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.

```r
# 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"
```

```r
# Check if all folders have the file "all_cities_20240214.csv"
all_have_file <- TRUE
folders_with_file <- 0
folders_missing_file <- character(0)

for (folder in folders) {
  file_path <- file.path(data_dir, folder, "all_cities_20240214.csv")
  if (file.exists(file_path)) {
    folders_with_file <- folders_with_file + 1
  } else {
    all_have_file <- FALSE
    folders_missing_file <- c(folders_missing_file, folder)
  }
}
```

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

```r
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.

```r
# Remove rows with extreme prices (price > 50,000,000 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 <= 50000000) |
                                                      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 146 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, descrip
##
## 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, descrip
##
## 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, descrip
##
## 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, descrip
##
## 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, descrip
##
## Dataset from folder: houses
## Number of rows: 7284
## Number of columns: 39
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descrip
##
## Dataset from folder: land
## Number of rows: 6322
## Number of columns: 27
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descrip
##
## 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, descrip
##
## Dataset from folder: premises
## Number of rows: 1491
## Number of columns: 37
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descrip
##
## Dataset from folder: premises_rent
## Number of rows: 2593
## Number of columns: 37
## Column names: listing_id, type_id, price, region, microdistrict, street, coordinates, images, descrip
```

```r
# 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
```

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

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

```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 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 sume
##
## 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: nepriklauso
##
## BAJORŲ LAJOS - tai miesto namai, kurie ribojasi su 300 ha ploto miško ramuma, Vanaginės geomorfologin
##
## Kviečiame prisijungti prie draugiškos ir jaunatviškos šeimų bendruomenės, jau įsikūrusios I ir II pro
##
```

```
## 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 mod
##
## 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 ...
##
```

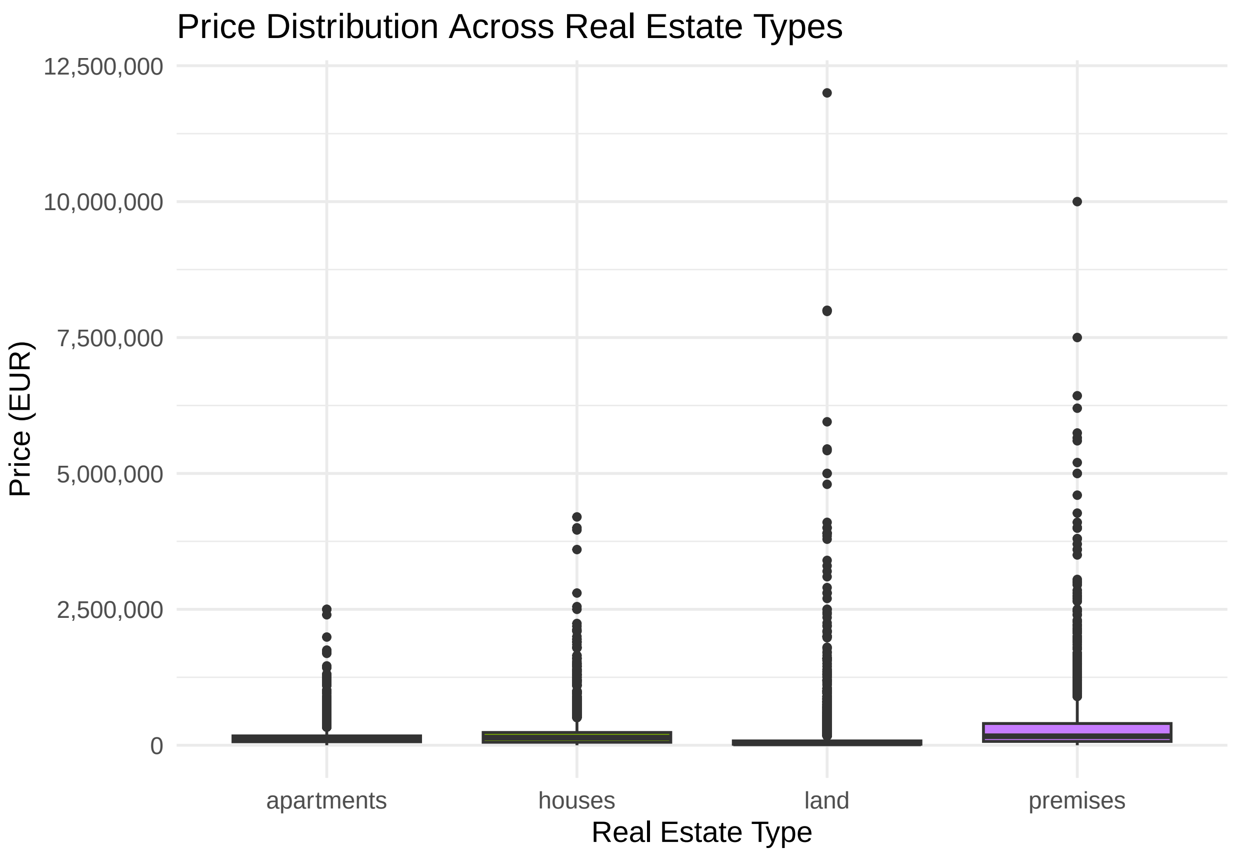
```
##  purpose :
##   From land: Residential land, Residential land, Agricultural, recreational ...
##
##  region :
##   From apartments: Kaunas, Kaunas, Vilnius ...
##
##  reserved :
##   From apartments: False, False, False ...
##
##  security :
##   From apartments: , , Steel doors, Code door lock, 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
# Real estate types to analyze
real_estate_types <- c("apartments", "houses", "land", "premises")

