

Group10 AmesHousing

December 14, 2025

Authors

Mattia Zanin - mattia.zanin@studenti.unipd.it
Matteo Giorgi - matteo.giorgi.1@studenti.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 believe that some of them have a negligible 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 example of such variables are:

- **PID** is just an identification number which 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:

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

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*

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

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*

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

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.

```
[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")

[2]: # SalePrice 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",
]
```

```

[3]: # 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, then 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:          0.854
Model:                  OLS    Adj. R-squared:      0.851
Method:                 Least Squares    F-statistic:      325.0
Date:                  Sun, 23 Nov 2025    Prob (F-statistic):      0.00
Time:                  20:54:50    Log-Likelihood:      -32523.
No. Observations:      2770    AIC:                6.515e+04
Df Residuals:          2720    BIC:                6.544e+04
Df Model:               49
Covariance Type:        nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
-----	-----	-----	-----	-----	-----	-----
const	-8.265e+05	1.29e+05	-6.422	0.000	-1.08e+06	-5.74e+05
x1	44.4270	2.022	21.971	0.000	40.462	48.392
x2	-4.1549	3.094	-1.343	0.179	-10.222	1.912
x3	23.9307	2.659	9.001	0.000	18.717	29.144
x4	0.6281	0.085	7.417	0.000	0.462	0.794
x5	-3545.9700	1656.644	-2.140	0.032	-6794.378	-297.562
x6	18.8011	6.634	2.834	0.005	5.793	31.809
x7	6711.8835	1918.202	3.499	0.000	2950.603	1.05e+04
x8	-40.6771	47.808	-0.851	0.395	-134.421	53.067
x9	230.0968	56.616	4.064	0.000	119.081	341.112
x10	260.2723	44.621	5.833	0.000	172.778	347.767
x11	1.267e+04	812.034	15.608	0.000	1.11e+04	1.43e+04
x12	-3.622e+04	5796.989	-6.248	0.000	-4.76e+04	-2.49e+04
x13	-2.949e+04	3185.023	-9.259	0.000	-3.57e+04	-2.32e+04
x14	-5.321e+04	3.15e+04	-1.691	0.091	-1.15e+05	8477.210
x15	-3.602e+04	3553.574	-10.137	0.000	-4.3e+04	-2.91e+04
x16	-4.074e+04	8569.363	-4.754	0.000	-5.75e+04	-2.39e+04
x17	-2.815e+04	4116.626	-6.838	0.000	-3.62e+04	-2.01e+04
x18	-2.756e+04	4655.430	-5.921	0.000	-3.67e+04	-1.84e+04
x19	-1.999e+04	4783.540	-4.179	0.000	-2.94e+04	-1.06e+04
x20	-2.212e+04	2488.641	-8.888	0.000	-2.7e+04	-1.72e+04
x21	-1.169e+04	2.23e+04	-0.523	0.601	-5.55e+04	3.21e+04
x22	-1.896e+04	2968.316	-6.389	0.000	-2.48e+04	-1.31e+04
x23	-7220.6411	1.15e+04	-0.626	0.531	-2.98e+04	1.54e+04
x24	-1.717e+04	8665.776	-1.982	0.048	-3.42e+04	-180.588
x25	3984.4299	7475.569	0.533	0.594	-1.07e+04	1.86e+04
x26	2.179e+04	7937.248	2.746	0.006	6229.705	3.74e+04
