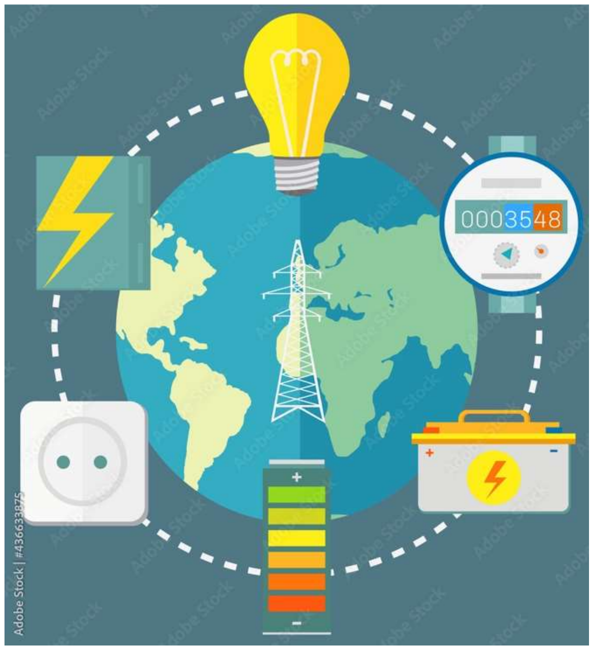
IST 687 – INTRODUCTION TO DATA SCIENCE

GROUP – 5

Optimizing Electricity Demand for eSC: A Data-Driven Approach



Project By:
Vishnu Charugundla
Lekhith Reddy Kambham
Likhith Kolli
Prathyusha Murala
Austin Rodrigues

INTRODUCTION

The need for sustainable practices and an increasing awareness of climate change are driving changes in the energy environment. Our project's main goal is to assist our client, an energy business (eSC), in addressing the problems caused by climate change, particularly as they relate to the need for power. Residential properties in South Carolina and a small area of North Carolina are served by eSC, which expects higher summertime demand, especially in July.

In response to the increasing concerns about the impact of global warming on energy demand, our team has undertaken a comprehensive analysis for our esteemed client, the energy company (eSC). By focusing on supplying electricity to residential properties in South Carolina and a small area of North Carolina, eSC hopes to reduce the possibility of blackouts during times of high demand, especially in July.

The primary goal is to comprehend the principal factors that influence energy consumption and create tactics to motivate consumers to conserve energy. eSC's goal for 'extra hot' summers is to reduce energy usage rather than expand energy production facilities, which is in line with their commitment to environmental sustainability and customer happiness.

PROJECT SCOPE

Problem Statement:

The energy company (eSC) faces a critical challenge in effectively managing the increased demand for electricity, especially during the summer months, driven by a growing awareness of climate change and environmental sustainability. The concern is that the current energy infrastructure may be insufficient to meet peak demand, leading to potential blackouts. Rather than resorting to the construction of new power plants, eSC seeks to understand and mitigate the factors influencing energy usage, particularly in July, and encourages consumers to adopt energy-saving practices.

Objectives:

1. **Optimize Energy Usage:**
   - Develop strategies to optimize energy consumption during peak periods, specifically targeting the summer month of July.
2. **Identify Key Drivers:**
   - Analyze static house data, energy usage data, and weather data to identify the key drivers of energy consumption in residential properties.
3. **Encourage Energy-Saving Practices:**
   - Devise measures to encourage consumers to adopt energy-saving practices, aligning with the company's goal of reducing demand during critical periods.
4. **Model Future Energy Demand:**
   - Build predictive models that forecast energy usage for a given hour in the month of July, considering various factors such as house attributes and weather conditions.
5. **Evaluate Model Accuracy:**
   - Assess the accuracy of predictive models using performance metrics such as Mean Absolute Error (MAE) and R-squared, ensuring robust and reliable predictions.
6. **Simulate 'Extra Hot' Conditions:**

- Modify weather data to simulate 'extra hot' conditions by increasing July temperatures by 5 degrees, enabling the assessment of energy demand under more extreme scenarios.

7. **Predict Future Peak Energy Demand:**
   - Utilize the best-performing model to predict and evaluate future peak energy demand, considering different geographic regions and relevant attributes.
8. **Identify Energy Demand Reduction Strategies:**
   - Explore potential data-driven approaches to reduce peak energy demand and model their impact, providing actionable insights for eSC.
9. **Ensure Sustainability and Customer Satisfaction:**
   - Align strategies and recommendations with the dual objectives of sustainability and customer satisfaction, ensuring a balanced and responsible approach to energy management.

UNDERSTANDING DATA PROVIDED

This section provides a detailed description of the data used in the project:

Static House Data:

This dataset contains basic information for approximately 5,000 single-family houses served by eSC. The data format is Parquet, an optimized format for storing large datasets efficiently. The information includes:

- **Building ID:** Unique identifier for each house used to access corresponding energy data.
- **House attributes:** Features such as size, age, construction type, number of bedrooms and bathrooms, etc.
- **Geographic information:** County and ZIP code.

Energy Usage Data:

This dataset contains hourly energy usage data for each house in the static data set. Each house has a separate file identified by its building ID. The data format is also Parquet. The data includes:

- **Timestamp:** Date and time of the energy measurement.
- **Energy consumption:** Total energy used in the house in kilowatt-hours (kWh).
- **Individual appliance usage:** Optional data on specific appliances, such as air conditioning, dryer, etc. (available in some files).

Meta Data:

This file provides a human-readable explanation of the data fields used in both static and energy usage datasets. It serves as a dictionary to understand the meaning and units of each data point.

Weather Data:

This dataset contains hourly weather information for each of the 50 counties in the study area. Each county has a separate file identified by its county code. The data format is a simple CSV file. The information includes:

- **Timestamp:** Date and time of the weather measurement.
- **Weather variables:** Temperature, humidity, wind speed, precipitation, etc.

METHODOLOGY

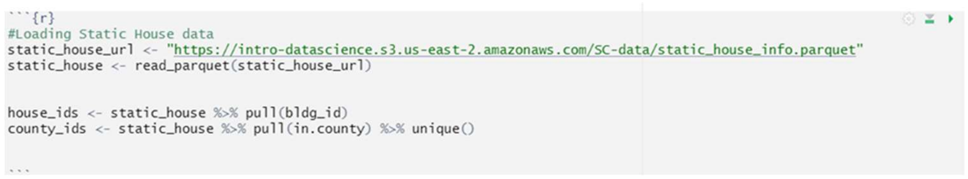
1. **Data Preparation:**
    a. Houses and energy usage data were merged based on building IDs.
    b. This merged data was merged with the weather data using the county IDs.
    c. Cleaning the data to provide refined results with the best accuracy.
    d. Exploratory data analysis was performed to understand data characteristics and relationships.
    e. Feature engineering was conducted to create features relevant to energy prediction.
2. **Model Building:**
    a. Various models were trained and evaluated, including linear regression, random forest, XGBoost.
    b. The best performing model was selected based on accuracy metrics like mean squared error and R-squared.
3. **Model Evaluation and Future Demand Prediction:**
    a. The chosen model's accuracy was evaluated and analysed for limitations.
    b. A modified weather dataset with July temperatures increased by 5 degrees was created.
    c. The model was used to predict peak energy demand for different geographic regions and other relevant dimensions.
4. **Shiny Application Development:**
    a. An interactive Shiny application was developed to allow eSC to:
        i. Visualize data and model predictions.
        ii. Explore future energy demand scenarios under different conditions.
5. **Potential Energy Reduction Strategies:**
    a. Data analysis and domain knowledge were used to identify potential approaches to reduce peak energy demand.
    b. These strategies were modelled and their impact on peak demand was quantified.

DATA PREPARATION

Data Reading and Merging:

Static House Data loading:

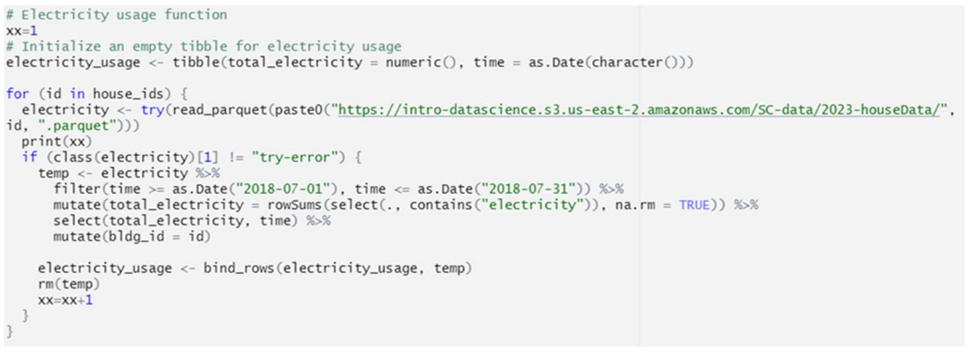
Our first step involves loading the fundamental details about the houses. This information, stored in a Parquet format file, includes critical identifiers like building IDs (bldg_id) and county details. These identifiers serve as the key link between the static house data, electricity consumption and weather data.



~~~
```{r}
#Loading Static House data
static_house_url <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/static_house_info.parquet"
static_house <- read_parquet(static_house_url)


house_ids <- static_house %>% pull(bldg_id)
county_ids <- static_house %>% pull(in.county) %>% unique()


```
~~~

Streamlining Energy Data for Each Building:

The system processes energy data for each building individually through a dedicated pipeline. This pipeline iterates through each building ID, retrieving corresponding data from Parquet files. To ensure smooth operation, robust error handling tackles any data loading issues. For each building, the pipeline calculates total energy consumption and retains crucial timestamps. This structured approach efficiently processes large amounts of data, paving the way for further analysis.

