DataUnderstandingAndPreparation

June 5, 2025

1 Lab 2: ML Life Cycle: Data Understanding and Data Preparation

[26]:
```
import os
import pandas as pd
import numpy as np
%matplotlib inline
import matplotlib.pyplot as plt
import seaborn as sns
```

In this lab, you will practice the second and third steps of the machine learning life cycle: data understanding and data preparation. You will beging preparing your data so that it can be used to train a machine learning model that solves a regression problem. Note that by the end of the lab, your data set won't be completely ready for the modeling phase, but you will gain experience using some common data preparation techniques.

You will complete the following tasks to transform your data:

1. Build your data matrix and define your ML problem:
   - Load the Airbnb "listings" data set into a DataFrame and inspect the data
   - Define the label and convert the label's data type to one that is more suitable for modeling
   - Identify features
2. Clean your data:
   - Handle outliers by building a new regression label column by winsorizing outliers
   - Handle missing data by replacing all missing values in the dataset with means
3. Perform feature transformation using one-hot encoding
4. Explore your data:
   - Identify two features with the highest correlation with label
   - Build appropriate bivariate plots to visualize the correlations between features and the label
5. Analysis:
   - Analyze the relationship between the features and the label
   - Brainstorm what else needs to be done to fully prepare the data for modeling

1.1 Part 1. Build Your Data Matrix (DataFrame) and Define Your ML Problem

Load a Data Set and Save it as a Pandas DataFrame We will be working with the Airbnb NYC "listings" data set. Use the specified path and name of the file to load the data. Save it as a Pandas DataFrame called `df`.

[27]:
```
# Do not remove or edit the line below:
filename = os.path.join(os.getcwd(), "data", "airbnbData.csv")
```

Task: Load the data and save it to DataFrame `df`.

Note: You may receive a warning message. Ignore this warning.

[28]:
```
# YOUR CODE HERE
df = pd.read_csv(filename)
```

```
/usr/local/lib/python3.6/dist-packages/IPython/core/interactiveshell.py:2728:
DtypeWarning: Columns (67) have mixed types.Specify dtype option on import or
set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Inspect the Data Task: Display the shape of `df` -- that is, the number of rows and columns.

[29]:
```
df.shape
```

[29]:
```
(38277, 74)
```

Task: Display the column names.

[30]:
```
df.columns
```

[30]:
```
Index(['id', 'listing_url', 'scrape_id', 'last_scraped', 'name', 'description',
       'neighborhood_overview', 'picture_url', 'host_id', 'host_url',
       'host_name', 'host_since', 'host_location', 'host_about',
       'host_response_time', 'host_response_rate', 'host_acceptance_rate',
       'host_is_superhost', 'host_thumbnail_url', 'host_picture_url',
       'host_neighbourhood', 'host_listings_count',
       'host_total_listings_count', 'host_verifications',
       'host_has_profile_pic', 'host_identity_verified', 'neighbourhood',
       'neighbourhood_cleansed', 'neighbourhood_group_cleansed', 'latitude',
       'longitude', 'property_type', 'room_type', 'accommodates', 'bathrooms',
       'bathrooms_text', 'bedrooms', 'beds', 'amenities', 'price',
       'minimum_nights', 'maximum_nights', 'minimum_minimum_nights',
       'maximum_minimum_nights', 'minimum_maximum_nights',
       'maximum_maximum_nights', 'minimum_nights_avg_ntm',
       'maximum_nights_avg_ntm', 'calendar_updated', 'has_availability',
       'availability_30', 'availability_60', 'availability_90',
       'availability_365', 'calendar_last_scraped', 'number_of_reviews',
       'number_of_reviews_ltm', 'number_of_reviews_l30d', 'first_review',
       'last_review', 'review_scores_rating', 'review_scores_accuracy',
       'review_scores_cleanliness', 'review_scores_checkin',
```

```
       'review_scores_communication', 'review_scores_location',
       'review_scores_value', 'license', 'instant_bookable',
       'calculated_host_listings_count',
       'calculated_host_listings_count_entire_homes',
       'calculated_host_listings_count_private_rooms',
       'calculated_host_listings_count_shared_rooms', 'reviews_per_month'],
      dtype='object')
```

Task: Get a peek at the data by displaying the first few rows, as you usually do.

[31]:
```
df.head()
```

[31]:
```
     id                          listing_url       scrape_id last_scraped  \
0  2595  https://www.airbnb.com/rooms/2595  20211204143024   2021-12-05
1  3831  https://www.airbnb.com/rooms/3831  20211204143024   2021-12-05
2  5121  https://www.airbnb.com/rooms/5121  20211204143024   2021-12-05
3  5136  https://www.airbnb.com/rooms/5136  20211204143024   2021-12-05
4  5178  https://www.airbnb.com/rooms/5178  20211204143024   2021-12-05

                                               name  \
0                             Skylit Midtown Castle
1  Whole flr w/private bdrm, bath & kitchen(pls r…
2                                   BlissArtsSpace!
3           Spacious Brooklyn Duplex, Patio + Garden
4                   Large Furnished Room Near B'way

                                        description  \
0  Beautiful, spacious skylit studio in the heart…
1  Enjoy 500 s.f. top floor in 1899 brownstone, w…
2  <b>The space</b><br />HELLO EVERYONE AND THANK…
3  We welcome you to stay in our lovely 2 br dupl…
4  Please don’t expect the luxury here just a bas…

                              neighborhood_overview  \
0  Centrally located in the heart of Manhattan ju…
1  Just the right mix of urban center and local n…
2                                               NaN
3                                               NaN
4    Theater district, many restaurants around here.

                                        picture_url  host_id  \
0  https://a0.muscache.com/pictures/f0813a11-40b2…     2845
1  https://a0.muscache.com/pictures/e49999c2-9fd5…     4869
2  https://a0.muscache.com/pictures/2090980c-b68e…     7356
3  https://a0.muscache.com/pictures/miso/Hosting-…     7378
4  https://a0.muscache.com/pictures/12065/f070997…     8967
```

```
                                  host_url  … review_scores_communication  \
0  https://www.airbnb.com/users/show/2845  …                        4.79
1  https://www.airbnb.com/users/show/4869  …                        4.80
2  https://www.airbnb.com/users/show/7356  …                        4.91
3  https://www.airbnb.com/users/show/7378  …                        5.00
4  https://www.airbnb.com/users/show/8967  …                        4.42

  review_scores_location review_scores_value license instant_bookable  \
0                   4.86                4.41     NaN                f
1                   4.71                4.64     NaN                f
2                   4.47                4.52     NaN                f
3                   4.50                5.00     NaN                f
4                   4.87                4.36     NaN                f

  calculated_host_listings_count calculated_host_listings_count_entire_homes  \
0                              3                                           3
1                              1                                           1
2                              2                                           0
3                              1                                           1
4                              1                                           0

  calculated_host_listings_count_private_rooms  \
0                                            0
1                                            0
2                                            2
3                                            0
4                                            1

  calculated_host_listings_count_shared_rooms reviews_per_month
0                                           0              0.33
1                                           0              4.86
2                                           0              0.52
3                                           0              0.02
4                                           0              3.68

[5 rows x 74 columns]
```

Define the Label Assume that your goal is to train a machine learning model that predicts the price of an Airbnb. This is an example of supervised learning and is a regression problem. In our dataset, our label will be the `price` column. Let's inspect the values in the `price` column.

[32]:
```python
df['price']
```

[32]:
```
0         $150.00
1          $75.00
2          $60.00
```

```
3        $275.00
4         $68.00
          …
38272     $79.00
38273     $76.00
38274    $116.00
38275    $106.00
38276    $689.00
Name: price, Length: 38277, dtype: object
```

Notice the `price` column contains values that are listed as <currency_name><numeric_value>. For example, it contains values that look like this: `$120`.

Task: Obtain the data type of the values in this column:

[33]:
```python
df.head()
```

[33]:
```
     id                          listing_url       scrape_id last_scraped  \
0  2595  https://www.airbnb.com/rooms/2595  20211204143024   2021-12-05
1  3831  https://www.airbnb.com/rooms/3831  20211204143024   2021-12-05
2  5121  https://www.airbnb.com/rooms/5121  20211204143024   2021-12-05
3  5136  https://www.airbnb.com/rooms/5136  20211204143024   2021-12-05
4  5178  https://www.airbnb.com/rooms/5178  20211204143024   2021-12-05

                                                name  \
0                            Skylit Midtown Castle
1  Whole flr w/private bdrm, bath & kitchen(pls r…
2                                   BlissArtsSpace!
3          Spacious Brooklyn Duplex, Patio + Garden
4                  Large Furnished Room Near B'way

                                         description  \
0  Beautiful, spacious skylit studio in the heart…
1  Enjoy 500 s.f. top floor in 1899 brownstone, w…
2  <b>The space</b><br />HELLO EVERYONE AND THANK…
3  We welcome you to stay in our lovely 2 br dupl…
4  Please don't expect the luxury here just a bas…

                               neighborhood_overview  \
0  Centrally located in the heart of Manhattan ju…
1  Just the right mix of urban center and local n…
2                                                NaN
3                                                NaN
4    Theater district, many restaurants around here.

