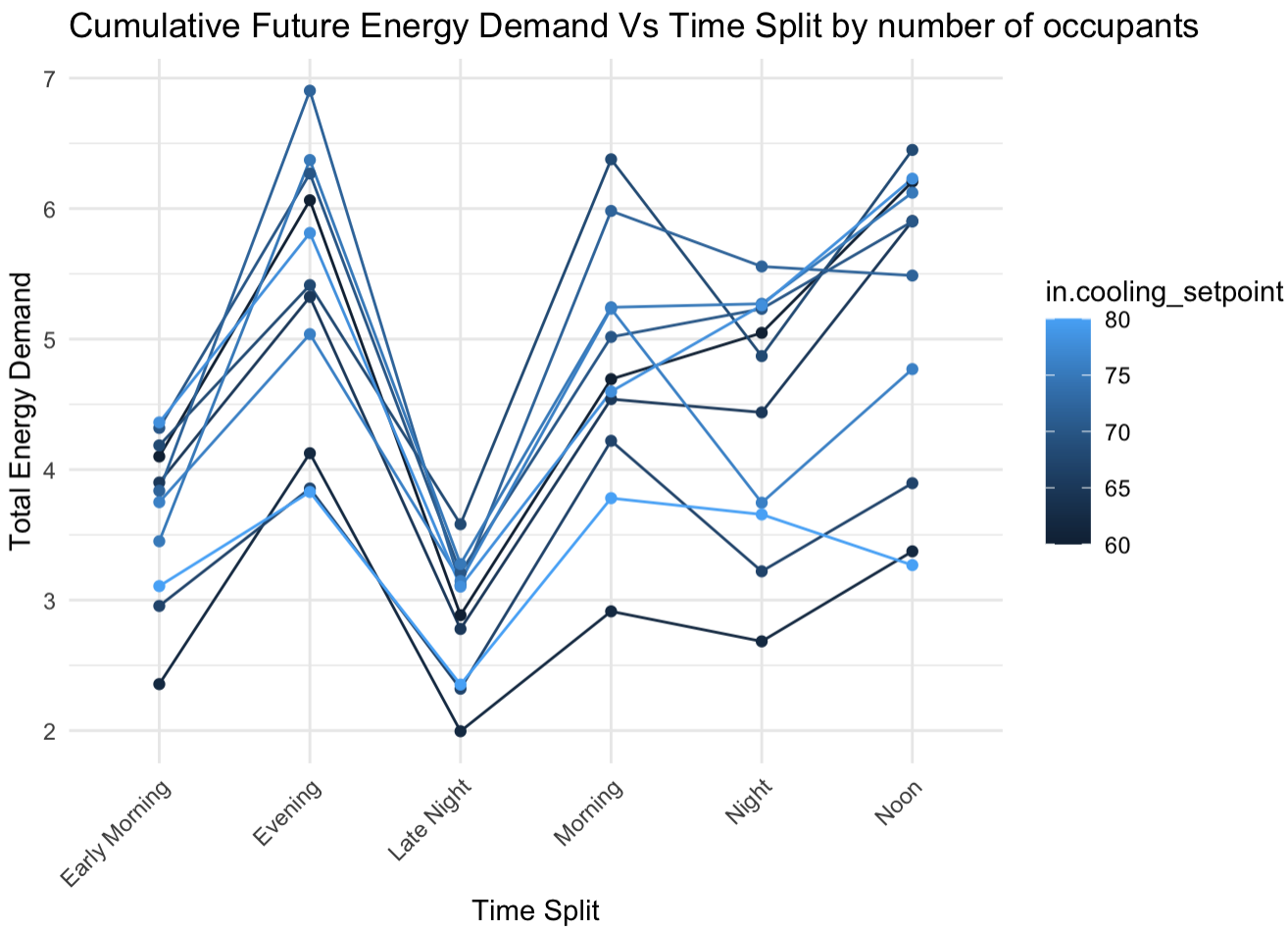
```
# Create the line plot a)
  ggplot(region_time_summaries, aes(x = time_split, y = Total_Energy_Demand, group = Cit
y_Name, color = City_Name)) +
  geom_line() +
  geom_point() +
  ggtitle("Future Energy Demand by Time Split Across Geographic Regions") +
  xlab("Time Split") +
  ylab("Total Energy Demand") +
  theme_minimal() + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```




```
# Data aggregation
grouped_data1 <- peak_energy_file %>%
  group_by(time_split,in.cooling_setpoint ) %>%
  summarize(Total_Energy_Demand = max(Next_year_Pred, na.rm = TRUE),) %>%
  ungroup()
```

```
## `summarise()` has grouped output by 'time_split'. You can override using the
## `.groups` argument.
```

