Statistikos laboratorinis darbas Nr. 2

VU

2025-03-27

Contents

1. Aprašykite turimus duomenis, nurodykite duomenų šaltinį.

Duomenys atsisiųsti iš https://github.com/valdas-v1/lithuanian-real-estate-listings. Duomenys buvo surinkti 2024 m. vasarį iš https://www.aruodas.lt/ puslapio. Duomenų rinkinyje yra informacija apie parduodamus ir nuomojamus butus, garažus, namus, sklypus ir patalpas.

```
# Set the path to the data directory
data_dir <- "C:/Users/zabit/Documents/GitHub/Statistikos-lab-2/data"

# Get all folder names inside the data directory
folders <- list.dirs(data_dir, full.names = FALSE, recursive = FALSE)

# Print all folder names
print(folders)
```

```
##  [1] "apartments"           "apartments_rent"      "garages_parking"
##  [4] "garages_parking_rent" "house_rent"           "houses"
##  [7] "land"                 "land_rent"            "premises"
## [10] "premises_rent"
```

```
# Read the CSV files into a list of dataframes
csv_data_list <- list()

for (folder in folders) {
  file_path <- file.path(data_dir, folder, "all_cities_20240214.csv")
  if (file.exists(file_path)) {
    # Try reading the file
    tryCatch({
      df <- read.csv(file_path)
      csv_data_list[[folder]] <- df
      cat("Read:", folder, "with", nrow(df), "rows and", ncol(df), "columns\n")
    }, error = function(e) {
      cat("Error", folder, ":", conditionMessage(e), "\n")
    })
  }
}
```

```
## Read: apartments with 7721 rows and 38 columns
## Read: apartments_rent with 3208 rows and 38 columns
## Read: garages_parking with 497 rows and 28 columns
## Read: garages_parking_rent with 307 rows and 27 columns
## Read: house_rent with 310 rows and 40 columns
## Read: houses with 7284 rows and 39 columns
## Read: land with 6322 rows and 27 columns
## Read: land_rent with 104 rows and 27 columns
## Read: premises with 1556 rows and 37 columns
## Read: premises_rent with 2739 rows and 37 columns
```

Ieškome galimai neteisingai įvestų duomenų.

```
# Remove rows with extreme prices (price > 25,000,000 or price < 20)
for (type in names(csv_data_list)) {
  if (!is.null(csv_data_list[[type]]) && "price" %in% colnames(csv_data_list[[type]])) {
    # Count rows with extreme prices
    extreme_high <- sum(csv_data_list[[type]]$price > 50000000, na.rm = TRUE)
    extreme_low <- sum(csv_data_list[[type]]$price < 20, na.rm = TRUE)
    extreme_prices <- extreme_high + extreme_low

    if (extreme_prices > 0) {
      # Store original row count
      original_count <- nrow(csv_data_list[[type]])

      # Remove rows with extreme prices, keep rows where price is within range or NA
      csv_data_list[[type]] <- csv_data_list[[type]][(csv_data_list[[type]]$price >= 20 &
                                                      csv_data_list[[type]]$price <= 25000000) |
                                                      is.na(csv_data_list[[type]]$price), ]

      # Verify how many rows were removed
      new_count <- nrow(csv_data_list[[type]])
      removed_count <- original_count - new_count

      cat(type, ": Removed ", removed_count, " rows with extreme prices (", extreme_high,
          " high, ", extreme_low, " low)\n", sep="")
    } else {
      cat(type, ": No rows with extreme prices\n", sep="")
    }
  } else if (!is.null(csv_data_list[[type]])) {
    cat(type, ": No price column found\n", sep="")
  }
}
```

```
## apartments: No rows with extreme prices
## apartments_rent: No rows with extreme prices
## garages_parking: No rows with extreme prices
## garages_parking_rent: No rows with extreme prices
## house_rent: No rows with extreme prices
## houses: No rows with extreme prices
## land: No rows with extreme prices
## land_rent: Removed 2 rows with extreme prices (0 high, 2 low)
## premises: Removed 65 rows with extreme prices (64 high, 1 low)
## premises_rent: Removed 192 rows with extreme prices (113 high, 33 low)
```

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

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

```r
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
```

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

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

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

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

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

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

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

```
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 3 samples
        sample_size <- min(3, 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, 1 ...
##
##  add_date :
##   From apartments: 2023-11-17, 2024-01-15, 2023-07-07 ...
##
##  additional_equipment :
##   From apartments: , ,  ...
##
##  additional_premises :
##   From apartments: , , Storeroom, Balcony, Terrace, Parking space ...
##
##  area :
##   From apartments: 29,75, 82.0, 27,08 ...
##
##  area_.a. :
##   From land: 97,8 a, 15 a, 120 a ...
##
##  build_year :
##   From apartments: 1966, 1981, 2023 ...
##
##  building_energy_efficiency_class :
##   From apartments: , , A++ ...
##
##  building_type :
##   From apartments: Block house, Block house, Block house ...
##
##  call_forwarding :
##   From apartments: False, False, False ...
##
##  closest_body_of_water :
##   From house_rent: , Pond,  ...
##
##  coordinates :
##   From apartments: 54.91171,23.97343, 54.93039,23.93837, 54.75778,25.25904 ...
##
##  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
##
## BAJORŲ LAJOS - tai miesto namai, kurie ribojasi su 300 ha ploto miško ramuma, Vanaginės geomorfologi
##
## Kviečiame prisijungti prie draugiškos ir jaunatviškos šeimų bendruomenės, jau įsikūrusios I ir II pr
##
## APIE PROJEKTĄ:
## • namas jau pastatytas, tad galite apžiūrėti jį gyvai, be to, pamatyti ankstesnius, jau gyvenamus eta
## • A++ energinė klasė;
## • didelis 1-4 kamb. butų pasirinkimas;
## • balkonai ir terasos jūsų rytiniam kavos puodeliui;
## • vitrininiai langai suteikia butams daugiau šviesos;
## • uždara, aptverta ir jaukiai apšviesta gyvenvietė;
## • liftai;
## • privatus „Lajų" parkas: gyvenvietės viduje įrengtos erdvės vaikams ir suaugusiems;
## • 1 min. kelio iki stotelės. Šalia - parduotuvės, darželiai;
## • Geležinio Vilko gatvė garantuoja greitą susisiekimą;
## • šalia 300 ha Vanaginės geomorfologinis draustinis ir Verkių regioninis parkas.
##
## BAJORŲ LAJOS - gyvenimo pasaka šalia miško.
##
## Numatoma projekto statybų pabaiga: 2024 m. III ketv.
##
## Daugiau informacijos apie projektą ir internetinė registracija:
```