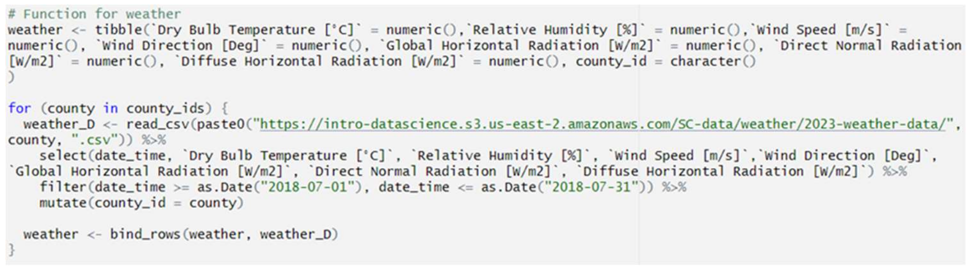


```
# Electricity usage function
xx=1
# Initialize an empty tibble for electricity usage
electricity_usage <- tibble(total_electricity = numeric(), time = as.Date(character()))

for (id in house_ids) {
  electricity <- try(read_parquet(paste0("https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/2023-houseData/",
id, ".parquet")))
  print(xx)
  if (class(electricity)[1] != "try-error") {
    temp <- electricity %>%
      filter(time >= as.Date("2018-07-01"), time <= as.Date("2018-07-31")) %>%
      mutate(total_electricity = rowSums(select(., contains("electricity")), na.rm = TRUE)) %>%
      select(total_electricity, time) %>%
      mutate(bldg_id = id)

    electricity_usage <- bind_rows(electricity_usage, temp)
    rm(temp)
    xx=xx+1
  }
}
```

The following function efficiently iterates through each building in the dataset, retrieves its electricity data from individual Parquet files, and calculates the total electricity consumption for each hour in July. Missing values are ignored during this process. The results are then consolidated into a single data structure containing columns for building ID, date, and total electricity usage. This process ensures efficient data handling and preparation for further analysis.

Integrating Weather Data into the Analysis:

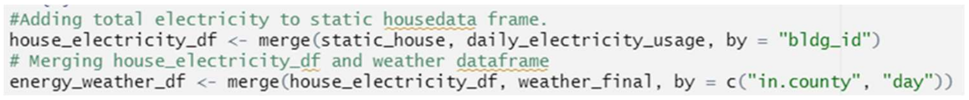
In tandem with processing electricity data, our project also integrates weather data, focusing particularly on temperature variations across different counties. This weather data, provided in CSV format, includes comprehensive temperature readings for each location. We process this information by iterating through each county and meticulously aligning the weather data with our other datasets using the county identifier as the key. This critical step enables us to analyze the influence of weather on electricity consumption patterns with high precision, thereby contributing significantly to our overall understanding of energy demand.



```r
# Function for weather
weather <- tibble(`Dry Bulb Temperature [°C]` = numeric(),`Relative Humidity [%]` = numeric(),`Wind Speed [m/s]` =
numeric(), `Wind Direction [Deg]` = numeric(), `Global Horizontal Radiation [W/m2]` = numeric(), `Direct Normal Radiation
[W/m2]` = numeric(), `Diffuse Horizontal Radiation [W/m2]` = numeric(), county_id = character()
)

for (county in county_ids) {
  weather_D <- read_csv(paste0("https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/weather/2023-weather-data/",
county, ".csv")) %>%
    select(date_time, `Dry Bulb Temperature [°C]`, `Relative Humidity [%]`, `Wind Speed [m/s]`,`Wind Direction [Deg]`,
`Global Horizontal Radiation [W/m2]`, `Direct Normal Radiation [W/m2]`, `Diffuse Horizontal Radiation [W/m2]`) %>%
    filter(date_time >= as.Date("2018-07-01"), date_time <= as.Date("2018-07-31")) %>%
    mutate(county_id = county)

  weather <- bind_rows(weather, weather_D)
}
```

Merging and Finalizing the Data Preparation:

The final act of the data preparation stage involves merging the processed datasets. This entails aligning the electricity consumption data with both the static house data and the weather data, thereby creating a comprehensive and meticulously organized dataset that serves as the foundation for our subsequent analysis. This process culminates in a clean, well-structured dataset encapsulating all necessary variables, readily prepared for in-depth exploration and insights.

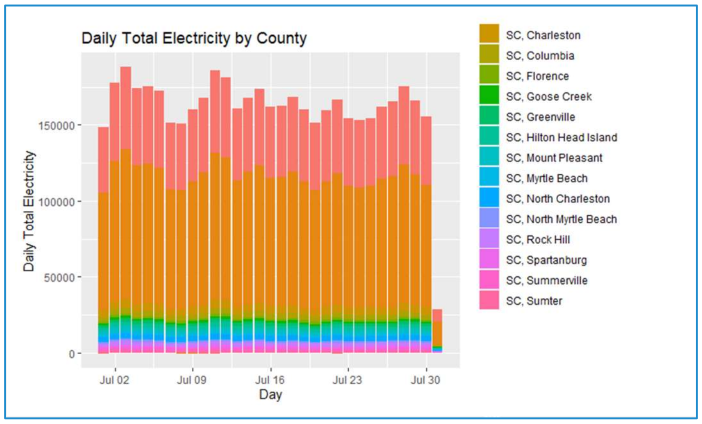


```r
#Adding total electricity to static housedata frame.
house_electricity_df <- merge(static_house, daily_electricity_usage, by = "bldg_id")
# Merging house_electricity_df and weather dataframe
energy_weather_df <- merge(house_electricity_df, weather_final, by = c("in.county", "day"))
```

EXPLORATORY DATA ANALYSIS (EDA)

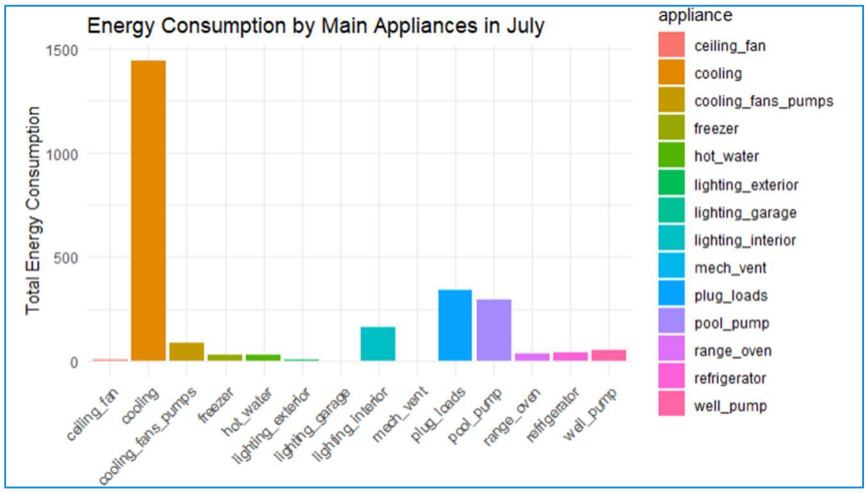
Visualizations:

1. **Daily Total Electricity Usage by Counties:**



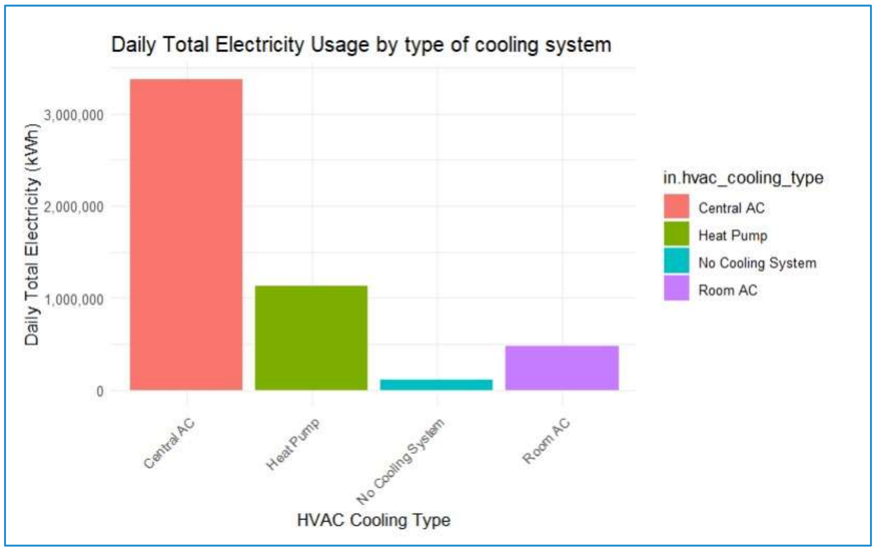
This graph is a representation of the daily total electricity usage by county in South Carolina for the month of July. The x-axis represents the day of the month, while the y-axis represents the daily total electricity usage. The different colors represent the different counties in South Carolina. The graph shows that the daily total electricity usage is highest in the counties of Charleston and Sumter, while it is lowest in the counties of Florence and Rock Hill. This information can be used to analyze the electricity consumption patterns of different counties in South Carolina during the month of July. It can also be used to identify the counties that consume the most and least electricity, which can be useful for energy conservation efforts.

