Feature Engineering Assignment

House Price Prediction Dataset

Submitted By

Anik Das

Student ID: **2025EM1100026**

Submitted To

BITS Pilani Digital

Course: Feature Engineering

Date of Submission

November 4, 2025

Table of Contents

1. Introduction

This report documents my work on the feature engineering assignment using the Ames Housing dataset. The goal was to clean the data, create new features, handle categorical variables, and prepare the dataset for machine learning.

1.1 Dataset Overview

> **Dataset Statistics:**
> • Rows: 1,460 houses
> • Features: 81 (80 features + SalePrice target)
> • Missing Values: 7,829 cells across 19 columns
> • Target Variable: SalePrice (house sale prices)

1.2 Student Random Feature

As per the assignment requirements, I generated a random feature using my student ID:

- **Student ID:** 2025EM1100026
- **Last 7 digits:** 1100026
- **Seed:** 26 (calculated as 1100026 % 1000)
- **Offset:** 4 (calculated as 1100026 % 7)
- **Feature name:** student_random_feature
- **Value range:** 5 to 103 (random integers with offset)

---

2. Data Cleaning

2.1 Missing Value Treatment

I handled missing values based on what they actually mean in the context of housing data. Different features required different strategies.

Strategy 1: "Doesn't Exist" Interpretation

For features where "missing" means "feature doesn't exist":

- Pool quality, garage features, basement features, fence, alley, etc.
- Filled with 'None' for categorical features
- Filled with 0 for numeric features
- **Reasoning:** NA in PoolQC means the house has no pool, not that pool quality is unknown

Strategy 2: Group-Based Imputation

For LotFrontage (street frontage):

- Used median value by neighborhood
- **Reasoning:** Lot sizes vary significantly by area, so using neighborhood-specific median preserves spatial patterns

Strategy 3: Mode Imputation

For truly missing categorical values:

- Used most common category
- Applied to features with only 1-2 missing values
- **Reasoning:** With minimal missing data, mode is a safe approach

Result: Zero missing values after treatment (7,829 → 0)

2.2 Outlier Detection and Treatment

I used the IQR (Interquartile Range) method to detect outliers:

- **Method:** Values beyond Q1 - 1.5×IQR or Q3 + 1.5×IQR
- **Findings:** Found 2 extreme cases with very large living area but unusually low sale prices
- **Decision:** Removed these outliers (likely data entry errors or unusual sale circumstances)

Final dataset: 1,459 rows (removed 1 extreme outlier)

3. Feature Engineering

3.1 Transforming Skewed Features

Many features in the dataset were right-skewed, meaning most values are low with a few very high values. This is common in real estate data.

Features transformed (29 total):

- Area features: LotArea, GrLivArea, TotalBsmtSF, GarageArea, etc.
- Square footage features: BsmtFinSF1, 1stFlrSF, 2ndFlrSF
- Porch features: WoodDeckSF, OpenPorchSF, EnclosedPorch
- Target variable: SalePrice

Why log transformation?

1. Makes distributions more normal (better for most ML algorithms)
2. Reduces impact of extreme values without removing them
3. Captures multiplicative relationships in pricing
4. Standard practice for real estate data where prices are proportional

Target skewness: 1.88 → 0.12 (93% reduction, nearly normal distribution)

3.2 Creating New Features

I created 10 new features based on domain knowledge about what matters for house pricing:

Aggregate Features (3)

1. **TotalSF** = TotalBsmtSF + 1stFlrSF + 2ndFlrSF
   Rationale: Total living space matters more than how it's distributed

2. **TotalBath** = FullBath + (0.5 × HalfBath)
   Rationale: Aggregated bathroom count with half baths weighted appropriately
3. **TotalPorchSF** = Sum of all porch areas
   Rationale: Total outdoor space is more relevant than individual porch types

Temporal Features (2)

1. **HouseAge** = YearSold - YearBuilt
   Rationale: Age at sale is more intuitive and affects depreciation
2. **RemodAge** = YearSold - YearRemodAdd
   Rationale: Time since renovation affects condition and value

Binary Indicator Features (5)

1. **HasPool** = 1 if PoolArea > 0, else 0
2. **HasGarage** = 1 if GarageArea > 0, else 0
3. **Has2ndFloor** = 1 if 2ndFlrSF > 0, else 0
4. **HasBasement** = 1 if TotalBsmtSF > 0, else 0
5. **HasFireplace** = 1 if Fireplaces > 0, else 0

Rationale for binary features: Presence/absence of amenities has pricing impact beyond just size

3.3 Text-Based Composite Features

I created 3 composite features by combining related categorical variables. This captures how properties are naturally described.