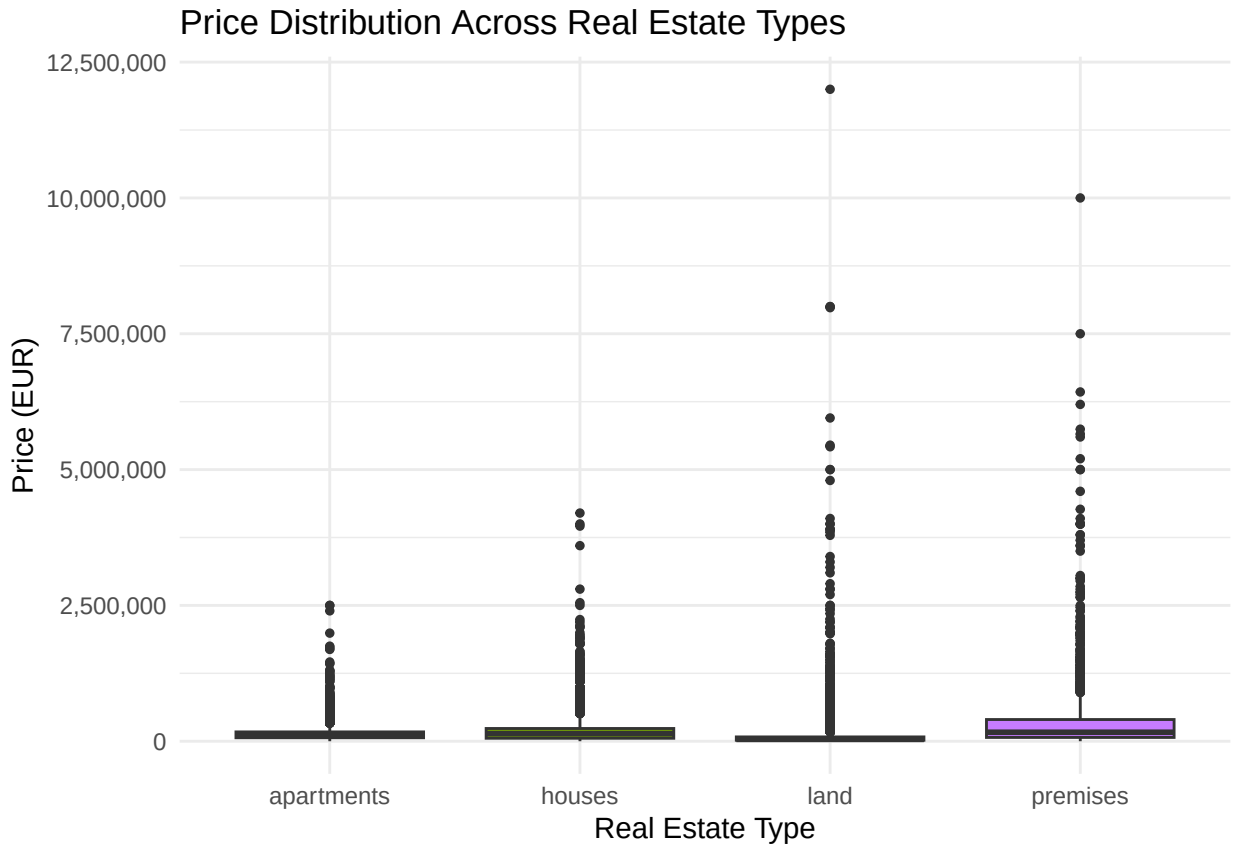
```
## www.bajorulajos.lt
##
## Susisiekite ir pamatykite projektą gyvai:
## +370 682 11050
##
## Projektą vysto: OMBERG GROUP
##
## 2023 m. OMBERG GROUP yra antra daugiausiai sostinės rinkoje naujos statybos butų pardavusi bendrovė.
##
## Bendradarbiaudama su Šiaurės Europos investiciniu fondu, įmonė Vilniuje baigia statyti išskirtinį mo
##
## Išskirtinėje Kauno vietoje, užtikrinančioje bene didžiausią automobilių srautą visoje šalyje, OMBERG
##
## K1-02-02 (ID 15848) ...
##
##  description_tags :
##   From apartments: , , Private entrance ...
##
##  distance_from_body_of_water :
##   From house_rent: , 1,  ...
##
##  equipment :
##   From apartments: Fully equipped, Fully equipped, Partially equipped ...
##
##  features :
##   From garages_parking: Pit, Automatic gates, under the roof,  ...
##
##  flat_no. :
##   From apartments: , 16, 02-02 ...
##
##  floor :
##   From apartments: 2, 6, 2 ...
##
##  heating_system :
##   From apartments: Central, Central, Central thermostat ...
##
##  house_no. :
##   From apartments: , 79, 29 ...
##
##  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, aruodas.lt/1-3342311 ...
##
##  listing_id :
##   From apartments: 3381778, 3395115, 3342311 ...
##
##  lot_no. :
##   From land: , ,  ...
##
##  microdistrict :
##   From apartments: Dainava, Eiguliai, Bajorai ...
##
```

```
##  modified :
##   From apartments: 2024-02-12, 2024-01-15, 2024-02-12 ...
##
##  no._of_floors :
##   From apartments: 5, 9, 7 ...
##
##  number :
##   From garages_parking: , 18, 1A ...
##
##  number_of_rooms :
##   From apartments: 1, 4, 1 ...
##
##  object :
##   From apartments: , ,  ...
##
##  phone_number :
##   From apartments: 37060707730, 37068211050, 37066648547 ...
##
##  plot_area :
##   From house_rent: 1 a, 37 a, 6 a ...
##
##  premises_nr. :
##   From premises: , 13,  ...
##
##  premises_sum :
##   From premises: 4, ,  ...
##
##  price :
##   From apartments: 63000, 98000, 78300 ...
##
##  price_per_month :
##   From apartments_rent: 380 €, 430 €, 350 € ...
##
##  private_seller :
##   From apartments: False, False, False ...
##
##  purpose :
##   From land: Residential land, Residential land, Agricultural, recreational ...
##
##  region :
##   From apartments: Kaunas, Kaunas, Vilnius ...
##
##  reserved :
##   From apartments: False, False, False ...
##
##  security :
##   From apartments: , , Steel doors, Code door lock, Video surveillance ...
##
##  selected :
##   From apartments: 21, 7, 38 ...
##
##  sold_or_rented :
##   From apartments: False, False, False ...
##
```