2. Energy Consumption of Appliances in July:



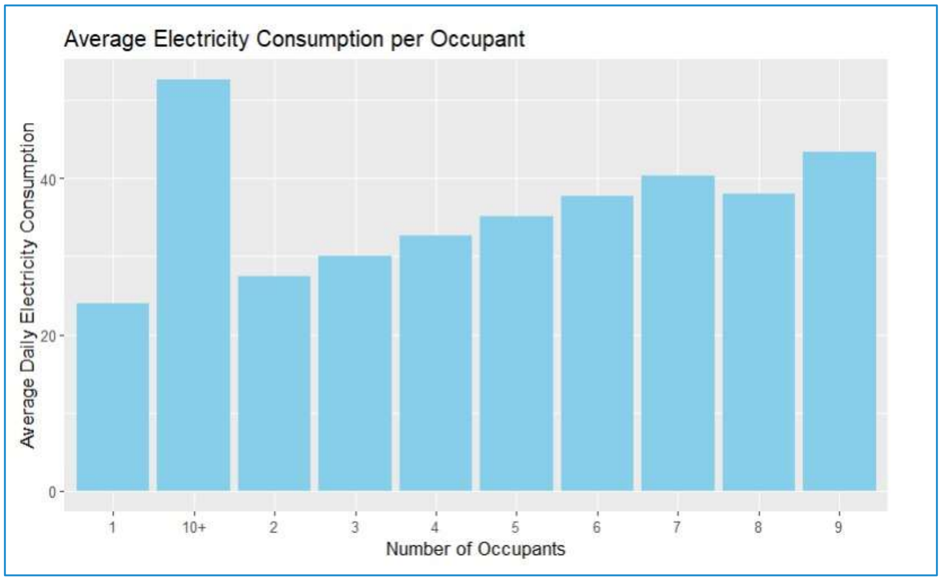
July's energy consumption reveals striking variations among appliances, with cooling systems consuming the highest electricity out of all the appliances. combined, likely fueled by the warm summer weather. In stark contrast, lighting_garage and mech_vent used the least energy. This data underlines the need for efficient energy usage and highlights opportunities for targeted energy-saving strategies.

3. Daily Total Electricity Usage by Type of Cooling System:



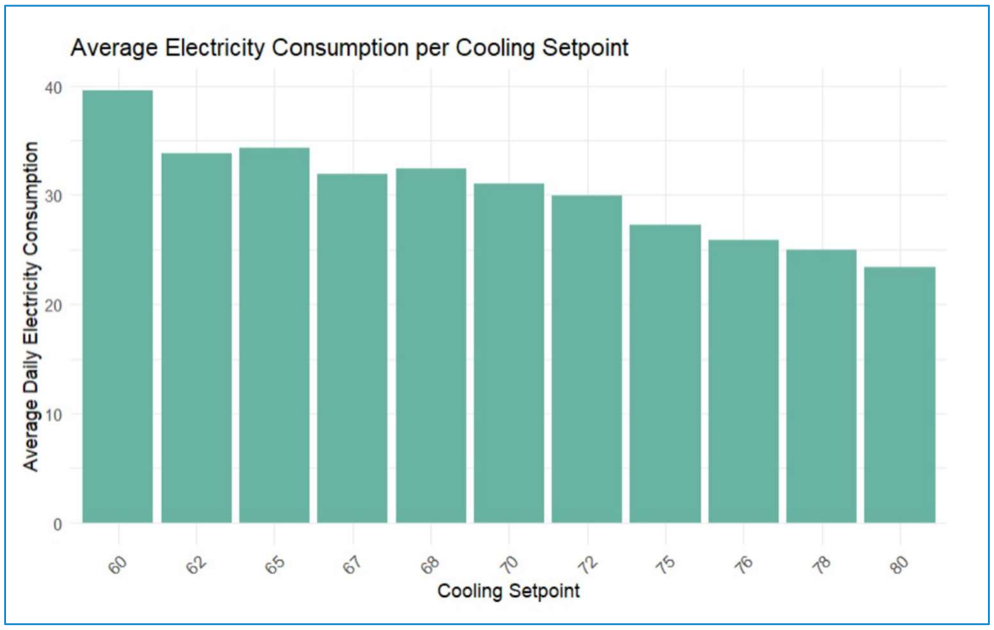
Central air conditioning systems reigned supreme, consuming the most electricity for cooling throughout the month of July. While heat pumps, room ACs, and homes without cooling systems followed behind, daily consumption across all types other than Central AC was significantly less. This highlights the critical need to promote energy-efficient alternatives and reduce overall cooling energy consumption.

4. **Average Electricity Consumption per Occupant:**



As the number of occupants in a household increase, the average electricity consumption per person decreases significantly. Strikingly, single occupants consume the most electricity in a month, while households with four occupants or more share the lowest usage a month. This trend suggests greater resource sharing and reliance on shared appliances in multi-occupant dwellings. Understanding this dynamic can be key to developing targeted energy-saving strategies tailored to different household sizes and needs.

5. **Average Energy Consumption per Cooling Checkpoint:**

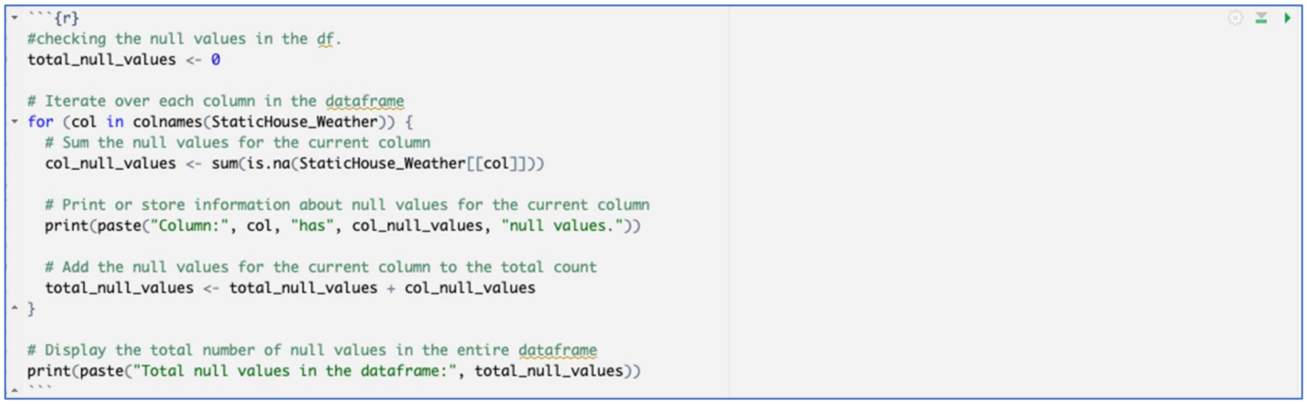


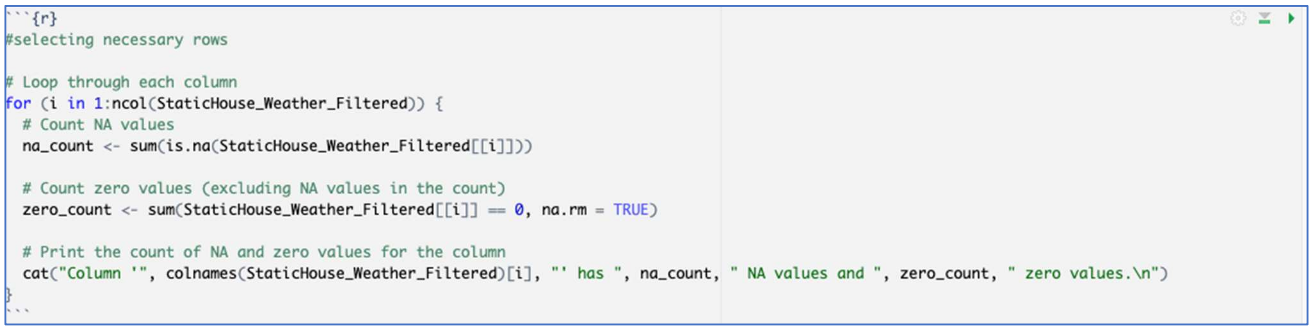
Choosing a higher cooling setpoint can significantly reduce your daily energy consumption. This graph shows how raising the setting from 60°F to 80°F cuts daily consumption in half, demonstrating the significant impact of this simple adjustment. By adjusting your cooling habits and embracing a slightly warmer temperature, you can contribute to a more sustainable energy future.