x27	8585.2979	6245.442	1.375	0.169	-3660.993	2.08e+04
x28	3.042e+04	7132.656	4.265	0.000	1.64e+04	4.44e+04
x29	-8869.8054	6842.549	-1.296	0.195	-2.23e+04	4547.315
x30	5017.8563	6491.215	0.773	0.440	-7710.355	1.77e+04
x31	1.343e+04	1.25e+04	1.074	0.283	-1.11e+04	3.79e+04
x32	1.037e+05	2.26e+04	4.592	0.000	5.94e+04	1.48e+05
x33	-5142.4837	7808.569	-0.659	0.510	-2.05e+04	1.02e+04

x34	-1.213e+04	3.13e+04	-0.387	0.699	-7.36e+04	4.93e+04
x35	-7190.2537	8936.048	-0.805	0.421	-2.47e+04	1.03e+04
x36	1422.6316	6861.945	0.207	0.836	-1.2e+04	1.49e+04
x37	3146.3528	6571.044	0.479	0.632	-9738.391	1.6e+04
x38	-5076.5644	8947.019	-0.567	0.570	-2.26e+04	1.25e+04
x39	3543.4928	6737.447	0.526	0.599	-9667.539	1.68e+04
x40	5.986e+04	7163.423	8.356	0.000	4.58e+04	7.39e+04
x41	3.16e+04	6580.385	4.802	0.000	1.87e+04	4.45e+04
x42	-8032.4382	7239.898	-1.109	0.267	-2.22e+04	6163.818
x43	-904.4774	8429.025	-0.107	0.915	-1.74e+04	1.56e+04
x44	3736.6645	6855.847	0.545	0.586	-9706.531	1.72e+04
x45	882.4611	6625.991	0.133	0.894	-1.21e+04	1.39e+04
x46	1.308e+04	6362.505	2.056	0.040	604.884	2.56e+04
x47	5.126e+04	7408.913	6.918	0.000	3.67e+04	6.58e+04
x48	1.653e+04	7003.835	2.361	0.018	2800.800	3.03e+04
x49	2.579e+04	8714.923	2.959	0.003	8698.137	4.29e+04

Omnibus:	1655.724	Durbin-Watson:	1.696
Prob(Omnibus):	0.000	Jarque-Bera (JB):	228465.961
Skew:	-1.838	Prob(JB):	0.00
Kurtosis:	47.339	Cond. No.	2.98e+06

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 2.98e+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 (x_1 – x_{11}) corresponds to the “core” numeric regressors (size, age, quality):

- $x_1 \approx 44.4$ (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.
- $x_3 \approx 23.4$ (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)
- $x_4 \approx 0.62$ (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
- $x_5 \approx -3,756$ (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
- $x_6 \approx 16.6$ (Garage Area) and $x_7 \approx 7,228$ (Garage Cars) are both positive and significant, meaning that larger garages and the capacity to park more cars are rewarded by the market
- x_8 (Garage Yr Blt) is not statistically significant ($p \approx 0.41$), suggesting that, after controlling for the other variables, the construction year of the garage does not have a clear marginal effect
- $x_9 \approx 247.3$ (Year Built) and $x_{10} \approx 259.2$ (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
- $x_{11} \approx 12,610$ (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 (x_{12} , x_{13} , x_{15} – x_{20} , x_{22} , ..., x_{57}) 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 x_{28} – x_{41} corresponds mostly to Neighborhood dummies. Some neighbourhoods show large positive and significant coefficients (for example, x_{32} , x_{40} , x_{41} , x_{47} , x_{48} , x_{49}), 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 x_{50} – x_{57}) 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.

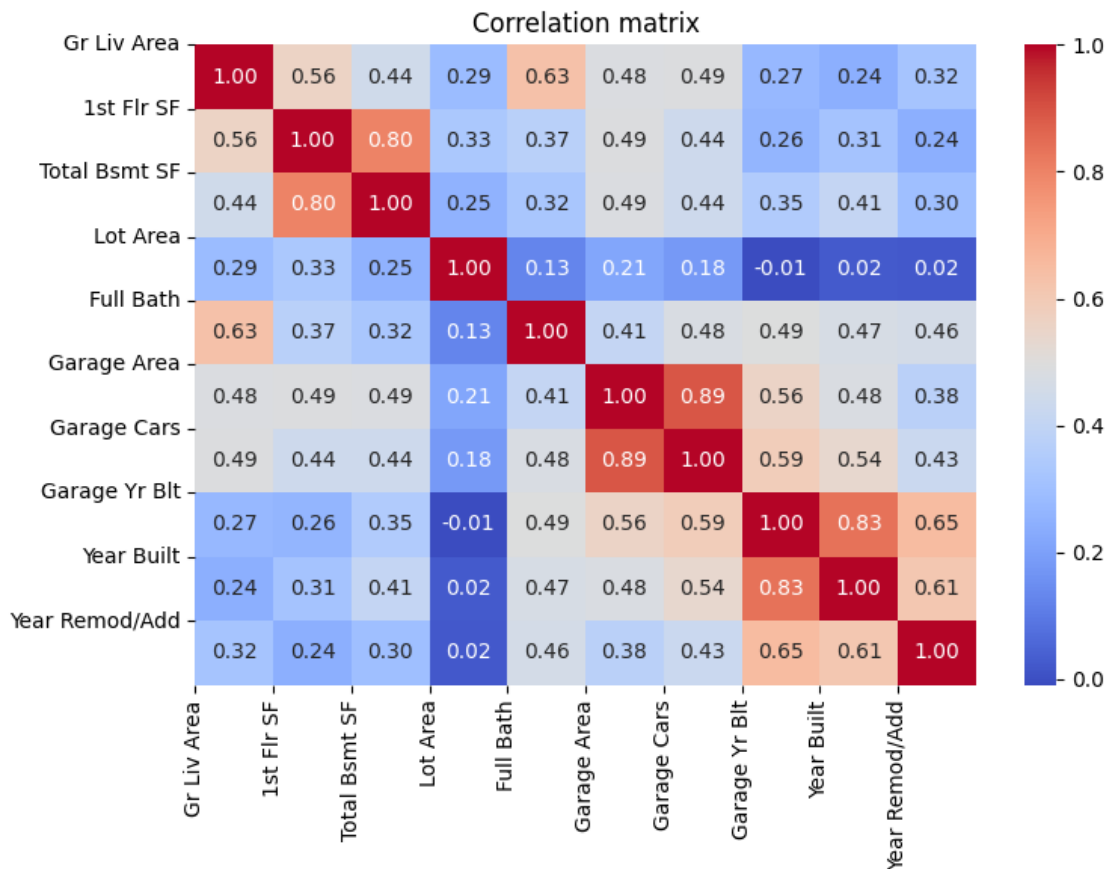
3.5 Covariance matrix