                                         picture_url  host_id  \
0  https://a0.muscache.com/pictures/f0813a11-40b2…     2845
1  https://a0.muscache.com/pictures/e49999c2-9fd5…     4869
```

```
2  https://a0.muscache.com/pictures/2090980c-b68e…     7356
3  https://a0.muscache.com/pictures/miso/Hosting-…     7378
4  https://a0.muscache.com/pictures/12065/f070997…     8967

                                 host_url  … review_scores_communication  \
0  https://www.airbnb.com/users/show/2845  …                        4.79
1  https://www.airbnb.com/users/show/4869  …                        4.80
2  https://www.airbnb.com/users/show/7356  …                        4.91
3  https://www.airbnb.com/users/show/7378  …                        5.00
4  https://www.airbnb.com/users/show/8967  …                        4.42

  review_scores_location review_scores_value license instant_bookable  \
0                   4.86                4.41     NaN                f
1                   4.71                4.64     NaN                f
2                   4.47                4.52     NaN                f
3                   4.50                5.00     NaN                f
4                   4.87                4.36     NaN                f

  calculated_host_listings_count calculated_host_listings_count_entire_homes  \
0                              3                                           3
1                              1                                           1
2                              2                                           0
3                              1                                           1
4                              1                                           0

  calculated_host_listings_count_private_rooms  \
0                                            0
1                                            0
2                                            2
3                                            0
4                                            1

  calculated_host_listings_count_shared_rooms reviews_per_month
0                                           0              0.33
1                                           0              4.86
2                                           0              0.52
3                                           0              0.02
4                                           0              3.68

[5 rows x 74 columns]
```

Notice that the data type is ”object,” which in Pandas translates to the String data type.

Task: Display the first 15 unique values of the `price` column:

[34]:
```
df['price'] = df['price'].astype(str)
```

In order for us to use the prices for modeling, we will have to transform the values in the `price` column from strings to floats. We will:

- remove the dollar signs (in this case, the platform forces the currency to be the USD, so we do not need to worry about targeting, say, the Japanese Yen sign, nor about converting the values into USD).
- remove the commas from all values that are in the thousands or above: for example, `$2,500`.

The code cell below accomplishes this.

[35]:
```
df['price'] = df['price'].str.replace(',', '')
df['price'] = df['price'].str.replace('$', '')
df['price'] = df['price'].astype(float)
```

Task: Display the first 15 unique values of the `price` column again to make sure they have been transformed.

[36]:
```
df['price'].unique()[:15]
```

[36]:
```
array([150.,  75.,  60., 275.,  68.,  98.,  89.,  65.,  62.,  90., 199.,
        96., 299., 140., 175.])
```

Identify Features Simply by inspecting the data, let's identify some columns that should not serve as features - those that will not help us solve our predictive ML problem.

Some that stand out are columns that contain website addresses (URLs).

Task: Create a list which contains the names of columns that contain URLs. Save the resulting list to variable `url_colnames`.

Tip: There are different ways to accomplish this, including using Python list comprehensions.

[37]:
```
url_colnames = [col for col in df.columns if 'url' in col.lower()]
url_colnames
```

[37]:
```
['listing_url',
 'picture_url',
 'host_url',
 'host_thumbnail_url',
 'host_picture_url']
```

Task: Drop the columns with the specified names contained in list `url_colnames` in place (that is, make sure this change applies to the original DataFrame `df`, instead of creating a temporary new DataFrame object with fewer columns).

[38]:
```
df.drop(columns=url_colnames, inplace=True)
```

Task: Display the shape of the data to verify that the new number of columns is what you expected.

[39]:
```
df.shape
```

[39]: (38277, 69)

Task: In the code cell below, display the features that we will use to solve our ML problem.

[40]:
```
# YOUR CODE HERE
features = [
    'neighbourhood_group_cleansed',
    'room_type',
    'minimum_nights',
    'number_of_reviews',
    'reviews_per_month',
    'availability_365',
    'calculated_host_listings_count',
    'review_scores_rating'
]
features
```

[40]:
```
['neighbourhood_group_cleansed',
 'room_type',
 'minimum_nights',
 'number_of_reviews',
 'reviews_per_month',
 'availability_365',
 'calculated_host_listings_count',
 'review_scores_rating']
```

Task: Are there any other features that you think may not be well suited for our machine learning problem? Note your findings in the markdown cell below.

After looking through the dataset, I found a few features that probably won't help us predict the price very well:

-id, scrape_id: These are just identifiers, so they don't add any value for prediction. -listing_url, picture_url, host_url, etc.: These are all just URLs, and we already removed them. -name, description, host_about: These are long text fields that could be useful, but they'd require natural language processing (which is out of scope for now). -calendar_last_scraped, last_scraped: These only tell us when the data was collected, not something about the listing itself. -license: A lot of values are missing, and it's not clear how it connects to price. -host_picture_url, host_thumbnail_url: These are just images of the host — not likely to impact price. -host_verifications: This is a list field with mixed values like email and phone, which would need special handling.

So, I think it's better to drop or ignore these columns and focus on cleaner, more useful features like room_type, neighbourhood_group_cleansed, and review_scores_rating.

1.2 Part 2. Clean Your Data

Let's now handle outliers and missing data.

1.2.1 a. Handle Outliers

Let us prepare the data in our label column. Namely, we will detect and replace outliers in the data using winsorization.

Task: Create a new version of the `price` column, named `label_price`, in which you will replace the top and bottom 1% outlier values with the corresponding percentile value. Add this new column to the DataFrame `df`.

Remember, you will first need to load the `stats` module from the `scipy` package:

[43]:
```
# YOUR CODE HERE
# Import the necessary module
from scipy import stats

# Apply winsorization: trim the lowest and highest 1% of the price data
df['label_price'] = stats.mstats.winsorize(df['price'], limits=[0.01, 0.01])
```

Let's verify that the new column `label_price` was added to DataFrame `df`:

[44]:
```
df.head()
```

[44]:
```
     id       scrape_id last_scraped  \
0  2595  20211204143024   2021-12-05
1  3831  20211204143024   2021-12-05
2  5121  20211204143024   2021-12-05
3  5136  20211204143024   2021-12-05
4  5178  20211204143024   2021-12-05

                                                name  \
0                                 Skylit Midtown Castle
1  Whole flr w/private bdrm, bath & kitchen(pls r…
2                                       BlissArtsSpace!
3                  Spacious Brooklyn Duplex, Patio + Garden
4                          Large Furnished Room Near B'way

                                         description  \
0  Beautiful, spacious skylit studio in the heart…
1  Enjoy 500 s.f. top floor in 1899 brownstone, w…
2  <b>The space</b><br />HELLO EVERYONE AND THANK…
3  We welcome you to stay in our lovely 2 br dupl…
4  Please don’t expect the luxury here just a bas…

                                neighborhood_overview  host_id    host_name  \
0  Centrally located in the heart of Manhattan ju…     2845     Jennifer
1  Just the right mix of urban center and local n…     4869  LisaRoxanne
2                                                NaN     7356        Garon
3                                                NaN     7378      Rebecca
4    Theater district, many restaurants around here.     8967     Shunichi
```

```
   host_since                     host_location  … review_scores_location  \
0  2008-09-09  New York, New York, United States  …                   4.86   
1  2008-12-07  New York, New York, United States  …                   4.71   
2  2009-02-03  New York, New York, United States  …                   4.47   
3  2009-02-03  Brooklyn, New York, United States  …                   4.50   
4  2009-03-03  New York, New York, United States  …                   4.87   

  review_scores_value license instant_bookable calculated_host_listings_count  \
0                4.41     NaN                f                              3   
1                4.64     NaN                f                              1   
2                4.52     NaN                f                              2   
3                5.00     NaN                f                              1   
4                4.36     NaN                f                              1   

  calculated_host_listings_count_entire_homes  \
0                                          3   
1                                          1   
2                                          0   
3                                          1   
4                                          0   

   calculated_host_listings_count_private_rooms  \
0                                            0   
1                                            0   
2                                            2   
3                                            0   
4                                            1   

   calculated_host_listings_count_shared_rooms reviews_per_month label_price
0                                            0              0.33       150.0
1                                            0              4.86        75.0
2                                            0              0.52        60.0
3                                            0              0.02       275.0
4                                            0              3.68        68.0

[5 rows x 70 columns]
```

Task: Check that the values of `price` and `label_price` are *not* identical.

You will do this by subtracting the two columns and finding the resulting *unique values* of the resulting difference. Note: If all values are identical, the difference would not contain unique values. If this is the case, outlier removal did not work.

[46]:
```
# YOUR CODE HERE
(df['price'] - df['label_price']).unique()
```

