Nekilnojamojo turto objektų kainų analizė Lietuvoje

Statistikos laboratorinis darbas Nr. 2

VU

2025-04-17

Contents

1 Įvadas

Šiame tyrime analizuojami Lietuvos nekilnojamojo turto rinkos duomenys, siekiant nustatyti įvairius dėsningumus ir statistines priklausomybes.

2 Duomenų aprašymas

Analizei naudojami duomenys buvo atsisiųsti iš Lithuanian Real Estate Listings GitHub repozitorijos. Duomenys buvo surinkti 2024 m. vasarį iš Aruodas.lt puslapio. Duomenų rinkinyje yra informacija apie parduodamus ir nuomojamus butus, garažus, namus, sklypus ir patalpas. Tyrime naudojami duomenys apima kainų, ploto, vietos ir kitų svarbių charakteristikų informaciją.

2.1 Duomenų nuskaitymas

```
# Duomenų vieta
data_dir <- "C:/Users/zabit/Documents/GitHub/Statistikos-lab-2/data"

# Gauname aplankų pavadinimus
folders <- list.dirs(data_dir, full.names = FALSE, recursive = FALSE)

# Atspausdiname visų aplankų pavadinimus
kable(data.frame(Kategorijos = folders),
      caption = "Nekilnojamojo turto duomenų kategorijos") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 1: Nekilnojamojo turto duomenų kategorijos

Kategorijos
apartments
apartments_rent
garages_parking
garages_parking_rent
house_rent
houses
land
land_rent
premises
premises_rent

```
# CSV failų nuskaitymas į sąrašą
csv_data_list <- list()

for (folder in folders) {
  file_path <- file.path(data_dir, folder, "all_cities_20240214.csv")
  if (file.exists(file_path)) {
    # Bandyti nuskaityti failą
    tryCatch({
      df <- read.csv(file_path)
      csv_data_list[[folder]] <- df
      cat("Nuskaityta:", folder, nrow(df), "eilutėmis ir", ncol(df), "stulpeliais\n")
    }, error = function(e) {
      cat("Klaida nuskaitant", folder, ":", conditionMessage(e), "\n")
    })
  }
}
```

```
## Nuskaityta: apartments 7721 eilutėmis ir 38 stulpeliais
## Nuskaityta: apartments_rent 3208 eilutėmis ir 38 stulpeliais
## Nuskaityta: garages_parking 497 eilutėmis ir 28 stulpeliais
## Nuskaityta: garages_parking_rent 307 eilutėmis ir 27 stulpeliais
## Nuskaityta: house_rent 310 eilutėmis ir 40 stulpeliais
## Nuskaityta: houses 7284 eilutėmis ir 39 stulpeliais
## Nuskaityta: land 6322 eilutėmis ir 27 stulpeliais
## Nuskaityta: land_rent 104 eilutėmis ir 27 stulpeliais
## Nuskaityta: premises 1556 eilutėmis ir 37 stulpeliais
## Nuskaityta: premises_rent 2739 eilutėmis ir 37 stulpeliais
```

2.2 Duomenų patikrinimas ir išskirčių šalinimas

Prieš pradedant statistinę analizę, būtina identifikuoti ir pašalinti galimai klaidingas ar nekorektiškas reikšmes duomenyse. Nekilnojamojo turto rinkoje egzistuoja neįprastai didelių ar mažų kainų, kurios gali atsirasti dėl duomenų įvedimo klaidų, klaidingo formato ar kitų priežasčių. Tokios išskirtys gali reikšmingai paveikti statistinės analizės rezultatus.

```r
# Apibrėžiame kainų ribas išskirčių identifikavimui
min_threshold <- 20          # Minimali patikima kaina eurais
max_threshold <- 25000000    # Maksimali patikima kaina eurais

# Sukuriame rezultatų lentelę
removal_results <- data.frame(
  Kategorija = character(),
  Pašalinta_eilučių = integer(),
  Per_didelės_kainos = integer(),
  Per_mažos_kainos = integer(),
  stringsAsFactors = FALSE
)

# Tikriname ir šaliname išskirtis kiekviename duomenų rinkinyje
for (type in names(csv_data_list)) {
  if (!is.null(csv_data_list[[type]]) && "price" %in% colnames(csv_data_list[[type]])) {
    # Identifikuojame kraštutines reikšmes
    extreme_high <- sum(csv_data_list[[type]]$price > max_threshold, na.rm = TRUE)
    extreme_low <- sum(csv_data_list[[type]]$price < min_threshold, na.rm = TRUE)
    extreme_total <- extreme_high + extreme_low

    if (extreme_total > 0) {
      # Išsaugome pradinį eilučių skaičių
      original_count <- nrow(csv_data_list[[type]])

      # Filtruojame duomenis, išlaikydami tik patikimas kainas arba NA reikšmes
      csv_data_list[[type]] <- csv_data_list[[type]][
        (csv_data_list[[type]]$price >= min_threshold &
         csv_data_list[[type]]$price <= max_threshold) |
          is.na(csv_data_list[[type]]$price), ]

      # Fiksuojame rezultatus
      new_count <- nrow(csv_data_list[[type]])
      removed_count <- original_count - new_count

      # Pridedame rezultatus į suvestinę
      removal_results <- rbind(removal_results, data.frame(
        Kategorija = type,
        Pašalinta_eilučių = removed_count,
        Per_didelės_kainos = extreme_high,
        Per_mažos_kainos = extreme_low
      ))
    }
  }
}

# Atvaizduojame išskirčių šalinimo rezultatus
if (nrow(removal_results) > 0) {
  kable(removal_results,
        caption = "Išskirčių šalinimo rezultatų suvestinė") %>%
    kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
} else {
  cat("Duomenyse nerasta kainų, kurios viršytų nustatytas ribas.")
```

```
}
```

Table 2: Išskirčių šalinimo rezultatų suvestinė

Kategorija	Pašalinta_eilučių	Per_didelės_kainos	Per_mažos_kainos
land_rent	2	0	2
premises	65	64	1
premises_rent	192	159	33

```
# Patikriname duomenų rinkinių dydžius po valymo
data_sizes <- data.frame(
  Eilučių_skaičius = sapply(csv_data_list, nrow),
  Stulpelių_skaičius = sapply(csv_data_list, ncol)
)

kable(data_sizes,
      caption = "Duomenų rinkinių dydžiai po išskirčių šalinimo") %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"))
```

Table 3: Duomenų rinkinių dydžiai po išskirčių šalinimo

	Eilučių_skaičius	Stulpelių_skaičius
apartments	7721	38
apartments_rent	3208	38
garages_parking	497	28
garages_parking_rent	307	27
house_rent	310	40
houses	7284	39
land	6322	27
land_rent	102	27
premises	1491	37
premises_rent	2547	37

Pašalintos ekstremalios kainos, kurios galėjo iškreipti vidutines reikšmes ir kitas statistines charakteristikas.

