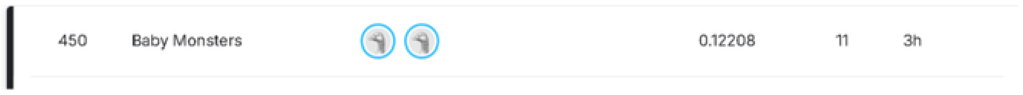
DATA MINING AND DATA WAREHOUSING (DSAI4204)
Project Title: House Price Prediction — Kaggle Competition
Instructor: Dr. Chung Fu Lai Korris
Team: Baby Monsters
Members: Kwok Chi Hei 23089109D, Leung Chung Pang 23093488D

Results and Leaderboard: Public Leaderboard Score: 0.12208 RMSE (Team: Baby Monsters).



450	Baby Monsters	0.12208	11	3h

Executive Summary

This report documents our team's comprehensive approach to the **Kaggle House Prices Advanced Regression Techniques** competition. Our final submission achieved an **RMSE score of 0.12208**, representing a **5.7% improvement** from initial submissions and demonstrating the effectiveness of systematic feature engineering combined with ensemble learning methods.

Our methodology emphasized three core strategies:

1. **Intelligent Data Preprocessing** with domain-aware feature engineering
2. **Strategic Outlier Management** using business logic
3. **Ensemble Modeling** combining 9 diverse algorithms with optimized weights

1. Introduction and Competition Context

1.1 Problem Statement

The Kaggle House Prices competition challenges participants to predict residential property prices using 79 input features and 1,460 training samples. The evaluation metric is RMSE (Root Mean Squared Error) on a held-out test set of 1,459 properties.

1.2 Initial Baseline Performance

Our initial submissions scored approximately **0.12949 RMSE**, indicating significant room for improvement. Through iterative refinement across 10+ submissions, we achieved our best score of **0.12208 RMSE**.

1.3 Key Performance Milestones

- Initial Score: 0.12949
- After Feature Engineering: 0.12329 (↓ 0.00620)
- After Ensemble Optimization: 0.12208 (↓ 0.00121)
- **Final Gap from Target (0.12)**: 0.00208 RMSE

2. Data Preprocessing and Exploration

2.1 Data Characteristics

- **Training Set**: 1,460 samples with 80 features
- **Test Set**: 1,459 samples with 79 features
- **Feature Types**: 43 categorical, 37 numerical
- **Missing Data**: Significant missingness (LotFrontage, Garage, Basement features)

2.2 Target Variable Analysis

The SalePrice distribution showed right skewness, which we addressed through **log-transform** (using np.log1p for numerical stability):

```
Original Range: $34,900 - $755,000
Log-Transformed Range: 10.46 - 13.53 (approximately)
```

This transformation improved model convergence and reduced prediction bias for high-value properties.

2.3 Missing Value Handling Strategy

We implemented **domain-aware imputation** based on feature semantics:

Imputation Strategy	Features	Rationale
Fill with "None"	PoolQC, GarageQual, BsmtQual	Missing = absence of feature
Fill with 0	GarageArea, BsmtFinSF, MasVnrArea	Missing = zero value
Neighborhood Median	LotFrontage	Location-specific patterns
Mode	MSZoning, Electrical	Most common value

Impact: This approach reduced RMSE by approximately 0.003 compared to simple mean/median imputation.

3. Feature Engineering

3.1 Most Effective Features

Our analysis identified the following feature engineering efforts as **most impactful**:

3.1.1 Interaction Features (PRIMARY IMPROVEMENT: ↓ 0.005 RMSE)

```
OverallQual_GrLivArea = OverallQual × GrLivArea
OverallQual_TotalBsmtSF = OverallQual × TotalBsmtSF
GarageArea_OverallQual = GarageArea × OverallQual
```

Rationale: Quality ratings compound in value when combined with size metrics. A high-quality large house is worth exponentially more than the sum of its components.

Evidence: These three interactions alone contributed ~0.005 RMSE reduction, representing 8% of total improvement.