for (type in real_estate_types) {
  cat("\n----------\n", toupper(type), "PRICE SUMMARY\n")

  # Check if this real estate type exists in our list
  if (type %in% names(csv_data_list)) {
    df <- csv_data_list[[type]]

    # Check if price column exists
    if ("price" %in% colnames(df)) {
      # Extract price data
      prices <- df$price

      # Generate summary statistics
      cat("Number of observations:", length(prices), "\n")
      cat("Summary statistics:\n")
      print(summary(prices))

      # Additional statistics
      cat("\nStandard deviation:", sd(prices, na.rm = TRUE), "\n")
    }
  }
}
```

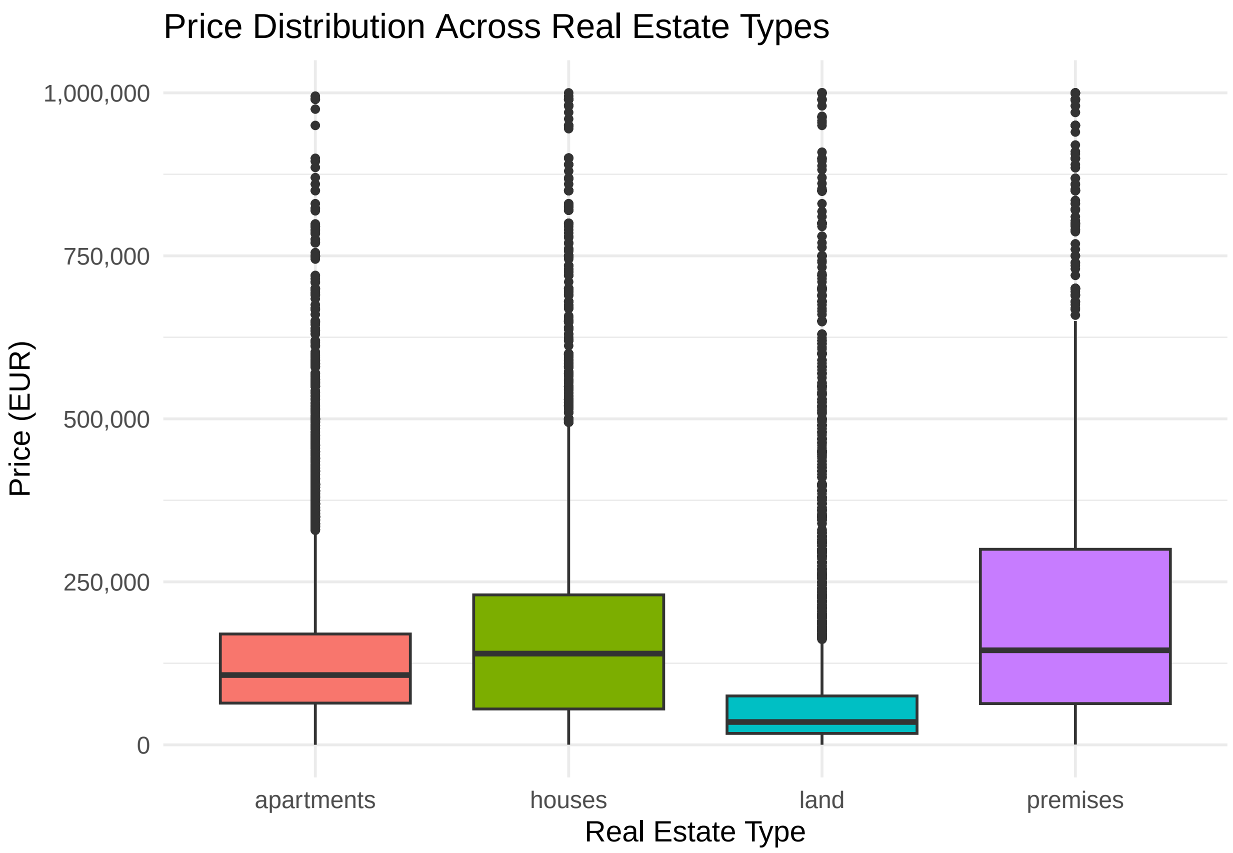
```
##
## ----------
##  APARTMENTS PRICE SUMMARY
## Number of observations: 7721
## Summary statistics:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##      43   64000  107558  143718  172000 2500000
##
## Standard deviation: 146129.7
##
## ----------
##  HOUSES PRICE SUMMARY
## Number of observations: 7284
## Summary statistics:
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     200   55000  140000  183734  235000 4200000
##
## Standard deviation: 223884.9
##
## ----------
##  LAND PRICE SUMMARY
## Number of observations: 6322
## Summary statistics:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      100    18000    35000   115389    79900 12000000
##
## Standard deviation: 386437.4
##
## ----------
##  PREMISES PRICE SUMMARY
## Number of observations: 1491
## Summary statistics:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##      490    70000   165000   413170   399850 10000000
##
## Standard deviation: 762212.4
```

