Group10 AmesHousing

November 23, 2025

Authors
Mattia Zanin - mattia.zanin@studenti.unipd.it
Matteo Giorgi - matteo.giorgi.1@tudenti.unipd.it
Enrico Zanello - enrico.zanello@studenti.unipd.it
Luca Lo Buono - luca.lobuono@studenti.unipd.it

1   Research hypotheses supported by the literature

The objective of this work is to study which structural, qualitative, and locational characteristics of a residential property significantly affect its market price.

The *Ames Housing Dataset* provides detailed information on dwellings sold in Ames (Iowa) and is an updated version of the Boston Housing Dataset, already widely used in the housing econometrics literature. With its 2930, the *Ames Housing Dataset* offers a larger sample size and a more comprehensive set of features and it is better suited for modern statistical analysis.

Ames dataset includes physical size, construction details, quality ratings, and neighborhood indicators, all variables on which we formulate a series of hypotheses grounded in empirical findings from the real-estate and housing-econometrics literature.

1.1   Fundamentals variables and assumptions

We found that about 80% of the variation in residential sales price can be explained by simply taking into consideration the neighborhood and total square footage (`Total Bsmt SF + Gr Liv Area`) of the dwelling [DC11].

According to our interpretation of the literature published by [Han23] and [ZZS08] plus our empirical reasoning, we consider the variables that significantly affect the sale price of a house are the following.

1.1.1   The overall quality of the house has a strong positive impact on sale price

Several studies show that global quality assessments summarise multiple latent characteristics (materials, workmanship, design), making them among the most informative predictors of house value.

[Abd22] identifies `Overall Qual` as one of the most influential variables in explaining sale price in a multiple regression framework.

Similar evidence is reported in [Han23], where overall quality consistently appears as the strongest determinant in both OLS and regularized regression models.

1.1.2 The above-ground living area (`Gr Liv Area`) is positively associated with the price of a house

The hedonic pricing literature traditionally recognises physical size as a primary contributor to housing value.

> The **hedonic pricing model** is an economic model that explains the price of a good as the sum (or combination) of the values of its attributes. In other words, a complex good is "decomposed" into its components, and the total price reflects the contribution of each of them.
>
> The concept was introduced by Lancaster in 1966 in consumer theory and was first applied to housing prices by Rosen in 1974.

[Abd22] highlights that the main livable area is among the variables with the highest explanatory power. Studies on the Ames dataset by [Ye24] and [Han23] similarly identify `Gr Liv Area` as one of the strongest continuous predictors.

1.1.3 The total basement area (`Total Bsmt SF`) also increases house price, independently from the above-ground area

Basements provide additional functional space and generally correlate with larger, higher-value homes.

Multiple analyses of the Ames dataset [Han23] show that basement size maintains statistical significance even when controlling for other structural variables. [Abd22] also finds that basement-related features contribute substantially to the model fit.

1.1.4 Newer or recently renovated houses (`Year Built`, `Year Remod/Add`) tend to have higher sale prices

The literature on housing depreciation demonstrates that structural aging reduces property value unless offset by renovations.

According to [Abd22], both the construction year and the remodeling year play an important role in predicting sale prices.

Other studies on the Ames dataset reinforce this conclusion, noting that newer homes or homes with extensive remodeling command a price premium.

1.1.5 Higher-quality kitchens and exterior materials positively affect sale price

Studies in real-estate economics show that buyers are particularly sensitive to the quality of kitchens, bathrooms, and exterior finishes, as these elements influence both functionality and aesthetic appeal.

`Kitchen Qual` and `Exter Qual` are repeatedly found to be statistically significant predictors in analyses using the Ames dataset ([Ye24], [Abd22]). Both variables capture qualitative assessments not reflected solely by house size.

1.1.6 Neighborhood characteristics significantly affect sale price, even after controlling for structural attributes.

Location remains one of the most important determinants of housing prices in hedonic models.

The Ames dataset includes a categorical variable (`Neighborhood`) that encodes proximity to schools, income areas, and local amenities.

Previous studies ([DC11], [Ye24]) demonstrate that location dummies remain significant even in fully specified regression models.

1.2 Variables we remove

First, we remove the variables `Pool QC`, `Alley`, `Fence` and `Misc Feature` since they contain too many missing values (`NaN`). The remaining variables would be available and ready for analysis but, as we have shown, including irrelevant variables increases the standard errors, which leads to less powerful significance tests and wider confidence intervals, and therefore to less precise statistical inference.

For this reason we have not considered many of the 82 explanatory variables, mainly because many of them refer to the same part of the house (such as `Bsmt Exposure`, `BsmtFin Type 1`, `BsmtFin Type 2` for the description of the basement), and therefore may be highly correlated and they generate high standard errors, also because we belive that some of them have a negligeable impact on sale prices. In order to obtain a more parsimonious model, with an higher R^2 adjusted and lower AIC and BIC we have decided to exclude these variables from the analysis.

An examlple of such variables are:

- `PID` is just an identification number wich does not provide any information about the house price
- `Lot Front` is the length of street connected to the property, which is not relevant for the price (on one hand a larger front may be better, on the other hand it may mean more noise and traffic)
- `Lot Shape` and `Land Contour` are less relevant than other locational attributes like neighborhood
- `Fireplaces` is the number of fireplaces, which is an outdated feature that does not affect the price significantly
- `FireplaceQu` is the quality of the fireplaces, which is of secondary importance compared to other quality features like kitchen and exterior quality
- `Roof Style` and `Roof Matl` are of secondary importance compared to other features such as quality and size
- `Condition 1` indicates the proximity to various conditions (e.g., arterial roads, railroads), but `Neighborhood` already captures location effects more comprehensively plus there is still the noise factor

2 Description of the Dataset

The analysis is based on the *Ames Housing Dataset*, a well-known real-estate dataset originally compiled by the Assessor's Office of Ames, Iowa. It contains detailed information on residential properties sold between 2006 and 2010.