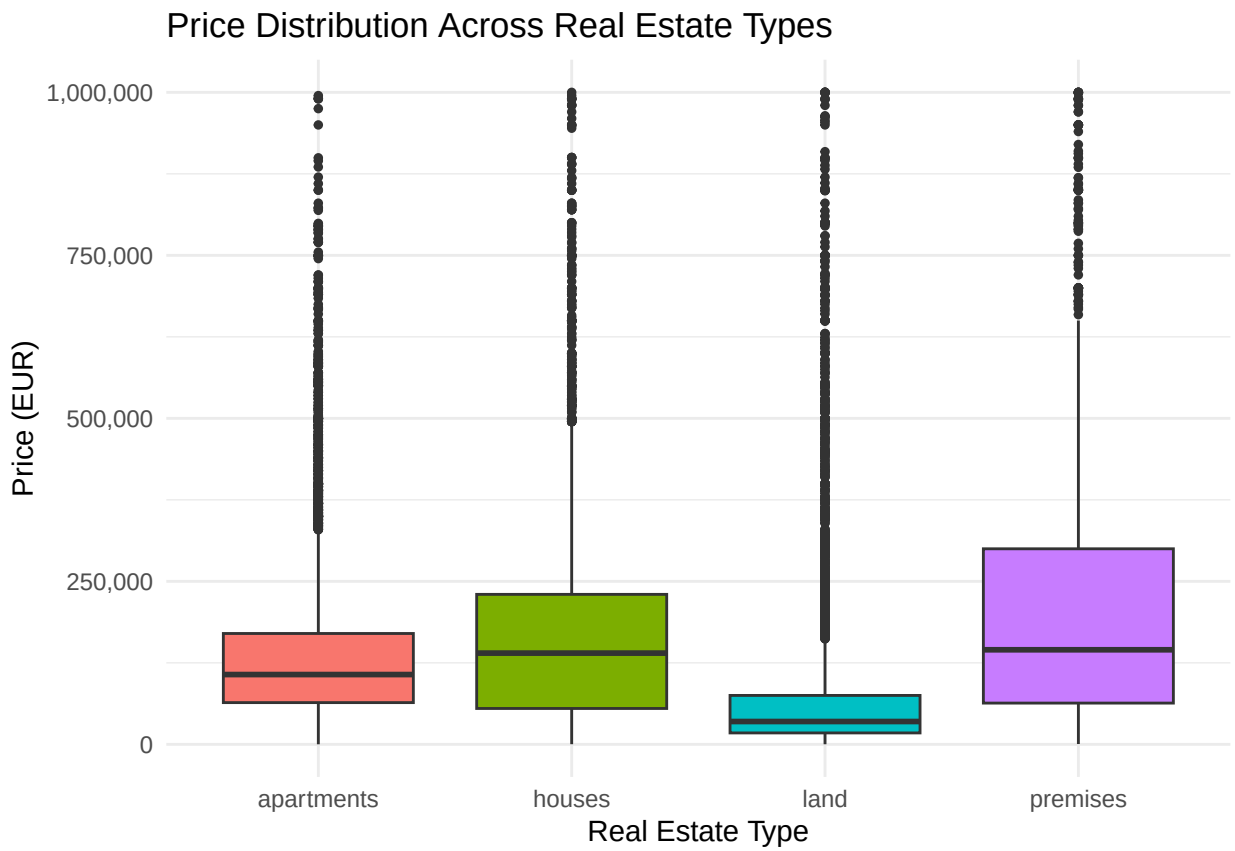
```
##  street :
##   From apartments: Kovo 11-osios g., Šiaurės pr., Bajorų kel. ...
##
##  type :
##   From garages_parking: For sale, For sale, For sale ...
##
##  type_id :
##   From apartments: 1, 1, 1 ...
##
##  unique_item_number :
##   From apartments: , ,  ...
##
##  valid_till :
##   From apartments: , ,  ...
##
##  views_today :
##   From apartments: 0, 0, 1 ...
##
##  views_total :
##   From apartments: 2264, 579, 5087 ...
##
##  water_system :
##   From house_rent: , Private water supply system,  ...
```

```r
library(ggplot2)

# Real estate types to analyze
real_estate_types <- c("apartments", "houses", "land", "premises")

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

price_data <- data.frame()

for (type in real_estate_types) {
  if (type %in% names(csv_data_list) && "price" %in% colnames(csv_data_list[[type]])) {
    # Extract prices and create a data frame
    temp_data <- data.frame(
      price = csv_data_list[[type]]$price,
      type = rep(type, length(csv_data_list[[type]]$price))
    )
    price_data <- rbind(price_data, temp_data)
  }
}

# Create combined box plot if we have data
if (nrow(price_data) > 0) {
  ggplot(price_data, aes(x = type, y = price, fill = type)) +
    geom_boxplot(outlier.size = 1) +
    scale_y_continuous(labels = scales::comma) +
    labs(title = "Price Distribution Across Real Estate Types",
         x = "Real Estate Type",
         y = "Price (EUR)") +
    theme_minimal() +
```

```
    theme(legend.position = "none")
}
```

