IST 687 final project report

Francisco Franco Arenas

2023-11-27

1. introduction

2. Data reading and merging approach

In the data preparation phase, we focus on consolidating and preparing three key datasets for our analysis of household energy consumption. These datasets include house static data, electricity consumption data, and weather data.

House Static Data Loading

We begin by loading the static information about the houses. This dataset, obtained in Parquet format, contains essential identifiers such as building IDs (`bldg_id`) and county information. These identifiers are crucial as they link the static house data with the dynamic electricity consumption and weather data.

```
url <- "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/static_house_info.parquet"
house_static_data <- read_parquet(url)

# store the ids to load the energy consumption data
ids <- house_static_data %>%
    pull(bldg_id)

# store the counties to load the weather data
counties <- house_static_data %>%
    pull(in.county) %>%
    unique()
```

Electricity Consumption Pipeline

The electricity consumption data is processed in a pipeline designed to handle data for each building individually. This involves iterating over each building ID, fetching electricity consumption data from corresponding Parquet files. The pipeline includes error handling to manage potential issues in data loading. For each building, we calculate the total electricity consumption and retain essential timestamps. This structured approach ensures efficient handling and processing of large-scale data, preparing it for further analytical steps.

```
# Initialize a counter for tracking progress in the loop
ii <- 1
```

```r
# Initialize an empty tibble for storing electricity consumption data
electricity_consumption <- tibble(total_electricity = numeric(), date = Date())

# Loop through each building ID to process electricity consumption data
for (id in ids) {
    # Attempt to read the electricity data for the current building ID
    response <- try(read_parquet(paste0("https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/2
        id, ".parquet")))

    # Print the counter to track progress
    print(ii)

    # Handle errors in data loading
    if (class(response)[1] == "try-error") {
        print(paste0("error loading house ", id))
    } else {
        # Process the electricity data
        response %>%
            mutate(total_electricity = rowSums(select(., contains("electricity")),
                na.rm = TRUE)) %>%
            select(total_electricity, time) %>%
            mutate(bldg_id = id) -> temp_data

        # Append the processed data to the main electricity consumption tibble
        electricity_consumption <- rbind(electricity_consumption, temp_data)

        # Clear the temporary data to free up memory
        rm(temp_data)

        # Increment the counter
        ii <- ii + 1
    }
}
```

Weather Data Pipeline

Parallel to processing electricity data, we also manage weather data, specifically focusing on temperature metrics across different counties. The weather data, sourced in CSV format, includes detailed temperature readings. We process this data by iterating over each county, aligning the weather data with our other datasets based on the county identifier. This step is crucial for analyzing the impact of weather on electricity consumption patterns.

```r
# Initialize an empty tibble for storing temperature data
temperature <- tibble(max_temp = numeric(), min_temp = numeric(), avg_temp = numeric(), date = Date())

# Loop through each county to process weather data
for (county in counties) {
  # Read the weather data CSV for the current county
  response <- read_csv(
    paste0(
      "https://intro-datascience.s3.us-east-2.amazonaws.com/SC-data/weather/2023-weather-data/",
      county,
      ".csv"))
```

```
  # Process the weather data
  response %>%
    select(date_time, contains("Temperature")) %>%
    rename(temperature = `Dry Bulb Temperature [°C]`) %>%
    filter(date_time >= as.Date("2018-07-01"), date_time <= as.Date("2018-07-31")) %>%
    # Additional processing can be done here such as summarizing daily temperatures
    mutate(county_id = county) -> temp_data

  # Append the processed data to the main temperature tibble
  temperature <- rbind(temperature, temp_data)

  # Clear the temporary data to free up memory
  rm(temp_data)
}

# Save the processed temperature data to a CSV file
write_csv(temperature, paste0("~/GitHub/household_energy_consumption/data/temperature.csv"))
```

Data Merging and Final Output

The final step in the data preparation phase is to merge these processed datasets. This involves aligning the electricity consumption data with the static house data and weather data, creating a comprehensive dataset that forms the backbone of our subsequent analysis. The output of this phase is a clean, well-structured dataset that encapsulates all necessary variables, ready for in-depth analysis.

3. Exploratory Analysis

In this phase, we commenced with a strategic preselection of features from the household static data. Our selection criteria were primarily guided by domain expertise, focusing on features directly related to energy consumption. The detailed list of these selected features can be found in Annex 1.

Upon examination, both the temperature data and the household static data were found to be complete with no missing entries, ensuring robustness in the subsequent analysis.