In its full version, the dataset includes 82 explanatory variables describing structural, qualitative, locational, and functional characteristics of each house.

For the purpose of this assignment, we selected a subset of the variables most commonly used in hedonic price models and supported by the literature.

2.1 General structure of the dataset

Each row of the dataset corresponds to a single residential property, while columns represent the attributes listed above.

Numerical variables are measured either in square feet or in calendar years, whereas qualitative assessments use an ordinal scale. The dataset does not contain missing values for the variables selected for the analysis, and therefore no imputation was required in this step.

The dependent variable of the regression is `SalePrice`, expressed in US dollars and the selected explanatory variables fall into the four following categories.

2.1.1 Size and structural characteristics

These variables capture the physical dimensions of the dwelling:

- `Gr Liv Area`: above-ground living area (in square feet)
- `1st Flr SF`: first-floor area
- `Total Bsmt SF`: total basement area
- `Lot Area`: size of the lot
- `Full Bath`: number of full bathrooms
- `Garage Area`: size of the garage
- `Garage Cars`: garage capacity (number of cars)
- `Garage Yr Blt`: year the garage was built
- `Year Built`: construction year of the house
- `Year Remod/Add`: year of the most recent remodeling
- `Utilities`: type of utilities available

These attributes quantify the amount of usable space and the structural age of the property.

2.1.2 Quality assessments

The dataset includes several ordinal ratings assigned by professional assessors:

- `Overall Qual`: overall material and finish quality
- `Kitchen Qual`: quality of kitchen materials
- `ExterQual`: assessment of exterior materials and workmanship
- `BsmtQual`: quality of basement materials

These variables summarize qualitative features that are not captured by size alone.

2.1.3 Locational and timing information

- `Neighborhood`: categorical variable identifying the physical location within the city of Ames
- `MS Zoning`: categorical variable indicating the general zoning classification such as low-density residential, medium-density residential, or commercial

- `Year Sold`: year the property was sold

This variable incorporates locational amenities and neighbourhood-level characteristics that influence market value.

3 Model Specification and Estimation

The aim of this section is to construct a multiple linear regression model in order to explain the variation in the sale price of residential properties.

The dependent variable is `SalePrice`, while the set of regressors includes the structural, qualitative, and locational characteristics identified in the previous section.

3.1 Model Specification

The regression model is specified as follows:

$$
\begin{aligned}
[\texttt{SalePrice}]_i = {} & \beta_0 + \beta_1\,[\texttt{Gr Liv Area}]_i + \beta_2\,[\texttt{1st Flr SF}]_i + \beta_3\,[\texttt{Total Bsmt SF}]_i \\
& + \beta_4\,[\texttt{Lot Area}]_i + \beta_5\,[\texttt{Full Bath}]_i + \beta_6\,[\texttt{Garage Area}]_i \\
& + \beta_7\,[\texttt{Garage Cars}]_i + \beta_8\,[\texttt{Garage Yr Blt}]_i + \beta_9\,[\texttt{Year Built}]_i \\
& + \beta_{10}\,[\texttt{Year Remod/Add}]_i + \beta_{11}\,[\texttt{Utilities}]_i + \beta_{12}\,[\texttt{Overall Qual}]_i \\
& + \beta_{13}\,[\texttt{Kitchen Qual}]_i + \beta_{14}\,[\texttt{Exter Qual}]_i + \beta_{15}\,[\texttt{Bsmt Qual}]_i \\
& + \beta_{15}\,[\texttt{Neighborhood}]_i + \beta_{16}\,[\texttt{MS Zoning}]_i + \beta_{16}\,[\texttt{Year Sold}]_i + \varepsilon_i
\end{aligned}
$$

where:

- β_0 is the intercept
- β's denote the coefficients associated with each regressor
- ε_i is the error term capturing unobserved factors
- the `Neighborhood` variable is treated as a categorical factor and encoded through dummy variables

3.2 Model Assumptions

Our multiple linear regression model is estimated under the following assumptions:

- linearity of the *conditional mean*

$$
\mathbb{E}[\varepsilon_i \mid X_i] = 0 \iff \mathbb{E}[Y_i \mid X_i] = \beta_0 + \beta_1 X_{i1} + \cdots + \beta_p X_{ip}
$$

This assumption implies that all systematic variation in the dependent variable is captured by the regressors.

- *independent and identically distributed* observations

$$
(Y_i, X_i) \perp\!\!\!\perp (Y_j, X_j) \quad \text{for } i \neq j
$$

and all observations are drawn from the same population distribution. This corresponds to the standard *i.i.d.* sampling framework.

- finite *fourth moments*

$$\mathbb{E}[X_{ik}^4] < \infty \qquad \text{and} \qquad \mathbb{E}[\varepsilon_i^4] < \infty$$

This condition ensures the applicability of asymptotic results and prevents extreme observations from dominating the estimation.

