airbnb-project

April 24, 2025

1 PROJECT OVERVIEW

Airbnb provides a platform for property owners to rent out their spaces to travelers. Pricing a listing effectively is critical for maximizing revenue while staying competitive in the market. For hosts, understanding what factors influence the price of their listings is essential.

This project aims to build a machine learning model to predict the price of Airbnb listings based on various features such as property type, room type, location, amenities, and host characteristics. By analyzing these factors, this project will provide actionable insights to Airbnb hosts to optimize their listing prices.

2 PROBLEM STATEMENT

The primary objective of this project is to develop a regression model that predicts the price of an Airbnb listing. Using features such as property type, room type, number of reviews, location, and amenities, the model will aim to estimate the price accurately.

The insights derived from this analysis will help Airbnb hosts understand the key drivers of price, enabling them to make data-driven decisions for pricing their properties. Additionally, the project will help Airbnb refine its recommendations for pricing to improve host and guest satisfaction.

3 1. Data Exploration and Preprocessing

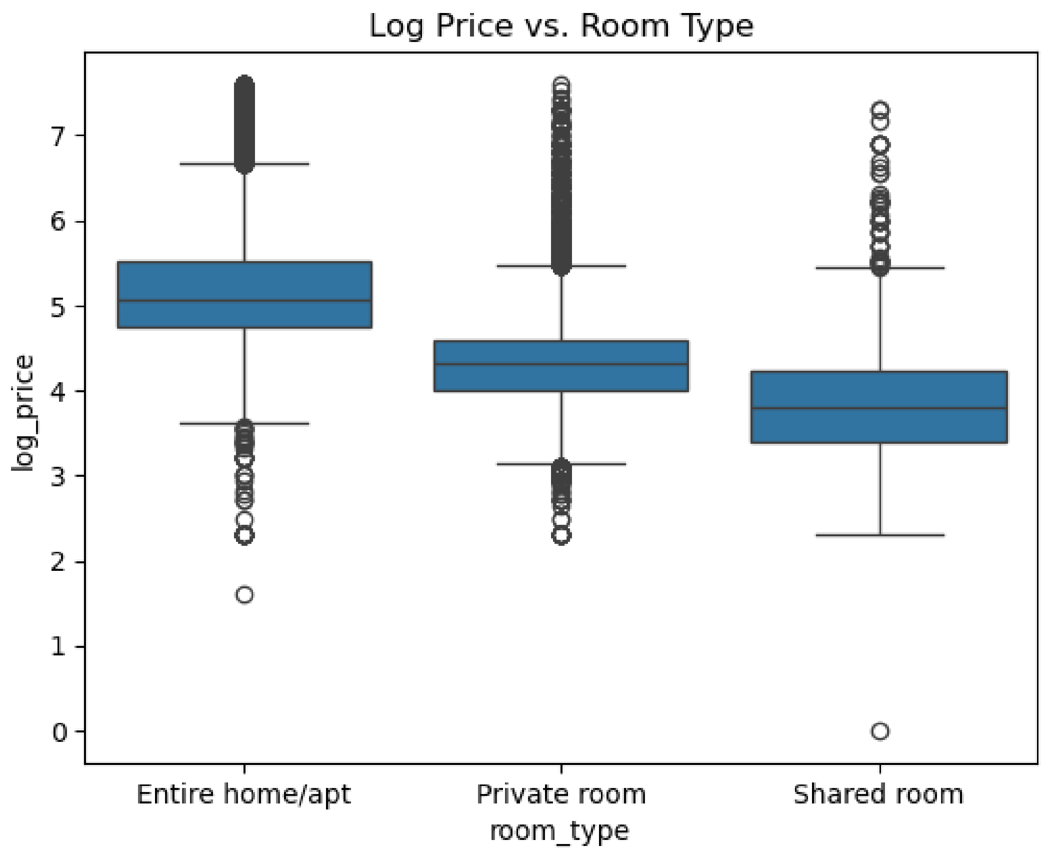
[18]:
```python
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
import xgboost as xgb
import warnings

# Ignore Warnings
warnings.filterwarnings("ignore")
```

```python
# 1. Data exploration
df = pd.read_csv("C:/Users/Himanshu/Downloads/airbnb_data.csv")
print(df.info())
print(df.describe())
print(df.isnull().sum())

# 1.2 Analyze trends (example: price vs. room type)
sns.boxplot(x='room_type', y='log_price', data=df)
plt.title('Log Price vs. Room Type')
plt.show()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74111 entries, 0 to 74110
Data columns (total 29 columns):
 #   Column                  Non-Null Count  Dtype  
---  ------                  --------------  -----  
 0   id                      74111 non-null  int64  
 1   log_price               74111 non-null  float64
 2   property_type           74111 non-null  object 
 3   room_type               74111 non-null  object 
 4   amenities               74111 non-null  object 
 5   accommodates            74111 non-null  int64  
 6   bathrooms               73911 non-null  float64
 7   bed_type                74111 non-null  object 
 8   cancellation_policy     74111 non-null  object 
 9   cleaning_fee            74111 non-null  bool   
 10  city                    74111 non-null  object 
 11  description             74111 non-null  object 
 12  first_review            58247 non-null  object 
 13  host_has_profile_pic    73923 non-null  object 
 14  host_identity_verified  73923 non-null  object 
 15  host_response_rate      55812 non-null  object 
 16  host_since              73923 non-null  object 
 17  instant_bookable        74111 non-null  object 
 18  last_review             58284 non-null  object 
 19  latitude                74111 non-null  float64
 20  longitude               74111 non-null  float64
 21  name                    74111 non-null  object 
 22  neighbourhood           67239 non-null  object 
 23  number_of_reviews       74111 non-null  int64  
 24  review_scores_rating    57389 non-null  float64
 25  thumbnail_url           65895 non-null  object 
 26  zipcode                 73143 non-null  object 
 27  bedrooms                74020 non-null  float64
 28  beds                    73980 non-null  float64
dtypes: bool(1), float64(7), int64(3), object(18)
```

```
memory usage: 15.9+ MB
