AirBnB Dataset

Section 2.3.4 AirBnB Dataset

Missing Values: The dataset exhibited substantial missingness across multiple columns. Review-related features (review scores, reviews_per_month) showed approximately 25% missing values, while optional fields such as security_deposit (72%) and cleaning_fee (25%) had considerably higher rates. To preserve information about missingness patterns, we created binary indicator features (e.g., has_security_deposit, no_reviews_yet) before imputation. Numeric features were filled with either 0 (for fees/deposits) or median values (for review scores and property characteristics), while binary features were imputed using mode values.

Encoding: This dataset required extensive categorical handling due to its heterogeneous structure. Boolean variables: 27 binary host verification fields (e.g., host_verifications_email, host_verifications_phone) were converted from string representations ('t'/'f') to numeric (0/1). Low-cardinality categoricals: One-hot encoding was applied to room_type, bed_type, cancellation_policy, and state (after normalization to consolidate spelling variations of Victoria, NSW, and Queensland). High-cardinality features: The property_type column (143 unique values) was encoded using frequency encoding to avoid excessive dimensionality explosion. Target encoding: The zipcode feature was target-encoded using the numeric price label with smoothing parameters to prevent overfitting. List-type features: Amenities (369 unique values) and host verifications were parsed from string representations and multi-label binarized, creating 369 and 24 binary columns respectively. Text features: Eleven text columns (name, summary, description, etc.) were combined and vectorized using sentence-transformers (all-MiniLM-L6-v2 model) to generate 384-dimensional embeddings per listing.

Scaling: StandardScaler was applied exclusively to numeric features within each cross-validation fold. Text embeddings, being pre-normalized by the transformer model, were not scaled. To prevent data leakage, the scaler was fitted only on training splits and then applied to both training and validation data.

Outliers: Outlier analysis revealed extreme values in several numeric columns, particularly in pricing-related features and review scores. However, these were retained as they represent legitimate market segments (e.g., luxury properties with ultra-high prices, or problematic listings with very low review scores). Removing such cases would have artificially constrained the model's ability to learn the full spectrum of price categories, especially the "ultra luxury" class.

Section 4.1 Efficacy

AirBnB	Precision	N/A	N/A	N/A
	Recall	N/A	N/A	N/A
	F1	0.308	0.316	0.358

Section 4.2 Efficiency

AirBnB	18,316	9s	71s	44s

Section 5.4 AirBnB

The Melbourne Airbnb dataset involved predicting price categories for 18,316 listings across 10 ordered classes from "very low" to "ultra luxury." The severe class imbalance (ratio of 0.030) and integration of diverse data types—numeric, categorical, text, and list-structured features—made this our most challenging task.

XGBoost significantly outperformed linear models, achieving 42.22% accuracy compared to Logistic Regression's 33.92% and SVM's 34.66%. This 8-percentage-point gap reveals that Airbnb pricing involves complex feature interactions that tree-based methods capture effectively while linear models cannot. F1-macro scores reflected the same pattern, with XGBoost reaching 0.358 versus ~0.31 for linear classifiers.

Surprisingly, adding full 384-dimensional text embeddings decreased XGBoost's accuracy to 41.05% (-2.78%), though Logistic Regression improved slightly (+0.40%). This suggests text descriptions introduce noise when predicting discretized price bins. However, PCA compression to 50 dimensions (64% variance retained) recovered performance to 42.06% while reducing training time from 164 to 34 seconds—demonstrating that dimensionality reduction filters signal from noise effectively.

Five-fold cross-validation confirmed these findings with remarkable stability: XGBoost achieved 41.80% ± 0.34%, nearly identical to the 42.22% holdout result. Hyperparameter tuning via GridSearchCV yielded 41.61% accuracy, slightly below baseline, indicating default parameters were already near-optimal. The 42% ceiling reflects fundamental challenges: discretizing continuous prices into 10 bins obscures natural groupings, and temporal factors like seasonality aren't captured in our static features.

Computational efficiency varied dramatically: Logistic Regression trained in 9 seconds, XGBoost in 44 seconds, while SVM exceeded 30 minutes with full features. For production deployment, XGBoost offers the best accuracy-speed balance.

The key lesson: more features don't guarantee better performance. Adding 384 text dimensions hurt XGBoost's accuracy, emphasizing careful feature evaluation over indiscriminate inclusion. Tree-based methods' success confirms real-world pricing involves complex non-linear interactions, while holdout-CV consistency validates our experimental methodology.