3.3 Python code

Here we list the full code used for the analysis with all comments and explanations.

```python
[1]: import pandas as pd
     import statsmodels.api as sm

     # Read AmesHousing.csv into the DataFrame df:
     # each row represents a house
     # each column represents a variable
     df = pd.read_csv("AmesHousing.csv")
```

```python
[ ]: # SalePricee is the target variable y
     # (the dependent variable)
     y = df["SalePrice"]

     # regressors is a list of column names chosen from the 82 variables available
     # in the dataset we want to use as regressors in the model
     regressors = [
         "Gr Liv Area",
         "1st Flr SF",
         "Total Bsmt SF",
         "Lot Area",
         "Full Bath",
         "Garage Area",
         "Garage Cars",
         "Garage Yr Blt",
         "Year Built",
         "Year Remod/Add",
         "Overall Qual",
         "Kitchen Qual",
         "Exter Qual",
         "Bsmt Qual",
         "Neighborhood",
         "MS Zoning",
         "Utilities",
     ]
```

```python
# X is a new DataFrame that contains only the columns listed in regressors
X = df[regressors]

# Create dummy variables for the categorical variables:
# get_dummies converts categorical variables (strings) into a set of binary
 ↪dummy columns
# "Neighborhood" -> ["Neighborhood_Blueste", "Neighborhood_BrDale", ...]
# "House Style"  -> ["House Style_1.5Unf", "House Style_1Story", ...]
# "Kitchen Qual" -> ["Kitchen Qual_Fa", "Kitchen Qual_Gd", ...]
# ...
# For each categorical variable, drop_first=True sorts the dummy columns
# alphanumerically, than drops the first one to avoid the dummy variable
# trap (perfect multicollinearity among the dummies). Numerical columns are
 ↪left unchanged
X = pd.get_dummies(X, drop_first=True)

# Force all columns of X and y to be numeric
# errors="coerce" converts anything that cannot be converted to a number
# (weird string, empty space, symbol) into NaN
X = X.apply(pd.to_numeric, errors="coerce")
y = pd.to_numeric(y, errors="coerce")

# Combine the y column and all X columns into a single DataFrame
# Use dropna() to remove all rows containing at least one NaN in any column.
# e.g., if "SalePrice", "Gr Liv Area", one of the dummy columns is missing,
# or a string was coerced to NaN
# | y  | x1 | x2 | ... |
# |----|----|----|-----|
# | ** | ** | ** | ... |
# | ** | ** | ** | ... |
data = pd.concat([y, X], axis=1).dropna()

# Take the SalePrice column (y) from the cleaned DataFrame and convert it to
# a NumPy float array -> y_clean: n*1 vector
#                     -> X_clean: n*k matrix
# Take all remaining columns and convert them to a NumPy float array
y_clean = data["SalePrice"].to_numpy(dtype=float)
X_clean = data.drop(columns=["SalePrice"]).to_numpy(dtype=float)

# Add a column of ones to the matrix (the intercept b_0)
X_clean = sm.add_constant(X_clean)

# Create the OLS model object and call fit() to estimate the coefficients (b^)
# by minimizing the sum of squared residuals: given y = X*b + e,
# it finds the b^ that make the residuals as small as possible
model = sm.OLS(y_clean, X_clean).fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.855
Model:                            OLS   Adj. R-squared:                  0.852
Method:                 Least Squares   F-statistic:                     280.9
Date:                Sat, 22 Nov 2025   Prob (F-statistic):               0.00
Time:                        22:26:35   Log-Likelihood:                 -32513.
No. Observations:                2770   AIC:                         6.514e+04
Df Residuals:                    2712   BIC:                         6.549e+04
Df Model:                          57
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -8.876e+05   1.32e+05     -6.724      0.000   -1.15e+06   -6.29e+05
x1             44.3995      2.027     21.906      0.000      40.425      48.374
x2             -3.9192      3.104     -1.263      0.207     -10.006       2.167
x3             23.4093      2.665      8.784      0.000      18.183      28.635
x4              0.6152      0.085      7.212      0.000       0.448       0.782
x5          -3756.4845   1658.574     -2.265      0.024   -7008.682    -504.287
x6             16.5662      6.695      2.475      0.013       3.439      29.693
x7           7228.4564   1931.315      3.743      0.000    3441.459     1.1e+04
x8            -39.5767     47.771     -0.828      0.407    -133.249      54.095
x9            247.2766     57.053      4.334      0.000     135.404     359.149
x10           259.1812     44.615      5.809      0.000     171.698     346.664
x11           1.261e+04    820.520     15.370      0.000     1.1e+04    1.42e+04
x12          -3.551e+04   5801.478     -6.120      0.000    -4.69e+04   -2.41e+04
x13          -2.923e+04   3179.782     -9.192      0.000    -3.55e+04    -2.3e+04
x14          -5.268e+04    3.14e+04     -1.678      0.094    -1.14e+05    8889.597
x15          -3.553e+04   3552.403    -10.001      0.000    -4.25e+04   -2.86e+04
x16           -3.79e+04   8692.842     -4.360      0.000    -5.49e+04   -2.09e+04
x17          -2.849e+04   4109.779     -6.931      0.000    -3.65e+04   -2.04e+04
x18          -2.825e+04   4651.880     -6.073      0.000    -3.74e+04   -1.91e+04
x19          -2.125e+04   4794.607     -4.433      0.000    -3.07e+04   -1.19e+04
x20          -2.251e+04   2492.245     -9.033      0.000    -2.74e+04   -1.76e+04
x21         -2909.7269    2.36e+04     -0.123      0.902    -4.93e+04    4.35e+04
x22           -1.96e+04   2978.602     -6.581      0.000    -2.54e+04   -1.38e+04
x23          2288.6955    1.19e+04      0.192      0.848    -2.11e+04    2.56e+04
x24         -7326.1849   9249.664     -0.792      0.428    -2.55e+04    1.08e+04
x25           1.098e+04   7863.407      1.397      0.163    -4436.016    2.64e+04
x26           2.192e+04   7928.231      2.765      0.006    6374.661    3.75e+04
x27          8528.3242   6233.377      1.368      0.171    -3694.325    2.08e+04
x28           3.186e+04   7146.519      4.459      0.000     1.79e+04    4.59e+04
x29         -8111.6946   6852.156     -1.184      0.237    -2.15e+04    5324.281
x30          4281.1987   6486.531      0.660      0.509    -8437.844     1.7e+04
x31           1.275e+04   1.25e+04      1.022      0.307    -1.17e+04    3.72e+04
x32           1.129e+05   2.28e+04      4.962      0.000     6.83e+04    1.57e+05
x33          7997.7135   8736.543      0.915      0.360    -9133.242    2.51e+04
```

```
x34        -3216.6788      3.2e+04      -0.101      0.920     -6.6e+04      5.95e+04
x35         2471.7518     9517.391       0.260      0.795    -1.62e+04      2.11e+04
x36         2205.6500     6858.340       0.322      0.748    -1.12e+04      1.57e+04
x37         3331.9295     6567.072       0.507      0.612    -9545.042      1.62e+04
x38        -5548.4113     8930.430      -0.621      0.534    -2.31e+04       1.2e+04
x39         3487.8154     6725.315       0.519      0.604    -9699.446      1.67e+04
x40         5.949e+04     7149.219       8.321      0.000     4.55e+04      7.35e+04
x41         3.104e+04     6570.613       4.724      0.000     1.82e+04      4.39e+04
x42         1234.4781     7905.291       0.156      0.876    -1.43e+04      1.67e+04
x43          868.9178     8469.032       0.103      0.918    -1.57e+04      1.75e+04
x44         3828.3917     6851.807       0.559      0.576    -9606.899      1.73e+04
x45          732.8101     6622.674       0.111      0.912    -1.23e+04      1.37e+04
x46         1.747e+04     7563.024       2.310      0.021     2640.863      3.23e+04
x47         5.065e+04     7397.165       6.847      0.000     3.61e+04      6.52e+04
x48         1.653e+04     7004.926       2.360      0.018     2793.712      3.03e+04
x49         2.556e+04     8697.728       2.939      0.003     8509.290      4.26e+04
x50         7187.1732      2.39e+04       0.301      0.764    -3.97e+04       5.4e+04
x51         2.269e+04      2.33e+04       0.976      0.329    -2.29e+04      6.83e+04
x52         1.445e+04      3.35e+04       0.431      0.667    -5.13e+04      8.02e+04
x53         1.963e+04      2.36e+04       0.831      0.406    -2.67e+04      6.59e+04
x54         2.938e+04      2.26e+04       1.299      0.194      -1.5e+04      7.37e+04
x55         1.912e+04      2.26e+04       0.845      0.398    -2.52e+04      6.35e+04
x56        -3.536e+04       3.1e+04      -1.141      0.254    -9.61e+04      2.54e+04
x57        -2.675e+04      2.53e+04      -1.058      0.290    -7.63e+04      2.28e+04
==============================================================================
Omnibus:                     1640.604   Durbin-Watson:                   1.703
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           230338.424
Skew:                          -1.805   Prob(JB):                         0.00
Kurtosis:                      47.527   Cond. No.                     3.08e+06
==============================================================================
```