```
[4]: 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()
```



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:

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

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 verify this identity in the data by computing the difference:

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

```
[5]: # 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 statistics about the differences
# Use absolute values 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 of these regressor, we computed the *Variance Inflation Factor* (VIF). For a given regressor X_j , the VIF is defined as

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

where R_j^2 is the coefficient of determination from the regression of X_j on all the remaining regressors. Large VIF values indicate that the variable is highly correlated with the others and that its coefficient may be imprecisely estimated.

```
[6]: from statsmodels.stats.outliers_influence import variance_inflation_factor

# Variables we want to check for multicollinearity:
# Garage Area <-> Garage Cars
# Year Built <-> Garage Yr Blt
# 1st Flr SF <-> Total Bsmt SF
# Gr Liv Area <-> Full Bath
# Year Remod/Add (alone)
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",
]

# New sub-dataframe with vif_vars columns (drop rows with Nan)
# Add constant for the intercept manually (can't use sm.add_constant() formally
↳ because it returns an ndarray):
# some statsmodels functions expect the constant column to be present for
↳ consistent behavior
X_vif = df[vif_vars].dropna().copy()
X_vif["const"] = 1.0

# Convert to numpy array (needed for VIF calculation)
# Build a DataFrame vif_data with two columns:
# variable -> column name (Garage Area, Garage Cars, ..., const),
# VIF      -> corresponding VIF value calculated iterating on each column
X_vals = X_vif.to_numpy()
vif_data = pd.DataFrame(
```

```

    {
        "variable": X_vif.columns,
        "VIF": [variance_inflation_factor(X_vals, i) for i in range(X_vals.
↪shape[1])],
    }
)

print(vif_data)
print("\nVariables with VIF > 10:")
print(vif_data[(vif_data["VIF"] > 10) & (vif_data["variable"] != "const")])

```

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

```

Variables with VIF > 10:
Empty DataFrame
Columns: [variable, VIF]
Index: []

```

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.

We report the VIF values for the selected variables in the output above and we observe that no variable exceed the threshold, indicating that multicollinearity is not a significant concern in our model.