An initial observation from our exploratory analysis indicates a positive correlation between energy consumption and temperature. This trend is evident at both the individual house and county levels, suggesting a consistent pattern across different scales of data. The following visualization illustrates this correlation:

```
# CODE PENDING for plot
```

Feature importance

To assess the importance of each house feature in energy consumption, we merged the static and dynamic data, selecting a sample of 5% of the rows to run a random forest model. This approach helped us determine the importance of each feature in explaining electricity consumption. The most significant features identified are hour, the floor area and the vacancy status. For further analysis, we only selected features with an importance higher than 0.95%.

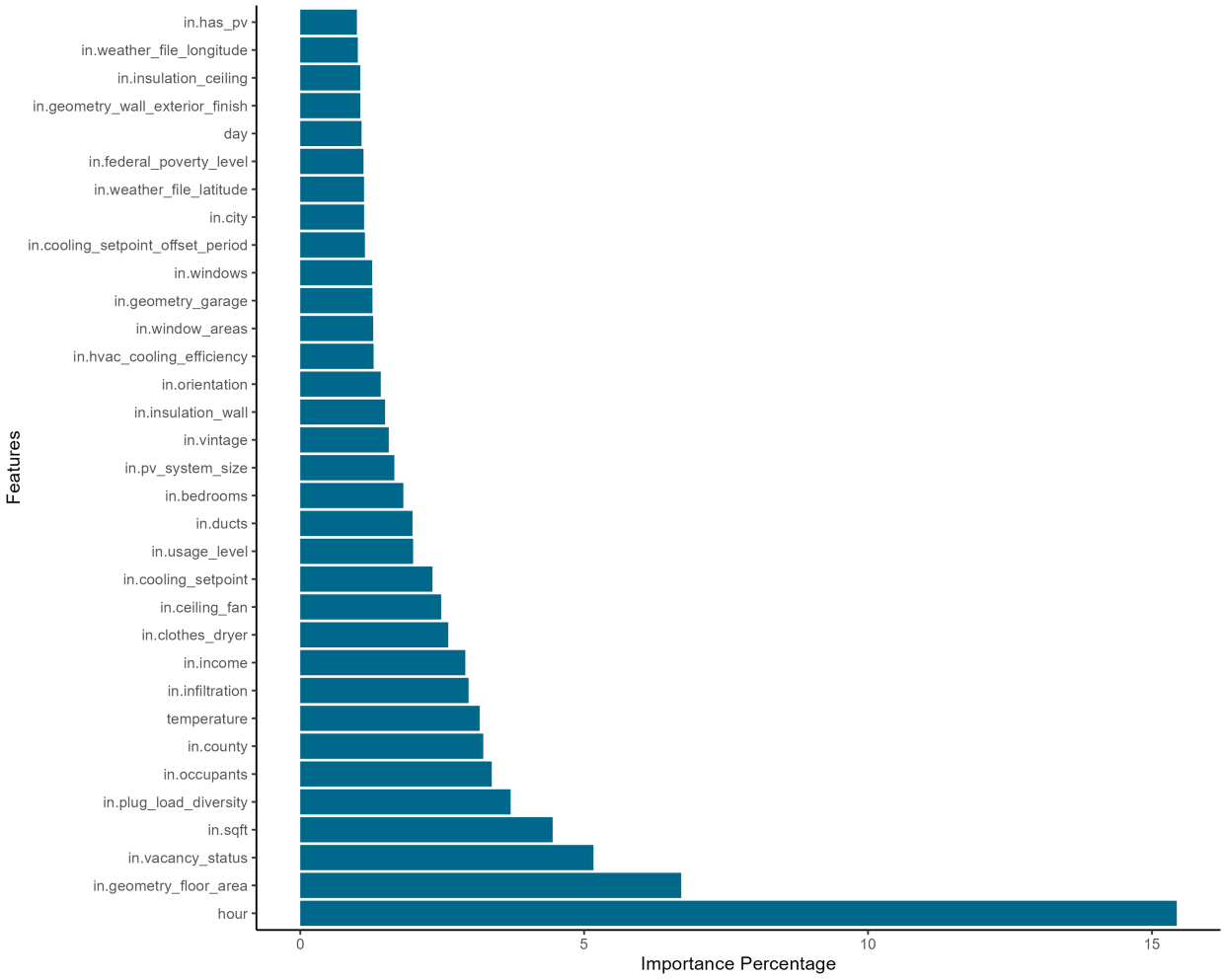

Figure 1: Your Plot Description

Feature engineering

Once the most important features were identified, we proceeded to refine the static house data. This process involved converting some categorical variables into numeric ones, such as insulation quality, HVAC efficiency, and window area. Other features were divided into multiple categories, especially those with composite information in their string values, like the "windows" feature. In some cases, the number of possible values for a factor was reduced, and ordinal factors, such as "usage_level," were converted to numeric. This extensive feature engineering required the use of regular expressions to extract information from strings.

```
house_data %>%
  select(-in.geometry_floor_area) %>%  # in sqft
  mutate(
    clothes_drier_type = str_extract(in.clothes_dryer, "^[^,]*"), #type of clothes dryer
    clothes_dryer_usage = str_extract(in.clothes_dryer, "(?<=\\s).+?(?=%)"), #usage of clothes dryer
    clothes_dryer_usage= as.numeric(clothes_dryer_usage), #to numeric
    clothes_dryer_usage = replace_na(clothes_dryer_usage,0)
    ) %>% #nas are 0
  select(-in.clothes_dryer) %>%  #drop in.clothes drier
  mutate(infiltration_ACH50 = as.numeric(str_extract(in.infiltration, "^[^ ]*"))) %>% #is a number in r
  select(-in.infiltration) %>%
  left_join(income_dictionary,by="in.income") %>%
  select(-in.income) %>%
  mutate(
    in.occupants=as.numeric(in.occupants),
    ) %>%
  mutate(
    in.plug_load_diversity=as.numeric(str_extract(in.plug_load_diversity, "^[^%]*")),
    duct_leakage=as.numeric(str_extract(in.ducts, "^[^%]*")),
    duct_leakage=replace_na(duct_leakage,0),
    duct_insulation_quality = as.numeric(str_extract(in.ducts, "(?<=-).*")),
    duct_insulation_quality=replace_na(duct_insulation_quality,0),
  ) %>%
  select(-in.ducts) %>%
  mutate(
    wall_material = str_extract(in.insulation_wall, "^[^,]+"),
    wall_insulation_quality = replace_na(as.numeric(str_extract(in.insulation_wall, "(?<=R-)\\d+")), 0)
  ) %>%
  select(-in.insulation_wall) %>%
  mutate(
    in.vintage=as.numeric(
        case_when(
          in.vintage=="<1940" ~ "1940",
          T ~ substr(in.vintage,1,4))
        )
  ) %>%
  separate(in.windows, into = c("winPane", "winGlazing", "winFrame", "winFill", "winAdditional"), sep =
  mutate(
    winPane = replace_na(winPane, "None"),
    winGlazing = replace_na(winGlazing, "None"),
    winFrame = replace_na(winFrame, "None"),
    winFill = replace_na(winFill, "None"),
    winAdditional = replace_na(winAdditional, "None")
  ) %>%
  mutate(
```

```
  winPane = factor(winPane),