```r
library(ggplot2)
```

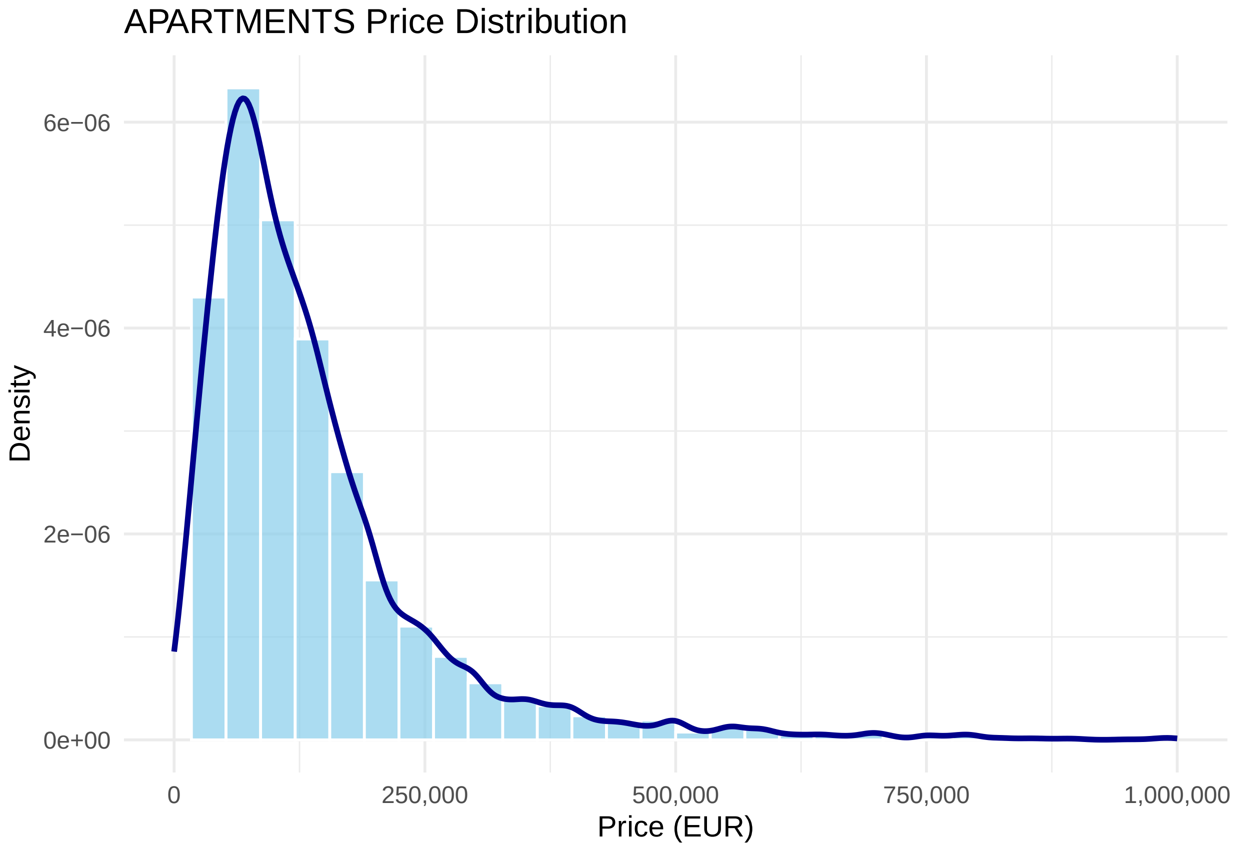
```r
# Real estate types to analyze
real_estate_types <- c("apartments", "houses", "land", "premises")