1. property_location_type

- **Combination:** MSZoning + Neighborhood + Condition1
- **Example:** "RL_CollgCr_Norm" = Residential Low Density property in College Creek neighborhood with Normal street conditions
- **Unique combinations:** 94

- **Rationale:** Location + context matters together (residential in downtown ≠ residential in suburbs)

2. property_architecture

- **Combination:** BldgType + HouseStyle + RoofStyle
- **Example:** "1Fam_2Story_Gable" = Single-family, two-story house with gable roof
- **Unique combinations:** 55
- **Rationale:** Architectural style is defined by the combination of these elements

3. property_exterior

- **Combination:** Exterior1st + Exterior2nd + Foundation
- **Example:** "VinylSd_VinylSd_PConc" = Vinyl siding with poured concrete foundation
- **Unique combinations:** 124
- **Rationale:** Material combinations indicate construction quality and era

Encoding Decision: Used Label Encoding (not one-hot) because of high cardinality (55-124 combinations each). One-hot encoding would have created 273 additional columns.

4. Categorical Encoding

I used three different encoding strategies based on the nature of each categorical feature.

4.1 Ordinal Encoding

Used for: Features with natural order (quality/condition ratings)

Mapping applied:

- **Quality scale:** None(0) < Poor(1) < Fair(2) < Typical(3) < Good(4) < Excellent(5)
- **Features encoded:** ExterQual, ExterCond, BsmtQual, BsmtCond, HeatingQC, KitchenQual, FireplaceQu, GarageQual, GarageCond, PoolQC (10 features)
- **Additional scales:** Exposure (0-4), Basement Finish (0-6), Functional (1-8)

Total ordinal encoded: 14 features

4.2 One-Hot Encoding

Used for: Nominal categorical features (no inherent order)

Features encoded:

- Neighborhood (25 categories)
- MSZoning (5 categories)
- BldgType, HouseStyle, RoofStyle, RoofMatl
- Exterior1st, Exterior2nd, Foundation
- Heating, Electrical, GarageType
- SaleType, SaleCondition
- And others...

Result: 24 original categorical features → 176 binary indicator columns

4.3 Label Encoding

Used for:

1. Binary categorical features (e.g., Street: Paved/Gravel, CentralAir: Y/N)
2. The 3 text-based composite features (high cardinality)

Final feature count: 81 original → 220 engineered features

5. Dimensionality Reduction (PCA)

With 220 features, there's significant risk of overfitting and many features are correlated with each other. I applied Principal Component Analysis (PCA) to address this.

5.1 Why PCA?

1. **Reduce dimensionality:** 220 features is too many for 1,459 observations
2. **Eliminate multicollinearity:** Many features are highly correlated
3. **Preserve information:** Keep most of the variance in the data
4. **Improve computational efficiency:** Faster model training
5. **Reduce overfitting risk:** Fewer features means less chance of fitting to noise

5.2 PCA Configuration

- **Variance retention threshold:** 95%
- **Reasoning:** Industry standard that balances information preservation with dimensionality reduction
- **Result:** 220 features → 136 principal components
- **Dimensionality reduction:** 38.2%

5.3 Student Random Feature in PCA

Assignment Question: Did the student_random_feature load significantly on any principal component?

Answer: No, it did not.

- **Maximum loading:** 0.043 (on PC84)
- **Mean absolute loading:** 0.018 (near zero)
- **Top 5 PC loadings:** All below 0.05

Explanation:

PCA identifies structured variance in the data - patterns that explain how features vary together. Random features, by definition, have no systematic patterns or correlations. They represent pure noise.

The fact that student_random_feature shows negligible loading confirms two things:

1. The feature is truly random (as intended)
2. PCA successfully distinguishes signal from noise

Conclusion: Random feature behaves as expected - no significant patterns detected

6. Exploratory Analysis Results

6.1 Correlation with SalePrice

Top 5 features most correlated with house prices:

1. **OverallQual:** 0.79 (overall material and finish quality)
2. **GrLivArea:** 0.71 (above grade living area)
3. **GarageCars:** 0.64 (garage capacity)
4. **GarageArea:** 0.62 (garage size)
5. **TotalBsmtSF:** 0.61 (basement area)

student_random_feature correlation with SalePrice: -0.006

This is essentially zero, as expected for a random feature.