```
[46]: array([ 0.000e+00,  1.500e+03,  3.000e+02,  1.000e+03,  1.979e+03,
       -1.000e+00,  8.990e+02,  2.000e+02,  9.990e+02,  5.000e+02,
       -8.000e+00,  5.000e+03,  4.250e+03,  5.500e+02,  2.500e+02,
        5.500e+03,  1.750e+03,  2.750e+03,  6.000e+02, -1.100e+01,
        1.249e+03,  4.330e+02,  5.700e+01,  3.930e+02, -4.000e+00,
        4.000e+02,  1.695e+03,  8.990e+03,  2.140e+02, -1.400e+01,
        8.999e+03,  7.630e+02, -2.000e+00, -9.000e+00,  2.430e+02,
        1.000e+02,  6.400e+01,  2.974e+03,  7.700e+01, -3.000e+00,
       -7.000e+00,  3.500e+02,  2.450e+02,  8.100e+01,  5.710e+02,
        6.314e+03, -5.000e+00, -1.000e+01,  2.000e+00,  9.900e+01,
        1.200e+03,  4.300e+02,  1.100e+03,  8.500e+01,  4.000e+03,
        9.000e+03,  1.350e+03,  5.000e+01,  2.000e+03,  1.299e+03,
        1.430e+02,  1.499e+03,  3.700e+02, -1.900e+01,  6.184e+03,
       -1.300e+01,  2.210e+02,  1.857e+03, -1.500e+01,  9.000e+02,
        7.500e+01, -6.000e+00,  6.430e+02,  3.929e+03,  2.910e+02,
        3.990e+02,  8.000e+03,  5.429e+03,  3.000e+03, -1.800e+01,
        5.143e+03,  1.400e+03,  4.750e+02,  2.214e+03,  1.910e+02,
        4.250e+02,  1.250e+02,  3.330e+02,  4.990e+02,  8.000e+02,
        2.250e+02,  2.500e+03,  8.190e+02,  6.000e+03,  3.030e+02,
        3.070e+02,  1.640e+02,  3.420e+02,  5.600e+01,  2.600e+03,
        2.200e+03,  5.700e+02,  1.642e+03,  7.000e+00,  9.810e+02,
        2.120e+02,  1.850e+03,  4.500e+01,  4.510e+02,  5.120e+02,
        2.360e+02,  6.200e+01,  1.020e+02,  2.590e+02,  7.500e+02,
        9.750e+02,  5.290e+02,  2.960e+02,  9.500e+02,  1.600e+03,
        2.750e+02,  4.640e+02,  2.570e+02, -2.900e+01, -1.700e+01,
        9.500e+01,  2.850e+02,  3.382e+03,  1.839e+03,  1.261e+03,
        2.900e+01,  2.260e+02,  1.130e+02,  9.000e+00,  2.160e+02,
        1.160e+02, -1.200e+01,  4.950e+02,  2.500e+01,  2.860e+02,
        2.557e+03,  1.614e+03,  7.100e+01,  5.400e+01,  5.750e+02,
        1.700e+03,  2.400e+01,  1.700e+01,  1.140e+02,  2.900e+02,
        2.990e+02,  9.950e+02,  1.760e+02,  8.300e+02,  2.520e+03,
        8.650e+02,  6.700e+01,  1.797e+03,  2.729e+03,  7.600e+02,
        1.640e+03,  6.860e+02,  2.490e+02,  3.730e+02,  5.500e+01,
        7.420e+02,  2.920e+02,  1.436e+03,  3.860e+02,  3.570e+02,
        4.740e+02,  2.333e+03,  1.100e+01,  1.400e+01,  3.143e+03,
        4.500e+02,  8.300e+01,  1.990e+02,  8.560e+02,  1.370e+02,
        7.600e+01,  1.290e+02,  6.540e+02,  3.400e+01,  3.690e+02,
        8.170e+02,  4.790e+02,  8.970e+02,  3.140e+02,  3.320e+02,
        2.820e+02,  1.090e+02,  1.260e+02,  1.490e+02,  2.110e+02,
        1.232e+03,  3.464e+03,  2.119e+03,  3.310e+02,  5.650e+02,
        1.071e+03,  2.855e+03,  1.050e+03,  1.157e+03,  4.655e+03,
        9.800e+02])
```

1.2.2 b. Handle Missing Data

Next we are going to find missing values in our entire dataset and impute the missing values by replace them with means.

Identifying missingness **Task**: Check if a given value in the data is missing, and sum up the resulting values by columns. Save this sum to variable `nan_count`. Print the results.

[47]:
```
nan_count = df.isnull().sum()
nan_count
```

[47]:
```
id                                              0
scrape_id                                       0
last_scraped                                    0
name                                           13
description                                  1192
                                             …
calculated_host_listings_count_entire_homes     0
calculated_host_listings_count_private_rooms    0
calculated_host_listings_count_shared_rooms     0
reviews_per_month                            9504
label_price                                     0
Length: 70, dtype: int64
```

Those are more columns than we can eyeball! For this exercise, we don't care about the number of missing values -- we just want to get a list of columns that have *any* missing values.

Task: From the variable `nan_count`, create a new series called `nan_detected` that contains `True` or `False` values that indicate whether the number of missing values is *not zero*:

[48]:
```
nan_detected = nan_count > 0
nan_detected
```

[48]:
```
id                                             False
scrape_id                                      False
last_scraped                                   False
name                                            True
description                                     True
                                             …
calculated_host_listings_count_entire_homes    False
calculated_host_listings_count_private_rooms   False
calculated_host_listings_count_shared_rooms    False
reviews_per_month                               True
label_price                                    False
Length: 70, dtype: bool
```

Since replacing the missing values with the mean only makes sense for the columns that contain numerical values (and not for strings), let us create another condition: the *type* of the column must be `int` or `float`.

Task: Create a series that contains `True` if the type of the column is either `int64` or `float64`. Save the results to the variable `is_int_or_float`.

```
[50]: is_int_or_float = df.dtypes.apply(lambda dtype: np.issubdtype(dtype, np.number))
      is_int_or_float
```

```
[50]: id                                               True
      scrape_id                                        True
      last_scraped                                    False
      name                                            False
      description                                     False
                                                      …
      calculated_host_listings_count_entire_homes      True
      calculated_host_listings_count_private_rooms     True
      calculated_host_listings_count_shared_rooms      True
      reviews_per_month                                True
      label_price                                      True
      Length: 70, dtype: bool
```

Task: Combine the two binary series (`nan_detected` and `is_int_or_float`) into a new series named `to_impute`. It will contain the value `True` if a column contains missing values *and* is of type 'int' or 'float'

```
[51]: to_impute = nan_detected & is_int_or_float
      to_impute
```

```
[51]: id                                              False
      scrape_id                                       False
      last_scraped                                    False
      name                                            False
      description                                     False
                                                      …
      calculated_host_listings_count_entire_homes     False
      calculated_host_listings_count_private_rooms    False
      calculated_host_listings_count_shared_rooms     False
      reviews_per_month                                True
      label_price                                     False
      Length: 70, dtype: bool
```

Finally, let's display a list that contains just the selected column names contained in `to_impute`:

```
[52]: df.columns[to_impute]
```

```
[52]: Index(['host_listings_count', 'host_total_listings_count', 'bathrooms',
             'bedrooms', 'beds', 'minimum_minimum_nights', 'maximum_minimum_nights',
             'minimum_maximum_nights', 'maximum_maximum_nights',
             'minimum_nights_avg_ntm', 'maximum_nights_avg_ntm', 'calendar_updated',
```

```
         'review_scores_rating', 'review_scores_accuracy',
         'review_scores_cleanliness', 'review_scores_checkin',
         'review_scores_communication', 'review_scores_location',
         'review_scores_value', 'reviews_per_month'],
        dtype='object')
```

We just identified and displayed the list of candidate columns for potentially replacing missing values with the column mean.

Assume that you have decided that you should impute the values for these specific columns: `host_listings_count`, `host_total_listings_count`, `bathrooms`, `bedrooms`, and `beds`:

[53]:
```python
to_impute_selected = ['host_listings_count', 'host_total_listings_count',␣
 ↪'bathrooms',
       'bedrooms', 'beds']
```

Keeping record of the missingness: creating dummy variables As a first step, you will now create dummy variables indicating the missingness of the values.

Task: For every column listed in `to_impute_selected`, create a new corresponding column called `<original-column-name>_na`. These columns should contain the a `True`or `False` value in place of `NaN`.

[70]:
```python
# YOUR CODE HERE
for col in to_impute_selected:
    df[col + '_na'] = df[col].isna()
```

Check that the DataFrame contains the new variables:

[71]:
```python
df.head()
```

[71]:
```
     id       scrape_id last_scraped  \
0  2595  20211204143024   2021-12-05
1  3831  20211204143024   2021-12-05
2  5121  20211204143024   2021-12-05
3  5136  20211204143024   2021-12-05
4  5178  20211204143024   2021-12-05


                                                name  \
0                                 Skylit Midtown Castle
1  Whole flr w/private bdrm, bath & kitchen(pls r…
2                                       BlissArtsSpace!
3            Spacious Brooklyn Duplex, Patio + Garden
4                      Large Furnished Room Near B'way


                                         description  \
0  Beautiful, spacious skylit studio in the heart…
1  Enjoy 500 s.f. top floor in 1899 brownstone, w…
```

```
2  <b>The space</b><br />HELLO EVERYONE AND THANK…
3  We welcome you to stay in our lovely 2 br dupl…
4  Please don’t expect the luxury here just a bas…

                          neighborhood_overview  host_id    host_name  \
0  Centrally located in the heart of Manhattan ju…     2845     Jennifer
1  Just the right mix of urban center and local n…     4869  LisaRoxanne
2                                             NaN     7356        Garon
3                                             NaN     7378      Rebecca
4    Theater district, many restaurants around here.     8967     Shunichi

   host_since                     host_location  … beds_na  \
0  2008-09-09  New York, New York, United States  …   False
1  2008-12-07  New York, New York, United States  …   False
2  2009-02-03  New York, New York, United States  …   False
3  2009-02-03  Brooklyn, New York, United States  …   False
4  2009-03-03  New York, New York, United States  …   False

  a few days or more unavailable within a day within a few hours  \
0                  0           0            1                  0
1                  1           0            0                  0
2                  0           0            0                  0
3                  0           0            1                  0
4                  0           0            1                  0

   within an hour  Entire home/apt Hotel room Private room Shared room
0               0                1          0            0           0
1               0                1          0            0           0
2               1                0          0            1           0
3               0                1          0            0           0
4               0                0          0            1           0

[5 rows x 82 columns]
```

Replacing the missing values with mean values of the column **Task**: For every column listed in `to_impute_selected`, fill the missing values with the corresponding mean of all values in the column (do not create new columns).

[56]:
```python
# YOUR CODE HERE
for col in to_impute_selected:
    df[col].fillna(df[col].mean(), inplace=True)
```

Check your results below. The code displays the count of missing values for each of the selected columns.

[73]:
```
for colname in to_impute_selected:
    print("{} missing values count :{}".format(colname, np.sum(df[colname].
 ↪isnull(), axis = 0)))
```