```
# Create the line plot b)
  ggplot(grouped_data1, aes(x = time_split, y = Total_Energy_Demand, group = in.cooling_
setpoint, color = in.cooling_setpoint)) +
  geom_line() +
  geom_point() +
  ggtitle("Cumulative Future Energy Demand Vs Time Split by number of occupants") +
  xlab("Time Split") +
  ylab("Total Energy Demand") +
  theme_minimal() + theme(axis.text.x = element_text(angle = 45, hjust = 1))
```


Cumulative Future Energy Demand Vs Time Split by number of occupants
Total Energy Demand
7
6
5
4
3
2
Early Morning
Evening
Late Night
Morning
Night
Noon
Time Split
in.cooling_setpoint
80
75
70
65
60

SHINY APPLICATION DASHBOARD

```
# Load necessary libraries
library(shiny)
library(ggplot2)
library(dplyr)
library(leaflet)
library(jsonlite)
library(sf)
library(viridis)
library(plotly)

# Read your data
sorted_data <- read.csv("Final_Merged_For_Shiny_Geo.csv")



file_path <- "Team2_Final_SEW_Ordinal_Modelling1.csv"
model <- read.csv(file_path)



total_energy_data <- sorted_data %>%
  summarise(Current_Year = sum(out.total_energy_consumption),
            Next_Year = sum(Next_year_Pred))

# Calculate percentage increase for total energy
total_energy_data$Percentage_Increase <- ((total_energy_data$Next_Year -
total_energy_data$Current_Year) / total_energy_data$Current_Year) * 100

# Reshape the data for ggplot
reshaped_data <- reshape2::melt(total_energy_data, id.vars = NULL,
                                measure.vars = c("Current_Year", "Next_Year", "Percentage_Increase"),
```

```
                    variable.name = "Energy_Type", value.name = "Value")


# Define UI for the Shiny app
ui <- fluidPage(


  navbarPage("eSC ~ Energy Consumption Comparison",


        # First tab for the comparison of current and future energy consumption
        tabPanel("Current vs. Future Energy",
              sidebarLayout(
                sidebarPanel(
                  selectInput("in_county", "Select County", unique(sorted_data$in.county)),
                  selectInput("time_split", "Select Time of Day", unique(sorted_data$time_split))
                ),
                mainPanel(
                  plotOutput("energyComparisonPlot")
                )
              )
        ),


        # Second tab for the comparison of total energy consumption
        tabPanel("Total Energy Comparison",
              sidebarLayout(
                sidebarPanel(
                  HTML("Explore the comparison of current and future energy consumption for the month of July in South Carolina. This section provides insights into the key drivers of energy usage, aiming to help an energy company (eSC) understand potential challenges in meeting electricity demand during hot summers. By analyzing historical and predicted energy data, users can gain valuable information to encourage energy-saving practices and contribute to both efficient energy usage and environmental sustainability.")
```

```
          ),
          mainPanel(
            plotOutput("totalEnergyPlot")
          )
        )
      ),

      tabPanel("Energy Categories Distribution",
        sidebarLayout(
          sidebarPanel(
            selectInput("in_county_pie", "Select County", unique(sorted_data$in.county)),
            selectInput("time_split_pie", "Select Time of Day", unique(sorted_data$time_split),
selected = unique(sorted_data$time_split)[1])
          ),
          mainPanel(
            plotOutput("energyCategoryPieChart")
          )
        )
      ),

      tabPanel("Aggregated Panel",
        sidebarLayout(
          sidebarPanel(
            HTML("Explore the analysis of sorted energy consumption data. This section provides
insights into the aggregated energy demand based on the selected time and city."),
            selectInput("new_in_city", "Select City", unique(sorted_data$in.weather_file_city)),
            sliderInput("new_time_slider_city", "Select Time Range",
                        min = min(sorted_data$time_split_numeric),
                        max = max(sorted_data$time_split_numeric),
```

```
                    value = c(min(sorted_data$time_split_numeric),
max(sorted_data$time_split_numeric)),
                    step = 1)
                ),
                mainPanel(
                  plotOutput("new_energyPlot")
                )
              )
        ),

        tabPanel("Leaflet Maps",
              fluidRow(
                tags$style(HTML(".leaflet-container { border: 2px solid black; margin-bottom: 10px; }")),
                leafletOutput("map_current_leaflet"),
                leafletOutput("map_future_leaflet")
              )
        ),