  winGlazing = factor(winGlazing),
  winFrame = factor(winFrame),
  winFill = factor(winFill),
  winAdditional = factor(winAdditional)
) %>%
mutate(in.cooling_setpoint=as.numeric(substr(in.cooling_setpoint,1,2))) %>%
mutate(
  HVAC_type = case_when(
    str_detect(in.hvac_cooling_efficiency, "AC,") ~ "AC",
    str_detect(in.hvac_cooling_efficiency, "Heat Pump") ~ "Heat Pump",
    str_detect(in.hvac_cooling_efficiency, "Room AC,") ~ "Room AC",
    TRUE ~ "None"  # Default case if none of the above matches
  ),
  HVAC_efficiency = as.numeric(str_extract(in.hvac_cooling_efficiency, "\\d+\\.?\\d*"))  # Extracts n
) %>%
# Replace NAs with zeros in efficiency if no number is found
mutate(
  HVAC_efficiency = replace_na(HVAC_efficiency, 0)
) %>%
select(-in.hvac_cooling_efficiency) %>%
mutate(
  in.geometry_garage = as.numeric(substr(in.geometry_garage, 1, 1)),
  in.geometry_garage = replace_na(in.geometry_garage, 0)
  ) %>%
select(-in.city) %>% #most are "other"
mutate(
  in.window_areas = 4*as.numeric(str_extract(in.window_areas, "\\d+"))
) %>%
select(-in.federal_poverty_level) %>%  #colinear with income
mutate(
  in.insulation_ceiling = case_when(
    in.insulation_ceiling == "Uninsulated" ~ 0,  # Assuming uninsulated means R-0
    in.insulation_ceiling == "None" ~ 0,         # Assuming none means R-0
    TRUE ~ as.numeric(str_extract(in.insulation_ceiling, "\\d+"))
  )
) %>%
  mutate(
  in.pv_system_size = case_when(
    in.pv_system_size == "None" ~ 0,  # Assign 0 to 'None'
    TRUE ~ as.numeric(str_extract(in.pv_system_size, "\\d+\\.?\\d*"))  # Extract numeric kW value
  )
) %>%
select(-in.has_pv) %>%  # in pv_system_size
mutate(
  cooling_Day_Adjustment = as.integer(grepl("Day", in.cooling_setpoint_offset_period)),
  cooling_Night_Adjustment = as.integer(grepl("Night", in.cooling_setpoint_offset_period)),
  cooling_Hour_Offset = as.numeric(str_extract(in.cooling_setpoint_offset_period, "[-+]?\\d+"))
) %>%
# Replace NAs with zeros if no offset is specified
mutate(
  cooling_Hour_Offset = replace_na(cooling_Hour_Offset, 0)
) %>%
```

```
  select(-in.cooling_setpoint_offset_period) %>%
  mutate(
    wall_material = case_when(
      str_detect(in.geometry_wall_exterior_finish, "Wood") ~ "Wood",
      str_detect(in.geometry_wall_exterior_finish, "Aluminum") ~ "Aluminum",
      str_detect(in.geometry_wall_exterior_finish, "Vinyl") ~ "Vinyl",
      str_detect(in.geometry_wall_exterior_finish, "Brick") ~ "Brick",
      str_detect(in.geometry_wall_exterior_finish, "Stucco") ~ "Stucco",
      str_detect(in.geometry_wall_exterior_finish, "Fiber-Cement") ~ "Fiber-Cement",
      str_detect(in.geometry_wall_exterior_finish, "Shingle") ~ "Shingle",
      in.geometry_wall_exterior_finish == "None" ~ "None",
      TRUE ~ "Other"
    ),
    wall_color_lightness = case_when(
      str_detect(in.geometry_wall_exterior_finish, "Light") ~ "Light",
      str_detect(in.geometry_wall_exterior_finish, "Medium") ~ "Medium",
      str_detect(in.geometry_wall_exterior_finish, "Dark") ~ "Dark",
      TRUE ~ "None"
    )
  ) %>%
  select(-in.geometry_wall_exterior_finish) %>%
  mutate(
    in.ceiling_fan = as.integer(in.ceiling_fan == "Standard Efficiency")  # 1 if in use, 0 otherwise
  ) %>%
  mutate(
    in.usage_level = case_when(
      in.usage_level == "Low" ~ 1,
      in.usage_level == "Medium" ~ 2,
      in.usage_level == "High" ~ 3,
      TRUE ~ NA_real_  # for any cases that might not match the above, resulting in an NA
    )
  ) -> encoded_house_data
```