```
# Output summary of data
cat("\nSummary of all loaded datasets:\n")
```

```
##
## Summary of all loaded datasets:
```

```
for (folder_name in names(csv_data_list)) {
  cat("\nDataset from folder:", folder_name, "\n")
  cat("Number of rows:", nrow(csv_data_list[[folder_name]]), "\n")
  cat("Number of columns:", ncol(csv_data_list[[folder_name]]), "\n")
  cat("Column names:", paste(colnames(csv_data_list[[folder_name]]), collapse = ", "), "\n")
}
```

```
##
## Dataset from folder: apartments
## Number of rows: 7721
## Number of columns: 38
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: apartments_rent
## Number of rows: 3208
## Number of columns: 38
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: garages_parking
## Number of rows: 497
## Number of columns: 28
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: garages_parking_rent
## Number of rows: 307
## Number of columns: 27
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: house_rent
## Number of rows: 310
## Number of columns: 40
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: houses
## Number of rows: 7284
## Number of columns: 39
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: land
## Number of rows: 6322
## Number of columns: 27
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: land_rent
## Number of rows: 102
## Number of columns: 27
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: premises
## Number of rows: 1491
## Number of columns: 37
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
##
## Dataset from folder: premises_rent
## Number of rows: 2547
## Number of columns: 37
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descri
```

```r
# Get all unique column names across all datasets
all_columns <- unique(unlist(lapply(csv_data_list, colnames)))
unique_columns <- sort(all_columns)
```

```r
# Display unique column names and count
cat("Total unique columns across all datasets:", length(unique_columns), "\n")
```

```
## Total unique columns across all datasets: 52
```

```r
cat("Unique column names:", paste(unique_columns, collapse = ", "), "\n")
```

```
## Unique column names: accommodates_no._of_cars, add_date, additional_equipment, additional_premises, a
```

```r
# Display sample values for each unique column
cat("Sample values for each unique column:\n")
```

```
## Sample values for each unique column:
```

```r
for (col_name in unique_columns) {
  cat("\n", col_name, ":\n")
  found_values <- FALSE

  # Look for this column in each dataset
  for (dataset_name in names(csv_data_list)) {
    df <- csv_data_list[[dataset_name]]

    # Check if this column exists in the current dataset
    if (col_name %in% colnames(df)) {
      # Extract non-NA values
      non_na_values <- df[[col_name]][!is.na(df[[col_name]])]

      # If we have non-NA values
      if (length(non_na_values) > 0) {
        # Take up to 2 samples
        sample_size <- min(2, length(non_na_values))
        samples <- non_na_values[1:sample_size]

        # Display the samples with the dataset name
        cat("  From ", dataset_name, ": ",
            paste(samples, collapse = ", "),
            if(length(non_na_values) > 3) " ..." else "", "\n", sep = "")

        found_values <- TRUE
        break  # Only show from one dataset to keep output manageable
      }
    }
  }

  if (!found_values) {
    cat("  No non-NA values found in any dataset\n")
  }
}
```

```
##
```

```
##  accommodates_no._of_cars :
##   From garages_parking: 1, 1 ...
##
##  add_date :
##   From apartments: 2023-11-17, 2024-01-15 ...
##
##  additional_equipment :
##   From apartments: ,  ...
##
##  additional_premises :
##   From apartments: ,  ...
##
##  area :
##   From apartments: 29,75, 82.0 ...
##
##  area_.a. :
##   From land: 97,8 a, 15 a ...
##
##  build_year :
##   From apartments: 1966, 1981 ...
##
##  building_energy_efficiency_class :
##   From apartments: ,  ...
##
##  building_type :
##   From apartments: Block house, Block house ...
##
##  call_forwarding :
##   From apartments: False, False ...
##
##  closest_body_of_water :
##   From house_rent: , Pond ...
##
##  coordinates :
##   From apartments: 54.91171,23.97343, 54.93039,23.93837 ...
##
##  description :
##   From apartments: PRIVALUMAI:
## PARDUOTAS!!!!
##
##  su visais jame esančiais baldais ir buitine technika;
##   Pageidaujantiems galima gyventi po savaites :)
## (prieš 5 metus buvo atliktas kapitalinis remontas
## remontas( nauja santechnika, naujai pravesta elektra . Savininkas ypatingai prižiūrėjo savo turtą, ta
##  2 aukštas
##  tvarkinga namo aplinka, tvarkingas, prižiūretas rūsys .
##  prižiūri draugiška namo bendruomenė;
##
## Didelis , tvarkingas balkonas!!
##
## VIETA
##
##  mokyklos, darželiai , 2 gražūs parkai- ejimo atstumu;
##  Girstučio baseinas 300m
```