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.08e+06. This might indicate that there are strong multicollinearity or other numerical problems.

3.4 Interpretation of the regression output

The OLS regression is estimated on 2,770 observations.
The dependent variable is SalePrice, while the regressors include size variables, quality indicators, age/renovation variables and a set of dummies for neighbourhood, zoning and utilities.

3.4.1 Overall fit of the model

The model displays a relatively high goodness of fit:

- R-squared $= 0.855$

- `Adjusted R-squared` $= 0.852$

This means that about 85% of the variation in house prices in the sample is explained by the regressors included in the model.

The `F-statistic` is 280.9 with a p-value essentially equal to zero, so the null hypothesis that all slope coefficients are jointly equal to zero is strongly rejected. Taken together, the explanatory variables have a statistically significant impact on `SalePrice`.

3.4.2 Interpretation of the main coefficients

The first group of coefficients (`x1`–`x11`) corresponds to the "core" numeric regressors (size, age, quality):

- `x1 ≈ 44.4` (`Gr Liv Area`) is positive and highly significant. Interpreting it as the coefficient on above-ground living area, it suggests that, ceteris paribus, an additional square foot of living area increases the sale price by about 44 dollars.
- `x3 ≈ 23.4` (`Total Bsmt SF`) is also positive and highly significant, indicating that extra basement area is rewarded in the market (around 23 dollars per square foot)
- `x4 ≈ 0.62` (`Lot Area`) is positive and significant: larger lots are associated with higher prices, although the marginal effect per square foot is much smaller than for interior living space or basement area
- `x5 ≈ –3,756` (`Full Bath`) is negative and statistically significant at the 5% level. This is somewhat counter-intuitive, but it can be explained by multicollinearity: once we condition on total size and quality, adding one more full bathroom at fixed size may capture houses with a different layout rather than genuinely higher quality
- `x6 ≈ 16.6` (`Garage Area`) and `x7 ≈ 7,228` (`Garage Cars`) are both positive and significant, meaning that larger garages and the capacity to park more cars are rewarded by the market
- `x8` (`Garage Yr Blt`) is not statistically significant (p ≈ 0.41), suggesting that, after controlling for the other variables, the construction year of the garage does not have a clear marginal effect
- `x9 ≈ 247.3` (`Year Built`) and `x10 ≈ 259.2` (`Year Remod/Add`) are positive and highly significant: newer houses and more recently renovated properties tend to sell at higher prices, in line with the idea that buyers pay a premium for newer structures and recent improvements
- `x11 ≈ 12,610` (`Overall Qual`) is positive and extremely significant. A one-point increase in the overall quality rating is associated with roughly 12 or 13 thousand dollars more in sale price, everything else being equal. **This confirms the strong role of quality emphasised in the hedonic pricing literature**