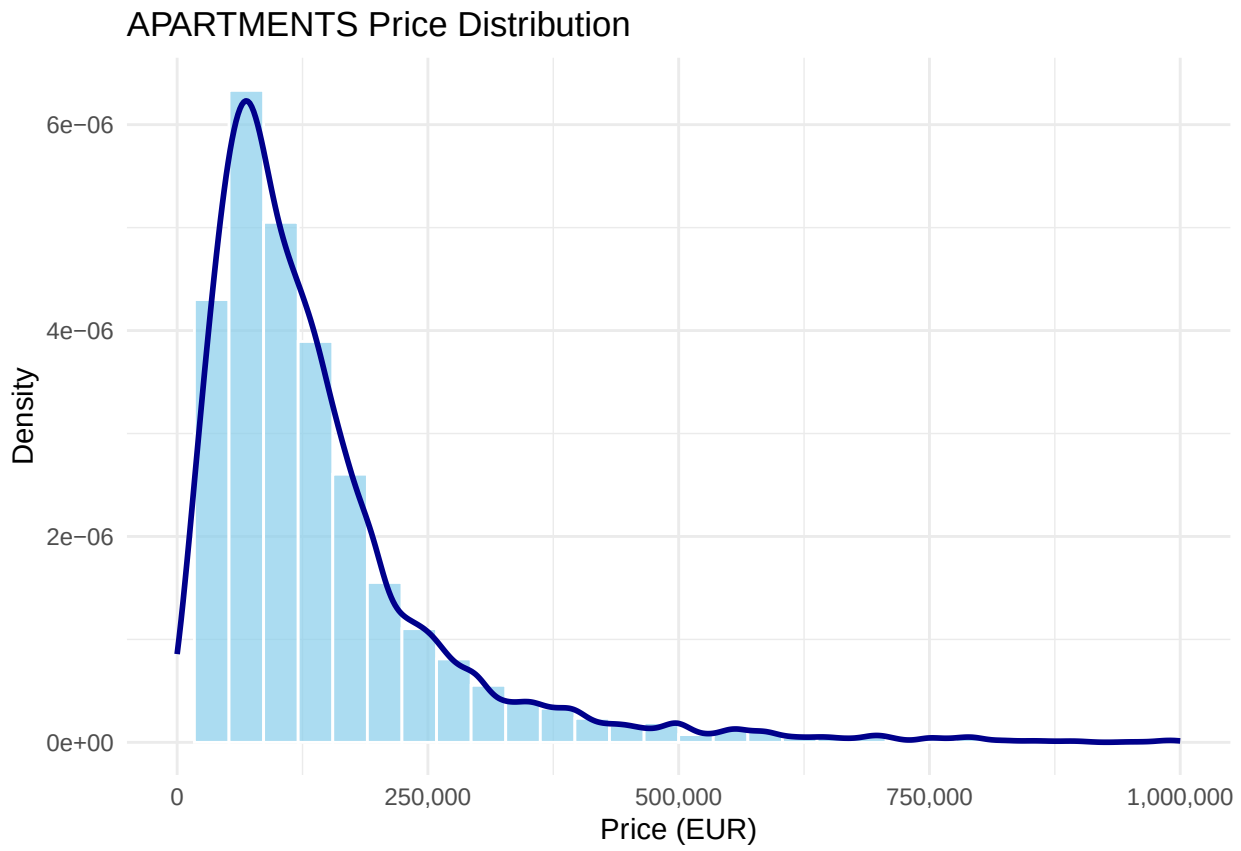


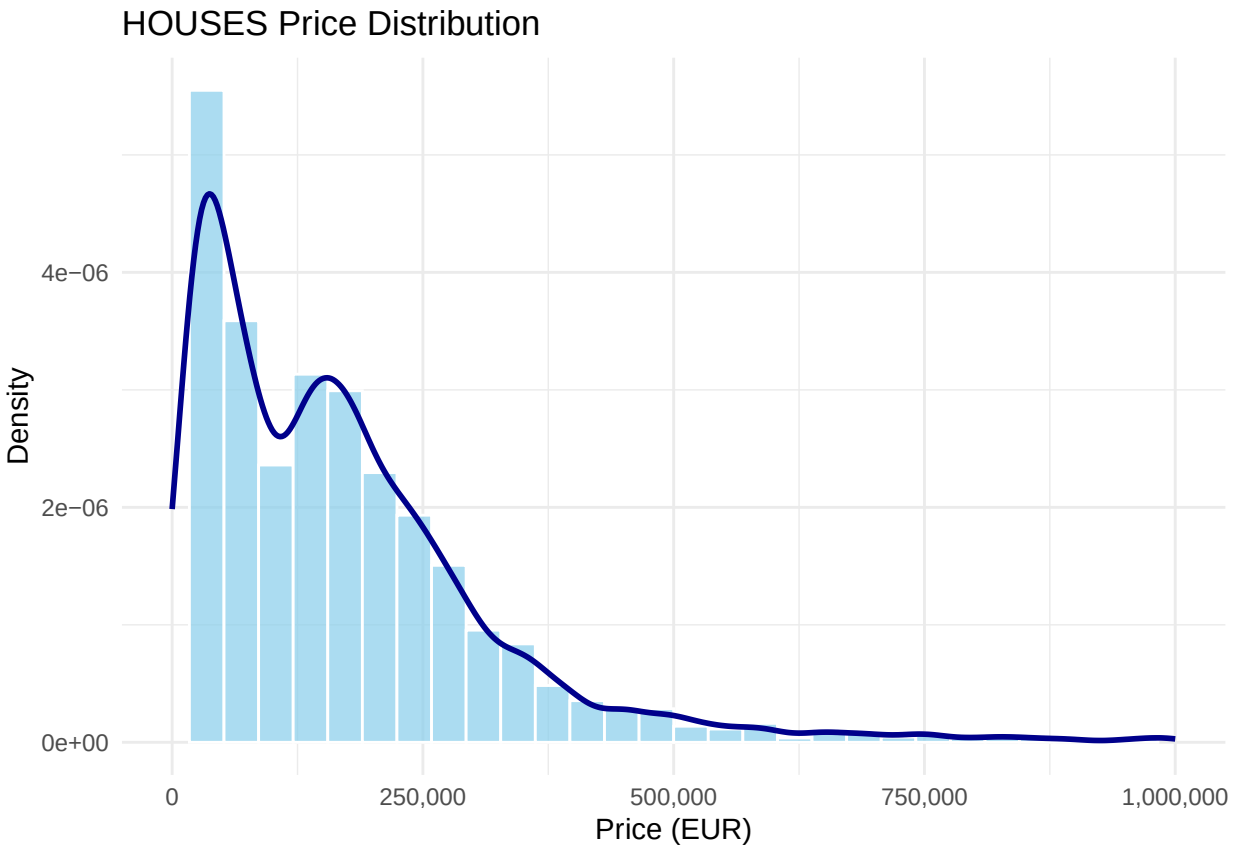

```
# Create combined box plot if we have data
if (nrow(price_data) > 0) {
  ggplot(price_data, aes(x = type, y = price, fill = type)) +
    geom_boxplot(outlier.size = 1) +
    scale_y_continuous(labels = scales::comma,
                       limits = c(NA, 1000000)) + # Set the upper limit
    labs(title = "Price Distribution Across Real Estate Types",
         x = "Real Estate Type",
         y = "Price (EUR)") +
    theme_minimal() +
    theme(legend.position = "none")
}
```

