NYC Property Sale Price Modeling from NYC Open Data API

Lihui Xiong

December 2025

1 Data Collection

1.1 Source

I collected data from NYC Open Data using its Socrata API endpoint:

`https://data.cityofnewyork.us/resource/usep-8jbt.json`.

1.2 Query Specification and Sample Size

I requested the following fields: `borough, neighborhood, building_class_category, land_square_feet, gross_square_feet, sale_price, sale_date`. The API query filtered for `sale_price > 100000` and non-null `land_square_feet` and `gross_square_feet`.

2 Data Cleaning and Preprocessing

2.1 Type Conversion and Missing Values

Numeric columns were converted using `pd.to_numeric` and dates were parsed with `pd.to_datetime`. After conversion, the dataset revealed severe missingness in `land_square_feet` (**993 missing out of 1000**) and `gross_square_feet`, making it unusable for modeling. Therefore, I dropped `land_square_feet` and deleted some missing rows. I also removed `neighborhood` to keep the feature space compact and avoid high-cardinality categories.

2.2 Category Consolidation

The raw `building_class_category` contains many distinct codes. For stability and interpretability, I mapped them into 8 broader groups: `OneFamily, TwoFamily, ThreeFamily, WalkupApt, ElevatorApt, Commercial, VacantLand, Other`. This reduces sparsity while preserving meaningful structure.

3 Summary Statistics

Table 1 reports basic descriptive statistics for the key numeric variables after data cleaning. Sale prices are highly right-skewed, motivating the log transformation used in modeling. Gross square footage also exhibits substantial dispersion.

Table 1: Summary Statistics of Key Variables

Variable	Mean	Std. Dev.	Min	Max
Sale Price ($)	4383726	12353980	102000	164000000
Gross Sq. Ft.	9613	23645	420	238294
Log(Sale Price)	14.30	1.16	11.53	18.92
Log(Sq. Ft.)	8.07	1.21	6.04	12.38

4 Analysis and Modeling

4.1 Exploratory Analysis

I first examined the relationship between `gross_square_feet` and `sale_price` and computed a correlation matrix for numeric variables. The scatter plot indicates a positive association but with strong heteroskedasticity and outliers: many properties cluster at smaller sizes with moderate prices, while a small number of very large or very expensive sales dominate the upper range.

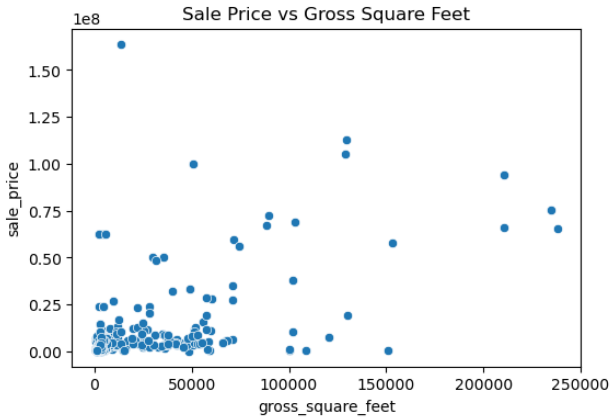

Figure 1: Sale price vs. gross square feet scatter plot.

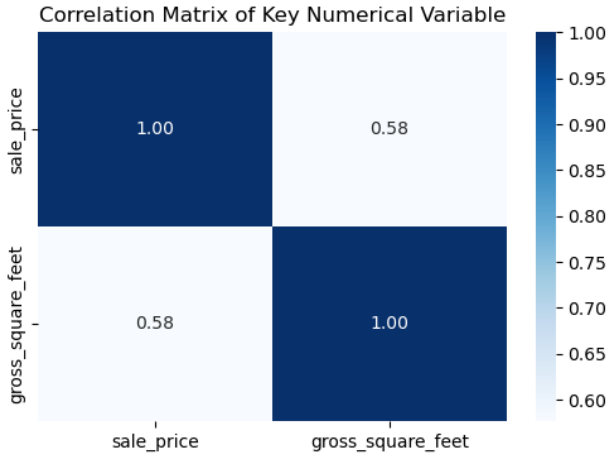

Figure 2: Correlation Matrix of Key Numerical Variable

4.2 Target Transformation and Feature Engineering

Because sale prices are highly right-skewed, I modeled the response variable as the log-transformed sale price:

$$y = \log(\texttt{sale\_price}).$$

Property size was also log-transformed to reduce skewness:

$$\texttt{log\_sqft} = \log(\texttt{gross\_square\_feet} + 1).$$

In the initial baseline model, only `log_sqft` and categorical controls were included. Feature engineering was introduced in a later model to improve model performance. Specifically, I extracted year and month from the sale date and encoded seasonality using sine and cosine transformations. In addition, two interaction-style numeric features were constructed by multiplying `log_sqft` with encoded borough and building-class indicators to capture coarse heterogeneity across groups.