The next coefficients (`x12`, `x13`, `x15`–`x20`, `x22`, …, `x57`) correspond to dummy variables created from `Kitchen Qual`, `Exter Qual` and `Bsmt Qual`. Most of these are negative and highly significant, because the baseline category is the highest quality level, and the dummies measure the price discount associated with having a lower quality relative to that baseline.
In other words, houses with worse kitchen, exterior or basement quality sell at systematically lower prices.

The block of coefficients starting around `x28`–`x41` corresponds mostly to `Neighborhood` dummies. Some neighbourhoods show large positive and significant coefficients (for example, `x32`, `x40`, `x41`, `x47`, `x48`, `x49`), indicating that these areas command sizeable price premia relative to the omitted reference neighbourhood.

Other neighbourhood coefficients are not statistically significant, suggesting that, after controlling for structural and quality characteristics, their effect on price is not clearly distinguishable from zero.

Finally, the last coefficients (approximately `x50`–`x57`) represent dummies for `MS Zoning` and `Utilities`.
Most of them are not statistically significant (p-values well above 0.05), which indicates that, in this specification, zoning and utilities categories do not add much explanatory power once size, quality and neighbourhood are already in the model.

3.5 Covariance matrix

```python
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Select only the relevant numeric columns
num_vars = [
    "Gr Liv Area",
    "1st Flr SF",
    "Total Bsmt SF",
    "Lot Area",
    "Full Bath",
    "Garage Area",
    "Garage Cars",
    "Garage Yr Blt",
    "Year Built",
    "Year Remod/Add",
]

# Compute the correlation matrix
corr = df[num_vars].corr()

# Plot the heatmap with annotations, ticks and labels
plt.figure(figsize=(8, 6))
sns.heatmap(corr, annot=True, fmt=".2f", cmap="coolwarm")
plt.xticks(np.arange(len(num_vars)), num_vars, rotation=90)
plt.yticks(np.arange(len(num_vars)), num_vars)
plt.title("Correlation matrix")
plt.tight_layout()
plt.show()
```

Correlation matrix
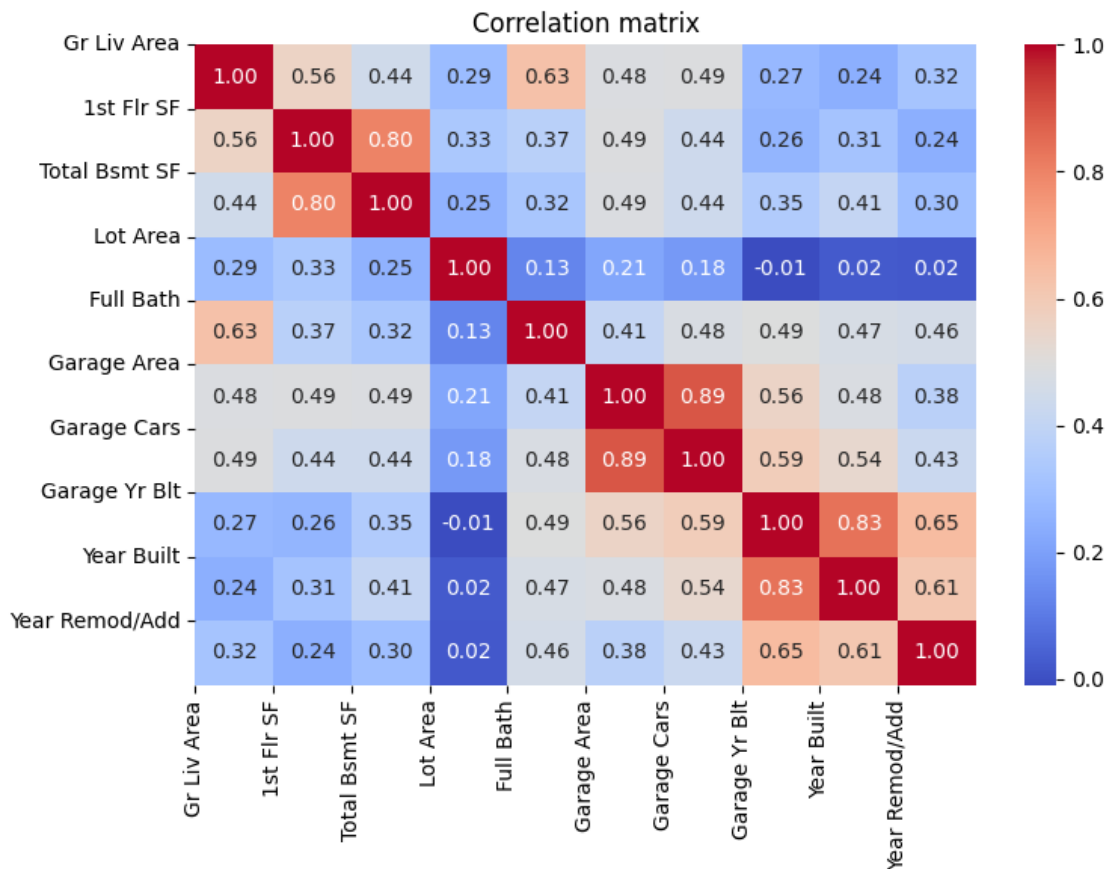
4 Diagnostic analysis