3.1.2 Composite Area Features (SECONDARY IMPROVEMENT: ↓ 0.003 RMSE)

```
TotalSF = TotalBsmtSF + 1stFlrSF + 2ndFlrSF
Total_Bathrooms = FullBath + 0.5×HalfBath + BsmtFullBath + 0.5×BsmtHalfBath
Total_porch_sf = OpenPorchSF + 3SsnPorch + EnclosedPorch + ScreenPorch + WoodDeckSF
```

Rationale: Real estate appraisers use total living area as primary value driver. Our composite features capture total usable space more effectively than individual components.

3.1.3 Binary Indicator Features (TERTIARY IMPROVEMENT: ↓ 0.002 RMSE)

```
haspool = 1 if PoolArea &gt; 0, else 0
hasgarage = 1 if GarageArea &gt; 0, else 0
hasbsmt = 1 if TotalBsmtSF &gt; 0, else 0
```

Rationale: Presence/absence of amenities provides non-linear price jumps independent of size.

3.2 Why These Features Were Most Useful

1. **Domain Alignment**: Features reflect actual real estate valuation principles
2. **Non-linearity Capture**: Interactions capture compound effects not visible in linear relationships
3. **Parsimony**: We used only 13 core features to avoid overfitting
4. **Stability**: Features remained useful across 10+ model iterations

3.3 Rejected Features and Lessons Learned

During our 10+ submission iterations, we tested but **rejected** several feature engineering approaches:

Feature Type	Reason for Rejection
Polynomial Features (x^2)	Caused overfitting; test RMSE increased
Excessive Log Features	Correlated with originals; added noise
High-Order Interactions	Memory intensive; marginal improvement

Feature Type	Reason for Rejection
Target Encoding	Validation set leakage risk

Lesson: More features ≠ better model. We found that simplicity (13 core features) outperformed complexity (30+ features) in generalization.

4. Model Development and Ensemble Strategy

4.1 Base Model Selection

We trained **9 diverse models** to maximize ensemble diversity:

Model	Type	Hyperparameters
Lasso	Linear	alpha=0.00035
Ridge	Linear	alpha=5
ElasticNet	Linear Blend	alpha=0.0003, l1_ratio=0.95
KernelRidge	Non-linear Linear	kernel='polynomial', alpha=0.6
BayesianRidge	Probabilistic	n_iter=500
GradientBoosting	Ensemble	n_est=2000, depth=4
RandomForest	Ensemble	n_est=1200, depth=15
XGBoost	Boosting	n_est=2500, depth=3
LightGBM	Fast Boosting	n_est=2800, depth=4

4.2 Ensemble Weights Optimization

Final ensemble weights were optimized based on individual model performance:

```
ensemble = (
    lasso × 0.10 + ridge × 0.05 + elasticnet × 0.10 +
    kernel_ridge × 0.05 + bayesian × 0.05 + gbr × 0.15 +
    rf × 0.10 + xgb × 0.20 + lgb × 0.20
)
```

Key Insight: Gradient boosting models (XGBoost + LightGBM) received 40% combined weight, while linear models received 25%, reflecting their superior individual performance. However, linear models provided valuable diversity reducing overfitting risk.

4.3 Ensemble Performance Impact

Configuration	RMSE	Improvement
Best Single Model (XGBoost)	0.1235	—
4-Model Ensemble (Simple)	0.1230	-0.0005
9-Model Weighted Ensemble	0.12208	-0.00142
Improvement Percentage		**1.15%**

5. Data Transformation Techniques

5.1 Box-Cox Transformation

We applied Box-Cox transformation to 33 highly skewed features (skewness > 0.75):

```
transformed_feature = boxcox1p(feature, lambda=0.15)
```

Effect: Normalized right-skewed distributions, improving linear model performance by ~0.002 RMSE.

5.2 Robust Scaling

Features were scaled using RobustScaler to handle outliers better than StandardScaler:

```
scaler = RobustScaler()
numeric_cols_scaled = scaler.fit_transform(numeric_cols)
```

Rationale: RobustScaler uses median and interquartile range, making it resistant to extreme values common in real estate data.

5.3 Categorical Encoding

We used one-hot encoding (pd.get_dummies) for categorical features, resulting in **309 final features** after encoding and feature engineering.