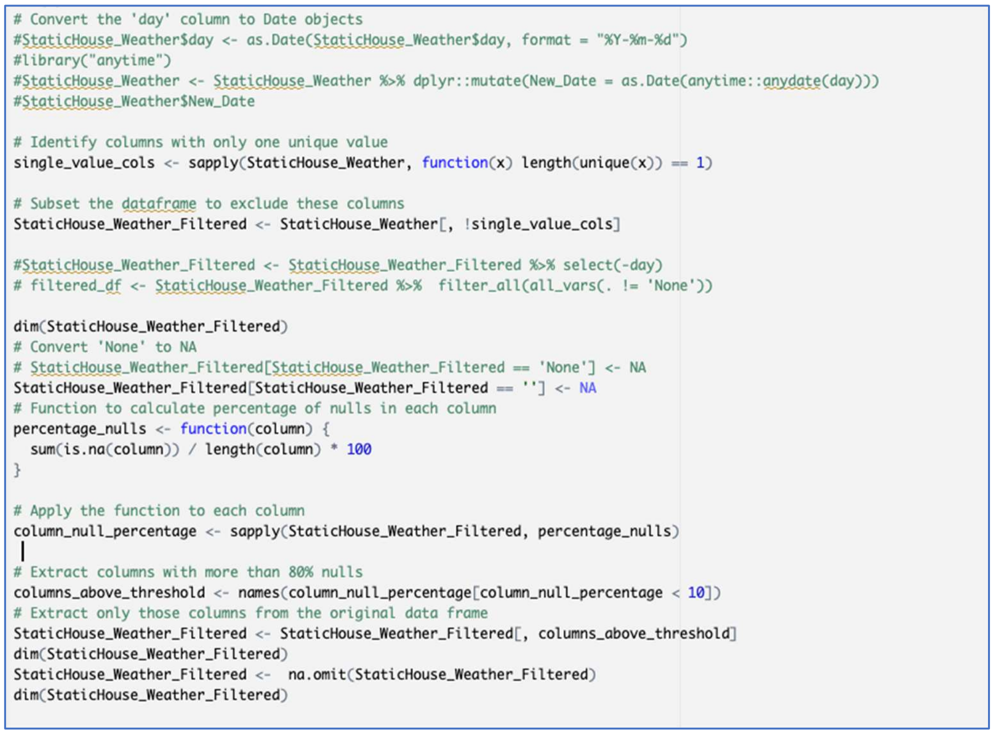
FEATURE ENGINEERING

Checked For the Null Values in The Data Frame:

We are doing data cleaning by removing unnecessary columns from a DataFrame. It starts by identifying columns that have only one unique value and then creates a new DataFrame, StaticHouse_Weather_Filtered, by excluding these columns. Next, it converts 'None' and empty strings to NA values in this filtered DataFrame. To further refine the data, a function is used to calculate the percentage of null values in each column, and columns with less than 10% null values are retained. The code then extracts only the selected columns from the original DataFrame, StaticHouse_Weather, resulting in a cleaned and more streamlined dataset. Finally, any rows with NA values are removed from the filtered DataFrame. This process helps improve data quality by eliminating redundant or uninformative columns and handling missing data, making the data more suitable for analysis or further processing.



~~~r
```{r}
#checking the null values in the df.
total_null_values <- 0

# Iterate over each column in the dataframe
for (col in colnames(StaticHouse_Weather)) {
  # Sum the null values for the current column
  col_null_values <- sum(is.na(StaticHouse_Weather[[col]]))

  # Print or store information about null values for the current column
  print(paste("Column:", col, "has", col_null_values, "null values."))

  # Add the null values for the current column to the total count
  total_null_values <- total_null_values + col_null_values
}

# Display the total number of null values in the entire dataframe
print(paste("Total null values in the dataframe:", total_null_values))
~~~
~~~



```r
```{r}
#selecting necessary rows

# Loop through each column
for (i in 1:ncol(StaticHouse_Weather_Filtered)) {
  # Count NA values
  na_count <- sum(is.na(StaticHouse_Weather_Filtered[[i]]))

  # Count zero values (excluding NA values in the count)
  zero_count <- sum(StaticHouse_Weather_Filtered[[i]] == 0, na.rm = TRUE)

  # Print the count of NA and zero values for the column
  cat("Column '", colnames(StaticHouse_Weather_Filtered)[i], "' has ", na_count, " NA values and ", zero_count, " zero values.\n")
}
~~~
```
```



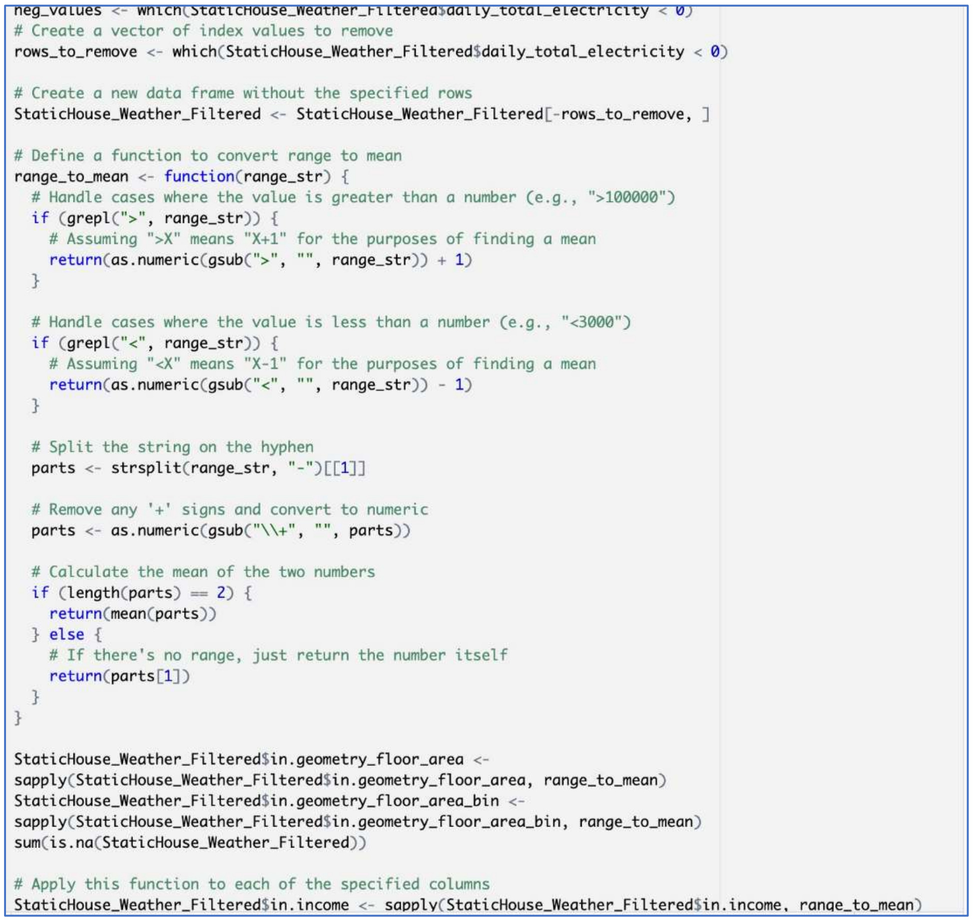
```
# Convert the 'day' column to Date objects
#StaticHouse_Weather$day <- as.Date(StaticHouse_Weather$day, format = "%Y-%m-%d")
#library("anytime")
#StaticHouse_Weather <- StaticHouse_Weather %>% dplyr::mutate(New_Date = as.Date(anytime::anydate(day)))
#StaticHouse_Weather$New_Date

# Identify columns with only one unique value
single_value_cols <- sapply(StaticHouse_Weather, function(x) length(unique(x)) == 1)

# Subset the dataframe to exclude these columns
StaticHouse_Weather_Filtered <- StaticHouse_Weather[, !single_value_cols]

#StaticHouse_Weather_Filtered <- StaticHouse_Weather_Filtered %>% select(-day)
# filtered_df <- StaticHouse_Weather_Filtered %>%  filter_all(all_vars(. != 'None'))

dim(StaticHouse_Weather_Filtered)
# Convert 'None' to NA
# StaticHouse_Weather_Filtered[StaticHouse_Weather_Filtered == 'None'] <- NA
StaticHouse_Weather_Filtered[StaticHouse_Weather_Filtered == ''] <- NA
# Function to calculate percentage of nulls in each column
percentage_nulls <- function(column) {
  sum(is.na(column)) / length(column) * 100
}