4.1 Perfect multicollinearity

Before proceeding with the regression diagnostics, we checked whether any exact linear relationships existed among the regressors. In the *Ames Housing Dataset*, the variable `Gr Liv Area` (above-ground living area) we suspect might be, by construction, equal to the sum of other area-related variables:

$$\texttt{Gr Liv Area = 1st Flr SF + 2nd Flr SF + Low Qual Fin SF}$$

If this relation holds exactly, including all four variables simultaneously in the regression would generate perfect multicollinearity, making the OLS estimator not uniquely defined. This means that the design matrix would be rank-deficient, and therefore the inverse of $X'X$ required for the OLS formula could not be computed. Let's verifie this identity in the data by computing the difference:

$$d_i = [\texttt{Gr Liv Area}]_i - \left([\texttt{1st Flr SF}]_i + [\texttt{2nd Flr SF}]_i + [\texttt{Low Qual Fin SF}]_i\right) \quad \forall i$$

12

```
[ ]:   # Check the following identity for multicollinearity:
       # Gr Liv Area = 1st Flr SF + 2nd Flr SF + Low Qual Fin SF
       df["GrLiv_diff"] = df["Gr Liv Area"] - (
           df["1st Flr SF"] + df["2nd Flr SF"] + df["Low Qual Fin SF"]
       )

       # Use pandas.Series.describe() to get various statistics about the differences␣
        ↪(mean, std, min, max, quartiles)
       # We use absolute values to check the maximum absolute difference to calculate␣
        ↪the largest deviation from zero
       print(df["GrLiv_diff"].describe())
       print("Max absolute difference:", df["GrLiv_diff"].abs().max())

       # Check if all values are equal up to a numerical tolerance
       # using numpy.allclose() that compares two arrays element-wise
       print(
           "All equal up to numerical tolerance:",
           np.allclose(
               df["Gr Liv Area"], df["1st Flr SF"] + df["2nd Flr SF"] + df["Low Qual␣
        ↪Fin SF"]
           ),
       )
```

```
count    2930.0
mean        0.0
std         0.0
min         0.0
25%         0.0
50%         0.0
75%         0.0
max         0.0
Name: GrLiv_diff, dtype: float64
Max absolute difference: 0
All equal up to numerical tolerance: True
```

The differences were identically equal to zero (up to numerical precision): this implies that including all four variables simultaneously as regressors would lead to perfect multicollinearity and a singular design matrix.

To avoid this problem and to keep the model parsimonious, we retain only `Gr Liv Area` in the regression and exclude `1st Flr SF`, `2nd Flr SF`, and `Low Qual Fin SF` from the set of explanatory variables considered simultaneously with `Gr Liv Area`.

4.2 Imperfect multicollinearity: variance inflation factors

After ruling out exact linear dependencies, we investigated imperfect multicollinearity among some regressors that are a priori expected to be strongly related. In particular, we focused on the following pairs of variables:

- `Garage Area` and `Garage Cars`
- `Year Built` and `Garage Yr Blt`
- `1st Flr SF` and `Total Bsmt SF`
- `Gr Liv Area` and `Full Bath`
- `Year Remod/Add` and `Year Built`

For each regressor in the model, we computed the *Variance Inflation Factor* (VIF), defined as:

$$\text{VIF}_j = \frac{1}{1 - R_j^2}$$

where R_j^2 is the coefficient of determination obtained by regressing the j-th regressor on all the remaining explanatory variables. Large VIF values indicate that the variable is highly correlated with the others and that its coefficient may be imprecisely estimated.

In line with standard practice, we used the threshold VIF>10 as an indicator of problematic multicollinearity. Whenever a variable within one of the above pairs exhibited a VIF larger than 10, we considered it as redundant and removed it from the model, while keeping the counterpart that is more interpretable and economically meaningful in the context of housing prices.

```python
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Variables you want to check for multicollinearity
vif_vars = [
    "Garage Area",
    "Garage Cars",
    "Year Built",
    "Garage Yr Blt",
    "1st Flr SF",
    "Total Bsmt SF",
    "Gr Liv Area",
    "Full Bath",
    "Year Remod/Add",
]

# Sub-dataframe with only those columns, drop rows with missing values
X_vif = df[vif_vars].dropna()

# Add constant
X_vif_const = sm.add_constant(X_vif)

vif_data = pd.DataFrame(
    {
        "variable": X_vif_const.columns,
        "VIF": [
            variance_inflation_factor(X_vif_const.values, i)
            for i in range(X_vif_const.shape[1])
        ],
```

```
        }
)

print(vif_data)

print("\nVariables with VIF > 10:")
print(vif_data[vif_data["VIF"] > 10])
```

```
          variable            VIF
0              const   15686.127006
1        Garage Area       4.101956
2        Garage Cars       4.172810
3         Year Built       3.816438
4       Garage Yr Blt      4.193325
5          1st Flr SF      3.390257
6       Total Bsmt SF      3.116352
7         Gr Liv Area      2.312216
8           Full Bath      2.186403
9       Year Remod/Add     1.921233


Variables with VIF > 10:
   variable            VIF
0     const   15686.127006
```

[9]:
```
high_vif = vif_data[(vif_data["VIF"] > 10) & (vif_data["variable"] != "const")]
print("Variables suggested for removal due to high VIF:", high_vif["variable"].
↪tolist())
```

```
Variables suggested for removal due to high VIF: []
```

4.3 Structural stability: Chow test around the 2008 financial crisis

The period covered by the dataset includes the years surrounding the 2008 financial crisis, which may have affected the housing market and, consequently, the relationship between the explanatory variables and SalePrice. To assess whether the regression coefficients are stable over time, we performed a Chow test for a structural break with respect to the year of sale (`Yr Sold`).