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_{PUC} 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:

$$F = \frac{(RSS_{PUC} - (RSS_P + RSS_C))/k}{(RSS_P + RSS_C)/(n_P + n_C - 2k)}$$

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

```
[7]: # Concatenate y, X, and year
# ATTENZIONE: cambia "Yr Sold" in "Yr.Sold" se il tuo CSV ha il punto nel nome
data = pd.concat([y, X, df["Yr Sold"]], axis=1).dropna()
data = data.rename(
    columns={"Yr Sold": "YrSold"}
) # rinominare evita problemi negli spazi

# Pre-crisis block: 2006-2007
data_pre = data[(data["YrSold"] == 2006) | (data["YrSold"] == 2007)]

# Post-crisis block: 2008-2010
data_post = data[(data["YrSold"] >= 2008) & (data["YrSold"] <= 2010)]

# y e X per il periodo 2006-2007
y_pre = data_pre["SalePrice"].to_numpy(dtype=float)
X_pre = data_pre.drop(columns=["SalePrice", "YrSold"]).to_numpy(dtype=float)
X_pre = sm.add_constant(X_pre)

model_pre = sm.OLS(y_pre, X_pre).fit()
print("Model for 2006-2007")
print(model_pre.summary())

# y e X per il periodo 2008-2010
y_post = data_post["SalePrice"].to_numpy(dtype=float)
X_post = data_post.drop(columns=["SalePrice", "YrSold"]).to_numpy(dtype=float)
X_post = sm.add_constant(X_post)

model_post = sm.OLS(y_post, X_post).fit()
print("Model for 2008-2010")
print(model_post.summary())
```

Model for 2006-2007

OLS Regression Results

```
=====
Dep. Variable:          y    R-squared:          0.854
Model:                OLS    Adj. R-squared:       0.849
Method:             Least Squares    F-statistic:      150.6
```


Date: Sun, 23 Nov 2025 Prob (F-statistic): 0.00
Time: 20:54:52 Log-Likelihood: -14766.
No. Observations: 1254 AIC: 2.963e+04
Df Residuals: 1206 BIC: 2.987e+04
Df Model: 47
Covariance Type: nonrobust

	coef	std err	t	P> t	[0.025	0.975]
const	-7.795e+05	2.05e+05	-3.797	0.000	-1.18e+06	-3.77e+05
x1	44.4591	3.152	14.103	0.000	38.274	50.644
x2	-7.9598	4.781	-1.665	0.096	-17.341	1.421
x3	23.3300	4.296	5.431	0.000	14.902	31.758
x4	0.8230	0.149	5.541	0.000	0.532	1.114
x5	-1109.5090	2615.054	-0.424	0.671	-6240.070	4021.052
x6	40.1021	10.827	3.704	0.000	18.860	61.344
x7	-231.5353	3135.931	-0.074	0.941	-6384.023	5920.952
x8	-72.5288	70.500	-1.029	0.304	-210.845	65.787
x9	236.7150	94.017	2.518	0.012	52.259	421.171
x10	263.6121	72.424	3.640	0.000	121.521	405.703
x11	1.389e+04	1274.799	10.897	0.000	1.14e+04	1.64e+04
x12	-3.377e+04	8625.180	-3.916	0.000	-5.07e+04	-1.69e+04
x13	-3.065e+04	4865.199	-6.300	0.000	-4.02e+04	-2.11e+04
x14	-1.686e-11	2.47e-11	-0.684	0.494	-6.52e-11	3.15e-11
x15	-3.604e+04	5479.585	-6.577	0.000	-4.68e+04	-2.53e+04
x16	-3.476e+04	1.68e+04	-2.068	0.039	-6.77e+04	-1778.573
x17	-1.789e+04	6827.577	-2.620	0.009	-3.13e+04	-4493.031
x18	-1.471e+04	7847.068	-1.875	0.061	-3.01e+04	680.614
x19	-2.648e+04	7293.371	-3.630	0.000	-4.08e+04	-1.22e+04
x20	-2.894e+04	3918.200	-7.386	0.000	-3.66e+04	-2.13e+04
x21	6.906e-11	1.77e-11	3.912	0.000	3.44e-11	1.04e-10
x22	-2.246e+04	4704.304	-4.775	0.000	-3.17e+04	-1.32e+04
x23	-2510.7771	2.45e+04	-0.102	0.918	-5.06e+04	4.56e+04
x24	-3.901e+04	1.3e+04	-2.998	0.003	-6.45e+04	-1.35e+04
x25	-1.075e+04	1.13e+04	-0.954	0.340	-3.28e+04	1.13e+04
x26	-1402.5834	1.23e+04	-0.114	0.909	-2.55e+04	2.27e+04
x27	-5391.8843	9045.990	-0.596	0.551	-2.31e+04	1.24e+04
x28	1.528e+04	1.04e+04	1.462	0.144	-5217.938	3.58e+04
x29	-2.803e+04	9952.287	-2.817	0.005	-4.76e+04	-8505.792
x30	-9134.4236	9414.390	-0.970	0.332	-2.76e+04	9335.978
x31	1806.1782	1.69e+04	0.107	0.915	-3.14e+04	3.5e+04
x32	8.707e+04	2.45e+04	3.558	0.000	3.91e+04	1.35e+05
x33	-2.016e+04	1.14e+04	-1.768	0.077	-4.25e+04	2212.258
x34	-2.804e+04	3.34e+04	-0.839	0.401	-9.36e+04	3.75e+04
x35	-2.909e+04	1.33e+04	-2.194	0.028	-5.51e+04	-3075.883
x36	-1.959e+04	1.02e+04	-1.918	0.055	-3.96e+04	447.955
x37	-1.407e+04	9611.862	-1.464	0.143	-3.29e+04	4784.089
x38	-1.817e+04	1.6e+04	-1.135	0.257	-4.96e+04	1.32e+04