# Apply the function to each column
column_null_percentage <- sapply(StaticHouse_Weather_Filtered, percentage_nulls)

# Extract columns with more than 80% nulls
columns_above_threshold <- names(column_null_percentage[column_null_percentage < 10])
# Extract only those columns from the original data frame
StaticHouse_Weather_Filtered <- StaticHouse_Weather_Filtered[, columns_above_threshold]
dim(StaticHouse_Weather_Filtered)
StaticHouse_Weather_Filtered <-  na.omit(StaticHouse_Weather_Filtered)
dim(StaticHouse_Weather_Filtered)
```

Then we furthered cleaned the data by the following steps:

- It identifies rows where the 'daily_total_electricity' column has negative values (neg_values).
- A vector of index values to remove these rows is created (rows_to_remove).
- It creates a new data frame, StaticHouse_Weather_Filtered, without the specified rows, effectively removing rows with negative electricity values.
- A custom function, range_to_mean, is defined to handle data transformations. This function calculates the mean value from a range or extracts numeric values from strings that contain symbols like ">" or "<".
- The range_to_mean function is applied to columns 'in.geometry_floor_area', 'in.geometry_floor_area_bin', 'in.income', 'in.income_recs_2015', and 'in.income_recs_2020'. This operation ensures that these columns contain meaningful numeric values and not ranges or symbols.
- Columns like 'in.bathroom_spot_vent_hour', 'in.range_spot_vent_hour', 'in.cooling_setpoint', 'in.cooling_setpoint_offset_magnitude', 'in.heating_setpoint', 'in.heating_setpoint_offset_magnitude', 'in.infiltration', and 'in.occupants' undergo

transformations to convert them into numeric values or remove unwanted characters like 'Hour', 'F', 'ACH50', and '+'.

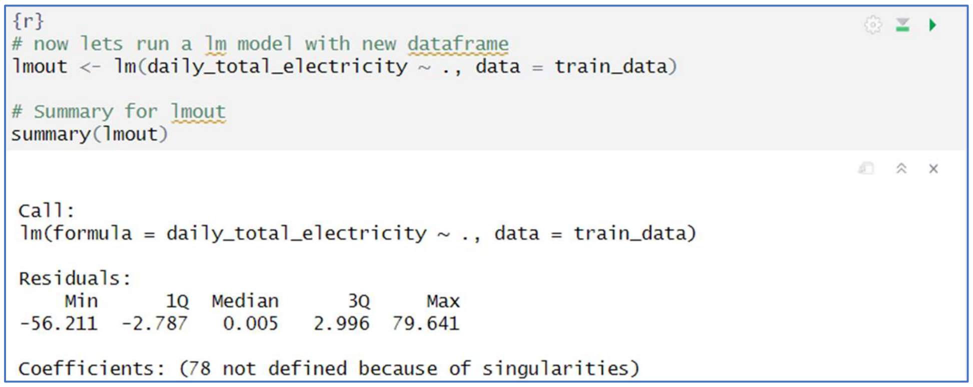
- The code checks for the presence of missing values (NA) in the StaticHouse_Weather_Filtered DataFrame using sum(is.na(StaticHouse_Weather_Filtered)).



```r
neg_values <- which(StaticHouse_Weather_Filtered$daily_total_electricity < 0)
# Create a vector of index values to remove
rows_to_remove <- which(StaticHouse_Weather_Filtered$daily_total_electricity < 0)

# Create a new data frame without the specified rows
StaticHouse_Weather_Filtered <- StaticHouse_Weather_Filtered[-rows_to_remove, ]

# Define a function to convert range to mean
range_to_mean <- function(range_str) {
  # Handle cases where the value is greater than a number (e.g., ">100000")
  if (grepl(">", range_str)) {
    # Assuming ">X" means "X+1" for the purposes of finding a mean
    return(as.numeric(gsub(">", "", range_str)) + 1)
  }

  # Handle cases where the value is less than a number (e.g., "<3000")
  if (grepl("<", range_str)) {
    # Assuming "<X" means "X-1" for the purposes of finding a mean
    return(as.numeric(gsub("<", "", range_str)) - 1)
  }

  # Split the string on the hyphen
  parts <- strsplit(range_str, "-")[[1]]

  # Remove any '+' signs and convert to numeric
  parts <- as.numeric(gsub("\\+", "", parts))

  # Calculate the mean of the two numbers
  if (length(parts) == 2) {
    return(mean(parts))
  } else {
    # If there's no range, just return the number itself
    return(parts[1])
  }
}

StaticHouse_Weather_Filtered$in.geometry_floor_area <-
sapply(StaticHouse_Weather_Filtered$in.geometry_floor_area, range_to_mean)
StaticHouse_Weather_Filtered$in.geometry_floor_area_bin <-
sapply(StaticHouse_Weather_Filtered$in.geometry_floor_area_bin, range_to_mean)
sum(is.na(StaticHouse_Weather_Filtered))

# Apply this function to each of the specified columns
StaticHouse_Weather_Filtered$in.income <- sapply(StaticHouse_Weather_Filtered$in.income, range_to_mean)
```

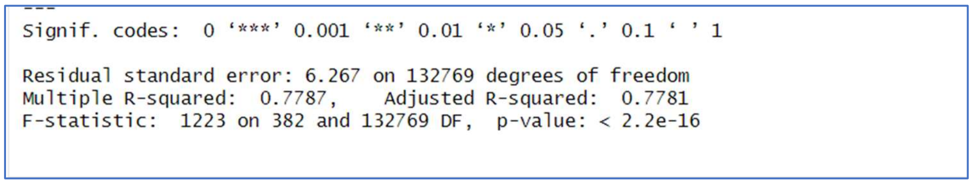
Model to predict the energy usage

We initiated the process of running machine learning models to predict the accuracy of total energy consumption. The objective is to develop a predictive model that can estimate energy consumption based on various factors, particularly temperature. By training our models on historical data, we aim to establish a relationship between temperature and energy consumption. Once the model is trained and validated, we can use it to make predictions, even with new temperature data that is +5 degrees different from the historical records. This predictive capability will enable us to anticipate energy consumption under varying temperature conditions, providing valuable insights for energy management and optimization.



```
{r}
# now lets run a lm model with new dataframe
lmout <- lm(daily_total_electricity ~ ., data = train_data)

# Summary for lmout
summary(lmout)
```

```
Call:
lm(formula = daily_total_electricity ~ ., data = train_data)

Residuals:
    Min      1Q  Median      3Q     Max 
-56.211  -2.787   0.005   2.996  79.641 

Coefficients: (78 not defined because of singularities)
```



```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 6.267 on 132769 degrees of freedom
Multiple R-squared:  0.7787,	Adjusted R-squared:  0.7781 
F-statistic:  1223 on 382 and 132769 DF,  p-value: < 2.2e-16
```

In this section, we will dive into the results of our linear regression model, which was developed to gain insights into the factors influencing daily electricity consumption in our residential complex. The following output summarizes the key findings of our analysis:

Model Summary

We initiated our analysis by fitting a linear regression model to the dataset using the `lm` function in R.

We aimed to predict daily electricity consumption (`daily_total_electricity`) based on a set of predictor variables represented by the `.` symbol. The "dot" notation indicates that we considered all available predictor variables for this analysis.

Residuals

The summary statistics of the residuals are crucial in assessing the goodness of fit of our model. These statistics provide insights into how well our model aligns with the actual data points. Here are the key statistics:

- **Residual Standard Error:**
  - The residual standard error is approximately 6.267. This represents the average amount that the observed values deviate from the fitted values. In other words, it's a measure of the spread of the residuals.
- **Multiple R-squared:**
  - The multiple R-squared value is 0.7787. This indicates the proportion of the variance in the dependent variable that is predictable from the independent variables. In this case, about 79.49% of the variability in the dependent variable is explained by the independent variables.
- **Adjusted R-squared:**
  - The adjusted R-squared value is 0.7781. It adjusts the R-squared value for the number of predictors in the model. It is useful when there are multiple predictors. A higher adjusted R-squared suggests a better fit of the model.
- **F-statistic:**
  - The F-statistic is 1223 with 382 and 132769 degrees of freedom. This statistic tests the overall significance of the model. The low p-value (< 2.2e-16) indicates that at least one of the predictors is related to the response variable.

Coefficients

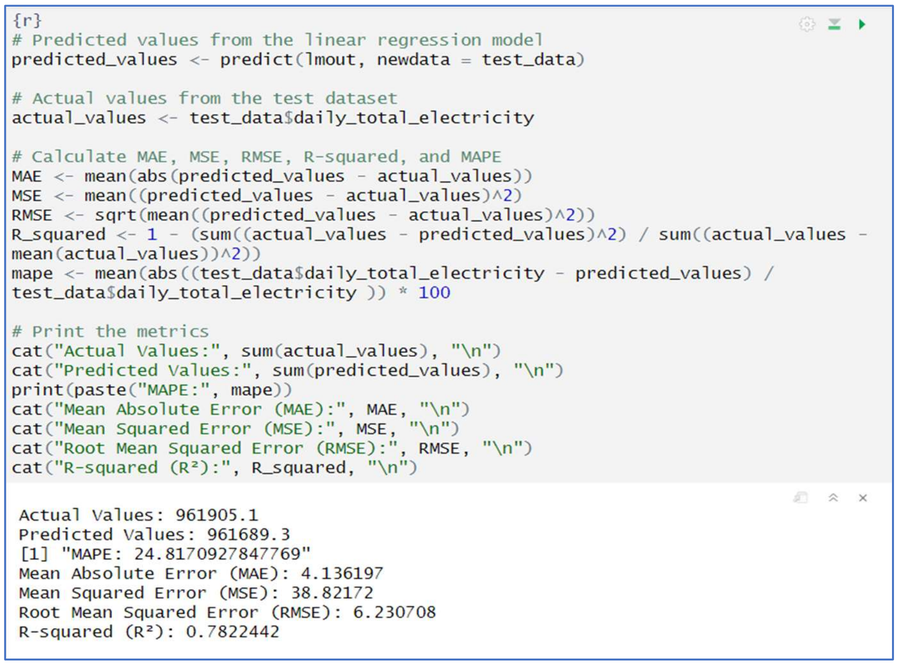
The coefficients estimated by our linear regression model provide insights into the relationships between the predictor variables and daily electricity consumption. Each coefficient corresponds to a predictor variable, and the following information is presented for each:

In practical terms, the coefficients offer insights into the impact of each predictor variable on daily electricity consumption. For instance, if we have a coefficient of -0.04597 for the "day" variable, it implies that, on average, daily electricity consumption decreases by approximately 0.046 units for each additional day.

Singularities

Our model revealed some issues related to singularities, denoted by the message "(78 not defined because of singularities)." This indicates the presence of multicollinearity or linear dependencies among predictor variables. It's essential to acknowledge this issue as it may affect the stability of our model.

Model Fit