```
host_listings_count missing values count :0
host_total_listings_count missing values count :0
bathrooms missing values count :38277
bedrooms missing values count :0
beds missing values count :0
```

Why did the `bathrooms` column retain missing values after our imputation?

Task: List the unique values of the `bathrooms` column.

[74]:
```
# YOUR CODE HERE
df['bathrooms'].unique()
```

[74]:
```
array([nan])
```

The column did not contain a single value (except the `NaN` indicator) to begin with.

1.3 Part 3. Perform One-Hot Encoding

Machine learning algorithms operate on numerical inputs. Therefore, we have to transform text data into some form of numerical representation to prepare our data for the model training phase. Some features that contain text data are categorical. Others are not. For example, we removed all of the features that contained URLs. These features were not categorical, but rather contained what is called unstructured text. However, not all features that contain unstructured text should be removed, as they can contain useful information for our machine learning problem. Unstructured text data is usually handled by Natural Language Processing (NLP) techniques. You will learn more about NLP later in this course.

However, for features that contain categorical values, one-hot encoding is a common feature engineering technique that transforms them into binary representations.

We will first choose one feature column to one-hot encode: `host_response_time`. Let's inspect the unique values this feature can have.

[59]:
```
df['host_response_time'].unique()
```

[59]:
```
array(['within a day', 'a few days or more', 'within an hour', nan,
       'within a few hours'], dtype=object)
```

Note that each entry can contain one of five possible values.

Task: Since one of these values is `NaN`, replace every entry in the column `host_response_time` that contains a `NaN` value with the string 'unavailable'.

[60]:
```
# YOUR CODE HERE
df['host_response_time'] = df['host_response_time'].fillna('unavailable')
```

Let's inspect the `host_response_time` column to see the new values.

[61]:
```
df['host_response_time'].unique()
```

[61]:
```
array(['within a day', 'a few days or more', 'within an hour',
       'unavailable', 'within a few hours'], dtype=object)
```

Task: Use `pd.get_dummies()` to one-hot encode the `host_response_time` column. Save the result to DataFrame `df_host_response_time`.

[62]:
```
df_host_response_time = pd.get_dummies(df['host_response_time'])
df_host_response_time
```

[62]:
```
       a few days or more  unavailable  within a day  within a few hours  \
0                       0            0             1                   0
1                       1            0             0                   0
2                       0            0             0                   0
3                       0            0             1                   0
4                       0            0             1                   0
…                     …          …          …                   …
38272                   0            0             0                   1
38273                   0            0             0                   1
38274                   0            0             0                   0
38275                   0            0             0                   0
38276                   0            0             0                   0

       within an hour
0                   0
1                   0
2                   1
3                   0
4                   0
…               …
38272               0
38273               0
38274               1
38275               1
38276               1

[38277 rows x 5 columns]
```

Task: Since the `pd.get_dummies()` function returned a new DataFrame rather than making the changes to the original DataFrame `df`, add the new DataFrame `df_host_response_time` to DataFrame `df`, and delete the original `host_response_time` column from DataFrame `df`.

[63]:
```
# YOUR CODE HERE
# Add the new one-hot encoded columns to df
df = pd.concat([df, df_host_response_time], axis=1)
```

```
# Remove the original 'host_response_time' column
df.drop(columns='host_response_time', inplace=True)
```

Let's inspect DataFrame `df` to see the changes that have been made.

[64]:
```
df.columns
```

[64]:
```
Index(['id', 'scrape_id', 'last_scraped', 'name', 'description',
       'neighborhood_overview', 'host_id', 'host_name', 'host_since',
       'host_location', 'host_about', 'host_response_rate',
       'host_acceptance_rate', 'host_is_superhost', 'host_neighbourhood',
       'host_listings_count', 'host_total_listings_count',
       'host_verifications', 'host_has_profile_pic', 'host_identity_verified',
       'neighbourhood', 'neighbourhood_cleansed',
       'neighbourhood_group_cleansed', 'latitude', 'longitude',
       'property_type', 'room_type', 'accommodates', 'bathrooms',
       'bathrooms_text', 'bedrooms', 'beds', 'amenities', 'price',
       'minimum_nights', 'maximum_nights', 'minimum_minimum_nights',
       'maximum_minimum_nights', 'minimum_maximum_nights',
       'maximum_maximum_nights', 'minimum_nights_avg_ntm',
       'maximum_nights_avg_ntm', 'calendar_updated', 'has_availability',
       'availability_30', 'availability_60', 'availability_90',
       'availability_365', 'calendar_last_scraped', 'number_of_reviews',
       'number_of_reviews_ltm', 'number_of_reviews_l30d', 'first_review',
       'last_review', 'review_scores_rating', 'review_scores_accuracy',
       'review_scores_cleanliness', 'review_scores_checkin',
       'review_scores_communication', 'review_scores_location',
       'review_scores_value', 'license', 'instant_bookable',
       'calculated_host_listings_count',
       'calculated_host_listings_count_entire_homes',
       'calculated_host_listings_count_private_rooms',
       'calculated_host_listings_count_shared_rooms', 'reviews_per_month',
       'label_price', 'host_listings_count_na', 'host_total_listings_count_na',
       'bathrooms_na', 'bedrooms_na', 'beds_na', 'a few days or more',
       'unavailable', 'within a day', 'within a few hours', 'within an hour'],
      dtype='object')
```

One-hot encode additional features **Task**: Use the code cell below to find columns that contain string values (the 'object' data type) and inspect the *number* of unique values each column has.

[65]:
```
# YOUR CODE HERE
# Find object (string) columns
object_columns = df.select_dtypes(include='object')

# Display the number of unique values for each object column
```

```python
object_columns.nunique()
```

[65]:
```
last_scraped                        2
name                            36870
description                     34133
neighborhood_overview           18616
host_name                        9123
host_since                       4289
host_location                    1747
host_about                      14424
host_response_rate                 88
host_acceptance_rate              101
host_is_superhost                   2
host_neighbourhood                484
host_verifications                526
host_has_profile_pic                2
host_identity_verified              2
neighbourhood                     207
neighbourhood_cleansed            222
neighbourhood_group_cleansed        5
property_type                      78
room_type                           4
bathrooms_text                     30
amenities                       31740
has_availability                    2
calendar_last_scraped               2
first_review                     3171
last_review                      2560
license                             1
instant_bookable                    2
dtype: int64
```

Task: Based on your findings, identify features that you think should be transformed using one-hot encoding.

1. Use the code cell below to inspect the unique *values* that each of these features have.

[66]:
```python
# YOUR CODE HERE
# List of candidate columns for one-hot encoding
to_one_hot = [
    'host_is_superhost',
    'host_has_profile_pic',
    'host_identity_verified',
    'neighbourhood_group_cleansed',
    'room_type',
    'has_availability',
    'instant_bookable',
    'property_type',
```

```python
    'bathrooms_text'
]

# Print unique values for each selected column
for col in to_one_hot:
    print(f"\nUnique values in '{col}':")
    print(df[col].unique())
```

```
Unique values in 'host_is_superhost':
['f' 't' nan]

Unique values in 'host_has_profile_pic':
['t' 'f' nan]

Unique values in 'host_identity_verified':
['t' 'f' nan]

Unique values in 'neighbourhood_group_cleansed':
['Manhattan' 'Brooklyn' 'Queens' 'Staten Island' 'Bronx']

Unique values in 'room_type':
['Entire home/apt' 'Private room' 'Hotel room' 'Shared room']

Unique values in 'has_availability':
['t' 'f']

Unique values in 'instant_bookable':
['f' 't']

Unique values in 'property_type':
['Entire rental unit' 'Entire guest suite' 'Private room in rental unit'
 'Private room in townhouse' 'Private room in condominium (condo)'
 'Private room in loft' 'Entire loft' 'Private room in residential home'
 'Entire condominium (condo)' 'Entire residential home' 'Entire townhouse'
 'Private room in bed and breakfast' 'Entire guesthouse'
 'Private room in guest suite' 'Room in boutique hotel'
 'Shared room in loft' 'Shared room in rental unit'
 'Shared room in residential home' 'Private room' 'Private room in hostel'
 'Entire place' 'Private room in guesthouse' 'Boat'
 'Entire serviced apartment' 'Room in aparthotel' 'Floor'
 'Private room in vacation home' 'Room in serviced apartment'
 'Entire cottage' 'Private room in serviced apartment' 'Room in hotel'
 'Cave' 'Tiny house' 'Private room in floor'
 'Shared room in condominium (condo)' 'Entire bungalow'
 'Private room in casa particular' 'Shared room in townhouse' 'Houseboat'
 'Private room in bungalow' 'Entire villa' 'Private room in resort'
```

```
 'Shared room in guest suite' 'Private room in castle'
 'Private room in villa' 'Shared room in floor' 'Entire bed and breakfast'
 'Entire home/apt' 'Private room in tiny house' 'Private room in tent'
 'Private room in in-law' 'Private room in barn' 'Shared room in hostel'
 'Camper/RV' 'Room in resort' 'Shared room in guesthouse' 'Bus'
 'Shared room in bed and breakfast' 'Private room in farm stay'
 'Private room in dorm' 'Room in bed and breakfast'
 'Shared room in island' 'Shared room in bungalow'
 'Shared room in serviced apartment' 'Private room in earth house'
 'Lighthouse' 'Private room in train' 'Barn' 'Private room in lighthouse'
 'Entire cabin' 'Private room in camper/rv' 'Castle' 'Tent' 'Tower'
 'Casa particular' 'Shared room in casa particular'
 'Private room in cycladic house' 'Entire vacation home']

Unique values in 'bathrooms_text':
['1 bath' nan '1.5 baths' '1 shared bath' '1 private bath'
 'Shared half-bath' '2 baths' '1.5 shared baths' '3 baths' 'Half-bath'
 '2.5 baths' '2 shared baths' '0 baths' '4 baths' '0 shared baths'
 'Private half-bath' '5 baths' '4.5 baths' '5.5 baths' '2.5 shared baths'
 '3.5 baths' '3 shared baths' '4 shared baths' '6 baths'
 '3.5 shared baths' '4.5 shared baths' '7.5 baths' '6.5 baths' '8 baths'
 '7 baths' '6 shared baths']
```

2. List these features and explain why they would be suitable for one-hot encoding. Note your findings in the markdown cell below.

Task: In the code cell below, one-hot encode one of the features you have identified and replace the original column in DataFrame `df` with the new one-hot encoded columns.

[67]:
```python
# YOUR CODE HERE
# One-hot encode 'room_type'
df_room_type = pd.get_dummies(df['room_type'])