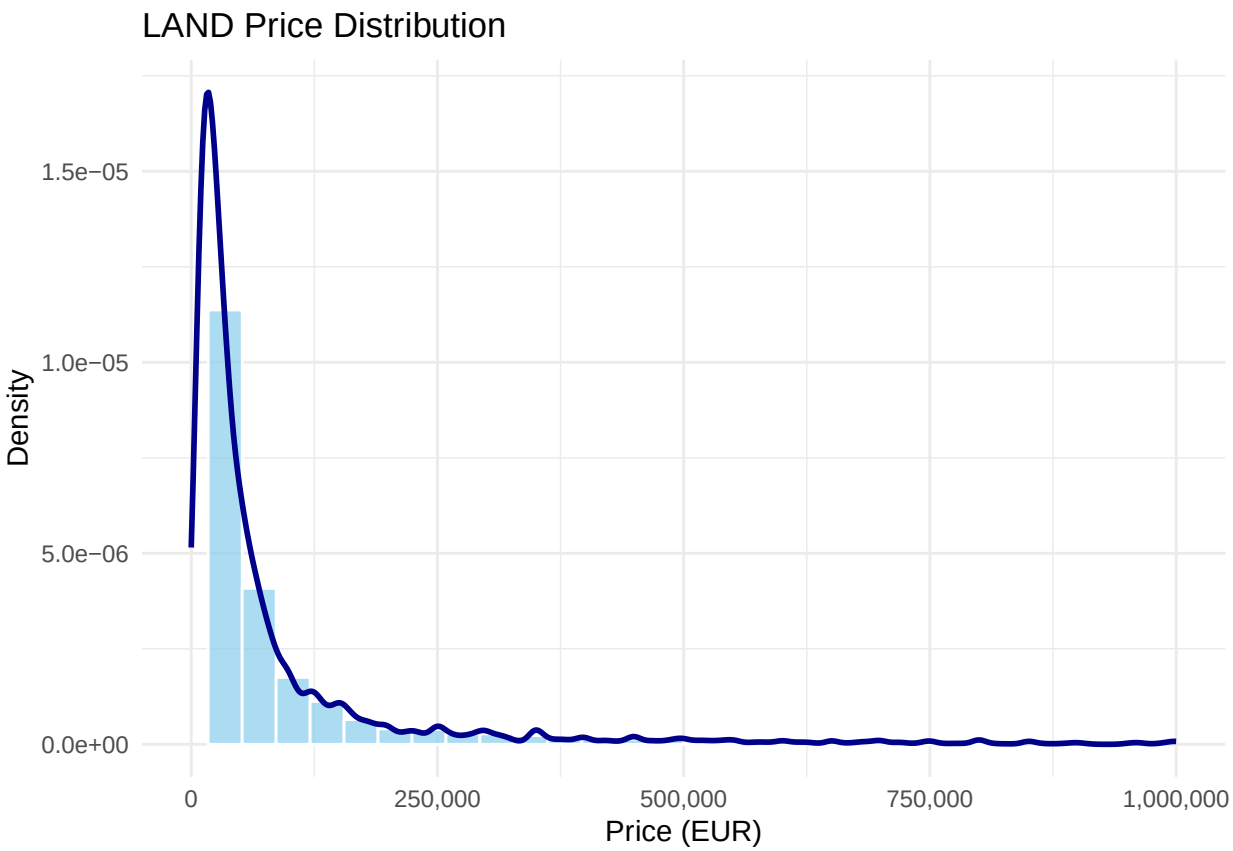


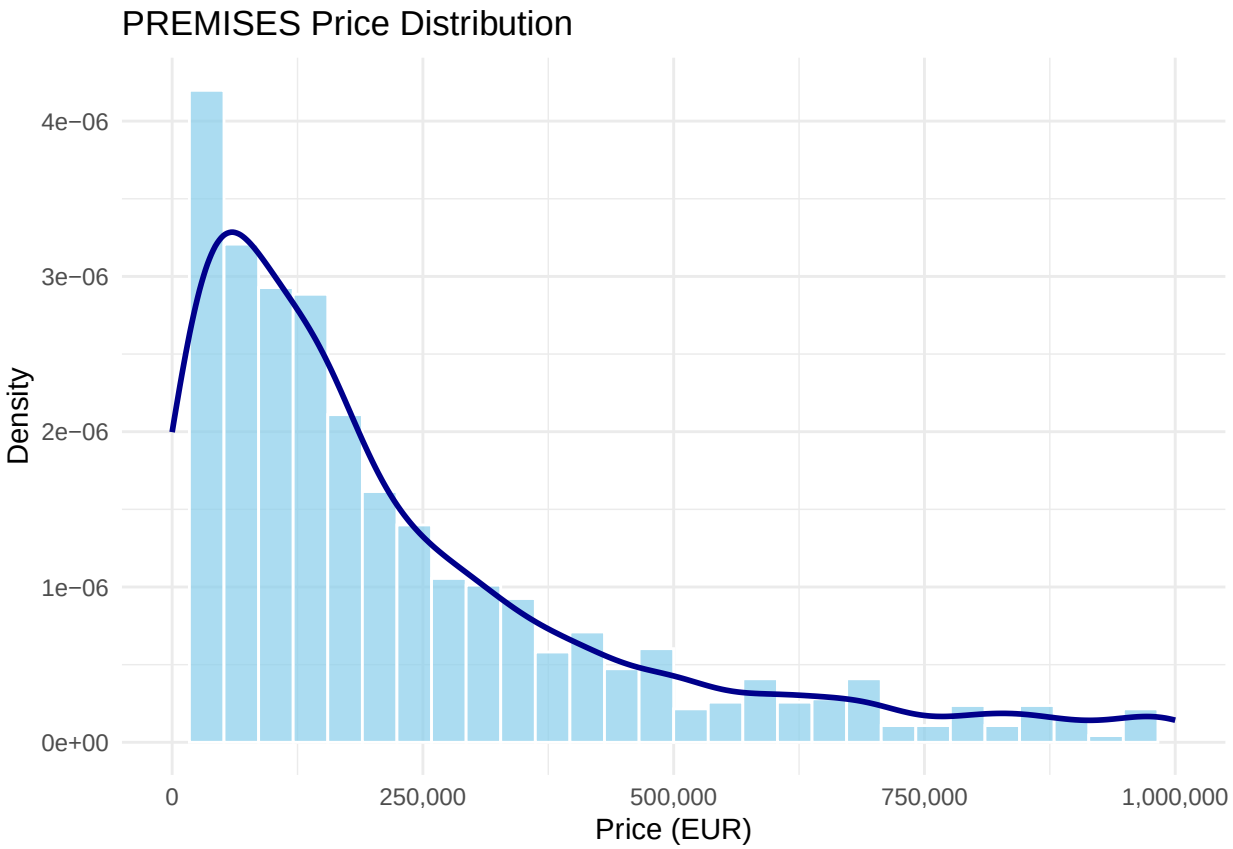

```
# Reset the plot layout
par(mfrow = c(1, 1))

for (type in real_estate_types) {
  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)
  }
```

```
}
```


APARTMENTS Price Distribution
6e−06
4e−06
2e−06
0e+00
Density
0
250,000
500,000
750,000
1,000,000
Price (EUR)


HOUSES Price Distribution
Density
4e−06
2e−06
0e+00
0
250,000
500,000
750,000
1,000,000
Price (EUR)


LAND Price Distribution
Density
1.5e−05
1.0e−05
5.0e−06
0.0e+00
0
250,000
500,000
750,000
1,000,000
Price (EUR)



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

   Tarpkvartilinis plotis (IQR)

   Minimumuas

   Maksimumas

   Diapazonas (Range = max - min)

   Skewness (Asimetrija) – parodo pasiskirstymo simetriškumą

   Kurtosis – parodo, ar duomenys labiau smailūs ar plokšti nei normalus pasiskirstymas

   Medianos absoliutus nuokrypis (MAD) – atsparus vidurkio vietoje naudoti

   Coeficient of Variation (CV) – santykinis dispersijos matas: SD / mean

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)
}
```

```
## Datasets with column price :
##  [1] "apartments"           "apartments_rent"      "garages_parking"
##  [4] "garages_parking_rent" "house_rent"           "houses"
##  [7] "land"                 "land_rent"            "premises"
## [10] "premises_rent"
## Datasets with column price_per_month :
## [1] "apartments_rent" "house_rent"      "premises_rent"
## Datasets with column views_total :
##  [1] "apartments"           "apartments_rent"      "garages_parking"
##  [4] "garages_parking_rent" "house_rent"           "houses"
##  [7] "land"                 "land_rent"            "premises"