```
{r}
# Predicted values from the linear regression model
predicted_values <- predict(lmout, newdata = test_data)

# Actual values from the test dataset
actual_values <- test_data$daily_total_electricity

# Calculate MAE, MSE, RMSE, R-squared, and MAPE
MAE <- mean(abs(predicted_values - actual_values))
MSE <- mean((predicted_values - actual_values)^2)
RMSE <- sqrt(mean((predicted_values - actual_values)^2))
R_squared <- 1 - (sum((actual_values - predicted_values)^2) / sum((actual_values -
mean(actual_values))^2))
mape <- mean(abs((test_data$daily_total_electricity - predicted_values) /
test_data$daily_total_electricity )) * 100

# Print the metrics
cat("Actual Values:", sum(actual_values), "\n")
cat("Predicted Values:", sum(predicted_values), "\n")
print(paste("MAPE:", mape))
cat("Mean Absolute Error (MAE):", MAE, "\n")
cat("Mean Squared Error (MSE):", MSE, "\n")
cat("Root Mean Squared Error (RMSE):", RMSE, "\n")
cat("R-squared (R²):", R_squared, "\n")
```

```
Actual Values: 961905.1
Predicted Values: 961689.3
[1] "MAPE: 24.8170927847769"
Mean Absolute Error (MAE): 4.136197
Mean Squared Error (MSE): 38.82172
Root Mean Squared Error (RMSE): 6.230708
R-squared (R²): 0.7822442
```

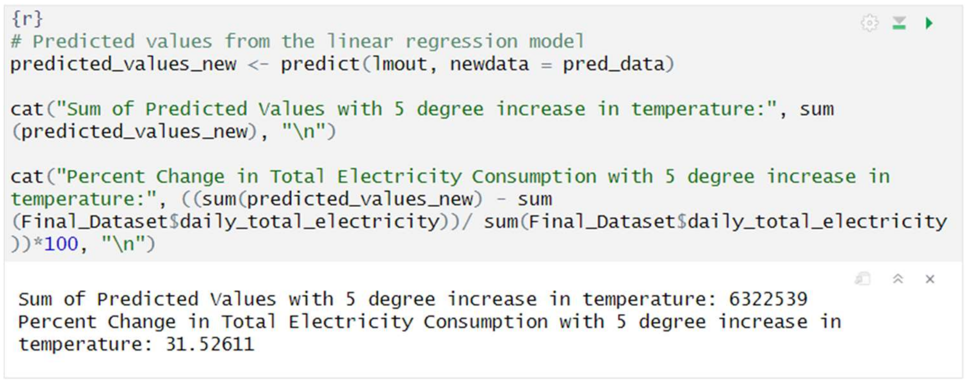
To assess the overall performance of our model, we considered the following metrics:

R-squared: This value, which stands at 0.7822, represents the proportion of variance in daily electricity consumption that our model explains. A higher R-squared value indicates a better fit.

Adjusted R-squared: This value, adjusted for the number of predictor variables, is a more reliable measure when comparing models with different complexity levels. Our model achieved an adjusted R-squared of 0.7781.

Overall, The model is statistically significant, with a high R-squared value suggesting a good fit to the data.

Now Let's predict the total energy consumed with temperature increased by 5 degrees.



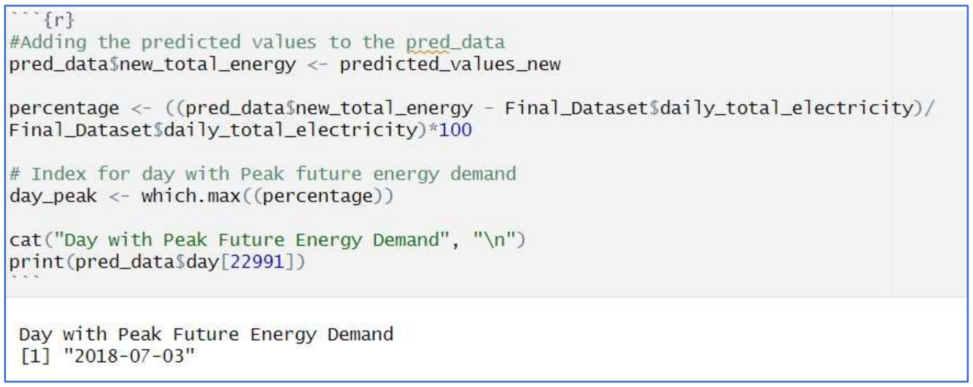
```
{r}
# Predicted values from the linear regression model
predicted_values_new <- predict(lmout, newdata = pred_data)

cat("Sum of Predicted Values with 5 degree increase in temperature:", sum
(predicted_values_new), "\n")

cat("Percent Change in Total Electricity Consumption with 5 degree increase in
temperature:", ((sum(predicted_values_new) - sum
(Final_Dataset$daily_total_electricity))/ sum(Final_Dataset$daily_total_electricity
))*100, "\n")
```

```
Sum of Predicted Values with 5 degree increase in temperature: 6322539
Percent Change in Total Electricity Consumption with 5 degree increase in
temperature: 31.52611
```

The 31.53% increase in total electricity consumption is a key metric. It suggests a positive correlation between temperature and electricity usage according to the model. This information can be valuable for planning and resource allocation in scenarios where temperature fluctuations impact electricity demand.

Future Peak Energy Demand in Total (By Day):

The day with peak energy demand predicted is July 3rd. This indicates that the first week of July will usually be the week with the highest energy demand.

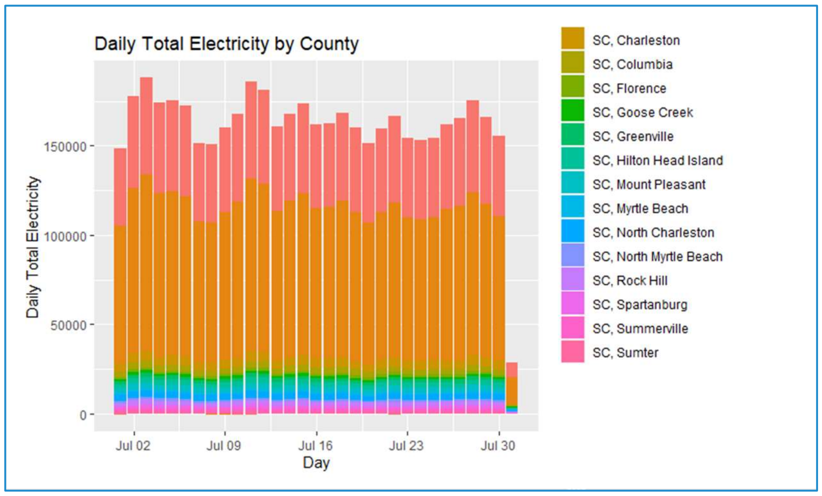


~~~
```{r}
#Adding the predicted values to the pred_data
pred_data$new_total_energy <- predicted_values_new

percentage <- ((pred_data$new_total_energy - Final_Dataset$daily_total_electricity)/
Final_Dataset$daily_total_electricity)*100

# Index for day with Peak future energy demand
day_peak <- which.max((percentage))

cat("Day with Peak Future Energy Demand", "\n")
print(pred_data$day[22991])
```
~~~