# Concatenate the new one-hot encoded DataFrame to the original DataFrame
df = pd.concat([df, df_room_type], axis=1)

# Drop the original 'room_type' column
df.drop(columns='room_type', inplace=True)

# Display the updated DataFrame columns to confirm the changes
df.columns
```

[67]:
```
Index(['id', 'scrape_id', 'last_scraped', 'name', 'description',
       'neighborhood_overview', 'host_id', 'host_name', 'host_since',
       'host_location', 'host_about', 'host_response_rate',
       'host_acceptance_rate', 'host_is_superhost', 'host_neighbourhood',
       'host_listings_count', 'host_total_listings_count',
       'host_verifications', 'host_has_profile_pic', 'host_identity_verified',
```

```
       'neighbourhood', 'neighbourhood_cleansed',
       'neighbourhood_group_cleansed', 'latitude', 'longitude',
       'property_type', 'accommodates', 'bathrooms', 'bathrooms_text',
       'bedrooms', 'beds', 'amenities', 'price', 'minimum_nights',
       'maximum_nights', 'minimum_minimum_nights', 'maximum_minimum_nights',
       'minimum_maximum_nights', 'maximum_maximum_nights',
       'minimum_nights_avg_ntm', 'maximum_nights_avg_ntm', 'calendar_updated',
       'has_availability', 'availability_30', 'availability_60',
       'availability_90', 'availability_365', 'calendar_last_scraped',
       'number_of_reviews', 'number_of_reviews_ltm', 'number_of_reviews_l30d',
       'first_review', 'last_review', 'review_scores_rating',
       'review_scores_accuracy', 'review_scores_cleanliness',
       'review_scores_checkin', 'review_scores_communication',
       'review_scores_location', 'review_scores_value', 'license',
       'instant_bookable', 'calculated_host_listings_count',
       'calculated_host_listings_count_entire_homes',
       'calculated_host_listings_count_private_rooms',
       'calculated_host_listings_count_shared_rooms', 'reviews_per_month',
       'label_price', 'host_listings_count_na', 'host_total_listings_count_na',
       'bathrooms_na', 'bedrooms_na', 'beds_na', 'a few days or more',
       'unavailable', 'within a day', 'within a few hours', 'within an hour',
       'Entire home/apt', 'Hotel room', 'Private room', 'Shared room'],
      dtype='object')
```

1.4 Part 4. Explore Your Data

You will now perform exploratory data analysis in preparation for selecting your features as part of feature engineering.

Identify Correlations We will focus on identifying which features in the data have the highest correlation with the label.

Let's first run the `corr()` method on DataFrame `df` and save the result to the variable `corr_matrix`. Let's round the resulting correlations to five decimal places:

[75]:
```
corr_matrix = round(df.corr(),5)
corr_matrix
```

[75]:
```
                                                   id  scrape_id  host_id  \
id                                            1.00000       -0.0  0.58617
scrape_id                                    -0.00000        1.0  0.00000
host_id                                       0.58617        0.0  1.00000
host_listings_count                           0.12986       -0.0  0.03189
host_total_listings_count                     0.12986       -0.0  0.03189
latitude                                      0.01000        0.0  0.04148
longitude                                     0.08708       -0.0  0.11620
```

accommodates	0.03540	0.0	0.02723
bathrooms	NaN	NaN	NaN
bedrooms	0.04503	0.0	0.02202
beds	0.03289	0.0	0.03689
price	0.04256	-0.0	0.02907
minimum_nights	-0.12067	0.0	-0.10640
maximum_nights	-0.00696	0.0	-0.00385
minimum_minimum_nights	-0.10234	0.0	-0.09188
maximum_minimum_nights	-0.00041	-0.0	-0.04521
minimum_maximum_nights	0.00747	-0.0	0.02572
maximum_maximum_nights	0.01461	0.0	0.04267
minimum_nights_avg_ntm	-0.00338	-0.0	-0.04707
maximum_nights_avg_ntm	0.01149	0.0	0.03438
calendar_updated	NaN	NaN	NaN
availability_30	0.25190	-0.0	0.26850
availability_60	0.32793	-0.0	0.32728
availability_90	0.34401	-0.0	0.33395
availability_365	0.28722	0.0	0.27332
number_of_reviews	-0.29164	0.0	-0.12215
number_of_reviews_ltm	0.07737	0.0	0.11469
number_of_reviews_l30d	0.15257	-0.0	0.15333
review_scores_rating	0.01187	0.0	-0.04397
review_scores_accuracy	-0.08867	0.0	-0.15428
review_scores_cleanliness	0.00424	0.0	-0.05183
review_scores_checkin	-0.09156	0.0	-0.14890
review_scores_communication	-0.11950	0.0	-0.17420
review_scores_location	0.00322	0.0	-0.07864
review_scores_value	-0.07080	0.0	-0.13340
calculated_host_listings_count	0.23667	-0.0	0.15754
calculated_host_listings_count_entire_homes	0.13713	0.0	0.02524
calculated_host_listings_count_private_rooms	0.21188	-0.0	0.19320
calculated_host_listings_count_shared_rooms	0.04671	-0.0	0.07831
reviews_per_month	0.23169	0.0	0.20844
label_price	0.07907	-0.0	0.04053
host_listings_count_na	NaN	NaN	NaN
host_total_listings_count_na	NaN	NaN	NaN
bathrooms_na	NaN	NaN	NaN
bedrooms_na	NaN	NaN	NaN
beds_na	NaN	NaN	NaN
a few days or more	0.01215	0.0	0.04055
unavailable	-0.35410	-0.0	-0.24094
within a day	-0.01164	-0.0	-0.05562
within a few hours	0.12780	-0.0	0.01844
within an hour	0.29187	-0.0	0.26491
Entire home/apt	-0.04284	-0.0	-0.12862
Hotel room	0.01698	0.0	0.07086
Private room	0.03813	0.0	0.10957

```
Shared room                                      0.00958       0.0  0.03676

                                              host_listings_count  \
id                                                        0.12986
scrape_id                                                -0.00000
host_id                                                   0.03189
host_listings_count                                       1.00000
host_total_listings_count                                 1.00000
latitude                                                  0.03475
longitude                                                -0.08843
accommodates                                             -0.02621
bathrooms                                                     NaN
bedrooms                                                 -0.01710
beds                                                     -0.03151
price                                                     0.07492
minimum_nights                                            0.19739
maximum_nights                                           -0.00080
minimum_minimum_nights                                    0.26125
maximum_minimum_nights                                    0.65300
minimum_maximum_nights                                   -0.00349
maximum_maximum_nights                                   -0.00529
minimum_nights_avg_ntm                                    0.65239
maximum_nights_avg_ntm                                   -0.00451
calendar_updated                                              NaN
availability_30                                           0.07148
availability_60                                           0.06218
availability_90                                           0.06279
availability_365                                          0.14287
number_of_reviews                                        -0.06617
number_of_reviews_ltm                                    -0.04448
number_of_reviews_l30d                                   -0.04962
review_scores_rating                                     -0.00742
review_scores_accuracy                                   -0.02365
review_scores_cleanliness                                -0.00694
review_scores_checkin                                    -0.01701
review_scores_communication                              -0.05032
review_scores_location                                    0.00638
review_scores_value                                      -0.07391
calculated_host_listings_count                            0.42944
calculated_host_listings_count_entire_homes               0.54188
calculated_host_listings_count_private_rooms              0.14915
calculated_host_listings_count_shared_rooms              -0.01595
reviews_per_month                                        -0.02096
label_price                                               0.13104
host_listings_count_na                                        NaN
host_total_listings_count_na                                  NaN
bathrooms_na                                                  NaN
```

```
bedrooms_na                                        NaN
beds_na                                            NaN
a few days or more                            -0.03124
unavailable                                   -0.11686
within a day                                  -0.03119
within a few hours                            -0.01468
within an hour                                 0.17132
Entire home/apt                                0.01040
Hotel room                                    -0.00877
Private room                                  -0.00468
Shared room                                   -0.01825