# 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")
}
```

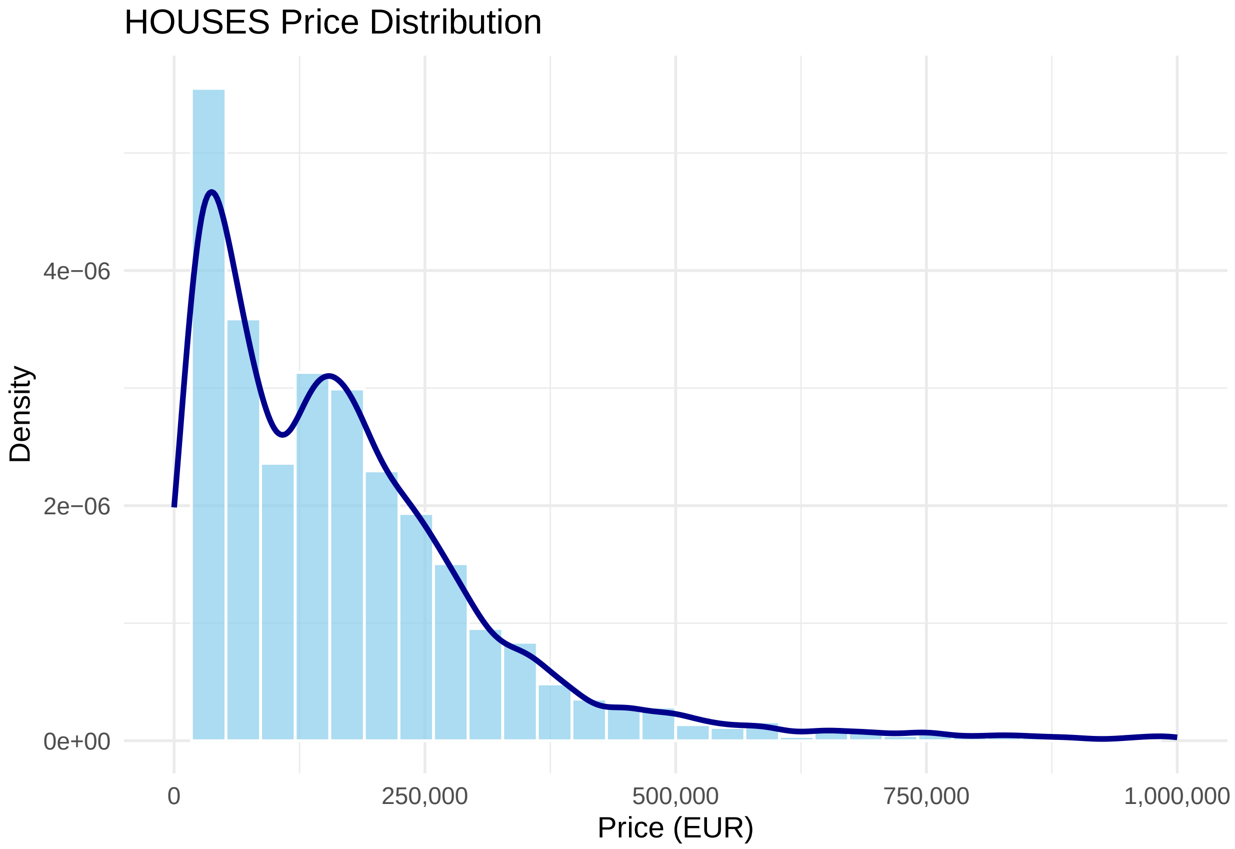


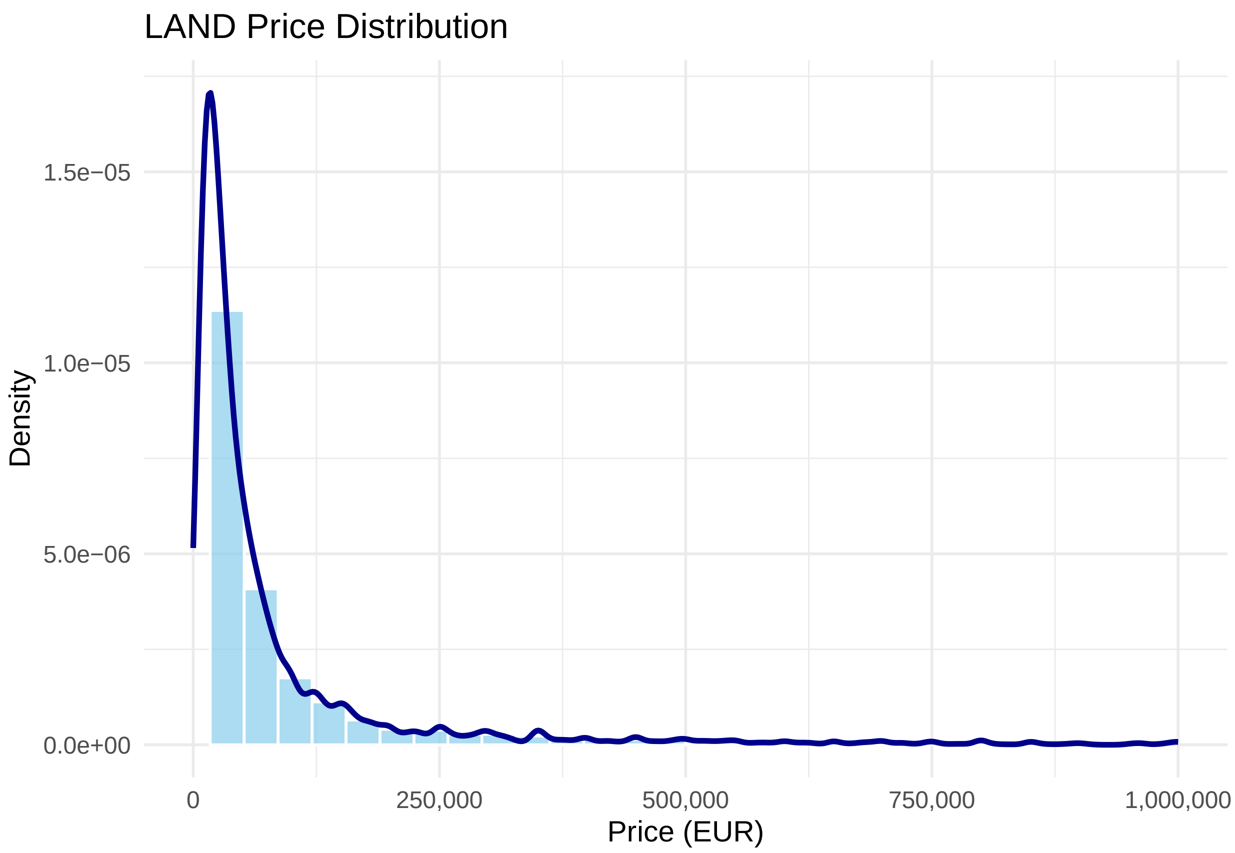
```r
# Create combined box plot if we have data
if (nrow(price_data) > 0) {
  ggplot(price_data, aes(x = type, y = price, fill = type)) +
    geom_boxplot(outlier.size = 1) +
    scale_y_continuous(labels = scales::comma,
                       limits = c(NA, 1000000)) + # Set the upper limit
    labs(title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