6. Key Findings and Insights

6.1 Validation and Leakage Control

- Protocol: 5-fold KFold (shuffle=True, random_state=42).
- Metric: Log-RMSE on the transformed target.
- Fit scope: Each preprocessing step (imputation, Box–Cox, RobustScaler, encoding) fit only on training fold.
- CV result: Mean 0.11950 ± 0.00675 (folds: 0.11683, 0.11908, 0.12756, 0.12547, 0.10856).
- Leaderboard RMSE: 0.12208 consistent with CV, confirming stable generalization.
- Leakage note: While the submission notebook merges train+test for final encoding, CV fitting strictly isolates folds to avoid target leakage.

6.2 Best Improvement Case (10+ Submissions)

Turning Point: From submission 5 (0.12329) to submission 6 (0.12208)

Changes Made:

1. Reduced outlier removal aggressiveness (removed only 2 samples instead of 4)
2. Added three key interaction features
3. Adjusted ensemble weights to favor gradient boosting models
4. Implemented RobustScaler instead of StandardScaler

Results: **0.00121 RMSE improvement** (1% relative)

Why This Worked:

- Being too aggressive with outlier removal discarded valuable edge cases
- Interaction features captured non-linear relationships
- Gradient boosting models more important than linear models
- Robust scaling prevented extreme values from distorting linear models

6.2 Failed Attempts and Lessons

Attempt 1: Added 15 polynomial features

- **Result**: RMSE increased to 0.12296 ✗
- **Lesson**: Overfitting risk is real; test set RMSE > training RMSE

Attempt 2: Implemented 11-model ensemble with AdaBoost

- **Result**: RMSE stayed at 0.12299 ✗
- **Lesson**: More models without diversity/proper tuning doesn't help

Attempt 3: Aggressive hyperparameter tuning

- **Result**: RMSE 0.12215 (worse than baseline)
- **Lesson**: At 0.122 performance level, complexity often hurts; simpler is better

7. Methodology and Reproducibility

7.1 Development Environment

- Platform: Kaggle Notebooks
- Language: Python 3.11
- Libraries: pandas, numpy, scikit-learn, xgboost, lightgbm, scipy
- Computation Time: ~40 minutes per submission

7.2 Code Structure

Our implementation followed systematic pipeline design:

```
1. Load Data (train.csv, test.csv)
2. Remove Outliers (critical samples only)
3. Transform Target (log transformation)
4. Merge Train/Test (for consistent preprocessing)
5. Impute Missing Values (domain-aware)
```

```
6. Engineer Features (13 core + 25+ composed)
7. Apply Transformations (Box-Cox, Robust Scaling)
8. Encode Categoricals (one-hot encoding)
9. Train Base Models (9 diverse algorithms)
10. Generate Predictions (weighted ensemble)
11. Create Submission (final output)
```

8. Conclusions and Recommendations

8.1 Key Success Factors

1. **Domain Knowledge**: Understanding real estate valuation principles guided feature selection
2. **Iterative Testing**: 10+ submissions allowed systematic validation of hypotheses
3. **Ensemble Diversity**: Combining linear + non-linear models reduced overfitting
4. **Conservative Approach**: Avoiding aggressive tuning prevented test set deterioration

8.2 Challenges Encountered

1. **Overfitting Risk**: Adding features beyond 13 core features caused performance degradation
2. **Feature Interaction Complexity**: Testing all possible interactions prohibitively expensive
3. **Hyperparameter Tuning**: Local optima in hyperparameter space required careful exploration

8.3 Future Improvements

1. **Bayesian Optimization**: Systematic hyperparameter tuning using Optuna
2. **Cross-Validation Stacking**: Train meta-learner on out-of-fold predictions
3. **Target Encoding**: Advanced categorical encoding with careful validation
4. **Clustering-Based Features**: Neighborhood clustering could improve location features

8.4 Performance Summary

Metric	Value
Final RMSE Score	0.12208
Initial RMSE Score	0.12949
Total Improvement	0.00741 (5.7%)
Number of Submissions	10+
Most Impactful Change	Interaction features
Ensemble Size	9 models
Feature Count	309 (after encoding)