```
##  Girstučio teatras 300m
##  turgus 350m
##  didieji prekybos centrai (IKI, MAXIMA, LIDL ir t.t)
##  Urmo bazė
##
## Parduodama iš pirmų rankų. Tarpininkavimo paslaugų nesiūlyti., PARDUODAMAS ERDVUS 82 KV. M. 4 KAMBARI
##
## ĮRENGIMAS. Parduodamas butas 82 kv.m. 4 nepereinamų kambarių. Virtuvė didelė ir erdvi. Virtuvėje sum
##
## NAMAS. Parduodamas butas yra devynių aukštų daugiabučio 6 aukšte. Namo laiptinė tvarkinga ir prižiūri
##
## VIETA. Parduodamas butas yra Šiaurės pr. šalia vadinamo Šiaurės žiedo. Iš šios vietos labai patogu pa
##
##  Daugiau informacijos telefonu!
## ------------------------------------------------------------------------------------------------
## Padėsime jums profesionaliai ir greitai gauti paskolą kredito įstaigose. Mūsų partneriai: nepriklaus
##
##  description_tags :
##   From apartments: ,  ...
##
##  distance_from_body_of_water :
##   From house_rent: , 1 ...
##
##  equipment :
##   From apartments: Fully equipped, Fully equipped ...
##
##  features :
##   From garages_parking: Pit, Automatic gates, under the roof ...
##
##  flat_no. :
##   From apartments: , 16 ...
##
##  floor :
##   From apartments: 2, 6 ...
##
##  heating_system :
##   From apartments: Central, Central ...
##
##  house_no. :
##   From apartments: , 79 ...
##
##  images :
##   From apartments: , https://aruodas-img.dgn.lt/object_61_115008957/kaunas-eiguliai-siaures-pr.jpg,h
##
##  link :
##   From apartments: aruodas.lt/1-3381778, aruodas.lt/1-3395115 ...
##
##  listing_id :
##   From apartments: 3381778, 3395115 ...
##
##  lot_no. :
##   From land: ,  ...
##
##  microdistrict :
```

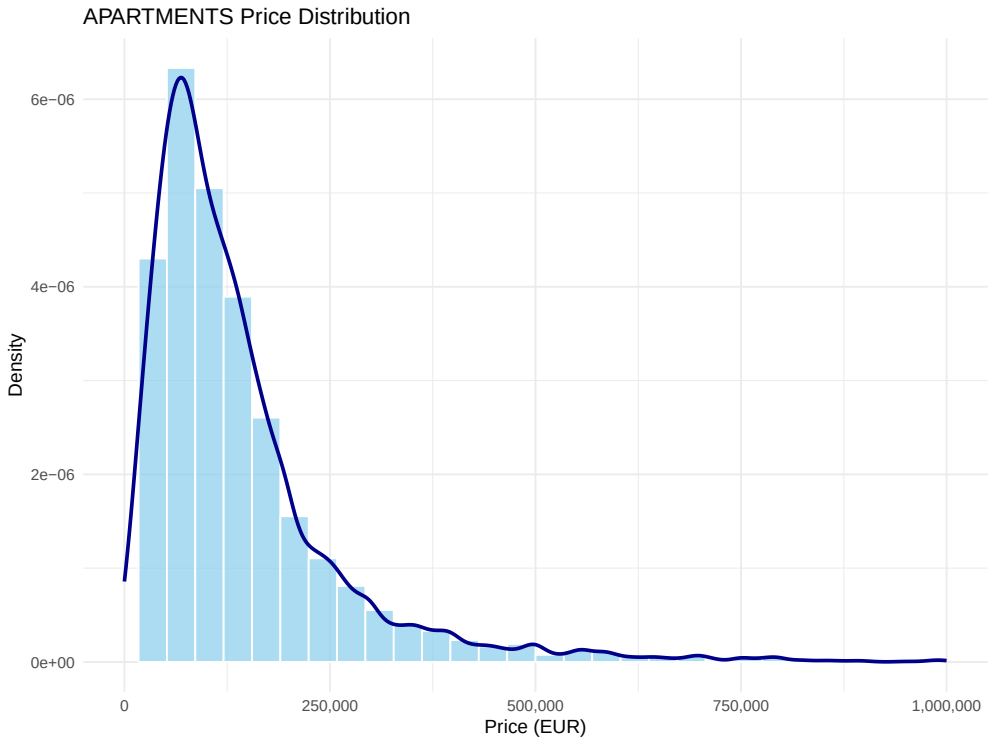
```
##   From apartments: Dainava, Eiguliai ...
##
##  modified :
##   From apartments: 2024-02-12, 2024-01-15 ...
##
##  no._of_floors :
##   From apartments: 5, 9 ...
##
##  number :
##   From garages_parking: , 18 ...
##
##  number_of_rooms :
##   From apartments: 1, 4 ...
##
##  object :
##   From apartments: ,  ...
##
##  phone_number :
##   From apartments: 37060707730, 37068211050 ...
##
##  plot_area :
##   From house_rent: 1 a, 37 a ...
##
##  premises_nr. :
##   From premises: , 13 ...
##
##  premises_sum :
##   From premises: 4,  ...
##
##  price :
##   From apartments: 63000, 98000 ...
##
##  price_per_month :
##   From apartments_rent: 380 €, 430 € ...
##
##  private_seller :
##   From apartments: False, False ...
##
##  purpose :
##   From land: Residential land, Residential land ...
##
##  region :
##   From apartments: Kaunas, Kaunas ...
##
##  reserved :
##   From apartments: False, False ...
##
##  security :
##   From apartments: ,  ...
##
##  selected :
##   From apartments: 21, 7 ...
##
##  sold_or_rented :
```

```
##   From apartments: False, False ...
##
##  street :
##   From apartments: Kovo 11-osios g., Šiaurės pr. ...
##
##  type :
##   From garages_parking: For sale, For sale ...
##
##  type_id :
##   From apartments: 1, 1 ...
##
##  unique_item_number :
##   From apartments: ,  ...
##
##  valid_till :
##   From apartments: ,  ...
##
##  views_today :
##   From apartments: 0, 0 ...
##
##  views_total :
##   From apartments: 2264, 579 ...
##
##  water_system :
##   From house_rent: , Private water supply system ...
```

2. Išbrėžkite turimų duomenų grafikus (parinkite tinkamiausius). Manau kokių 4 užtektų

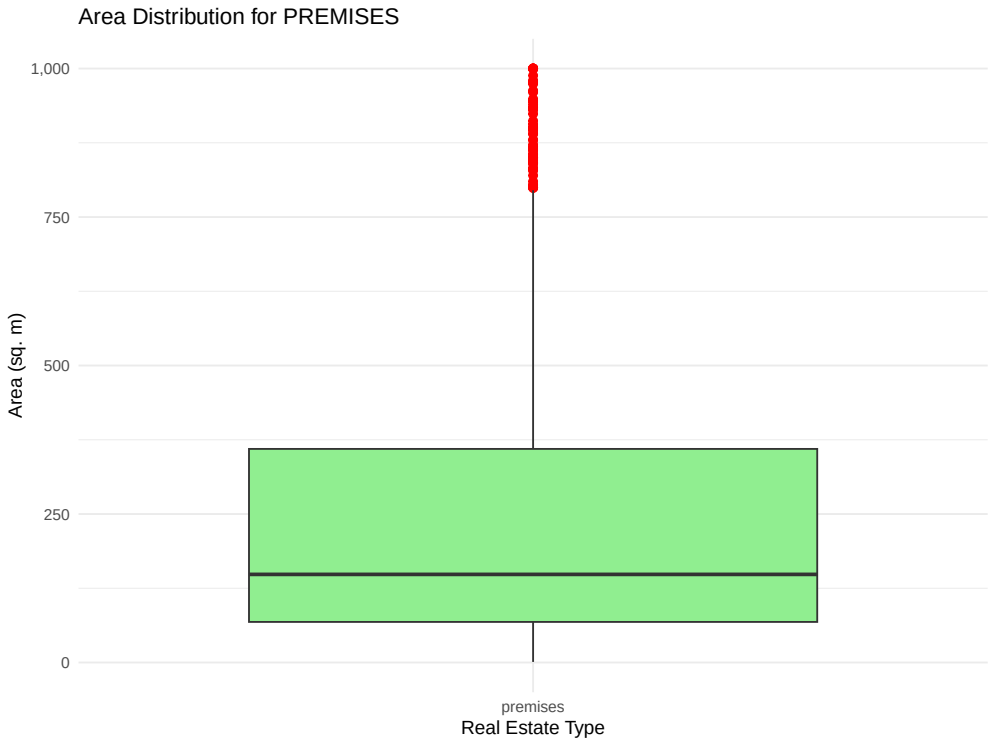
```r
# Reset the plot layout
par(mfrow = c(1, 1))

library(ggplot2)

for (type in c("apartments")) {
  if (type %in% names(csv_data_list) && "price" %in% colnames(csv_data_list[[type]])) {
    # Get price data and create a data frame
    df <- data.frame(price = csv_data_list[[type]]$price)

    # Create plot object with fixed deprecated features
    p <- ggplot(df, aes(x = price)) +
      geom_histogram(aes(y = after_stat(density)),
                     bins = 30,
                     fill = "skyblue",
                     color = "white",
                     alpha = 0.7) +
      geom_density(color = "darkblue", linewidth = 1) + # Fixed: size -> linewidth
      labs(title = paste(toupper(type), "Price Distribution"),
           x = "Price (EUR)",
           y = "Density") +
      theme_minimal() +
      scale_x_continuous(labels = scales::comma, limits = c(0, 1000000)) +
      coord_cartesian(xlim = c(0, 1000000))