        tabPanel("Energy Consumption Analysis",
              plotOutput("plot_a"),
              plotOutput("plot_b"),
              plotOutput("plot_c"),
              plotOutput("plot_d"))
  )
)
```

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

  output$energyComparisonPlot <- renderPlot({
    # Filter data based on user input
    filtered_data <- subset(sorted_data, in.county == input$in_county & time_split == input$time_split)

    # Check if the filtered data is not empty
    if (nrow(filtered_data) > 0) {
      # Summarize data for current year
      current_year_data <- filtered_data %>%
        group_by(time_split) %>%
        summarise(Current_Year = sum(out.total_energy_consumption))

      # Summarize data for the next year
      next_year_data <- filtered_data %>%
        group_by(time_split) %>%
        summarise(Next_Year = sum(Next_year_Pred))

      # Combine the two datasets
      combined_data <- merge(current_year_data, next_year_data, by = "time_split", all = TRUE)

      # Calculate percentage increase
      combined_data$Percentage_Increase <- ((combined_data$Next_Year -
combined_data$Current_Year) / combined_data$Current_Year) * 100

      # Reshape the data for ggplot
      reshaped_data <- reshape2::melt(combined_data, id.vars = "time_split",
                                      measure.vars = c("Current_Year", "Next_Year", "Percentage_Increase"),
```

```
                    variable.name = "Energy_Type", value.name = "Value")


    # Create a bar plot for current and future energy consumption

    ggplot(reshaped_data, aes(x = time_split, y = Value, fill = Energy_Type, label = ifelse(Energy_Type == "Percentage_Increase", paste(round(Value, 2), "%"), ""))) +

      geom_bar(stat = "identity", position = "dodge", width = 0.7) +

      geom_text(position = position_dodge(width = 0.7), vjust = -0.5) +

      labs(title = "Comparison of Current and Future Energy Consumption",

          x = "Time of Day",

          y = "Energy Consumption",

          fill = "Energy Type") +

      theme_minimal() +

      theme(legend.position = "top") +

      scale_fill_manual(values = c("Current_Year" = "#66c2a5", "Next_Year" = "#fc8d62", "Percentage_Increase" = "#8da0cb"))

   } else {

    # If filtered data is empty, display a message or a default plot

    ggplot() + theme_void() +

      annotate("text", x = 0.5, y = 0.5, label = "No data available for the selected filters.",

             color = "red", size = 5, hjust = 0.5, vjust = 0.5)

   }

 })


 output$totalEnergyPlot <- renderPlot({

   # Filter data based on user input

   filtered_data <- subset(sorted_data)


   # Check if the filtered data is not empty

   if (nrow(filtered_data) > 0) {
```

```
    # Reshape the data for ggplot
    reshaped_data <- reshape2::melt(total_energy_data, id.vars = NULL,
                                    measure.vars = c("Current_Year", "Next_Year", "Percentage_Increase"),
                                    variable.name = "Energy_Type", value.name = "Value")

    # Create a line plot for total energy consumption
    p <- ggplot(reshaped_data, aes(x = Energy_Type, y = Value, color = Energy_Type, linetype = Energy_Type)) +
      geom_line() +
      geom_point() +
      labs(title = "Total Energy Consumption",
           x = NULL,
           y = "Total Energy Consumption") +
      theme_minimal() +
      theme(legend.position = "top") +
      scale_color_manual(values = c("Current_Year" = "#66c2a5", "Next_Year" = "#fc8d62", "Percentage_Increase" = "#8da0cb"))

    # Add labels outside the plot using annotate
    p <- p +
      annotate("text", x = 2, y = max(reshaped_data$Value) - 0.2,
               label = paste("Percentage Increase: ", round(total_energy_data$Percentage_Increase, 2), "%"),
               color = "#8da0cb", size = 4, hjust = 0, vjust = -0.5)

    print(p)
  } else {
    # If filtered data is empty, display a message or a default plot
    ggplot() + theme_void() +
      annotate("text", x = 1, y = 1, label = "No data available for the selected filters.",
```

```
            color = "red", size = 5, hjust = 0.5, vjust = 0.5)
  }
 })

 output$energyCategoryPieChart <- renderPlot({
  # Filter data based on user input
  filtered_data_pie <- subset(sorted_data, in.county == input$in_county_pie & time_split %in% input$time_split_pie)

  # Check if the filtered data is not empty
  if (nrow(filtered_data_pie) > 0) {
   # Summarize data for energy categories
   category_data <- filtered_data_pie %>%
    summarise(
     Kitchen = sum(out.kitchen_energy_consumption),
     Laundry = sum(out.laundry_energy_consumption),
     Heating_Cooling = sum(out.heating_cooling_energy_consumption),
     Water_Heating = sum(out.water_heating_energy_consumption),
     Electrical_Appliances = sum(out.electrical_appliances_energy_consumption),
     Outdoor_Appliances = sum(out.outdoor_appliances_energy_consumption),
     Renewable_Energy = sum(out.renewable_energy_energy_consumption),
     Total_Energy = sum(out.total_energy_consumption)
    )

   # Reshape the data for ggplot
   reshaped_category_data <- reshape2::melt(category_data, id.vars = "Total_Energy", variable.name = "Energy_Category", value.name = "Value")

   # Create a pie chart for energy categories
```

```
    ggplot(reshaped_category_data, aes(x = "", y = Value, fill = Energy_Category)) +
     geom_bar(stat = "identity", width = 1, color = "white") +
     coord_polar("y") +
     labs(title = "Energy Categories Distribution",
          fill = "Energy Category") +
     theme_void() +
     theme(legend.position = "right") +
     scale_fill_manual(values = c(
       "Kitchen" = "#66c2a5",