## [10] "premises_rent"
## Datasets with column area :
## [1] "apartments"           "apartments_rent"      "garages_parking"
## [4] "garages_parking_rent" "house_rent"           "houses"
## [7] "premises"             "premises_rent"
## Datasets with column area_.a. :
## [1] "land"      "land_rent"
## Datasets with column build_year :
## [1] "apartments"      "apartments_rent" "house_rent"      "houses"
## [5] "premises"        "premises_rent"
## Datasets with column no._of_floors :
## [1] "apartments"      "apartments_rent" "house_rent"      "houses"
## [5] "premises"        "premises_rent"
## Datasets with column floor :
## [1] "apartments"      "apartments_rent" "premises"        "premises_rent"
## Datasets with column number_of_rooms :
## [1] "apartments"      "apartments_rent" "house_rent"      "houses"
## Datasets with column plot_area :
## [1] "house_rent" "houses"
```

```r
# Helper function to calculate the mode
get_mode <- function(v) {
  v <- v[!is.na(v)]
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

# Load required packages
library(e1071)  # For skewness and kurtosis
library(knitr)  # For table formatting

# Function to calculate statistics for a specified variable across multiple datasets
calculate_statistics <- function(data_list, variable_name, target_datasets) {
  # Create an empty data frame to store our statistics
  stats_table <- data.frame(
    Dataset = character(),
    Mean = numeric(),
    Median = numeric(),
    Mode = numeric(),
    SD = numeric(),
    Q1 = numeric(),
    Q2 = numeric(),
    Q3 = numeric(),
    IQR = numeric(),
    Min = numeric(),
    Max = numeric(),
    MAD = numeric(),
    CV = numeric(),
    stringsAsFactors = FALSE
  )