    # Print the plot
```

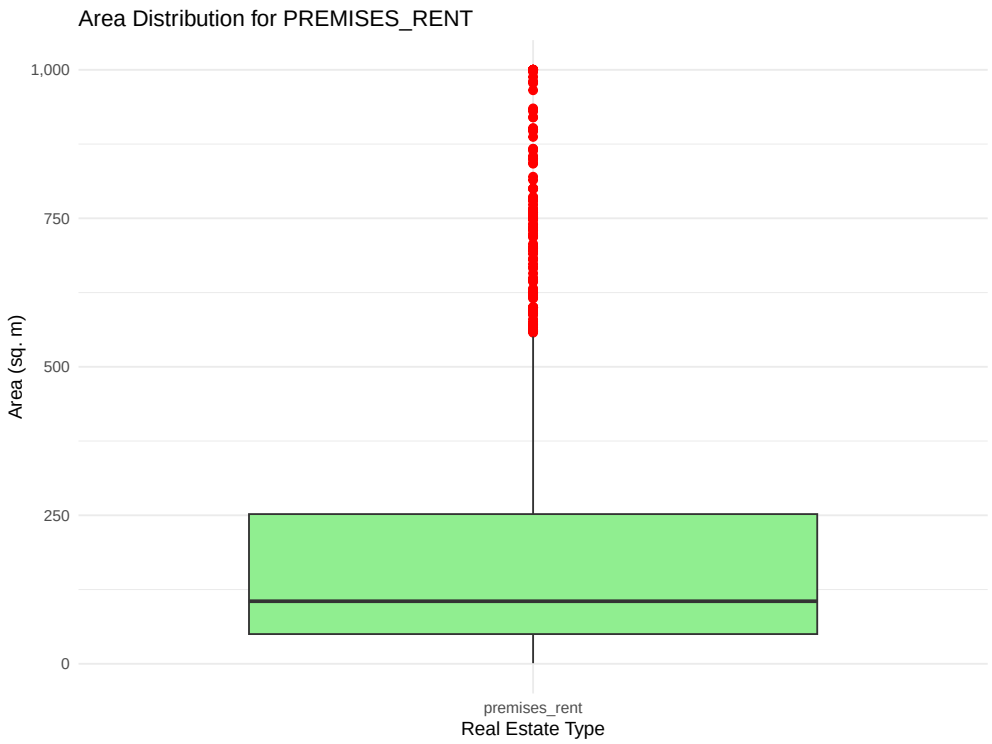
```
    print(p)
  }
}
```



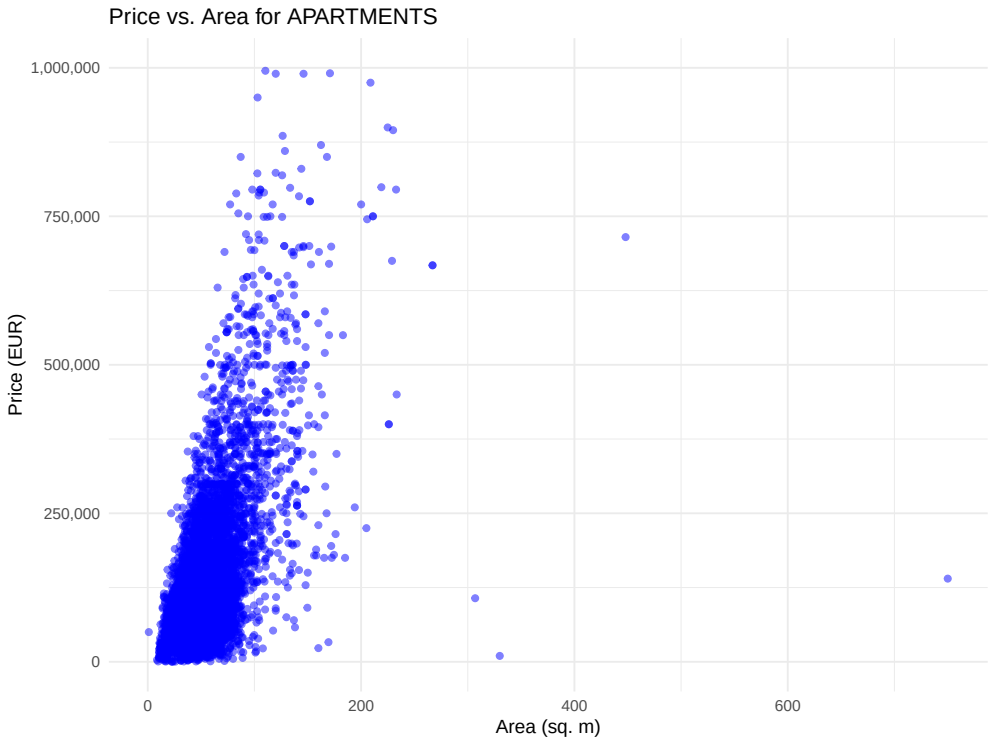

```
# Boxplot: Area distribution for premises and premises_rent only
premises_types <- c("premises", "premises_rent")

# Create a list to store both plots
boxplot_list <- list()

for (type in premises_types) {
  if (type %in% names(csv_data_list) && "area" %in% colnames(csv_data_list[[type]])) {
    df <- csv_data_list[[type]]
    df$type <- type  # Add type as a column