                              host_total_listings_count  \
id                                                0.12986
scrape_id                                        -0.00000
host_id                                           0.03189
host_listings_count                               1.00000
host_total_listings_count                         1.00000
latitude                                          0.03475
longitude                                        -0.08843
accommodates                                     -0.02621
bathrooms                                             NaN
bedrooms                                         -0.01710
beds                                             -0.03151
price                                             0.07492
minimum_nights                                    0.19739
maximum_nights                                   -0.00080
minimum_minimum_nights                            0.26125
maximum_minimum_nights                            0.65300
minimum_maximum_nights                           -0.00349
maximum_maximum_nights                           -0.00529
minimum_nights_avg_ntm                            0.65239
maximum_nights_avg_ntm                           -0.00451
calendar_updated                                      NaN
availability_30                                   0.07148
availability_60                                   0.06218
availability_90                                   0.06279
availability_365                                  0.14287
number_of_reviews                                -0.06617
number_of_reviews_ltm                            -0.04448
number_of_reviews_l30d                           -0.04962
review_scores_rating                             -0.00742
review_scores_accuracy                           -0.02365
review_scores_cleanliness                        -0.00694
review_scores_checkin                            -0.01701
review_scores_communication                      -0.05032
review_scores_location                            0.00638
```

review_scores_value	-0.07391
calculated_host_listings_count	0.42944
calculated_host_listings_count_entire_homes	0.54188
calculated_host_listings_count_private_rooms	0.14915
calculated_host_listings_count_shared_rooms	-0.01595
reviews_per_month	-0.02096
label_price	0.13104
host_listings_count_na	NaN
host_total_listings_count_na	NaN
bathrooms_na	NaN
bedrooms_na	NaN
beds_na	NaN
a few days or more	-0.03124
unavailable	-0.11686
within a day	-0.03119
within a few hours	-0.01468
within an hour	0.17132
Entire home/apt	0.01040
Hotel room	-0.00877
Private room	-0.00468
Shared room	-0.01825

	latitude	longitude \
id	0.01000	0.08708
scrape_id	0.00000	-0.00000
host_id	0.04148	0.11620
host_listings_count	0.03475	-0.08843
host_total_listings_count	0.03475	-0.08843
latitude	1.00000	0.05718
longitude	0.05718	1.00000
accommodates	-0.04745	0.00374
bathrooms	NaN	NaN
bedrooms	-0.07150	0.00752
beds	-0.05388	0.03136
price	0.02734	-0.11484
minimum_nights	0.03422	-0.08550
maximum_nights	0.00561	-0.00296
minimum_minimum_nights	0.03317	-0.08397
maximum_minimum_nights	0.04352	-0.09520
minimum_maximum_nights	0.01735	-0.00780
maximum_maximum_nights	0.01598	-0.01993
minimum_nights_avg_ntm	0.04379	-0.09507
maximum_nights_avg_ntm	0.01828	-0.01401
calendar_updated	NaN	NaN
availability_30	0.00261	0.13025
availability_60	0.00026	0.15062
availability_90	-0.00157	0.14953

availability_365	0.01383	0.09596
number_of_reviews	-0.04801	0.06759
number_of_reviews_ltm	-0.04884	0.06458
number_of_reviews_l30d	-0.04339	0.07309
review_scores_rating	-0.03767	0.00523
review_scores_accuracy	-0.04076	-0.01136
review_scores_cleanliness	-0.03469	0.00772
review_scores_checkin	-0.04612	-0.00525
review_scores_communication	-0.04250	-0.01358
review_scores_location	0.01355	-0.13822
review_scores_value	-0.04887	0.00052
calculated_host_listings_count	0.07954	-0.06543
calculated_host_listings_count_entire_homes	0.07065	-0.12713
calculated_host_listings_count_private_rooms	0.05096	0.01401
calculated_host_listings_count_shared_rooms	0.00762	0.02066
reviews_per_month	-0.03667	0.07121
label_price	0.04330	-0.20695
host_listings_count_na	NaN	NaN
host_total_listings_count_na	NaN	NaN
bathrooms_na	NaN	NaN
bedrooms_na	NaN	NaN
beds_na	NaN	NaN
a few days or more	0.02052	-0.01400
unavailable	0.01134	-0.07471
within a day	0.01410	-0.03805
within a few hours	-0.00499	0.03534
within an hour	-0.02598	0.08358
Entire home/apt	-0.02656	-0.14909
Hotel room	0.02825	-0.04860
Private room	0.01830	0.15128
Shared room	0.01707	0.02280

	accommodates	bathrooms \
id	0.03540	NaN
scrape_id	0.00000	NaN
host_id	0.02723	NaN
host_listings_count	-0.02621	NaN
host_total_listings_count	-0.02621	NaN
latitude	-0.04745	NaN
longitude	0.00374	NaN
accommodates	1.00000	NaN
bathrooms	NaN	NaN
bedrooms	0.70586	NaN
beds	0.73665	NaN
price	0.30803	NaN
minimum_nights	-0.08474	NaN
maximum_nights	-0.00494	NaN

```
minimum_minimum_nights                        -0.07485      NaN
maximum_minimum_nights                        -0.05134      NaN
minimum_maximum_nights                        -0.00249      NaN
maximum_maximum_nights                        -0.00931      NaN
minimum_nights_avg_ntm                        -0.05266      NaN
maximum_nights_avg_ntm                        -0.00558      NaN
calendar_updated                                   NaN      NaN
availability_30                                0.04429      NaN
availability_60                                0.07983      NaN
availability_90                                0.09096      NaN
availability_365                               0.10293      NaN
number_of_reviews                              0.07255      NaN
number_of_reviews_ltm                          0.08118      NaN
number_of_reviews_l30d                         0.08552      NaN
review_scores_rating                           0.03097      NaN
review_scores_accuracy                        -0.00422      NaN
review_scores_cleanliness                      0.03702      NaN
review_scores_checkin                         -0.00125      NaN
review_scores_communication                   -0.00067      NaN
review_scores_location                        -0.01220      NaN
review_scores_value                           -0.00778      NaN
calculated_host_listings_count                -0.11818      NaN
calculated_host_listings_count_entire_homes   -0.01929      NaN
calculated_host_listings_count_private_rooms  -0.14499      NaN
calculated_host_listings_count_shared_rooms   -0.05161      NaN
reviews_per_month                              0.06850      NaN
label_price                                    0.50062      NaN
host_listings_count_na                             NaN      NaN
host_total_listings_count_na                       NaN      NaN
bathrooms_na                                       NaN      NaN
bedrooms_na                                        NaN      NaN
beds_na                                            NaN      NaN
a few days or more                             0.01101      NaN
unavailable                                   -0.11168      NaN
within a day                                   0.01642      NaN
within a few hours                            -0.00382      NaN
within an hour                                 0.11060      NaN
Entire home/apt                                0.45742      NaN
Hotel room                                    -0.01671      NaN
Private room                                  -0.44105      NaN
Shared room                                   -0.06358      NaN