x39	-1.113e+04	1e+04	-1.111	0.267	-3.08e+04	8514.978
x40	5.579e+04	1.05e+04	5.305	0.000	3.52e+04	7.64e+04
x41	1.364e+04	9576.876	1.424	0.155	-5149.071	3.24e+04
x42	-2.495e+04	1.08e+04	-2.315	0.021	-4.61e+04	-3808.977
x43	-1.683e+04	1.32e+04	-1.272	0.204	-4.28e+04	9130.198
x44	-1.469e+04	9993.298	-1.470	0.142	-3.43e+04	4915.600
x45	-1.022e+04	1.01e+04	-1.011	0.312	-3.01e+04	9611.148
x46	-2665.7254	9280.906	-0.287	0.774	-2.09e+04	1.55e+04
x47	5.431e+04	1.07e+04	5.067	0.000	3.33e+04	7.53e+04
x48	8986.4805	1.02e+04	0.877	0.381	-1.11e+04	2.91e+04
x49	2.858e+04	1.25e+04	2.287	0.022	4064.612	5.31e+04

```

=====
Omnibus:                636.553    Durbin-Watson:                1.676
Prob(Omnibus):           0.000    Jarque-Bera (JB):            53906.323
Skew:                    -1.443    Prob(JB):                     0.00
Kurtosis:                34.990    Cond. No.                     1.19e+16
=====

```

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The smallest eigenvalue is 1.59e-21. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

Model for 2008-2010

OLS Regression Results

```

=====
Dep. Variable:            y    R-squared:                0.863
Model:                    OLS    Adj. R-squared:            0.858
Method:                    Least Squares    F-statistic:                196.3
Date:                      Sun, 23 Nov 2025    Prob (F-statistic):          0.00
Time:                      20:54:52    Log-Likelihood:              -17705.
No. Observations:          1516    AIC:                         3.551e+04
Df Residuals:              1468    BIC:                         3.576e+04
Df Model:                  47
Covariance Type:           nonrobust
=====

```

	coef	std err	t	P> t	[0.025	0.975]
const	-9.125e+05	1.65e+05	-5.529	0.000	-1.24e+06	-5.89e+05
x1	41.4674	2.646	15.673	0.000	36.278	46.657
x2	2.5668	4.038	0.636	0.525	-5.354	10.488
x3	21.7012	3.347	6.483	0.000	15.135	28.267
x4	0.5563	0.103	5.416	0.000	0.355	0.758
x5	-4404.3313	2132.340	-2.065	0.039	-8587.089	-221.574
x6	3.1729	8.284	0.383	0.702	-13.078	19.423
x7	1.217e+04	2394.701	5.080	0.000	7468.858	1.69e+04
x8	-14.0158	65.424	-0.214	0.830	-142.350	114.318
x9	241.2184	71.137	3.391	0.001	101.678	380.759