    # Ensure area is numeric
    df$area <- as.numeric(gsub(",", ".", as.character(df$area)))

    # Create boxplot
    p <- ggplot(df, aes(x = type, y = area)) +
      geom_boxplot(fill = "lightgreen", outlier.color = "red", outlier.size = 2) +
      labs(title = paste("Area Distribution for", toupper(type)),
           x = "Real Estate Type",
           y = "Area (sq. m)") +
      theme_minimal() +
      scale_y_continuous(labels = scales::comma, limits = c(0, 1000)) +
      coord_cartesian(ylim = c(0, 1000))
```

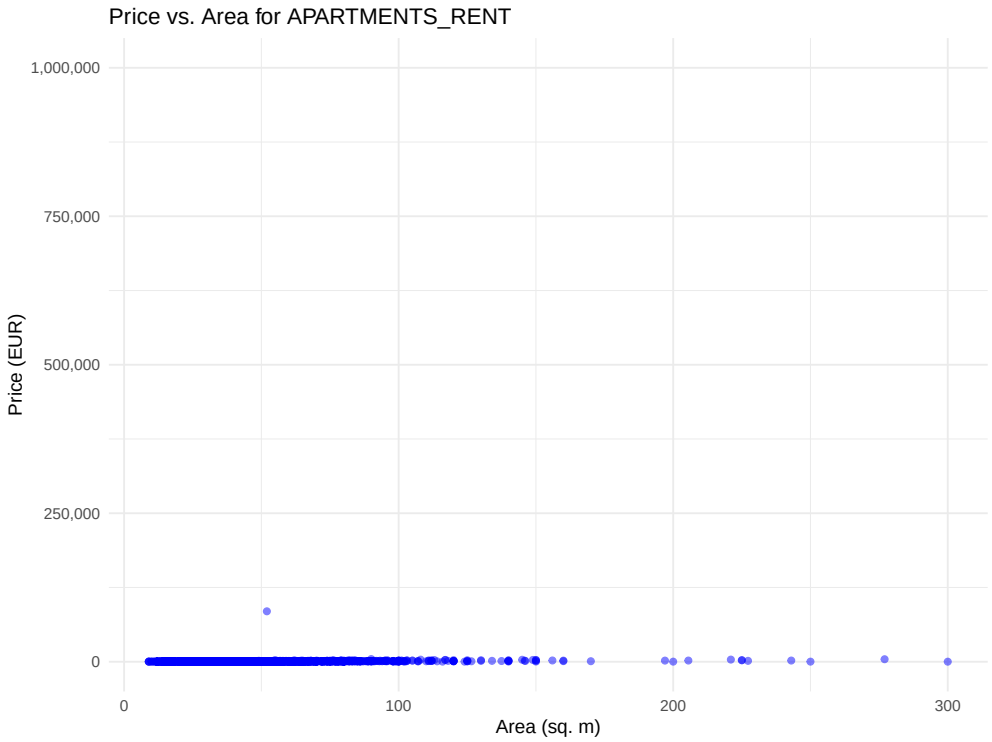
```
    # Print the plot
    print(p)
  } else {
    cat("Dataset", type, "is not available or doesn't have 'area' column\n")
  }
}
```

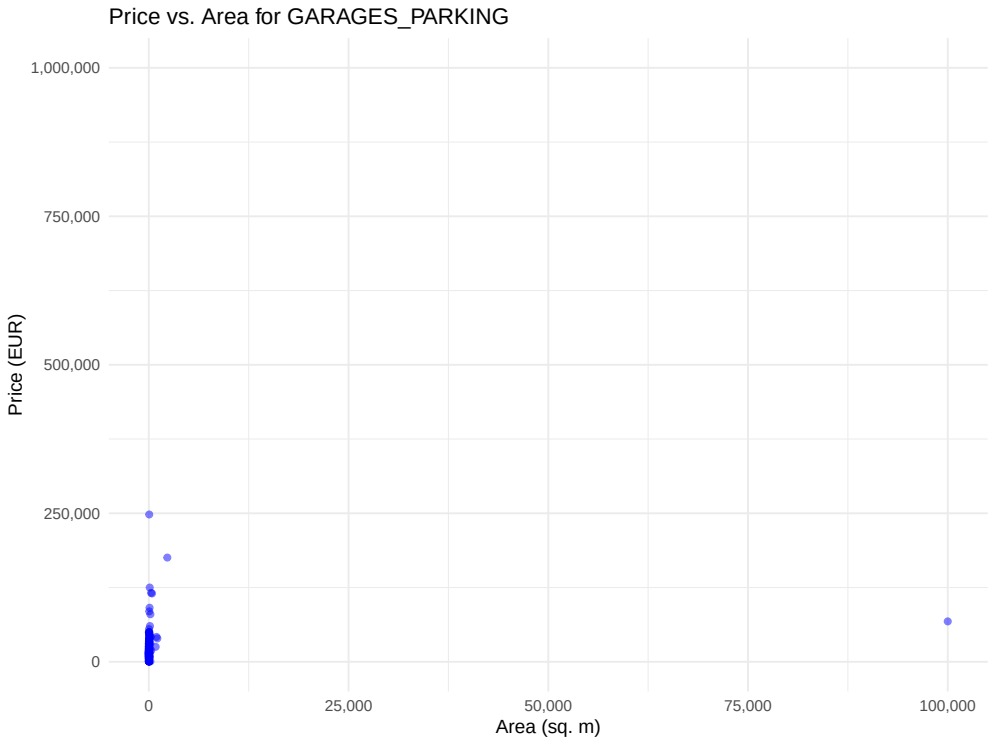


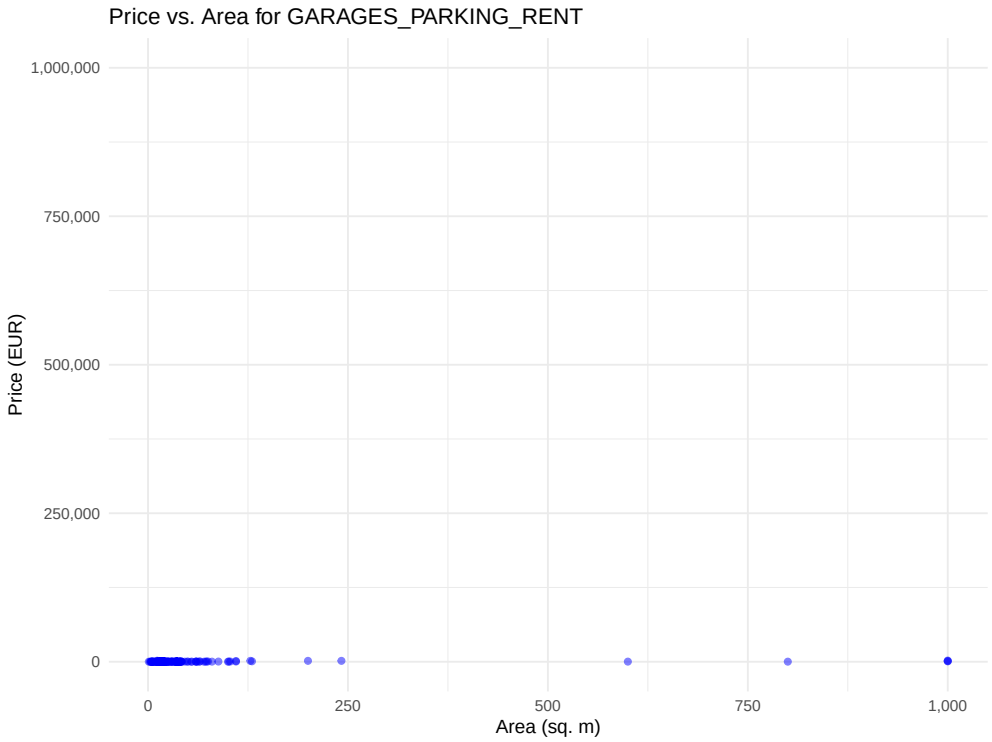


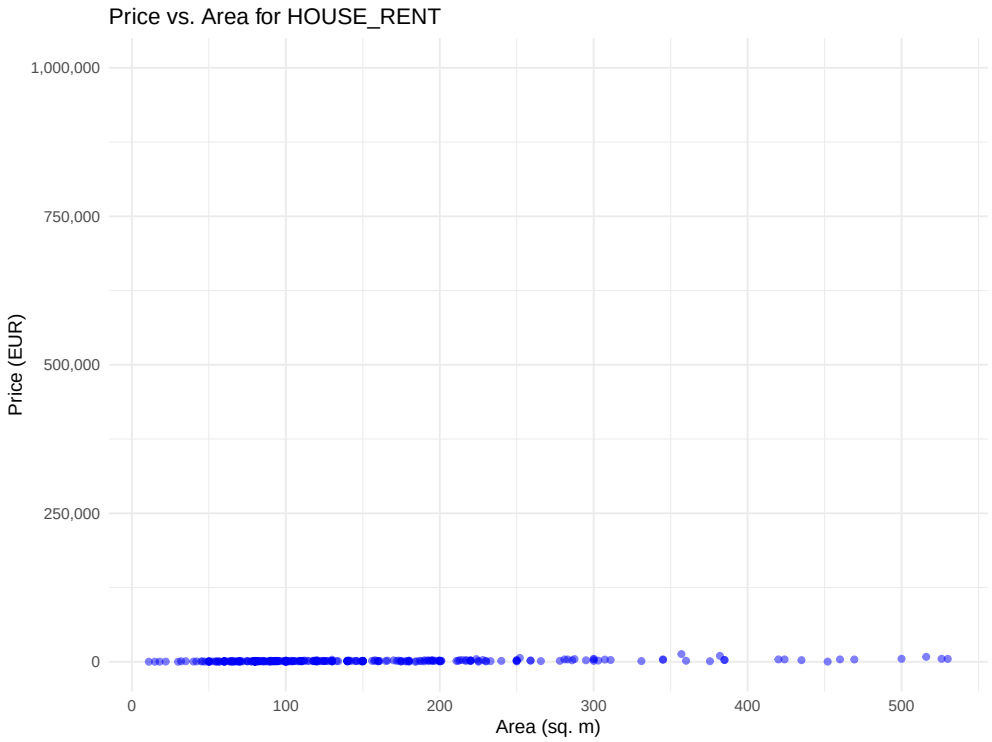