  # Calculate statistics for each dataframe
  for (df_name in target_datasets) {
    if (df_name %in% names(data_list) && variable_name %in% colnames(data_list[[df_name]])) {
      # Show first 10 values (keep this outside the table)
      cat("Sample from the '", variable_name, "' column for ", df_name, ":\n", sep="")
      print(head(data_list[[df_name]][[variable_name]], 10))
      cat("\n")

      # Calculate all statistics
      var_data <- data_list[[df_name]][[variable_name]]
      mean_val <- mean(var_data, na.rm = TRUE)
      median_val <- median(var_data, na.rm = TRUE)
      mode_val <- get_mode(var_data)
      sd_val <- sd(var_data, na.rm = TRUE)
      quartiles <- quantile(var_data, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)
      iqr_val <- IQR(var_data, na.rm = TRUE)
      min_val <- min(var_data, na.rm = TRUE)
      max_val <- max(var_data, na.rm = TRUE)
      mad_val <- mad(var_data, na.rm = TRUE)
      cv_val <- (sd_val / mean_val) * 100

      # Add row to stats table
```

```
      stats_table <- rbind(stats_table, data.frame(
        Dataset = df_name,
        Mean = mean_val,
        Median = median_val,
        Mode = mode_val,
        SD = sd_val,
        Q1 = quartiles[1],
        Q2 = quartiles[2],
        Q3 = quartiles[3],
        IQR = iqr_val,
        Min = min_val,
        Max = max_val,
        MAD = mad_val,
        CV = cv_val
      ))
    } else {
      cat("Dataframe", df_name, "does not exist or does not have a '", variable_name, "' column.\n", sep
    }
  }

  return(stats_table)
}

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

# Calculate statistics for price in sale datasets
price_sale_stats <- calculate_statistics(csv_data_list, "price", sale_datasets)
```

```
## Sample from the 'price' column for apartments:
##  [1]  63000  98000  78300 249000 389000  65000 158000  55000  28000  31900
##
## Sample from the 'price' column for garages_parking:
##  [1] 10000  7000 20000 25000  8000  3100 16500 17000  1420 23000
##
## Sample from the 'price' column for houses:
##  [1] 395000 119000  59500 119800  23000  84000 189900  19800 219900  59000
##
## Sample from the 'price' column for land:
##  [1] 59000 28500 29000  9900 15555  9000 12000  5500 53000 26500
##
## Sample from the 'price' column for premises:
##  [1]  87000   4536 348000 121000  29000 140000 787000 107000 300000 700000
```

```
kable(price_sale_stats,
      caption = "Summary Statistics for Price Across Different Real Estate Types (Sale)",
      digits = 2,  # Round to 2 decimal places
      format.args = list(big.mark = ","))  # Add thousands separator
```

Table 1: Summary Statistics for Price Across Different Real Estate Types (Sale)

	Dataset	Mean	Median	Mode	SD	Q1	Q2	Q3	IQR	Min	Max	MAD	CV
25%	apartments	143,718.13	107,558	125,000	146,129.76	64,000	107,558	172,000	108,000	43	2,500,000	75,105.55	101.68
25%1	garages_parking	19,015.55	15,000	15,000	19,477.64	10,000	15,000	22,499	12,499	500	248,000	8,154.30	102.43
25%2	houses	183,734.43	140,000	35,000	223,884.95	75,000	140,000	235,000	180,000	200	4,200,000	131,951.40	121.85
25%3	land	115,388.68	85,000	25,000	386,437.38	38,000	35,000	79,900	61,900	100	12,000,000	32,617.20	334.90
25%4	premises	413,170.38	165,000	145,000	762,212.47	70,000	165,000	399,850	329,850	490	10,000,000	82,359.80	184.48

```
# Calculate statistics for price in rent datasets
price_rent_stats <- calculate_statistics(csv_data_list, "price", rent_datasets)
```

```
## Sample from the 'price' column for apartments_rent:
##  [1]  380  430  350  540  600  210  350  550  460 1150
##
## Sample from the 'price' column for house_rent:
##  [1]  500  150 1000 2150 1700  900  600 1400 1300 1800
##
## Sample from the 'price' column for premises_rent:
##  [1]     500 2503600    2900  900900     100    1500    2250     255    1300
## [10]     850
```

```
kable(price_rent_stats,
      caption = "Summary Statistics for Price Across Different Real Estate Types (Rent)",
      digits = 2,  # Round to 2 decimal places
      format.args = list(big.mark = ","))  # Add thousands separator
```

Table 2: Summary Statistics for Price Across Different Real Estate Types (Rent)

	Dataset	Mean	Median	Mode	SD	Q1	Q2	Q3	IQR	Min	Max	MAD	CV
25%	apartments_rent	609.95	525	600	1,529.12	380	525	690.0	310.0	20	84,900	222.39	250.70
25%1	house_rent	1,428.76	1,200	1,500	1,327.40	750	1,200	1,500.0	750.0	50	13,000	610.09	92.91
25%2	premises_rent	886,472.97	1,300	1,000	3,213,628.35	700	1,300	5,268.5	4,768.5	22	24,045,000	1,482.60	362.52

```
# Calculate statistics for views_total across all datasets
views_stats <- calculate_statistics(csv_data_list, "views_total", all_datasets)
```

```
## Sample from the 'views_total' column for apartments:
##  [1]  2264   579  5087   694  6571   492  1347   868   637 10670
##
## Sample from the 'views_total' column for apartments_rent:
##  [1]   838  1102   255  1176   268   932  7847  1679  1861 10211
##
## Sample from the 'views_total' column for garages_parking:
##  [1]  679  197  289  248  844  758  806 2139  148  440
##
## Sample from the 'views_total' column for garages_parking_rent:
```

