Published in Towards Data Science

Soner Yıldırım Follow

Dec 14, 2021 · 9 min read · ✦ · Listen

Data Cleaning and EDA on Airbnb Dataset with Python Pandas and Seaborn

Discover the features that change the price

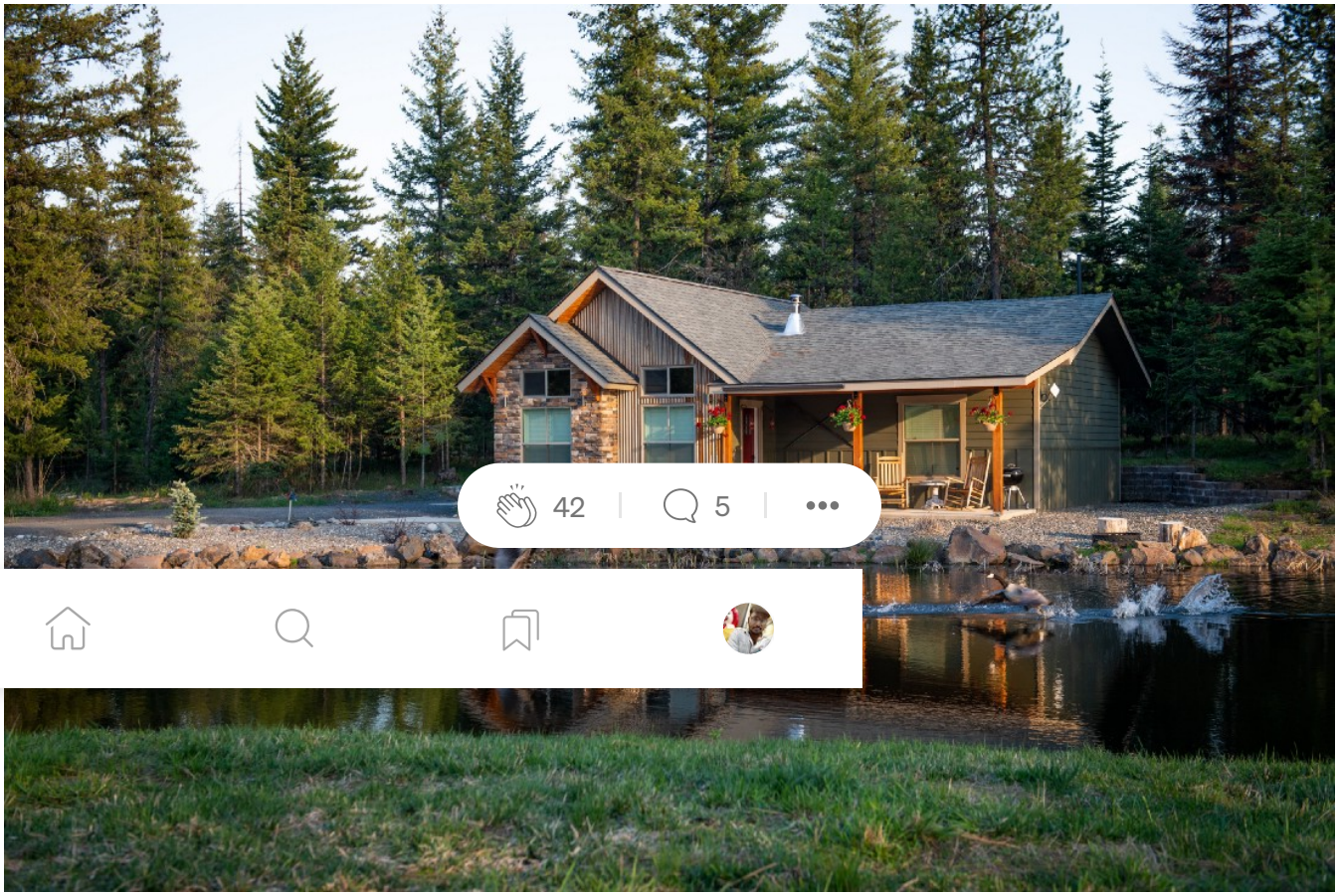

Photo by Stephen Wheeler on Unsplash

Airbnb connects people who have a place to rent and people who need a place to stay. It has become so popular and successful that most of us consider Airbnb as an option in our travel plans.

There are several factors that play a key role in defining the price of a place. Hosts are expected to list a reasonable price for their places.

On the other hand, people who look for a place to stay evaluate the listings with regards to several features such as location, size, amenities, and most importantly the price.

In this article, we will try to find out which features have an impact on the price of a place. There are many Airbnb datasets available with a creative commons license so feel free to use and explore them.

We will be using Pandas and Seaborn libraries for Python. Thus, this article will also be a practical guide for these libraries.