                                            bedrooms  …  beds_na  \
id                                           0.04503  …      NaN
scrape_id                                    0.00000  …      NaN
host_id                                      0.02202  …      NaN
host_listings_count                         -0.01710  …      NaN
```

host_total_listings_count	-0.01710	…	NaN
latitude	-0.07150	…	NaN
longitude	0.00752	…	NaN
accommodates	0.70586	…	NaN
bathrooms	NaN	…	NaN
bedrooms	1.00000	…	NaN
beds	0.72914	…	NaN
price	0.25383	…	NaN
minimum_nights	-0.02749	…	NaN
maximum_nights	0.00002	…	NaN
minimum_minimum_nights	-0.02546	…	NaN
maximum_minimum_nights	-0.01708	…	NaN
minimum_maximum_nights	-0.01161	…	NaN
maximum_maximum_nights	-0.01705	…	NaN
minimum_nights_avg_ntm	-0.01782	…	NaN
maximum_nights_avg_ntm	-0.01465	…	NaN
calendar_updated	NaN	…	NaN
availability_30	0.01816	…	NaN
availability_60	0.04432	…	NaN
availability_90	0.05567	…	NaN
availability_365	0.08280	…	NaN
number_of_reviews	0.00408	…	NaN
number_of_reviews_ltm	0.02836	…	NaN
number_of_reviews_l30d	0.03271	…	NaN
review_scores_rating	0.01686	…	NaN
review_scores_accuracy	-0.00323	…	NaN
review_scores_cleanliness	0.03206	…	NaN
review_scores_checkin	0.00638	…	NaN
review_scores_communication	-0.00019	…	NaN
review_scores_location	-0.01053	…	NaN
review_scores_value	0.00074	…	NaN
calculated_host_listings_count	-0.05754	…	NaN
calculated_host_listings_count_entire_homes	-0.00212	…	NaN
calculated_host_listings_count_private_rooms	-0.07591	…	NaN
calculated_host_listings_count_shared_rooms	-0.04902	…	NaN
reviews_per_month	0.03030	…	NaN
label_price	0.41996	…	NaN
host_listings_count_na	NaN	…	NaN
host_total_listings_count_na	NaN	…	NaN
bathrooms_na	NaN	…	NaN
bedrooms_na	NaN	…	NaN
beds_na	NaN	…	NaN
a few days or more	0.01969	…	NaN
unavailable	-0.09343	…	NaN
within a day	0.03512	…	NaN
within a few hours	0.01114	…	NaN
within an hour	0.06432	…	NaN

Entire home/apt	0.35604	…	NaN
Hotel room	-0.02448	…	NaN
Private room	-0.33917	…	NaN
Shared room	-0.05944	…	NaN

	a few days or more	unavailable	\
id	0.01215	-0.35410	
scrape_id	0.00000	-0.00000	
host_id	0.04055	-0.24094	
host_listings_count	-0.03124	-0.11686	
host_total_listings_count	-0.03124	-0.11686	
latitude	0.02052	0.01134	
longitude	-0.01400	-0.07471	
accommodates	0.01101	-0.11168	
bathrooms	NaN	NaN	
bedrooms	0.01969	-0.09343	
beds	0.02056	-0.10810	
price	0.02432	-0.05266	
minimum_nights	0.03087	0.18254	
maximum_nights	-0.00108	0.00577	
minimum_minimum_nights	0.02434	0.15024	
maximum_minimum_nights	-0.00457	0.00076	
minimum_maximum_nights	-0.00543	-0.00942	
maximum_maximum_nights	-0.00845	-0.02371	
minimum_nights_avg_ntm	-0.00364	0.00547	
maximum_nights_avg_ntm	-0.00717	-0.01380	
calendar_updated	NaN	NaN	
availability_30	0.20254	-0.29428	
availability_60	0.18352	-0.43295	
availability_90	0.17710	-0.47929	
availability_365	0.12545	-0.47520	
number_of_reviews	-0.03115	-0.16121	
number_of_reviews_ltm	-0.06060	-0.22794	
number_of_reviews_l30d	-0.07216	-0.24822	
review_scores_rating	-0.06101	-0.09901	
review_scores_accuracy	-0.07606	0.04080	
review_scores_cleanliness	-0.06482	-0.06196	
review_scores_checkin	-0.08196	0.02230	
review_scores_communication	-0.08031	0.05199	
review_scores_location	-0.04102	0.01118	
review_scores_value	-0.06118	0.04111	
calculated_host_listings_count	-0.05406	-0.08352	
calculated_host_listings_count_entire_homes	-0.04190	-0.14256	
calculated_host_listings_count_private_rooms	-0.03991	0.00213	
calculated_host_listings_count_shared_rooms	0.02082	-0.01928	
reviews_per_month	-0.04892	-0.20592	
label_price	0.00792	-0.10279	

host_listings_count_na	NaN	NaN
host_total_listings_count_na	NaN	NaN
bathrooms_na	NaN	NaN
bedrooms_na	NaN	NaN
beds_na	NaN	NaN
a few days or more	1.00000	-0.18787
unavailable	-0.18787	1.00000
within a day	-0.06088	-0.26424
within a few hours	-0.08364	-0.36305
within an hour	-0.13339	-0.57898
Entire home/apt	-0.00872	-0.04946
Hotel room	-0.01191	-0.03010
Private room	0.00571	0.05008
Shared room	0.01973	0.01648

	within a day \
id	-0.01164
scrape_id	-0.00000
host_id	-0.05562
host_listings_count	-0.03119
host_total_listings_count	-0.03119
latitude	0.01410
longitude	-0.03805
accommodates	0.01642
bathrooms	NaN
bedrooms	0.03512
beds	0.01886
price	-0.00026
minimum_nights	-0.00695
maximum_nights	-0.00153
minimum_minimum_nights	-0.01002
maximum_minimum_nights	-0.02714
minimum_maximum_nights	0.02956
maximum_maximum_nights	0.02398
minimum_nights_avg_ntm	-0.02691
maximum_nights_avg_ntm	0.02215
calendar_updated	NaN
availability_30	0.04232
availability_60	0.05946
availability_90	0.08130
availability_365	0.10797
number_of_reviews	0.00818
number_of_reviews_ltm	-0.03950
number_of_reviews_l30d	-0.05445
review_scores_rating	0.02862
review_scores_accuracy	0.00761
review_scores_cleanliness	0.01355

review_scores_checkin	0.01290
review_scores_communication	-0.00556
review_scores_location	0.00999
review_scores_value	-0.00564
calculated_host_listings_count	-0.01243
calculated_host_listings_count_entire_homes	0.04999
calculated_host_listings_count_private_rooms	-0.05728
calculated_host_listings_count_shared_rooms	-0.01131
reviews_per_month	-0.04801
label_price	0.01335
host_listings_count_na	NaN
host_total_listings_count_na	NaN
bathrooms_na	NaN
bedrooms_na	NaN
beds_na	NaN
a few days or more	-0.06088
unavailable	-0.26424
within a day	1.00000
within a few hours	-0.11764
within an hour	-0.18761
Entire home/apt	0.06668
Hotel room	0.00451
Private room	-0.06601
Shared room	-0.00648

	within a few hours \
id	0.12780
scrape_id	-0.00000
host_id	0.01844
host_listings_count	-0.01468
host_total_listings_count	-0.01468
latitude	-0.00499
longitude	0.03534
accommodates	-0.00382
bathrooms	NaN
bedrooms	0.01114
beds	0.00242
price	-0.01433
minimum_nights	0.00592
maximum_nights	-0.00210
minimum_minimum_nights	-0.00678
maximum_minimum_nights	-0.02551
minimum_maximum_nights	-0.01049
maximum_maximum_nights	-0.01634
minimum_nights_avg_ntm	-0.02692
maximum_nights_avg_ntm	-0.01386
calendar_updated	NaN

availability_30	0.10312
availability_60	0.13745
availability_90	0.14265
availability_365	0.17218
number_of_reviews	-0.00846
number_of_reviews_ltm	-0.02346
number_of_reviews_l30d	-0.02925
review_scores_rating	0.02229
review_scores_accuracy	-0.04651
review_scores_cleanliness	-0.01300
review_scores_checkin	-0.01974
review_scores_communication	-0.04243
review_scores_location	-0.01360
review_scores_value	-0.05498
calculated_host_listings_count	0.09949
calculated_host_listings_count_entire_homes	0.01930
calculated_host_listings_count_private_rooms	0.11927
calculated_host_listings_count_shared_rooms	0.01389
reviews_per_month	-0.02663
label_price	-0.02111
host_listings_count_na	NaN
host_total_listings_count_na	NaN
bathrooms_na	NaN
bedrooms_na	NaN
beds_na	NaN
a few days or more	-0.08364
unavailable	-0.36305
within a day	-0.11764
within a few hours	1.00000
within an hour	-0.25777
Entire home/apt	0.00195
Hotel room	-0.01658
Private room	0.00196
Shared room	-0.00597

	within an hour	Entire home/apt \
id	0.29187	-0.04284
scrape_id	-0.00000	-0.00000
host_id	0.26491	-0.12862
host_listings_count	0.17132	0.01040
host_total_listings_count	0.17132	0.01040
latitude	-0.02598	-0.02656
longitude	0.08358	-0.14909
accommodates	0.11060	0.45742
bathrooms	NaN	NaN
bedrooms	0.06432	0.35604
beds	0.09628	0.32487

price	0.05805	0.17365
minimum_nights	-0.21377	0.00925
maximum_nights	-0.00334	0.00478
minimum_minimum_nights	-0.16408	0.02079
maximum_minimum_nights	0.03672	0.07891
minimum_maximum_nights	0.00315	-0.02184
maximum_maximum_nights	0.02789	-0.03952
minimum_nights_avg_ntm	0.03209	0.07834
maximum_nights_avg_ntm	0.01568	-0.03164
calendar_updated	NaN	NaN
availability_30	0.12962	-0.10800
availability_60	0.25344	-0.08439
availability_90	0.29008	-0.06442
availability_365	0.26996	-0.00816
number_of_reviews	0.19174	0.02319
number_of_reviews_ltm	0.31743	0.02510
number_of_reviews_l30d	0.35797	0.03656
review_scores_rating	0.09629	0.08109
review_scores_accuracy	0.01555	0.09148
review_scores_cleanliness	0.09180	0.10695
review_scores_checkin	0.01498	0.07370
review_scores_communication	0.01028	0.08425
review_scores_location	0.00806	0.09444
review_scores_value	0.02334	0.04539
calculated_host_listings_count	0.04675	-0.04794
calculated_host_listings_count_entire_homes	0.13010	0.16276
calculated_host_listings_count_private_rooms	-0.04169	-0.19529
calculated_host_listings_count_shared_rooms	0.00810	-0.11059
reviews_per_month	0.28524	-0.00268
label_price	0.11721	0.33529
host_listings_count_na	NaN	NaN
host_total_listings_count_na	NaN	NaN
bathrooms_na	NaN	NaN
bedrooms_na	NaN	NaN
beds_na	NaN	NaN
a few days or more	-0.13339	-0.00872
unavailable	-0.57898	-0.04946
within a day	-0.18761	0.06668
within a few hours	-0.25777	0.00195
within an hour	1.00000	0.01693
Entire home/apt	0.01693	1.00000
Hotel room	0.04812	-0.07933
Private room	-0.01967	-0.95966
Shared room	-0.01831	-0.13155

	Hotel room	Private room \
id	0.01698	0.03813

scrape_id	0.00000	0.00000
host_id	0.07086	0.10957
host_listings_count	-0.00877	-0.00468
host_total_listings_count	-0.00877	-0.00468
latitude	0.02825	0.01830
longitude	-0.04860	0.15128
accommodates	-0.01671	-0.44105
bathrooms	NaN	NaN
bedrooms	-0.02448	-0.33917
beds	-0.01256	-0.32660
price	0.05119	-0.18024
minimum_nights	-0.03447	-0.00313
maximum_nights	-0.00039	-0.00458
minimum_minimum_nights	-0.02844	-0.01574
maximum_minimum_nights	-0.01886	-0.07349
minimum_maximum_nights	0.14009	0.00279
maximum_maximum_nights	0.11571	0.02444
minimum_nights_avg_ntm	-0.01984	-0.07289
maximum_nights_avg_ntm	0.15595	0.01063
calendar_updated	NaN	NaN
availability_30	0.04272	0.08909
availability_60	0.03851	0.06938
availability_90	0.03578	0.05103
availability_365	0.05067	-0.00435
number_of_reviews	0.03582	-0.02639
number_of_reviews_ltm	0.08765	-0.03482
number_of_reviews_l30d	0.00086	-0.03389
review_scores_rating	-0.01071	-0.07572
review_scores_accuracy	-0.03556	-0.08241
review_scores_cleanliness	0.00819	-0.10530
review_scores_checkin	-0.02068	-0.06553
review_scores_communication	-0.02970	-0.07540
review_scores_location	0.01197	-0.09296
review_scores_value	-0.03393	-0.03770
calculated_host_listings_count	-0.00784	0.05666
calculated_host_listings_count_entire_homes	-0.00853	-0.15528
calculated_host_listings_count_private_rooms	-0.01535	0.20438
calculated_host_listings_count_shared_rooms	-0.00835	-0.04520
reviews_per_month	0.03322	-0.00053
label_price	0.10587	-0.34108
host_listings_count_na	NaN	NaN
host_total_listings_count_na	NaN	NaN
bathrooms_na	NaN	NaN
bedrooms_na	NaN	NaN
beds_na	NaN	NaN
a few days or more	-0.01191	0.00571
unavailable	-0.03010	0.05008

within a day	0.00451	-0.06601
within a few hours	-0.01658	0.00196
within an hour	0.04812	-0.01967
Entire home/apt	-0.07933	-0.95966
Hotel room	1.00000	-0.06674
Private room	-0.06674	1.00000
Shared room	-0.00915	-0.11067

	Shared room
id	0.00958
scrape_id	0.00000
host_id	0.03676
host_listings_count	-0.01825
host_total_listings_count	-0.01825
latitude	0.01707
longitude	0.02280
accommodates	-0.06358
bathrooms	NaN
bedrooms	-0.05944
beds	0.01000
price	-0.00669
minimum_nights	-0.00423
maximum_nights	-0.00064
minimum_minimum_nights	-0.00443
maximum_minimum_nights	-0.01233
minimum_maximum_nights	-0.00322
maximum_maximum_nights	-0.00500
minimum_nights_avg_ntm	-0.01192
maximum_nights_avg_ntm	-0.00425
calendar_updated	NaN
availability_30	0.05305
availability_60	0.03931
availability_90	0.03405
availability_365	0.02056
number_of_reviews	-0.00903
number_of_reviews_ltm	-0.01391
number_of_reviews_l30d	-0.01197
review_scores_rating	-0.01767
review_scores_accuracy	-0.01757
review_scores_cleanliness	-0.01476
review_scores_checkin	-0.02305
review_scores_communication	-0.02031
review_scores_location	-0.01618
review_scores_value	-0.01173
calculated_host_listings_count	-0.03027
calculated_host_listings_count_entire_homes	-0.02785
calculated_host_listings_count_private_rooms	-0.02503

