Introduction to Machine learning and AI

REPORT

Airbnb price prediction

Introduction

Machine Learning models for Airbnb price prediction. Pricing a rental property on Airbnb is a challenging task for the owner as it determines the number of customers for the place. On the other hand, customers have to evaluate an offered price with minimal knowledge of an optimal value for the property.

We have performed and compared various regressions techniques (linear regression and Random Forest regression) on Airbnb dataset to investigate which regression technique gives an accurate price.

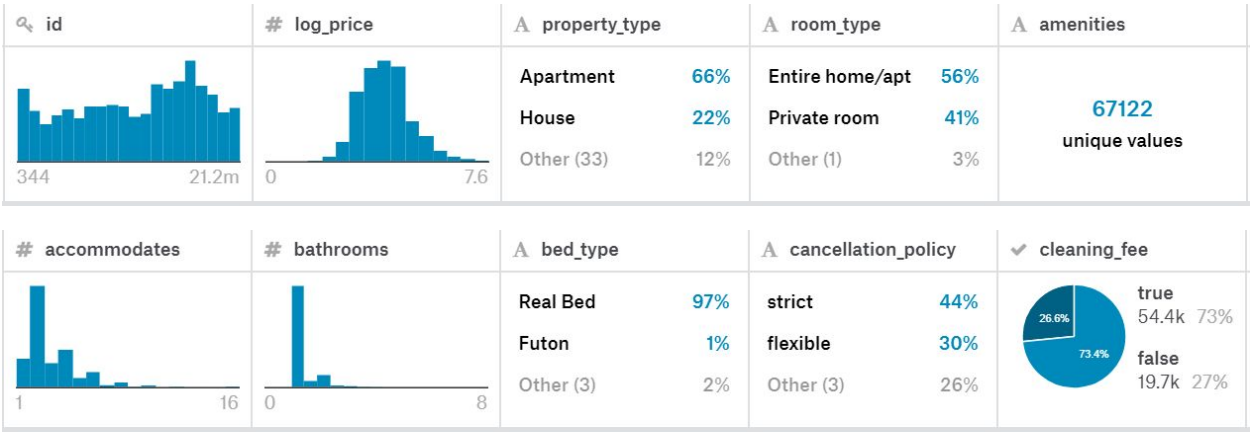
Exploratory Data Analysis (EDA):

The dataset

The dataset we used has 74111 rows and 29 columns which are:

<Id, log_price, property_type, room_type, amenities, accommodates, bathrooms, bed_type, cancellation_policy, cleaning_fee, city, description, first_review, host_has_profile_pic, host_identity_verified, host_response_rate, host_since, instant_bookable, last_review, latitude, longitude, name, neighbourhood, number_of_reviews, review_scores_rating, thumbnail_url, zip code, bedrooms, beds.>

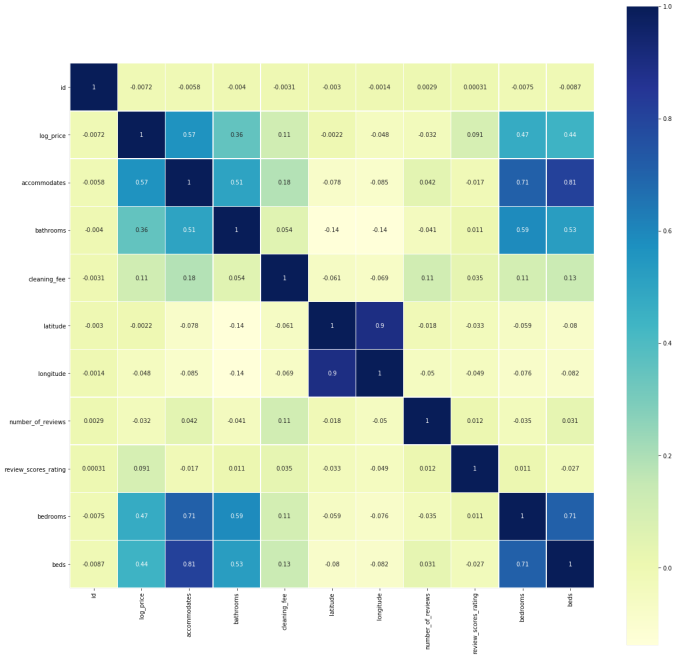
This housing and accommodation dataset shows housing in different cities like NYC, SF, DC, LA, Chicago and Boston. It also shows houses with different types of rooms (Apartment/entire house, Private room and shared rooms), the number of people that can stay in that place (ranging from 1-15) ectra.
Detailed information of a few of the 29 features in the dataset are as follows:



Correlation matrix

To use linear regression for modelling,its necessary to remove correlated variables to improve your model. We have plotted a correlation matrix in the form of a heat map to help us visualize it better. Lighter shades represents positive correlation while darker shades represents negative correlation. Then we remove the correlated variables during feature selection. Initially the correlation matrix for the given data looks like shown in the figure below.