There are many datasets available on the website. The one we will be using is the listings file on 07 July 2021 from Barcelona, Spain. Let's start by importing Pandas and reading the data from the CSV file to create a DataFrame.

```
import pandas as pd

listings = pd.read_csv("listings.csv")

print(listings.shape)
(17079, 74)
```

There are two listings files. Make sure to use the one with 74 columns.

. . .

Data Cleaning

Some of the columns are not in a proper format for analysis. For instance, the price, host acceptance rate, and host response rate columns are stored as strings.

```
listings[
    ["price","host_response_rate","host_acceptance_rate"]
].dtypes

# output
price                   object
host_response_rate      object
host_acceptance_rate    object
dtype: object

_____________________________________________________________


listings[
    ["price","host_response_rate","host_acceptance_rate"]
].head()

# output
```

(image by author)

These columns need to be converted to a numerical format. We can use the `astype` function Pandas to change the data types. However, the “$” and “%” characters need to be removed beforehand.

```
listings["price"] =
listings["price"].str[1:].str.replace(",","").astype("float")

listings["host_response_rate"] =
listings["host_response_rate"].str[:-1].astype("float") / 100

listings["host_acceptance_rate"] =
listings["host_acceptance_rate"].str[:-1].astype("float") / 100
```

Since the “$” and “%” characters are the first and last ones in the string, we are able to remove them using their indices.

The `str[1:]` selects all the characters starting from the second one whereas `str[:-1]` selects all the characters up to the last one (last one is exclusive). Then, the `astype` function changes the data type to float.

These columns look as follows now:

(image by author)

. . .

There is a column called “bathrooms text” which contains a couple of information: bathroom quantity and bathroom type.

```
listings[["bathrooms_text"]].head()
```

	bathrooms_text
0	2 baths
1	2 baths
2	1.5 baths
3	1 private bath
4	3 baths

(image by author)

I think it is better two have separate columns for the number of bathrooms and the type of bathroom. We can do this task by using the `split` function available under the `str` accessor.

```
listings["bathroom_qty"] =
listings["bathrooms_text"].str.split(" ", expand=True)[0]

listings["bathroom_type"] =
listings["bathrooms_text"].str.split(" ", expand=True)[1]
```

(image by author)

. . .

Exploratory Data Analysis

There are 74 features in the dataset. It is better to divide them into some main groups to maintain the integrity of our analysis.

I would like to start with the target variable which is the price.

Price

Let's create a histogram of the price column to get an overview of its distribution. I will use the Seaborn library for data visualizations so the first step is to import it.

```
import seaborn as sns
sns.set_theme(font_scale=1.5, style="darkgrid")
```

The histogram can be created by using the `displot` function.

```
sns.displot(data=listings, x="price", kind="hist", aspect=1.5)
```


The histogram of price (image by author)

The distribution is highly skewed because of the outliers with very high prices. In such cases, it is better to take the logarithm of prices.

```
listings = listings[listings.price!=0]

listings.loc[:,"log_price"] = np.log(listings.loc[:,"price"])

sns.displot(data=listings, x="log_price", kind="hist",
aspect=1.5)
```

Natural log of price (image by author)

I removed a few listings with 0 prices from the data frame before taking the log. The price seems to have a log-normal distribution.

. . .

Host

Some of the features about the hosts play a key role in the price of a place. For instance, people tend to trust hosts with a verified identity more than unverified hosts.

```
listings.groupby(
    ["host_identity_verified"],
    as_index=False
).agg(
    avg_price = ("price","mean"),
    qty = ("price","count")
)
```

(image by author)

The listings are grouped by `host_identity_verified` column and the average price is calculated for each group. The “t” stands for true and “f” stands for false. The listings with a verified host have much higher prices on average.

It is better to also check the number of listings in each group to make sure data is not highly unbalanced.

. . .

Let’s also check the host response time with a similar method. Hosts who respond quicker might have higher prices on average because no one likes waiting for too long.

```
listings.groupby(
    ["host_response_time"], as_index=False
).agg(
    avg_price = ("price","mean"),
    qty = ("price","count")
).sort_values(
    by="avg_price", ascending=False
).reset_index(drop=True)
```

	host_response_time	avg_price	qty
0	within a few hours	148.848148	2160
1	within an hour	126.102222	6975
2	within a day	105.506459	2245
3	a few days or more	82.363158	950

(image by author)

In addition to the previous example, the results are sorted by the average prices using the `sort_values` function. This helps us see the relationship between response time and price more clearly.

The average price decreases with increasing response time as expected.

. . .

Size

The size or capacity of a place is a determining factor for the price. The `accommodates` column is an indication of how many people can stay in the place.

Let's use a box plot to compare the price distribution for different `accommodates` values.

```
sns.catplot(
    data=listings,
    x='accommodates', y='log_price', kind='box',
```

```
    height=6, aspect=1.8, width=0.5
)
```

The `catplot` function of Seaborn with the `kind` parameter set as box creates a box plot. The `height` and `aspect` parameters adjust the size of the plot. Finally, the `width` parameter changes the size of boxes in the plot.