```r
# Scatter Plot: Price vs. Area
for (type in names(csv_data_list)) {
  if (!is.null(csv_data_list[[type]]) && all(c("price", "area") %in% colnames(csv_data_list[[type]]))) {
    df <- csv_data_list[[type]]

    # Standardize the area column: replace commas with dots and convert to numeric
    df$area <- as.numeric(gsub(",", ".", as.character(df$area)))

    # Create scatter plot
    p <- ggplot(df, aes(x = area, y = price)) +
      geom_point(color = "blue", alpha = 0.5) +
      labs(title = paste("Price vs. Area for", toupper(type)),
           x = "Area (sq. m)",
           y = "Price (EUR)") +
      theme_minimal() +
      scale_y_continuous(labels = scales::comma, limits = c(0, 1000000)) +
      scale_x_continuous(labels = scales::comma)

    print(p)
  }
}
```

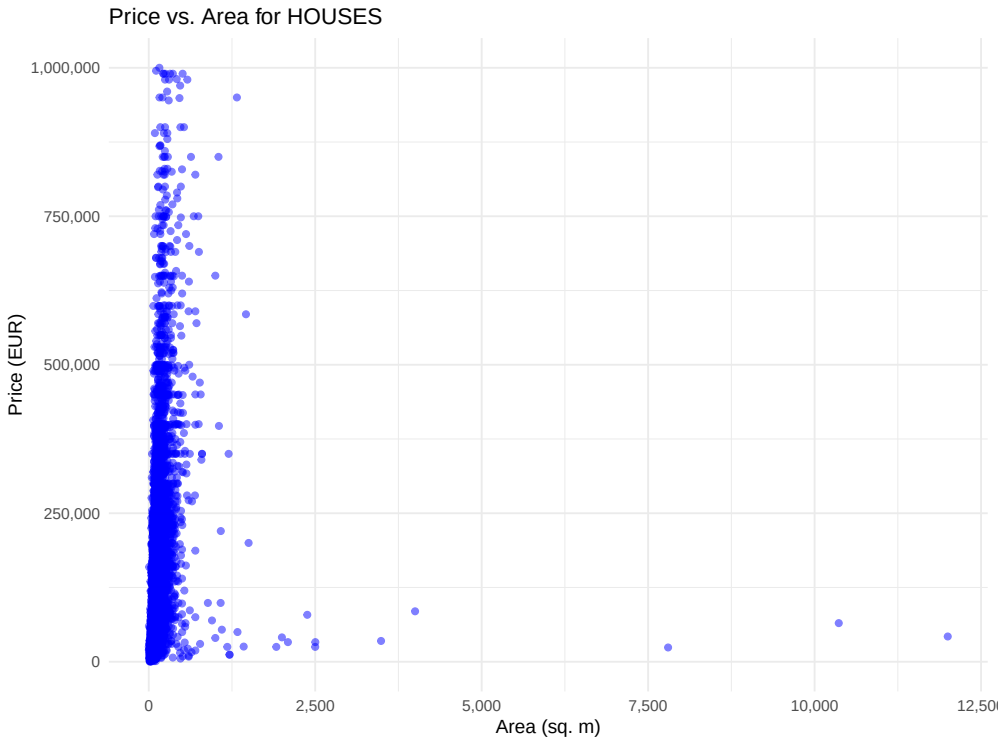

Price vs. Area for APARTMENTS
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
200
400
600
Area (sq. m)

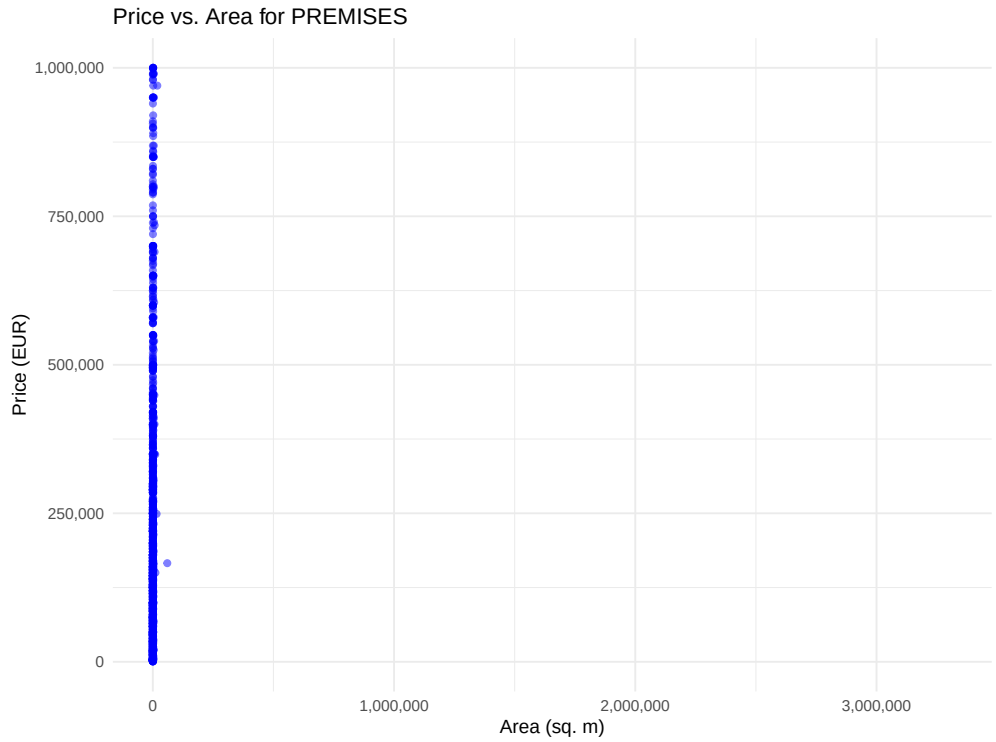

Price vs. Area for APARTMENTS_RENT
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
100
200
300
Area (sq. m)

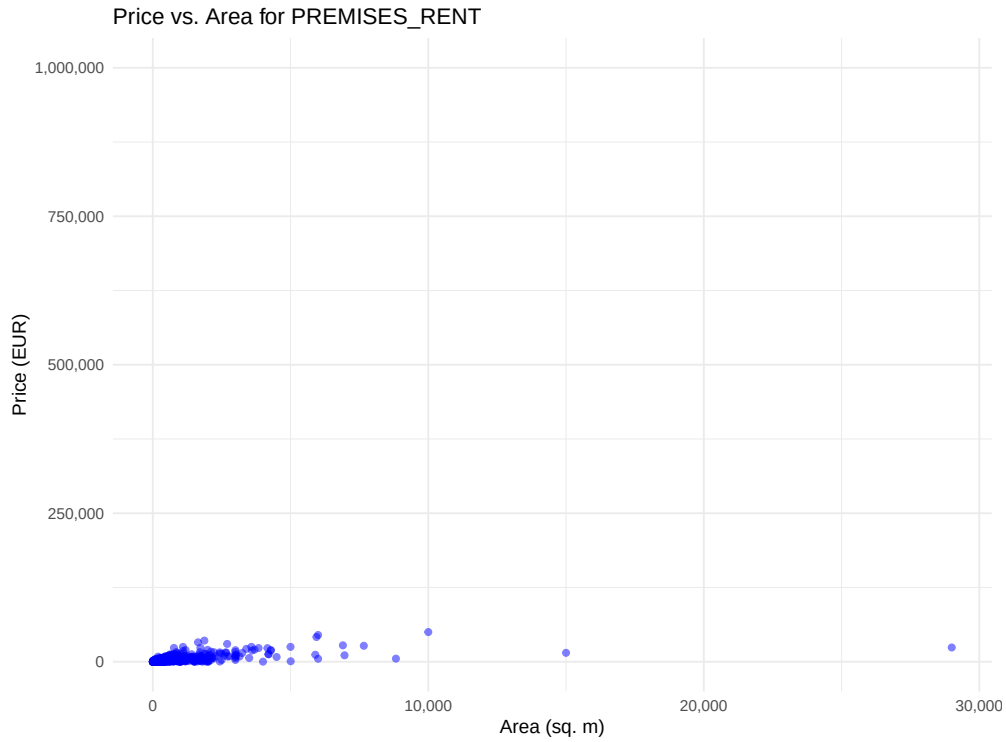

Price vs. Area for GARAGES_PARKING
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
25,000
50,000
75,000
100,000
Area (sq. m)


Price vs. Area for GARAGES_PARKING_RENT
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
250
500
750
1,000
Area (sq. m)


Price vs. Area for HOUSE_RENT
