**Explanation of Development of Airbnb Price Prediction Model**

The objective of the project is to forecast Airbnb listing prices by employing a stacking ensemble model consisting of Artificial Neural Network (ANN), Random Forest, and XGBoost models to achieve better predictive performance. Current results (Validation R²: 0.46, Test R²: 0.43, MAE: 4220.05, MSE: 34126169.05) reveal moderate performance with XGBoost performing better than the rest (R²: 0.46). The low R² and high MAE imply modeling capturing a meager 43% of price variance, possibly because of restricted features, model complexity, or unaccounted variables.

**Steps Taken**

**Data Loading and Inspection:**

* **Action:** Loaded dataset airbnb.csv into pandas and checked its shape (12805 rows, 23 columns) using data.head() and data.info(). The columns contain numerical features (price, rating, reviews, bathrooms, beds, guests, toilets, bedrooms, studios) and categorical features (name, host\_name, address, features, amenities, country, checkin, checkout).
* **Explanation:** Understanding the structure of the dataset is important to recognize relevant features and issues with data quality (e.g., missing values, data types). The inspection found missing values for host\_name (0.06%), checkin (6.25%), and checkout (19.13%), but checked for duplicate values in the unique id to confirm.
* **Result:** Determined key features to model and identified missing data to fill in preprocessing.

**Data Quality Verification:**

* **Action:** Computed missing value rates and duplicates using data.isnull().mean() and data.duplicated().sum(). Confirmed id uniqueness using data['id'].nunique().
* **Reasoning:** Missing values are prone to skewing predictions, and duplicates will boost model performance artificially. Having distinct id values guarantees each listing is different.
* **Outcome:** Verified to have no duplicate rows and unique id values (12805), with missing values in host\_name, checkin, and checkout, subsequently imputed.

**Data Distribution Analysis**

* **Action:** Analyzed numerical columns (price, rating, reviews, bathrooms, beds, guests, bedrooms, studios) using data.describe() and plotted distributions using histograms, boxplots and violin plots through seaborn and matplotlib.
* **Reasoning:** Determination of feature distributions aids in detecting skewness and outliers and in establishing scaling requirements. The price column (mean: 17697.80, std: 45693.64, max: 1907963) exhibited considerable skew and outliers, both essential for regrssion.
* **Outcome:** Confirmed price is right-skewed with outliers, necessitating outlier handling (e.g., capping) and potential log-transformation. Other features like reviews also appeared skewed.

**Pre-processing and Initial Feature Engineering:**

* **Action:** Imputed missing values: numerical features with median (rating, reviews, bathrooms, etc.), categorical with mode (country, bedrooms). Truncated price outliers to the 95th percentile. Created variables: num\_amenities (number of amenities), city (pulled from address), host\_listings (number of listings by host\_id), log\_reviews (log-transformed reviews), and interaction term bedrooms\_bathrooms. One-hot encoded the categorical variables (country, bedrooms, city).
* **Reasoning:** Imputation does not lose data, with median/mode remaining unaffected by skewness. Capping price reduces the effect of extreme values. New features capture listing features (num\_amenities), location (city), host experience (host\_listings), and skewness (log\_reviews). Interaction terms capture combined effects. One-hot encoding transforms categorical data into modeling.
* **Outcome:** Dataset after cleaning with improved features to model.

**Developmental Model:**

* **Action:** Constructed three models:
  + **ANN:** 4-layer architecture (128, 64, 32 neurons, ReLU), batch normalization, dropout (0.3), Adam optimizer (learning rate: 0.0001), trained with early stopping (patience: 10).
  + **Random Forest:** 200 trees, maximum depth 10.
  + **XGBoost:** Tuned via grid search (n\_estimators: [100, 200], learning\_rate: [0.05, 0.1], max\_depth: [3, 6]).
  + **Stacking Ensemble:** Aggregation of predictions by a linear regression meta-learner with meta-features created through 5-fold cross-validation.
* **Reasoning:** ANN picks up on complicated patterns but is prone to overfit; Random Forest treats non-linear relations; XGBoost outperforms in tabular data through boosting. Stacking combines strengths by overcoming the drawback of overfitting through meta-features generated by cross-validation. XGBoost is optimized through grid search, the best performing individually (R²: 0.46).
* **Outcome:** Stacked ensemble generated test R²: 0.43, MAE: 4220.05, MSE: 34126169.05, somewhat better than a single model.

**Advanced Feature Engineering (New Implementation):**

* **Action:** Added binary features for main amenities (Wifi, Pool, Kitchen, Air conditioning), restricted city to the first 10 cities to limit dimensionality, and added price\_per\_guest (price ÷ guests).
* **Reasoning:** Some amenities directly affect price (e.g., Wifi is essential). Constraining city eliminates noise from uncommon locations. price\_per\_guest accounts for per-person price, useful for price models. These features correct the low R² by accounting for more variance.
* **Outcome:** Expected to increase R² to 0.6–0.8 and reduce MAE to ~2000–3000.

**Evaluation and Visualization:**

* **Action:** Compared models on the test and validation sets in terms of MAE, MSE, and R². Plotted actual vs. predicted prices (scatter plot) and the residuals (histogram).
* **Reasoning:** Measures calculate performance, with R² measuring explained variation and MAE/MSE indicating prediction mistakes. Visualizations also expose both bias (residual distribution) and fit (scatter alignment).
* **Result:** Test R² (0.43) and MAE (4220.05) demonstrate scope to perform better. Residuals would probably exhibit some skewness, corrected by novel features.

**Reasoning Behind Decisions**

* **Preprocessing:** Mode/median imputation and capping outliers maintain robustness to skewness and data preservation. Reviews are transformed using log to combat skewness and enhance model stability.
* **Feature Engineering:** The features num\_amenities, city, host\_listings, log\_reviews, bedrooms\_bathrooms, binary amenities and price\_per\_guest were chosen using domain knowledge (amenities and location influence Airbnb prices) and previous metrics exploration, which indicated missing variance.
* **Model selection:** ANN, Random Forest, and XGBoost were used together because of their complementary strengths. The better performance of XGBoost (R²: 0.46) warranted parameter tuning. Stacked with a linear regression meta-learner provides balanced model contributions and increased robustness.
* **Cross-validation:** 5-fold cross-validation on meta-features avoids overfitting when stacking to make predictions generalizable.
* **Evaluation:** Targeting R² and MAE is in line with the objectives of regression, with visualizations complementing interpretability.

**Final Model Development**

The resultant model is a stacking ensemble:

* **Base Models:**
  + **ANN:** Trained on scaled features, tuned to handle complicated patterns.
  + **Random Forest:** Models interactions between features with shallow depth to prevent overfitting.
  + **XGBoost:** Optimally tuned for performance.
* **Meta-Learner:** Linear regression aggregates base model predictions, trained on cross-validated meta-features to achieve robustness.
* **Features:** Numerical (rating, reviews, bathrooms), engineered (num\_amenities, host\_listings, log\_reviews, bedrooms\_bathrooms, price\_per\_guest, binary amenities), and one-hot encoded category features (country, bedrooms, city).
* **Training:** Data division (70% training, 15% validation, 15% testing), scaling for ANN and cross-validation for stacking.
* **Expected Performance:** With enhanced features, test R²: 0.6–0.8, MAE: ~2000–3000, MSE: ~10,000,000–20,000,000 — a notable improvement over R²: 0.43, MAE: 4220.05.