(image by author)

We clearly see that the price increases with increasing accommodates value in general.

- The bottom line of the box is the first quartile so 25% of values are below this line.
- The line in the middle of the box indicates the median value. Half of the values are below this line.

- The top line of the box is the third quartile which means 75% of values are below this line.

Half of the values are between the bottom and top line of the box so we get an overview of the distribution of values. The dots on the bottom and top indicate outliers.

. . .

Availability

There are several columns related to the availability of a place. For instance, the `availability_30` column shows the number of available days in the next 30 days.

We can also see the number of available days in the next 60, 90, and 365 days.

Let's check if there is a correlation between the number of available days and price.

```
listings[[
    'availability_30', 'availability_60',
    'availability_90', 'availability_365',
    'log_price'
]].corr()
```

(image by author)

There seems to be a small positive correlation between the number of available days and the price. We also see a high positive correlation between the number of available days in the next 30, 60, and 90 days which is actually not a surprise.

There is a column called `instant_bookable` which might have an impact on the price. People who need a place urgently look for places that are instantly bookable. Money is usually a second degree of concern for them.

```
listings.groupby(
    ['instant_bookable'], as_index=False
).agg(
    avg_price = ("price","mean"),
    qty = ("price","count")
)
```

(image by author)

. . .

Type of the place

There are hotel rooms, private rooms, shared rooms, and entire homes in the dataset. We can use a box plot of the group by function to check how price changes according to the type of the place.

```
sns.catplot(
    data=listings,
    x='room_type', y='log_price', kind='box',
    height=6, aspect=1.8, width=0.5
)
```


(image by author)

Private and shared rooms are cheaper in general. We see lots of outliers in the private room category. They might be located in very popular places of the city.

. . .

Amenities

We usually consider the amenities in a place when making a reservation or renting a place. For instance, a decent coffee maker might be a game-changer.

There is a column called `amenities` in the dataset but it contains all the amenities available in the place. In order to check the effect of how a certain feature or facility on the price, we need to extract it from this column.

Thankfully, it is an easy operation with Python and Pandas. The following code snippet uses a list comprehension to check if "coffee maker" exists in the amenities column.

```
listings.loc[:,"has_coffee_maker"] = ['coffee maker' in
row.lower() for row in listings.amenities]
```

Each value is converted to lower case before the check because we are interested in "coffee maker", "Coffee maker" or any other upper and lower case combination of the coffee maker.

```
listings.has_coffee_maker.value_counts()

False    8777
True     8298
Name: has_coffee_maker, dtype: int64
```

Almost half of the places have a coffee maker. Let's check the average price for places with and without a coffee maker.

```
listings.groupby(
    ["has_coffee_maker"]
).agg(
    avg_price = ("price", "mean")
)
```


(image by author)

. . .

Location

Location is a very significant feature of a place. In some cases, it is extremely important to be close to a certain place and location is what people pay money for.

The neighborhood and neighborhood group columns give us an idea about the location of the place. We will check the average price based on these values.

If you want to take it one step further, there is a separate neighborhood file that contains the geographical coordinates for each neighborhood. It helps to make a thorough location-based evaluation. For instance, being close to Camp Nou when there is an important football game can boost the price.

```
listings.groupby(
    ["neighbourhood_group_cleansed"], as_index=False
).agg(
    avg_price = ("price", "mean"),
    qty = ("price", "count")
).sort_values(
    by="avg_price", ascending=False
).reset_index(drop=True)
```

(image by author)

Les Corts is the most expensive neighborhood group. Let's also check how average price changes for different neighborhoods within Les Corts.

```
listings[listings.neighbourhood_group_cleansed=="Les
Corts"].groupby(
    ["neighbourhood_cleansed"]
).agg(
    avg_price = ("price", "mean"),
    qty = ("price", "count")
).sort_values(
    by="avg_price", ascending=False
)
```

(image by author)

. . .

Conclusion

We have tried to explore the Airbnb dataset from Barcelona, Spain. Our focus was to learn how different features of a listing affect its price.

We can, of course, go deeper and look for more specific features and relationships. However, what we have done demonstrates the typical operations in an exploratory data analysis process.

We also did some data cleaning in order to make the data better suited for data analysis.

. . .

Don't forget to subscribe if you'd like to get an email whenever I publish a new

article.

You can become a Medium member to unlock full access to my writing, plus the rest of Medium. If you do so using the following link, I will receive a portion of your membership fee at no additional cost to you.

. . .

Thank you for reading. Please let me know if you have any feedback.

Sign up for The Variable

By Towards Data Science

Every Thursday, the Variable delivers the very best of Towards Data Science: from hands-on tutorials and cutting-edge research to original features you don't want to miss. Take a look.

Emails will be sent to narendrayadav1610@gmail.com. Not you?

Get this newsletter