x10	260.2804	56.208	4.631	0.000	150.024	370.536
x11	1.188e+04	1045.270	11.362	0.000	9825.665	1.39e+04
x12	-3.92e+04	7851.268	-4.993	0.000	-5.46e+04	-2.38e+04
x13	-2.943e+04	4191.226	-7.022	0.000	-3.77e+04	-2.12e+04
x14	-5.658e+04	3.03e+04	-1.865	0.062	-1.16e+05	2939.314
x15	-3.63e+04	4650.766	-7.805	0.000	-4.54e+04	-2.72e+04
x16	-4.83e+04	9967.916	-4.846	0.000	-6.79e+04	-2.87e+04
x17	-3.615e+04	5107.042	-7.079	0.000	-4.62e+04	-2.61e+04
x18	-3.765e+04	5739.188	-6.559	0.000	-4.89e+04	-2.64e+04
x19	-1.126e+04	6333.758	-1.778	0.076	-2.37e+04	1164.982
x20	-1.326e+04	3203.592	-4.139	0.000	-1.95e+04	-6976.388
x21	-3264.3904	2.17e+04	-0.151	0.880	-4.58e+04	3.92e+04
x22	-1.258e+04	3813.072	-3.299	0.001	-2.01e+04	-5100.138
x23	3732.6709	1.34e+04	0.279	0.780	-2.25e+04	3e+04
x24	6277.9667	1.16e+04	0.543	0.587	-1.64e+04	2.89e+04
x25	2.215e+04	1e+04	2.204	0.028	2439.312	4.19e+04
x26	4.318e+04	1.05e+04	4.098	0.000	2.25e+04	6.38e+04
x27	2.398e+04	8606.775	2.786	0.005	7096.357	4.09e+04
x28	4.757e+04	9773.064	4.868	0.000	2.84e+04	6.67e+04
x29	1.197e+04	9392.652	1.275	0.203	-6451.388	3.04e+04
x30	2.064e+04	8940.350	2.308	0.021	3098.197	3.82e+04
x31	2.784e+04	1.89e+04	1.477	0.140	-9137.954	6.48e+04
x32	-1.24e-07	4.33e-08	-2.862	0.004	-2.09e-07	-3.9e-08
x33	1.194e+04	1.07e+04	1.118	0.264	-9008.464	3.29e+04
x34	2.598e-08	9.07e-09	2.864	0.004	8.19e-09	4.38e-08
x35	1.502e+04	1.2e+04	1.248	0.212	-8579.637	3.86e+04
x36	2.28e+04	9288.235	2.455	0.014	4584.847	4.1e+04
x37	2.029e+04	8998.637	2.255	0.024	2640.855	3.79e+04
x38	1.002e+04	1.11e+04	0.903	0.367	-1.18e+04	3.18e+04
x39	1.87e+04	9133.483	2.048	0.041	786.717	3.66e+04
x40	6.675e+04	9732.180	6.859	0.000	4.77e+04	8.58e+04
x41	5.069e+04	9025.989	5.616	0.000	3.3e+04	6.84e+04
x42	1.04e+04	9797.291	1.061	0.289	-8822.569	2.96e+04
x43	1.876e+04	1.11e+04	1.696	0.090	-2939.062	4.05e+04
x44	2.28e+04	9407.396	2.424	0.015	4351.162	4.13e+04
x45	1.523e+04	8915.450	1.709	0.088	-2255.270	3.27e+04
x46	2.987e+04	8721.389	3.425	0.001	1.28e+04	4.7e+04
x47	4.724e+04	1.02e+04	4.623	0.000	2.72e+04	6.73e+04
x48	2.751e+04	9558.207	2.878	0.004	8762.913	4.63e+04
x49	1.99e+04	1.21e+04	1.641	0.101	-3884.587	4.37e+04

Omnibus:	1079.625	Durbin-Watson:	1.781
Prob(Omnibus):	0.000	Jarque-Bera (JB):	181010.834
Skew:	-2.359	Prob(JB):	0.00
Kurtosis:	56.323	Cond. No.	1.24e+16