```
##  [1]  191 2099   38   37  320  112  102  122   27  879
##
## Sample from the 'views_total' column for house_rent:
##  [1]   628 24014   316 16307   200   438   204   616   568   212
##
## Sample from the 'views_total' column for houses:
##  [1] 1214 6284  379  374 1822   99  472 1000 1614 3157
##
## Sample from the 'views_total' column for land:
##  [1]   62  436  133  117  429  119   37 1027 4504 1978
##
## Sample from the 'views_total' column for land_rent:
##  [1]  263  162  196 1168   85   91   34   20   75  298
##
## Sample from the 'views_total' column for premises:
##  [1]  214  138  213  672   72 2044  110  143  130  980
##
## Sample from the 'views_total' column for premises_rent:
##  [1]  481 2416  303  430  949  321   33  240 1063   96
```

```
kable(views_stats,
      caption = "Summary Statistics for Total Views Across Different Real Estate Types",
      digits = 2,  # Round to 2 decimal places
      format.args = list(big.mark = ","))  # Add thousands separator
```

Table 3: Summary Statistics for Total Views Across Different Real Estate Types

	Dataset	Mean	Median	Mode	SD	Q1	Q2	Q3	IQR	Min	Max	MAD	CV
25%	apartments	1,572.55	892.0	527	2,244.28	425.00	892.0	1,860.00	1,435.00	0	56,297	852.50	142.72
25%1	apartments_rent	1,806.03	606.5	193	9,702.95	285.75	606.5	1,315.25	1,029.50	2	355,786	593.04	537.25
25%2	garages_parking	726.71	433.0	161	1,017.41	194.00	433.0	876.00	682.00	13	12,209	410.68	140.00
25%3	garages_parking_rent	374.08	173.0	23	728.12	80.00	173.0	404.50	324.50	6	7,521	176.43	194.64
25%4	house_rent	1,274.64	582.5	20	2,331.94	261.75	582.5	1,411.25	1,149.50	20	24,014	587.85	182.95
25%5	houses	2,247.23	1,133.0	412	3,549.20	501.00	1,133.0	2,612.00	2,111.00	2	71,418	1,146.05	157.94
25%6	land	869.22	346.5	76	2,965.45	140.00	346.5	871.75	731.75	1	191,374	377.32	341.16
25%7	land_rent	477.37	255.5	70	559.83	100.25	255.5	619.00	518.75	11	2,658	268.35	117.27
25%8	premises	646.84	310.0	58	1,295.60	132.00	310.0	710.50	578.50	0	21,298	324.69	200.30
25%9	premises_rent	742.19	257.0	42	2,340.73	106.00	257.0	607.00	501.00	1	46,715	271.32	315.38

```
# Calculate statistics for no._of_floors
floors_stats <- calculate_statistics(csv_data_list, "no._of_floors", floors_datasets)
```

```
## Sample from the 'no._of_floors' column for apartments:
##  [1]  5  9  7 18  3  5  3 12  3  1
##
## Sample from the 'no._of_floors' column for apartments_rent:
##  [1] 5 5 9 6 5 5 5 2 5 3
##
## Sample from the 'no._of_floors' column for house_rent:
##  [1] 2 2 2 2 2 2 2 1 3 2
```

```
##
## Sample from the 'no._of_floors' column for houses:
##  [1] 2 2 2 3 1 2 2 1 2 1
##
## Sample from the 'no._of_floors' column for premises:
##  [1]  2  3 NA NA NA  1 NA  3 NA  2
##
## Sample from the 'no._of_floors' column for premises_rent:
##  [1]  2 NA NA  3  1  1 NA NA NA NA
```

```
kable(floors_stats,
      caption = "Summary Statistics for Number of Floors Across Different Real Estate Types",
      digits = 2,
      format.args = list(big.mark = ","))
```

Table 4: Summary Statistics for Number of Floors Across Different Real Estate Types

	Dataset	Mean	Median	Mode	SD	Q1	Q2	Q3	IQR	Min	Max	MAD	CV
25%	apartments	5.07	5	5	3.02	3	5	5	2	1	34	1.48	59.58
25%1	apartments_rent	5.32	5	5	3.00	4	5	6	2	1	34	1.48	56.35
25%2	house_rent	1.82	2	2	0.59	1	2	2	1	1	4	0.00	32.63
25%3	houses	1.58	2	2	0.59	1	2	2	1	1	15	0.00	37.51
25%4	premises	2.36	2	1	1.93	1	2	3	2	1	18	1.48	81.61
25%5	premises_rent	2.82	2	1	2.91	1	2	3	2	1	31	1.48	103.24

```
# Calculate statistics for number_of_rooms
rooms_stats <- calculate_statistics(csv_data_list, "number_of_rooms", rooms_datasets)
```

```
## Sample from the 'number_of_rooms' column for apartments:
##  [1] 1 4 1 3 5 3 1 1 1 1
##
## Sample from the 'number_of_rooms' column for apartments_rent:
##  [1] 2 2 1 2 2 1 1 3 1 5
##
## Sample from the 'number_of_rooms' column for house_rent:
##  [1] NA  5  5  4  6  2  5  4 NA  4
##
## Sample from the 'number_of_rooms' column for houses:
##  [1] NA  4  3  8  3  7  5 NA NA NA
```

```
kable(rooms_stats,
      caption = "Summary Statistics for Number of Rooms Across Different Real Estate Types",
      digits = 2,
      format.args = list(big.mark = ","))
```

Table 5: Summary Statistics for Number of Rooms Across Different Real Estate Types

	Dataset	Mean	Median	Mode	SD	Q1	Q2	Q3	IQR	Min	Max	MAD	CV
25%	apartments	2.38	2	2	0.96	2	2	3	1	1	13	1.48	40.44
25%1	apartments_rent	1.96	2	2	0.84	1	2	2	1	1	10	0.74	43.13
25%2	house_rent	4.17	4	4	1.71	3	4	5	2	1	13	1.48	41.02
25%3	houses	4.20	4	4	2.01	3	4	5	2	1	54	1.48	47.91

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 duomenų.
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.