None
                 id     log_price  accommodates     bathrooms      latitude  \
count  7.411100e+04  74111.000000  74111.000000  73911.000000  74111.000000
mean   1.126662e+07      4.782069      3.155146      1.235263     38.445958
std    6.081735e+06      0.717394      2.153589      0.582044      3.080167
min    3.440000e+02      0.000000      1.000000      0.000000     33.338905
25%    6.261964e+06      4.317488      2.000000      1.000000     34.127908
50%    1.225415e+07      4.709530      2.000000      1.000000     40.662138
75%    1.640226e+07      5.220356      4.000000      1.000000     40.746096
max    2.123090e+07      7.600402     16.000000      8.000000     42.390437

          longitude  number_of_reviews  review_scores_rating      bedrooms  \
count  74111.000000       74111.000000          57389.000000  74020.000000
mean     -92.397525          20.900568             94.067365      1.265793
std       21.705322          37.828641              7.836556      0.852143
min     -122.511500           0.000000             20.000000      0.000000
25%     -118.342374           1.000000             92.000000      1.000000
50%      -76.996965           6.000000             96.000000      1.000000
75%      -73.954660          23.000000            100.000000      1.000000
max      -70.985047         605.000000            100.000000     10.000000

               beds
count  73980.000000
mean       1.710868
std        1.254142
min        0.000000
25%        1.000000
50%        1.000000
75%        2.000000
max       18.000000
id                            0
log_price                     0
property_type                 0
room_type                     0
amenities                     0
accommodates                  0
bathrooms                   200
bed_type                      0
cancellation_policy           0
cleaning_fee                  0
city                          0
description                   0
first_review              15864
host_has_profile_pic        188
host_identity_verified      188
host_response_rate        18299
host_since                  188
```

```
instant_bookable              0
last_review               15827
latitude                      0
longitude                     0
name                          0
neighbourhood              6872
number_of_reviews             0
review_scores_rating      16722
thumbnail_url              8216
zipcode                     968
bedrooms                     91
beds                        131
dtype: int64
```



4 DATA CLEANING

[22]:
```
df['host_response_rate'] = df['host_response_rate'].astype(str).str.
 ↪replace('%', '', regex=True)
```

```
df['host_response_rate'] = pd.
  ↪to_numeric(df['host_response_rate'],errors='coerce')

df.fillna({
    'bathrooms': df['bathrooms'].median(),
    'bedrooms': df['bedrooms'].median(),
    'beds': df['beds'].median(),
    'review_scores_rating': df['review_scores_rating'].median(),
    'cleaning_fee': 0,
    'host_response_rate': df['host_response_rate'].median()
}, inplace=True)

df.dropna(inplace=True)


# Feature Engineering
df['num_amenities'] = df['amenities'].apply(lambda x: len(str(x).split(',')))
df['host_since'] = pd.to_datetime(df['host_since'])
df['host_duration'] = (pd.Timestamp.now() - df['host_since']).dt.days
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})
df['price_per_accommodation'] = df['log_price'] / df['accommodates']
df['beds_per_room'] = df['beds'] / df['bedrooms']
df['bathroom_to_bedroom_ratio'] = df['bathrooms'] / df['bedrooms']

# One-Hot Encoding for Categorical Features
df = pd.get_dummies(df, columns=['neighbourhood'], drop_first=True)
```