       "Laundry" = "#fc8d62",
       "Heating_Cooling" = "#8da0cb",
       "Water_Heating" = "#e78ac3",
       "Electrical_Appliances" = "#a6d854",
       "Outdoor_Appliances" = "#ffd92f",
       "Renewable_Energy" = "#66c2a5"
     ))
  } else {
    # If filtered data is empty, display a message or a default plot
    ggplot() + theme_void() +
     annotate("text", x = 0.5, y = 0.5, label = "No data available for the selected filters.",
              color = "red", size = 5, hjust = 0.5, vjust = 0.5)
  }
})



output$new_energyPlot <- renderPlot({
  # Filter data based on user input
  new_filtered_data <- subset(sorted_data,
                              in.weather_file_city == input$new_in_city &
```

```
                    between(time_split_numeric, input$new_time_slider_city[1],
input$new_time_slider_city[2]))


  # Check if the filtered data is not empty
  if (nrow(new_filtered_data) > 0) {
    # Data aggregation
    new_grouped_data <- new_filtered_data %>%
      group_by(time_split, in.weather_file_city) %>%
      summarize(Total_Energy_Demand = sum(Next_year_Pred, na.rm = TRUE),
                Avg_Latitude = mean(in.weather_file_latitude, na.rm = TRUE),
                Avg_Longitude = mean(in.weather_file_longitude, na.rm = TRUE),
                City_Name = first(in.weather_file_city[!is.na(in.weather_file_city)])) %>%
      ungroup()


    # Create a bar plot for total energy consumption with vibrant colors
    ggplot(new_grouped_data, aes(x = time_split, y = Total_Energy_Demand, fill = in.weather_file_city))
+
      geom_bar(stat = "identity", position = "dodge", width = 0.7) +
      labs(title = "Aggregated Energy Demand",
           x = "Time of Day",
           y = "Total Energy Demand",
           fill = "City") +
      theme_minimal() +
      theme(legend.position = "top") +
      scale_fill_viridis_d(option = "A", direction = -1)  # Adjust the option and direction as needed
  } else {
    # If filtered data is empty, display a message or a default plot
    ggplot() + theme_void() +
      annotate("text", x = 0.5, y = 0.5, label = "No data available for the selected filters.",
```

```
            color = "red", size = 5, hjust = 0.5, vjust = 0.5)
  }
})
```

```
# Your data processing code here (Summarize, Join, Create Palette, etc.)
# Create a palette for time zones
zone_palette <- c("blue", "green", "yellow", "orange", "red", "purple")  # Corresponding to 1-6
summary_data<- sorted_data %>%
  group_by(bldg_id) %>%
  summarize(max_time_zone = time_split_numeric[which.max(out.total_energy_consumption)])

# Join summary data with the original dataset to retain all rows for each bldg_id
sorted_data1 <- left_join(sorted_data, summary_data, by = "bldg_id")

# Render Leaflet map for current year consumption
output$map_current <- renderLeaflet({
  leaflet(data = sorted_data1) %>%
    addTiles() %>%
    addCircleMarkers(
      lat = ~in.weather_file_latitude,
      lng = ~in.weather_file_longitude,
      popup = ~paste("House Number: ", bldg_id, "<br>",
                     "Time Zone: ", max_time_zone, "<br>",
                     "Current Year Consumption: ", out.total_energy_consumption, "kWh"),
      color = ~zone_palette[as.integer(max_time_zone)],
```

```
    fillOpacity = 0.7
  ) %>%
  addLegend(
    position = "bottomright",
    colors = zone_palette,
    labels = c("Late Night", "Early Morning", "Morning", "Noon", "Evening", "Night"),
    title = "Time Zones"
  )
})

# Render Leaflet map for future consumption predictions
output$map_future <- renderLeaflet({
 leaflet(data = sorted_data1) %>%
  addTiles() %>%
  addCircleMarkers(
    lat = ~in.weather_file_latitude,
    lng = ~in.weather_file_longitude,
    popup = ~paste("House Number: ", bldg_id, "<br>",
                   "Time Zone: ", max_time_zone, "<br>",
                   "Future Consumption Prediction: ", Next_year_Pred, "kWh"),
    color = ~zone_palette[as.integer(max_time_zone)],
    fillOpacity = 0.7
  ) %>%
  addLegend(
    position = "bottomright",
    colors = zone_palette,
    labels = c("Late Night", "Early Morning", "Morning", "Noon", "Evening", "Night"),
    title = "Time Zones"
  )
```

```
})

# ... (previous code)

# New Leaflet map for current year consumption in the fourth tabPanel
output$map_current_leaflet <- renderLeaflet({
  leaflet(data = sorted_data1) %>%
    addTiles() %>%
    addCircleMarkers(
      lat = ~in.weather_file_latitude,
      lng = ~in.weather_file_longitude,
      popup = ~paste("House Number: ", bldg_id, "<br>",
                     "Time Zone: ", max_time_zone, "<br>",
                     "Current Year Consumption: ", out.total_energy_consumption, "kWh"),
      color = ~zone_palette[as.integer(max_time_zone)],
      fillOpacity = 0.7
    ) %>%
    addLegend(
      position = "bottomright",
      colors = zone_palette,
      labels = c("Late Night", "Early Morning", "Morning", "Noon", "Evening", "Night"),
      title = "Time Zones"
    ) %>%
    addControl(
      html = '<div style="position: absolute; top: 10px; right: 10px; background: white; padding: 5px;
border: 1px solid gray; border-radius: 5px;">Current Year Consumption</div>',
      position = "topright",
      layerId = "current_info"
    )
```

```
})

# New Leaflet map for future consumption predictions in the fourth tabPanel
output$map_future_leaflet <- renderLeaflet({
  leaflet(data = sorted_data1) %>%
    addTiles() %>%
    addCircleMarkers(
      lat = ~in.weather_file_latitude,
      lng = ~in.weather_file_longitude,
      popup = ~paste("House Number: ", bldg_id, "<br>",
                     "Time Zone: ", max_time_zone, "<br>",
                     "Future Consumption Prediction: ", Next_year_Pred, "kWh"),