With the engineered features, we replicate the previous random forest to get more precise insights on the feature importance:

4. Building a Model for Prediction

The assignment consisted of predicting hourly energy consumption for each county in the month of July, assuming a temperature increase of 5 degrees Celsius.

Data Preprocessing and Initial Approach

As a preprocessing step, the energy consumption data was transformed using logarithms. This adjustment is particularly relevant given that our initial plots suggest an exponential relationship between energy consumption and temperature. By applying a logarithmic transformation, we not only facilitate a linear interpretation of this relationship but also allow the model's coefficients (betas) to be interpreted in terms of percentage change. Specifically, this transformation enables us to quantify the percentage increase in energy consumption corresponding to a one-unit change in each explanatory variable, thereby linearizing the originally exponential relationship for more effective analysis.

The initial concept involved developing a bottom-up model that incorporated key household features identified as significant influencers of energy consumption, with the aim of predicting individual household electricity consumption on an hourly basis. However, this approach was set aside due to computational constraints. Implementing such a model would have necessitated replicating each house's features 744 times to account for the data across 24 hours a day and 31 days, which proved to be impractical with the available computational resources. Consequently, we shifted our focus to an aggregated model that analyzes energy consumption at the county level, which offered a more feasible and computationally manageable approach.

Panel Data Regression with Fixed Effects

This subsequent approach treated this problem as a panel data regression with fixed effects, considering counties as the cross-sectional variable and the hour as the time variable. However, this approach yielded an R-squared of 0.13, resulting in an ineffective model.