We split the sample into two subperiods:

- pre-crisis period: transactions in 2006–2007
- crisis and post-crisis period: transactions in 2008–2010

Let RSS_P and RSS_C denote the residual sum of squares of the model estimated separately on the pre-crisis and crisis/post-crisis subsamples, and let $RSS_{P \cup C}$ be the residual sum of squares of the model estimated on the full pooled sample. Denoting by k the number of estimated parameters (including the intercept) and by n_P and n_C the sample sizes of the two subsamples, the Chow test statistic is:

$$F = \frac{\dfrac{RSS_{P \cup C} - (RSS_P + RSS_C)}{k}}{\dfrac{RSS_P + RSS_C}{n_P + n_C - 2k}}$$

which under the null hypothesis of parameter stability follows an F-distribution with $(k, n_P + n_C - 2k)$ degrees of freedom.

```python
# invece di:
# data = pd.concat([y, X], axis=1).dropna()

data = pd.concat([y, X, df["Yr Sold"]], axis=1).dropna()

y_clean = data["SalePrice"].to_numpy(dtype=float)
X_clean = data.drop(columns=["SalePrice", "Yr Sold"]).to_numpy(dtype=float)

X_clean = sm.add_constant(X_clean)

model = sm.OLS(y_clean, X_clean).fit()
print(model.summary())
```

```
                           OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.855
Model:                            OLS   Adj. R-squared:                  0.852
Method:                 Least Squares   F-statistic:                     280.9
Date:                Sat, 22 Nov 2025   Prob (F-statistic):               0.00
Time:                        22:33:26   Log-Likelihood:                -32513.
No. Observations:                2770   AIC:                         6.514e+04
Df Residuals:                    2712   BIC:                         6.549e+04
Df Model:                          57
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const       -8.876e+05    1.32e+05     -6.724      0.000   -1.15e+06   -6.29e+05
x1             44.3995      2.027     21.906      0.000      40.425      48.374
x2             -3.9192      3.104     -1.263      0.207     -10.006       2.167
x3             23.4093      2.665      8.784      0.000      18.183      28.635
x4              0.6152      0.085      7.212      0.000       0.448       0.782
x5          -3756.4845   1658.574     -2.265      0.024   -7008.682    -504.287
x6             16.5662      6.695      2.475      0.013       3.439      29.693
x7           7228.4564   1931.315      3.743      0.000    3441.459       1.1e+04
x8            -39.5767     47.771     -0.828      0.407    -133.249      54.095
x9            247.2766     57.053      4.334      0.000     135.404     359.149
x10           259.1812     44.615      5.809      0.000     171.698     346.664
x11          1.261e+04    820.520     15.370      0.000       1.1e+04    1.42e+04
x12         -3.551e+04   5801.478     -6.120      0.000   -4.69e+04   -2.41e+04
```