```
 Day with Peak Future Energy Demand
 [1] "2018-07-03"
```

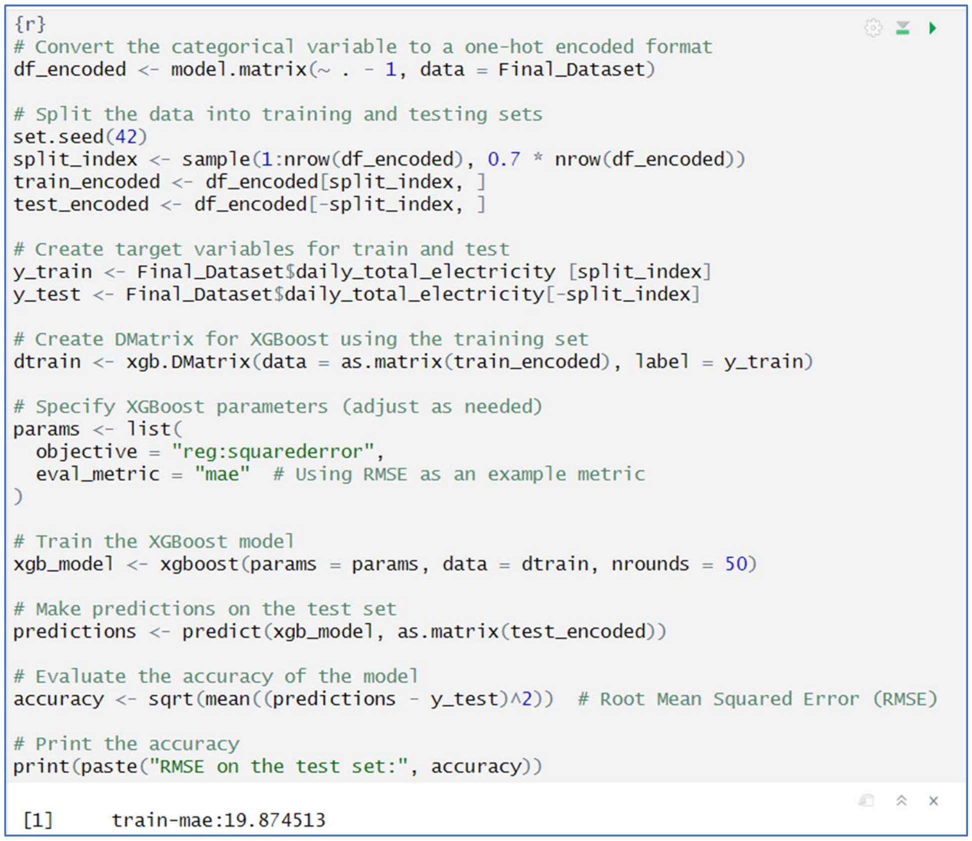
For Different Geographic Regions:

Highest electricity consumption is predicted to be in the counties of Charleston and Sumter, while it is lowest in the counties of Florence and Rock Hill



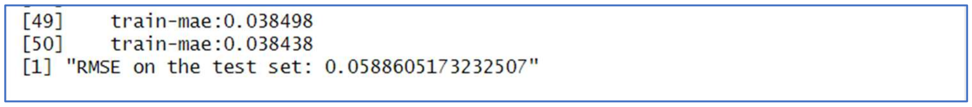
Models we didn't go ahead with:

XGBOOST



```
{r}
# Convert the categorical variable to a one-hot encoded format
df_encoded <- model.matrix(~ . - 1, data = Final_Dataset)

# Split the data into training and testing sets
set.seed(42)
split_index <- sample(1:nrow(df_encoded), 0.7 * nrow(df_encoded))
train_encoded <- df_encoded[split_index, ]
test_encoded <- df_encoded[-split_index, ]

# Create target variables for train and test
y_train <- Final_Dataset$daily_total_electricity [split_index]
y_test <- Final_Dataset$daily_total_electricity[-split_index]

# Create DMatrix for XGBoost using the training set
dtrain <- xgb.DMatrix(data = as.matrix(train_encoded), label = y_train)

# Specify XGBoost parameters (adjust as needed)
params <- list(
  objective = "reg:squarederror",
  eval_metric = "mae"  # Using RMSE as an example metric
)

# Train the XGBoost model
xgb_model <- xgboost(params = params, data = dtrain, nrounds = 50)

# Make predictions on the test set
predictions <- predict(xgb_model, as.matrix(test_encoded))

# Evaluate the accuracy of the model
accuracy <- sqrt(mean((predictions - y_test)^2))  # Root Mean Squared Error (RMSE)

# Print the accuracy
print(paste("RMSE on the test set:", accuracy))
```

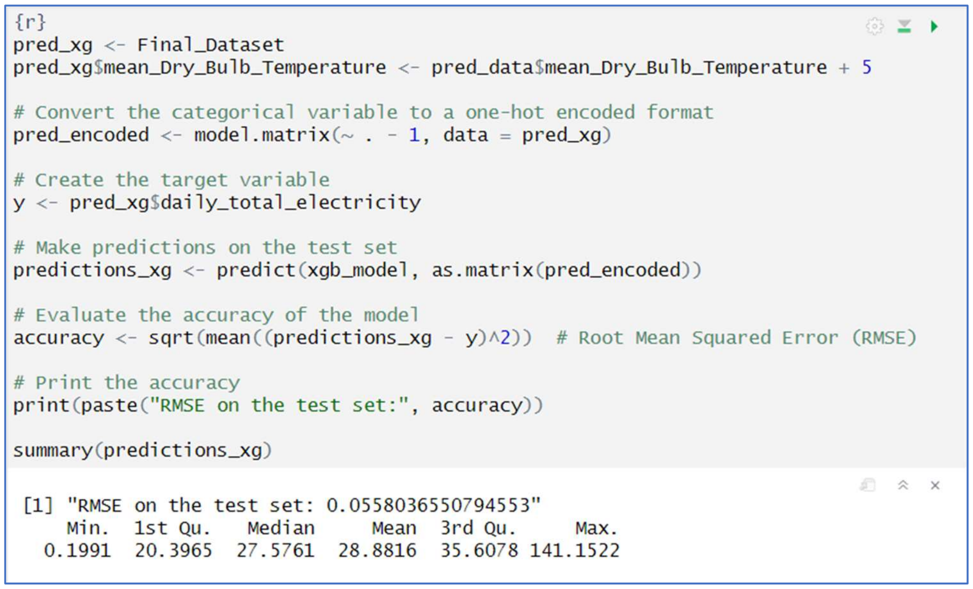
```
[1]     train-mae:19.874513
```



```
[49]    train-mae:0.038498
[50]    train-mae:0.038438
[1] "RMSE on the test set: 0.0588605173232507"
```

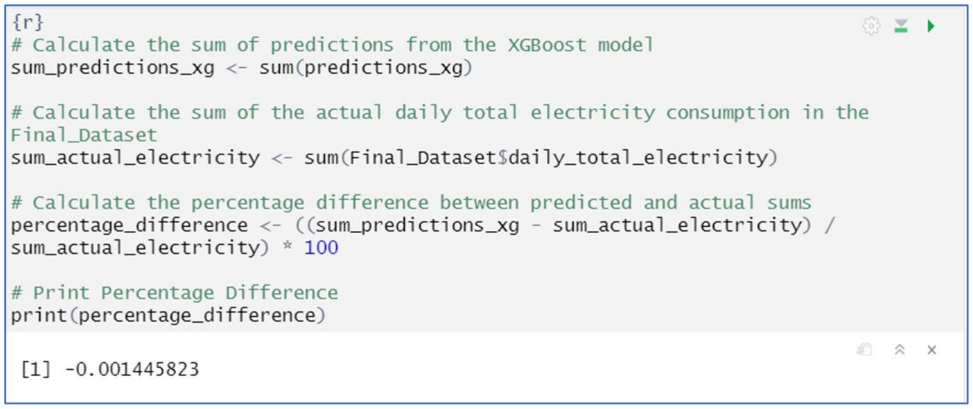
On the XGBOOST test set, we obtained an RMSE of 0.05886. A lower RMSE indicates better model performance because it indicates that the model's predictions are more accurate.

Now Let's predict the total energy consumed with temperature increased by 5 degrees



```r
{r}
pred_xg <- Final_Dataset
pred_xg$mean_Dry_Bulb_Temperature <- pred_data$mean_Dry_Bulb_Temperature + 5

# Convert the categorical variable to a one-hot encoded format
pred_encoded <- model.matrix(~ . - 1, data = pred_xg)

# Create the target variable
y <- pred_xg$daily_total_electricity

# Make predictions on the test set
predictions_xg <- predict(xgb_model, as.matrix(pred_encoded))

# Evaluate the accuracy of the model
accuracy <- sqrt(mean((predictions_xg - y)^2))  # Root Mean Squared Error (RMSE)

# Print the accuracy
print(paste("RMSE on the test set:", accuracy))

summary(predictions_xg)
```

```
[1] "RMSE on the test set: 0.0558036550794553"
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
  0.1991  20.3965  27.5761  28.8816  35.6078 141.1522
```

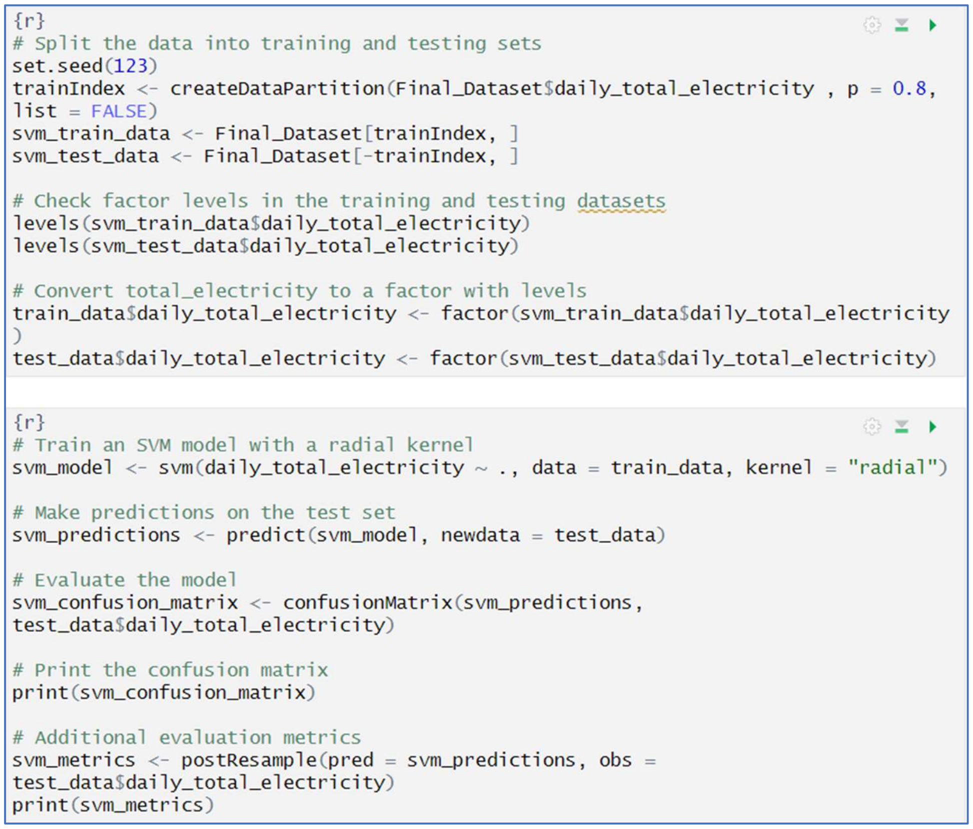