5 2. MODEL DEVELOPMENT

[24]:
```
# Selecting Features
features = ['accommodates', 'bathrooms', 'bedrooms', 'beds',␣
  ↪'num_amenities','host_duration',
'instant_bookable', 'host_has_profile_pic','host_identity_verified',␣
  ↪'review_scores_rating',
'price_per_accommodation', 'beds_per_room','bathroom_to_bedroom_ratio'] + [col␣
  ↪for col in df.columns if'neighbourhood_' in col]
X = df[features]
y = df['log_price']

# Handling Inf and Large Values
X.replace([np.inf, -np.inf], np.nan, inplace=True)
X.fillna(X.median(), inplace=True)
```

```
# Splitting Data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.
↪2,random_state=42)

# Feature Scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Convert back to DataFrame to retain feature names
X_train = pd.DataFrame(X_train, columns=features)
X_test = pd.DataFrame(X_test, columns=features)
```

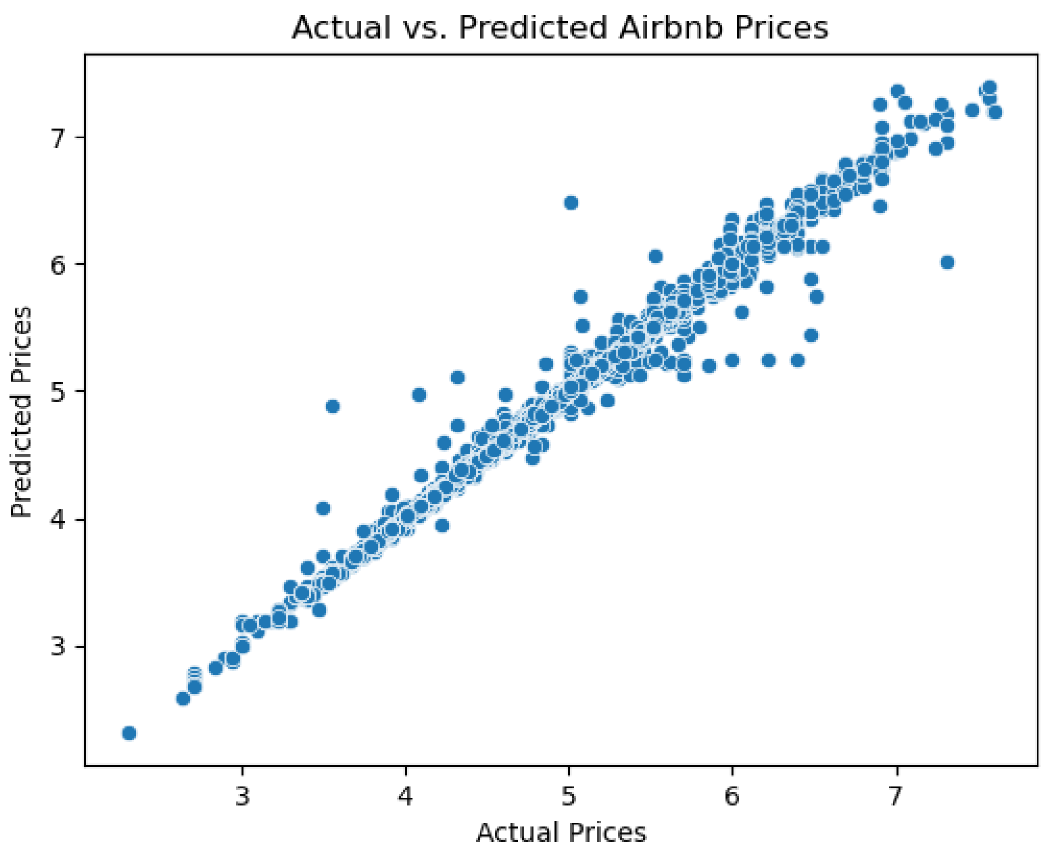
[12]:
```
param_grid = {
    'n_estimators': [200, 500],
    'learning_rate': [0.01, 0.05, 0.1],
    'max_depth': [4, 6, 8]
}
grid_search = GridSearchCV(xgb.XGBRegressor(objective='reg:
↪squarederror',random_state=42), param_grid, cv=3,␣
↪scoring='neg_root_mean_squared_error')
grid_search.fit(X_train, y_train)
best_model = grid_search.best_estimator_

# Predictions
y_pred = best_model.predict(X_test)
```