x13	-2.923e+04	3179.782	-9.192	0.000	-3.55e+04	-2.3e+04
x14	-5.268e+04	3.14e+04	-1.678	0.094	-1.14e+05	8889.597
x15	-3.553e+04	3552.403	-10.001	0.000	-4.25e+04	-2.86e+04
x16	-3.79e+04	8692.842	-4.360	0.000	-5.49e+04	-2.09e+04
x17	-2.849e+04	4109.779	-6.931	0.000	-3.65e+04	-2.04e+04
x18	-2.825e+04	4651.880	-6.073	0.000	-3.74e+04	-1.91e+04
x19	-2.125e+04	4794.607	-4.433	0.000	-3.07e+04	-1.19e+04
x20	-2.251e+04	2492.245	-9.033	0.000	-2.74e+04	-1.76e+04
x21	-2909.7269	2.36e+04	-0.123	0.902	-4.93e+04	4.35e+04
x22	-1.96e+04	2978.602	-6.581	0.000	-2.54e+04	-1.38e+04
x23	2288.6955	1.19e+04	0.192	0.848	-2.11e+04	2.56e+04
x24	-7326.1849	9249.664	-0.792	0.428	-2.55e+04	1.08e+04
x25	1.098e+04	7863.407	1.397	0.163	-4436.016	2.64e+04
x26	2.192e+04	7928.231	2.765	0.006	6374.661	3.75e+04
x27	8528.3242	6233.377	1.368	0.171	-3694.325	2.08e+04
x28	3.186e+04	7146.519	4.459	0.000	1.79e+04	4.59e+04
x29	-8111.6946	6852.156	-1.184	0.237	-2.15e+04	5324.281
x30	4281.1987	6486.531	0.660	0.509	-8437.844	1.7e+04
x31	1.275e+04	1.25e+04	1.022	0.307	-1.17e+04	3.72e+04
x32	1.129e+05	2.28e+04	4.962	0.000	6.83e+04	1.57e+05
x33	7997.7135	8736.543	0.915	0.360	-9133.242	2.51e+04
x34	-3216.6788	3.2e+04	-0.101	0.920	-6.6e+04	5.95e+04
x35	2471.7518	9517.391	0.260	0.795	-1.62e+04	2.11e+04
x36	2205.6500	6858.340	0.322	0.748	-1.12e+04	1.57e+04
x37	3331.9295	6567.072	0.507	0.612	-9545.042	1.62e+04
x38	-5548.4113	8930.430	-0.621	0.534	-2.31e+04	1.2e+04
x39	3487.8154	6725.315	0.519	0.604	-9699.446	1.67e+04
x40	5.949e+04	7149.219	8.321	0.000	4.55e+04	7.35e+04
x41	3.104e+04	6570.613	4.724	0.000	1.82e+04	4.39e+04
x42	1234.4781	7905.291	0.156	0.876	-1.43e+04	1.67e+04
x43	868.9178	8469.032	0.103	0.918	-1.57e+04	1.75e+04
x44	3828.3917	6851.807	0.559	0.576	-9606.899	1.73e+04
x45	732.8101	6622.674	0.111	0.912	-1.23e+04	1.37e+04
x46	1.747e+04	7563.024	2.310	0.021	2640.863	3.23e+04
x47	5.065e+04	7397.165	6.847	0.000	3.61e+04	6.52e+04
x48	1.653e+04	7004.926	2.360	0.018	2793.712	3.03e+04
x49	2.556e+04	8697.728	2.939	0.003	8509.290	4.26e+04
x50	7187.1732	2.39e+04	0.301	0.764	-3.97e+04	5.4e+04
x51	2.269e+04	2.33e+04	0.976	0.329	-2.29e+04	6.83e+04
x52	1.445e+04	3.35e+04	0.431	0.667	-5.13e+04	8.02e+04
x53	1.963e+04	2.36e+04	0.831	0.406	-2.67e+04	6.59e+04
x54	2.938e+04	2.26e+04	1.299	0.194	-1.5e+04	7.37e+04
x55	1.912e+04	2.26e+04	0.845	0.398	-2.52e+04	6.35e+04
x56	-3.536e+04	3.1e+04	-1.141	0.254	-9.61e+04	2.54e+04
x57	-2.675e+04	2.53e+04	-1.058	0.290	-7.63e+04	2.28e+04

```
==============================================================================
Omnibus:                       1640.604   Durbin-Watson:                  1.703
Prob(Omnibus):                    0.000   Jarque-Bera (JB):          230338.424
```

```
Skew:                          -1.805   Prob(JB):                          0.00
Kurtosis:                      47.527   Cond. No.                      3.08e+06
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 3.08e+06. This might indicate that there are
strong multicollinearity or other numerical problems.
```

[11]:
```python
from scipy import stats

# Masks for the two periods
pre_mask = data["Yr Sold"].isin([2006, 2007])
post_mask = data["Yr Sold"].between(2008, 2010)

data_pre = data[pre_mask].copy()
data_post = data[post_mask].copy()

# Build matrices for each subsample
y_pre = data_pre["SalePrice"].to_numpy(dtype=float)
X_pre = data_pre.drop(columns=["SalePrice", "Yr Sold"]).to_numpy(dtype=float)
X_pre = sm.add_constant(X_pre)

y_post = data_post["SalePrice"].to_numpy(dtype=float)
X_post = data_post.drop(columns=["SalePrice", "Yr Sold"]).to_numpy(dtype=float)
X_post = sm.add_constant(X_post)

# Refit OLS on full sample (if not already done)
y_full = data["SalePrice"].to_numpy(dtype=float)
X_full = data.drop(columns=["SalePrice", "Yr Sold"]).to_numpy(dtype=float)
X_full = sm.add_constant(X_full)

model_full = sm.OLS(y_full, X_full).fit()
model_pre = sm.OLS(y_pre, X_pre).fit()
model_post = sm.OLS(y_post, X_post).fit()

# RSS
RSS_full = (model_full.resid**2).sum()
RSS_pre = (model_pre.resid**2).sum()
RSS_post = (model_post.resid**2).sum()

k = len(model_full.params)   # number of parameters including intercept
n_pre = len(y_pre)
n_post = len(y_post)

num = (RSS_full - (RSS_pre + RSS_post)) / k
```

```
den = (RSS_pre + RSS_post) / (n_pre + n_post - 2 * k)
F_chow = num / den

p_value = 1 - stats.f.cdf(F_chow, k, n_pre + n_post - 2 * k)

print(f"Chow test F-statistic: {F_chow:.3f}")
print(f"p-value: {p_value:.4f}")
print(f"n_pre = {n_pre}, n_post = {n_post}, k = {k}")
```

```
Chow test F-statistic: 1.610
p-value: 0.0026
n_pre = 1254, n_post = 1516, k = 58
```

References

[Abd22] Azad Abdulhafedh. Incorporating multiple linear regression in predicting the house prices using a big real estate dataset with 80 independent variables. *Open Access Library Journal*, 2022.

[DC11]   Dean De Cock. Ames, Iowa: alternative to the Boston housing data as an end of semester segression project. *Journal of Statistics Education*, 2011.

[Han23] Yueting Han. Price prediction of Ames housing through advanced regression techniques. *BCP Business & Management*, 2023.

[Ye24]   Qiongwei Ye. House price prediction using machine learning for Ames, Iowa. *Applied and Computational Engineering*, 2024.

[ZZS08] Joachim Zietz, Emily Norman Zietz, and Stacy Sirmans. Determinants of house prices: a quantile regression approach. *The Journal of Real Estate Finance and Economics*, 2008.