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
100
200
300
400
500
Area (sq. m)

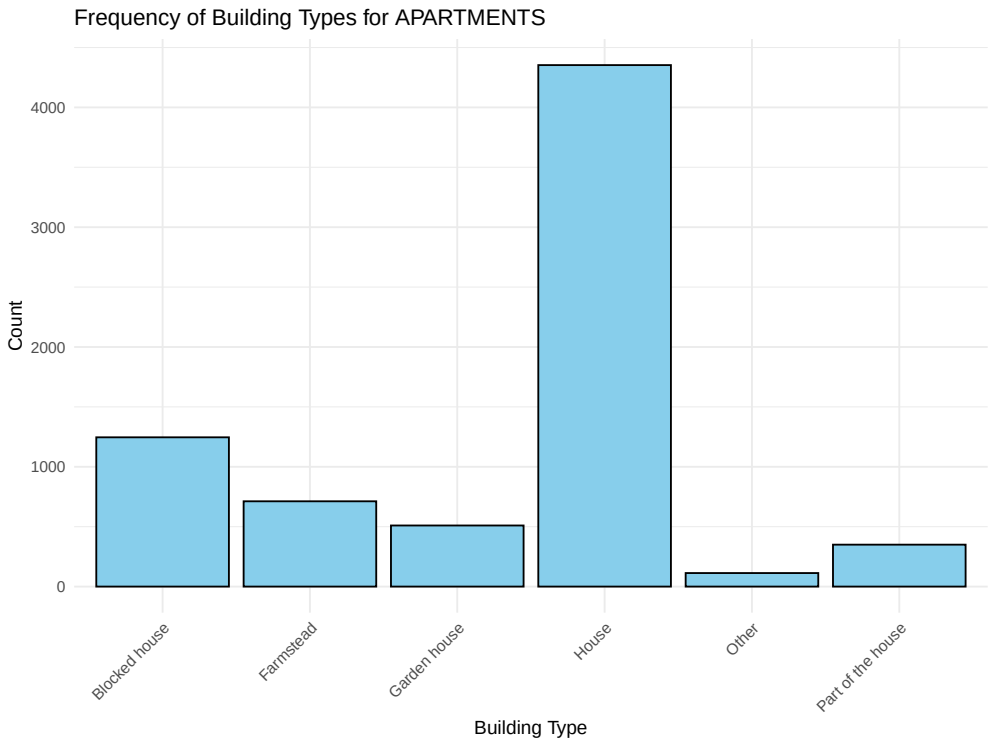

Price vs. Area for HOUSES
1,000,000
750,000
500,000
250,000
0
Price (EUR)
0
2,500
5,000
7,500
10,000
Area (sq. m)






```r
# Bar Chart: Frequency of Building Types for "apartments"
if ("houses" %in% names(csv_data_list) && "building_type" %in% colnames(csv_data_list[["houses"]])) {
  df <- csv_data_list[["houses"]]
```

```r
  # Create bar chart
  p <- ggplot(df, aes(x = building_type)) +
    geom_bar(fill = "skyblue", color = "black") +
    labs(title = "Frequency of Building Types for APARTMENTS",
         x = "Building Type",
         y = "Count") +
    theme_minimal() +
    theme(axis.text.x = element_text(angle = 45, hjust = 1))

  print(p)
}
```



3. Apskaičiuokite pagrindines skaitines charakteristikas kiekybiniams kintamiesiems. Mes apskaičiavome šias skaitines charakteristikas:

   Vidurkis (Mean)

   Mediana (Median)

   Moda (Mode)

   Dispersija (Variance)

   Standartinis nuokrypis (Standard deviation)

   Kvartiliai (Quartiles) - 0.25, 0.5, 0.75

   Minimumuas

   Maksimumas

Kiekybiniai duomenys: kaina (“price”), peržiūrų skaičius (“views_total”), būsto dydis (“area” iš apartments), žemės ploto dydis (“area_.a.” iš land), build_year iš apartments, buto aukštas (“floor”), kambarių skaičius (“number_of_rooms”), plot_area, price_per_month.