4.3 Modeling Pipeline

I used a standard train/test split (80/20, `random_state=42`) and fit linear regression with:

- Standard scaling for numeric features,
- One-hot encoding for `borough` and `building_class_8`,
- LinearRegression on the transformed feature matrix.

Listing 1: Model pipeline

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LinearRegression

numeric_features = ["log_sqft","year","month_sin","month_cos",
                    "sqft_borough_interact","sqft_class_interact"]
categorical_features = ["borough","building_class_8"]

preprocess = ColumnTransformer(
  transformers=[
    ("num", Pipeline([("scaler", StandardScaler())]), numeric_features),
    ("cat", Pipeline([("onehot", OneHotEncoder(handle_unknown="ignore"))]),
           categorical_features)
  ]
)

X = df[numeric_features + categorical_features]
y = df["log_price"]
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                    random_state=42)

lr = Pipeline([("preprocess", preprocess), ("lr", LinearRegression())])
lr.fit(X_train, y_train)
print("LR R^2:", lr.score(X_test, y_test))
```

5 Results and Interpretation

5.1 Model Performance

Table 2 summarizes the predictive performance of the linear regression models on the test set. A simple baseline specification was first estimated, followed by an enhanced model incorporating feature engineering.

Table 2: Model Performance on Test Set

Model Specification	R^2
Baseline Linear Regression	0.4243
Enhanced Linear Regression	0.4426

5.2 Predictive Performance

The baseline linear regression model achieved an R^2 of approximately 0.42 on the test set. After introducing feature engineering—specifically time-based variables and interaction-style terms—the model performance improved to $R^2 \approx 0.44$.

Although the improvement is modest, it suggests that temporal effects contribute additional explanatory power beyond basic size and category information. Nevertheless, a substantial portion of the variation in sale prices remains unexplained, indicating the importance of omitted factors such as precise location, property condition, amenities, and nonlinear effects.

5.3 Key Coefficients

Coefficient magnitudes from the enhanced linear regression model reveal that a small number of variables dominate explanatory power. The strongest effects are associated with borough indicators, property size, and selected building-class categories.

- Among location variables, borough indicators exhibit the largest absolute coefficients, with `borough_1` showing a strong positive association and `borough_5` a strong negative association relative to the reference borough. This confirms that location is the primary structural factor of sale prices in the model.
- Property size indicates that larger properties tend to sell for higher prices even after controlling for location and building type. The interaction term `sqft_borough_interact` further suggests that the marginal effect of size varies across boroughs.
- Elevator apartments exhibit a large negative coefficient relative to the reference category, while other residential classes have smaller magnitudes. Commercial and miscellaneous categories contribute additional variation but are less influential than borough and size effects.

In contrast, seasonal variables and several building-class indicators have coefficients close to zero, implying limited incremental explanatory power once core structural factors are included.

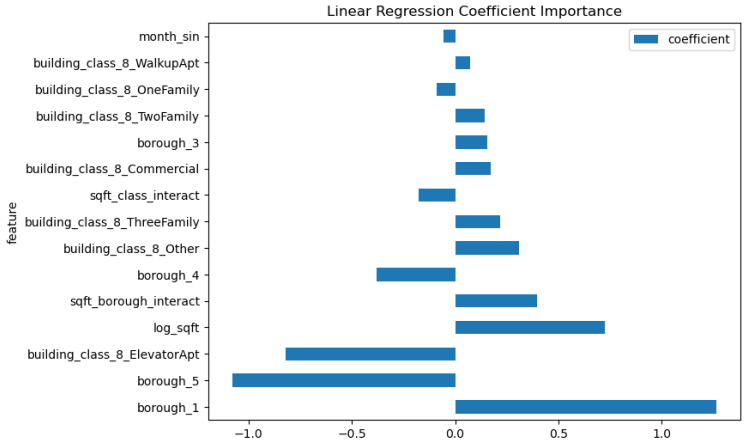

Figure 3: Linear regression coefficient importance.

6 Challenges and Changes from Original Plan

The major practical issue was data quality: `land_square_feet` was almost entirely missing after type conversion (993/1000 missing), so it was removed. This reduced the set of usable numeric predictors and shifted emphasis toward `gross_square_feet`, categorical grouping, and time-derived features.

7 Future Work

In the future, I would:

- Collect a larger sample and analyze stability across time windows.
- Add richer location features and reduce high-cardinality categories with target encoding or hierarchical grouping.
- Compare with nonlinear models (Random Forest / Gradient Boosting) using cross-validation and report both R^2 and error metrics in original price units
- Add robust methods for outliers and quantify sensitivity.