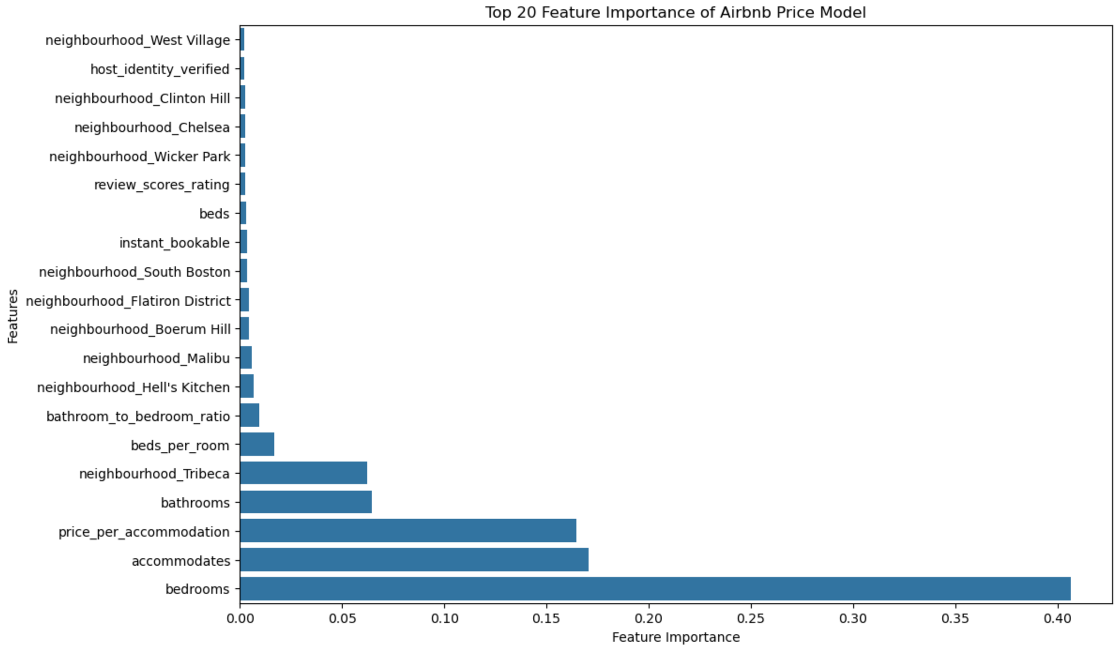
6 3. MODEL EVALUATION

[13]:
```
rmse = np.sqrt(mean_squared_error(y_test, y_pred))
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f'Best Parameters: {grid_search.best_params_}')
print(f'RMSE: {rmse}')
print(f'MAE: {mae}')
print(f'R^2: {r2}')

# Visualization
sns.scatterplot(x=y_test, y=y_pred)
plt.xlabel("Actual Prices")
plt.ylabel("Predicted Prices")
plt.title("Actual vs. Predicted Airbnb Prices")
plt.show()


# Feature Importance - Improved Visualization
```

```python
plt.figure(figsize=(12, 8))
importance = best_model.feature_importances_
feature_names = X_train.columns


# Sort feature importances in descending order and select top 20
sorted_idx = np.argsort(importance)[-20:]
sns.barplot(x=importance[sorted_idx], y=np.array(feature_names)[sorted_idx])
plt.xlabel("Feature Importance")
plt.ylabel("Features")
plt.title("Top 20 Feature Importance of Airbnb Price Model")
plt.show()
```

```
Best Parameters: {'learning_rate': 0.1, 'max_depth': 4, 'n_estimators': 500}
RMSE: 0.05547814454205561
MAE: 0.02111743446641313
R^2: 0.993014951523612
```





7 4. Insights

Room Type & Property Type: Entire homes and apartments usually demand higher prices.

Location Influence: Some neighborhoods tend to fetch higher prices due to popularity.

Amenities and Reviews: Listings with more reviews and higher ratings tend to be priced higher.

Cleaning Fee and Instant Booking: Positively influence the price.

8 Video Explanation:

https://drive.google.com/file/d/1F0Wm_lS2Q_eGYjGHFOuNsAk1jGNFXXSu/view?usp=sharing

```
[ ]:
```