Part A: Airbnb Price Prediction and Insights

1. Project Overview

Airbnb is an online marketplace that connects people looking to rent out their homes with people looking for accommodations. One of the critical components of Airbnb's success is the ability to price listings appropriately. This project focuses on predicting the price of a listing using machine learning (regression).

Objective: Build a regression model that predicts the price of an Airbnb listing using features such as property type, room type, host information, location, and amenities.

Business Importance:

1. Helps hosts set competitive prices.
2. Increases revenue and occupancy.
3. Enhances Airbnb's automated pricing engine.

Task 1: Data Exploration and Preprocessing

Import Libraries

We start by importing essential Python libraries like pandas for data manipulation, NumPy for numeric operations, matplotlib and seaborn for visualizations, and scikit-learn for machine learning and evaluation.

CODE :

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model_selection import train_test_split
```

Step 3: Create `price` Column

```
df['price'] = np.exp(df['log_price'])
```

Insight : Converts logarithmic prices to real prices for interpretability and modeling.

Step 4: Missing Value Handling

```
df['review_scores_rating'].fillna(df['review_scores_rating'].median(), inplace=True)

df['bedrooms'].fillna(df['bedrooms'].median(), inplace=True)

df['bathrooms'].fillna(df['bathrooms'].median(), inplace=True)

df['cleaning_fee'].fillna('$0.00', inplace=True)
```

Insight : Numerical columns are filled with median to handle skewed distributions. Categorical fields use mode or $0.00 where appropriate.

Step 5: Boolean Conversion

```
df['instant_bookable'] = df['instant_bookable'].map({'t': 1, 'f': 0})

df['host_has_profile_pic'] = df['host_has_profile_pic'].map({'t': 1, 'f': 0})

df['host_identity_verified'] = df['host_identity_verified'].map({'t': 1, 'f': 0})
```

Insight : Boolean fields are converted to numeric binary (1/0) for compatibility with ML models.

Step 6: Feature Engineering

```
# Convert host_since to host_year

df['host_since'] = pd.to_datetime(df['host_since'], errors='coerce')

df['host_year'] = df['host_since'].dt.year


# Clean host_response_rate

df['host_response_rate'] = df['host_response_rate'].astype(str).str.replace('%', '')
```

```
df['host_response_rate'] = df['host_response_rate'].replace('nan', np.nan).astype(float)

df['host_response_rate'].fillna(df['host_response_rate'].median(), inplace=True)


# Handle amenities

df['amenities_count'] = df['amenities'].apply(lambda x: len(str(x).split(',')))

df.drop('amenities', axis=1, inplace=True)
```

Insight : We derive new features that might capture important pricing patterns, like the host’s experience and listing richness (amenities).

Step 7: Drop Irrelevant Columns

```
columns_to_drop = ['id', 'log_price', 'name', 'description', 'thumbnail_url']

df.drop(columns=[col for col in columns_to_drop if col in df.columns], inplace=True)

df.drop('host_since', axis=1, inplace=True)
```

Insight : These columns are either redundant or irrelevant for prediction.

Step 8: Categorical Encoding

```
cat_cols = ['property_type', 'room_type', 'cancellation_policy',
            'bed_type', 'city', 'neighbourhood', 'zipcode']

df = pd.get_dummies(df, columns=[col for col in cat_cols if col in df.columns], drop_first=True)
```

Insight : One-hot encoding is used for categorical variables to prepare them for machine learning models.

Step 9: Split the Dataset

```
X = df.drop('price', axis=1)

y = df['price']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Insight : We split the dataset into training and testing parts to train and validate the model properly.

Insights:

1. **Listings with more amenities are priced higher.**
2. **Verified hosts with profile pictures generally set higher prices.**

3. **Feature engineering improved the quality and usability of the dataset.**
4. **KDE and boxplots helped visualize price distributions and detect outliers.**

Task 2: Regression Model Development

Model Used: Random Forest Regressor

```
from sklearn.ensemble import RandomForestRegressor

model = RandomForestRegressor(random_state=42)

model.fit(X_train, y_train)

y_pred = model.predict(X_test)
```

Random Forest

1. Works well with mixed data types.
2. Captures non-linear relationships.
3. Automatically handles feature importance.
4. More accurate than linear models on this dataset.

Feature Importance Example:

```
importances = model.feature_importances_

important_features = pd.Series(importances, index=X.columns).sort_values(ascending=False)

print(important_features.head(10))
```

Helps understand which features (e.g., city, room_type) influence price the most

Task 3: Model Evaluation

```
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error


r2 = r2_score(y_test, y_pred)

mae = mean_absolute_error(y_test, y_pred)

rmse = mean_squared_error(y_test, y_pred, squared=False)


print("R^2:", r2)

print("MAE:", mae)

print("RMSE:", rmse)
```

Insights:

1. **R^2 score close to 0.8 means strong correlation between predicted and actual prices.**
2. **MAE indicates the average error in currency terms.**
3. **RMSE shows prediction variance; lower is better.**
4. **Good performance validates model's generalization**.

This Airbnb price prediction project applied end-to-end machine learning techniques to transform raw data into a predictive model. The model helps hosts understand what drives listing price and enables smarter pricing decisions.

Results:

1. RMSE and MAE within acceptable range
2. $R^2 > 0.75$ shows high predictive accuracy
3. Hosts can use insights to adjust listings and improve revenue
4. Use Random Forest for possible performance gain
5. Perform hyperparameter tuning (GridSearchCV)
6. Add external data (like seasonality, event calendars)

TASK 4 (VIDEO EXPLANATION)

https://drive.google.com/drive/folders/1P5q3bRkaOYml939AiubozIZDMMPQuAVM?usp=sharing