```
calculated_host_listings_count_shared_rooms        0.64509
reviews_per_month                                 -0.00766
label_price                                       -0.04563
host_listings_count_na                                 NaN
host_total_listings_count_na                           NaN
bathrooms_na                                           NaN
bedrooms_na                                            NaN
beds_na                                                NaN
a few days or more                                 0.01973
unavailable                                        0.01648
within a day                                      -0.00648
within a few hours                                -0.00597
within an hour                                    -0.01831
Entire home/apt                                   -0.13155
Hotel room                                        -0.00915
Private room                                      -0.11067
Shared room                                        1.00000

[55 rows x 55 columns]
```

The result is a computed *correlation matrix*. The values on the diagonal are all equal to 1 because they represent the correlations between each column with itself. The matrix is symmetrical with respect to the diagonal.

We only need to observe correlations of all features with the column `label_price` (as opposed to every possible pairwise correlation). Se let's query the `label_price` column of this matrix:

Task: Extract the `label_price` column of the correlation matrix and save the results to the variable `corrs`.

[78]:
```
# Extract the 'label_price' column from the correlation matrix
corrs = corr_matrix['label_price']
corrs
```

[78]:
```
id                                  0.07907
scrape_id                          -0.00000
host_id                             0.04053
host_listings_count                 0.13104
host_total_listings_count           0.13104
latitude                            0.04330
longitude                          -0.20695
accommodates                        0.50062
bathrooms                               NaN
bedrooms                            0.41996
beds                                0.37370
price                               0.71112
minimum_nights                     -0.07589
maximum_nights                     -0.00097
```

```
minimum_minimum_nights                          -0.03804
maximum_minimum_nights                           0.06554
minimum_maximum_nights                           0.06582
maximum_maximum_nights                           0.11169
minimum_nights_avg_ntm                           0.06388
maximum_nights_avg_ntm                           0.08210
calendar_updated                                     NaN
availability_30                                  0.14569
availability_60                                  0.14701
availability_90                                  0.14391
availability_365                                 0.12356
number_of_reviews                               -0.04197
number_of_reviews_ltm                            0.02757
number_of_reviews_l30d                           0.02159
review_scores_rating                             0.04320
review_scores_accuracy                           0.00536
review_scores_cleanliness                        0.08254
review_scores_checkin                           -0.00367
review_scores_communication                      0.00012
review_scores_location                           0.09724
review_scores_value                             -0.00482
calculated_host_listings_count                  -0.01582
calculated_host_listings_count_entire_homes      0.09509
calculated_host_listings_count_private_rooms    -0.09978
calculated_host_listings_count_shared_rooms     -0.04334
reviews_per_month                                0.03114
label_price                                      1.00000
host_listings_count_na                               NaN
host_total_listings_count_na                         NaN
bathrooms_na                                         NaN
bedrooms_na                                          NaN
beds_na                                              NaN
a few days or more                               0.00792
unavailable                                     -0.10279
within a day                                     0.01335
within a few hours                              -0.02111
within an hour                                   0.11721
Entire home/apt                                  0.33529
Hotel room                                       0.10587
Private room                                    -0.34108
Shared room                                     -0.04563
Name: label_price, dtype: float64
```

Task: Sort the values of the series we just obtained in the descending order and save the results to the variable `corrs_sorted`.

[79]:
```
corrs_sorted = corrs.sort_values(ascending=False)
corrs_sorted
```

[79]:
```
label_price                                        1.00000
price                                              0.71112
accommodates                                       0.50062
bedrooms                                           0.41996
beds                                               0.37370
Entire home/apt                                    0.33529
availability_60                                    0.14701
availability_30                                    0.14569
availability_90                                    0.14391
host_listings_count                                0.13104
host_total_listings_count                          0.13104
availability_365                                   0.12356
within an hour                                     0.11721
maximum_maximum_nights                             0.11169
Hotel room                                         0.10587
review_scores_location                             0.09724
calculated_host_listings_count_entire_homes        0.09509
review_scores_cleanliness                          0.08254
maximum_nights_avg_ntm                             0.08210
id                                                 0.07907
minimum_maximum_nights                             0.06582
maximum_minimum_nights                             0.06554
minimum_nights_avg_ntm                             0.06388
latitude                                           0.04330
review_scores_rating                               0.04320
host_id                                            0.04053
reviews_per_month                                  0.03114
number_of_reviews_ltm                              0.02757
number_of_reviews_l30d                             0.02159
within a day                                       0.01335
a few days or more                                 0.00792
review_scores_accuracy                             0.00536
review_scores_communication                        0.00012
scrape_id                                         -0.00000
maximum_nights                                    -0.00097
review_scores_checkin                             -0.00367
review_scores_value                               -0.00482
calculated_host_listings_count                    -0.01582
within a few hours                                -0.02111
minimum_minimum_nights                            -0.03804
number_of_reviews                                 -0.04197
calculated_host_listings_count_shared_rooms       -0.04334
Shared room                                       -0.04563
minimum_nights                                    -0.07589
```

```
calculated_host_listings_count_private_rooms   -0.09978
unavailable                                    -0.10279
longitude                                      -0.20695
Private room                                   -0.34108
bathrooms                                           NaN
calendar_updated                                    NaN
host_listings_count_na                              NaN
host_total_listings_count_na                        NaN
bathrooms_na                                        NaN
bedrooms_na                                         NaN
beds_na                                             NaN
Name: label_price, dtype: float64
```

Task: Use Pandas indexing to extract the column names for the top two correlation values and save the results to the Python list `top_two_corr`. Add the feature names to the list in the order in which they appear in the output above.

Note: Do not count the correlation of `label` column with itself, nor the `price` column -- which is the `label` column prior to outlier removal.

[80]:
```python
# Drop 'label_price' and 'price' from the sorted correlation Series
top_two_corr = corrs_sorted.drop(['label_price', 'price']).head(2).index.
    ↪tolist()
top_two_corr
```

[80]:
```
['accommodates', 'bedrooms']
```

Bivariate Plotting: Produce Plots for the Label and Its Top Correlates Let us visualize our data.

We will use the `pairplot()` function in `seaborn` to plot the relationships between the two features and the label.

Task: Create a DataFrame `df_corrs` that contains only three columns from DataFrame `df`: the label, and the two columns which correlate with it the most.

[81]:
```python
# Create a DataFrame with the label and its top two correlates
df_corrs = df[['label_price', 'accommodates', 'bedrooms']]
df_corrs
```

[81]:

	label_price	accommodates	bedrooms
0	150.0	1	1.323567
1	75.0	3	1.000000
2	60.0	2	1.000000
3	275.0	4	2.000000
4	68.0	2	1.000000
…	…	…	…
38272	79.0	2	1.000000

```
38273         76.0              2  1.000000
38274        116.0              2  1.000000
38275        106.0              2  1.000000
38276        689.0             14  6.000000

[38277 rows x 3 columns]
```

Task: Create a `seaborn` pairplot of the data subset you just created. Specify the *kernel density estimator* as the kind of the plot, and make sure that you don't plot redundant plots.

Note: It will take a few minutes to run and produce a plot.

[ ]:
```python
import seaborn as sns
import matplotlib.pyplot as plt

# Sample or dropna if needed for speed
sns.pairplot(df_corrs.dropna(), kind='kde', corner=True)
plt.show()
```

1.5 Part 5: Analysis

1. Think about the possible interpretation of the plot. Recall that the label is the listing price. How would you explain the relationship between the label and the two features? Is there a slight tilt to the points cluster, as the price goes up?
2. Are the top two correlated features strongly or weakly correlated with the label? Are they features that should be used for our predictive machine learning problem?
3. Inspect your data matrix. It has a few features that contain unstructured text, meaning text data that is neither numerical nor categorical. List some features that contain unstructured text that you think are valuable for our predictive machine learning problem. Are there other remaining features that you think need to be prepared for the modeling phase? Do you have any suggestions on how to prepare these features?

Record your findings in the cell below.

[ ]:
```
1. How would you explain the relationship between the label (label_price) and
 ↪the two most correlated features (accommodates and bedrooms)? Is there a
 ↪slight tilt in the cluster of points as the price increases?

Yes, there is a positive upward trend in the scatter plots, which shows that as
 ↪accommodates (number of guests the listing can host) and bedrooms increase,
 ↪the price also tends to increase. This is expected, as larger listings
 ↪typically have higher prices.

2. Are the top two correlated features strongly or weakly correlated with the
 ↪label? Should they be used in our predictive machine learning model?
-accommodates has a moderately strong correlation with label_price.
-bedrooms has a moderate correlation.
```

```
Both features should definitely be included in the machine learning model, as
↪they directly influence listing value.

3. List some features that contain unstructured text which might be valuable
↪for our predictive machine learning model.
The following columns contain free-form text that could provide useful insights:
-name: may include keywords like "luxury", "studio", etc.
-description: typically describes the property’s features.
-neighborhood_overview: gives context about the neighborhood.
-host_about: provides information about the host.
-amenities: although long, it often includes structured items like "Wifi",
↪"TV", "Kitchen", etc.

To make them usable, we can apply Natural Language Processing (NLP) techniques
↪such as:

-Text cleaning (lowercase, punctuation removal).
-Tokenization and lemmatization.
-Vectorization (e.g., TF-IDF or Bag-of-Words).
-For amenities, split by commas and use multi-hot encoding.

4. Are there other remaining features that need to be prepared before modeling?
↪Any suggestions on how to prepare them?

Yes, the following features should be prepared:

-bathrooms_text: values like "1.5 baths" or "Shared bath" need to be converted
↪into numeric features, possibly using regex or manual mapping.
-Date columns such as first_review, last_review, host_since should be
↪transformed into: Listing age (e.g., how many days since first review) and
↪time since last review.
-Binary variables like host_is_superhost, host_has_profile_pic, etc., should be
↪converted from 't'/'f' into True/False or 0/1.
```

```
[ ]:
```