Table 1:

	Dependent variable:
	total_electricity
temperature	0.032^{***}
	(0.0004)
Observations	32,982
R^2	0.139
Adjusted R^2	0.138
F Statistic	$5{,}328.392^{***}$ (df = 1; 32935)
Note:	*p<0.1; **p<0.05; ***p<0.01

Linear Regression Model

The next approach involved modeling a linear regression, where we extracted the hour and the day of the week as independent features. The regression model is as follows:

$y = a + bX + e$

where y represents the electricity consumption, and X includes the temperature,the county, the hour, and the day of the week. Notably, 'county', 'hour', and 'day of the week' are treated as categorical variables. The regression model is structured to provide insights into the impact of these variables on electricity consumption.Upon running the linear regression, we derived key summary statistics, which are presented in the table below:

Table 2: Summary Statistics of Linear Model

	Residual.Standard.Error	R.Squared	Adjusted.R.Squared	F.Statistic	DF1	DF2	P.Value
value	0.1009612	0.9920827	0.9920646	54977.1	75	32906	0

The output of the regression model indicates a strong fit, as evidenced by the significance of all coefficients for each iteration of the categorical variables, and an adjusted R-squared value of 0.99[1]. This high R-squared

[1]It is important to note that, given the time series nature of this problem with prominent daily and weekly seasonality, the most influential predictors for electricity consumption in each county are identified as the day of the week and the hour. These time-related variables capture the cyclic patterns of energy usage, making them crucial for understanding and predicting consumption trends.

value suggests that the model explains a substantial portion of the variance in electricity consumption based on the selected variables.

Model Evaluation and Comparison

To compare the predictive capability with other models, we conducted a train-test split of 80%-20% and computed performance metrics from 10-fold cross-validation:

Table 3: Performance Metrics on Test Set

	Value
RMSE	0.1012035
Rsquared	0.9920848
MAE	0.0771170

The metrics suggest that the model performs well. Considering an average hourly energy consumption of 4.4 kWh, the Root Mean Squared Error (RMSE) corresponds to an average error of approximately 2.3%. This relatively low error percentage indicates a high level of accuracy in the model's predictions.

Considering Non-linearities in Temperature

Nevertheless, we postulated that there might be non-linearities associated with temperature changes, as air conditioners, for instance, switch off below a certain temperature and on above a certain threshold. Consequently, we retried the regression with a quadratic term for temperature. The results were as follows:

Table 4: Performance Metrics on Test Set for non linearities in Temperature

	Value
RMSE	0.1008487
Rsquared	0.9921393
MAE	0.0768656

The differences in performance are minimal and does not justify the increased complexity of the model.

Assessment of Tree-Based Methods

Another potential approach is using a tree-based method, such as random forest regression. However, we do not prefer this method for our task, which involves extrapolation. Regression trees inherently struggle with extrapolation. They are non-parametric and local in nature, meaning they make predictions based on the structure discovered within the data they have been trained on. Consequently, a regression tree can only make predictions within the confines of the training data's range.

5. Forecast of future energy demand

6. Approaches to reduce energy demand

7. Conclusion

Annex 1

List of preselected features using domain expertise:

Feature
in.bedrooms
in.building_america_climate_zone
in.ceiling_fan
in.census_division_recs
in.city
in.clothes_dryer
in.cooling_setpoint
in.cooling_setpoint_has_offset
in.cooling_setpoint_offset_magnitude
in.cooling_setpoint_offset_period
in.corridor
in.county
in.ducts
in.federal_poverty_level
in.geometry_attic_type
in.geometry_building_horizontal_location_mf
in.geometry_building_horizontal_location_sfa
in.geometry_building_level_mf
in.geometry_building_number_units_mf
in.geometry_building_number_units_sfa
in.geometry_building_type_acs
in.geometry_floor_area
in.geometry_foundation_type
in.geometry_garage
in.geometry_stories
in.geometry_stories_low_rise
in.geometry_wall_exterior_finish
in.geometry_wall_type
in.has_pv
in.hvac_cooling_efficiency
in.hvac_cooling_partial_space_conditioning
in.hvac_cooling_type
in.hvac_has_ducts
in.hvac_has_shared_system
in.income
in.infiltration
in.insulation_ceiling
in.insulation_floor
in.insulation_foundation_wall

Feature
in.insulation_rim_joist
in.insulation_roof
in.insulation_slab
in.insulation_wall
in.interior_shading
in.misc_extra_refrigerator
in.misc_freezer
in.misc_hot_tub_spa
in.misc_pool
in.misc_pool_heater
in.misc_pool_pump
in.misc_well_pump
in.occupants
in.orientation
in.plug_load_diversity
in.plug_loads
in.pv_system_size
in.refrigerator
in.roof_material
in.sqft
in.state
in.tenure
in.usage_level
in.vacancy_status
in.vintage
in.water_heater_efficiency
in.water_heater_fuel
in.weather_file_latitude
in.weather_file_longitude
in.window_areas
in.windows