6.2 Student Random Feature Correlations

Assignment Question: Which 3 features are most correlated with your random feature? Why does this occur?

Answer:

Top 3 correlations:

1. **LowQualFinSF:** 0.046
2. **KitchenAbvGr:** 0.036
3. **MSSubClass:** 0.034

Why these tiny correlations occur:

These correlations are **extremely weak** (all < 0.05) and represent **random statistical noise**, not real relationships.

Statistical Explanation:

- With 38 numeric features, **purely by chance** we'd expect 2-3 features to show small correlations ($|r| > 0.03$) with any random variable
- This is a consequence of **spurious correlation** - when you test many pairs, some will appear related just by coincidence
- A correlation of 0.046 explains only 0.2% of variance ($r^2 = 0.002$) - essentially meaningless

Conclusion: The absence of any strong correlations validates that the feature is truly random and not inadvertently capturing real housing patterns.

7. Final Datasets

The feature engineering pipeline produced two ready-to-use datasets, each suited for different modeling approaches.

7.1 processed_data_engineered.csv

> **Dataset Specifications:**
> • Size: 1,459 rows × 220 columns
> • Content: All engineered features + one-hot encoded variables
> • Target: SalePrice_Log (log-transformed)
> • Missing values: 0
> • All features: Numeric

Use cases:

- Linear regression models (can see explicit feature coefficients)
- Interpretable models where feature importance matters
- When you need to understand which original features drive predictions

7.2 processed_data_pca.csv

> **Dataset Specifications:**
> • Size: 1,459 rows × 137 columns (136 PCA components + target)
> • Content: Principal components retaining 95% variance
> • Target: SalePrice_Log (log-transformed)
> • Variance retained: 95.0%
> • Dimensionality reduction: 38.2%

Use cases:

- Models sensitive to multicollinearity (Ridge/Lasso regression)

- High-dimensional models (faster training with 136 vs 220 features)
- When interpretability of individual features is less important
- Neural networks (reduced input dimensionality)

7.3 Common Properties

Both datasets share these characteristics:

- Zero missing values (100% complete)
- Log-transformed target variable (normalized distribution)
- Standardized features (mean=0, standard deviation=1)
- Same number of observations (1,459 rows)
- Ready for immediate use in machine learning models

8. Summary

8.1 What Was Accomplished

Data Cleaning

- Handled 7,829 missing values using context-based strategies
- Removed 2 extreme outliers
- Achieved 100% data completeness without compromising data integrity

Feature Engineering

- Applied log transformation to 29 skewed features
- Created 10 new numeric features based on domain knowledge
- Developed 3 composite text-based features
- Reduced target variable skewness from 1.88 to 0.12 (93% improvement)

Encoding

- Ordinal encoding: 14 features (preserving natural order)
- One-hot encoding: 24 features → 176 binary columns
- Label encoding: 3 binary features + 3 text composites
- Total transformation: 81 → 220 features

Dimensionality Reduction

- Applied PCA: 220 → 136 components (38% reduction)
- Retained 95% of original variance
- Eliminated multicollinearity through orthogonal components

Student Feature Analysis

- Generated student_random_feature (ID: 1100026, seed: 26, offset: 4)
- Correlation with SalePrice: -0.006 (negligible)
- Maximum PCA loading: 0.043 (no significant pattern)
- Confirms random nature and validates analysis methodology

8.2 Key Learnings

1. **Context matters in data cleaning** - Missing values require different strategies based on their meaning
2. **Domain knowledge is essential** - Real estate principles guided effective feature creation
3. **Multiple encoding strategies needed** - Different feature types require different approaches
4. **Dimensionality reduction is valuable** - PCA reduced complexity while preserving information
5. **Random features don't survive proper analysis** - Validates that correlation and PCA work correctly

8.3 Files Submitted

1. **FE_assignment.ipynb** - Complete Jupyter notebook with code, outputs, and analysis
2. **processed_data_engineered.csv** - Dataset with 220 engineered features
3. **processed_data_pca.csv** - Dataset with 136 PCA components
4. **This report** - Summary and documentation of all work

Student Declaration

I confirm that this work represents my own efforts and understanding. All analysis, decisions, and implementations were completed based on the assignment requirements and my knowledge of feature engineering principles.

Anik Das
Student ID: 2025EM1100026
Date: November 4, 2025