      color = ~zone_palette[as.integer(max_time_zone)],
      fillOpacity = 0.7
    ) %>%
    addLegend(
      position = "bottomright",
      colors = zone_palette,
      labels = c("Late Night", "Early Morning", "Morning", "Noon", "Evening", "Night"),
      title = "Time Zones"
    ) %>%
    addControl(
      html = '<div style="position: absolute; top: 10px; right: 10px; background: white; padding: 5px;
border: 1px solid gray; border-radius: 5px;">Future Expected Consumption</div>',
      position = "topright",
      layerId = "future_info"
    )
})
```

```
 # Plot A: Total Energy Consumption by Income Category
 output$plot_a <- renderPlot({
  total_energy_consumption <- model %>%
    mutate(income_category = case_when(
     in.income <= 39999 ~ "Low",
     in.income > 39999 & in.income <= 99999 ~ "Middle",
     in.income > 99999 ~ "High"
    )) %>%
    group_by(income_category) %>%
    summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))

  total_energy_consumption$income_category <- factor(total_energy_consumption$income_category,
levels = c("Low", "Middle", "High"))

  ggplot(total_energy_consumption, aes(x = TotalEnergyConsumption, y = income_category, fill =
income_category)) +
    geom_bar(stat = "identity") +
    labs(title = "Total Energy Consumption by Income Category",
        x = "Total Energy Consumption",
        y = "Income Category") +
    theme_minimal() +
    scale_fill_manual(values = c("#A6CEE3", "#FFD700", "#98FB98"))  # Customize pastel color values
 })

 # Plot B: Total Energy Consumption by Square Feet Category
 output$plot_b <- renderPlot({
  quantiles <- quantile(model$in.sqft, probs = c(0, 0.25, 0.75, 1), na.rm = TRUE)

  model$sqft_category <- cut(model$in.sqft,
```

```
                    breaks = quantiles,
                    labels = c("Low", "Medium", "High"),
                    include.lowest = TRUE)

  total_energy_by_sqft <- model %>%
    group_by(sqft_category) %>%
    summarize(TotalEnergy = sum(out.total_energy_consumption, na.rm = TRUE)) %>%
    ungroup()

  ggplot(total_energy_by_sqft, aes(x = TotalEnergy, y = sqft_category, fill = sqft_category)) +
    geom_bar(stat = "identity") +
    labs(title = "Total Energy Consumption by Square Feet Category",
         x = "Total Energy Consumption",
         y = "Sqft Category") +
    theme_minimal() +
    scale_fill_manual(values = c("#FFB6C1", "#87CEEB", "#98FB98"))  # Customize pastel color values
})


# Encoding in.federal_poverty_level
model$encoded_poverty_level <- case_when(
  model$in.federal_poverty_level %in% c(1, 2, 3, 4, 5) ~ as.character(model$in.federal_poverty_level),
  TRUE ~ "Other"
)

# Encoding in.heating_fuel
model$encoded_heating_fuel <- case_when(
  model$in.heating_fuel %in% c(1, 2, 3, 4, 5) ~ as.character(model$in.heating_fuel),
  TRUE ~ "Other"
```

```
)

# Debug Print Statements
cat("Unique Values in in.federal_poverty_level:", unique(model$in.federal_poverty_level), "\n")
cat("Unique Values in encoded_poverty_level:", unique(model$encoded_poverty_level), "\n")
cat("Unique Values in in.heating_fuel:", unique(model$in.heating_fuel), "\n")
cat("Unique Values in encoded_heating_fuel:", unique(model$encoded_heating_fuel), "\n")



# Plot C: Total Energy Consumption for Different Federal Poverty Level
output$plot_c <- renderPlot({
  total_energy_bypoverty <- model %>%
    group_by(encoded_poverty_level) %>%
    summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))


  ggplot(total_energy_bypoverty, aes(x = TotalEnergyConsumption, y = reorder(encoded_poverty_level,
-TotalEnergyConsumption), fill = encoded_poverty_level)) +
    geom_col() +
    labs(title = "Total Energy Consumption for Different Federal Poverty Level",
        x = "Total Energy Consumption",
        y = "Federal Poverty Level") +
    theme_minimal() +
    scale_fill_manual(values = c("#FFDAB9", "#87CEEB", "#98FB98", "#FF69B4", "#FFA07A", "#FF6347"))
+  # Customize pastel color values
    annotate("text", x = -Inf, y = Inf, label = "Federal Poverty Level Encoding:", vjust = 2, hjust = 0, size = 4)
+
    annotate("text", x = -Inf, y = Inf, label = "1: 0-100%, 2: 100-150%, 3: 150-200%, 4: 200-300%, 5: 300-
400%, Other: Other", vjust = 1, hjust = 0, size = 3)
 })
```

```
  # Plot D: Total Energy Consumption for Different Fuel Type
  output$plot_d <- renderPlot({
    total_energy_by_Heatingfuel <- model %>%
      group_by(encoded_heating_fuel) %>%
      summarize(TotalEnergyConsumption = sum(out.total_energy_consumption, na.rm = TRUE))


    ggplot(total_energy_by_Heatingfuel, aes(x = TotalEnergyConsumption, y =
reorder(encoded_heating_fuel, -TotalEnergyConsumption), fill = encoded_heating_fuel)) +
      geom_col() +
      labs(title = "Total Energy Consumption for Different Fuel Type",
           x = "Total Energy Consumption",
           y = "Fuel Type") +
      theme_minimal() +
      scale_fill_manual(values = c("#FFDAB9", "#87CEEB", "#98FB98", "#FF69B4", "#FFA07A", "#FF6347"))
+  # Include color for "Other"
      annotate("text", x = -Inf, y = Inf, label = "Fuel Type Encoding:", vjust = 2, hjust = 0, size = 4) +
      annotate("text", x = -Inf, y = Inf, label = "1: Electricity, 2: Fuel Oil, 3: Natural Gas, 4: Other Fuel, 5:
Propane, Other: Other", vjust = 1, hjust = 0, size = 3)
  })




  # Arrange plots in a 2x2 grid side by side
  output$plots_2x2_grid <- renderPlotly({
    subplot(
      plotly::plotlyOutput("plot_a"),
      plotly::plotlyOutput("plot_b"),
      plotly::plotlyOutput("plot_c"),
      plotly::plotlyOutput("plot_d"),
      nrows = 2, margin = 0.05
```

```
  )
 })

}
# Run the Shiny app
shinyApp(ui = ui, server = server)
```