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The smallest eigenvalue is 1.8e-21. This might indicate that there are strong multicollinearity problems or that the design matrix is singular.

```
[8]: # import numpy as np
from scipy import stats

# import statsmodels.api as sm

# Create a new dataframe combining the following:
# - target variable SalePrice
# - explanatory variables used in X
# - year-of-sale variable Yr Sold
# (we used pandas.DataFrame.dropna() because each subsample must have the same
↳ regression structure)
data = pd.concat([y, X, df["Yr Sold"]], axis=1).dropna()

# Build the model on the full dataset including "Yr Sold" as regressor:
# separate y, X and add the intercept column
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)

# Estimate the model OLS from 2006 to 2010
# and obtain the RSS full
model = sm.OLS(y_clean, X_clean).fit()
print(model.summary())

def chow_test(data, y_col, split_col, split_value):
    """
    data: DataFrame con y, regressori dummy-izzati e split_col
    y_col: variabile dipendente, es. "SalePrice"
    split_col: variabile di split, es. "Yr Sold"
    split_value: valore da cui inizia il gruppo post (es. 2008)
    """
    data = data.dropna().copy()

    # Regressors = tutte le colonne tranne y_col e split_col
    x_cols = [c for c in data.columns if c not in [y_col, split_col]]

    # Funzione helper per OLS
    def fit_ols(sub):
        y = sub[y_col].to_numpy(dtype=float)
        X = sub[x_cols].to_numpy(dtype=float)
        X = sm.add_constant(X)
```

```

    return sm.OLS(y, X).fit()

# Modello full su tutto il campione
model_full = fit_ols(data)

# Split: pre = < split_value, post = >= split_value
pre = data[data[split_col] < split_value]
post = data[data[split_col] >= split_value]

model_pre = fit_ols(pre)
model_post = fit_ols(post)

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

k = len(model_full.params) # num. parametri inclusa costante
n_pre = pre.shape[0]
n_post = post.shape[0]

# Statistica F
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
df1 = k
df2 = n_pre + n_post - 2 * k
p_value = 1 - stats.f.cdf(F_chow, df1, df2)

return F_chow, p_value, df1, df2, n_pre, n_post

F_chow, p_value, df1, df2, n_pre, n_post = chow_test(
    data=data,
    y_col="SalePrice",
    split_col="Yr Sold",
    split_value=2008, # <-- 2008 va nel POST
)

print(f"Chow F-statistic: {F_chow:.3f}")
print(f"p-value: {p_value:.4f}")
print(f"df1 = {df1}, df2 = {df2}")
print(f"n_pre = {n_pre}, n_post = {n_post}")

```

OLS Regression Results

```

=====
Dep. Variable:          y    R-squared:          0.854

```

```

Model:                      OLS      Adj. R-squared:      0.851
Method:                     Least Squares    F-statistic:      325.0
Date:                       Sun, 23 Nov 2025    Prob (F-statistic):    0.00
Time:                       20:54:52      Log-Likelihood:      -32523.
No. Observations:          2770      AIC:                6.515e+04
Df Residuals:              2720      BIC:                6.544e+04
Df Model:                  49
Covariance Type:           nonrobust