```r
# Create a helper function to filter datasets by column name
filter_datasets_by_column <- function(data_list, column_name) {
  filtered <- data_list[sapply(data_list, function(df) column_name %in% colnames(df))]
  # cat("Datasets with column", column_name, ":\n")
  # print(names(filtered))
  return(filtered)
}

# List of columns to check
columns_to_check <- c(
  "price", "price_per_month", "views_total", "area", "area_.a.",
  "build_year", "no._of_floors", "floor", "number_of_rooms", "plot_area"
)

# Create a list to store results
column_results <- list()

# Process each column and store results
for (col in columns_to_check) {
  column_results[[col]] <- filter_datasets_by_column(csv_data_list, col)
}
```

```r
# Simplified function to calculate summary statistics for a specified variable across multiple datasets
calculate_summary <- function(data_list, variable_name, target_datasets) {
  for (df_name in target_datasets) {
    if (df_name %in% names(data_list) && variable_name %in% colnames(data_list[[df_name]])) {
      cat("Summary for variable '", variable_name, "' in dataset '", df_name, "':\n", sep="")
      print(summary(data_list[[df_name]][[variable_name]]))
      cat("\n")
    } else {
      cat("Dataset '", df_name, "' does not exist or does not have a '", variable_name, "' column.\n", s
    }
  }
}

# Define dataset groups
sale_datasets <- c("apartments", "garages_parking", "houses", "land", "premises")
rent_datasets <- c("apartments_rent", "house_rent", "premises_rent")
all_datasets <- c("apartments", "apartments_rent", "garages_parking", "garages_parking_rent",
                  "house_rent", "houses", "land", "land_rent", "premises", "premises_rent")

# Calculate summary for price in sale datasets
calculate_summary(csv_data_list, "price", sale_datasets)
```

```
## Summary for variable 'price' in dataset 'apartments':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      43   64000  107558  143718  172000 2500000
##
## Summary for variable 'price' in dataset 'garages_parking':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     500   10000   15000   19016   22499  248000
##
## Summary for variable 'price' in dataset 'houses':
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     200   55000  140000  183734  235000 4200000
##
## Summary for variable 'price' in dataset 'land':
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      100    18000    35000   115389    79900 12000000
##
## Summary for variable 'price' in dataset 'premises':
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      490    70000   165000   413170   399850 10000000
```

```r
# Calculate summary for price in rent datasets
calculate_summary(csv_data_list, "price", rent_datasets)
```

```
## Summary for variable 'price' in dataset 'apartments_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      20     380     525     610     690   84900
##
## Summary for variable 'price' in dataset 'house_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      50     750    1200    1429    1500   13000
##
## Summary for variable 'price' in dataset 'premises_rent':
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##       22      500     1300   886473     5268 24045000
```

```r
# Calculate summary for views_total across all datasets
calculate_summary(csv_data_list, "views_total", all_datasets)
```

```
## Summary for variable 'views_total' in dataset 'apartments':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       0     425     892    1573    1860   56297
##
## Summary for variable 'views_total' in dataset 'apartments_rent':
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      2.0    285.8    606.5   1806.0   1315.2 355786.0
##
## Summary for variable 'views_total' in dataset 'garages_parking':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    13.0   194.0   433.0   726.7   876.0 12209.0
##
## Summary for variable 'views_total' in dataset 'garages_parking_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     6.0    80.0   173.0   374.1   404.5  7521.0
##
## Summary for variable 'views_total' in dataset 'house_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    20.0   261.8   582.5  1274.6  1411.2 24014.0
##
## Summary for variable 'views_total' in dataset 'houses':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2     501    1133    2247    2612   71418
##
```

```
## Summary for variable 'views_total' in dataset 'land':
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      1.0    140.0    346.5    869.2    871.8 191374.0
##
## Summary for variable 'views_total' in dataset 'land_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    11.0   100.2   255.5   477.4   619.0  2658.0
##
## Summary for variable 'views_total' in dataset 'premises':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   132.0   310.0   646.8   710.5 21298.0
##
## Summary for variable 'views_total' in dataset 'premises_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     1.0   106.0   257.0   742.2   607.0 46715.0
```

```
# Calculate summary for no._of_floors
calculate_summary(csv_data_list, "no._of_floors", all_datasets)
```

```
## Summary for variable 'no._of_floors' in dataset 'apartments':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   3.000   5.000   5.066   5.000  34.000
##
## Summary for variable 'no._of_floors' in dataset 'apartments_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   4.000   5.000   5.322   6.000  34.000
##
## Dataset 'garages_parking' does not exist or does not have a 'no._of_floors' column.
## Dataset 'garages_parking_rent' does not exist or does not have a 'no._of_floors' column.
## Summary for variable 'no._of_floors' in dataset 'house_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   1.816   2.000   4.000
##
## Summary for variable 'no._of_floors' in dataset 'houses':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   1.000   2.000   1.579   2.000  15.000     152
##
## Dataset 'land' does not exist or does not have a 'no._of_floors' column.
## Dataset 'land_rent' does not exist or does not have a 'no._of_floors' column.
## Summary for variable 'no._of_floors' in dataset 'premises':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   1.000   2.000   2.363   3.000  18.000     725
##
## Summary for variable 'no._of_floors' in dataset 'premises_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   1.000   2.000   2.815   3.000  31.000    1617
```

```
# Calculate summary for number_of_rooms
calculate_summary(csv_data_list, "number_of_rooms", all_datasets)
```

```
## Summary for variable 'number_of_rooms' in dataset 'apartments':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   2.000   2.000   2.385   3.000  13.000
```

```
##
## Summary for variable 'number_of_rooms' in dataset 'apartments_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   1.000   1.000   2.000   1.956   2.000  10.000
##
## Dataset 'garages_parking' does not exist or does not have a 'number_of_rooms' column.
## Dataset 'garages_parking_rent' does not exist or does not have a 'number_of_rooms' column.
## Summary for variable 'number_of_rooms' in dataset 'house_rent':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   3.000   4.000   4.174   5.000  13.000     103
##
## Summary for variable 'number_of_rooms' in dataset 'houses':
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##   1.000   3.000   4.000   4.205   5.000  54.000    2762
##
## Dataset 'land' does not exist or does not have a 'number_of_rooms' column.
## Dataset 'land_rent' does not exist or does not have a 'number_of_rooms' column.
## Dataset 'premises' does not exist or does not have a 'number_of_rooms' column.
## Dataset 'premises_rent' does not exist or does not have a 'number_of_rooms' column.
```

4. Sudarykite dažnių lenteles kategoriniams kintamiesiems.
5. Suformuluokite bent 6 tyrimo hipotezes iš savo duomenų rinkinio
6. Užrašykite kokius testus parinkote savo tyrimo hipotezėms. Hipotezės turi būti skirtos skirtingų testų naudojimui. Jei reikia susikurkite naujus kintamuosius iš turimų duomen.
7. Patikrinkite, ar kintamieji tenkina būtinas sąlygas testų taikymui. Jei netenkina, atlikite duomenų transformacijas.
8. Atlikite statistinį tyrimą savo suformuluotoms hipotezėms.
9. Pateikite tyrimo atsakymą.