SUGGESTED IMPACT AND MODELING THE IMPACT

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.3     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.3     ✓ tidyr     1.3.0
## ✓ 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 becom
e errors
```

```
Ordinality_File <- read.csv("C:/Users/Soundarya Ravi/Desktop/Shiny/Team2_Final_SEW_Ordinal_Model
ling1.csv")
View(Ordinality_File)
```

```
columns_to_remove <- c("Next_year_Pred", "Change_in_energy")
sorted_data_for_IJ<- Ordinality_File[, !(names(Ordinality_File) %in% columns_to_remove)]
```

```
columns_to_drop <- c(
  "out.kitchen_energy_consumption",
  "out.laundry_energy_consumption",
  "out.heating_cooling_energy_consumption",
  "out.water_heating_energy_consumption",
  "out.electrical_appliances_energy_consumption",
  "out.renewable_energy_energy_consumption",
  "out.outdoor_appliances_energy_consumption"
)
sorted_data_for_IJ <- sorted_data_for_IJ[, !(names(sorted_data_for_IJ) %in% columns_to_drop)]
```

```
# Ordinal coding for in.hvac_cooling_efficiency
#in_hvac_cooling_efficiency_mapping <- c(
#  "AC, SEER 15"=1, "AC, SEER 13"=2, "None"=0, "AC, SEER 10"=4,
#  "Heat Pump"=5, "Room AC, EER 10.7"=6, "Room AC, EER 8.5"=7,
#  "AC, SEER 8"=8, "Room AC, EER 9.8"=9, "Room AC, EER 12.0"=10
#)
#static_house_filtered$in.hvac_cooling_efficiency <- #as.numeric(in_hvac_cooling_efficiency_mapp
ing[static_house_filtered$in.hvac_cooling_efficien#cy])

## Ordinal coding for in.cooling_setpoint (assuming the given order)
#in_cooling_setpoint_mapping <- c(
 # "72F"=7, "76F"=9, "70F"=6, "60F"=1, "78F"=10, "75F"=8, "68F"=5, "62F"=2, "65F"=3,
  #"80F"=11, "67F"=4
#)
```

```
sorted_data_for_IJ$Dry_Bulb_Temperature_C <- sorted_data_for_IJ$Dry_Bulb_Temperature_C + 5
```

```
set.seed(123)  # Setting seed for reproducibility
unique_bldg_ids <- unique(sorted_data_for_IJ$bldg_id)
unique_values <- c(10, 6, 5, 1, 2)

# Shuffle the unique values for randomness
shuffled_values <- sample(unique_values)

# Create a mapping between bldg_id and shuffled_values
value_mapping <- rep(shuffled_values, length.out = length(unique_bldg_ids))

# Assign the values to the in.hvac_cooling_efficiency column
sorted_data_for_IJ$in.hvac_cooling_efficiency <- value_mapping[match(sorted_data_for_IJ$bldg_id,
unique_bldg_ids)]
```

```
library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
# Test and Train Data
set.seed(0) # Set seed for generating random data.