```

	coef	std err	t	P> t	[0.025	0.975]
const	-8.265e+05	1.29e+05	-6.422	0.000	-1.08e+06	-5.74e+05
x1	44.4270	2.022	21.971	0.000	40.462	48.392
x2	-4.1549	3.094	-1.343	0.179	-10.222	1.912
x3	23.9307	2.659	9.001	0.000	18.717	29.144
x4	0.6281	0.085	7.417	0.000	0.462	0.794
x5	-3545.9700	1656.644	-2.140	0.032	-6794.378	-297.562
x6	18.8011	6.634	2.834	0.005	5.793	31.809
x7	6711.8835	1918.202	3.499	0.000	2950.603	1.05e+04
x8	-40.6771	47.808	-0.851	0.395	-134.421	53.067
x9	230.0968	56.616	4.064	0.000	119.081	341.112
x10	260.2723	44.621	5.833	0.000	172.778	347.767
x11	1.267e+04	812.034	15.608	0.000	1.11e+04	1.43e+04
x12	-3.622e+04	5796.989	-6.248	0.000	-4.76e+04	-2.49e+04
x13	-2.949e+04	3185.023	-9.259	0.000	-3.57e+04	-2.32e+04
x14	-5.321e+04	3.15e+04	-1.691	0.091	-1.15e+05	8477.210
x15	-3.602e+04	3553.574	-10.137	0.000	-4.3e+04	-2.91e+04
x16	-4.074e+04	8569.363	-4.754	0.000	-5.75e+04	-2.39e+04
x17	-2.815e+04	4116.626	-6.838	0.000	-3.62e+04	-2.01e+04
x18	-2.756e+04	4655.430	-5.921	0.000	-3.67e+04	-1.84e+04
x19	-1.999e+04	4783.540	-4.179	0.000	-2.94e+04	-1.06e+04
x20	-2.212e+04	2488.641	-8.888	0.000	-2.7e+04	-1.72e+04
x21	-1.169e+04	2.23e+04	-0.523	0.601	-5.55e+04	3.21e+04
x22	-1.896e+04	2968.316	-6.389	0.000	-2.48e+04	-1.31e+04
x23	-7220.6411	1.15e+04	-0.626	0.531	-2.98e+04	1.54e+04
x24	-1.717e+04	8665.776	-1.982	0.048	-3.42e+04	-180.588
x25	3984.4299	7475.569	0.533	0.594	-1.07e+04	1.86e+04
x26	2.179e+04	7937.248	2.746	0.006	6229.705	3.74e+04
x27	8585.2979	6245.442	1.375	0.169	-3660.993	2.08e+04
x28	3.042e+04	7132.656	4.265	0.000	1.64e+04	4.44e+04
x29	-8869.8054	6842.549	-1.296	0.195	-2.23e+04	4547.315
x30	5017.8563	6491.215	0.773	0.440	-7710.355	1.77e+04
x31	1.343e+04	1.25e+04	1.074	0.283	-1.11e+04	3.79e+04
x32	1.037e+05	2.26e+04	4.592	0.000	5.94e+04	1.48e+05
x33	-5142.4837	7808.569	-0.659	0.510	-2.05e+04	1.02e+04
x34	-1.213e+04	3.13e+04	-0.387	0.699	-7.36e+04	4.93e+04
x35	-7190.2537	8936.048	-0.805	0.421	-2.47e+04	1.03e+04
x36	1422.6316	6861.945	0.207	0.836	-1.2e+04	1.49e+04

x37	3146.3528	6571.044	0.479	0.632	-9738.391	1.6e+04
x38	-5076.5644	8947.019	-0.567	0.570	-2.26e+04	1.25e+04
x39	3543.4928	6737.447	0.526	0.599	-9667.539	1.68e+04
x40	5.986e+04	7163.423	8.356	0.000	4.58e+04	7.39e+04
x41	3.16e+04	6580.385	4.802	0.000	1.87e+04	4.45e+04
x42	-8032.4382	7239.898	-1.109	0.267	-2.22e+04	6163.818
x43	-904.4774	8429.025	-0.107	0.915	-1.74e+04	1.56e+04
x44	3736.6645	6855.847	0.545	0.586	-9706.531	1.72e+04
x45	882.4611	6625.991	0.133	0.894	-1.21e+04	1.39e+04
x46	1.308e+04	6362.505	2.056	0.040	604.884	2.56e+04
x47	5.126e+04	7408.913	6.918	0.000	3.67e+04	6.58e+04
x48	1.653e+04	7003.835	2.361	0.018	2800.800	3.03e+04
x49	2.579e+04	8714.923	2.959	0.003	8698.137	4.29e+04

Omnibus:	1655.724	Durbin-Watson:	1.696
Prob(Omnibus):	0.000	Jarque-Bera (JB):	228465.961
Skew:	-1.838	Prob(JB):	0.00
Kurtosis:	47.339	Cond. No.	2.98e+06

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 2.98e+06. This might indicate that there are strong multicollinearity or other numerical problems.

Chow F-statistic: 1.763

p-value: 0.0008

df1 = 50, df2 = 2670

n_pre = 1254, n_post = 1516

In our application, the test returns:

- $F = 1.610$
- $p\text{-value} = 0.0026$

Since the p-value is below 0.05, we reject the null hypothesis of parameter stability. This indicates the presence of a statistically significant structural break between the pre-crisis and post-crisis periods. In other words, the relationship between the explanatory variables and `SalePrice` changed after the 2008 financial crisis, suggesting that the housing market in Ames did not follow the same pricing dynamics before and after the economic downturn.

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 regression 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.