Here the correlation matrix is a 29x29 matrix which shows the relation between every feature with every other feature in the dataset. Therefore if the correlation is high (positive or negative) we can infer that there is no linear relationship between these two predictors. Hence it is safe to drop these features when we are applying Linear Regression model to the dataset.



Feature Engineering

Feature reduction

After studying about matrix and the correlation between the features we can see that we can drop following features as they will not be useful to us in the prediction of price:

<*'id','number_of_reviews','review_scores_rating','latitude', 'longitude'*>

There is also correlation between bathroom and accommodates and bedrooms and bed, we have decided to keep only accommodates and drop the other (i.e <*'bathrooms'* , *'beds'*, *'bedrooms'*>)

Now we check how many features have missing data and how many values in these are missing. We immediately drop the feature which has lots of missing values and which will not be of use to us even if we replace it with zeros. The features are:

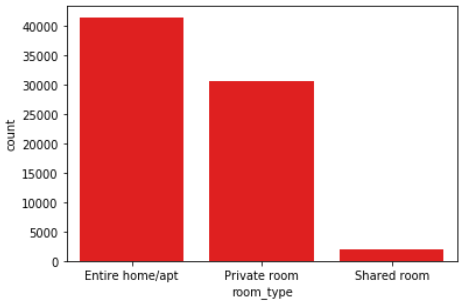
<*'first_review'*, *'host_has_profile_pic'*, *'host_identity_verified'*, *'host_response_rate'*, *'host_since'*, *'last_review'*, *'neighbourhood'*, *'thumbnail_url'*, *'zipcode'*>

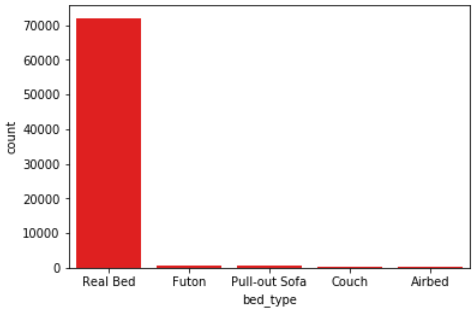
These features are not numerical and will not be of any use even if we categorize them because they have unique values ,so we drop all of them.

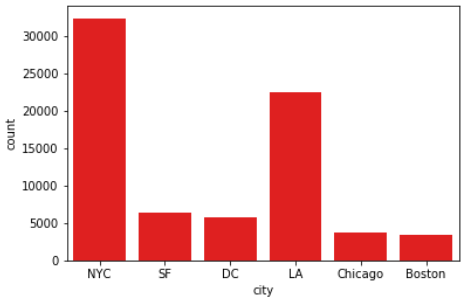
We also drop other features which are not numeric such as <'*property_type*','*amenities*', '*description*'> as these text data will not help us in the prediction.

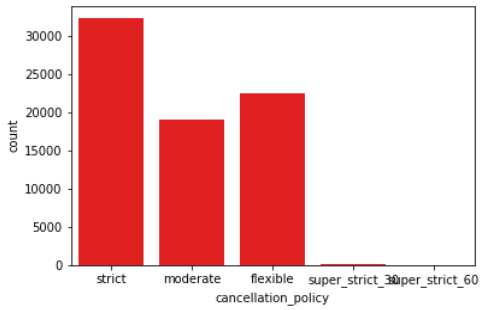
Handling categorical data

We require the input to be numerical and hence categorical features must be transformed into numerical features before we can use them. One of the most common ways to make this transformation is to **one-hot encode** the categorical features. For each unique value of a feature (say, 'room_type') one column is created (say, 'shared_room') where the value is 1 if for that data the original feature takes that value and 0 otherwise.
Here we one-hot encode the features <'room_type', 'bed_type', 'cancellation policy', 'city'>
The representation of the categories in the feature is given below:









The new set of features now are:
*<'log_price', 'accommodates', 'cleaning_fee', '**instant_bookable**', 'name', 'room_type_Entire home/apt', 'room_type_Private room', 'room_type_Shared room', i'bed_type_Airbed', 'bed_type_Couch', 'bed_type_Futon', 'bed_type_Pull-out Sofa', 'bed_type_Real Bed', 'cancellation_policy_flexible', 'cancellation_policy_moderate', 'cancellation_policy_strict', 'cancellation_policy_super_strict_30', 'cancellation_policy_super_strict_60', 'city_Boston', 'city_Chicago', 'city_DC', 'city_LA', 'city_NYC', 'city_SF'>*

We still have a feature '**instant_bookable**' which has text data (i.e 't' and 'f') so we replace 't' with 1 and 'f' with 0.

Regression models

From the data all the features except 'log_price' is X and 'log_price' is Y. X and Y data frames are then divided into test and train datasets.

Linear regression vs. Random Forest regression model

We trained and evaluated our dataset for both Linear and random forest regression and we found out that random forest regression model is more accurate than linear regression model.

The accuracy of the linear regression model was 51.76% whereas the accuracy of the Random Forest regression model was 52.66%.