# CreateDataPartition() function from the caret package to split the original dataset into a tra
ining and testing set
# Split data into training (80%) and testing set (20%)
parts <- createDataPartition(sorted_data_for_IJ$out.total_energy_consumption, p = 0.8, list = FA
LSE)
train <- sorted_data_for_IJ[parts,]
test <-  sorted_data_for_IJ[-parts,]

# Define predictor and response variables in the training set
train_x <- data.matrix(train[, -which(names(train) == "out.total_energy_consumption")])
train_y <- train[["out.total_energy_consumption"]]

# Define predictor and response variables in the testing set
test_x <- data.matrix(test[, -which(names(train) == "out.total_energy_consumption")])
test_y <- test[["out.total_energy_consumption"]]

print(c("Length of train_y:", length(train_y)))
```

```
## [1] "Length of train_y:" "27409"
```

```
print(c("Number of rows in train_x:", nrow(train_x)))
```

```
## [1] "Number of rows in train_x:" "27409"
```

```
# Check if lengths match before creating xgb.DMatrix
stopifnot(length(train_y) == nrow(train_x))

# Continue with the rest of your code...
#define final training and testing sets
xgb_train = xgb.DMatrix(data = train_x, label = train_y)
xgb_test = xgb.DMatrix(data = test_x, label = test_y)
```

```
#defining a watchlist
watchlist = list(train=xgb_train, test=xgb_test)

#fit XGBoost model and display training and testing data at each iteartion
model = xgb.train(data = xgb_train, max.depth = 3, watchlist=watchlist, nrounds = 100)
```

```
## [1]  train-rmse:0.833092 test-rmse:0.842933
## [2]  train-rmse:0.680968 test-rmse:0.691506
## [3]  train-rmse:0.585979 test-rmse:0.596611
## [4]  train-rmse:0.518126 test-rmse:0.530523
## [5]  train-rmse:0.476218 test-rmse:0.488790
## [6]  train-rmse:0.446253 test-rmse:0.457011
## [7]  train-rmse:0.424445 test-rmse:0.434300
## [8]  train-rmse:0.403667 test-rmse:0.412880
## [9]  train-rmse:0.388399 test-rmse:0.397403
## [10] train-rmse:0.374278 test-rmse:0.382507
## [11] train-rmse:0.363412 test-rmse:0.371542
## [12] train-rmse:0.351793 test-rmse:0.359639
## [13] train-rmse:0.342499 test-rmse:0.349885
## [14] train-rmse:0.334069 test-rmse:0.342189
## [15] train-rmse:0.327858 test-rmse:0.336103
## [16] train-rmse:0.321776 test-rmse:0.330337
## [17] train-rmse:0.317107 test-rmse:0.326245
## [18] train-rmse:0.312281 test-rmse:0.321411
## [19] train-rmse:0.306833 test-rmse:0.315967
## [20] train-rmse:0.301131 test-rmse:0.311059
## [21] train-rmse:0.297154 test-rmse:0.306879
## [22] train-rmse:0.293112 test-rmse:0.303141
## [23] train-rmse:0.290468 test-rmse:0.300729
## [24] train-rmse:0.287294 test-rmse:0.297740
## [25] train-rmse:0.284692 test-rmse:0.295810
## [26] train-rmse:0.281242 test-rmse:0.292274
## [27] train-rmse:0.279173 test-rmse:0.290263
## [28] train-rmse:0.277378 test-rmse:0.288521
## [29] train-rmse:0.275524 test-rmse:0.287134
## [30] train-rmse:0.274000 test-rmse:0.286048
## [31] train-rmse:0.272365 test-rmse:0.284250
## [32] train-rmse:0.270603 test-rmse:0.282245
## [33] train-rmse:0.269128 test-rmse:0.280679
## [34] train-rmse:0.267258 test-rmse:0.278790
## [35] train-rmse:0.265728 test-rmse:0.277065
## [36] train-rmse:0.264366 test-rmse:0.275657
## [37] train-rmse:0.263226 test-rmse:0.274820
## [38] train-rmse:0.261472 test-rmse:0.273376
## [39] train-rmse:0.260419 test-rmse:0.272303
## [40] train-rmse:0.259678 test-rmse:0.271532
## [41] train-rmse:0.258012 test-rmse:0.269774
## [42] train-rmse:0.256728 test-rmse:0.268644
## [43] train-rmse:0.255991 test-rmse:0.268100
## [44] train-rmse:0.254766 test-rmse:0.266978
## [45] train-rmse:0.253874 test-rmse:0.266013
## [46] train-rmse:0.252328 test-rmse:0.264839
## [47] train-rmse:0.251204 test-rmse:0.263574
## [48] train-rmse:0.250552 test-rmse:0.263162
## [49] train-rmse:0.249871 test-rmse:0.262524
## [50] train-rmse:0.249188 test-rmse:0.261832
## [51] train-rmse:0.248113 test-rmse:0.260812
## [52] train-rmse:0.247490 test-rmse:0.259977
```