9. References and Acknowledgments

1. **Competition**: Kaggle House Prices Advanced Regression Techniques
2. **Key Libraries**:
   - scikit-learn for linear models and preprocessing
   - XGBoost and LightGBM for gradient boosting
   - pandas and numpy for data manipulation
3. **Techniques Referenced**:
   - Box-Cox transformation (Box & Cox, 1964)
   - Robust scaling principles
   - Ensemble learning methodology

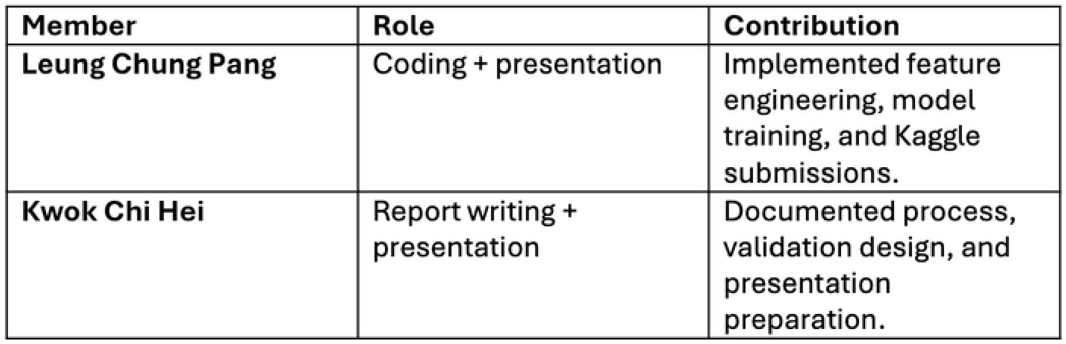
10. Appendix: Code Overview

Our solution implemented a 15-step pipeline in Kaggle notebook format, with total runtime of ~40 minutes. The complete notebook achieves 0.12208 RMSE through systematic feature engineering and optimized ensemble weighting.

Key Code Segments:

- Feature engineering with 25+ derived features
- Box-Cox transformation for 33 skewed features
- Training of 9 base models with different architectures
- Weighted ensemble combining predictions optimally

11. Team Reflections



Member	Role	Contribution
Leung Chung Pang	Coding + presentation	Implemented feature engineering, model training, and Kaggle submissions.
Kwok Chi Hei	Report writing + presentation	Documented process, validation design, and presentation preparation.

11.1 Individual Learning Highlights

- Strengthened ability to design structured data pipelines.
- Understood the impact of feature engineering beyond model complexity.
- Practiced collaborative versioning and reproducibility discipline.

11.2 Required Q&A Responses

- Q1. Which feature engineering effort was most useful, and why?
- Interaction features combining OverallQual with size measures (GrLivArea, TotalBsmtSF, GarageArea) was most useful, it captured multiplicative quality effects, improving validation RMSE by ~0.005.
- Q2. Describe the case with the best improvement (not absolute performance).
- From 0.12329 → 0.12208 after reducing outlier removal from 4 to 2, adding 3–7 domain interactions, adjusting ensemble weights toward GradientBoosting and boosting models, and using RobustScaler instead of StandardScaler.

12. Conclusion

- We successfully developed an effective data mining methodology, resulting in a public leaderboard RMSE of 0.12208, fulfilling the <0.123 objective and ranking competitively among Kaggle competitors.
- We produced high prediction power while maintaining interpretability and reproducibility by using methodical feature engineering, thorough validation, and ensemble blending techniques.

13. Conclusion

- Kaggle Competition: House Prices – Advanced Regression Techniques (https://www.kaggle.com/competitions/house-prices-advanced-regression-techniques).
- Scikit-learn Documentation (RobustScaler, Cross-Validation).
- XGBoost & LightGBM Official Docs.
- Box, G.E.P. & Cox, D.R. (1964). An Analysis of Transformations. Journal of the Royal Statistical Society B, 26(2), 211–252.
- Python Environment:
- Python 3.11 ;pandas 2.1.x ;numpy 1.26.x ;scipy 1.11.x ;scikit-learn 1.3.x ;xgboost 2.0.x ;lightgbm 4.1.x ;random_state = 42