```r
{r}
# Calculate the sum of predictions from the XGBoost model
sum_predictions_xg <- sum(predictions_xg)

# Calculate the sum of the actual daily total electricity consumption in the
Final_Dataset
sum_actual_electricity <- sum(Final_Dataset$daily_total_electricity)

# Calculate the percentage difference between predicted and actual sums
percentage_difference <- ((sum_predictions_xg - sum_actual_electricity) /
sum_actual_electricity) * 100

# Print Percentage Difference
print(percentage_difference)
```

```
[1] -0.001445823
```

As we can see, we obtained a percentage difference of -0.0014, which is far from the expected predictions, so we decided to discard this model.

SVM



```
{r}
# Split the data into training and testing sets
set.seed(123)
trainIndex <- createDataPartition(Final_Dataset$daily_total_electricity , p = 0.8,
list = FALSE)
svm_train_data <- Final_Dataset[trainIndex, ]
svm_test_data <- Final_Dataset[-trainIndex, ]

# Check factor levels in the training and testing datasets
levels(svm_train_data$daily_total_electricity)
levels(svm_test_data$daily_total_electricity)

# Convert total_electricity to a factor with levels
train_data$daily_total_electricity <- factor(svm_train_data$daily_total_electricity
)
test_data$daily_total_electricity <- factor(svm_test_data$daily_total_electricity)
```

```
{r}
# Train an SVM model with a radial kernel
svm_model <- svm(daily_total_electricity ~ ., data = train_data, kernel = "radial")

# Make predictions on the test set
svm_predictions <- predict(svm_model, newdata = test_data)

# Evaluate the model
svm_confusion_matrix <- confusionMatrix(svm_predictions,
test_data$daily_total_electricity)

# Print the confusion matrix
print(svm_confusion_matrix)

# Additional evaluation metrics
svm_metrics <- postResample(pred = svm_predictions, obs =
test_data$daily_total_electricity)
print(svm_metrics)
```

We attempted to use SVM to predict the peak energy consumption for July of the following year, but the size of the dataset hindered us from running the code due to memory allocation issues, and sampling the data produced predictions that were far from the expected values. Thus, we chose to discontinue this model.

RECOMMENDATIONS

Demand-side management (DSM) strategies:

- **Time-of-use (TOU) pricing:** Implement dynamic pricing schemes where electricity costs are higher during peak periods and lower during off-peak hours. This incentivizes consumers to shift their energy usage to less demanding times.
- **Peak demand reduction programs:** Offer financial incentives or rewards to customers who reduce their energy consumption during peak periods. This could include rebates on energy-efficient appliances or bill credits for exceeding reduction targets.
- **Direct load control (DLC) programs**: Implement technology that allows eSC to remotely manage and adjust certain appliances, like air conditioners, during peak periods to reduce overall demand.

Technology-driven solutions:

- **Smart meters and home energy management systems (HEMS):** Implement smart meters and HEMS to provide customers with real-time energy usage data and insights. This empowers them to make informed decisions about their energy consumption and take action to reduce it.
- **Machine learning and AI-based demand forecasting:** Develop algorithms and models to predict future energy demand with greater accuracy. This information can be used to optimize resource allocation and inform grid management strategies.
- **Dynamic pricing and demand response platforms:** Utilize technology platforms to implement dynamic pricing schemes and manage demand response programs effectively.

Policy and regulatory mechanisms:

- **Building codes and energy efficiency standards:** Advocate for stricter building codes that require new homes to be built with energy-efficient features and appliances.

- **Renewable energy mandates and incentives:** Support policies that encourage the development and adoption of renewable energy sources, such as solar and wind power.
- **Carbon pricing and carbon reduction programs:** Implement mechanisms to put a price on carbon emissions, incentivizing businesses and individuals to reduce their carbon footprint and energy consumption.

Community engagement and social initiatives:

- **Public awareness campaigns:** Partner with community organizations and local governments to raise awareness about peak energy demand and the importance of energy conservation.
- **Energy efficiency workshops and events:** Organize workshops and events to educate residents about energy-saving tips and technologies.
- **Community solar projects:** Develop community-owned solar projects to benefit local communities and generate renewable energy.

Reducing Energy Consumption through Cooling Setpoint Adjustment

This information highlights a valuable opportunity to reduce electricity consumption and save money for eSC's customers. Here's a breakdown of the key points and potential actions:

Key Points:

- Air conditioners are the major energy consumers, accounting for over 60% of daily household electricity usage.
- Raising the cooling setpoint can significantly reduce electricity consumption. For example, increasing it from 72°F to 75°F can lead to a 20% reduction.
- This suggests a substantial potential for energy savings and cost reductions.

Potential Actions:

- Educate customers: Launch campaigns to inform customers about the significant energy savings achievable by raising their cooling setpoint. Provide clear and concise information about the potential benefits and practical tips for adjusting their thermostats.

- Incentivize setpoint adjustments: Implement programs that reward customers for raising their cooling setpoints during peak demand periods. This could include rebates on energy bills, participation in raffles or contests, or gamified challenges.
- Offer smart thermostats: Partner with thermostat manufacturers to offer discounted smart thermostats to customers. These devices can learn individual preferences, automatically adjust setpoints during peak hours, and track energy usage, further encouraging conservation.
- Develop dynamic pricing models: Consider implementing time-of-use pricing, where electricity costs vary depending on the time of day. Higher prices during peak hours incentivize customers to adjust their thermostats and reduce energy consumption.
- Promote energy-efficient alternatives: Encourage customers to consider alternative cooling methods like ceiling fans or natural ventilation during moderate weather. These energy-efficient options can further reduce their reliance on air conditioning.

Additional Considerations:

- **Analyze customer demographics:** Tailor outreach and educational programs to consider the specific needs and preferences of different customer segments.
- **Monitor the impact:** Continuously assess the effectiveness of implemented programs and measure the actual energy savings achieved.
- **Collaborate with stakeholders:** Partner with local governments and community organizations to amplify educational initiatives and promote energy conservation efforts.

To encourage customers to raise their cooling setpoints, eSC can offer a variety of incentives, appealing to different motivations and preferences. Here are some potential options:

Financial Incentives:

- **Bill credits:** Offer direct credits on electricity bills proportional to the amount of energy saved by raising the setpoint. This provides a tangible financial benefit and incentivizes sustained behavior change.

- **Rebates:** Provide rebates for purchasing and installing smart thermostats that automatically adjust setpoints during peak hours. This encourages long-term adoption of energy-efficient technology.
- **Discounts on energy-efficient appliances:** Offer discounts on energy-efficient air conditioners or other appliances, further reducing customers' energy consumption and long-term costs.

Rewards and Recognition:

- **Loyalty points:** Implement a loyalty program where customers earn points for raising their setpoint during peak hours. These points can be redeemed for rewards like bill discounts, merchandise, or participation in special events.
- **Gamified challenges:** Utilize gamification elements to create interactive challenges where customers compete for individual, or team rewards based on their energy savings achieved through setpoint adjustments.
- **Public recognition:** Feature top performers on eSC's website or social media platforms, acknowledging their contribution to energy conservation and inspiring others to participate.

Convenience and Comfort:

- **Free installation of smart thermostats:** Offer free installation of smart thermostats for interested customers, removing any technical barriers to adopting this energy-saving technology.
- **Remote access and control:** Provide customers with mobile apps or web interfaces to remotely access and adjust their thermostats, offering convenience and flexibility.
- **Personalized recommendations:** Develop personalized recommendations for setpoint adjustments based on individual preferences, weather conditions, and energy consumption patterns.

CONCLUSION

In summary, our project aims to assist eSC, an energy company, in addressing the challenges of increased electricity demand during peak periods, especially in the summer. Through data-driven analysis and modeling, we identified key factors influencing energy consumption, developed strategies for optimization, and proposed recommendations for sustainable energy management. Our findings emphasize the importance of adjusting cooling setpoints as a practical way to reduce electricity usage. The linear regression model showcased strong correlations between temperature and energy consumption, providing valuable insights for future demand predictions. Our recommendations span demand-side management, technology solutions, policy advocacy, and community engagement, offering a holistic approach to energy sustainability. Overall, our project equips eSC with actionable strategies and tools to navigate the complex landscape of energy demand, aligning with their commitment to environmental responsibility and customer satisfaction.

Shiny App URL:

https://klikhith.shinyapps.io/final_shiny/