```
## [53] train-rmse:0.246849 test-rmse:0.259555
## [54] train-rmse:0.245920 test-rmse:0.258856
## [55] train-rmse:0.245414 test-rmse:0.258346
## [56] train-rmse:0.243692 test-rmse:0.256744
## [57] train-rmse:0.241820 test-rmse:0.255065
## [58] train-rmse:0.241120 test-rmse:0.254471
## [59] train-rmse:0.240394 test-rmse:0.254053
## [60] train-rmse:0.239688 test-rmse:0.253589
## [61] train-rmse:0.239182 test-rmse:0.253298
## [62] train-rmse:0.238772 test-rmse:0.252994
## [63] train-rmse:0.238179 test-rmse:0.252232
## [64] train-rmse:0.237656 test-rmse:0.251733
## [65] train-rmse:0.237129 test-rmse:0.251182
## [66] train-rmse:0.236576 test-rmse:0.250802
## [67] train-rmse:0.236035 test-rmse:0.250667
## [68] train-rmse:0.235641 test-rmse:0.250267
## [69] train-rmse:0.235291 test-rmse:0.249826
## [70] train-rmse:0.234622 test-rmse:0.249365
## [71] train-rmse:0.234056 test-rmse:0.249062
## [72] train-rmse:0.233671 test-rmse:0.248838
## [73] train-rmse:0.233262 test-rmse:0.248478
## [74] train-rmse:0.232923 test-rmse:0.248056
## [75] train-rmse:0.232604 test-rmse:0.247801
## [76] train-rmse:0.231851 test-rmse:0.247106
## [77] train-rmse:0.231537 test-rmse:0.246938
## [78] train-rmse:0.231271 test-rmse:0.246692
## [79] train-rmse:0.231036 test-rmse:0.246438
## [80] train-rmse:0.230709 test-rmse:0.246321
## [81] train-rmse:0.230416 test-rmse:0.246143
## [82] train-rmse:0.229999 test-rmse:0.245719
## [83] train-rmse:0.229533 test-rmse:0.245265
## [84] train-rmse:0.228986 test-rmse:0.244681
## [85] train-rmse:0.228608 test-rmse:0.244493
## [86] train-rmse:0.228271 test-rmse:0.244368
## [87] train-rmse:0.227731 test-rmse:0.244313
## [88] train-rmse:0.227591 test-rmse:0.244173
## [89] train-rmse:0.227385 test-rmse:0.243960
## [90] train-rmse:0.226698 test-rmse:0.243547
## [91] train-rmse:0.226413 test-rmse:0.243395
## [92] train-rmse:0.226172 test-rmse:0.243173
## [93] train-rmse:0.225961 test-rmse:0.243045
## [94] train-rmse:0.225787 test-rmse:0.242876
## [95] train-rmse:0.225598 test-rmse:0.242797
## [96] train-rmse:0.225354 test-rmse:0.242693
## [97] train-rmse:0.225031 test-rmse:0.242172
## [98] train-rmse:0.224555 test-rmse:0.241666
## [99] train-rmse:0.224158 test-rmse:0.241266
## [100]    train-rmse:0.223957 test-rmse:0.241104
```

```
#use model to make predictions on test data
pred_y <-  predict(model, xgb_test)
pred_x <-  predict(model, xgb_train)
```

```
# Assuming pred_y is your predicted values

# Calculate Mean Squared Error (MSE)
mse <- mean((test_y - pred_y)^2)
cat('Mean Squared Error (MSE): ', round(mse, 3), '\n')
```

```
## Mean Squared Error (MSE):  0.058
```

```
# Calculate Root Mean Squared Error (RMSE) using caret package
rmse <- caret::RMSE(test_y, pred_y)
cat('Root Mean Squared Error (RMSE): ', round(rmse, 3), '\n')
```

```
## Root Mean Squared Error (RMSE):  0.241
```

```
# Calculate R-squared
y_test_mean <- mean(test_y)
tss <- sum((test_y - y_test_mean)^2)
rss <- sum((test_y - pred_y)^2)  # Using predicted values to calculate residuals
rsq <- 1 - (rss/tss)
cat('The R-squared of the test data is ', round(rsq, 3), '\n')
```

```
## The R-squared of the test data is  0.88
```

```
predictions_xgb <- predict(model, newdata = xgb_test)

mape <- mean(abs((test$out.total_energy_consumption - predictions_xgb) / test$out.total_energy_c
onsumption )) * 100

# Print the result
print(paste("MAPE:", mape))
```

```
## [1] "MAPE: 13.9243384816242"
```

```
train$Possible_New_Energy <- pred_x
test$Possible_New_Energy <- pred_y
```

```
combined_df <- rbind(train,test)
```

```
sorted_combined_df <- combined_df[order(combined_df$bldg_id, combined_df$time_split), ]
```

```
sorted_combined_df$New_Change <- sorted_combined_df$Possible_New_Energy - sorted_combined_df$ou
t.total_energy_consumption
```

```
Percentage_difference <- (sum(sorted_combined_df$New_Change)/sum(sorted_combined_df$out.total_en
ergy_consumption)) * 100
Percentage_difference
```

```
## [1] -0.05292819
```