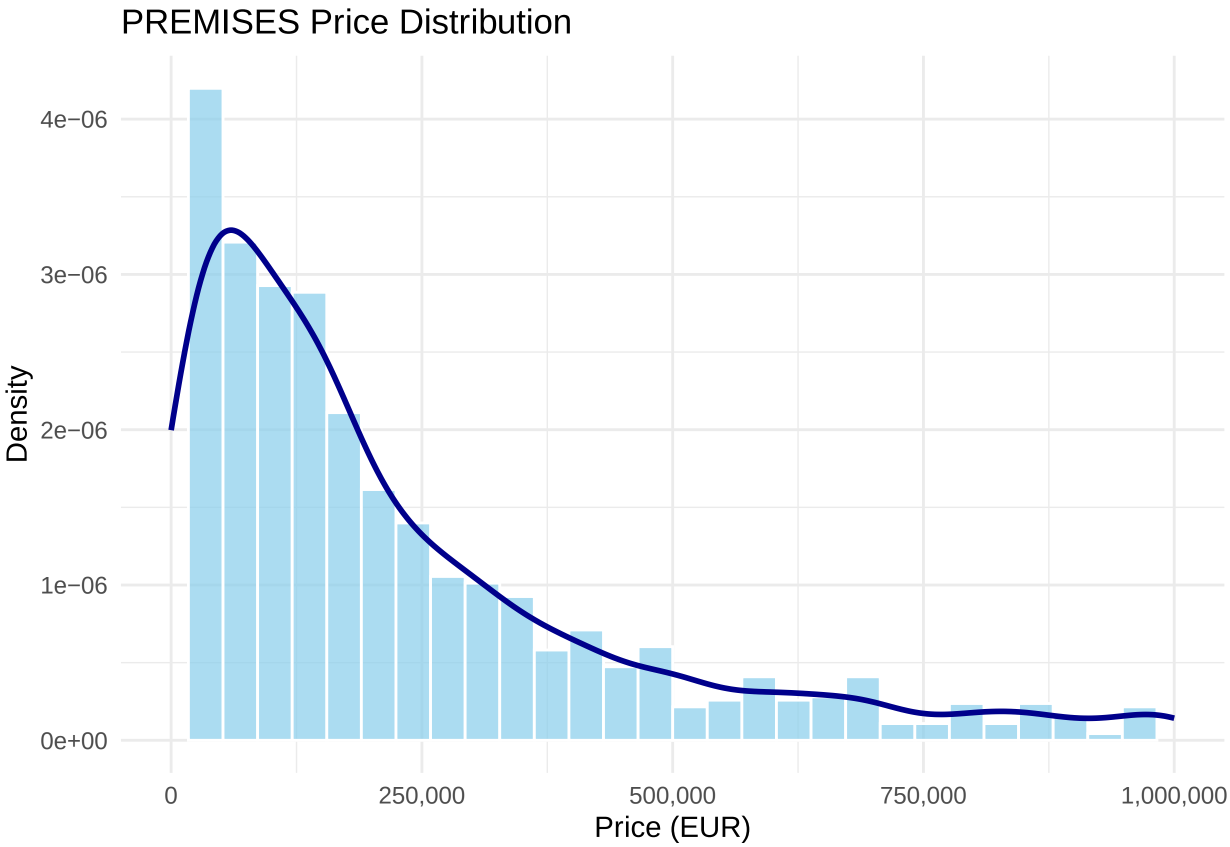


```r
# Reset the plot layout
par(mfrow = c(1, 1))

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
Density
6e−06
4e−06
2e−06
0e+00
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. Vidurkis (Mean), Mediana (Median), Moda (Mode), Dispersija (Variance), Standartinis nuokrypis (Standard Deviation), Kvartiliai (Quartiles), Tarpkvartilinis plotis (Interquartile Range, IQR), Diapazonas (Range, max-min) Kiekybiniai duomenys: kaina ("price"), aukštų skaičius ("no._of_floor"), peržiūrų skaičius, namo, buto 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 ,

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.